EDA322
Digital Design

Lecture 17:

Timing, Delay, Power

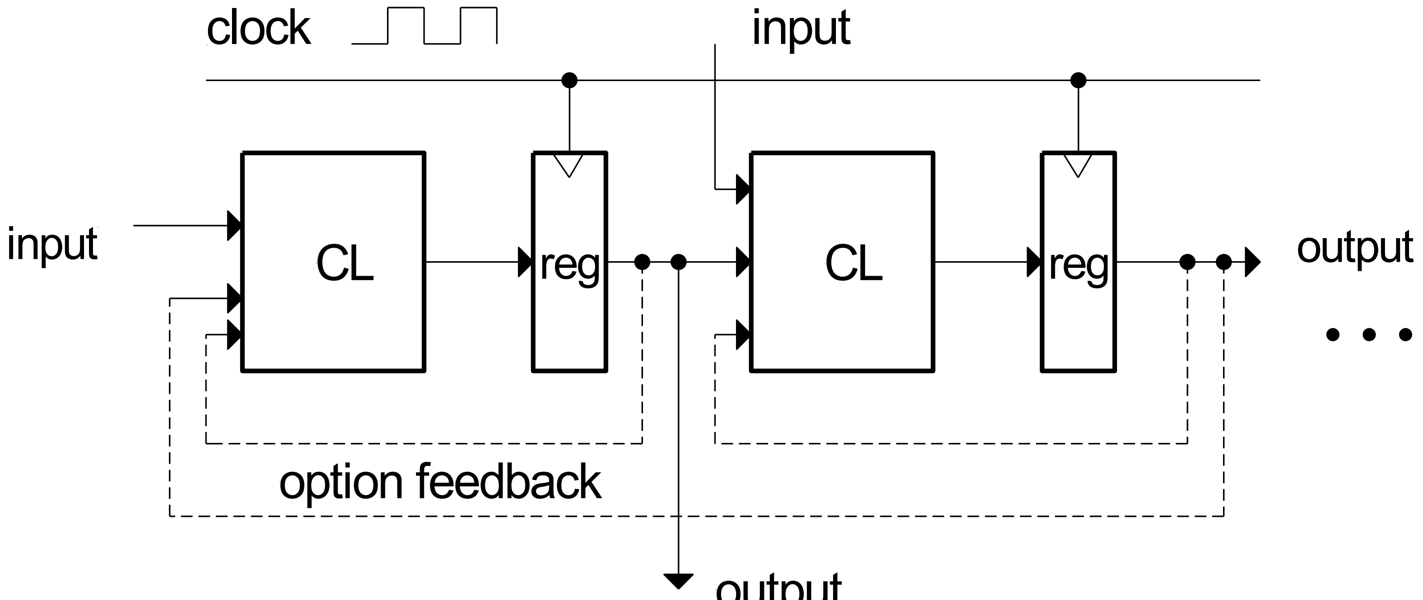
Ioannis Sourdis

Outline of Lecture 17

- Intro: Synchronous Systems
- Delay in logic gates
- Delay in wires
- Delay in flip-flops
- Timing constraints of flip-flops and memories
- clock skew
- Metastability
- Why power consumption is important
- Power metrics
- How can a logic designer control power?

Timing and Delay

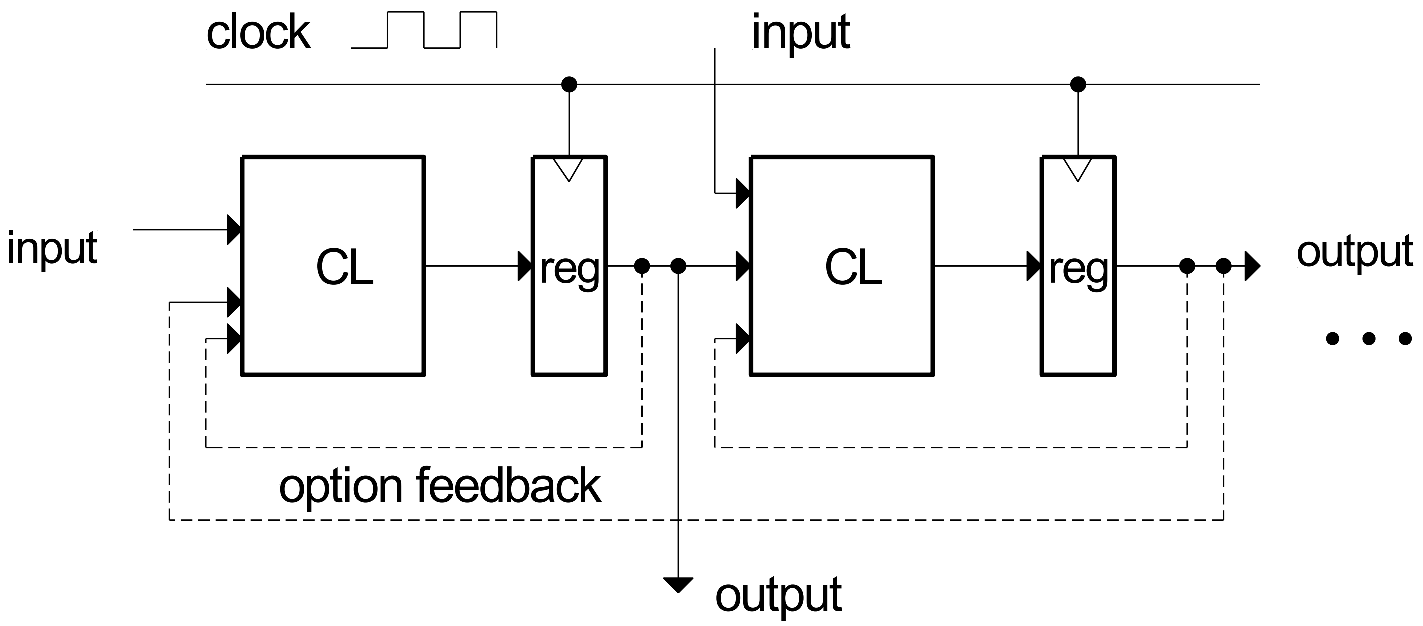
General Model of Synchronous Circuit



- All wires, except clock, may be multiple bits wide.
- Registers (reg)
  - collections of flip-flops
- clock
  - distributed to all flip-flops

- Combinational Logic Blocks (CL)
  - no internal state
  - output only a function of inputs
- Particular inputs/outputs are optional
- Optional Feedback

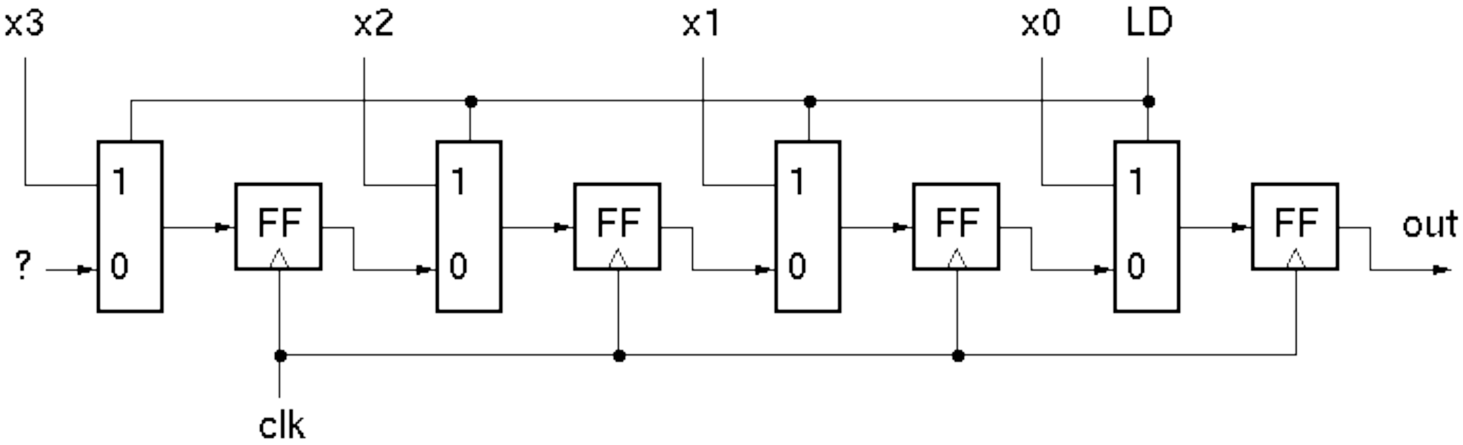
General Model of Synchronous Circuit



- How do we measure the performance/ how fast a circuit is?
  - operations/sec?
  - cycles/sec?
- What limits the clock rate?
- What happens as we increase the clock rate?

Example Circuit

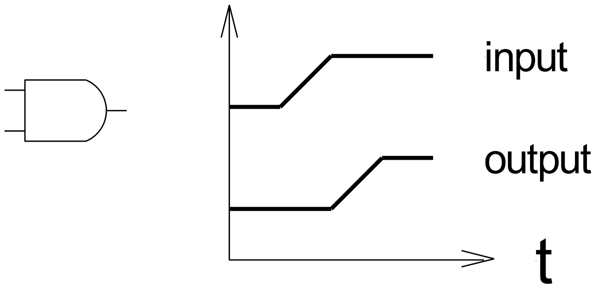
- Parallel to Serial Converter



- All signal paths single bit wide
- Registers are single flip-flops
- Combinational Logic blocks are simple multiplexors
- No feedback.

