Design of Digital Circuits and Systems

Static Timing Analysis

Instructor: Justin Hsia

Teaching Assistants:

Colton Harris
Deepti Anoop
Gayathri Vadhyan
Jared Yoder
Lancelot Wathieu
Matthew Hung

Relevant Course Information

- Lab 4 due Friday (5/3)

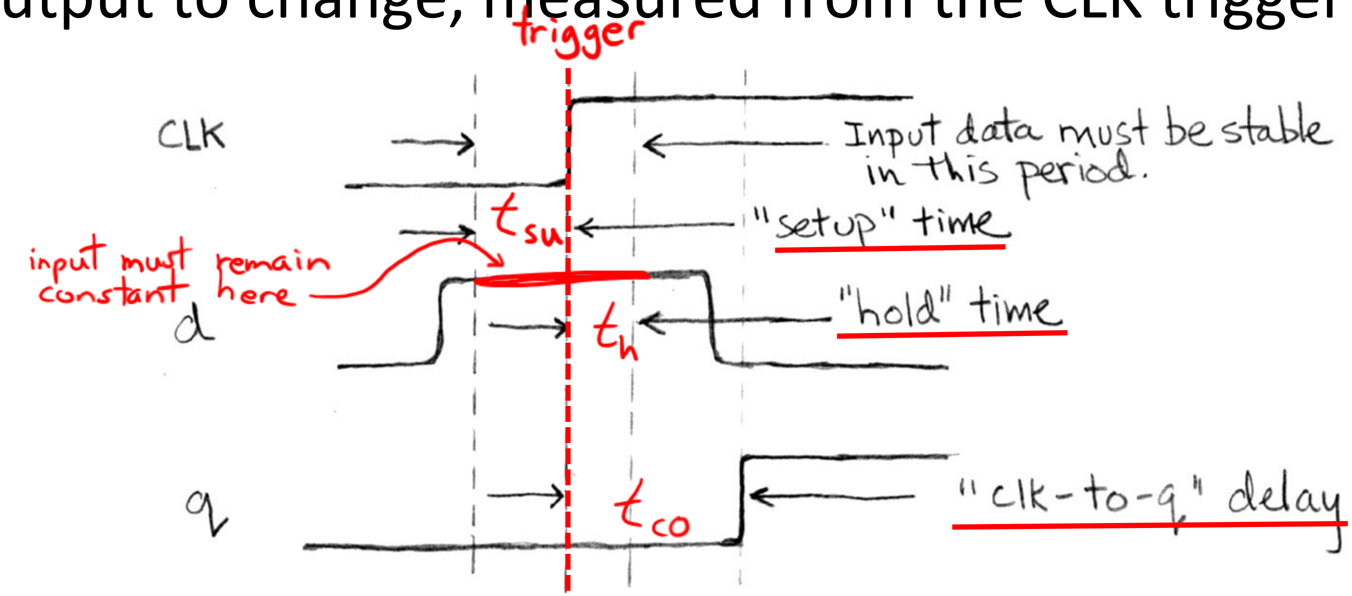
- Quiz 3 is this Thursday at 11:**50** am (30 min)
  - ASM and ASMD charts

- Homework 5 (5/10) and Lab 5 (5/17) will be released on Thursday
  - Lab 5 is the longest “regular” lab, mostly because of debugging – careful with number representation

- Rest of course material will NOT show up in labs

Review: Sequential Timing Constraints

- *Setup Time (t_s or t_{su}):* how long the input must be stable *before* the CLK trigger for proper input read
- *Hold Time (t_h):* how long the input must be stable *after* the CLK trigger for proper input read
- *"CLK-to-Q" Delay (t_{C2Q} or t_{CO}):* how long it takes the output to change, measured from the CLK trigger

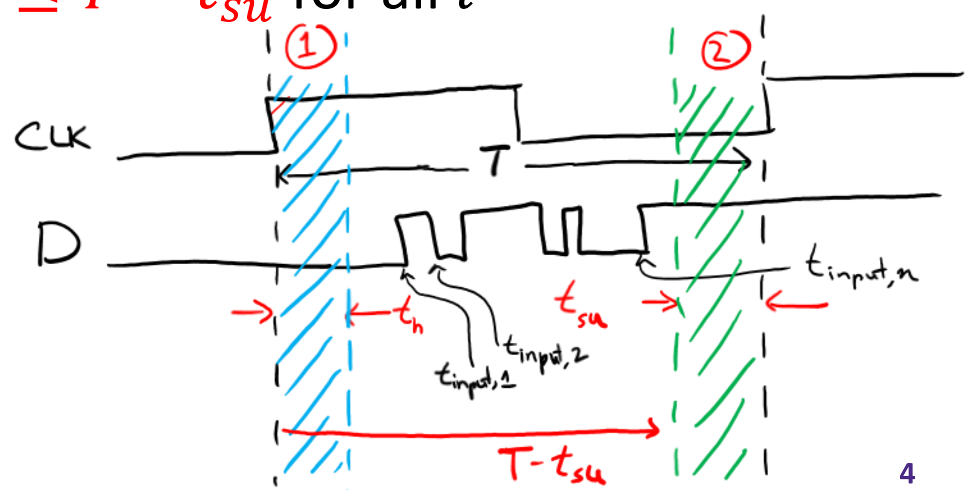


When Can the Input Change?

- When a register input changes shouldn't violate hold time (t_h) or setup time (t_{su}) constraints within a clock period (T)
- Let $t_{input,i}$ be the time it takes for the input of a register to change for the i-th time in a single clock cycle, measured from the CLK trigger:
  - Then we need $t_h \overset{①}{\leq} t_{input,i} \overset{②}{\leq} T - t_{su}$ for all i
  - Two separate constraints!

① $t_{input,1} \geq t_h$

② $t_{input,n} \leq T - t_{su}$

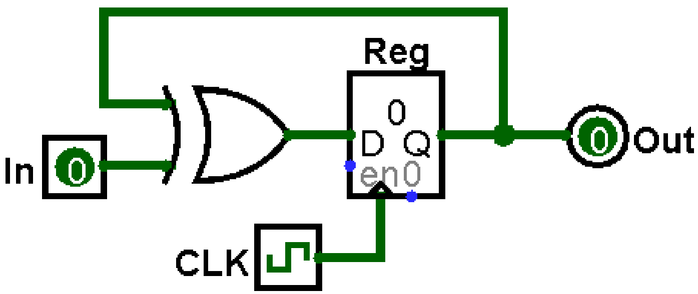


Timing Constraint Questions

1) What are you solving for? Which constraint/inequality is relevant?
   - $t_h \rightarrow$ hold time constraint
   - t_{su} or $T \rightarrow$ setup time constraint
   - Combinational logic delay → "max" means setup time constraint, "min" means hold time constraint
2) Find the most relevant pathway to a register input
   - Hold time constraint → shortest pathway
   - Setup time constraint → longest pathway
3) Solve the inequality for chosen path using the given constants

Practice Question

- Let T = 300 ps,
  t_{su} = 80 ps, t_h = 50 ps,
  t_{CO} = 100 ps.



1) If `In` changes exactly on clock triggers, what are the limits on the XOR gate delay that ensure proper behavior?

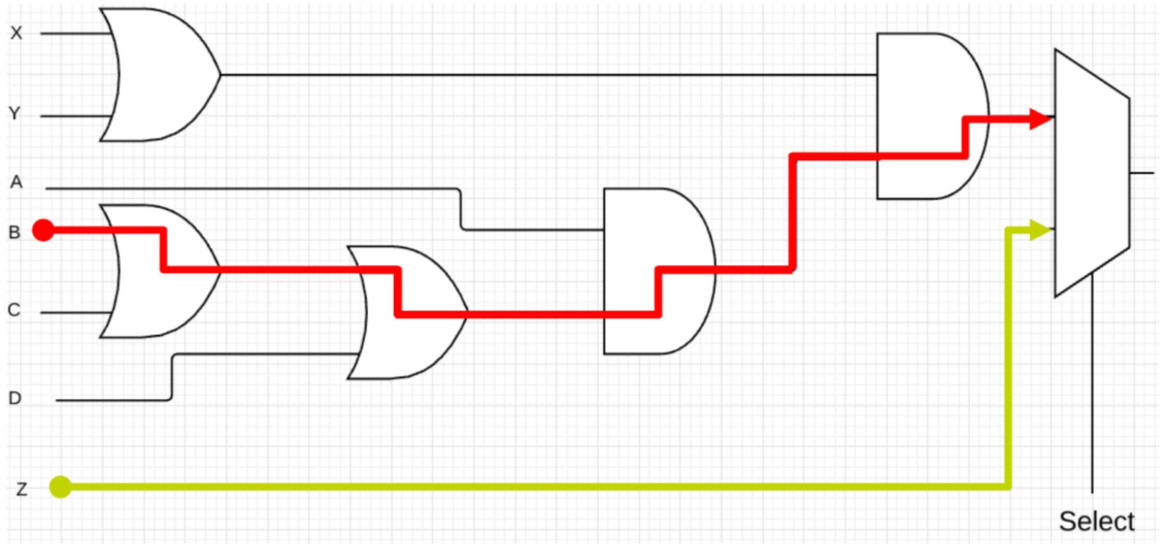
$$t_{XOR} \in (\_\_\_\_, \_\_\_\_) \text{ ps}$$

