Agenda

- How to read an *arch* paper
- A brief history of architecture
- Technology building blocks
- Technology scaling
- Fundamentals of performance

Grading

- Read and *think about* each paper.
- Submit a summary.
- It is essential that you do this. Your grade depends substantially upon it.
- It is also essential that you learn to do this well.
    - Extracting content from papers is one of the most important skills in grad school.

Paper summaries

- Goal 1: Extract the good ideas from the paper.
  - This means discarding the junk.
  - Identifying the good parts.

Goal 2: Understand how it fits into its context

  - How is it similar/different/an extension of...?

What's the paper's goal?

- Does it solve a problem?
- Demonstrate an opportunity?
- Does it provide information?

Taylor's IMD Taxonomy for Architecture Literature

- INSIGHT - here's some data that shows something that we didn't know before about programs/architecture/technology
- e.g., Moore's Law

MECHANISM - here's a cool algorithm you can embed in silicon

- e.g., 2-bit predictor

DINOSAUR - Here's something we built

- e.g., Raw processor, R10000, Wattch Simulator, etc.

How do the authors substantiate their claims?

- Experiments?
  - Real systems?
  - Simulation?
- Prose arguments?
- Examples from "the real world"

How does the paper relate to others?

- Refute?
- Confirm?
- Extend?
- Synthesize?
- Re-examine?
  - In light of new tech./new app./new idea

What conclusions do they draw?

- Small conclusions
  - Did their idea work?
  - How well?
  - Do you believe them?
- Big conclusions
  - How do they think it should shape the future?
  - Do you believe them?

How well is the paper crafted?

- Does it tell a story?
- Is it interesting?
- Are the figures easy to understand?
- Do they properly highlight the important parts?
- Could you summarize the paper after looking at it for 5 minutes? (not for this class, you can't ;-)

How would you improve the paper?

- Technically
  - Different approach (maybe you should write a paper?)
  - Methodology
  - Experiments
- Presentation
  - Organization
  - Additional background
  - Be concrete -- “make it more clear” is not useful.

What questions does it raise?

- Issues with their approach?
- Directions for new work?
- Broader questions about architecture?
- What didn't you understand?

Paper summaries

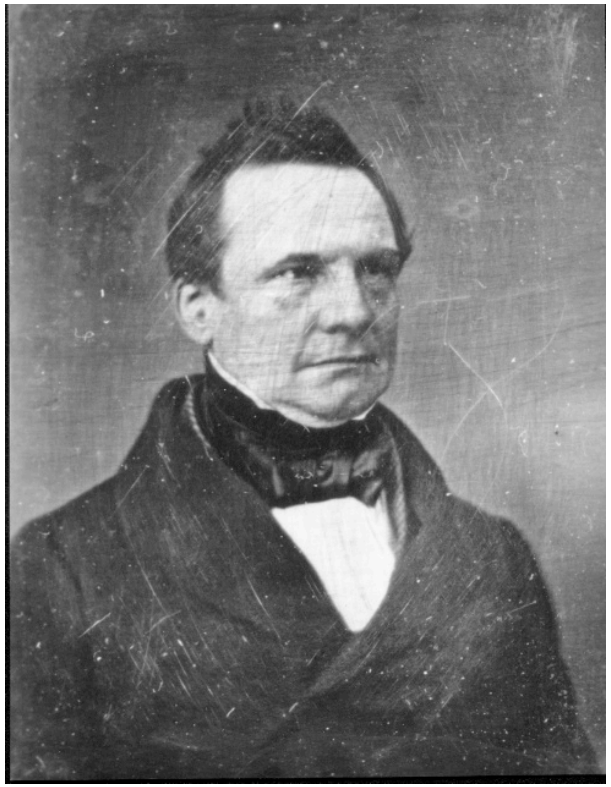
- Submitted via a google form (see website)
- Due 1 hour before class -- no exceptions.
  - You should never miss class for this
- You should bring a printed version of each paper to class!

An Incomplete History of Computation

Abacus: ca. 2700 B.C., Sumeria

Charles Babbage 1791-1871

Lucasian Professor of Mathematics,
Cambridge University, 1827-1839
First computer designer



Ada Lovelace 1815-1852

First computer programmer

Charles Babbage

- *Difference Engine* 1823
- *Analytic Engine* 1833
  - The forerunner of modern digital computer!

Application

- Mathematical Tables – Astronomy
- Nautical Tables – Navy

Background

- Some efforts at mechanical calculators in the past.

Technology

- mechanical - gears, Jacquard's loom, simple calculators

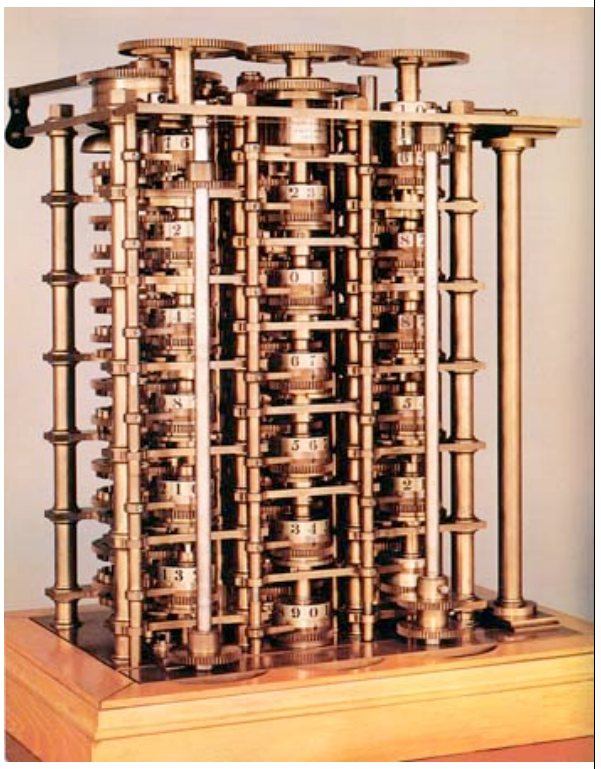
Difference Engine

1823
- Babbage's paper is published

1834
- The paper is read by Scheutz & his son in Sweden

1842
- Babbage gives up the idea of building it;he is on to the Analytic Engine!

1855
- Scheutz displays his machine at the Paris World Fare
- Can compute any 6th degree polynomial
- *Speed:* 33 to 44 32-digit numbers per minute!



Now the machine is at the Smithsonian

Analytic Engine

The first conception of a general purpose computer

1. The *store* in which all variables to be operated upon, as well as all those quantities which have arisen from the results of the operations are placed.
2. The *mill* into which the quantities about to be operated upon are always brought.

An operation in the *mill* required feeding two punched cards and producing a new punched card for the *store.*

An operation to alter the sequence (i.e., a branch) was also provided!

Analytic Engine

1833: Babbage's paper was published
- *conceived during a hiatus in the development of the difference engine*
- *output devices: Printer, Curve Plotter, and Bell*
- *input devices: Punch Cards*

Inspiration: *Jacquard Looms*
- looms were controlled by punched cards
  - The set of cards with fixed punched holes dictated the pattern of weave ⇒ *program*
  - The same set of cards could be used with different colored threads ⇒ *numbers*

1871: Babbage dies
- The machine remains unrealized.
- Technology not up to the task.

Linear Equation Solver

John Atanasoff, Iowa State University

1930's:

- Atanasoff built the Linear Equation Solver.
- It had 300 tubes!

Application:

- Linear and Integral differential equations

Background:

- Vannevar Bush's Differential Analyzer
  --- *an analog computer*

Technology:

- Tubes and Electromechanical relays

Atanasoff decided that the correct mode of computation was by electronic digital means.

Harvard Mark I

- Built in 1944 in IBM Endicott laboratories
  - Howard Aiken – Professor of Physics at Harvard
  - Essentially mechanical but had some electro-magnetically controlled relays and gears
  - Weighed *5 tons* and had *750,000* components
  - A synchronizing clock that beat every *0.015* seconds

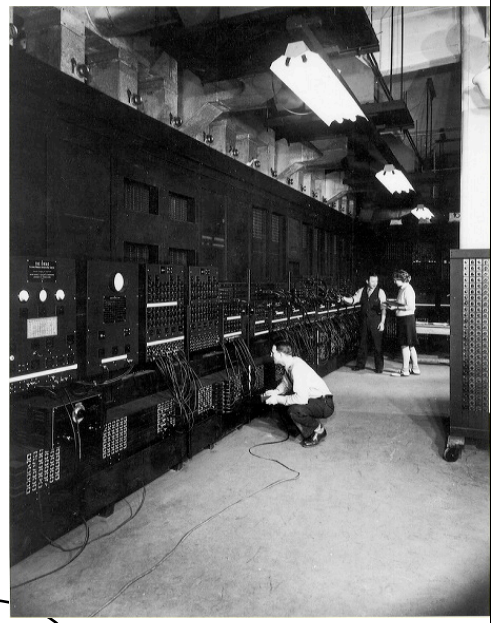
Performance:

0.3 seconds for addition
6 seconds for multiplication
1 minute for a sine calculation

Broke down once a week!

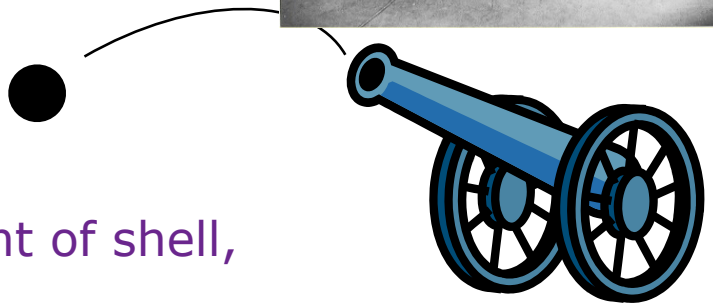
Electronic Numerical Integrator and Computer (ENIAC)

- Inspired by Atanasoff and Berry, Eckert and Mauchly designed and built ENIAC (1943-45) at the University of Pennsylvania
- The first, completely electronic, operational, general-purpose analytical calculator!
  - 30 tons, 72 square meters, 200KW
- Performance
  - Read in 120 cards per minute
  - Addition took 200 μs, Division 6 ms
  - 1000 times faster than Mark I
- Not very reliable!



WW-2 Effort

Application: Ballistic calculations



angle = f (location, tail wind, cross wind,
air density, temperature, weight of shell,
propellant charge, ...)

Electronic Discrete Variable Automatic Computer (EDVAC)

- ENIAC's programming system was external
  - Sequences of instructions were executed independently of the results of the calculation
  - Human intervention required to take instructions "out of order"
- Eckert, Mauchly, John von Neumann and others designed EDVAC (1944) to solve this problem
  - Solution was the *stored program computer*

    ⇒ "*program can be manipulated as data*"
- *First Draft of a report on EDVAC* was published in 1945, but just had von Neumann's signature!
  - In 1973 the court of Minneapolis attributed the honor of *inventing the computer* to John Atanasoff

Stored Program Computer

Program = A sequence of instructions

How to control instruction sequencing?

manual control — calculators

automatic control

- *external (paper tape)* — Harvard Mark I , 1944; Zuse's Z1, WW2
- *internal*
  - *plug board* — ENIAC 1946
  - *read-only memory* — ENIAC 1948
  - *read-write memory* — EDVAC 1947 *(concept)*

– The same storage can be used to store program and data

EDSAC 1950 Maurice Wilkes

first stored program computer

And then there was IBM 701

IBM 701 -- 30 machines were sold in 1953-54

IBM 650 -- a cheaper, drum based machine,
more than 120 were sold in 1954
and there were orders for 750 more!
- eventually sold about 2000 of them

Users stopped building their own machines.

Why was IBM late getting into computer technology?

IBM was making too much money!
Even without computers, IBM revenues were doubling every 4 to 5 years in 40's and 50's.

Dominant Problem: *Reliability*

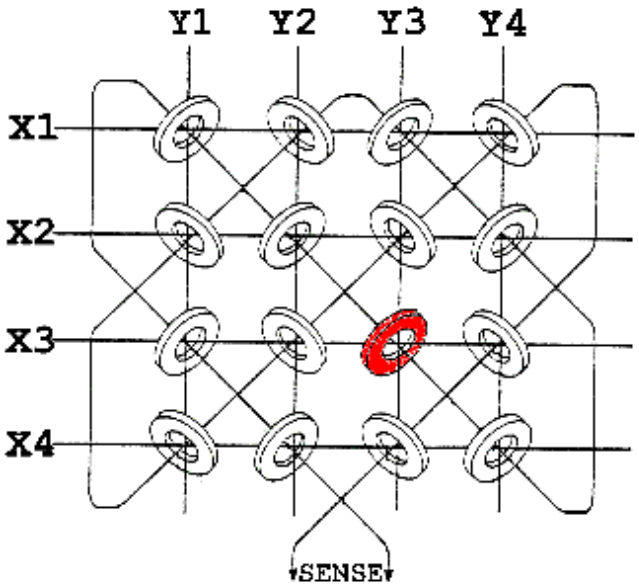
Mean time between failures (MTBF)

MIT's Whirlwind with an MTBF of 20 min. was perhaps the most reliable machine !

Reasons for unreliability:

1. Vacuum Tubes

2. Storage medium
   - acoustic delay lines
   - mercury delay lines
   - Williams tubes
   - Selections



Magnetic Core Memory J. Forrester 1951

-first cheap, reliable memory (~ 1 MHz)
-also called "core" (e.g., "core dump")
-non volatile!
-destructive read cycle
-array of ferrite toroids (or "cores")
-dominant memory technology until 70's (→ ICs)

Computers in mid 50's

- Hardware was expensive
- Stores were small (1000 words)
  ⇒ No resident system-software!
- Memory access time was 10 to 50 times slower than the processor cycle
  ⇒ Instruction execution time was totally dominated by the *memory reference time*.
- The *ability to design complex control circuits* to execute an instruction was the central design concern as opposed to *the speed* of decoding or an ALU operation
- Programmer's view of the machine was inseparable from the actual hardware implementation

Into the 60's…:

Compatibility Problem at IBM

By early 60's, *IBM had 4 incompatible lines of computers!*

701	→	7094
650	→	7074
702	→	7080
1401	→	7010

Each system had its own

- Instruction set
- I/O system and Secondary Storage: magnetic tapes, drums and disks
- assemblers, compilers, libraries,...
- market niche
  business, scientific, real time, ...

⇒ *IBM 360*

IBM 360 : Design Premises

Amdahl, Blaauw and Brooks, 1964

http://www.research.ibm.com/journal/rd/441/amdahl.pdf

- Upward and downward, machine-language compatibility across a family of machines
- General purpose machine organization, general I/O interfaces, storage > 32K
- Easier to use (answers-per-month vs. bits-per-second)
- Machine must be capable of *supervising itself* without manual intervention →OS/360 (simple OS's in IBM 700/7000)
- Built-in *hardware fault checking* and locating aids to reduce down time
- Simple to assemble systems with redundant I/O devices, memories etc. for *fault tolerance*

… **the use of the "ISA" as a compatibility layer**
$5 billion project (1964 dollars)

The Amdahl .. from Amdahl's Law.
The Brooks .. from The Mythical Man-Month.


IBM
SYSTEM 360

IBM 360: *A General-Purpose Register (GPR) Machine*

- Processor State
  - 16 General-Purpose 32-bit Registers
    - *may be used as index and base register*
    - *Register 0 has some special properties*
  - 4 Floating Point 64-bit Registers
  - A Program Status Word (PSW)
    - *PC, Condition codes, Control flags*
- A 32-bit machine with 24-bit addresses
  - No instruction contains a 24-bit address !
- Data Formats
  - 8-bit bytes, 16-bit half-words, 32-bit words, 64-bit double-words

IBM 360: Implementation

	Model 30	. . .	*Model 70*
Storage	8K - 64 KB		256K - 512 KB
Datapath	8-bit		64-bit
Circuit Delay	30 nsec/level		5 nsec/level
Local Store	Main Store		Transistor Registers
Control Store	Read only 1μsec (i.e. microcoded)		Conventional circuits

IBM 360 instruction set architecture completely hid the underlying technological differences between various models.

With minor modifications it survives till today

High performance competitor to IBM/S360

CDC 6600 *Seymour Cray, 1964*

a scientific supercomputer

- A fast pipelined machine with 60-bit words
- Ten functional units
  - Floating Point: adder, multiplier, divider
  - Integer: adder, multiplier
  - ...
- Hardwired control (no microcoding)
- Dynamic scheduling of instructions using a scoreboard (see Appendix A)
- Ten Peripheral Processors for Input/Output
  - a fast time-shared 12-bit integer ALU
- Very fast clock
- Novel freon-based technology for cooling

Seymour's 5P's: Packaging, Plumbing (bits and heat flow), Parallelism, Programming, and understanding Programs

CDC6600: Fastest Machine 1964-1969

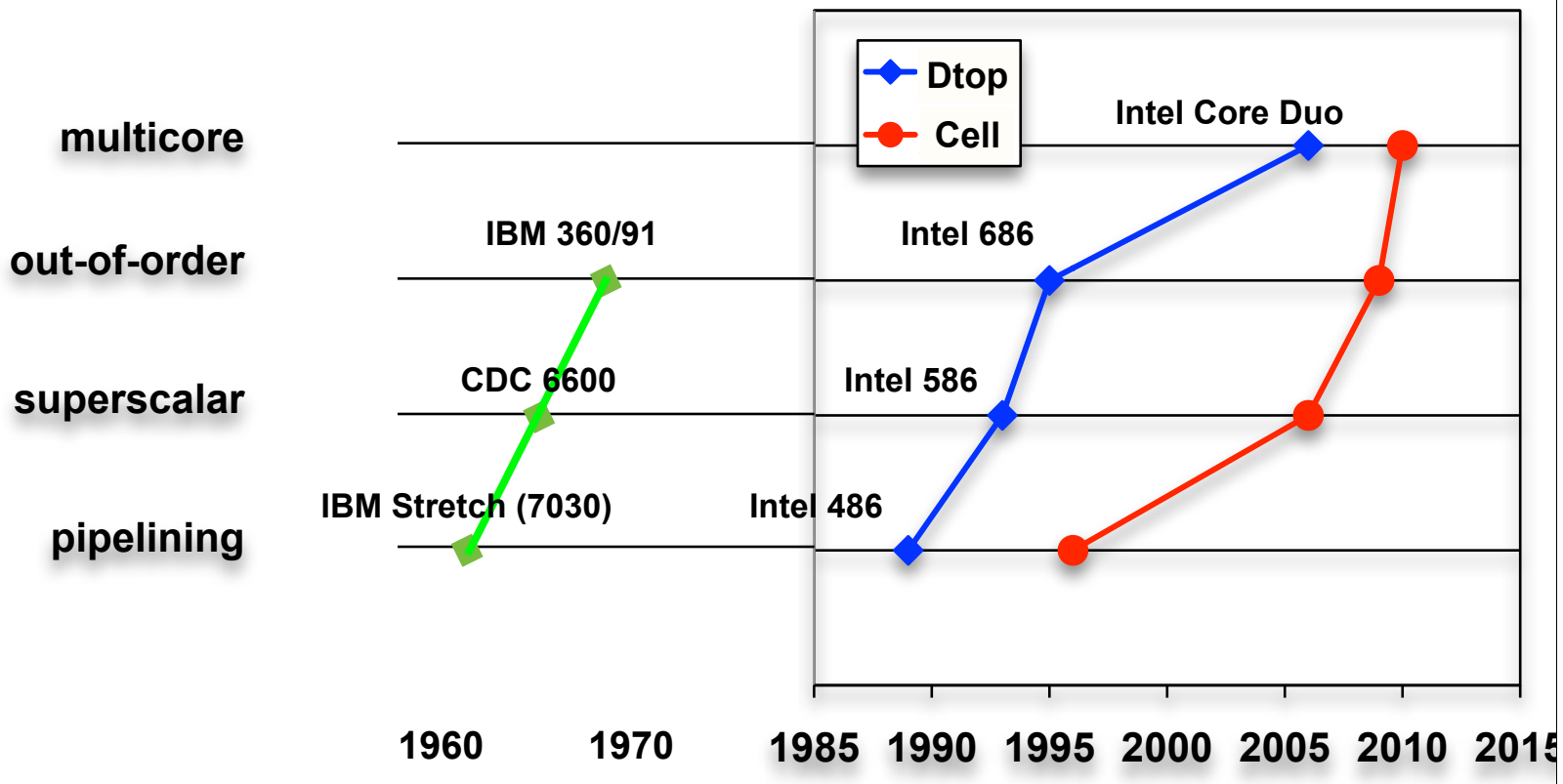
"Last week, Control Data ... announced the 6600 system. I understand that in the laboratory developing the system there are only 34 people including the janitor. Of these, 14 are engineers and 4 are programmers... Contrasting this modest effort with our vast development activities, I fail to understand why we have lost our industry leadership position by letting someone else offer the world's most powerful computer." -- T.J. Watson, IBM CEO

"It seems like Mr. Watson has answered his own question."