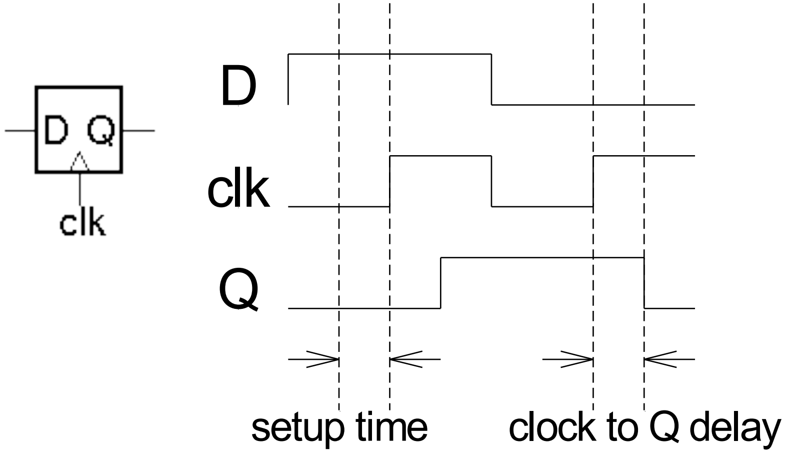
What contributes to the circuit delay? Limitations on Clock Rate

1. Logic Gate Delay



2. Wire Delay

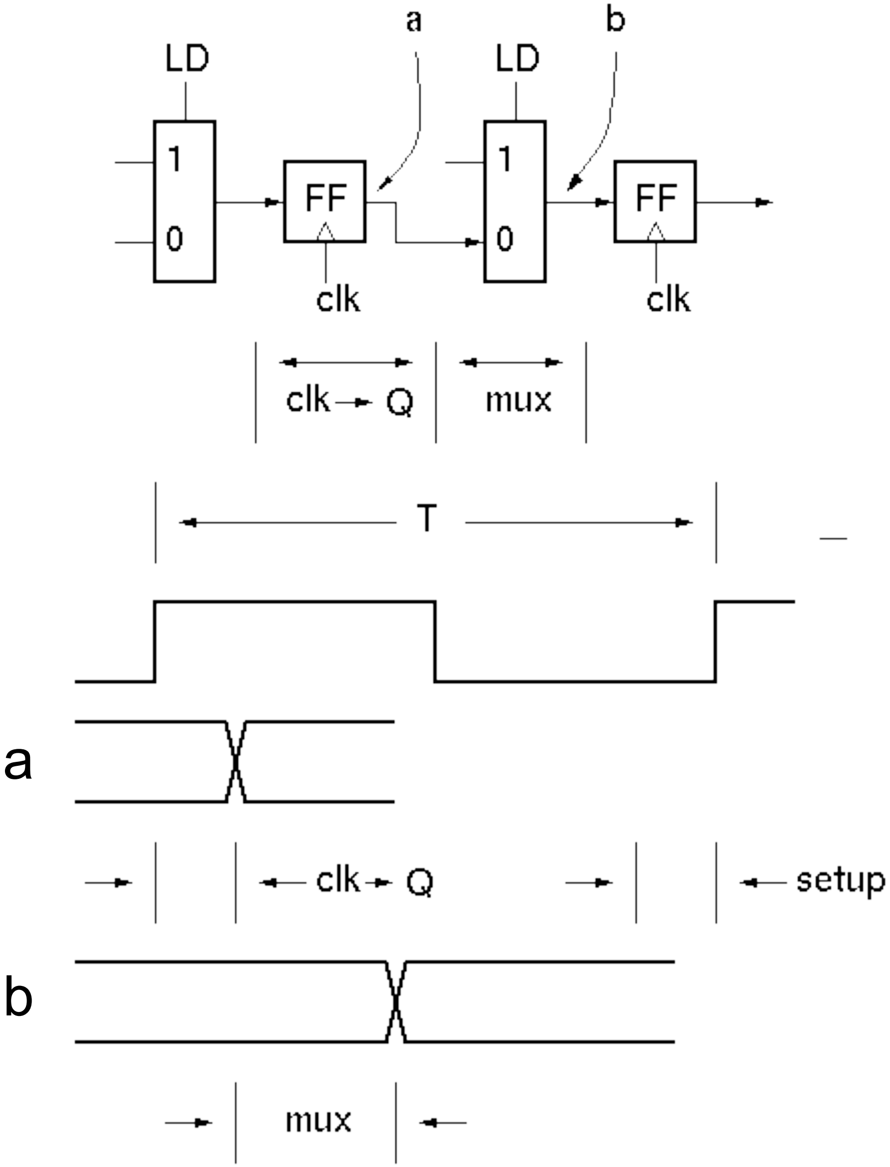
3. Delays in flip-flops



- Both times contribute to the delay.

- What must happen in one clock cycle for correct operation?
  - **Assuming perfect clock distribution (all flip-flops see the clock at the same time):**
    - All signals must be ready and “setup” before rising edge of clock.

Example



- Operating Frequency= 1/clock-period

Clock Period **T** should be:

T > time(clk->Q) + time(mux) + time(setup)

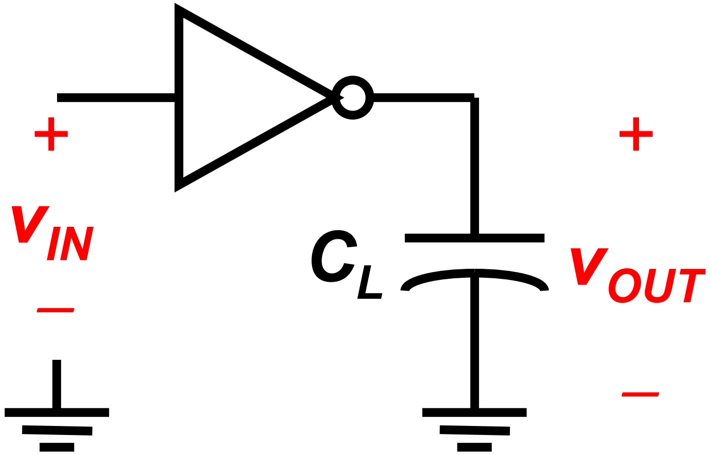
FF Propagation delay

+ wire delay

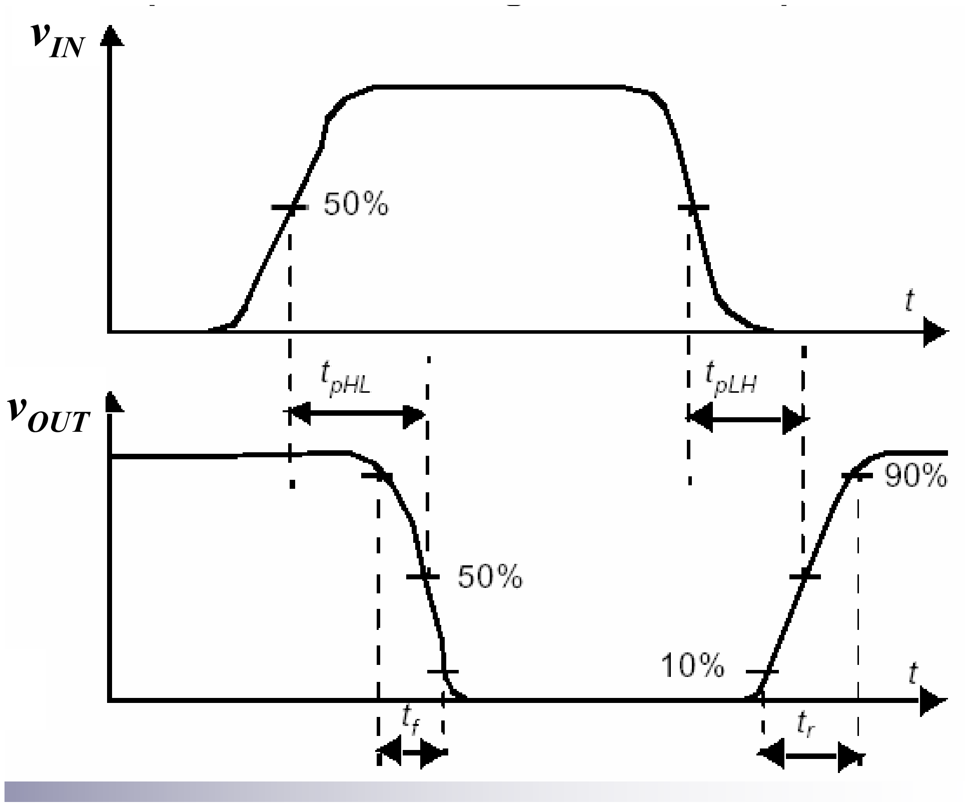
In general, Clock Period T should be:

T > time(clk->Q) + time(combinational logic) + time(wire)+ time(setup)

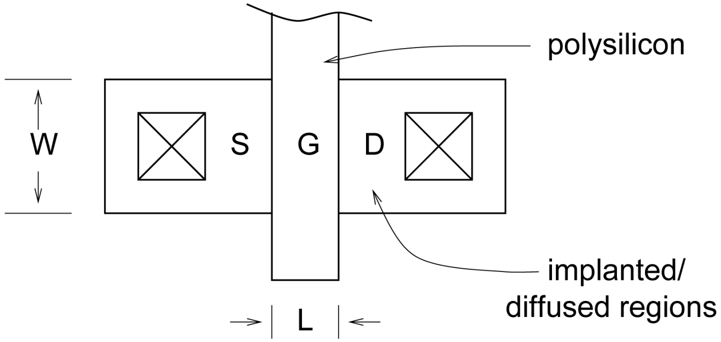
Gate Delay

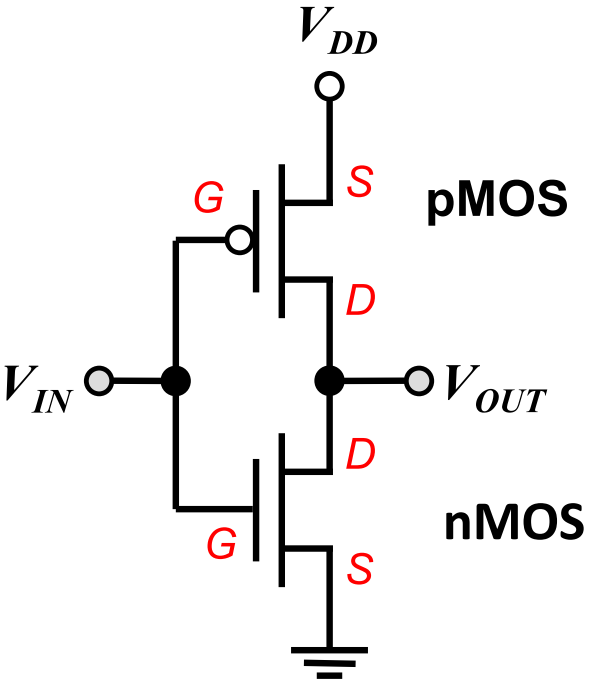


- The time needed for the output of a gate to change from the moment an input of the gate changes
- Depends on:
  - Transistor parameters
  - Fan-out: how many wires it will drive
  - Fan-in: number of inputs of a gate



Transistor Sizing for gate delay





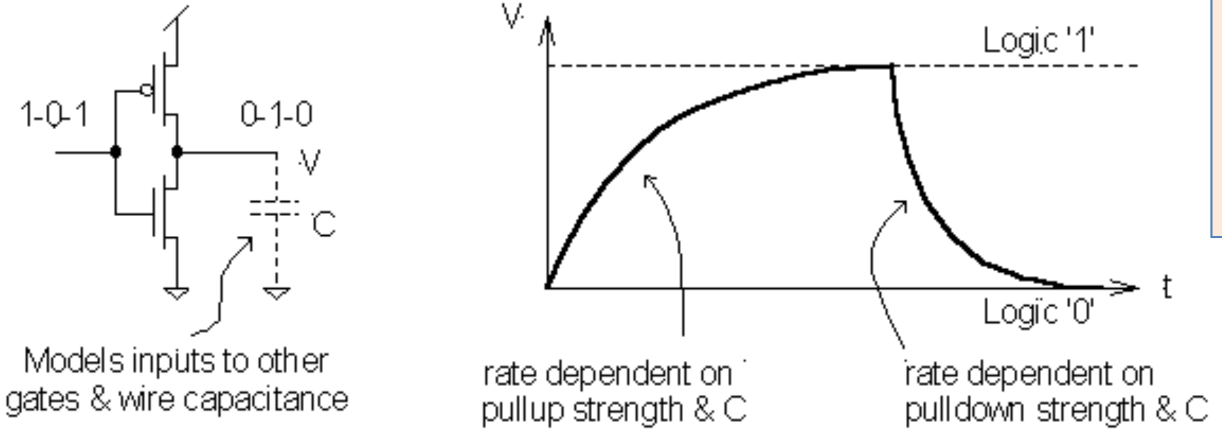
- Gate delay depends on W and L of the transistor

- Widening the transistors reduces resistance **R**, but increases capacitance **C**
  - Delay proportional to **RC**

- In order to have the on-state resistance of the PMOS transistor match that of the NMOS transistor (e.g. to achieve a symmetric voltage transfer curve), its W/L ratio must be larger by a factor of ~3. To achieve minimum propagation delay, however, the optimum factor is ~2.

