**Vivado Design Suite User**

**Guide**

***I/O and Clock Planning***

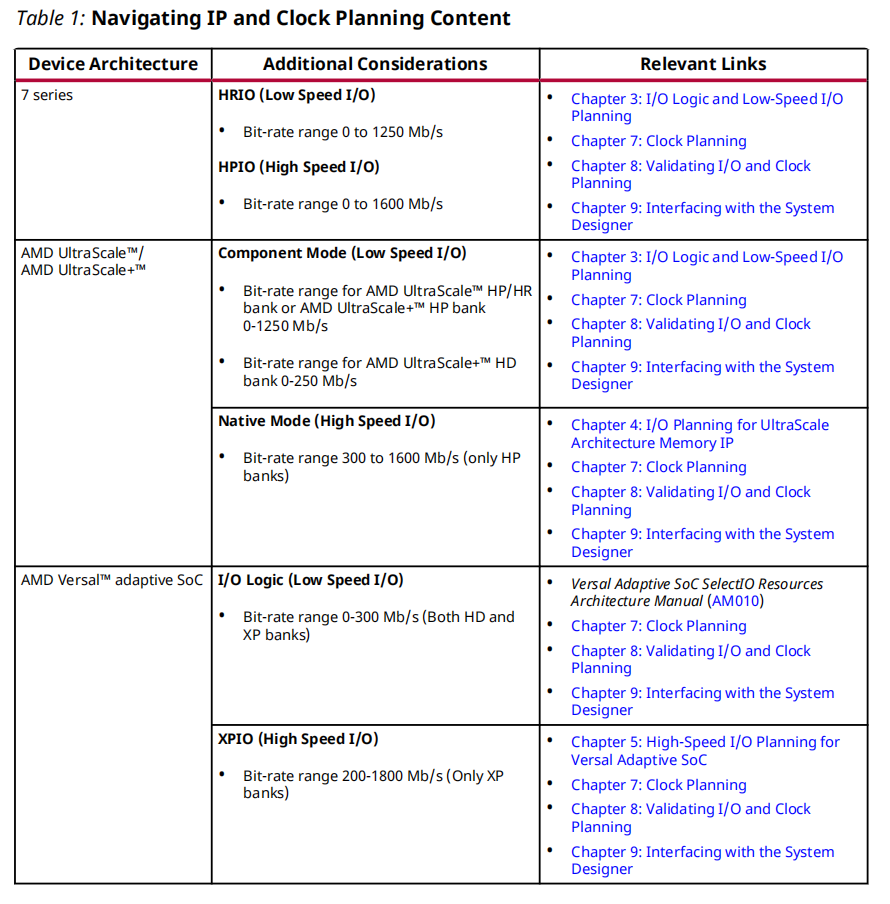
**UG899 (v2024.1) May 30, 2024**

**Chapter 1: Introduction**

I/O and clock planning is the process of defining and analyzing the connectivity between the FPGA/adaptive SoC and the printed circuit board (PCB) and assigning the various interconnect signals to physical pins of the device. By considering the data flow from the PCB to the FPGA/adaptive SoC die, you can achieve optimal pinout configurations quickly, thus reducing internal and external trace lengths as well as routing congestion.

* 1. **Navigating Content by Design Process**

System and Solution Planning



* 1. **I/O and Clock Planning Stages**

Digitally controlled impedance (DCI)cascade and internal voltage reference (VREF). See the Board and Device Planning section in the UltraFast Design Methodology Guide for FPGAs and SoCs (UG949).

You can perform I/O planning at any stage in the design flow. For example, you can begin the I/O assignment process from a top-level ports list, a register transfer level (RTL) design. Whenever possible, it is best to perform I/O assignment **with a synthesized design**.

I/O Planning can be done in many different ways. Using Advanced I/O Planner that can auto-place all the I/O interfaces to maximize the clocking and I/O architecture. If you need to place an individual I/O, the classic pin planning tools that write out pin constraints to an XDC file are still supported. Finally, you can also design your pin plan with a user-defined XDC file.

Certain types of IP, such as Memory IP, gigabit transceivers (GT), AMD High Speed IO IP, PCI Express® (PCIe), and Ethernet interfaces have I/O ports associated with them. You must properly configure this IP using the IP capabilities in the Vivado Design Suite prior to beginning the I/O planning process. Because these interfaces are usually the most timing-critical, use this IP as the starting point when considering the device pin assignments.

**1.2.1 I/O and Clock Planning Design Flow**

**Pre-RTL I/O Planning**

You can create an empty I/O planning project to enable early device exploration and initial I/O port assignment before the design source files are available. This enables PCB and FPGA designers to agree on an early pinout definition, which can eliminate iterations related to device pinout changes later in the design cycle. Using the I/O planning project, you can:

• Import device and I/O port assignments from the PCB designer or create I/O ports manually.

• Export device and I/O port assignments to hand off to the PCB designer or 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.

• Create a Verilog or VHDL module definition for the top-level of the design based on your port definitions.

**Note**: For information on creating an I/O planning project, see section I/O Planning Projects in the Vivado Design Suite User Guide: System-Level Design Entry(UG895). For information on importing pin assignments defined by the PCB designer or from another Vivado Design Suite project, see **Defining and Configuring I/O Ports**.

**RTL I/O Planning**

You can perform I/O planning in an elaborated RTL project. In an elaborated design, the Vivado tools provide basic DRCs to check port assignments, I/O standards, clock resources, and other design details. You can do initial I/O and clock planning with the elaborated design and export device and I/O port assignments for use in PCB schematic symbol generation or save the constraints in an XDC file for use during synthesis or implementation.

**Netlist I/O Planning**

You can also perform I/O planning with a synthesized netlist. With this method, you use either a synthesized RTL project or a netlist project created with a post-synthesis netlist. Whenever possible, use a synthesized design to perform I/O and clock planning.

