# Timing Analysis

# Contents

- Foundry Library; Liberty format;
- Gates: Propagation Delays;
- Analyze Flops: Propagation Delay; Setup time; hold Time
- Describe Contamination delay; Recovery time; Removal time
- Describe Clock frequency; Jitter; Skew(source & network latency);
- Analyze Timing Paths; Multi-input path; Clock Budget;
- Describe Multi-Clock; Multi-Cycle Path; False Path; Retiming

# Library

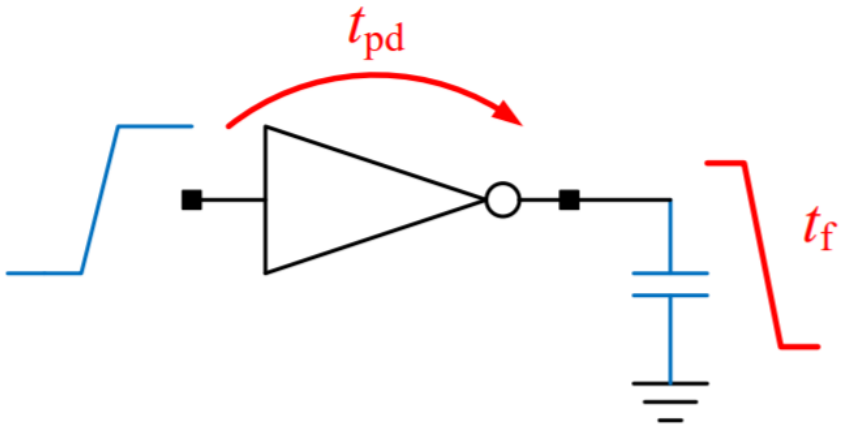
- Standard Cell Library:
- Standard cell library is a collection of well defined and pre-characterized logic cells with multi-drive strength and multi-threshold voltage cells in the form of a predefined standard cell layout.
- Cell Collections:
- In general, a standard cell library contains the following types of cell:
  - All basic and universal gates (like AND, OR, NOT, NAND, NOR, XOR etc)
  - Complex gates (like MUX, HA, FA, Comparators, AOI, OAI etc)
  - Clock tree cells (like Clock buffers, clock inverters etc)
  - Flip flops and latches
  - Delay cells
  - Physical only cells
  - Scannable Flip flops

# Library

- File Collections:
- Apart from the standard cells, Standard cell library is delivered with a collection of files which contains all the information required to auto place and route. These files are mainly:
  - LIB files (.lib) - Timing library (LIB or DB) files are generated during the characterization of cells. Library files contain cell delay, power and area information
  - LEF files (.lef) - Physical library (LEF) file is an abstract view of the layout of the cells. LEF file contains the information of cell boundary, Pins inside the cell, location, direction, and metal layer of each pin.
  - Netlist file (.v ) - Netlist file is a Verilog file of the standard cell which defines the functionality of a cell
  - GDS file (.gds) - GDS file is the layout of the standard cell
  - SPICE Netlist (.sp) - SPICE netlist is the netlist of cell in SPICE format is used for simulation
  - Model file (.m) - Model file contains the various design parameters of the cell required for SPICE simulation

# Liberty

- Liberty files with characterization of timing and power for STA
- Liberty Timing Models (.lib)
  - How do we know the delay through a gate in a logic path?
    - Running SPICE is way too complex.
  - Instead, create a timing model that will simplify the calculation.
- Goal:
  - For every timing arc, calculate:
    - Propagation Delay (tpd)
    - Output transition (trise, tfall)
  - Based on:
    - Input net transition.
    - Output Load Capacitance

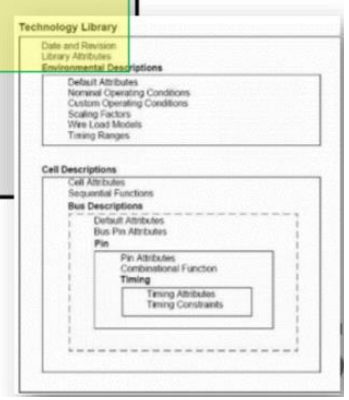



Note that every .lib will provide timing/power/noise information for a single corner, i.e., process, voltage, temperature, RCX, etc.

