CSCI-2500: Computer Organization

Computer Abstractions, Technology and History

The Computer Revolution

- Progress in computer technology
  - Underpinned by Moore's Law
- Makes novel applications feasible
  - Computers in automobiles
  - Cell phones
  - Human genome project
  - World Wide Web
  - Search engines
- Computers are pervasive

Bill Gates "joke"

- At a computer expo (COMDEX), Bill Gates reportedly compared the computer industry with the auto industry and stated that :-

- "If GM had kept up with technology like the computer industry has, we would all be driving twenty-five dollar cars that got 1000 miles to the gallon."

GM's "joke" response!

- In response to Gates' comments, General Motors issued a press release stating:

  If GM had developed technology like Microsoft, we would all be driving cars with the following characteristics:

  1. For no reason whatsoever your car would crash twice a day.

  2. Every time they repainted the lines on the road you would have to buy a new car.

  3. Occasionally your car would die on the freeway for no reason, and you would just accept this, restart and drive on.

  4. Occasionally, executing a maneuver such as a left turn, would cause your car to shut down and refuse to restart, in which case you would have to reinstall the engine.

  5. Only one person at a time could use the car, unless you bought "Car95" or "CarNT". But then you would have to buy more seats.

  6. Macintosh/Apple would make a car that was powered by the sun, reliable, five times as fast, and twice as easy to drive, but would only work on five percent of the roads.

GM "joke" continues…

7. The oil, water temperature and alternator warning lights would be replaced by a single "general car default" warning light.

8. New seats would force everyone to have the same size butt.

9. The airbag system would say "Are you sure?" before going off.

10. Occasionally for no reason whatsoever, your car would lock you out and refuse to let you in until you simultaneously lifted the door handle, turned the key, and grab hold of the radio antenna.

11. GM would require all car buyers to also purchase a deluxe set of Rand McNally road maps (now a GM subsidiary), even though they neither need them nor want them. Attempting to delete this option would immediately cause the car's performance to diminish by 50% or more. Moreover, GM would become a target for investigation by the Justice Department.

12. Every time GM introduced a new model car buyers would have to learn to drive all over again because none of the controls would operate in the same manner as the old car.

13. You'd press the "start" button to shut off the engine.

Lesson…

- So, while computers have improved in performance vastly over the last 50 years, other "usability" aspects of computers remain open problems

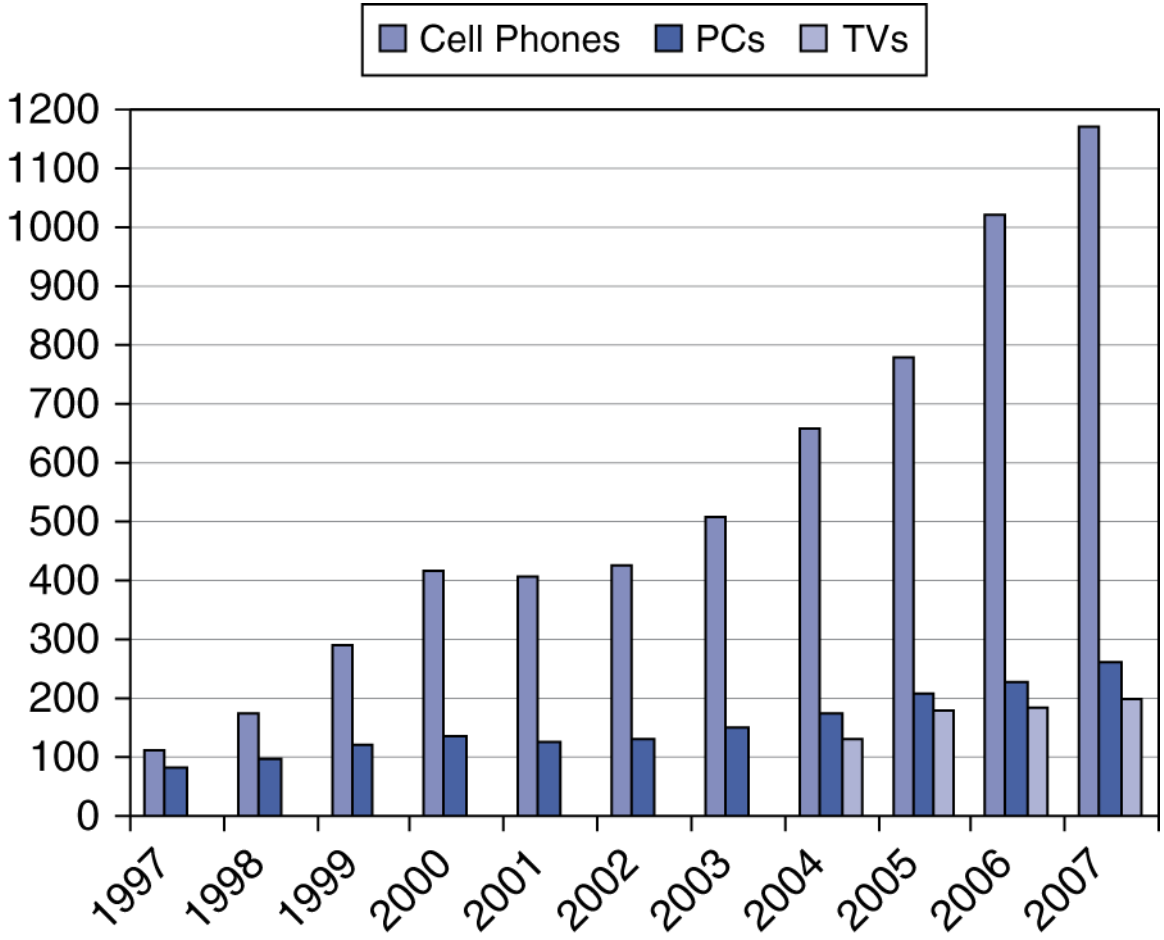
- …but this is comp org and so we'll focus a great deal on computer system performance…

  - Note, this exchange is Internet "lore"
  - See: http://www.snopes.com/humor/jokes/autos.asp
  - Thanks to R. Wellington IV

Classes of Computers

- Desktop computers
  - General purpose, variety of software
  - Subject to cost/performance tradeoff
- Server computers
  - Network based
  - High capacity, performance, reliability
  - Range from small servers to building sized
- Embedded computers
  - Hidden as components of systems
  - Stringent power/performance/cost constraints

The Processor Market



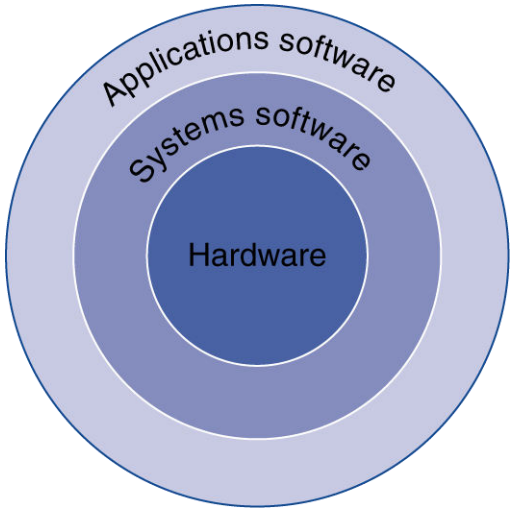
Topics You Will Learn

- Process by which programs are translated into the machine language
  - And how the hardware executes them
- The hardware/software interface
- Factors that determine program performance
  - And how it can be improved
- Approaches to improving performance
- Multicore parallel processing

Understanding Performance

- Algorithm
  - Determines number of operations executed
- Programming language, compiler, architecture
  - Determine number of machine instructions executed per operation
- Processor and memory system
  - Determine how fast instructions are executed
- I/O system (including OS)
  - Determines how fast I/O operations are executed

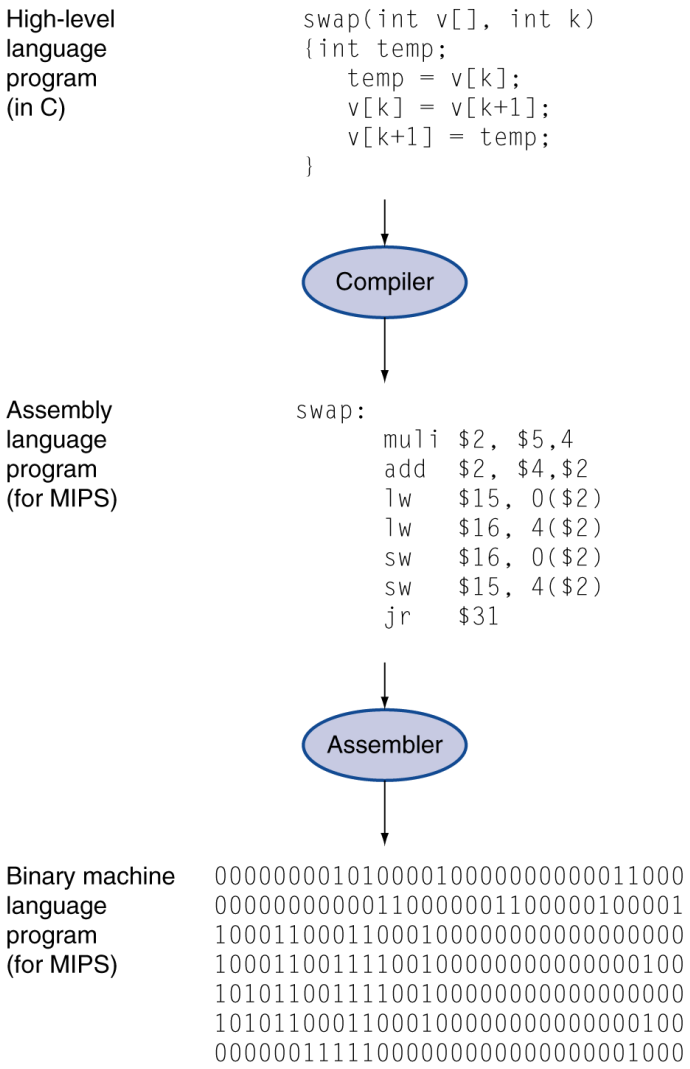
Below Your Program



- Application software
  - Written in high-level language
- System software
  - Compiler: translates HLL code to machine code
  - Operating System: service code
    - Handling input/output
    - Managing memory and storage
    - Scheduling tasks & sharing resources
- Hardware
  - Processor, memory, I/O controllers

