Vivado Design Suite User Guide

I/O and Clock Planning

UG899 (v2013.2) June 19, 2013

XILINX®

Notice of Disclaimer
The information disclosed to you hereunder (the "Materials") is provided solely for the selection and use of Xilinx products. To the maximum extent permitted by applicable law: (1) Materials are made available "AS IS" and with all faults, Xilinx hereby DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY 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 the Limited Warranties which can be viewed at http://www.xilinx.com/warranty.htm; 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 Critical Applications: http://www.xilinx.com/warranty.htm#critapps.

© Copyright 2012-2013 Xilinx, Inc. Xilinx, the Xilinx logo, Artix, ISE, Kintex, Spartan, Virtex, Vivado, Zynq, and other designated brands included herein are trademarks of Xilinx in the United States and other countries. All other trademarks are the property of their respective owners.

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
04/10/2013	2013.1	Updated Overview and added Netlist-Based I/O Pin Planning in Chapter 2. Rearranged sections, added Tcl command information, added Setting Device Constraints, added Setting the Configuration Bank Voltage Select (CFGBVS) Pin, updated Defining and Configuring I/O Ports, updated Clock Planning, updated Running DRCs, updated Working with SSN Analysis, updated Working with IBIS Models, added Interfacing with the PCB Design, added I/O Pin Planning and Clock Planning with SSI Technology Devices, and removed Examining Clock Connections sections in Chapter 3, I/O Pin Planning. Removed Appendix B, I/O Port DRC Rule Descriptions. Updated Figure 2-1, Figure 2-2, Figure 2-3, Figure 3-1, Figure 3-2, Figure 3-3, Figure 3-4, Figure 3-5, Figure 3-6, Figure 3-21, Figure 3-30, Figure 3-33, and Figure 3-34.
06/19/2013	2013.2	Added recommendation to run DRC and SSN analysis after synthesis and after implementation in Chapter 2, I/O Pin Planning Design Flow. Added recommendation to complete clock planning prior to pinout selection in Clock Planning, added Tcl Command for Placing Ports into I/O Banks, added Phase option to Running SSN Analysis, and added Adding Temperature Information to SSN Analysis in Chapter 3, I/O Pin Planning. Updated Figure 3-1 and Figure 3-32.

XILINX®

Table of Contents

Revision History 2

Chapter 1: Introduction

Overview 5
I/O Planning Stages 5
I/O Planning View Layout 8

Chapter 2: I/O Pin Planning Design Flow

Overview 9
Pre-RTL I/O Pin Planning and Device Exploration 9
RTL-Based I/O Pin Planning 13
Netlist-Based I/O Pin Planning 16

Chapter 3: I/O Pin Planning

Using the I/O Planning View Layout 18
Viewing Device Resources 19
Defining Alternate Compatible Parts 23
Setting Device Configuration Modes 25
Setting Device Constraints 27
Setting the Configuration Bank Voltage Select (CFGBVS) Pin 31
Defining and Configuring I/O Ports 31
Clock Planning 40
Placing I/O Ports 43
Running DRCs 50
Migrating to an RTL Design 55
Working with SSN Analysis 56
Exporting I/O Pin and Package Data 61
Working with IBIS Models 62
Interfacing with the PCB Design 64
I/O Pin Planning and Clock Planning with SSI Technology Devices 65

Appendix A: I/O Port Lists in CSV File Format

CSV File 66

Differential Pairs in the CSV File 68

Appendix B: Additional Resources

Xilinx Resources 69
Solution Centers 69
References 69

Introduction

Overview

I/O pin planning is a process that includes printed circuit board (PCB) designers, field programmable gate array (FPGA) designers, and system designers; each with their own specific set of concerns and requirements. Often, designers are hindered by a non-optimal pinout that causes further delays when trying to meet timing and signal integrity requirements. By considering the data flow from PCB to FPGA die, you can achieve optimal pinout configurations quickly, thus reducing internal and external trace lengths as well as routing congestion. This chapter provides an overview of the I/O planning process using the graphical user interface (GUI) known as the Vivado® Integrated Design Environment (IDE).

I/O Planning Stages

The Vivado IDE facilitates I/O planning at different stages of the design process. As the design progresses, more information becomes available, enabling more complex rule checking as the design is synthesized and implemented. The type of work you can do in each step of the design process varies. For example, early in the process, some data is missing; therefore, analysis is an estimate only. Later in the process, additional data is available; therefore, analysis is more accurate.

In the Vivado tools, you can start I/O planning with an empty project, move to register-transfer level (RTL) source files and synthesized netlist, and finally work in an implemented design. Working with an implemented design is the final validation of an I/O pin and clock configuration. Proper clock resource validation requires full implementation of all clocks.

Proper I/O assignment can depend on how the clocks are configured; assigning I/Os and clock logic often go together. For the I/O placement design rule checks (DRCs) to account for clocks, you must have a synthesized design. Whenever possible, it is best to perform I/O assignment with a synthesized design.

Following is an overview of the I/O planning stages:

1. **Create an I/O planning project.**

   You can create an empty project to enable early device exploration and I/O port configuration. This allows for early pinout definition to eliminate pinout-related changes that typically happen late in the design cycle. When creating the project, you can create I/O ports manually or import them from comma separated values (CSV) files or Xilinx® design constraints (XDC) files. After the project is created, you can:

   - Export device and I/O port assignments for use later in the design process.
   - Migrate an I/O planning project to an RTL project when the port definitions and pin assignments are resolved.

2. **Elaborate and check RTL source files.**

   You can perform I/O planning in an RTL project. In an elaborated design, the tool provides basic DRCs.

RECOMMENDED: *To check clock logic, validation with a synthesized design is recommended.*

3. **Synthesize the design.**

   After synthesis, you can perform I/O planning in the synthesized design. Because all clocks are determined and the Vivado IDE has visibility into all clocks, a more thorough validation is performed. Whenever possible, perform I/O assignment using a synthesized design.

4. **Implement the design and conduct final I/O validation.**

   The design must be fully implemented to ensure a legal I/O pinout. Examine the implementation reports for I/O and clock-related messages.

Table 1-1 shows the features supported at each I/O planning stage.

Table 1-1: **I/O Planning Stages and Features**

Feature	I/O Planning Project	RTL Design	Synthesized Design	Implemented Design
Read Ports from CSV and XDC Files	Supported	N/A	N/A	N/A
Create or Delete Ports	Supported	N/A	N/A	N/A
Migrate to RTL Project	Supported	N/A	N/A	N/A
Set Part Compatibility	Supported	Supported	Supported	Supported
Set Configuration Modes	Supported	Supported	Supported	Supported
Base I/O Standard DRC	Supported	Supported	Supported	Supported
Simultaneous Switching Noise (SSN) Analysis	Supported	Supported	Supported	Supported
Clock-Aware Placement and Clock-Aware DRC	N/A	N/A	Supported	Supported
Final Sign-Off DRC	N/A	N/A	N/A	Supported

For detailed information on these stages, see Chapter 2, I/O Pin Planning Design Flow. For more information on working with projects, such as creating an empty I/O planning project or elaborating an RTL design, see the *Vivado Design Suite User Guide: System-Level Design Entry* (UG895) [Ref 1].

I/O Planning View Layout

In the Vivado IDE, the I/O Planning view layout provides an interface to:

- Create, import, and configure the initial list of I/O ports early in the design flow.
- Perform final verification of the pinout at the end of the design flow.
- Group related ports into interfaces, then assign them to package pins.
- Use fully automatic pin placement or semi-automatic interactive modes for controlled I/O port assignment.
- View the relationship of the physical package pins and banks with their corresponding I/O die pads.
- Make intelligent decisions to optimize the connectivity between the PCB and the FPGA device.
- Analyze the design and device I/O requirements.
- Define an I/O pinout configuration or *pinout* that satisfies the requirements of both PCB and FPGA designs.

For detailed information on the I/O Planning view layout, see Chapter 3, I/O Pin Planning.

I/O Pin Planning Design Flow

Overview

Using the Vivado® IDE, you can work on I/O pin planning at any stage in the design flow. Following are the most commonly used methods:

- **Pre-RTL I/O Pin Planning and Device Exploration**: In this flow, you do not have a netlist, and you are working on initial I/O and board planning. You can perform I/O exploration and assignment with an I/O planning project even before the design source files are available. You can import a CSV file for I/O planning or export it for use in PCB schematic symbol generation or hardware description language (HDL) header generation.
- **RTL-Based I/O Pin Planning**: In this flow, you have an RTL design available with access to the Vivado IP catalog. The IP catalog enables you to generate connectivity IP, customize clocking components using the clocking wizard, and configure SelectIO™ interface resources using the SelectIO Interface Wizard. At this stage, you can do initial planning with the elaborated design and then export the CSV file for use in PCB schematic symbol generation. This stage is also useful for planning a new FPGA design.
- **Netlist-Based I/O Pin Planning**: In this flow, the design is synthesized or implemented, and you are working with a synthesized or implemented netlist for I/O and board planning. The tools have more information about the design at this stage, and you can use the automatic placement command or the semi-automatic interactive modes to control I/O port assignment. You can also use the I/O Planning layout view to see the relationship of the physical package pins and banks with corresponding I/O die pads. This enables you to make intelligent decisions to optimize the connectivity between the PCB and the FPGA device.

Pre-RTL I/O Pin Planning and Device Exploration

1. **Create an I/O planning project using the New Project wizard.**

   For information on creating an I/O planning project, see the *Vivado Design Suite User Guide: System-Level Design Entry* (UG895) [Ref 1].

2. **Import CSV or XDC files.**

   For information, see Defining and Configuring I/O Ports in Chapter 3.

3. **Select the device.**

   Determine the device size based on resource needs. Select the package based on PCB requirements, such as critical routes to memories. For designs using stacked silicon interconnect (SSI) technology, see the *Large FPGA Methodology Guide* (UG872) [Ref 2].

   Note: In addition to the selected device, you can also identify alternate compatible parts, as described in Defining Alternate Compatible Parts in Chapter 3.

4. **Select configuration, digitally controlled impedance (DCI) cascade, and internal voltage reference (V_{REF}).**

   For information, see Setting Device Configuration Modes and Setting Device Constraints in Chapter 3.

5. **Configure I/O ports.**

   For information, see Configuring I/O Ports in Chapter 3. Also, see the following resources available from the Xilinx® website or Xilinx Documentation Navigator:

   - *7 Series FPGAs Packaging and Pinout Product Specification* (UG475) [Ref 3] and *Zynq-7000 All Programmable SoC Packaging and Pinout Specifications* (UG865) [Ref 4] provide information on the packaging and pinout specifications for 7 series and Zynq® device families.
   - *7 Series FPGAs SelectIO Resources User Guide* (UG471) [Ref 5] provides information on banking rules. For example, some I/O standards can be combined within a single bank and some cannot.

6. **Optionally, manually place I/O ports.**

   For information, see Placing I/O Ports in Chapter 3.

7. **Optionally, migrate to an RTL project.**

   You can migrate the I/O port assignments made in the I/O planning project to an RTL project. For more information, see Migrating to an RTL Design in Chapter 3.

   a. **Define MIG, GT, and connectivity IP.**

      You can use the Vivado IP Catalog to define memory interface generator (MIG), gigabit transceiver (GT), and connectivity intellectual property (IP). For more information on working with IP, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 6].

Note: Some IP, such as Ethernet IP and PCI Express® (PCIe) technology IP, has specific pinout requirements. In addition, high speed memory interfaces have specific pinout requirements driven by clocking and skew needs.

IMPORTANT: *After generating MIG IP, you must add the resulting XDC file to your project. In some cases, generating GT and connectivity IP also results in an XDC file that you must add to your project.*

b. **Define major clock structures.**

For information, see Clock Planning in Chapter 3. For information on achieving timing closure, see the *Vivado Design Suite User Guide: Design Analysis and Closure Techniques* (UG906) [Ref 7].

RECOMMENDED: *It is recommended that you use the Clocking Wizard in the Vivado IP catalog to generate mixed-mode clock manager (MMCM) or phase-locked loop (PLL) modules to define clock connections. For information, see the LogiCORE IP Clocking Wizard Product Guide (PG065) [Ref 8].*

c. **Manually place I/O ports.**

For information, see Placing I/O Ports in Chapter 3.

d. **Run synthesis and implementation.**

For information, see the *Vivado Design Suite User Guide: Synthesis* (UG901) [Ref 9] and *Vivado Design Suite User Guide: Implementation* (UG904) [Ref 10].

RECOMMENDED: *It is recommended that you run DRCs and SSN analysis after synthesis, prior to implementation, as well as after implementation. This enables you to catch issues early in the design cycle.*

8. **Run DRCs.**

For information, see Running DRCs in Chapter 3.

9. **Run SSN analysis.**

For information, see Working with SSN Analysis in Chapter 3.

10. **Start board planning.**

Following are considerations when board planning:

- For board-level validation, perform signal integrity analysis using I/O buffer information specification (IBIS) or HSPICE models. For information, see Exporting IBIS Models in Chapter 3.
- To optimize the pinout within the context of the entire board, import the FPGA into third-party products, such as Cadence Allegro FPGA System Planner or Mentor Graphics I/O Designer.

Figure 2-1 shows the pre-RTL I/O pin planning flow using an I/O planning project.

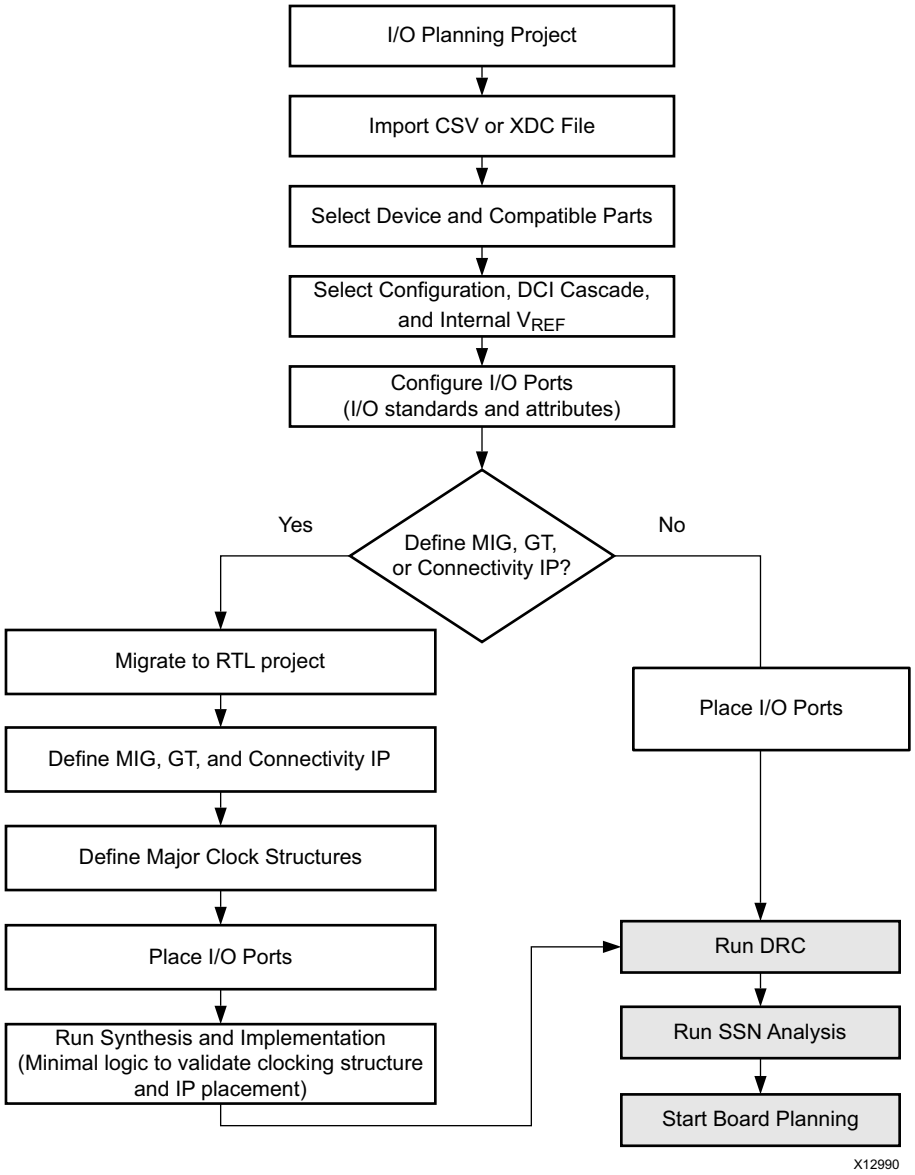

Figure 2-1: **Pre-RTL I/O Pin Planning and Device Exploration**

RTL-Based I/O Pin Planning

1. **Create an RTL project using the New Project wizard.**

   For information on creating an RTL project, see the *Vivado Design Suite User Guide: System-Level Design Entry* (UG895) [Ref 1].

2. **Select the device.**

   Determine the device size based on resource needs. Select the package based on PCB requirements, such as critical routes to memories. For designs using SSI technology, see the *Large FPGA Methodology Guide* (UG872) [Ref 2].

   Note: In addition to the selected device, you can also identify alternate compatible parts, as described in Defining Alternate Compatible Parts in Chapter 3.

