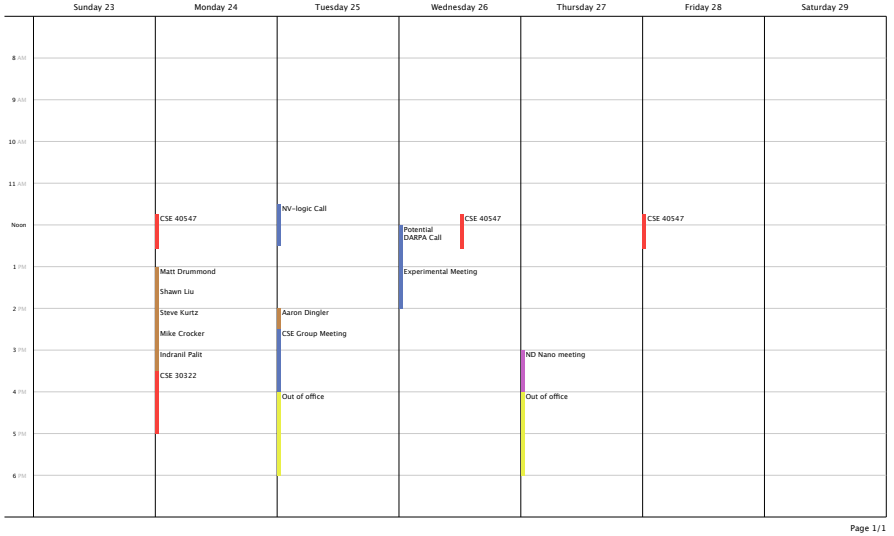
Lecture 01
Introduction to CSE 40547 / 60547

Rescheduling?

- Could we meet 2 days a week instead of 3?



A history of computing

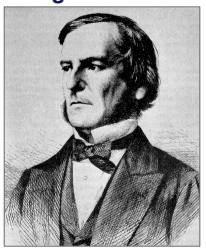
- Today:
  - I'd like to start by explaining "how we got to where we are" when we look at modern information processing systems
- My Focus:
  - Devices
  - How devices are organized into an architecture
  - How a system-level architecture might address an application level task

With emerging technologies with sub-100 nm feature sizes, it's important to consider devices, architectures, and applications simultaneously – which we will do too!

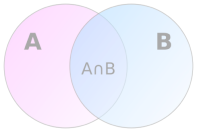
(Things didn't use to be this way)

Four People

George Boole



Proposes a complete system for algebraic logical operations



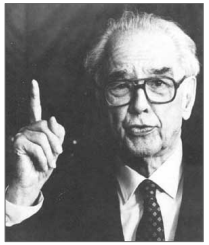
Claude Shannon

Proves that circuits based on electromechanical relays could be used to solve Boolean algebra problems

Konrad Zuse

- Use binary numbers to encode information
- Binary numbers can be represented with on/off state of current switch

John Atanasoff



Suggests correct mode of computation is to use electronic binary digits

Binary Math

- How do computers add numbers?
  - (binary arithmetic ... e.g. all 1s and 0s)
  - What number (in decimal) is 110010 in binary?
    - $1x2^5 + 1x2^4 + 0x2^3 + 0x2^2 + 1x2^1 + 0x2^0$
    - 32 + 16 + 0 + 0 + 2 + 0 = 50
  - What is 110010 + 000011 ?

```
    1
  1 1 0 0 1 0
+ 0 0 0 0 1 1
-------------
  1 1 0 1 0 1
```

    - $1x2^5 + 1x2^4 + 0x2^3 + 1x2^2 + 0x2^1 + 1x2^0$
    - 32 + 16 + 0 + 4 + 0 + 1 = 53
- (multiplication works just like decimal multiplication)
  - e.g. 0x0 = 0 0x1 = 0 1x0 = 0 1x1 = 1

Math via Boolean logic

A lot like multiplication

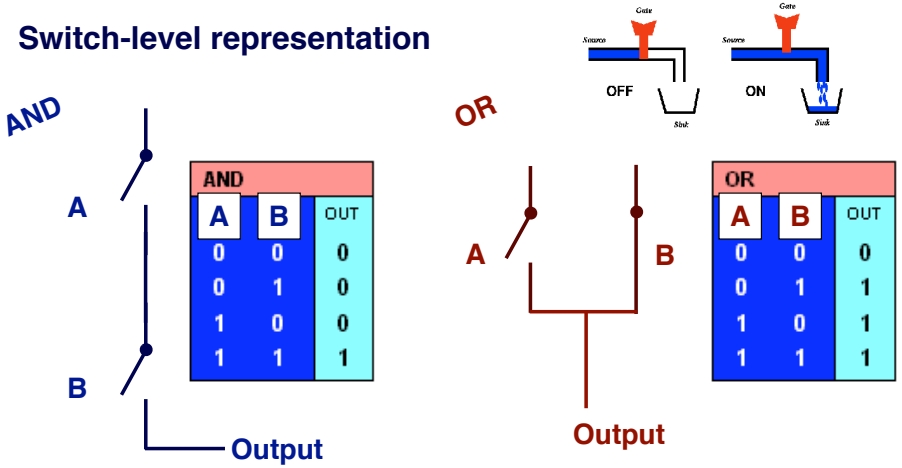
From Computer Desktop Encyclopedia
© 1998 The Computer Language Co. Inc.

With AND, OR, NOT, can implement any function.

