

---

## Lecture 9:

# More about wires and wire models

Computer Systems Laboratory  
Stanford University  
ronho@vlsi.stanford.edu

Copyright © 2007 Ron Ho, Mark Horowitz

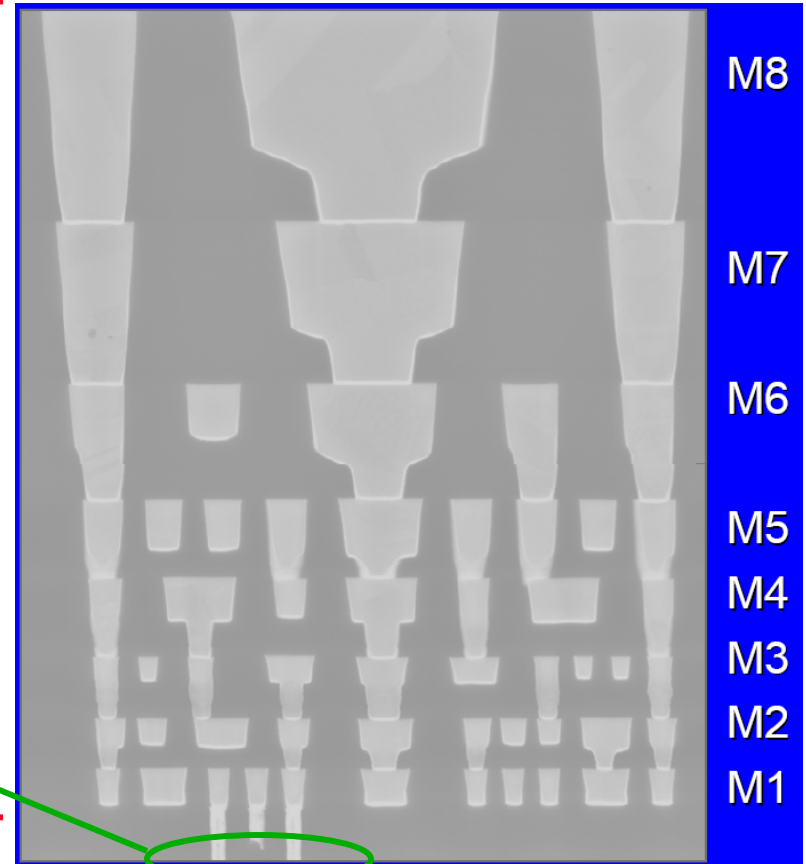
# Introduction

---

- Readings
- Today's topics
  - Wires become more important with scaling
    - Smaller features mean faster devices but not faster wires.
  - Different kinds of wires have different scaled performance
    - Wires that scale in length
    - Wires that are fixed-length
  - How to deal with and estimate wire performance

# A modern technology is mostly wires

- Cross-section, Intel's 65nm tech
  - 8 metal layers
  - Low- $\kappa_r$  dielectrics ( $\kappa_r = 2.7$ )
  - Wires are 2x taller than wide
- Wires are here
- Transistors are here

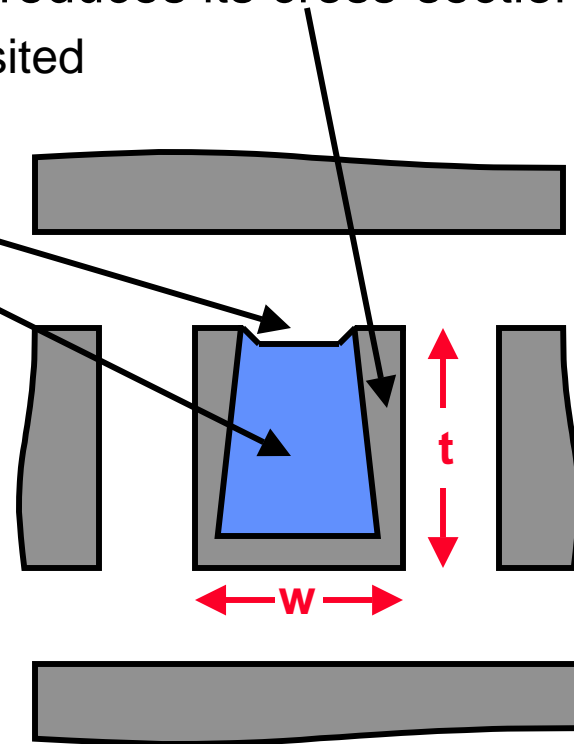


P. Bai, *et al.*, "A 65nm logic technology featuring 35nm gate lengths, enhanced channel strain, 8 Cu interconnect...", *IEDM* 2004.

# Resistance, revisited...

Resistance is resistivity/area, but...

- Copper needs a diffusion barrier that reduces its cross-section
  - Also, barrier may not be evenly deposited
- Copper can be overpolished
  - Can cause dishing (less thick)
- Electrons can scatter off the edges
  - Happens more for thinner wires
  - Increases the base resistivity



$$R_{len} = \alpha_{scatter} \frac{\rho}{(w - 2\delta) \cdot (t - \delta - dish)}$$
$$\rho_{copper} = 2.2m\Omega \cdot cm$$

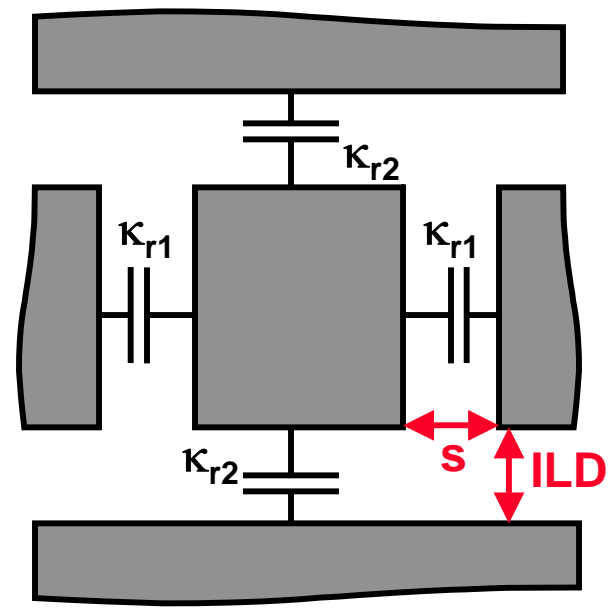
P. Kapur, "Technology and reliability constrained future copper interconnects: Resistance modeling," *IEEE Trans. Electron Devices*, April 2002.

