



Quartus® II Software Design Series: Timing Analysis

Design Verification

- Most PLD development time is spent verifying your design
- Verification includes
 - Timing analysis
 - Simulation (internal & system-level)
 - Formal verification
 - Power analysis
 - Signal integrity analysis
 - In-system testing

Quartus II Software Support

- Quartus® II software provides features to aid & accelerate the verification process
- TimeQuest Timing Analyzer (TA)
- Quartus II Simulator and 3rd-party support*
- PowerPlay Power Analyzer*
- Debugging tools (in-system testing)*

**These topics are covered in the “Quartus II Software Design Series: Verification” course*

Other Quartus II Design Series courses

■ Quartus II Software Design Series: Foundation

- Project creation and management
- Design entry methods and tools
- Compilation and compilation results analysis
- Creating and editing settings and assignments
- I/O planning and management
- Introduction to timing analysis with the TimeQuest timing analyzer

■ Quartus II Software Design Series: Verification

- Basic design simulation with ModelSim-Altera
- Power analysis
- Debugging solutions

■ Quartus II Software Design Series: Optimization

- Incremental compilation
- Quartus II optimization features & techniques

Objectives

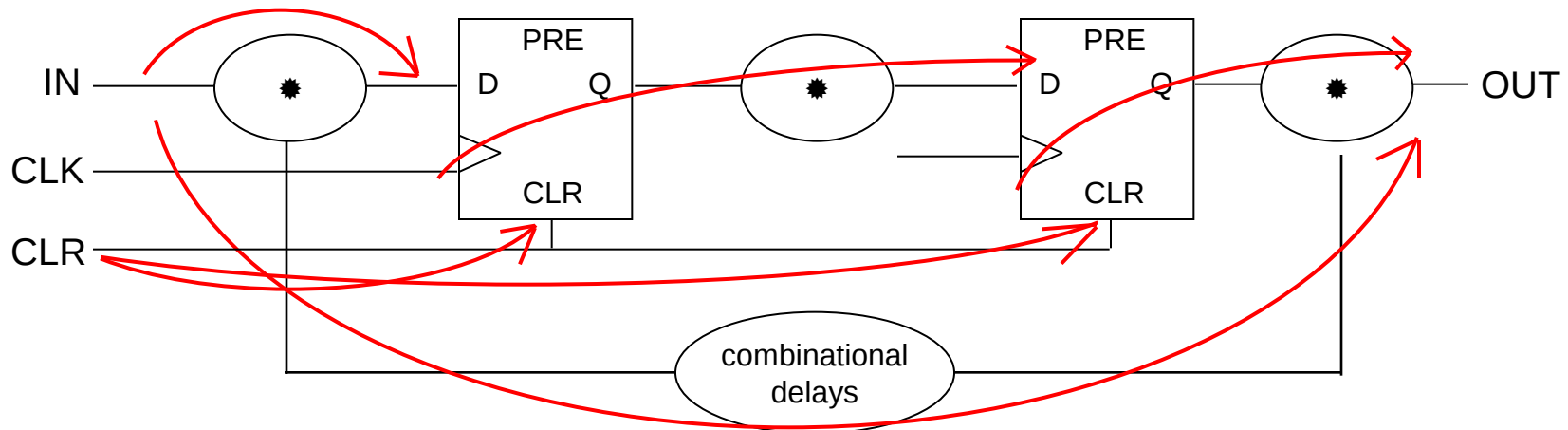
- Display a complete understanding of timing analysis
- Build SDC files for constraining PLD designs
- Verify timing on simple & complex designs using TimeQuest TA

Class Agenda

- TimeQuest basics
- Timing analysis basics
 - **Exercise 1**
- TimeQuest reporting
- Clock constraints
 - **Exercise 2**
- Synchronous I/O constraints
 - **Exercise 3**
- Source Synchronous I/O constraints
 - **Exercise 4**
- Constraining asynchronous signals
- Timing exceptions
 - False paths
 - Multicycle constraints
 - **Exercise 5**

How does timing verification work?

- Every device path in design must be analyzed with respect to timing specifications/requirements
 - Catch timing-related errors faster and easier than gate-level simulation & board testing
- Designer must enter timing requirements & exceptions
 - Used to guide fitter during placement & routing
 - Used to compare against actual results





Quartus® II Software Design Series: Timing Analysis

TimeQuest Basics

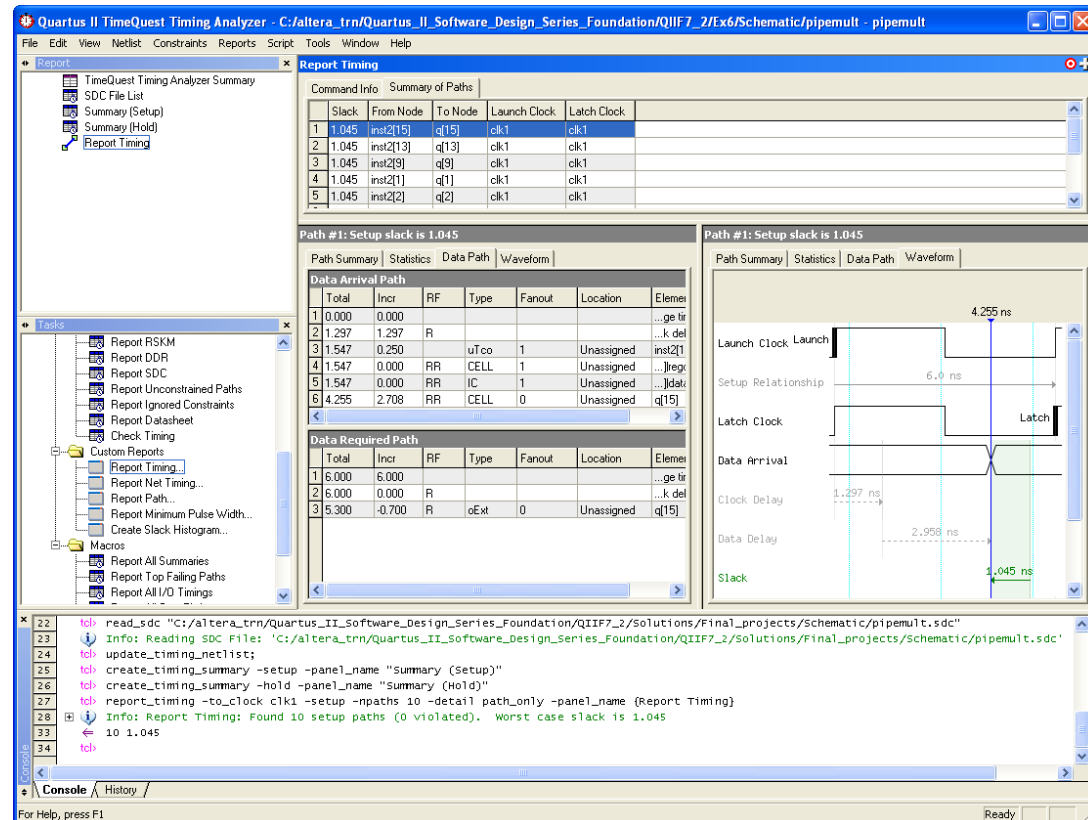


Timing Analysis Agenda

- TimeQuest basics ←
- Timing analysis basics
- Timing reports
- Timing constraints
- Example application

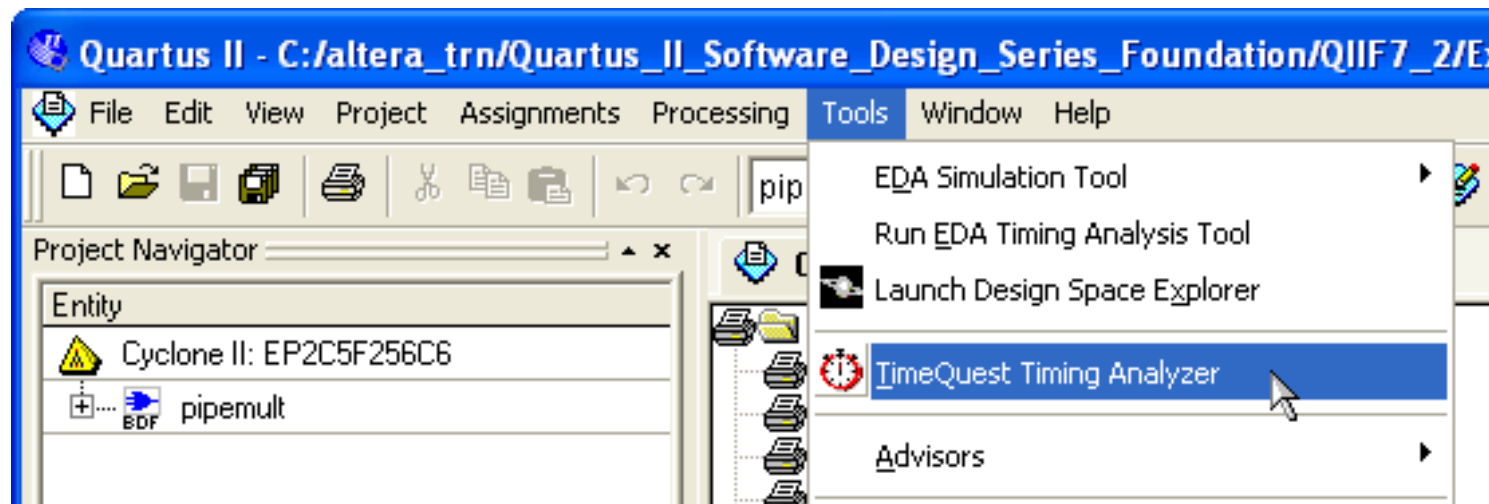
TimeQuest Timing Analyzer

- Timing engine in Quartus II software
- Provides timing analysis solution for all levels of experience
- Features
 - Synopsys Design Constraints (SDC) support
 - Standardized constraint methodology
 - Easy-to-use interface
 - Constraint entry
 - Standard reporting
 - Scripting emphasis
 - Presentation focuses on using GUI



Opening the TimeQuest Interface

- Toolbar button 
- **Tools** menu
- Tasks window
- Stand-alone mode
 - `quartus_staw`
- Command line



Quartus Settings File (QSF)

- SDC constraints are **not** stored in QSF
- For 90nm and older devices, TimeQuest TA uses script to convert QSF timing assignments to SDC
 - TimeQuest **Constraints** menu
 - Done automatically if no SDC file exists when first opening timing analyzer



- See Quartus II Handbook chapter, *Switching to the TimeQuest Timing Analyzer* for details
 - Differences between Classic Timing Analyzer and TimeQuest TA
 - Details on conversion utility
- Online training also available
 - [Switching to the TimeQuest Timing Analyzer](#)

TimeQuest GUI

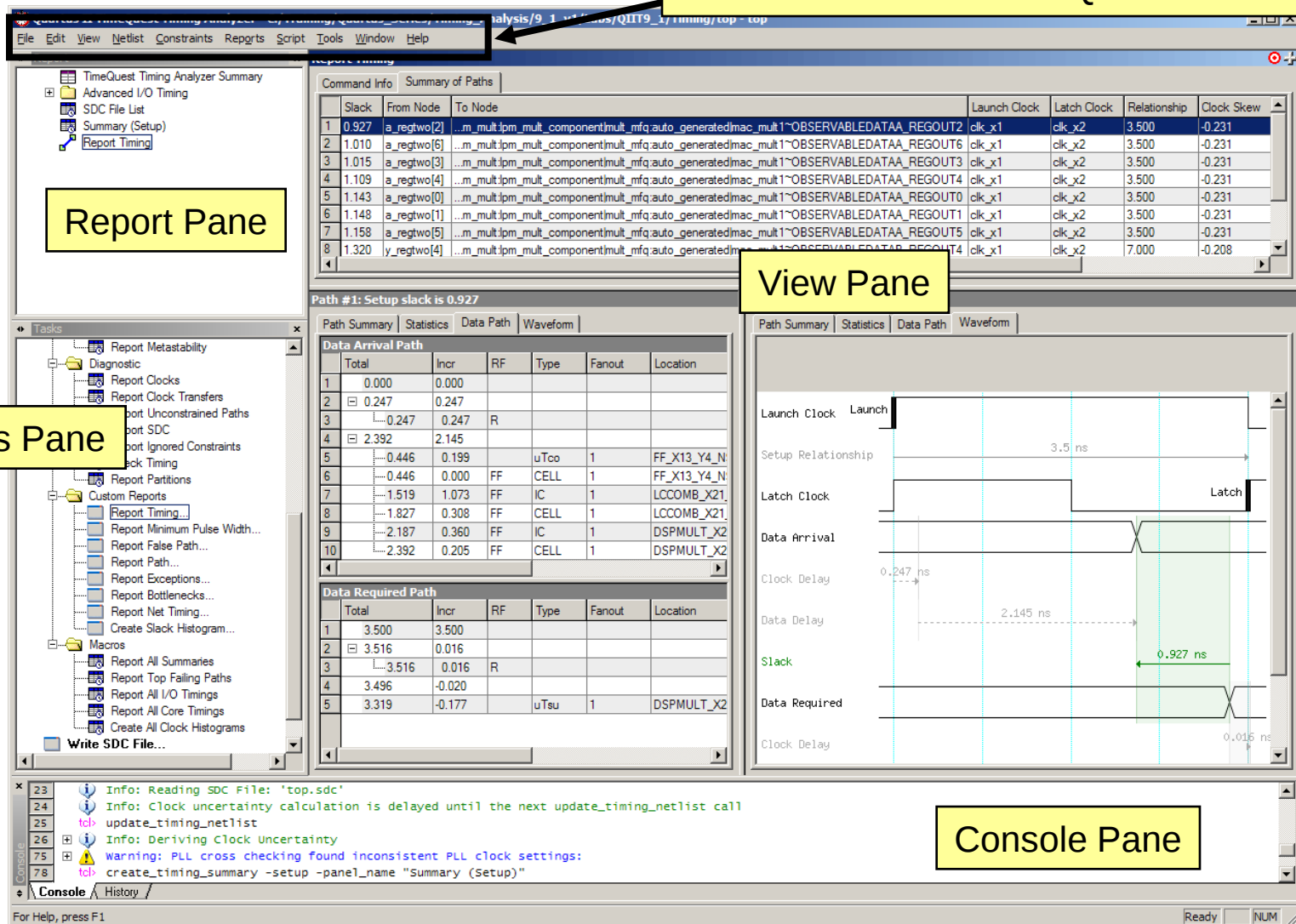
Menu access to all TimeQuest features

Report Pane

Tasks Pane

View Pane

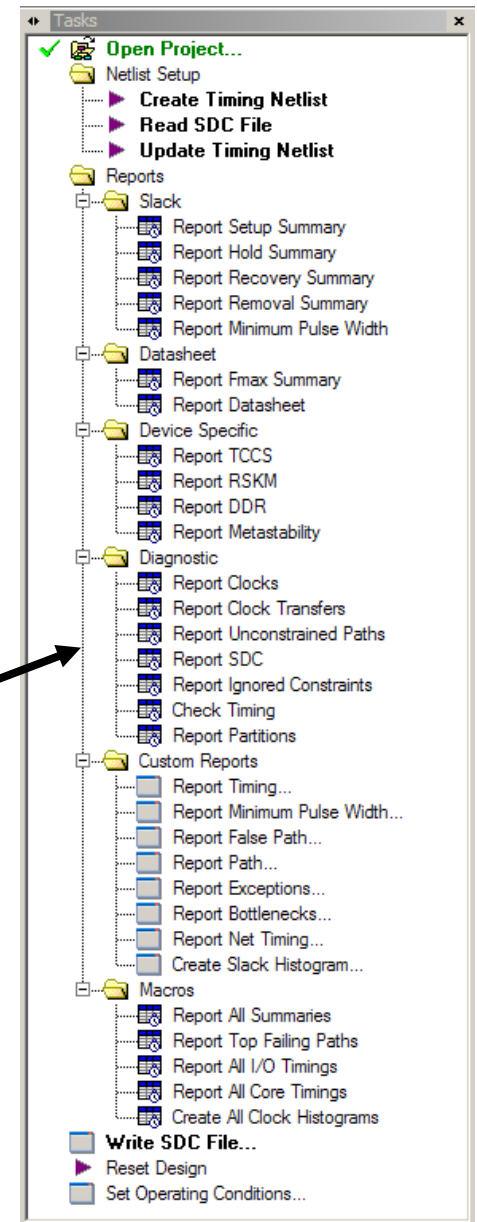
Console Pane



Tasks Pane

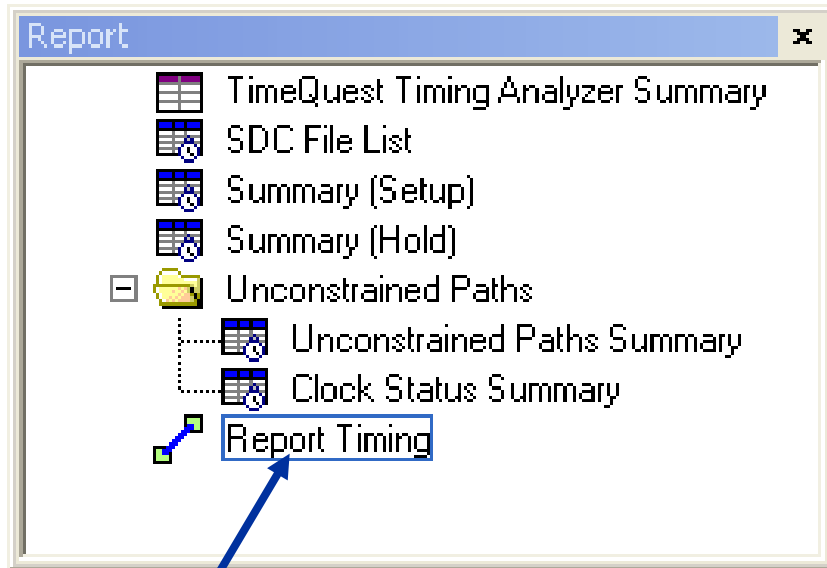
- Provides quick access to common operations
 - Command execution
 - Report generation
- Executes most commands with **default** settings
- Use menus for non-default settings

Double-click to
execute any command



Report Pane

- Displays list of generated reports currently available for viewing
 - Reports generated by Tasks pane
 - Reports generated using report commands



Highlight report to see detail in View window

Viewing Multiple Reports

Report Timing

Command Info | Summary of Paths

	Slack	From Node	To Node	Launch Clock	Latch Clock	Delay
1	0.927	a_regtwo[2]	...A_REGOUT2	clk_x1	clk_x2	
2	1.010	a_regtwo[6]	...A_REGOUT6	clk_x1	clk_x2	
3	1.015	a_regtwo[3]	...A_REGOUT3	clk_x1	clk_x2	
4	1.109	a_regtwo[4]	...A_REGOUT4	clk_x1	clk_x2	
5	1.143	a_regtwo[0]	...A_REGOUT0	clk_x1	clk_x2	3.500 -0.231 1.929
6	1.148	a_regtwo[1]	...A_REGOUT1	clk_x1	clk_x2	3.500 -0.231 1.924

Click & drag '+' sign to divide view pane into multiple windows

Path #1: Setup slack is 0.927

Path Summary | Statistics | Data Path | Waveform

Data Arrival Path

	Total	Incr	RF	Type	Fanout	Local
1	0.000	0.000				
2	0.247	0.247				
3	0.247	0.247	R			

Data Required Path

	Total	Incr	RF	Type	Fanout	Local
1	3.500	3.500				
2	3.516	0.016				
3	3.516	0.016	R			
4	3.496	-0.020				
5	3.319	-0.177		uTsu	1	DSPM

Path #1: Setup slack is 0.927

Path Summary | Statistics | Data Path | Waveform

Launch Clock Launch

Setup Relationship 3.5 ns

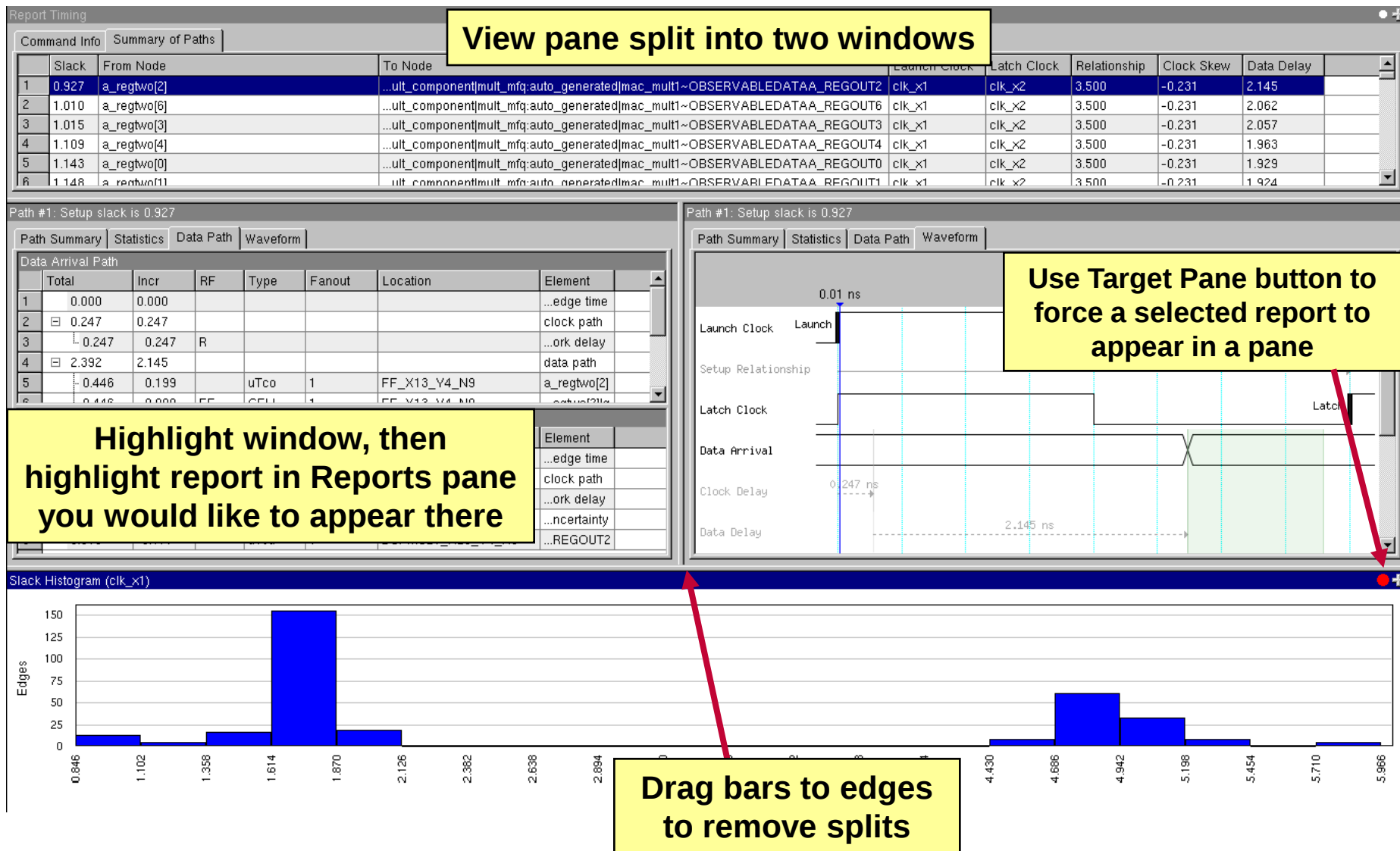
Latch Clock Latch

Data Arrival

Clock Delay 0.247 ns

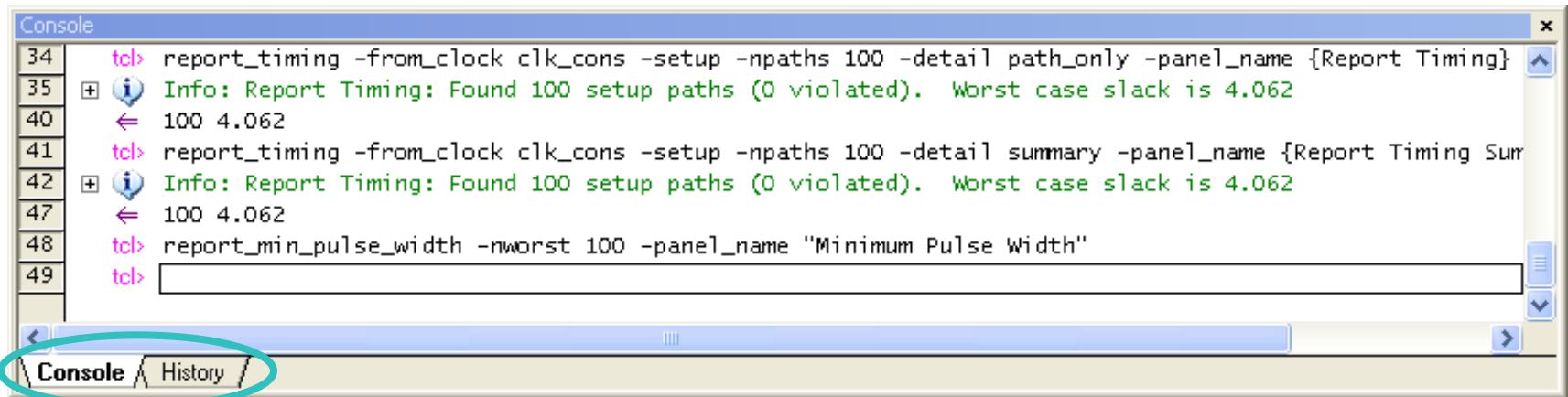
Data Delay 2.145 ns

Viewing Multiple Reports Example



Console pane

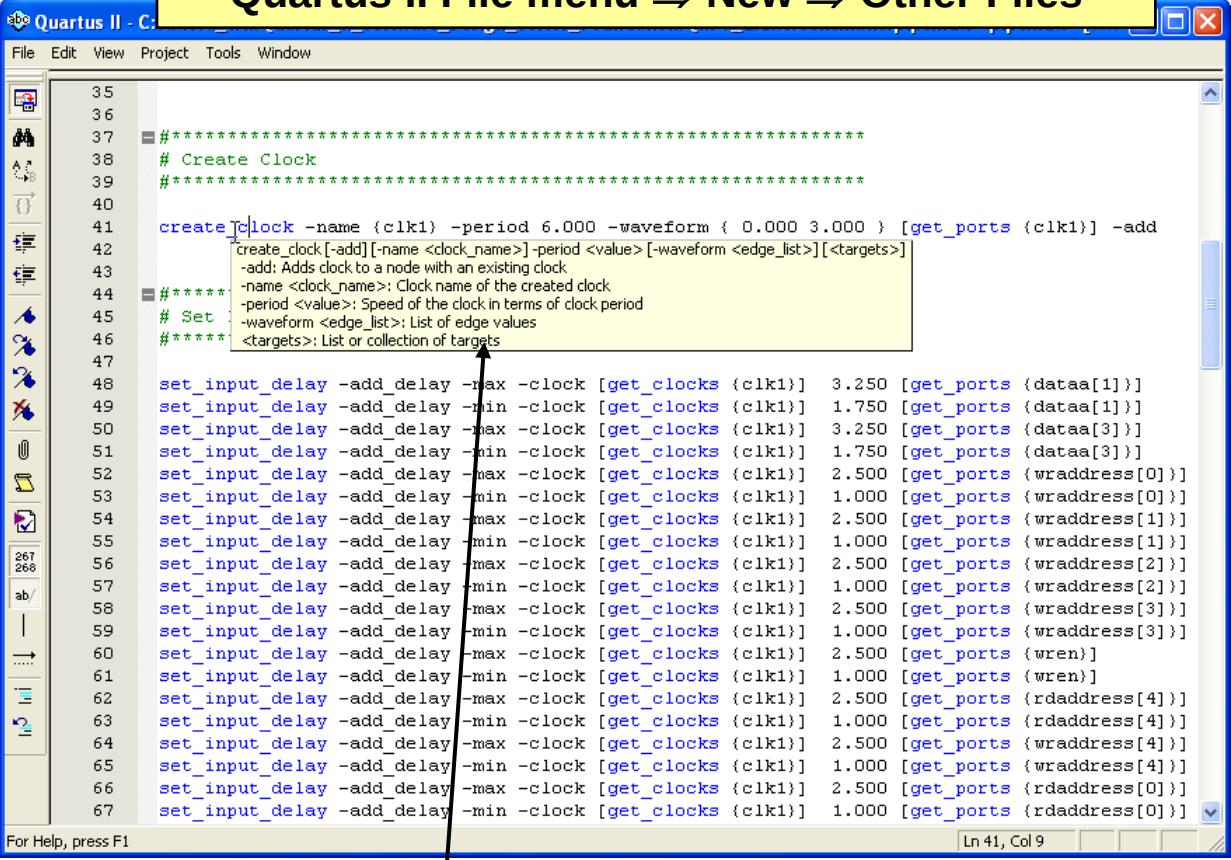
- Allows direct entry and execution of SDC & Tcl commands
 - Displays equivalent of command executed by GUI
- Displays TimeQuest output messages
- History tab records all executed SDC & Tcl commands
 - Copy & paste to create scripts or SDC files



SDC File Editor = Quartus II Text Editor

- Use Quartus II editor to create and/or edit SDC
- SDC editing unique features (for .sdc files)
 - Access to GUI dialog boxes for constraint entry (**Edit ⇒ Insert Constraint**)
 - Syntax coloring
 - Tooltip syntax help
 - SDC templates

TimeQuest File menu ⇒ New/Open SDC File
Quartus II File menu ⇒ New ⇒ Other Files



The screenshot shows the Quartus II SDC File Editor with the following SDC commands:

```
35  
36  
37 *****  
38 # Create Clock  
39 *****  
40  
41 create_clock -name {clk1} -period 6.000 -waveform { 0.000 3.000 } [get_ports {clk1}] -add  
42  
43  
44 *****  
45 # Set  
46 *****  
47  
48 set_input_delay -add_delay -max -clock [get_clocks {clk1}] 3.250 [get_ports {dataa[1]}]  
49 set_input_delay -add_delay -min -clock [get_clocks {clk1}] 1.750 [get_ports {dataa[1]}]  
50 set_input_delay -add_delay -max -clock [get_clocks {clk1}] 3.250 [get_ports {dataa[3]}]  
51 set_input_delay -add_delay -min -clock [get_clocks {clk1}] 1.750 [get_ports {dataa[3]}]  
52 set_input_delay -add_delay -max -clock [get_clocks {clk1}] 2.500 [get_ports {waddress[0]}]  
53 set_input_delay -add_delay -min -clock [get_clocks {clk1}] 1.000 [get_ports {waddress[0]}]  
54 set_input_delay -add_delay -max -clock [get_clocks {clk1}] 2.500 [get_ports {waddress[1]}]  
55 set_input_delay -add_delay -min -clock [get_clocks {clk1}] 1.000 [get_ports {waddress[1]}]  
56 set_input_delay -add_delay -max -clock [get_clocks {clk1}] 2.500 [get_ports {waddress[2]}]  
57 set_input_delay -add_delay -min -clock [get_clocks {clk1}] 1.000 [get_ports {waddress[2]}]  
58 set_input_delay -add_delay -max -clock [get_clocks {clk1}] 2.500 [get_ports {waddress[3]}]  
59 set_input_delay -add_delay -min -clock [get_clocks {clk1}] 1.000 [get_ports {waddress[3]}]  
60 set_input_delay -add_delay -max -clock [get_clocks {clk1}] 2.500 [get_ports {wren}]  
61 set_input_delay -add_delay -min -clock [get_clocks {clk1}] 1.000 [get_ports {wren}]  
62 set_input_delay -add_delay -max -clock [get_clocks {clk1}] 2.500 [get_ports {rdaddress[4]}]  
63 set_input_delay -add_delay -min -clock [get_clocks {clk1}] 1.000 [get_ports {rdaddress[4]}]  
64 set_input_delay -add_delay -max -clock [get_clocks {clk1}] 2.500 [get_ports {waddress[4]}]  
65 set_input_delay -add_delay -min -clock [get_clocks {clk1}] 1.000 [get_ports {waddress[4]}]  
66 set_input_delay -add_delay -max -clock [get_clocks {clk1}] 2.500 [get_ports {rdaddress[0]}]  
67 set_input_delay -add_delay -min -clock [get_clocks {clk1}] 1.000 [get_ports {rdaddress[0]}]
```

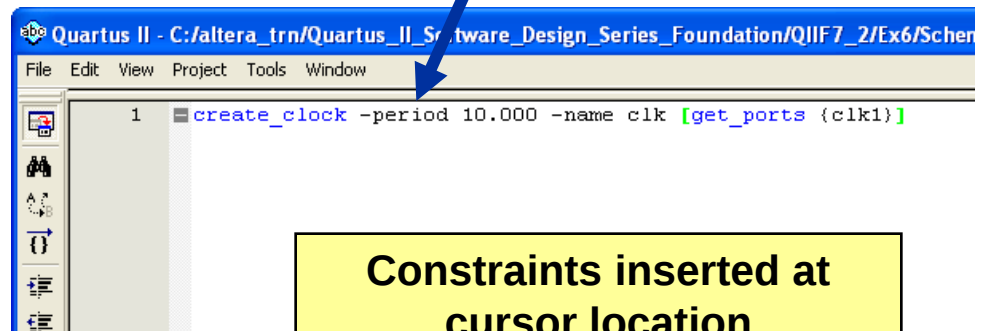
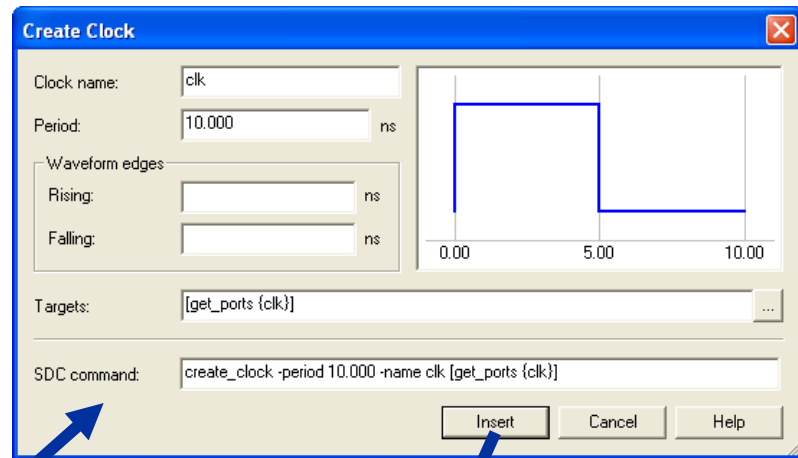
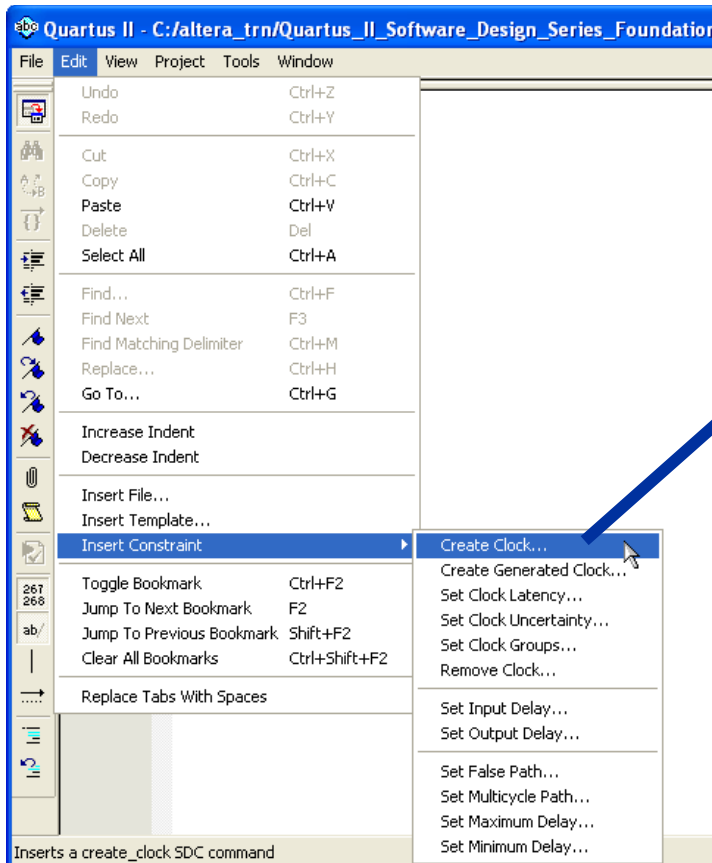
A tooltip is displayed over the `create_clock` command on line 41, showing the following details:

- `-add`: Adds clock to a node with an existing clock
- `-name <clock_name>`: Clock name of the created clock
- `-period <value>`: Speed of the clock in terms of clock period
- `-waveform <edge_list>`: List of edge values
- `<targets>`: List or collection of targets

Place cursor over
command to see tooltip

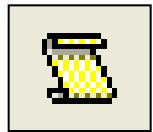
SDC File Editor (cont.)

Construct an SDC file using
TimeQuest graphical
constraint creation tools



Constraints inserted at
cursor location

SDC Templates



- Quickly add customized constraint templates

Insert Template

Language templates:

- TimeQuest
 - SDC Commands
 - Collections
 - Clocks
 - Clock Attributes
 - I/O Delays
 - Exceptions
 - Scripting
 - Basic Multi-Corner Signoff
 - SDC Cookbook**
 - Create Base Clocks and PLL C
 - Create Base Clocks Manually a
 - Create Base Clocks Manually a
 - 2:1 Clock Muxing
 - Externally Switched Clock Con
 - PLL Clock Switchover
 - System Synchronous Input Cor
 - System Synchronous Output C
 - ISU, tH, and tCO Constraints
 - Multicycle Clock-to-Clock
 - Multicycle Clock-to-Register
 - False Path Clock-to-Clock
 - False Path Clock-to-Register
 - JTAG Signal Constraints
 - User

Preview:

```
# Create the source clock
create_clock -period __period -name __sourceClock [get_ports __port]
# Create the destination clock
create_clock -period __period -name __destinationClock [get_ports __port]
# Set a multicycle exception from a source register clocked by __sourceClock
# to a destination register __destinationRegisterInputPin
set_multicycle_path -from [get_clocks __sourceClock] -to [get_pins __destinationRegisterInputPin]
```

Save User Template

You made changes to an Altera-provided template, however you cannot save changes to the template. If you want to save the changes, you can save the template as a user template.

User template name:

OK Cancel

Save Insert Close

Toolbar button or Edit menu

SDC "Cookbook" templates from TimeQuest TA Online Resource Center

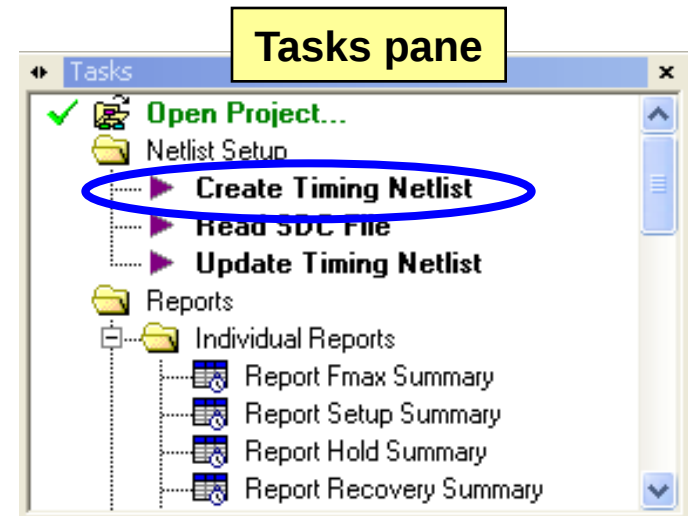
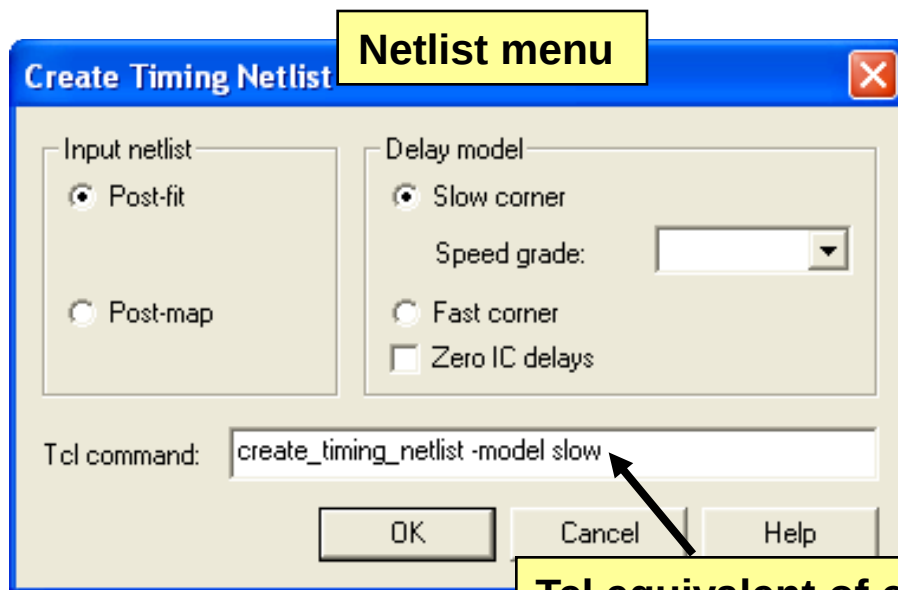
Preview window: edit before inserting & save as user template

Basic Steps to Using TimeQuest TA

1. Generate timing netlist
2. Enter SDC constraints
 - a. Create and/or read in SDC file (recommended method)
 - or***
 - b. Constrain design directly in console
3. Update timing netlist
4. Generate timing reports
5. Save timing constraints (optional)

1) Generate Timing Netlist

- Create a timing netlist (i.e. database) based on compilation results
 - Post-synthesis (mapping) or post-fit (if design already fully compiled)
 - Worst-case (slow; maximum operating temperature), best-case (fast; minimum operating temperature) timing models
 - Set custom operating conditions (65 nm technology devices; military; industrial, etc.)
- To execute:

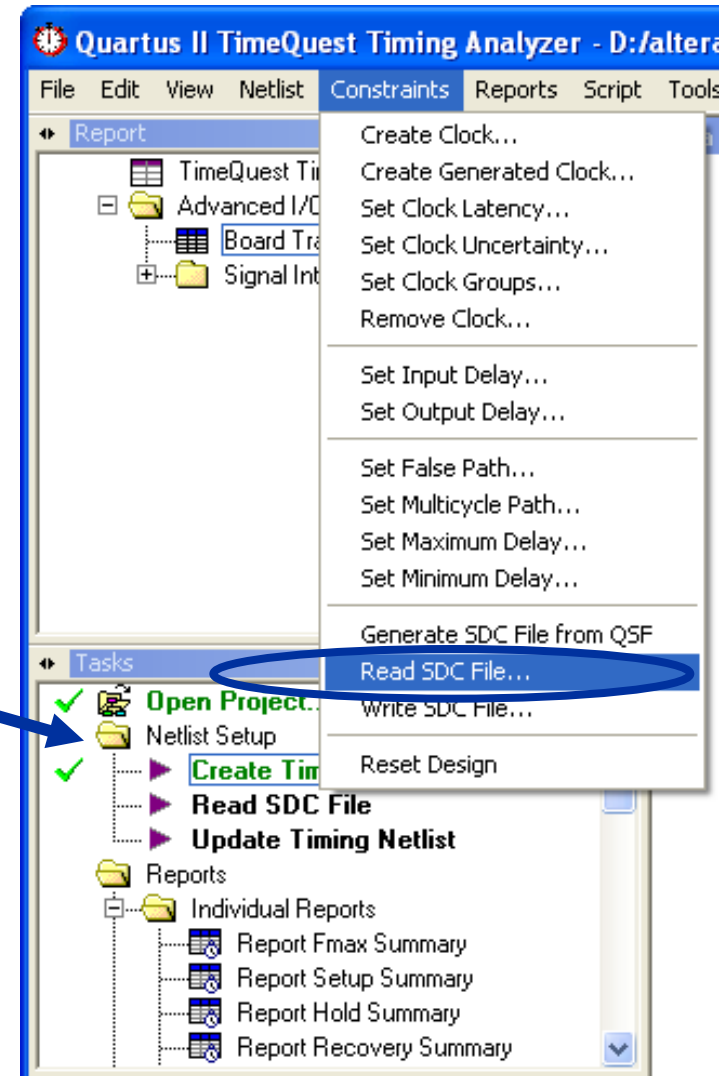


Tcl equivalent of command

Tcl: create_timing_netlist

2a) Create or Read in SDC File

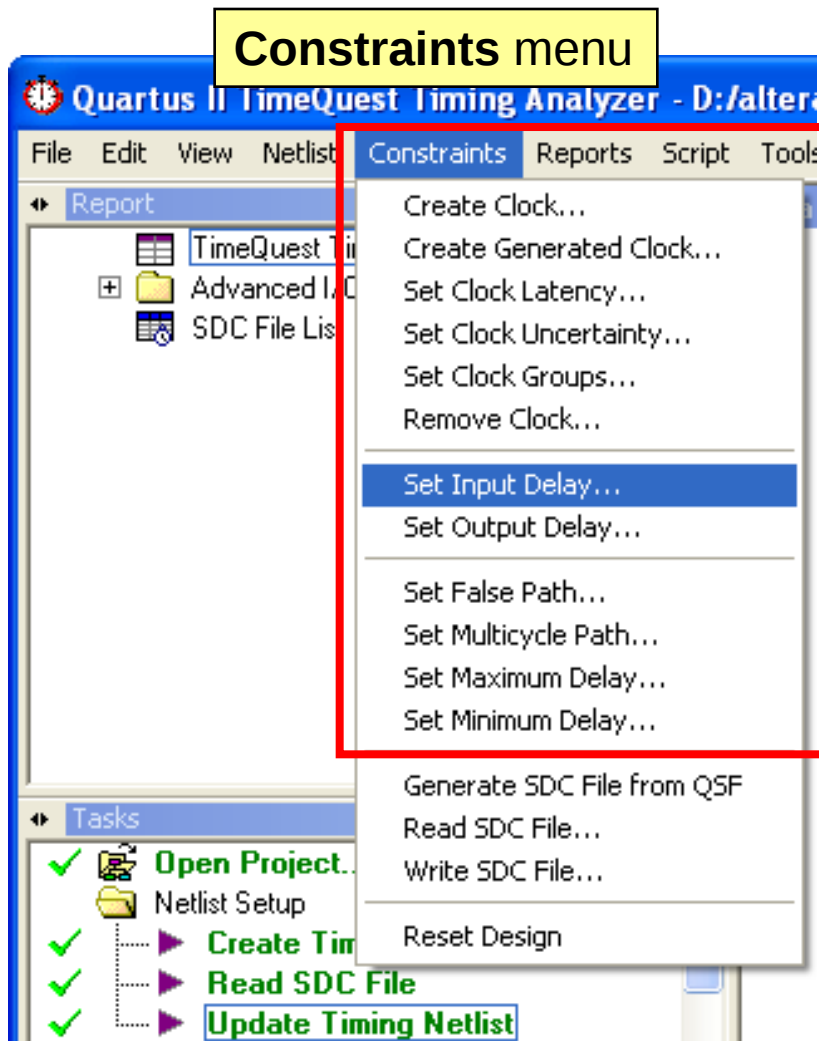
- Create SDC file using SDC file editor
 - Don't enter constraints using **Constraints** menu
- Read in constraints & exceptions from existing SDC file and/or HDL
 - **-hdl**: looks for **ALTERA_ATTRIBUTE** embedded in HDL first before reading SDC files
- Execution
 - **Read SDC File** (Tasks pane or **Constraints** menu)
- File precedence (if no filename specified)
 - Files specifically added to Quartus II project
 - `<current_revision>.sdc` (if it exists in project directory)



2b) Constrain Directly in Console

- Apply new constraints directly to netlist with console SDC commands or from the **Constraints** menu
 - Not automatically added to SDC file
 - Not needed if all constraints in SDC file
- Use **remove_<command>** to remove applied constraints
 - Only **remove_clock** is in GUI
- *Recommend using SDC file (step 2a) instead to ease management and storage of constraints*

Using GUI to Enter Constraints Directly



- Most common constraints can be accessed from the **Constraints** menu
- Same as **Edit** menu ⇒ **Insert Constraints** in SDC file editor
- Use if unfamiliar with SDC syntax

Constraining

- User **MUST** enter constraints for all paths to fully analyze design
 - Timing analyzer only performs slack analysis on *constrained* design paths
 - Constraints guide the fitter to place & route design in order to meet timing requirements
 - Recommendation: Constrain ***all*** paths (at least clocks & I/O)
- Not as difficult a task as it may sound
 - Wildcards
 - Single, generalized constraints cover many paths, even all paths in an entire clock domain

