Design Manager/ Flow Engine Guide

About This Manual

This manual describes the Xilinx Design Manager and Flow Engine, tools that manage and process your design implementation. This manual describes the use of these tools with the Xilinx Alliance Series software. Because the Design Manager and Flow Engine are closely integrated in the Alliance Series, this manual covers both software applications in detail and also covers how to access other tools from the Design Manager.

In the Xilinx Foundation Series software, the Project Manager, rather than the Design Manager, is the top level software tool that is closely integrated with the Flow Engine. For information on the Foundation Series Flow Engine, refer to this manual. For information on the Project Manager, see the *Foundation Series User Guide*.

The Design Manager software documented in this manual is available as a standalone tool from the Foundation Series software. Use the **`Xilinx Foundation Series`** → **`Accessories`** → **`Design Manager`** command to access the standalone Design Manager.

Note This Xilinx software release is certified as Year 2000 compliant.

Manual Contents

This manual covers the following topics.

- Chapter 1, "Introduction," describes the Design Manager and Flow Engine main functions, their place in the Xilinx design flow, key features, inputs and outputs, and supported architectures. It also outlines the basic procedure for using the tools.

- Chapter 2, "Getting Started," describes how to start and exit the Design Manager and Flow Engine; how to use the menus, icons, and dialog boxes; and how to use online help.
- Chapter 3, "Using the Design Manager and Flow Engine," explains how to perform Design Manager and Flow Engine functions.
- "Glossary," defines important words, terms, concepts, and ideas used in this manual.

Additional Resources

For additional information, go to `http://support.xilinx.com`. The following table lists some of the resources you can access from this Web site. You can also directly access these resources using the provided URLs.

Resource	Description/URL
Tutorials	Tutorials covering Xilinx design flows, from design entry to verification and debugging `http://support.xilinx.com/support/techsup/tutorials/index.htm`
Answers Database	Current listing of solution records for the Xilinx software tools Search this database using the search function at `http://support.xilinx.com/support/searchtd.htm`
Application Notes	Descriptions of device-specific design techniques and approaches `http://support.xilinx.com/apps/appsweb.htm`
Data Book	Pages from *The Programmable Logic Data Book*, which contain device-specific information on Xilinx device characteristics, including readback, boundary scan, configuration, length count, and debugging `http://support.xilinx.com/partinfo/databook.htm`
Xcell Journals	Quarterly journals for Xilinx programmable logic users `http://support.xilinx.com/xcell/xcell.htm`
Technical Tips	Latest news, design tips, and patch information for the Xilinx design environment `http://support.xilinx.com/support/techsup/journals/index.htm`

Conventions

This manual uses the following conventions. An example illustrates each convention.

Typographical

The following conventions are used for all documents.

- `Courier font` indicates messages, prompts, and program files that the system displays.

  ```
  speed grade: - 100
  ```

- **`Courier bold`** indicates literal commands that you enter in a syntactical statement. However, braces "{ }" in Courier bold are not literal and square brackets "[ ]" in Courier bold are literal only in the case of bus specifications, such as bus [7:0].

  ```
  rpt_del_net=
  ```

  `Courier bold` also indicates commands that you select from a menu.

  `File` → **`Open`**

- *Italic font* denotes the following items.

  - Variables in a syntax statement for which you must supply values

    `edif2ngd` *`design_name`*

  - References to other manuals

See the *Development System Reference Guide* for more information.

- Emphasis in text

  If a wire is drawn so that it overlaps the pin of a symbol, the two nets are *not* connected.

- Square brackets "[ ]" indicate an optional entry or parameter. However, in bus specifications, such as bus [7:0], they are required.

  edif2ngd *[option_name] design_name*

- Braces "{ }" enclose a list of items from which you must choose one or more.

  lowpwr ={**on**|**off**}

- A vertical bar " | " separates items in a list of choices.

  lowpwr ={**on**|**off**}

- A vertical ellipsis indicates repetitive material that has been omitted.

  ```
  IOB #1: Name = QOUT'
  IOB #2: Name = CLKIN'
  .
  .
  .
  ```

- A horizontal ellipsis "...." indicates that an item can be repeated one or more times.

  `allow block` *block_name loc1 loc2locn;*

Online Document

The following conventions are used for online documents.

- Red-underlined text indicates an interbook link, which is a cross-reference to another book. Click the red-underlined text to open the specified cross-reference.

- Blue-underlined text indicates an intrabook link, which is a cross-reference within a book. Click the blue-underlined text to open the specified cross-reference.

Contents

Chapter 3 Using the Design Manager and Flow Engine

Chapter 1

Introduction

This chapter briefly describes the Design Manager and Flow Engine. It describes their main functions, their place in the design flow, major features, inputs and outputs, and supported device architectures. It also outlines the basic procedure for using these tools. This chapter contains the following sections.

Overview

The Design Manager is the top level software module in the Xilinx Alliance Series Development System. Use the Xilinx Development System tool suite to implement a design into a Xilinx device. The Design Manager provides access to all the tools you need to read a design file from a design entry tool and implement it in a Xilinx device. The Design Manager performs the following functions.

- Organizes and manages your design implementation data
- Creates multiple design versions for management of design changes
- Creates multiple implementation revisions for management of implementation strategies

- Provides access to reports
- Manages data for and provides access to the following tools. The tools differ for Field Programmable Gate Array (FPGA) and Complex Programmable Logic Device (CPLD) families.
  - Flow Engine (FPGA and CPLD)
  - Timing Analyzer (FPGA and CPLD)
  - Floorplanner (All XC4000 families and Virtex, Virtex-E, Virtex-II, Spartan, SpartanXL, and Spartan-II FPGAs only)
  - PROM File Formatter (FPGA)
  - Hardware Debugger (FPGA)
  - FPGA Editor (FPGA)
  - ChipViewer (CPLD)
  - JTAG Programmer (All CPLD families, all XC4000 families, and XC5200, Virtex, Virtex-E, Virtex-II, Spartan, SpartanXL, and Spartan-II FPGAs only)

The Design Manager manages your Xilinx designs. The Flow Engine implements your designs. The Flow Engine is closely integrated with the Design Manager, sharing many of the same menus and dialog boxes. You can use the Design Manager and Flow Engine together to perform the following functions.

- Target different devices
- Generate and export timing simulation data for external simulation tools
- Generate and export configuration data

See the "Implementation Tools Tutorial" chapter of the *Alliance Series 3.1i Quick Start Guide* or the "Basic Tutorial" chapter of the *Foundation Series 3.1i Quick Start Guide* for information on running through a typical implementation flow with Xilinx tools.

Design Flow

You can use a variety of schematic and HDL tools for design entry. The netlist formats supported as inputs to the Design Manager are listed in the "Inputs and Outputs" section.

After design entry, you can use the Design Manager and Flow Engine to process your design in the following basic steps.

1. Implementation of your design for a specific target device
2. Report generation showing the status of your design
3. Timing analysis for design verification
4. Export of your design for timing simulation and programming

The following figures illustrate the processing steps through the Design Manager and Flow Engine for FPGAs and CPLDs.

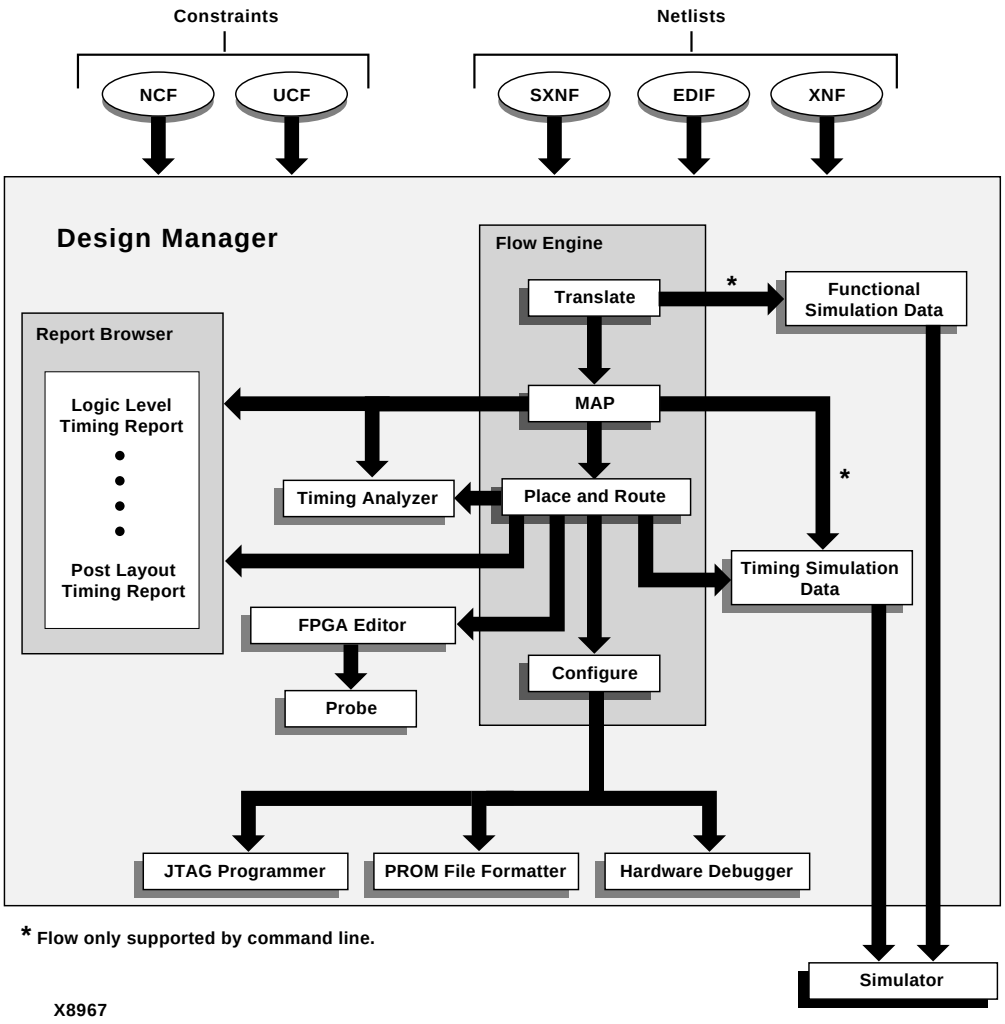


Figure 1-1 Design Manager/Flow Engine Design Flow (FPGA)

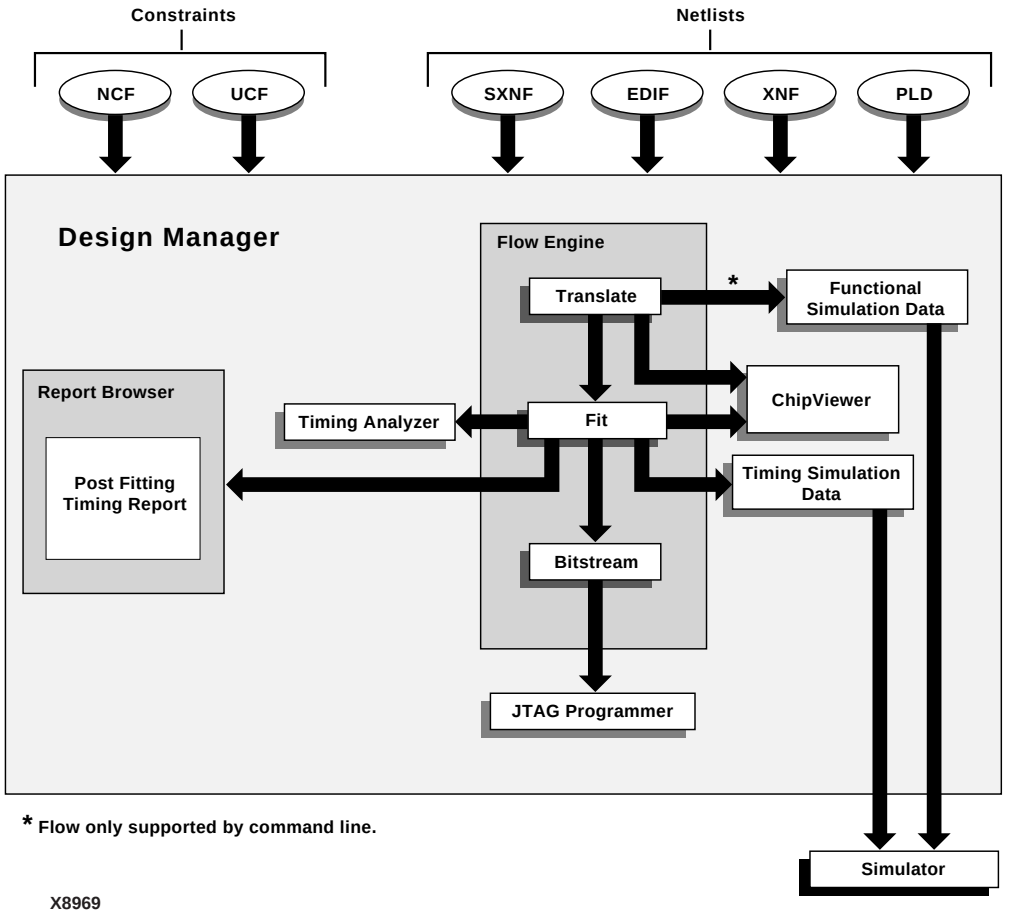


Figure 1-2 Design Manager/Flow Engine Design Flow (CPLD)

Inputs and Outputs

The Design Manager accepts the following file types as inputs.

.edf, .edn, .edif, .sedif	EDIF netlist file
.pld	PLUSASM file (CPLD only)
.xnf, .xtf, .sxnf	Xilinx hierarchical netlist file

The Design Manager and Flow Engine output report, timing simulation, and programming files.

Architecture Support

The Design Manager and Flow Engine support the following architecture families in this release. The XC9500 families compose the CPLD families. All other device families compose the FPGA families.

- Spartan™ / XL / -II
- Virtex™ / -E / -II
- XC9500™ / XL / XV
- XC4000™ E / L / EX / XL / XV / XLA
- XC3000™ A / L
- XC3100™ A / L
- XC5200™

Design Manager Fundamentals

You can perform multiple functions from within the Design Manager, such as launching Xilinx tools and managing your projects, design versions, and implementation revisions.

Projects

The Design Manager window displays all Xilinx data related to a single project. When you create a project, a design version and implementation revision are automatically created so you always have a revision on which to run the software tools. A project includes all design versions and implementation revisions that are created as you implement your design. You can work with multiple projects, but only one is active at a time. The hierarchical structure in the Design Manager project view shows the relationships of the data elements to each other.

Design Versions

You create your design using third party front-end tools that support schematic and Hardware Description Language (HDL) entry. The Design Manager reads in the design netlist and creates a design

version in the Design Manager project view. If the design netlist changes, the Design Manager reads in the modified design and creates a new design version.

You can try modified versions of your design and easily keep track of them. Each new set of changes becomes a new design version and is assigned a design version name by the Design Manager. You can choose any one of the available design versions for processing.

When you create a new design version, a new implementation revision is automatically created. Each design version can contain multiple implementation revisions.

Note The Design Manager saves design versions in the Xilinx database format, not in the format of the front-end editor.

Implementation Revisions

After you create a design version, you can try different implementation strategies on your design. The data associated with each of these implementation strategies is called an implementation revision. Each implementation revision contains the data files and reports that are created based on a specific set of implementation strategies.

This method allows you to vary how your design is implemented in order to achieve your design objectives. For example, you can maximize speed and density in your design by controlling the implementation flow settings or by targeting a different device family better suited for your design.

Note If you are using an HDL flow, it is strongly recommended that you create a new synthesis netlist when you change FPGA or CPLD families. This ensures that the tools make the best use of the target device.

When you create a new revision either manually or automatically, the data from the “last” revision is copied to the new revision by default. The “last” revision is bottommost in the Design Manager project view. You can copy data from other revisions by setting options in the Copy Persistent Data field of the New Version or New Revision dialog box. You can also copy data from other revisions by using the Set Constraint File, Set Guide File(s), and Set Floorplan File(s) menu commands after the revision is created. If you want to change any of the implementation flow options, use the Options command.

Because you must always have an implementation revision on which to run the tools, at least one revision must always exist for each version in the Design Manager. If only one revision exists inside a version, you can only delete the revision by deleting the version.

Design Management Basic Procedure

The typical procedure for managing a design is as follows.

1. In your design entry tool, create a design.
2. In your design entry tool, output your design as an EDF, EDIF, EDN, SEDIF, PLD, SXNF, XNF, or XTF file.

   Note The PLD format is supported for CPLDs only.
3. In the Design Manager, open an existing project or create a new project in which to implement your logic design.
4. Implement your design.
   - If necessary, choose a different target device in which to implement your design. The initial target device is specified in the input design or when you create a new implementation revision.
   - Select implementation, simulation, and configuration options. You can use the default settings, an existing set of options, or set new options.
   - Run your design implementation and create timing simulation files and a device programming file.
5. Review your design reports to verify that your design fits within the target device and that your timing requirements are met.
6. If your design requirements are not met, you can do any of the following and process your design again.
   - Change your logic design.
   - Choose a different target device, package, or speed grade.
   - Define a different set of implementation, simulation, or configuration options.

Flow Engine Fundamentals

The Flow Engine allows you to easily process and control the implementation of your design.

Design Implementation

When you process your design, the Flow Engine translates the design file into the Xilinx Native Generic Database (NGD) format. The Flow Engine then implements your design and generates bitstream data.