2) Now let t_{XOR} = 100 ps and `In` change a fixed t_{in} after each clock trigger, what ranges of t_{in} will result in proper behavior? Answer within a single clock cycle:

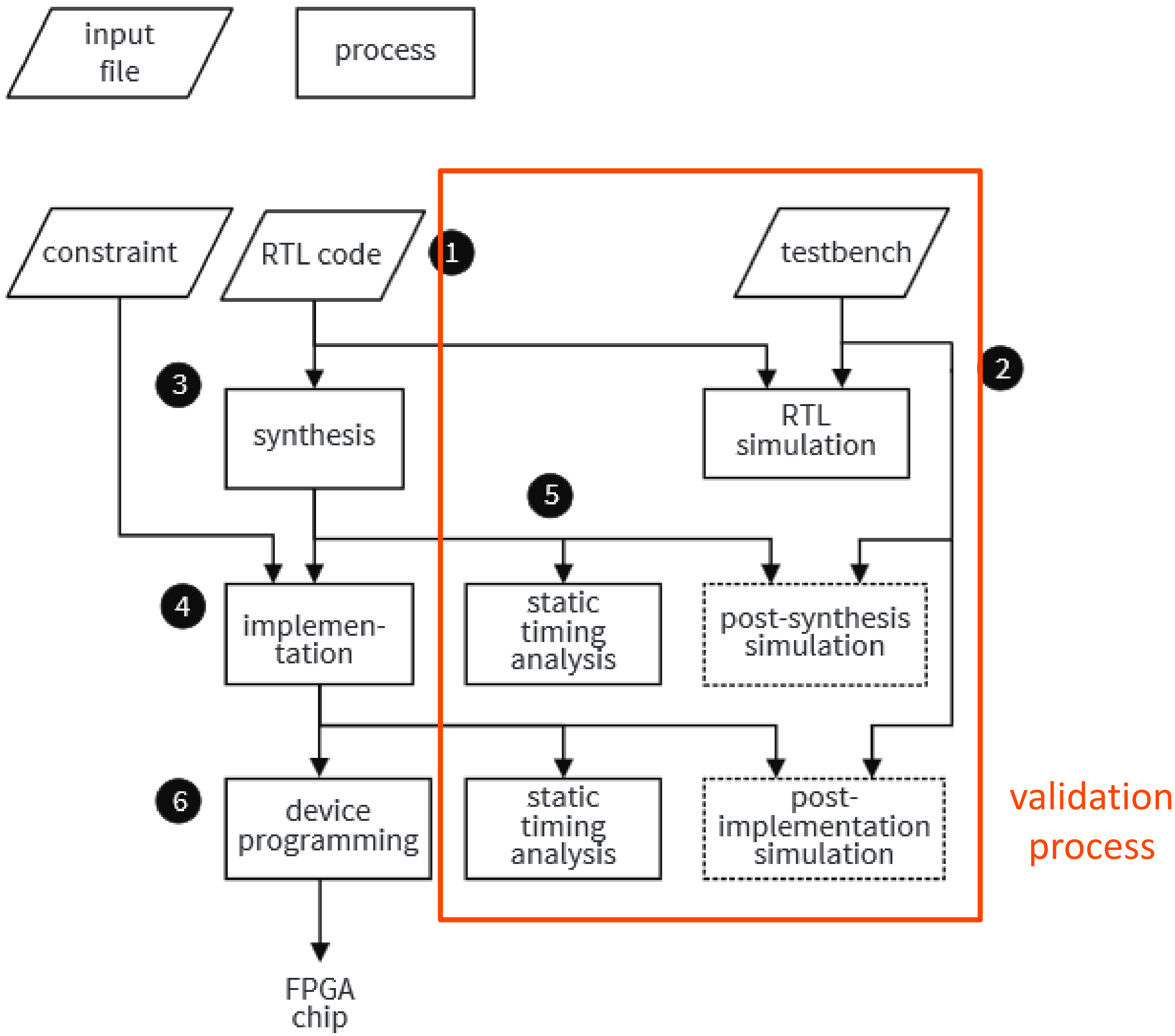
$$t_{in} \in [0, \_\_\_\_) \cup (\_\_\_\_, 300] \text{ ps}$$

What Affects Circuit Timing?

- A more realistic picture:
  - **Logic depth of circuit pathways**
  - Length, resistance, and capacitance of wire
  - Size of the transistors
  - Operating conditions: Voltage & Temperature

