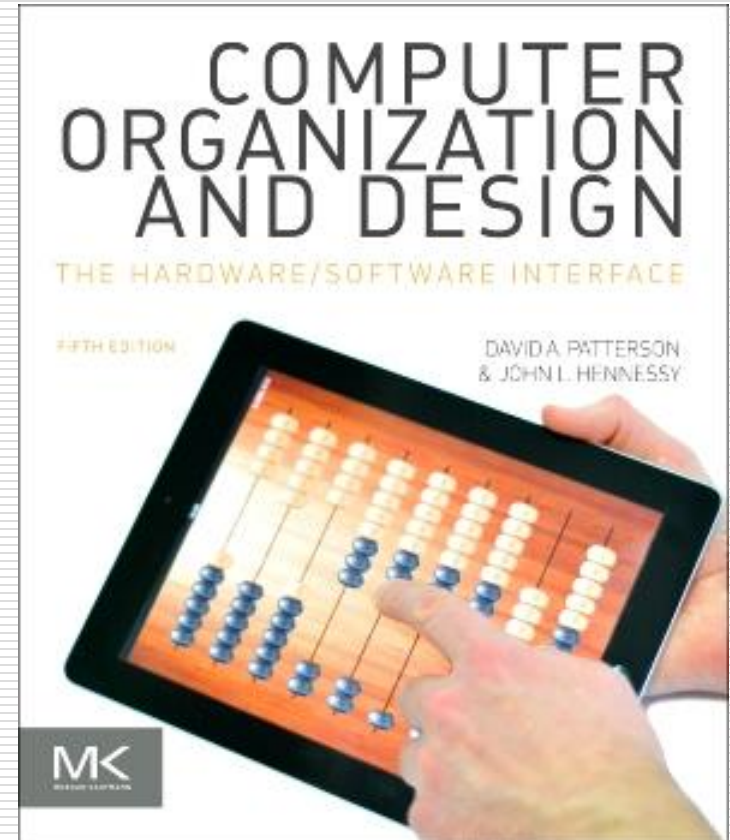


Computer Organization and Structure

Bing-Yu Chen
National Taiwan University

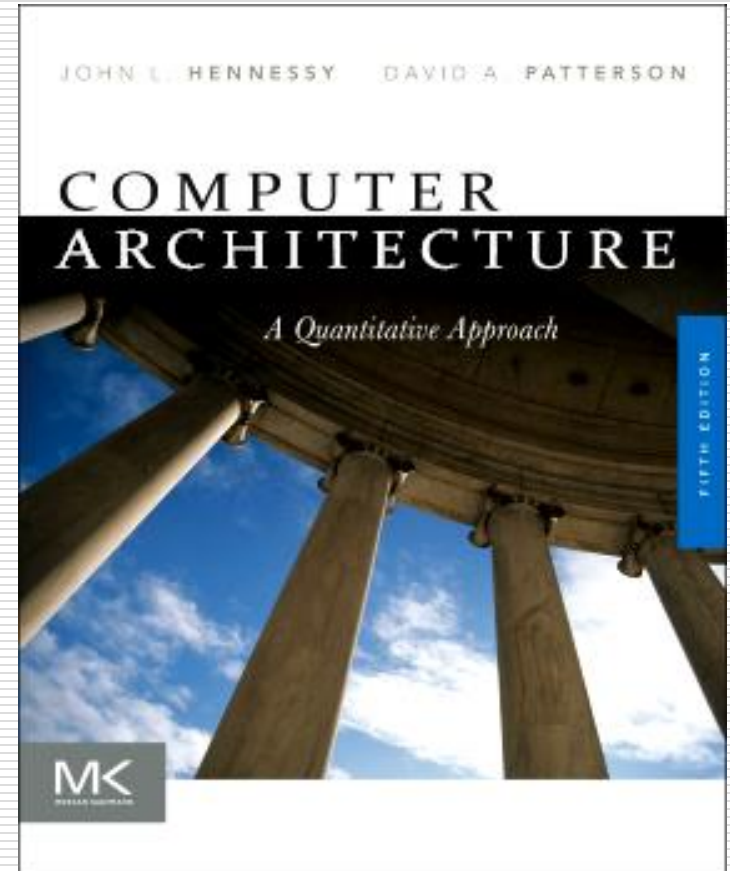
Textbook

- D. A. Patterson,
J. L. Hennessy.
*Computer
Organization & Design:
The Hardware/Software
Interface, 5th. ed.*,
Morgan Kaufmann, 2013.



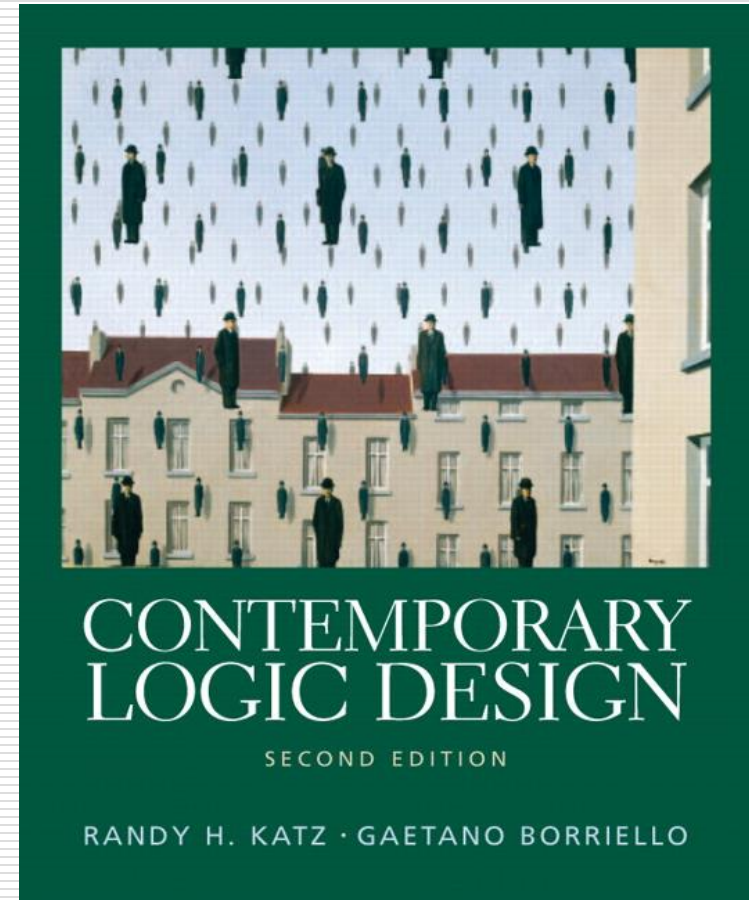
Reference

- J. L. Hennessy,
D. A. Patterson.
*Computer Architecture:
A Quantitative Approach,
5th. ed.*,
Morgan Kaufmann, 2011.



Reference

- R. H. Katz,
G. Borriello.
*Contemporary
Logic Design, 2nd ed.*,
Prentice Hall, 2004.



Pre-requirements

- Binary Digital Systems
 - Introduction to Computer

Requirements

- Participants
- Homework
 - maybe five or six times
with some small programs
- Examinations
 - twice

Why and What is the course ?

- This is the only **Computer Hardware** related course in IM department.

