Slide Set 5

Resistance, Capacitance, and IRSIM

Steve Wilton
Dept. of ECE
University of British Columbia
stevew@ece.ubc.ca

Overview

There are a number of factors that a good VLSI designer can optimize a design for: area, speed, power, etc. There are many models you can use to estimate these quantities, and many circuit design styles you can employ to optimize for them. However, when we are thinking about millions of transistors, we can't solve millions of differential equations to get the "optimal solution". Instead, it is really important to be able to be able to come up with "back-of-the-envelope" calculations to estimate how certain design decisions will affect speed/power. In this section, we will talk about some simple models you can use to help you do this.

Parasitic Capacitance and Resistance

Each transistor and piece of metal is associated with a parasitic resistance and capacitance.

- These aren't part of the design, but they are what determines speed

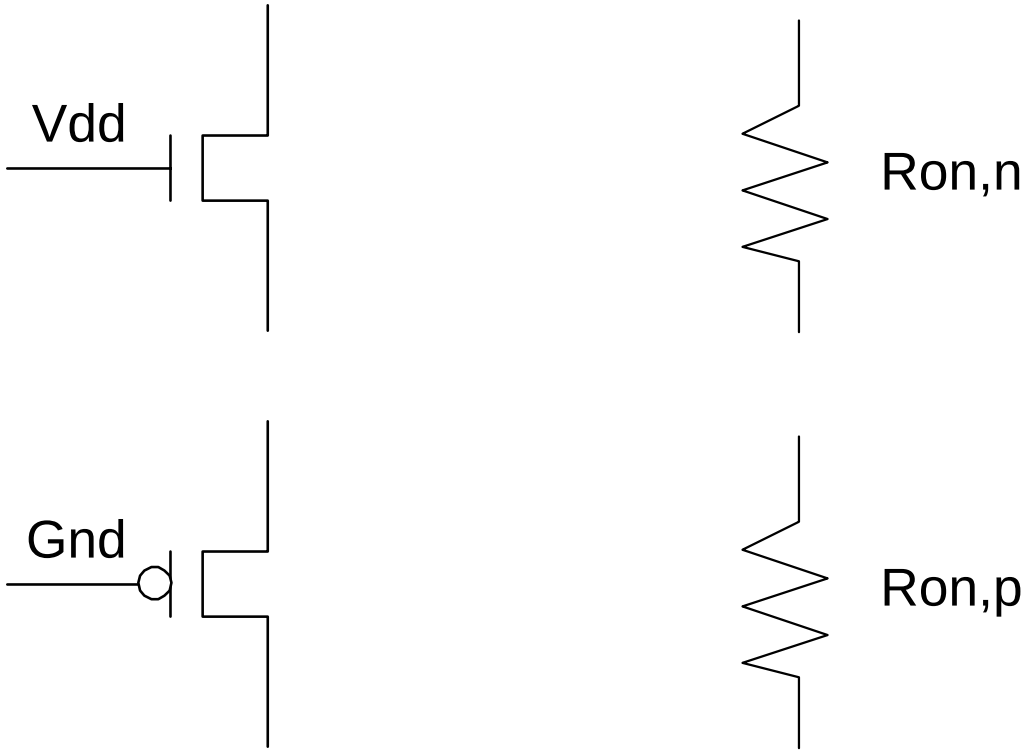
Transistor:

- Source to drain is really a resistor when a transistor is turned on
- Each transistor terminal (source, drain, gate) has some capacitance associated with it

Wire:

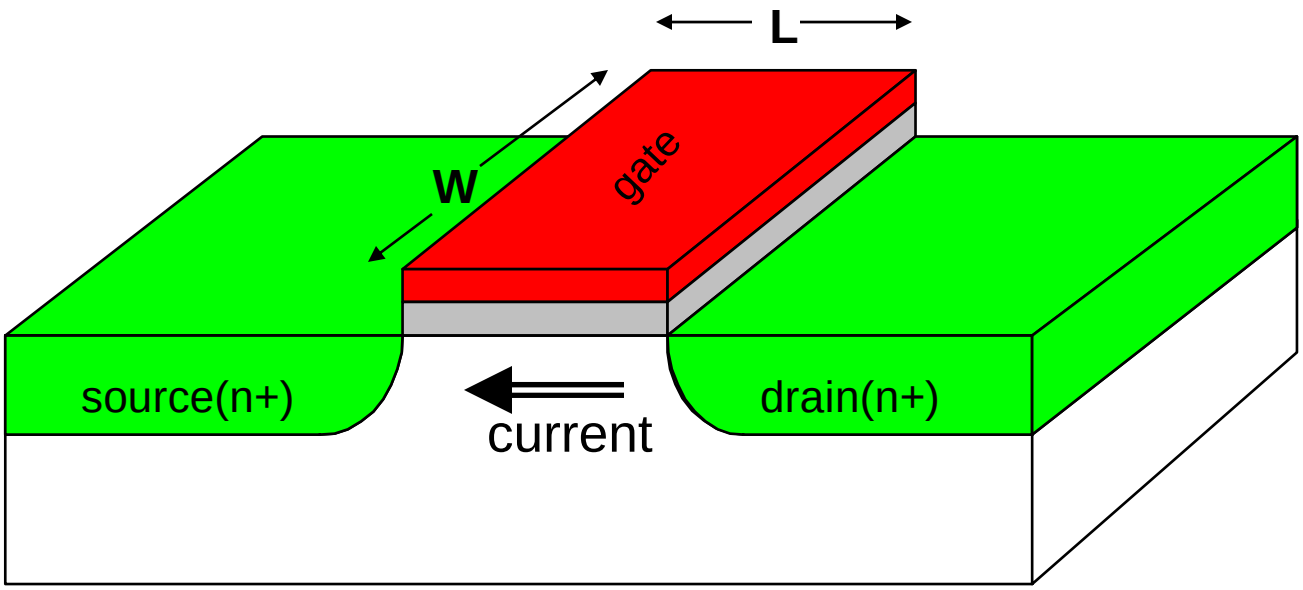
- Each wire is really a long resistor – different layers have diff. resistances
- Each wire has a capacitance per unit length

Transistor:



Question: how do you think the W and L affect the resistance?

Transistor:



As L increases….. Resistance increases

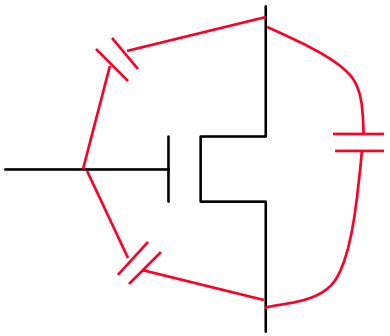
As W increases….. Resistance decreases

} R is proportional to L/W

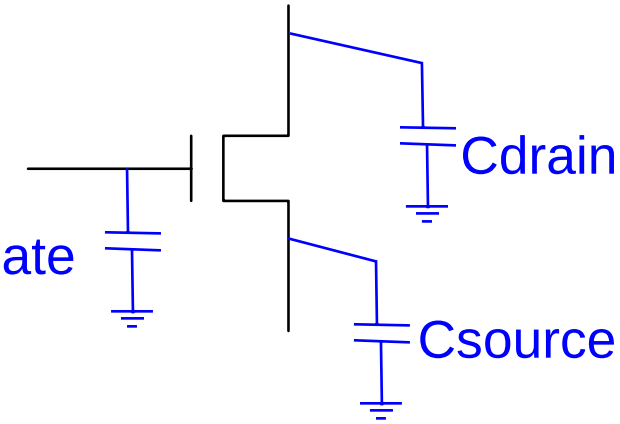
For our 1um technology, nMOS = 13Kohms * L/W,

pMOS = 26Kohms* L/W

Transistor:



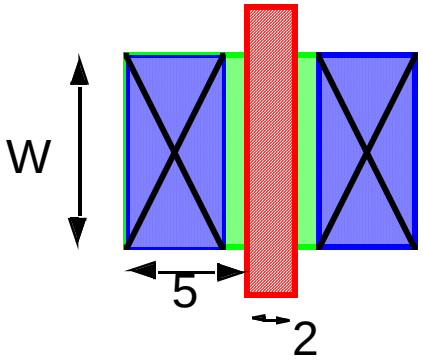
model as:



Simple Capacitance Numbers

Want to have numbers that make it easy to estimate the capacitance
Want the estimates to depend on the fewest number of parameters
Willing to make some approximations

Gate length is usually minimum (2λ, 1μm), width varies
Diffusion region kept small, size depends on transistor width
Give cap of these region as capacitance per unit transistor width



Simple Capacitance Numbers

For our 1um technology:

For a transistor with minimum L and diffusion region 5 or 6 um wide:

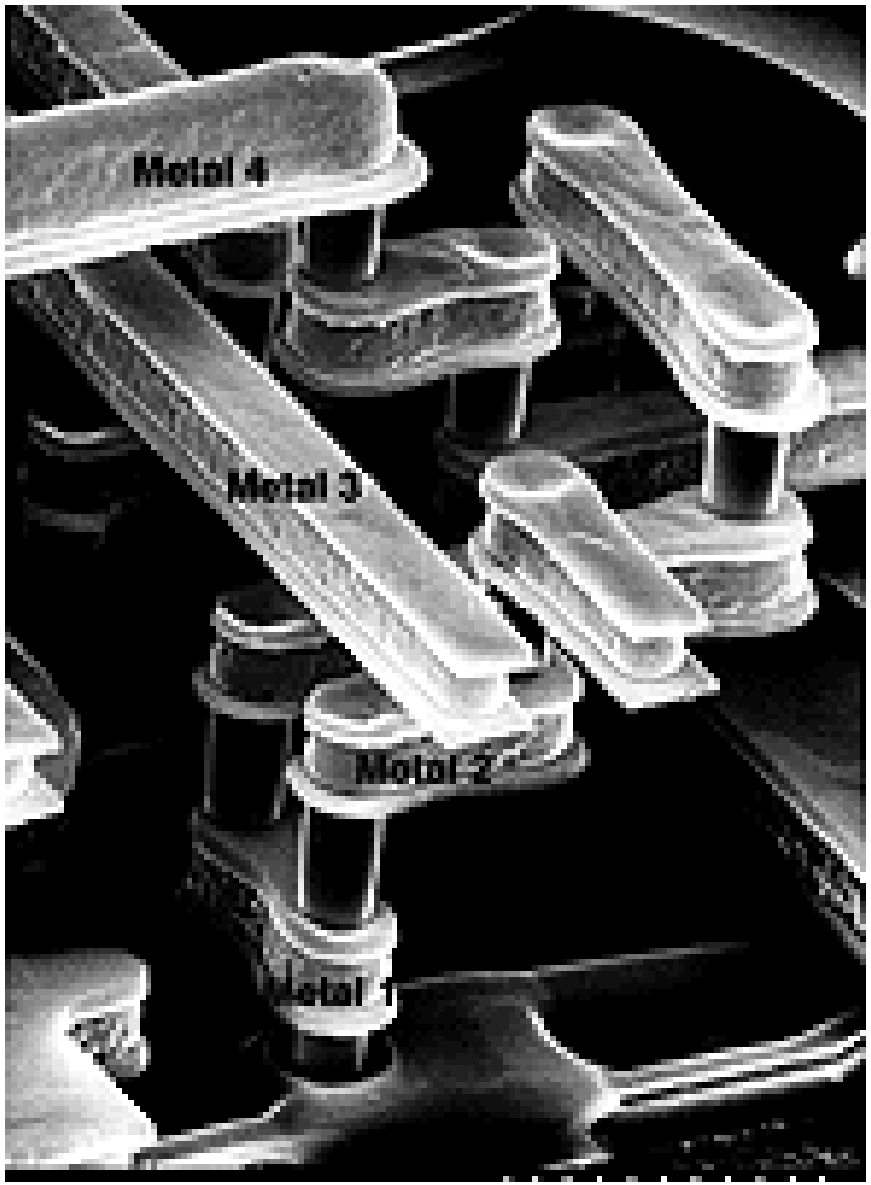
Transistor Cap	Capacitance per μ of transistor width
Gate (Poly over diff)	2.0 fF/μ
ndiff (5λ or 6λ wide)	2.0 fF/μ
pdiff (5λ or 6λ wide)	2.0 fF/μ

Exercise: Write the gate/source/drain caps for a transistor that has an L twice the minimum length (this is not common in digital circuits)

