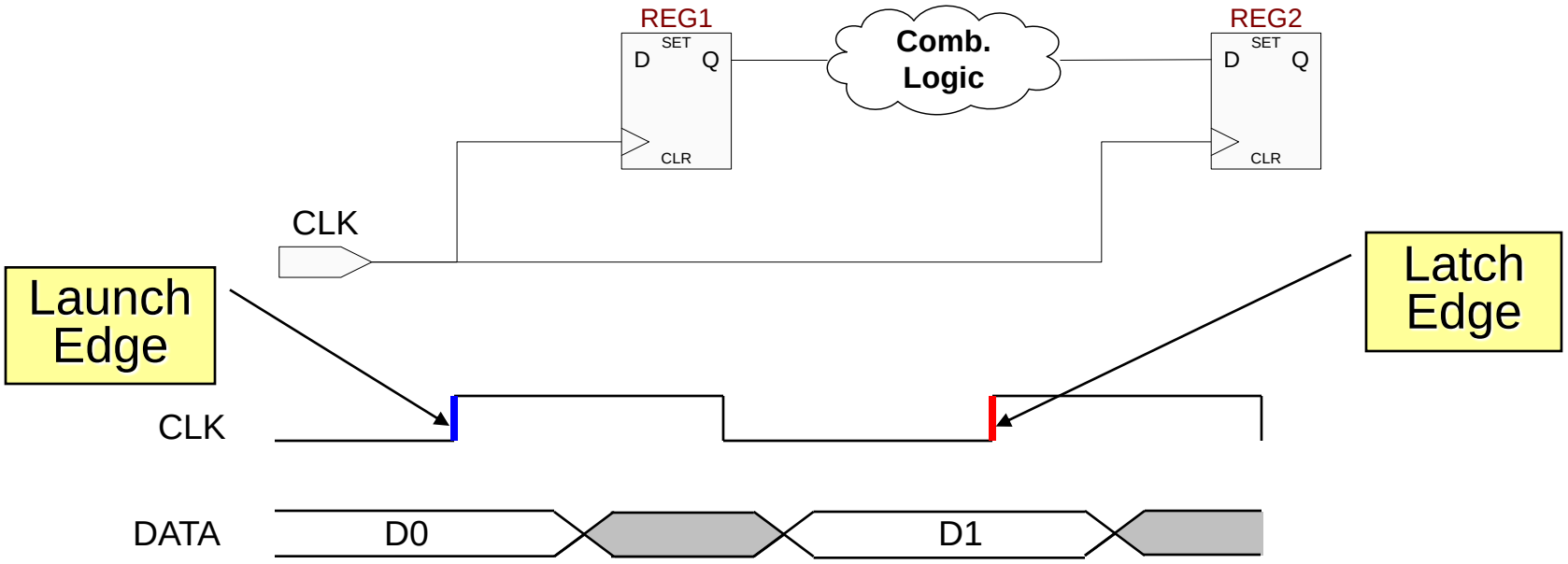
Timing Analysis with Timequest

Timing Analysis Basics

Timing Analysis Basics Objectives

- Define the basic timing parameters used in timing analysis
- Understand the calculations performed by the timing analyzer for reporting timing results

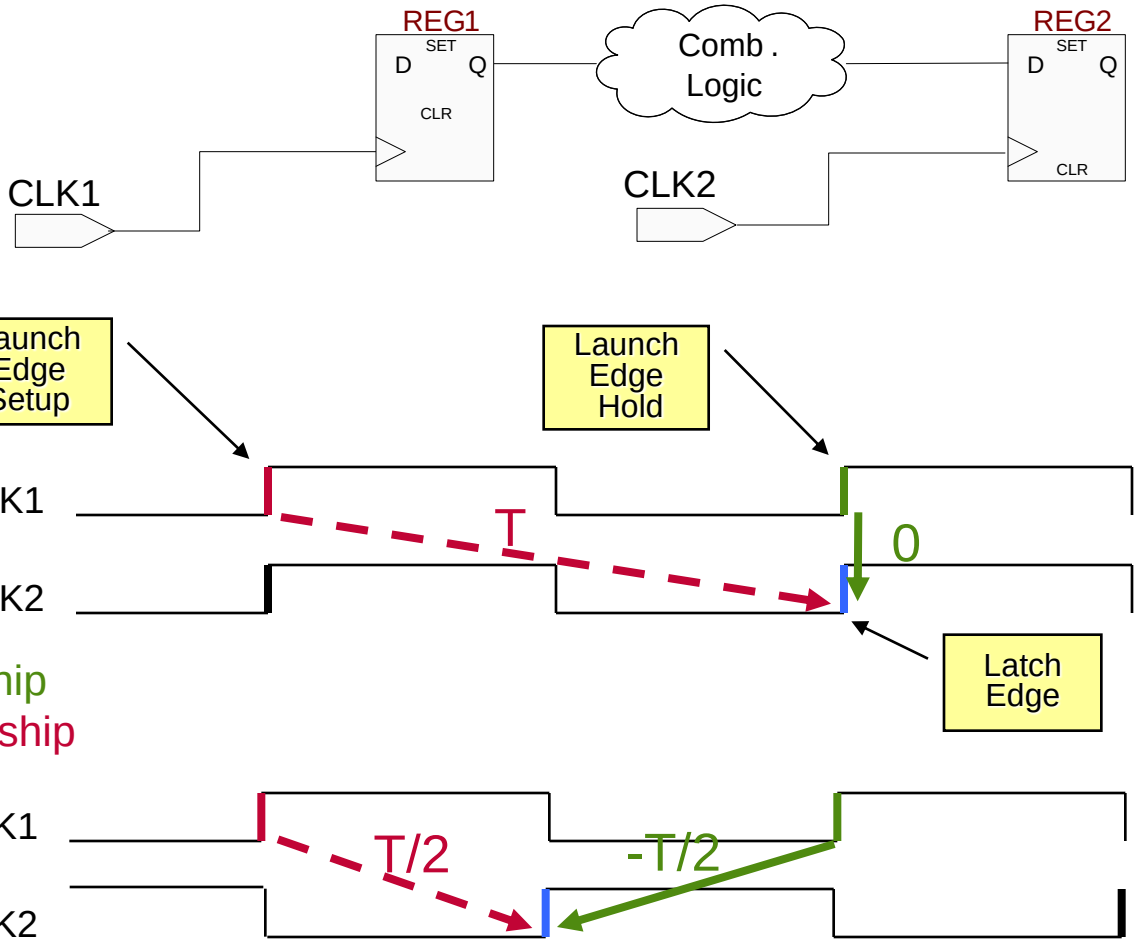
Launch & Latch Edges



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)

Setup & Hold relationships

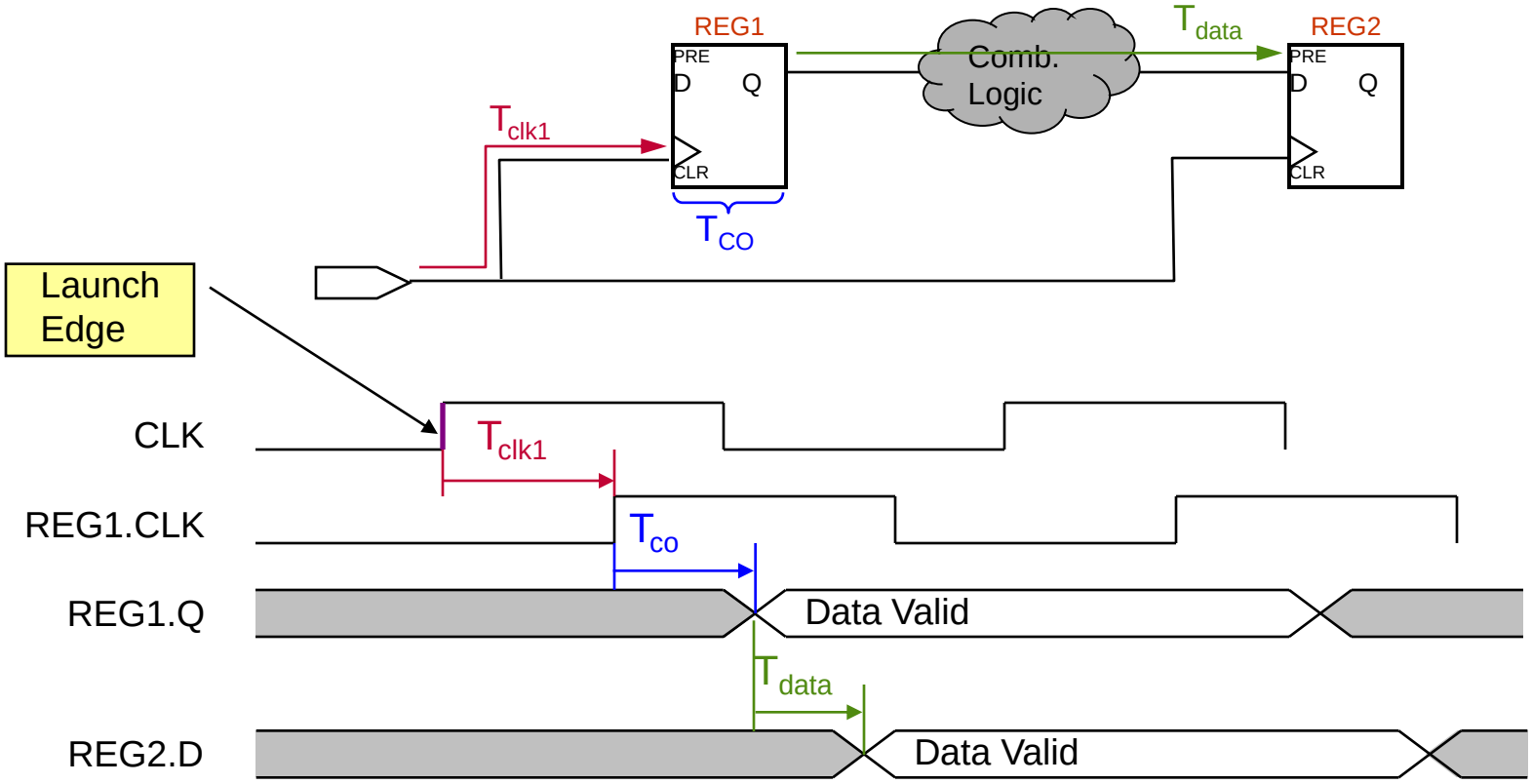


– Hold relationship

- - Setup relationship

Data Arrival Time

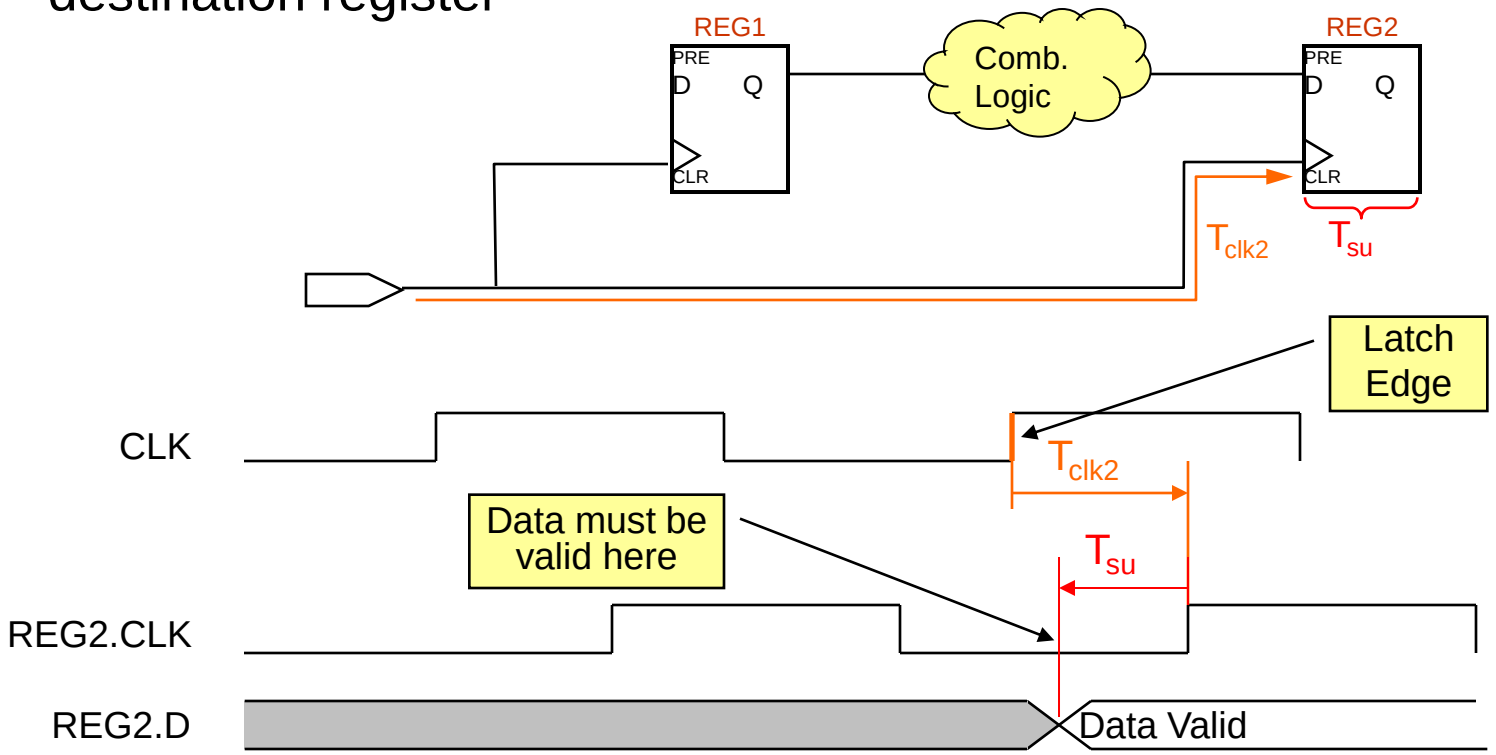
- The time for data to arrive at destination register's D input



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

Data Required Time - Setup

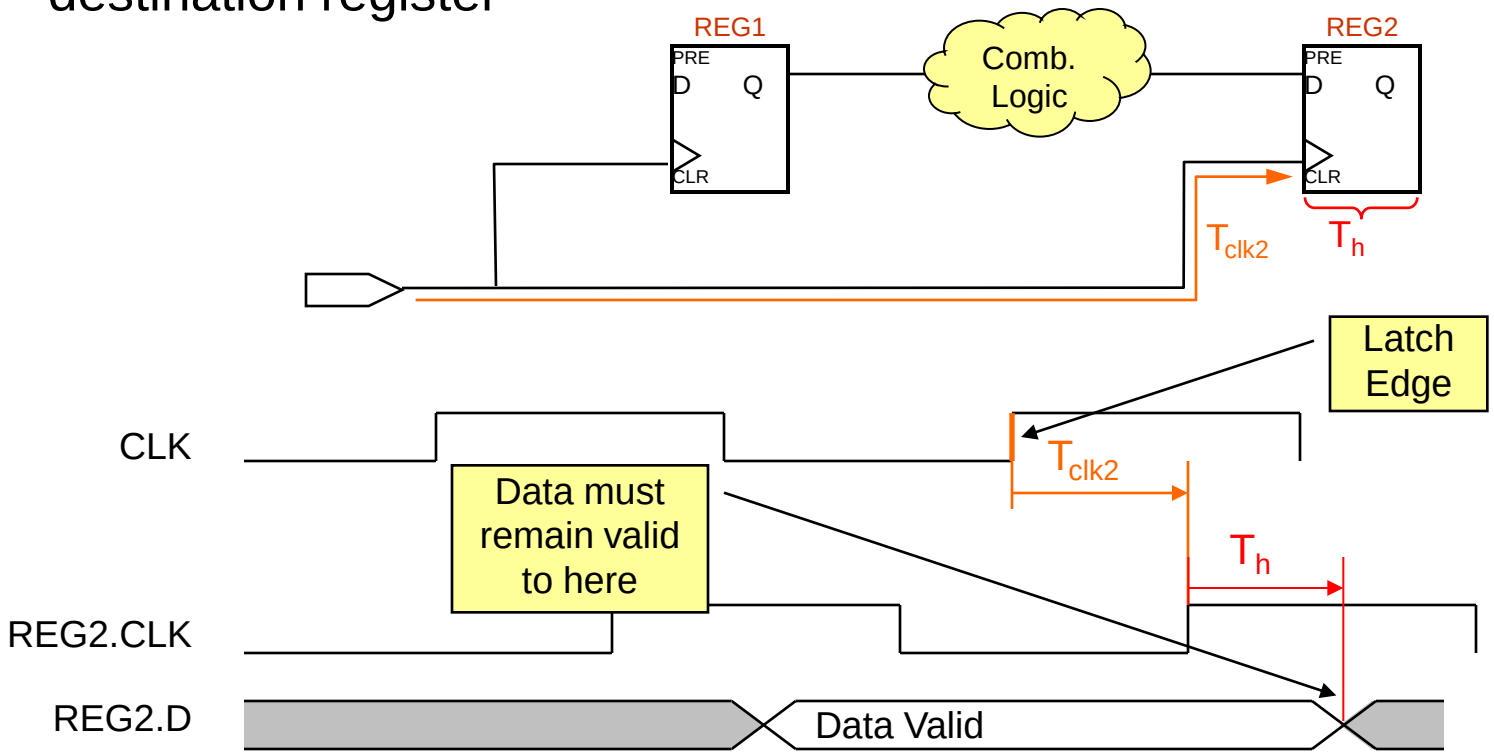
- The minimum time required for the data to get latched into the destination register



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

Data Required Time - Hold

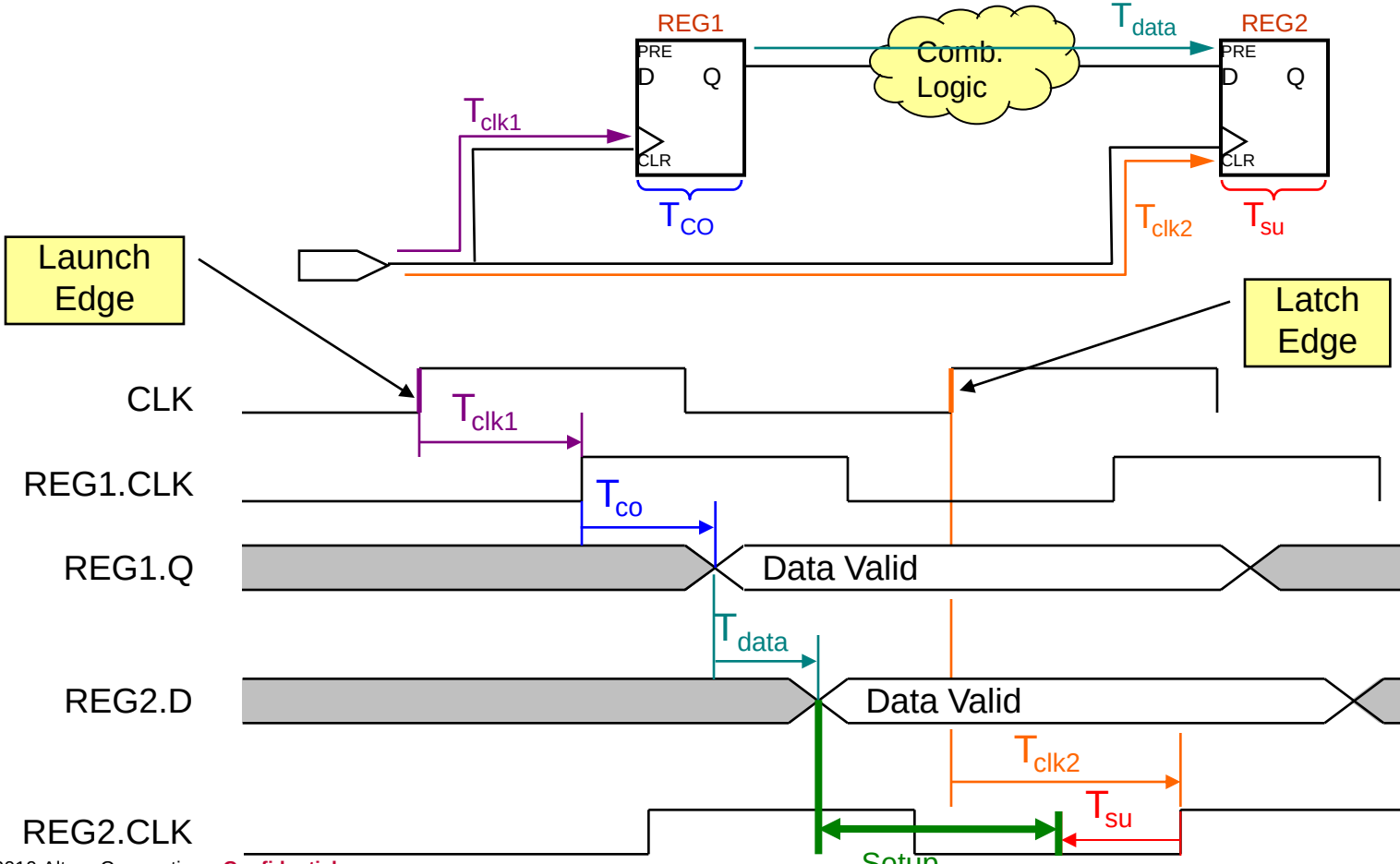
- The minimum time required for the data to get latched into the destination register



Data Required Time = Clock Arrival Time + T_h + Hold Uncertainty

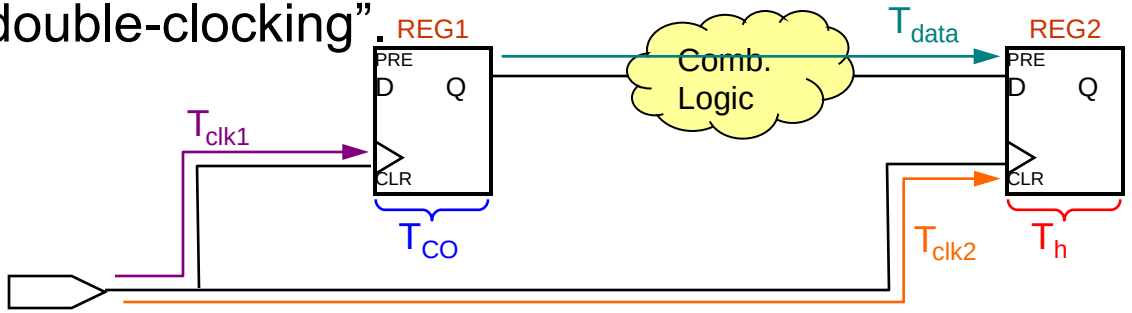
Setup Slack

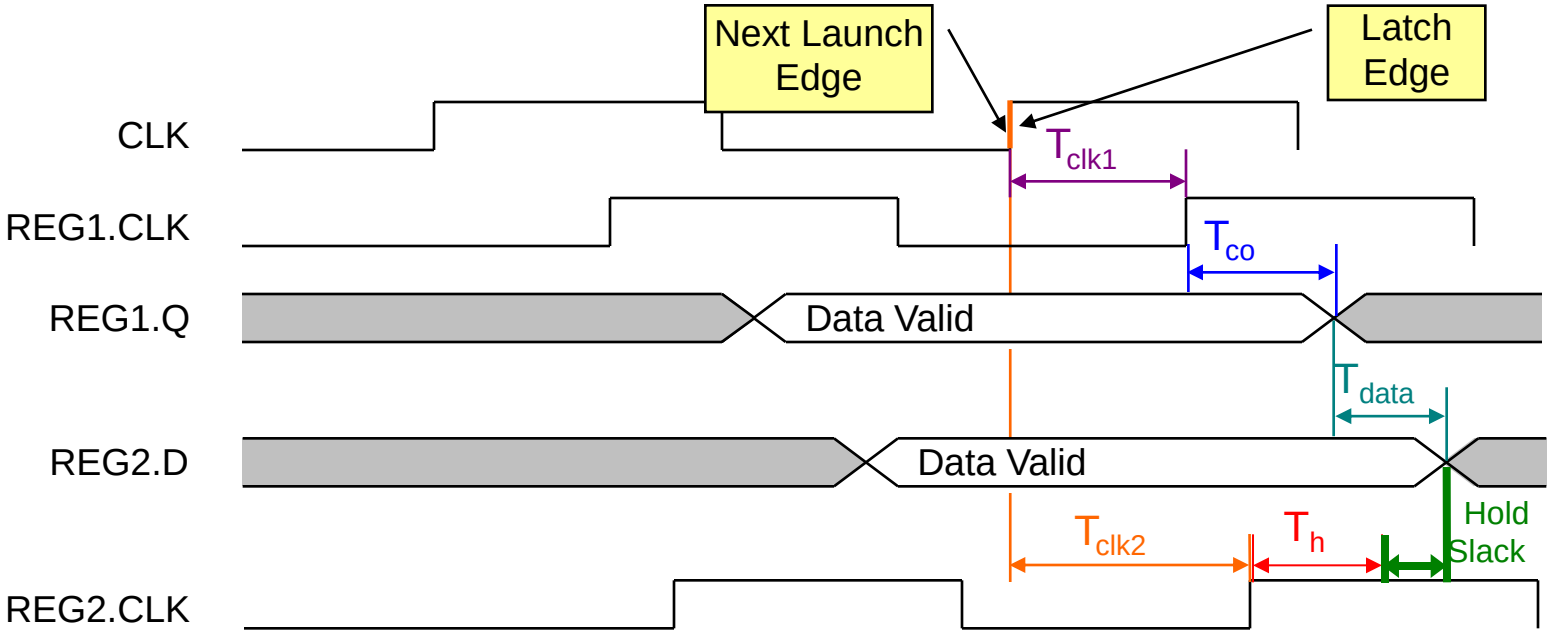
- The margin by which the setup timing requirement is met. It ensures launched data arrives in time to meet the latching requirement.



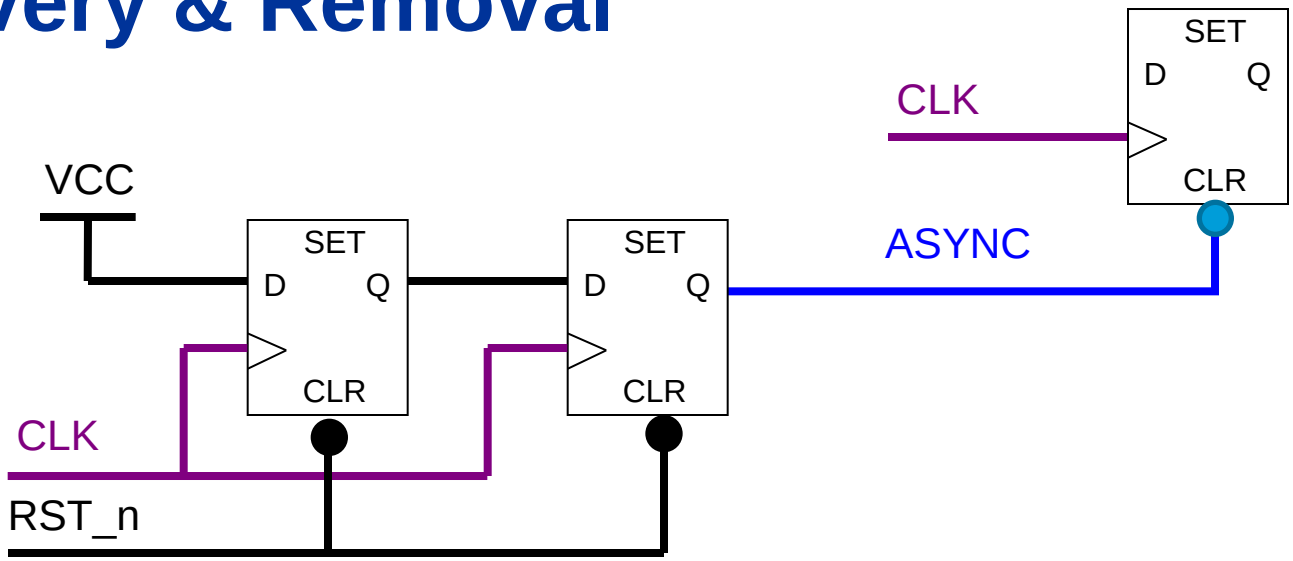
Hold Slack

- 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”.