- The contents will cover
 - Logic Design
 - one or two weeks (maybe)
 - Assembly Language
 - two or three weeks (maybe)
 - Computer Architecture
 - the rest weeks

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

*doubling "every 18 months"

Classes of Computers

□ Personal 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

Classes of Computers

□ Supercomputers

- High-end scientific and engineering calculations
- Highest capability but represent a small fraction of the overall computer market

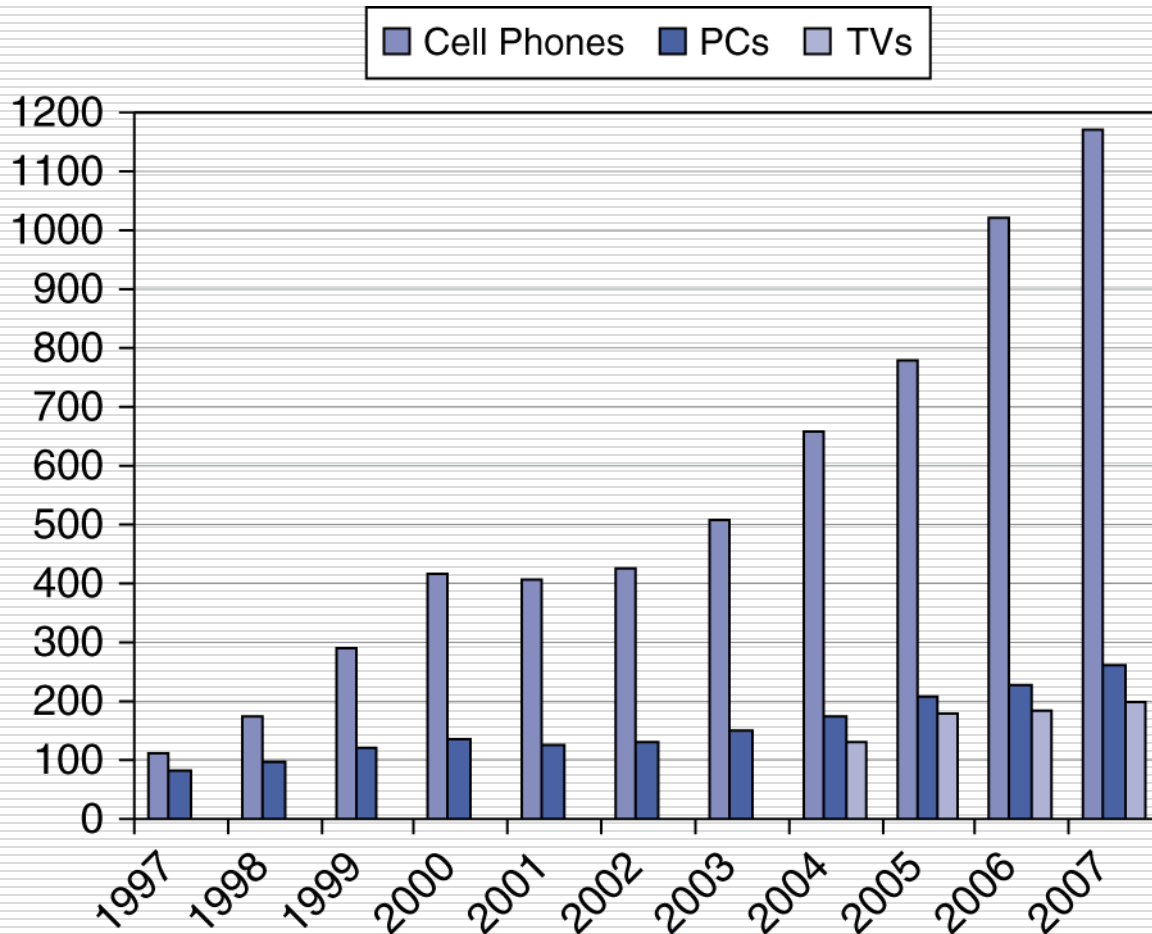
□ Embedded computers

- Hidden as components of systems
- Stringent power/performance/cost constraints

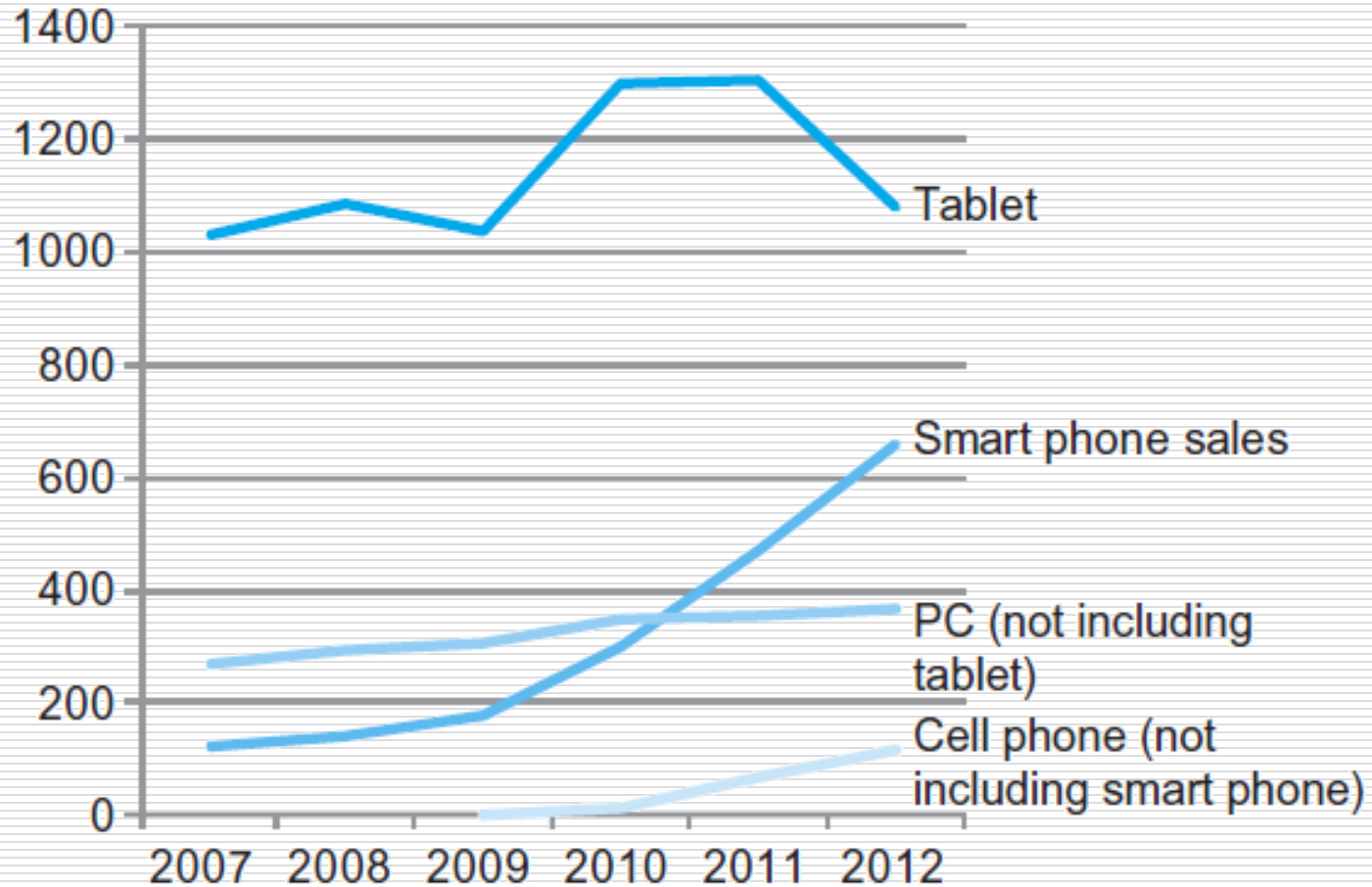
History of Intel® CPU

- 1978 8086 / 8088 5-10 MHz
- 1982 80286 6-25 MHz
- 1985 Intel386™ 16-33 MHz
- 1989 Intel486™ DX 25-50 MHz
- 1993 Pentium® 60-233 MHz
- 1997 Pentium® II 233-450 MHz
- 1999 Pentium® III 450M-1.4 GHz
- 2000 Pentium® 4 1.4-3.8 GHz

The Processor Market



The PostPC Era



The PostPC Era

- Personal Mobile Device (PMD)
 - Battery operated
 - Connects to the Internet
 - Hundreds of dollars
 - Smart phones, tablets, electronic glasses
- Cloud computing
 - Warehouse Scale Computers (WSC)
 - Software as a Service (SaaS)
 - Portion of software run on a PMD and a portion run in the Cloud
 - Amazon and Google