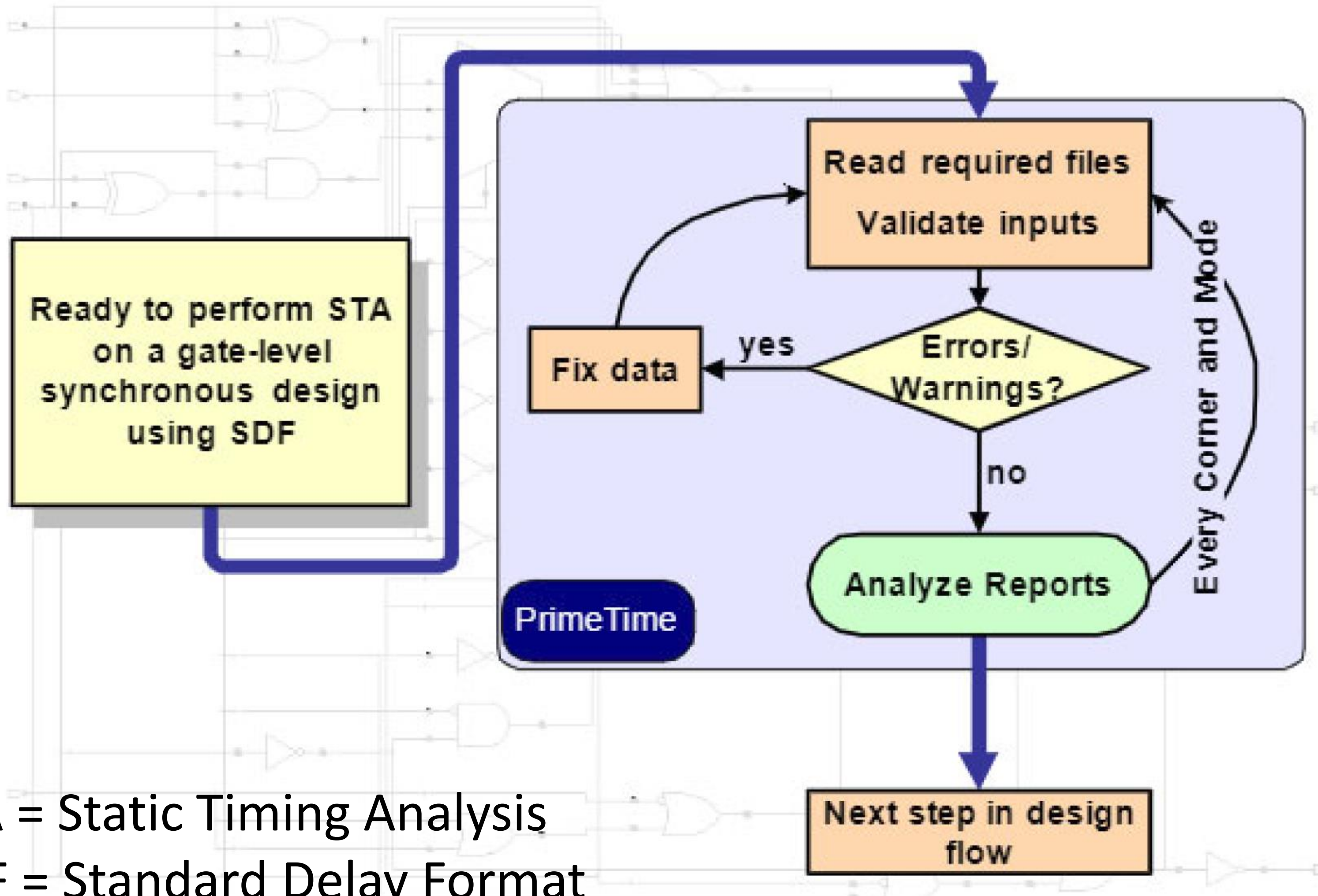


Review: FPGA-Based Design Flow



Source: Figure 2.3 from "FPGA Prototyping" by P. Chu

Static Timing Analysis Flow

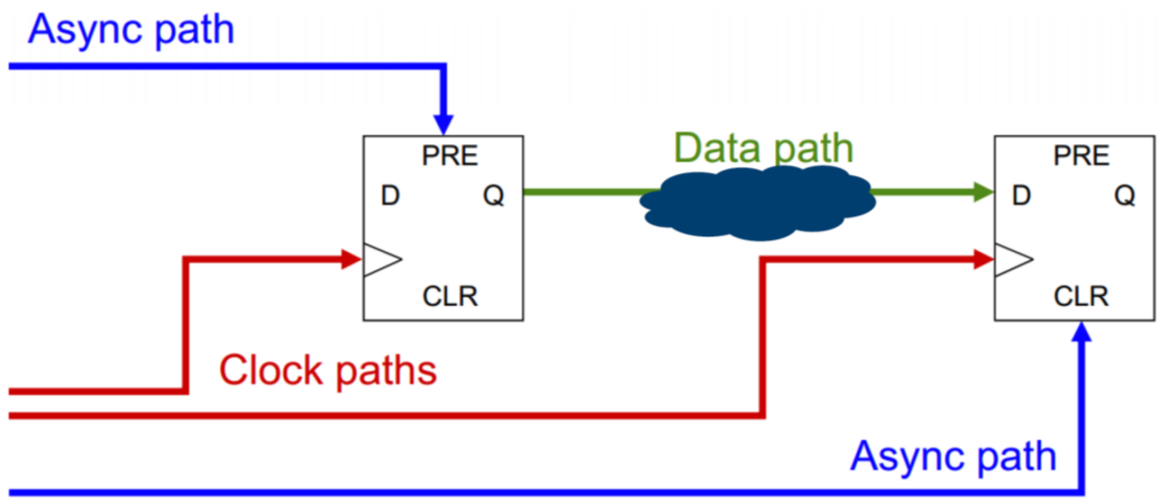


STA = Static Timing Analysis

SDF = Standard Delay Format

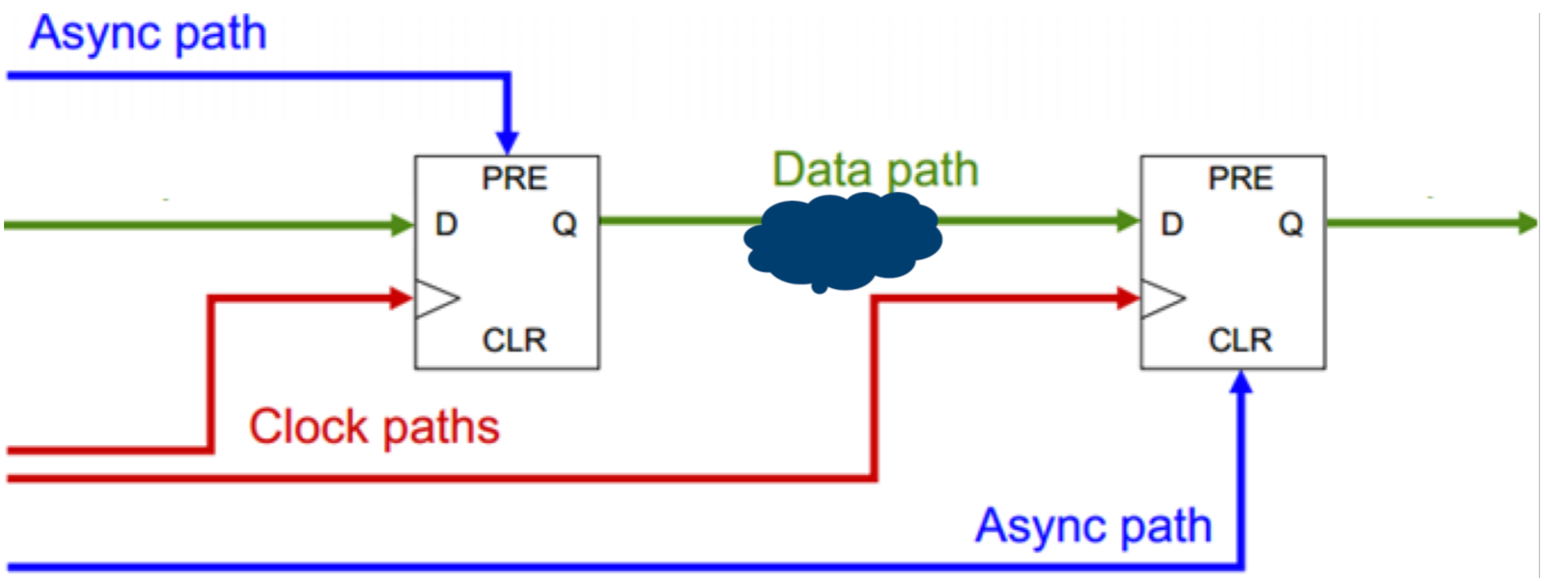
Circuit Path Categorization

- **Data paths** are between inputs, sequential elements, and outputs
- **Clock paths** are from device ports or internally-generated clocks to the clock pins of sequential elements
- **Asynchronous paths** are between inputs and asynchronous set and clear pins of sequential elements



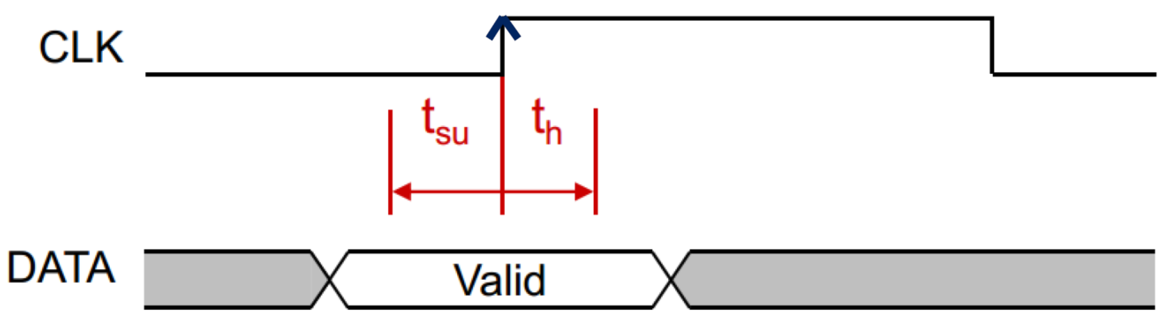
Data Paths

- Three data path types:
  1) Input to sequential element
  2) Sequential element to sequential element
  3) Sequential element to output



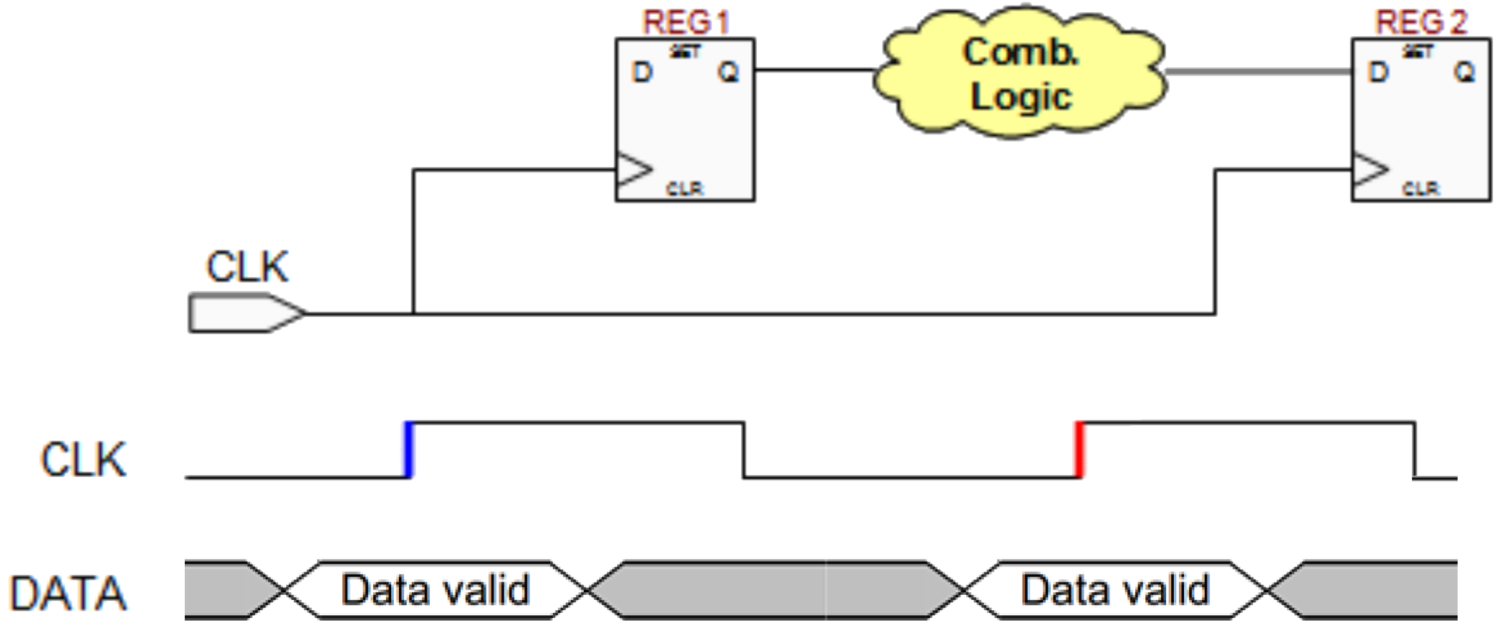
Data Path Validity

- A data signal is considered **valid** when is has finished computing and is stable
  - Note that this remains true until the next round of computation begins, which can cross clock cycle boundaries
- For proper behavior (*i.e.*, correct reads), the input to a register must remain valid throughout the setup and hold time constraints



Clock Edges

- Looking at a single clock cycle
  - **Launch Edge:** the clock edge that activates or *launches* the source register
  - **Latch Edge:** the clock edge that *latches* the data into the destination register