# Liberty

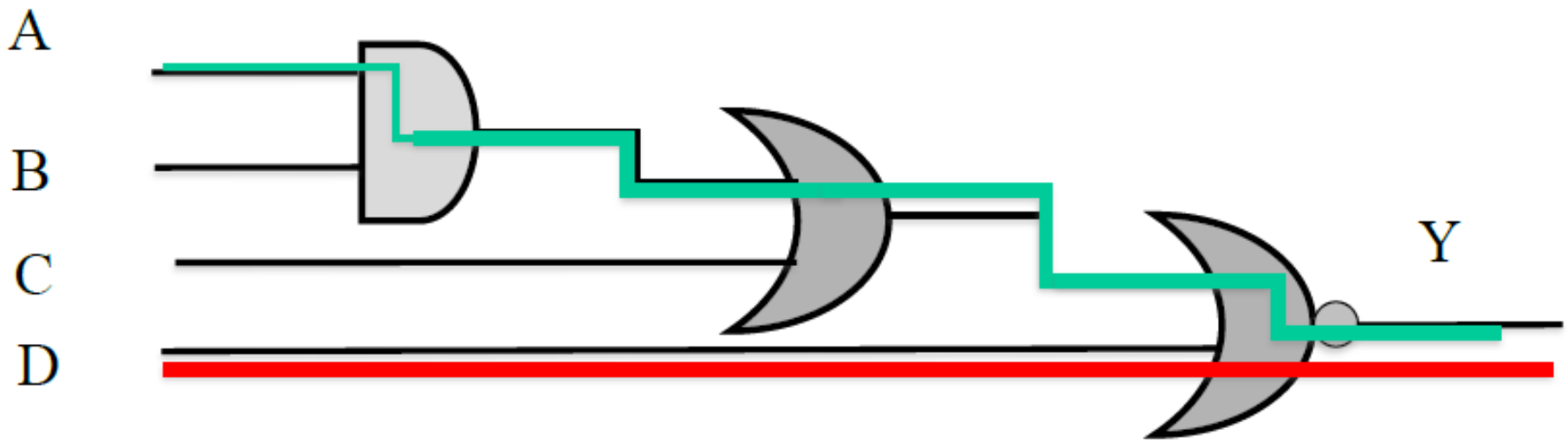
- Timing data of standard cells is provided in the Liberty format.
- Library:
  - General information common to all cells in the library.
  - For example, operating conditions, wire load models, look-up tables
- Cell:
  - Specific information about each standard cell.
  - For example, function, area.
- Pin:
  - Timing
  - Power
  - Capacitance
  - Leakage
  - Functionality

```
library (nameoflibrary) {
... /* Library level simple and complex attributes */

/* Cell definitions */
cell (cell_name) {
   ... /* cell level simple attributes */

   /* pin groups within the cell */
   pin(pin_name) {
       ... /* pin level simple attributes */

       /* timing group within the pin level */
       timing(){
            ... /* timing level simple attributes */ }
       ... /* additional timing groups */

   } /* end of pin */
   ... /* more pin descriptions */
} /* end of cell */
... /* more cells */

} /* end of library */
```



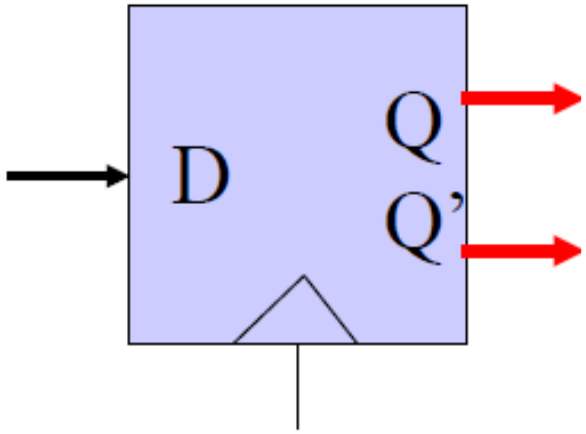
# Gates - Propagation Delays

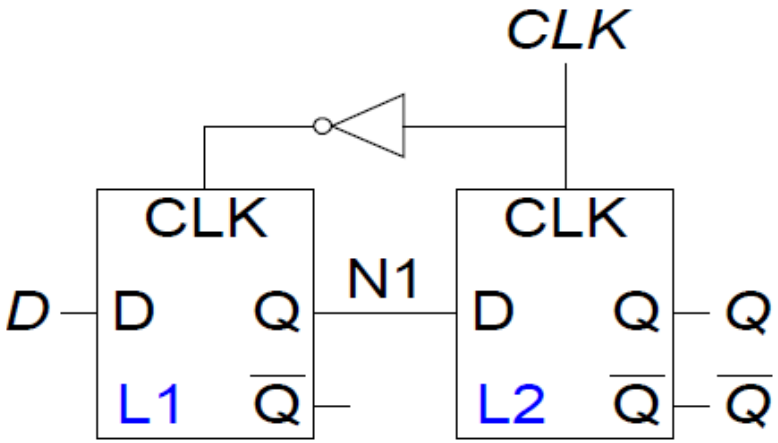
- Illustrate propagation delay in combinational gates
- Contamination delay: *tcd*
  - Minimum time from when an input changes until the output *starts* to change
- Propagation delay: *tpd*
  - Maximum time from when an input changes until the output *is* guaranteed to reach its final value (i.e., stop changing)



# Analyze Flops

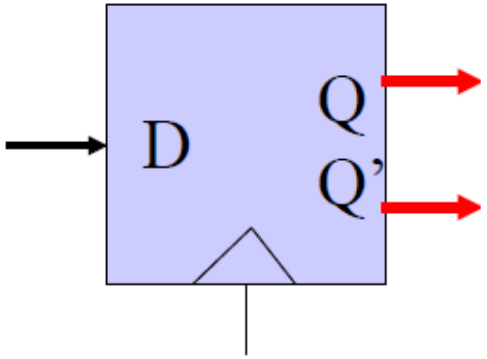
- Illustrate propagation delay, setup time, hold time
- Once a flip flop has been ‘built’ we are stuck with its timing characteristics:
- $t_{setup}$, $t_{hold}$ timing relation between D and CLK
- $t_{ccq}$, $t_{pcq}$ timing relation between CLK and Q
- No direct timing relation between input D and output Q

