Quartus® II Software Design Series: Timing Analysis

Timing Constraints

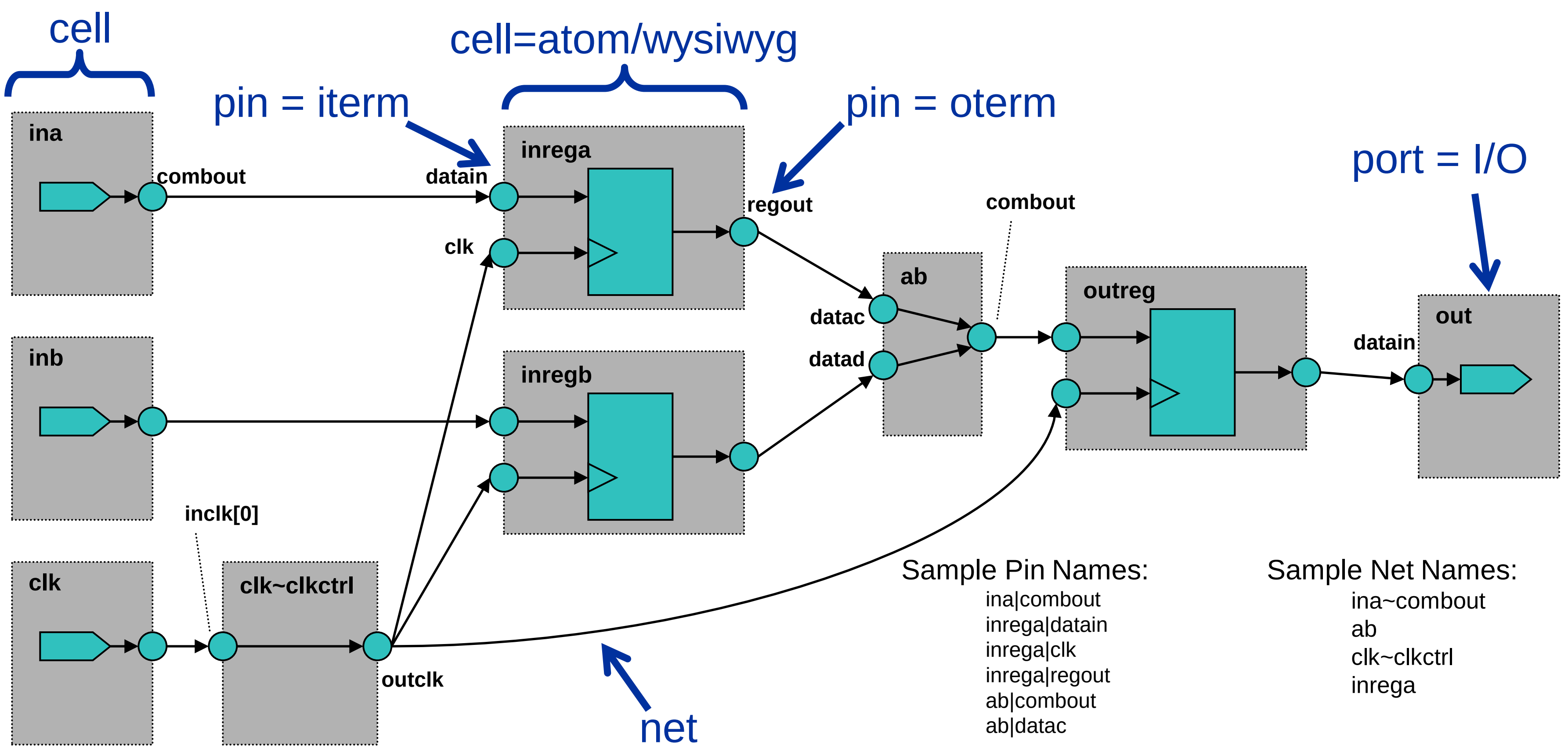
Importance of Constraining

- 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 timing analyzer 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



Sample Pin Names:
- ina|combout
- inrega|datain
- inrega|clk
- inrega|regout
- ab|combout
- ab|datac

Sample Net Names:
- ina~combout
- ab
- clk~clkctrl
- inrega

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

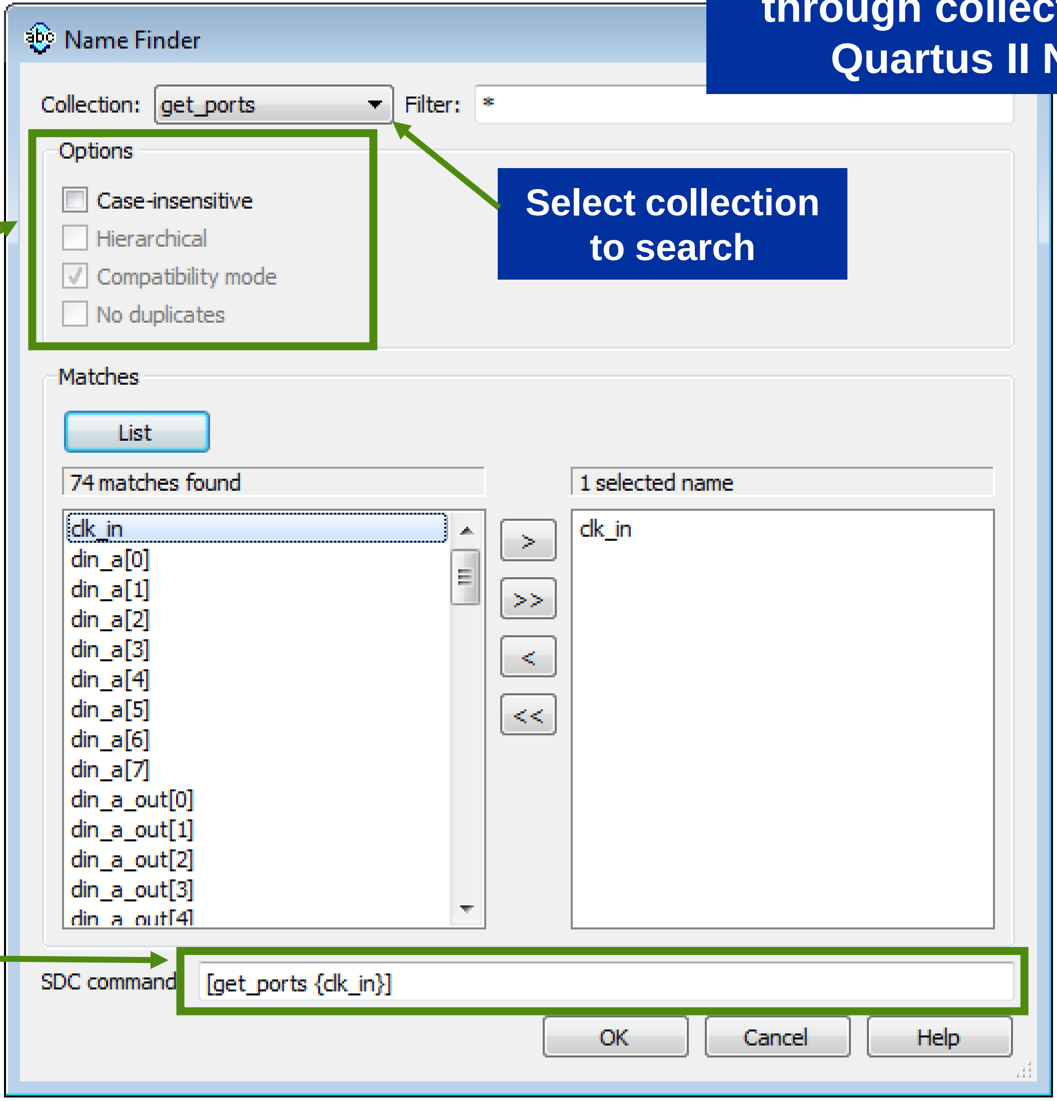
Collections

- 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

Name Finder

Clicking Browse button in any GUI dialog box opens Name Finder allowing you to search through collections (similar to Quartus II Node Finder)



clk_in
din_a[0]
din_a[1]
din_a[2]
din_a[3]
din_a[4]
din_a[5]
din_a[6]
din_a[7]
din_a_out[0]
din_a_out[1]
din_a_out[2]
din_a_out[3]
din_a_out[4]

Selected: clk_in

SDC command: [get_ports {clk_in}]

Select collection to search

Options available depend on selected collection

Edit command here or final command to use wildcards

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

SDC Timing Constraints

- Clocks ⟵
- I/O
- Asynchronous paths
- False paths
- Multicycle paths

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

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

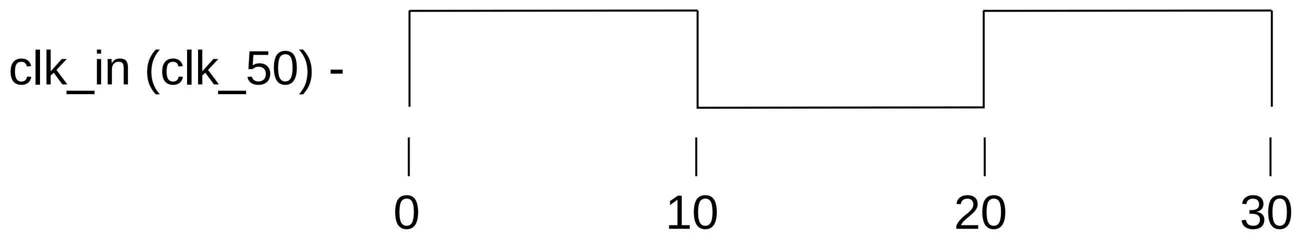
Note: In general, the more options added to a constraint command, the more specific the constraint is. When options are not specified, the constraint is more generalized and pertains to more of the target.

`create_clock` Notes

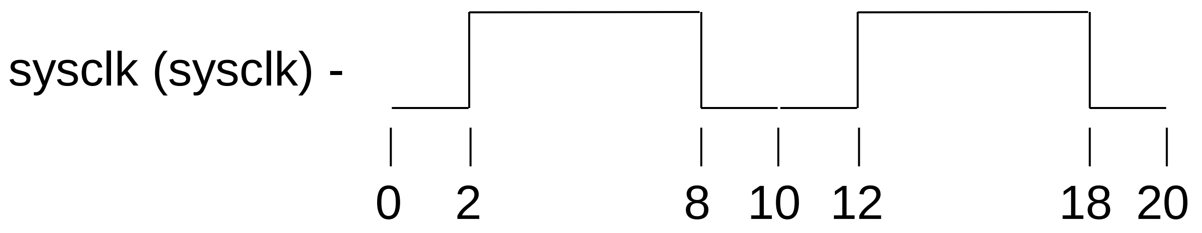
- `-name`: Assigns name to the clock to be used in other commands & reports when referring to clock
  - Optional; defaults to target name if not specified
- `-waveform`: Indicates clock offset or non-50% duty cycle clocks
  - 50% duty cycle (rise at 0 ns, fall halfway through period) is assumed unless otherwise indicated
- *`<targets>`*: Target ports or pins for clock setting
  - Virtual clock created if no target specified
- `-add`: Adds clock to node with existing clock
  - Without `-add`, warning given former clock constraint is over written

`create_clock` Examples

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

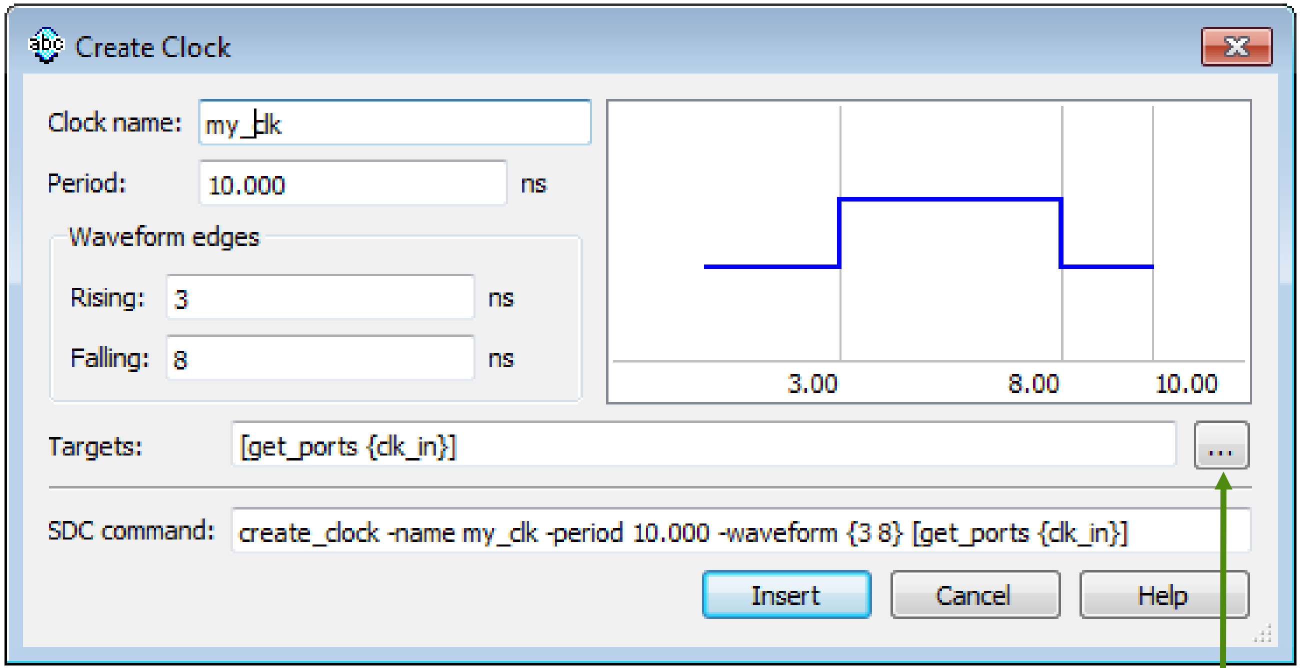


```
create_clock -period 10.0 -waveform {2.0 8.0} [get_ports sysclk]
```



Create Clock using GUI

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



Create Clock

Clock name: my_clk

Period: 10.000 ns

Waveform edges

Rising: 3 ns

Falling: 8 ns

Targets: [get_ports {clk_in}]

SDC command:

```
create_clock -name my_clk -period 10.000 -waveform {3 8} [get_ports {clk_in}]
```

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

Name Finder

Creating a Generated Clock

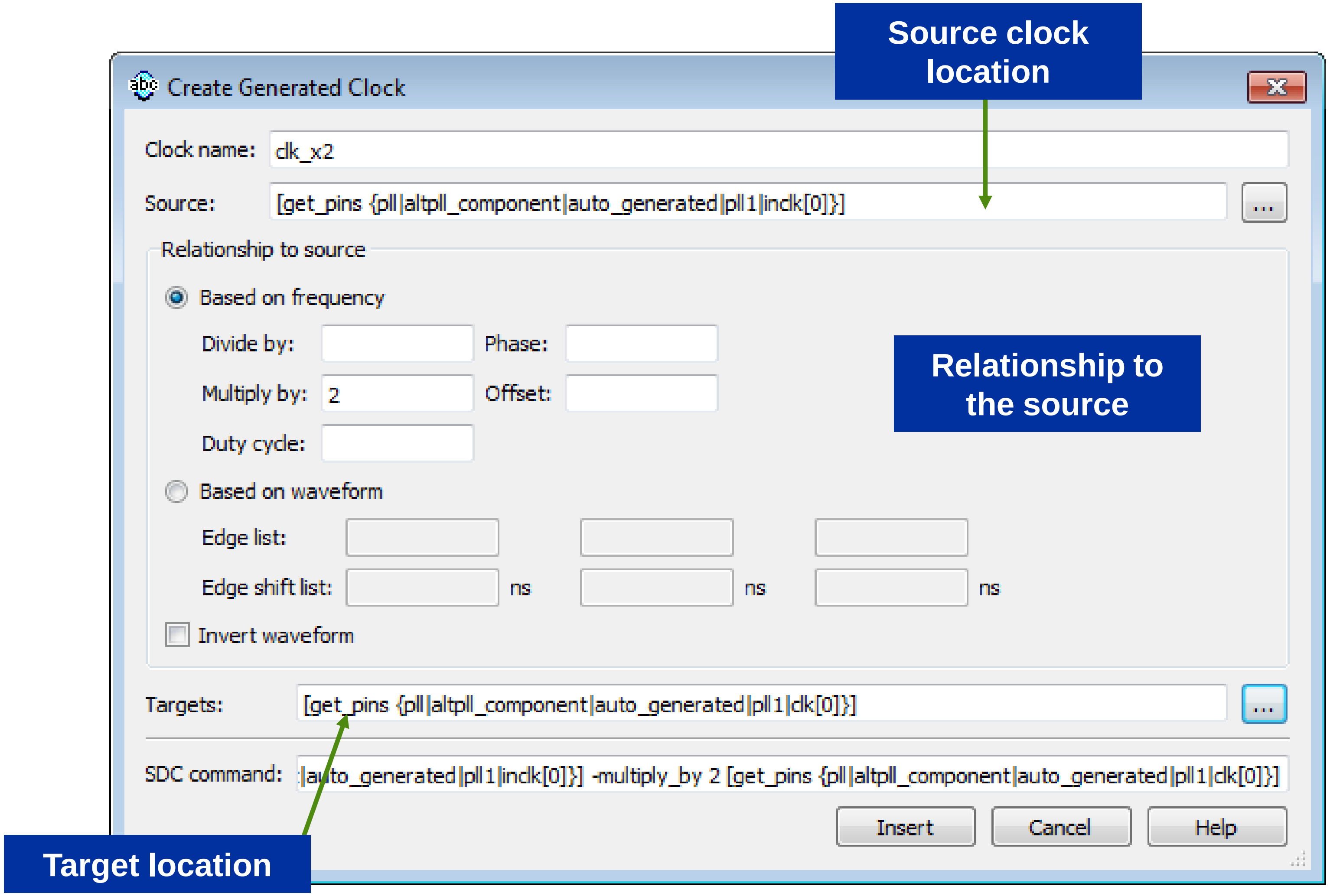
- Command: `create_generated_clock`
- Options

```
[-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`: Specifies the physical 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



Clock name: clk_x2

Source: [get_pins {pll|altpll_component|auto_generated|pll1|inclk[0]}]

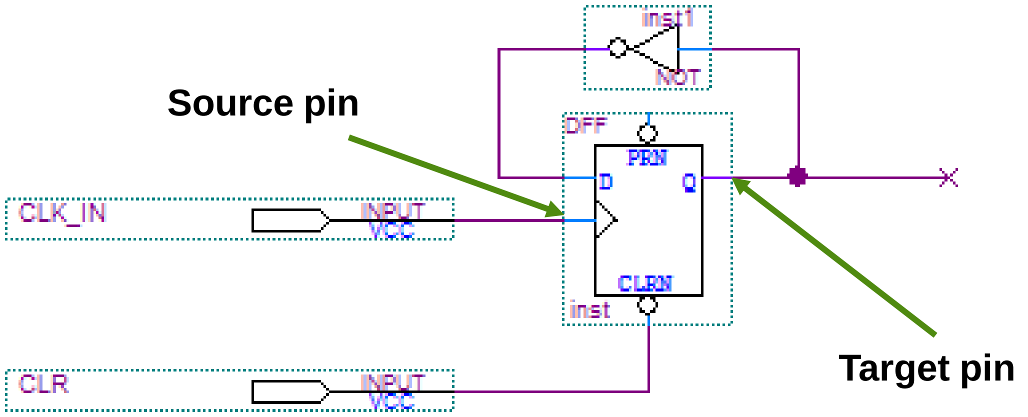
Relationship to source

- [✓ Based on frequency]
  - Divide by:
  - Phase:
  - Multiply by: 2
  - Offset:
  - Duty cycle:
- Based on waveform
  - Edge list:
  - Edge shift list: ns ns ns
- Invert waveform

Targets: [get_pins {pll|altpll_component|auto_generated|pll1|clk[0]}]

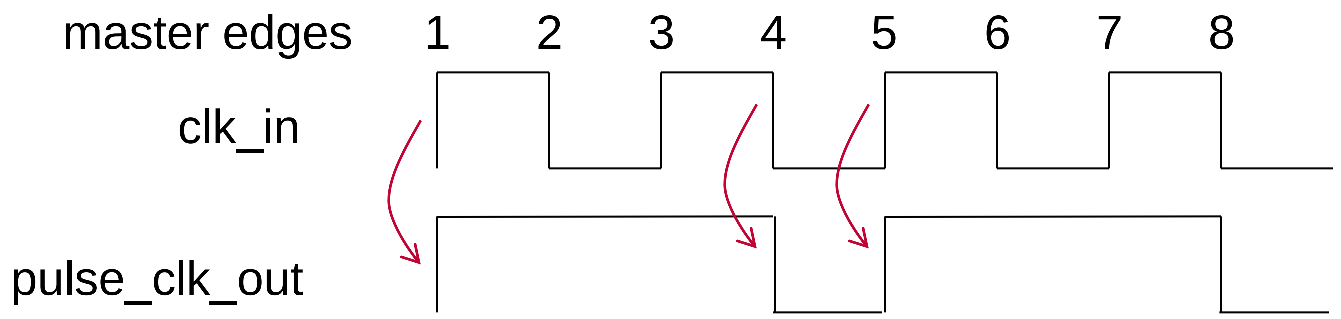
SDC command: |auto_generated|pll1|inclk[0]}] -multiply_by 2 [get_pins {pll|altpll_component|auto_generated|pll1|clk[0]}]

Generated Clock Example 1



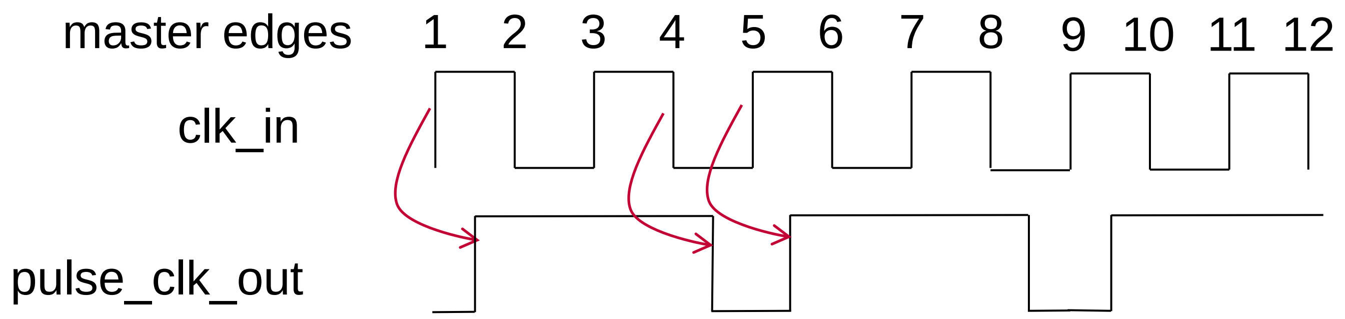
```
create_clock -period 10 [get_ports clk_in]

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.
```

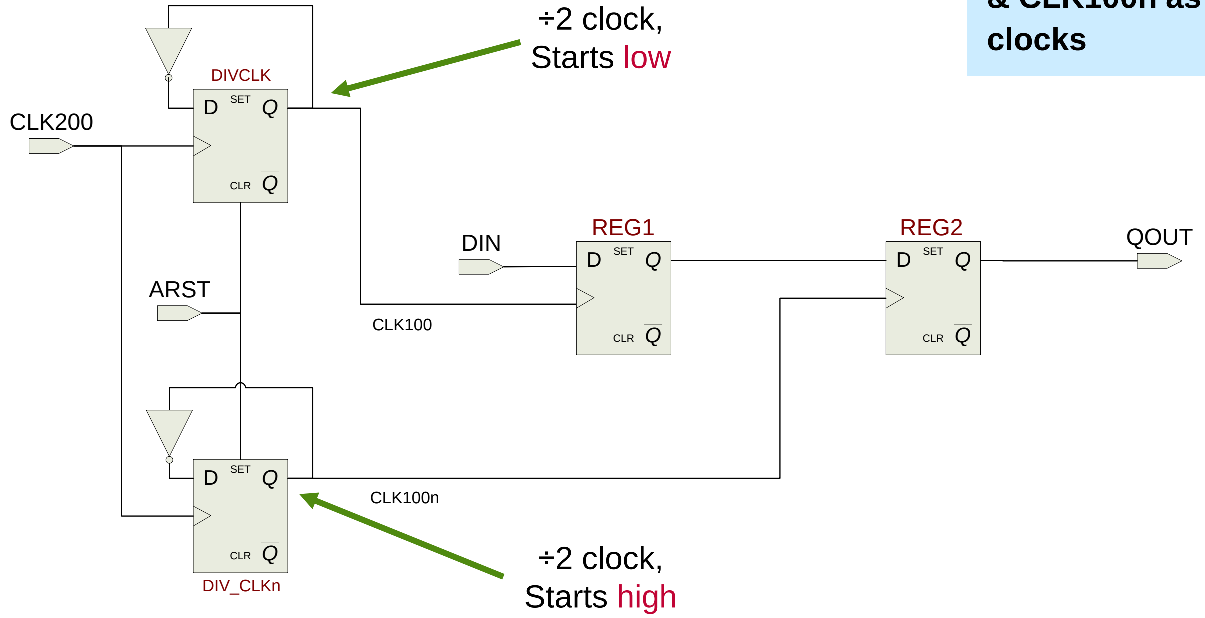
Generated Clock Example 3



