Lecture 1: EITF20 Computer Architecture

Anders Ardö

EIT – Electrical and Information Technology, Lund University

November 5, 2014

Lecture 1 agenda

Chapter 1 in "Computer Architecture"

Outline

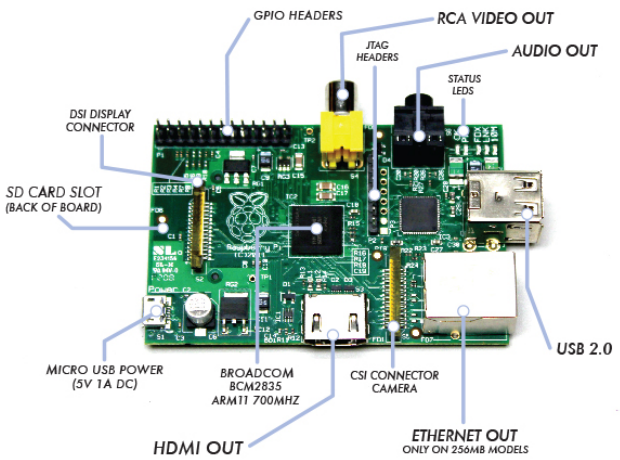
Build a computer

Desktop computer

Part	**Price (SEK)**	**Example** (2012-10-02)	**Price**
Case	400 - 1000	Silencio 450 ATX	490
Power supply	300 - 800	Corsair CX 430W	399
Motherboard	600 - 1400	Gigabyte ATX	790
CPU	500 - 2000	Intel Core I7	2490
Memory	300 - 800	Corsair 2 x 8GB	698
Disk	500 - 1000	Intel 330 120GB SSD	790
DVD/Blue-ray	200 - 500	Asus blu-ray combo	590
Graphics	0 - 1500		-
Sound, net, ...	0 - 1000		-
Keyboard, mouse, cables, ...	300 - 500		?

$\sum$ = 3000 - 5000 - 10000 SEK 6247

Buy a (simple) computer



Raspberry Pi: 250 + SD-card + Box + (Power) $\approx 500 SEK$

Classes of computers

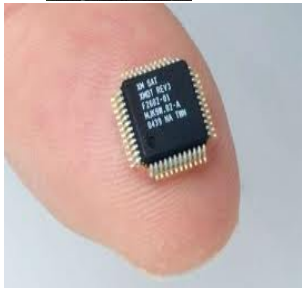
Feature	Personal mobile device (PMD)	Desktop	Server	Clusters/warehouse-scale computer	Embedded
Price of system	$100–$1000	$300–$2500	$5000–$10,000,000	$100,000–$200,000,000	$10–$100,000
Price of micro-processor	$10–$100	$50–$500	$200–$2000	$50–$250	$0.01–$100
Critical system design issues	Cost, energy, media performance, responsiveness	Price-performance, energy, graphics performance	Throughput, availability, scalability, energy	Price-performance, throughput, energy proportionality	Price, energy, application-specific performance

Figure 1.2 A summary of the five mainstream computing classes and their system characteristics. Sales in 2010 included about 1.8 billion PMDs (90% cell phones), 350 million desktop PCs, and 20 million servers. The total number of embedded processors sold was nearly 19 billion. In total, 6.1 billion ARM-technology based chips were shipped in 2010. Note the wide range in system price for servers and embedded systems, which go from USB keys to network routers. For servers, this range arises from the need for very large-scale multiprocessor systems for high-end transaction processing.

Rough time-line

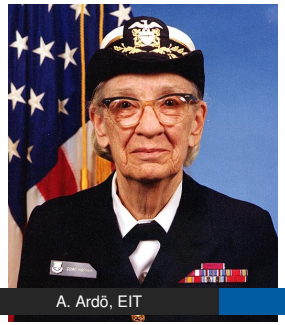
- Mid-1800 Programmable computer
  Charles Babbage (analytical engine), Ada Lovelace (programmer)
- 1940s First modern computers
  Zuse Z1 and Z3, MARK I, ENIAC, ...
- 1960s Mainframe — room
  1964 IBM System/360
- 1970s Minicomputer — rack
  1971 First microprocessor; Graphics Xerox Alto
- 1980s Desktop — pizza-box
  1977 Apple II, 1981 IBM PC
- 1990s PDA — cigarette-box
- 2000s Embedded computers — chip