# Capacitance, revisited...

Model capacitance by four plates, each  $\kappa(A/d)$

- Plus a near-constant fringe term 0.1fF/um (fringe scales slowly)
- Relative dielectrics differ, with low- $\kappa$  within a layer
  - SiOF (3.5) or SiOC (2.5)
  - Reduces (dominant) sidewall cap
  - Wires are taller than they are wide
- No low- $\kappa$  between wire layers
  - For material strength

$$C_{len} = \epsilon_0 \left( 2M\kappa_{r1} \frac{t}{s} + 2\kappa_{r2} \frac{w}{ILD} \right) + fringe$$
$$M = MillerFactor$$
$$fringe \approx 100fF/\mu m$$



A "sandwich" model

R. Ho, "The Future of Wires," Proc. IEEE, April 2001.

# Capacitance

---

- Miller factor is important: sidewall cap is 75% of the total today
  - The fringe has a sidewall component, too
- If the neighboring wires are moving with you simultaneously
  - $C_c=0$ , and total capacitance is only 25% of the normal total
  - Worst case for minimum path (race) timing checks
- If the neighboring wires are moving against you simultaneously
  - $C_c$  is doubled, and total cap grows to 175% of the normal total
  - (Really) worst case for maximum path timing checks
- This would be a 7X variation in capacitance (and delay)
  - Often use a “realistic worst-case” of 1.5X  $C_c$ , for 140% total cap

# Is inductive delay important?

- Should we model wires as full transmission lines? (no)
  - Unless we intentionally make inductance important: very wide wires
  - Or unless we're designing the clock grid (picoseconds count!)
- Transmission-line effects can be ignored if the wire is...
  - Short: wire transitions slower than two transmission-line roundtrips

Not exact, but close

$$\begin{aligned} 2.2R_{gate}C_{wire} &> 4\sqrt{L_{wire}C_{wire}} \\ R_{gate} &> 2Z_0 \end{aligned}$$

Driver resistance swamps out wire inductance

- Long: wire's transmission-line attenuation constant exceeds unity

$$\begin{aligned} R_{wire}/(2Z_0) &> 1 \\ R_{wire} &> 2Z_0 \end{aligned}$$

Wire resistance swamps out wire inductance

A. Deutsch *et al.*, "The importance of inductance and inductive coupling for on-chip wiring," 6<sup>th</sup> Topical Meeting EPEP, October 1997.

## Is inductive delay important? (con't)

---

- $Z_0$  is roughly in the ballpark of  $30\Omega$ 
  - Assume no dielectric loss and reasonably gridded power supplies
  - $C_{wire} \sim 0.3\text{pF/mm}$  (and stable under scaling)
  - $L_{wire} \sim 0.3\text{nH/mm}$  (and stable under scaling)
- A chicken and egg problem in calculating  $L_{wire}$ :
  - You don't know  $L_{wire}$  unless you know the current loop paths
  - But you don't know the current loop until you know the impedances!
- Use a construct called “partial inductance”
  - Divide up loop inductance into parts on each (possible) segment
  - (For each segment, assume return path is at some fixed reference)
  - Let SPICE figure out currents on the path(s) of lowest impedance

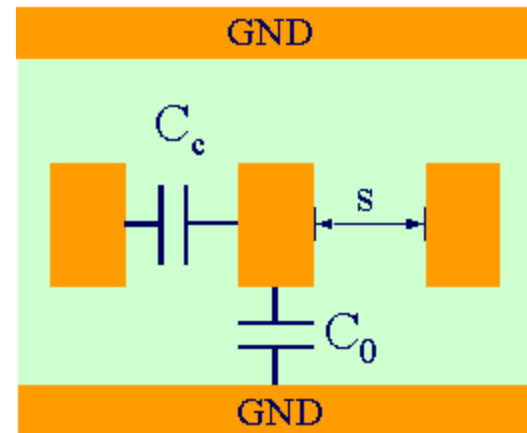
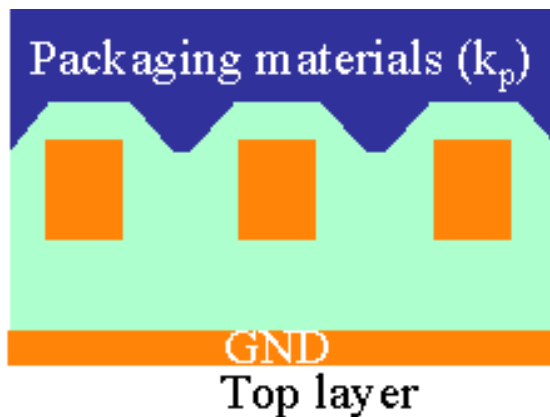
$$Z_0 \approx \sqrt{\frac{L_{wire}}{C_{wire}}}$$

A. Ruehli, “Inductance calculations in a complex integrated circuit environment,” *IBM J. Res. Devel.*, No. 5, September 1972.

M. Beattie, “On-chip induction modeling: basics and advanced models,” *IEEE Trans VLSI*, Vol. 10, No. 6, December 2002.

# Beyond Hand Estimates

- Simple models of R and C take you only so far
  - Beyond that, need 2D or 3D field solvers
- We will use the Berkeley wire calculator (simple 2D field solver)
  - <http://www-device.eecs.berkeley.edu/~ptm/interconnect.html>



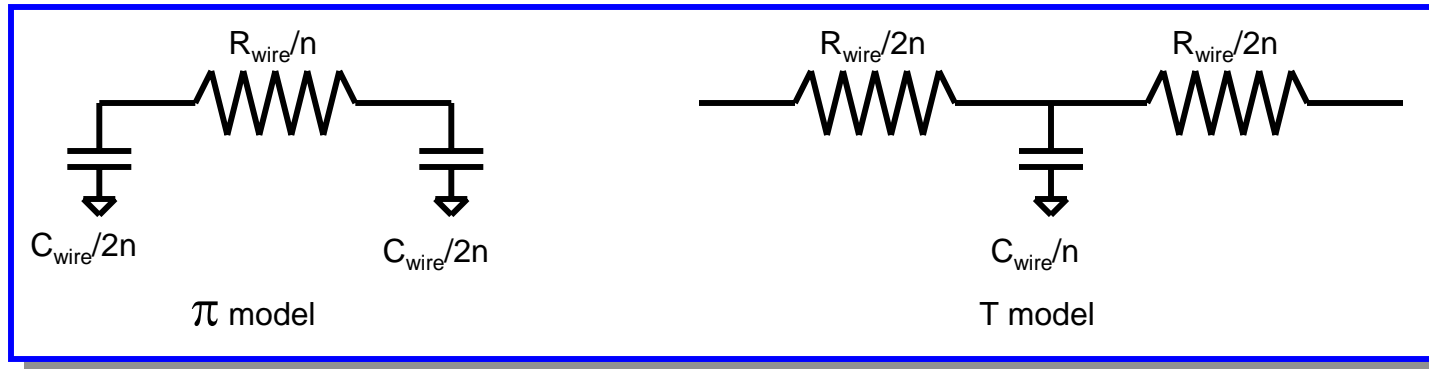
# Wire extraction

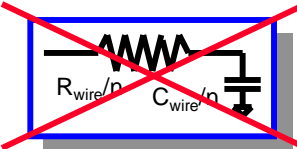
---

- In design, we model wires with Rs and Cs based on routing
  - Or based on guesses... when we remember to guess
- After layout, we typically extract the “real” Rs and Cs
  - Back-annotate the extracted Rs and Cs into our netlists for SPICE
  - Extraction pattern-matches layout with pre-characterized templates
  - This process is typically good to only about 5-10% of a field-solver
  - ...Why accurate wire simulators (AWE) are not a low-hanging fruit
- Extraction process generates huge amounts of netlist data
  - Can easily swamp out the poor SPICE simulation engine
  - 100s of GB of data for modern CPUs – how do you verify it?
  - This is RC data only; extracted L data is 10X more data!