```
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

Need to define CLK100 & CLK100n as generated clocks

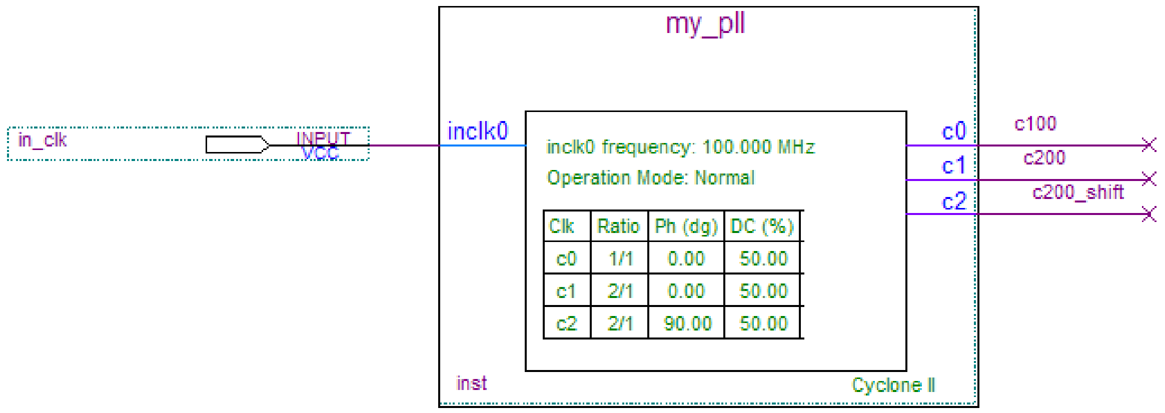


```
create_clock -period 5 [get_ports clk200]

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`
  - `[-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



Clk	Ratio	Ph (dg)	DC (%)
c0	1/1	0.00	50.00
c1	2/1	0.00	50.00
c2	2/1	90.00	50.00

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`
  - `derive_clock_uncertainty`
  - `set_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: 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>
```

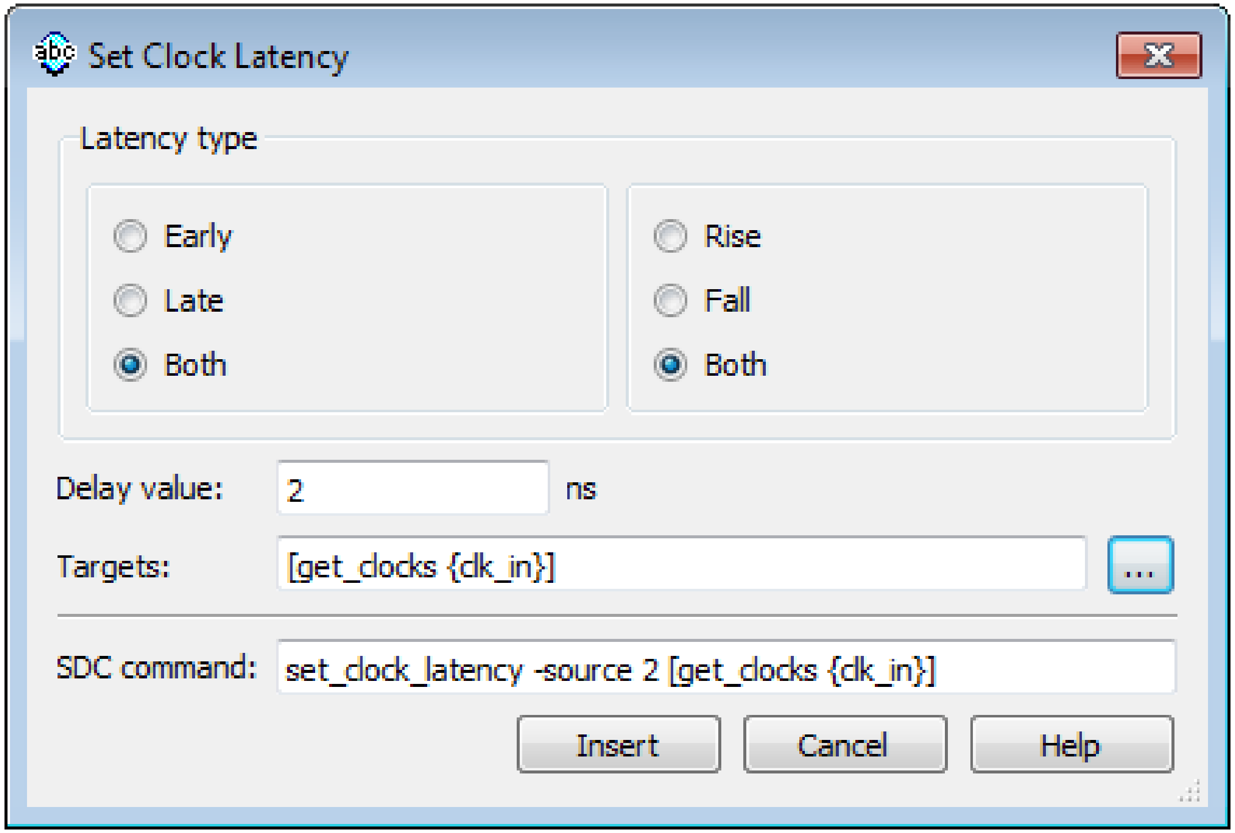
`set_clock_latency` Notes

- `-source`: required argument for constraint (no options)
- `-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
  - Setup analysis
    - Data ***arrival*** path (launch clock) uses ***late*** latency
    - Data ***required*** path (latch clock) uses ***early*** latency
  - Hold analysis
    - Data ***arrival*** path (launch clock) uses ***early*** latency
    - Data ***required*** path (latch clock) uses ***late*** latency
  - *As pessimistic as possible!*

`set_clock_latency` Notes (cont.)

- `-fall | -rise`: latency applied on only falling or rising edge of clock
- *`<target>`*: a clock input port or a previously constrained clock
  - All clock paths driven by clock affected
- `-clock` *`<clock list>`*: only needed if multiple clocks at clock input port

Clock Latency (GUI)


Set Clock Latency

Latency type

Early [ ]
Late [ ]
[✓ Both]

Rise [ ]
Fall [ ]
[✓ Both]

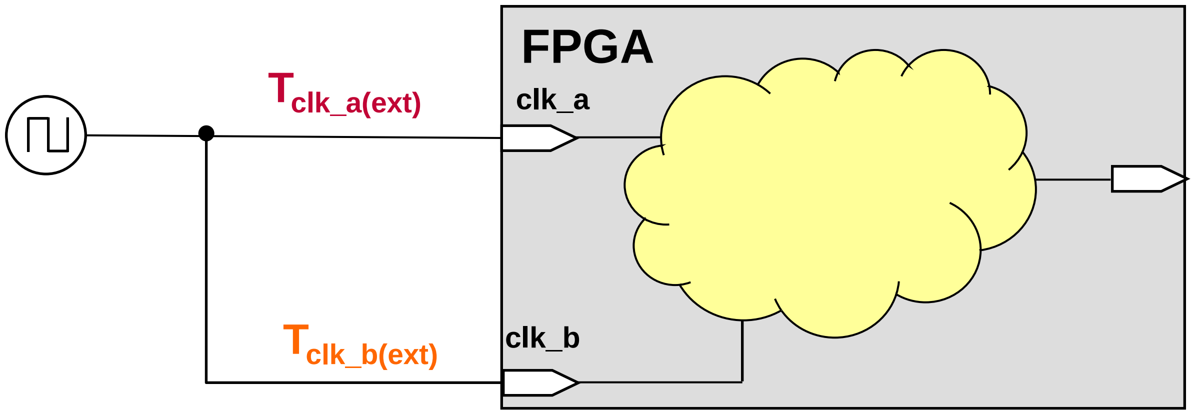
Delay value: 2 ns

Targets: [get_clocks {clk_in}]

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

Clock Latency Use Example

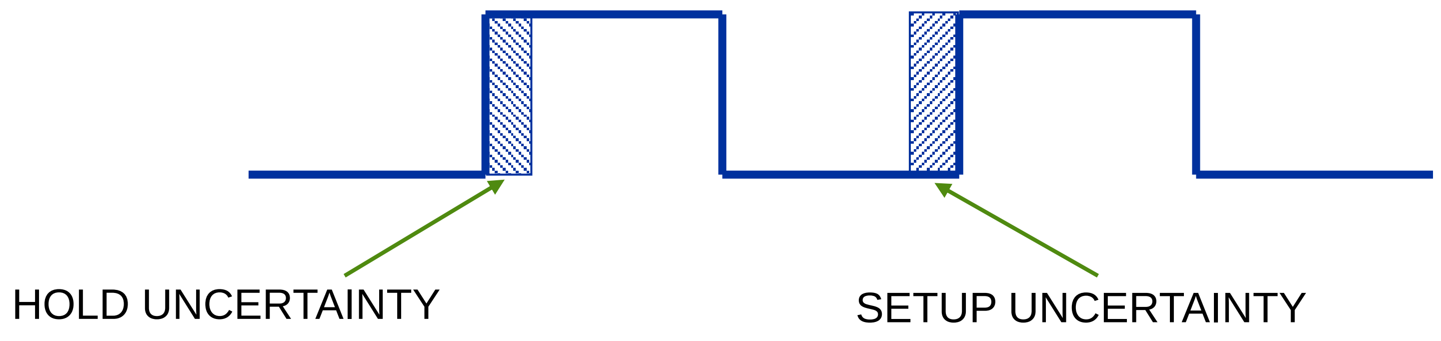
- Model difference in arrival time of clock to FPGA
- One clock can arrive at time "0" with latency on the other
- Or add latency to both clocks



$T_{clk\_a(ext)} \neq T_{clk\_b(ext)}$

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.

- Used to model jitter, guard band, or skew
  - Allows generation of clocks that are non-ideal

Manually Added Clock Uncertainty

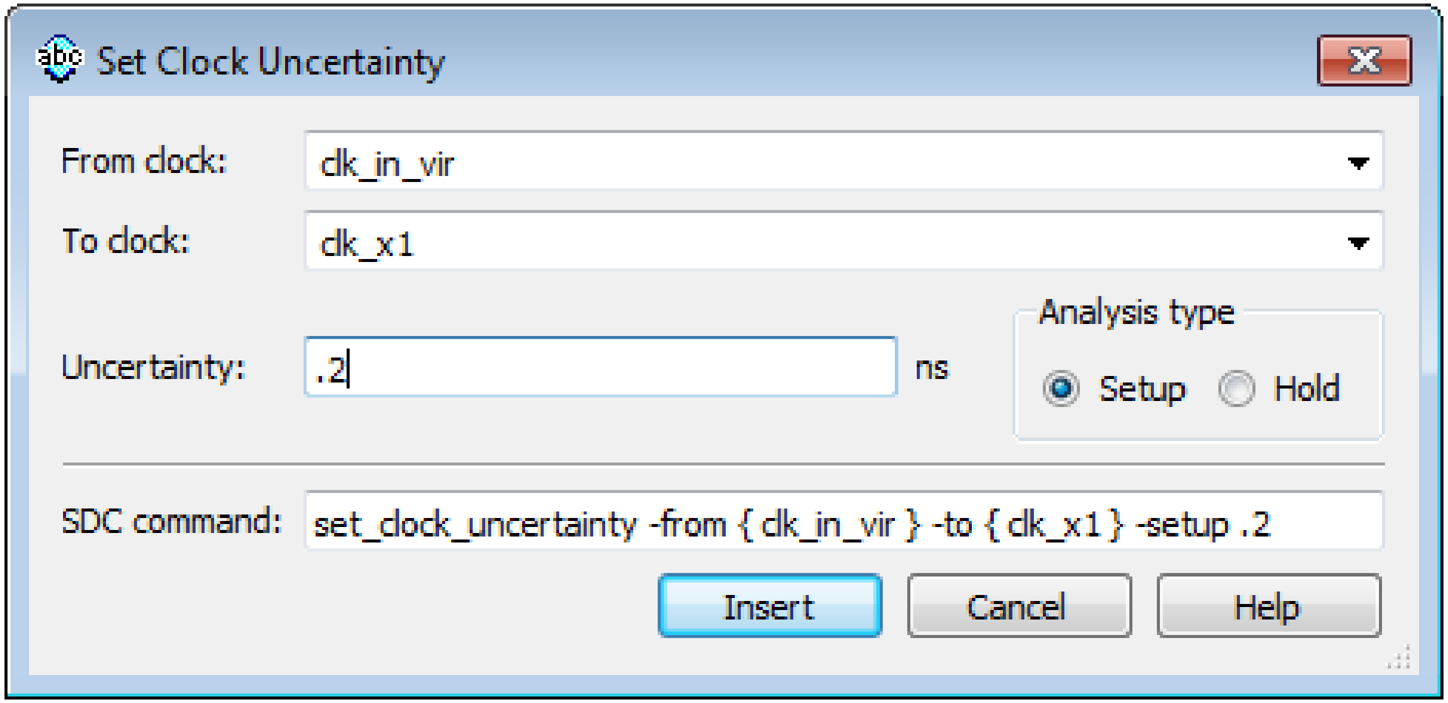
- Command: `set_clock_uncertainty`
- Manually add uncertainties between or within clock domains
- Options

```
[-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>
```

`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*
- Uncertainties created with `set_clock_uncertainty` have higher precedence than derived uncertainties

Clock Uncertainty (GUI)



From clock: clk_in_vir

To clock: clk_x1

Uncertainty: .2 ns

Analysis type: [✓ Setup] Hold

SDC command: set_clock_uncertainty -from { clk_in_vir } -to { clk_x1 } -setup .2

Automatically Derived Uncertainties

- Command: `derive_clock_uncertainty`
- Automatically derive clock uncertainties in supported devices
  - Cyclone III, Stratix II, HardCopy® II, and all newer devices
- Options
  - `[-overwrite]`: overwrites any existing uncertainty constraints
  - `[-add]`: adds derived uncertainties to existing constraints
- SDC extension expanded with `write_sdc -expand`
- View derived uncertainties with `report_sdc`

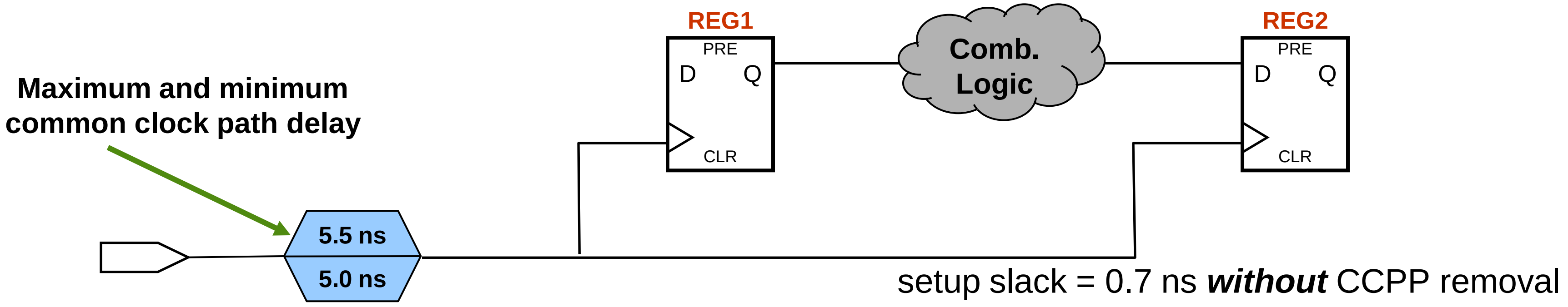
- *Not in GUI*
- *Use is recommended with all supported devices*
- *Results only seen after place and route*

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
    - 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

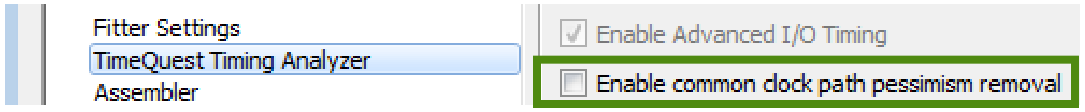
Common Clock Path Pessimism (CCPP) Removal

- 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



setup slack = 0.7 ns ***without*** CCPP removal
setup slack = 1.2 ns ***with*** CCPP removal

CCPP = 5.5 – 5.0 = 0.5 ns



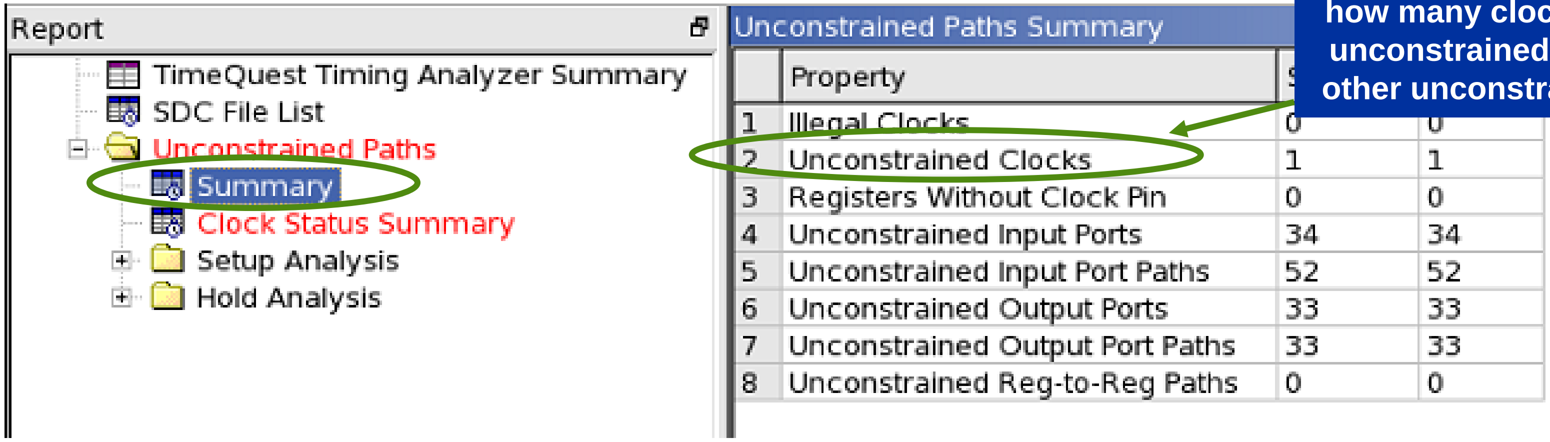
[✓] Enable Advanced I/O Timing
[ ] Enable common clock path pessimism removal

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 (`report_clocks`) to verify clocks are constrained correctly

Unconstrained Path Report

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

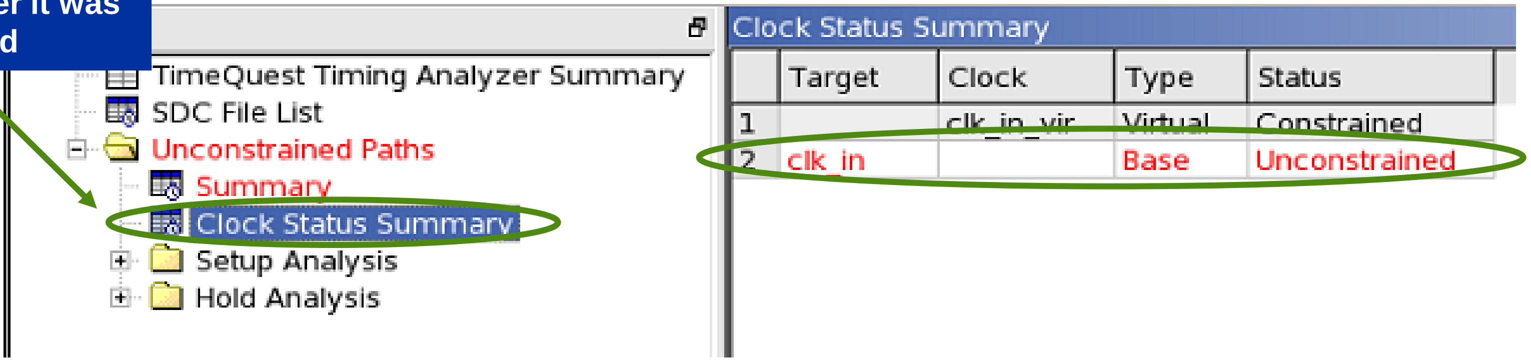


Report
- TimeQuest Timing Analyzer Summary
- SDC File List
- Unconstrained Paths
  - Summary
  - Clock Status Summary
- Setup Analysis
- Hold Analysis

Unconstrained Paths Summary

	Property	S	
1	Illegal Clocks	0	0
2	Unconstrained Clocks	1	1
3	Registers Without Clock Pin	0	0
4	Unconstrained Input Ports	34	34
5	Unconstrained Input Port Paths	52	52
6	Unconstrained Output Ports	33	33
7	Unconstrained Output Port Paths	33	33
8	Unconstrained Reg-to-Reg Paths	0	0

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



- TimeQuest Timing Analyzer Summary
- SDC File List
- Unconstrained Paths
  - Summary
  - Clock Status Summary
- Setup Analysis
- Hold Analysis

Clock Status Summary

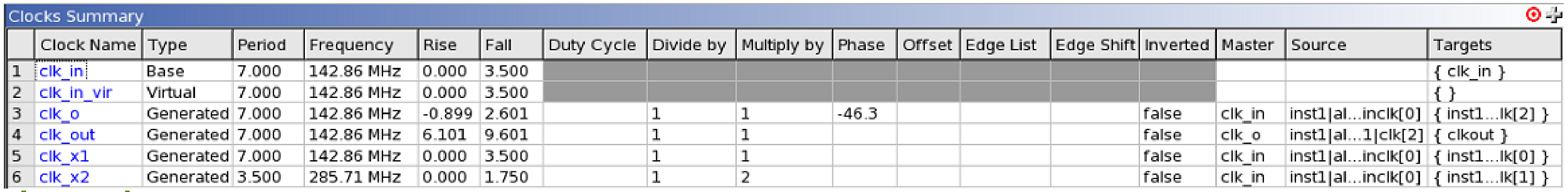
	Target	Clock	Type	Status
1		clk_in_vir	Virtual	Constrained
2	clk_in		Base	Unconstrained

ALTERA®

Clocks Summary (`report_clocks`)

- List details about the properties of constrained clocks

Clock properties



Clocks Summary

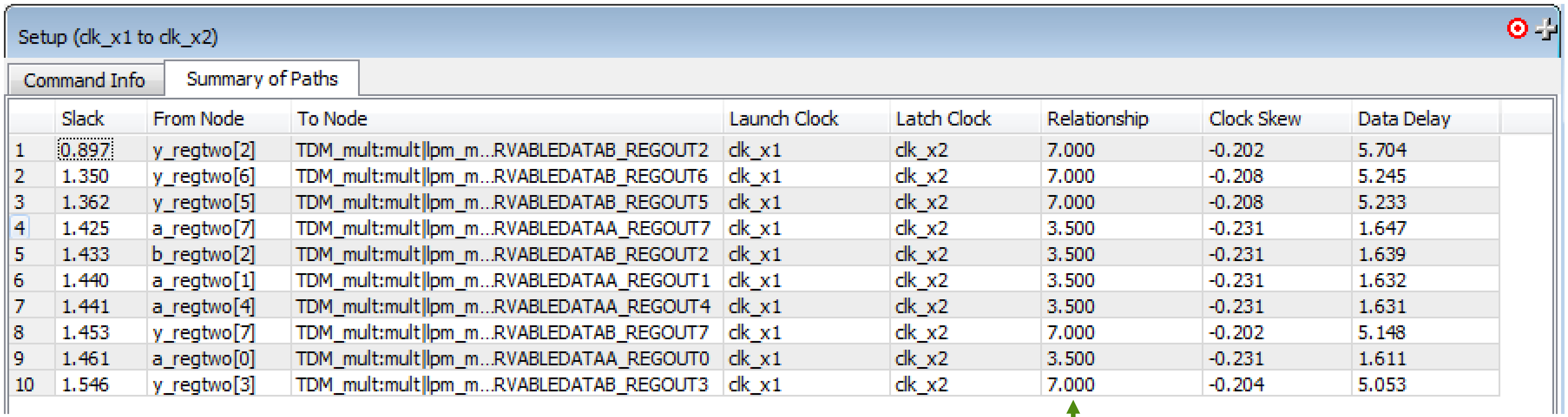
	Clock Name	Type	Period	Frequency	Rise	Fall	Duty Cycle	Divide by	Multiply by	Phase	Offset	Edge List	Edge Shift	Inverted	Master	Source	Targets
1	clk_in	Base	7.000	142.86 MHz	0.000	3.500											{ clk_in }
2	clk_in_vir	Virtual	7.000	142.86 MHz	0.000	3.500											{ }
3	clk_o	Generated	7.000	142.86 MHz	-0.899	2.601		1	1	-46.3				false	clk_in	inst1|al...inclk[0]	{ inst1...lk[2] }
4	clk_out	Generated	7.000	142.86 MHz	6.101	9.601		1	1					false	clk_o	inst1|al...1|clk[2]	{ clkout }
5	clk_x1	Generated	7.000	142.86 MHz	0.000	3.500		1	1					false	clk_in	inst1|al...inclk[0]	{ inst1...lk[0] }
6	clk_x2	Generated	3.500	285.71 MHz	0.000	1.750		1	2					false	clk_in	inst1|al...inclk[0]	{ inst1...lk[1] }

Clock names (`-name` argument or default name)

Verifying Clock Timing

- Generate Setup & Hold Summary reports to check worst slack for each clock
  - 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
  - **Report All Summaries** macro
- For detailed slack/path analysis
  - Run **Report Timing** from **Tasks** pane, **Constraints** menu, or right-click menu of path to analyze
  - Use `report_timing` command

Recall: Setup Report (Summary)