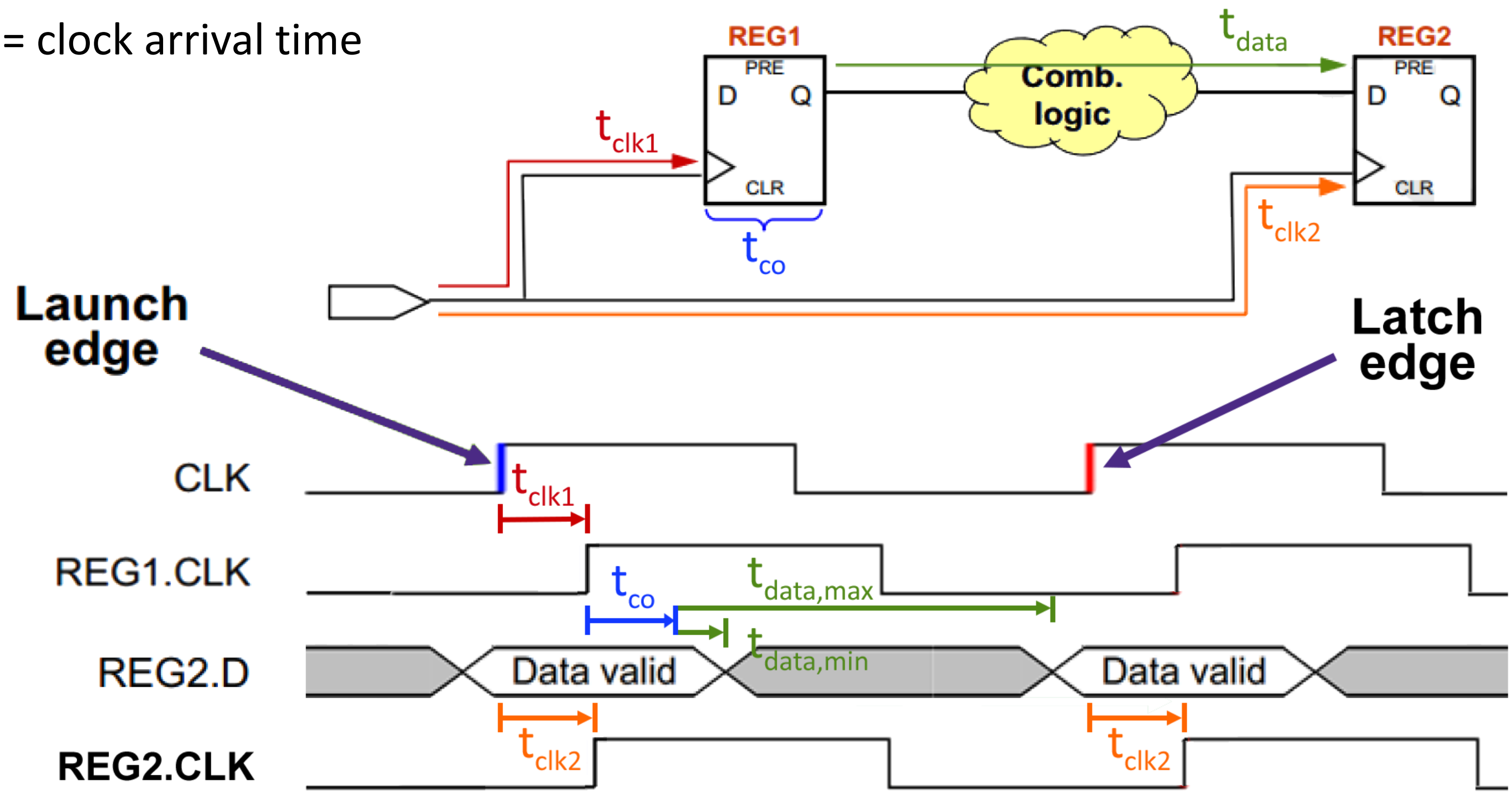
Timing Delays and Points of Interest

- **Clock Network Delay** (t_{clk})
  - How long it takes for changes in the clock signal to arrive at the register – the cause of *clock skew*
- **Clock-to-Q or Clock-to-out Delay** (t_{C2Q}, t_{CQ}, or t_{CO})
  - How long it takes the output to change, measured from the register's *received* clock edge – a property of the FF/register
- **Data Arrival Time** (DAT)
  - The time at which the destination register's input is settled
- **Clock Arrival Time** (CAT)
  - The time at which a clock edge arrives at the destination register

Timing Delays and Points of Interest

DAT = data arrival time

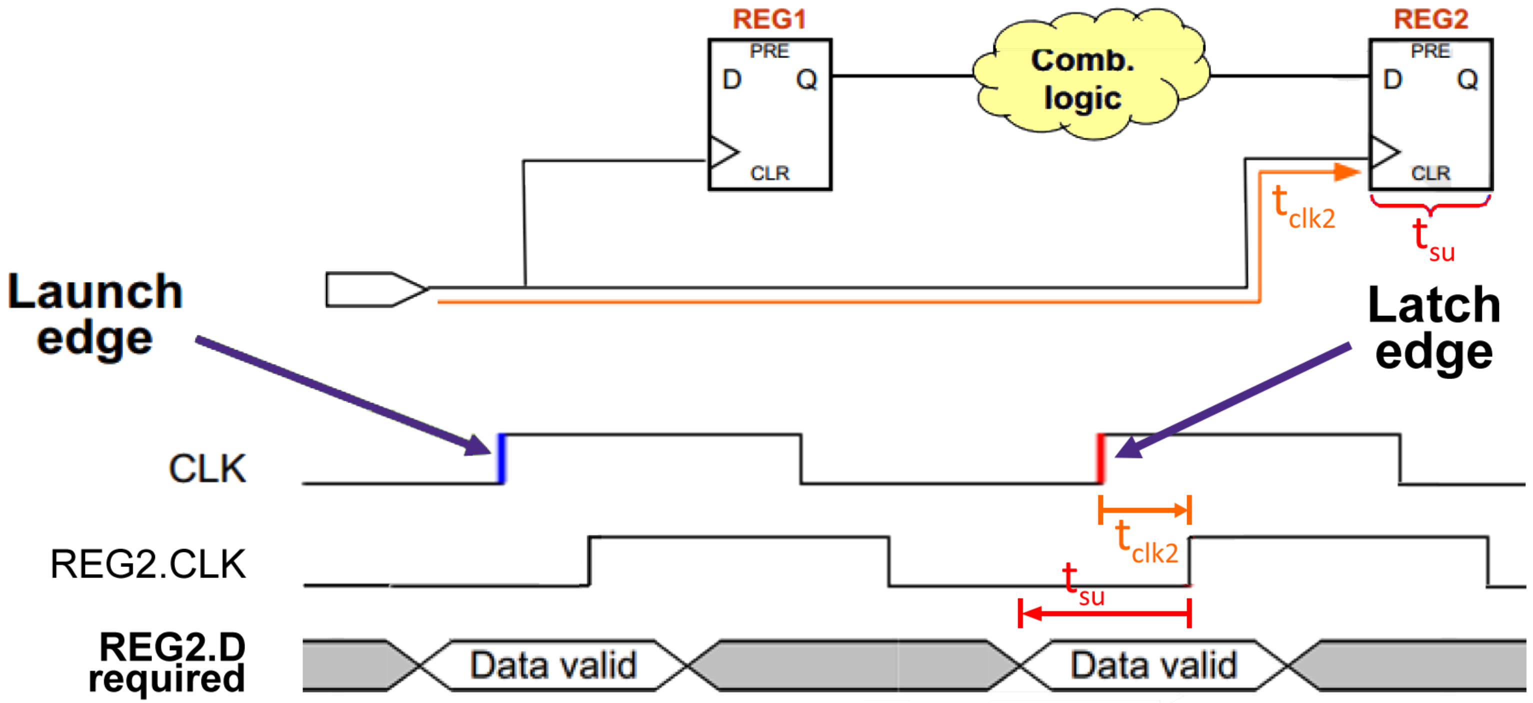
CAT = clock arrival time



- DAT_{su} = *launch* edge + $t_{clk1,max}$ + $t_{co,max}$ + $t_{data,max}$
- DAT_{h} = *launch* edge + $t_{clk1,min}$ + $t_{co,min}$ + $t_{data,min}$
- $CAT_{latch/launch}$ = edge + t_{clk2}

