EITF20: Computer Architecture

Part1.1.1: Introduction

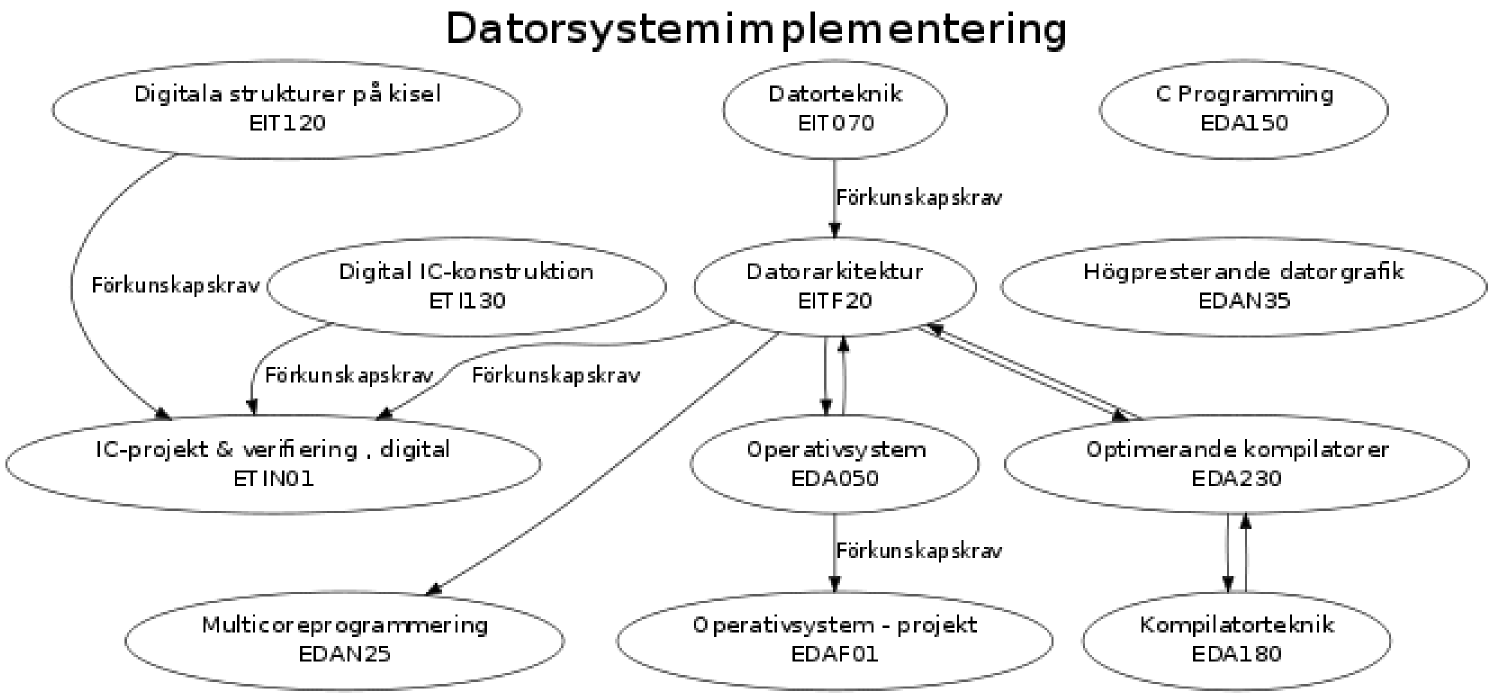
Liang Liu
liang.liu@eit.lth.se

Course Factor

- Computer Architecture (7.5HP)

http://www.eit.lth.se/kurs/eitf20

- **EIT's Course Service Desk (studerandeexpedition)**
    - Course secretary: Doris Glöck, Room 3152B
    - e-mail: doris.glock@eit.lth.se



Outline

- Computers
- Computer Architecture
- This Course
- Trends
- Performance
- Quantitative Principles