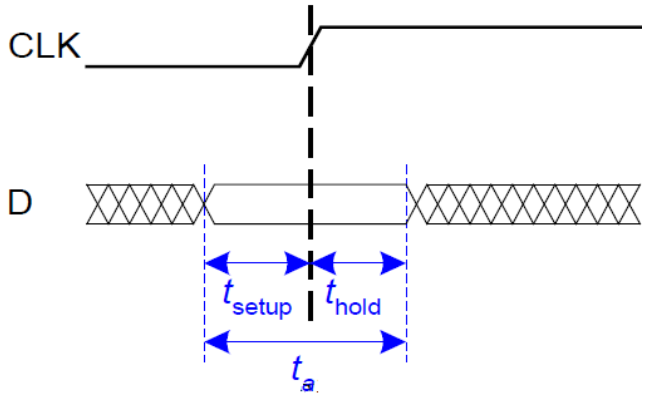


# Analyze Flops

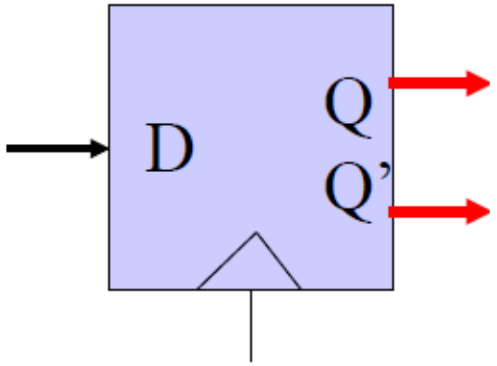
- Setup time: $t_{setup,}$ Time *before* the clock edge that data must be stable (i.e. not change)
- Hold time: $t_{hold,}$ Time *after* the clock edge that data must be stable
- Aperture time: $t_{a,}$ Time around clock edge that data must be stable ($t_a = t_{setup} + t_{hold}$)
- Setup time violation
  - This occurs if the input signal D does not settle to the stable value at least $t_{setup}$ *before* the clock edge.
- Hold time violation
  - This occurs if the input signal D does not remain unchanged for at least $t_{hold}$ *after* the clock edge.

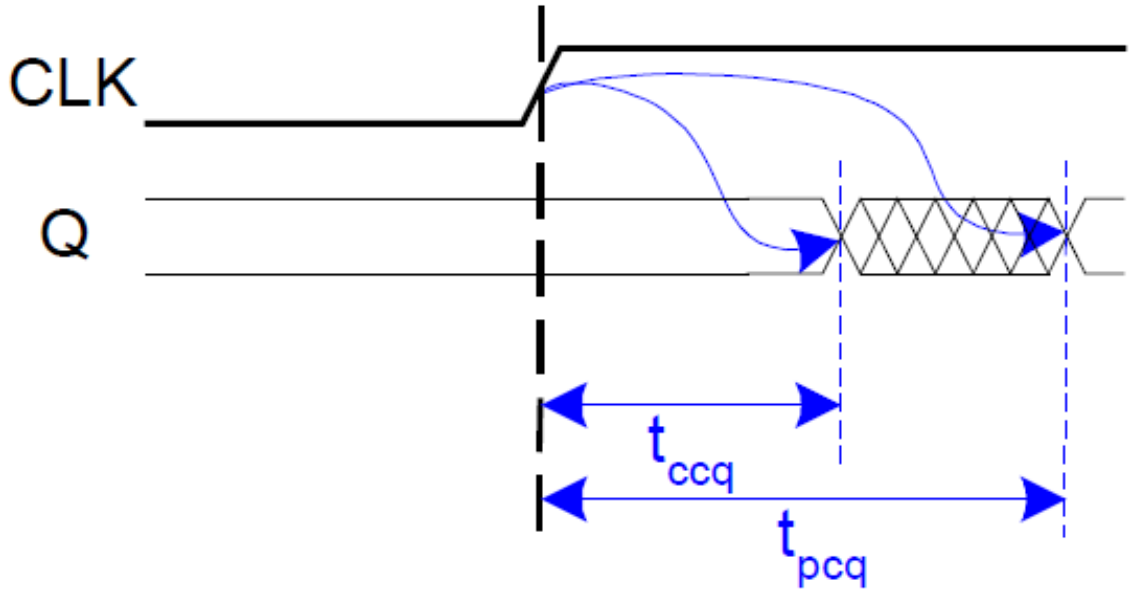


# Analyze Flops

- Propagation delay: $t_{pcq}$ = time after clock edge that the output $Q$ is guaranteed to be stable (i.e., to stop changing)
- Contamination delay: $t_{ccq}$ = time after clock edge that $Q$ might be unstable (i.e., start changing)

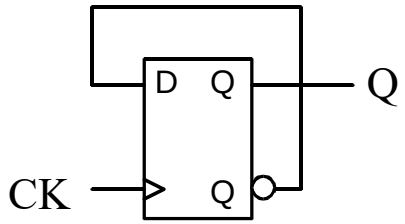


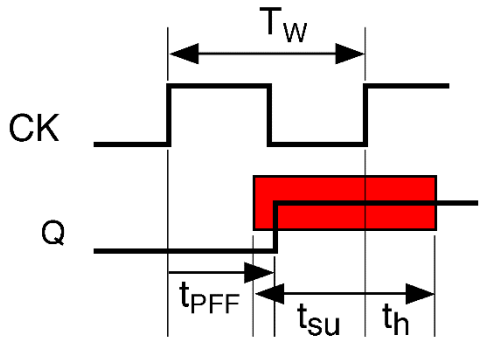



# Maximum Clock Frequency

- The clock frequency for a synchronous sequential circuit is limited by the timing parameters of its flip-flops and gates.
- This limit is called *the maximum clock frequency* for the circuit.
- The *minimum clock period* is the reciprocal of this frequency.
- Relevant timing parameters
  - Gates:
    - Propagation delays: min $t_{PLH}$, min $t_{PHL}$, max $t_{PLH}$, max $t_{PHL}$
  - Flip-Flops:
    - Propagation delays: min $t_{PLH}$, min $t_{PHL}$, max $t_{PLH}$, max $t_{PHL}$
    - Setup time: $t_{su}$
    - Hold time: $t_h$