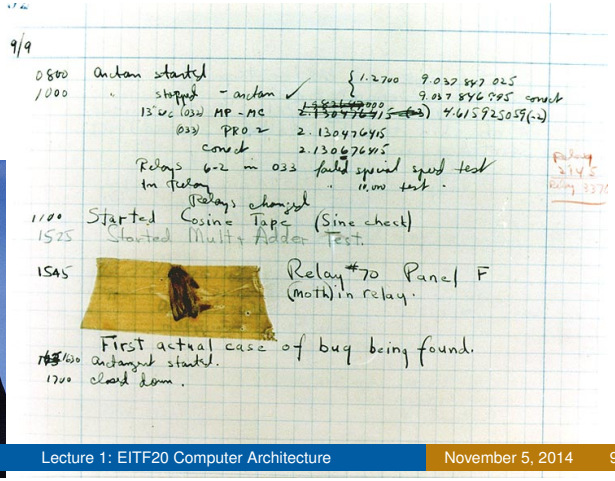
Time-line - images



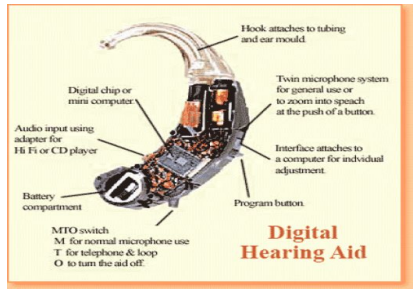
Interlude - Debugging

In 1947, Rear Admiral Grace Murray Hopper and associates was working on Mark II, the machine was experiencing problems. An investigation showed that there was a moth trapped in a relay. The operators removed the moth and affixed it to the log. The computer had been “debugged”.





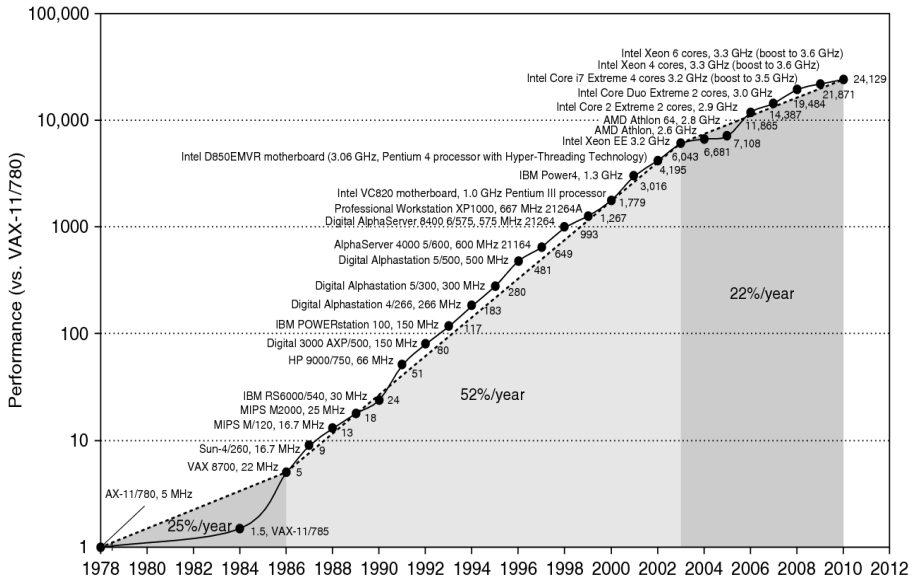
Modern (Embedded) computers



Development of (micro)processors

	Year	Transistors	Frequency	cores	Cache
Intel4004	1971	2300	108 kHz	“1”	None
Z80	1976	8500	2.5 MHz	1	None
Intel386	1985	280 000	16 MHz	1	None
Intel486	1989	1 185 000	20 - 50 MHz	1	8 kB
Pentium 4	2000	44 000 000	1 - 2 GHz	1	256 kB
Nehalem	2008	731 000 000	> 3.6 GHz	4	8 MB
Sandy Bridge	2011	995 000 000	3.8 GHz	4+	8 + 1 MB
Haswell	2013	1 860 000 000	> 3.6 GHz	6	15 + 1.5 MB
Itanium 9560	2012	3 100 000 000	2.5 GHz	8	32 + 6 MB

Performance of processors



Performance of cars compared

	Improve rate computers	Car speed	Car fuel economy
1977		160 km/h	9 km/l
1987	35 %	3216 km/h	181 km/l
2000	50 %	31136 km/h	1751 km/l

Price would drop to 25 US $ per car.