3. **Select configuration, DCI cascade, and internal V_{REF}.**

   For information, see Setting Device Configuration Modes and Setting Device Constraints in Chapter 3

4. **Configure I/O ports.**

   For information, see Configuring I/O Ports in Chapter 3. Also, see the following resources available from the Xilinx website or Xilinx Documentation Navigator:

   - *7 Series FPGAs Packaging and Pinout Product Specification* (UG475) [Ref 3] and *Zynq-7000 All Programmable SoC Packaging and Pinout Specifications* (UG865) [Ref 4] provide information on the packaging and pinout specifications for 7 series and Zynq device families.
   - *7 Series FPGAs SelectIO Resources User Guide* (UG471) [Ref 5] provides information on banking rules. For example, some I/O standards can be combined within a single bank and some cannot.

5. **Define MIG, GT, and connectivity IP.**

   You can use the IP Catalog to define MIG, GT, and connectivity IP. For more information on working with IP, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 6].

   Note: Some IP, such as Ethernet IP and PCI Express® (PCIe) technology IP, has specific pinout requirements. In addition, high speed memory interfaces have specific pinout requirements driven by clocking and skew needs.

IMPORTANT: *After generating MIG IP, you must add the resulting XDC file to your project. In some cases, generating GT and connectivity IP also results in an XDC file that you must add to your project.*

6. **Define major clock structures.**

   For information, see Clock Planning in Chapter 3. For information on achieving timing closure, see the *Vivado Design Suite User Guide: Design Analysis and Closure Techniques* (UG906) [Ref 7].

RECOMMENDED: *It is recommended that you use the Clocking Wizard in the Vivado IP catalog to generate MMCM or PLL modules to define clock connections. For information, see the LogiCORE IP Clocking Wizard Product Guide (PG065) [Ref 8].*

7. **Place I/O ports.**

   You can place I/O ports using either of the following methods:

   - **Run synthesis and auto-place I/O ports.**

     For information, see the *Vivado Design Suite User Guide: Synthesis* (UG901) [Ref 9] and Automatically Placing I/O Ports in Chapter 3.

   - **Manually place I/O ports.**

     For information, see Placing I/O Ports in Chapter 3.

8. **Run DRCs.**

   For information, see Running DRCs in Chapter 3.

9. **Run SSN analysis.**

   For information, see Working with SSN Analysis in Chapter 3.

10. **Start board planning.**

    Following are considerations when board planning:

    - For board-level validation, perform signal integrity analysis using I/O buffer information specification (IBIS) or HSPICE models. For information, see Exporting IBIS Models in Chapter 3.
    - To optimize the pinout within the context of the entire board, import the FPGA into third-party products, such as Cadence Allegro FPGA System Planner or Mentor Graphics I/O Designer.

Figure 2-2 shows the RTL-based I/O pin planning flow using an RTL project.

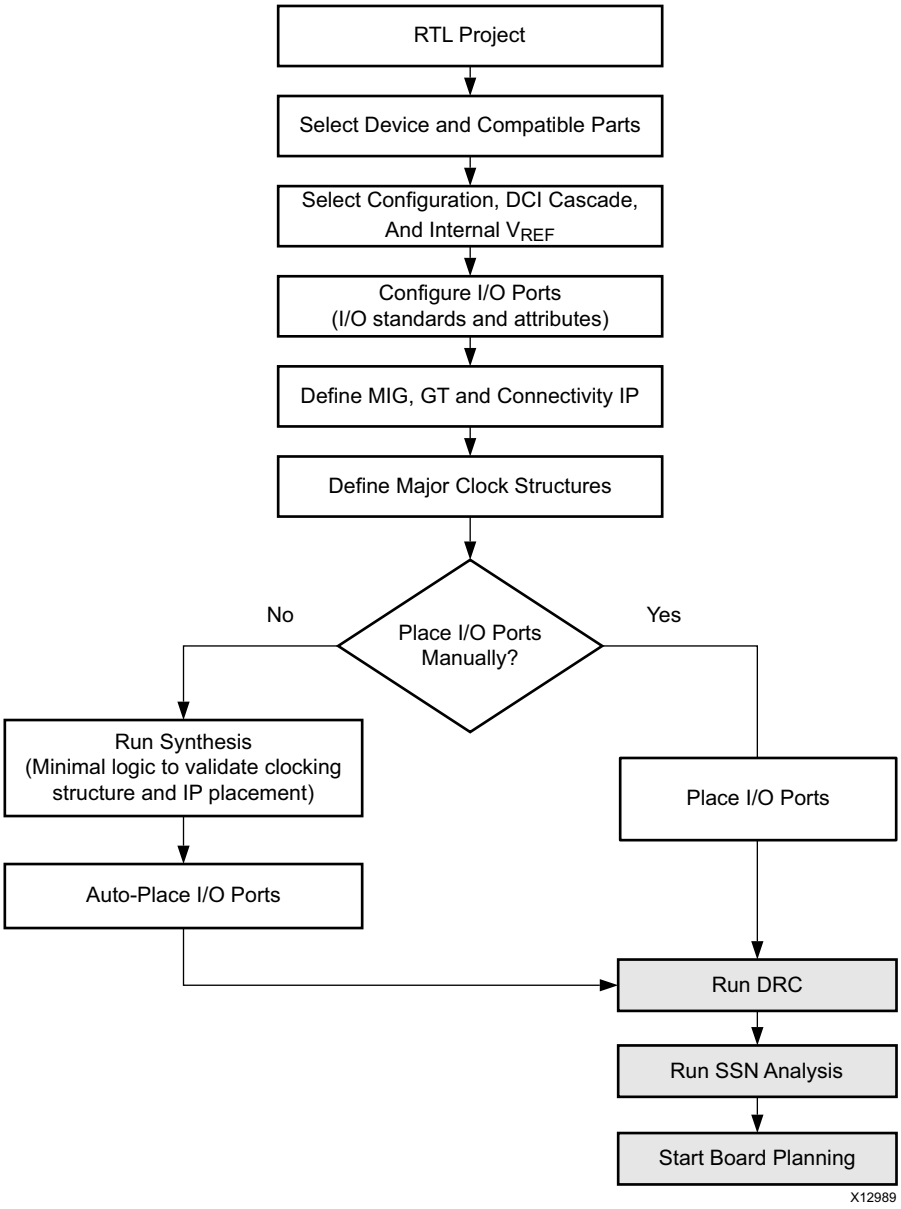

Figure 2-2: **RTL-Based I/O Pin Planning**

Netlist-Based I/O Pin Planning

1. **Create a project using the New Project wizard.**

   For information on creating a post-synthesis project, see the *Vivado Design Suite User Guide: System-Level Design Entry* (UG895) [Ref 1].

2. **Place I/O ports.**

   Open the synthesized or implemented design, and place I/O ports using either of the following methods:

   - **Auto-place I/O ports.**

     For information, see Automatically Placing I/O Ports in Chapter 3.

   - **Manually place I/O ports.**

     For information, see Placing I/O Ports in Chapter 3.

3. **Run DRCs.**

   For information, see Running DRCs in Chapter 3.

4. **Run SSN analysis.**

   For information, see Working with SSN Analysis in Chapter 3.

5. **Start board planning.**

   Following are considerations when board planning:

   - For board-level validation, perform signal integrity analysis using I/O buffer information specification (IBIS) or HSPICE models. For information, see Exporting IBIS Models in Chapter 3.
   - To optimize the pinout within the context of the entire board, import the FPGA into third-party products, such as Cadence Allegro FPGA System Planner or Mentor Graphics I/O Designer.

Figure 2-3 shows the netlist-based I/O pin planning flow using a synthesized or implemented netlist.

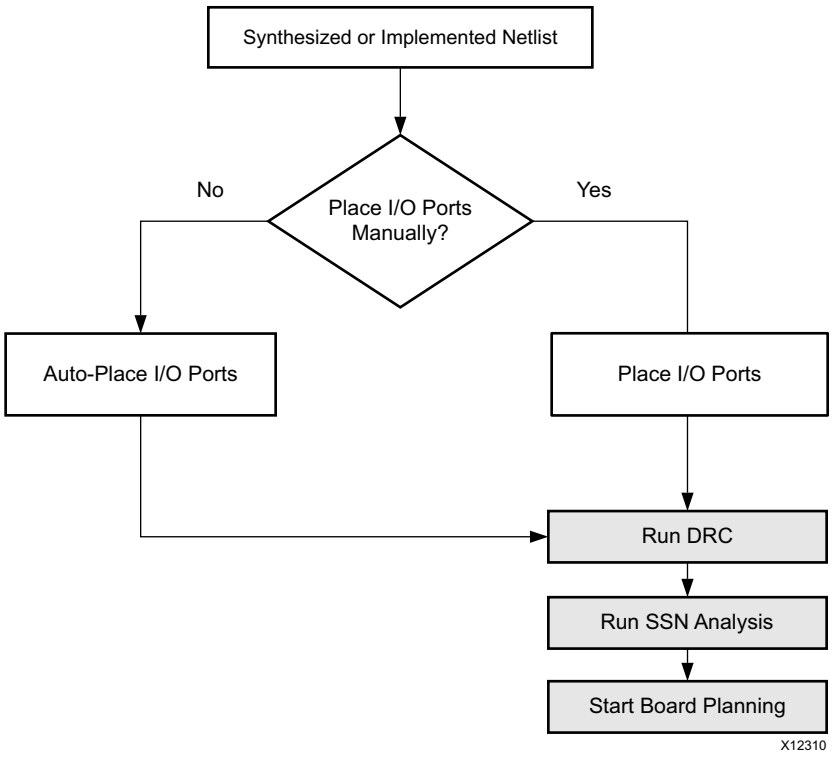

Figure 2-3: **Netlist-Based I/O Pin Planning**

I/O Pin Planning

Using the I/O Planning View Layout

In the Vivado® IDE, you can use the I/O Planning view layout on elaborated, synthesized, and implemented designs. The view layout uses both Device and Package windows. Additional I/O information appears in the following windows: Clock Resources, Clock Regions, Package Pins, I/O Ports, Device Constraints, and Properties windows.

Note: For more information on the windows in the Vivado IDE, see the *Vivado Design Suite User Guide: Using the Vivado IDE* (UG893) [Ref 11].

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

To launch the I/O Planning view layout, use any of the following methods:

- Select **Layout > I/O Planning**.
- From the Layout selector, select **I/O Planning**.
- Using the New Project wizard, create a new **I/O Planning Project**.

  Note: For more information on creating an I/O planning project, see the *Vivado Design Suite User Guide: System-Level Design Entry* (UG895). [Ref 1]

Figure 3-1 shows the I/O Planning view layout.

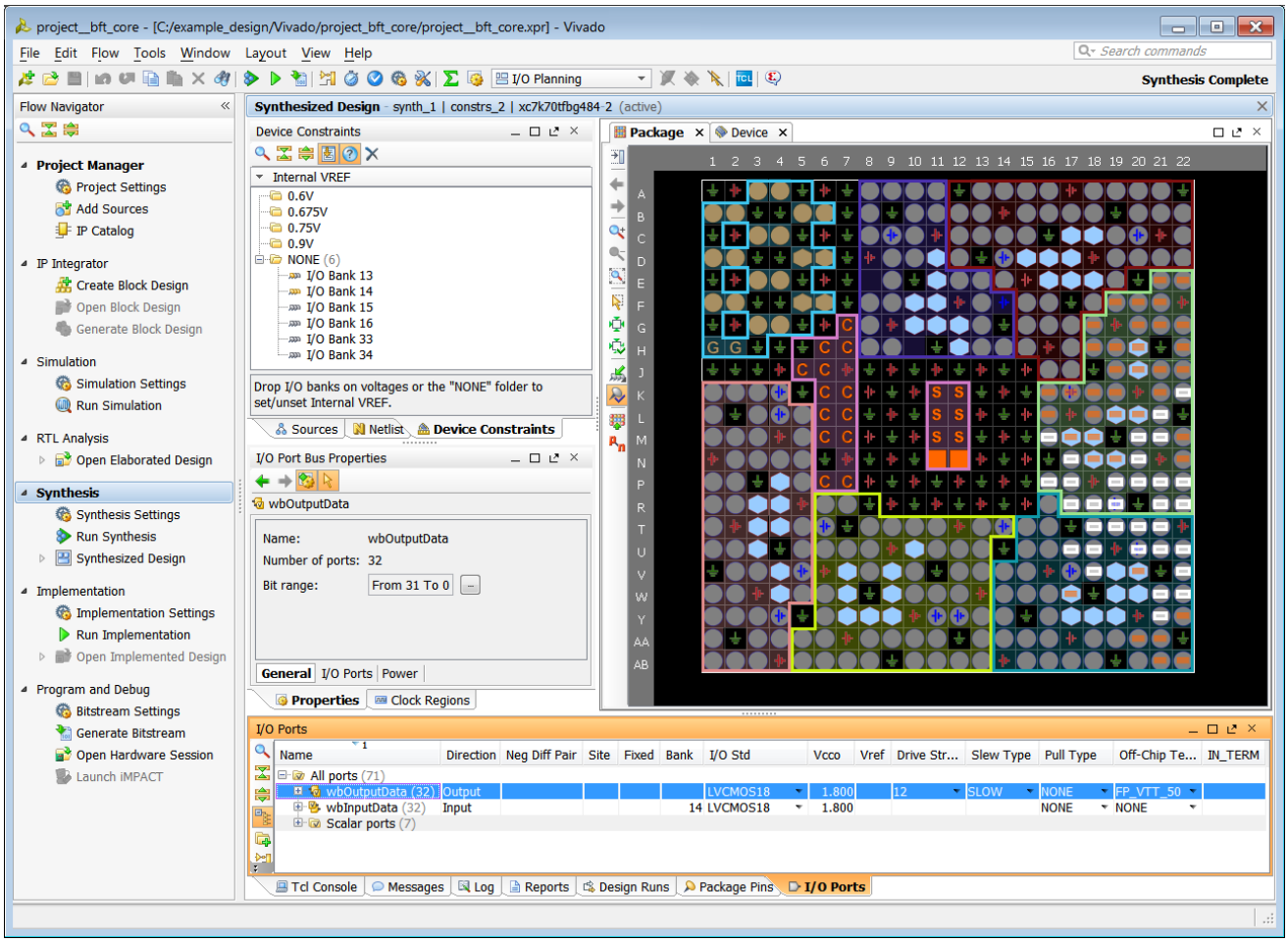

Figure 3-1: **I/O Planning View Layout**

Viewing Device Resources

The Device and Package windows show a graphical representation of the device and placed logic resources. When a logic object or device site is selected in these windows, information appears in the Properties window. The following sections describe these windows in detail.

TIP: *To search for specific objects or device sites, use the **Edit > Find** command. The Find dialog box includes many searchable object types and robust filtering capabilities to search the device or design for specific objects. You can select objects directly from the Find Results window.*

Properties

The Properties window shows properties for a selected object. The title bar changes to reflect the type of object selected. In most cases, the Properties window includes various views to show different information about the object. For example, in Figure 3-2, the Properties window shows the I/O Port Properties and includes the General, Properties, and Configure views. To show the Properties window, select **Window > Properties**.

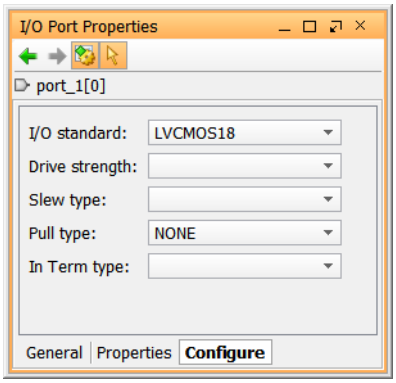

Figure 3-2: **I/O Port Properties**

Clock Region Resources and Statistics

The Clock Regions window allows easy selection of the clock regions. When you select a clock region in the Clock Regions window, the related I/O banks and regional clock resources are highlighted in the Package and Device windows, as shown in Figure 3-3.

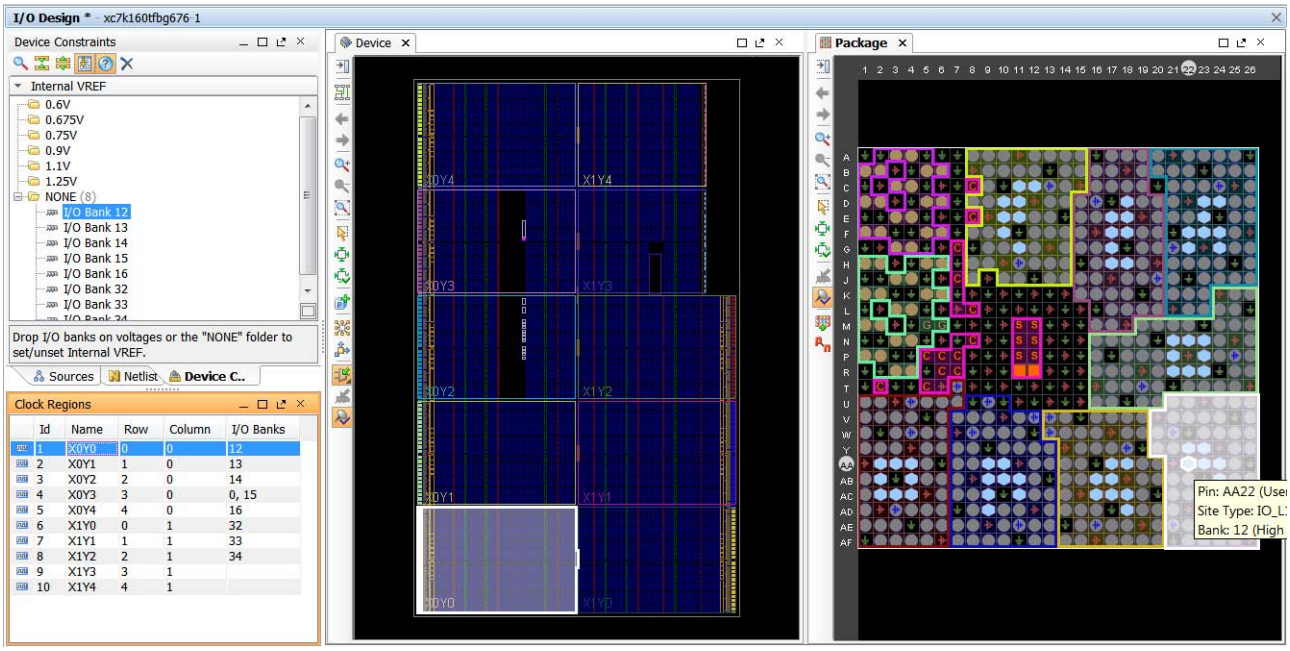

Figure 3-3: **Clock Regions Window**

When clock regions are highlighted, you can click the **Properties** tab to view the properties for the selected clock region. In the Clock Region Properties window, you can:

- Select the **Statistics** view to display the resource statistics available within the clock region as well as the logic content of the selected clock region.
- Select the **Resources** view to locate device clock resources for logic assignment, as shown in Figure 3-4.