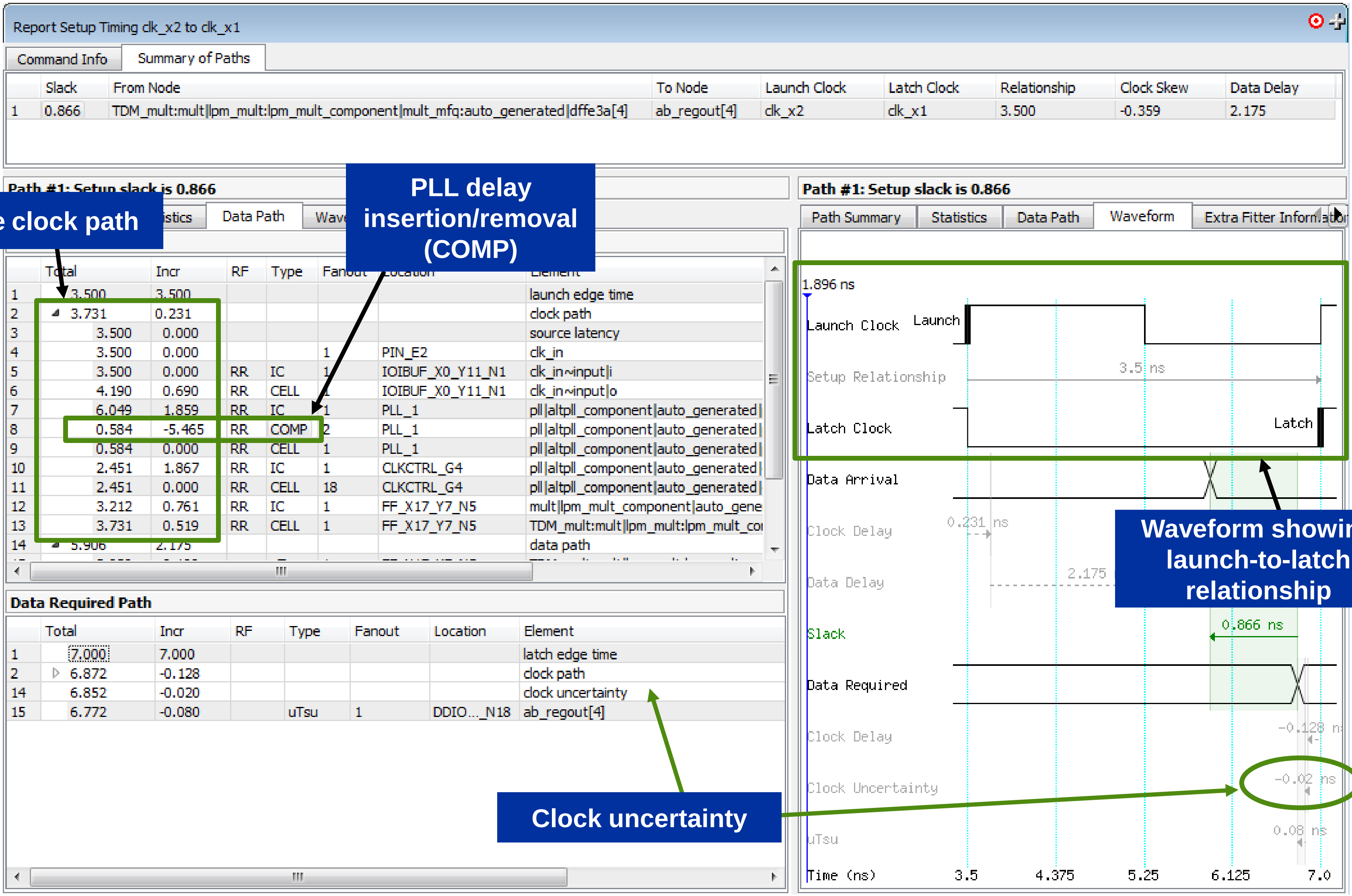
Setup (clk_x1 to clk_x2)

Command Info | Summary of Paths

	Slack	From Node	To Node	Launch Clock	Latch Clock	Relationship	Clock Skew	Data Delay
1	0.897	y_regtwo[2]	TDM_mult:mult|lpm_m...RVABLEDATAB_REGOUT2	clk_x1	clk_x2	7.000	-0.202	5.704
2	1.350	y_regtwo[6]	TDM_mult:mult|lpm_m...RVABLEDATAB_REGOUT6	clk_x1	clk_x2	7.000	-0.208	5.245
3	1.362	y_regtwo[5]	TDM_mult:mult|lpm_m...RVABLEDATAB_REGOUT5	clk_x1	clk_x2	7.000	-0.208	5.233
4	1.425	a_regtwo[7]	TDM_mult:mult|lpm_m...RVABLEDATAA_REGOUT7	clk_x1	clk_x2	3.500	-0.231	1.647
5	1.433	b_regtwo[2]	TDM_mult:mult|lpm_m...RVABLEDATAB_REGOUT2	clk_x1	clk_x2	3.500	-0.231	1.639
6	1.440	a_regtwo[1]	TDM_mult:mult|lpm_m...RVABLEDATAA_REGOUT1	clk_x1	clk_x2	3.500	-0.231	1.632
7	1.441	a_regtwo[4]	TDM_mult:mult|lpm_m...RVABLEDATAA_REGOUT4	clk_x1	clk_x2	3.500	-0.231	1.631
8	1.453	y_regtwo[7]	TDM_mult:mult|lpm_m...RVABLEDATAB_REGOUT7	clk_x1	clk_x2	7.000	-0.202	5.148
9	1.461	a_regtwo[0]	TDM_mult:mult|lpm_m...RVABLEDATAA_REGOUT0	clk_x1	clk_x2	3.500	-0.231	1.611
10	1.546	y_regtwo[3]	TDM_mult:mult|lpm_m...RVABLEDATAB_REGOUT3	clk_x1	clk_x2	7.000	-0.204	5.053

Launch-to-latch relationship based directly on clock constraints (see the Rise and Fall columns of the Clock Report)

Example Setup Report (Full Detail)



Report Setup Timing clk_x2 to clk_x1

Command Info | Summary of Paths

	Slack	From Node	To Node	Launch Clock	Latch Clock	Relationship	Clock Skew	Data Delay
1	0.866	TDM_mult:mult|lpm_mult:lpm_mult_component|mult_mfq:auto_generated|dffe3a[4]	ab_regout[4]	clk_x2	clk_x1	3.500	-0.359	2.175

Path #1: Setup slack is 0.866

	Total	Incr	RF	Type	Fanout	Location	Element
1	3.500	3.500					launch edge time
2	3.731	0.231					clock path
3	3.500	0.000					source latency
4	3.500	0.000			1	PIN_E2	clk_in
5	3.500	0.000	RR	IC	1	IOIBUF_X0_Y11_N1	clk_in~input|i
6	4.190	0.690	RR	CELL	1	IOIBUF_X0_Y11_N1	clk_in~input|o
7	6.049	1.859	RR	IC	1	PLL_1	pll|altpll_component|auto_generated|
8	0.584	-5.465	RR	COMP	2	PLL_1	pll|altpll_component|auto_generated|
9	0.584	0.000	RR	CELL	1	PLL_1	pll|altpll_component|auto_generated|
10	2.451	1.867	RR	IC	1	CLKCTRL_G4	pll|altpll_component|auto_generated|
11	2.451	0.000	RR	CELL	18	CLKCTRL_G4	pll|altpll_component|auto_generated|
12	3.212	0.761	RR	IC	1	FF_X17_Y7_N5	mult|lpm_mult_component|auto_gene
13	3.731	0.519	RR	CELL	1	FF_X17_Y7_N5	TDM_mult:mult|lpm_mult:lpm_mult_co
14	5.906	2.175					data path

Data Required Path

	Total	Incr	RF	Type	Fanout	Location	Element
1	7.000	7.000					latch edge time
2	6.872	-0.128					clock path
14	6.852	-0.020					clock uncertainty
15	6.772	-0.080		uTsu	1	DDIO..._N18	ab_regout[4]

Path #1: Setup slack is 0.866

Trace clock path

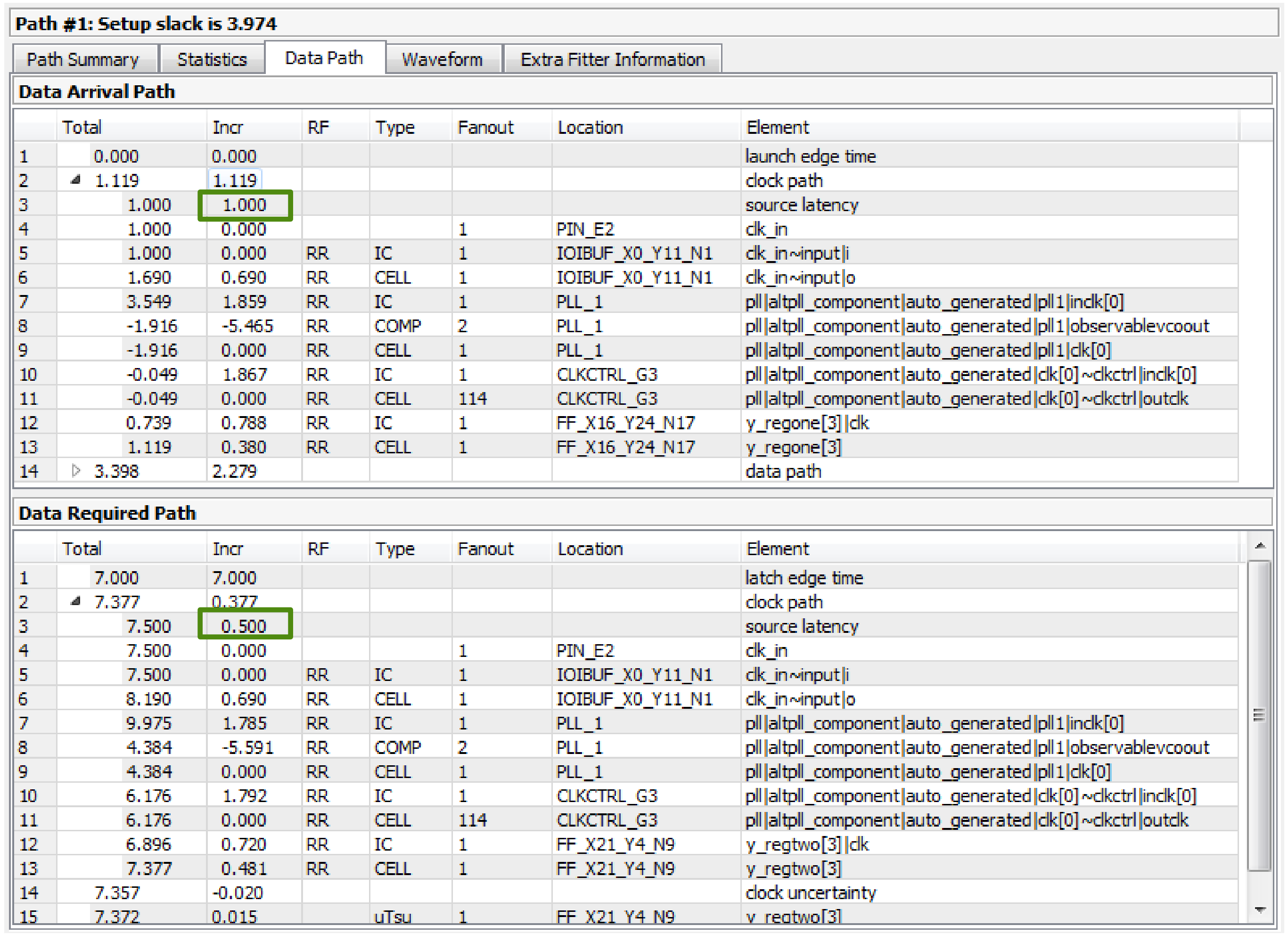
PLL delay insertion/removal (COMP)

Waveform showing launch-to-latch relationship

Clock uncertainty

Example Setup Report with Latency

```
set_clock_latency -source -early .5 [get_clocks clk_in]
set_clock_latency -source -late 1 [get_clocks clk_in]
```



Path #1: Setup slack is 3.974

Data Arrival Path

	Total	Incr	RF	Type	Fanout	Location	Element
1	0.000	0.000					launch edge time
2	◢ 1.119	1.119					clock path
3	1.000	1.000					source latency
4	1.000	0.000			1	PIN_E2	clk_in
5	1.000	0.000	RR	IC	1	IOIBUF_X0_Y11_N1	clk_in~input|i
6	1.690	0.690	RR	CELL	1	IOIBUF_X0_Y11_N1	clk_in~input|o
7	3.549	1.859	RR	IC	1	PLL_1	pll|altpll_component|auto_generated|pll1|inclk[0]
8	-1.916	-5.465	RR	COMP	2	PLL_1	pll|altpll_component|auto_generated|pll1|observablevcoout
9	-1.916	0.000	RR	CELL	1	PLL_1	pll|altpll_component|auto_generated|pll1|clk[0]
10	-0.049	1.867	RR	IC	1	CLKCTRL_G3	pll|altpll_component|auto_generated|clk[0]~clkctrl|inclk[0]
11	-0.049	0.000	RR	CELL	114	CLKCTRL_G3	pll|altpll_component|auto_generated|clk[0]~clkctrl|outclk
12	0.739	0.788	RR	IC	1	FF_X16_Y24_N17	y_regone[3]|clk
13	1.119	0.380	RR	CELL	1	FF_X16_Y24_N17	y_regone[3]
14	▷ 3.398	2.279					data path

Data Required Path

	Total	Incr	RF	Type	Fanout	Location	Element
1	7.000	7.000					latch edge time
2	◢ 7.377	0.377					clock path
3	7.500	0.500					source latency
4	7.500	0.000			1	PIN_E2	clk_in
5	7.500	0.000	RR	IC	1	IOIBUF_X0_Y11_N1	clk_in~input|i
6	8.190	0.690	RR	CELL	1	IOIBUF_X0_Y11_N1	clk_in~input|o
7	9.975	1.785	RR	IC	1	PLL_1	pll|altpll_component|auto_generated|pll1|inclk[0]
8	4.384	-5.591	RR	COMP	2	PLL_1	pll|altpll_component|auto_generated|pll1|observablevcoout
9	4.384	0.000	RR	CELL	1	PLL_1	pll|altpll_component|auto_generated|pll1|clk[0]
10	6.176	1.792	RR	IC	1	CLKCTRL_G3	pll|altpll_component|auto_generated|clk[0]~clkctrl|inclk[0]
11	6.176	0.000	RR	CELL	114	CLKCTRL_G3	pll|altpll_component|auto_generated|clk[0]~clkctrl|outclk
12	6.896	0.720	RR	IC	1	FF_X21_Y4_N9	y_regtwo[3]|clk
13	7.377	0.481	RR	CELL	1	FF_X21_Y4_N9	y_regtwo[3]
14	7.357	-0.020					clock uncertainty
15	7.372	0.015		uTsu	1	FF_X21_Y4_N9	y_regtwo[3]

Please go to Exercise 2

SDC Timing Constraints

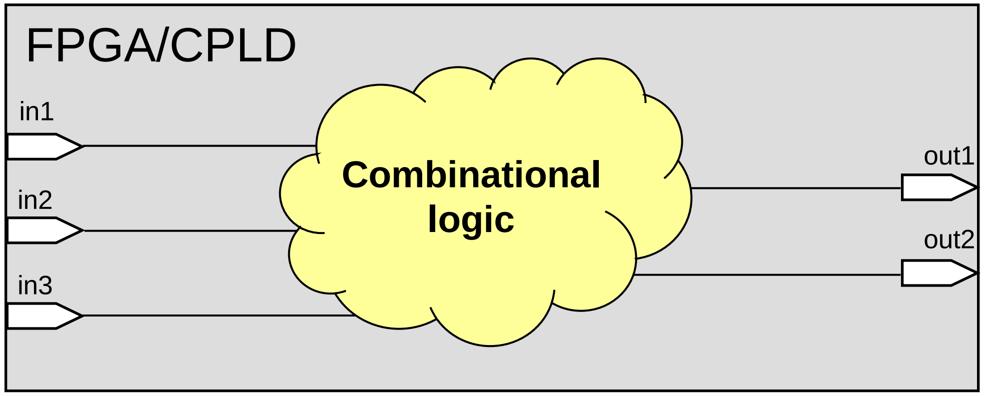
- Clocks
- I/O ←
- Asynchronous paths
- False paths
- Multicycle paths

I/O Constraints

- Combinational I/O interface
- Synchronous I/O 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



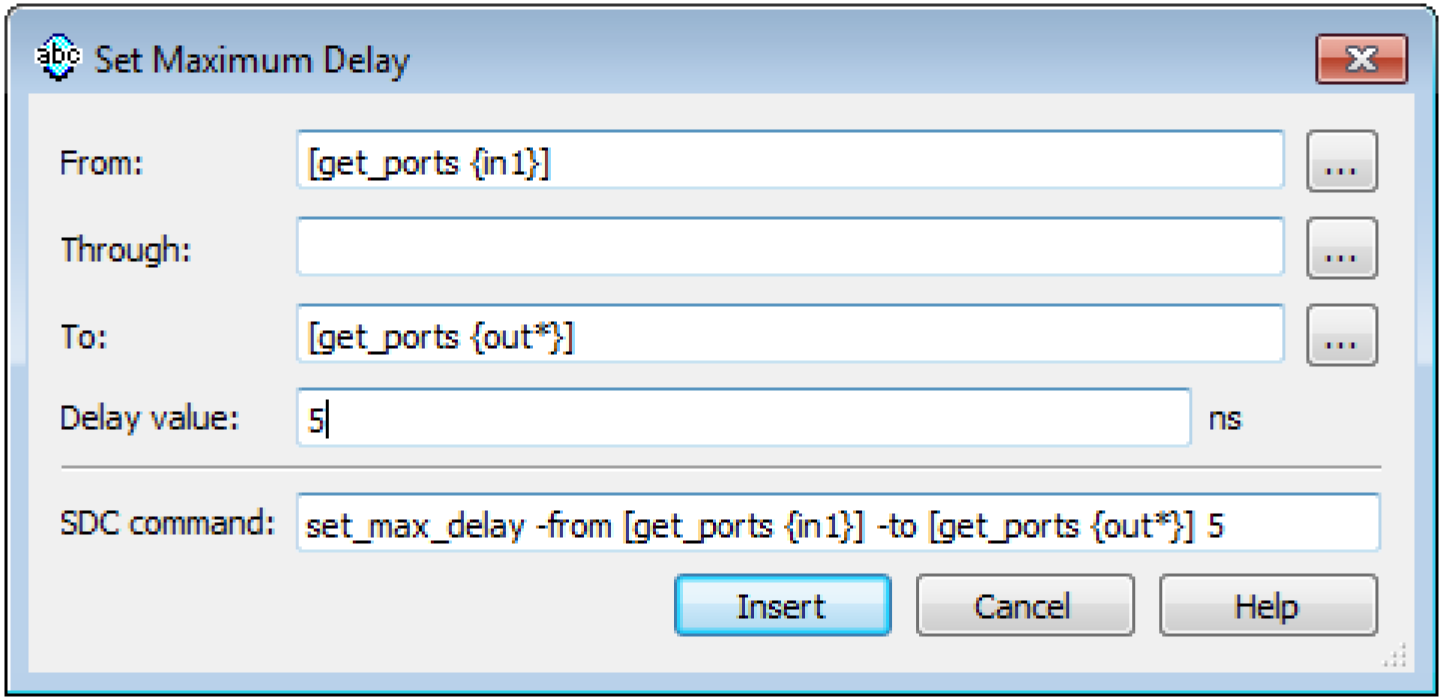
- Options

```
[-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)



From: [get_ports {in1}]

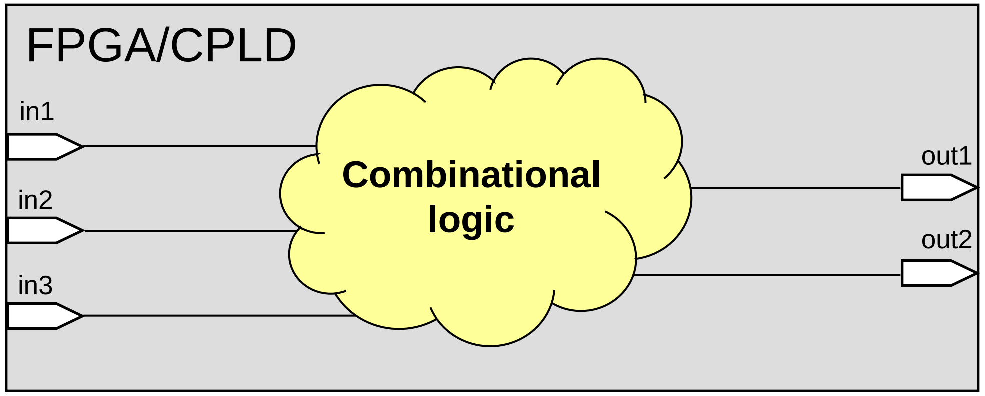
Through:

To: [get_ports {out*}]

Delay value: 5 ns

SDC command: set_max_delay -from [get_ports {in1}] -to [get_ports {out*}] 5

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
```

Constraining Synchronous I/O

- Two methods to constrain I/O, depending on what I/O information is known
  - Can use both in the same design/SDC file

1. Using external timing parameters
   - Surrounding chips, board delays, chip-to-chip skews are provided and Quartus II software derives required FPGA T_{su}, T_h, T_{co}
2. Using FPGA timing requirements
   - Target T_{su}, T_h, T_{co} specs are pre-determined by user/board
   - Useful when FPGA may end up in a variety of environments or interacts with defined bus interface (e.g. PCI)

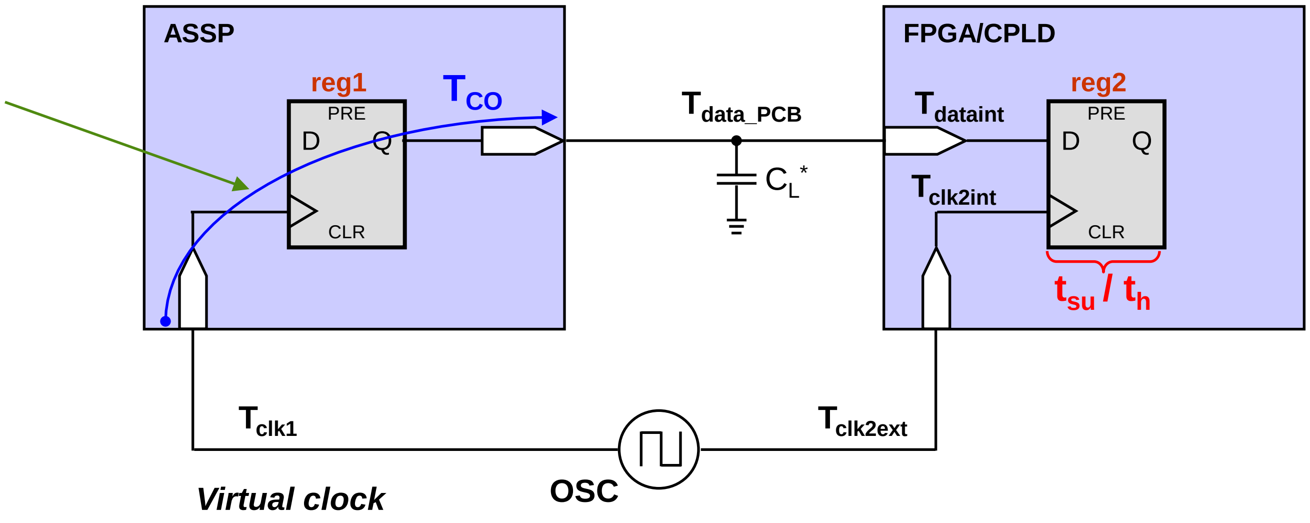
I/O Timing – Virtual Clocks

- Recommended to use virtual clocks for specifying input / output delays
- Separate clock to represent external clock timing allows `derive_clock_uncertainty` to calculate I/O transfer uncertainties 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

T_{co} represents total clock-to-output time of ASSP (i.e. datasheet spec)