Outline

Computer Architecture

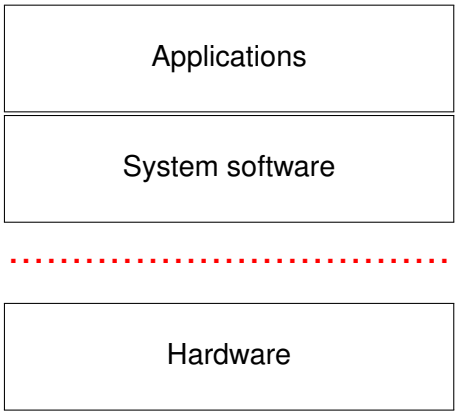
The Art of designing computers based on engineering principles and quantitative performance evaluations

Architecture:

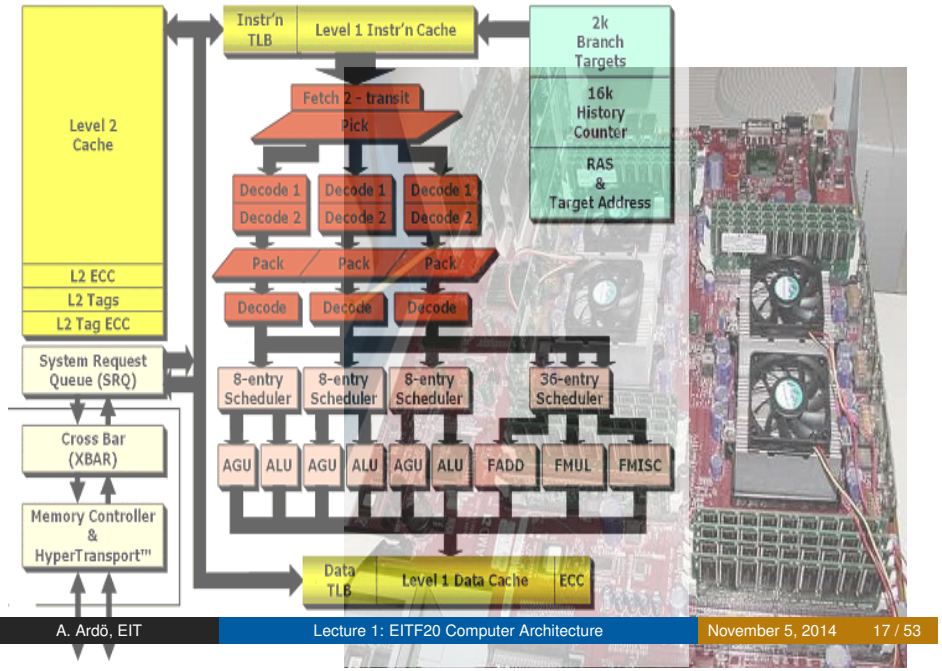
- instruction set architecture
- implementation
  - organization
  - hardware

The role of the computer architect

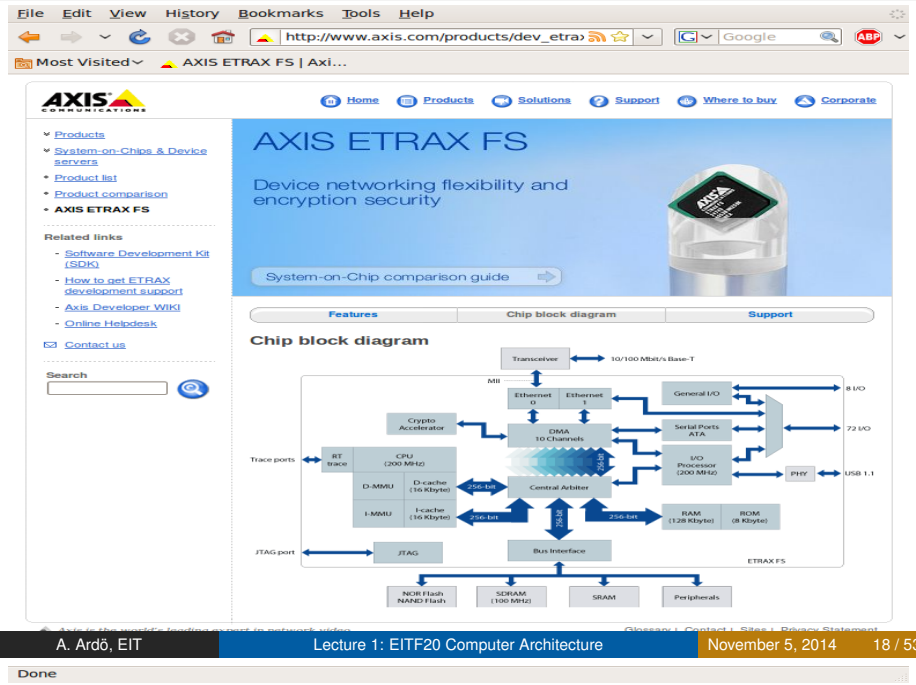
Make design decisions across the interface between hardware and software in order to meet functional and performance goals.