-- Seymour Cray

Cell Phone ("Mobile Application") Processors Recapitulate Desktop Processors Recapitulate Mainframe Processors

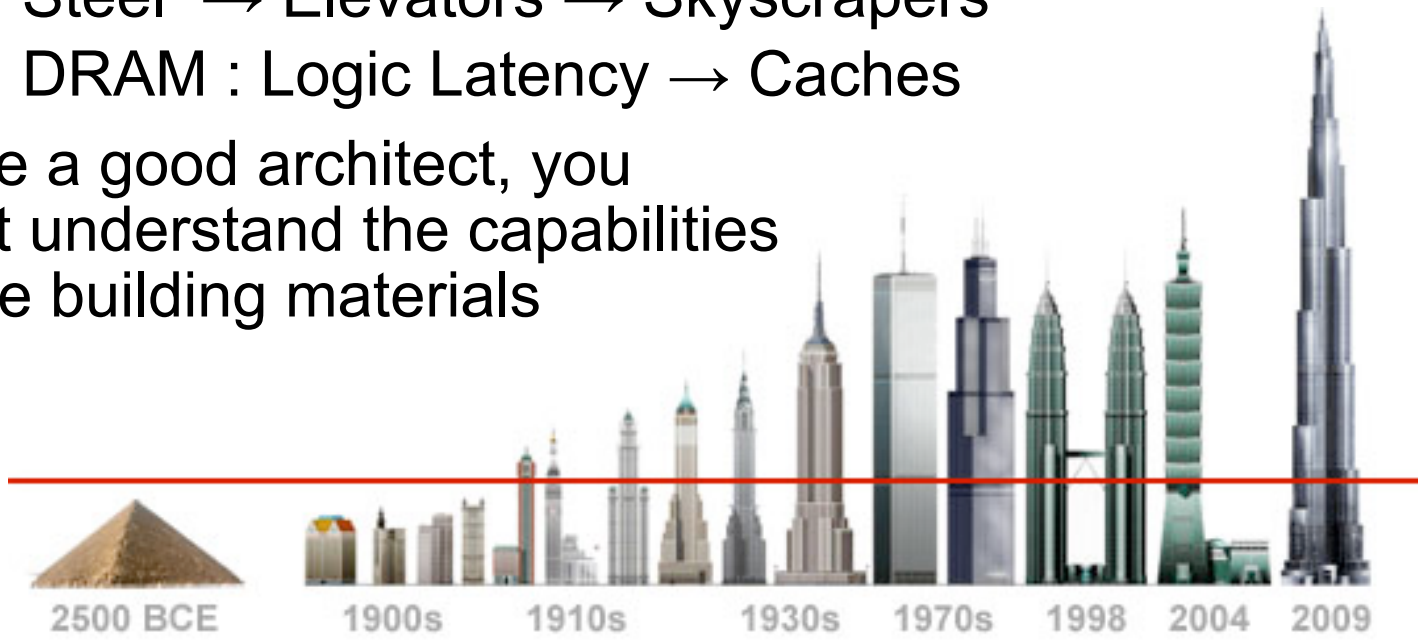
- The evolution of cell phone processors mirrors that of microprocessors, which mirrors that of mainframes

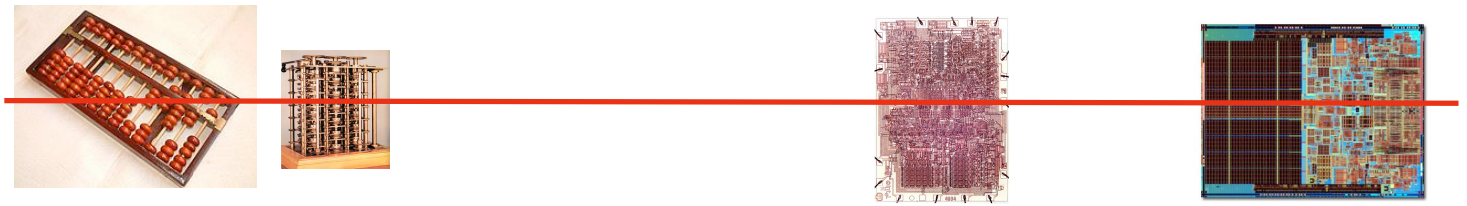


Understanding The Architect's Raw Materials

Modern Day Architecture is a Consequence of Modern Construction Properties

- E.g., Stone → Pyramids
- E.g., Steel → Elevators → Skyscrapers
- E.g., DRAM : Logic Latency → Caches
- To be a good architect, you must understand the capabilities of the building materials

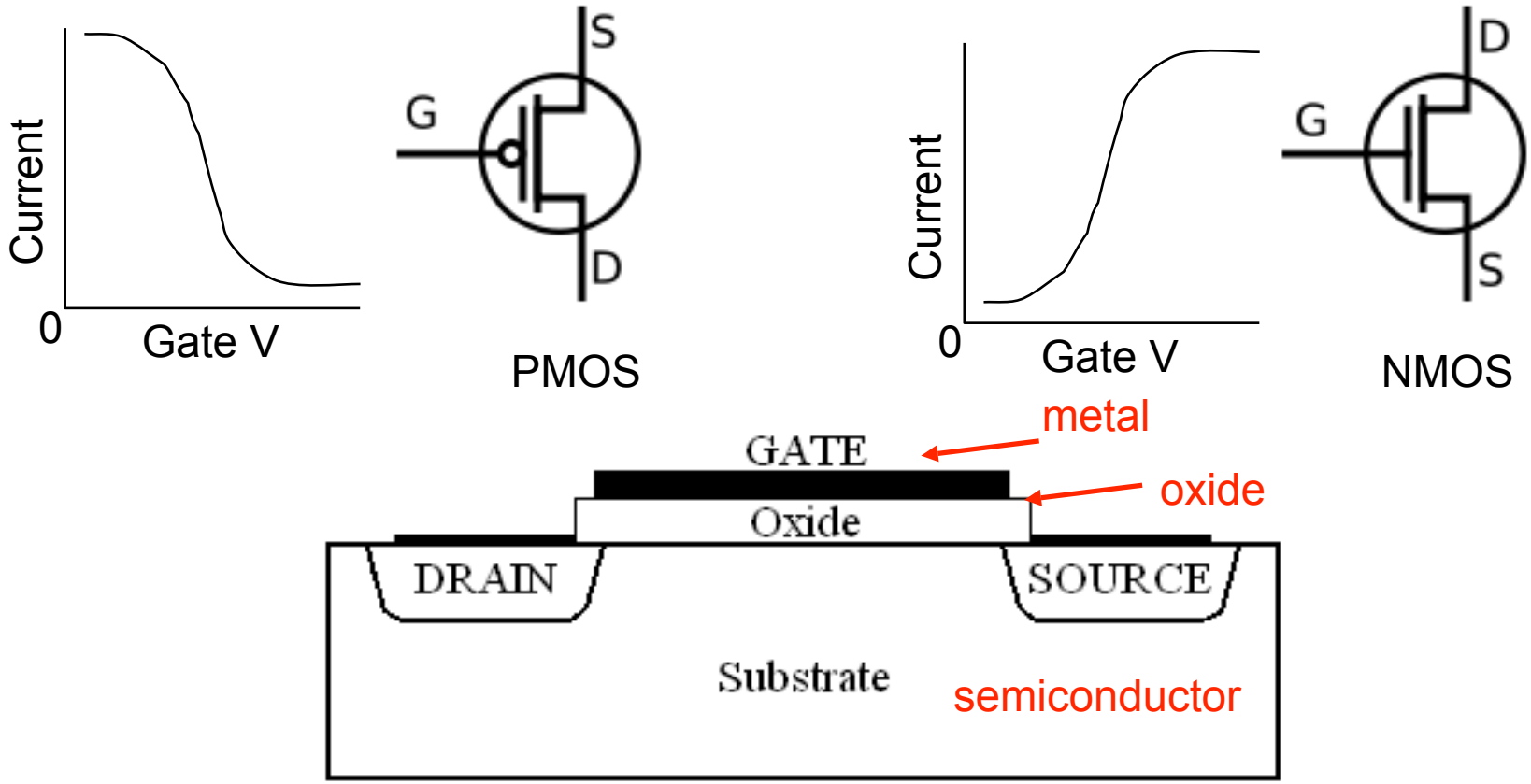




(buildings: http://www.newraleigh.com/articles/archive/edifice-rex/)

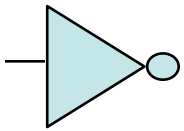
CMOS Logic: One Transistor

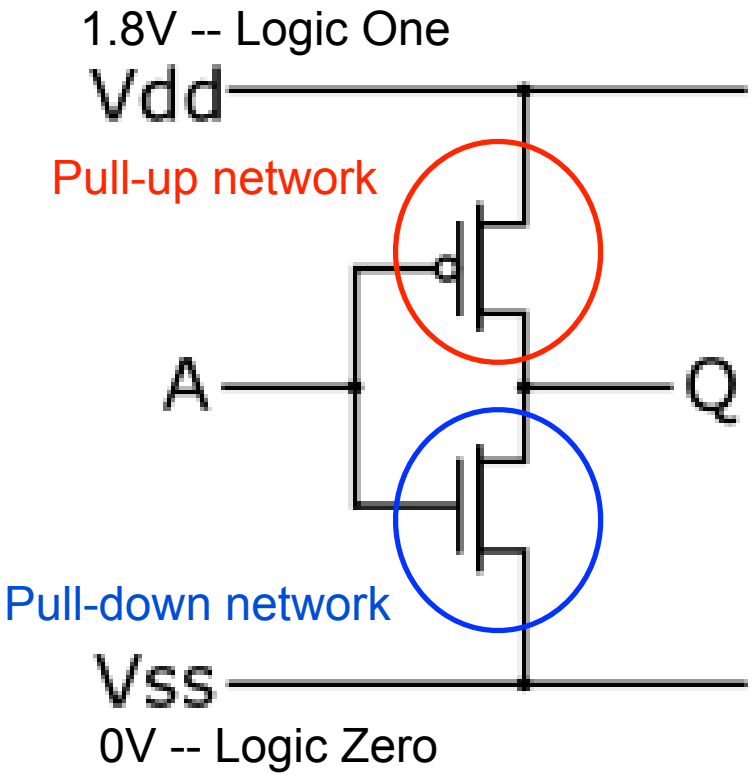
- Complimentary Metal Oxide Semiconductor
  - Logic family (an abstraction!)