Note: When you select an object in the Clock Regions Properties window, the object is cross-selected in another open window, such as the Device window.

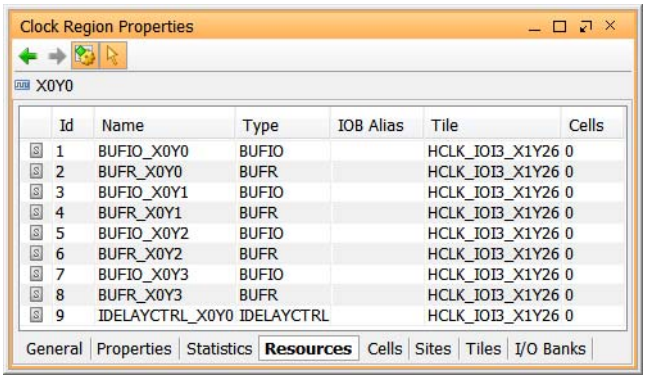

Figure 3-4: **Clock Region Properties—Resources View**

The Clock Resources window also provides a view of available clock resources to aid in planning and placing elements of global and regional clock trees. For information, see Using the Clock Resources Window.

I/O Bank Resources

You can select I/O resources in any of the I/O planning windows and their corresponding data is highlighted in all other windows as shown in Figure 3-3. This provides a visual indication of the relationship between the physical package and the internal die.

To access information about a specific I/O bank:

1. In the Package Pins window, select an I/O bank.
2. In the I/O Bank Properties window (Figure 3-5), click the views at the bottom of the window to see the different types of information available.

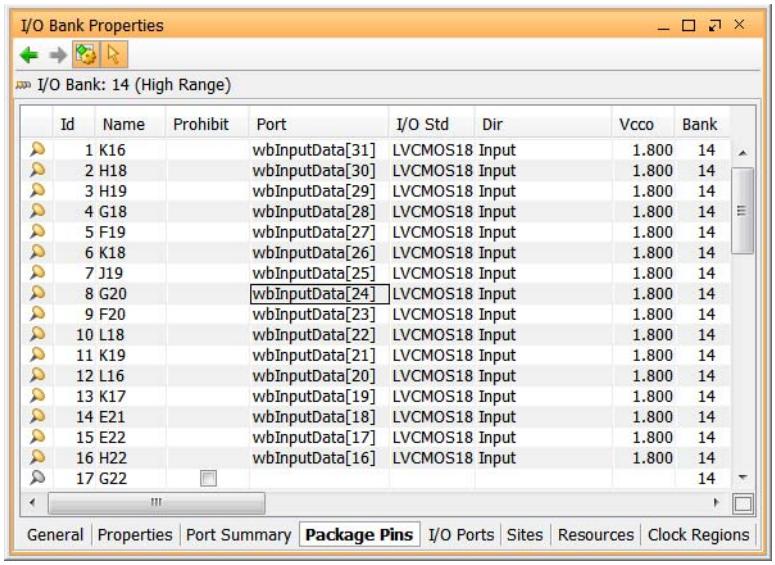

Figure 3-5: **I/O Bank Properties**

Multifunction Pins

The Package Pins window (Figure 3-6) contains several data types that display in columns similar to a spreadsheet. In this window, you can:

- Flatten, filter, and sort data.
- Move, hide, and configure columns to display and compare the various multifunction pins.
- Directly edit certain cells by entering text or selecting a value from drop-down menus.

The Package Pins window includes the following information:

- Type column that identifies multifunction pin types.
- Config column that shows the pin definition of the multifunction pin after you set the device configuration mode.

  Note: Many device configuration modes use multifunction pins. For more information, see Setting Device Configuration Modes.
- Other columns that contain information about logic or configuration modes involving multifunction type pins.
- Information that identifies conflicting multifunction pins for designs that contain GTs, memory controllers, or PCI™ logic.

In the Package window, the following symbols indicate the function of multifunction pins:

- Clock capable pins display as a hexagon icon
- V_{REF} pins display as a small power icon

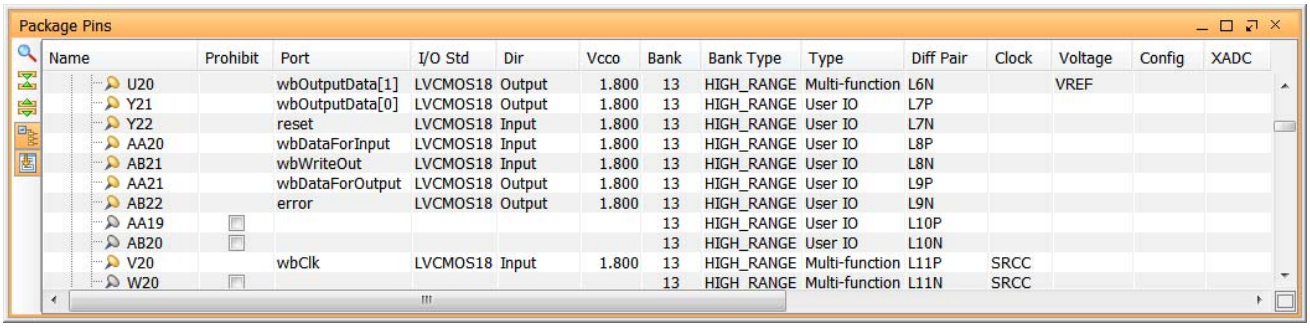

Figure 3-6: **Package Pins Window**

Defining Alternate Compatible Parts

You can select compatible devices for the design to allow you to retarget the design to alternate Xilinx® FPGAs if necessary. This feature checks that I/O pin assignments are defined to work across selected alternate devices. Compatible Xilinx devices are selected in the same package as the current target part to preserve as much of the I/O assignment as possible.

To define an alternate compatible part:

1. Select **Tools > I/O Planning > Set Part Compatibility**.
2. In the Set Part Compatibility dialog box (Figure 3-7), select the alternate parts, and click **OK**.

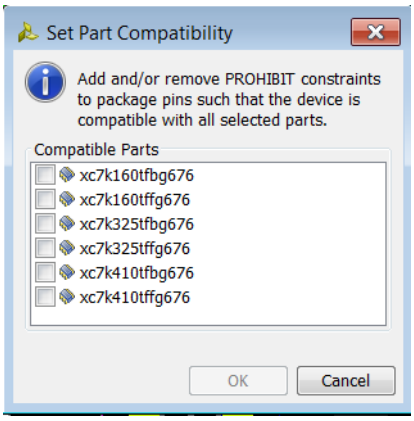

Figure 3-7: **Set Part Compatibility Dialog Box**

The Vivado IDE identifies the pins that are common to all selected alternate parts, and assigns PROHIBIT constraints to pins that are not common to all devices. The number of pins available for placement might be reduced when you select additional alternate parts.

In addition, the Vivado IDE automatically prohibits signals from being assigned to any unbonded pins in the selected alternate devices. A dialog box displays the number of prohibited package pins. You can view prohibits in the Package, Package Pins, and Device windows. Prohibited pins are indicated by a slashed circle icon .

Tcl Command for Defining Alternate Compatible Parts

Following is the associated Tcl command:

- **Tcl Command**: `set_property KEEP_COMPATIBLE`
- **Tcl Command Example**: `set_property KEEP_COMPATIBLE xc7k160tfbg676-1 [current_design]`

Note: For more information on Tcl commands, see the *Vivado Design Suite Tcl Command Reference Guide* (UG835) [Ref 12], or type `<command> -help`.

Setting Device Configuration Modes

To set device configuration modes as well as view and print information about the modes:

1. Select **Tools > I/O Planning > Set Configuration Modes**.
2. In the Set Configuration Modes dialog box (Figure 3-8), do the following, and click **OK**.
   - Select a configuration mode to view information, including a configuration diagram.
   - Click **Print** to print the configuration diagram.
   - Select the configuration modes.

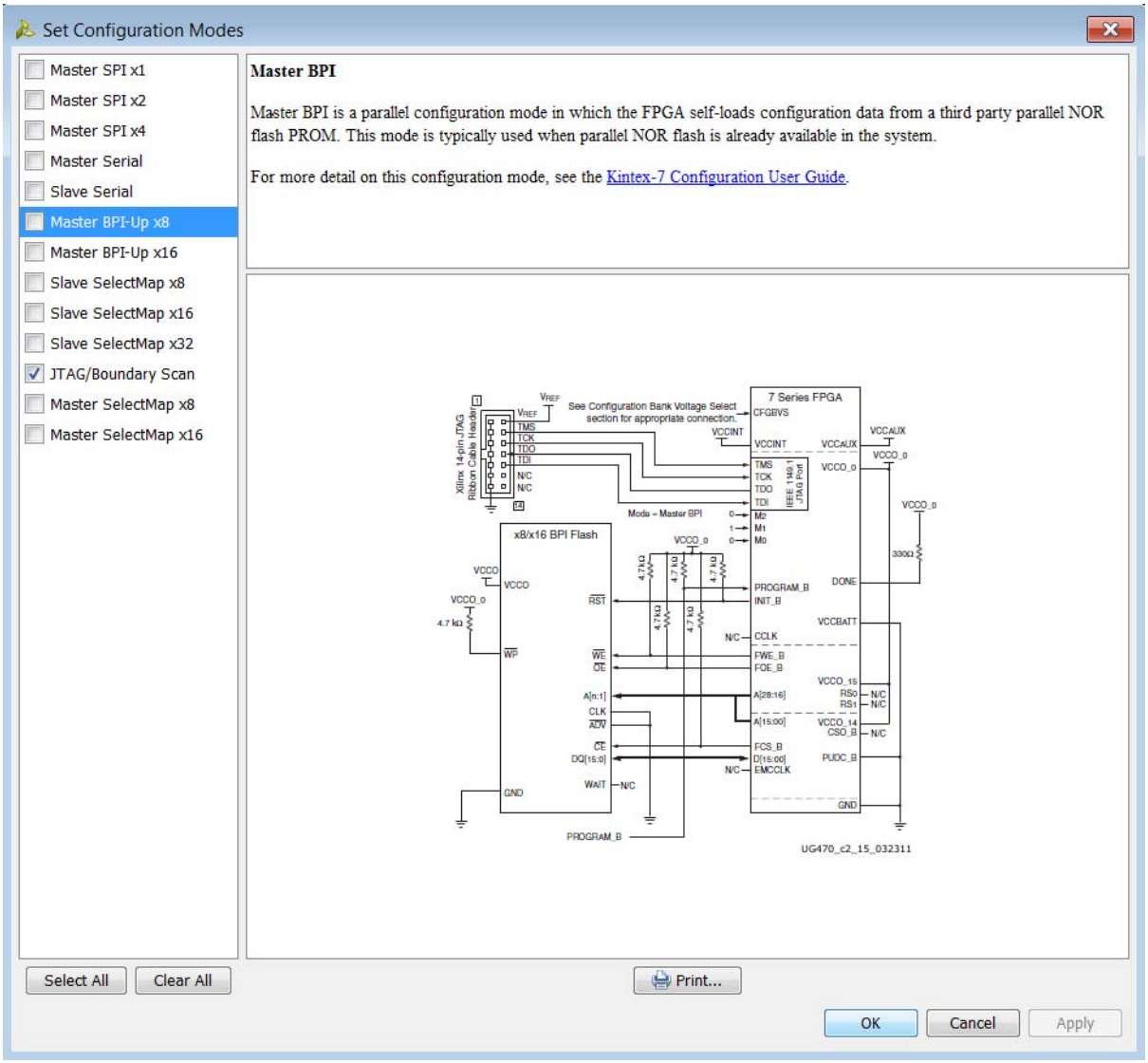

Figure 3-8: **Set Configuration Modes Dialog Box**

When you set the configuration modes, the associated I/O pins display in the Config column of the Package Pins window. The Vivado IDE prompts you to automatically sort the I/O pins using the Config column header.

Note: For more information on analyzing how the configuration modes might conflict with other multifunction pins, see Multifunction Pins.

CAUTION! *The tool does not differentiate between prohibits you set and configuration mode prohibits. Therefore, if you change or remove a configuration mode, the Vivado IDE does not remove prohibits.*

TIP: *When setting device configuration modes, you can undo your last action using **Edit > Undo**. Select **Undo** twice, first to clear the prohibit and again to clear the compatible configuration modes. Alternatively, you can enter* `undo` *twice in the Tcl Console.*

Tcl Commands for Setting Device Configuration Modes

Following is the associated Tcl command for setting the compatible configuration mode:

- **Tcl Command**: `set_property COMPATIBLE_CONFIG_MODES`
- **Tcl Command Example**: `set_property COMPATIBLE_CONFIG_MODES {{JTAG/Boundary Scan} {Master SelectMap x8}} [current_design]`

Following is the associated Tcl command for setting prohibits:

- **Tcl Command**: `set_property PROHIBIT`
- **Tcl Command Example**: `set_property PROHIBIT 1 [get_sites [list AW38 AU42 AJ28 AP37 AN33 AW42 AH28 AK37 AG29 AM33 BA42 BB38 AJ37 AJ30 AN34 AU41 AY38 AP36 AM32 AK30 AY37 BB39 AF30 AV41 AH30 AL36 AN36 BA39 AG31 BA41 AH29 AP35 AJ36 BA40 BB37 BB41 AL30 AK28 AN35 AT41 BA37 AW41 AL29 AK29 AM36 AL37]]`

Setting Device Constraints

In the Device Constraints window (Figure 3-9), you can set constraints, including DCI_ CASCADE and INTERNAL_VREF. FPGAs have configurable SelectIO™ interface drivers and receivers that support a variety of standard interfaces. This feature set includes programmable control of output strength and slew rate, on-chip termination using digitally-controlled impedance (DCI), and the ability to internally generate a reference voltage (INTERNAL_VREF). For more information, see the *7 Series FPGAs SelectIO Resources User Guide* (UG471) [Ref 5].

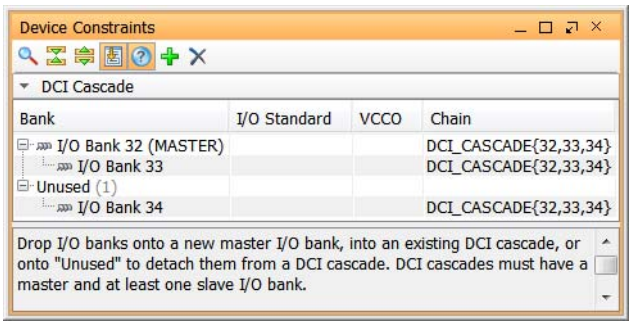

Figure 3-9: **Device Constraints Window**

Creating a DCI_CASCADE Constraint

The DCI_CASCADE constraint links two or more adjacent I/O banks together for DCI reference voltage purposes. The I/O bank with the DCI reference voltage is the *master*. All other I/O banks in the cascade are *slaves*. All banks in a cascade must be in the same I/O column of the device.

Note: You can configure the DCI_CASCADE constraint for Xilinx 7 series devices. For more information on this constraint, see the *Constraints Guide* (UG625) [Ref 13].

To create a DCI_CASCADE constraint:

1. In the Device Constraints window, select **DCI Cascade** from the drop-down list at the top of the window (Figure 3-10).

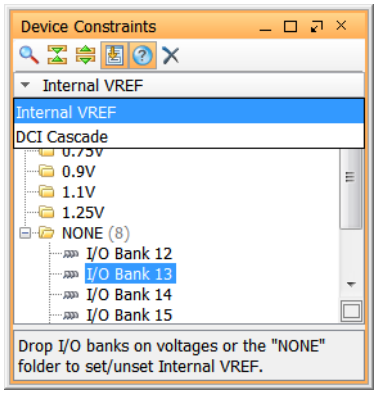

Figure 3-10: **DCI Cascade Drop-Down Menu**

2. Select the I/O banks you want to cascade, right-click, and select **Add DCI Cascade** from the popup menu.
3. In the Add DCI Cascade dialog box, select an I/O bank to be the master of the new DCI cascade, and click **OK**.

   The master bank appears in the Device Constraints window (Figure 3-11).

   Note: DCI cascades must have a master and at least one slave I/O bank.