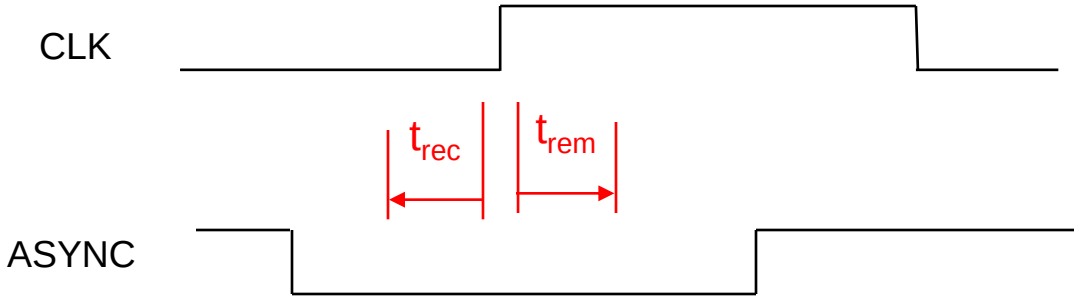


Recovery & Removal



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

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



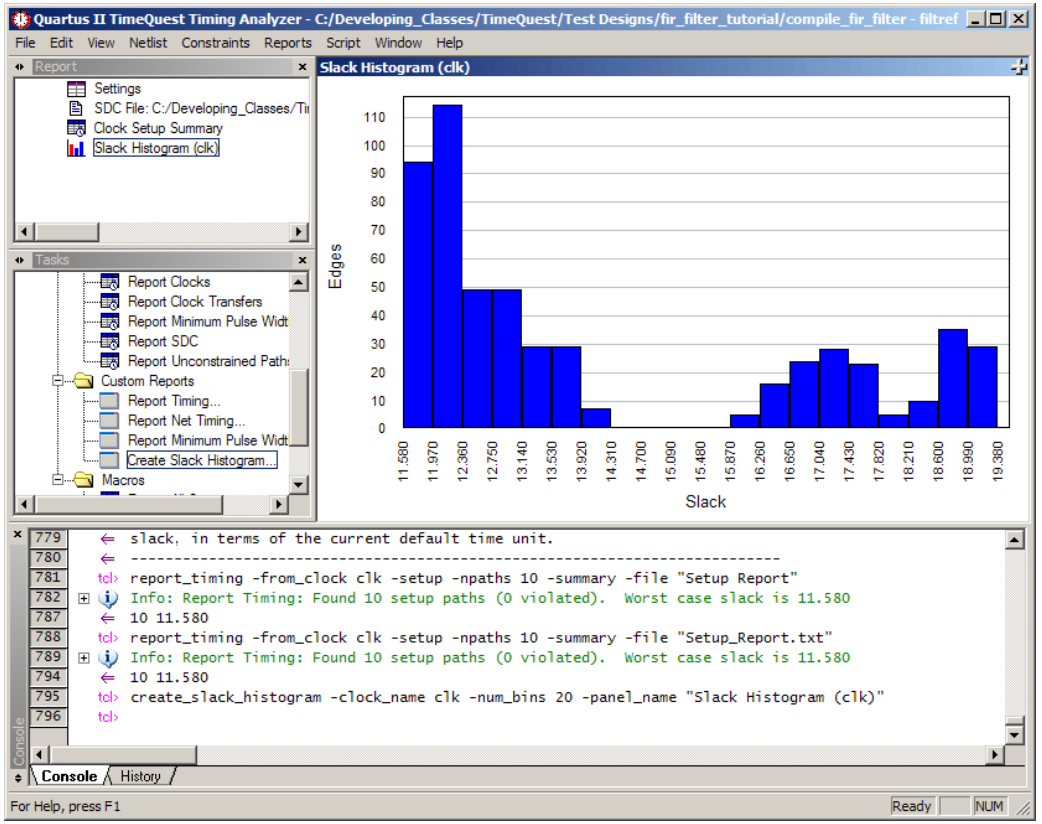
Timing Models in Detail

- Quartus II software models device timing at multiple process voltage temperature (PVT) conditions by default
  - ***Slow corner*** model
    - Indicates slowest possible performance for any single path
    - Timing for slowest device at maximum operating temperature and VCCMIN
  - ***Fast corner*** model
    - Indicates fastest possible performance for any single path
    - Timing for fastest device at minimum operating temperature and VCCMAX
  - ***2nd slow*** *and* ***2nd fast*** models (temperature inversion phenomenon)
    - Slow timing at minimum operating temperature
    - Fast timing at maximum operating temperature
      - Available only for Stratix® V devices
- Why analyze for multiple corner timing models?
  - Ensure **setup** timing is met in the **slow** models
  - Ensure **hold** timing is met in **fast** model
    - Essential for source synchronous interfaces
- Read the following white paper for more information
  - http://www.altera.com/literature/wp/wp-01139-timing-model.pdf

Timing Analysis with Timequest

TimeQuest Timing Analyzer

- New 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

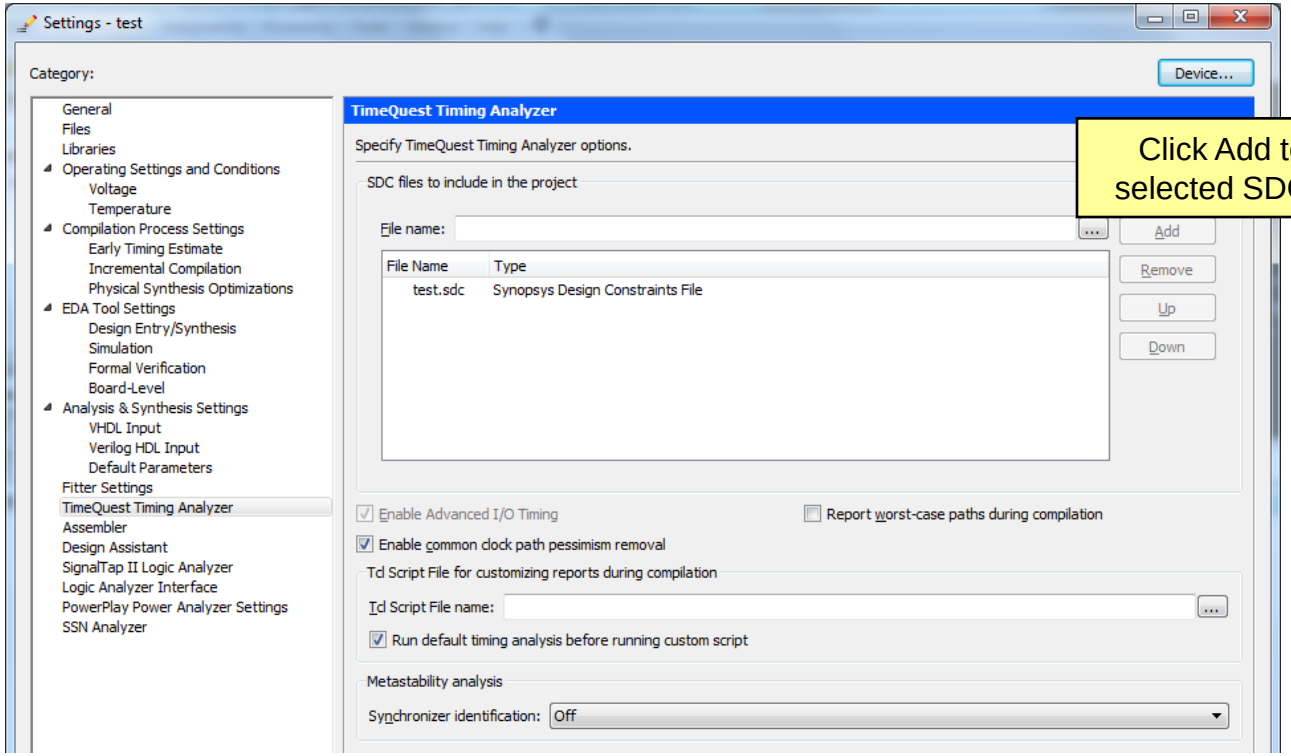


Timing Analysis Agenda

- TimeQuest flow 
- Timing constraints
- Timing reports

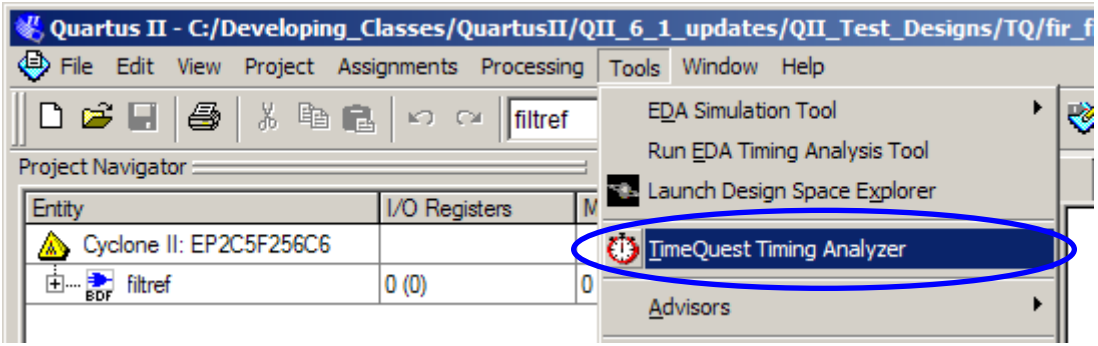
Quartus II TimeQuest Settings

- Add SDC files to TimeQuest Timing Analyzer page of Settings dialog box
- Multicorner analysis checks all process corners in one analysis
- Report worst-case paths in each clock domain
- Select Tcl script to customize report generation

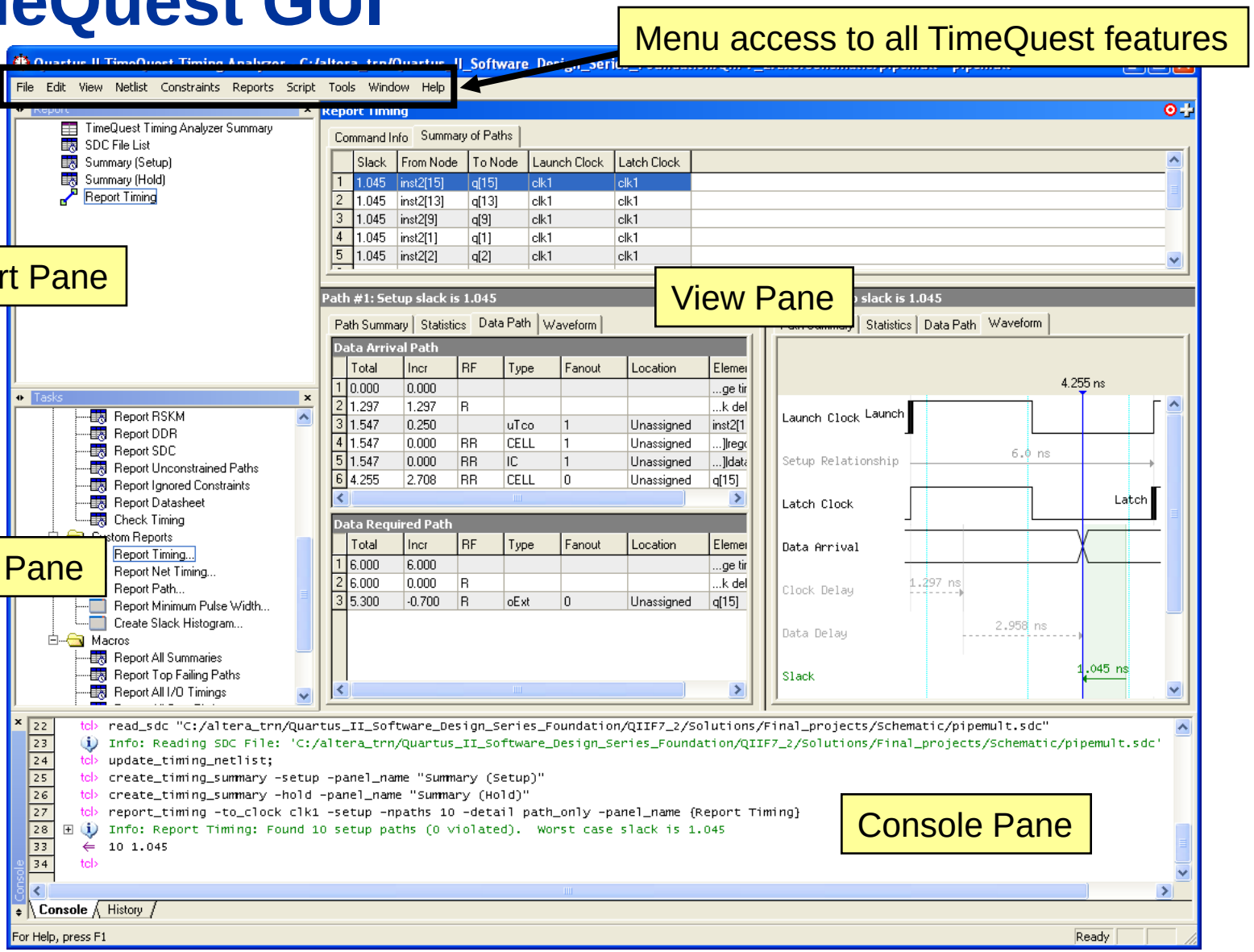


Opening TimeQuest

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

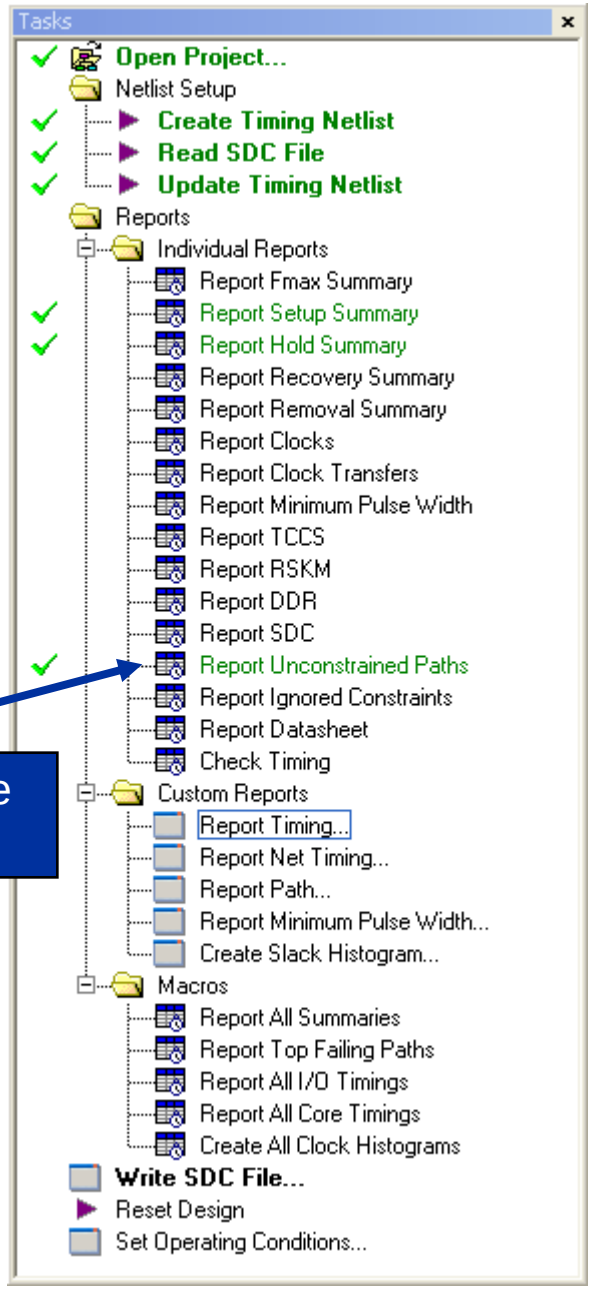


TimeQuest GUI

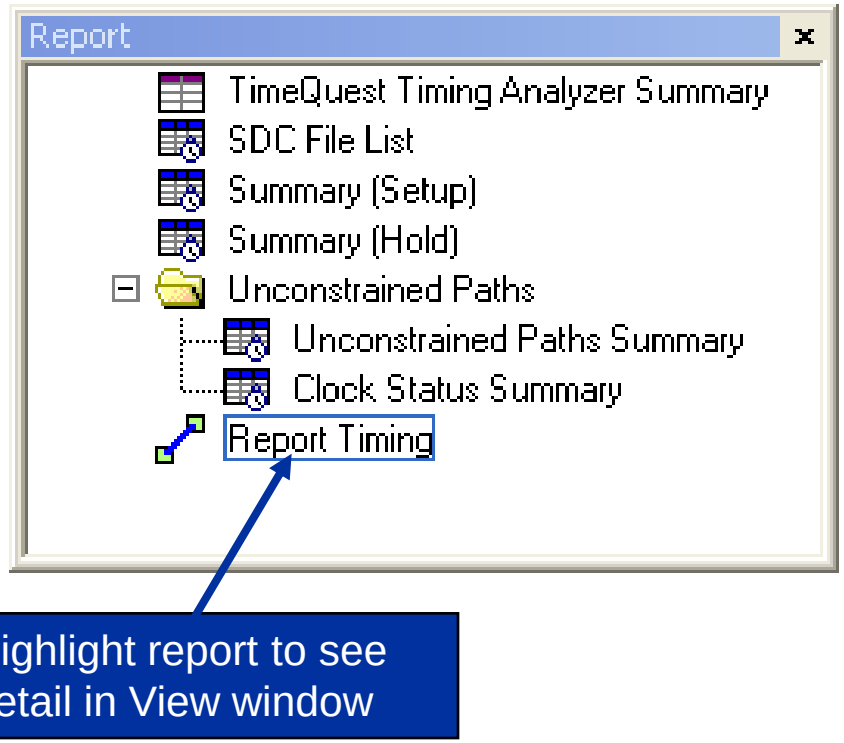


Tasks Pane

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



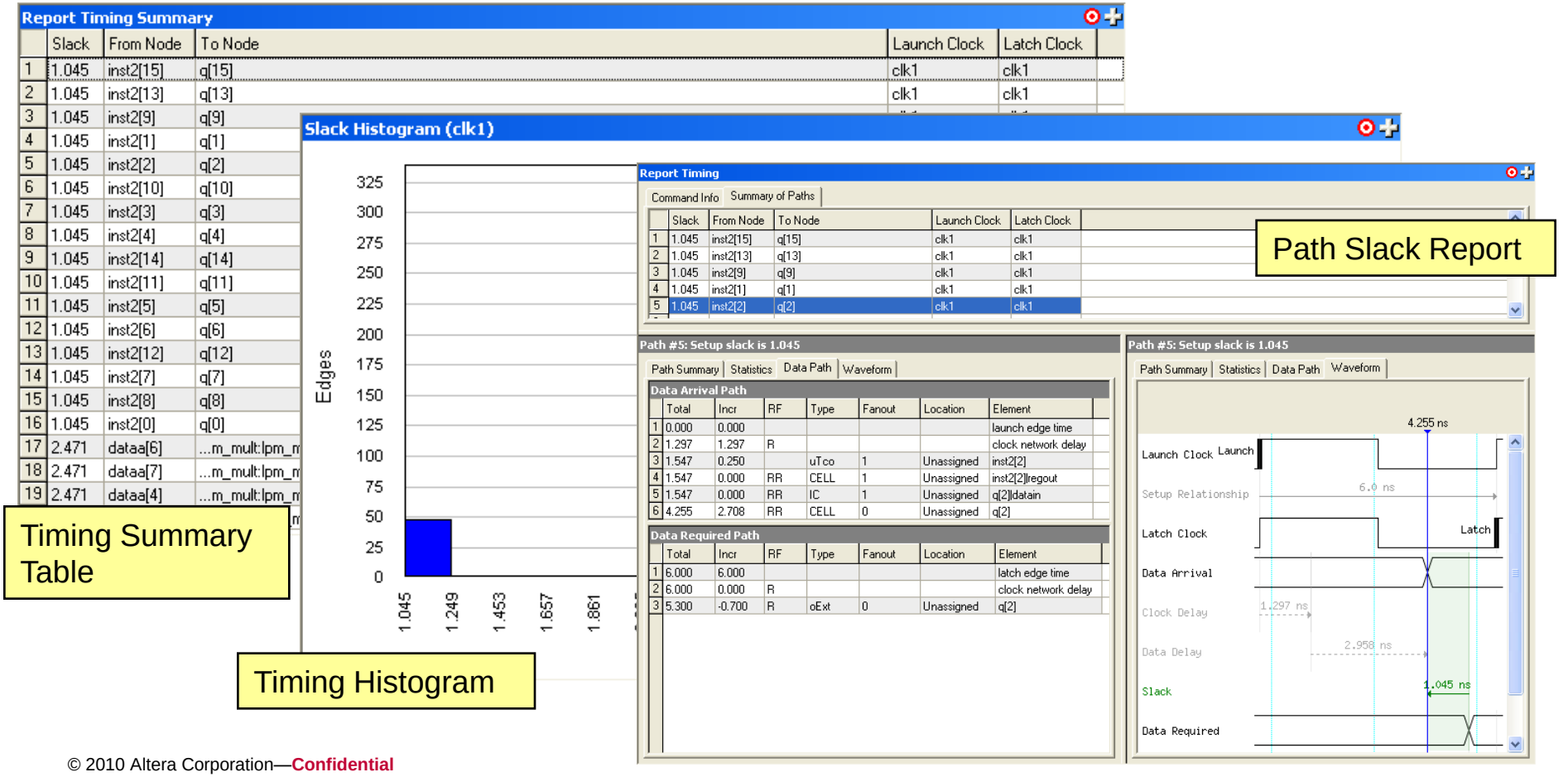
Report Pane



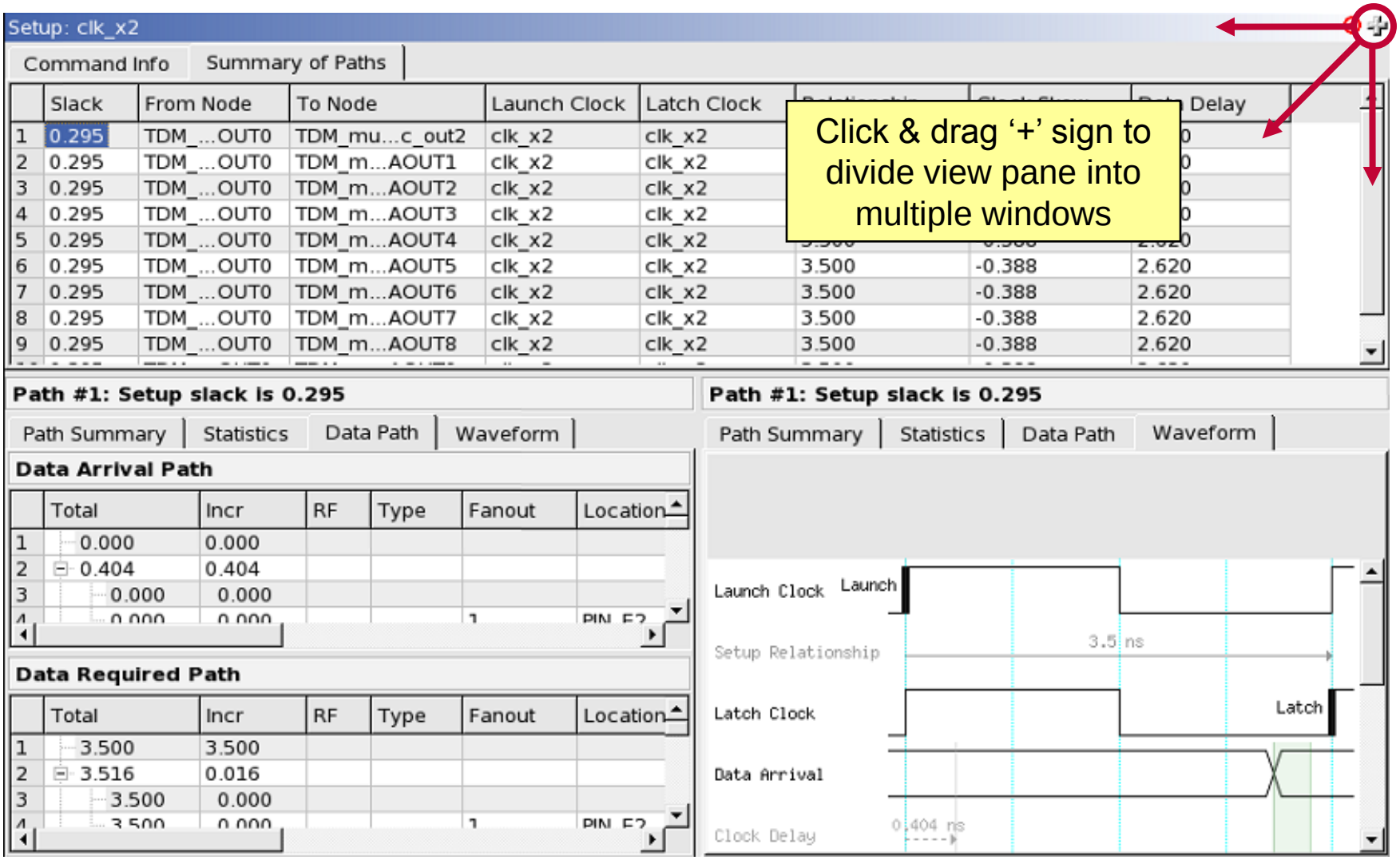
- Displays list of generated reports currently available for viewing
  - Reports generated by Tasks pane
  - Reports generated using report commands

