Vivado Design Suite User Guide

Using the Vivado IDE

UG893 (v2014.1) April 30, 2014

XILINX
ALL PROGRAMMABLE™

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
04/30/2014	2014.1	Removed reference to XPS throughout. Removed reference to XilinxNotify in Launching the Vivado Design Suite, updated Using the Getting Started Page, and added Adding Design Tools or Devices in Chapter 1, Introduction. Added link to Vivado Design Suite QuickTake Video Tutorials in the Overview, updated showing and resizing the Flow Navigator in Flow Navigator, added information about get_* Tcl command in Finding Unplaced BUFGs, and added Finding Black Box Cells in Chapter 2, Using the Viewing Environment. Added Hierarchy View Icons, added Using the Language Templates Window, updated Using Mouse Strokes to Zoom and Pan, added information on toggling layers to Layers, added information on the Timing Constraints wizard to Using the Timing Constraints Window, added links to the Vivado Design Suite QuickTake Video: Understanding Messaging and Vivado Design Suite QuickTake Video: Messages, Reports, and Log Files Overview, and updated commands throughout in Chapter 3, Using Windows. Updated options in Specifying General Options, added Adjusting Fonts, and moved Creating Custom View Layouts in Chapter 4, Configuring the Environment. Updated Table A-5 and Table A-6 in Appendix A, Input and Output Files. Updated Figure 1-2, Figure 2-1, Figure 2-3, Figure 2-6, Figure 2-7, Figure 2-16, Figure 2-19, Figure 3-29, Figure 3-35, Figure 3-36, Figure 3-38, Figure 3-39, Figure 3-40, Figure 4-2, Figure 4-3, Figure 4-4, Figure 4-5, Figure 4-8, Figure 4-10, and Figure 4-13.

Table of Contents

Introduction

Overview

The Vivado® Integrated Design Environment (IDE) provides an intuitive graphical user interface (GUI) with powerful features. All of the tools and tool options are written in native Tool Command Language (Tcl) format, which enables use both in the Vivado IDE or Vivado Design Suite Tcl shell. Analysis and constraint assignment is enabled throughout the entire design process. For example, you can run timing or power estimations after synthesis, placement, or routing. Because the database is accessible through Tcl, changes to constraints, design configuration or tool settings happen in real time, often without forcing re-implementation.

You can improve design performance using the new algorithms delivered by the Vivado IDE, including:

- Register transfer level (RTL) design in VHDL, Verilog, and SystemVerilog
- Intellectual property (IP) integration for cores
- Behavioral simulation with Vivado simulator
- Vivado synthesis
- Vivado implementation for place and route
- Vivado serial I/O and logic analyzer for debugging
- Vivado power analysis
- SDC-based Xilinx® Design Constraints (XDC) for timing constraints entry
- Static timing analysis
- High-level floorplanning
- Detailed placement and routing modification
- Bitstream generation

The Vivado IDE uses a concept of opening designs in memory. Opening a design loads the design netlist at that particular stage of the design flow, assigns the constraints to the design, and then applies the design to the target device. This provides the ability to visualize and interact with the design at each design stage. You can experiment with

different implementation options, refine timing constraints, explore the Vivado IP catalog, perform simulation, and apply physical constraints with floorplanning techniques to help improve design results. Early estimates of resource utilization, interconnect delay, power consumption, and routing connectivity can assist with appropriate logic design, device selection, and floorplanning. As the design moves through the implementation flow you can further refine the inputs.

IMPORTANT: *The Vivado IDE supports designs that target 7 series, Zynq®-7000 All Programmable, and UltraScale™ devices only.*

Project Mode and Non-Project Mode

There are two design flow modes available in the Vivado Design Suite: Project Mode and Non-Project Mode. In general, you run Project Mode in the Vivado IDE. In this mode, you create a project in the Vivado IDE, and the Vivado IDE automatically saves the state of the design, generates reports and messaging, and manages source files. In general, you run Non-Project Mode using Tcl commands or scripts. In this mode, you have full control of the design flow, but the Vivado tools do not automatically manage source files or report the design state. However, in Non-Project Mode, you can open the Vivado IDE at each design stage for design analysis and constraints assignment. You are viewing the active design in memory, so any changes are automatically passed forward in the flow. You can save updates to new constraint files or design checkpoints. For more information, see the *Vivado Design Suite User Guide: Design Flows Overview* (UG892) [Ref 1].

In Project Mode, the Vivado IDE supports several features that are not available in Non-Project Mode:

- Source file management and status
- Flow Navigator and Project Summary
- Consolidated Messages and automatically generated standard reports
- Cross probing back to RTL
- Storage of tool settings and design configuration
- Experimentation with multiple synthesis and implementation runs
- Use and management of constraint sets
- Run results management and status
- IP configuration and integration with the IP catalog

These features provide several advantages from an ease-of-use perspective. For example, when opening a previously created project in the Vivado IDE, you see the current state of the design, run results, and previously generated reports and messages. Using the Flow Navigator, a single click on **Generate Bitstream** synthesizes and implements the design, and generates a bitstream file. In addition, you can cross probe from an error message directly to the source file.

Launching the Vivado Design Suite

You can launch the Vivado Design Suite and run the tools using different methods depending on your preference. For example, you can choose a Tcl script-based compilation style method in which you manage sources and the design process yourself, also known as Non-Project Mode. Alternatively, you can use a project-based method to automatically manage your design process and design data using projects and project states, also known as Project Mode. Either of these methods can be run using a Tcl scripted batch mode or run interactively in the Vivado IDE. For more information on the different design flow modes, see the *Vivado Design Suite User Guide: Design Flows Overview* (UG892) [Ref 1].

Note: Installation, licensing, and release information is available in the *Vivado Design Suite User Guide: Release Notes, Installation, and Licensing* (UG973) [Ref 2].

Working with the Vivado IDE

You can launch the Vivado IDE from Windows or Linux.

RECOMMENDED: *You can open the Vivado IDE from any directory. However, running it from a project directory is recommended, because the Vivado IDE log and journal files are written to the launch directory. When running from a command prompt, launch the Vivado IDE from the project directory, or use the vivado* `-log` *and* `-journal` *options to specify a location. When using a Windows shortcut, you must modify the* ***Start in*** *folder, which is a* ***Property*** *of the shortcut.*

Launching the Vivado IDE on Windows

Select **Start > All Programs > Xilinx Design Tools > Vivado 2014.x > Vivado 2014.x**.

Note: You can also double-click the Vivado IDE shortcut icon on your desktop.

Figure 1-1: **Vivado Desktop Icon**

TIP: *On Windows 7 systems, you can right-click the Vivado IDE shortcut icon, and select **Pin to Start Menu** or **Pin to Taskbar** to provide quick access to the Vivado IDE.*

Launching the Vivado IDE from the Command Line on Windows or Linux

Enter the following command at the command prompt:

```
vivado
```

Note: When you enter this command, it automatically runs `vivado -mode gui` to launch the Vivado IDE. If you need help, type `vivado -help`.

Launching the Vivado IDE from the Vivado Design Suite Tcl Shell

Enter the following command at the Tcl command prompt:

```
start_gui
```

Working with Tcl

If you prefer to work directly with Tcl, you can interact with your design using Tcl commands using either of the following methods:

- Enter individual Tcl commands in the Vivado Design Suite Tcl shell outside of the Vivado IDE.
- Run Tcl scripts from the Vivado Design Suite Tcl shell.
- Enter individual Tcl commands in the Tcl Console at the bottom of the Vivado IDE.
- Run Tcl scripts from the Vivado IDE.

For more information about using Tcl and Tcl scripting, see the *Vivado Design Suite User Guide: Using Tcl Scripting* (UG894) [Ref 4] and *Vivado Design Suite Tcl Command Reference Guide* (UG835) [Ref 5]. For a step-by-step tutorial that shows how to use Tcl in the Vivado tools, see the *Vivado Design Suite Tutorial: Design Flows Overview* (UG888) [Ref 6].

Note: Alternatively, you can type `<command_name> -help` in the Tcl Console or at the Vivado Design Suite Tcl shell for information about the specified command.

Launching the Vivado Design Suite Tcl Shell

Use the following command to invoke the Vivado Design Suite Tcl shell either at the Linux command prompt or within a Windows Command Prompt window:

```
vivado -mode tcl
```

Note: On Windows, you can also select **Start > All Programs > Xilinx Design Tools > Vivado 2014.x > Vivado 2014.x Tcl Shell**.

Launching the Vivado Tools Using a Batch Tcl Script

You can use the Vivado tools in batch mode by supplying a Tcl script when invoking the tool. Use the following command either at the Linux command prompt or within a Windows Command Prompt window:

```
vivado -mode batch -source <your_Tcl_script>
```

Note: When working in batch mode, the Vivado tools exit after running the specified script.

Using the Tcl Console in the Vivado IDE

The Vivado IDE runs on top of a powerful engine with integrated Tcl support. In the Vivado IDE, you can issue Tcl commands from the Tcl Console, as described in Using the Tcl Console in Chapter 3.

Running a Tcl Script from the Vivado IDE

To run a script, select **Tools > Run Tcl Script**.

TIP: *To create scripts, you can copy the Vivado IDE Tcl commands from the `vivado.jou` file or from the Tcl Console.*

Using the Getting Started Page

When you open the Vivado IDE, the Getting Started Page appears (Figure 1-2).

Note: To open the Getting Started Page, all open projects must be closed.

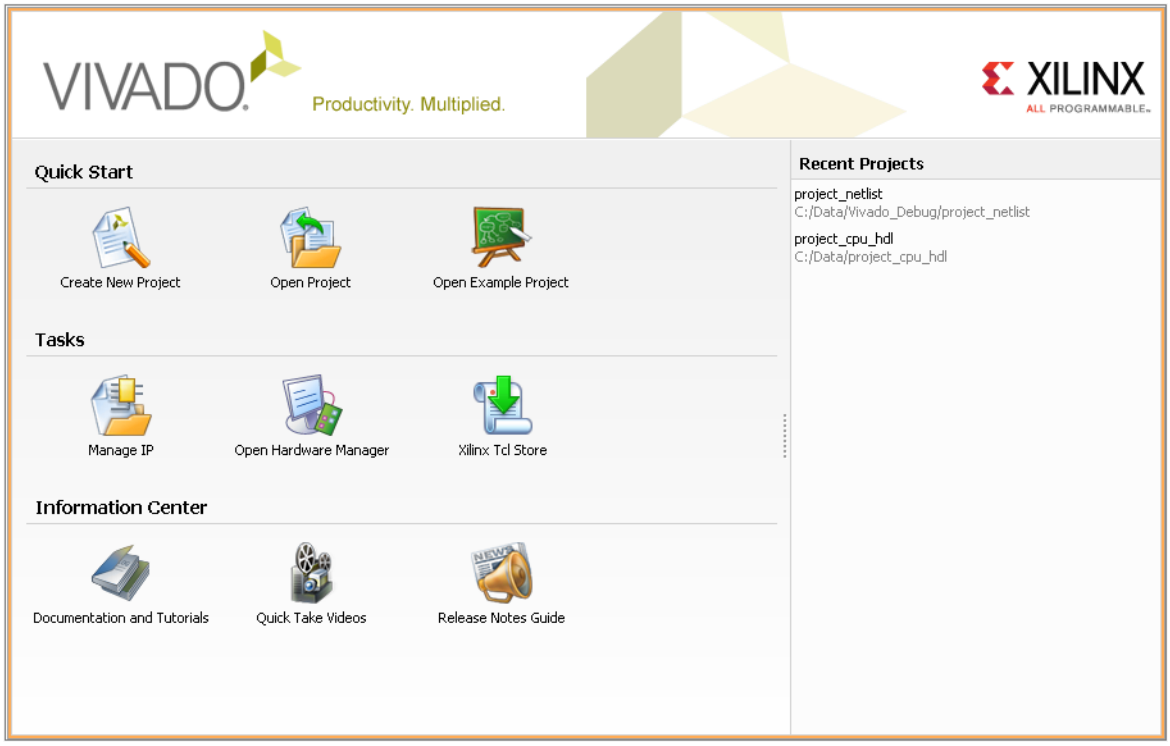


Figure 1-2: **Vivado IDE Getting Started Page**

The Vivado IDE Getting Started Page assists you with creating and opening projects, running Vivado IDE commands, and viewing documentation as follows:

- **Quick Start**
  - **Create New Project**: Opens the New Project wizard to guide you through creating various supported project types. You can also use the wizard to import previously created projects from the PlanAhead™ tool (.ppr extension) or from the ISE® Design Suite (.xise extension).
  - **Open Project**: Opens a browser that enables you to open any Vivado IDE project file (.xpr extension). It also displays the last ten previously-opened projects. Ten is the default. To change this number, use **Tools > Options** to update the General options. The Vivado IDE checks to ensure that the project data is available before displaying the projects.

- **Open Example Project**: Shows the following example projects categorized by installation configuration and device family. Select an example project that is valid for your installation:
  - **BFT Core**: Small RTL project.
  - **CPU (HDL)**: Large, mixed language RTL project.
  - **CPU (Synthesized)**: Large synthesized netlist project.
  - **Wave (HDL)**: Small project that includes three embedded IP cores. You can use this design to learn how to use integrated IP cores with Vivado IDE projects.
  - **Zynq® System**: Small Vivado IP integrator Zynq device design targeting the ZC702 evaluation board. The design walks you through bitstream generation in the Vivado Design Suite and application code development in Software Development Kit (SDK) for the Zynq device.
  - **Microblaze™ System:** Small Vivado IP integrator Microblaze processor design targeting the KC705 board. The design walks you through bitstream generation in the Vivado Design Suite, application code development in SDK, and simulation of the design in the Vivado Design Suite using an ELF file generated by SDK.

- **Tasks**
  - **Manage IP**: Opens or creates an IP project for customizing and managing IP. The Vivado IP catalog displays Xilinx, third-party, or user-created IP, which can be customized to create IP cores for a specified FPGA device. You can also view or re-customize existing IP cores and generate output products, including a netlist of the IP standalone.
  - **Open Hardware Manager**: Opens the Vivado Design Suite Hardware Manager to connect to a target JTAG cable or board, which enables you to program your design into a device. The Vivado logic analyzer and Vivado serial I/O analyzer features of the tool enable you to debug your design.
  - **Xilinx Tcl Store**: Opens the Xilinx Tcl Store, an open source repository of Tcl code designed primarily for use in FPGA designs with the Vivado Design Suite. The Tcl Store provides access to multiple scripts and utilities contributed from different sources, which solve various issues and improve productivity. You can install Tcl scripts and also contribute Tcl scripts to share your expertise with others. For more information, see the *Vivado Design Suite User Guide: Using Tcl Scripting* (UG894) [Ref 4].

- **Information Center**
  - **Documentation and Tutorials**: Opens or downloads Vivado IDE documentation using the Xilinx Documentation Navigator or your default web browser.
  - **Quick Take Videos**: Opens Xilinx video tutorials.
  - **Release Notes Guide**: Opens the *Vivado Design Suite User Guide: Release Notes, Installation, and Licensing* (UG973) [Ref 2].

  Note: For more information about the Xilinx Documentation Navigator, see the *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 3].
- **Recent Project and Recent Checkpoints**: Provides one-click access to recently opened projects or checkpoints. These lists only appear after you open projects or checkpoints.

Adding Design Tools or Devices

If you want to add a design tool or device that you did not initially install, you can select **Help > Add Design Tools or Devices**. This command launches the Xilinx installer, which allows you to modify your installation options. For more information, see the *Vivado Design Suite User Guide: Release Notes, Installation, and Licensing* (UG973) [Ref 2].

Using the Viewing Environment

Overview

This chapter contains general information on the terminology, layout, and project features of the Vivado® IDE. It does not contain information on the Vivado IDE design flow. For information on the design flow, see the following documents:

- *Vivado Design Suite User Guide: Design Flows Overview* (UG892) [Ref 1]
- *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 7]
- *Vivado Design Suite User Guide: Designing IP Subsystems Using IP Integrator* (UG994) [Ref 8]
- *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 9]
- *Vivado Design Suite User Guide: Synthesis* (UG901) [Ref 10]
- *Vivado Design Suite User Guide: Using Constraints* (UG903) [Ref 11]
- *Vivado Design Suite User Guide: Implementation* (UG904) [Ref 12]
- *Vivado Design Suite User Guide: Design Analysis and Closure Techniques* (UG906) [Ref 13]
- *Vivado Design Suite User Guide: Programming and Debugging* (UG908) [Ref 14]

VIDEO: *For more information on tool usage and features, see the Vivado Design Suite QuickTake Video Tutorials. These video tutorials target specific topics in brief video presentations.*

Vivado IDE Viewing Environment

Figure 2-1 shows the Vivado IDE viewing environment. You can interact with the Vivado IDE through mouse, keyboard, or Tcl input.

The main components of the viewing environment are:

1. Menu Bar
2. Main Toolbar
3. Flow Navigator
4. Layout Selector
5. Data Windows Area
6. Workspace
7. Menu Command Search Field
8. Project Status Bar
9. Status Bar
10. Results Windows Area

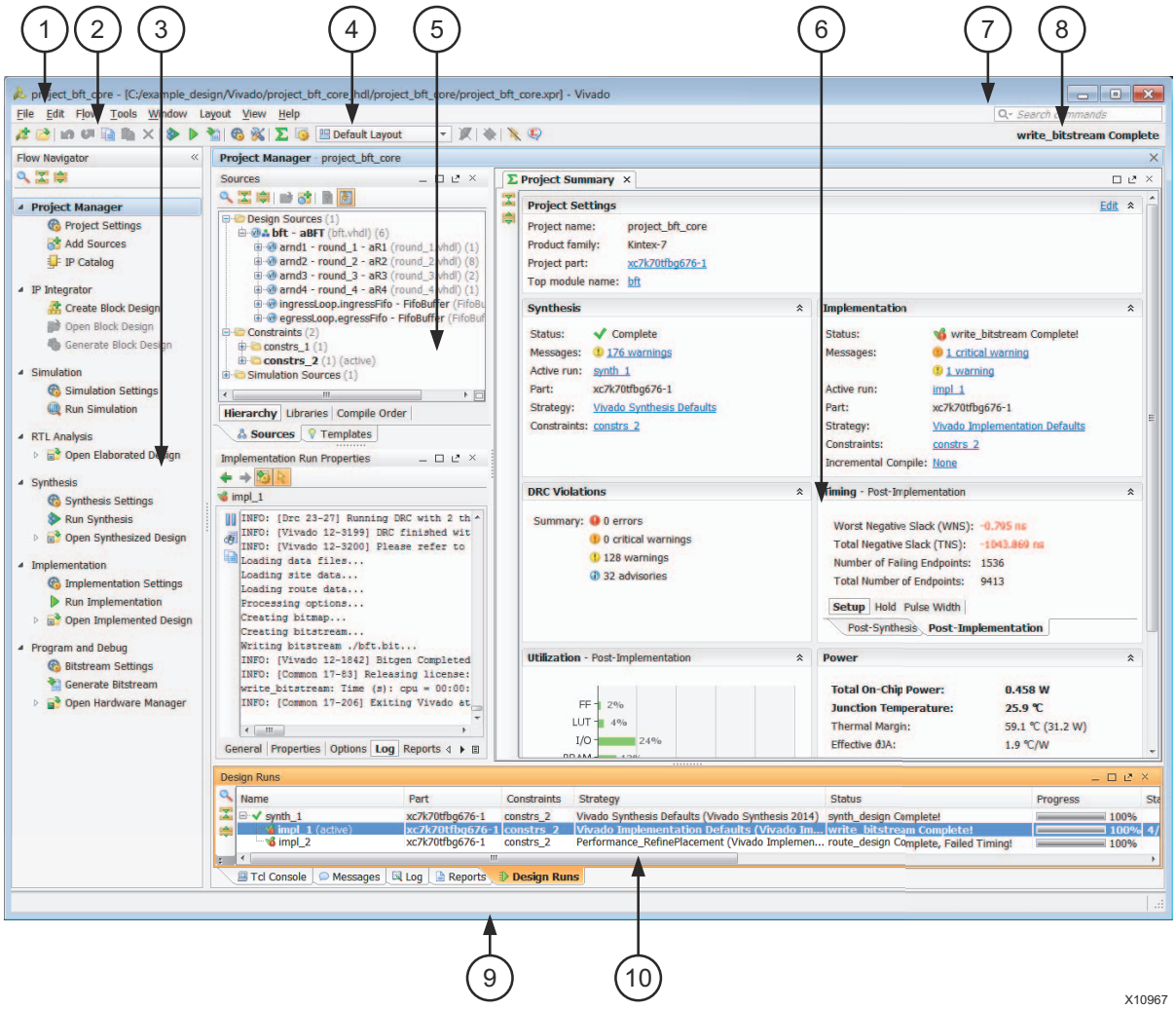


Figure 2-1: **Vivado IDE Viewing Environment**

Menu Bar

The main menu bar provides access to Vivado IDE commands. Commonly-used commands always display (for example, **File > Open Project**) while others display only when a design is active (for example, **Tools > Report > Report DRC**). Some menu commands have a related keyboard shortcut that is listed next to the menu command. For information on defining your own keyboard shortcuts, see Configuring Shortcut Keys in Chapter 4.

Menu Command Search Field

To the right of the menu commands, the menu command search field enables you to quickly locate and execute a command from the menu bar. In the search field, type a few letters of a command name. A list of commands that match your search criteria appear. Select a command from the list to execute that command.

The search mechanism uses a wildcard search for the string of characters you typed in the search field. For example, as shown in Figure 2-2, typing the letters "cl" finds the following commands: Clear List, Clear Placement, Clock Regions, and so forth.

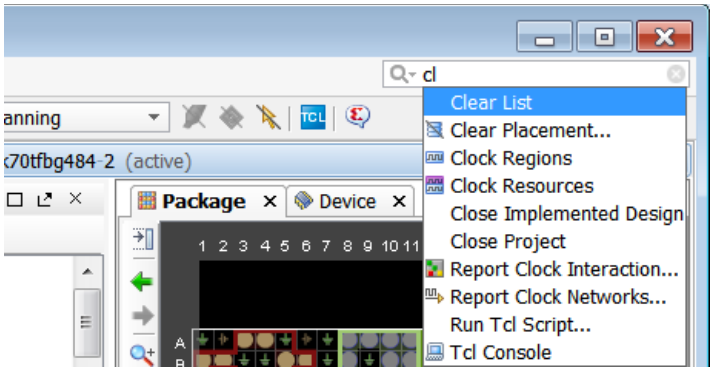


Figure 2-2: **Menu Command Search Field**

The list of commands that appear is based on the current design context in the project. For example, an open elaborated design offers a different set of commands than an open implemented design.

Note: In addition to menu commands, the command search field reports project names and files that display under the **Open Recent Project** and **Open Example Project** commands in the **File** menu.

Main Toolbar

The main toolbar provides one-click access to the most commonly used commands in the Vivado IDE. When you hover the mouse cursor over a button, a tooltip appears that provides more information about the command.

TIP: *You can set the amount of time before a tooltip appears as well as the amount of time before it disappears. You can also set whether to show tooltips for menu commands. Select* ***Tools*** *>* ***Options****. In the Vivado Options dialog box, click the* ***General*** *category, and set the Language and Tooltip options.*

Flow Navigator

The Flow Navigator provides access to commands and tools to take your design from design entry to bitstream creation. As you run these commands and tools, the design data, graphical windows, and results windows update. The different sections in the Flow Navigator enable you to do the following:

- **Project Manager**: Change general settings, add or create sources, and open the Vivado IP catalog. For information on adding sources, see the *Vivado Design Suite User Guide: System-Level Design Entry* (UG895) [Ref 15]. For information on the IP catalog, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 7].
- **IP Integrator**: Create, open, or generate a block design. For information on the Vivado IP integrator, see the *Vivado Design Suite User Guide: Designing IP Subsystems Using IP Integrator* (UG994) [Ref 8].
- **Simulation**: Change simulation settings or simulate the active design. For information on simulation, see the *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 9].
- **RTL Analysis**: Open an elaborated design, run design rule checks (DRCs), and generate an RTL schematic. For information on the Schematic window, see Using the Schematic Window in Chapter 3. For information on elaborating the RTL design, see the *Vivado Design Suite User Guide: System-Level Design Entry* (UG895) [Ref 15].
- **Synthesis**: Change synthesis settings, synthesize the active design, or open the synthesized design. You can right-click **Open Synthesized Design**, and select **New Synthesized Design** to load a second design. You can also right-click and select **Open Netlist in New Window** to compare the designs side by side. For information on synthesis, see the *Vivado Design Suite User Guide: Synthesis* (UG901) [Ref 10].
- **Implementation**: Change implementation settings, implement the active design, or open the implemented design. For information on implementation, see *Vivado Design Suite User Guide: Implementation* (UG904) [Ref 12].
- **Program and Debug**: Change bitstream settings, generate a bitstream file, open a hardware session in the Vivado IDE, and launch the Vivado logic analyzer. For information on programming and debugging, see *Vivado Design Suite User Guide: Programming and Debugging* (UG908) [Ref 14].

TIP: *Right-click a Run command to see available commands. For information on run management, see Design Runs Window Commands in Chapter 3.*

After opening a design, the section displays in bold to show that a design is loaded into memory. In addition, the Open command changes. For example, **Open Synthesized Design** changes to **Synthesized Design**. If you have multiple designs loaded, you can click the sections in the Flow Navigator to switch between designs (for example, **RTL Analysis** or **Synthesis**). For more information on comparing multiple runs, see the *Vivado Design Suite User Guide: Design Analysis and Closure Techniques* (UG906) [Ref 13].

To make more screen space available for other windows during design analysis, you can hide the Flow Navigator as follows:

- Select **View > Hide Flow Navigator**.
- Use the **Ctrl+Q** keyboard shortcut.
- In the upper right corner of the Flow Navigator, click the Hide Navigator button **<<** shown in Figure 2-3.

Note: To show the Flow Navigator, select **View > Show Flow Navigator**, press **Ctrl+Q**, or click the Show Flow Navigator **>>** button on the left edge of the Vivado IDE. To resize the Flow Navigator, use the window slider shown in Figure 2-3.

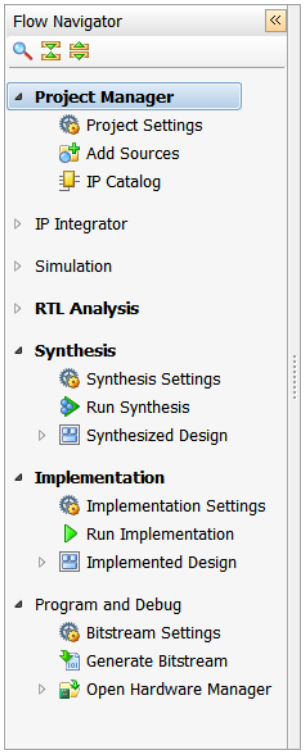


Figure 2-3: **Hide Flow Navigator Button and Resize Window Slider**

Layout Selector

The Vivado IDE provides predefined window layouts to facilitate various tasks in the design process. The layout selector (Figure 2-4) enables you to easily change window layouts. Alternatively, you can change layouts using the **Layout** menu in the menu bar.

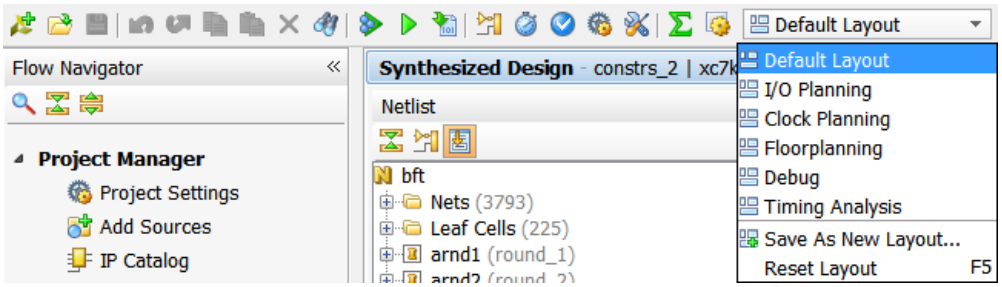


Figure 2-4: **Layout Selector**

Use the predefined layouts as follows:

- **Default Layout**: Analyze your design with a minimum set of windows.
- **I/O Planning**: Define I/O placement constraints and place ports.
- **Clock Planning**: Cross probe between the Clock Resources window, Device window, and I/O Port window to plan and place clock resources in the design.
- **Floorplanning**: Define Pblocks, manage partitions, and perform hierarchical floorplanning.
- **Debug**: Define debug nets and configure debug cores.
- **Timing Analysis:** Run timing reports and analyze timing.

TIP: *You can also create custom view layouts that meet your specific requirements as described in Creating Custom View Layouts in Chapter 4.*

Project Status Bar

The project status bar displays the current status of the active design.

TIP: *When one or more designs become out-of-date, a **More Info** link appears in the project status bar. Click the link to view information about the changes that caused the design to become out-of-date.*

Data Windows Area

By default, this area of the Vivado IDE displays information related to design sources and data, such as:

- **Sources window**: Displays the Hierarchy, IP Sources, Libraries, and Compile Order views.
- **Netlist window**: Provides a hierarchical view of the elaborated or synthesized logic design.
- **Properties window**: Displays information about selected logic objects or device resources.

For more information, see Chapter 3, Using Windows.

Workspace

The workspace displays windows with a graphical interface and those that require more screen space, including:

- Text Editor for displaying and editing text-based files and reports
- Schematic window
- Device window
- Package window

For more information, see Chapter 3, Using Windows.

Results Windows Area

The status and results of commands run in the Vivado IDE display in the results windows area, a set of windows grouped at the bottom of the viewing environment. As commands are run, messages are generated, and log files and report files are created, the related information appears in this area. By default, this area includes the following windows:

- **Tcl Console**: Allows you to enter Tcl commands, and view the history of previous commands and output.
- **Messages**: Shows all messages for the current design, categorized by process and severity.
- **Log**: Shows the log files created by the synthesis, implementation and simulation runs.
- **Reports**: Provides quick access to the reports generated throughout the design flow for the active run.
- **Designs Runs**: Manages runs for the current project.

The Find Results, Package Pins, and I/O Ports windows as well as various reports appear in this area as needed. For more information, see Chapter 3, Using Windows.

Status Bar

The status bar displays a variety of information:

- Detailed descriptions for menu and toolbar commands appear on the lower left side of the status bar when accessing the command.
- During placement and constraint creation in the Device and Package windows, constraint type and validity appear on the left side of the status bar. Site coordinates and type display on the right side.
- Hover over an object in the Schematic window, and the object details appear in the status bar.
- Selecting the **Background** button on a running task pushes the task progress bar down to the right side of the status bar.

IMPORTANT: *Any operation that uses Tcl is blocked while a task is running in the background. You can still view reports or view an open design, but you cannot make modifications.*

TIP: *To display the total memory heap size and amount used by the Vivado IDE, double-click the drag handle in the status bar. By default, memory cleanup occurs automatically, but you can click the trash can button*  *to force a memory cleanup.*

Creating Projects

You can use the New Project wizard to easily create a variety of projects in the Vivado IDE. To open the New Project wizard, select **File > New Project**. This wizard enables you to specify a project location and name and create the types of projects shown in Figure 2-5. As you proceed through the wizard, you optionally specify sources, IP, and constraint files, followed by a Xilinx® board or part to complete project creation. For more information, see the *Vivado Design Suite User Guide: System-Level Design Entry* (UG895) [Ref 15].

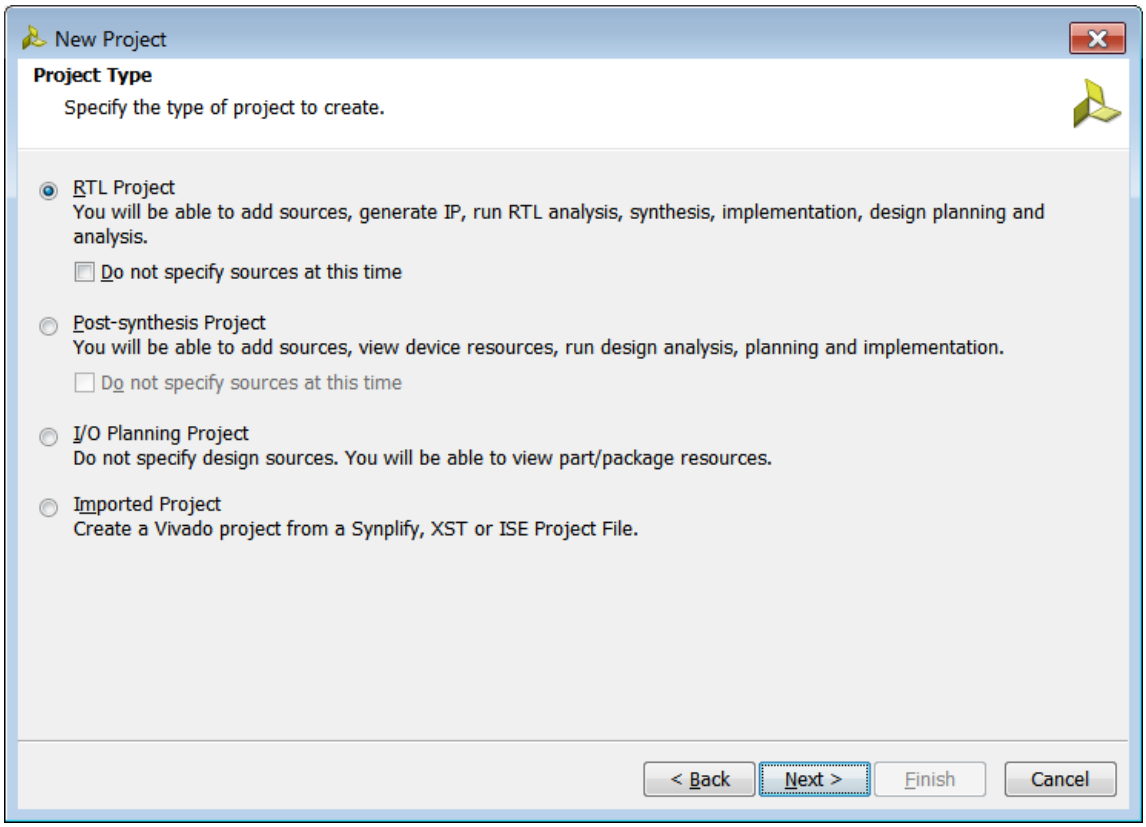


Figure 2-5: **New Project Wizard—Project Type Page**

You can also create a project using Tcl commands. Enter the commands in the Tcl Console of the Vivado IDE or source them from a Tcl file:

```
create_project project_Name ./exampleDesigns/project_8 -part xc7vx485tffg1157-1
```

The default project type is RTL. If you want to create a netlist project specify:

```
set_property design_mode GateLvl [current_fileset]
```

You can now add files to the project:

```
add_files -norecurse -scan_for_includes ./designs/oneFlop.v
```

You can also make them local to the project:

```
import_files -norecurse ./designs/oneFlop.v
```

Note: This command corresponds to the **Copy Sources into Project** option in the Add Sources wizard.

For more information on creating a project using Tcl, see the following documents:

- *Vivado Design Suite User Guide: Design Flows Overview* (UG892) [Ref 1]
- *Vivado Design Suite User Guide: Using Tcl Scripting* (UG894) [Ref 4]

Configuring Project Settings

Use the Project Settings dialog box (Figure 2-6) to configure settings to meet specific needs. These settings include general settings, related to the top module definition and language options, as well as simulation, synthesis, implementation, bitstream, and Vivado IP catalog settings.

TIP: *When entering or modifying data in a text box, if a value is used and editable, the text is black and the background is white. If a value is used but not editable, the text is black and the background is gray. If a value is unused or not applicable, the text is gray, including the label that precedes or follows it.*

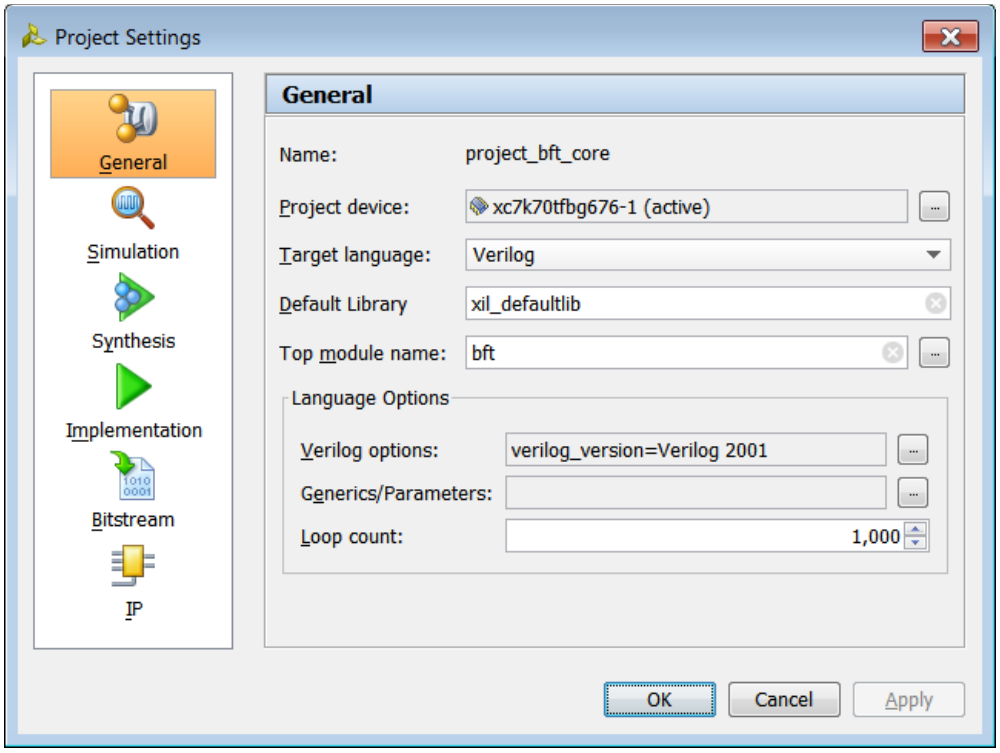


Figure 2-6: **Project Settings Dialog Box**

The Vivado IDE provides access to the project settings from a variety of windows and menus. Based upon where you invoke the project settings, the dialog box appears with the appropriate pane displayed. To open the Project Settings dialog box, use any of the following methods:

- In the Flow Navigator Project Manager section, click **Project Settings**.
- Select **Tools** > **Project Settings**.
- Click the **Project Settings** toolbar button .

- In the Flow Navigator, click **Project Settings** in the Project Manager section, or click one of the following: **Simulation Settings**, **Synthesis Settings**, **Implementation Settings**, or **Bitstream Settings**.
- In the Project Summary, click the **Edit** link next to Project Settings.

The Project Settings dialog box opens with the following categories on the left side:

- **General**: Shows the project name and enables you to change the part, specify the top module name, and set language options. For more information, see the *Vivado Design Suite User Guide: System-Level Design Entry* (UG895) [Ref 15].
- **Simulation**: Enables you to specify the target simulator, including the Vivado simulator or the Mentor Graphics® ModelSim or QuestaSim tool. Displays the simulation set, the simulation top module name, top module (design under test), and a tabbed listing of compilation, simulation, netlist, and advanced options. For more information, see the *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 9].
- **Synthesis**: Shows the default constraints set. It also provides an options area for selecting a synthesis strategy and for setting synthesis command line options. The command line options are defined by the selected synthesis strategy, but you can override these with your own selections. A description of the selected command line option displays at the bottom of the dialog box. For more information, see the *Vivado Design Suite User Guide: Synthesis* (UG901) [Ref 10].
- **Implementation**: Shows the default constraints set. It also enables you to specify a placed and routed checkpoint to use as a reference for the next implementation run. It provides an options area for selecting an implementation strategy and for setting command line options for the `opt_design`, `power_opt_design`, `place_design`, `phys_opt_design`, and `route_design` tool steps that occur during implementation. The command line options are defined by the selected implementation strategy, but you can override the setting with your own selections. A description of the selected command line option displays at the bottom of the dialog box. For more information, see the *Vivado Design Suite User Guide: Implementation* (UG904) [Ref 12].
- **Bitstream**: Specifies the bitstream options to use. A description of the selected command line option displays at the bottom of the dialog box. For more information, see the *Vivado Design Suite User Guide: Programming and Debugging* (UG908) [Ref 14].

  Note: After a design is loaded, additional bitstream settings are available by selecting **Tools > Edit Device Properties**. For more information, see Editing Device Properties.
- **IP**: Shows all user-specified repositories and allows you to specify additional locations. You can also specify default values for the Vivado IP packager. For more information, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 7].

