EE141-Spring 2012 Digital Integrated Circuits

Lecture 22
Timing + Clocks

Administrativia

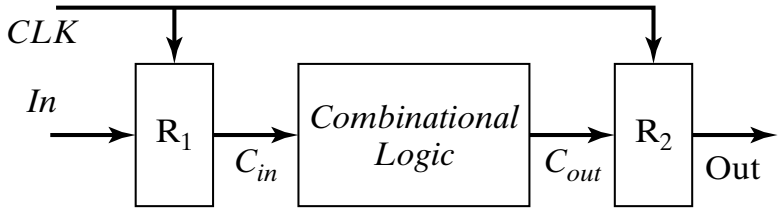
- Project Phase 2 due Today.
- Project Phase 3 to be launched today
- Assignment 9 posted today
  - One more assignmnet (#10) – will not be graded

Class Material

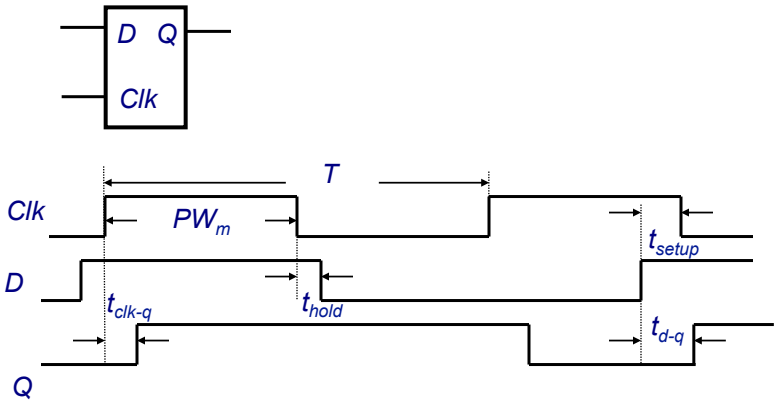
- Last lecture
  - Registers
- Today's lecture
  - Timing
- Reading (Ch 10)

Timing

Synchronous Timing

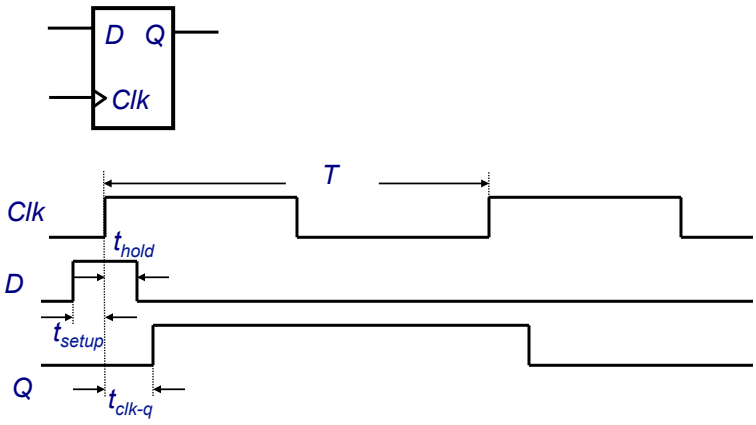


Latch Parameters



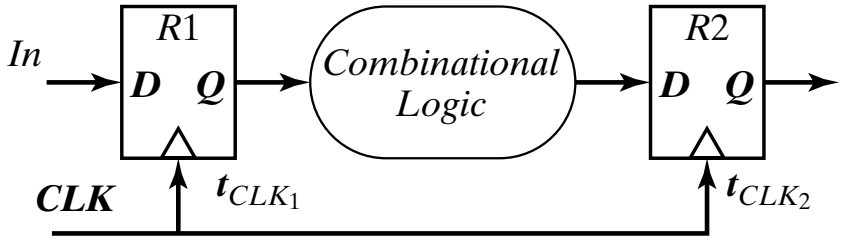
Delays can be different for rising and falling data transitions