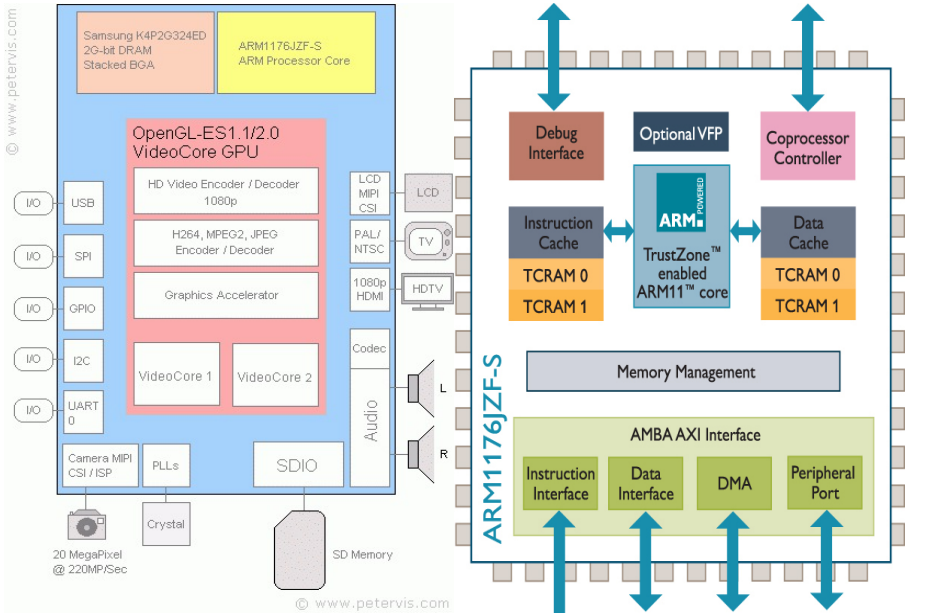
What is this?



Axis Etrax



Raspberry Pi SoC



Why computer architecture?

- Understand how to evaluate and choose
- Design your own specialized architecture
- Embedded special purpose processors
  - Axis Communications
  - Ericsson
  - Nokia
  - ARM
  - TeliaSonera
  - SAAB
  - ...

What computer architecture?

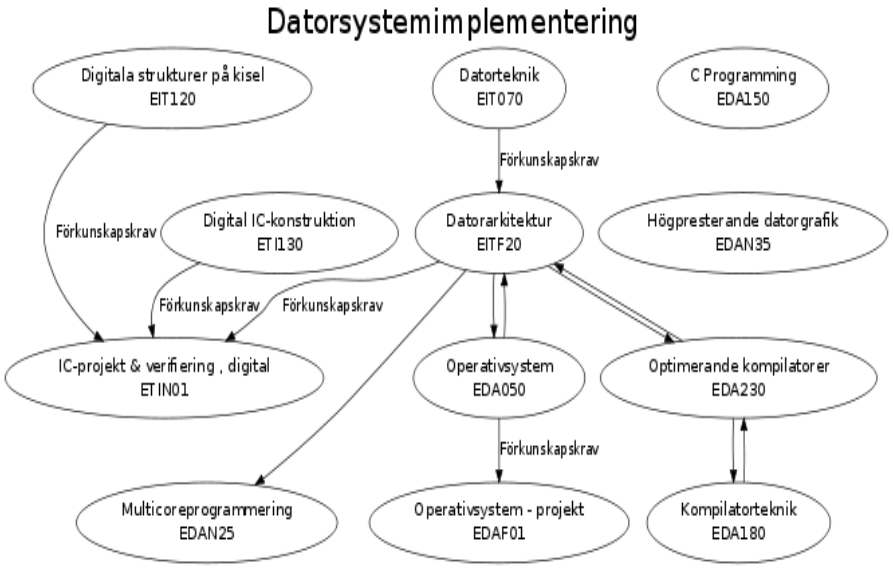
Designing

- ISA
- Organization
- Hardware

to meet

- functional requirements (applications, standards, ...)
- price
- power
- performance
- availability
- dependability

Other courses



Outline

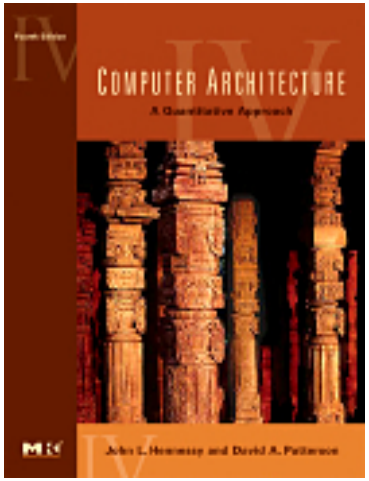
Course goals

After this course you will ...