- Example



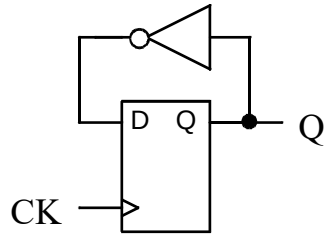

$$T_W \geq \max t_{PFF} + t_{su}$$

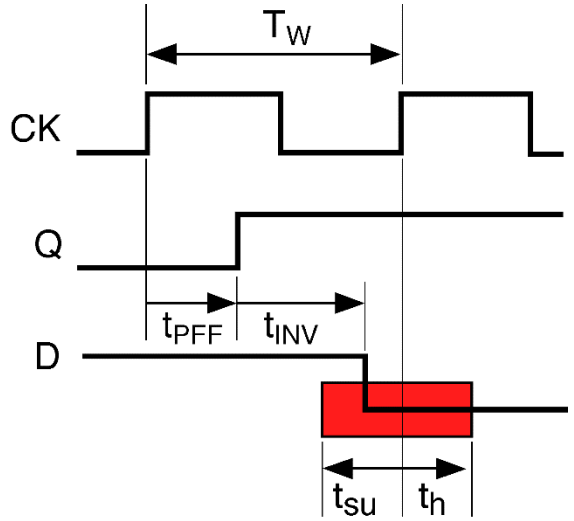
- Assume, max $t_{PLH}$ = 25ns, max $t_{PHL}$ = 40ns, $t_{su}$ = 20ns

$T_W \geq \max (\max t_{PLH} + t_{su,} \max t_{PHL} + t_{su})$

$T_W \geq \max (25+20, 40+20) = 60$

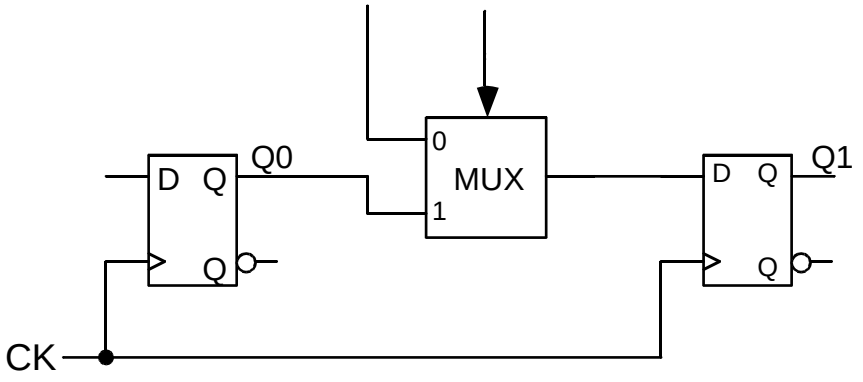
- Example

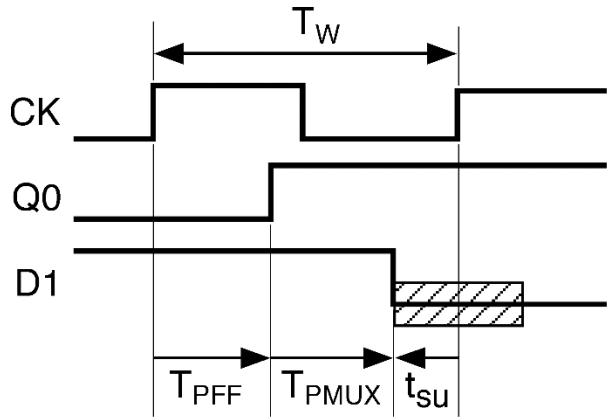



$$T_W \geq \max t_{PFF} + \max t_{PINV} + t_{su}$$

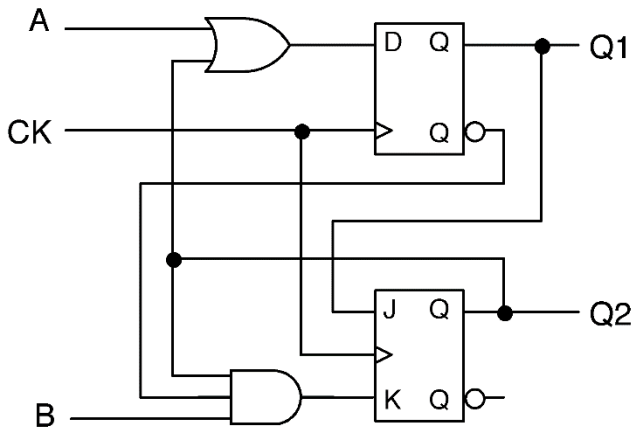
- Example




$$T_W \geq \max t_{PFF} + \max t_{PMUX} + t_{su}$$

- Example




| | $t_P$ | $t_{su}$ |
|---|---|---|
| D Flip-Flop: | 20 ns | 5 ns |
| JK Flip-Fllp: | 25 ns | 10 ns |
| AND Gate: | 12 ns | |
| OR Gate: | 10 ns | |