Register Parameters



Delays can be different for rising and falling data transitions

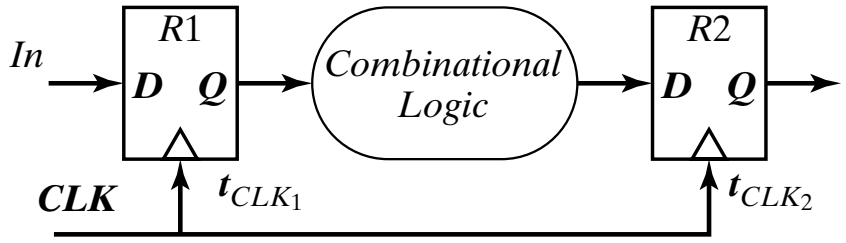
Timing Constraints



t_{clk-q}
$t_{clk-q,min}$
t_{setup}, t_{hold}

t_{logic}
$t_{logic,min}$

Timing Constraints



$t_{clk\text{-}q}$
$t_{clk\text{-}q,min}$
t_{setup}, t_{hold}

t_{logic}
$t_{logic,min}$

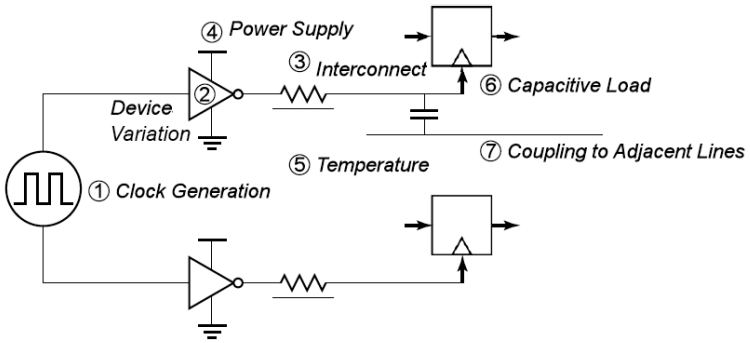
Cycle time (max): $T_{Clk} > t_{clk\text{-}q} + t_{logic} + t_{setup}$

Race margin (min): $t_{hold} < t_{clk\text{-}q,min} + t_{logic,min}$

Clock Nonidealities

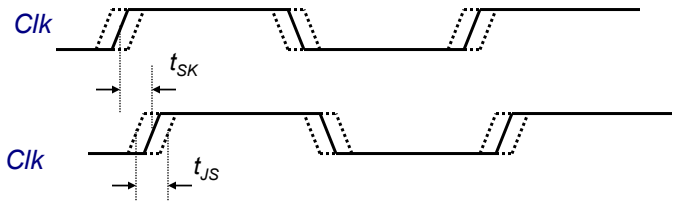
- **Clock skew**
  - Spatial variation in temporally equivalent clock edges; deterministic + random, t_{SK}
- **Clock jitter**
  - Temporal variations in consecutive edges of the clock signal; modulation + random noise
  - Cycle-to-cycle (short-term) t_{JS}
  - Long term t_{JL}
- **Variation of the pulse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

