Vivado Design Suite User Guide

Using the Vivado IDE

Vivado Design Suite

UG893 (v2022.1) April 27, 2022

Xilinx is creating an environment where employees, customers, and partners feel welcome and included. To that end, we're removing non-inclusive language from our products and related collateral. We've launched an internal initiative to remove language that could exclude people or reinforce historical biases, including terms embedded in our software and IPs. You may still find examples of non-inclusive language in our older products as we work to make these changes and align with evolving industry standards. Follow this link for more information.

AMD
XILINX

AMD
XILINX

Table of Contents

AMD XILINX

Chapter 1

Using the Vivado IDE

Navigating Content by Design Process

Xilinx® documentation is organized around a set of standard design processes to help you find relevant content for your current development task. All Versal® ACAP design process Design Hubs and the Design Flow Assistant materials can be found on the Xilinx.com website. This document covers the following design processes:

- **Hardware, IP, and Platform Development:** Creating the PL IP blocks for the hardware platform, creating PL kernels, functional simulation, and evaluating the Vivado® timing, resource use, and power closure. Also involves developing the hardware platform for system integration.
- **System Integration and Validation:** Integrating and validating the system functional performance, including timing, resource use, and power closure.
- **Board System Design:** Designing a PCB through schematics and board layout. Also involves power, thermal, and signal integrity considerations.

Introduction

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, functional, and timing 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 design.

IMPORTANT! *The Vivado IDE supports designs that target 7 series and newer devices only.*

TRAINING: *To help you learn more about the concepts presented in this document, you can attend the Designing FPGAs Using the Vivado Design Suite 1, Designing FPGAs Using the Vivado Design Suite 2, Designing FPGAs Using the Vivado Design Suite 3, and Designing FPGAs Using the Vivado Design Suite 4 Training Courses.*

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 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).

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 this link in the *Vivado Design Suite User Guide: Design Flows Overview* (UG892).

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

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, Xilinx recommends running it from a project directory, 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. Alternatively, you can launch the Vivado IDE by double-clicking the project file (.xpr extension) to ensure that the log and journal files are written to the project directory.*

Launching the Vivado IDE on Windows

Select **Start → All Programs → Xilinx Design Tools → Vivado <version> → Vivado <version>**.

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

Figure 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:

```
<install_path>/Vivado/<version>/bin/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`.

TIP: *To add the Vivado tools path to your current shell/command prompt, run `settings64.bat` or `settings64.sh` from the `<install_path>/Vivado/<version>` directory.*

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) and *Vivado Design Suite Tcl Command Reference Guide* (UG835). 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).

***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 <version> → Vivado <version> 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.

Related Information

Using the Tcl Console

Running a Tcl Script from the Vivado IDE

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

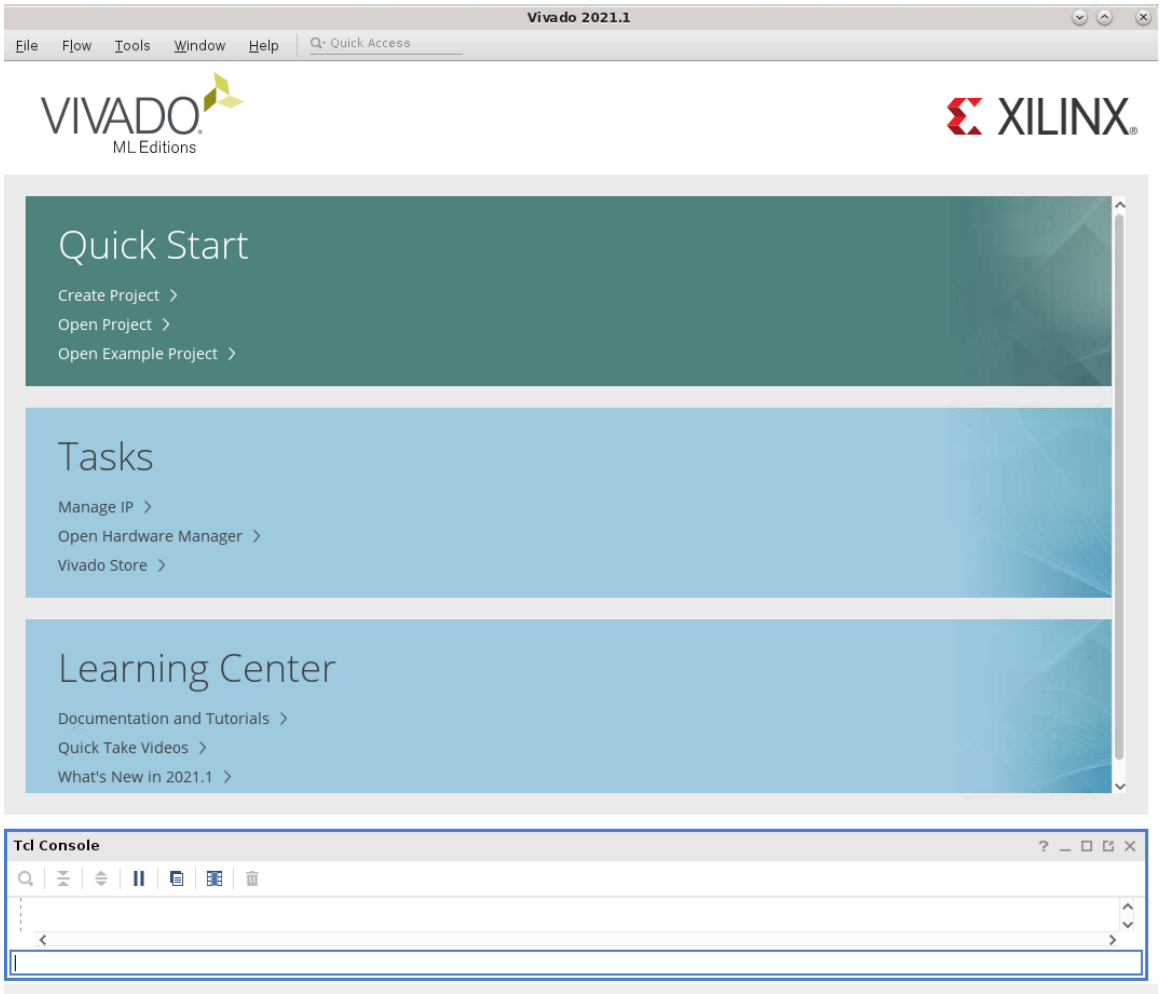
Note: To create scripts, you can copy the Vivado IDE Tcl commands from the vivado.jou file or from the Tcl Console.

Using the Getting Started Page

When you open the Vivado IDE, the Getting Started Page appears as shown in the following figure.

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

Figure 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 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 Synplify tool.
  - **Open Project:** Opens a browser that enables you to open any Vivado IDE project file (`.xpr` extension).
  - **Open Example Project:** Opens the Open Example Project wizard to guide you through creating a project. Following is an example project, including specifying a project name and location, and choosing from a list of valid parts:

    . BFT: Small RTL project

    Below are few designs:

    - **Configurable MicroBlaze Design Presets:** Vivado IP integrator MicroBlaze™ processor design targeting various Xilinx evaluation boards. The design allows users to configure in three different modes as a MicroController, Real-Time Processor & Application Class processor. You can implement the design in the Vivado Design Suite, export the hardware to the Vitis™ software platform for application code development, and simulate the design in the Vivado Design Suite using a test bench that you supply and an ELF file generated by the Vitis software platform.

      Note: Configurable MicroBlaze is not available by default. It must be installed through Git.

    - **Configurable Zynq UltraScale+ MPSoC Design:** Zynq® UltraScale+™ MPSoC design targeting various Xilinx evaluation boards. The designs provide the users with memory virtualization, hardware virtualization, memory protection units, and tightly coupled memories that are required for real-time deterministic applications and executing platform OSes. You can implement the design in the Vivado Design Suite, export the hardware to the Vitis software platform for application code development, and simulate the design in the Vivado Design Suite using a test bench that you supply and an ELF file generated by the Vitis software platform.

      Note: Configurable Zynq UltraScale+ MPSoC is not available by default. It must be installed through Git.

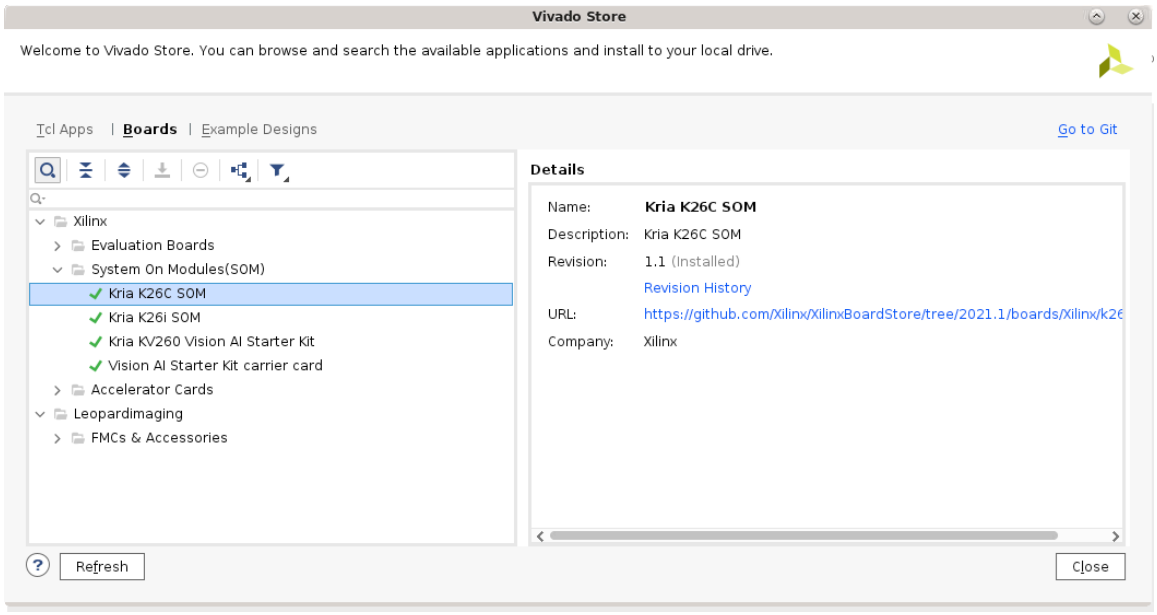
    - **CPU (HDL):** Large mixed-language RTL project.
    - **CPU (Synthesized):** Large synthesized netlist project.
    - **Wavegen (HDL):** Small project that includes three embedded IP cores. You can use this design to learn how to use integrated IP cores with Vivado IDE projects.

TIP: *Click **Refresh** to update the installed designs or download the latest Xilinx® Example designs.*

- 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 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.
- **Vivado Store:** The Vivado store, shown in the following figure, consolidates Tcl apps, board files and configurable example designs into a single location. A catalog file maintains the list of all items available in the stores. To update the catalog, click the refresh button for the respective store in the lower left-hand corner. Individual items can be installed or removed. Xilinx delivers a set of board files and example designs and installs, which cannot be uninstalled. This is because if customers are inside a firewall or do not have access to the internet, they should have access to the board files of Xilinx Proprietary Boards.
  - **Tcl Apps:** An open source repository of Tcl code designed primarily for use 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. For more information, see this link in the *Vivado Design Suite User Guide: Using Tcl Scripting* (UG894).
  - **Boards:** A GitHub repository for Xilinx and third-party hosted board files. Using a board file with Vivado can simplify design creation by integrating board level resources into the design environment. For more information about contributing boards, refer to https://github.com/Xilinx/XilinxBoardStore.
  - **Example Designs:** A GitHub repository comprised of Xilinx and third-party configurable example designs. These designs are intended to demonstrate specific capabilities of the tool and provide a baseline design. For more information about contributing example designs, refer to https://github.com/Xilinx/XilinxCEDStore.

Figure 3: **Vivado Store**



- Learning Center
  - **Documentation and Tutorials:** Opens or downloads Vivado Design Suite documentation using the Xilinx Documentation Navigator or your default web browser.
  - **QuickTake Videos:** Opens Xilinx video tutorials.
  - **What's new in 2021.2:** Opens "What's new browser page."

***Note*:** For more information about the Xilinx Documentation Navigator, see the *Vivado Design Suite User Guide: Getting Started* (UG910).

- Recent Project, Recent Checkpoints, and Recent IP Locations:

  Provides one-click access to recently opened projects, checkpoints, or IP locations. These lists only appear after you open projects, checkpoints, or IP locations.

***Note*:** By default, the last ten previously opened projects, checkpoints, or IP locations are listed. To change this number, select **Tools → Settings**, and update the Recent settings in the Project options under Tool Settings. The Vivado IDE checks that the project data is available before displaying the projects.

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).

AMD XILINX

Chapter 2

Using the Viewing Environment

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)
- *Vivado Design Suite User Guide: Designing with IP* (UG896)
- *Vivado Design Suite User Guide: Designing IP Subsystems Using IP Integrator* (UG994)
- *Vivado Design Suite User Guide: Logic Simulation* (UG900)
- *Vivado Design Suite User Guide: Synthesis* (UG901)
- *Vivado Design Suite User Guide: Using Constraints* (UG903)
- *Vivado Design Suite User Guide: Implementation* (UG904)
- *Vivado Design Suite User Guide: Design Analysis and Closure Techniques* (UG906)
- *Vivado Design Suite User Guide: Programming and Debugging* (UG908)

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

The following figure shows the Vivado IDE viewing environment. You can interact with the Vivado IDE through mouse, keyboard, or Tcl input.

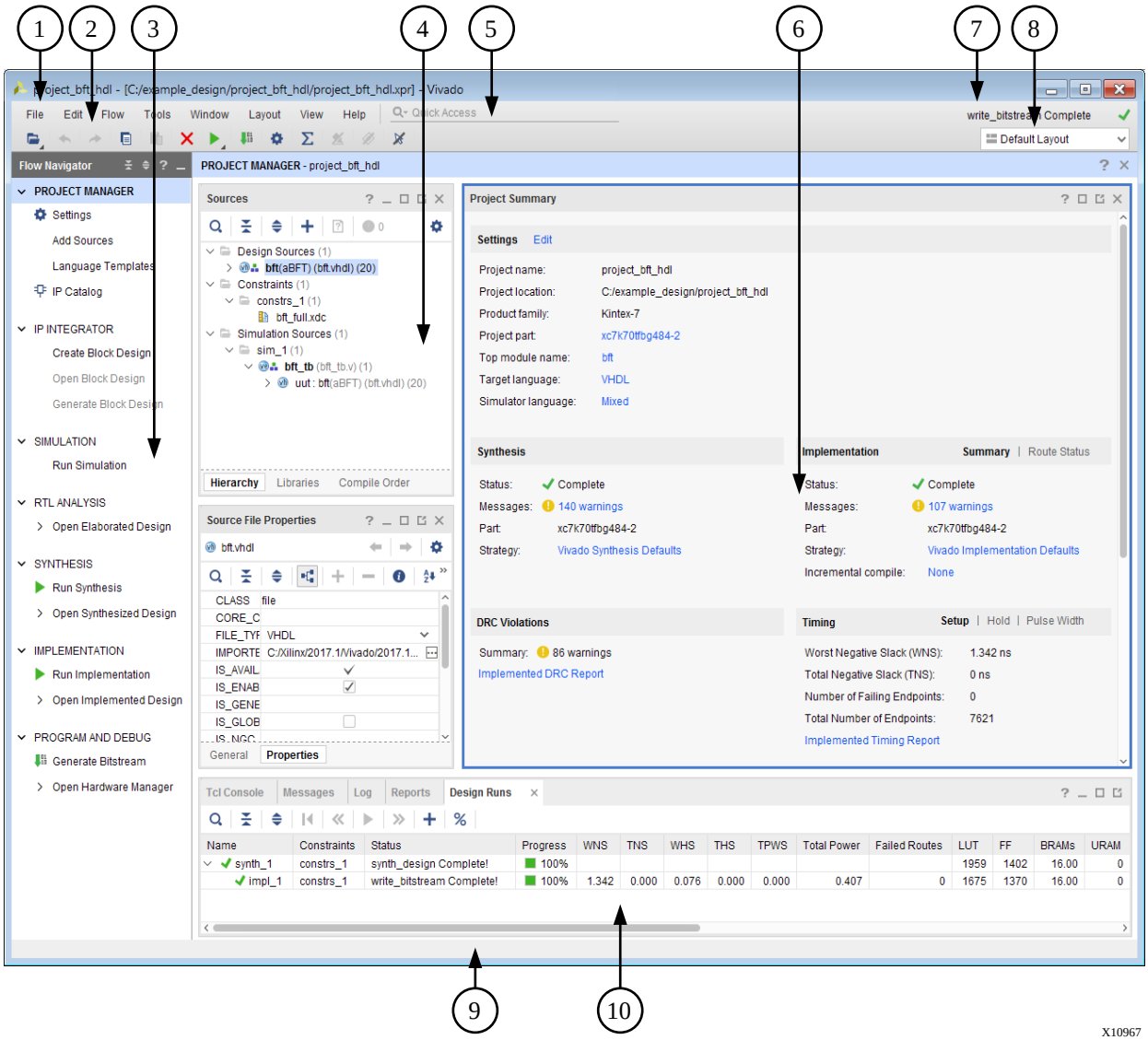
TIP: *For quick access to information on different parts of the Vivado IDE, click the Quick Help button ? in the window or dialog box, or press the **F1** key.*

The main components of the viewing environment are:

1. Menu Bar
2. Main Toolbar
3. Flow Navigator
4. Data Windows Area

5. Menu Command Quick Access Search Field
6. Workspace
7. Project Status Bar
8. Layout Selector
9. Status Bar
10. Results Windows Area

Figure 4: **Vivado IDE Viewing Environment**



Menu Bar

The main menu bar provides access to Vivado IDE commands. Commonly-used commands always display (for example, **File → Project → Open**) while others display only when a design is active (for example, **Reports → 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.

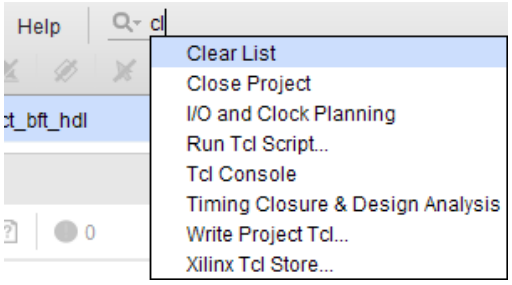
Related Information

Configuring Shortcut Keys

Menu Command Quick Access Search Field

To the right of the menu commands, the menu command Quick Access search field enables you to 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 include your search criteria appear, as shown in the following figure. Select a command from the list to execute that command.

Figure 5: **Menu Command Quick Access 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 and disappears. You can also set whether to show tooltips for menu commands. Select* ***Tools → Settings****. In the Tool Settings section of the Settings dialog box, click the* ***Help*** *category, and set the Tooltips and Quick Help settings.*

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 settings, add or create sources, view language templates, and open the Vivado IP catalog. For information on adding sources, see this link in the *Vivado Design Suite User Guide: System-Level Design Entry* (UG895). For information on language templates, see Using Language Templates. For information on the IP catalog, see this link in the *Vivado Design Suite User Guide: Designing with IP* (UG896).
- **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).
- **Simulation:** Change simulation settings or simulate the active design. For information on simulation, see the *Vivado Design Suite User Guide: Logic Simulation* (UG900)
- **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. For information on elaborating the RTL design, see this link in the *Vivado Design Suite User Guide: System-Level Design Entry* (UG895).
- **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).
- **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).
- **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).

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

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 stages of the design loaded, you can click the sections in the Flow Navigator to switch between design stages (for example, RTL Analysis or Synthesis).

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 Flow Navigator button .

Note: To show the Flow Navigator, select **View → Show Flow Navigator**, press **Ctrl+Q**, or click the **Flow Navigator** tab on the left edge of the Vivado IDE.

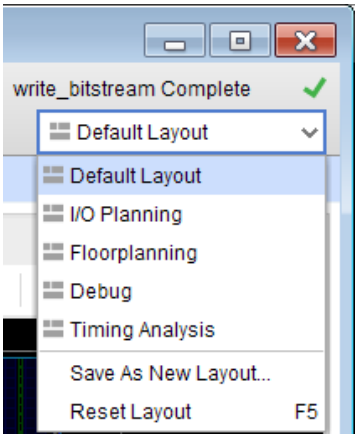
Related Information

Using Language Templates
Using the Schematic Window
Design Runs Window Commands

Layout Selector

The Vivado IDE provides predefined window layouts to facilitate various tasks in the design process. The layout selector (shown in the following figure) enables you to easily change window layouts. Alternatively, you can change layouts using the Layout menu in the menu bar.

Figure 6: **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.

  Note: When working with I/O planning projects, the I/O Planning view layout is called the Default Layout.

- **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.
- **ECO:** Open the ECO Navigator to make engineering change orders (ECOs) to the post-synthesis netlist.

  ***Note*:** The ECO layout is only available when you open a design checkpoint (DCP). For more information on the ECO flow, see this link in the *Vivado Design Suite User Guide: Implementation* (UG904) and this link in the *Vivado Design Suite User Guide: Programming and Debugging* (UG908).

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

Related Information

Configuring Custom View Layouts

Project Status Bar

The project status bar displays the following:

- Current status of the active design
- Descriptions for menu commands and toolbar buttons that you hover over
- Information about the selected text file that you are editing, including line numbers and modes
- Position of elements that you hover over in the Device and Package windows

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.*

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.*

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 Using Windows.

Related Information

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 Using Windows.

Related Information

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 Using Windows.

Related Information

Using Windows

Status Bar

The status bar displays the following information:

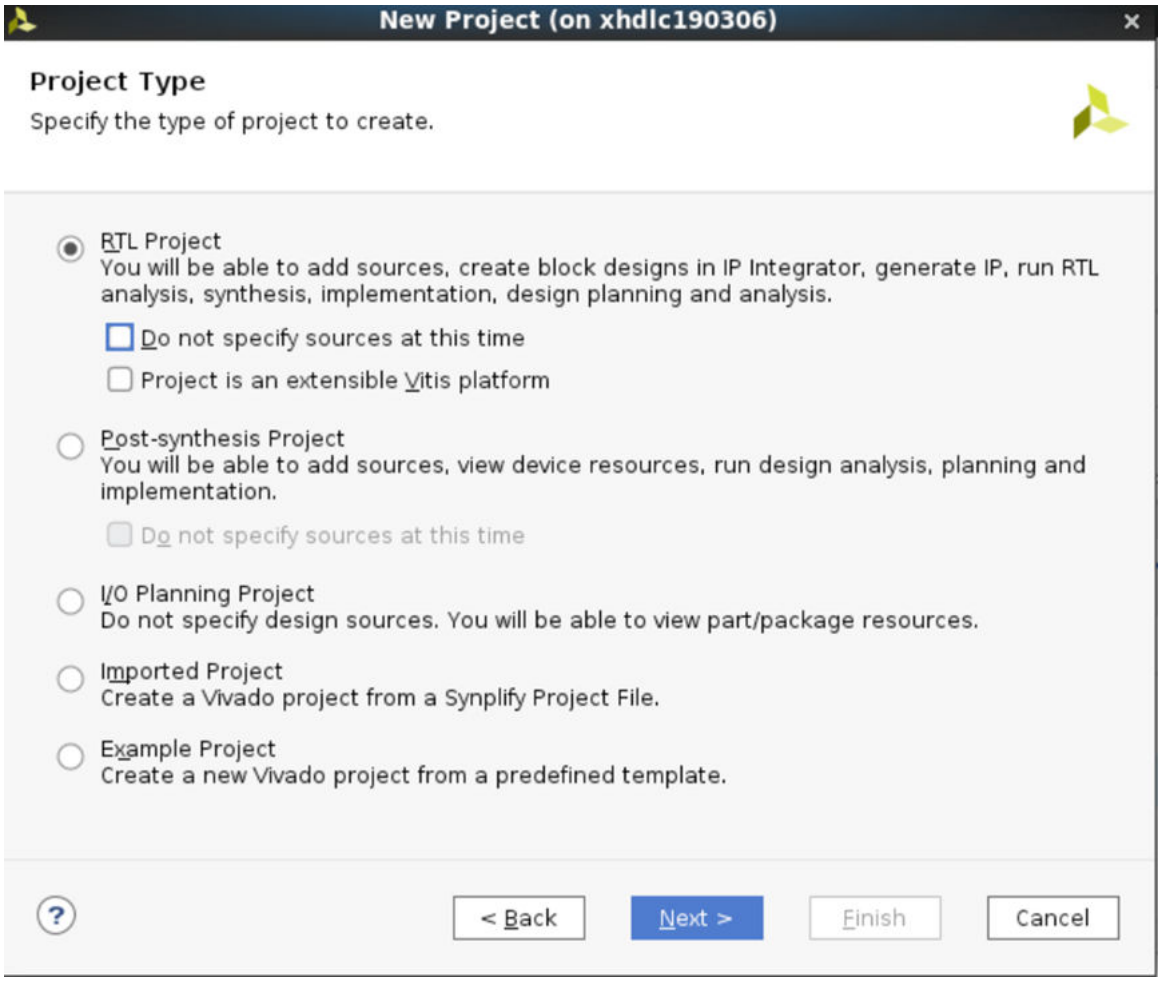
- Detailed descriptions for menu and toolbar commands appear on the lower left side of the status bar when you access the command.
- During placement and constraint creation in the Device and Package windows, constraint type and validity appear on the left side of the status bar, and site coordinates and type display on the right side.
- Object details appear in the status bar when you hover over an object in the Schematic window.
- The task progress bar appears on the right side of the status bar when you select the Background button on a running task.

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.*

Creating Projects

You can use the New Project wizard to easily create different types of projects in the Vivado IDE. To open the New Project wizard, select **File → Project → New**. This wizard enables you to specify a project location and name and create the types of projects shown in the following figure. 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 this link in the *Vivado Design Suite User Guide: System-Level Design Entry* (UG895).

Figure 7: **New Project Wizard—Project Type Page**



In the New Project wizard, selecting the Project is an extensible Vitis platform check box designates an RTL or Example design as an extensible platform. In Tcl this is achieved using the following command:

```
set_property platform.extensible (true|false) [get_project]
```

Creating Projects Using Tcl Commands

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.

TIP: *You can use the PATH_MODE property with the `add_files` Tcl command to specify whether to use absolute or relative paths. By default, relative paths are used. For more information, see this link in the Vivado Design Suite Properties Reference Guide (UG912).*

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

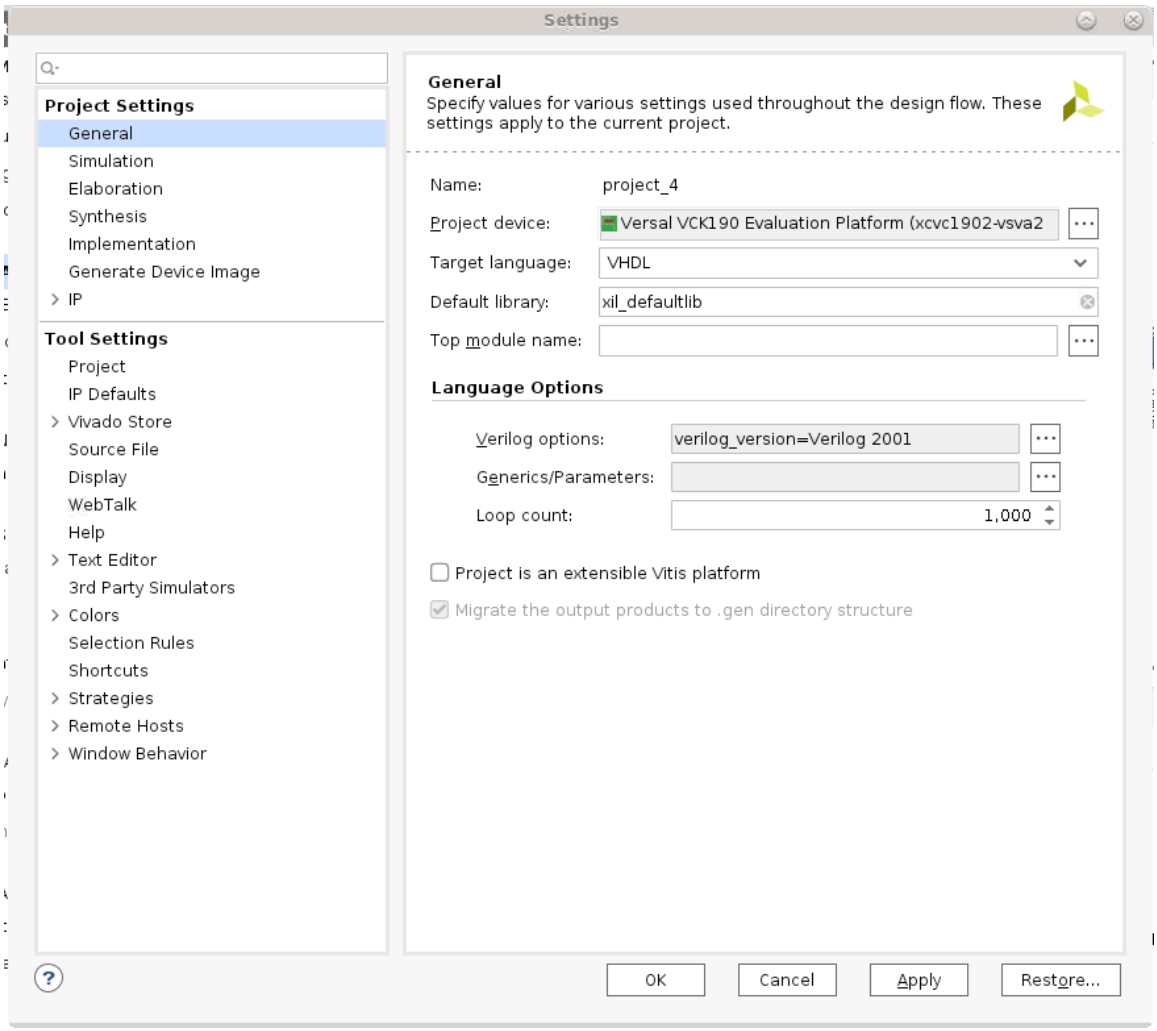
- *Vivado Design Suite User Guide: Design Flows Overview* (UG892)
- *Vivado Design Suite User Guide: Using Tcl Scripting* (UG894)

Configuring Project Settings

You can configure the Project Settings in the Settings dialog box (shown in the following figure) to meet your design needs. These settings include general settings related to the top module definition and language options, as well as simulation, elaboration, synthesis, implementation, bitstream, and IP settings.

TIP: *You can also access the Tool Settings from the Settings dialog box. For more information, see Specifying Tool Settings.*

Figure 8: **Settings Dialog Box—Project Settings General Category**



To open the Settings dialog box, use any of the following methods:

- In the Flow Navigator Project Manager section, click **Settings**.
- Select **Tools → Settings**.
- In the main toolbar, click the **Settings** toolbar button .
- In the Project Summary, click **Edit** next to Settings.

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

- **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 this link in the *Vivado Design Suite User Guide: System-Level Design Entry* (UG895).

Unchecking the box Project is an extensible Vitis platform will get the standard flow from previous versions of Vivado.

If you check the box, you will get a radically changed flow when you try to export the platform **Export → Export Platform**.

For more information, see link.

Migrate the output products to .gen dir structure will migrate the projects created in Version 2020.1 or older to the new directory structure, where sources and generated output products reside separately in `<prjoect_name>.srcs` and `<prjoect_name>.gen` directories, respectively.

- **Simulation:** Enables you to specify the target simulator, including the Vivado simulator and supported third-party simulators. Displays the simulation set, the simulation top module name, top module (design under test), and a tabbed listing of compilation, elaboration, simulation, netlist, and advanced options. For more information, see this link in the *Vivado Design Suite User Guide: Logic Simulation* (UG900).
- **Elaboration:** Enables you to specify whether to load the Netlist model or the Black box model (stub file) for IP in the elaborated design. Selecting the Netlist model provides more details about the IP and allows you to perform I/O planning of IP. Selecting the Black box model (stub file) enables faster loading. With either option, you must generate the IP first. However, the black box model does not load the IP checkpoint, which saves time. Selecting the Load constraints option includes constraints in the design. The Vivado IDE only displays these constraints if you select the Netlist model for the IP. For more information, see this link in the *Vivado Design Suite User Guide: System-Level Design Entry* (UG895).
- **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 this link in the *Vivado Design Suite User Guide: Synthesis* (UG901).
- **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 this link in the *Vivado Design Suite User Guide: Implementation* (UG904).
- **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 this link in the *Vivado Design Suite User Guide: Programming and Debugging* (UG908).

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 settings for the Vivado IP packager, IP caching, core containers, Resource Estimation for IP & BD and simulation scripts. For more information, see this link in the *Vivado Design Suite User Guide: Designing with IP* (UG896).

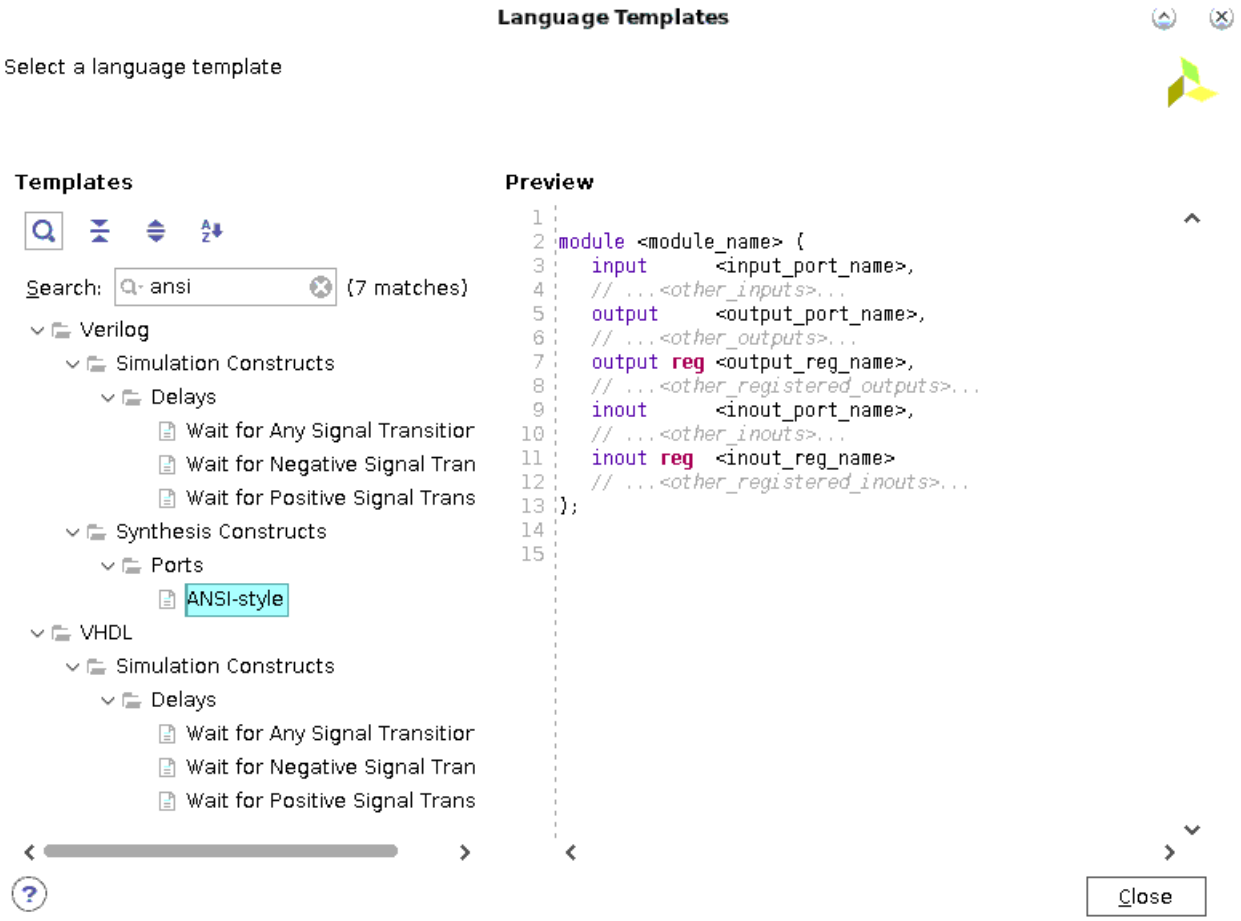
Related Information

Specifying Tool Settings
Editing Device Properties

Using Language Templates

The Language Templates (shown in the following figure) provide access to various 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 **Tools → Language Templates**, select **Language Templates** from the Project Manager section of the Flow Navigator, or click the **Language Templates** toolbar button in the Text Editor .

Figure 9: **Language Templates**



Related Information

Using the Text Editor

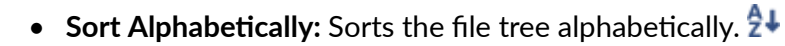
Language Templates Toolbar Commands

The local toolbar contains the following commands:

- **Search:** Opens the search bar to allow you to quickly locate objects in the Language Templates.

  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.

- **Sort Alphabetically:** Sorts the file tree alphabetically. 

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)
- *Vivado Design Suite User Guide: System-Level Design Entry* (UG895)
- *Vivado Design Suite User Guide: Synthesis* (UG901)
- *Vivado Design Suite User Guide: Implementation* (UG904)

Opening Designs

Use the Flow Navigator or Flow menu to select the following commands:

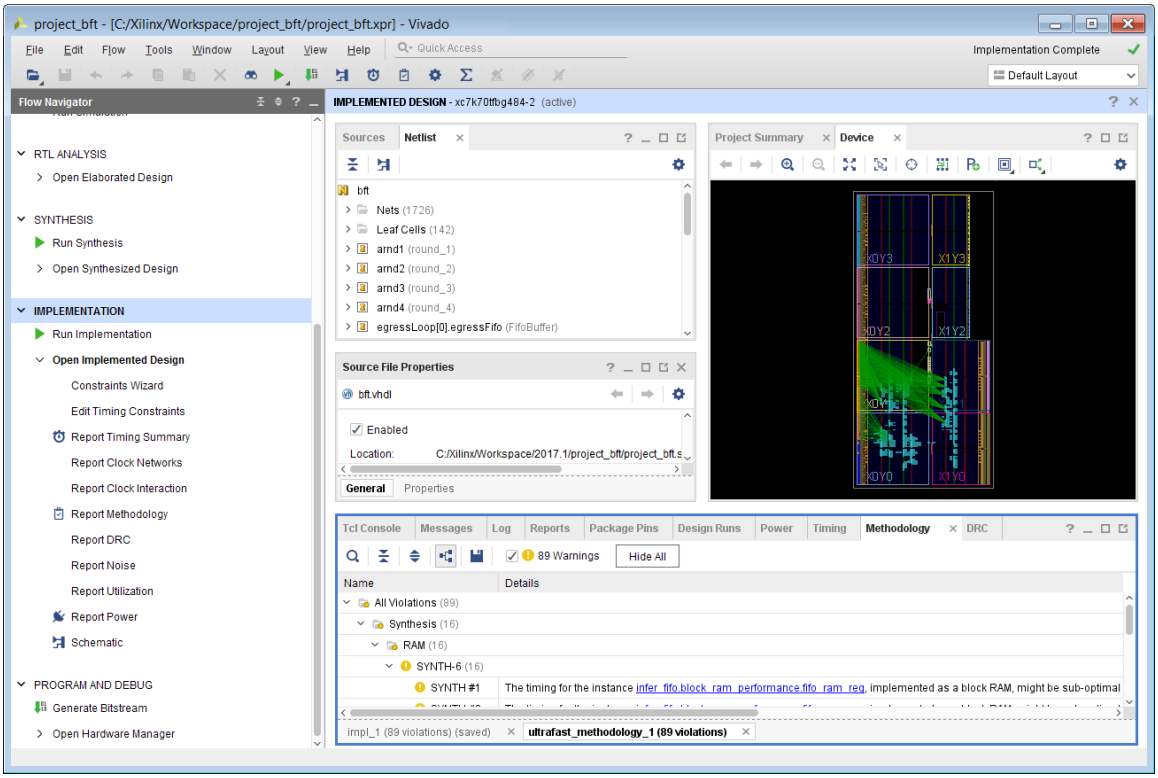
- Open Elaborated Design
- Open Synthesized Design
- Open Implemented Design

TIP: *In the Design Runs window, you can right-click a design run, and select* ***Open Run*** *to open the design.*

The **Flow → Open Implemented Design** command populates the Vivado IDE as shown in the following figure.

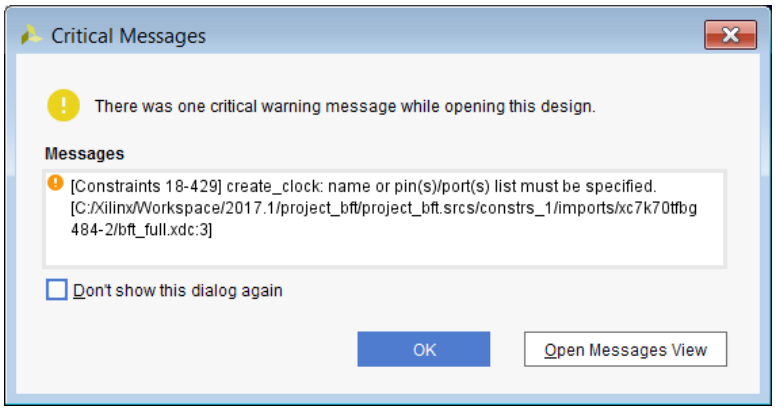
***Note*:** When you open an implemented design, the Vivado IDE opens the timing summary and power report created with the implementation run. You cannot modify these reports. If you make design changes, update constraints, or want to modify the report settings, select **Reports → Timing → Report Timing Summary** or **Reports → Report Power** to create a new, configurable report. To open an existing report, select **Reports → Open Interactive Report** or use the open_report Tcl command.

Figure 10: **Implemented Design**



Critical warnings and errors are displayed in a popup dialog box (see the following figure) 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 11: **Critical Messages**



For more information, see the following documents:

- *Vivado Design Suite User Guide: Design Flows Overview* (UG892)
- *Vivado Design Suite User Guide: Design Analysis and Closure Techniques* (UG906)

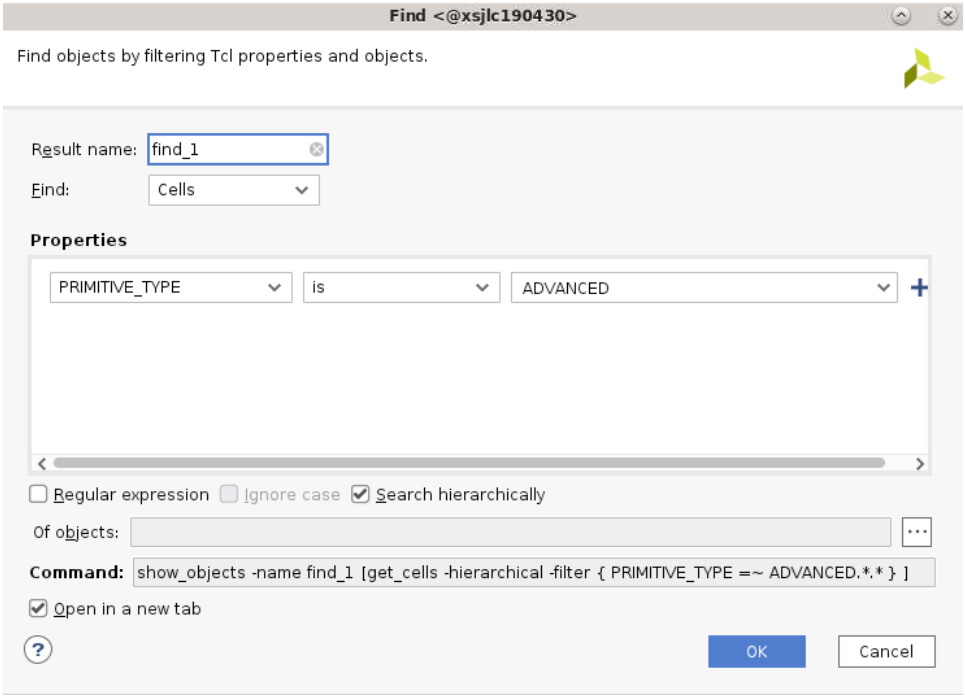
Finding Design or Device Objects

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 (shown in the following figure), 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.
- **Find:** Filters the type of object to search.
- **Properties:** Specifies the Tcl properties used to find the design or device objects. Click the add ✚ button to add properties. Click the remove ▬ 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 execute the search.

- **Open in a new tab:** Opens a new Find Results window instead of replacing the previous results.

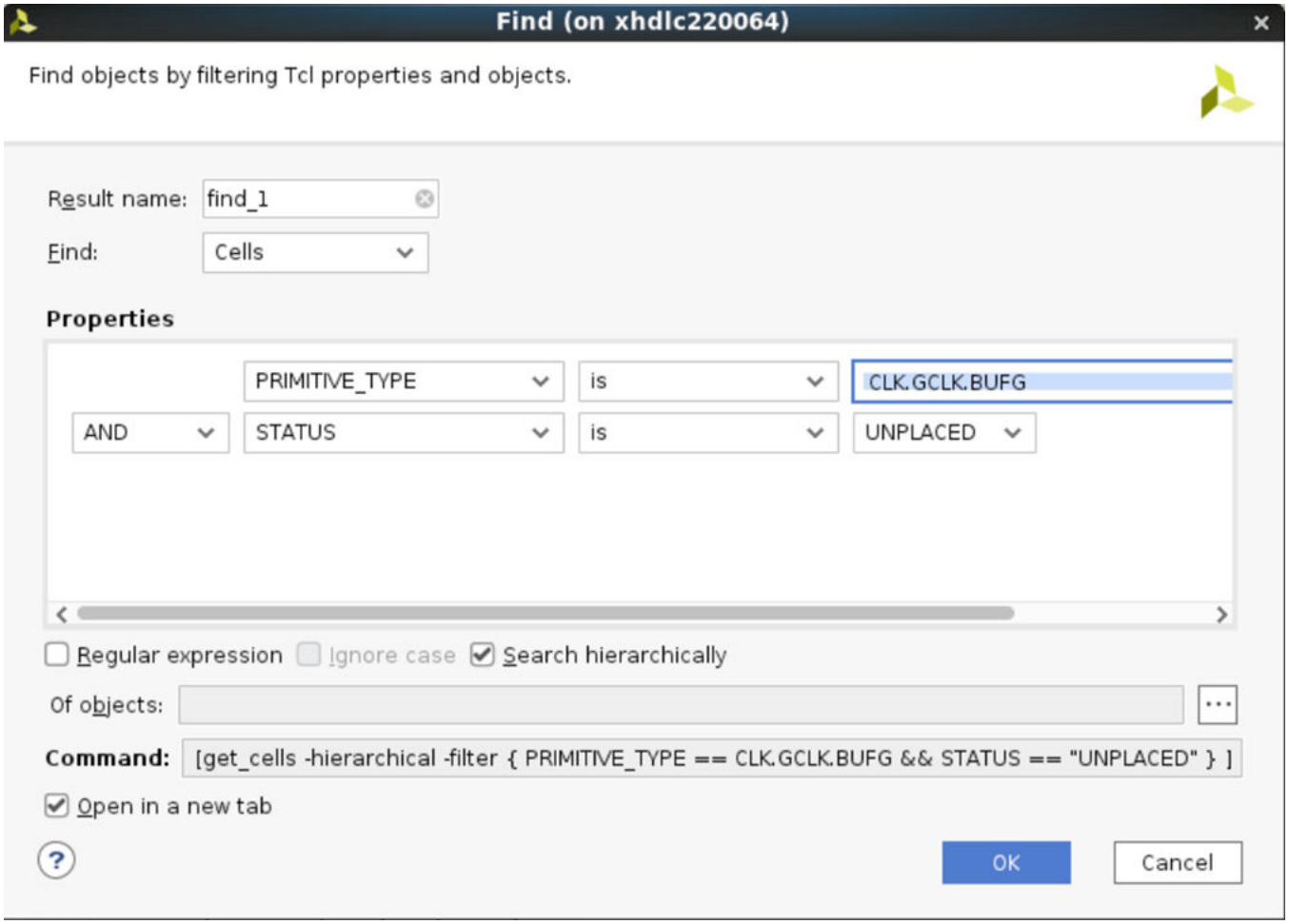
Figure 12: **Find Dialog Box**



Finding Unplaced BUFGs

The following figure shows the Find dialog box settings for finding unplaced BUFGs.

Figure 13: **Finding Unplaced BUFGs**



Tcl Command Example for Finding Unplaced BUFGs

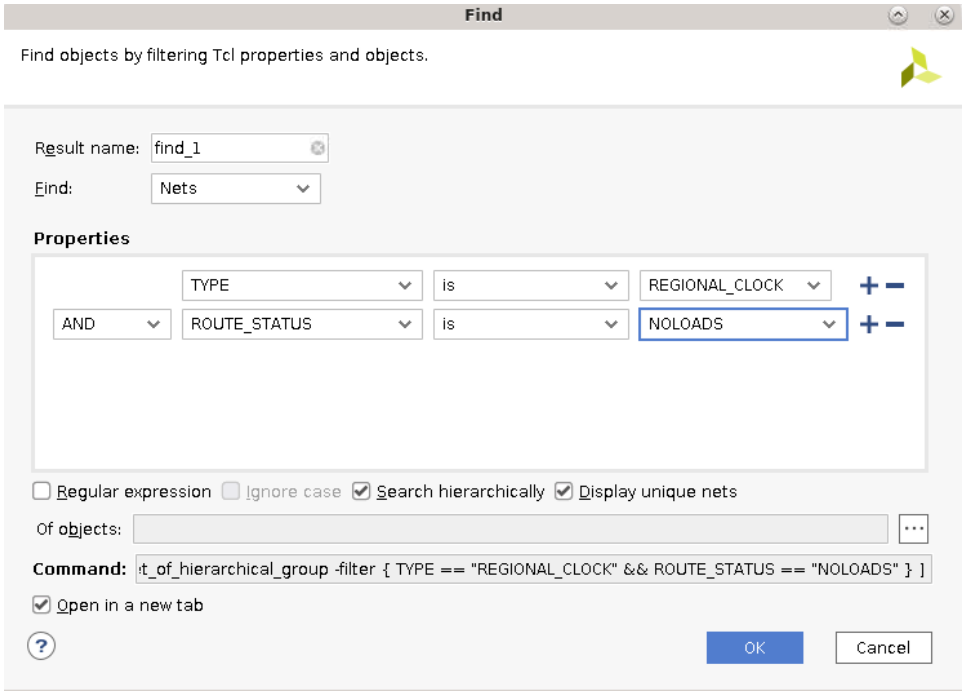
```
show_objects -name unplaced_BUFGs [get_cells -hierarchical -filter
{ PRIMITIVE_TYPE == CLK.gclk.BUFG && STATUS == "UNPLACED" }]
```

***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).

Finding Regional Clocks with No Loads

The following figure shows the Find dialog box settings for finding regional clocks with no loads.

Figure 14: **Finding Regional Clocks with No Loads**



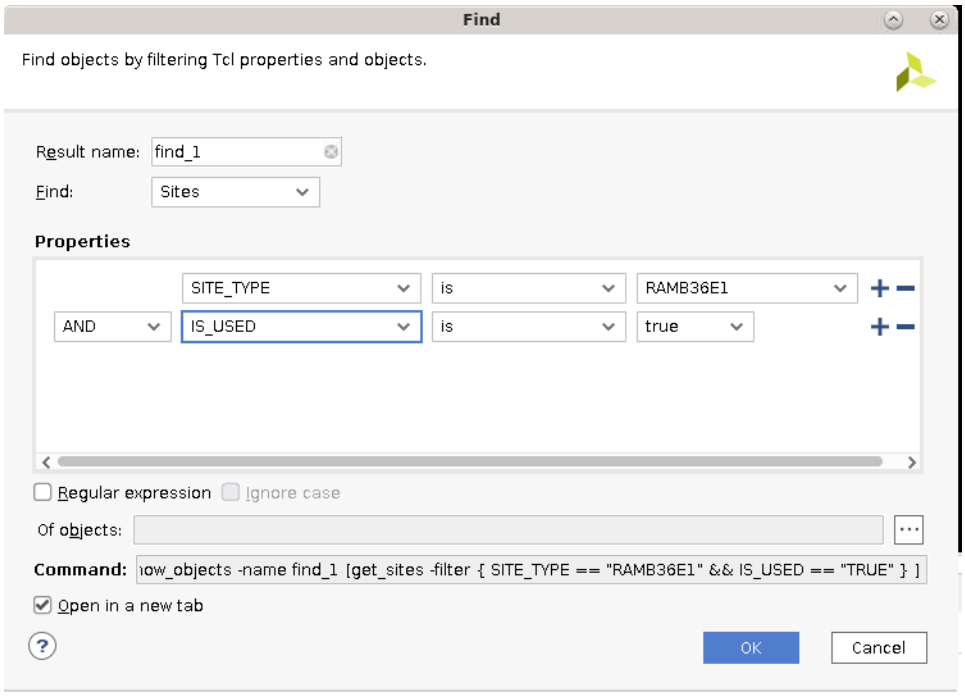
Tcl Command Example for Finding Regional Clocks with No Loads

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

Finding Placed RAMB36 Cells

The following figure shows the Find dialog box settings for finding RAMB36 sites that contain a block RAM cell.

Figure 15: **Finding Placed RAMB36 Cells**



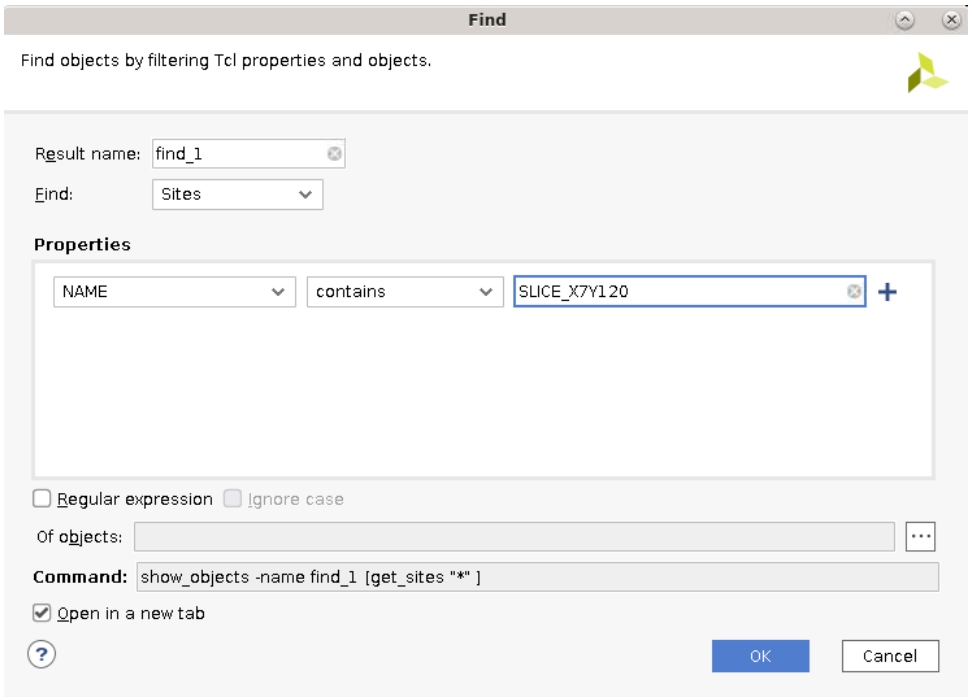
Tcl Command Example for Finding Placed RAMB36 Cells

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

Finding Objects

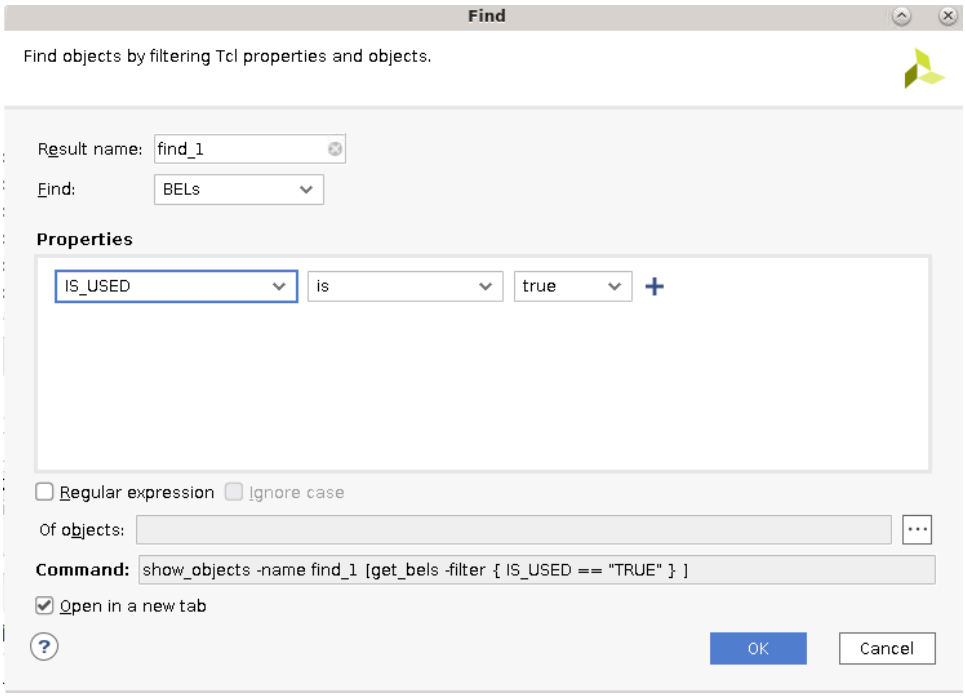
In the Find dialog box, you can click the Of Objects button (...) to open the Of Objects dialog box, which enables you to specify a particular object to search. The following figure shows the Of Objects dialog box settings for a search for a specific slice.

Figure 16: **Finding an Object**



After specifying the slice, you can search for the occupied BELs, as shown in the following figure.

Figure 17: **Finding Occupied BELs within a Slice**



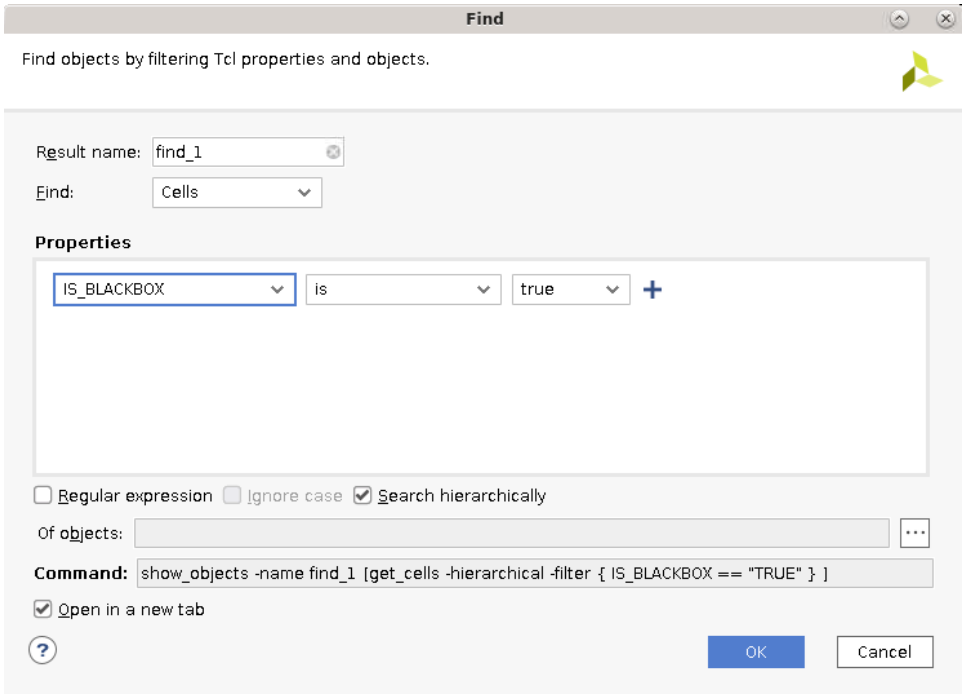
Tcl Command for Finding Objects

```
show_objects -name BELs_Slice_X7Y120 [get_bels -filter { IS_USED ==
"TRUE" }
-of_objects [get_sites -filter { NAME =~  "SLICE_X7Y120" } ]]
```

Finding Black Box Cells

The following figure shows the Find dialog box settings for finding black box cells.

Figure 18: **Finding Black Box Cells**



Tcl Command for Finding Black Box Cells

```
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. To edit device properties, including programming and configuration properties, use the Edit Device Properties dialog box. To update properties for multiple objects, use the Property Editor.

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

Related Information

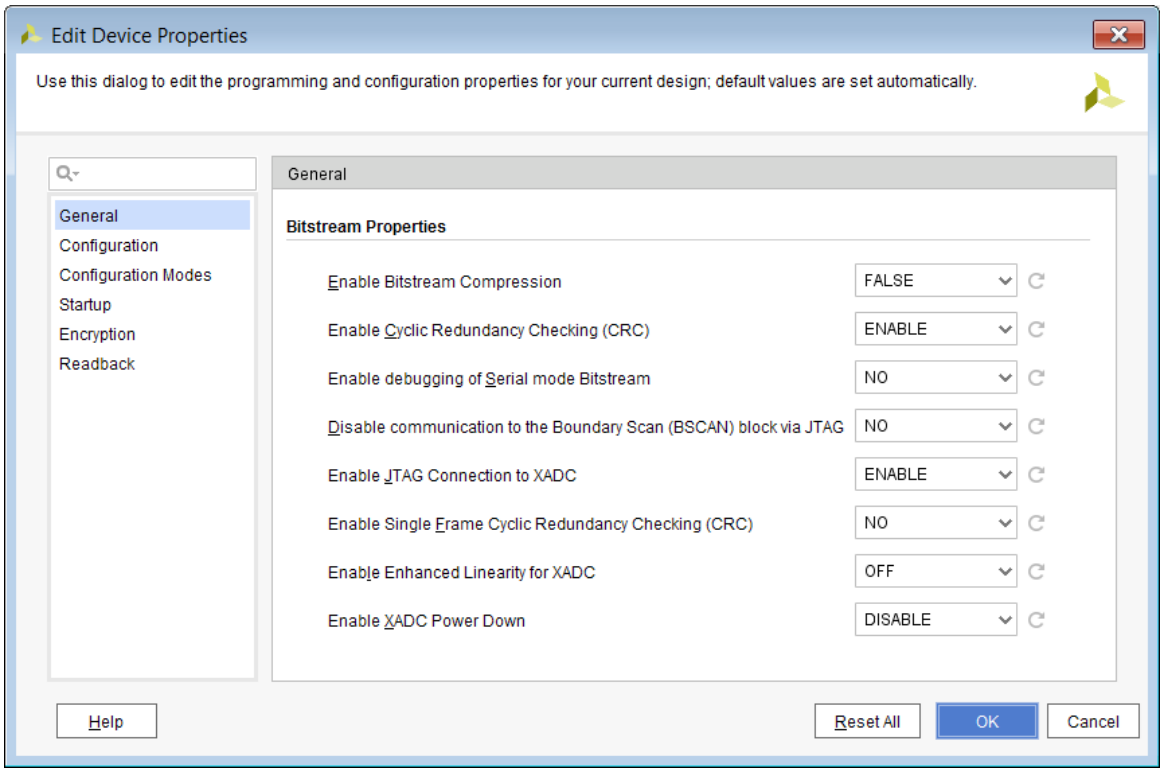
Using the Properties Window

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 (shown in the following figure), 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 this link in the *Vivado Design Suite User Guide: Programming and Debugging* (UG908). For information on setting device configuration modes, see this link in the *Vivado Design Suite User Guide: I/O and Clock Planning* (UG899).