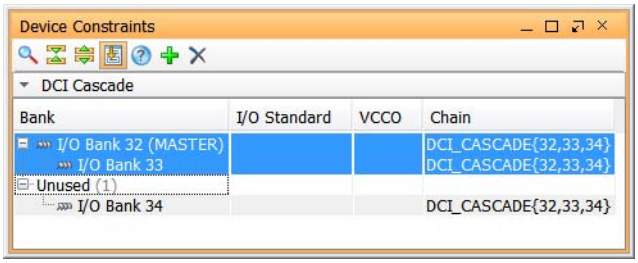

Figure 3-11: **DCI Cascade Master Bank**

TIP: *Alternatively, you can create a DCI_CASCADE constraint in the Package window or Package Pins window. Right-click the banks to cascade, and select* ***Create a DCI Cascade*** *from the popup menu.*

Tcl Command for Creating a DCI_CASCADE Constraint

Following is the associated Tcl command:

- **Tcl Command**: `set_property DCI_CASCADE`
- **Tcl Command Example**: `set_property DCI_CASCADE {31 32} [get_iobanks 36]`

Modifying or Removing DCI Cascade Constraints

To modify DCI cascades, do any of the following in the Device Constraints window:

- To change the master, right-click a DCI cascade, and select **Add DCI Cascade** from the popup menu. In the Add DCI Cascade dialog box, select a different bank to be the master.
- To add an I/O bank to a DCI cascade, drag and drop the I/O bank onto the DCI cascade.
- To remove an I/O bank from a DCI cascade, drag and drop the I/O bank onto the **Unused** folder.
- To remove an entire DCI cascade, right-click the DCI cascade, and select **Remove DCI Cascade Banks** from the popup menu (Figure 3-12).

  Note: The Tcl command equivalent for this action is: `set_property DCI_CASCADE {31 32} [get_iobanks 36]`

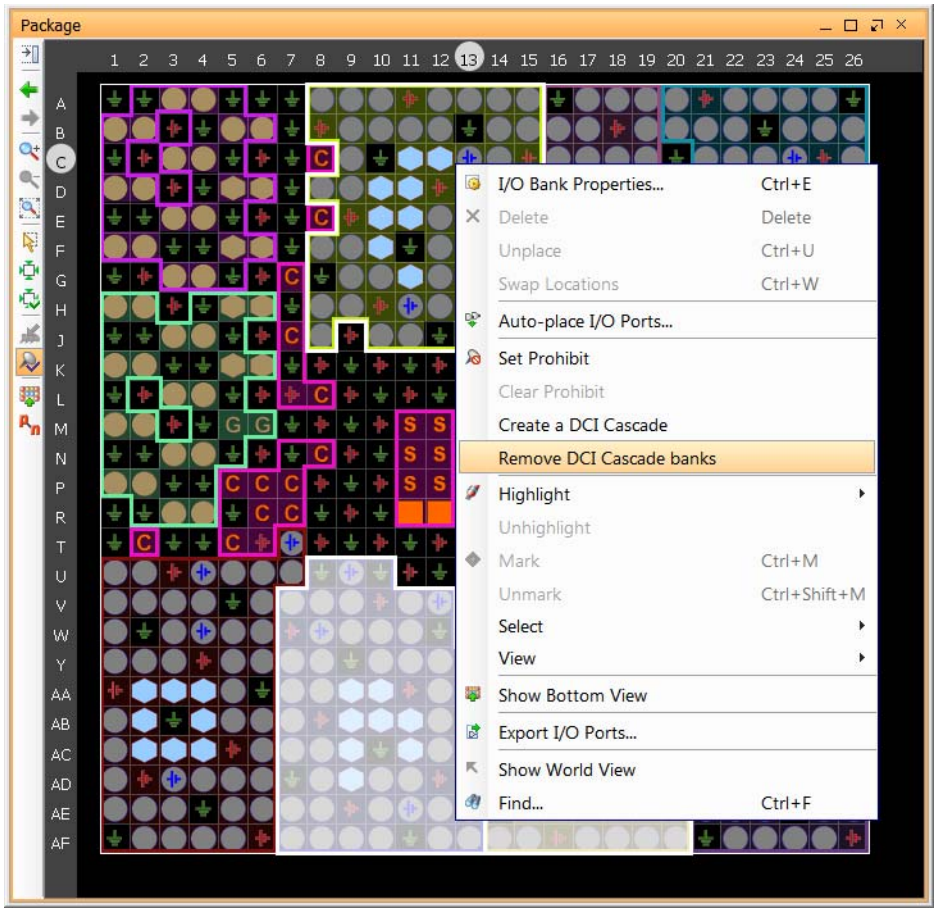

Figure 3-12: **Removing DCI Cascade Banks**

Creating an INTERNAL_VREF Constraint

FPGAs can optionally use an internally generated reference voltage by enabling the INTERNAL_VREF constraint. Internal generation removes the need to provide for a particular VREF supply rail on the PCB and frees the multi-purpose VREF pins in a given I/O bank to be used as normal I/O pins.

TIP: *All I/O banks that do not have an INTERNAL_VREF constraint are shown under the **NONE** folder in the Device Constraints window.*

To create an INTERNAL_VREF constraint, drag and drop the I/O bank onto the desired voltage folder (for example, **0.75V**) in the Device Constraints window (Figure 3-13).

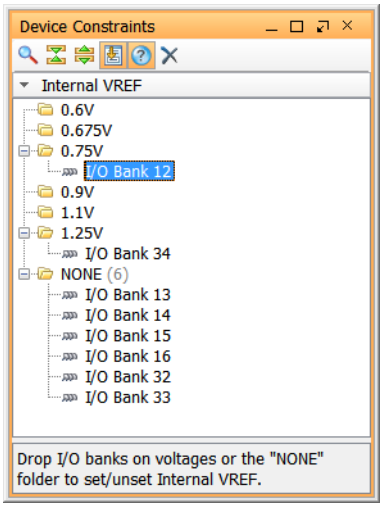

Figure 3-13: **Creating an Internal V_{REF} Constraint**

Tcl Command for Creating an INTERNAL_VREF Constraint

Following is the associated Tcl command:

- **Tcl Command**: `set_property INTERNAL_VREF`
- **Tcl Command Example**: `set_property INTERNAL_VREF 0.750 [get_iobanks 17]`

Setting the Configuration Bank Voltage Select (CFGBVS) Pin

The configuration bank voltage select (CFGBVS) logic input pin is referenced between V_{CCO_0} and GND. The CFGBVS pin must be set to High or Low to determine the I/O voltage support for the pins in bank 0. In the Vivado tools, you can use Tcl commands to set the CFGBVS tie off information to either V_{CCO} or GND. You can set the configuration voltage, or V_{CCO_0} voltage, to 3.3, 2.5, or 1.8. Based on these settings, DRC checks are run on Bank 0, 14, and 15. These value are also used when exporting IBIS models.

Following is an example:

```
set_property CFGBVS VCCO [current_design]
set_property CONFIG_VOLTAGE 3.3 [current_design]
```

By default, the CFGBVS property is empty. The Vivado tools check whether the CFGBVS property is set to V_{CCO} or GND. If the CFGBVS property has a value, the Vivado tools check for a COMPATIBLE_CONFIG_MODE property. DRCs are issued based on IOSTANDARD and CONFIG_VOLTAGE settings for the bank.

When you export to a CSV file, the Vivado tools provide V_{CCO} tie off information for Bank 0, 14, and 15 based on the setting for the COMPATIBLE_CONFIG_MODE property. For example, if you use JTAG/Boundary Scan, CFGBVS is GND, and CONFIG_VOLTAGE is 3.3, the tools issue the critical warning: DRC CFGBVS-4. This indicates that the CONFIG_VOLTAGE is set to 3.3 and must instead be set to V_{CCO}, which has a value of 1.8.

Note: For more information on the CFGBVS pin, see the *7 Series FPGAs Configuration User Guide* (UG470) [Ref 14].

Defining and Configuring I/O Ports

You can use the Vivado IDE to import, create, and configure I/O ports as described in the following sections.

Importing I/O Ports

Depending on the project type, you can use the following methods to import I/O ports:

- **I/O Planning Project**: You can import XDC and CSV files into an empty I/O planning project when you create the project or later using the file import capability. For details, see Importing a CSV File and Importing an XDC File.
- **RTL Project**: Use RTL files or headers to create an RTL project for I/O pin planning, then add more complete RTL source files to the project later as the design progresses. When

you create an RTL-based or synthesized netlist-based project, the I/O Ports window automatically populates with the I/O ports defined in the design.

- **Migrate from I/O Planning Project to RTL Project**: You can convert an I/O planning project to an RTL project, turning the I/O ports into a top-level Verilog or VHDL module definition for the design. For more information, see Migrating to an RTL Design.

Importing a CSV File

You can import a CSV file to populate the I/O Ports window within the I/O Planning layout view. You can then assign these I/O ports to physical package pins to define the device pin configuration.

To import an I/O ports list from a CSV file:

1. Select **File > Import > Import I/O Ports**.
2. In the Import I/O Ports dialog box (Figure 3-14), select **CSV File**, and browse to select the file to import.

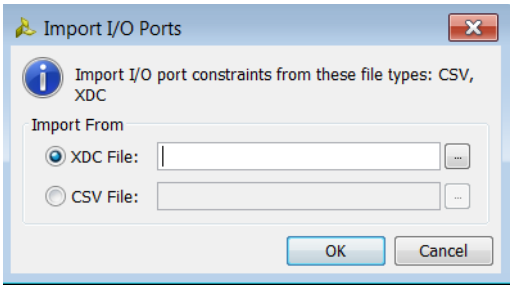

Figure 3-14: **Import I/O Ports Dialog Box**

Figure 3-15 shows the CSV file format. CSV is a standard file format used by FPGA and board designers to exchange information about device pins and pinout. The Vivado IDE requires a specific CSV file format for importing I/O pin-related data, as described in Appendix A, I/O Port Lists in CSV File Format.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
#Top: ios Floorplan: io_xc7k160tffg676-1 Part: xc7k160tffg676-1															
#Build: Vivado v2012.2															
IO Bank	Pin Numbe	IOB Alias	Site Type	Min Trace	Max Trace	Trace Ler	Prohibit	Interface	Signal Nam	Direction	DiffPair Ty	DiffPair Sig	IO Standard	Drive (mA)	Slew Rate
14	A25	IOB_X0Y147	IO_L1N_T0_D01_DIN_14	191.467	193.391				port_1[0]	OUT			LVCMOS18	12	SLOW
14	B24	IOB_X0Y148	IO_L1P_T0_D00_MOSI_1	180.881	182.699				port_1[1]	OUT			LVCMOS18	12	SLOW
14	K21	IOB_X0Y149	IO_0_14	73.73	74.471				port_1[2]	OUT			LVCMOS18	12	SLOW
13	U16	IOB_X0Y50	IO_25_13	73.041	73.775				port_1[3]	OUT			LVCMOS18	12	SLOW
13	P18	IOB_X0Y51	IO_L24N_T3_13	73.003	73.737				port_1[4]	OUT			LVCMOS18	12	SLOW
13	R18	IOB_X0Y52	IO_L24P_T3_13	63.796	64.437				port_1[5]	OUT			LVCMOS18	12	SLOW
13	T17	IOB_X0Y53	IO_L23N_T3_13	73.336	74.073				port_1[6]	OUT			LVCMOS18	12	SLOW
13	U17	IOB_X0Y54	IO_L23P_T3_13	78.405	79.193				port_1[7]	OUT			LVCMOS18	12	SLOW
13	M19	IOB_X0Y55	IO_L22N_T3_13	100.104	101.11				port_1[8]	OUT			LVCMOS18	12	SLOW
13	N18	IOB_X0Y56	IO_L22P_T3_13	86.656	87.527				port_1[9]	OUT			LVCMOS18	12	SLOW
13	R17	IOB_X0Y57	IO_L21N_T3_DQS_13	66.334	67.001				port_1[10]	OUT			LVCMOS18	12	SLOW
13	R16	IOB_X0Y58	IO_L21P_T3_DQS_13	66.536	67.205				port_1[11]	OUT			LVCMOS18	12	SLOW
13	N17	IOB_X0Y59	IO_L20N_T3_13	65.725	66.385				port_1[12]	OUT			LVCMOS18	12	SLOW
13	P16	IOB_X0Y60	IO_L20P_T3_13	53.671	54.21				port_1[13]	OUT			LVCMOS18	12	SLOW
13	T19	IOB_X0Y61	IO_L19N_T3_VREF_13	53.96	54.503				port_1[14]	OUT			LVCMOS18	12	SLOW

Figure 3-15: **I/O Port List in CSV File Format**

You can define differential pairs in the CSV file in several ways. For example, the Vivado IDE recognizes differential pairs directly defined with DiffPair Signal and DiffPair Type properties. In addition, the Vivado IDE can infer diff pairs when only one port of the pair is defined in the CSV file or two named nets imply a differential pair. For more information, see Differential Pairs in the CSV File in Appendix A.

When inferring differential pairs, the Vivado IDE displays a prompt to confirm the assignment of the pairs, as shown in Figure 3-16.

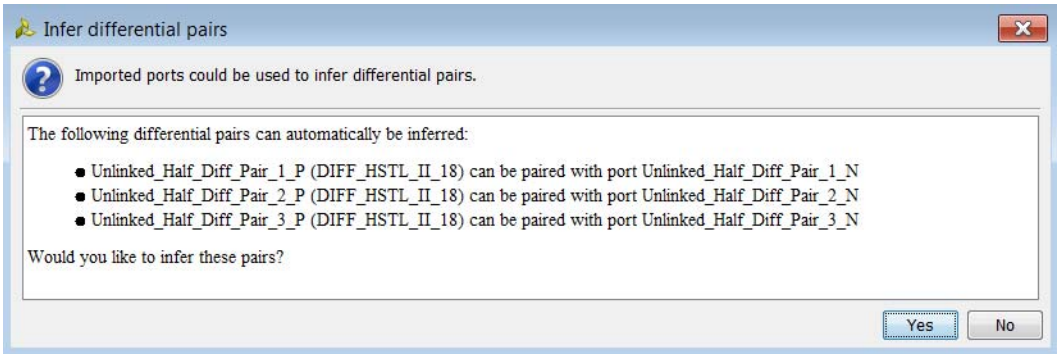

Figure 3-16: **Infer Differential Pairs Dialog Box**

CSV files can also contain additional information not recognized by the Vivado IDE. If unrecognized information is found in the imported CSV file, the information is displayed in user columns of the Package Pins window for your review and use. To modify or define values in the user CSV fields, right-click in the Package Pins window, and select the **Set User Column Values** from the popup menu.

Note: For information on exporting a CSV file, see Exporting I/O Pin and Package Data.

Importing an XDC File

To import I/O port definitions from an XDC file:

1. Select **File > Import > Import I/O Ports**.
2. In the Import I/O Ports dialog box (Figure 3-14), select **XDC File**, and browse to select the file to import.

Because the XDC format does not define port direction, the direction is undefined. To define the I/O port direction, select **Set Direction** from the popup menu in the I/O Ports window. You can also directly modify the direction of a specific I/O port in the I/O Ports window. For more information, see Setting I/O Port Direction.

Creating Single-Ended or Differential I/O Ports

You can manually define new ports in an I/O planning project. Refer to Xilinx device documentation for information regarding voltage capabilities of the device.

To create I/O ports:

1. In the I/O Ports window, select **Create I/O Ports** from the popup menu.
2. In the Create I/O Ports dialog box (Figure 3-17), edit the following options, and click **OK**:
   - **Name**: Enter the port or bus name to create.
   - **Direction**: Select the port direction.
   - **Diff Pair**: Define differential pair signals or buses.

     Note: To create a differential I/O port, enable this option. This creates two ports and adds `_P` and `_N` to the specified name.
   - **Create Bus**: Enter a bus range for bus creation.
   - **I/O Standard**: Select the I/O standard constraint.
   - **Drive Strength**: Select the drive strength value.
   - **Slew Type**: Select the slew type value.
   - **Pull Type**: Select the pull type value.
   - **In Term Type**: Define the parallel termination properties of the input signal.

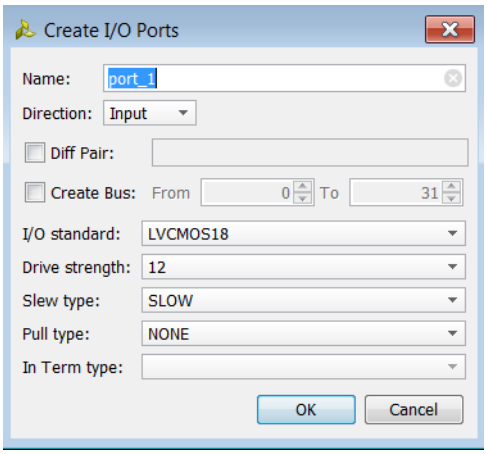

Figure 3-17: **Create I/O Ports Dialog Box**

Tcl Command for Creating Single-Ended or Differential I/O Ports

Following is the associated Tcl command:

- **Tcl Command**: `create_port`
- **Tcl Command Example (Single-Ended I/O Port)**: `create_port port_1 -direction in`
- **Tcl Command Example (Differential I/O Port)**: `create_port port_2 -direction in -diff_pair`

Making and Splitting Differential Pairs

To define a differential pin pair in an I/O planning project:

1. In the I/O Ports window, select any two I/O ports, and select **Make Diff Pair** in the popup menu.

IMPORTANT: *The Make Diff Pair option is not available in RTL projects. In RTL projects, differential ports must be defined in the source code using appropriate I/O buffer instantiations.*

In the Make I/O Diff Pair dialog box (Figure 3-18), the two I/O Ports display with Positive End and Negative End assignments made by the tool.