Clock Uncertainties



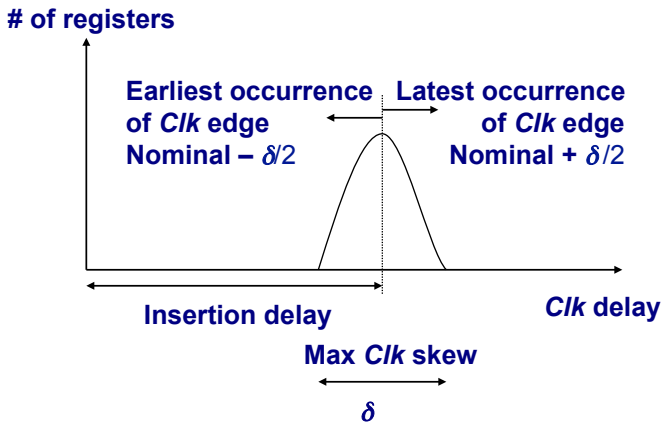
Sources of clock uncertainty

Clock Skew and Jitter

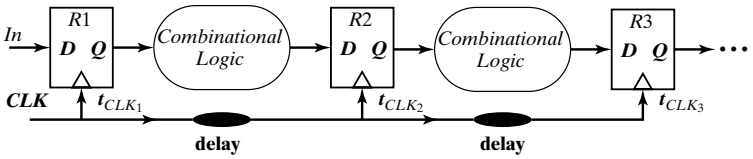


- Both skew and jitter affect the effective cycle time and the race margin

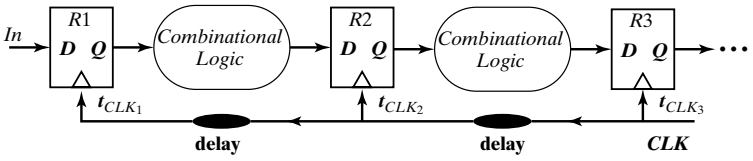
Clock Skew



Positive and Negative Skew

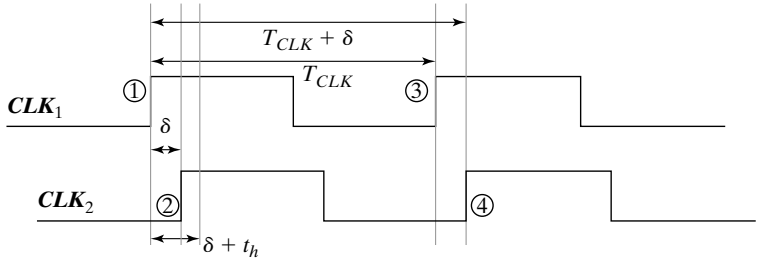


(a) Positive skew



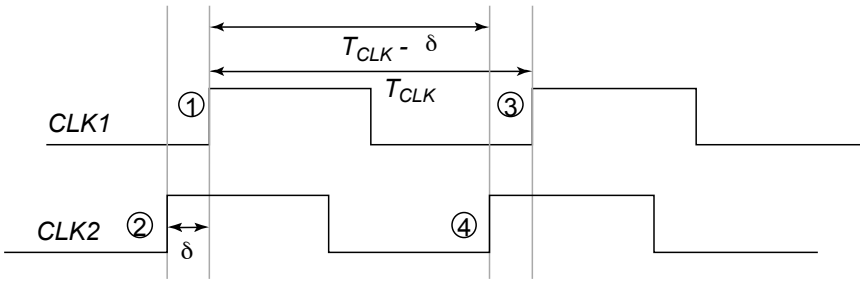
(b) Negative skew

Positive Skew



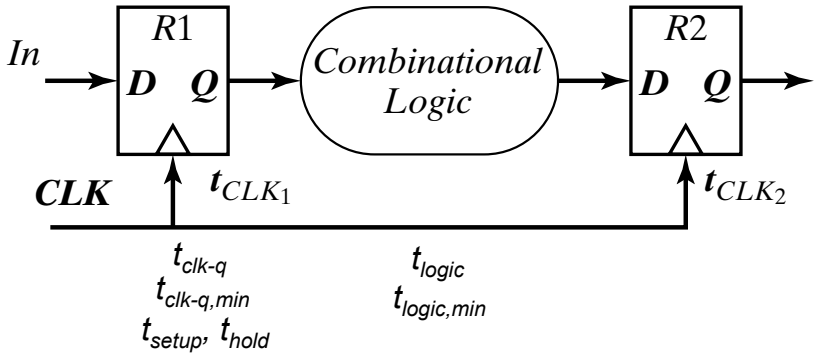
Launching edge arrives before the receiving edge

Negative Skew



Receiving edge arrives before the launching edge

Timing Constraints

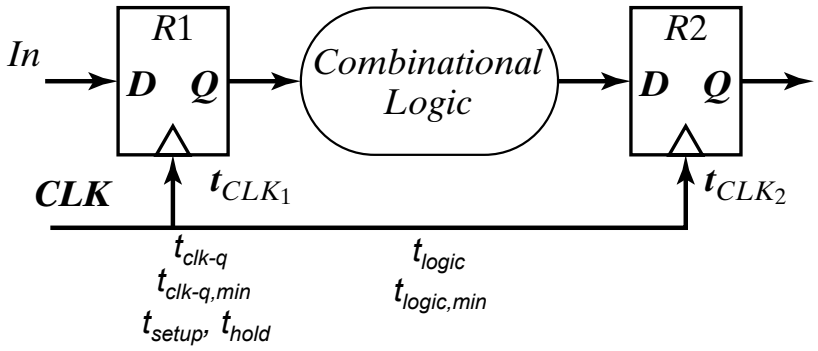


Minimum cycle time:

$T_{clk} + \delta = t_{clk\text{-}q} + t_{setup} + t_{logic}$

Worst case is when receiving edge arrives early (negative δ)

Timing Constraints

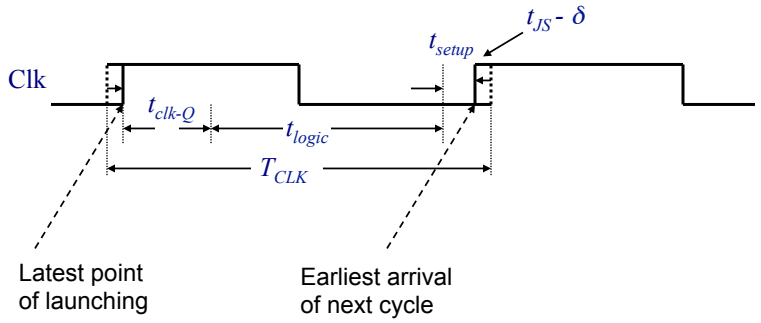


Hold time constraint:

$t_{(clk\text{-}q,min)} + t_{(logic,min)} > t_{hold} + \delta$

Worst case is when receiving edge arrives late
Race between data and clock

Longest Logic Path in Edge-Triggered Systems



Clock Constraints in Edge-Triggered Systems

If launching edge is late and receiving edge is early, the data will not be too late if:

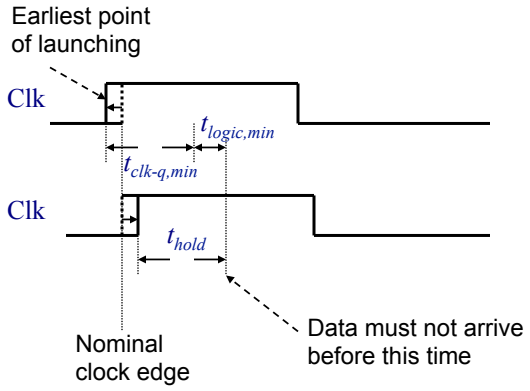
$$t_{clk-q} + t_{logic} + t_{setup} < T_{CLK} - t_{JS,1} - t_{JS,2} + \delta$$

Minimum cycle time is determined by the maximum delays through the logic

$$t_{clk-q} + t_{logic} + t_{setup} - \delta + 2t_{JS} < T_{CLK}$$

Skew can be either positive or negative

Shortest Path



Clock Constraints in Edge-Triggered Systems

If launching edge is early and receiving edge is late:

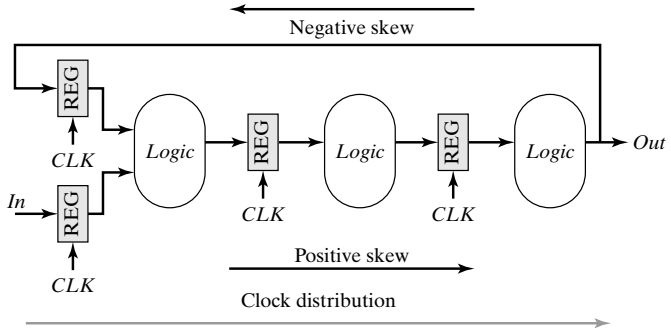
$$t_{clk\text{-}q,min} + t_{logic,min} - t_{JS,1} > t_{hold} + t_{JS,2} + \delta$$

Minimum logic delay

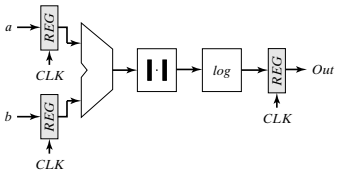
$$t_{clk\text{-}q,min} + t_{logic,min} > t_{hold} + 2t_{JS} + \delta$$