- have a thorough knowledge about the design principles for modern computer systems
- have an understanding of the relations between
  - the design of the instruction set of a processor
  - the micro architecture of a processor
- be able to evaluate design alternatives towards design goals using quantitative evaluation methods

Course contents

Hennessy/Patterson:
Computer Architecture - A Quantitative Approach
4th ed. ISBN 978-0-12-370490-0 (5th ed. OK)



Plus selected articles
(see course-page).

- Performance, Evaluation
- Instruction Set Architecture
- Pipeline
- Memory Systems
  cache/virtual
- Storage

Course design

- Lectures
  - Covers design principles and analysis methodology
  - Read the literature before each lecture
  - Does not cover all of the literature
  - Ask many questions!
- Laboratory exercises
  - 4 labs
  - Micro-architecture simulation
  - Read manual and literature before the lab
  - Do Home Assignments before lab (or be sent home)
  - Experiment and discuss with assistants
  - Understand what you have done (or fail the exam)
- Question hours
- Individual problem solving - exercises

Course design - Exam

Online Quiz

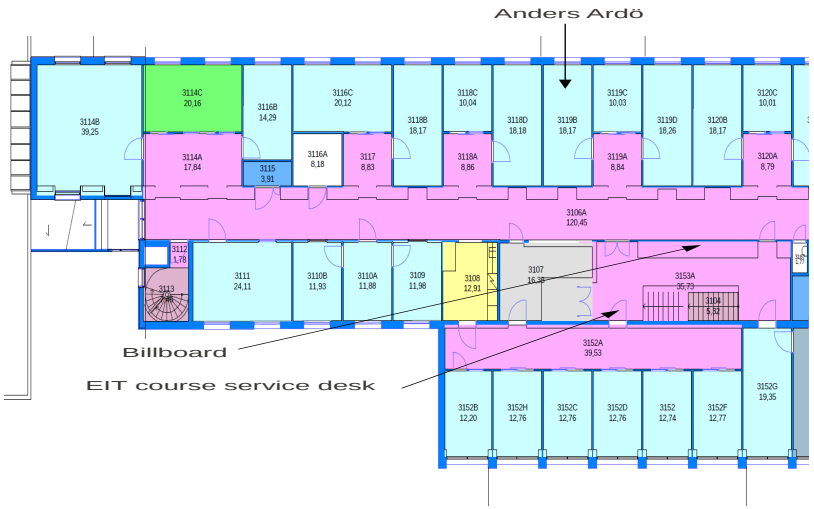
Approved labs
- **Prepare labs beforehand!!**

Written exam
- Five problems
  - problem solving
  - descriptive nature

Administrative stuff

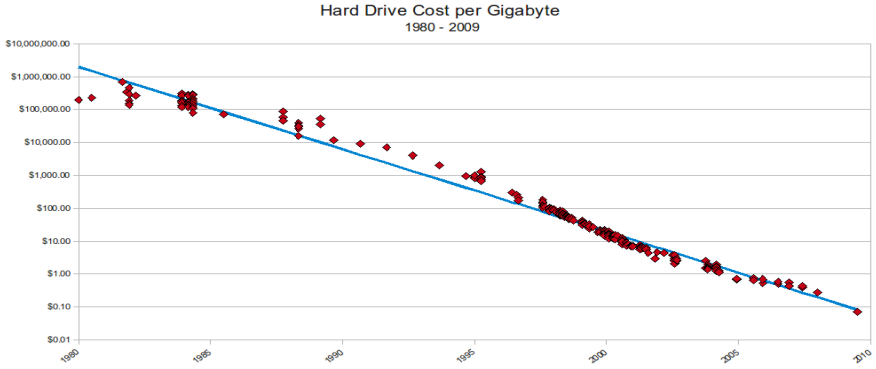
- Course home page: http://www.eit.lth.se/course/eitf20
- Anders Ardö. Room E3119b, Anders.Ardo@eit.lth.se
- EIT's Course Service Desk (studerandeexpedition)
  - Course secretary: Doris Glöck
  - room 3152B, the third floor of the E building, staircase A (north)
  - e-mail: kursexp@eit.lth.se