Gate Switching Behavior

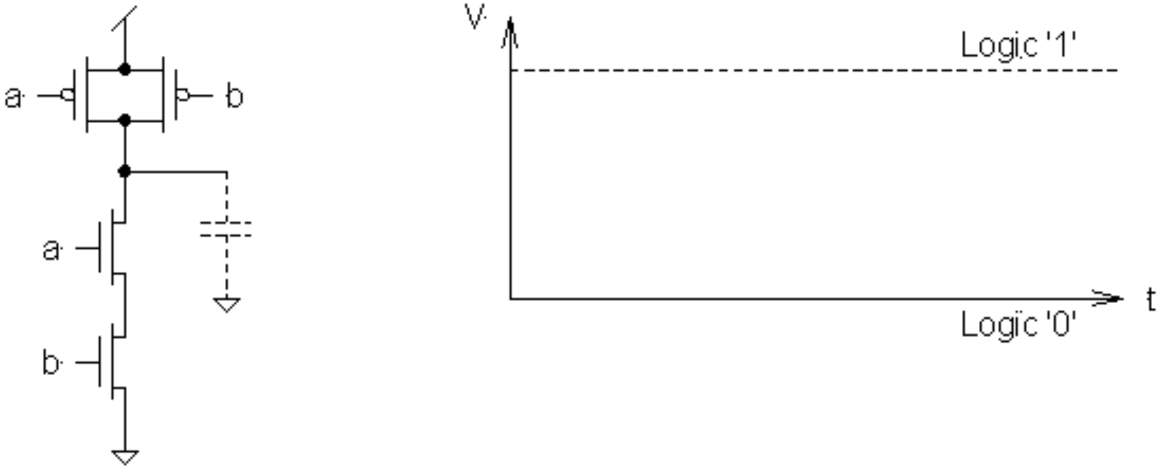
- Inverter:



1-0-1
0-1-0
V
C
Models inputs to other gates & wire capacitance
V
Logic '1'
t
Logic '0'
rate dependent on pullup strength & C
rate dependent on pulldown strength & C

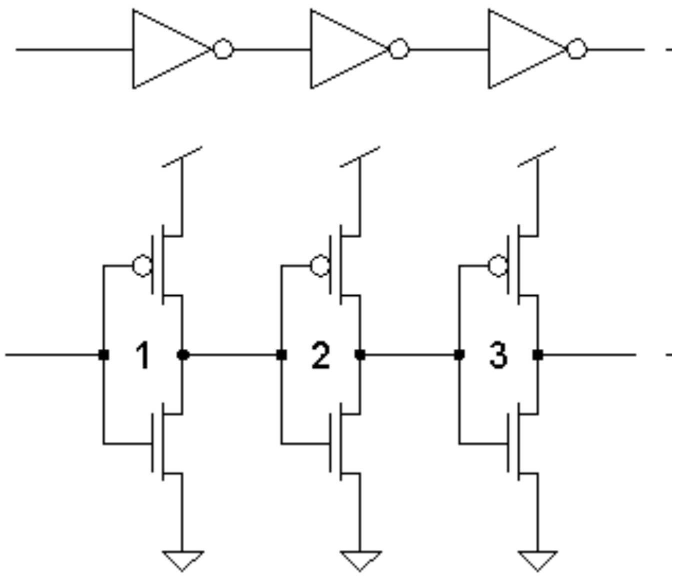
Pull-up and pull-down strength of transistor depends on W and L

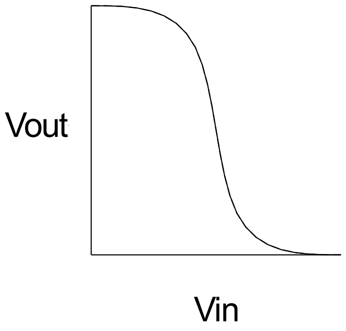
- NAND gate:

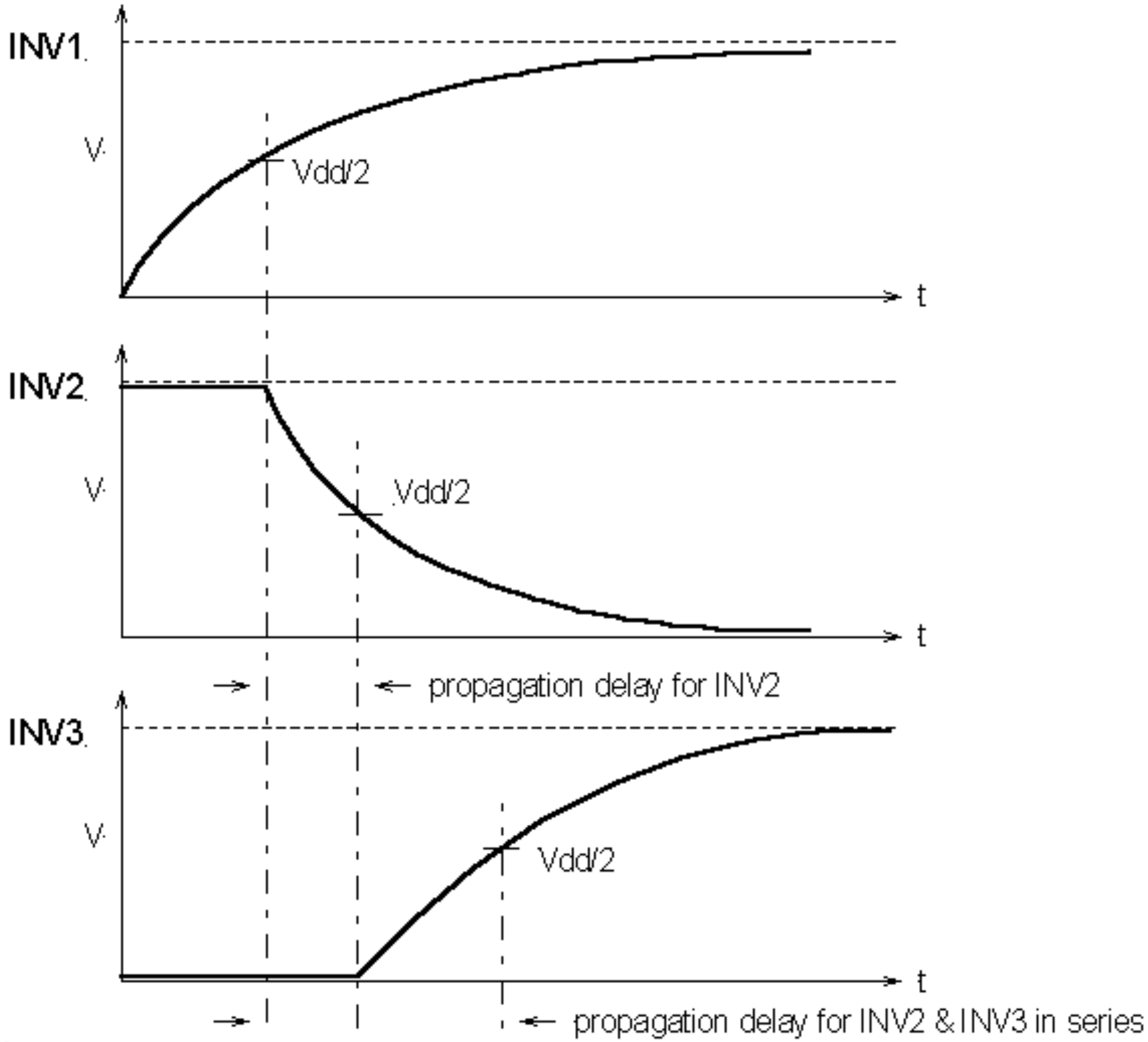


Gate Delay:

- Cascaded gates:





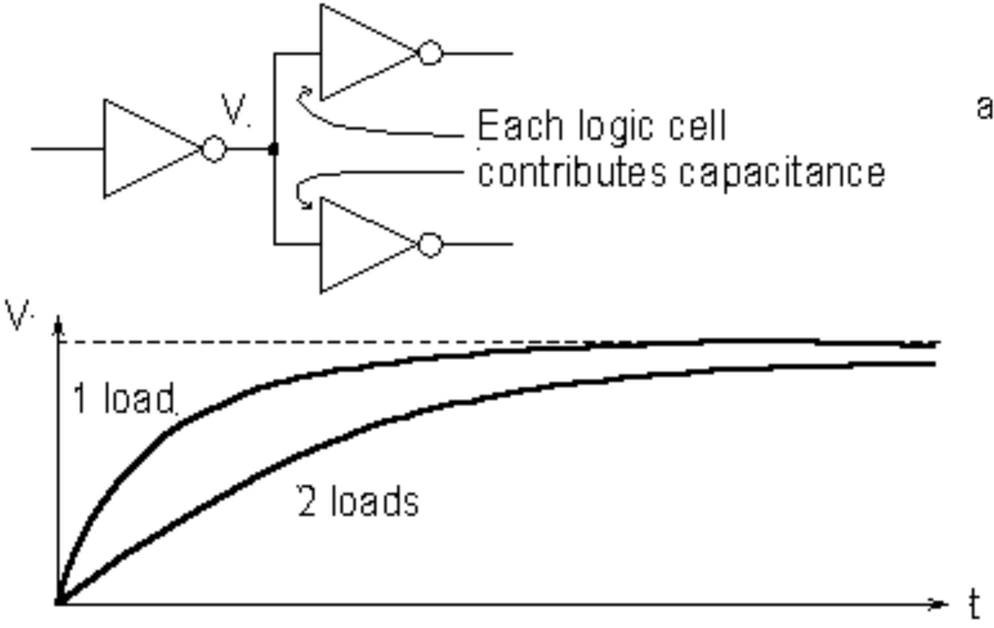


In general:

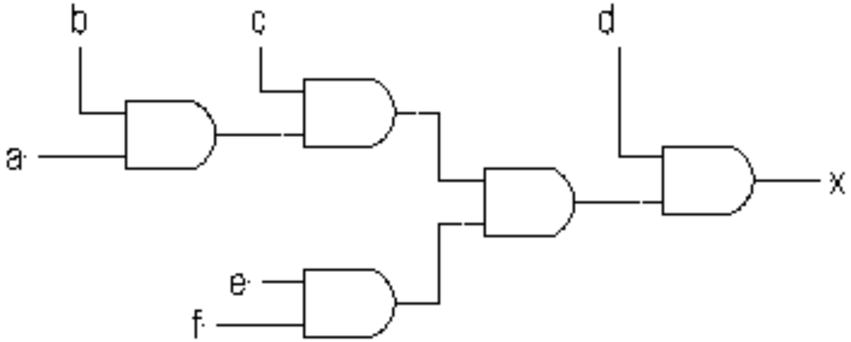
prop. delay = sum of individual prop. delays of gates in series.