View Pane

- Main viewing area that displays report table contents & graphical results

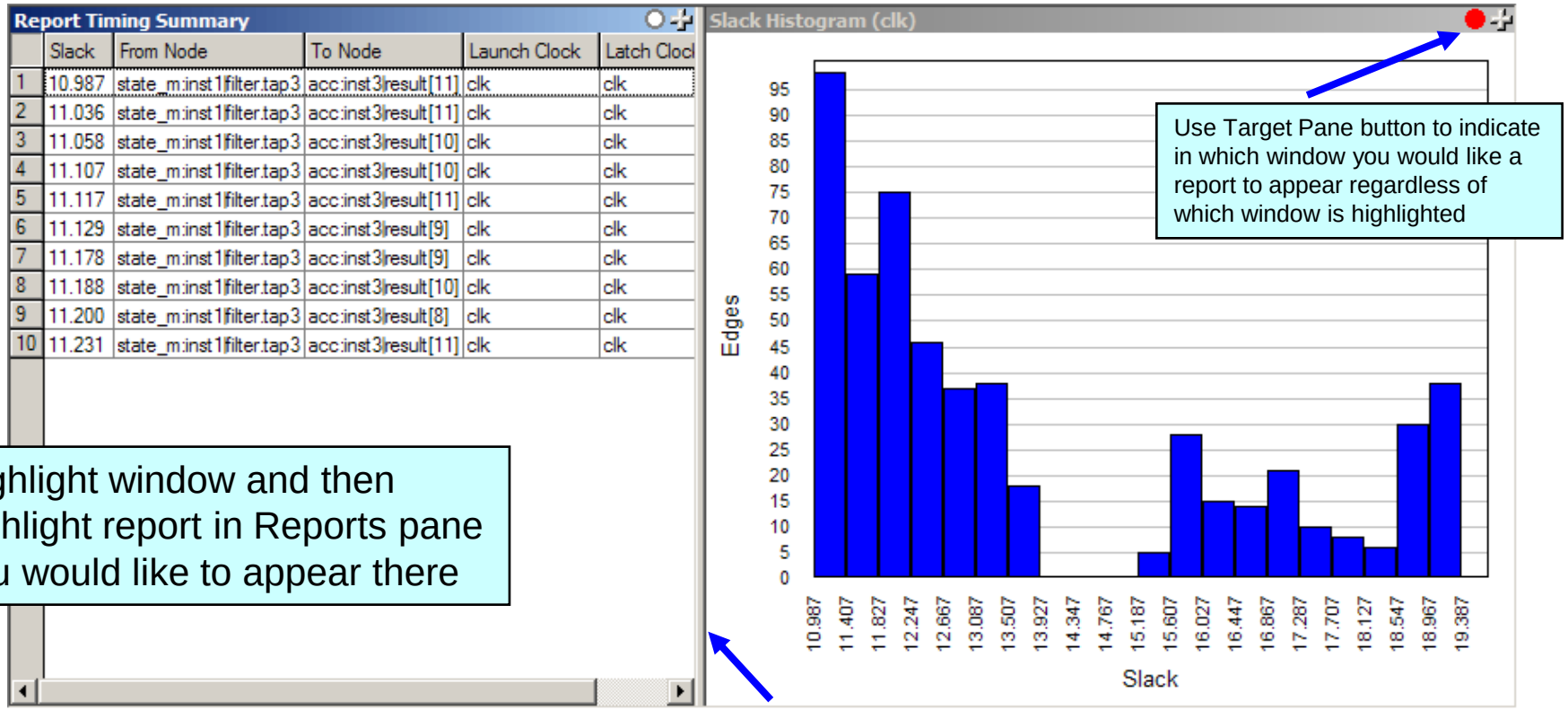


Viewing Multiple Reports



Viewing Multiple Reports Example

View pane split into two side-by-side windows



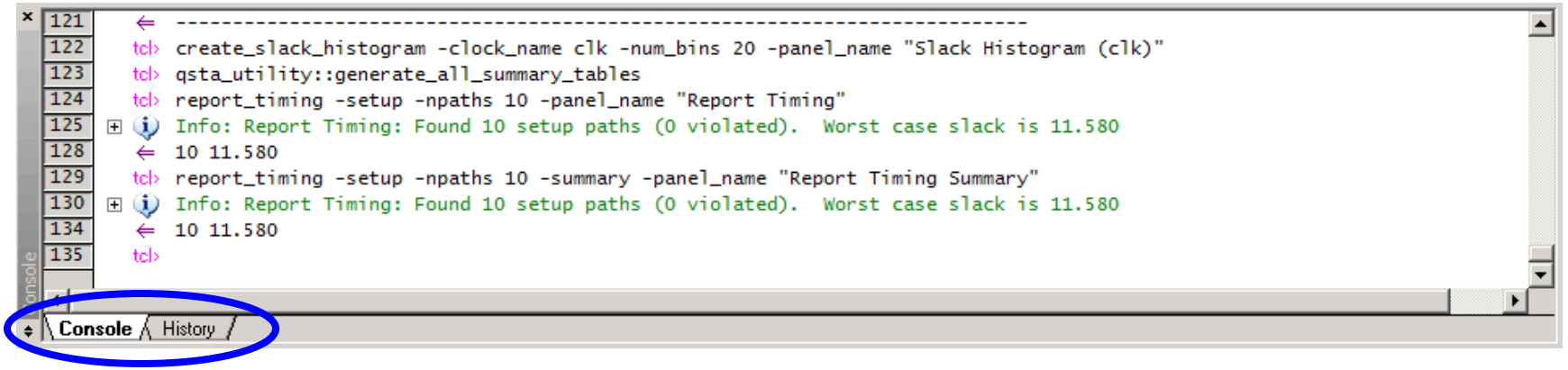
	Slack	From Node	To Node	Launch Clock	Latch Clocl
1	10.987	state_m:inst1|filter:tap3	acc:inst3|result[11]	clk	clk
2	11.036	state_m:inst1|filter:tap3	acc:inst3|result[11]	clk	clk
3	11.058	state_m:inst1|filter:tap3	acc:inst3|result[10]	clk	clk
4	11.107	state_m:inst1|filter:tap3	acc:inst3|result[10]	clk	clk
5	11.117	state_m:inst1|filter:tap3	acc:inst3|result[11]	clk	clk
6	11.129	state_m:inst1|filter:tap3	acc:inst3|result[9]	clk	clk
7	11.178	state_m:inst1|filter:tap3	acc:inst3|result[9]	clk	clk
8	11.188	state_m:inst1|filter:tap3	acc:inst3|result[10]	clk	clk
9	11.200	state_m:inst1|filter:tap3	acc:inst3|result[8]	clk	clk
10	11.231	state_m:inst1|filter:tap3	acc:inst3|result[11]	clk	clk

Highlight window and then highlight report in Reports pane you would like to appear there

Drag bar to left or right to remove split

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

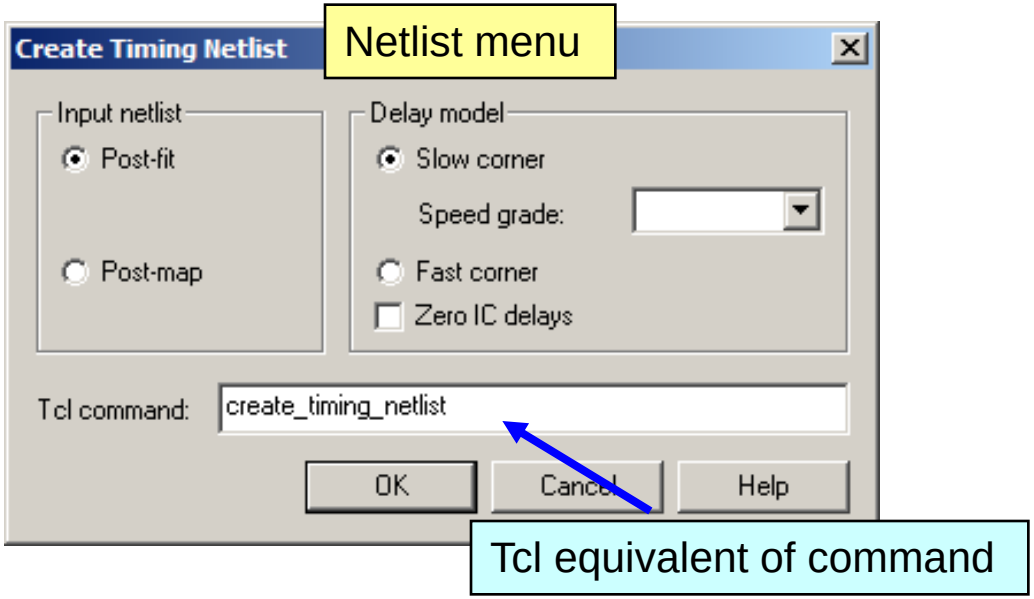


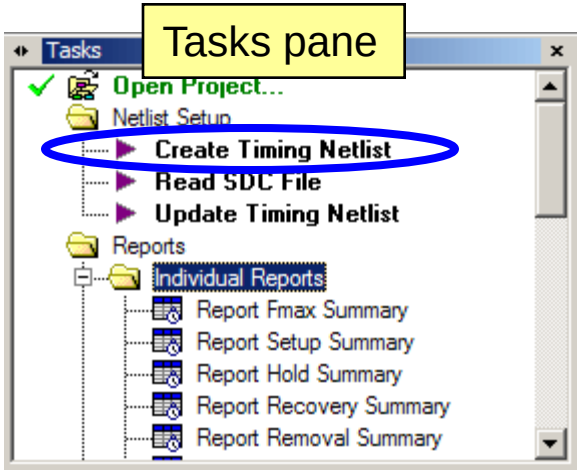
Steps to Using TimeQuest

1. Generate timing netlist
2. Read SDC file
3. Update timing netlist
4. Generate timing reports

1) Generate Timing Netlist

- Creates timing netlist (i.e. database) based on compilation results
  - Post-synthesis (mapping) or post-fit
  - Worst-case (slow) or best-case (fast) timing model
- To execute:

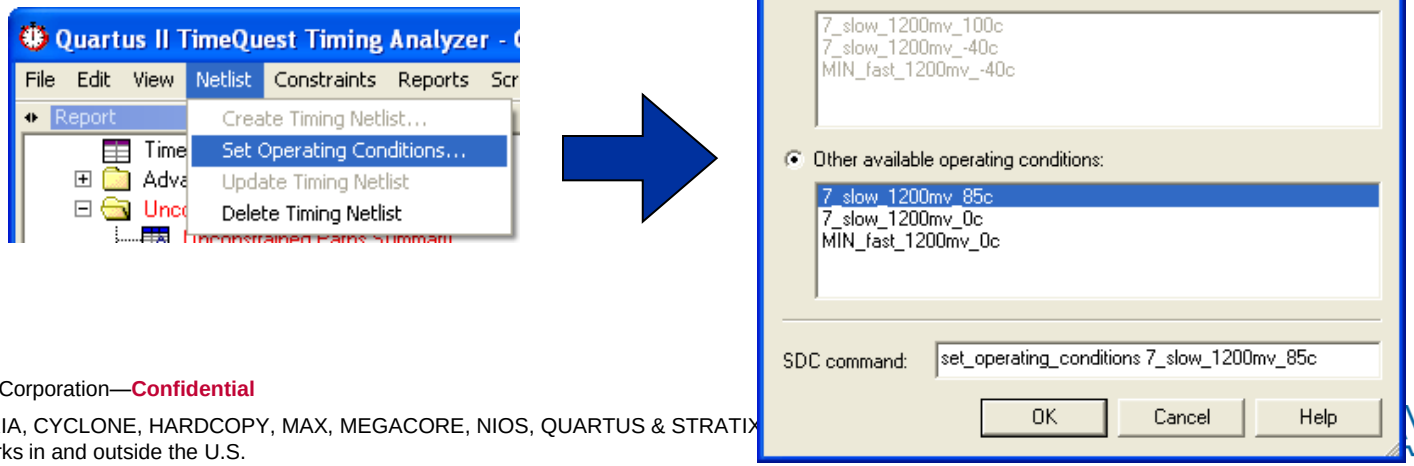




Tcl: create_timing_netlist

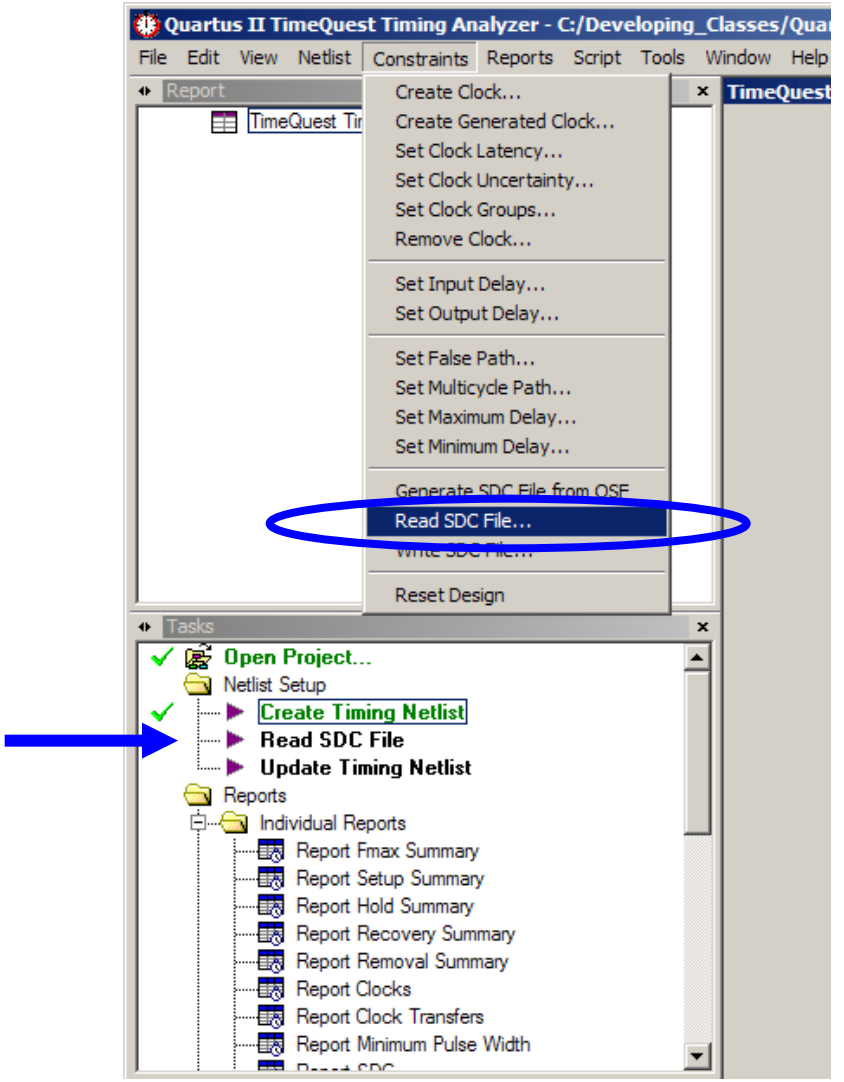
Specifying Operating Conditions

- 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



2) Read SDC File

- Reads constraints & exceptions from SDC file
  - Skip if no SDC file
- 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)



Tcl: read_sdc [<filename>]

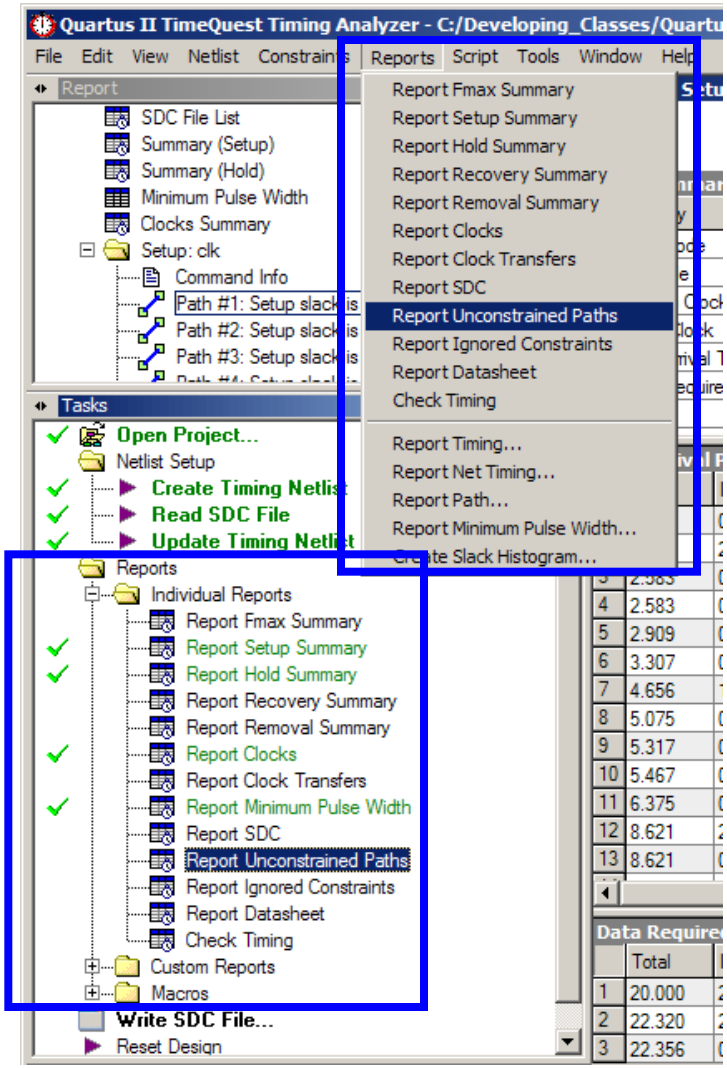
3) Update Timing Netlist

- Applies SDC constraints/exceptions to current timing netlist
- Generates warnings
  - Undefined clocks
  - Partially defined I/O delays
  - Combinatorial 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
    - Automatically creates/updates netlist & reads default SDC file if needed
  - Reports menu
    - Must have valid netlist to access
    - Tasks pane or Reports menu



Double-click on individual report

Reset Design Command

- Located in Tasks pane
- 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

Steps to Using TimeQuest (Review)

1. Generate timing netlist
2. Read SDC file
3. Update timing netlist
4. Generate timing reports

(Optionally)

5. Modify SDC file
6. Reset Design and go back to step 2

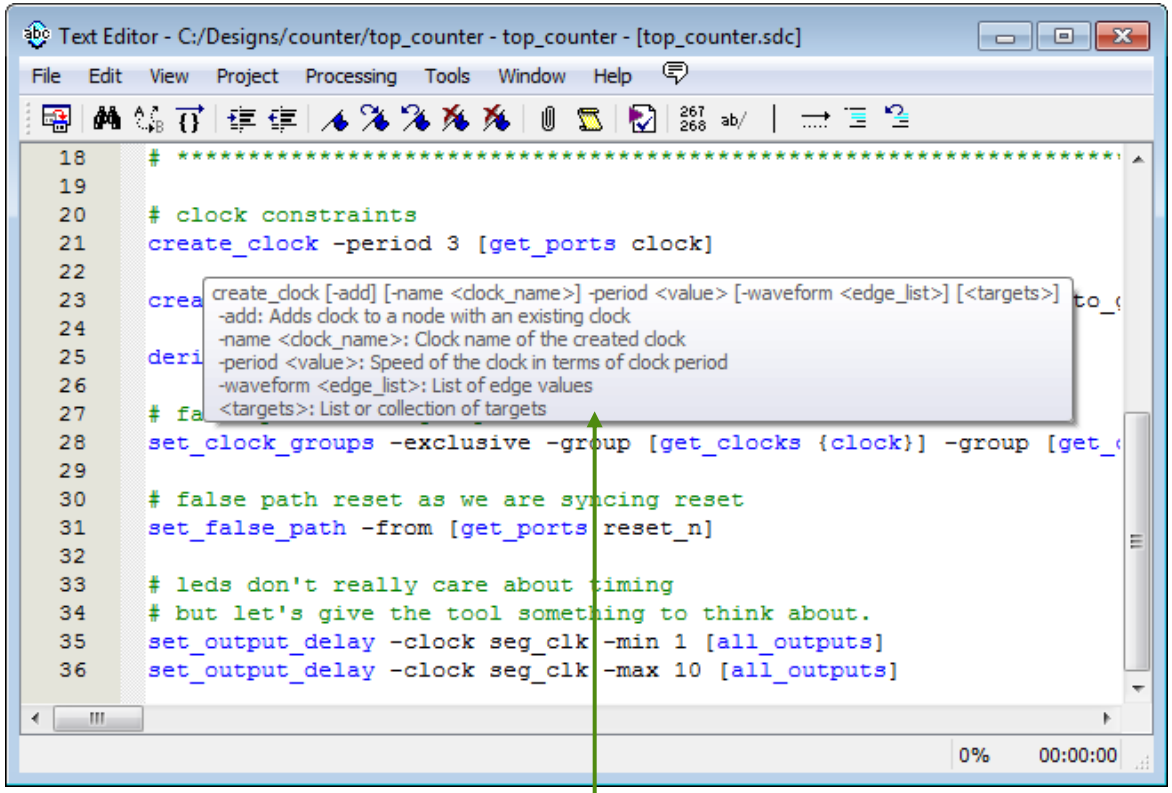
Timing Analysis Agenda

- TimeQuest flow
- Timing constraints 
- Timing reports

SDC File Editor = Quartus II Text Editor

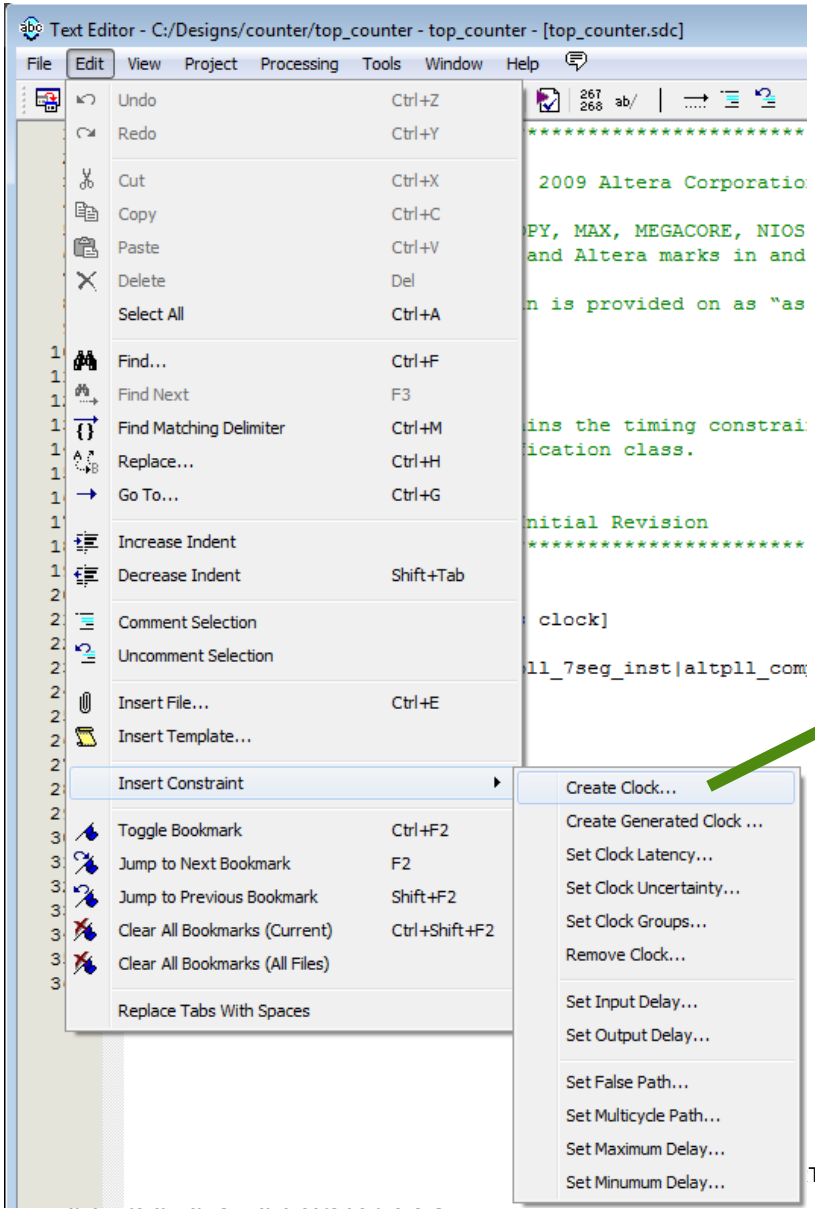
- Use Quartus II editor to create and/or edit SDC
- SDC editing unique features (for .sdc files)
  - Access to GUI dialog boxes for constraint entry (**Edit ⇒ Insert Constraint**)
  - Syntax coloring
  - Tooltip syntax help
  - SDC templates

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

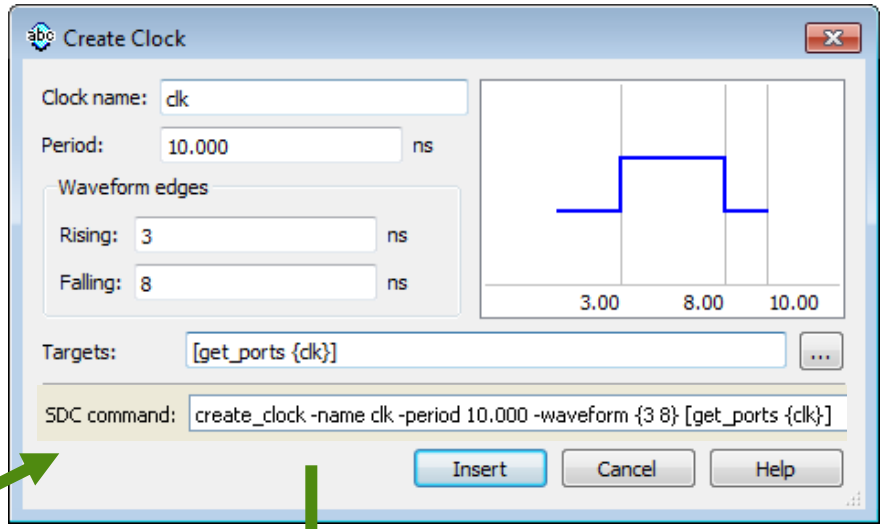


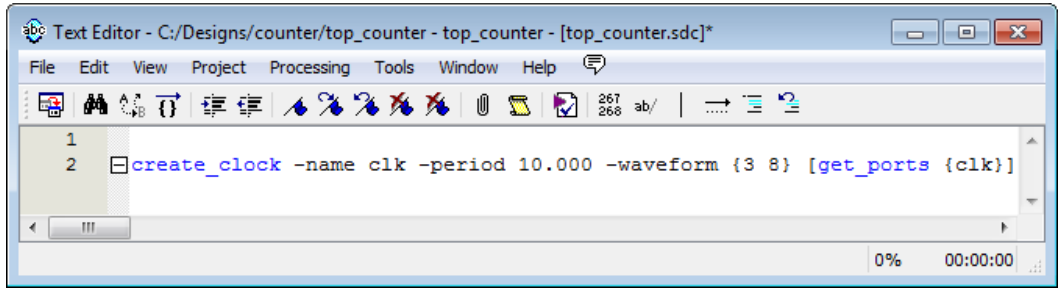
Place cursor over command to see tooltip

SDC File Editor (cont.)