Map EIT - E-huset north end, floor 3



Outline

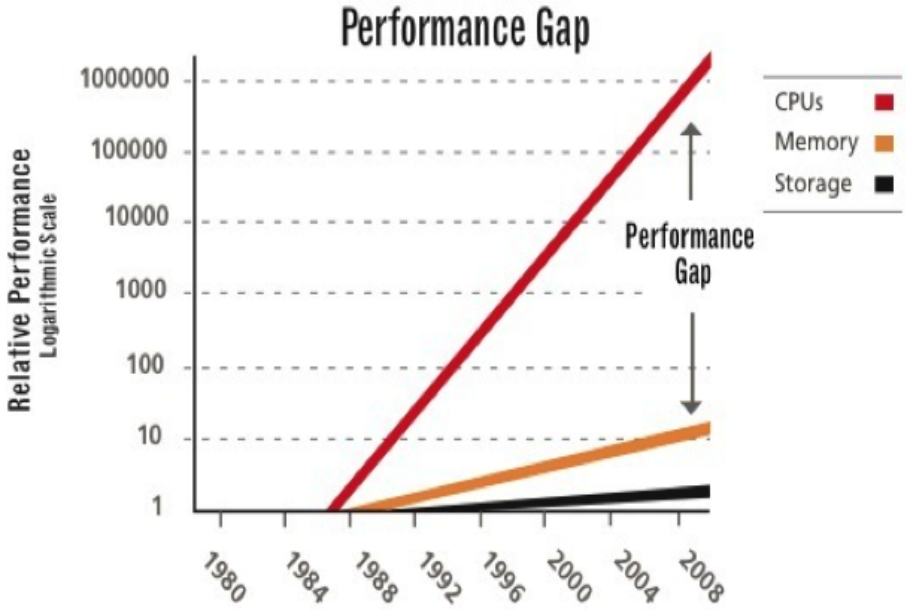
Memory trends



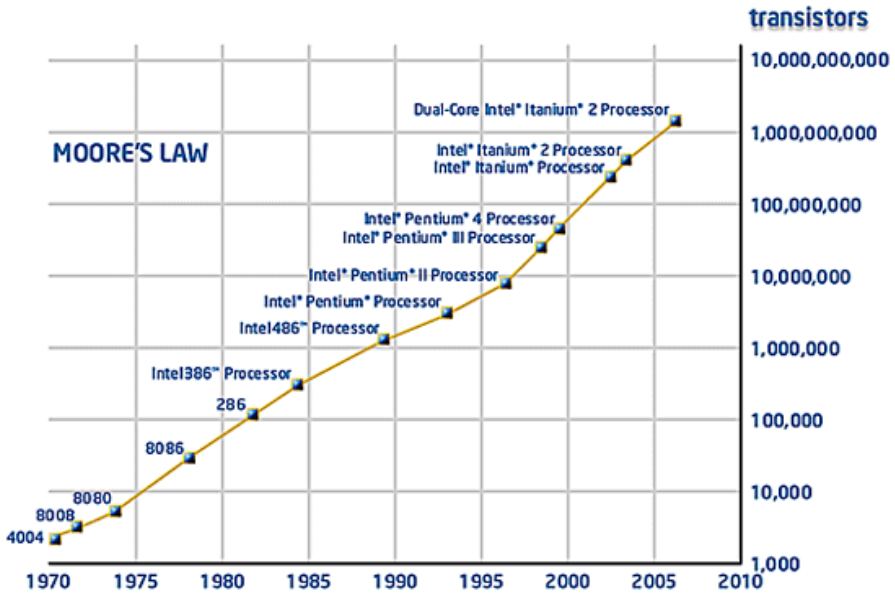
- Processing power doubles every 18 months
- Memory size doubles every 18 months
- Disk capacity doubles every 18 months
- ***Disk positioning rate (seek & rotate) doubles every ten years!***
- Speed of DRAM and disk improves a few % per year

does NOT match CPU speed improvements!