Gate Delay

- **Fan-out:** is the number of outputs connected to a gate
  - the higher the fanout the slower the gate)

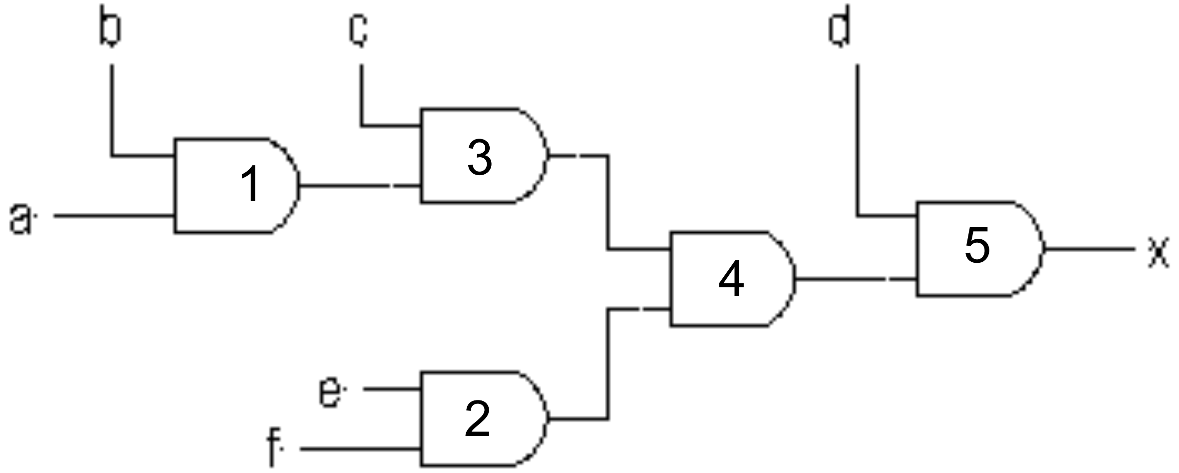


- **Fan-in:** the number of inputs connected to a gate



  - The more inputs a gate has the slower it becomes

Gate Delay

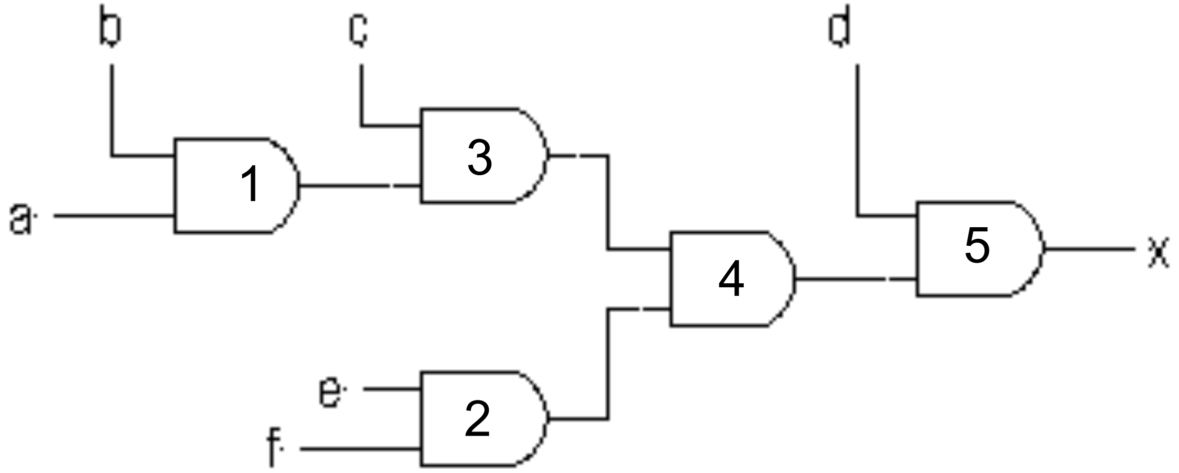


- **Example:** 2-input AND delay is 0.5ns, (a,b,c,d,e,f arrive with 0ns delay)
  – To calculate the delay of this combinational circuit:
  1. Find all paths from any input to any output in the logic,
  2. Calculate the path delays
  3. The longest (critical) path gives the delay of the circuit

- Paths: 6 inputs, 1 output => 6 paths in total
  1. a -> 1-> 3 -> 4 -> 5-> x =0+0.5+0.5+0.5+0.5=2ns
  2. b -> 1-> 3 -> 4 -> 5-> x =0+0.5+0.5+0.5+0.5=2ns
  3. c -> 3-> 4 -> 5 -> x =0+0.5+0.5+0.5=1.5ns
  4. d -> 5 -> x =0+0.5=0.5ns
  5. e -> 2-> 4 -> 5 -> x =0+0.5+0.5+0.5=1.5ns
  6. f -> 2-> 4 -> 5 -> x =0+0.5+0.5+0.5=1.5ns

Circuit delay 2 ns
Critical path a to x or b to x

Gate Delay

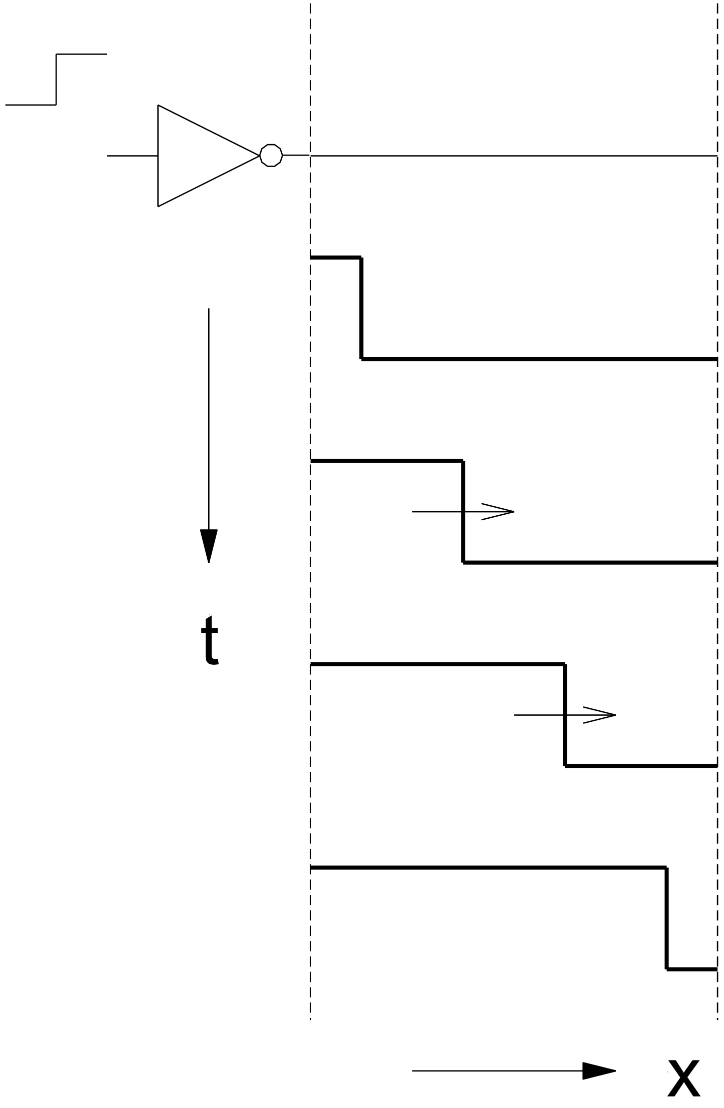


- What if **f** has a delay of 1 ns instead of 0 ns?

- Paths: 6 inputs, 1 output => 6 paths in total:
  1. a -> 1-> 3 -> 4 -> 5-> x =0+0.5+0.5+0.5+0.5=2ns
  2. b -> 1-> 3 -> 4 -> 5-> x =0+0.5+0.5+0.5+0.5=2ns
  3. c -> 3-> 4 -> 5 -> x =0+0.5+0.5+0.5=1.5ns
  4. d -> 5 -> x =0+0.5=0.5ns
  5. e -> 2-> 4 -> 5 -> x =0+0.5+0.5+0.5=1.5ns
  6. f -> 2-> 4 -> 5 -> x =1+0.5+0.5+0.5=2.5ns

Circuit delay **2.5 ns**
Critical path **f to x**

Wire Delay

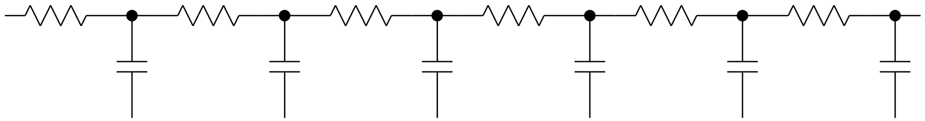


- In general wires behave as "transcribed lines":

Wire Delay

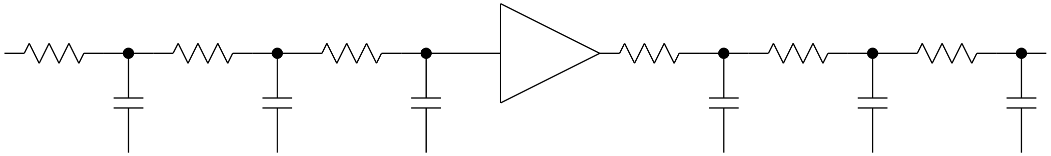
Even in cases where the transmission line effect is negligible:

- Wires posses distributed resistance **R** and capacitance **C**



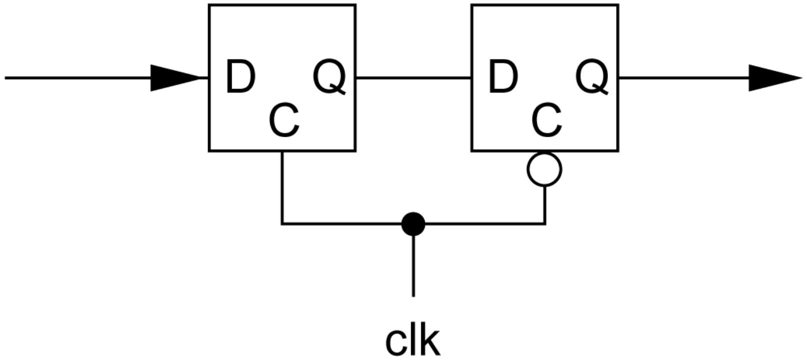
- Time constant associated with distributed RC is proportional to the ***square* of the length O(L^2)**
- For short wires on ICs, resistance is insignificant (relative to effective R of transistors), but C is important.
- Typically around half of C of gate load is in the wires.

