



# Xilinx Design Constraints

# Objectives

- ▶ After completing this module, you will be able to:
  - Assign pin locations using the I/O Planner
  - Describe static timing paths
  - Create real and virtual clocks
  - Create appropriate input and output delays
  - Use virtual clocks for input and output delays
  - Use the Constraints Wizard

# Outline

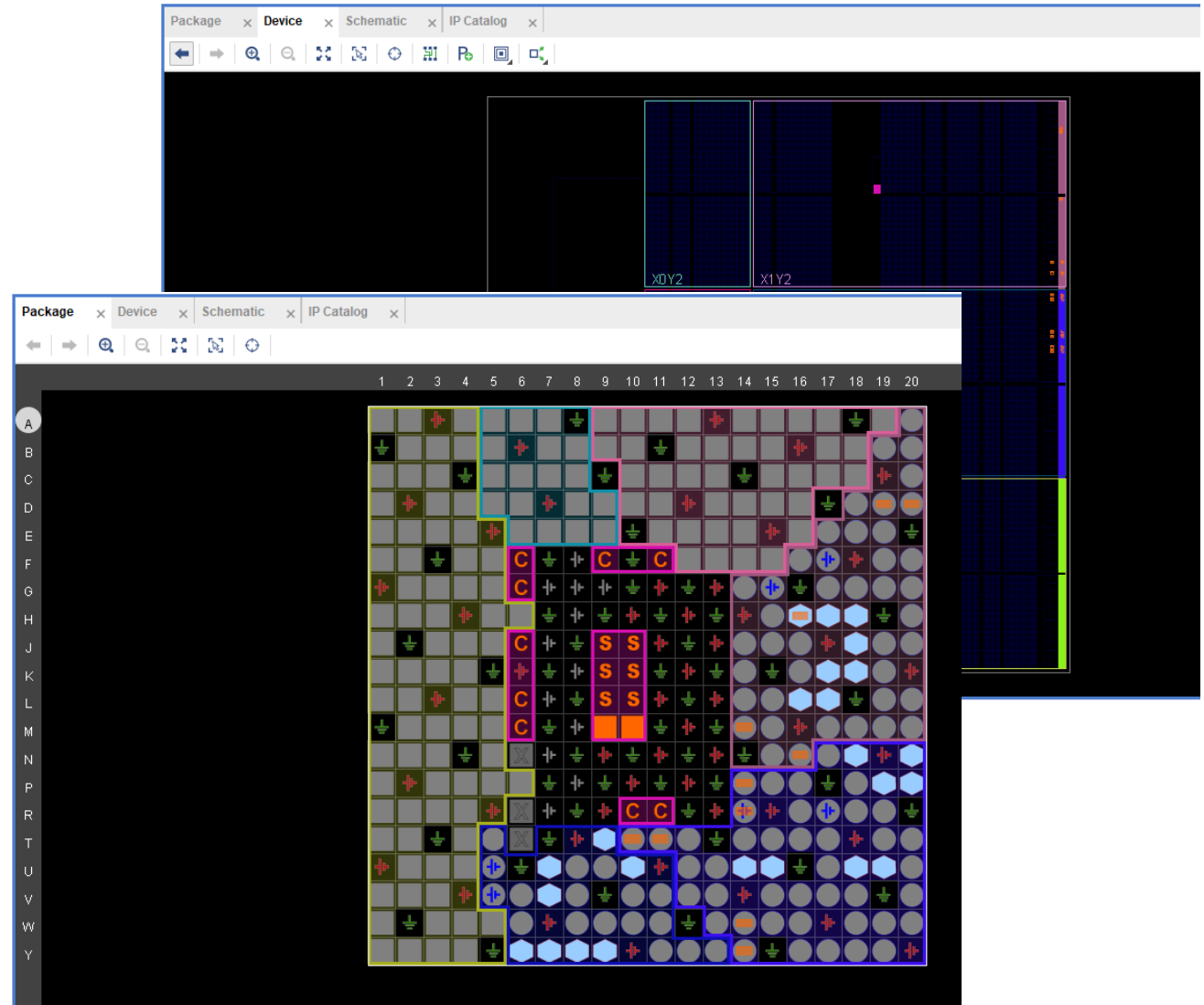
- ▶ *Pin Constraints*
- ▶ Timing Constraints
  - Period
  - Input Delay
  - Output Delay
  - Virtual Clocks
- ▶ Constraints Wizard
- ▶ Summary

# Pin and Clock Planning

- ▶ Pin and clock planning often happens early in the project
  - Decisions here can have significant effects throughout the design
    - Excessive clock skew
    - Poor I/O timing
    - Timing-hazardous clock domain crossing
    - Less flexible design placement
    - Fewer clocking resource choices
    - Poor logic placement
    - Excessive routing delays
    - Reduced device utilization
- ▶ Pin and clock planning should be considered together
  - Choices made for clock pins affect clocking timing and resources choices
  - Choices made for data pins affect clock pin placement decisions

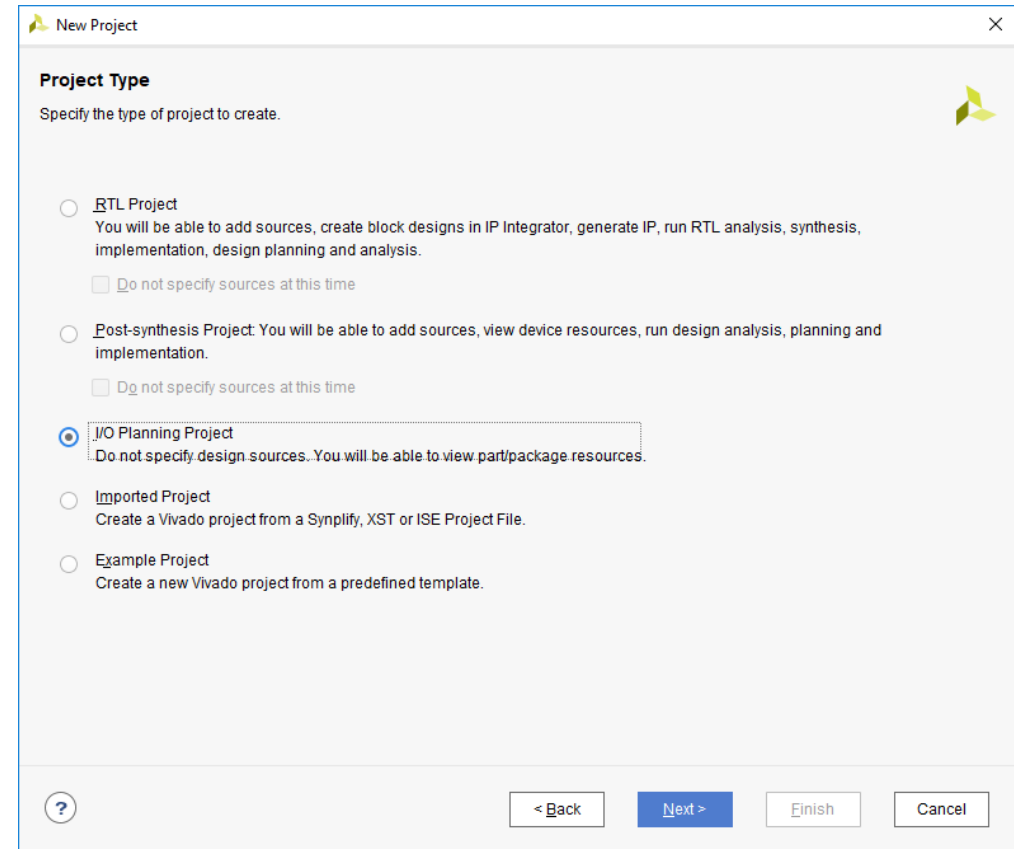
# Design View Layouts

- ▶ Several default view layouts available
  - Toolbar pull-down (I/O Planning, Clock Planning, Floorplanning, etc)
- ▶ Designs load with "Design Analysis" view by default
- ▶ Use "I/O Planning" for easy access to I/O planning functions
  - Adds Package view, clock resources, I/O ports, package pins



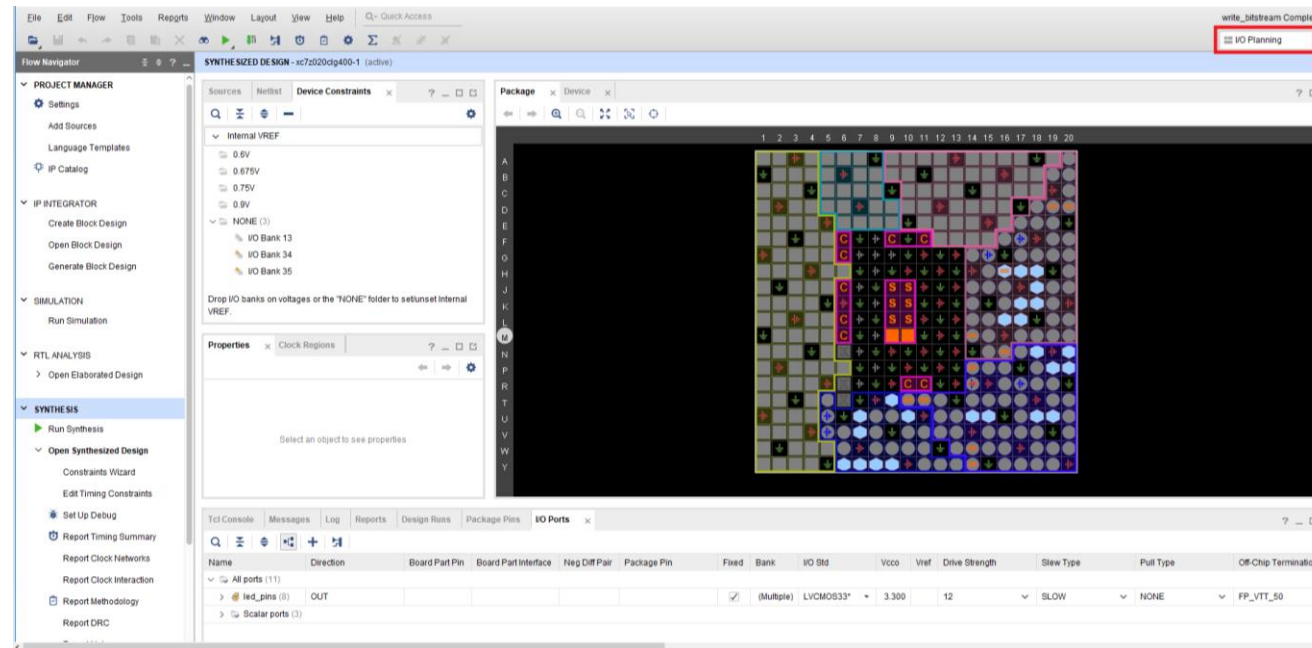
# I/O Planning Project

- ▶ When creating a new project from the Getting Started page you have the option to create an I/O planning project *without* RTL
  - This allows testing of pin assignments
    - I/O banking rules
    - Avoid ground bounce
    - The I/O Planner performs error checking for your pinout
  - However, it is recommended that you have RTL associated
    - Error checking is better



# Launching I/O Planner with RTL

- ▶ This flow is used if you already have a RTL project created
  - Synthesize the design
    - Open synthesized design by clicking Open Synthesized Design
    - Open the I/O Planner by selecting the I/O Planning view from the drop-down box on the horizontal toolbar
      - This allows you to view and/or enter the I/O locations and properties