# Wire delays

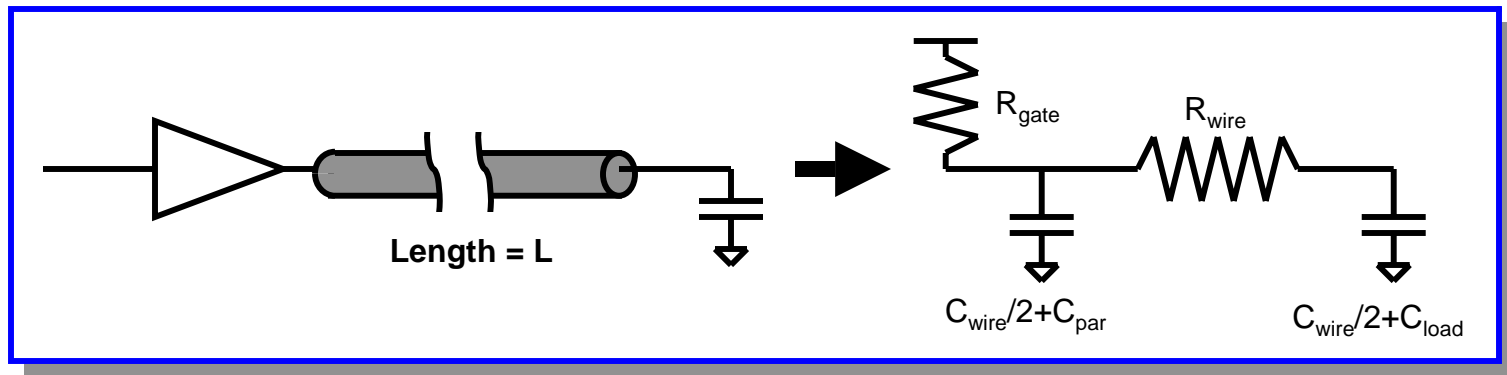
- How do you model a wire with component R and C elements?
  - Break it into  $n$  sections, each either a “ $\pi$ ” or “T” model
  - Elmore delay is  $RC/2$  regardless of the number of sections



- How many sections? Depends on wire resistance...
  - One section per  $2000\lambda$  is probably overkill (channel length =  $2\lambda$ )
- Using an L-model is not very good 
  - Its Elmore delay isn't  $RC/2$ ... need 10 sections to get to within 5%

## Wire performance: latency

- Delay of gate driving a long wire governed by RC time constants



- Elmore delay:  $D = R_{gate}(C_{wire} + C_{par} + C_{load}) + R_{wire}(0.5C_{wire} + C_{load})$
  - For long wires:  $D = k \cdot FO_4 + 0.5R_{wire}C_{wire}$
  - Quadratic in total wire length
- For long wires, this delay quickly becomes untenable
    - In a 65nm process, wire delay looks like 2-3 FO4/mm<sup>2</sup>

## Aside: why do we use Elmore delay?

---

- Consider the impulse response  $h(t)$  of an RC circuit (like a wire)
  - $h(t) > 0 \quad \forall t$       *i.e.*, step response monotonically increases
  - $\int_0^\infty h(t) dt = 1$       *i.e.*, step response reaches 1 in the limit
- In other words,  $h(t)$  looks like a probability distribution function
  - The step response is the associated cumulative density function
  - Notice that delay is the time for which the step response is 50%
  - Therefore, delay is the median of the impulse response
- Elmore noticed that for most RC circuits
  - The median of the impulse response may not be easy to calculate
  - But the mean of the impulse response (RC formula) is pretty close!
  - Good to understand when it isn't (*e.g.*, large  $R_{wire}$  shielding  $C_{wire}$ )



W. Elmore, "The transient response of damped linear networks with particular regard to wide-band amplifiers," *J. Appl. Phys.*, Vol.19, No.1, Jan. 1948.

## Wire performance: noise

---

- Wires are skinny and tall and have lots of sidewall capacitance
  - Aspect ratios at 2.2 now and are projected to scale up to 3-3.5
  - We will have to live with some coupled noise
- Traditional estimates use a simple capacitive divider
  - $V_{\text{noise}}$  is  $\frac{C_{\text{side}}}{C_{\text{side}} + C_{\text{top+bot}}}$
  - But this is pessimistic, because the “victim” is usually driven, too
- In reality, you must account for both victim and attacker drivers

–  $V_{\text{noise}}$  is  $\frac{C_{\text{side}}}{C_{\text{side}} + C_{\text{top+bot}}} \cdot \frac{1}{1 + \frac{\tau_a}{\tau_v}}$

about 75%  From 2 to 4 

# Wire performance: analyzing noise

- Can we quickly sketch the waveform of coupled noise?
- Use the “Miller” and “anti-Miller” effects
  - Coupling cap doubles if wires swing in opposite directions
  - Coupling cap is 0 if wires swing the same direction
- Superposition: break victim response into even and odd modes
  - Even mode uses anti-Miller effect; odd mode uses Miller effect
  - Can now analyze victim response without considering attacker!

A bit pessimistic due to linear versus saturated resistances →

(Simplistic) First-order model

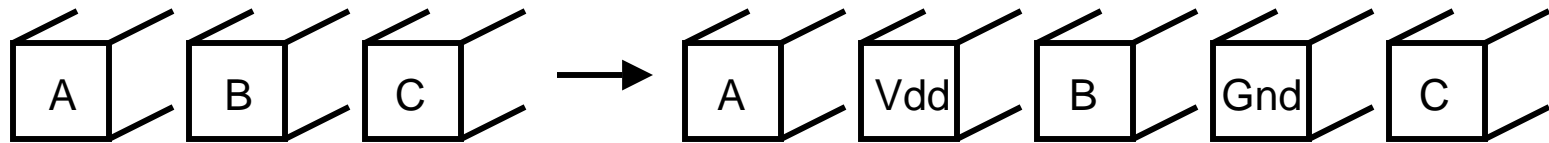
$$V_v(t) = 0.5 \left( e^{\frac{-t}{RC_m}} - e^{\frac{-t}{RC_a}} \right)$$

$$C_m = \text{Cap with } 2x C_c \text{ (Miller effect)}$$

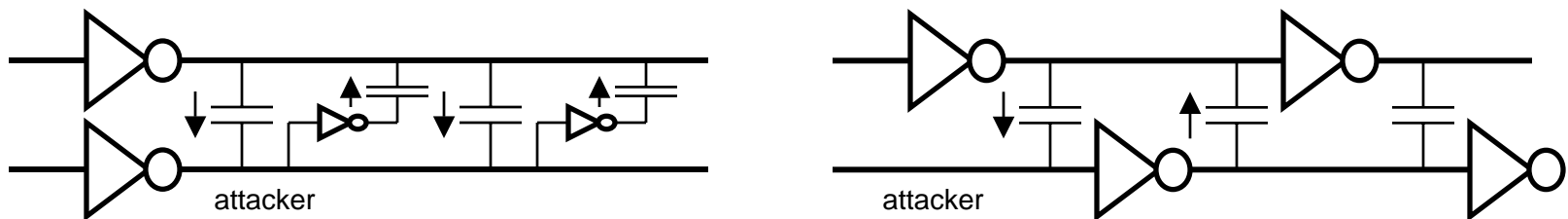
$$C_a = \text{Cap with no } C_c \text{ (anti-Miller)}$$

