EE371

Advanced VLSI Circuit Design

Mark Horowitz

Computer Systems Laboratory
Stanford University
horowitz@stanford.edu

Class Overview

- Centered on the “job of a modern circuit designer”
  - Not just arranging transistors
  - Also dealing with wires, signal integrity, debugging…
- Builds on EE271 and EE313 (required prerequisites)
  - I assume you know: MOS models, logical effort, memory design
  - I also assume you know: running SPICE, writing UNIX scripts
  - As in EE271, we will examine large functional blocks like adders
  - As in EE313, we will use SPICE a lot
- Looks at the art of circuit design
  - Simulating large blocks in SPICE is not fun
  - We will focus on ways to evaluate circuits without simulation

Course Goal

- Question: What is the **right** circuit solution to a problem?
  - Usually there are many techniques that don’t work. (Ignore those.)
  - But there are also many techniques that **do** work. Which is best?
  - “Best” solution depends on the application, and changes over time

- Goal: Give you tools to understand circuit tradeoffs
  - For example: “Top performance” != “lowest power”
    - Pick the right set of constraints
  - No circuit is perfect; all circuits have warts
    - Pick the right set of warts for the application and the constraints
  - Know what to optimize for…and know when you’re done
    - Pick the right design and verification methodology
    - And know when you should use an existing solution

Caveats

- People lie
  - Intentionally, accidentally, or perhaps you just heard them wrong
  - Never take a circuit on blind faith; always work it out yourself
  - Many clever published circuits don’t work or are very sensitive

- (Corollary) Don’t trust me
  - I will tell you what I know, but I have my own biases and blind spots
  - If you think what I am saying is wrong, please tell me

- SPICE is no substitute for thinking
  - SPICE can make your life much easier **or** much harder
  - Remember, garbage in…

Class Topics

- First, establish some fundamental tools
  - What to model and what to verify
  - MOS device models, wire models, transistor matching and scaling
  - Worst-case design, energy efficient design, corners, and margins

- Second, consider circuit “environments”
  - Clocked elements, clock distribution, asynchronous circuits
  - Power delivery and dissipation, low-power design
  - Signal integrity, noise, test and debug

- Third, examine large circuit structures
  - Adders, multipliers, FIR and DFE
  - High-speed IO

Class Logistics

- Classes Monday and Wed, 11am-12:15pm, Gates B03
  - Office hours: Monday 12:15-1pm and Wed 8:30-9:30 Gates 306
  - Handouts and notes are available
    - Gates 3rd floor cabinet; grab them quickly
    - http://eeclass.stanford.edu/ee371
  - Announcements, class signups on web page
  - Please come to class
    - If you don't, nobody asks questions, and nobody learns as much
    - I promise to keep you awake, and have random acts of kindness ☺

- TA is Tom Deane
  - TA office hours: TBD
  - Review session: TBD

Tools and Books

- We will use mostly three CAD tools
  - SUE for schematic entry
  - HSPICE for simulations
    - (SITN) If you don’t have access to HSPICE, let me know ASAP
    - We’ll use a Berkeley model of a 90nm CMOS technology
    - We’ll provide “corner” transistor models
  - Verilog for logical design

- No textbook is required
  - You are responsible for lecture notes and all reading handouts
  - Very good supplementary reference is available in bookstore
    - *Design of High-Performance Microprocessor Circuits* (IEEE, 1991)
  - Weste/Harris, and Hodges are good references

Grading

- Expect 4 homeworks, two midterms, and a final project
  - One “midterm” will be during finals week (*i.e.*, a 1:15-hour final)

- Homeworks will be handed out Wed and due next Wed
  - They will give experience with technology and design techniques
  - No late homeworks
  - Some homeworks will have 1-2 open-ended design problems
    - As well as some analytical pencil-paper questions
  - All turned-in work must be your own
    - But I encourage you to discuss the homework with your classmates

- Midterms are in-class, closed book, one sheet of notes
  - May 2 (but date is subject to change) and last week of class.

Project

- Design front end processing for high-speed link
  - You will look at the energy/delay trade-off for your design
  - More details to come later…

- What you will deliver
  - Working Verilog
  - Some critical path sims, characterization of your cells
    - Some noise analysis work as well
  - Written report and short in-class presentation detailing the project

- Work in teams of 2 people
  - Try to find a partner early in the quarter
  - Find somebody you can work with soon
  - Project will start next week!

Honor Code

- Please remember you are bound by the honor code:
  - I will trust you not to cheat
  - I will try not to tempt you

- But if you are found cheating it is very serious
  - There is a formal hearing
  - You can be thrown out of Stanford