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

Figure 19: **Edit Device Properties Dialog Box**



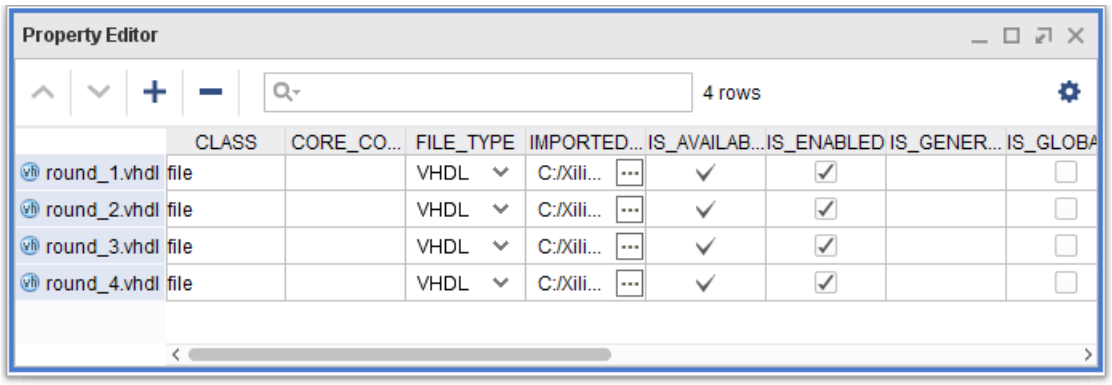
Tcl Command for Enabling Bitstream Compression

```
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 (shown in the following figure) and edit properties for the selected objects.

Figure 20: **Property Editor**

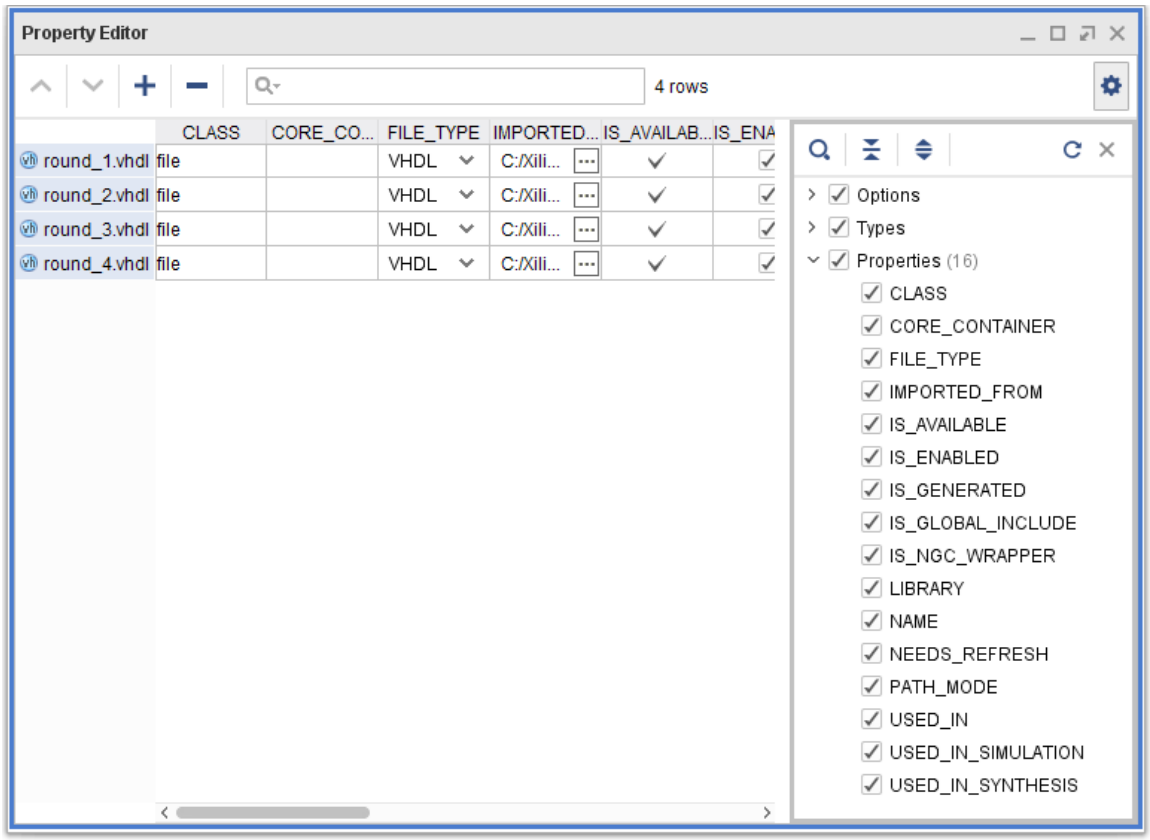


In the Property Editor, you can also do the following:

- To adjust the Property Editor display, click the **Settings** toolbar button. Edit the options that control the display of the header, types of objects, and properties (as shown in the following figure).
- To filter the displayed data, right-click in the Property Options, and select a filter command. For example, select **Show Columns with Differences** to filter out columns in which all of the data is the same.
- 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, select the objects and click the **Add Selected Objects** toolbar button.

TIP: *Editable text strings are indicated by a pencil icon.*

Figure 21: **Property Editor Options**

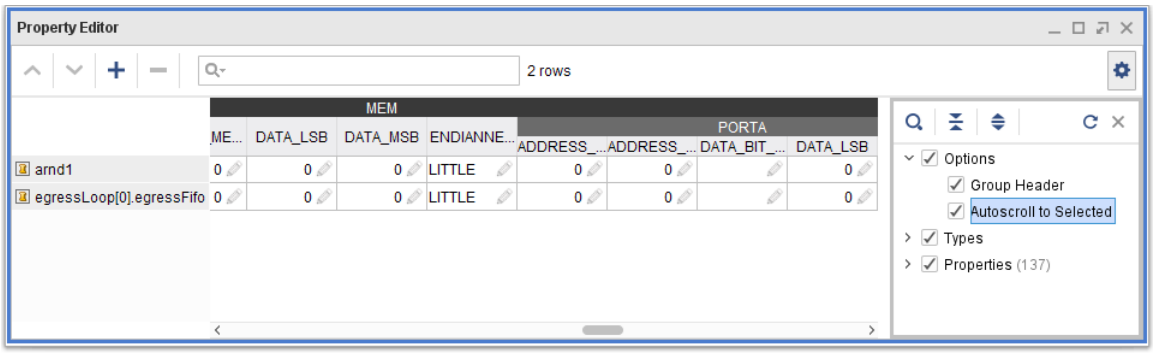


Property Editor Toolbar Commands

The local toolbar contains the following commands:

- **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.
- **Remove Selected Objects:** Removes the selected object from the list.
- **Settings:** Controls the following settings that affect the display of information.
  - Options
    - **Group Header:** Groups headers for related properties, such as MEM properties (shown in the following figure).
    - **Autoscroll to Selected:** Scrolls the list of objects in the Property Editor to show the objects selected in other windows, such as the Sources or Netlist windows.
- **Types:** Shows or hides each property type.
- **Properties:** Shows or hides each property column.

Figure 22: **Property Editor with Group Header Option**



Using Windows

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

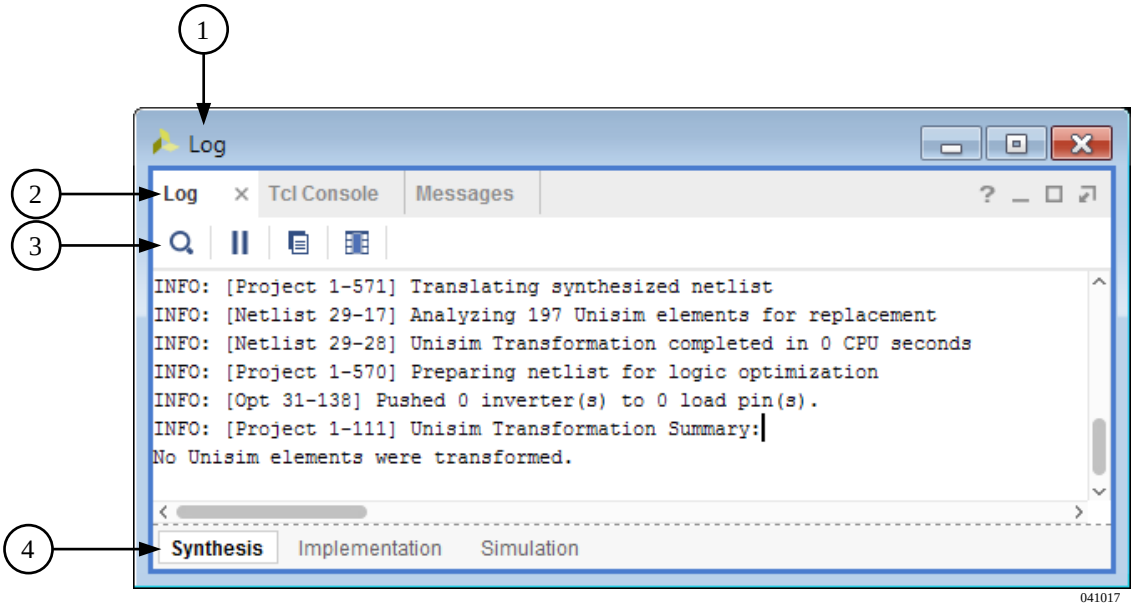
Note: For more information on these features, see the *Vivado Design Suite User Guide: Design Flows Overview* (UG892) and *Vivado Design Suite User Guide: Design Analysis and Closure Techniques* (UG906).

Working with Windows

The following figure shows the parts of a window:

- Title bar
- Window tabs
- Local toolbar
- Window views

Figure 23: **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 top of most windows, such as the Log, Tcl Console, and Messages windows.

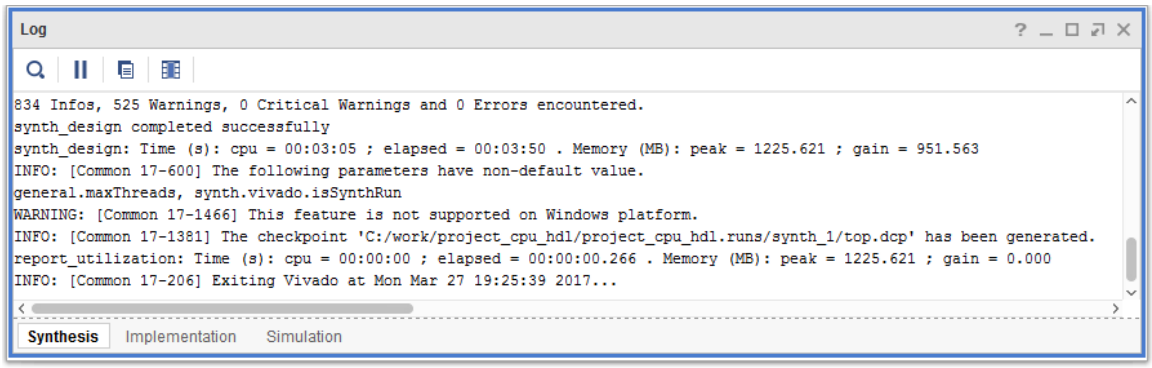
TIP: *To make the next tab active in the workspace, press **Ctrl+Tab**. To make the previous tab active in the workspace, press **Ctrl+Shift+Tab**. To maximize or minimize the window, double-click the window tab, or press **Alt -**.*

Window Views

Some windows include different views of the same data. For example, the Log window (shown in the following figure) includes views for Synthesis, Implementation, and Simulation.

TIP: *When the number of views is greater than the space available, you can use the left and right arrows that appear on the right side of the window to scroll the tabs. Alternatively, you can hover over the tabs and use the scroll wheel on your mouse to scroll the tabs.*

Figure 24: **Log Window with Multiple Views**

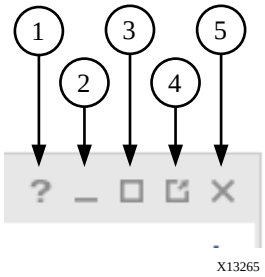


Window Controls

Each window has the following window controls, which enable you to manipulate the window (shown in the following figure):

- Quick Help
- Minimize
- Maximize
- Float/Dock
- Close

Figure 25: **Window 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 Configuring Custom View Layouts.*

Related Information

Configuring Custom View Layouts
Using the Workspace

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.

***Note*:** 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 tab or title bar, 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.
- Right-click the window tab or title bar, and 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.

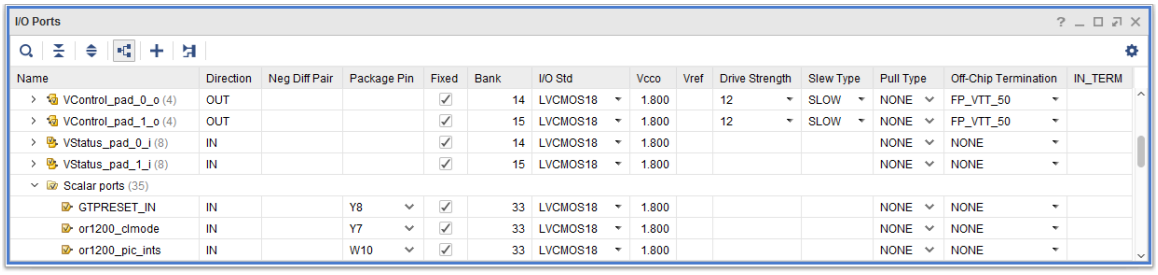
- Right-click the window tab or title bar, and select Close from the popup menu.

> **TIP:** *You can also press the **Esc** key to close the window.*

Using Data Table Windows

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

Figure 26: **Data Table Window**



Expanding and Collapsing the Table

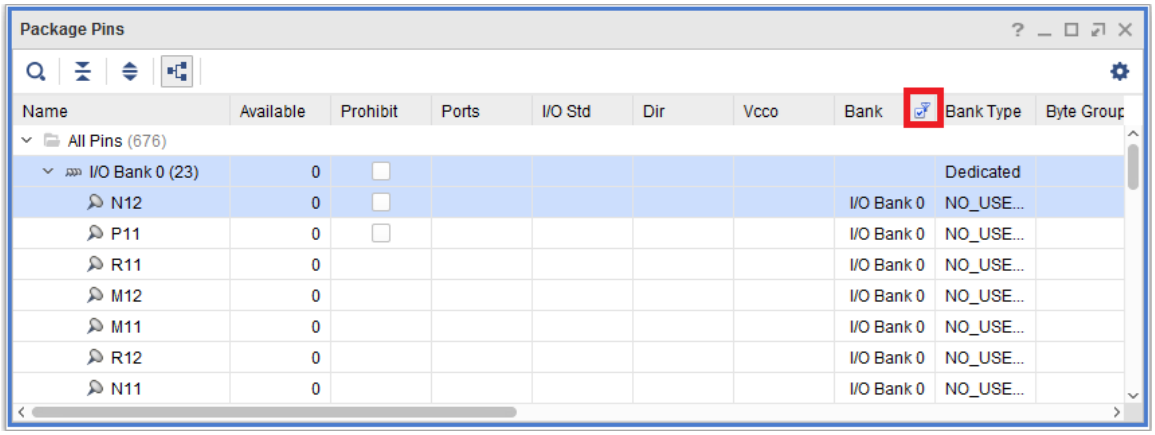
To expand or collapse the table:

- Use 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.

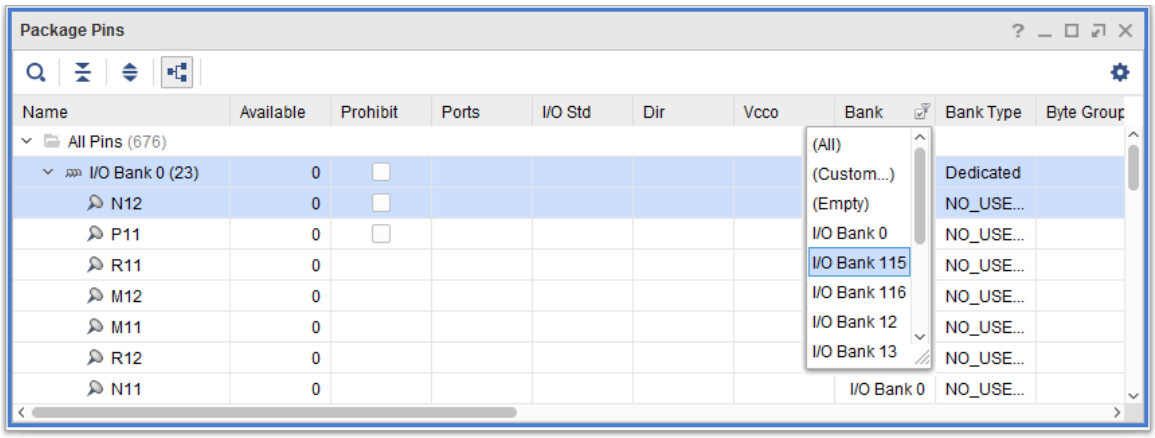
Filtering Table Data

To filter table data to display only specified values:

1. Right-click a column header, and select **Enable Column Filtering**.
2. Hover over the column to filter, and click the Filter Column icon as shown in the following figure.



3. Select the column value to display as shown in the following figure.



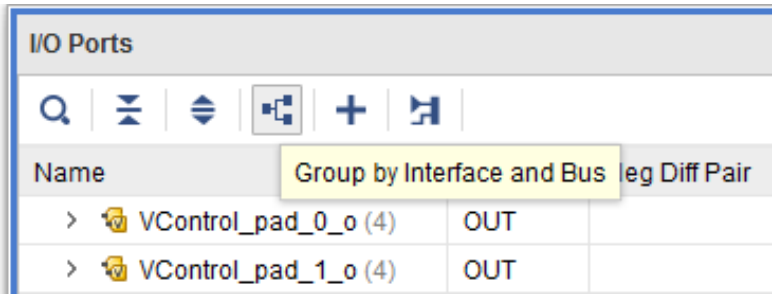
The Filtered Column icon appears at the top of each column that you filtered, and the table shows only the rows of data that include the selected value.

TIP: *To clear a filter, right-click the Filtered Column icon, and select (**All**). To clear all filters but leave the filter feature enabled, select **Clear All Column Filters**.*

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 (see the following figure).

Figure 27: **Group by Type or Flat List Toolbar Button**



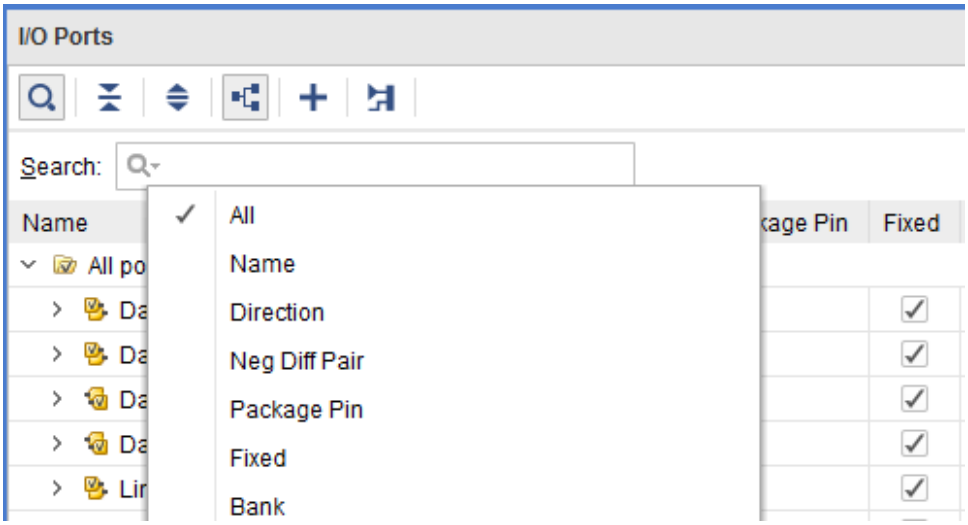
Using the Search Capability to Filter the List

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

1. In the local toolbar, click the 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 drop-down menu on the left of the search field, and select search criteria, including which columns to search (see the following figure).



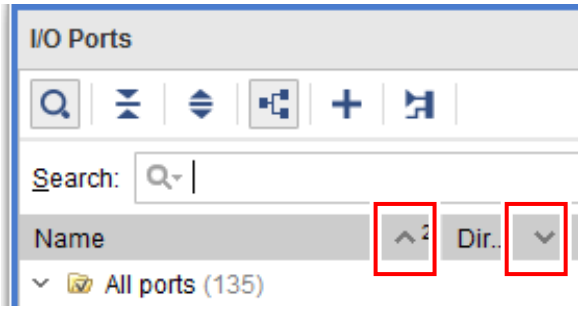
3. Enter a text string to filter the list displayed in the table window.

   When you enter a text string, the list adjusts dynamically to list only those entries that contain the string. Click the 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 the following figure.

Figure 28: **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 the previous figure, 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

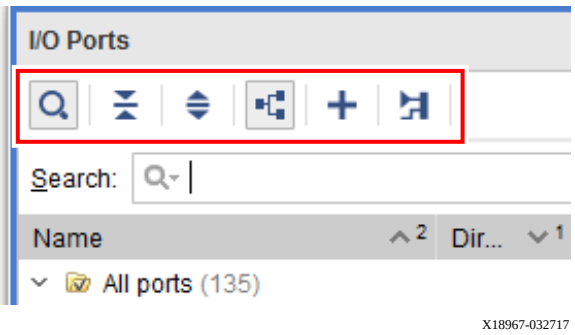
To organize columns:

- 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 (see the following figure). 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 29: **Local Toolbar**



Using the Project Summary

The Vivado IDE includes an interactive Project Summary (shown in the following figure) that updates dynamically as design commands are run and the design progresses through the design flow. The Project Summary includes the Overview tab and a user-configurable Dashboard, as described in the following sections.

To open the Project Summary, do either of the following:

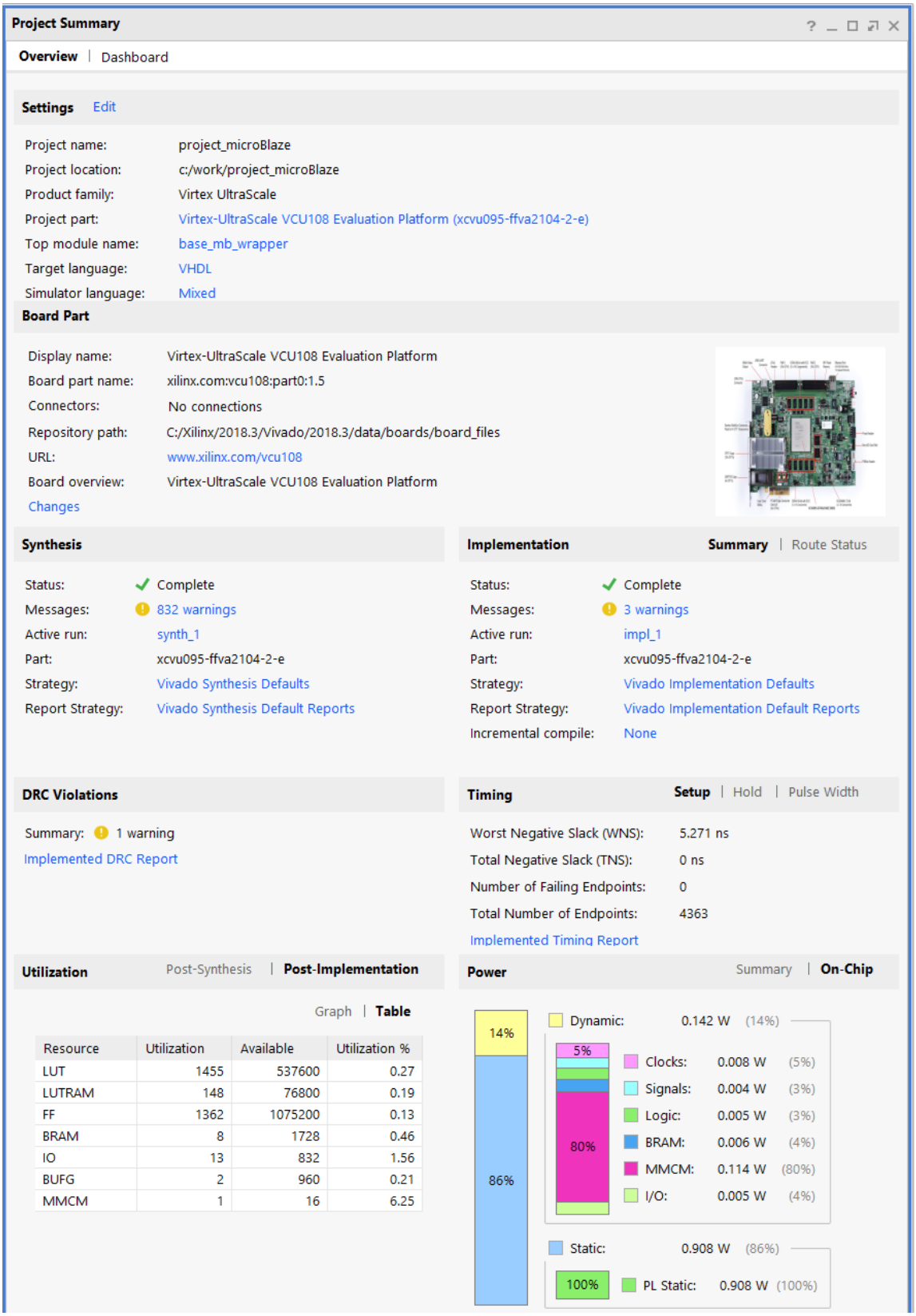
- Select **Window → Project Summary**.
- Click the **Project Summary** toolbar button Σ.

***Note*:** The Overview tab in the Project Summary appears by default.

Using the Project Summary Overview Tab

The Overview tab (shown in the following figure) 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 and Reports windows as well as the Settings dialog box. As synthesis and implementation complete, DRC violations, timing values, utilization percentages, and power estimates are also populated.

Figure 30: **Project Summary Overview Tab**



Using the Project Summary Dashboard

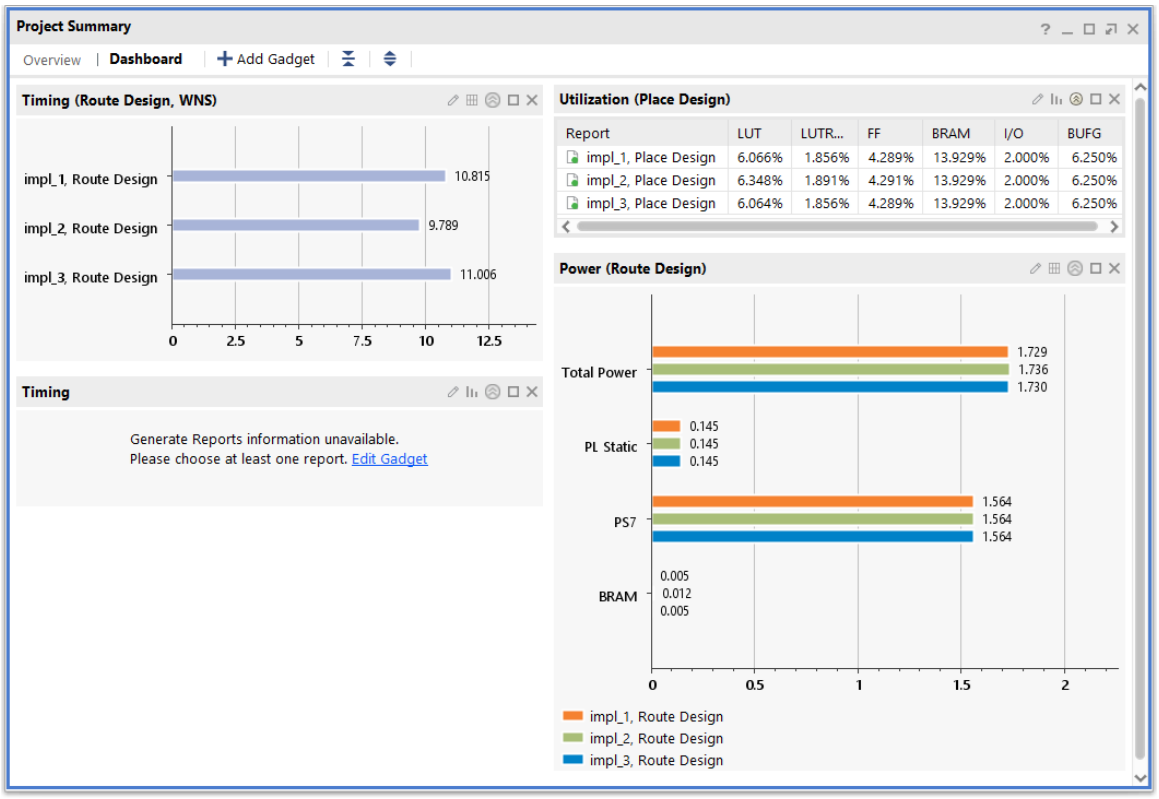
You can configure the Project Summary Dashboard (shown in the following figure) to view and analyze data as follows:

- Show various data in either tabular or graphical form
- Compare values across multiple runs
- Create a gadget that shows various data points for a single or multiple runs

  ***Note*:** The data for the gadget is gathered from the reports associated with the runs. To create a gadget, you must first set up reports for the runs. For more information on reports, see Using the Reports Window.

TIP: *When you request data from a report that was reset or is not part of the run, the gadget displays a message stating the information is unavailable. To resolve this issue, verify that the report type is part of the Report Strategy. Then, add the report type, change the report type in the gadget, or launch the run to generate the report as needed.*

Figure 31: **Project Summary Dashboard**



Related Information

Using the Reports Window

Using Gadgets

You can click the **Add Gadget** button + to open the Configure Gadget dialog box (shown in the following figure), which allows you to create a gadget that shows customized data for your runs. In the Configure Gadget dialog box, set the following options, and click **OK** to add the gadget to your Dashboard.

Note: You can also open the Configure Gadget dialog box using the Edit button in the gadget header.

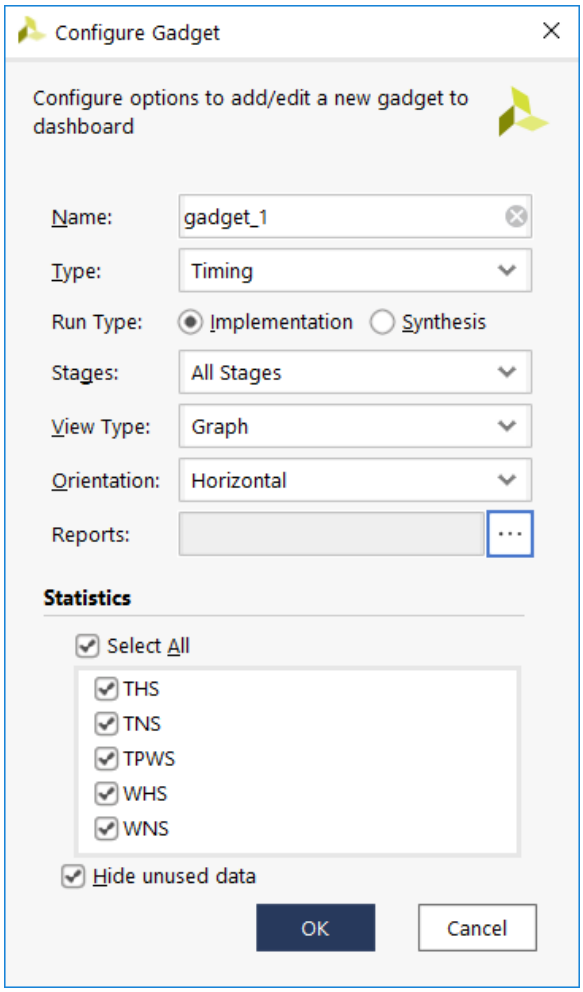
- **Name:** Specify a name to identify the gadget for use in running Tcl commands.
- **Type:** Select the report type to use to generate the gadget data (for example, Timing).
- **Run Type:** Select either the Synthesis or Implementation run.
- **Stages:** If you are analyzing an Implementation run, select an implementation stage (for example, Place), or select **All Stages**. If you are analyzing a Synthesis run, only one stage is available.
- **View Type:** Set the gadget to display as a graph or as a table. You can change the display after adding the gadget using the Graph/Table toolbar button / in the gadget header.
- **Orientation:** Select a **Vertical** or **Horizontal** orientation for the graph.

  Note: This is not available for tables.
- **Reports:** Select reports from one or more runs to display related data for your gadget.
- **Statistics:** Select the statistics to display in the gadget. The available statistics are based on the selected reports.
- **Hide Unused Data:** Select this option to hide statistics entries that contain no data. To display all statistics, deselect this option.

To customize the gadget layout:

- To move a gadget, click and drag the gadget to any location in the Dashboard.
- To widen or narrow the gadget along with all gadgets in the selected column, click and drag the edge of the gadget.
- To view or hide data in the Dashboard, use the Maximize or Collapse toolbar button in the gadget header.

Figure 32: **Configure Gadget Dialog Box**



Tcl Command Example for Adding a Gadget

```
create_dashboard_gadget -name {gadget_1} -type timing
set_property view.type graph [get_dashboard_gadgets [list {gadget_1}]]
set_property reports {impl_1#impl_1_route_report_timing_summary_0}
[get_dashboard_gadgets [list {gadget_1}]]
set_property active_reports {impl_1#impl_1_route_report_timing_summary_0}
[get_dashboard_gadgets [list {gadget_1}]]
set_property run.step all_stages [get_dashboard_gadgets [list {gadget_1}]]
```

Using the Sources Window

The Sources window (shown in the following figure) 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

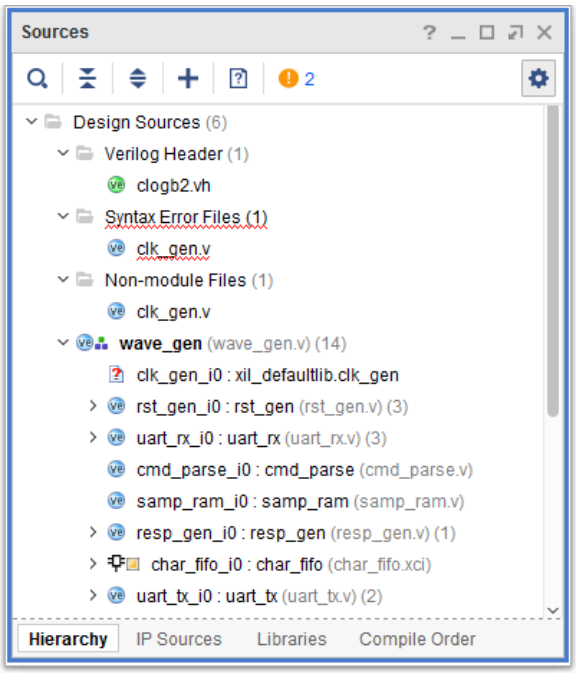
Generally, the Sources window is available in the Vivado IDE whenever a project is open. To open the Sources window, select **Window → Sources**. The Sources window includes the following folders:

- **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.
  - **Syntax Error Files:** Displays files with syntax errors that affect the design hierarchy.
  - **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.

    ***Note*:** NGC format files are not supported in the Vivado Design Suite for UltraScale™ devices. Xilinx recommends that you regenerate the IP using the Vivado Design Suite IP customization tools with native output products. Alternatively, you can use the NGC2EDIF command to migrate the NGC file to EDIF format for importing, as described in this link in the *ISE to Vivado Design Suite Migration Guide* (UG911). However, Xilinx recommends using native Vivado IP rather than XST-generated NGC format files going forward.
- **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) and *Vivado Design Suite User Guide: Using Constraints* (UG903).
- **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).

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 33: **Sources Window**



Sources Window Views

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

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

Related Information

Sources Window Popup Menu Commands

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 Commands.

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 
- 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 Show only missing sources 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 this link in the *Vivado Design Suite User Guide: Designing with IP* (UG896).

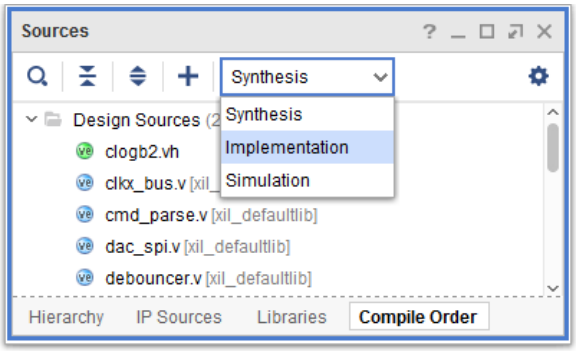
Libraries View

The Libraries view displays the sources sorted into the various libraries. You can use this view to create new libraries and manage files.

Compile Order View

The Compile Order view displays source files in the order in the files will be compiled, from first to last, and shows the processing order for constraints. At the top of the Compile Order view, you can select **Synthesis**, **Implementation**, or **Simulation** from the drop-down menu to show the source files for each design flow step, as shown in the following figure.

Figure 34: **Compile Order View Drop-Down Menu**



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 Commands.

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
```

Related Information

Sources Window Popup Menu Commands

Sources Window Commands

To add, view, or modify source files, use the Sources window local toolbar or popup menu commands.

TIP: *To display the popup menu, right-click in the window.*

Sources Window Toolbar Commands

The local toolbar contains the following commands:

- **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.
- **Add Sources:** Adds or creates constraint files, simulation source files, and design sources. Design sources include HDL and netlist files as well as existing IP and block designs.
- **Show Only Missing Sources:** 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.*

- **Messages:** Shows a summary of any messages generated during the design run. For more information, see Using the Messages Window. If there are no messages associated with your source files, the icon is disabled.
- **Settings:** Controls the display of information in the window.
  - **Scroll to Selected Objects:** 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.

Related Information

Using the Messages Window

Sources Window Popup Menu Commands

The popup menu contains the following commands:

- **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 source files as follows:
  - RTL source files or constraint files open in the Text Editor
  - IP cores open in the Customize IP dialog box
  - BD files open in the IP integrator
- **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.

  ***Note*:** Disabling the file removes the file from the compile list and hierarchy but does not remove the file from the project.

- **Move to Simulation Sources:** Relocates currently selected design 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 Design Sources:** Relocates currently selected simulation source files into the design sources.

- **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! *The Move to Top, Move Up, Move Down, and Move to Bottom commands are only available from the Compile Order view. Alternatively, you can drag and drop files in the Compile Order view to change the compile order.*

- **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.

    ***Note*:** For imported ISE® Design Suite projects, this setting is selected by default to preserve the compile order. If you do not need to preserve the compile order, you can change this setting to Automatic Update and Compile Order.