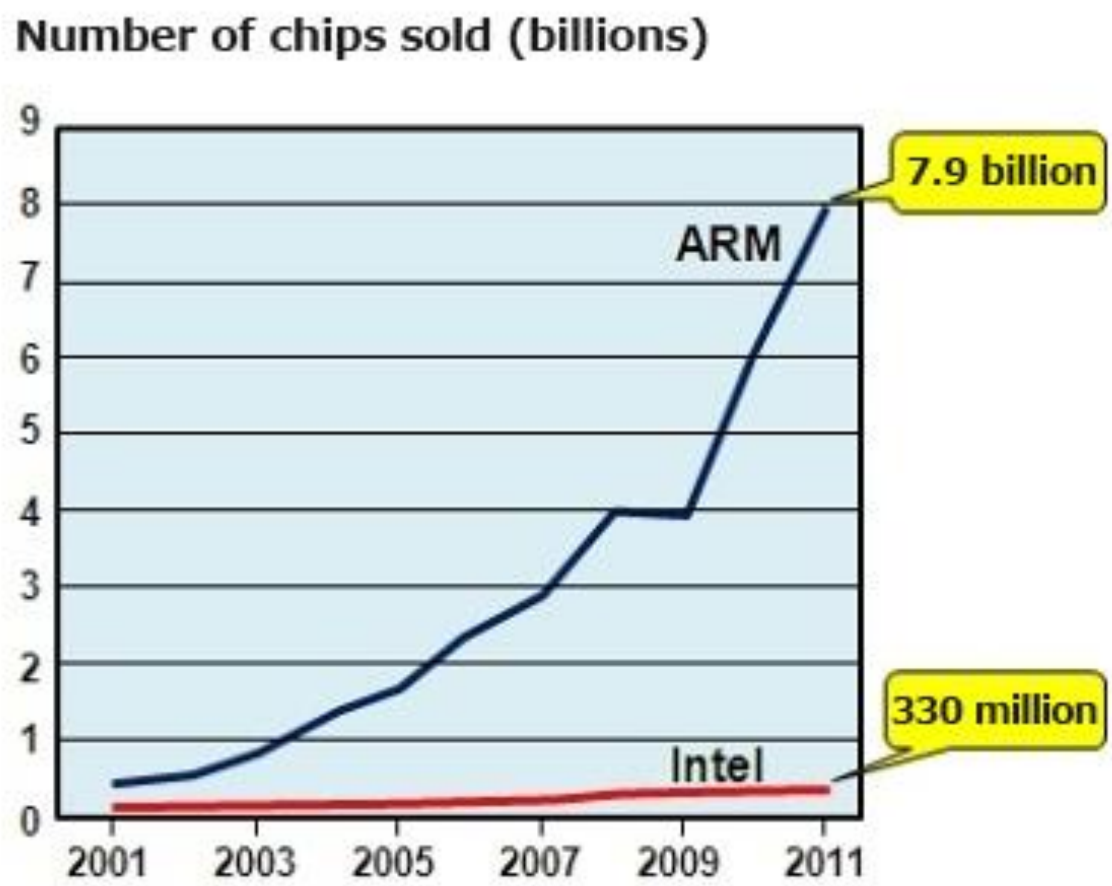
Computer is everywhere

Class of Computers

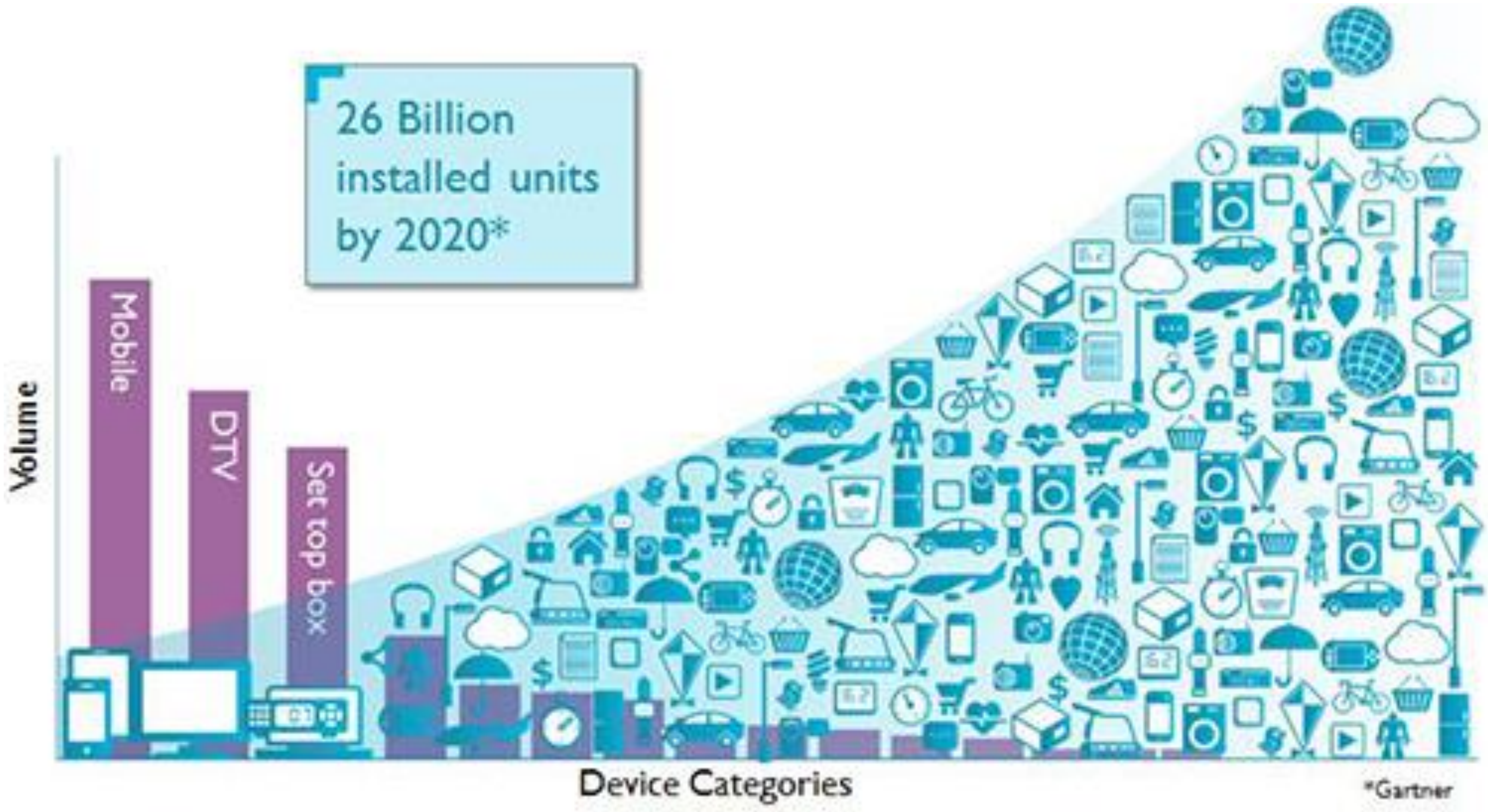
Feature	Personal mobile device (PMD)	Desktop	Server	Clusters/warehouse-scale computer	Embedded
Price of system	$100–$1000	$300–$2500	$5000–$10,000,000	$100,000–$200,000,000	$10–$100,000
Price of micro-processor	$10–$100	$50–$500	$200–$2000	$50–$250	$0.01–$100
Critical system design issues	Cost, energy, 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.

Intel v.s. ARM

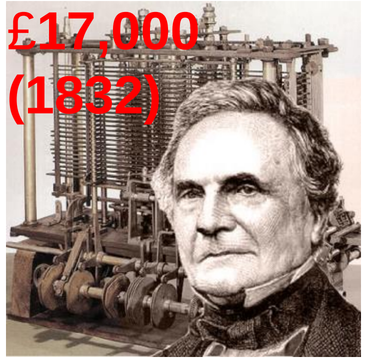

Number of chips sold (billions)
9
8
7
6
5
4
3
2
1
0
2001
2003
2005
2007
2009
2011
ARM
Intel
7.9 billion
330 million

