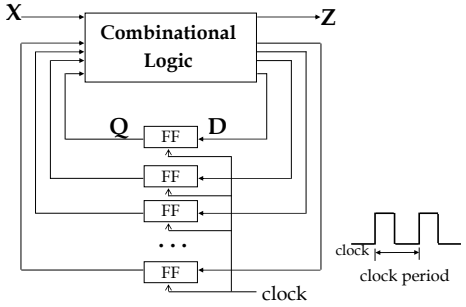
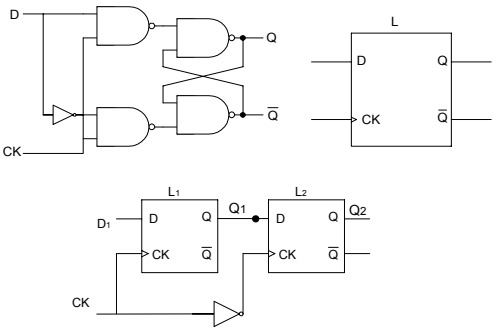
ISSUES ON TIMING AND CLOCKING

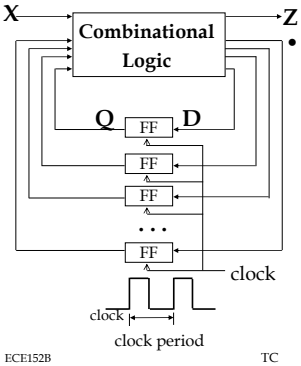
Issues on Timing and Clocking



Latch and Flip-Flop



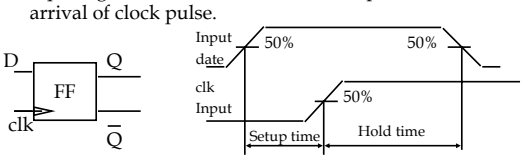
Clocking



- For correct operation of a synchronous circuit:
  - The clock period must be longer than the delay of the longest path in the combinational logic.
  - The width of the clock pulse must be long enough to allow the flip-flops to change state.

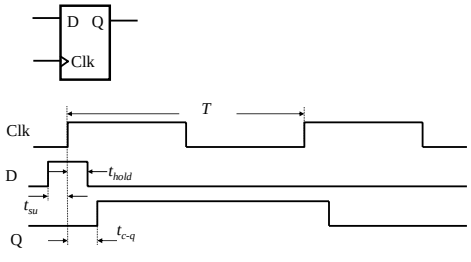
Flip-Flop Setup and Hold Times

- Flip-flop setup time T_{su}: the required time the data input signal value must be held stable prior to the arrival of clock pulse.



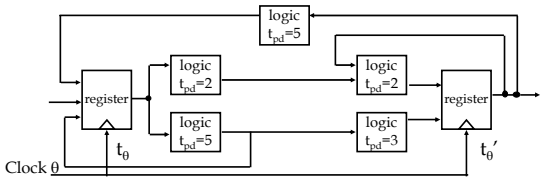
- Flip-flop hold time T_h: the required time the data input signal value must be held stable after the arrival of clock pulse.

Flip-flip Timing Parameters



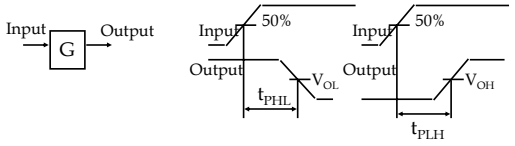
Delays can be different for rising and falling data transitions

Example: For the circuit shown below, assume the delay through the register (t_{pd}) is 0.6 and the delay through each logic block is indicated inside the box. Assume that the registers, which are positive edge-triggered, have a set-up time T_{su} of 0.4. What is the minimum clock period?