# Getting Rid of Crosstalk

- Shields

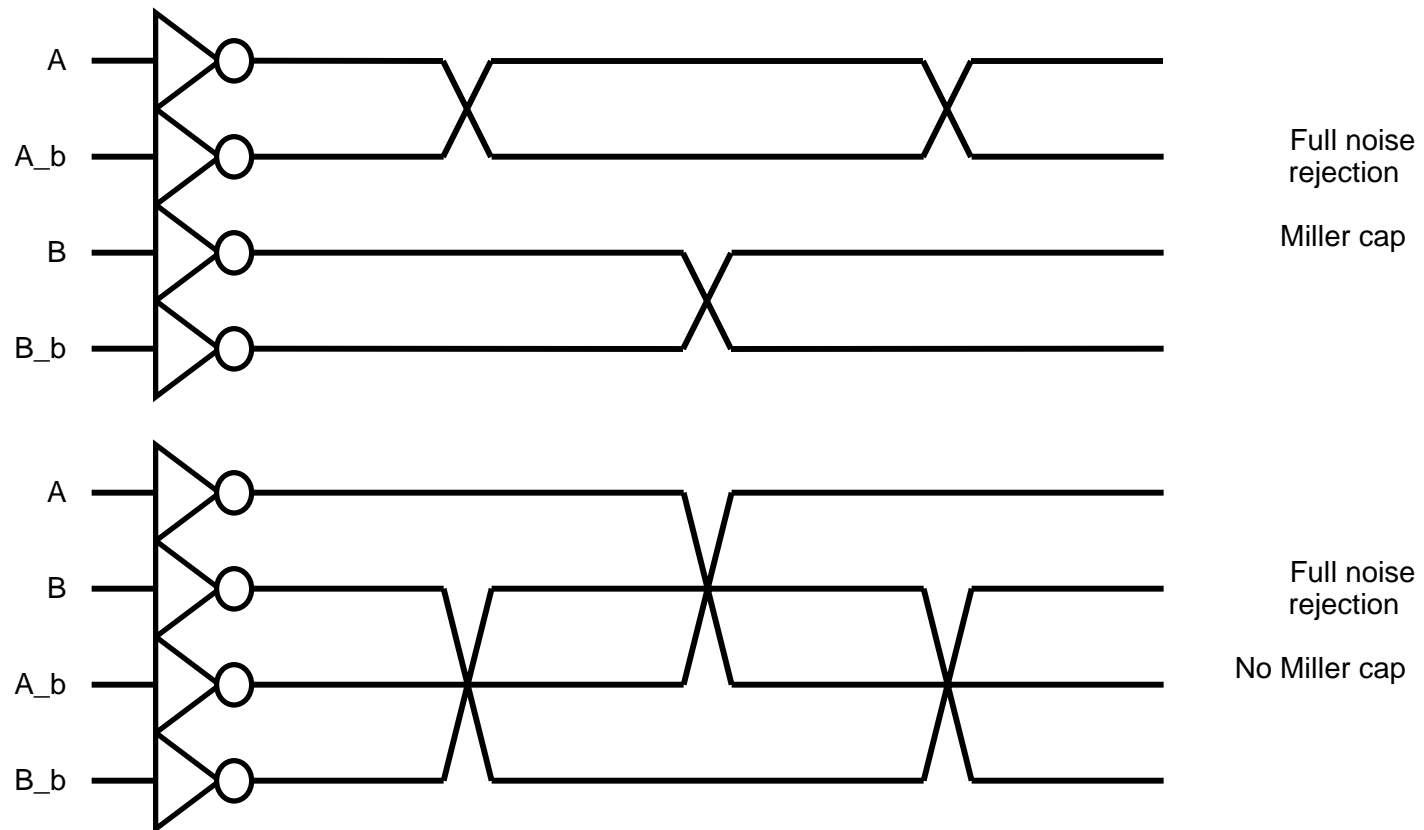


- Active noise cancellation and staggered drivers



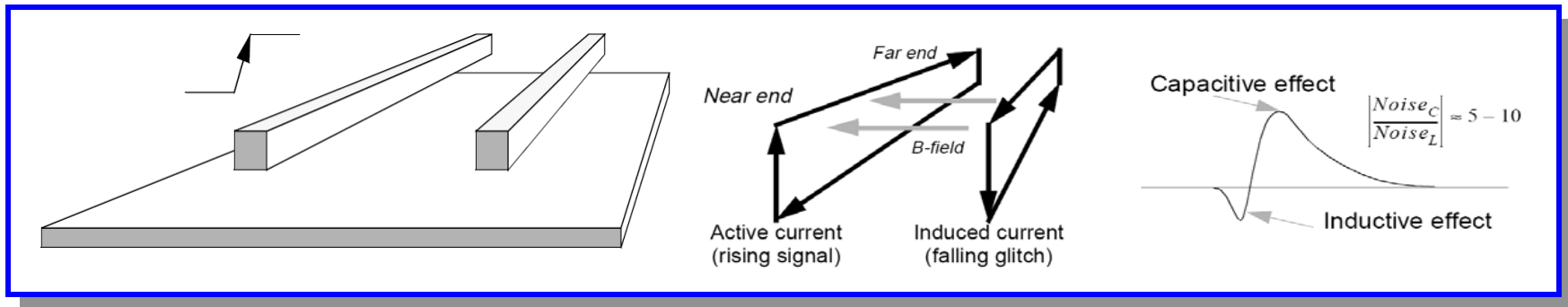
# Getting Rid of Crosstalk, con't

- Twisting and differential wires are very effective
  - Cuts down on inductive coupling as well, but costs 2x wires

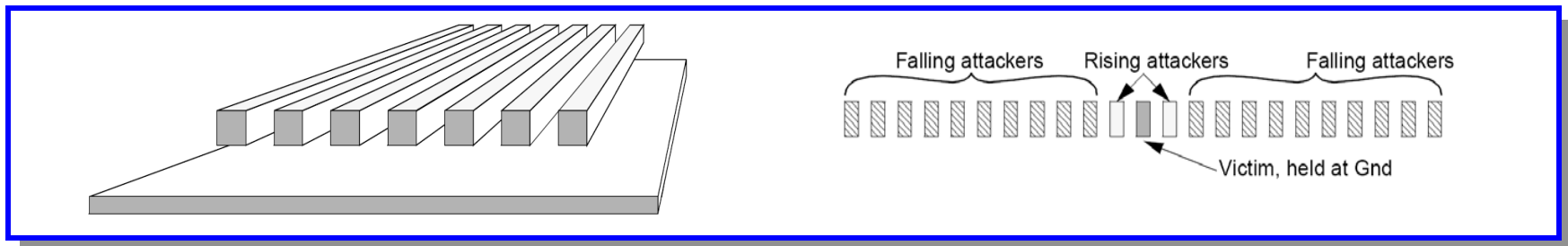


# Wire performance: inductive noise

- Inductive noise can bite if you're not careful
  - A single attacker isn't too bad...



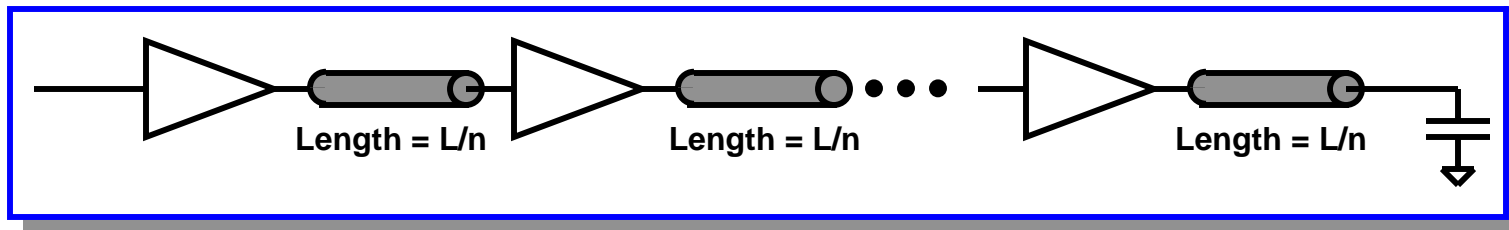
- But many attackers can add up...