  - **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 file to which Vivado IDE writes new constraints. For more information on design constraints, see this link in the *Vivado Design Suite User Guide: System-Level Design Entry* (UG895) and see the *Vivado Design Suite User Guide: Using Constraints* (UG903).
- **Set as Out-of-Context for Synthesis:** 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 this link in the *Vivado Design Suite User Guide: Synthesis* (UG901).

  Note: This option only works on levels of RTL hierarchy shown in the Sources window.

- **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 File 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.

- **Edit Constraint Sets:** Creates and modifies constraint sets.
- **Edit Simulation Sets:** Creates and modifies simulation sets.
- **Add Sources:** Adds or creates constraint files, simulation source files, and design sources. Design sources include HDL and netlist files as well as existing IP and block designs.
- **Go to Source:** Opens the source file in which the module or instance is defined.

Related Information

Viewing Source File Properties

Sources Window Popup Menu Commands for IP Sources

The following commands are available in the popup menu when an IP core is selected in the Sources window.

Note: For more information, see the *Vivado Design Suite User Guide: Designing with IP* (UG896).

- **Enable/Disable Core Container:** Toggles storing IP as a single file on disk.
- **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.
- **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 Records:** Searches the Xilinx® Support database for Answer Records associated with the IP.
- **Copy Shared Logic into Project:** Specifies a destination directory for shared logic files.
- **Report IP Status:** Opens an IP status report that displays the status, version, change log, part, and other information for each IP in the design.

Sources Window Popup Menu Commands for Block Design Sources

The following commands are available in the popup menu when a block design (BD) module is selected in the Sources window.

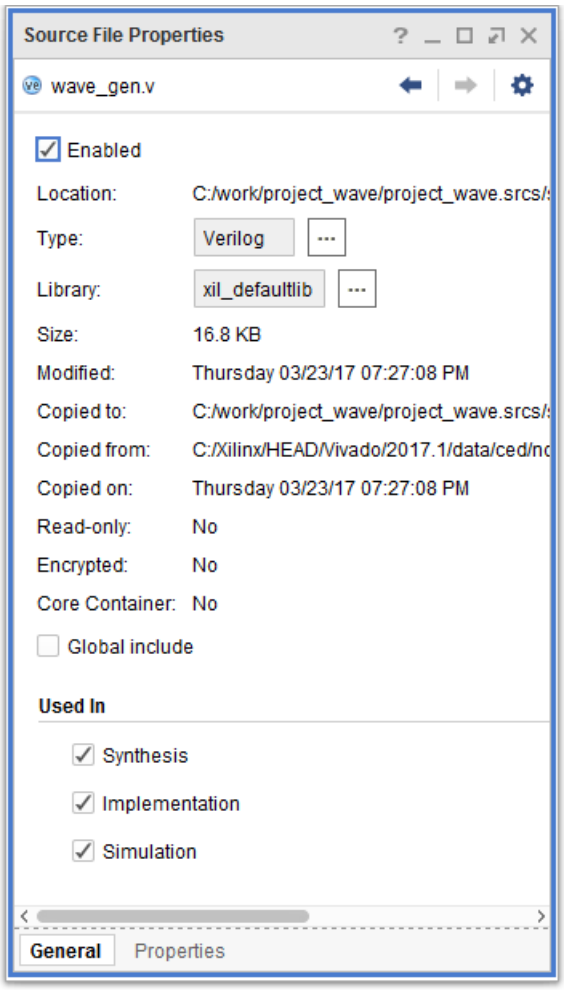
Note: For more information, see the *Vivado Design Suite User Guide: Designing IP Subsystems Using IP Integrator* (UG994).

- **Create HDL Wrapper:** Creates a top-level Verilog or VHDL module that contains the selected block design.
- **View Instantiation Template:** Opens the instantiation template for the block design to instantiate it into another RTL file.
- **Generate Output Products:** Generates target data for the block design as needed.
- **Reset Output Products:** Removes the currently generated target data.

Viewing Source File Properties

Selecting an RTL source file in the Sources window displays information in the Source File Properties window (see the following figure).

Figure 35: **Source File Properties Window**



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:

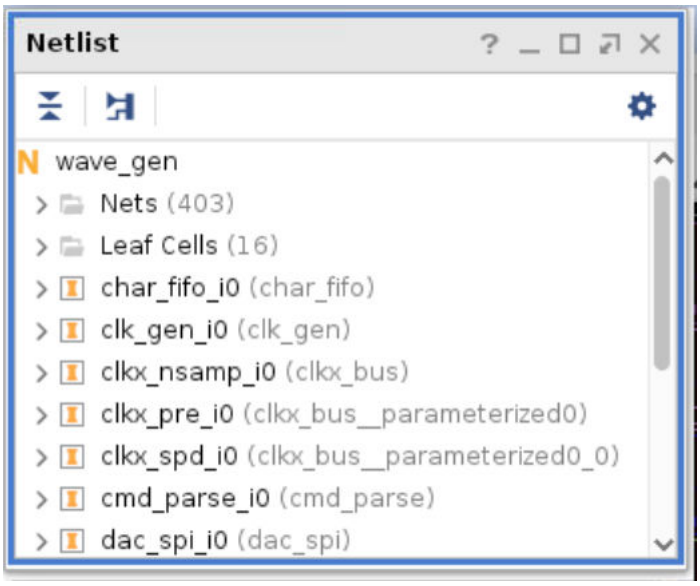
   - **Type:** Changes the file type. This is useful in cases where files have non-standard extensions, and the file type is not properly detected.
   - **Library:** Specifies a new target library for a source file. Select from the list of defined libraries, or type a library name.

- **Global Include:** Sets 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 Netlist Window

The Netlist window (shown in the following figure) 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. To open the Netlist window, select **Window → Netlist**.

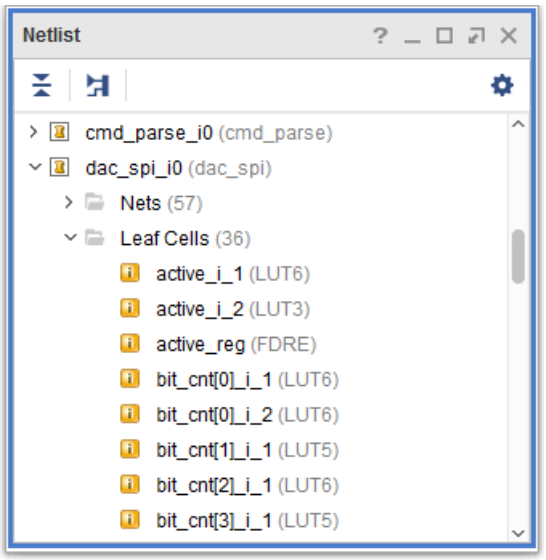
Figure 36: **Netlist Window**



The Netlist window includes the following folders:

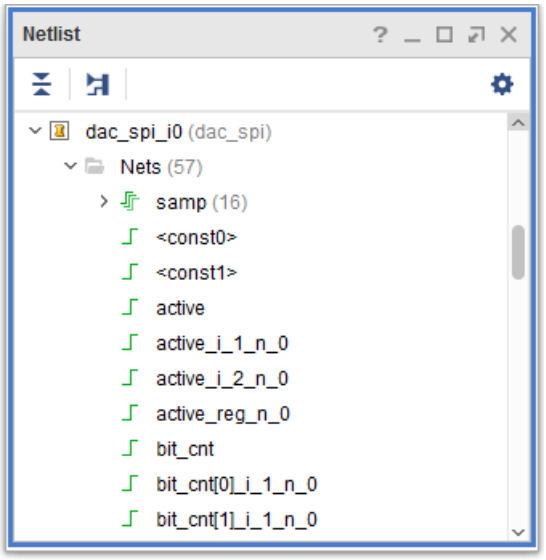
- **Leaf Cells:** Displays primitive logic for each level of the hierarchy. This folder condenses the display of logic content and hierarchical modules in the Netlist window (shown in the following figure).

Figure 37: **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 (shown in the following figure).

Figure 38: **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.

- Click the Collapse All toolbar button to collapse the entire tree.

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

***Note*:** The Netlist tree dynamically expands to display objects selected in other windows. To disable this feature, click the Settings toolbar button and deselect **Scroll to selected objects**.

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 mark nets for debug testing, right-click the net, and select the **Mark Debug** popup command. For more information, see this link in the *Vivado Design Suite User Guide: Programming and Debugging* (UG908).

***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.

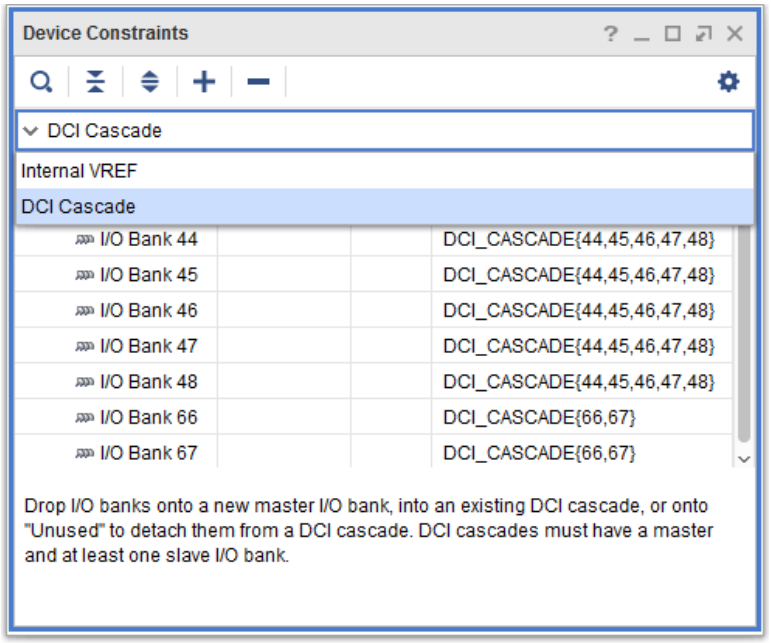
- Hierarchical cell (assigned to a Pblock)
- Hierarchical cell (black box assigned to a Pblock)

- Primitive cell (assigned to a Pblock)
- Primitive cell (placed and assigned to a Pblock)
- Primitive cell (unplaced)
- Primitive cell (placed)

Using the Device Constraints Window

The Device Constraints window (shown in the following figure) enables you to create, edit, and view internal VREF and DCI_CASCADE constraints. At the top of the Device Constraints window, you can select Internal VREF or DCI Cascade from the drop-down menu, as shown in the following figure. For more information, see this link in the *Vivado Design Suite User Guide: I/O and Clock Planning* (UG899). To open the Device Constraints window, select **Window → Device Constraints**.

Figure 39: **Device Constraints Window**



Device Constraints Window Toolbar Commands

The local toolbar contains the following commands:

- **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.
- **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.
- **Settings:** Controls the display of information in the window.
  - **Scroll to selected object:** Scrolls the Device Constraints window to display objects selected in other windows such as the Package Pins or Device windows.

Using the Properties Window

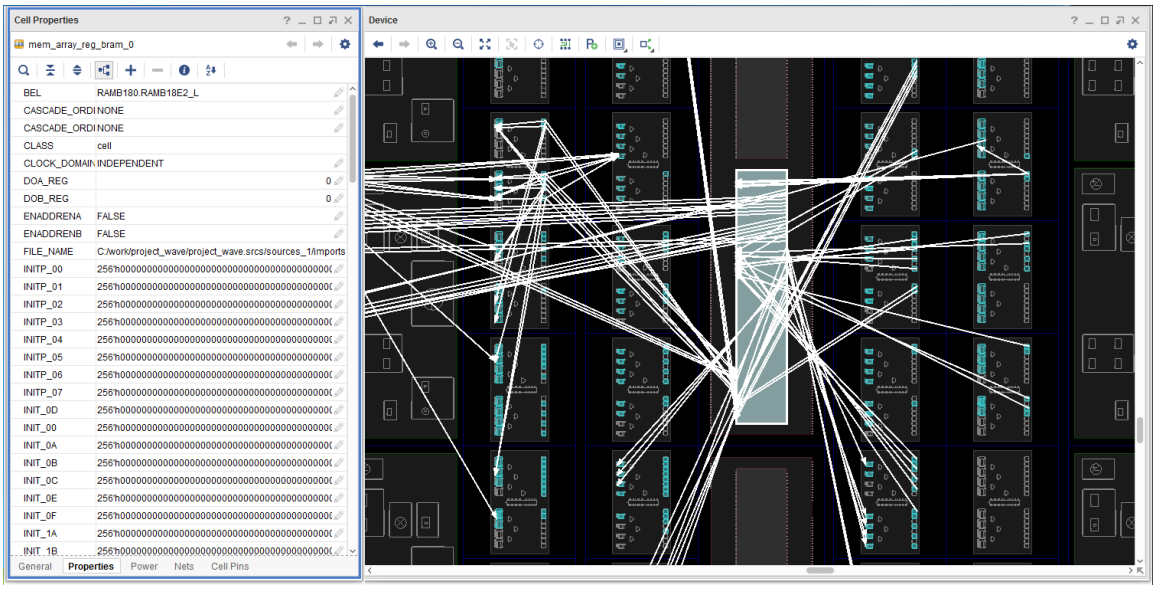
The Properties window displays information about selected logic objects or device resources. When you select an object, its properties dynamically display in the Properties window. To open the Properties window, select **Window → Properties**. Alternatively, you can right-click an object, and select **<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, the following figure 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.*

Figure 40: **Properties Window**



Related Information

Editing Properties for Multiple Objects

Properties Window Toolbar Commands

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

- **Search:** Opens the search bar to allow you to quickly locate objects in the Properties window.
- **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 Device Constraints window.
- **Show or Flatten All Property Hierarchies:** 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.
- **Remove Properties:** Removes a property from the selected object.
- **Show Description:** Toggles the display of detailed information in the description area at the bottom of the Properties window.
- **Sort Properties:** Sorts the property list alphabetically.

- **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.
- **Settings:** Controls the display of information in the window.
  - **Automatically Update:** Toggles the Properties window to auto-update as new objects are selected or to remain static displaying the properties of the currently selected object. By default, the Properties window is updated to display the properties of the latest object as new objects are selected.

Properties Window Popup Menu Commands

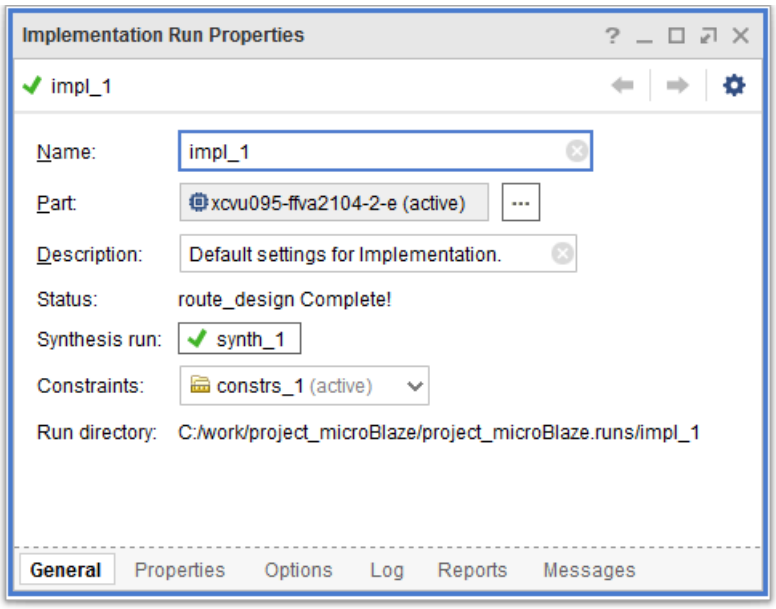
The popup menu contains the following commands:

- **Add Properties:** Adds a new property to the selected object. This command is available for certain object types only.
- **Remove Properties:** Removes a property from the selected object.
- **Reset Properties:** Resets a property or object from within one of the views of the Properties window. This command is available for certain object types in certain windows only.
- **Copy Properties:** Copies the property to the clipboard.
- **Export to Spreadsheet:** Exports the information in the Properties window to a spreadsheet file.

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. The following figure shows the Implementation Run Properties window for a selected run. To open the Run Properties window, select a run in the Design Runs window, and select **Window → Properties**.

Figure 41: **Implementation Run Properties Window**



Run Properties Window Views

The Run Properties window includes the following views to display information and set options for the design run.

- 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.

Related Information

Configuring Project Settings

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 design checkpoint, the strategy to use for the run, and the detailed command line options and values for the strategy. It includes the following fields:

- **Write Incremental Synthesis:** Includes information in the synthesis checkpoint for the tools to compare during the next incremental synthesis run.
- **Incremental Synthesis/Incremental Implementation:** Specifies a design checkpoint to use as a reference for the next synthesis or implementation run. For more information, see this link in the *Vivado Design Suite User Guide: Synthesis* (UG901) and this link in the *Vivado Design Suite User Guide: Implementation* (UG904).
- **Strategy:** Specifies the predefined strategy to use. You can modify the values of the command options related to the selected strategy. An asterisk appears next to options with modified values to indicate that the value was changed from the default.
- **Description:** Provides details about the selected strategy.

You can use the following popup menu commands:

- **Save Strategy As:** Saves the new option settings as a strategy for later use in other runs.
- **Refresh:** Restores 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. Xilinx recommends that you cancel the run and reset it. For more information, see this link in the Vivado Design Suite User Guide: Implementation (UG904).*

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 **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 a report to open it.

In the Reports view, you can also specify an existing report strategy. If a user-defined strategy is selected, you can add or remove reports, edit the options for existing reports, and enable or disable reports. For more information, see Creating Report Strategies.

For more information on implementation reports, see this link in the *Vivado Design Suite User Guide: Design Analysis and Closure Techniques* (UG906).

Related Information

Creating Report Strategies

Messages View

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

Using the Selection Window

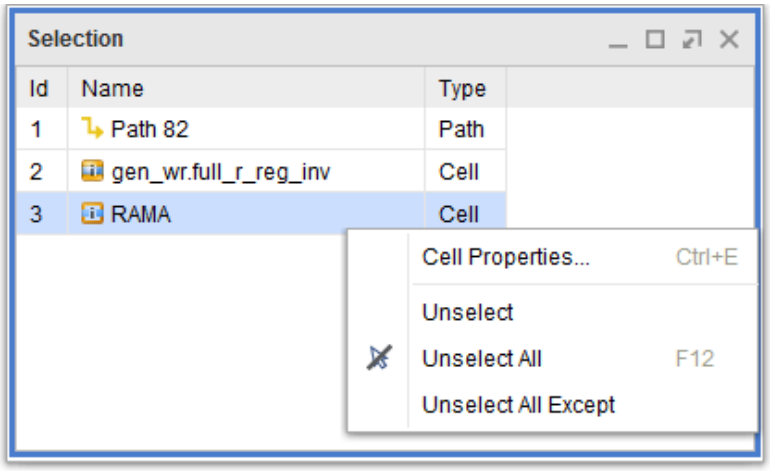
In the Vivado IDE, you can select objects as follows:

- **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.
- **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 (shown in the following figure) 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 42: **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 **Unselect**, **Unselect All**, and **Unselect All Except** commands from the popup menu.
- 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.

Related Information

Creating Report Strategies
Setting Selection Rules

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, the **Ctrl+M** keyboard shortcut, or the `mark_objects` Tcl command to mark objects using the default color.

- 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 across all open windows.

  ***Note*:** Alternatively, you can use the **Highlight** command from the popup menu, the **Ctrl+H** keyboard shortcut, or the `highlight_objects` Tcl command to highlight objects using the default color.

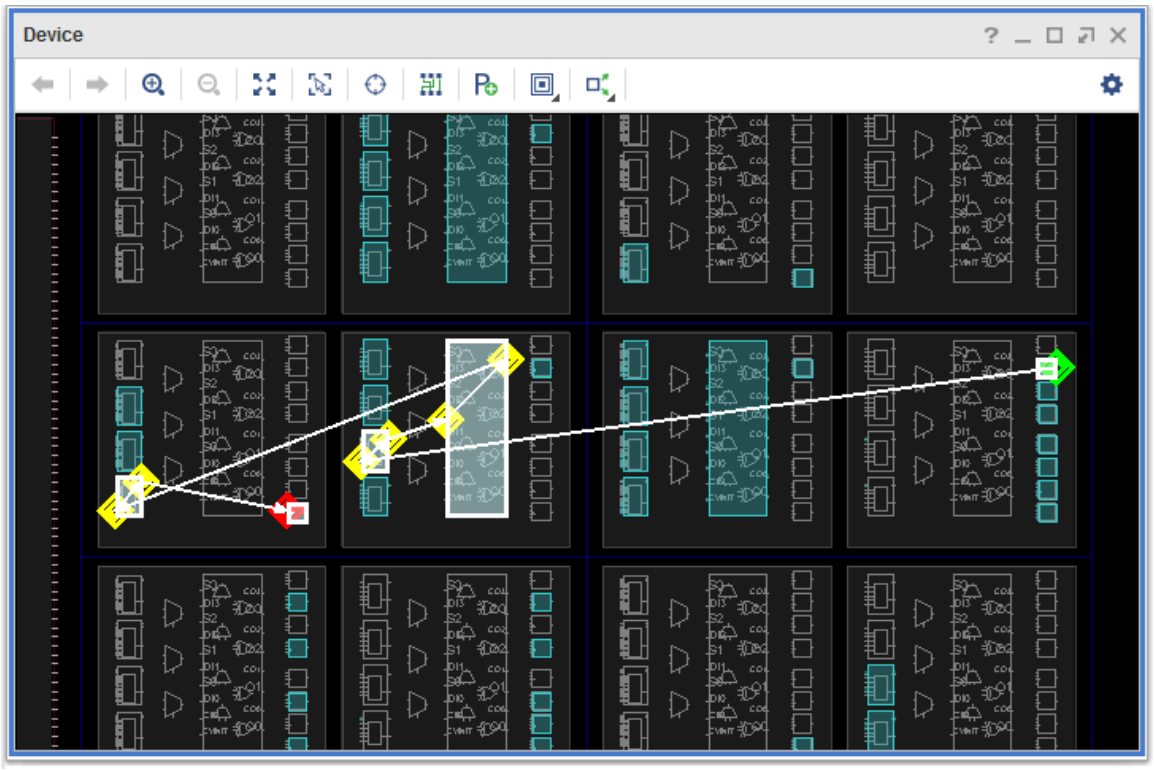
TIP: *To adjust colors, select **Tools → Settings**. In the Settings dialog box, click the **Colors** category and adjust the colors in the Highlight and Mark subcategories. For more information, see Specifying Colors.*

TIP: *To show the mark and highlight commands in the Tcl console, enable the **Record Tcl commands for highlight and mark actions** option in the Settings dialog box. For more information, see Specifying Project Default Settings.*

IMPORTANT! *Highlighting is design-specific, and Vivado IDE removes any highlighting when you reload or modify the design.*

The Mark command is also available in other windows, including the Netlist window, Hierarchy window, and Timing Report window. The following figure 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 43: **Marked Timing Path Symbols in Device Window**



Related Information

Specifying Colors
Specifying Project Default Settings

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 Color** to remove all highlights of a specified color.
- **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 window
- Schematic window
- Hierarchy window
- Timing Constraints window
- Waveform window
- Diagram window (for block designs)
- Property Editor
- IP Catalog

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 Schematic Window and Using the Hierarchy Window.

Related Information

Using the Schematic Window
Using the Hierarchy 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, move cells, or drag to 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. 

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.

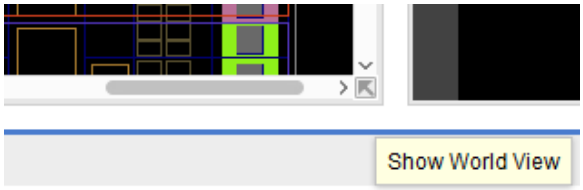
Using the World View

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 **Show World View** button in the lower right corner of a graphical window, as shown in the following figure.

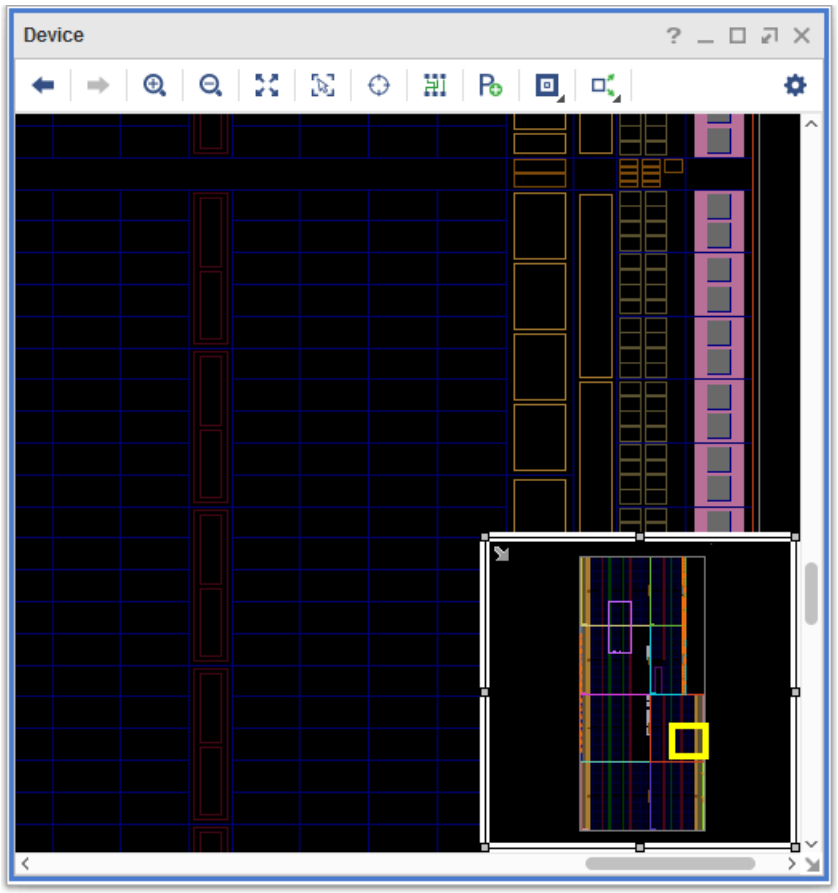
TIP: *If the Show World View button is hidden, zoom in on the window. You can also right-click in the window, and select* ***Show World View****.*

Figure 44: **Show World View Button**



The World view reflects the zoom area and the selected objects for the active window. In the following figure, 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 45: **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*:** To close the World view, click the downward pointing arrow icon in the view.

Printing the Workspace Window

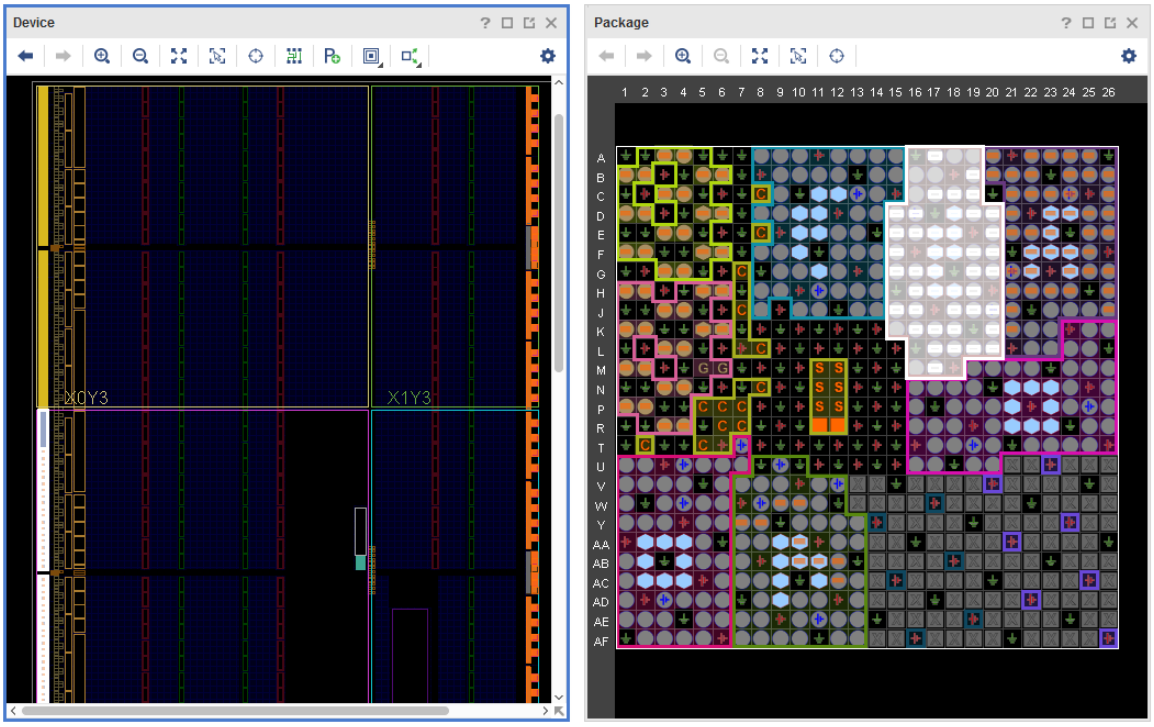
To print the window that is active in the workspace, select **File → Print**. This feature is available for the Device window, Package window, Schematic window, and Hierarchy window.

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 (shown in the following figure).

Figure 46: **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 use 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 Tab Group** or **Move to Next Tab 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 is a configurable, integrated text editor that supports syntax highlighting, on-the-fly syntax checking, assistance with errors and warnings, code folding, code completion, and file comparison. The Text Editor supports the following file types:

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

The Text Editor supports the following features, as shown in the following figure.

- On-the-fly syntax checking

  A wavy red underline indicates a syntax error, such as shown on Line 41.

  ***Note*:** Syntax errors can affect syntax that occurs later in the file. Therefore, fixing errors at the start of the file can fix errors that occur later in the file.

  IMPORTANT! *If you are using a third-party text editor, you must run syntax checking manually using the `check_syntax` Tcl command.*

- Assistance with errors and warnings

  To the right of the scroll bar, a red marker indicates the location of a syntax error or warning. You can click the marker to scroll to the line with the issue. You can also hover over the marker to read information about the issue. For Line 41, the syntax error is `<std_logi> is not declared`.

- Code completion

  You can insert your cursor in a line with an error, and press **Ctrl+Space** for code completion suggestions to resolve the error. On Line 41, the drop-down shows multiple suggestions.

  > **TIP:** *For information on adjusting the display of syntax highlighting, warnings, errors, and code completion, see Specifying Text Editor Settings.*

You can also use the following features in the Text Editor:

- Go to signal, type, and constant declarations

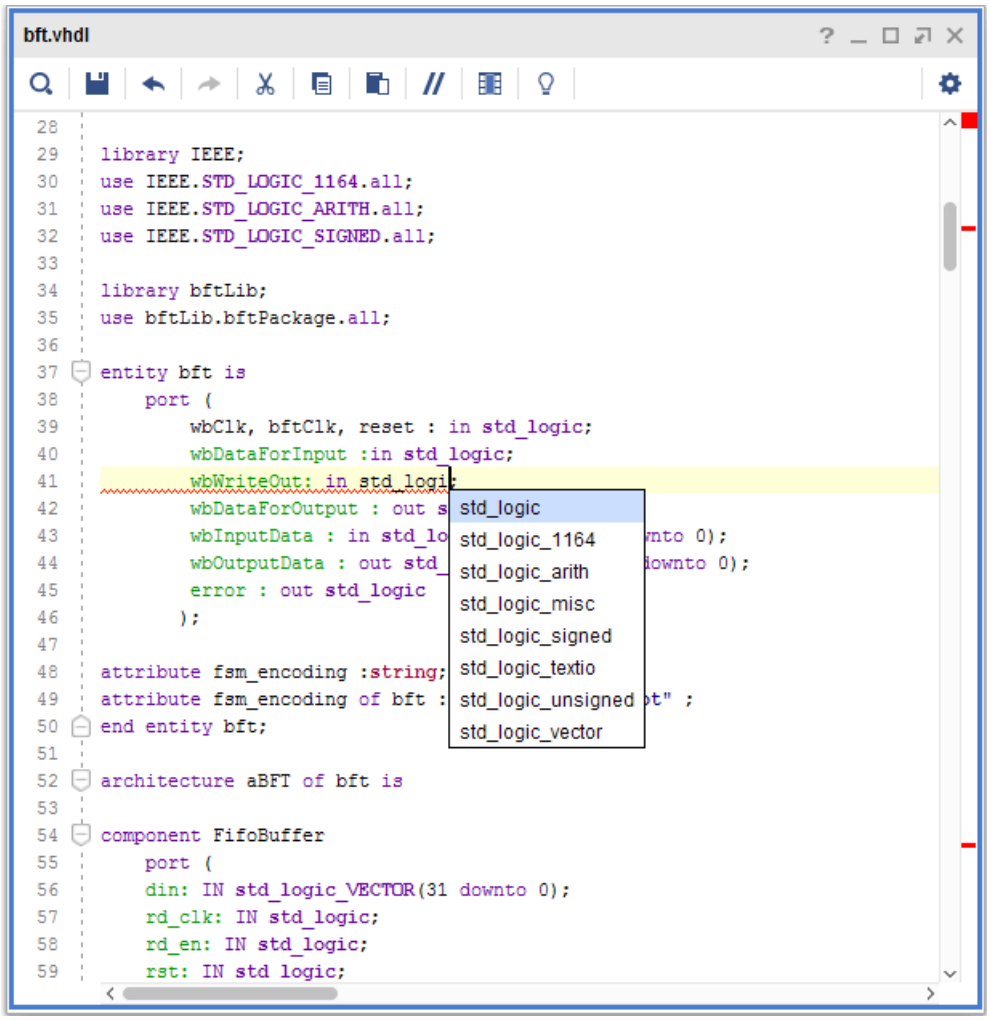
  Select the signal, type, or constant, right-click, and select **Go to Definitions**.
- Show usages for signals, types, and constants

  Select the signal, type, or constant, right-click, and select **Find Usages**.
- Display constant values in a tool tip

  Hover over the variable to show the tool tip.
- Show type definition in a tool tip

  Hover over the variable to show the tool tip.

Figure 47: **Text Editor Features**



Related Information

Specifying Text Editor Settings

Text Editor Commands

The following commands are available from the local toolbar and popup menu.