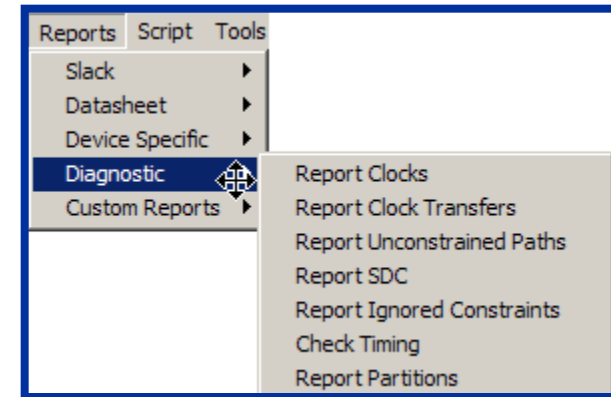
3) Update Timing Netlist

- Apply SDC constraints/exceptions to current timing netlist
- Generates warnings
 - Undefined clocks
 - Partially defined I/O delays
 - Combinational loops
- Update timing netlist after adding any new constraint
- Execution
 - **Update Timing Netlist** (**Tasks** pane or **Netlist** menu)

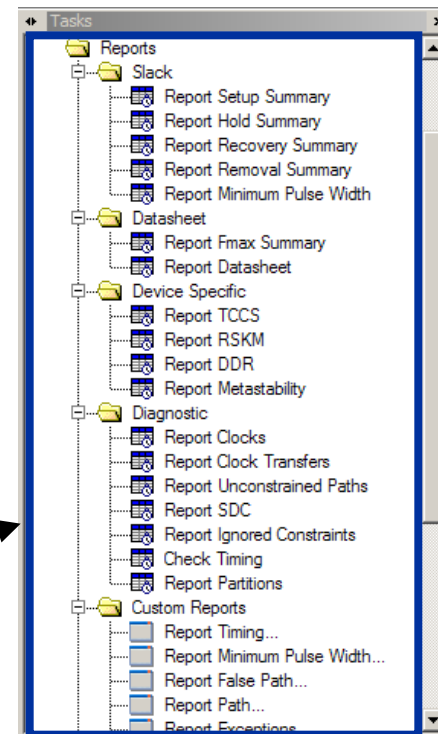
Tcl: update_timing_netlist

4) Generate Timing Reports

- Verify timing requirements and locate violations
- Check for fully constrained design or ignored timing constraints
- Two methods
 - **Tasks** pane
 - *Shortcut:* Automatically creates/updates netlist & reads default SDC file if needed
 - **Reports** menu
 - Must have valid netlist to access



**Double-click to
create individual
report**



“Out of Date” Reports

- Adding new constraints interactively in console causes current reports to be “out of date”
- Update timing netlist & regenerate reports (Report pane right-click menu)
- No such warning when using SDC file

The screenshot shows the 'Report' pane in Quartus II. The 'Clocks Summary' report is selected, and its right-click context menu is open. The menu options are: 'Export...', 'Regenerate', 'Regenerate All Out of Date' (highlighted), 'Delete', 'Delete All Out of Date', and 'Delete All'. The report table shows the following data:

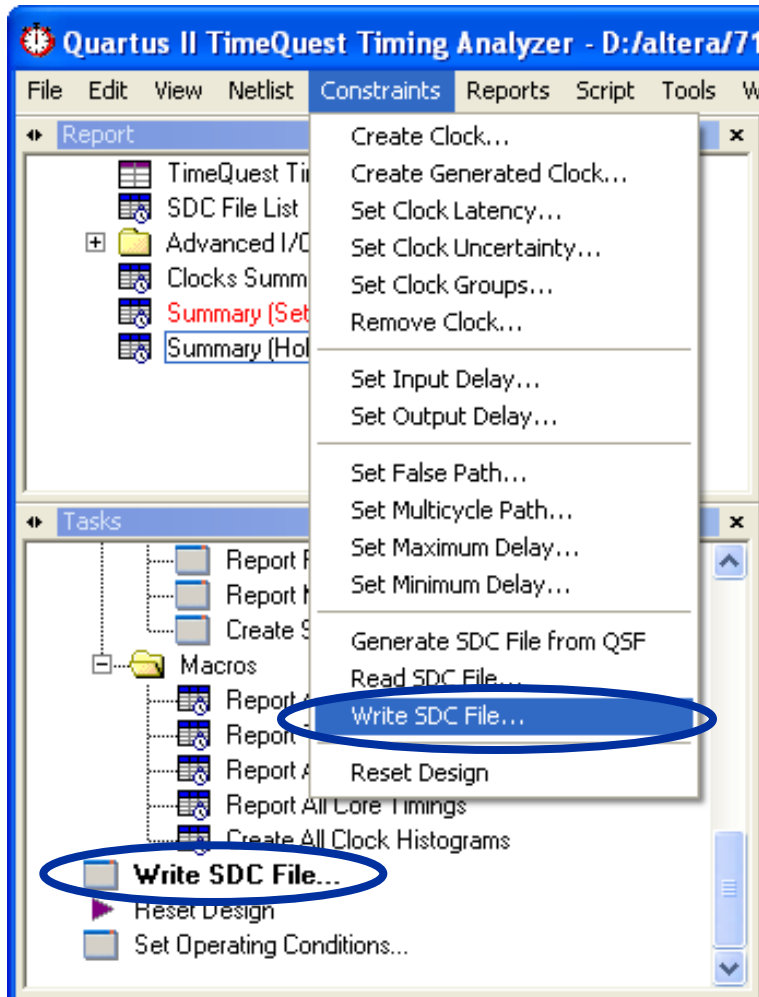
	Clock Name	Type	Period	Rise	Fall	Duty Cycle	Divide by	Multiply by	Phase
1	clk_cons	Base	10.000	0.000	5.000				
2	clkx2_cons	Base	10.000	0.000	5.000				



Reset Design Command

- Tasks pane or **Constraints** menu
- Flushes all timing constraints from current timing netlist
 - Functional Tcl equivalent: `delete_timing_netlist` command followed by `create_timing_netlist`
- Uses
 - “Re-starting” timing analysis on same timing netlist applying different constraints or SDC file
 - Starting analysis over if results seem to be unexpected

5) Save Timing Constraints (Optional)



■ write_sdc command

- Saves all constraints & exceptions applied to current netlist into SDC file
- Use if constraints added during TimeQuest session in console instead of SDC file

■ Notes

- SDC files generated by TimeQuest TA only if requested
- Use -expand option (*not in GUI*) to convert Altera-specific SDC commands (discussed later) into standard SDC
- Run report_t_sdc command (console, Tasks pane, or **Report** menu) to see what will get written to SDC file

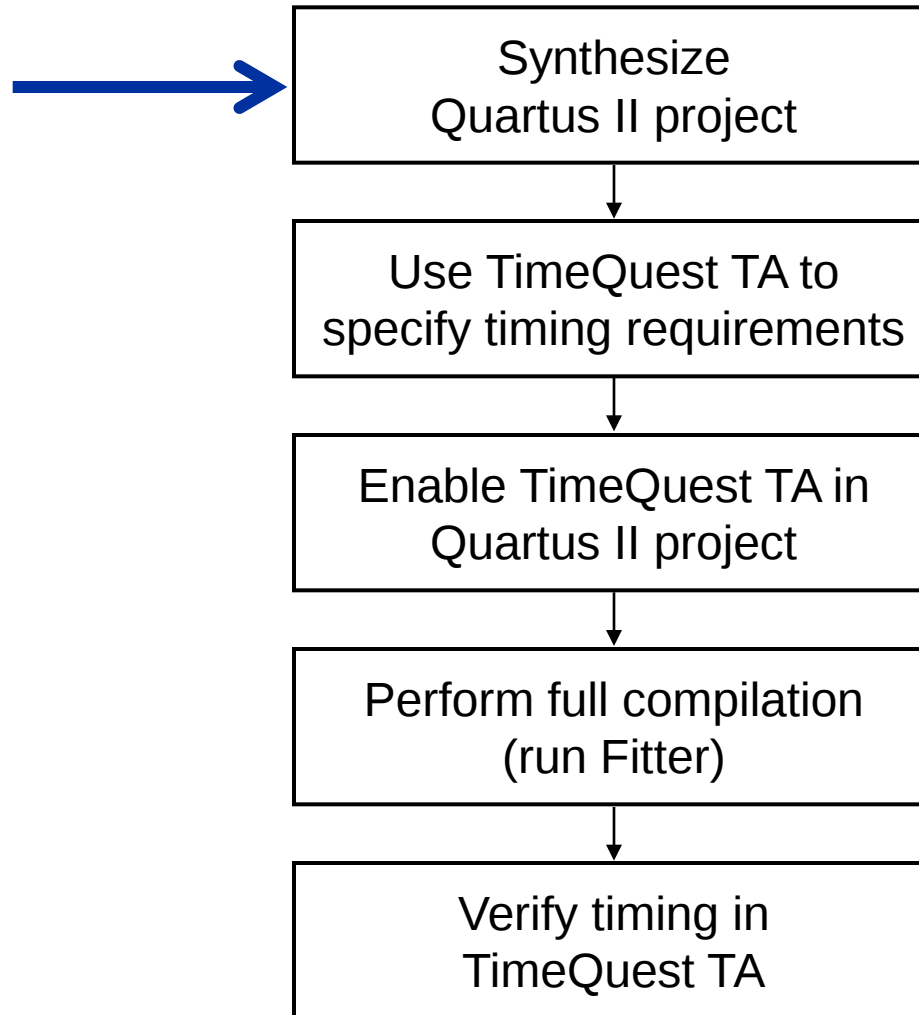
Basic Steps to Using TimeQuest TA (Review)

1. Generate timing netlist
2. Enter SDC constraints
 - a. Create and/or read in SDC file (recommended method)

or

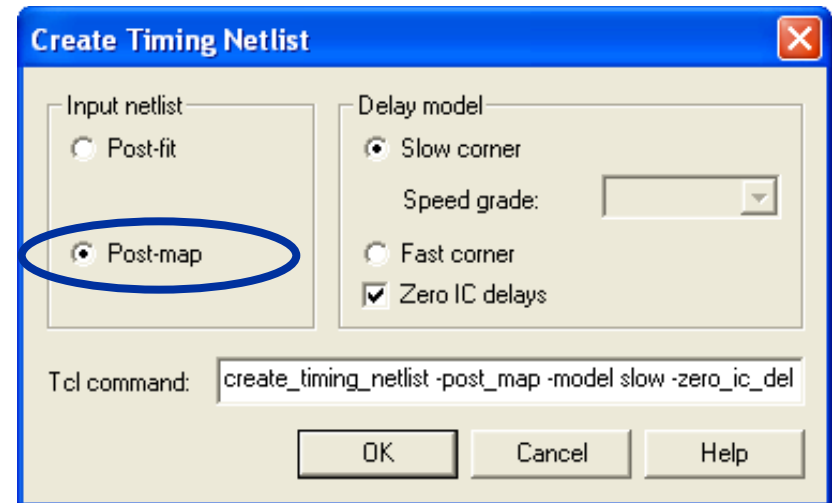
 - b. Constrain design directly in console
3. Update timing netlist
4. Generate timing reports
5. Save timing constraints (optional)

Using TimeQuest TA in Quartus II Flow

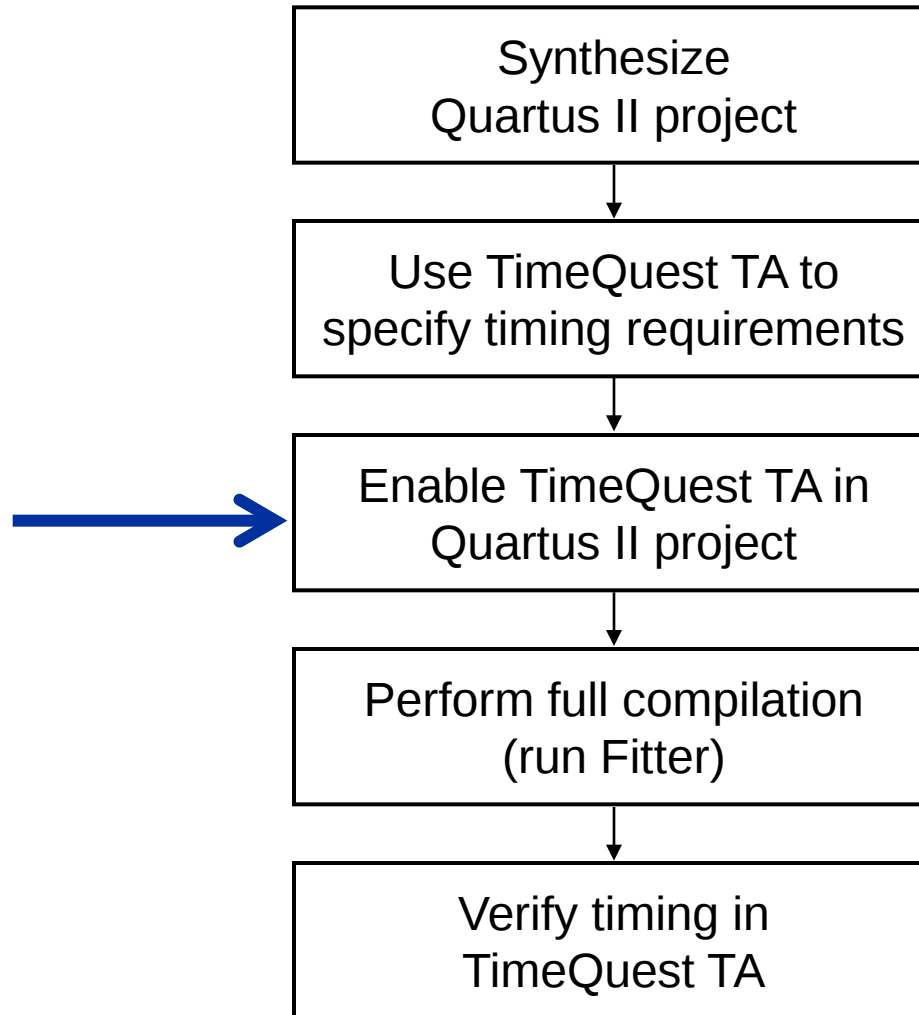


Timing Requirements: Create Post-Map Netlist

- Follow TimeQuest flow
- Use `-post_map` argument for synthesis (mapping) only netlist
 - If design already fully compiled, choose `-post_fit` (default)
- Tasks list command defaults to post-fit, so must use **Netlist** menu in GUI
- **Zero IC delays** auto-enabled with **Post-map**
 - Assumes no interconnect delays to determine if it will be possible to meet timing



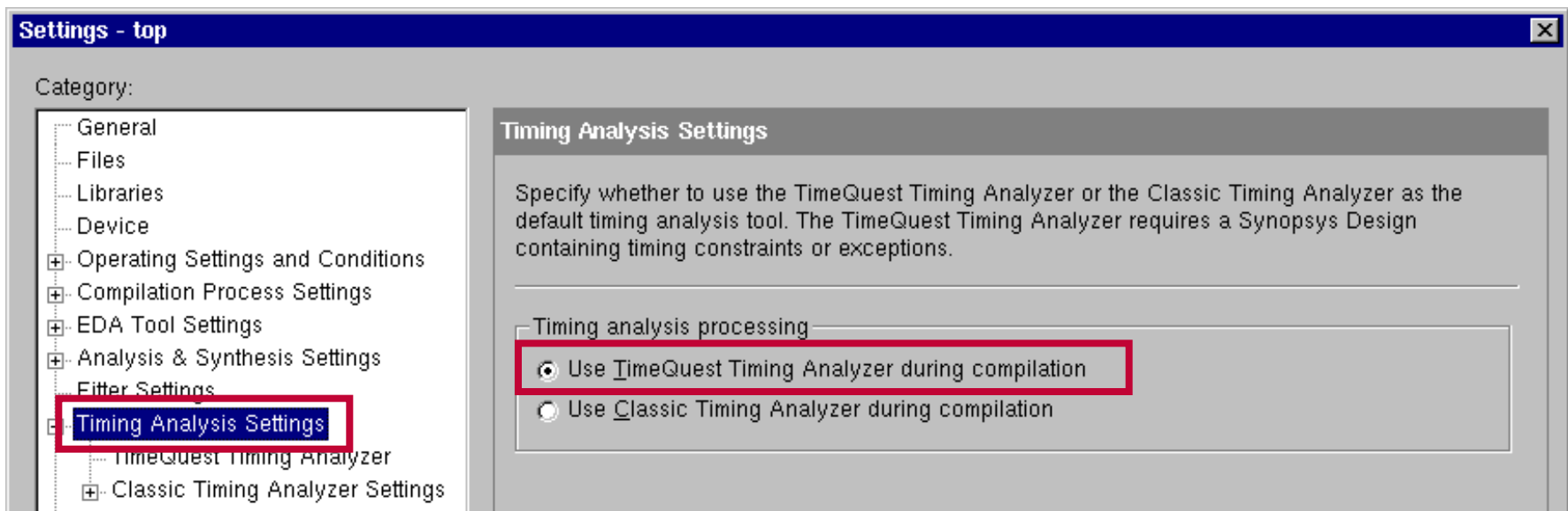
Using TimeQuest TA in Quartus II Flow



Enable TimeQuest TA in Quartus II Software

- Tells the Quartus II software to use SDC constraints during fitting
- File order precedence
 1. Any SDC files manually added to Quartus II project (in order)
 2. *<current_revision>*.SDC located in project directory

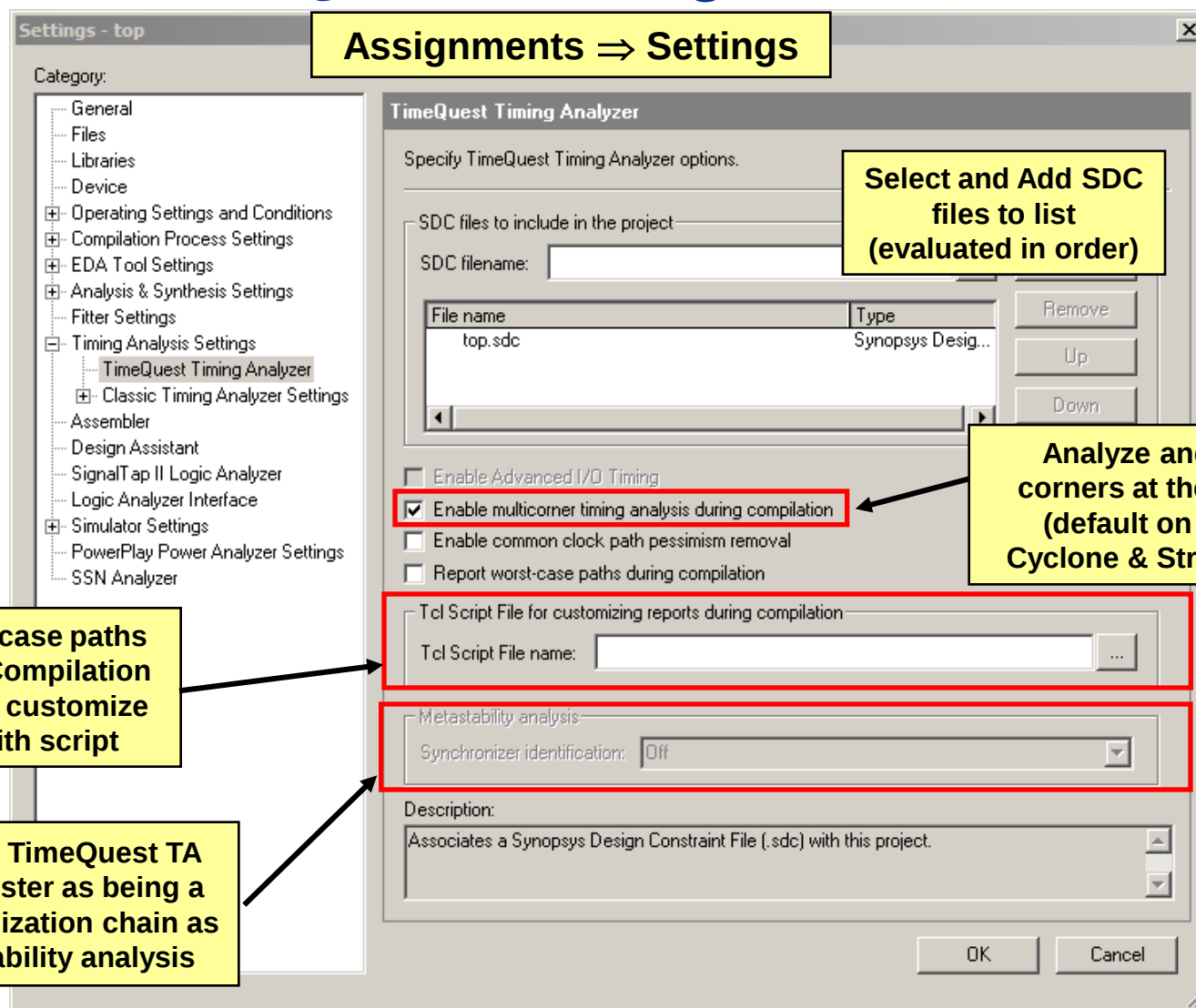
Enabling TimeQuest in Quartus II Software



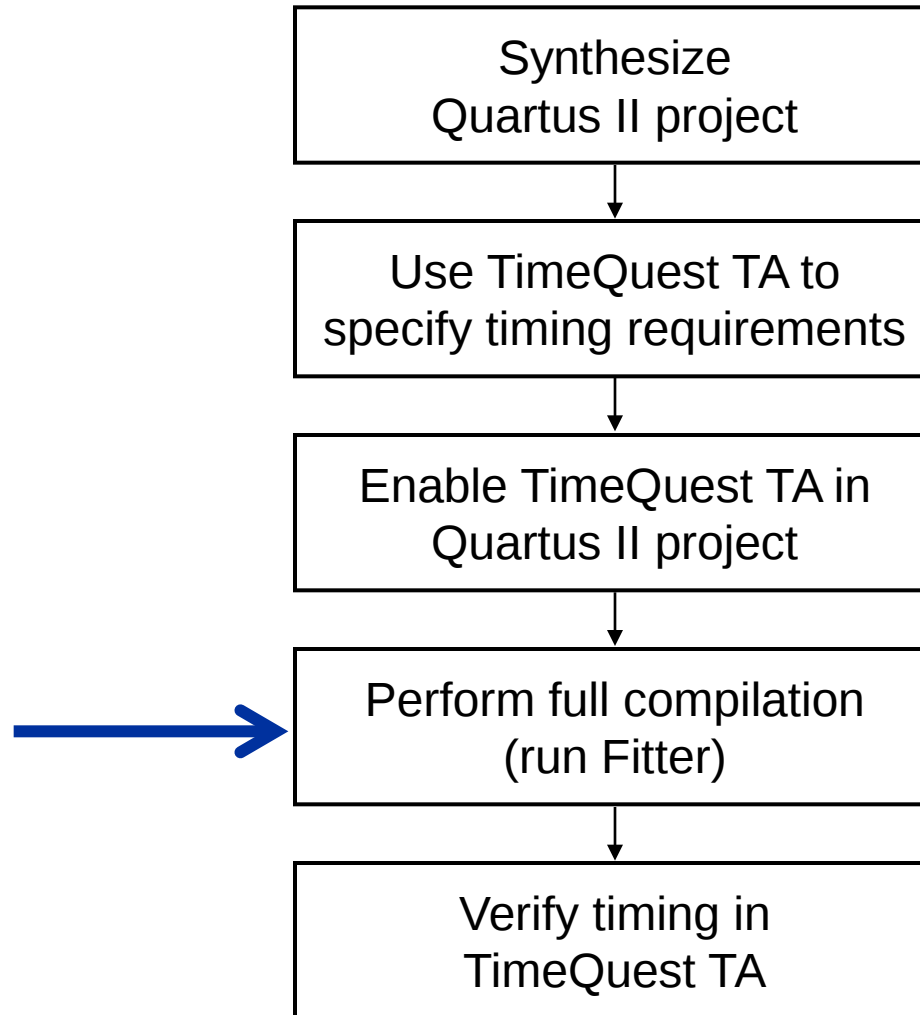
Notes:

- Arria® GX and newer devices *only* support Timequest TA.
- TimeQuest TA is enabled by default for new Stratix® III and Cyclone® III designs.

Quartus II TimeQuest Settings



Using TimeQuest TA in Quartus II Flow



Verifying Timing Requirements

- View TimeQuest summary information directly in Quartus II Compilation Report
- Open TimeQuest TA for more thorough analysis
 - Follow TimeQuest flow, selecting **Post-fit** netlist
 - Optional: Enable **Zero IC Delays** to see if there is any chance of meeting timing without having to enable optimization options
 - Run TimeQuest easy-to-use reporting capabilities (**Tasks** pane)
 - Many different reporting options available
 - Place Tcl reporting commands into script file
 - Easy repetition
- ***Verify whether Fitter was able to meet timing requirements***

TimeQuest Reports in Compilation Report

The screenshot shows the Quartus II interface with the 'Compilation Report - Slow 1200mV 85C Model Setup S...' window open. The left pane displays a tree view of the report structure. Two red boxes highlight specific sections: the top box covers the 'TimeQuest Timing Analyzer' folder and its sub-items, and the bottom box covers the 'TimeQuest Timing Analyzer C' folder and its sub-items. The right pane shows the 'Slow 1200mV 85C Model Setup Summary' table.

	Clock	Slack	End Point	TNS
1	clk_x2	0.295	0.000	
2	clk_x1	0.846	0.000	
3	clk_out	2.256	0.000	

Reports generated during compile (as specified in the Settings window). For example:

- SDC files used during fitting
- Clocks generated
- Timing violations
- Unconstrained paths
- Worst-case paths

Reports generated in TimeQuest TA GUI



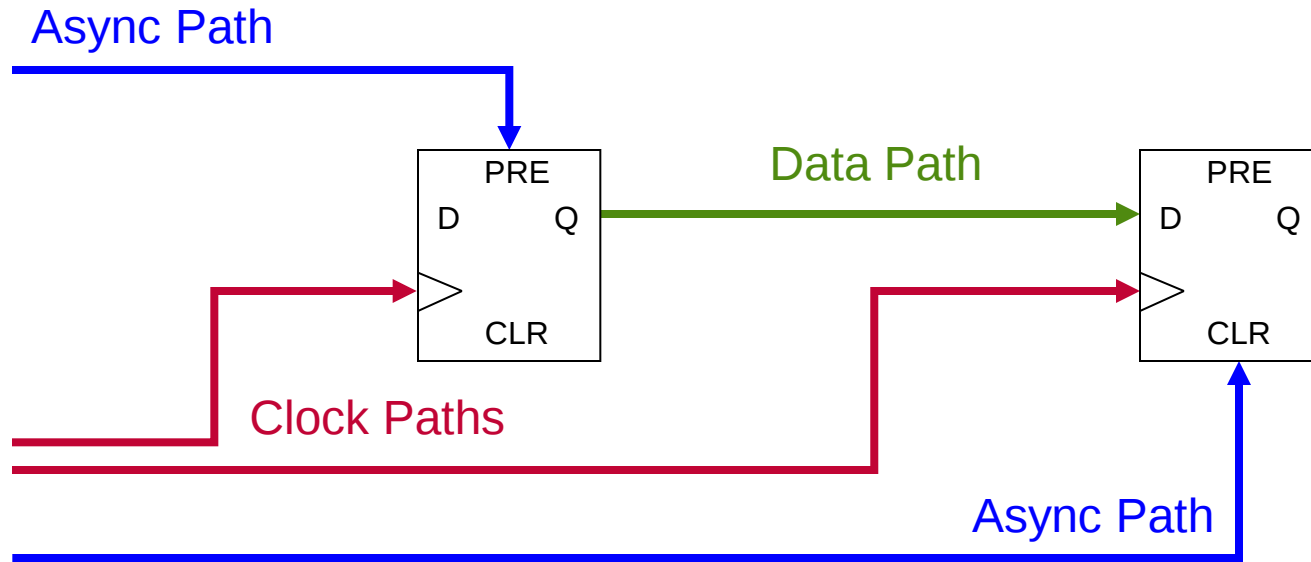
Quartus® II Software Design Series: Timing Analysis

Timing Analysis Basics

Timing Analysis Basics

- Launch vs. latch edges
- Setup & hold times
- Data & clock arrival time
- Data required time
- Setup & hold slack analysis
- I/O analysis
- Recovery & removal
- Timing models

Path & Analysis Types



Three types of Paths:

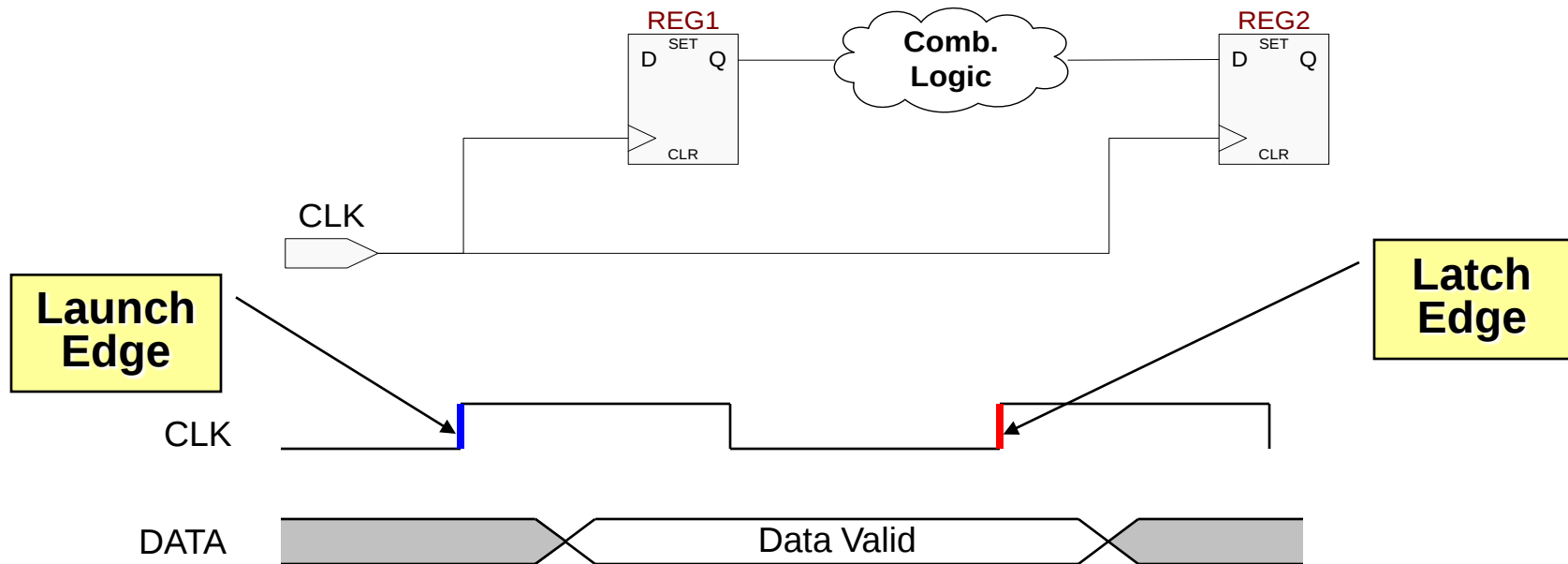
1. Clock Paths
2. Data Path
3. Asynchronous Paths*

Two types of Analysis:

1. Synchronous – clock & data paths
2. Asynchronous* – clock & async paths

**Asynchronous refers to signals feeding the asynchronous control ports of the registers*

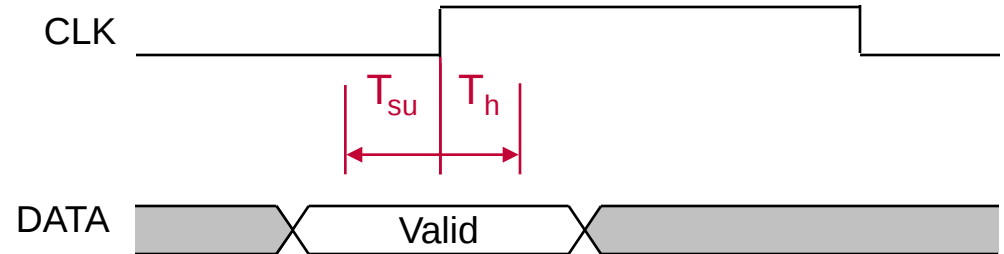
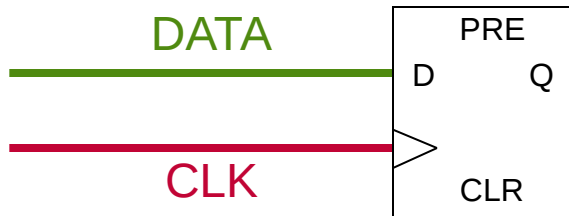
Launch & Latch Edges



Launch Edge: the edge which “launches” the data from source register

Latch Edge: the edge which “latches” the data at destination register (with respect to the launch edge, selected by timing analyzer; typically 1 cycle)

Setup & Hold



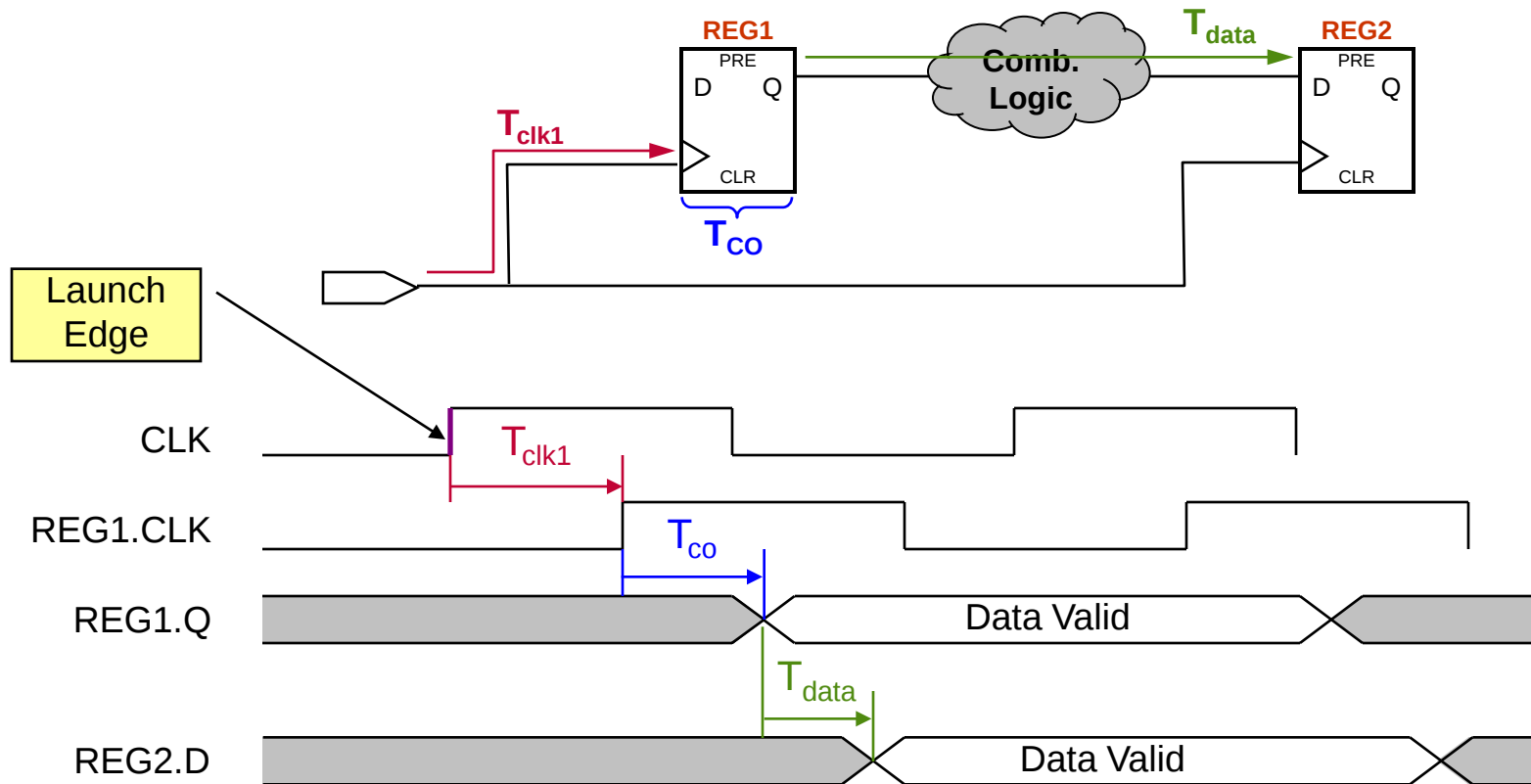
Setup: The minimum time data signal must be stable BEFORE clock edge

Hold: The minimum time data signal must be stable AFTER clock edge

Together, the setup time and hold time form a Data Required Window, the time around a clock edge in which data must be stable.

Data Arrival Time

- The time for data to arrive at destination register's D input



$$\text{Data Arrival Time} = \text{launch edge} + T_{clk1} + T_{co} + T_{data}$$

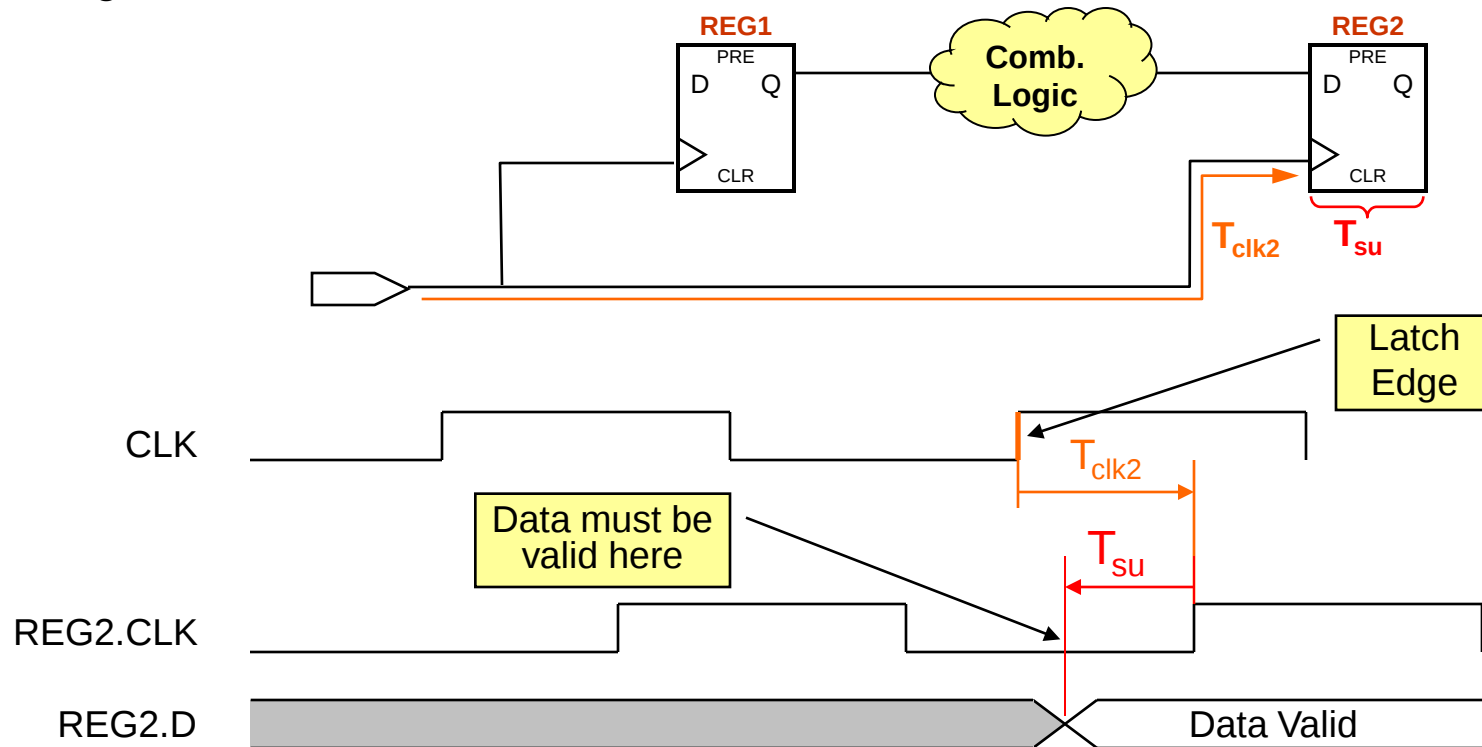


-
- The diagram illustrates a two-stage register with a combinatorial logic block between them. The first register, REG1, has a clock input (CLK) and a clear input (CLR). The output of REG1 (Q1) is connected to the input of the combinatorial logic block (Comb. Logic). The output of the combinatorial logic block is connected to the input of the second register (REG2). The clock signal for REG2 (REG2.CLK) is derived from the output of the combinatorial logic block. The diagram shows the timing relationship between the clock signal (CLK) and the clock signal for the second register (REG2.CLK). The time from the clock edge to the data input is labeled T_{clk2} . A yellow box labeled "Latch Edge" points to the clock edge of REG2.

Clock Arrival Time = latch edge + T_{clk2}

Data Required Time - Setup

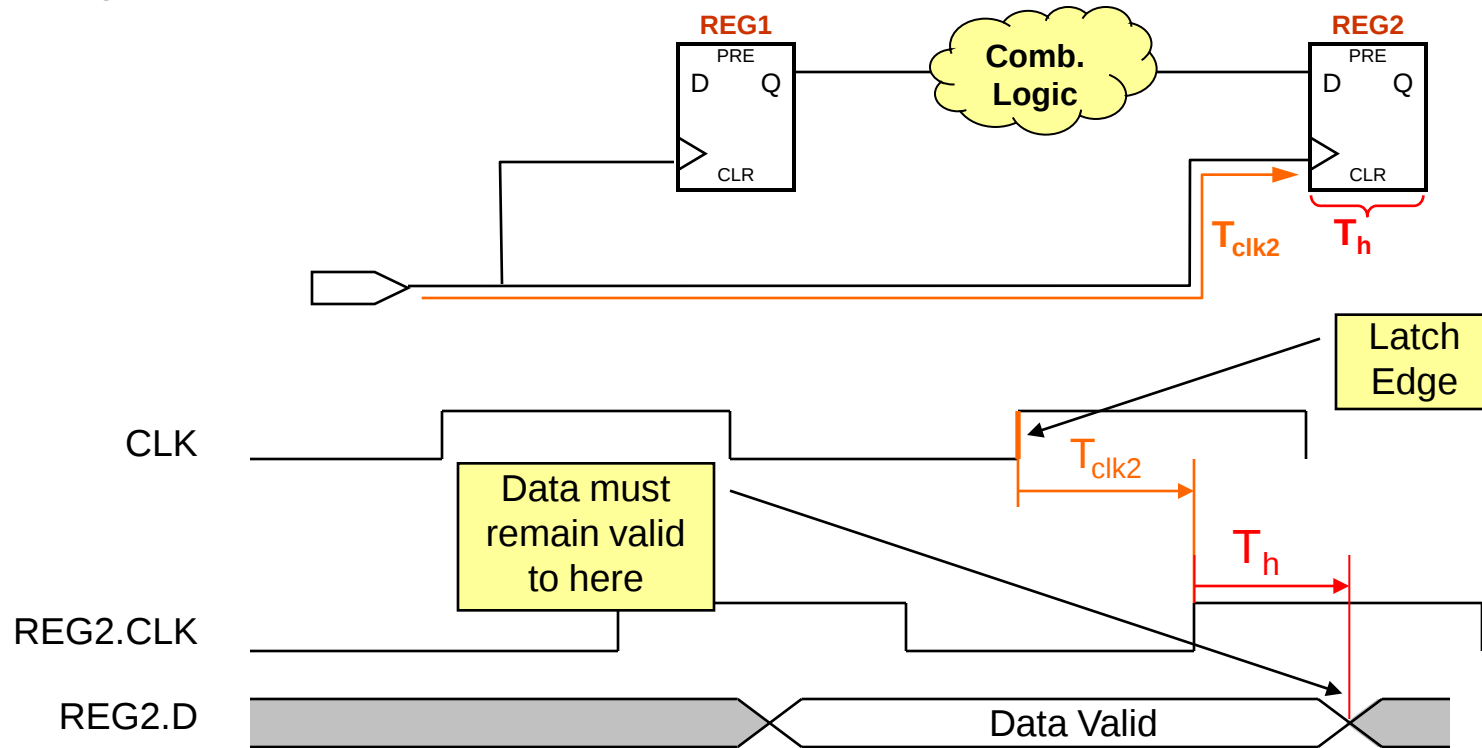
- The minimum time required for the data to get latched into the destination register



$$\text{Data Required Time} = \text{Clock Arrival Time} - T_{su} - \text{Setup Uncertainty}$$

Data Required Time - Hold

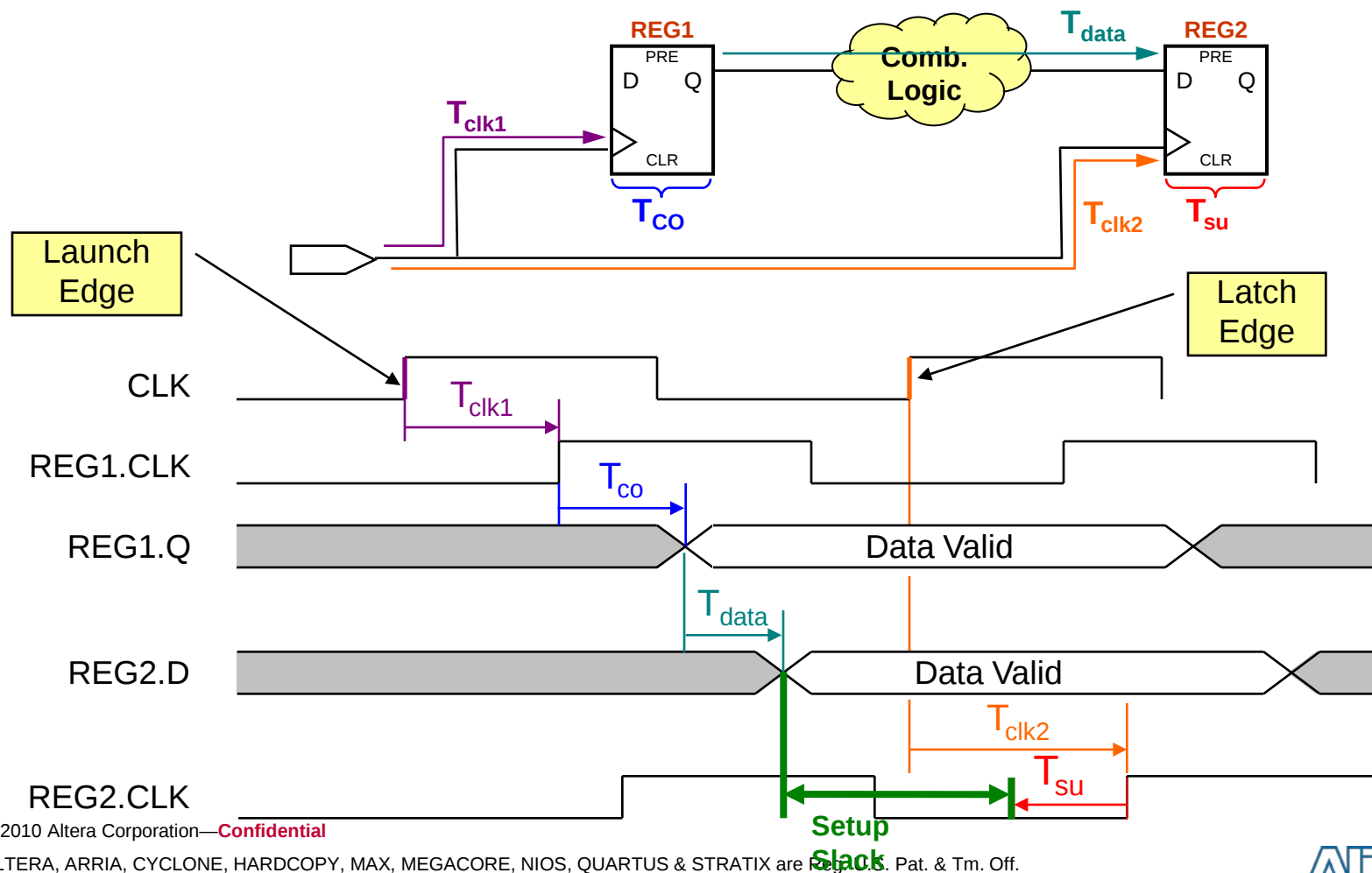
- The minimum time required for the data to get latched into the destination register



$$\text{Data Required Time} = \text{Clock Arrival Time} + T_h + \text{Hold Uncertainty}$$

Setup Slack

- The margin by which the setup timing requirement is met. It ensures launched data arrives in time to meet the latching requirement.



© 2010 Altera Corporation—Confidential

ALTERA, ARRIA, CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS & STRATIX are registered trademarks of Altera Corporation. Pat. & Tm. Off.
and Altera marks in and outside the U.S.