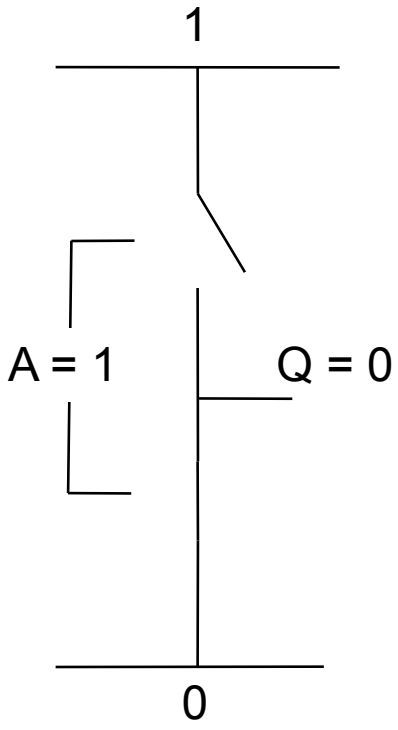


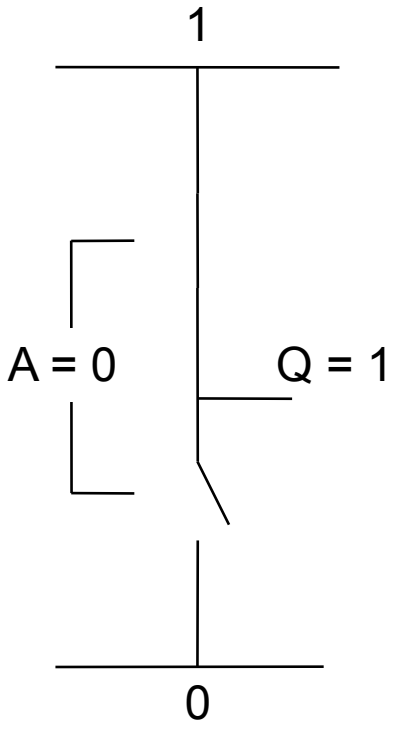
CMOS Logic: One Gate

- Complimentary MOS

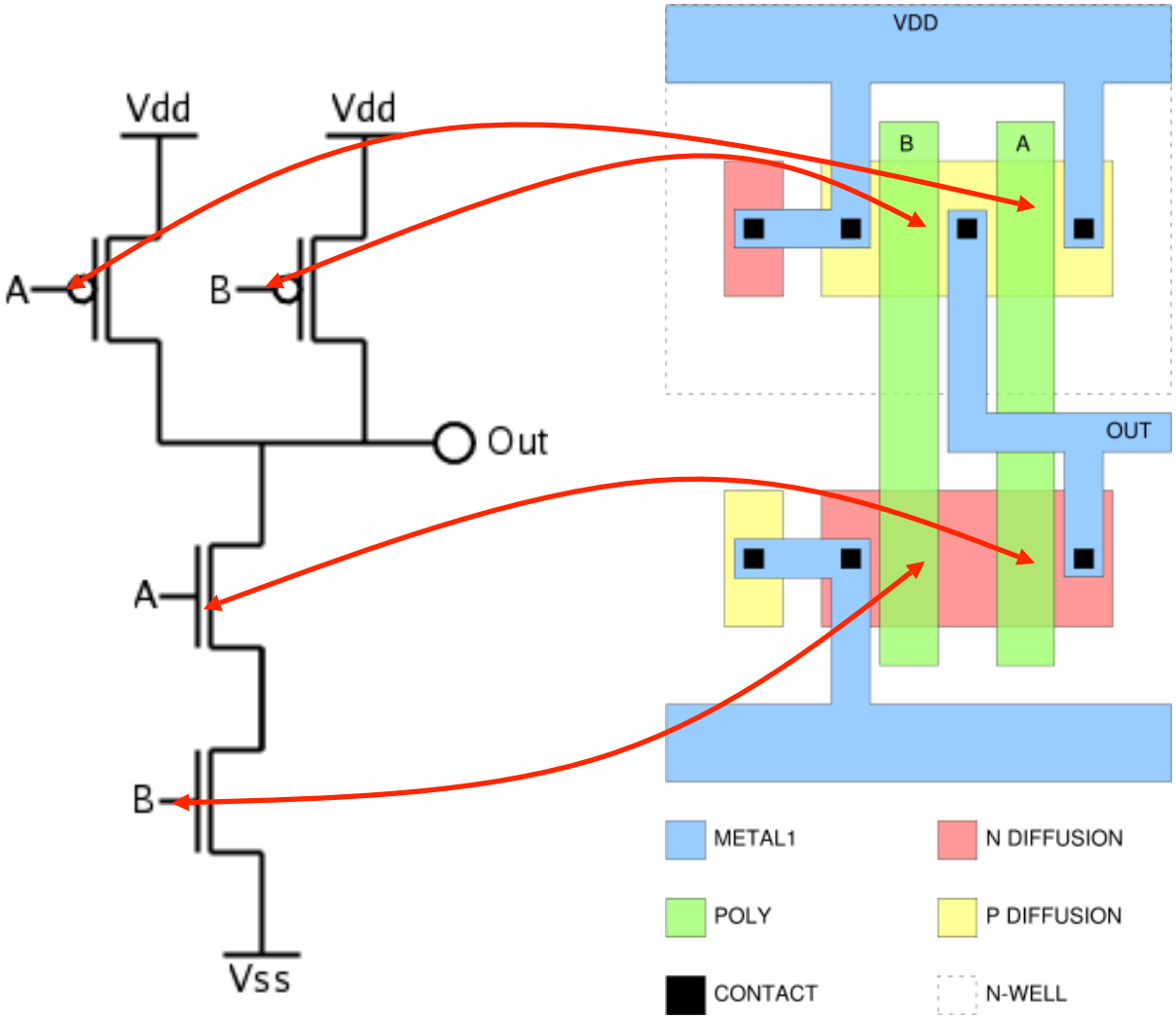






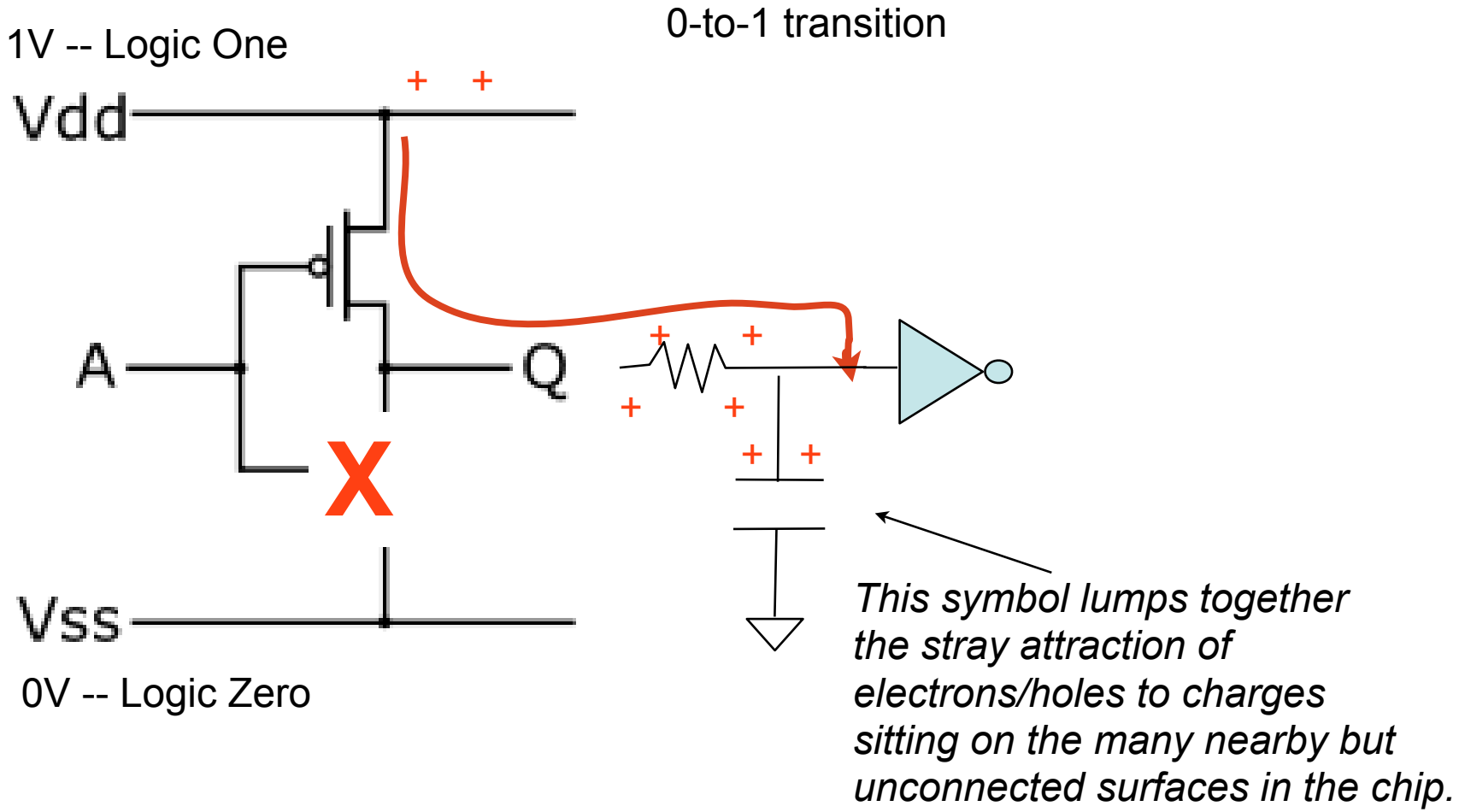


CMOS Gate: Layout



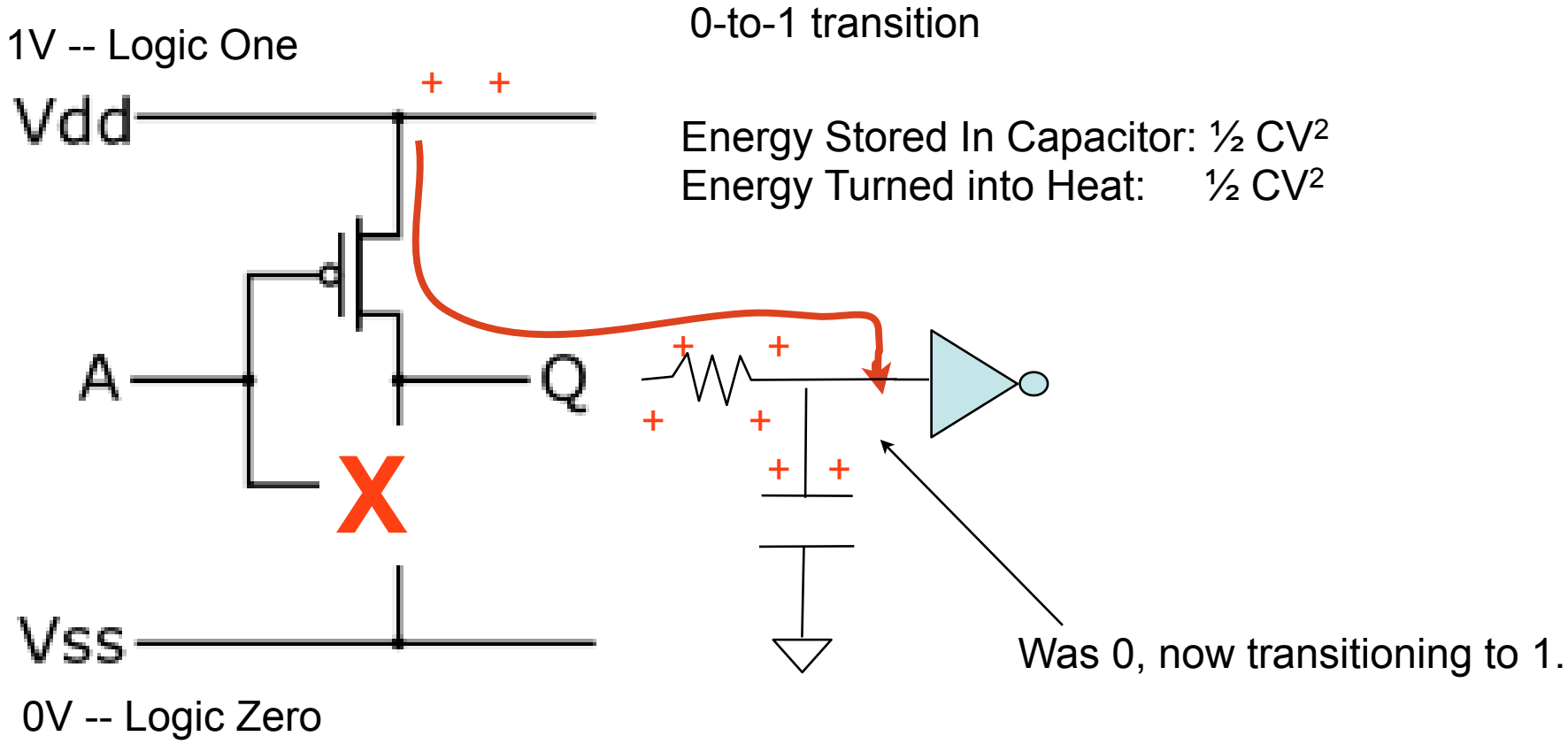
CMOS Logic: Capacitance

- Connection to Vdd transfers charge to the wires and gates on the output; these surfaces form “parasitic” capacitance with other non-connected surfaces at different voltages



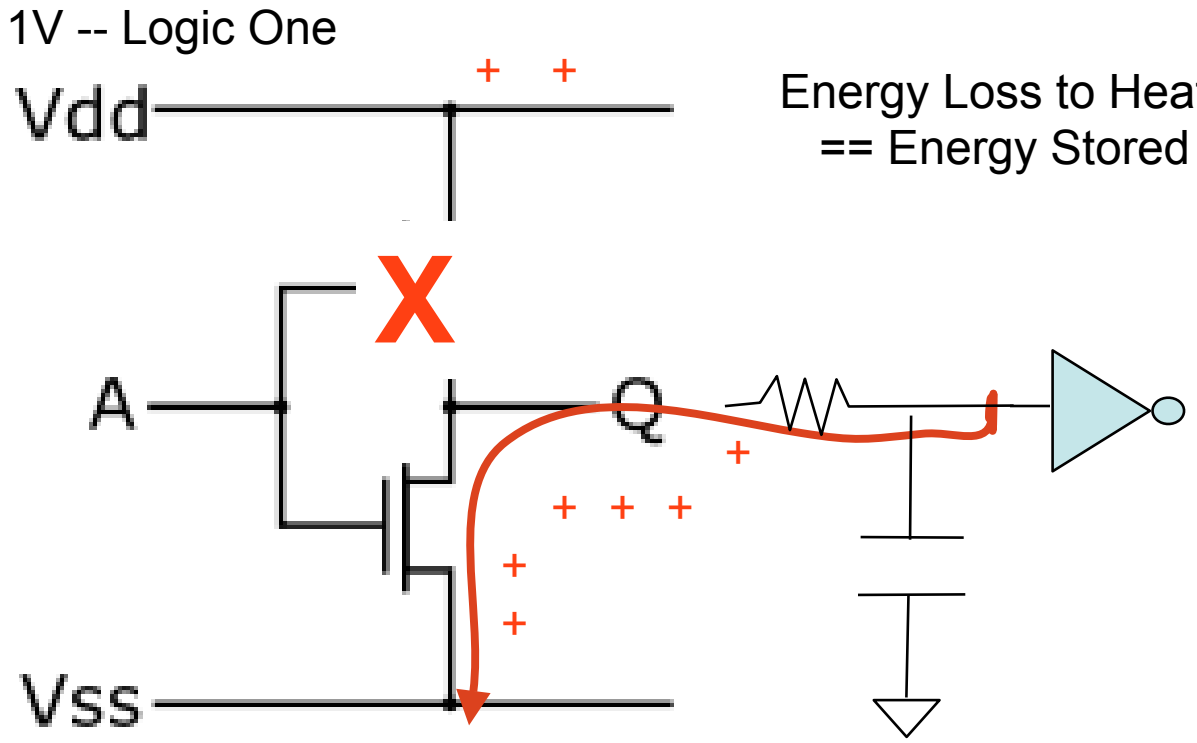
CMOS Logic: Cap~Energy

- Energy is expended when charging the parasitic capacitance (i.e 0-to-1 transition);
- Some energy is stored via the parasitic capacitance



CMOS Logic: Cap~Energy

- Connection to Vss discharges those capacitances to the ground rail.
- Where does the energy stored in capacitors go? Heat.



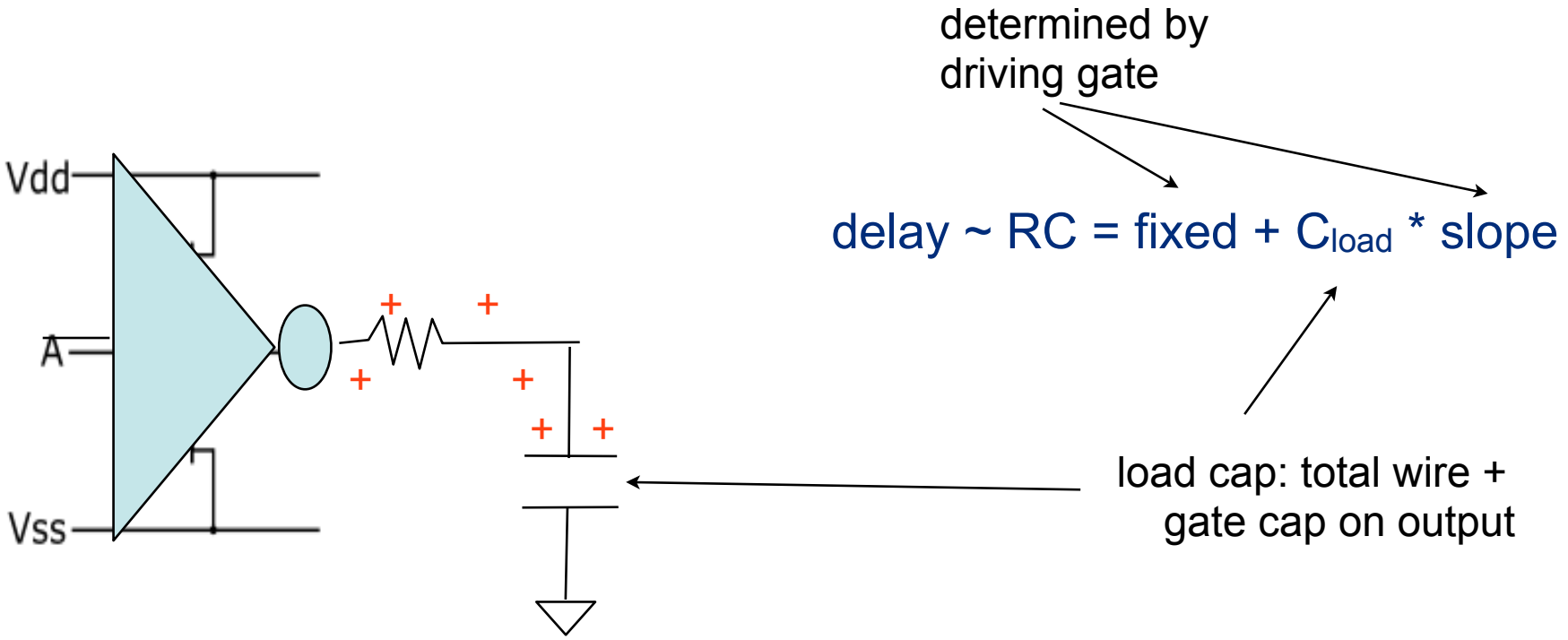
Energy Loss to Heat: ½ CV^2
== Energy Stored In Capacitor

Since # 0→1 transitions ≈ # 1→0 transitions, we often just count one or the other and assign the aggregate cost: CV^2.

CMOS Logic: Delay

- Delay of switching a transistor the corresponds to delay of discharging or charging load capacitance over a resistor

  (i.e. the on resistance of the charging/discharging transistor.)

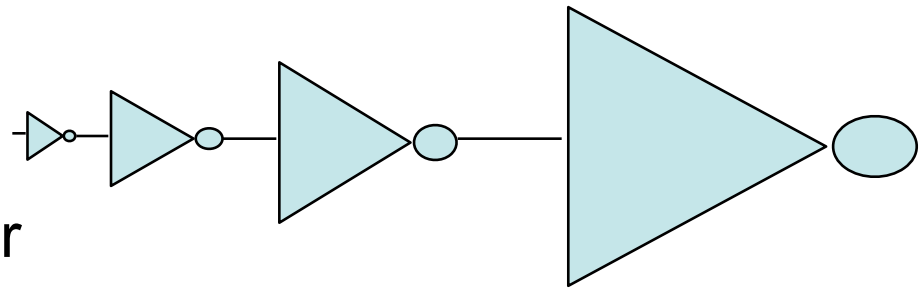


CMOS Logic: Driving Large Loads

- Best gate to large loads: Inverter
  - Lowest Capacitance and Lowest Resistance
- The larger the inverter:
  - the larger the transistors
  - the smaller the resistance through the transistor
    - smaller slope
  - the larger the capacitance of the inverter itself
    - larger fixed delay

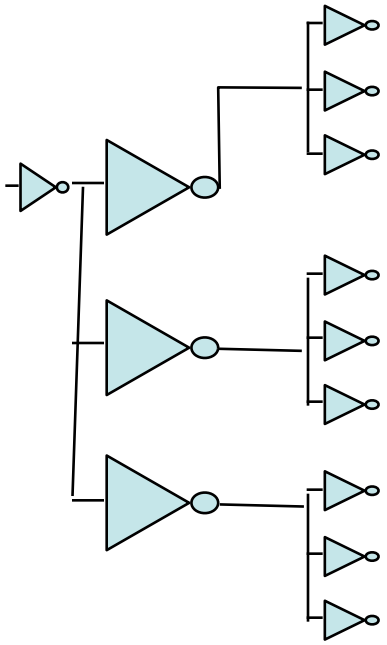
$$\text{delay} = \text{fixed} + C_{load} * \text{slope}$$

- Driving large loads typically uses stages of inverters, where the load of the output is 4x the input load of each inverter



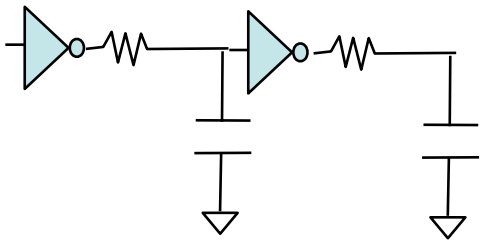
CMOS Logic: Driving Large Loads

- Examples of large loads:
  - Stall cycle in your 5-stage MIPS pipeline
    - fans out to maybe 20 32-bit registers spread across large sections of the chip (wire capacitance!)
    - load: 640x → lg_4(640) -> 4-6 inverters stages
  - I/O Driver for Package Pin (~1000x cap of a gate)
  - Clock Tree
    - every register in your design!



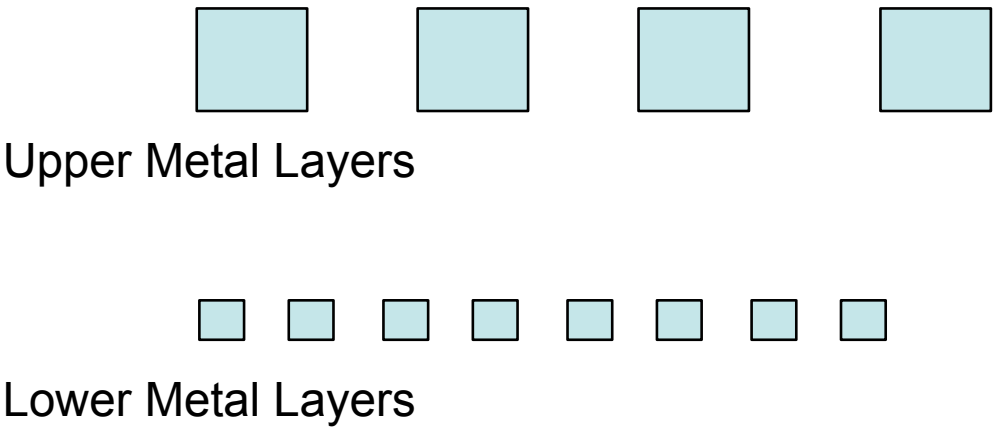
Wire Properties

- 1 mm of wire:
  - ~50x the capacitance of a simple gate
    - scales linearly with wire length
  - high resistance
    - scales linearly with wire length
  - delay ~ RC
    - delays scale quadratically with wire length
    - we can add distributed *repeaters* (large inverters)
      - make delay linear
      - burns additional power
      - tradeoff of delay through repeaters versus delay in wire



Wire Properties

- Latencies:
  - 30 cm in 1 ns (light in a vacuum)
  - 15 cm in 1 ns (light in silicon dioxide)
  - 1 cm in 1 ns (electrical field; RC limited)
    - including repeaters, etc.
    - slower wire
- How to make a faster wire



How does capacitance compare? Why?