The Flow Engine allows you to control the implementation of your design in different ways. For example, if you are new to Xilinx software or want to quickly check your design, you can run the *automatic* Flow Engine using the Design Manager's Design → Implement command. However, if you are an experienced user, you can open the Flow Engine from the Design Manager Tools menu and use the *interactive* Flow Engine to execute steps separately.

You can also fine tune your design by modifying your implementation, simulation, and configuration options. You can access these options through the Design Manager's Design → Options command or the Flow Engine's Setup → Options command. Use the Design Manager Options command with the automatic Flow Engine and the Flow Engine Options command with the interactive Flow Engine.

Note For more information on setting implementation, simulation, and configuration options, see the "Specifying Implementation Flow Options" section of the "Using the Design Manager and Flow Engine" chapter.

If you use the interactive Flow Engine, you can control how far to process your design using the Flow Engine's Setup → Stop After command. In the Stop After dialog box, select a step as your target break point. For example, if you want to map, place, and route your design but not create device programming file, select Stop After Place&Route. A stop sign indicates where the process flow will stop.

The following sections describe the steps run by the Flow Engine. The following figures shows these steps in detail for FPGAs and CPLDs.

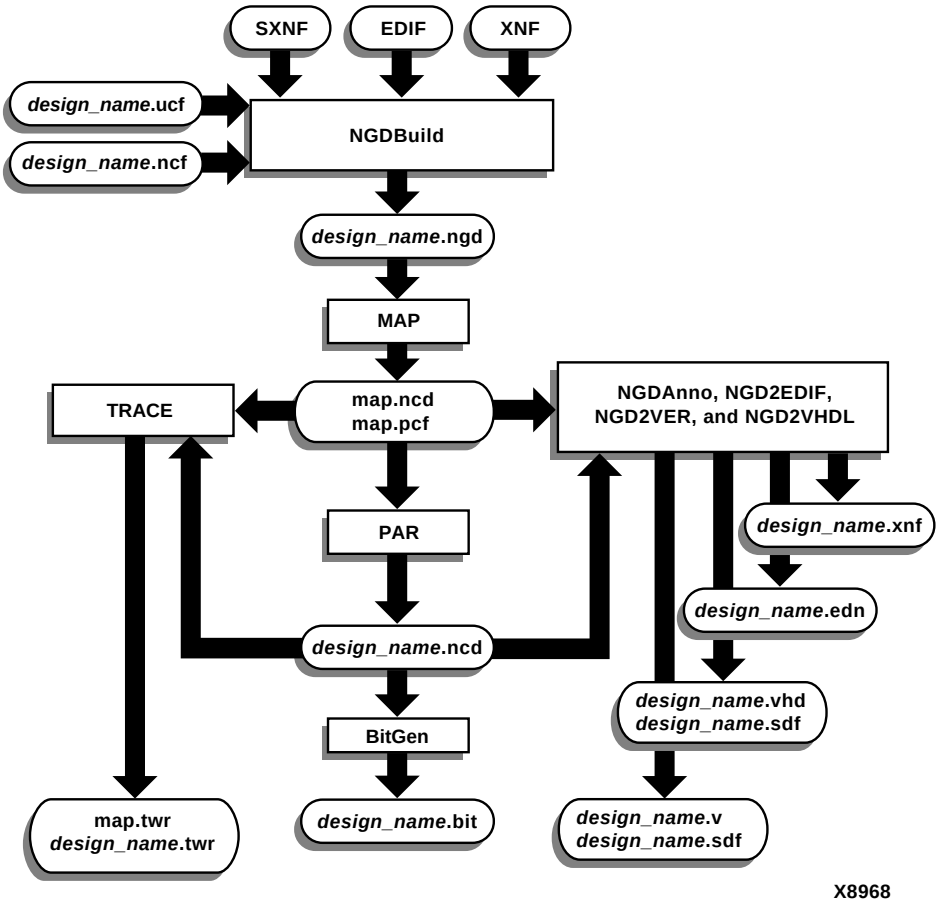


Figure 1-3 Flow Engine Steps (FPGA)

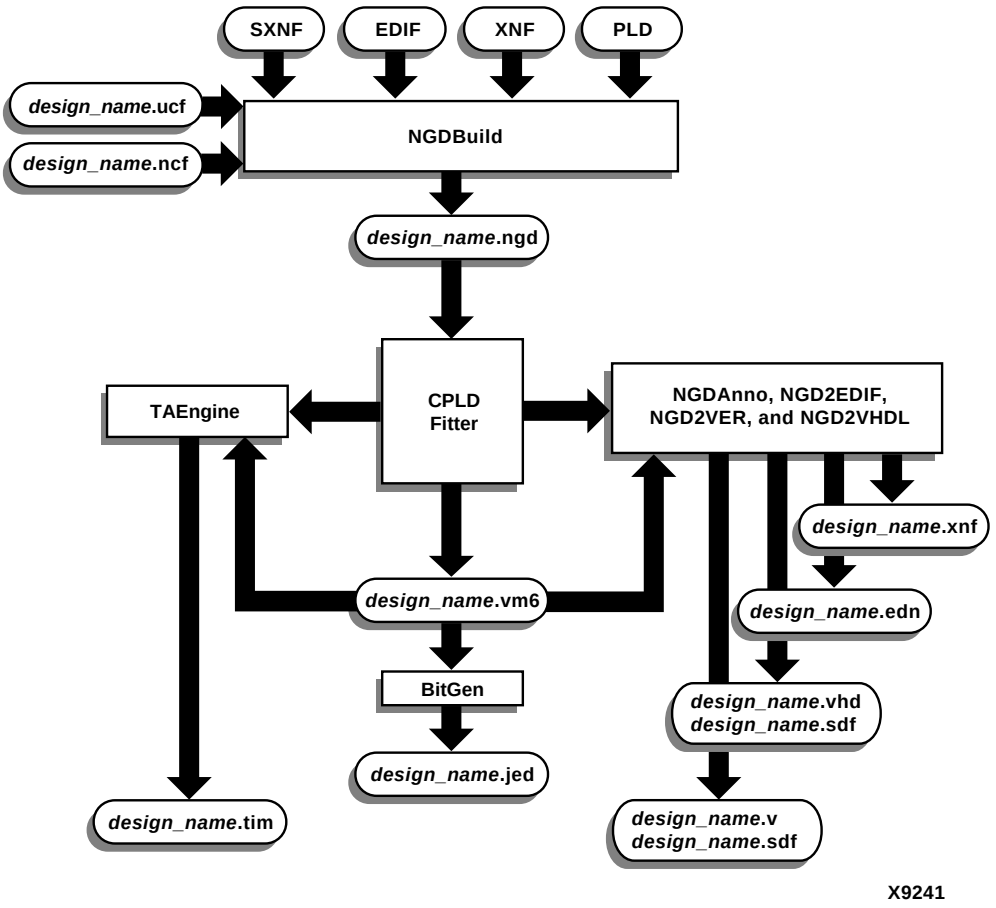


Figure 1-4 Flow Engine Steps (CPLD)

Translate

During this step, the Flow Engine merges all of the input netlists. You can control aspects of the Translate step by setting implementation options using the Options command from the Design Manager or Flow Engine. The Flow Engine accomplishes this step by running NGDBuild, which is described in the "NGDBuild" chapter of the *Development System Reference Guide*.

The Translation Report is generated during this step. It contains warning and error messages from the three translation processes: conversion of the EDIF or XNF style netlist to the Xilinx NGD netlist, timing specification checks, and logical design rule checks. The report lists the following.

- Hierarchical blocks that are missing or cannot be translated
- Invalid or incomplete timing constraints
- Output contention, loadless outputs, and sourceless inputs

Map (FPGA)

During this step, the Flow Engine maps a logical design to a Xilinx FPGA. The input is an NGD file, which contains a logical description of the design and macro library (NMC) files. The Flow Engine first performs a logical Design Rule Check (DRC) on the design in the NGD file. It then maps the logic to the components in the target Xilinx FPGA. The output design is a Native Circuit Description (NCD) file that physically represents the design mapped to the components in the Xilinx FPGA. You can control aspects of the Map step by setting implementation options using the Options command from the Design Manager or Flow Engine. The Flow Engine accomplishes the Map step by running the MAP program, which is described in the "MAP—The Technology Mapper" chapter of the *Development System Reference Guide*.

The Map Report is generated during this step. It contains warning and error messages detailing logic optimization and logic mapping to physical resources. The report lists the following information.

- Removed logic
- Added or expanded logic due to speed optimization
- Number and percentage of used CLBs, IOBs, flip-flops, and latches
- Use of architecture-specific resources such as global buffers and boundary scan logic

The Logic Level Timing Report can also be generated during this step. This report provides a summary analysis of your timing constraints based on block delays and absolute minimum route delays. You can use this report to determine whether you need to revise the timing constraints in your design or create more efficient

logic. This feature provides a useful analysis of your timing constraints without the wait required for place and route. To obtain a detailed analysis, use the Timing Analyzer tool after you place and route your design.

Place&Route (FPGA)

During this step, the Flow Engine takes the NCD file produced during Map and places and routes the design to produce a routed NCD file. The output NCD file can also act as a guide file if you place and route the design again. You can control aspects of the Place and Route step by setting implementation options using the Options command from the Design Manager or Flow Engine. To accomplish this step, the Flow Engine runs the PAR program, which is described in the "PAR—Place and Route" chapter of the *Development System Reference Guide*.

The Place and Route Report is generated during this step. It contains the following information.

- Design score
- Number of signals not completely routed
- Timing performance

The Pad Report is generated during this step. It lists your design's pinout sorted by signal name, and then by pin number.

By default, the Post Layout Timing Report is also generated during this step. Depending on the report settings chosen, this report provides an analysis of the maximum clock speed and constraint information for a placed and routed design. To obtain a detailed analysis, use the Timing Analyzer tool.

The Asynchronous Delay Report is also generated during this step. This report lists all nets in the design and the delays of all loads on the net.

Fit (CPLD)

During Fit, the Flow Engine launches the CPLD Fitter to minimize and collapse the combinatorial logic of your design so that it requires the least number of macrocell and product term resources. It also partitions and maps your design to fit within the architecture of the CPLD. You can control aspects of this step by setting implementation

options using the Options command from the Design Manager or Flow Engine.

A Post Fitting Timing Report can also be generated during this step. The timing report provides a brief analysis of the maximum clock speed for the design after it is fitted. To obtain a detailed analysis, use the Timing Analyzer tool.

Timing (Sim)

The Flow Engine runs this step to produce timing simulation data. The data that is produced depends on the simulation options you set using the Options command from the Design Manager or Flow Engine. To accomplish this step, the Flow Engine runs NGDAnno and one of the following tools, all of which are described in the *Development System Reference Guide*.

- NGD2EDIF
- NGD2VER
- NGD2VHDL

Configure (FPGA)

After the design has been completely routed, the Flow Engine configures the device so that it can execute the desired function. Using a fully routed NCD file as input, it produces a configuration bitstream, a binary file with a .bit extension. The BIT file contains all of the configuration information from the NCD file defining the internal logic and interconnections of the FPGA, plus device-specific information from other files associated with the target device. The binary data in the BIT file can then be downloaded into the FPGA's memory cells, or it can be used to create a PROM file. You can control aspects of this step by setting configuration options using the Options command from the Design Manager or Flow Engine. The Flow Engine accomplishes this step by running BitGen, which is described in the "BitGen" chapter of the *Development System Reference Guide*.

Bitstream (CPLD)

During this step, the Flow Engine produces a JED programming file. The JTAG Programmer uses this file to configure CPLD devices. You can control aspects of this step by setting implementation options

using the Options command from the Design Manager or Flow Engine.

Smart Flow Engine

When the Flow Engine is first invoked, it automatically looks for changes made to certain files. If the Flow Engine detects changes, it restarts the flow for the implementation revision as follows. This feature is called the Smart Flow Engine.

- If a previously set target break point has already been reached at the time you launch the Flow Engine, the Smart Flow Engine removes the break point and implements the next process in the flow.
- If any of the following changes are made, the Smart Flow Engine determines which process to rerun.
  - Changes to implementation flow options from the Options dialog box or Template Manager
  - Changes to design, constraints, guide, or output files

    The following table shows which flow process the Flow Engine reruns if you make changes to a particular file.

Table 1-1 Smart Flow Engine Change Detection

File Changed	Process Rerun
design_name.ucf (or custom constraints file) *design_name*.ncf	Translate
design_name.ngd *design_name*.mfp (or custom floorplan file)	Map (FPGA)
map.ncd guide.ncd (or custom guide file)	Place&Route (FPGA)
design_name.ngd *design_name*.gyd	Fit (CPLD)
design_name.ncd	Timing (Sim) (FPGA)
design_name.vm6	Timing (Sim) (CPLD)

Table 1-1 Smart Flow Engine Change Detection

File Changed	Process Rerun
design_name.ncd	Configure (FPGA)
design_name.vm6	Bitstream (CPLD)

The Smart Flow Engine notifies the Design Manager of the state and status of your implementation revision and the Design Manager updates the information in the main window.

The Flow Engine checks for changes related to successfully completed processes. If a process did not complete successfully, the Flow Engine does not look for changes related to this process but automatically restarts the flow at this process or, if appropriate, at a previous process. If the Flow Engine is already open and you want to check for changes, use the **Setup** → **Update Flow** command.

Chapter 2

Getting Started

This chapter leads you through the basic operation of the Design Manager and Flow Engine. This chapter contains the following sections.

Preparing the Input Design File

Create the input design in your design entry tool and save it in one of acceptable formats described in the "Inputs and Outputs" section of the "Introduction" chapter.

Starting and Exiting the Design Manager

The Design Manager runs on PCs and workstations. You can start the Design Manager as a standalone tool on the PC or from the command line. Use the following procedures to start and exit the Design Manager.

Starting as a Standalone Tool

If you installed the Design Manager as a standalone tool on a PC, click the Design Manager icon (shown in the following figure) on the Windows desktop or select dsgnmgr.exe from the Windows 98®, Windows 2000®, or Windows NT® Start button.

Starting from the Command Line

To start the Design Manager from the UNIX® or DOS™ command prompt, enter one of the following commands.

- `dsgnmgr`
- `xilinx`

Exiting the Design Manager

To exit the Design Manager, select the **`File`** → **`Exit`** menu command. A confirmation box appears. Click **`Yes`** to exit the Design Manager.

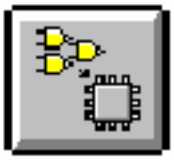
Starting and Exiting the Flow Engine

The Flow Engine runs on PCs and workstations. Use the following procedures to start and exit the Flow Engine.

Starting from the Alliance Series

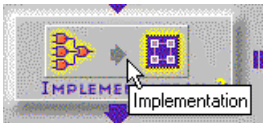
To launch the Flow Engine from the Alliance Design Manager, select an implementation revision and do one of the following.

- Select **Tools** → **Flow Engine**.
- Click the Flow Engine toolbox button, shown in the following figure.



Starting from the Foundation Series

To launch the Flow Engine from the Foundation Project Manager, select the Implementation phase button on the project flow chart, shown in the following figure.



Note For more information about accessing and using the Flow Engine from the Foundation Project Manager, see the "Design Implementation" chapter of the *Foundation Series User Guide*.

Exiting the Flow Engine

To exit the Flow Engine, select the **Flow** → **Close** command.

Using the Interface

This section describes the elements that compose the Design Manager and Flow Engine interfaces and how to use them.

Main Window

This section describes the Design Manager and Flow Engine main windows, their menus, toolbar, and status bar. By default, the main

window displays a menu bar and toolbar at the top and status bar at the bottom of the window. You can hide the toolbar or status bar from view by selecting the Toolbar or the Status Bar commands, respectively, from the View menu. In the Design Manager, the toolbox also appears by default. You can hide it from view by selecting the Toolbox command from the View menu.

Design Manager Window

To work with the Design Manager, you must create a project. See the "Creating a New Project" section of the "Using the Design Manager and Flow Engine" chapter for information. After you create a project, the Design Manager window appears configured for the loaded design, as shown in the following figure.

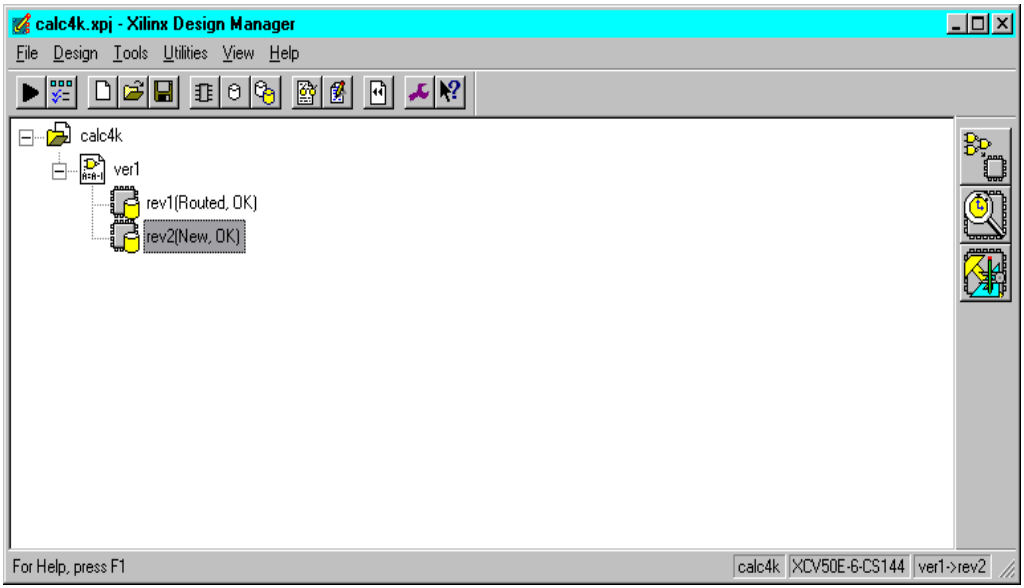


Figure 2-1 Design Manager Window

The Design Manager project view displays design version and implementation revision icons. The toolbox allows you to launch Xilinx tools. You can select Design Manager commands from the menus and toolbar.