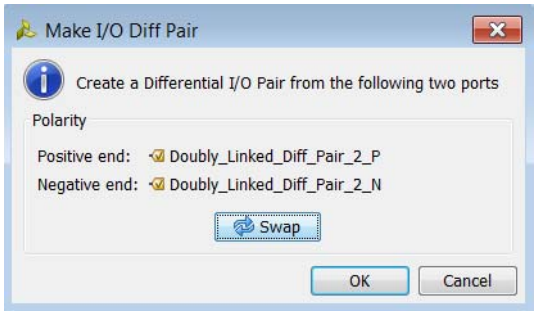

Figure 3-18: **Make I/O Diff Pair Dialog Box**

2. To reverse the Positive End and Negative End signals, click **Swap**, and click **OK**.

TIP: *Use the **Split Diff Pair** command from the popup menu to separate a diff pair into two ports.*

Tcl Command for Splitting Differential Pairs

Following is the associated Tcl command:

- **Tcl Command**: `make_diff_pair_ports`
- **Tcl Command Example**: `make_diff_pair_ports port_6 port_5`

Configuring I/O Ports

You can configure one or more I/O ports to define I/O standard, drive strength, slew type, pull type, and in term. This is useful to configure ports that were imported from CSV or XDC files without the appropriate characteristics. Refer to Xilinx device documentation for information regarding voltage capabilities of the device.

To configure a port or a group of ports:

1. In the I/O Ports window, select the ports.
2. Select **Configure I/O Ports** from the popup menu.
3. In the Configure Ports dialog box (Figure 3-19), edit the following options, and click **OK**:
   - **I/O Standard**: Select the I/O standard constraint. The tool does not check the I/O standard when it is assigned. You can assign any I/O standard to any port, but this might result in errors when running DRCs.
   - **Drive Strength**: Select the drive strength value.
   - **Slew Type**: Select the slew type value.

- **Pull Type**: Select the pull type value.
  - **PULLUP**: Applies a weak logic High level on a tristateable output or bidirectional port to prevent it from floating when not being driven.
  - **PULLDOWN**: Applies a weak logic Low level on a tristateable output or bidirectional port to prevent it from floating when not being driven.
  - **KEEPER**: Applies a weak driver on an tristateable output or bidirectional port to preserve its value when not being driven.
  - **NONE**: Does not apply a driver.

  Note: Alternatively, you can set the pull type constraint by clicking in the Pull Type column of the I/O Ports window.
- **In Term Type**: Define the parallel termination properties of the input signal.

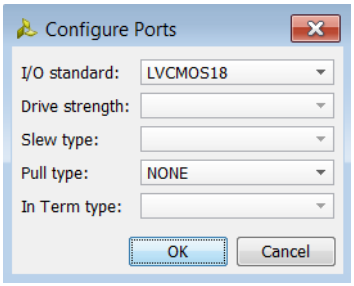

Figure 3-19: **Configure Ports Dialog Box**

IMPORTANT: *For 7 series devices, all I/O ports must have explicit values for the PACKAGE_PIN and IOSTANDARD constraints to generate a bitstream file. In the I/O Ports window, the word* `default` *is displayed in red to indicate that these values must be applied manually. This is because 7 series devices have low and high voltage I/O banks, and you must apply extra care when assigning I/O standards.*

Setting I/O Port Direction

To set I/O port direction, use any of the following methods:

- For I/O planning projects only, in the Direction (Dir) column of the I/O Ports window, click a port and change the direction using the drop-down menu.
- For I/O planning projects only, click a port in the I/O Ports window, and change the direction in the I/O Port Properties window.
- For I/O planning projects only, select the I/O ports, buses, or interfaces to be configured, and select **Set Direction** from the popup menu in the I/O Ports window.
- For RTL projects only, define the port direction in the RTL source.

Creating I/O Port Interfaces

To group multiple ports or buses together, you can create an interface. This aids in pin assignment by treating all of the interface ports as one group. Assigning all of the pins simultaneously helps condense and isolate the interface for clock region or PCB routing. This also makes it easier to visualize and manage the signals associated with a particular logic interface.

To create an interface:

1. In the I/O Ports window, select the signals to group together.
2. Select **Create I/O Port Interface** from the popup menu.
3. In the Create I/O Port Interface dialog box (Figure 3-20), enter a name for the interface, adjust assignment selection, and click **OK**.

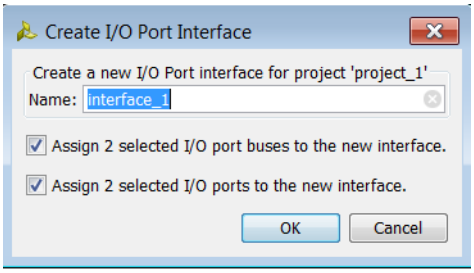

Figure 3-20: **Create I/O Port Interface Dialog Box**

The interfaces appear as expandable folders in the I/O Ports window (Figure 3-21).

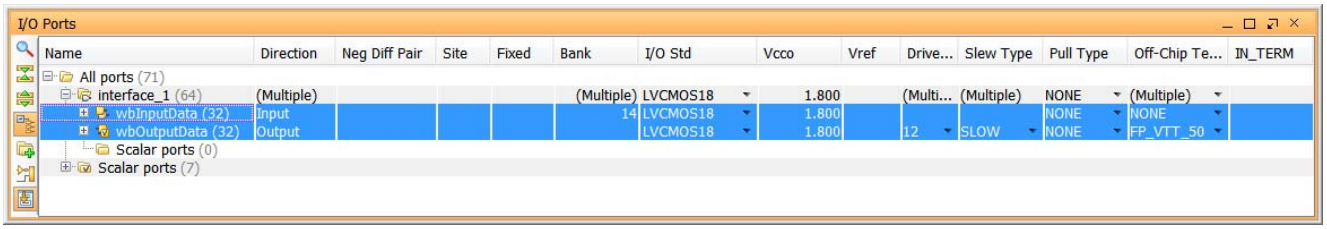

Figure 3-21: **I/O Port Interfaces in I/O Ports Window**

TIP: *To delete interfaces, select an interface, and select **Delete** from the popup menu, or press the **Delete** key.*

Adding I/O Ports to an Interface

To add I/O ports to an interface, do either of the following in the I/O Ports window:

- Select the I/O ports, and drag them into the interface folder.
- Right-click a port or bus, and select **Assign to Interface**. In the Select I/O Port Interface dialog box, select the target interface.

Removing I/O Ports from an Interface

To remove I/O ports, do the following in the I/O Ports window:

1. Right-click a port.
2. From the popup menu, click **Unassign from Interface**.

Tcl Commands for Working with I/O Port Interfaces

Following is the associated Tcl command for creating I/O port interfaces:

- **Tcl Command**: `create_interface`
- **Tcl Command Example**:

```
create_interface interface_1

set_property interface interface_1 [get_ports [list  {test_1[3]} {test_1[2]} {test_1[1]} {test_1[0]} {test_1_n[3]} {test_1_n[2]} {test_1_n[1]} {test_1_n[0]}]]

set_property interface interface_1 [get_ports [list  port_2 port_2_N port_1 port_4]]

endgroup
```

Following is the associated Tcl command for removing I/O ports from an interface:

- **Tcl Command**: `set_property INTERFACE`
- **Tcl Command Example**: `set_property INTERFACE "" [get_ports [list port_2 port_2_N]]`

Prohibiting I/O Pins and I/O Banks

The I/O Planning view layout provides an interface to selectively prohibit port placement onto individual I/O pins, groups of I/O pins, or I/O banks. You can select and prohibit pins in the Device, Package, and Package Pins windows.

To prohibit I/O pins or I/O banks:

1. Select the I/O pins or I/O banks in the Device, Package, or Package Pins window.
2. Select **Set Prohibit** from the popup menu.

Prohibited pins are indicated by a:

- Slashed circle in the Device window and Package window (Figure 3-22)
- Check mark in the Prohibit column of the Package Pins window

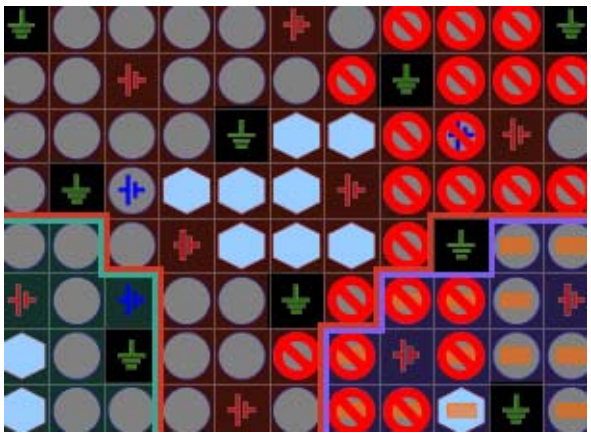

Figure 3-22: **Prohibits on Package Pins**

TIP: *You can clear prohibits from the Prohibit column of the Package Pins view. Select an individual prohibit or use **Ctrl+A** to select all pins, right-click, and select **Clear Prohibit** from the popup menu. Alternatively, you can use a Tcl command to clear the prohibit, for example:* `set_property prohibit 0 [get_sites U17]`

Clock Planning

When working with a synthesized or implemented design, you can manually place global and regional clock-related logic, such as BUFGCTRLs, MMCMs, BUFRs, and IDELAYCTRLs, using the Clock Resources window as described in Using the Clock Resources Window. You can also manually place clock logic in the Device window. Appropriate logic sites are displayed in the Device window for all device-specific resources.

Note: For more information on clock planning, see the *7 Series FPGAs Clocking Resources User Guide* (UG472) [Ref 15].

RECOMMENDED: *It is recommended that you select clocking resources prior to pinout selection. This is because clocking selections can dictate a particular pinout and can also direct logic placement for that logic. Proper clocking selections can yield superior results.*

Locating Logic Cells

To locate logic cells for placing onto the device:

1. Select **Edit > Find**.
2. In the Find dialog box, specify **Cells** in the Find field, and define the criteria to locate the specific logic cell or cells.
3. From the Find Results window, drag logic cells onto the Clock Resources window or the Device window to assign to the appropriate device resource.

Note: For more information, see the *Vivado Design Suite User Guide: Using the Vivado IDE* (UG893) [Ref 11].

TIP: *You can also locate physical resources on the device, such as global clock buffers, for placing logic cells. Specify **Sites** in the Find field, and define the criteria as needed. Select a result in the Find Results window to highlight the physical device resource in the Clock Resources window or Device window.*

Using the Clock Resources Window

The Clock Resources window shows the relationship and interactions between regional and global clocking resources: BUFRs, BUFIOs, BUFGs, MMCMs and GTs. The Clock Resources window shows a simplified view of the device resources but maintains proper relative positioning between these resources.

RECOMMENDED: *It is recommended that you use the Clocking Wizard in the Vivado IP catalog to generate MMCM or PLL modules to define clock connections. For information, see the LogiCORE IP Clocking Wizard Product Guide (PG065) [Ref 8].*

Note: Most of the details of the FPGA device shown in the Device window are not shown in the Clock Resources window.

Figure 3-23 shows the Clock Resources window for a Kintex™-7 K70T device, which indicates:

- The device has eight clock regions arranged in a 4x2 matrix, numbered from X0Y0 in the lower-left of the device to X1Y3 in the upper-right.
- Each of these clock regions also has I/O banks containing clock-capable pins (CCIOs), BUFIOs, and BUFRs that display in the Clock Resources window for each clock region.
- The device itself is divided into a top "half" containing four clock regions, and a bottom "half" containing four clock regions.
- The BUFGs for managing global clocks on the device are in the center column of the device.

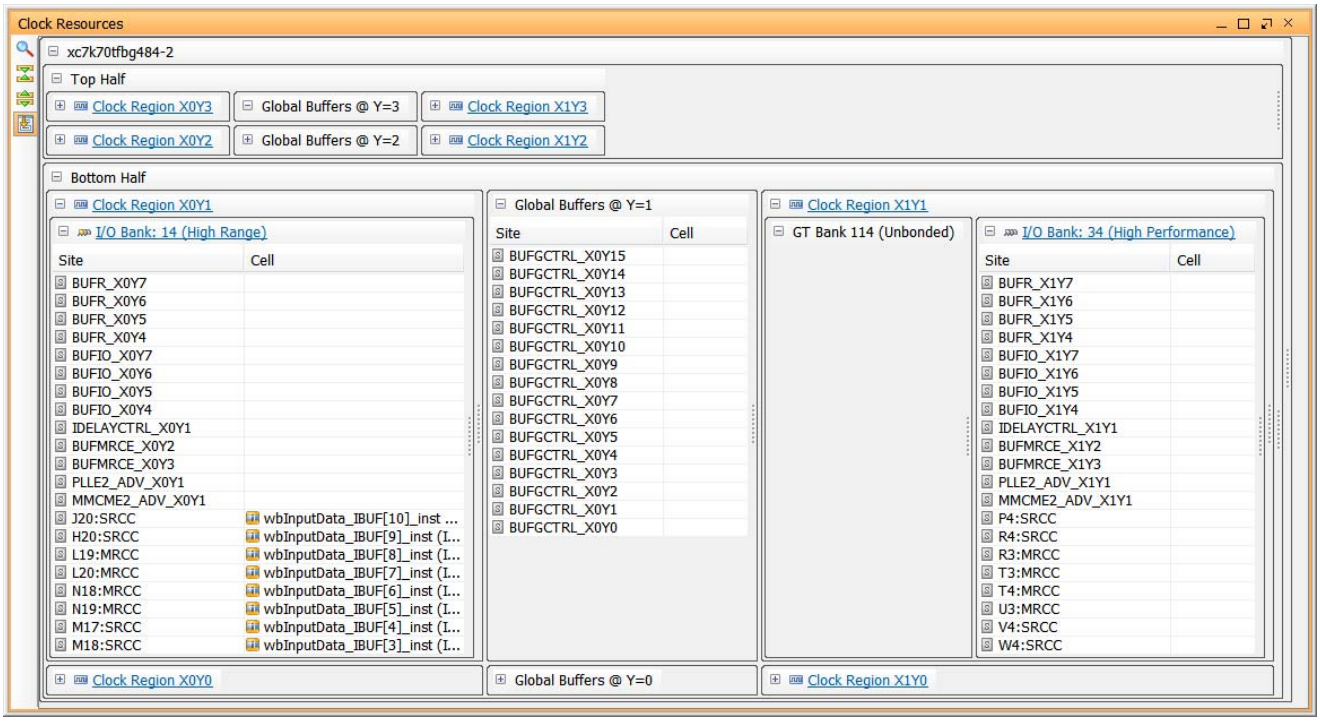

Figure 3-23: **Clock Resources Window**

Collapsing and Expanding Levels

To expand or collapse levels to display only the information of interest:

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

Cross-Selecting Objects

To cross-select objects between windows, do either of the following in the Clock Resources window:

- Click the name of a specific clock region or I/O bank.

  Use this method to locate a specific object on the device, on the package, or in the netlist.

- Click the **Automatically Scroll to Selected Object** toolbar button to scroll to an object that is selected in another window.

  Use this method to locate a specific resource on the device in the Clock Resources window.

TIP: *You can toggle off automatic scrolling to prevent the Vivado IDE from changing the displayed resources every time an object is selected in a different window.*

Placing Design Cells

The Clock Resources window displays two columns: Site and Cell. These columns report both the device resource and the design cell to which it is assigned.

To place design cells:

1. In the Find Results, Schematic, Netlist, or I/O Ports windows, select logic cells from the design to place into the device resources.
2. Drag the cells onto the Cell column of the appropriate device resource in the Clock Resources window.

   As you drag the cell in the Clock Resources window, tooltips indicate sites where the cell cannot be placed with a slashed circle symbol and sites where the cell can be placed with a rectangle.

IMPORTANT: *When you place cells from the design, the Vivado IDE enforces specific rules and limitations regarding global and regional clock tree structures. For specific information on these rules and limitations, see the 7 Series FPGAs Clocking Resources User Guide (UG472) [Ref 15].*

Placing Clock Logic in the Device Window

To place clock logic manually:

1. In the Device window, zoom to locate the appropriate device site to place the logic.
2. Select the **Cell Drag & Drop Modes** toolbar button, and select **Create Site Constraint Mode** .
3. Select the logic cell to place from the Find Results, Schematic, Netlist, or I/O Ports window, and drag it onto the appropriate device resource in the Device window.

Placing I/O Ports

The I/O Planning view layout provides several ways to assign I/O ports to package pins. You can select individual I/O ports, groups of I/O ports, or interfaces in the I/O Ports window, and assign them to package pins in the Package window or to I/O pads in the Device window.

In the Package window, you can:

- Drag and drop ports to package pins.
- View port placement and constraints.
- Move the cursor over the pins to show the I/O pin coordinates on the top and left sides of the window.
- Hold the cursor over a pin to show a tooltip that displays the pin information.
- Display differential pairs using the **Show Differential Pairs** toolbar button .

Note: Additional I/O pin and bank information displays in the status bar located at the bottom of the Vivado IDE.

Placing I/O Ports Sequentially

To place I/O ports sequentially:

1. In the I/O Ports window, select an individual I/O port, a group of I/O ports, or interfaces.
2. Use one of the following commands:
   - In the I/O Ports window, select **Place I/O Ports Sequentially** from the popup menu.
   - In either the Package window or the Device window, click the **Place Ports** toolbar button, and select **Place I/O Ports Sequentially** .