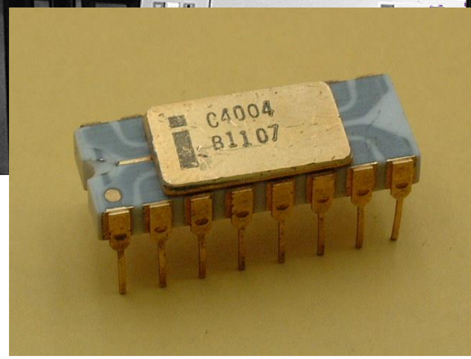
IoT - ARM


26 Billion
installed units
by 2020*
Volume
Mobile
DTV
Set top box
Device Categories
*Gartner

Time-line

- **Mid-1800 Programmable computer**
  - Charles Babbage (analytical engine)
  - Ada Lovelace (programmer)
- **1940s First modern computers**
  - Zuse, MARK, ENIAC, ...
- **1960s Mainframe**
  - 1964 IBM System/360
- **1970s Minicomputer**
  - 1971 First microprocessor
  - Graphics Xerox Alto

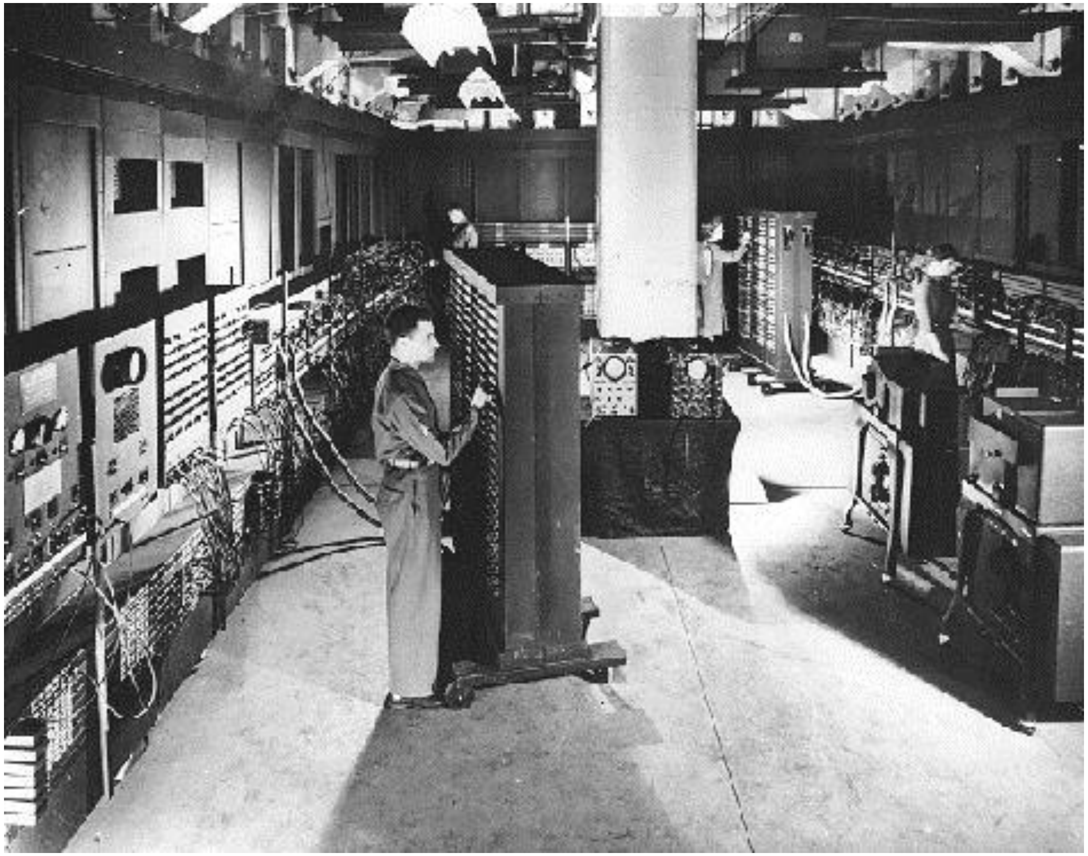




Time-line

- **1980s Desktop**
  - 1977 Apple II
  - 1981 IBM PC
- **1990s PDA**
- **2000s Embeded computers**
- **2010s Cloud computing**
- **2020s Boundless computing**
- http://anddum.com/timeline/history_short.htm