Understanding Performance

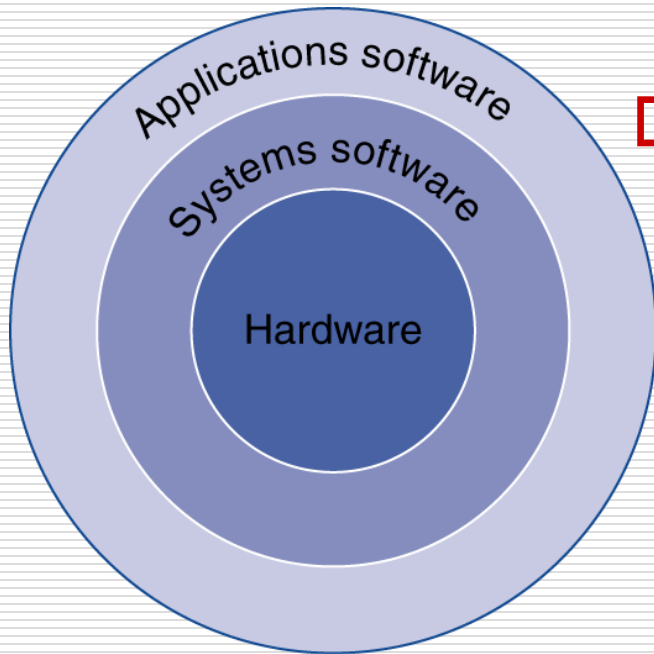
- Algorithm
 - Determines number of operations executed
- Programming language, compiler, architecture
 - Determine number of machine instructions executed per operation
- Processor and memory system
 - Determine how fast instructions are executed
- I/O system (including OS)
 - Determines how fast I/O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

Eight Great Ideas

- ❑ Design for **Moore's Law**
- ❑ Use **abstraction** to simplify design
- ❑ Make the **common case fast**
- ❑ Performance via **parallelism**
- ❑ Performance via **pipelining**
- ❑ Performance via **prediction**
- ❑ **Hierarchy** of memories
- ❑ **Dependability** via redundancy



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

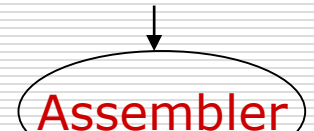
High-level language program (in C)

```
swap(int v[],int k)
{int temp;
  temp=v[k];
  v[k]=v[k+1];
  v[k+1]=temp;
}
```



Assembly language program (for MIPS)

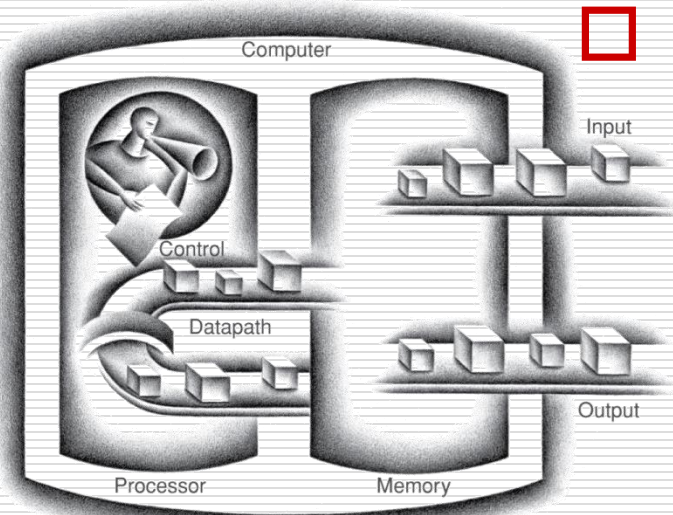
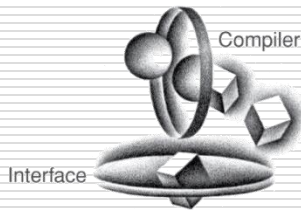
```
swap:
  muli $2, $5, 4
  add  $2, $4, $2
  lw   $15, 0($2)
  lw   $16, 4($2)
  sw   $16, 0($2)
  sw   $15, 4($2)
  jr   $31
```



Binary machine language program (for MIPS)

```
00000000101000010000000000011000
00000000000110000001100000100001
10001100011000100000000000000000
10001100111100100000000000000100
10101100111100100000000000000000
10101100011000100000000000000100
0000001111100000000000000001000
```