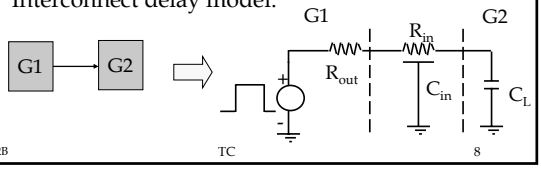


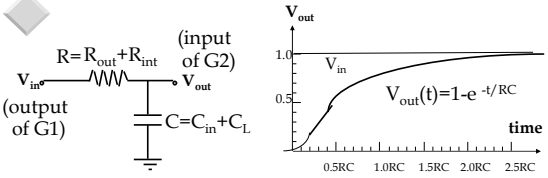
Delay Modeling

- Gate propagation delay: t_{PHL} & t_{PLH}

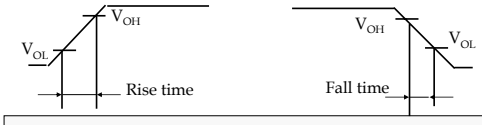


- Interconnect delay model:





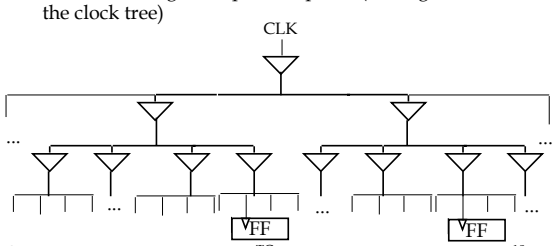
Rise time (Fall time): time for a waveform to rise from 10% to 90%(90% to 10%) of its steady state value



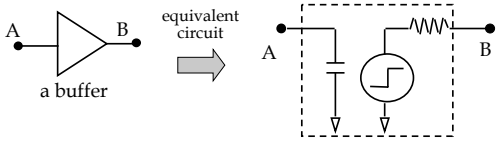
Rise time ≅ 2.2 RC =>
Rise time is proportional to the loading capacitance

Clock Tree

- If the # of flip-flops driven by the clock line is large, the clock rise time (also called slew rate) will be unacceptably long.
- Solution: Using clock power up tree (adding buffers into the clock tree)

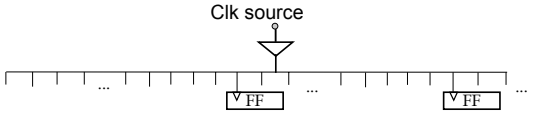


➔The load seen by the clock source is significantly reduced



An Example

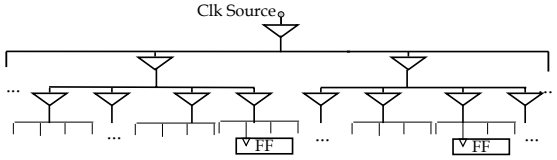
- Assume 64 flip-flips to be driven by a single clock source
- Buffer delay (with zero load): 0.2 ns
- Interconnect delay:
  - 0.2 ns with a single fanout (either to a flip-flop or to a buffer)
  - Addition 0.1ns delay for each additional fanout



- Total delay from clock source to clock ports of flip-flops: 6.9 ns
  0.2ns + 0.2ns + (0.2ns + 63 x 0.1ns) = 6.9ns

Example – Clock Tree

- Assume each buffer has four fanouts



- Total delay from clock source to clock ports of flip-flops: 2.3ns
  0.2ns {0th-level wire delay} + 0.2ns {1st-level buffer delay} +
  (0.2ns + 3 x 0.1ns) {1th-level wire delay} + 0.2ns {2nd-level buffer delay} +
  (0.2ns + 3 x 0.1ns) {2nd-level wire delay} + 0.2ns {3rd-level buffer delay} +
  (0.2ns + 3 x 0.1ns) {3rd-level wire delay}