- Save yourself and me a huge hassle and be honest

Lecture 1

Models, Simulation, and Circuit Design

Computer Systems Laboratory
Stanford University
horowitz@stanford.edu

Readings

Readings

Siridopoulos Modern Chip to Chip IO

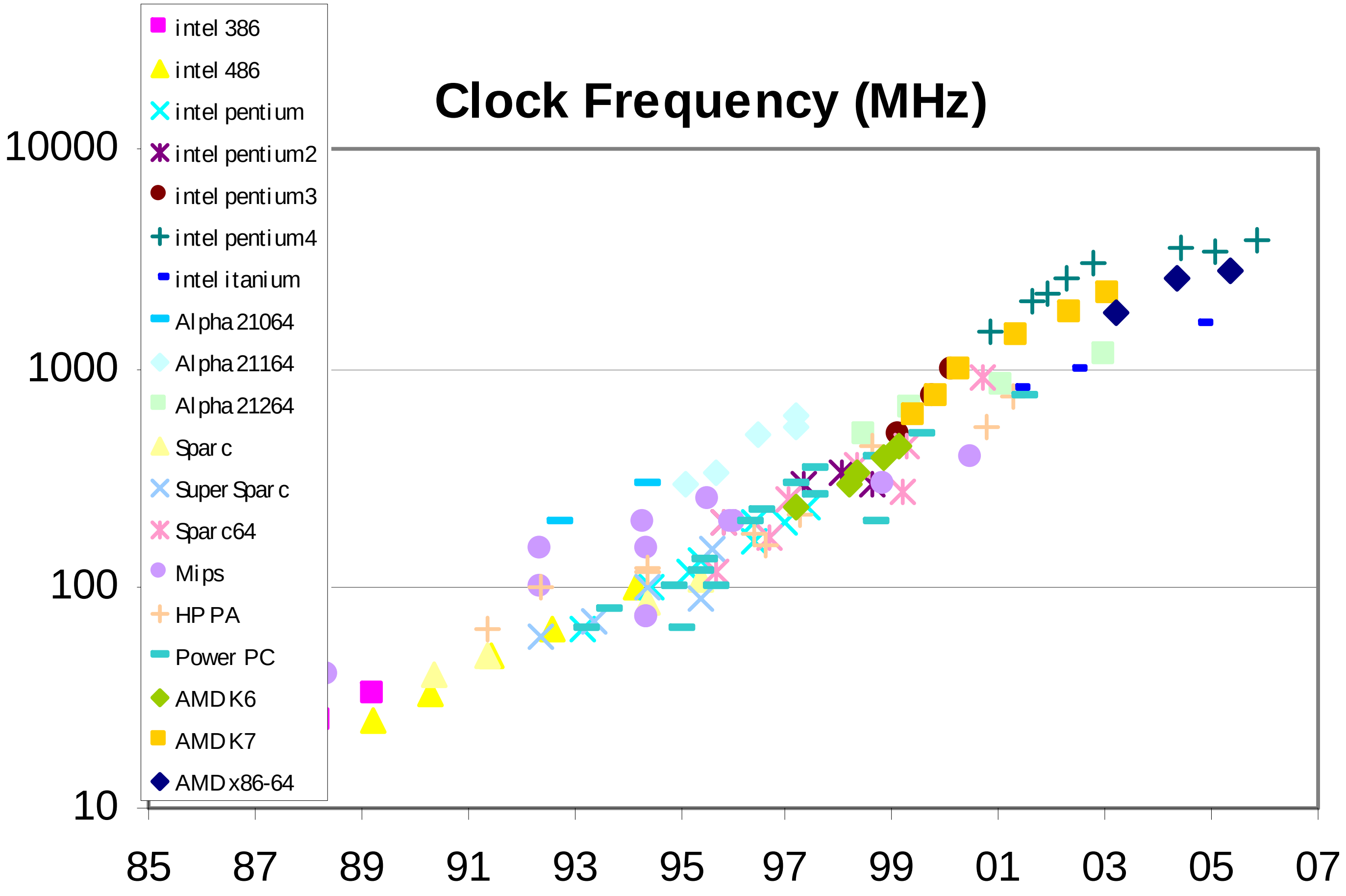
Chapter 19 Chandrakasan (on web page)

B. Colwell, “Phrenology, Astrology, Risk, and Product Quality,” *At Random column*, IEEE Computer, January 2002, pp. 12-14.

Notes:

- Readings usually on the website, (the first one is there)
  - The third one you will have to get from IEEE Explore
    - http://ieeexplore.ieee.org
    - Look at it from within Stanford for full access to the site
- I’ll typically only assign technical readings
  - But sometimes I’ll list a more entertaining article, like this one
- Readings will often refer to next lecture
  - Read before coming to lecture, so you can follow the material