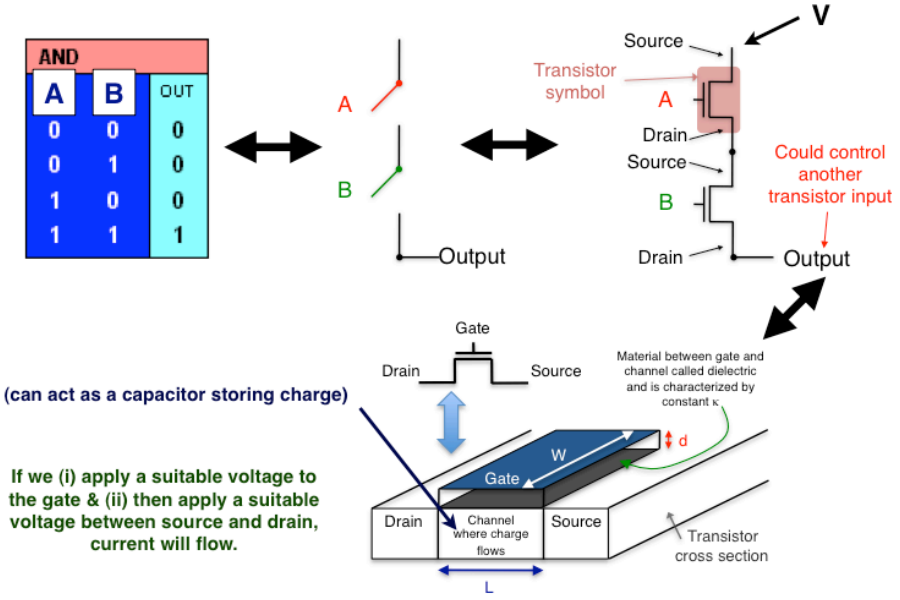
NOT	
IN	OUT
0	1
1	0

Can perform AND, OR ops with switches

Switch-level representation

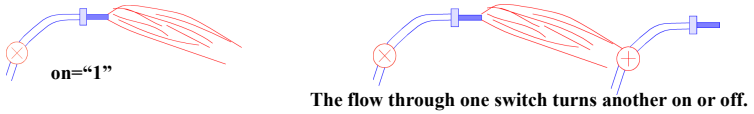


Switches now transistors on IC

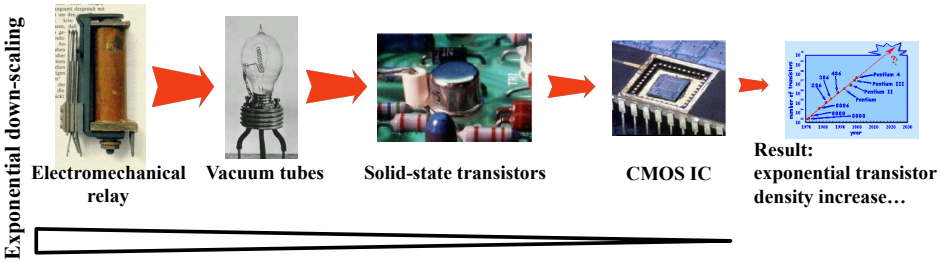


Historically, this idea seems to have worked out rather well…

- Long since predominant mode of information processing
  - Represent binary digits as on/off state of a current switch

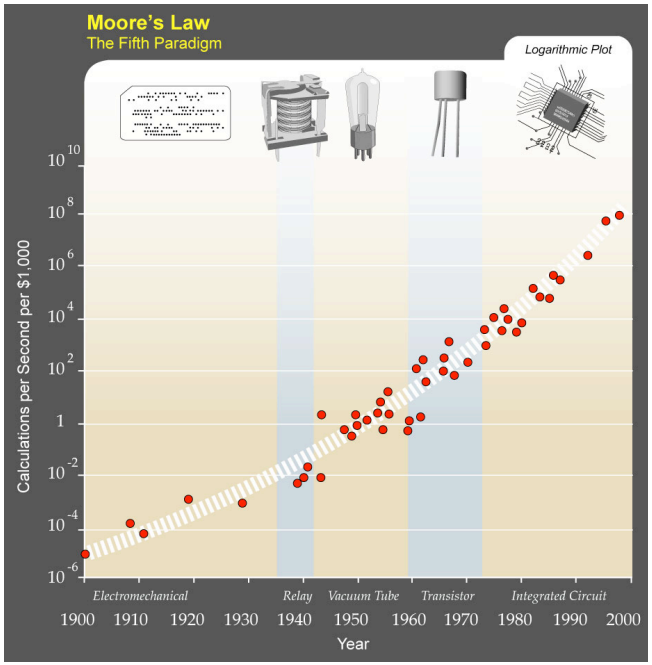


The flow through one switch turns another on or off.

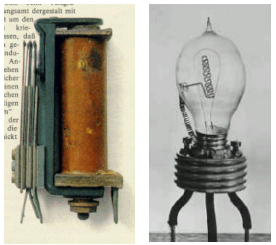


Put another way…

…for what it's worth, that's 14 orders of magnitude – *enabled by technology scaling*



Let's start with relay and vacuum tube machines … *ideas that evolved from this work still persist and influence work today*



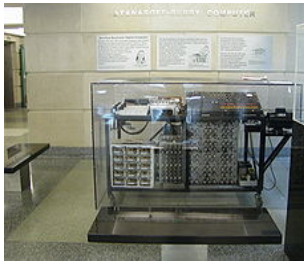
Acknowledgements

- These slides contain material developed and copyright by:
  - Arvind (MIT)
  - Krste Asanovic (MIT/UCB)
  - Joel Emer (Intel/MIT)
  - James Hoe (CMU)
  - John Kubiatowicz (UCB)
  - David Patterson (UCB)

- MIT material derived from course 6.823
- UCB material derived from course CS252

Linear Equation Solver

John Atanasoff, Iowa State University



1930's:
- Atanasoff built the Linear Equation Solver.
- It had 300 tubes!
- Special-purpose binary digital calculator
- Dynamic RAM (stored values on refreshed capacitors)

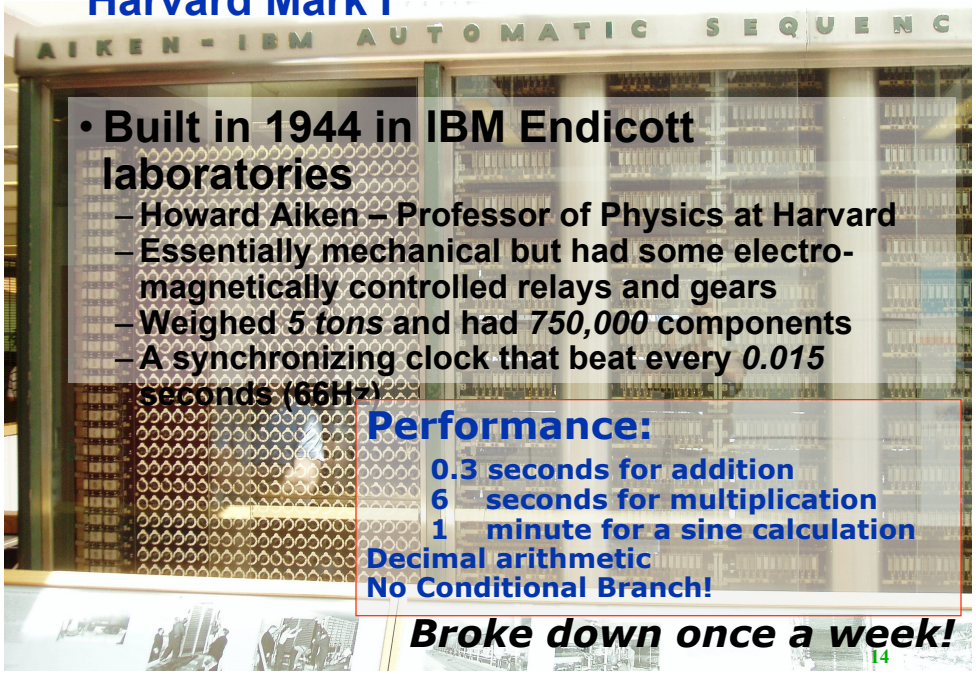
Application:
- Linear and Integral differential equations

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

Technology:
- Tubes and Electromechanical relays

Atanasoff decided that the correct mode of computation was using electronic binary digits.

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 (66Hz)**

Performance:
0.3 seconds for addition
6 seconds for multiplication
1 minute for a sine calculation
Decimal arithmetic
No Conditional Branch!

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

WW-2 Effort

- Not very reliable!

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 instructio
- Eckert, Mauchly, John von Neumann an
  designed EDVAC (1944) to solve this pr
  - Solution was the *stored program computer*

⇒ "*program can be manipulated as*

von Neumann's stored program model warrants a slightly longer discussion

Stored Programs (Part 1)

First Draft of a Report
on the EDVAC

by

John von Neumann

Contract No. W–670–ORD–4926

Between the

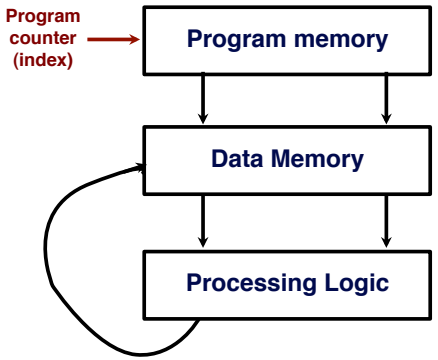
United States Army Ordnance Department

and the

University of Pennsylvania

Moore School of Electrical Engineering
University of Pennsylvania

June 30, 1945



Look familiar?