Techniques for Improving Speed

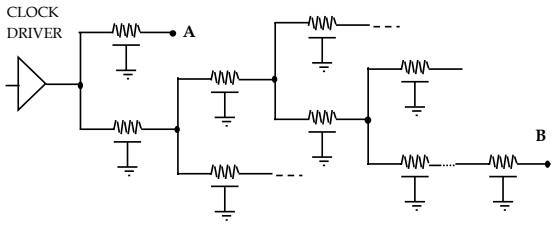
1. Keep the logic gate depth shallow between flip-flops.
2. Avoid circuit designs that have highly loaded gates in the critical path.
   - A gate delay will increase as the capacitive load is increased on the output of the gate.
   - The primary sources of load capacitance are routing capacitance and the input capacitance of the driven gates.
3. Duplicate logic to reduce fanouts.
4. Use low fan-in logic gates.
5. Avoid long interconnects

Clock Non-idealities

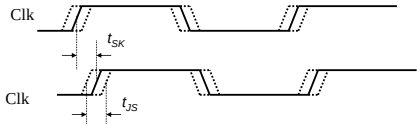
- **Clock skew**
  - Spatial variation in temporally equivalent clock edges; deterministic + random, t_{SK}
- **Clock jitter**
  - Temporal variations in consecutive edges of the clock signal; modulation + random noise
  - Cycle-to-cycle (short-term) t_{JS}
  - Long term t_{JL}
- **Variation of the pulse width**
  - Important for level sensitive clocking

Clock Skew

- The delays from the clock source to the clock inputs of different flip-flops are different

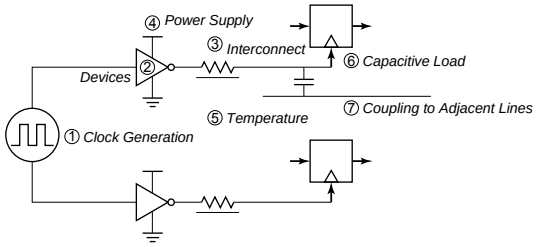


Clock Skew and Jitter



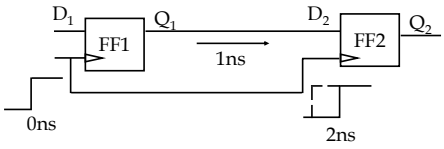
- Both skew and jitter affect the effective cycle time
- Only skew affects the race margin

Clock Uncertainties – Sources of Skew and Jitter



Sources of clock uncertainty

- The clock skew problem: the race problem

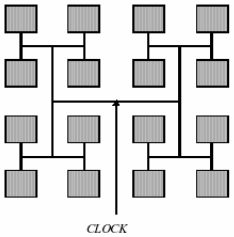


Correct operation: D_1 -> Q_1 , D_2 -> Q_2
Due to clock skew: D_1 -> Q_2 (error!!)

- Minimizing clock skew:

  Distribute the clock signal in such a way that the interconnections from the clock source to the FF's clock inputs are of equal length.

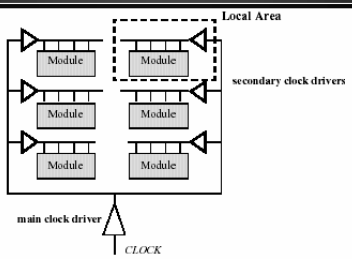
Clock Distribution



H-Tree Network

Observe: Only Relative Skew is Important

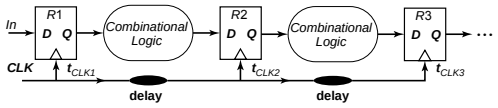
Clock Network with Distributed Buffering



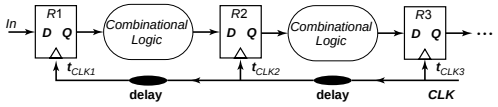
Reduces absolute delay, and makes Power-Down easier