Related Information

Using Language Templates
Specifying Text Editor Settings

Text Editor Toolbar Commands

The local toolbar contains the following commands:

- **Find:** 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.*

- **Save File:** Saves changes to the currently displayed file. 
- **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. 
- **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. 
- **Toggle Column Selection Mode:** 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. 
- **Reformat Code:** Adjusts indenting and alignment of the code to improve its readability. The button operates on the entire file or a selected block of text. 
- **Language Templates:** Opens the Language Templates. For more information, see Using Language Templates. 
- **Settings:** Controls the display of information in the window. For details, see Specifying Text Editor Window Settings. 

Text Editor Popup Menu Commands

The popup menu contains the following commands:

- **Save File:** Saves changes to the currently displayed file.
- **Save File As:** Saves a file to a new name.
- **Save All Files:** Saves all open files that were modified.

- **Find Usages:** Shows all files in which the selected module, architecture, entity, or signal is located.
- **Go to Definition:** Shows the definition for the selected module, architecture, entity, or signal.
- **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.
- **Duplicate Selection:** Copies the selected text and pastes it to the cursor location, immediately in front of the current selection.
- **Select All:** Selects all text in the Text Editor.
- **Toggle column selection mode:** 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.
- **Find/Replace:** Finds and replaces text in the current file.
- **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 replace the search string with a new string using the Replace in Files command.
- **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.
- **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*:** The comment symbol inserted is contextually dependent on the type of file displayed. This command is only available for Verilog, SystemVerilog, and VHDL.

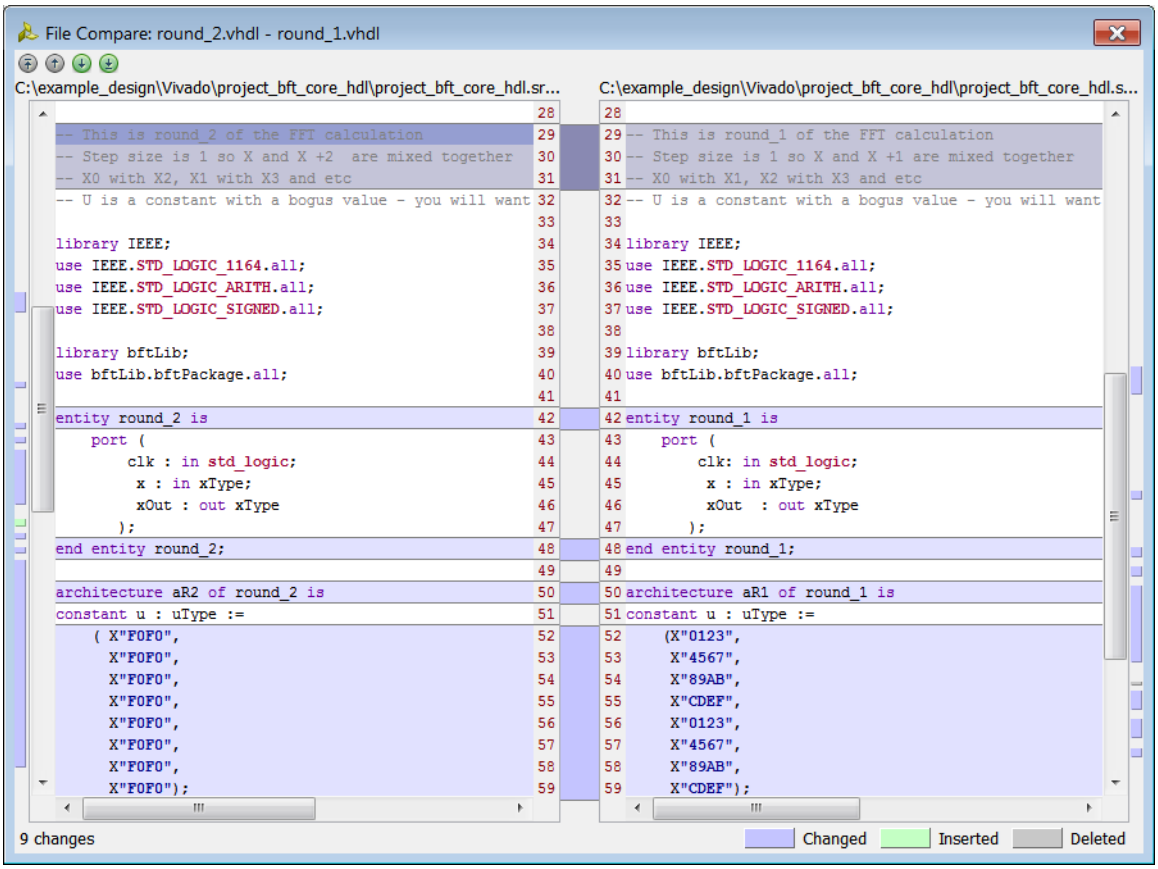
- **Blank Operations:** Configures the display of tabs, spaces, and special characters for the selected text. If no text is selected, configures the display for the entire document.
  - **Trim Leading Whitespace:** Removes the leading whitespace for the selected text. This operation works on the entire 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 (shown in the following figure) 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.
- **Folding:** Collapses and expands text based on hierarchy, making it easier to navigate and edit large files.
  - **Expand Current:** Expands the text at the cursor location.
  - **Collapse Current:** Collapses the text around the cursor location.
  - **Expand All:** Expands all text in the current file.
  - **Collapse All:** Collapses all text in the current file.
  - **Expand Comments:** Expands all comments in the current file.
  - **Collapse Comments:** Collapses all comments in the current file.

  Note: To enable or disable code folding, select **Tools → Settings**, click the **Text Editor** category, and select or deselect the **Perform Code Folding** option.
- **Insert Template:** Inserts the currently selected Language Template into your text file at the location of your cursor.

Figure 48: **File Compare Dialog Box**



Specifying Text Editor Window Settings

The Text Editor window settings control the display of code completion, syntax checking, line numbers, and the file path. To set the Text Editor window settings, click the **Settings** toolbar button .

After you finish specifying settings, 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 .*

Code Completion

The Code Completion settings control the following:

- **Code Completion:** Sets the preference for activating the code completion drop-down, whether using a shortcut key, displaying as you type, or disabling code completion.

- **Keyboard Choice Selection:** Sets whether to use the Tab key or Space bar to select the displayed value.

Syntax Checking

The Syntax Checking settings control the following:

- Third party Sigasi checking is enabled by default.
- Display Options: Specifies whether to display warnings and notes.
- Cache Setting: Shows the location of Sigasi cache and allows you to change the location from the User home directory. By default the Sigasi cache will be stored in one of the following locations:
  - Linux: <User Home Directory>/.Xilinx/Vivado/20xx. x/sigasi-cache
  - Windows: %APPDATA% \Roaming\Xilinx\20xx.x\sigasi-cache

General

The General settings control the following:

- **Perform code folding:** Enables code folding to manage the display of large files.
- **Show file path:** Shows the complete path for the text file.
- **Display line numbers:** Shows line numbers to improve navigation in the file.
- **Display matches for the selected word:** Highlights words that match the currently selected word.

Opening a Text File

To open a text file in the Text Editor:

1. Select **File → Text Editor → Open File**. This command opens a file browser that allows you to navigate to the file and open it for editing.
2. 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.
3. Click the file name from a warning or error message in the Messages window to open the selected file in the Text Editor.
4. To open the Vivado IDE journal and log files, select **File → Project → Open Log File** or **File → Project → Open Journal File**.

Creating a New Text File

Select **File → Text Editor → 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.*

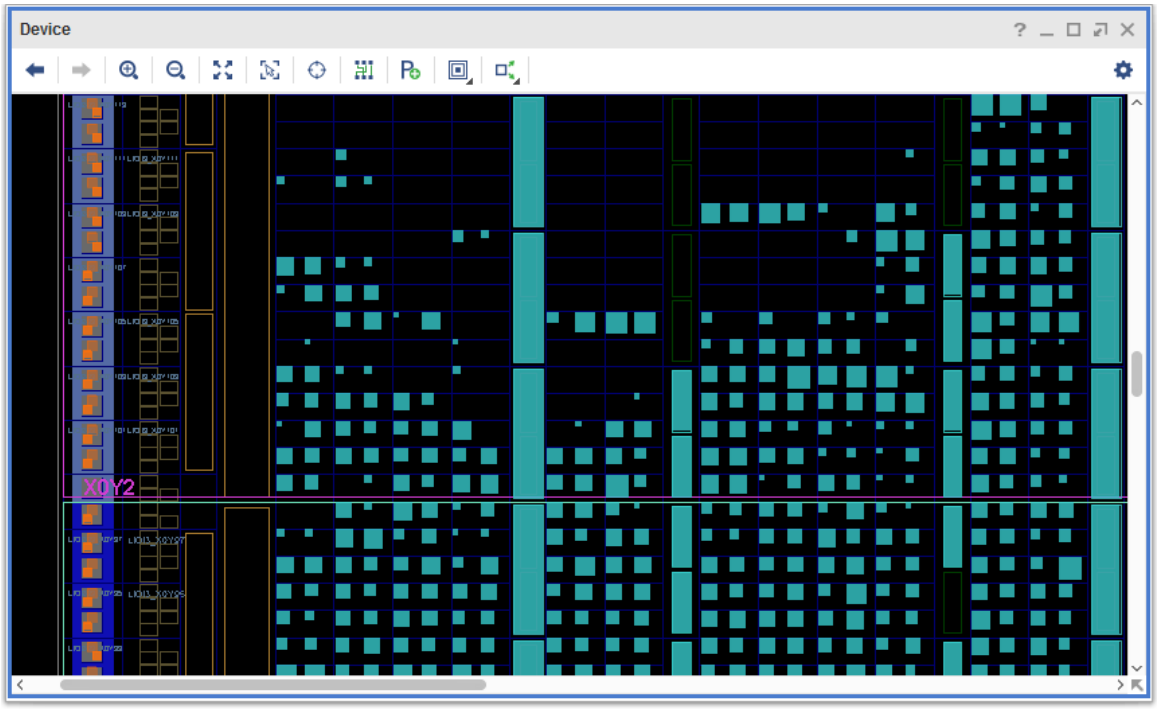
Using the Device Window

The Device window (shown in the following figure) is the main graphical interface used for design analysis and floorplanning. The Device window displays the device resources, including device 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. To open the Device window, select **Window → Device**.

For more information, see the following documents:

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

Figure 49: **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.

Device Window Toolbar Commands

The local toolbar contains the following commands:

- **Previous:** Resets the Device window to display the prior zoom and coordinates.
- **Next:** Returns the Device window to display the original zoom and coordinates after the Previous toolbar button 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.
- **Autofit Selection:** Automatically redraws the Device window around newly selected objects. This mode can be enabled or disabled.
- **Routing Resources:** Displays routing resources in the Device window.
- **Draw Pblock:** Places the cursor in Draw Pblock mode (crosshair) allowing you to create a new Pblock rectangle to place cells.
- **Cell Drag & Drop Modes:** Specifies how 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 the cell to use any available BELs within the slice.
  - **Assign Cell to Pblock Mode:** Assigns logic cells to Pblocks. This allows the implementation tools the most flexibility and is the default mode.
- **Show Cell Connections:** Shows the connectivity for selected objects based on the following settings.

- **Show Input Connections:** Shows the input connections for selected cells.
- **Show Output Connections:** Shows the output connections for selected cells.
- **Max Cell Count:** Limits the number of connections that are shown to improve drawing performance. You can increase this value to visualize cells with a large number of connections, but Xilinx recommends a setting of 1000 or less for optimal graphics display.

- **Settings:** Controls the display of information in the window. For details, see Specifying Device Window Settings. 

Related Information

Specifying Device Window Settings

Zooming and Panning

To zoom and pan:

1. Use the zoom commands in the popup menu and local toolbar.
2. 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.
3. Use scroll bars and dynamic pan capabilities to pan the viewable area of the device.

Related Information

Using Mouse Strokes to Zoom and Pan

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 Design or Device Objects.

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.

Related Information

Finding Design or Device Objects
Understanding the Context-Sensitive Cursor

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 resources, including all clock resources, such as BUFG, BUFGCTRL, BUFR, and BUFHCE components. The interior of the device is broken up into smaller rectangles called tiles, which are placement sites for the different types of logic primitives for the architecture.

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 you zoom into the device logic. Other resources, such as clock regions and I/O banks, appear even when viewing the whole device. In addition, you can control the display of specific objects or resources in the Device window, as described in Specifying Device Window Settings.

The Vivado IDE Device window displays resources as follows:

- **I/O pads and clock objects:** Rectangles around the periphery and down the center of the device.
- **I/O banks:** Thin, color-shaded rectangles just outside the row of I/O pads.
- **Available I/O bank sites:** Color-filled I/O bank rectangles.
- **Unbonded I/O banks:** Rectangles with a white X.
- **I/O clock pads:** Filled-in rectangles.

TIP: *Hover your cursor over a logic site to see a tooltip that identifies each site in the Device window.*

Related Information

Specifying Device Window Settings

Specifying Device Window Settings

The Device window settings define the layers and objects that are visible in the Device window and provide a legend of the icons used in the window. To set Device window settings, click the **Settings** toolbar button  .

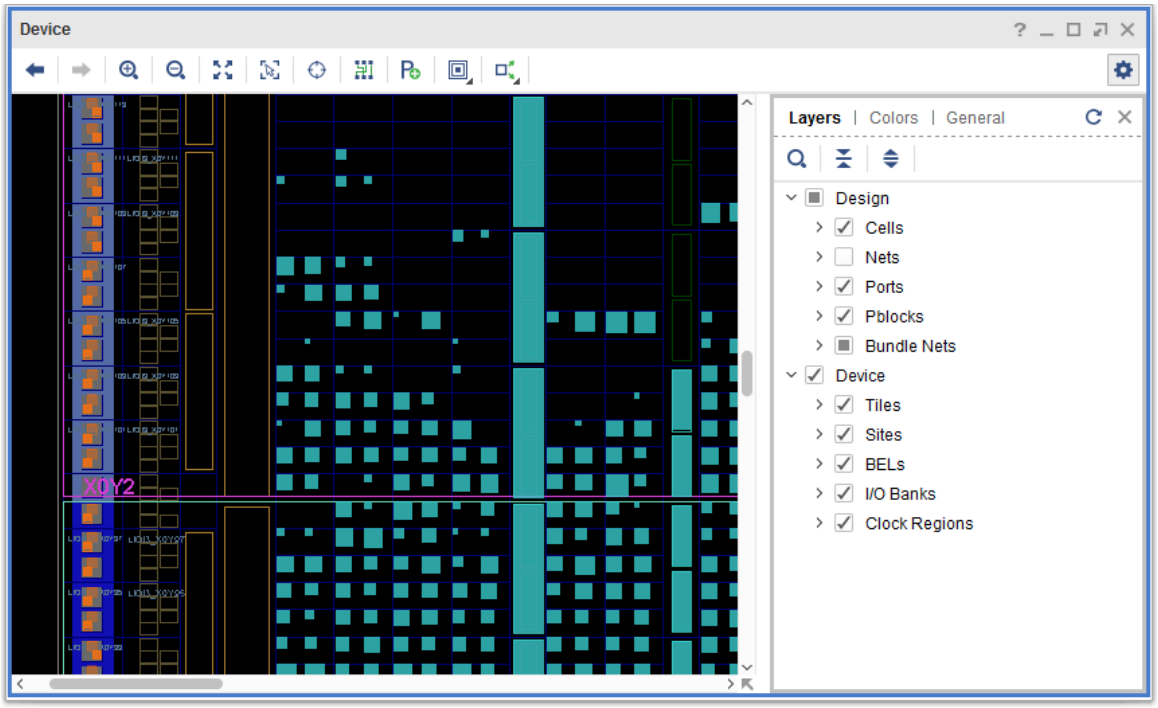
After you finish specifying settings, click the Close button in the upper right corner. The Vivado IDE stores your settings and reloads them each time the tool is launched.

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

Device Window Layers

The Layers settings (shown int the following figure) 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 50: **Device Window—Layers Settings**



Following are the two primary branches:

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

Using the Layers settings, you can do the following:

- Use the expand › and collapse ⌄ buttons to expand or collapse the levels of the tree view.
- Click the check box 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.

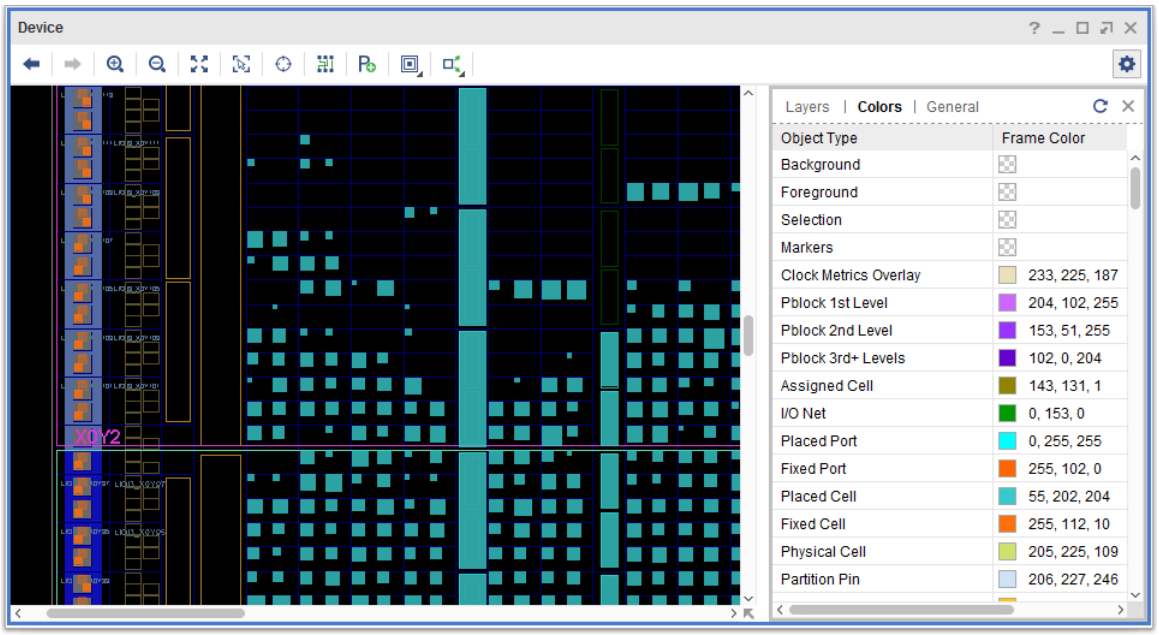
  Note: If you cannot see a specific object or layer in the Device window, check the Layers settings to see if the design object or device resource is currently hidden.

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

Device Window Colors

The Colors settings (shown in the following figure) change the color and fill values for elements in the Device window. For information, see Changing Colors.

Figure 51: **Device Window—Colors Settings**



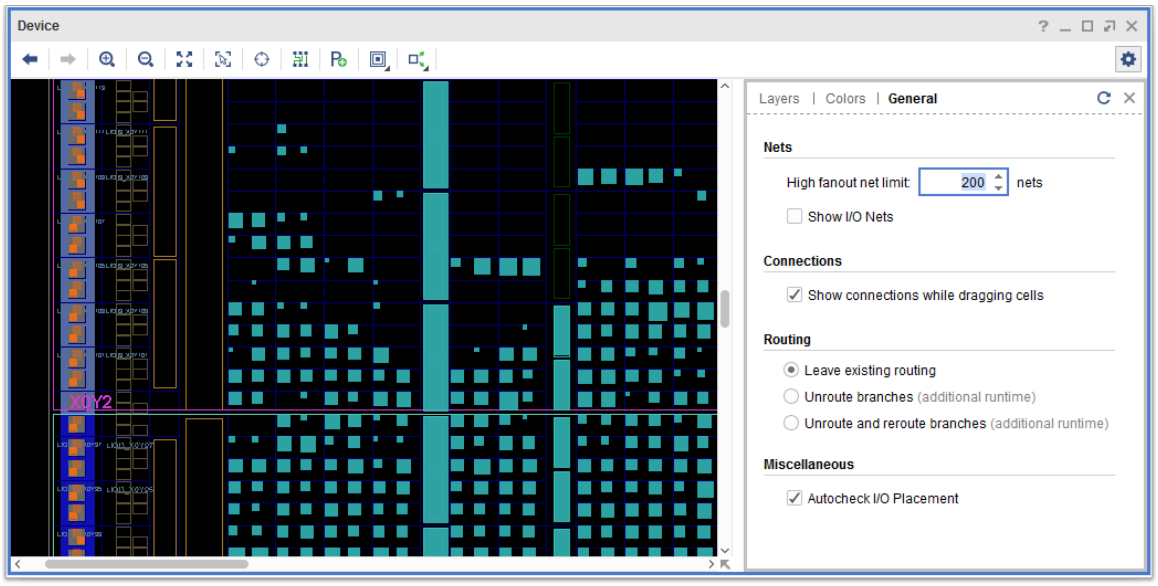
Related Information

Changing Colors

Device Window General

The General settings (shown in the following figure) define the display characteristics of the net connections on the device.

Figure 52: **Device Window—General Settings**



The following settings are available:

- **Nets:**
  - **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.
  - **Show I/O Nets:** Toggles the display of I/O connectivity to placed logic or Pblocks.
- **Connections:**
  - **Show connections while dragging cells:** Displays nets connected to the selected cell when dragging and placing the cell in the Device window.
- **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:**
  - **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.

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.

***Note*:** When you select the clock region, the Vivado IDE also selects the associated I/O banks

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.

Related Information

Splitting the Workspace

Using the Package Window

The Package window (shown in the following figure) 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). To open the Package window, select **Window → Package**.

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:

- **Previous:** Resets the Package window to display the prior zoom and coordinates. 
- **Next:** Return the Package window to display the original zoom and coordinates after the Previous toolbar button 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. 
- **Autofit Selection:** Automatically redraws the Package window around newly selected objects. This mode can be enabled or disabled. 
- **Settings:** Controls the display of information in the window. For details, see Specifying Package Window Settings. 

Related Information

Specifying Package Window Settings

Specifying Package Window Settings

The Package window settings define the layers and objects that are visible in the Package window and provide a legend of the icons used in the Package window. To set Package window settings, click the **Settings** toolbar button  .

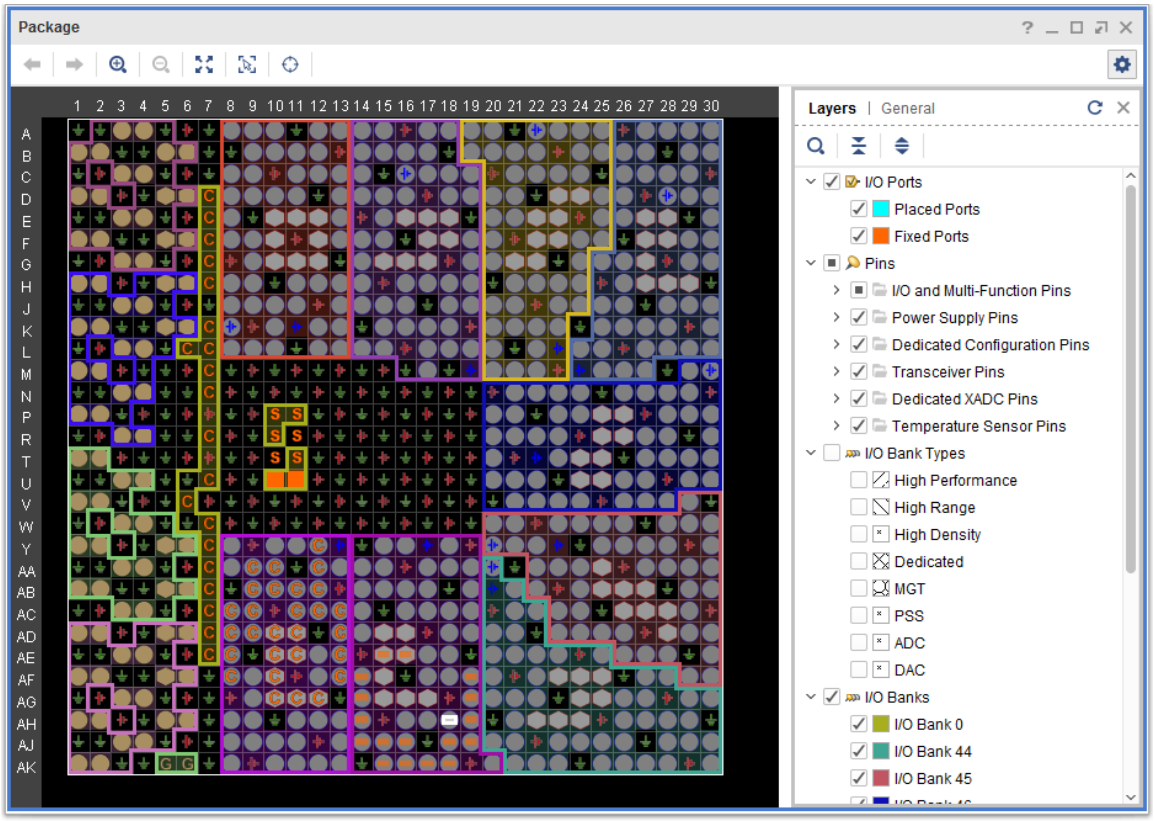
After you finish specifying settings, click the Close button in the upper right corner. The Vivado IDE stores your settings and reloads them each time the tool is launched.

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

Package Window Layers

The Layers settings (shown in the following figure) control the elements displayed in the Package window.

Figure 53: **Package Window—Layers Settings**



The available layers are listed hierarchically in a tree view:

- Use the expand › and collapse ⌄ buttons to expand or collapse the levels of the tree view to see the different layers.
- Click a check box 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

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

The display of a specific pin in the Package window depends on the combination of layers that represent the pin in the Package window settings. For example, if you deselect **I/O Banks** in the Package window settings, ground pins are displayed, but the user I/O and multi-function pins are not displayed even if you select **I/O and Multi-Function Pins** under the Pins heading.

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 window settings.*

Following are the different categories of layers in the Package window settings:

- **I/O Ports:** 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:** 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 Bank Types:** Different types of I/O banks, which vary based on the targeted device. For example, the following figure shows the high performance banks that are displayed when you select the High Performance layer.

Figure 54: **High-Performance I/O Bank**

- **I/O Banks:** Pin sites for each of the banks on the device as well as for the GT pins. Each I/O bank and GT bank is color-coded to allow you to differentiate between the banks of pins.

TIP: *Turning off an I/O bank layer is an easy way to prevent pin assignment. Using this method enables you to reserve a bank for later use or show that a bank is full.*

- **Other:** Grid lines for x-axis and y-axis drawn behind the sites.

Device Window General

The General settings control the following:

- **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 View:** Displays the package pins as viewed from the bottom.
- **Show Differential I/O Pairs:** Displays the differential pair pins in the Package window.

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.

Related Information

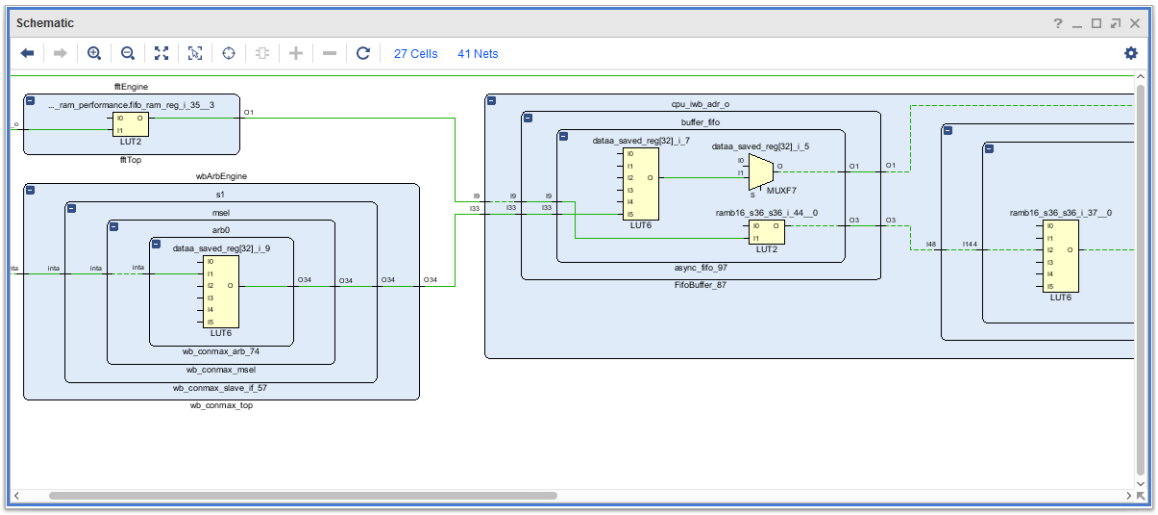
Splitting the Workspace

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 the following figure. 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 this link in the *Vivado Design Suite User Guide: System-Level Design Entry* (UG895). For more information on synthesized netlist analysis, see the *Vivado Design Suite User Guide: Design Analysis and Closure Techniques* (UG906).

Figure 55: **Schematic Window**



***Note*:** Dotted lines indicate that the net is connected to additional logic that is not displayed in the schematic.

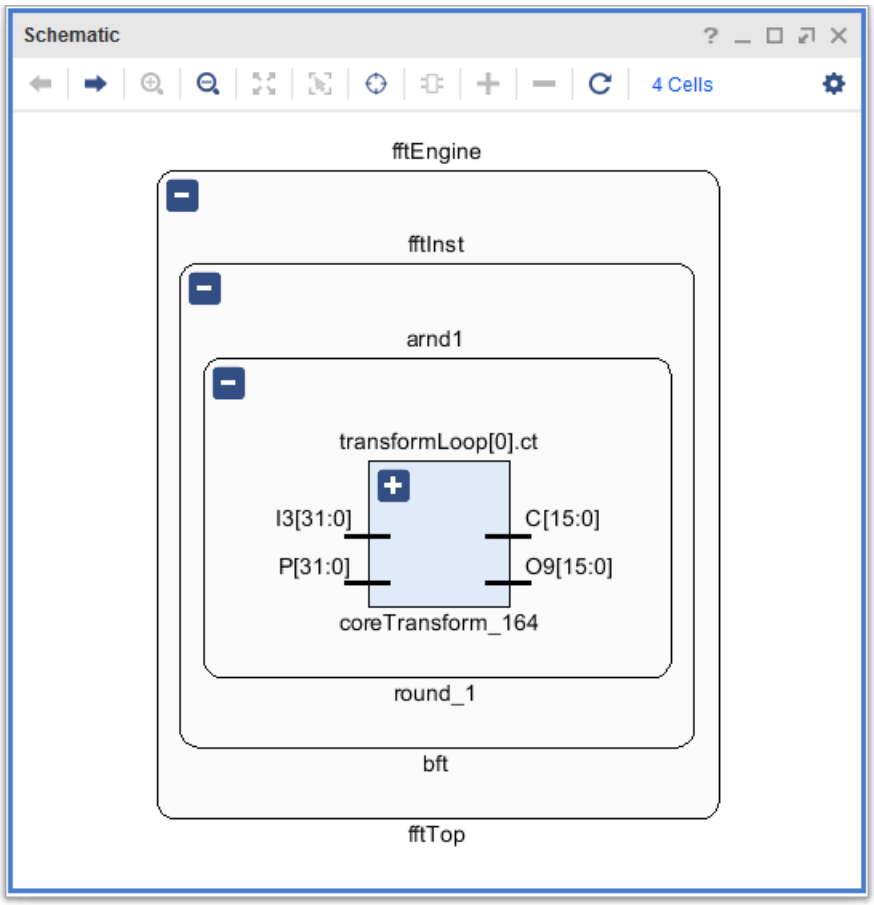
Creating a Schematic Window

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 the following figure.

Figure 56: **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:

- **Previous:** Resets the Schematic window to display the prior zoom, coordinates and logic content.
- **Next:** Returns the Schematic window to display the original zoom, coordinates and logic content after Previous is used.

Note: This command is only available for schematics with less than 12,000 objects.

- **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.
- **Autofit Selection:** Automatically redraws the Schematic window around newly selected objects. This mode can be enabled or disabled.
- **Autohide Pins:** Toggles the pin display on selected hierarchical modules. When a schematic is generated, higher levels of the hierarchy display as concentric rectangles without pins, and cells hide the unconnected pins, as shown in the previous figure. 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:** Recreates the Schematic window with the newly selected elements added to the existing schematic.
- **Remove:** Recreates the Schematic window with the currently selected elements removed from the existing schematic.
- **Regenerate Schematic:** Redraws the active Schematic window.
- **Settings:** Controls the display of information in the window. For details, see Specifying Schematic Window Settings.

Related Information

Specifying Schematic Window Settings

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 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:

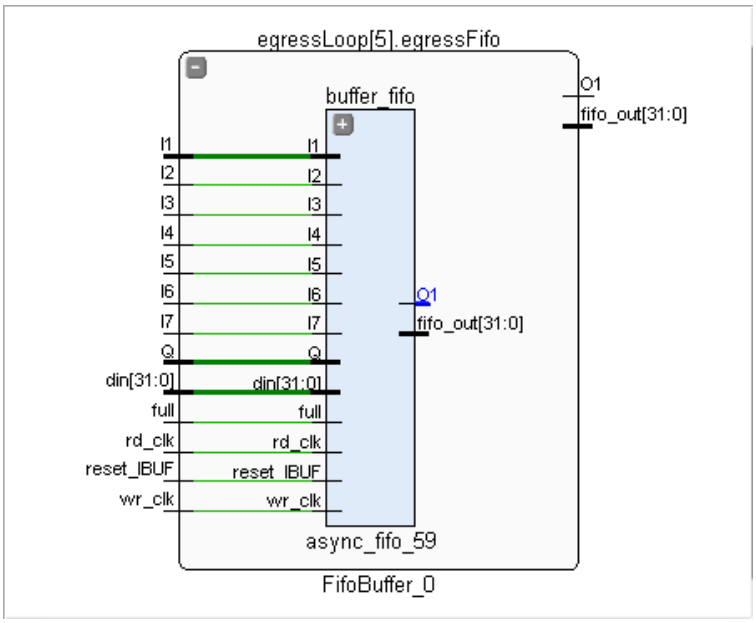
- Double-click a pin of a cell to trace the net down into, or up out of the hierarchy. A pin is displayed on the schematic symbol with a stub inside and outside of the symbol, as shown for `O1` and `fifo_out[31:0]` in the following figure. This reflects the ability to expand inside or outside the symbol.