Running RTL Analysis, Synthesis, Implementation, and Bitstream Generation

Run commands are available in several areas of the Vivado IDE:

- Flow Navigator
- Flow menu
- Main toolbar
- Design Runs window

The Vivado IDE provides "one click" execution for any stage of the design. For example, to view the RTL analysis elaborated design, click **Open Elaborated Design** in the Flow Navigator or the **Flow** menu. The design displays with the default layout.

To run the design through the entire flow and generate a bitstream file, click **Generate Bitstream** in the Flow Navigator or the **Flow** menu. Synthesis and implementation are run (if required), and the bitstream file is created. The state of the design is tracked in the Vivado IDE, so only the required implementation steps are run. For example, modifying implementation-specific constraints does not result in synthesis becoming out-of-date.

For more information, see the following documents:

- *Vivado Design Suite User Guide: Design Flows Overview* (UG892) [Ref 1]
- *Vivado Design Suite User Guide: System-Level Design Entry* (UG895) [Ref 15]
- *Vivado Design Suite User Guide: Synthesis* (UG901) [Ref 10]
- *Vivado Design Suite User Guide: Implementation* (UG904) [Ref 12]

Opening Designs

Use the Flow Navigator, **Flow** menu, or popup menu in the Design Runs window to open available designs:

- **Open Elaborated Design**
- **Open Synthesized Design**
- **Open Implemented Design**

The **Flow > Open Implemented Design** command populates the Vivado IDE as shown in Figure 2-7.

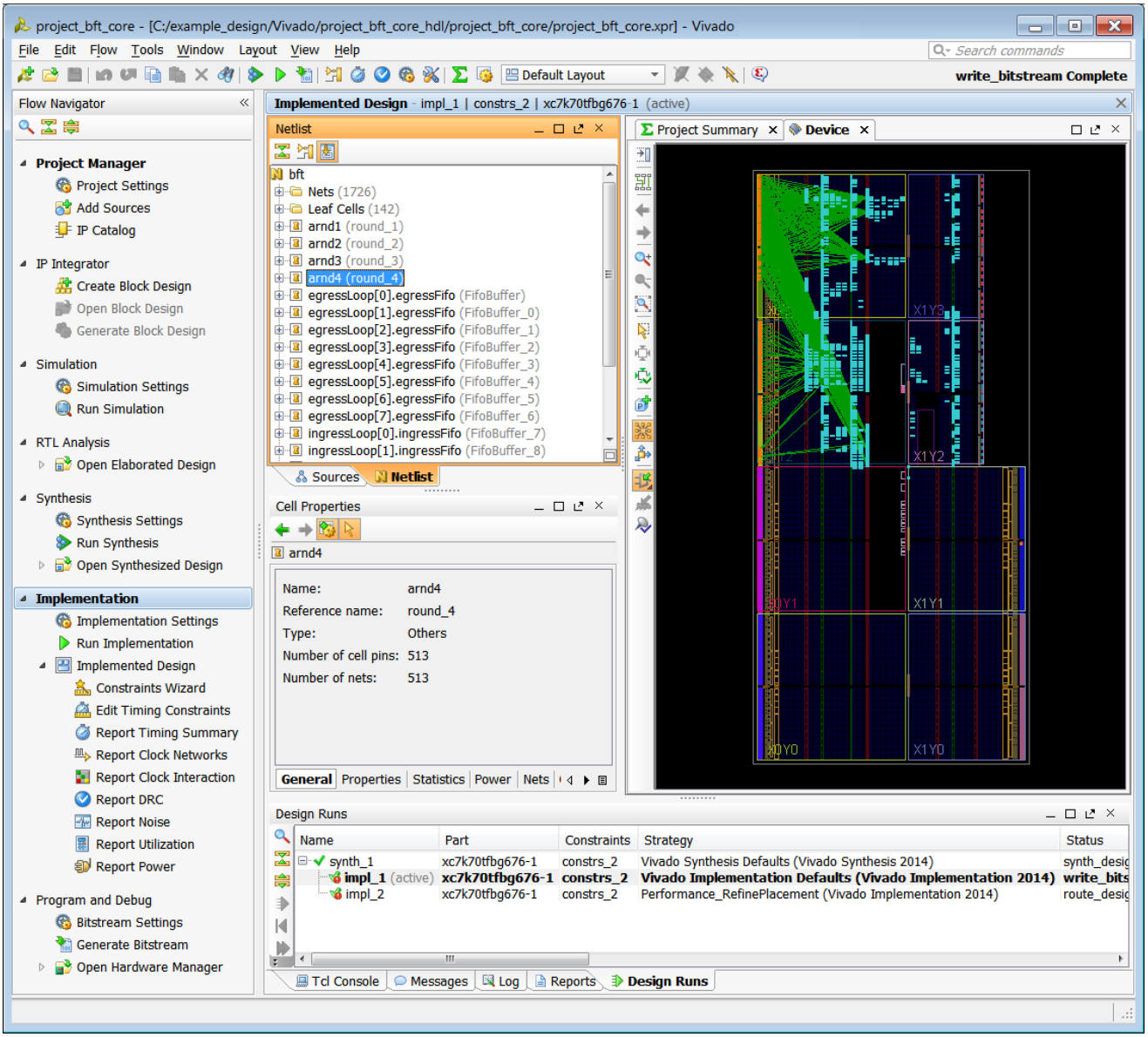


Figure 2-7: **Implemented Design**

Critical warnings and errors are displayed in a popup dialog (Figure 2-8) when opening a project, loading a design, or creating or launching runs. This ensures that you are aware of any issues that might require your attention. These messages also display in the Messages window.

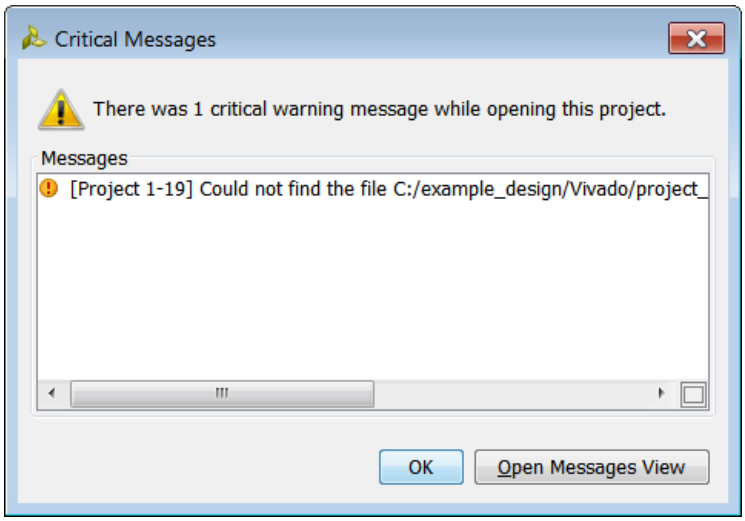


Figure 2-8: **Critical Messages**

For more information, see the following documents:

- *Vivado Design Suite User Guide: Design Flows Overview* (UG892) [Ref 1]
- *Vivado Design Suite User Guide: Design Analysis and Closure Techniques* (UG906) [Ref 13]

Finding Objects

After loading a design, you can use the **Edit > Find** command or **Ctrl**+**F** keyboard shortcut to search for design or device objects. In the Find dialog box (Figure 2-9), you can specify Tcl properties to filter the data using the following options. When you click **OK**, a Tcl command is run to populate the Find Results window.

- **Results name**: Labels the Find Results window that shows the found objects.
- **Properties**: Specifies the Tcl properties used to find the design or device objects. Click the add (+) button to add properties. Click the remove (X) button to remove properties.
- **Regular expression**: Searches for the specified string by matching text patterns based on regular expression syntax.
- **Ignore case**: Searches for the specified regular expression string, regardless of whether the string uses upper or lowercase.

- **Search hierarchically**: Searches through the entire design hierarchy.
- **Of Objects**: Specifies a particular object to search. Click the Of Objects (...) button to open a new dialog box and specify the objects to search.
- **Command**: Shows the Tcl command that is run to populate the Find Results window.
- **Open in a new tab**: Opens a new Find Results tab instead of replacing the previous results.

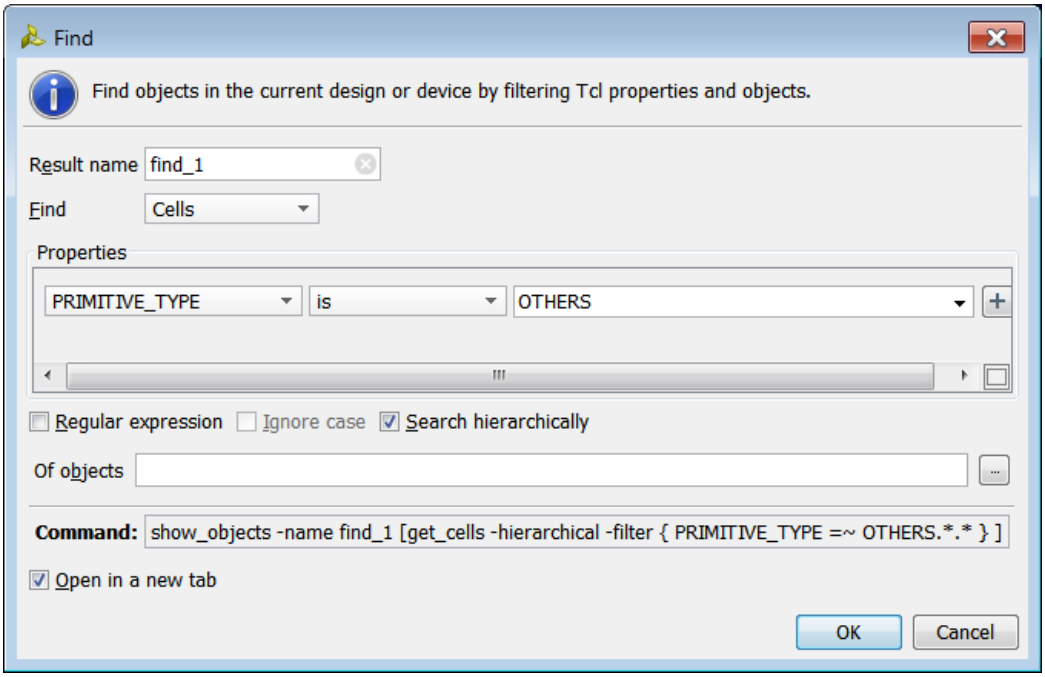


Figure 2-9: **Find Dialog Box**

Finding Unplaced BUFGs

Figure 2-10 shows the Find dialog box settings for finding unplaced BUFGs.

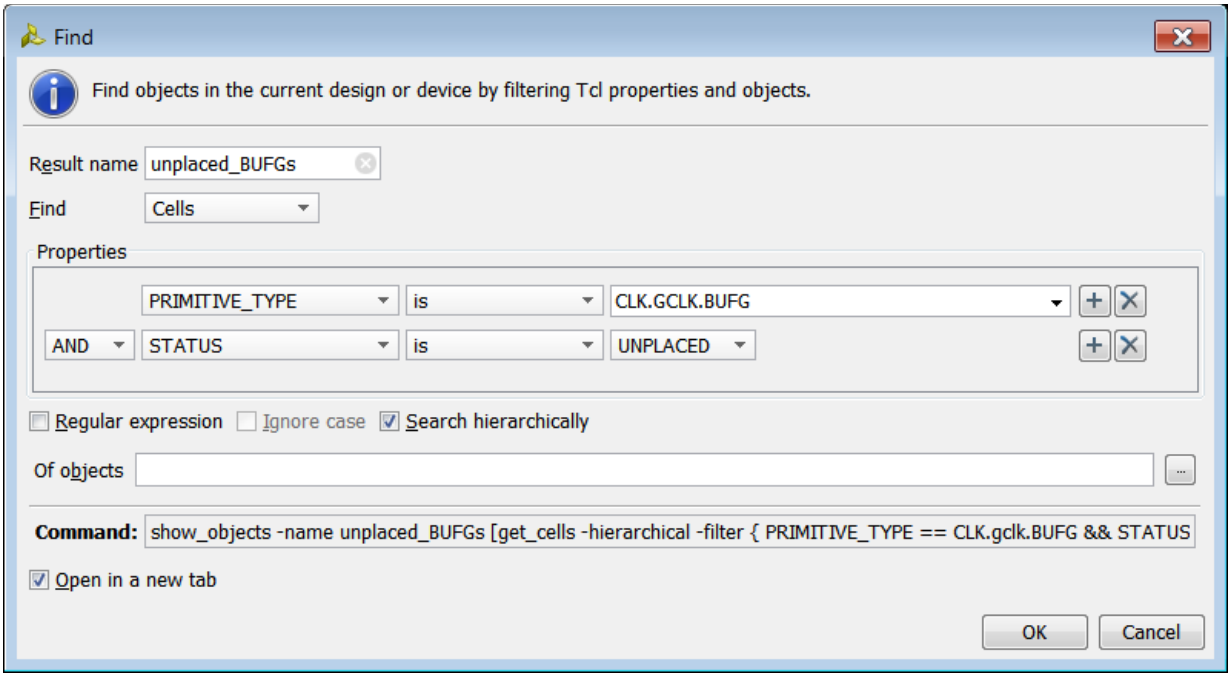


Figure 2-10: **Finding Unplaced BUFGs**

Following is the Tcl command that is run to populate the Find Results window:

```
show_objects -name find_1 [get_cells -hierarchical -filter { PRIMITIVE_TYPE ==
CLK.gclk.BUFG && STATUS == "UNPLACED" } ]
```

Note: By default, the `get_*` Tcl command truncates the returned results in the Tcl Console and log file after the first 500 results. For more information, including how to change the default setting, see the *Vivado Design Suite Tcl Command Reference Guide* (UG835) [Ref 5].

Finding Regional Clocks with No Loads

Figure 2-11 shows the Find dialog box settings for finding regional clocks with no loads.

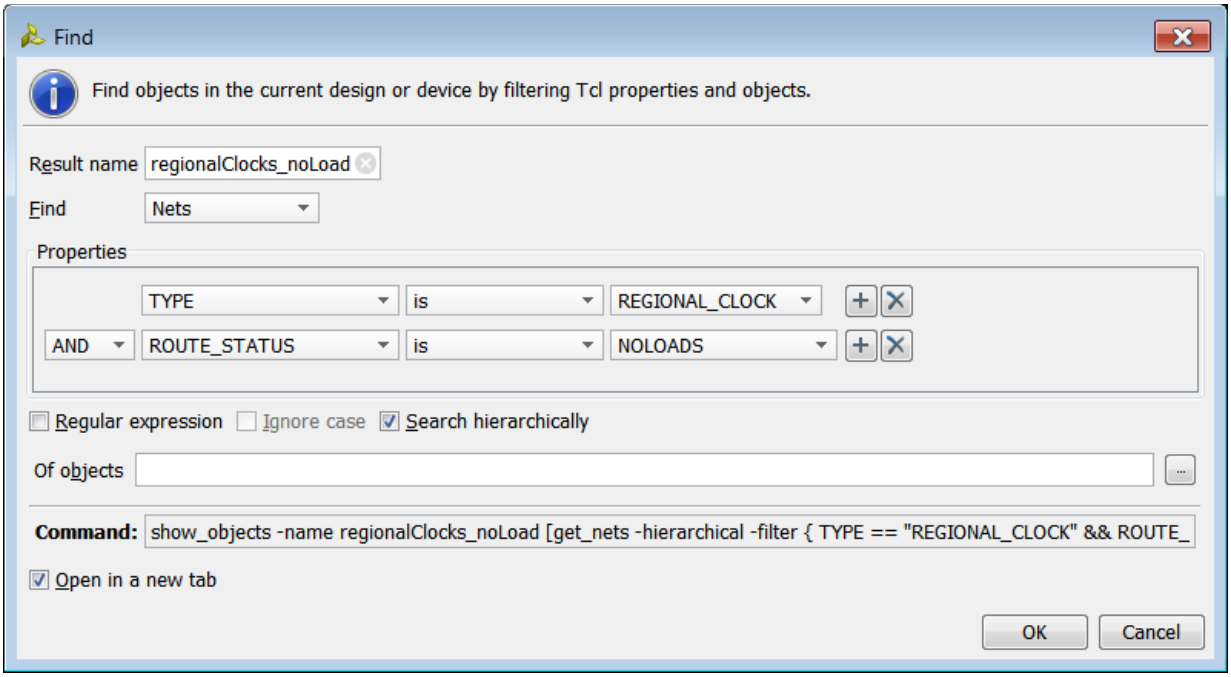


Figure 2-11: **Finding Regional Clocks with No Loads**

Following is the Tcl command that is run to populate the Find Results window:

```
show_objects -name regionalClocks_noLoads [get_nets -hierarchical -filter { TYPE
== "REGIONAL_CLOCK" && ROUTE_STATUS =~  "NOLOADS" } ]
```

Finding Placed RAMB36 Cells

Figure 2-12 shows the Find dialog box settings for finding placed RAMB36 cells.

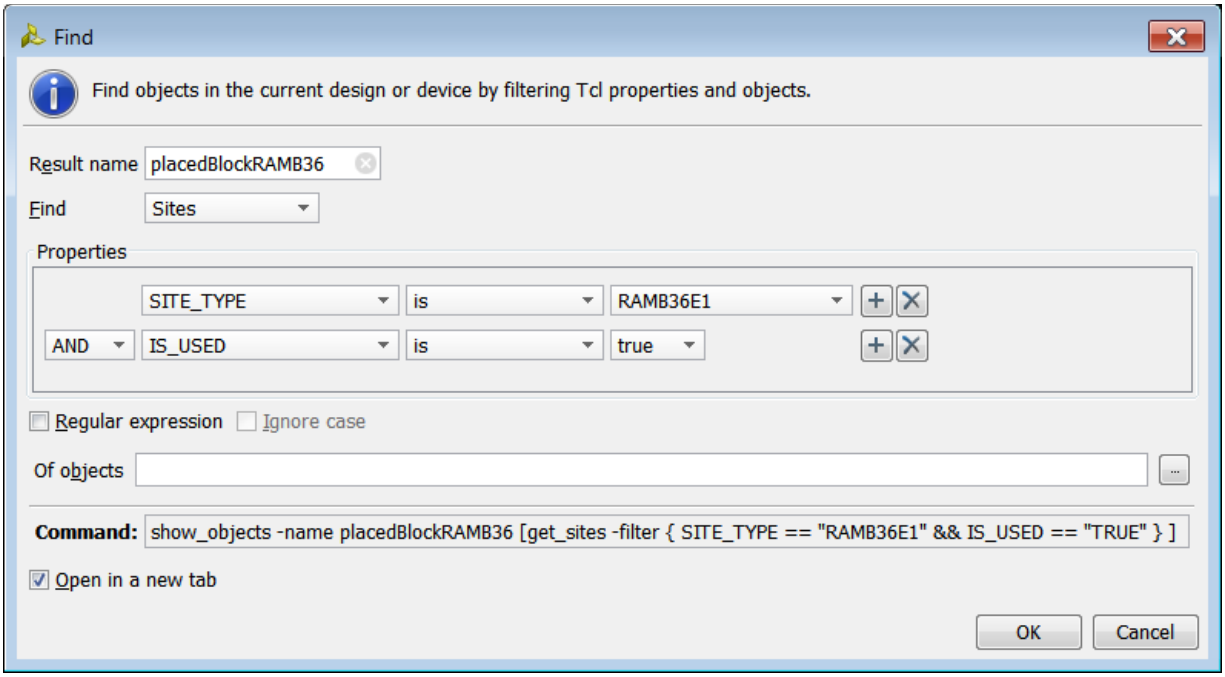


Figure 2-12: **Finding Placed RAMB36 Cells**

Following is the Tcl command that is run to populate the Find Results window:

```
show_objects -name placedBlockRAMB36 [get_sites -filter { SITE_TYPE =~  "RAMB36*"
&& IS_USED == "TRUE" } ]
```

Finding Objects

In the Find dialog box, you can click the Of Objects button (...) to open the Of Objects dialog box, which enables you to specify a particular object to search. Figure 2-13 shows the Of Objects dialog box settings for a search for a specific slice.

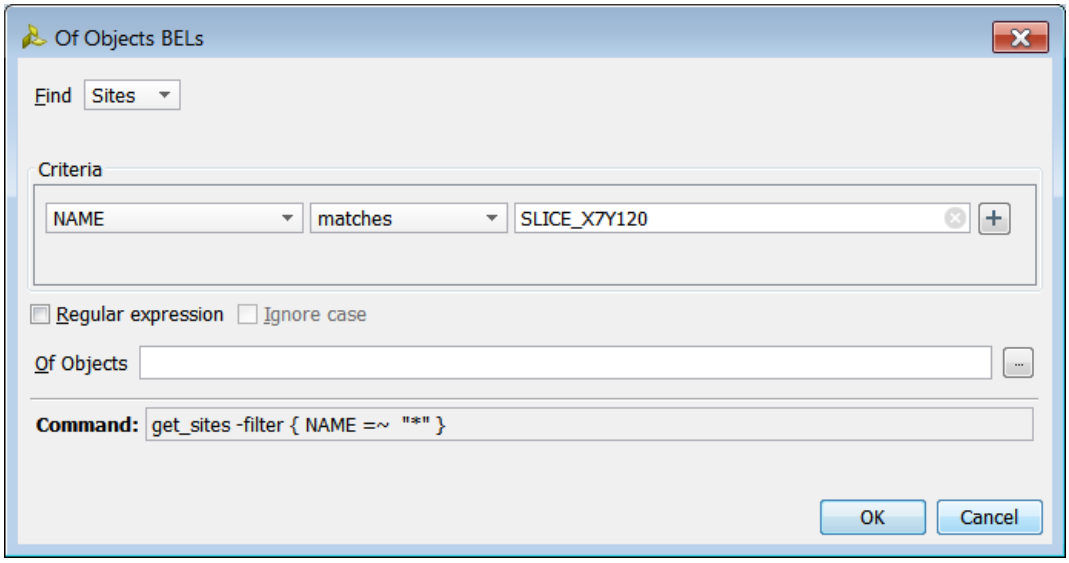


Figure 2-13: **Finding an Object**

After specifying the slice, you can specify the occupied BELs, as shown in Figure 2-14.

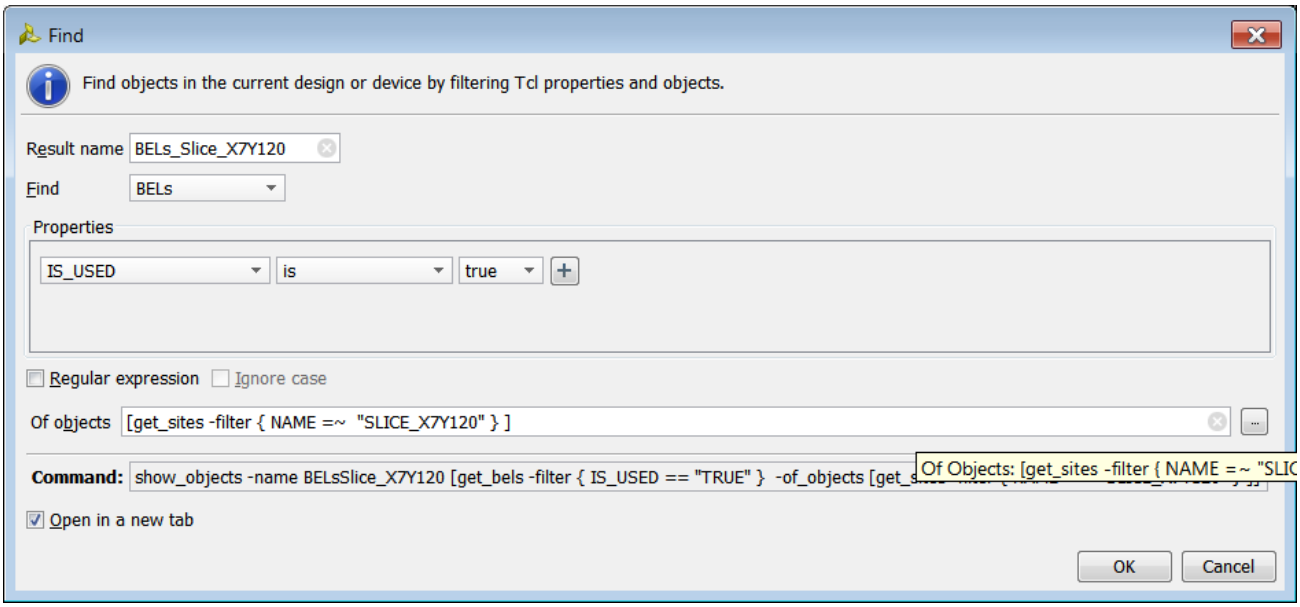


Figure 2-14: **Finding Occupied BELs within a Slice**

Following is the Tcl command that is run to populate the Find Results window:

```
show_objects -name find_1 [get_bels -filter { IS_USED == "TRUE" }  -of_objects
[get_sites -filter { NAME =~  "SLICE_X7Y120" } ]]
```

Finding Black Box Cells

Figure 2-15 shows the Find dialog box settings for finding black box cells.

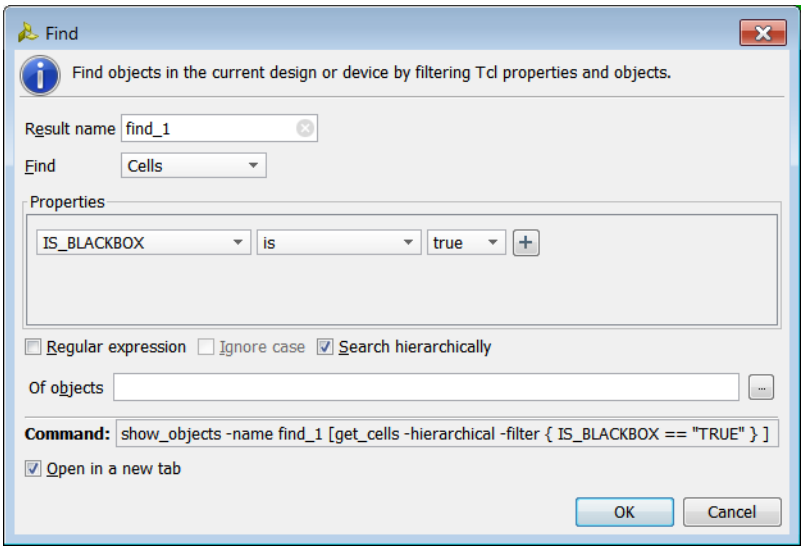


Figure 2-15: **Finding Black Box Cells**

Following is the Tcl command that is run to populate the Find Results window:

```
show_objects -name find_1 [get_cells -hierarchical -filter { IS_BLACKBOX ==
"TRUE" } ]
```

Editing Properties

You can edit object properties, such as files, cells, designs and I/Os, as follows:

- To update programming and configuration properties, use the Edit Device Properties dialog box.
- To update properties for multiple objects, use the Properties Editor.

TIP: *To edit properties for a single object, use the Properties window as described in Using the Properties Window in Chapter 3.*

Editing Device Properties

After loading a design, you can use the **Tools > Edit Device Properties** command to edit programming and configuration properties. In the Edit Device Properties dialog box (Figure 2-16), hover the mouse cursor over the property values to see the associated constraint property name. For example, the Enable Bitstream Compression property is associated with the BITSTREAM.GENERAL.COMPRESS constraint. For information on each property, see Appendix A in the *Vivado Design Suite User Guide: Programming and Debugging* (UG908) [Ref 14]. For information on setting device configuration modes, see the *Vivado Design Suite User Guide: I/O and Clock Planning* (UG899) [Ref 16].

IMPORTANT: *When you edit the properties, the constraints are in memory. Select **File > Save Constraints** to write the properties to the target constraint file.*

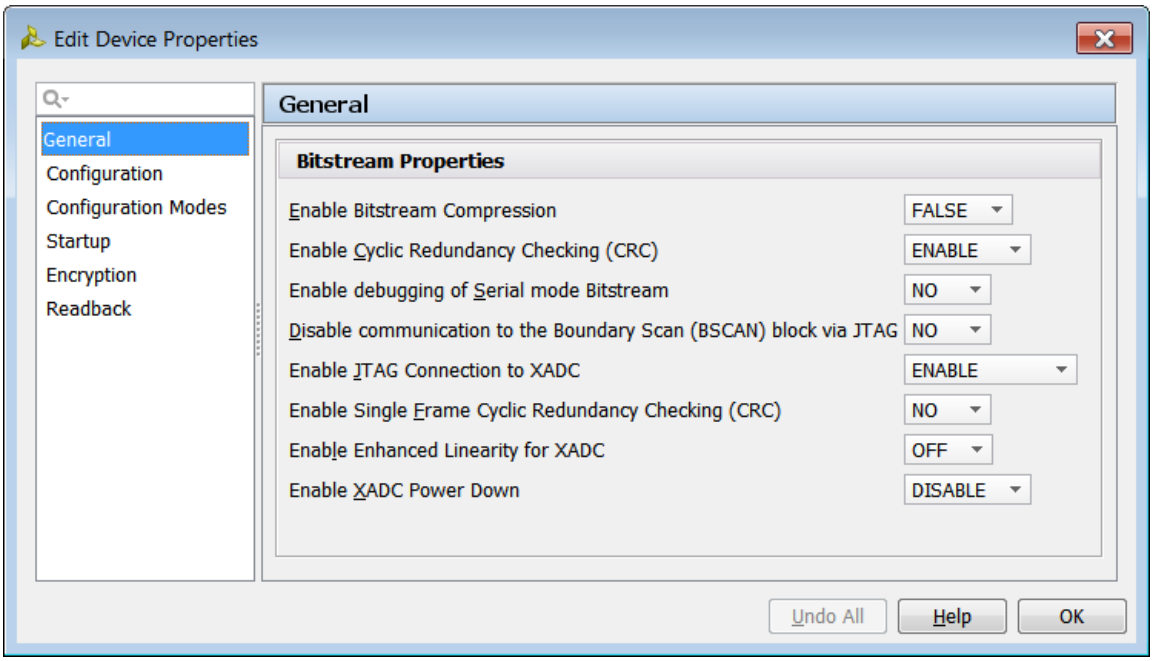


Figure 2-16: **Edit Device Properties Dialog Box**

Following is an example Tcl command that shows enabling of bitstream compression:

```
set_property BITSTREAM.GENERAL.COMPRESS TRUE [get_designs netlist_1]
```

Editing Properties for Multiple Objects

After selecting multiple objects in a workspace window, you can use the **Tools > Property Editor** command to open the Property Editor (Figure 2-17) and edit properties for the selected objects.

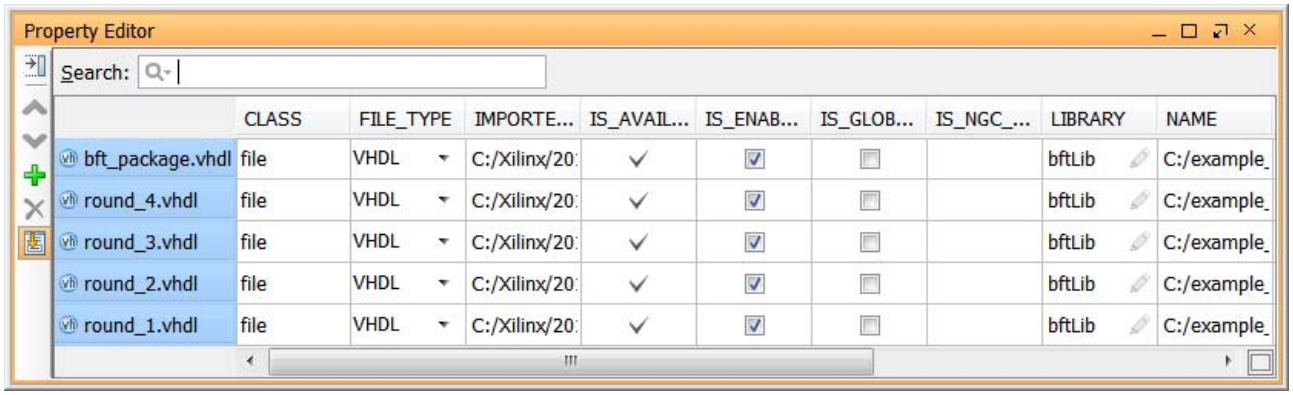


Figure 2-17: **Property Editor**

In the Property Editor, you can also do the following:

- To adjust the Property Editor display, click the **Property Options** toolbar button. Edit the options that control the display of the header, types of objects, and properties (Figure 2-18).
- To change the value of a property for multiple objects, change a value in a cell at the top or bottom of the list. Press **Ctrl** or **Shift** and select the modified cell as well as the cells you want to change. Click the **Fill Up** or **Fill Down** toolbar button.
- To add more objects to the Property Editor, drag the objects from the workspace window, and drop them onto the Property Editor. Alternatively, click the **Add** toolbar button.

TIP: *Editable cells are indicated by a pencil icon* .

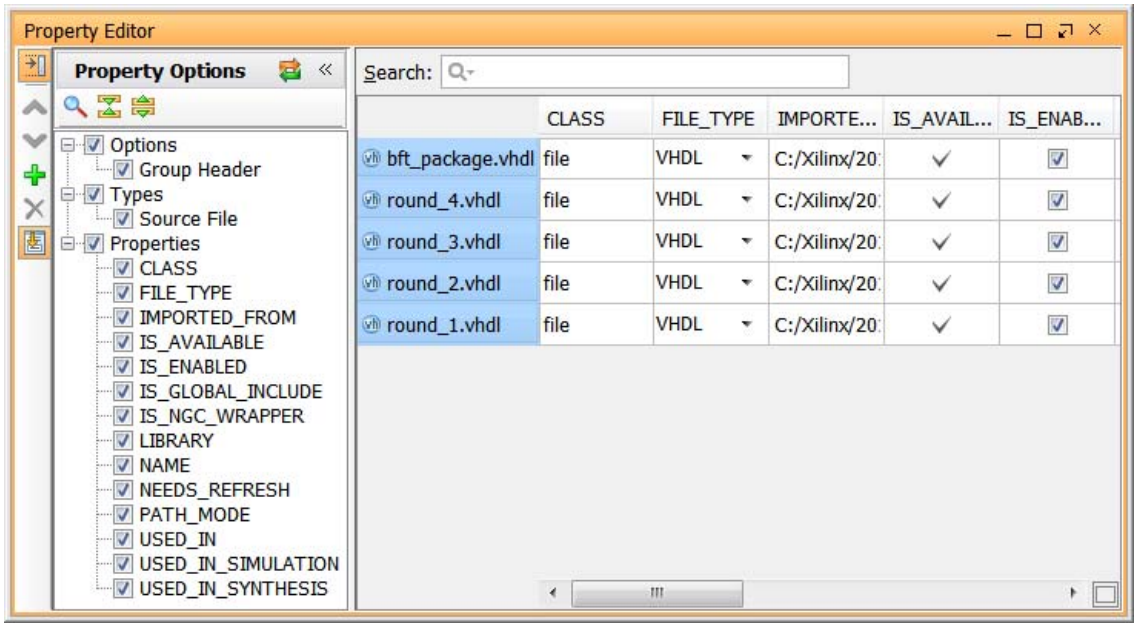


Figure 2-18: **Property Editor Options**

Property Editor Toolbar Commands

The local toolbar contains the following commands:

- **Property Options**: Modifies the display of the header, types of objects, and properties.
- **Fill Up**: Applies the changed value to all selected cells above the changed cell.
- **Fill Down**: Applies the changed value to all selected cells below the changed cell.
- **Add**: Adds selected objects to the list.
- **Remove**: Removes the selected object from the list.
- **Automatically Scroll to Selected Objects**: Scrolls the list of objects in the Property Editor to show the objects selected in other windows, such as the Sources or Netlist windows.

Using the Project Summary

The Vivado IDE includes an interactive Project Summary (Figure 2-19) that updates dynamically as design commands are run and as the design progresses through the design flow. It provides project and design information, such as the project part, board, and state of synthesis and implementation. It also provides links to detailed information, such as links to the Messages, Log, and Reports windows as well as the Project Settings dialog box. As synthesis and implementation complete, DRC violations, timing values, utilization percentages, and power estimates are also populated.

To customize the displayed data, you can use the scroll bar or the **Collapse** and **Expand** buttons to view or hide the different data categories. To open the Project Summary, do either of the following:

- Select **Windows > Project Summary**.
- Select the **Project Summary** toolbar button  .

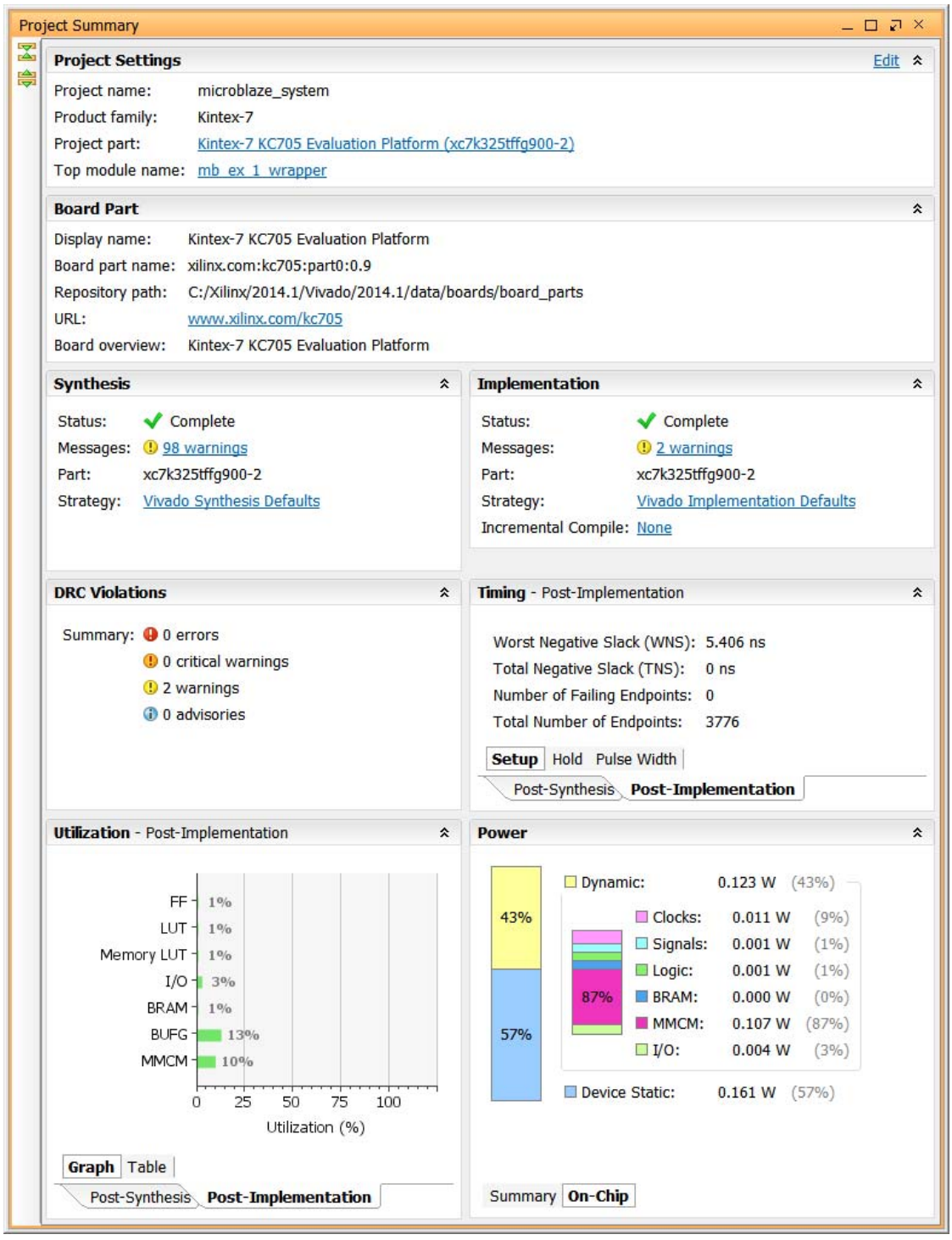


Figure 2-19: **Project Summary**

Using Windows

Overview

This chapter contains general information that applies to all windows in the Vivado® IDE. For example, it covers controlling the size and location of a window. In addition, it covers features that apply only to specific windows, such as:

- Sorting and filtering large data sets
- Importing or updating a file into the project
- Viewing file properties
- Visualizing the design hierarchy
- Creating a port interface
- Determining which objects are selected

Note: For more information on these features, see the *Vivado Design Suite User Guide: Design Flows Overview* (UG892) [Ref 1] and *Vivado Design Suite User Guide: Design Analysis and Closure Techniques* (UG906) [Ref 13].