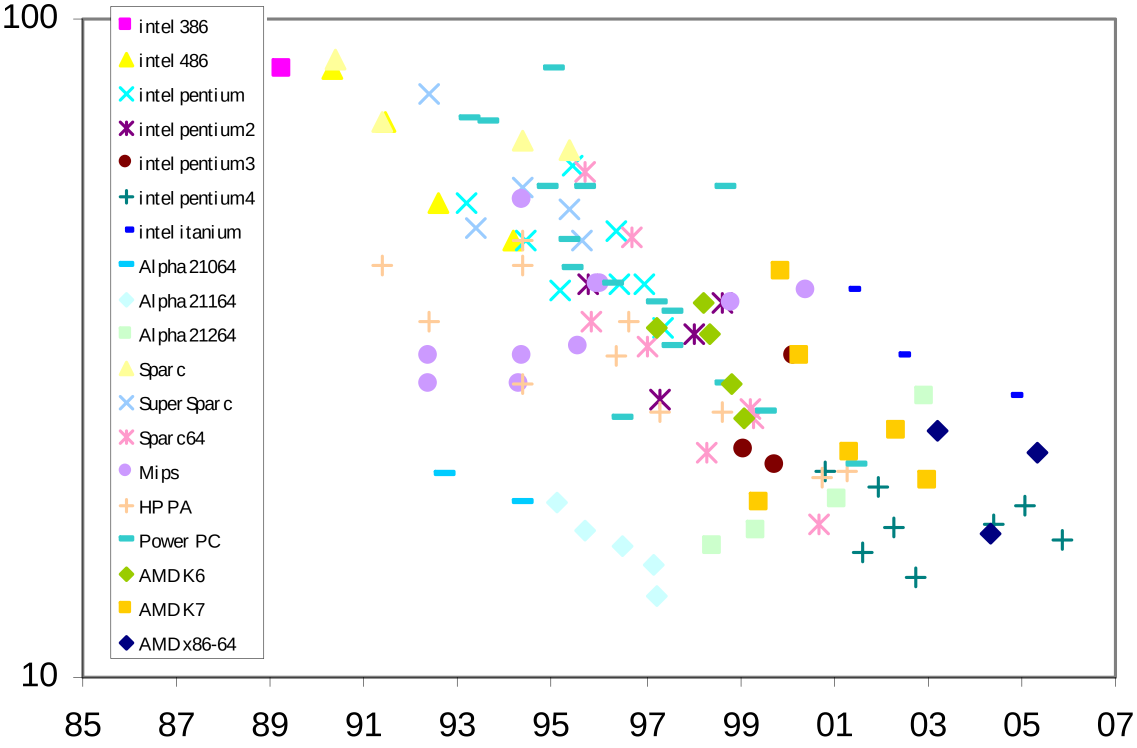
VLSI Trends – Clock Rates Scaling Up



- Not just due to faster technologies

VLSI Trends – Cycle Times in FO4 Falling

- In terms of FO4 (fanout-of-four inverter delay) – process independent

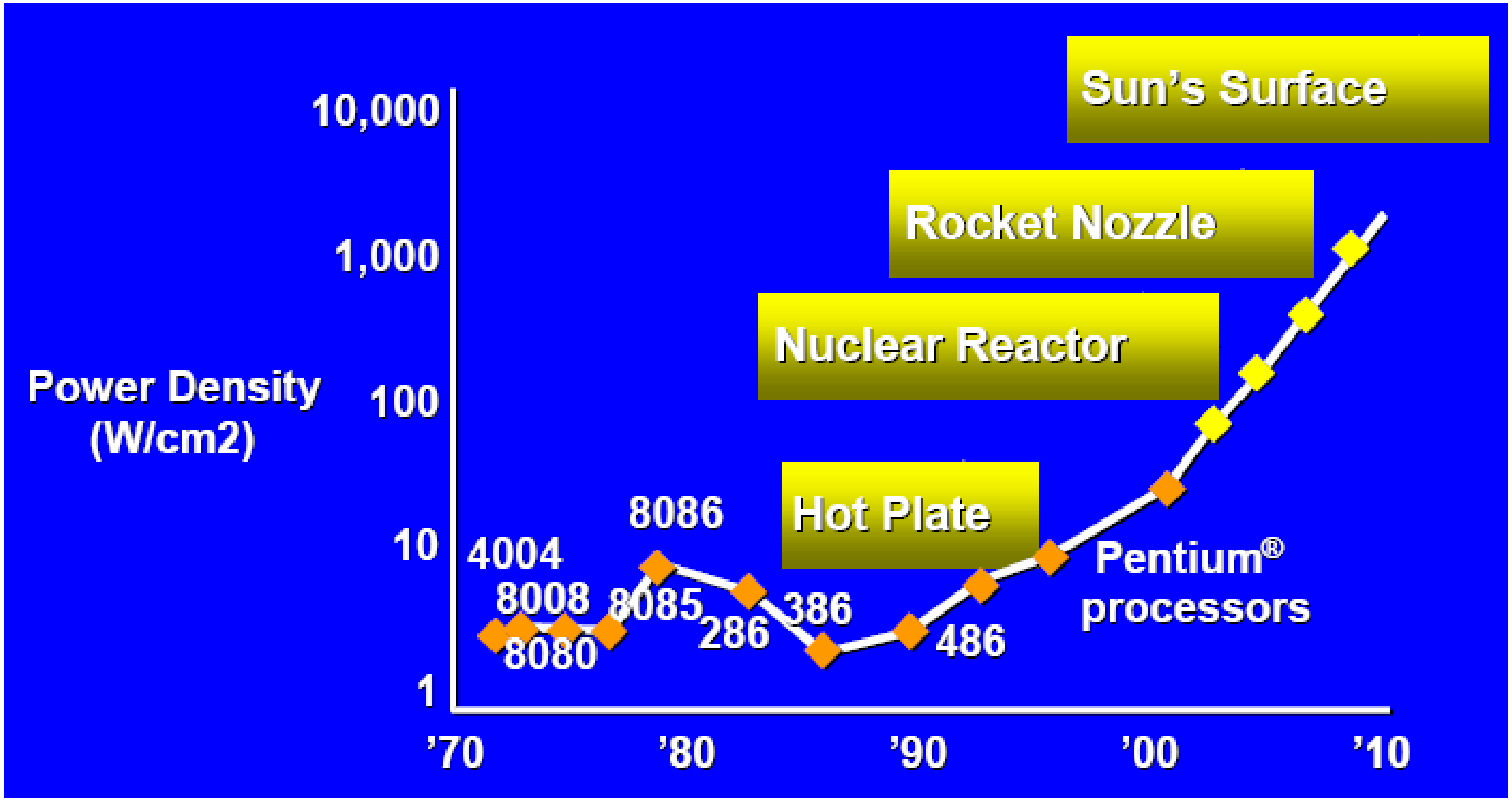


- We need to build functional blocks that run faster than last time
  - By “faster” I mean relative to the basic underlying gates

How Do We Build Faster Functions?

- Build faster logic gates
  - Use experience with transistors to craft faster basic gates
  - Some examples include dynamic logic and passgate logic
  - Customize the layout to make the critical wires short
- Build faster architectures
  - Leverage all those additional transistors that are possible
  - Do more things in parallel, do some speculative calculations
  - Simply put, throw lots of (cheap) transistors at the problem
- Today faster architectures are the norm
  - Circuit designers need to speed them up, too
  - And they need to cope with designs with lots of gates and wires