**Final I/O Validation with an Implemented Design**

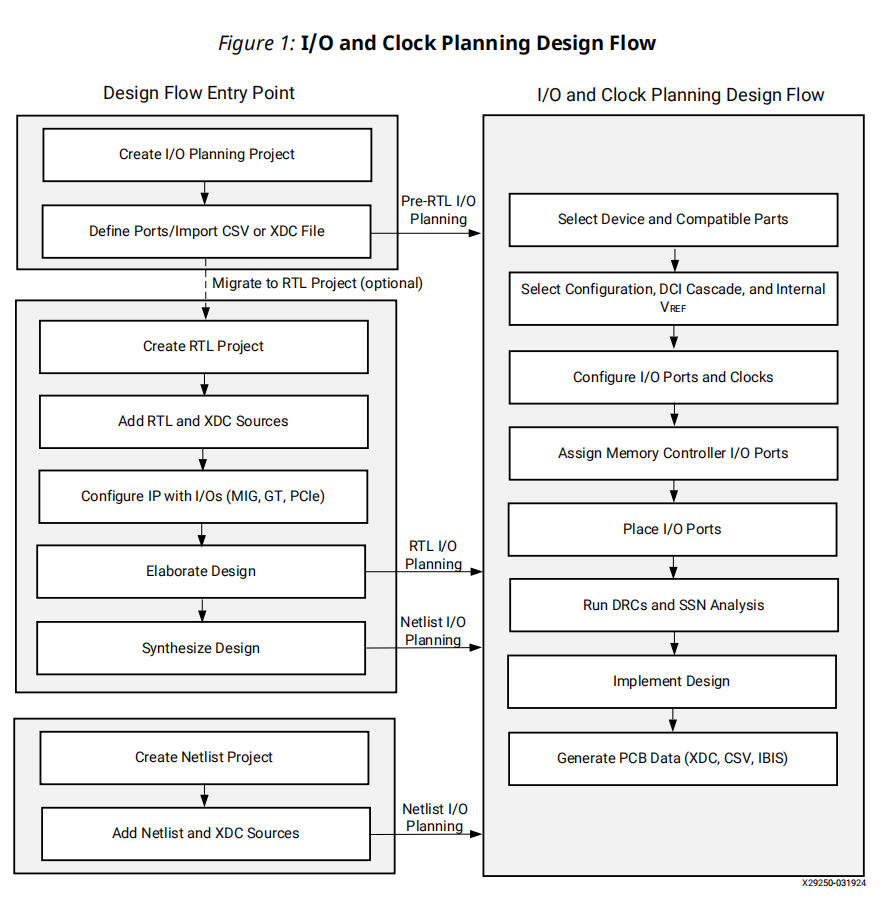
You must use a fully implemented design to validate the final valid I/O pinout and clock configuration. Proper clock resource validation requires fully routed implementation of all clocks. You can examine the implementation reports for I/O and clock-related messages. Finally, double check the I/O port assignments with the PCB designer to ensure that the FPGA is correctly defined for the system-level design.

**RECOMMENDED**:

To check clock logic, AMD recommends validation with a synthesized design.

To check clock timing, AMD recommends validation with an implemented design.

**1.2.2 I/O and Clock Planning Design Flow Steps**



The project design flow starts with an empty I/O planning project, an RTL design project, or a post-synthesis netlist project. Using any of these project types, you can perform the following steps in the I/O and clock planning design flow:

**1. Select Device and Compatible Parts**

For designs using stacked silicon interconnect (SSI) technology, see the UltraFast Design Methodology Guide for FPGAs and SoCs (UG949).

**2. Select Configuration, DCI Cascade(级连、瀑布), and Internal VREF**

An AMD device must be configured each time it is powered up. The bit-stream is loaded into the device through special configuration pins that enable different configuration modes. For Versal adaptive SoC, a device image is loaded. It is important to determine and plan for the configuration modes before beginning I/O assignment. The configuration mode not only determines the connectivity of certain pins, it also determines the VCCO voltage required for the I/O bank that includes multi-function pins.

Depending on the I/O standard, digitally controlled impedance (DCI) can control the output impedance of a driver or add a parallel termination to the driver, receiver, or both to match the characteristic impedance of a transmission line and improve signal integrity. DCI uses two multi-purpose reference pins in each I/O bank to control the impedance of the driver or the parallel-termination value for all of the I/Os in the bank.

Single-ended I/O standards with a differential input buffer require reference voltage (VREF). You can generate an internal VREF using the INTERNAL\_VREF constraint, which eliminates the need to provide a particular reference voltage supply rail on the PCB. In 7 series and AMD UltraScale™ architecture, this can free multi-purpose VREF pins in a given I/O bank for other I/O port assignments.

通过 INTERNAL\_VREF constraint可将具有差分输入缓冲的一个端设置为固定VREF，形成单端输入。 可减少PCB上需要提供的参考电源，同时也省出配置引脚而另作它用。

**3. Configure I/O Ports and Clocks**

The I/O ports on the device support multiple I/O-related **constraints, such as** IOSTANDARD, SLEW, and DRIVE.

AMD devices are subdivided into columns and rows of clock regions. A clock region contains configurable logic blocks (CLBs), I/O banks, digital signal processing (DSP) slices, block random access memory (RAM), interconnect, and associated clocking resources. Each I/O bank contains clock-capable input pins to bring system or board clocks onto the device and into clock routing resources. You must plan the use of these clock resources to distribute the clocks in your design across your device.

**RECOMMENDED:** AMD recommends 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.You can also generate clocking with an I/O interface design through the Advanced I/O Wizard.

**4. Assign Memory Controller I/O Ports**

The Memory IP defines a memory controller using a pre-engineered controller and physical layer (PHY) for interfacing FPGA designs to support external memory devices.You must define the I/O physical pin assignments for the gigabit transceiver (GT), PCIe technology, and Memory IP as part of IP customization when the core is added to the design.

**5. Place I/O Ports**

The Advanced I/O Planner is also a tool for nibble and bank-level pin placement. This tool allows for automatic placement that can understand all the I/O interfaces in an XPIO bank and place them effectively all at once.