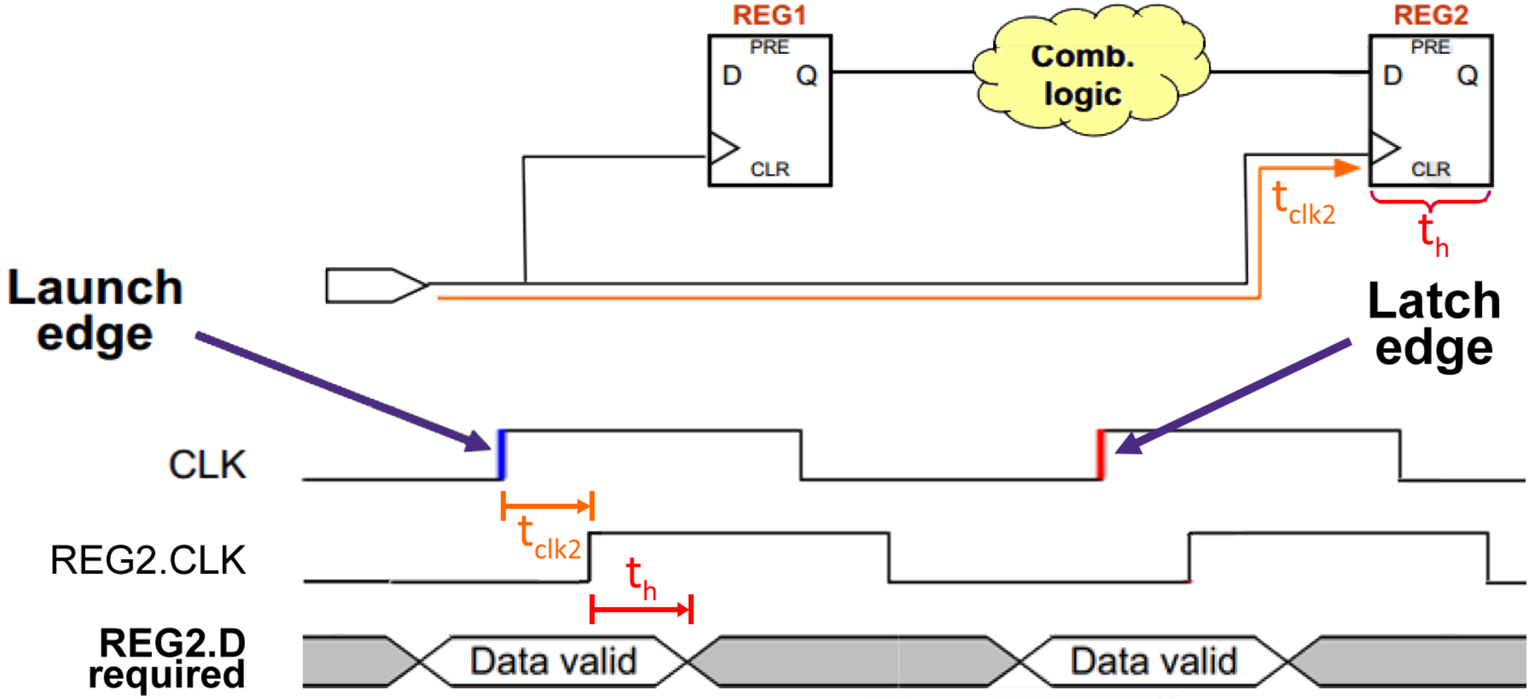
Data Required Time (DRT)

- Data Required Time for Setup (DRT_{su})
  - The minimum time required *before* the *latch* edge for data to get latched into the destination register
  - Clock Arrival Time$_{latch}$ – Setup Time



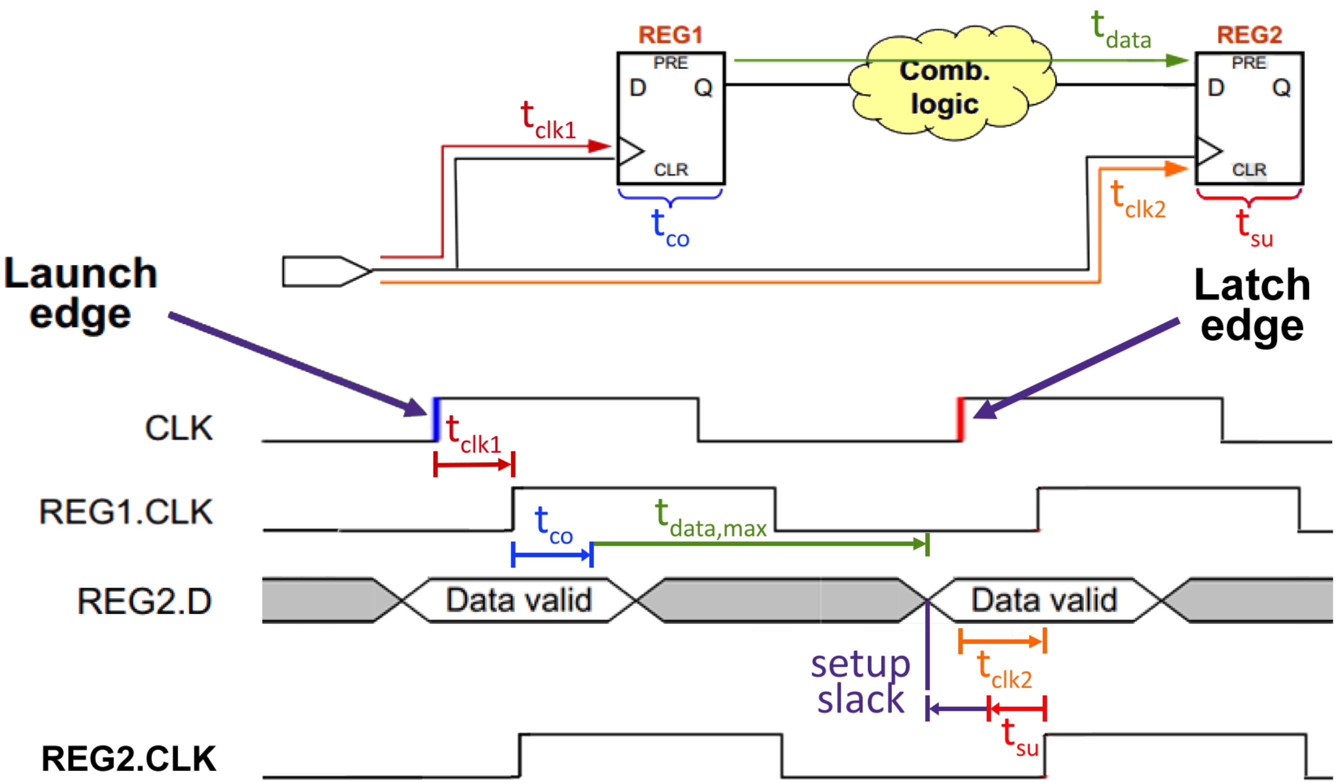
Data Required Time (DRT)

- Data Required Time for Hold (DRT_h)
  - The minimum time required *after* the *launch* edge for the data to remain valid for successful latching
  - Clock Arrival Time$_{launch}$ + Hold Time



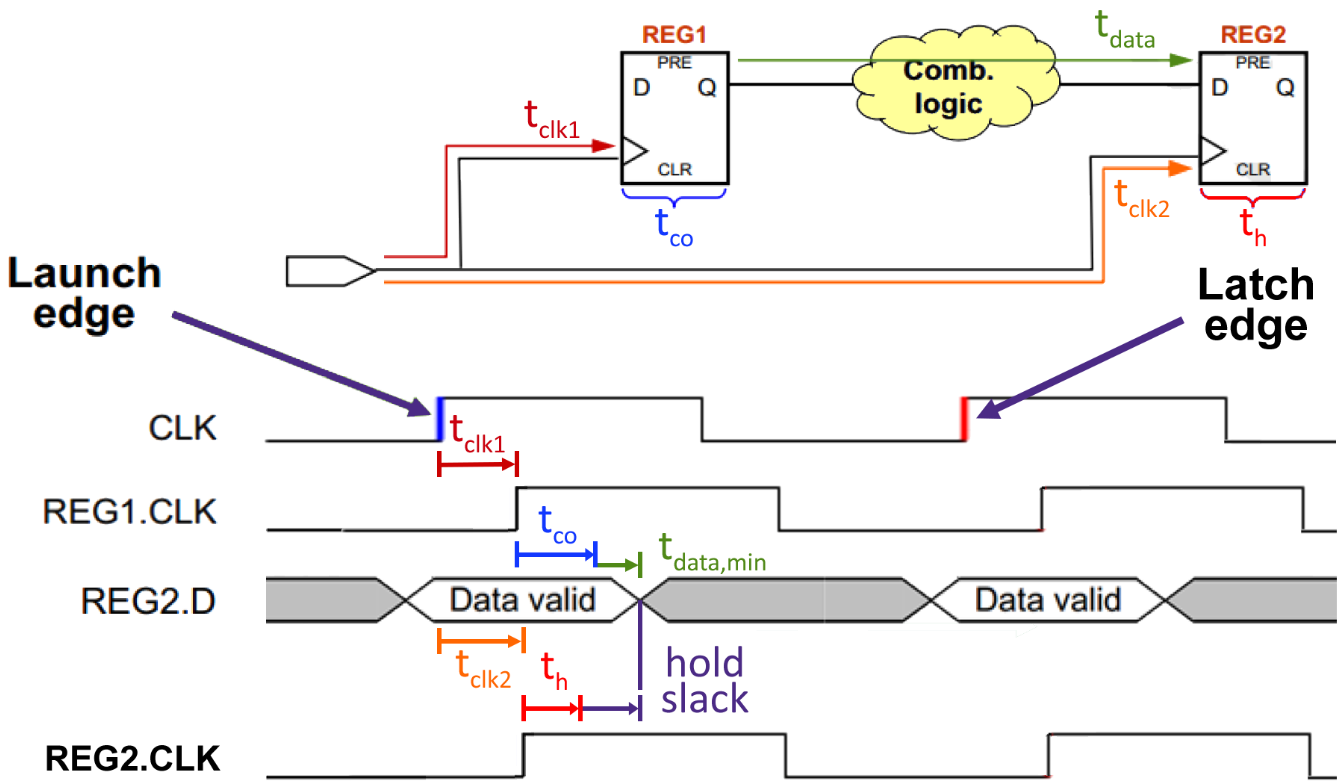
Setup Slack

- Wiggle room for setup time violations
  - Setup slack = $DRT_{su} - DAT_{su}$
  - Depends on clock frequency