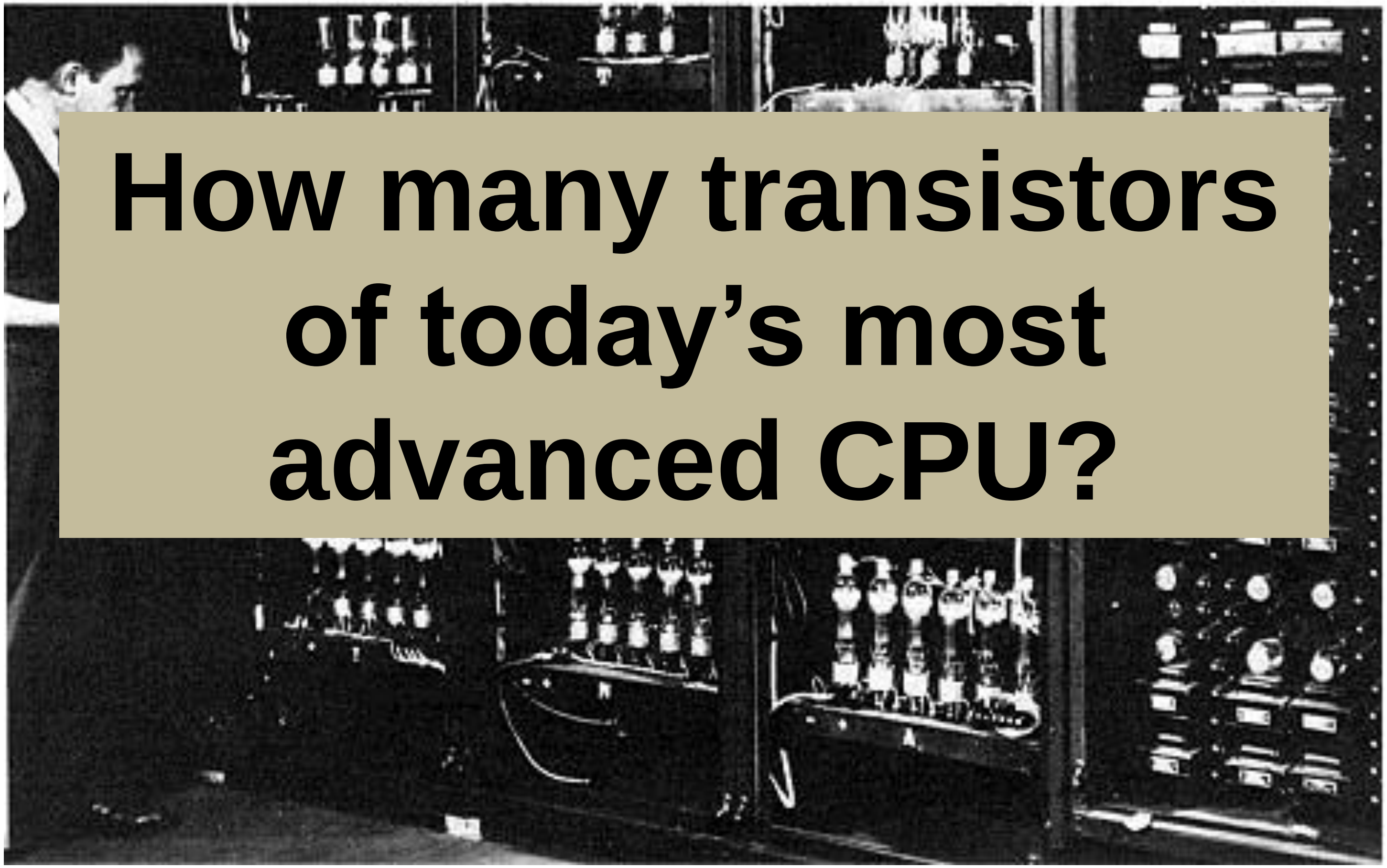
The first electronic computer



ENIAC-1946

18 000 vacuum tubes, 30 ton, 150m^2 ,140kW