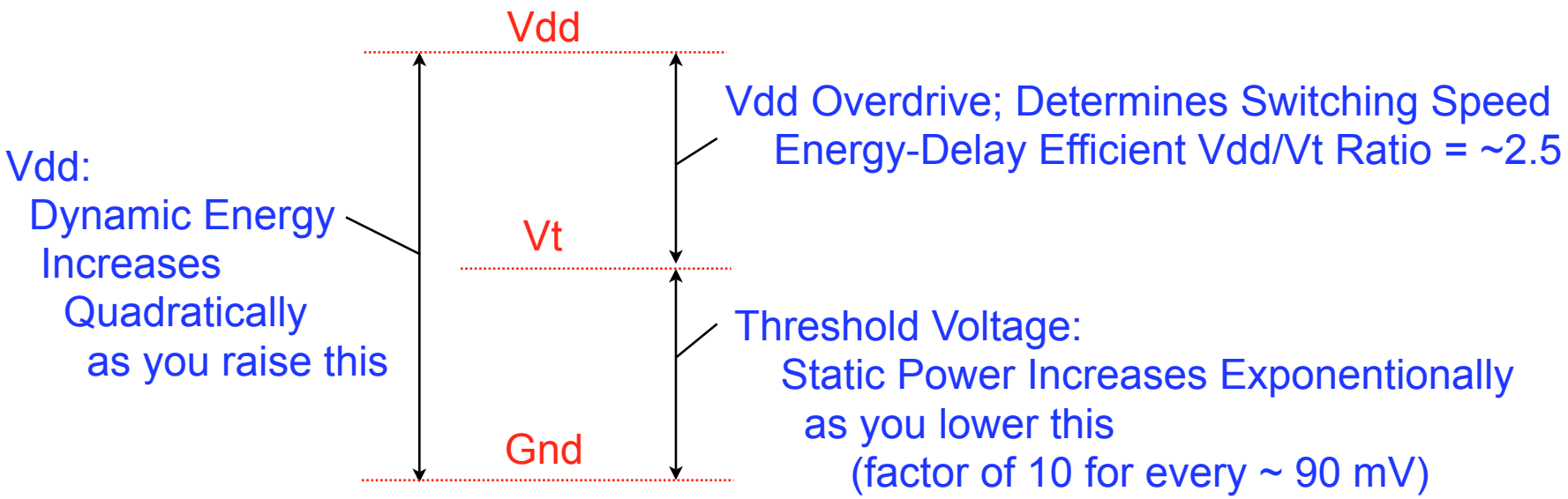
How does resistance compare? Why?

How does latency compare?

How does bandwidth compare?

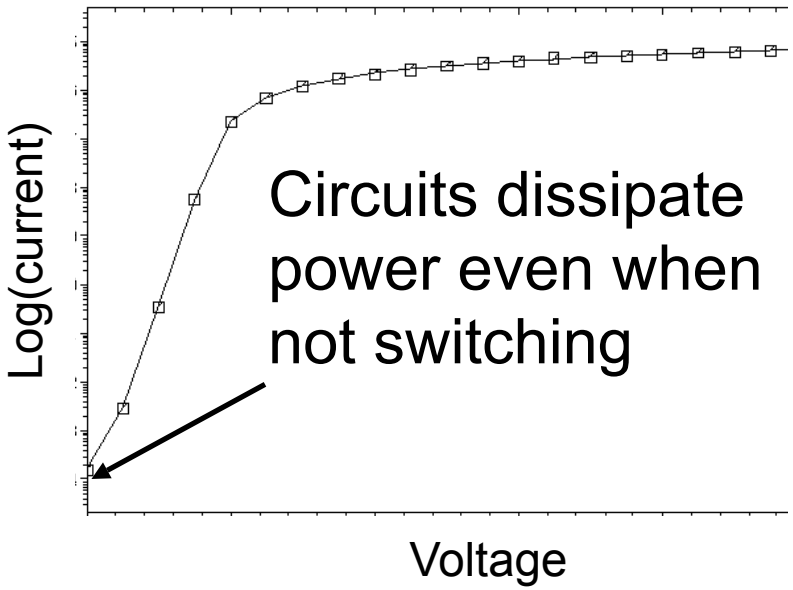
Optimizing CMOS Power

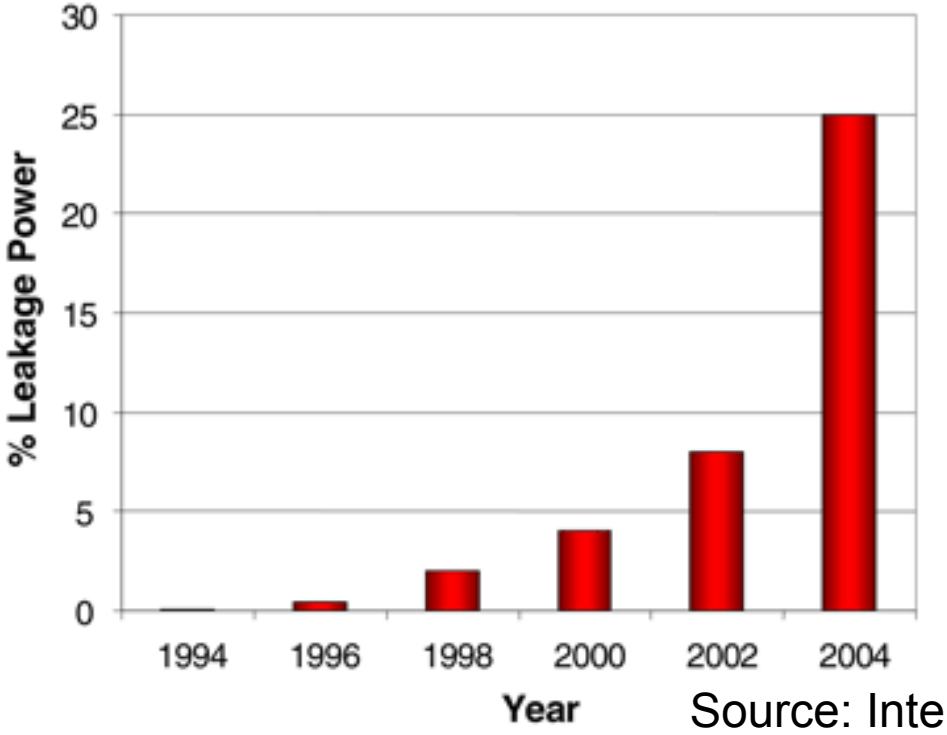
- Two key parameters are settable on a transistor:
  - Supply power (Vdd) - around 700 mV to 1 V
  - Threshold Voltage (Vt) - around 250 mV to 400 mV



CMOS Leakage

- Leakage is a growing problem, but it's a settable parameter.
  - We can limit it, but it impacts speed.
  - 30% of power in leakage seems like a good balance point in practice for desktop processors.





Source: Intel

Silicon Memories

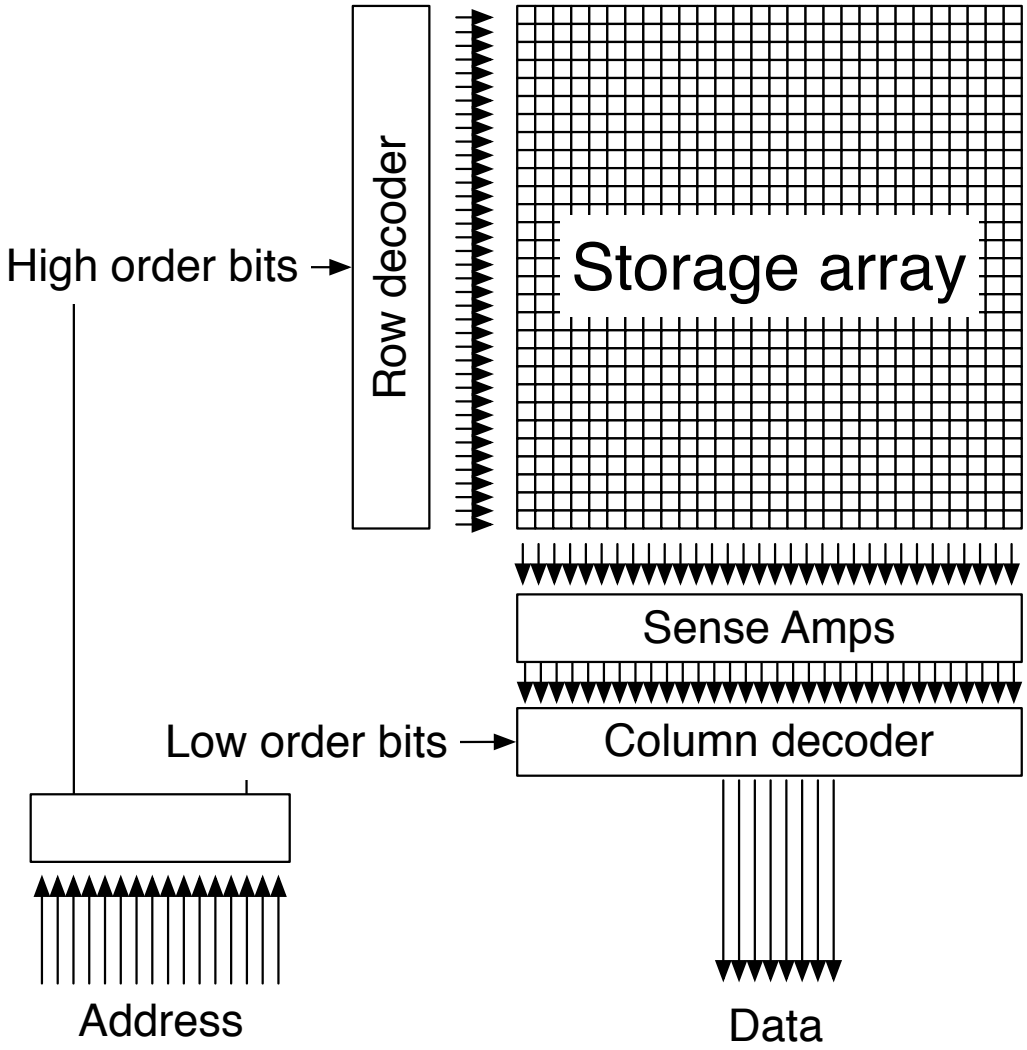
- Why store things in silicon?
  - It's fast!!!
  - Compatible with logic devices (mostly)
- The main goal is to be cheap
  - Dense -- The smaller the bits, the less area you need, and the more bits you can fit on a chip/wafer/through your fab.
  - Bit sizes are measured in F^2 -- the smallest feature you can create.
  - F^2 is a function of the memory technology, not the manufacturing technology.

Questions

- What physical quantity should represent the bit?
  - Voltage/charge -- SRAMs, DRAMs, Flash memories
  - Magnetic orientation -- MRAMs
  - Crystal structure -- phase change memories
  - The orientation of organic molecules -- various exotic technologies
  - All that's required is that we can sense it and turn it into a logic one or zero.
- How do we achieve maximum density?
- How do we make them fast?

Anatomy of a Memory

- Dense: Build a big array
  - bigger the better
  - less other stuff
  - Bigger -> slower
- Row decoder
  - Select the row by raising a "word line"
- Column decoder
  - Select a slice of the row
- Decoders are pretty big.



The Storage Array

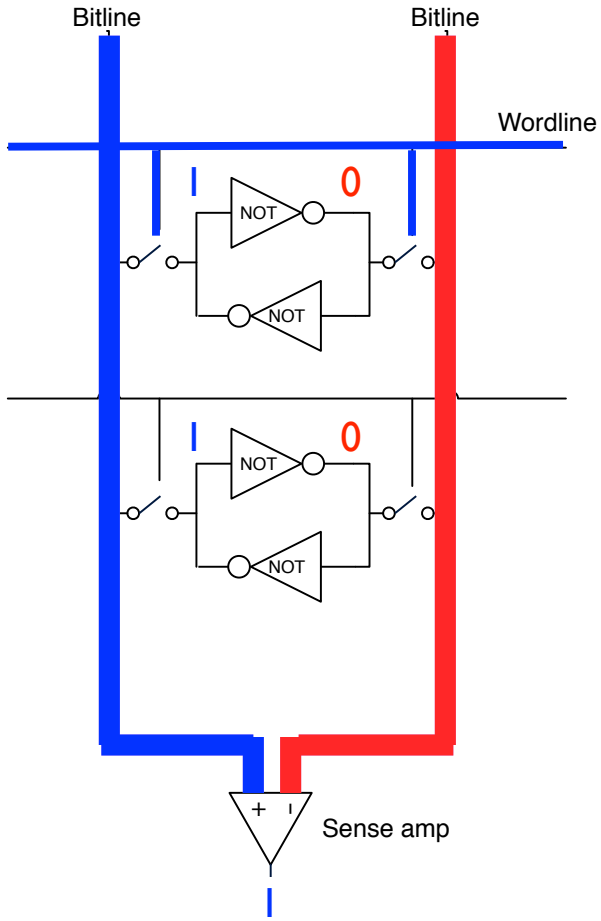
- Density is king.
  - Highly engineered, carefully tuned, automatically generated.
  - The smaller the devices, the better.
- Making them big makes them slow.
  - Bit/word lines are long (millimeters)
  - They have large capacitance, so their RC delay is long
  - For the row decoder, use large transistors to drive them hard.
  - For the bit cells...
    - There are lots of these, so they need to be as small as possible (but not smaller)

Sense Amps

- Sense amplifiers take a difference between two signals and amplify it
- Two scenarios
  - Inputs are initially equal ("precharged") -- they each move in opposite directions
  - One input is a reference -- so only one signal moves
- Frequently used in memories
  - Sense amps can detect small analog signals from the storage cell, and convert it into a logic one or logic zero.

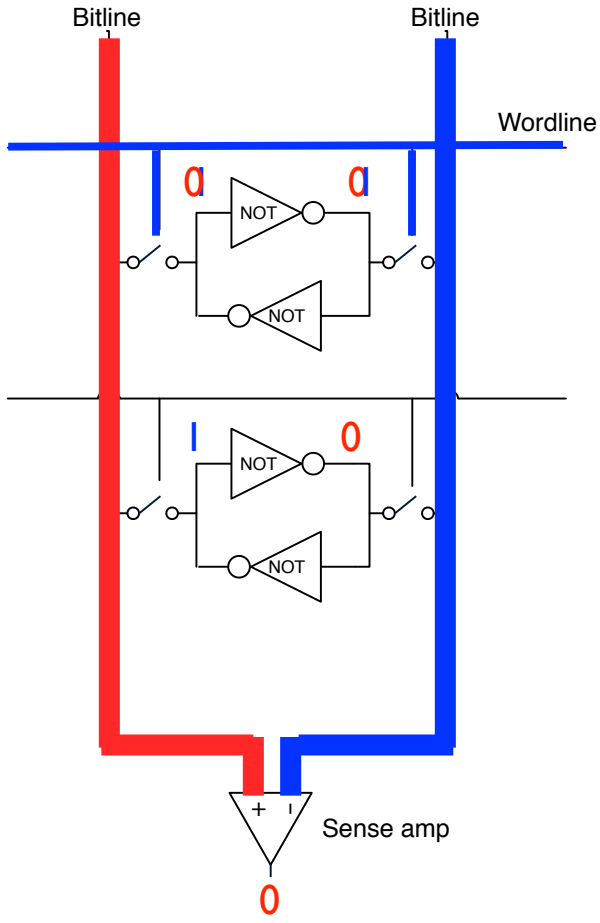
Static Random Access Memory (SRAM)

- Storage
  - Voltage on a pair of cross-coupled inverters
  - Durable in presence of power
- To read
  - Pre-charge two bit lines to Vcc/2
  - Turn on the "word line"
  - Read the output of the sense-amp



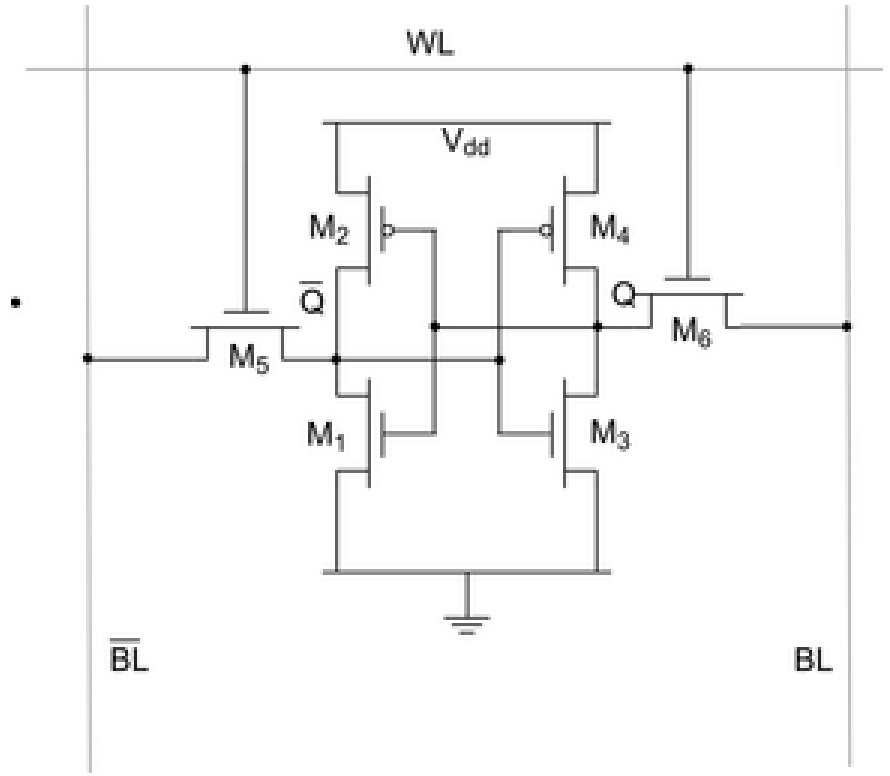
SRAM Writes

- To write
  - Turn off the sense-amp
  - Turn on the wordline
  - Drive the bitlines to the correct state
  - Turn off the wordline