Sensitive to variations in Buffer Delay

Positive and Negative Skew

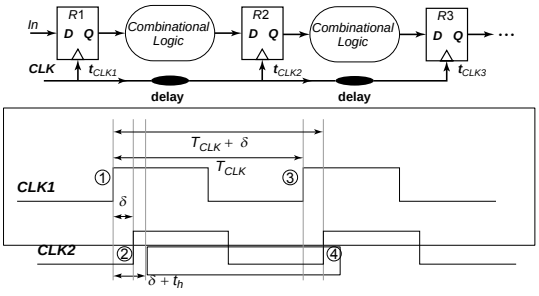


(a) Positive skew



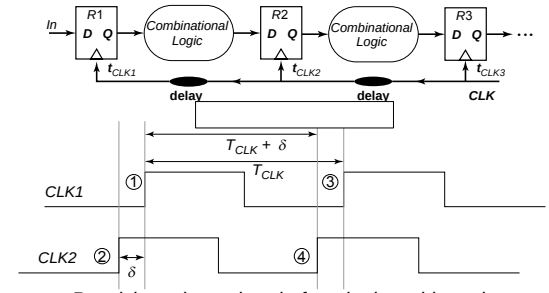
(b) Negative skew

Positive Skew



Launching edge arrives before the receiving edge

Negative Skew

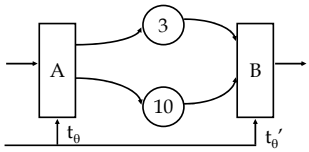


Receiving edge arrives before the launching edge

"Useful" Clock Skew

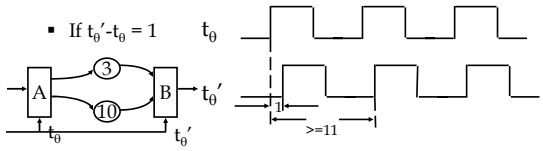
- Clock skew is not always bad!!

Example:

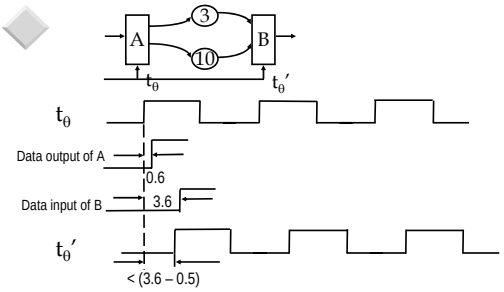


- Assume the FF propagation delay (clock to Q delay) $t_{c\to q}$=0.6, the FF setup time t_{su}=0.4, the FF hold time t_{hd}=0.5
- If t_θ'-t_θ = 0 (no clock skew),
  - minimum clock period = 11

- If t_θ'-t_θ = 1



- For proper operation, the time between positive edges at registers A and B must be greater than or equal to 11
  - ➔clock period + clock skew >= 11
  - ➔minimum clock period = 10



- On the other hand, the clock skew cannot exceed 3.1 ns.
- Otherwise the data latched into register A may propagate through the short path and reach the data input of register B before the rising edge of the clock pulse of the same cycle reaching θ'.

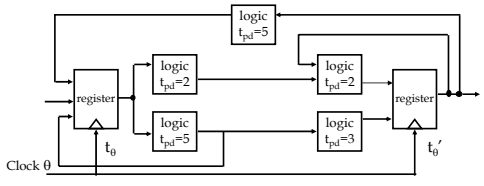
Summary

- Clock Period: T
- Longest delay from Reg. A to Reg. B: LD $_{A\to B}$
- Shortest delay from Reg. A to Reg. B: SD $_{A\to B}$
- FF propagation delay, setup time, hold time: $t_{c\to q}$, t_{su}, t_{hold}

$$T + (t_\theta' - t_\theta) \geq (t_{c\to q} + LD_{A\to B} + t_{su})$$
$$(t_\theta' - t_\theta) \leq (t_{c\to q} + SD_{A\to B} - t_{hold})$$

These requirements have to be satisfied for datapath between any pair of registers

Example: Assume $t_{c\to q}$ is 0.6, t_{su} is 0.4 and t_{hold} is 0.5.