Paths from Q1 to Q1: None

Paths from Q1 to Q2: $T_W \geq \max t_{PDFF} + t_{JKsu} = 20 + 10 = 30$ ns

$T_W \geq \max t_{PDFF} + \max t_{AND} + t_{JKsu} = 20 + 12 + 10 = 42$ ns

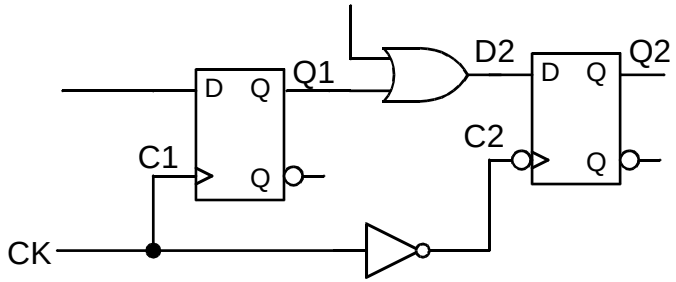
Paths from Q2 to Q1: $T_W \geq \max t_{PJKFF} + t_{OR} + T_{Dsu} = 25 + 10 + 5 = 40$ ns

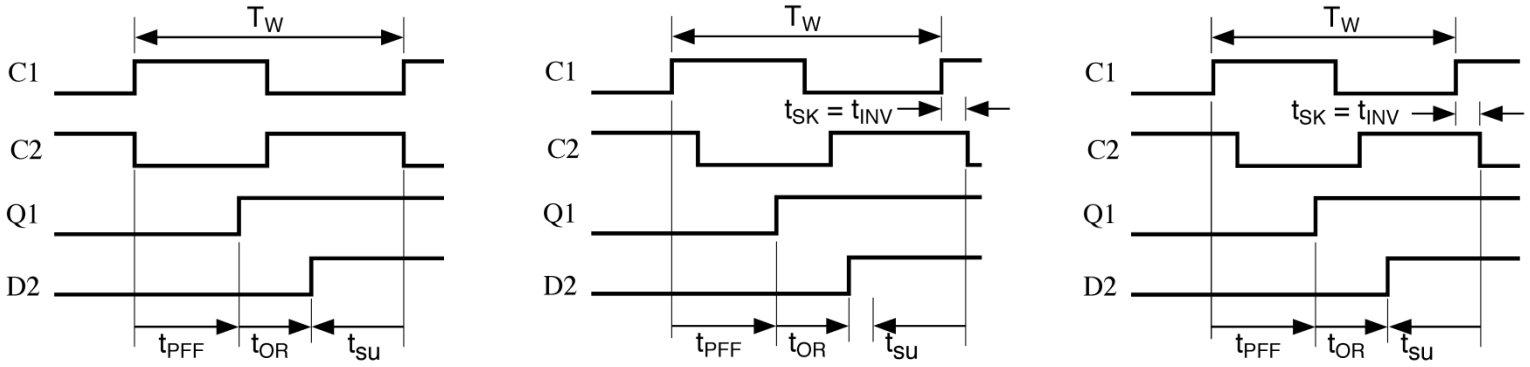
Paths from Q2 to Q2: $T_W \geq \max t_{PJKFF} + \max t_{AND} + t_{JKsu} = 25 + 12 + 10 = 47$ ns

$$T_W \geq 47 \text{ ns}$$

- Clock Skew
  - If a clock edge does not arrive at different flip-flops at exactly the same time, then the clock is said to be *skewed* between these flip-flops.
  - The difference between the times of arrival at the flip-flops is said to be the amount of *clock skew*.
  - Clock skew is due to different delays on different paths from the clock generator to the various flip-flops.
    - Different length wires (wires have delay)
    - Gates (buffers) on the paths
    - Flip-Flops that clock on different edges (need to invert clock for some flip-flops)
    - Gating the clock to control loading of registers