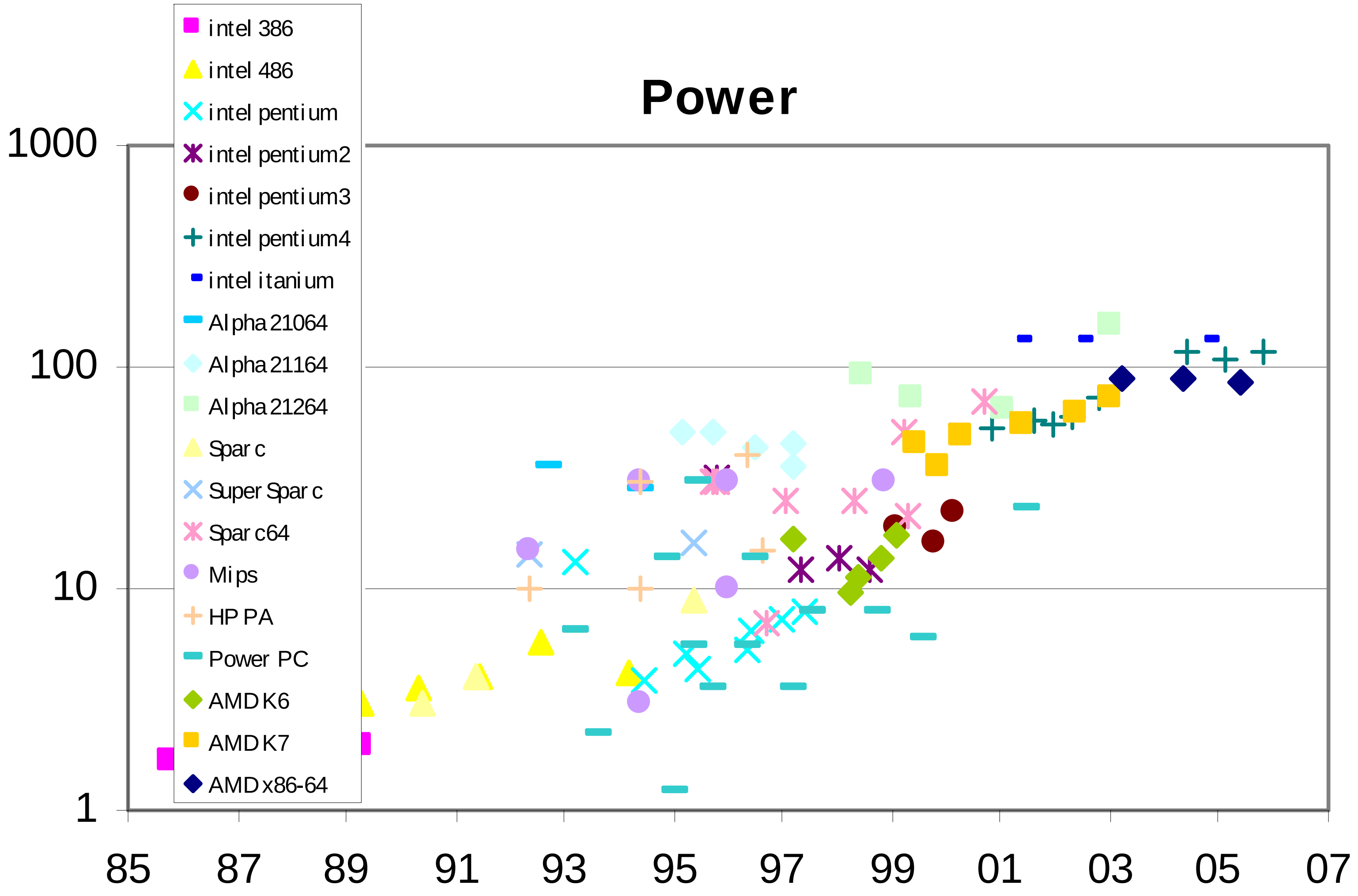
VLSI Trends – Power Density



Source: Gelsinger, Intel, ISSCC2001

- We also need to build functional blocks that use less energy

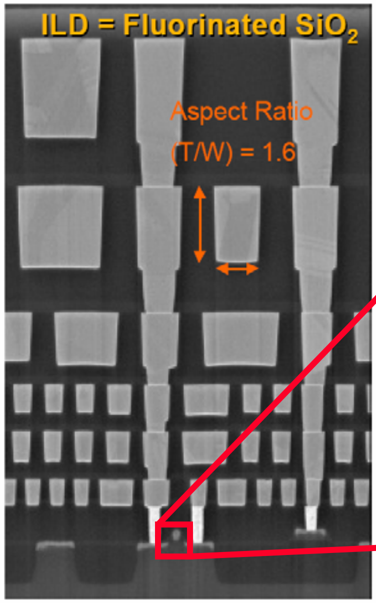
Of Course That Did Not Happen

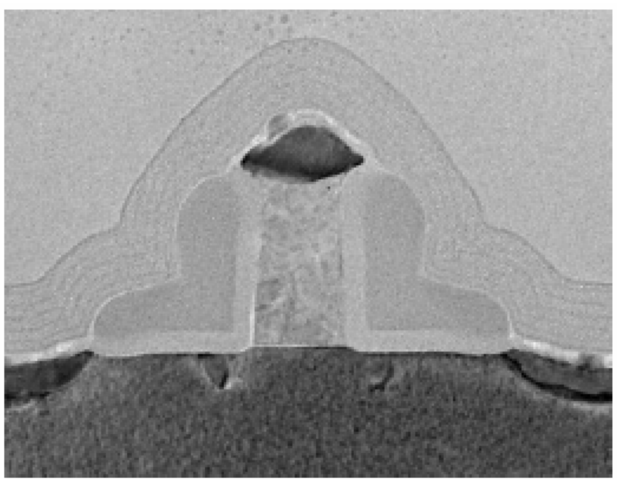


But The Design Objective Changed

- Before 2000 most design focused on performance
  - Could not dissipate too much power
    - If you used CMOS and were not stupid
- With scaling, and circuit improvements
  - Almost all designs are power limited
  - Problem is going to be worse moving forward (Vdd is not scaling)
  - High performance now implied high power efficiency
- We will look at this issue in the class

Review: The Problem





Source: Intel Technical Journal, May 2002

- ICs are really complex and contain billions of tiny 3D structures
  - Have lots of interleaved conductors and insulators
  - Create transistors from semiconductor impurities and oxides
- We could think about circuits in terms of 3D E-fields and carriers
  - But it would take way too long to reason about a single transistor

The Solution: Models

- Models are approximations to the real world
  - To be useful, must leave out many details
  - To be useful, must also retain the important ones
  - What level is appropriate depends on the question being asked