Construct an SDC file using TimeQuest graphical constraint creation tools

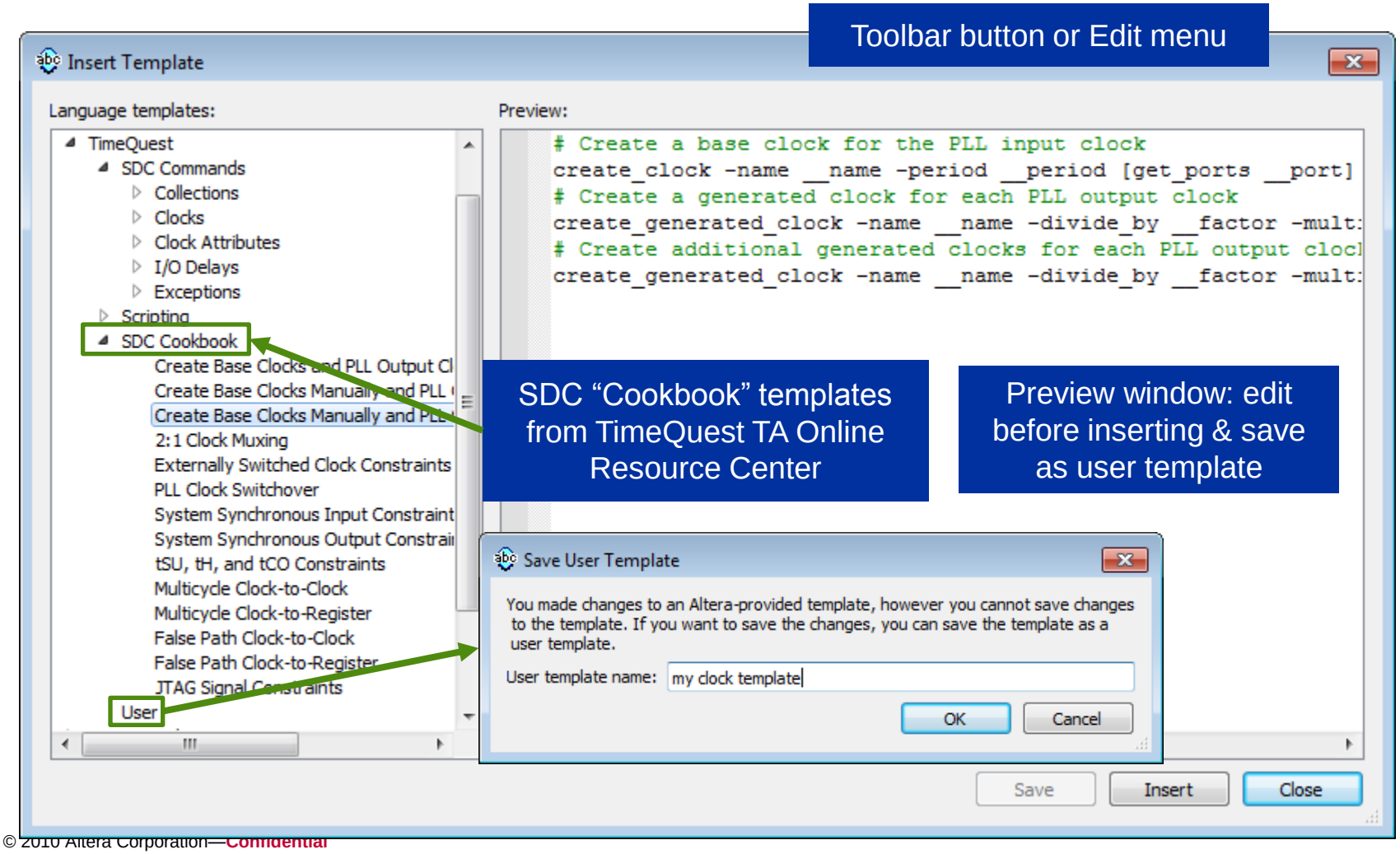




Constraints inserted at cursor location

SDC Templates

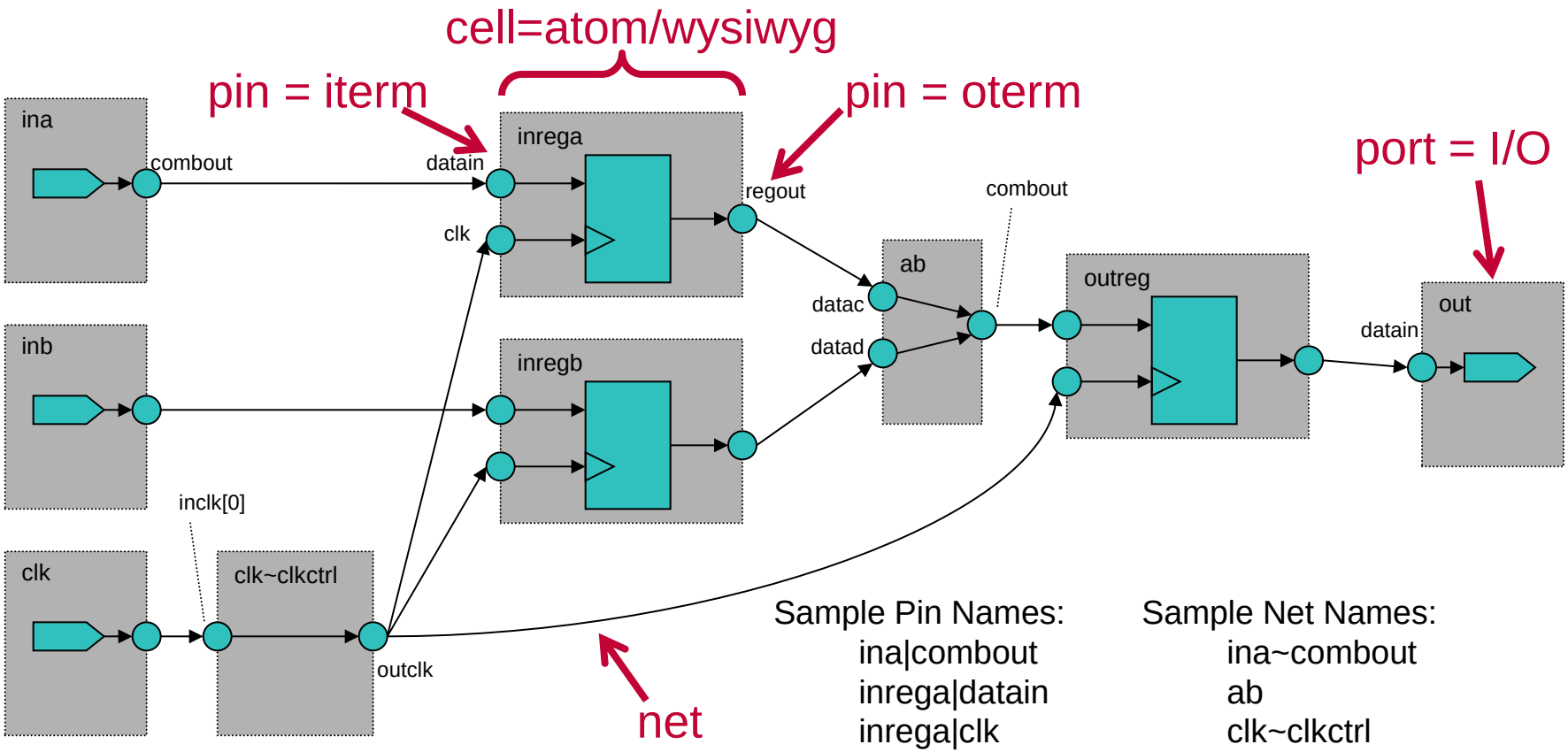
- Quickly add customized constraint templates



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



Sample Pin Names:

- ina|combout
- inrega|datain
- inrega|clk
- inrega|regout
- ab|combout
- ab|datac

Sample Net Names:

- ina~combout
- ab
- clk~clkctrl
- inrega

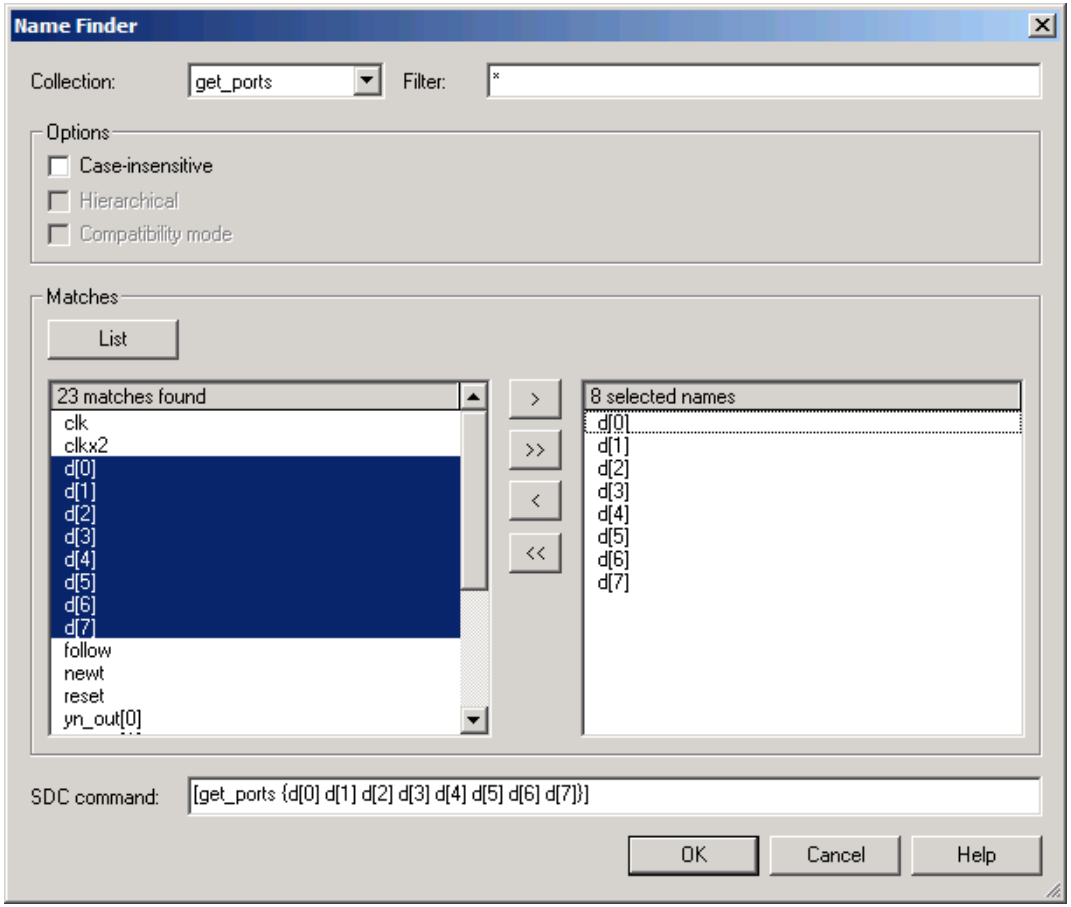
Collections

- Searches and returns from the design netlist with a list of names meeting criteria
- Used in SDC commands
  - Some collections searched automatically during commands 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 for a complete list & description of each

Name Finder

Clicking on Browse button opens Name Finder allowing you to search netlist for node names



SDC: create_clock

Name Finder Search Options

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

SDC Timing Constraints

- Clocks 
- I/O
- False paths
- Multicycle paths
- Absolute delays

Clocks in SDC

- 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.
      - Clock inversions automatically detected unless derived from more complex logic structure
- All are related by default
  - Cross-domain transfers analyzed

Clock Constraints

- Create Clock
- Create Generated Clock
- Clock Uncertainty
- Clock Latency
- PLL clocks

Creating a Clock

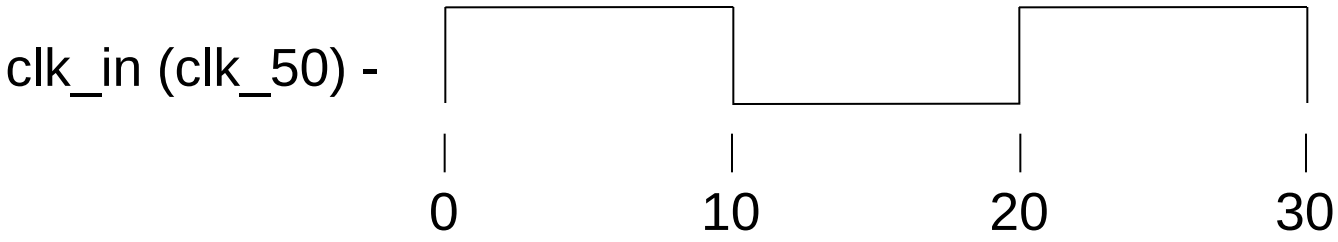
- Command: `create_clock`
- Options

```
[-name <clock_name>]
-period <time>
[-waveform {<rise_time> <fall_time>}]
[<targets>]
[-add]
```

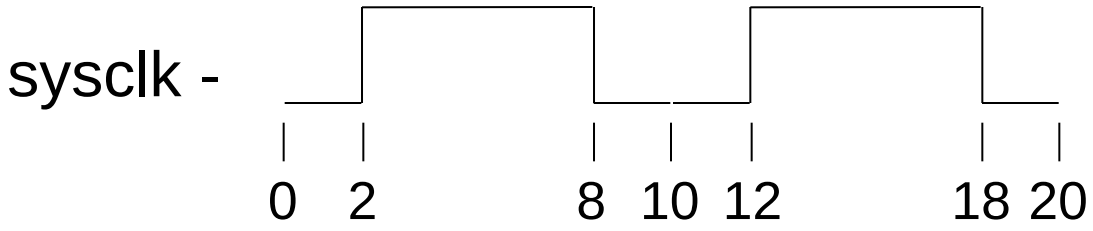
[] = optional

create_clock Examples

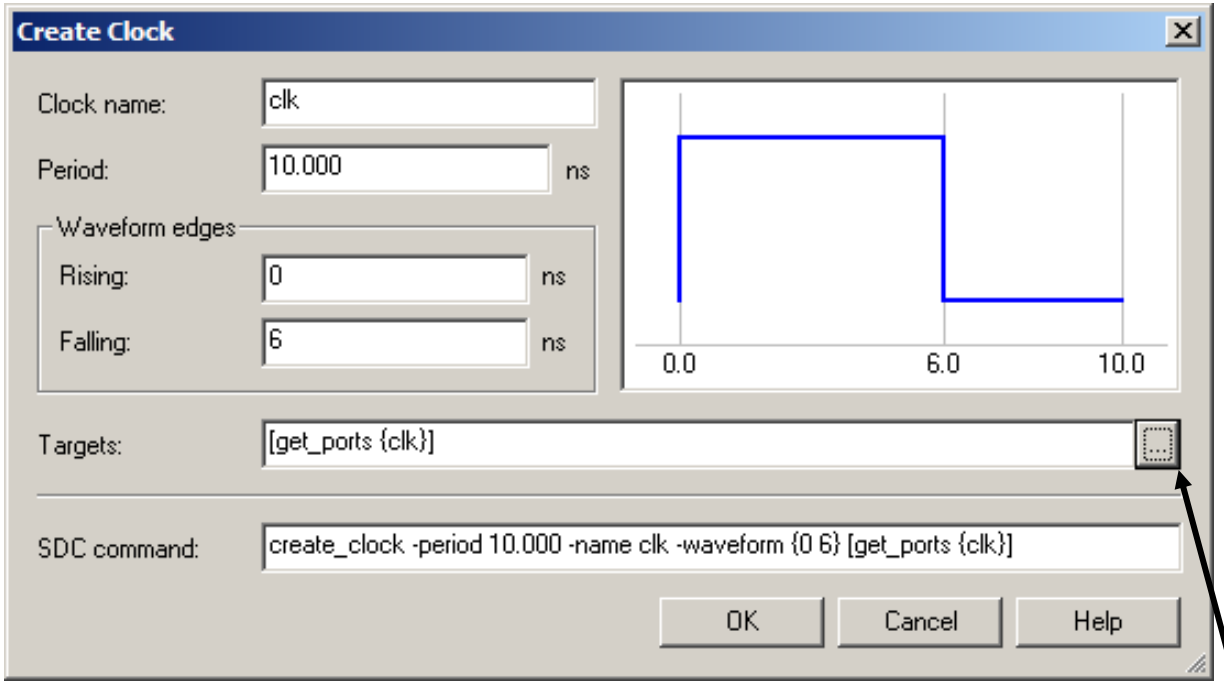
```
create_clock -period 20.0 -name clk_50 [get_ports clk_in]
```




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



Create Clock using GUI



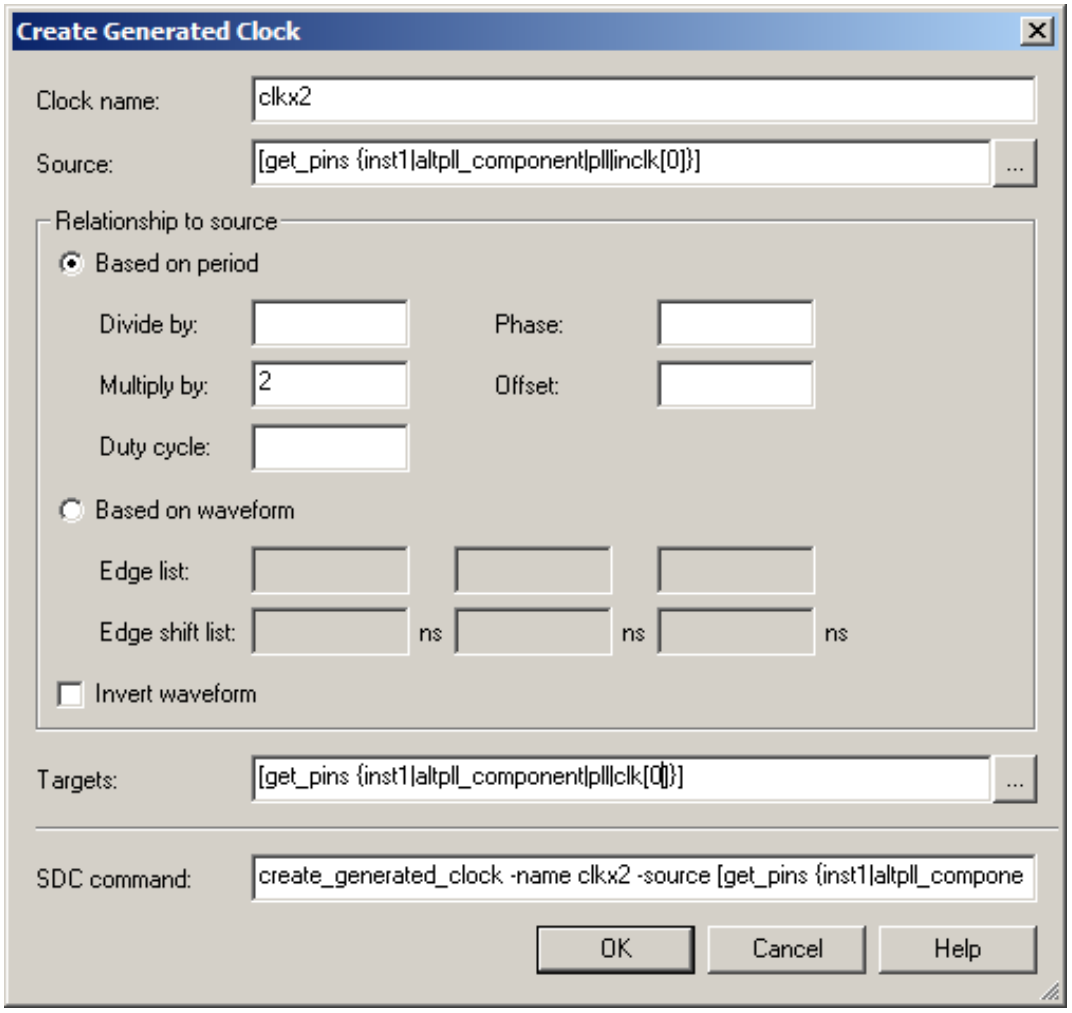
Use Name Finder to search collections

Creating a Generated Clock

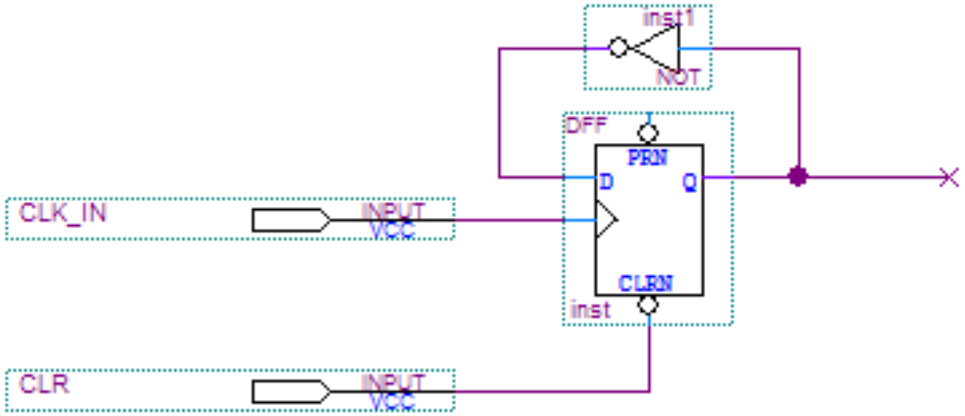
- Command: `create_generated_clock`
- Options

```
[-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 using GUI



Generated Clock Example

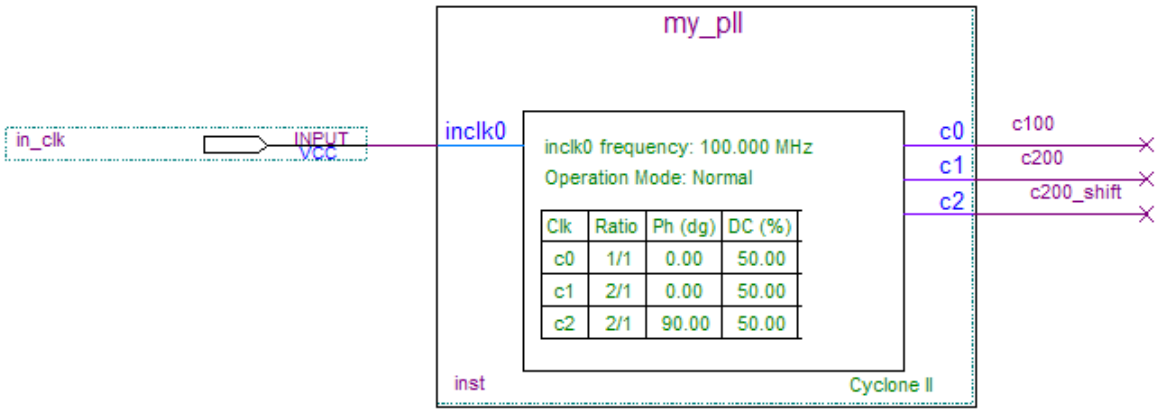



```
create_clock -period 10 [get_ports clk_in]

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

PLL Clocks

- Command: `derive_pll_clocks`
- Use to create generated clocks on all PLL outputs
  - Based on input clock & PLL settings
- Requires defining PLL input as clock

derive_pll_clocks Example



Using generated clock commands

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

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

SDC Timing Constraints

- Clocks
- I/O 
- False paths
- Multicycle paths
- Absolute delays

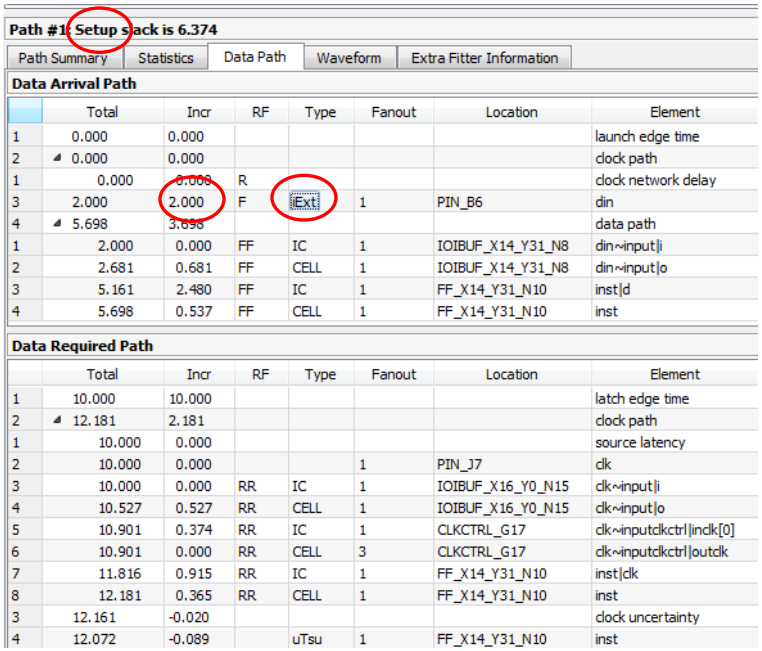
I/O Constraining

- Specify system-level timing constraints
- Settings
  - Input maximum delay
  - Input minimum delay
  - Output maximum delay
  - Output minimum delay

Input Min/Max Delay Definition

- External delays added to the input data arrival path
- Max value used in Setup and Min in Hold

Input Maximum Delay = 2ns



Path #1: Setup slack is 6.374

Path Summary | Statistics | Data Path | Waveform | Extra Fitter Information

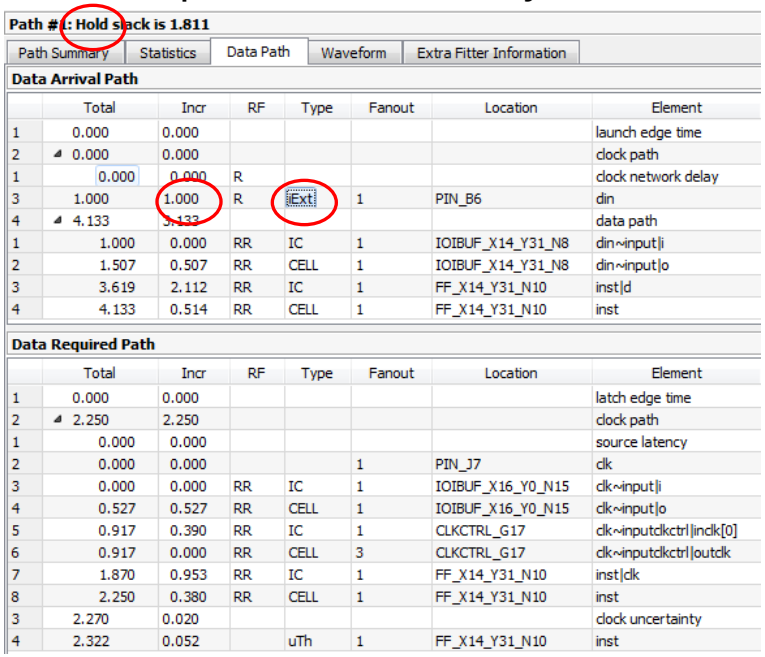
Data Arrival Path

	Total	Incr	RF	Type	Fanout	Location	Element
1	0.000	0.000					launch edge time
2	◢ 0.000	0.000					clock path
1	0.000	0.000	R				clock network delay
3	2.000	2.000	F	iExt	1	PIN_B6	din
4	◢ 5.698	3.698					data path
1	2.000	0.000	FF	IC	1	IOIBUF_X14_Y31_N8	din~input|i
2	2.681	0.681	FF	CELL	1	IOIBUF_X14_Y31_N8	din~input|o
3	5.161	2.480	FF	IC	1	FF_X14_Y31_N10	inst|d
4	5.698	0.537	FF	CELL	1	FF_X14_Y31_N10	inst

Data Required Path

	Total	Incr	RF	Type	Fanout	Location	Element
1	10.000	10.000					latch edge time
2	◢ 12.181	2.181					clock path
1	10.000	0.000					source latency
2	10.000	0.000			1	PIN_J7	clk
3	10.000	0.000	RR	IC	1	IOIBUF_X16_Y0_N15	clk~input|i
4	10.527	0.527	RR	CELL	1	IOIBUF_X16_Y0_N15	clk~input|o
5	10.901	0.374	RR	IC	1	CLKCTRL_G17	clk~inputclkctrl|inclk[0]
6	10.901	0.000	RR	CELL	3	CLKCTRL_G17	clk~inputclkctrl|outclk
7	11.816	0.915	RR	IC	1	FF_X14_Y31_N10	inst|clk
8	12.181	0.365	RR	CELL	1	FF_X14_Y31_N10	inst
3	12.161	-0.020					clock uncertainty
4	12.072	-0.089		uTsu	1	FF_X14_Y31_N10	inst

Input Minimum Delay = 1ns



Path #1: Hold slack is 1.811

Path Summary | Statistics | Data Path | Waveform | Extra Fitter Information

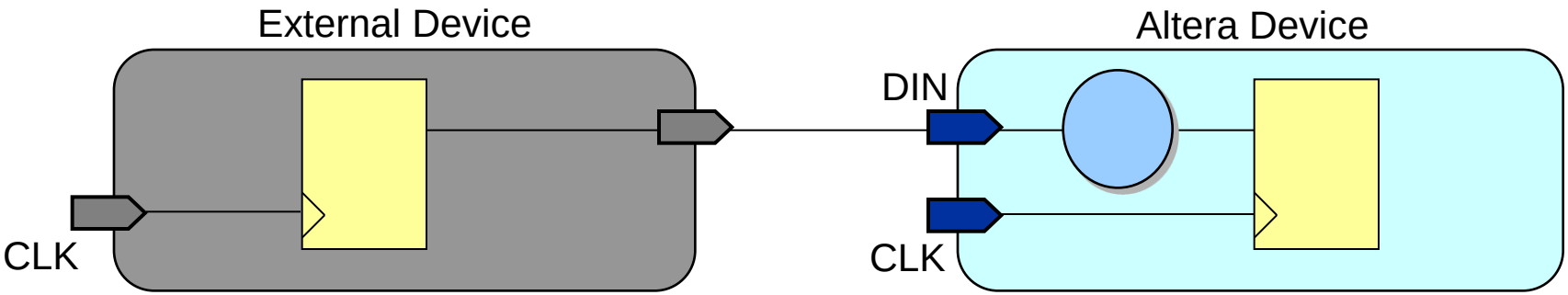
Data Arrival Path

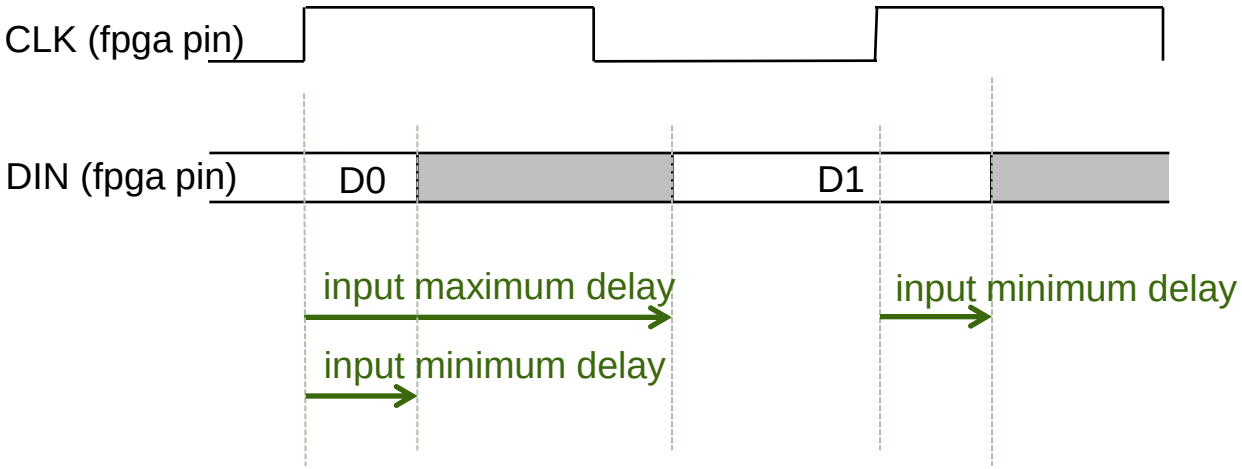
	Total	Incr	RF	Type	Fanout	Location	Element
1	0.000	0.000					launch edge time
2	◢ 0.000	0.000					clock path
1	0.000	0.000	R				clock network delay
3	1.000	1.000	R	iExt	1	PIN_B6	din
4	◢ 4.133	3.133					data path
1	1.000	0.000	RR	IC	1	IOIBUF_X14_Y31_N8	din~input|i
2	1.507	0.507	RR	CELL	1	IOIBUF_X14_Y31_N8	din~input|o
3	3.619	2.112	RR	IC	1	FF_X14_Y31_N10	inst|d
4	4.133	0.514	RR	CELL	1	FF_X14_Y31_N10	inst