Levels of Program Code

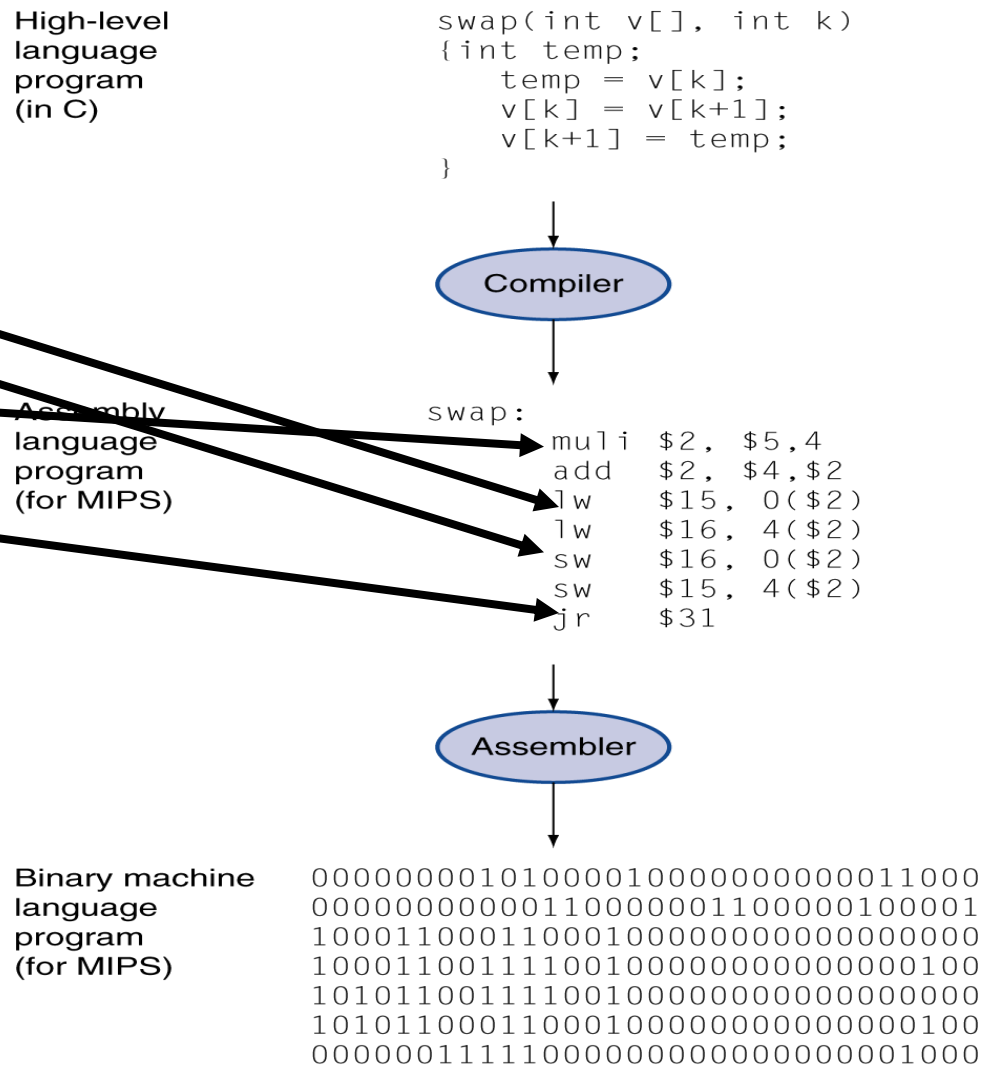
- High-level language
  - Level of abstraction closer to problem domain
  - Provides for productivity and portability
- Assembly language
  - Textual representation of instructions
- Hardware representation
  - Binary digits (bits)
  - Encoded instructions and data



Breaking down the hierarchy

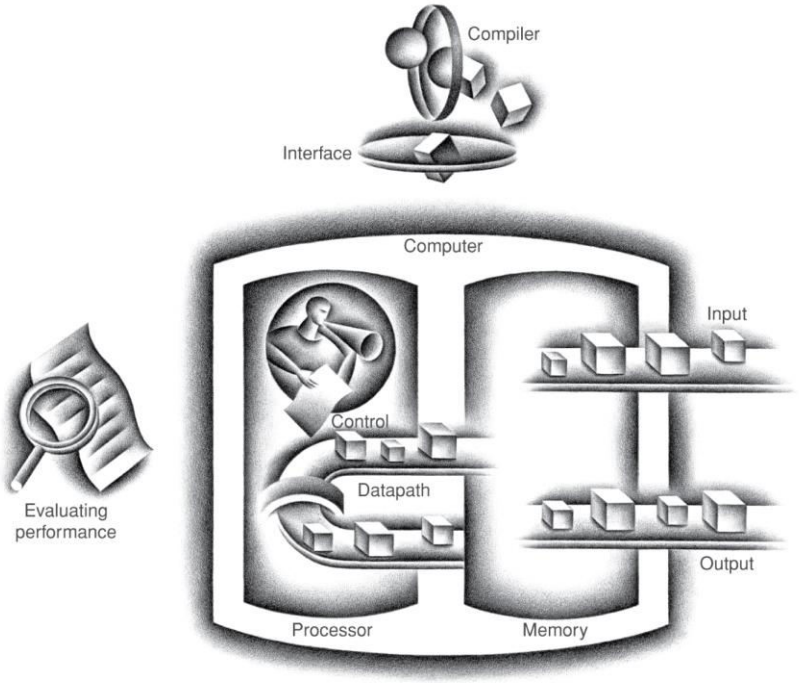
High-level code becomes a collection of:

1. Data movement operations
2. Compute operations
3. Program flow



Components of a Computer

The BIG Picture

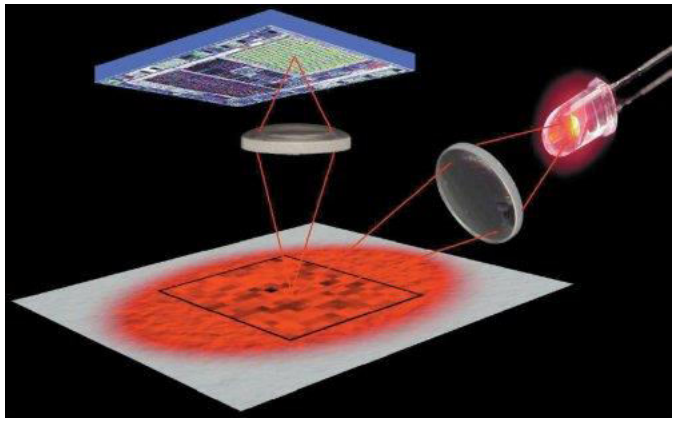


- Same components for all kinds of computer
  - Desktop, server, embedded
- Input/output includes
  - User-interface devices
    - Display, keyboard, mouse
  - Storage devices
    - Hard disk, CD/DVD, flash
  - Network adapters
    - For communicating with other computers

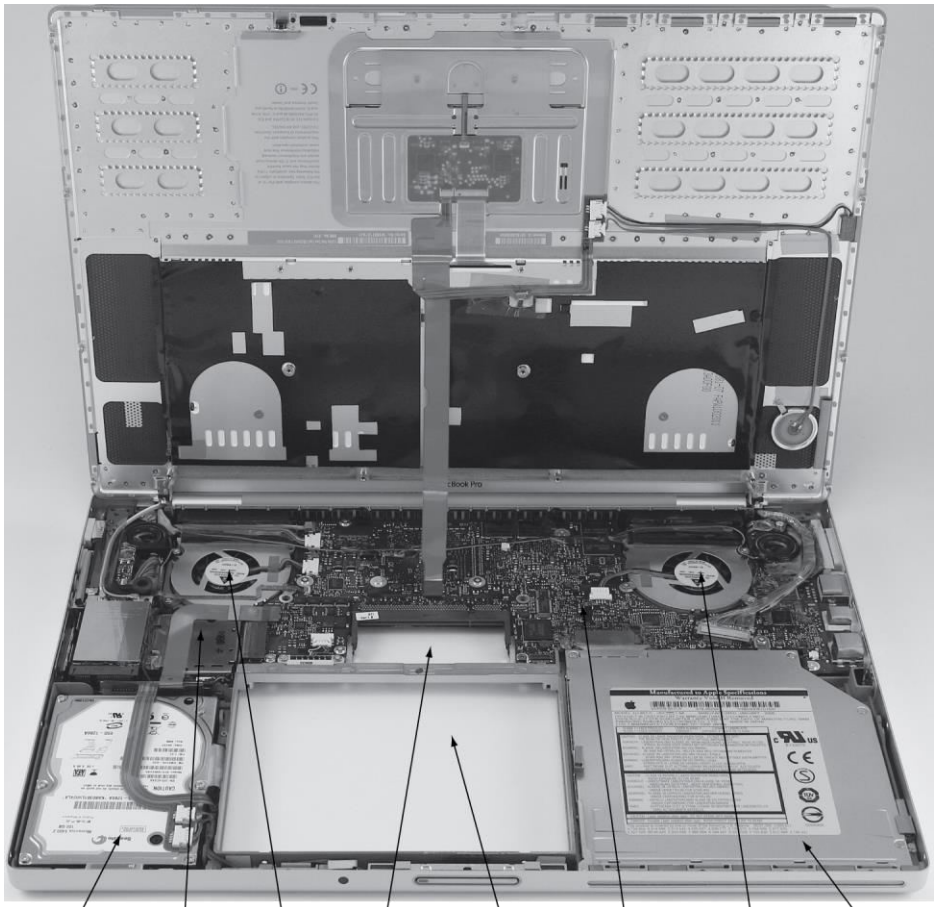
Anatomy of a Computer

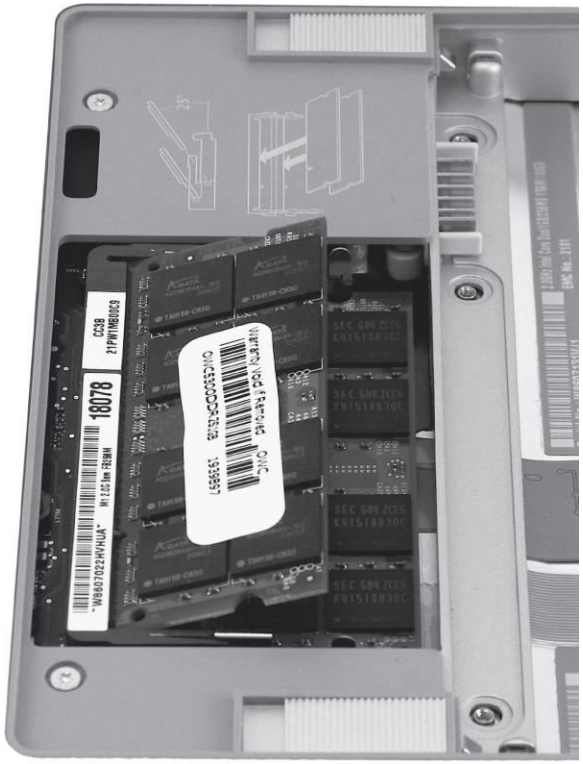
Anatomy of a Mouse

- Optical mouse
  - LED illuminates desktop
  - Small low-res camera
  - Basic image processor
    - Looks for x, y movement
  - Buttons & wheel
- Supersedes roller-ball mechanical mouse