(a) Determine the minimum clock period assuming a positive clock skew: $\delta = (t_\theta' - t_\theta) = 1$.

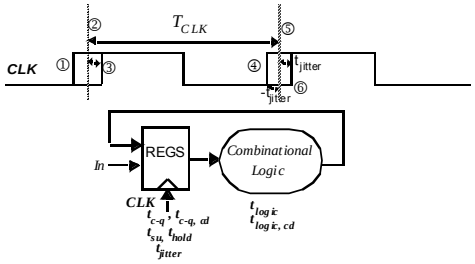
(b) Repeat part(a), factoring in a positive clock skew: $\delta = 3$.

(c) Repeat part(a), factoring in a negative clock skew: $\delta = -2$.

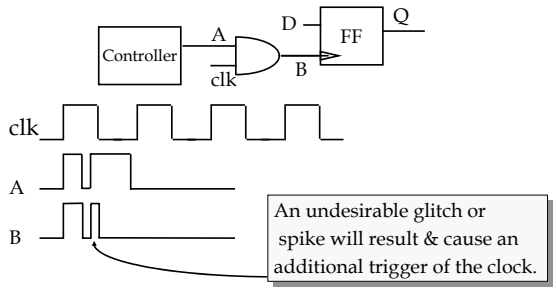
(d) Derive the maximum positive clock skew (i.e. $t_\theta' > t_\theta$) that can be tolerated before the circuit fails.

(e) Derive the maximum negative clock skew (i.e. $t_\theta' < t_\theta$) that can be tolerated before the circuit fails.

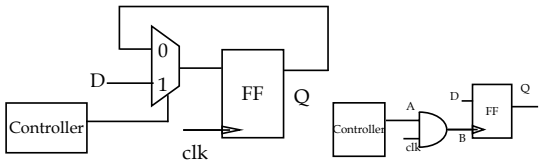
Impact of Jitter



Be Very Careful About Gated Clock



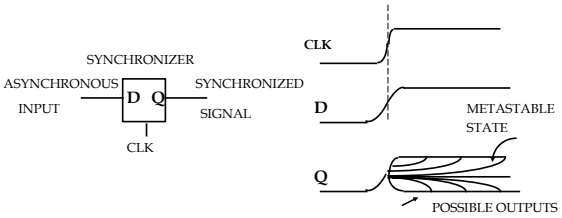
- An alternative design:



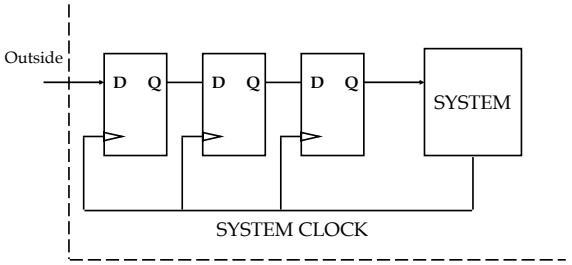
- This design causes less timing problem but consume more power.

Dealing with Asynchronous Inputs: Synchronizer

- Asynchronous signals incoming to a synchronous system must be synchronized with the rest of the system

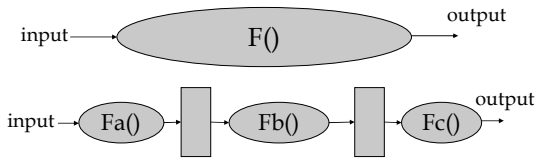


- A multiple-stage synchronizer reduces the chance of synchronization failure.



A Timing Optimization Technique Pipelining

- Pipelining: a technique to break up a timing-critical data path into a series of small data paths by placing registers between sections.



Another Timing Optimization Technique - Retiming

- Retiming: a technique to transform a given synchronous circuit into a faster circuit.