Data Required Path

	Total	Incr	RF	Type	Fanout	Location	Element
1	0.000	0.000					latch edge time
2	◢ 2.250	2.250					clock path
1	0.000	0.000					source latency
2	0.000	0.000			1	PIN_J7	clk
3	0.000	0.000	RR	IC	1	IOIBUF_X16_Y0_N15	clk~input|i
4	0.527	0.527	RR	CELL	1	IOIBUF_X16_Y0_N15	clk~input|o
5	0.917	0.390	RR	IC	1	CLKCTRL_G17	clk~inputclkctrl|inclk[0]
6	0.917	0.000	RR	CELL	3	CLKCTRL_G17	clk~inputclkctrl|outclk
7	1.870	0.953	RR	IC	1	FF_X14_Y31_N10	inst|clk
8	2.250	0.380	RR	CELL	1	FF_X14_Y31_N10	inst
3	2.270	0.020					clock uncertainty
4	2.322	0.052		uTh	1	FF_X14_Y31_N10	inst

Input Min/Max Delay in Waveform



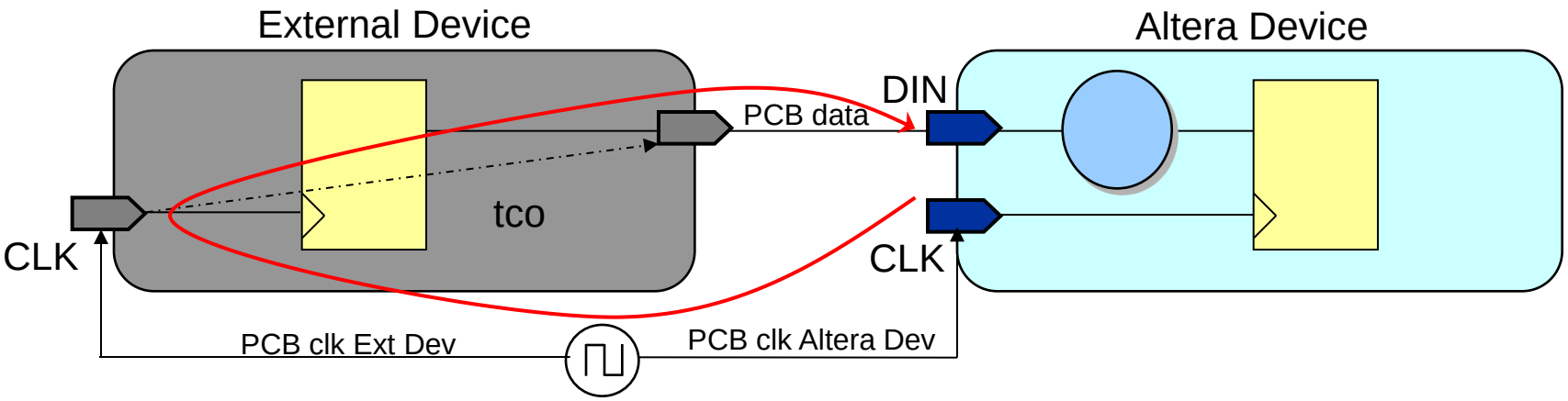


Tsetup(Altera) = Setup Relationship (=Tck) – Input Maximum Delay

Thold(Altera) = – Hold Relationship (=0) + Input Minimum Delay

Input Maximum Delay Calculation (Example)

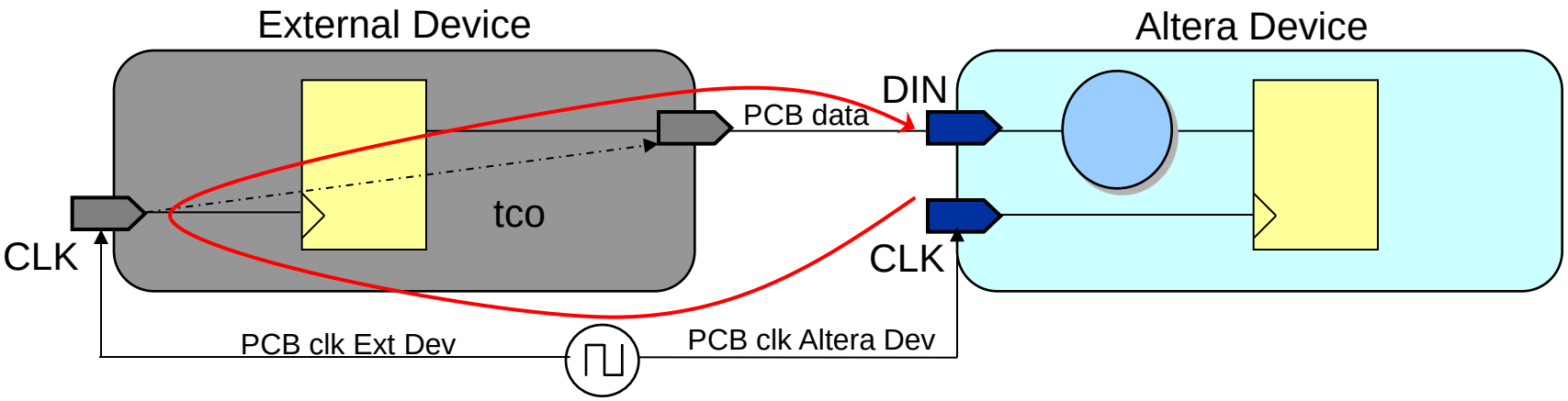
- Constraining registered input path in setup



- Input Maximum Delay =
  - PCB clk(Altera Dev) + PCB clk(Ext Dev)
  + Tco max(Ext Dev)
  + PCB data max

Input Minimum Delay Calculation (Example)

- Constraining registered input path in hold



- Input Minimum Delay =
  - PCB clk(Altera Dev) + PCB clk(Ext Dev)
  + Tco min(Ext Dev)
  + PCB data min

Output Min/Max Delay Definition

- External delays added to the output data arrival path
  - in fact, subtracted to the output data required path !
- Max value used in Setup and Min in Hold analysis

Output Maximum Delay = 2ns

Path #1: Setup slack is 2.145

Path Summary | Statistics | Data Path | Waveform | Extra Fitter Information

Data Arrival Path

	Total	Incr	RF	Type	Fanout	Location	Element
1	0.000	0.000					launch edge time
2	2.332	2.332					clock path
1	0.000	0.000					source latency
2	0.000	0.000			1	PIN_J7	clk
3	0.000	0.000	RR	IC	1	IOIBUF_X16_Y0_N15	clk~input|i
4	0.527	0.527	RR	CELL	1	IOIBUF_X16_Y0_N15	clk~input|o
5	0.917	0.390	RR	IC	1	CLKCTRL_G17	clk~inputclkctrl|inclk[0]
6	0.917	0.000	RR	CELL	3	CLKCTRL_G17	clk~inputclkctrl|outclk
7	1.870	0.953	RR	IC	1	DDIOOUTCELL_X12_Y31_N4	inst2|clk

Data Required Path

	Total	Incr	RF	Type	Fanout	Location	Element
1	10.000	10.000					latch edge time
2	10.000	0.000					clock path
1	10.000	0.000	R				clock network delay
3	9.980	-0.020					clock uncertainty
4	7.980	-2.000	R	oExt	0	PIN_A7	dout

Output Minimum Delay = -1ns

Path #1: Hold slack is 4.446

Path Summary | Statistics | Data Path | Waveform | Extra Fitter Information

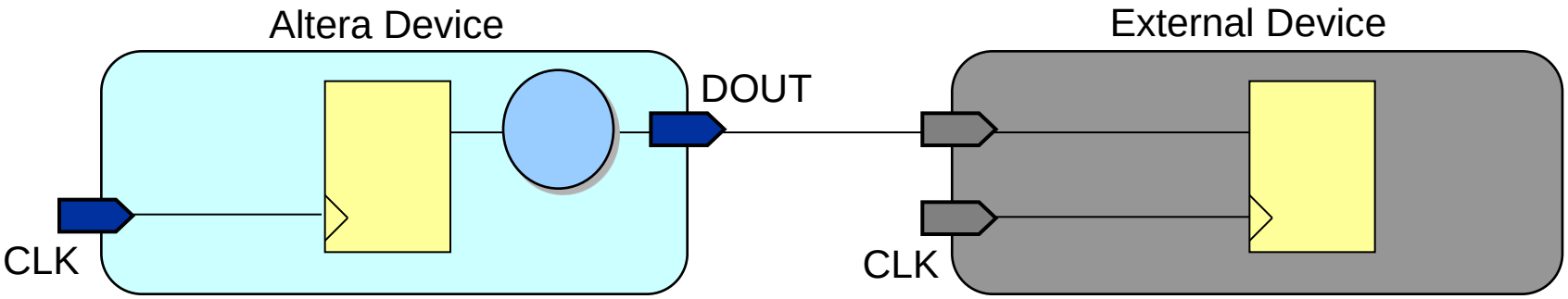
Data Arrival Path

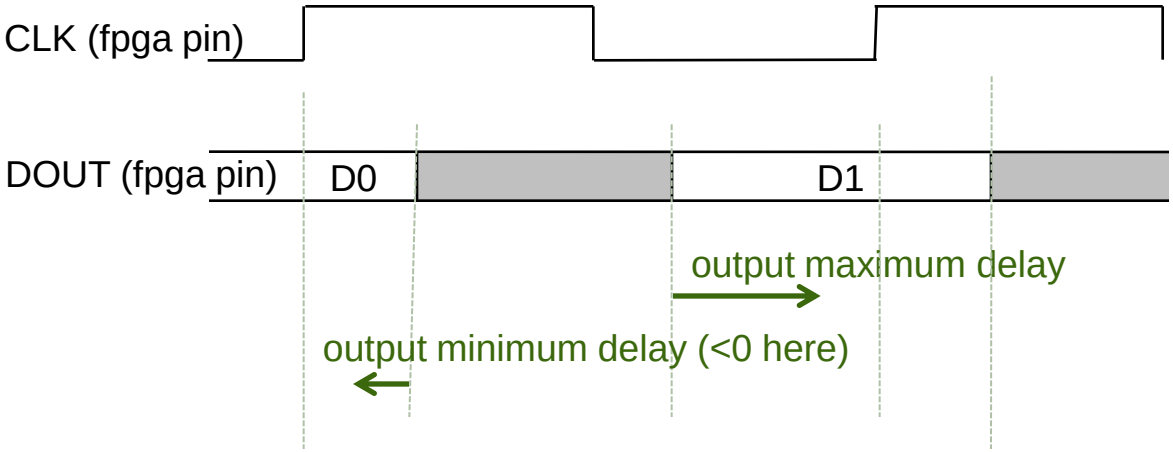
	Total	Incr	RF	Type	Fanout	Location	Element
1	0.000	0.000					launch edge time
2	2.260	2.260					clock path
1	0.000	0.000					source latency
2	0.000	0.000			1	PIN_J7	clk
3	0.000	0.000	RR	IC	1	IOIBUF_X16_Y0_N15	clk~input|i
4	0.527	0.527	RR	CELL	1	IOIBUF_X16_Y0_N15	clk~input|o
5	0.901	0.374	RR	IC	1	CLKCTRL_G17	clk~inputclkctrl|inclk[0]
6	0.901	0.000	RR	CELL	3	CLKCTRL_G17	clk~inputclkctrl|outclk
7	1.816	0.915	RR	IC	1	DDIOOUTCELL_X12_Y31_N4	inst2|clk

Data Required Path

	Total	Incr	RF	Type	Fanout	Location	Element
1	0.000	0.000					latch edge time
2	0.000	0.000					clock path
1	0.000	0.000	R				clock network delay
3	0.020	0.020					clock uncertainty
4	1.020	1.000	F	oExt	0	PIN_A7	dout

Output Min/Max Delay in Waveform



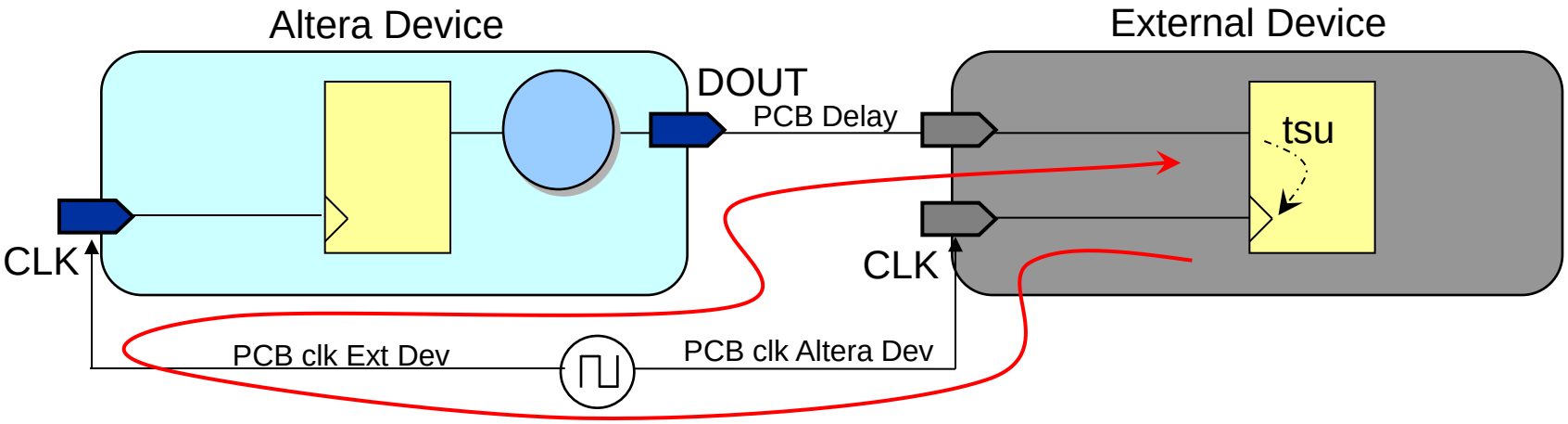


Tco max(Altera) + Output Maximum Delay ≤ Setup Relationship (=Tck)

Tco min(Altera) + Output Minimum Delay ≥ Hold Relationship (=0)

Output Maximum Delay Calculation (Example)

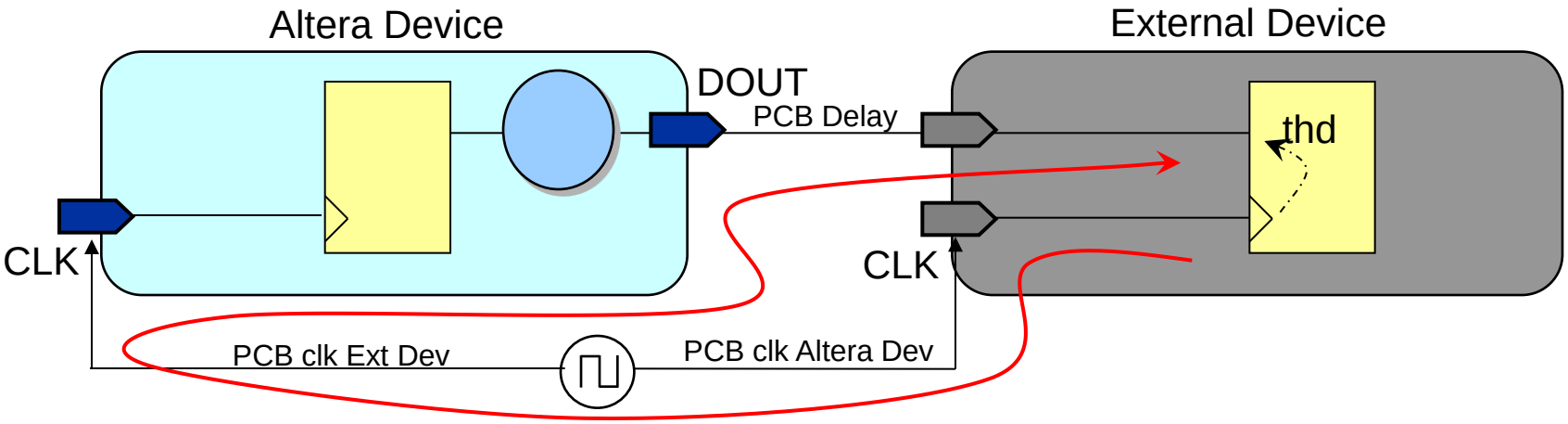
- Constraining registered output path in setup



- Output Maximum Delay =
  - PCB clk(Ext Dev) + PCB clk(Altera Dev)
  + PCB data max
  + Tsu(Ext Dev)

Output Minimum Delay Calculation (Example)

- Constraining registered output path in hold



- Output Minimum Delay =
  - PCB clk(Ext Dev) + PCB clk(Altera Dev)
  + PCB data min
  - Thd(Ext Dev)

`set_input_delay` Command

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

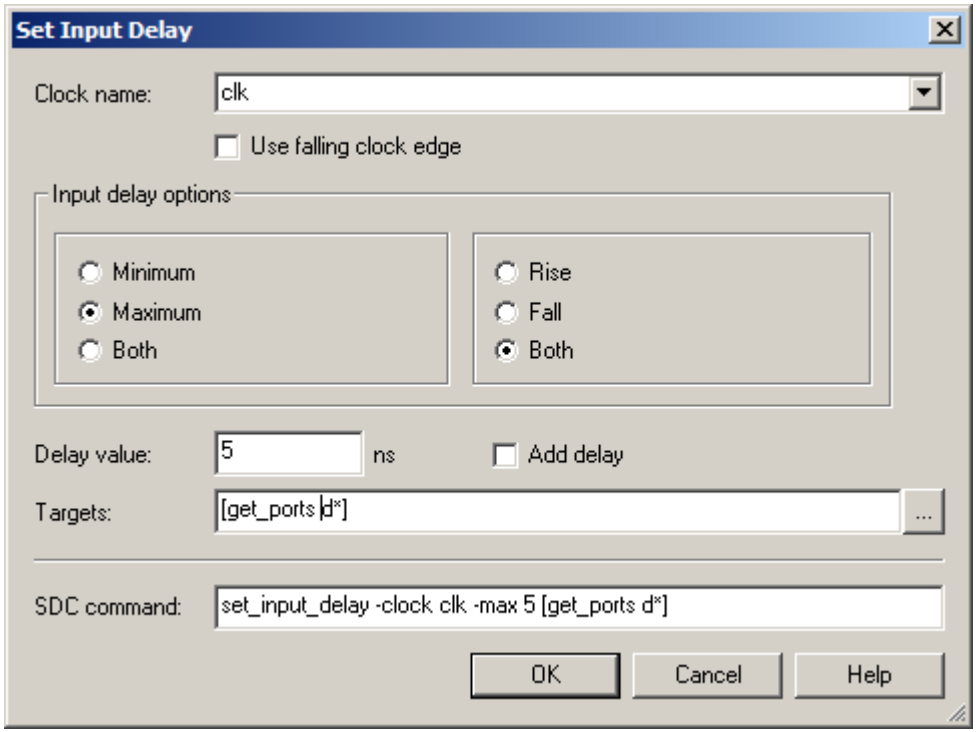
```
-clock <clock_name>
[-clock_fall]
[-rise | -fall]
[-max | -min]
[-add_delay]
[-reference_pin <target>]
[-source_latency_included]
<delay value>
<targets>
```

`set_output_delay` Command

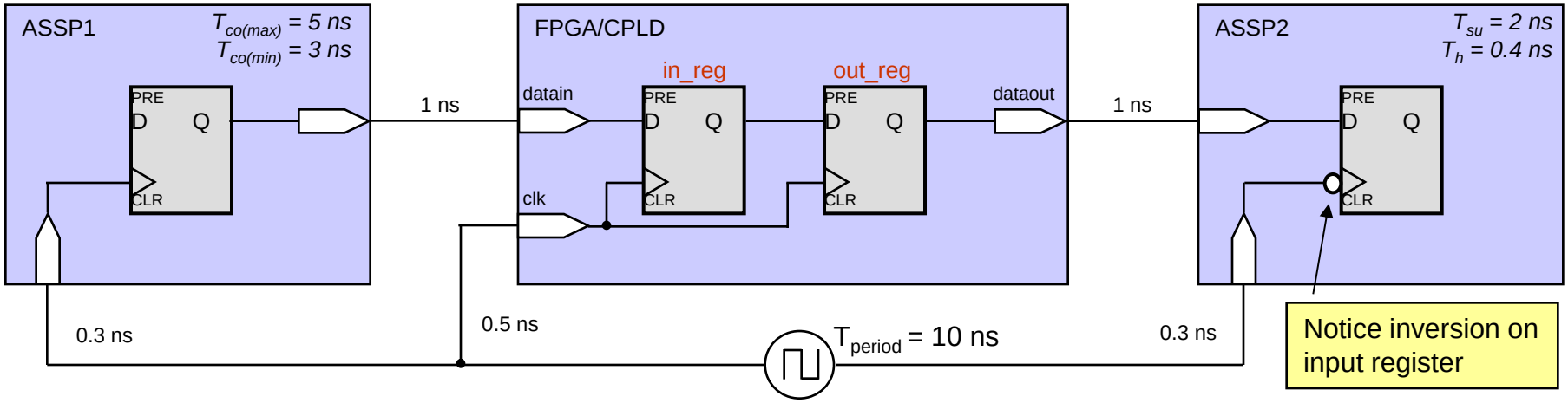
- Constrains output pins by specifying external device timing parameters
- Options

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

Input/Output Delays (GUI)

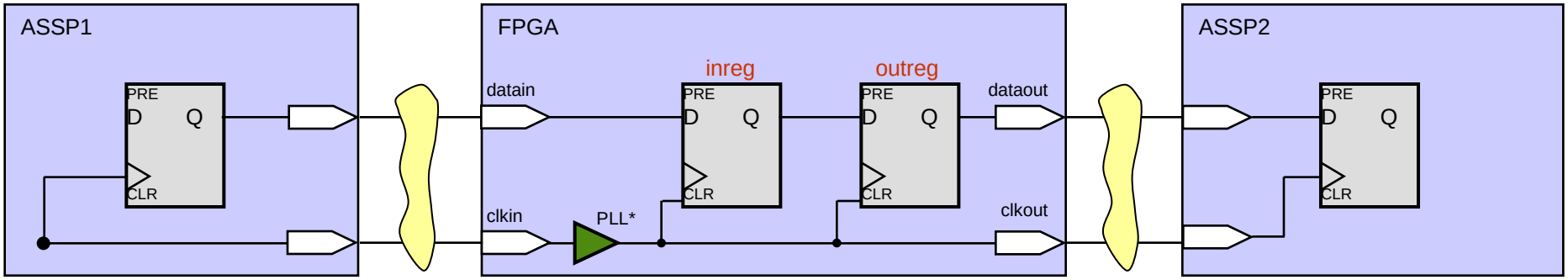


Synchronous I/O Example




```
set_input_delay -clock clk -max [expr -0.5 + 0.3 + 5 + 1] datain
set_input_delay -clock clk -min [expr -0.5 + 0.3 + 3 + 1] datain

set_output_delay -clock clk -max [expr -0.3 + 0.5 + 1 + 2 ] -clock_fall dataout
set_output_delay -clock clk -min [expr -0.3 + 0.5 + 1 -0.4] -clock_fall dataout
```

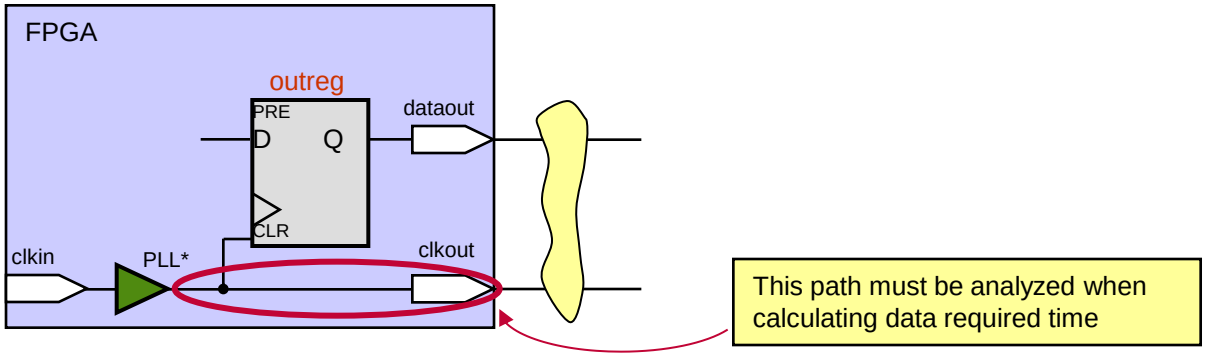
Source-Synchronous Interfaces



- 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
- Data & clock routed identically to maintain phase relationship at destination device
  - Board delay not included in external delay calculations
    - Clock trace delay (data required time) & Data trace delay (data arrival time) are equal and offset
  - Enables higher interface speeds (compared to using system clock)

**PLL, represented by 1 symbol, could be multiple PLLs or generating multiple clock outputs*

Using SDC with Source-Synch



- Must tell TimeQuest to analyze path from clock source to output clock pin during analysis
- Two Methods using `set_output_delay` command
  - Method 1
    - Add `-reference_pin` argument to output delay assignment
      `-reference_pin` is a TimeQuest SDC extension
  - Method 2
    1. Assign generated clock on output clock pin
    2. Use `-clock` argument in output delay assignment to associate output clock to output data bus