Opening the Box



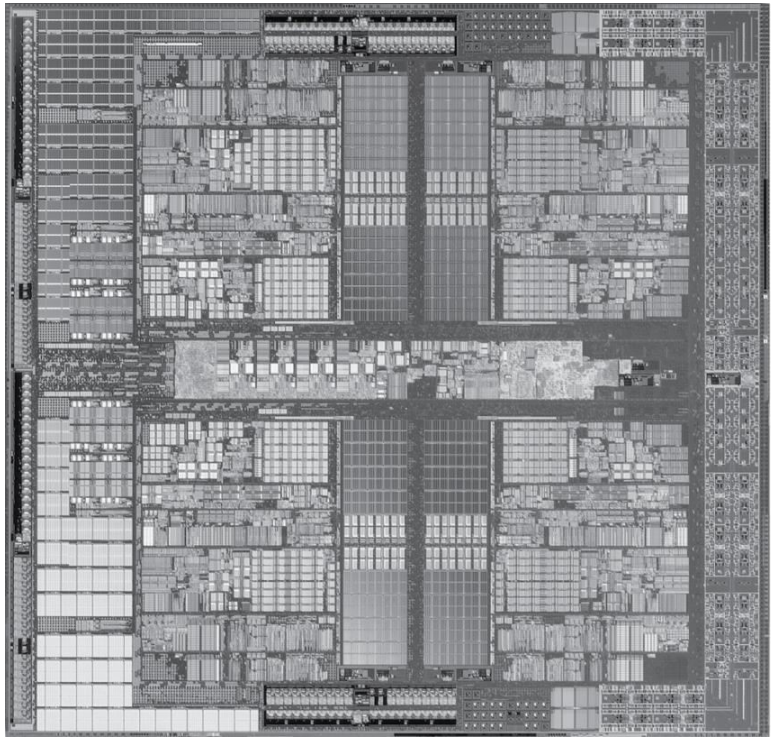


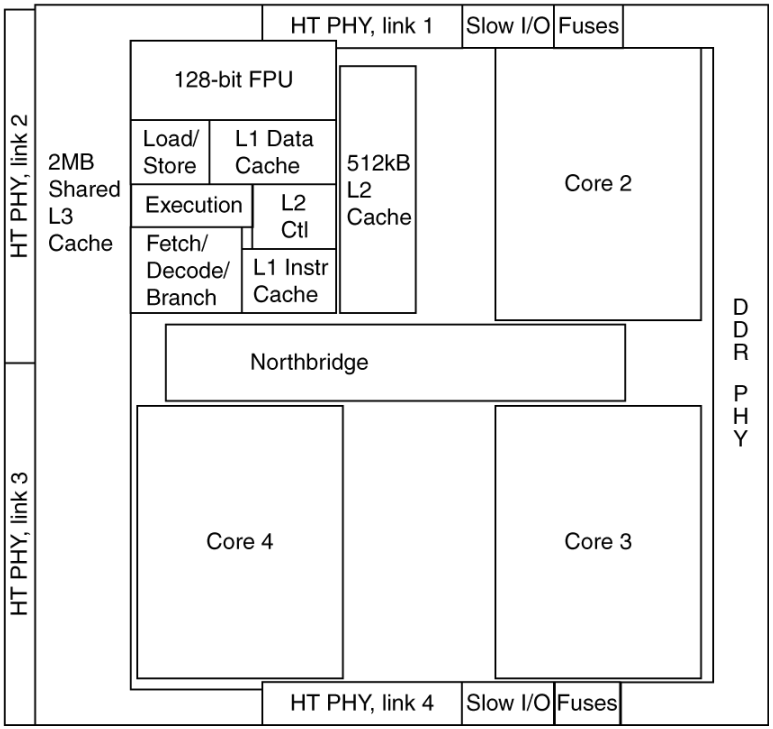
Inside the Processor (CPU)

- Datapath: performs operations on data
- Control: sequences datapath, memory, …
- Cache memory
  - Small fast SRAM memory for immediate access to data

Inside the Processor

- AMD Barcelona: 4 processor cores





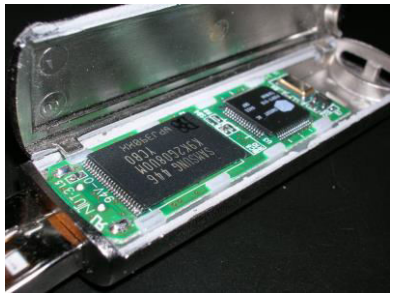
Abstractions

The BIG Picture

- Abstraction helps us deal with complexity
  - Hide lower-level detail
- Instruction set architecture (ISA)
  - The hardware/software interface
- Application binary interface (ABI)
  - The ISA plus system software interface
- Implementation
  - The underlying details and interface

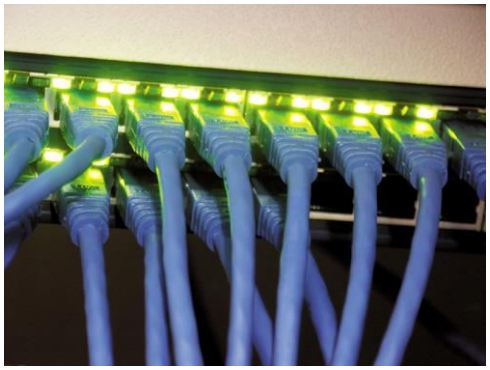
A Safe Place for Data

- Volatile main memory
  - Loses instructions and data when power off
- Non-volatile secondary memory
  - Magnetic disk
  - Flash memory
  - Optical disk (CDROM, DVD)



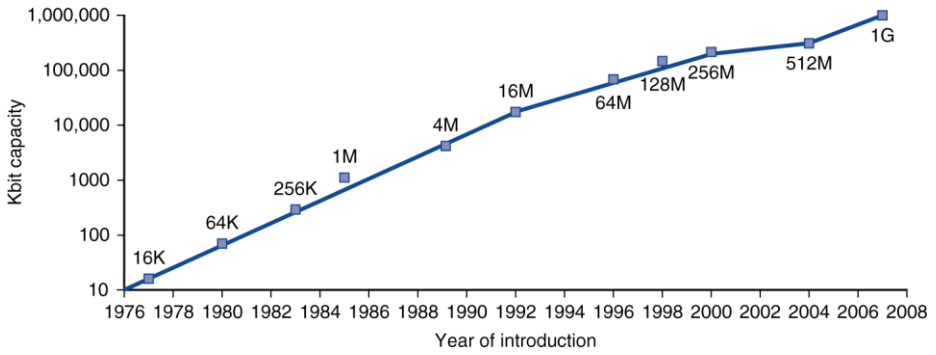
Networks

- Communication and resource sharing
- Local area network (LAN): Ethernet
  - Within a building
- Wide area network (WAN): the Internet
- Wireless network: WiFi, Bluetooth



Technology Trends

- Electronics technology continues to evolve
  - Increased capacity and performance
  - Reduced cost

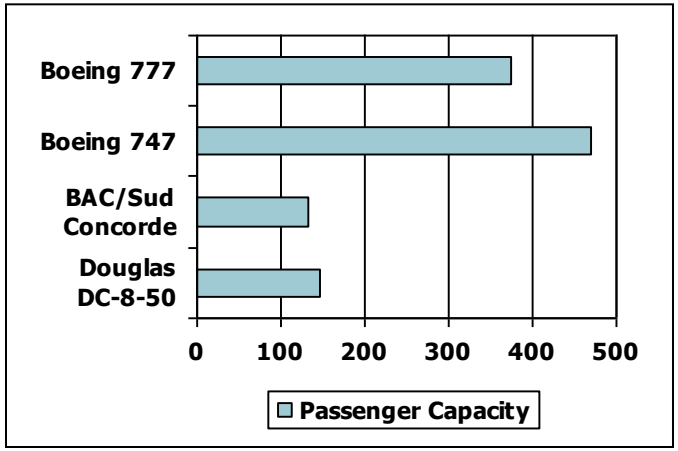


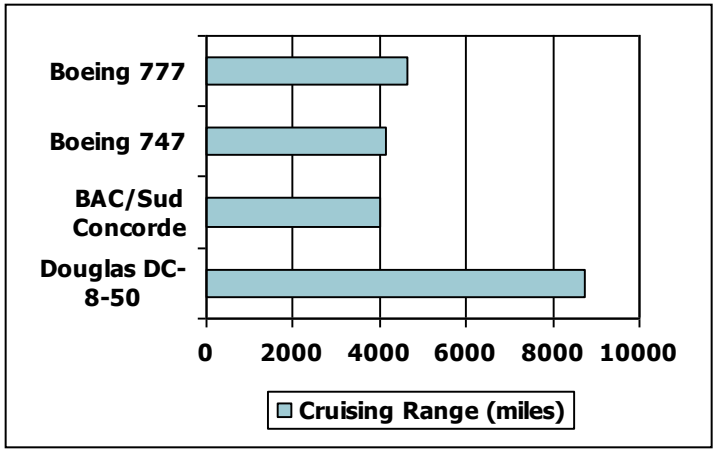
DRAM capacity

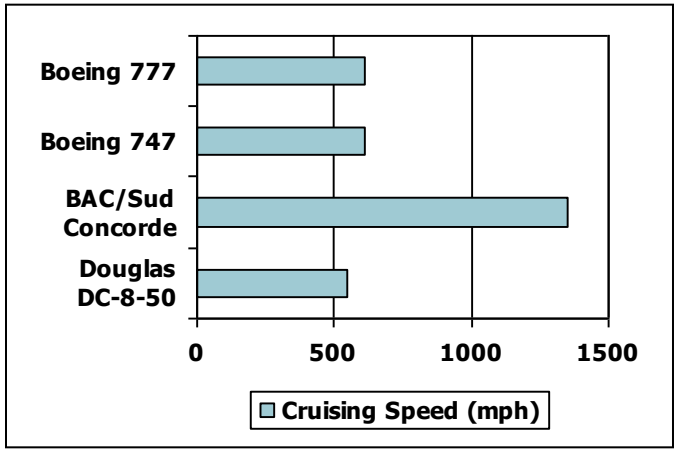
Year	Technology	Relative performance/cost
1951	Vacuum tube	1
1965	Transistor	35
1975	Integrated circuit (IC)	900
1995	Very large scale IC (VLSI)	2,400,000
2005	Ultra large scale IC	6,200,000,000

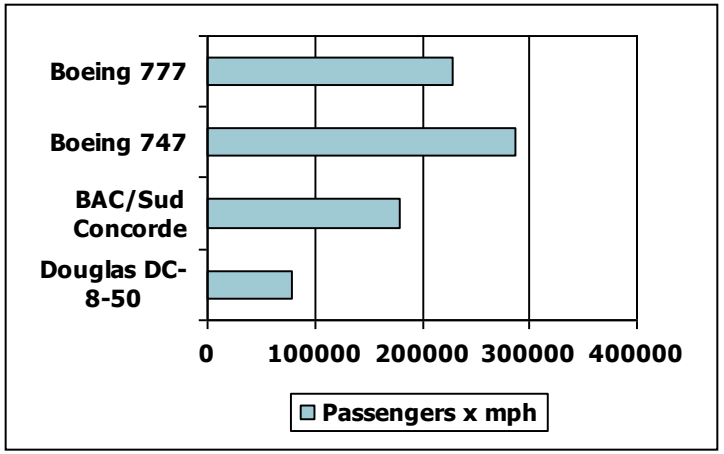
Defining Performance

- Which airplane has the best performance?