- Need to use models to simulate anything
  - Even using 3D E-fields and carriers would require models
    - For carrier behavior in a crystalline solid
    - Based on Maxwell's equations

- Complex simulations require "bigger" base models
  - For circuits, simulate transistors, not carriers
  - For big circuits, simulate logical functions / abstractions of circuits
    - That is why we will start with verilog

The SPICE Circuit Level Model

- SPICE uses simple lumped element models
  - Quasi-static: sizes small compared to wavelength
  - Approximate device behavior by terminal characteristics
  - All connections in the model are equipotentials

- Major equation

$$i = C \cdot \frac{\Delta V}{\Delta t} \Rightarrow \Delta t = C \cdot \frac{\Delta V}{i}$$

$$\Delta t = \frac{\Delta Q}{i}$$

Charge-control model

  - We model devices by their terminal I-V and C-V behavior
  - Sometimes we need inductor models, too…

What Should We Model?

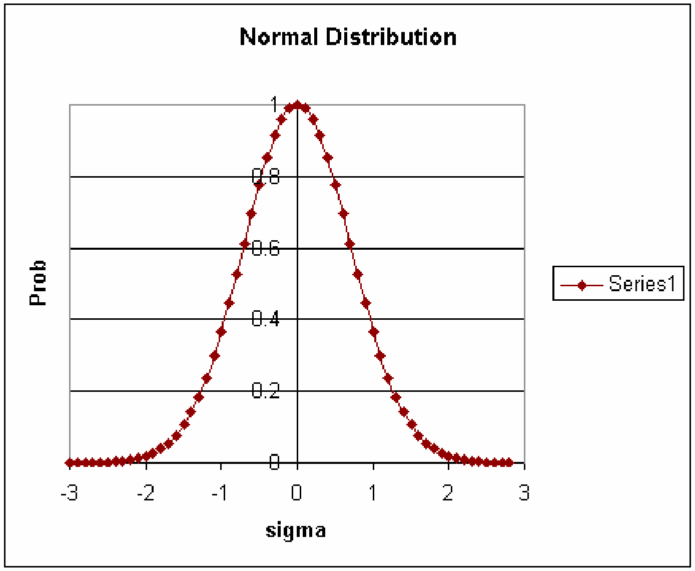
- Transistors
  - nMOS, pMOS
- Wires
  - No longer ideal connectors
    - Resistance, capacitance, and perhaps inductance
- Circuit Environment
  - Temperature, Power Supply, Substrate Voltage
  - Package power and ground distribution
  - Heat generation and flow
- EE313 focused on transistors, but …

Parameter Variations

- We've been talking about transistors as if they're all the same
  - But no two transistors are exactly the same
  - They vary from wafer to wafer and from die to die

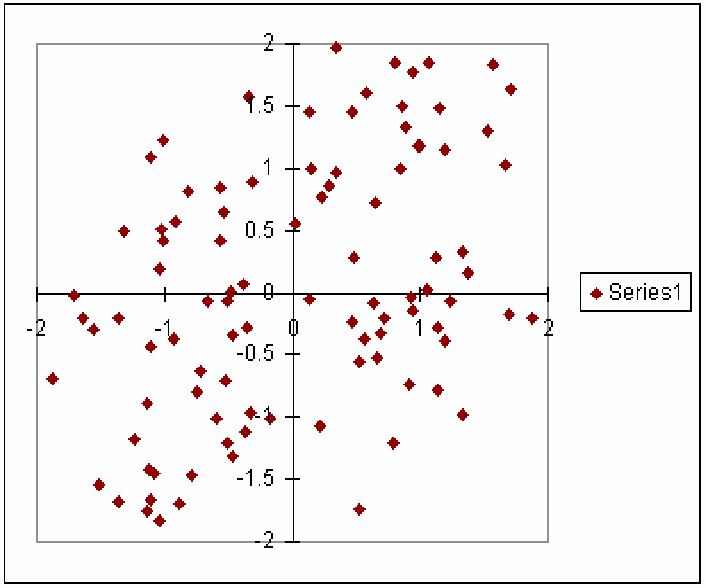
- Parameters of a fabrication run generally normally distributed

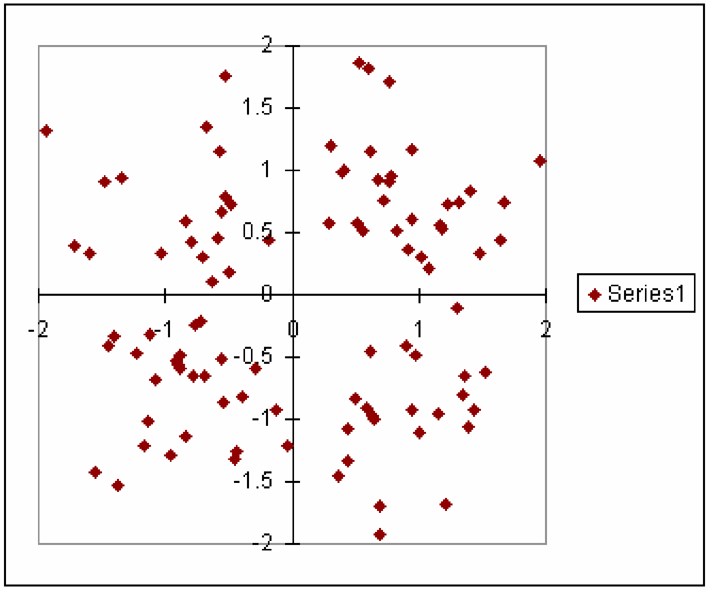
- Extract data from real wafers
  - 3-σ (or 4/5/6-σ) parameters
  - Use it in design



Multiple Variations

- But really there are many parameters (V_{th}, doping, linewidth…)