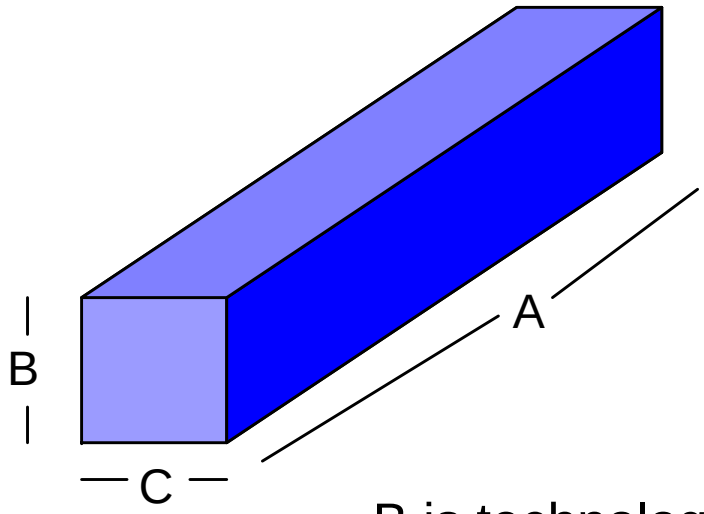
A word about our technology:

Today, a 1μ technology is a really cheap technology that students use, while advanced processes are running at 0.18μ to 0.06μ. We will use 1μ technology numbers for this class. This technology is different from the numbers in the book. The ratio of the various numbers does not change much with technology, but the absolute numbers do vary. You should always find the correct numbers for the technology that you will use before starting a design. And, since you don't want to extract the numbers by hand, make sure that the CAD tools have the right numbers too.

The metric that I will use in class, resistance/square for transistors, and capacitance/micron don't change much with technology scaling. For a 0.25μ technology Rsq of a nMOS device is 15K, pMOS is 36K, which is similar to the 1μ numbers. The cap/micron numbers are nearly the same. The reason the gates get faster is that the cap/lambda goes down, so the cap of a 10:2 device scales down, while the resistance remains constant.


Metal 4
Metal 3
Metal 2
Metal 1

Wires

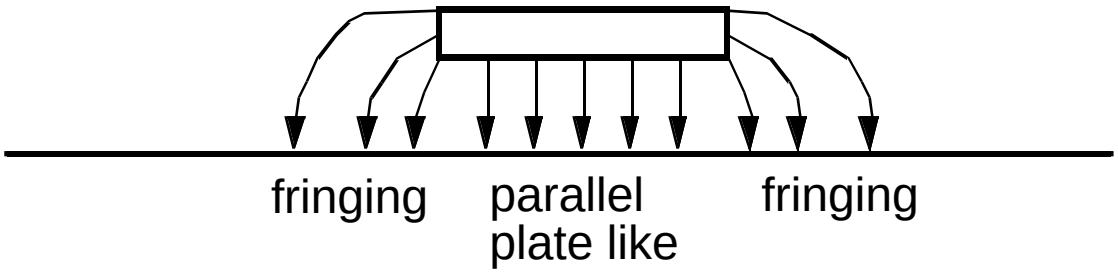


B is technology-specific

The designer has control over C and A

Resistance proportional to A/C

Wires



So, wires have two components to capacitance, one that is proportional to the wire's area, and the other proportional to the wire's perimeter. **For minimum width metal wires, the edge component is much larger than the area component**, so forgetting the edge is a large error.

The area capacitance depends on the thickness of the oxide between the capacitor plates, and that thickness depends on what is below it.

Wires

Most CAD systems have tools that take care of all this complexity by using large tables of numbers, one for each type of legal layer crossing. The tools take the capacitance numbers, multiply by the correct area and perimeter coefficients and then add all the numbers together.

That is far too much work for us to do by hand. Also it requires that the layout be finished to get the capacitance numbers. We often want to estimate the capacitance numbers to size transistors from either crude layout, or layout estimates.

Since most of the wires are minimum width, we will use an effective capacitance per running micron of length, assuming an average number of wire crossings. This number will include both the area (plate) and perimeter (fringe) capacitance terms. Diffusion is treated almost the same, but the width for diffusion we assume to be the extension of a transistor drain, and the length should therefore be the width of the transistor. This assumes that diffusion is only used to make a connection to a transistor, which is normally the case since the diffusion capacitance is so large.

.