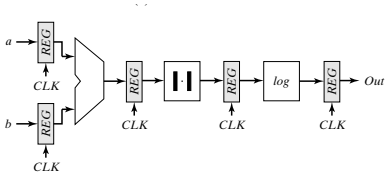
(This assumes jitter at launching and receiving clocks are independent – which usually is not true)

Datapath with Feedback



Pipelining



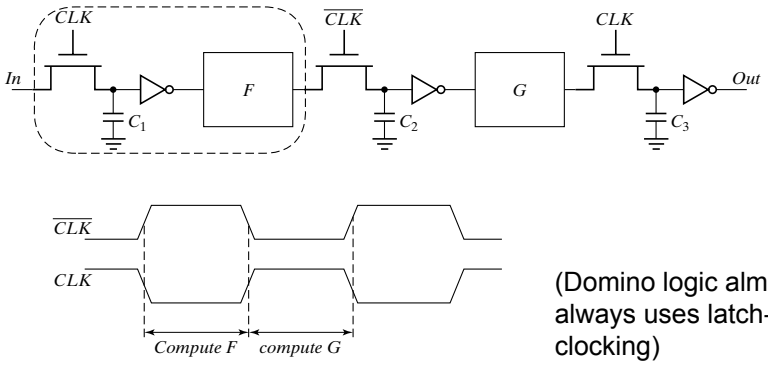


Reference

Clock Period	Adder	Absolute Value	Logarithm
1	$a_1 + b_1$		
2	$a_2 + b_2$	$\lvert a_1 + b_1 \rvert$	
3	$a_3 + b_3$	$\lvert a_2 + b_2 \rvert$	$\log(\lvert a_1 + b_1 \rvert)$
4	$a_4 + b_4$	$\lvert a_3 + b_3 \rvert$	$\log(\lvert a_2 + b_2 \rvert)$
5	$a_5 + b_5$	$\lvert a_4 + b_4 \rvert$	$\log(\lvert a_3 + b_3 \rvert)$

Pipelined

Latch-Based Clocking

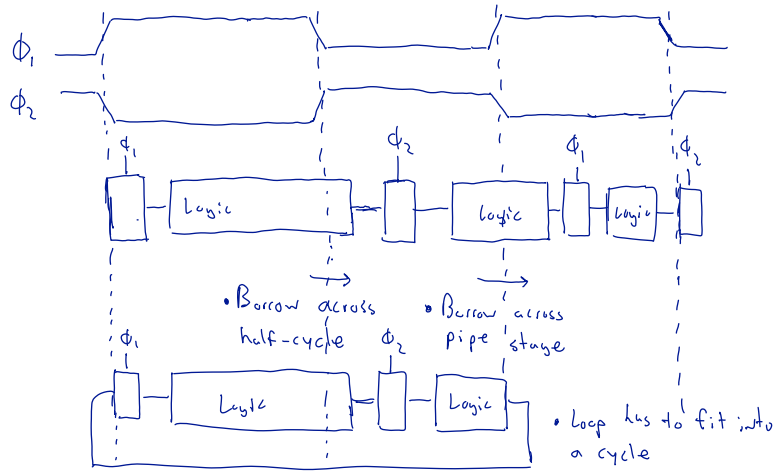


(Domino logic almost always uses latch-based clocking)

Latch vs. Flip-flop

- In a flip-flop based system:
  - Data launches on one rising edge
    - And must arrive before next rising edge
  - If data arrives late, system fails
    - If it arrives early, wasting time
  - Flip-flops have hard edges
- In a latch-based system:
  - Data can pass through latch while it is transparent
  - Long cycle of logic can borrow time into next cycle
    - As long as each loop finished in one cycle

Time Borrowing Example



Latch vs. Flip-flop Summary

- Flip-flops generally easier to use
  - Most digital ASICs designed with register-based timing
- But, latches (both pulsed and level-sensitive) allow more flexibility
  - And hence can potentially achieve higher performance
  - Latches can also be made more tolerant of clock un-certainty
  - More in EE241