The first electronic computer

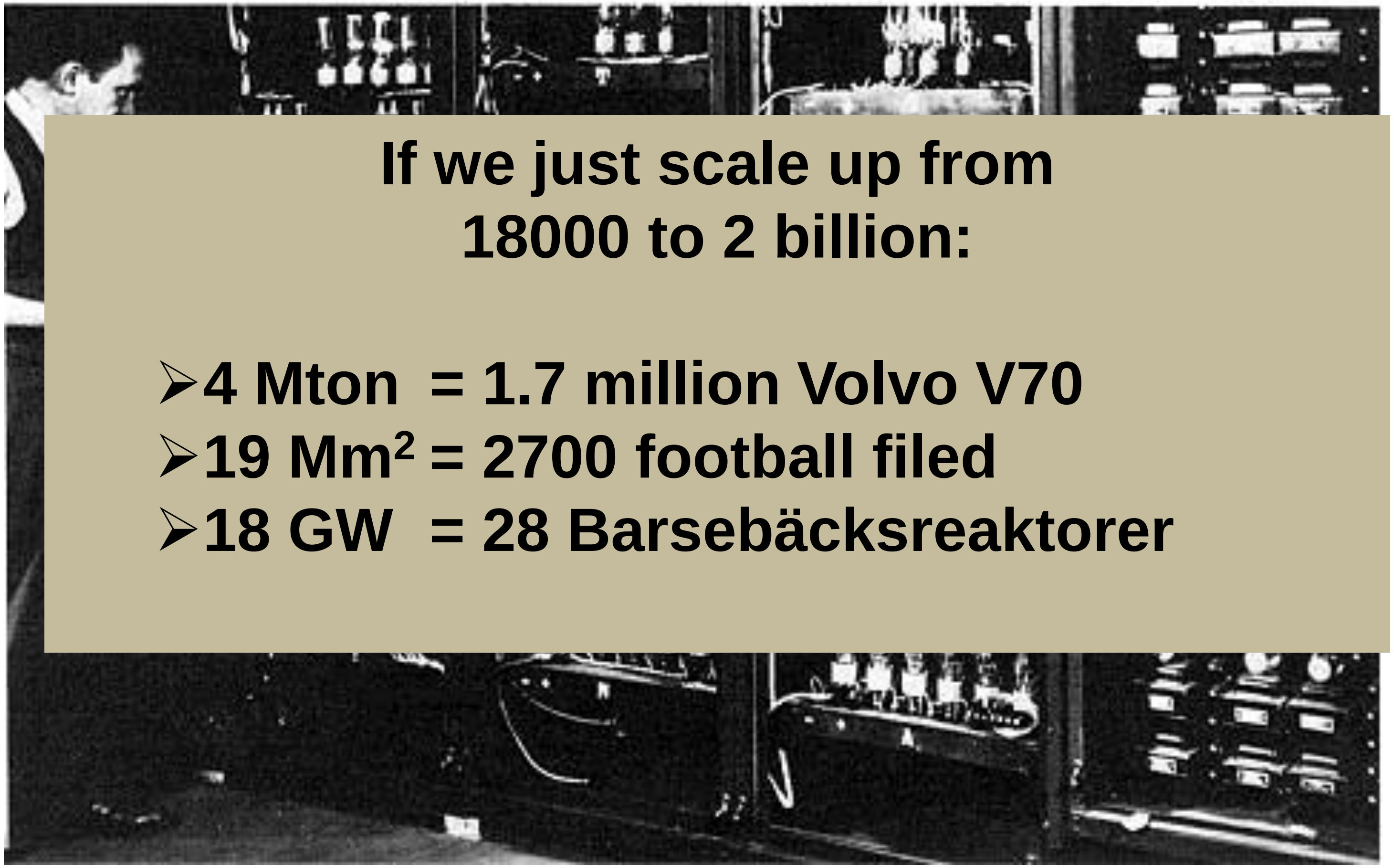


How many transistors of today's most advanced CPU?