An example: A digital correlator:

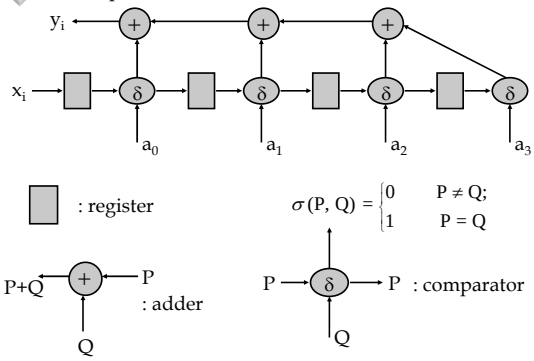
The correlator takes a stream of bits $x_0, x_1, \ldots x_k$ as input & compares it with a fixed-length pattern $a_0, a_1, \ldots, a_k$. After receiving each input x_i, the correlator produces as output the number of matches.

I.e.

$$y_i = \sum_{j=0}^{k} \sigma(x_{i-j}, a_j)$$

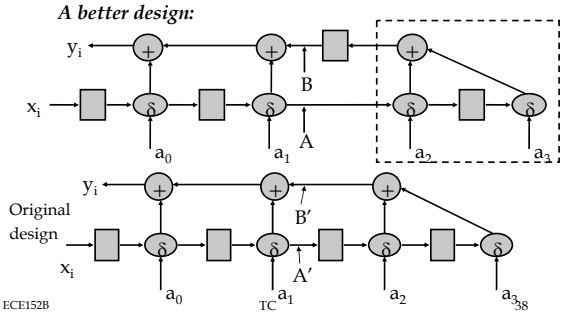
where $\sigma(x, y) = \begin{cases} 1 & \text{if } x = y; \\ 0 & \text{otherwise} \end{cases}$

An implementation for k=3



- Suppose each adder has a propagation delay of 7 ns. & each comparator 3ns.
  - ➔ The longest propagatim delay is 24ns
  - ➔ The clock period must be >= 24ns.

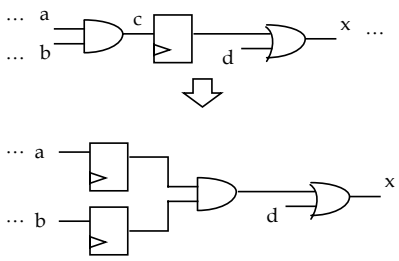
A better design:



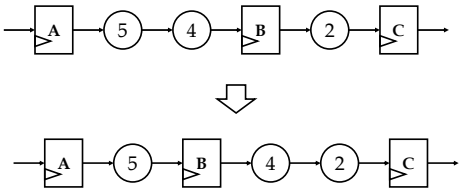
- These two designs are functionally equivalent:
  - all input signals to the box portion arrive one clock tick earlier.
  - thus, the boxed portion performs the same sequence of computation as the first design, but one clock tick earlier.
  - Since the output from the boxed portion is delayed one clock tick by the new register at B, the remainder of the circuit sees the same behavior as in the 1st design.
- The longest propagation delay is reduced to 17 nsec.
- The elements in the boxed portion "lead" by one clock tick.

Retiming - the technique of inserting & deleting registers to speed the design while preserve the function.

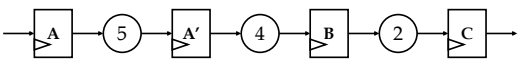
Retiming Transformation



Example:

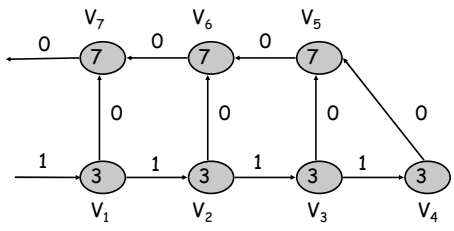


Comparison: pipelining



Retiming

Edge label: # of registers



Retiming

Edge label: # of registers