Flow Engine Window

You can open the Flow Engine automatically through the Design Manager implementation process or manually from the Design Manager Tools menu or toolbox. See the "Implementing a Design from the Design Manager" or "Implementing a Design from the Flow Engine" section of the "Using the Design Manager and Flow Engine" chapter for descriptions of the different ways to open the Flow Engine and implement a design. When you open the Flow Engine, the Flow Engine window appears as shown in the following figures.

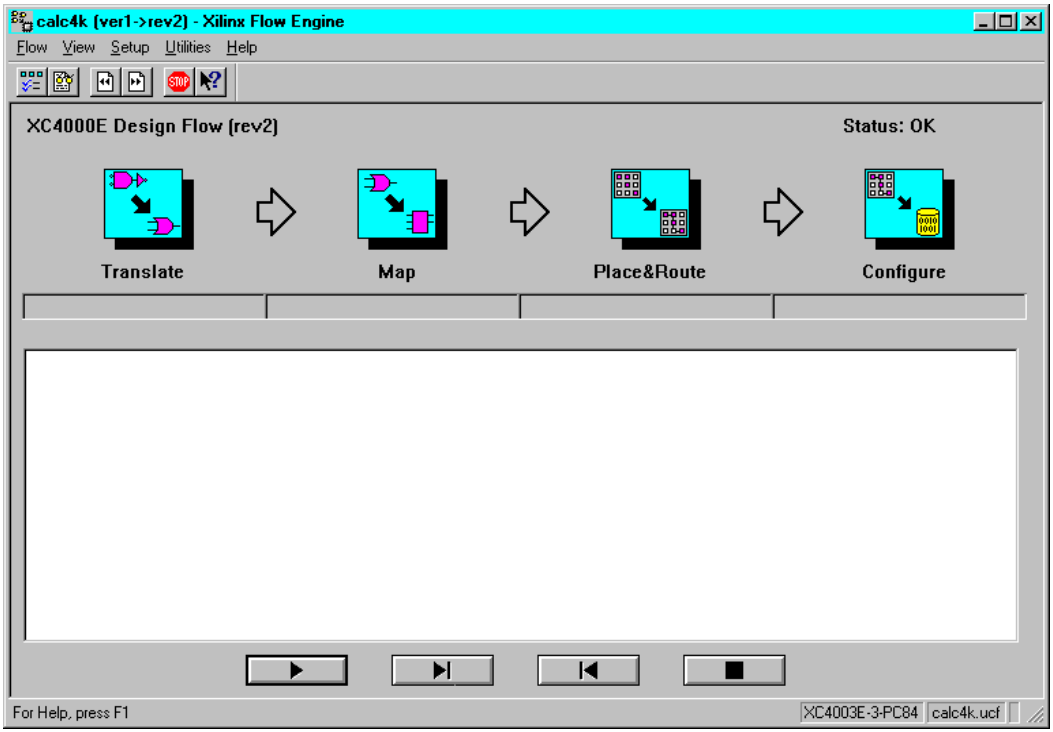


Figure 2-2 Flow Engine Window (FPGA)

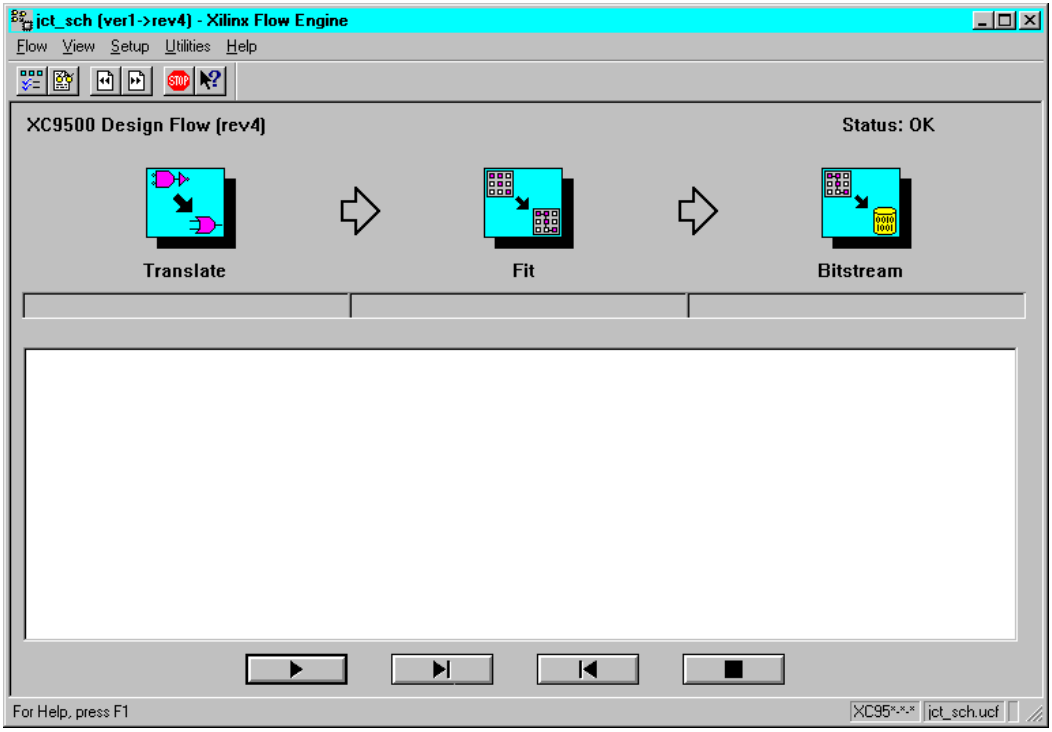


Figure 2-3 Flow Engine Window (CPLD)

Process indicators in the Flow Engine main window show which step in the design flow is currently processing. The arrows between each step turn black after the previous step is completed. Below each process indicator, a progress field shows the status of each processing step, whether Running, Completed, Aborted, or Failed. During the Place&Route process, the progress field shows additional information about the processing step, whether Running, Placing, Routing, Improving, or Reporting.

Note For small designs, Place&Route may complete before all of the additional processing step information can be displayed. Reporting is only shown if you selected the Post Layout Timing Report setting in the Timing Report tab of the Implementation Options dialog box. See the online help for more information on this setting.

You can control each step in the processing of your design by using the run control buttons at the bottom of the Flow Engine main window. Select Flow Engine commands from the menus and toolbar.

Title Bar

The title bar displays the program name and the name of the currently loaded design.

Menu Bar

All of the Design Manager commands are available in the pull-down menus of the Design Manager window after a design is loaded. Flow Engine commands are available in the pull-down menus of the Flow Engine window.

Toolbar Buttons

You can access frequently used commands by clicking the appropriate toolbar button. See the online help for an explanation of each toolbar button.

Toolbox Buttons

You can access the available set of Xilinx interactive tools from the Design Manager Tools menu or from the toolbox. To launch a tool from the toolbox, click the appropriate toolbox button. See the online help for a detailed explanation of the toolbox.

Status Bar

By default, the status bar appears at the bottom of the main window. When you select a menu command, a brief description of the command's function appears in the status bar. As the software processes, status messages are dynamically updated and displayed.

Dialog Boxes

Many menu commands display dialog boxes in which you can enter information and set options.

Common Fields

The fields shown in the following table are common to most dialog boxes.

Table 2-1 Common Dialog Box Fields

Dialog Box Field	Function
OK	Closes the dialog box and implements the intended action according to the settings in the dialog box
Cancel	Closes the dialog box without effecting any action
Help	Displays information on the dialog box

Using Help

The Design Manager and Flow Engine contain context-sensitive help and a Help menu. You can obtain help on commands and procedures with the Help menu or by selecting the Help toolbar button. In addition, the dialog boxes associated with many commands have a Help button that you can click to obtain context-sensitive help.

Note The online help includes information on menu commands, dialog boxes, and implementation options.

Help Menu

Use the following Help menu commands to obtain help.

- The Help Topics command opens Help and lists the online help topics available. From the Contents page, you can jump to command information or step-by-step instructions. After you open help, you can click the Help Topics button in the Help window whenever you want to return to the help topic list.
- The Online Documentation command provides access to the Software Manuals Online.
- The Xilinx on the Web command provides access to the support.xilinx.com page and the Xilinx home page on the Web.
- The About Design Manager command opens a popup window that displays the registration and version number of the Design Manager software and a copyright notice.

- The About Flow Engine command opens a popup window that displays the registration and version number of the Flow Engine software and a copyright notice.

Toolbar Help Button

You can obtain context-sensitive help from the toolbar as follows.

1. Click the Help button in the toolbar.

   The cursor changes to a question mark.

2. With the left mouse button, click the menu item or toolbar button for which you want help.

   Help appears for the selected command or option.

Note You can also press **`Shift F1`** to obtain context-sensitive help.

F1 Key

Pressing the F1 key on a dialog box displays help on that dialog box. Pressing the F1 key is the same as selecting Help → Help Topics, if no dialog boxes are displayed.

Help Button in Dialog Boxes

Many of the dialog boxes have a Help button that you can click to obtain help for the dialog box with which you are working. You can also press **`Alt H`** on your keyboard while positioned over the dialog box to obtain help.

Chapter 3

Using the Design Manager and Flow Engine

This chapter shows you how to perform common design tasks in the Design Manager and Flow Engine. These procedures are described in the following sections.

- "Creating a New Project"
- "Creating a New Design Version"
- "Creating a New Implementation Revision"
- "Setting a Part"
- "Deleting Items from the Project View"
- "Specifying Implementation Flow Options"
- "Implementing a Design from the Design Manager"
- "Copying Constraints, Guide, and Floorplan File Data"
- "Viewing Reports"
- "Producing Timing Reports"
- "Generating Pin Locking Constraints"
- "Producing Timing Simulation Data"
- "Exporting Design Data"
- "Archiving a Project"

Following are advanced procedures you can use after you are comfortable with the preceding basic procedures. The advanced procedures may help improve your run time and design performance.

- "Implementing a Design from the Flow Engine"
- "Placing and Routing Non-Timing Driven Designs"

- "Running Multiple Place and Route Passes"
- "Running Re-Entrant Routing on FPGAs"
- "Working with Templates"

Note Most commands run from the Design Manager require that an implementation revision be selected.

Creating a New Project

When you open the Design Manager for the first time, you must create a new project for your design before you can use the Design Manager. A project includes all design versions, implementation revisions, reports, and any other Xilinx data created while you work with a design.

The Design Manager graphically displays information about these items in the project view. When you create a new project, you specify a design to open and a directory for the project. You can create as many projects as you want, but you can only work with one at a time. The following procedure explains how to create a new project by importing a design.

1. In the Design Manager, select **`File`** → **`New Project`** or click the New Project toolbar button.

The dialog box shown in the following figure appears.

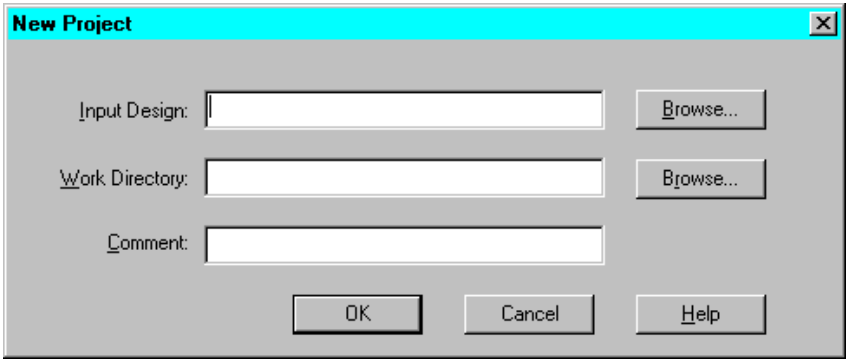


Figure 3-1 New Project Dialog Box

2. In the Input Design field, specify a design file to open, or click the **Browse** button to select a file from the Open dialog box. Use only the file formats listed in the "Inputs and Outputs" section of the "Introduction" chapter.

3. To change the default work directory, type a path in the Work Directory field or use **Browse** to select a directory.

   Note The Design Manager automatically creates a subdirectory named xproj under the input design directory and uses it as the work directory. The Design Manager uses the xproj subdirectory to store all the data files for the project.

4. Enter any comments in the Comment field. Use this field to note options and strategies.

5. In the New Project dialog box, click **OK**.

   After your design is loaded, the Design Manager is configured for the loaded design.

Creating a New Design Version

While working on a project, you may need to modify the initial input design and bring these changes into an existing project in the Design Manager. You can do this with the New Version command. The Design Manager automatically assigns a name for the design version, but you can enter a different name in the New Version dialog box. A

new implementation revision is automatically created when you create a new version.

Note When implementing your design, the New Version dialog box appears automatically if you made changes to your design netlist.

1. In the Design Manager, select **Design** → **New Version**.

   The dialog box shown in the following figure appears.

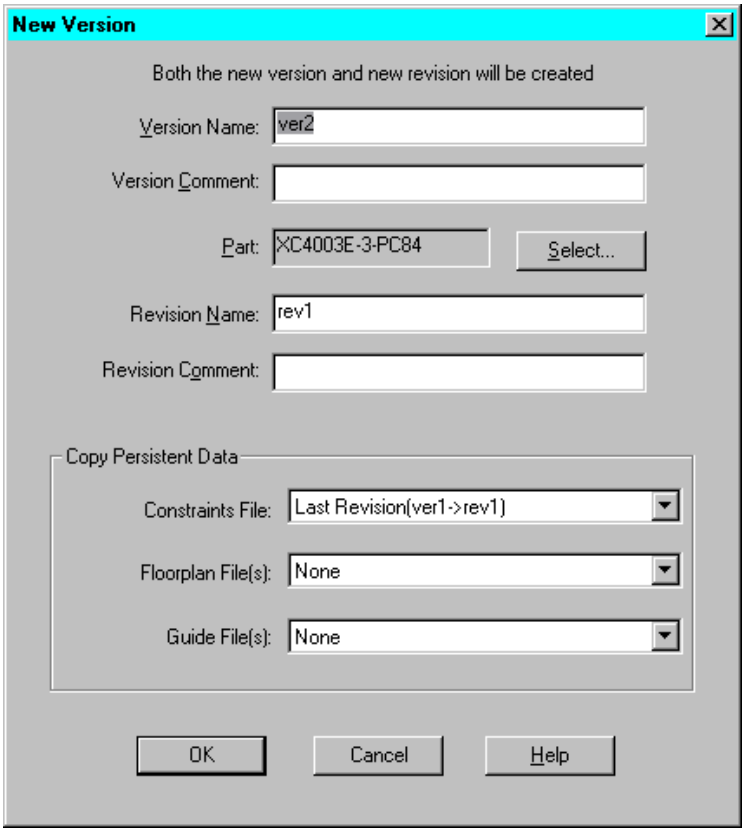


Figure 3-2 New Version Dialog Box

2. Enter a name in the Version Name field if you do not want to use the default name. Use the characters A through Z, a through z, 0 through 9, period (.), underscore (_), or hyphen (-) only. The name should be unique within the project.

3. Enter any comments in the Version Comment field. Use this field to note options and strategies.
4. To choose a device from the Part Selector dialog box, click **`Select`**.

   The dialog box shown in the following figure appears.

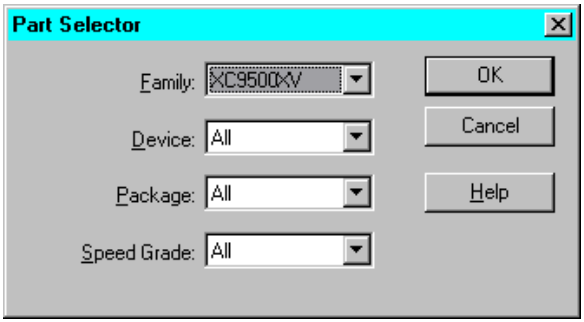


Figure 3-3 Part Selector Dialog Box

5. Specify the desired settings in the Family, Device, Package, and Speed Grade drop-down list boxes.
6. Click **OK**.
7. Enter the name for the new implementation revision in the Revision Name field. Use the characters A through Z, a through z, 0 through 9, period (.), underscore (_), or hyphen (-) only. The name should be unique within the design version.
8. Enter any comments in the Revision Comment field. Use this field to note options and strategies.
9. To copy data to your new revision from another revision or from a custom file, use the settings in the Copy Persistent Data field. Select **None** if you do not want to copy data.

   Note By default, the Design Manager copies floorplan and constraints file data from the "last" revision. The "last" revision is the bottommost revision in the Design Manager project view. When initially creating a project, the Design Manager copies constraints file data from the project directory to the revision directory.
10. Click **OK** in the New Version dialog box.