Response Time and Throughput

- Response time
  - How long it takes to do a task
- Throughput
  - Total work done per unit time
    - e.g., tasks/transactions/… per hour
- How are response time and throughput affected by
  - Replacing the processor with a faster version?
  - Adding more processors?
- We'll focus on response time for now…

Relative Performance

- Define Performance = 1/Execution Time
- "X is n times faster than Y"

$$\text{Performance}_X / \text{Performance}_Y$$
$$= \text{Execution time}_Y / \text{Execution time}_X = n$$

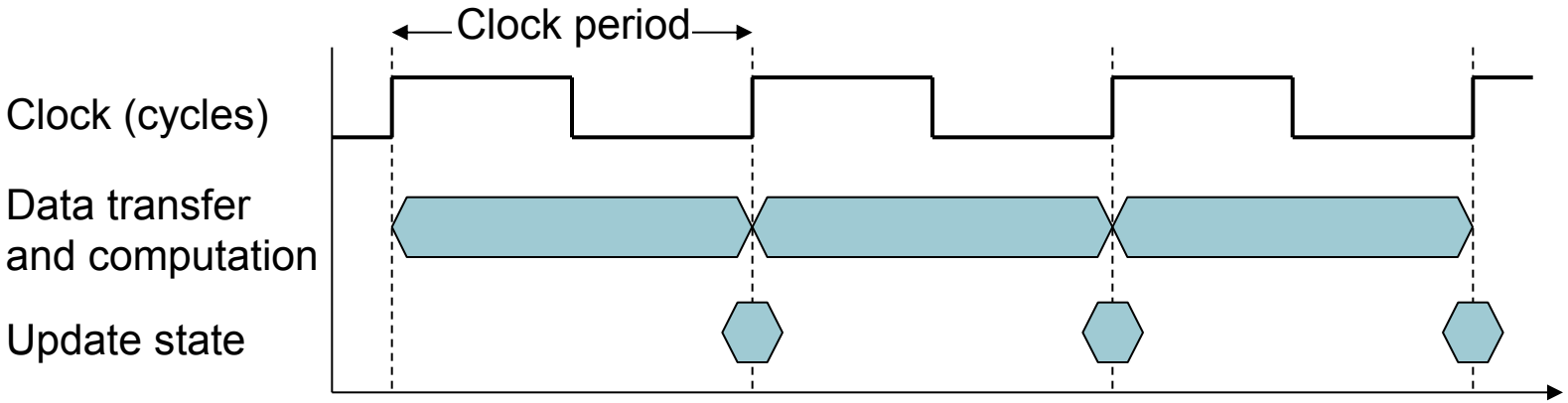
- Example: time taken to run a program
  - 10s on A, 15s on B
  - Execution Time_B / Execution Time_A = 15s / 10s = 1.5
  - So A is 1.5 times faster than B

Measuring Execution Time

- Elapsed time
  - Total response time, including all aspects
    - Processing, I/O, OS overhead, idle time
  - Determines system performance
- CPU time
  - Time spent processing a given job
    - Discounts I/O time, other jobs' shares
  - Comprises user CPU time and system CPU time
  - Different programs are affected differently by CPU and system performance

CPU Clocking

- Operation of digital hardware governed by a clock, which can change to save power



  - Clock period: duration of a clock cycle
    - e.g., 250ps = 0.25ns = 250×10^{-12}s
  - Clock frequency (rate): cycles per second
    - e.g., 4.0GHz = 4000MHz = 4.0×10^{9}Hz = 4 GHz

CPU Time

$$\text{CPU Time} = \text{CPU Clock Cycles} \times \text{Clock Cycle Time}$$

$$= \frac{\text{CPU Clock Cycles}}{\text{Clock Rate}}$$

- Performance improved by
  - Reducing number of clock cycles
  - Increasing clock rate
  - Hardware designer must often trade off clock rate against cycle count

CPU Time Example

- Computer A: 2GHz clock, 10s CPU time
- Designing Computer B
  - Aim for 6s CPU time
  - Can do faster clock, but causes 1.2 × clock cycles
- How fast must Computer B clock be?

$$\text{Clock Rate}_B = \frac{\text{Clock Cycles}_B}{\text{CPU Time}_B} = \frac{1.2 \times \text{Clock Cycles}_A}{6\text{s}}$$

$$\text{Clock Cycles}_A = \text{CPU Time}_A \times \text{Clock Rate}_A$$

$$= 10\text{s} \times 2\text{GHz} = 20 \times 10^9$$

$$\text{Clock Rate}_B = \frac{1.2 \times 20 \times 10^9}{6\text{s}} = \frac{24 \times 10^9}{6\text{s}} = 4\text{GHz}$$

Instruction Count and CPI

$$\text{Clock Cycles} = \text{Instruction Count} \times \text{Cycles per Instruction}$$

$$\text{CPU Time} = \text{Instruction Count} \times \text{CPI} \times \text{Clock Cycle Time}$$

$$= \frac{\text{Instruction Count} \times \text{CPI}}{\text{Clock Rate}}$$

- Instruction Count for a program
  - Determined by program, ISA and compiler
- Average cycles per instruction
  - Determined by CPU hardware
  - If different instructions have different CPI
    - Average CPI affected by instruction mix

CPI Example

- Computer A: Cycle Time = 250ps, CPI = 2.0
- Computer B: Cycle Time = 500ps, CPI = 1.2
- Same ISA
- Which is faster, and by how much?

$$\text{CPUTime}_A = \text{Instruction Count} \times \text{CPI}_A \times \text{Cycle Time}_A$$
$$= I \times 2.0 \times 250\text{ps} = I \times 500\text{ps}$$

A is faster…

$$\text{CPUTime}_B = \text{Instruction Count} \times \text{CPI}_B \times \text{Cycle Time}_B$$
$$= I \times 1.2 \times 500\text{ps} = I \times 600\text{ps}$$

$$\frac{\text{CPUTime}_B}{\text{CPUTime}_A} = \frac{I \times 600\text{ps}}{I \times 500\text{ps}} = 1.2$$

…by this much

CPI in More Detail

- If different instruction classes take different numbers of cycles

$$\text{Clock Cycles} = \sum_{i=1}^{n} (\text{CPI}_i \times \text{Instruction Count}_i)$$

- Weighted average CPI

$$\text{CPI} = \frac{\text{Clock Cycles}}{\text{Instruction Count}} = \sum_{i=1}^{n} \left(\text{CPI}_i \times \underbrace{\frac{\text{Instruction Count}_i}{\text{Instruction Count}}}_{\text{Relative frequency}} \right)$$

CPI Example

- Alternative compiled code sequences using instructions in classes A, B, C

Class	A	B	C
CPI for class	1	2	3
IC in sequence 1	2	1	2
IC in sequence 2	4	1	1

- Sequence 1: IC = 5
  - Clock Cycles
    = 2 × 1 + 1 × 2 + 2 × 3
    = 10
  - Avg. CPI = 10/5 = 2.0
- Sequence 2: IC = 6
  - Clock Cycles
    = 4 × 1 + 1 × 2 + 1 × 3
    = 9
  - Avg. CPI = 9/6 = 1.5

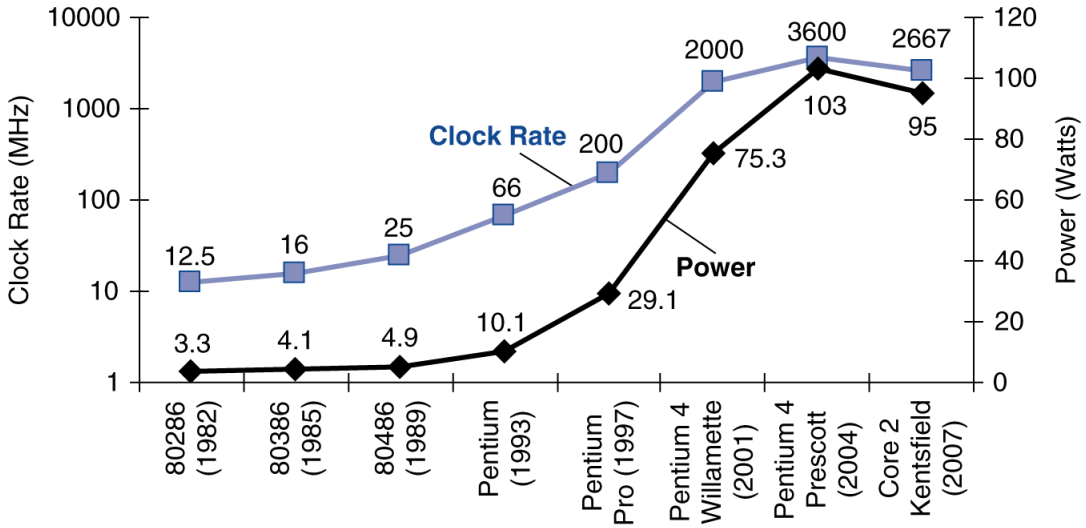
Performance Summary

The BIG Picture

$$\text{CPU Time} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Clock cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Clock cycle}}$$

- Performance depends on
  - Algorithm: affects IC, possibly CPI
  - Programming language: affects IC, CPI
  - Compiler: affects IC, CPI
  - Instruction set architecture: affects IC, CPI, T_c

Power Trends



- In CMOS IC technology

$$\text{Power} = \text{Capacitive load} \times \text{Voltage}^2 \times \text{Frequency}$$