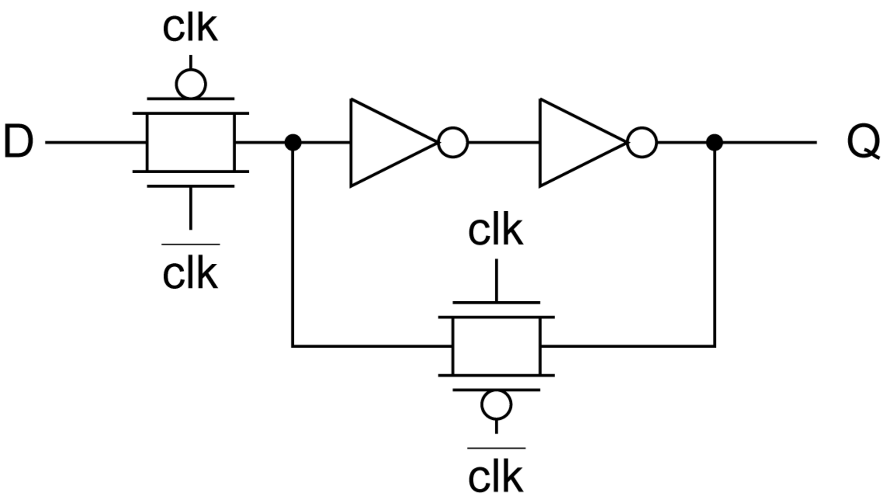
- For long wires on ICs:
  - busses, clock lines, global control signal (e.g. reset), etc.
  - distributed RC (and therefore long delay) significant
  - For long wires signals need to be "rebuffered" which contributes in the delay, too.



Wire delay is proportional to the ***square* of its length O(L^2)**

Delay in Flip-flops



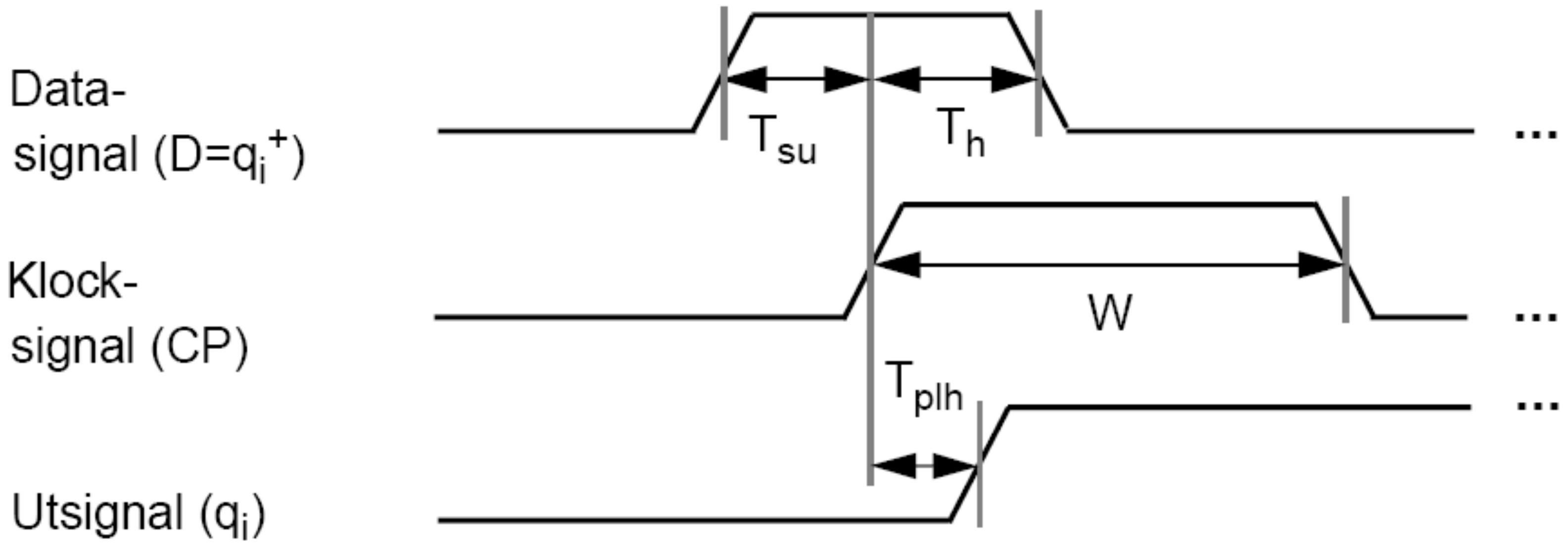


T_{su}= setup time
(the time the input of a flip flop needs to be stable before the clock edge to be stored correctly)

T_h= hold time
(the time the input of a flip flop needs to be held after the clock edge to be stored correctly)

T_p=propagation time (clock to Q)
(the time a flip-flop needs to propagate a value stored to the output)

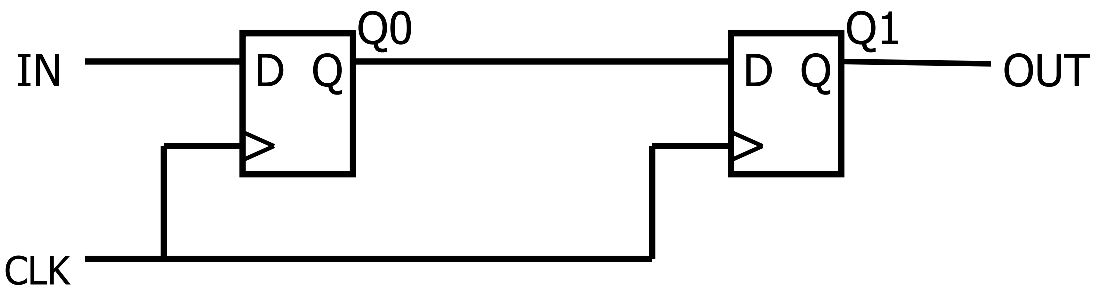
Time parameters for clocked memory elements



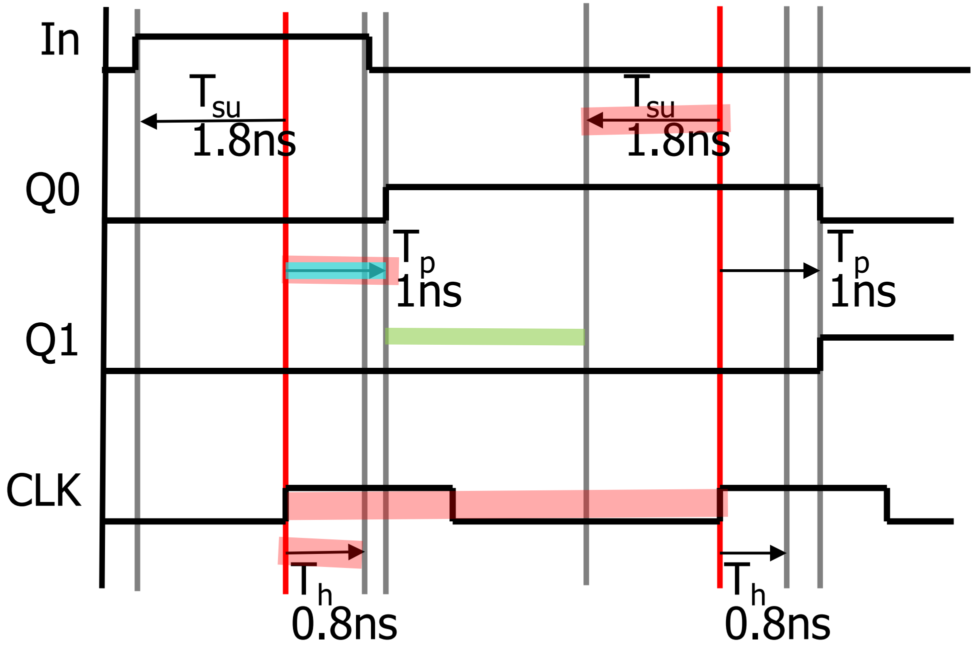
- **Shift register**

  new values go into the first stage

  while the previous value to the next step



- Why this works
  - Propagation time of Q0 greater than 'hold' time ●
  - (Clock Period – Propagation time) greater than the 'setup' time ●
  - This ensures that the next step samples the value before it is changed to a new

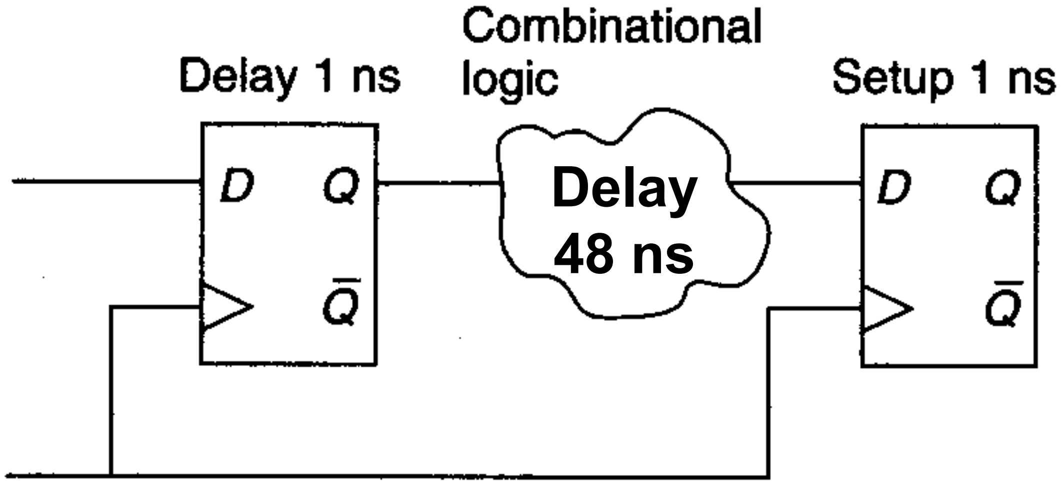


1) $T_p > T_h$
e.g. 1ns > 0.8

2) Clock Period – $T_p > T_{su}$
Or
Clock Period > $T_{su} + T_p$
e.g. Clock Period > 1.8 + 1 = 2.8ns

		Speed Grade				
		-5		-4		
Symbol	Description	Min	Max	Min	Max	Units
Clock-to-Output Times						
T_{CKO}	When reading from the FFX (FFY) Flip-Flop, the time from the active transition at the CLK input to data appearing at the XQ (YQ) output	-	0.52	-	0.60	ns
Setup Times						
T_{AS}	Time from the setup of data at the F or G input to the active transition at the CLK input of the CLB	0.37	-	0.42	-	ns
T_{DICK}	Time from the setup of data at the BX or BY input to the active transition at the CLK input of the CLB	0.32	-	0.36	-	ns
Hold Times						
T_{AH}	Time from the active transition at the CLK input to the point where data is last held at the F or G input	0	-	0	-	ns
T_{CKDI}	Time from the active transition at the CLK input to the point where data is last held at the BX or BY input	0	-	0	-	ns
Clock Timing						
T_{CH}	The High pulse width of the CLB's CLK signal	0.70	-	0.80	-	ns
T_{CL}	The Low pulse width of the CLK signal	0.70	-	0.80	-	ns
F_{TOG}	Toggle frequency (for export control)	0	657	0	572	MHz