The Design Manager displays a new design version and implementation revision icon in the project view.

Creating a New Implementation Revision

After you create a design version, you can try different implementation strategies on that design. For instance, you can reduce area, increase speed, and vary placement effort. Each set of implementation strategies makes up a new implementation revision. Generating new implementation revisions allows you to vary how your design is implemented in order to achieve your design objectives.

You can select new options or target a new device to change the way your design is implemented. Targeting a new device allows you to try your design in various devices to determine the most suitable fit. For example, if a particular device proves to be too large or too slow for your needs, you can select a smaller or faster target device from a different family.

Note If you are using an HDL flow, it is strongly recommended that you create a new synthesis netlist when you change FPGA or CPLD families. This ensures that the tools make the best use of the target device.

When you create an implementation revision, an implementation revision icon is placed in the Design Manager project view. Each time you create a new implementation revision, a default revision name and the part used in the "last" revision are automatically selected. If you want to change these default values, you can enter your own revision name and change the target part. The implementation revision name, implementation state, implementation status, and any user comments for the implementation revision are indicated next to its icon in the project view when you process the implementation. Replace or delete implementation revisions that are no longer useful.

1. In the Design Manager project view, select a design version icon.
2. Choose **Design** → **New Revision** from the Design Manager menu or click the New Revision toolbar button.

The dialog box shown in the following figure appears.

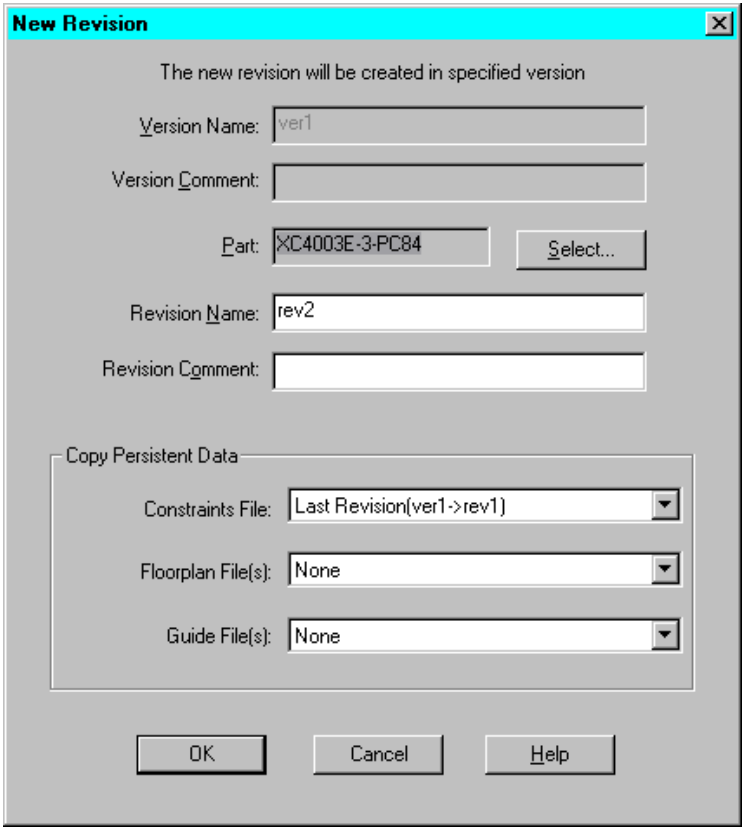


Figure 3-4 New Revision Dialog Box

3. To choose a device from the Part Selector dialog box, click **`Select`**.

The dialog box shown in the following figure appears.

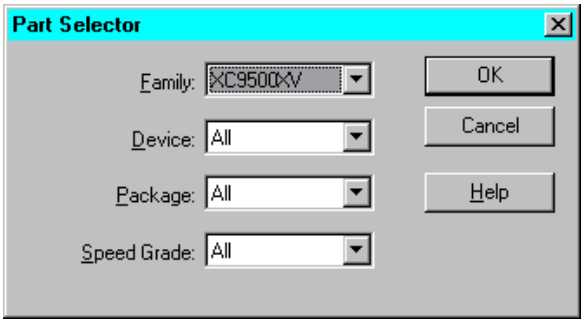


Figure 3-5 Part Selector Dialog Box

4. Specify the desired settings in the Family, Device, Package, and Speed Grade drop-down list boxes.
5. Click **OK**.
6. Enter a name for the new implementation revision in the Revision Name field. Use the characters A through Z, a through z, 0 through 9, period (.), underscore (_), or hyphen (-) only. The name should be unique within the design version.
7. Enter any comments in the Revision Comment field. Use this field to note options and strategies.
8. To copy data to your new revision from another revision or from a custom file, use the settings in the Copy Persistent Data field. Select **None** if you do not want to copy data.

   Note By default, the Design Manager copies floorplan and constraints file data from the "last" revision. The "last" revision is the bottommost revision in the Design Manager project view. When initially creating a project, the Design Manager copies constraints file data from the project directory to the revision directory.
9. In the New Revision dialog box, click **OK**.

   The Design Manager creates a new implementation revision and displays its icon in the project view.

Setting a Part

Use the Set Part command to set the part number for a new implementation revision. You can change the part number to select the device best suited for your design.

Note You cannot change the part number of an existing implementation revision.

1. In the Design Manager, select **`Design`** → **`Set Part`** or click the Set Part toolbar button.

The dialog box shown in the following figure appears.

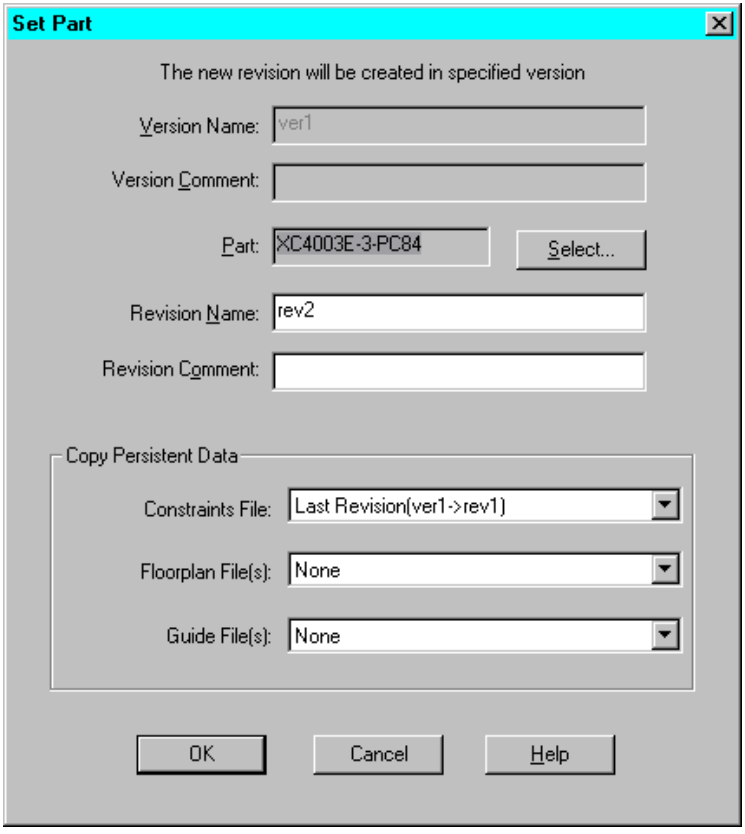


Figure 3-6 Set Part Dialog Box

2. Click **`Select`** to choose a device from the Part Selector dialog box.

The dialog box shown in the following figure appears.

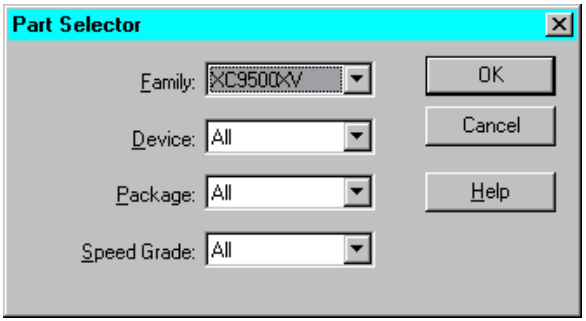


Figure 3-7 Part Selector Dialog Box

3. Specify the desired settings in the Family, Device, Package, and Speed Grade drop-down list boxes.
4. Click **OK**.
5. In the Set Part dialog box, click **OK**.

   The Design Manager displays the new implementation revision in the project view and the part number in the status bar.

Deleting Items from the Project View

Warning

When deleting an item from the project view, the Design Manager deletes the item and all accompanying data. Deleted data cannot be recovered.

Use the Delete command to delete a project, design version, or implementation revision from the project view.

1. In the Design Manager project view, select the icon of the project, design version, or implementation revision that you want to delete.
2. Select **Design** → **Delete**.
3. A confirmation box appears asking if you want to delete the selected item. Click **OK**.

Specifying Implementation Flow Options

You can specify options that control how the Flow Engine implements a design, configures a device, creates netlist files, and generates reports, timing data, and configuration data. The available options depend on the target device family.

You can specify these options by choosing settings within the Options dialog box. You can set implementation, simulation, and configuration options. The implementation options control how the software translates, maps, places, routes, and optimizes an FPGA design and how it translates and fits a CPLD design. The simulation options control the creation of netlists in terms of Xilinx primitives, allowing you to perform simulation and back-annotation. The configuration options define the initial configuration parameters of a device, the startup sequence, and readback capabilities. See the online help for information on each implementation, simulation, and configuration option.

Changes you make in the Options dialog box are applied to the implementation revision that was selected when you invoked the dialog box. The Flow Engine picks up changes after the currently running step is finished if a flow is running, or when you implement your design if no flow is running.

1. Open the Options dialog box using one of the following methods.
   - From the Design Manager, select **Design** → **Options**.
   - From the Flow Engine, select **Setup** → **Options**.
   - Click the Set Options toolbar button.

If you are targeting an FPGA, the dialog box shown in the following figure appears.

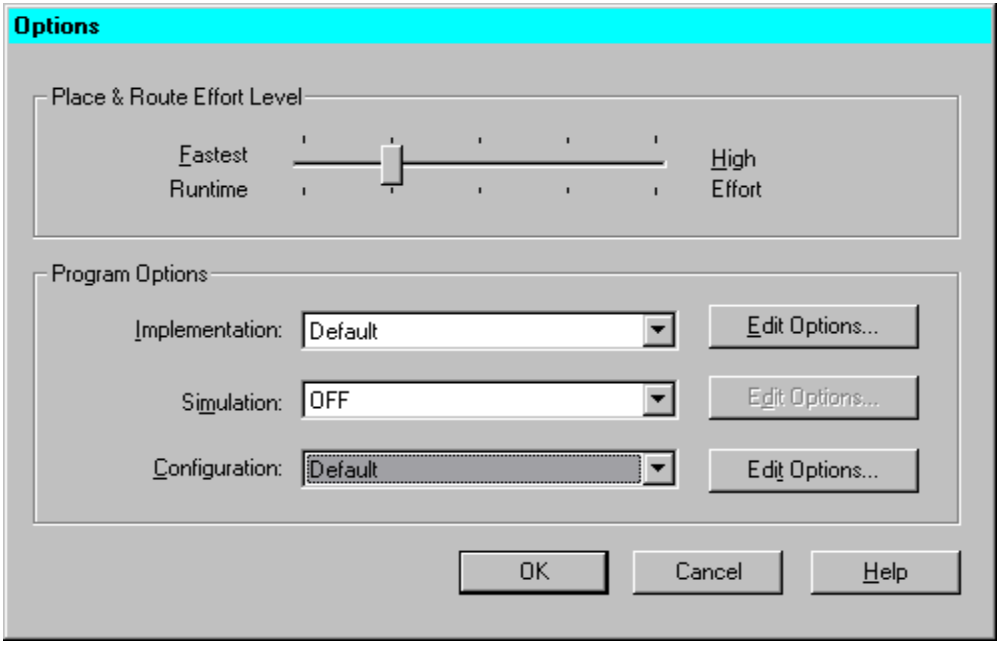


Figure 3-8 Options Dialog Box (FPGA)

If you are targeting a CPLD, the dialog box shown in the following figure appears.

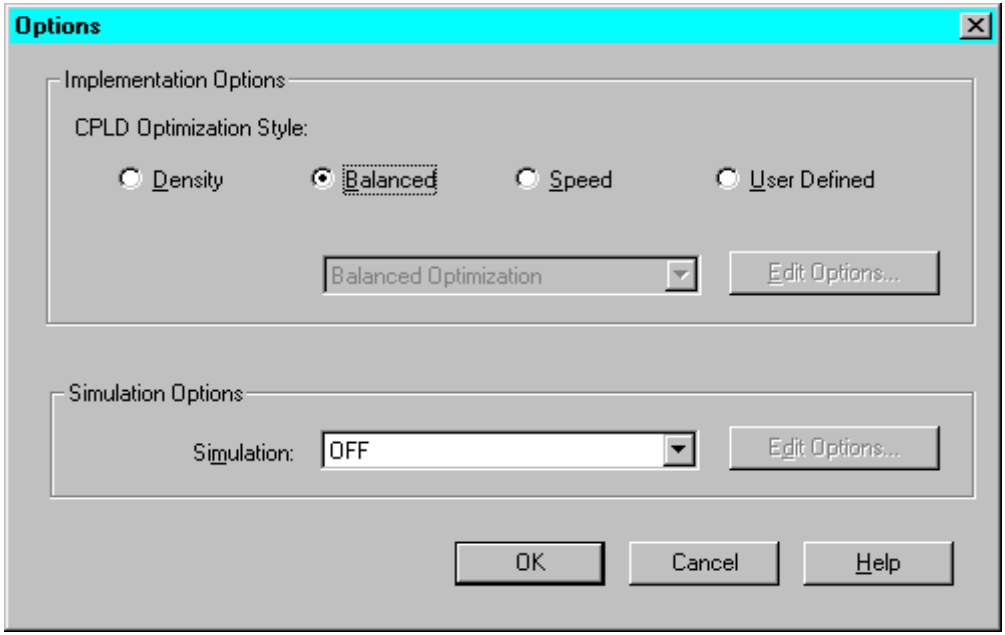


Figure 3-9 Options Dialog Box (CPLD)

2. If you are targeting an FPGA, do the following.

   a) Select a Place & Route Effort Level setting. A setting closer to High Effort provides better place and route results at the expense of longer run times.

   b) To modify the default implementation options, click the **`Edit Options`** button to the right of the Implementation field. These options affect the Translate, Map, and Place&Route steps in the implementation flow.

   c) To create timing simulation data, select one of the simulators in the drop-down list box next to the Simulation field. To edit the default option settings, click the **`Edit Options`** button to the right of the Simulation field. Select **`OFF`** if you do not want to generate simulation data.

   Note The Edit Options button is disabled if you set Simulation to OFF.

d) To modify the default configuration options, click the **Edit Options** button to the right of the Configuration field. These options affect the Configure step in the implementation flow. Select **JTAG** if you want to program FPGAs using JTAG software and the JTAG port. Select **OFF** if you do not want to generate configuration data.

Note The Edit Options button is disabled if you set Configuration to OFF. For information on setting Express Mode configuration for the SpartanXL family, see `http://support.xilinx.com/techdocs/2328.htm`.

e) Click **OK**.

3. If you are targeting a CPLD, do the following.

a) In the Implementation Options field, select **Area**, **Balanced**, or **Speed** for the CPLD Optimization Style, or select **User Defined** and choose a set of options to edit from the drop-down list box. To edit the options, click the **Edit Options** button. These options affect the Translate, Fit, and Bitstream steps in the implementation flow.

b) To create timing simulation data, select one of the simulators in the drop-down list box next to the Simulation field. If you want to edit the default option settings, click the **Edit Options** button to the right of the Simulation field. Select **OFF** if you do not want to generate simulation data.

Note The Edit Options button is disabled if you set Simulation to OFF.

c) Click **OK**.

4. Click **OK** to maintain your settings and exit the Options dialog box.

Implementing a Design from the Design Manager

You can implement your design in an automatic step from the Design Manager. When you implement your design automatically, the Flow Engine implements your design to completion.

You can implement your design with default values set for copying guide, floorplan, and constraint file data and default values for implementation flow options, or you can set your own values. If you

want to set your own values, follow the procedures described in the "Specifying Implementation Flow Options" and "Copying Constraints, Guide, and Floorplan File Data" sections before implementing your design.

Implementing a Design Automatically

The following procedure describes how to implement a design automatically from the Design Manager.

1. Select **Design** → **Implement** or click the Implement toolbar button.

   One of the following occurs.

   - If you altered your source design, the software opens the New Version dialog box. See the "Creating a New Design Version" section for information on setting the options in this dialog box. The Design Manager creates a new version and revision and the Flow Engine implements this new revision.
   - If you did not alter the source design or part type, the Smart Flow Engine appears and implements your design starting with the appropriate step.

   When processing is complete, the Smart Flow Engine closes and the Implement Status dialog box appears.

   Note The Flow Engine is run on the "last" revision regardless of the currently selected revision. To implement a revision other than the "last" one, follow the procedure in the following section.

2. In the Implement Status dialog box, click **Reports** to view the reports generated by the Flow Engine or click **View Log File** to view the implementation log file.

   Note At this point you can also perform timing simulation, if you set options as described in the "Producing Timing Simulation Data" section, and program the device. Timing simulation is described in the interface user guide for your system. Device programming is described in the programmer user guide for your system.