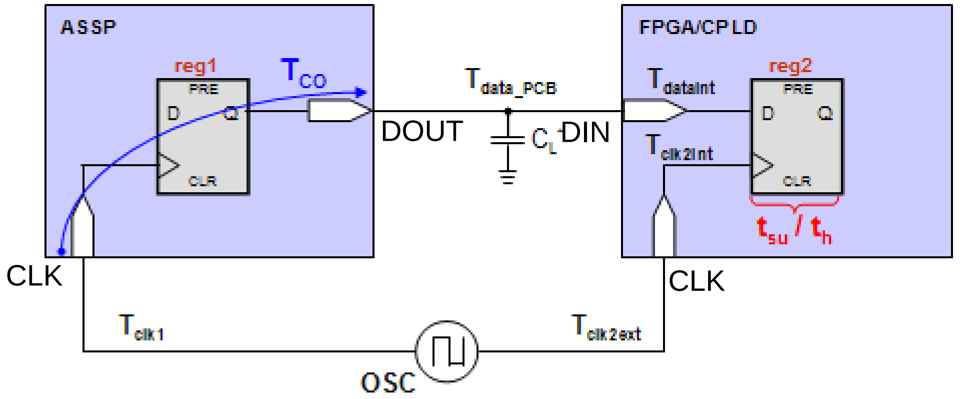


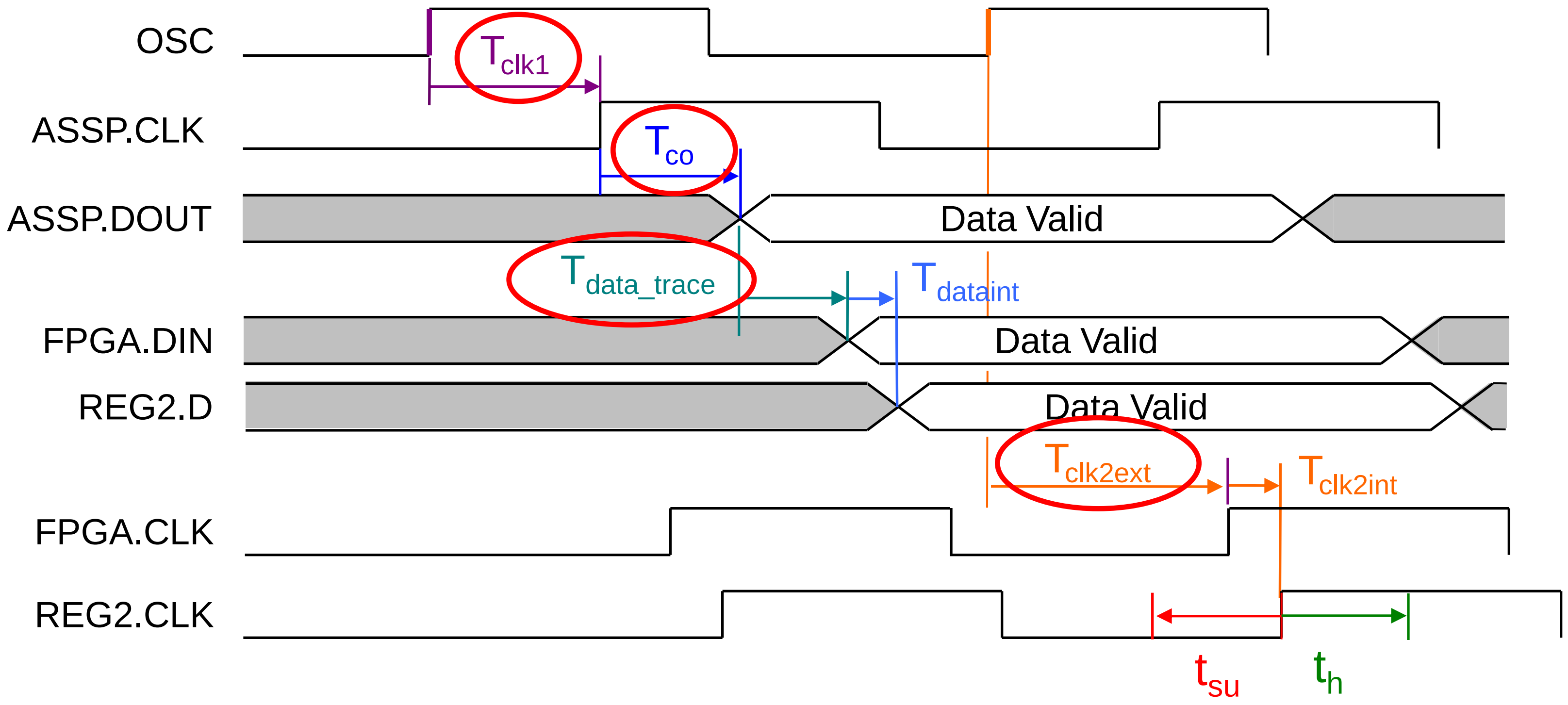
* Represents delay due to capacitive loading

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

Synchronous Inputs





Constraining Synchronous Inputs

- Using external parameters to calculate maximum external delay:

input delay max **= Data trace (max) – Board clock skew (min) + $T_{co(max)}$**

$= (T_{data\_PCB(max)} + T_{CL}) - (T_{clk2ext(min)} - T_{clk1(max)}) + T_{co(max)}$

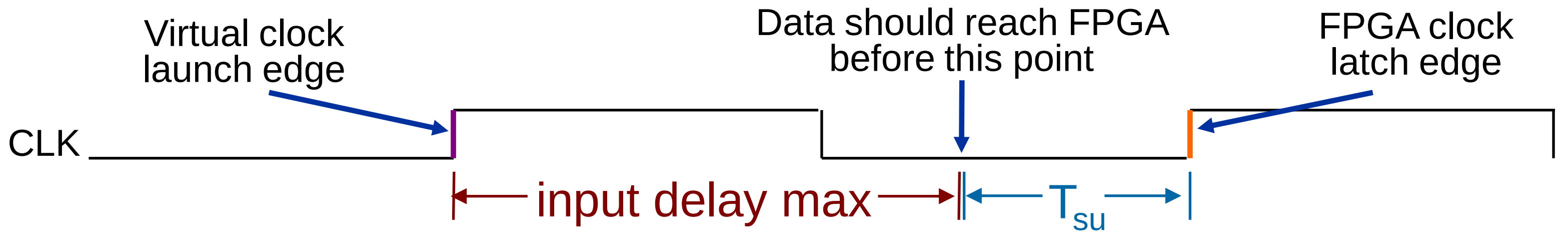
- Using external parameters to calculate minimum external delay:

input delay min **= Data trace (min) - Board clock skew (max) + $T_{co(min)}$**

$= (T_{data\_PCB(min)} + T_{CL}) - (T_{clk2ext(max)} - T_{clk1(min)}) + T_{co(min)}$

Constraining Synchronous Inputs (cont)

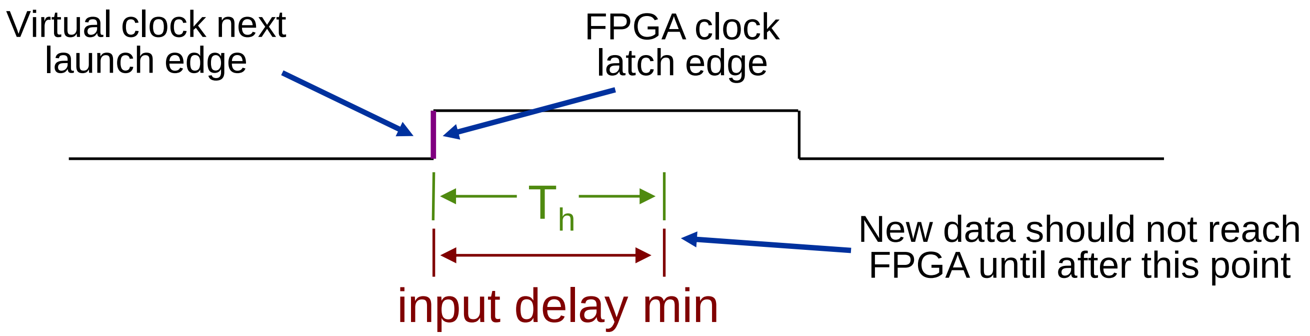
- Using FPGA setup requirements (T_{su} at FPGA boundary)



Virtual clock launch edge
Data should reach FPGA before this point
FPGA clock latch edge
CLK
input delay max
T_{su}

input delay max = $(T_{latch} – T_{launch}) - T_{SU}$

- Using FPGA hold requirements (T_h at FPGA boundary)



Virtual clock next launch edge
FPGA clock latch edge
T_h
New data should not reach FPGA until after this point
input delay min

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

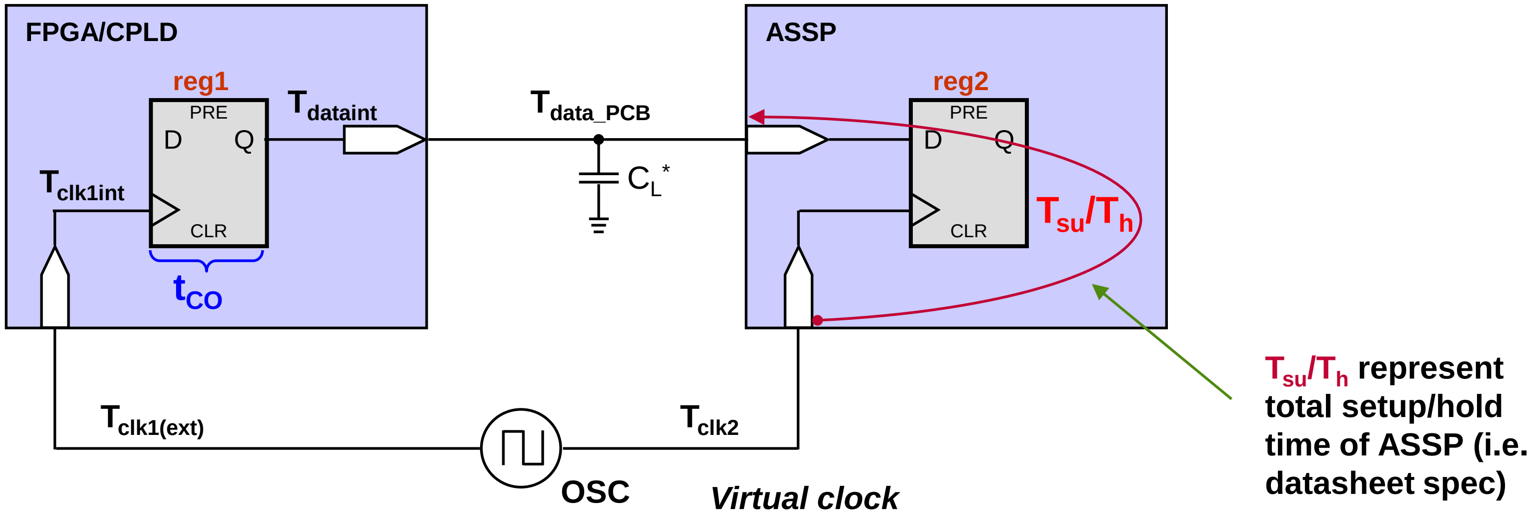
- `-clock`: Specifies the clock driving the source (external) register
  - Use the virtual clock
  - Used to determine launch edge vs. latch edge relationship
- `-clock_fall`: Use to specify input signal was launched by a falling edge clock transition
- `-rise` | `-fall`: Use to indicate whether input 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



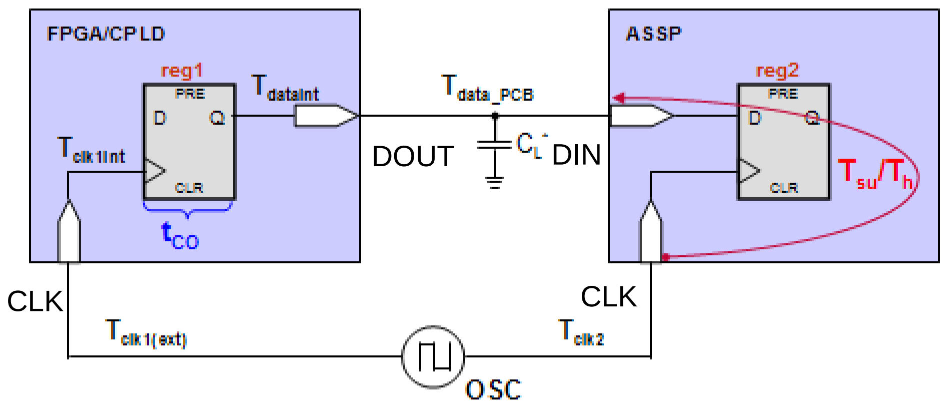
** Represents delay due to capacitive loading*

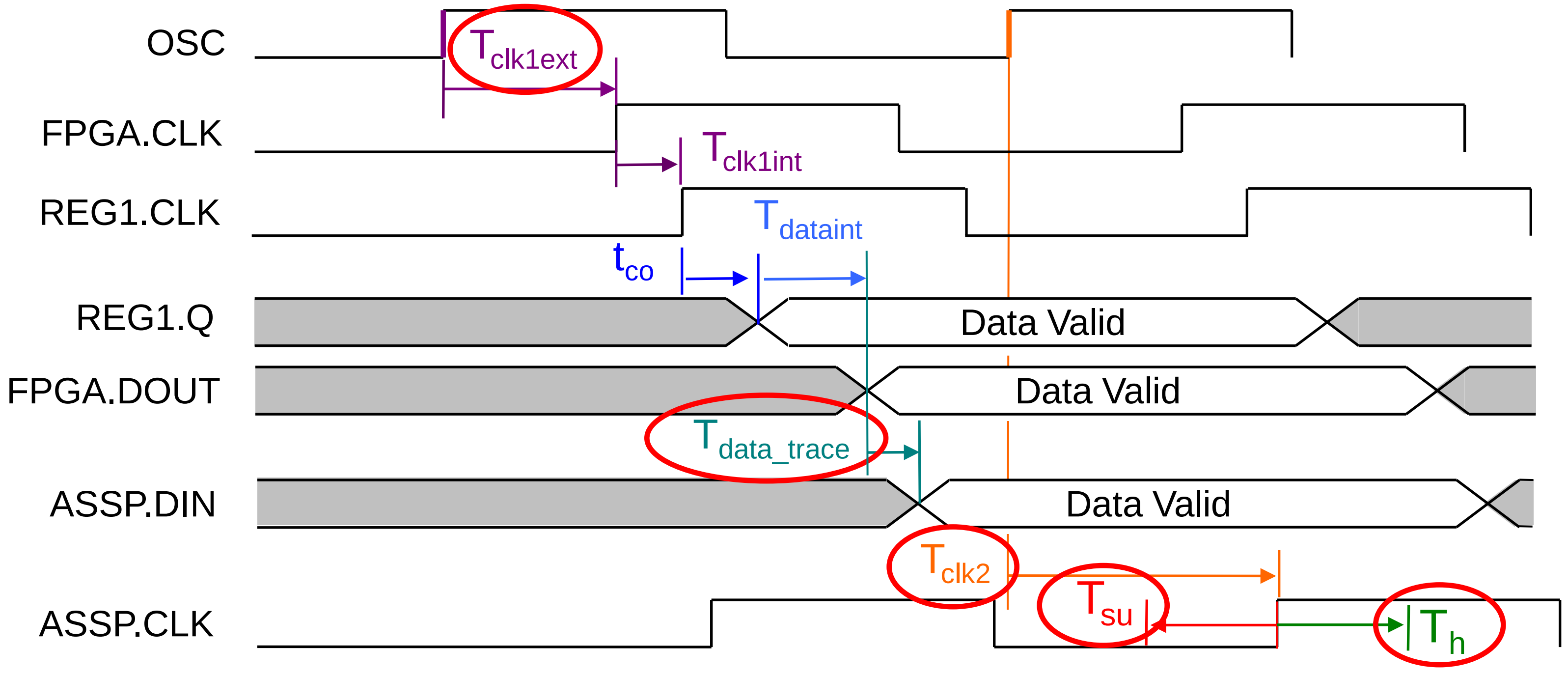
Constraining Synchronous Outputs

- 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 maximum output delay value represents all delays external to device

- 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 minimum output delay value represents all delays external to device

Synchronous Outputs





Constraining Synchronous Outputs

- Using external parameters to calculate maximum external delay:

output delay max **= Data trace (max) – Board clock skew (min) + T_{SU}**

$= (T_{data\_PCB(max)} + T_{CL}) - (T_{clk2(min)} - T_{clk1ext(max)}) + T_{su}$

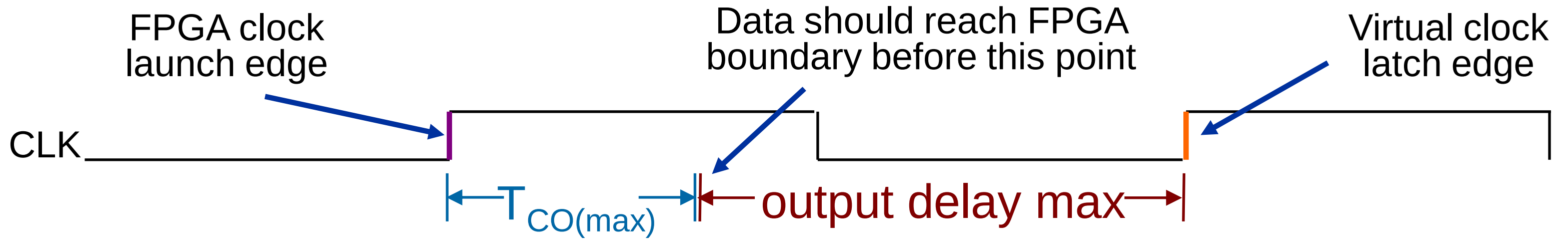
- Using external parameters to calculate minimum external delay:

output delay min **= Data trace (min) - Board clock skew (max) – T_h**

$= (T_{data\_PCB(min)} + T_{CL}) - (T_{clk2(max)} - T_{clk1ext(min)}) - T_h$

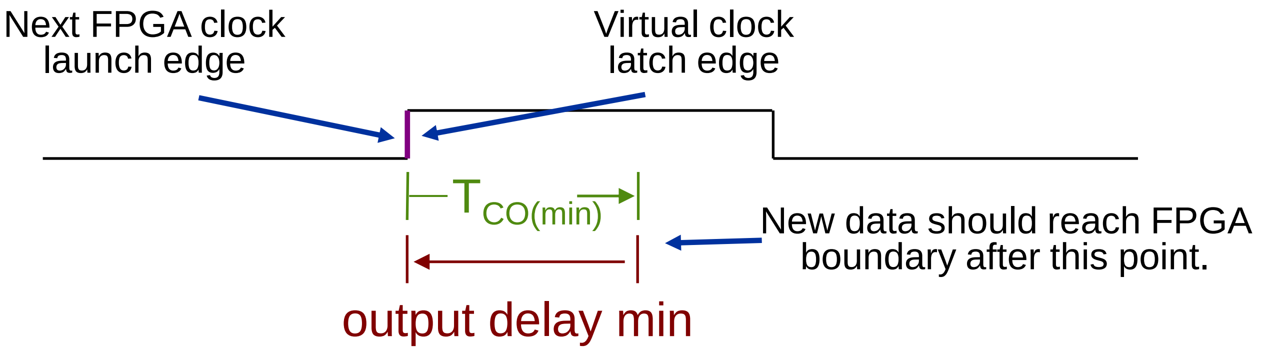
Constraining Synchronous Outputs (cont)

- Using FPGA maximum T_{CO} requirements ($T_{CO(max)}$ at FPGA boundary)



output delay max = (T_{latch} – T_{launch}) – $T_{CO(max)}$

- Using FPGA minimum T_{CO} requirements ($T_{CO(min)}$ at FPGA boundary)



Next FPGA clock launch edge
Virtual clock latch edge
$T_{CO(min)}$
New data should reach FPGA boundary after this point.
output delay min

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

`set_output_delay` Command

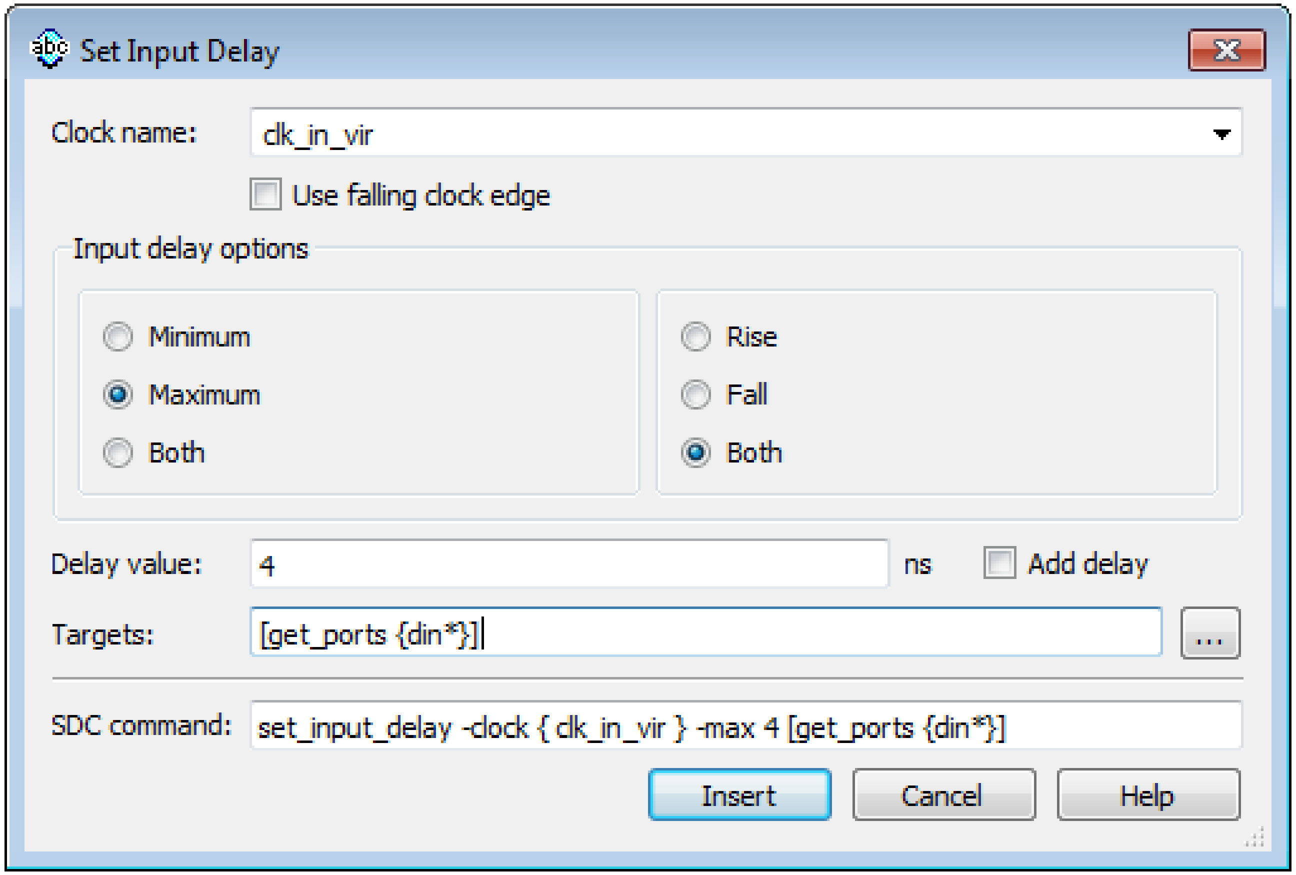
- Constrains output pins by specifying external device timing parameters
- Options

```
-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)


Set Input Delay

Clock name: dlk_in_vir

[ ] Use falling clock edge

Input delay options

( ) Minimum [✓ Maximum] ( ) Both

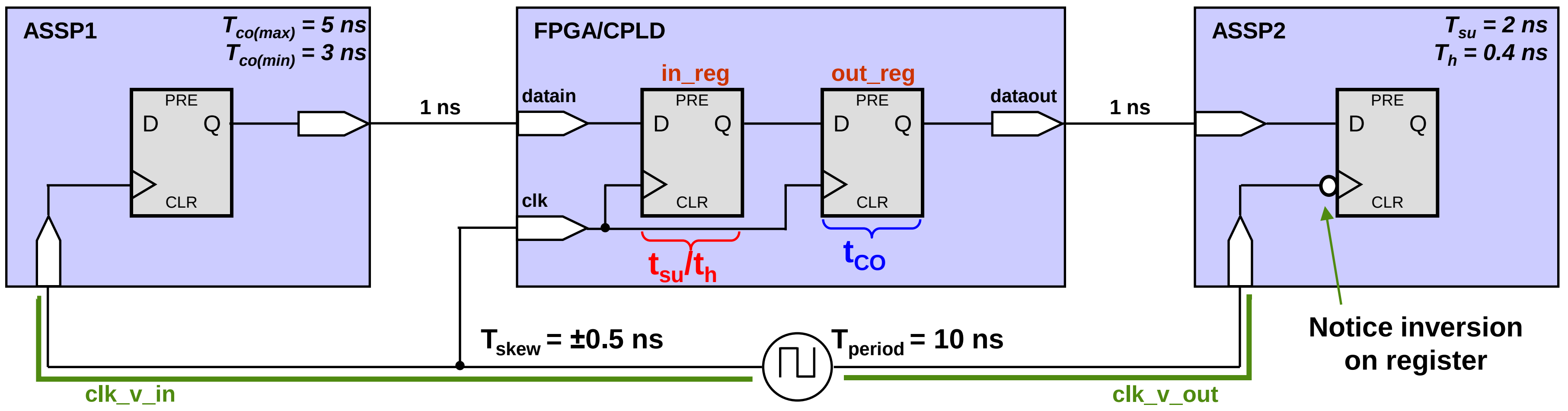
( ) Rise ( ) Fall [✓ Both]

Delay value: 4 ns [ ] Add delay

Targets: [get_ports {din*}]

SDC command:
```
set_input_delay -clock { clk_in_vir } -max 4 [get_ports {din*}]
```

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]
```

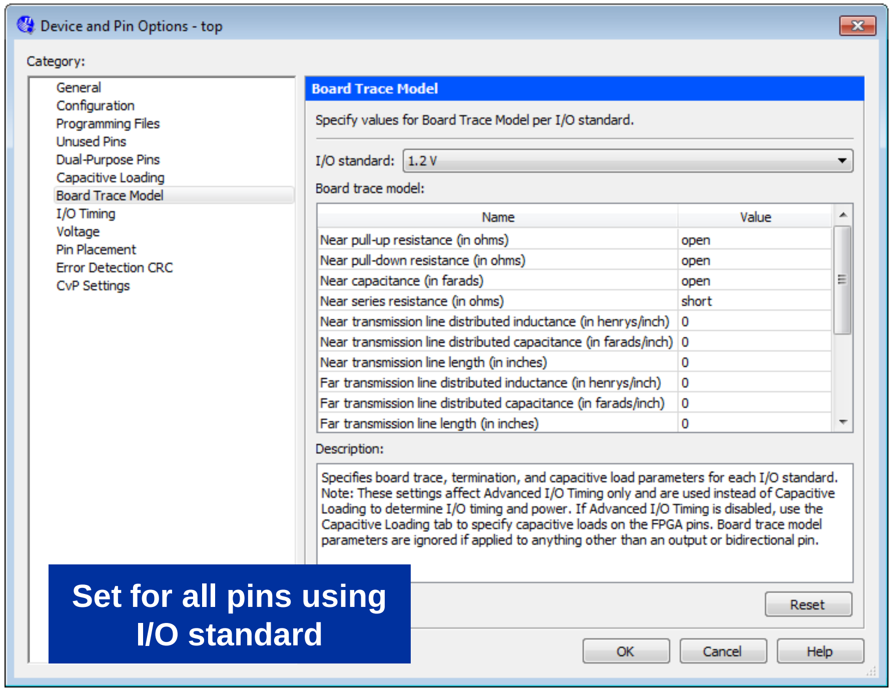
Note: `expr` *in these constraints is used to simply calculate the value of the equation broken down into the 3 parts defined by the input/output delay equations*