- Example (Effect of clock skew on clock rate)
  - Clock C2 skewed after C1

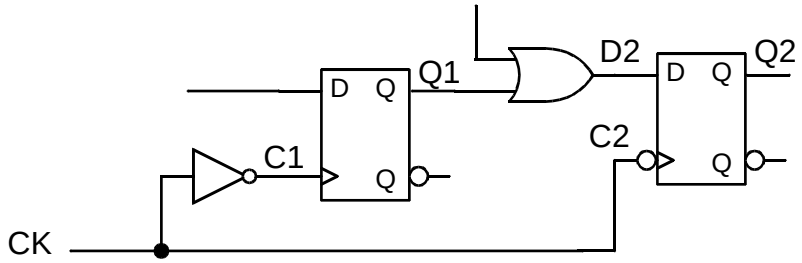


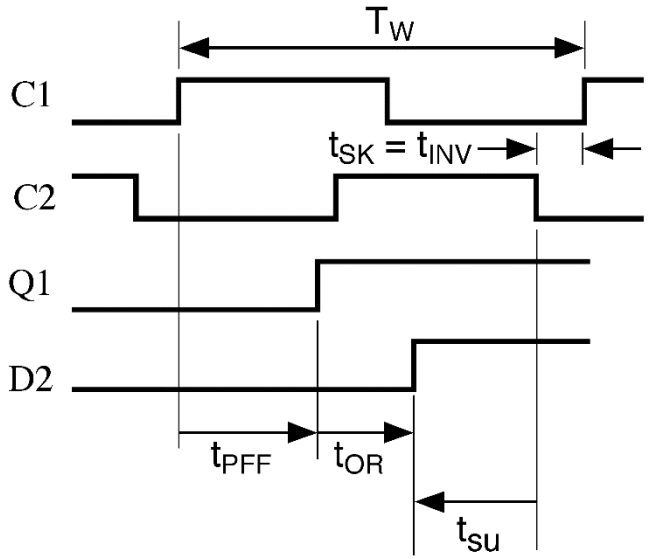




$T_W \geq \max T_{PFF} + \max t_{OR} + t_{su}$
(if clock not skewed, i.e., $t_{INV} = 0$)

$T_W \geq \max T_{PFF} + \max t_{OR} + t_{su} - \min t_{INV}$
(if clock skewed, i.e., $t_{INV} > 0$)

- Clock C1 skewed after C2

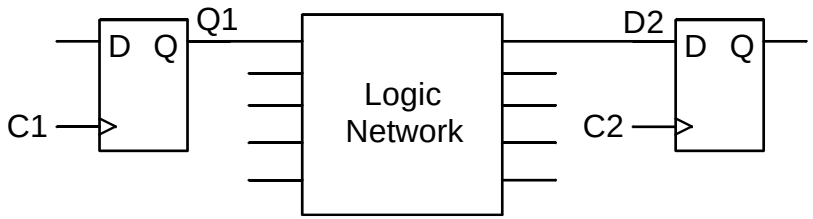


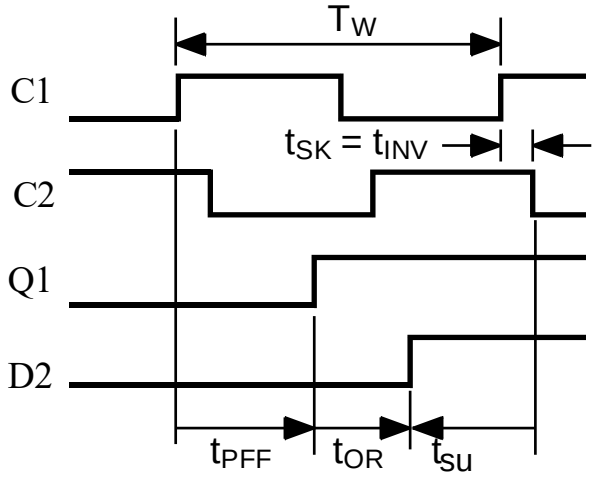



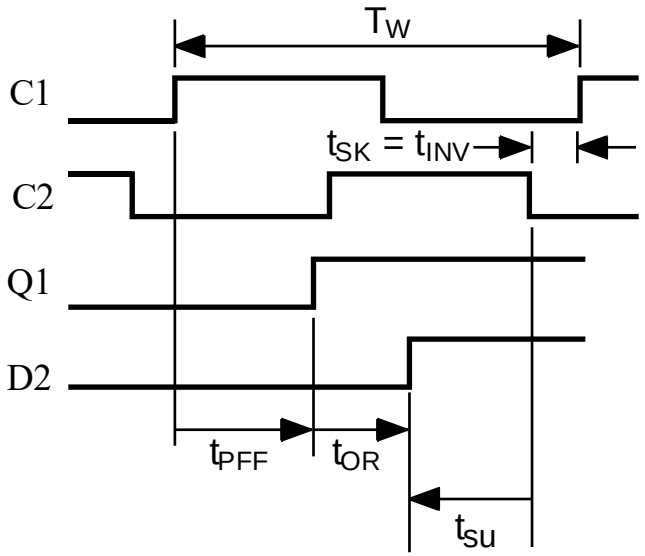

$T_W \geq \max T_{PFF} + \max t_{OR} + t_{su}$
(if clock not skewed, i.e., $t_{INV} = 0$)

$T_W \geq \max T_{PFF} + \max t_{OR} + t_{su} + \max t_{INV}$
(if clock skewed, i.e., $t_{INV} > 0$)

- Summary of maximum clock frequency calculations



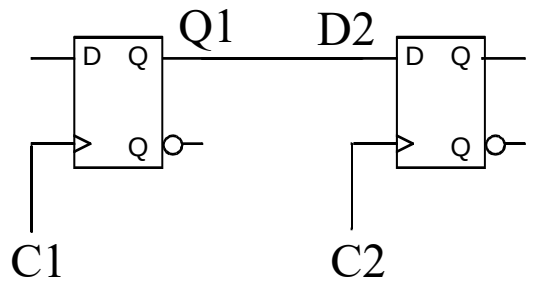




C2 skewed after C1: $T_W \geq \max T_{PFF} + \max t_{NET} + t_{su} - \min t_{INV}$