Components of a Computer



- 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

output
device



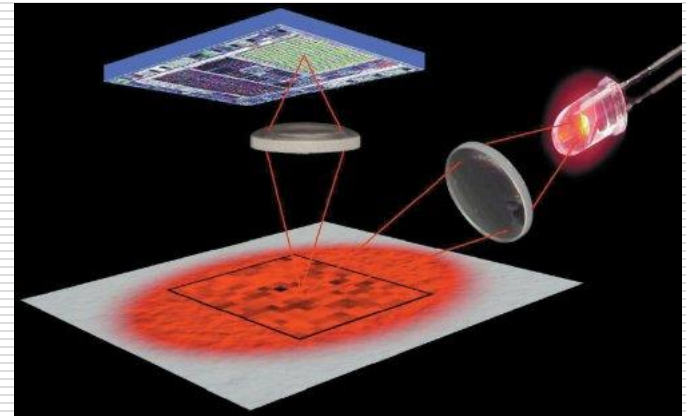
network
cable

input
device

input
device

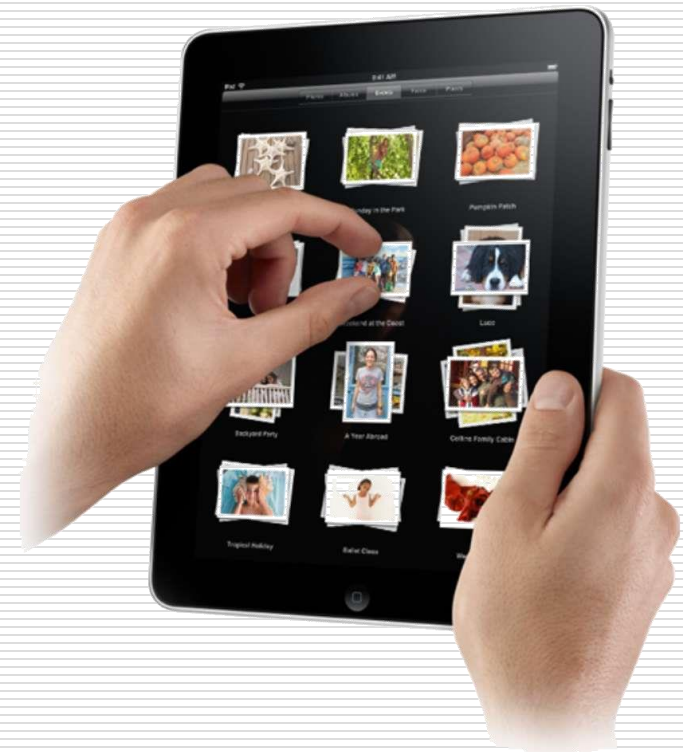
Anatomy of a Mouse

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



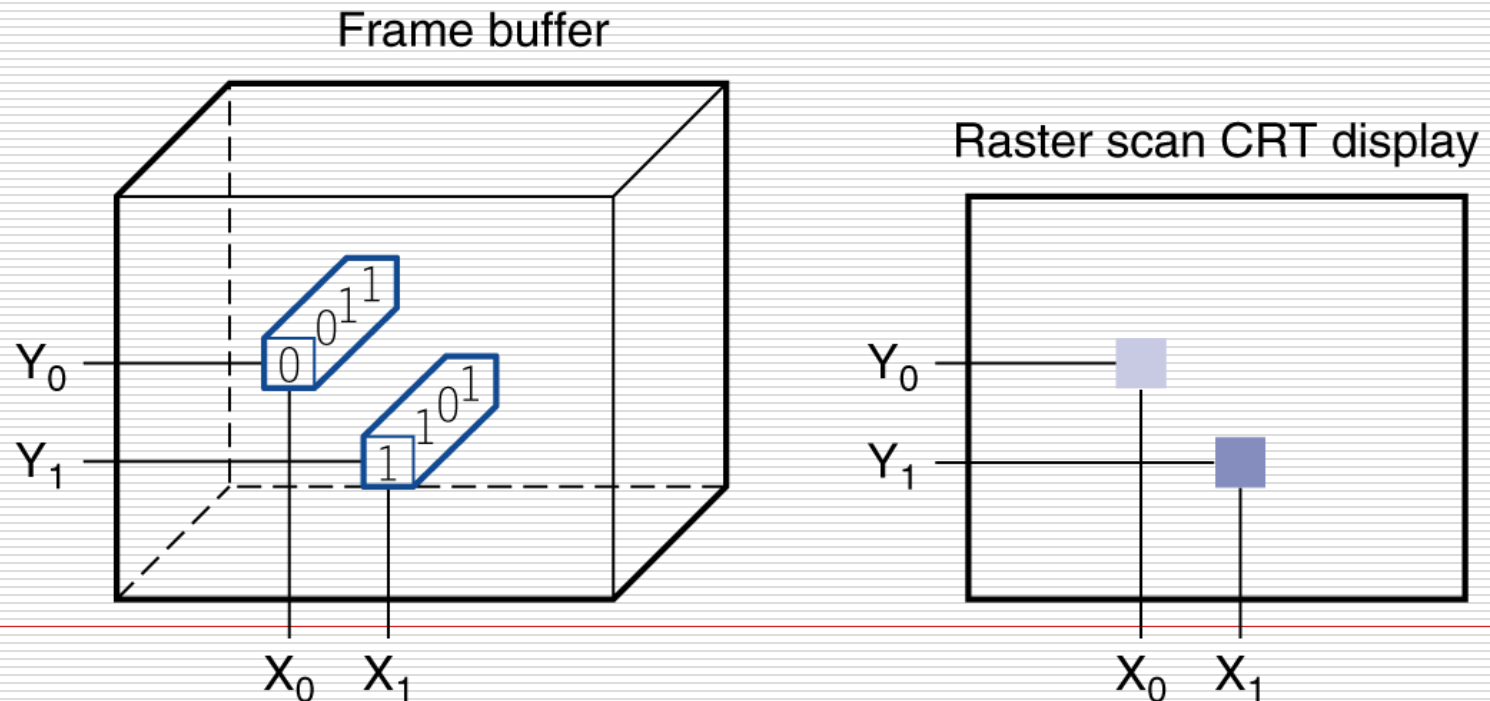
Touchscreen

- ❑ PostPC device
- ❑ Supersedes keyboard and mouse
- ❑ Resistive and Capacitive types
 - Most tablets, smart phones use capacitive
 - Capacitive allows multiple touches simultaneously

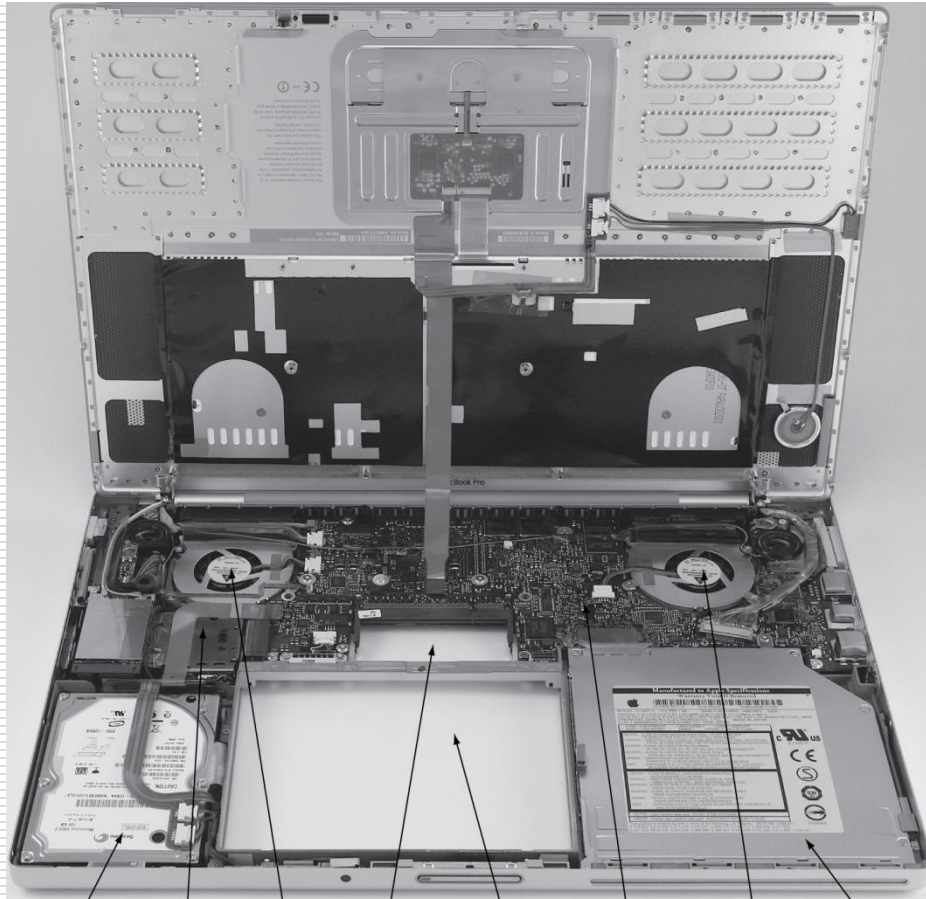


Through the Looking Glass

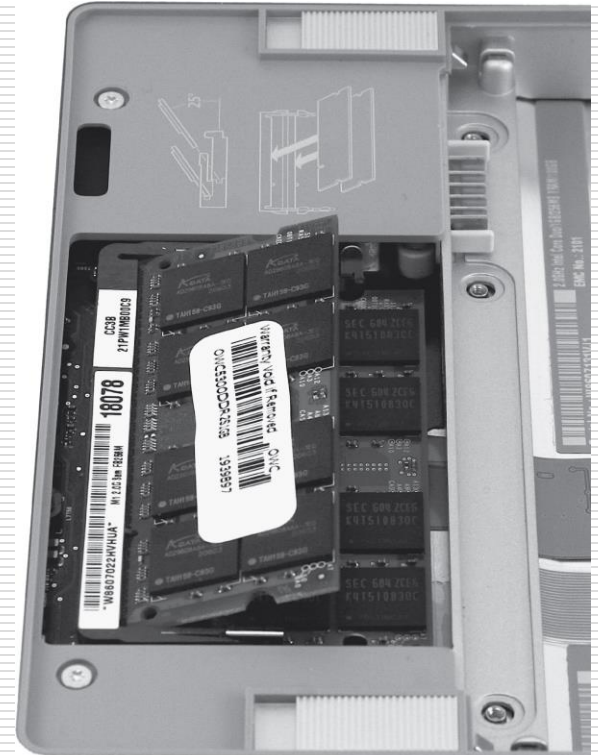
- LCD screen: picture elements (pixels)
 - Mirrors content of frame buffer memory