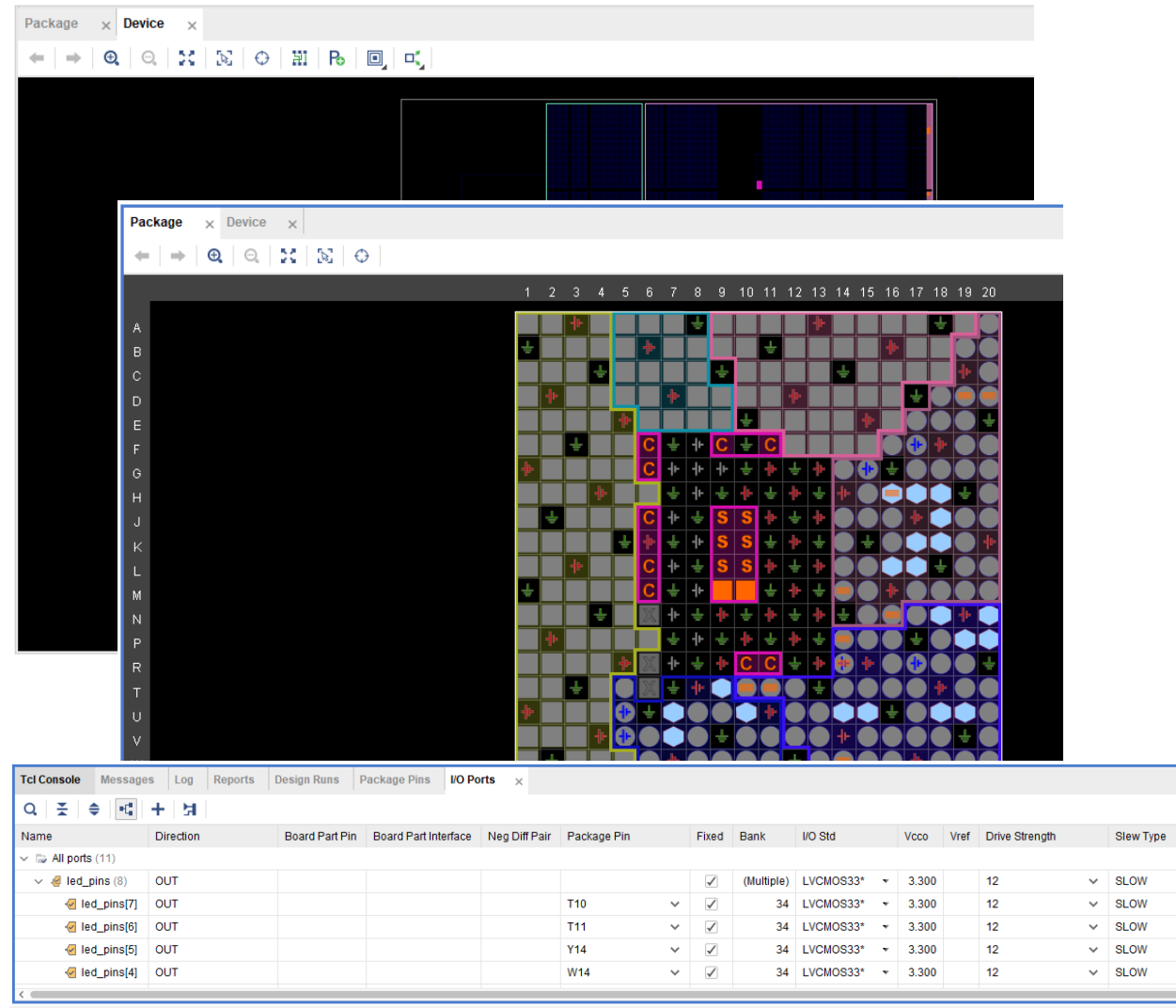
# I/O Planner

- ▶ I/O Planner performs error checking on the design pin layout
  - This requires rules-based I/O assignments
    - DRC provides guidance for pin assignments connecting to dedicated FPGA logic (microprocessor, MGT, or differential pairs, for example)
    - Noise analysis (to avoid ground bounce)
    - Verify I/O banking rules
  - Semi or fully automatic pin assignment capabilities
    - Xilinx recommends that you place timing-critical ports before allowing automatic pin assignment of the remaining pins
  - Supports grouping-related pins to simplify I/O interface management



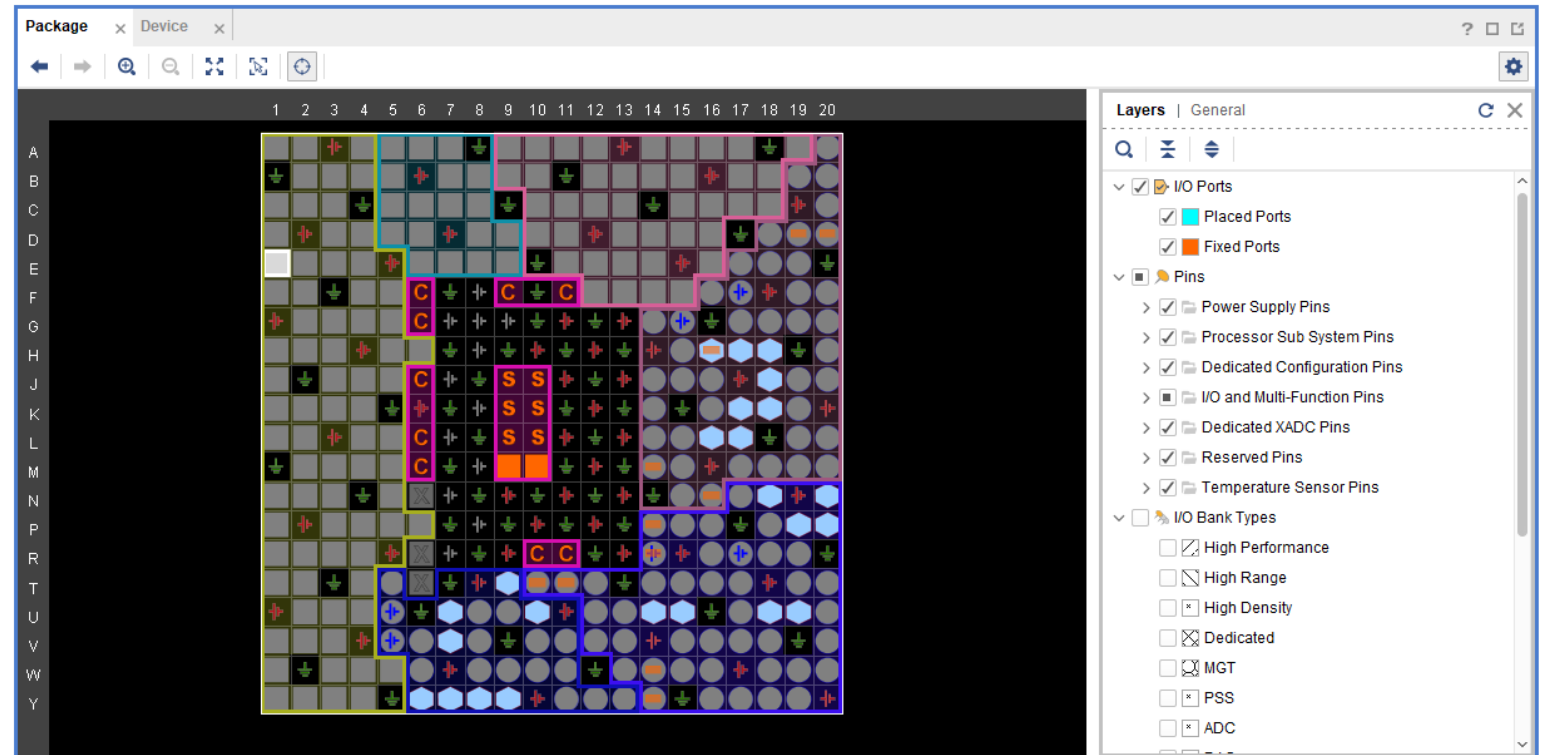
# I/O Planner Layout View

- ▶ I/O Planner allows you to view both the Die and Package views so that you can understand the I/O bank relationship with your logic
- ▶ Package and Device views
  - Graphically displays package pins, die pads, and I/O banks
  - I/O ports can be assigned within either of these two views
- ▶ Package Pins view
  - Displays I/O package specifications and assignment status
    - Allows you to see the Trace delays, pin type, voltage standard, and differential pair partners
  - Can display pins as a group, in an I/O bank, or as a list
  - As you drag across the display, icons show pin and I/O bank placement status



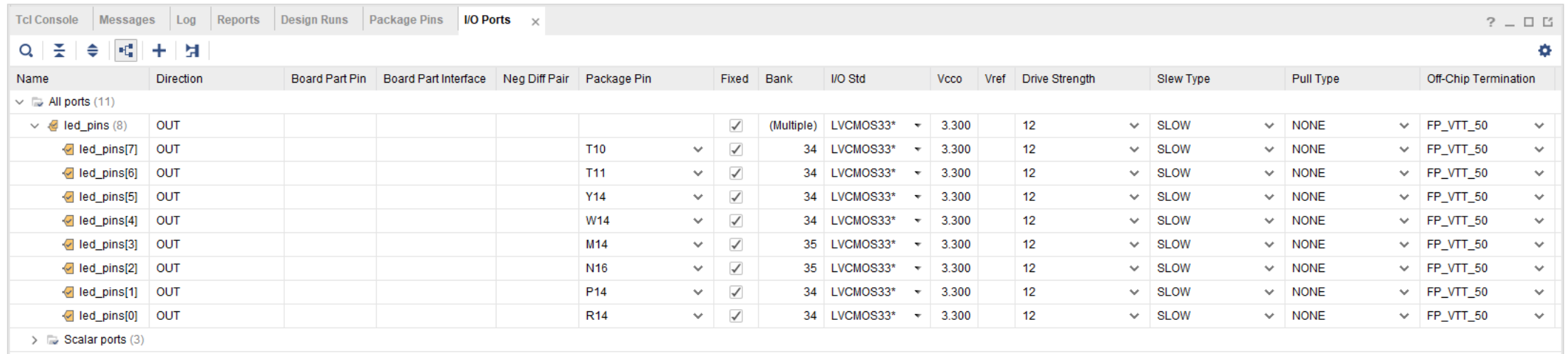
# Package View

- ▶ The colored areas between the pins display the I/O banks
- ▶ Show differential pairs
- ▶ Clock-capable pins (⬡), VCC (⬢), GND (⬣), no connection (⬤), XADC (⬥), Temperature Sensor (⬦)



# I/O Ports View

- ▶ Displays all I/O ports defined in the project
- ▶ Groups buses into expandable folders
- ▶ Can be displayed as groups of buses and interfaces or as a list
- ▶ Most I/O port assignment is initiated from this view
- ▶ Icons indicate I/O port direction and status



Name	Direction	Board Part Pin	Board Part Interface	Neg Diff Pair	Package Pin	Fixed	Bank	I/O Std	Vcco	Vref	Drive Strength	Slew Type	Pull Type	Off-Chip Termination
▼ All ports (11)														
▼ led_pins (8)	OUT					✓	(Multiple)	LVC MOS33*	3.300		12	SLOW	NONE	FP_VTT_50
led_pins[7]	OUT				T10	✓	34	LVC MOS33*	3.300		12	SLOW	NONE	FP_VTT_50
led_pins[6]	OUT				T11	✓	34	LVC MOS33*	3.300		12	SLOW	NONE	FP_VTT_50
led_pins[5]	OUT				Y14	✓	34	LVC MOS33*	3.300		12	SLOW	NONE	FP_VTT_50
led_pins[4]	OUT				W14	✓	34	LVC MOS33*	3.300		12	SLOW	NONE	FP_VTT_50
led_pins[3]	OUT				M14	✓	35	LVC MOS33*	3.300		12	SLOW	NONE	FP_VTT_50
led_pins[2]	OUT				N16	✓	35	LVC MOS33*	3.300		12	SLOW	NONE	FP_VTT_50
led_pins[1]	OUT				P14	✓	34	LVC MOS33*	3.300		12	SLOW	NONE	FP_VTT_50
led_pins[0]	OUT				R14	✓	34	LVC MOS33*	3.300		12	SLOW	NONE	FP_VTT_50
> Scalar ports (3)														

# set\_property command

## ► Use set\_property command

- `set_property PACKAGE_PIN T22 [get_ports led_pins[0]]`
- `set_property IOSTANDARD LVCMOS33 [get_ports led_pins[0]]`

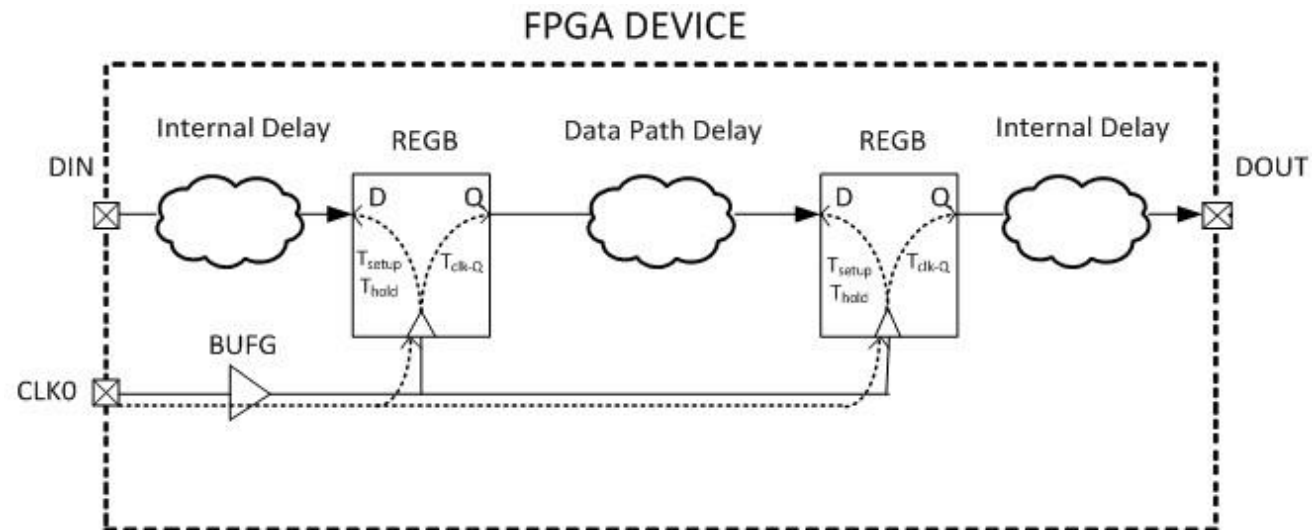
Or

- `set_property -dict {PACKAGE_PIN H17 IOSTANDARD LVCMOS33} [get_ports { led_pins[0] }]`