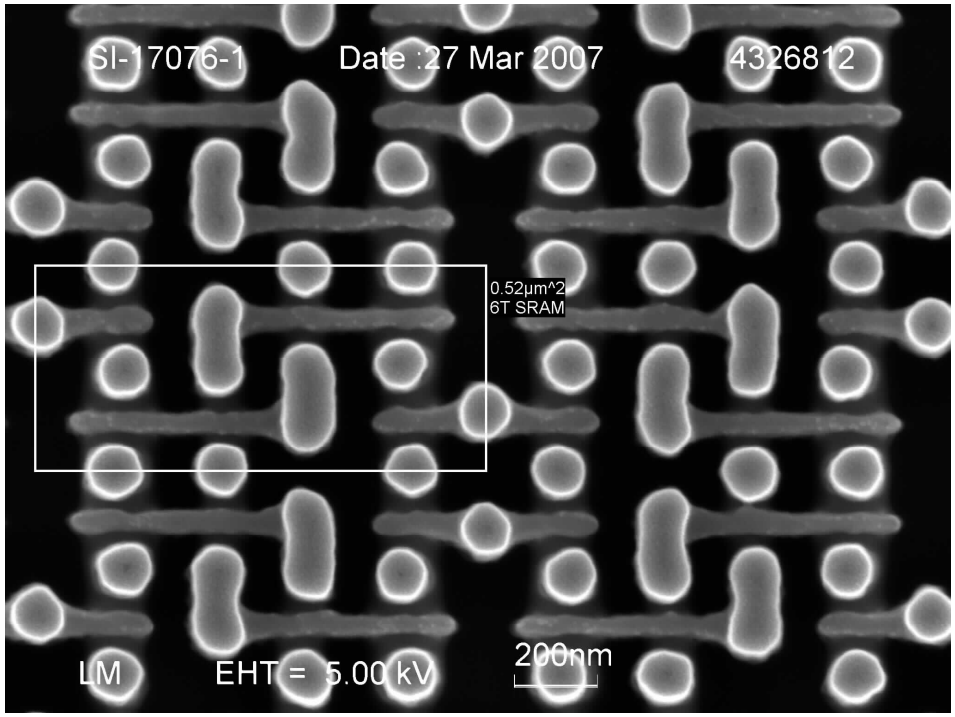
Building SRAM

- This is "6T SRAM"
- 6 "basic devices" is pretty big
- SRAMs are not dense



SRAM Density

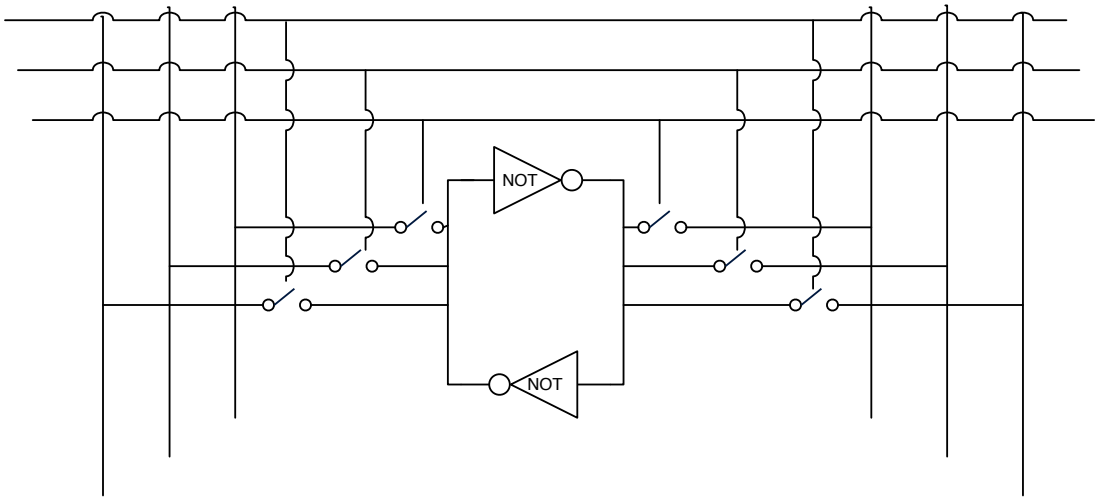
- At 65nm: 0.52um^2
- 123-140 F^2
- 1 F^2 is one "square feature"
- [ITRS 2008]



65nm TSMC 6T SRAM

SRAM Ports

- Add word and bit lines
- Read/write multiple things at once
- Density decreases quadratically
- Bandwidth increase linearly



SRAM Performance

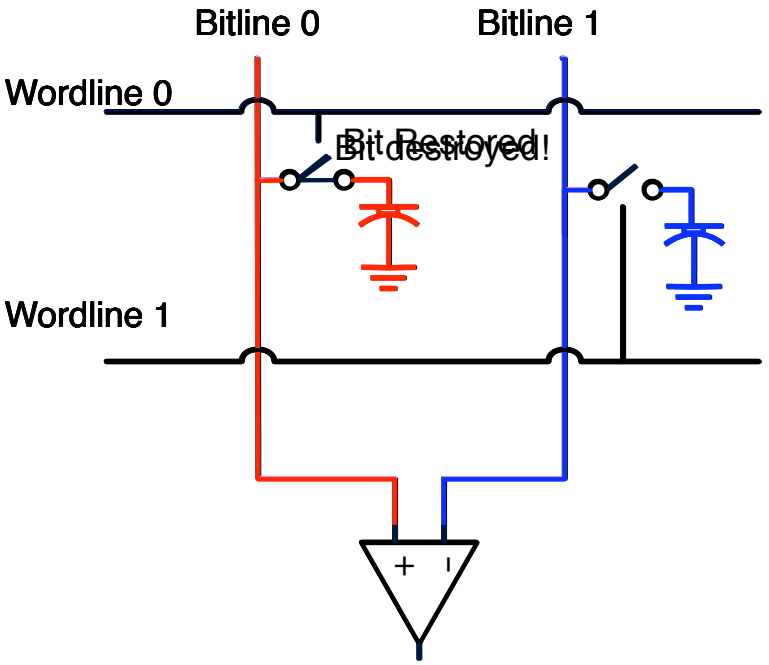
- Read and write times
  - 10s-100s of ps
- Bandwidth
  - Registers -- 324GB/s
  - L1 cache -- 128GB/s
  - Samsung K7D323674C -- 3.6GB/s
- Durability
  - Infinite (not quite actually, but very close)

SRAM's future

- SRAM is a mature technology. No new, big breakthroughs or advances are expected beyond CMOS scaling.

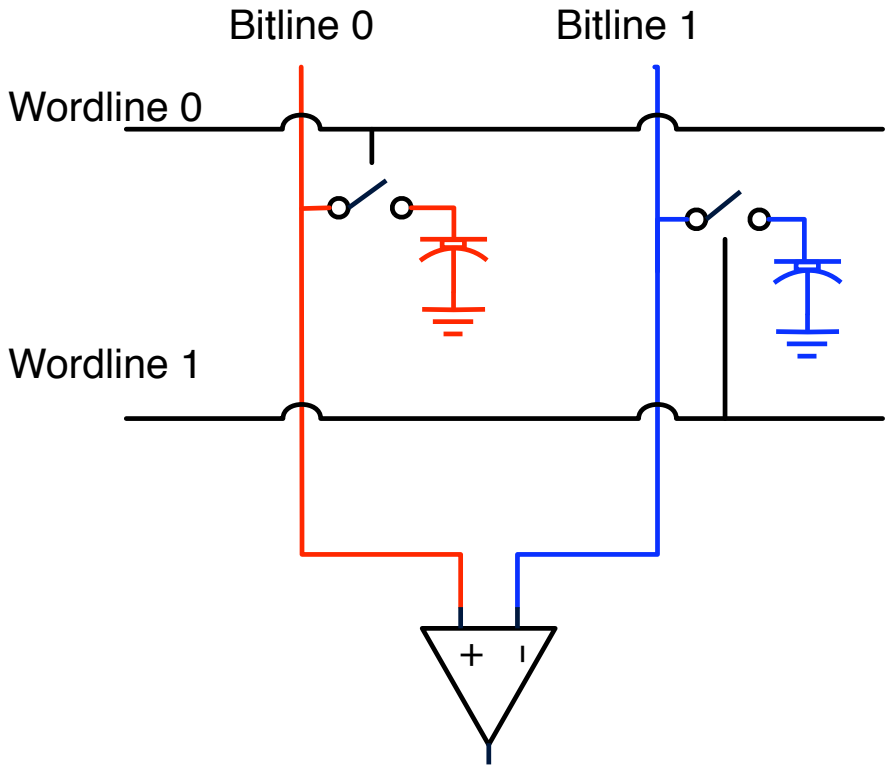
Dynamic Random Access Memory (DRAM)

- Storage
  - Charge on a capacitor
  - Decays over time (us-scale)
  - This is the "dyanamic" part.
  - About $6F^2$: 20x better than SRAM
- Reading
  - Precharge
  - Assert word line
  - Sense output
  - Refresh data

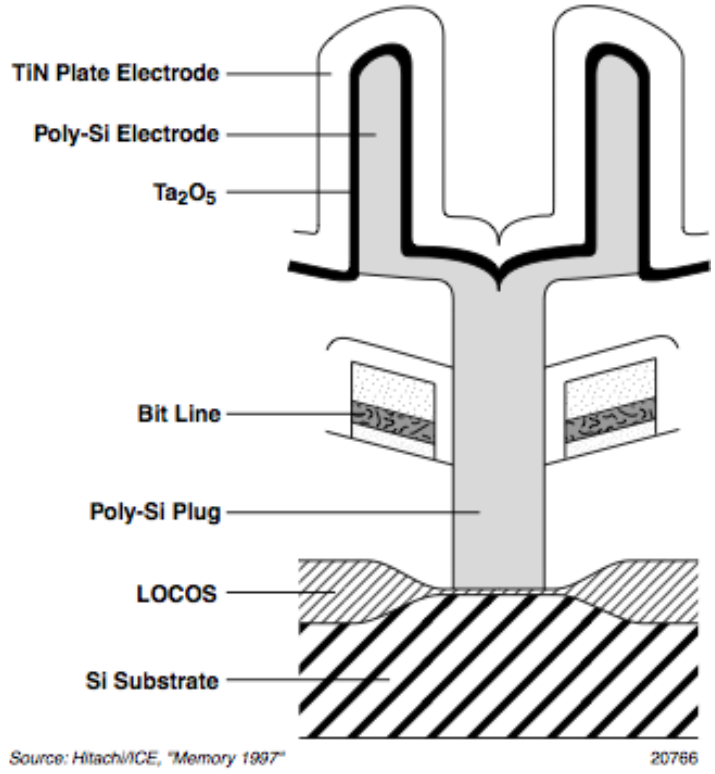


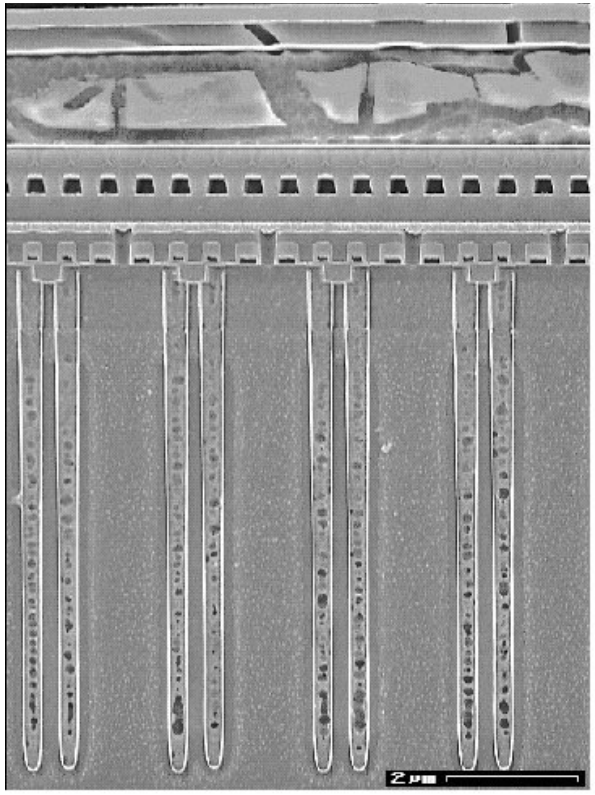
DRAM: Write and Refresh

- Writing
  - Turn on the wordline
  - Override the sense amp.
- Refresh
  - Every few micro-seconds, read and re-write every bit.
  - Consumes power
  - Takes time



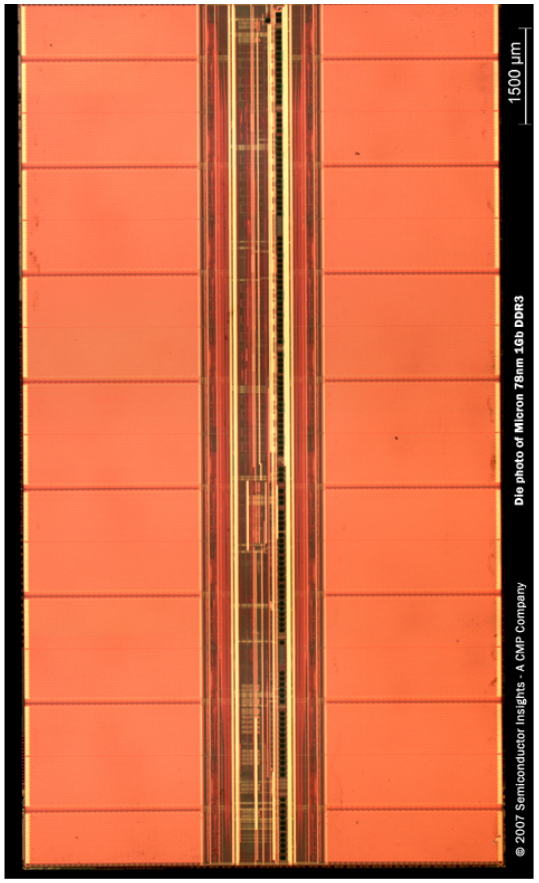
DRAM Lithography





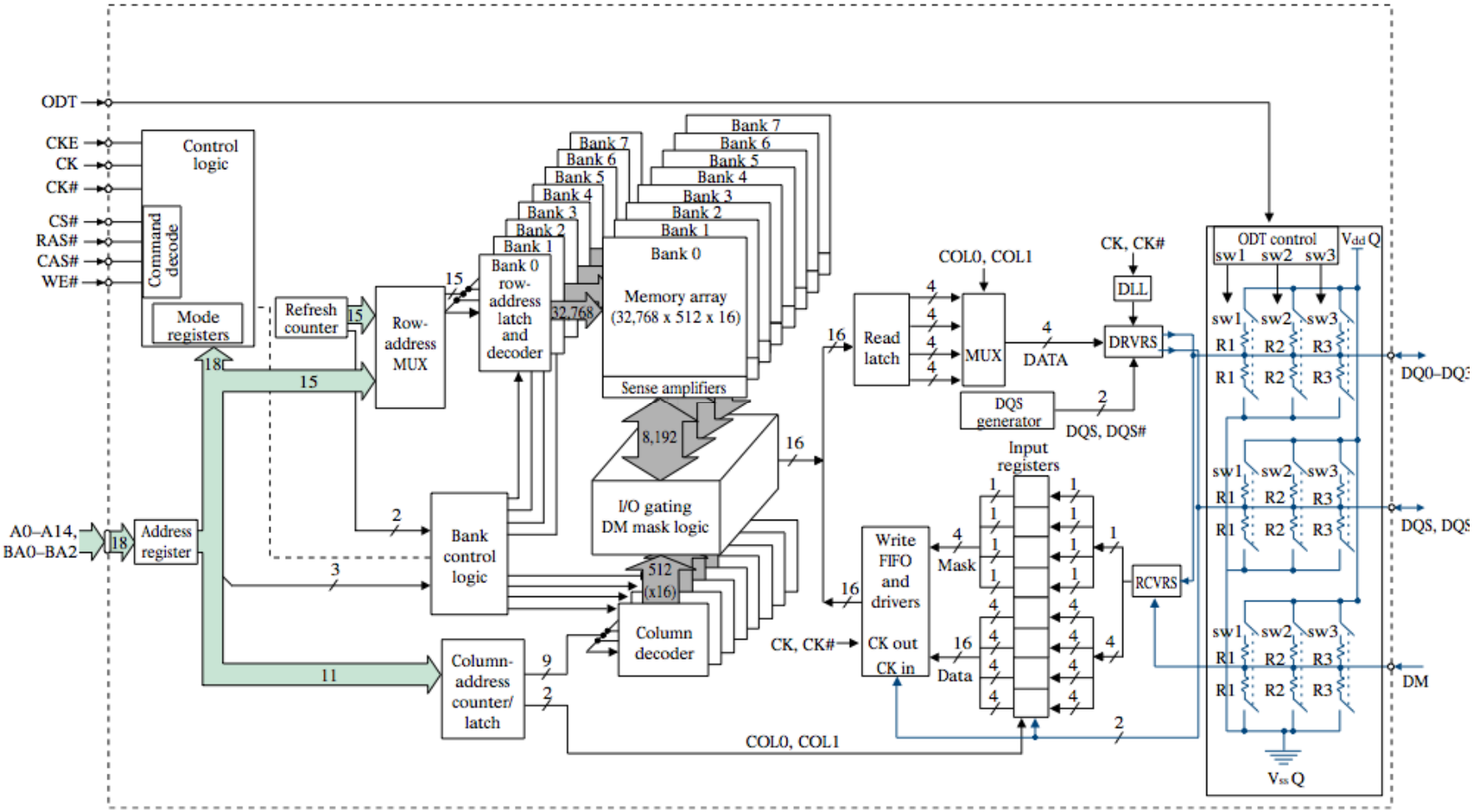
DRAM Devices

- There are many banks per die (16 at left)
  - Multiple can be active at once to hide latencies
  - Parallelism!!!
- Example
  - open bank 1, row 4
  - open bank 2, row 7
  - open bank 3, row 10
  - read bank 1, column 8
  - read bank 2, column 32
  - ...