Re-Implementing a Design

If you want to automatically implement an implementation revision that is not your "last" revision, use the following procedure.

1. Select an implementation revision.
2. Right-click the selected revision.
3. Select **`Re-Implement`**.

   The Flow Engine determines where it should start running in the flow and implements your revision to completion.

Copying Constraints, Guide, and Floorplan File Data

You can copy constraints, guide, and floorplan file data to an existing implementation revision. Copy this data to your implementation revision to control the implementation of your design. Specify the data you want to copy using the Design Manager. The following sections describe how to copy this data to an existing revision.

- "Setting a Constraints File"
- "Setting a Guide File"
- "Setting Floorplan Files"

Note The following procedures describe how to copy data to an existing implementation revision. If you want to copy data to a new design version or implementation revision, use the Copy Persistent Data option described in the "Creating a New Design Version" or "Creating a New Implementation Revision" section.

Setting a Constraints File

You can use a user constraints (UCF) file to control the implementation of your design. The user constraints file can contain information on where to place I/O pins and blocks of logic and timing requirements for the design.

If you want to control the implementation of your design with a user constraints file, you can specify this file in the Set Constraints File dialog box. The software tries to implement your design to meet the specified timing requirements and other constraints.

1. In the Design Manager, select **Design** → **Set Constraints File** to open the dialog box shown in the following figure.

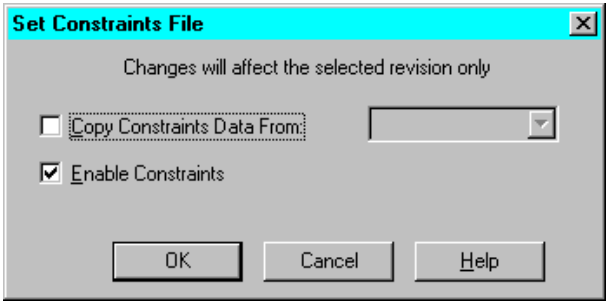


Figure 3-10 Set Constraints File Dialog Box

2. Make sure the **Copy Constraints Data From** and **Enable Constraints** options are selected.
3. In the drop-down list box, select one of the following.
   - A revision that contains the user constraints file (UCF) you want to use for this implementation
   - **None** if you do not want to copy constraints data
   - **Custom** to guide from a specific file

     If you select Custom, the following dialog box appears. Type the name of a specific file in the Constraints File field or click **Browse** to open a file selection dialog box in which you can choose an existing UCF file.

   Note By default, the Copy the Constraints File to the Revision and Operate Using the Revision Copy option is enabled and the UCF file is automatically copied into each new revision directory. Each revision reads the UCF file in its local directory. If you want each revision to read the same UCF file stored in the project directory, deselect this option.

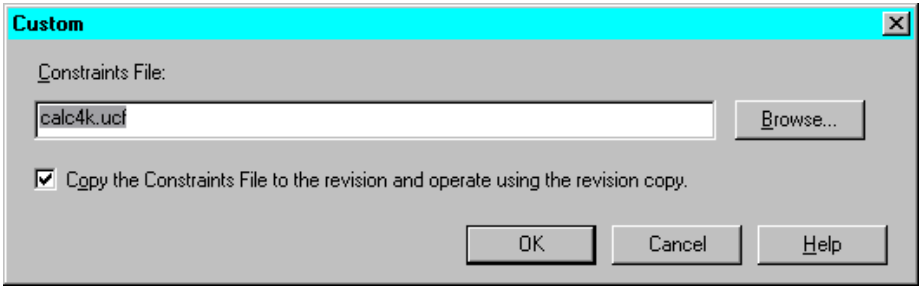


Figure 3-11 Set Constraints File Custom Dialog Box

4. In the Set Constraints File dialog box, click **OK**.

   When you implement the design, the Flow Engine uses the copied data to constrain the implementation.

Setting a Guide File

You can select a guide file or a previously routed or fitted implementation revision to use as a guide for implementation. The procedure for guiding your implementation is the same for FPGAs and CPLDs. However, the way the design is guided differs between the two.

When guiding an FPGA design, the software attempts to use the guide for placing logic and routing signals for the current implementation revision of the design. This can reduce the amount of time the software takes to place and route. Guiding a design for an FPGA works as follows.

- If a component in the new design has the same name as that of the guide design or file, it is placed as in the guide.
- If an unnamed component in the new design is the same type as a component within the guide, it is placed as in the guide.
- If the signals attached to a component in the new design match the signals attached to the component of the guide, the pins are swapped to match the guide, where possible.
- If the signal names in the input design match the guide, and have the same sources and loads, the routing information from the guide design is copied to the new design.

After these components and signals are placed and routed, the remainder of the logic is placed and routed. If you have made only

minor changes to your design and want the remaining logic placed and routed exactly as in your guide design, select the **Match Guide Design Exactly** option. This option locks the placement and routing of the matching logic so that it cannot change to accommodate additional logic.

Note Setting the Match Guide Design Exactly option is not recommended for synthesis-based designs.

For CPLDs, each time you implement your design, a guide (GYD) file is created which contains your pinout information. You can reuse this file in subsequent iterations of your design if you want to keep the same pinouts. If you select a valid implementation revision or guide file name, the pinouts from that file will be used when the design is processed.

Note You can override guide file locations by assigning locations in your design file or constraints file.

1. In the Design Manager, select **Design** → **Set Guide File(s)** to open the dialog box shown in the following figure.

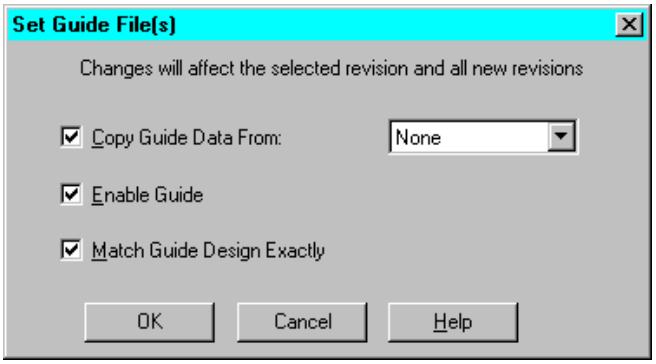


Figure 3-12 Set Guide File(s) Dialog Box

2. Make sure the **Copy Guide Data From** and **Enable Guide** options are selected.
3. In the drop-down list box, select one of the following.
   - A revision that contains the guide file you want to use for this implementation
   - **None** if you do not want to copy a guide file

- **Custom** to guide from any mapped or routed file for FPGAs or fitted file for CPLDs, including designs not generated from within the Design Manager

  If you select Custom, the following dialog box appears. Type the name of a mapped, routed, or fitted file in the Guide File field, or click **Browse** to open a file selection dialog box in which you can choose an existing file. Choose an NCD file for FPGAs or a GYD file for CPLDs. You can also specify a mapping guide file for FPGAs.

Figure 3-13 Set Guide File(s) Custom Dialog Box

Note The implementation revision or revision data is based on a placed and routed design. Guide from a placed and routed file rather than a mapped file to reduce run time. To guide from a mapped file, you must use the Custom option. If you use this option, you cannot guide mapping using the Set Floorplan File(s) command.

4. For FPGA devices, select **Match Guide Design Exactly** if you want to lock the placement and routing of matching logic.

   If you do *not* select this option, the guide files are used as a starting point only. This allows the mapper, placer, and router greater flexibility in accommodating design modifications, often resulting in greater overall success.

   Note For synthesis-based designs, use the Match Guide Design Exactly option only if the guide file is from the same design version.

5. Click **OK**.

When you implement the design, the Flow Engine uses the copied data to guide the implementation.

Setting Floorplan Files

When you use the Floorplanner, an MFP file is generated that contains mapping information. You can instruct the Design Manager to use this file as a guide for mapping an implementation revision using the Set Floorplan File(s) command. To use this command, you must select an implementation revision that has been mapped and modified using the Floorplanner. For information on using the Floorplanner, see the *Floorplanner Guide*.

Note If you use the Set Floorplan File(s) command, you cannot guide mapping using the Set Guide File(s) command Custom option. The Set Floorplan File(s) command is available for the XC4000 families and Virtex, Virtex-E, Virtex-II, Spartan, SpartanXL, and Spartan-II FPGAs only.

1. In the Design Manager, select **Design** → **Set Floorplan File(s)** to open the dialog box shown in the following figure.

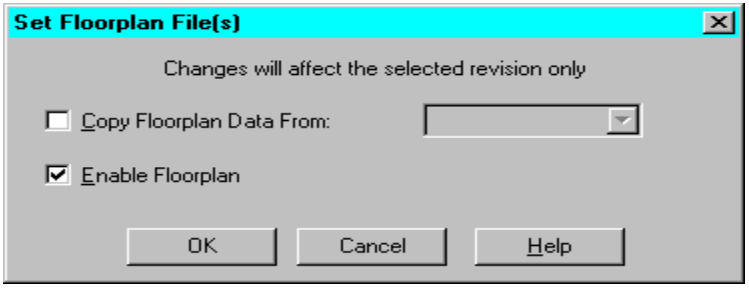


Figure 3-14 Set Floorplan File(s) Dialog Box

2. Make sure the **Copy Floorplan Data From** and **Enable Floorplan** options are selected.
3. In the drop-down list box, select one of the following.
   - A revision that contains the floorplan files you want to use for this implementation
   - **None** if you do not want to copy floorplan data

- **Custom** to guide from any mapped file in your file system, including designs not generated from within the Design Manager

  If you select **Custom**, the following dialog box appears. Type the name of a specific file in the Floorplanning File field, or click **Browse** to open a file selection dialog box in which you can choose an existing file. Specify an FNF file for the Floorplanning File field and an MFP file for the Floorplanned Guide File field.

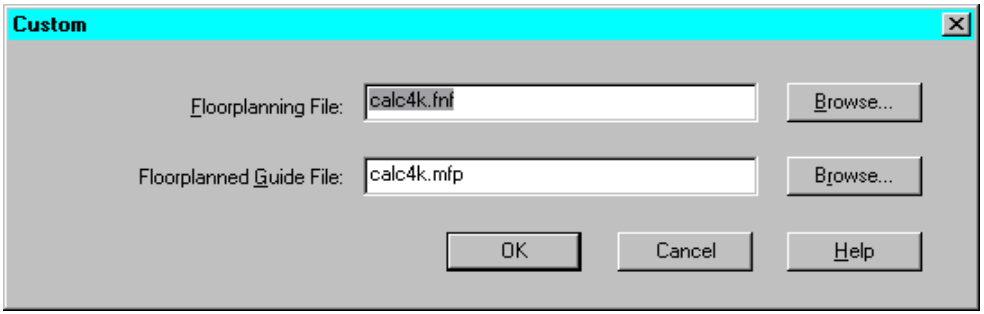


Figure 3-15 Set Floorplan File(s) Custom Dialog Box

4. Click **OK**.

   The Flow Engine uses the copied data to guide the implementation.

Viewing Reports

The Flow Engine generates various reports which you can view in the Report Browser. The Report Browser opens reports for the selected implementation revision.

1. Select **Utilities** → **Report Browser** or click the Browse Reports button.

   The Report Browser window opens and displays reports for the active implementation revision.

2. In the Report Browser window, double-click the report icon for the report you want to view.

   Timing reports are displayed in the Timing Analyzer, but these ASCII files can also be viewed using a text editor. All other reports are displayed in the standard text editor that you specified with the **File** → **Preferences** command.

   Note The icons change appearance to indicate whether or not you have read a report. A yellow mark in the upper left corner of the report icon indicates that the report has been generated but not read. A report icon without this mark indicates that the report has been generated and read.

Producing Timing Reports

You can generate timing reports after you implement your design using the Produce Timing Reports command. You can use this command if Map for FPGAs or Fit for CPLDs has completed successfully. The timing report created is based on the state of your design when you run the command.

Note If you select the Produce Logic Level Timing Report or Produce Post Layout Timing Report option in the Timing Reports tab of the Implementation Options dialog box *before* you implement your design, the software automatically generates timing reports with the Map and Place&Route steps. See the "Specifying Implementation Flow Options" section for information on making settings in the Implementation Options dialog box.

1. Select **Utilities** → **Produce Timing Reports**.

The following dialog box appears if you are targeting an FPGA.

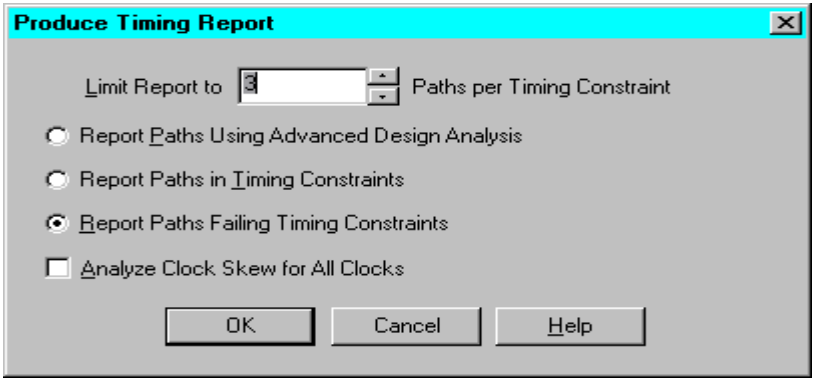


Figure 3-16 Produce Timing Report Dialog Box (FPGA)

The following dialog box appears if you are targeting a CPLD.

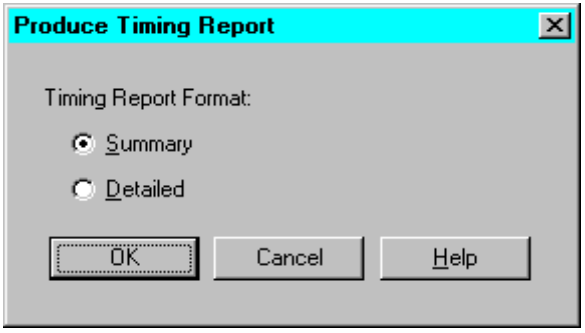


Figure 3-17 Produce Timing Report Dialog Box (CPLD)

2. Select a timing report format.

   The timing report is created and placed in the Report Browser. See the "Viewing Reports" section for information on viewing reports.

Generating Pin Locking Constraints

You can generate pin locking constraints in your UCF file for use with other Xilinx implementation tools. Pinout information is taken from a placed and routed NCD file for FPGAs or a fitted GYD file for CPLDs.

1. In the Design Manager, select **Design** → **Lock Pins**.
2. In the confirmation dialog box, click **Yes**.

   Pin locking constraints that you created with this command are added to your UCF file in the PINLOCK section.
3. After the constraints are added, the dialog box shown in the following figure appears. Click **View Lock Pins Report** to view the report.

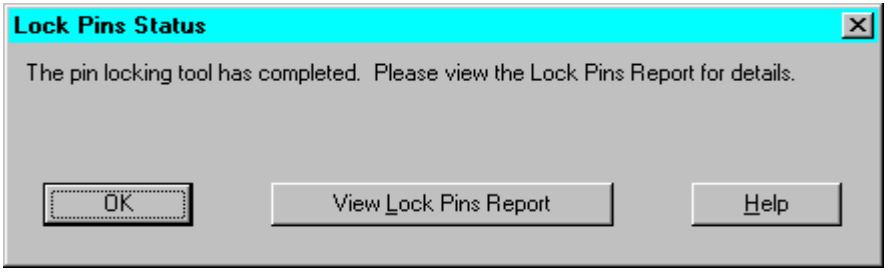


Figure 3-18 Lock Pins Status Dialog Box

You can view the pinouts using the Constraints Editor. See the *Constraints Editor Guide* for more information.

Note If you want to view the report after you have dismissed the Lock Pins Status dialog box, use the **Utilities** → **Lock Pins Report** from the Design Manager.

Producing Timing Simulation Data

The Flow Engine can produce timing simulation data for use in a third party simulation tool. This data is produced for the selected implementation revision.

After you implement your design, you can perform timing simulation to test the timing requirements and functionality of your design. Timing simulation can save considerable time by reducing time spent debugging test boards in the lab.

You can create timing simulation netlists after the design has been placed and routed. Additionally, you can create simulation data after the design has been mapped or after the design has been placed but not routed.

Simulation data created after the design has only been mapped contains timing data based on the CLB and IOB block delays, and most net delays are zero. Post-map simulation allows you to ensure that the design's current implementation will give the place and route software sufficient margin to route the design within your timing requirements.

Simulation data created after the design has been placed but not routed contains accurate block delays and estimates for the net delays. Post-place simulation can be used as an incremental simulation step between post-map simulation and a complete post-route timing simulation.

1. Open the Options dialog box using one of the following methods.
   - From the Design Manager, select **Design** → **Options**.
   - From the Flow Engine, select **Setup** → **Options**.
   - Click the Set Options toolbar button.

If you are targeting an FPGA, the dialog box shown in the following figure appears.