# Timing Constraints

# Static Timing Paths and I/O

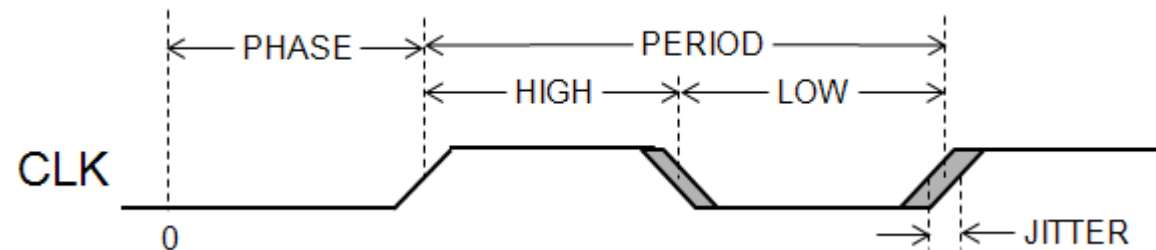
- ▶ Static timing paths start at clocked elements and end at clocked elements
  - Paths from internal flip-flop to internal flip-flop are constrained by clock
- ▶ Inputs and outputs of the FPGA are not start-points/end-points of static timing paths
  - By default, any logic between a primary I/O and an internal clocked element are not part of a complete static timing path
  - Without additional commands, no setup/hold checks are done on logic associated with I/O



# Timing Constraints - Period

# Clocks

- ▶ Clocks are periodic signals
- ▶ Clocks have certain attributes
  - Period
    - Nominal time from rising edge of the clock to the next rising edge of the clock
  - Duty cycle
    - Ratio of the high time to the low time of the clock
  - Jitter
    - Variation of the period from its nominal value
  - Phase
    - Position of the rising edge





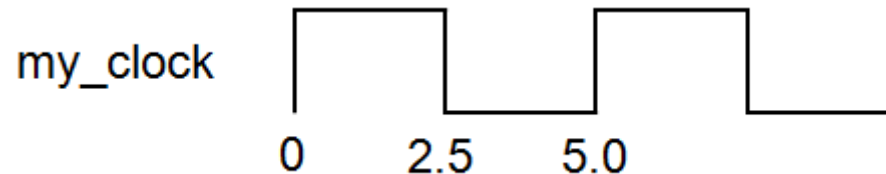
# Clocks as Objects

- ▶ In XDC clocks are primary objects
- ▶ Clocks have properties
  - *NAME* is the name of the clock
    - Will be user assigned or auto generated depending on the clock
  - *PERIOD* is the period of the clock
  - *WAVEFORM* describes the position of the edges of the clock
  - *IS\_GENERATED*, *IS\_VIRTUAL* are flags that describe how the clock was created
  - *SOURCE\_PINS* are the pins/ports/nets which the clock is attached to

# Creating Clocks

- ▶ Clocks are created with the `create_clock` Tcl command
  - `create_clock -name <name> -period <period> <objects>`
  - `<period>` is the period of the clock
  - `<name>` is the user assigned name for the clock
  - `<objects>` are the list of pins, ports, or nets to which to attach the clock
  - If `<objects>` is not present (or is a null list), the clock will not be attached to any objects, and will be a virtual clock

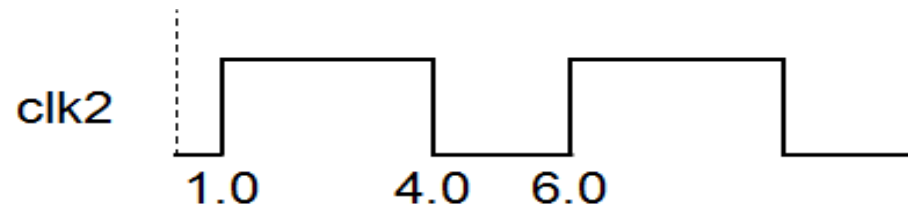
```
create_clock -name my_clock -period 5.0
```



# Clock Waveform

- ▶ Clocks can be created with edges at different positions
  - Allows for the description of clocks with phase offsets and clocks with different duty cycles
  - Uses the `-waveform <edges>` option
    - `<edges>` is a list of numbers representing the times of successive edges
      - The first number is the time of the first rising edge
    - Default is 0.00 for the rising edge and PERIOD/2 for the falling edge

```
create_clock -name clk2 -period 5.0 -waveform {1.0 4.0}
```



# Setting Jitter

- ▶ The Vivado Design Suite timing engine allows for two sources of jitter
  - System Jitter: Jitter introduced by the clocking network inside the FPGA
    - A single value for all clocks in the system
    - Set with the `set_system_jitter` command
    - `set_system_jitter <value>`
      - `<value>` is the jitter in time units (nanoseconds)
  - Input Jitter: Jitter that exists on the input clock
    - Set independently for each clock source
    - Set with the `set_input_jitter` command
    - `set_input_jitter <clock_name> <value>`
      - `<clock_name>` is the name of a clock (not the clock object)
      - `<value>` is the jitter in time units
- ▶ Both sources of jitter will be combined appropriately in STA calculations

# Clock Latency

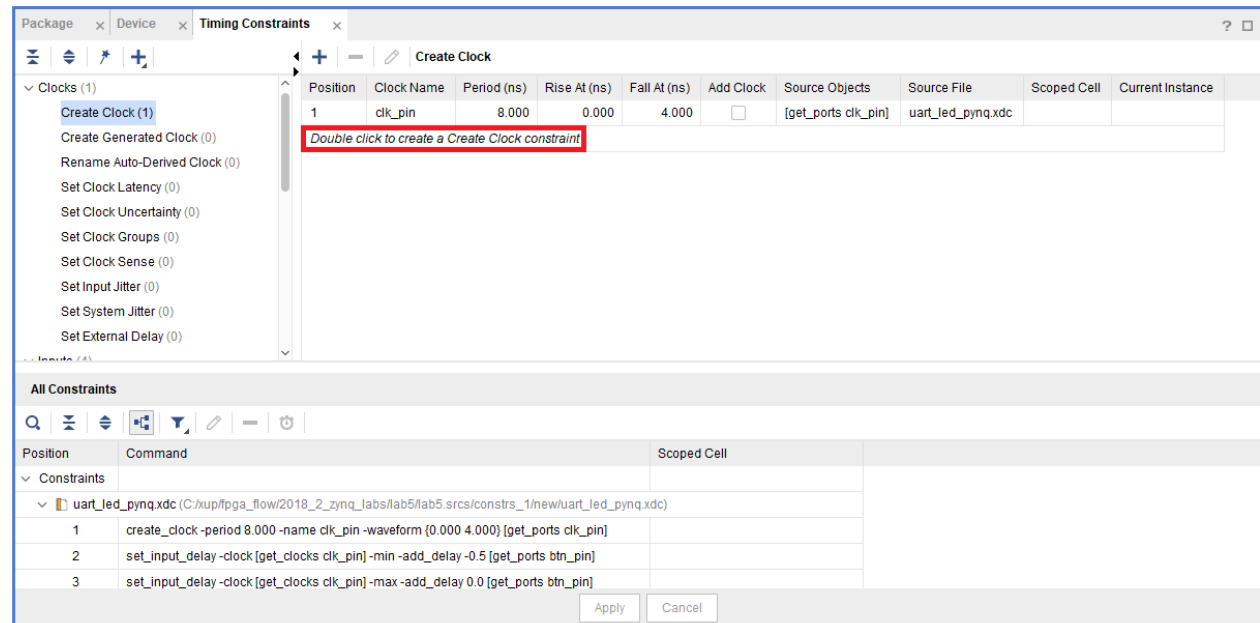
- ▶ The latency of the clock can be controlled with the `set_clock_latency` Tcl command
  - `set_clock_latency -source <latency> <objects>`
  - `<latency>` is the latency to apply
  - `<objects>` is the list of clocks, ports or pins to which to apply the latency
- ▶ The latency is an additional clock delay that is added between the clock object and the pin, port, or net to which the clock is attached
  - If the `set_clock_latency` specifies a clock object, the latency is added to all destinations of the clock
  - If the `set_clock_latency` specifies a port or pin, it applies to all clocks that go through that port or pin
    - If the port or pin has more than one clock associated with it, the `-clock <clocks>` option can be used to specify which clocks to apply the latency to

# Options of the `set_clock_latency` Command

- ▶ The `set_clock_latency` command has several options
  - `-rise`: The specified latency applies only to the rising edge of the clock
  - `-fall`: The specified latency applies only to the falling edge of the clock
  - `-min`: Specifies the latency to apply when the shortest path is used
  - `-max`: Specifies the latency to apply when the longest path is used
- ▶ If the `-min/-max` are not specified, the latency applies to both min and max
- ▶ If the `-rise/-fall` are not specified the latency applies to both rise and fall

# Creating Clocks using the GUI

- ▶ The Timing Constraint window can be opened using the menu Window > Timing Constraints
  - A clock can be created by double clicking on the Create Clock, or a new row in the Create Clock table
- ▶ Alternatively, can be set via the Constraints Wizard
  - Covered later in this presentation



# The Create Clock Wizard

Name of clock to be created

The 'Create Clock' dialog box contains the following fields and controls:

- Header:** 'Create Clock' with a close button (X).
- Description:** 'Creates a clock object. The created clock is applied to the specified source objects. If you do not specify source objects, but give a clock name, a virtual clock is created.'
- Fields:**
  - Clock name:** A text input field.
  - Source objects:** A text input field with a browse button (three dots).
- Waveform Section:**
  - Period:** A spinner set to 10 ns.
  - Rise at:** A spinner set to 0 ns.
  - Fall at:** A spinner set to 5 ns.
- Checkbox:** 'Add this clock to the existing clock (no overwriting)'.
- Command:** A text area containing the Tcl command: `create_clock -period 10.000 -waveform {0.000 5.000}`.
- Buttons:** '?', 'Reference', 'Reset to Defaults', 'OK', and 'Cancel'.

Red arrows in the image point to the 'Clock name' field, the 'Source objects' field, the 'Period', 'Rise at', and 'Fall at' spinners, and the 'Command' text area.