× 30 | 5V → 1V | × 1000

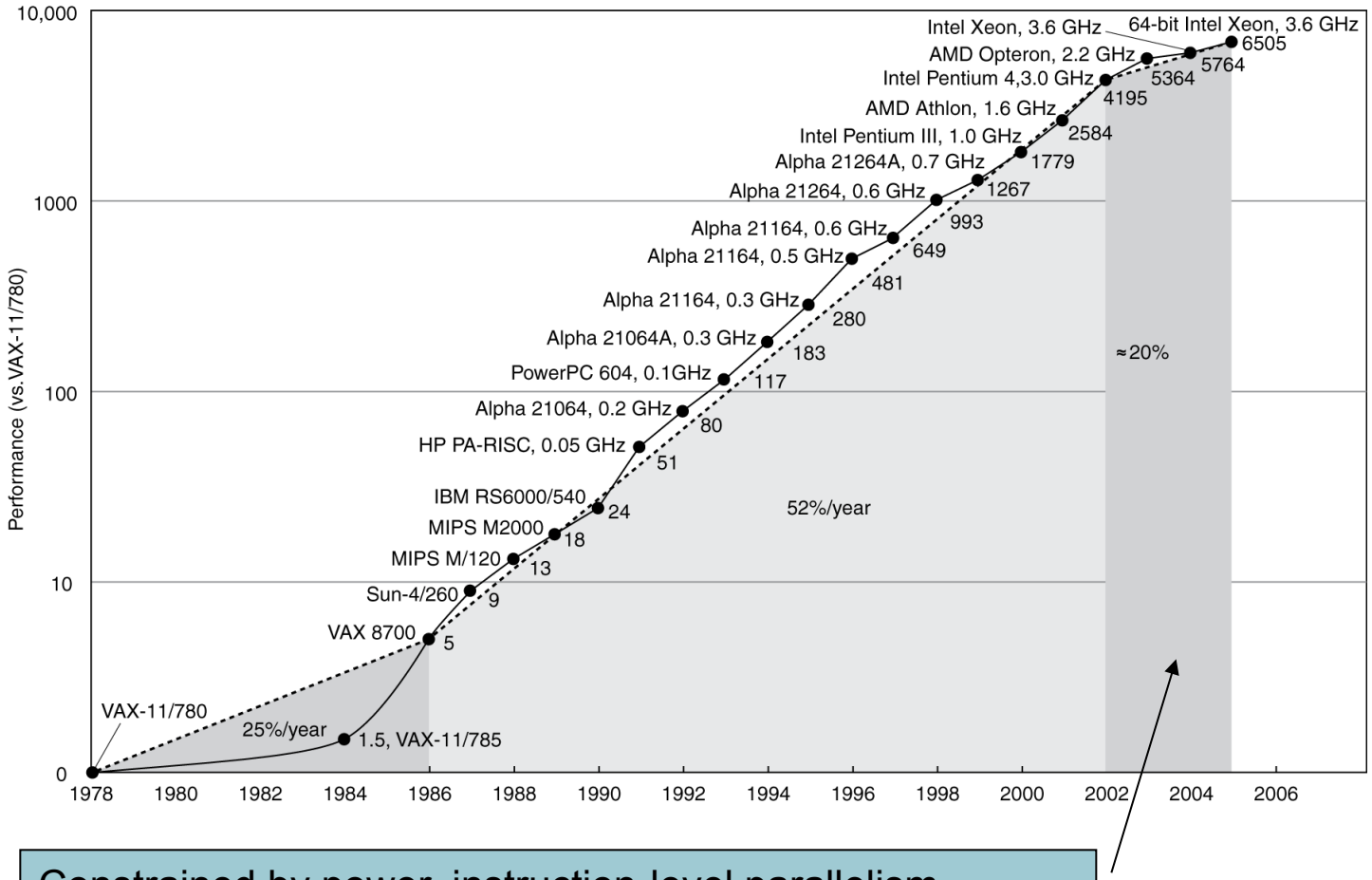
Reducing Power

- Suppose a new CPU has
  - 85% of capacitive load of old CPU
  - 15% voltage and 15% frequency reduction

$$\frac{P_{new}}{P_{old}} = \frac{C_{old} \times 0.85 \times (V_{old} \times 0.85)^2 \times F_{old} \times 0.85}{C_{old} \times V_{old}^{\ 2} \times F_{old}} = 0.85^4 = 0.52$$

- The power wall
  - We can't reduce voltage further
  - We can't remove more heat
- How else can we improve performance?

Uniprocessor Performance

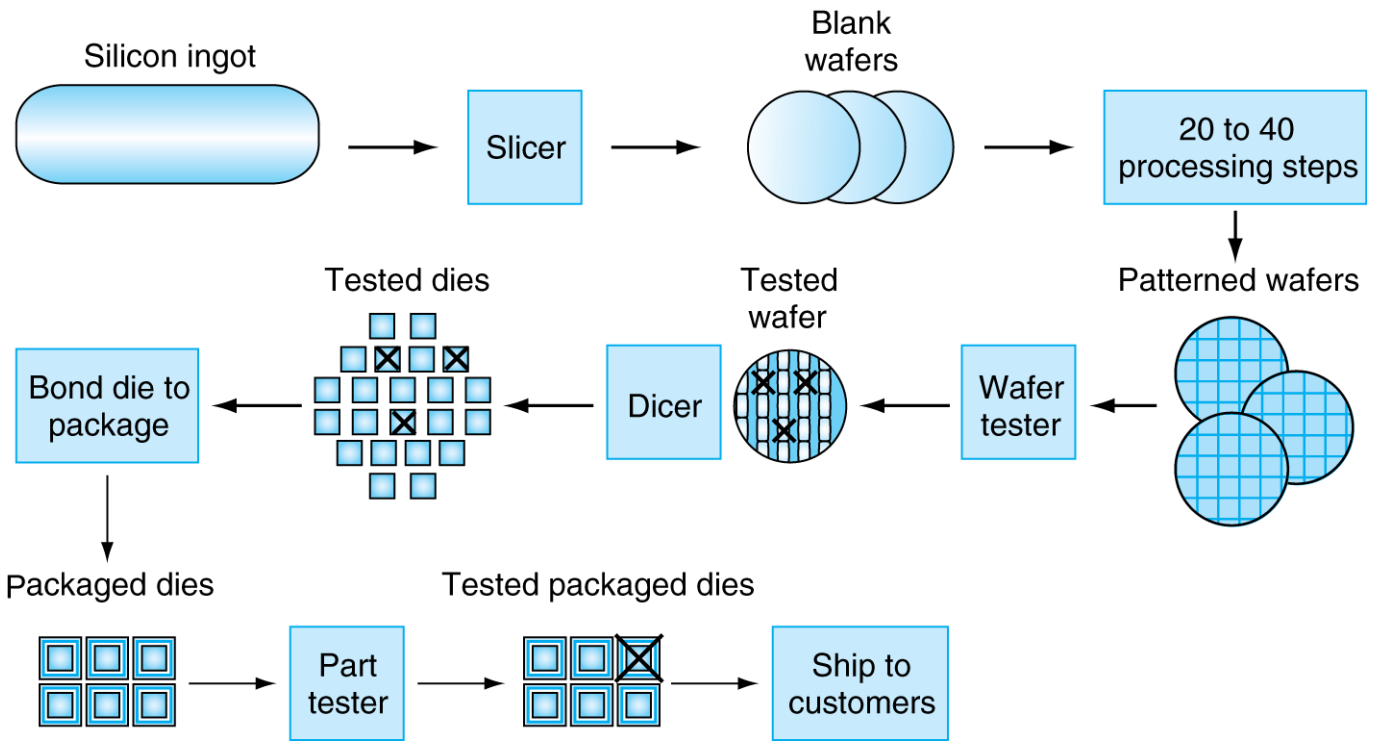


Constrained by power, instruction-level parallelism, memory latency

Multiprocessors

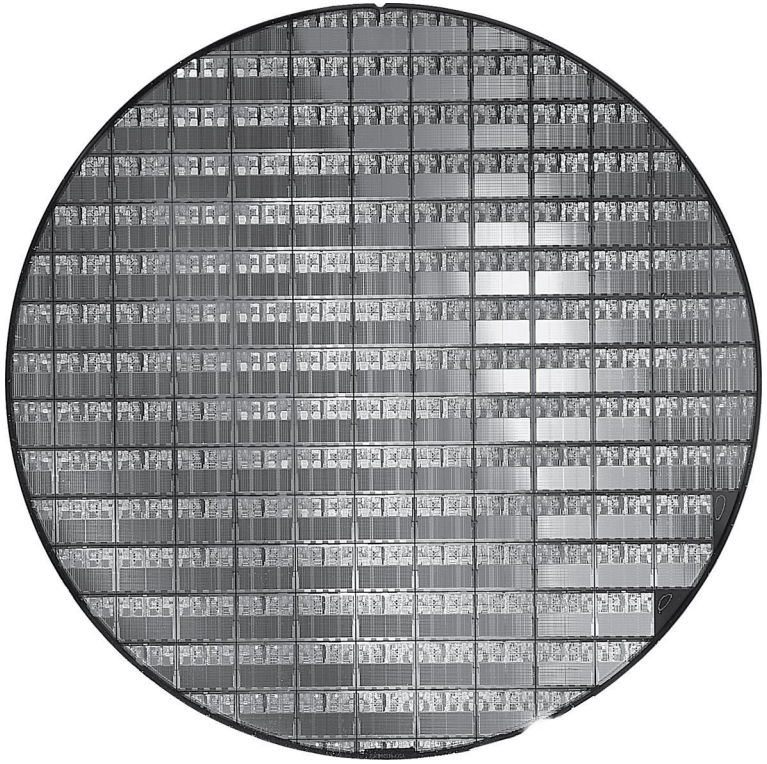
- Multicore microprocessors
  - More than one processor per chip
- Requires explicitly parallel programming
  - Compare with instruction level parallelism
    - Hardware executes multiple instructions at once
    - Hidden from the programmer
  - Hard to do
    - Programming for performance
    - Load balancing
    - Optimizing communication and synchronization

Manufacturing ICs



- Yield: proportion of working dies per wafer

AMD Opteron X2 Wafer



- X2: 300mm wafer, 117 chips, 90nm technology
- X4: 45nm technology

Integrated Circuit Cost

$$\text{Cost per die} = \frac{\text{Cost per wafer}}{\text{Dies per wafer} \times \text{Yield}}$$

$$\text{Dies per wafer} \approx \text{Wafer area}/\text{Die area}$$

$$\text{Yield} = \frac{1}{(1+(\text{Defects per area} \times \text{Die area}/2))^2}$$

- Nonlinear relation to area and defect rate
  - Wafer cost and area are fixed
  - Defect rate determined by manufacturing process
  - Die area determined by architecture and circuit design