Working with Windows

Figure 3-1 shows the parts of a window:

1. Title bar
2. Local toolbar
3. Window views
4. Window tab

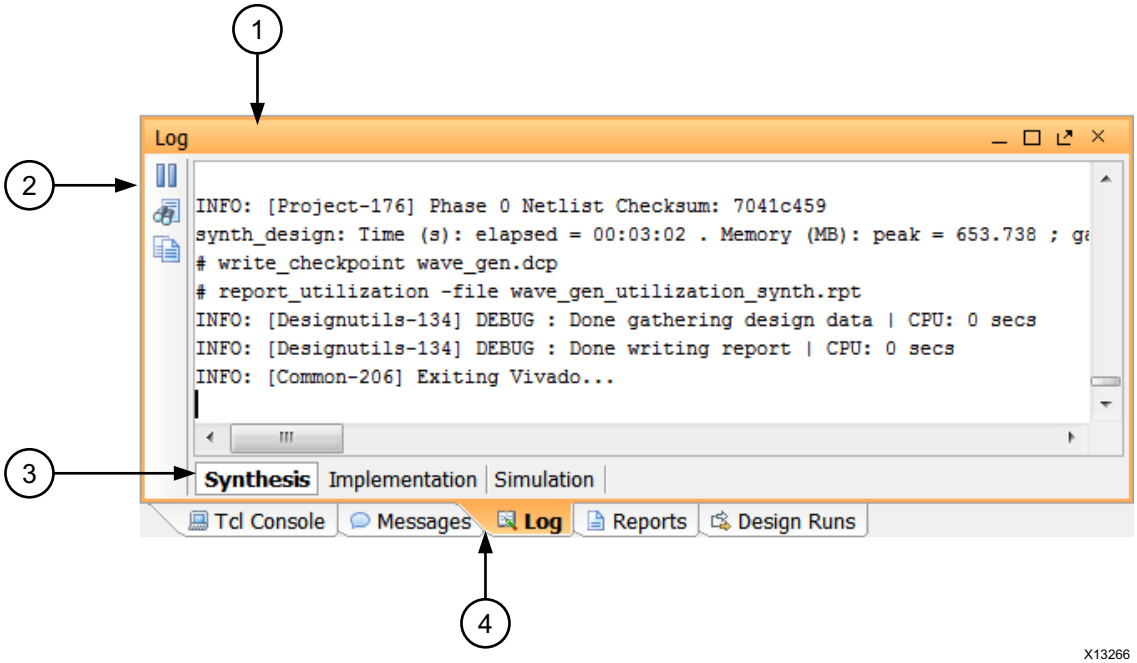


Figure 3-1: **Parts of a Window**

Window Tabs

Each window has a tab that you can select to make that window active. The tab is at the bottom of some windows, such as the Tcl Console and Messages windows, and at the top of others, such as the Project Summary and Device windows.

TIP: *To make the next tab active, press **Ctrl+Tab**. To make the previous tab active, press **Ctrl+Shift+Tab**. To maximize or minimize the window, double-click the window tab, or press **Alt -**.*

Window Views

Some windows include different views of same data. For example, the Log window includes views for Synthesis, Implementation, and Simulation (Figure 3-2).

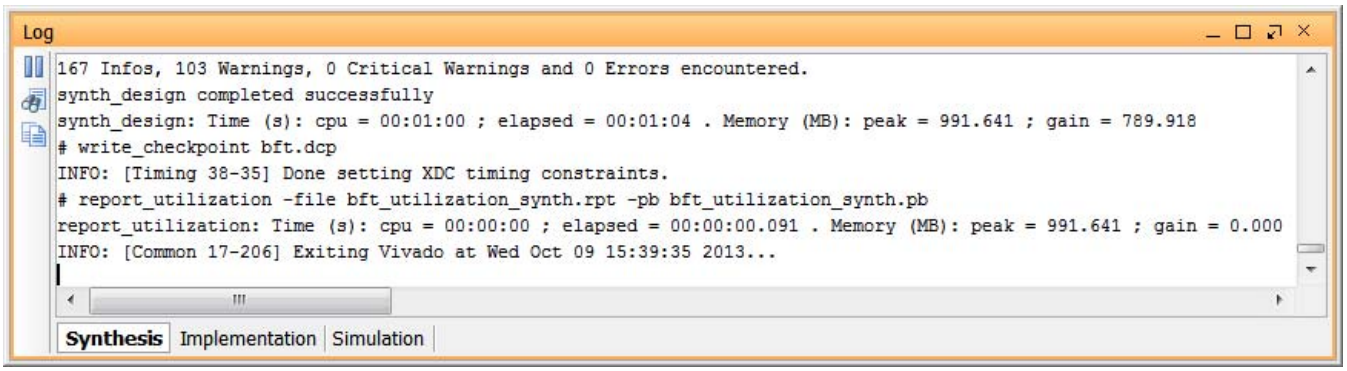


Figure 3-2: **Log Window with Multiple Views**

Window Controls

Each window has the following title bar controls, which enable you to manipulate the window (Figure 3-3):

1. Minimize
2. Maximize
3. Float
4. Close

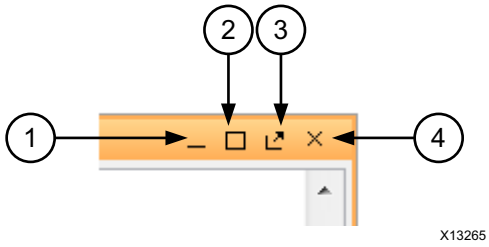


Figure 3-3: **Window Title Bar Controls**

You can move, resize, float, or close windows as described in the following sections.

TIP: *After arranging windows in a configuration that works for you, you can save the layout for future use, as described in Creating Custom View Layouts in Chapter 4.*

Moving Windows

1. Select the window tab or title bar, and drag the window.

   A gray outline indicates where the window will be located after the move.

2. To commit to the placement, release the mouse button.

   Note: Dropping one window onto an existing window places the two window tabs in the same region.

IMPORTANT: *You cannot move windows into or out of the workspace. However, you can resize and move the windows within the workspace as described in Using the Workspace.*

Resizing Windows

To resize windows:

- Click and drag a window border.

  Note: The mouse cursor changes to a resize cursor when positioned over a window border or drag handle, indicating that you can click and drag the window border to resize the window.

- To expand a window to use the all of the viewing environment, click the **Maximize** button in the upper right corner of the window.

  Note: The Vivado IDE minimizes all other open windows, except the Flow Navigator, and expands the selected window to fill all available screen area.

- To restore a window to its original size, click the **Restore** button in the upper right corner of the window.

TIP: *To maximize or restore the window, double-click the window title bar or tab, or press **Alt -**.*

Floating Windows

You can undock a window, including windows in the workspace, from the display docking area. The window appears in a separate floating window, which allows it to be moved and sized independently.

To float a window:

- In the upper right corner of the window, click the **Float** button.
- Select **Float** from the popup menu.

Note: If windows overlap, you can move a floating window by dragging the window title bar. You can also move a floating window to another monitor display.

Closing Windows

To close windows:

- In the upper right corner of the window, click the **Close** button.

  Note: In some cases, this button is also available in the window tab.

- Right-click a window tab or title bar, and select **Close** from the popup menu.

Using Data Table Windows

The Vivado IDE contains windows that display as expandable data tables (Figure 3-4). These windows share common characteristics and features as described in the following sections.

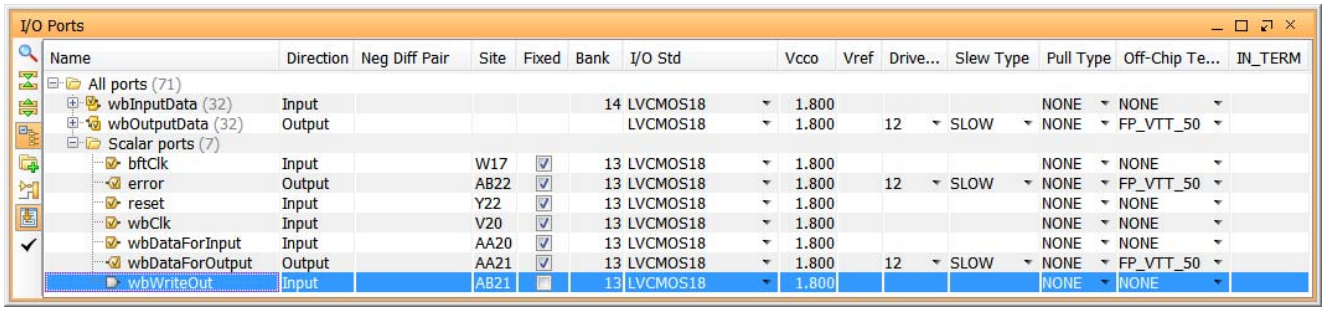


Figure 3-4: **Data Table Window**

Expanding and Collapsing the Table

To expand or collapse the table:

- In the Name column, click the expand (+) and collapse (-) buttons to expand or collapse portions of the tree.
- In the local toolbar, use the **Expand All** and **Collapse All** buttons to expand or collapse the entire tree.

Displaying Entries in a Flat List or a Group

In the local toolbar, click the **Group by Type** button to show the entries either grouped by a particular type or as a single flat list of entries. For example, in the I/O Ports window, you can toggle between a display grouped by interface and bus or shown as a flat list (Figure 3-5).

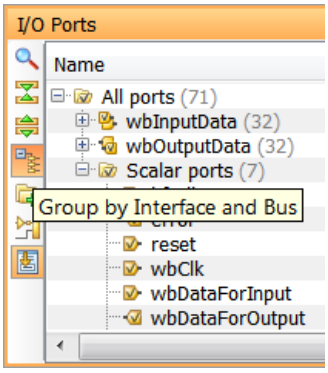


Figure 3-5: **Group by Type or Flat List Toolbar Button**

Using the Show Search Capability to Filter the List

RECOMMENDED: *For best results, flatten the list before searching and filtering, as described in the preceding section.*

1. In the local toolbar, click the **Show Search** button to display a search field in the banner of the window.

   Note: You can also access this command through the **Alt+/** keyboard shortcut.

2. Optionally, select the pull-down menu on the left of the search field, and select search criteria, including which columns to search (Figure 3-6).

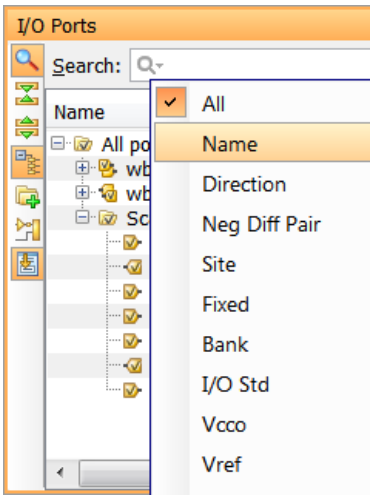


Figure 3-6: **Search Pull-Down Menu**

3. Enter a text string to filter the list displayed in the table window.

   When you enter a text string, the list adjusts dynamically to list only those entries that contain the string. Click the **Show Search** button again to hide the Search field and filtering.

Sorting Columns

You can sort table columns in increasing or decreasing order according to the sort criteria of the selected column. A visual indication of the sort order displays in the column header, as shown in Figure 3-7.

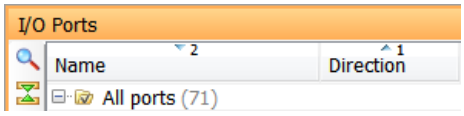


Figure 3-7: **Sort Order Arrows**

To sort columns:

- Click a column header to sort data in the table in an increasing order.
- Click the column header again to sort the data in the table in a decreasing order.
- To add sort criteria for additional columns, press **Ctrl** and click the header of the column.

  Note: For example, in Figure 3-7, the Direction column is the primary sort criteria, and the Name column is the secondary sort criteria.
- To remove sort criteria from a column, press **Ctrl** and click the column header.

Organizing Columns

To organize columns:

- To move a column, select the column and drag it to a new location.
- To hide a column, right-click the column header, and select **Hide This Column** from the popup menu.
- To adjust the width of the columns based on the displayed data, right-click the column header, and select **Auto Resize Column** from the popup menu.
- To restore the table default settings, right-click a column header, and select **Reset to Default** from the popup menu.

Using Window-Specific Toolbar Commands

Most windows have local toolbar buttons to run commonly used commands that are specific to the window (Figure 3-8). Certain buttons are enabled only when you select specific objects, such as files, sites, ports, instances, and cells. Toolbar commands are covered in detail in the specific window sections that follow.

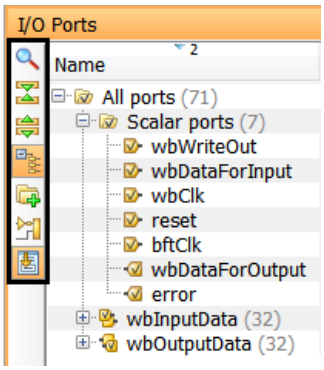


Figure 3-8: **Local Toolbar**

Using the Sources Window

The Sources window (Figure 3-9) allows you to manage project source files, including adding, removing, and reordering the sources to meet specific design requirements. The Sources window displays the following sources when they are part of the project:

- Design sources
- Constraint files
- Simulation sources
- IP cores

Generally, the Sources window is available in the Vivado IDE whenever a project is open. To open the Sources window, select **Windows > Sources**. The Sources window includes the following folders:

- **Design Sources**: Displays source file types, including Verilog, VHDL, NGC/NGO, EDIF, IP cores, digital signal processing (DSP) modules, and XDC and SDC constraint files.
  - **Non-Module Files**: Displays files that produced issues during parsing.
  - **Disabled Sources**: Displays disabled files.
  - **Text**: Displays text files that are part of the project.
- **Constraints**: Displays constraint files, which are assigned to constraint sets. For more information on design constraints, see the *Vivado Design Suite User Guide: System-Level Design Entry* (UG895) [Ref 15] and *Vivado Design Suite User Guide: Using Constraints* (UG903) [Ref 11].
- **Simulation Sources**: Displays the source files that are used for simulation. For more information on defining and using simulation files, see the *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 9].

IMPORTANT: *Messages, such as Critical Warnings, encountered during the building of the hierarchy display at the top of the hierarchy tree in the Sources window.*

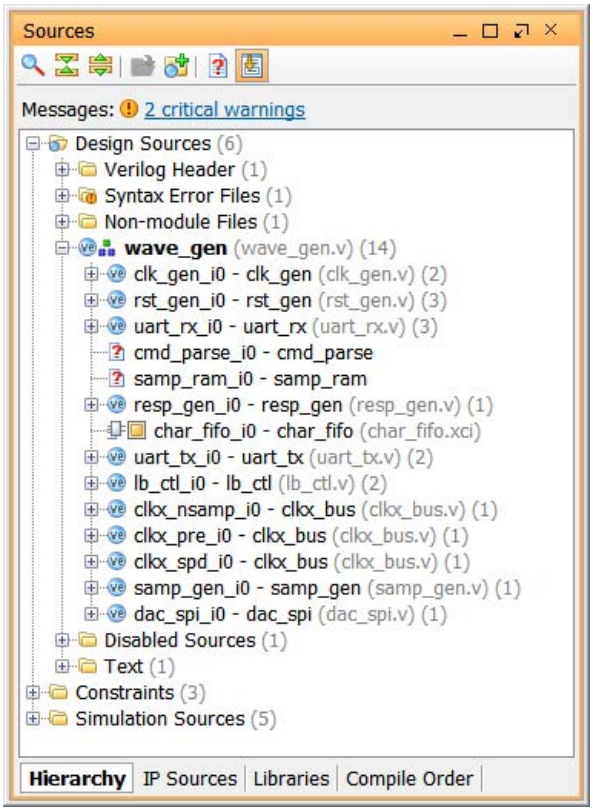


Figure 3-9: **Sources Window**

Sources Window Views

The Sources window has the following views to display the source files in different ways:

- Hierarchy View
- IP Sources View
- Libraries View
- Compile Order View

Hierarchy View

The Hierarchy view displays the hierarchy of the design modules and instances, along with the source files that contain them. The top module defines the hierarchy of the design for compilation, synthesis, and implementation. The Vivado IDE automatically detects the top module, but you can also manually define the top module using the **Set as Top** command. For information, see Sources Window Popup Menu.

Hierarchy View Icons

The Hierarchy view uses the following icons:

- Top module 
- Missing File/Module/Instance 
- Out-of-Context Module 
- Global Include File 
- Verilog Header File 
- Verilog File 
- SystemVerilog File 
- VHDL File 
- Constraint File 
- Tcl File 
- IP 
- Locked IP 
- System Generator/DSP 
- Block Design 
- Design Checkpoint 
- Netlist 
- Hidden Instantiation 
- Report 

TIP: *When a file, module definition, or instantiation of a module is missing in the design hierarchy, the* ***Filter Sources by Missing Files or Instantiations*** *button*  *is enabled in the Sources window local toolbar.*

IP Sources View

The IP Sources view displays all of the files defined by an IP core. For more information, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 7].

Libraries View

The Libraries view displays the sources sorted into the various libraries.

Compile Order View

The Compile Order view is available for synthesis, implementation, and simulation. At the top of the view, you can select the appropriate design flow step from the drop-down list, as shown in Figure 3-10. The Compile Order view displays source files in the order in that they will be compiled, first to last, and shows the processing order for constraints.

When working with sources, the top module is usually the last file to be compiled. You can allow the Vivado IDE to automatically determine the compile order based on the defined top module and the elaborated design. Alternatively, you can manually control the compile order of the design by using the **Hierarchy Update** popup menu command and reordering the source files. For more information, see Sources Window Popup Menu.

When working with constraints, the processing order is controlled by the `PROCESSING_ORDER` property of the constraints file, which you can set to `EARLY`, `NORMAL`, or `LATE`. For example:

```
set_property PROCESSING_ORDER {EARLY} [get_files myConstraintFile.xdc]
```

Alternatively, you can examine the compile order using Tcl commands. For example:

```
report_compile_order
report_compile_order -fileset sources_1
report_compile_order_-constraints
report_compile_order -constraints -fileset constrs_1
```

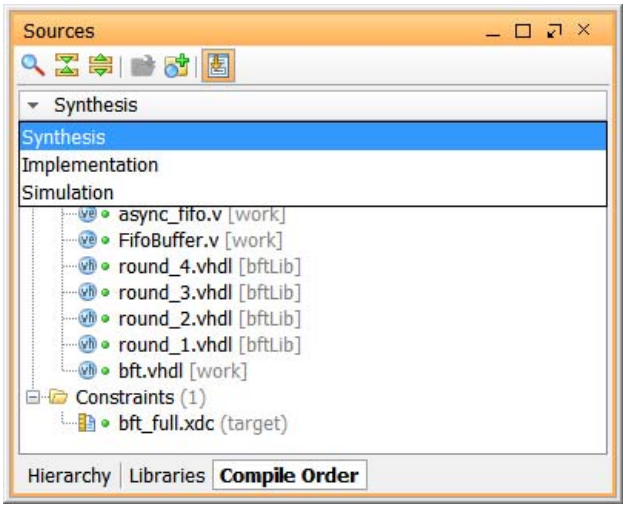


Figure 3-10: **Compile Order View Drop-Down List**

Sources Window Commands

To add, view, or modify source files, use the Sources window local toolbar or popup menu commands. To display the popup menu, click the right mouse button.

Sources Window Toolbar Commands

The local toolbar contains the following commands:

- **Show Search**: Opens the search bar to allow you to quickly locate objects in the Sources window.

  Note: You can also access this command through the **Alt+/** keyboard shortcut.
- **Collapse All**: Collapses all hierarchical tree objects to display only the top-level objects.
- **Expand All**: Expands all hierarchical tree objects to display all elements of the Sources window.
- **Open Selected Source Files**: Opens selected files as follows:
  - RTL source files or constraint files open in the Text Editor.
  - Selected IP cores open in the Customize IP dialog box.
  - DSP modules open in System Generator. For more information, see the *System Generator for DSP User Guide* (UG640) [Ref 17].
- **Add Sources:** Adds or creates RTL source files, simulation source files, constraint files, or DSP modules as well as adds existing IP.

- **Filter Sources by Missing Files or Instantiations**: Filters sources to display missing files or missing instances. This command is enabled when a file, module definition, or instantiation is missing in the design hierarchy. When you select the command, the Sources window is filtered to display the missing files or modules. 

TIP: *When the toolbar button icon is gray, or disabled, there are no problems with the design hierarchy.*

- **Automatically Scroll to Selected Object**: Updates the Sources window to focus on the currently selected object. This can be useful on large designs with many source files. This feature is on by default. 

Sources Window Popup Menu

The Sources window popup menu commands are:

- **Source File Properties**: Opens the Source File Properties window. For more information, see Viewing Source File Properties.

  Note: In the Hierarchy window, this command is called **Source Node Properties**.

- **Open File**: Opens selected files as follows:
  - RTL source files or constraint files open in the Text Editor.
  - DSP modules open in System Generator. For more information, see the *System Generator for DSP User Guide* (UG640) [Ref 17].
- **Replace File**: Replaces the specified source file with another file.
- **Copy File Into Project**: Copies selected source files and directories into the project directory. This command is enabled only when the selected source file is not currently local to the project.
- **Copy All Files Into Project**: Copies all remotely referenced source files into the local project directory. This command is available only when the source files are not local to the project.
- **Remove File From Project**: Deletes the selected source files from the project. Optionally, removes the files from the local project disk location.
- **Enable File**: Sets the source file status to active for the project. You can toggle source files between enabled and disabled to define different design configurations.

  Note: You can also set the **Enabled** property in the Source File Properties window. For information, see Viewing Source File Properties.

- **Disable File:** Sets the source file status to inactive for the project. You can toggle source files between enabled and disabled to define different design configurations. Disabled source files display as shaded gray in the Sources window.

- **Move to Simulation Sources**: Relocates currently selected source files into the simulation set. If there is more than one simulation set, the application prompts you to select the simulation set to use.
- **Move to Top**: Relocates the currently selected source file to the top of the source file list in the Compile Order view. The compilation and synthesis of source files is handled in the order listed in Compile Order view, from top to bottom. The order of files affects the elaboration, synthesis, and simulation results. The file order displayed in the Compile Order view is automatically updated or can be manually defined depending on the setting of the **Hierarchy Update** command.

IMPORTANT: *Except for the **Move to Simulation Sources** command, the **Move** commands are not available from the Hierarchy view of the Sources window.*

- **Move Up**: Moves the currently selected source file up in the source file list.
- **Move Down**: Moves the currently selected source file down in the source file list.
- **Move to Bottom**: Moves the currently selected source file to the bottom of the source file list.
- **Hierarchy Update**: Determines how the Vivado IDE responds to changes of the source files such as redefined top module, added or removed files, or changed file order. Select one of the following:
  - **Automatic Update and Compile Order**: Specifies that the Hierarchy view containing the design and the compilation order is automatically updated as source files are changed. The Vivado IDE automatically identifies and sets the best top module candidate. The compile order is also automatically managed, as the top module file and all sources that are under the active hierarchy are passed to synthesis and simulation in the correct order. The files that are outside of the hierarchy defined by the top module are not used.

    Note: This setting is selected by default.
  - **Automatic Update, Manual Compile Order**: Specifies that the Hierarchy view containing the design is automatically updated as source files are changed, but that the compilation order is determined manually. All files in the project are passed to synthesis and simulation. The compilation order is manually defined by ordering the files using the **Move to Top**, **Move Up**, **Move Down**, and **Move to Bottom** commands from the Compile Order view.
  - **No Update, Manual Compile Order**: Specifies that the Hierarchy view is not automatically updated, and that the compilation order is determined manually. To update the design hierarchy in this mode, use the **Refresh Hierarchy** command.

- **Refresh Hierarchy**: Updates the design hierarchy to reflect the latest source file changes and top module definition. Use this command to manually refresh the hierarchy as needed.
- **IP Hierarchy**: Controls the expansion of the IP displayed in the Hierarchy view. By default, all IP hierarchy is collapsed.
  - **Show All IP Hierarchy**: Expands the hierarchy for all IP in the Hierarchy view.

    Note: Depending on the number of IP in your design, this command might slow down the refresh for the automatic update of the Hierarchy view.
  - **Hide All IP Hierarchy:** Collapses the hierarchy for all IP in the Hierarchy view.
  - **Show IP Hierarchy**: Shows the hierarchy for the selected IP.
  - **Hide IP Hierarchy**: Hides the hierarchy for the selected IP.
- **Set as Top**: Specifies the Top Module to define the starting point for elaboration of the design hierarchy for synthesis and simulation purposes.

IMPORTANT: *The top module is automatically reset to the best candidate if the specified top module cannot be found in the design source files, and the hierarchy update mode is set to automatic. In the Sources window, the top module is indicated by the top module icon*  .

- **Set Global Include**: Defines the specified file as a global include file. This command is available for Verilog source files only.

  Note: You can also set the **Global Include** property in the Source File Properties window. For information, see Viewing Source File Properties.
- **Clear Global Include**: Clears the **Global Include** property from the selected Verilog source file.
- **Make Active**: Makes the selected Constraint Set the active constraint set for synthesis or implementation.
- **Set as Target Constraint File**: Specifies the constraint file that the Vivado IDE targets to write newly-created constraints. For more information on design constraints, see the *Vivado Design Suite User Guide: System-Level Design Entry* (UG895) [Ref 15] and *Vivado Design Suite User Guide: Using Constraints* (UG903) [Ref 11].
- **Set as Out-of-Context Module**: Creates a new file set and synthesis run, which enables you to synthesize the selected level of hierarchy out of context from the rest of the design. For more information, see the *Vivado Design Suite User Guide: Synthesis* (UG901) [Ref 10].

- **Set Library**: Sets a library for the selected RTL source files. You can choose from a list of libraries that are currently defined in the project, or type a new library in the text entry field. Entering a new library adds it to the list of currently defined libraries.

  Note: You can also set the **Library** property in the Source File Properties window. For information, see Viewing Source File Properties.

- **Set File Type**: Sets the type of the currently selected file or files. The Vivado IDE automatically recognizes the type of a file as it is added to the project based on appropriate file extensions. However, you can use the Set Type command to redefine the file type in cases of non-standard file extensions.

  Note: You can also set the **Type** property in the Source File Properties window. For information, see Viewing Source File Properties.

- **Set Used In**: Specifies the tools the file is used for. You can specify a source file to be used or not used during synthesis, simulation, or implementation. Disabling a source file for a particular tool prevents that file from being used by that tool.

  For example, if you set a source file as not used in synthesis, and then open the elaborated design, a black box displays for that source file. Disabling an EDIF or NGC source file from implementation prevents it from being used during implementation.

  Note: You can also set the **Used In** property in the Source File Properties window. For information, see Viewing Source File Properties.

- **Add Sources**: Adds or creates RTL source files, simulation source files, constraint files, or DSP modules as well as adds existing IP.
- **Edit Constraint Sets**: Creates and modifies constraint sets.
- **Edit Simulation Sets**: Creates and modifies simulation sets.

Sources Window Popup Menu for IP Sources

The following commands are available in the Sources window popup menu when an IP core is selected in the Sources window. For more information, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 7].

- **Re-Customize IP**: Opens the IP core to allow modification of properties.
- **Generate Output Products**: Generates target data for the IP core as needed.
- **Reset Output Products**: Removes the current target data to allow the IP core to be regenerated as needed.

- **Out-of-Context Settings**: Enables or disables IP out-of-context settings and specifies the number of processors to use during synthesis.
- **Upgrade IP**: Upgrades the IP core from an older version to the latest available version.
- **Copy IP**: Makes a copy of the selected IP and specifies a new name and location.
- **Open IP Example Design**: Opens an example project for the IP core. This feature is not available for all IP.
- **IP Documentation**
  - **View Product Guide**: Opens the IP product guide for the selected IP core.
  - **View Change Log**: Opens the change log for the selected IP core.
  - **View Product Web Page**: Opens the IP web page for the selected IP core if one is available.
  - **View Answer Record**: Searches the Xilinx® Support database for Answer Records associated with the IP.

Sources Window Popup Menu for DSP Sources

The following commands are available in the Sources window popup menu when a DSP module is selected in the Sources window.

Note: For more information, see the *System Generator for DSP User Guide* (UG640) [Ref 17].

- **Create Top HDL**: Creates a top-level Verilog or VHDL module that contains the selected DSP.
- **View Instantiation Template**: Opens the instantiation template for the DSP to instantiate it into another RTL file.
- **Create Test Bench**: Creates the simulation test bench for the selected DSP module.
- **Generate Output Products**: Generates target data for the DSP module as needed.
- **Reset Output Products**: Removes the currently generated target data.
- **Export Hardware for SDK**: Exports hardware description for use with the Software Development Kit (SDK).

Viewing Source File Properties

Selecting an RTL source file in the Sources window displays information in the Source File Properties window (Figure 3-11).

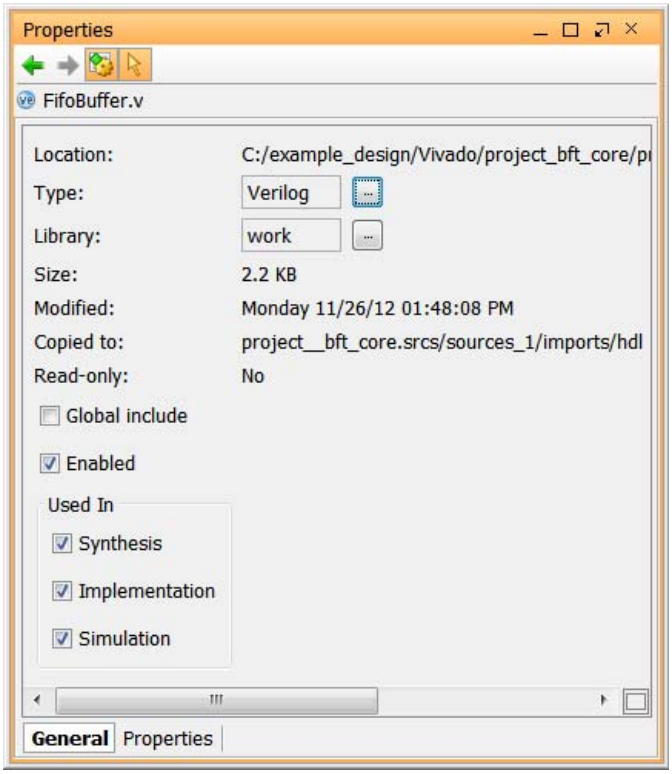


Figure 3-11: **Source File Properties**

To view and modify source file properties:

1. Select a source file in the Sources window.

   The Source File Properties window, located below the Sources window by default, populates with information such as file location, type, library, size, modified timestamp date, location copied from, copy date, and parent module.

   Note: If the Source File Properties window is hidden, right-click a source file in the Sources window, and select **Source File Properties** from the popup menu.

2. In the Source File Properties window, you can change the following settings:
   - **Type**: Changes the file type. This is useful in cases where files have non-standard extensions, and the file type is not properly detected.
   - **Library**: Specifies a new target library for a source file. Select from the list of defined libraries, or type a library name.

- **Global Include**: Enables Verilog source files as global include files. This option forces the selected file to list at the start of the compile order for elaboration and synthesis.
- **Enabled**: Enables the source file in the design. Disabled files display in the source files in gray text and are not considered part of the design for elaboration or compilation.
- **Used In**: Specifies that the source file is used during **Synthesis**, **Simulation**, or **Implementation**. Disabling a source file for a particular tool prevents that file from being used by that tool. For example, if you set a source file to not be used in synthesis, and then open the elaborated design, a black box displays for that source file. Disabling an EDIF or NGC source file from implementation prevents it from being used during implementation.

Using the Language Templates Window

The Language Templates window (Figure 3-12) provides access to a variety of constructs for use in synthesis, constraints, and debugging. You can browse the available files and select a file to preview it. When you select a file, you can use the **Insert Template** popup menu command in the Vivado IDE Text Editor, as described in Using the Text Editor. To display the language templates, select **Window > Language Templates**.

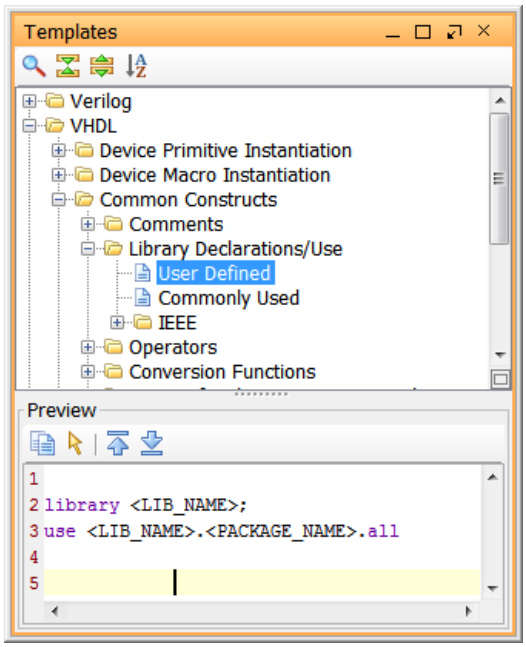


Figure 3-12: **Language Templates**

Language Templates Toolbar Commands

The Templates pane local toolbar contains the following commands:

- **Show Search**: Opens the search bar to allow you to quickly locate objects in the Sources window.

  Note: You can also access this command through the **Alt+/** keyboard shortcut.
- **Collapse All**: Collapses all hierarchical tree objects to display only the top-level objects.
- **Expand All**: Expands all hierarchical tree objects to display all elements of the Sources window.
- **Sort Alphabetically**: Sorts the file tree alphabetically.

The Preview pane local toolbar contains the following commands:

- **Copy**: Copies the selected text.
- **Select All**: Selects all text.
- **Document Home**: Moves cursor to the top of the pane.
- **Document End**: Moves cursor to the bottom of the pane.

Using the Netlist Window

The Netlist window (Figure 3-13) provides a hierarchical view of the elaborated or synthesized logic design including the nets, logic primitives, and hierarchical modules of the design, starting with the currently defined top module.

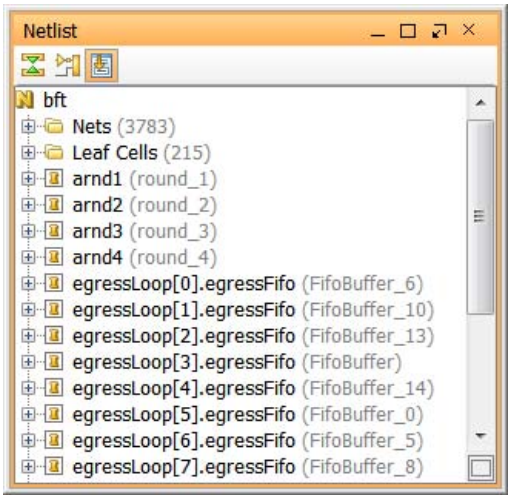


Figure 3-13: **Netlist Window**

The Netlist window includes the following folders:

- **Leaf Cells**: Displays primitive logic for each level of the hierarchy. This folder condenses the display of logic content and hierarchical modules in the Netlist window (Figure 3-14).

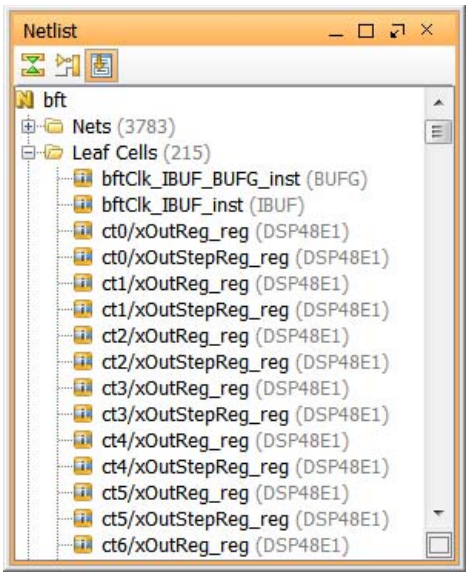


Figure 3-14: **Netlist Window Leaf Cells Folder**

- **Nets**: Displays nets, or wires, for each level of the hierarchy. All of the bits of a bus are collapsed under the bus by default, but you can expand buses to show each individual bit (Figure 3-15).

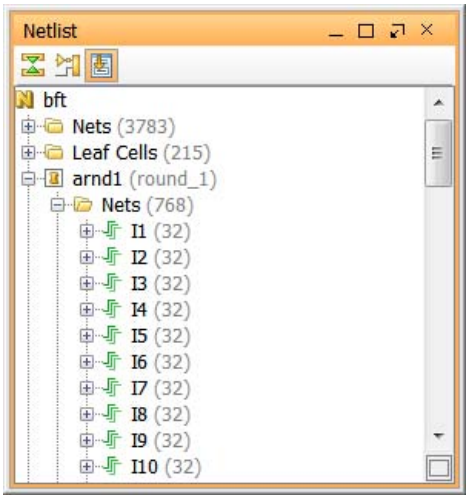


Figure 3-15: **Netlist Window Nets Folder**

Expanding and Collapsing the Logic Tree

To expand or collapse the logic tree:

- Click the expand (+) and collapse (-) buttons to expand or collapse portions of the tree.

  The Netlist tree dynamically expands to display objects selected in other windows. In the Netlist window local toolbar, you can disable this feature by deselecting the **Automatically Scroll to Selected Objects** button  .

- In the local toolbar, use **Collapse All**  buttons to collapse the entire tree.

  When collapsed, the Netlist window displays only the top-level logic modules.

Selecting Elements

In the Netlist window, selection rules work as follows:

- To select multiple elements in the Netlist window, use the **Shift** key or the **Ctrl** key combined with a mouse click. Selected logic is highlighted in the Netlist window.
- When you select logic in a different window, such as the Schematic or Device windows, the logic is cross-selected in the Netlist window. The Netlist tree expands automatically to display all selected logic. You might need to scroll the tree to view all selected logic.
- When you select nets, they highlight in the Device window. Selecting a bus highlights all nets contained within that bus. You can also view nets in the Schematic window.
- To select nets for debug testing, select the **Mark Debug** popup command. For more information, see the *Vivado Design Suite User Guide: Programming and Debugging* (UG908) [Ref 14].

Note: Collapsing the Netlist tree does *not* deselect logic.

Understanding the Netlist Window Icons

The Netlist window uses the following icons to represent the state of netlist logic:

- Bus 
- I/O bus 
- Net 
- I/O net 
- Hierarchical cell (logic) 

- Hierarchical cell (black box)

  Note: Hierarchical cells that do not contain netlists or logic content are interpreted by the Vivado IDE as black boxes. A hierarchical cell might be a black box by design or might be the result of a coding error or missing file.
- Hierarchical cell (assigned to a Pblock)
- Hierarchical cell (black box assigned to a Pblock)
- Primitive cell (assigned to a Pblock)
- Primitive cell (placed and assigned to a Pblock)
- Primitive cell (without assigned placement constraints)
- Primitive cell (with assigned placement constraints)

Using the Device Constraints Window

The Device Constraints window (Figure 3-16) enables you to create, edit, and view DCI_CASCADE and internal VREF constraints. For more information, see the *Vivado Design Suite User Guide: I/O and Clock Planning* (UG899) [Ref 16].

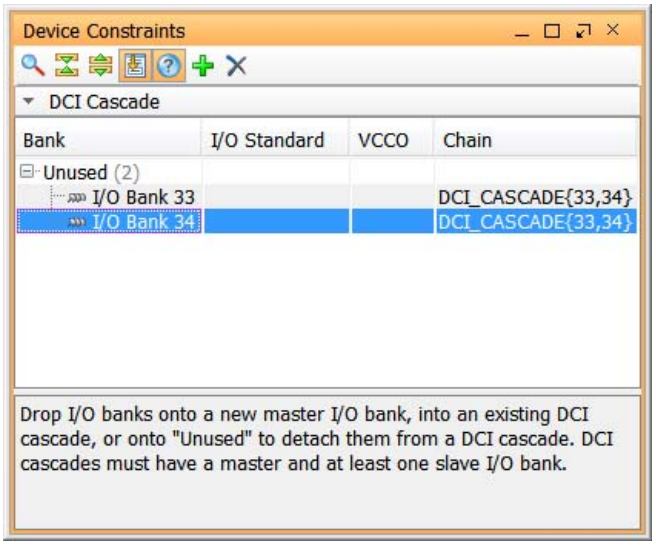


Figure 3-16: **Device Constraints Window**

Device Constraints Window Toolbar Commands

The local toolbar contains the following commands:

- **Show Search**: Opens the search bar to allow you to quickly locate objects in the Device Constraints window. 
- **Expand All**: Expands all hierarchical tree objects to display all elements of the Device Constraints window. 
- **Collapse All**: Collapses all hierarchical tree objects to display only the top-level objects. 
- **Automatically Scroll to Selected Objects**: Scrolls the Device Constraints window to display objects selected in other windows such as the Package Pins or Device windows. 
- **Show Help**: Toggles the display of interactive help information in the description area at the bottom of the Device Constraints window. 
- **Add Constraint**: Adds a new DCI Cascade constraint when two or more banks are selected. Use the Add DCI Cascade dialog box to specify the master bank. 
- **Remove Constraint**: Removes the selected constraint or constraints. 