- So: interleave power/gnd lines (every 4-8 wires) to reduce loops

# Wire performance: repeaters

- So long wires are slow – use repeaters!
  - Gain stages that break up the wire and “refresh” the signal
  - Inverters are the simplest gain stage
  - Buffers also popular (no inversion, less crowbar current, but slower)



- Delay of a repeated line is linear in total length, not quadratic
  - Delay is the geometric mean of the wire delay and the gate delay
  - $D = \text{constant} * \text{sqrt}(\text{FO4} * \text{RwCw})$
  - You will work out the details in a homework

See H. Bakoglu (*Circuits, Interconnections, and Packaging for VLSI*, Addison-Wesley, 1990) for a sample derivation, but that does not include parasitics.

# Wire performance: Using repeaters

---

- Issue: Repeaters consume real estate and block wiring with vias
  - Cluster repeaters (“gas stations”) and route wires to them
  - Exploit the shallowness of the repeater optimization curves
- Issue: As I scale a design, I’ll need more repeaters for each wire
  - Use (close to) the optimal number of repeaters for each critical wire
  - As opposed to the bare minimum to just meet timing
  - Pay a slight power cost, but this enables your proliferations to scale
- How many repeaters are we talking about?
  - 180nm Itanium: 20K (est) [1]; 130nm Ultrasparc: 16K [2]
  - Some extrapolations (“70% of the total chip”) won’t happen [3]
  - *e.g.*, due to multi-core architectures or clever circuit designs

[1] S. Rusu *et al.*, “The first IA-64 microprocessor,” *IEEE JSSC*, Vol. 35, No.11, Nov. 2000.

[2] G. Konstantinidis *et al.*, “Implementation of a third-generation 1.1-GHz 64b microprocessor,” *IEEE JSSC*, Vol. 37, No.11, Nov. 2002.

[3] P. Saxena *et al.*, “The scaling challenge: Can correct-by-construction help?” *Proc. IEEE ISPD*, 2003.

# Improving wire performance

---

- Increase wire pitch (if you can afford the density hit)
  - If width increases, resistance decreases accordingly
    - A bit faster than linear due to barrier metal's larger effect on thin wires
  - Capacitance grows very slowly with width and falls with spacing
    - Side-to-side capacitance dominates
  - Total RC delay falls, and repeated delay falls by  $\sqrt{RC}$
- Move the signal to a higher metal layer
  - Effectively makes the wire wider and usually thicker, too
- Reduce effective capacitance
  - Interleave buses that never switch simultaneously (avoid Miller)
  - Using codes [1] is often not worth it (unless you're coding already)

[1] K. Kim *et al.*, "Coupling-driven signal encoding scheme for low-power interface design," *IEEE ICCAD* 2000.

# Wire Limitations

---

- There are other good reasons for wider wires
- $iR$  drops in power supply line
  - Need to worry about transient currents too
- Electromigration
  - Electron “wind” can move metal atoms and break wires over time
- $iR$  heating
  - Oxide is a good thermal insulator
  - Wires are resistive and can heat up

# Wire performance: reliability

---

- Wires degrade due to electromigration and self-heating

- Electromigration (EM) creates wire opens
  - Caused by unidirectional current flow
  - Electrons smack into lattice, displacing atoms
  - Wires with bidirectional current is “self-healing”
  - Check those short segments to Vdd or Gnd
  - Copper’s MTTF is 5x better than Aluminum’s

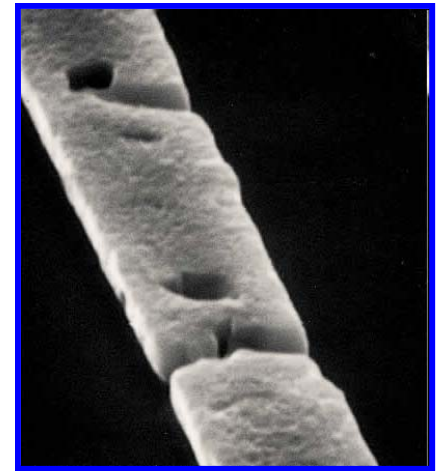


Photo: Sanyo Electronics

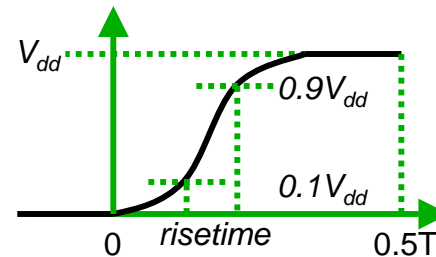
- Calculate max DC current, which depends on total capacitance
  - Rule is “max current per wire cross-section” (e.g., 1mA/ $\mu\text{m}^2$ )
  - Also have “max current per via” to enforce arrays of vias

J. Black, “Electromigration—A brief survey and some recent results,” *IEEE Trans. Electron Devices*, ED-16(4), 1969.

## Wire performance: reliability (con't)

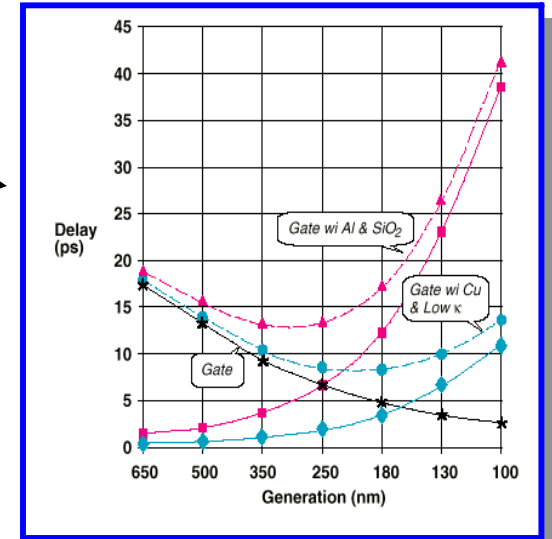
- Self-heating (joule heating) can be a problem
  - Would like wire to not be much hotter than the rest of the chip
  - Takes place in wires with bidirectional current (most long wires)
- Rules are similar to EM, with “max current per wire cross-sec”
  - But this time it's rms current  $I_{rms} \approx 1.25C_{wire}V_{dd}\sqrt{\frac{freq \cdot AF}{risetime}}$

$$\begin{aligned}
 I_{rms} &= \sqrt{\frac{1}{0.5T} \int_0^{0.5T} i^2(t) dt} \\
 &= \sqrt{\frac{1}{0.5T} \int_0^{0.5T} (C \frac{dv}{dt})^2 dt} \\
 &= \sqrt{\frac{1}{0.5T} C^2 \frac{dv}{dt} \int_{t=0}^{t=0.5T} dv} \\
 &\approx \sqrt{\frac{1}{0.5T} C^2 \left(\frac{0.8V_{dd}}{risetime}\right) (V_{dd})} \\
 &= CV_{dd} \sqrt{\frac{0.8}{0.5T \cdot risetime}} \approx 1.25CV_{dd} \sqrt{\frac{freq \cdot AF}{risetime}}
 \end{aligned}$$