Method 1 Example

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

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

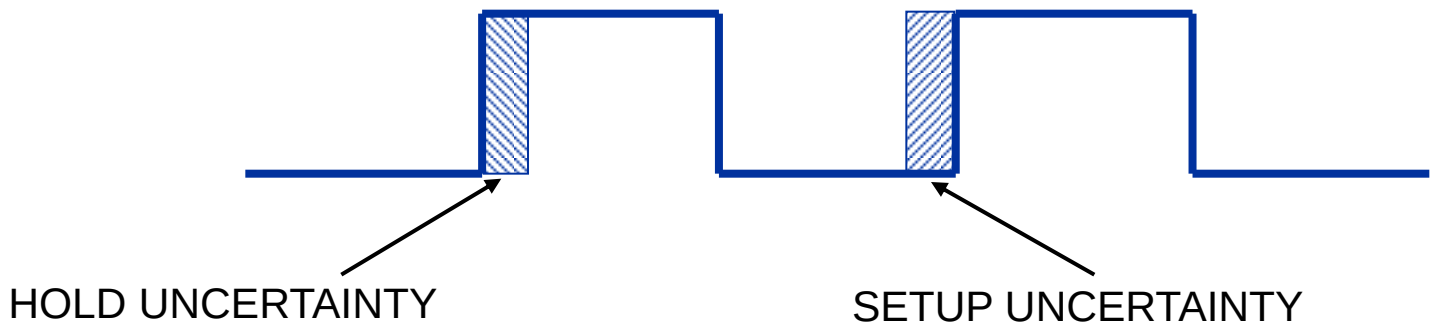
Method 2 Example

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

# Place clock on external clock output
create_generated_clock -name clkout \
       -source [get_pins inst|altpll_component|pll|clk[0]] \
       [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 clkout \
       -max 0.500 \
       [get_ports dataout]
set_output_delay -clock clkout \
       -min -0.500 \
       [get_ports dataout]
```

Virtual Clocks

Clock Uncertainty

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

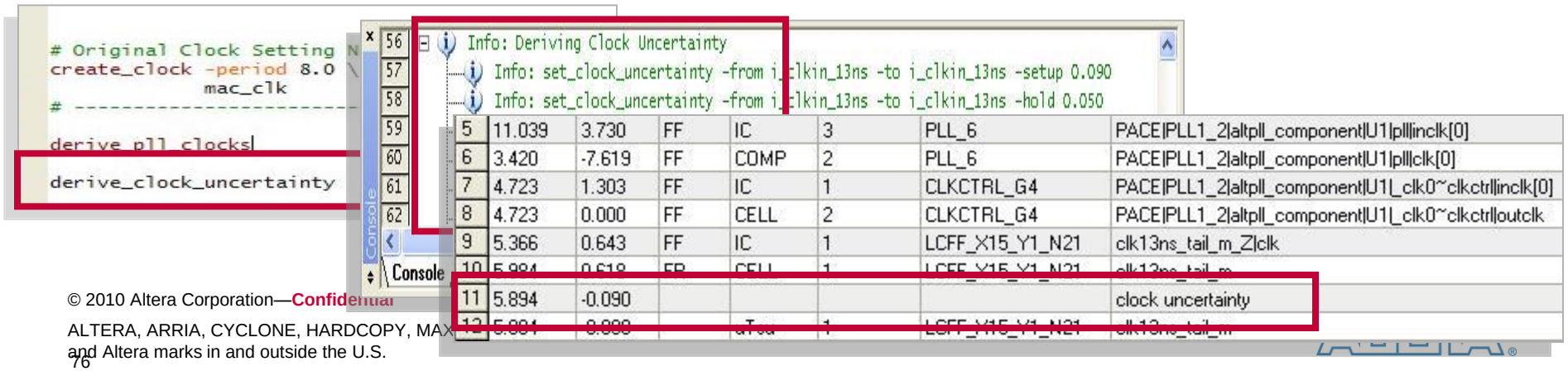


Clock Uncertainty

- 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>`

SDC Enhancement - *derive_clock_uncertainty*

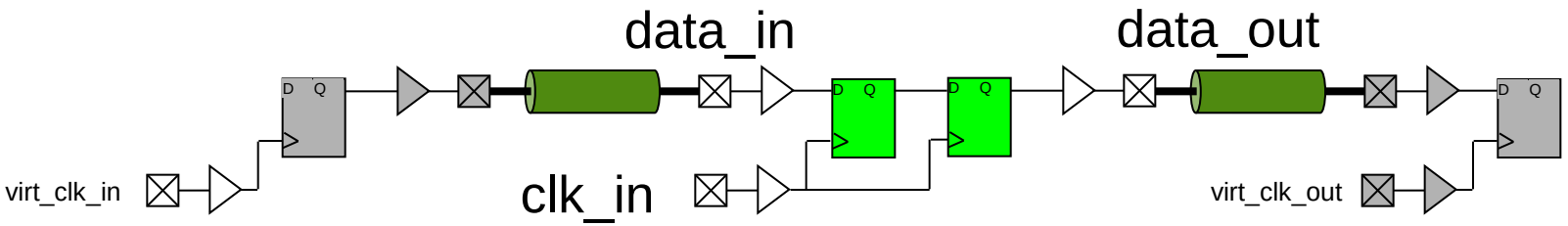
- New SDC constraint - *derive_clock_uncertainty*
  - Computes all the required clock uncertainty for every clock transfer (e.g. PLL settings)
- Advantage in using *derive_clock_uncertainty*
  - Automatically applies SDC constraint, *set_clock_uncertainty*
  - Provides flexibility for accounting system considerations
    - Optionally, modify generated uncertainty values to account for external factors (e.g. board skew, external PLL jitter, etc.)
    - Optionally, disregard if accounted for in timing constraints and/or in controlled environment



Virtual Clocks

- Virtual clocks could be defined for all input clocks and output clocks in current design
  - Virtual input clock for external launch clock in the constraint for the input case
  - Virtual output clock for external latch clock in the constraint for the output case.
- This will separate the clock domains for the IO and core.
- This is required to set the correct clock uncertainty numbers in the clock uncertainty constraints, which differ for the core and the IO transfers
- Not used for output source synchronous interface (output generated clock)

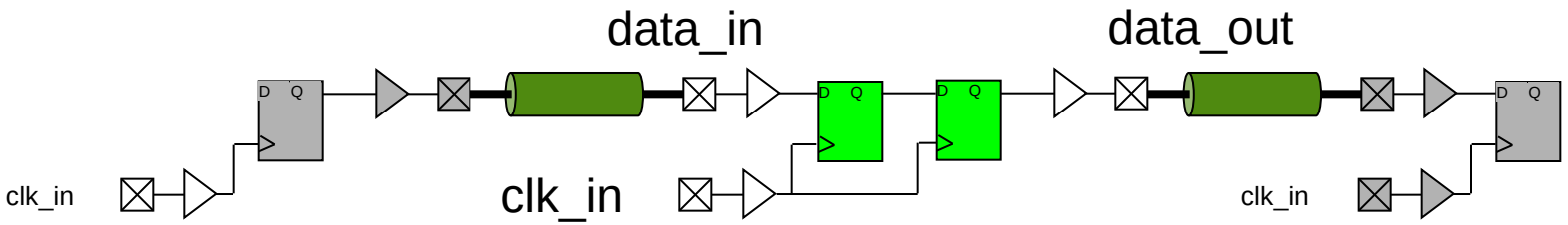
Virtual Clocks



- Traditionally, you'd set input delay and output delay with clk_in as the reference clock
  - This intra-clock transfer would appear the same as the IO clock transfer
  - Incorrect clock uncertainty would be applied for IO transfers
- Virtual clocks for input transfer and output transfer provide separate and distinct clock transfers from clk_in core intra-clock transfer.

Virtual Clocks

Case A: BAD!



- **Old Constraints:**

  create_clock -period 10 -name clk_in [get_ports {clk_in}]
  set_input_delay -clock [get_clocks {clk_in}] -max 2 [get_ports {data_in}]
  set_input_delay -clock [get_clocks {clk_in}] -min 0 [get_ports {data_in}]
  set_output_delay -clock [get_clocks {clk_in}] -max 2 [get_ports {data_out}]
  set_output_delay -clock [get_clocks {clk_in}] -min 0 [get_ports {data_out}]

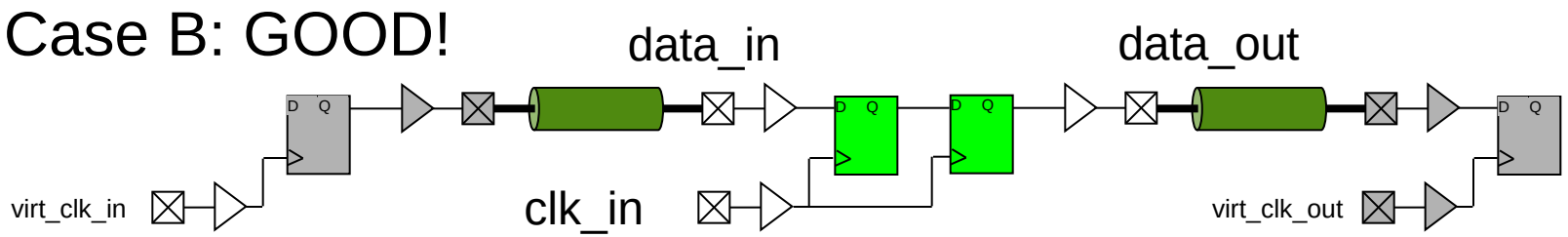
- **Clock Transfers**

  From clk_in to clk_in only

- **Resultant Clock Uncertainty**

  From clk_in to clk_in **Setup: 150ps Hold: 50ps**

Virtual Clocks



- **New Constraints:**
  - create_clock -period 10 -name clk_in [get_ports {clk_in}]
  - **create_clock -period 10 -name virt_clk_in**
  - **create_clock -period 10 -name virt_clk_out**
  - set_input_delay -clock **[get_clocks {virt_clk_in}]** -max 2 [get_ports {data_in}]
  - set_input_delay -clock **[get_clocks {virt_clk_in}]** -min 0 [get_ports {data_in}]
  - set_output_delay -clock **[get_clocks {virt_clk_out}]** -max 2 [get_ports {data_out}]
  - set_output_delay -clock **[get_clocks {virt_clk_out}]** -min 0 [get_ports {data_out}]
- **Clock Transfers**
  - From clk_in to clk_in
  - From virt_clk_in to clk_in
  - From clk_in to virt_clk_out
- **Resultant Clock Uncertainty**

From clk_in to clk_in	**Setup: 150ps Hold: 50ps**
From virt_clk_in to clk_in	**Setup: 130ps Hold: 130ps**
From clk_in to virt_clk_out	**Setup: 130ps Hold: 130ps**

SDC Timing Constraints

- Clocks
- I/O
- False paths 
- Multicycle paths
- Absolute delays

False Paths

- Logic-based
  - Paths not relevant during for circuit operations
  - 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

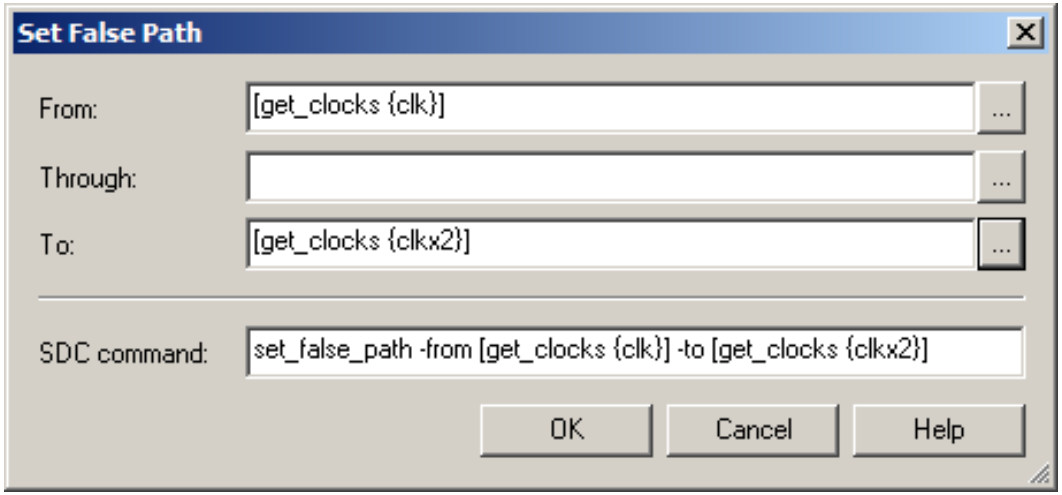
`set_false_path` Command

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

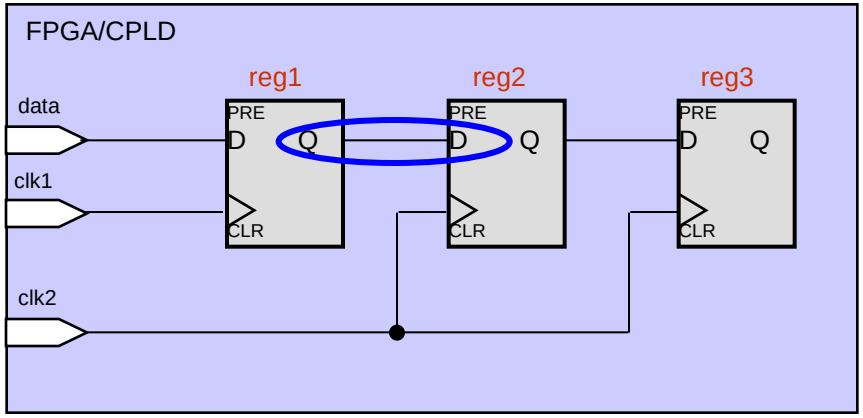
```
[-fall_from <names>]
[-rise_from <names>]
[-from <names>]
[-through <names>]
[-to <names>]
[-fall_to <names>]
[-rise_to <names>]
[-setup]
[-hold]
<targets>
```

Set False Path (GUI)

Constraints menu



False Path Example



Simple synchronizer circuit between two asynchronous clock domains

```
Set_false_path -from [get_pins reg1|regout] \
               -to [get_pins reg2|datain]
```

`set_clock_groups` Command

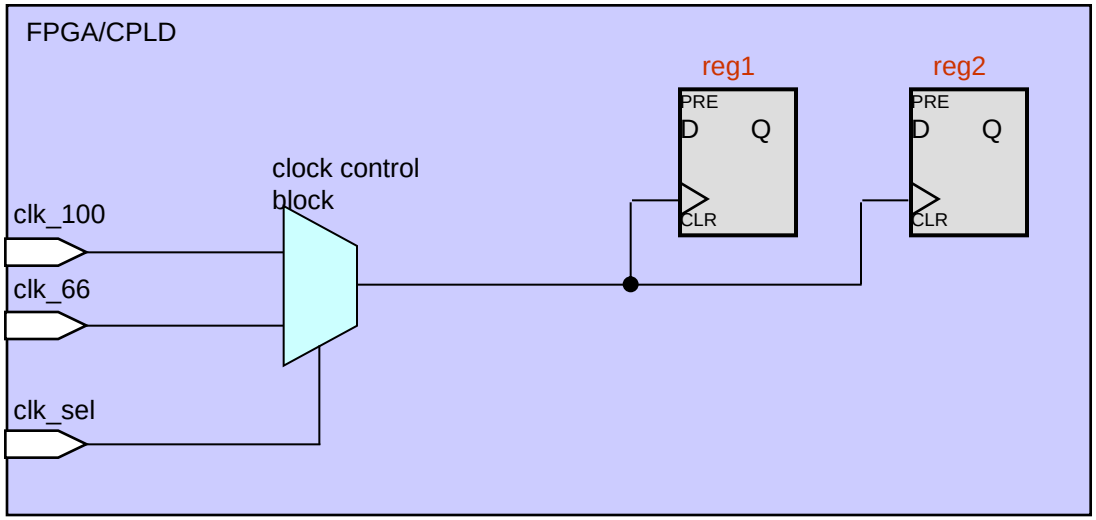
- Tells fitter and TimeQuest to ignore **ALL** paths between specified clock domains
  - Great for clock muxes
  - Equivalent to setting false paths (`-from` & `-to`) on all paths between domains
- Options

```
[-asynchronous | -exclusive]
-group <clock name>
-group <clock_name>
[-group <clock name>]…
```

Asynchronous clock domains example

```
set_clock_groups -asynchronous                          \
                                                        \
  -group { clkin1                                       \
           inst1|altpll_component|pll|clk[0]            \
           inst1|altpll_component|pll|clk[1] }          \
                                                        \
  -group { clkin2                                       \
           inst2|altpll_component|pll|clk[0]            \
           inst2|altpll_component|pll|clk[1]            \
           inst2|altpll_component|pll|clk[2] }          \
                                                        \
  -group {PCI_CLOCK }
```

Clock Mux Example




```
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, TimeQuest should not analyze
#   cross-domain paths as only one clock will be driving the
#   registers at any one time.
```

SDC Timing Constraints

- Clocks
- I/O
- False paths
- Multicycle paths 
- Absolute delays

When to Use Multicycle Paths

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

Multicycle Types (1)

- Destination
  - Constraint based on destination clock edges
  - Moves latch edge backward to relax required setup/hold time
  - Used in most multicycle situations
- Source
  - Constraint based on source clock edges
  - Moves launch edge forward to relax required setup/hold time
  - Useful when source clock is at higher frequency than destination

Multicycle Types (2)

- Setup
  - Increases the number of cycles for setup analysis
  - Default is 1
- Hold
  - Increases the number of cycles for hold analysis
  - Default is 0*

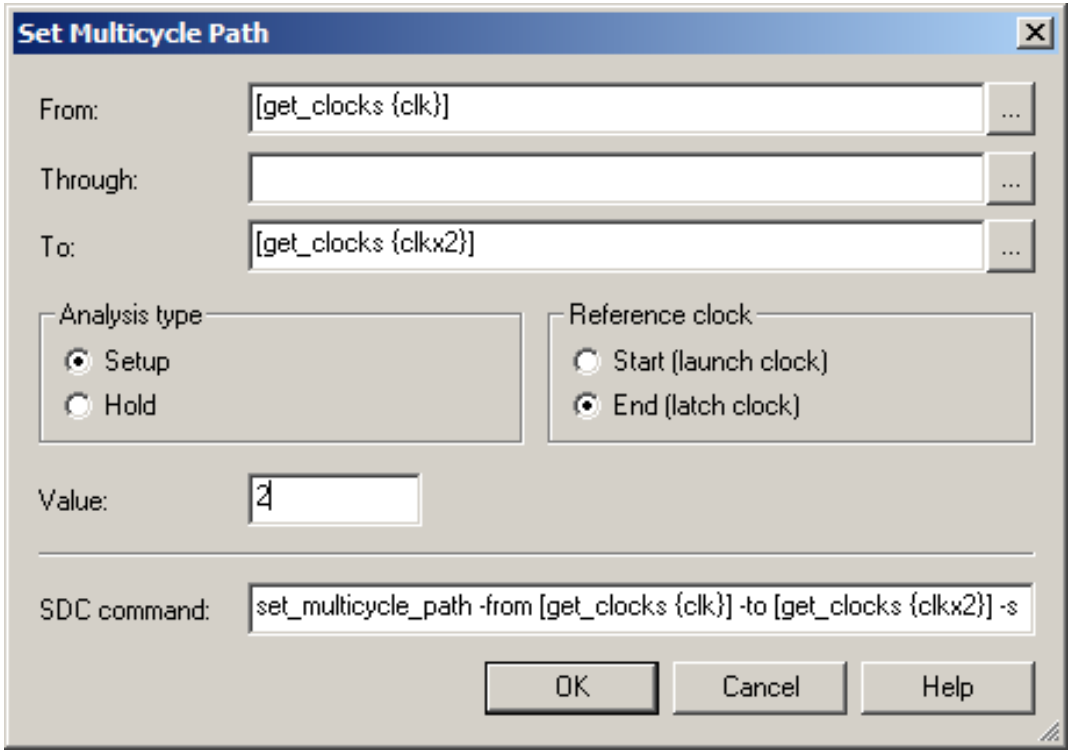
**Note: Subtract 1 from the Classic Timing Analyzer hold multicycle value to convert to SDC*

set_multicycle_path Command

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

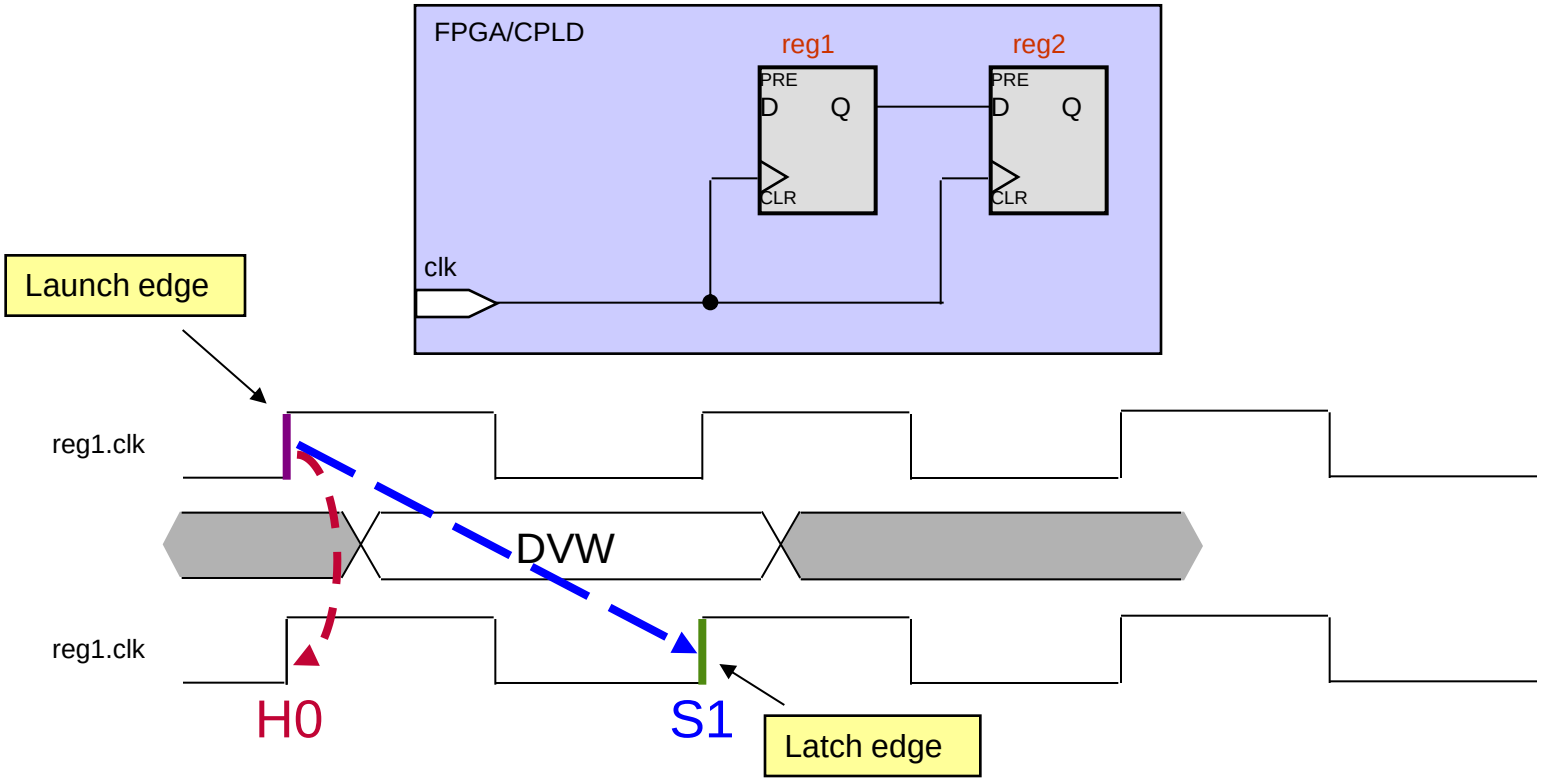
```
[-start | -end]
[-setup | -hold]
[-fall_from <names>]
[-rise_from <names>]
[-from <names>]
[-through <names>]
[-to <names>]
[-fall_to <names>]
[-rise_to <names>]
<targets>
<value>
```

Set Multicycle Path (GUI)



Understanding Multicycle (1)

Standard single-cycle register transfer



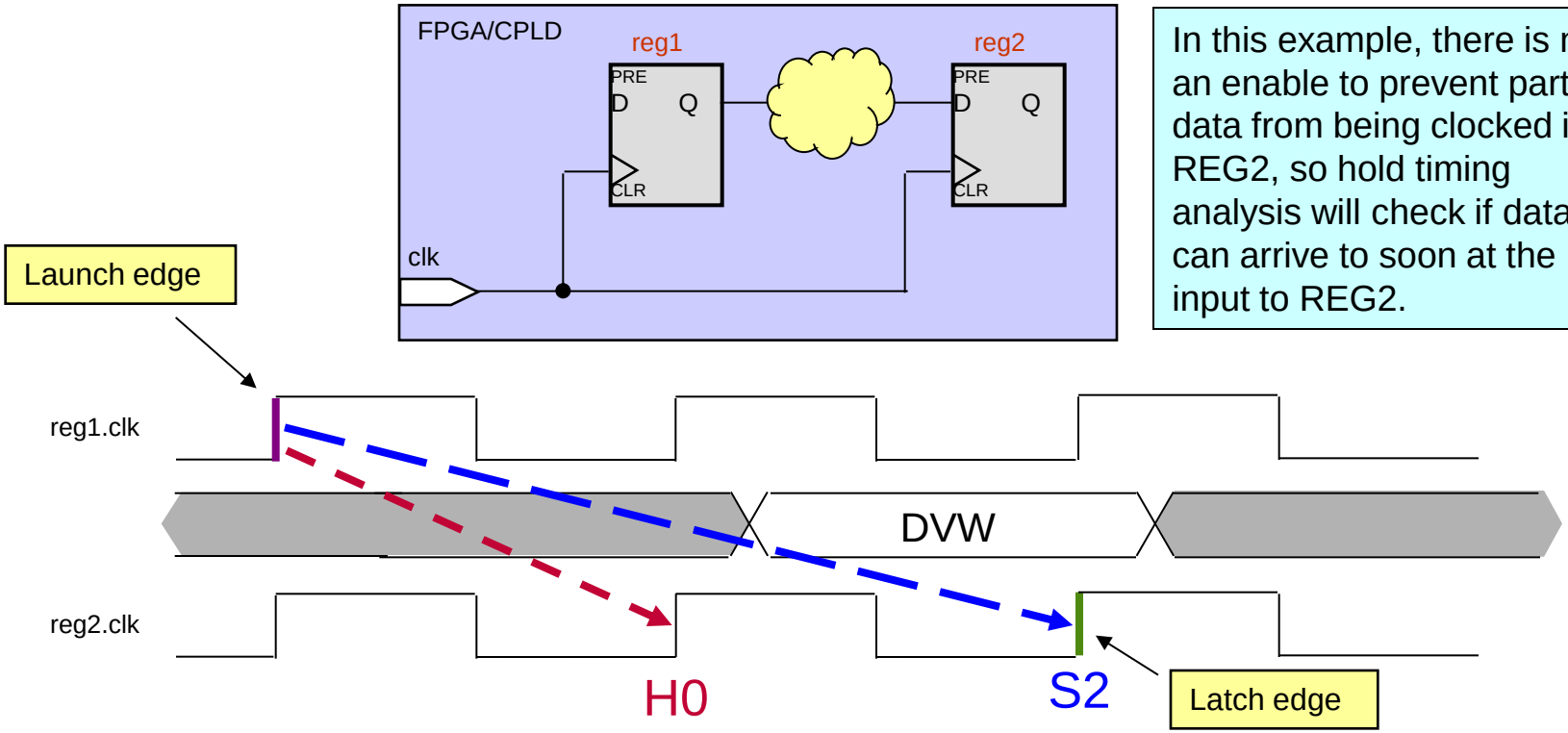
Multicycle Setup = 1 (Default)

Multicycle Hold = 0 (Default)*

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

Understanding Multicycle (2)

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



In this example, there is not an enable to prevent partial data from being clocked into REG2, so hold timing analysis will check if data can arrive to soon at the input to REG2.

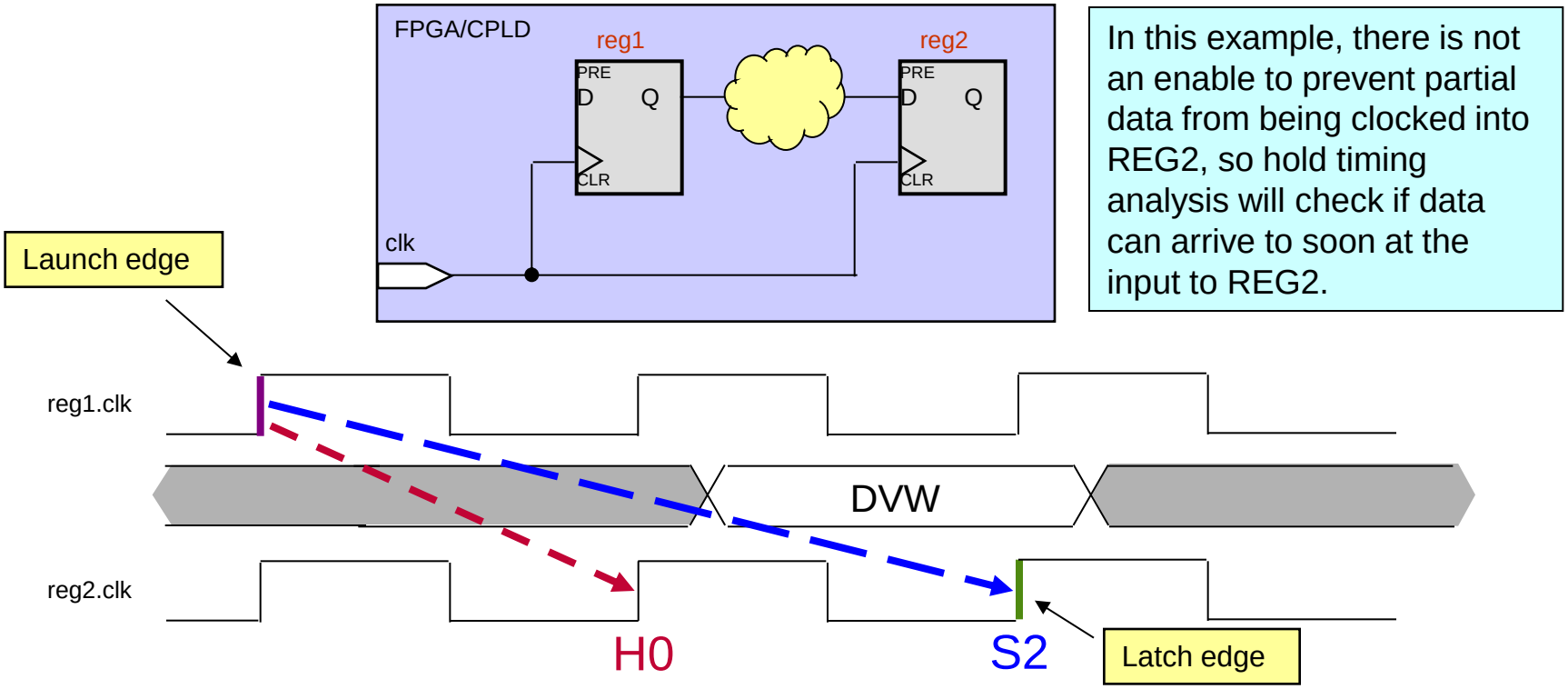
Multicycle Setup = 2

Multicycle Hold = 0 (Default)

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

Understanding Multicycle (2) (cont.)

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



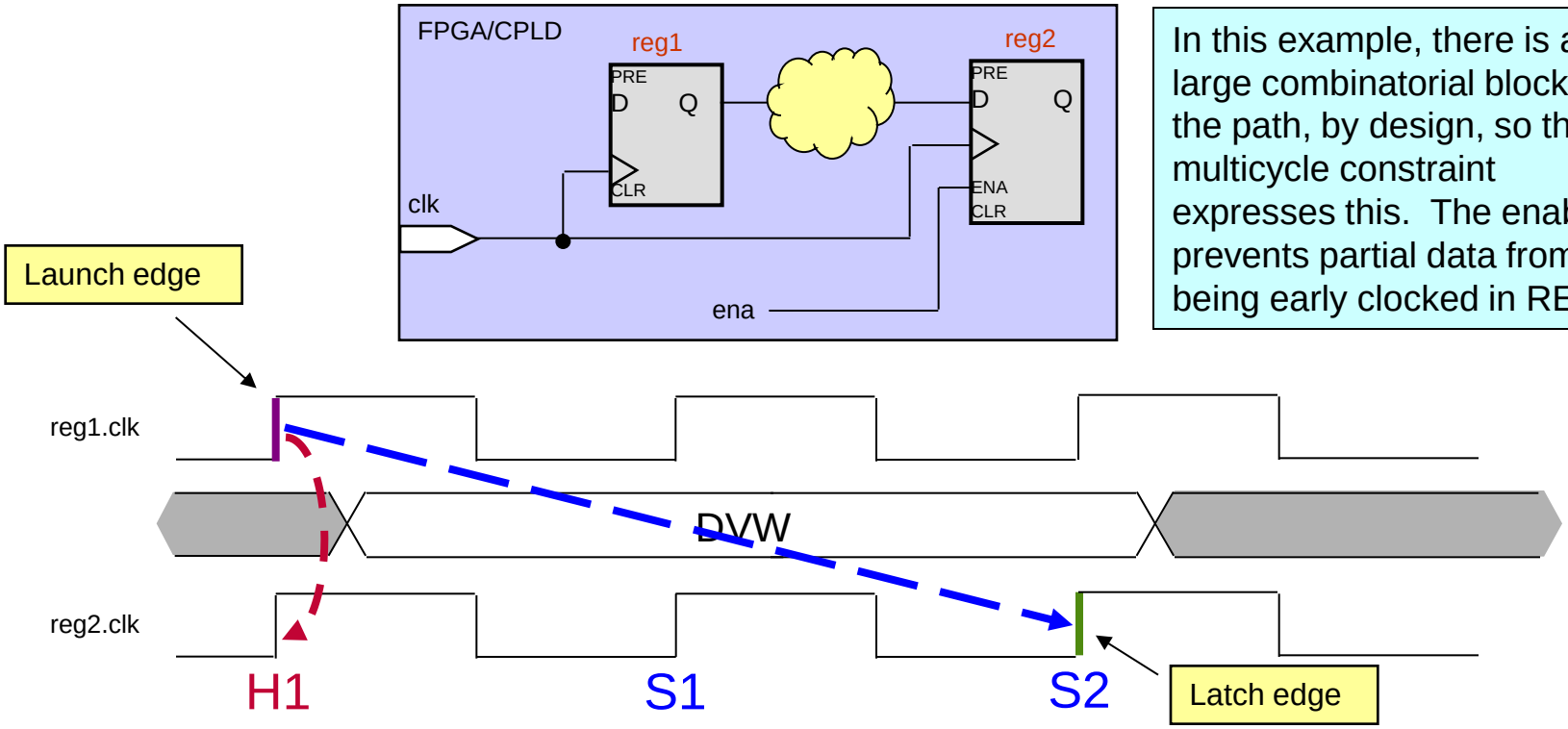
In this example, there is not an enable to prevent partial data from being clocked into REG2, so hold timing analysis will check if data can arrive to soon at the input to REG2.