Setup Slack (cont'd)

$$\text{Setup Slack} = \text{Data Required Time (Setup)} \\ - \text{Data Arrival Time}$$

Positive slack

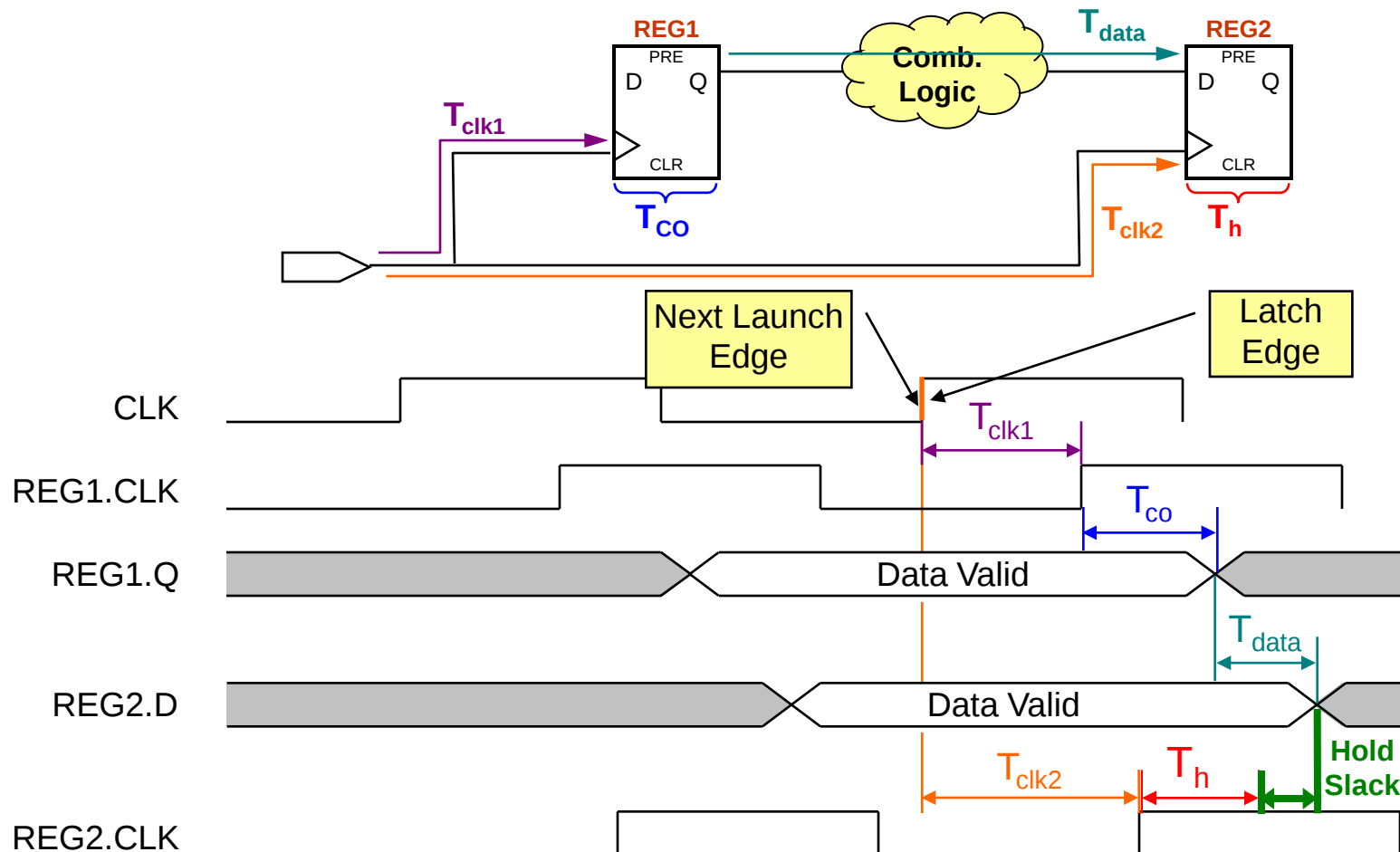
- Timing requirement met

Negative slack

- Timing requirement not met

Hold Slack

- The margin by which the hold timing requirement is met. It ensures latch data is not corrupted by data from another launch edge. It also prevents “double-clocking”.



© 2010 Altera Corporation—Confidential

ALTERA, ARRIA, CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS & STRATIX are Reg. U.S. Pat. & Tm. Off.
and Altera marks in and outside the U.S.



Hold Slack (cont'd)

$$\text{Hold Slack} = \text{Data Arrival Time} \\ - \text{Data Required Time (Hold)}$$

Positive slack

- Timing requirement met

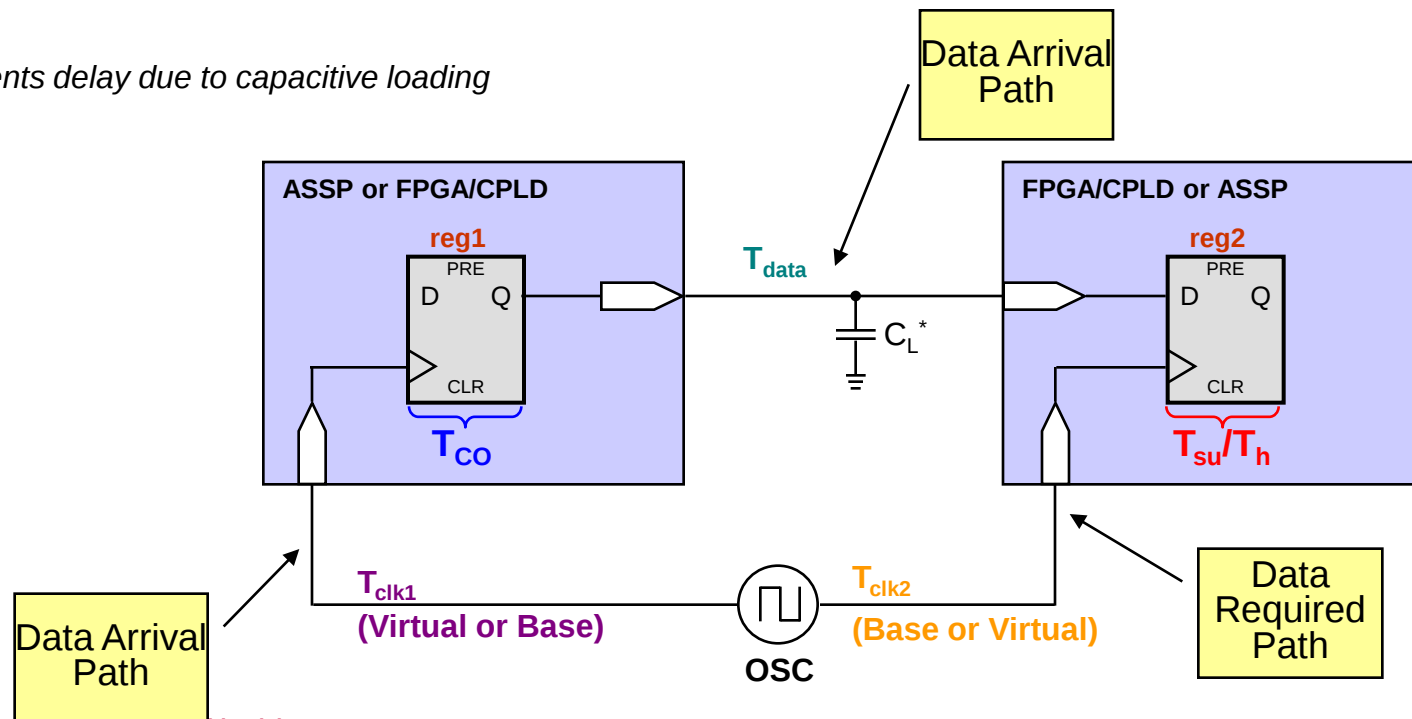
Negative slack

- Timing requirement not met

I/O Analysis (Common Clock Source)

- Analyzing I/O performance in a synchronous design uses the same slack equations
 - Must include external device & PCB timing parameters
 - Recommend use virtual clock for specifying input/output delays
 - Otherwise, can be difficult to accurately constrain I/Os

* Represents delay due to capacitive loading

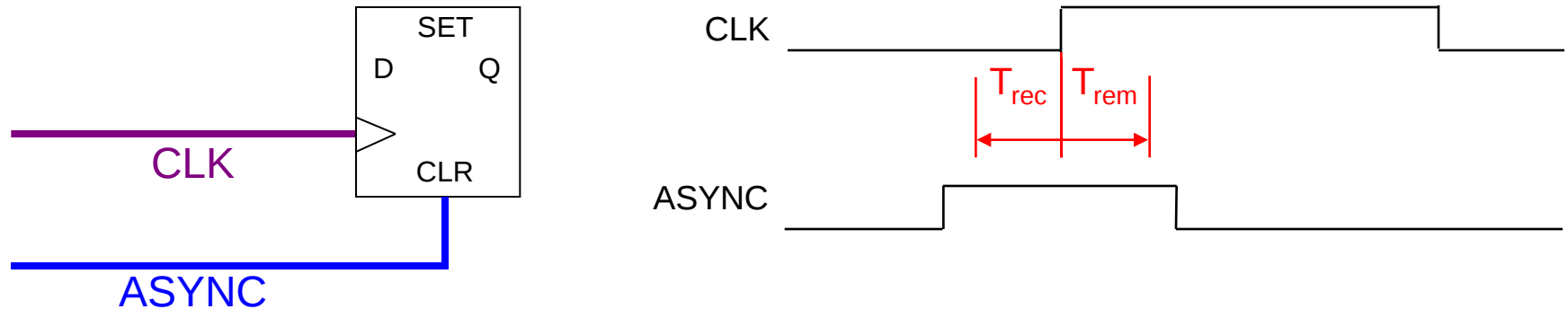


© 2010 Altera Corporation—Confidential

ALTERA, ARRIA, CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS & STRATIX are Reg. U.S. Pat. & Tm. Off. and Altera marks in and outside the U.S.



Recovery & Removal



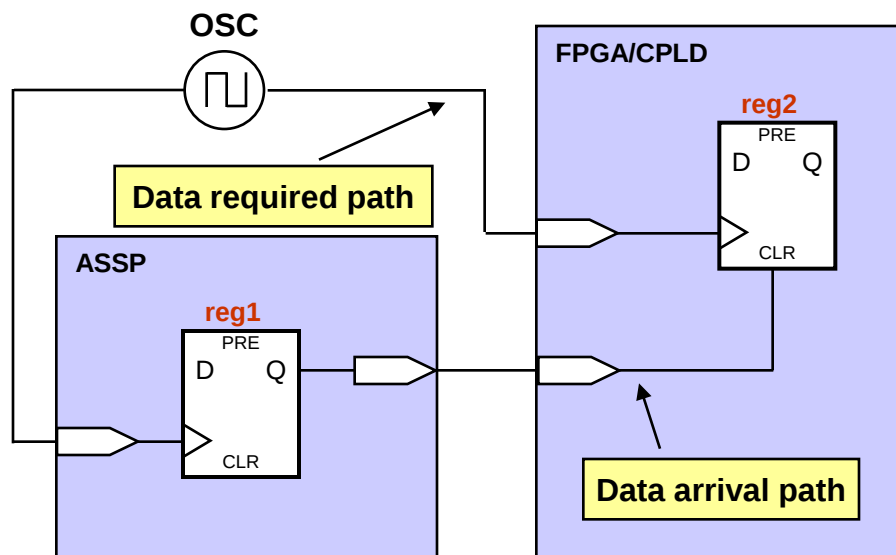
Recovery: The minimum time an asynchronous signal can be de-asserted BEFORE clock edge

Removal: The minimum time an asynchronous signal can be de-asserted AFTER clock edge

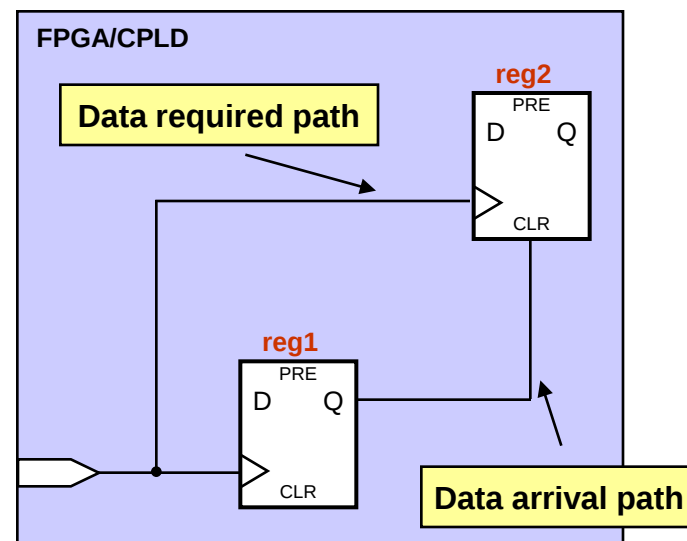
Asynchronous = Synchronous?

■ Asynchronous control signal source is assumed synchronous

- Slack equations still apply
 - data arrival path = asynchronous control path
 - $T_{su} \approx T_{rec}$; $T_h \approx T_{rem}$
- External device & board timing parameters may be needed (Ex. 1)



Example 1



Example 2

Why Are These Calculations Important?

- Calculations are important when timing violations occur
 - Need to be able to understand cause of violation
- Example causes
 - Data path too long
 - Requirement too short (incorrect analysis)
 - Large clock skew signifying a gated clock, etc.
- TimeQuest timing analyzer uses them
 - Equations to calculate slack
 - Terminology (launch and latch edges, Data Arrival Path, Data Required Path, etc.) in timing reports

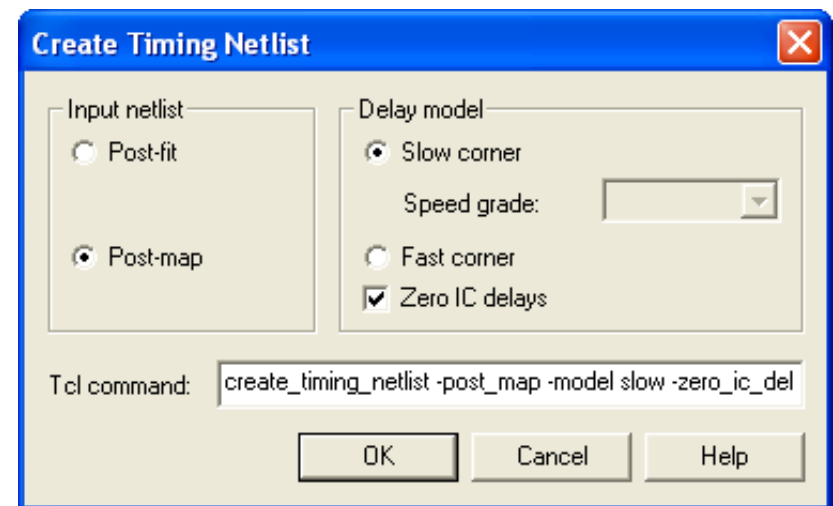


Timing Models in Detail

- Quartus II software models device timing at two PVT conditions by default
 - **Slow Corner** Model
 - Indicates slowest possible performance for any single path
 - Timing for slowest device at maximum operating temperature and VCC_{MIN}
 - **Fast Corner** Model
 - Indicates fastest possible performance for any single path
 - Timing for fastest device at minimum operating temperature and VCC_{MAX}
- Why two corner timing models?
 - Ensure **setup** timing is met in **slow** model
 - Ensure **hold** timing is met in **fast** model
 - Essential for source synchronous interfaces
- Third model (slow, min. temp.) available only for 65 nm and smaller technology devices (temperature inversion phenomenon)

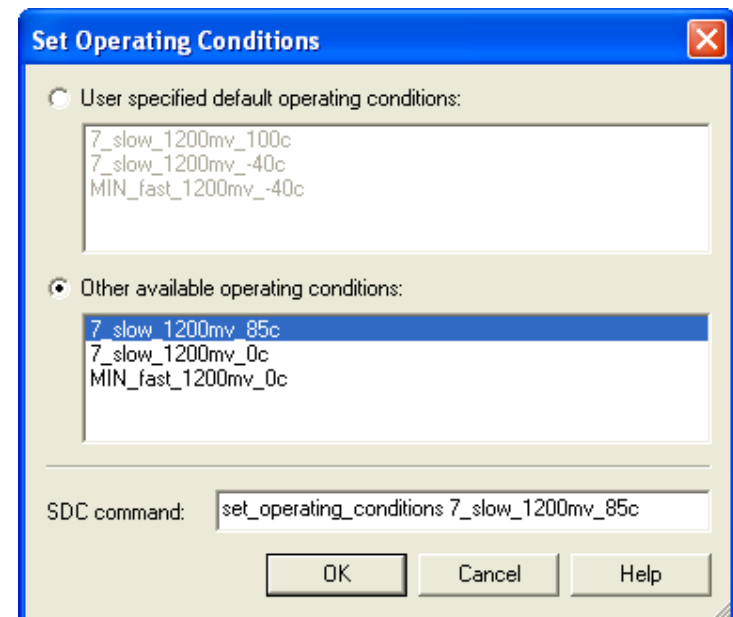
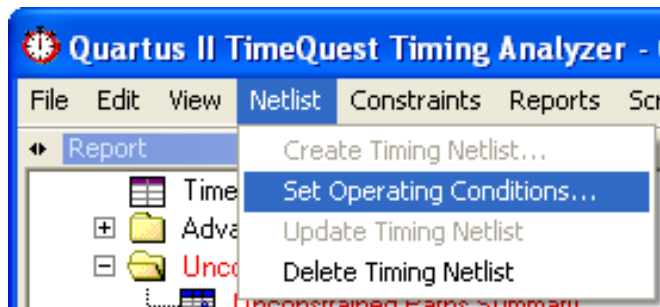
Generating Fast/Slow Netlist

- Specify one of the default timing models to be used when creating your netlist
- Default is the slow timing netlist
- To specify fast timing netlist
 - Use `-fast_model` option with `create_timing_netlist` command
 - Choose **Fast corner** in GUI when executing **Create Timing Netlist** from **Netlist** menu
 - CANNOT select fast corner from Tasks Pane



Specifying Operating Conditions

- Perform timing analysis for different delay models without recreating the existing timing netlist
- Takes precedence over already generated netlist
- Required for selecting slow, min. temp. model and other models (industrial, military, etc.) depending on device
- Use `get_available_operating_conditions` to see available conditions for target device



Please go to Exercise 1



Quartus® II Software Design Series: Timing Analysis

Timing Reports

Timing Reports

- Timing results available in both the Quartus II Compilation Report and TimeQuest GUI
- TimeQuest TA includes more extensive reporting capabilities
- Create reports while creating constraints (**post-map** netlist) before fitting to see if design can meet timing requirements
- Create reports after fitting (**post-fit** netlist) to verify that placed & routed design meets timing requirements

Summary Reports

- Simplest, most common type of timing report
- Each row reports on a clock domain in the design
 - Worst case (positive or **negative** slack) listed first
 - If negative, total negative slack (TNS) on all edges in clock domain
- Command: `create_timing_summary`
 - `-setup`, `-hold`, `-recovery`, `-removal`: create report for selected analysis type

Report Setup Summary
Report Hold Summary
Report Recovery Summary
Report Removal Summary

Summary (Setup)

	Clock	Slack	End Point TNS
1	c200	1.748	0.000
2	c100	2.010	0.000
3	clkout	3.783	0.000

Copy
Select All
Undo Sort
Create Setup Slack Histogram
Report Timing...

Create advanced reports about selected clock domain

© 2010 Altera Corporation. All rights reserved.

ALTERA, ARRIA, CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS & STRATIX are Reg. U.S. Pat. & Tm. Off. and Altera marks in and outside the U.S.





Detailed Slack/Path Analysis

- Create more specific/detailed reports
 - Ex. Details on a specific clock domain
 - Ex. View timing paths between particular I/O & registers
- Create using Tcl commands or GUI
 - Use GUI to see report immediately
 - Use Tcl file for repeatability

Advanced Reporting: Report Timing

report_timing

- from <source_nodes>
- from_clock <source_clock_names>
- rise_from_clock <source_clock_names>
- fall_from_clock <source_clock_names>
- through <thru_node>
- to <destination_nodes>
- to_clock <destination_clock_names>
- rise_to_clock <destination_clock_names>
- fall_to_clock <destination_clock_names>
- setup|-hold|-recovery|-removal
- detail <summary|path_only|path_and_clock|full_path>
- file <file_name>
- append
- panel_name <report_name>
- stdout
- less_than_slack <slack_limit>
- npaths <#_of_paths_to_display>
- nworst <max_#_of_paths_per_endpoint>
- false_path
- pairs_only
- show_routing

report_timing Arguments

- -setup|hold|recovery|removal are mutually exclusive
 - Default is -setup
- -detail <option> how to report clock path detail
 - path_only: lumps clock network delay together (default option)
 - summary: lists individual path (condense path report)
 - path_and_clock: shows clock network delay in detail
 - full_path: shows clock network in more detail, particularly generated clock
- -npaths: number of paths to report; defaults to 10

Report Timing (GUI)

Choose Report Timing (Reports menu) or double-click on Report Timing (Tasks pane)

Select level of detail

Select where to send output report

Tcl: `report_timing`

Report Timing Dialog Box:

- Clocks:** From clock: `clk_cons`, To clock: `clkx2_cons`
- Targets:** From: `*`, Through: , To: `*mult*`
- Analysis type:** ☒ Setup, ☐ Recovery, ☐ Hold, ☐ Removal
- Paths:** Report number of paths: `32`, Maximum slack limit: ns
- Output:** Detail level: `Path Only`, Report panel name: `Summary`, File name:
- File options:** ☒ Overwrite, ☐ Append
- Tcl command:** `report_timing -from_clock clk_cons -to_clock clkx2_cons -from * -to`

© 2010 Altera Corporation—Confidential

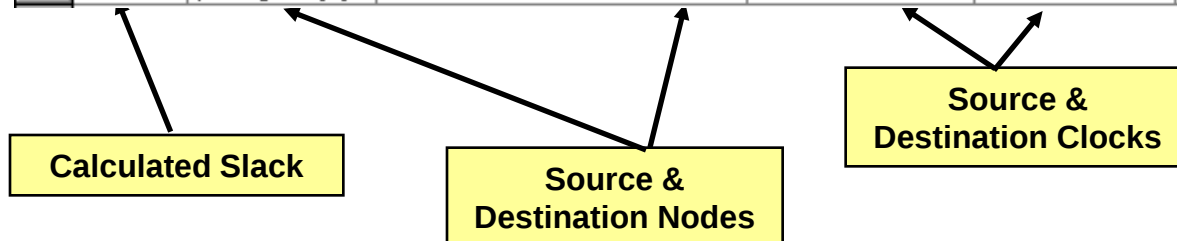
ALTERA, ARRIA, CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS & STRATIX are Reg. U.S. Pat. & Tm. Off. and Altera marks in and outside the U.S.



Summary Slack/Path Report

```
report_timing -from_clock clk_x1 -to_clock clk_x2 \
    -setup -npaths 10 -detail summary \
    -panel_name "Setup (clk_x1 to clk_x2) Summary"
```

	Slack	From Node	To Node	Launch Clock	Latch Clock	Relationship	Clock Skew	Data Delay
1	0.927	a_regtwo[2]	...ABLEDATAA_REGOUT2	clk_x1	clk_x2	3.500	-0.231	2.145
2	1.010	a_regtwo[6]	...ABLEDATAA_REGOUT6	clk_x1	clk_x2	3.500	-0.231	2.062
3	1.015	a_regtwo[3]	...ABLEDATAA_REGOUT3	clk_x1	clk_x2	3.500	-0.231	2.057
4	1.109	a_regtwo[4]	...ABLEDATAA_REGOUT4	clk_x1	clk_x2	3.500	-0.231	1.963
5	1.143	a_regtwo[0]	...ABLEDATAA_REGOUT0	clk_x1	clk_x2	3.500	-0.231	1.929
6	1.148	a_regtwo[1]	...ABLEDATAA_REGOUT1	clk_x1	clk_x2	3.500	-0.231	1.924
7	1.158	a_regtwo[5]	...ABLEDATAA_REGOUT5	clk_x1	clk_x2	3.500	-0.231	1.914
8	1.318	y_regtwo[4]	...ABLEDATAB_REGOUT4	clk_x1	clk_x2	7.000	-0.208	5.277
9	1.347	y_regtwo[6]	...ABLEDATAB_REGOUT6	clk_x1	clk_x2	7.000	-0.208	5.248
10	1.382	y_regtwo[5]	...ABLEDATAB_REGOUT5	clk_x1	clk_x2	7.000	-0.207	5.214



Detailed Slack/Path Report

```
report_timing -from_clock clk_x1 -to_clock clk_x2 \
  -setup -npaths 10 -detail path_only \
  -panel_name "Setup (clk_x1 to clk_x2)"
```

Setup (clk_x1 to clk_x2)

	Slack	From Node	To Node	Launch Clo...	Latch Clock	Relationship	Clock Skew	Data Delay
1	0.927	a_regtwo[2]	...REGOUT2	clk_x1	clk_x2	3.500	-0.231	2.145
2	1.010	a_regtwo[6]	...REGOUT6	clk_x1	clk_x2	3.500	-0.231	2.062
3	1.015	a_regtwo[3]	...REGOUT3	clk_x1	clk_x2	3.500	-0.231	2.057
4	1.109	a_regtwo[4]	...REGOUT4	clk_x1	clk_x2	3.500	-0.231	1.963
5	1.143	a_regtwo[0]	...REGOUT0	clk_x1	clk_x2	3.500	-0.231	1.929
6	1.148	a_regtwo[1]	...REGOUT1	clk_x1	clk_x2	3.500	-0.231	1.924
7	1.158	a_regtwo[5]	...REGOUT5	clk_x1	clk_x2	3.500	-0.231	1.914
8	1.318	y_regtwo[4]	...B_REGOUT4	clk_x1	clk_x2	7.000	-0.208	5.277
9	1.347	y_regtwo[6]	...B_REGOUT6	clk_x1	clk_x2	7.000	-0.208	5.248
10	1.382	y_regtwo[5]	...B_REGOUT5	clk_x1	clk_x2	7.000	-0.207	5.214

Path #1: Setup slack is 0.927

Path Summary | Statistics | Data Path | Waveform

Property	Value
1 From Node	a_regtwo[2]
2 To Node	...mult_component[mult_mfq:auto_g...
3 Launch Clock	clk_x1
4 Latch Clock	clk_x2
5 Data Arrival Time	2.392
6 Data Required Time	3.319
7 Slack	0.927

Calculated slack & path Summary

Path #1: Setup slack is 0.927

Path Summary | Statistics | Data Path | Waveform

Data Arrival Path

	Total	Incr	RF	Type	Fanout
1	0.000	0.000			
2	0.247	0.247			
3	0.247	0.247	R		
4	2.392	2.145			
5	0.446	0.199		uTco	1
6	0.446	0.000	FF	CELL	1
7	-1.519	1.073	FF	IC	1
8	-1.827	0.308	FF	CELL	1
9	-2.187	0.360	FF	IC	1

Data arrival path details

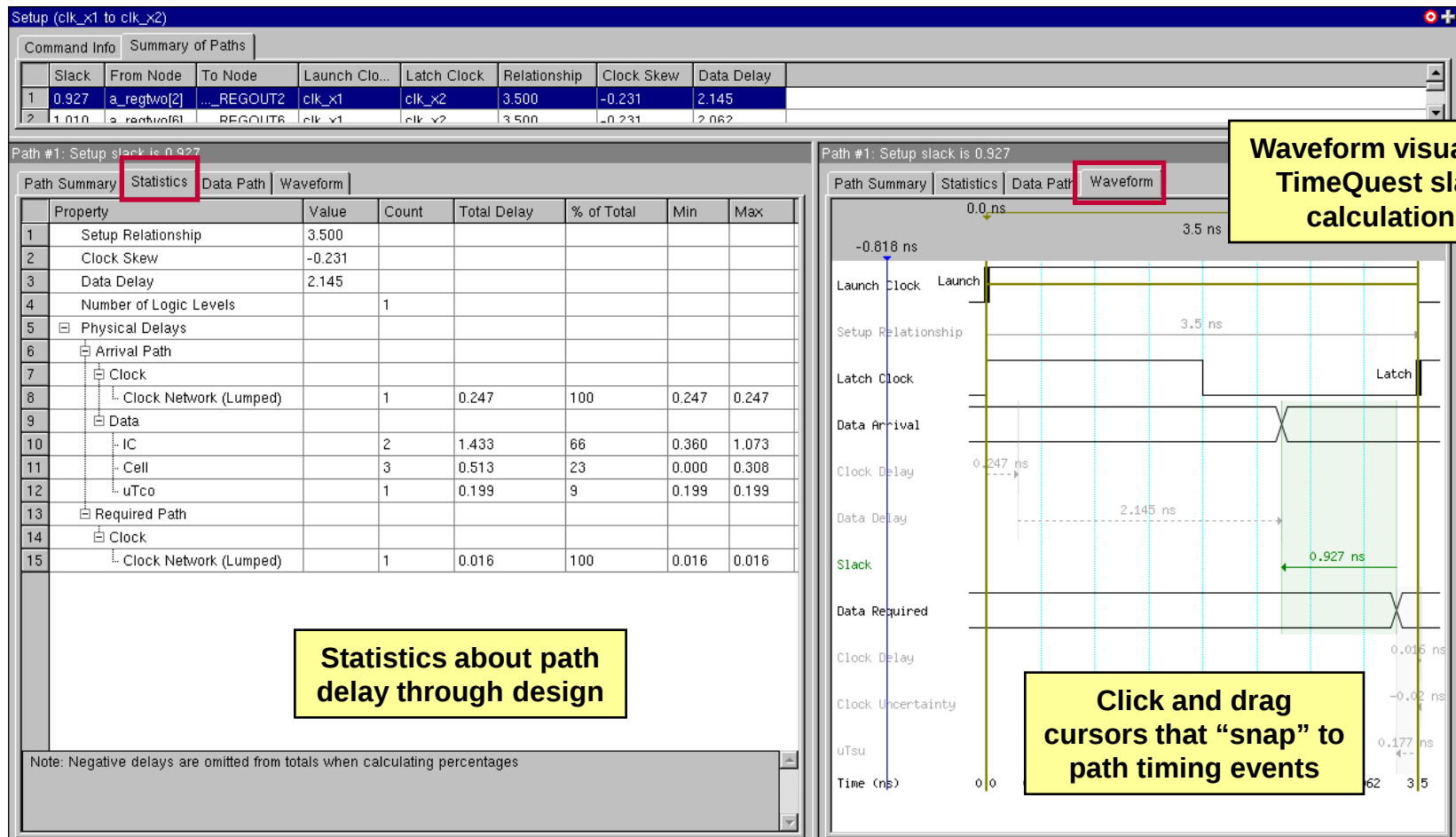
Data Required Path

	Total	Incr	RF	Type	Fanout	Location	Elem...
1	3.500	3.500					...e time
2	3.516	0.016					
3	3.516	0.016	R				
4	3.496	-0.020					
5	3.319	-0.177		uTsu	1	DSPMULT_X20_Y4_N0	...CORE

Data required path details

Detailed Slack/Path Report (cont.)

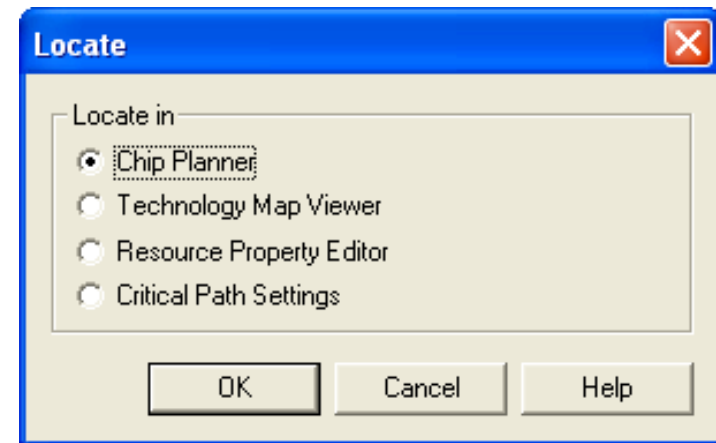
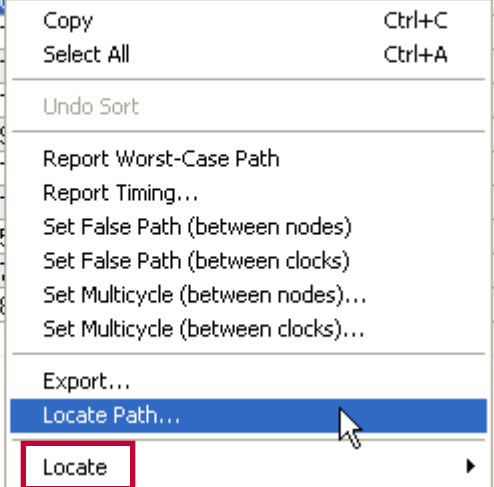
```
report_timing -from_clock c100 -to_clock c200 \
    -setup -npaths 10 -detail path_only \
    -panel_name "Setup (c100 to c200)"
```



Further Path Analysis

- Right-click path(s) to cross-probe to other Quartus II tools or design files
- `locate` command in Console

	Slack	From Node	To Node	Launch Clock	Latch Clock
1	1.045	inst2[0]	-101	-11.1	-11.1
2	1.045	inst2[1]			
3	1.045	inst2[2]			
4	1.045	inst2[3]			
5	1.045	inst2[4]			
6	1.045	inst2[5]			
7	1.045	inst2[6]			
8	1.045	inst2[7]			
9	1.045	inst2[8]			
10	1.045	inst2[9]			





Quartus® II Software Design Series: Timing Analysis

Timing Constraints



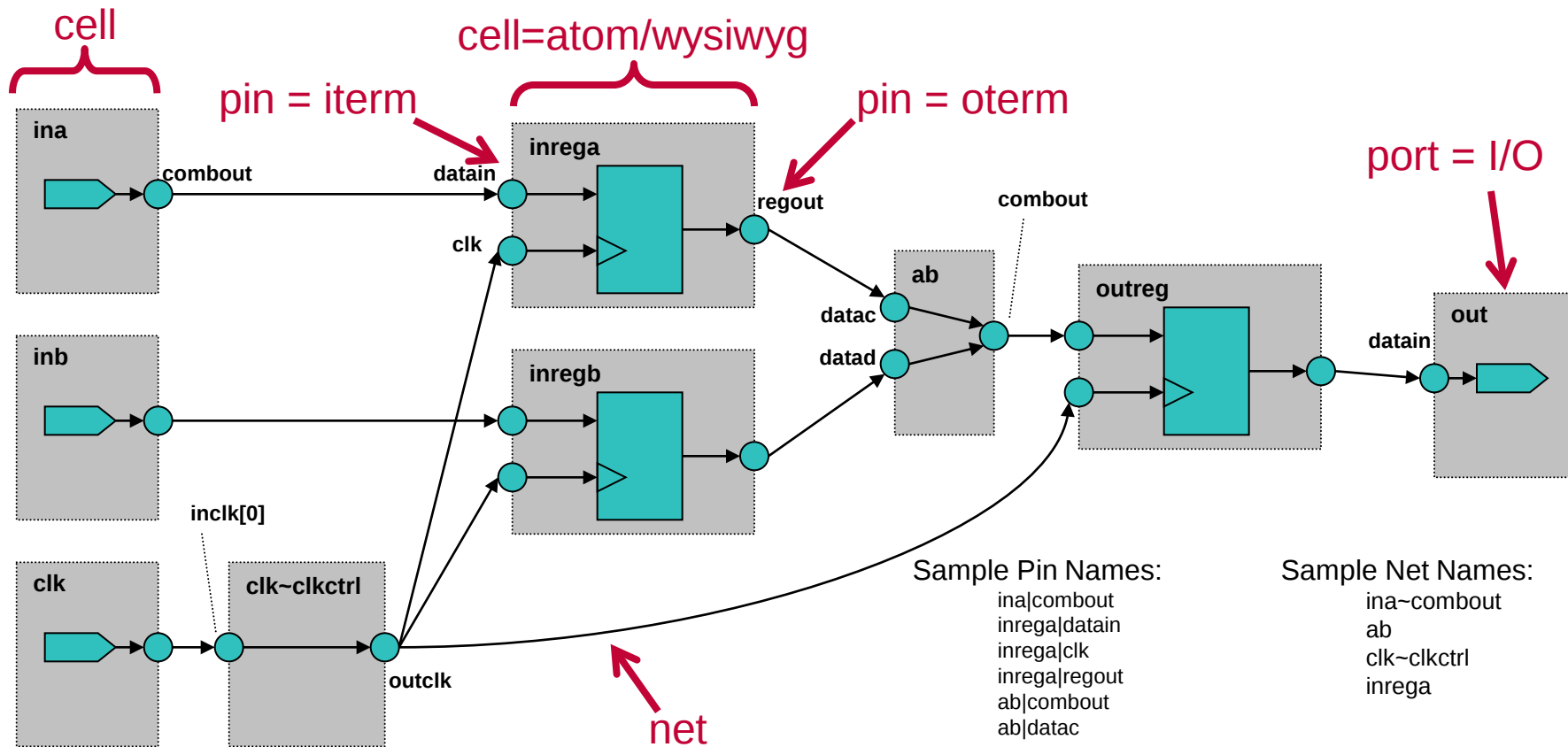
Importance of Constraining

- Timing analysis tells how a circuit **WILL** behave
- Providing timing constraints tells tools how you **WANT** the design to behave
 - Constraints paint picture of how design should operate
 - Based on design specs & specs from other devices on PCB
 - Provide goals for fitter to target during compilation
 - Provide values to which to compare timing results
- TimeQuest TA performs limited analysis without timing constraints

SDC Netlist Terminology

Term	Definition
Cell	Device building blocks (e.g. look-up tables, registers, embedded multipliers, memory blocks, I/O elements, PLLs, etc.)
Pin	Input or outputs of cells
Net	Connections between pins
Port	Top-level inputs and outputs (e.g. device pins)

SDC Netlist Example



- Paths defined in constraints by targeted endpoints (pins or ports)

© 2010 Altera Corporation—**Confidential**

ALTERA, ARRIA, CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS & STRATIX are Reg. U.S. Pat. & Tm. Off. and Altera marks in and outside the U.S.



Collections

- Searches and returns from the design netlist with a list of names meeting criteria
- Used in SDC commands
 - Some collections searched automatically during a command's usage and may not need to be specified
- Examples
 - get_ports
 - get_pins
 - get_clocks
 - all_clocks
 - all_registers
 - all_inputs
 - all_outputs

See “TimeQuest Timing Analyzer” chapter of the Quartus II Software Handbook (Volume 3) for a complete list & description of each



SDC Timing Constraints

- Clocks 
- I/O
- Asynchronous paths
- False paths
- Multicycle paths
- Delay and skew specifications



What are clocks in SDC?

- Defined, repeating signal characteristics applied to a point anywhere in the design
 - Internal: applied to a specific node being used as a clock in design (port or pin)
 - “Virtual”: No real source in, or direct interaction with design
 - Example: Clocks on external devices that feed or are fed by the FPGA design, required for I/O analysis
- Name clocks after node to which they are applied or something more meaningful
- Similar to clock settings in older Quartus II timing engine (Classic timing analyzer)



Clocks in SDC (cont.)

■ Two types

- Clock
 - Absolute or base clock
- Generated clock
 - Timing derived from another clock in design
 - Must have defined relation with source clock
 - Apply to output of logic function that modifies clock input
 - PLLs, clock dividers, output clocks, ripple clocks, etc.

■ ***All clocks are related by default***

- Cross-domain transfers analyzed

Clock Constraints

- Create clock
- Create generated clock
- PLL clocks
- Automatic clock detection & creation
- Default constraints
- Clock latency
- Clock uncertainty
- Common clock path pessimism removal

Creating a Clock

■ Command: `create_clock`

■ Options

`[-name <clock_name>]`

`-period <time>`

`[-waveform {<rise_time> <fall_time>}]`

`[<targets>]`

`[-add]`

`[ ]` = optional

Note: In general, the more options added to a constraint command, the more specific the constraint is. When options are not specified, the constraint is more generalized and pertains to more of the target.



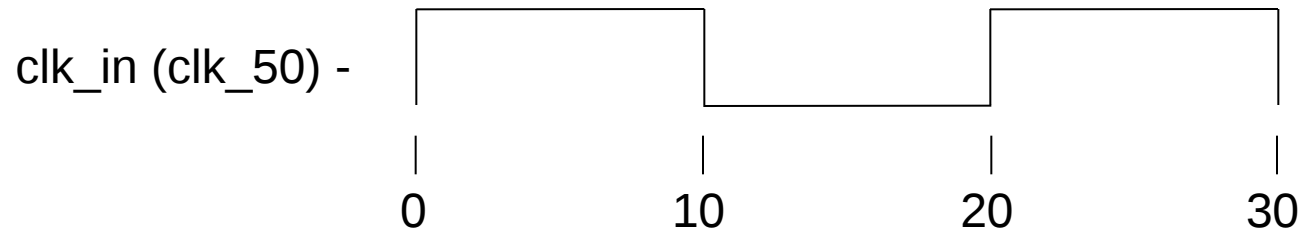
create_clock Notes

- -name: Assigns name to the clock to be used in other commands & reports when referring to clock
 - Optional; defaults to target name if not specified
- -waveform: Indicates clock offset or non-50% duty cycle clocks
 - 50% duty cycle is assumed unless otherwise indicated
- -add: Adds clock to node with existing clock
 - Without -add, warning given former clock constraint is over written
- <targets>: Target ports or pins for clock setting
 - Virtual clock created if no target specified

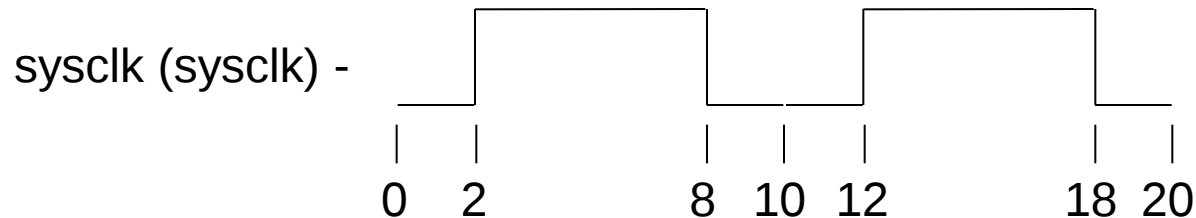


create_clock Examples

```
create_clock -period 20.0 -name clk_50 [get_ports clk_in]
```



```
create_clock -period 10.0 -waveform {2.0 8.0} [get_ports sysclk]
```



Create Clock using GUI

TimeQuest main: Constraints ⇒ Create Clock
SDC Editor: Edit ⇒ Insert Constraint ⇒ Create Clock

Create Clock

Clock name:

Period: ns

Waveform edges

Rising: ns

Falling: ns

Targets: 

SDC command:

**Edit any field
(change values; use wildcards
in targets or command)**

**Name Finder
(next slide)**

Name Finder

Clicking on Browse button opens Name Finder allowing you to search netlist for node names (similar to Quartus II Node Finder)

The Name Finder dialog box is shown with the following components and annotations:

- Collection:** A dropdown menu set to "get_ports". An arrow points to it with the annotation "Select collection to search".
- Filter:** A text field containing an asterisk (*).
- Options:** A group box containing four checkboxes: "Case-insensitive", "Hierarchical", "Compatibility mode", and "No duplicates". An arrow points to this group with the annotation "Options available depend on selected collection".
- Matches:** A section containing a "List" button and a list of 67 matches. The first few matches are "clk_in_100mhz", "clkout", and a series of "din_a" and "din_b" signals. An arrow points to the "SDC command" field with the annotation "Edit command here or final command to use wildcards".
- 9 selected names:** A list of 9 selected names, including "clk_in_100mhz" and "din_a" signals.
- SDC command:** A text field containing the command: `[get_ports {clk_in_100mhz din_a[0] din_a[1] din_a[2] din_a[3] din_a[4] din_a[5] din_a[6] din_a[7]}]`.
- Buttons:** "OK", "Cancel", and "Help" buttons at the bottom right.

Name Finder Search Options (FYI)

- All options off
 - Hierarchy levels in **Filter** match results except for *
 - * finds all names in all levels of hierarchy in selected collection
 - Ex: * | **data*** finds names starting with **data** at second level *only*
- Case-insensitive (all collections)
 - Names match **Filter** ignoring capitalization
- Hierarchical (**get_pins**; **get_cells** collections only)
 - **Filter** must be just cell name or in form of <cell> | <pin>
 - Ex: **foo** | * finds all pins on cell named **foo**
 - Ex: * | **data*** finds all pins starting with **data** at *any* level of hierarchy
- Compatibility mode (**get_pins**; **get_cells** collections only)
 - Always searches entire hierarchy
 - Ex: * | **data*** finds all pins starting with **data** at *any* level of hierarchy
 - Ex: * | * | **data*** performs the same search; extra * | not required

Creating a Generated Clock

■ Command: `create_generated_clock`

■ Options

```
[-name <clock_name>]  
[-source <master_pin>]  
[-master_clock <clock_name>]  
[-divide_by <factor>]  
[-multiply_by <factor>]  
[-duty_cycle <percent>]  
[-invert]  
[-phase <degrees>]  
[-edges <edge_list>]  
[-edge_shift <shift_list>]  
[<targets>]  
[-add]
```

create_generated_clock Notes

- –source: Species the node in design from which generated clock is derived
 - Ex. Placing source before vs. after an inverter would yield different results
- –master_clock: Used if multiple clocks exist at source due to -add option
- –edges: Relates rising/falling edges of generated clock to rising/falling edges of source based on numbered edges
- -edge_shift: Relates edges based on amount of time shifted (requires -edges)

Create Generated Clock using GUI

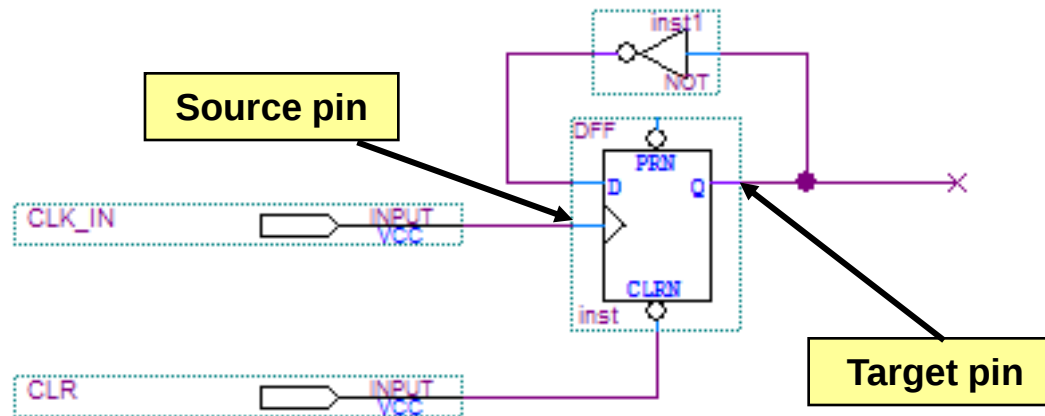
The screenshot shows the 'Create Generated Clock' dialog box. It has a blue title bar with the text 'Create Generated Clock' and a close button. The dialog is divided into several sections. At the top, there is a 'Clock name:' field with the value 'clkx2'. Below it is a 'Source:' field containing the text '[get_pins {inst1|altpll_component|pll|inclk[0]}]'. An arrow points from a yellow box labeled 'Source clock location' to this field. Below the 'Source' field is a section titled 'Relationship to source' which is enclosed in a red border. This section contains two radio buttons: 'Based on frequency' (which is selected) and 'Based on waveform'. Under 'Based on frequency', there are four input fields: 'Divide by:' (empty), 'Phase:' (empty), 'Multiply by:' (containing '2'), and 'Offset:' (empty). There is also a 'Duty cycle:' field (empty). Under 'Based on waveform', there are three 'Edge list:' fields (all empty) and three 'Edge shift list:' fields (all empty, each followed by 'ns'). At the bottom of this section is an 'Invert waveform' checkbox (unchecked). Below the 'Relationship to source' section is a 'Targets:' field containing the text '[get_pins {inst1|altpll_component|pll|clk[0]}]'. An arrow points from a yellow box labeled 'Target location' to this field. At the bottom of the dialog is an 'SDC command:' field containing the text 'create_generated_clock -name clkx2 -source [get_pins {inst1|altpll_compone'. At the very bottom are three buttons: 'Insert', 'Cancel', and 'Help'.