IMPORTANT: *In the Constraint Selector located beneath the toolbar, select the* ***DCI Cascade*** *or* ***Internal VREF*** *constraint to display in the Device Constraints window.*

Using the Properties Window

The Properties window displays information about selected logic objects or device resources. When you select an object, its properties dynamically display in the Properties window. To open the Properties window, select **Windows > Properties**. Alternatively, you can right-click an object, and select **<ObjectType> Properties** from the popup menu.

The name of the Properties window changes to reflect the selected object. For example, the window is called the BEL Properties window when a BEL is selected or the Clock Region Properties window when a clock region is selected.

The Properties window includes several views to organize information under different categories. The available views and the information they display depend on the type of object selected. For example, Figure 3-17 shows the Cell Properties window with the Properties view displayed for the selected cell.

IMPORTANT: *If multiple objects are selected, the Properties window displays the properties for the most recently selected object. To view and edit properties for multiple objects, use the Property Editor as described in Editing Properties for Multiple Objects in Chapter 2.*

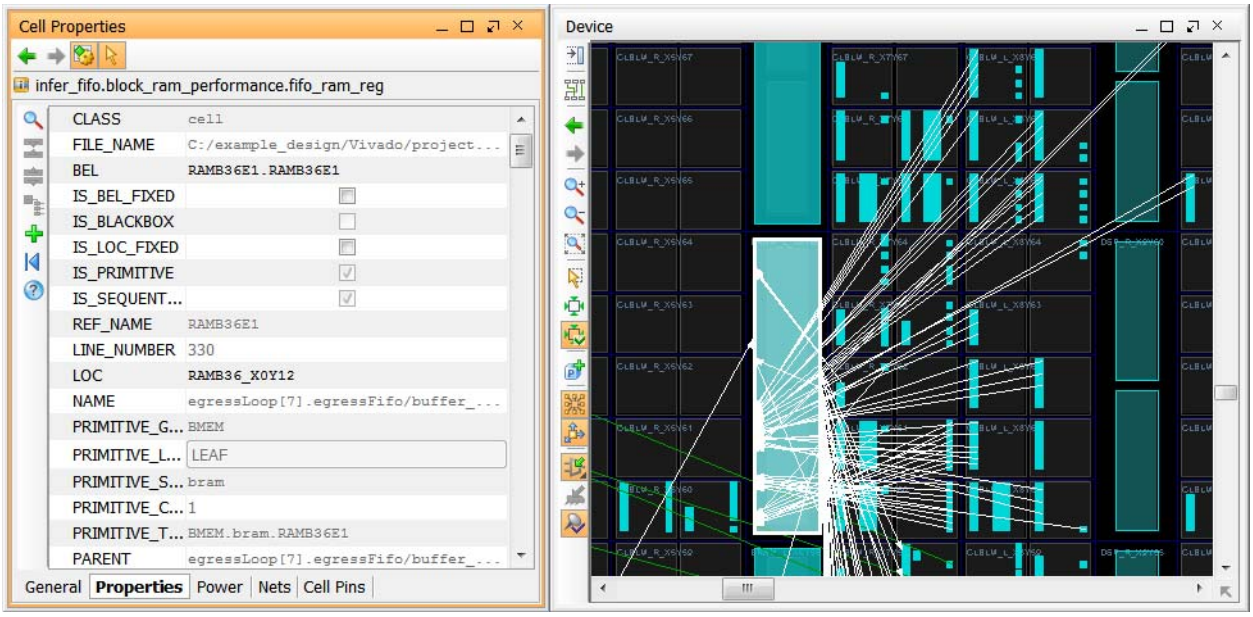


Figure 3-17: **Properties Window**

Properties Window Commands

The Properties window toolbar contains different commands depending on the selected object and the view displayed. Some of the common commands are:

- **Previous Object**: Displays the properties of the previously selected object rather than the currently selected object. You can use this command iteratively to scroll backward through the selected objects. 
- **Next Object**: Scrolls forward through the selected objects to display the object properties. This command is available only after using the **Previous Object** command. 
- **Automatically Update the View When New Objects are Selected**: By default, the Properties window is updated to display the properties of the latest object as new objects are selected. This command toggles the Properties window to auto-update as new objects are selected or to remain static displaying the properties of the currently selected object. 
- **Select/Unselect Object**: Selects or deselects the object reported in the Properties window. 
- **Show Search**: Opens the search bar to allow you to quickly locate objects in the Properties window. 
- **Collapse All**: Collapses the hierarchical tree objects to display only the top-level objects. 
- **Expand All**: Expands all hierarchical tree objects in the Properties window. 
- **Group by Type**: Groups the selected items by type. 
- **Add Properties**: Adds a new property to the selected object. This command is available for certain object types only in the Properties view. 
- **Reset Selected Properties**: Resets a property or object from within one of the views of the Properties window. This command is available for certain object types only and in specific windows. 
- **Show/Hide**: Toggles the display of detailed information in the description area at the bottom of the Properties window. 

Using the Run Properties Window

The Run Properties window, which is one form of the Properties window, displays information about a selected synthesis or implementation run. The title bar label is either Synthesis Run Properties or Implementation Run Properties. Figure 3-18 shows the Implementation Run Properties window for a selected run.

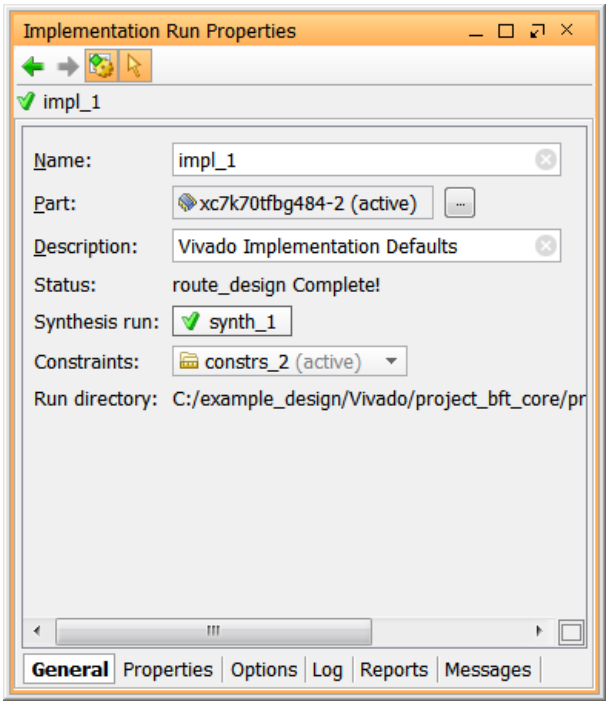


Figure 3-18: **Implementation Run Properties Window**

As you select runs in the Design Runs window, the properties of the run appear in the Run Properties window. You can directly edit most of the properties reported in the Run Properties window. However, for synthesized or implemented designs, the run becomes out-of-date when the edit is applied. Relaunch the run to update the results. To reset a run, select **Reset Runs** from the Design Runs window popup menu.

Run Properties Window Views

The Run Properties window has the following views to display the properties in different ways:

- General View
- Properties View
- Options View

- Log View
- Reports View
- Messages View

General View

The General view reports the configuration of the run and includes the following fields:

- **Name**: Defines the run name.
- **Part**: Displays the target part for the current run and allows you to change the project part for the run. The target part is defined under Project Settings, but can be changed in the Run Properties window. For information on setting the target part for the entire project, see Configuring Project Settings.
- **Description**: Provides a brief description of the current run strategy.
- **Status**: Displays the status of the run.
- **Synthesis Run**: Displays the parent synthesis run of a selected implementation run.

  Note: This is a property of the implementation run and does not appear on synthesis runs.
- **Constraints**: Accepts or changes the constraint set for the run.
- **Run Directory**: Displays the location of the run data.

Properties View

The Properties view displays a table of properties for the selected run.

Note: To obtain this information in Tcl, use the `report_property -all [get_runs impl_1]`command.

Options View

The Options view displays the incremental compile checkpoint, the strategy to use for the run, and the detailed command line options and values for the strategy. It includes the following fields:

- **Set Incremental Compile**: Specifies a placed and routed design checkpoint to use as a reference for the next implementation run. For more information, see the *Vivado Design Suite User Guide: Implementation* (UG904) [Ref 12].

CAUTION! *If you are using a design checkpoint from the active run, you must copy the design checkpoint outside of the run directory prior to using it for incremental compile. When you run incremental compile, the run is reset and all contents are deleted, including the design checkpoint.*

- **Strategy**: You can select a predefined strategy from the drop-down menu. You can also modify the values of the command options. Select a command option to see a description of the command. To edit the option, click the checkbox to enable or disable the option, enter a value, or select the value from the drop-down menu. Modified values show an asterisk (*) next to the option to indicate that the value was changed from the default.

You can use the following popup menu commands:

- **Save Strategy As** to save the new option settings as a strategy for later use in other runs.
- **Refresh** to restore the command options window layout to the default.

RECOMMENDED: *If you modify the run strategy after you launch the run, the run becomes out-of-date. It is recommended that you cancel the run and reset it. For more information, see Vivado Design Suite User Guide: Implementation (UG904) [Ref 12].*

Log View

The Log view displays the same STDOUT command status logs that display in the Log window. The Log view continues to update as commands run. You can use the scroll bar to browse through the command log reports. Click **Pause output** to stop the active reporting. This allows you to scroll more easily and read results while the command is running.

TIP: *Click the **Show Find** button or **Ctrl+F** to use the Find bar to locate specific text.*

Reports View

The Reports view displays report files generated by the Vivado design tools. In the Implementation Run Properties window, select the run, and then select the **Reports** view to display the list of available report files. Double-click on a report to open it. For more information, see the *Vivado Design Suite User Guide: Design Analysis and Closure Techniques* (UG906) [Ref 13].

Messages View

The Messages view displays only the messages generated by the active run.

Using the Selection Window

In the Vivado IDE, you can select objects as follows:

- **Single object**: Click an object to select it in the current window.
- **Secondary object**: By default, when you click a primary object, secondary objects are also selected. For more information, see Setting Selection Rules in Chapter 4.
- **Multiple objects**: Click to select the first object, then press and hold the **Ctrl** key and click to select additional objects.
- **Range of objects**: Click a primary object, then press and hold the **Shift** key and select the last object in a range of elements from a tree or table view.
- **Timing path**: Click a timing path to select the objects within it.
- **All objects**: Use the **Select Area** cursor to select all the objects in an area of a graphical view. Alternatively, most windows support the **Ctrl+A** keyboard shortcut.

The Selection window (Figure 3-19) displays the list of currently selected objects. You can sort, deselect, or mark objects from this window. The list updates dynamically as you manipulate objects. To open the Selection window, select **Window > Selection**.

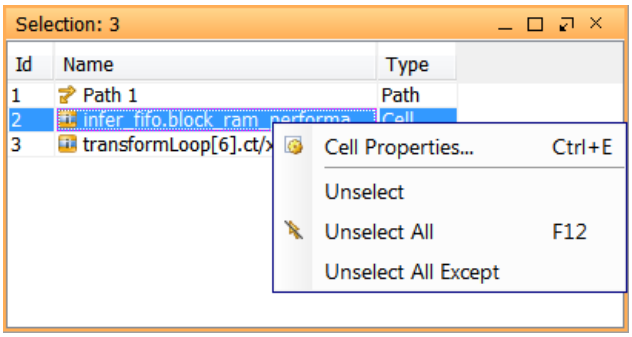


Figure 3-19: **Selection Window**

In the Selection window, you can do the following:

- To sort objects by **Name**, **ID**, or **Type**, click the banner of the sort column.
- To remove selected items from the list, use the following commands from the popup menu: **Unselect**, **Unselect All**, and **Unselect All Except**.
- Select multiple objects using the **Ctrl** and **Shift** keys, or using the **Select Area** command.

  Note: The total number of objects selected displays in the window banner.

Marking and Highlighting Objects

Marking a selected object is helpful when displaying small objects that you want to see in the Device window. Highlighting allows you to keep track of objects after selection changes. You can verify all objects you want to mark or highlight are selected by viewing the Selection window.

To mark and highlight objects:

- To mark selected objects, select the object and then select **View > Mark**.

  Note: Alternatively, you can use the **Mark** command from the popup menu or the **Ctrl+M** keyboard shortcut.

- To highlight selected objects, select the object and then select **View > Highlight** to specify a color for the highlight. The objects are updated with the highlight color, even in the Netlist window (Figure 3-20).

  Note: Alternatively, you can use the **Highlight** command from the popup menu.

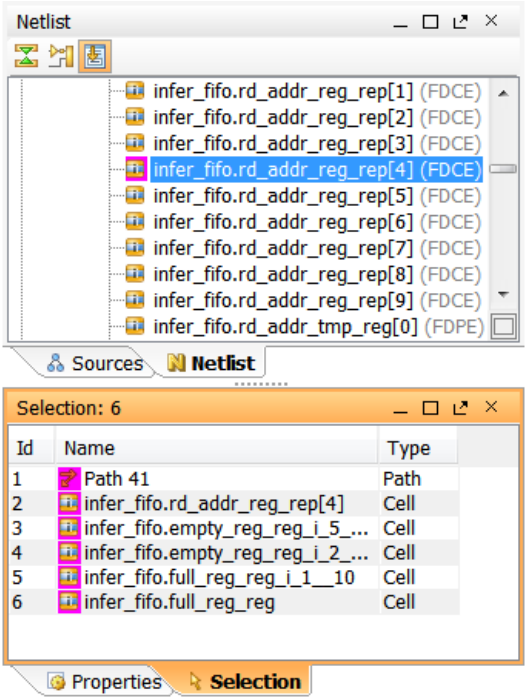


Figure 3-20: **Highlight Selected Objects**

The Mark command is also available in other windows, including the Netlist window, Hierarchy window, and Timing Report window. Figure 3-21 shows a timing path marked from the Timing Report window. The start point of the timing path is marked in green, the end point in red, and the through points are marked in yellow.

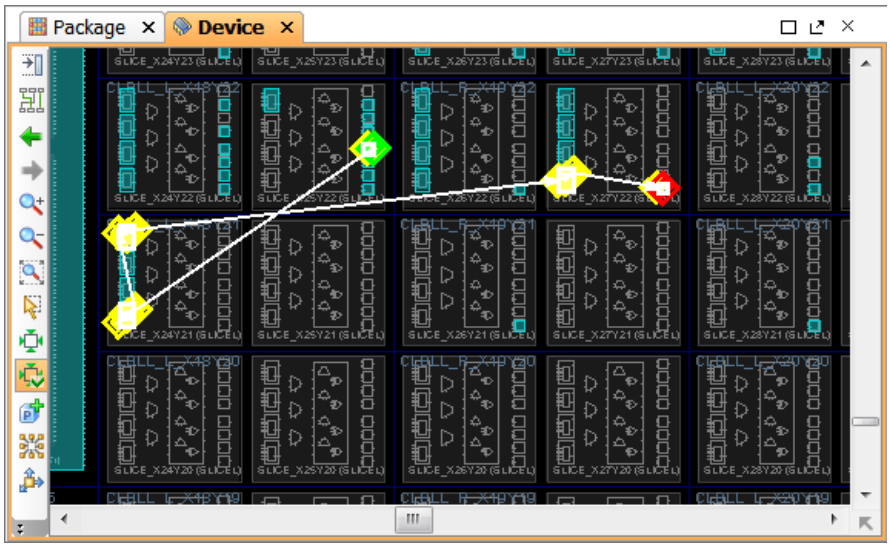


Figure 3-21: **Marked Timing Path Symbols in Device Window**

Unmarking and Unhighlighting Objects

To remove marks or highlights on a selected object or on all objects, select one of the following commands from the **View** menu or popup menu:

- **Unmark** to unmark the selected cell.
- **Unmark All** to unmark all cells.
- **Unhighlight** to remove the highlight from the selected objects.
- **Unhighlight All** to clear all highlights.

Using the Workspace

Windows with a graphical interface and windows that require more screen space, such as the Text Editor or the Package window, display in an area called the workspace. These windows are different than other windows, because you can open more than one window simultaneously to view or compare different information. These windows maximize, minimize, and float like the standard windows in the Vivado IDE, but they also support a split view available through the popup menu on the window tab.

The windows that display in the workspace are:

- Project Summary
- Text Editor
- Device window
- Package window
- Clock Resources
- Schematic window
- Hierarchy window
- Timing Constraints window
- Waveform window
- Diagram window (for block designs)
- Property Editor

You can open multiple windows of the same type within the workspace. For example, if you have one Device window open, you can open a new Device window by selecting **Window > Device**. You can use the two Device windows to display different areas of the device.

Note: Although most windows can be opened from the **Window** menu, the Schematic and Hierarchy windows must be opened after selecting a logic element from another window. For more information, see Using the Hierarchy Window and Using the Schematic Window.

Understanding the Context-Sensitive Cursor

The cursor symbol changes based on the available command mode for a given context and window:

- **Horizontal, vertical, or diagonal stretch bar symbol**: You can stretch Pblock edges and window view borders.
- **Hand symbol**: You can move Pblocks, cells, or drag pan the view.
- **Cross symbol**: You can draw rectangles for zooming in, defining pin assignment areas, or drawing Pblock rectangles.
- **Slashed circle symbol**: You are dragging objects are over illegal placement sites.
- **Move point-to-point symbol**: You are dragging objects over legal placement sites.

Using Mouse Strokes to Zoom and Pan

The following functions are available in the Device, Package, Schematic, Waveform, Histogram, and Hierarchy windows:

- **Zoom Area**: Press and hold the left mouse button while drawing a rectangle from top left to bottom right to define the area to zoom into.
- **Zoom In**: Press and hold the left mouse button while drawing a diagonal line from upper right to lower left. This zooms out the window by a variable amount. The length of the line drawn determines the zoom factor applied. Alternatively, press **Ctrl** and scroll the wheel mouse button up to zoom in.
- **Zoom Out**: Press and hold the left mouse button while drawing a diagonal line from lower left to upper right. This zooms out the window by a variable amount. The length of the line drawn determines the zoom factor applied. Alternatively, press **Ctrl** and scroll the wheel mouse button down to zoom out.
- **Zoom Fit**: Press and hold the left mouse button while drawing a diagonal line from lower right to upper left. The window zooms out to display the entire device.
- **Pan**: Press **Ctrl**, and press and hold the left mouse button while dragging to pan. Alternatively, press and hold the wheel mouse button while dragging to pan.

Using the World View

When zoomed in on a graphical window, such as the Device window, you can open the World view to navigate around the overall design area. The World view displays a less detailed overview of the active graphical window to enable a quick pan of the viewed area. This feature is available in the Device window, Schematic window, Package window, and Hierarchy window when you are zoomed into a small region of the device or design.

To display the World view, click the World view button in the lower right corner of a graphical window, such as the Device window, as shown in Figure 3-22.

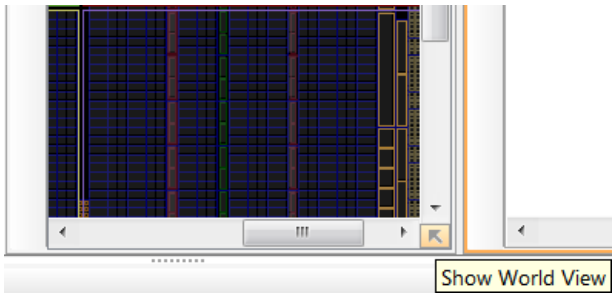


Figure 3-22: **Show World View Button**

The World view reflects the zoom area and the selected objects for the active window. In Figure 3-23, the World view shows the overall Device window that is currently zoomed into the area identified by the navigation rectangle in the World view. You can select and drag the navigation rectangle to reposition the displayed area in the graphical window.

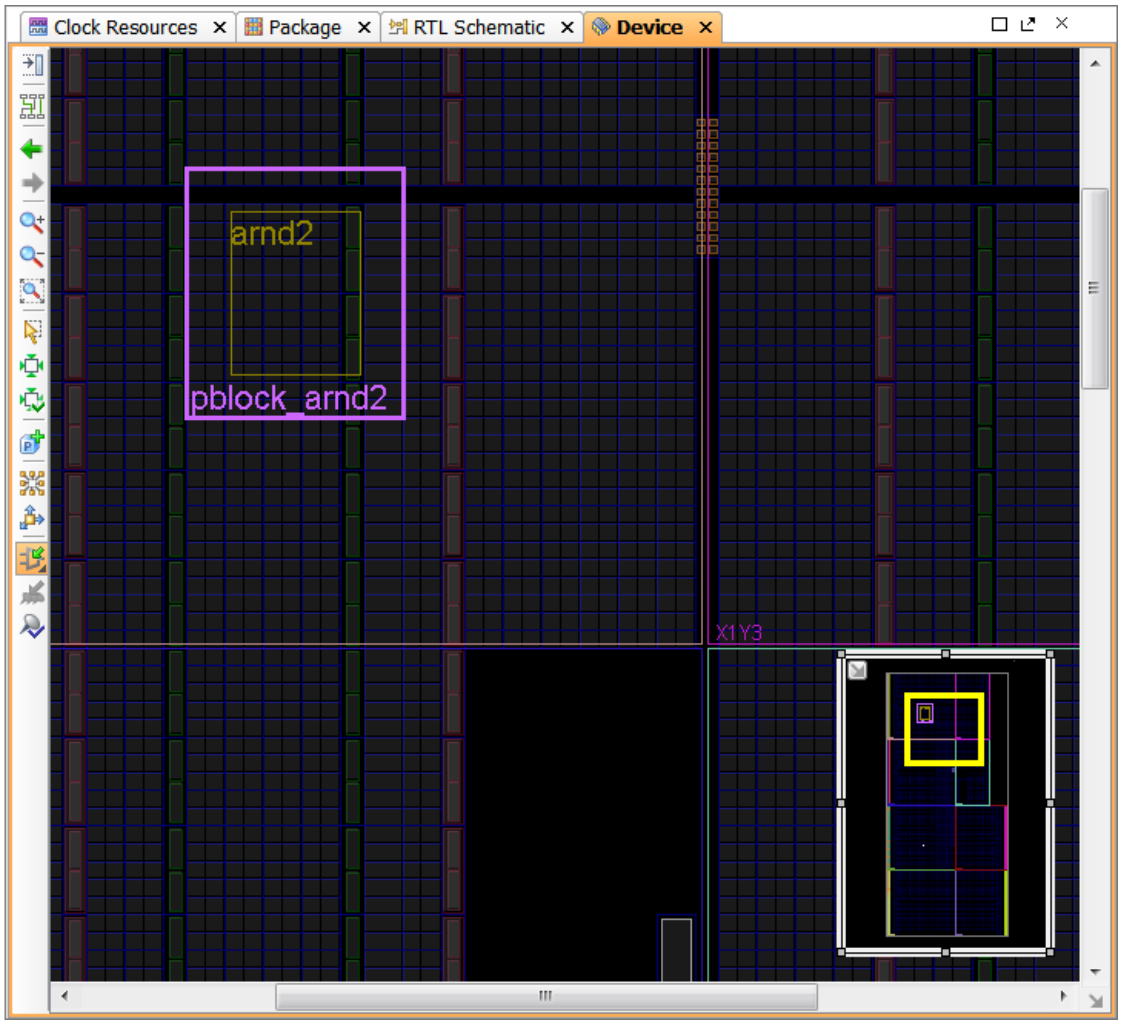


Figure 3-23: **World View**

The World view opens to a default size. To resize the World view, click and drag any of the drag handles on the edge of the World view.

To reposition the World view, click anywhere on the perimeter of the view, except on the drag handles, and drag the view to a new position. Use this feature to reposition the World view anywhere within the limits of the graphical workspace window.

To close the World view, click the downward pointing arrow icon in the view .

Note: When you resize or reposition the World view, it remains at the size and position you chose even when you close and reopen the view.

Printing the Workspace Window

To print the window that is active in the workspace, select **File > Print**. This feature is available for the Device window, Package window, Schematic window, and Hierarchy window.

Splitting the Workspace

You can split the workspace horizontally or vertically to enable multiple simultaneously displayed windows. Each panel acts independently, allowing multiple windows to be docked for viewing.

You can open two windows of the same type, such as two Device windows for viewing different areas of the device or different zoom levels. You can also open two different windows to permit better interaction between the two windows, such as the Device and Package windows (Figure 3-24), or the Device and Clock Resources windows.

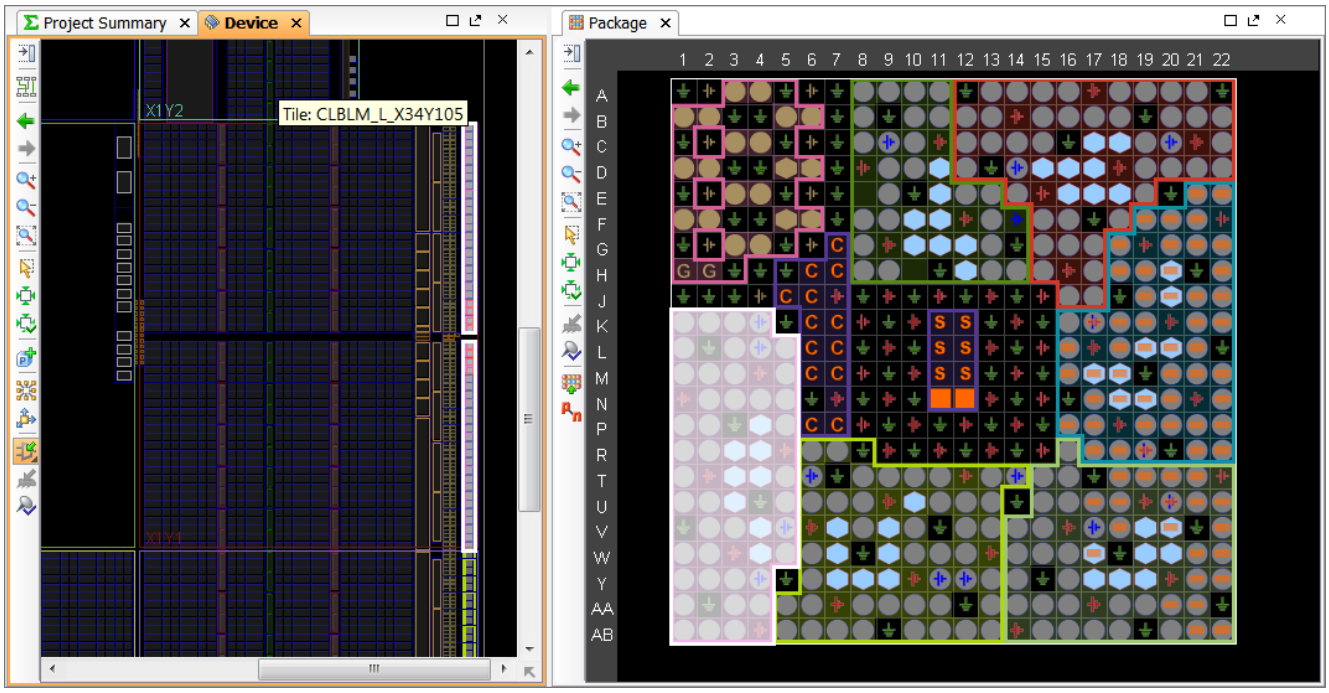


Figure 3-24: **Vertically Tiled Windows**

To split the workspace, use either of the following methods:

- Right-click a window tab, and select **New Horizontal Group** or **New Vertical Group** from the popup menu.

  Note: These commands are available in the workspace windows only.

- Select a window tab, and drag it to the edge of the workspace. A gray rectangle shows a preview of the window location. Position the cursor to arrange the windows as desired, and release the mouse to move the window and split the workspace.

Merging Split Windows

If you split the workspace windows, you might need to merge the windows to better utilize the available viewing area. To collapse the windows back into one tabbed area, do one of the following:

- Right-click a window tab, and select **Move to Previous Tag Group** or **Move to Next Tag Group** from the popup menu.
- Select a window tab, and drag it onto another window. A gray rectangle appears around the entire window, showing how the windows will merge.

Using the Text Editor

The Vivado IDE Text Editor allows you to edit a variety of text files in a context-sensitive editor, including:

- Verilog and Verilog header files
- VHDL files
- Constraint files
- Tcl scripts
- Vivado IDE journal and log files
- Simple text files

TIP: *Some of these files display with context-sensitive syntax highlighting to make it easy to identify keywords and comment lines within the file.*

Opening a Text File

To open a text file in the Text Editor:

- Select **File > Open File**. This command opens a file browser that allows you to navigate to the file and open it for editing.
- Select the file in the Sources window, and select **Open File** from the popup menu.

  Note: Alternatively, you can double-click the file in the Sources window.
- Click the file name from a warning or error message in the Messages window to open the selected file in the Text Editor.
- To open the Vivado IDE journal and log files, select **File > Open Log File** or **File > Open Journal File**.

Creating a New Text File

Select **File > New File** from the main menu to create a new file and open it in the Text Editor. This command opens a file browser so you can navigate to a folder and specify the name of the new file to be created.

TIP: *You can create text files that capture portions of the Tcl Console, compilation logs, or errors or warnings from the Messages window.*

Text Editor Commands

The following commands are available from the local toolbar or popup menu:

- **Save File**: Saves the individually displayed file. 
- **Save File As**: Saves a file to a new name.
- **Undo**: Undoes changes in sequential order. 
- **Redo**: Redoes changes in sequential order. 
- **Cut**: Cuts selected text to the clipboard. 
- **Copy**: Copies selected text to the clipboard. 
- **Paste**: Pastes the contents of the clipboard to the cursor location. 
- **Delete**: Deletes the selected text. 
- **Duplicate Selection**: Copies the selected text and pastes it to the cursor location, immediately in front of the current selection.

- **Show Find/Replace**: Opens the Find field to enter a text string to search for, or search and replace the specified text strings. 

TIP: *Use the **Ctrl+F** keyboard shortcut to search for text strings, and use the **Ctrl+R** keyboard shortcut to replace.*

- **Indent Selection/Unindent Selection**: Inserts or removes a tab space on the selected line or lines.
- **Toggle Line Comments**: Selects a line of text or group of lines, and inserts a line comment symbol at the start of the line. This command removes the line comment symbol if the selected lines currently contain the comment symbol. //

  Note: The comment symbol inserted is contextually dependent on the type of file displayed.
- **Toggle Block Comments**: Adds or removes a block comment (/*...*/) at the start and end of a selected block of text. This command is useful for commenting out a section of text in a single command.

  Note: VHDL files do not support this feature.
- **Select All:** Selects all text in the Text Editor.
- **Toggle Column Selection**: Specifies whether to select a block of text characters as a grid of rows and columns or as lines of text. This command can be toggled on or off. 
- **Tabs**: Specifies whether the Text Editor uses the tab character (\t) or a specified number of spaces when a tab is inserted. This allows the text file to be portable to third-party applications that might not properly handle tab characters.
- **Blank Operations**: Configures the display of tabs, spaces, and special characters for the selected text. If no text is selected, configures the display for the entire document. 
  - **Trim Leading Whitespace**: Removes the leading whitespace for the selected text. This operation works on the entire document if no text is selected.
  - **Trim Trailing Whitespace**: Removes the trailing whitespace for the selected text. This operation works on the entire document if no text is selected.
  - **Trim Leading and Trailing Whitespace**: Combines the actions of the previous two commands, trimming both the leading and trailing whitespace. This operation works on the entire document if no text is selected.
  - **Tab to Space**: Converts each tab character into a space character.
  - **Space to Tab**: Converts each space character into a tab character.

- **Show Special Characters**: Displays special characters that are typically hidden, such as tabs, spaces, and end-of-line characters.
- **Diff with <File_Name>**: Opens the File Compare dialog box (Figure 3-25) and performs a comparison of the current file and the file you select.

  Note: You must have both files you want to compare loaded in the Text Editor.
- **Find in Files/Replace in Files**: Opens the Find in Files dialog box for you to enter text strings for searching the selected files. The Find in Files window displays at the bottom of the Vivado IDE environment with the results of the search. You can also replace the search string with a new string using the **Replace in Files** command. 
- **Change Font Style**: Changes the fonts and font colors associated with the Text Editor.

TIP: *To change the font, you can also select **Tools** > **Options**. In the Vivado Options dialog box, click the **Fonts** category on the left side, and modify the font style and color. You can also change the background color from this dialog box. Click the **Colors** category on the left side, scroll to the Code Editor settings, and modify the colors.*

- **Language Templates**: Opens the Language Templates. For more information, see Using the Language Templates Window. 
- **Insert Template**: Inserts the currently selected Language Template into your text file at the location of your cursor. 

  Note: To enable this command, you must first select a language template.
- **Select**: Selects the current file in the sources window.
- **Define Module**: Opens the Define Module dialog box to specify a new Verilog or VHDL module definition to add to the open text file. When you close the Define Module dialog box, the new module is inserted into the open text file at the location of the cursor.
- **Move Caret to Document Start**: Moves the cursor to the beginning of the file. 
- **Move Caret to Document End**: Moves the cursor to the end of the file. 

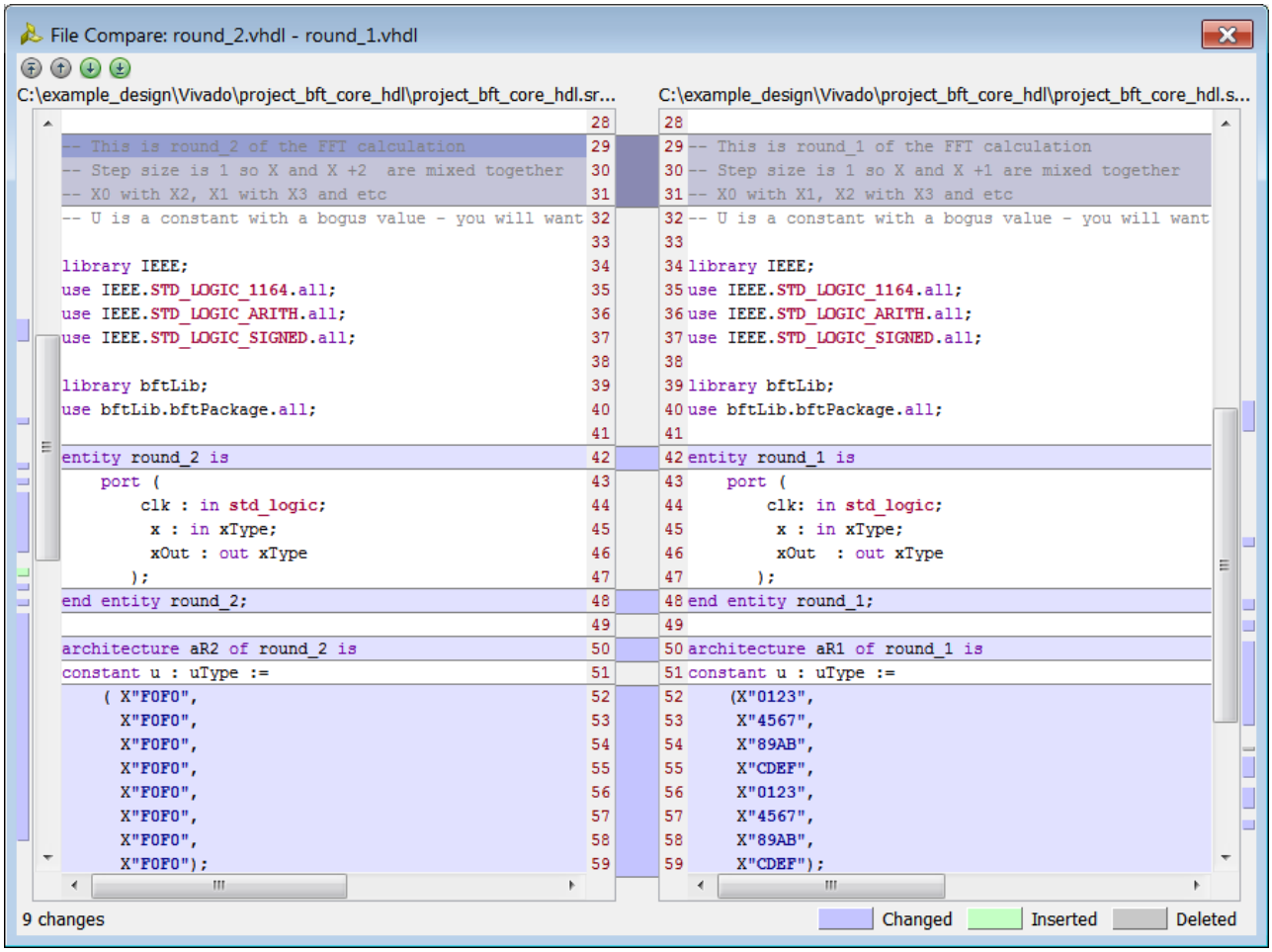


Figure 3-25: **File Compare Dialog Box**

Using the Device Window

The Device window (Figure 3-26) is the main graphical interface used for design analysis and floorplanning. For more information, see:

- *Vivado Design Suite User Guide: I/O and Clock Planning* (UG899) [Ref 16]
- *Vivado Design Suite User Guide: Design Analysis and Closure Techniques* (UG906) [Ref 13]

The Device window displays the FPGA device resources, including FPGA logic, clock regions, I/O pads, BUFGs, MMCMs, Pblocks, cell locations, and net connectivity. The locations on the device to which specific logic can be assigned are called sites.

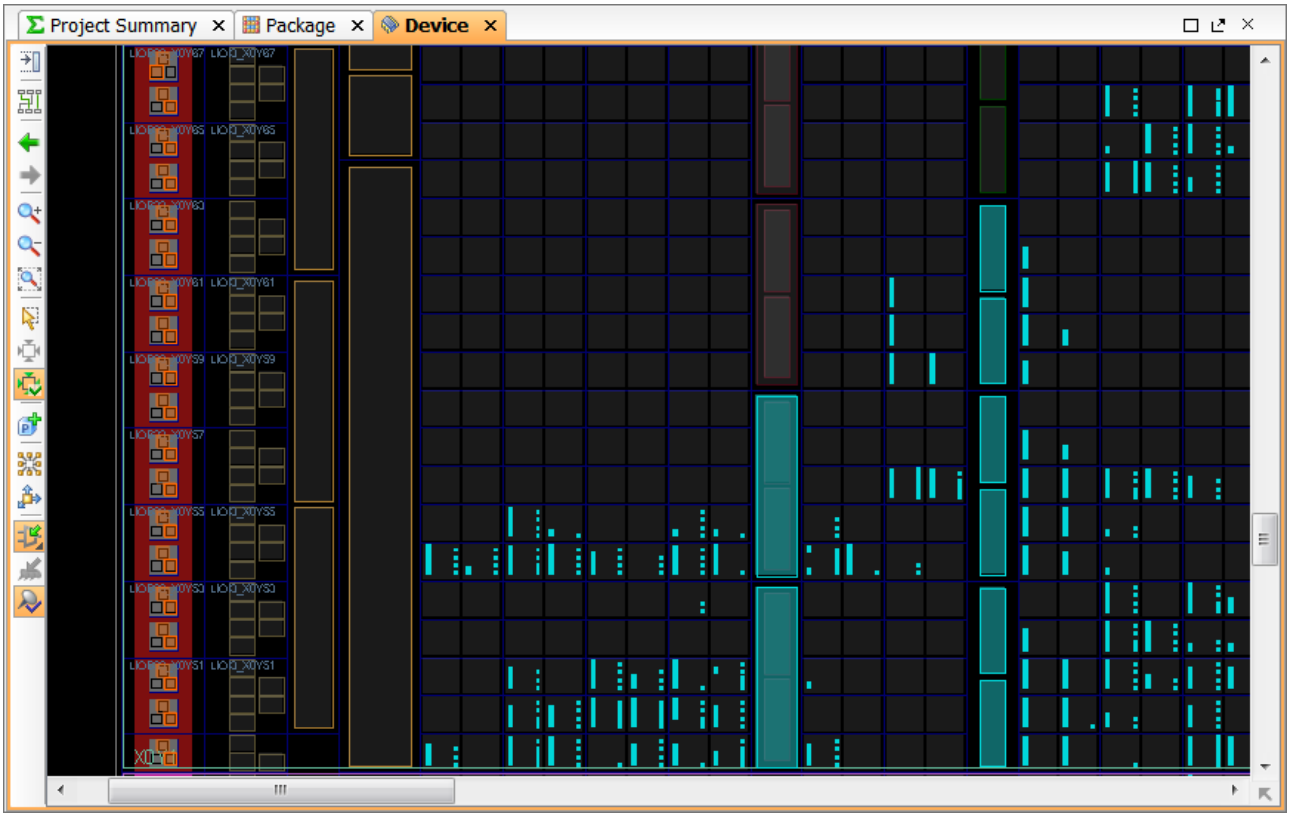


Figure 3-26: **Device Window**

The amount of logic object detail displayed is determined by the selected zoom level. The more you increase the zoom level, the more logic object detail displays. This is especially true when viewing the **Routing Resources** for the entire device, where the logic displays in an abstract form to show approximate placement and congestion. As you zoom in, you can see exact placement and routing.