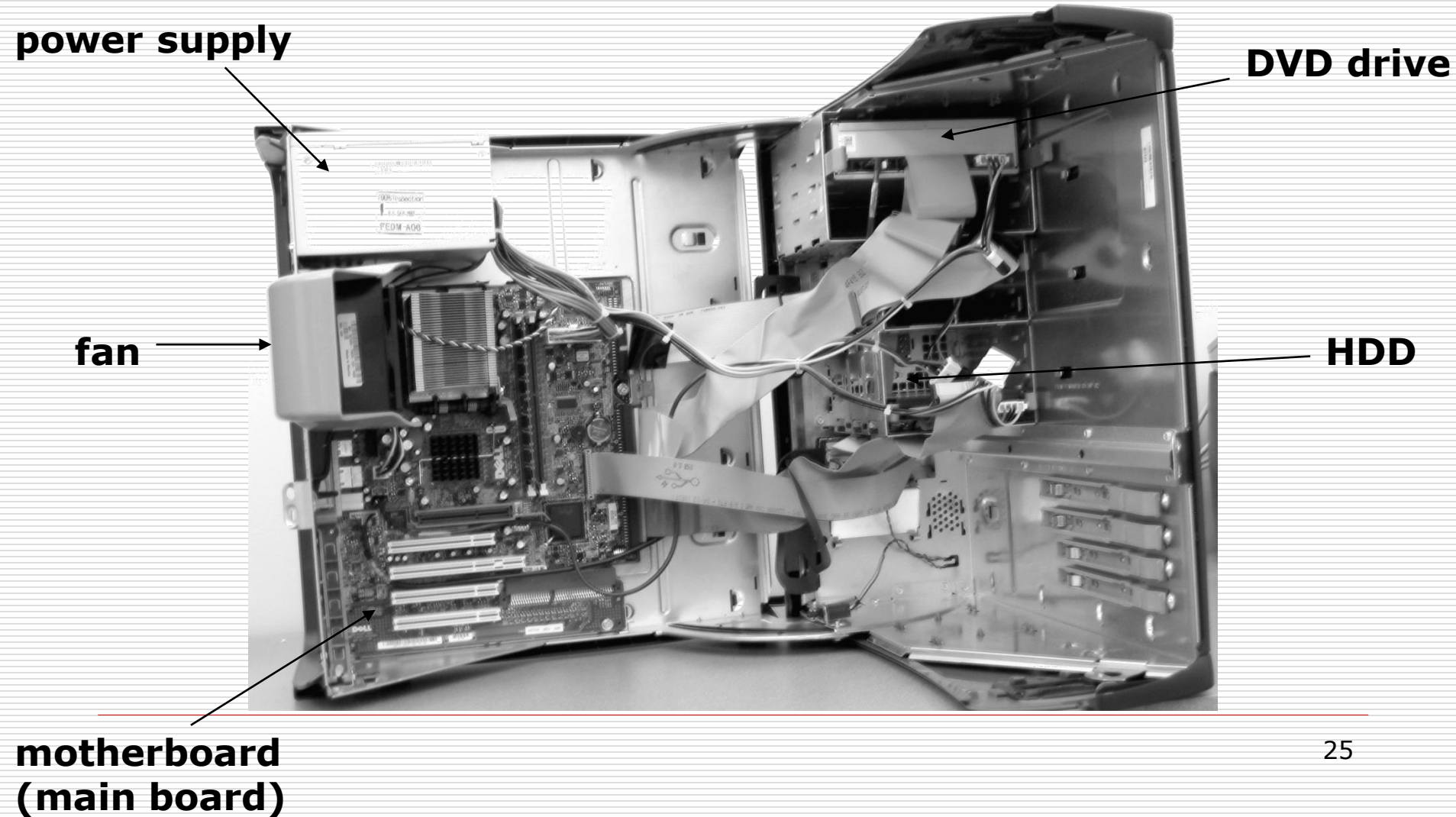
Opening the Box



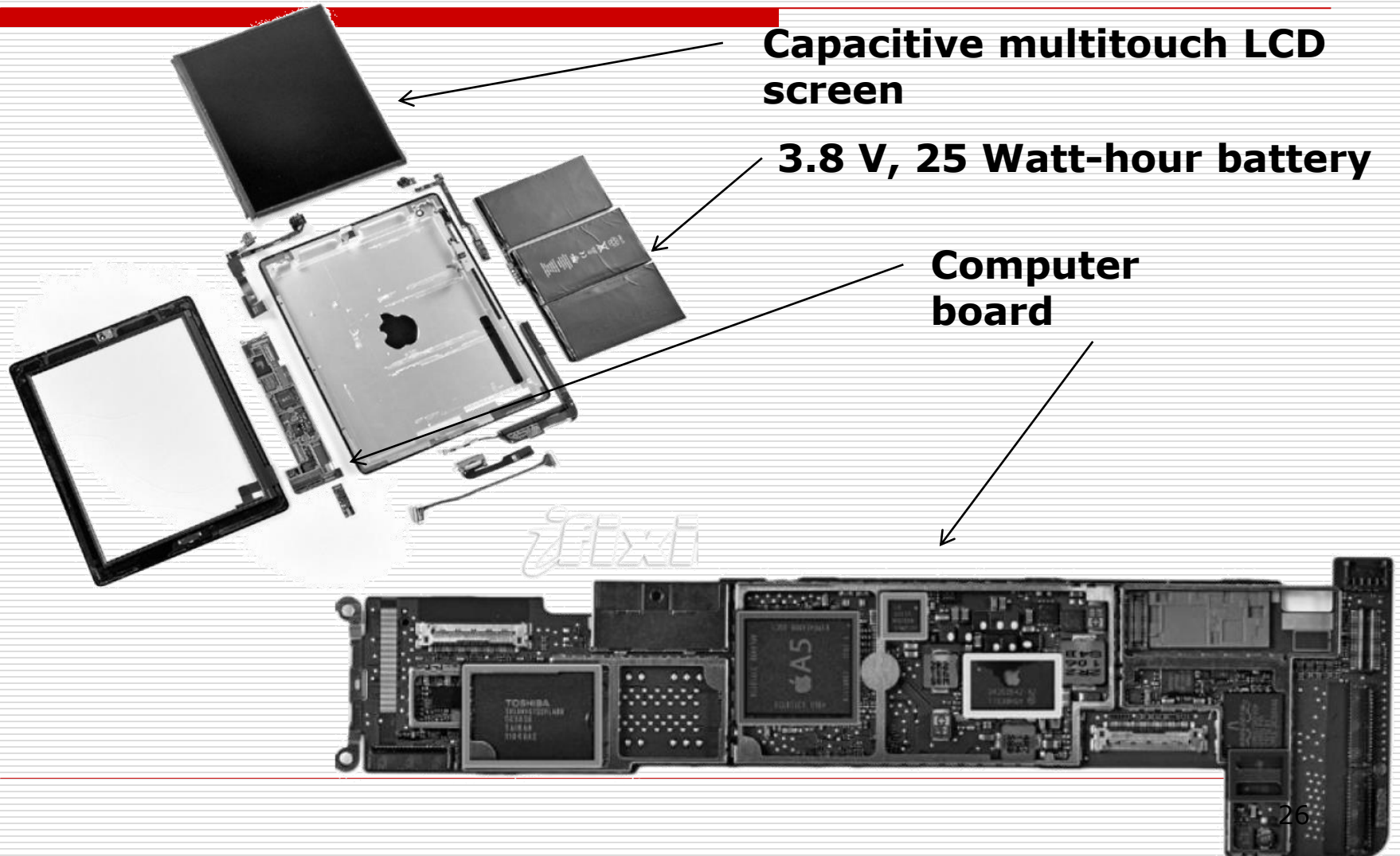
Hard drive Processor Fan with cover Spot for memory DIMMs Spot for battery Motherboard Fan with cover DVD drive cover