**6. Run DRCs and SSN Analysis**

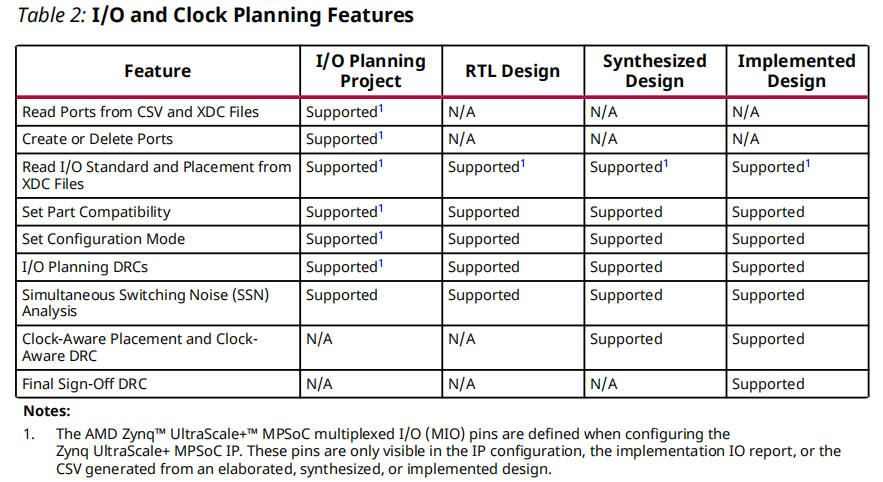
**RECOMMENDED:** AMD recommends that you run DRCs and SSN analysis after synthesis, prior to implementation, as well as after implementation. This enables you to catch issues early in the design cycle.

**7. Implement Design**

**8. Generate PCB Data (XDC, CSV, IBIS)**

When you complete the steps in the I/O and clock planning flow, you can return the pinout, along with device models for signal integrity analysis, using a comma-separated values (CSV) file and I/O buffer information specification (IBIS) models.

**1.2.3 I/O and Clock Planning Features**



* 1. **I/O and Clock Planning Using the Platform Board Flow**

For more information on the platform board flow, see section The Vivado Design Suit Board Flow in the Vivado Design Suite User Guide: System Level Design Entry (UG895).

* 1. **I/O and Clock Planning for SSI Technology Devices**

I/O and clock planning are critical when working with stacked silicon interconnect (SSI) technology.

* 1. **I/O and Clock Planning for IP with I/O Ports**

You must properly configure this IP using the IP capabilities in the Vivado Design Suite prior to beginning the I/O planning process. Because these interfaces are usually the most timing critical, use this IP as the starting point when considering the device pin assignments. In addition, use an RTL or synthesized design for the I/O pin planning process when using this IP. Define the I/O physical pin assignments for the GT, PCIe IP, Ethernet, and Memory IP as part of IP customization when the core is added to the design.

* 1. **I/O Planning for Zynq UltraScale+ MPSoCs**

The multiplexed I/O (MIO) pins do not show up in the user design or constraints. The I/O Planning Project does not show the utilization of the MIO ports, nor does it

write out the placement for the MIO pins. The only way to see a full list of the used pins for a schematic or for communicating with the board designer, is by using the **File → Export → Export I/O Ports** command to generate a CSV file. AMD recommends using an HDL project for all pin planning in Zynq UltraScale+ MPSoCs.

* 1. **I/O Planning for UltraScale and UltraScale+**

In 7 series devices, the bank types were **High Range** and **High Performance**; in UltraScale architecture devices, the names are **High Density** (replacing High Range), and **High Performance**. For more information, see section Introduction to High Density I/O Banks in the UltraScale Architecture SelectIO Resources User Guide (UG571).

* 1. **I/O Planning for Versal Adaptive SoC**

The two primary types of I/O in Versal adaptive SoCs are high-performance XP I/O (XPIO) and high density HD I/O (HDIO). The XPIO includes dedicated logic to support high-speed interfaces with voltage ranges between 1.0 V and 1.5 V. The HDIO supports interfaces with voltages ranging from 1.8 V to 3.3 V. The HDIO provides logic for both single data rate (SDR) and double data rate (DDR) interfaces

at reduced clocking speeds. See Versal Adaptive SoC SelectIO Resources Architecture Manual (AM010) for architecture information and Vivado Design Suite Properties Reference Guide (UG912) for detailed information on Advanced IO Wizard/Advanced IO Planner. HDIO and XPIO banks do not have overlapping

voltages or I/O standards.

All Versal devices have configurable SelectIO interface drivers and receivers, supporting a wide variety of standard interfaces. The robust feature set includes programmable control of **output strength** and **slew rate**, **on-chip termination,** and an internally generate a reference voltage (**INTERNAL\_VREF**). Each Versal device contain **XPIO banks that contains 54 SelectIO™ pins** and can implement both single-ended and differential I/O standards. XPIO banks support the highest speed interfaces powered at or below 1.5 V. Some Versal devices contain HDIO banks that can interface with voltage levels between 1.8 V and 3.3 V. **The HDIO banks contain 22 SelectIO pins** that can implement both single-ended I/O standards and differential I/O standards. Every SelectIO™ IOB resource contains input, output, and tristate drivers. The SelectIO™ pins can be configured to various I/O standards, both single-ended and differential.

• Single-ended I/O standards are, for example, LVCMOS, LVTTL, HSTL, SSTL, HSUL, LVSTL, and POD.

• Pseudo['sju:dəʊ，伪]-differential standards are, for example, differential HSTL, POD, HSUL, LVSTL, and SSTL.

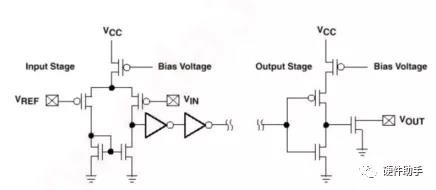
• LVDS compatible.