- Sometimes they are correlated, and sometimes they aren't
  - Example: T_{ox} affects nMOS/pMOS similarly, giving some correlation
- Remember that each transistor is unique
  - Not all the transistors on your chip will be affected the same

Parameter Variations

Variations come from many sources

1. Die to die variations
   - All devices in the die are correlated
   - Processing for this die/wafer varies from die to die and run to run

2. Across die variations
   - Two transistors on die have different parameters
   - Caused by many layout proximity effects
   - Across die processing variations

3. Random variations
   - Random dopant fluctuations, line edge roughness

1 used to dominate, but with scaling 2 and 3 are comparable issues

Process Corners

- Extreme points in parameter distributions (spec limits)
  - What is “extreme”? Often “3-σ die-to-die” variation
  - Circuits get stressed when operating at these places
  - Good place to test your design

- Generally, four attributes define a process corner
  - nMOS performance (L_{eff}, T_{ox}, V_{th}, etc.)
  - pMOS performance (L_{eff}, T_{ox}, V_{th}, etc.)
  - Temperature (0-100^{o} C)
  - Operating voltage (+/- 10% of nominal V_{dd})

- Can make corners more complicated if you wish
  - Add wire attribute as well

EE371 Corners

- We write our corners with a 3-letter code
  - nMOS and pMOS can each be **S**low, **T**ypical, **F**ast
  - V_{dd} can be low (**S**low devices), **T**ypical, or high (**F**ast devices)
  - Temp can be cold (**F**ast devices), **T**ypical, or hot (**S**low devices)

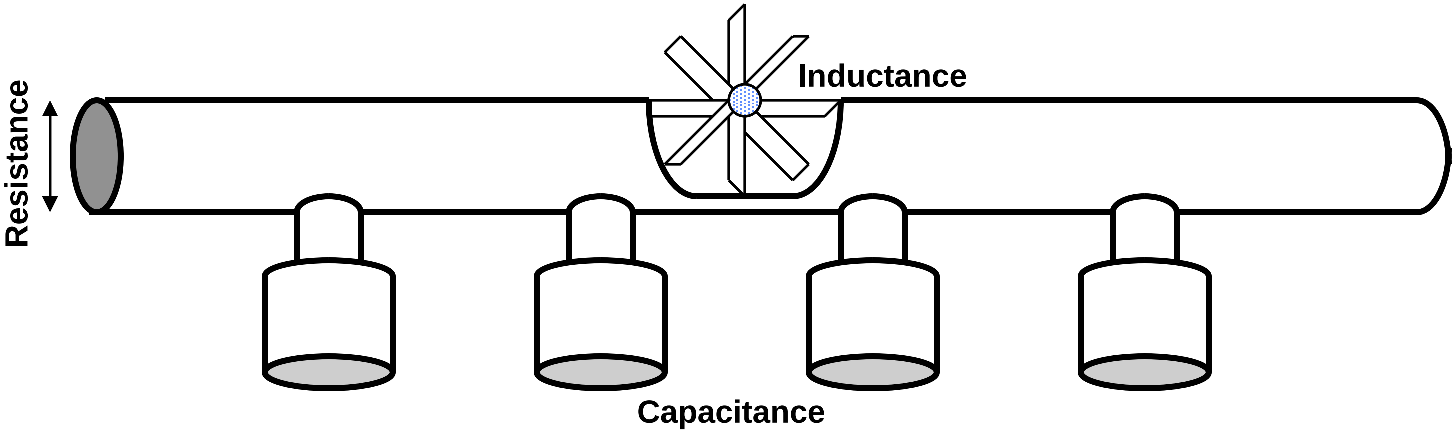
- Example: TTSS corner
  - Typical nMOS
  - Typical pMOS
  - Slow voltage = Low V_{dd}
    - Say, 10% below nominal
  - Slow temperature = Hot
    - Say, 100^{o} C → junction temperature

Wire Models

Wires have three important characteristics

- Resistance - relates i to V (carrier flow)
- Capacitance - relates charge (Q) to V (electric energy)
- Inductance - relates flux to i (magnetic energy) ←can usually ignore

Water analogies are fun…



Resistance

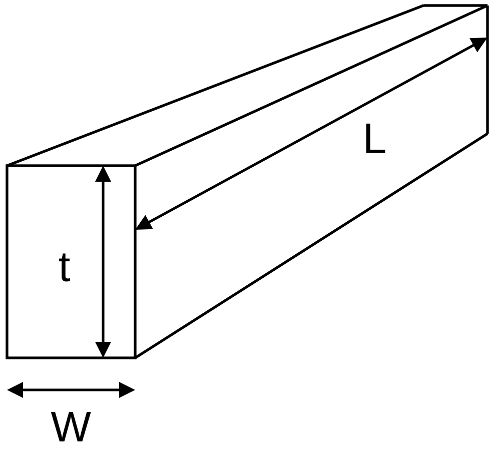
Easy to think about resistance: current is confined to material

Intrinsic resistance is small, so only worry about “long-ish” wires

- And they have a nice rectangular shape
- Complex shapes are difficult to model

Formulae are simple

- $R = \rho/t * L/W$
- $R_{sq} = \rho/t$ (sheet resistance)
- $R = R_{sq} * L/W$



Wire Resistance

The thickness of the wire is fixed by the fab, not by the designer

- So characterize wire by R_{sq}
- Aluminum: $R_{sq} = 0.03\Omega*\mu m/t_{metal}$
- Copper: $R_{sq} = 0.02\Omega*\mu m/t_{metal}$
  - But beware: Copper would normally diffuse into the surrounding SiO_2
  - We use barrier metals that reduce the thickness from what you expect

Metal thickness must shrink w/ pitch (or you get stalagmites)