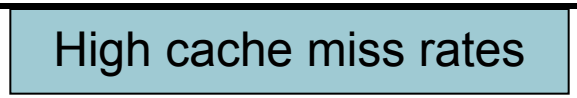
SPEC CPU Benchmark

- Programs used to measure performance
  - Supposedly typical of actual workload
- Standard Performance Evaluation Corp (SPEC)
  - Develops benchmarks for CPU, I/O, Web, …
- SPEC CPU2006
  - Elapsed time to execute a selection of programs
    - Negligible I/O, so focuses on CPU performance
  - Normalize relative to reference machine
  - Summarize as geometric mean of performance ratios
    - CINT2006 (integer) and CFP2006 (floating-point)

$$\sqrt[n]{\prod_{i=1}^{n} \text{Execution time ratio}_i}$$

CINT2006 for Opteron X4 2356

Name	Description	IC × 10^9	CPI	Tc (ns)	Exec time	Ref time	SPECratio
perl	Interpreted string processing	2,118	0.75	0.40	637	9,777	15.3
bzip2	Block-sorting compression	2,389	0.85	0.40	817	9,650	11.8
gcc	GNU C Compiler	1,050	1.72	0.47	24	8,050	11.1
mcf	Combinatorial optimization	336	10.00	0.40	1,345	9,120	6.8
go	Go game (AI)	1,658	1.09	0.40	721	10,490	14.6
hmmer	Search gene sequence	2,783	0.80	0.40	890	9,330	10.5
sjeng	Chess game (AI)	2,176	0.96	0.48	37	12,100	14.5
libquantum	Quantum computer simulation	1,623	1.61	0.40	1,047	20,720	19.8
h264avc	Video compression	3,102	0.80	0.40	993	22,130	22.3
omnetpp	Discrete event simulation	587	2.94	0.40	690	6,250	9.1
astar	Games/path finding	1,082	1.79	0.40	773	7,020	9.1
xalancbmk	XML parsing	1,058	2.70	0.40	1,143	6,900	6.0
Geometric mean							11.7



SPEC Power Benchmark

- Power consumption of server at different workload levels
  - Performance: ssj_ops/sec
  - Power: Watts (Joules/sec)

$$\text{Overall ssj\_ops per Watt} = \left(\sum_{i=0}^{10} \text{ssj\_ops}_i\right) \Big/ \left(\sum_{i=0}^{10} \text{power}_i\right)$$

SPECpower_ssj2008 for X4

Target Load %	Performance (ssj_ops/sec)	Average Power (Watts)
100%	231,867	295
90%	211,282	286
80%	185,803	275
70%	163,427	265
60%	140,160	256
50%	118,324	246
40%	920,35	233
30%	70,500	222
20%	47,126	206
10%	23,066	180
0%	0	141
Overall sum	1,283,590	2,605
Σssj_ops/ Σpower		493

Pitfall: Amdahl's Law

- Improving an aspect of a computer and expecting a proportional improvement in overall performance

$$T_{improved} = \frac{T_{affected}}{improvement\ factor} + T_{unaffected}$$

- Example: multiply accounts for 80s/100s
  - How much improvement in multiply performance to get 5× overall?

$$20 = \frac{80}{n} + 20$$

  - Can't be done!
- Corollary: make the common case fast

Fallacy: Low Power at Idle

- Look back at X4 power benchmark
  - At 100% load: 295W
  - At 50% load: 246W (83%)
  - At 10% load: 180W (61%)
- Google data center
  - Mostly operates at 10% – 50% load
  - At 100% load less than 1% of the time
- Consider designing processors to make power proportional to load

Pitfall: MIPS as a Performance Metric

- MIPS: Millions of Instructions Per Second
  - Doesn't account for
    - Differences in ISAs between computers
    - Differences in complexity between instructions

$$\text{MIPS} = \frac{\text{Instruction count}}{\text{Execution time} \times 10^6}$$

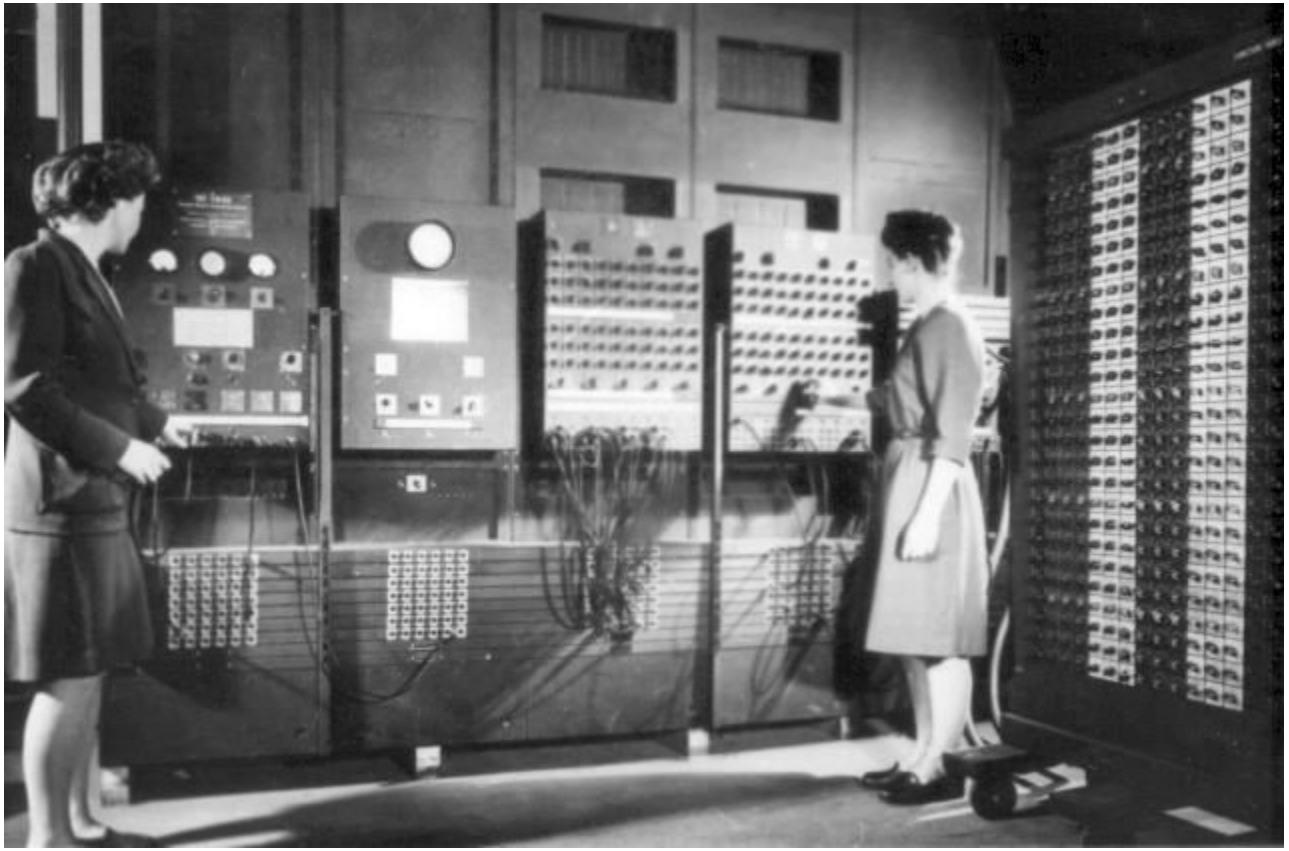
$$= \frac{\text{Instruction count}}{\frac{\text{Instruction count} \times \text{CPI}}{\text{Clock rate}} \times 10^6} = \frac{\text{Clock rate}}{\text{CPI} \times 10^6}$$

  - CPI varies between programs on a given CPU

History: The Beginning…

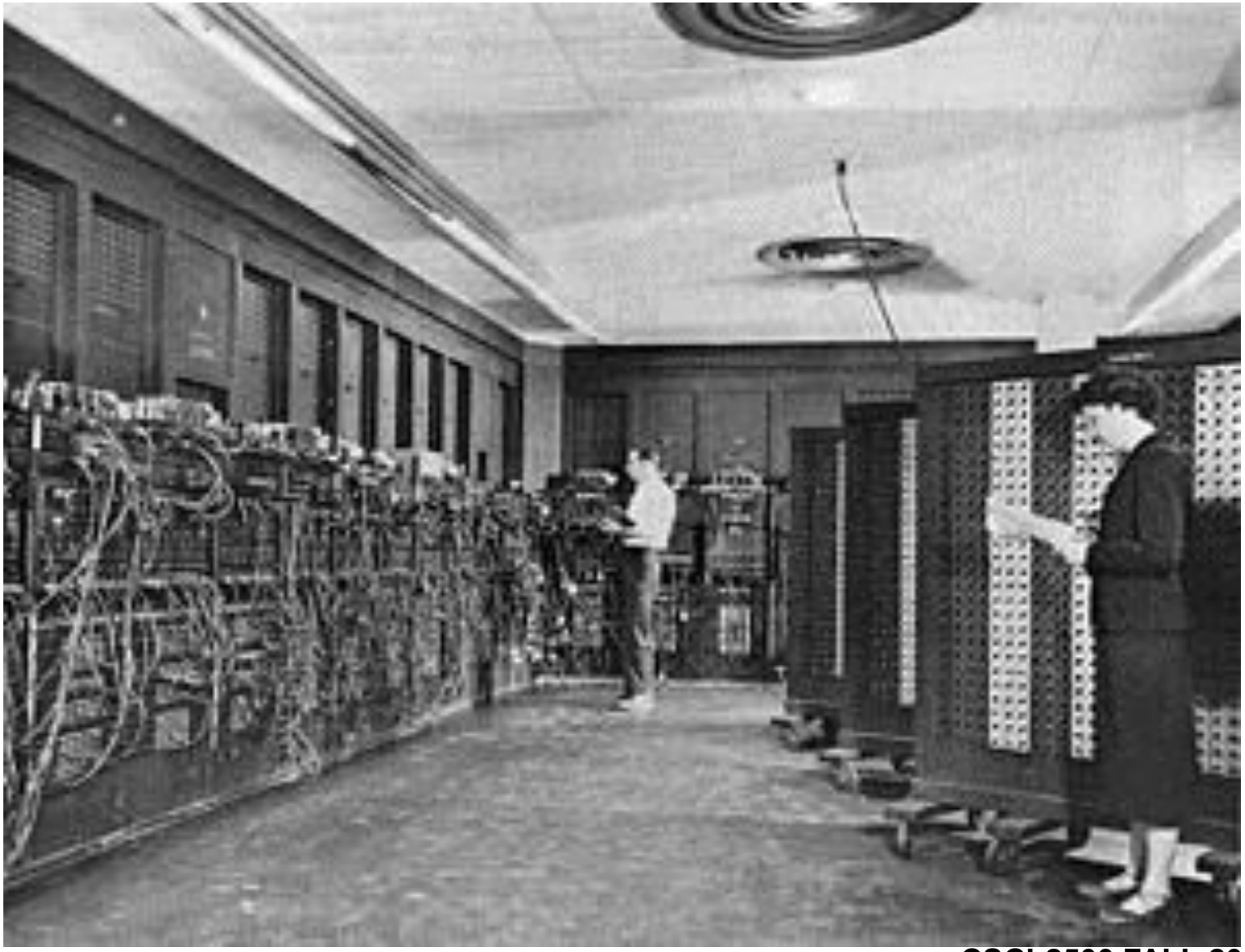
- ENIAC – ~1940s, First Electronic Computer
  - Built @ UPenn by Eckert and Mauchly
  - ENIAC ➔ Electronic Numerical Integrator and Calculator
  - HUGE! ➔ 80 feet long, 8.5 feet high and a couple of feet wide.
  - Each of the 20, 10 bit registers where 2 ft long.
  - Had a total of 18,000 vacuum tubes
  - Used to compute artillery firing tables

Pictures of ENIAC



Programmers/Early "Computer Geeks" Betty Jean Jennings (left) and Fran Bilas (right) operate the ENIAC's main control panel at the Moore School of Electrical Engineering. (U.S. Army photo from the archives of the ARL Technical Library)

Pictures of ENIAC…



The von Neumann Computer

- 1944: John von Neumann proposes idea for a "stored program" computer – e.g., that a "program" can be stored in memory just like data…
- EDVAC – Electronic Discrete Variable Automatic Computer
- All modern day computer systems are built with this key concept!