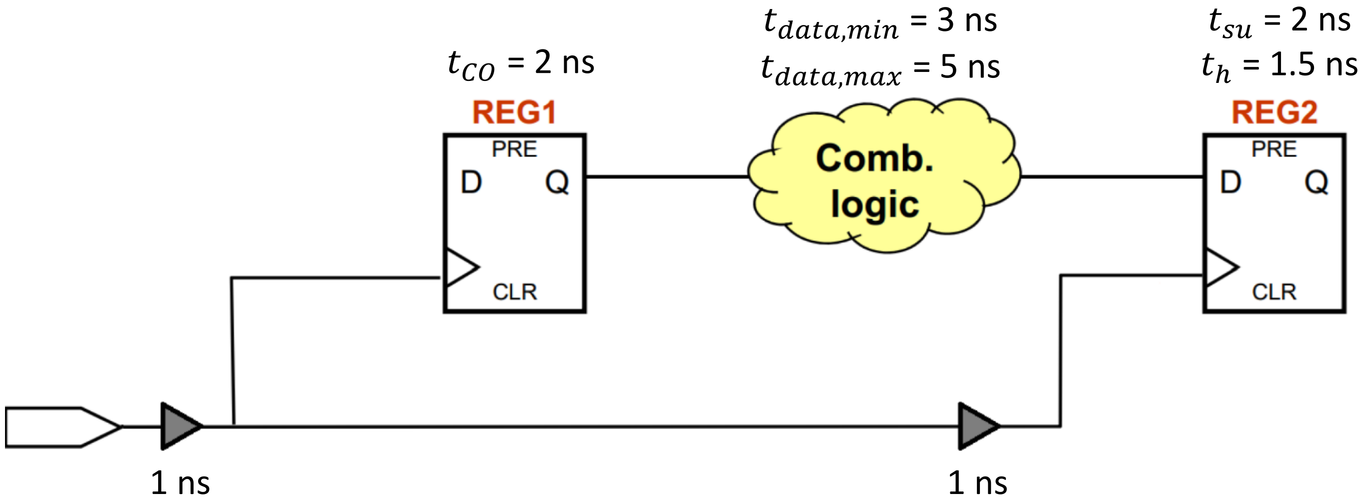
Hold Slack

- Wiggle room for hold time violations
  - Hold slack = $DAT_h - DRT_h$
  - Does not depend on clock frequency



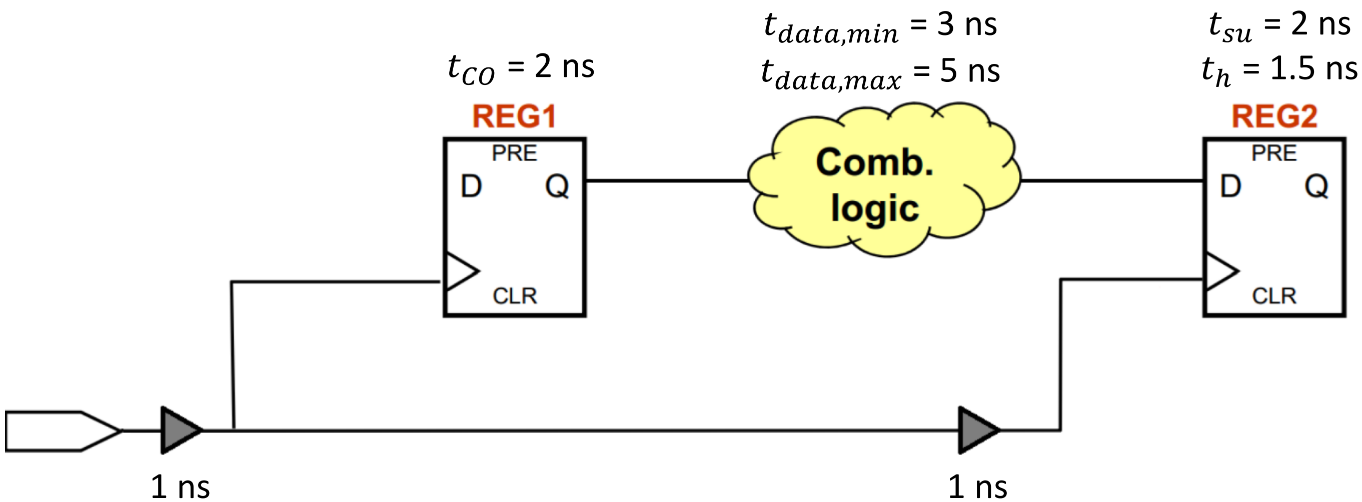
Technology Break

Setup Slack Example



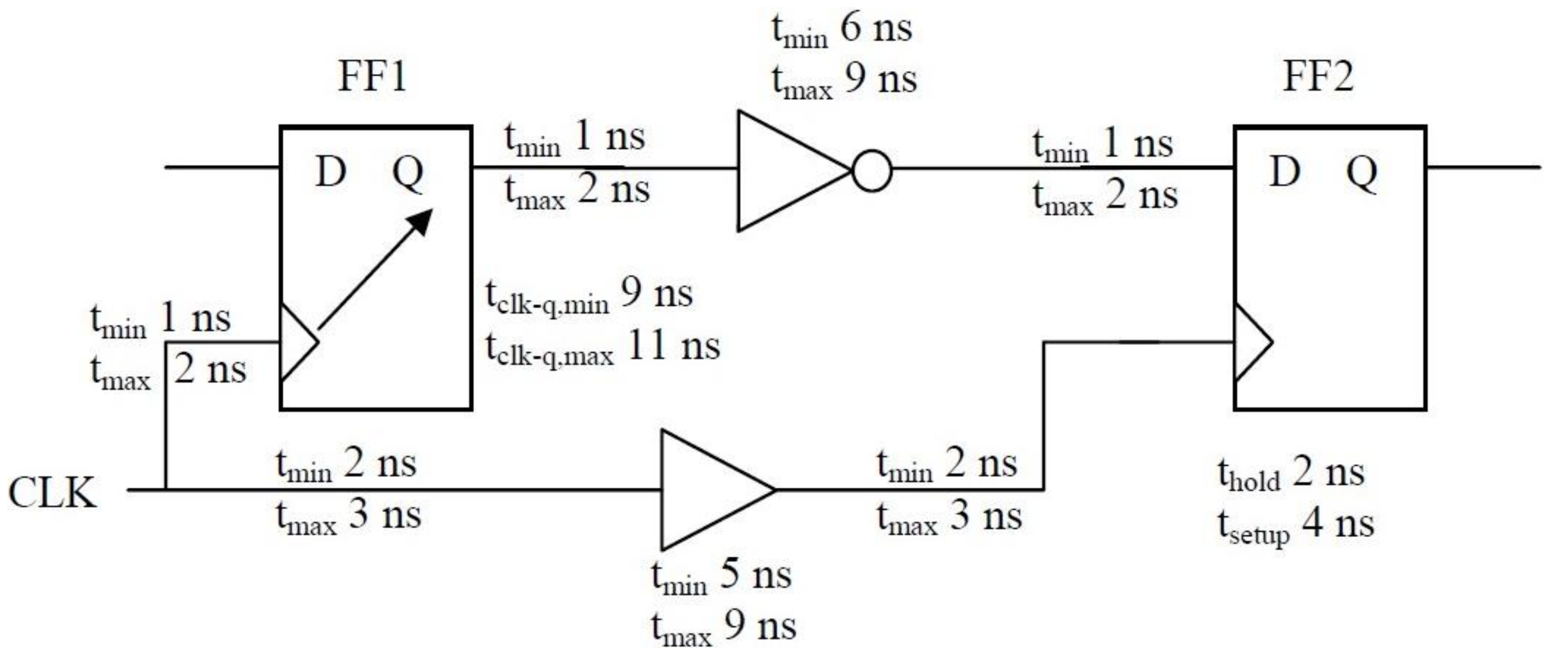
- Assume 100 MHz clock; clock period T = 10 ns
- Setup slack = DRT_{su} – DAT_{su}
  = *min* clock path – *max* data path