Objects to which  
attach clock

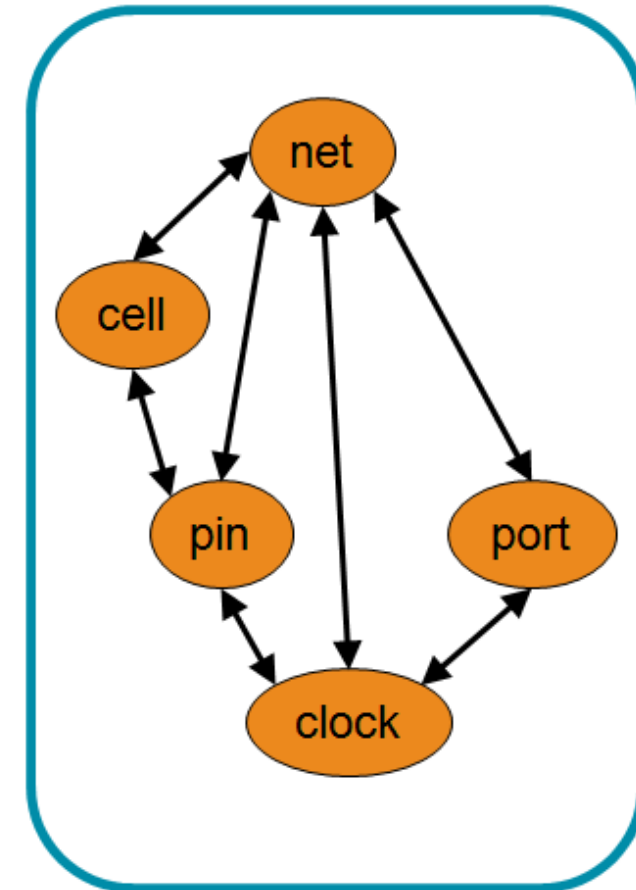
Period and edges of  
the clock

Tcl command to  
create clock



# Handling Clock Objects

- ▶ Clocks can be attached to pins, ports, and nets
  - Specify the list of pins, ports, or nets which are to be attached to the clock when the clock is created
- ▶ Once created, clock objects can be gotten using the `get_clocks` command
  - `get_clocks <name>`
    - Returns list of clocks that match `<name>`
    - `<name>` may include wildcards
  - Has similar options as the other `get_*` commands
    - `-regexp`
    - `-filter <expression>`
    - `-of_objects <object>`



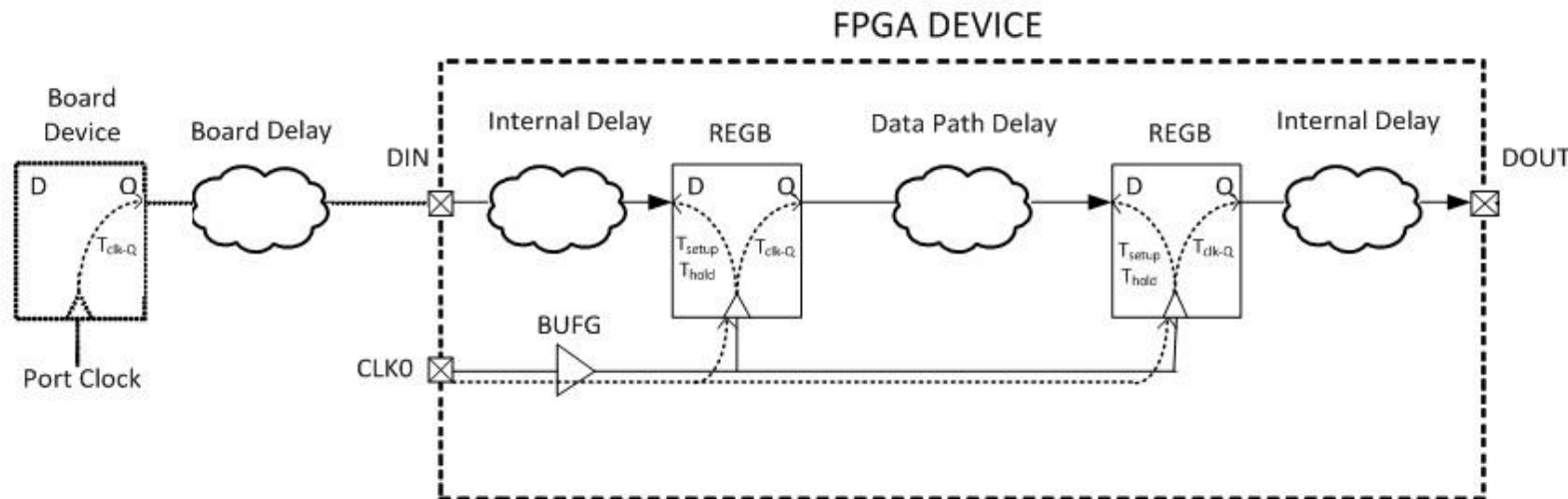
# Timing Constraints – Input Delay

# Synchronous Input Interfaces

- ▶ Most interfaces to an FPGA use synchronous communication
  - FPGA and the device driving the FPGA have some shared timing reference
    - This is usually a common clock or a related clock
- ▶ Complete static timing path through an input
  - Starts at a clocked element in the driving device
    - Referenced to a clock provided to the driving device
  - Ends at a clocked element in the FPGA
    - Referenced to the clock that propagates to the destination clocked element in the FPGA
  - Propagates through the elements between them
    - $CLK > Q$  of the external device
    - Board propagation time
    - Port of the FPGA
    - Combinatorial elements in the FPGA before the destination clocked element

# Completing the Static Timing Input Path

- ▶ To complete the static timing path, you need to describe the external elements to the Vivado static timing engine
  - What clock is used by the external device
  - Delay between the external device's clock and the arrival at the input port of the FPGA
    - Includes the CLK > Q time of the external device and the board delay



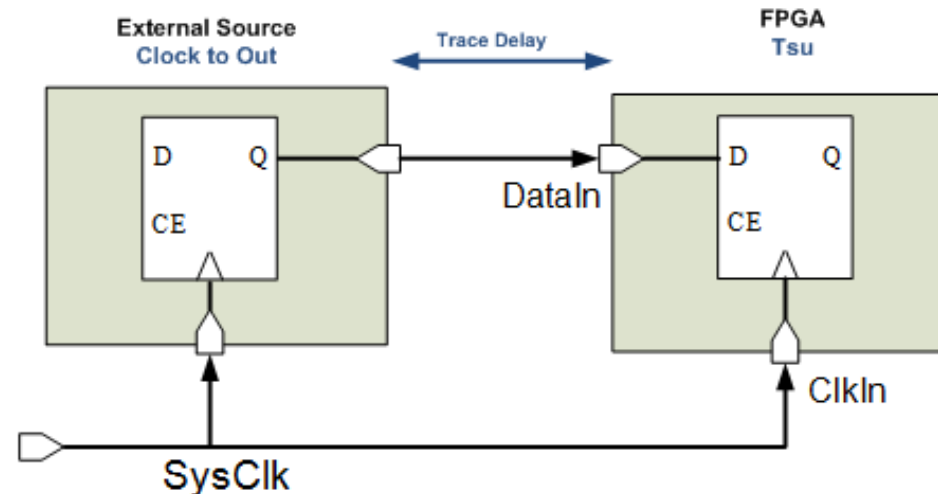
# set\_input\_delay Command

- ▶ `set_input_delay` command supplies the information required to complete the static timing path
  - `set_input_delay -clock <clock_name> <delay> <objects>`
    - `<clock_name>` is the name of the clock used by the external device
      - Can be a real or virtual clock
      - Can be the **name** of a clock; does not need to be a clock object
        - Can use a clock object if desired
    - `<objects>` is the list of objects to which to attach the `set_input_delay`
      - Usually a set of input and/or inout ports
      - Usually uses the `get_ports` command or the `all_inputs` command
    - `<delay>` is the delay from `<clock_name>` to the attached `<objects>`
      - Includes the external device and board delay

# Using a Common Clock

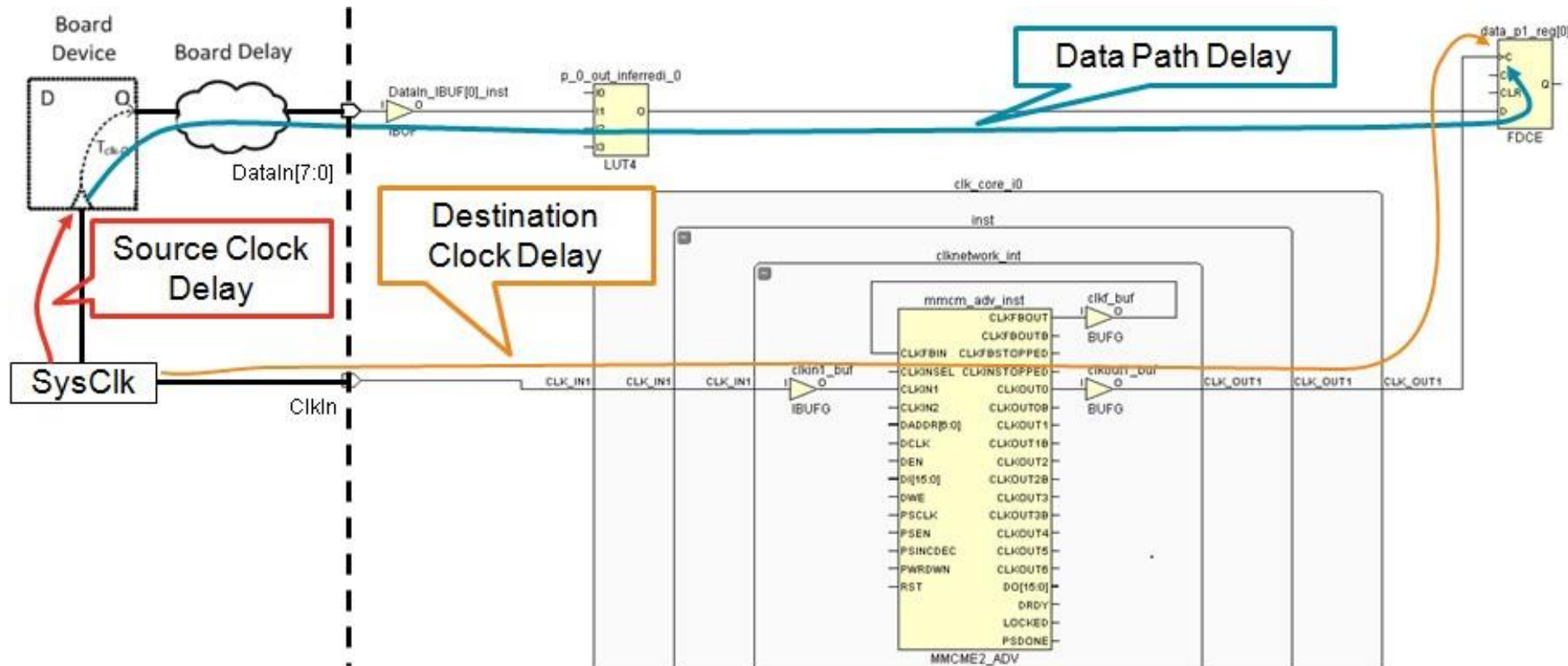
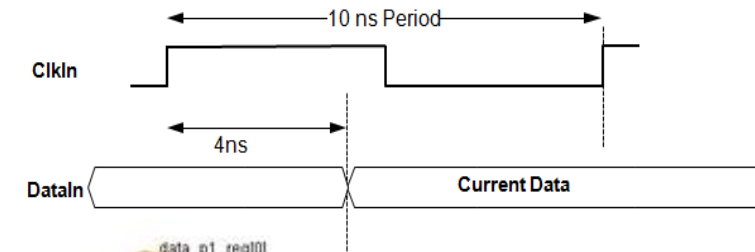
- ▶ A `set_input_delay` can be related to an already existing clock
  - Can be the clock attached to the FPGA clock pin
- ▶ Value used for the `set_input_delay` is the sum of
  - Clock to out of the external source
  - Trace delay on the board

```
create_clock -name SysClk -period 10 [get_ports ClkIn]
set_input_delay -clock SysClk 4 [get_ports DataIn]
```



# set\_input\_delay example

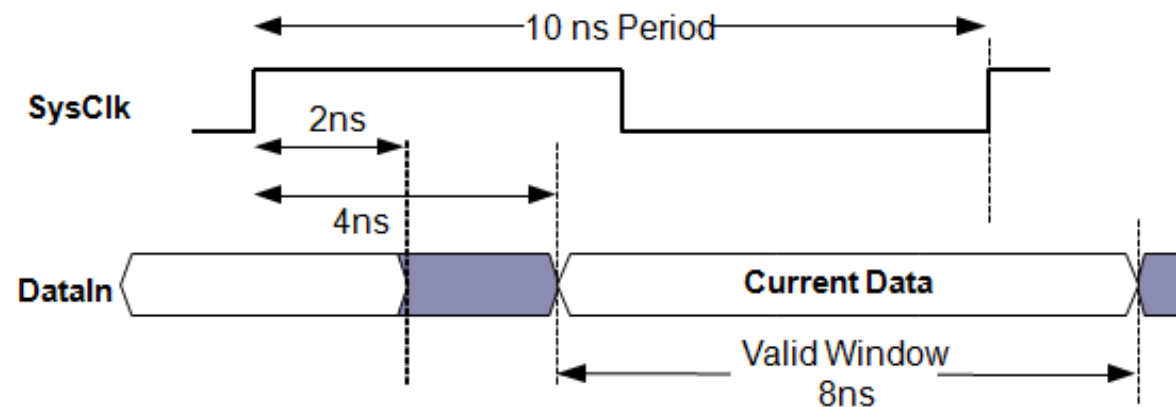
```
create_clock -name SysClk -period 10 [get_ports ClkIn]
set_input_delay -clock SysClk 4 [get_ports DataIn]
```