ENIAC-1946

18 000 tubes, 30 ton, 150m^2 ,140kW

The first electronic computer



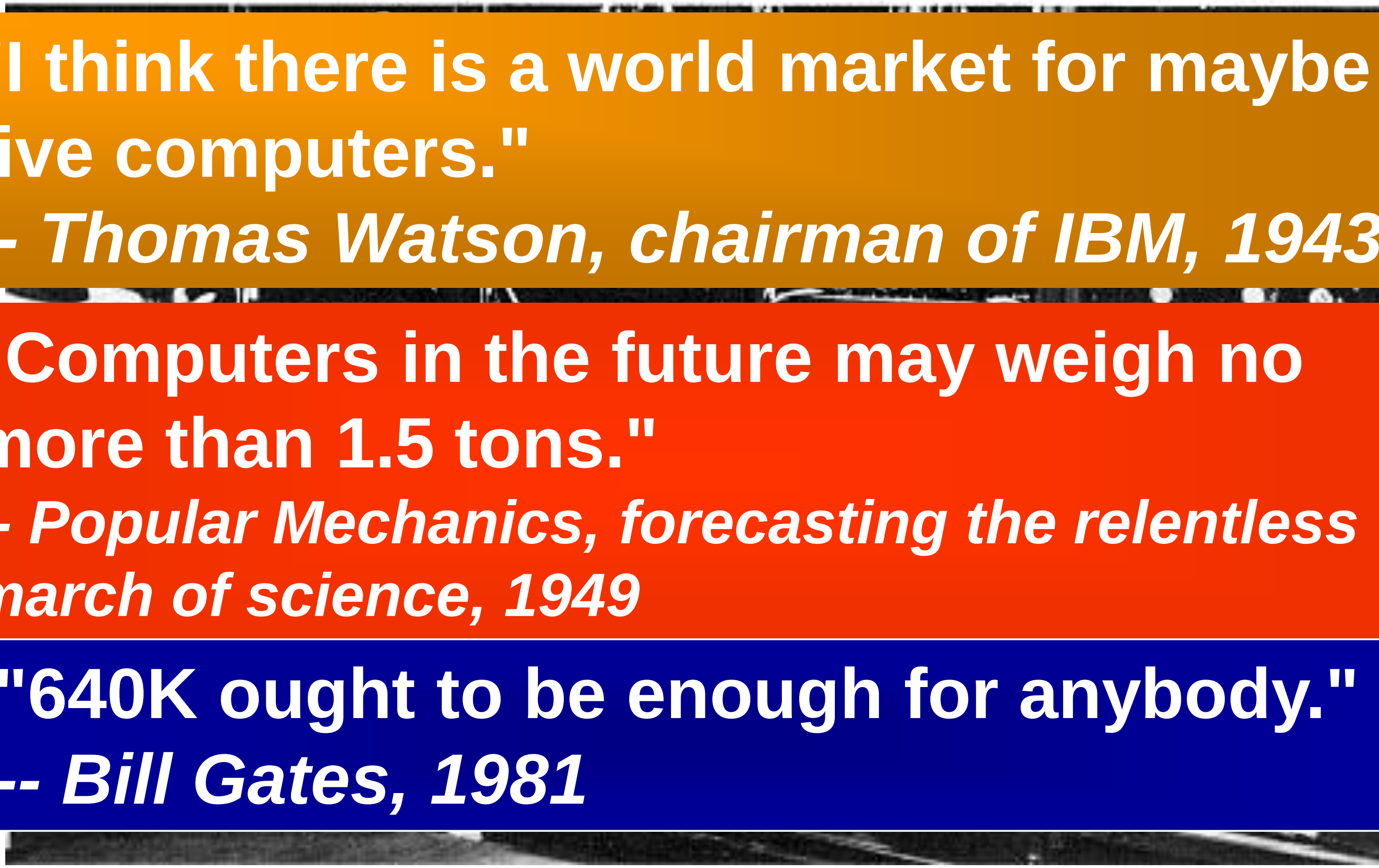
If we just scale up from 18000 to 2 billion:

- **4 Mton = 1.7 million Volvo V70**
- **19 Mm^2 = 2700 football filed**
- **18 GW = 28 Barsebäcksreaktorer**