Conrad Zuse

- In 1930's and early 40's in Germany, Conrad had the design for a programmable computer ready.
- This was before von Neumann's draft memo!
- Zuse's computer would take 2 years to build
- German Gov. decided not to fund it since they predicted the war would be over before the machine was built…

Other Early Systems..

- Colossus – 1943 by Park and Turing (yes, Alan Turing built real machines).
- Harvard Architecture
  - Howard Akien in 1940's built the Mark-series of system which had separate memories – Main memory and cache!
  - All systems today are a "Harvard" architecture.
- MIT Whirlwind – 1947
  - Aimed at radar applications
  - Key innovation: magnetic core memory!
  - This type of memory would last for 30 years!

Early Commerical Computers

- Eckert-Mauchly Corporation - 1947
  - First system was BINAC, 1949
  - Bought out by Remington-Rand, which is not just Rand Corporation
  - Built UNIVAC I, 1951, sold for $1,000,000
    - 48 of these where built!
  - Then came UNIVAC II – more memory but "backwards compatible"

More early commerical systems

- IBM
  - Previously had punch card and office automation systems
  - 1952 – IBM 701 shipped its first full fledged computer system.
  - We'll see a lot more from IBM ! ☺

2nd Generation System..

- Transistors… what the heck are these silly little things good for!
  - Invented @ Bell Labs in 1947
    - Recall, ENIAC was not realized until 1946 at least publicly
    - So folks at Bell Labs had no clue about computers…
  - Smaller… Ok, that might be useful.
  - Cheaper… Ok, but at doing what.
  - Dissipated less heat… Fine, but was heat really a problem?
  - Actually, first patent goes back a bit further
    - First "FET" patent filed in Canada by Julius Lilenfeld on 22 Oct 1925
    - In 1934 Dr. Heil (Germany) patented another FET.
  - Not until the IBM 7000 series in 1953 did IBM use transistor technology in their computer systems.

Digital Equipment Corp…

- DEC, aka Compaq, aka HP…
  - Founded in 1957
  - Built world's first "mini" computer in ~1965
    - ➔ PDP-8
  - Mini's were smaller and cheaper than giant room sized mainframes.
  - Cost … only $20,000

The first Supercomputers…

- In 1963, Control Data Corporation (CDC) built the CDC 6600 series
  - This was dubbed "The 1st Supercomputer"
- Built by Seymour Cray (died in a car accident in 1996)
- He left CDC and formed Cray Research
- In 1976, the Cray-I was released
  - It was simultaneously the world's fastest, most expensive and best cost-performance system at that time!
  - Cray Research (at least its technology was purchased) was bought by SGI in 1996
  - SGI died, but Cray still lives on today!

3rd Gen Systems: IBM OS/360

- In 1965, IBM invested $5 BILLION in this new line of computer systems
- First "planned" family of computers or "line of products".
- Varied in price and performance by a factor of 25
- Now, at that time, all "processors" were the same (just like Intel today), IBM just sped them up by setting "dip switches".

Better Memories…

- Magnetic Core was the prevailing technology
  - Here, tiny rings of "ferromagnetic" material are strung together on wire.
  - The grids of wires were suspended on small screens inside the computer
  - Each ring was 1 bit of memory.
    - Magnetized +, the bit set to 1
    - Magnetized -, the bit set to 0
  - Fast access time ➔ 1 microsecond to read
  - CAVEAT: reading was DESTRUCTIVE … Ouch!
  - Required extra circuits to restore a value after reading.

1970 Fairchild Semiconductor

- Invented/Designed – memory on a chip using transistors
- Took 70 nanoseconds to read a single bit compared with 1000 nanoseconds for Magnetic Core memory.
- However, at this time, it costs more per bit than magnetic core memory…
- But in 1974, transistor memory became cheaper than magnetic core…

First microprocessor

- Intel 4004
- Co-invented by Dr. Ted Hoff in 1971
  - Dr. Hoff is an RPI alumnus! :-)
- Could add 2, 4 bit numbers
- Multiplication was done by repeated addition
- This led to first 8-bit microprocessor, Intel 8008 in 1974
  - First General CPU on a single chip!

"RISC" Processors…

- RISC ➔ Reduced Instruction Set Computer
- This "label" denotes a computer which favored "doing the common case fast"
  - E.g., Remove any instruction which slows the common case down.
  - Early RISC systems
    - IBM 801 (~1975) @ IBM
      - John Cocke – won Turing Award and Pres. Medal of Honor
    - MIPS – John Hennsey @ Stanford/MIPS/SGI
    - SPARC – David Patterson @ UCB used by Sun
  - CISC – Complex…
    - E.g., DEC's VAX
    - Had single instructions for computing polynomials
    - Tailored for human assembly language programmers and not compilers!
  - Hybrid ➔ Intel Pentium and beyond such as MMX, SSE, etc.

End of the "Cowboy" Era

- Prior to RISC, computers were designed trial-and-error / seat of the pants
  - Read "Soul of a New Machine".
- Lacked real notion about how to measure a computer's performance
- What RISC brought to computer architecture was a methodology/framework for evaluating a design and making the right tradeoffs to maximize a computer system's *overall* performance.

Half-Century of Progress

Year	Name	Size (cu. ft.)	Power (watts)	Performance (adds/sec)	Memory (KB)	Price	Price/ performance vs. UNIVAC	Adjusted price (2007 $)	Adjusted price/ performance vs. UNIVAC
1951	UNIVAC I	1,000	125,000	2,000	48	$1,000,000	1	$7,670,724	1
1964	IBM S/360 model 50	60	10,000	500,000	64	$1,000,000	263	$6,018,798	319
1965	PDP-8	8	500	330,000	4	$16,000	10,855	$94,685	13,367
1976	Cray-1	58	60,000	166,000,000	32,000	$4,000,000	21,842	$13,509,798	47,127
1981	IBM PC	1	150	240,000	256	$3,000	42,105	$6,859	134,208
1991	HP 9000/ model 750	2	500	50,000,000	16,384	$7,400	3,556,188	$11,807	16,241,889
1996	Intel PPro PC (200 MHz)	2	500	400,000,000	16,384	$4,400	47,846,890	$6,211	247,021,234
2003	Intel Pentium 4 PC (3.0 GHz)	2	500	6,000,000,000	262,144	$1,600	1,875,000,000	$2,009	11,451,750,000
2007	AMD Barcelona PC (2.5 GHz)	2	250	20,000,000,000	2,097,152	$800	12,500,000,000	$800	95,884,051,042

FIGURE 1.10.7 Characteristics of key commercial computers since 1950, in actual dollars and in 2007 dollars adjusted for inflation. The last row assumes we can fully utilize the potential performance of the four cores in Barcelona. In contrast to Figure 1.10.3, here the price of the IBM S/360 model 50 includes I/O devices. (Source: The Computer History Museum and Producer Price Index for Industrial Commodities.)

What about Today's Supercomputers?

So glad you asked…

Cray XT3 – BigBen @ PSC

- 2068 nodes (4136 cores)
  - dual-core 2.6 GHz AMD Opteron
  - 2 GB of RAM
- Seastar 3-D Torus
  - The peak bidirectional BW XT3 link is 7.6 GB/s, 4GB/s sustained
- Catamount OS

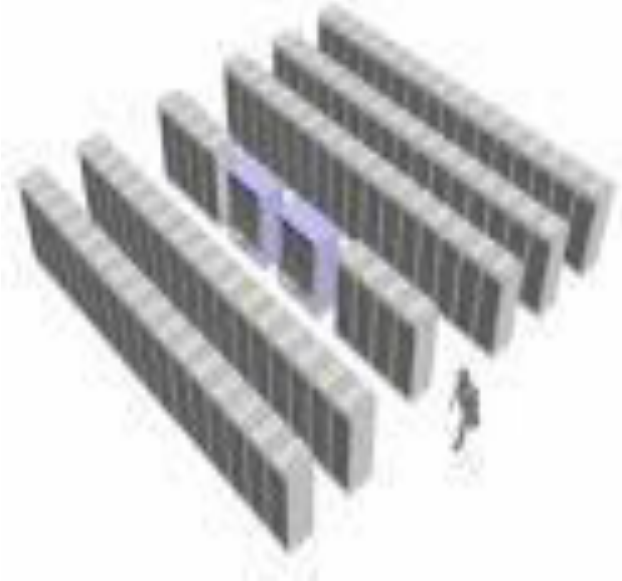
Different from the newer Cray XT4/XT5 (e.g. Kraken @ ORNL)