# Wire Scaling

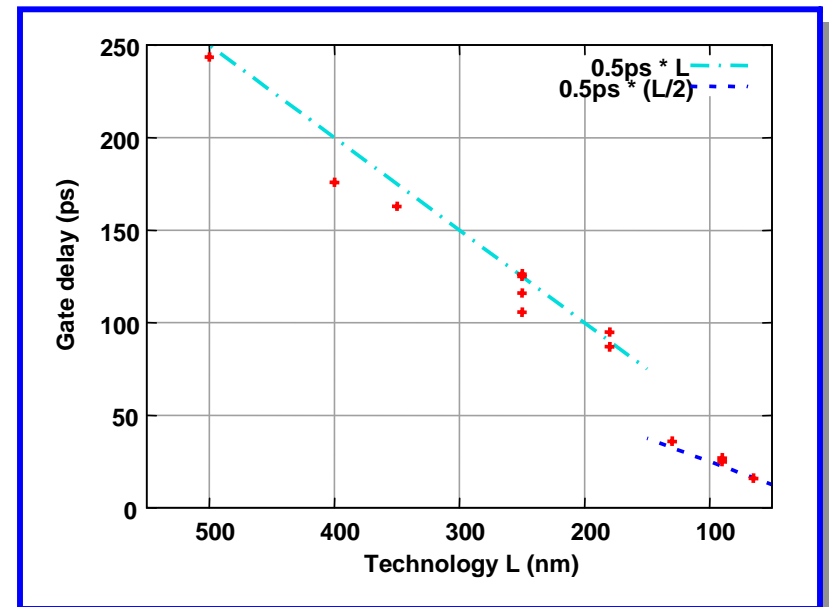
- What happens to wire delay under scaling?
  - Many claim(ed) that wires would blow up
  - But it depends on how you scale the wires



- In a technology shrink there are **two** types of wires
  - Wires of constant logical span (reaching the same number of gates)
  - Wires of growing logical span (covering the same fraction of die)
- The delay scaling of these wires is different

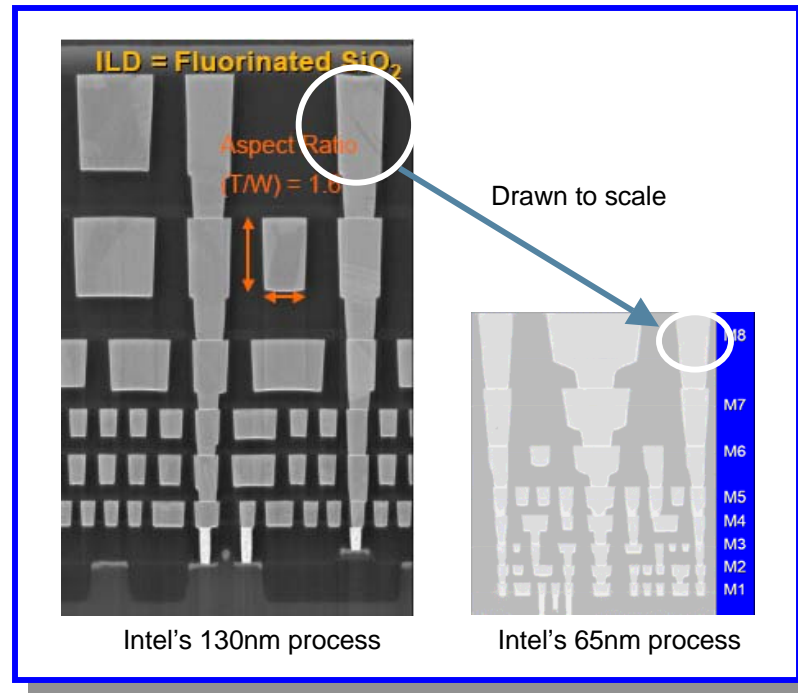
# First, look at gate delay scaling

- Wire scaling matters only if slower (different) than gate scaling
- Use FO4 as our gate delay metric
  - Delay (in ps) =  $0.5 * L_{phys}$  (in nm)
- Poly undercuts change  $L_{phys}$ 
  - $L_{phys} = L_{drawn}$  up until 180nm
  - $L_{phys} = 0.5L_{drawn}$  after 130nm
- Will this hold in the future?
  - Transistors may look very different: FinFETs, carbon nanotube
  - Continued scaling of FO4 may be a self-fulfilling prophecy
  - In any case, aggressive FO4 trends highlight wire problems



# Wire scaling basics

- Shrinking wire dimensions changes characteristics
  - $R_{wire}$  depends on cross-sectional areas: grows quickly
  - $C_{wire}$  and  $L_{wire}$  depend on the ratio of dimensions: stays constant

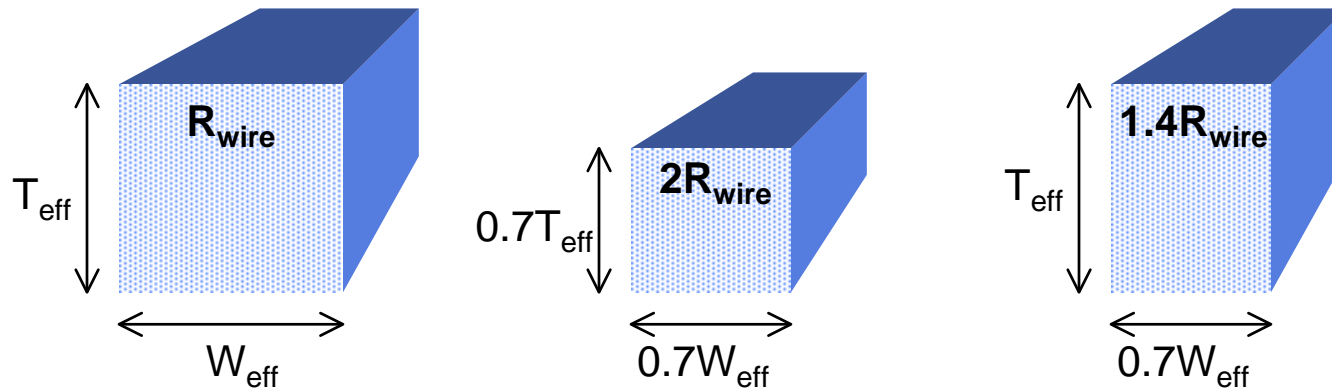


S. Thompson, "An enhanced 130nm generation logic technology featuring 60nm transistors...", *IEDM* 2001.

P. Bai, *et al.*, "A 65nm logic technology featuring 35nm gate lengths, enhanced channel strain, 8 Cu interconnect...", *IEDM* 2004.

# Wire R and C Scaling

- Resistance grows as cross-sectional area decreases
  - Strict scaling makes resistance increase quickly
  - Wires have been growing taller to compensate



- Capacitance grows slowly
  - Depends on ratio of dimensions
  - As wires stay tall (to help resistance) sideways cap grows
  - Dielectric constants get smaller (slowly) with technologies

# Scaling of R and C

- **Actual numbers are unimportant, but note the trends**
  - There is a difference between accuracy and precision!
  - Make assumptions to get optimistic and conservative trends
  - It's really hard to predict the future; easier to predict bounds

Unrepeated wires	Technology node (nm)					
	65	45	32	22	15	10
Resistance, $\Omega$ /mm, optimistic to conservative	135	240 350	470 725	1000 1560	2150 3420	4000 6950
Capacitance, pF/mm, optimistic to conservative	0.36	0.34 0.35	0.31 0.33	0.285 0.325	0.260 0.325	0.240 0.315
0.5RC, ps/mm/mm, optimistic to conservative	24	41 61	73 120	140 255	280 560	480 1090

20-30x

# Scaling of R and C in FO4

- What really matters for scaling is how wires compare to gates
  - Normalize the delay to FO4 delays
  - Scaling story is quite a bit worse
  - Keep in mind this is also quadratic with wire length, too...