# Minimum and Maximum Delays

- ▶ By default, each input port can have one maximum delay and one minimum delay
  - Maximum delay is used for the setup check
  - Minimum delay is used for the hold check
- ▶ Without the `-max` or `-min` option, the value supplied is used for both

```
create_clock -name SysClk -period 10 [get_ports ClkIn]
set_input_delay -clock SysClk 4 [get_ports DataIn]
set_input_delay -clock SysClk -min 2 [get_ports DataIn]
```

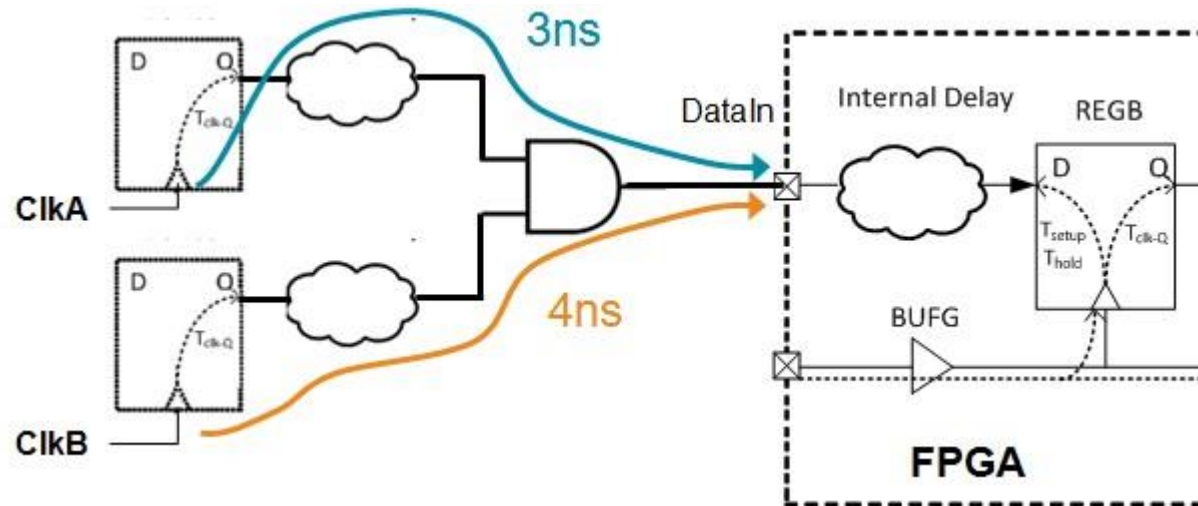




# Multiple Input Delays on the Same Port

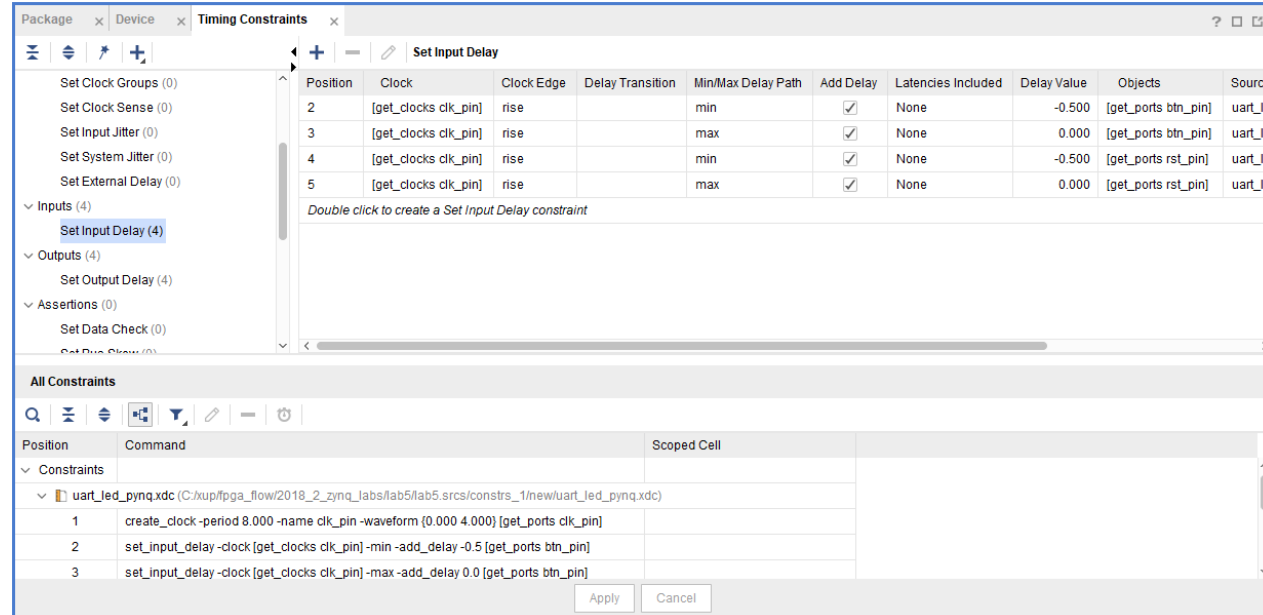
- ▶ An input can have multiple `set_input_delay` commands associated with it
  - Use the `-add_delay` option
  - Results in multiple static timing paths to check

```
set_input_delay -clock ClkA 3 [get_ports DataIn]
set_input_delay -clock ClkB 4 [get_ports DataIn] -add_delay
```



# Creating Input Delays Using the GUI

- ▶ Timing Constraint window can be opened by selecting Window > Timing Constraints
  - Clock can be created by double-clicking Set Input Delay, or a new row in the Set Input Delay table
- ▶ Alternatively, can be set via the Constraints Wizard
  - More details to follow in this presentation



# Set Input Delay Wizard

- ▶ Separate constraint is required for the maximum and minimum
  - Wizard retains its options between invocations, making it easy to specify the minimum after the maximum has been specified

Specify input delay for ports or pins relative to a clock edge.

Clock: [get\_clocks clk\_pin]

Objects (ports): [get\_ports btn\_pin]

Delay value: -0.5 ns

**Delay Value Options**

Delay value is relative to clock edge: rise

Delay value already includes latencies of the specified clock: None

**Rise/Fall**

☐ Delay value specifies rising delay

**Min/Max**

☐ Delay value specifies min delay (shortest path)

☐ Add delay information to the existing delay (no overwrite)

Command: set\_input\_delay -clock [get\_clocks clk\_pin] -0.5 [get\_ports btn\_pin]

Specify input delay for ports or pins relative to a clock edge.

Clock: [get\_clocks clk\_pin]

Objects (ports): [get\_ports btn\_pin]

Delay value: -0.5 ns

**Delay Value Options**

Delay value is relative to clock edge: rise

Delay value already includes latencies of the specified clock: None

**Rise/Fall**

☐ Delay value specifies rising delay

**Min/Max**

☒ Delay value specifies min delay (shortest path)

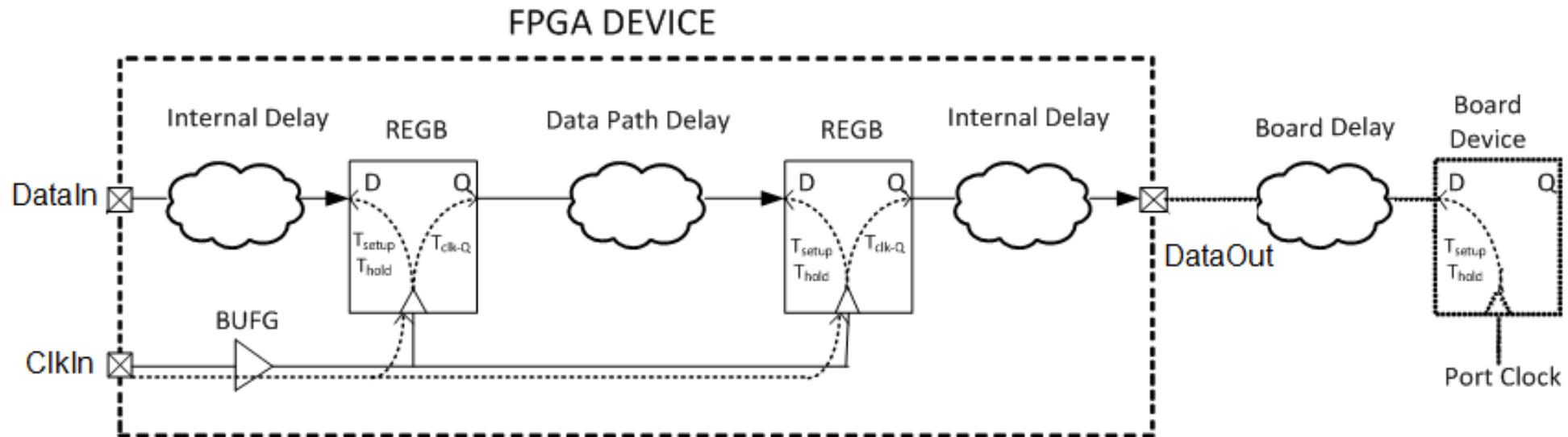
☐ Add delay information to the existing delay (no overwrite)

Command: set\_input\_delay -clock [get\_clocks clk\_pin] -min -0.5 [get\_ports btn\_pin]

# Timing Constraints - Output Delay

# Completing the Static Timing Output Path

- ▶ To complete the static timing path, you need to describe the external elements to the Vivado Design Suite static timing engine
  - What clock is used by the external device
  - Delay between the output port of the FPGA and the external device's clock
    - Includes the required time of the external device and the board delay



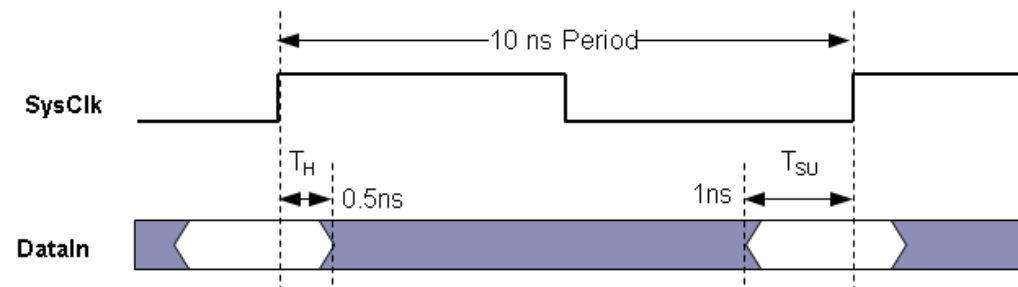
# set\_output\_delay Command

- ▶ `set_output_delay` command supplies the information required to complete the static timing path
  - `set_output_delay -clock <clock_name> <delay> <objects>`
    - `<clock_name>` is the name of the clock used by the external device
      - Can be a real or virtual clock
      - Can be the *name* of a clock; does not need to be a clock object
        - Can use a clock object if desired
    - `<objects>` is the list of objects to which to attach the `set_output_delay`
      - Usually a set of output and/or inout ports
      - Usually uses the `get_ports` command or the `all_outputs` command
    - `<delay>` is the delay from the attached `<objects>` to the external device's clock
      - Includes the external device's requirements and board delay