- So maintain a mix of fine pitch **and** thick (faster) wires

Different classes of wires have different properties

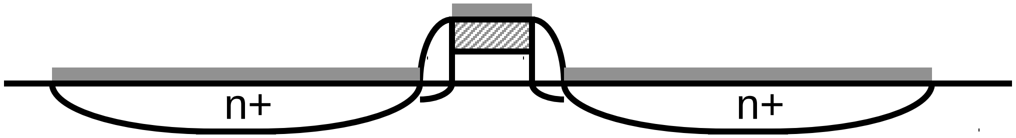
Metals 1,2	Finest pitch, thinnest metal, used only in cells
Metals 3..N-2	Semi-global wires, wider pitch and thicker
Metals N-1,N	Global wires for power and signals, thickest

Wire Resistance For EE371

EE371 wire models have seven layers of metal

Layer	Min Pitch	Thickness	R_{sq}
M1, M2	0.30	0.30	.073
M3,4,5,6	0.40	0.45	.055
M7	1.0	1.2	.018

Silicide



Silicide is a metal-silicon alloy

- Formed to provide lower effective diffusion and poly resistance

Salicide is Self-Aligned silicide covering diff and gate

- Covers all the way up to the diffusion edge b/c aligned to spacers

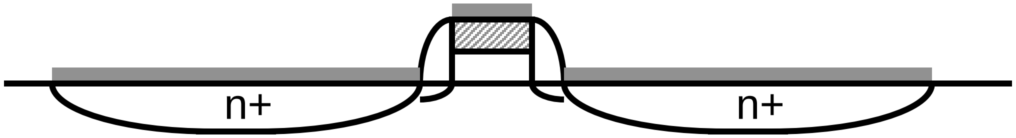
Thickness is not well controlled in general

- Fabs provide an upper bound, not a lower bound

Building an explicit resistor is hard in many processes

- Every resistive layer (diffusion, poly) has silicide on it by default
- Need to use a special mask to block the silicide

Other Resistances



Carriers see a series resistance in getting to the channel

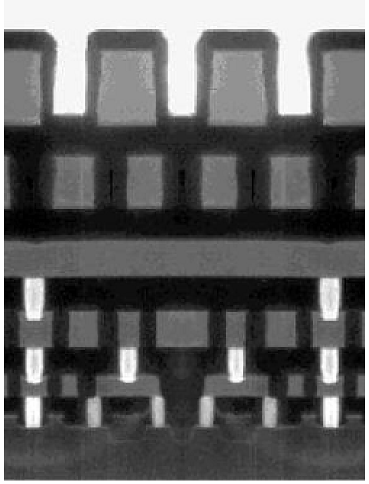
- Caused by the spacer and the lightly-doped diffusion

Contacts between layers have resistance too

- Values are very process dependent
- Depends on the metal used (W versus Cu)

Assume a couple of ohms for each contact (2Ω)

- To get low resistance, you need many contacts
- Need many contacts for current density anyhow



Capacitance

Two simple models for capacitance

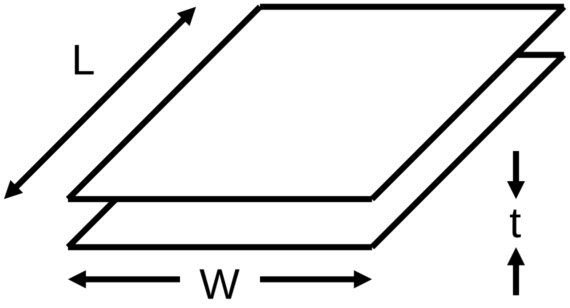
- Parallel plates for the capacitance of a structure's **area**
- Cylinders for the fringe capacitance of a structure's **edge**
- Some combination of the two works well for most real objects

Parallel plates can be solved with Laplace's equation

- Assume E-field constant vertically (infinite plate approximation)

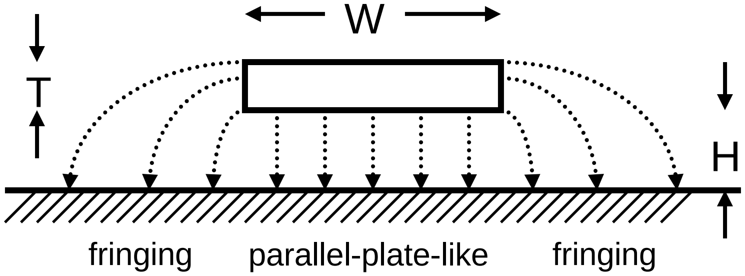
$C = \varepsilon L * W/t \quad (\varepsilon = 0.0345\text{fF}/\mu\text{m})$

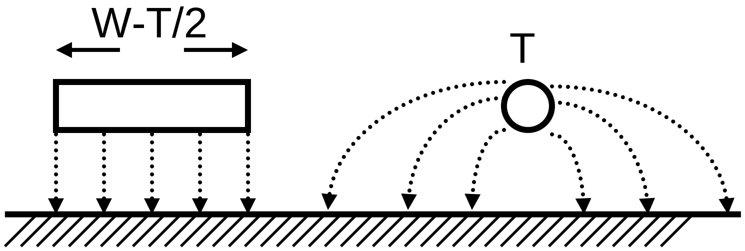
- Note that ε/t is fixed by the foundry
- $C = C_{\text{per-square-micron}} * W * L$