Opening the Box



Opening the Box

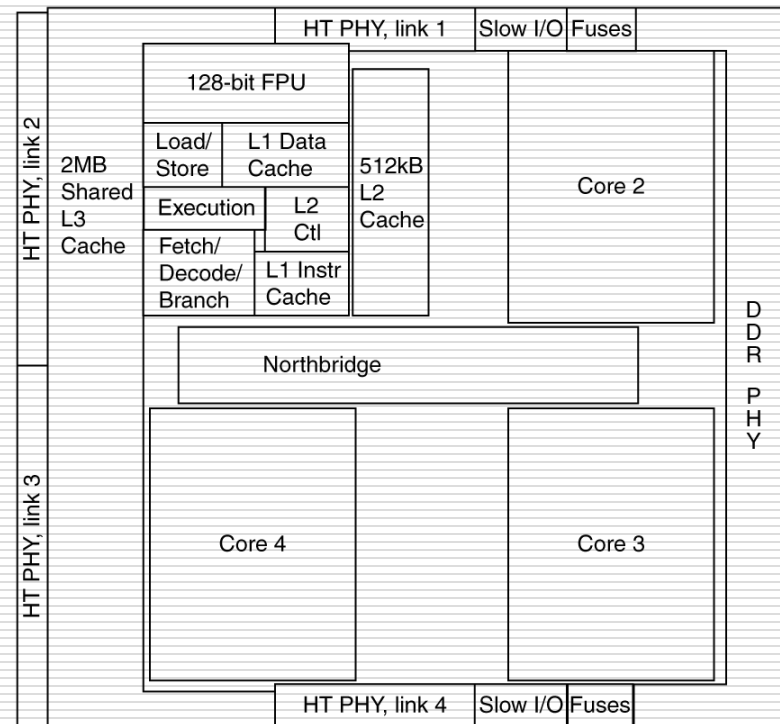
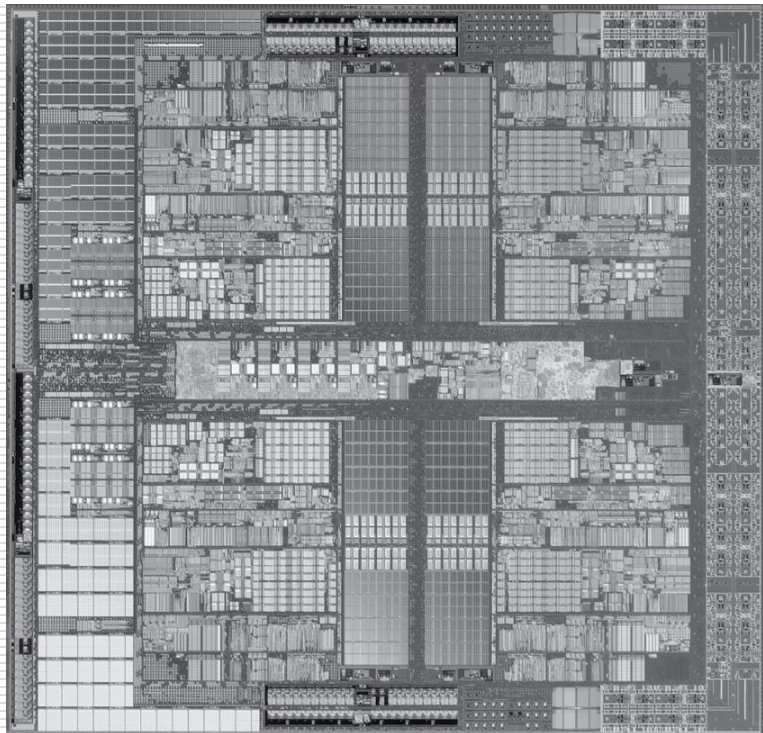


Inside the Processor (CPU)

- 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



Inside the Processor

□ Apple A5

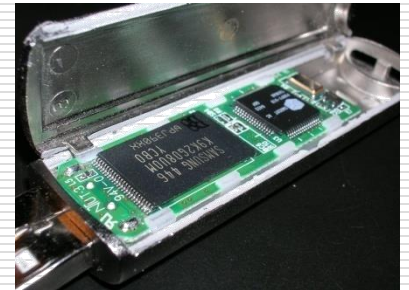


Abstractions

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

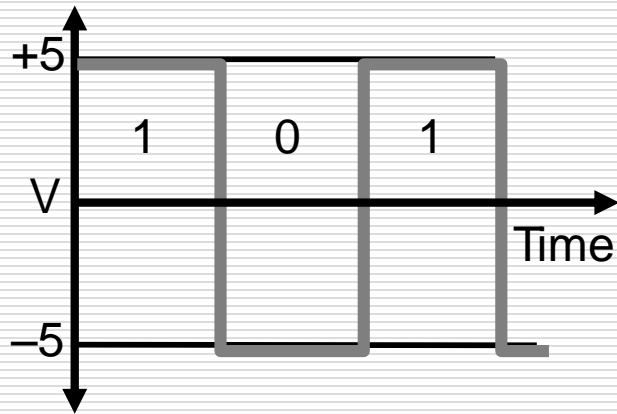


Networks

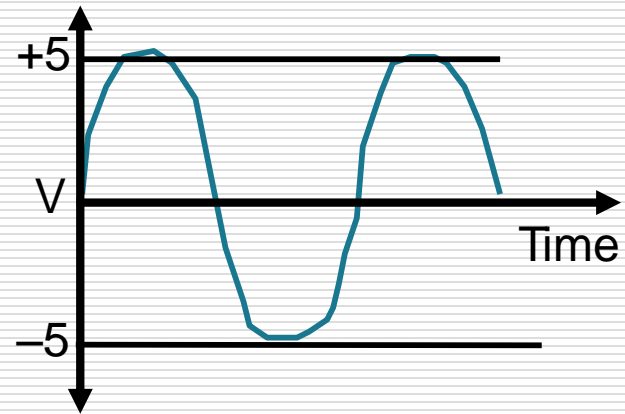
- ❑ Communication, resource sharing, nonlocal access
- ❑ Local area network (LAN): Ethernet
- ❑ Wide area network (WAN): the Internet
- ❑ Wireless network: WiFi, Bluetooth



Digital Systems



Digital:
only assumes
discrete values



Analog:
values vary over
a broad range
continuously

Digital Binary Systems

- Two discrete values:

yes	on	5 volts	current flowing	magnetized North	“1”
no	off	0 volts	no current flowing	magnetized South	“0”

Advantage of Binary Systems

- rigorous mathematical foundation based on logic

- **IF** the garage door is open
- **AND** the car is running
- **THEN** the car can be backed out of the garage

Boolean Algebra & Logical Operators

- Algebra: variables, values, operations
- Values: 0 and 1
- Operations: AND, OR, NOT

X	Y	X AND Y
0	0	0
0	1	0
1	0	0
1	1	1

X	Y	X OR Y
0	0	0
0	1	1
1	0	1
1	1	1

X	NOT X
0	1
1	0

Combinational vs. Sequential Logic

- Combinational logic: without a memory
 - no feedback among inputs and outputs
 - outputs are a pure function of the inputs

- Sequential logic: with a memory
 - inputs and outputs overlap
 - outputs depend on inputs and the entire history of execution
 - ex. add in elementary school

Synchronous vs. Asynchronous System

□ Synchronous system

- period reference signal, the clock, causes the storage elements to accept new values and to change state