OS now is "Compute Node Linux" (CNL) vs. custom Catamount OS

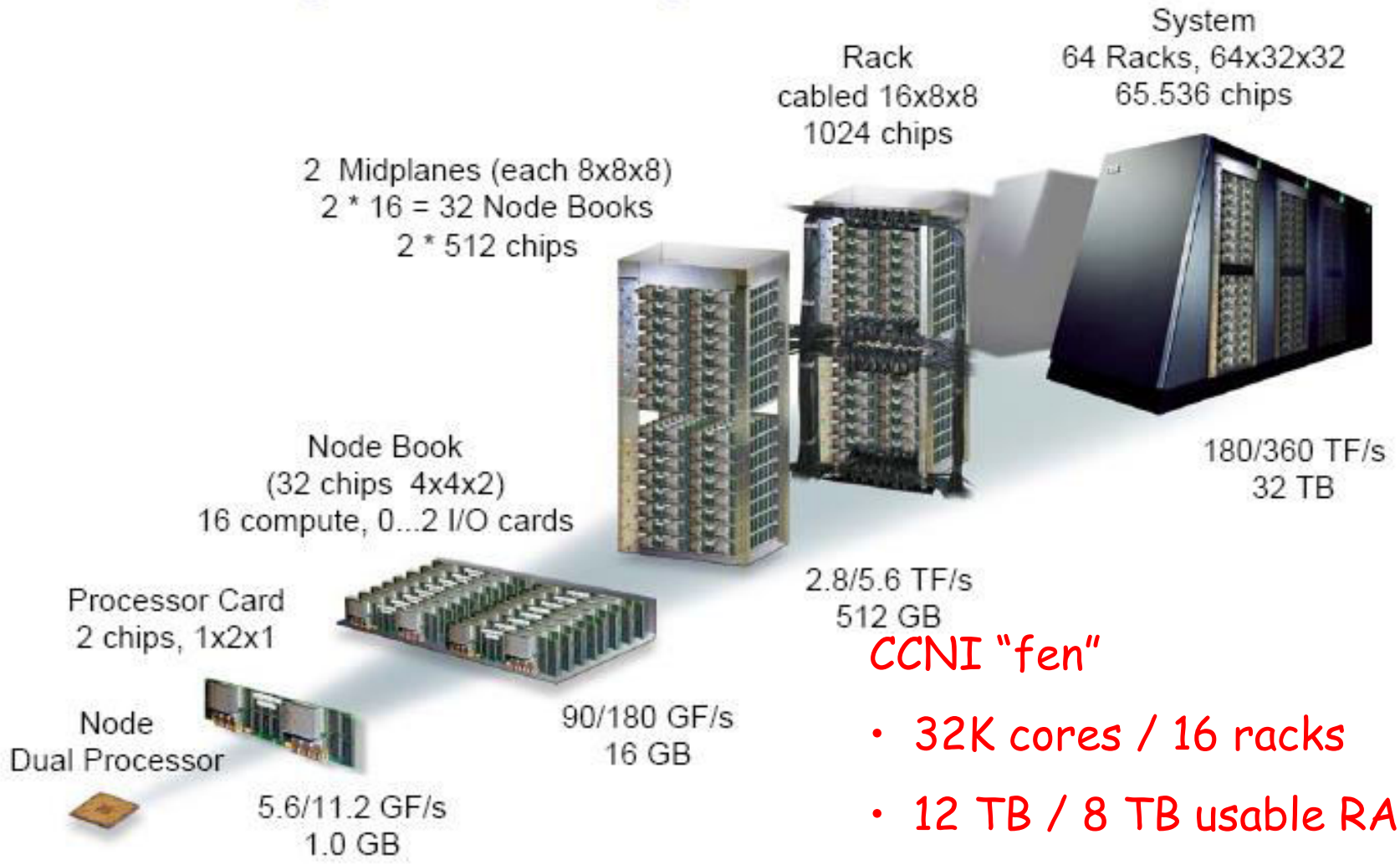
More cores per node … Kraken has 8 cores per node.

Sun/TACC Ranger

- 3,936 Nodes with 16 cores each
  - 62,976 cores total
- 123TB RAM / 32 GB RAM per node
- 1.73 PB of disk
- Network is a FULL-CLOS 7 stage IB network.
  - 2.1 μsec latency between any two nodes
- Linux CENTOS distro
- 72 I/O servers
- Lustre filesystem



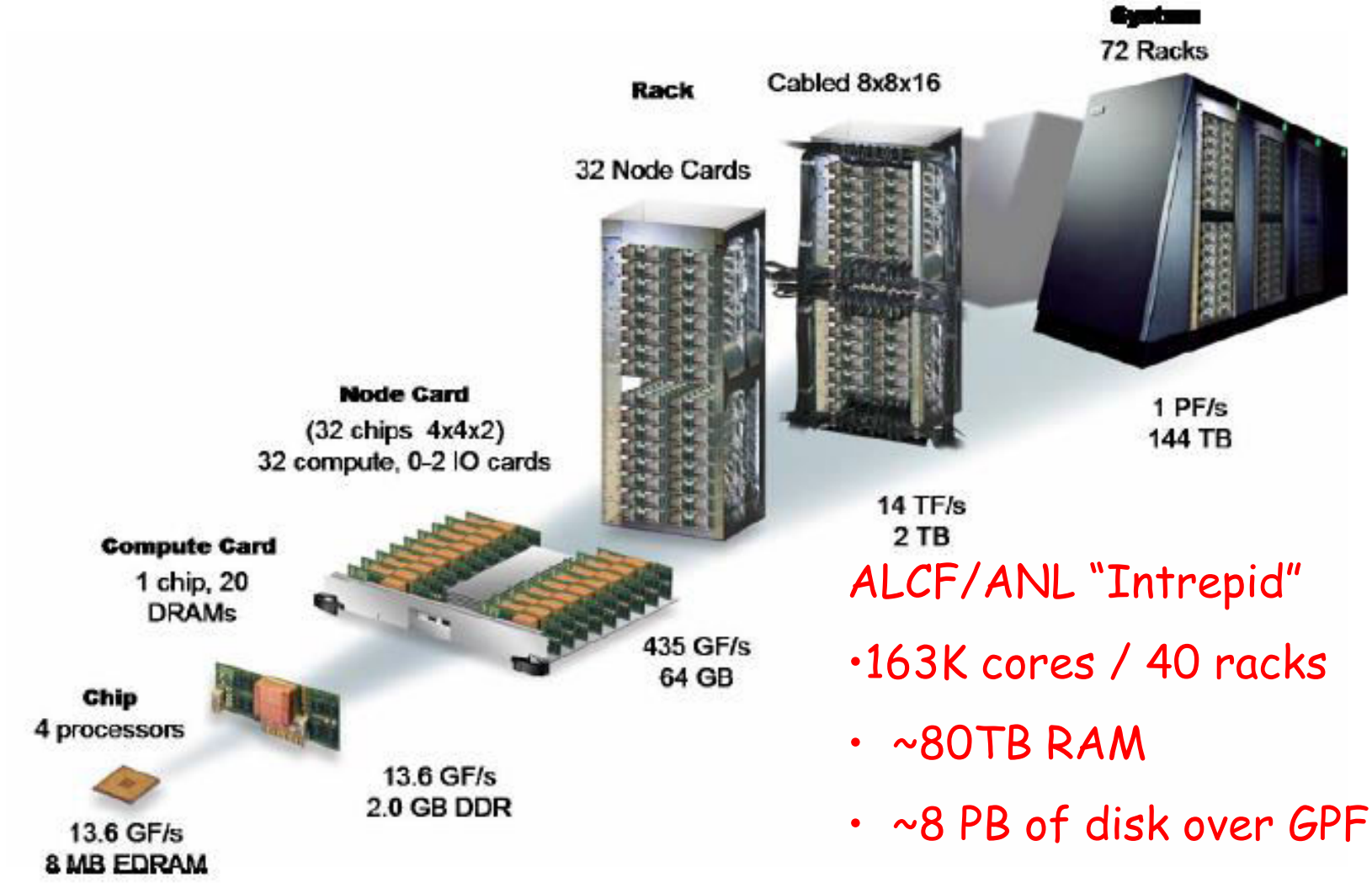
Blue Gene /L Layout



CCNI "fen"

- 32K cores / 16 racks
- 12 TB / 8 TB usable RAM
- ~1 PB of disk over GPFS
- Custom OS kernel

Blue Gene /P Layout



ALCF/ANL "Intrepid"

- 163K cores / 40 racks
- ~80TB RAM
- ~8 PB of disk over GPFS
- Custom OS kernel

Blue Gene: L vs. P

Property		BG/L	BG/P
Node Properties	Node Processors	2 * 440 PowerPC	4 * 450 PowerPC
	Processor Frequency	0.7 GHz	0.85 GHz
	Coherency	Software managed	SMP
	L1 Cache (private)	32KB / core	32KB / core
	L2 Cache (private)	14 stream prefetching	14 stream prefetching
	L3 Cache size (shared)	4 MB	8 MB
	Main Store/Node	512 MB, later 1 GB version	2 GB
	Main Store Bandwidth	5.6 GB/s (16B wide)	13.6 GB/s (2*16B wide)
	Peak Performance	5.6 GF/node	13.6 GF/node
Torus Network	Bandwidth	6*2*175MB/s = 2.1 GB/s	6*2*425MB/s = 5.1 GB/s
	Hardware Latency (Nearest Neighbor)	200 ns (32B packet) 1.6 us (256B packet)	100 ns (32B packet) 800 ns (256B packet)
	Hardware Latency (Worst Case)	6.4 us (64 hops)	3.0 us (64 hops)
Collective Network	Bandwidth	2*350MB/s = 700 MB/s	2*0.85GB/s = 1.7 GB/s
	Hardware Latency (Round Trip Worst Case)	5.0 us	3.0 us
System Properties	Peak Performance	360 TF (64k nodes)	1 PF (72k nodes)
	Total Power	1,7 MW	TBD (about 2,3 MW)

Jaguar – #1 on T500 @ 2.33 PF

Jaguar is a Cray XT5 system with over 255K processing cores…

Jaguar cont.

Jaguar Specifications	XT5	XT4
Peak Teraflops	2,332	263
Six-Core AMD Opterons	37,376	
Quad-Core AMD Opterons		7,832
AMD Opteron Cores	224,256	31,328
Compute Nodes	18,688	7,832
Memory (TB)	299	62
Memory Bandwidth (GB/s)	478	100
Disk Space (TB)	10,000	750
Interconnect Bandwidth	374	157
Floor Space (ft^2)	4,352	1,344
Cooling Technology	Liquid	Air

Concluding Remarks

- Cost/performance is improving
  - Due to underlying technology development
- Hierarchical layers of abstraction
  - In both hardware and software
- Instruction set architecture
  - The hardware/software interface
- Execution time: the best performance measure
- Power is a limiting factor
  - Use parallelism to improve performance

What does the Future…

- Blue Waters – IBM @ NCSA
  - NSF Track 1 Supercomputer System
  - $208 million over nearly 5 years
  - > 200,000s of CPUs capable of 10 PF in performance
  - > 1 PB of RAM
  - > 10 PB of disk
  - Goal: many applications will sustain 1 PF
  - To be online in 2011

And by 2012…

- Blue Gene /Q "Sequoia"
  - 20 Petaflops
  - Low power: 3 GFlops per watt
    - Total power draw of only ~6 megawatts!
  - 1.6 million cores in 98,304 compute nodes
  - 1.6 petabytes of RAM