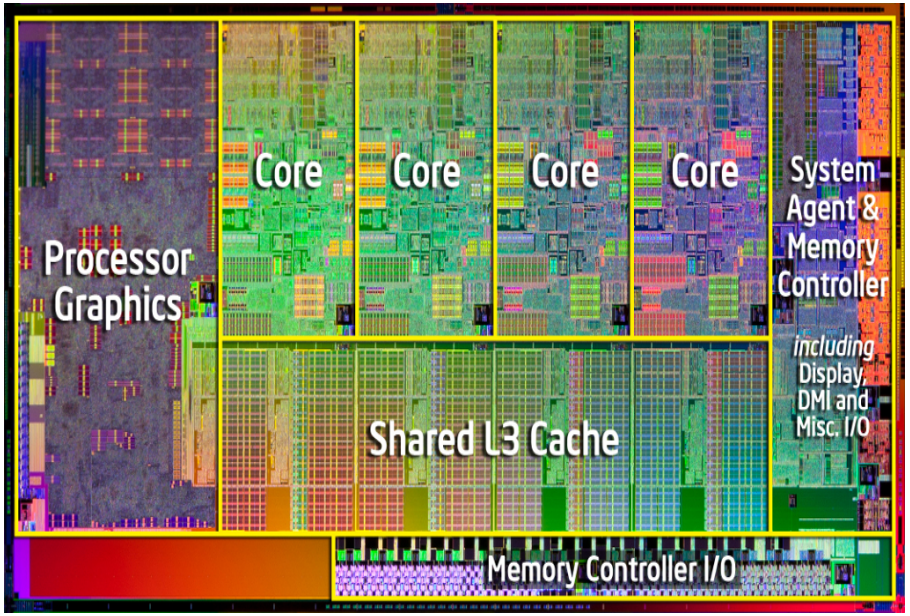
CPU/Memory performance gap



Transistors in a CPU



Transistors in a CPU



Power consumption of a CPU

Intel Core i5, Core i7
chip area $1.3 - 2.5 cm^2$
power consumption $77 - 130 W \Rightarrow \approx 50 W/cm^2$ to be removed

compare $\approx 10 W/cm^2$

Cooling!

Outline

What is performance?

Plane	DC to Paris	Speed	Persons	Throughput (pkm/h)
Boeing 747	6.5 h	980 km/h	470	460 000
Concorde	3 h	2160 km/h	132	285 120

- Time to complete a task (T_{exe})
  - Execution time, response time, latency
- Task per day, hour, second, ... (Performance)
  - Throughput, bandwidth

Performance

$$Performance(X) = \frac{1}{T_{exe}(X)}$$

"X is n times faster than Y" means:

$$\frac{T_{exe}(Y)}{T_{exe}(X)} = \frac{Performance(X)}{Performance(Y)} = n$$

- Speed of Concorde vs Boeing 747
- Throughput of Boeing 747 vs Concorde

Metrics of performance

Application	$\Longleftarrow$	Answers/month
Programming	$\Longleftarrow$	Response time (seconds)
language	$\Longleftarrow$	Operations/second
Compiler		
Instruction set	$\Longleftarrow$	MIPS/MFLOPS
Data-path control	$\Longleftarrow$	Megabytes/second
Functional units		
Transistors, wires, pins	$\Longleftarrow$	Cycles per second (clock rate)

MIPS = millions of instructions per second
MFLOPS = millions of FP operations per second

Programs to evaluate performance

- Real programs: e.g. TeX, spice, SPEC benchmarks, ...
- Kernels - Livermore loops
- Toy programs - sort, prime number generation
- Synthetic benchmarks - "The average program"
  - **Real programs are the only true measurement objects**
- SPEC benchmarks will be used here (plus some toy programs)

SPEC - Standard Performance Evaluation Corporation

http://www.spec.org

- First round 1989
  - 10 programs yielding a single number “SPECMarks”
- CPU92
  - SPECint92 - 6 integer programs
    SPECfp92 - 14 floating point programs
  - Compiler flags unlimited
- CPU95
  - New set of programs, SPECint95, SPECfp95
  - Single compiler flag setting for all programs: SPECint_base95, SPECfp_base95
- CPU2000
- CPU2006
- Lots of other performance tests

Which computer is the fastest?

Execution time			
Computer	**A**	**B**	**C**
Program P1	1	10	20
Program P2	1000	100	20
Total time	1001	110	40

- A is 10 times faster than B for P1
- B is 10 times faster than A for P2
- A and B are faster than C for P1
- C is faster than A and B if both P1 and P2 are run

How to summarize performance

- Arithmetic mean of execution time: $\frac{\sum T_i}{n}$
  or weighted execution time $\frac{\sum W_i * T_i}{n}$
- Normalized execution time R_i (SPECRatio) is handy for comparing performance $R_i = \frac{(T_{exe})_{RefComputer}}{(T_{exe})_i}$
  - Use geometric mean for normalized execution time: $\sqrt[n]{\prod R_i}$
    (independent of running times of the individual programs)

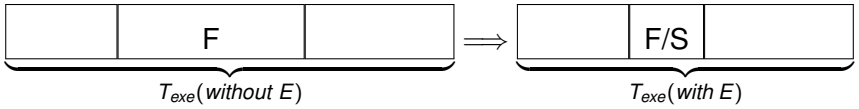
Outline

Quantitative principles

- Take advantage of parallelism
- Principle of locality
- Focus on the common case
- Amdahl's law
- Processor performance equation