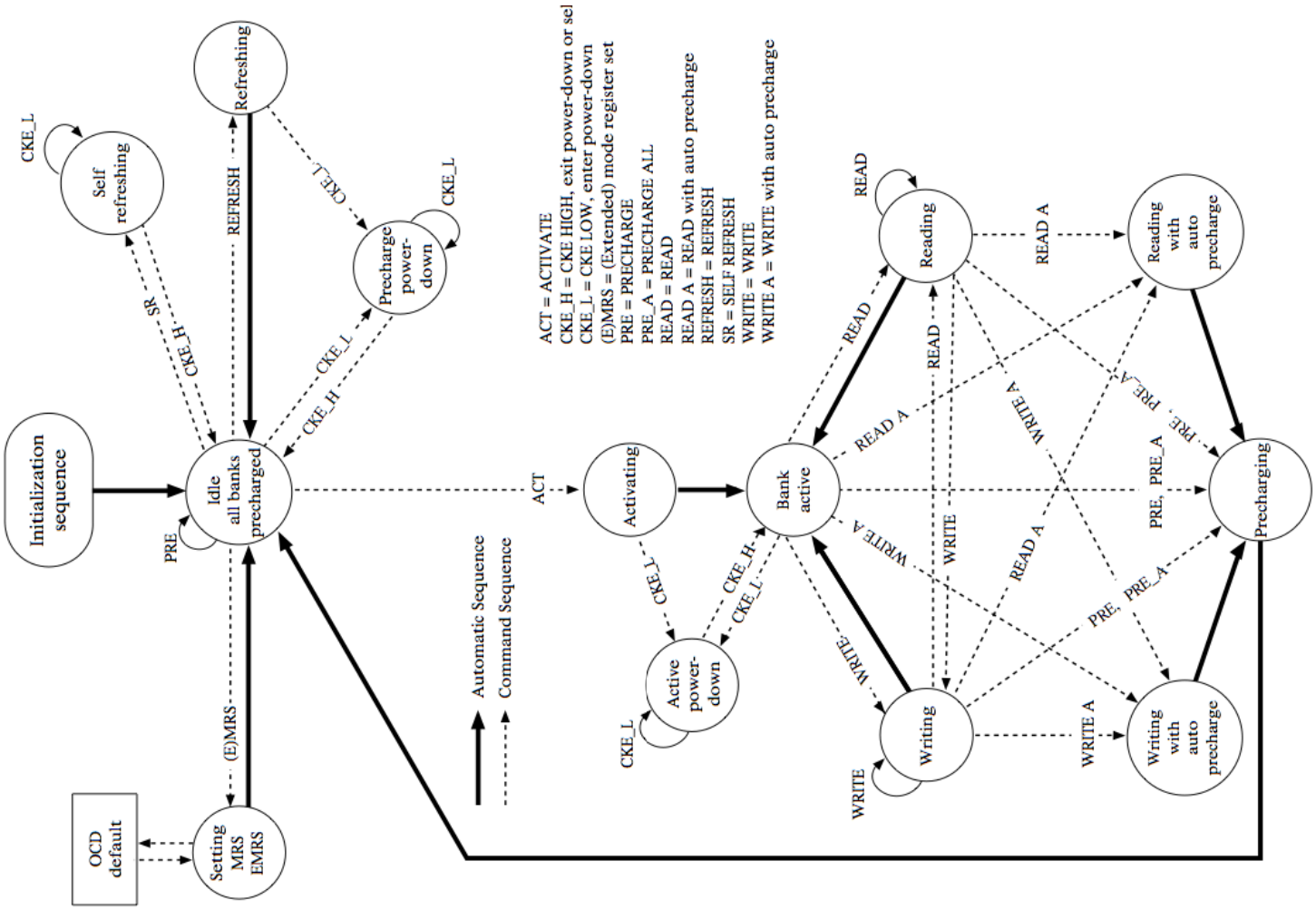


Micron 78nm 1Gb DDR3

DRAM: Micron MT47H512M4



DRAM: Micron MT47H512M4

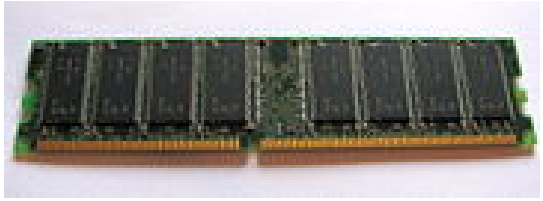


DRAM Variants

- The basic DRAM technology has been wrapped in several different interfaces.
- SDRAM (synchronous)
- DDR SDRAM (double data-rate)
  - Data clocked on rising *and* falling edge of the clock.
- DDR2
  - Example on previous slides
- DDR3
- RDRAM
  - Rambus RAM
- GDDR2-5 -- For graphics cards.
- FB-DIMMS

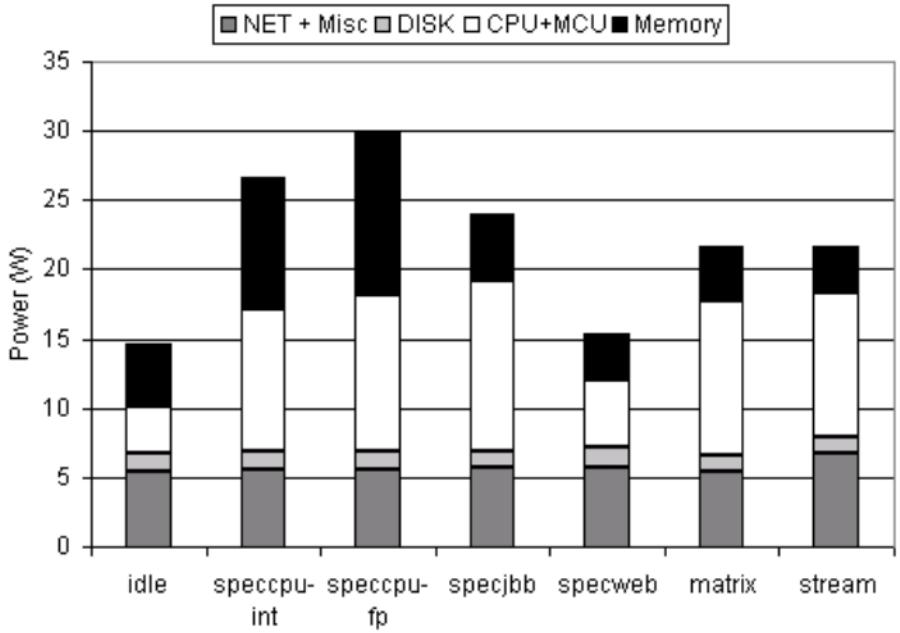
DDR2 SDRAM

- Fewer, larger banks.
- Pin count per package (DIMM): 14 address, 16 data
- DIMM data path is 64bits
- Data rate: up to 800Mhz DDR (1600Mhz effective)
- Bandwidth per DIMM GTNE: 12.8GB/s
  - *guaranteed not to exceed*
- Multiple DIMMs can attach to a bus
  - Reduces bandwidth/GB (a good idea?)



Power

- DRAM is a major power sink.
- Idle power: 2-4W/ DIMM
- Active power: 5-8W/ DIMM



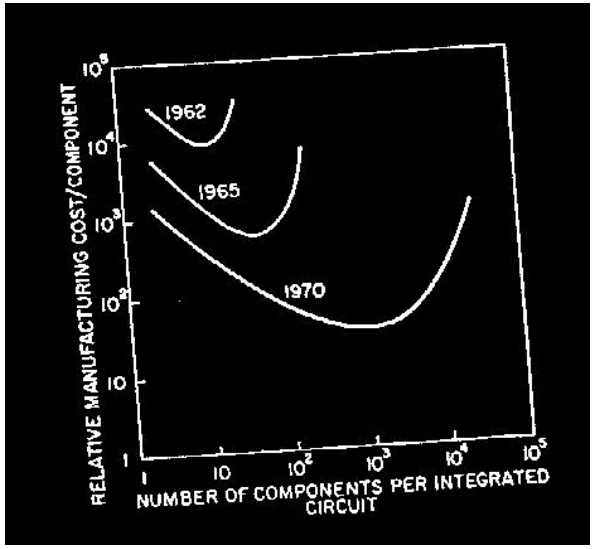
Economou, et. al 2006

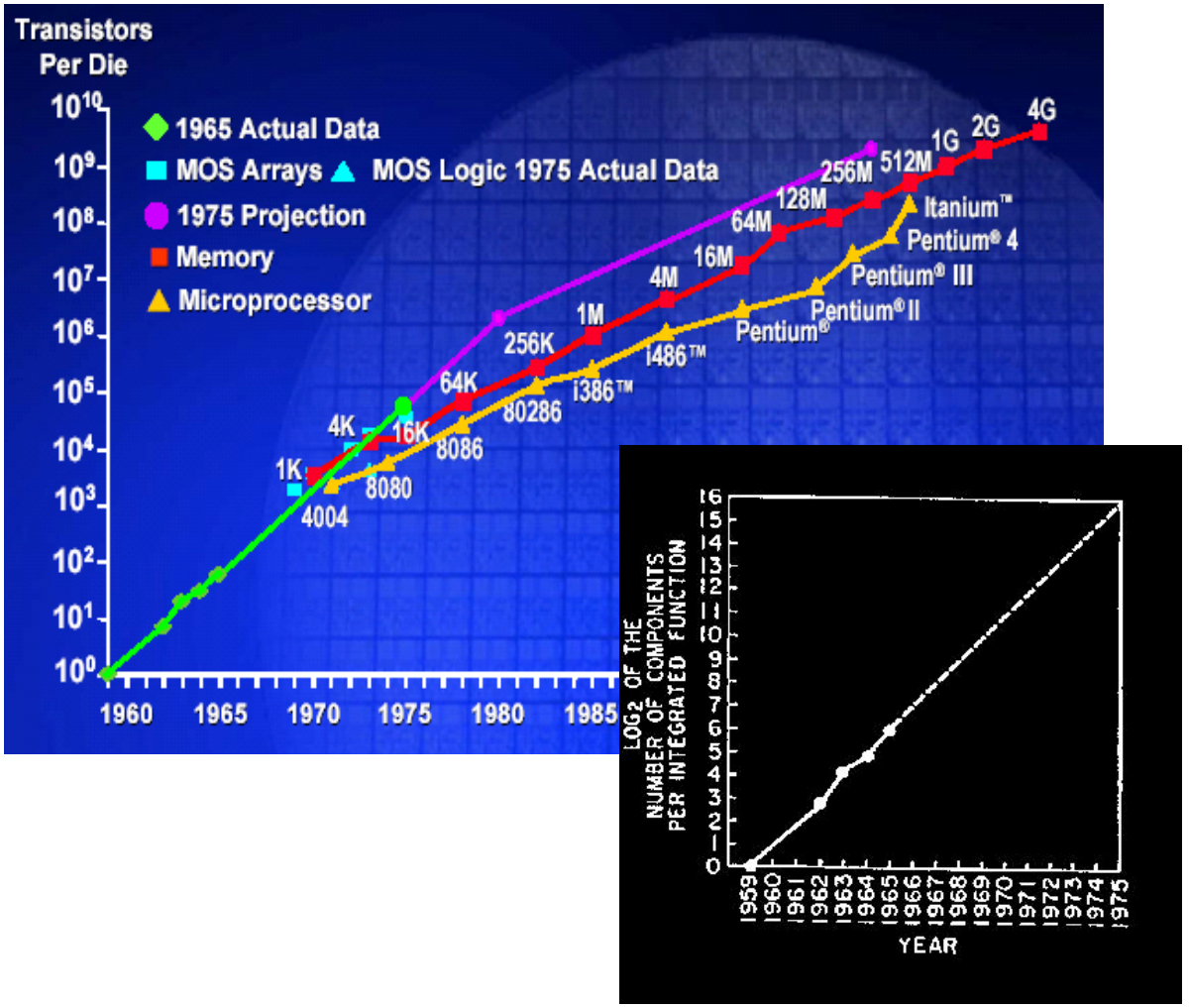
DRAM Scaling

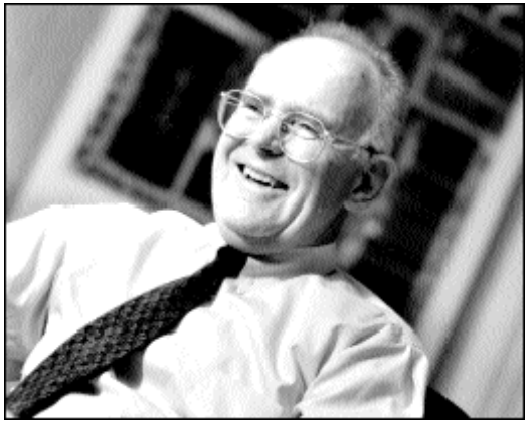
- Long term need for performance has driven DRAM hard
  - complex interface.
  - High performance
  - High power.
- DRAM used to be the main driver for process scaling, now it's flash.
- Scaling is expected to match CMOS tech scaling
- F^2 cell size will probably not decrease

Technology Scaling

Moore's Law: 2X transistors / "year"





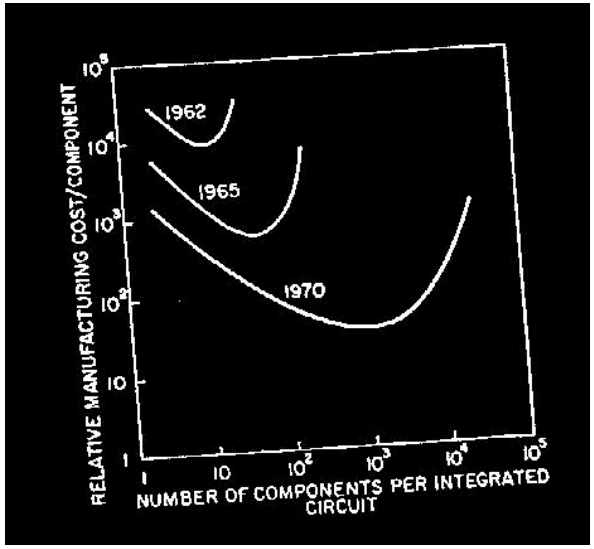


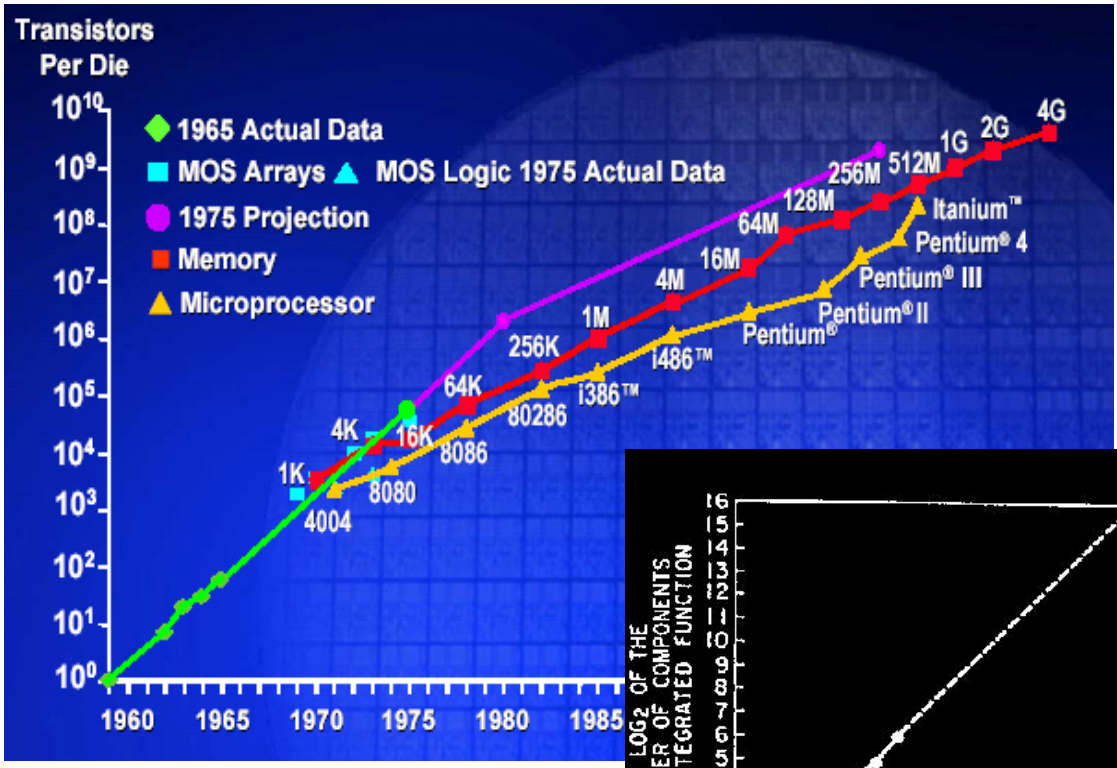
"Cramming More Components onto Integrated Circuits"

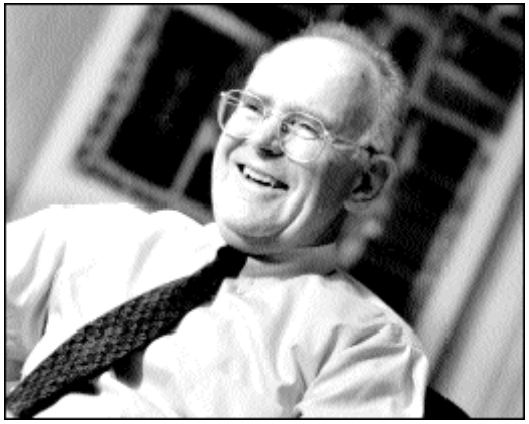
– Gordon Moore, Electronics, 1965

on transistors / cost-effective integrated circuit double every N months ($12 \leq N \leq 24$)

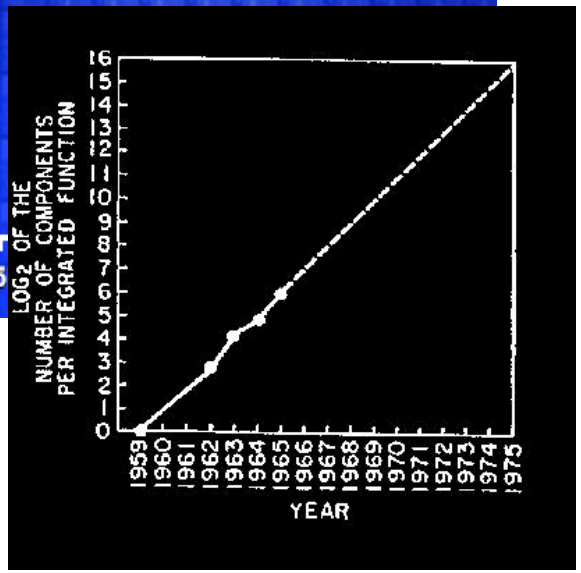
Moore's Law: 2X transistors / "year"







N seems to be rising - now 30?