Zooming and Panning

To zoom and pan:

- Use the zoom commands in the popup menu and local toolbar.
- Hold down the left mouse button, and drag the cursor in the Device window to zoom into an area or to zoom out. For more information, see Using Mouse Strokes to Zoom and Pan.
- Use scroll bars and dynamic pan capabilities to pan the viewable area of the device.

Getting Information about Device Resources

When you place the cursor over an object in the Device window, a tooltip identifies the object. The Properties window displays object properties for selected sites or logic cells. To search for specific device resource sites, use **Edit > Find**. For more information, see Finding Objects in Chapter 2.

The Device window also provides dynamic feedback during device exploration and design modification. For example, if you attempt a logic resource assignment that is illegal, the dynamic cursor changes to allow you to make adjustments. For more information, see Understanding the Context-Sensitive Cursor.

Device Window Toolbar Commands

The local toolbar contains the following commands:

- **Device Options**: Configures the display, color, and fill of specific layers in the Device window. For more information, see Setting Device Window Display Options. 
- **Routing Resources**: Displays routing resources in the Device window. 
- **Previous Position**: Resets the Device window to display the prior zoom and coordinates. 
- **Next Position**: Returns the Device window to display the original zoom and coordinates after Previous Position is used. 
- **Zoom In**: Zooms in the Device window. 
- **Zoom Out**: Zooms out the Device window. 
- **Zoom Fit**: Zooms out to fit the whole device into the display area of the Device window. 
- **Select Area**: Selects the objects in the specified rectangular area. 
- **Fit Selection**: Redraws the Device window to display the currently selected objects. This is useful when selecting objects in another window, such as the Netlist window, and you want to redraw the display around those selected objects. 

- **Autofit Selection**: Automatically redraws the Device window around newly selected objects. This mode can be enabled or disabled.
- **Draw Pblock**: Places the cursor in Draw Pblock mode (crosshair) allowing you to create a new Pblock rectangle to place cells.
- **Show I/O Nets**: Toggles the display of I/O connectivity to placed logic or Pblocks.
- **Show Cell Connections**: Displays connectivity automatically for selected objects. This button toggles the mode on and off.
- **Cell Drag & Drop Mode**: Sets the manner in which cells placed onto the device are assigned placement constraints. The button displayed reflects the currently selected mode:
  - **Create BEL Constraint Mode**: Assigns a LOC and BEL constraint to the cell being placed. This fixes the cell to the specified BEL within the slice.
  - **Create Site Constraint Mode**: Assigns a LOC placement constraint to the cell being placed. This fixes the cell to the specified Slice, but allows it to float to available BELs within the slice.
  - **Assign Cell to Pblock Mode**: Assigns logic cells to Pblocks. This is the default mode. Use this mode whenever possible to ensure proper command behavior.
- **Place Ports Mode**: Determines how I/O ports are placed onto the device from the I/O Ports window or Netlist window. The button displayed reflects the currently selected mode.
  - **Place I/O Ports in an I/O Bank**: Assigns the currently selected ports onto pins in the specified I/O Bank.
  - **Place I/O Ports in Area**: Assigns the currently selected ports onto pins in the specified area.
  - **Place I/O Ports Sequentially**: Assigns the currently selected ports individually onto pins.
- **Autocheck I/O Placement**: Toggles Automatic enforcement of interactive I/O placement DRCs. When this mode is enabled, interactive I/O port placement will be checked against enabled design rules.

Understanding the Device Resource Display

The Vivado IDE displays the resources contained in a selected device in the Device window. Graphical sites display and are available for all of the device-specific FPGA resources. The level of detail for displaying the device resources depends on the zoom level within the Device window.

Some resources, such as specific slice resources, are not visible until zoomed in quite close onto the FPGA logic. Some resources, such as Clock Regions and I/O Banks appear even when viewing the whole device. In addition, the **Device Window Options** command lets

you turn on or off the display of specific objects or resources in the Device window. For more information see Setting Device Window Display Options.

The Vivado IDE Device window displays resources as follows:

- The I/O pads and clock objects display around the periphery and down the center of the device.
- I/O banks display as thin color-shaded rectangles just outside the row of I/O pads.
- High-performance (HP) and high-range (HR) I/O banks display in the Device window as right or left angled lines.
- Available I/O bank sites display as color-filled I/O bank rectangles.
- Some devices have unbonded I/O banks that display rectangles with a white X.
- The I/O clock pads display as filled-in rectangles.
- All clock resources, such as BUFG, BUFGCTRL, BUFR, and BUFHCE components, show in the Device window.
- When you select an I/O bank or clock region, the available device resources display in the Properties window.
- The interior of the device is broken up into smaller rectangles called tiles. Tiles are placement sites for the different types of logic primitives for the architecture.
- A tooltip identifies each site in the Device window when you hover the cursor over a logic site.

In the Device window, you can assign primitive logic cells to the appropriate displayed sites. At a lower zoom level, the placed cells appear as rectangles within a slice. When you increase the zoom level, the logic symbols display. You can assign logic to specific sites that generate LOC placement constraints. Using BEL constraints, you can assign cells to specific device resources in a slice.

Setting Device Window Display Options

To set Device window display options, click the **Device Options** toolbar button  . The Device Options include the following views:

- Layers
- Colors
- General

After you finish setting options, click the Close (**<<**) button in the upper right corner. The Vivado IDE stores your settings and reloads them each time the tool is launched.

TIP: *To restore the options to the default settings, click the Reset button in the upper right corner*  .

Layers

The options in the Layers view (Figure 3-27) define what device and design objects are displayed in the Device window. You can control the level of detail displayed in the Device window, which is especially useful when the display is overcrowded with information.

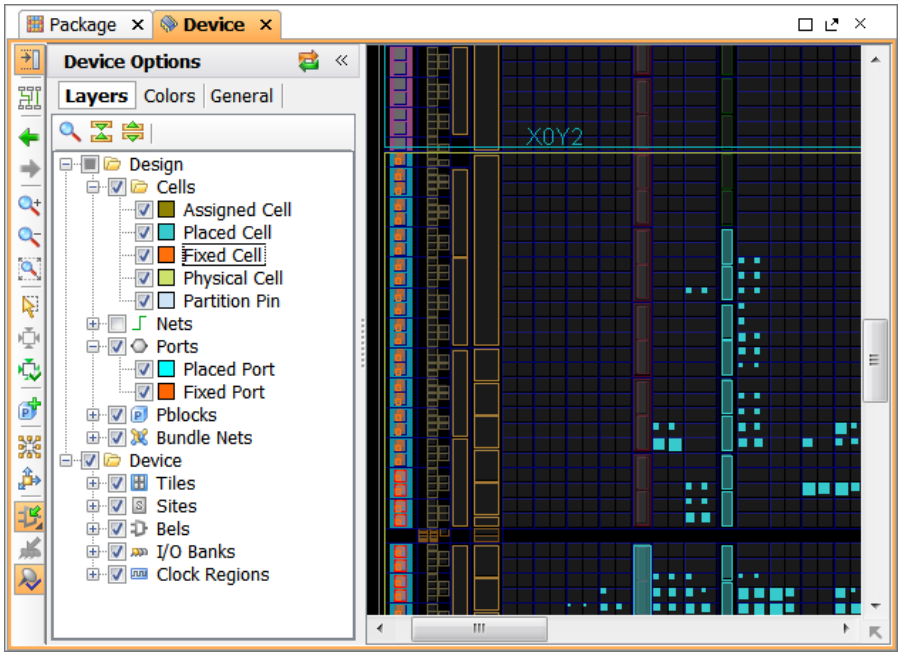


Figure 3-27: **Device Options—Layers**

Following are the two primary branches:

- **Design objects**: Elements from the design sources, such as cells, nets, and ports that are placed on the device.
- **Device objects**: Resources on the device such as I/O banks, clock regions, and tiles on which design objects can be placed.

In the Layers view, you can do the following:

- Click the plus (+) sign to expand, or the minus (-) sign to collapse the levels of the tree view to allow you to see the layer or object.
- Click the checkbox to enable or disable the layer or object for display in the Device window. A check mark indicates the currently displayed layer. You can display or hide groups of objects or layers by clicking the category of the layers. Select individual layers or objects directory to display or hide them.

  Note: If you cannot see a specific object or layer of the Device window, check the configuration of the Device Options command to see if the design object or device resource is currently hidden.

TIP: *You can use the **Shift** key to select multiple layers, and use the **Space** bar to toggle the selected layers on or off.*

Colors

The options in the Colors view (Figure 3-28) change the color and fill values for elements in the Device window. For information, see Changing Colors in Chapter 4.

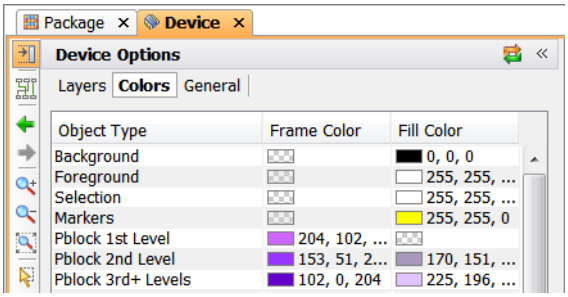


Figure 3-28: **Device Options—Colors**

General

The options in the General view (Figure 3-29) define the display characteristics of the net connections on the device.

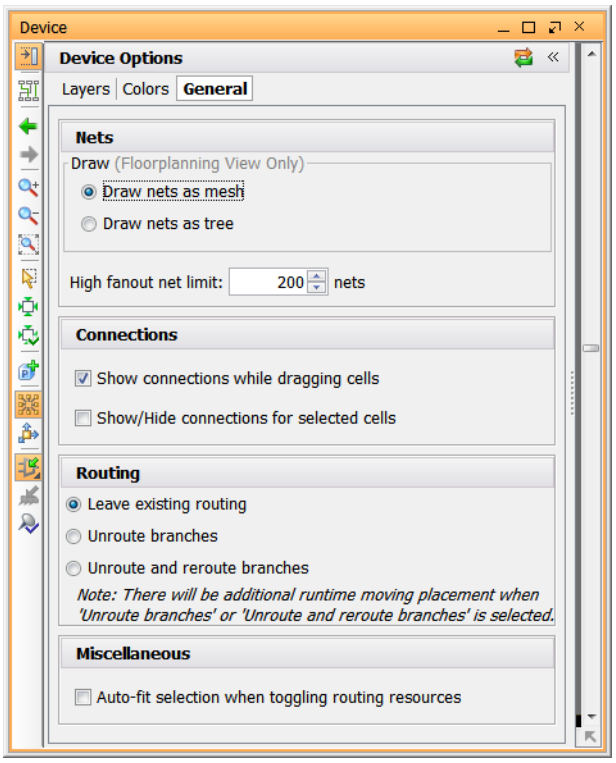


Figure 3-29: **Device Options—General**

The following options are available:

- **Nets**
  - **Draw nets as mesh**: Pin to pin connections making a mesh of all connected pins (Figure 3-30).

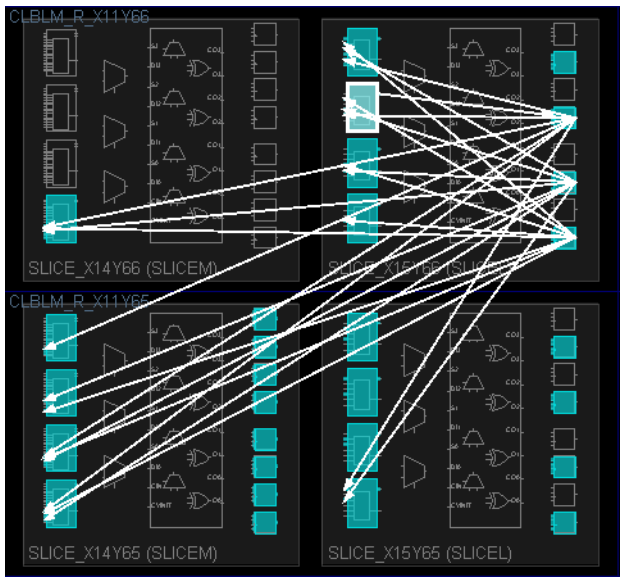


Figure 3-30: **Nets Drawn as Mesh**

  - **Draw nets as tree**: A branching structure, grouping nearby pin connections (Figure 3-31).

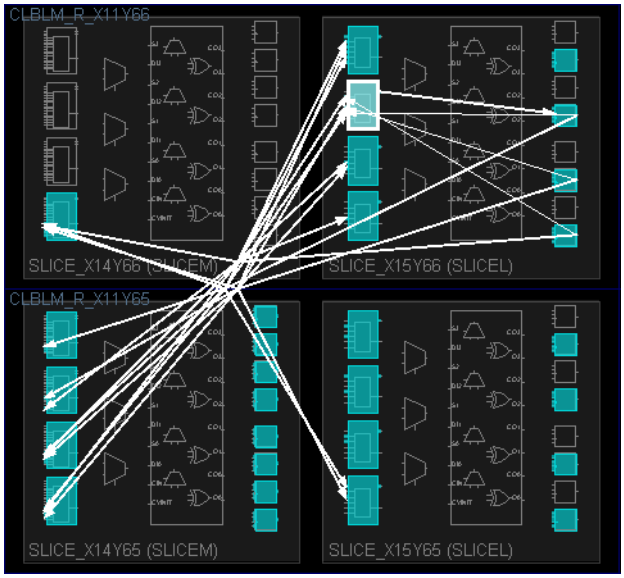


Figure 3-31: **Nets Drawn as Tree**

- **High fanout net limit**: Limits the number of connections a pin can have in order to be displayed. The nets on a pin with a fanout greater than the specified number of connections are *not* displayed.
- **Connections**
  - **Show connections while dragging cells:** Toggles the display of nets connected to the selected cell while dragging and placing the cell in the device window.
  - **Show/Hide connections for selected cells:** Toggles the display of nets connected to the selected cell.
- **Routing**
  - **Leave existing routing**: Leaves all routes intact. In most cases, this results in antennas and unrouted branches.
  - **Unroute branches**: Unroutes branches to the original placement. In most cases, this results in unrouted branches.
  - **Unroute and reroute branches**: Unroutes branches to the original placement and reroutes the branches to the new placement. In most cases, this results in completely routed nets.
- **Miscellaneous**
  - **Auto-fit selection when toggling routing resources**: Adjusts the display in the window to keep the selection visible when routing resources are turned on and off.

Selecting Clock Regions

Clock regions display as large rectangles, indicating the periphery of the device clock regions. These outlines can help guide floorplanning for critical circuitry. In the Device window, you can:

- Select the clock regions in the Clock Regions window.
- Select and specify that clock regions display their resource statistical properties.
- View the clock placement statistics after importing the implementation results.
- Change the display color of Clock Regions in the Device window using **Tools > Options**, and changing the Colors settings. For more information, see Specifying Colors in Chapter 4.

Note: When you select the clock region, the Vivado IDE also selects the associated I/O banks and clock related logic sites.

Opening Multiple Device Windows

You can open multiple Device windows for the same floorplan. This enables you to work on different areas of the device. For more information, see Splitting the Workspace.

Using the Package Window

The Package window (Figure 3-32) displays the physical characteristics of the target Xilinx part. This window is used primarily during the I/O planning process or during port placement. Pin types display in different colors and shapes for better visualization. For information on using the Package window for I/O planning, see the *Vivado Design Suite User Guide: I/O and Clock Planning* (UG899) [Ref 16]. To open the Package window, select **Window > Package**.

Note: The Package window opens automatically when using the I/O Planning layout.

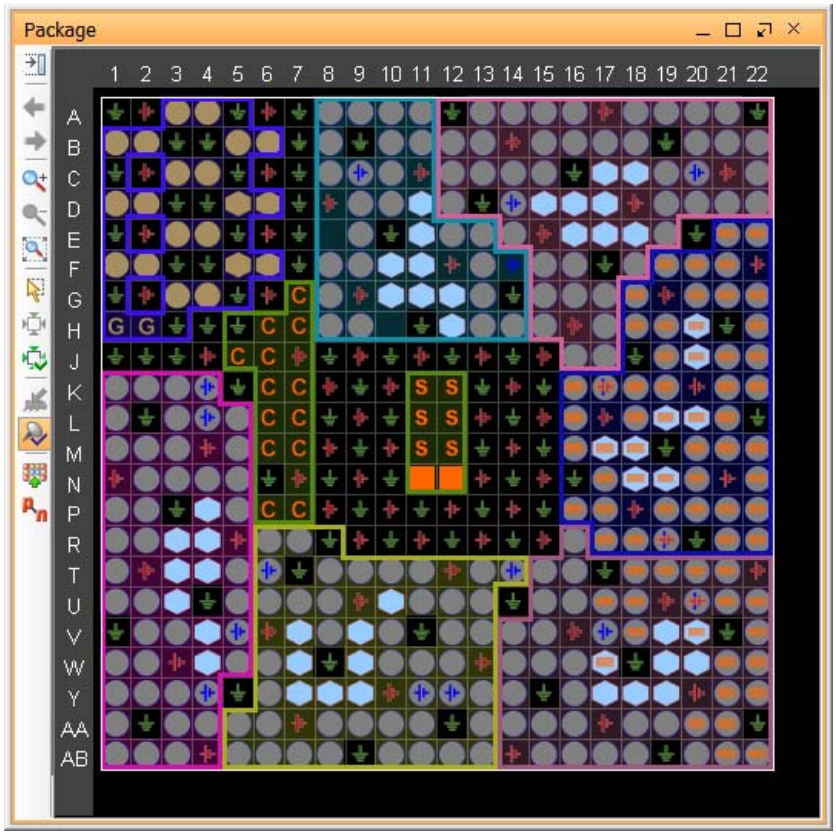


Figure 3-32: **Package Window**

In the Package window, you can do the following:

- Drag ports into the Package window for assignment, and reassign placed cells to other I/O pins within the Package window.

  Note: Autocheck I/O Placement is on by default, allowing only legal pin placement during drag and drop.

- Visualize pins and I/O banks as follows:
  - V_{CC} and GND pins show as red V_{CC} symbols and green GND symbols.
  - Clock-capable pins display as hexagon pins.
  - User and multipurpose pins display as circles.
  - Colored regions display the different I/O banks on the device.
- Move the cursor within the Package window to show the I/O pin coordinates actively on the top and left sides of the window.
- Hold the cursor over a pin to show a tooltip that displays the pin information. Additional I/O pin and bank information displays in the Information bar located at the bottom of the environment in the status bar.
- Select I/O pins or banks to cross probe between the Device and Package windows, and see pin information in the Package Pins Properties window.

Package Window Toolbar Commands

The local toolbar contains the following commands:

- **Package Options**: Configures the display of specific objects in the Package window. For more information, see Setting Package Options. 
- **Previous Position**: Resets the Package window to display the prior zoom and coordinates. 
- **Next Position**: Return the Package window to display the original zoom and coordinates after Previous Position is used. 
- **Zoom In**: Zooms in the Package window. 
- **Zoom Out**: Zooms out the Package window. 
- **Zoom Fit**: Zooms out to fit the whole package into the display area. 
- **Select Area**: Selects the objects in the specified rectangular area. 
- **Fit Selection**: Redraws the Package window to display the currently selected objects. This is useful when selecting objects in another window, such as the I/O Ports or Package Pins window, and you want to redraw the display around those selected objects. 

- **Autofit Selection**: Automatically redraws the Package window around newly selected objects. This mode can be enabled or disabled.
- **Place Ports Mode**: Determines how I/O ports are placed onto the package from the I/O Ports window or Netlist window. The button displayed reflects the currently selected mode.
  - **Place I/O Ports in an I/O Bank**: Assigns the currently selected ports onto pins on the specified I/O Bank.
  - **Place I/O Ports in Area**: Assigns the currently selected ports onto pins in the specified area.
  - **Place I/O Ports Sequentially**: Assigns the currently selected ports individually onto pins.
- **Autocheck I/O Placement**: Toggles automatic enforcement of interactive I/O placement DRCs. When this mode is enabled, interactive I/O port placement will be checked against enabled design rules.
- **Show Bottom/Top View**: Toggles the Package window to display the package pins as viewed from the top or as viewed from the bottom.
- **Show Differential I/O Pairs**: Displays the differential pair pins in the Package window.

Setting Package Options

The Package Options (Figure 3-33) define the layers and objects that are visible in the Package window. To set Package window display options, click the **Package Options** toolbar button .

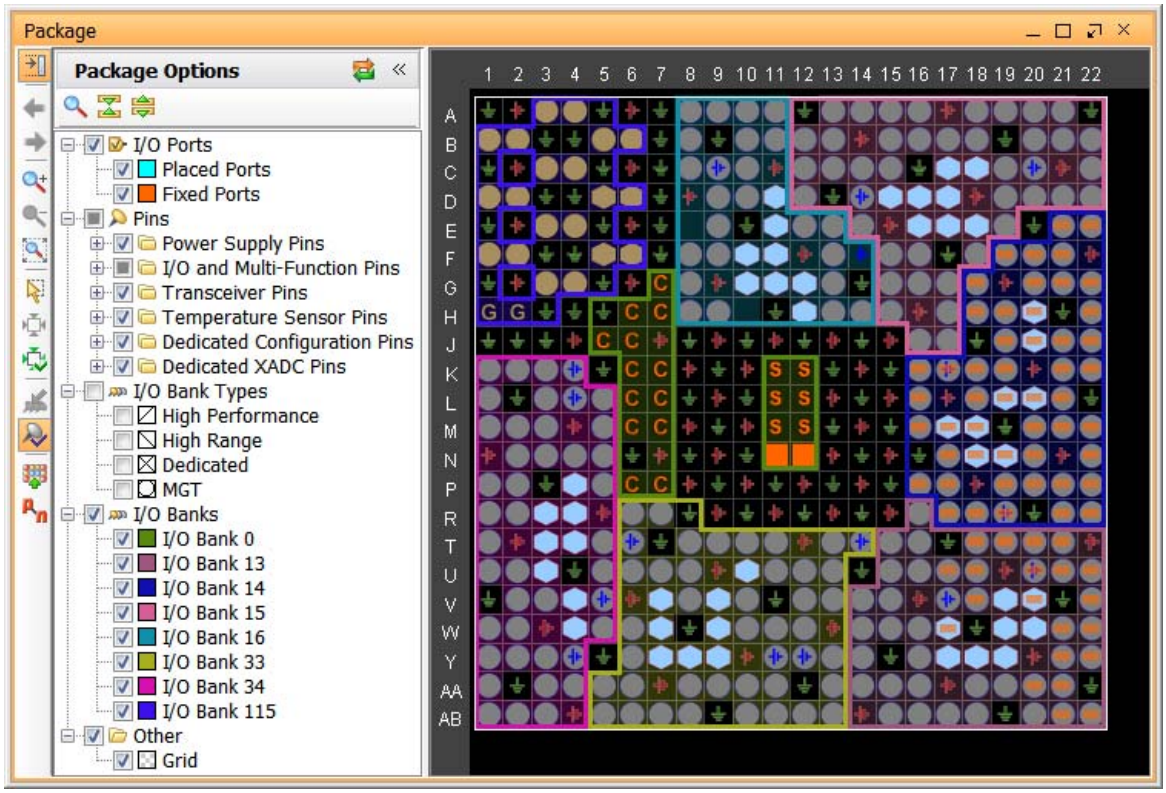


Figure 3-33: **Package Options**

The available layers are listed hierarchically in a tree view:

- Click the plus sign (+) to expand or the minus sign (-) to collapse the levels of the tree view to see the layer of interest.
- Click a checkbox to enable or disable the layer for display in the Package window. A check mark indicates the currently displayed layer. You can display or hide:
  - Groups of objects by clicking the category of the objects
  - Individual objects by selecting the item directly

IMPORTANT: *If a specific pin is not visible in the Package window, you cannot assign a port to it. Check that both the pin and the I/O block it is contained in are selected for display in the Package Options.*

TIP: *The Package Options provide a legend of the icons used for both dedicated and multifunction pins.*

In the Package window, the three primary categories of layers are:

- I/O Ports Layer
- Pins Layer
- I/O Banks Layer

I/O Ports Layer

The I/O Ports layer shows the ports in the design that are currently placed in either a fixed or unfixed state. The design might have currently unplaced ports that are not displayed in the Package window.

Pins Layer

The Pins layer includes the available package pins grouped into specific categories, such as multifunction pins, power pins, and unconnected pins. Pins display as follows:

- Power pins display separately from the I/O banks.
- Multifunction pins display as part of the I/O bank they are contained in, and display with symbols representing their available functions. For example:
  - Basic I/O pins display as gray circles by default. 
  - Clock capable pins display as blue hexagons by default. 
  - V_{REF}, V_{RP} and V_{RN} pins display with a small power icon by default. 
  - The remaining pins display with an asterisk (*) and are not displayed by default.

I/O Banks Layer

The I/O Banks layer shows the sites of the pins for each of the banks on the device and the sites of the GTX pins. Each I/O bank and GT bank is color-coded to allow you to easily differentiate the different banks of pins.

Xilinx FPGAs typically offer both high performance (HP) and high range (HR) I/O banks. To distinguish high performance banks, enable the **High Performance** layer as shown in Figure 3-34.

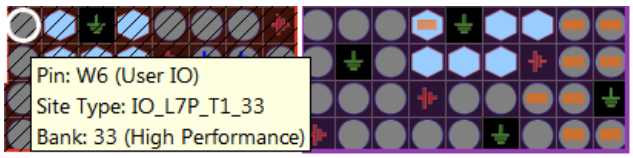


Figure 3-34: **High-Performance I/O Bank**

The display of a specific pin in the Package window depends on a combination of layers that represent the pin in the Package Options. For example, if the I/O banks are not displayed, ground pins are displayed but the user I/O and multifunction pins are not displayed even though both are selected under the Pins heading of the Package Options.

TIP: *Turning off a bank layer is an easy way to prevent pin assignment. This enables you to reserve a bank for later use or show that a bank is full.*

Opening Multiple Package Windows

You can open multiple Package windows for the same design. This enables you to display and work on different areas of the package. For information, see Splitting the Workspace.

Using the Schematic Window

You can generate a Schematic window for any level of the logical or physical hierarchy. You can select a logic element in an open window, such as a primitive or net in the Netlist window, and use the **Schematic** command in the popup menu to create a Schematic window for the selected object. An elaborated design always opens with a Schematic window of the top-level of the design, as shown in Figure 3-35. In the Schematic window, you can view design interconnect, hierarchy structure, or trace signal paths for the elaborated design, synthesized design, or implemented design.

For more information on analyzing RTL netlists, see the *Vivado Design Suite User Guide: Synthesis* (UG901) [Ref 10]. For more information on synthesized netlist analysis, see the *Vivado Design Suite User Guide: Design Analysis and Closure Techniques* (UG906) [Ref 13].

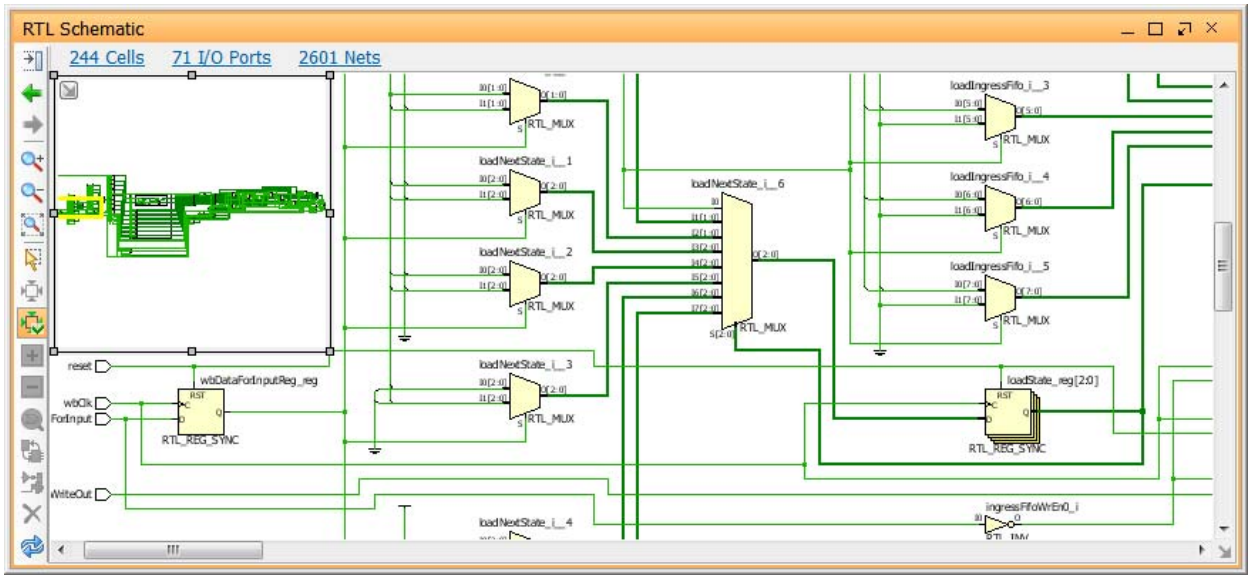


Figure 3-35: **Schematic Window**

Creating a Schematic Window

To create a Schematic window:

1. Select one or more logic elements in an open window, such as the Netlist window.
2. Right-click and select **Schematic** from the popup menu, select the **Schematic** toolbar button , or press **F4**.

   The Schematic window displays the selected logic cells or nets. If only one cell is selected, a schematic symbol for that module is displayed, as shown in Figure 3-36.

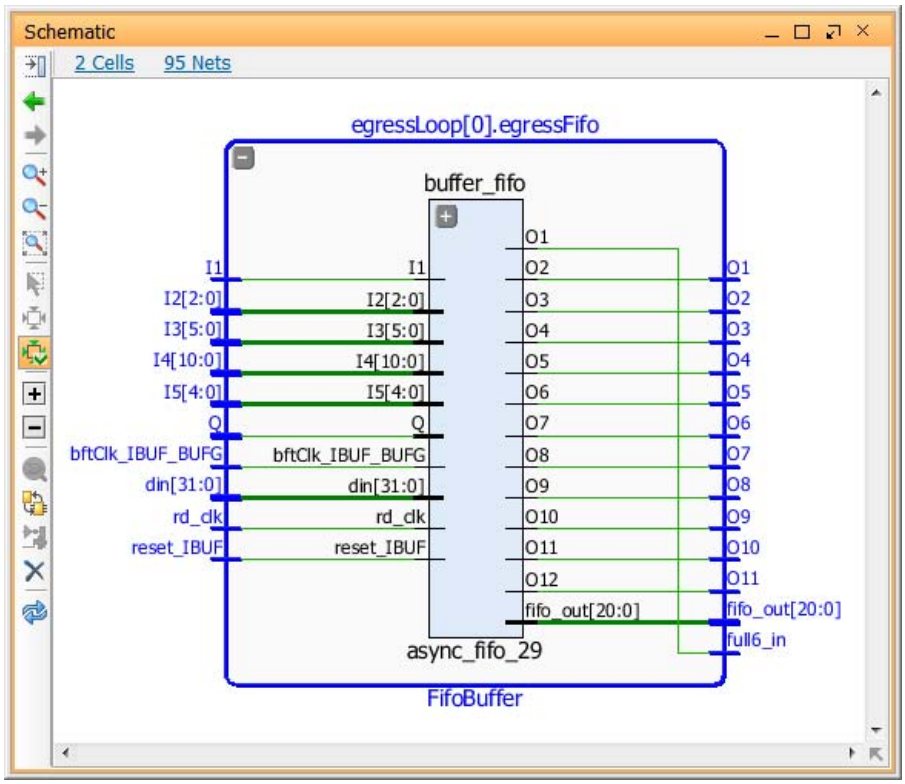


Figure 3-36: **Schematic Symbol**

In the Schematic window, you can find and view objects as follows:

- The links at the top of the schematic sheet, labeled **Cells**, **I/O Ports**, and **Nets**, open a searchable list in the Find Results window, making it easier to find specific items in the schematic.
- When you select objects in the Schematic window, those objects are also selected in all other windows. If you opened an implemented design, the cells and nets display in the Device window.

Schematic Window Toolbar Commands

The local toolbar contains the following commands:

- **Schematic Options**: Configures the display of specific objects in the Schematic window. For more information, see Setting Schematic Window Options. 
- **Previous Position**: Resets the Schematic window to display the prior zoom, coordinates and logic content. 
- **Next Position**: Returns the Schematic window to display the original zoom, coordinates and logic content after Previous Position is used. 
- **Zoom In**: Zooms in the Schematic window. 
- **Zoom Out**: Zooms out the Schematic window. 
- **Zoom Fit**: Zooms out to fit the whole schematic into the display area. 
- **Select Area**: Selects the objects in the specified rectangular area. 
- **Fit Selection**: Redraws the Schematic window to display the currently selected objects. This is useful when selecting objects in another window and you want to redraw the display around those selected objects. 
- **Autofit Selection**: Automatically redraws the Schematic window around newly selected objects. This mode can be enabled or disabled. 
- **Expand all logic inside selected cell**: Expands a hierarchical module from the symbol view to the logic view. 

  Note: You can also expand hierarchical modules directly from the schematic by clicking the expand button  on the schematic symbol.
- **Collapse all logic inside selected cell**: Collapses a hierarchical module from the logic view to the symbol view. 

  Note: You can also collapse the expanded hierarchical block directly from the schematic by clicking the collapse button  on the hierarchical block.
- **Magnify**: Displays a detailed popup view of the selected bus pin. 

  Note: Alternatively, you can press **Ctrl** and double-click a bus pin.
- **Toggle autohide pins for selected cell**: Toggles the pin display on selected hierarchical modules. Higher levels of the hierarchy display as concentric rectangles without pins, when a Schematic window is generated, as shown in Figure 3-36. In most cases, the lack of pins makes the Schematic window more readable. However, you can display the pins for selected cells as needed. 
- **Add selected elements to schematic**: Recreates the Schematic window with the newly selected elements added to the existing schematic. 

- **Remove selected elements from the schematic**: Recreates the Schematic window with the currently selected elements removed from the existing schematic.
- **Regenerate Schematic**: Redraws the active Schematic window.

Expanding Logic from Selected Cells and Pins

With a selected schematic cell or a selected pin on a schematic cell, you can:

- Individually expand or collapse module pins and logic.
- Selectively expand the logic either from individual pins, cells, or the entire logic content inside or outside the module.

You can expand or collapse logic contained either inside a selected module or outside in the next level of hierarchy. You can expand a single module or multiple selected modules. From the popup menu, the commands to expand schematic logic are:

- **Expand/Collapse > Expand Inside**: Displays the schematic hierarchy inside of a selected cell. The Vivado IDE regenerates the Schematic window to expand the contents of the selected cell. You can also use the expand button available within the schematic.

  Note: This command is not available if the selected cell is a primitive within the design hierarchy.
- **Expand/Collapse > Collapse Inside**: Hides the expanded contents of a selected hierarchical block. You can also use the collapse button available within the schematic.
- **Expand/Collapse > Expand Outside**: Displays the hierarchy upward from a selected cell. The Vivado IDE regenerates the Schematic window to expand the hierarchy up from the selected cell.

  Note: This command has no effect if the selected cell is at the top-level of the design hierarchy.
- **Expand/Collapse > Collapse Outside**: Hides the expanded hierarchy outside the selected cell.

In addition, you can expand schematic logic by double-clicking as follows:

- Double-click a pin of a cell to trace the net down into, or up out of the hierarchy. A pin is displayed on the schematic symbol with a stub inside and outside of the symbol, as shown for `O1` and `fifo_out[31:0]` in Figure 3-37. This reflects the ability to expand inside or outside the symbol.

  Note: Pins labelled with `n/c` indicate that there is no connection to the pin.
- Double-click a pin inside a schematic symbol to trace the net downward into the hierarchy.
- Double-click the pin outside a schematic symbol to trace the net up the hierarchy.

Note: Net expansion has a different result than expanding the hierarchical module using the Expand Inside/Expand Outside commands. Double-clicking a pin expands the hierarchy to follow the net, and does not display the full contents of the hierarchy.

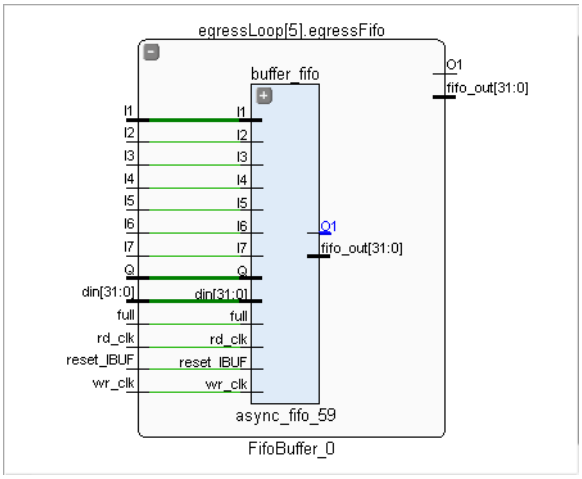


Figure 3-37: **Double-Click Schematic Pins**

You can expand buses to include all bits of the bus. Buses show as thick wires. Use the **Expand Cone** command from the popup menu to expand the cone of logic from a selected pin or cell, or between two selected cells. Expansion of logic can go beyond hierarchical boundaries. The window zooms to fit the expansion. The available **Expand Cone** commands are:

- **To Flops or I/Os**: Displays the entire cone of logic to the first flops or I/Os, or to any sequential element, such as block RAMs and FIFOs.
- **To Primitives**: Displays the entire cone of output logic to the first primitives.

  Note: Alternatively, you can double-click a pin or cell to use this command.
- **Between Selected Cells**: Displays the entire cone of logic between two selected cells.

Selecting Objects in the Schematic Window

To select objects in the Schematic window:

- Click an object in the Schematic window.
- Use the **Ctrl** key to select multiple objects.
- Click the **Select Area** local toolbar button, and draw a rectangle around multiple cells, ports, and nets.

When you select objects in the Schematic window:

- Objects are also selected in all other windows. Similarly, when you select objects in other windows, they are also selected in the Schematic window.
- The Properties window for the selected object opens or updates to display the object properties.

  For example, when a net is selected, the Connectivity view traverses the hierarchy to report all primitive cells connected to the net. This is different from the Pins view, which reports the pins of all cells connected to the net, reporting both primitive and hierarchical cells. Select a net that is connected to a hierarchical cell to see the difference between these views.

Setting Schematic Window Options

The Schematic Options define which properties to display on schematic symbols and pins as well as configure the colors to use when creating the Schematic window. To set Schematic window display options, click the **Schematic Options** toolbar button.

The Schematic Options are:

- Display Properties
- Colors

After you finish setting options, click the Close (**<<**) button in the upper right corner. The Vivado IDE stores your settings and reloads them each time the tool is launched.

TIP: *To restore the options to the default settings, click the Reset button in the upper right corner*  .

Display Properties

The options in the Display Properties view (Figure 3-38) enable or disable the display of the following features in the Schematic window:

- **Inst Equation**: Labels cells with truth table equations.
- **Slack for Scalar/Bus Pin**: Labels destination pins with slack values. Slack values do not display until after data is generated by Report Timing Summary or Report Timing. For more information, see the *Vivado Design Suite User Guide: Design Analysis and Closure Techniques* (UG906) [Ref 13].
- **Fanout for Scalar/Bus Pin**: Labels cell pins with fanout values.
- **Bus Value**: Labels bus pins with bus values.
- **Cell Name**: Labels cells with cell names.
- **Bundle Nets**: Shows buses as a bundled net.
- **Split into Multiple Pages**: Shows large schematics on multiple pages.