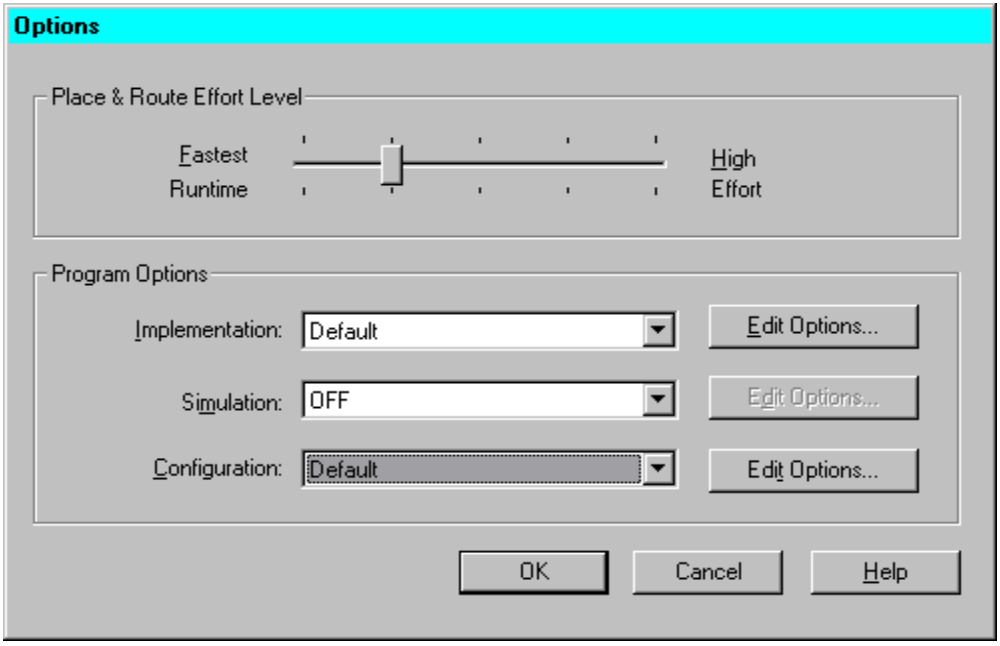


Figure 3-19 Options Dialog Box (FPGA)

If you are targeting a CPLD, the dialog box shown in the following figure appears.

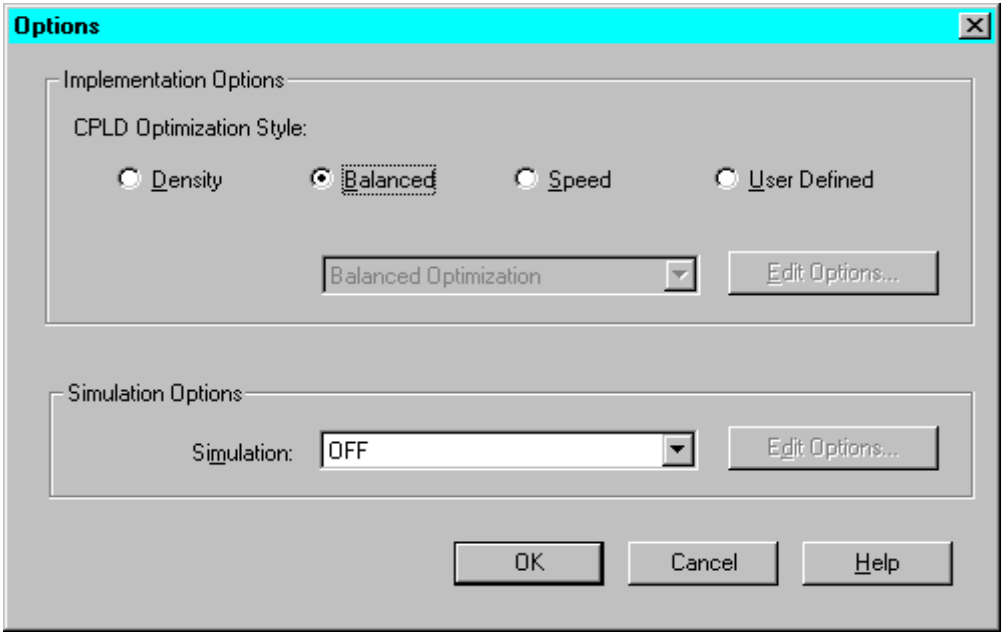


Figure 3-20 Options Dialog Box (CPLD)

2. Select one of the simulators in the drop-down list box next to the Simulation field.
3. To edit the default option settings, click the **`Edit Options`** button to the right of the Simulation field.

   Note If you set Simulation to OFF, no timing simulation data is created and the Edit Options button is disabled.
4. Make settings in the Simulation Options dialog box and click **OK**. See the online help for information on each simulation option.
5. Click **OK**.

   When you implement the design, the Flow Engine produces timing simulation data files. Each time the data is produced, it is automatically exported to your design directory. You can use these files to simulate the design with a supported third party simulation tool.

Exporting Design Data

You can export design data created in the Design Manager to other environments. The Design Manager exports design data for the selected implementation revision.

1. In the Design Manager, select **Design** → **Export** to open the dialog box shown in the following figure.

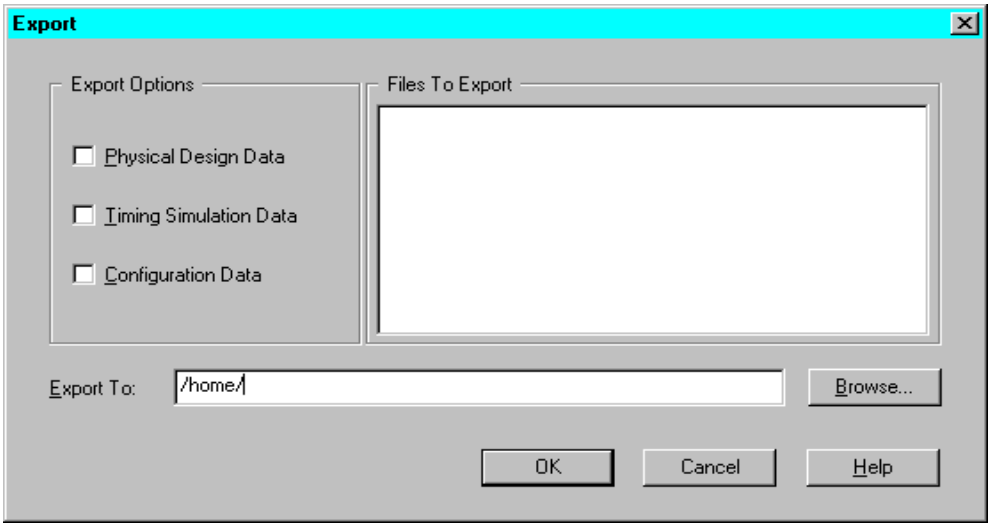


Figure 3-21 Export Dialog Box

2. In the Export Options field, select the type or types of data to export, whether **Physical Design Data**, **Timing Simulation Data**, or **Configuration Data**.

   The associated files appear in the Files to Export field.

3. Enter a directory path in the Export To field or click Browse to browse for a directory.

4. Click **OK**.

Archiving a Project

Use the following procedure to archive your project using the zip tool provided by Xilinx. This tool is useful for storing different revisions of your project and for sending projects to co-workers. If you are

having trouble with your design, you can also send your project to Xilinx Technical Support for assistance.

1. Select **File** → **Archive Project** to open the dialog box shown in the following figure.

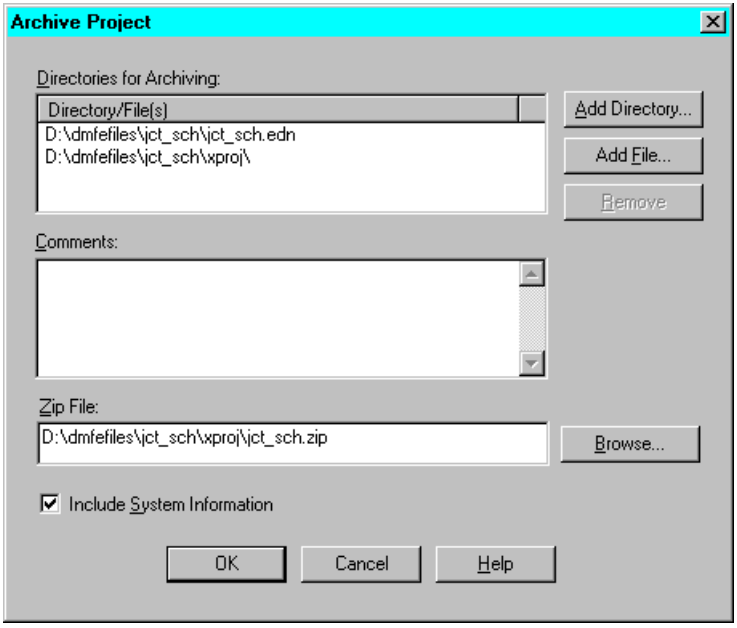


Figure 3-22 Archive Project Dialog Box

   By default, the xproj directory and the top level netlist file for the current project appears in the Directories for Archiving list box.

2. Click the **Add Directory** button to open a standard directory selection dialog box in which you can specify a directory to add, or click the **Add File** button to open a standard file selection dialog box in which you can specify a file. To remove a directory or file, click its name in the Directories for Archiving list box and click **Remove**.

3. Type any comments related to your design in the Comments field.

4. To generate diagnostic information and include it with your ZIP file, make sure **Include System Information** is selected. This information is stored in a file called xray.out.

5. By default, the Zip File field shows the name of the ZIP file as *design_name*.zip. If you want to specify a different name, use the **Browse** button.
6. Click **OK**.
7. If you want to send your file to Xilinx, contact Xilinx Technical Support. For contact information, see `http://www.support.xilinx.com/support/techsup/tappinfo.htm` or refer to the Technical Support page in the online help.

Implementing a Design from the Flow Engine

If you want to control the implementation processes, you can implement your design in separate steps from the Flow Engine. Implementing your design from the Flow Engine allows you to control aspects of the flow such as updating the implementation status, enabling flashing icons, and setting run targets. You can implement your design with default implementation flow options set, or you can set your own values. This section contains the following procedures.

- "Implementing a Design in Separate Steps"
- "Setting the State of the Flow"
- "Enabling Flashing Icons"
- "Setting a Run Target"
- "Updating the Flow"

Implementing a Design in Separate Steps

The following procedure describes how to implement your design in separate steps. Follow this procedure according to how you want to implement your design.

1. If you want to set implementation flow options or specify constraint, floorplan, and guide file data for copying, follow the procedures in the "Specifying Implementation Flow Options" and "Copying Constraints, Guide, and Floorplan File Data" sections.

2. If you want to set a run target, enable flashing icons, or update the implementation status or flow, use the procedures in the sections that follow.
3. Do one of the following.
   - Select **Flow** → **Run** to run through the entire implementation process.
   - Select **Flow** → **Step** to single step through the implementation process.

   Note If you stop processing while the Flow Engine is running the Place&Route step, the Flow Engine asks whether you want to save the data that has processed to this point or exit immediately without saving the data.

Setting the State of the Flow

Use the following procedure to set the state of the implementation flow.

Note This procedure is not used in normal Flow Engine use. It is used if some processing on the design was performed outside of the Design Manager or Flow Engine, such as in the FPGA Editor. It can also be used if you ran the Flow Engine Step Back button by mistake and want to reset the implementation state to its original state.

1. Select **Setup** → **Advanced** to open the dialog box shown in the following figure.

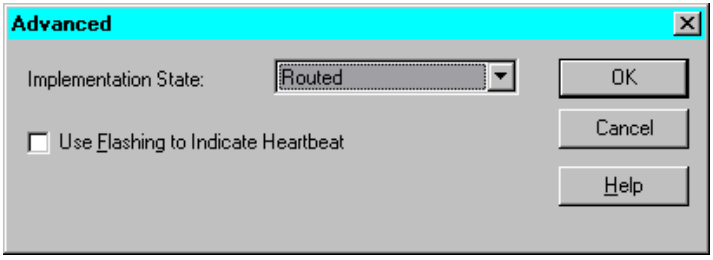


Figure 3-23 Advanced Dialog Box

2. Select a state from the Implementation State drop-down list box.
3. Click **OK**.

Enabling Flashing Icons

You can enable flashing icons to indicate that a process step is being processed. A trade-off of this feature is that flashing icons slow down the implementation process.

1. Select **Setup** → **Advanced** to open the Advanced dialog box shown in the preceding figure.
2. Select **Use Flashing to Indicate Heartbeat**.
3. Click **OK**.

Setting a Run Target

The Flow Engine flow includes several process steps. You can specify that the flow stop at a certain point with the Stop After command. When specified, the Flow Engine does not process beyond that point. You do not need to set a run target for typical Flow Engine use. By default, the Flow Engine processes all the steps.

1. Select **Setup** → **Stop After** or click the Set Target toolbar button.

The dialog box shown in the following figure appears.

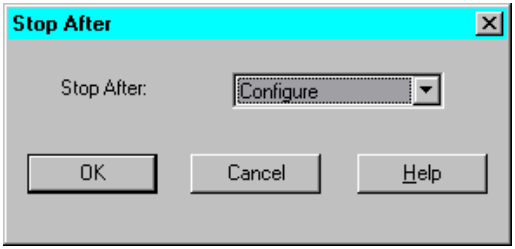


Figure 3-24 Stop After Dialog Box

2. Select the desired run target from the Stop After drop-down list box.
3. Click **OK**.

The Flow Engine stops processing after the specified point. A red stop sign indicates the run target.

Updating the Flow

When the Flow Engine is first invoked, it automatically looks for changes made to certain files. If changes are detected, the Flow Engine resets the flow for the implementation revision. For information on which file changes cause the flow to be reset, see the "Design Implementation" section of the "Introduction" chapter. If you want to check for changes and the Flow Engine is already open, use the following procedure.

1. Select **Setup** → **Update Flow**.

   The Flow Engine resets the flow if certain file changes are detected.

2. Do one of the following.

   - Select **Flow** → **Run** to run through the entire implementation process.
   - Select **Flow** → **Step** to single step through the implementation process.

Placing and Routing Non-Timing Driven Designs

When targeting Virtex, Virtex-E, Virtex-II, and Spartan-II devices on designs that are run without timing constraints, you may notice lower design performance results from the place and route process. To get a more realistic estimate of how your design will perform without timing constraints applied, do the following.

1. In the Flow Engine, set your run target to **Place&Route**.

   See the "Setting a Run Target" section for more information.

2. Select **Flow** → **Run**.

3. After the Flow Engine has finished routing your design, select **Setup** → **FPGA Re-entrant Route** to open the dialog box shown in the following figure.

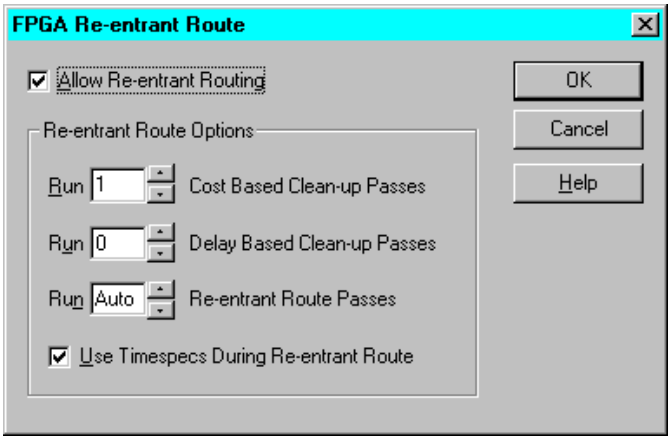


Figure 3-25 FPGA Re-Entrant Route Dialog Box

4. Specify the number of Clean-up Passes to run.
5. Click **OK**.
6. Select **Flow** → **Run**.
7. Check the Place & Route Report and Post Layout Timing Report in the Report Browser for improved performance results. See the "Viewing Reports" section for information on viewing the report.

Running Multiple Place and Route Passes

Running multiple place and route passes allows you to automatically generate multiple place and route solutions for a design. The place and route process offers both the ability to generate a determinate, repeatable solution by using the same starting point, or cost table, and the ability to generate a range of unique solutions by using different cost tables.

The FPGA Multi-Pass Place & Route command automates the ability to run Place and Route several times with different cost tables. It scores each place and route pass and uses the score to determine the best passes to save. Scores are based on factors such as the number of unrouted nets, the delays on nets, and conformity to your timing

constraints. The passes you save appear as implementation revisions in the Design Manager project view.

On average, design speed from an arbitrary multiple pass place and route will vary plus or minus 5% to 10% from the median design speed across all cost tables. About 1/3 of the cost tables will result in speeds within 5% of the median, 1/3 in the 5% to 10% range, and 1/3 in the 10% to 15% range. When comparing performance from the absolute worst cost table to the absolute best, a spread of 25% to 30% is possible.

Note Ranges are narrower for Virtex families and higher for XC4000 families.

Use the FPGA Multi-Pass Place & Route command in the following situations.

- You want to evaluate whether worse than expected results are due to a poor starting point or cost table.

  In this case, run multiple place and route passes at any time during the design cycle by running 3 or 4 cost tables.

- You made small changes to a design at the end of the design cycle and performance falls 5% to 10% as a result of these changes.

  In this case, run 5 to 10 cost tables.

If you are working on a workstation, you can use the Nodelist File option to execute the place and route process on multiple machines. This significantly reduces the run time.

Note The FPGA Multi-Pass Place & Route command is supported for the FPGA device families only. Using the FPGA Multi-Pass Place & Route command is not recommended for every design iteration. If you need the top performing cost tables to meet design performance, your design should be modified to remove one or more logic levels.

1. In the Design Manager, select **Design** → **FPGA Multi-Pass Place & Route** to open the dialog box shown in the following figure.