Advanced I/O Timing

*See **I/O System Design** online training for details and use*

- Enhances analysis by accurately modeling board-level parameters (65 nm and newer devices only)
  - Use in lieu of or in addition to HSPICE & IBIS modeling
- View signal integrity metrics in Compilation Report (**TimeQuest** folder)

Assignments menu ⇒ Device ⇒ Device & Pin Options



Device and Pin Options - top

Category:

- General
- Configuration
- Programming Files
- Unused Pins
- Dual-Purpose Pins
- Capacitive Loading
- Board Trace Model
- I/O Timing
- Voltage
- Pin Placement
- Error Detection CRC
- CvP Settings

Board Trace Model

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

I/O standard: 1.2 V

Board trace model:

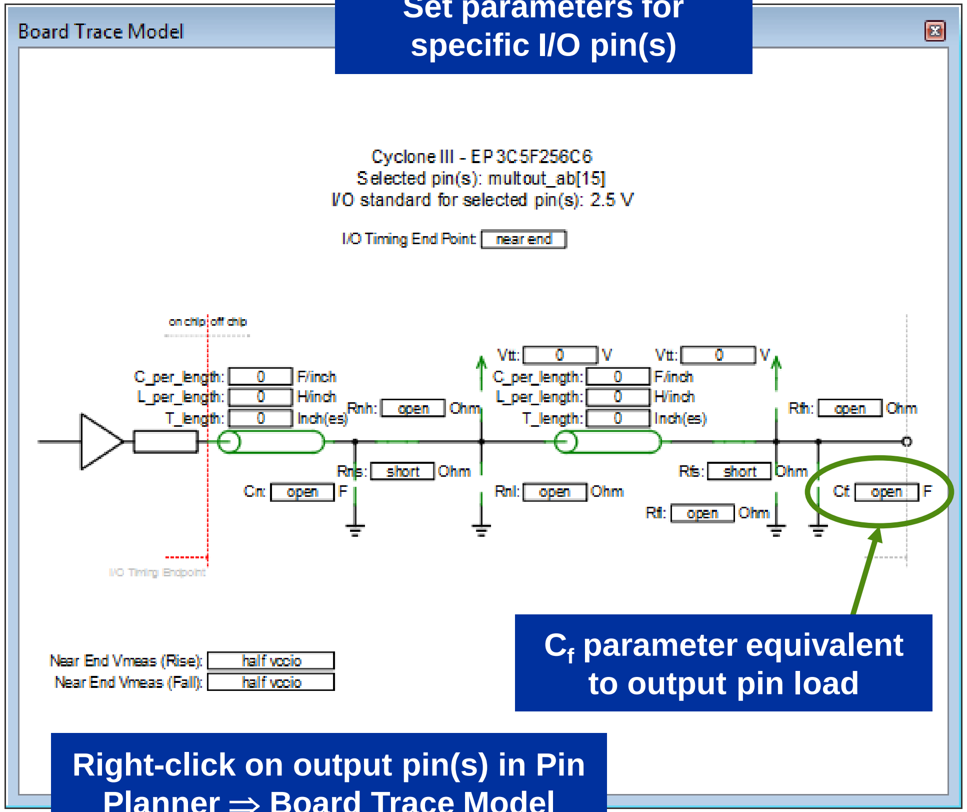
Name	Value
Near pull-up resistance (in ohms)	open
Near pull-down resistance (in ohms)	open
Near capacitance (in farads)	open
Near series resistance (in ohms)	short
Near transmission line distributed inductance (in henrys/inch)	0
Near transmission line distributed capacitance (in farads/inch)	0
Near transmission line length (in inches)	0
Far transmission line distributed inductance (in henrys/inch)	0
Far transmission line distributed capacitance (in farads/inch)	0
Far transmission line length (in inches)	0

Description:

Specifies board trace, termination, and capacitive load parameters for each I/O standard. Note: These settings affect Advanced I/O Timing only and are used instead of Capacitive Loading to determine I/O timing and power. If Advanced I/O Timing is disabled, use the Capacitive Loading tab to specify capacitive loads on the FPGA pins. Board trace model parameters are ignored if applied to anything other than an output or bidirectional pin.

Set for all pins using I/O standard

Set parameters for specific I/O pin(s)



Board Trace Model

Cyclone III - EP3C5F256C6
Selected pin(s): multout_ab[15]
I/O standard for selected pin(s): 2.5 V

I/O Timing End Point: near end

Near End Vmeas (Rise): half vccio
Near End Vmeas (Fall): half vccio

C_f parameter equivalent to output pin load

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

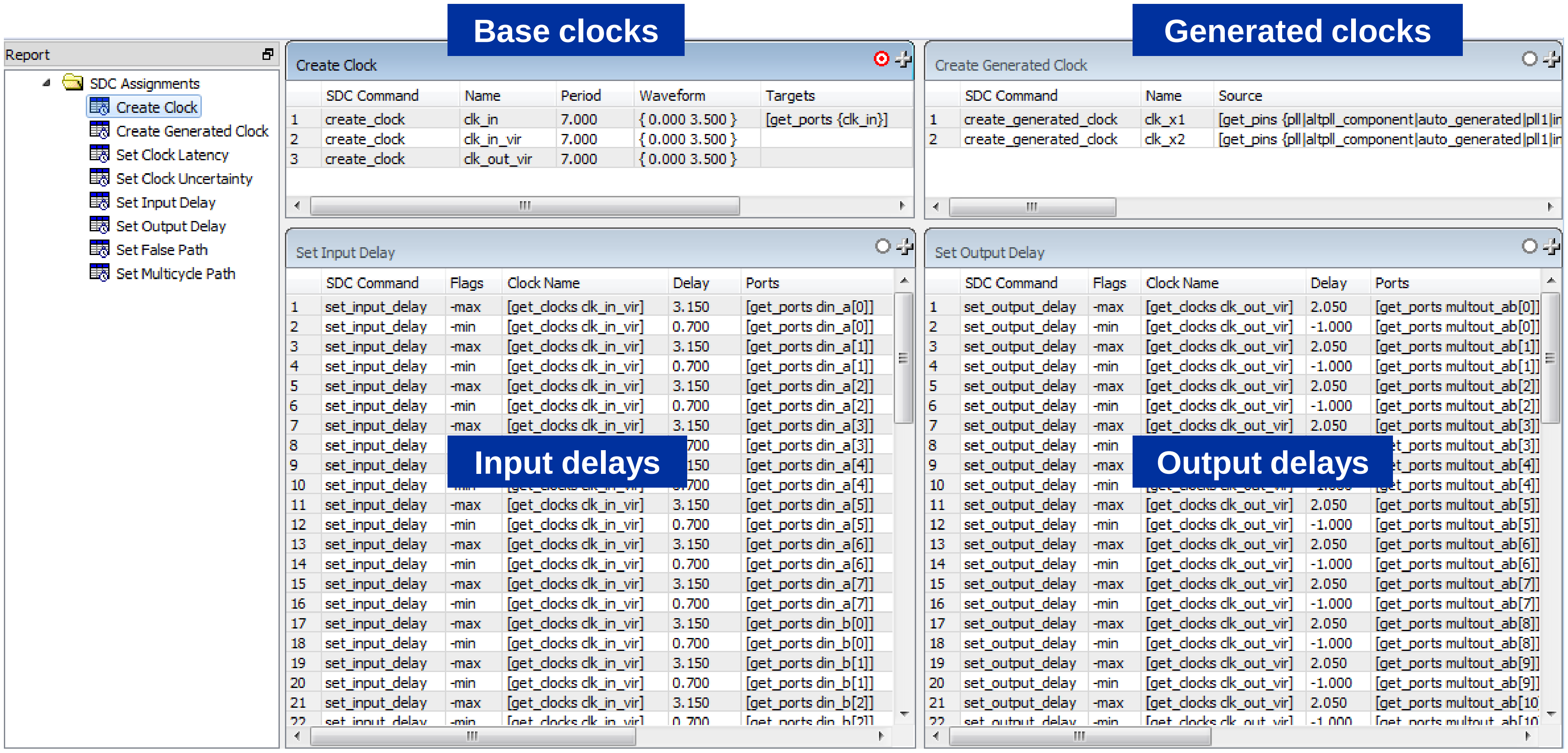
Checking I/O Constraints

- Helpful TimeQuest reports to run to verify constraints

- Report SDC
- Report Unconstrained Paths
- Report Ignored Constraints

Report SDC (`report_sdc`)

- List SDC constraints applied to netlist



Report

- SDC Assignments
  - Create Clock
  - Create Generated Clock
  - Set Clock Latency
  - Set Clock Uncertainty
  - 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	7.000	{ 0.000 3.500 }	[get_ports {clk_in}]
2	create_clock	clk_in_vir	7.000	{ 0.000 3.500 }	
3	create_clock	clk_out_vir	7.000	{ 0.000 3.500 }	

Create Generated Clock

	SDC Command	Name	Source
1	create_generated_clock	clk_x1	[get_pins {pll|altpll_component|auto_generated|pll1|in
2	create_generated_clock	clk_x2	[get_pins {pll|altpll_component|auto_generated|pll1|in

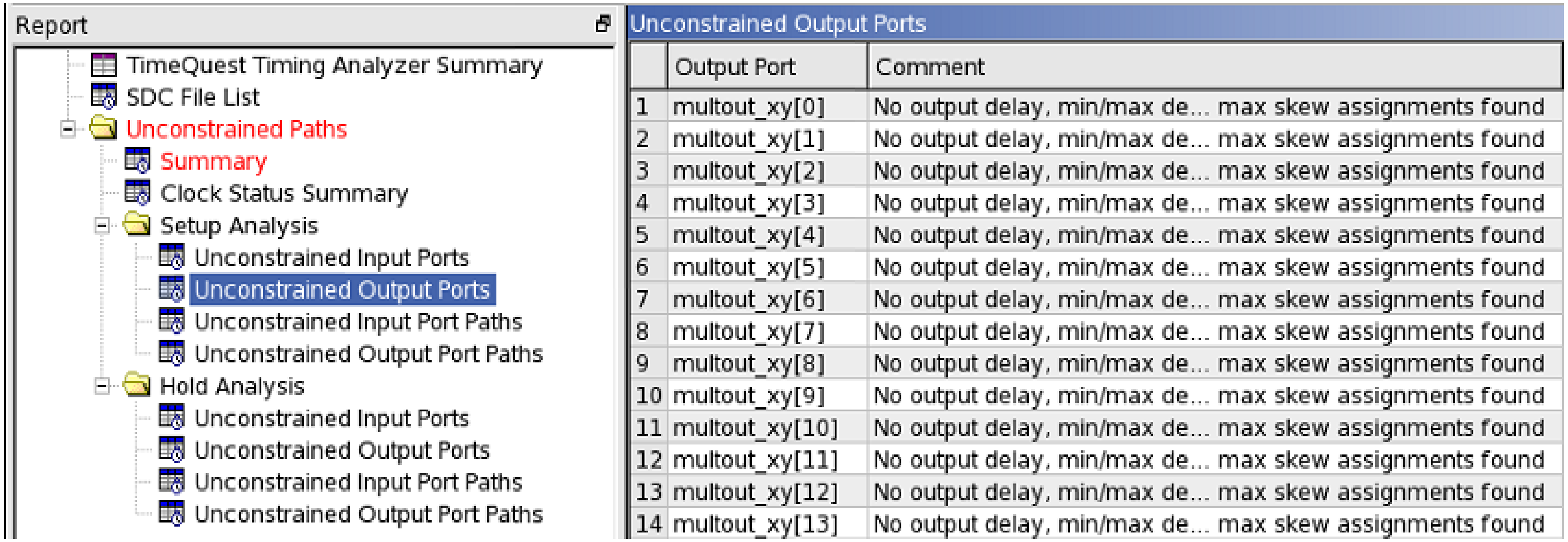
Set Input Delay

	SDC Command	Flags	Clock Name	Delay	Ports
1	set_input_delay	-max	[get_clocks clk_in_vir]	3.150	[get_ports din_a[0]]
2	set_input_delay	-min	[get_clocks clk_in_vir]	0.700	[get_ports din_a[0]]
3	set_input_delay	-max	[get_clocks clk_in_vir]	3.150	[get_ports din_a[1]]
4	set_input_delay	-min	[get_clocks clk_in_vir]	0.700	[get_ports din_a[1]]
5	set_input_delay	-max	[get_clocks clk_in_vir]	3.150	[get_ports din_a[2]]
6	set_input_delay	-min	[get_clocks clk_in_vir]	0.700	[get_ports din_a[2]]
7	set_input_delay	-max	[get_clocks clk_in_vir]	3.150	[get_ports din_a[3]]
8	set_input_delay	[illegible]	[illegible]	[illegible]	[get_ports din_a[3]]
9	set_input_delay	[illegible]	[illegible]	[illegible]	[get_ports din_a[4]]
10	set_input_delay	[illegible]	[illegible]	[illegible]	[get_ports din_a[4]]
11	set_input_delay	-max	[get_clocks clk_in_vir]	3.150	[get_ports din_a[5]]
12	set_input_delay	-min	[get_clocks clk_in_vir]	0.700	[get_ports din_a[5]]
13	set_input_delay	-max	[get_clocks clk_in_vir]	3.150	[get_ports din_a[6]]
14	set_input_delay	-min	[get_clocks clk_in_vir]	0.700	[get_ports din_a[6]]
15	set_input_delay	-max	[get_clocks clk_in_vir]	3.150	[get_ports din_a[7]]
16	set_input_delay	-min	[get_clocks clk_in_vir]	0.700	[get_ports din_a[7]]
17	set_input_delay	-max	[get_clocks clk_in_vir]	3.150	[get_ports din_b[0]]
18	set_input_delay	-min	[get_clocks clk_in_vir]	0.700	[get_ports din_b[0]]
19	set_input_delay	-max	[get_clocks clk_in_vir]	3.150	[get_ports din_b[1]]
20	set_input_delay	-min	[get_clocks clk_in_vir]	0.700	[get_ports din_b[1]]
21	set_input_delay	-max	[get_clocks clk_in_vir]	3.150	[get_ports din_b[2]]
22	set_input_delay	-min	[get_clocks clk_in_vir]	0.700	[get_ports din_b[2]]

Set Output Delay

	SDC Command	Flags	Clock Name	Delay	Ports
1	set_output_delay	-max	[get_clocks clk_out_vir]	2.050	[get_ports multout_ab[0]]
2	set_output_delay	-min	[get_clocks clk_out_vir]	-1.000	[get_ports multout_ab[0]]
3	set_output_delay	-max	[get_clocks clk_out_vir]	2.050	[get_ports multout_ab[1]]
4	set_output_delay	-min	[get_clocks clk_out_vir]	-1.000	[get_ports multout_ab[1]]
5	set_output_delay	-max	[get_clocks clk_out_vir]	2.050	[get_ports multout_ab[2]]
6	set_output_delay	-min	[get_clocks clk_out_vir]	-1.000	[get_ports multout_ab[2]]
7	set_output_delay	-max	[get_clocks clk_out_vir]	2.050	[get_ports multout_ab[3]]
8	set_output_delay	-min	[illegible]	[illegible]	[illegible] multout_ab[3]]
9	set_output_delay	-max	[illegible]	[illegible]	[illegible] multout_ab[4]]
10	set_output_delay	-min	[illegible]	[illegible]	[illegible] multout_ab[4]]
11	set_output_delay	-max	[get_clocks clk_out_vir]	2.050	[get_ports multout_ab[5]]
12	set_output_delay	-min	[get_clocks clk_out_vir]	-1.000	[get_ports multout_ab[5]]
13	set_output_delay	-max	[get_clocks clk_out_vir]	2.050	[get_ports multout_ab[6]]
14	set_output_delay	-min	[get_clocks clk_out_vir]	-1.000	[get_ports multout_ab[6]]
15	set_output_delay	-max	[get_clocks clk_out_vir]	2.050	[get_ports multout_ab[7]]
16	set_output_delay	-min	[get_clocks clk_out_vir]	-1.000	[get_ports multout_ab[7]]
17	set_output_delay	-max	[get_clocks clk_out_vir]	2.050	[get_ports multout_ab[8]]
18	set_output_delay	-min	[get_clocks clk_out_vir]	-1.000	[get_ports multout_ab[8]]
19	set_output_delay	-max	[get_clocks clk_out_vir]	2.050	[get_ports multout_ab[9]]
20	set_output_delay	-min	[get_clocks clk_out_vir]	-1.000	[get_ports multout_ab[9]]
21	set_output_delay	-max	[get_clocks clk_out_vir]	2.050	[get_ports multout_ab[10
22	set_output_delay	-min	[get_clocks clk_out_vir]	-1.000	[get_ports multout_ab[10

Report Unconstrained Paths (`report_ucp`)


Report
TimeQuest Timing Analyzer Summary
SDC File List
Unconstrained Paths
Summary
Clock Status Summary
Setup Analysis
Unconstrained Input Ports
Unconstrained Output Ports
Unconstrained Input Port Paths
Unconstrained Output Port Paths
Hold Analysis
Unconstrained Input Ports
Unconstrained Output Ports
Unconstrained Input Port Paths
Unconstrained Output Port Paths

Unconstrained Output Ports

	Output Port	Comment
1	multout_xy[0]	No output delay, min/max de... max skew assignments found
2	multout_xy[1]	No output delay, min/max de... max skew assignments found
3	multout_xy[2]	No output delay, min/max de... max skew assignments found
4	multout_xy[3]	No output delay, min/max de... max skew assignments found
5	multout_xy[4]	No output delay, min/max de... max skew assignments found
6	multout_xy[5]	No output delay, min/max de... max skew assignments found
7	multout_xy[6]	No output delay, min/max de... max skew assignments found
8	multout_xy[7]	No output delay, min/max de... max skew assignments found
9	multout_xy[8]	No output delay, min/max de... max skew assignments found
10	multout_xy[9]	No output delay, min/max de... max skew assignments found
11	multout_xy[10]	No output delay, min/max de... max skew assignments found
12	multout_xy[11]	No output delay, min/max de... max skew assignments found
13	multout_xy[12]	No output delay, min/max de... max skew assignments found
14	multout_xy[13]	No output delay, min/max de... max skew assignments 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

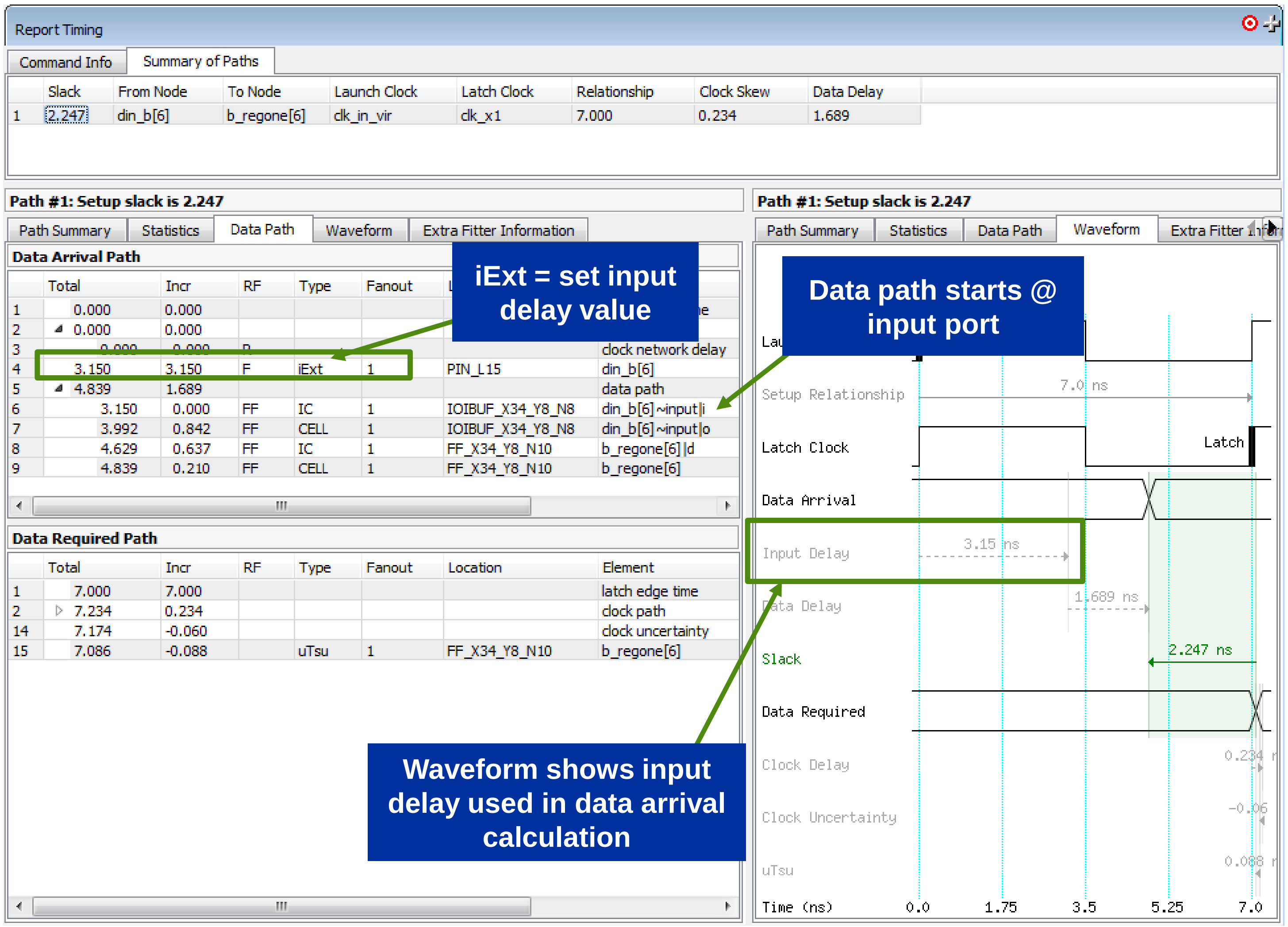
Verifying I/O Timing

- Use setup & hold summary reports (same as for clock analysis as seen earlier)
- Use **Report Timing** task for detailed analysis

- I/O timing reported based on clock domain of destination node
- Input analysis
  - Input ports are start of data arrival path (**From Node** column)
  - Input delay values appear in data arrival path
- Output analysis
  - Output ports are end of data arrival path (**To Node** column)
  - Output delay values appear in data required path

Example Input Setup Report

set_input_delay –max –clock clk_in_vir 3.15 [get_ports din*]



Report Timing

Command Info | Summary of Paths

	Slack	From Node	To Node	Launch Clock	Latch Clock	Relationship	Clock Skew	Data Delay
1	2.247	din_b[6]	b_regone[6]	clk_in_vir	clk_x1	7.000	0.234	1.689

Path #1: Setup slack is 2.247

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

Data Arrival Path

	Total	Incr	RF	Type	Fanout	L	Element
1	0.000	0.000					e
2	◢ 0.000	0.000					
3	0.000	0.000	R				clock network delay
4	3.150	3.150	F	iExt	1	PIN_L15	din_b[6]
5	◢ 4.839	1.689					data path
6	3.150	0.000	FF	IC	1	IOIBUF_X34_Y8_N8	din_b[6]~input|i
7	3.992	0.842	FF	CELL	1	IOIBUF_X34_Y8_N8	din_b[6]~input|o
8	4.629	0.637	FF	IC	1	FF_X34_Y8_N10	b_regone[6]|d
9	4.839	0.210	FF	CELL	1	FF_X34_Y8_N10	b_regone[6]

Data Required Path

	Total	Incr	RF	Type	Fanout	Location	Element
1	7.000	7.000					latch edge time
2	▷ 7.234	0.234					clock path
14	7.174	-0.060					clock uncertainty
15	7.086	-0.088		uTsu	1	FF_X34_Y8_N10	b_regone[6]