  Note: Pins labeled 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 57: **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 Leaf Cells:** 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.
- **To 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:

1. Click an object in the Schematic window.
2. Use the **Ctrl** key to select multiple objects.
3. 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 Cell 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.

Specifying Schematic Window Settings

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

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.*

Schematic Window Display Settings

The Display settings (shown in the following figure) control the display of the following features in the Schematic window:

- **Inst Equation:** Labels cells with truth table equations.
- **Fanout for Scalar/Bus Pin:** Labels cell pins with fanout values.
- **Setup 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 this link in the *Vivado Design Suite User Guide: Design Analysis and Closure Techniques* (UG906).

  Note: If you updated timing data, you must refresh the schematic to display the new slack values.

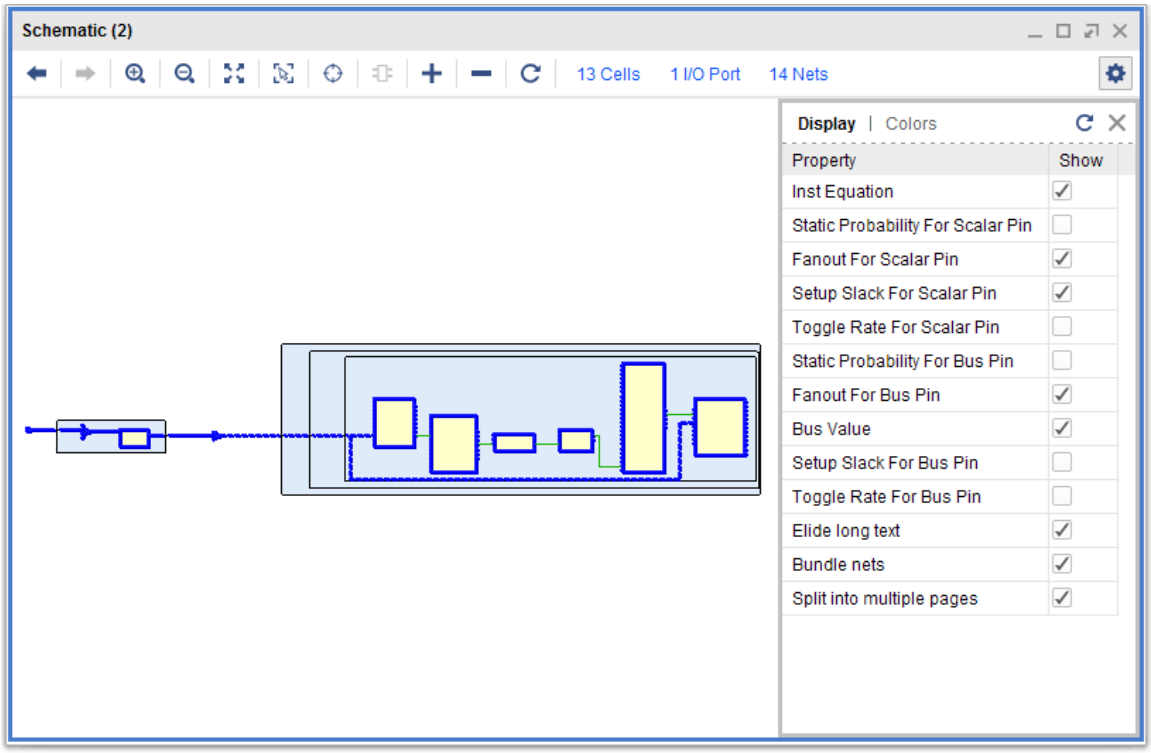
- **Static Probability for Scalar/Bus Pin:** Labels nets connected to cell pins with static probability. For more information, see this link in the *Vivado Design Suite User Guide: Power Analysis and Optimization* (UG907).
- **Toggle Rate for Scalar/Bus Pin:** Labels the net connected to cell pins with toggle rate. For more information, see this link in the *Vivado Design Suite User Guide: Power Analysis and Optimization* (UG907).

Note: Static probability and toggle rate values do not display until after data is generated by the **Reports → Report Power** command. If you update the power data, you must run the Report Power command again to display the new static probability and toggle rate.

- **Bus Value:** Labels bus pins with bus values.

- **Elide Long Text:** Truncates long text for a cleaner display.
- **Bundle Nets/Cells:** Shows buses or cells as a bundled net.
- **Split into Multiple Pages:** Shows large schematics on multiple pages.

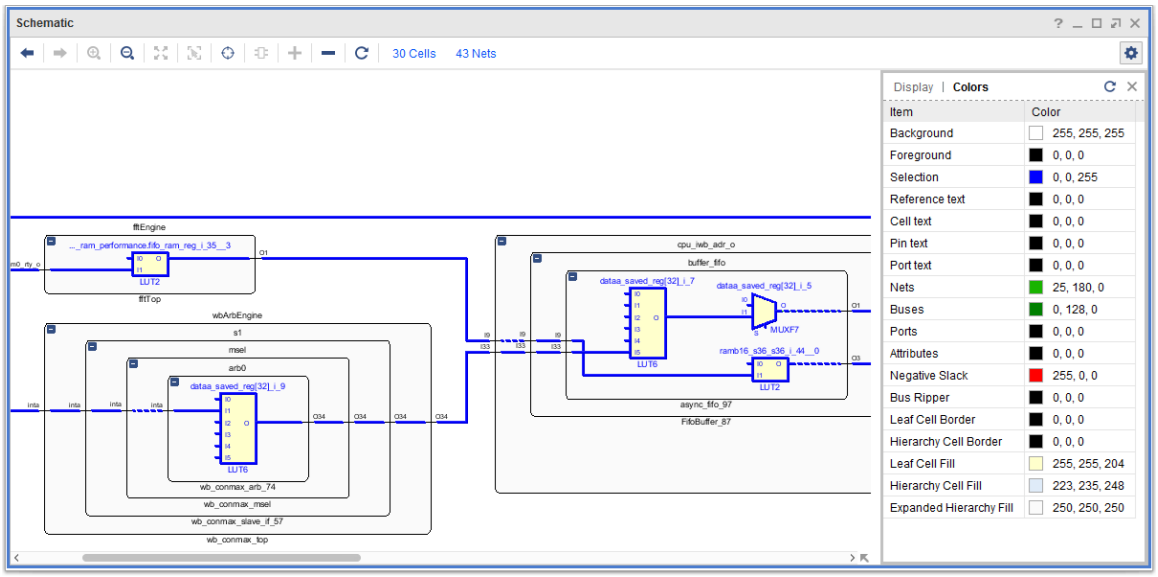
Figure 58: **Schematic Window—Display Settings**



Schematic Window Colors Settings

The Colors settings (shown in the following figure) change the color of elements of the Schematic window:

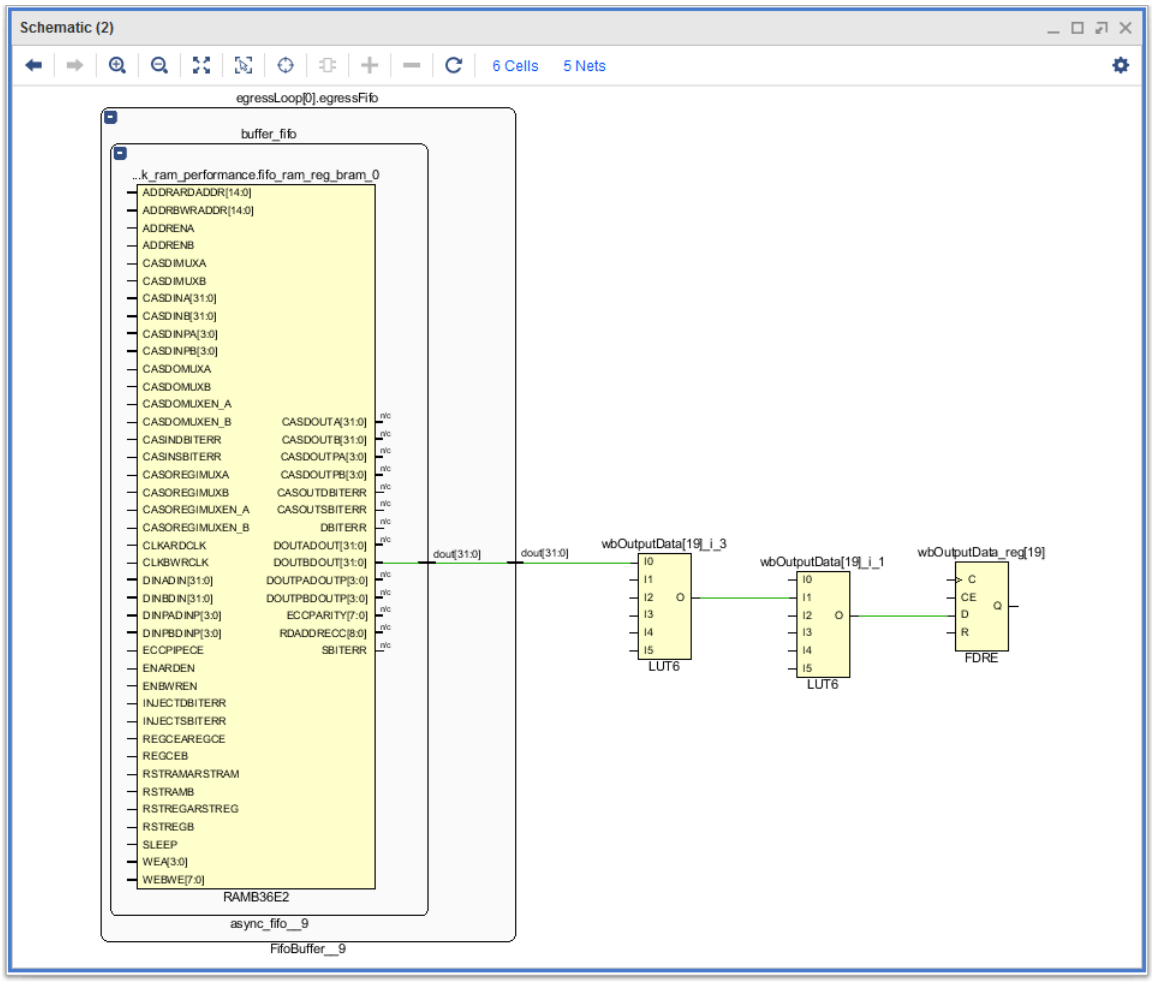
1. Click a color box to expose a drop-down menu, and select from a list of available colors.
2. Select **More Colors** to display more colors to choose from.
3. Enter a specific RGB value directly in the text field for the color.



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 the following figure. You can use the Report Timing popup menu to perform more detailed analysis on from/through/to ports, pins, cells, and nets. For more information on setting the timing path logic, see this link in the *Vivado Design Suite User Guide: Design Analysis and Closure Techniques* (UG906).

Figure 59: **Timing Path in Schematic Window**

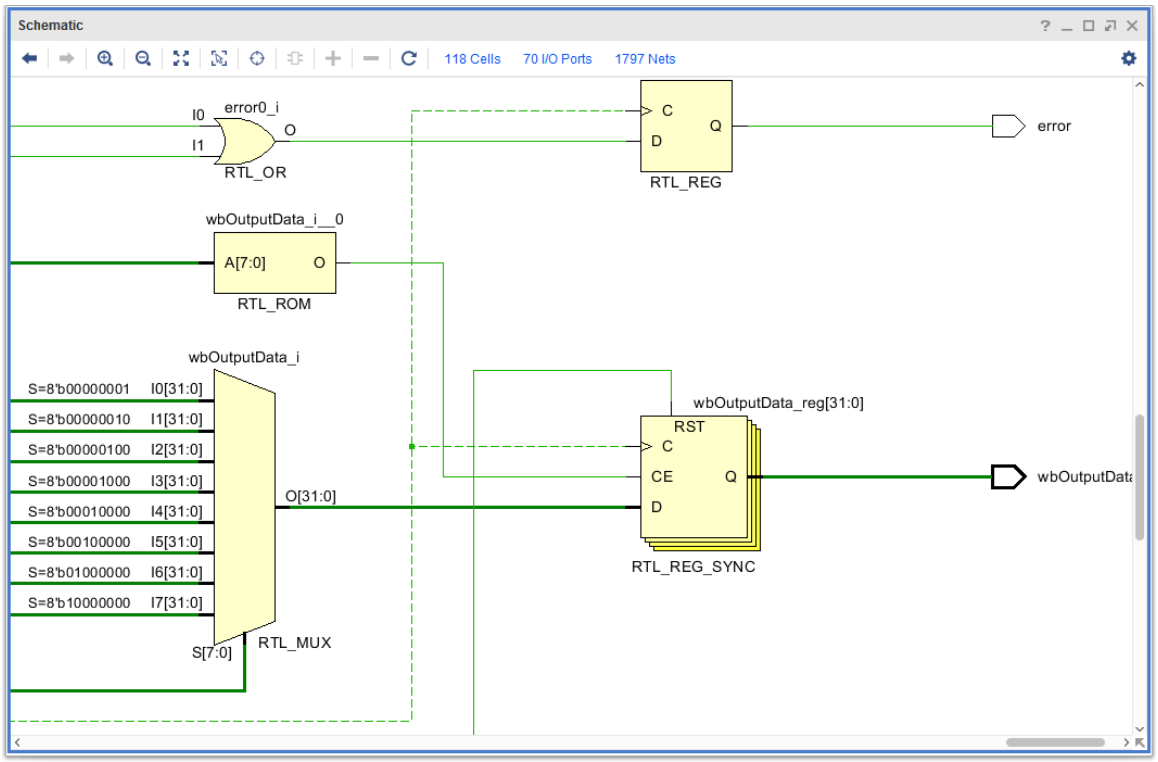


Viewing Bundled Logic in the RTL Schematic Window

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 the following figure.

To set this property, click the Settings toolbar button , and select **Bundle Nets**.

Figure 60: **RTL Schematic with Bundled Logic**

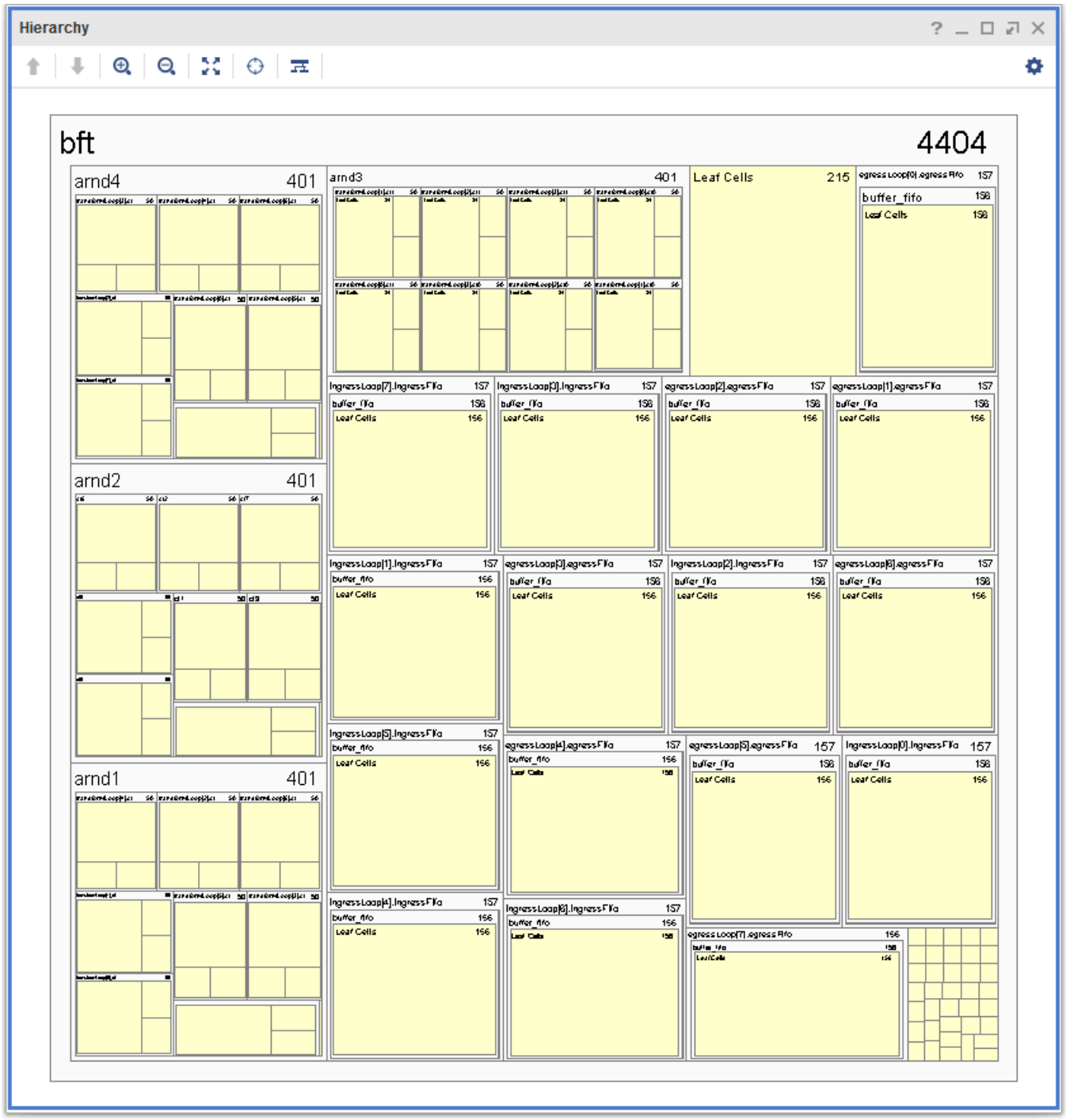


Using the Hierarchy Window

The Hierarchy window displays a graphical representation of the logic hierarchy for the current design, based on the current top module. This 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 Hierarchy window is used primarily during design analysis and floorplanning, as described in this link in the *Vivado Design Suite User Guide: Design Analysis and Closure Techniques* (UG906). To open the Hierarchy window, right-click in a window, such as the Netlist window, and select the **Show Hierarchy** command.

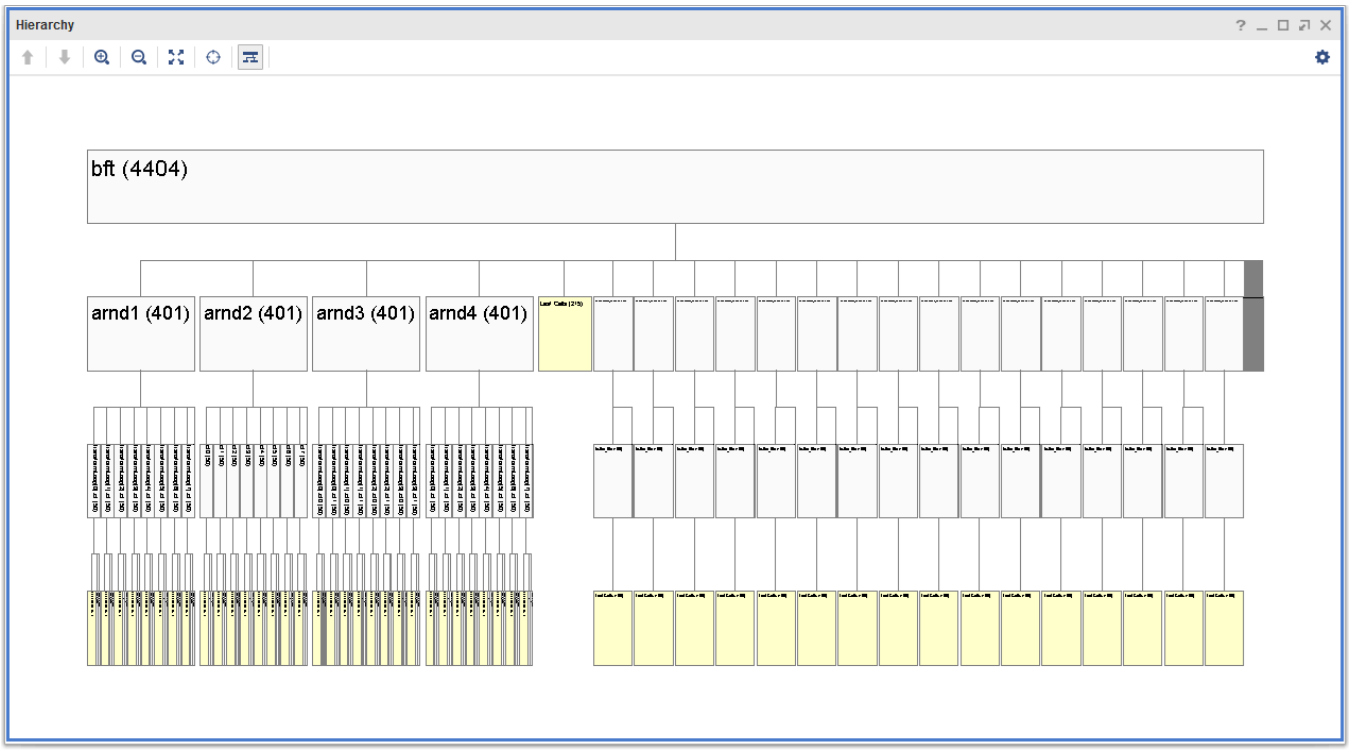
In the Block view of the Hierarchy window (shown in the following figure), each instance displays within the hierarchical block that contains it. Primitive logic is grouped into folders that are represented as sub modules. For more information about primitive logic folders, see Using the Netlist Window.

Figure 61: **Hierarchy Window Block View**



To view the design from top to bottom, click the **Show Tree View** button . In the Tree view of the Hierarchy window (shown in the following figure), you can identify the relationship between hierarchical modules, approximate module sizes, and module location within the design. The widths of the blocks in the Hierarchy window are based on the relative device resources consumed by that instance of hierarchy.

Figure 62: **Hierarchy Window Tree View**



TIP: *To select logic parent modules for Pblock assignment in the Hierarchy window, right-click a module and select* ***Select Leaf Cell Parents****. From the parent module, select* ***FloorplanningDraw Pblock*** *or* ***Assign to Pblock****.*

Related Information

Using the Netlist Window

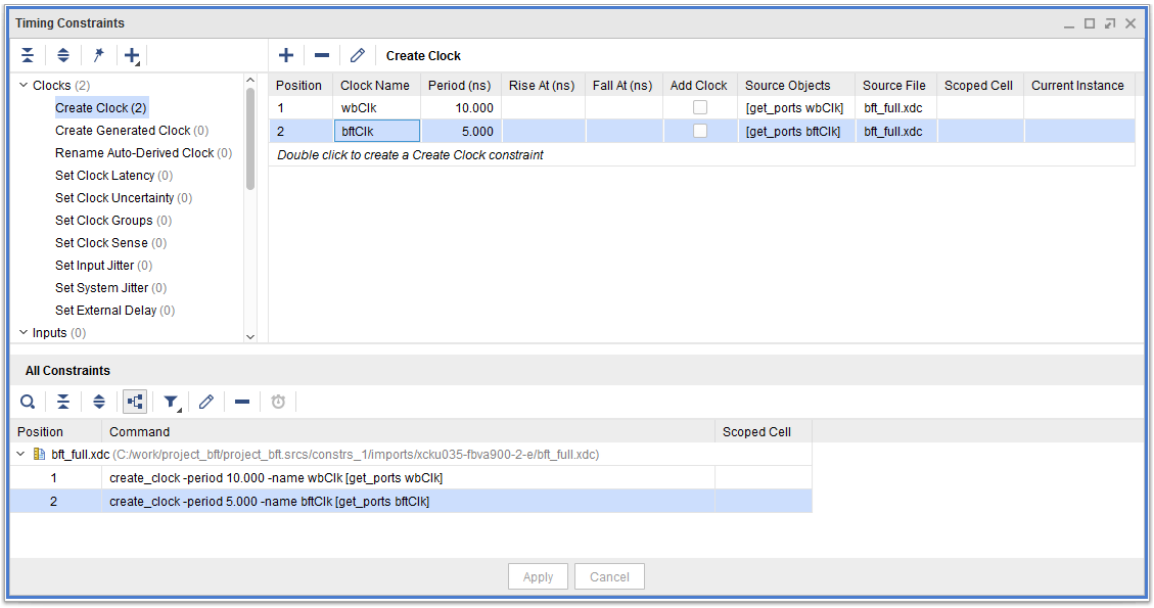
Using the Timing Constraints Window

The Timing Constraints window (shown in the following figure) 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 this link in the *Vivado Design Suite User Guide: Using Constraints* (UG903). 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.

***Note*:** 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.

VIDEO: *Select **Tools → Timing → Constraints Wizard** on a synthesized design to create a top-level XDC file based on design methodologies recommended by Xilinx. This wizard guides you through specifying clocks, setting up input and output constraints, and properly constraining cross-clock domain clock groups. For an overview, see the Vivado Design Suite QuickTake Video: Using the Timing Constraints Wizard.*

Figure 63: **Timing Constraints Window**



Timing Constraints Window Toolbar Commands

The local toolbar in the tree view pane contains the following commands:

- **Collapse All:** Collapses all constraints.
- **Expand All:** Expands all constraints.
- **Constraints Wizard:** Opens the Timing Constraints Wizard, which identifies missing constraints in your synthesized or implemented design and makes constraint recommendations. For more information, see this link in the *Vivado Design Suite User Guide: Using Constraints* (UG903).
- **Create Timing Constraint:** Creates 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.

The local toolbar in the Create Clock pane contains the following commands:

- **Create Constraint:** Opens the Create Clock dialog box in which you can create a constraint.
- **Remove Constraint:** Deletes the selected constraint.

- **Edit Constraint:** Opens the Edit Create Clock dialog box for the selected constraint.

The local toolbar in the All Constraints pane contains the following commands:

- **Search:** Opens the search bar to allow you to quickly locate constraints.

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

- **Collapse All:** Collapses all constraints.
- **Expand All:** Expands all constraints.
- **Group by Source:** Groups constraints based on the source file from which they originate.
- **Filter Constraints:** Filters constraints to show all constraints, valid constraints only, or invalid constraints only.
- **Edit Constraint:** Opens the Edit Create Clock dialog box for the selected constraint.
- **Remove Constraint:** Deletes the selected constraint.
- **Report Timing for This Constraint:** Opens the Report Timing dialog box to generate a timing report for the selected constraint.

Using the Waveform Window

For information on using the Waveform window, see this link in the *Vivado Design Suite User Guide: Logic Simulation* (UG900) and this link in the *Vivado Design Suite User Guide: Programming and Debugging* (UG908).

Using the Tcl Console

The Tcl Console (shown in the following figure) displays:

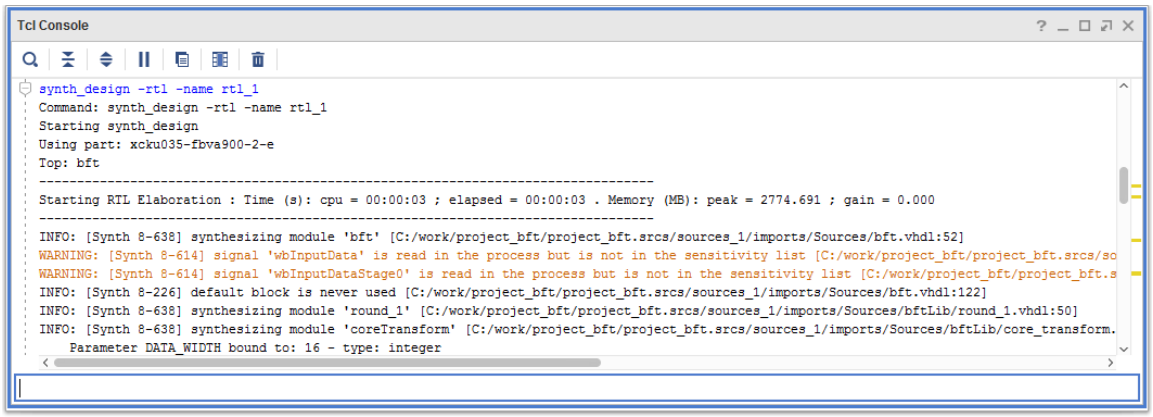
- Messages from previously executed Tcl commands.

  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 64: **Tcl Console**



Tcl Console Toolbar Commands

The local toolbar contains the following commands:

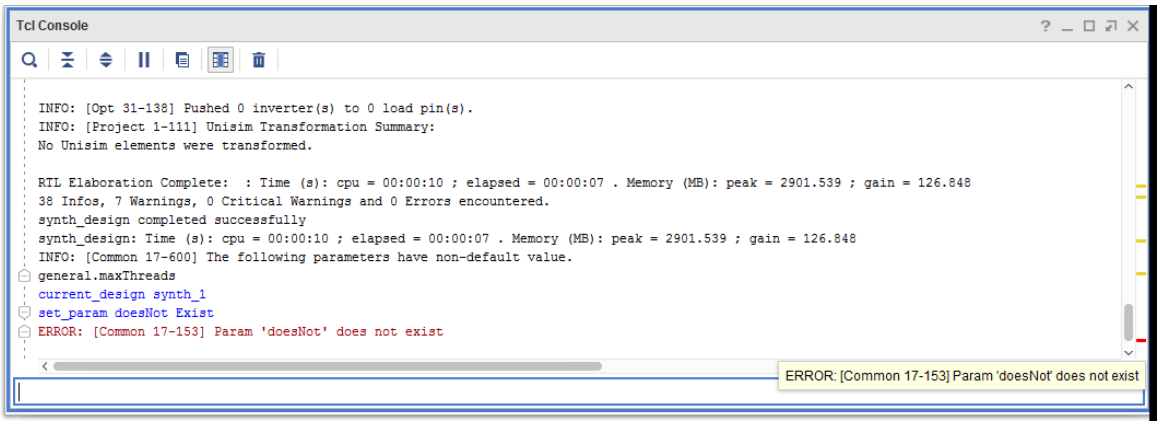
- **Find:** Finds text strings in the displayed messages.
- **Collapse All:** Collapses all constraints.
- **Expand All:** Expands all constraints.
- **Pause:** Allows you to scroll in the window or read reports as commands are running.
- **Toggle Column Selection Mode:** Toggles between selecting a block of text characters as a column or as lines of text.
- **Clear:** Clears all output in the Tcl Console.

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 the following figure. 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 65: **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.

IMPORTANT! *To get input from Tcl using the gets stdin command, you must launch the Vivado IDE from a command prompt, xterm, or Tcl shell. This ensures that the Vivado IDE does not hang while waiting for input. Alternatively, you can use the* ***Vivado <version> Tcl Shell*** *shortcut, and enter the* `start_gui` *command to open the Vivado IDE. When you use the gets stdin command from the Vivado IDE, the Vivado IDE reads the data entered in the command prompt, xterm, or Tcl shell.*

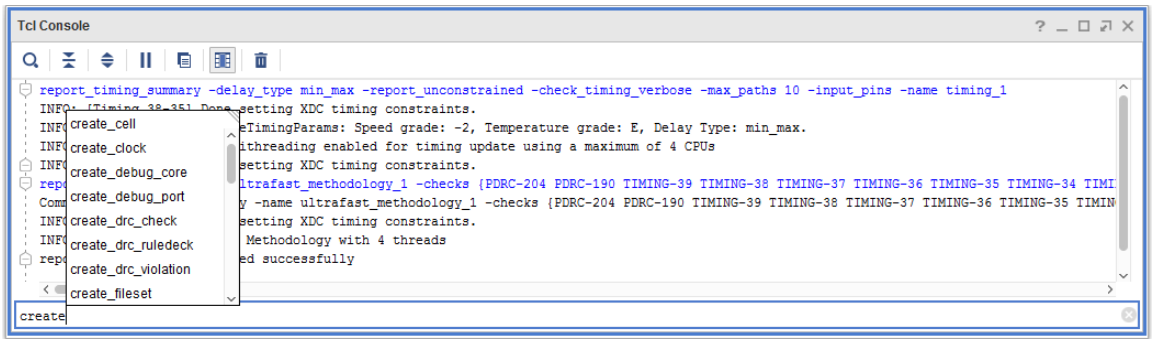
Using Auto-Complete

As you type commands, the Tcl Console auto-complete feature attempts to complete the name of the command or command parameters. For example, the following figure 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 66: **Auto-Completion of Tcl Commands**



Improving Readability

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. The Tcl Console shows both the Tcl command and a transcript of the results. From the Tcl Console, you can do the following:

- To show just the Tcl commands and hide the transcript, select the **Collapse all** toolbar button. Then, copy and paste the Tcl commands from the Tcl Console to create Tcl scripts.
- To show 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.

You can create Tcl scripts by copying commands from the journal file for later replay. To view the journal file, select **File → Project → Open Journal File**. You might need 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. For more information on journal files, see Output Files in Appendix B.

TIP: *If you want comments to appear in the journal file, enter the pound sign (#) followed by the comment in the Tcl Console. The Vivado IDE writes the comment to the journal file but does not execute it as a command. This is helpful when you want to take notes on the Tcl commands you entered.*

Related Information

Output Files

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). For a complete list of Tcl commands, refer to the *Vivado Design Suite Tcl Command Reference Guide* (UG835).

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 (shown in the following figure) displays design and report messages, which are grouped to enable you to locate messages from different tools or processes. Messages display with a link to the relevant object or source file. To open the Messages window, select **Window → Messages**. You can use the check boxes in the banner of the Messages window to hide or display the errors, critical warnings, warnings, info, and status messages.

***Note*:** If the location of a source file changes, the Vivado IDE removes the link from related messages to prevent confusion.

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 `get_msg_config -count -severity {CRITICAL WARNING}` Tcl command.*

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.*

Figure 67: **Messages Window**

Messages Window Toolbar Commands

The local toolbar contains the following commands:

- **Search:** Opens the search bar to allow you to locate messages.

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

- **Collapse All:** Collapses all constraints.

  ***Note*:** You can also access this command through the **Ctrl+-** keyboard shortcut.

- **Expand All:** Expands all constraints.

  ***Note*:** You can also access this command through the **Ctrl+=** keyboard shortcut.

- **Filter Messages:** Adjusts message filtering as follows:
  - **Suppression:**
    - **Show suppressed:** Displays only suppressed messages. To suppress messages, use the Manage Suppression popup menu command or `set_msg_config` Tcl command.

