# Timing Design in Digital Systems

# Outline

1. Timing design in Synchronous (clocked) Logic
   - Min/Max timing with flip-flops
   - Latch-based design
2. Timing Issues in CMOS circuits
3. Timing verification Flow
4. Techniques to Improve Performance

# Course “Mantras”

- One clock, one edge, Flip-flops only
- Design BEFORE coding
- Behavior implies function
- Clearly separate control and datapath

# Mantra #1

One clock, one edge; Flip-flops only

- For your design (at least for each module) use one clock source and only one edge of that clock
- Only use edge-triggered flip-flops

Why?

- Moving data between different clock domains requires careful timing design and synthesis “scripting”

If you need multiple clocks in your design

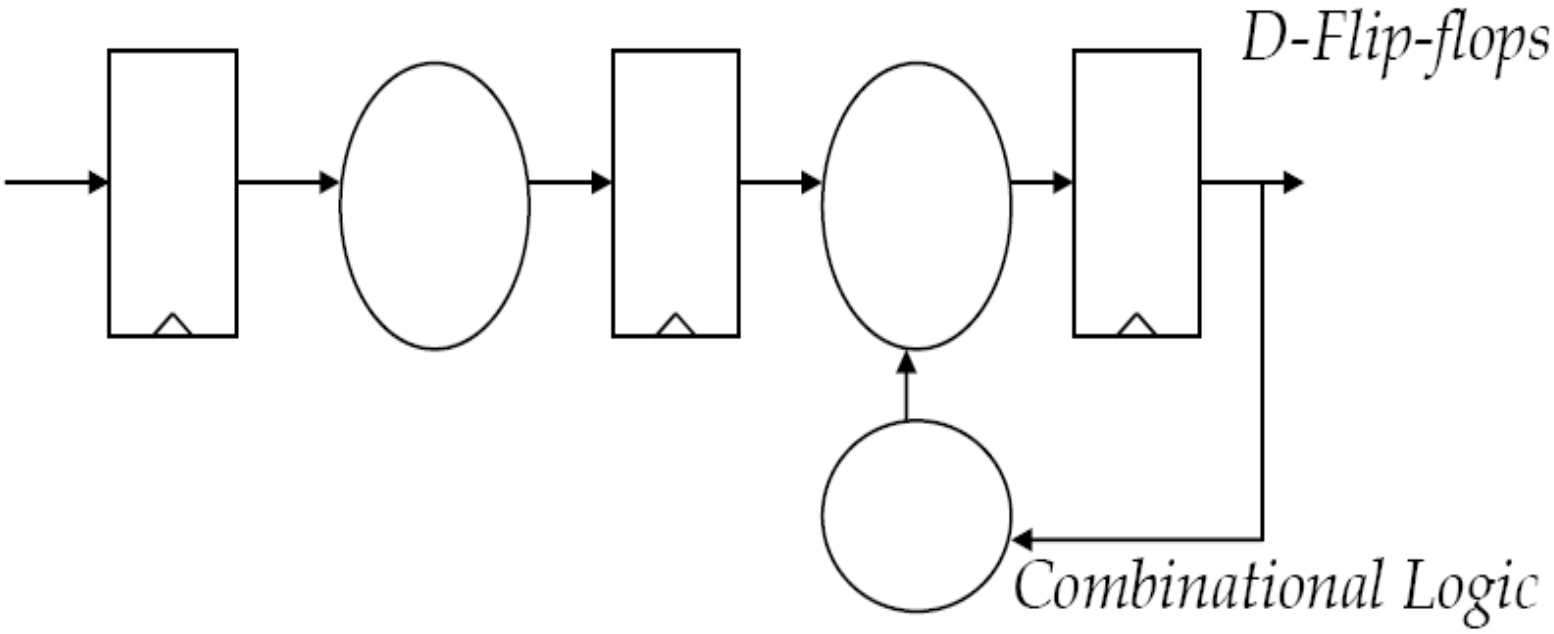
- Make them related by a powers of 2
  - E.g 50, 100 and 200 MHz
- Consider one clock per module
- Consider resynchronizing using flip-flops between clock domains

Caveat

- Tools and designers are getting better at using latches and multi-phase clocks. However, this requires some experience to get correct.

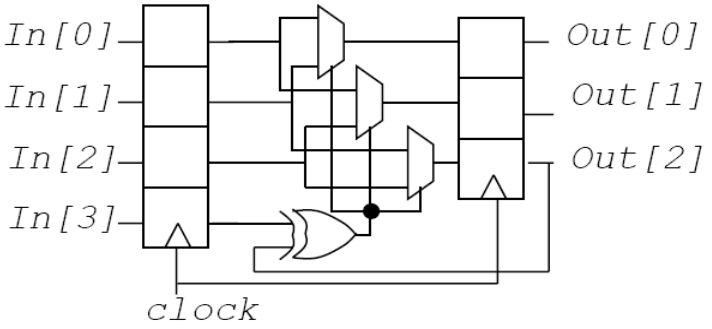
# General Approach to Timing Design

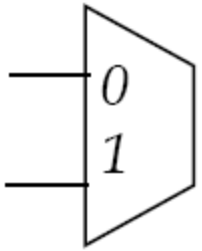
- In general, all signals start and end in registers every clock period

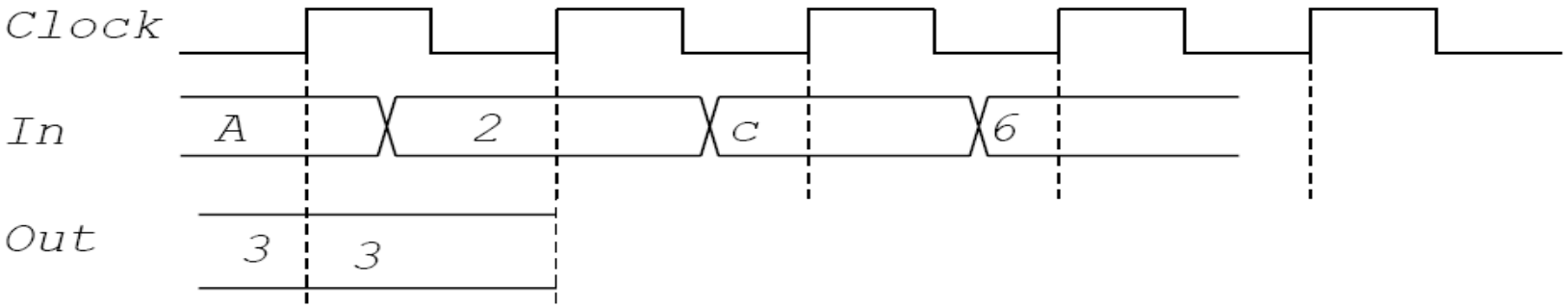


# Clock Level Timing

Example (Revision):