Source clock location

Relationship to the source

Target location

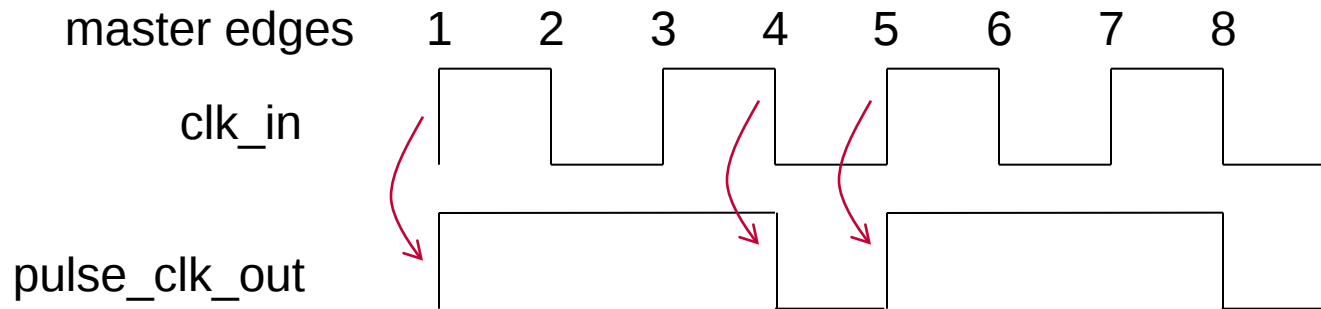
Generated Clock Example 1



```
create_clock -period 10 [get_ports clk_in]

create_generated_clock -name clk_div \
    -source [get_pins inst|clk] \
    -divide_by 2 \
    [get_pins inst|regout]
```

Generated Clock Example 2

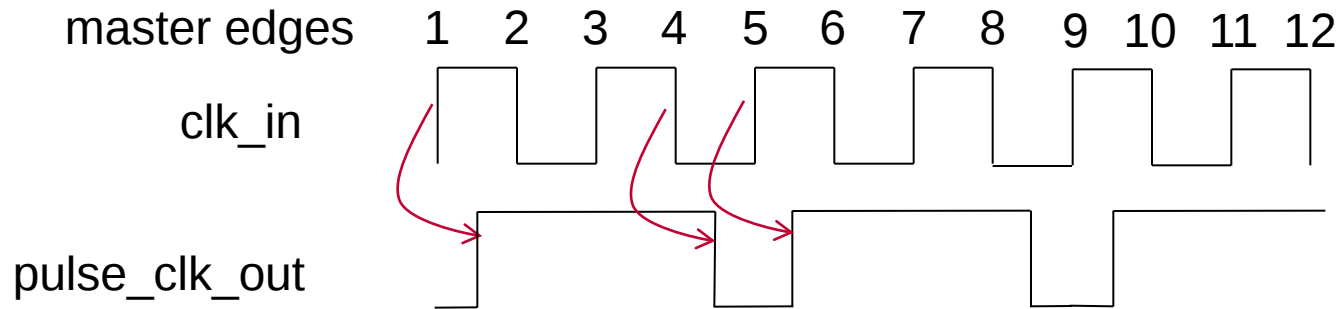


```
create_clock -period 10 [get_ports clk_in]
```

```
create_generated_clock -name pulse_clk_out -source clk_in \  
    -edges {1 4 5} \  
    [get_pins pulse_logic|out]
```

Master edges are numbered 1..<n>. In the edge list, the first
number corresponds to the first rising edge of the generated
clock. The second number is the first falling edge. The third
number is the second rising edge. Thus, a clock is created that
is half the period of the source with a 75% duty cycle.

Generated Clock Example 3

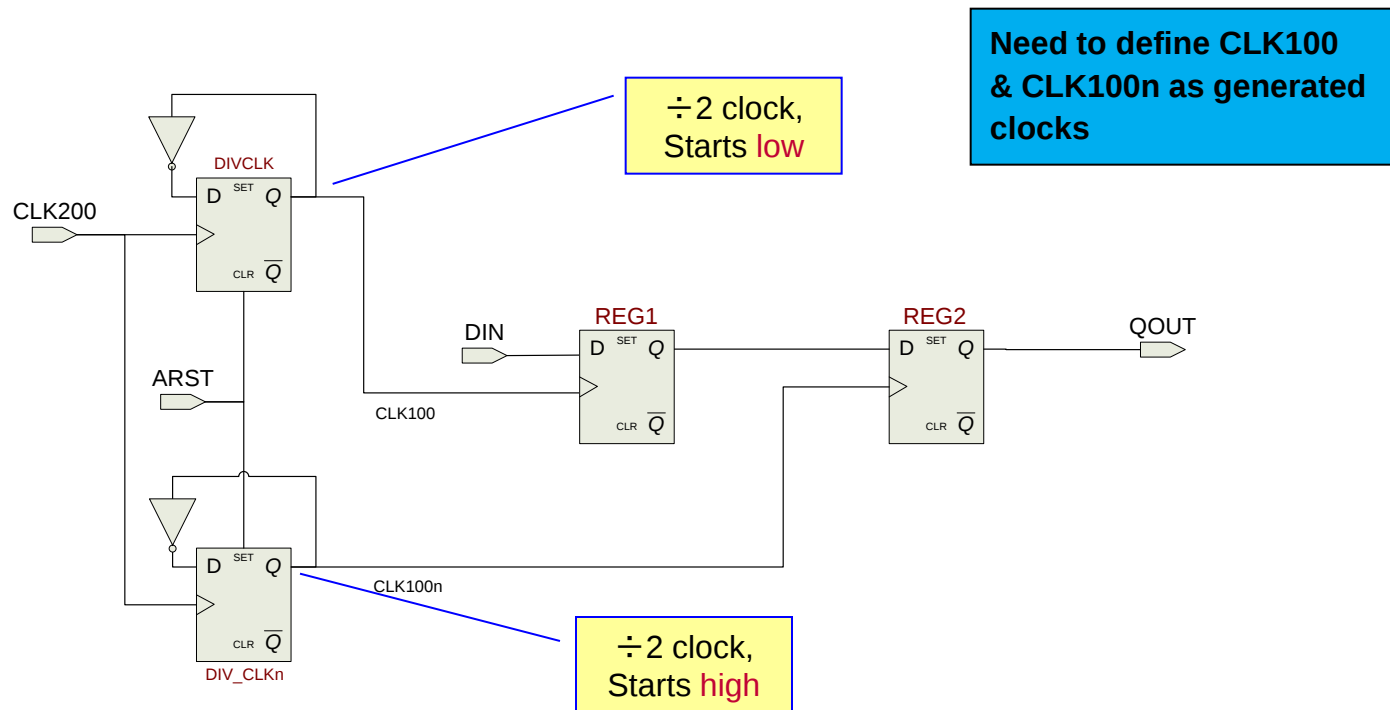


```
create_clock -period 10 [get_ports clk_in]
```

```
create_generated_clock -name pulse_clk_out -source clk_in \  
    -edges {1 4 5} -edge_shift {2.5 2.5 2.5} \  
    [get_pins pulse_logic|out]
```

Same as example 2 except -edge_shift shifts each edge indicated
amount of time

Inverted Clock Example



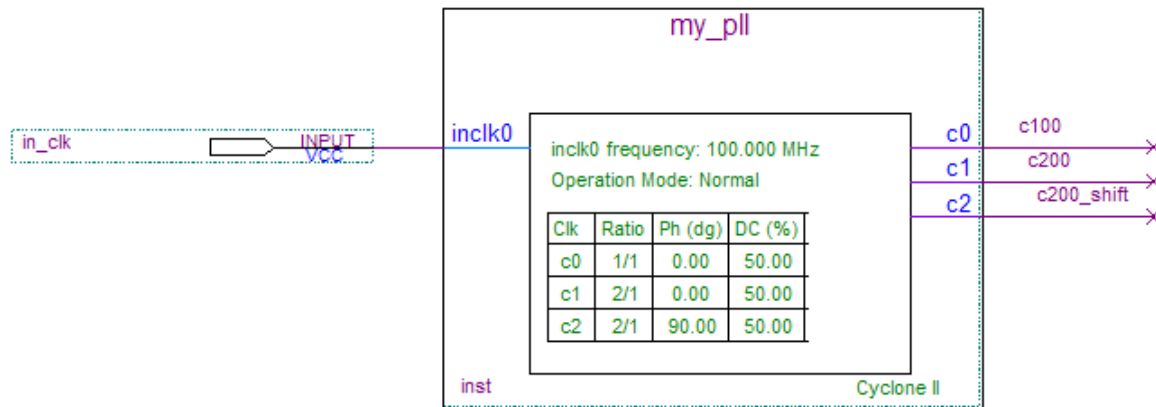
```
create_clock -period 5 [get_ports clk200]

create_generated_clock -name clk100 -source [get_pins divclk|clk] \
    -divide_by 2 [get_pins divclk|regout]
create_generated_clock -name clk100n -source [get_pins div_clkn|clk] \
    -divide_by 2 -invert [get_pins div_clkn|regout]
```

PLL Clocks (Altera SDC Extension)

- Command: `derive_pll_clocks`
 - `[-use_tan_name]`: names clock after design net name from Classic timing analyzer settings instead of the default PLL output SDC pin name
 - `[-create_base_clocks]`: generates `create_clock` constraint(s) for PLL input clocks
- Create generated clocks on all PLL outputs
 - Based on input clock & PLL settings
- Requires defining PLL input as clock unless `-create_base_clocks` is used
- Automatically updates generated clocks on PLL outputs as changes made to PLL design
- `write_sdc -expand` expands constraint into standard `create_clock` and `create_generated_clock` commands
- *Not in GUI; must be entered in SDC manually*

derive_pll_clocks Example



Using generated clock commands

```
create_clock -period 10.0 [get_ports in_clk]
create_generated_clock -name c100 \
    -source [get_pins {inst|altpll_component|pll|inclk[0]}] \
    -divide_by 1 \
    [get_pins {inst|altpll_component|pll|clk[0]}]
create_generated_clock -name c200 \
    -source [get_pins {inst|altpll_component|pll|inclk[0]}] \
    -multiply_by 2 \
    [get_pins {inst|altpll_component|pll|clk[1]}]
create_generated_clock -name c200_shift \
    -source [get_pins {inst|altpll_component|pll|inclk[0]}] \
    -multiply_by 2 \
    -phase 90 \
    [get_pins {inst|altpll_component|pll|clk[2]}]
```

Using derive_pll_command

```
create_clock -period 10.0 \
    [get_ports in_clk]
derive_pll_clocks

# or simply:

derive_pll_clocks \
    -create_base_clocks

# Note the clock names for
# the generated clocks
# will be the names of
# the PLL output pins
```

Automatic Clock Detection & Creation

- Command: `derive_clocks`
 - `[-period]`: same use as with `create_clock`
 - `[-waveform]`: same use as with `create_clock`
 - No target required
- Automatically create clocks on clock pins in design that don't already have clocks defined
- Does not work with PLL outputs (use `derive_pll_clocks`)
- SDC extension expanded with `write_sdc -expand`
- *Not in GUI*
- *Not recommended for final timing sign-off*

Default Clock Constraints

- Remember, all clocks must be constrained to analyze design with timing analysis
- If no clock constraints exist, default constraints created through two commands

```
derive_clocks -period 1.0
```

```
derive_pll_clocks
```

- Default constraints not applied if at least one clock constraint exists

- *Not in GUI*

- *Not recommended for final timing sign-off*

Non-Ideal Clock Constraints

- So far, all clocks have been ideal
 - Nice square waves
 - No accounting for delays outside of FPGA
- Add extra constraints to define realistic, non-ideal clocks
- Three special constraints
 - `set_clock_latency`
 - `set_clock_uncertainty`
 - `derive_clock_uncertainty`



Clock Latency

- Two types of latency
 - **Source:** From clock source to input port (board latency)
 - **Network:** From input port to destination register clock pin
- Network latency handled and understood by timing analysis automatically
- Need to model source latency
 - TimeQuest TA knows nothing about delays external to device
- Provide a more realistic picture of external clock behavior
- Example
 - External feedback clock: need to specify delay from clock output I/O to clock input I/O
- Clocks created with `create_clock` have default source latency of 0

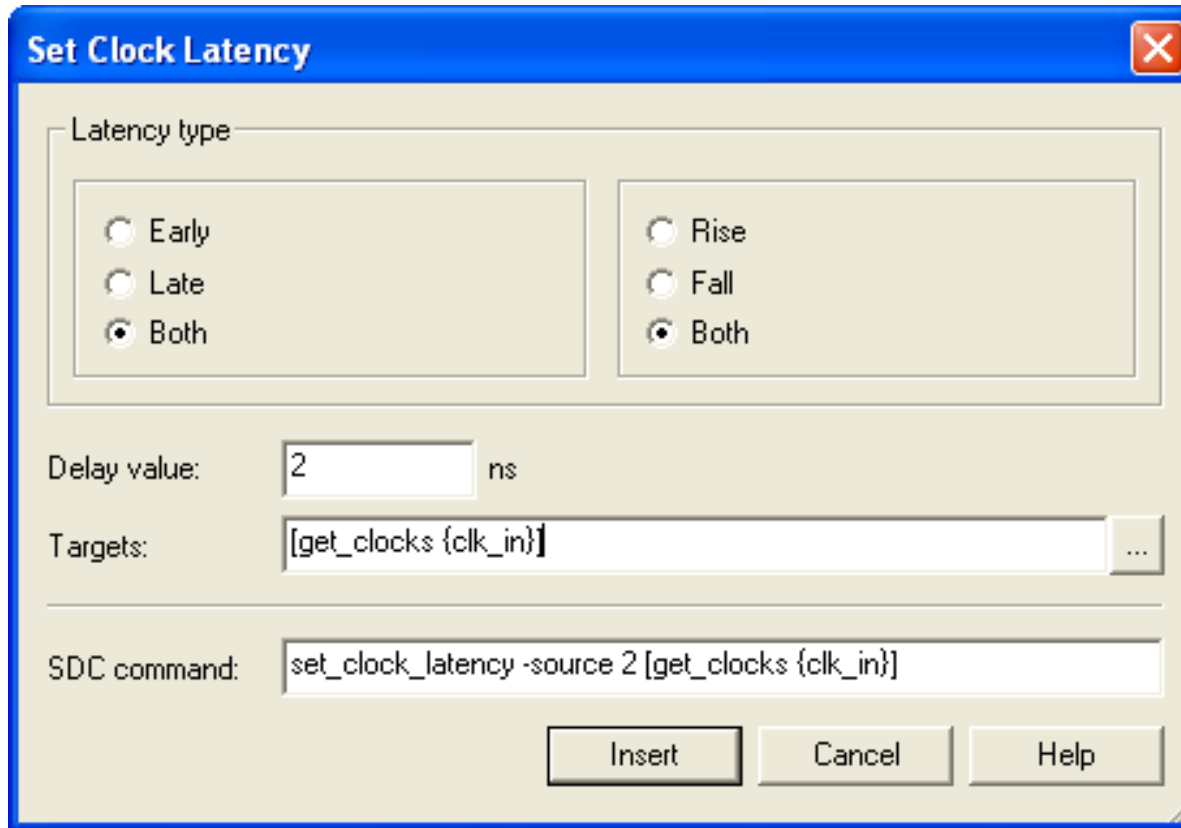
Clock Latency (cont.)

- Command: `set_clock_latency`
- Specify source latency on external path(s) to device
- Options
 - `-source`
 - `[-clock <clock_list>]`
 - `[-early | -late]`
 - `[-fall | -rise]`
 - `<delay>`
 - `<targets>`

set_clock_latency Notes

- -source: required argument for constraint (no options)
- -fall | -rise: latency applied on only falling or rising edge of clock
- -early | -late: latency on shortest/longest external path
 - Used by timing analyzer as part of definition of data/clock arrival paths for setup/hold analyses

Clock Latency (GUI)



The image shows a 'Set Clock Latency' dialog box with a blue title bar and a close button (X) in the top right corner. The dialog is divided into several sections. The 'Latency type' section contains two groups of radio buttons. The first group has 'Early', 'Late', and 'Both' options, with 'Both' selected. The second group has 'Rise', 'Fall', and 'Both' options, with 'Both' selected. Below this, the 'Delay value' is set to '2' ns. The 'Targets' field contains the text '[get_clocks {clk_in}]' and a button with three dots. The 'SDC command' field contains the text 'set_clock_latency -source 2 [get_clocks {clk_in}]'. At the bottom, there are three buttons: 'Insert', 'Cancel', and 'Help'.

Set Clock Latency

Latency type

☐ Early
☐ Late
☒ Both

☐ Rise
☐ Fall
☒ Both

Delay value: 2 ns

Targets: [get_clocks {clk_in}] ...

SDC command: set_clock_latency -source 2 [get_clocks {clk_in}]

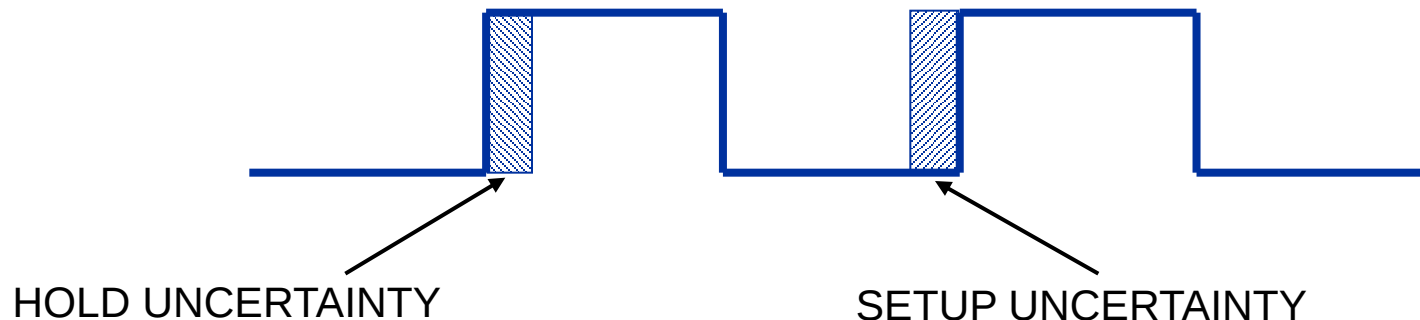
Insert Cancel Help

Clock Uncertainty

- Command: `set_clock_uncertainty`
- Use to model jitter, guard band, or skew
 - Allows generation of clocks that are non-ideal
- Options
 - `[-setup | -hold]`
 - `[-fall_from <fall_from_clock>]`
 - `[-fall_to <fall_to_clock>]`
 - `[-from <from_clock>]`
 - `[-rise_from <rise_from_clock>]`
 - `[-rise_to <rise_to_clock>]`
 - `[-to <to_clock>]`
 - `<value>`

Clock Uncertainty

- Setup uncertainty decreases setup required time
- Hold uncertainty increases hold required time

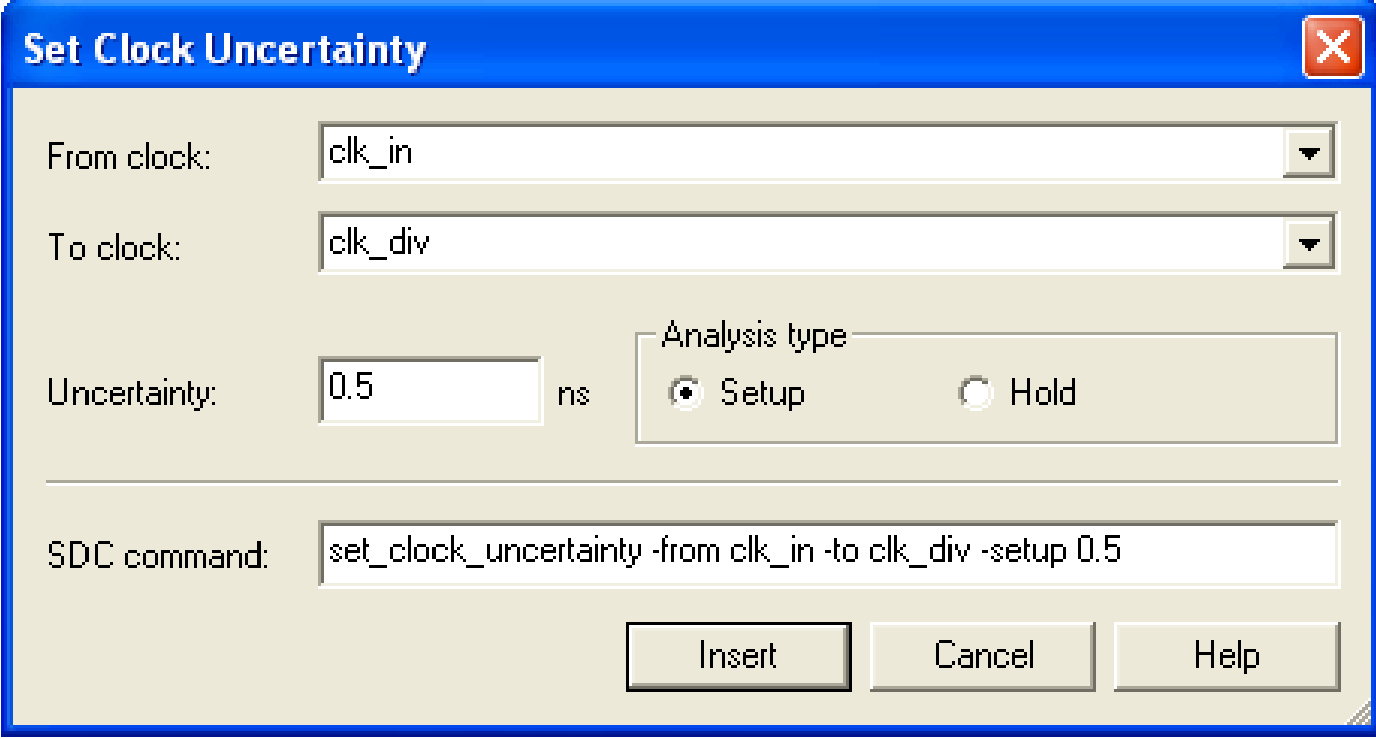


Ex. To add a 0.5-ns guardband around clock, use 250 ps of setup uncertainty and 250 ps of hold uncertainty.

set_clock_uncertainty Notes

- -from, -to: uncertainty added to transfers within single clock domain or between different domains
- -fall_from, -fall_to, -rise_from, -rise_to: apply uncertainty only on rising/falling edges of source/destination clock domain
 - *Not available in the GUI; add options manually*

Clock Uncertainty (GUI)



The image shows a GUI dialog box titled "Set Clock Uncertainty". It has a blue title bar with a close button (X) in the top right corner. The dialog contains several input fields and a section for analysis type.

From clock:

To clock:

Uncertainty: ns

Analysis type: ☒ Setup ☐ Hold

SDC command:

Buttons: Insert, Cancel, Help

Automatically Derive Uncertainty

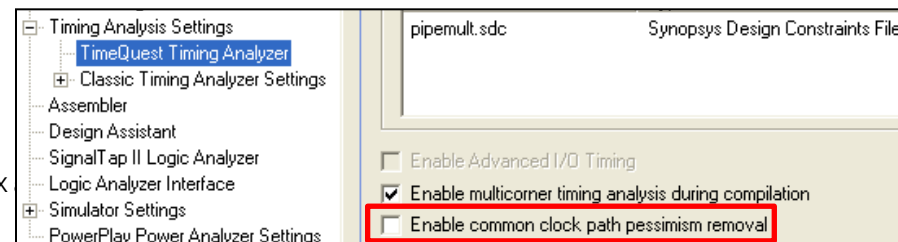
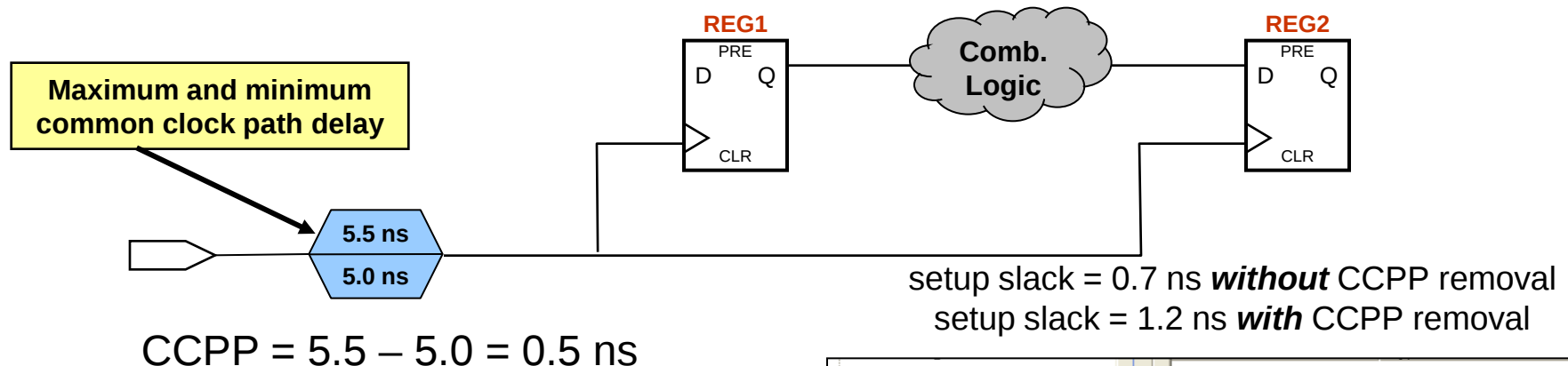
- Command: `derive_clock_uncertainty`
- Automatically derive clock uncertainties in supported devices
 - Cyclone III, Stratix II, HardCopy® II, Stratix III, and new devices
- Uncertainties created manually with `set_clock_uncertainty` have higher precedence
- Options
 - `[-overwrite]`: overwrites any existing uncertainty constraints
 - `[-add]`: adds derived uncertainties to existing constraints
- SDC extension expanded with `write_sdc -expand`
- *Not in GUI*
- *Use is recommended with supported devices*

Types of Derived Uncertainties

- Intra-clock transfers
 - Transfers within a single clock domain within FPGA
- Inter-clock transfers
 - Transfers between different clock domains within FPGA
- I/O interface clock transfers
 - Transfers between an I/O port and internal design registers
 - Requires creation of virtual clock (same as base clock)
 - Reference clock for set_input_delay and set_output_delay constraints (described later)
 - Timing analyzer derives intra- and inter-clock transfers for I/O if virtual clock not defined

Common Clock Path Pessimism Removal

- Remove clock delay pessimism to account for min/max delays on common clock paths (Cyclone III, Stratix III and newer devices)
 - Ex: Max delay for data arrival time; min delay for data required time
- Also used to improve minimum required clock pulse widths
- Enable for Fitter and for timing analysis
 - TimeQuest Timing Analyzer settings in Quartus II software
 - `enable_ccpp_removal` in TimeQuest script or console
 - May result in longer compilation time





Checking Clock Constraints

- Nodes used as clocks but not defined with SDC clock constraint considered unconstrained
- Solution
 - Use Unconstrained Paths Report to find unconstrained clocks
 - Quartus II Compilation Report timing summary
 - Run `report_ucp` command
 - Choose **Report Unconstrained Paths** (**Tasks** Pane or **Reports** menu)
 - Use Clock Report to verify clocks are constrained correctly

Unconstrained Path Report

Quartus II TimeQuest Timing Analyzer - C:/altera_trn/Quartus_II_Software_Design_Ser

File Edit View Netlist Constraints Reports Script Tools Window Help

Report

- TimeQuest Timing Analyzer Summary
- Unconstrained Paths
- Unconstrained Paths Summary**
- Clock Status Summary
- Setup Analysis
- Hold Analysis

Property	Setup	Hold
1 Illegal Clocks	0	0
2 Unconstrained Clocks	1	1
3 Unconstrained Input Ports	34	34
4 Unconstrained Input Port Paths	51	51
5 Unconstrained Output Ports	32	32
6 Unconstrained Output Port Paths	32	32

Unconstrained Paths Summary Report indicates how many clock nodes are unconstrained (along with other unconstrained paths)

Clock Status Summary Report lists each clock found and whether it was constrained

Quartus II TimeQuest Timing Analyzer - C:/altera_trn/Quartus_II_Software_D

File Edit View Netlist Constraints Reports Script Tools Window Help

Report

- TimeQuest Timing Analyzer Summary
- Unconstrained Paths
- Unconstrained Paths Summary
- Clock Status Summary**
- Setup Analysis
- Hold Analysis

	Clock	Type	Status
1	clk_test	Base	Constrained
2	clk_in_100mhz	Base	Unconstrained

Report Clocks (report_clocks)

- List details about the properties of constrained clocks

Clock properties

	Clock Name	Type	Period	Frequency	Rise	Fall	Duty Cycle	Divide by	Multiply by	Phase	Offset	Edge List	Edge Shift	Inverted	Master	Source	Targets
1	clk_in	Base	7.000	142.86 MHz	0.000	3.500											{ clk_in }
2	clk_in_vir	Virtual	7.000	142.86 MHz	0.000	3.500											{ }
3	clk_o	Generated	7.000	142.86 MHz	-0.899	2.601		1	1	-46.3				false	clk_in	...t auto_generated pll1 inclk[0]	...t auto_generated pll1 clk[2] }
4	clk_out	Generated	7.000	142.86 MHz	6.101	9.601		1	1					false	clk_o	...nt auto_generated pll1 clk[2]	{ clkout }
5	clk_x1	Generated	7.000	142.86 MHz	0.000	3.500		1	1					false	clk_in	...t auto_generated pll1 inclk[0]	...t auto_generated pll1 clk[0] }
6	clk_x2	Generated	3.500	285.71 MHz	0.000	1.750		1	2					false	clk_in	...t auto_generated pll1 inclk[0]	...t auto_generated pll1 clk[1] }

Clock names
(- name argument
or default name)

Please go to Exercise 2



SDC Timing Constraints

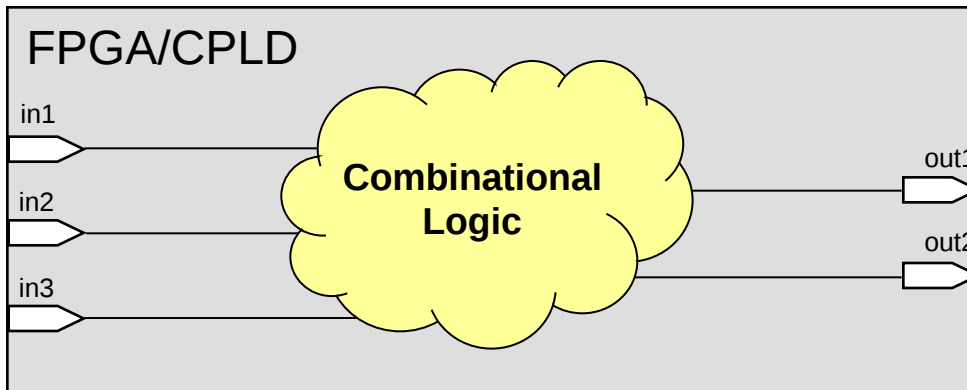
- Clocks
- I/O 
- Asynchronous paths
- False paths
- Multicycle paths
- Delay and skew specifications

I/O Constraints

- Combinational I/O interface
- Synchronous I/O interface
- Source synchronous interface

Combinational Interface

- All paths from IN to OUT need to be constrained
- Use `set_max_delay` & `set_min_delay` commands
 - Specify an absolute maximum & minimum delay between points



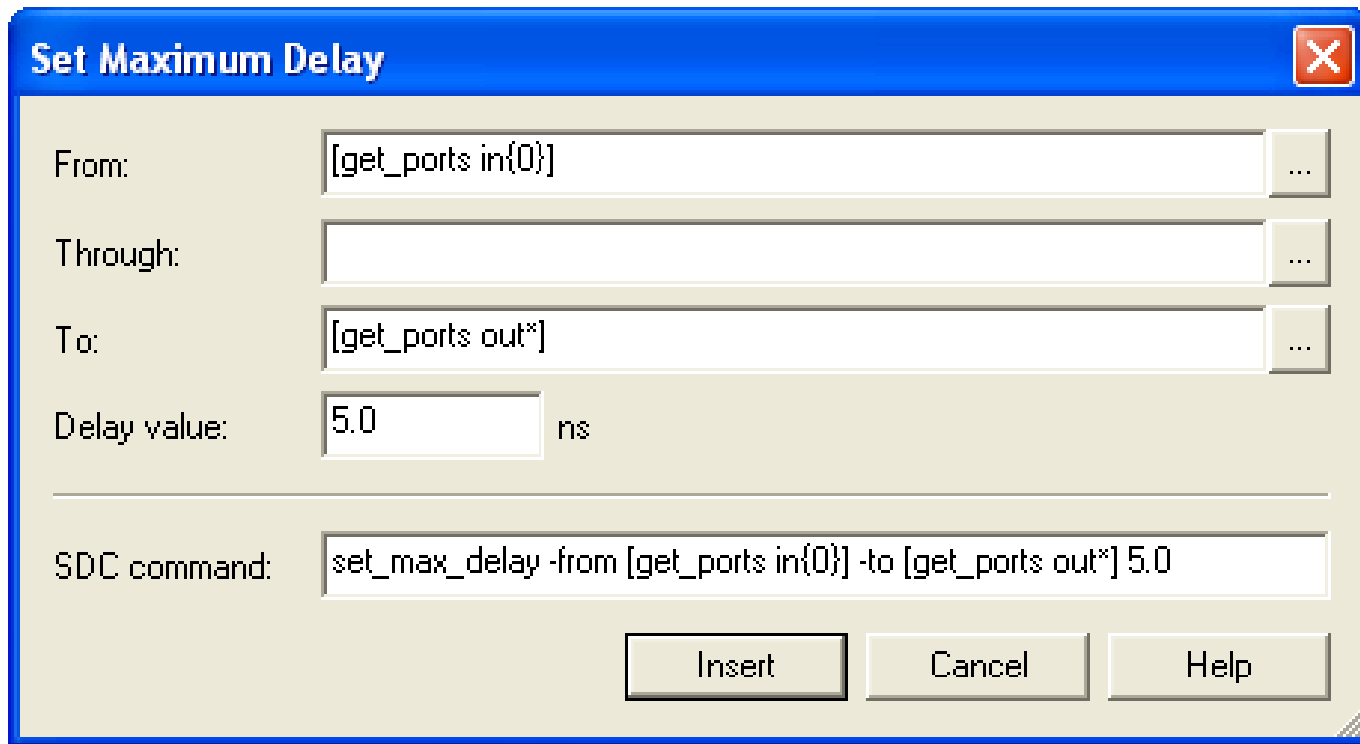
■ Options

`[-from <names>]`
`[-to <names>]`
`[-through]`
`<delay>`

set_max_delay & set_min_delay Notes

- -from & -to: Use to indicate source & destination nodes for constraints
- -through: Use to indicate the constraint should only be applied to path(s) going through a particular node name

set_max_delay & set_min_delay (GUI)



The image shows a GUI dialog box titled "Set Maximum Delay". It has a blue title bar with a close button (X) in the top right corner. The dialog contains several input fields and buttons. The "From:" field is set to "[get_ports in{0}]", the "Through:" field is empty, and the "To:" field is set to "[get_ports out*]". The "Delay value:" field is set to "5.0" with a unit of "ns". Below these fields is a text box for the "SDC command:" which contains the command "set_max_delay -from [get_ports in{0}] -to [get_ports out*] 5.0". At the bottom of the dialog are three buttons: "Insert", "Cancel", and "Help".

Set Maximum Delay

From: [get_ports in{0}] ...

Through: ...

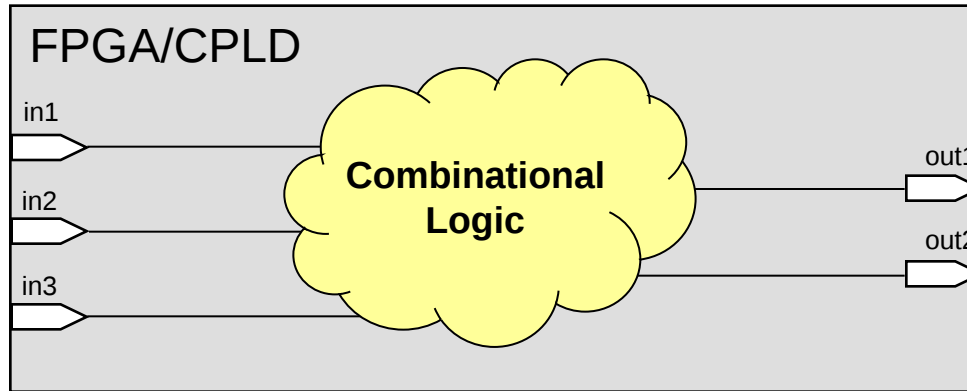
To: [get_ports out*] ...

Delay value: 5.0 ns

SDC command: set_max_delay -from [get_ports in{0}] -to [get_ports out*] 5.0

Insert Cancel Help

Combinational Interface Example



```
set_max_delay -from [get_ports in1] -to [get_ports out*] 5.0
set_max_delay -from [get_ports in2] -to [get_ports out*] 7.5
set_max_delay -from [get_ports in3] -to [get_ports out*] 9.0
```

```
set_min_delay -from [get_ports in1] -to [get_ports out*] 1.0
set_min_delay -from [get_ports in2] -to [get_ports out*] 2.0
set_min_delay -from [get_ports in3] -to [get_ports out*] 3.0
```

I/O Timing – FPGA-Centric vs. System-Centric

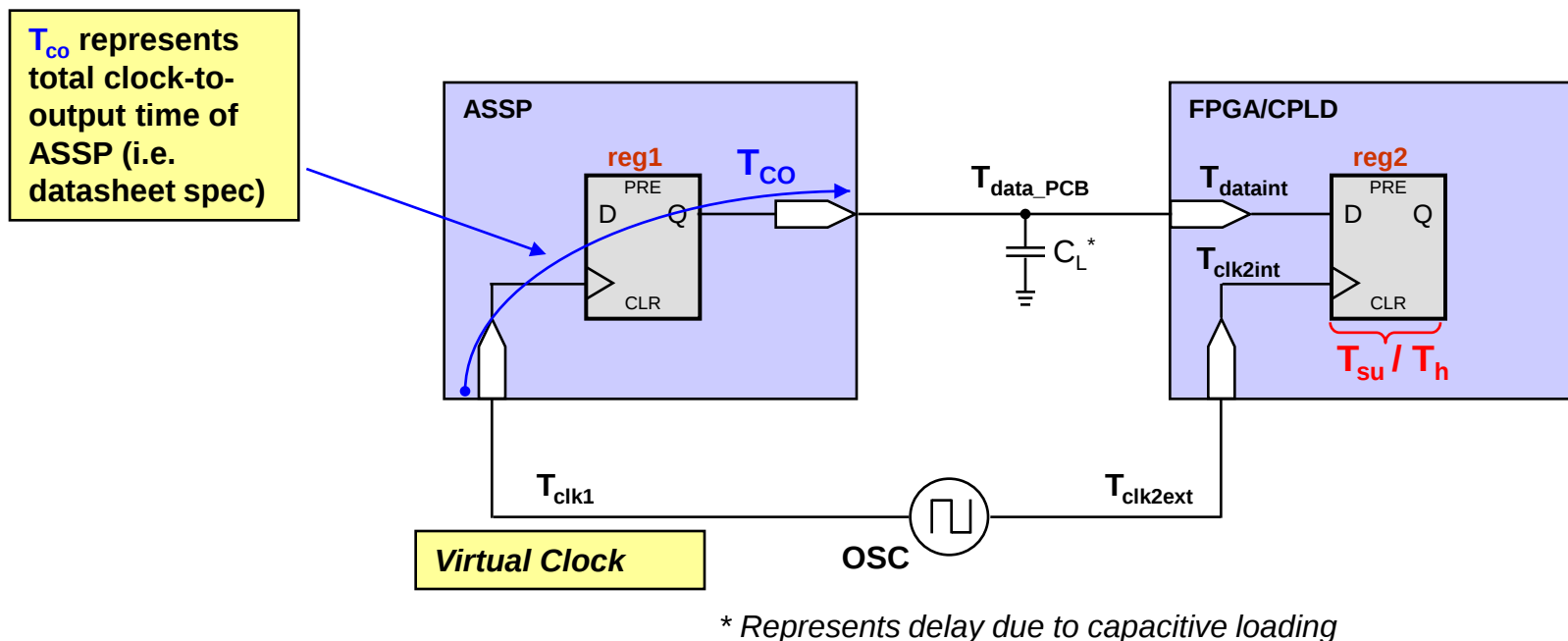
- To specify I/O timing, we must decide whether we want to look at the I/O timing FPGA-centric or system-centric
- FPGA-centric means we determine what chip-level T_{SU} , T_H , T_{CO} specs we need to meet
 - Useful when FPGA may end up in a variety of environments or interacts with defined bus interface (e.g. PCI)
- System-centric means we take into account specs of specific surrounding chips, board delays, chip-to-chip skews
- Can specify some interfaces FPGA-centric and others system-centric
 - Concentrate on system-centric in the class.

I/O Timing – Virtual Clocks

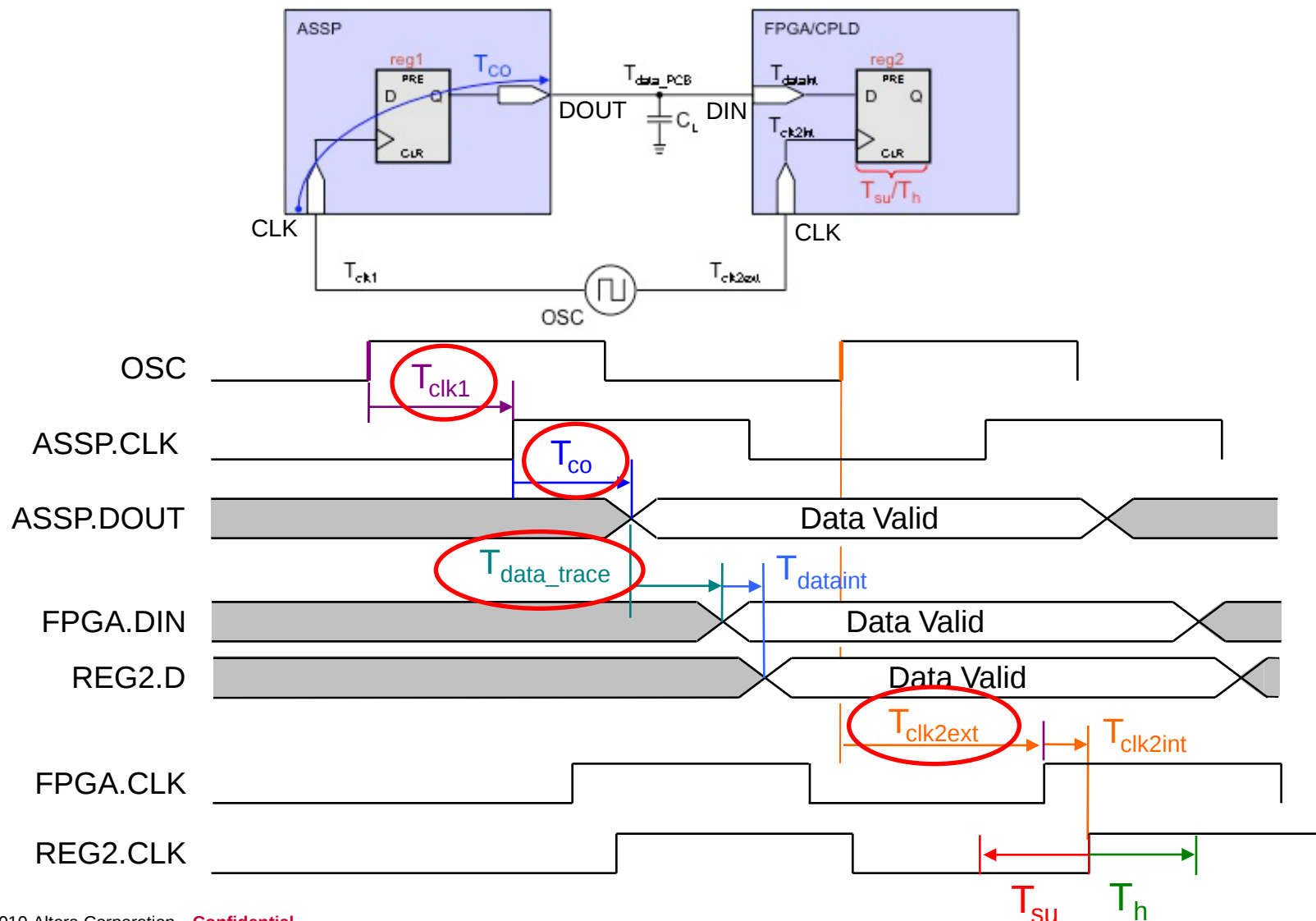
- Recommended to use virtual clocks for specifying input / output delays
- Separate clock to represent external clock timing allows `derive_clock_uncertainty` to calculate correctly
- Easier to identify input / output paths in timing reports by virtual clocks at launch or latch edge
- In some cases (e.g. DDR), difficult to accurately constrain I/O without using virtual clocks

Synchronous Inputs

- Need to specify timing relationship from ASSP to FPGA/CPLD to guarantee setup/hold in FPGA/CPLD



Synchronous Inputs



Constraining Synchronous Inputs

- Use `set_input_delay (-max option)` command to constrain **input setup time** (maximum time to arrive and still meet T_{su})
 - Calculated input delay value represents all delays external to device

System-centric:

$$\begin{aligned}\text{input delay max} &= \text{Data trace (max)} - \text{Board clock skew (min)} + T_{co(max)} \\ &= (T_{data_PCB(max)} + T_{CL}) - (T_{clk2ext(min)} - T_{clk1(max)}) + T_{co(max)}\end{aligned}$$

FPGA-centric:

$$\text{input delay max} = (T_{launch} - T_{latch}) - T_{su}$$

- Use `set_input_delay (-min option)` command to constrain **input hold time** (minimum time to stay active and still meet T_h)
 - Calculated input delay value represents all delays external to device

System-centric:

$$\begin{aligned}\text{input delay min} &= \text{Data trace (min)} - \text{Board clock skew (max)} + T_{co(min)} \\ &= (T_{data_PCB(min)} + T_{CL}) - (T_{clk2ext(max)} - T_{clk1(min)}) + T_{co(min)}\end{aligned}$$

FPGA-centric:

$$\text{input delay min} = T_h$$

set_input_delay Command

- Constrains input pins by specifying *external* device timing parameters

- Options

```
-clock <clock_name>  
[-clock_fall]  
[-rise | -fall]  
[-max | -min]  
[-add_delay]  
[-source_latency_included]  
<delay value>  
<targets>
```



set_input_delay Notes

- -clock: Specifies the clock driving the source (external) register
 - Use the virtual clock
 - Used to determine launch edge vs. latch edge relationship
- -clock_fall: Use to specify input signal was launched by a falling edge clock transition
- -rise | -fall: Use to indicate whether input delay value is for a rising or falling edge transaction
- To fully constrain, must specify both -max & -min
 - Each will default to the value of the other setting if only one assigned (same with rise/fall)
 - Warning message if one or the other not specified

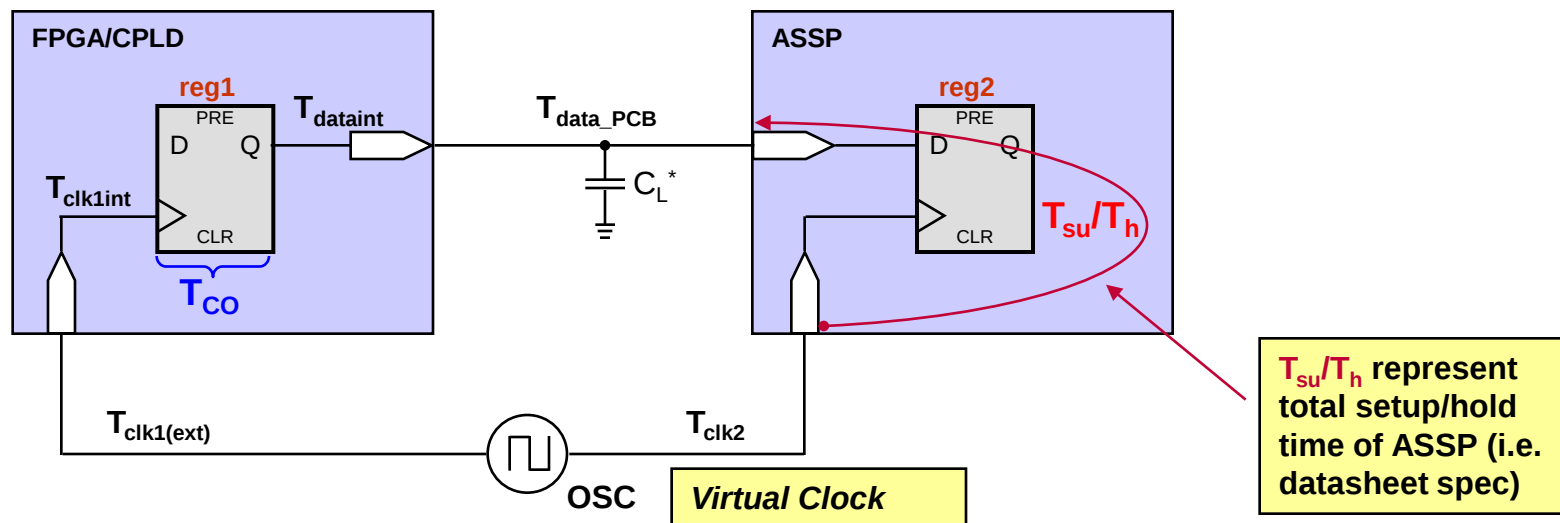


set_input_delay Notes

- -add_delay: Use to specify multiple constraints on single input
 - Only one **set** of max/min & rise/fall constraints allowed on an input pin
- -source_latency_included: input delay value specified includes clock source latency normally added automatically
 - Tells TimeQuest to ignore any clock latency constraints applied to source clock