Fringe Capacitance

Yuan and Trick (1982) used a simple model for fringe





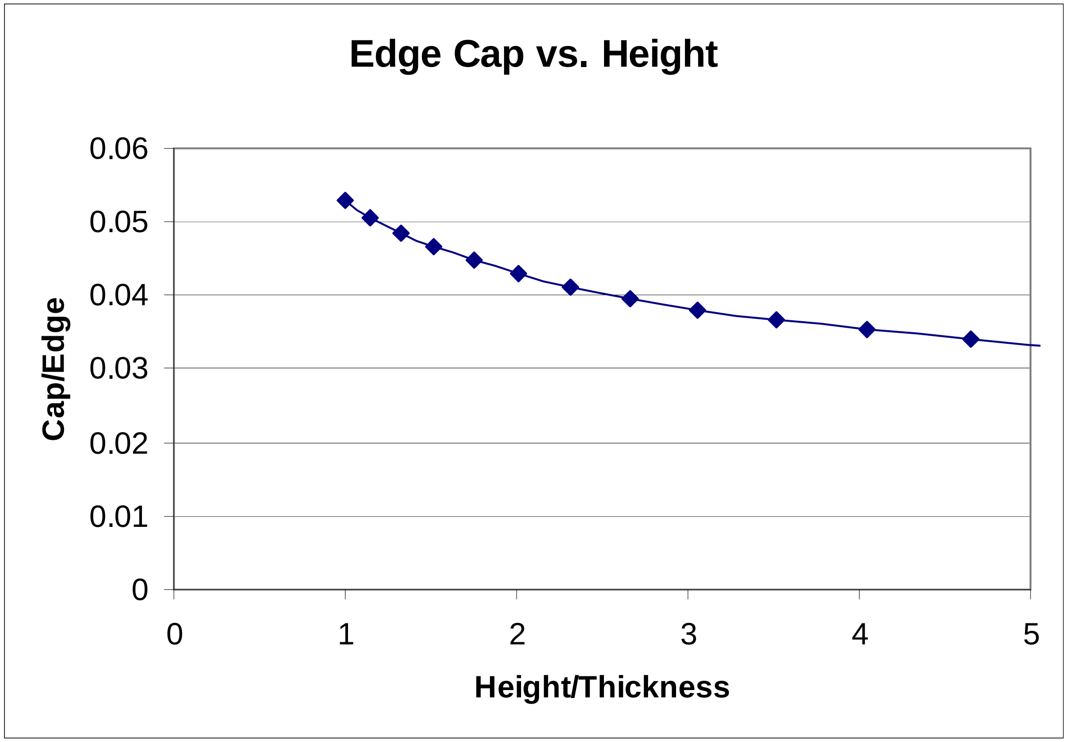
$$C_{edge} = \epsilon L \left(\frac{\pi}{ln\left(1+\frac{2H}{T}\left(1+\sqrt{1+\frac{T}{H}}\right)\right)} - \frac{T}{4H} \right)$$

Fringe Capacitance Does Not Vary Much

For a reasonable range of H/T (1 to 5) the fringe is pretty constant

0.035fF/μm edge to 0.05fF/μm edge

Suggestion: use 0.05fF/μm for hand calculations (each edge)



Pizzas Versus Sandwiches

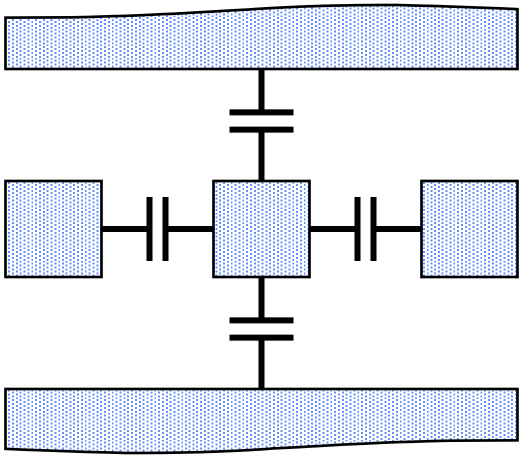
So far: wire cap for wires resembling pizza toppings

As if on the top layer with nothing above them

Many wires fit into the sandwich model better

Four parallel plate caps from neighboring wires and ground planes

Upper and lower neighbors may not be perfect “planes” (but close)



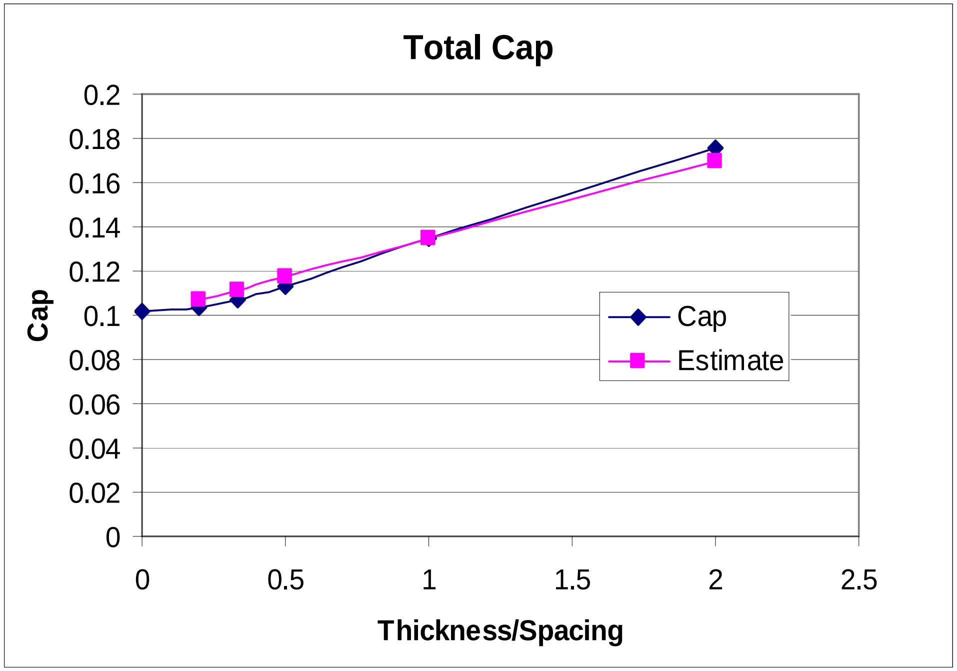
Total Capacitance

For calculating the total wire capacitance

Add fringe (0.1fF/μm) to four parallel plate capacitors

Seems surprising, but it does fit the data

And is easily remembered


Total Cap
Cap
Thickness/Spacing
Cap
Estimate