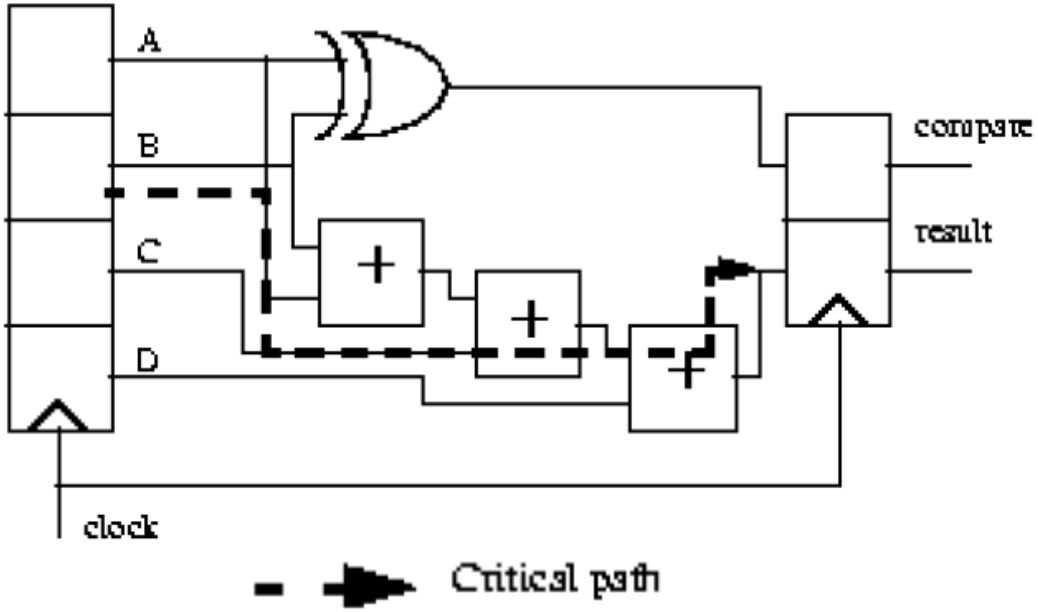



# Critical Path

Thus, the clock speed is determined by the slowest feasible path between registers in the design

- *Often referred to as "the critical path"*




Critical path is longer with increased *logic depth* (# gates in series)

# Synchronous Clock Distribution

The goal of clock tree is for the clock to arrive at every leaf node at the same time:

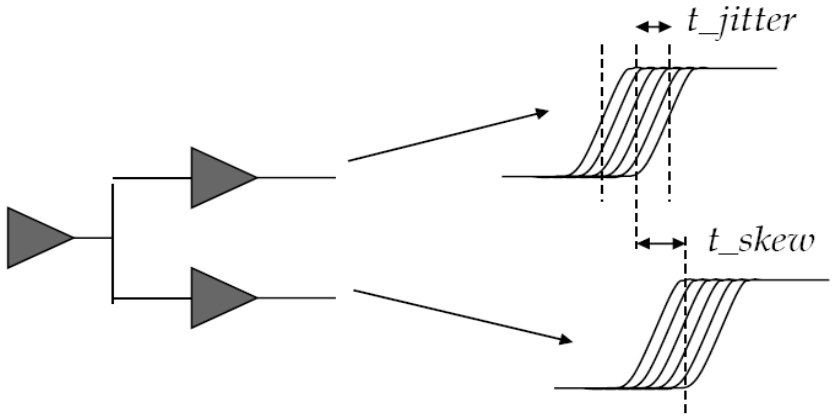
Usually designed after synthesis: Matched buffers; matched capacitance loads

Common design method:

- “H tree”
- Clock tree done after design
- Balance RC delays
- Balance buffer delays

# Clock skew and jitter

- Clock skew = systematic clock edge variation between sites
  - Mainly caused by delay variations introduced by manufacturing variations
  - Random variation
- Clock jitter = variation in clock edge timing between clock cycles
  - Mainly caused by noise



# Comments on Clock Skew

- ASIC design relies on automatic clock tree synthesis
  - Works to guarantee a global skew target
- Custom clock distribution can be used to add the following features to a clock:
  - Smaller skews
  - Local skews < Global skew
  - Multiple non-overlapping clock phases
  - Deliberate non-random skew at flip-flop/latch level
  - Future automatic clock tree synthesis tools might include features like this

# Flip-Flop based design

*Edge triggered D-flip-flop*

- Q becomes D after clock edge

*Set-up time:*

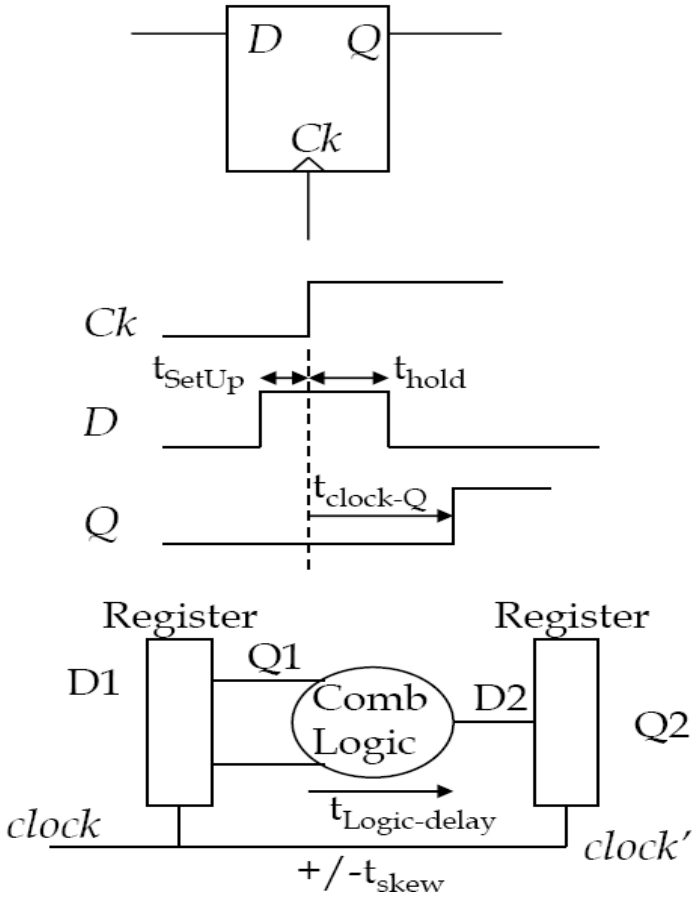
- Data can not change no later than this point before the clock edge.

*Hold time:*

- Data can not change during this time after the clock edge.

*t_clock-Q*

- Delay on output (Q) changing from positive clock edge



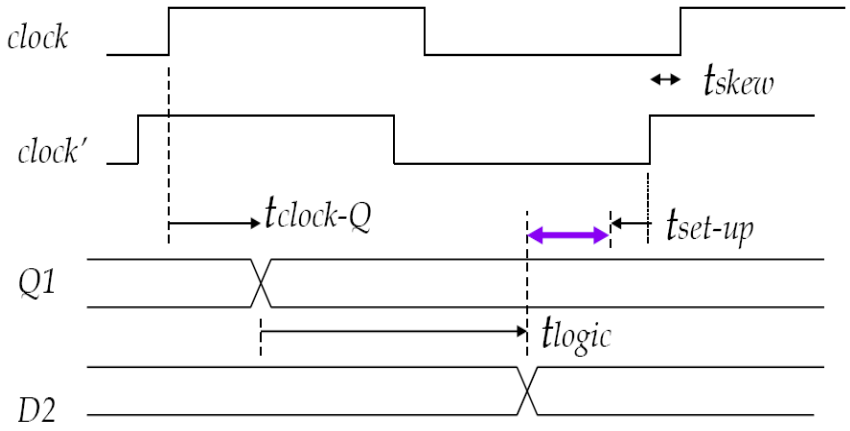
# Preventing Set-Up Violations

Set-up violation:

Logic is too slow for the correct logic value to arrive at the inputs to the register on the right before one set-up time before the clock edge

Constraint to prevent this:

$$t_{clock} \geq t_{clock-Q-\max} + t_{\log ic-\max} + t_{set-up} + t_{skew}$$



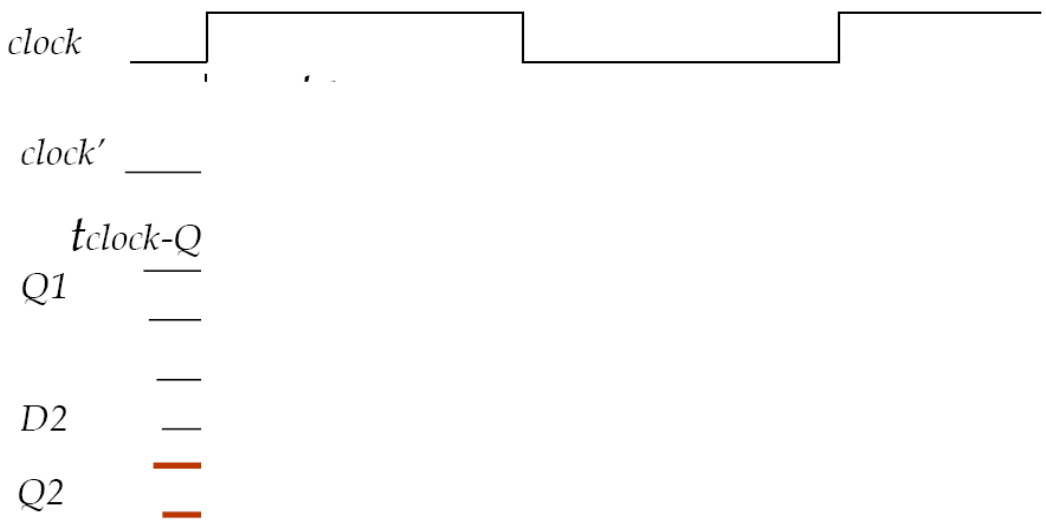

The amount of time required to turn '>' into '=' is referred to as **timing slack**

# Preventing hold Violations

Hold violations occur when race-through is possible

Constraint to prevent hold violations:

$$t_{hold} + t_{skew} \leq t_{clock-Q-\min} + t_{logic-\min}$$




sometimes have to insert additional logic to prevent hold violations

# Latch Based Design

*D-latch*

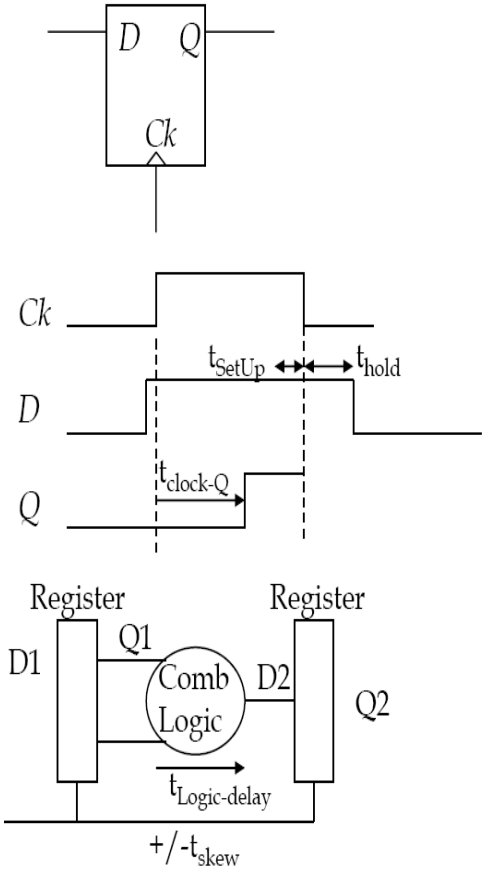
- Q follows D while clock is high ("transparent")
- Value on D when clock goes low is stored on Q

*Set-up and hold times:*

- D can not change close to the falling ('latching') clock edge.

*t_clock-Q*

- Delay from clock going high to Q changing



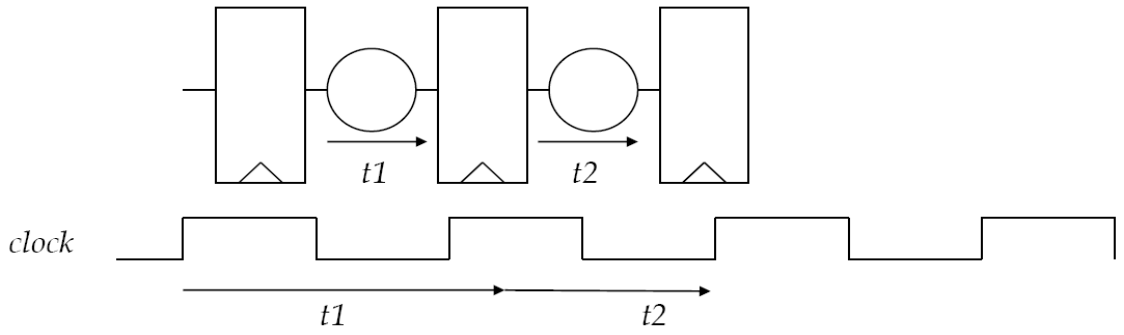
# Latch Timing Constraints

- Set-up constraints under nominal design same as for flip-flop

$$t_{clock} \geq t_{clock-Q-\max} + t_{logic-\max} + t_{set-up} + t_{skew}$$

- "Transparency" of latch can be used to improve flexibility of timing

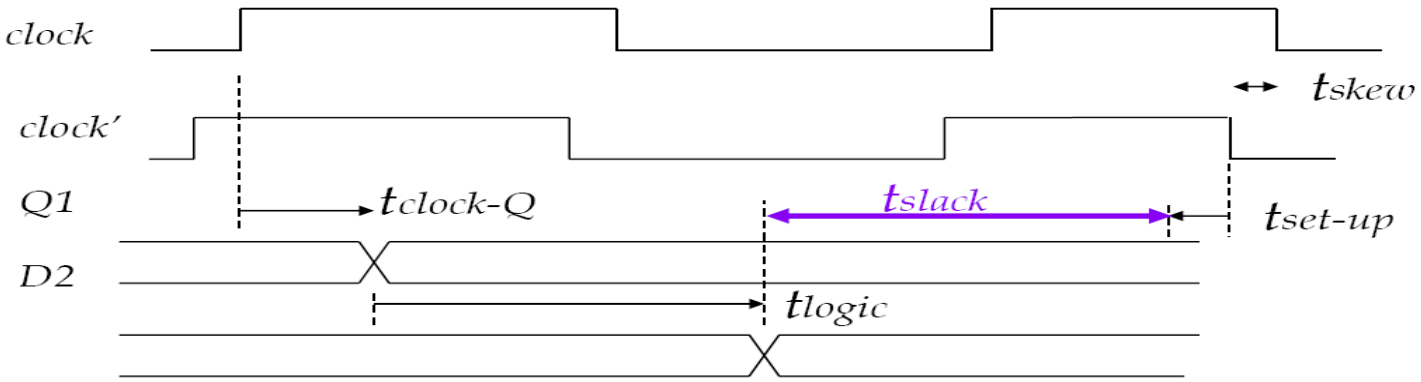
  e.g.: If critical path is in logic block 1:



# Latch Timing Constraints WITH cycle stealing

To prevent set-up violations:

$$t_{clock} + t_{clock-high-\max} \geq t_{clock-Q-\max} + t_{\log ic-\max} + t_{set-up} + t_{skew}$$




Notes:

- The percentage of time the clock is high is referred to as the *duty-cycle*
- If part of the following clock-high time is used to allow this logic to be slower, then the logic-block connected to Q2 must be proportionally faster
- Using the clock-high time like this is called *cycle-stealing*

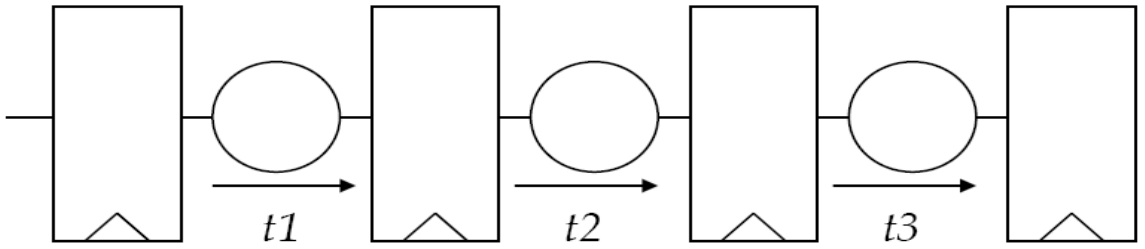
# Latch Set Up Violations

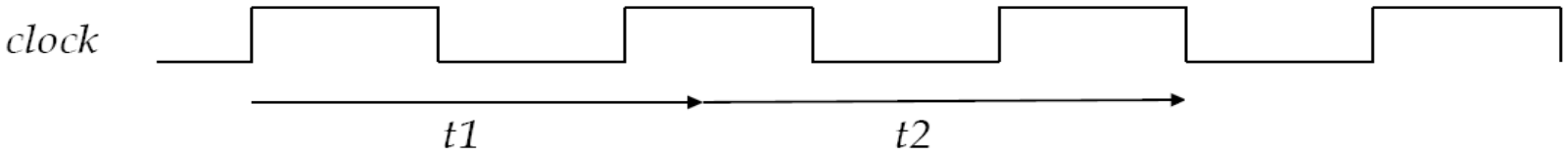
Notes:

- The percentage of time the clock is high is referred to as the *duty-cycle*
- If part of the following clock-high time is used to allow this logic to be slower, then the logic-block connected to Q2 must be proportionally faster
- Using the clock-high time like this is called *cycle-stealing*
  - Normally cycle stealing is not enabled

# Latches … Cycle Stealing

- Can use up to a total of t_clock_high within a pipeline structure, to help in timing closure
  - Example:



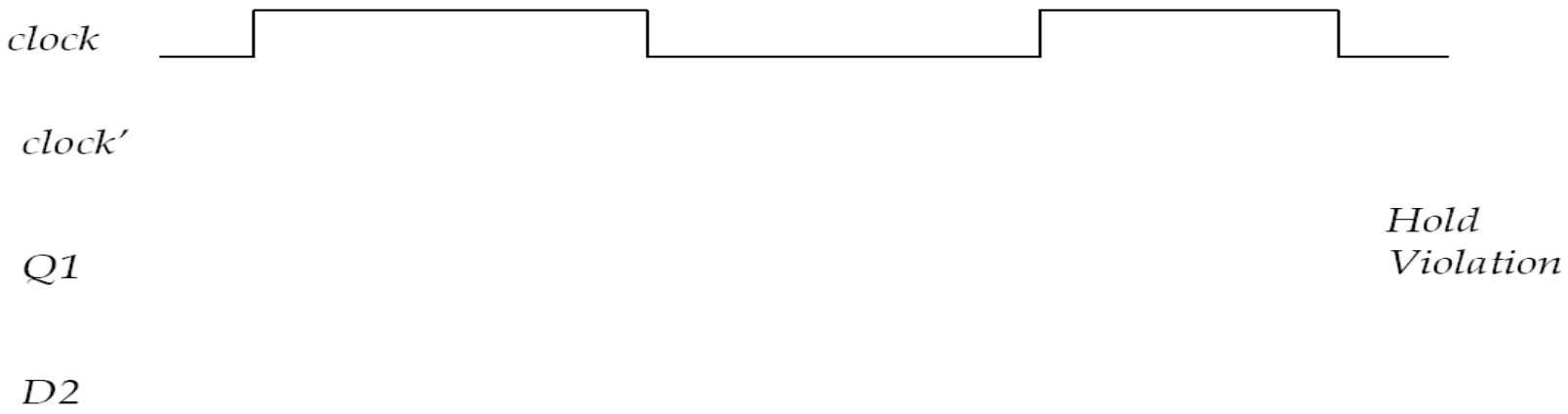

*t1,t2 > tclock (cycle stealing)*

*What can t3 be?*

# …Latch timing constraints

To prevent hold violations:

$$t_{clock-high-\max} + t_{hold} + t_{skew} \leq t_{clock-Q-\min} + t_{\log ic-\min}$$



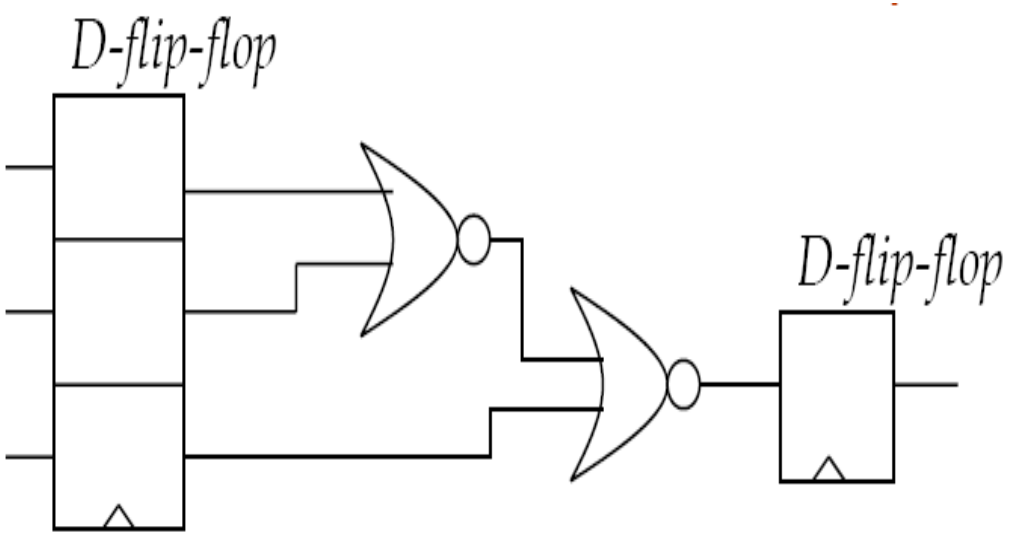

*Note:*

- Hold violations are harder to prevent in latch-based designs

# Revision – So far

- What is a set-up violation?
- How is a set-up violation fixed?
- What is a hold-violation?
- How is a hold-violation fixed?
- Why are edge-triggered flip-flops preferred over latches?