A hypothetical translation:

```
for i=0; i<5; i++ {
      a = (a*b) + c;
}
```

```
MULT temp,a,b  # temp ← a*b
MULT r1,r2,r3  # r1 ← r2*r3
```

```
ADD a,temp,c   # a ← temp+c
ADD r2,r1,r4   # r2 ← r1+r4
```

Can define codes for MULT and ADD
Assume MULT = 110011 & ADD = 001110

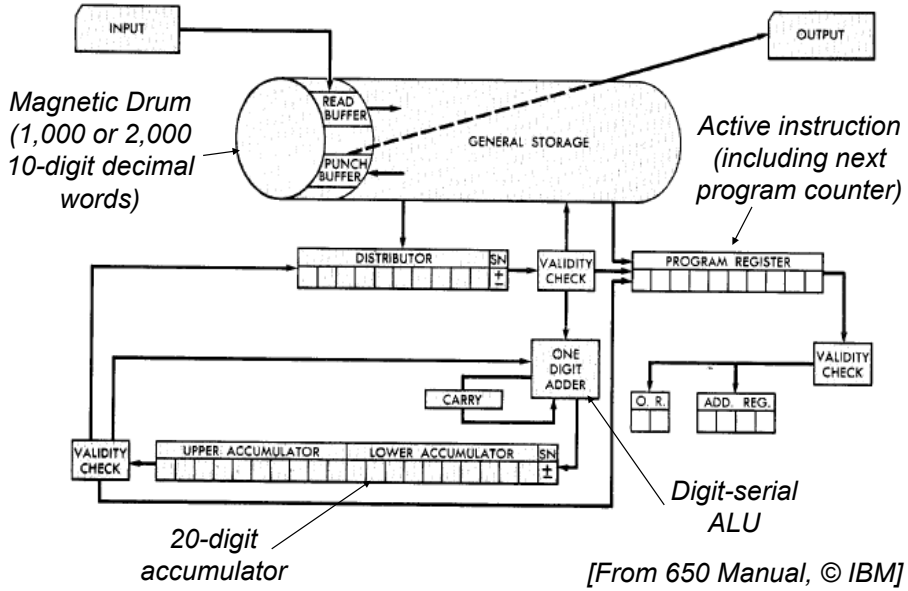
stored program becomes

PC	110011	000001	000010	000011
PC+1	001110	000010	000001	000100

This idea has staying power!

How we process information hasn't changed much since 1930s and 1940s

The IBM 650 (1953-4)



Magnetic Drum (1,000 or 2,000 10-digit decimal words)

Active instruction (including next program counter)

Digit-serial ALU

20-digit accumulator

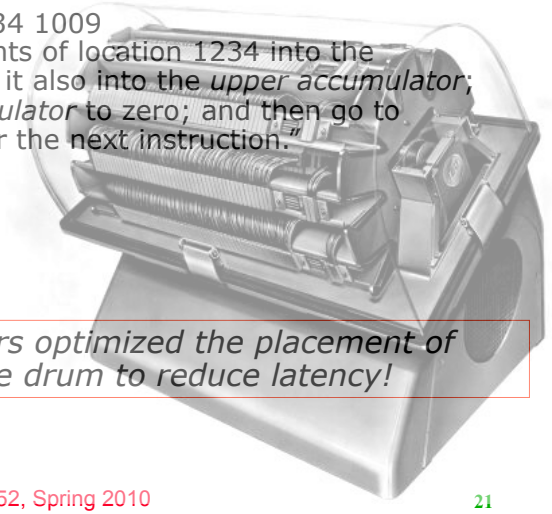
[From 650 Manual, © IBM]

Programmer's view of the IBM 650

A drum machine with 44 instructions

Instruction: 60 1234 1009

- "Load the contents of location 1234 into the *distribution*; put it also into the *upper accumulator*; set *lower accumulator* to zero; and then go to location 1009 for the next instruction."



Good programmers optimized the placement of instructions on the drum to reduce latency!

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

Stored program model persists with device miniaturization

IBM 360: *A General-Purpose Register (GPR) Machine* 1964

- 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
  - But no instruction contains a 24-bit address!
- Data Formats
  - 8-bit bytes, 16-bit half-words, 32-bit words, 64-bit double-words

The IBM 360 is why bytes are 8-bits long today!

IBM 360: Initial Implementations

	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	Conventional circuits

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

Milestone: The first true ISA designed as portable hardware-software interface!

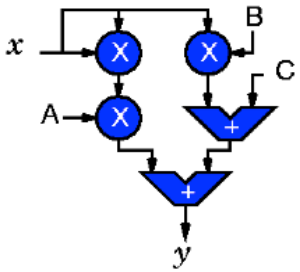
With minor modifications it still survives today!

One caveat to stored program model: Field Programmable Gate Arrays

What is Configurable Computing?

Spatially-programmed connection of processing elements

$$y = Ax^2 + Bx + C$$

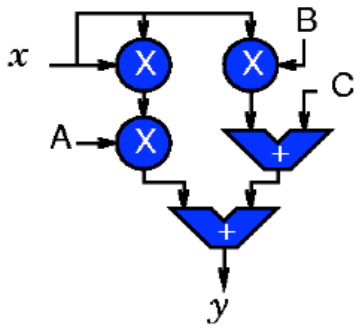


"Hardware" customized to specifics of problem.
- Direct map of problem specific dataflow, control.

Circuits "adapted" as problem requirements change.

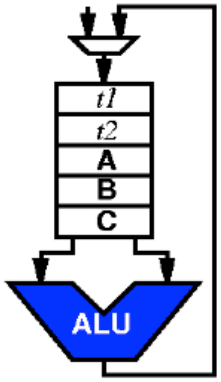
Spatial vs. Temporal Computing

Spatial

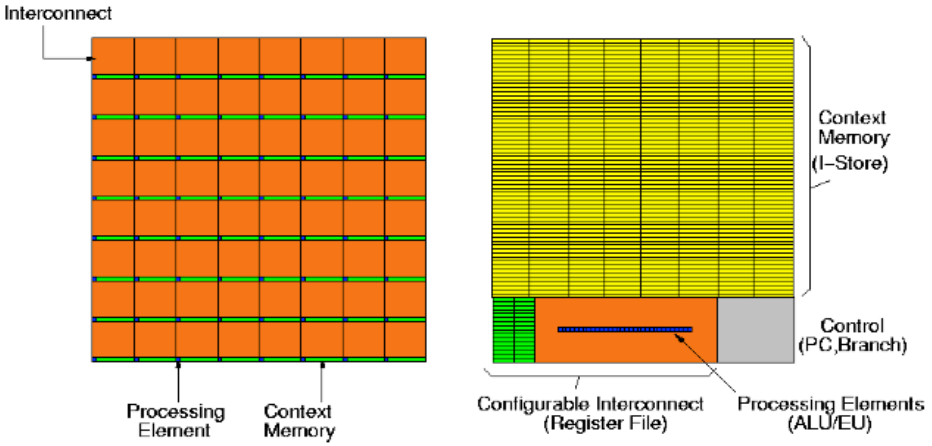


Temporal

$t1 \leftarrow x$
$t2 \leftarrow \mathbf{A} \times t1$
$t2 \leftarrow t2 + \mathbf{B}$
$t2 \leftarrow \mathbf{t2} \times t1$
$y \leftarrow \mathbf{t2} + \mathbf{C}$

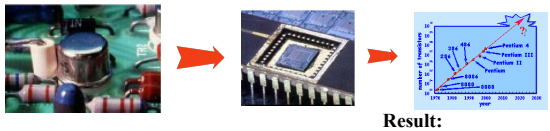


Processor vs. FPGA Area



Remember this: We'll revisit this idea later in the semester!