1、TTL电平一般过冲都会比较严重，可以在始端串22欧或33欧电阻（因为TTL电路的输出阻抗大约为17欧左右，从阻抗匹配的角度解释）；TTL电平输入脚悬空时内部认为是高电平。常见的TTL电平有5V TTL，3.3V LVTTL，2.5V LVTTL，1.8V LVTTL等。

2、CMOS输入引脚的最大输入电流非常小，只有1微安左右（最多几微安），因此选用1M欧作为上拉电阻或下拉电阻就可以。常见的CMOS电平有5V CMOS，3.3V LVCMOS，2.5V LVCMOS，1.8V LVCMOS，1.5V LVCMOS，1.2V LVCMOS，0.8V LVCMOS等。

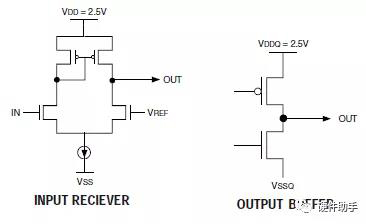
3、GTL(Gunning Transceiver Logic)。GTL输入电路是一个电压比较器，输入电压同一个外部连接的参考电压进行比较，输入门限设计为精确的窗口电压，可以提高最大的抗噪性能。输出电路是一个漏极开路N通道器件，当电路关闭时输出电压被上拉到末端匹配电压VTT，当输出电路打开时，器件可以吸收40mA的电流，可以产生最大的输出电压0.4V。输出电阻为25欧姆，输入输出被设计为与VCC的电压独立，器件可以工作在5V、3.3V，甚至是2.5V的VCC电压。

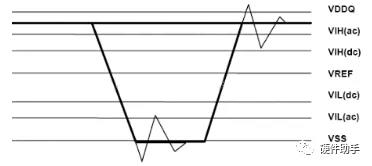
GTL和GTL+信号的参考电平Vref为信号上拉电平的2/3，这是同GTL电平的特点相关的，GTL信号的低电平一般为上拉电平的1/3左右，当GTL信号的参考电平设置为上拉电平的2/3时信号的高低电平有最大的抗噪冗余量，可以得到最佳的传输效果。现在很多厂家提供的GTL芯片的Vref都是可以通过外部进行调整，提供最佳的信号传输要求。同时因为GTL的输入阈值电平都很小，可以提供大的噪声容限，而小的输出电平提供的信号变化也很小。这些对信号的完整性有利。GTL＋的信号的电平更高，有更大的驱动能力，一般对于重负载情况下使用GTL＋的效果会更好一些。

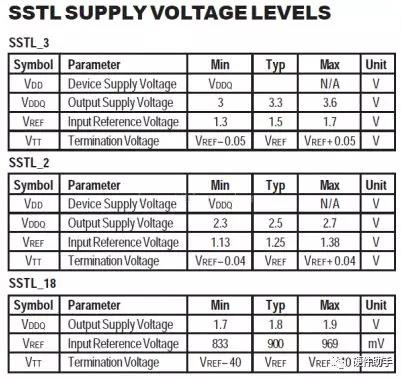


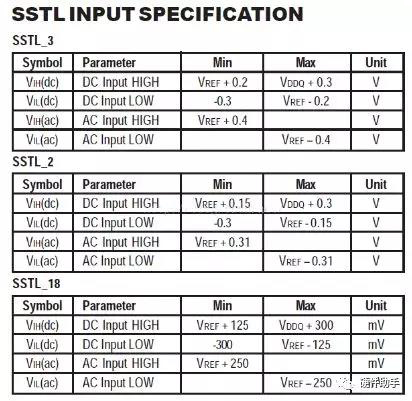
GTL图

1. SSTL即Stub Series Termination Logic，分为SSTL\_3（3.3V）、SSTL\_2（2.5V）、SSTL\_18（1.8V）、SSTL\_15（1.5V）（对应的VREF=VTT分别为1.5V、1.25V、0.9V、0.75V），对应不同的供电电压，SSTL是传输线终端匹配的，因此SSTL具有输出阻抗和匹配方法的要求。SSTL的输入是一个差分比较电路，一端为输入，另一端为参考电压VREF。DDR使用的就是SSTL电平标准。









SSTL图

5、HSTL即High Speed Transceiver Logic，其最主要用于高速存储器读写。在中频区域（100~180MHz），可供选择的单端信号IO结构有：HSTL、GTL/GTL+、SSTL、LVTTL；在180MHz以上，HSTL是唯一可用的单端IO接口。QDR使用的就是HSTL电平标准

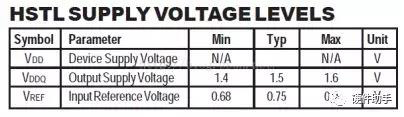
JEDEC定义了四种驱动模式：Class I~IV，其区别仅在于输出电流的不同：

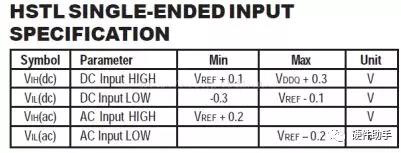
  •   Class I：IOH≥8mA，IOL≥-8mA；并行终端负载

  •   Class II：IOH≥16mA，IOL≥-16mA；串行终端负载

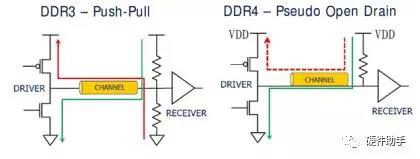
  •   Class III：IOH≥8mA，IOL≥-24mA；并行终端负载

  •   Class IV：IOH≥8mA，IOL≥-48mA；并行终端负载





6、POD（伪漏极开路）和SSTL的最大区别在于接收端的终端电压（POD为VDDQ，SSTL为VDDQ/2）。POD可以降低寄生引脚电容和I/O终端功耗，并且即使在VDD电压降低的情况下也能稳定工作。当驱动端的上拉电路导通，电路处于高电平时，回路上没有电流流过，这样的设计较少了功耗。



Pseudo Open Drain电路

**Chapter 2: Configuring the Device**

**2.1 Defining Alternate Compatible Parts(定义兼容器件)**

The Vivado tools select compatible AMD devices 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, select the alternate parts, and click **OK**.

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.

**2.1.1 Tcl Command Example for Defining Alternate Compatible Parts**

set\_property KEEP\_COMPATIBLE xc7k160tfbg676 [current\_design]

**2.2 Setting Device Configuration Modes（设置配置模式）**

Because AMD device configuration data is stored in CMOS latches, the device must be reconfigured each time it is powered up. Bank 0 contains the **dedicated configuration pins** and is always a part of every configuration interface. Bank 65 in AMD UltraScale and AMD UltraScale+ devices and Banks 14 and 15 in 7 series devices contain **multifunction pins** that are used in various configuration modes. For information on the available device configuration modes, see the Configuration User Guide for your device. For information on analyzing how the configuration modes might conflict with other multifunction pins, see Multifunction Pins.