Timing Constraints

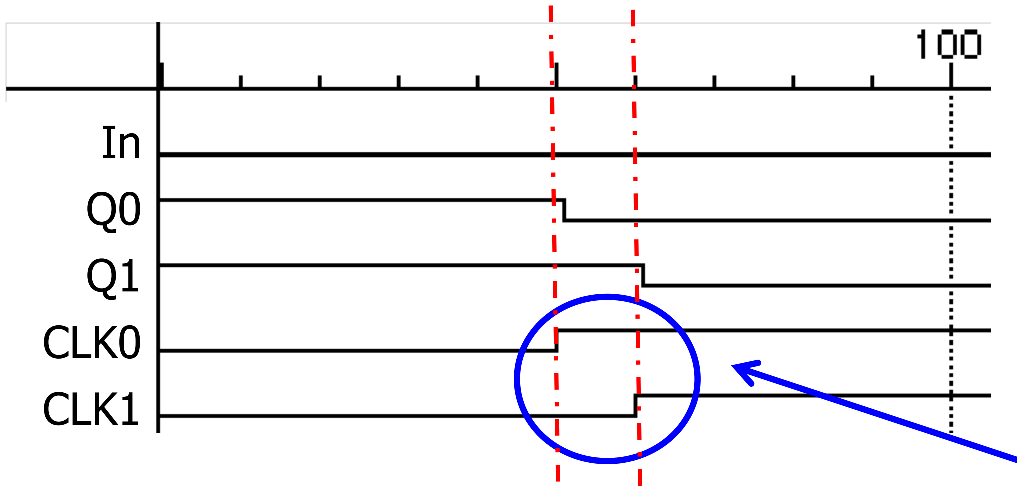


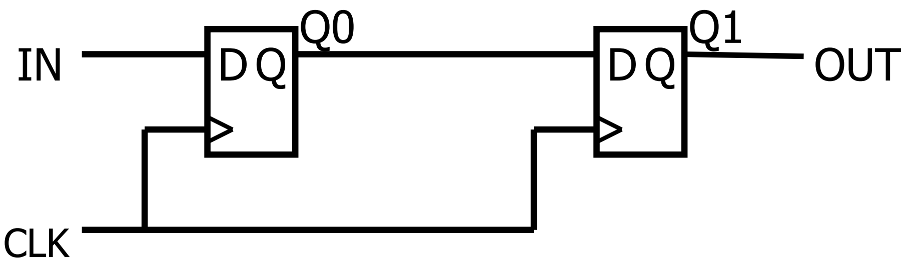
- Clock Period > 1+48+1 = 50ns
- Clock Frequency < 20MHz

- In general the minimum clock period of a design is determined by the critical path, which is the longest path not interrupted by a register (FF):
  - from IN to OUT, or
  - from IN to FF, or
  - from FF to FF, or
  - From FF to OUT

Clock Skew

- The Problem
  - correct behavior requires that the next state of all memory elements are determined by all the memory elements thus at the same time
  - This is difficult in high performance systems since the time it takes for the clock to arrive, are of the same magnitude as the delay through the logic
  - The skew effect:



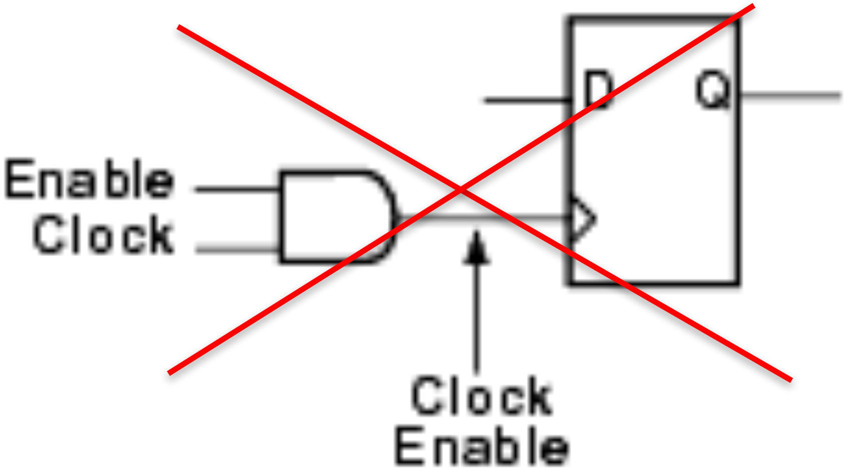


CLK1 is a delayed version of CKL0

Initial state: IN = 0, Q0 = 1, Q1 = 1
Due to skew next state is : Q0 = 0, Q1 = 0,
instead of Q0 = 0, Q1 = 1

Treating the clock right

- No clock-gating unless using dedicated synthesis libraries!
  - Otherwise clock skew is introduced
  - Use instead Enable



Quiz 16-1

http://m.socrative.com/student/#joinRoom

room number: 713113

- Q1: What are the 3 parameters that affect the delay of a gate?
- Q2: The delay of a wire with respect to its length L is:
  a. O(L)
  b. O(L^2)
  c. O(L^3)
  d. O(2^L)
- Q3: The propagation time of a Flip-flop can be shorter than its hold time (True/False)
- Q4: all synchronous designs have clock skew (True/False)

Metastability

Metastability and Asynchronous Inputs

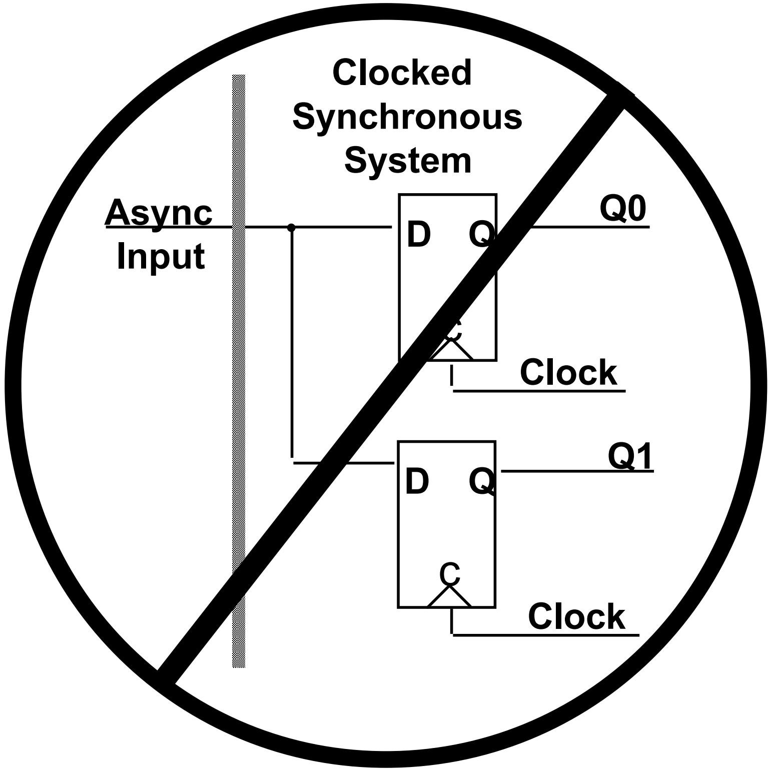
- Terms and Definitions
  - Clocked synchronous circuits
    - common reference signal called the clock
    - state of the circuit changes in relation to this clock signal
  - Asynchronous circuits
    - inputs, state, and outputs sampled or changed independent of a common reference signal
  - Synchronous inputs
    - active only when the clock edge or level is active
  - Asynchronous inputs
    - take effect immediately, without consideration of the clock

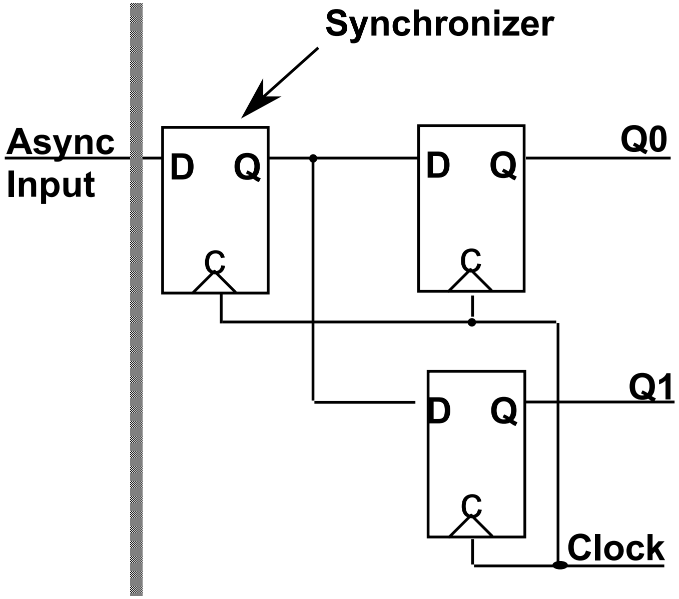
Metastability and Asynchronous Inputs

- Asynchronous Inputs Are Dangerous!
  - Since they take effect immediately, glitches can be disastrous
  - Synchronous inputs are greatly preferred!
  - But sometimes, asynchronous inputs cannot be avoided
    - e.g., reset signal, memory wait signal

Metastability and Asynchronous Inputs

Handling Asynchronous Inputs

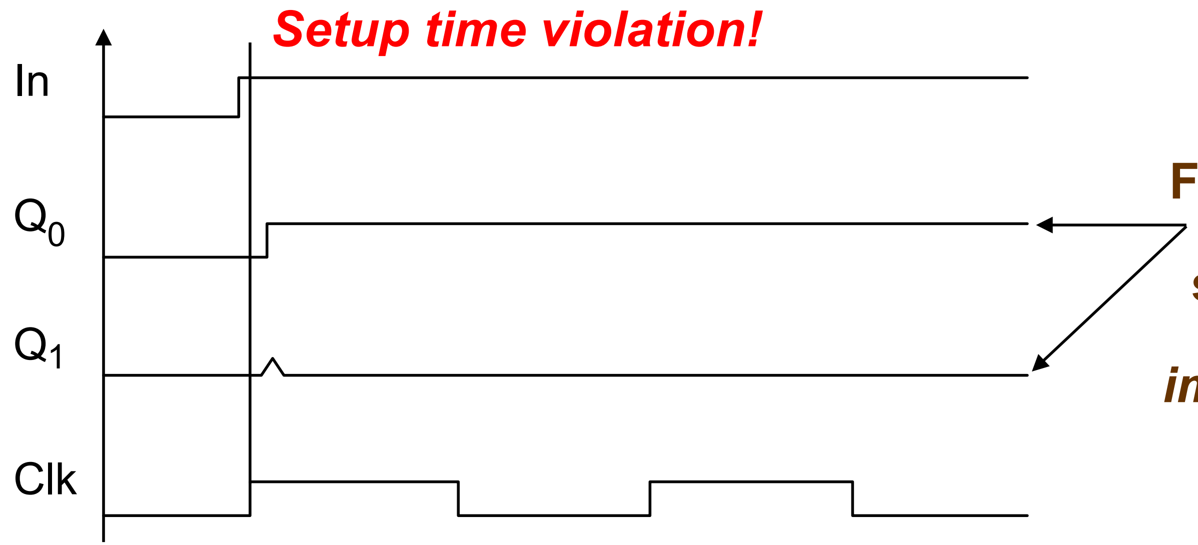




Never allow asynchronous inputs to be fanned out to more than one FF within the synchronous system

Metastability and Asynchronous Inputs

What Can Go Wrong



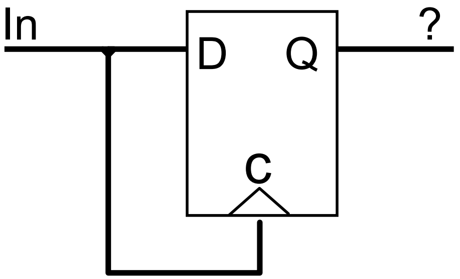
In is asynchronous
Fans out to D0 and D1
One FF catches the signal, one does not

impossible state might be reached!