Path #1: Setup slack is 2.247

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

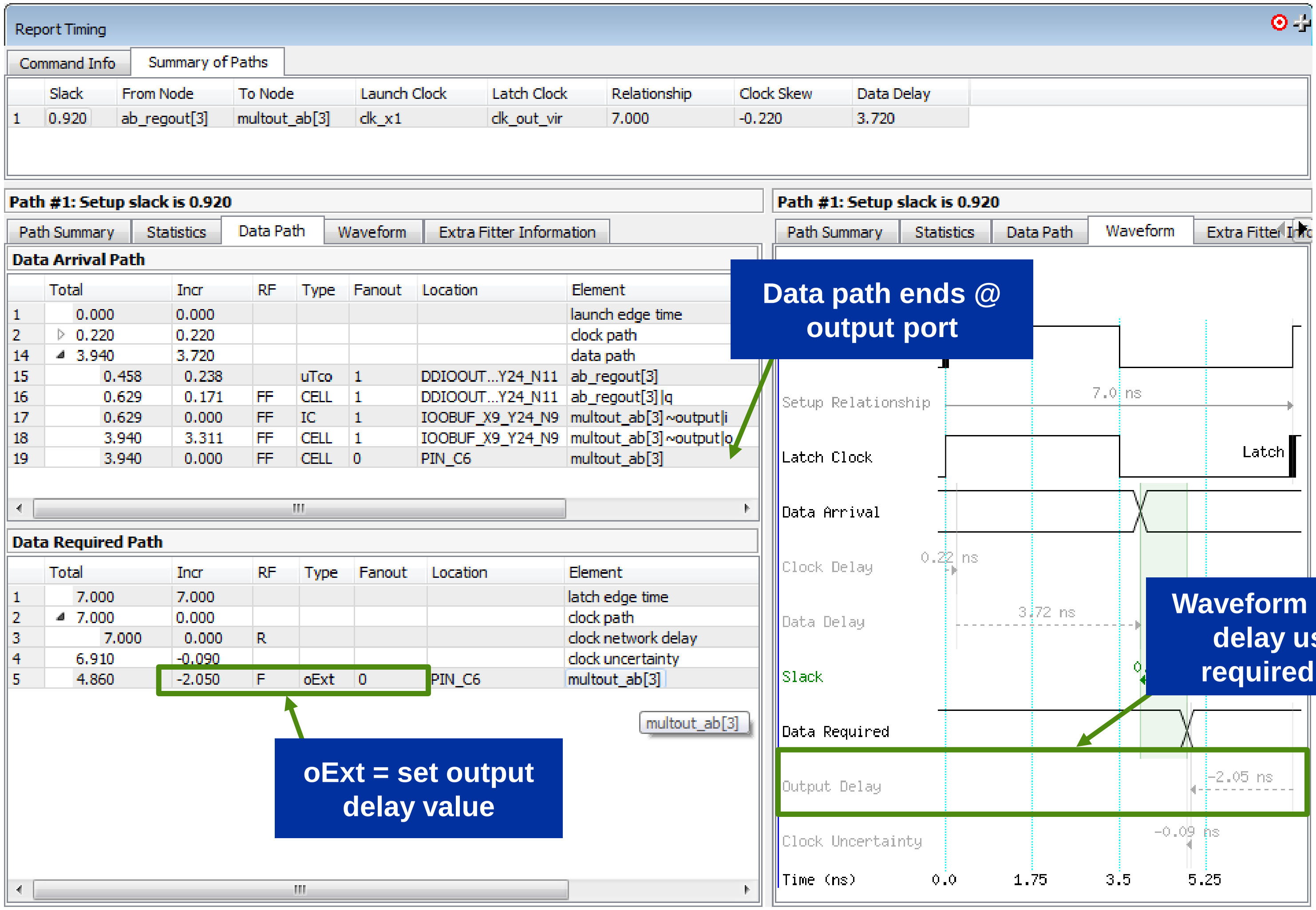
iExt = set input delay value

Data path starts @ input port

Waveform shows input delay used in data arrival calculation

Example Output Setup Report

set_output_delay –max –clock clk_out_vir 2.05 [get_ports multout_ab*]



Report Timing

Command Info | Summary of Paths

	Slack	From Node	To Node	Launch Clock	Latch Clock	Relationship	Clock Skew	Data Delay
1	0.920	ab_regout[3]	multout_ab[3]	clk_x1	clk_out_vir	7.000	-0.220	3.720

Path #1: Setup slack is 0.920

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	▷ 0.220	0.220					clock path
14	◢ 3.940	3.720					data path
15	0.458	0.238		uTco	1	DDIOOUT...Y24_N11	ab_regout[3]
16	0.629	0.171	FF	CELL	1	DDIOOUT...Y24_N11	ab_regout[3]|q
17	0.629	0.000	FF	IC	1	IOOBUF_X9_Y24_N9	multout_ab[3]~output|i
18	3.940	3.311	FF	CELL	1	IOOBUF_X9_Y24_N9	multout_ab[3]~output|o
19	3.940	0.000	FF	CELL	0	PIN_C6	multout_ab[3]

Data Required Path

	Total	Incr	RF	Type	Fanout	Location	Element
1	7.000	7.000					latch edge time
2	◢ 7.000	0.000					clock path
3	7.000	0.000	R				clock network delay
4	6.910	-0.090					clock uncertainty
5	4.860	-2.050	F	oExt	0	PIN_C6	multout_ab[3]

multout_ab[3]

Path #1: Setup slack is 0.920

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

Data path ends @ output port

Waveform shows output delay used in data required calculation

oExt = set output delay value

Synchronous I/O Timing Summary

	Using external parameters	Using FPGA requirements[1]
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) = $T_{clk2(max)} - T_{clk1ext(min)}$	**$-T_{CO(min)}$**

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

$T = (T_{latch} – T_{launch})$

Please go to Exercise 3

SDC Timing Constraints

- Clocks
- I/O
- Asynchronous paths
- False paths
- Multicycle paths

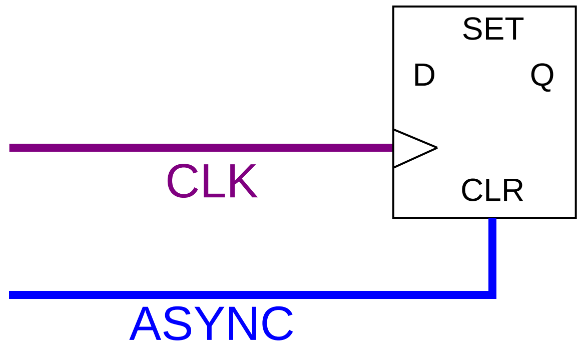
Asynchronous Paths

- 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

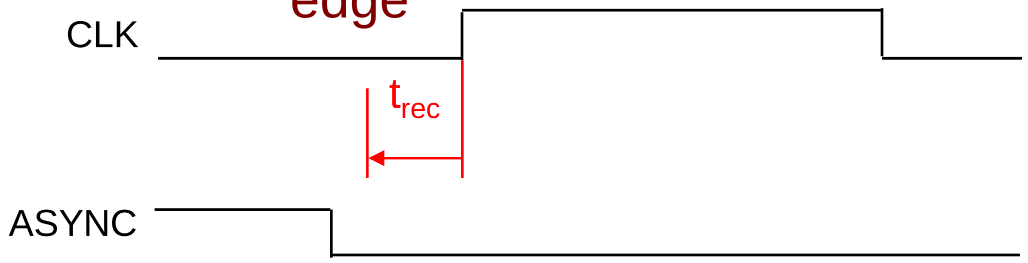
TimeQuest TA & Asynchronous Ports

- 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

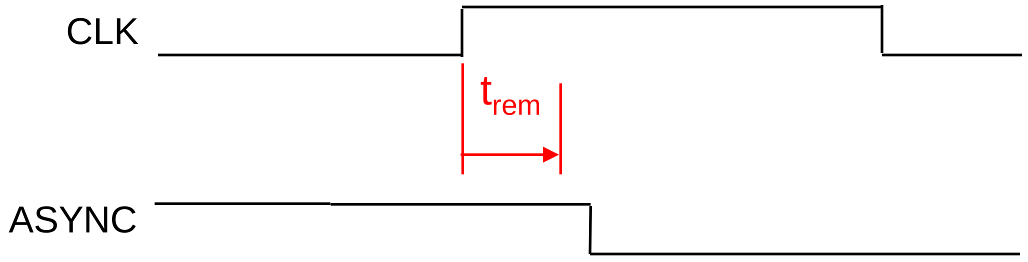
Recovery & Removal (Review)



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



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

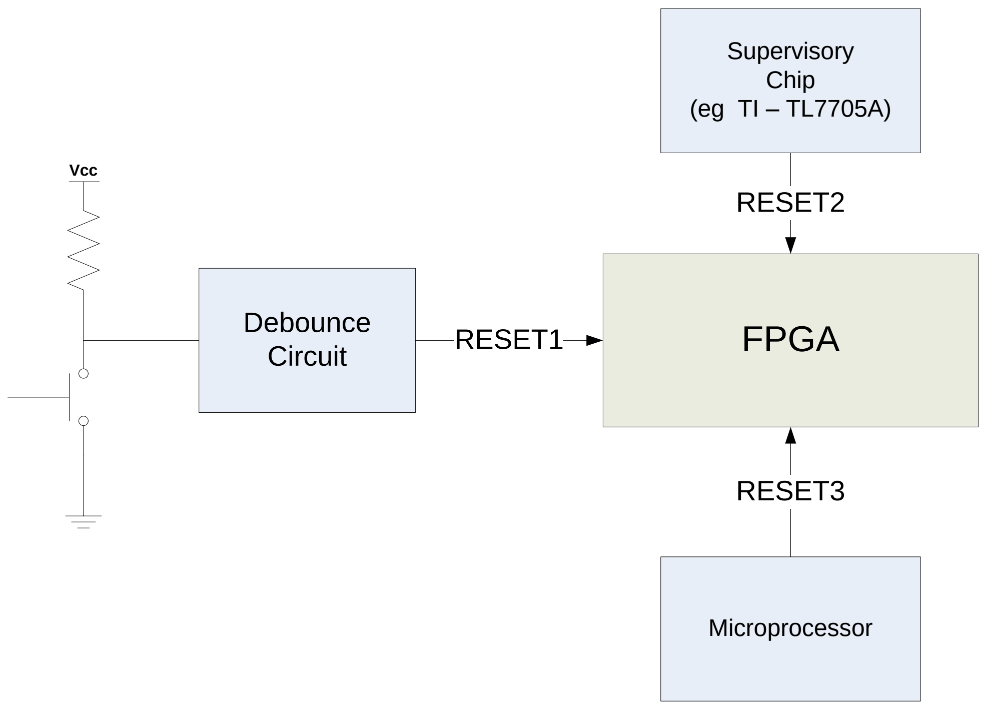


Types of Asynchronous Paths

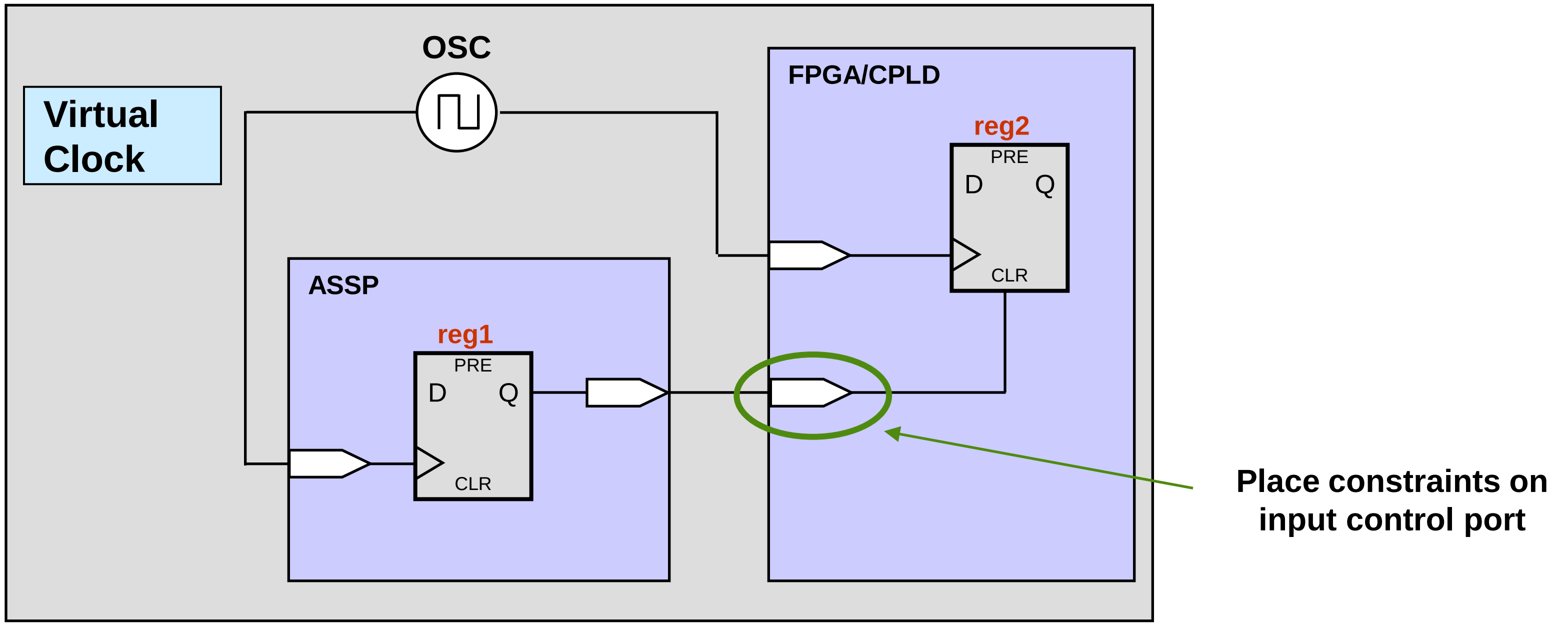
- Externally registered
- Internally registered

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

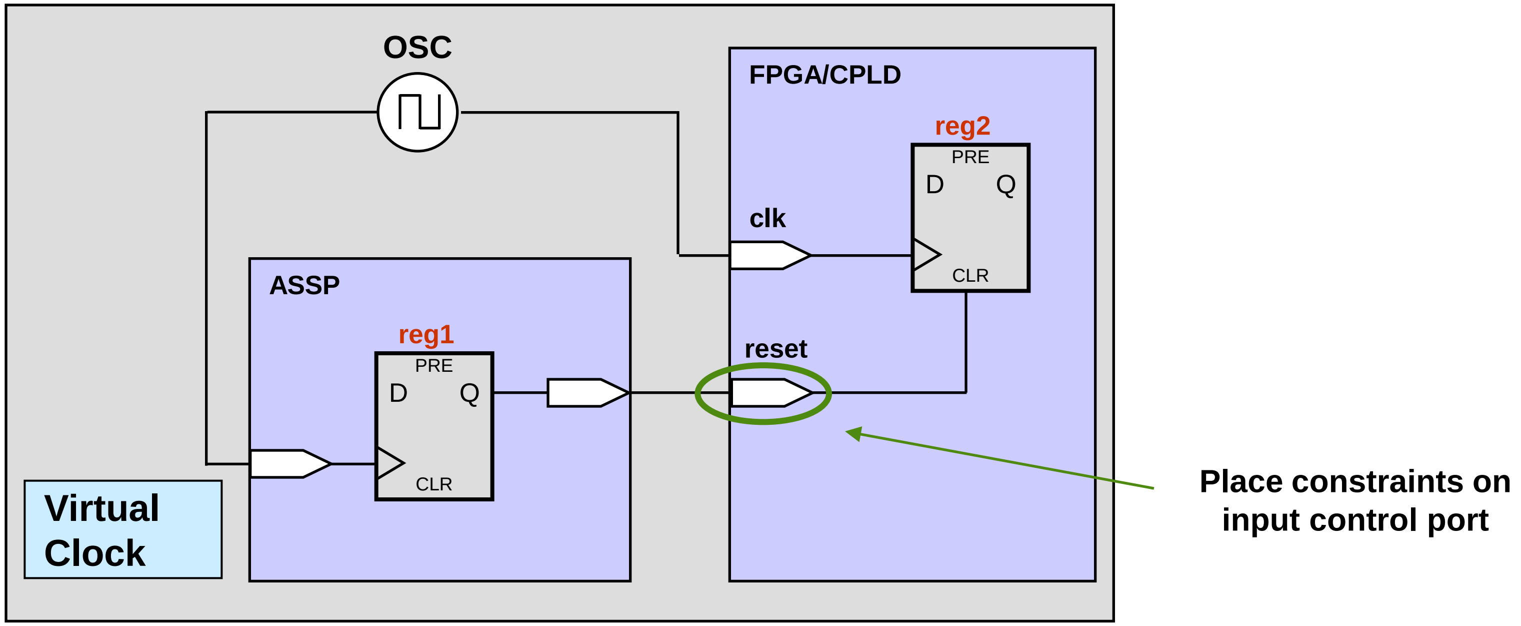


Externally Registered (cont.)



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

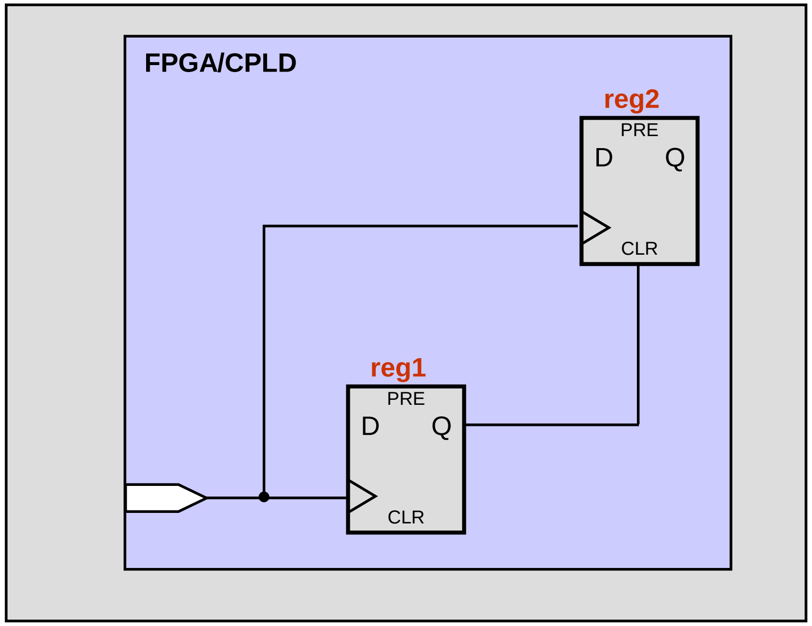
Externally Registered Example



```
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

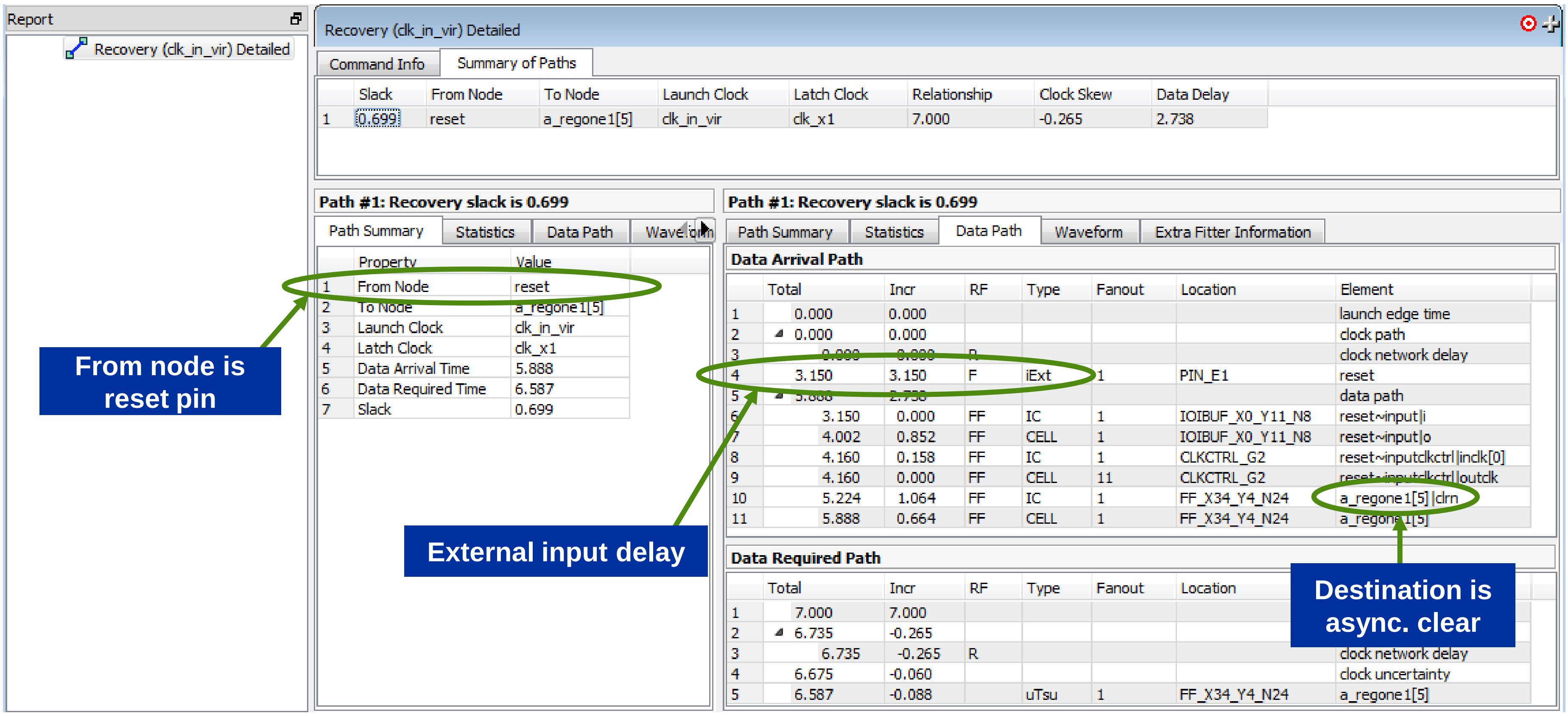
Reporting Asynchronous Control Paths

- 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_vir -recovery -npaths 1 \
        -detail path_only -panel_name {Recovery (clk_in_vir) Detailed}
```



Recovery (clk_in_vir) Detailed

	Slack	From Node	To Node	Launch Clock	Latch Clock	Relationship	Clock Skew	Data Delay
1	0.699	reset	a_regone1[5]	clk_in_vir	clk_x1	7.000	-0.265	2.738

Path #1: Recovery slack is 0.699

	Property	Value
1	From Node	reset
2	To Node	a_regone1[5]
3	Launch Clock	clk_in_vir
4	Latch Clock	clk_x1
5	Data Arrival Time	5.888
6	Data Required Time	6.587
7	Slack	0.699

Path #1: Recovery slack is 0.699

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
3	0.000	0.000	R				clock network delay
4	3.150	3.150	F	iExt	1	PIN_E1	reset
5	5.888	2.738					data path
6	3.150	0.000	FF	IC	1	IOIBUF_X0_Y11_N8	reset~input|i
7	4.002	0.852	FF	CELL	1	IOIBUF_X0_Y11_N8	reset~input|o
8	4.160	0.158	FF	IC	1	CLKCTRL_G2	reset~inputclkctrl|inclk[0]
9	4.160	0.000	FF	CELL	11	CLKCTRL_G2	reset~inputclkctrl|outclk
10	5.224	1.064	FF	IC	1	FF_X34_Y4_N24	a_regone1[5]|clrn
11	5.888	0.664	FF	CELL	1	FF_X34_Y4_N24	a_regone1[5]

Data Required Path

	Total	Incr	RF	Type	Fanout	Location	Element
1	7.000	7.000					
2	6.735	-0.265					
3	6.735	-0.265	R				clock network delay
4	6.675	-0.060					clock uncertainty
5	6.587	-0.088		uTsu	1	FF_X34_Y4_N24	a_regone1[5]

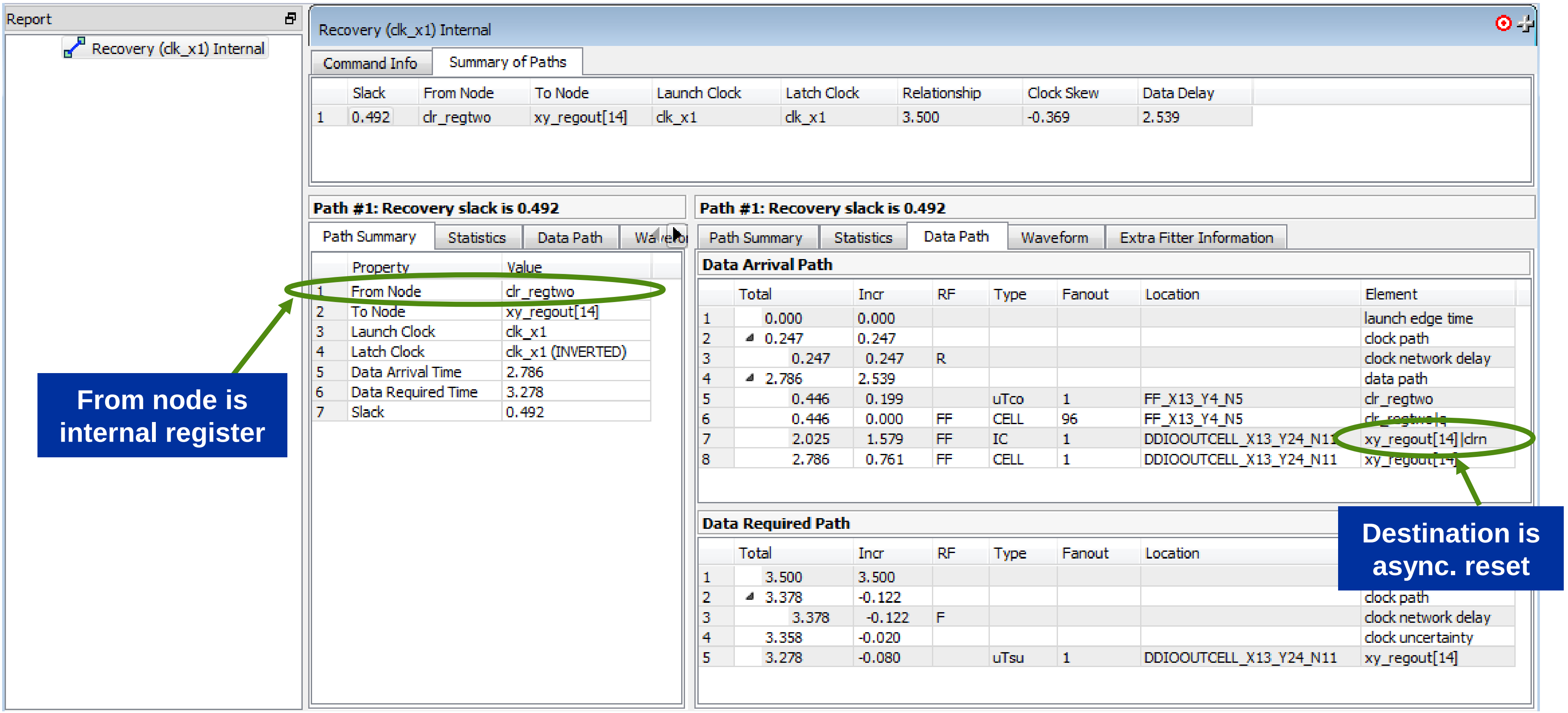
From node is reset pin

External input delay

Destination is async. clear

Example Recovery Report (Int. Registered)

```
report_timing -from_clock clk_x1 -recovery -npaths 1 \
       -detail path_only -panel_name {Recovery (clk_x1) Internal}
```