To set device configuration modes and view information about the modes:

1. Select Tools → Edit Device Properties.

2. In the Edit Device Properties dialog box , select the Configuration Modes category, do the following, and click OK to close the dialog box:

• Enable the check box for a configuration mode to set that configuration mode. When you set a configuration mode:

○ The associated I/O pins display in the Config column of the Package Pins window.

○ The following constraints are created when you save the design:

set\_property BITSTREAM.CONFIG.PERSIST NO [current\_design]

set\_property CONFIG\_MODE <configuration\_mode> [current\_design]

• Click a configuration mode to open a dialog box in which you can view information.

• Enable Prohibit usage of the configuration pins as user I/O and persist after configuration to ensure that pins are used as configuration pins and not as general purpose I/Os after configuration. When you select this option, the following constraint is created when you save the design:

set\_property BITSTREAM.CONFIG.PERSIST YES [current\_design]

**IMPORTANT!** The JTAG configuration mode is always selected. You can select one configuration mode in addition to the JTAG configuration mode.

3. Select File → Constraints → Save to save the constraints to the target XDC file.

Note: The Tcl command equivalent is: save\_constraints.

**TIP:** When setting device configuration modes, you can undo your last action using Edit → Undo. Alternatively, you can enter undo in the Tcl console.

**2.2.1 Tcl Command Example for Setting Device Configuration Modes**

set\_property CONFIG\_MODE SPIx2 [current\_design]

By default, configuration pins are not set to persist after configuration. To ensure that pins are used as configuration pins and not as general purpose I/Os after configuration, enter the following Tcl command:

set\_property BITSTREAM.CONFIG.PERSIST YES [current\_design]

**2.3 Setting Device Constraints（设置器件约束）**

In the Device Constraints window (shown in the following figure), you can set constraints, including **DCI\_ CASCADE** and **INTERNAL\_VREF**.

AMD devices have configurable SelectIO interface drivers and receivers that support many 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).

AMD DCI can either control the output impedance of a driver or add a parallel termination to the driver, receiver, or both, with the goal of accurately matching the

characteristic impedance of a transmission line.

DCI actively adjusts the impedance inside an I/O bank to calibrate to external precision reference resistors placed on the VRN and VRP pins. This compensates for changes in I/O impedance due to process variation or variations of temperature and supply voltage. DCI uses two multi-purpose reference pins in each I/O bank to control the impedance of the driver or the parallel-termination value for all of the I/Os in the bank.

Single-ended I/O standards with a differential input buffer require a VREF. When a VREF is required within an I/O bank, use the following pins for the bank as VREF supply inputs:

• For UltraScale architecture-based devices, use the dedicated VREF pin

• For 7 series devices, use the two multifunction VREF pins

• Alternatively, you can generate an internal VREF using the INTERNAL\_VREF constraint.

Using internal reference voltages can remove the need to provide a particular VREF supply rail on the PCB and can free multipurpose VREF pins in a given I/O bank for other I/O port assignments.

Each I/O bank has one VREF plane, and each bank can have the optional INTERNAL\_VREF.

the 7 Series FPGAs SelectIO Resources User Guide (UG471)

the UltraScale Architecture SelectIO Resources User Guide (UG571)

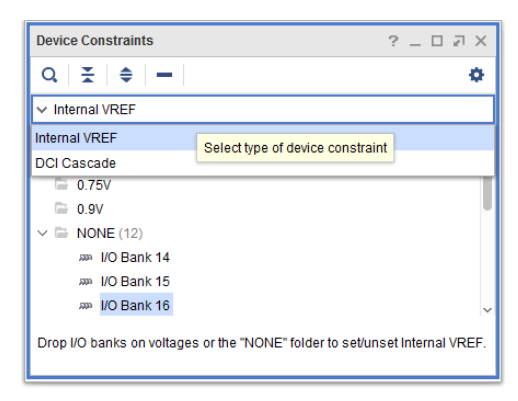
**2.3.1 Creating a DCI\_CASCADE Constraint**

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, and 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.

You can configure the DCI\_CASCADE constraint for 7 series devices, Zynq 7000, and UltraScale architecture-based devices. For more information on this constraint, see the Vivado Design Suite Properties Reference Guide (UG912).

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.



2. Select the I/O banks you want to cascade, right-click, and select Add DCI Cascade.

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.

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

**Tcl Command Example for Creating a DCI\_CASCADE Constraint**

set\_property DCI\_CASCADE {31 32} [get\_iobanks 36]

get\_iobanks specifies the master bank. In this example, 31 and 32 are the slave banks, and 36 is the master bank.

**2.3.2 Modifying or Removing DCI Cascade Constraints**

The Tcl command equivalent for remove action is:

set\_property DCI\_CASCADE {} [get\_iobanks 36]

**2.3.3 Creating an INTERNAL\_VREF Constraint**

Internal generation removes the need to provide for a particular VREF supply rail on the PCB and frees the multipurpose VREF pins in a given I/O bank to be used as normal I/O pins.

To create an INTERNAL\_VREF constraint, drag and drop the I/O bank onto the desired voltage folder (for example, 0.7V or 0.84V) in the Device Constraints window.

**Tcl Command Example for Creating an INTERNAL\_VREF Constraint**

set\_property INTERNAL\_VREF 0.7 [get\_iobanks 65]

set\_property INTERNAL\_VREF 0.84 [get\_iobanks 67]

**2.4 Setting the Configuration Bank Voltage Select Pin**

The configuration bank voltage select (CFGBVS) logic input pin（配置组电压选择逻辑输入引脚） is referenced between VCCO\_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.

You can set the configuration voltage, or VCCO\_0 voltage, to 1.5, 1.8, 2.5, or 3.3. Based on these settings, DRCs are run on banks 0, 14, and 15 for 7 series devices. For UltraScale devices, DRCs are run on banks 0 and 65. These values 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. If the CFGBVS property has a value, the Vivado tools check for a CONFIG\_MODE property. DRCs are issued based on IOSTANDARD and CONFIG\_VOLTAGE settings for the bank.