# Example:




min : typ : max

Tclock-Q = 3 : 4 : 5

TNOR = 1 : 2 : 3

Tsu_max = 1

Thold_max = 2

Tskew = 1 ns

If this is the critical path, what is the fastest clock frequency?

Is there potential for a hold violation?

# CMOS Drive Strength

Revision: CMOS transistors operating in the linear region:

$$I_{ds} = \beta((V_{gs} - V_t)V_{ds} - V_{ds}^2 / 2$$

$$\text{where } \beta = (\mu\varepsilon / t_{ox})(W / L)$$

where is the transistor width, and is the channel length

i.e. To a first approximation, $I_{ds} \approx V_{ds} / R_{on}$ $\boxed{R_{on} \approx 1/\beta(V_{GS}\text{-}V_T)}$

*Thus, delay in CMOS circuits depends largely on W/L of the drive transistor* and the capacitance of the load it is driving. That capacitance consists of:

- Input gates of cells being driven, and
- Capacitance of wiring

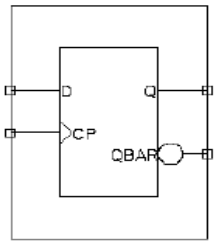
$$\boxed{\tau = R_{on}C_{load}}$$

# Cell Library Example

AμE Advanced Microelectronics
A Division of ITD

**ms080cmosxCells**
CMOSX 0.8 Micron
Standard Cell Library

**D flip-flop** **DFF**




**Logic Equation:**
Q = [(D & CP|rise|) | (Q' & !CP|rise|)]
QBAR = !Q

**Truth Table**

| CP | Q | QBAR |
|---|---|---|
| 01 | D | !D |
| 10 | Q | Q&F |

**Size in microns (W x H):**
63.4 x 45.0

**Pin Capacitance (fF)**

| pin | best | typical | worst |
|---|---|---|---|
| CP | 14.7 | 21.6 | 30.9 |
| D | 10.4 | 13.5 | 18.4 |
| Q | 5.01 | 10.6 | 11.7 |
| QBAR | 11.4 | 12.7 | 22.3 |

**Delay Information**

| Info | Timing | | best, 5.5V, -55C, load:0.35pF tr:0.504ns, tf:0.638ns | | | typical, 5V, 25C, load:0.35pF tr:1.15ns, tf:1.12ns | | | worst, 4.5V, 125C, load:0.35pF tr:2.78ns, tf:2.27ns | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 0.25 * load | 1 * load | 4 * load | 0.25 * load | 1 * load | 4 * load | 0.25 * load | 1 * load | 4 * load |
| cp->qbar | 01->10 | PD | 0.209 | 0.315 | 0.647 | 0.553 | 0.579 | 1.14 | 1.52 | 1.76 | 2.61 |
| | | | 0.213 + 0.308 * CL | | | 0.517 + 0.478 * CL | | | 1.46 + 0.831 * CL | | |
| | | TR | 0.0957 | 0.206 | 0.811 | 0.144 | 0.354 | 1.23 | 0.281 | 0.668 | 2.12 |
| cp->q | 01->01 | PD | 0.211 | 0.266 | 0.545 | 0.459 | 0.624 | 1.23 | 1.24 | 1.76 | 3.14 |
| | | | 0.180 + 0.238 * CL | | | 0.416 + 0.609 * CL | | | 1.12 + 1.59 * CL | | |
| | | TR | 0.0874 | 0.228 | 0.847 | 0.198 | 0.537 | 1.89 | 0.456 | 1.33 | 4.86 |
| cp->qbar | 01->01 | PD | 0.222 | 0.251 | 0.440 | 0.506 | 0.633 | 1.08 | 1.44 | 1.72 | 2.80 |
| | | | 0.202 + 0.169 * CL | | | 0.475 + 0.408 * CL | | | 1.35 + 1.05 * CL | | |
| | | TR | 0.0959 | 0.154 | 0.565 | 0.140 | 0.447 | 1.26 | 0.364 | 0.910 | 3.24 |
| cp->q | 01->10 | PD | 0.209 | 0.316 | 0.797 | 0.429 | 0.631 | 1.37 | 1.16 | 1.58 | 2.91 |
| | | | 0.162 + 0.451 * CL | | | 0.378 + 0.714 * CL | | | 1.10 + 1.32 * CL | | |
| | | TR | 0.107 | 0.310 | 1.39 | 0.210 | 0.537 | 1.92 | 0.441 | 1.06 | 3.55 |

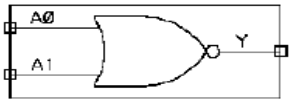
**Special Timing Information**

| | best, 5.5V, -55C | typical, 5V, 25C | worst, 4.5V, 125C |
|---|---|---|---|
| Setup-time on D | 0.3 | 0.6 | 1.4 |
| Hold-time on D | 0.1 | 0.05 | 0.01 |
| Minimum-pulse-width-low on CP | 0.2 | 0.3 | 0.9 |
| Minimum-pulse-width-high on CP | 0.08 | 0.2 | 0.6 |
| Minimum_period on CP | 0.4 | 0.8 | 2.2 |
| Maximum-fall-time on CP | 4 | 35 | 3.8e-02 |

AμE Advanced Microelectronics
A Division of ITD

**ms080cmosxCells**
CMOSX 0.8 Micron
Standard Cell Library

**2 Input NOR 1x drive** **NOR2**




**Size in microns (W x H):**
12.2 x 45.0

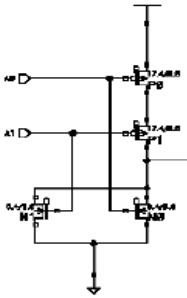
**Pin Capacitance (fF)**

| pin | best | typical | worst |
|---|---|---|---|
| A0 | 16.5 | 24.2 | 37.1 |
| A1 | 15.9 | 23.1 | 34.6 |
| Y | 5.51 | 10.1 | 13.6 |