Wires

For our technology:

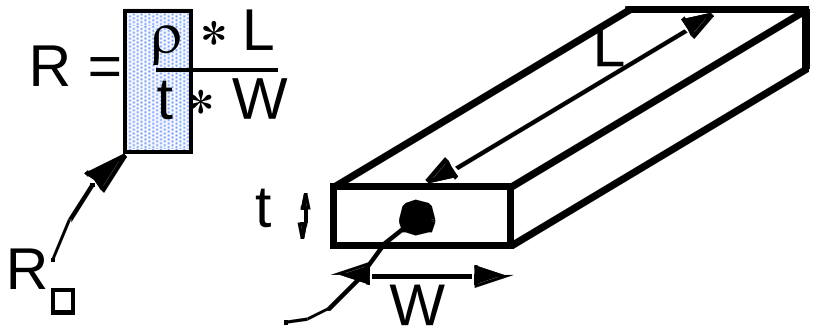
Wire Cap	Capacitance per unit length	Length when C = Cinv
poly wiring	0.2 fF/μ	40μ
metal1 (3λ or 4λ wide)	0.3 fF/μ	30μ
metal2 (3λ or 4λ wide)	0.2 fF/μ	40μ

Cinv is 8fF, the input capacitance of a 4λ:2λ nMOS, 4λ:2λ pMOS inverter

For modern technologies, wiring is becoming more and more important

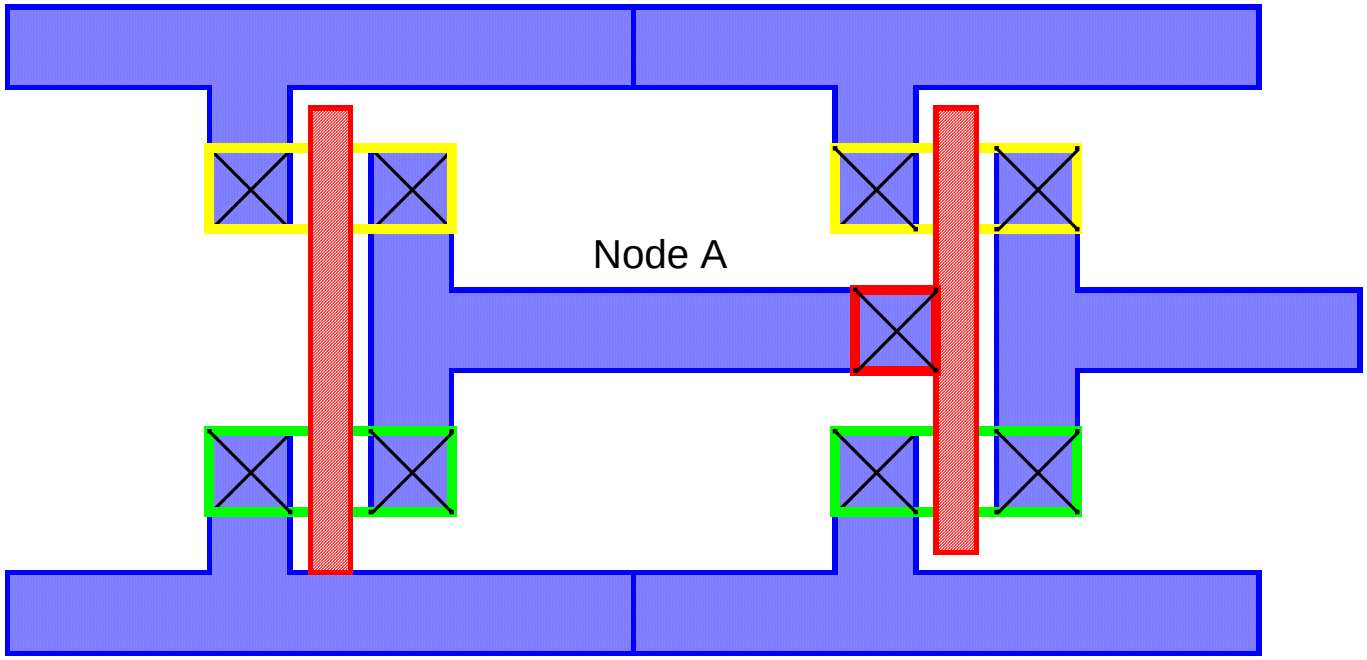
Wires: Resistance

Like a transistor, the resistance of metal or poly layer is given by Rsq times the number of squares.



	Rsq
metal	**0.05 Ω**
poly	**5 Ω**
ndiff	**5 Ω**
pdiff	**5 Ω**
NMOS	**13 KΩ**
PMOS	**26 KΩ**

Example:



Node A: 2 diffusion regions each 2μ (4λ), 2 gate regions each 2μ, 16μ Metal 1 (12λ vertical, 20λ horizontal), 14λ poly= 2*2μ*2fF/μ + 2*2μ*2fF/μ+16μ*0.3fF/μ + 7μ*0.2fF/μ = 8fF + 8fF + 4.8fF+1.4fF = 22.2fF

(wow, what an ugly slide. We'll work it out carefully in class)

Who cares about R and C?