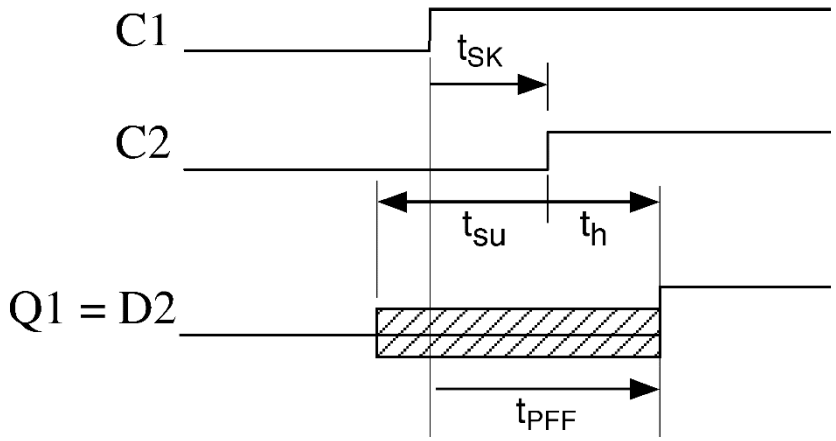
C2 skewed before C1: $T_W \geq \max T_{PFF} + \max t_{NET} + t_{su} + \max t_{INV}$

# Maximum Allowable Clock Skew

- How much skew between C1 and C2 can be tolerated in the following circuit?



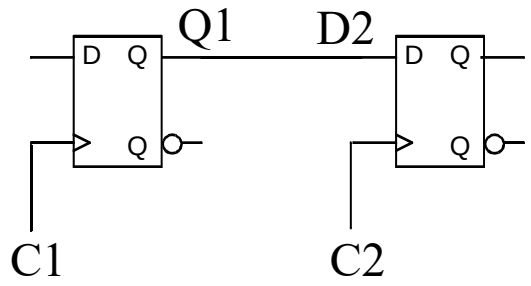

  - Case 1: C2 delayed after C1

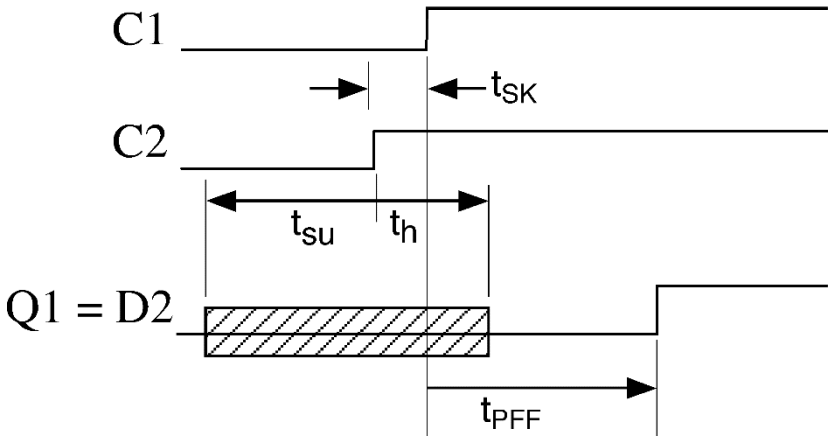



$$t_{PFF} > t_h + t_{SK}$$

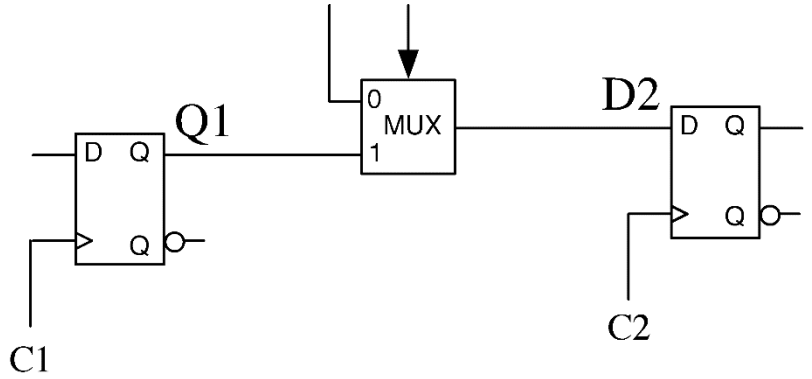
$$t_{SK} < \min t_{PFF} - t_h$$

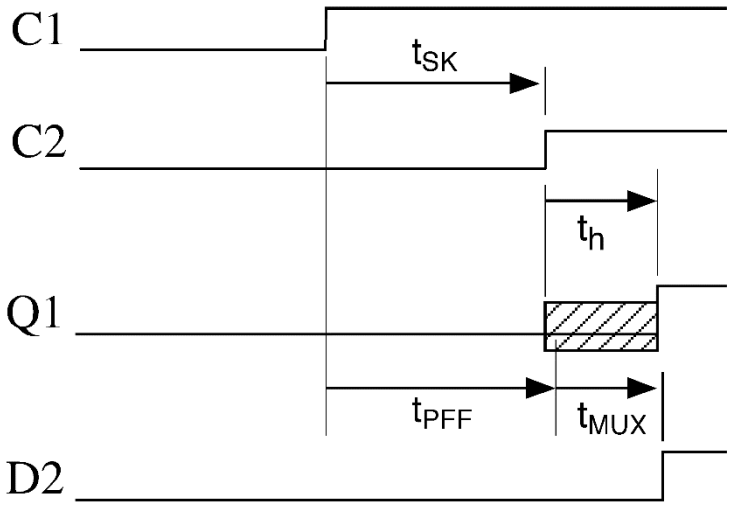
- Case 2: C1 delayed from C2



- How does additional delay between the flip-flops affect the skew calculations?