      ***Note*:** This setting results in an empty Messages window if no messages are suppressed.

    - **Show unsuppressed:** Displays only unsuppressed messages.

      ***Note*:** This is the default setting.

    - **Show both:** Displays both suppressed and unsuppressed messages. Suppressed messages have a backslash (\) through the severity icon.
  - **Severity:**
    - **Show modified:** Displays only messages with a modified severity. To modify message severity, use the **Message Severity** popup menu command or `set_msg_config -new_severity` Tcl command.

      ***Note*:** This setting results in an empty Messages window if no messages are modified.

    - **Show unmodified:** Displays only messages that have their original severity.

      ***Note*:** This is the default setting.

    - **Show both:** Displays messages with both modified and original severity. Messages with a modified severity have an asterisk (*) on the severity icon.

- **Manage Message Suppression:** Opens the Manage Suppression dialog box in which you can add or remove suppression rules. For more information, see Suppressing Messages and Unsuppressing Messages.

- **Discard User Created Messages:** Removes messages related to project load and analysis as well as messages output from scripts and Tcl commands entered in the Tcl Console.

  Note: You *cannot* use this command to clear messages output from design runs. Instead, use the Reset Runs popup menu command to reset the run and clear the messages for the run.

- **Settings:** Controls the display of information in the window.
  - **Group by File:** Groups messages by file.
  - **Group by ID:** Groups messages by message ID.
  - **Wrap Lines:** Wraps messages to the next line to fit the message to the width of the Messages window.

> **TIP:** *If the message limit is exceeded, a Message limit exceeded prompt appears. You can disable line wrapping to improve message display performance.*

Related Information

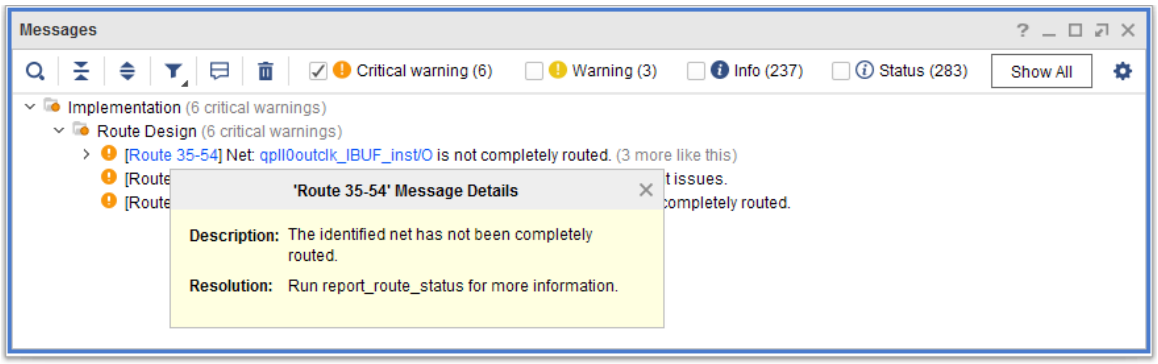
Suppressing Messages
Unsuppressing Messages

Viewing Message Details

When additional details are available for a message, a unique message ID appears as a link in the Messages window. Click the link to open a window (shown in the following figure) that contains a description and possible resolutions.

> **TIP:** *To search for Answer Records related to the message, right-click the message and select* ***Search for Answer Record****.*

Figure 68: **Message Details**



Cross Probing Message Objects

When design objects, such as cells, nets or pins, are displayed as a link (see the following figure), you can click the link to cross probe to different windows in the workspace. If necessary, Vivado IDE loads the design that contains the object.

TIP: *In a graphical window, you can use the Auto Fit Selection toolbar button to automatically zoom to the selected object. Alternatively, you can click* ***F9*** *to manually fit the selection.*

Figure 69: **Cross Probing Message Objects**

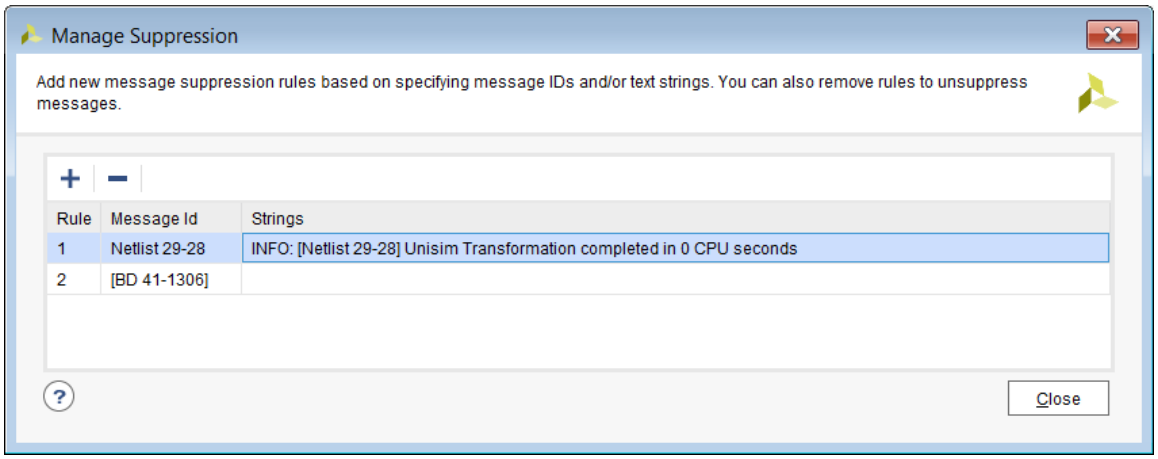
Suppressing Messages

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 all messages of a specific severity (for example, Info), right-click a message, and select **Suppress Messages with this Severity**.
- 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 (shown in the following figure), click the add button +. In the New Suppression Rule dialog box, enter the message ID or text string to add the suppression rule, and click **OK**.

TIP: *To limit the number of messages without suppressing them, you can use the* `set_msg_config` *Tcl command, for example,* `set_msg_config -id {[Common 17-349]} -limit 10`*. The default message limit is 100. For more information, see* `set_msg_config` *in the Vivado Design Suite Tcl Command Reference Guide (UG835).*

Figure 70: **Manage Suppression Dialog Box**



Unsuppressing Messages

To unsuppress messages, right-click in the Messages window, and select **Manage Suppression**. In the Manage Suppression dialog box (shown in the previous figure), select the rule to remove, and click the remove button .

***Note*:** Alternatively, you can click the **Manage Message Suppression** toolbar button to open the Manage Suppression dialog box.

TIP: *To temporarily display suppressed messages, click the **Filter Messages** toolbar button , and select **Show suppressed** or **Show both**. The message icon has a backslash (\) to indicate that the message is suppressed but displayed.*

Changing Message Severity

You can modify the severity for most messages, including:

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

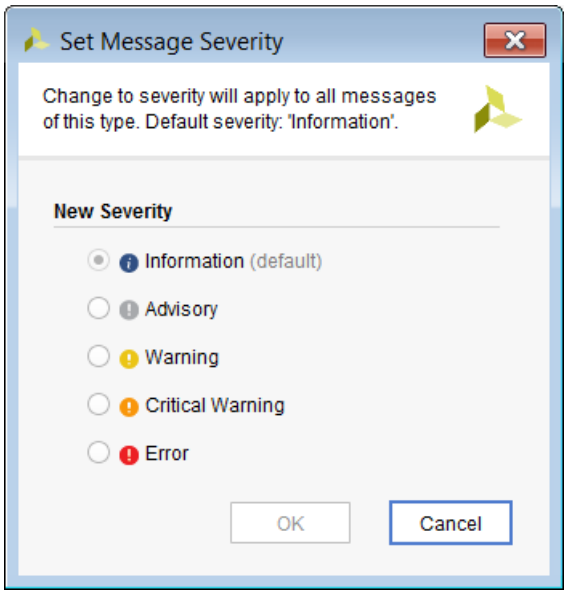
CAUTION! *You cannot demote error messages. Use caution when demoting critical warnings, because these messages flag problems that might result in errors later in the design flow.*

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 (shown in the following figure), 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 71: **Set Message Severity Dialog Box**



- Modified Information
- Modified Advisory
- Modified Warning
- Modified Critical Warning
- Modified Error

You can also modify message severity using the `set_msg_config` Tcl command. For example, the following Tcl command upgrades message ID Place 30-12 to a Critical Warning:

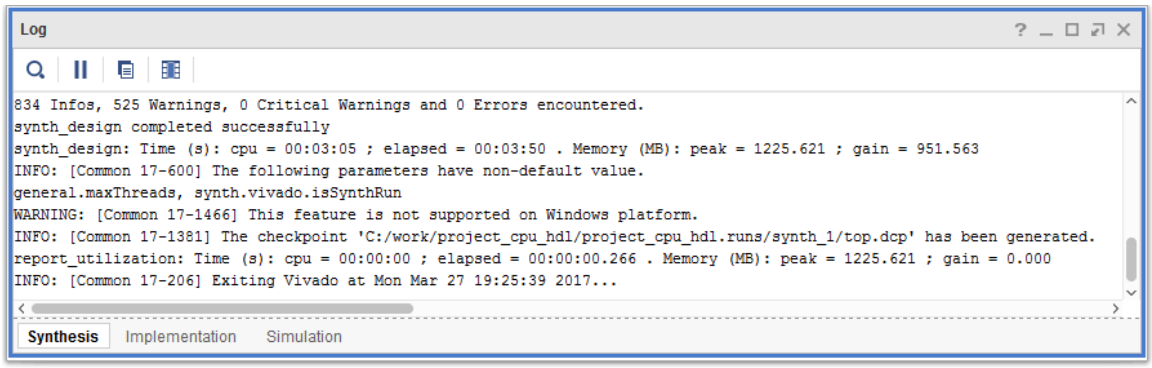
```
set_msg_config -id {Place 30-12} -new_severity {CRITICAL WARNING}
```

Using the Log Window

The Log window (shown in the following figure) 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. If the Log window is hidden, to open it, select **Window → Log**.

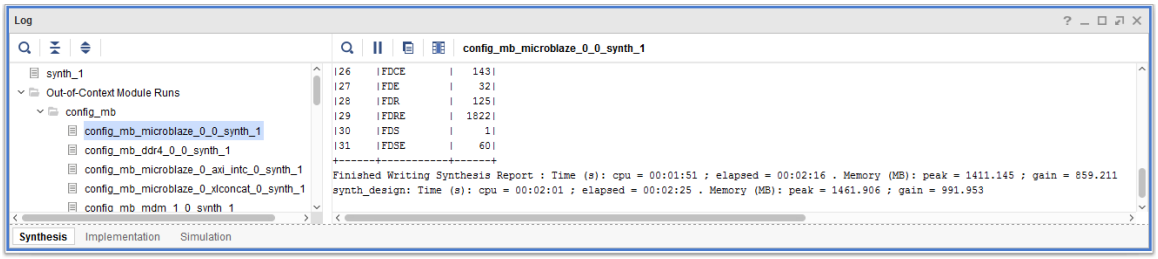
TIP: *You can click the **Pause** output toolbar button , and scroll back or read reports while commands are running.*

Figure 72: **Log Window**



When out-of-context synthesis runs are present, the Log window shows a tree view on the left and the log for the selected run on the right (as shown in the following figure).

Figure 73: **Log Window for Out-of-Context Synthesis Runs**



Using the Log Window Toolbar

The local toolbar contains the following commands:

- **Find:** Opens the search bar to allow you to locate text in the Log window.

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

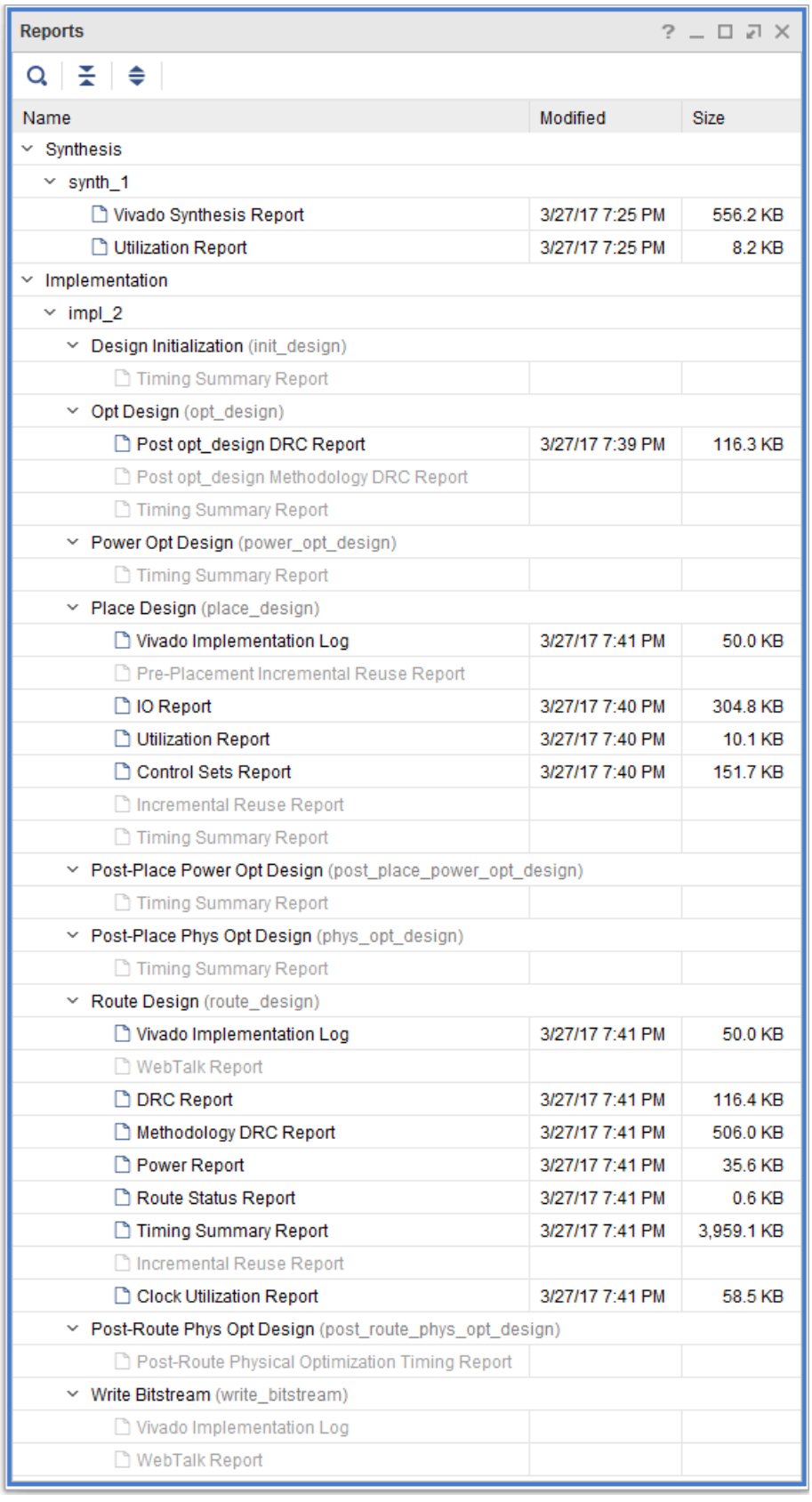
- **Pause Output:** Allows you to scroll in the window or read reports as commands are running.
- **Copy:** Copies selected text to the clipboard.
- **Toggle Column Selection Mode:** Specifies whether to select a block of text characters as a grid of rows and columns or as lines of text.

Using the Reports Window

The Reports window (shown in the following figure) 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 **Window → Reports**.

Note: If you close a custom report, you must recreate it. Custom reports are not stored in memory. If you close a report that was automatically generated as part of the design run, you can reopen the report by selecting **Reports → Open Interactive Report**.

Figure 74: **Reports Window**



Using the Reports Window Toolbar

The local toolbar contains the following commands:

- **Search:** Opens the search bar to allow you to locate text in the Reports window. 

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

- **Collapse All:** Collapses all reports. 

  ***Note*:** You can also access this command through the **Ctrl+-** keyboard shortcut.

- **Expand All:** Expands all reports. 

  ***Note*:** You can also access this command through the **Ctrl+=** keyboard shortcut.

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 Reports menu command, you can generate the following reports.

***Note*:** To identify common design issues, run Report Methodology the first time you synthesize the design. Run this report again when there is a change in constraints, clocking topologies, or large logic changes. For more information, see this link in the *Vivado Design Suite User Guide: System-Level Design Entry* (UG895).

- Report IP Status
- I/O
  - Report Noise
  - Report I/O
- Timing
  - Report Timing Summary
  - Report Bus Skew
  - Report Timing
  - Check Timing
  - Config Timing
  - Create Slack Histogram
  - Report Clock Interaction
  - Report CDC
  - Report Exceptions

    - Report Clock Networks
    - Report Pulse Width
    - Report Datasheet
- Report Methodology
- Report DRC
- Report QoR Suggestion
- Report QoR Assesment
- Report Utilization
- Report Clock Utilization
- Report Power
- Report Power Optimization
- Report High Fanout Nets
- Report Design Analysis

> **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.*

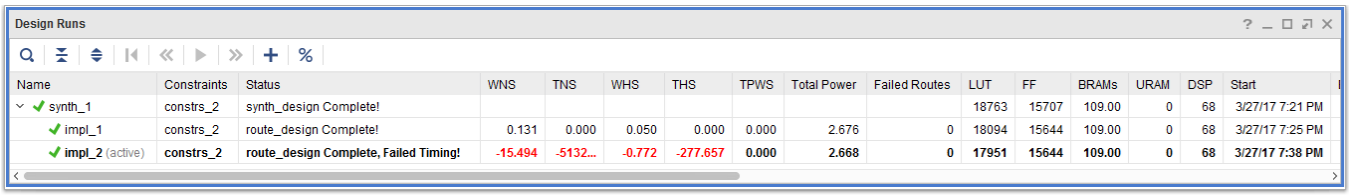
For more information on these reports, see the following documents:

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

Using the Design Runs Window

You can use the Design Runs window (shown in the following figure) to view, configure, launch, reset, and analyze synthesis and implementation runs. The Design Runs window lists synthesis and implementation runs in the order of the design flow and lists out-of-context runs based on dependency, for example, dependent IP appear under the top-level IP. To open the Design Runs window, select **Window → Design Runs**.

Figure 75: **Design Runs Window**



Design Runs Window Commands

The following commands are available from the local toolbar and popup menu.

Design Runs Window Toolbar Commands

The local toolbar contains the following commands:

- **Search:** Opens the search bar to allow you to locate messages.
- **Collapse All:** Collapses all messages.

  ***Note*:** You can also access this command through the **Ctrl+-** keyboard shortcut.
- **Expand All:** Expands all messages.

  ***Note*:** You can also access this command through the **Ctrl+=** keyboard shortcut.
- **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.
- **Reset to Previous Step:** Resets the selected run to the preceding step. This allows you to step backward through a run, make any needed changes, and then step forward to complete the run.
- **Launch Runs:** Invokes the Launch Runs dialog box to launch the selected runs.
- **Launch Next 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`.
- **Create Runs:** Invokes the Create New Runs wizard to create and configure new synthesis or implementation runs. For more information, see this link in the *Vivado Design Suite User Guide: Synthesis* (UG901) or this link in the *Vivado Design Suite User Guide: Implementation* (UG904).
- **Show Percentage:** Shows utilization by percentage rather than by number.

Design Runs Window Popup Commands

The popup menu contains the following commands:

- **Run Properties:** Displays the Run Properties window. For more information, 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 this link in the *Vivado Design Suite User Guide: Synthesis* (UG901) or this link in the *Vivado Design Suite User Guide: Implementation* (UG904).
- **Set Incremental Synthesis/Set Incremental Implementation:** Specifies a design checkpoint to use as a reference for the next synthesis or implementation run. For more information, see this link in the *Vivado Design Suite User Guide: Synthesis* (UG901) and this link in the *Vivado Design Suite User Guide: Implementation* (UG904).
- **Include Incremental Synthesis Information in DCP:** Includes information in the synthesis checkpoint for the tools to compare during the next incremental synthesis run.
- **Set QoR Suggestions:** Specifies a QoR suggestions file to use for the next synthesis or implementation run. For more information, see this link in the *Vivado Design Suite User Guide: Design Analysis and Closure Techniques* (UG906).
- **Generate ML Strategies:** Invokes the ML strategies window and is used to generate ML strategies for the designs that have not met timing. For more information, see this link in the *Vivado Design Suite User Guide:Design Analysis and Closure Techniques* (UG906).
- **Create ML Strategy Runs:** Creates different ML strategy implementation runs from the generated ML strategies. This option is available only after generating the ML strategies. For more information, see this link in the *Vivado Design Suite User Guide:Design Analysis and Closure Techniques*(UG906).
- **Save as Report 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 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:** 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`.
- **Launch Step To:** Launches the selected step for the selected run.
- **Reset to Previous Step:** Resets the selected run to the preceding step. This allows you to step backward through a run, make any needed changes, and 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 this link in the *Vivado Design Suite User Guide: Programming and Debugging* (UG908).
- **Display Log:** Displays the Log view of the Run Properties window.
- **Display Reports:** Displays the Reports view of the Run Properties window.
- **Display Messages:** Displays the Messages view 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.*

- **Create Runs:** Invokes the Create New Runs wizard to create and configure new synthesis or implementation runs. For more information, see this link in the *Vivado Design Suite User Guide: Synthesis* (UG901) or this link in the *Vivado Design Suite User Guide: Implementation* (UG904).
- **Open Run Directory:** Opens a file browser in the selected run directory on disk.
- **Export to Spreadsheet:** Exports the information in the Design Runs window to a spreadsheet file.

Related Information

Using the Run Properties 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
- Run queued
- 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.
- **Constraints:** Displays the constraint set used for the run.
- **Status:** Indicates run status as not started, running, complete, or error.
- **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.
- **Total Power:** Displays total on-chip power.
- **Failed Routes:** Displays the number of nets that failed to route, are partially routed, or have conflicts.
- **Methodology:** Displays the total number of critical warnings and warnings present in the design.
- **RQA Score:** Displays the RQA score for the design.
- **QoR Suggestions:** Displays the number of suggestions generated for the design.
- **LUT:** Displays the LUT utilization number or percentage.
- **FF:** Displays the flip-flop utilization number or percentage.
- **BRAMs:** Displays the block RAM utilization number or percentage.
- **URAM:** Displays the URAM utilization number or percentage.
- **DSP:** Displays the DSP utilization number or percentage.

> **TIP:** *By default, the Design Runs window shows utilization as a number. To show utilization as a percentage, click the Show Percentage toolbar button* %.

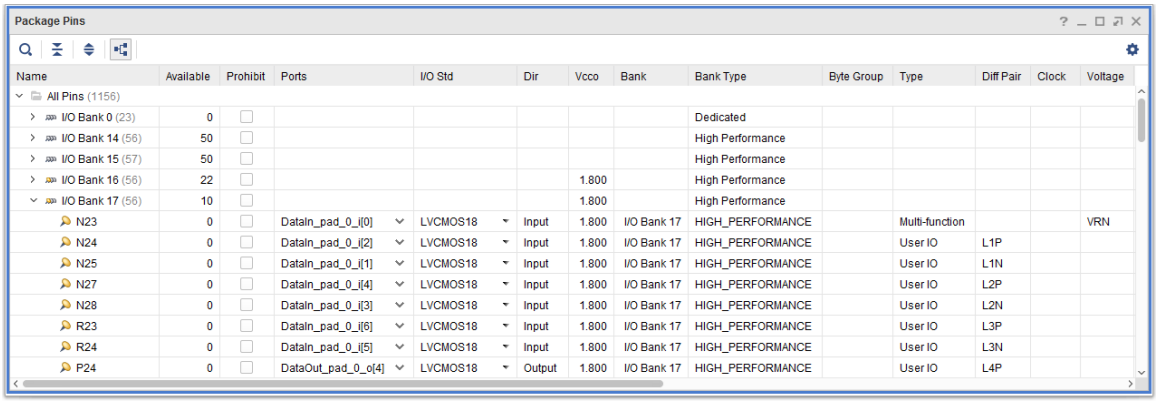
- **Start:** Reports the start time for the run.
- **Elapsed:** Reports the elapsed time for the run.
- **Strategy:** Displays the strategy assigned to the run. Strategies with an asterisk (*) indicate that some command options for the strategy have been overridden.
- **Part:** Indicates the target part selected for the run.
- **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.

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**.

Figure 76: **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.*

Related Information

Using Data Table Windows

Package Pins Window Toolbar Commands

The local toolbar contains the following commands:

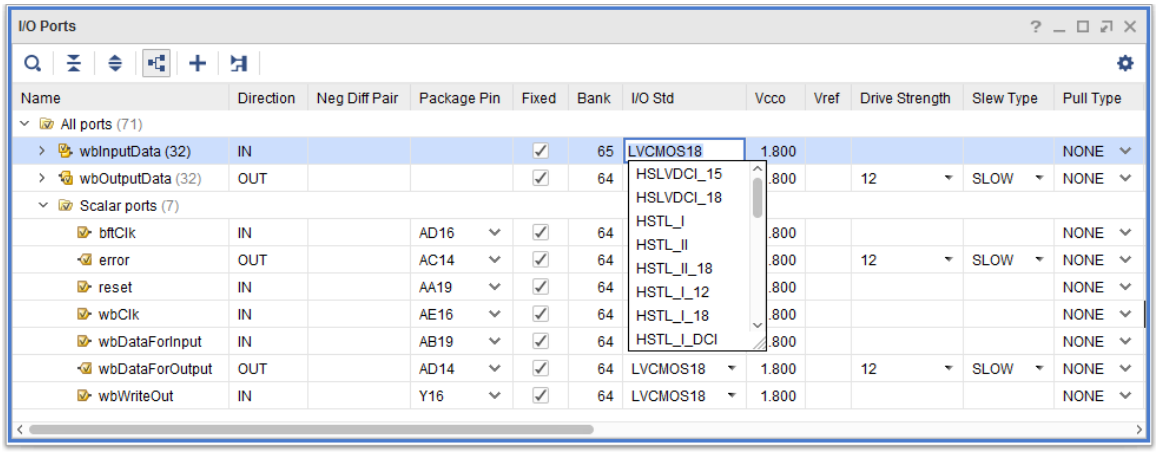
- **Search:** Searches the Package Pins window for ports by name, or by keywords or values within the various pin properties.
- **Collapse All:** Displays I/O Banks by name, and does not display individual pins of the bank.
- **Expand All:** Shows all pins of an I/O Bank expanded.
- **Group by I/O Bank:** Groups the pins by I/O Bank, or lists them alphabetically by name.
- **Settings:** Controls the display of information in the window.
  - **Scroll to Selected Objects:** Scrolls the Package Pins window to display objects selected in other windows like the Netlist or Device windows.

Using the I/O Ports Window

The I/O Ports window (shown in the following figure) 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 this link in the *Vivado Design Suite User Guide: I/O and Clock Planning* (UG899).

Figure 77: **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 platform board flow, see this link in the *Vivado Design Suite User Guide: System-Level Design Entry* (UG895).
- 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.

Related Information

Using Data Table Windows

I/O Ports Window Commands

The following commands are available from the local toolbar and popup menu.

I/O Ports Window Toolbar Commands

The local toolbar contains the following commands:

- **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. 

- **Settings:** Scrolls the I/O Ports window to display objects selected in other windows like the Netlist or Device windows.

I/O Ports Window Popup Menu

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. The popup menu contains the following commands:

- **I/O Port Bus Properties:** Allows you to view or edit properties for the selected object.
- **Delete:** Deletes the current selection.
- **Expand Selection:** Expands the selected rows.
- **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.
- **Assign to Interface:** Assigns a group of ports, port buses, or interfaces to a parent interface.
- **Unassign from Interface:** Unassigns a group of ports, port buses, or interfaces from their parent interface.
- **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.
- **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.
- **Unplace:** Unplaces the selected I/O ports.
- **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>]]`.

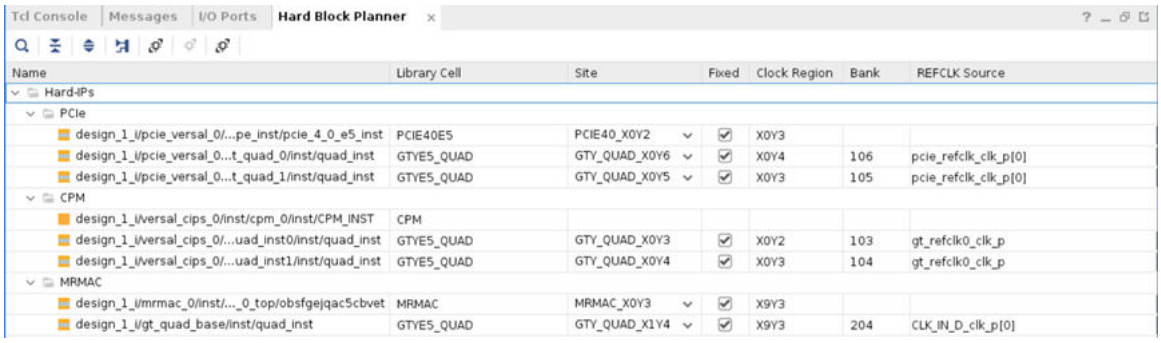
- **Unfix Ports:** Unfixes the selected placed I/O ports.
- **Swap Locations:** Swaps the sites for two selected ports.
- **Schematic:** Creates a schematic from the selected objects.
- **Highlight:** Highlights the selected objects.
- **Unhighlight:** Unhighlights the selected objects.
- **Mark:** Draws a marker for the selected object.
- **Unmark:** Removes the marker for the selected object.
- **Export I/O Ports:** Writes the contents of the I/O Ports window to a CSV, XDC, Verilog, or VHDL file.
- **Export to Spreadsheet:** Exports the information in the I/O Ports window to a spreadsheet file.

  Note: By default, the Vivado IDE exports seven levels of hierarchy to the Excel spreadsheet.

Using the Hard Block Planner Window

The Hard Block Planner window allows you to place the hard blocks within a device and provides a visual feedback in the Device window for assigning the location of the REFCLK pins, the GT_QUADs, and the Hard-IP blocks. Once you open the synthesized design, it reads and processes the netlist objects and collects all hard IPs available in a design. This planner allows you to cross-probe the location in the device window view for changing or assigning the site.

Figure 78: **Hard Block Planner Window**



To view the Hard Block Planner window, open Synthesized/Implemented design. The Hard Block Planner window opens by default after opening Synthesized/Implemented design. Alternately, it can be opened from the Windows menu

Hard Block Planner Window Columns

The Hard Block Planner window has the following columns:

- Name: Specifies the instance name of the hard block.
- Library Cell: Provides the library name of the hard block.
- Site: Provides the location of the library cell instance.
- Fixed: Specifies whether the BEL location is fixed or not.
- Clock Region: Specifies the clock region.
- Bank: Specifies the bank.
- REFCLK Source: Provides the clock name of the reference clock.

Hard Block Planner Window Tool Bar Commands

The local tool bar contains the following commands:

- **Search**: Opens the search bar to allow you to quickly locate objects in the hard block planner window.
- **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 hard block planner window.
- **Schematic**: Creates a schematic from the selected objects.
- **Show Hard-IP Connectivity**: Selects and displays all GTs and IPs of the design on the device view.
- **Show Hard-IP Connectivity for selected IP groups**: Selects and displays all GTs and IPs of the design on the device view for the selected IP Group.
- **Hide Hard-IP Connectivity**: Deselects all GTs and IPs of the design on the device view.

***Note*:** For more information on using the Hard Block Planner, see this link in the *Vivado Design Suite User Guide: I/O and Clock Planning* (UG899).

AMD XILINX

Chapter 4

Configuring the Environment

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. You can control these settings using the Tools and Layout menus as described in this chapter.

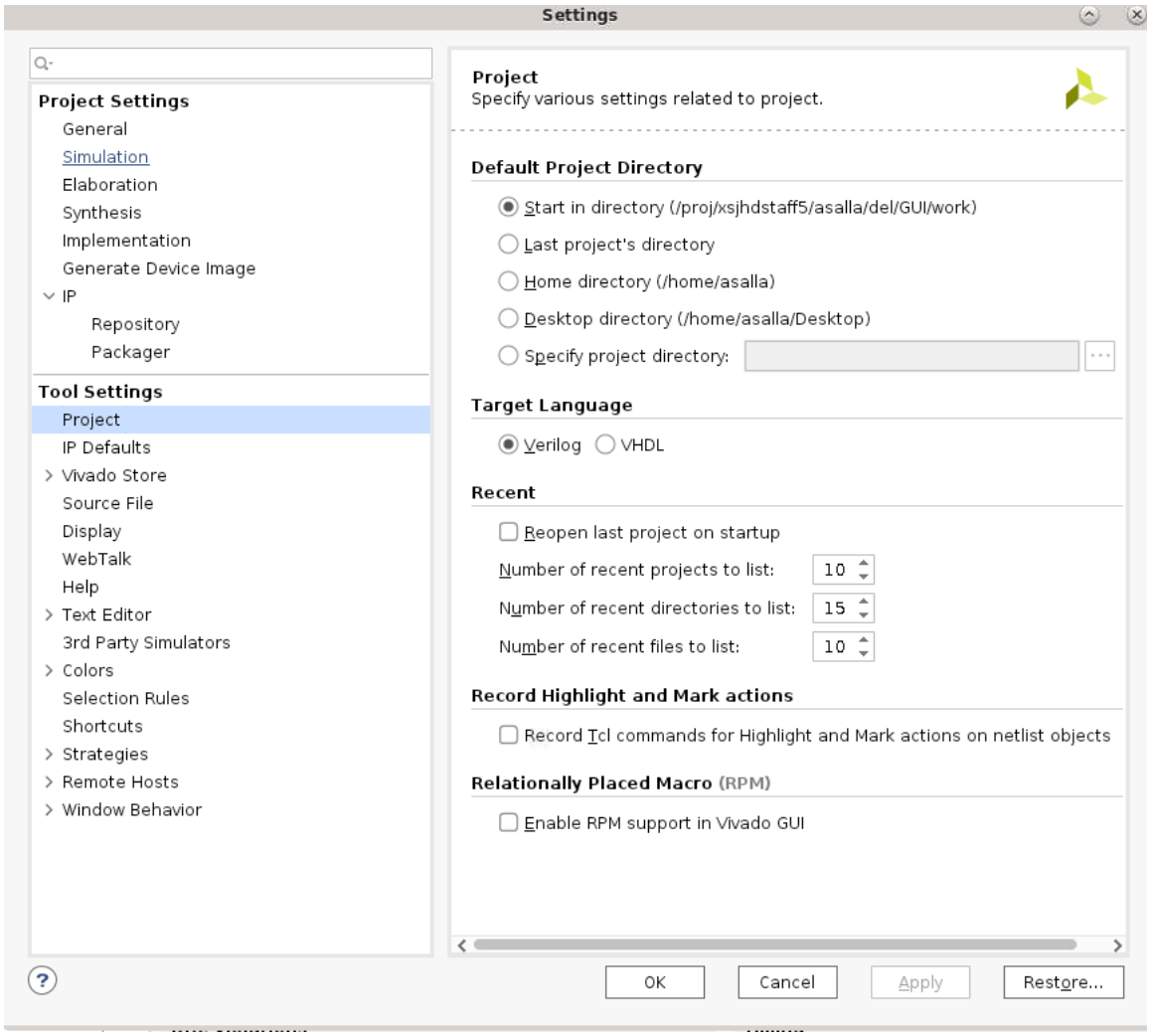
Specifying Tool Settings

You can use the Settings dialog box to specify both Project Settings and Tool Settings. Use the Project Settings to configure general, simulation, elaboration, synthesis, implementation, bitstream, and IP settings as described in Configuring Project Settings. Use the Tools Settings to configure the default behaviors in the Vivado IDE as described in the following sections.