Recovery (clk_x1) Internal

	Slack	From Node	To Node	Launch Clock	Latch Clock	Relationship	Clock Skew	Data Delay
1	0.492	clr_regtwo	xy_regout[14]	clk_x1	clk_x1	3.500	-0.369	2.539

Path #1: Recovery slack is 0.492

	Property	Value
1	From Node	clr_regtwo
2	To Node	xy_regout[14]
3	Launch Clock	clk_x1
4	Latch Clock	clk_x1 (INVERTED)
5	Data Arrival Time	2.786
6	Data Required Time	3.278
7	Slack	0.492

Path #1: Recovery slack is 0.492

Data Arrival Path

	Total	Incr	RF	Type	Fanout	Location	Element
1	0.000	0.000					launch edge time
2	◢ 0.247	0.247					clock path
3	0.247	0.247	R				clock network delay
4	◢ 2.786	2.539					data path
5	0.446	0.199		uTco	1	FF_X13_Y4_N5	clr_regtwo
6	0.446	0.000	FF	CELL	96	FF_X13_Y4_N5	clr_regtwo|q
7	2.025	1.579	FF	IC	1	DDIOOUTCELL_X13_Y24_N11	xy_regout[14]|clrn
8	2.786	0.761	FF	CELL	1	DDIOOUTCELL_X13_Y24_N11	xy_regout[14]

Data Required Path

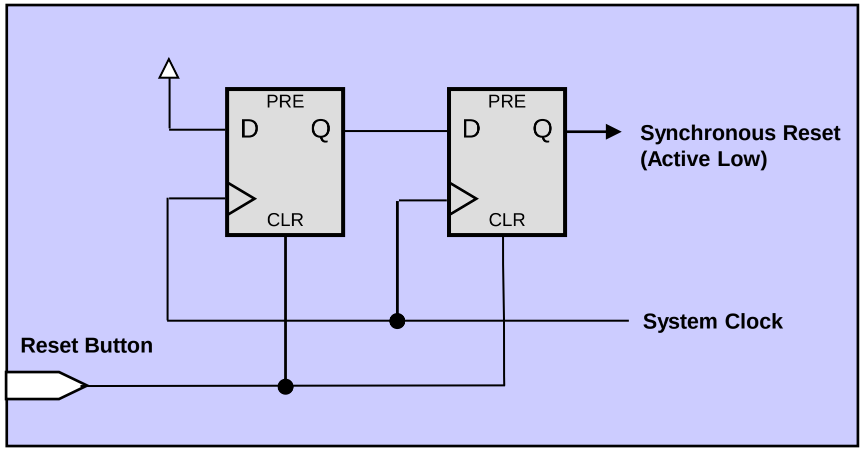
	Total	Incr	RF	Type	Fanout	Location	Element
1	3.500	3.500					
2	◢ 3.378	-0.122					clock path
3	3.378	-0.122	F				clock network delay
4	3.358	-0.020					clock uncertainty
5	3.278	-0.080		uTsu	1	DDIOOUTCELL_X13_Y24_N11	xy_regout[14]

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

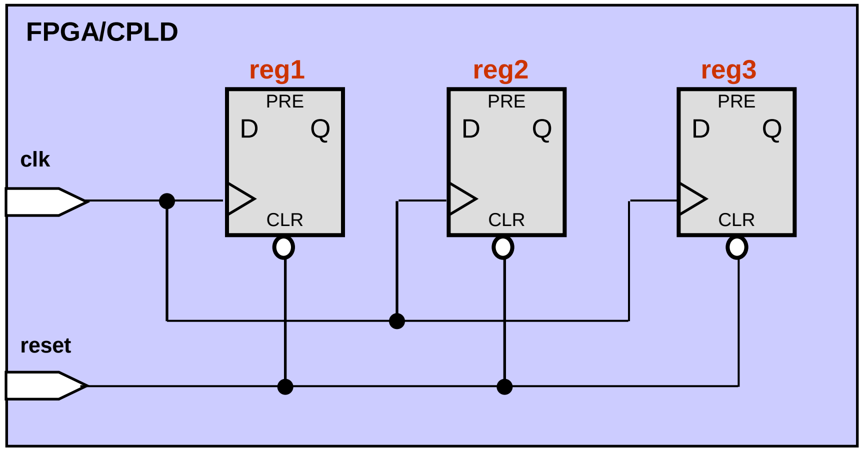
OR

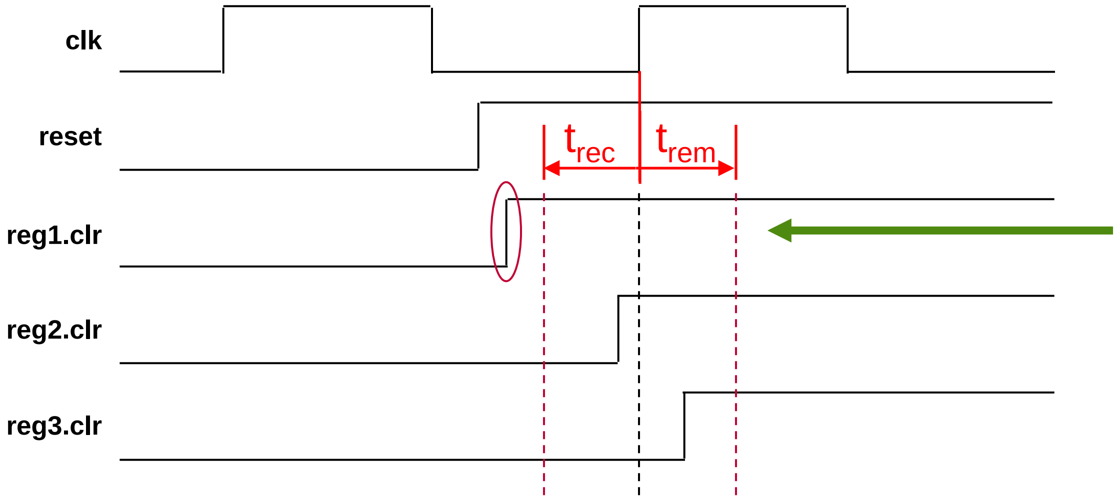
  - Use false path and remove from timing analysis



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 Exceptions

- False paths
  - Perform no analysis on path
  - Sometimes referred to as “cut paths”
- Multicycle paths
  - Adjust clock relationship of path based on specified number of launch-to-latch edges
- Max/min delay paths
  - Apply arbitrary timing relationship to path using `set_max[min]_delay`
  - No further discussion in class
    - See appendix and Advanced Timing Analysis with TimeQuest course

SDC Timing Constraints

- Clocks
- I/O
- Asynchronous paths
- False paths
- Multicycle paths

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

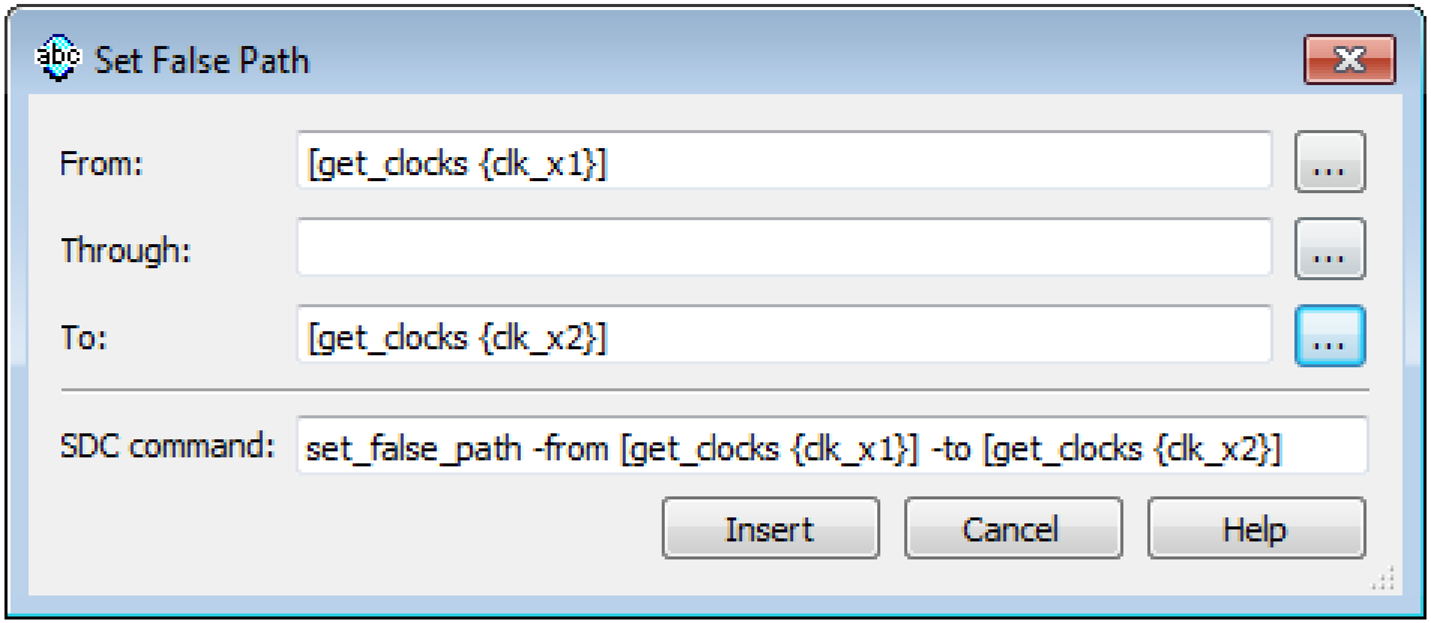
- Indicates paths that should be ignored during fitting and timing analysis
- Options

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

`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**, not **q** output
  - 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`: indicates that no analysis should be performed on the targeted path at the rising or falling edge transition of the path's launch clock; *not in GUI*
- `-rise_to` & `-fall_to`: indicates that no analysis should be performed on the targeted path at the rising or falling edge transition of the path's latch clock; *not in GUI*
- `-setup` & `-hold`: indicates that no setup/recovery or hold/removal analysis should be performed on the targeted path; *not in GUI*

Set False Path (GUI)



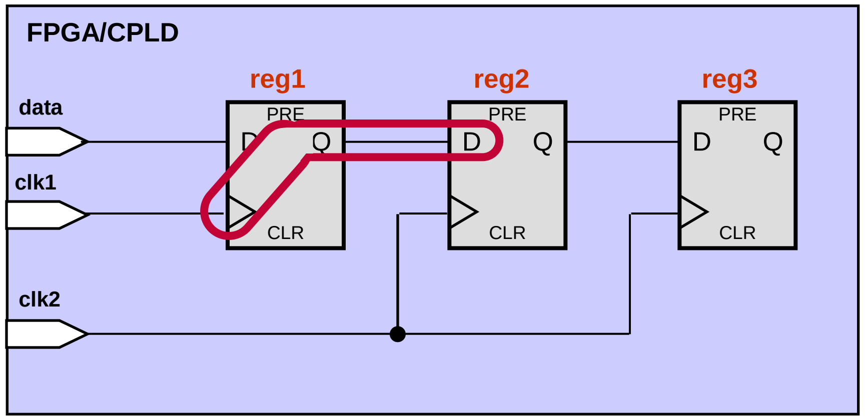
From: [get_clocks {clk_x1}]

Through:

To: [get_clocks {clk_x2}]

SDC command:
```
set_false_path -from [get_clocks {clk_x1}] -to [get_clocks {clk_x2}]
```

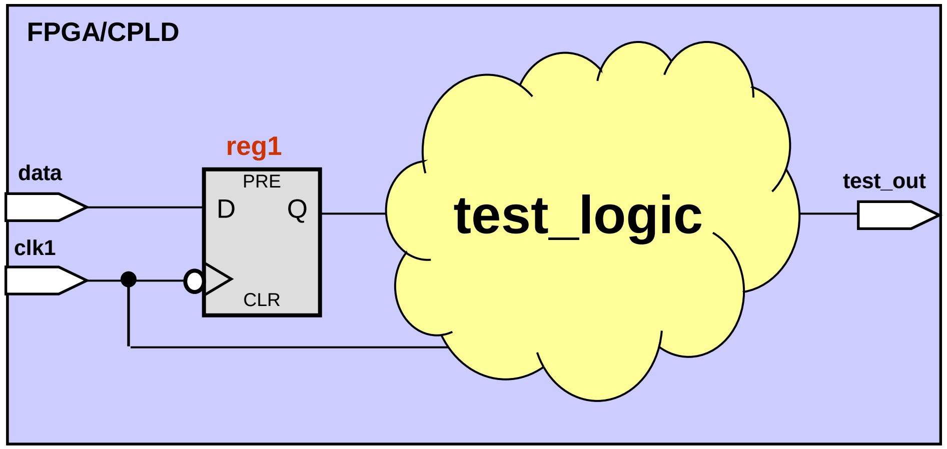
False Path Example 1



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



Cutting analysis of inserted test logic

```
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>]…
```

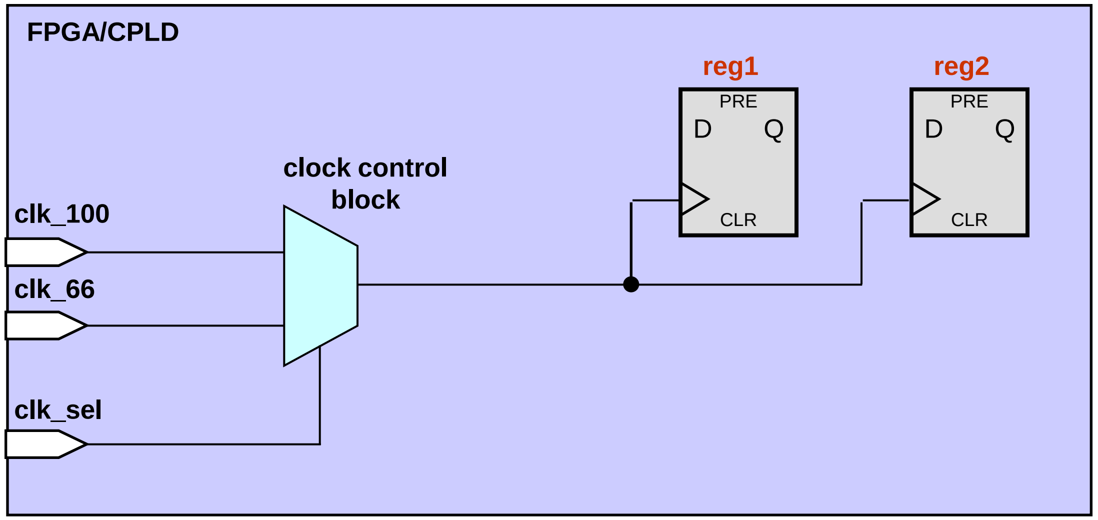
`set_clock_groups` Notes

- `-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:

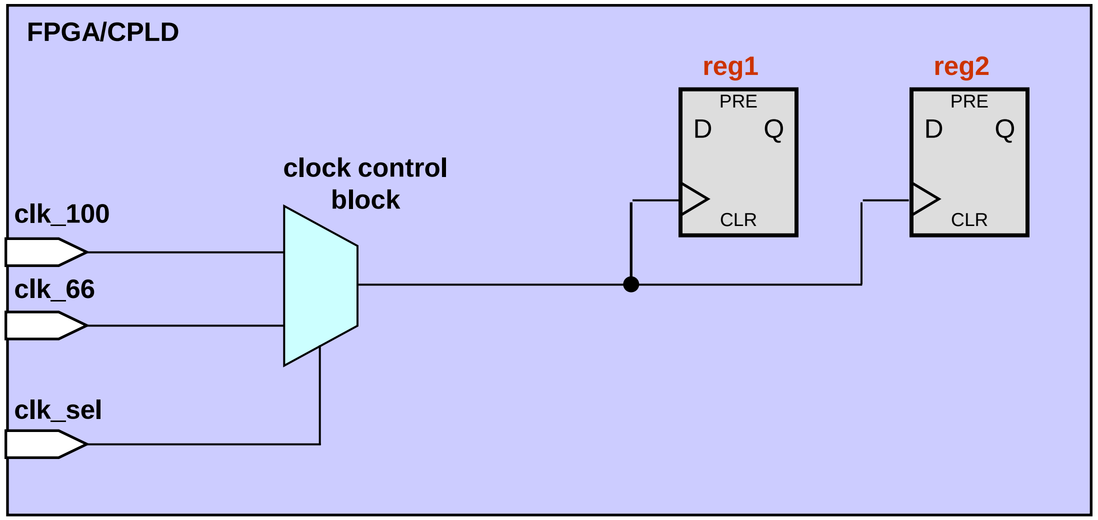
- Need at least one of the two arguments (`-asynchronous` or `-exclusive`)
- TimeQuest Timing Analyzer treats both options as if they were the same
- With one `-group` argument, TimeQuest Timing Analyzer cut analysis of ALL paths to that group of clocks.

Clock Mux Example 1



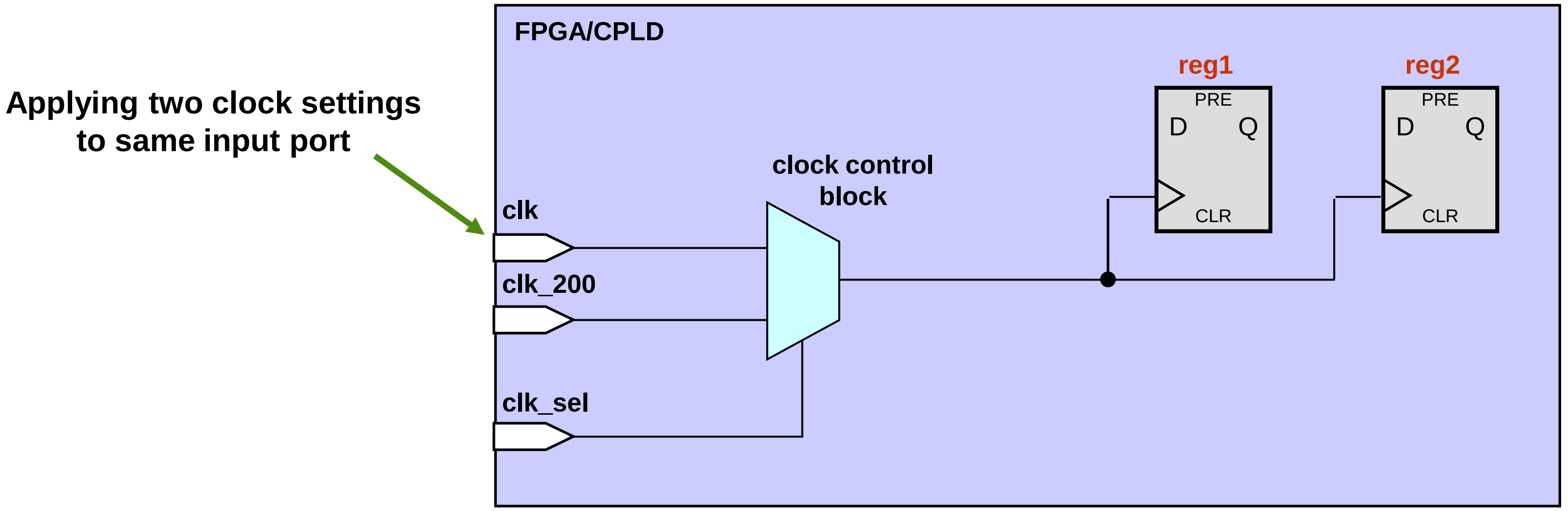
```
create_clock -period 10.0 [get_ports clk_100]
create_clock -period 15.0 [get_ports clk_66]

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)



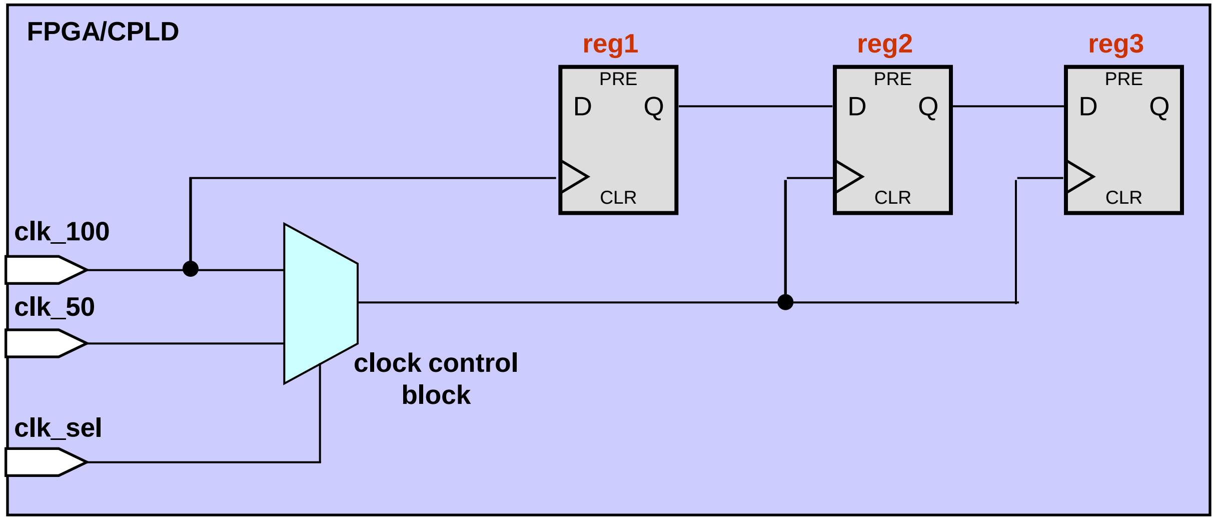
```
create_clock -period 10.0 [get_ports clk_100]
create_clock -period 15.0 [get_ports clk_66]

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
```

Clock Mux Example 2



```
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



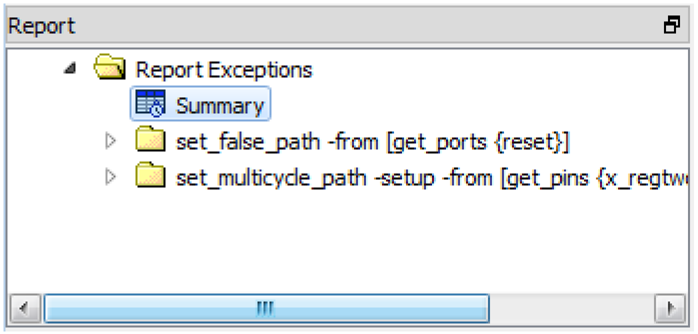
```
create_clock -period 10.0 [get_ports clk_100]
create_clock -period 20.0 [get_ports clk_50]

create_generated_clock -name clkmux_100 -source clk_100 \
         -multiply_by 1 [get_pins clkmux|clkout]
create_generated_clock -name clkmux_50 -source clk_50 \
         -multiply_by 1 [get_pins clkmux|clkout] -add

set_clock_groups -exclusive -group {clkmux_100} -group {clkmux_50}

# 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
```

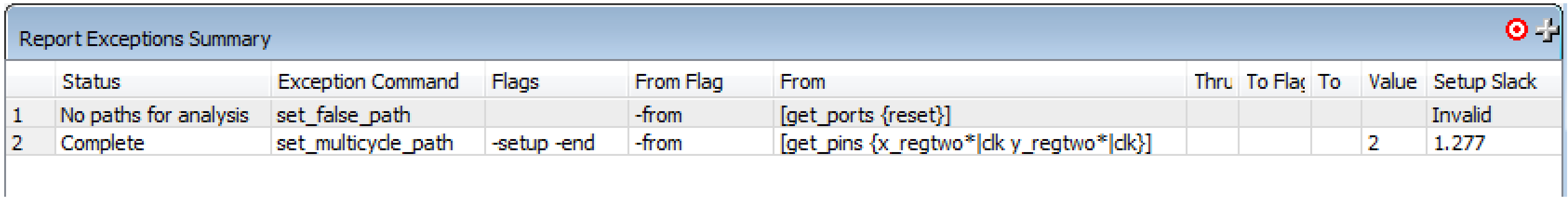
Verifying False Paths & Groups

- False paths
  - Create timing exceptions report
    - `report_exceptions`
    - **Tasks** pane or **Reports** menu: **Report Exceptions** *(next slide)*
- Clock groups
  - Check clock transfers to ensure no paths are returned
    - `report_clock_transfers`
    - **Tasks** pane or **Reports** menu: **Report Clock Transfers**

Report Exceptions

- Provide information specifically about timing exceptions
  - `report_exceptions`
  - From **Tasks** pane or **Report** menu (under **Custom Reports**)
- All min/max delays, false paths, and multicycle paths *(discussed next)*


Report
- Report Exceptions
  - Summary
  - set_false_path -from [get_ports {reset}]
  - set_multicycle_path -setup -from [get_pins {x_regtw


Report Exceptions Summary

	Status	Exception Command	Flags	From Flag	From	Thru	To Flag	To	Value	Setup Slack
1	No paths for analysis	set_false_path		-from	[get_ports {reset}]					Invalid
2	Complete	set_multicycle_path	-setup -end	-from	[get_pins {x_regtwo*|clk y_regtwo*|clk}]				2	1.277

SDC Timing Constraints

- Clocks
- I/O
- Asynchronous paths
- False paths
- Multicycle paths ⬅

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! If it doesn't, simply loosens timing requirements
  - 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 4
- 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 *(default and most common)*
  - 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