Stored program model – *in face of transistor scaling on integrated circuits* – not without challenges



Solid-state transistors | CMOS IC | Result: exponential transistor density increase…

Challenge #1: Memory is still (relatively) slow!

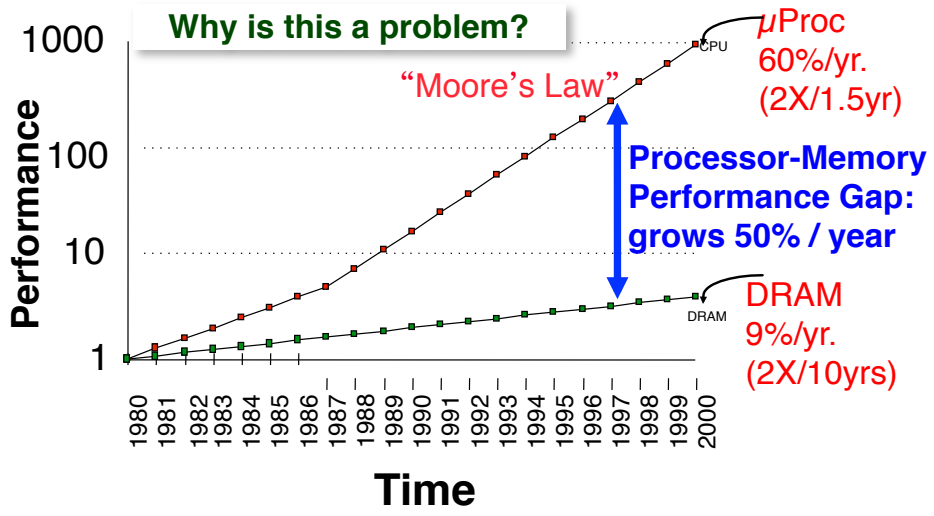
Remember…

Computers in mid 50's

- Memory access time was 10 to 50 times slower than the processor cycle

  ⇒ Instruction execution time was totally dominated by the *memory reference time*.

Processor-DRAM Memory Gap (latency)



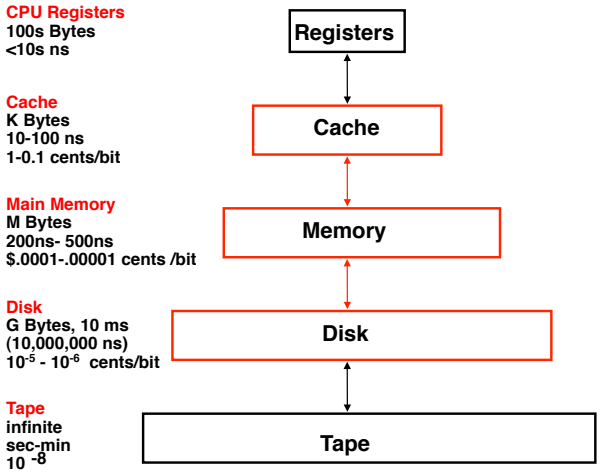
Solution: Memory Hierarchies
(The principle of locality…)

- …says that most programs don't access all code or data uniformly
  - i.e. in a loop, small subset of instructions might be executed over and over again…
  - …& a block of memory addresses might be accessed sequentially…

- This has lead to "memory hierarchies"
- Some important things to note:
  - Fast memory is expensive
  - Levels of memory usually smaller/faster than previous
  - Levels of memory usually "subset" one another
    - All the stuff in a higher level is in some level below it

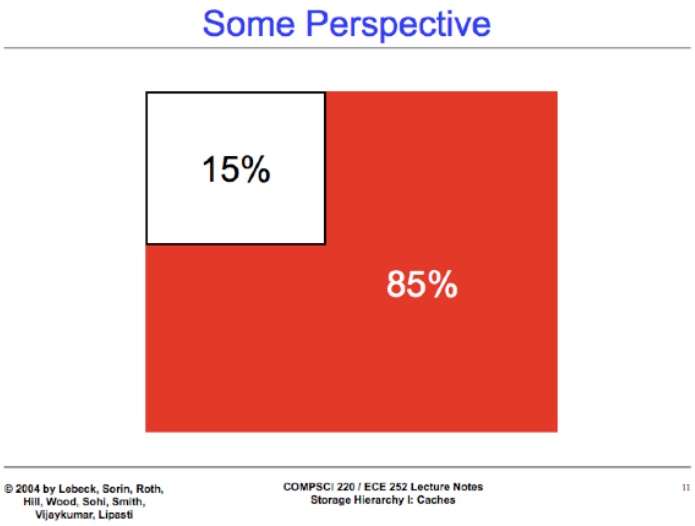
An example memory hierarchy

Data structure may not fit in fast memory



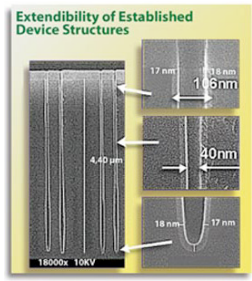
Question:

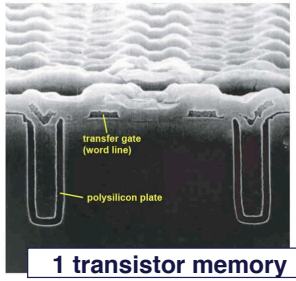
- How much of a chip is "memory"?
  - 10%
  - 25%
  - 50%
  - 75%
  - 85%



DRAM vs. SRAM: Different Technology Processes

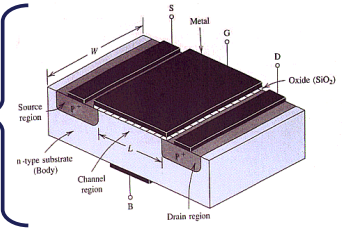
DRAM transistors are "deep trenches"

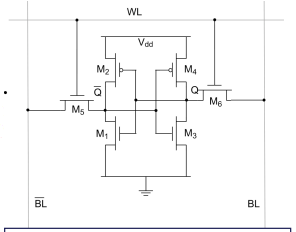




1 transistor memory

SRAM transistors made with logic process





6 transistor memory

http://web.sfc.keio.ac.jp/~rdv/keio/sfc/teaching/architecture/architecture-2008/6t-SRAM-cell.png
http://www.micromagazine.com/archive/02/06/0206MI22b.jpg
http://www.ieee.org/portal/site/sscs/menuitem.f07ee9e3b2a01d06bb9305765bac26c8/index.jsp?&pName=sscs_level1_article&TheCat=2171&path=sscs/08Winter&file=Sunami.xml

Challenge #2:
Chips have gotten hot!