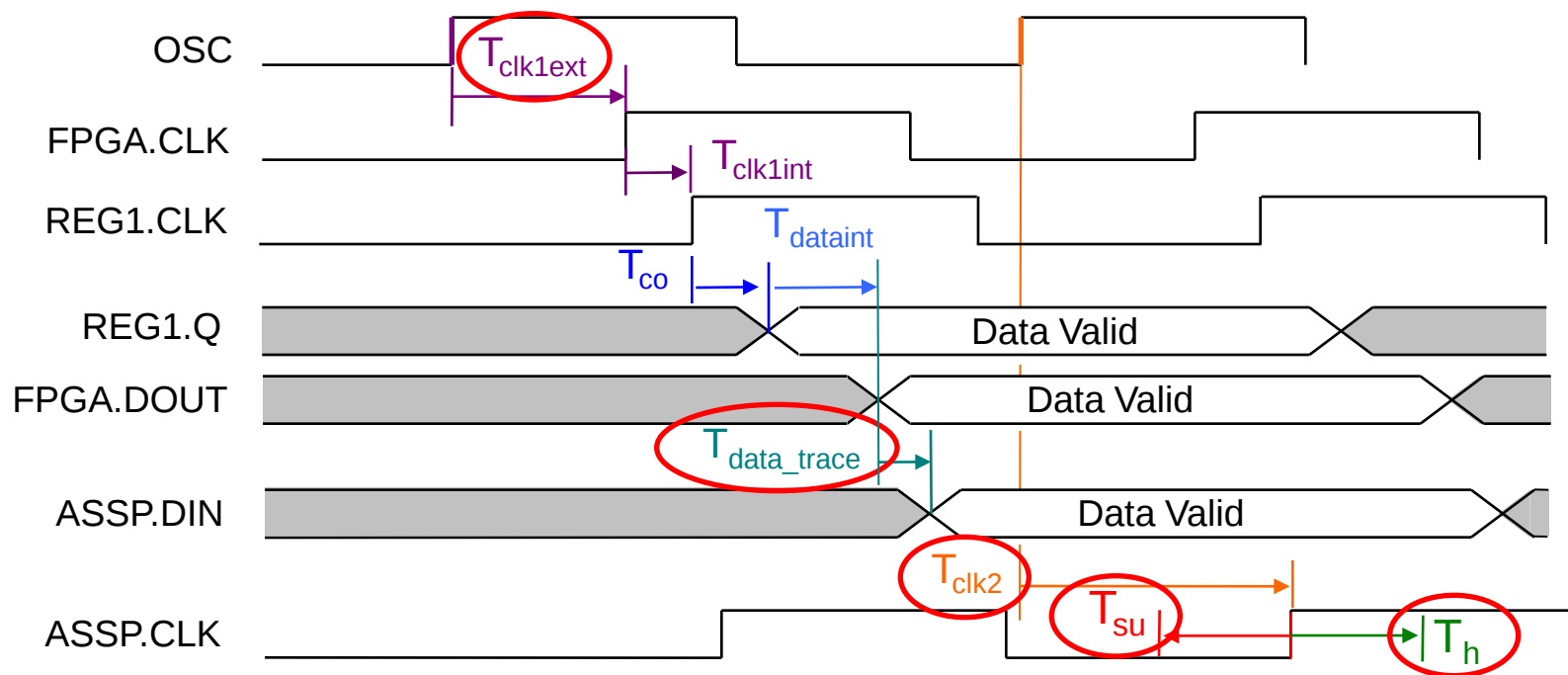
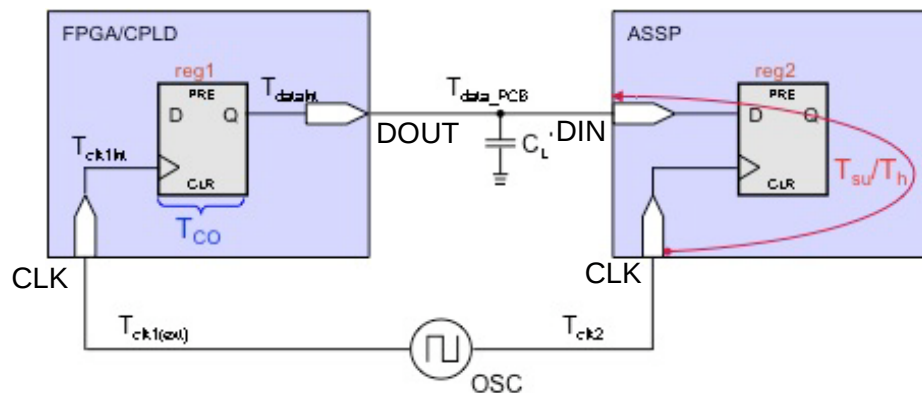
Synchronous Outputs

- Need to specify timing relationship from FPGA/CPLD to ASSP to guarantee clock-to-output times in FPGA/CPLD



** Represents delay due to capacitive loading*

Synchronous Outputs



Constraining Synchronous Outputs

- Use `set_output_delay (-max option)` command to constrain **maximum clock-to-output** (maximum time to arrive and still meet ASSP's T_{su})
 - Calculated output delay value represents all delays external to device

System-Centric:

$$\begin{aligned}\text{output delay max} &= \text{Data trace (max)} - \text{Board clock skew (min)} + T_{su} \\ &= (T_{data_PCB(max)} + T_{CL}) - (T_{clk2(min)} - T_{clk1ext(max)}) + T_{su}\end{aligned}$$

FPGA-Centric:

$$\text{output delay max} = (T_{launch} - T_{latch}) - T_{CO(max)}$$

- Use `set_output_delay (-min option)` command to constrain **minimum clock-to-output** (minimum time to stay active and still meet ASSP's T_h)
 - Calculated output delay value represents all delays external to device

System-Centric:

$$\begin{aligned}\text{output delay min} &= \text{Data trace (min)} - \text{Board clock skew (max)} - T_h \\ &= (T_{data_PCB(min)} + T_{CL}) - (T_{clk2(max)} - T_{clk1ext(min)}) - T_h\end{aligned}$$

FPGA-Centric:

$$\text{output delay min} = -T_{CO(min)}$$

set_output_delay Command

- Constrains output pins by specifying external device timing parameters

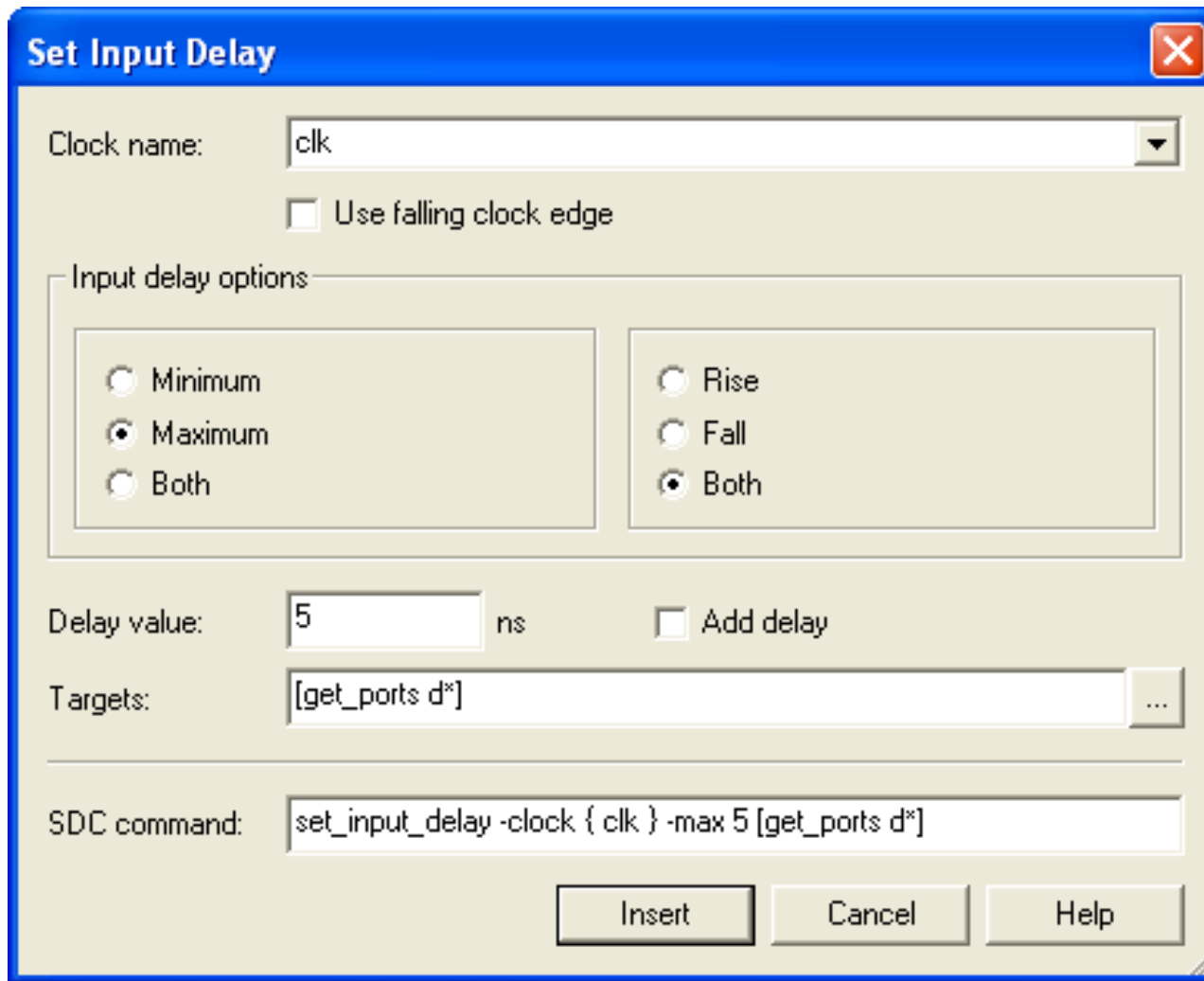
- Options

```
-clock <clock_name>  
[-clock_fall]  
[-rise | -fall]  
[-max | -min]  
[-add_delay]  
<delay value>  
<targets>
```

set_output_delay Notes

- -clock_fall: output signal was latched by a falling edge clock transition
- All others same as set_input_delay command

Input/Output Delays (GUI)



The image shows a 'Set Input Delay' dialog box with a blue title bar and a close button. It contains several input fields and radio buttons for configuring input delays.

Set Input Delay

Clock name:

☐ Use falling clock edge

Input delay options

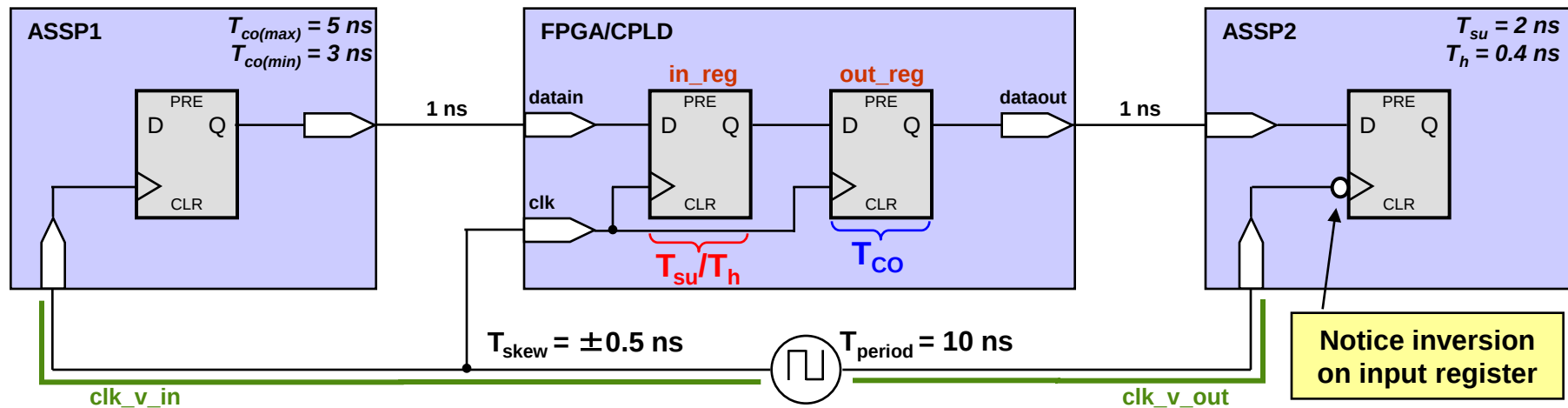
☐ Minimum	☐ Rise
☒ Maximum	☐ Fall
☐ Both	☒ Both

Delay value: ns ☐ Add delay

Targets:

SDC command:

Synchronous I/O Example



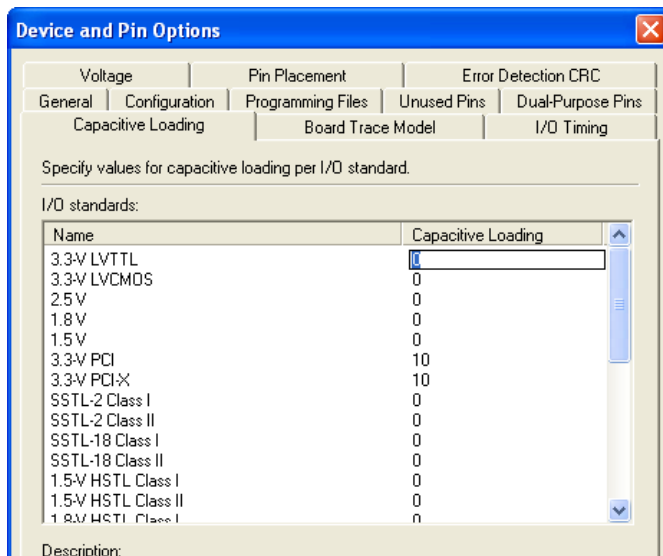
```
create_clock -period 10 -name clk [get_ports clk]
create_clock -period 10 -name clk_v_in # virtual clock for input constraint
create_clock -period 10 -name clk_v_out # virtual clock for output constraint

set_input_delay -clock clk_v_in -max [expr 1 - (-0.5) + 5] [get_ports datain]
set_input_delay -clock clk_v_in -min [expr 1 - 0.5 + 3] [get_ports datain]

set_output_delay -clock clk_v_out -max [expr 1 - (-0.5) + 2] \
    -clock_fall [get_ports dataout]
set_output_delay -clock clk_v_out -min [expr 1 - 0.5 - 0.4] \
    -clock_fall [get_ports dataout]
```

*Note: **expr** in these constraints is used to simply calculate the value of the equation broken down into the 3 parts defined by the input/output delay equations*

Output Pin Load



Capacitive Loading tab of Device and Pin Options button in Device Settings

- Specifies output pin loading in picofarads (pf)
 - Changes default loading value of I/O standard
 - Changes t_{co} of output pins
- Allows designer to accurately model board conditions
- Specify for entire I/O standard in Device Settings
- Apply to individual output or bidirectional pins in Assignment Editor or Pin Planner All Pins list
- Applies to 90-nm or older devices only

Named: [] Edit: [X] [Y] 20 Filter: Pins: all

	Node Name	Direction	Location	I/O Standard	Output Pin Load	I/O Bank
13	reset	Input	PIN_N3	3.3-V LVTTTL (default)		6
14	yn_out[7]	Output	PIN_J6	3.3-V LVTTTL (default)	20	5
15	yn_out[6]	Output	PIN_L8	3.3-V LVTTTL (default)	20	5
16	yn_out[5]	Output	PIN_H1	3.3-V LVTTTL (default)	20	5
17	yn_out[4]	Output	PIN_K2	3.3-V LVTTTL (default)	20	5
18	yn_out[3]	Output	PIN_H2	3.3-V LVTTTL (default)	20	5
19	yn_out[2]	Output	PIN_J5	3.3-V LVTTTL (default)	20	5
20	yn_out[1]	Output	PIN_L2	3.3-V LVTTTL (default)	20	5
21	yn_out[0]	Output	PIN_K5	3.3-V LVTTTL (default)	20	5
22	yvalid	Output	PIN_L7	3.3-V LVTTTL (default)		5
23	~DATA0~	Input	PIN_E13	3.3-V LVTTTL (default)		3

Tcl: set_instance_assignment -name OUTPUT_PIN_LOAD <value> -to <pin name>

Advanced I/O Timing

- Enhances analysis (over capacitive loading) by allowing user to enter board-level parameters (Cyclone III, Stratix II, III, & IV devices only)
 - Use in lieu of or in addition to HSPICE & IBIS modeling
- View signal integrity metrics in Compilation Report (TimeQuest folder)

Enable in TQ settings, then
Device Settings ⇒ Device &
Pin Options

Specify values for Board Trace Model per I/O standard.

I/O standard: 3.3-V LVTTTL

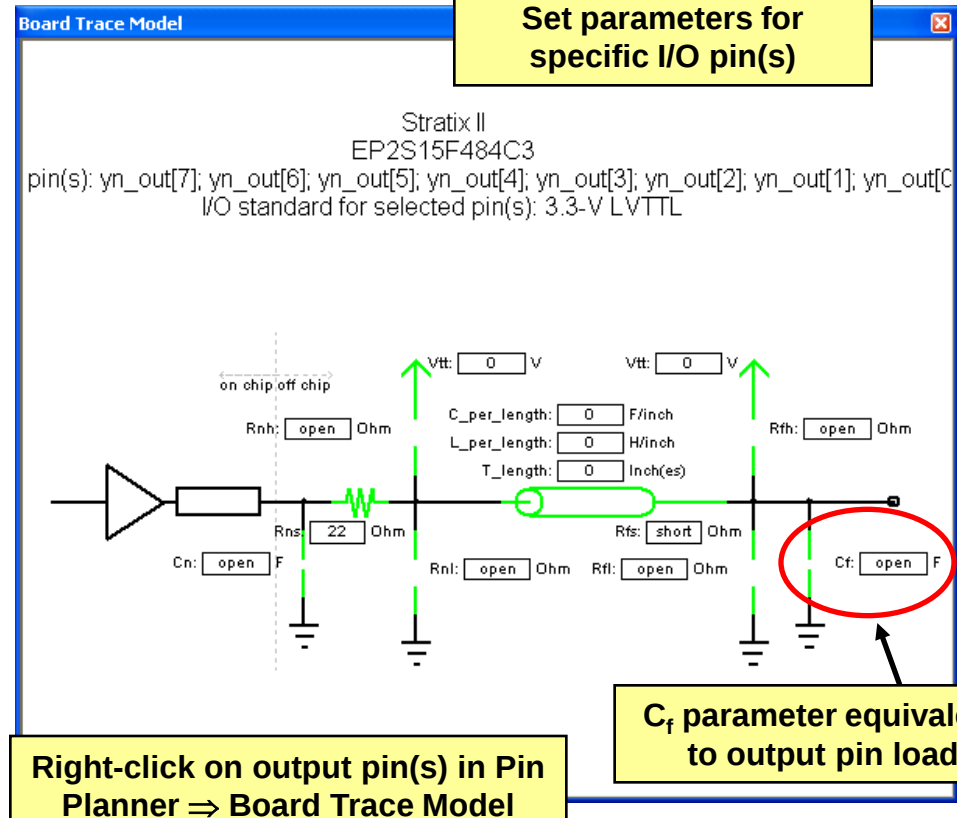
Board trace model:

Name	Value
Near pull-up resistance (in ohms)	open
Near pull-down resistance (in ohms)	open
Near capacitance (in farads)	open
Near series resistance (in ohms)	short
Near transmission line distributed inductance (in h...)	0
Near transmission line distributed capacitance (in f...)	0
Near transmission line length (in inches)	0
Far transmission line distributed inductance (in he...)	0
Far transmission line distributed capacitance (in fa...)	0
Far transmission line length (in inches)	0
Far pull-up resistance (in ohms)	open
Far pull-down resistance (in ohms)	open
Far capacitance (in farads)	open

Reset

OK Cancel

Set for all pins using
I/O standard



Set parameters for
specific I/O pin(s)

Right-click on output pin(s) in Pin
Planner ⇒ Board Trace Model

C_f parameter equivalent
to output pin load

Synchronous I/O Timing Summary

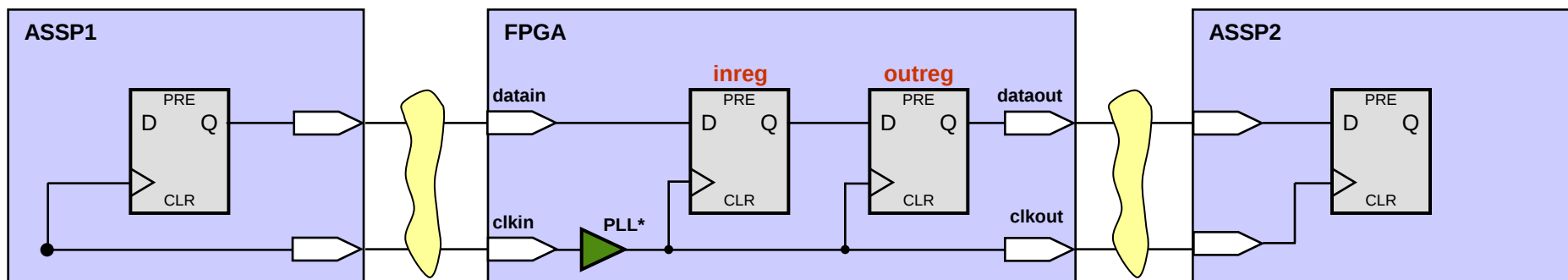
	System-centric	FPGA-centric
Input delay (max)	Board delay (max) - Board clock Skew (min) + $T_{CO(max)}$ Board delay (max) = $T_{data_PCB(max)} + T_{CL}$ Board clock skew (min) = $T_{clk2ext(min)} - T_{clk1(max)}$	$T - T_{SU}$
Input delay (min)	Board Delay (min) - Board clock skew (max) + $T_{CO(min)}$ Board delay (min) = $T_{data_PCB(min)} + T_{CL}$ Board clock skew (max) = $T_{clk2ext(max)} - T_{clk1(min)}$	T_h
Output delay (max)	Board Delay (max) - Board clock skew (min) + T_{SU} Board delay (max) = $T_{data_PCB(max)} + T_{CL}$ Board clock skew (min) = $T_{clk2(min)} - T_{clk1ext(max)}$	$T - T_{CO(max)}$
Output delay (min)	Board Delay (min) - Board clock skew (max) - T_h Board delay (min) = $T_{data_PCB(min)} + T_{CL}$ Board clock skew (max) = $T_{clk2(max)} - T_{clk1ext(min)}$	$-T_{CO(min)}$

Note: The T_{SU} , T_h , $T_{CO(max)}$ and $T_{CO(min)}$, for FPGA-centric, are chip-level timing requirements.

$$T = (T_{latch} - T_{launch})$$

Please go to Exercise 3

Source-Synchronous Interfaces



- Both data & clock transmitted by host device with designated phase relationship (e.g. edge or center-aligned)
 - No clock tree skew included in calculation
 - Target device uses transmitted clock to sample incoming data
- Skew between data and clock is the limitation factor of transmission speed
 - Enables higher interface speeds (compared to using system clock)

*** The optional PLL in this example, represented by a single symbol, is actually generating multiple outputs clocks**

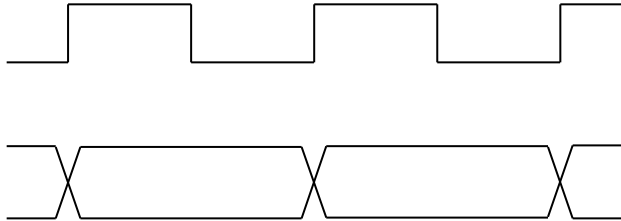
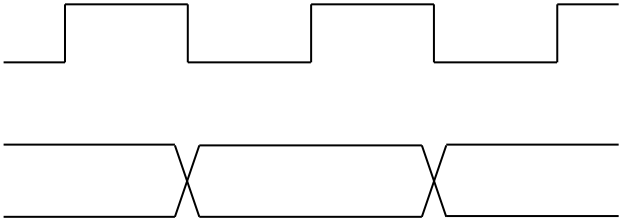
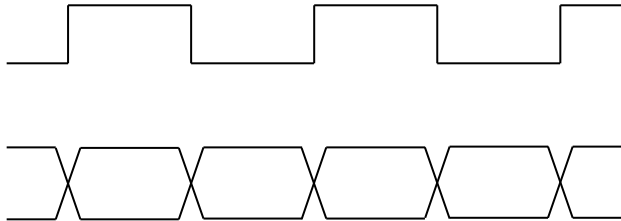
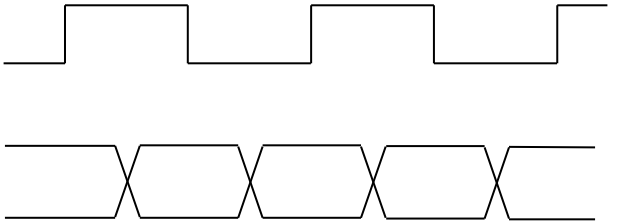
© 2010 Altera Corporation—**Confidential**

ALTERA, ARRIA, CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS & STRATIX are Reg. U.S. Pat. & Tm. Off. and Altera marks in and outside the U.S.



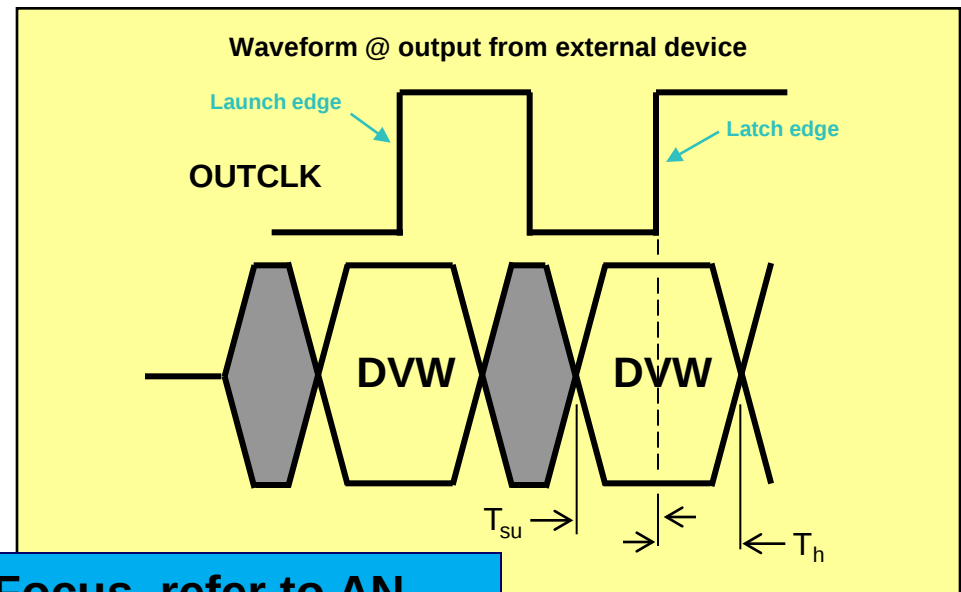
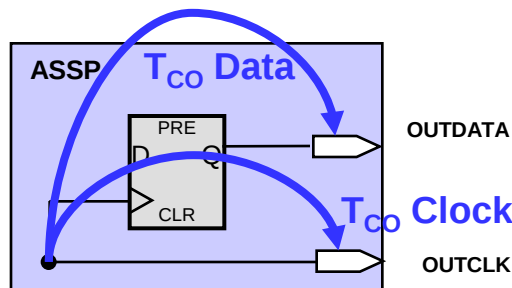
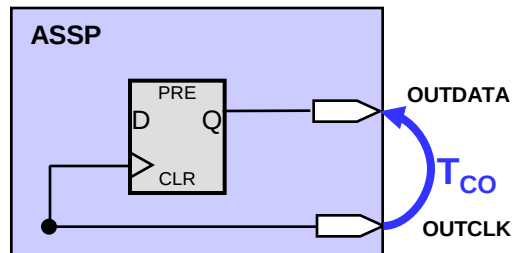


Source Synchronous Clocking Schemes

	Edge-Aligned	Center-Aligned
SDR		
DDR		

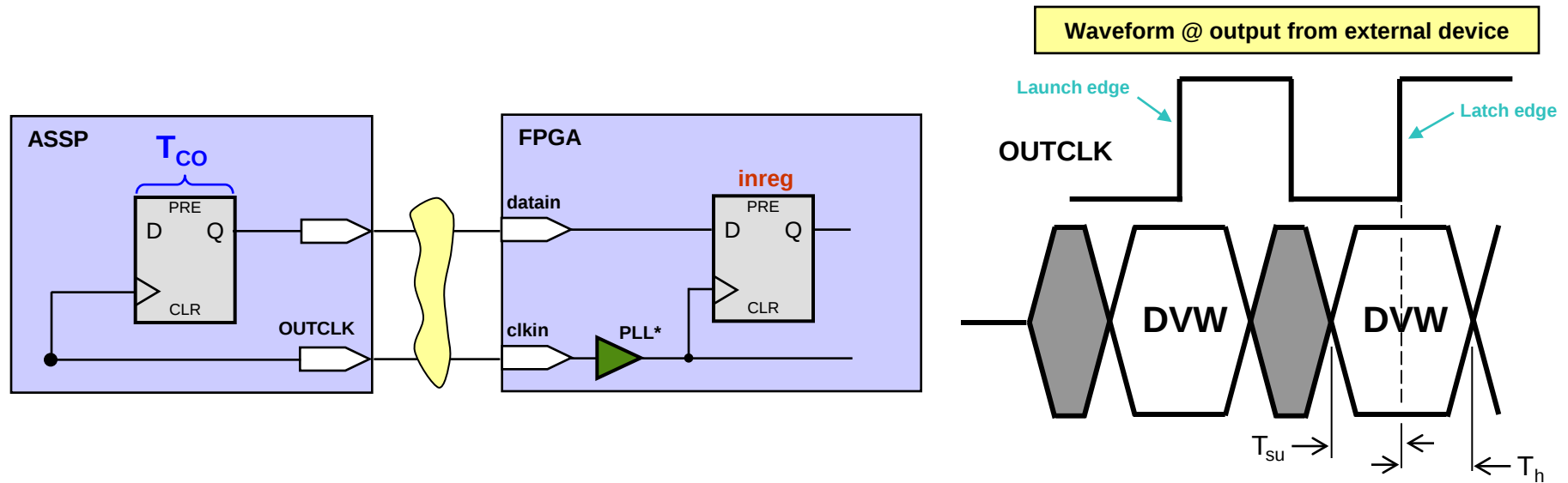
SDR Source-Synchronous Input (Data Sheet)

- The FPGA input constraints vary depending on what's given by the data sheet of the ASSP:
 - T_{CO} relative to the output clock
 - T_{CO} relative to the input clock
 - Specify setup and hold parameters for the data output



Class Focus, refer to AN 433 for other T_{CO} methods

SDR Source-Synchronous Input (Center-Aligned)



- Total setup/hold relationship of FPGA to clock (clkkin) already defined by output waveform of external device
 - T_{su} is start of DVW
 - T_h is end of DVW
- Must derive `set_input_delay` values from T_{su} & T_h

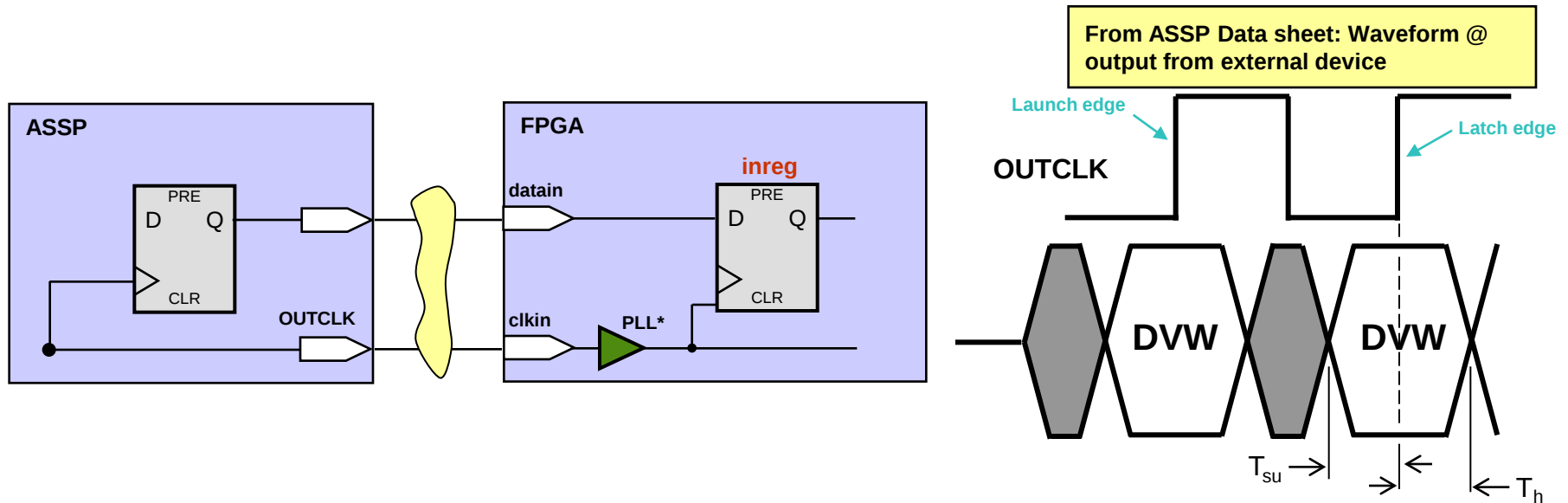
* **The PLL in this example is used to maintain the input clock to data relationship**

© 2010 Altera Corporation—**Confidential**

ALTERA, ARRIA, CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS & STRATIX are Reg. U.S. Pat. & Tm. Off. and Altera marks in and outside the U.S.



SDR Source-Synchronous Input (Center-Aligned)



System-centric approach:

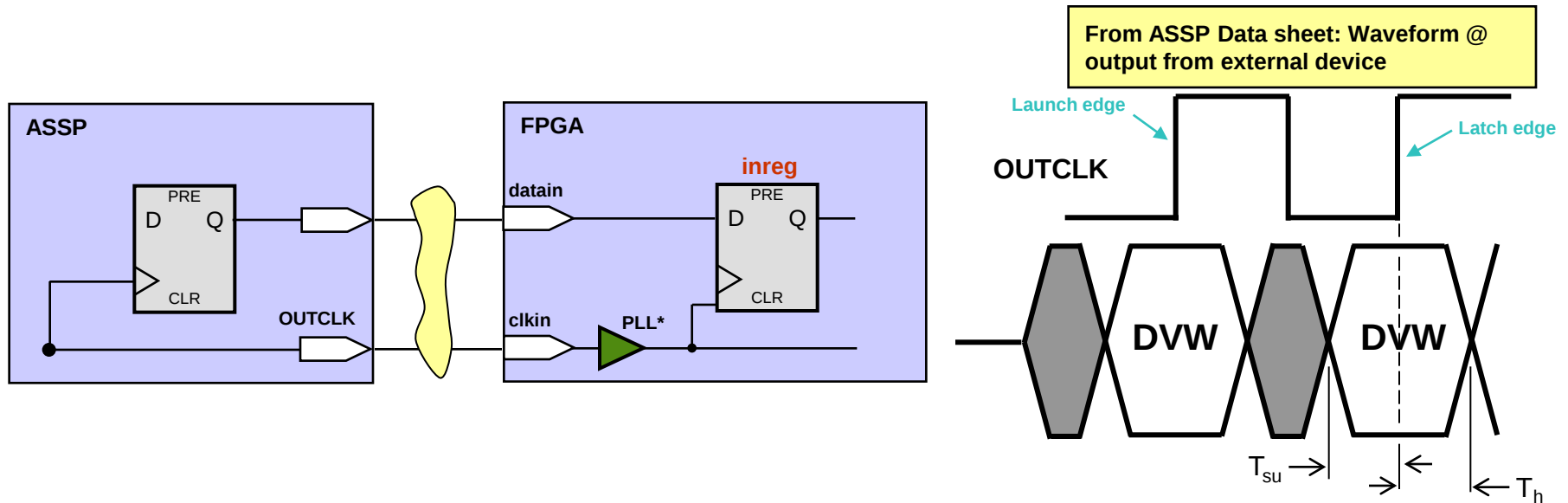
$$\text{input delay max} = \text{data trace (max)} - \text{clock trace (min)} + (\text{latch edge} - \text{launch edge})^* - T_{su}$$

FPGA-centric approach:

$$\text{input delay max} = (\text{latch edge} - \text{launch edge})^* - T_{su}$$

**Typically 1 clock period for SDR*

SDR Source-Synchronous Input (Center-Aligned)



System-centric approach:

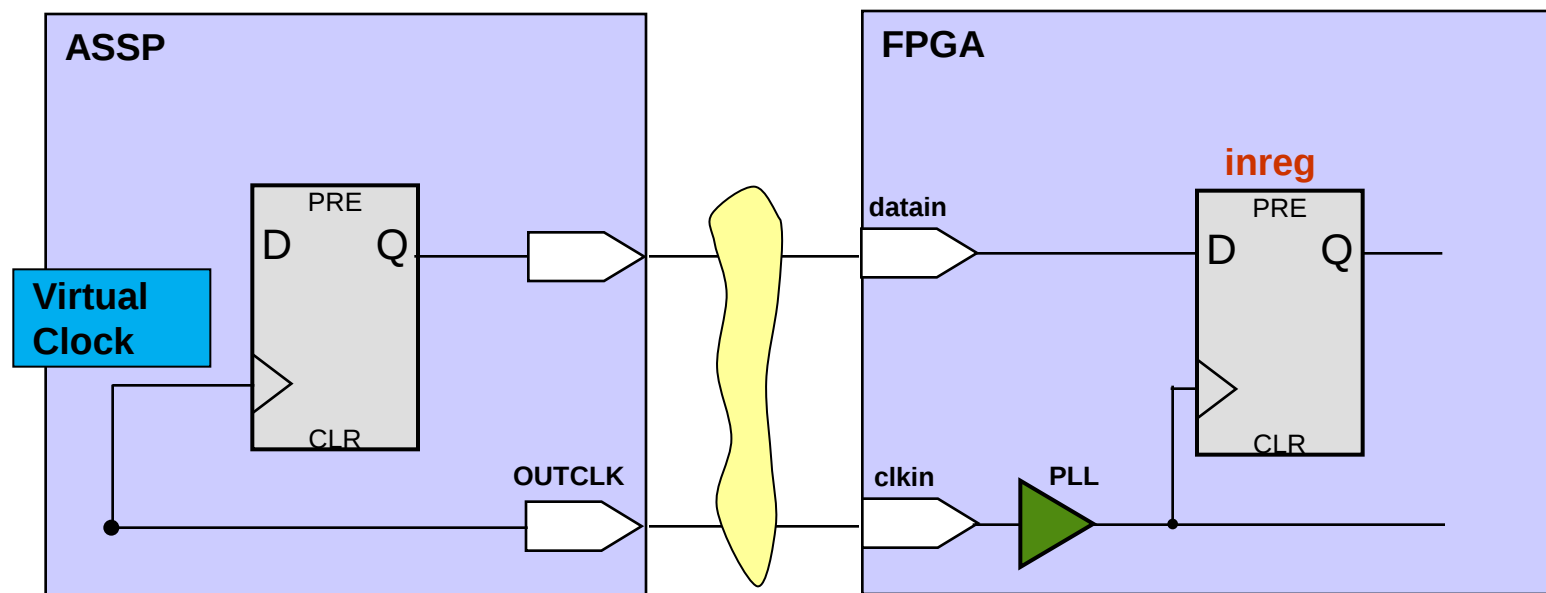
$$\text{input delay min} = \text{data trace (min)} - \text{clock trace (max)} + T_h$$

FPGA-centric approach:

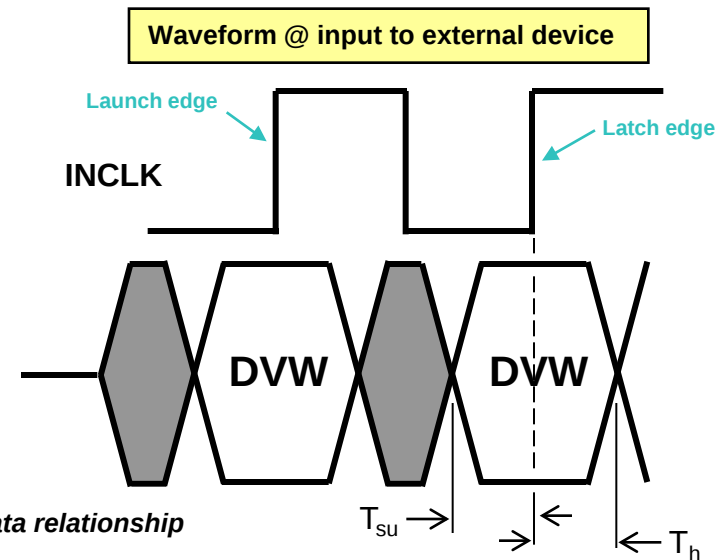
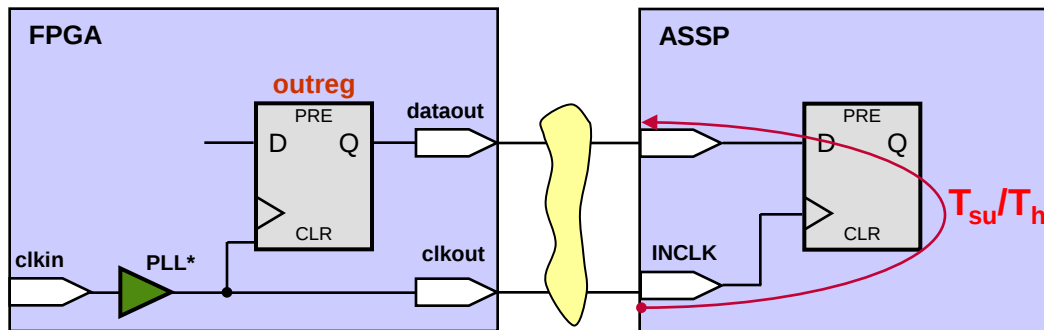
$$\text{input delay min} = T_h$$

Using SDC with Source-Sync Input

- Create clock on clock input port
- Use `set_input_delay` command with reference to virtual clock
 - Same as with synchronous input



SDR Source-Synchronous Output (Center-Aligned)



* The PLL in this example is used to shift output clock to establish an output clock to data relationship

System-Centric:

$$\text{output delay max} = \text{data trace (max)} - \text{clock trace (min)} + T_{su}$$

$$\text{output delay min} = \text{data trace (min)} - \text{clock trace (max)} - T_h$$

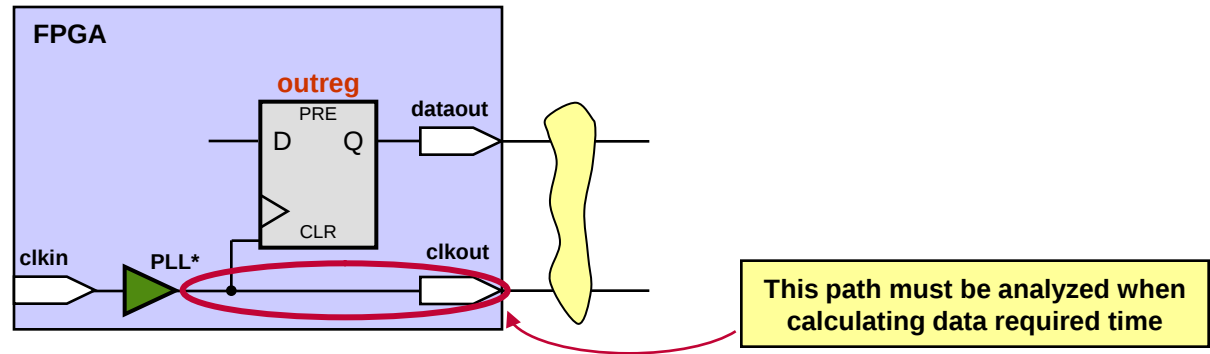
FPGA-Centric:

$$\text{output delay max} = T_{su}$$

$$\text{output delay min} = -T_h$$

Notice output delay minimum is negative

Using SDC with Source-Synch Output



- Must tell timing analyzer to analyze path from clock source to output clock port during analysis
- Use `set_output_delay` command on **dataout** with reference to new generated clock on output port
 - Create generated clock on output clock port (source is PLL output pin)
 - Use `-clock` argument in output delay assignment to associate output clock to output data bus
- Path from PLL output pin to output port still considered unconstrained (clock path viewed as a data path by timing analyzer)
 - Constrain path from PLL pin to output port with false path (described later), `set_min/max_delay`, or `set_output_delay`

Constraining Source-Sync Output Example

```
create_clock -period 5 -name clkin \  
    [get_ports clkin]  
create_generated_clock -name pllclk -divide_by 1 \  
    -source [get_ports clkin]  
    [get_pins inst|altp11_component|pll|clk[0]]
```

```
# Place clock on external clock output  
create_generated_clock -name clkout \  
    -source [get_pins inst|altp11_component|pll|clk[0]] \  
    -divide_by 1 [get_ports clkout]
```

```
# Constrain dataout with an external tsu of 0.5 ns  
# and th of 0.5 ns using clkout as clock  
set_output_delay -clock [get_clocks clkout] \  
    -max 0.500 [get_ports dataout]  
set_output_delay -clock [get_clocks clkout] \  
    -min -0.500 [get_ports dataout]
```

Source Synchronous I/O Timing Summary

	System-centric	FPGA-centric
Input delay (-max)	Data trace (max) – clock trace (min) + (latch edge – launch edge) - T_{SU}	(latch edge – launch edge) – T_{SU}
Input delay (-min)	Data trace (min) – clock trace (max) + T_h	T_h
Input delay –clock	Target virtual input clock	
Output delay (-max)	Data trace (max) – clock trace (min) + T_{SU}	T_{SU}
Output delay (-min)	Data trace (max) – clock trace (min) - T_h	- T_h
Output delay (-clock)	Target generated clock on FPGA output port	

Notes:

- The above only applies to center-aligned only. Also there are many other ways to constrain source synchronous I/Os. For more information, please refer to [AN 433: Constraining and Analyzing Source-Synchronous Interfaces](#)
- May also require some other timing exception constraints (to be discussed later).