Q = CV <-charge is proportional to the voltage on a node

This equation can be put in a more useful form

$$i = \frac{dQ}{dt} \Rightarrow i = C\frac{dV}{dt} \Rightarrow \frac{C\Delta V}{i} = \Delta t$$

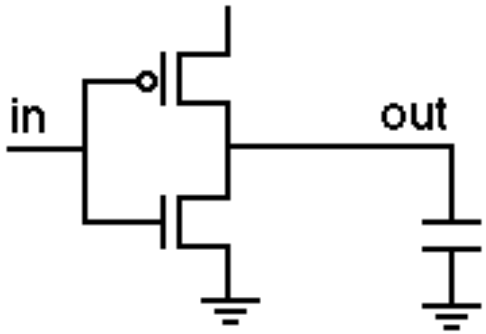
So to change the value of node (from 0 to 1 for example), the transistor or gate that is driving that node must charge (up, in our example) the capacitance associated with that node. The larger the capacitance, the larger the required charge, and the longer it will take to switch the node.

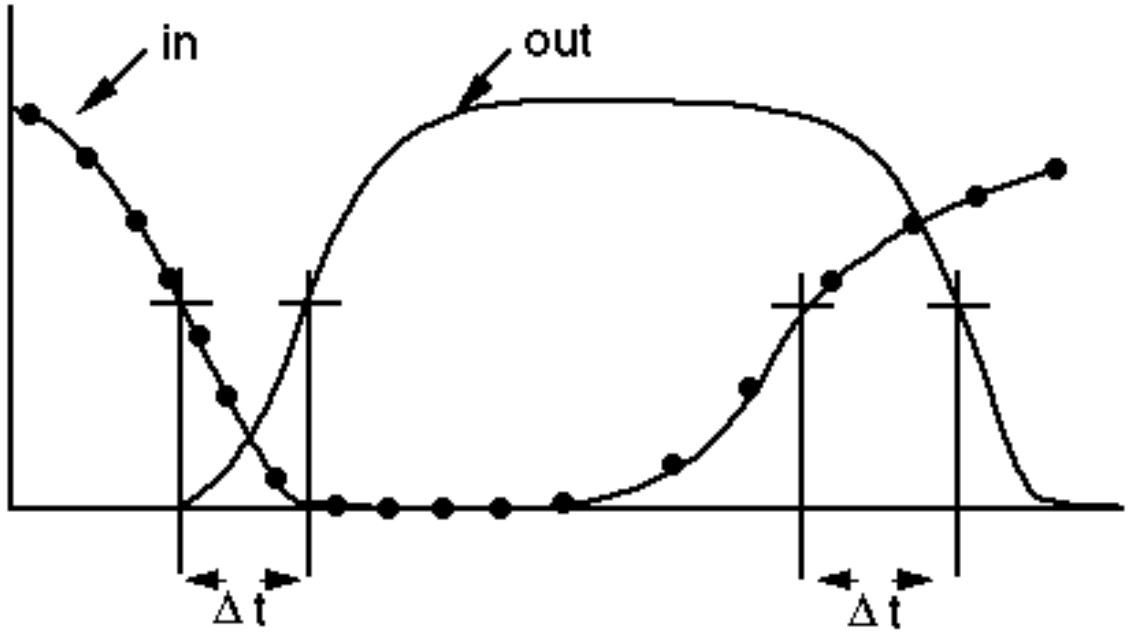
Since the current (i) through a transistor is approximately V/Rtrans

$$\Delta t = \frac{C\Delta V}{i} = \frac{C\Delta V}{V/R} = R_{trans}C$$

Simple Delay Model

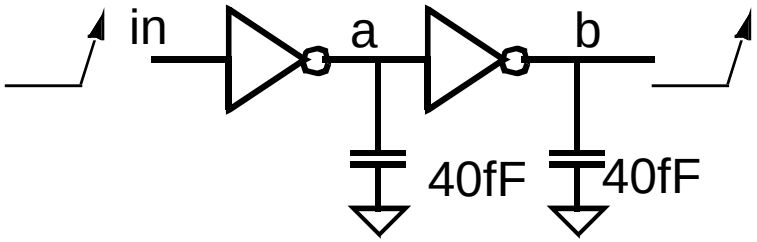
Delay measured from 50% crossing point on input and output swings, because need the same point to allow additive composition of delays.