Single FF that receives the asynchronous signal is *a synchronizer*

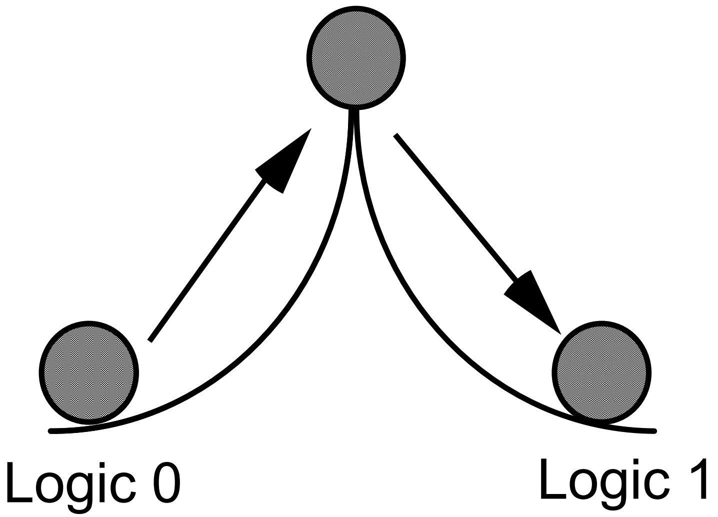
Metastability and Asynchronous Inputs

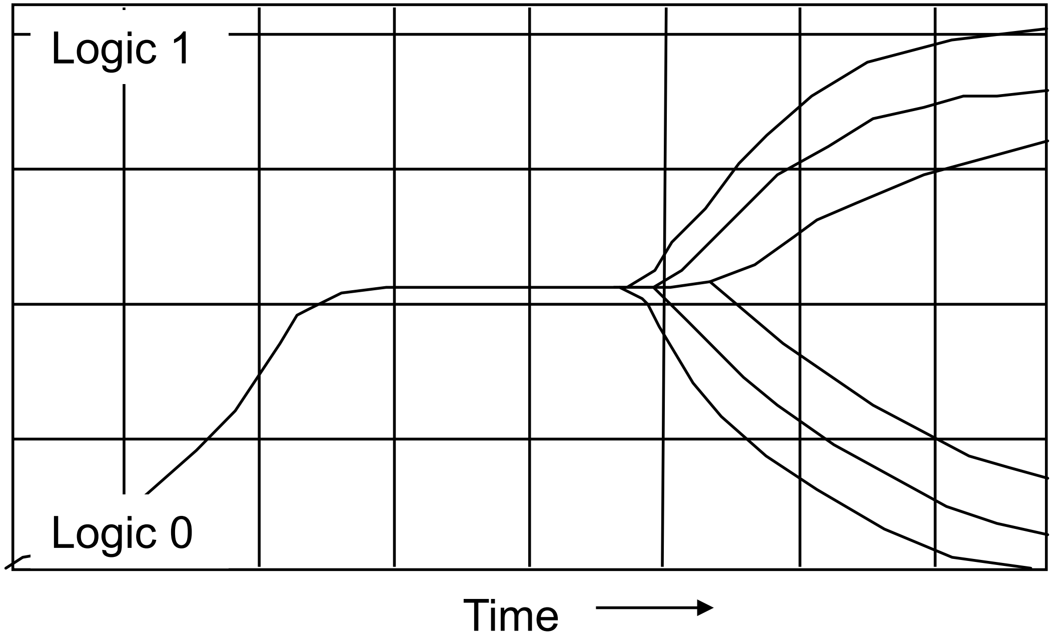
Synchronizer Failure



When FF input changes close to clock edge, the FF may enter the *metastable* state: neither a logic 0 nor a logic 1

It may stay in this state an indefinate amount of time, although this is not likely in real circuits





Small, but non-zero probability that the FF output will get stuck in an in-between state

Oscilloscope Traces Demonstrating Synchronizer Failure and Eventual Decay to Steady State

Metastability and Asynchronous Inputs

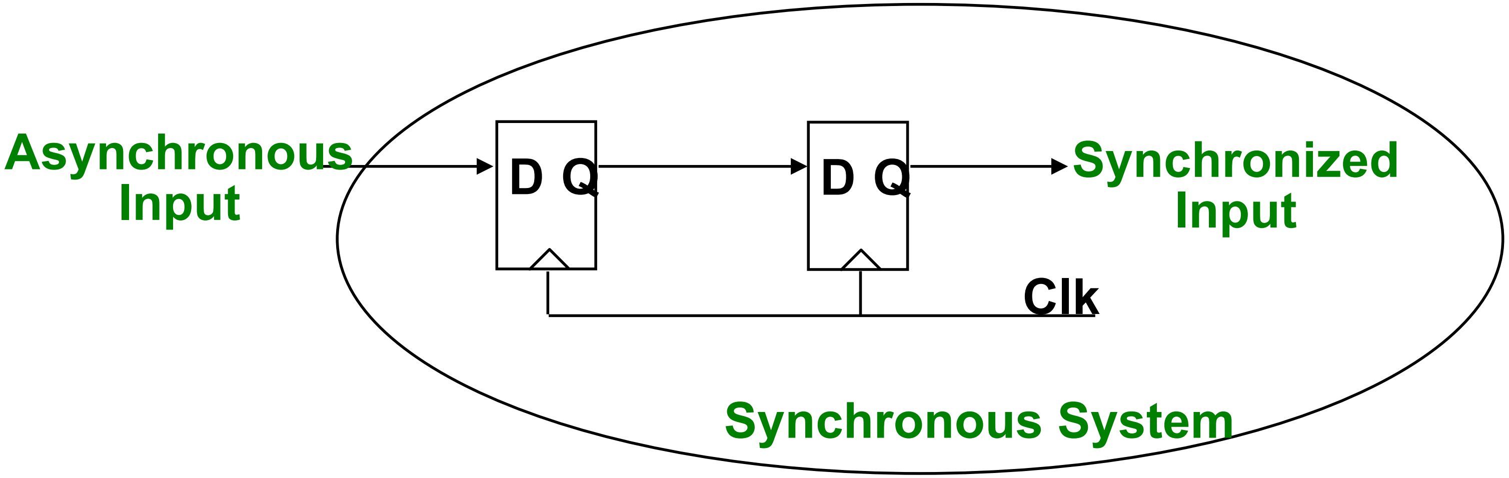
Solutions to Synchronizer Failure

- the probability of failure can never be reduced to 0, but it can be reduced

- slow down the system clock
  this gives the synchronizer more time to decay into a steady state
  synchronizer failure becomes a big problem for very high speed
  systems

- use fastest possible logic in the synchronizer
  this makes for a very sharp "peak" upon which to balance
  S or AS TTL D-FFs are recommended

- cascade two synchronizers



Metastability and MTBF

- A synchronizer design is characterised by it's Mean Time Between Failure (MTBF)
  - A failure is declared when the first synchronizer FF goes metastable and the output is not resolved before the 2nd FF is clocked
  - Depends on:
    - **FF setup/hold and prop. times,**
    - **clock frequency and**
    - **average rate of input change**
  - Different flip-flops can have greatly different MTBFs
  - Even a "small" change of the clock can be significant

Metastability MTBF

- The MTBF equation is:

$$\mathrm{MTBF}(t_r) = \frac{e^{\left(t_r / \tau \right)}}{T_0 \cdot f \cdot a}$$

Where:

t_r = resolution time (clock period - FF setup time)

T_0, τ = flip-flop characteristic constants

f = clock frequency

a = average input rate of change

Metastability MTBF

- For example using a 74LS74 Flip-Flop ($T_0 = 0.4$, $\tau = 1.5$) at a clock rate of 10 MHz and in input av. rate of change = 100 KHz
  - t_r = 80 ns (100 ns clock period - 20 ns t_{su})
  - MTBF = $3.6 \cdot 10^{11}$ sec.
- If we just change the clock to 16 MHz, things get really strange
  - t_r = 42.5 ns (62.5 ns clock period - 20 ns t_{su})
  - MTBF = 3.1 sec.!

Metastability MTBF

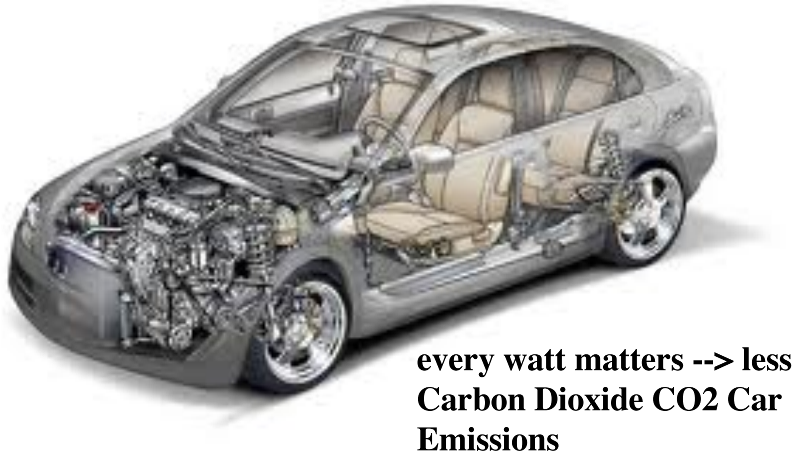
- We can improve the performance if we change to a 74**A**LS74 Flip-flop ($T_0 = 8.7 \cdot 10^{-6}$, $\tau = 1.0$)
  - $t_r = 52.5$ ns (62.5 ns clock period - 10 ns t_{su})
  - MTBF = $4.54 \cdot 10^{15}$ sec.

Power

Is Power Consumption Important?

- Embedded systems
  - Autonomy
  - peak-power
- Portable devices:
  - handhelds, laptops, phones, MP3 players, cameras, … all need to run for extended periods on small batteries without recharging
  - Devices that need regular recharging or large heavy batteries will lose out to those that don’t.
- Other embedded systems e.g. vehicles:
  - Power consumption determines CO2 emissions

Energy efficiency will keep your phone/tablet/laptop/ mp3 player ON for longer



every watt matters --> less Carbon Dioxide CO2 Car Emissions

Battery Technology for portable devices

- Battery technology has moved very slowly
- Li-Ion and NiMh still the dominate technologies
- Batteries still contribute significant to the weight of mobile devices

Nokia 61xx - 33%

Handspring PDA - 10%

Toshiba Portege 3110 laptop - 20%

Is Power Consumption Important?