Checking I/O Constraints

- Helpful TimeQuest reports to run to verify constraints
- Report SDC
- Report Unconstrained
- Report Ignored Constraints

Report SDC (report_sdc)

■ List SDC constraints applied to netlist

Report

- SDC Assignments
 - Create Clock
 - Create Generated Clock
 - Set Input Delay
 - Set Output Delay
 - Set False Path
 - Set Multicycle Path

Create Clock

SDC Command	Name	Period	Waveform	Targets
1 create_clock	clk_in_100mhz	10.000	{ 0.000 5.000 }	{ get_ports { clk_in_100mhz } }

Base clock

Create Generated Clock

SDC Command	Name	Source	Duty Cycle	Multiply
1 create_generated_clock	c100	[get_pins {inst1 altpll_component pll inclk[0]}]	50.000	1
2 create_generated_clock	c200	[get_pins {inst1 altpll_component pll inclk[0]}]	50.000	2
3 create_generated_clock	c100_out	[get_pins {inst1 altpll_component pll inclk[0]}]	50.000	1
4 create_generated_clock	clkout	[get_pins {inst1 altpll_component pll inclk[0]}]	50.000	1

Generated clocks

Set Input Delay

SDC Command	Flags	Clock Name	Delay	Ports
1 set_input_delay	-max	[get_clocks clk_in_100mhz]	4.500	[get_ports din_a[0]]
2 set_input_delay	-min	[get_clocks clk_in_100mhz]	1.000	[get_ports din_a[0]]
3 set_input_delay	-max	[get_clocks clk_in_100mhz]	4.500	[get_ports din_b[0]]
4 set_input_delay	-min	[get_clocks clk_in_100mhz]	1.000	[get_ports din_b[0]]
5 set_input_delay	-max	[get_clocks clk_in_100mhz]	4.500	[get_ports din_x[0]]
6 set_input_delay	-min	[get_clocks clk_in_100mhz]	1.000	[get_ports din_x[0]]
7 set_input_delay	-max	[get_clocks clk_in_100mhz]	4.500	[get_ports din_y[0]]
8 set_input_delay	-min	[get_clocks clk_in_100mhz]	1.000	[get_ports din_y[0]]
9 set_input_delay	-max	[get_clocks clk_in_100mhz]	4.500	[get_ports din_a[1]]
10 set_input_delay	-min	[get_clocks clk_in_100mhz]	1.000	[get_ports din_a[1]]
11 set_input_delay	-max	[get_clocks clk_in_100mhz]	4.500	[get_ports din_b[1]]
12 set_input_delay	-min	[get_clocks clk_in_100mhz]	1.000	[get_ports din_b[1]]
13 set_input_delay	-max	[get_clocks clk_in_100mhz]	4.500	[get_ports din_x[1]]
14 set_input_delay	-min	[get_clocks clk_in_100mhz]	1.000	[get_ports din_x[1]]
15 set_input_delay	-max	[get_clocks clk_in_100mhz]	4.500	[get_ports din_y[1]]
16 set_input_delay	-min	[get_clocks clk_in_100mhz]	1.000	[get_ports din_y[1]]
17 set_input_delay	-max	[get_clocks clk_in_100mhz]	4.500	[get_ports din_a[2]]
18 set_input_delay	-min	[get_clocks clk_in_100mhz]	1.000	[get_ports din_a[2]]
19 set_input_delay	-max	[get_clocks clk_in_100mhz]	4.500	[get_ports din_b[2]]

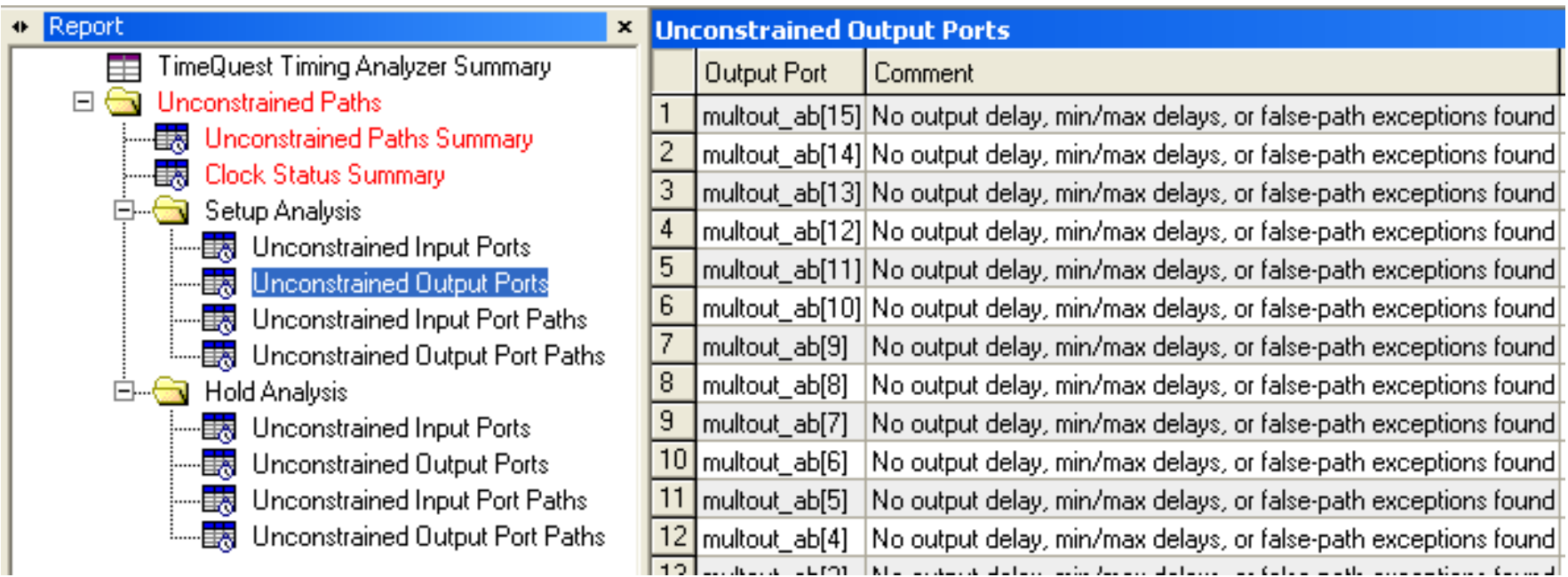
Input delays

Set Output Delay

SDC Command	Flags	Clock Name	Delay	Ports	Source Latency
1 set_output_delay	-max	[get_clocks clkout]	0.500	[get_ports multout_ab[15]]	
2 set_output_delay	-min	[get_clocks clkout]	-0.500	[get_ports multout_ab[15]]	
3 set_output_delay	-max	[get_clocks clkout]	0.500	[get_ports multout_ab[14]]	
4 set_output_delay	-min	[get_clocks clkout]	-0.500	[get_ports multout_ab[14]]	
5 set_output_delay	-max	[get_clocks clkout]	0.500	[get_ports multout_ab[13]]	
6 set_output_delay	-min	[get_clocks clkout]	-0.500	[get_ports multout_ab[13]]	
7 set_output_delay	-max	[get_clocks clkout]	0.500	[get_ports multout_ab[12]]	
8 set_output_delay	-min	[get_clocks clkout]	-0.500	[get_ports multout_ab[12]]	
9 set_output_delay	-max	[get_clocks clkout]	0.500	[get_ports multout_ab[11]]	
10 set_output_delay	-min	[get_clocks clkout]	-0.500	[get_ports multout_ab[11]]	
11 set_output_delay	-max	[get_clocks clkout]	0.500	[get_ports multout_ab[10]]	
12 set_output_delay	-min	[get_clocks clkout]	-0.500	[get_ports multout_ab[10]]	
13 set_output_delay	-max	[get_clocks clkout]	0.500	[get_ports multout_ab[9]]	
14 set_output_delay	-min	[get_clocks clkout]	-0.500	[get_ports multout_ab[9]]	
15 set_output_delay	-max	[get_clocks clkout]	0.500	[get_ports multout_ab[8]]	
16 set_output_delay	-min	[get_clocks clkout]	-0.500	[get_ports multout_ab[8]]	
17 set_output_delay	-max	[get_clocks clkout]	0.500	[get_ports multout_ab[7]]	
18 set_output_delay	-min	[get_clocks clkout]	-0.500	[get_ports multout_ab[7]]	
19 set_output_delay	-max	[get_clocks clkout]	0.500	[get_ports multout_ab[6]]	

Output delays

Report Unconstrained Paths (report_ucp)



	Output Port	Comment
1	multout_ab[15]	No output delay, min/max delays, or false-path exceptions found
2	multout_ab[14]	No output delay, min/max delays, or false-path exceptions found
3	multout_ab[13]	No output delay, min/max delays, or false-path exceptions found
4	multout_ab[12]	No output delay, min/max delays, or false-path exceptions found
5	multout_ab[11]	No output delay, min/max delays, or false-path exceptions found
6	multout_ab[10]	No output delay, min/max delays, or false-path exceptions found
7	multout_ab[9]	No output delay, min/max delays, or false-path exceptions found
8	multout_ab[8]	No output delay, min/max delays, or false-path exceptions found
9	multout_ab[7]	No output delay, min/max delays, or false-path exceptions found
10	multout_ab[6]	No output delay, min/max delays, or false-path exceptions found
11	multout_ab[5]	No output delay, min/max delays, or false-path exceptions found
12	multout_ab[4]	No output delay, min/max delays, or false-path exceptions found
13	multout_ab[3]	No output delay, min/max delays, or false-path exceptions found

- Same report as before used for unconstrained clocks (Clock Status Summary report)
- Setup and Hold Analysis folders list unconstrained I/O ports and paths



Verifying Clocks & I/O Timing

- Use Setup & Hold Summary reports to check worst slack for each clock
 - Obtaining summary reports
 - Use `create_timing_summary` Tcl command
 - TimeQuest folder of Compilation Report
 - Run Report Setup Summary & Report Hold Summary reports from Tasks pane or Reports menu
- For detailed slack/path analysis
 - Run Report Timing from Tasks pane or Constraints menu
 - Use `report_timing` command

Please go to Exercise 4



SDC Timing Constraints

- Clocks
- I/O
- Asynchronous paths 
- False paths
- Multicycle paths
- Delay and skew specifications

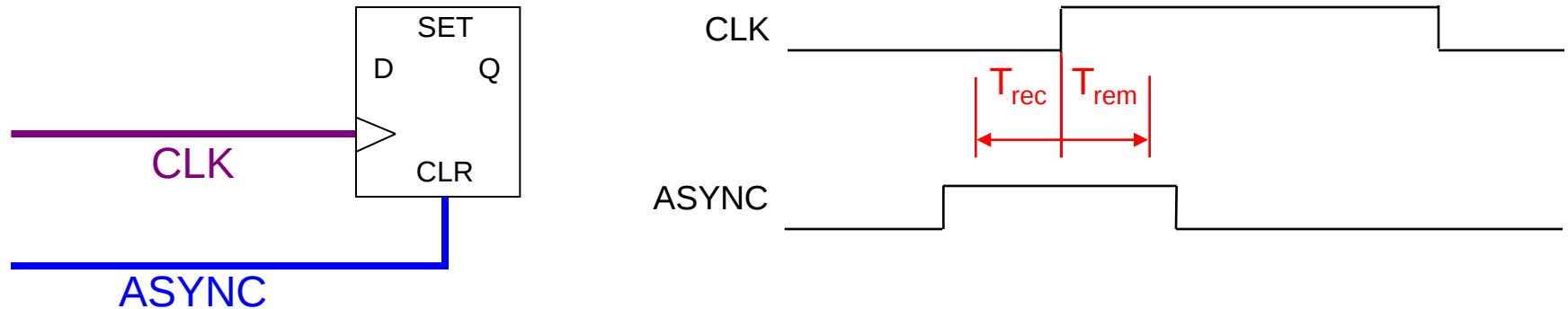
Asynchronous Paths

- Definition: signals that drive asynchronous inputs on internal registers (e.g. clear, preset)
- Used for design initialization & as outputs of control structures
- Must be constrained

TimeQuest TA & Asynchronous Ports

- Asynchronous inputs assumed registered either internally or externally
- Timing analyzer performs recovery (setup) & removal (hold) analysis on asynchronous inputs
 - Required times & arrival times are calculated just like for synchronous data

Recovery & Removal (Review)



Recovery: The minimum time an asynchronous signal can be de-asserted BEFORE clock edge

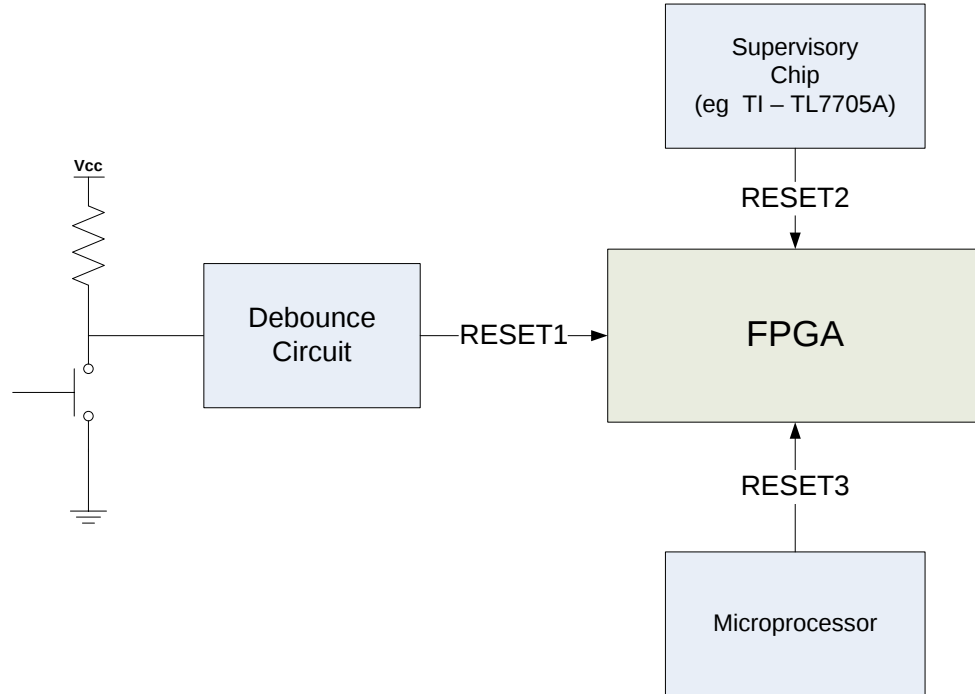
Removal: The minimum time an asynchronous signal can be de-asserted AFTER clock edge

Types of Asynchronous Paths

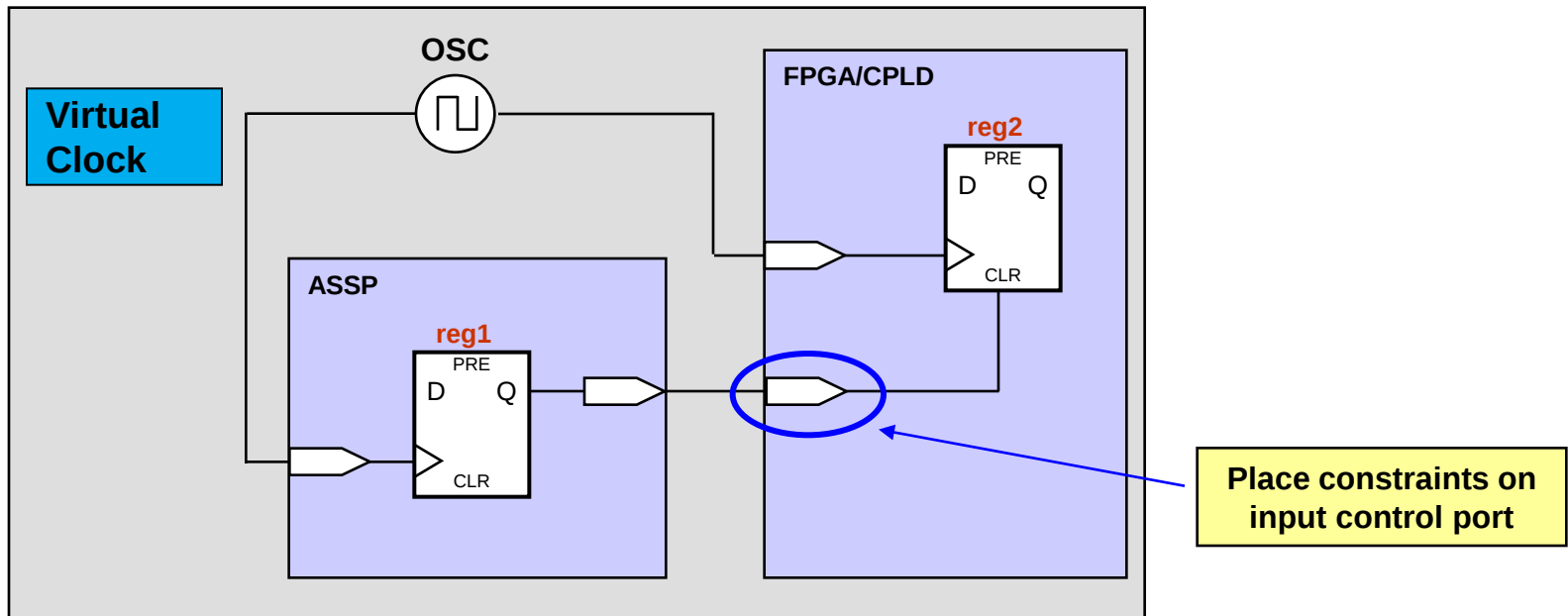
- Externally registered
- Internally registered

Externally Registered

- Control signal generated by a registered output of another device
- Typical sources are:
 - Push button reset thru debounce circuit
 - Supervisory chip
 - Micro-processor GPIO

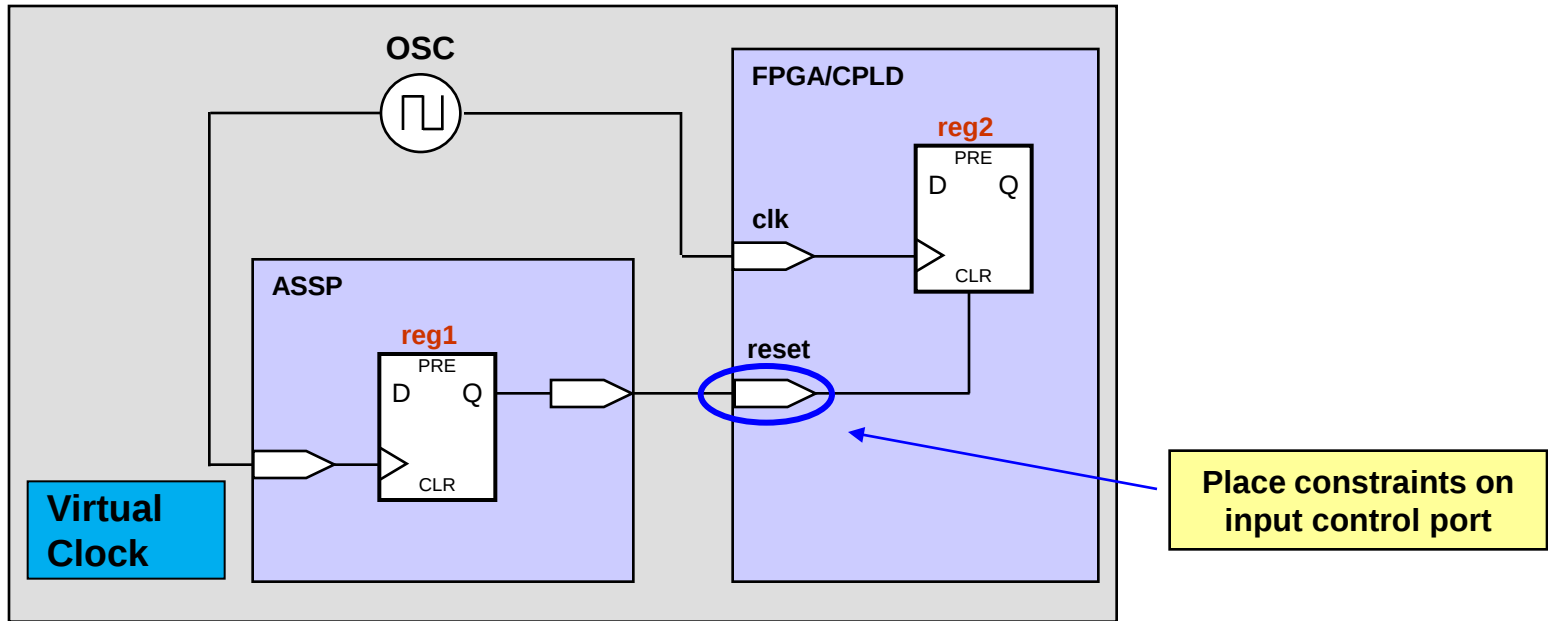


Externally Registered (cont.)



- Apply `set_input_delay -max` & `set_input_delay -min` to input port to constrain

Externally Registered Example

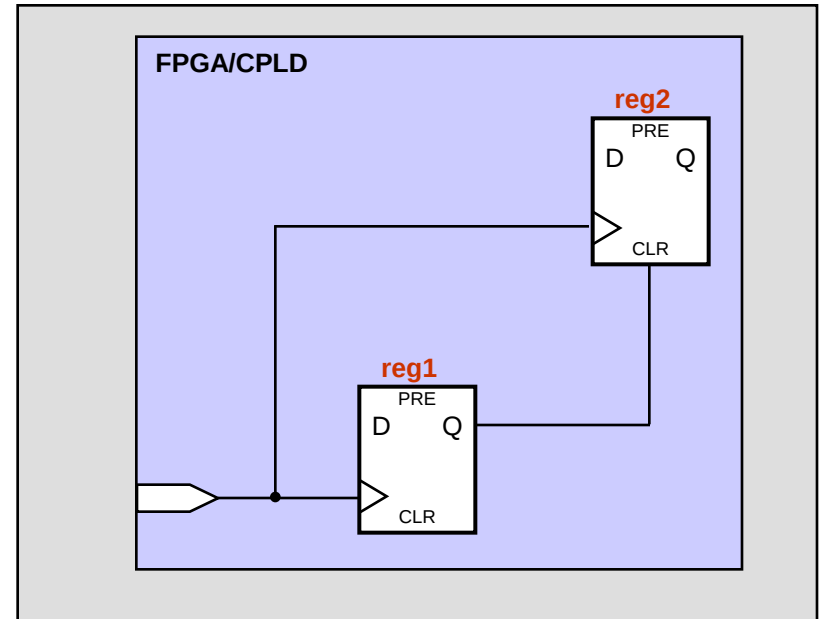


```
create_clock -name clk -period 10 [get_ports clk]
create_clock -name clk_vir -period 10
```

```
set_input_delay -clock [get_clocks clk_vir] -max 5 \
    [get_ports reset]
set_input_delay -clock [get_clocks clk_vir] -min 2 \
    [get_ports reset]
```

Internally Registered

- Control signal generated as output of internal register
- Paths are covered by clock constraints



Checking Asynchronous Control Constraints

- Use same reports as for clocks & I/O
- Externally registered
 - If unconstrained, paths show up as unconstrained input ports & paths
- Internally registered
 - If unconstrained, clock driving register appears as unconstrained

Reporting Asynchronous Control Paths

- Use same methods as clocks & I/O
- Summary reports
 - Use `-recovery|-removal` options with `create_timing_summary`
 - Run Report Recovery/Removal Summary (**Tasks** pane or **Reports** menu)
- Detailed slack/path reports
 - Use `-recovery|-removal` options with `report_timing`
 - Choose Recovery or Removal as Analysis Type when running Report Timing (**Tasks** pane or **Reports** menu)

Example Recovery Report (Ext. Registered)

```
report_timing -from_clock clk_in_100mhz -recovery -npaths 10 \
-detail path_only -panel_name {Recovery (clk_in_100mhz) Detailed}
```

Report

- TimeQuest Timing Analyzer Summary
- SDC File List
- Unconstrained Paths
- Recovery (clk_in_100mhz) Detailed**

Recovery (clk_in_100mhz) Detailed

Slack	From Node	To Node	Launch Clock	Latch Clock
1 3.068	reset	inst3	clk_in_100mhz	c100
2 3.068	reset	inst2	clk_in_100mhz	c100

Path #1: Recovery slack is 3.068

Property	Value
1 From Node	reset
2 To Node	inst3
3 Launch Clock	clk_in_100mhz
4 Latch Clock	c100
5 Data Arrival Time	7.080
6 Data Required Time	10.148
7 Slack	3.068

Path #1: Recovery slack is 3.068

Data Arrival Path

Total	Incr	RF	Type	Fanout	Location	Element
1 0.000	0.000					launch edge time
2 0.000	0.000	R				clock network delay
3 4.500	4.500	R	iExt	1	PIN_H1	reset
4 5.479	0.979	RR	CELL	1	IOC_X0_Y6_N1	reset~combout
5 5.601	0.122	RR	IC	1	CLKCTRL_G1	reset~clkctrlinclk[0]
6 5.601	0.000	RR	CELL	2	CLKCTRL_G1	reset~clkctrloutclk
7 6.324	0.723	RR	IC	1	LCFF_X17_Y12_N17	inst3laclr
8 7.080	0.756	RF	CELL	1	LCFF_X17_Y12_N17	inst3

Data Required Path

Total	Incr	RF	Type	Fanout	Location	Element
1 10.000	10.000					la
2 10.112	0.112	R				c
3 10.148	0.036		uTsu	1	LCFF_X17_Y12_N17	ir

Annotations:

- From node is reset**: Points to the 'From Node' property in the Path Summary table.
- External input delay**: Points to the 'Data Arrival Path' table, specifically the delay from the reset signal to the flip-flop input.
- Destination is async. clear**: Points to the 'inst3laclr' element in the 'Data Arrival Path' table, indicating the asynchronous clear input of the flip-flop.

Example Recovery Report (Int. Registered)

```
report_timing -from_clock c100 -recovery -npaths 10 \
  -detail path_only -panel_name {Recovery (c100) Internal}
```

Recovery (c100) Internal

	Slack	From Node	To Node	Launch Clock	Latch Clock
1	2.320	inst3	inst14[7]	c100	c100
2	2.320	inst3	inst14[4]	c100	c100
3	2.320	inst3	inst14[1]	c100	c100
4	2.413	inst3	inst14[13]	c100	c100
5	2.478	inst3	inst14[3]	c100	c100
6	2.478	inst3	inst14[6]	c100	c100
7	2.478	inst3	inst14[8]	c100	c100
8	2.625	inst3	inst14[0]	c100	c100
9	2.625	inst3	inst14[14]	c100	c100
10	2.625	inst3	inst14[2]	c100	c100

Path #1: Recovery slack is 2.320

Property	Value
1 From Node	inst3
2 To Node	inst14[7]
3 Launch Clock	c100
4 Latch Clock	c100 (INVERTED)
5 Data Arrival Time	2.564
6 Data Required Time	4.884
7 Slack	2.320

Data Arrival Path

Total	Incr	RF	Type	Fanout	Location	Element
1 0.000	0.000					launch edge time
2 0.112	0.112	R				clock network delay
3 0.362	0.250		uTco	1	LCFF_X17_Y12_N17	inst3
4 0.362	0.000	RR	CELL	96	LCFF_X17_Y12_N17	inst3 regout
5 1.890	1.528	RR	IC	1	IOC_X28_Y11_N3	multout_xy[7] areset
6 2.564	0.674	RR	CELL	1	IOC_X28_Y11_N3	inst14[7]

Data Required Path

Total	Incr	RF	Type	Fanout	Location	Element
1 5.000	5.000					latch edge time
2 4.960	-0.040	F				clock
3 4.884	-0.076		uTsu	1	IOC_X28_Y11_N3	inst14

From node is internal register

Destination is async. reset

© 2010 Altera Corporation—Confidential

ALTERA, ARRIA, CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS & STRATIX are Reg. U.S. Pat. & Tm. Off. and Altera marks in and outside the U.S.

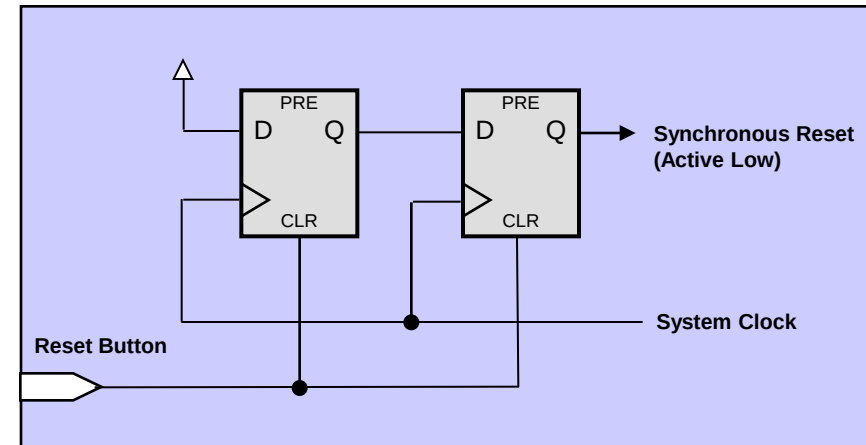


What about truly asynchronous control inputs?

- **BAD IDEA to use it directly!!!!**
- Solution: Synchronize inputs with internal clock
 - Input may then become false path (discussed in next section)

- But if you must...
 - Use `set_max_delay` & `set_min_delay` to constrain paths

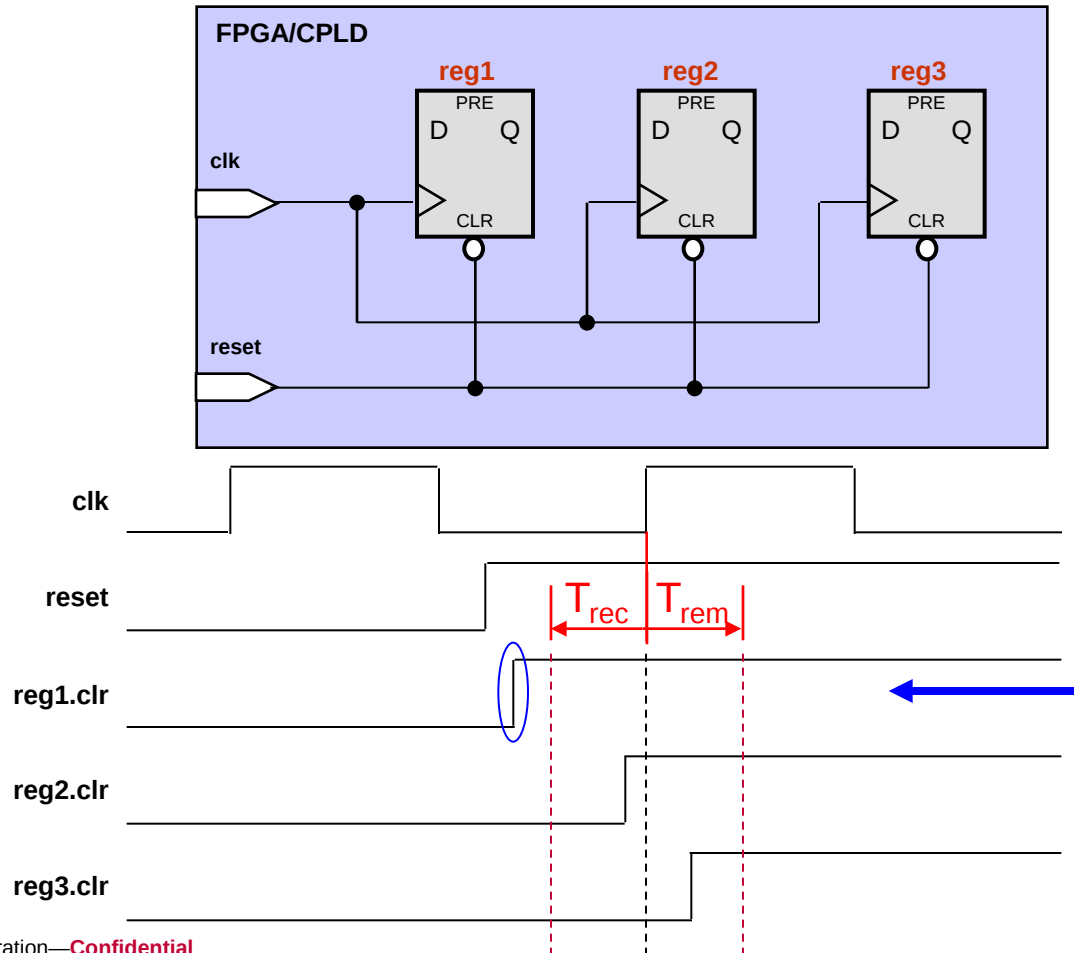
OR



- Use `set_input_delay` of 0 on input
 - Asynchronous signal valid as soon as it arrives on input port

Need More Proof?

For example, these state machine registers should all be de-asserted together, but...



Due to routing delay (skew) of the clear signal, only **reg1** comes out of reset correctly, the rest may de-assert on the next clock cycle. This could mean starting in the wrong state (or even an illegal state).

SDC Timing Constraints

- Clocks
- I/O
- Asynchronous paths
- False paths 
- Multicycle paths
- Delay and skew specifications



Timing Exceptions: False Paths

■ Logic-based

- Paths not relevant during normal circuit operation
- e.g. Test logic, static or quasi-static registers

■ Timing-based

- Paths intentionally not analyzed by designer
- e.g. Bridging asynchronous clock domains using synchronizer circuits

■ Must be marked by constraint to tell TimeQuest to ignore them

Two Methods to Create False Paths

■ `set_false_path` command

- Use when particular nodes are involved
- Examples
 - All paths from an input pin to a set of registers
 - All paths from a register to another clock domain

■ `set_clock_groups` command

- Use when just clock domains are involved

set_false_path Command

- Indicates paths that should be ignored during fitting and timing analysis

- Options

- `[-fall_from <clocks>]`

- `[-rise_from <clocks>]`

- `[-from <names>]`

- `[-through <names>]`

- `[-to <names>]`

- `[-fall_to <clocks>]`

- `[-rise_to <clocks>]`

- `[-setup]`

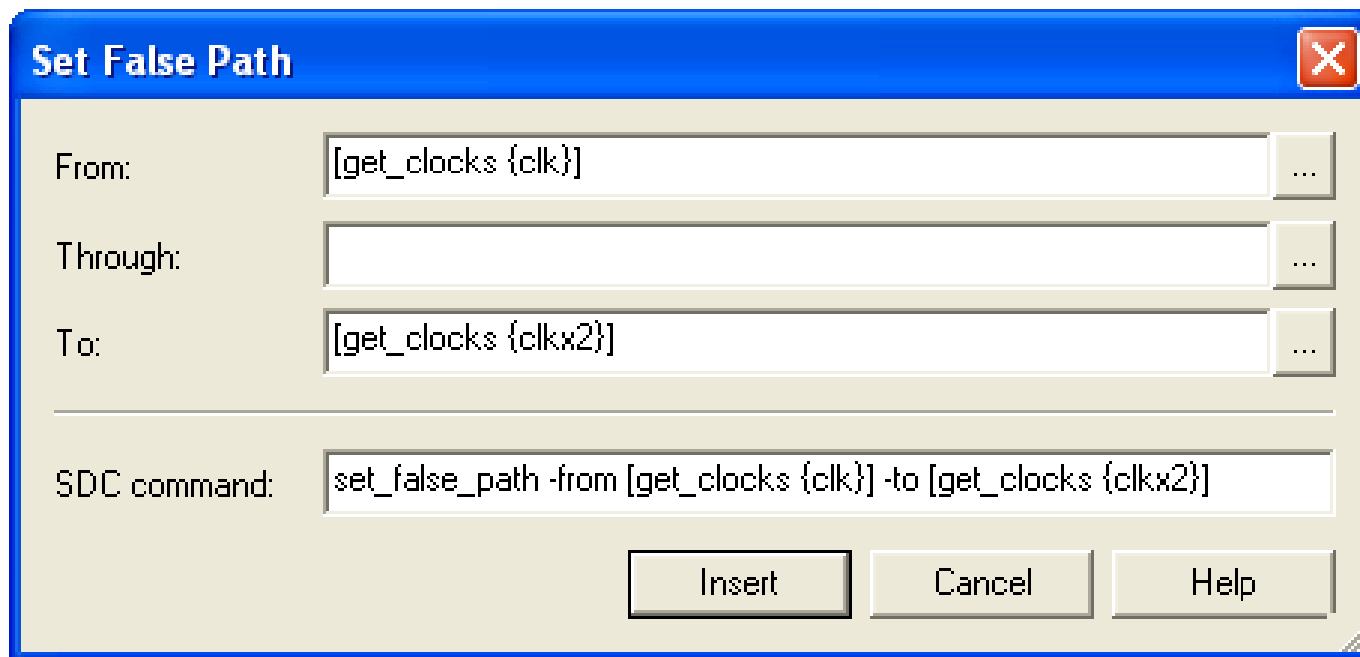
- `[-hold]`

- `<targets>`

set_false_path Notes

- -from & -to: Use to specify source & target nodes
 - Target nodes can be clocks, registers, ports, pins or cells
 - For registers, -from should be source register clock pin
 - Specify a clock name to constrain all paths going into or out of its domain
 - Constrains both rising and falling edge clock transitions
 - More efficient than specifying individual nodes
- -rise_from & -fall_from: Use to indicate clocks for the source node & whether constraint is for a rising or falling edge clock transition; *not in GUI*
- -rise_to & -fall_to: Use to indicate clocks for destination node & direction of transition; *not in GUI*
- -setup & -hold: Use to apply false paths to only setup/recovery or hold/removal analysis; *not in GUI*

Set False Path (GUI)



The image shows a GUI dialog box titled "Set False Path". It has a blue title bar with a close button (X) in the top right corner. The dialog contains three input fields: "From:" with the text "[get_clocks {clk}]", "Through:" which is empty, and "To:" with the text "[get_clocks {clkx2}]". Each of these three fields has a small button with three dots to its right. Below these fields is a horizontal line, and then an "SDC command:" label followed by a text area containing the command "set_false_path -from [get_clocks {clk}] -to [get_clocks {clkx2}]". At the bottom of the dialog are three buttons: "Insert", "Cancel", and "Help".

Set False Path

From: [get_clocks {clk}] ...

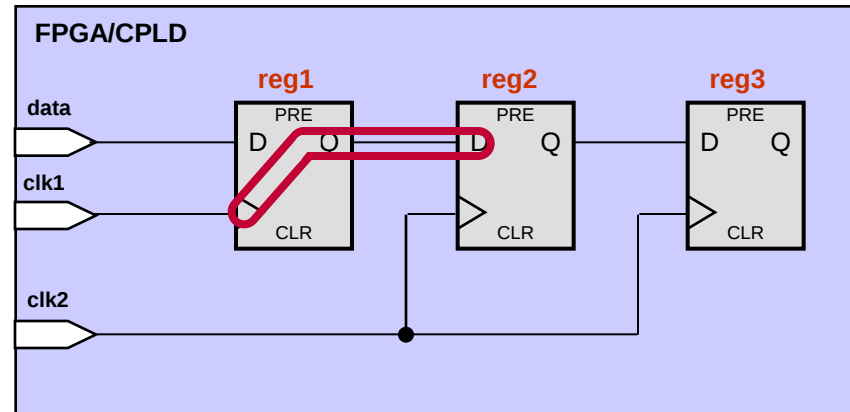
Through: ...

To: [get_clocks {clkx2}] ...

SDC command: set_false_path -from [get_clocks {clk}] -to [get_clocks {clkx2}]

Insert Cancel Help

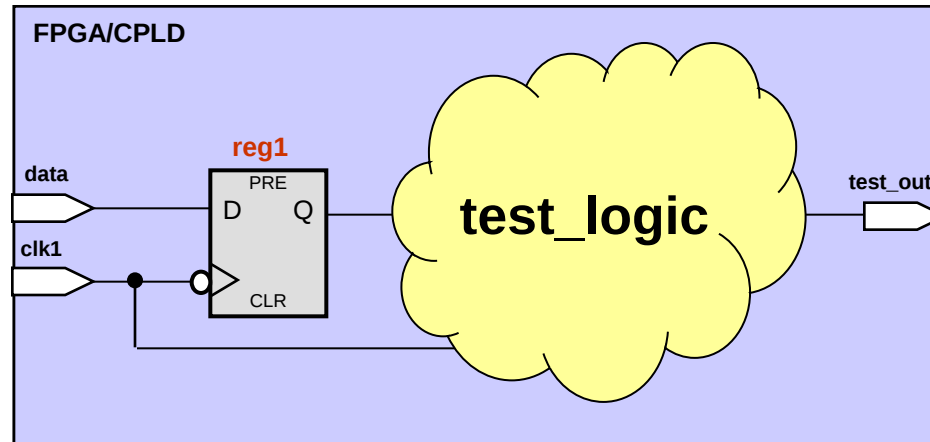
False Path Example 1



Simple synchronizer
circuit between two
asynchronous clock
domains

```
set_false_path -from [get_pins reg1|clk] \  
               -to [get_pins reg2|datain]
```

False Path Example 2



```
set_false_path -fall_from clk1 \  
               -to [get_pins test_logic|*|datain]  
  
set_false_path -from [get_pins test_logic|*|clk] \  
               -to [get_pins test_logic|*|datain]  
  
set_false_path -from [get_pins test_logic|*|clk] \  
               -to [get_ports test_out]
```

set_clock_groups Command

- Tells Fitter and timing analyzer to ignore **ALL** paths between specified clock domains
 - Great for clock muxes
 - Equivalent to setting false paths (-from & -to) on all paths between domains
- Options
 - `[-asynchronous | -exclusive]`
 - `-group <clock name>`
 - `-group <clock_name>`
 - `[-group <clock name>]...`

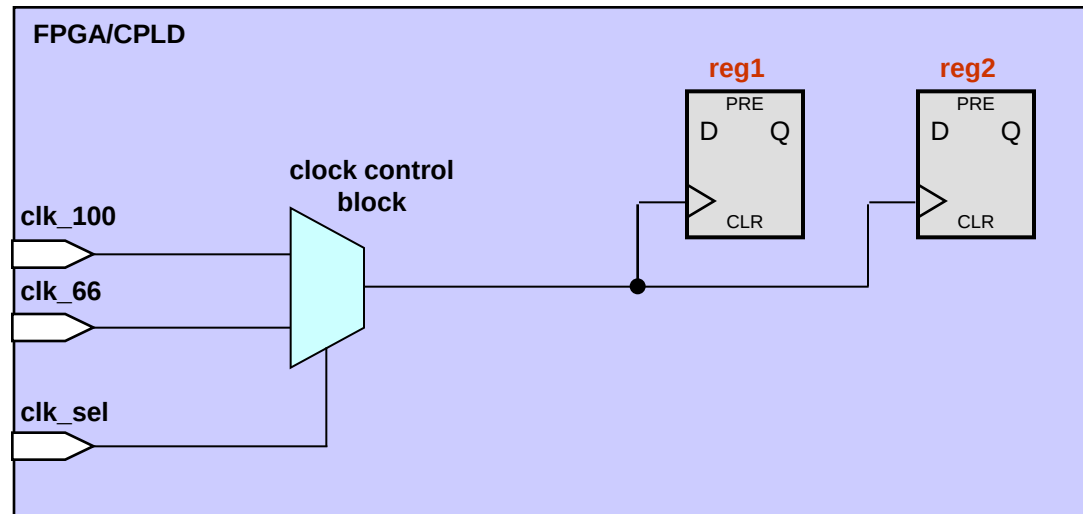
set_clock_groups Notes

- -group: each group of clock names is mutually exclusive to other clock groups
 - e.g. `set_clock_groups -exclusive \`
`-group {clkA clkB} -group {clkC clkD}`
- Additional argument*:
 - -asynchronous: no phase relationship, but clocks active at the same time
 - -exclusive: clocks *not* active at the same time
 - Example: clock muxes

*Notes:

- Need at least one of the two arguments (-asynchronous or -exclusive)
- TimeQuest Timing Analyzer treats both options as if they were the same
- With one -group argument, TimeQuest Timing Analyzer cut analysis of ALL paths to that group of clocks.

Clock Mux Example 1

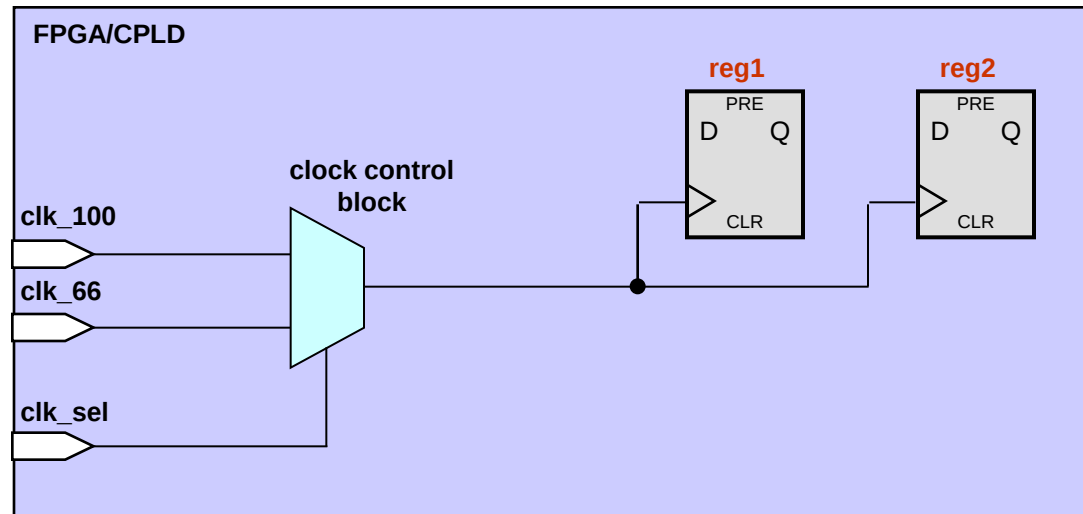


```
create_clock -period 10.0 [get_ports clk_100]
create_clock -period 15.0 [get_ports clk_66]

set_clock_groups -exclusive -group {clk_100} -group {clk_66}

# Since clocks are muxed, timing analyzer should not analyze
# cross-domain paths as only one clock will be driving the
# registers at any one time.
```

Clock Mux Example 1 (Alternative)



```
create_clock -period 10.0 [get_ports clk_100]
create_clock -period 15.0 [get_ports clk_66]

set_false_paths -from [get_clocks clk_100] -to [get_clocks clk_66]
set_false_paths -from [get_clocks clk_66] -to [get_clocks clk_100]

# For an equivalent constraint using false paths, you must
# consider paths going both directions
```

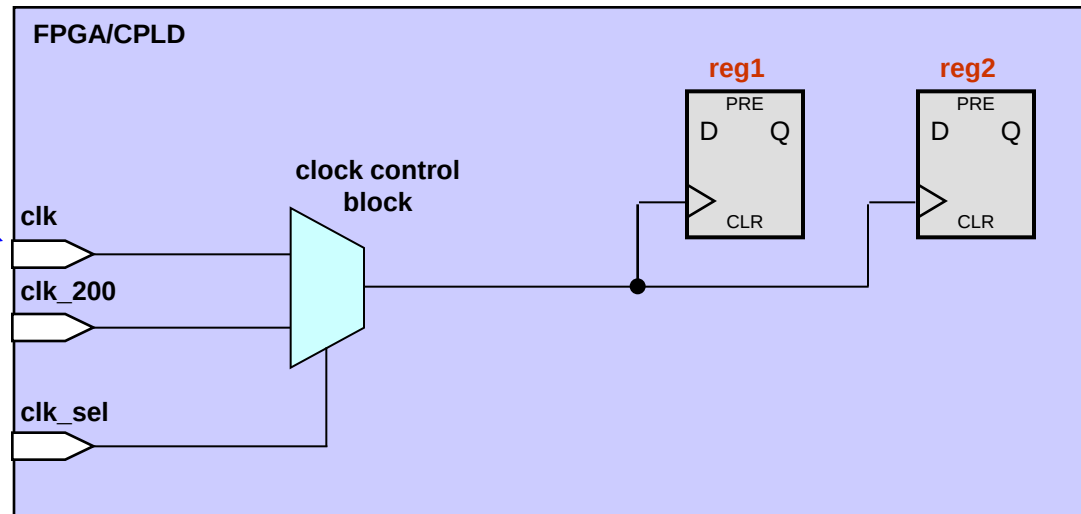
© 2010 Altera Corporation—**Confidential**

ALTERA, ARRIA, CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS & STRATIX are Reg. U.S. Pat. & Tm. Off.
and Altera marks in and outside the U.S.