Amdahl's law

Enhancement E accelerates a fraction F of a program by a factor S



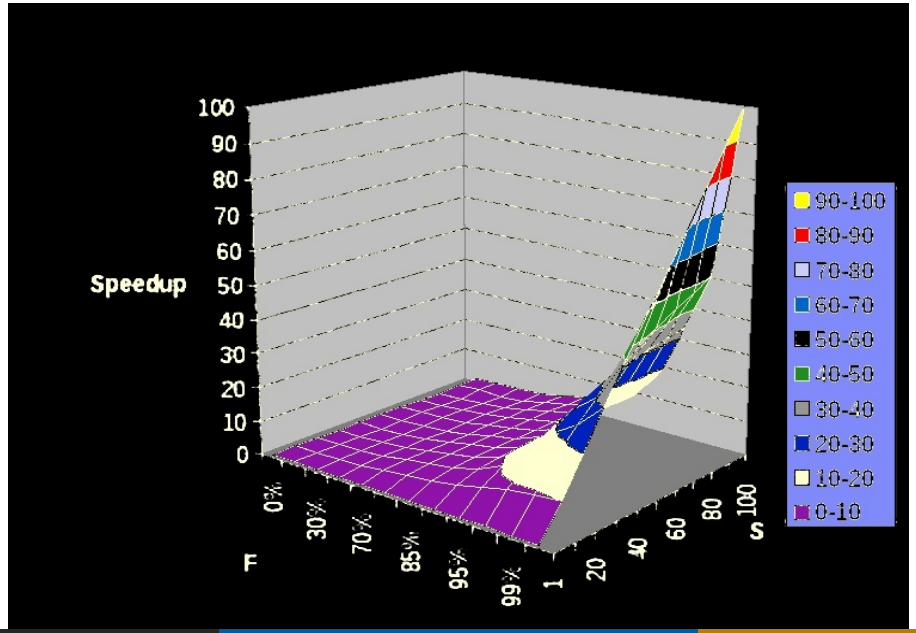
Speedup due to enhancement E:

$$Speedup(E) = \frac{T_{exe}(without\ E)}{T_{exe}(with\ E)} = \frac{Performance(with\ E)}{Performance(without\ E)}$$

$$T_{exe}(with\ E) = T_{exe}(without\ E) * [(1 - F) + F/S]$$

$$Speedup(E) = \frac{T_{exe}(without\ E)}{T_{exe}(with\ E)} = \frac{1}{(1-F)+F/S}$$

Amdahl's law in practice



Aspects of CPU performance

CPUtime = Execution time =
$seconds/program =$

$$\underbrace{(executed)instr./program}_{IC} * \underbrace{cycles/instr.}_{CPI} * \underbrace{seconds/cycle}_{T_c}$$

	IC	CPI	T_c
Program	X		
Compiler	X	(X)	
Instr. Set	X	X	
Organization		X	X
Technology			X

Instructions are not created equal

"Average Cycles per Instruction"

CPI_{op} = Cycles per Instruction of type *op*

IC_{op} = Number of executed instructions of type *op*

$$CPUtime = T_c * \sum (CPI_{op} * IC_{op})$$

"Instruction frequency"

$$\overline{CPI} = \sum (CPI_{op} * F_{op}) \text{ where } F_{op} = IC_{op}/IC$$

Invest resources where time is spent!

Example calculating CPI

Op	F_{op}	CPI_{op}	$F_{op} * CPI_{op}$	**% time**
ALU	50 %	1	0.5	(33 %)
Load	20 %	2	0.4	(27 %)
Store	10 %	2	0.2	(13 %)
Branch	20 %	2	0.4	(27 %)

$\overline{CPI}$ = 1.5

Principle of Locality

Program access a relatively small portion of the address space at any instant of time.
Two Different Types of Locality

- Temporal Locality (Locality in Time):
  If an item is referenced, it will tend to be referenced again soon (e.g., loops, reuse)
- Spatial Locality (Locality in Space):
  If an item is referenced, items whose addresses are close by tend to be referenced soon (e.g., straight-line code, array access)

Used to improve memory access!

Measuring components of CPU performance

- Choose a set of benchmark programs (e.g. SPEC)
- Determine what you want to measure (T_c, CPI, IC)
- Construct an analysis tool
  - Simulator
  - Profiling tool
- Run experiments
- Present data

Price-Performance

Computer	**Price** $	**Performance** TPM	**price-performance** TPM/1000 $
IBM p5 595	16300000	3200000	197
DELL PE 2900	62000	100000	1600
(32 * DELL PE 2900	1984000	3200000	1600)