Hold Slack Example



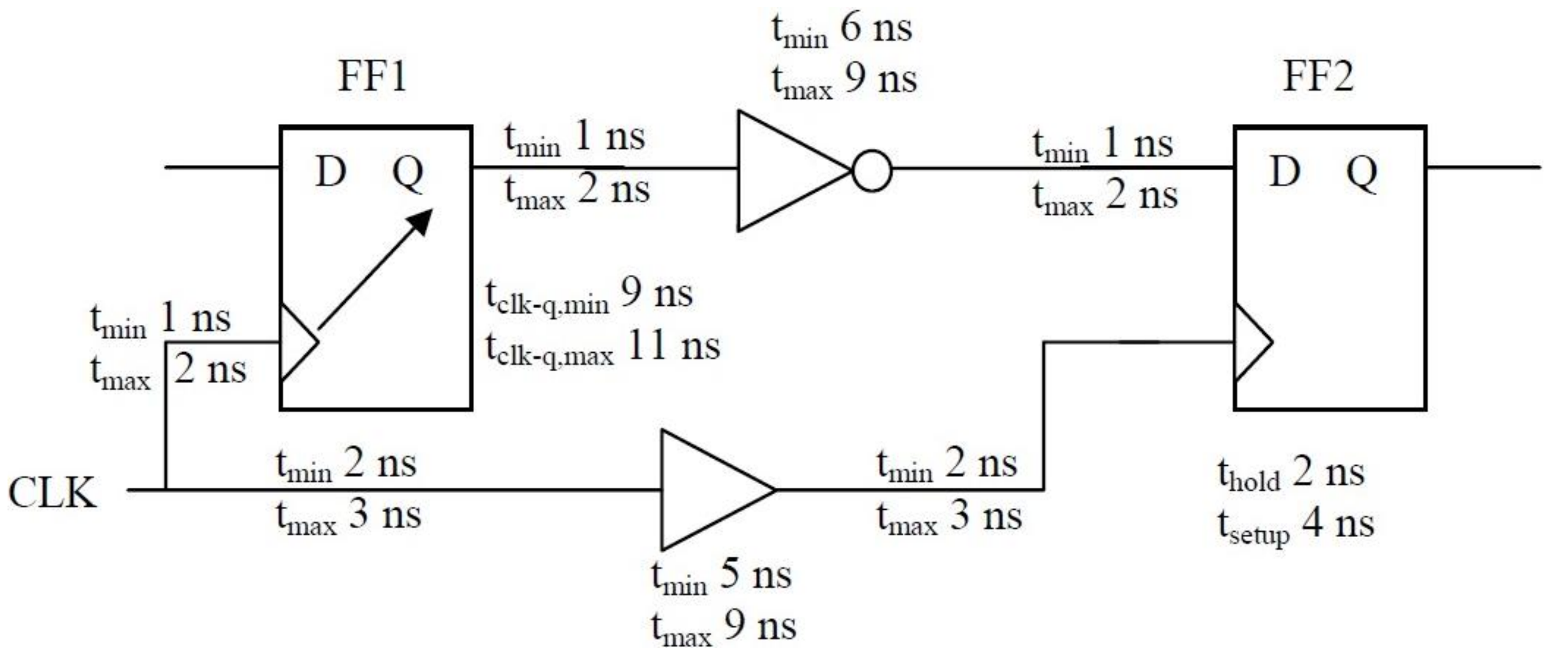
- Assume 100 MHz clock; clock period T = 10 ns
- Hold slack = $DAT_h - DRT_h$
  = *min* data path – *max* clock path

Slack Example



- Calculate the setup and hold slacks:
  - Assume 50 MHz clock; clock period T = 20 ns
  - setup slack = *min* clock path – *max* data path
  - hold slack = *min* data path – *max* clock path

Slack Example



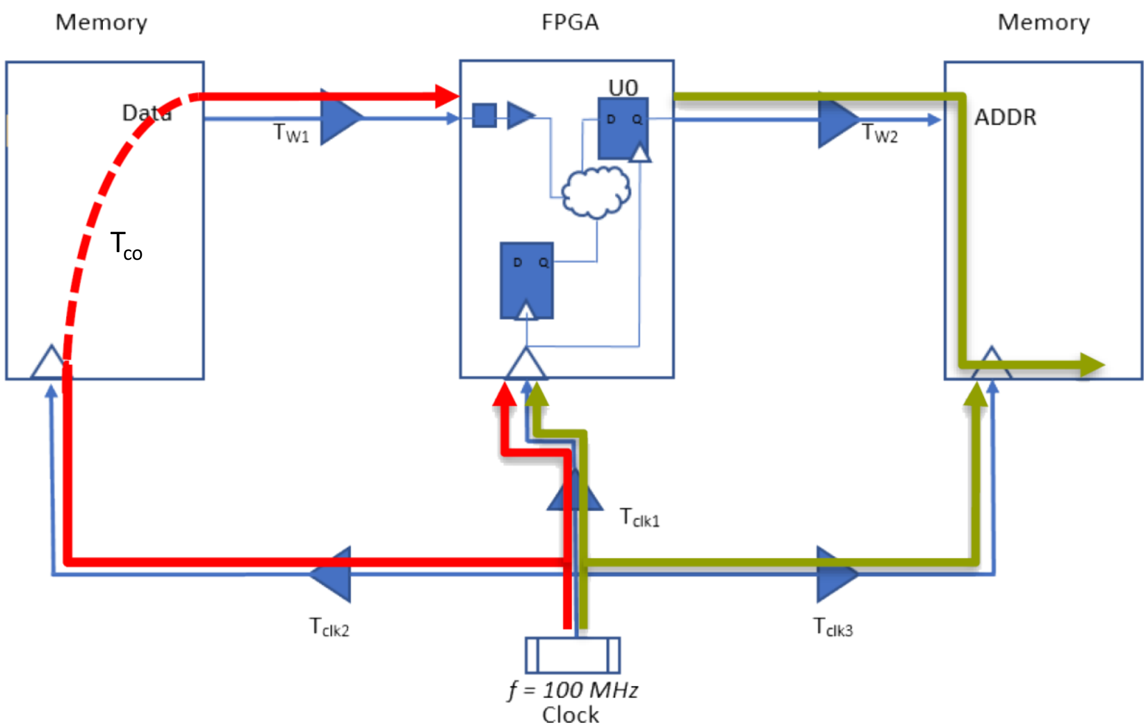
- What is the fastest clock we can use with this circuit?

Synopsys Design Constraints (SDC)

- Synopsys Design Constraints describe timing constraints and exceptions
  - Quartus uses *.sdc* files to define clocks, I/O constraints, and other information that the fitter needs to make the best decisions
  - You can make and edit these using the TimeQuestGUI or by editing the *.sdc* file in a text editor (see hw5)

I/O Constraints

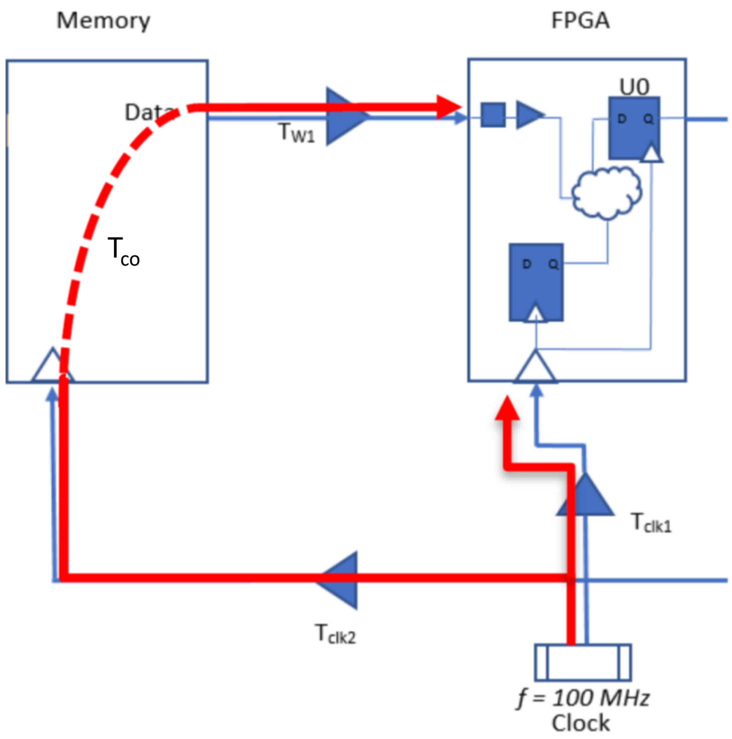
- Let's examine the design constraints for the following example FPGA setup:
  - FPGA circuit between two external chips
  - T_w are wire/board delays, T_{clk} are clock delays *



* every delay has a min and max range

Input Delays

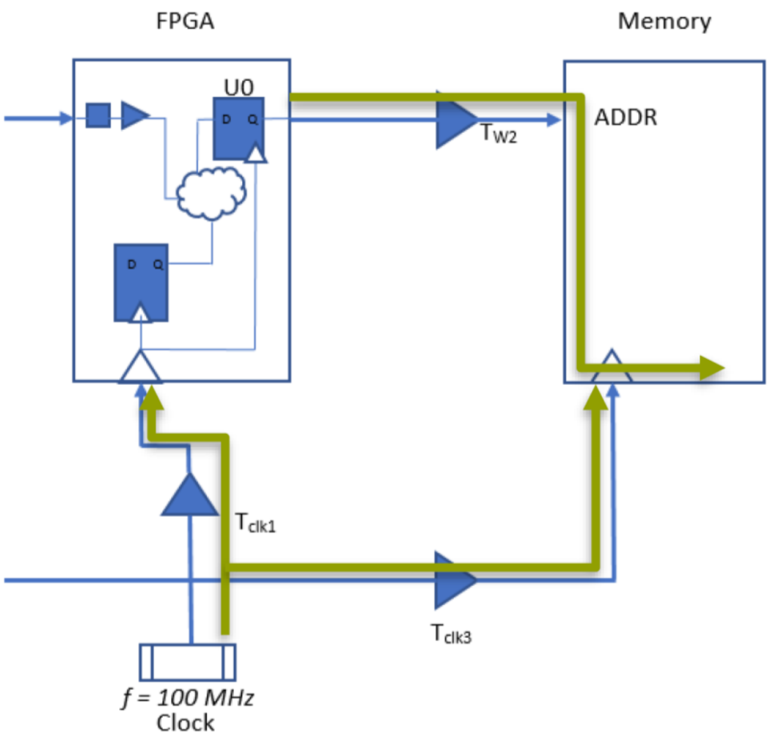
- `set_input_delay`
  - Command to specify external delays feeding into the FPGA's input ports
  - https://www.intel.com/content/www/us/en/programmable/quartushelp/13.0/mergedProjects/tafs/tafs/tcl_pkg_sdc_ver_1.5_cmd_set_input_delay.htm