• 7 Series FPGAs Configuration User Guide (UG470)

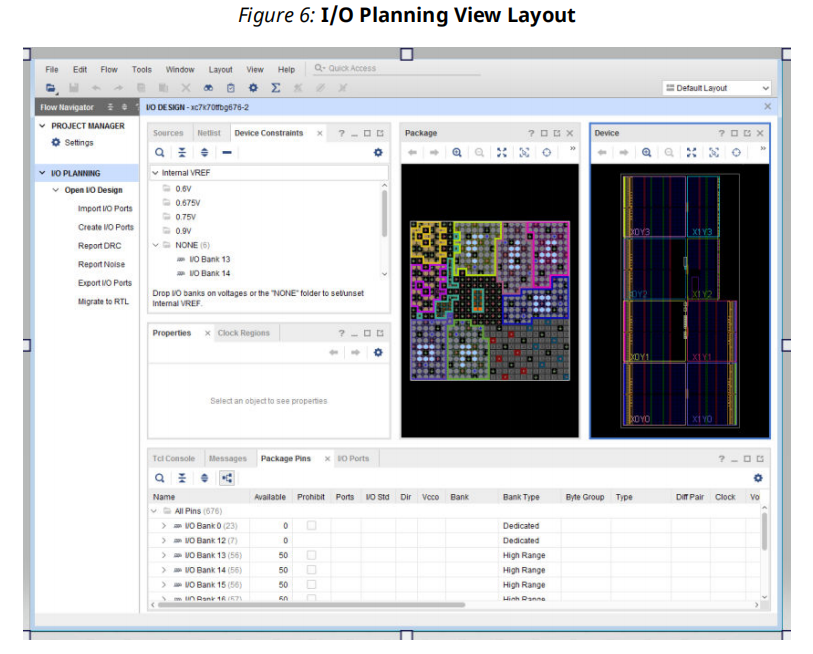
• UltraScale Architecture Configuration User Guide (UG570)

• Zynq UltraScale+ Device Technical Reference Manual (UG1085)

**Chapter 3: I/O Logic and Low-Speed I/O Planning**

You can use the I/O Planning view layout in the AMD Vivado™ IDE 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.

**3.1 Using the I/O Planning View Layout**



**3.2 Viewing Device Resources**

TIP: To search for specific objects or device sites, use the **Edit → Find** command.

**3.2.1 Properties**

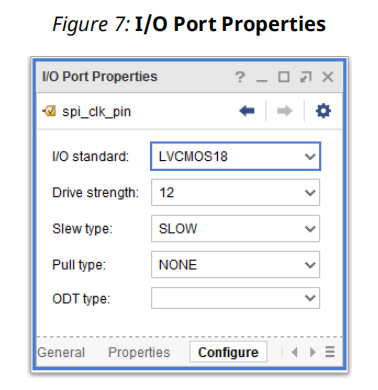
To show the Properties window, select **Window → Properties**.

TIP: You can get property information on package pins using Tcl commands. For example, the following command shows all the properties associated with the specified package pin:

report\_property [get\_package\_pins <pin\_number>]

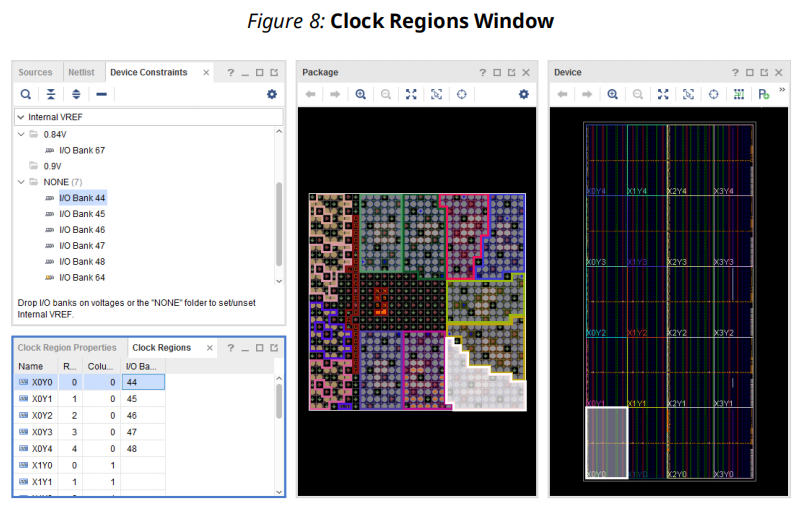
The following command shows the maximum trace delay for the specified package pin:

get\_property MAX\_DELAY [get\_package\_pins <pin\_number>]



**3.2.2 Clock Region Resources and Statistics**

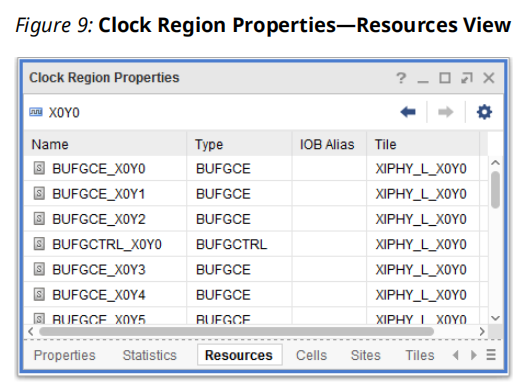
When you select a clock region in the Clock Regions window, the related I/O banks and regional clock resources are highlighted in the Package and Device windows, as shown in the following figure.



You can click the Clock Region 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 the following figure.

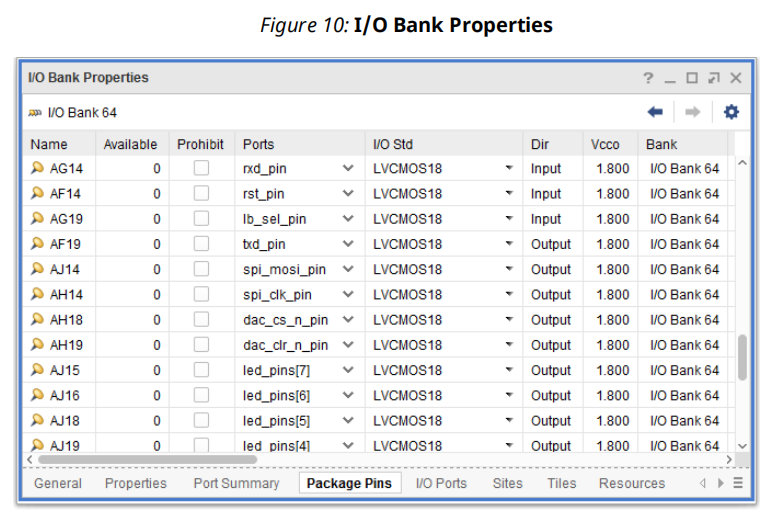


**3.2.3 I/O Bank Resources**

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(Ctrl+E, or Windows -> Properties), click the views at the bottom of the window to see the different types of information available.