```
Set_multicycle_path -from [get_pins reg1|regout] -to [get_pins reg2|datain] \
        -end -setup 2
```

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

Understanding Multicycle (3)

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



In this example, there is a large combinatorial block in the path, by design, so the multicycle constraint expresses this. The enable prevents partial data from being early clocked in REG2.

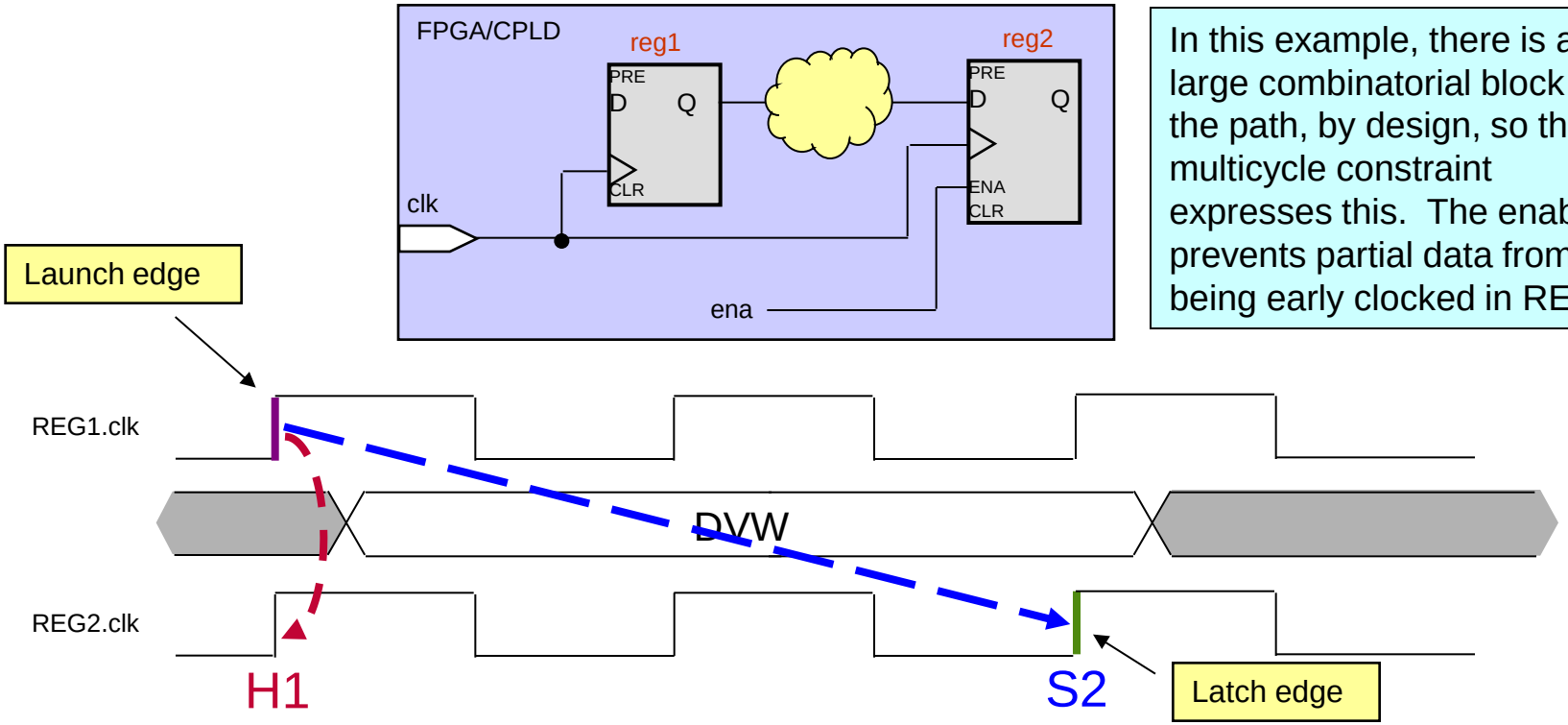
Multicycle Setup = 2

Multicycle Hold = 1

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

Understanding Multicycle (3) (cont.)

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

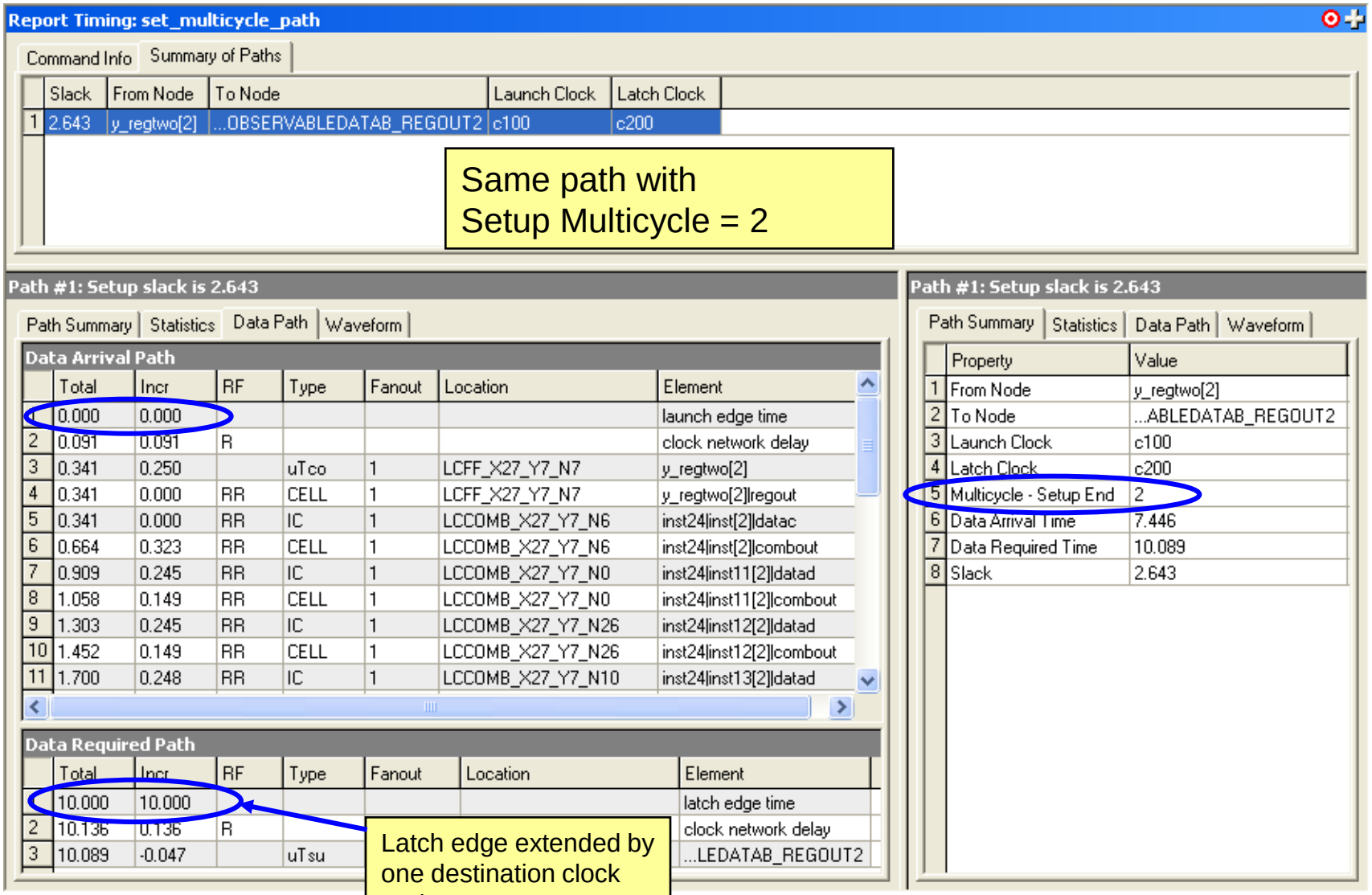


In this example, there is a large combinatorial block in the path, by design, so the multicycle constraint expresses this. The enable prevents partial data from being early clocked in REG2.

```
Set_multicycle_path -from [get_pins reg1|regout] -to [get_pins reg2|datain] \
        -setup 2
Set_multicycle_path -from [get_pins reg1|regout] -to [get_pins reg2|datain] \
        -hold 1
```

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

Reporting Multicycles



SDC Timing Constraints

- Clocks
- I/O
- False paths
- Multicycle paths
- Absolute delays 

Absolute Delays

- Applys 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 a input port-to-register or register-to-output port constraint without using input & output delays

- Use `-rise_from`/`-fall_from` & `-rise_to`/`-fall_to` to restict 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
```

Timing Analysis Agenda

- TimeQuest flow
- Timing constraints
- Timing reports 

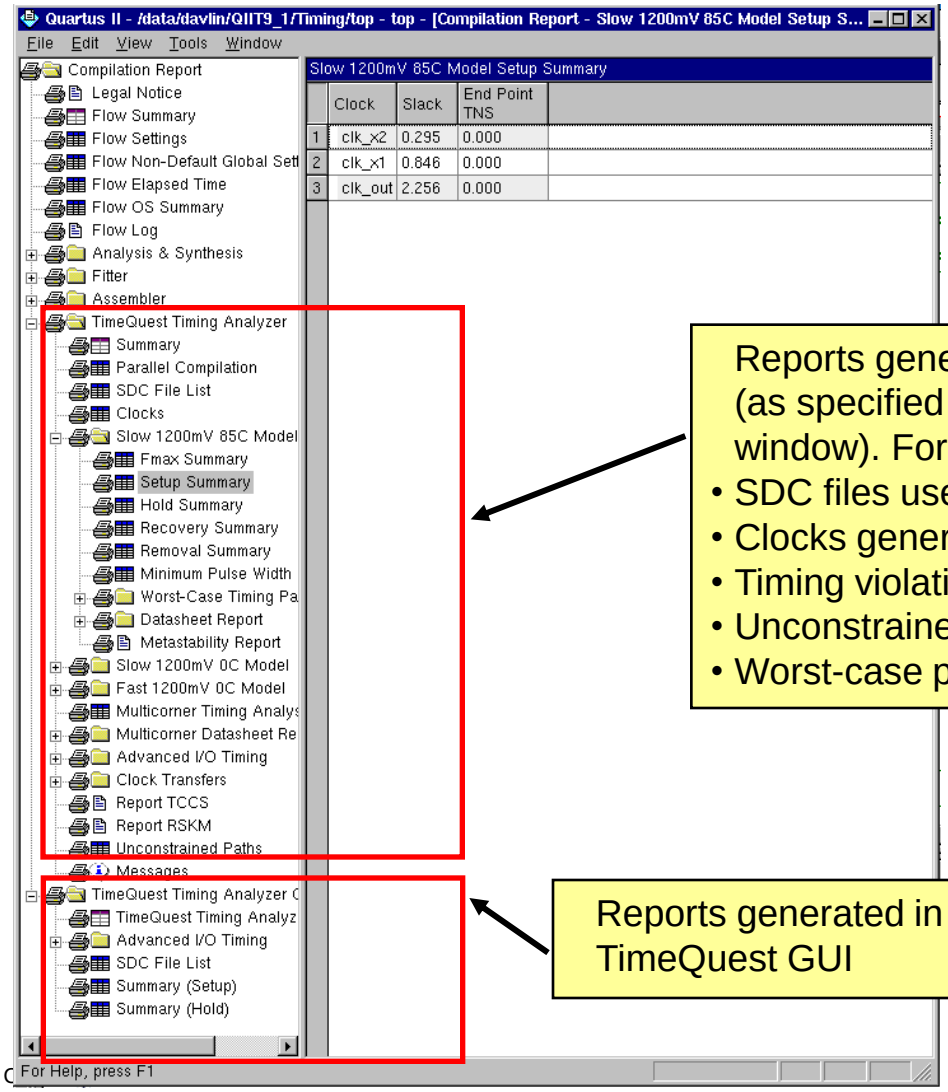
Verifying Clocks & I/O Timing

- Use Clock Setup & Hold summary reports to check worst slack for each clock

  "Did I make it or did I not make it?"

  - Positive slack displayed in **black**, negative in **red**
  - 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

TimeQuest Reports in Compilation Report



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

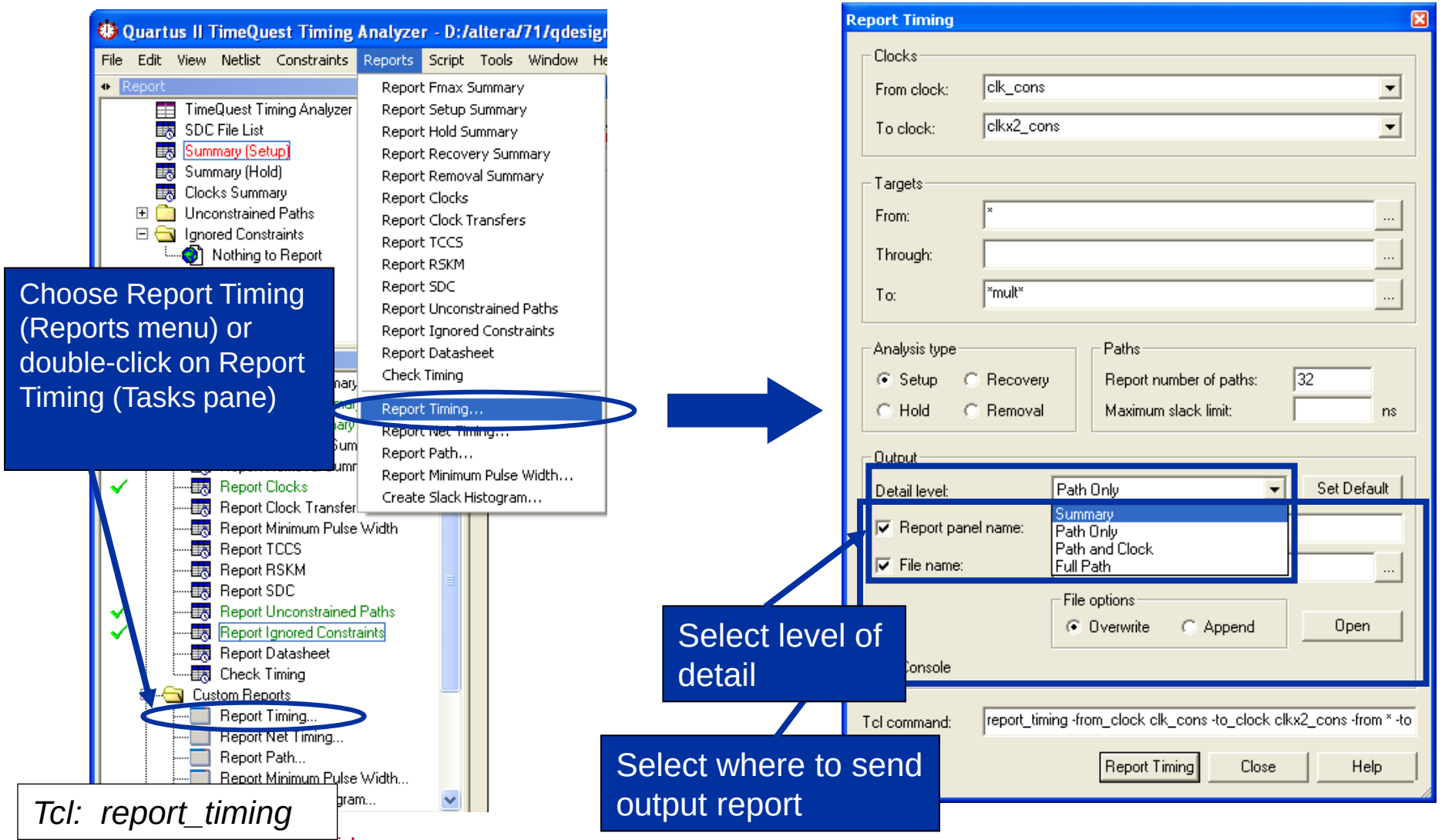
- SDC files used during fitting
- Clocks generated
- Timing violations
- Unconstrained paths
- Worst-case paths

Reports generated in TimeQuest GUI

Detailed Slack/Path Analysis

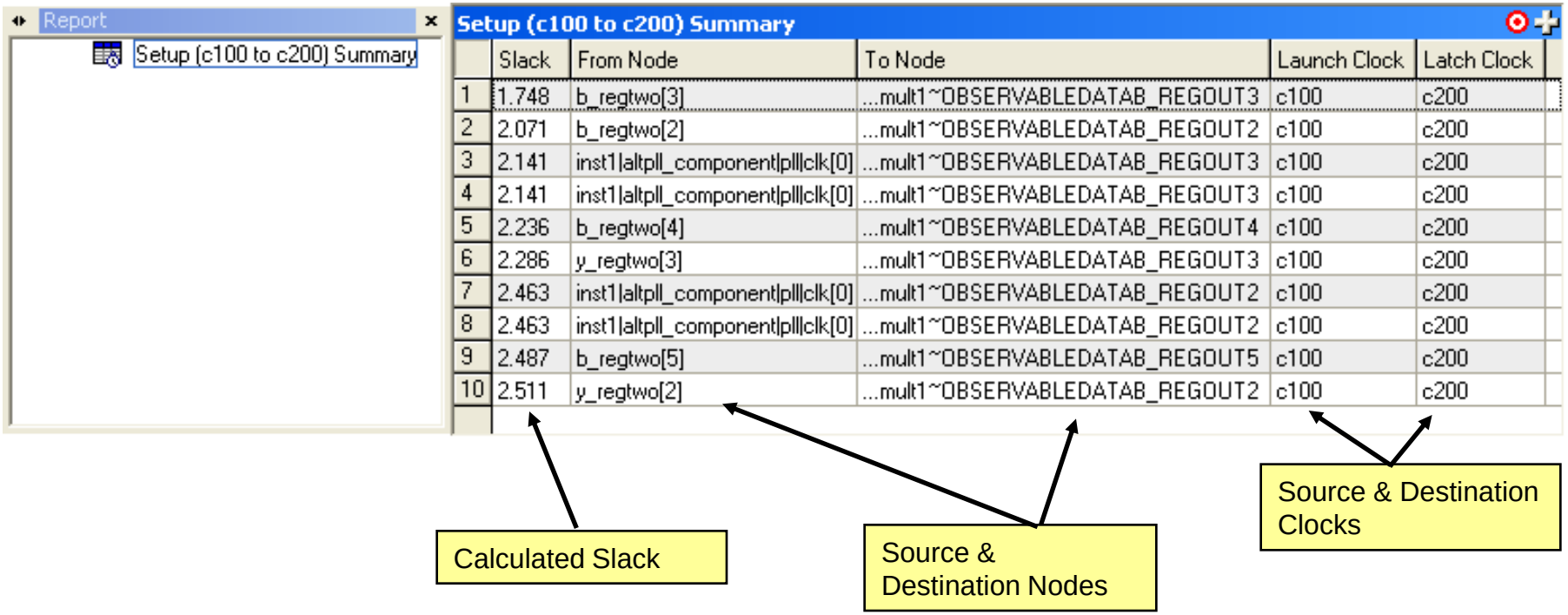
- Create more specific/detailed reports
  - Ex. Details on a specific clock domain
  - Ex. View timing paths between particular I/O & registers
- Create using Tcl commands or GUI
  - Use GUI to see report immediately
  - Use Tcl file for repeatability
    - Recommendation: Do not place reporting commands in SDC file

Report Timing (GUI)