# External Setup and Hold Requirements

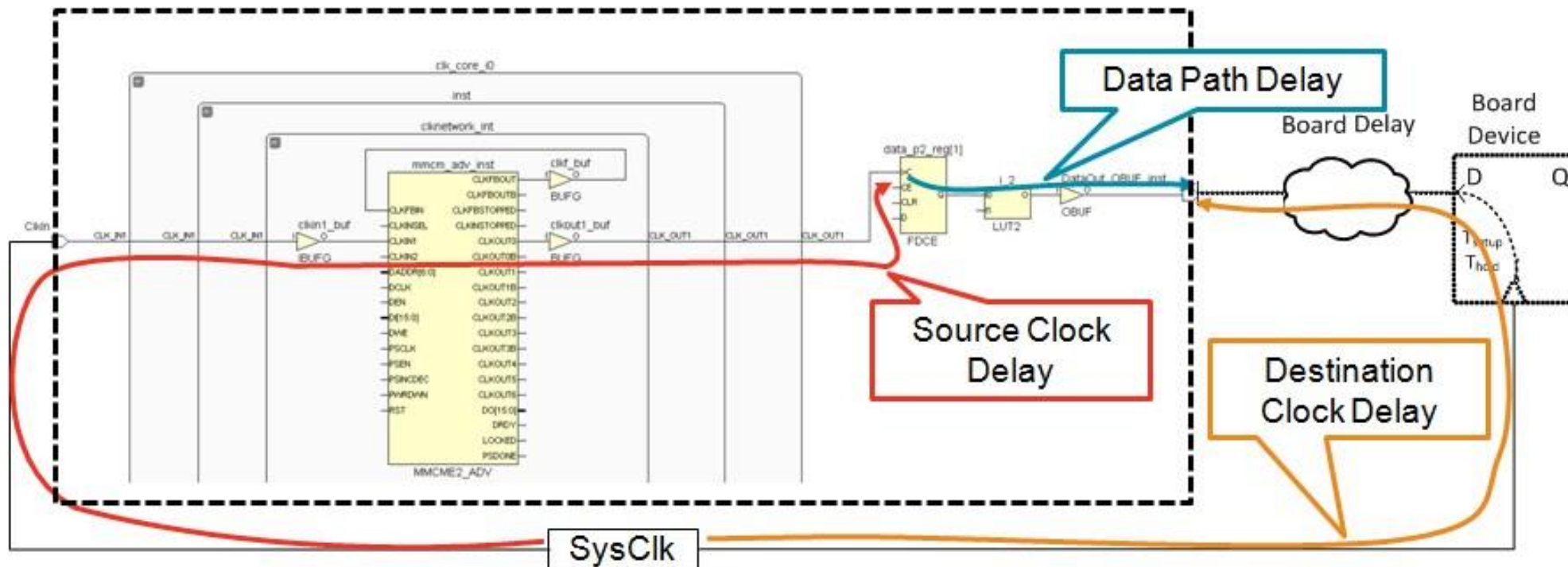
- ▶ External devices need a setup and hold time around the clock
  - `set_output_delay -max` specifies the required setup time
  - `set_output_delay -min` specifies the negative of the required hold time

```
create_clock -name SysClk -period 10 [get_ports ClkIn]
set_output_delay -clock SysClk 1 [get_ports DataIn]
set_output_delay -clock SysClk -min -0.5 [get_ports DataIn]
```



# Complete Output Static Timing Path

- ▶ Output static timing path is segmented slightly differently
  - Data path delay ends at the port of the FPGA
  - Destination clock path traces back through the board device to the port of the FPGA

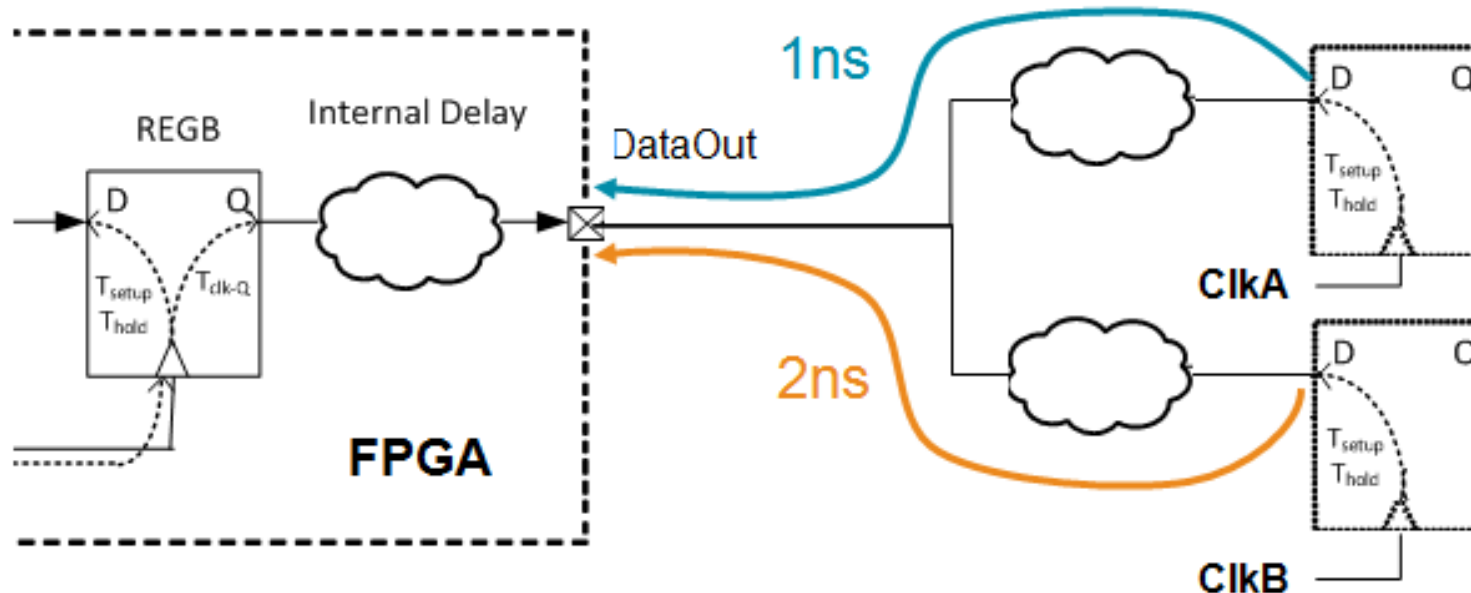




# Multiple Output Delays on the Same Port

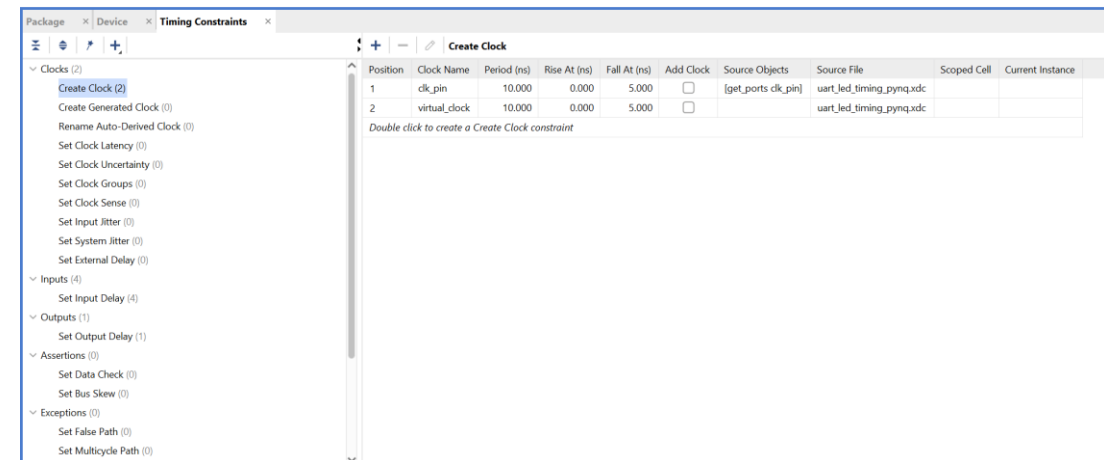
- ▶ An output can have multiple `set_output_delay` commands associated with it
  - Use the `-add_delay` option
  - Results in multiple static timing paths to check

```
set_output_delay -clock ClkA 1 [get_ports DataOut]
set_output_delay -clock ClkB 2 [get_ports DataOut] -add_delay
```



# Creating Output Delays Using the GUI

- ▶ Timing Constraint window can be opened by selecting Window > Timing Constraints
  - Clock can be created by double-clicking Set Output Delay, or a new row in the Set Output Delay table
- ▶ Alternatively, can be set via Constraints Wizard
  - To be covered later in this presentation



# Set Output Delay Wizard

- ▶ Separate constraint is required for the maximum and minimum
  - Wizard retains its options between invocations, making it easy to specify the minimum after the maximum has been specified

Specify output delay for ports or pins relative to a clock edge.

Clock: [get\_clocks clk\_pin]

Objects (ports): [get\_ports {led\_pins[\*]}]

Delay value: -2.25 ns

**Delay Value Options**

Delay value is relative to clock edge: rise

Delay value already includes latencies of the specified clock: None

**Rise/Fall**

☐ Delay value specifies rising delay

**Min/Max**

☐ Delay value specifies min delay (shortest path)

☐ Add delay information to the existing delay (no overwrite)

Command: set\_output\_delay -clock [get\_clocks clk\_pin] -2.25 [get\_ports {led\_pins[\*]}]

? Reference Regret to Defaults OK Cancel

Specify output delay for ports or pins relative to a clock edge.

Clock: [get\_clocks clk\_pin]

Objects (ports): [get\_ports {led\_pins[\*]}]

Delay value: -2.25 ns

**Delay Value Options**

Delay value is relative to clock edge: rise

Delay value already includes latencies of the specified clock: None

**Rise/Fall**

☐ Delay value specifies rising delay

**Min/Max**

☒ Delay value specifies min delay (shortest path)

☐ Add delay information to the existing delay (no overwrite)

Command: set\_output\_delay -clock [get\_clocks clk\_pin] -min -2.25 [get\_ports {led\_pins[\*]}]

? Reference Regret to Defaults OK Cancel

# Timing Constraints - Virtual Clocks

# Clocks for Input and Output Delay

- ▶ Clock specified by the `set_input_delay` and `set_output_delay` can be any clock from the clock database
  - Manually created clock attached to a clock input port of the FPGA
  - Derived clock generated inside the FPGA
    - This is legal, but rarely useful
- ▶ Sometimes the proper clock to use does not already exist
  - Virtual clocks can be created solely for the purpose of specifying input and output delays

# Reasons for Virtual Clocks

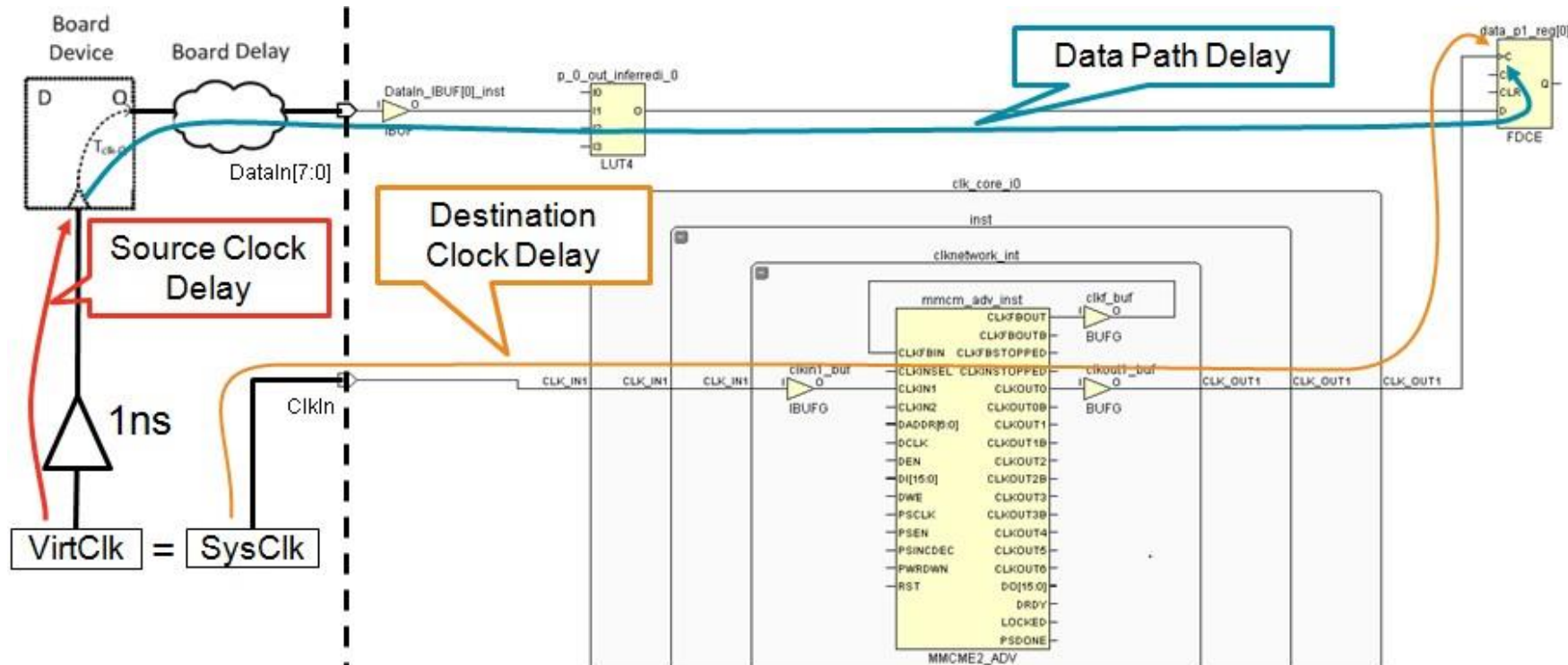
- ▶ There are many reasons for using virtual clocks for clocking I/O
  - Device external to the FPGA uses a different clock than the FPGA
    - Runs at a different frequency
    - Maybe a multiple/division of the FPGA clock
    - Maybe the frequency of an internal FPGA clock generated by an MMCM/PLL
  - Has a different delay path on the board
    - Maybe has a clock buffer chip on the board
- ▶ XDC provides powerful mechanisms for describing clocks
  - Remember, all clocks in XDC are related by default