**3.2.4 Multifunction Pins**

The **Package Pins** window includes the following information:

• **Type** column identifies multifunction pin types.

• Config column shows the pin definition of the multifunction pin after you set the device configuration mode.

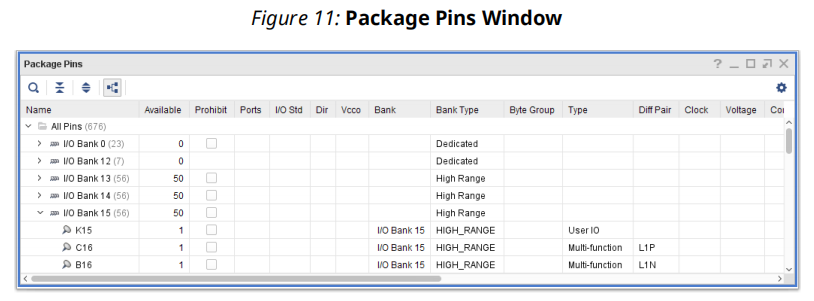
• Information identifies conflicting multifunction pins for designs that contain GTs, memory controllers, or PCI logic.

• Asterisks indicate the I/O standard or attribute, such as drive strength or slew type, is not set to the default generated by the system.

• Clock capable pins are displayed as a hexagon icon

• VREF pins are displayed as a small power icon

**IMPORTANT!** Dedicated I/O pins are dedicated to the targeted device not to the bank. For example, dedicated I/O pins such as VCCO and GND are device-specific rather than bank-specific.



**3.3 Defining and Configuring I/O Ports**

**3.3.1 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

• RTL Project: Use RTL files or headers to create an RTL project for I/O planning, then add more complete RTL source files to the project later as the design progresses.

• 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.

**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.

Related Information

Using I/O Port Lists in CSV File Format

Differential Pairs in the CSV File

Exporting I/O Pin and Package Data

**Importing an XDC File**

Because the XDC format does not define port direction, the direction is undefined. To define the I/O port direction, right-click in the I/O Ports window, and select Set Direction. 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.

**3.3.2 Creating Single-Ended or Differential I/O Ports**

You can manually define new ports in an **I/O planning project**.

**Note:** The I/O Ports window groups each differential pair into a single row. To get a list of signals that matches the total number of ports in the I/O Ports window, enter the following Tcl command:

get\_ports \* -filter {BUS\_WIDTH == "" }

To create I/O ports:

1. In the I/O Ports window, right-click, and select Create I/O Ports.

2. In the Create I/O Ports dialog box, edit the following options, and click OK.

Note: To create a differential I/O port, enable this option. This creates two ports and adds an \_N suffix to the name of the negative port.

Note: Each I/O standard indicates its voltage number in which the I/O operates. The Tcl command -

get\_property VCCO\_IN [get\_io\_standards "I/O Standard"]

Note: The allowable values are PULLUP, PULLDOWN, and KEEPER. The default is NONE. The property guarantees a logic High (PULLUP) or Low (PULLDOWN) level to allow tri-stated nets to avoid floating when not being driven. The value KEEPER retains the value of the output net to which the port is attached.

Note: The DELAY\_VALUE\_XPHYproperty on PORT objects specifies the amount of delay to add on an input or output path for AMD Versal™ XPHY Logic Interfaces. During opt\_design when the Advanced I/O Wizard IP is being regenerated, the DELAY\_VALUE\_XPHY value will be copied from the PORT onto the XPHY instance for the input or output path.

Note: Not all I/O standards are compatible with each location. For HD I/O banks, the operating voltage range is 1.8V-3.3V. For XP I/O banks, the operating voltage range is 1V-1.5V.

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

• Creating a single-ended I/O port:

create\_port port\_1 -direction in

• Creating a differential I/O port:

create\_port port\_2 -direction in -diff\_pair

**3.3.3 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, right-click, and select **Make Diff Pair**.

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.

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

TIP: Right-click, and select Split Diff Pair to separate a diff pair into two ports.

IMPORTANT! When you apply certain constraints to one side of a differential pin pair in an AMD UltraScale™ architecture, the opposite constraint is automatically applied to the other side. For example, if you apply a PULLDOWN constraint to the P side, a PULLUP constraint is applied to the N side. If you make changes to the constraints, the most recent settings overwrite previous settings.

**Tcl Command Examples for Making and Splitting Differential Pairs**

• Making a differential pair:

make\_diff\_pair\_ports txp\_out txn\_out

• Splitting a differential pair:

split\_diff\_pair\_ports txp\_out txn\_out

**3.3.4 Configuring I/O Ports**

AMD devices support configurable SelectIO™ interface drivers and receivers, which support various standard interfaces. These standard interfaces include programmable control of output strength and slew rate, on-chip termination using DCI, and generation of internal VREF. You can configure one or more I/O ports to define I/O standard, drive strength, slew type, pull type, and in term.

• 7 Series FPGAs SelectIO Resources User Guide (UG471)

• UltraScale Architecture SelectIO Resources User Guide (UG571)

For information on packaging and pinout specifications, see the following, depending on your device:

• 7 Series FPGAs Packaging and Pinout Product Specification (UG475)

• UltraScale and UltraScale+ FPGAs Packaging and Pinouts Product Specification (UG575)

• Zynq 7000 SoC Packaging and Pinout Product Specifications (UG865)

For detailed information on AMD Zynq™ 7000 pins, including MIO pins, see the Zynq 7000 SoC Technical Reference Manual (UG585).

For detailed information on Zynq UltraScale+ MPSoC pins, including MIO pins, see Zynq UltraScale+ Device Technical Reference Manual (UG1085).