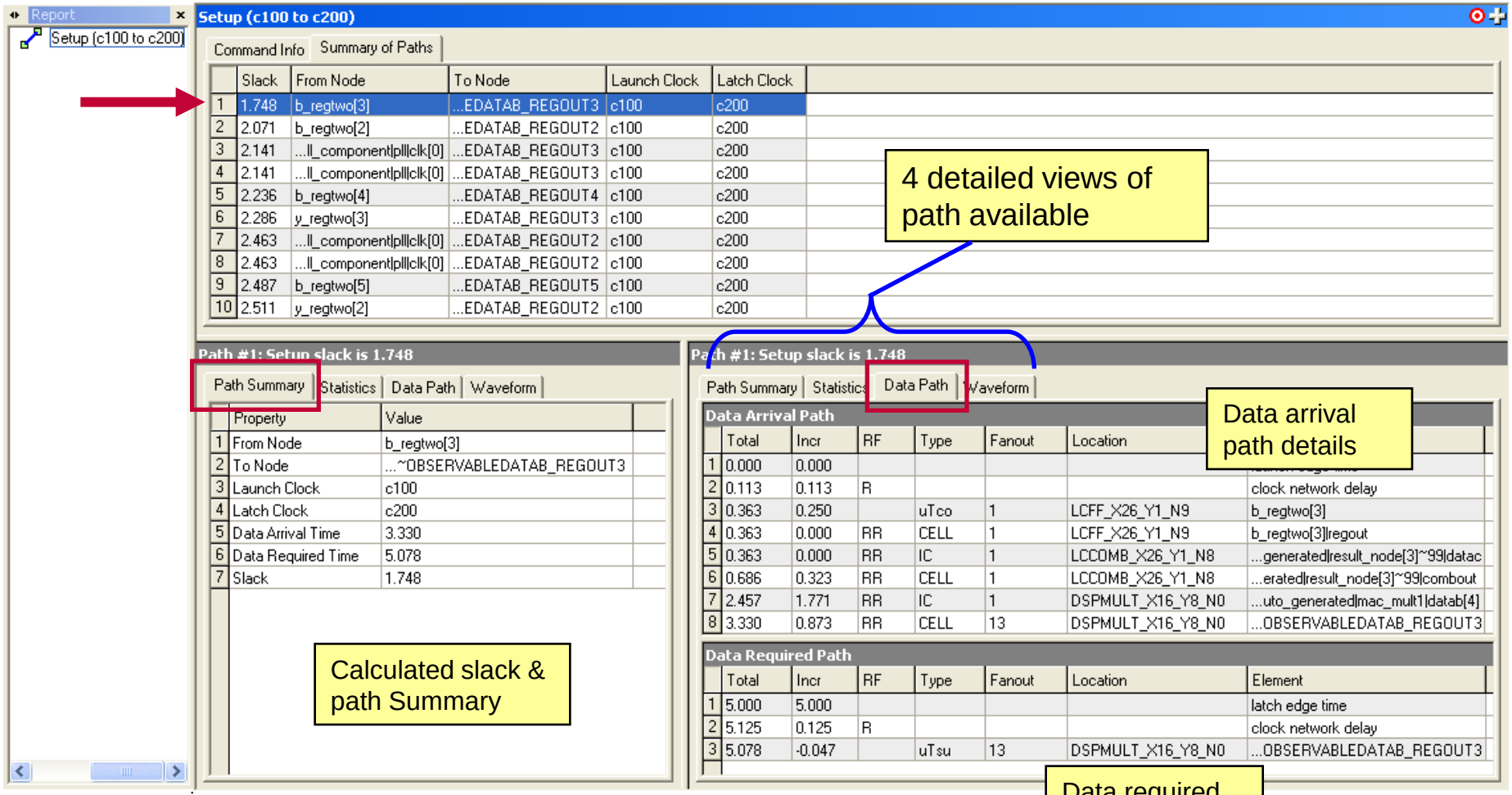
Summary Slack/Path Report

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



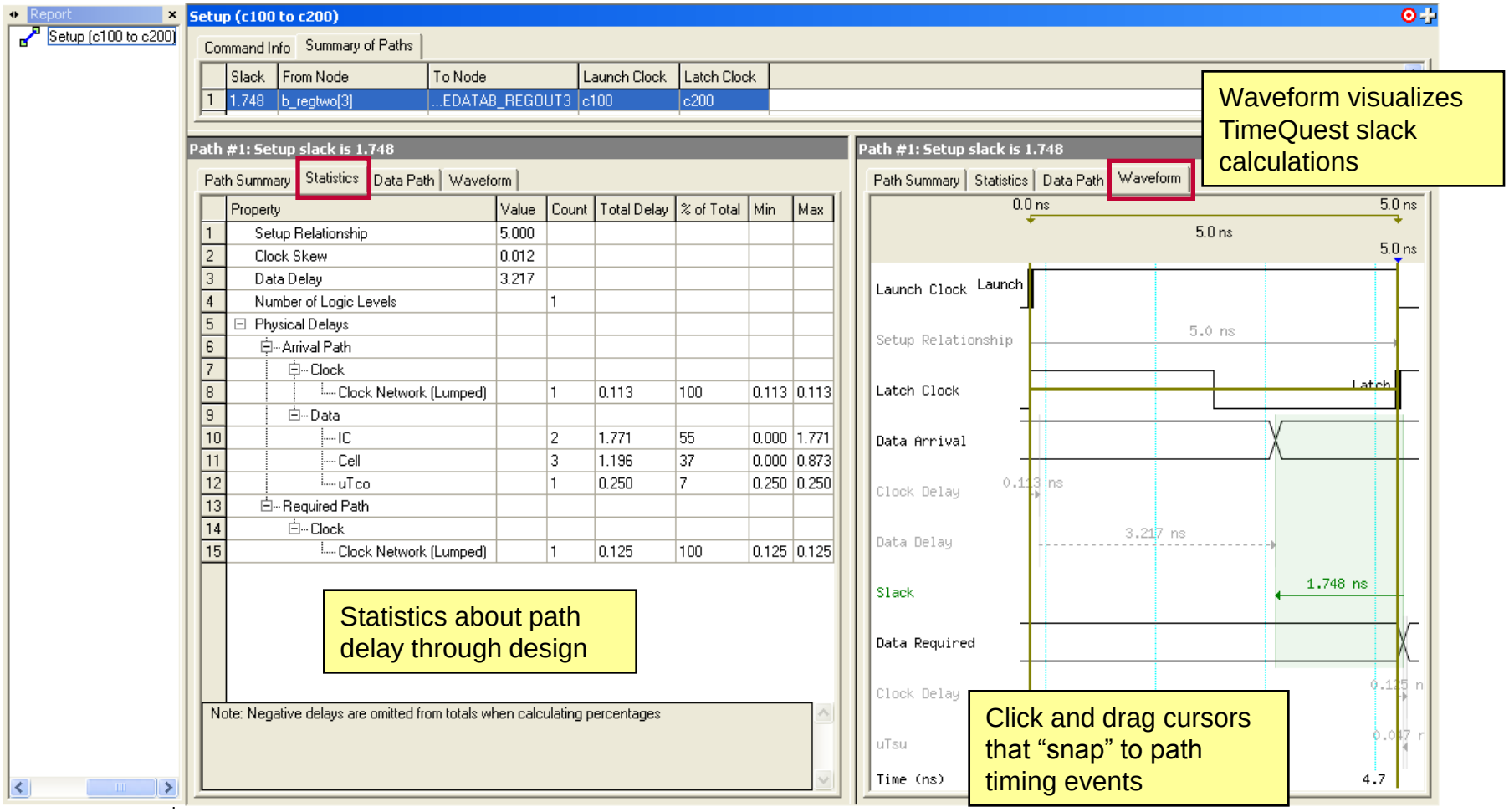
Detailed Slack/Path Report

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



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)"
```

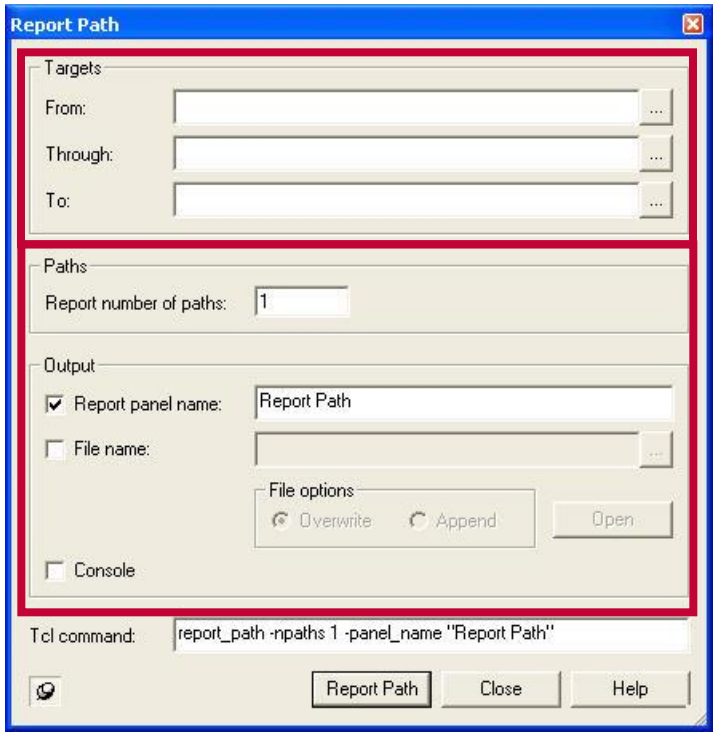


Report Path

- Report arbitrary point-to-point path (does not need to be constrained)

Nodes

Specifies what you are look for

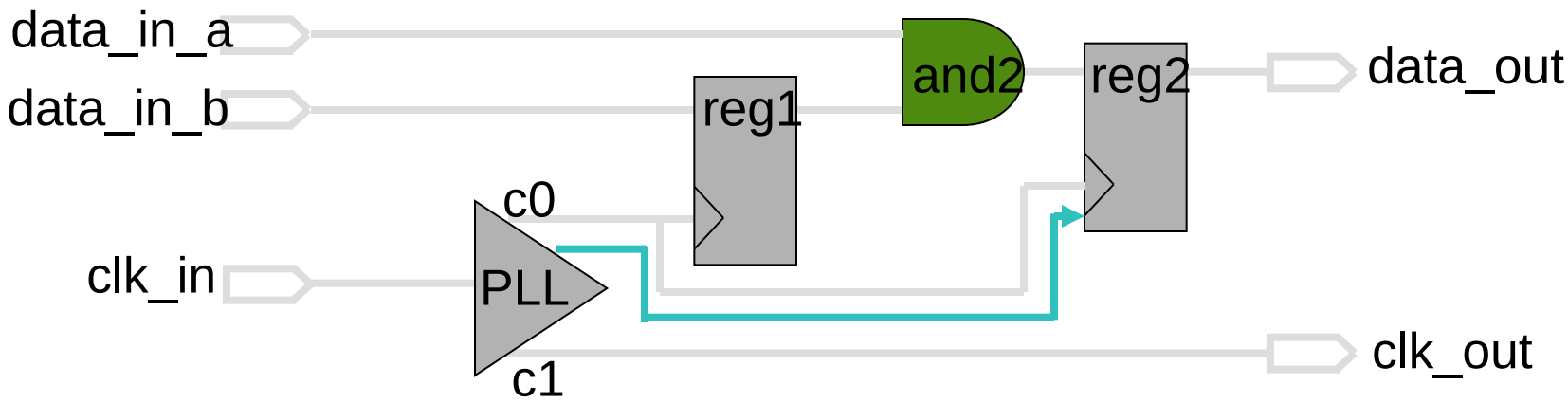


Output options

Specifies output customizations

- Does not go flow through keepers

Report Path

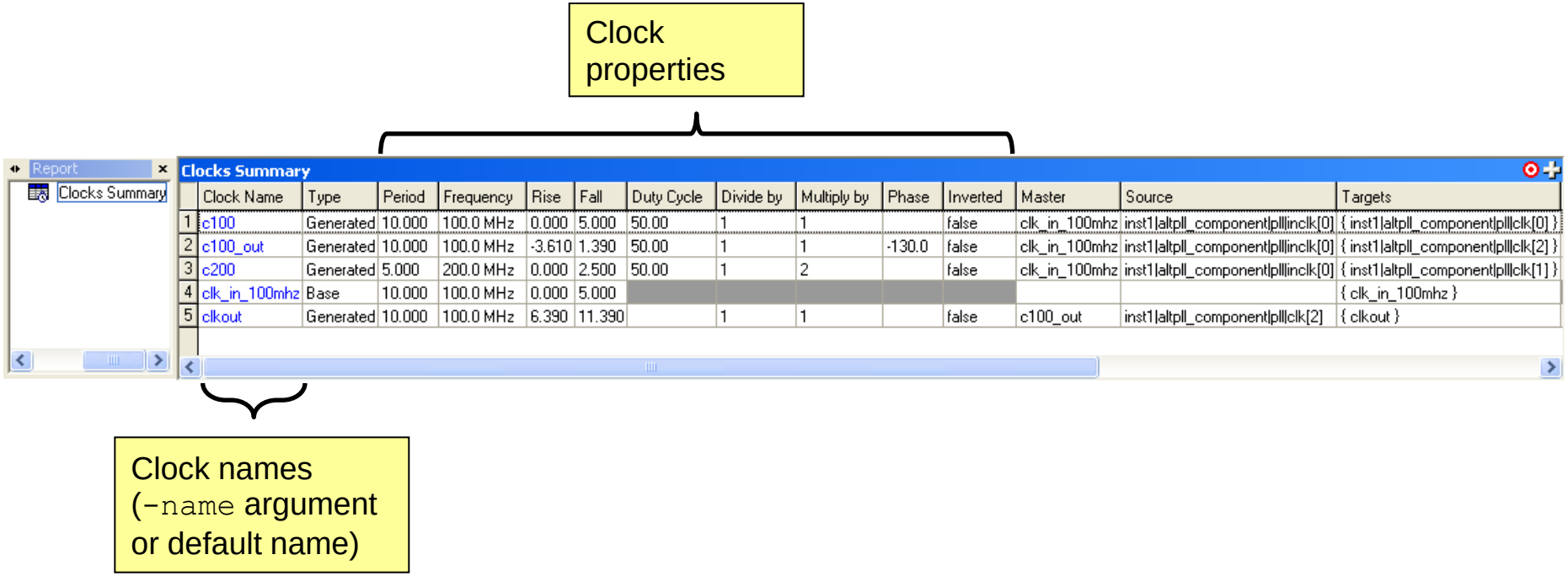



```
>report_path -from [get_pins {inst1|altpll_component|pll|clk[0]}] -to [get_pins \
{reg2|clk}] -npaths 1
```

```
Info: Path #1: Delay is 1.974
    Info: ======================================================================
    Info: From Node    : inst1|altpll_component|pll|clk[0]
    Info: To Node      : reg2|clk
    Info:
    Info: Path:
    Info:
    Info: Total (ns)       Incr (ns)          Type    Node
    Info: ==========  ========= ==  ====  ==================================
    Info:      0.000       0.000                    inst1|altpll_component|pll|clk[0]
    Info:      1.303       1.303        RR  IC      inst1|altpll_component|_clk0~clkctrl|inclk[0]
    Info:      1.303       0.000        RR  CELL    inst1|altpll_component|_clk0~clkctrl|outclk
    Info:      1.974       0.671        RR  IC      reg2|clk
    Info:
    Info: Total Path Delay :     1.974
    Info: ======================================================================
```

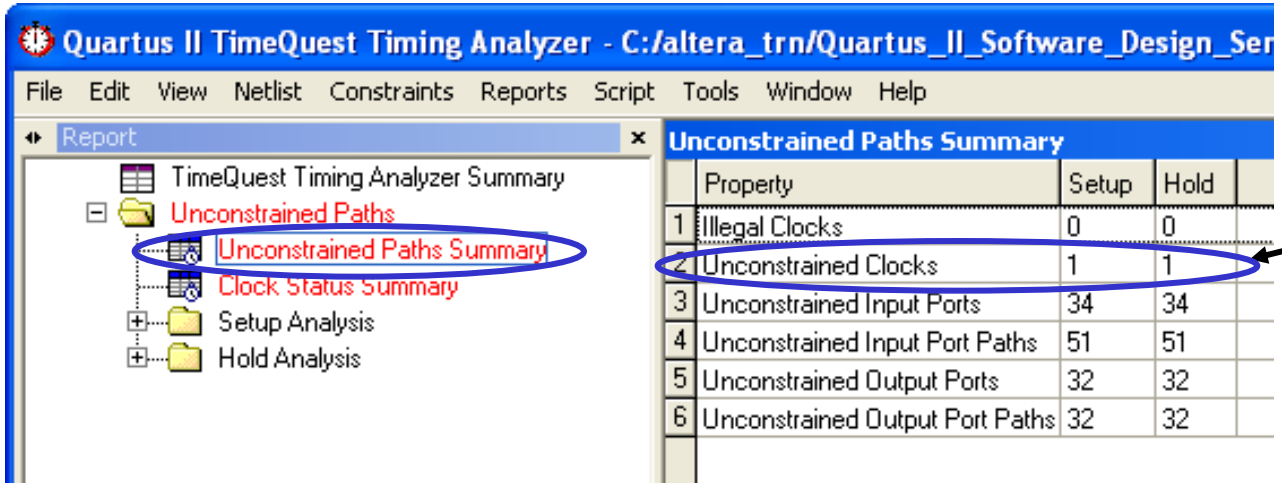
Report Clocks (`report_clocks`)

- List details about the properties of constrained clocks



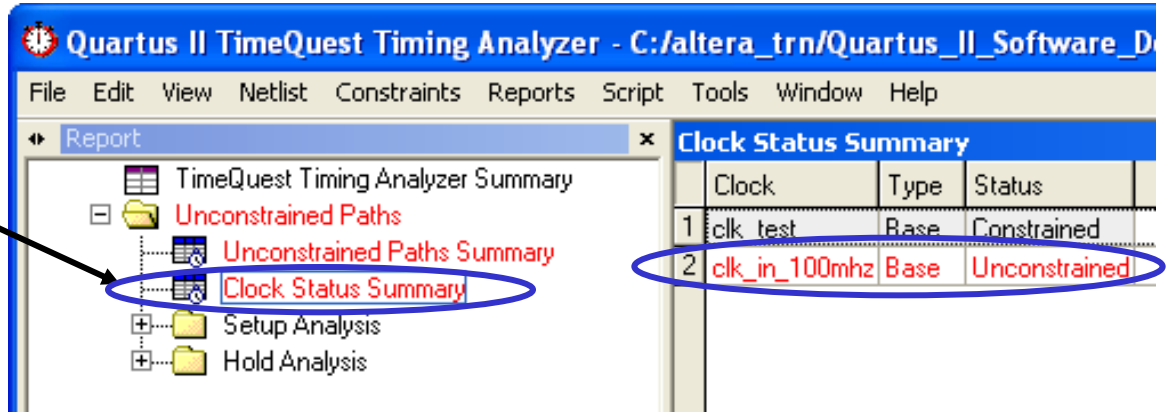
	Clock Name	Type	Period	Frequency	Rise	Fall	Duty Cycle	Divide by	Multiply by	Phase	Inverted	Master	Source	Targets
1	c100	Generated	10.000	100.0 MHz	0.000	5.000	50.00	1	1		false	clk_in_100mhz	inst1|altpll_component|pll|inclk[0]	{ inst1|altpll_component|pll|clk[0] }
2	c100_out	Generated	10.000	100.0 MHz	-3.610	1.390	50.00	1	1	-130.0	false	clk_in_100mhz	inst1|altpll_component|pll|inclk[0]	{ inst1|altpll_component|pll|clk[2] }
3	c200	Generated	5.000	200.0 MHz	0.000	2.500	50.00	1	2		false	clk_in_100mhz	inst1|altpll_component|pll|inclk[0]	{ inst1|altpll_component|pll|clk[1] }
4	clk_in_100mhz	Base	10.000	100.0 MHz	0.000	5.000								{ clk_in_100mhz }
5	clkout	Generated	10.000	100.0 MHz	6.390	11.390		1	1		false	c100_out	inst1|altpll_component|pll|clk[2]	{ clkout }

Unconstrained Path Report



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

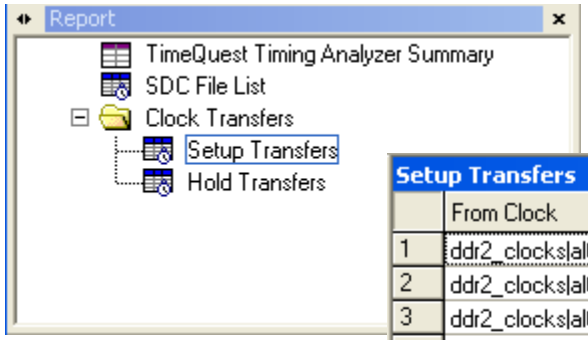
Clock Status Summary Report lists each clock found and whether it was constrained



Verifying False Paths & Groups

- False paths
  - Check ignored constraints
  - Perform report timing on specified paths to ensure no results are returned
- Clock groups
  - Check clock transfers to ensure no paths are returned

Report Clock Transfers

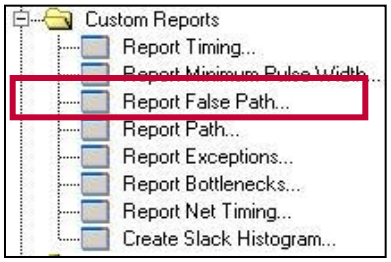


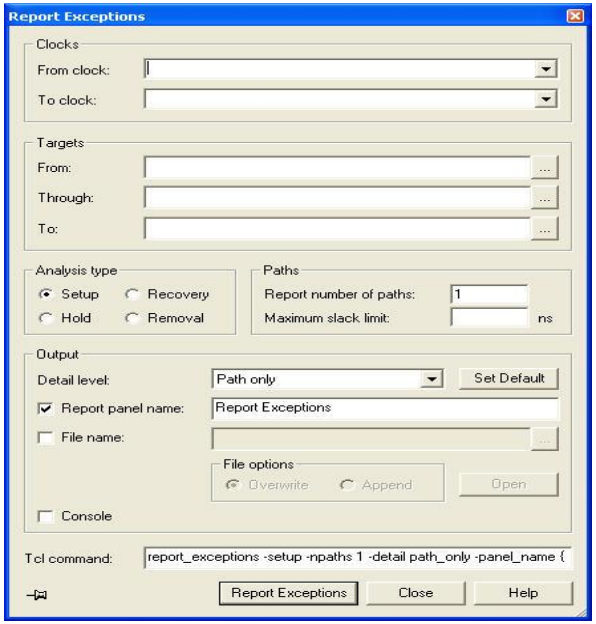
Setup Transfers

	From Clock	To Clock	RR Paths	FR Paths	RF Paths	FF Paths
1	ddr2_clocks|altpll_component|pll|clk[0]	ddr2_clocks|altpll_component|pll|clk[0]	26198	0	19	0
2	ddr2_clocks|altpll_component|pll|clk[3]	ddr2_clocks|altpll_component|pll|clk[0]	64	0	0	0
3	ddr2_clocks|altpll_component|pll|clk[0]	ddr2_clocks|altpll_component|pll|clk[1]	108	0	0	0
4	ddr_ck0_nddr_dqs[0]	ddr2_clocks|altpll_component|pll|clk[3]	0	16	0	0
5	ddr_ck0_nddr_dqs[1]	ddr2_clocks|altpll_component|pll|clk[3]	0	16	0	0
6	ddr_ck0_nddr_dqs[2]	ddr2_clocks|altpll_component|pll|clk[3]	0	16	0	0
7	ddr_ck0_nddr_dqs[3]	ddr2_clocks|altpll_component|pll|clk[3]	0	16	0	0
8	ddr_ck0ddr_dqs[0]	ddr2_clocks|altpll_component|pll|clk[3]	0	16	0	0
9	ddr_ck0ddr_dqs[1]	ddr2_clocks|altpll_component|pll|clk[3]	0	16	0	0
10	ddr_ck0ddr_dqs[2]	ddr2_clocks|altpll_component|pll|clk[3]	0	16	0	0
11	ddr_ck0ddr_dqs[3]	ddr2_clocks|altpll_component|pll|clk[3]	0	16	0	0
12	ddr_ck1_nddr_dqs[0]	ddr2_clocks|altpll_component|pll|clk[3]	0	16	0	0
13	ddr_ck1_nddr_dqs[1]	ddr2_clocks|altpll_component|pll|clk[3]	0	16	0	0
14	ddr_ck1_nddr_dqs[2]	ddr2_clocks|altpll_component|pll|clk[3]	0	16	0	0
15	ddr_ck1_nddr_dqs[3]	ddr2_clocks|altpll_component|pll|clk[3]	0	16	0	0
16	ddr_ck1ddr_dqs[0]	ddr2_clocks|altpll_component|pll|clk[3]	0	16	0	0
17	ddr_ck1ddr_dqs[1]	ddr2_clocks|altpll_component|pll|clk[3]	0	16	0	0
18	ddr_ck1ddr_dqs[2]	ddr2_clocks|altpll_component|pll|clk[3]	0	16	0	0
19	ddr_ck1ddr_dqs[3]	ddr2_clocks|altpll_component|pll|clk[3]	0	16	0	0
20	dtw_read_ddr_dqs[0]	ddr2_clocks|altpll_component|pll|clk[3]	0	false path	0	0
21	dtw_read_ddr_dqs[1]	ddr2_clocks|altpll_component|pll|clk[3]	0	false path	0	0
22	dtw_read_ddr_dqs[2]	ddr2_clocks|altpll_component|pll|clk[3]	0	false path	0	0

Report Exceptions

- Debug potential timing failures due to unintended or conflicting timing exceptions
  - Wildcards and Tcl lists are very convenient and commonly used to easily constrain parts of the design (e.g. set_false_path –from {sig[*] sig_ena sig_overflow} –to [all_outputs]
  - Users can easily miss paths, incorrectly relax paths, or accidentally override other exceptions
- Includes all exceptions that are set by the SDC constraints
  - set_false_path, set_multicycle, set_min_delay, and set_max_delay



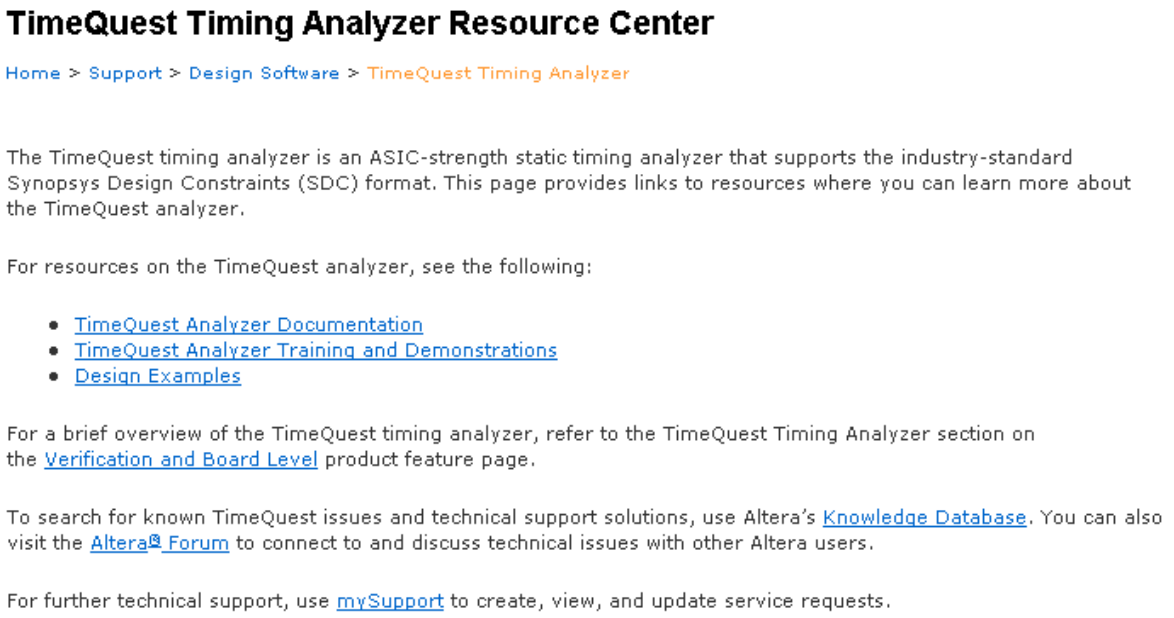


Report Exceptions Summary

	Status	Exception	Setup Slack	Hold Slack	Recovery Slack	Removal Slack
186	Complete	...R2_90_phy_alt_mem_phy_inst|clk|pll_reconfig_reset_ams_n|clrn)]	-3.492	n/a	n/a	n/a
187	Complete	...|quartusDDR2_90_phy_alt_mem_phy_inst|clk|phs_shft_busy_siii|d)]	-3.492	n/a	n/a	n/a
188	Invalid	...]} {mem_ba[0]} {mem_ba[1]} mem_cas_n mem_ras_n mem_we_n)]	n/a	n/a	n/a	n/a
189	Partially Overridden	... {mem_ba[0]} {mem_ba[1]} mem_cas_n mem_ras_n mem_we_n)] 2	4.997	n/a	n/a	n/a

Timequest resource center

www.altera.com/support/software/timequest/sof-qts-timequest.html



TimeQuest Timing Analyzer Resource Center

Home > Support > Design Software > TimeQuest Timing Analyzer

The TimeQuest timing analyzer is an ASIC-strength static timing analyzer that supports the industry-standard Synopsys Design Constraints (SDC) format. This page provides links to resources where you can learn more about the TimeQuest analyzer.

For resources on the TimeQuest analyzer, see the following:

- TimeQuest Analyzer Documentation
- TimeQuest Analyzer Training and Demonstrations
- Design Examples

For a brief overview of the TimeQuest timing analyzer, refer to the TimeQuest Timing Analyzer section on the Verification and Board Level product feature page.

To search for known TimeQuest issues and technical support solutions, use Altera's Knowledge Database. You can also visit the Altera® Forum to connect to and discuss technical issues with other Altera users.

For further technical support, use mySupport to create, view, and update service requests.

www.alterawiki.com/wiki/TimeQuest_User_Guide

A TimeQuest User Guide to help users understand the details of SDC constraints and using TimeQuest for static timing analysis

Thank You