Moore's Law

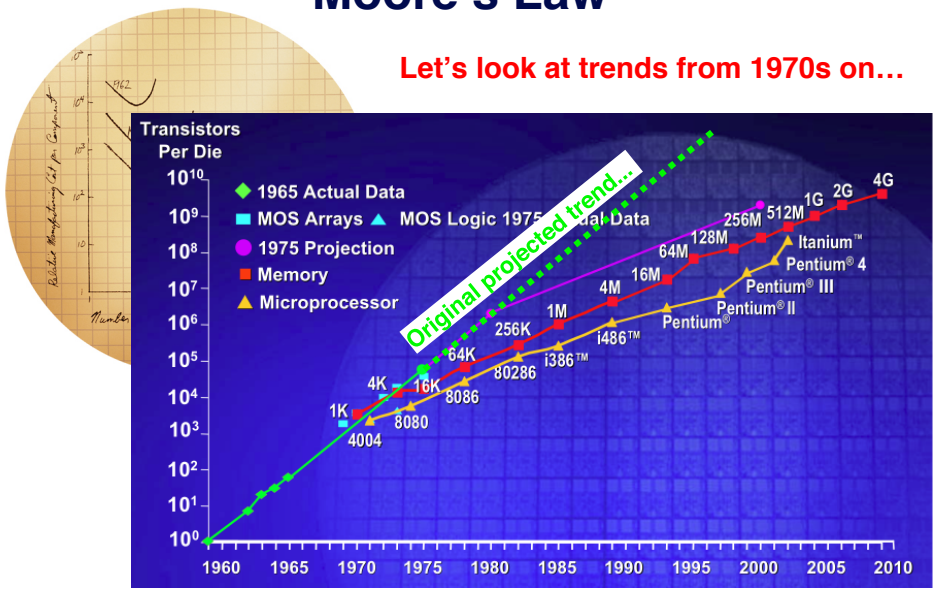
- "Cramming more components onto integrated circuits."

  - G.E. Moore, Electronics 1965

  - Observation: DRAM transistor density doubles annually
    - Became known as "Moore's Law"
    - Actually, a bit off:
      - Density doubles every 18 months (now more like 24)
      - (in 1965 they only had 4 data points!)
  - Corollaries:
    - Cost per transistor halves annually (18 months)
    - Power per transistor decreases with scaling
    - Speed increases with scaling
      - Of course, it depends on how small you try to make things
        - » (I.e. no exponential lasts forever)

Moore's Law

Let's look at trends from 1970s on…

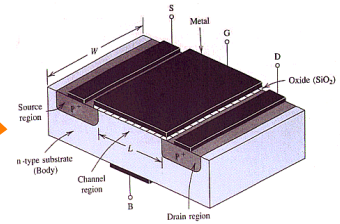


A bit on device performance...

- One way to think about switching time:
  - Charge is carried by electrons
  - Time for charge to cross channel = length/speed
    - $= L^2/(mV_{ds})$

Thus, to make a device faster, we want to either increase V_{ds} or decrease feature sizes (i.e. L)

- What about power (i.e. heat)?
  - Dynamic power is: $P_{dyn} = C_L V_{dd}^2 f_{0\text{-}1}$
    - $C_L = (e_{ox}WL)/d$
      - e_{ox} = dielectric, WL = parallel plate area, d = distance between gate and substrate



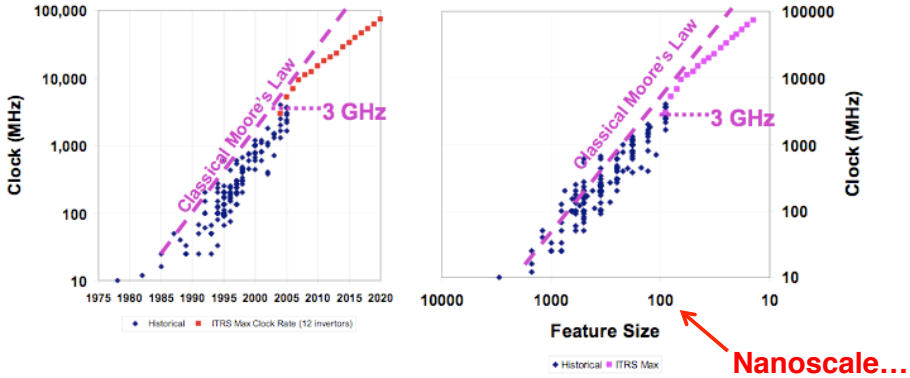
Summary of relationships

- (+) If V increases, speed (performance) increases
- (-) If V increases, power (heat) increases
- (+) If L decreases, speed (performance) increases
- (?) If L decreases, power (heat) does what?
  - P could improve because of lower C
  - P could increase because >> # of devices switch
  - P could increase because >> # of devices switch faster!

Need to carefully consider tradeoffs between speed and heat

A funny thing happened on the way to 45 nm

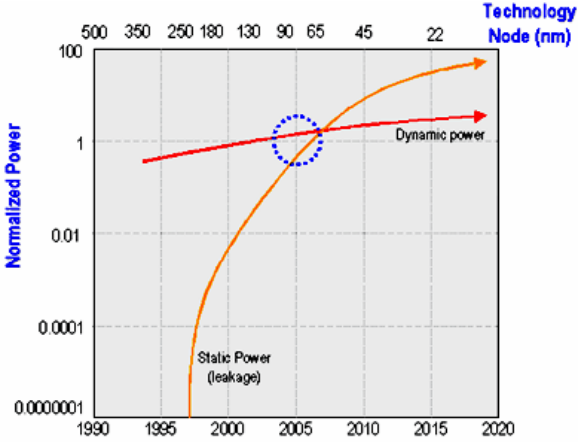
- Speed increases with scaling...



2005 projection was for 5.2 GHz - and we didn't make it in production. Further, we're still stuck at 3+ GHz in production.

A funny thing happened on the way to 45 nm

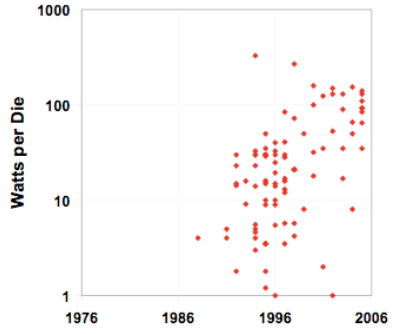
- Power decreases with scaling...



A funny thing happened on the way to 45 nm

- Speed increases with scaling...
- Power decreases with scaling...

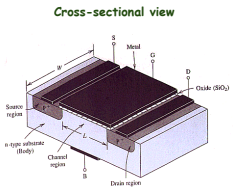
Why the clock flattening? POWER!!!!

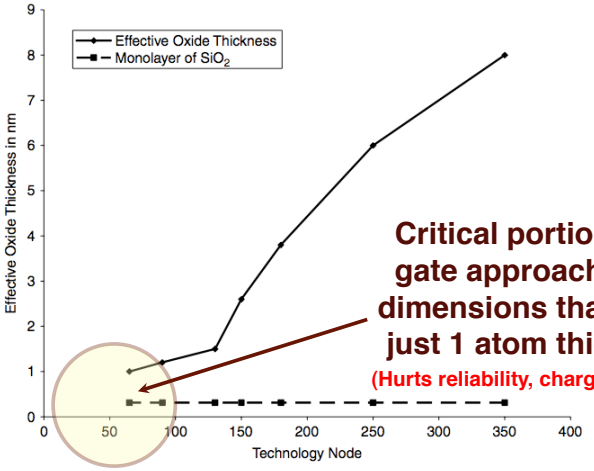


A funny thing happened on the way to 45 nm

- What about scaling...