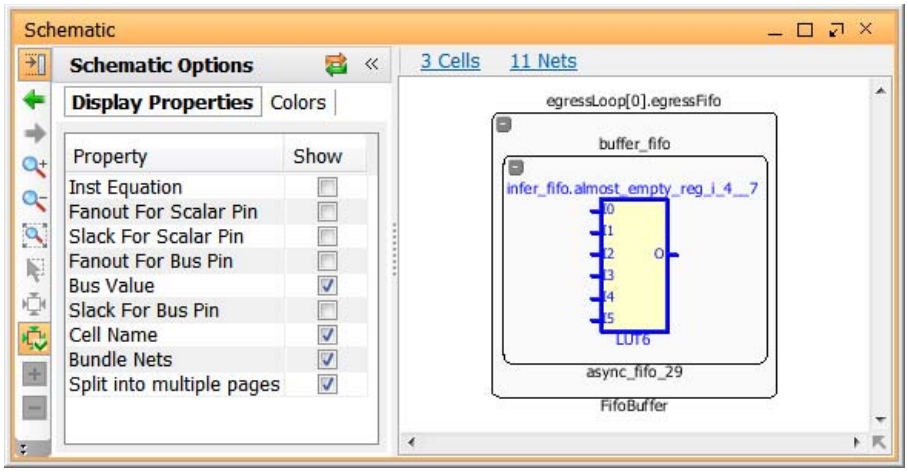


Figure 3-38: **Schematic Options—Display Properties**

Colors

The options in the Colors view (Figure 3-39) change the color of elements of the Schematic window:

- Click a color box to expose a drop-down menu, and select from a list of available colors.
- Select **More Colors** to display more colors to choose from.
- Enter a specific RGB value directly in the text field for the color.

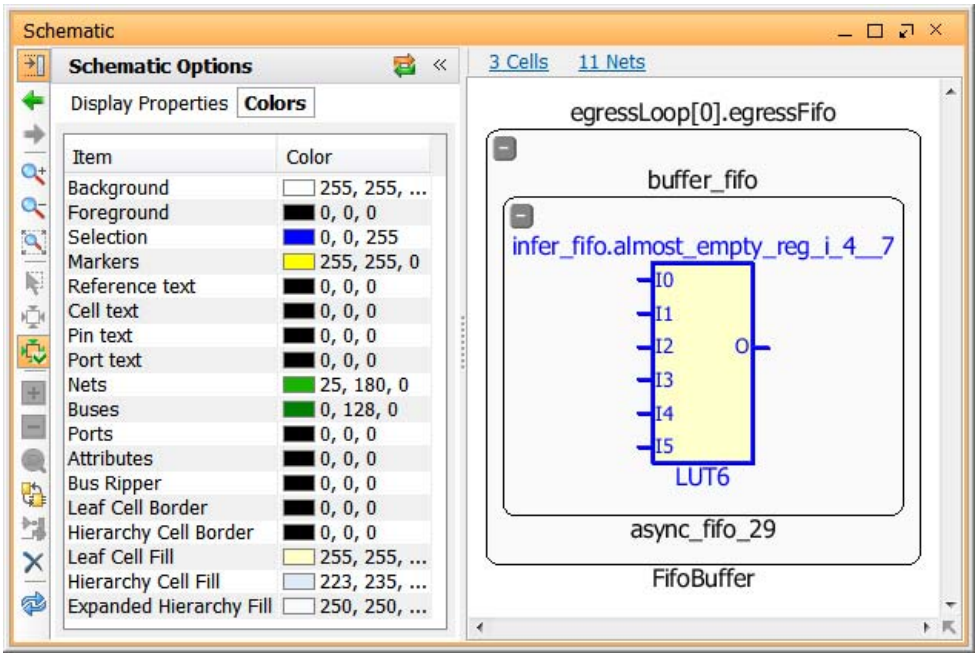


Figure 3-39: **Schematic Options—Colors**

Viewing Timing Path Logic in the Schematic Window

You can select Timing paths from the Timing Results window and use the **Schematic** command in the popup menu to display the full timing path in the Schematic window. All of the objects on the selected path or group of paths display with the logic hierarchy boundaries and the interconnect wires, as shown in Figure 3-40. For more information on setting the timing path logic, see the *Vivado Design Suite User Guide: Design Analysis and Closure Techniques* (UG906) [Ref 13].

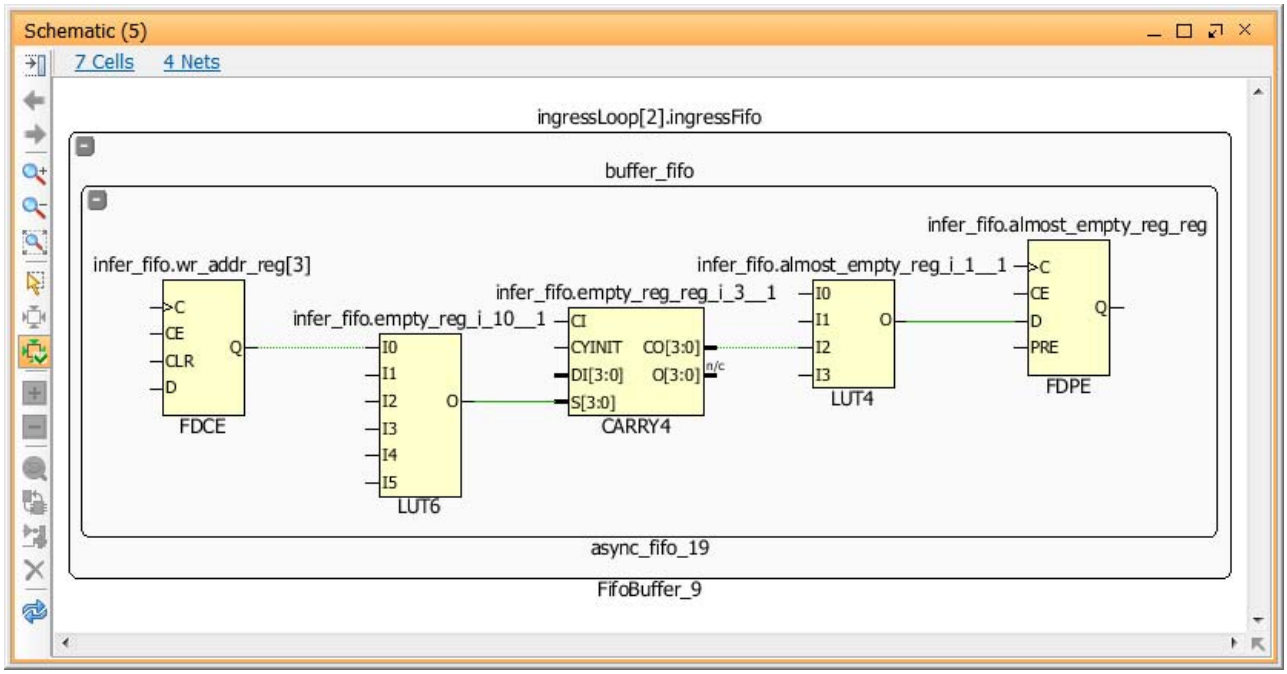


Figure 3-40: **Timing Path in Schematic Window**

Viewing Bundled Logic in the RTL Schematic Window

In the elaborated design, low-level logic connected to buses is represented as grouped logic to make the RTL schematic easier to view, as shown in Figure 3-41.

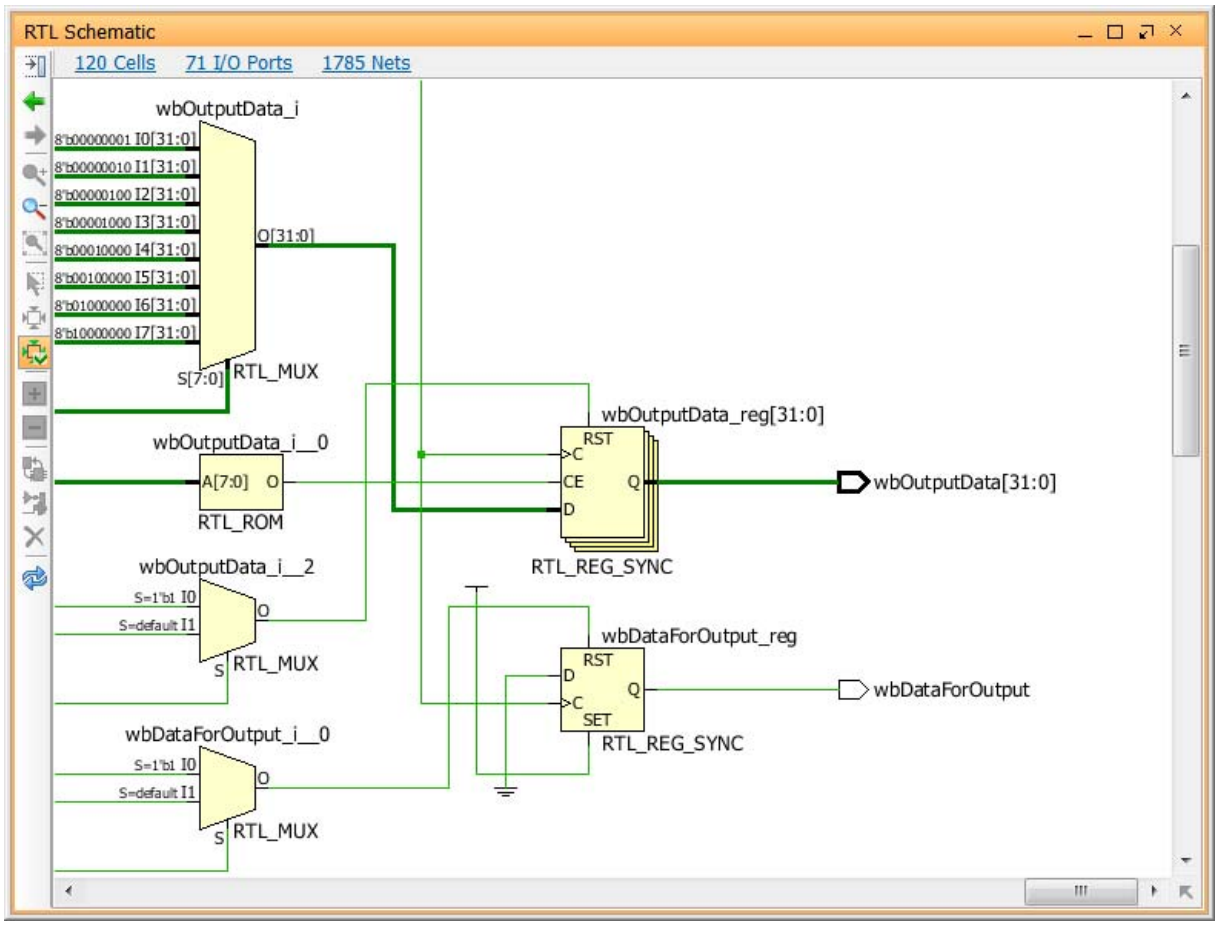


Figure 3-41: **RTL Schematic with Bundled Logic**

Using the Hierarchy Window

The Hierarchy window (Figure 3-42) displays a graphical representation of the logic hierarchy for the current design, based on the current top module. Viewing the design from top to bottom, you can identify the relationship between hierarchical modules, approximate module sizes, and module location within the design. To open the Hierarchy window, right-click in a window, such as the Netlist window, and select the **Show Hierarchy** command.

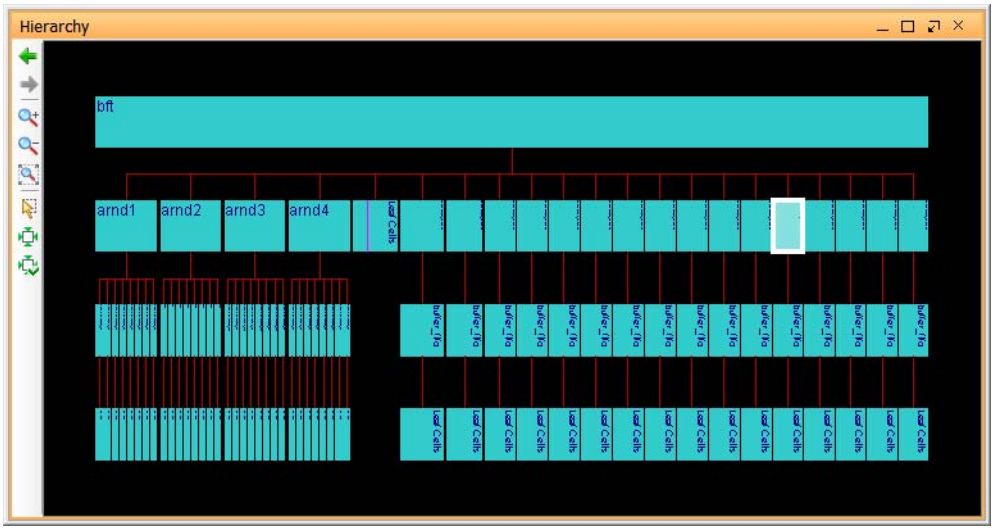


Figure 3-42: **Hierarchy Window**

The Hierarchy window is used primarily during design analysis and floorplanning, as described in the *Vivado Design Suite User Guide: Design Analysis and Closure Techniques* (UG906) [Ref 13]. The Hierarchy window allows you to see how a timing path traverses the logic hierarchy or gauge how big a module is before you floorplan the module. The widths of the blocks in the Hierarchy window are based on the relative FPGA resources consumed by that instance of hierarchy.

Each hierarchical instance displays in the Hierarchy window. Primitive logic is grouped into folders that are represented as sub-modules. For more information about primitive logic folders, see Using the Netlist Window.

TIP: *To select logic parent modules for Pblock assignment in the Hierarchy window, use the* ***Select Primitive Parents*** *popup menu command.*

Using the Timing Constraints Window

The Timing Constraints window (Figure 3-43) shows the timing constraints used for the loaded design. You can create new constraints, modify existing timing constraints, and run timing reports against the constraints. After the timing constraints are working as desired, you must save the changes to the original constraint set or create a new constraint set to preserve the constraints for the next implementation run. For more information, see the *Vivado Design Suite User Guide: Using Constraints* (UG903) [Ref 11]. To open the Timing Constraints window, select **Window > Timing Constraints**, or select **Edit Timing Constraints** in the Flow Navigator under Synthesized Design or Implemented Design.

IMPORTANT: *To ensure that the report tools recognize the constraint changes, you must press the **Apply** button in the Timing Constraints window to apply the changes.*

TIP: *Select **Tools** > **Timing** > **Constraints Wizard** on a synthesized design to create a top-level XDC file based on design methodologies recommended by Xilinx. The wizard guides you through specifying clocks, setting up input and output constraints, and properly constraining cross-clock domain clock groups.*

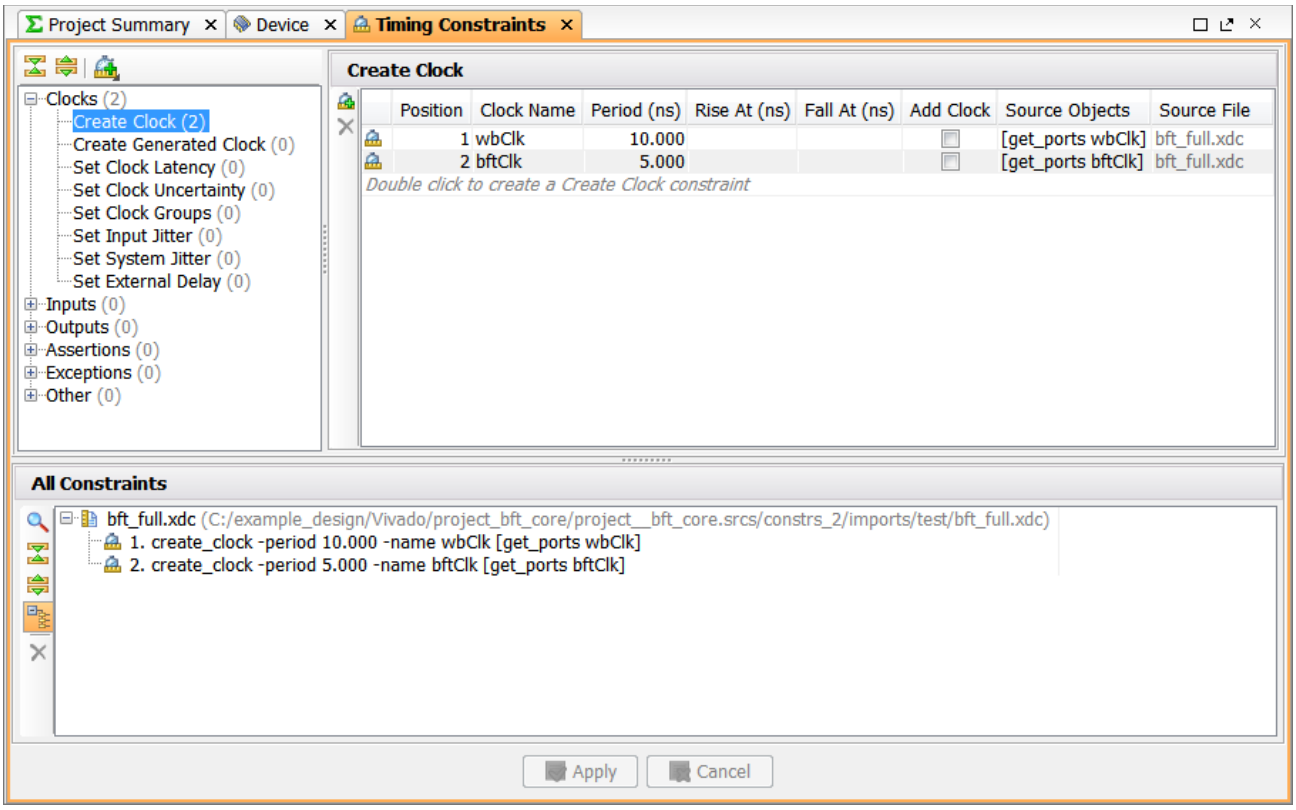


Figure 3-43: **Timing Constraints Window**

Using the Timing Constraints Window Toolbar Commands

In the Timing Constraints window, you can:

- Expand or collapse the constraints by toggling the tree widgets in the window or by clicking the **Expand all** or **Collapse all** toolbar buttons.
- Use the **Show Search** toolbar button to search for and display specific constraints.

  Note: You can also access this command through the **Alt+/** keyboard shortcut.
- Use the **Remove** toolbar button to remove a constraint.

- Use the **Group by Source** toolbar button to group constraints based on the source file from which they originate.
- Use the **Create Constraint** toolbar button to create a new constraint of the selected type.

  Note: You can also double-click a constraint name in the constraint tree to create a new constraint of the selected type.

Using the Waveform Window

For information on using the Waveform window, see the *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 9] and *Vivado Design Suite User Guide: Programming and Debugging* (UG908) [Ref 14].

Using the Tcl Console

The Tcl Console (Figure 3-44) displays:

- Messages from previously executed Tcl commands.

  Note: The Vivado IDE also writes these messages to the `vivado.log` file.
- Command errors, warnings, and successful completion.
- Status of design loads and reading constraints.

To open the Tcl Console, select **Window > Tcl Console**.

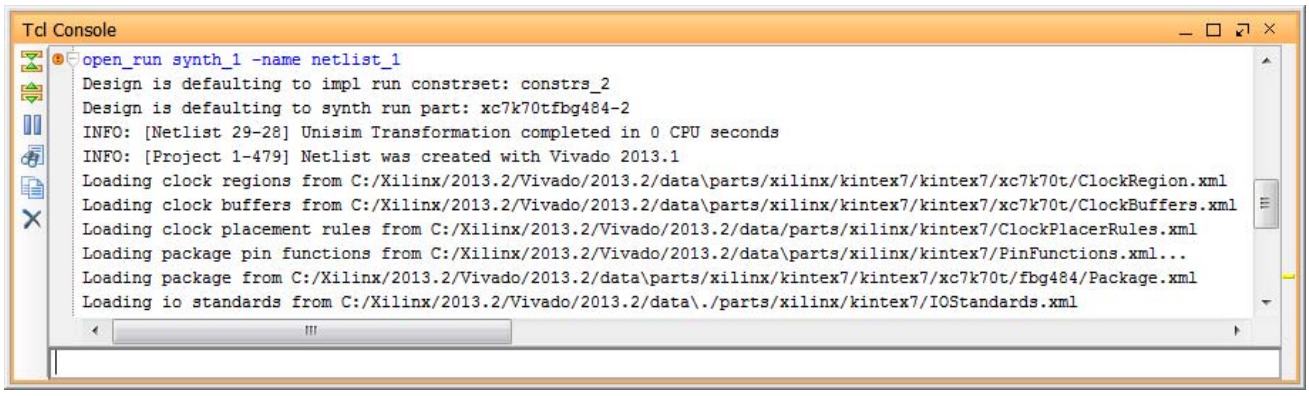


Figure 3-44: **Tcl Console**

Using the Tcl Console Toolbar and Popup Menu Commands

In the Tcl Console window, you can:

- Expand or collapse the messages reported by each Tcl command in the Tcl Console by toggling the tree widgets or by clicking the **Expand all** or **Collapse all** toolbar buttons.
- Use the **Pause** toolbar button to scroll in the window or read reports as commands are running.
- Find text strings in the displayed messages using the **Show Find** toolbar button.
- Copy Tcl commands using the **Copy** toolbar button.
- Clear the Tcl Console using the **Clear all output** toolbar button or **Clear All Output** popup menu command.

Locating Warnings and Errors in the Tcl Console

Warnings and Errors display with a yellow or red indicators on the right side of the Tcl Console as shown in Figure 3-45. You can use these indicator as follows:

- Hold the mouse over an indicator to display the messages in a tooltip.
- Click an indicator to scroll to the associated message in the Tcl Console.

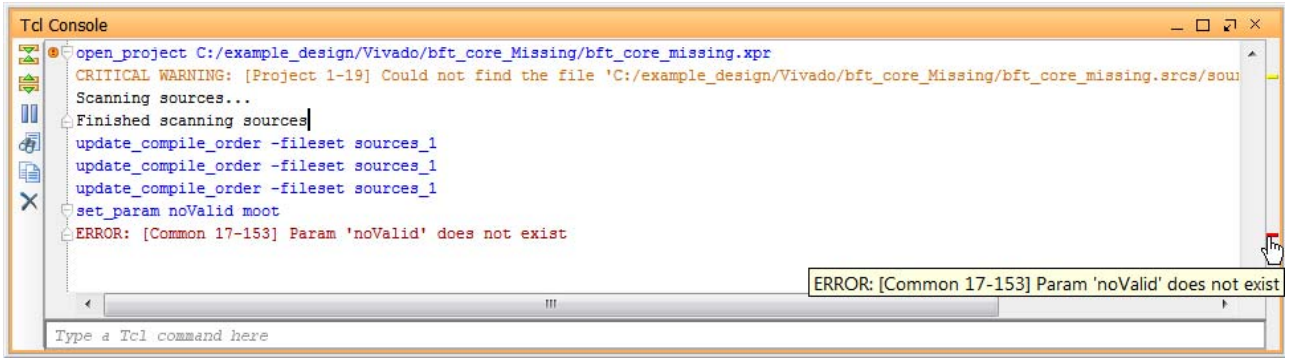


Figure 3-45: **Warnings and Errors in the Tcl Console**

Entering Tcl Commands

To enter Tcl commands, click in the command line entry box at the bottom of the Tcl Console, and type the commands.

As you type commands, the Tcl Console auto-complete feature attempts to complete the name of the command or command parameters. For example, Figure 3-46 shows a list of commands that match the entry: `create_`. From the auto-complete list, you can:

- Click a command to select it.
- Use the up or down arrow key to scroll to the command, and press **Enter** to select it.
- Continue to type until the auto-complete feature narrows the command to one choice, and press the **Tab** key to select it.

After you select a command, the Tcl Console attempts to auto-complete any arguments of the command. You can select from the auto-complete list as described above.

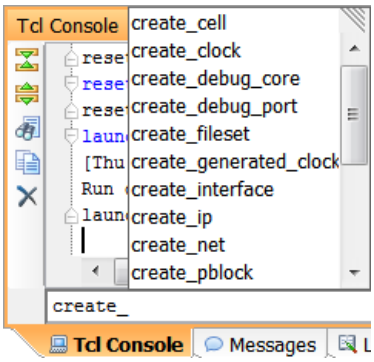


Figure 3-46: **Auto-Completion of Tcl Commands**

The output of Tcl commands is optimized for processing, not viewing. To improve the readability of the single line of output returned by a Tcl command, use the `join` command and a newline (`\n`) as shown:

```
join <command> \n
```

For example:

```
join [get_cells -hier *buffer*] \n
```

Viewing the Tcl Command History

When you perform an action in the Vivado IDE, such as using a menu command or performing drag and drop, the tool writes a Tcl command equivalent to the Tcl Console.

To display the command history in the Tcl Console, type the following in the command line entry box at the bottom of the Tcl Console:

```
history
```

TIP: *In the command line entry box, you can press the arrow keys to scroll through the command history one command at a time.*

In addition, the Vivado IDE writes the Tcl commands to a journal file (`vivado.jou`) and a log file (`vivado.log`). The `vivado.jou` file contains just the commands, and the `vivado.log` file contains both commands and any returned messages. When the Vivado IDE is launched, backup versions of the journal file (`vivado_<id>.backup.jou`) and log file (`vivado_<id>.backup.log`) are written to save the details of the previous run. The `<id>` is a unique identifier that enables the tools to create and store multiple backup versions of the log and journal files. For more information, see Output Files in Appendix A.

Using Tcl Help

In the command line entry box at the bottom of the Tcl Console, type:

```
help
```

To get detailed information about a command, type:

```
help <command_name>
```

or

```
<command_name> -help
```

For example:

```
help add_files
```

or

```
add_files -help
```

The Tcl Console displays the list of available commands or command options based on the command you enter.

TIP: *To make it easier to read the command help, double-click the Tcl Console tab or press* ***Alt -*** *to maximize it.*

For explicit command syntax, perform the command once, then view the `vivado.jou` file in the invocation directory. For more information on creating Tcl scripts, see the*Vivado Design Suite User Guide: Using Tcl Scripting* (UG894) [Ref 4]. For a complete list of Tcl commands, refer to the *Vivado Design Suite Tcl Command Reference Guide* (UG835) [Ref 5].

IMPORTANT: *The `vivado.jou` file is a good starting point for creating a Tcl script. However, it is not intended to be used as a script itself.*

Using the Messages Window

The Messages window (Figure 3-47) displays design and reports messages. Messages are grouped under different headings to enable quick location of messages from different tools or processes. Messages display with a link to the relevant source file. You can click the link to open the RTL source file in the Text Editor with the appropriate line of code highlighted. To open the Messages window, select **Windows > Messages**.

VIDEO: *For an overview of the Messages window, including information on reviewing critical messages, cross probing design objects, adjusting message severity, and suppressing messages, see the* Vivado Design Suite QuickTake Video: Understanding Messaging *or* Vivado Design Suite QuickTake Video: Messages, Reports, and Log Files Overview.

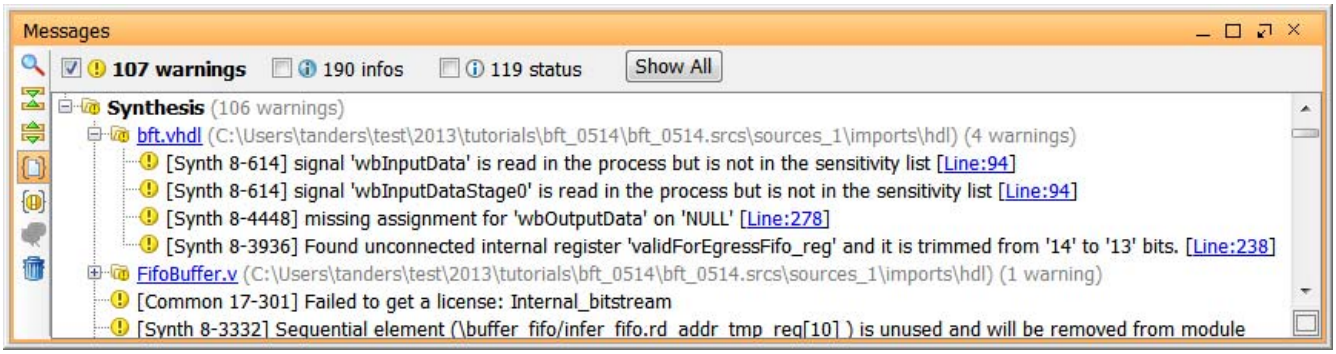


Figure 3-47: **Messages Window**

Using the Messages Window Toolbar and Popup Menu Commands

In the Messages window, you can:

- Use the checkboxes in the banner of the Messages window to hide or display the Errors, Critical Warnings, Warnings, and Info messages.

TIP: *To see only one message type, double-click the message type in the banner of the Messages window. For example, double-click **errors** to display only error messages. To get the message count for critical warnings, use the following Tcl command:* `get_msg_config -count -severity {CRITICAL WARNING}`.

- Use the **Show Search** toolbar button to search for and display specific messages.
- Expand or collapse the messages by toggling the tree widgets in the window or by clicking the **Expand all** or **Collapse all** toolbar buttons.

  Note: You can also access this command through the **Alt+/** keyboard shortcut.
- Use the **Group Messages by File** toolbar button to group messages by file, as shown in Figure 3-47.
- Use the **Group Messages by ID** toolbar button to sort the messages by message ID.
- Use the **Show Messages** toolbar button to adjust the filtering of messages.

  Note: This toolbar button is only enabled if you filtered messages from the view. When this button is enabled, you can click the button and select **Show Unsuppressed Messages**, **Show Suppressed Messages**, and **Manage Suppression**.
- Use the **Discard Old Messages** toolbar button to remove project load and analysis messages, messages output by Tcl commands entered in the Tcl Console, and messages output by scripts.

  Note: You *cannot* use this command for messages output from design runs.
- Use the **Wrap Messages** popup menu command to enable or disable line wrapping.

  By default, messages written to the Messages window are line-wrapped to allow the display of the whole message within the message window. When line wrap is enabled, the lines adjust to fit the width of the Messages window. If the Messages window is resized, line wrapping is adjusted accordingly. If line wrap is disabled, the messages display in a single line, regardless of the width of the Messages window.
- Use the **Search for Answer Record** popup menu to open the Xilinx Answers Search page populated with a search for Answer Records related to the reported message.

Viewing Message Details

When additional details are available for a message, a unique message ID appears as a link in the Messages window. Click this link to open a window (Figure 3-48) that contains a description and possible resolutions.

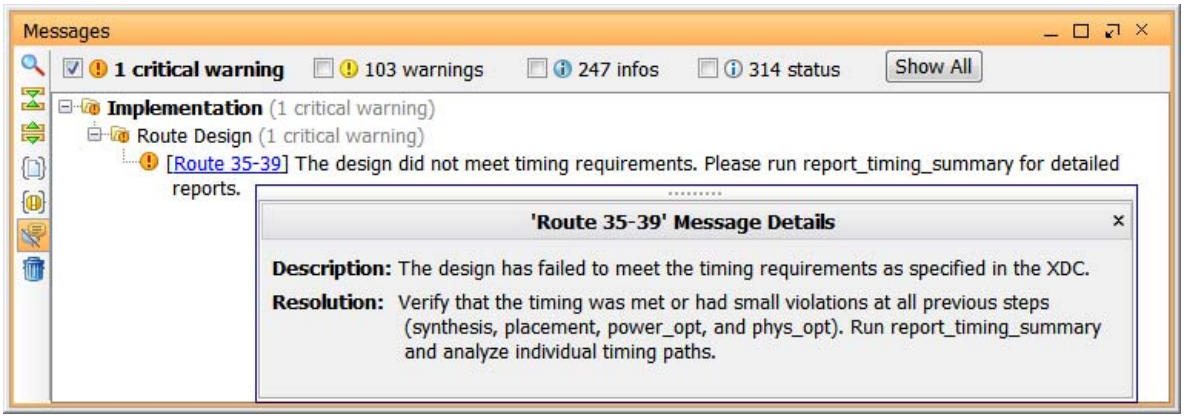


Figure 3-48: **Message Details**

Cross Probing Message Objects

When design objects, such as cells, nets or pins, are displayed as a link (Figure 3-49), you can click the link to cross probe to different windows in the workspace.

TIP: *Use the **Auto Fit Selection** toolbar button to zoom to the selected object.*

Figure 3-49: **Cross Probing Message Objects**

Suppressing Messages

You can suppress messages from appearing in the Vivado IDE Messages window as follows:

- To suppress a specific message, right-click the message, and select **Suppress this Message**.
- To suppress all messages with a specific message ID, right-click the message, and select **Suppress Messages with this ID**.
- To suppress a message that contains a specific text string or collection of strings, right-click in the Messages window, and select **Manage Suppression**. In the Manage Suppression dialog box (Figure 3-50), select the **Suppression Rules** tab, and click **Add** to enter the string. Use the **+** button to include additional strings, and click **OK**.

  Note: Messages that are filtered from the Messages window still exist in the log files.

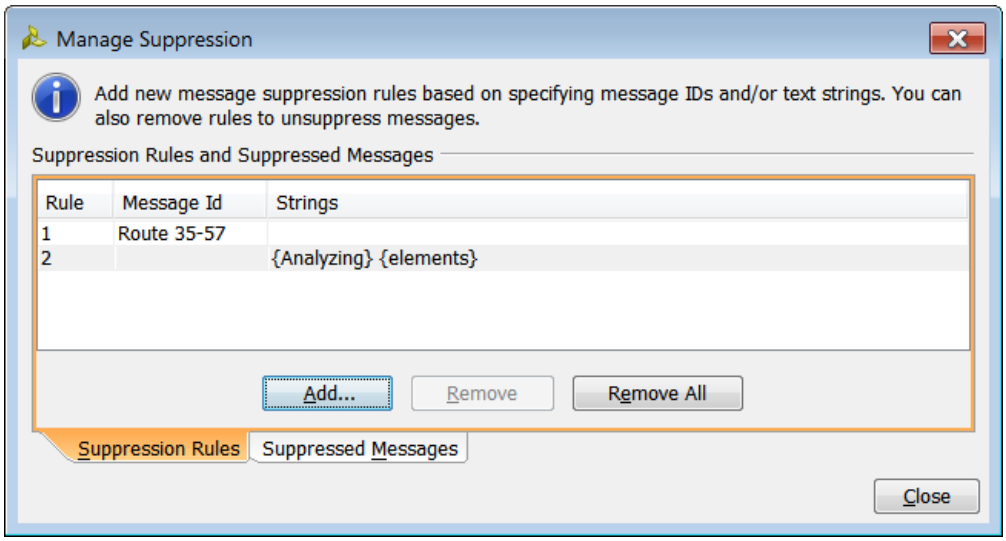


Figure 3-50: **Manage Suppression Dialog Box—Suppression Rules Tab**

Unsuppressing Messages

To unsuppress messages, right-click in the Messages window, and select **Manage Suppression**. In the Manage Suppression dialog box (Figure 3-51), do any of the following:

- For string based rules, click the **Suppression Rules** tab. Click the **Remove** button to remove the selected rule. Click the **Remove All** button to remove all rules.
- For messages suppressed by ID or by the complete message (ID plus message string), select the **Suppressed Messages** tab. Click the **Unsuppress** button to unsuppress a selected message or ID. Click the **Unsuppress All** button to unsuppress all messages.

Note: Alternatively, you can click the **Show Messages** toolbar button, and select **Manage Suppression** to open the Manage Suppression dialog box.

TIP: *If you want to temporarily display suppressed messages, click the* ***Show Messages*** *toolbar button , and select* ***Show Suppressed Messages****. In the Messages window, the message icon has a backslash (\) to indicate that the message is suppressed but displayed.*

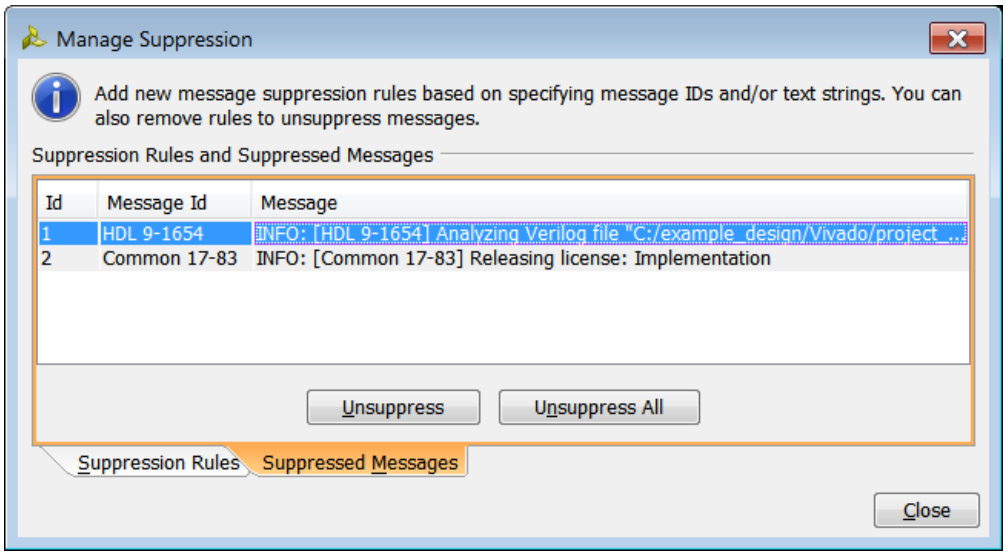


Figure 3-51: **Manage Suppression Dialog Box—Suppressed Messages Tab**

Changing Message Severity

You can modify the severity for any type of message. You can:

- Promote all messages to errors except for status messages.
- Demote warnings and critical warnings.

CAUTION! *You cannot demote error messages. Use caution when demoting critical warnings, because these messages flag problems that might result in errors later in the design flow.*

To change the severity of a message:

1. Right-click the message, and select **Message Severity > Set Message Severity** from the popup menu.
2. In the Set Message Severity dialog box (Figure 3-52), set the severity.

Note: To reset the message to the default severity, right-click the message, and select **Message Severity > Unset Message Severity** from the popup menu.

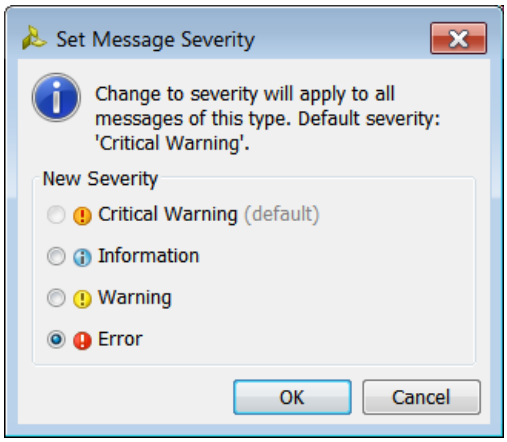


Figure 3-52: **Set Message Severity Dialog Box**

Following are the icons for modified messages:

- Modified Info
- Modified Warning
- Modified Critical Warning
- Modified Error

Following is an example Tcl command that upgrades message ID `Place 30-12` to a Critical Warning:

```
set_msg_config -id {Place 30-12} -new_severity {CRITICAL WARNING}
```

Using the Log Window

The Log window (Figure 3-53) displays the active output status of commands that compile the design such as synthesis, implementation and simulation. The output displays in a continuous scrollable format and is overwritten when new commands are run. This window opens automatically when you launch a command on an active run. If the Log window is hidden, to open it, select **Window > Log**.