Unrepeated wires	Technology node (nm)					
	65	45	32	22	15	10
Resistance, $\Omega$ /mm, optimistic to conservative	135	240 350	470 725	1000 1560	2150 3420	4000 6950
Capacitance, pF/mm, optimistic to conservative	0.36	0.34 0.35	0.31 0.33	0.285 0.325	0.260 0.325	0.240 0.315
0.5RC, <b>FO4</b> /mm/mm, optimistic to conservative	1.6	3.6 5.4	9.1 15	25 46	75 150	175 400

100-200x

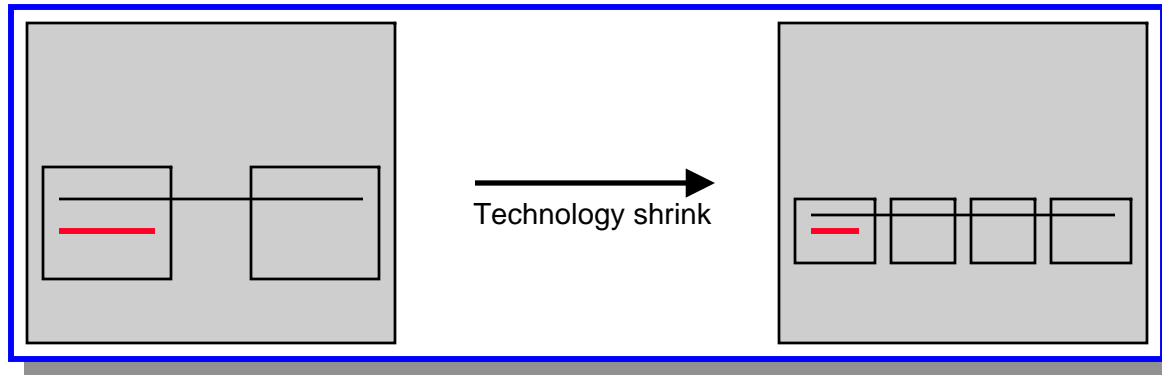
# Wire Scaling

---

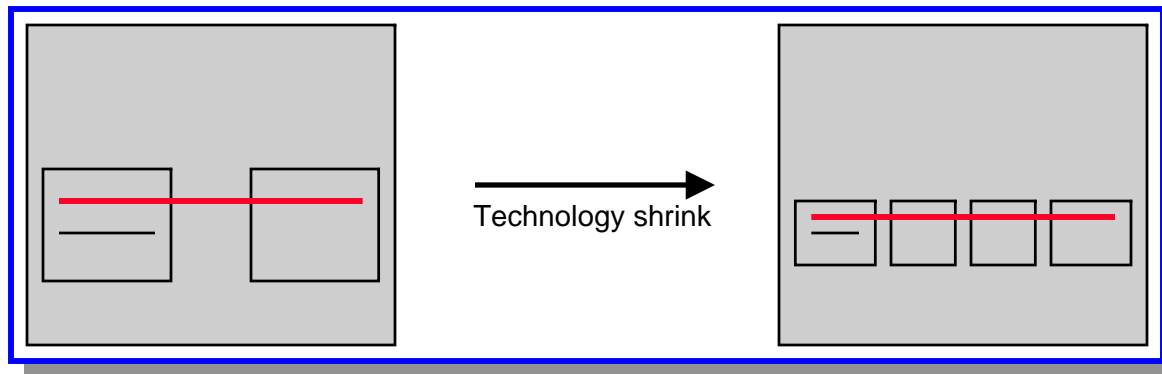
- Key is to remember there are two kinds of wires
  - Wires with a constant logical reach
    - All the wires when you shrink a chip to a new technology
    - All module level wires
    - $L$  scales, so the delay of the wire decreases as  $T/W$  grows
    - Ratio of wire to gate delay grows, but slowly
  - Wires of constant length
    - Global wires spanning a chip
    - Delay increases, while the gate delays decrease
- Duality should not be surprising
  - Communication has some cost
  - Starting to have communication delays on-chip
  - Today's chips are like yesterday's boards

# Scaling: two kinds of wires

- Local wires, inside modules or blocks, get shorter under scaling



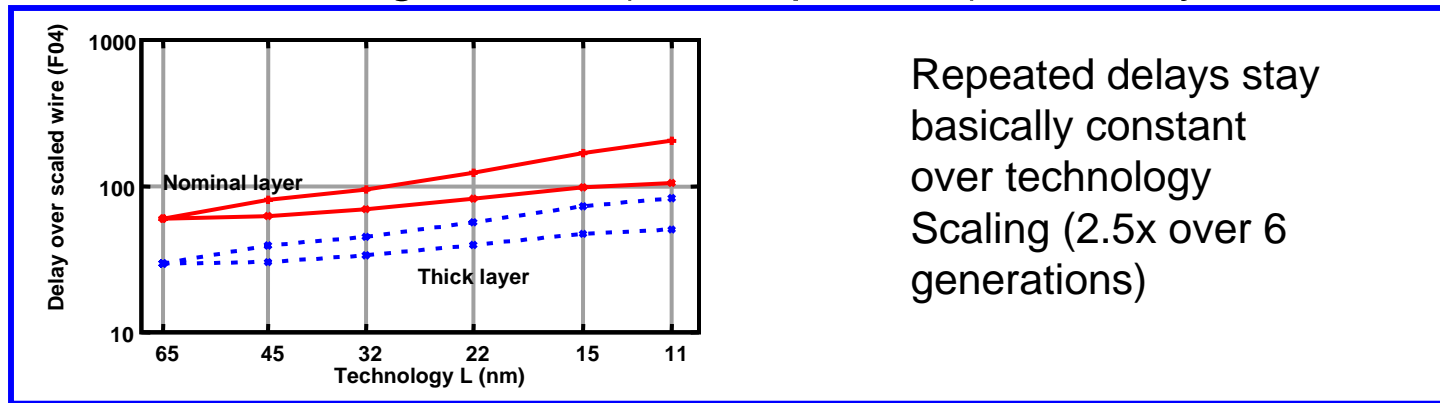
- Global wires, connecting modules together, do not get shorter



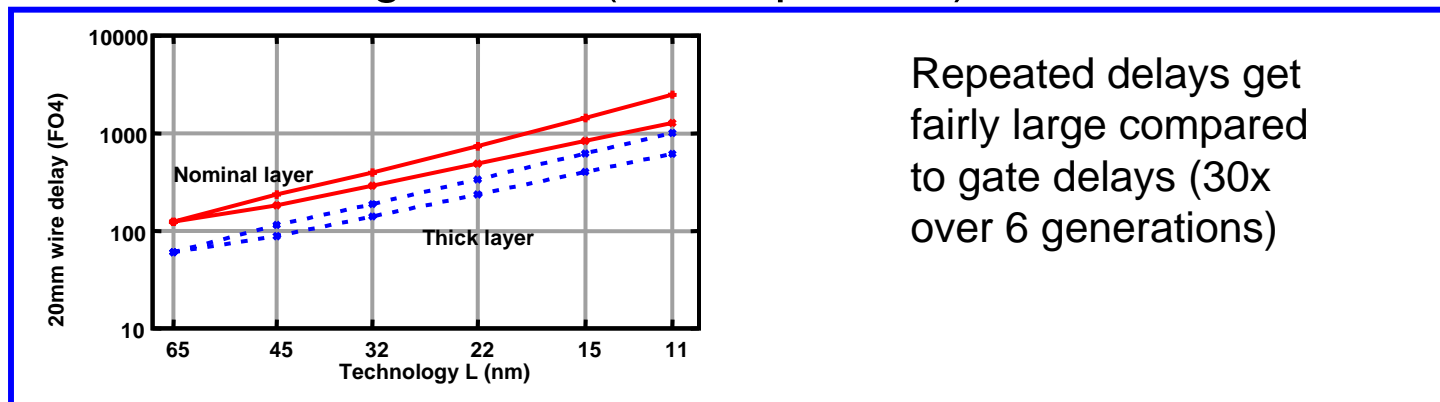
D. Sylvester *et al.*, "Getting to the bottom of deep submicron design," *IEEE/ACM ICCAD* 1998.