To configure a port or a group of ports:

1. In the I/O Ports window, select the ports.

2. Right-click, and select **Configure I/O Ports**.

3. In the Configure Ports dialog box (see the following figure), edit the following options, and click **OK**.

• In Term type: (7 series devices only) Define the parallel termination properties of the input signal. For more information, see the 7 Series FPGAs SelectIO Resources User Guide (UG471).

• ODT: (UltraScale architecture-based devices only) Define the value of the on-die

termination (ODT) at the input for both DCI and non-DCI versions of the standards

supported. For more information, see the UltraScale Architecture SelectIO Resources User Guide (UG571).

• Fixed: Indicates that the logical ports are user assigned. Ports must be fixed to ensure that the bitstream generates without errors.

In the Configure Ports dialog box, the Fixed option is read only. To fix ports, select the ports in the I/O Ports window, right-click, and select Fix Ports, or enter the following Tcl command in the Tcl Console:

set\_property IS\_LOC\_FIXED true [get\_selected\_objects]

Alternatively, you can enter the following Tcl command to fix ports:

set\_property IS\_LOC\_FIXED true [get\_ports <list\_of\_ports>]

CAUTION! For 7 series devices, Zynq 7000 SoC, AMD Zynq™ UltraScale devices, UltraScale+ devices, and Zynq UltraScale+ MPSoCs, 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. You must apply extra care when assigning I/O standards, because these devices have Low and High voltage I/O banks.

**3.3.5 Setting I/O Port Direction**

• For I/O planning projects only:

○ In the Direction column of the I/O Ports window, click a port and change the direction using the drop-down menu.

○ Click a port in the I/O Ports window, and change the direction in the I/O Port Properties window.

○ Select the I/O ports, buses, or interfaces to be configured in the I/O Ports window, rightclick, and select Set Direction.

• For RTL projects only, define the port direction in the RTL source.

IMPORTANT! You can only set the port direction property in an I/O planning project. If you attempt to modify this property outside of an I/O planning project, a critical warning is issued.

**3.3.6 Creating I/O Port Interfaces**

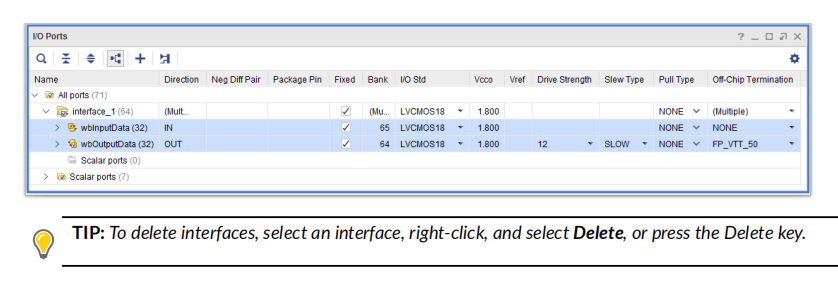
To group multiple ports or buses together, you can create an interface. This aids in pin assignment by treating all of the interface ports as one group(将接口中的所有端口视为一组). This aids in pin assignment by treating all of the interface ports as one group. Assigning all of the pins simultaneously helps condense 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. Right-click, and select Create I/O Port Interface.

3. In the Create I/O Port Interface dialog box (see the following figure), enter a name for the interface, adjust assignment selection, and click OK.



**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, in the I/O Ports window right-click a port, and select Unassign from Interface.

**Tcl Command Examples for Working with I/O Port Interfaces**

• Creating I/O port interfaces:

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]]

• Removing I/O ports from an interface:

set\_property INTERFACE "" [get\_ports [list port\_2 port\_2\_N]]

**3.3.7 Automatically Inferring I/O Port Interfaces**

RECOMMENDED: If your project targets a platform board rather than a part, AMD recommends that you use the Vivado Design Suite platform board flow to configure and apply board pinout constraints using the Board tab in the Customize IP dialog box or in the Board window in the Vivado IP integrator. For more information on the platform board flow, see the Vivado Design Suite User Guide: System-Level Design Entry (UG895).

You can view the interfaces that are connected from an IP to the top-level ports of your design. For these IP interfaces, the Vivado tools automatically infer a pin planning interface that groups the related top-level I/O ports. This provides a symbolic way of referring to the interface in the context of the top-level design.

Tcl Command Examples for Working with the Platform Board Flow

• Getting board part interfaces:

get\_board\_part\_interfaces

• Getting ports associated with the board interface:

get\_board\_interface\_ports -of [get\_board\_part\_interfaces]

• Getting properties of the port:

get\_property NAME [lindex [get\_board\_interface\_ports] 5]

get\_property DIRECTION [lindex [get\_board\_interface\_ports] 5]

• Getting properties from the board part pin:

get\_property IOSTANDARD [get\_board\_part\_pins]

get\_property LOC [get\_board\_part\_pins]

Note: For information on the

get\_board\_part\_interfaces

Tcl command and other Tcl commands, see the Vivado Design Suite Tcl Command Reference Guide (UG835).

**3.3.8 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. Right-click, and select **Set Prohibit**.

**Tcl Command Example for Prohibiting I/O Pins**

set\_property PROHIBIT 1 [get\_sites U17]

set\_property prohibit 0 [get\_sites U17]

**3.4 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.

**3.4.1 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, right-click, and select **Place I/O Ports Sequentially**.

• In either the Package window or the Device window, right-click, 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 subsequent 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.

**3.4.2 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, right-click, and select **Place I/O Ports in an I/O Bank**.

• In either the Package window or the Device window, right-click, 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 until all of the I/O ports are placed, or you press **Esc**.

Selecting a pin at one end of an I/O bank results in a continuous bus assignment across the I/O bank. 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.

**Tcl Command Examples for Placing Ports into I/O Banks**

• Placing ports into I/O banks:

place\_ports -iobank [get\_iobanks {12 13 14 15}] [all\_inputs]

• Placing ports into all I/O banks:

place\_ports -iobank [lrange [get\_iobanks] 1 end] <port list>

Note: The place\_ports command is not supported for bank 0.

**3.4.3 Placing I/O Ports in a Defined Area**

The cursor turns into a cross symbol, which indicates that you can define a rectangle for port placement.

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**.

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.

**3.4.4 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. Right-click, and select Swap Locations.

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.

**3.4.5 Moving Previously Placed I/O Ports**

To move a port or group 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.

**3.4.6 Automatically Placing I/O Ports**