TIP: *You can use the **Pause output** toolbar button to scroll back or read reports while commands are running.*

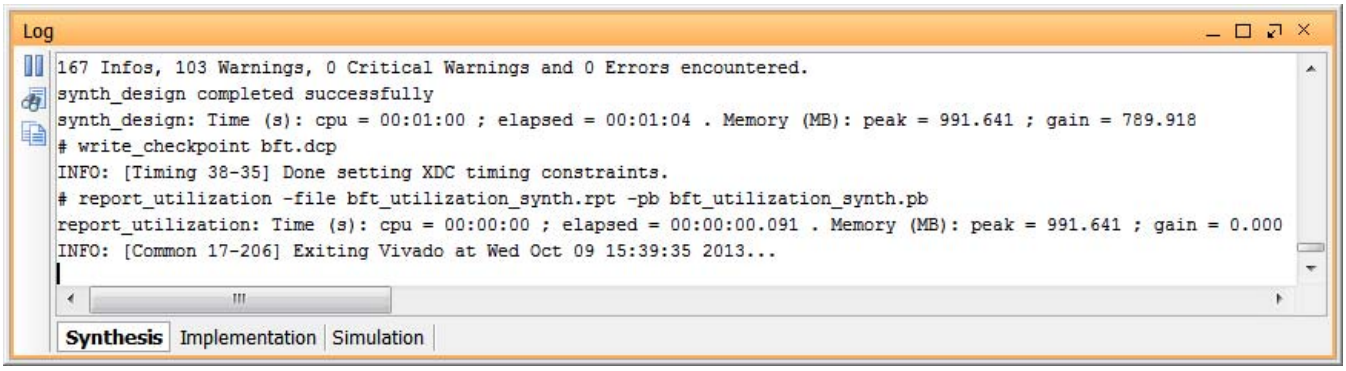


Figure 3-53: **Log Window**

Using the Reports Window

The Reports window (Figure 3-54) displays reports for the active run and updates as various steps complete. Reports are grouped under headings named after the different steps to enable quick location of information. Double-click a report to open the file in the Text Editor. To open the Reports window, select **Windows > Reports**.

VIDEO: *For an overview of the Reports window and various reports, see the* Vivado Design Suite QuickTake Video: Messages, Reports, and Log Files Overview.

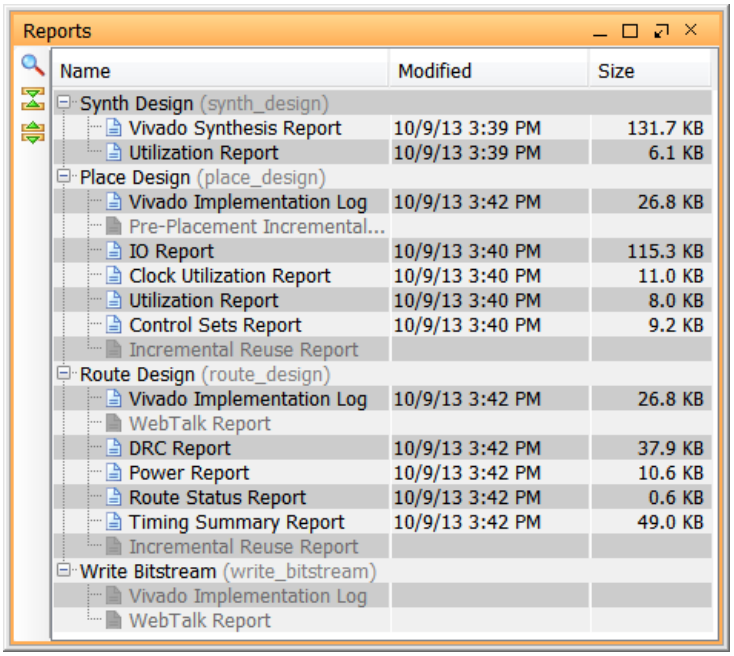


Figure 3-54: **Reports Window**

Using the Reports Window Toolbar

In the Reports window, you can:

- Use the **Show Search** toolbar button  to search for and display specific messages.

  Note: You can also access this command through the **Alt+/** keyboard shortcut.

- Expand or collapse the reports by toggling the tree widgets in the window or by clicking the **Expand all**  or **Collapse all**  toolbar buttons.

Generating Additional Reports

In addition to the reports generated during synthesis and implementation, you can generate custom reports on demand at different stages in the design flow. From the Flow Navigator or using the **Tools > Reports** menu command, you can generate the following reports:

- Report DRC
- Report Noise
- Report Utilization
- Report Power
- Report Power Optimization
- Report High Fanout Nets
- Report Clock Networks
- Report IP Status
- Timing
  - Check Timing
  - Report Timing
  - Report Timing Summary
  - Report Pulse Width
  - Create Slack Histogram
  - Report Clock Interaction
  - Report Datasheet

TIP: *If multiple timing reports are open, you can tile the reports horizontally or vertically. Right-click a report tab, and select* ***New Horizontal Group*** *or* ***New Vertical Group****.*

For more information on these reports, see the following documents:

- *Vivado Design Suite User Guide: Power Analysis and Optimization* (UG907) [Ref 18]
- *Vivado Design Suite User Guide: I/O and Clock Planning* (UG899) [Ref 16]
- *Vivado Design Suite User Guide: Design Analysis and Closure Techniques* (UG906) [Ref 13]
- *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 7]

Using the Design Runs Window

You can use the Design Runs window (Figure 3-55) to view, configure, launch, and analyze synthesis and implementation runs. The Design Runs window offers commands to manage, launch, and reset runs for synthesis and implementation. To open the Design Runs window, select **Window > Design Runs**.

Figure 3-55: **Design Runs Window**

Understanding Run Status

As runs are created, or launched, the run status updates in the Design Runs window. The window displays the status and results of the design runs defined, and it also provides commands to modify, launch, and manage the design runs.

The Design Runs window icons indicates the run state as follows:

- Reset and ready to run
- Currently running
- Completed runs
- Completed runs in which design files have changed
- Completed runs that failed to meet timing
- Runs with errors

Run information updates as the runs proceed. You can close the Vivado IDE without affecting in-progress runs. When you re-open a project, the Vivado IDE updates the run status to reflect the latest status, which displays in the Design Runs table. The columns used for tracking information are:

- **Name**: Displays run name.
- **Part**: Indicates the target part selected for the run.
- **Constraint**: Displays the constraint set used for the run.
- **Strategy**: Displays the strategy assigned to the run. Strategies with an asterisk (*) indicate that some command options for the strategy have been overridden.
- **Status**: Indicates run status as not started, running, complete, or error.
- **Progress**: Displays the percentage complete (0 to 100%).
- **Start**: Reports the start time for the run.
- **Elapsed**: Reports the elapsed time for the run.
- **Timing Summary Data**
  - **WNS**: Displays worst negative slack.
  - **TNS**: Displays total negative slack.
  - **WHS**: Displays worst hold slack.
  - **THS**: Displays total hold slack.
  - **TPWS**: Displays total pulse width negative slack.
- **Failed Routes**: Displays the number of nets that failed to route, are partially routed, or have conflicts.
- **LUT**: Displays the LUT utilization percentage.
- **FF**: Displays the flip-flop utilization percentage.
- **BRAM**: Displays the BRAM utilization percentage.
- **DSP**: Displays the DSP utilization percentage.
- **Description**: Displays the description associated with the run. This description is set initially to a strategy description when that strategy is applied to the run. However, you can modify the description later in the Run Properties window.

Note: The table is updated dynamically as the run commands progress. Runs that are launched outside of the Vivado IDE using generated scripts cause the table to update upon invoking the software.

Design Runs Window Commands

The Design Runs window commands are:

- **Run Properties**: Displays the Run Properties window. See Using the Run Properties Window. 
- **Delete**: Deletes the selected, non-active, runs and removes the associated run data from disk. You are prompted to confirm before the runs are deleted. 

  Note: You cannot delete the active runs.
- **Make Active**: Sets the selected run as the active run. The active run launches automatically when the **Run Synthesis** or **Run Implementation** command is used. The results for the active run display in the Messages, Compilation, Reports, and Project Summary windows.
- **Change Run Settings**: Changes the strategy and command line options for the selected synthesis or implementation run. For more information, see the *Vivado Design Suite User Guide: Synthesis* (UG901) [Ref 10] or *Vivado Design Suite User Guide: Implementation* (UG904) [Ref 12].
- **Set Incremental Compile**: Specifies a placed and routed design checkpoint to use as a reference for the next implementation run. For more information, see the *Vivado Design Suite User Guide: Implementation* (UG904) [Ref 12].

CAUTION! *If you are using a design checkpoint from the active run, you must copy the design checkpoint outside of the run directory prior to using it for incremental compile. When you run incremental compile, the run is reset and all contents are deleted, including the design checkpoint.*

- **Save as Strategy**: Saves the current strategy and command options to a new strategy for future use and modification.
- **Open Run**: Opens the design for the selected run. 
- **Launch Runs**: Invokes the Launch Selected Runs dialog box to launch the selected runs. 
- **Reset Runs**: Invokes the Reset Runs dialog box to remove previous run results and to set the run status back to **Not Started** for the selected runs. 
- **Launch Next Step: <Step>**: Launches the next step of the selected run. For implementation runs, the available steps are `opt_design`, `place_design`, `route_design`, `write_bitstream`. Synthesis only has one step: `synth_design`. 
- **Reset to Previous Step: <Step>**: Resets the selected run to the prior step. This allows you to step backward through a run, to make any needed changes, then step forward to complete the run. 
- **Generate Bitstream**: Invokes the `write_bitstream` step. This command is available for completed implementation runs only. For more information, see the *Vivado Design*

Suite User Guide: Programming and Debugging (UG908) [Ref 14].

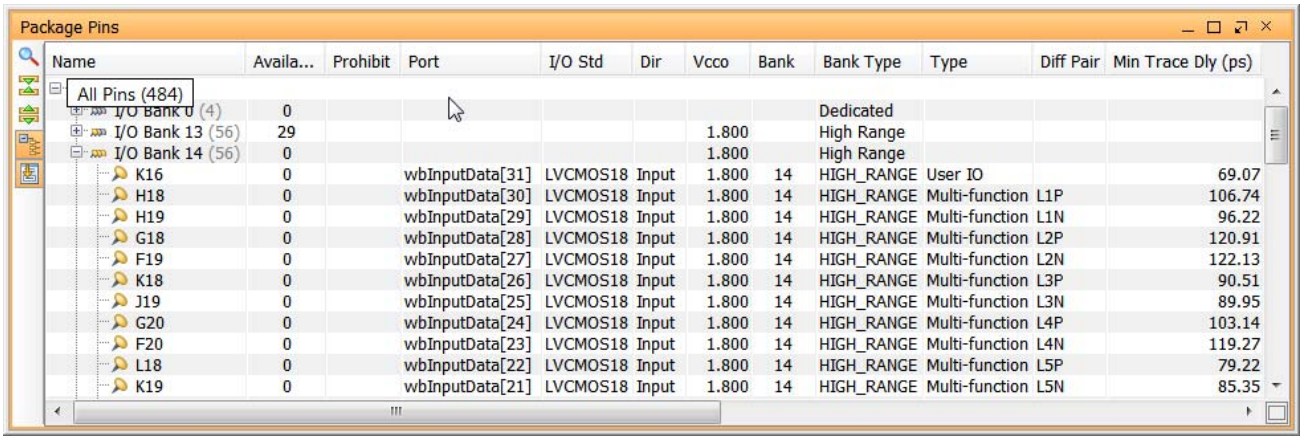
- **Display Run Log**: Displays the Log tab of the Run Properties window.
- **Display Run Reports**: Displays the Reports tab of the Run Properties window.
- **Display Run Messages**: Displays the Messages tab of the Run Properties window.

TIP: *The Display commands are useful for displaying data for out-of-context runs, which are not displayed in the Messages, Reports, or Log windows.*

- **Copy Run**: Creates a new run using the same strategy as the selected run.
- **Create New Runs**: Invokes the Create New Runs wizard to create and configure new synthesis or implementation runs. For more information, see the *Vivado Design Suite User Guide: Synthesis* (UG901) [Ref 10] or *Vivado Design Suite User Guide: Implementation* (UG904) [Ref 12].
- **Open Run Directory**: Opens a file browser in the selected run directory on disk.
- **Export to Spreadsheet**: Export information in the Design Runs window into a spreadsheet file.

Using the Package Pins Window

The Package Pins window displays I/O-related package information. You can sort and filter the table to analyze the I/O pins and I/O ports information. To open the Package Pins window, select **Window > Package Pins**.



Name	Availa...	Prohibit	Port	I/O Std	Dir	Vcco	Bank	Bank Type	Type	Diff Pair	Min Trace Dly (ps)
I/O Bank 0 (4)	0							Dedicated			
I/O Bank 13 (56)	29					1.800		High Range			
I/O Bank 14 (56)	0					1.800		High Range			
K16	0		wbInputData[31]	LVCMOS18	Input	1.800	14	HIGH_RANGE	User IO		69.07
H18	0		wbInputData[30]	LVCMOS18	Input	1.800	14	HIGH_RANGE	Multi-function	L1P	106.74
H19	0		wbInputData[29]	LVCMOS18	Input	1.800	14	HIGH_RANGE	Multi-function	L1N	96.22
G18	0		wbInputData[28]	LVCMOS18	Input	1.800	14	HIGH_RANGE	Multi-function	L2P	120.91
F19	0		wbInputData[27]	LVCMOS18	Input	1.800	14	HIGH_RANGE	Multi-function	L2N	122.13
K18	0		wbInputData[26]	LVCMOS18	Input	1.800	14	HIGH_RANGE	Multi-function	L3P	90.51
J19	0		wbInputData[25]	LVCMOS18	Input	1.800	14	HIGH_RANGE	Multi-function	L3N	89.95
G20	0		wbInputData[24]	LVCMOS18	Input	1.800	14	HIGH_RANGE	Multi-function	L4P	103.14
F20	0		wbInputData[23]	LVCMOS18	Input	1.800	14	HIGH_RANGE	Multi-function	L4N	119.27
L18	0		wbInputData[22]	LVCMOS18	Input	1.800	14	HIGH_RANGE	Multi-function	L5P	79.22
K19	0		wbInputData[21]	LVCMOS18	Input	1.800	14	HIGH_RANGE	Multi-function	L5N	85.35

Figure 3-56: **Package Pins Window**

Device pin information such as the following is listed for each package pin:

- I/O Bank number
- Bank Type

  Note: The Bank Type column identifies High Performance and High Range banks.
- Differential pair partners
- Site Types
- Min/Max package delay

  Note: The unit of measurement for the Min/Max package trace delay in the Package Pins window is in picoseconds (ps).

Table values appear as follows:

- Black for default values
- Black with an asterisk (*) for non-default values
- Red for illegal values

To sort the information in the Package Pins window:

- To sort, click a column header. Click again to reverse the sort order.
- To sort by a second column, press **Ctrl** and click another column. You can add as many sort criteria as necessary to refine the list order.

Note: For more information on sorting the information in the Package Pins window, see Using Data Table Windows.

TIP: *In the Package Pins window, you can directly edit cells with editable values. Either enter text, or select it from a drop-down menu.*

Package Pins Window Commands

The Package Pins window commands are:

- **Search**: Searches the Package Pins window for ports by name, or by keywords or values within the various pin properties.
- **Collapse All**: Displays I/O Banks by name, and does not display individual pins of the bank.
- **Expand All**: Shows all pins of an I/O Bank expanded.
- **Group by I/O Bank**: Groups the pins by I/O Bank, or lists them alphabetically by name.
- **Automatically scroll to selected objects**: Scrolls the Package Pins window to display objects selected in other windows like the Netlist or Device windows.
- **Fix Ports**: Constrains the selected, placed ports to their current locations, or if no ports are selected, constrains all placed ports. Upon completion, a dialog box appears with summary information.

  Note: This operation is only enabled for *placed* I/O ports. The resulting Tcl command is:
  `set_property is_loc_fixed true [get_ports [list <list of ports>]]`

Using the I/O Ports Window

The I/O Ports window (Figure 3-57) enables you to create, configure, and place I/O ports onto I/O sites in either the Package window or Device window. The I/O Ports window shows the I/O signal ports defined in the design. To open the I/O Ports window, select **Window > I/O Ports**.

Creating RTL source or netlist projects populates the I/O Ports window with the I/O ports defined in the design source files. In an I/O Planning project, you can import a port list from a CSV or XDC file and create ports manually for the project. For more information, see the *Vivado Design Suite User Guide: I/O and Clock Planning* (UG899) [Ref 16].

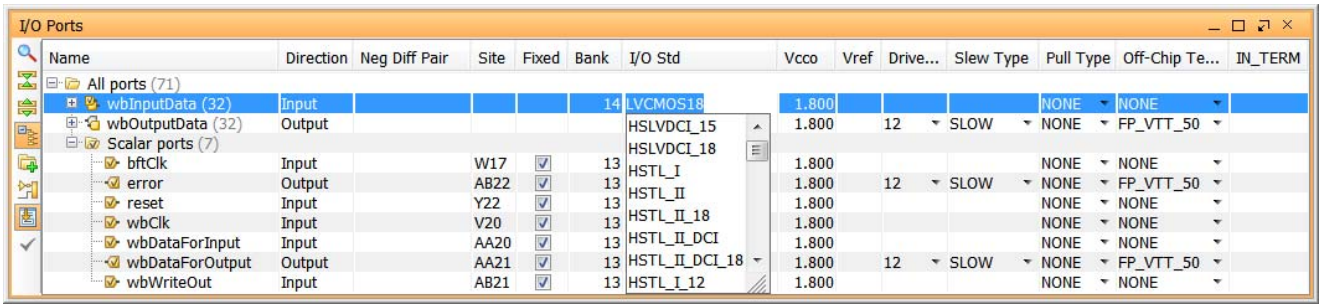


Figure 3-57: **I/O Ports Window**

The I/O Ports window lists the following for each I/O port and sorts the I/O ports based on column values:

- Port signal names
- Direction
- Board part pin (when using board part only)
- Board part interface (when using board part only)

  Note: For more information on using the board flow, see *Vivado Design Suite User Guide: System-Level Design Entry* (UG895) [Ref 15].
- Package pin
- I/O bank
- I/O Standard
- Drive strength
- Diff pair partner
- Slew type
- Voltage requirements
- Other signal information

The table in the I/O Ports window includes the following information:

- Buses are in expandable folders that you can select as one object for analysis, configuration, and assignment.
- Port Interfaces are in expandable folders that can contain buses and individual ports that you defined.
- Cells with editable values allow you to enter text or select text from drop-down menus.

Table values appear as follows:

- Blank for default values
- An asterisk (*) for non-default values
- Red for illegal or undefined values

Note: For more information on working with table views, see Using Data Table Windows.

I/O Ports Window Commands

The I/O Ports window commands are:

- **Create I/O Port**: Manually defines I/O Ports in an I/O Planning project. 
- **Search**: Searches the I/O Ports window for ports by name, or by keywords or values within the various port properties. 
- **Collapse All**: Displays buses by name, and does not display individual bits of the bus. 
- **Expand All**: Shows all pins of a bus expanded. 
- **Group by Interface and Bus**: Displays the ports by interface, or alphabetically by name. 
- **Create I/O Port Interface**: Defines a new port interface to group ports. You can select and place port interfaces as one object within the I/O Planning environment. 
- **Schematic**: Opens a Schematic window for selected I/O ports. 
- **Automatically scroll to selected objects**: Scrolls the I/O Ports window to display objects selected in other windows like the Netlist or Device windows. 
- **Fix Ports**: Constrains the selected, placed ports to their current locations, or if no ports are selected, constrains all placed ports. Upon completion, a dialog box appears with summary information. 

You can select ports and interfaces from the I/O Ports window and assign them to Package pins or Device resources, using the I/O Planning window layout. Using the popup menu of the I/O Ports window, you can:

- **Place I/O Ports in an I/O Bank**: Assigns the currently selected ports onto pins on the specified I/O Bank.
- **Place I/O Ports in Area**: Assigns the currently selected ports onto pins in the specified area.
- **Place I/O Ports Sequentially**: Assigns the currently selected ports individually onto pins.
- **Configure I/O Ports**: Assigns various properties of the selected I/O ports.
- **Reset Invalid Port Properties**: Resets any invalid properties on the specified port to the default value.
- **Reset Port Properties**: Resets all properties on the specified port to the default values.
- **Set Direction**: Specifies the direction of a port only in an I/O Planning project.
- **Make Diff Pair**: Defines two ports as a differential pair in an I/O Planning project.
- **Split Diff Pair**: Removes the differential pair association from the selected port in an I/O Planning project.

- **Auto-place I/O Ports**: Places I/O Ports using the Autoplace I/O ports wizard.
- **Fix/Unfix:** Fixes or Unfixes the selected placed I/O ports.
- **Unplace**: Unplaces the selected I/O ports.
- **Swap Locations**: Swaps the sites for two select ports.
- **Export I/O Ports**: Writes the contents of the I/O Ports window to a CSV, XDC, Verilog, or VHDL file.
- **Export to Spreadsheet**: Writes the contents of the I/O Ports window to a Microsoft® Excel spreadsheet.

Configuring the Environment

Overview

You can configure the look and feel of the Vivado® IDE along with many of the default actions. For example, you can change the default display colors to emphasize sites and properties of interest to you, adjust default settings for file paths and properties, assign custom keyboard shortcuts for frequently used commands, and create custom flow strategies. These settings are controlled by options available from the **Tools** and **Layout** menus. This chapter covers how to set these options in detail.

Specifying General Options

You can use the General Options (Figure 4-1) to configure many of the default behaviors in the Vivado IDE. To open the dialog box, select **Tools > Options**. On the left side of the dialog box, click the **General** category.

TIP: *When entering or modifying data in a text box, if a value is used and editable, the text is black and the background is white. If a value is used but not editable, the text is black and the background is gray. If a value is unused or not applicable, the text is gray, including the label that precedes or follows it.*

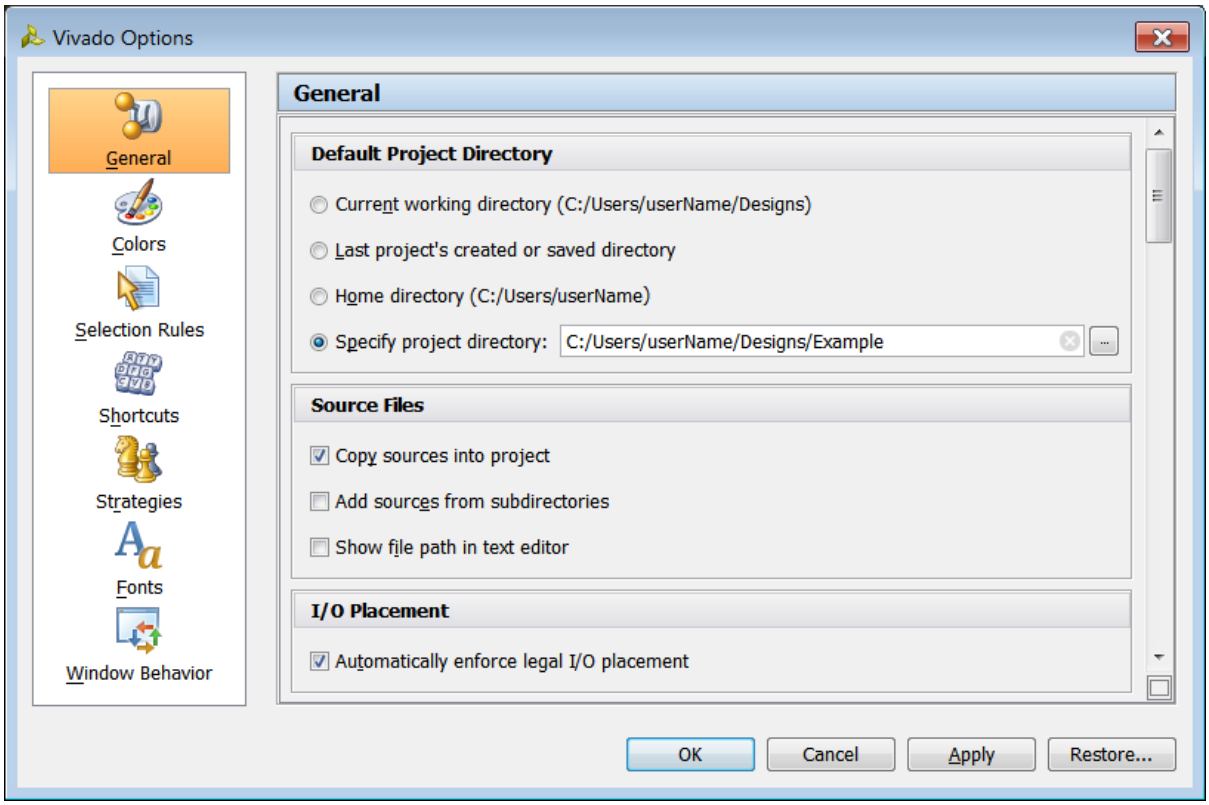


Figure 4-1: **General Options**

- **Default Project Directory**: Specifies where the Vivado IDE searches for existing projects, and where it writes newly created projects.
- **Source Files**: Specifies the default settings used when adding sources and displaying sources in the Vivado IDE Text Editor.
- **Target Language**: Sets the default target language used when a new project is created.
- **Recent**: Specifies the number of recent projects, directories, and files to list. You can also specify whether to automatically open the most recently used project when starting the Vivado IDE.
- **Message**: Specifies a limit for the number of messages to display with the same ID.
- **I/O Placement**: Specifies whether interactive I/O placement DRCs are enabled or disabled.
- **Connectivity Display**: Specifies the default connectivity display in the Device window. For more information, see Setting Device Window Display Options in Chapter 3.
- **File Saving**: Specifies where the Vivado IDE saves project files automatically when closing, or prompts to save changes.
- **Text Editor**: Sets the text editor to use when opening RTL sources or XDC files for modification. You can choose from a list of text editors, or you can specify the command line to launch a custom text editor. The Vivado IDE Text Editor is the default

editor. If you specify an alternate editor and the Vivado IDE detects an issue, the Vivado IDE Text Editor is also used.

You can also specify whether to use the Tab character or a specified number of spaces when a tab is inserted. This allows the text file to be portable to third-party applications that might not properly handle tab characters.

Note: Some features of the Vivado IDE are *not* supported when you use a third-party text editor.

- **Tooltips & Help**: Specifies the language and behavior of the tooltips that appear when you hover over commands in the Vivado IDE. You can set the amount of time before a tooltip appears as well as the amount of time before it disappears. You can also set whether to show tooltips for menu commands.
- **Documentation**: Specifies whether to open documents in Xilinx® Documentation Navigator or in your default web browser.
- **WebTalk**: Controls whether WebTalk is able to send Xilinx usage information.

  Note: For more information on WebTalk, see the *Vivado Design Suite User Guide: Release Notes, Installation, and Licensing* (UG973) [Ref 2].
- **IP Catalog**: Configures and updates the default IP catalog used in your newly created projects.
- **I/O Placement**: Specifies whether to run interactive I/O placement DRCs.
- **Package Pins**: Prohibits and sorts pins after using Set Configuration Modes.
- **3rd Party Tools**: Specifies the path to an alternate simulator, such as Mentor Graphics® ModelSim or QuestaSim tool. For more information, see the *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 9].

Specifying Colors

You can use the Colors Options to control the appearance of the viewing environment. To open the dialog box, select **Tools > Options**. In the dialog box, click the **Colors** category.

Using the Colors Options

At the bottom of the Colors Options dialog box, click one of the following views:

- General
- Device
- Package
- Bundle Nets

General

The General view (Figure 4-2) defines the colors used for elements such as the foreground, background, and highlight colors.

TIP: *The **Vivado Default Theme** is designed for optimal display on computer monitors. However, the **Vivado Light Theme** displays well when using a projector.*

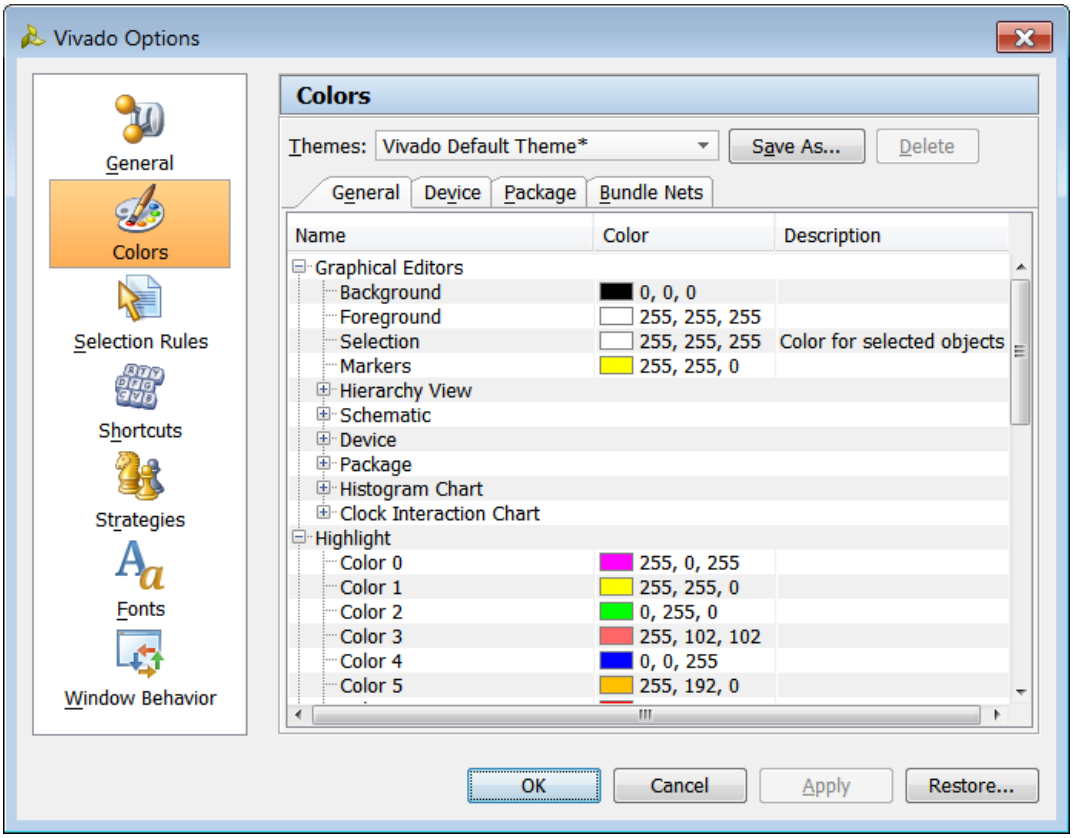


Figure 4-2: **Colors Options—General**

Device

The Device view (Figure 4-3) controls the colors of the objects shown in the Device window, including the following: Pblocks, I/O nets, slices, look-up tables (LUTs), and other device primitives. You can control the display of these objects in the Device window using the Device Options. For more information, see Setting Device Window Display Options in Chapter 3.

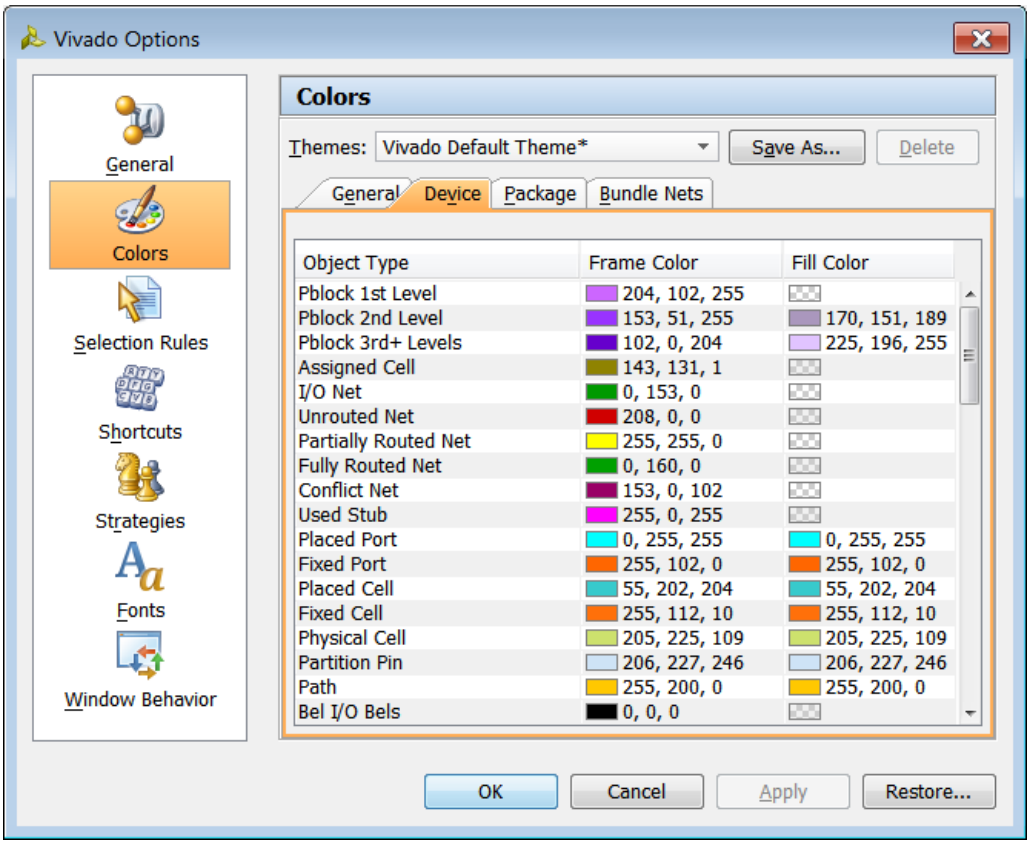


Figure 4-3: **Colors Options—Device**

Package

The Package view (Figure 4-4) controls the colors of the package pins that make up the FPGA I/O banks and appear in the Package window.

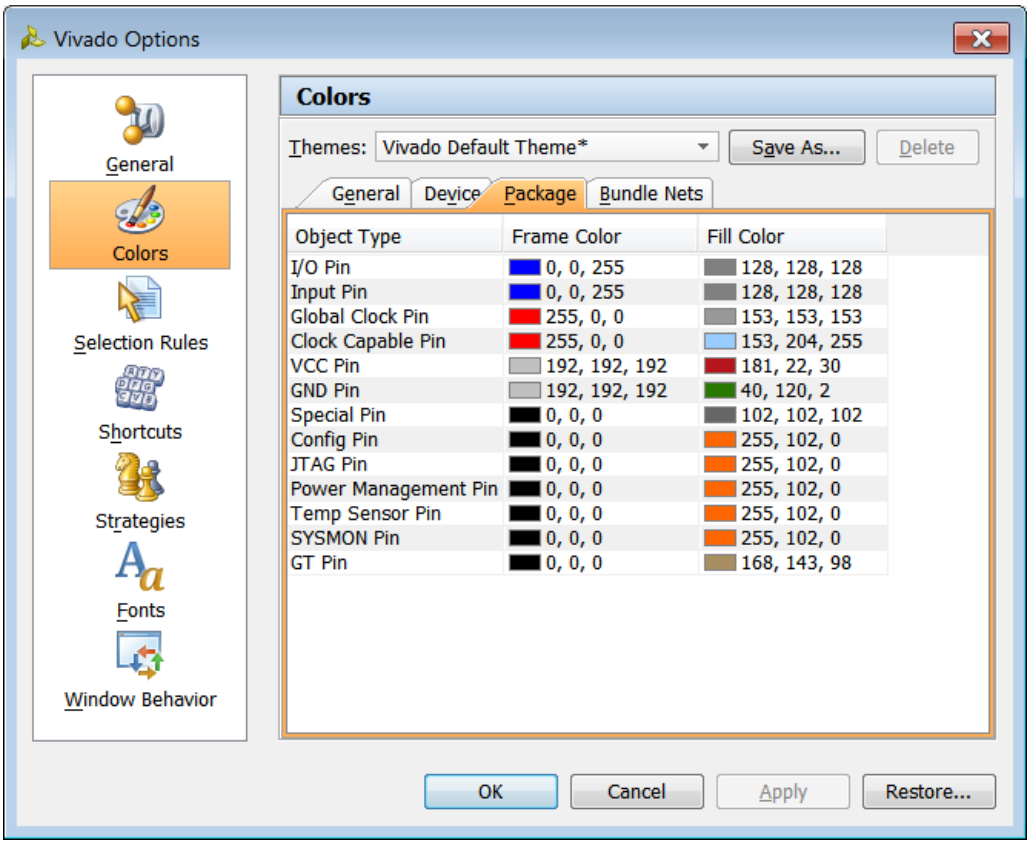


Figure 4-4: **Colors Options—Package**

Bundle Nets

The Bundle Nets view (Figure 4-5) controls the display for bundle nets, including color, width, and the number of nets joined into each bundle. The following settings are available:

- **From/To**: Adjusts the signal count ranges for bundles. Each column represents a bundle net range that can be independently configured.
- **Display**: Shows or hides the bundle. Enable the checkbox to show the bundle.
- **Width**: Defines the line width for the bundle.
- **Color**: Sets the color of the bundle nets.

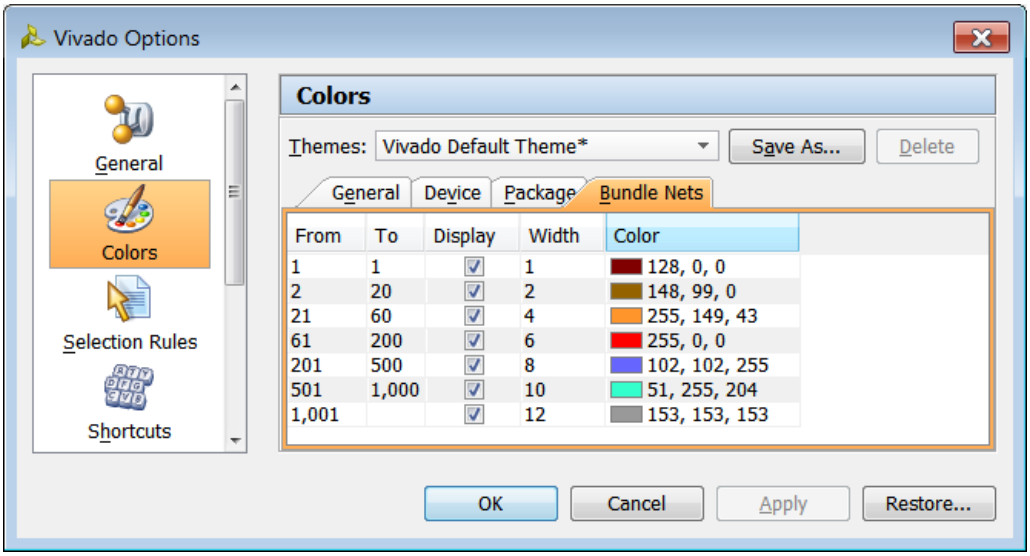


Figure 4-5: **Colors Options—Bundle Nets**

Customizing Display Themes

The Vivado IDE provides predefined light and dark background themes. To change the theme, select the **Light Theme** or **Default Theme** from the drop-down list.

Note: These default options are defined in the `vivado.ini` file. For more information, see Outputs for Environment Configuration in Appendix A.

Changing Colors

To change the color of an element, do one of the following:

- Click a color cell to enter a specific RGB value directly, as shown in Figure 4-6.

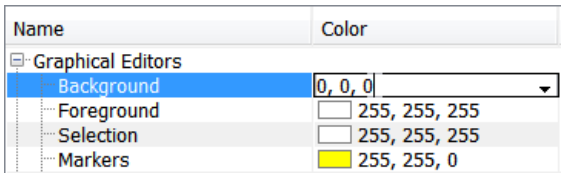


Figure 4-6: **RGB Color Values**

- Click the drop-down arrow to display the palette of available colors, and click a color to select it, as shown in Figure 4-7. Click **More Colors** to display additional color definition options.

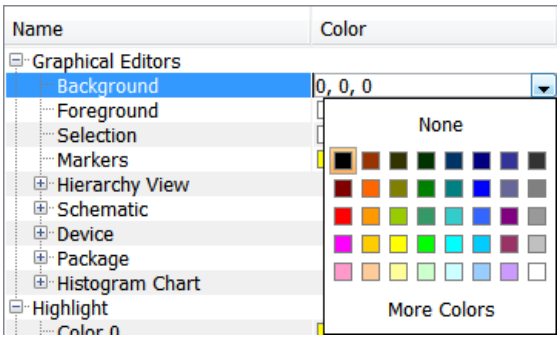


Figure 4-7: **Color Palette**

TIP: *You can also modify color settings in the Device window. For more information, see Setting Device Window Display Options in Chapter 3.*

Saving a Custom Theme

After you configure the color settings using the Color Options, you can save the settings for future use. Click the **Save As** button to specify a theme name and save the theme.

Setting Selection Rules

When selecting an object, associated or connected objects might also be selected. For example, selecting a Pblock might also select the netlist cells assigned to the Pblock. Selecting a port object might also select the pin object of the port.

You can use the Selection Rules Options (Figure 4-8) to control the secondary elements that are selected when you select a primary object. To open the dialog box, select **Tools > Options**. On the left side of the dialog box, click the **Selection Rules** category.