**Logic Equation:**
Y = !(A0 | A1)

**Truth Table**

| A0 | A1 | Y |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

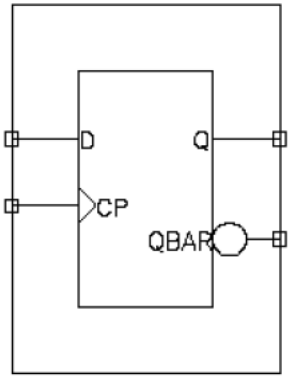



**Delay Information**

| Info | Timing | | best, 5.5V, -55C, load:0.35pF tr:0.504ns, tf:0.638ns | | | typical, 5V, 25C, load:0.35pF tr:1.15ns, tf:1.12ns | | | worst, 4.5V, 125C, load:0.35pF tr:2.78ns, tf:2.27ns | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 0.25 * load | 1 * load | 4 * load | 0.25 * load | 1 * load | 4 * load | 0.25 * load | 1 * load | 4 * load |
| a1->y | 10->01 | PD | 0.108 | 0.281 | 0.827 | 0.273 | 0.629 | 1.84 | 0.749 | 1.68 | 5.07 |
| | | | 0.0757 + 0.543 * CL | | | 0.189 + 1.21 * CL | | | 0.515 + 3.27 * CL | | |
| | | TR | 0.309 | 0.562 | 1.79 | 0.760 | 1.20 | 3.99 | 1.15 | 2.91 | 10.3 |
| a1->y | 01->10 | PD | 0.141 | 0.337 | 1.08 | 0.357 | 0.695 | 1.72 | 0.902 | 1.25 | 3.40 |
| | | | 0.0955 + 0.668 * CL | | | 0.196 + 1.10 * CL | | | 0.420 + 2.17 * CL | | |
| | | TR | 0.297 | 0.551 | 1.78 | 0.491 | 0.948 | 2.93 | 0.920 | 1.87 | 5.61 |
| a0->y | 10->01 | PD | 0.0542 | 0.225 | 0.767 | 0.212 | 0.549 | 1.79 | 0.636 | 1.59 | 4.85 |
| | | | 0.0244 + 0.555 * CL | | | 0.123 + 1.19 * CL | | | 0.371 + 3.21 * CL | | |
| | | TR | 0.268 | 0.555 | 1.77 | 0.451 | 1.17 | 3.91 | 1.10 | 2.84 | 10.1 |
| a0->y | 01->10 | PD | 0.166 | 0.340 | 1.08 | 0.300 | 0.627 | 1.72 | 0.809 | 1.31 | 3.41 |
| | | | 0.114 + 0.657 * CL | | | 0.250 + 1.01 * CL | | | 0.595 + 2.12 * CL | | |
| | | TR | 0.319 | 0.556 | 1.79 | 0.544 | 0.982 | 2.94 | 1.05 | 1.97 | 5.68 |

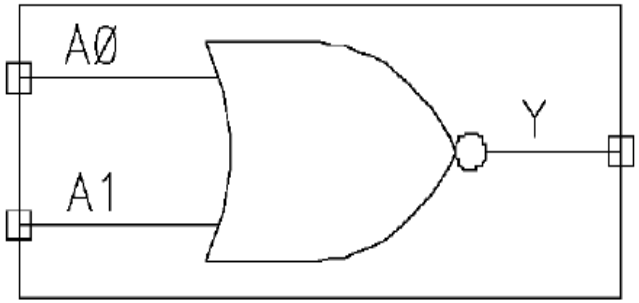
# Cload on flip-flop

- = output capacitance of flip-flop + input capacitance of NOR gate




**Pin Capacitance (fF)**

| pin | best | typical | worst |
|---|---|---|---|
| CP | 14.7 | 21.6 | 30.9 |
| D | 10.4 | 13.5 | 18.4 |
| Q | 5.04 | 10.6 | 11.7 |
| QBAR | 11.4 | 12.7 | 22.3 |




**Pin Capacitance (fF)**

| pin | best | typical | worst |
|---|---|---|---|
| A0 | 16.5 | 24.2 | 37.1 |
| A1 | 15.9 | 23.3 | 34.6 |
| Y | 5.54 | 10.1 | 13.6 |

# Timing Tables

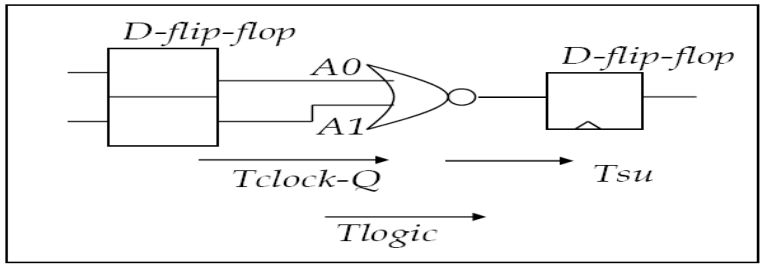
- Predict delay
  - Note delay different for rising and falling edges

**Delay Information**

| Path | Timing | | best, 5.5V, -55C, load:0.35pF tr:0.584ns, tf:0.638ns | | | typical, 5V, 25C, load:0.35pF tr:1.15ns, tf:1.12ns | | | worst, 4.5V, 125C, load:0.35pF tr:2.78ns, tf:2.27ns | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 0.25 * load | 1 * load | 4 * load | 0.25 * load | 1 * load | 4 * load | 0.25 * load | 1 * load | 4 * load |
| cp->qbar | 01->10 | PD | 0.249 | 0.315 | 0.647 | 0.553 | 0.679 | 1.18 | 1.52 | 1.76 | 2.61 |
| | | | 0.213 + 0.308 * CL | | | 0.517 + 0.478 * CL | | | 1.46 + 0.831 * CL | | |
| | | TR | 0.0697 | 0.206 | 0.811 | 0.144 | 0.354 | 1.29 | 0.281 | 0.668 | 2.32 |
| cp->q | 01->01 | PD | 0.211 | 0.266 | 0.545 | 0.459 | 0.624 | 1.25 | 1.24 | 1.78 | 3.34 |
| | | | 0.180 + 0.258 * CL | | | 0.416 + 0.609 * CL | | | 1.12 + 1.59 * CL | | |
| | | TR | 0.0874 | 0.229 | 0.847 | 0.198 | 0.517 | 1.89 | 0.456 | 1.33 | 4.86 |
| cp->qbar | 01->01 | PD | 0.222 | 0.258 | 0.440 | 0.506 | 0.613 | 1.03 | 1.44 | 1.72 | 2.80 |
| | | | 0.202 + 0.169 * CL | | | 0.475 + 0.403 * CL | | | 1.35 + 1.03 * CL | | |
| | | TR | 0.0669 | 0.154 | 0.565 | 0.140 | 0.347 | 1.26 | 0.364 | 0.901 | 3.24 |
| cp->q | 01->10 | PD | 0.209 | 0.316 | 0.797 | 0.429 | 0.631 | 1.37 | 1.16 | 1.59 | 2.91 |
| | | | 0.162 + 0.451 * CL | | | 0.378 + 0.714 * CL | | | 1.09 + 1.32 * CL | | |
| | | TR | 0.107 | 0.310 | 1.19 | 0.210 | 0.537 | 1.92 | 0.441 | 1.06 | 3.55 |

# Data Sheet Example

- Using timing approximations in the datasheet, what is the maximum clock frequency for this circuit (ignore wire load, Tskew):




*Tclock-Q:*

| | | | | | |
|---|---|---|---|---|---|
| cp->q | 01->01 | PD | 1.24 | 1.70 | 3.34 |
| | | TR | 1.12 + 1.59 * CL | | |
| | | | 0.456 | 1.33 | 4.86 |
| cp->q | 01->10 | PD | 1.16 | 1.59 | 2.91 |
| | | TR | 1.09 + 1.32 * CL | | |
| | | | 0.441 | 1.06 | 3.55 |

*CL:*

**Pin Capacitance (fF)**

| pin | best | typical | worst |
|---|---|---|---|
| A0 | 16.5 | 24.2 | 37.1 |
| A1 | 15.9 | 23.3 | 34.6 |
| Y | 5.54 | 10.1 | 13.6 |

DFF

**Pin Capacitance (fF)**

| pin | best | typical | worst |
|---|---|---|---|
| CP | 14.7 | 21.6 | 30.9 |
| D | 10.4 | 13.5 | 18.4 |
| Q | 5.04 | 10.6 | 11.7 |
| QBAR | 11.4 | 12.7 | 22.3 |

Q -> A0:
CL_max = 0.0371+0.0117 pF = 0.0488 pF
T_cp-Q_max = max (1.12+1.59*CL, 1.09 + 1.32*CL)
= max (1.12+1.59*0.0488, 1.09 + 1.32*0.0488)
= max (1.2, 1.15) = 1.2 ns
Q->A1: T_cp-Q_max = 1.2 ns