To open the dialog box, select **Tools → Settings**. The dialog box changes based on the category you select in the left pane. For example, the following figure shows the Settings dialog box with the Project category selected under the Tools 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 79: **Settings Dialog Box—Tool Settings Project Category**



Related Information

Configuring Project Settings

Specifying Project Default Settings

You can use the Project default settings to control how the Vivado IDE handles projects. To open the Settings dialog box (shown in the previous figure), select **Tools → Settings**, and then click the **Project** category.

- **Default Project Directory:** Specifies the location where the Vivado IDE writes newly created projects.

- **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.
- **Record Tcl commands for highlight and mark actions:** Shows the corresponding commands in the Tcl Console when you mark or highlight objects, as described in Marking and Highlighting Objects.

Related Information

Marking and Highlighting Objects

Specifying IP Defaults

You can use the IP Defaults settings to specify the default IP directories and repositories. To open the Settings dialog box (shown in the previous figure), select **Tools → Settings**, and then click the **IP Defaults** category.

- **Default IP Example Project Directory:** Specifies where the Vivado IDE writes newly created IP examples. By default, the default project directory is used, or you can specify a different directory.
- **IP Catalog:** Specifies the default repository search paths and the order in which to search the repositories.

Specifying Vivado Store Settings

Configure Download Location: By default Vivado store is installed as follows:

- **Linux:** `<user home directory>/.Xilinx/Vivado/20xx.x/xhub/board_store/`
- **Windows:** `%APPDATA%\Roaming\Xilinx\20xx.x\xhub\board_store\`

The default location can be modified by configuring a custom path. The boards and the configurable example designs will be downloaded to the specified location.

The following are also found in the Vivado Store:

- **Board Repository**: An ordered list of repositories from which boards are discovered. If there are duplicate boards in separate repositories, the board is selected from the repository at the top first.

- **Example Project Repository**: An ordered list of repositories from which example designs are discovered. If there are duplicate example designs in separate repositories, the example design is selected from the repository at the top first.

Specifying Source File Settings

You can use the Source File settings to control how the Vivado IDE handles source files. To open the Settings dialog box (shown in the previous figure), select **Tools → Settings**, and then click the **Source File** category.

- **Source Files:** Specifies the default settings to use when adding sources to the project. You can create a local copy of the source files in the default project directory. You can also add source files from the sub directories in the default project directory.
- **File Saving:** Specifies whether the Vivado IDE saves project files automatically when closing or prompts you to save changes.

Specifying Display Settings

You can use the Display settings to control the appearance of the Vivado IDE. To open the Settings dialog box (shown in the previous figure), select **Tools → Settings**, and then click the **Display** category.

- **Scaling:** Sets the font scaling for the display, making the Vivado IDE easier to use on high resolution monitors. By default, this option is set to Use OS font scaling, which uses the value set for your primary monitor. Alternatively, you can select **User defined setting** to specify a value between 90% to 300% that is used by the Vivado tools only.
- **Spacing:** Sets the amount of space between elements, such as icons and text, in the Vivado IDE. Comfortable is the default setting, and Compact reduces the amount of space between elements to fit more elements into a smaller space.
- **Messages:** Specifies the maximum number of messages to display.
- **Hyperlinks:** Specifies the action to take when you click a path delay link in a timing report, including whether to provide a description, display a menu, or select a cell pin, cell, or site.

Specifying Help Settings

You can use the Help settings to control how tooltips, Quick Help, and documentation display in the Vivado IDE. To open the Settings dialog box (shown in the previous figure), select **Tools → Settings**, and then click the **Help** category.

- **Tooltips and Quick Help:** Specifies the language for tooltips and Quick Help. You can also set the amount of time before a tooltip appears and disappears.
- **Documentation:** Specifies whether to open documents in Xilinx® Documentation Navigator or in your default web browser.

***Note*:** For more information on Documentation Navigator, see Documentation Navigator and Design Hubs.

Related Information

Documentation Navigator and Design Hubs

Specifying Text Editor Settings

You can use the Text Editor settings to specify the text editor used by the Vivado IDE. If you select the Vivado Text Editor, you can also customize various options, which are not supported for third-party text editors. To open the Settings dialog box (shown in the previous figure), select **ToolsSettings**, and then click the **Text Editor** category.

- **Current Editor:** Sets the text editor used by the Vivado IDE. If you select the Vivado Text Editor, the following Vivado Editor General Settings and Vivado Editor Display Settings are also available.
- **Vivado Editor General Settings:** Sets options that control code folding, line comments, column selection shortcuts, split views, the number of undo operations, and the number of recent files.
- **Vivado Editor Display Settings:** Sets options that control the display of the file path at the top of the Vivado Text Editor, line numbers, and matches for the selected word.

In addition, if you select the **Vivado Text Editor**, the following subcategories are available in the left pane of the Settings dialog box:

- **Code Completion:** Sets the preference for activating the code completion drop-down, whether using a shortcut key, displaying as you type, or disabling code completion. You can also set whether to use the **Tab** key or **Space** bar to select the displayed value.

- **Syntax Checking:** Enables syntax checking and specifies whether to display warnings and notes. You can also specify the formatting for errors, warnings, and notes and view the formatting in the Preview window.
- **Tabs:** Specifies whether to indent the line, whether to use the tab character instead of spaces, and how many spaces to use per tab.
- **Fonts and Colors:** Specifies a display theme, which is a group of color settings. You can choose from a default theme or create your own, as described in Setting Display Themes. You can also set the font style, size, and color for standard text (foreground) as well as the color for the background, line highlighting, and matching words. You can view this formatting in the Preview window.

  ***Note*:** You can also set the text styles and colors for different languages using the Verilog, VHDL, Tcl, Xdc, and Trigger State Machine subcategories.

Related Information

Setting Display Themes

Specifying Third-Party Simulators

You can use the 3rd Party Simulators settings to specify install paths and default compiled library paths. To open the Settings dialog box (shown in the previous figure), select **Tools → Settings**, and then click the **3rd Party Simulators** category.

- **Install Paths:** Specifies the path to a third-party simulator.
- **GCC Instal Paths:**

  Specifies the directory to locate GCC installation for each supported Simulator. This option is required if you want to use the GCC path rather than the path specified in the tool installation.
- **Default Compiled Library Paths:** Specifies the path to the compiled libraries for each supported simulator. The compiled library path is only applied when you create a new project.

***Note*:** For more information on third-party simulators, see this link in the *Vivado Design Suite User Guide: Logic Simulation* (UG900).

Specifying Colors

You can use the Colors settings to control the appearance of the viewing environment. To open the Settings dialog box (shown in the previous figure), select **Tools → Settings**, and then click the **Colors** category. Click the subcategories to set colors for effects like highlighting and for objects in different windows, such as the Device and Package windows.

Setting Display Themes

A display theme is a group of color settings that you can change using the Themes drop-down menu. The Vivado IDE provides predefined light and dark background themes. To change the theme, select the **Vivado Light Theme** or **Vivado Default Theme** from the drop-down menu.

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

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

Related Information

Outputs for Environment Configuration

Creating a Custom Display Theme

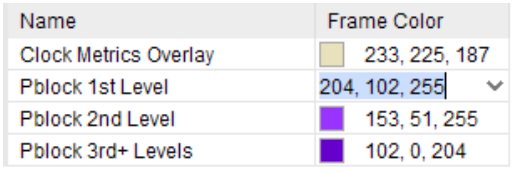
After you configure the color settings, you can save the settings for future use. Click the **Save As** button to specify a name and save your custom display theme.

Changing Colors

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

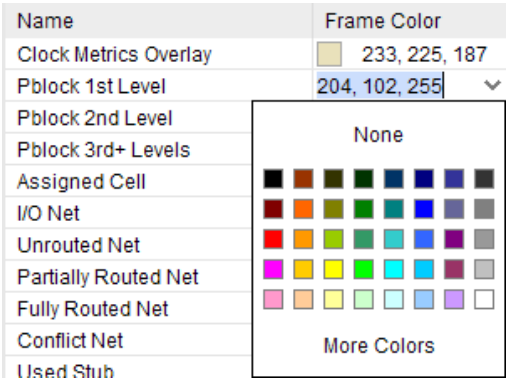
- Click a color cell, and enter an RGB value, as shown in the following figure.

Figure 80: **RGB Color Values**



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

Figure 81: **Color Palette**



TIP: *You can also modify color settings in the Device window. For more information, see Specifying Device Window Settings.*

Related Information

Specifying Device Window Settings

Setting Selection Rules

You can use the Selection Rules settings to control the secondary elements that are selected when you select a primary object. 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. To open the Settings dialog box, select **Tools → Settings**, and then click the **Selection Rules** category.

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** check box 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 settings to create custom shortcuts. To open the Settings dialog box, select **Tools → Settings**, and then click the **Shortcuts** category.

To create and delete custom shortcuts:

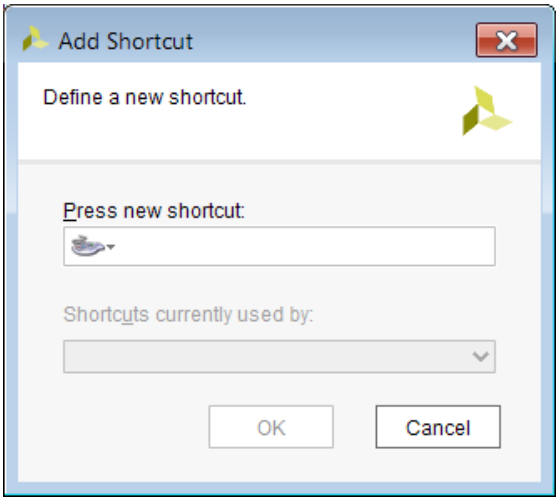
1. In the Settings dialog box, click **Copy** to create a new schema from the Vivado Default schema.

   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. Click **Add**.
5. In the Add Shortcut dialog box (shown in the following figure), select the new shortcut from the drop-down menu, 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 B.

6. To delete a shortcut, select a command, and click **Remove**.

***Note*:** To show the currently defined shortcuts, deselect the **Group by Usage toolbar** button , and double-click the Shortcut column heading.

Related Information

Outputs for Environment Configuration

Creating Run Strategies

A run strategy is a set of pre-configured command line options for the synthesis or implementation tools designed to resolve synthesis or implementation challenges in the design. The Vivado IDE provides several commonly used strategies that are tested against internal benchmarks. Strategies are tool and version specific.

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. You can use the Run Strategies settings to create custom strategies. To open the Settings dialog box, select **Tools → Settings**, and then click the **Strategies → Run Strategies** category.

TIP: *You can also use the Strategies settings to view the command line options associated with the predefined Vivado IDE synthesis and implementation strategies.*

The Vivado IDE writes the user-defined run strategies to:

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

To review, copy, and modify strategies:

1. From the Flow drop-down menu, select a Vivado Synthesis or Vivado Implementation version.

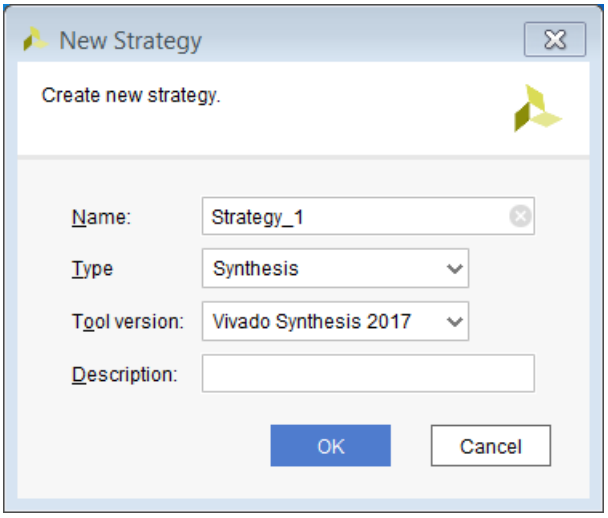
   A list of strategies and related command line options appear. For more information on command line options, see the *Vivado Design Suite User Guide: Synthesis* (UG901) and *Vivado Design Suite User Guide: Implementation* (UG904).

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

   Note: Alternatively, you can create a copy of an existing strategy using **Copy Strategy** from the popup menu. 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 (shown in the following figure), set the following options, and click **OK**:
   - **Name:** Specifies the strategy name.
   - **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 Runs window.



4. Edit the options for the command line tools used during the synthesis or implementation run as follows:
   a. Click the check box to enable or disable an option.
   b. Select a different value from the drop-down menu.
   c. 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.

Creating Report Strategies

A report strategy is a set of reports created during synthesis or implementation. The Vivado IDE provides several reports that can be created at various steps of the design runs.

You cannot change the command line settings for predefined report strategies. However, you can copy and modify supplied strategies to create your own custom strategies. This includes disabling the default reports as well as running the same report during various points in the design flow.

You can use the Report Strategies settings to create custom report strategies. To open the Settings dialog box, select **Tools → Settings**, and then click the **Strategies → Report Strategies** category.

The Vivado IDE writes the user-defined report strategies to:

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

To review, copy, and modify strategies:

1. From the Flow drop-down menu, select a Vivado Synthesis or Vivado Implementation version.

   A list of strategies and related reports appear. For more information on reports, see this link in the *Vivado Design Suite User Guide: Design Analysis and Closure Techniques* (UG906).

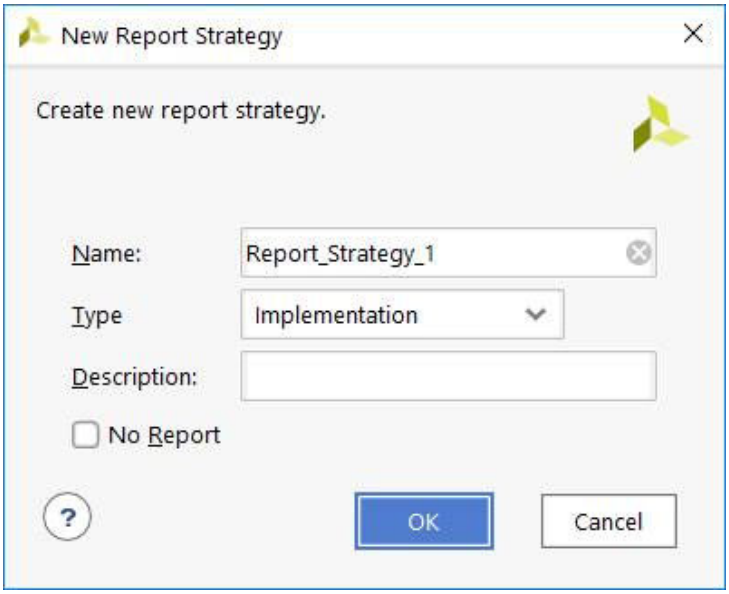
   IMPORTANT! *You cannot modify the default options for predefined Vivado IDE strategies. To customize strategies, you must copy or add a strategy.*

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

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

3. In the New Report Strategy dialog box (shown in the following figure), set the following options, and click **OK**:

   - **Name:** Specifies the strategy name.
   - **Type:** Specifies whether the strategy applies to synthesis or implementation.
   - **Description:** Provides the strategy description for display in the Design Runs window.
   - **No Report:** Creates a strategy without generating reports.



4. In the Reports field, edit the list of reports used during the synthesis or implementation run as follows:
   - To add a report, select **Add Report** from the popup menu or toolbar. In the Add Report for Report Strategy dialog box, specify the Run Step and Report Type, and click **OK**.

     Note: The Run Step is only available for implementation runs.
   - To remove a report, click the report under the desired design step, and select **Remove Report** from the popup menu or toolbar.
   - To modify report options, click the report, and modify the options in the Options field.

   TIP: *To get syntax examples to use for the MORE_OPTIONS values, see the Summary section of an existing report. You can also type the Tcl command for the report (shown in parenthesis after the report name) followed by -help in the Tcl Console to get a complete list of command line options.*
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.

Customizing Window Behavior

You can use the Window Behavior settings to control how warnings, confirmations, notifications, and alerts display in the Vivado IDE. To open the Settings dialog box, select **Tools → Settings**, and then click the **Window Behavior** category.

Click the following subcategories to specify these settings:

- **Warnings:** Defines how the Vivado IDE shows warning dialog boxes, such as warnings when closing a design or project.
- **Confirmations:** Defines how the Vivado IDE shows confirmation dialog boxes, such as confirmations when switching to a different design.
- **Notifications:** Defines how the Vivado IDE shows notifications, such as notifications when synthesis or implementation completes successfully.
- **Alerts:** Defines how the Vivado IDE shows alerts, such as alerts for the success of non-active runs.

Configuring 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.
- **Undo:** Undoes the most recent view manipulation.
- **Redo:** Redoes the most recent view manipulation.
- **Reset Layout:** Restores resized or moved windows to the original configuration for the active layout.

***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.

TIP: *You can also use predefined layouts as described in Layout Selector.*

Related Information

Outputs for Environment Configuration

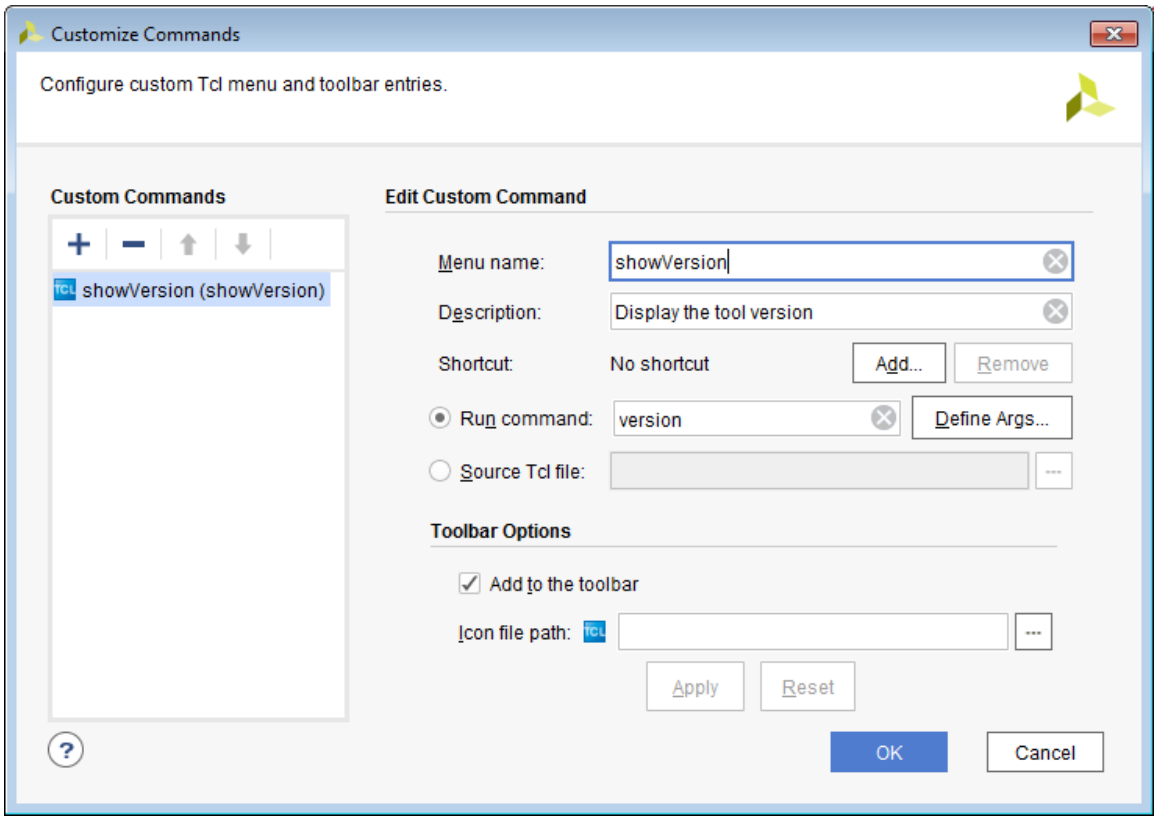
Adding Custom Menu Commands

You can use the Customize Commands dialog box (shown in the following figure) 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**. The custom commands are available from the **Tools → Custom Commands** menu, with each command listed on the submenu.

***Note*:** The Customize 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.

TIP: *You can also use the `create_gui_custom_command` Tcl command to add custom menu commands. For details, see Adding Custom Menu Commands Using a Tcl Command.*

Figure 82: **Customize Commands Dialog Box**



You can specify the following options:

- **Custom Commands:**
  - **Add:** Adds new commands 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:** Removes the selected commands from the custom menu.
  - **Move Up:** Moves the selected command up in the list.
  - **Move Down:** Moves the selected command down in the list.
- **Edit Custom Command:** Specifies the properties for the selected command in the Custom Commands list.
  - **Menu Name:** Specifies the name of the custom command.
  - **Description:** Specifies the text to display in the status bar when hovering over the menu command.

- **Shortcut:** Defines a keyboard shortcut for the custom command. Click **Add** to open the Add Shortcut dialog box, and select the new shortcut from the drop-down menu. To delete a shortcut, click **Remove**.
- **Run Command:** Runs the specified Tcl command or procedure for the custom command.
- **Source Tcl File:** Sources the specified Tcl script file for the custom command rather than running a single Tcl command or procedure.
- **Toolbar Options:** Specifies whether to add a toolbar button icon for the custom command to the main toolbar.
  - **Add to the Toolbar:** Enables the toolbar icon. When this check box is disabled, the custom command does not appear on the main toolbar.
  - **Icon File Path:** Specifies the file path to the toolbar button icon. Use a PNG, JPG, or GIF file of approximately 20x20 pixels. The Vivado IDE resizes larger images to fit onto the toolbar.

Related Information

Outputs for Environment Configuration
Adding Custom Menu Commands Using a Tcl Command

Adding Custom Menu Commands Using a Tcl Command

You can also add custom menu commands using the `create_gui_custom_command` Tcl command. Enter the command in the Tcl Console of the Vivado IDE or source the command from a Tcl file. Following is an example:

```
create_gui_custom_command -name showVersion -menu_name showVersion -
description
“Display the tool version” -show_on_toolbar -command version
```

AMD XILINX

Appendix A

Vivado IDE Tips

Using the Viewing Environment

Table 1: **Using the Viewing Environment**

To	Do This
Show the currently defined shortcuts	Select **Tools** → **Settings**. In the Settings dialog box, click the **Shortcuts** category. Deselect the **Group by Usage** toolbar button, and double-click the **Shortcut** column heading.
Set display preferences for tooltips	Select **Tools** → **Settings**. In the Settings dialog box, click the **Help** category, and specify the Tooltips and Quick Help settings.

Using Windows

Table 2: **Using Windows**

To	Do This
Show or hide the Flow Navigator	Press **Ctrl+Q**.
Make the next tab active in the workspace	Press **Ctrl+Tab**.
Make the previous tab active in the workspace	Press **Ctrl+Shift+Tab**.
Maximize or minimize the window	Double-click the window tab or press **Alt -**.
Reset the window layout	Press **F5**.
Locate specific text	Press **Ctrl+F**.
Replace specific text	Press **Ctrl+R**.

Working with Messages and Reports

Table 3: **Working with Messages and Reports**

To	Do This
See only one message type	In the banner of the Messages window, use the check boxes next to the message types, or double-click the message type you want to view.
Show only suppressed messages	In the Messages window, click the Filter Messages toolbar button, and select **Show Suppressed**. ***Note*:** This setting results in an empty Messages window if no messages are suppressed.

Table 3: **Working with Messages and Reports** *(cont'd)*

To	Do This
Show only messages with modified severity	In the Messages window, click the Filter Messages toolbar button, and select **Show Modified**. ***Note*:** This setting results in an empty Messages window if no messages are modified.
Read reports while commands are running in the Log window	In the Messages window, click the **Pause Output** toolbar button.
Tile reports horizontally or vertically	Right-click a report tab, and select **New Horizontal Group** or **New Vertical Group**.
Reopen a graphical report window automatically created as part of a design run	Select **Reports** → **Open Interactive Report**.

AMD XILINX

Appendix B

Input and Output Files

Input Files

Table 4: **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 window 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 to exchange information with board designers 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. ***Note*:** NGC format files are not supported in the Vivado Design Suite for UltraScale™ devices. Xilinx recommends that you regenerate the IP using the Vivado Design Suite IP customization tools with native output products. Alternatively, you can use the NGC2EDIF command to migrate the NGC file to EDIF format for importing, as described in the ISE to *ISE to Vivado Design Suite Migration Guide* (UG911). However, Xilinx recommends using native Vivado IP rather than XST-generated NGC format files going forward.

Table 4: **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.
Xilinx IP and IP Integrator Block Designs	• XCI • XCIX • BD	• The Vivado IDE supports importing configured Xilinx IP by using the IP XCI file or the core container XCIX file. • The Vivado IDE supports importing a BD file containing a block design created with Vivado IP integrator.
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 this link in the *Vivado Design Suite Tcl Command Reference Guide* (UG835).

RECOMMENDED: *You can open the Vivado IDE from any directory. However, Xilinx recommends running the Vivado IDE from a project directory, because the 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. Alternatively, you can launch the Vivado IDE by double-clicking the* **project** *file (`.xpr` extension) to ensure that the log and journal files are written to the project directory.*

The following table lists the Vivado IDE output files, including files types and descriptions.

Table 5: **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** → **Project** → **Open Log File** from the main menu.

Table 5: **Vivado IDE Output Files** *(cont'd)*

Output File	File Type	Description
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** → **Project** → **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 the journal 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.
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** → **Project** → **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 the journal 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 5: **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>.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. • 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)
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 Report Noise 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 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>`

The following table lists the environment configuration files. input files, including files names and descriptions.

Table 6: **Vivado IDE Environment Configuration Outputs**

Environment Configuration File	File Name	Description

Tool-Specific Tcl Initialization Script	`Vivado_init.tcl`
	
	Note: In Vivado Design Suite 2016.4 and earlier releases, the file name is `init.tcl`.
		• Software installation directory: `installdir/Vivado/version/scripts/Vivado_init.tcl`
		• Local user directory:
		  ◦ Windows: `%APPDATA%/Xilinx/Vivado/Vivado_init.tcl`
		  ◦ Linux: `$HOME/.Xilinx/Vivado/Vivado_init.tcl`
		• You can create the file in the software installation directory to share a common initialization script across a set of users.
		• You can create the file in the local user directory to specify additional commands or override commands.
		Note: For more information, see this link in *Vivado Design Suite Tcl Command Reference Guide* (UG835).

View Display Options	vivado.xml	• `../vivado/<version>/vivado.xml` file captures the settings in **Tools** → **Settings** that include display color and other viewing options for the environment. • The Vivado IDE saves your settings to the vivado.xml 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 6: **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.

Related Information

Configuring the Environment

Outputs for Project Data

The following table lists the Vivado IDE project data output, including file names and descriptions.

Table 7: **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.

Table 7: **Vivado IDE Project Data Outputs** *(cont'd)*

Project Data Output	File Name	Description
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.
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 (by default, named `constrs_2`, `constrs_3`, etc.).
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.
Project Generated Output Products Directory	`<projectname>.gen`	An additional project directory called `project.gen` is automatically created in the project directory. This directory stores all output products created by the IP and Block Diagram(BD). The result of this change is that the `project.srcs` directory contains only the sources used to create the design and `project.gen` has output products.

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.*

The following table lists the file/directory name, simulation type, and description for the simulation runs.

Table 8: **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

The following table 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 9: **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`, etc. by default).
Run Implementation and Launch Vivado IDE Runs	• `<name>.rpt` • `<name>.rpx` • `<name>.pb` • `<top>_<step>.dcp` • `runme.log`	• Reports generated as part of the run appear as `<name>.rpt` and `<name>.rpx`. You can view `<name>.rpt` files in the Vivado IDE Text Editor and load `<name>.rpx` files in the Vivado IDE. • 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`.

Table 9: **Outputs For Implementation** *(cont'd)*

Output File	File Name	Description
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.

AMD XILINX

Appendix C

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.

Documentation Navigator and Design Hubs

Xilinx® Documentation Navigator (DocNav) provides access to Xilinx documents, videos, and support resources, which you can filter and search to find information. To open DocNav:

- From the Vivado® IDE, select **Help → Documentation and Tutorials**.
- On Windows, select **Start → All Programs → Xilinx Design Tools → DocNav**.
- At the Linux command prompt, enter `docnav`.

Xilinx Design Hubs provide links to documentation organized by design tasks and other topics, which you can use to learn key concepts and address frequently asked questions. To access the Design Hubs:

- In DocNav, click the **Design Hubs View** tab.
- On the Xilinx website, see the Design Hubs page.

***Note*:** For more information on DocNav, see the Documentation Navigator page on the Xilinx website.

References

These documents provide supplemental material useful with this guide:

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: Using Tcl Scripting* (UG894)
4. *Vivado Design Suite Tcl Command Reference Guide* (UG835)
5. *Vivado Design Suite Tutorial: Design Flows Overview* (UG888)
6. *Vivado Design Suite User Guide: Getting Started* (UG910)
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 Properties Reference Guide* (UG912)
17. *Vivado Design Suite User Guide: I/O and Clock Planning* (UG899)
18. *ISE to Vivado Design Suite Migration Guide* (UG911)
19. *Vivado Design Suite User Guide: Model-Based DSP Design Using System Generator* (UG897)
20. *Vivado Design Suite User Guide: Power Analysis and Optimization* (UG907)
21. Vivado Design Suite Documentation

Training Resources

Xilinx provides a variety of training courses and QuickTake videos to help you learn more about the concepts presented in this document. Use these links to explore related training resources:

1. Designing FPGAs Using the Vivado Design Suite 1 Training Course
2. Designing FPGAs Using the Vivado Design Suite 2 Training Course
3. Designing FPGAs Using the Vivado Design Suite 3 Training Course
4. Designing FPGAs Using the Vivado Design Suite 4 Training Course
5. Vivado Design Suite QuickTake Video: Introduction to the Xilinx Tcl Store
6. Vivado Design Suite QuickTake Video: Using the Timing Constraints Wizard
7. Vivado Design Suite QuickTake Video: Understanding Messaging
8. Vivado Design Suite QuickTake Video Tutorials

Revision History

The following table shows the revision history for this document.

Section	Revision Summary
04/27/2022 Version 2022.1	
Navigating Content by Design Process	Added new section.
Using the Getting Started Page	Updated the section.
Creating Projects	Updated screen capture.
Finding Unplaced BUFGs	Updated screen capture.
Configuring Project Settings	Updated the section.
Using the Netlist Window	Updated screen capture.
Understanding Run Status	Updated the section.
Syntax Checking	Updated the section.
Specifying IP Defaults	Updated the section.
Specifying IP Defaults	Added new section.
Specifying Third-Party Simulators	Updated the section.
Using the Hard Block Planner Window	Added new section.
General	Moved revision history table.

Please Read: Important Legal Notices

PARTICULAR PURPOSE; and (2) Xilinx shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under, or in connection with, the Materials (including your use of the Materials), including for any direct, indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or Xilinx had been advised of the possibility of the same. Xilinx assumes no obligation to correct any errors contained in the Materials or to notify you of updates to the Materials or to product specifications. You may not reproduce, modify, distribute, or publicly display the Materials without prior written consent. Certain products are subject to the terms and conditions of Xilinx's limited warranty, please refer to Xilinx's Terms of Sale which can be viewed at https://www.xilinx.com/legal.htm#tos; IP cores may be subject to warranty and support terms contained in a license issued to you by Xilinx. Xilinx products are not designed or intended to be fail-safe or for use in any application requiring fail-safe performance; you assume sole risk and liability for use of Xilinx products in such critical applications, please refer to Xilinx's Terms of Sale which can be viewed at https://www.xilinx.com/legal.htm#tos.

AUTOMOTIVE APPLICATIONS DISCLAIMER

AUTOMOTIVE PRODUCTS (IDENTIFIED AS "XA" IN THE PART NUMBER) ARE NOT WARRANTED FOR USE IN THE DEPLOYMENT OF AIRBAGS OR FOR USE IN APPLICATIONS THAT AFFECT CONTROL OF A VEHICLE ("SAFETY APPLICATION") UNLESS THERE IS A SAFETY CONCEPT OR REDUNDANCY FEATURE CONSISTENT WITH THE ISO 26262 AUTOMOTIVE SAFETY STANDARD ("SAFETY DESIGN"). CUSTOMER SHALL, PRIOR TO USING OR DISTRIBUTING ANY SYSTEMS THAT INCORPORATE PRODUCTS, THOROUGHLY TEST SUCH SYSTEMS FOR SAFETY PURPOSES. USE OF PRODUCTS IN A SAFETY APPLICATION WITHOUT A SAFETY DESIGN IS FULLY AT THE RISK OF CUSTOMER, SUBJECT ONLY TO APPLICABLE LAWS AND REGULATIONS GOVERNING LIMITATIONS ON PRODUCT LIABILITY.

Copyright