   The first I/O port in the group is attached to the cursor when you move it over a package pin or I/O pad. A tooltip displays the I/O port and package pin names.
3. To assign an I/O port, click a pin or a pad.

   If you select more I/O ports, the command is continued. The cursor drags the next I/O ports and so on until all of the I/O ports are placed, or you press **Esc**.

TIP: *The Vivado IDE assigns ports in the order that they appear in the I/O Ports window. You can adjust the assignment order by applying sorting techniques in the I/O Ports window prior to assignment.*

Figure 3-24 shows I/O ports placed sequentially.

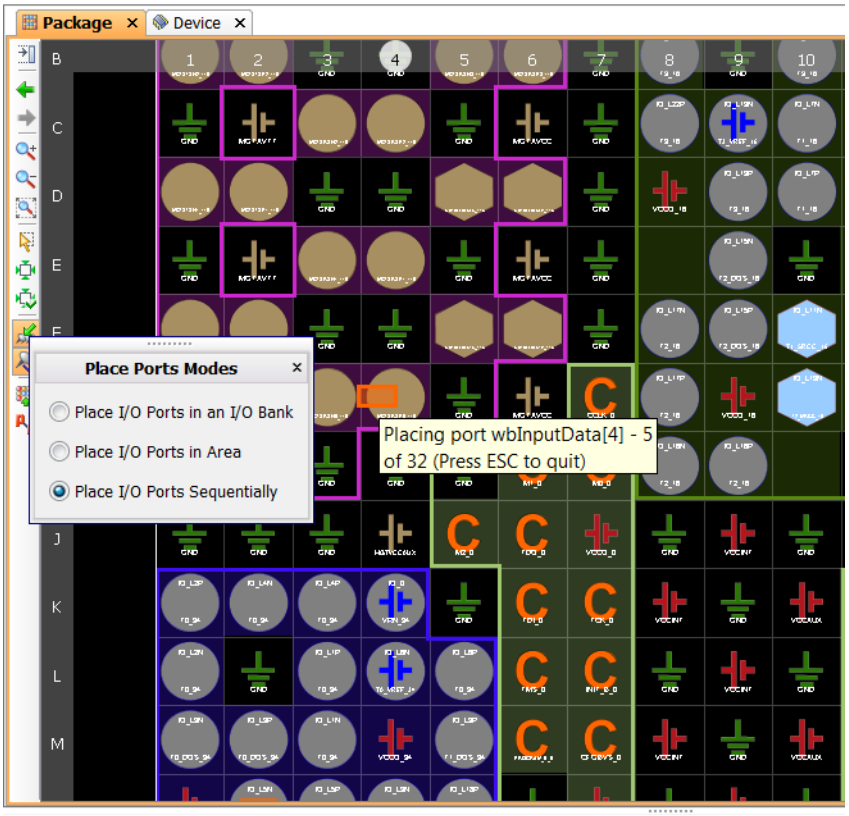

Figure 3-24: **I/O Ports Placed Sequentially**

Placing I/O Ports into I/O Banks

To place I/O ports into I/O banks:

1. In the I/O Ports window, select an individual I/O port, a group of I/O ports, or interfaces.
2. Use one of the following commands:
   - In the I/O Ports window, select **Place I/O Ports in an I/O Bank** from the popup menu.
   - In either the Package window or the Device window, click the **Place Ports** toolbar button, and select **Place I/O Ports in an I/O Bank** .

   The group of I/O ports is attached to the cursor when it is dragged over a package pin or I/O pad. A tooltip shows the number of pins that can be placed in the selected I/O bank.

3. Click a pin or pad to assign the selected I/O ports.

   If more I/O ports are selected than fit in the I/O bank, the Vivado IDE places as many as possible in the selected I/O bank, then lets you select another I/O bank into which to place the remaining ports. The cursor drags the remaining I/O ports to the next selected I/O bank and so on until all of the I/O ports are placed, or you press **Esc**.

TIP: *The Vivado IDE assigns ports in the order that they appear in the I/O Ports window. You can adjust the assignment order by applying sorting techniques in the I/O Ports window prior to assignment.*

Port assignment to device resources is also driven from the initial selection from the I/O bank. Selecting a pin at one end of an I/O bank results in a continuous bus assignment across the I/O bank.

The Vivado IDE also keeps track of PCB routing concerns for buses. Pin ordering during assignment attempts to keep the bus bits vectored within the assignment area. You can customize assignment patterns to address other bus routing concerns.

Figure 3-25 shows I/O ports placed in an I/O bank.

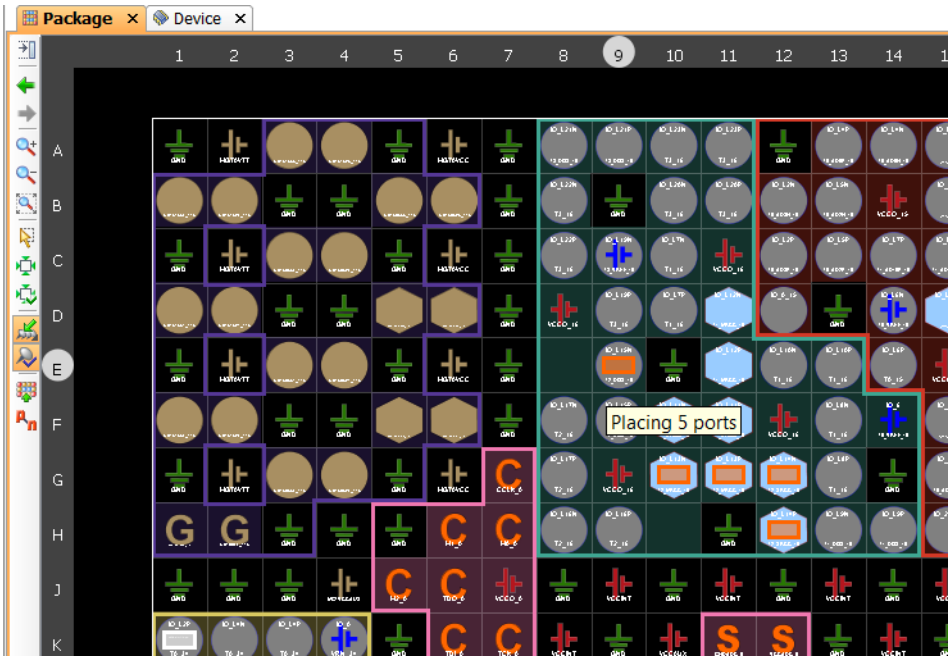

Figure 3-25: **I/O Ports Placed in I/O Banks**

Tcl Command for Placing Ports into I/O Banks

Following is the associated Tcl command:

- **Tcl Command**: `place_ports -iobank`
- **Tcl Command Example**: `place_ports -iobank [get_iobanks {12 13 14 15}] [all_inputs]`

TIP: *To place ports on all banks, use the following Tcl command: `place_ports -iobanks [lrange [get_iobanks] 1 end] <port list>`. The `place_ports` command is not supported for bank 0.*

Placing I/O Ports in a Defined Area

To place I/O Ports into a defined area:

1. In the I/O Ports window, select individual I/O ports, groups of I/O ports, or interfaces.
2. Use one of the following commands:
   - In the I/O Ports window, select **Place I/O Ports in Area** from the popup menu.
   - In either the Package window or the Device window, click the **Place Ports** toolbar button, and select **Place I/O Ports in Area** .

   The cursor turns into a cross symbol, which indicates that you can define a rectangle for port placement.
3. In either the Package window or the Device window, draw a rectangle to define the assignment area.

   If you select more I/O Ports than fit in the defined area, the command is continued. The cursor continues to display as a cross to draw another area to place the remaining I/O ports until all of the I/O ports are placed, or you press **Esc**.

TIP: *The Vivado IDE assigns ports in the order that they appear in the I/O Ports window. You can adjust the assignment order by applying sorting techniques in the I/O Ports window prior to assignment.*

The direction in which you draw the rectangle dictates the I/O ports assignment order. I/O ports are assigned from the inside pin of the first rectangle coordinate selected. Creative definition of the area rectangles can provide useful pinout configurations from a PCB routing perspective.

Figure 3-26 shows I/O ports placed in an area.

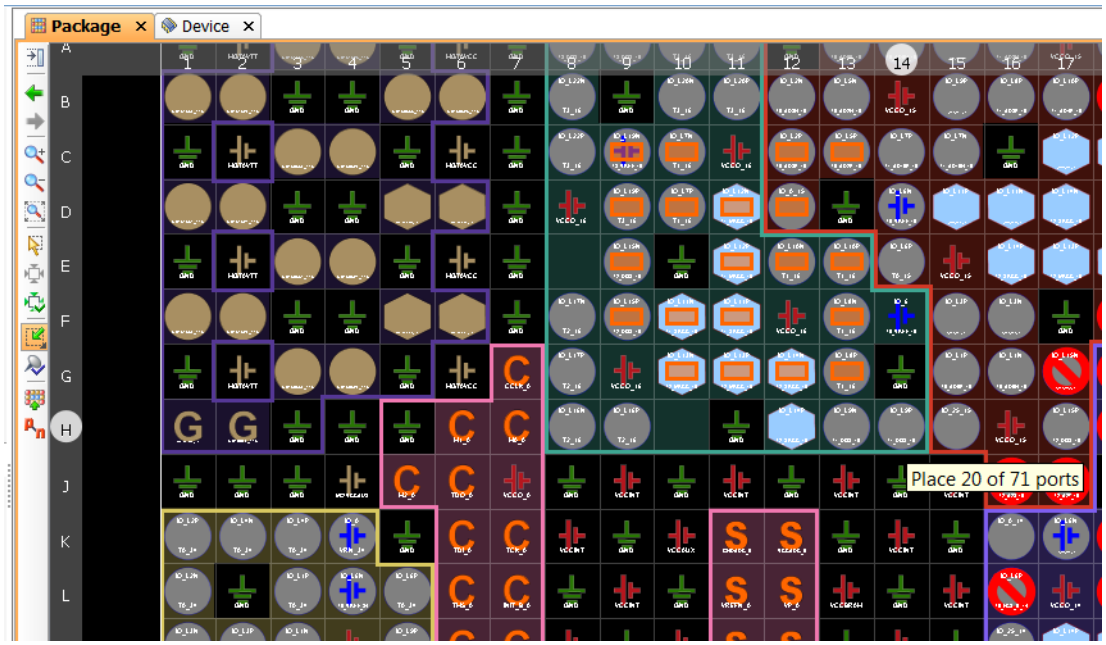

Figure 3-26: **I/O Ports Placed in an Area**

Swapping Previously Placed I/O Ports

To swap the location of two placed I/O ports that are already assigned:

1. Select two I/O ports from any of the available windows.
2. Select **Swap Locations** from the popup menu.

IMPORTANT: *If you are working in an implemented design and you swap two ports that are not yet fixed, swapping the ports fixes the ports and writes constraints to the XDC file.*

Moving Previously Placed I/O Ports

To move a port or groups of ports that are already assigned, select the port or group of ports, and drag them from one location to another. When you move a group of ports from one I/O bank to another, the Vivado IDE automatically finds suitable locations for the selected ports.

Note: This is similar to using the **Place I/O Ports in an I/O Bank** command.

Automatically Placing I/O Ports

You can automatically assign I/O ports to package pins. The Vivado IDE obeys I/O standard and differential pair rules and places global clock pins appropriately.

To automatically assign I/O ports:

1. In the I/O Ports window, select the I/O ports to place.
2. Select **Tools > I/O Planning > Auto-place I/O Ports**.

   Note: Alternatively, you can select **Auto-place I/O Ports** from the popup menu in the I/O Ports window.
3. In the Autoplace I/O Ports wizard (Figure 3-27), select the group of I/O ports to place, and click **Next**.

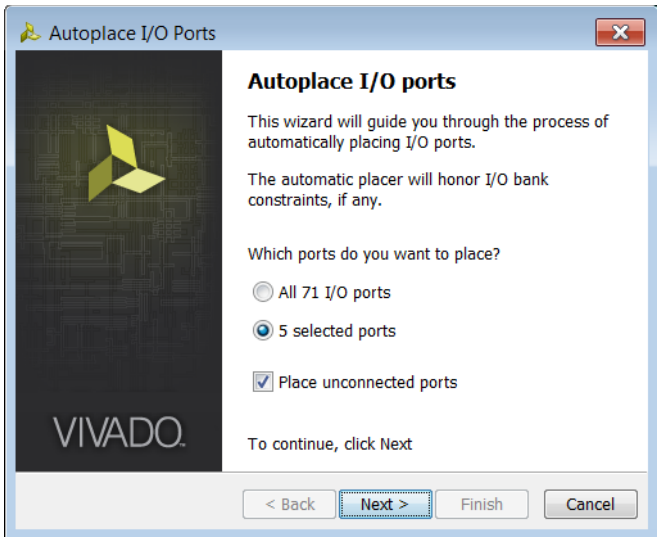

Figure 3-27: **Autoplace I/O Ports Wizard**

4. If you selected I/O ports that are already assigned to package pins, select an option in the Placed I/O Ports page (Figure 3-28), and click **Next**.

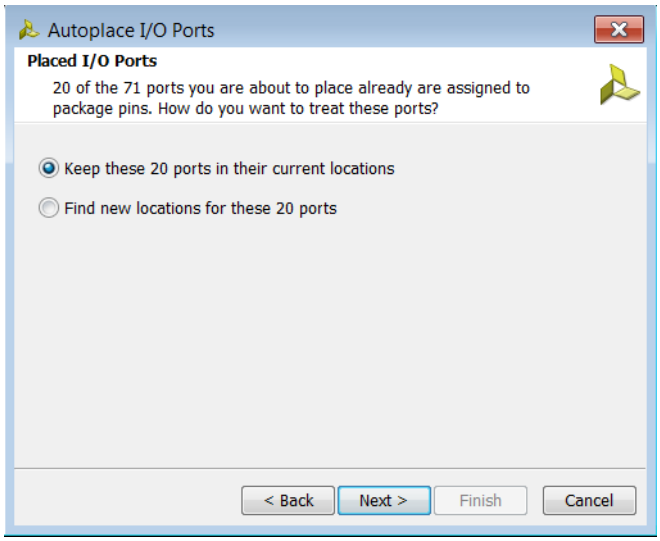

Figure 3-28: **Autoplace I/O Ports Wizard—Placed I/O Ports Page**

5. In the Summary page, click **Finish**.

Placing Gigabit Transceiver I/O Ports

To better manage GTs, the I/O planning windows group the two related I/O diff pairs and the GT logic object automatically during selection, placement, and moving. The GT objects are selected as one object and move together, which prohibits illegal assignment of the GT resources.

If the interactive DRCs are enabled, the noise sensitive I/O pins surrounding the GTXs are prohibited automatically during port placement. For more information, see Disabling or Enabling Interactive DRCs.

Removing I/O Placement Constraints

To remove placement constraints, select the placed logic, and select **Unplace** from the popup menu.

Running DRCs

Running DRCs is one the most critical steps in pin planning. DRCs check the current design against a specified set of design rule checks, or a rule deck, and report any errors or violations. This section describes the required steps for running I/O port and clock-related DRCs and viewing DRC violations.

Note: For advanced DRC control, see the *Vivado Design Suite Tcl Command Reference Guide* (UG835) [Ref 12].

Running I/O Port and Clock Logic Related DRCs

To select and run individual rules:

1. Select **Tools > Report DRC**.

   Note: Alternatively, you can click **Report DRC** from the Flow Navigator.

2. In the Report DRC dialog box (Figure 3-29), enter a name in the Results Name field.

   Entering a unique name makes it easier to identify the results for a particular run during debug in the DRC Violations browser. The output file name matches the entered name.

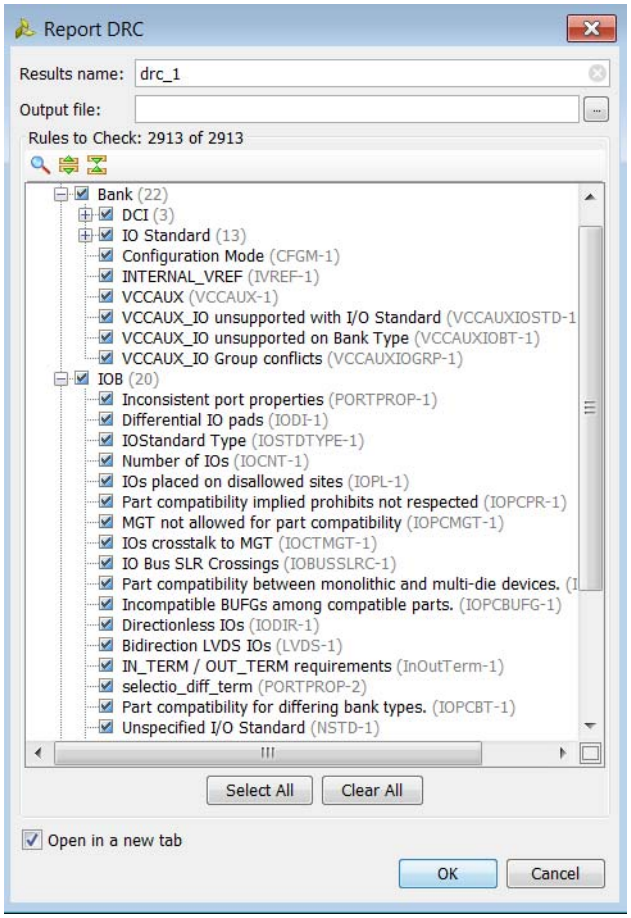

Figure 3-29: **Report DRC Dialog Box**

3. In the Rules to Check field, use the checkboxes to select the design rules to check for each design object. In this field, you can:
   - Expand the hierarchy using the **Expand All** toolbar button , or click the expand (+) button next to each category or design object.
   - Click **All Rules** to run a complete DRC, which runs all rules for all design objects.
   - Click a design object to run all DRCs for that object.
   - Click individual DRCs.
4. Click **OK** to run the selected DRCs.