Timing Example

Assume that pMOS are 4λ:2λ and nMOS are 2λ:2λ



40fF includes the diffusion and gate cap

When the ‘in’ rises, ‘a’ will fall:

delay = RC = 13K * 40fF = 0.52ns (nMOS transistor is on)

When ‘a’ falls ‘b’ will rise:

delay = RC = 26K/2 * 40fF = 0.52ns (pMOS transistor is on)

Total delay from ‘in’ to ‘b’ = 0.52ns + 0.52ns = 1.04ns

Board notes: - Transistor Sizing for equal rise/fall times

Need for Switch-Level Simulation

Need some way to check the gate/switch circuits

- Nice if the method could handle all legal switch circuits.
- Want to find common bugs in circuits, yet not produce false errors.
- Be fast and easy to use.

Tool should answer the questions:

- Does this pile of transistors do the logic function that I want?
- Are there any sneak paths, floating outputs?

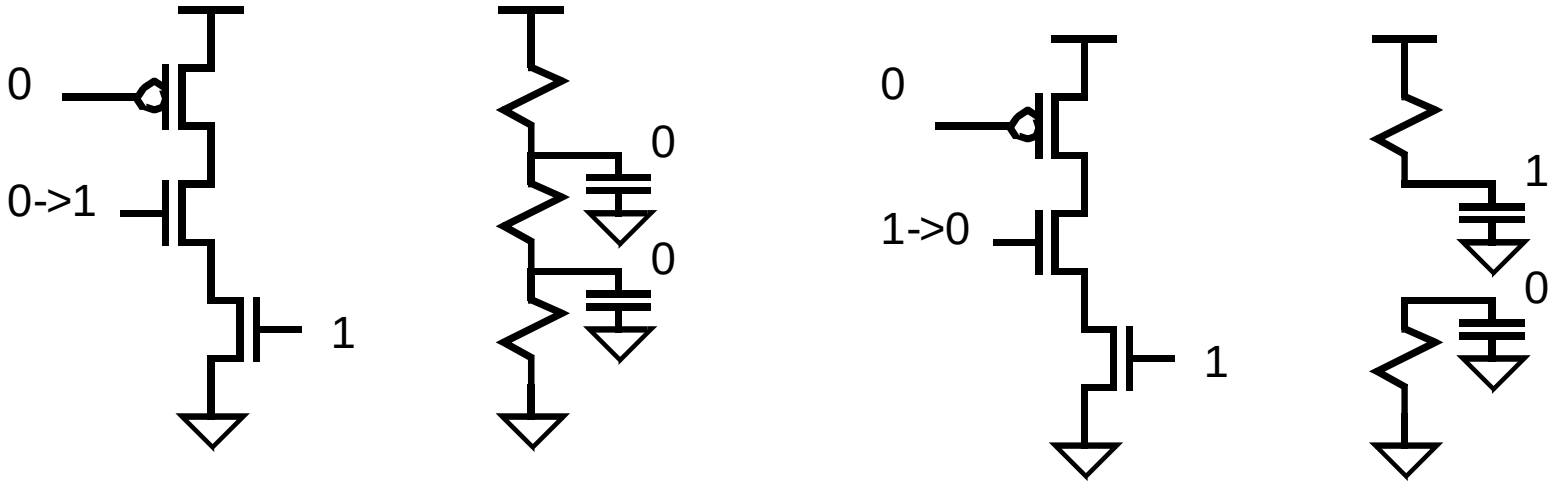
Switch-level simulation is one good way to answer these questions.