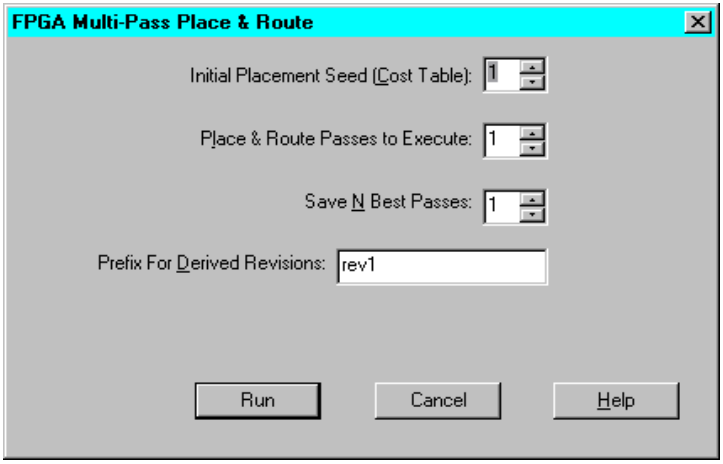


Figure 3-26 FPGA Multi-Pass Place & Route Dialog Box

2. In the Initial Placement Seed (Cost Table) field, specify a placement initialization value.

   The placement initialization value sets the number with which to begin the place and route attempts. Each pass receives an incremental value. The value you choose corresponds to a cost table index which initializes the place and route algorithms.

3. In the Place & Route Passes to Execute field, specify the number of place and route passes to attempt.

4. In the Save N Best Passes field, specify the number of place and route passes to save.

   The software saves the best passes based on the iteration design scores.

5. If you want to change the default prefix of the pass names, enter a new name in the Prefix for Derived Revisions field. The default prefix is rev#.

6. If you are working on a workstation, you can click the Nodelist File **Browse** button to choose a nodelist file that you generated.

   This file allows you to use multiple machine (nodes) that are networked together to run the place and route passes. For more information on generating a nodelist file and setting up environ-

ment variables, see the "Turns Engine (PAR Multi-Tasking Option)" section of the *Development System Reference Guide*.

7. In the FPGA Multi-Pass Place & Route dialog box, click **Run**.

   The Design Manager creates implementation revisions for the number of passes you specified in the Save N Best Passes field. After the passes are complete, the dialog box shown in the following figure appears. Click **`View Log File`** to view the log file or click **`View Summary Report`** to view the summary report.

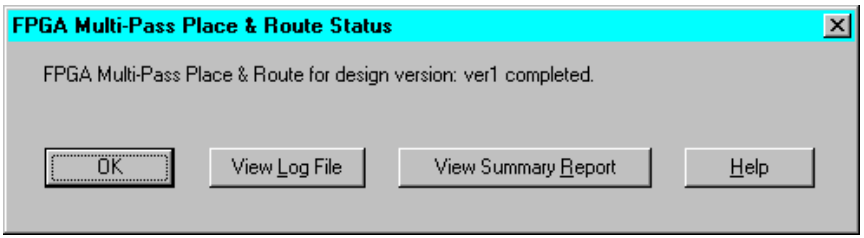


Figure 3-27 FPGA Multi-Pass Place & Route Status Dialog Box

Running Re-Entrant Routing on FPGAs

Use re-entrant routing to further route an already routed design. The design maintains its current routing and additional routing is added. You can reroute connections by running cost-based cleanup, delay-based cleanup, and additional re-entrant route passes. Cleanup passes attempt to minimize the delays on all nets and decrease the number of routing resources used. Cost-based cleanup routing is faster while delay-based cleanup is more intensive.

Re-entrant routing offers the following advantages.

- Cleanup passes significantly reduce delays, especially on non-timing driven runs.
- For timing-driven runs, cleanup passes can improve timing on elements not covered by timing constraints.
- For designs which do not meet timing goals by a narrow margin, delay-based cleanup passes can reorganize routing so that additional re-entrant route passes enable the design to meet timing goals.

Note The FPGA Re-entrant Route command is supported for the FPGA device families only.

1. In the Flow Engine, select **Setup** → **FPGA Re-entrant Route** to open the dialog box shown in the following figure.

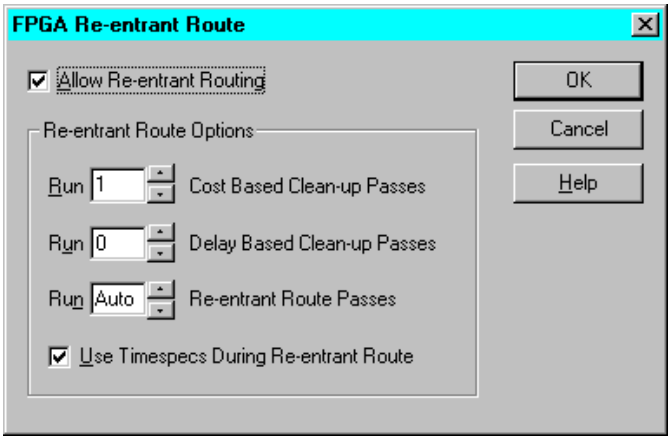


Figure 3-28 FPGA Re-Entrant Route Dialog Box

2. Select **Allow Re-entrant Routing** to route the previously routed design again.
3. Select a number between 1 and 5 for the Run _ Cost-Based Cleanup Passes field.

   These cleanup passes reroute nets if the new routing uses less costly resources than the original configuration. Cost is based on pre-determined cost tables. Cost-based cleanup usually has a faster run time than the delay-based cleanup, but does not reduce delays as significantly.

   Note If you run both cost-based and delay-based cleanup passes, the cost-based passes run first.
4. Select a number between 1 and 5 for the Run _ Delay-Based Cleanup Passes field.

   These cleanup passes reroute nets if new routing will minimize the delay for a given connection. Delay-based cleanup usually produces faster in-circuit performance.
5. Select a number between 1 to 2000 for the Run _ Re-entrant Route Passes field to run additional re-entrant routing passes.

These passes are either timing driven or non-timing driven depending on whether you specified timing constraints.

6. Select **`Use Timespecs During Re-entrant Route`** if you want to reroute the design within the specified timing constraints in your design file.
7. Click **`OK`**.

Working with Templates

An option template is a group of implementation flow option settings. Instead of setting options each time you create an implementation revision, templates provide a convenient way to have several groups of option settings that you can select from when you implement a design. For example, you can have a template for quick place and route and another one for maximum placement effort.

Note If you want to set the implementation flow options without using templates, follow the procedure in the "Specifying Implementation Flow Options" section.

The three types of option templates are implementation, simulation, and configuration templates. The implementation templates control how the software translates, maps, places, routes, and optimizes an FPGA design and how it translates and fits a CPLD design. The simulation templates control the creation of netlists in terms of Xilinx primitives, allowing you to perform simulation and back-annotation. The configuration templates set options which define the initial configuration parameters of a device, the startup sequence, and readback capabilities.

Xilinx provides default option templates with the software. You can create your own implementation, simulation, and configuration templates with the Template Manager utility. This utility also allows you to set advanced options using the Customize Options dialog box. The following sections describe these procedures.

- "Starting the Template Manager"
- "Creating a New Template"
- "Creating a Template Based on a Xilinx Template"
- "Editing a Template"
- "Setting Custom Template Options"

- "Using a Template"
- "Restoring a Template"

Starting the Template Manager

Use the following procedure to start the Template Manager. You must complete this procedure before using the procedures in the following sections.

1. In the Design Manager, select **`Utilities`** → **`Template Manager`** to open the dialog box shown in the following figure.

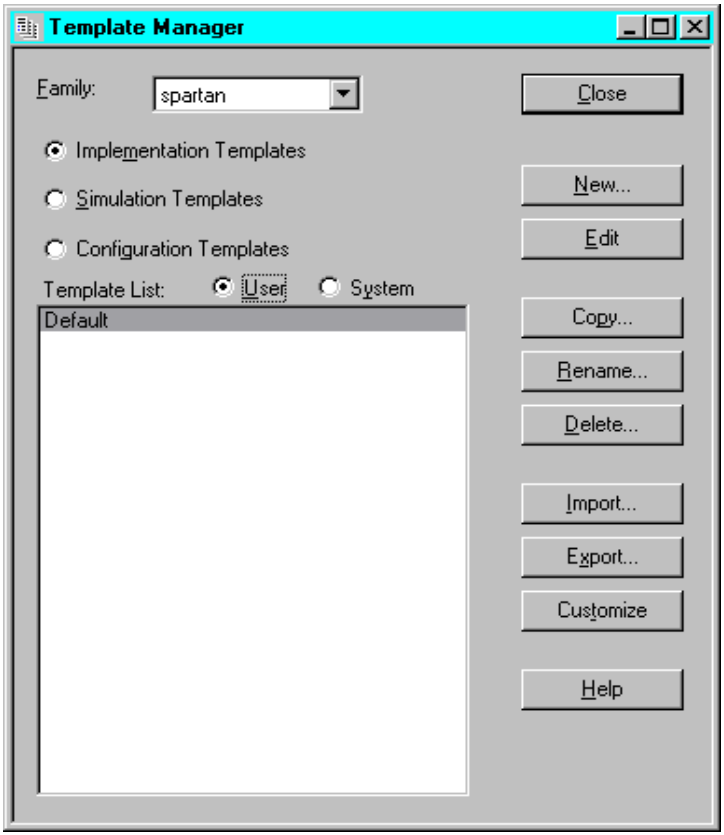


Figure 3-29 Template Manager Dialog Box

2. Select the appropriate device family in the Family drop-down list.

The following table shows which template family to select based on the FPGA or CPLD family you are targeting.

Table 3-1 Template Manager Family Usage

Targeted Device Family	Template Family
XC3000A / L	xc3000
XC3100A / L	xc3000
XC4000E / L / EX / XL / XV / XLA	xc4000
XC5200	xc5200
Spartan / XL	spartan
Spartan-II	spartan2
Virtex / -E	virtex
Virtex-II	virtex2
XC9500 / XL / XV	xc9500

3. Select either **`Implementation Templates`**, **`Simulation Templates`**, or **`Configuration Templates`** to specify the type of template with which to work.

   Note Configuration options are supported for the FPGA device families only. There are no configuration options for the CPLD family.

4. Create and modify templates from the Template Manager dialog box as described in the sections that follow.

Creating a New Template

Use the following procedure to create a new template.

1. In the Template Manager dialog box, click **`User`** to display a list of editable templates.

2. Click **New** to open the dialog box shown in the following figure.

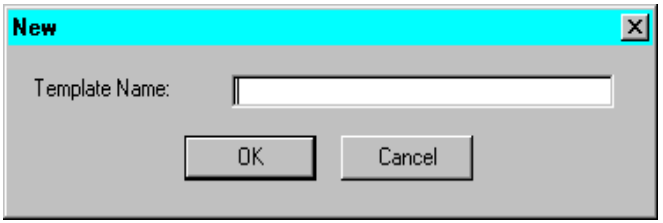


Figure 3-30 New Dialog Box

3. Type a name for the new template.
4. Click **OK**.

   The name of the new template is added to the list of User option templates in the Template Manager dialog box.

Creating a Template Based on a Xilinx Template

You can also create a template based on an existing Xilinx template.

1. In the Template Manager dialog box, click **System** to display a list of templates provided by Xilinx.

   Note You cannot edit System templates.

2. In the Template List field, click the name of the template on which you want to base your template.
3. Click **Copy**.
4. In the Copy dialog box, type a name for the new template.
5. Click **OK**.
6. Click **User**.

   The name of the new template is added to the list of User option templates in the Template Manager dialog box.

Editing a Template

After you create an option template as described in the preceding sections, you can edit the template using the following procedure.

1. In the Template Manager dialog box, click **User** to display a list of editable templates.

2. In the Template List field, click the name of the template you want to edit.
3. Click **Edit**.

   The Implementation, Simulation, or Configuration Options dialog box opens, depending on which type of template you are working with.
4. Select the desired settings.

   The available settings depend on the device family and the template type. See the online help for information on the options in the Implementation, Simulation, and Configuration Options dialog boxes.
5. Click **OK**.

   The Template Manager saves the options settings and closes the Implementation, Simulation, or Configuration Options dialog box.

Setting Custom Template Options

The options available in the option templates permit you to set the most commonly used options. If you want to set more advanced options, use the Customize feature in the Template Manager.

1. In the Template Manager dialog box, click **User** to display a list of editable templates.
2. Select a template from the Template List field.

3. Click **Customize** to open the dialog box shown in the following figure.

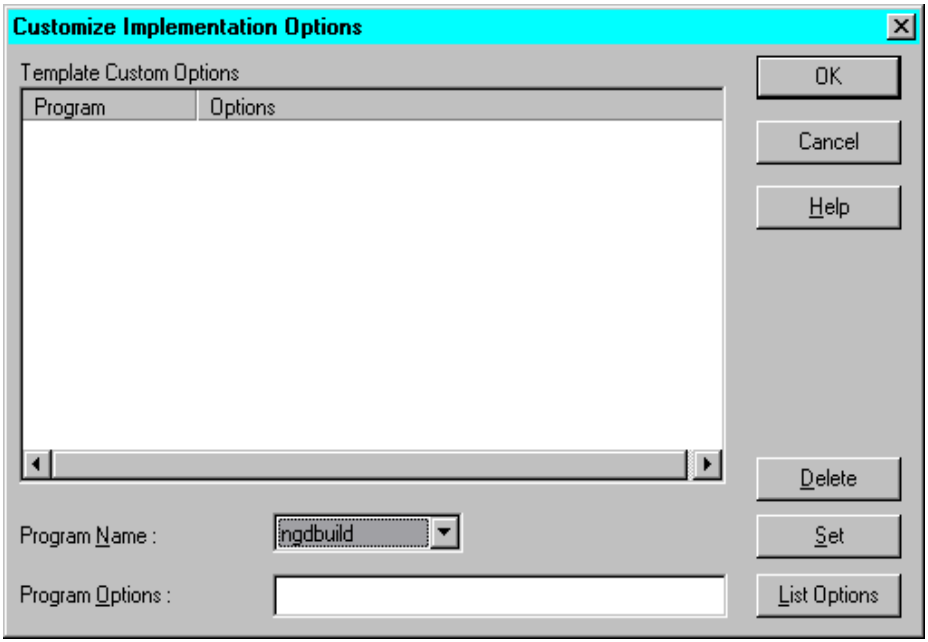


Figure 3-31 Customize Options Dialog Box

Note The title of this dialog box changes based on whether you are working with implementation, simulation, or configuration options. Any program options you have entered previously are saved and appear the next time you open this dialog box.

4. Select the program name from the Program Name drop down list box.

5. Enter the option names in the Program Options field. Enter the options as they would appear on the command line.

   Note To open a file that lists the options and usage messages associated with the selected program name, click **List Options**.

6. Click **Set**.

   The program and option names are added to the Template Custom Options list box.

7. Repeat steps 4 through 6 for each program and option you want to enter.
8. To remove a program, click the program name in the Template Custom Options field and click **Delete**.
9. Click **OK** in the Customize Options dialog box.

   Note It is possible to enter options in the Custom Options dialog box that can conflict with normal Flow Engine options. It is beyond the scope of this manual to explain all the possible conflicts.

Using a Template

After you create a template, you must specify the template to use in the Options dialog box before you implement your design.

1. Open the Options dialog box using one of the following methods.
   - From the Design Manager, select **Design** → **Options**.
   - From the Flow Engine, select **Setup** → **Options**.
   - Click the Set Options toolbar button.

If you are targeting an FPGA, the dialog box shown in the following figure appears.

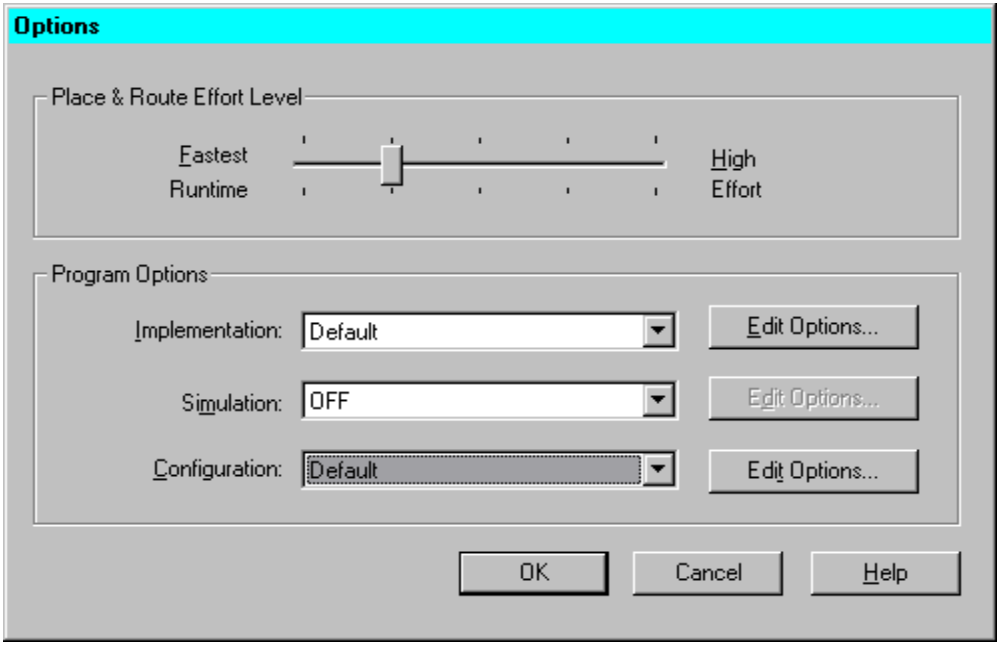


Figure 3-32 Options Dialog Box (FPGA)

If you are targeting a CPLD, the dialog box shown in the following figure appears.