Tcl Command for Running DRCs

Following is the associated Tcl command:

- **Tcl Command**: `report_drc`
- **Tcl Command Example**: `report_drc -ruledeck placer_checks -file C:/Data/DRC_Rpt1.txt`

Disabling or Enabling Interactive DRCs

During I/O planning, the Vivado IDE checks to ensure a legal pinout. However, complete sign-off DRCs are only run during Vivado implementation. Therefore, you need to run your design through Vivado implementation to ensure final legal pinouts.

The interactive I/O placement routines check common error cases during pin placement. You can toggle this capability on and off using either of the following methods:

- In the Device window or Package window, click the **Autocheck I/O Placement** toolbar button .
- Select **Tools > Options**. In the General Options, enable the **Automatically enforce legal I/O placement** option.

When you enable automatic checking, the tool does not allow placement of I/O ports on pins that cause a design issue. In **Place I/O Ports Sequentially** mode, if you attempt to place an I/O Port on a problematic pin, a tooltip appears that describes why the I/O port cannot be placed. The interactive DRCs are enabled by default.

IMPORTANT: *Many of these DRCs are only run on a synthesized or implemented design.*

The interactive I/O placement rules include:

- **Prohibiting**:
  - Placement on noise-sensitive pins associated with GTs or on I/O package pins that are potentially noise-sensitive.
  - I/O standard violations.
- **Ensuring**:
  - I/O standards are not used in banks that do not support them.
  - Banks do not have incompatible V_{CC} ports assigned.
  - Banks that need V_{REF} ports have free V_{REF} pins.
  - Proper assignment of global clocks and regional clocks (only with an imported netlist and XDC file).
  - Differential I/O ports are set to the proper sense pin.
  - No output pins are placed on input-only pins.

RECOMMENDED: *It is recommended that you begin your I/O port placement with DRCs enabled. See the device documentation for more information on I/O ports and clock region specifications.*

Viewing DRC Violations

If violations are found, the DRC window opens, as shown in Figure 3-30. The DRC window shows the rule violations, grouped under the various rule categories defined in the Report DRC dialog box. The rule violations are also categorized by severity and are color coded for quick review as follows:

- **Informational only**: Provides general status and feedback on design processing.
- **Warning**: Indicates that design results might be sub-optimal, because constraints or specifications might not be applied as intended.
- **Critical warning**: Indicates that certain user input or constraints will not be applied or do not adhere to best practices. It is highly recommended that you examine these issues and make changes.
- **Error**: Indicates an issue that renders design results unusable and cannot be resolved without your intervention. The design flow stops.

TIP: *You can toggle the **Hide Warnings and Informational Messages** toolbar button to turn off warnings and informational messages to see only the errors reported.*

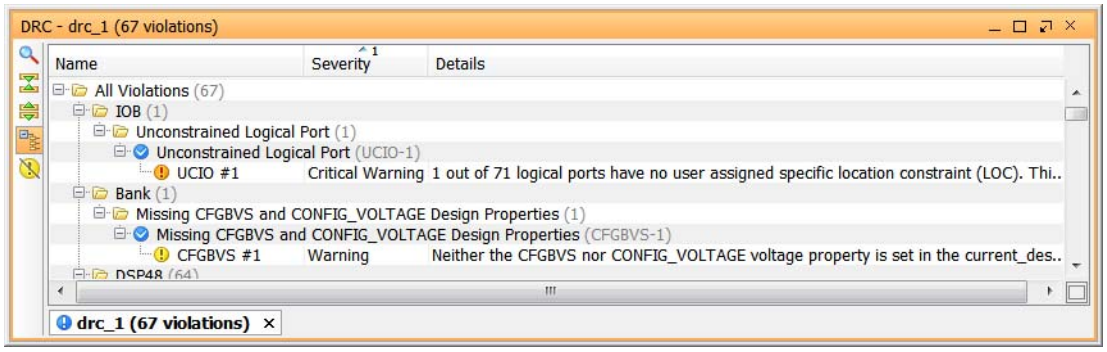

Figure 3-30: **DRC Violations**

Sorting DRC Violations

To sort DRC violations by severity, click the Severity column header as follows:

- Click the column header to sort in an increasing order.
- Click the column header again to sort in a decreasing order.

Note: For more information, see the *Vivado Design Suite User Guide: Using the Vivado IDE* (UG893) [Ref 11].

Viewing DRC Violation Properties

In the DRC window, right-click a violation message, and select **Violations Properties** from the popup menu. In the Violation Properties window, click the following views:

- **General**: Provides high-level information about the DRC rule violation, including type, severity, and description.
- **Details**: Provides information about the design elements that violate the rule. In addition, this view might include links to specific design objects that violate the DRC. Click these links to view the design object in the RTL Netlist window, Device window, Schematic window, or source RTL file.

Creating Custom DRCs

You can use the following Tcl commands to create custom DRCs: `create_drc_check`, `create_drc_ruledeck`, and `create_drc_violation`. For more information, see the *Vivado Design Suite Tcl Command Reference Guide* (UG835) [Ref 12] and *Vivado Design Suite User Guide: Using Tcl Scripting* (UG894) [Ref 16].

Querying DRCs

To get a list of the currently defined DRCs, use the `get_drc_checks` Tcl command. For more information, see the *Vivado Design Suite Tcl Command Reference Guide* (UG835) [Ref 12] and *Vivado Design Suite User Guide: Using Tcl Scripting* (UG894) [Ref 16].

Migrating to an RTL Design

After the I/O ports are defined and placed onto the package pins, you can migrate the I/O planning project to an RTL project. The port definitions are used to create a top module for the RTL design in either Verilog or VHDL, as specified. Differential pair buffers are added to the top module, and bus definitions are also included in the RTL. The project properties are changed to reflect the RTL project type.

IMPORTANT: *After migration, the RTL project cannot be converted back into an I/O planning project.*

To convert the project:

1. Select **File > Migrate to RTL**.

   Note: Alternatively, you can select **Migrate to RTL** from the Flow Navigator.

2. In the Migrate to RTL dialog box (Figure 3-31), set the following options, and click **OK**.
   - **Top RTL file**: Specify the Verilog (.v extension) or VHDL (.vhd extension) file to create for the top module of the design. The HDL file will include the module definition with port definitions, direction, and width for bus pins.
   - **Netlist format**: Specify Verilog or VHDL format for the top module.
   - **Write diff buffers**: Write the diff pair buffers as part of the top module definition. This preserves any differential pairs defined in the I/O planning project.

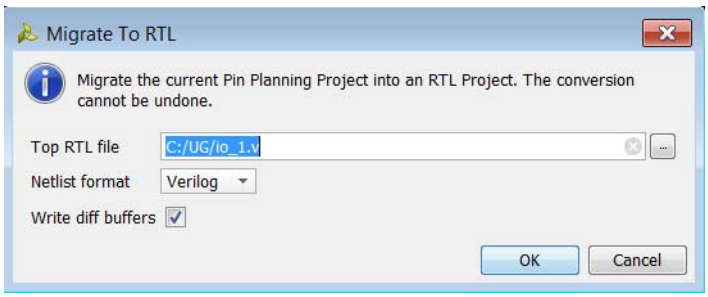

Figure 3-31: **Migrate to RTL Dialog Box**

After the I/O planning project is converted to an RTL project, you can begin adding sources to the project and working on your design. For more information, see the *Vivado Design Suite User Guide: System-Level Design Entry* (UG895) [Ref 1].

Working with SSN Analysis

The Vivado IDE provides analysis of the switching noise levels associated with the I/O of different devices. SSN analysis provides estimates of the disruption that simultaneously switching outputs can cause on other output ports in the I/O bank. SSN analysis incorporates I/O bank-specific electrical characteristics into the prediction to better model package effects on SSN.

I/Os are grouped into separate isolated I/O banks, each with its own unique power distribution networks, and each with unique responses to switching activity. Because power distribution networks within a packaged FPGA have different responses to noise, it is important to understand the I/O standards and number of I/Os in a design as well as the response of the device power system to this switching.

Xilinx characterizes all banks through three-dimensional extraction and simulation. This information is incorporated into SSN analysis. SSN analysis uses the expected switching profile of a device to predict how the switching affects the power network of the system and in turn, how other outputs in the I/O bank are affected.

IMPORTANT: *SSN analysis is the most accurate method available for predicting how output switching affects interface noise margins. The calculation and results are based on a range of variables. These estimates are intended to identify potential noise-related issues in your design and are not intended as final design sign-off criteria.*

Running SSN Analysis

To run SSN analysis:

1. Select **Tools > Report Noise**.

   Note: Alternatively, you can select **Report Noise** from the Flow Navigator.

2. In the Run SSN Analysis dialog box (Figure 3-32), set the following options, and click **OK**.

   - **Results Name**: Enter a name to identify the results in the Noise window.
   - **Export to File**: Export the analysis to an external report file. Enter an output file name, or browse and select a location. Specify the output format for the file as either **CSV** or **HTML**.
   - **Phase**: Considers clocking information available in the design to more accurately report SSN noise. Clocks must be defined using the `create_clock` and `create_generated_clock` Tcl commands. The period, phase shift, and duty cycle of the generated clocks have significant impact on SSN analysis. For more information, see Adding Phase Information to SSN Analysis.

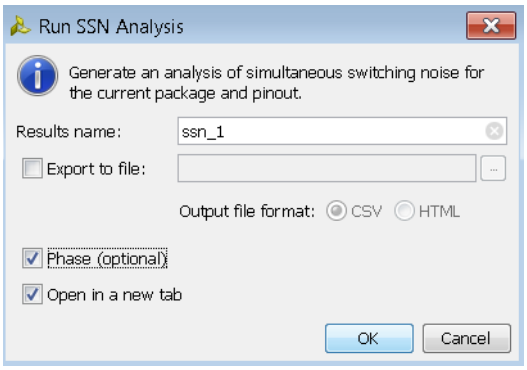

Figure 3-32: **Run SSN Analysis Dialog Box**

Viewing SSN Results

After the analysis is complete, the Noise window (Figure 3-33) opens.

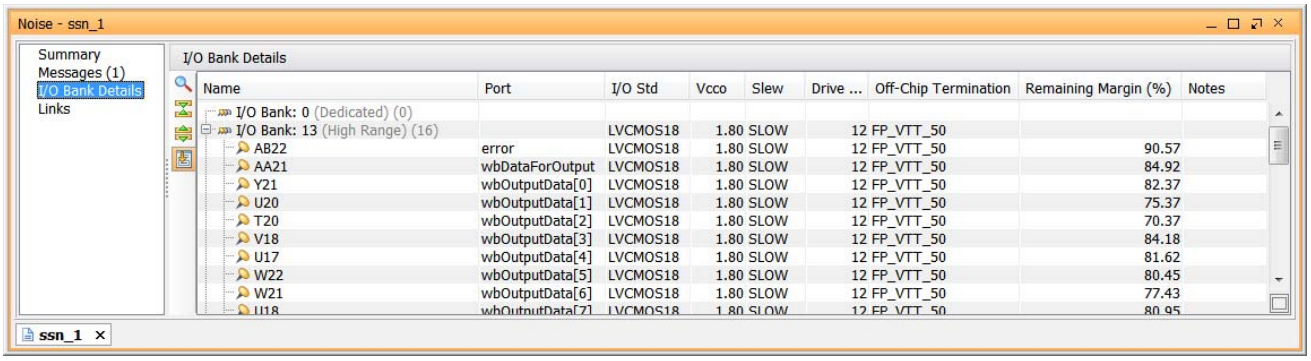

Figure 3-33: **Noise Window**

Click the links on the left side of the window to view different information about the SSN analysis. For example, click **I/O Bank Details** to view the following information:

- **Name**: Displays the I/O banks available in the device. Each I/O bank has pin icons indicating how full the bank is. A check mark indicates a passing result, and a red circle indicates a failure.
- **Port**: Displays the name of the user I/O in the FPGA design.
- **I/O Std, V_{CCO}, Slew, Drive Strength**: Displays the appropriate values for the port or bank.
- **Off-Chip Termination**: Displays the default terminations for each I/O standard, if one exists. Displays either **None** or a short description of the expected or defined off-chip termination style. For example, `FP_VTT_50` describes a far-end parallel 50 Ω termination to VTT termination style. The full list of termination styles is available in the *7 Series FPGAs SelectIO Resources User Guide* (UG471) [Ref 5].

For LVTTL (at 2mA, 4mA, 6mA, and 8mA) no termination is assumed. However, for LVTTL (at 12mA, 16mA, and 24mA) a far-end parallel termination of 50 ohms to VTT is assumed. As a result of this termination, the available noise margin is less for signals with drive strength of 12mA, or more, when compared to 2mA to 8mA. Xilinx 7 series FPGA devices use this assumption.

To change the settings, use either of the following methods:

- Use the CSV file import feature described in Importing a CSV File.
- In the I/O Ports table, select an item from the pull-down menu.

- **Remaining Margin %**: Displays the amount of noise margin that is left over after accounting for all SSN in the bank.
- **Notes**: Displays information about the I/O bank or groups.

IMPORTANT: *The SSN results are relative to the state of the design when the SSN analysis is run. It is not a dynamic report.*

Viewing I/O Bank Properties in SSN Results

You can select an I/O bank in the Noise window to display information about the I/O ports, pins, and groups assigned to the I/O bank in the I/O Bank Properties window. In the I/O Bank Properties window, you can:

- Select the **General** view to view information about the number and types of ports assigned to the I/O bank.
- Select the **Package Pins** or **I/O Ports** view to view the detailed information about the pins or ports within the bank, as shown in Figure 3-34.

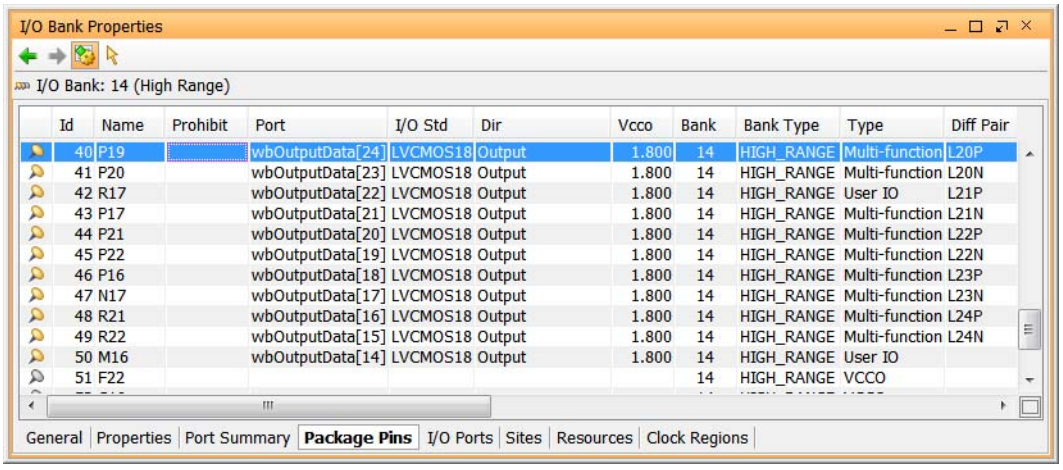

Figure 3-34: **I/O Bank Properties with Package Pins**

Improving SSN Results

To improve SSN results when a violation occurs:

- Use I/O standards that have a lower SSN impact for the failing group. Changing to a lower drive strength, a parallel-terminated DCI I/O standard, or a lower class of driver can improve SSN; for example, changing the SSTL Class II to an SSTL Class I.
- Spread the failing pins across multiple banks. This reduces the number of aggressive outputs on the power system of one bank.
- Spread the failing groups across multiple synchronous phases.
- Add phase information.

Adding Phase Information to SSN Analysis

You can increase margin in SSN analysis by adding phase information. By default, SSN analysis assumes that every output port toggles synchronously. This assumption covers the worst-case scenario that often yields an overly pessimistic SSN Analysis report. If clock information for the design is available, SSN analysis reports more accurate SSN noise.

To use this feature, enable SSN phase analysis using the following Tcl command:

```
report_ssn -phase
```

Enter clocking information using the `create_clock` and `create_generated_clock` Tcl commands. These commands provide the following required inputs to SSN analysis:

- Phase group

  Note: This groups generated clocks that belong to a single MMCM or PLL.
- Period
- Duty cycle
- Phase shift

  Note: This covers an absolute phase shift from zero degrees.

For more information, see the *Vivado Design Suite Tcl Command Reference Guide* (UG835) [Ref 12] and *Vivado Design Suite User Guide: Using Tcl Scripting* (UG894) [Ref 16].

TIP: *When you enable SSN phase analysis, the SSN Analysis report shows information in the Phases column.*

Following are important considerations:

- Multiple master clocks do not improve SSN results. There must be multiple phases *within* each master clock to reduce SSN results.
- A single port within a phase group does not improve SSN results. For each clock group or phase group, there must at least two ports.
- To minimize SSN noise, shift the clock transitions of one clock more than 700 picoseconds (ps) of another clock.
- Phase shift within a given phase group must be greater than 200 ps to improve SSN results.