One quick example:





Critical portion of gate approaching dimensions that are just 1 atom thick!!!

(Hurts reliability, charge leaks)

Materials innovations were – and still are – needed

One solution: new, high-κ dielectrics

$$C = \frac{\kappa \varepsilon_0 A}{t}$$

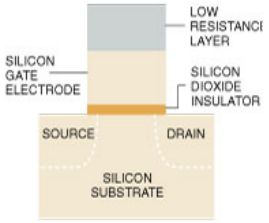
Increase thickness to reduce gate leakage

Increase κ to maintain capacitance

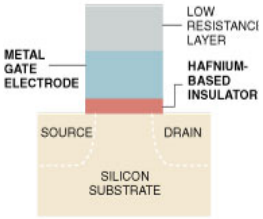
Small and Efficient

As microprocessor transistors become smaller, stopping undesired current leakage becomes more difficult. This leakage leads to shortened battery life. Intel's coming chips use a new insulation material to prevent this, reducing power consumption.

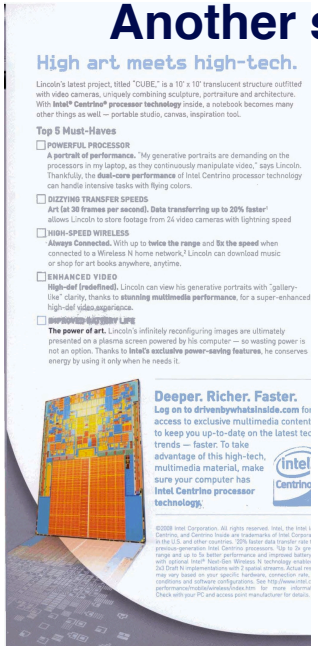
Current transistors use extremely thin silicon dioxide insulators, which lead to current leakage. Thickening them decreases this leakage but reduces the electric charge passing through, impeding performance.



New transistors use a hafnium-based insulator and a metal gate electrode. Hafnium provides stronger electrical coupling, so the insulator can be made thicker to reduce leakage without degrading the performance of the transistor.

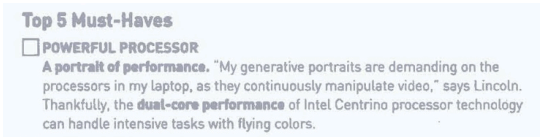


Another solution: parallelism



- Processor complexity is good enough
- Transistor sizes can still scale
- Slow processors down to manage power
- Get performance from...

Parallelism



(i.e. 1 processor, 1 ns clock cycle vs. 2 processors, 2 ns clock cycle)

Even solutions have limitations

More caching?
(What about "state bloat"?)

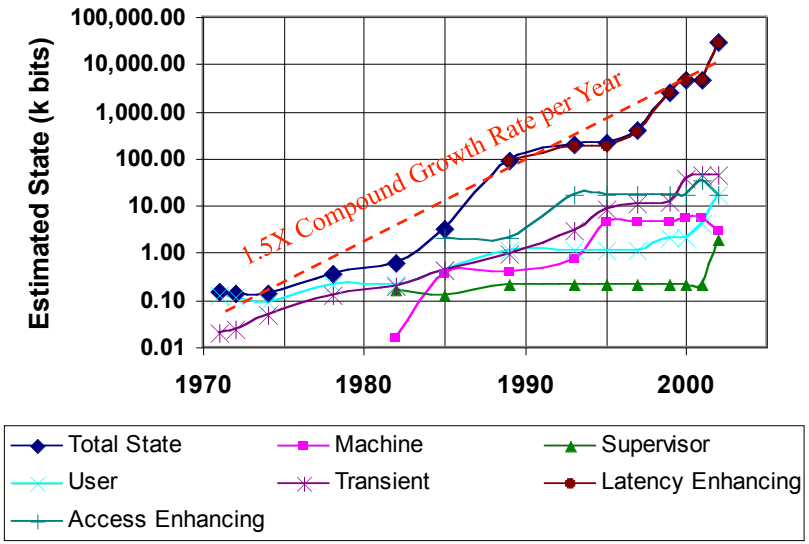
Impact of modern processing principles (Lots of "state")

- User:
  - state used for application execution
- Supervisor:
  - state used to manage user state
- Machine:
  - state that configures the system
- Transient:
  - state used during instruction execution
- Access-Enhancing:
  - state used to simplify translation of other state names
- Latency-Enhancing:
  - state used to reduce latency to other state values

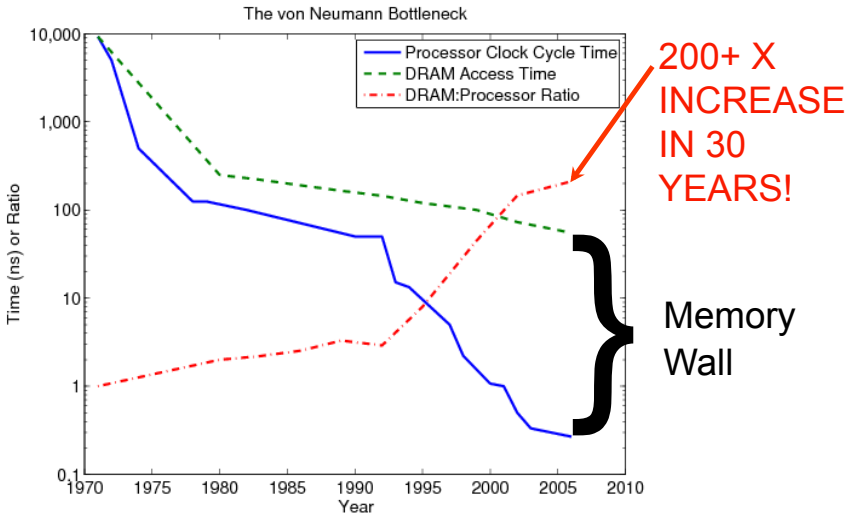
Lots of "state" – but how much is *directly* associated with a computation?

What if you want to add 2, 32-bit numbers together?

Impact of modern processing principles
(Total State vs. Time)



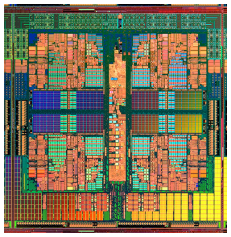
Impact of modern processing principles
(Why so much latency enhancing state?)



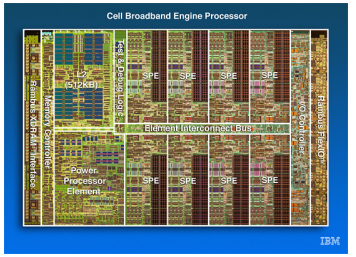
More cores?

This idea has been extended...

Quad core chips...



7, 8, and 9 core chips...



Practical problems must be addressed!

Advances in parallel programming are necessary!

stop?

Impediments to Parallel Performance

- ★ Contention for access to shared resources
  - i.e. multiple accesses to limited # of memory banks may dominate system scalability
- ★ Programming languages, environments, & methods:
  - Need simple semantics that can expose computational properties to be exploited by large-scale architectures
- ★ Algorithms
  - What if you write good code for a 4-core chip, and then get an 8-core chip?
- ★ Cache coherency
  - P1 writes, P2 can read
    - Protocols can enable $ coherency but add overhead

★ Overhead where no actual processing is done.