# Creating Virtual Clocks

- ▶ Virtual clocks are created with `create_clock`
  - Created clock is not attached to any design objects
  - `create_clock -name <name> -period <period>`
    - `<period>` is the period of the clock
    - `<name>` is the user assigned name for the clock
  - Can use the `-waveform` option
- ▶ Can specify jitter with the `set_input_jitter` command
- ▶ Can set clock latency with the `set_clock_latency -source` command
- ▶ Virtual clocks are placed in the design database and can be accessed like other clocks
  - Can be seen via the `report_clocks` command
  - Can be accessed by the `get_clocks` command

# Input Static Timing Path with External Buffer

```
create_clock -name SysClk -period 10 [get_ports ClkIn]
create_clock -name VirtClk -period 10
set_clock_latency -source 1 [get_clocks VirtClk]
set_input_delay -clock VirtClk 4 [get_ports DataIn]
```

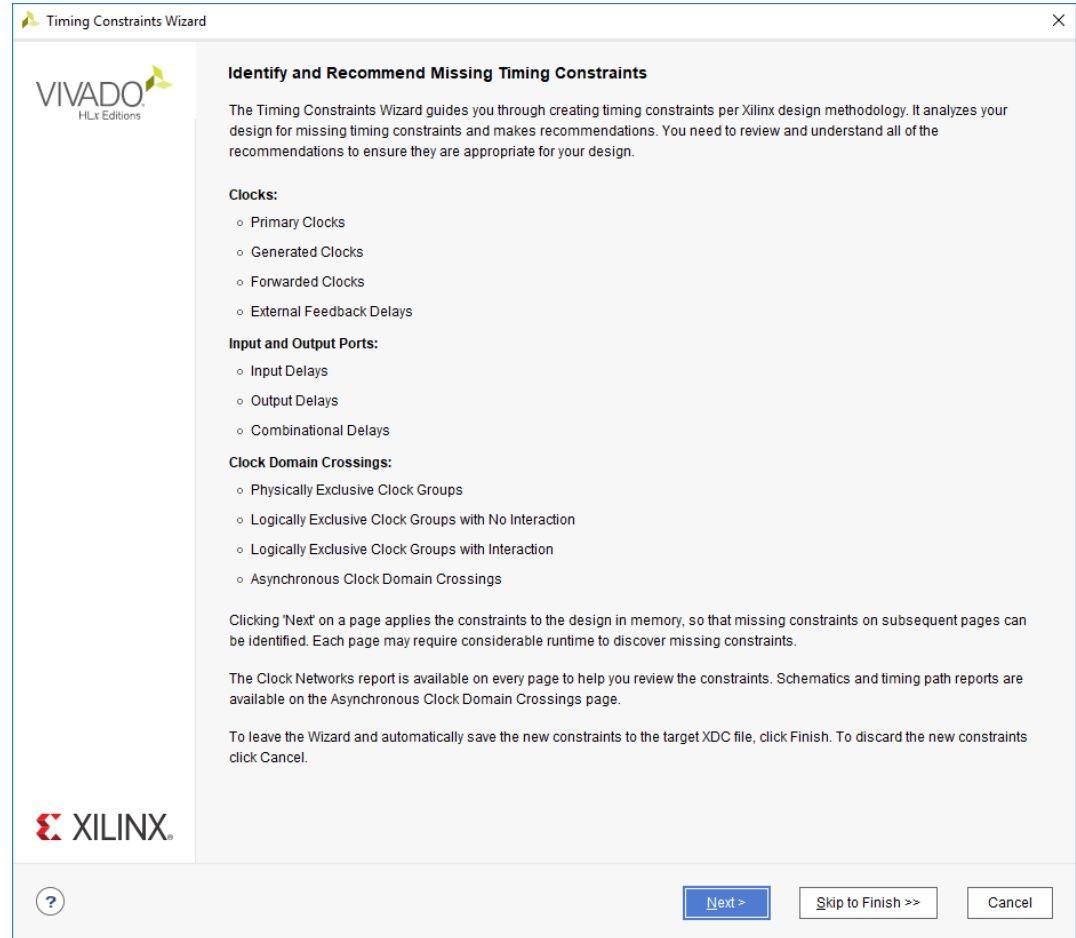




# Constraints Wizard

# The Constraints Wizard

- ▶ Can only be launched after either Synthesis or Implementation
  - Allows Vivado to identify and suggest timing constraints
  - Writes the constraints into the XDC file via the Timing Constraints Editor
  - Not mandatory but HIGHLY recommended
- ▶ User flexibility
  - The user has the choice to ignore the selected constraints
  - The wizard can be launched even if timing constraints have already been entered into the XDC file



# The Constraints Wizard, Primary Clock

- ▶ Defines the input clock waveform
  - Object
  - Name
  - Frequency (MHz)
  - Period (ns)
  - Rise At (ns)
  - Fall At (ns)
- ▶ The resultant Tcl command can be previewed at the bottom of the wizard

Tcl Command Preview (1) Existing Create Clock Constraints (0)

create\_clock -period 10.000 -name clk\_pin -waveform {0.000 5.000} [get\_ports {clk\_pin}]

Timing Constraints Wizard

### Primary Clocks

Primary clocks usually enter the design through input ports. Specify the period and optionally a name and waveform (rising and falling edge times) to describe the duty cycle if not 50%. [More info](#)

#### Recommended Constraints

<input checked="" type="checkbox"/>	Object	Name	Frequency (MHz)	Period (ns)	Rise At (ns)	Fall At (ns)	Jitter (ns)
<input checked="" type="checkbox"/>	clk_pin	clk_pin	undefined	undefined			

#### Constraints for Pulse Width Check Only

<input type="checkbox"/>	Object	Name	Frequency (MHz)	Period (ns)	Rise At (ns)	Fall At (ns)	Jitter (ns)
--------------------------	--------	------	-----------------	-------------	--------------	--------------	-------------

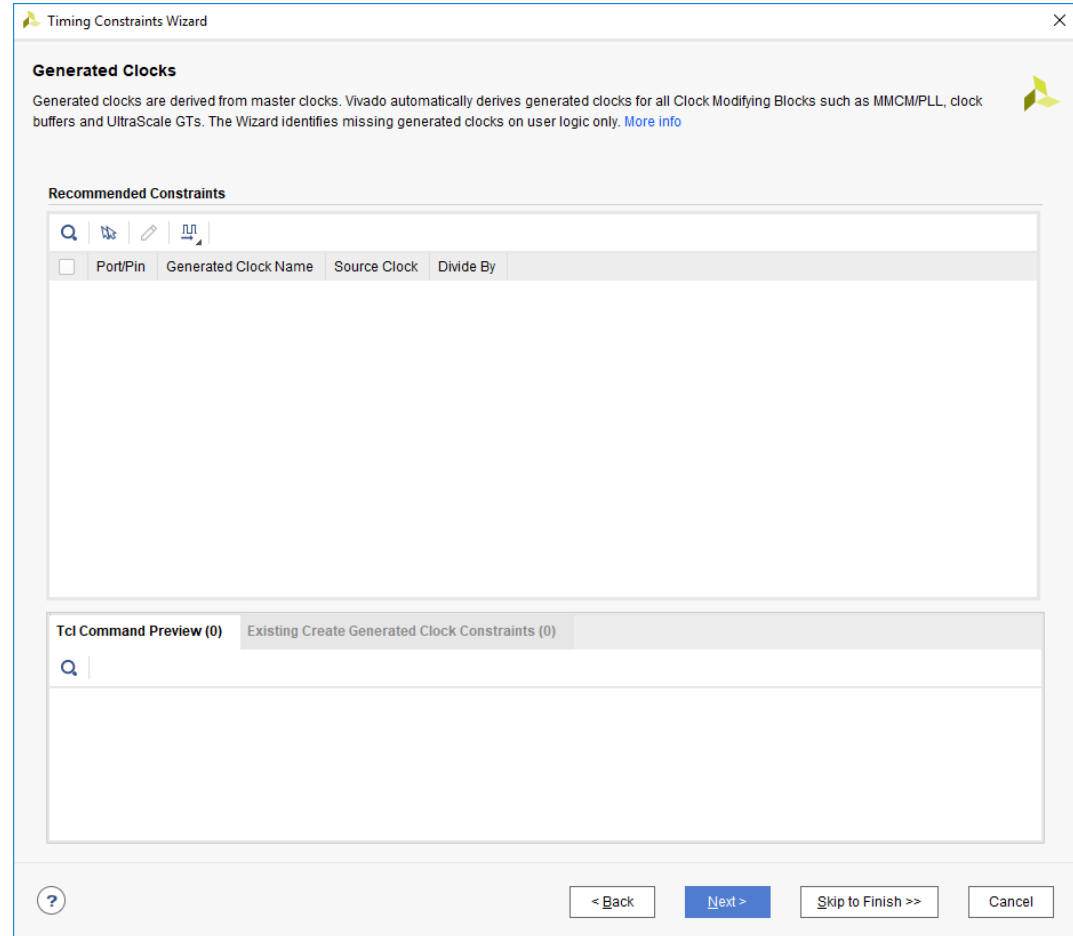
#### Tcl Command Preview (1) Existing Create Clock Constraints (0)

```
create_clock -name clk_pin [get_ports {clk_pin}]
```

< Back Next > Skip to Finish >> Cancel

# The Constraints Wizard, Generated Clocks

- ▶ Vivado analyzes design to locate clocks derived from the master clock
- ▶ Covers derived clocks from a main clock source
  - MMCM/PLL
  - Clock buffers
  - GTs (UltraScale)



# The Constraints Wizard, Input Delays

- ▶ Another way to specify Input Delays
  - Identifies input signals/interfaces going into the FPGA
- ▶ Defines delay parameters
  - tco\_min
  - trce\_dly\_min
  - tco\_max
  - trce\_dly\_max

**Timing Constraints Wizard**

**Input Delays**

Input delays describe relative phase between reference clocks (usually board clocks) and input signals at the FPGA boundary. Inaccurate input delay values can make timing fail and affect implementation quality of results. [More info](#)

**Recommended Constraints**

Interface	Clock	Synchronous	Alignment	Data Rate and Edge
<input checked="" type="checkbox"/> btn_pin	clk_pin	System	Edge	Single Rise
<input checked="" type="checkbox"/> rst_pin	clk_pin	System	Edge	Single Rise

**Delay Parameters**

Clock period: 10 ns

tco\_min: undefined ns

tco\_max: undefined ns

trce\_dly\_min: undefined ns

trce\_dly\_max: undefined ns

Rise Max = tco\_max + trce\_dly\_max  
Rise Min = tco\_min + trce\_dly\_min

**Tcl Command Preview (4)** Existing Set Input Delay Constraints (0) **Waveform - System | Edge | Single Rise**

input clock

data

(tco\_min + trce\_dly\_min)

(tco\_max + trce\_dly\_max)

< Back Next > Skip to Finish >> Cancel

# The Constraints Wizard, Output Delays

- ▶ Another way to specify Output Delays
  - Identifies output signals/interfaces from the FPGA
- ▶ Defines delay parameters
  - tsu
  - trce\_dly\_min
  - thd
  - trce\_dly\_max

The screenshot shows the 'Timing Constraints Wizard' window, specifically the 'Output Delays' tab. The window title is 'Timing Constraints Wizard'. Below the title bar, there's a description: 'Output delays describe relative phase between reference clocks (usually board or forwarded clocks) and output signals at the FPGA boundary. Inaccurate output delay values can make timing fail and affect implementation quality of results. [More info](#)'. Below this is a 'Recommended Constraints' section with a table. The table has columns: 'Interface', 'Clock', 'Synchronous', 'Alignment', 'Data Rate and Edge'. The first row is checked and shows 'led\_pins[\*]' for the interface, 'clk\_pin' for the clock, 'System' for synchronous, 'Setup/Hold' for alignment, and 'Single Rise' for data rate and edge. To the right of the table is a 'Delay Parameters' section with input fields for 'Clock period' (10 ns), 'tsu' (undefined ns), 'thd' (undefined ns), 'trce\_dly\_max' (undefined ns), and 'trce\_dly\_min' (undefined ns). Below these fields are formulas: 'Rise Max = trce\_dly\_max + tsu' and 'Rise Min = trce\_dly\_min - thd'. At the bottom of the wizard is a 'Tcl Command Preview' section with tabs for 'Existing Set Output Delay Constraints (0)' and 'Waveform - System | Setup/Hold | Single Rise'. The waveform section shows a timing diagram with a 'destination clock' and a 'data' signal. The data signal is shown with a setup time of  $(trce\_dly\_max + tsu)$  and a hold time of  $(trce\_dly\_min - thd)$  relative to the clock edge. At the bottom of the window are navigation buttons: '< Back', 'Next >', 'Skip to Finish >>', and 'Cancel'.