Following are additional considerations:

- For large designs, running SSN analysis with phase support might take in the tens of minutes.
- Shifting 180 degrees does not improve SSN results. Although clocking information includes rising and falling transition information, SSN analysis does not include actual output logic of the ports. When a clock is transition from Low to High, the output of a port may go in either direction. To ensure conservative SSN reporting, the algorithm assumes 180 degrees is the same as zero phase shift. Due to the lack of information on the output ports, the analysis overestimates SSN noise for ports with 180 degree shift. In reality, the SSN actual reduces with a 180 degree shift, but the algorithm cannot account for the reduction.
- Only 50% duty cycle is supported, and non-conforming clocks are considered as asynchronous signals.

Adding Temperature Information to SSN Analysis

You can increase the accuracy of SSN analysis by specifying the temperature grade for the targeted part. To add temperature grade information, use one of the following Tcl commands, and then run SSN analysis.

```
set_operating_conditions -grade Commercial
set_operating_conditions -grade Industrial
set_operating_conditions -grade Military
set_operating_conditions -grade Q-Grade
set_operating_conditions -grade Extended
```

TIP: *To verify operating conditions. use the* `report_operating_conditions -grade` *Tcl command. To reset the temperature grade to the default, use the* `reset_operating_conditions -grade` *Tcl command.*

Following are additional considerations:

- By default, the temperature grade is based on the device used in the project.
- When running SSN analysis on an elaborated design, you cannot change the default temperature grade for the targeted part.
- Operating conditions are also used for power analysis. For information on operating conditions that impact power analysis, see *Vivado Design Suite User Guide: Power Analysis and Optimization* (UG907) [Ref 17].

IMPORTANT: *To increase the accuracy of SSN analysis results, you must use the correct temperature grade.*

Exporting I/O Pin and Package Data

You can export I/O pin and package pin information for the following purposes:

- **I/O Pin Information**: You can export the I/O port list to a file for use in RTL coding or PCB schematic symbol creation.
- **Package Pin Information**: When working with an elaborated, synthesized, or implemented design, you can export the device package pin information to a CSV file. The package pin section of the exported list is a good starting point for defining I/O port definitions in a spreadsheet format. The exported information includes information about all of the package pins in the device as well as design-specific I/O port assignments and their configuration. Added columns and user-defined values are preserved and exported to the output file. For information on the exported CSV file format, see Defining and Configuring I/O Ports.

To export the I/O ports list information:

1. Select **File > Export > Export I/O Ports**.
2. In the Export I/O Ports dialog box, specify the type of I/O port to generate and the path, and click **OK**.

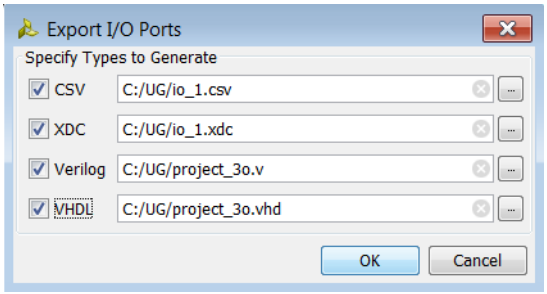

Figure 3-35: **Export I/O Ports Dialog Box**

Working with IBIS Models

The Input/Output Buffer Information Specification (IBIS) is a device modeling standard. IBIS allows for the development of behavioral models used to describe the signal behavior of device interconnects. These models preserve proprietary circuit information, unlike structural models such as those generated from SPICE (Simulation Program with Integrated Circuit Emphasis) simulations. IBIS buffer models are based on V/I curve data produced either by measurement or by circuit simulation.

IBIS models are constructed for each IOB standard, and an IBIS file is a collection of IBIS models for all I/O standards in the device. An IBIS file also contains a list of the used pins on a device that are bonded to IOBs configured to support a particular I/O standard, which associates the pins with a particular IBIS buffer model.

The IBIS standard specifies the format of the output information file, which contains a file header section and a component description section. The Golden Parser has been developed by the IBIS Open Forum Group [Ref 18] to validate the resulting IBIS model file by verifying that the syntax conforms to the IBIS data format.

When you export an IBIS model in the Vivado IDE, the tool outputs a .ibs file. This file comprises a list of pins used by your design, the signals internal to the device that connect to those pins, and the IBIS buffer models for the IOBs connected to the pins.

Exporting IBIS Models

To better understand the signal integrity at the system level, PCB designers often need to simulate the design with IBIS models. Designers must consider signal integrity issues such

as crosstalk, ground bounce, and SSN. IBIS models help characterize current-voltage (I-V) curves and parasitic information of the packaged device.

TIP: *You can download generic IBIS models from the Download Center [Ref 19] on the Xilinx website.*

From the Vivado IDE, you can generate IBIS models from the design and per-pin package data. The Vivado IDE uses the netlist and implementation details from the design, and combines that information with the available per-pin parasitic package information to create a custom IBIS model for the design.

With an elaborated, synthesized, or implemented design open, you can export an IBIS file for use in analyzing the design as follows:

1. Select **File > Export > Export IBIS Model**.
2. In the Export IBIS Model dialog box (Figure 3-36), set the following options, and click **OK**:
   - **Output File:** Specify the filename and path for the output IBIS file.
   - **Include all models**: Include all available I/O buffer models for this device. By default, only buffer models used in the design are included.
   - **Disable per pin modeling**: Disable inclusion of the per-pin modeling of the package. This is the path from the die pad of the device to the pin of the package. With per-pin modeling disabled, the package is reduced to a single RLC transmission line model applied to all pins and defined in the [Package] section of the IBIS file.
   - **Maximum length of signal names**: Truncate signal names to the specified limit:
     - **40**: Truncate signal names to 40 characters, which is supported by IBIS version 4.2 as the default.
     - **20**: Truncate signal names to 20 characters.
     - **Unlimited**: Do not truncate signal names.
   - **Updated generic IBIS model file**: Optionally, provide an IBIS model file for the device. This is used to override the IBIS models found in the installation under the parts directory.

IMPORTANT: *The IBIS model file is required for devices that do not have IBIS models included with software installation.*

   - **Updated parasitic package data file**: Optionally, provide a parasitic package file (.pkg extension) file to use for per-pin extractions. This is used to override the parasitic package file found in the installation hierarchy under the parts directory.

IMPORTANT: *The parasitic package file is required for devices that do not have IBIS models included with software installation.*

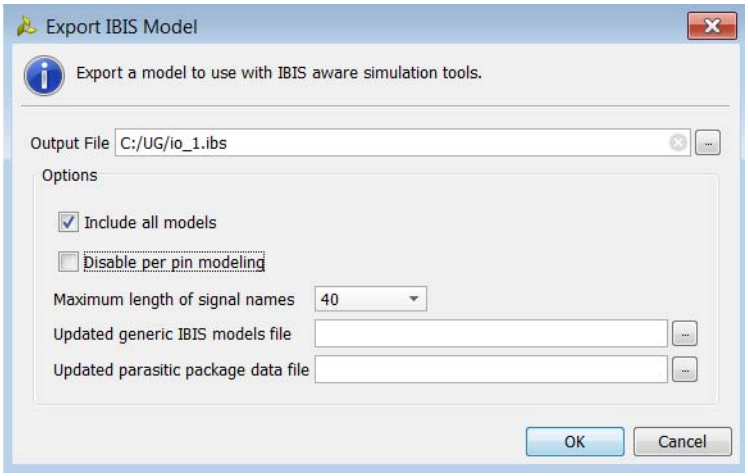

Figure 3-36: **Export IBIS Model Dialog Box**

Interfacing with the PCB Design

The Vivado pin planner provides you with an effective way to select the pin assignments for the design. Choosing correct resources enables a faster and cleaner design process. The following recommendations help prevent board layout, pin assignment, and FPGA resource conflict.

Note: For more information on PCB and pin planning for 7 series and Zynq® device families, see the *7 Series FPGAs PCB Design and Pin Planning Guide* (UG483) [Ref 20] and *Zynq-7000 All Programmable SoC PCB Design and Pin Planning Guide* (UG933) [Ref 21].

Part Compatibility

Set the part compatibility for your design as described in Defining Alternate Compatible Parts. The Vivado IDE identifies the pins that are common to all selected alternate parts and assigns prohibit constraints to pins that are not common to all devices, eliminating the possibility of using these pins. For more information, see the product table for the targeted device.

DRC

To check the overall integrity of the I/O assignments, run DRCs as described in Running DRCs. It is important that you resolve all warnings and errors reported by the DRCs before starting board planning.

SSN Analysis

To generate an estimate of potential noise disruptions, run SSN analysis as described in Working with SSN Analysis. It is recommended that you address noise-related issues before starting board planning. For more information, see the *7 Series FPGAs SelectIO Resources User Guide* (UG471) [Ref 5] and *7 Series FPGAs Memory Interface Solutions User Guide* (UG586) [Ref 22].

IBIS Simulation

To run IBIS simulations, use the IBIS file generated by the Vivado IDE as described in Working with IBIS Models.

CSV Export

After running DRCs and SSN analysis, export the CSV file to assist with board planning as described in Exporting I/O Pin and Package Data.

Supported Third-Party PCB Tools

Xilinx supports Cadence Allegro FPGA System Planner and Mentor Graphics I/O Designer. These tools provide a way to optimize the pin assignments in the context of the entire board. For more information, see the third-party tool documentation.

I/O Pin Planning and Clock Planning with SSI Technology Devices

I/O pin planning and clock planning are critical when working with stacked silicon interconnect (SSI) technology. Because SSI technology devices exist in large dies, poor placement can create longer routes that increase power consumption and reduce performance. For information on pinout selection and clocking, see the *Large FPGA Methodology Guide* (UG872) [Ref 2].

Appendix A

I/O Port Lists in CSV File Format

CSV File

A CSV file is a standard file format used by FPGA and board designers to exchange information about device pins and pinout. For more information, see Importing a CSV File and Exporting I/O Pin and Package Data in Chapter 3.

The CSV columns are:

- **I/O Bank**: The I/O Bank in which the pin is located. The tool fills in this field for all pins in the device. Values are a number or blank. This is not required in the input CSV file.
- **Pin Number**: The name (or location) of the package pin. The tool writes this out for all pins in the device. This is not required in the input file. If used for input, it is used to define placement. Values are legal pins in the device.
- **IOB Alias**: An alternate part name for the package pin. This field is specified by the tool and is unused if specified in the input CSV file.
- **Site Type**: The pin name from the device data sheet. This field is specified by the tool and is unused if specified in the input CSV file.
- **Min/Max Trace Delay (ps)**: The distance between the pad site of the die and the ball on the package, in picoseconds. This is specified by the tool to help the board engineer match trace delays. The Trace Delay fields are in the output file only. They are not expected in the input file.
- **Trace Length (um)**: Specifies the length of the internal trace between the package pin and the die pad.
- **Prohibit**: Certain sites can be prohibited for many reasons to prevent user I/O from being added to the site. For example, sites can be prohibited to:
  - Prohibit ease board layout issues.
  - Reduce crosstalk between signals.
  - Ensure that a pinout works between multiple FPGAs in the same package.

  Note: In the XDC file, this is represented by a PROHIBIT property.

- **Interface**: An optional user-specified grouping for an arbitrary set of user I/O. For example, this field provides a means to specify a relationship for the data, address, and enable signals for a memory interface. Values are a text string or blank.
- **Signal Name**: The name of the user I/O in the FPGA design. Values are a string or blank for an unassigned Package Pin.
- **Direction**: The direction of the signal. Values are IN, OUT, INOUT, or blank when a user I/O is not assigned to the site.
- **DiffPair Type**: This value instructs the software about which pin is the N side of a differential pair, and which pin is the P side. This is used for differential signals only. The tool uses this column instead of a naming convention to determine which pin is the N side of the pair, and which pin is the P side. Values are P, N, or blank when a user I/O is not assigned to the site.
- **DiffPair Signal**: Specifies the name of the other pin in the differential pair. Values are the name of the user I/O or blank when unused.
- **I/O Standard**: I/O standard for a specific user I/O. When this field is blank for a user I/O, the tool uses the appropriate device defaults. Values are a legal I/O standard for the user I/O in the device or blank.
- **Drive**: Drive strength of the I/O standard for a specific user I/O. Not all I/O standards accept a drive strength. If this field is blank, the tool uses the default. Values are a number or blank.
- **Slew Rate**: Slew rate of the I/O standard for a specific user I/O. Not all I/O standards accept a slew rate. If this field is blank, the tool uses the default. Values are FAST and SLOW.
- **Pull Type**: Specifies the pull type for the selected port. When using tristate output (OBUFT) or bidirectional (IOBUF) buffers, the output can have a weak pull-up resistor, a weak pull-down resistor, or a weak “keeper” circuit. For input (IBUF) buffers, the input can have either a weak pull-up resistor or a weak pull-down resistor.
- **Phase**: Specifies the phase of an I/O relative to the phase of other I/O in the bank in cases of a synchronous phase offset.
- **Board Signal**: The name of the signal coming into the I/O from the board-level design.
- **Board Voltage**: Specifies the voltage level of the signal coming into the I/O from the board-level design.
- **BUFIO2_REGION**: Defines the BUFIO2 clocking region a port is related to.
- **IN_TERM/OUT_TERM**: Defines the optional IN_TERM or OUT_TERM driver impedance properties. This is most commonly left blank and is not yet supported for production devices. Using this terminal definition overrides the SLEW and DRIVE STRENGTH properties and is not supported in SSN calculations.
- **OFFCHIP_TERM**: Specifies the external board level termination of the I/O. This is used for SSN calculations. If the field is left blank, the tool uses the expected terminations in

the SSN calculations and shows this expected termination by default in the SSN report and I/O Ports table.

Note: For information on the expected terminations as well as the corresponding shortened names that display in the tool, see the *7 Series FPGAs SelectIO Resources User Guide* (UG471) [Ref 5].

IMPORTANT: *When reading the CSV file, the Vivado® tools retain undefined column values, reporting them as user-defined columns in the I/O Ports window.*

Differential Pairs in the CSV File

There are several properties used to define differential pairs in the CSV file:

- Signal Name
- DiffPair Signal
- DiffPair Type
- I/O Standard

The other values in the CSV file are used to validate a diff pair, to ensure they are fully compatible, but they are not used to define the pair. You can define differential pairs in the CSV file in the following ways:

- **Diff Pair**: This is a direct definition of the two signals which make up a differential pair. Two port entries each have DiffPair Signal values linking to the Signal Name of the other and have complementary DiffPair Type values, one N and one P. The tool checks to ensure that the other properties such as I/O Standard are compatible when forming the diff pair.
- **Single-link Diff Pair**: Two port entries with complementary DiffPair Type values (one N, one P), but only one port has a DiffPair Signal linking to the other Signal Name. The tool creates the differential pair if all other properties are compatible.
- **Single Port Diff Pair**: A single port entry with a differential I/O Standard, a DiffPair Type value, and a DiffPair Signal that does not otherwise appear in the CSV file. The tool creates the opposite side of the differential pair (the N or P side) with all properties matching those on the original port.
- **Inferred Diff Pair**: Two ports entries with differential pair I/O standards (for example, DIFF_HSTL, DIFF_SSTL) and Signal Names that infer the N and P side. The tool infers a differential pair if all other properties are compatible.

Appendix B

Additional Resources

Xilinx Resources

For support resources such as Answers, Documentation, Downloads, and Forums, see the Xilinx® Support website at:

www.xilinx.com/support

For a glossary of technical terms used in Xilinx documentation, see:

www.xilinx.com/company/terms.htm

Solution Centers

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

References

1. *Vivado® Design Suite User Guide: System-Level Design Entry* (UG895)
2. *Large FPGA Methodology Guide* (UG872)
3. *7 Series FPGAs Packaging and Pinout Product Specification* (UG475)
4. *Zynq®-7000 All Programmable SoC Packaging and Pinout Specifications* (UG865)
5. *7 Series FPGAs SelectIO Resources User Guide* (UG471)
6. *Vivado Design Suite User Guide: Designing with IP (*UG896)
7. *Vivado Design Suite User Guide: Design Analysis and Closure Techniques* (UG906)
8. *LogiCORE IP Clocking Wizard Product Guide* (PG065)
9. *Vivado Design Suite User Guide: Synthesis* (UG901)

10. *Vivado Design Suite User Guide: Implementation* (UG904)
11. *Vivado Design Suite User Guide: Using the Vivado IDE* (UG893)
12. *Vivado Design Suite Tcl Command Reference Guide* (UG835)
13. *Constraints Guide* (UG625)
14. *7 Series FPGAs Configuration User Guide* (UG470)
15. *7 Series FPGAs Clocking Resources User Guide* (UG472)
16. *Vivado Design Suite User Guide: Using Tcl Scripting* (UG894)
17. *Vivado Design Suite User Guide: Power Analysis and Optimization* (UG907)
18. IBIS Open Forum Group (www.eda.org/ibis)
19. Download Center on the Xilinx website (www.xilinx.com/support/download/index.htm)
20. *7 Series FPGAs PCB Design and Pin Planning Guide* (UG483)
21. *Zynq-7000 All Programmable SoC PCB Design and Pin Planning Guide* (UG933)
22. *7 Series FPGAs Memory Interface Solutions User Guide* (UG586)
23. Vivado Design Suite Video Tutorials (www.xilinx.com/training/vivado/index.htm)
24. Vivado Design Suite Documentation (www.xilinx.com/support/documentation/dt_vivado2013-2.htm)