"Cramming More Components onto Integrated Circuits"

– Gordon Moore, Electronics, 1965

on transistors / cost-effective integrated circuit double every N months ($12 \leq N \leq 24$)

The essence of Moore's Law:

scale each dimension by ~1/√2 = ~0.71

each scaling halves the area of a fixed design

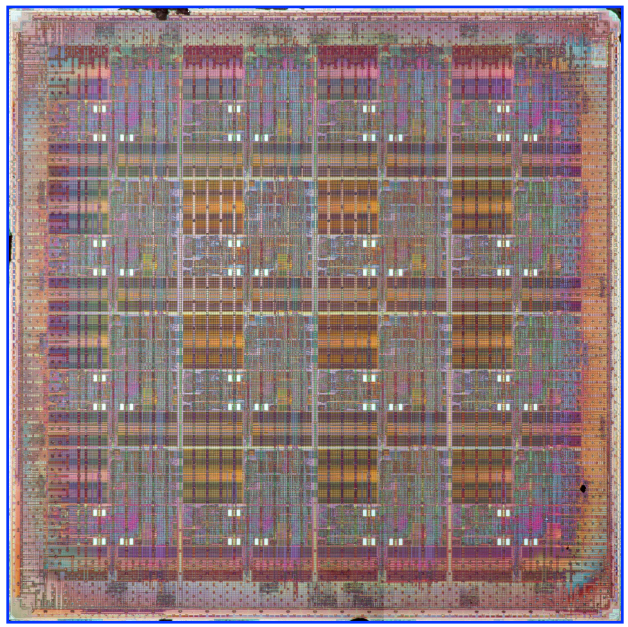
× .72 =

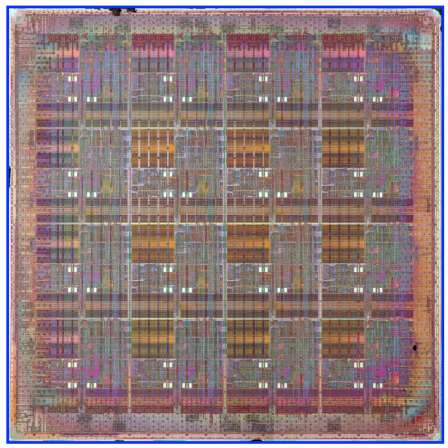
× .69 =

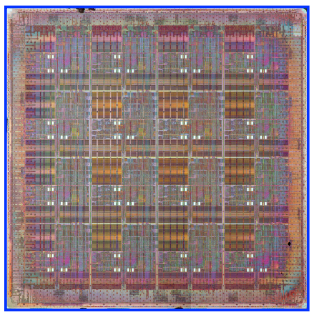
18.2 mm

13.1 mm

9.04 mm

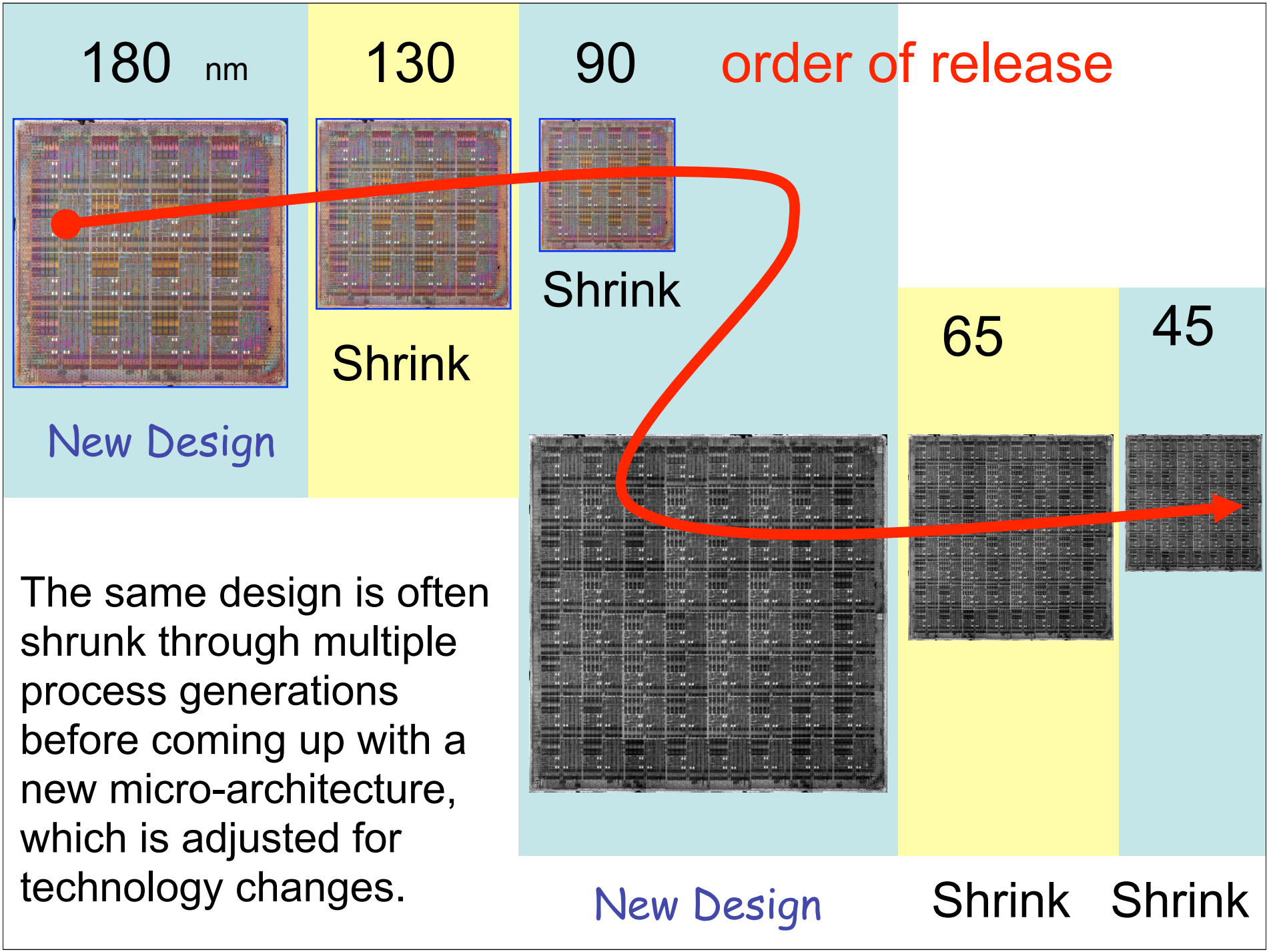




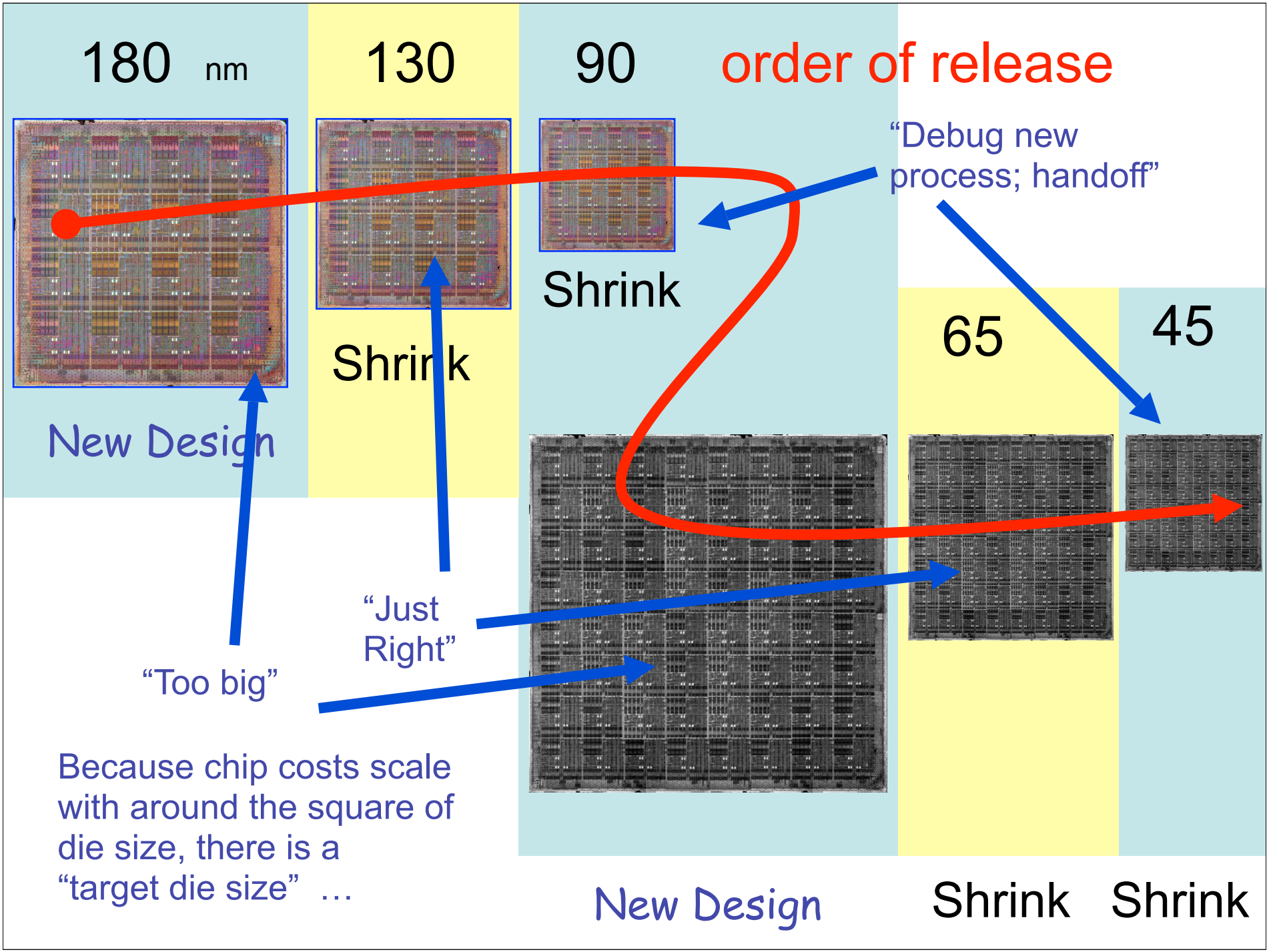


"Process" or "Lithography"
or "Litho" Generation
smallest wire pitch = ~ 2-3x litho

180 nm → 130 nm → 90 nm

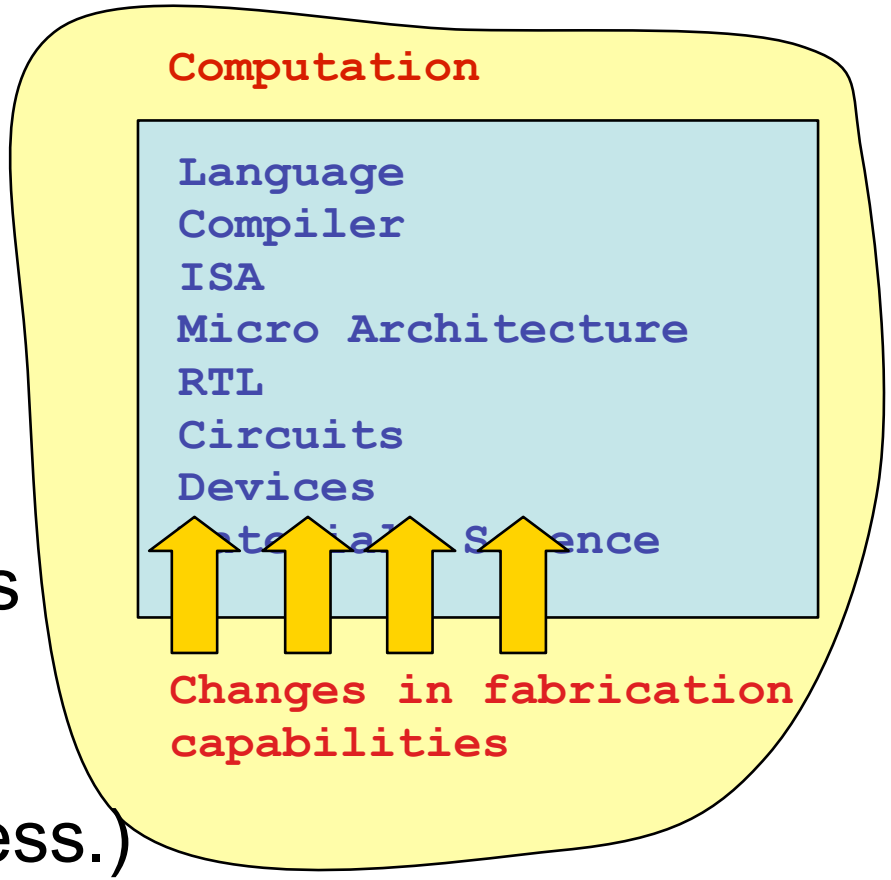


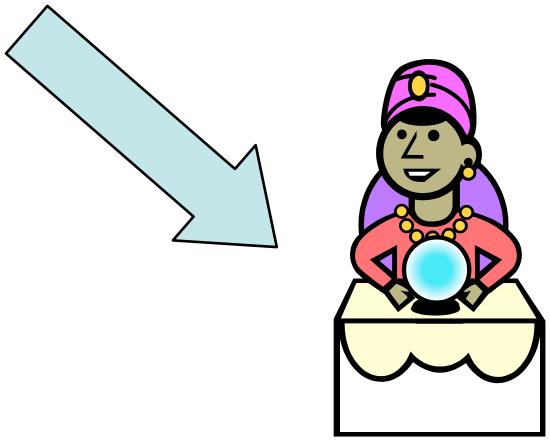
The same design is often shrunk through multiple process generations before coming up with a new micro-architecture, which is adjusted for technology changes.


180 nm
130
90
order of release
"Debug new process; handoff"
Shrink
Shrink
New Design
65
45
"Just Right"
"Too big"
Because chip costs scale with around the square of die size, there is a "target die size" …
New Design
Shrink
Shrink

Tech Trends

Since technology
change is such a
big influence in architecture,
and because it takes 3-6 years
to create a totally new design,
we try to predict & exploit it
(with varying degrees of success.)

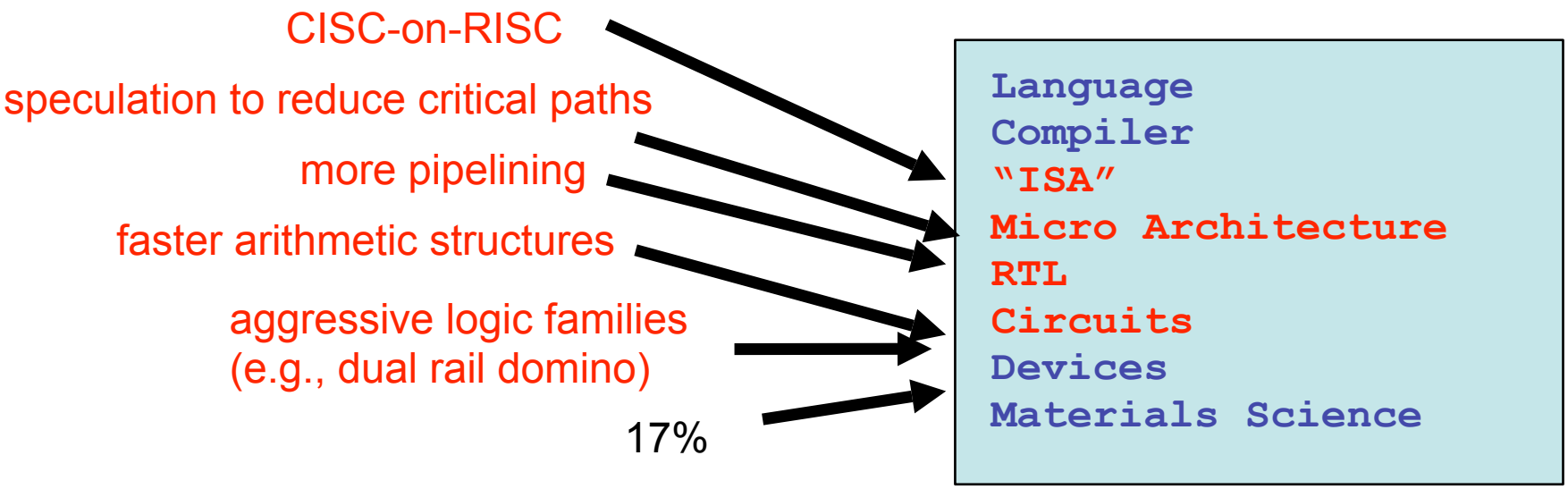




Transistor Frequency Scaling

Transistor (not processor) frequency scales approximately linearly with feature size.(e.g., 1.4x / generation), est. 17%/year

So where did the remainder of the
39% per year and 58% per year come from?



Transistor Frequency Scaling

Transistor (not processor) frequency scales approximately linearly with feature size. (e.g., 1.4x / generation), est. 17%/ year

So where did the remainder of the 39% per year and 58% per year come from?

the power wall - has reduced ability of these things to further improve frequency

CISC-on-RISC

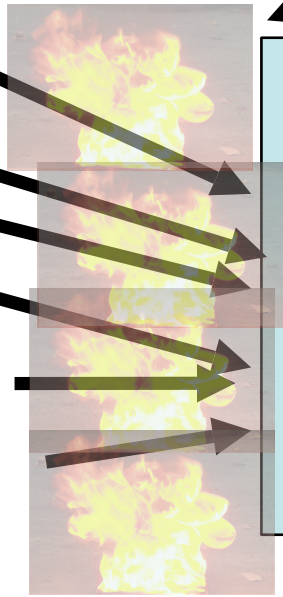
speculation to reduce critical paths

more pipelining

faster arithmetic structures

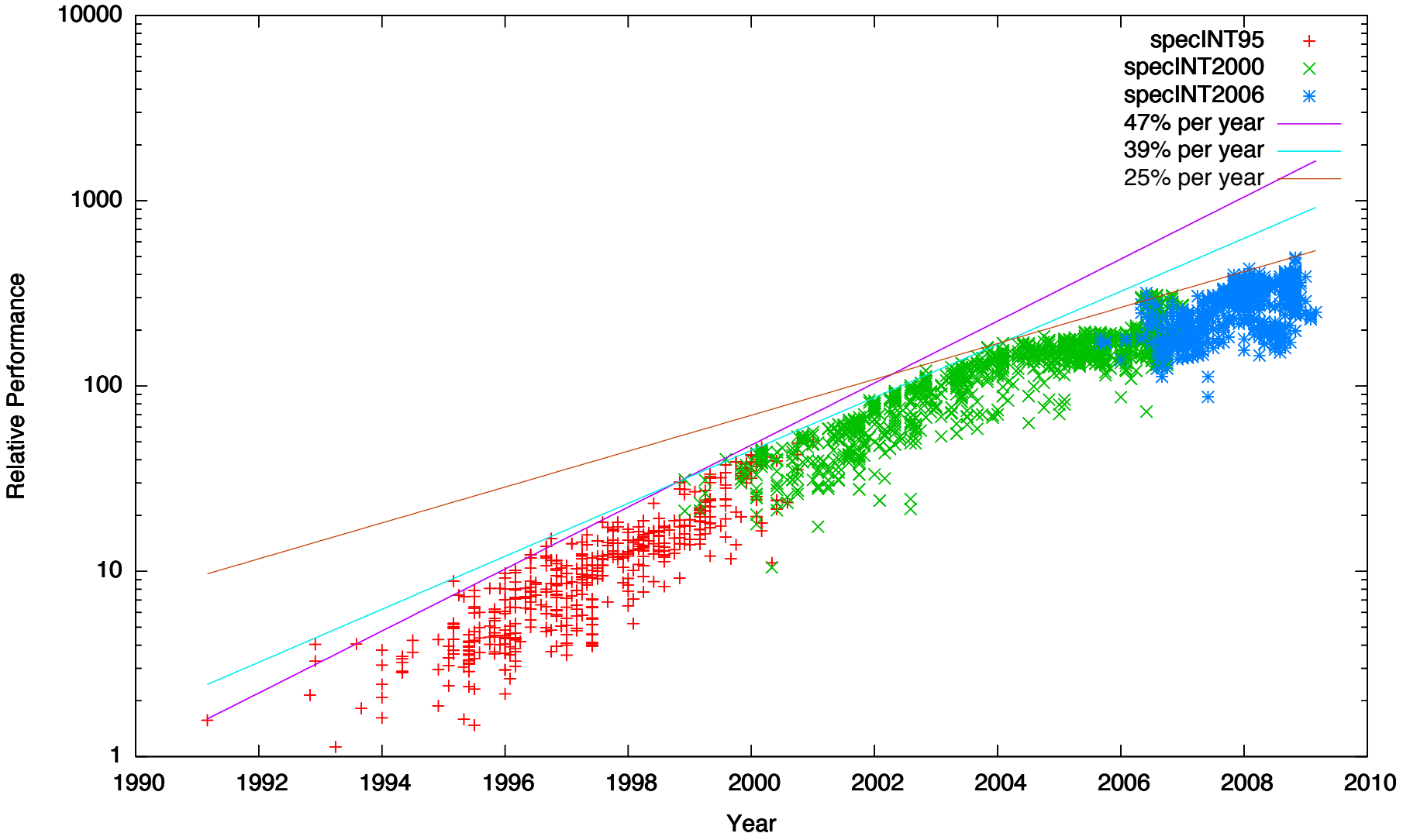
aggressive logic families (e.g., dual rail domino)