Interface	Clock	Synchronous	Alignment	Data Rate and Edge
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
led_pins[*]	clk_pin	System	Setup/Hold	Single Rise

Delay Parameters

Clock period: 10 ns

tsu: undefined ns

thd: undefined ns

trce\_dly\_max: undefined ns

trce\_dly\_min: undefined ns

Rise Max = trce\_dly\_max + tsu

Rise Min = trce\_dly\_min - thd

Tcl Command Preview (2) | Existing Set Output Delay Constraints (0) | Waveform - System | Setup/Hold | Single Rise

destination clock

data

$(trce\_dly\_max + tsu)$

$(trce\_dly\_min - thd)$

# The Constraints Wizard: Other Constraints

- ▶ Forwarded Clocks
  - The Forwarded Clocks constraint covers clock reference for output signals associated with “downstream” devices clocked by an FPGA output pin
- ▶ External Feedback Delays
  - Pertinent if the MMCM/PLL feedback loops used in the design is routed out of the FPGA and back into the device via input and output ports
- ▶ Combinatorial Delay
  - Cover paths that traverse the FPGA without being captured by any sequential elements
- ▶ Physically Exclusive Clock Groups
  - Define clocks that do not exist in the design at the same time

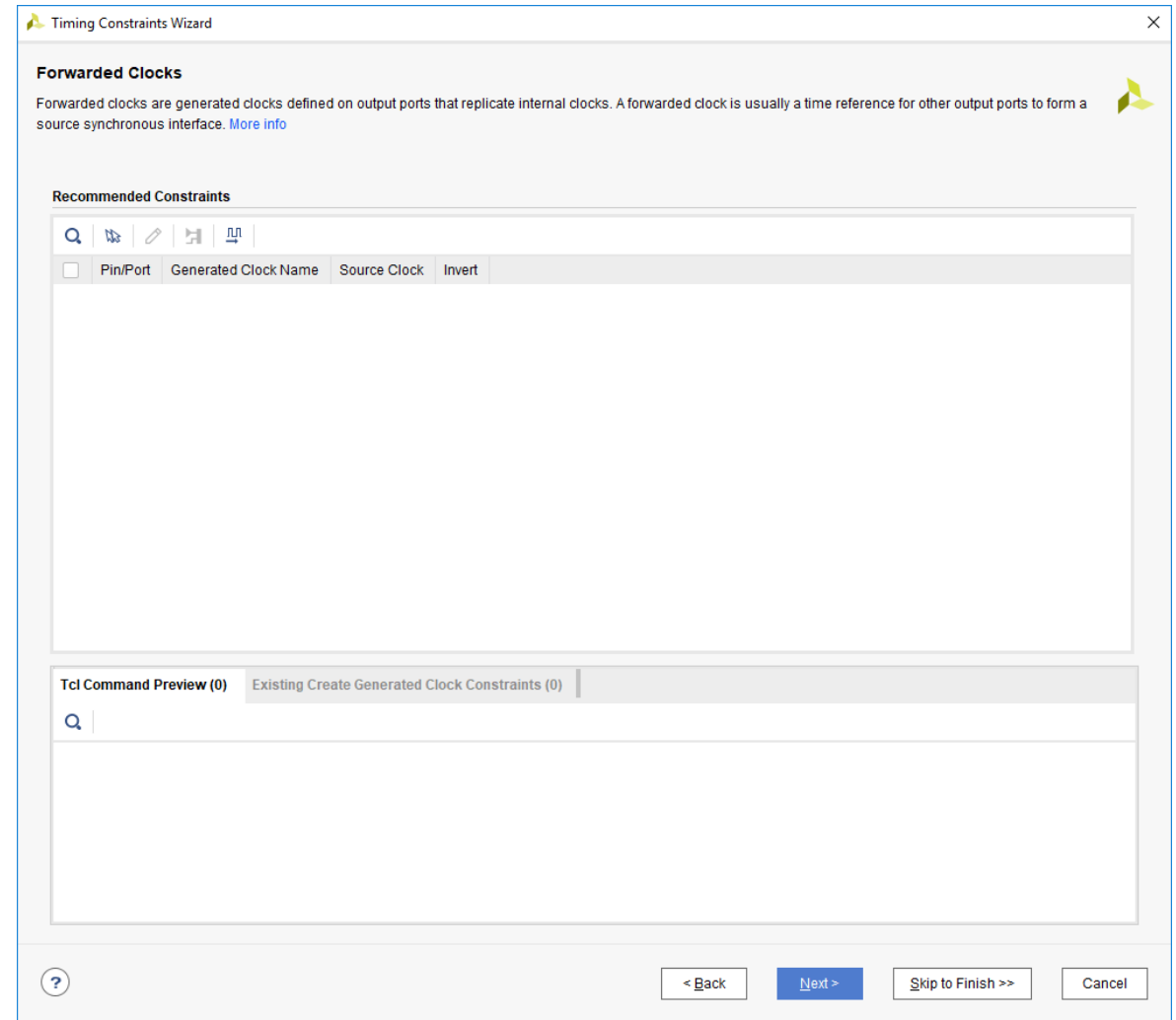
# The Constraints Wizard: Other Constraints

- ▶ Logically Exclusive Clock Groups with No Interaction
  - Define logically exclusive clocks that do not have paths between each other outside of shared sections (such as clock trees)
- ▶ Logically Exclusive Clock Groups with Interaction
  - Define logically exclusive clocks that have paths between each other, only clocks limited to the shared clock tree sections are logically exclusive
- ▶ Asynchronous Clock Domain Crossings
  - Define paths that transfer data between two clocks without a known phase relationship



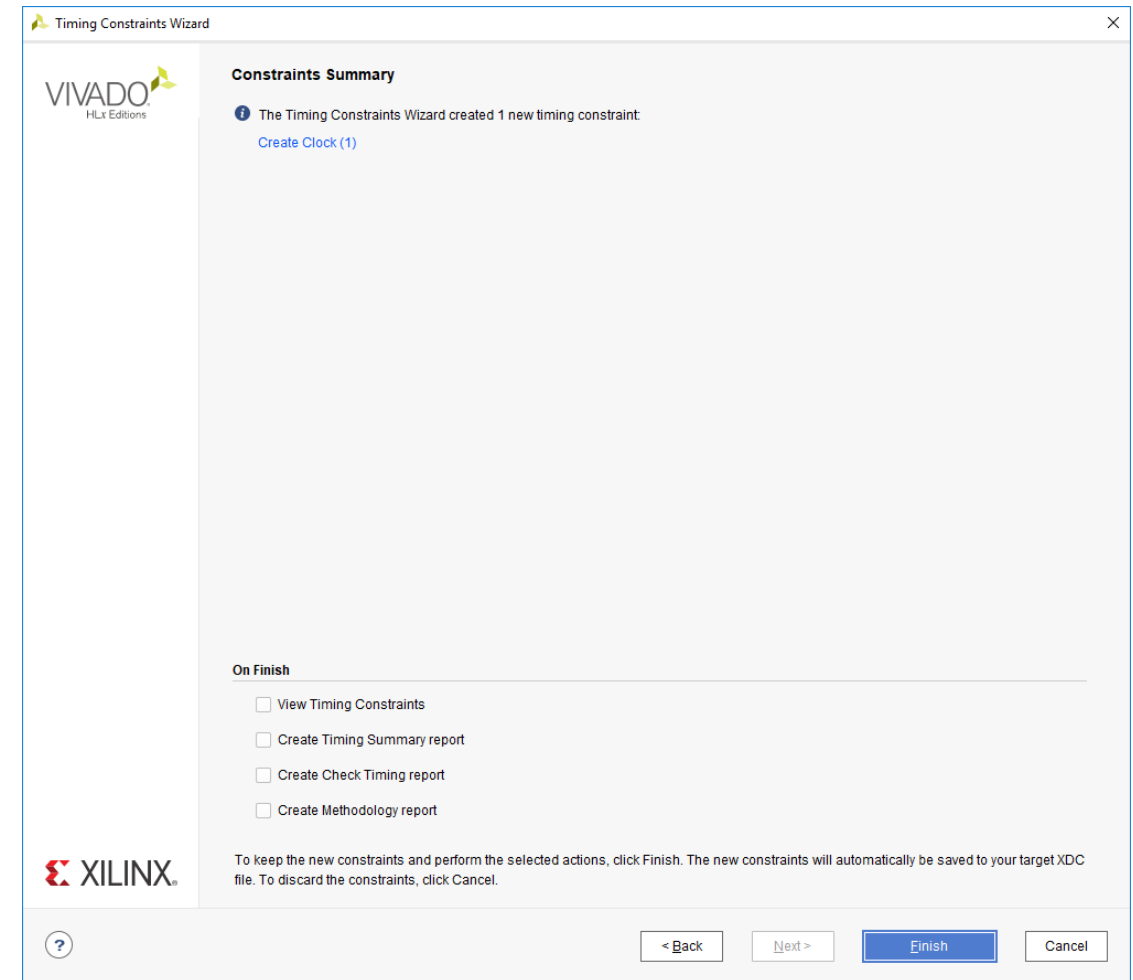
# The Constraints Wizard: Non-Applicable Constraints

- Constraints that are not applicable to the design do not have any options shown in the wizard (blank grey screen)



# The Constraints Wizard: Constraints Summary

- ▶ The final screen of the wizard summarizes all the constraints selected for insertion into the XDC file
  - Generated constraints are not immediately applied to the target XDC
  - On Finish options allow for follow-up operations
    - View Timing Constraints
    - Create Timing Summary Report
    - Create Check Timing Report
    - Create DRC report using only timing checks



# Summary

# Summary

- ▶ The I/O Planner view in the Vivado IDE provides an easy-to-use interface for assigning pin locations
- ▶ Use I/O Planning project for pin planning early in the design analysis
  - DRC checking
  - SSO analysis
  - Verify I/O banking rules
- ▶ Static timing paths start at clocked elements and end at clocked elements
- ▶ Static timing paths are analyzed for setup and hold violations at both fastest and slowest process corners
- ▶ Clocks are objects
- ▶ Clocks can be created with the `create_clock` command

# Summary

- ▶ `set_input_delay` and `set_output_delay` commands provide the details to complete the static timing path
- ▶ `set_input_delay` specifies the clock of the driving component and the delay to the port
- ▶ `set_output_delay` specifies the clock of the receiving component and the delay to the port
  - `-max` is the required setup time
  - `-min` is the negative of the required hold time
- ▶ Port can have one `-min` and one `-max` by default
- ▶ Additional delays can be specified with the `-add_delay` option
- ▶ I/O delays can be specified with respect to virtual clocks
- ▶ The Constraints Wizard is a powerful tool for creating constraints pertinent to the design



# Thank You

## Disclaimer and Attribution

The information contained herein is for informational purposes only and is subject to change without notice. While every precaution has been taken in the preparation of this document, it may contain technical inaccuracies, omissions and typographical errors, and AMD is under no obligation to update or otherwise correct this information. Advanced Micro Devices, Inc. makes no representations or warranties with respect to the accuracy or completeness of the contents of this document, and assumes no liability of any kind, including the implied warranties of noninfringement, merchantability or fitness for particular purposes, with respect to the operation or use of AMD hardware, software or other products described herein. No license, including implied or arising by estoppel, to any intellectual property rights is granted by this document. Terms and limitations applicable to the purchase or use of AMD's products are as set forth in a signed agreement between the parties or in AMD's Standard Terms and Conditions of Sale. GD-18

© Copyright 2022 Advanced Micro Devices, Inc. All rights reserved. Xilinx, the Xilinx logo, AMD, the AMD Arrow logo, Alveo, Artix, Kintex, Kria, Spartan, Versal, Vitis, Virtex, Vivado, Zynq, and other designated brands included herein are trademarks of Advanced Micro Devices, Inc. Other product names used in this publication are for identification purposes only and may be trademarks of their respective companies.