Clock Mux Example 2

Applying two clock settings
to same input port

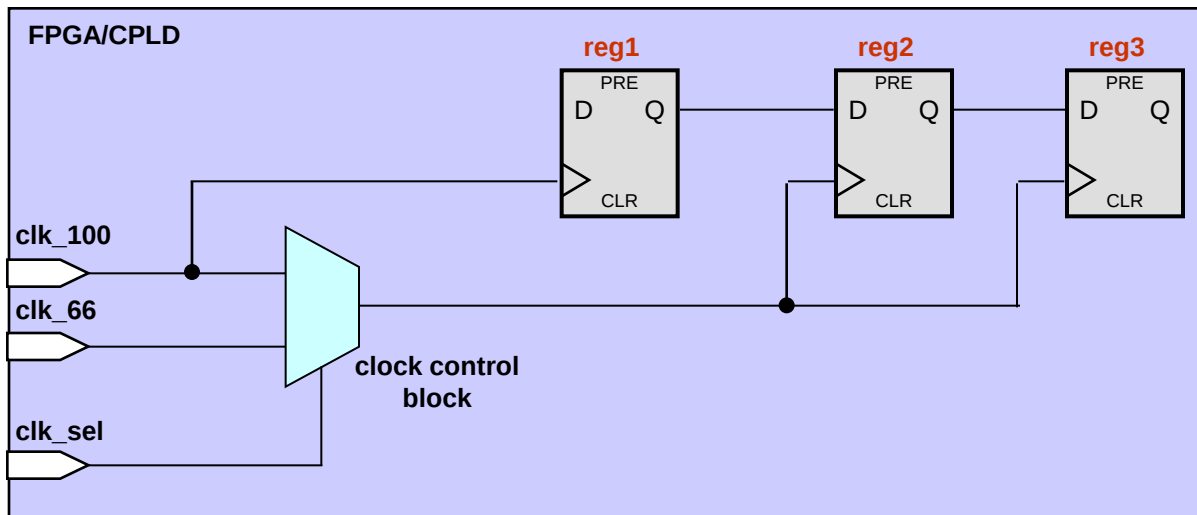


```
create_clock -name clk_100 -period 10.0 [get_ports clk]
create_clock -name clk_66 -period 15.0 [get_ports clk] -add
create_clock -period 5.0 [get_ports clk_200]
```

```
set_clock_groups -exclusive -group {clk_100} \
    -group {clk_66} -group {clk_200}
```

```
# As before, never will more than one clock be driving all
# registers
```

Clock Mux Example 3



```
create_clock -period 10.0 [get_ports clk_100]
create_clock -period 15.0 [get_ports clk_66]

create_generated_clock -name clkmux_100 -source clk_100 \
    -multiply_by 1 [get_pins clkmux|clkout]
create_generated_clock -name clkmux_66 -source clk_66 \
    -multiply_by 1 [get_pins clkmux|clkout] -add

set_clock_groups -exclusive -group {clkmux_100} -group {clkmux_66}

# Since clk_100 is also feeding into the core, now you need to make generated
# clocks on the mux outputs and use them for the clock groups
```



Verifying False Paths & Groups

■ False paths

- Create timing exceptions report
 - `report_exceptions`
 - **Tasks** pane or **Reports** menu: **Report Exceptions**

■ Clock groups

- Check clock transfers to ensure no paths are returned
 - `report_clock_transfers`
 - **Tasks** pane or **Reports** menu: **Report Clock Transfers**

SDC Timing Constraints

- Clocks
- I/O
- Asynchronous paths
- False paths
- Multicycle paths 
- Delay and skew specifications

Timing Exceptions: Multicycle Paths

- Paths requiring more than one cycle for data to propagate
- Causes timing analyzer to select another latch or launch edge
- Designer specifies number of cycles to move edge
- Logic *must* be designed to work this way
 - Constraint informs timing analysis how logic is supposed to function



Other Instances to Use Multicycle Paths

- Design does not require single cycle to transfer data (non-critical paths)
 - Otherwise needlessly over-constrain paths
- Clocks are integer multiples of each other with or without offset
 - Demonstrated in Exercise 5
- Clock enables ensuring register(s) not sampling data every clock edge

Multicycle Types

Type	Clock	Timing Check	Shorthand
End Multicycle Setup	Destination	Setup	EMS
End Multicycle Hold	Destination	Hold	EMH
Start Multicycle Setup	Source	Setup	SMS
Start Multicycle Hold	Source	Hold	SMH

■ Destination

- Constraint based on destination clock edges
- Moves latch edge backward (later in time) to relax required setup/hold time
- Used in most multicycle situations

■ Source

- Constraint based on source clock edges
- Moves launch edge forward (earlier in time) to relax required setup/hold time
- Useful when source clock is at higher frequency than destination

■ Setup

- Increases the number of cycles for setup analysis
- Default is 1

■ Hold

- Increases the number of cycles for hold analysis
- Default is 0

set_multicycle_path Command

- Indicates by how many cycles the required time (setup or hold) should be extended from defaults

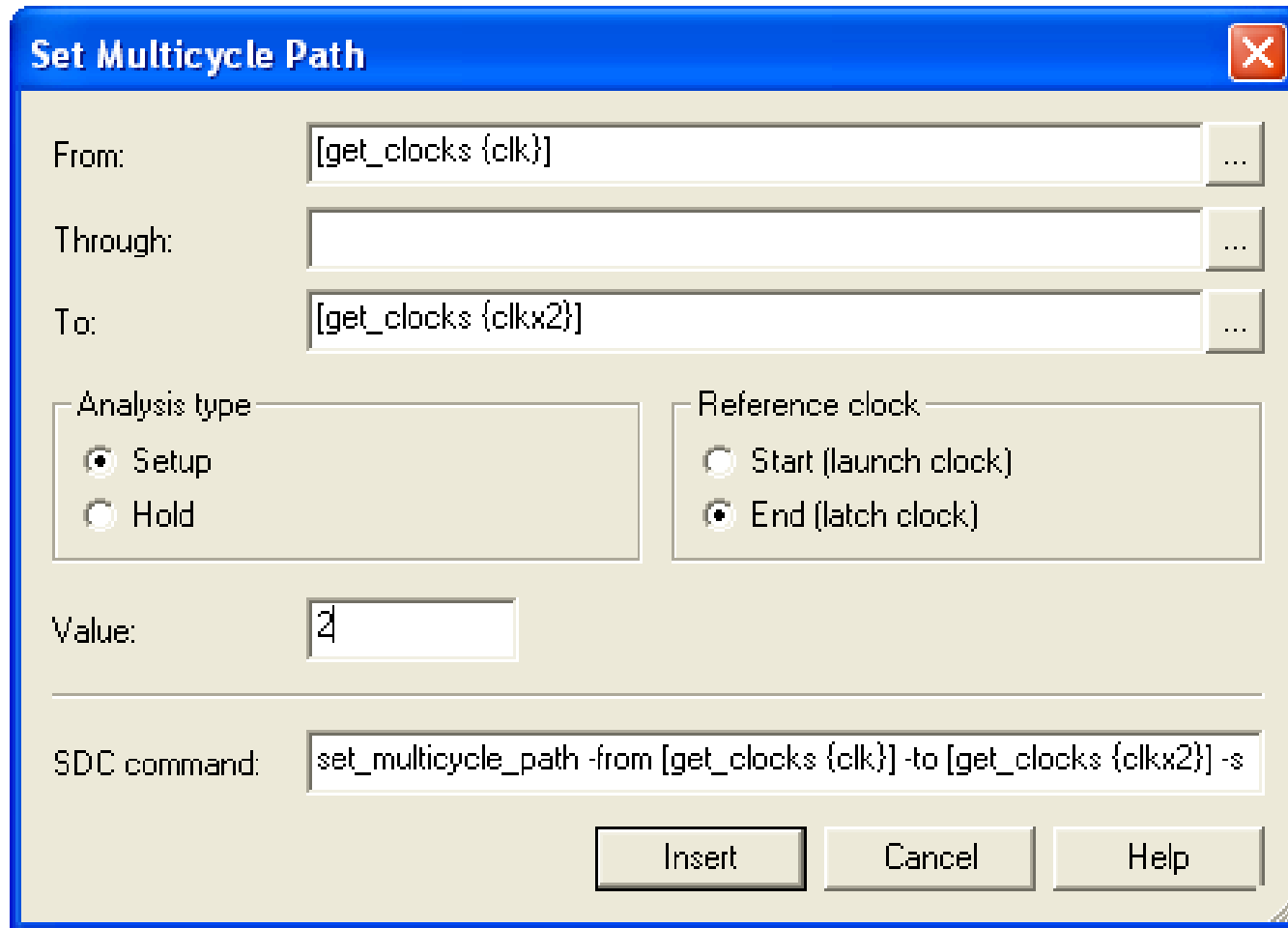
- Options

- `[-start | -end]`
 - `[-setup | -hold]`
 - `[-fall_from <clocks>]`
 - `[-rise_from <clocks>]`
 - `[-from <names>]`
 - `[-through <names>]`
 - `[-to <names>]`
 - `[-fall_to <clocks>]`
 - `[-rise_to <clocks>]`
 - `<targets>`
 - `<value>`

set_multicycle_path Notes

- -start: Use to select a source multicycle
- -end: Use to select a destination multicycle (default)
- -setup | -hold: Specifies if the multicycle value is applied to the setup or hold calculation
- <value>: Cycle multiplier - Number of edges by which to extend analysis
- All other options behave similar to set_false_path options

Set Multicycle Path (GUI)



The dialog box titled "Set Multicycle Path" has a blue title bar with a close button. It contains several input fields and two groups of radio buttons. The "From:" field contains "[get_clocks {clk}]", the "Through:" field is empty, and the "To:" field contains "[get_clocks {clkx2}]". Each field has a browse button ("..."). The "Analysis type" group has "Setup" selected. The "Reference clock" group has "End (latch clock)" selected. The "Value:" field contains "2". The "SDC command:" field contains the text "set_multicycle_path -from [get_clocks {clk}] -to [get_clocks {clkx2}] -s". At the bottom are "Insert", "Cancel", and "Help" buttons.

Set Multicycle Path

From: [get_clocks {clk}] ...

Through: ...

To: [get_clocks {clkx2}] ...

Analysis type

- ☒ Setup
- ☐ Hold

Reference clock

- ☐ Start (launch clock)
- ☒ End (latch clock)

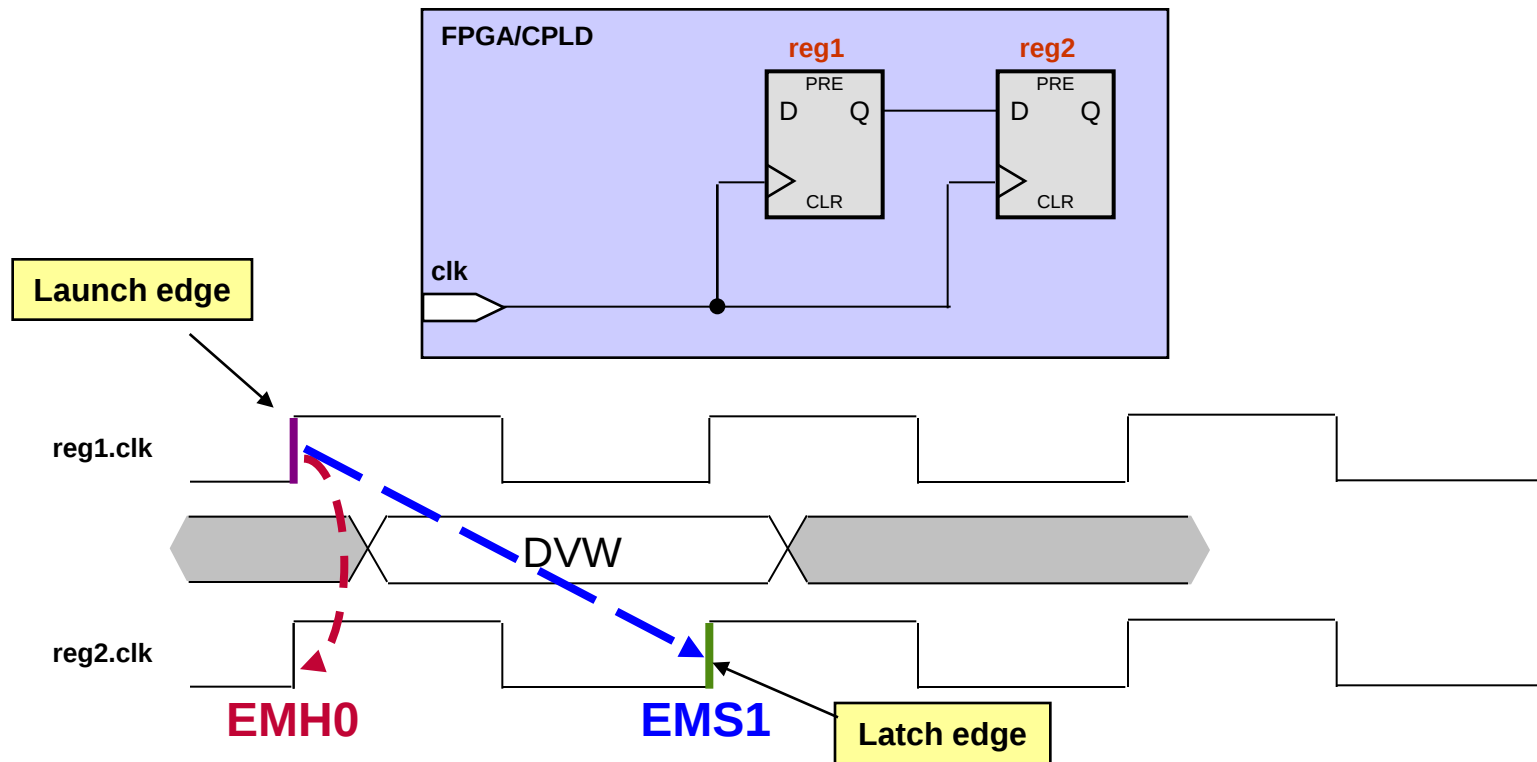
Value: 2

SDC command: set_multicycle_path -from [get_clocks {clk}] -to [get_clocks {clkx2}] -s

Insert Cancel Help

Understanding Multicycle (1)

Standard single-cycle register transfer



— — — • Multicycle Setup = 1 (Default)

- - - - Multicycle Hold = 0 (Default)*

**Default hold edge is one edge before/after setup edge*

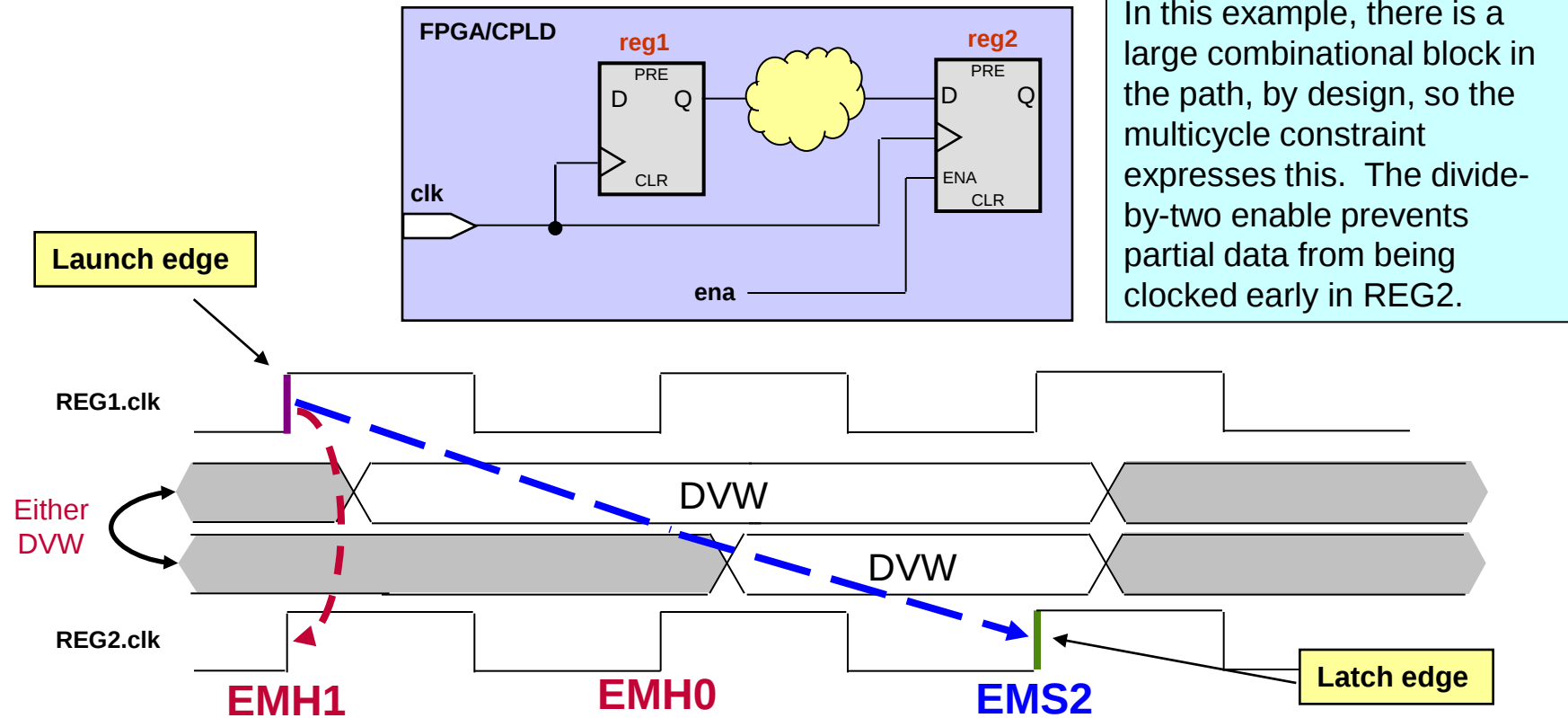
© 2010 Altera Corporation—**Confidential**

ALTERA, ARRIA, CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS & STRATIX are Reg. U.S. Pat. & Tm. Off. and Altera marks in and outside the U.S.



Understanding Multicycle (2)

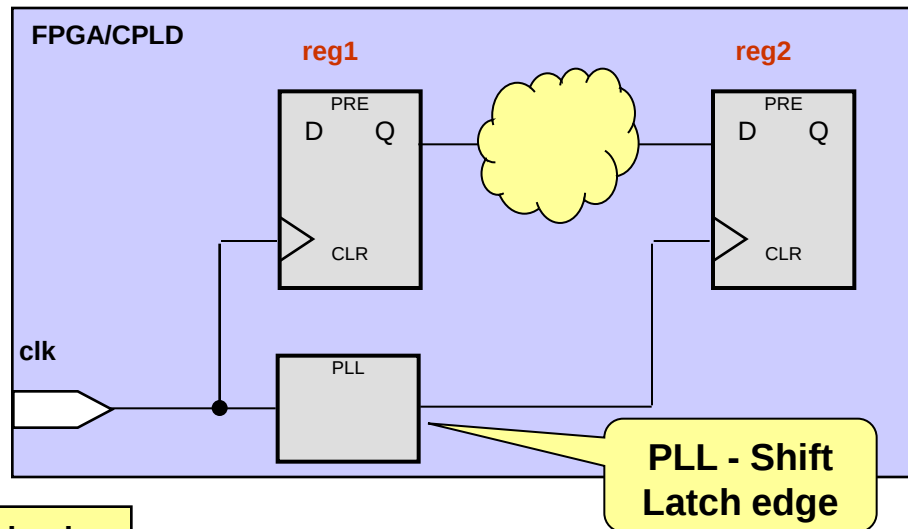
Change to a *two cycle setup; two cycle hold* transfer



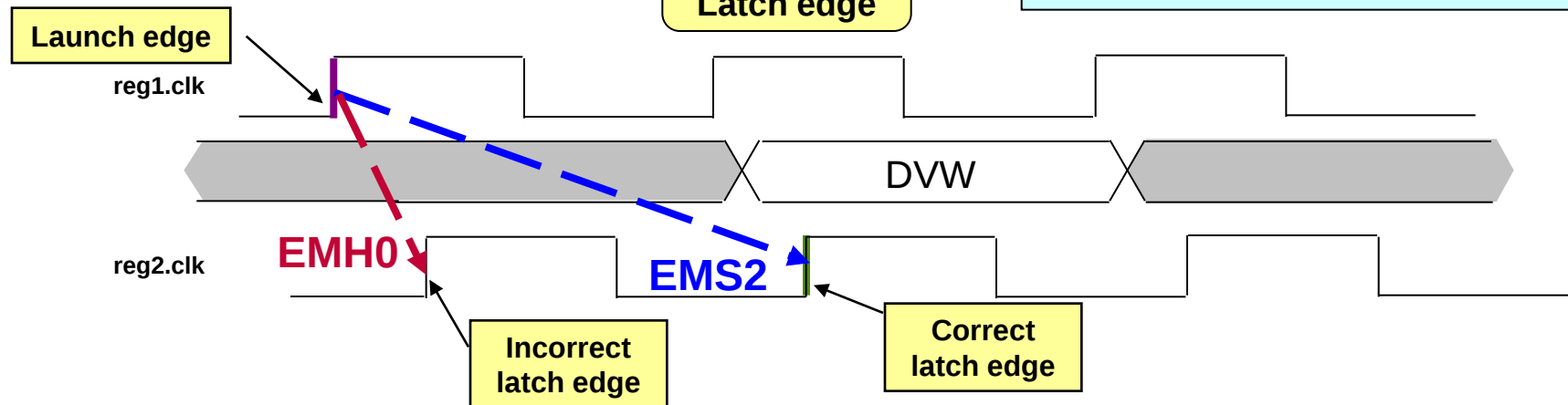
```
set_multicycle_path -from [get_pins reg1|clk] -to [get_pins reg2|datain] -setup 2
set_multicycle_path -from [get_pins reg1|clk] -to [get_pins reg2|datain] -hold 1
```

Understanding Multicycle (3)

Change to a *two cycle setup*; *single cycle hold* transfer



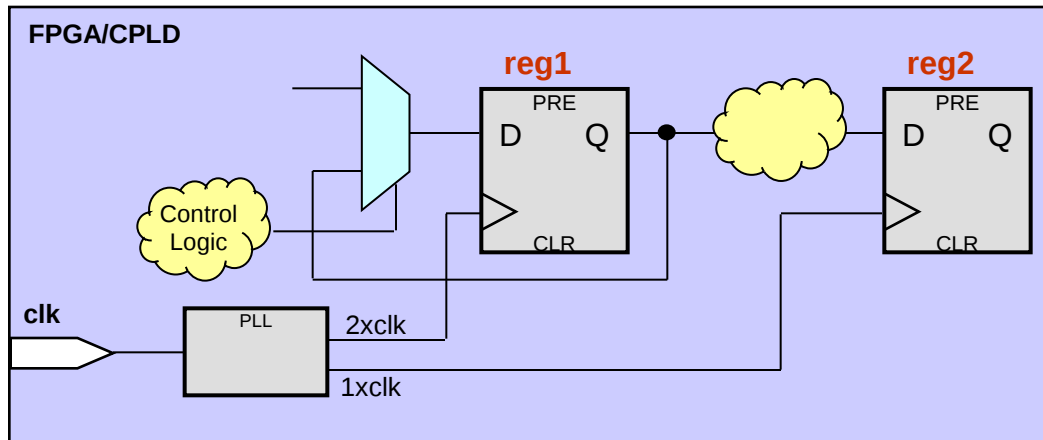
In this example, there is too much logic in the cloud between registers. In order to meet the setup restriction, a PLL is used to shift the clock edge in time. Reg2 doesn't need a clock enable because we are merely shifting the latch clock edge; the data is naturally shifted because of the cloud of logic. Reg2 will latch data every shifted clock.



```
set_multicycle_path -from [get_pins reg1|clk] -to [get_pins reg2|datain] -setup 2
```

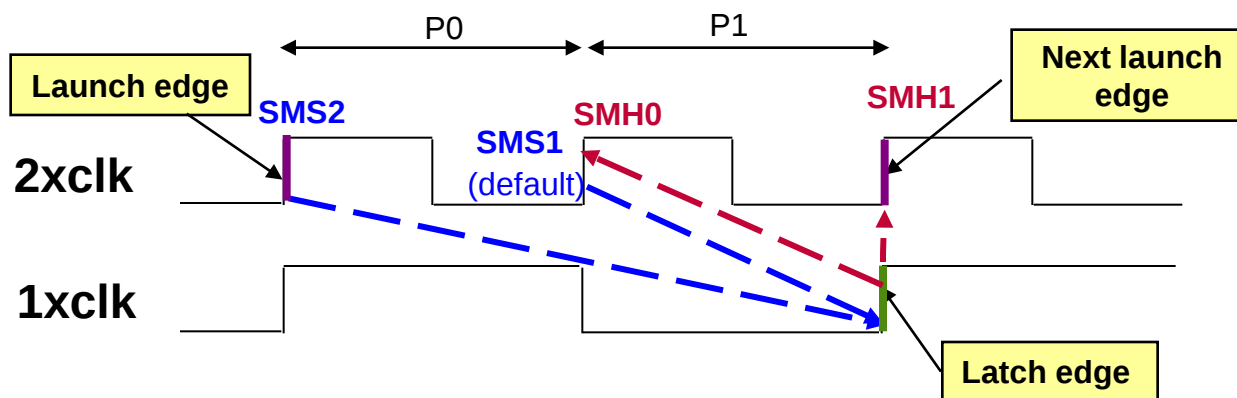
Understanding Multicycle (4)

Move the launch edge



In this example, a register clocked by a 2x clock feeds a register clocked with a 1x clock. The launch register only changes value in phase P0 only and never in P1.

Rather than moving the latch edge, we are going to move the launch edge instead, with the -start argument.



```
set_multicycle_path -from [get_pins reg1|clk] -to [get_pins reg2|datain] -start -setup 2
set_multicycle_path -from [get_pins reg1|clk] -to [get_pins reg2|datain] -start -hold 1
```

Reporting Multicycles

Report Timing: No multicycle

Command Info Summary of Paths

	Slack	From Node	To Node	Launch Clock	Latch Clock
1	-2.357	y_regtwo[2]	...OBSERVABLEDATAB_REGOUT2	c100	c200

No Multicycle

Path #1: Setup slack is -2.357 (VIOLATED)

Path Summary Statistics Data Path Waveform

Data Arrival Path

	Total	Incr	RF	Type	Fanout	Location	Element
1	0.000	0.000					launch edge time
2	0.091	0.091	R				clock network delay
3	0.341	0.250		uTco	1	LCFF_X27_Y7_N7	y_regtwo[2]
4	0.341	0.000	RR	CELL	1	LCFF_X27_Y7_N7	y_regtwo[2]regout
5	0.341	0.000	RR	IC	1	LCCOMB_X27_Y7_N6	inst24[inst[2]]datac
6	0.664	0.323	RR	CELL	1	LCCOMB_X27_Y7_N6	inst24[inst[2]]combout
7	0.909	0.245	RR	IC	1	LCCOMB_X27_Y7_N0	inst24[inst11[2]]datad
8	1.058	0.149	RR	CELL	1	LCCOMB_X27_Y7_N0	...24[inst11[2]]combout

Data Required Path

	Total	Incr	RF	Type	Fanout	Location	Element
1	5.000	5.000					latch edge time
2	5.136	0.136	R				clock network delay
3	5.089	-0.047		uTsu	14	DSPMULT_X16_Y10_N0	...BLEDATAB_REGOUT2

Reporting Multicycles

Report Timing: set_multicycle_path

Command Info | Summary of Paths

	Slack	From Node	To Node	Launch Clock	Latch Clock
1	2.643	y_regtwo[2]	...OBSERVABLEDATAB_REGOUT2	c100	c200

Same path with Setup Multicycle = 2

Path #1: Setup slack is 2.643

Path Summary | Statistics | Data Path | Waveform

Data Arrival Path

	Total	Incr	RF	Type	Fanout	Location	Element
1	0.000	0.000					launch edge time
2	0.091	0.091	R				clock network delay
3	0.341	0.250		uTco	1	LCFF_X27_Y7_N7	y_regtwo[2]
4	0.341	0.000	RR	CELL	1	LCFF_X27_Y7_N7	y_regtwo[2]regout
5	0.341	0.000	RR	IC	1	LCCOMB_X27_Y7_N6	inst24[inst[2]]datac
6	0.664	0.323	RR	CELL	1	LCCOMB_X27_Y7_N6	inst24[inst[2]]combout
7	0.909	0.245	RR	IC	1	LCCOMB_X27_Y7_N0	inst24[inst11[2]]datad
8	1.058	0.149	RR	CELL	1	LCCOMB_X27_Y7_N0	inst24[inst11[2]]combout
9	1.303	0.245	RR	IC	1	LCCOMB_X27_Y7_N26	inst24[inst12[2]]datad
10	1.452	0.149	RR	CELL	1	LCCOMB_X27_Y7_N26	inst24[inst12[2]]combout
11	1.700	0.248	RR	IC	1	LCCOMB_X27_Y7_N10	inst24[inst13[2]]datad

Data Required Path

	Total	Incr	RF	Type	Fanout	Location	Element
1	10.000	10.000					latch edge time
2	10.136	0.136	R				clock network delay
3	10.089	-0.047		uTsu			...LEDATAB_REGOUT2

Latch edge extended by one destination clock cycle

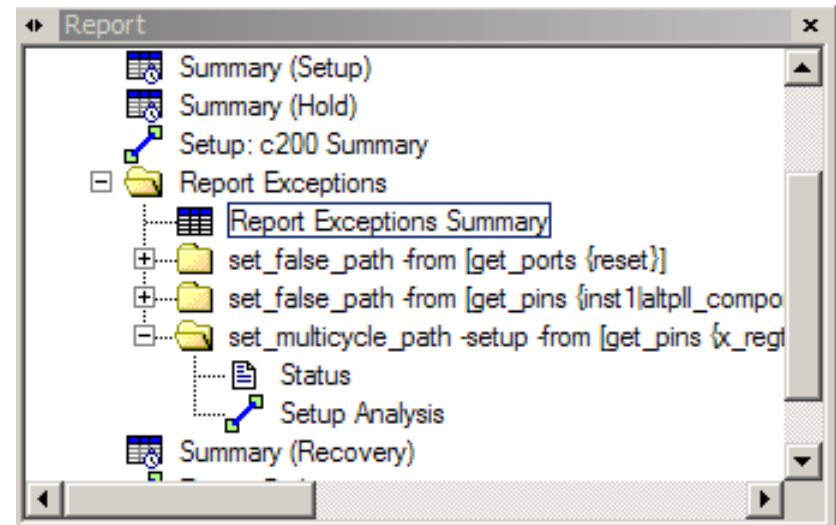
Path #1: Setup slack is 2.643

Path Summary | Statistics | Data Path | Waveform

	Property	Value
1	From Node	y_regtwo[2]
2	To Node	...ABLEDATAB_REGOUT2
3	Launch Clock	c100
4	Latch Clock	c200
5	Multicycle - Setup End	2
6	Data Arrival Time	7.446
7	Data Required Time	10.089
8	Slack	2.643

Report Exceptions

- Provide information specifically about timing exceptions (false paths and multicycle paths)
 - report_exceptions
 - From **Tasks** pane or **Report** menu



Report Exceptions Summary						
	Status	Exception	Setup Slack	Hold Slack	Recovery Slack	Removal Slack
1	Complete	set_false_path from [get_ports {reset}]	Invalid	n/a	n/a	n/a
2	Complete	...t1altpll_component[p1]clk[2]] to [get_ports {clkout}]	Invalid	n/a	n/a	n/a
3	Complete	...tup from [get_pins {x_regtwo}clk y_regtwo}clk]] 2	2.961	n/a	n/a	n/a

SDC Timing Constraints

- Clocks
- I/O
- Asynchronous paths
- False paths
- Multicycle paths
- Delay and skew specifications 

Absolute Delays

- Applies a timing value to a particular path
- Overrides the current setup/hold information for the path derived from clock and I/O constraints
- Apply `set_max_delay` & `set_min_delay` constraints to paths

Absolute Delay Example

- Specify an input port-to-register or register-to-output port constraint without using input & output delays
- Use `-rise_from/-fall_from` & `-rise_to/-fall_to` (*not in GUI*) to restrict timing value to only registers responding to a rising or falling edge transition

Ex. DDR input

```
# Apply a 2ns max delay for an input port only to nodes clocked by  
# the rising edge of clock CLK  
set_max_delay -from [get_ports in*] -rise_to [get_clocks CLK] 2.000
```

Specify skew

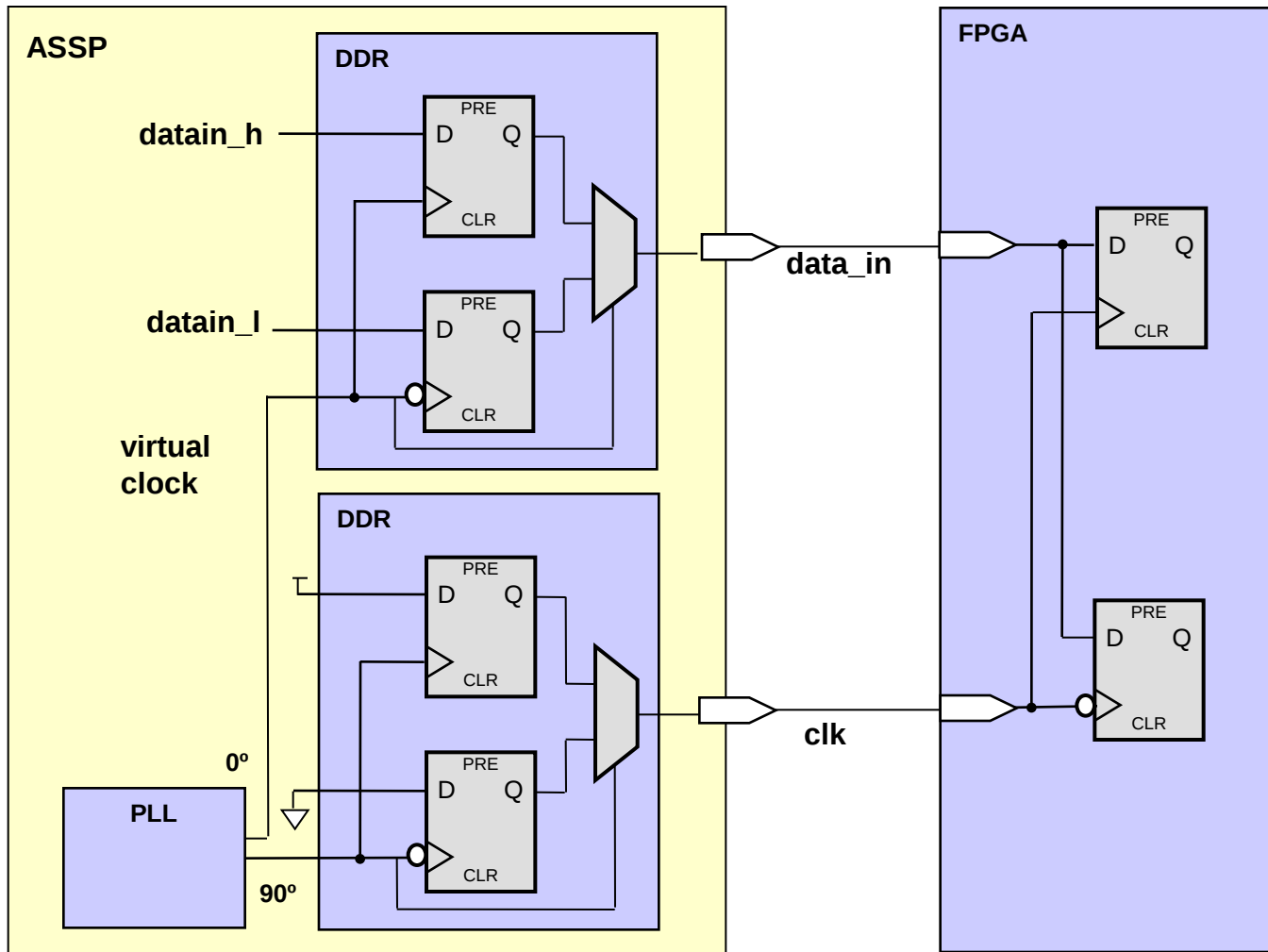
- `set_max_skew`
 - `-from`, `-include`, `-to`: specify paths or pins of a cell
 - `<skew>`: required maximum skew
- Specify maximum path-based skew requirements for registers and ports in the design.
- By default, the command excludes `set_input_delay` and `set_output_delay` values
- When used, results are reported with `report_max_skew`



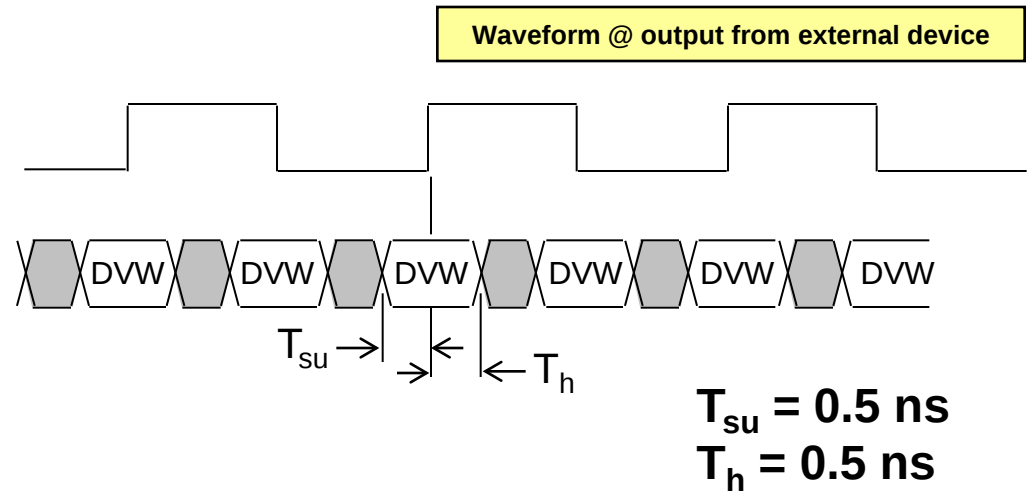
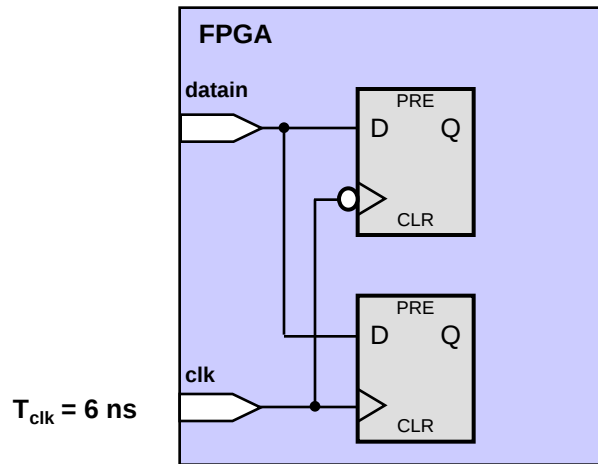
Quartus® II Software Design Series: Timing Analysis

Example Application – DDR Input

DDR Input Example

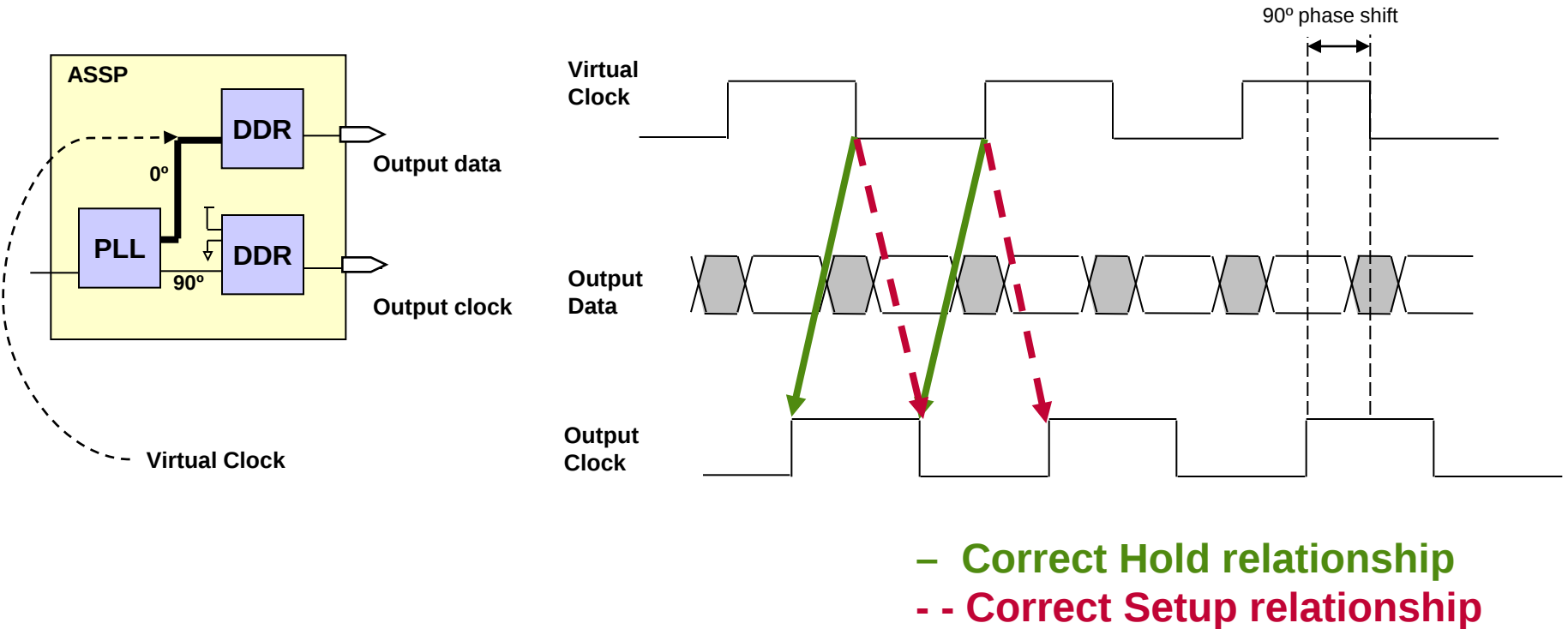


DDR Input Example (cont.)



- What's different about this circuit than prior examples?
- Rising & falling edge input registers from same input port
- Registers have $\frac{1}{2}$ clock period for required time

DDR Input Example (cont.)



- Need false path exceptions to prevent timing analysis on opposite-edge transfers

DDR Input Example (cont.)

```
# Define variables
set clk_period 6
set Tsu 0.5
set Th 0.5

# Create clocks and virtual clocks
create_clock -period $clk_period [get_ports clk]
create_clock -period $clk_period -name clk_v

# Rising edge clock constraint
set_input_delay -clock clk_virt -max [expr $clk_period / 2 - $Tsu] [get_ports {datain}]
set_input_delay -clock clk_virt -min $Th [get_ports {datain}]

# Falling clock edge constraint
set_input_delay -clock clk_virt -max [expr $clk_period / 2 - $Tsu] [get_ports {datain}] \
    -clock_fall -add_delay
set_input_delay -clock clk_virt -min $Th [get_ports {datain}] \
    -clock_fall -add_delay

# Set false paths
set_false_path -setup -rise_from {clk_virt} -fall_to {clk}
set_false_path -setup -fall_from {clk_virt} -rise_to {clk}
set_false_path -hold -rise_from {clk_virt} -rise_to {clk}
set_false_path -hold -fall_from {clk_virt} -fall_to {clk}
```

DDR Reporting

- Use `report_timing` Command
- Must check all rising & falling edge transitions
 - Two data valid windows to check
 - One from a rising edge source clock
 - One from a falling edge source clock
 - Use `rise_from`, `rise_to`, `fall_from`, `fall_to`

Please go to Exercise 5

Timing Analysis Summary

- Timing constraints are very important in FPGA/CPLD design
- Use timing constraints to tell fitter & timing analyzer how logic is designed to function
- SDC provides an easy-to-use, standard interface for constraining design
- See the Quartus II Handbook: Volume 3, Section II, for more information about timing analysis

Learn More Through Technical Training

Instructor-Led Training



With Altera's instructor-led training courses, you can:

- Listen to a lecture from an Altera technical training engineer (instructor)
- Complete hands-on exercises with guidance from an Altera instructor
- Ask questions & receive real-time answers from an Altera instructor
- Each instructor-led class is one or two days in length (8 working hours per day).

Online Training



With Altera's online training courses, you can:

- Take a course at any time that is convenient for you
- Take a course from the comfort of your home or office (no need to travel as with instructor-led courses)

Each online course will take approximate one to three hours to complete.

<http://www.altera.com/training>

View training class schedule & register for a class

Other Quartus II Design Series courses

■ Quartus II Software Design Series: Foundation

- Project creation and management
- Design entry methods and tools
- Compilation and compilation results analysis
- Creating and editing settings and assignments
- I/O planning and management
- Introduction to timing analysis with the TimeQuest timing analyzer

■ Quartus II Software Design Series: Verification

- Basic design simulation with ModelSim-Altera
- Power analysis
- Debugging solutions

■ Quartus II Software Design Series: Optimization

- Incremental compilation
- Quartus II optimization features & techniques

Altera Technical Support

- Reference Quartus II software on-line help
- [Quartus II Handbook](#)
- Altera forum: <http://www.alteraforum.com/>
 - Discuss issues, ask questions, and share solutions with other Altera users
- Consult Altera applications (factory applications engineers)
 - MySupport: <http://www.altera.com/mysupport>
 - Hotline: (800) 800-EPLD (7:00 a.m. - 5:00 p.m. PST)
- Field applications engineers: contact your local Altera sales office
- Receive literature by mail: (888) 3-ALTERA
- FTP: <ftp.altera.com>
- World-wide web: <http://www.altera.com>
 - Use solutions to search for answers to technical problems
 - View design examples

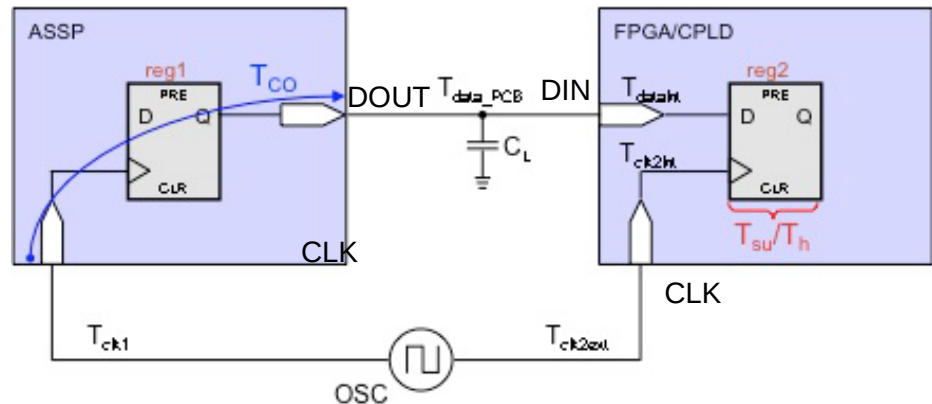
Appendix

Appendix

- System-centric synchronous I/O
- FPGA-centric synchronous I/O
- System-centric to FPGA-centric source synchronous I/O
- Timing Reports
- Latches
- False Path Example
- Annotated Delay
- Time Groups

Synchronous Inputs

System-centric Approach



Setup:

Data Required Time

$$= (T_{\text{clk2ext}} + T_{\text{clk2int}}) + T - T_{\text{SU}}$$

Data Arrival Time

$$= (T_{\text{clk1ext}} + T_{\text{CO}}) + T_{\text{data_trace}} + T_{\text{data2_int}}$$

Setup Slack

$$= \text{Required Time} - \text{Arrival Time} > 0$$

$$= T - T_{\text{SU}} + T_{\text{clk2int}} - T_{\text{data2_int}} - [-T_{\text{skew}} + T_{\text{data_trace}} + T_{\text{CO}}]$$

$$= T - T_{\text{SU}} + T_{\text{clk2int}} - T_{\text{data2_int}} - \text{input delay (max)}$$

Input delay max

$$= T_{\text{data_trace(max)}} + T_{\text{CO(max)}} - T_{\text{skew(min)}}$$

where:

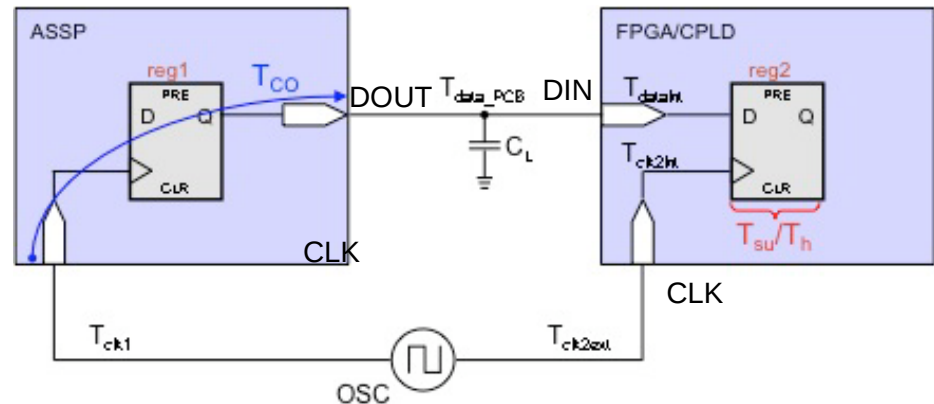
$$T_{\text{skew}} = T_{\text{clk2ext}} - T_{\text{clk1}}$$

and

$$T_{\text{data_trace}} = T_{\text{data_PCB}} + T_{\text{CL}}$$

Synchronous Inputs

System-centric Approach



Hold:

Data Required Time

$$= (T_{\text{clk2ext}} + T_{\text{clk2int}}) + T_h$$

Data Arrival Time

$$= (T_{\text{clk1ext}} + T_{\text{CO}}) + T_{\text{data_trace}} + T_{\text{data2_int}}$$

Hold Slack

$$= \text{Arrival Time} - \text{Required Time} > 0$$

$$= [T_{\text{data_trace}} - T_{\text{skew}} + T_{\text{CO}}] + T_{\text{data2_int}} - T_{\text{clk2int}} - T_h > 0$$

$$= \text{Input delay min} + T_{\text{data2_int}} - T_{\text{clk2int}} - T_h > 0$$

Input delay min

$$= T_{\text{data_trace}(\text{min})} - T_{\text{skew}(\text{max})} + T_{\text{CO}(\text{min})}$$

where:

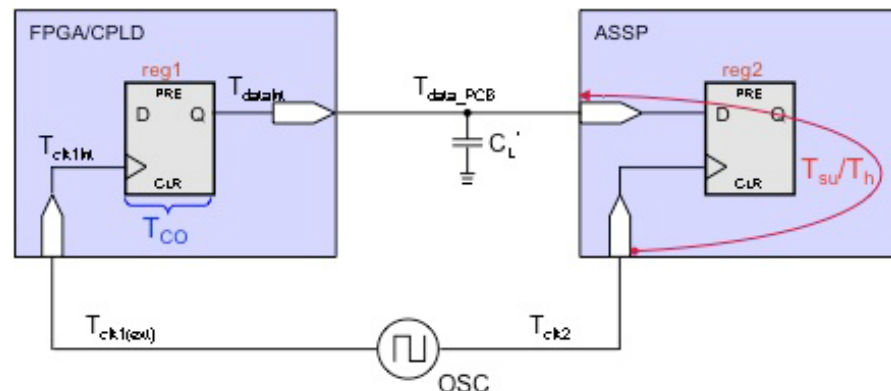
$$T_{\text{skew}} = T_{\text{clk2ext}} - T_{\text{clk1}}$$

and

$$T_{\text{data_trace}} = T_{\text{data_PCB}} + T_{\text{CL}}$$

Synchronous Outputs

System-centric Approach



Setup:

Data Required Time

$$= T_{clk2} + T - T_{SU}$$

Data Arrival Time

$$= (T_{clk1ext} + T_{clk1int} + T_{CO}) + T_{data1_int} + T_{data_trace}$$

Setup Slack

$$= \text{Required Time} - \text{Arrival Time} > 0$$

$$= T - T_{clk1int} - T_{co} - T_{data1_int} - [T_{SU} + T_{data_trace} - T_{skew}] > 0$$

$$= T - T_{co} - T_{data1_int} - \text{output delay (max)}$$

Output delay max

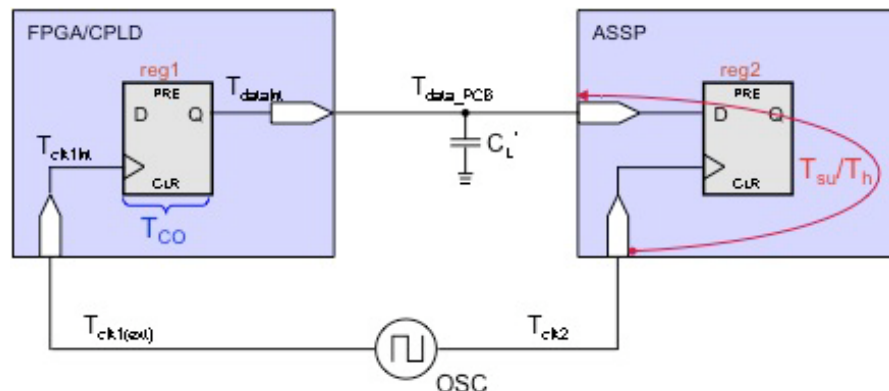
$$= T_{SU} + T_{data_trace(max)} - T_{skew(min)}$$

where:

$$T_{skew} = T_{skew} = T_{clk2ext} - T_{clk1} \quad \text{and} \quad T_{data_trace} = T_{data_PCB} + T_{CL}$$

Synchronous Outputs

System-centric Approach



Hold:

$$\begin{aligned} \text{Data Required Time} &= T_{\text{clk2}} + T - T_h \\ \text{Data Arrival Time} &= T + (T_{\text{clk1ext}} + T_{\text{clk1int}} + T_{\text{CO}}) + T_{\text{data1_int}} + T_{\text{data_trace}} \end{aligned}$$