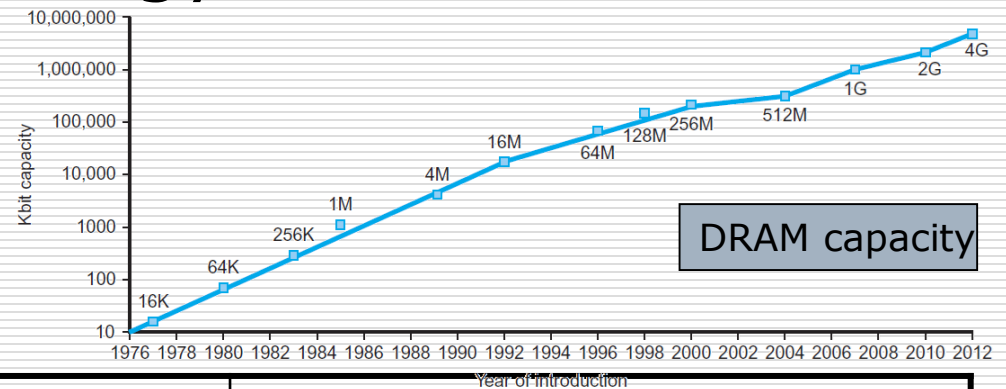
□ Asynchronous system

- no single indication of when to change state

Technology Trends

□ Electronics technology continues to evolve

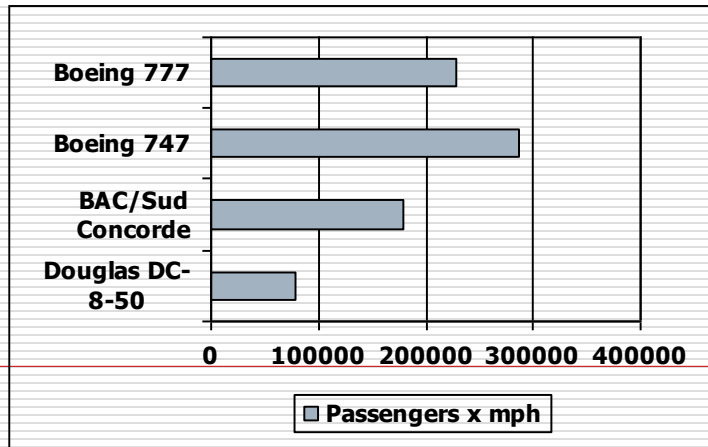
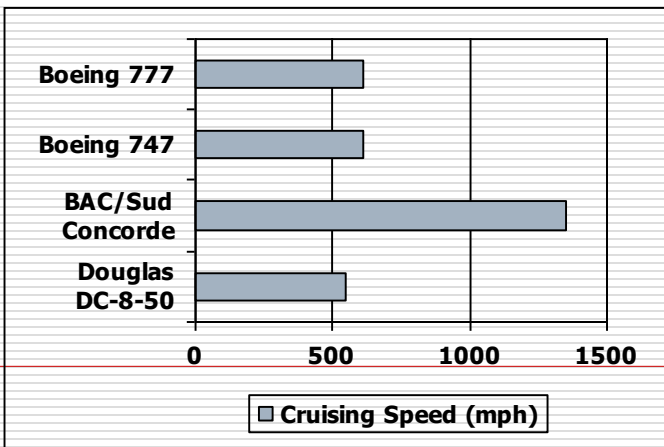
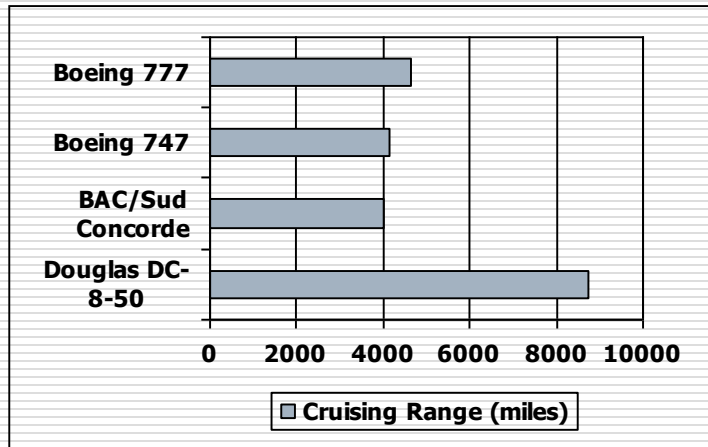
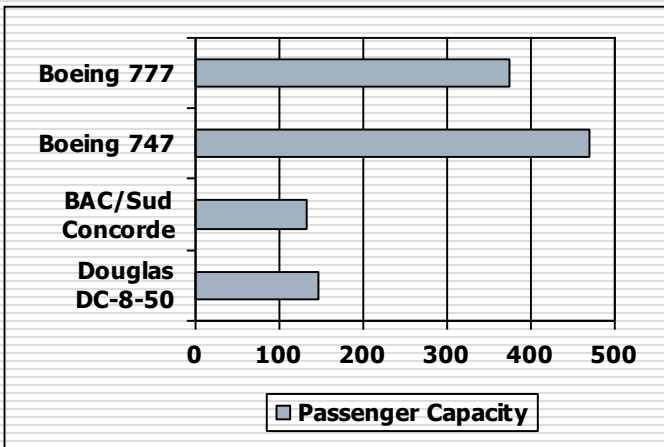
- Increased capacity and performance
- Reduced cost



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
2013	Ultra large scale IC	250,000,000,000

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 = $1/\text{Execution Time}$
- “X is n time faster than Y”

$$\begin{aligned} & \text{Performance}_X / \text{Performance}_Y \\ &= \text{Execution Time}_Y / \text{Execution Time}_X = n \end{aligned}$$

- Example: time taken to run a program

- 10s on A, 15s on B

- how much faster is A than B ? $\Rightarrow \frac{15}{10} = 1.5$

Measuring Execution Time

□ Elapsed time

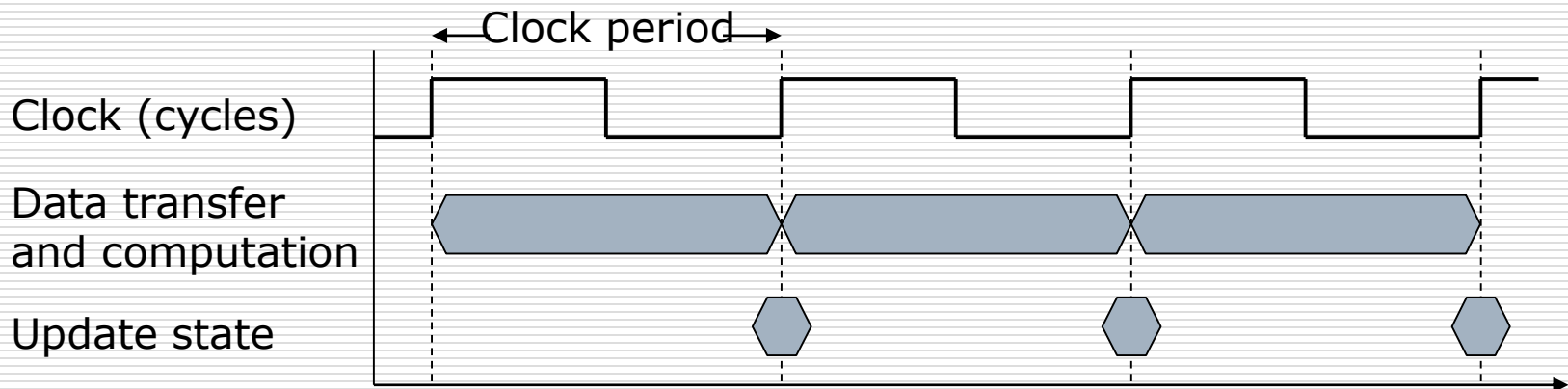
- Total response time, including all aspects
 - Processing, I/O, OS overhead, idle time
- Determines system performance

□ 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 constant-rate clock



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

CPU Time

$$\begin{aligned}\text{CPU Time} &= \text{CPU Clock Cycles} \times \text{Clock Cycle Time} \\ &= \frac{\text{CPU Clock Cycles}}{\text{Clock Rate}}\end{aligned}$$

- 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 $1.2 \times$ clock cycles
- How fast must Computer B clock be?

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

$$\begin{aligned}\text{Clock Cycles}_A &= \text{CPU Time}_A \times \text{Clock Rate}_A \\ &= 10\text{s} \times 2\text{GHz} = 20 \times 10^9\end{aligned}$$

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

Instruction Count and CPI

Clock Cycles = Instruction Count \times Cycles per Instruction

CPU Time = Instruction Count \times CPI \times 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?

$$\begin{aligned}\text{CPU Time}_A &= \text{Instruction Count} \times \text{CPI}_A \times \text{Cycle Time}_A \\ &= I \times 2.0 \times 250\text{ps} = I \times 500\text{ps} \leftarrow \text{A is faster...}\end{aligned}$$

$$\begin{aligned}\text{CPU Time}_B &= \text{Instruction Count} \times \text{CPI}_B \times \text{Cycle Time}_B \\ &= I \times 1.2 \times 500\text{ps} = I \times 600\text{ps}\end{aligned}$$

$$\frac{\text{CPU Time}_B}{\text{CPU Time}_A} = \frac{I \times 600\text{ps}}{I \times 500\text{ps}} = 1.2 \leftarrow \text{...by this much}$$

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

Performance Summary

$$\text{CPU Time} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Clock Cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Clock Cycle}}$$