# Scaling: two kinds of trends

- Local, scaled length wires (with repeaters) are okay

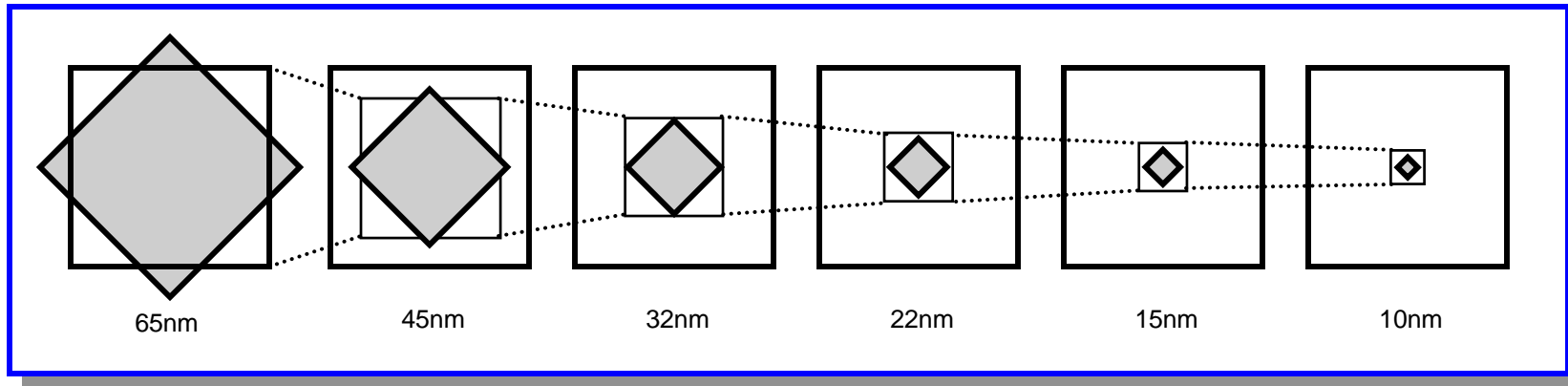


- Global, fixed length wires (with repeaters) are not



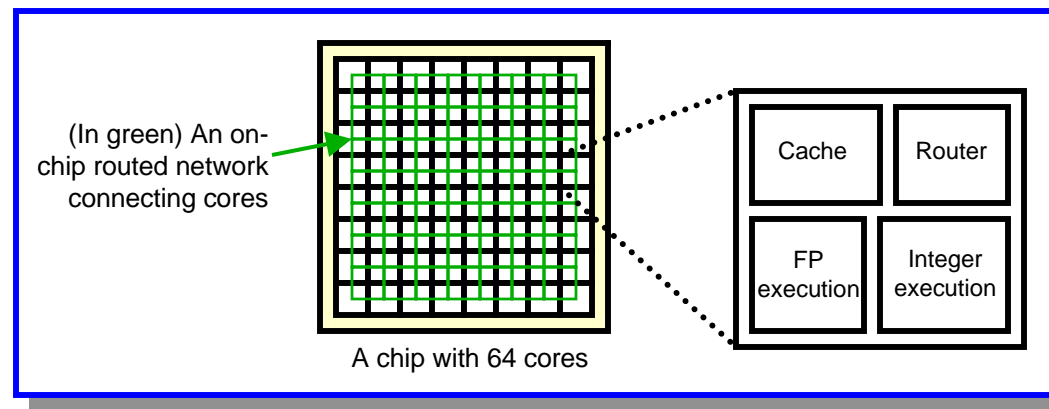
# Scaling: another perspective

- How far can a wire “reach” in a clock cycle?
  - Assume a 35FO4 clock cycle and a small 85mm<sup>2</sup> die
- Each technology scaling step reduces everything
  - Except die size: assume die complexity grows
  - Wire reach falls a little faster than the original block size



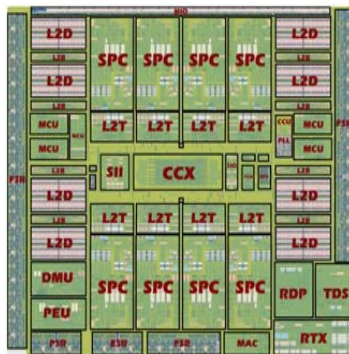
# Scaling implications for architecture

- Look towards architectures that can handle the duality of wires?
  - Slow global communication does not throttle performance
  - Performance is set by fast local (scaled-length) wires
- Answer: modular architectures using multiple tiled cores
  - Cores are built with local wires that are fast and that scale well
  - Global core-to-core communication is explicitly slow
  - Can even make global communication visible to the programmer

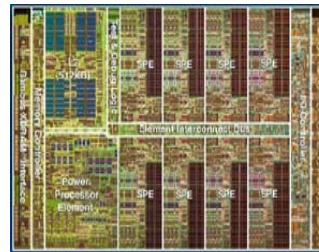


# Scaling: modular machines

- Not a new idea: lots of early work in mid- to late-1990s
  - Academic: MIT RAW, Stanford Smart Memories, UT GRID/TRIPS
  - Industry: Dual-core Sparc, Pentium, and Opteron processors
- Now everybody is getting in the game, and in spades



Sun's Niagara2: 8 cores, 64 threads



IBM's Cell: 8 symmetric cores, 1 PPC



Intel's 80-core terascale prototype

G. Grohoski, "Niagara2: A highly-threaded server-on-a-chip," *IEEE Hot Chips*, 2006.

J. Warnock, *et al.*, "Circuit design techniques for a first-generation Cell broadband engine processor," *IEEE JSSC* Vol.41, No.8, August 2006.

Intel Developer Forum, 2006.

# Modularity: good for other reasons

---

- Moderate energy using selective block power-down
  - The popular question in 1997: “What do you do with x billion xstrs?”
  - The practical answer in 2007: “Don’t turn them all on at once!”
  - Deep sleep for blocks you aren’t currently using
- Manage design complexity with divide-and-conquer
  - CPU teams regularly exceed hundreds of designers
  - Yet Intel’s Pentium4 design team had only a handful of “wizards” [1]
- Modularity can lead to reconfigurability and IP reuse
  - Various estimates of ASIC design costs in 65nm: \$100M
  - So build “general-purpose ASICs” with reconfigurable cores/logic
  - Exploit heterogeneity in the cores for repurposing logic

[1] B. Colwell, “CPU performance: complexity is a spent force,” Industry talk, 2005.