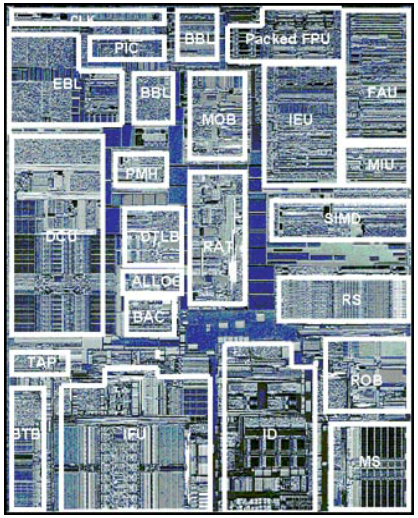
Impediments to Parallel Performance

- Latency ★
  - Is already a major source of performance degradation
  - Architecture charged with hiding local latency
    - (that's why we talked about registers & caches)
  - Hiding global latency is also task of programmer
    - (I.e. manual resource allocation)
- Today:
  - access to DRAM in 100s of CCs
  - round trip remote access in 1000s of CCs
  - multiple clock cycles to cross chip or to communicate from core-to-core
    - Not "free"

★ Overhead where no actual processing is done.

Pentium III Die Photo

Deterministic connections as needed.

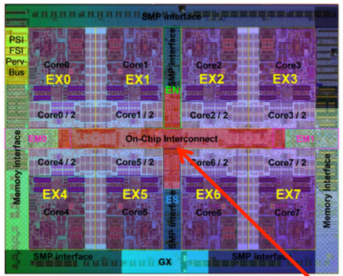


1st Pentium III, Katmai: 9.5 M transistors, 12.3 * 10.4 mm in 0.25-mi. **with 5 layers of aluminum**

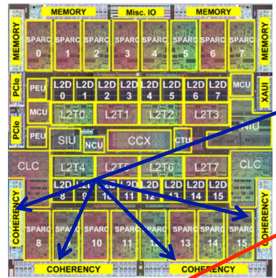
- **EBL/BBL - Bus logic, Front, Back**
- **MOB - Memory Order Buffer**
- **Packed FPU - MMX Fl. Pt. (SSE)**
- **IEU - Integer Execution Unit**
- **FAU - Fl. Pt. Arithmetic Unit**
- **MIU - Memory Interface Unit**
- **DCU - Data Cache Unit**
- **PMH - Page Miss Handler**
- **DTLB - Data TLB**
- **BAC - Branch Address Calculator**
- **RAT – Register Alias Table**
- **SIMD - Packed Fl. Pt.**
- **RS - Reservation Station**
- **BTB - Branch Target Buffer**
- **IFU - Instruction Fetch Unit (+I$)**
- **ID - Instruction Decode**
- **ROB - Reorder Buffer**
- **MS - Micro-instruction Sequencer**

Recent multi-core die photos

(Route packets, not wires?)



http://dx.doi.org/10.1109/ISSCC.2010.5434074

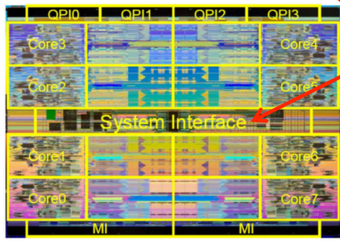


http://dx.doi.org/10.1109/ISSCC.2010.5434030

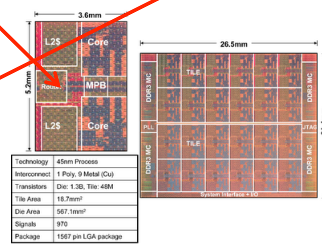
Likely to see HW support for parallel processor configurations:
- **Coherency**

+

- **On-chip IC NWs**



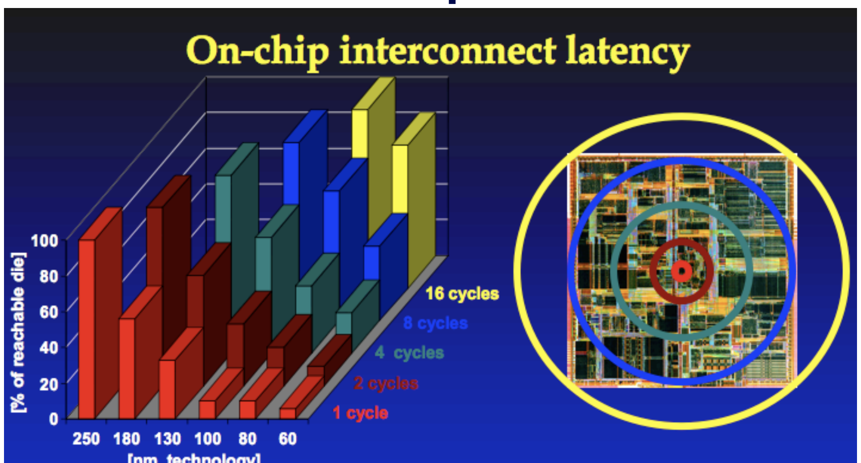
http://dx.doi.org/10.1109/ASSCC.2009.5357230



http://dx.doi.org/10.1109/ISSCC.2010.5434077

…takes advantage of 8 voltage and 28 frequency islands to allow independent **DVFS** of cores and mesh. As performance scales, the processor dissipates between 25 W and 125 W. … 567 mm^2 processor on **45 nm CMOS** integrates **48** IA-32 **cores** and 4 DDR3 channels in a **2D-mesh network**. **Cores communicate through message passing** using 384 KB of on-die shared memory. Fine-grain power management

Some Perspective…



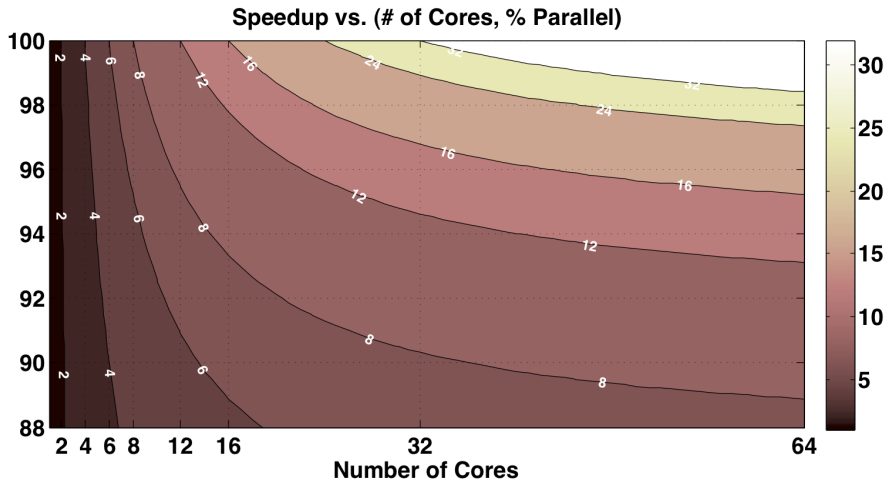
- *"For a 60-nanometer process a signal can reach only 5% of the die's length in a clock cycle"* [D. Matzke (Texas Instruments), IEEE Computer Sept. 97]
- Shift from function-centric to communication-centric design

Impediments to Parallel Performance

- **All ★'ed items also affect Fraction$_{parallelizable}$**
  - **(and hence speedup)**