- 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

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

$$\begin{aligned} \text{MIPS} &= \frac{\text{Instruction Count}}{\text{Execution Time} \times 10^6} \\ &= \frac{\text{Instruction Count}}{\frac{\text{Instruction Count} \times \text{CPI}}{\text{Clock Rate}} \times 10^6} = \frac{\text{Clock Rate}}{\text{CPI} \times 10^6} \end{aligned}$$

- CPI varies between programs on a given CPU

Amdahl's Law

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

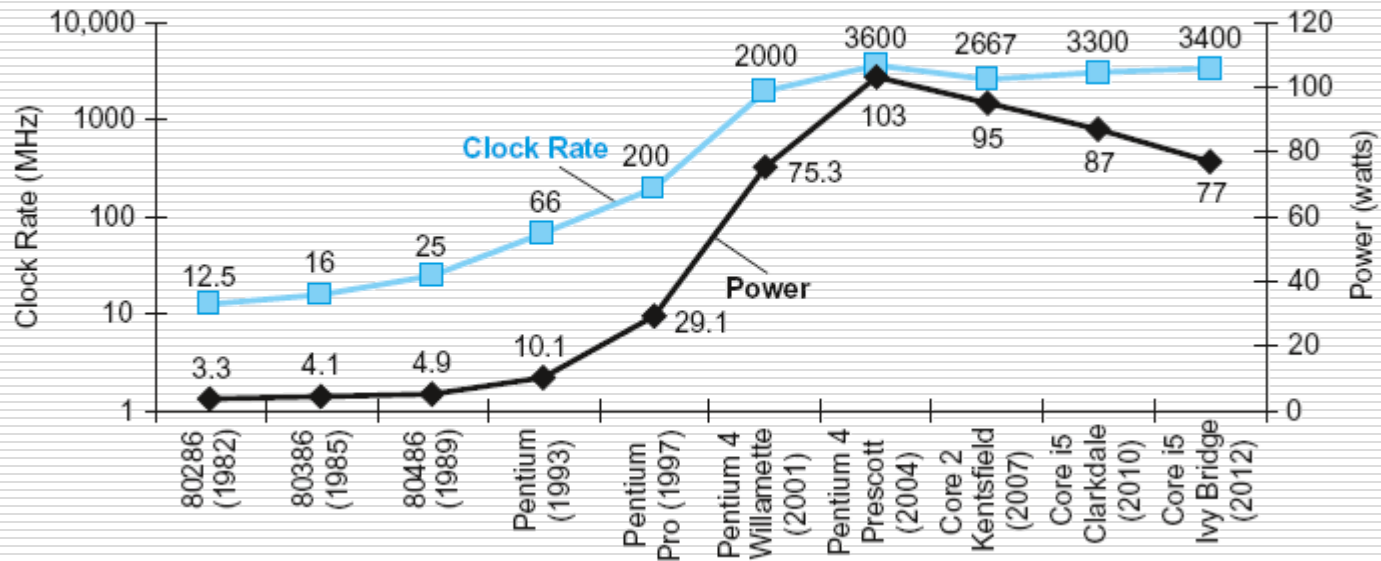
$$T_{\text{improved}} = \frac{T_{\text{affected}}}{\text{improvement factor}} + T_{\text{unaffected}}$$

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

$$20 = \frac{80}{n} + 20 \longleftarrow \text{Can't be done!}$$

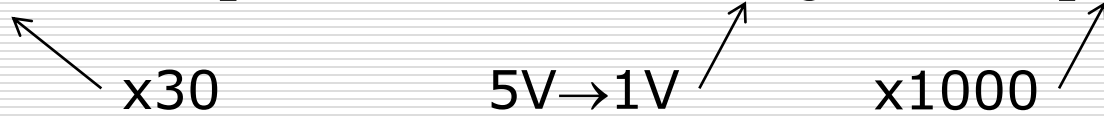
- Corollary: make the common case fast
-

Power Trends



□ In CMOS IC technology

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



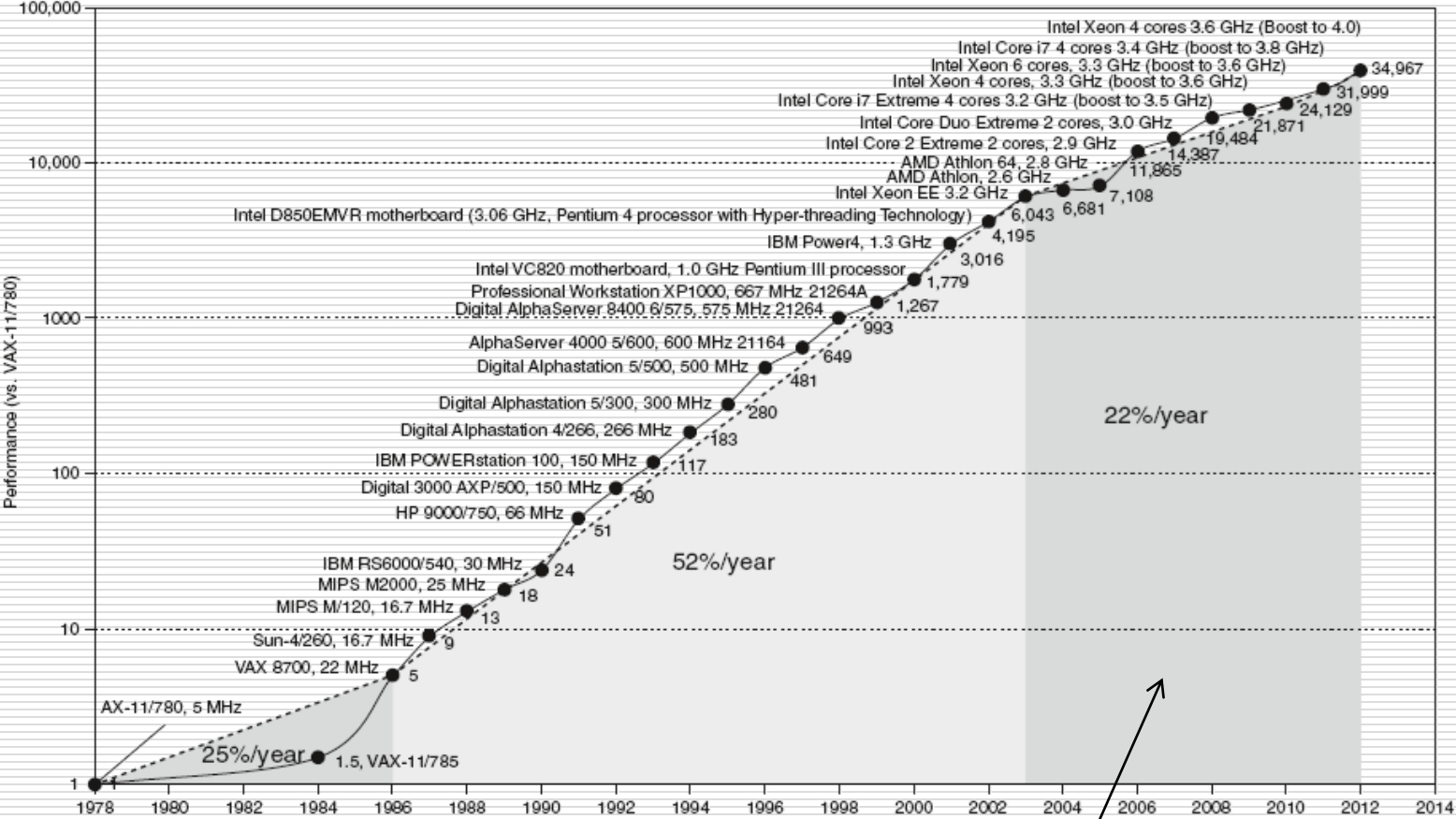
Reducing Power

- 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

Low Power at Idle

- Look back at i7 power benchmark
 - At 100% load: 258W
 - At 50% load: 170W (66%)
 - At 10% load: 121W (47%)
- 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