- Uses the same type of model we have been talking about in class:
- Nodes are modeled as capacitors
- Values on the nodes are 0,1,X
- Transistor is modeled as a switch in series with a resistor, where the value of resistor depends on the type of transistor

How Switch-Level Simulation Works

To find a value in a switch network:

- Build a cluster of connected transistors
- Walk out from a node through all the **on** transistors
- Replace all transistors by their equivalent resistance
- Replace all nodes by a capacitor, charged to the old value
- Solve the RC circuit for final value, delay



More details about the Simulation Algorithm:

For each event when a node changes:

1. Update the changed node to a new value
2. For each transistor with a gate connected to a changed node:
   A. Find the transistor cluster
   B. Find the DC values of the nodes in the cluster
   C. If, for any node, the new value is different than the previous value, calculate the delay of that node using RC equations and schedule the node to change after the delay

This is hard, because a node might have an X value. In that case, the simulator has to deal with ranges of values for a node

irsim

The Switch level simulator we will use is called irsim:

Finds the final voltage for each node in the circuit, correctly handling all ratios. All R,C, and intermediate voltage ranges are floating point computations. Quantizes to 0,1,X state only at the end of each event.

Delay (Quantized to 0.1nS for efficiency in scheduling events)

- Uses a better model than the one we have discussed in class.
- But it is conceptually similar

Correct charge sharing

irsim

Of course, because it is an approximation, the program is not perfect.

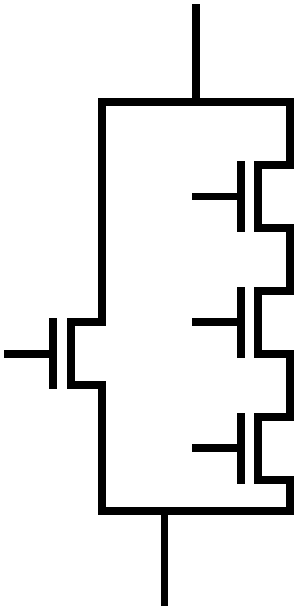
- Sometimes too generous with X values
  - propagation of X values can be too fast
- Some legal circuits will not simulate (but most digital circuits will be ok)
- But no tool is perfect, and irsim is fairly robust.

irsim Difficulties:

There are two kinds of problems that irsim has trouble handling:

1. Transistor Loops:

In this structure there are a set of transistors that form a loop, and neither end of the loop is a power supply. In this case the loop will be broken, (to convert it to a tree) and the simulation will continue. Note that multiple paths to supplies are ok, as are single transistor loops (CMOS transmission gates)

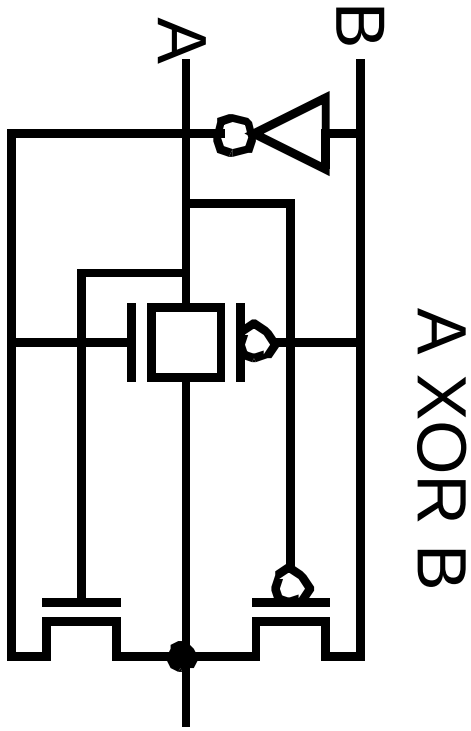


irsim Difficulties:

There are two kinds of problems that irsim has trouble handling:

2. Self-Connected Transistors:

These structures have the gate of a transistor connected to the same cluster as one of the outputs of the circuit. Since irsim needs to set the inputs to figure out the outputs, the program can have problems with this type of circuit. This circuit rarely comes up, except for the 6T XOR gate.



Using irsim:

- Read the manual page and the irsim hints page
- Manual page has complete list of commands, you really should read.
- The two required input files are a transistor parameter file (called .prm format) and the network connection file (called .sim format). Can also put multiple command files (preceded by '-') on the command line using the same commands as are listed for interactive use.

Using irsim:

- The .sim network description file is **flat** (no hierarchy), and has a simple format (documented in the .sim manual page). Each line is one of:
  - n gate_node source_node drain_node length[m] width[m]
  - p gate_node source_node drain_node length[m] width[m]
  - C node cap_value[fF] (adds cap between node and gnd)
  - C node1 node2 cap_value[fF] (adds cap between node1 and gnd, and another cap between node2 and gnd)

- Capacitors need only be added to model the wiring interconnect. The irsim program will automatically add gate loading (and source/drain loading too, depending on the **diffext** parameter in the .prm file)