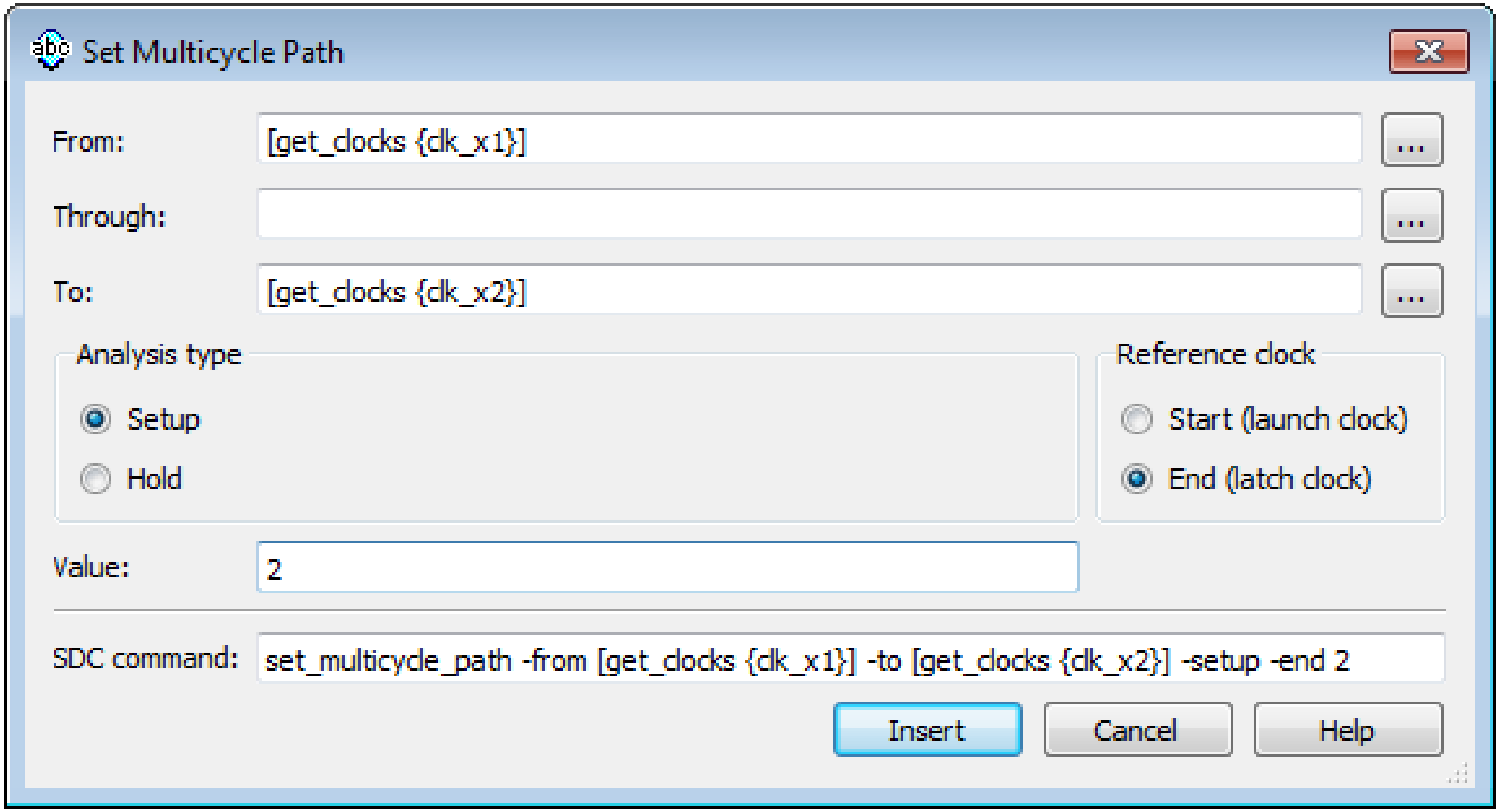
```
[-start | -end]
[-setup | -hold]
[-fall_from <clocks>]
[-rise_from <clocks>]
[-from <names>]
[-through <names>]
[-to <names>]
[-fall_to <clocks>]
[-rise_to <clocks>]
<value>
```

`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 analysis
- *`<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)



Set Multicycle Path

From: [get_clocks {clk_x1}]

Through:

To: [get_clocks {clk_x2}]

Analysis type: [✓ Setup] Hold

Reference clock: Start (launch clock) [✓ End (latch clock)]

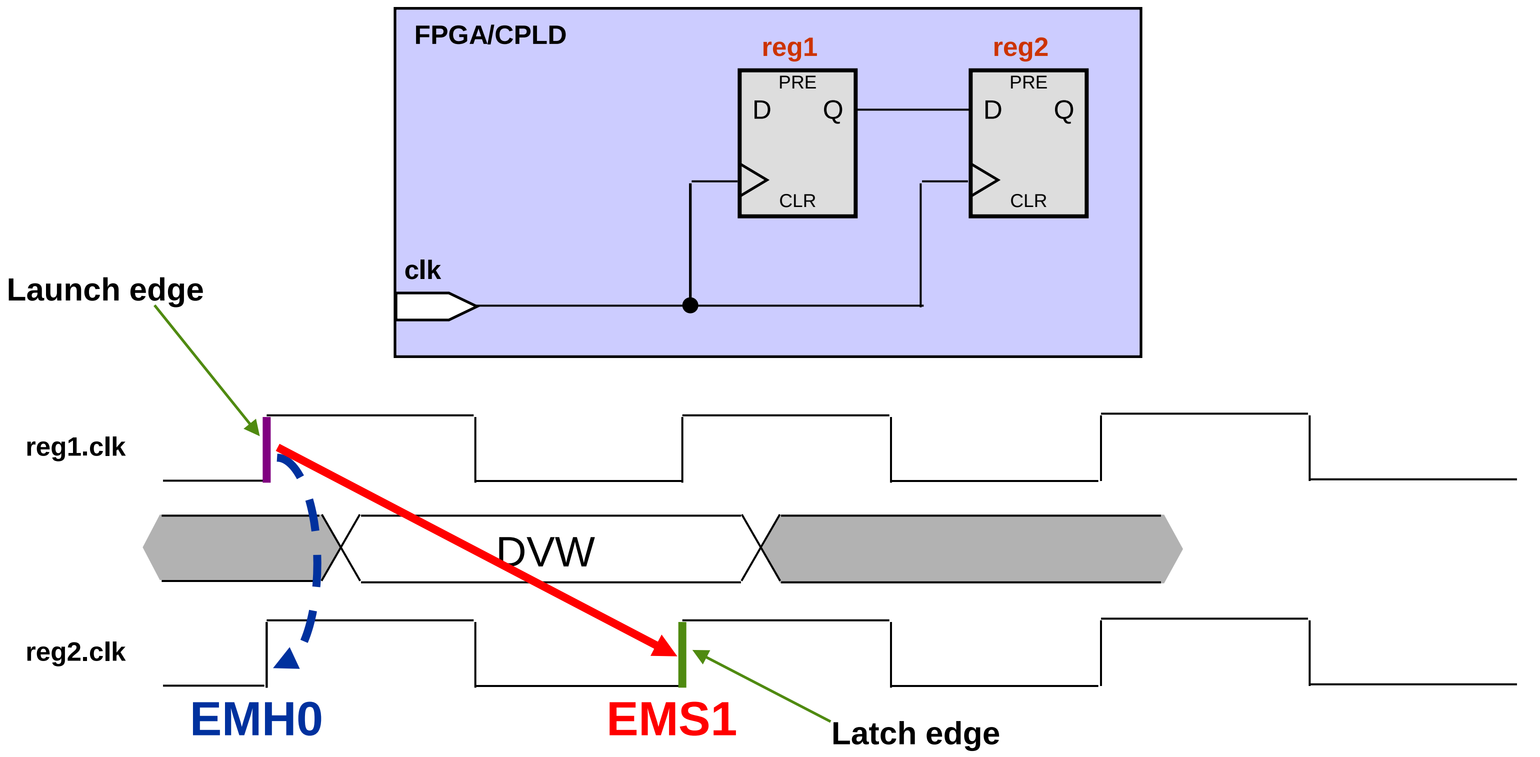
Value: 2

SDC command:

```
set_multicycle_path -from [get_clocks {clk_x1}] -to [get_clocks {clk_x2}] -setup -end 2
```

Understanding Multicycle

- Standard single-cycle register transfer

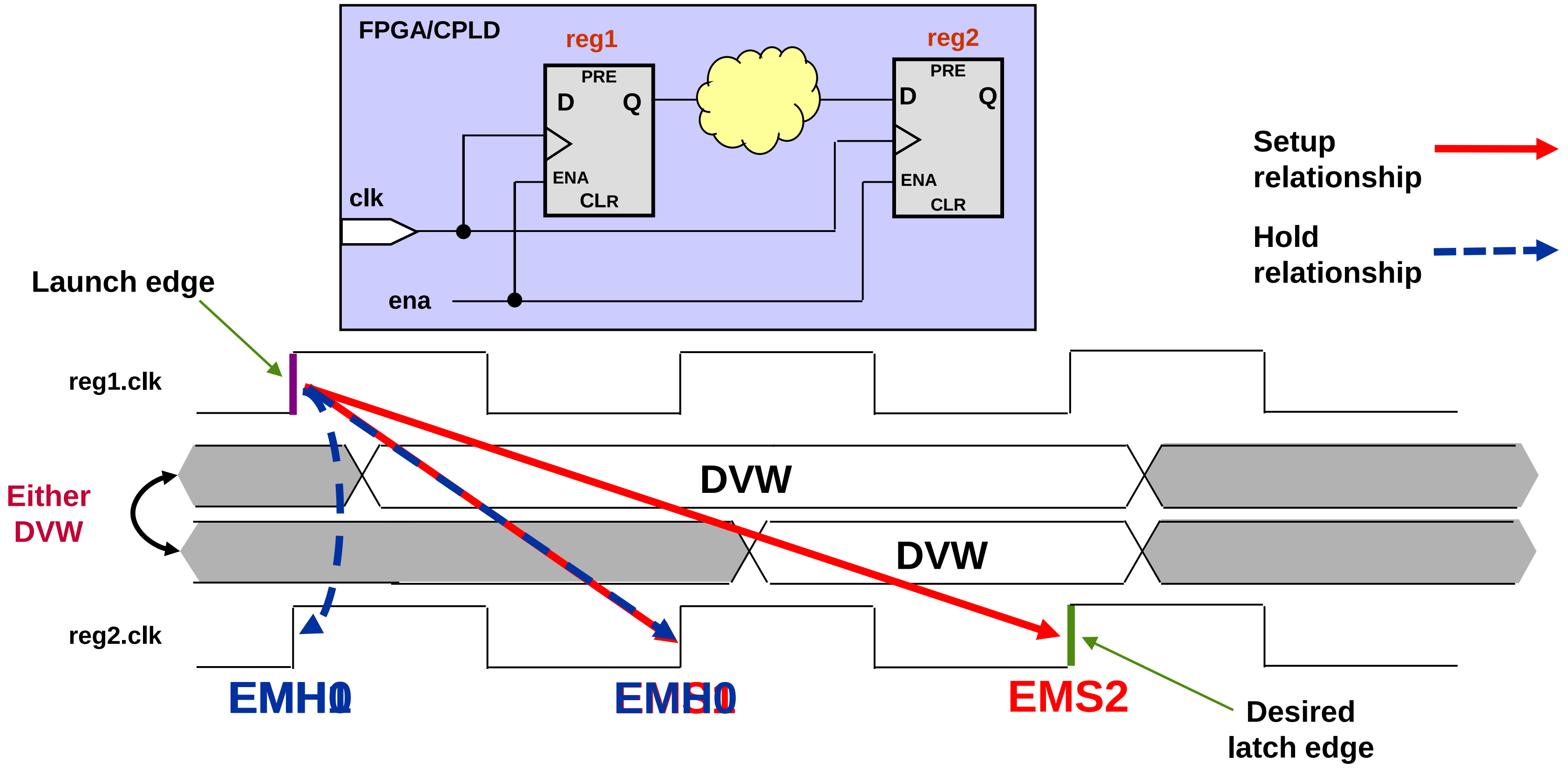


Multicycle Setup = 1 (Default)

Multicycle Hold = 0 (Default)*

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

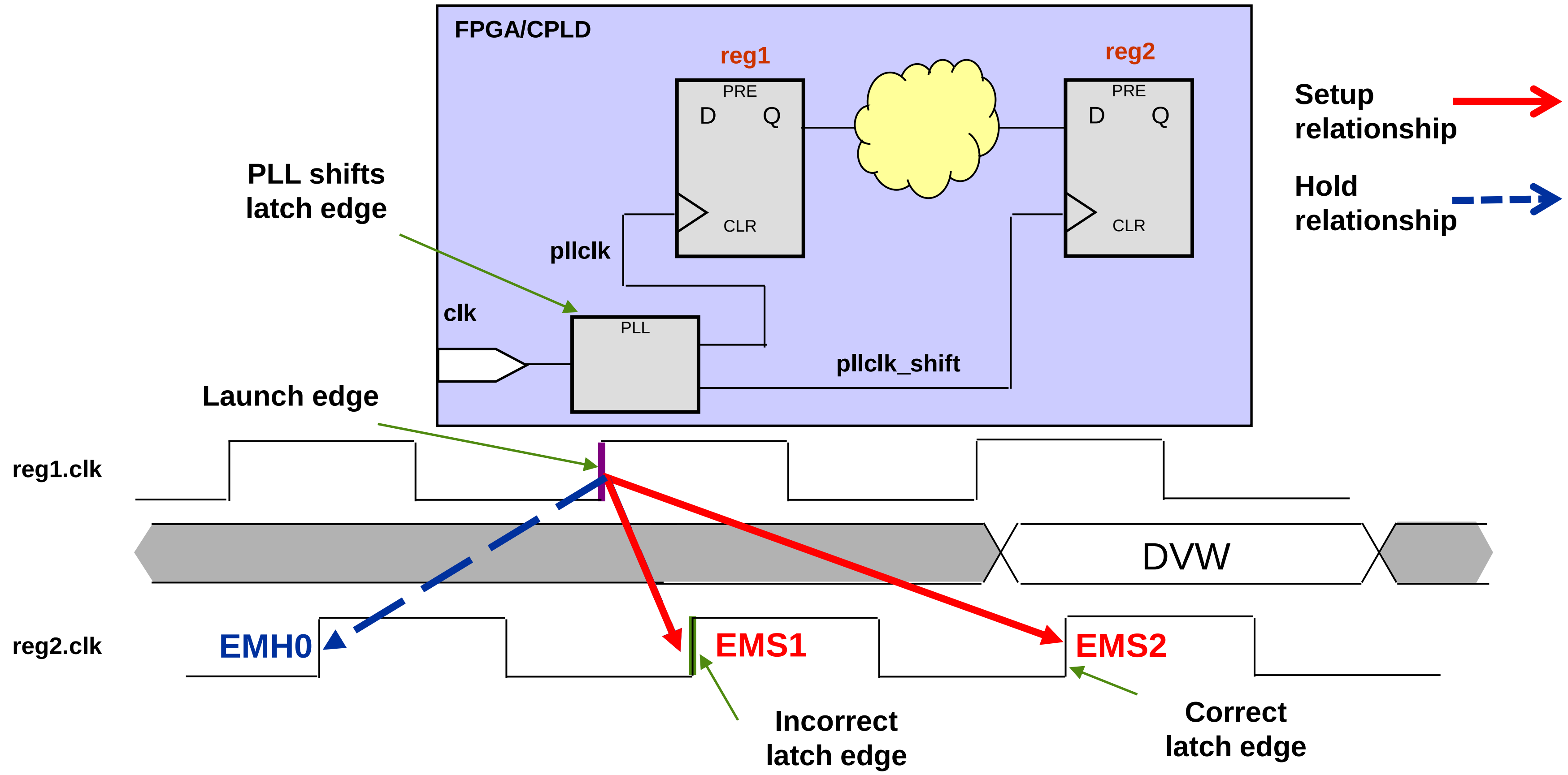
Opening the Window



```
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
```

Shifting the Window

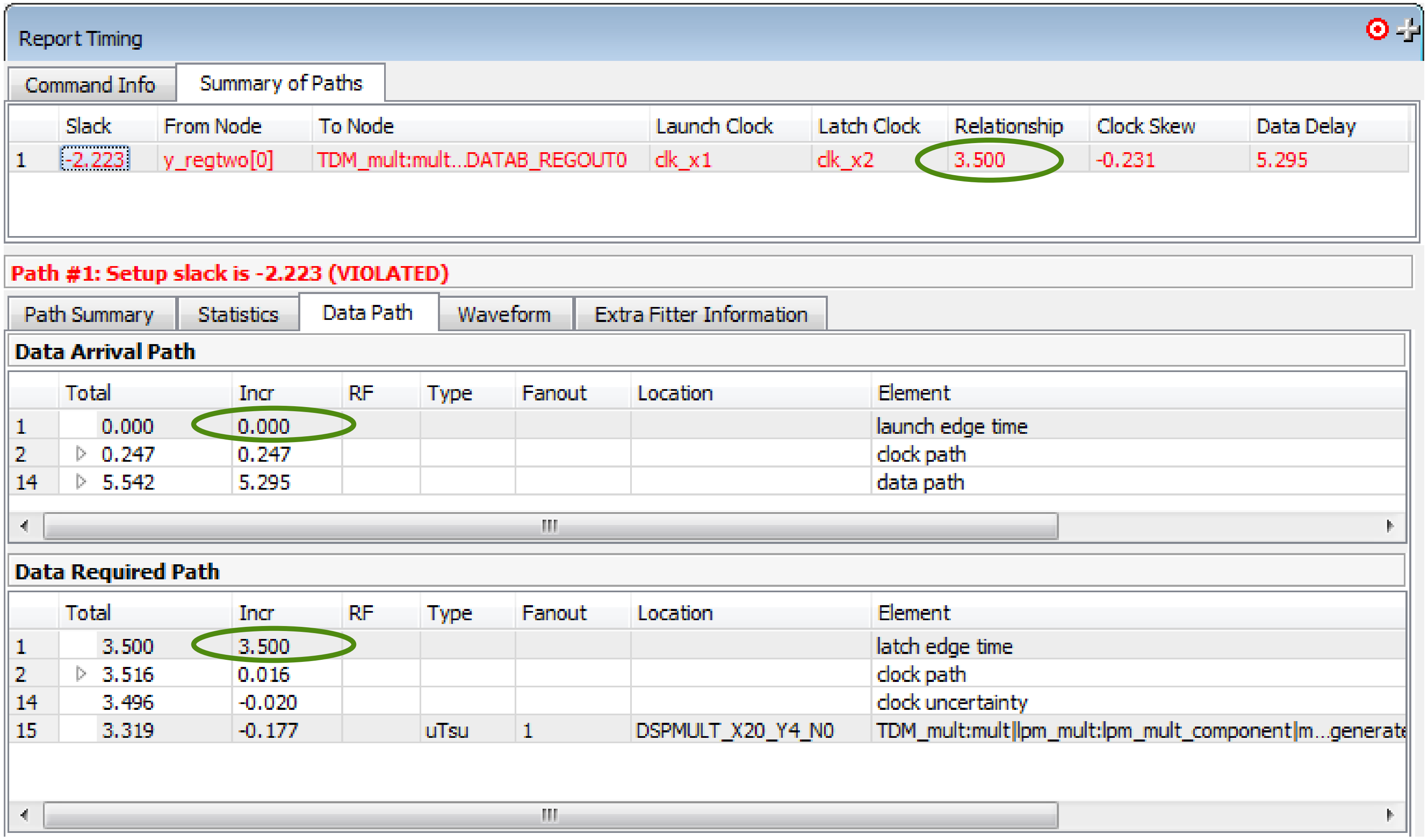
- Change to a *two cycle setup*; *single cycle hold* transfer



```
set_multicycle_path -from [get_pins reg1|clk] -to [get_pins reg2|datain] -setup 2
or
set_multicycle_path -from [get_clocks pllclk] -to [get_clocks pllclk_shift] -setup 2
```

Reporting Multicycles

No multicycle



Report Timing

Command Info | Summary of Paths

	Slack	From Node	To Node	Launch Clock	Latch Clock	Relationship	Clock Skew	Data Delay
1	-2.223	y_regtwo[0]	TDM_mult:mult...DATAB_REGOUT0	clk_x1	clk_x2	3.500	-0.231	5.295

Path #1: Setup slack is -2.223 (VIOLATED)

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

Data Arrival Path

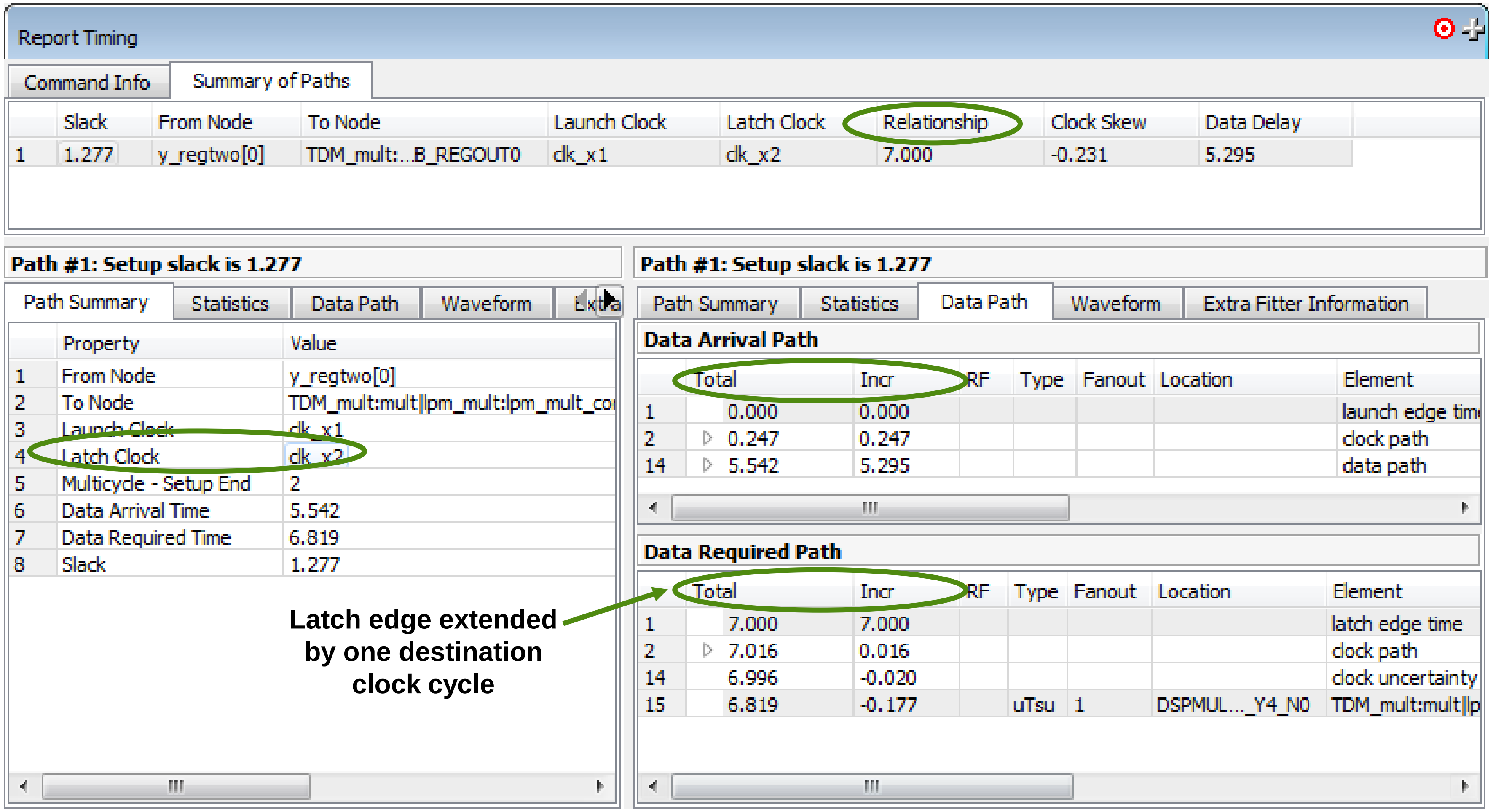
	Total	Incr	RF	Type	Fanout	Location	Element
1	0.000	0.000					launch edge time
2	▷ 0.247	0.247					clock path
14	▷ 5.542	5.295					data path

Data Required Path

	Total	Incr	RF	Type	Fanout	Location	Element
1	3.500	3.500					latch edge time
2	▷ 3.516	0.016					clock path
14	3.496	-0.020					clock uncertainty
15	3.319	-0.177		uTsu	1	DSPMULT_X20_Y4_N0	TDM_mult:mult|lpm_mult:lpm_mult_component|m...generat

Reporting Multicycles

Same path with Setup Multicycle = 2



Report Timing

Command Info | Summary of Paths

	Slack	From Node	To Node	Launch Clock	Latch Clock	Relationship	Clock Skew	Data Delay
1	1.277	y_regtwo[0]	TDM_mult:...B_REGOUT0	clk_x1	clk_x2	7.000	-0.231	5.295

Path #1: Setup slack is 1.277

Path Summary | Statistics | Data Path | Waveform | Extra

	Property	Value
1	From Node	y_regtwo[0]
2	To Node	TDM_mult:mult|lpm_mult:lpm_mult_co
3	Launch Clock	clk_x1
4	Latch Clock	clk_x2
5	Multicycle - Setup End	2
6	Data Arrival Time	5.542
7	Data Required Time	6.819
8	Slack	1.277

Path #1: Setup slack is 1.277

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 tim
2	▷ 0.247	0.247					clock path
14	▷ 5.542	5.295					data path

Data Required Path

	Total	Incr	RF	Type	Fanout	Location	Element
1	7.000	7.000					latch edge time
2	▷ 7.016	0.016					clock path
14	6.996	-0.020					clock uncertainty
15	6.819	-0.177		uTsu	1	DSPMUL..._Y4_N0	TDM_mult:mult|lp

Latch edge extended by one destination clock cycle

Please go to Exercise 4

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