17%

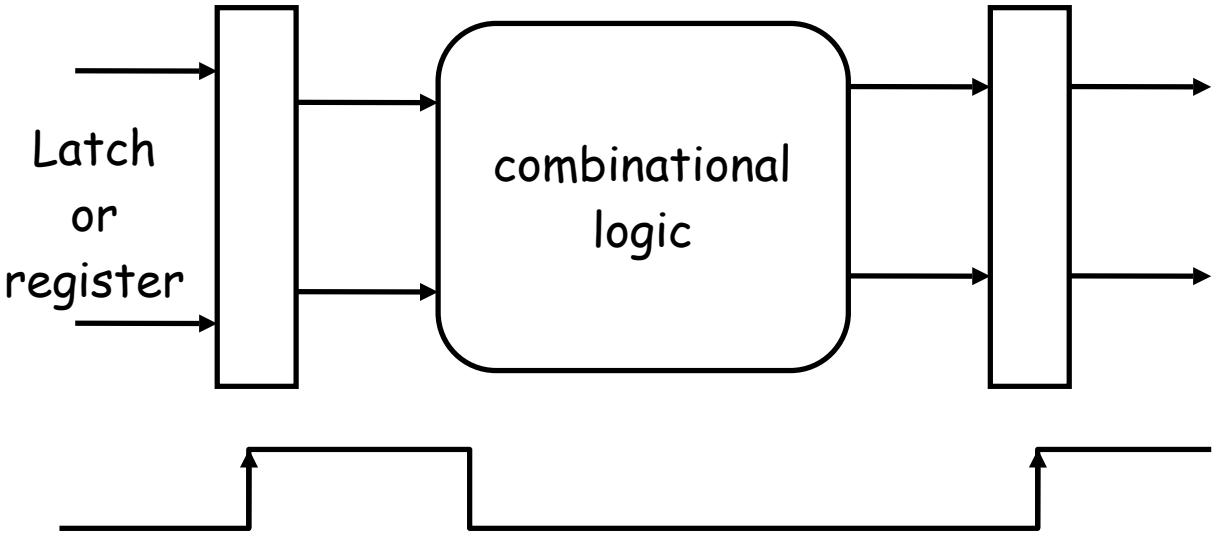


Language
Compiler
"ISA"
Micro Architecture
RTL
Circuits
Devices
Materials Science

Computer Performance

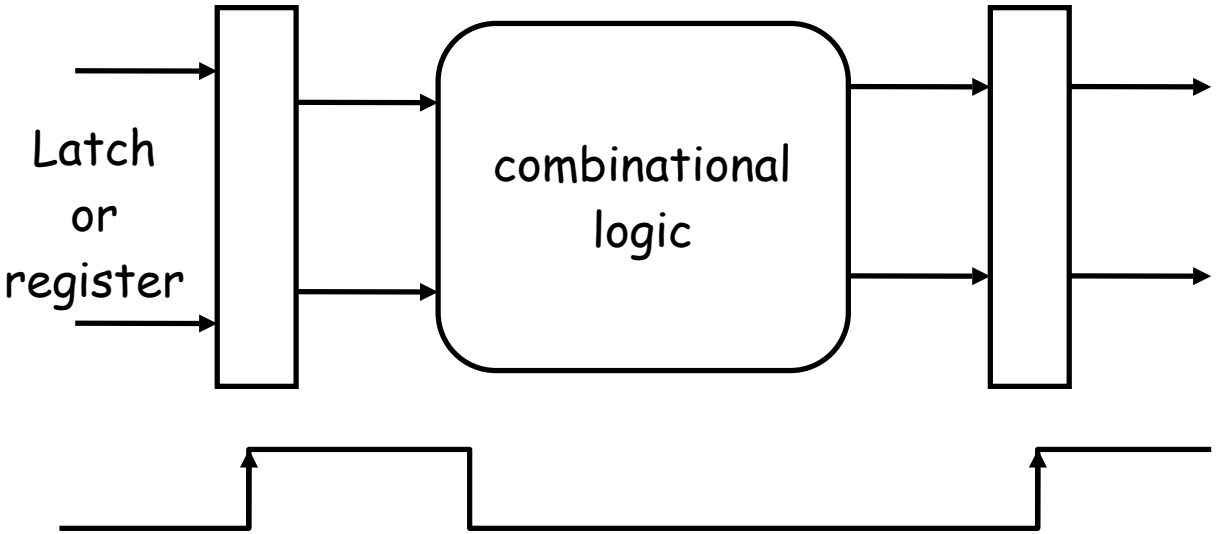


What determines the cycle time today?



- Old days: 10-16 levels of gates
- Today: gate delays, clock overhead, wire delays, + POWER

Processor Frequency and Power



$P = V^2FCa$

V = Processor voltage

F = frequency

C = Output load

a = Activity factor

- Lower frequency
  - Less overhead
  - Linear power savings
  - Potentially lower voltage -> quadratic power savings
  - More layers of logic per clock
  - Performance impact?

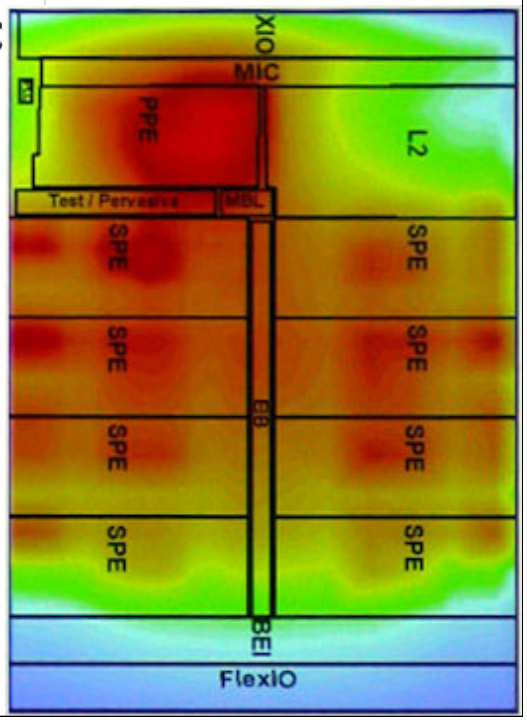
Processor Frequency and Power

Vdd (Volts)	2	2.2	2.4	2.6	2.8	3	3.2	3.4	3.6	3.8	4	4.2	4.4	4.6	4.8	5	5.2
1.3	48C 4W	49C 4W	50C 5W	50C 6W	51C 6W	52C 7W	53C 7W	54C 7W	55C 8W	56C 8W	57C 9W	58C 9W	59C 10W	60C 10W	61C 10W	63C 11W	61C
1.2	39C 2W	39C 3W	40C 3W	41C 4W	42C 4W	42C 4W	43C 5W	44C 5W	45C 5W	45C 5W	46C 6W	47C 6W	47C 7W	48C	49C		
1.1	32C 2W	33C 2W	33C 3W	35C 3W	35C 3W	36C 3W	36C 4W	37C 4W	37C 4W	38C 4W	38C 4W	39C	39C				
1	28C 2W	28C 2W	29C 2W	29C 2W	30C 2W	30C 3W	30C 3W	31C 3W	31C 3W	31C 3W	32C						
0.9	25C 1W	26C 1W	26C 1W	26C 2W	27C 2W	27C 2W	27C	Failed									

Cell Processor SPE
Top Speed: 5.2 GHz
Shipping speed: 3.2 GHz

2 GHz lost due to power constraints

source: ISSCC 2005, p. 135

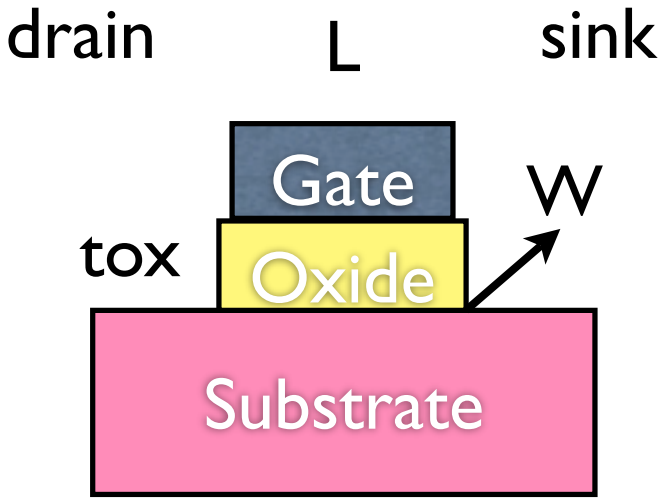


More on Scaling

- Seminal paper on scaling is Dennard et. al. “Design of ion-implanted MOSFET's with very small physical dimensions”, 1974
- Lays out how to build truly scalable transistors.

Dennardian Scaling

- Given a scaling factor *S*.

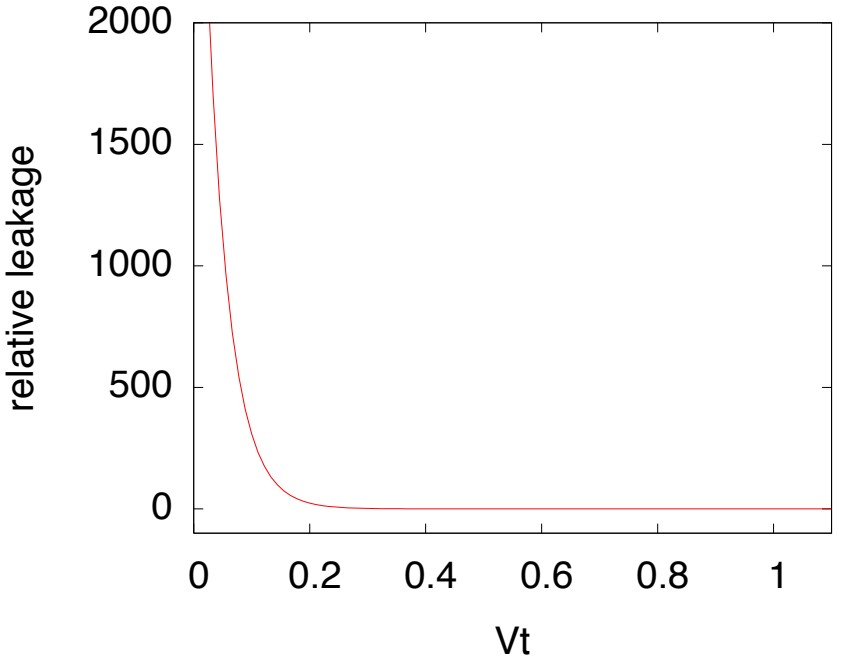


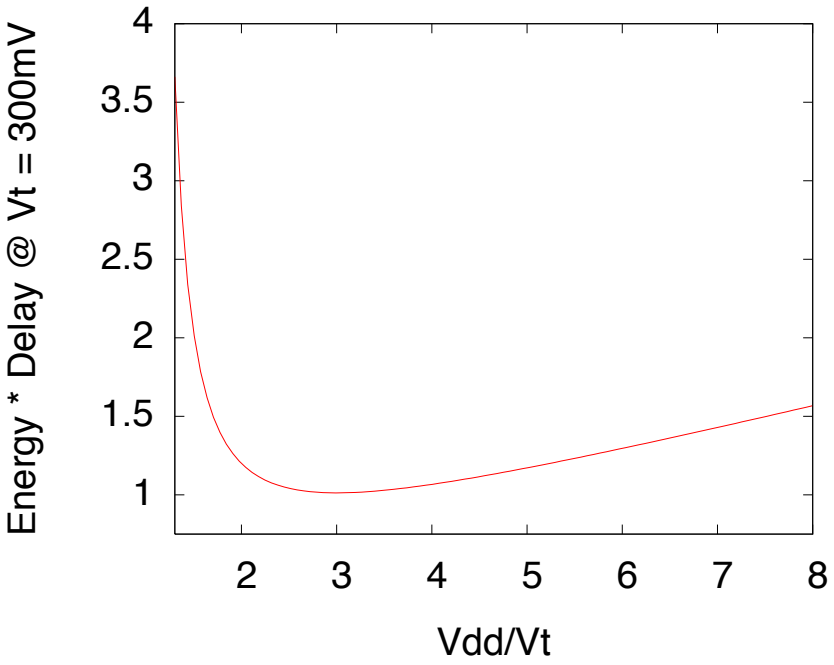
Parameter	Relation	Classical Scaling
power budget		1
chip size		1
V_{dd} (supply voltage)		$1/S$
V_t (threshold voltage)	$1/S$	$1/S$
t_{ox} (oxide thickness)		$1/S$
W, L (transistor dimensions)		$1/S$
C_{gate} (gate capacitance)	WL/t_{ox}	$1/S$
I_{sat} (saturation current)	WV_{dd}/t_{ox}	$1/S$
F (device frequency)	$I_{sat}/(C_{gate}V_{dd})$	S
D (device/area)	$1/(WL)$	S^2
p (device power)	$I_{sat}V_{dd}$	$1/S^2$
P (full die, full frequency power)	$D \times p$	1
U (utilization at fixed power)	$1/P$	1

Per-transistor power scales with density!

Dennardian Breakdown

- The problem with leakage





Dennardian Breakdown

Parameter	Relation	Classical Scaling	Leakage Limited
power budget		1	1
chip size		1	1
V_{dd} (supply voltage)		$1/S$	1
V_t (threshold voltage)	$1/S$	$1/S$	1
t_{ox} (oxide thickness)		$1/S$	$1/S$
W, L (transistor dimensions)		$1/S$	$1/S$
C_{gate} (gate capacitance)	WL/t_{ox}	$1/S$	$1/S$
I_{sat} (saturation current)	WV_{dd}/t_{ox}	$1/S$	1
F (device frequency)	$I_{sat}/(C_{gate}V_{dd})$	S	S
D (device/area)	$1/(WL)$	S^2	S^2
p (device power)	$I_{sat}V_{dd}$	$1/S^2$	1
P (full die, full frequency power)	$D \times p$	1	S^2
U (utilization at fixed power)	$1/P$	1	$1/S^2$

Per-transistor power is constant!

Dennardian Breakdown

Processor numberΔ	L2 cache	L3 cache	Clock speed	Front side bus	System type	Power	Number of cores
45-nm technology							
X7460	9MB	16MB	2.66 GHz	1066 MHz	MP	130W	6
L7455	9MB	12MB	2.13 GHz	1066 MHz	MP	65W	6
L7445	6MB	12MB	2.13 GHz	1066 MHz	MP	50W	4
E7450	9MB	12MB	2.40 GHz	1066 MHz	MP	90W	6
E7440	6MB	16MB	2.40 GHz	1066 MHz	MP	90W	4
E7430	6MB	12MB	2.13 GHz	1066 MHz	MP	90W	4
E7420	6MB	8MB	2.13 GHz	1066 MHz	MP	90W	4
65-nm technology							
X7350	8MB	0	2.93 GHz	1066 MHz	MP	130W	4
L7345	8MB	0	1.86 GHz	1066 MHz	MP	50W	4
E7340	8MB	0	2.40 GHz	1066 MHz	MP	80W	4
E7330	6MB	0	2.40 GHz	1066 MHz	MP	80W	4
E7320	4MB	0	2.13 GHz	1066 MHz	MP	80W	4
E7310	4MB	0	1.60 GHz	1066 MHz	MP	80W	4
E7220	8MB	0	2.93 GHz	1066 MHz	MP	80W	2
E7210	8MB	0	2.40 GHz	1066 MHz	MP	80W	2

Final Thoughts on Moore's Law

- Moore's Law is a *conspiracy*

  Webster conspiracy:

  *- **2 :** to act in harmony toward a common end*

The chip (semiconductor) industry consists of many players – equipment manufacturers (e.g. lithography, mask making equipment), chip makers, computer aided design (CAD) companies, and end-sellers. It more or less runs in lock step. No one company can go too far ahead in process generations without the others.

In fact – they all plan together what to shoot for according to a schedule over the next 15 years!

International Technology Roadmap for Semiconductors (ITRS)

Future Target

Yellow: some research will get us there

Red: we have no idea

Year of Production	*2005*	*2006*	*2007*	*2008*	*2009*	*2010*	*2011*	*2012*	*2013*
DRAM ½ Pitch (nm) (contacted)	*80*	*70*	*65*	*57*	*50*	*45*	*40*	*36*	*32*
MPU/ASIC Metal 1 (M1) ½ Pitch (nm)(contacted)	*90*	*78*	*68*	*59*	*52*	*45*	*40*	*36*	*32*
MPU Physical Gate Length (nm)	*32*	*28*	*25*	*22*	*20*	*18*	*16*	*14*	*13*
R_{sd}: Effective Parasitic series source/drain resistance [12]									
Planar Bulk (Ω-µm)	180	170	140	140	120	105	80	70	
UTB FD (Ω-µm)				155	140	125	110	90	75
DG (Ω-µm)							110	100	90
$C_{g,ideal}$: Ideal NMOS Device Gate Capacitance [13]									
Extended Planar Bulk (F/µm)	5.73E-16	5.25E-16	4.69E-16	6.37E-16	6.72E-16	6.78E-16	7.39E-16	6.41E-16	
UTB FD (F/µm)				5.84E-16	5.75E-16	5.65E-16	5.52E-16	5.37E-16	4.98E-16
DG (F/µm)							4.60E-16	4.39E-16	4.48E-16
$C_{g,total}$: Total gate capacitance for calculation of CV/I [14]									
Extended Planar Bulk (F/µm)	8.13E-16	7.65E-16	6.99E-16	8.47E-16	8.42E-16	8.28E-16	8.59E-16	7.51E-16	
UTB FD (F/µm)				8.04E-16	7.55E-16	7.35E-16	6.92E-16	6.67E-16	6.18E-16
DG (F/µm)							6.50E-16	6.29E-16	5.28E-16
$\tau = CV/I$: NMOSFET intrinsic delay (ps) [15]	0.870	0.740	0.640	0.540	0.460	0.400	0.340	0.290	0.250
$1/\tau$: NMOSFET intrinsic switching speed (GHz) [16]	1149	1351	1563	1852	2174	2500	2941	3448	4000

Current

× ~13.5 = 1 “FO4” (gate delay) = 81 GHz

http://www.itrs.net/Links/2005ITRS/Home2005.htm