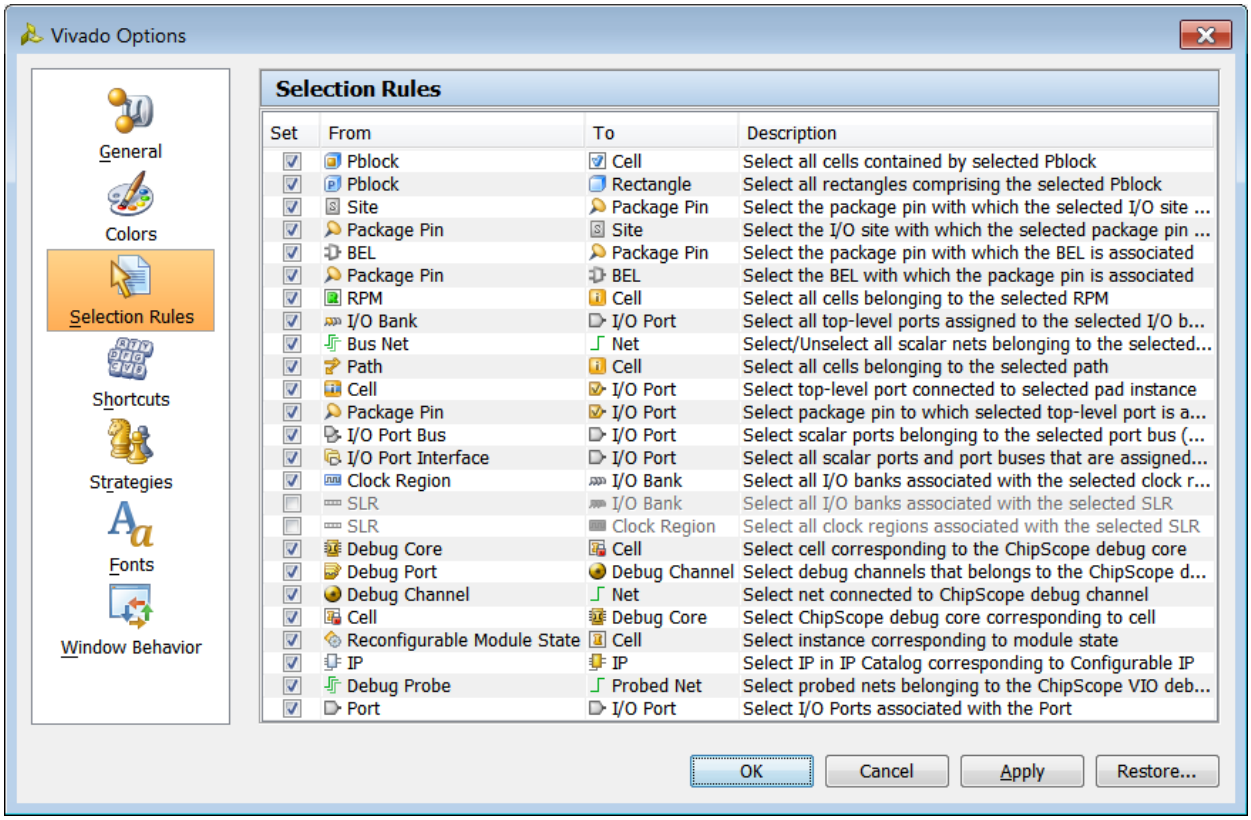


Figure 4-8: **Selection Rules Options**

The default selection rules enable Vivado IDE to operate in the most efficient manner. You can change these selection rules if you have trouble selecting a specific object.

To change a selection rule, enable or disable the **Set** checkbox next to the rule:

- When you enable a selection rule, both the primary **From** object type and secondary **To** object types are selected.
- When you disable a selection rule, only the primary **From** object type is selected.

Configuring Shortcut Keys

Many commonly used commands have predefined keyboard shortcuts. Shortcuts are displayed next to the command in the menu bar or popup menu. You can use the Shortcuts Options (Figure 4-9) to create custom shortcuts. To open the dialog box, select **Tools > Options**. On the left side of the dialog box, click the **Shortcuts** category.

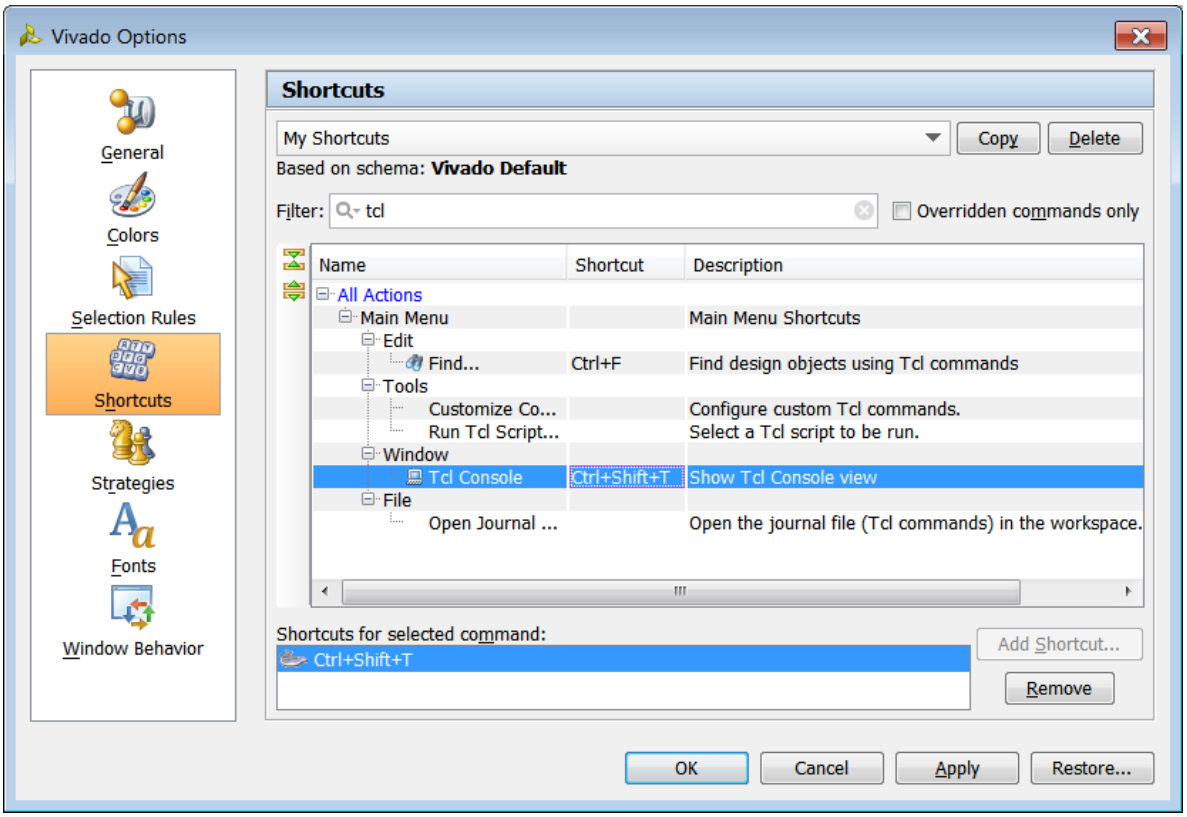


Figure 4-9: **Shortcuts Options**

To create custom shortcuts:

1. At the top of the dialog box, click **Copy** to create a new schema from the Default schema.

IMPORTANT: *You cannot modify the default shortcuts provided by the Vivado IDE. To customize shortcuts, you must create a new shortcut schema.*

2. In the popup window, specify a name for the new shortcut schema, and press **Enter**.
3. Search through the list of menus and windows, and select a command.

TIP: *Use the Filter field to filter the commands listed for shortcut assignment. Enter a text string to filter the list of available commands. You can use different shortcuts for the same command in different windows.*

4. Select **Add Shortcut**.
5. In the Add Shortcut dialog box, type the new shortcut, and click **OK**.

   Note: User-specified shortcuts are saved to the `shortcuts.xml` file in the Vivado IDE configuration directories. For more information, see Outputs for Environment Configuration in Appendix A.
6. To delete shortcuts, click **Remove**.

Creating Run Strategies

A strategy is a set of pre-configured command line options for the synthesis or implementation tools that is designed to resolve synthesis or implementation challenges in the design. Strategies are tool and version specific.

The Vivado IDE provides several commonly used strategies that are tested against internal benchmarks. You cannot change the command line settings for predefined Vivado IDE synthesis and implementation strategies. However, you can copy and modify supplied strategies to create your own custom strategies.

The Vivado IDE writes the user-defined strategies to:

- **Windows**: `%APPDATA%\Xilinx\Vivado\strategies`
- **Linux**: `~/.Xilinx/Vivado/strategies`

You can use the Strategies Options (Figure 4-10) to create custom strategies. To open the dialog box, select **Tools > Options**. On the left side of the dialog box, click the **Strategies** category.

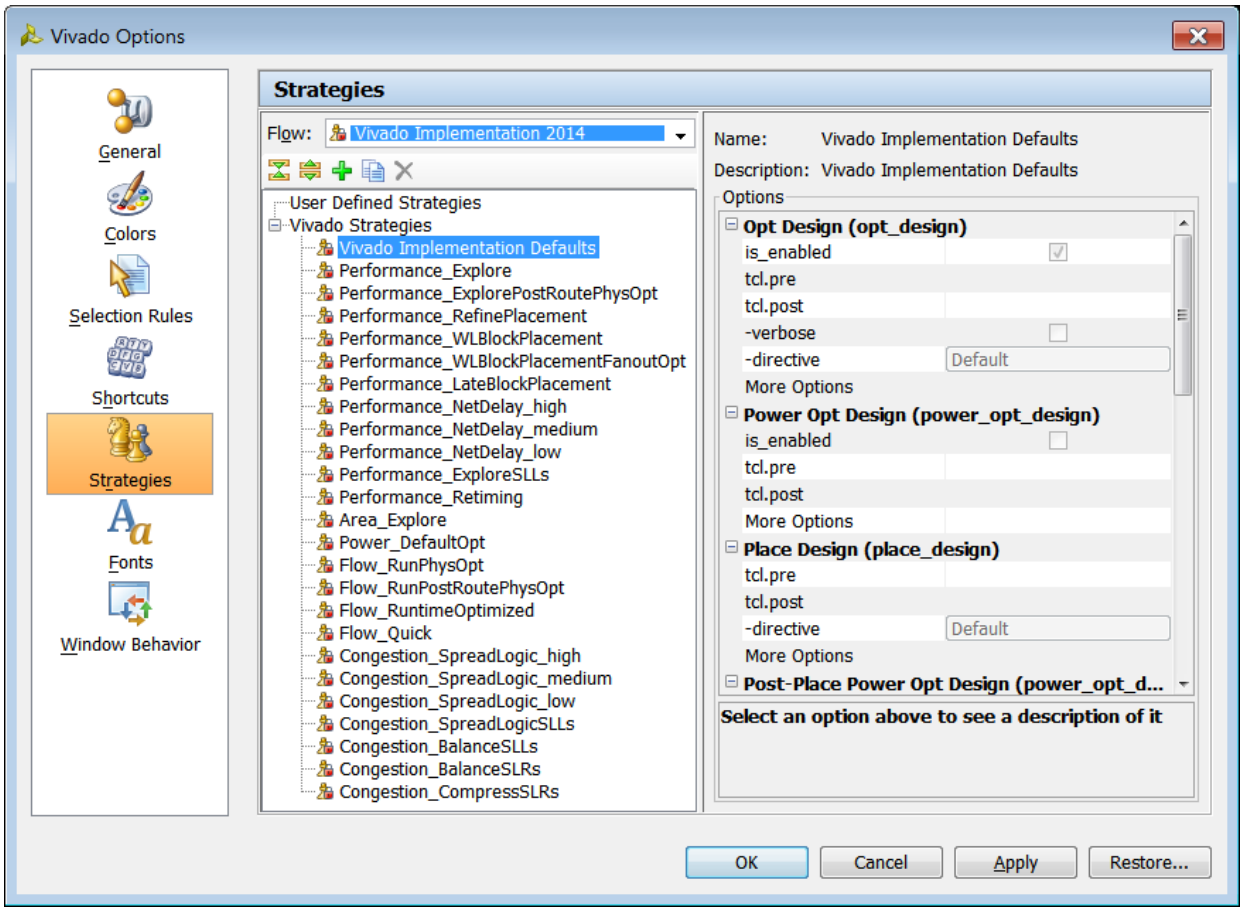


Figure 4-10: **Strategies Options**

To review, copy, and modify strategies:

1. From the Flow drop-down list, select a Vivado synthesis or Vivado implementation version.

   A list of strategies and their command line options displays. For more information on command line options, see the *Vivado Design Suite User Guide: Synthesis* (UG901) [Ref 10] and *Vivado Design Suite User Guide: Implementation* (UG904) [Ref 12].

IMPORTANT: *You cannot modify the default options for predefined Vivado IDE strategies. To customize strategies, you must copy or add a strategy.*

2. To create a new strategy, select **Create New Strategy** from the popup menu or toolbar  .

   Note: Alternatively, you can create a copy of an existing strategy using **Create a Copy of This Strategy** from the popup menu or toolbar  . The Vivado IDE creates a copy of the strategy in the User Defined Strategies list, and displays the command line options on the right side of the dialog box for you to modify.

3. In the New Strategy dialog box, set the following options, and click **OK**:
   - **Name**: Specifies the strategy name to assign to a run.
   - **Type**: Specifies whether the strategy applies to synthesis or implementation.
   - **Tool Version**: Specifies the Vivado Design Suite version.
   - **Description**: Provides the strategy description for display in the Design Run results table.

4. Edit the options for the command line tools used during the synthesis or implementation run as follows:
   - Click the checkbox to enable or disable an option.
   - Select a different value from the drop-down list.
   - Enter the appropriate text (for example, in the More Options field).

TIP: *Click a command option to view a description of the option at the bottom of the dialog box.*

5. Click **Apply** and **OK** to save the new strategy.

   The new strategy is listed under **User Defined Strategies** and can be used for synthesis or implementation.

Adjusting Fonts

You can use the Fonts Options dialog box (Figure 4-11) to customize the font style, size, and color for the Vivado IDE Text Editor, Log window, and Tcl Console. To open the dialog box, select **Tools > Options**. On the left side of the dialog box, click the **Fonts** category.

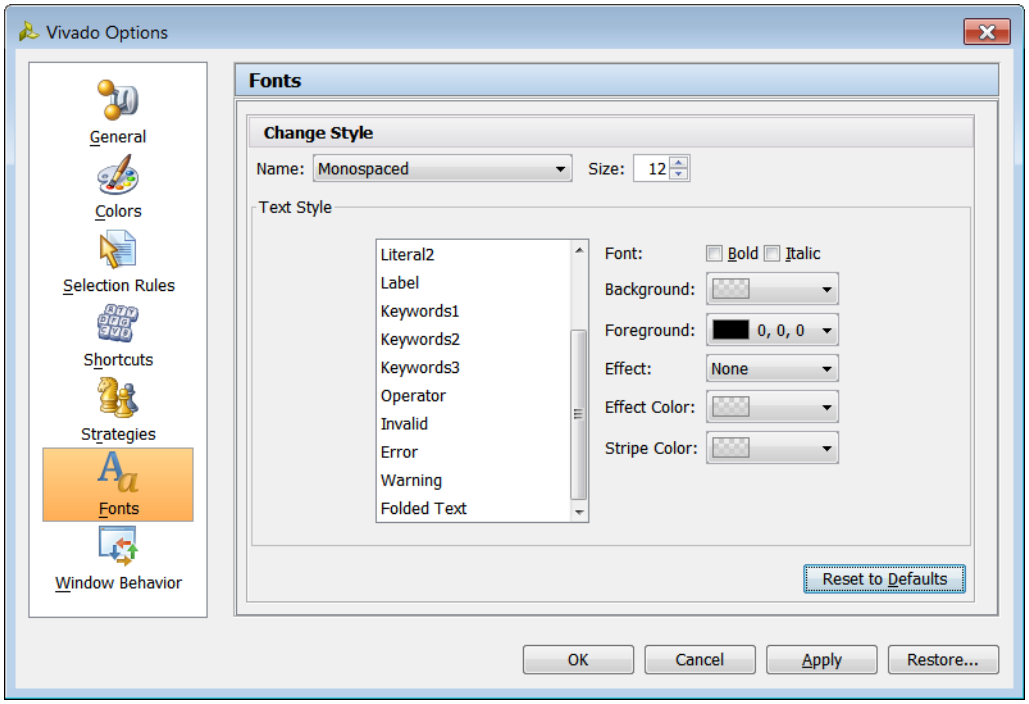


Figure 4-11: **Fonts Options**

Customizing Window Behavior

You can use the Window Behavior Options (Figure 4-12) to control how warnings, confirmations, notifications, and alerts display in the Vivado IDE. To open the dialog box, select **Tools > Options**. On the left side of the dialog box, click the **Window Behavior** category.

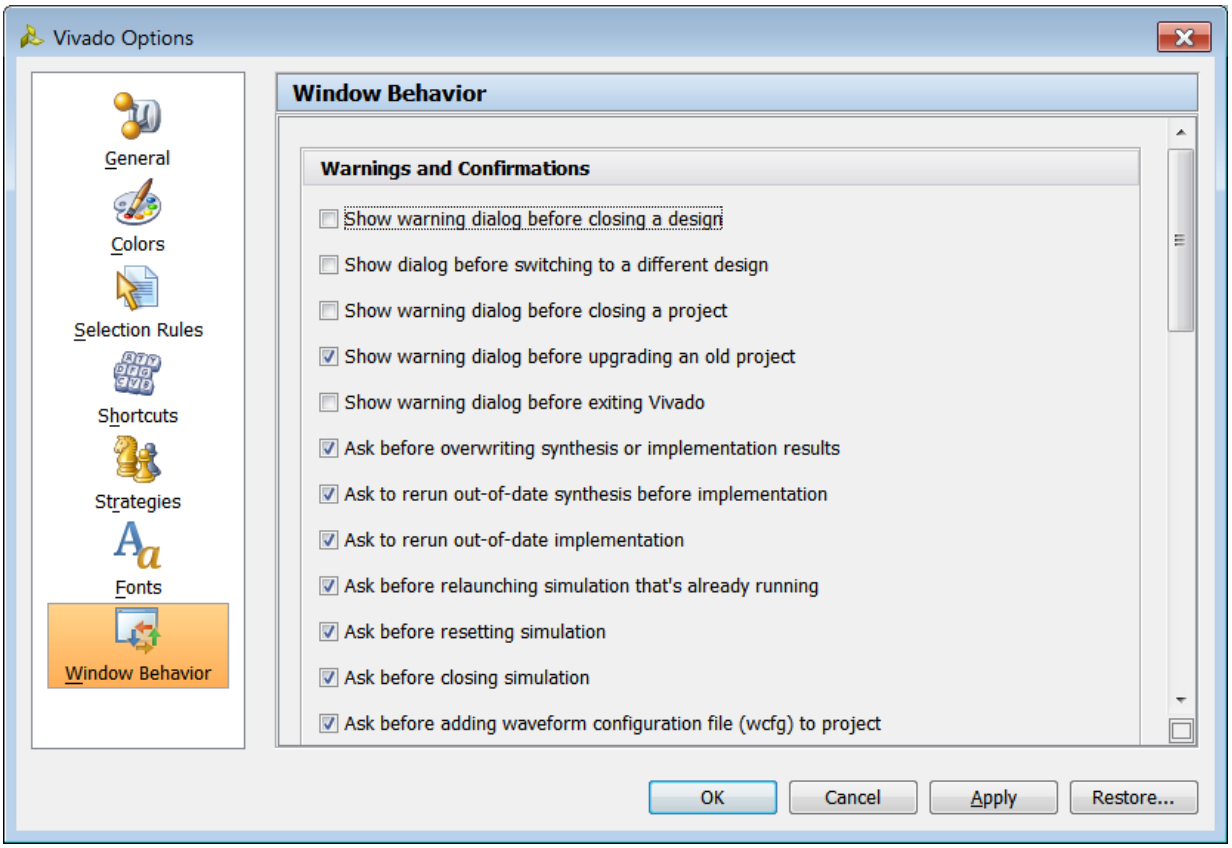


Figure 4-12: **Window Behavior Options**

You can specify the following Window Behavior Options:

- **Warnings & Confirmations**: Defines how the Vivado IDE shows warning and confirmation dialog boxes.
- **Notifications**: Defines which notifications the Vivado IDE displays.
- **Alerts**: Defines alerts for success or failure of non-active runs.

Creating Custom View Layouts

The Vivado IDE provides predefined view layout configurations to complete specific design tasks, such as the I/O Planning layout or the Design Analysis layout. These layouts define the location and size of views commonly used for a specific design task. You can also create, remove, and reset user-defined layouts using the following **Layout** menu commands:

- **Save Layout As**: Creates a user-defined layout based on the current layout configuration.
- **Remove Layout**: Removes the user-defined layout of your choice.
- **Reset Layout**: Restores resized or moved windows to the original configuration.
- **Undo**: Undoes the most recent view manipulation.
- **Redo**: Redoes the most recent view manipulation.

Note: User-defined layouts are saved to a layout file in your installation directory for use in all your design projects. For more information, see Outputs for Environment Configuration in Appendix A.

Adding Custom Menu Commands

You can use the Customize Commands dialog box (Figure 4-13) to add system or user-defined Tcl commands to the Vivado IDE main menu and toolbar menu. To open the dialog box, select **Tools > Custom Commands > Customize Commands**.

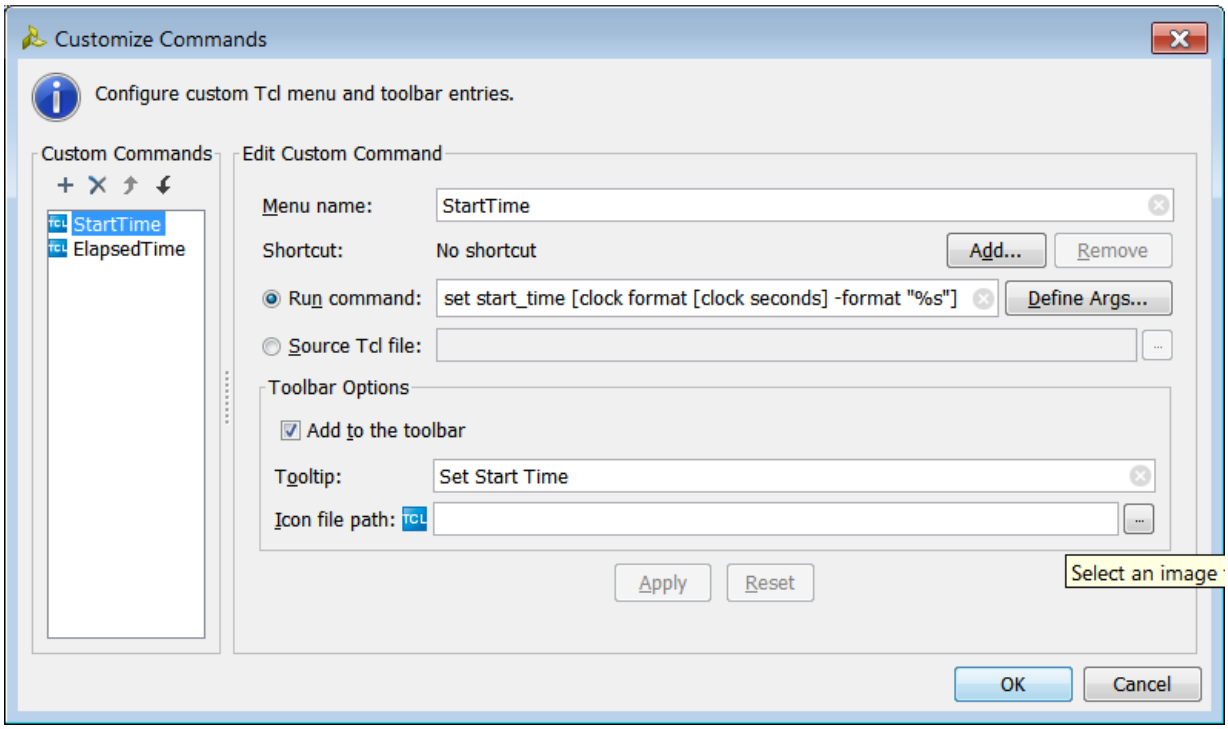


Figure 4-13: **Customize Commands Dialog Box**

You can specify the following options:

- **Custom Commands**: Displays the list of currently defined custom commands.
  - **Add Commands**: Click the plus (+) icon to specify new commands to add to the custom menu. In the popup window, type the command name, and press **Enter** to add the command to the list of custom commands.
  - **Remove Commands**: Click the X icon after selecting one or more commands to remove from the custom menu.
  - **Move Command Up**: Moves the selected command up in the list.
  - **Move Command Down**: Moves the selected command down in the list.
- **Edit Custom Command**: Specifies the properties of the selected command in the Custom Commands list.
  - **Menu Name**: Specifies the name of the command in the Custom Commands menu.
  - **Shortcut**: Defines a keyboard shortcut to use to run the Tcl command. Click **Add** to open the Add Shortcut dialog box in which you can enter a keystroke combination to run the selected command. The dialog box reports any commands that already use the specified shortcut.
  - **Run Command**: Runs the specified Tcl command or procedure from the custom menu.

- **Source Tcl File**: Causes the specified Tcl script file to be sourced rather than a single Tcl command or procedure to be run from the custom menu.
- **Toolbar Options**: Specifies whether to add an icon for the command to the main toolbar.
  - **Add to the Toolbar**: Enables the toolbar icon. When this checkbox is disabled, the command does not appear on the main toolbar.
  - **Tooltip**: Specifies the text to display as a tooltip when hovering over the command icon.
  - **Icon File Path**: Specifies the file path to the toolbar icon. Use a PNG, JPG, or GIF file of approximately 20x20 pixels. The Vivado IDE resizes larger images to fit onto the toolbar.

User-defined custom commands are available from the **Tools > Custom Commands** menu, with each command listed on the submenu.

Note: The Custom Command menu is persistent with the Vivado IDE and is restored each time the tool is launched. Custom commands are specific to each user and are saved to the `commands.paini` file output by the Vivado IDE. For more information, see Outputs for Environment Configuration in Appendix A.

Input and Output Files

Input Files

In the Vivado® IDE, you can specify the location of the files used as input.

Table A-1 lists the Vivado IDE input files, including files types and descriptions.

Table A-1: **Vivado IDE Input Files**

Input File	File Type	Description
Design Source Files	• VHDL • Verilog • Verilog Header • SystemVerilog	• You can import and elaborate files to analyze the logic or modify the source. • The original source files can be referenced and left in place, or they can be copied into the project for portability. • You can specify directories when importing RTL source files. • All recognized files and file types contained in the directories are imported into the project.
I/O Port Lists	CSV	• You can import a Comma Separated Values (CSV) format file to populate the I/O Ports view within the I/O Planning layout. This functionality is intended for use in an I/O Planning project only. • You can assign the I/O ports to physical package pins to define the device pin configuration. • CSV is a standard file format used by FPGA and board designers to exchange information about device pins and pinout.
Module-Level Netlists and Cores	• EDIF • NGC • NGO	• The Vivado IDE can construct a design using multiple EDIF or NGC netlists supporting a hierarchical design methodology. • When you select the top-level logic, lower-level modules are imported automatically. This process has more flexibility when updating the design. • The Vivado IDE incremental netlist import capability allows netlist updates at any level of the design hierarchy.

Table A-1: **Vivado IDE Input Files *(Cont'd)***

Input File	File Type	Description
Top-Level Netlists	• EDIF • NGC	• The Vivado IDE supports importing EDIF or NGC netlists. • The Vivado IDE can construct the design hierarchy using multiple netlists. • When you select the top-level logic, lower-level modules are imported automatically. Incremental netlist import capabilities allow netlist updates at any level of design hierarchy. • In-process floorplanning constraints are maintained through iterations.
Constraint Files	• XDC • SDC	• The Vivado IDE supports Synopsys Design Constraints (SDC) and Xilinx® Design Constraints (XDC) file formats. • The Vivado IDE can import multiple constraints files, which allows for separation of physical constraints, I/Os, and timing constraints.
Other Files	• BMM • ELF • MIF • COE	• BMM: Block RAM Memory Map (BMM) file is a text file that syntactically describes how individual block RAMs make up a contiguous logical data space. • ELF: An Executable and Linkable Format (ELF) file is a binary data file that contains an executable CPU code image ready for running on a CPU. • MIF: This file describes the memory contents that are used by a core, a cell, or simulation models. • COE: This file describes the initial memory and coefficients contents as input for core generation.

Output Files

By default, the Vivado IDE stores report output files as follows:

- Outputs from Tcl commands are written to the start-in directory, from which the Vivado IDE was launched.
- Outputs from the GUI are written to the project directory by default.
- When running synthesis or implementation, the output files are written to the run directories for the project.
- The default location of the journal and log files depend on the operating system:
  - **Windows**:
    - **Start menu**: `%APPDATA%\Xilinx\vivado`
    - **Command prompt**: Directory from which Vivado IDE is opened.
  - **Linux**: Directory from which Vivado IDE is opened.

Note: When the Vivado IDE is launched, backup versions of the journal file (`vivado_<id>.backup.jou`) and log file (`vivado_<id>.backup.log`) are written to save the details of the previous run. The `<id>` is a unique identifier that enables the tools to create and store multiple backup versions of the log and journal files. For more information on the journal and log file, see *Vivado Design Suite User Guide: Design Analysis and Closure Techniques* (UG906) [Ref 13].

Table A-2 lists the Vivado IDE output files, including files types and descriptions.

Table A-2: **Vivado IDE Output Files**

Output File	File Type	Description
I/O Pin Assignment	CSV	• I/O pin assignments are stored in a CSV format file that contains the I/O port assignment and relative package pin information. • This file is used for port definition without RTL header definition and PCB schematic symbol generation.
I/O Pin Assignment	• RTL • Verilog • VHDL	• Verilog or VHDL format file contains the I/O port assignments defined as ports in the file header in a legal language format. • This file is used for RTL port header definition.
Log File	• `vivado.log` • `vivado_<id>.backup.log`	• The log file (`vivado.log`) captures the contents of the messages created from running Vivado IDE commands. • To view the file, select **File > Open Log File** from the main menu.
Journal File	• `vivado.jou` • `vivado_<id>.backup.jou`	• The journal file (`vivado.jou`) captures the Tcl commands from a session. • To view the file, select **File > Open Journal File** from the main menu. • You can replay the journal file to reproduce the commands of the previous session. • You can create Tcl scripts by copying commands from the journal file for later replay. • It might be necessary to edit this file to remove any erroneous commands or commands from multiple sessions prior to replay. • Not every action logs a Tcl command into the journal file.

Table A-2: **Vivado IDE Output Files *(Cont'd)***

Output File	File Type	Description
Error Log Files	• vivado_pid<id>.debug • hs_err_pid<id>.log • vivado_pid<id>.str • vivado_pid<id>.png • vivado_pid<id>.zip	• The error files can provide valuable information for debugging the Vivado IDE in the event of an unexpected interrupt. • The steps to reproduce (STR) file provides useful information about the actions taken in the Vivado IDE prior to the error. • If the Vivado IDE issues a dialog box that warns of an internal exception error, the error files are stored. • These files contain no design data. • The PNG file is a image of the Vivado IDE at the time of the error. • When you open a case with Xilinx Technical Support, include the archive file (vivado_pid<id>.zip). If the archive file does not exist, include the following files instead: ◦ Journal file (vivado.jou) ◦ Log file (vivado.log) ◦ Error log debug file (vivado_pid<id>.debug) ◦ Error log file (hs_err_pid<id>.log) ◦ Error log steps to reproduce file (vivado_pid<id>.str) ◦ Vivado IDE image file (vivado_pid<id>.png)
DRC Results	User-Defined	• Each time Design Rule Checks (DRC) is run, the results can be written to a file.
Table Data	Excel File	• Most data displayed in a table format can be exported to a spreadsheet format file. • To export the data, select **Export to Spreadsheet** from the popup menu in any window that displays data in table form.
SSN Analysis Report	HTML, CSV	• The results from Simultaneous Switching Noise (SSN) analysis can be exported to a CSV or HTML report file by specifying a file name and location in the Run SSN Analysis dialog box.
Strategy Files	PSG	• The /Strategy directory contains files with your specified default command line options. • You can apply a strategy to any given run. • You can either create strategies or copy a supplied strategy.

Outputs for Environment Configuration

The Vivado IDE saves the current configuration of window layouts and themes to configuration and initialization files that are loaded when the tool is launched. You can also save custom themes, window layouts, and run strategies to be loaded when you need them. For more information, see Chapter 4, Configuring the Environment.

The files defining these themes and layouts are written to the Vivado IDE environment folders in the following locations:

- **Windows**: `%APPDATA%\Xilinx\vivado\<version>`
- **Linux**: `~/.Xilinx/vivado/<version>`

Table A-3 lists the environment configuration files. input files, including files names and descriptions.

Table A-3: **Vivado IDE Environment Configuration Outputs**

Environment Configuration File	File Name	Description
View Display Options	`vivado.ini`	• `../vivado/<version>/vivado.ini` file captures the settings in **Tools > Options** that include display color and other viewing options for the environment. • The Vivado IDE saves your settings to the `vivado.ini` file when you exit the tool. Upon opening, it imports the file automatically and applies the settings to initialize the tool.
Vivado IDE Themes	`<theme_name>.patheme`	• `../vivado/<version>/themes` directory contains `*.patheme` files that are created when you customize color and fill pattern themes for displaying layers in the Vivado IDE. • You can select a theme file to use during the active session from a pull-down selection menu.
View Layout Files	`<layout_name>.layout`	• `../vivado/<version>/layouts` directory contains `*.layout` files that define the layout configuration of the Vivado IDE. • You can create custom view layouts by selecting **Layout > Save Layout As**.

Table A-3: **Vivado IDE Environment Configuration Outputs *(Cont'd)***

Environment Configuration File	File Name	Description
Keyboard Shortcuts	`shortcuts.xml`	• `../vivado/<version>/shortcuts` directory contains a `shortcuts.xml` file that maps keyboard shortcuts to tool commands. • You can define and configure multiple keyboard shortcuts, which are stored in the shortcuts file.
Custom Commands	`commands.paini`	• `../vivado/<version>/commands` directory contains the `commands.paini` file that stores custom Tcl commands added to the Vivado IDE. • You can create custom commands by selecting **Tools > Custom Commands > Customize Commands**.

Outputs for Project Data

Table A-4 lists the Vivado IDE project data output, including file names and descriptions.

Table A-4: **Vivado IDE Project Data Outputs**

Project Data Output	File Name	Description
Project Directory	`<projectname>`	• When you create a new project, the Vivado IDE optionally creates a project directory in which to store the project file, the project data directory, and the implementation results. • The project directory has the same name as the project name entered in the New Project wizard.
Project File	`<projectname>.xpr`	• When you create a new project, the Vivado IDE creates a project file. • The project file has the same name as the project name entered in the New Project wizard.
Message Suppression File	`<projectname>.filter`	• When you suppress a message, the Vivado IDE creates a message suppression file, which is a binary file that contains message suppression rules. • The message suppression file has the same name as the project file.

Table A-4: **Vivado IDE Project Data Outputs *(Cont'd)***

Project Data Output	File Name	Description
Project Data Directory	`<projectname>.data`	• When you create a new project, the Vivado IDE creates a project data directory in which to put project metadata. • The project data directory has the same name as the project name entered in the New Project wizard.
Project Data Constraint Set Subdirectories	`<constraint_set_name>`	• This is the main subdirectory that corresponds to the metadata for the constraint fileset (named `constrs_1` by default). • For each additional constraint set created for a project, a top-level directory is created (named `constrs_2`, `constrs_3`, and so forth by default).
Project Sources Directory	`<projectname>.srcs`	• The project sources directory stores the HDL source files that are imported into a project. • `ip` and `bd` subdirectories contain files from imported IP, generated IP, and block designs.

Project IP Integrator Directory	`<projectname>.srcs/`
	`<source_set>/bd/`
	`<design_name>`
		• The `hdl` subdirectory contains the top-level HDL file and wrapper.
		• The `ip` subdirectory contains a subfolder for each IP in the block design.
		• The `ui` subdirectory contains data files for the graphical layout of the block design.

Outputs for Project Data Simulation

The project simulation directory structure for behavioral simulation runs is:

```
project_name/project_name.sim/sim_run_name/sim_#
```

CAUTION! *By default, the contents of this directory are deleted when the run is reset and are regenerated when the run is launched again.*

Table A-5 lists the file/directory name, simulation type, and description for the simulation runs.

Table A-5: **Behavioral and Timing Simulation Files and Directories**

File/Directory Name	Simulation Type	Description
`exelab.log`	Behavioral	Vivado simulator compilation and elaboration log file.
`exelab.pb`	Behavioral	Vivado simulator compilation and elaboration message file.
`<testbench>.tcl`	Behavioral	Vivado simulator Tcl command for waveform manipulation.
`<testbenc>.prj`	Behavioral	List of project files with library association sent for compilation to Vivado simulator XELAB command.
`<testbench>_behav.wdb`	Behavioral	Waveform database file created by Vivado simulator.
`xsim.ini`	Behavioral	File containing logical-to-physical mappings of libraries.

Outputs for Implementation

Table A-6 provides a brief description of the files that the Vivado IDE creates during implementation design operations.

IMPORTANT: *Do not modify the implementation files manually. These files are maintained by the Vivado IDE.*

Table A-6: **Outputs For Implementation**

Output File	File Name	Description
Run Directory	`<projectname>.runs`	• This directory contains all the scripts, input, and output files necessary for the first implementation run in the project (named `impl_1` by default). • For each additional implementation run created, a directory parallel to `impl_1` is created (named `impl_2`, `impl_3`, and so forth by default).
Run Implementation and Launch Vivado IDE Runs	• `<name>.rpt` • `<name>.pb` • `<top>_<step>.dcp` • `runme.log`	• Reports generated as part of the run appear as `<name>.rpt` and are viewable in the Vivado IDE Text Editor. • Messages are stored in `<name>.pb` files. • A checkpoint is written at each step of the implementation flow and stored as `<top>_<step>.dcp`. • The log of the run is stored as `runme.log`.
Launch Scripts	• `<top>.tcl` • `runme.bat, runme.sh` • `vrs_config_<#>.xml` • `vivado.begin.rst, vivado.end.rst`	• When you launch a run, the Vivado IDE creates launch scripts automatically. These scripts contain commands and command-line options specified in the Vivado IDE strategy. • The `vrs_config_<#>.xml` files are located under the project run directory in a `/jobs` subdirectory. These files define the run name, directory, steps, and so forth. • `vivado.begin.rst` tracks that the run launched, and `vivado.end.rst` tracks that it finished.

Appendix B

Additional Resources and Legal Notices

Xilinx Resources

For support resources such as Answers, Documentation, Downloads, and Forums, see Xilinx Support.

Solution Centers

See the Xilinx Solution Centers for support on devices, software tools, and intellectual property at all stages of the design cycle. Topics include design assistance, advisories, and troubleshooting tips.

References

1. *Vivado® Design Suite User Guide: Design Flows Overview* (UG892)
2. *Vivado Design Suite User Guide: Release Notes, Installation, and Licensing* (UG973)
3. *Vivado Design Suite User Guide: Getting Started* (UG910)
4. *Vivado Design Suite User Guide: Using Tcl Scripting* (UG894)
5. *Vivado Design Suite Tcl Command Reference Guide* (UG835)
6. *Vivado Design Suite Tutorial: Design Flows Overview* (UG888)
7. *Vivado Design Suite User Guide: Designing with IP (*UG896)
8. *Vivado Design Suite User Guide: Designing IP Subsystems Using IP Integrator* (UG994)
9. *Vivado Design Suite User Guide: Logic Simulation* (UG900)
10. *Vivado Design Suite User Guide: Synthesis* (UG901)
11. *Vivado Design Suite User Guide: Using Constraints* (UG903)
12. *Vivado Design Suite User Guide: Implementation* (UG904)

13. *Vivado Design Suite User Guide: Design Analysis and Closure Techniques* (UG906)
14. *Vivado Design Suite User Guide: Programming and Debugging* (UG908)
15. *Vivado Design Suite User Guide: System-Level Design Entry* (UG895)
16. *Vivado Design Suite User Guide: I/O and Clock Planning* (UG899)
17. *System Generator for DSP User Guide* (UG640)
18. *Vivado Design Suite User Guide: Power Analysis and Optimization* (UG907)
19. Vivado Design Suite QuickTake Video: Understanding Messaging
20. Vivado Design Suite QuickTake Video: Messages, Reports, and Log Files Overview
21. Vivado Design Suite QuickTake Video Tutorials
22. Vivado Design Suite Documentation

Please Read: Important Legal Notices