- High-performance, supercomputers, datacenters
  - Performance/Watt
  - Less power means
    - less cost,
    - less energy spend for cooling
    - environmental concerns

Barcelona supercomputer

Copyright 2005. Barcelona Supercomputing Center - BSC

Google Datacenter

Is Power Consumption Important?

- General purpose computer
  - Power dissipation limits performance
  - Cannot increase frequency beyond few GHz anymore
  - Energy cost
  - Environmental concerns

Definitions

- Power supply provides energy for charging and discharging wires and transistor gates. **The energy supplied is stored and dissipated as heat.**

$\boxed{P \equiv dE / dt}$ *Rate of energy being used w.r.t time.*

Units: $P = \Delta E / \Delta t$ $\quad Watts = Joules/seconds$

- **Voltage** (potential of the charge) is increased the amount of energy dE needed to move an amount of charge dq : $V = dE / dq$
- By definition of **Current**: $I = dq / dt$
- Then: $$dE / dt = \frac{dE}{dq} \times \frac{dq}{dt} = \boxed{P = V \times I}$$

$$E = \int_{-\infty}^{t} P\, dt \quad \boxed{\text{total energy}}$$

Definitions

- Warning! In everyday language, the term "power" is used incorrectly in place of "energy."
- Power is not energy.
- Power is not something you can run out of.
- Power can not be lost or used up.
- It is not a thing, it is merely a rate.
- It can not be put into a battery any more than velocity can be put in the gas tank of a car.

Metrics

How do we measure power consumption?

- Average power: $P_{avg} = \Delta E / \Delta t$
- Peak power: $P_{peak} = \max(\lim_{\Delta t \to 0} \frac{\Delta E}{\Delta t}) = \max(\frac{dE}{dt})$

- One popular metric for computing systems is **performance per watt** (measures energy efficiency)
  - Performance can be e.g. MIPS, millions of instructions/second.
  - it measures the rate of computation(s) that can be delivered by a computer for every watt of power consumed.
  - Watt, standard unit of power consumption.
  - performance/watt is reflective of the tradeoff between performance and power. Increasing performance requires increasing power.

Metrics

- How does MIPS/watt relate to *energy*?
- Average power consumption = energy / time

MIPS/watt = (instructions/sec) / (joules/sec) = instructions/joule

i.e. operations per energy unit

  - therefore an similar metric (the inverse) is energy per operation (E/op)

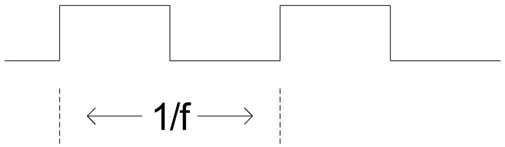
- E/op is more general - applies to more than processors
  - also, usually more relevant, as batteries life is limited by total energy draw.
  - This metric gives us a measure to use to compare two alternative implementations of a particular function.

Switching (Dynamic) Power

- Gate power consumption:
  - Assume a gate is switching its output at a rate of:

$$\alpha \cdot f$$

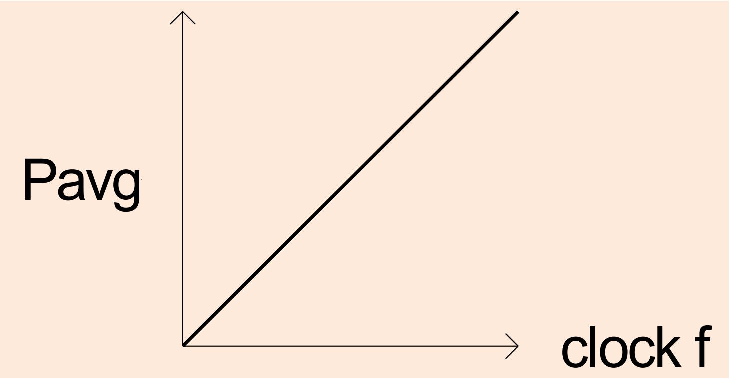
activity factor *clock rate*



$$P_{avg} = E/\Delta t = switching\ rate \cdot E_{sw}$$

$$E_{sw} = Q*V_{dd} = C*V_{dd}^{2}$$

Therefore: $$\boxed{P_{avg} = \alpha \cdot f \cdot cV_{dd}^{2}}$$



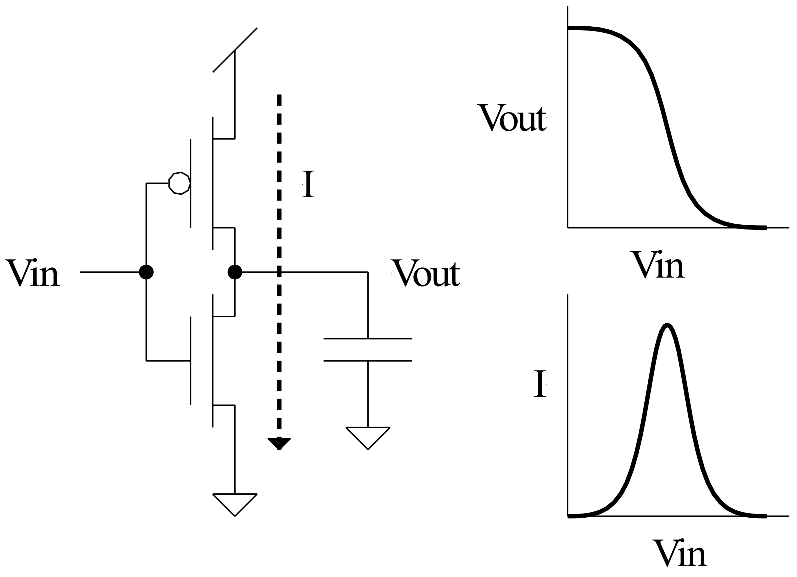
- Chip power consumption:

$$P_{avg} = n \cdot \alpha_{avg} \cdot f \cdot c_{avg} V_{dd}^{2}$$

number of nodes (or gates)

Other Sources of Energy Consumption

- **"Short Circuit" Current:**



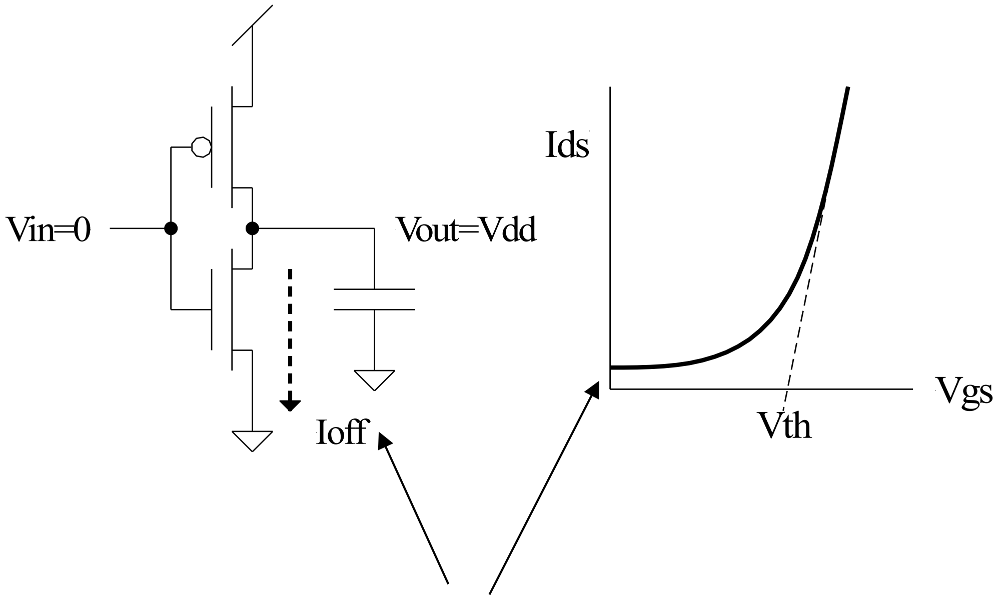
10-20% of total chip power

- There is a finite rise/fall time for both pMOS and nMOS, during transition, e.g., from OFF to ON, when both transistors will be ON for a small period of time

- During this time interval current will find a path directly from V_{DD} to ground, hence creating a short circuit current

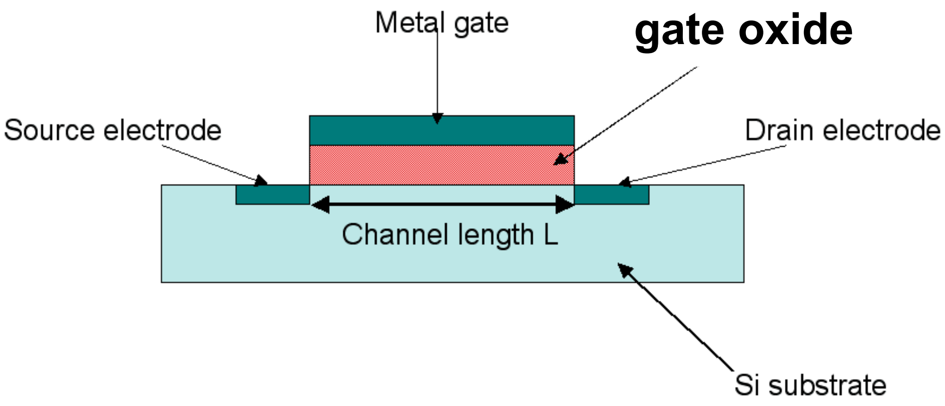
Other Sources of Energy Consumption

- **Device Ids Leakage:**



Transistor s/d conductance never turns off all the way.

Tunneling current through gate



SiO_2 is a very good insulator, but at very small thickness levels electrons can tunnel across the very thin insulation

Controlling Power Consumption

- Largest contributing component to CMOS power consumption is switching (dynamic) power:

Lower frequency less power, but longer execution time so, maybe more energy (?)

$$\boxed{P_{avg} = n \cdot \alpha_{avg} \cdot f \cdot c_{avg} V_{dd}^{\;2}}$$

- What control do you have over each factor?
- How does each effect the total Energy? (think about f)

Less activity of the submodules of A design results in lower power

Smaller area
Fewer transistors+wires
Lower power

Quiz 16-2

http://m.socrative.com/student/#joinRoom

room number: 713113

- Q1: Metastability is a violation of Flip-flop's
  a. Setup
  b. Hold time
  c. Propagation time
  d. All the above
- Q2: Dynamic power of a design depends on:
  - Number of transistors,
  - clock frequency,
  - Activity of circuits (how often they switch 0->1 or 1->0)
  - All the above

Summary

- Delay in logic gates
- Delay in wires
- Delay in flip-flops
- Timing constraints of flip-flops and memories
- clock skew
- Metastability
- Why power consumption is important
- Power metrics
- How a designer controls power?

- Book (complimentary to the slides):
  - Chapters 5, 28, 29-29.2

- Next Lecture:
  - Asynchronous circuits