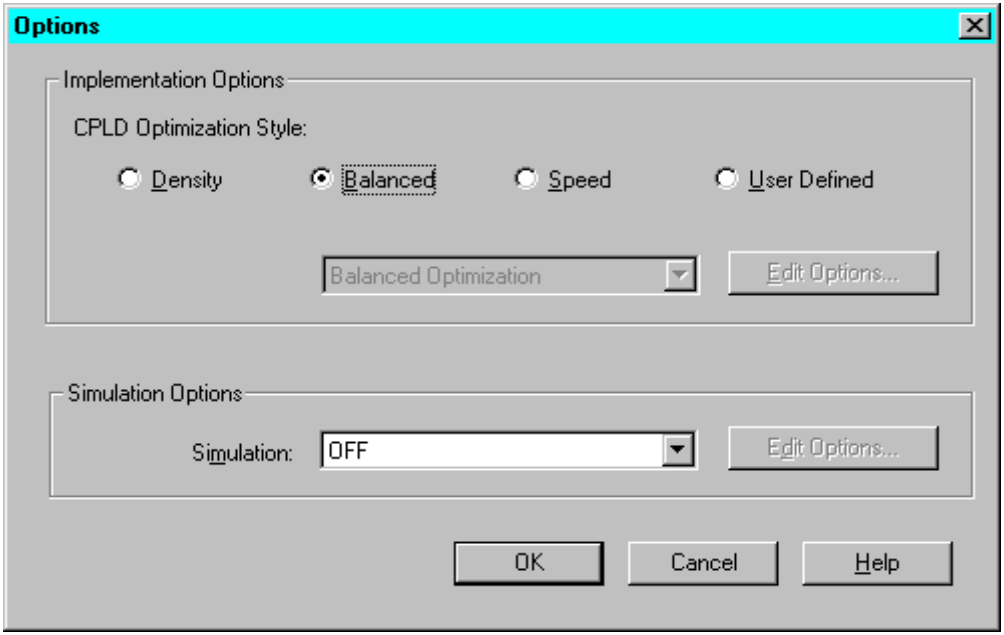


Figure 3-33 Options Dialog Box (CPLD)

2. If you are targeting an FPGA, do the following.

   a) If you created an implementation option template, select the appropriate template name from the drop-down list box next to the Implementation field.

   b) If you created a simulation option template, select the appropriate template name from the drop-down list box next to the Simulation field.

   c) If you created a configuration option template, select the appropriate template name from the drop-down list box next to the Configuration field.

3. If you are targeting a CPLD, do the following.

   a) If you created an implementation option template, select **`User Defined`** in the Implementation Options group box

and select the appropriate template name from the drop-down list box.

b) If you created a simulation option template, select the appropriate template name from the drop-down list box next to the Simulation field.

4. Click **OK** to maintain your settings and exit the Options dialog box.
5. Implement your design as described in the "Implementing a Design from the Design Manager" or "Implementing a Design from the Flow Engine" section.

Restoring a Template

If you want to restore your template settings to the original Xilinx settings, click the **Reset** button.

Note If you click Reset in a template you created in a previous release, template settings revert to the settings in this release.

Glossary

asynchronous debugging

Asynchronous debugging is a debugging mode in which you capture data without controlling your system clock.

BIT file

BIT file is the same as a bitstream file. See bitstream.

bitstream (BIT file)

A bitstream file is a stream of data that contains location information for logic on a device, that is, the placement of Configurable Logic Blocks (CLBs), Input / Output Blocks (IOBs), TBUFs, pins, and routing elements. The bitstream also includes empty placeholders that are filled with the logical states sent by the device during a readback. Only the memory elements, such as flip-flops, RAMs, and CLB outputs, are mapped to these placeholders, because their contents are likely to change from one state to another. When downloaded to a device, a bitstream configures the logic of a device and programs the device so that the states of that device can be read back.
A bitstream file has a .bit extension.

block

A block is a group of one or more logic functions.

bottom-up design

Bottom-up design is an HDL design methodology where already defined HDL blocks are merged into one overall desired design behavior. The lowest level portion of your design is completed first. Only after the low-level building blocks are complete do you finish

higher-level hierarchical blocks in your design. This methodology is typically used with schematic capture programs.

byte-wide PROM

A byte-wide PROM is a byte-wide programmable read-only memory (PROM) supplies data one byte at a time.

CCLK pin

The CCLK pin is the XChecker pin that provides the configuration clock for the device or daisy chain of devices during a download.

CLKI pin

The CLKI pin is the clock input pin to XChecker. CLKI provides an external clock to the Hardware Debugger so that in conjunction with the CLKO pin, the Debugger can control the application of the external clock to the device being debugged.

CLKO pin

The CLKO pin is the XChecker clock output pin. CLKO supplies the Hardware Debugger controlled clock to the device being debugged. The source of CLKO is one of the following: CLKI, logic 1, logic 0, or the internal XChecker clock.

clock input path

A clock input path starts at either an input of the chip or at the output of a flip-flop, latch, or RAM, and ends at any clock pin on a flip-flip or latch enable. The clock input path time is the maximum time required for the signal to arrive at the flip-flop clock input. Clock input paths help to determine system-level design timing.

clock skew

Clock skew is the difference between the time a clock signal arrives at the source flip-flop in a path and the time it arrives at the destination flip-flop. It is also referred to as clock delay.

critical path

A critical path is a signal in a section of combinatorial logic that limits the speed of the logic. Storage elements begin and end a critical path, which may include I/O pads.

daisy chain

A daisy chain is a series of bitstream files concatenated in one file. It can be used to program several FPGAs connected in a daisy chain board configuration.

debugging

Debugging is the process of reading back or probing the states of a configured device to ensure that the device is behaving as expected while in circuit.

DIN pin

In an FPGA, the DIN pin loads a bitstream in serial mode. On the XChecker cable, it provides the bitstream data and connects to the DIN pin of the target FPGA.

DONE pin (Spartan/XC4000/XC5200)

The DONE pin is a dual function pin. As an input, it can be configured to delay the global logic initialization or the enabling of outputs. As an output, it indicates the completion of the configuration process. For Virtex devices, this pin is called DONE_CFG.

downloading

Downloading is the process of configuring or programming a device by sending bitstream data to the device.

D/P pin (XC3000)

The D/P pin is dual-function pin. As an input, it initiates a reconfiguration of a configured device. As an output, it signals the end of configuration.

EDIF

EDIF is an acronym for Electronic Data Interchange Format, an industry standard file format for specifying a design netlist. It is generated by a third-party design-entry tool.

EXORmacs (Motorola)

This is a PROM format supported by the Xilinx tools. Its maximum address is 16 777 216. This format supports PROM files of up to (8 x 16 777 216) = 134 217 728 bits.

external clock

The external clock is the system clock that XChecker uses from the target board during synchronous mode debugging. To use an external clock, connect the system clock to the XChecker cable using the CLKI pin and the XChecker clock to the FPGA device using the CLKO pin.

fitting

Fitting is the process of putting logic from your design into physical macrocell locations in the CPLD. Routing is performed automatically, and because of the UIM architecture, all designs are routable.

GND pin

A GND pin is Ground (0 volts).

group

A group is a collection of common signals to form a bus. In the case of a counter, for example, the different signals that produce the actual counter values can be combined to form an alias, or group.

guide file

A guide file is a previously placed and routed FPGA or fitted CPLD file that can be used in a subsequent place and route or fitting operation.

HDL

HDL is an acronym for Hardware Description Language. The most common HDLs in use today are Verilog and VHDL. They describe designs in a technology-independent manner using a high level of abstraction.

HEX

HEX refers to a simple text dump of the PROM data in HEX format. It has unlimited data capacity.

hold time

Hold time is the time following a clock event during which the data input to a latch or flip-flop must remain stable in order to guarantee that the latched data is correct.

INIT pin

The INIT pin is a device pin indicating when a device is ready to receive configuration data after power-up.

instance

An instance is one specific gate or hierarchical element in a design or netlist. The term “symbol” often describes instances in a schematic drawing. Instances are interconnected by pins and nets. Pins are ports through which connections are made from an instance to a net. A design that is flattened to the lowest level constituents is described using primitive instances.

internal XChecker clock

The internal XChecker clock is internal to XChecker and can be applied by the XChecker CLKO pin to a device being debugged.

IOB (input/output block)

An IOB is a collection or grouping of basic elements that implement the input and output functions of an FPGA device.

.ll file

The .ll file is the logic allocation file, which indicates the bitstream position of storage elements such as latches, flip-flops, and IOB inputs

and outputs. The Hardware Debugger uses this file to locate signal values inside a readback bitstream.

loading direction

Loading direction is the direction of the addresses in which data is stored on your PROM. In the Up direction, the data is stored in ascending order. In the Down direction, the data is stored in descending order.

logic icon

A logic icon is a graphical representation of a logic resource, such as a flip-flop, buffer, or register.

logic synthesis

Logic synthesis is a process that starts from a high level of logic abstraction (typically Verilog or VHDL) and automatically creates a lower level of logic abstraction using a library containing primitives.

mapping

Mapping is the process of assigning a design's logic elements to the specific physical elements that actually implement logic functions in a device.

MCS-86 (Intel)

MCS-86 is a PROM format supported by the Xilinx tools. Its maximum address is 1 048 576. This format supports PROM files of up to (8 x 1 048 576) = 8 388 608 bits.

net

A net is a logical connection between two or more symbol instance pins. After routing, the abstract concept of a net is transformed to a physical connection called a wire.

number of clock cycles

The number of clock cycles is the number of clocks that occur between snapshots during synchronous mode debugging. When capturing multiple snapshots, the number of snapshots is used as a trigger for capturing each snapshot.

one-to-one logic

In the context of Xilinx FPGA devices, one-to-one logic is the exact correspondence between the logic specified in the design entry phase and the logic implemented in the device. For example, if you draw three inverters in your design, there are three corresponding inverters in the programmed device. This correspondence makes back-annotation of timing delays very straightforward and ensures that there are no differences between your original design and the finished device.

optimization

Optimization is the process that decreases the area or increases the speed of a design.

pad

A pad is the physical bonding pad on an integrated circuit. All signals on a chip must enter and leave by way of a pad. Pads are connected to package pins in order for signals to enter or leave an integrated circuit package.

pin

A pin can be a symbol pin or package pin. A package pin is a physical connector on an integrated circuit package that carries signals into and out of an integrated circuit. A symbol pin, also referred to as an instance pin, is the connection point of an instance to a net.

place effort

Place effort is a user-controlled parameter that balances run-time with placement efficiency for the Flow Engine.

placer

The placer is a utility that maps logic from your design into specific locations in the target FPGA chip.

placing

Placing is the process of assigning physical device cell locations to the logic in a design.

primitive

A primitive is a logic element that directly corresponds, or maps, to a basic silicon component.

probing

Probing is the process of examining the states of an FPGA device.

PROG pin

A PROG pin is an XChecker pin that provides a reprogram pulse to XC4000, XC5200, and Virtex devices when connected to the PROG pin of the device.

programming

Programming is the process of configuring the programmable interconnect in the FPGA.

PROM

PROM is an acronym for programmable read-only memory.

PROM file

A PROM file consists of one or more BIT files (bitstreams) formed into one or more datastreams. The file is formatted in one of four industry-standard formats: Intel MCS-86, Tektronics TEKHEX, Motorola EXORmacs, or HEX. The PROM file includes headers that specify the length of the bitstreams, as well as all the framing and control information necessary to configure the FPGAs. It can be used to program one or more devices.

RBT file

An RBT file is a raw BIT format file; the ASCII version of the BIT file.

RD pin

The RD pin is the XChecker readback data pin.

readback

Readback is the process of reading the logic downloaded to an FPGA device. There are two types of readbacks.

- A readback with a filter that extracts the configuration bits to verify that a design was downloaded in its entirety.
- A readback with a filter that extracts the state of design storage elements, CLB outputs, and IOB outputs to ensure that the device is behaving as expected.

route effort

Route effort is the user-controlled parameter that balances run-time with routing efficiency for the Flow Engine.

router

The router connects all appropriate pins to create the design's nets.

routing

Routing is the process of assigning logical nets to physical wire segments in the FPGA that interconnect logic cells.

RPM

A Relationally Placed Macro (RPM) defines the spatial relationship of the primitives that constitute its logic. An indivisible block of logic elements that are placed as a unit into a design.

RST pin

An RST pin is an XChecker pin that can be driven Low after configuration to reset the target FPGA internal latches and flip-flops.

RT pin

The RT pin is the XChecker readback trigger pin.

schematic

A schematic is a hierarchical drawing representing a design in terms of user and library components.

script

A script is a series of commands that automatically execute a complex operation such as the steps in a design flow.

SDF

Standard Delay Format (SDF) is an industry-standard file format for specifying timing information. It is usually used for simulation.

serial PROM

A serial PROM is a PROM that is read one bit at a time.

setup time

Setup time is the time prior to a clock event during which the data input to a latch or flip-flop must remain stable in order to guarantee that the latched data is correct.

snapshot

A snapshot is the readback data that contains the values of all storage elements, CLB outputs, and IOB inputs and outputs of a design at a point in time.

state

A state is a set of values stored in the memory elements of a device (flip-flops, latches, RAMs, CLB outputs, and IOBs) that represent the state of that device for a particular readback. To each state there corresponds a specific set of logical values.

static timing analysis

Static timing analysis is a point-to-point delay analysis of a design network.

status bar

The status bar is an area located at the bottom of a window that provides information about the commands that you are about to select or that are being processed.

synchronous debugging

Synchronous debugging is a debugging mode in which you use the XChecker cable to have full control of the clock.

synthesis

See logic synthesis.

TCK pin

A TCK pin is an XChecker pin. This output supplies clocks for a boundary scan port on an XC9500 device. The JTAG software must be used to drive the boundary scan port on the XChecker cable.

TDI pin

A TDI pin is an XChecker pin. This input receives data from the boundary scan chain. The JTAG software must be used to drive the boundary scan port on the XChecker cable.

TEKHEX (Tektronix)

TEKHEX is a PROM format supported by Xilinx. Its maximum address is 65 536. This format supports PROM files of up to (8 x 65 536) = 524 288 bits.

timing

Timing is the process that calculates the delays associated with each of the routed nets in the design.

timing constraints

Timing constraints are user specifications of the maximum allowable delay on any given set of paths in a design. Timing constraints can be entered on a schematic or in a user constraints file (UCF).

TMS pin

A TMS pin is an XChecker pin. This output drives the mode of the boundary scan state machine. The JTAG software must be used to drive the boundary scan port on the XChecker cable.

toolbar

The toolbar is a field located under the menu bar at the top of a window. It contains a series of buttons that you click to execute some of the most commonly used commands. These buttons are an alternative to the menu commands.

toolbox

The toolbox is a field located in the Design Manager main window. It contains a series of buttons that you click to invoke tools such as the Flow Engine, Timing Analyzer, Floorplanner, Hardware Debugger, PROM File Formatter, FPGA Editor, Chip Viewer, and JTAG Programmer.

top-down design

Top-down design starts a design with the highest level of abstraction and gradually designs underlying blocks until the complete design is implemented in the target technology. Top-down design is often technology-independent at the highest levels of design abstraction.

TRIG pin

A TRIG pin is an XChecker external trigger pin that causes the Hardware Debugger to initiate a readback of the device being debugged.

trigger

A trigger is an external signal that tells the Hardware Debugger to start the readback operation. It applies the clock to the bitstream.

TTY

TTY is a textual command line interface.

universal interconnect matrix (UIM)

The UIM is the routing matrix for CPLD devices. This fully populated switching matrix allows any output to be routed to any input, guaranteeing 100% routability of all designs. The UIM can also function as a very wide AND gate, which can allow more logic to be placed in macrocells.

VCC pin

The VCC pin is Power (5 volts). It is the supply voltage.

verification

Verification is the process of reading back the configuration data of a device and comparing it to the original design to ensure that all of the design was correctly received by the device.

Verilog

Verilog is a commonly used Hardware Description Language (HDL) that can be used to model a digital system at many levels of abstraction ranging from the algorithmic level to the gate level. It is IEEE standard 1364-1995. Verilog was originally developed by Cadence Design Systems and is now maintained by OVI.
A Verilog file has a .v extension.

VHDL

VHDL is an acronym for VHSIC Hardware Description Language (VHSIC an acronym for Very High-Speed Integrated Circuits). It can be used to describe the concurrent and sequential behavior of a digital system at many levels of abstraction ranging from the algorithmic level to the gate level. VHDL is IEEE standard 1076-1993.
A VHDL file has a .vhd or .vhdl extension.

waveform

A waveform is a graphical representation of a set of simulation transitions that depicts the digital or electrical values of a node on the schematic.

WIR file

Viewlogic netlist files built by ViewDraw, PROcapture, or ViewSynthesis.
A WIR file is an intermediate design file generated by the Viewlogic design tools.

workspace

In the PROM File Formatter, the workspace is a frame and an empty datastream. When you add files into the datastream, the horizontal arrows indicate the concatenation of files.