- Applied to this setup:
  - input $delay_{min}$ = $(T_{clk2,min} - T_{clk1,max}) + T_{CO,min} + T_{W1,min}$
  - input $delay_{max}$ = $(T_{clk2,max} - T_{clk1,min}) + T_{CO,max} + T_{W1,max}$

Output Delays

- `set_output_delay`
  - Command to specify external delays leaving the FPGA's output ports
  - https://www.intel.com/content/www/us/en/programmable/quartushelp/13.0/mergedProjects/tafs/tafs/tcl_pkg_sdc_ver_1.5_cmd_set_output_delay.htm



- Applied to this setup:
  - $\text{output delay}_{min} = (T_{clk1,min} - T_{clk3,max}) + T_{W2,min} - T_{h,max}$
  - $\text{output delay}_{max} = (T_{clk1,max} - T_{clk3,min}) + T_{W2,max} + T_{su,max}$

I/O Constraints

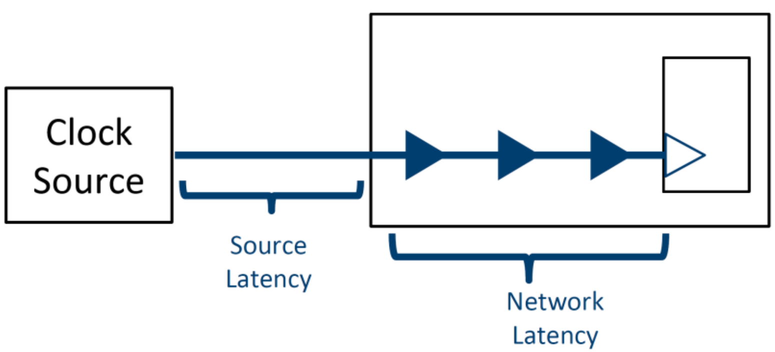
- Check the datasheet!

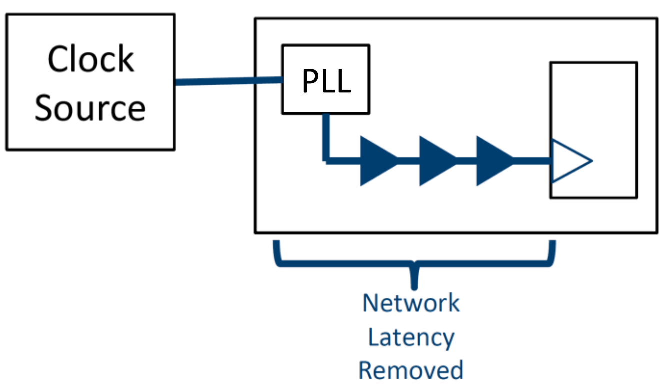
READ/WRITE CYCLE SWITCHING CHARACTERISTICS (Over Operating Range)

		-166		-133		
Symbol	**Parameter**	**Min.**	**Max.**	**Min.**	**Max.**	**Unit**
f_{MAX}[3]	Clock Frequency	—	166	—	133	MHz
t_{KC}[3]	Cycle Time	6	—	7.5	—	ns
t_{KH}	Clock High Time	2.4	—	2.8	—	ns
t_{KL}[3]	Clock Low Time	2.4	—	2.8	—	ns
t_{KQ}[3]	Clock Access Time	—	3.5	—	4	ns
t_{KQX}[1]	Clock High to Output Invalid	3	—	3	—	ns
t_{KQLZ}[1,2]	Clock High to Output Low-Z	0	—	0	—	ns
t_{KQHZ}[1,2]	Clock High to Output High-Z	1.5	3.5	1.5	3.5	ns
t_{OEQ}[3]	Output Enable to Output Valid	—	3.5	—	3.8	ns
t_{OEQX}[1]	Output Disable to Output Invalid	0	—	0	—	ns
t_{OELZ}[1,2]	Output Enable to Output Low-Z	0	—	0	—	ns
t_{OEHZ}[1,2]	Output Disable to Output High-Z	2	4.5	2	5	ns
t_{AS}[3]	Address Setup Time	2.1	—	2.1	—	ns
t_{SS}[3]	Address Status Setup Time	1.5	—	1.5	—	ns
t_{WS}[3]	Write Setup Time	1.5	—	1.5	—	ns
t_{CES}[3]	Chip Enable Setup Time	1.5	—	1.5	—	ns
t_{AVS}[3]	Address Advance Setup Time	1.5	—	1.5	—	ns
t_{AH}[3]	Address Hold Time	1.0	—	1.0	—	ns
t_{SH}[3]	Address Status Hold Time	0.5	—	0.5	—	ns
t_{WH}[3]	Write Hold Time	0.5	—	0.5	—	ns
t_{CEH}[3]	Chip Enable Hold Time	0.5	—	0.5	—	ns
t_{AVH}[3]	Address Advance Hold Time	0.5	—	0.5	—	ns

Clock Effects

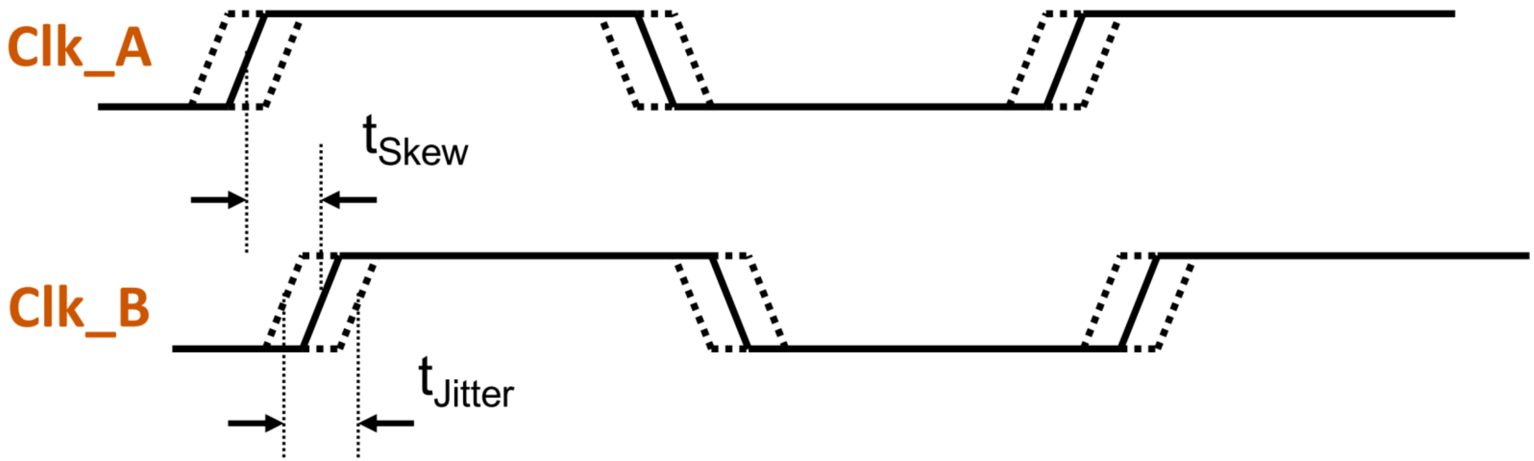
- **Clock Latency**: delay for clock signal to reach components
  - Source latency: the delay between the clock definition and its source
  - Network latency: the delay between the clock definition and register clock pins
    - [extra] Often dealt with using an onboard phase-locked loop (PLL) – see `derive_pll_clocks` command





Clock Effects

- **Clock Uncertainty**: clock signal does not arrive at every clock pin simultaneously
  - Skew: differences in arrival time of a clock signal to different components
  - Jitter: deviations from defined period
  - Synchronous clock domain crossings (future lecture)



Timing Closure

- Timing constraints need to be met *simultaneously*
  - Sometimes fixing one violation will introduce another
- Fixing **hold violations**: caused by fast data path and destination register's clock latency
  - Add delay in the data path with buffers or pairs of inverters (done automatically by Quartus)
- Fixing **setup violations**: data arrives too late compared to the destination register's clock speed
  - Slow down the clock (undesirable)
  - Tell fitter to try harder or confine logic to a smaller area
  - Rewrite code to simplify logic
  - Add *pipelining* (next lecture)