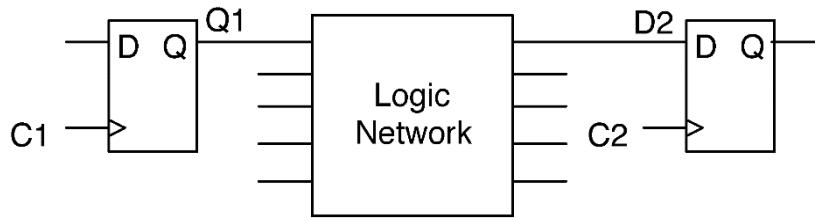


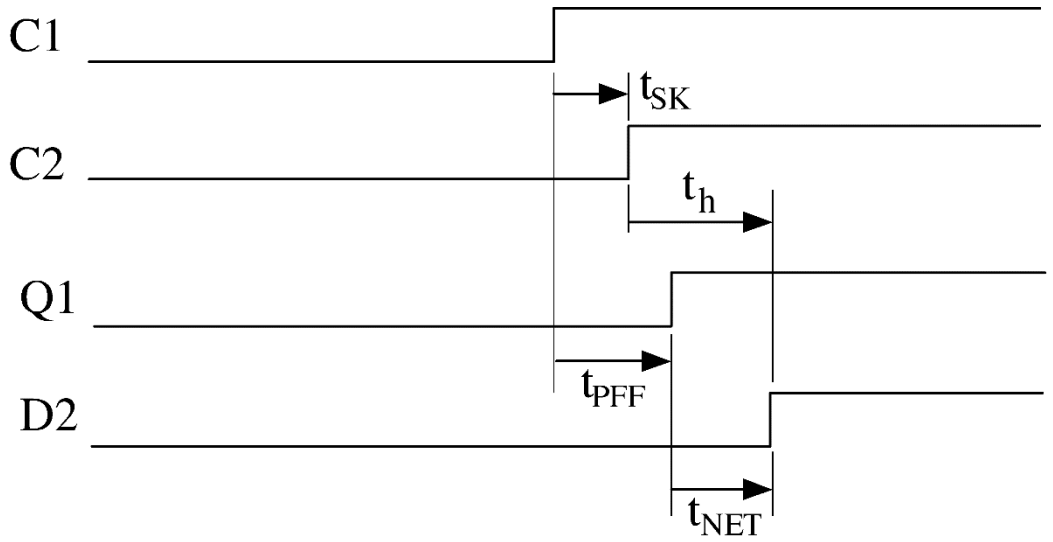




$$t_{SK} \leq \min t_{PFF} - t_h$$

$$t_{sk} \leq \min t_{PFF} + \min t_{MUX} - t_h$$

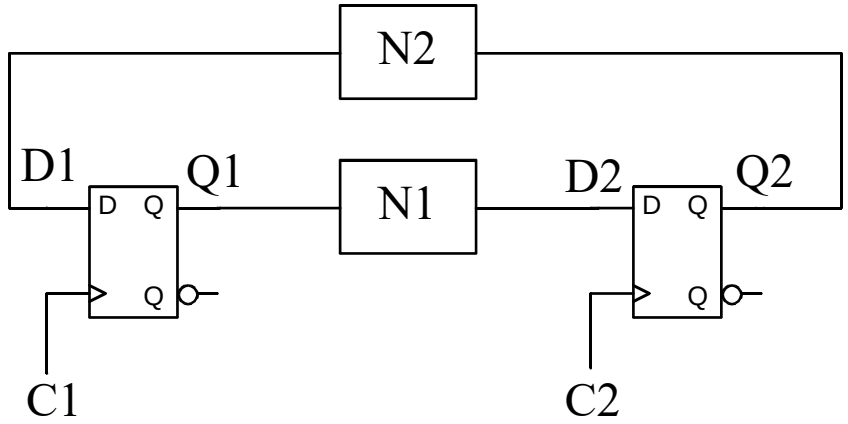
- Summary of allowable clock skew calculations

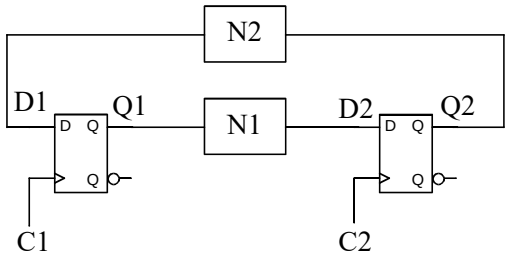



$$t_{SK} + t_h \leq t_{PFF} + t_{NET}$$
$$t_{SK} \leq \min t_{PFF} + \min t_{NET} - t_h$$

- Example: What is the minimum clock period for the following circuit under the assumption that the clock C2 is skewed after C1 (i.e., C2 is delayed from C1)?






- First calculate the maximum allowable clock skew.

$$t_{SK} < \min t_{PFF} + \min t_{N1} - t_h$$

- Next calculate the minimum clock period due to the path from Q1 to D2.

$$T_W > \max t_{PFF} + \max t_{N1} + t_{su} - \min t_{SK}$$

- Finally calculate the minimum clock period due to the path from Q2 to D1

$$T_W > \max t_{PFF} + \max t_{N2} + t_{su} + \max t_{SK}$$

$$T_W > \max t_{PFF} + \max t_{N2} + t_{su} + (\min t_{PFF} + \min t_{N1} - t_h)$$

$$T_W > \max t_{PFF} + \min t_{PFF} + \max t_{N2} + \min t_{N1} + t_{su} - t_h$$