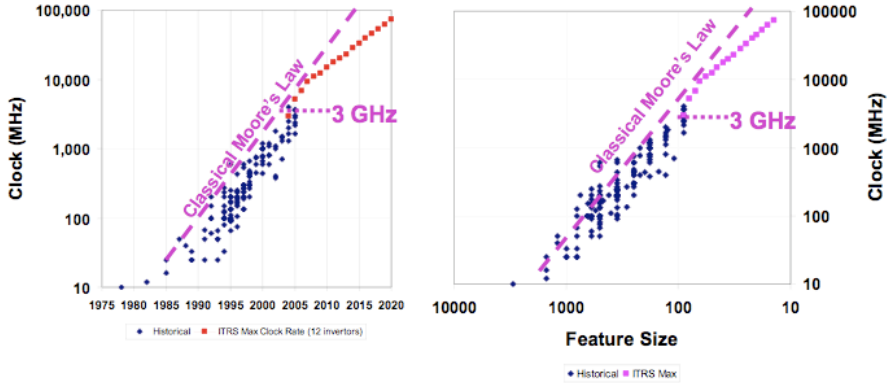
$$\text{Speedup} = \frac{1}{\left[1 - \text{Fraction}_{\text{parallelizable}}\right] + \dfrac{\text{Fraction}_{\text{parallelizable}}}{N}}$$

Multi-core only as good as algorithms that use it



Summary

- For many years, could double performance just by making device smaller
  - Clock rates increased with manageable power impact

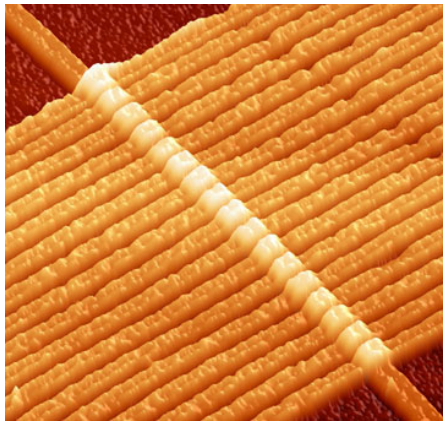


Summary

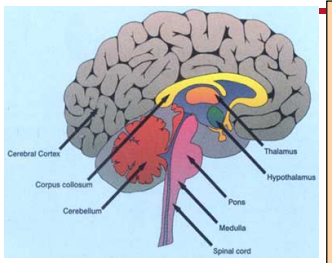
- Now:
  - Devices get smaller, but also run slower!
  - Performance comes from parallelism
  - But to parallelize, need new algorithms, software support
  - Also must overcome non-parallelizable overheads that degrade performance
  - …
- Low hanging fruit very much gone
  - New logic (and memory!) devices that **(a)** don't have same inherent problems as switch-based logic and/or **(b)** enable new system architectures are sought…

Motivating example 1

- Brain-inspired computation:
  - New, memristive devices may enable neuromorphic computer architectures…



Most complex information-management system in the universe…



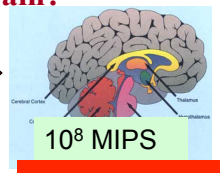
	Dell 8250 (Pentium® 4)	Brain
Mass	~25 kg	1.4 kg
Volume	34200 cm^3	1350 cm^3
MIPS	~10^3 MIPS	10^8 MIPS
BIT	<10^{16} bit/s	10^{19}bit/s
Power	**200 W**	**30 W (max)**
	~ 5 MIPS/W	**3x10^6 MIPS/W**
	5x10^6 k_BT / bit	**700 k_BT/bit**

A CMOS machine at the limits of scaling would use prodigious amounts of power

When will computer hardware match the human brain?

Dell 8250 (Pentium®4)
10^3 MIPS
200 W

x 10^5
x 10^5

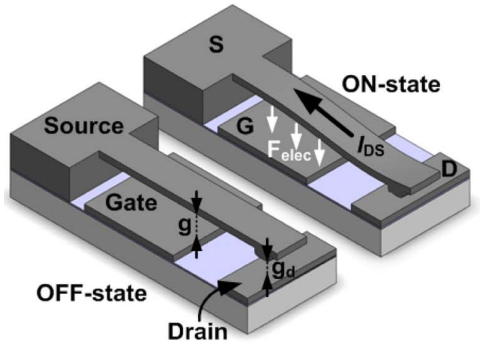


10^8 MIPS
=20,000 000 W

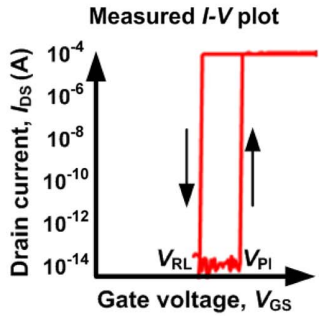
Example 2: Back to relays???

Mechanical Computing Redux: Relays for Integrated Circuit Applications

By Vincent Pott, *Member IEEE*, Hei Kam, *Member IEEE*, Rhesa Nathanael, *Student Member IEEE*, Jaeseok Jeon, *Student Member IEEE*, Elad Alon, *Member IEEE*, and Tsu-Jae King Liu, *Fellow IEEE*



(a)

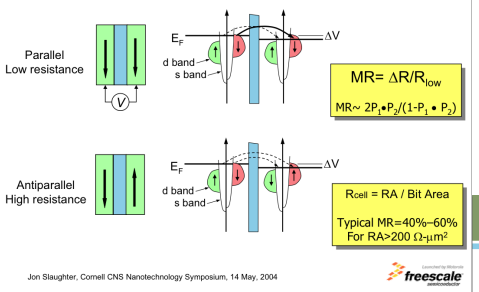


(b)

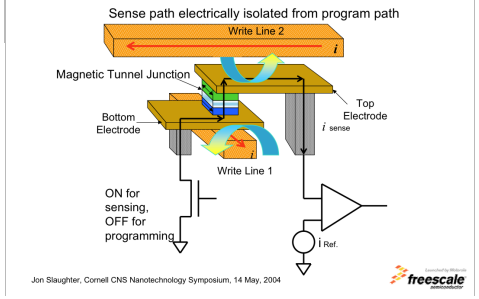
Possible advantages: no leakage, programmable, ….

Example 3: Universal memories

Tunneling Magnetoresistance



4Mb MRAM Bit Cell



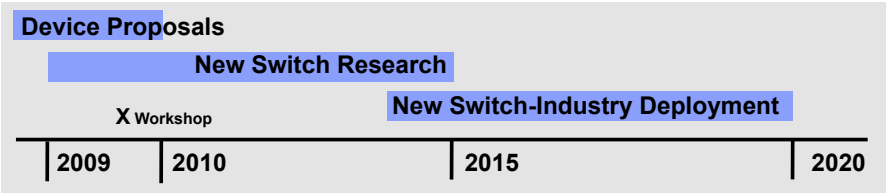
IBM

Where We're At

1. CMOS is a hard act to follow! None of the proposed post-CMOS switch candidates appear to be "drop-in" replacements.
2. Electron/charge state variables so far superior to alternatives. Not necessarily the best, if better architectures are developed.

Where We're Headed

Analog for example…



- MIND Workshop on Architectures for New Devices 08/2009
- Monthly Center Chief Operating Officer coordination
- NRI / MIND Read-outs for member companies

Now, onto the syllabus…