Hold Slack

$$\begin{aligned} &= \text{Arrival Time} - \text{Required Time} > 0 \\ &= [T_{\text{data_trace}} - T_{\text{skew}} - T_h] + T_{\text{CO}} + T_{\text{data1_int}} + T_{\text{clk1int}} > 0 \\ &= \text{Output delay min} + T_{\text{CO}} + T_{\text{data1_int}} + T_{\text{clk1int}} > 0 \end{aligned}$$

Output delay min

$$= T_{\text{data_trace}(\text{min})} - T_{\text{skew}(\text{max})} - T_h$$

where:

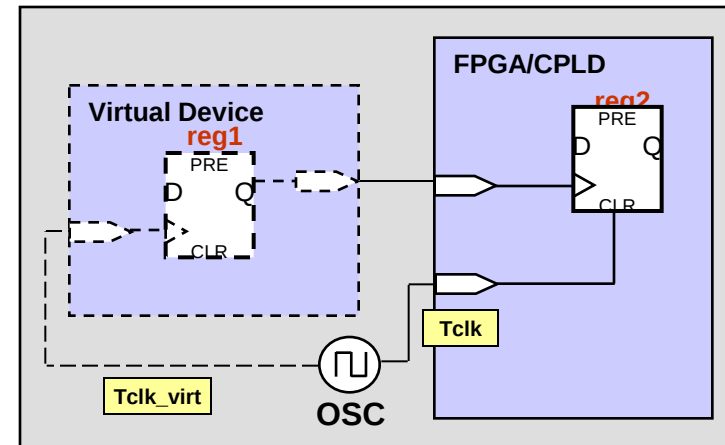
$$T_{\text{skew}} = T_{\text{clk2ext}} - T_{\text{clk1}} \text{ and } T_{\text{data_trace}} = T_{\text{data_PCB}} + T_{\text{CL}}$$

Appendix

- System-centric synchronous I/O
- **FPGA-centric synchronous I/O**
- System-centric to FPGA-centric source synchronous I/O
- Timing Reports
- Latches
- False Path Example
- Annotated Delay
- Time Groups

Synchronous Inputs

FPGA-centric Approach



Setup:

$$\begin{aligned} \text{Data Required Time} &= T_{\text{clk}} - T_{\text{SU}} \\ \text{Data Arrival Time} &= T_{\text{clk_virt}} + \text{input delay max} \end{aligned}$$

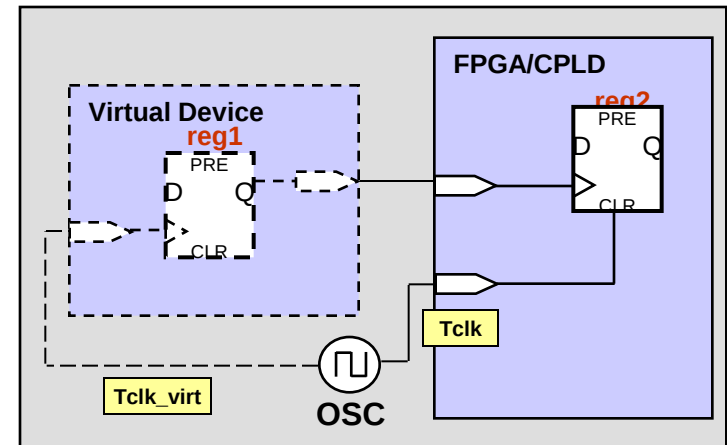
$$\begin{aligned} \text{Setup Slack} &= \text{Data Required time} - \text{Data Arrival Time} > 0 \\ &= [T_{\text{clk}} - T_{\text{SU}}] - [T_{\text{clk_virt}} + \text{input delay max}] > 0 \\ &= [T_{\text{clk}} - T_{\text{clk_virt}}] - T_{\text{SU}} - \text{input delay max} > 0 \end{aligned}$$

$$\text{Input delay max} < [T_{\text{clk}} - T_{\text{clk_virt}}]^* - T_{\text{SU}} = T - T_{\text{SU}}$$

*Note: Tclk and Tclk_virt have same spec and
 Latch edge is one cycle after launch edge => $[T_{\text{clk}} - T_{\text{clk_virt}}] = T$

Synchronous Inputs

FPGA-centric Approach



Hold:

$$\text{Data Required Time} = T_{\text{clk}} + T_h$$

$$\text{Data Arrival Time} = T_{\text{clk_virt}} + \text{input delay min}$$

$$\text{Setup Slack} = \text{Data Arrival time} - \text{Data Required Time} > 0$$

$$= [T_{\text{clk_virt}} + \text{input delay min}] - [T_{\text{clk}} + T_h] > 0$$

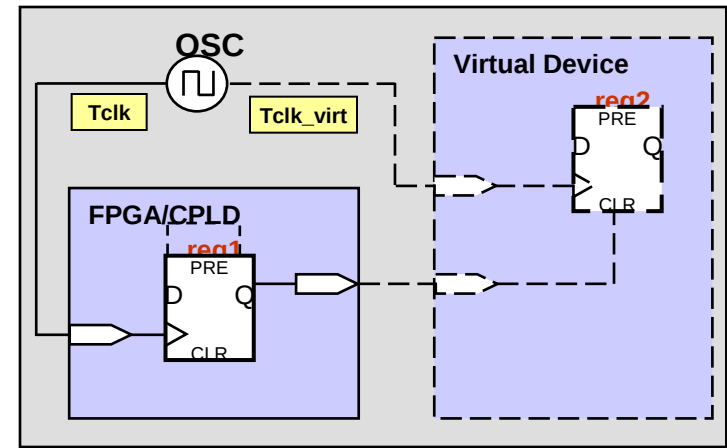
$$\text{Input delay max} < [T_{\text{clk}} - T_{\text{clk_virt}}] + T_h = T_h^*$$

*Note: Tclk and Tclk_virt have same spec and

Hold check is against the current rising edge => $[T_{\text{clk}} - T_{\text{clk_virt}}] = 0$

Synchronous Outputs

FPGA-centric Approach



Setup:

$$\begin{aligned} \text{Data Required Time} &= T_{\text{clk_virt}} \\ \text{Data Arrival Time} &= T_{\text{clk}} + T_{\text{CO}} + \text{output delay max} \end{aligned}$$

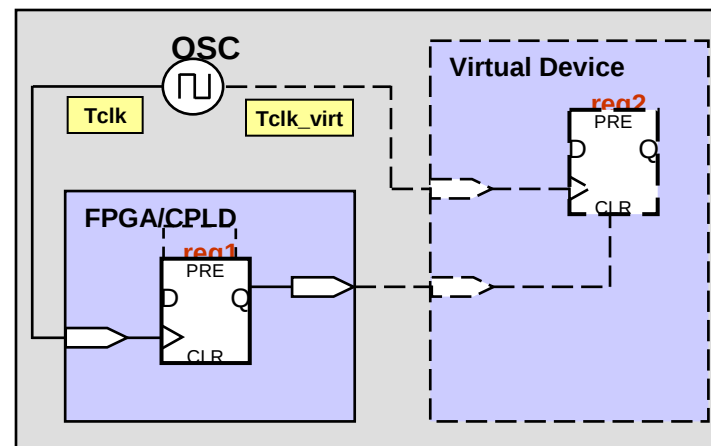
$$\begin{aligned} \text{Setup Slack} &= \text{Data Required time} - \text{Data Arrival Time} > 0 \\ &= [T_{\text{clk_virt}}] - [T_{\text{clk}} + T_{\text{CO}} + \text{output delay max}] > 0 \\ &= - [T_{\text{clk}} - T_{\text{clk_virt}}] - T_{\text{CO}} - \text{input delay max} > 0 \end{aligned}$$

$$\text{Output delay max} = [T_{\text{clk}} - T_{\text{clk_virt}}]^* - T_{\text{CO(max)}} = T - T_{\text{CO(max)}}$$

*Note: Tclk and Tclk_virt have same spec and
 Latch edge is one cycle after launch edge => $[T_{\text{clk}} - T_{\text{clk_virt}}] = T$

Synchronous Outputs

FPGA-centric Approach



Hold:

$$\begin{aligned} \text{Data Required Time} &= T_{\text{clk_virt}} \\ \text{Data Arrival Time} &= T_{\text{clk}} + T_{\text{CO}} + \text{output delay min} \end{aligned}$$

$$\begin{aligned} \text{Setup Slack} &= \text{Data Arrival time} - \text{Data Required Time} > 0 \\ &= [T_{\text{clk}} + T_{\text{CO}} + \text{output delay min}] - [T_{\text{clk_virt}}] > 0 \end{aligned}$$

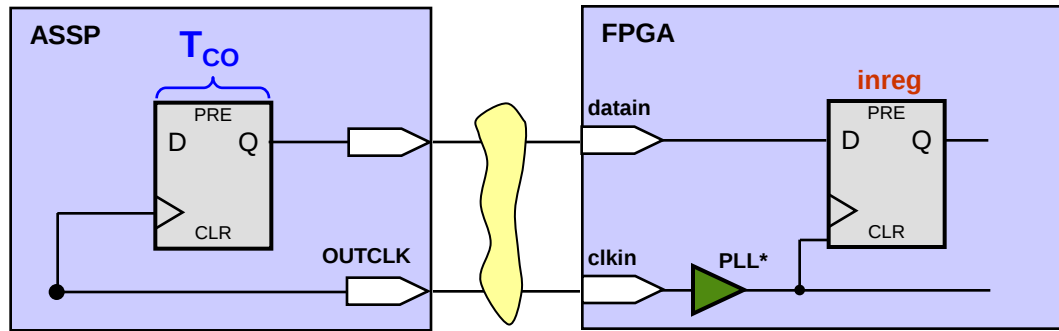
$$\text{Output delay max} < [T_{\text{clk}} - T_{\text{clk_virt}}] - T_{\text{CO(min)}} = T_{\text{CO(min)}}^*$$

*Note: Tclk and Tclk_virt have same spec and
Hold check is against the current rising edge => $[T_{\text{clk}} - T_{\text{clk_virt}}] = 0$

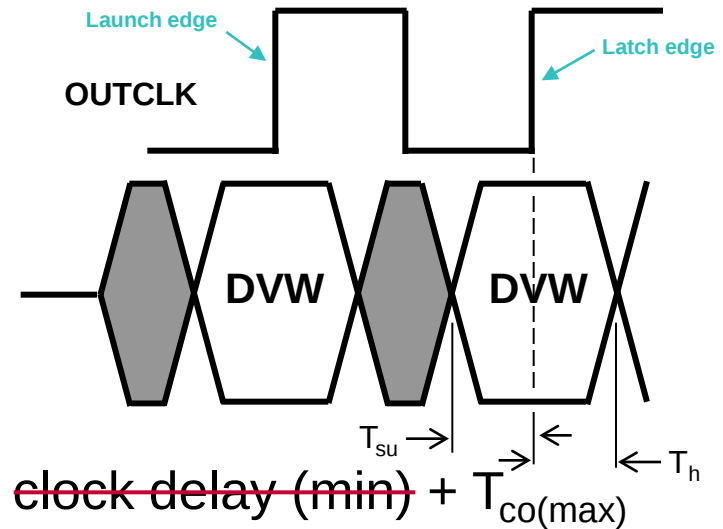
Appendix

- System-centric synchronous I/O
- FPGA-centric synchronous I/O
- System-centric to FPGA-centric source synchronous I/O
- Timing Reports
- Latches
- False Path Example
- Annotated Delay
- Time Groups

SDR Source-Synchronous Input (Center-Aligned)



Waveform @ output from external device



$$\begin{aligned} \text{input delay max} &= \cancel{\text{board delay (max)}} - \cancel{\text{clock delay (min)}} + T_{co(\text{max})} \\ \text{setup slack} &= \text{data required time} - \text{data arrival time} \end{aligned}$$

If setup slack = 0 (start of DVW):

$$\text{data arrival time} = \text{data required time}$$

$$\text{latch edge} - T_{su} = \text{launch edge} + \text{input delay max}$$

SO

$$\text{input delay max} = (\text{latch edge} - \text{launch edge})^* - T_{su} \quad \text{*Typically 1 clock period for SDR}$$

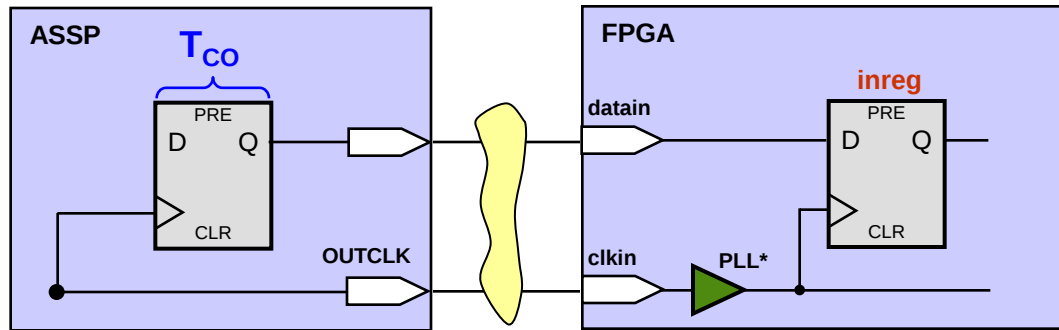
Note: In reality for high-speed designs, there would be some max/min board & clock delay that would need to be figured into the analysis.

© 2010 Altera Corporation—Confidential

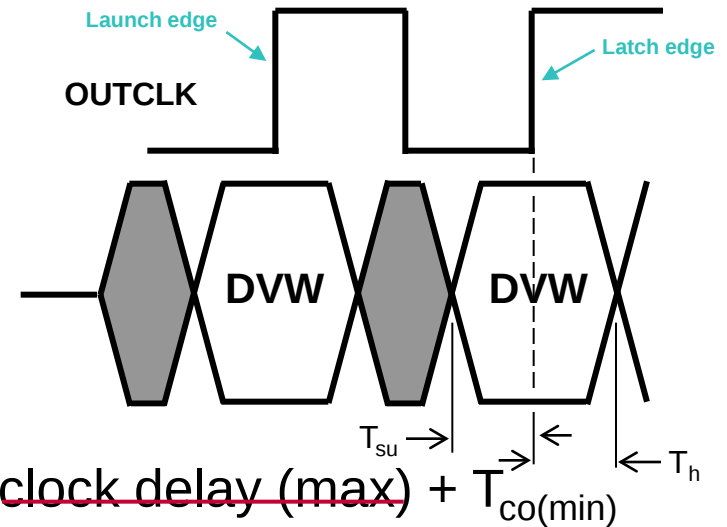
ALTERA, ARRIA, CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS & STRATIX are Reg. U.S. Pat. & Tm. Off. and Altera marks in and outside the U.S.



SDR Source-Synchronous Input (Center-Aligned)



Waveform @ output from external device



$$\begin{aligned} \text{input delay min} &= \cancel{\text{board delay (min)}} - \cancel{\text{clock delay (max)}} + T_{co(\text{min})} \\ &= T_{co(\text{min})} \end{aligned}$$

$$\text{hold slack} = \text{data arrival time} - \text{data required time}$$

If hold slack = 0 (end of DVW):

$$\text{data required time} = \text{data arrival time}$$

$$\text{latch edge} + T_h = \text{launch edge} + \text{input delay min}$$

For hold analysis, latch and launch edges cancel out, so

$$\text{input delay min} = T_h$$

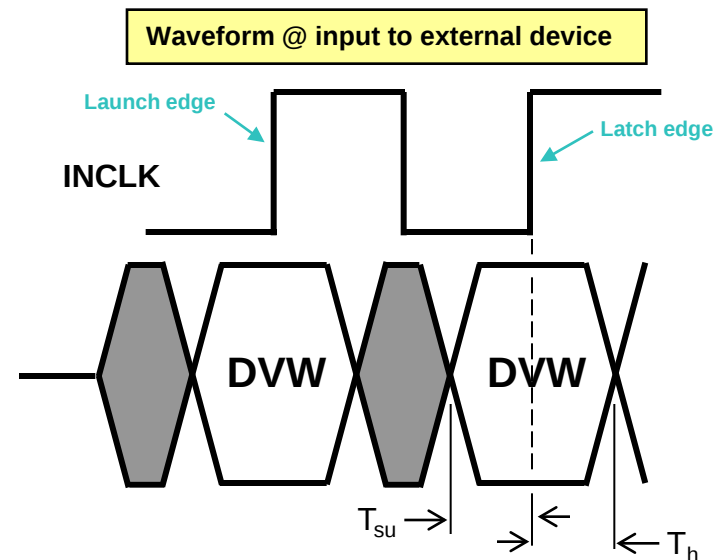
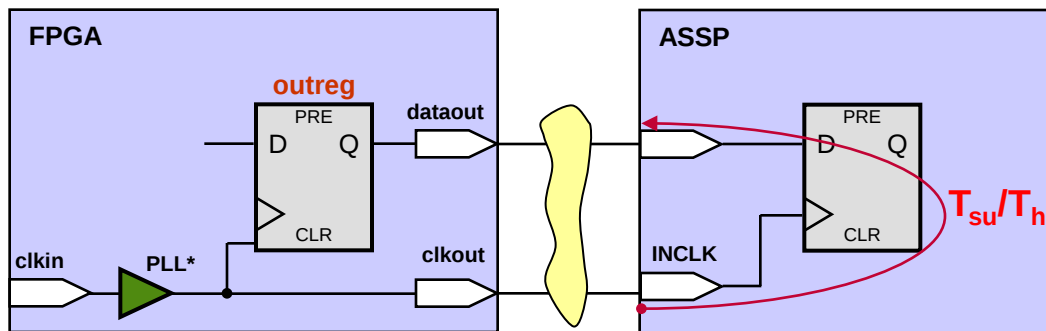
Note: In reality for high-speed designs, there would be some max/min board & clock delay that would need to be figured into the analysis.

© 2010 Altera Corporation—Confidential

ALTERA, ARRIA, CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS & STRATIX are Reg. U.S. Pat. & Tm. Off. and Altera marks in and outside the U.S.

ALTERA®

SDR Source-Synchronous Output (Center-Aligned)



* The PLL in this example is used to shift output clock to establish an output clock to data relationship

$$\begin{aligned} \text{output delay max} &= \cancel{\text{board delay (max)}} - \cancel{\text{clock delay (min)}} + T_{su} \\ &= T_{su} \end{aligned}$$

$$\text{output delay min} = \cancel{\text{board delay (min)}} - \cancel{\text{clock delay (max)}} - T_h$$

Notice output delay minimum is negative

$$\Rightarrow -T_h$$

Notes:

- 1) In reality for high-speed designs, there would be some max/min board & clock delay that would need to be figured into the analysis.
- 2) The PLL in this example is used to shift output clock to establish an output clock to data relationship

© 2010 Altera Corporation—Confidential

ALTERA, ARRIA, CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS & STRATIX are Reg. U.S. Pat. & Tm. Off. and Altera marks in and outside the U.S.

ALTERA®

Appendix

- System-centric synchronous I/O
- FPGA-centric synchronous I/O
- System-centric to FPGA-centric source synchronous I/O
- Timing Reports
- Latches
- False Path Example
- Annotated Delay
- Time Groups

Reporting in Quartus II Comp. Report

- By default, only basic reports generated
- Enable multi-corner analysis to view summaries for all corners
- Enable worst-case path reporting for each clock domain
- Customize reporting with Tcl script

Quartus II - /data/davlin/QIIT9_1/Timing/top - top - [Compilation Rep...

File Edit View Tools Window

TimeQuest Timing Analyzer

- Summary
- Parallel Compilation
- SDC File List
- Clocks
- Slow 1200mV 85C Model Setup Summary
- Fmax Summary
- Setup Summary
- Hold Summary
- Recovery Summary
- Removal Summary
- Minimum Pulse Width
- Worst-Case Timing
- Datasheet Report
- Metastability Report
- Slow 1200mV 0C Model
- Fast 1200mV 0C Model
- Multicorner Timing Analysis
- Multicorner Datasheet
- Advanced I/O Timing
- Clock Transfers
- Report TCCS
- Report RSKM
- Unconstrained Paths
- Messages

Slow 1200mV 85C Model Setup Summary

	Clock	Slack	End Point TNS
1	clk_x2	0.295	0.000
2	clk_x1	0.846	0.000
3	clk_out	2.256	0.000

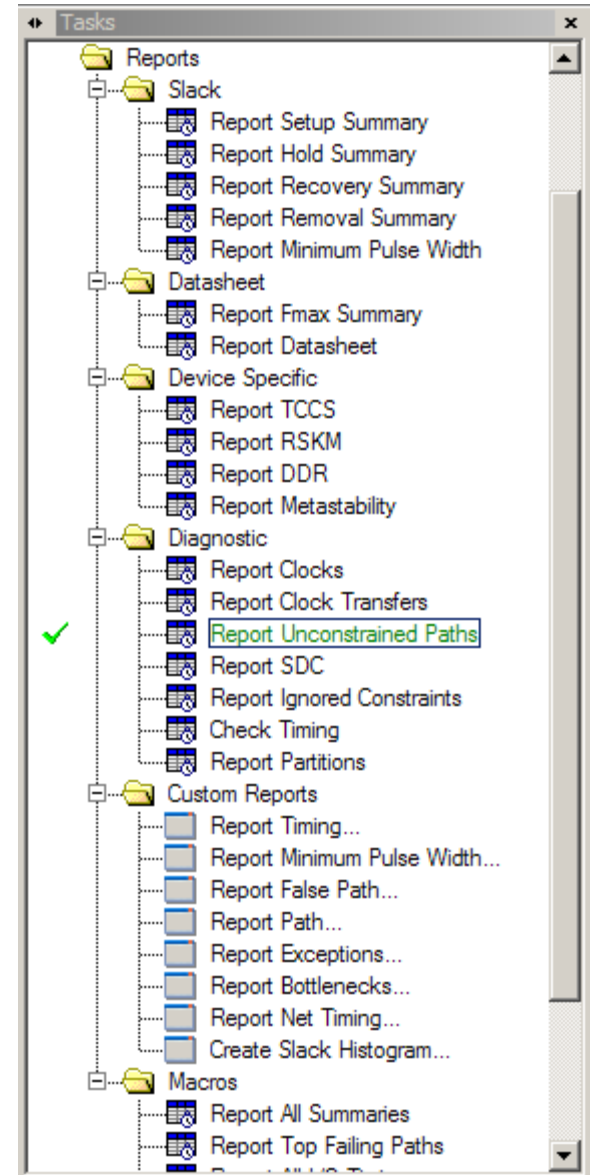
Three process corners analyzed for supported devices

- SDC files used during fitting
- Clocks generated
- Timing violations
- Unconstrained paths

For Help, press F1

Reporting in TimeQuest TA

- Much more control over generation of many different types of reports
- Simple report creation by double-clicking common report types in the **Tasks** pane or selecting from **Reports** menu
- Complex report creation with custom reports or command line reporting features
- Use Tcl scripts (run from **Script** menu) with reporting commands to quickly regenerate reports for analysis



Report Destinations

■ Targeted viewing pane in the GUI

- Default destination for all reports
- `-panel_name <name>`: customize report panel name

■ Console

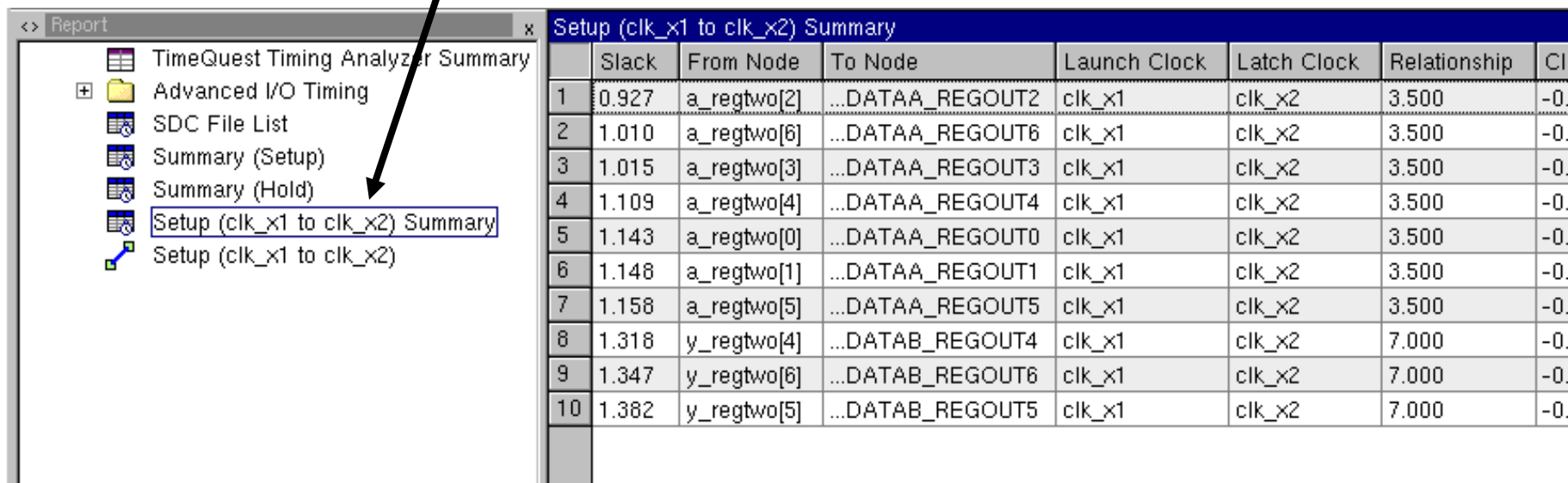
- Report results displayed in the console
- `-stdout`: enable console reporting

■ Output file

- Store report results in a .txt or .html file
- `-file <name>`: name file to store results
- `-append`: append results to existing file specified by `-file` option

Report Output (GUI)

Custom report sent to GUI report panel



The screenshot shows the 'Report' window in a GUI. On the left is a tree view of reports. On the right is a table titled 'Setup (clk_x1 to clk_x2) Summary'. An arrow points from the text 'Custom report sent to GUI report panel' to the 'Setup (clk_x1 to clk_x2) Summary' report in the tree view.

	Slack	From Node	To Node	Launch Clock	Latch Clock	Relationship	CI
1	0.927	a_regtwo[2]	...DATAA_REGOUT2	clk_x1	clk_x2	3.500	-0.
2	1.010	a_regtwo[6]	...DATAA_REGOUT6	clk_x1	clk_x2	3.500	-0.
3	1.015	a_regtwo[3]	...DATAA_REGOUT3	clk_x1	clk_x2	3.500	-0.
4	1.109	a_regtwo[4]	...DATAA_REGOUT4	clk_x1	clk_x2	3.500	-0.
5	1.143	a_regtwo[0]	...DATAA_REGOUT0	clk_x1	clk_x2	3.500	-0.
6	1.148	a_regtwo[1]	...DATAA_REGOUT1	clk_x1	clk_x2	3.500	-0.
7	1.158	a_regtwo[5]	...DATAA_REGOUT5	clk_x1	clk_x2	3.500	-0.
8	1.318	y_regtwo[4]	...DATAB_REGOUT4	clk_x1	clk_x2	7.000	-0.
9	1.347	y_regtwo[6]	...DATAB_REGOUT6	clk_x1	clk_x2	7.000	-0.
10	1.382	y_regtwo[5]	...DATAB_REGOUT5	clk_x1	clk_x2	7.000	-0.

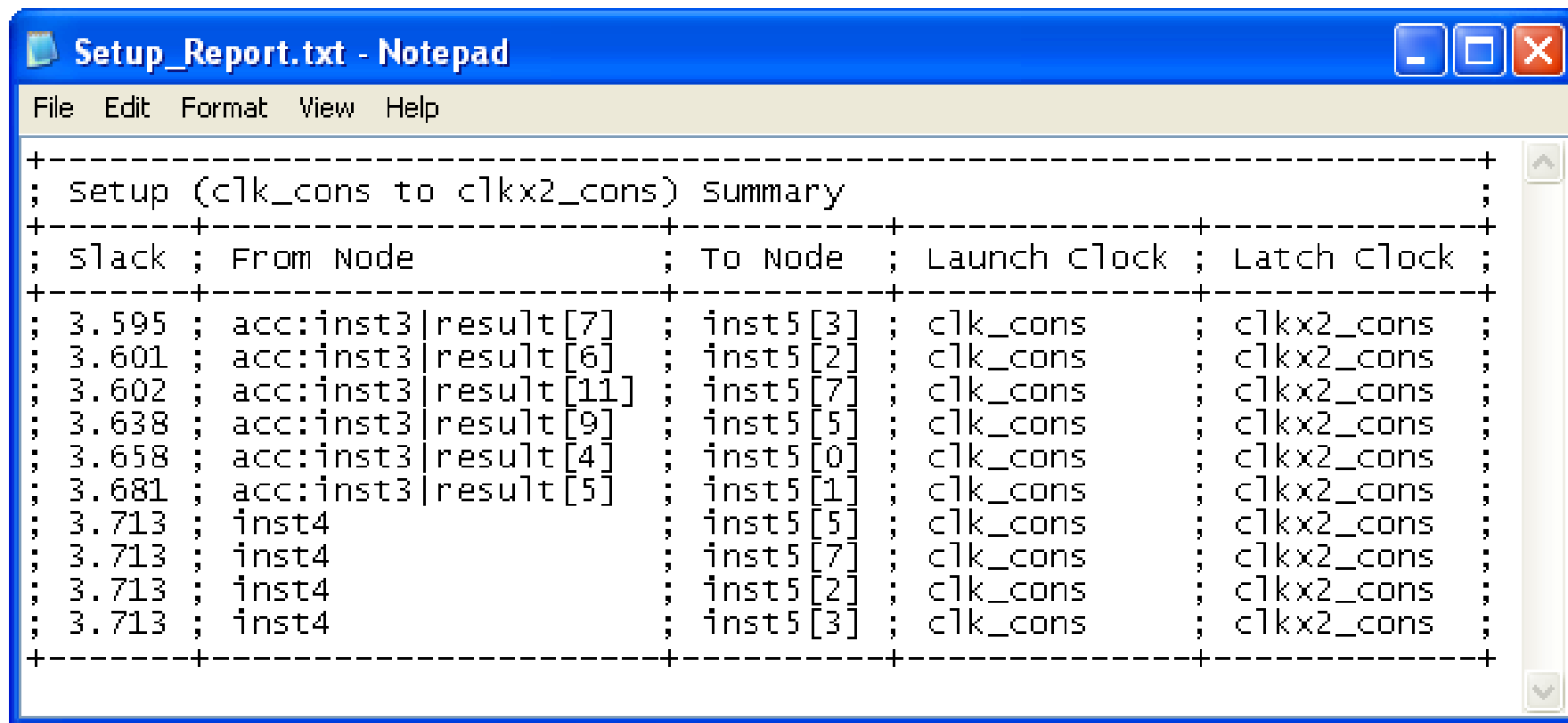
Custom Report Output (Console)

```
tcl> report_timing -from_clock clk_cons -to_clock clkx2_cons -setup -npaths 10 -detail summary -stdout -panel_name "Setu
87
88 Info: Report Timing: Found 10 setup paths (0 violated). Worst case slack is 3.595
95 Info: Path #1: slack is 3.595
102 Info: Path #2: slack is 3.601
109 Info: Path #3: slack is 3.602
116 Info: Path #4: slack is 3.638
123 Info: Path #5: slack is 3.658
130 Info: Path #6: slack is 3.681
137 Info: Path #7: slack is 3.713
144 Info: Path #8: slack is 3.713
151 Info: Path #9: slack is 3.713
158 Info: Path #10: slack is 3.713
165 10 3.595
```

Custom report (ASCII)
sent to Console pane

Expand for more detail

Custom Report Output (File)



```

+-----+-----+-----+-----+
; Setup (clk_cons to clkx2_cons) Summary
+-----+-----+-----+-----+
; slack ; From Node           ; To Node   ; Launch clock ; Latch clock ;
+-----+-----+-----+-----+
; 3.595 ; acc:inst3|result[7]   ; inst5[3]  ; clk_cons     ; clkx2_cons  ;
; 3.601 ; acc:inst3|result[6]   ; inst5[2]  ; clk_cons     ; clkx2_cons  ;
; 3.602 ; acc:inst3|result[11]  ; inst5[7]  ; clk_cons     ; clkx2_cons  ;
; 3.638 ; acc:inst3|result[9]   ; inst5[5]  ; clk_cons     ; clkx2_cons  ;
; 3.658 ; acc:inst3|result[4]   ; inst5[0]  ; clk_cons     ; clkx2_cons  ;
; 3.681 ; acc:inst3|result[5]   ; inst5[1]  ; clk_cons     ; clkx2_cons  ;
; 3.713 ; inst4                ; inst5[5]  ; clk_cons     ; clkx2_cons  ;
; 3.713 ; inst4                ; inst5[7]  ; clk_cons     ; clkx2_cons  ;
; 3.713 ; inst4                ; inst5[2]  ; clk_cons     ; clkx2_cons  ;
; 3.713 ; inst4                ; inst5[3]  ; clk_cons     ; clkx2_cons  ;
+-----+-----+-----+-----+
  
```

Custom report (ASCII) sent to file

Diagnostic Reports

■ Report SDC

- Lists constraints successfully applied to the netlist, organized by constraint type
- Command: `report_sdc`

■ Report Clocks

- List all the clocks defined by constraints in the design
- Command: `report_clocks`

■ Report Ignored Constraints

- Lists commands ignored by the TA, usually due to typos or incorrect constraint arguments
- Command: `report_sdc -ignored`

■ Report Unconstrained Paths

- Lists input and output ports and paths that have not been constrained
- Command: `report_ucp`

■ *See examples later with actual constraints*

Other Basic Timing Reports

■ Report Clock Transfers

- Summarizes number of paths that cross between clock domains
- Command: `report_clock_transfers`

■ Report Datasheet

- Summarizes timing requirements for the entire design
- T_{su} , T_h , T_{co} , $T_{co (min)}$, T_{pd} , $T_{pd (min)}$
- Command: `report_datasheet`

■ Check Timing

- Checks for potential timing problems with design or constraints
- `-include <check_list>`: perform check only on listed checks
- Command: `check_timing`

■ Report Fmax Summary

- Report potential Fmax for all clocks in the design
- Command: `report_clock_fmax_summary`

Reporting Macros

- Built-in shortcut tasks that generate multiple reports
 - Report All Summaries
 - Quick command to generate all summary reports
 - Report Top Failing Paths
 - Report All I/O Timings
 - Report All Core Timings
 - Reports worst case slack on worst register-to-register pairs throughout design
 - Create All Clock Histograms (described next)

Advanced Reporting: Slack Histogram

- Create histograms showing number of edges with a certain amount of slack within a clock domain
- Command: `create_slack_histogram`

Create Slack Histogram

Clock name:

Slack limits

Maximum: ns

Minimum: ns

Display options

Number of bins:

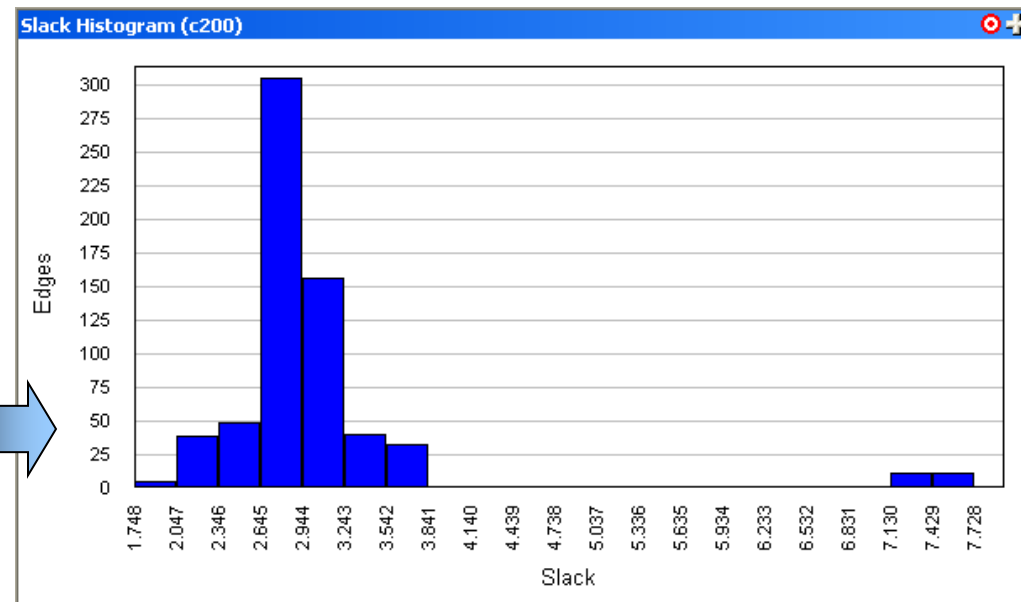
Report panel name:

Tcl command:

OK Cancel Help

Range of slack amount to report
(-max_slack, -min_slack)

of "bins" along X axis to place bars
(-num_bins)



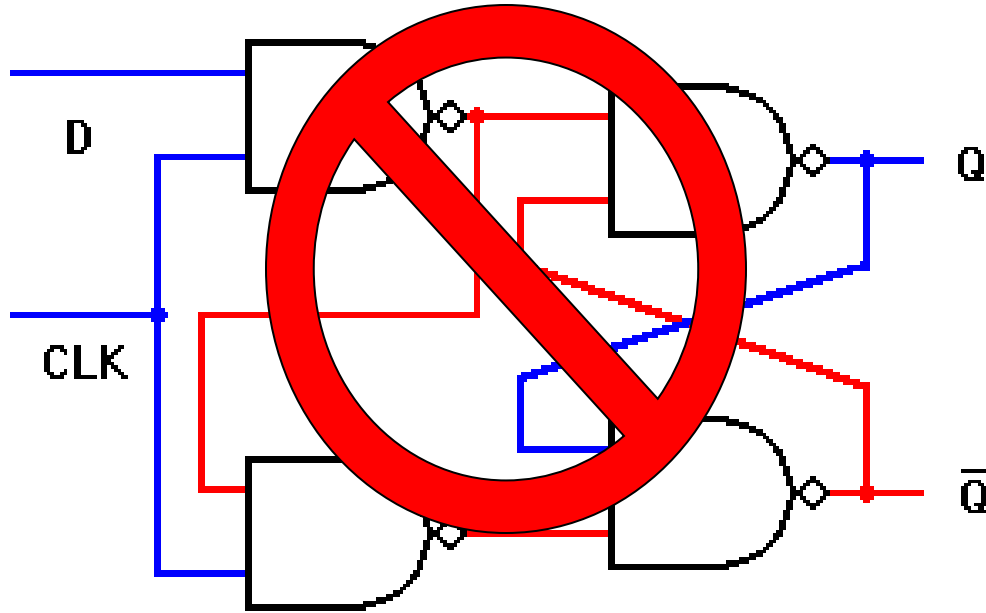
Other Reports

- Report Metastability
 - Size and names of found synchronization chains and MTBF for each
- Report TCCS & Report TSKM
 - Channel-to-channel & receiver skew margins for LVDS interfaces
- Report DDR
 - Custom reporting for use with the ALTMEMPHY high performance memory controller megafunction
- Report Minimum Pulse Width
 - Reports minimum widths of clock pulses required to recognize clock transitions
- Report Net Timing & Report Path
 - Detailed information about specific nets or paths based on selected criteria
- *See Handbook and on-line help for details*

Appendix

- System-centric synchronous I/O
- FPGA-centric synchronous I/O
- System-centric to FPGA-centric source synchronous I/O
- Timing Reports
- **Latches**
- False Path Example
- Annotated Delay
- Time Groups

Latches



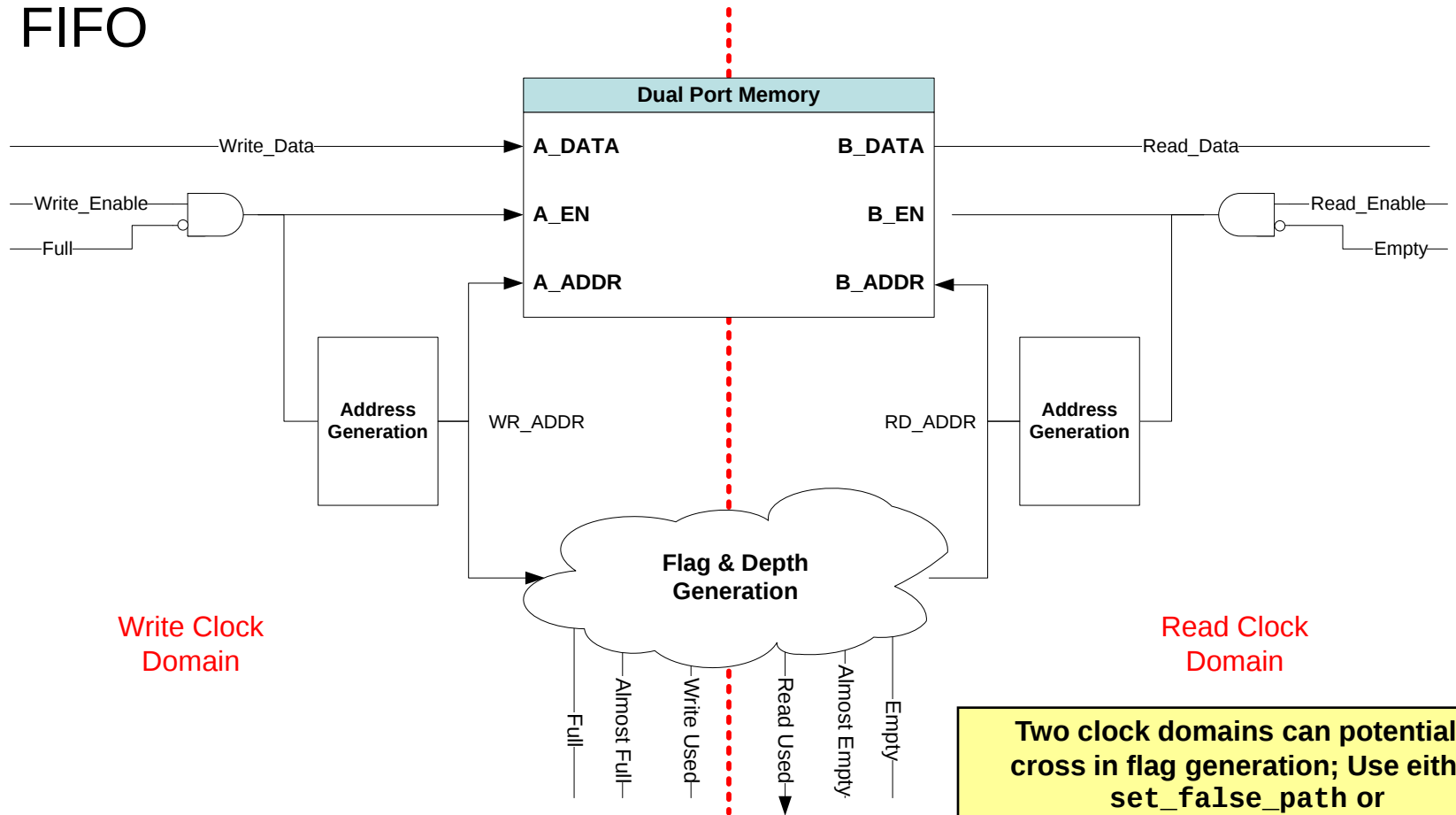
- Simply put, don't use them! Use registers!
- Really, I'm serious – not a good idea
 - If you insist on it, check out the Quartus II Handbook

Appendix

- System-centric synchronous I/O
- FPGA-centric synchronous I/O
- System-centric to FPGA-centric source synchronous I/O
- Timing Reports
- Latches
- False Path Example
- Annotated Delay
- Time Groups

Real World Example: Memory FIFO

- FIFO bridging two clock domains; Flags indicate status of FIFO



Two clock domains can potentially cross in flag generation; Use either `set_false_path` or `set_clock_groups` depending on additional logic in design

Appendix

- System-centric synchronous I/O
- FPGA-centric synchronous I/O
- System-centric to FPGA-centric source synchronous I/O
- Timing Reports
- Latches
- False Path Example
- **Annotated Delay**
- Time Groups

Annotated Delays

- `set_annotated_delay`
 - `-net`, `-cell`: apply to paths or to pins of a cell specified with `-from`, `-to`
 - `-ff`, `-fr`, `-rf`, `-rr`: delay applied to specified edges
- Set specific delay values for paths or between cell outputs without overriding clock relationships
- Good for output buffers and tweaking
- **Zero IC Delays** when creating post-map netlist is essentially:
`set_annotated_delay -net 0.000`

Appendix

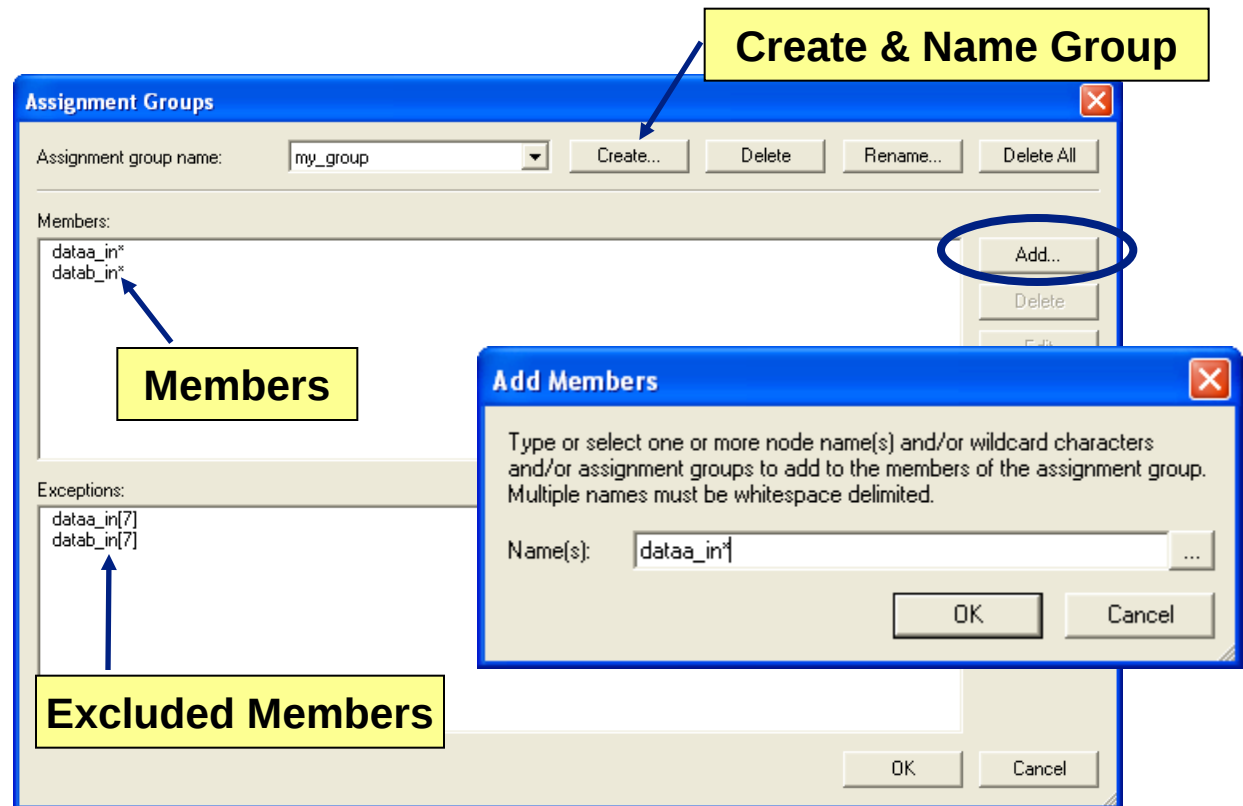
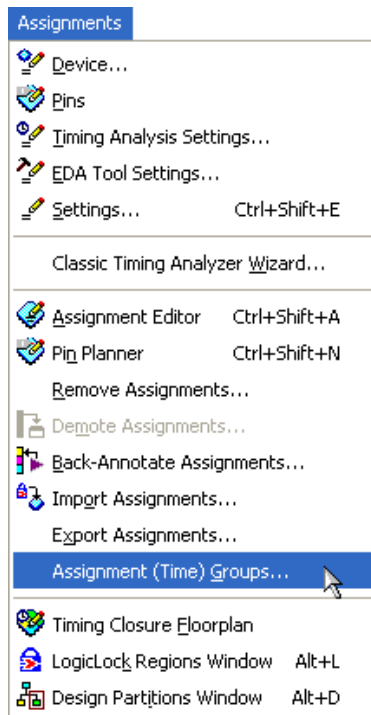
- System-centric synchronous I/O
- FPGA-centric synchronous I/O
- System-centric to FPGA-centric source synchronous I/O
- Timing Reports
- Latches
- False Path Example
- Annotated Delay
- Time Groups

Time Groups

- Define a custom group of nodes to which you can assign timing assignments and/or requirements
- Members can include regular node names, wildcards, and/or other time group names
- Can improve overall software performance
- Tcl “set” command also supported

Time Groups

- Use Quartus II software to create groups



```
Tcl: timegroup <group name> \  
      -add_member <names>
```

Accessing in TimeQuest

- Use `get_assignment_groups` (SDC extension) collection to apply constraints to nodes
- `get_assignment_groups` options
 - `[-keepers]`
 - `[-ports]`
 - `[-registers]`
 - `<name>`
- Example

```
set_multicycle_path -from [get_assignment_groups src_group]
                    -to  [get_assignment_groups dst_group] 2
```