# …Example

*Tlogic:*

| | | | worst, 4.5V, 125C, load:0.35pF tr:2.78ns, tf:2.27ns | | |
|---|---|---|---|---|---|
| | | | 0.25 * load | 1 * load | 4 * load |
| a1->y | 10->01 | PD | 0.749 | 1.68 | 5.07 |
| | | TR | 0.503 + 3.27 * CL | | |
| a1->y | 01->10 | PD | 1.15 | 2.91 | 10.3 |
| | | TR | 0.502 | 1.25 | 3.40 |
| a0->y | 10->01 | PD | 0.420 + 2.17 * CL | | |
| | | TR | 0.920 | 1.87 | 5.61 |
| a0->y | 01->10 | PD | 0.636 | 1.50 | 4.85 |
| | | TR | 0.371 + 3.21 * CL | | |
| | | | 1.10 | 2.84 | 10.3 |
| | | | 0.609 | 1.31 | 3.43 |
| | | | 0.505 + 2.12 * CL | | |
| | | | 1.05 | 1.97 | 5.68 |

NOR2

*CL:*

**Pin Capacitance (fF)** NOR2

| pin | best | typical | worst |
|---|---|---|---|
| A0 | 16.5 | 24.2 | 37.1 |
| A1 | 15.9 | 23.3 | 34.6 |
| Y | 5.54 | 10.1 | 13.6 |

DFF

**Pin Capacitance (fF)**

| pin | best | typical | worst |
|---|---|---|---|
| CP | 14.7 | 21.6 | 30.9 |
| D | 10.4 | 13.5 | 18.4 |
| Q | 5.04 | 10.6 | 11.7 |
| QBAR | 11.4 | 12.7 | 22.3 |

CL= 0.0136 + 0.0184 = 0.032 pF

From A0 : Tlogic = max (0.503 + 3.27*0.032, 0.420+2.17*0.032)
= 0.61 ns

From A1 : Tlogic = max (0.505+2.12*0.032, 0.371 +3.21*0.032)
= 0.57 ns

# …Example

*Tsu*

**Special Timing Information**

| | best, 5.5V, -55C | typical, 5V, 25C | worst, 4.5V, 125C |
|---|---|---|---|
| Setup-time on D | 0.3 | 0.6 | 1.4 |
| Hold-time on D | 0.1 | 0.05 | 0.01 |
| Minimum-pulse-width-low on CP | 0.2 | 0.3 | 0.9 |
| Minimum-pulse-width-high on CP | 0.08 | 0.2 | 0.6 |
| Minimum-period on CP | 0.4 | 0.8 | 2.2 |
| Maximum-fall-time on CP | 4 | 39 | 3.8e+02 |

$$\begin{aligned} Tclock &> Tcp\text{-}Q\_max + Tlogic\_max + Tsu\_max \\ &= 1.2 + 0.61 + 1.4 \\ &= 3.21\ ns \\ Fclock &< 311\ MHz \end{aligned}$$

# Estimating and Improving Performance

- With a focus on timing:
- Topics:
  - Metrics : FO-4
  - Typical timing budgets
  - Pipelining and Parallelism
  - Logic style

# Delay Metric

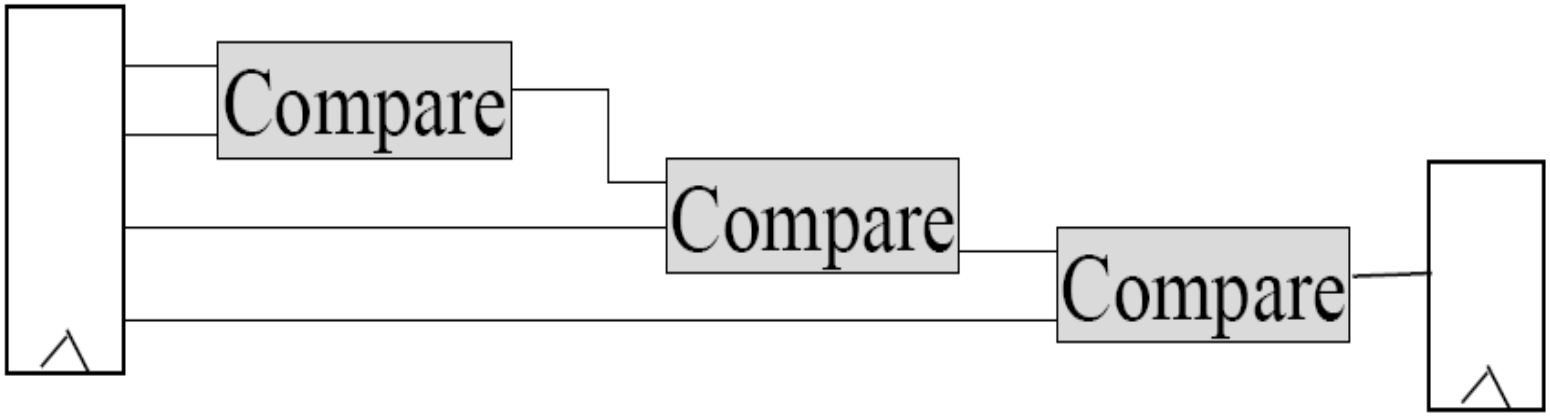
- Usual Metric for delay:
  - Fanout of 4 inverter delay: FO4
- Estimating FO4:
  - Typical ~ 360 x Leff (ps)
  - Worst Case ~ 600 x Leff (ps)
  - Leff = Effective gate length in um ~ 0.7 x Ldrawn
  - E.g. In a 0.18 um process, Leff = 0.126 um and FO-4 <= 75 ps
- Exemplar delays:

Inverter = FO-4
1-bit adder = 10 FO4
Flip-flop t_cp-Q = 4 FO
Clock skew = 4 FO4
Clock jitter = 2 FO4

2-input NAND gate = 2FO4
2-input Multiplexer = 4 FO4
4 Flip-flop t_su / t_h = 2 FO4

# Examples of Improving Timing Performance

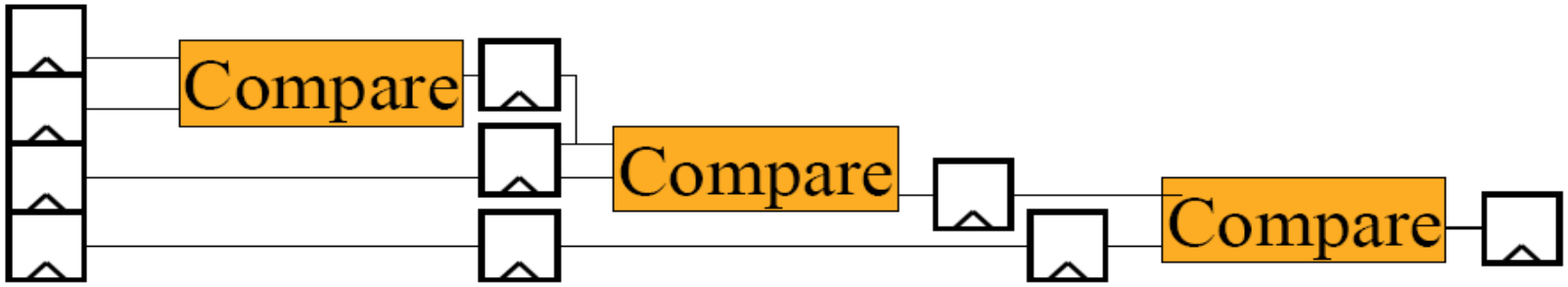
- Example 1 : Benefits of Pipelining and Parallelism
- Example:




If t_comparator = 20 FO4, what is the clock period?
(Use values on previous page)

# Pipelining

- Replace with:

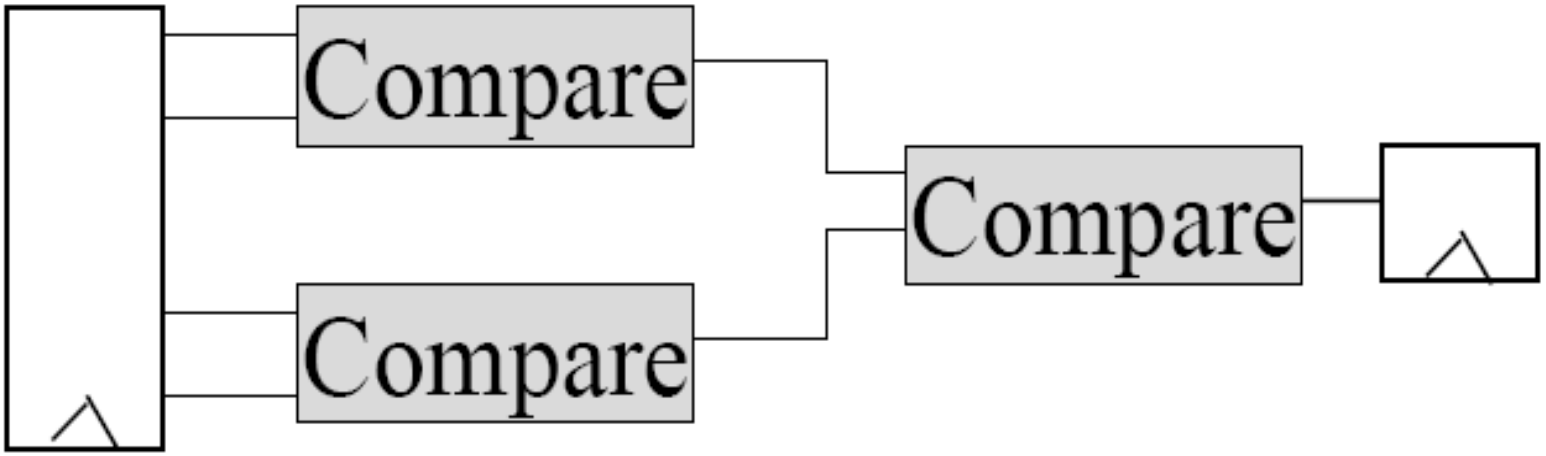



$$T\_cp = t\_ck\text{-}Q + t\_logic + t\_su + t\_skew + t\_jitter$$
$$= 4 + 20 + 2 + 4 + 2 = 32\ FO4$$

- What is the delay improvement?
- What is the drawback?

# Logic Level Parallelism

- Replace with:



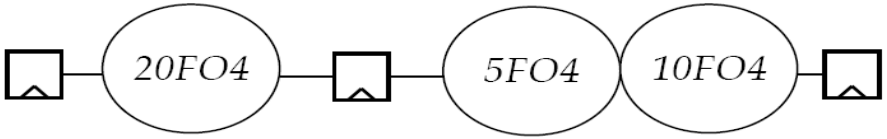

- Clock Period = 52 FO-4
- No increase in area

# Retiming

- Impact of critical paths can often be reduced by retiming or rebalancing a design:
- Example:
  - Before:


$$T\_cp = 4 + 20 + 5 + 2 + 4 + 2 = 37\ FO4$$

  - After:




$$T\_cp = 4 + 20 + 2 + 4 + 2 = 32\ FO4$$

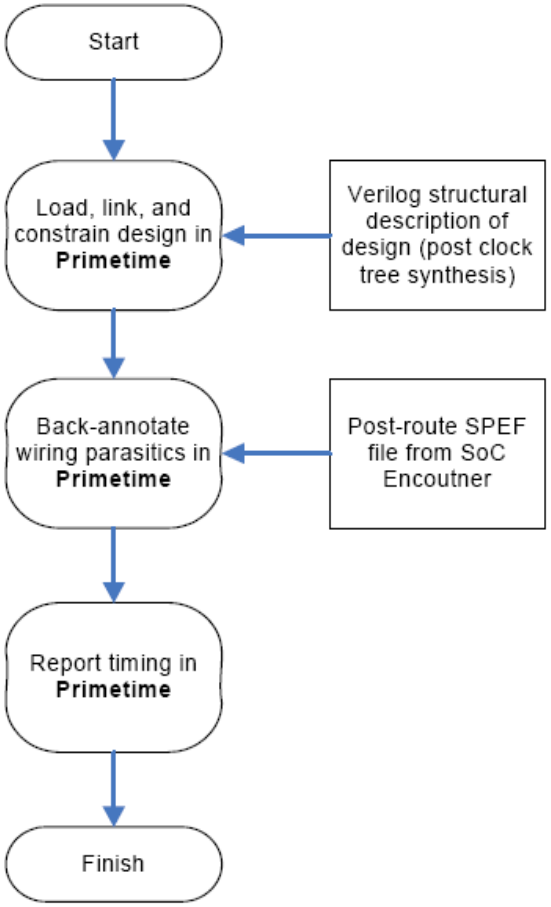
Note: Clock level logic sequence has been changed

# Timing in CAD Flow

- During synthesis
  - Tool calculates path delays under worst-case delay conditions
  - Determines critical path
  - Moves logic to a faster path if setup violation predicted
- After synthesis:
  - Perform Static Timing Analysis
  - Determine no setup violations exist under worst case conditions
  - Determine no hold violations exist under best case conditions
- After place and route:
  - Perform Timing Analysis
  - i.e. Run timing verification tools on netlist with actual delays
  - Back-annotate actual delays to netlist from later tools

# Initial Delay Estimation Flow

- Primetime: Gate-level static timing analysis tool
  - Report timing for critical path
- Back-annotate wire parasitics for more accuracy
  - SPEF file from place and route tool



# Sidebar

Not Examinable!

- What is asynchronous design?
- Can we use deliberate local clock skew in a design?
- Are you sure flip-flops are better, cycle stealing sounds useful?

Message: The CAD tools' capabilities constrain your design flexibility.

# Summary

1. What determines the maximum clock frequency?
2. What is a hold violation?
3. Why do we prefer flip-flop designs over using latches?
4. What tool is used to check timing at design closure?

# Remember

Methodology for purposes of ASIC Design class

- If at all possible one-edge of one clock
- If you need multiple clocks, they must have a common root and be related by factors of 2
  - E.g. Root clock : Tclock = 5 ns
  - This is BEST: Tclock only!!
  - This is OK:
    - Tclock, Tclock10, Tclock20
    - Tclock10 = 10 ns (Tclock*2)
    - Tclock20 = 20 ns (Tclock*4)
  - This is NOT OK
    - Tclock, Tclock15, Tclock17
    - Tclock15 = 15 ns (Tclock*3)
    - Tclock17 = 17 ns (??)