ENIAC-1946

18 000 tubes, 30 ton, 150m^2 ,140kW

The first electronic computer



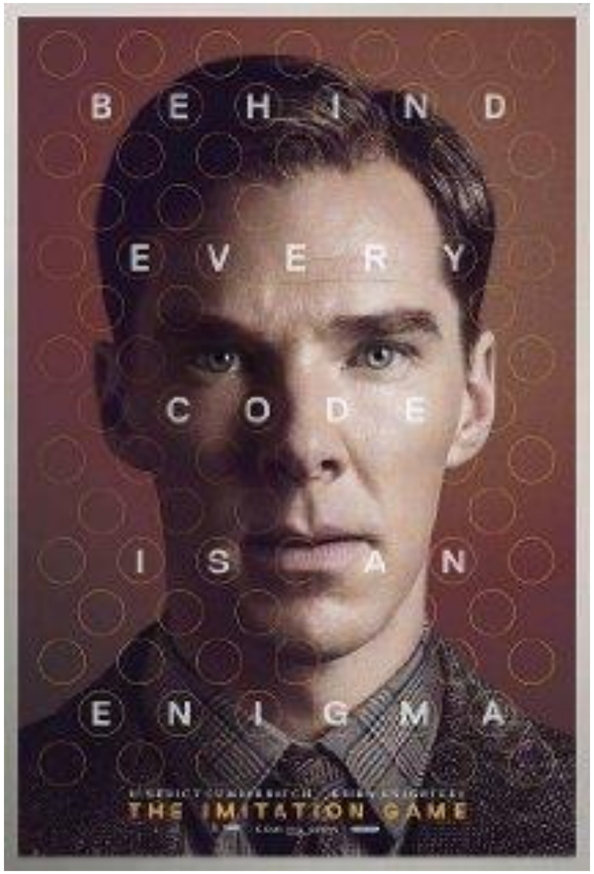
"I think there is a world market for maybe five computers."
-- Thomas Watson, chairman of IBM, 1943

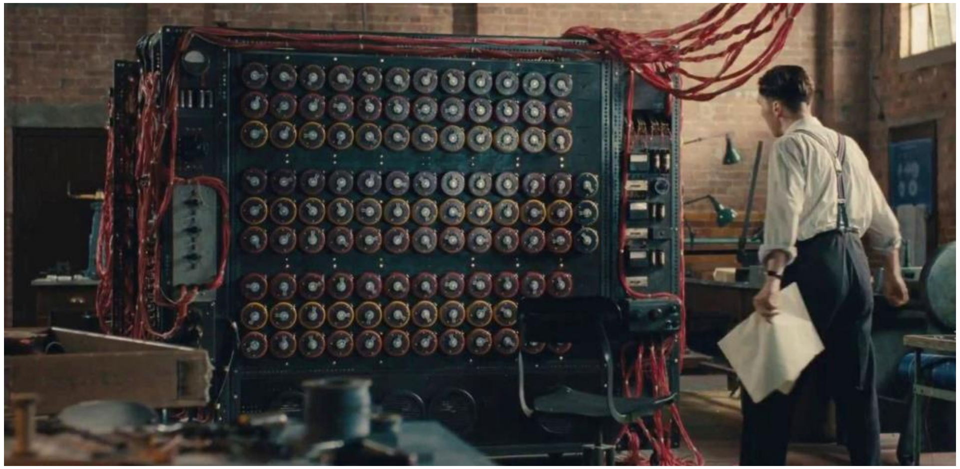
"Computers in the future may weigh no more than 1.5 tons."
-- Popular Mechanics, forecasting the relentless march of science, 1949

"640K ought to be enough for anybody."
-- Bill Gates, 1981

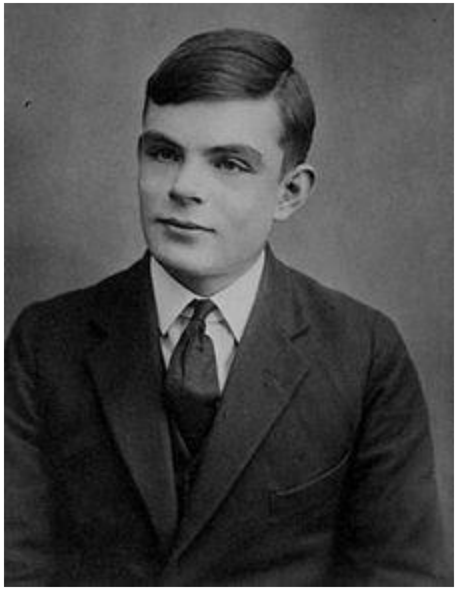
ENIAC-1946
18 000 tubes, 30 ton, 150m^2 ,140kW

The imitation game



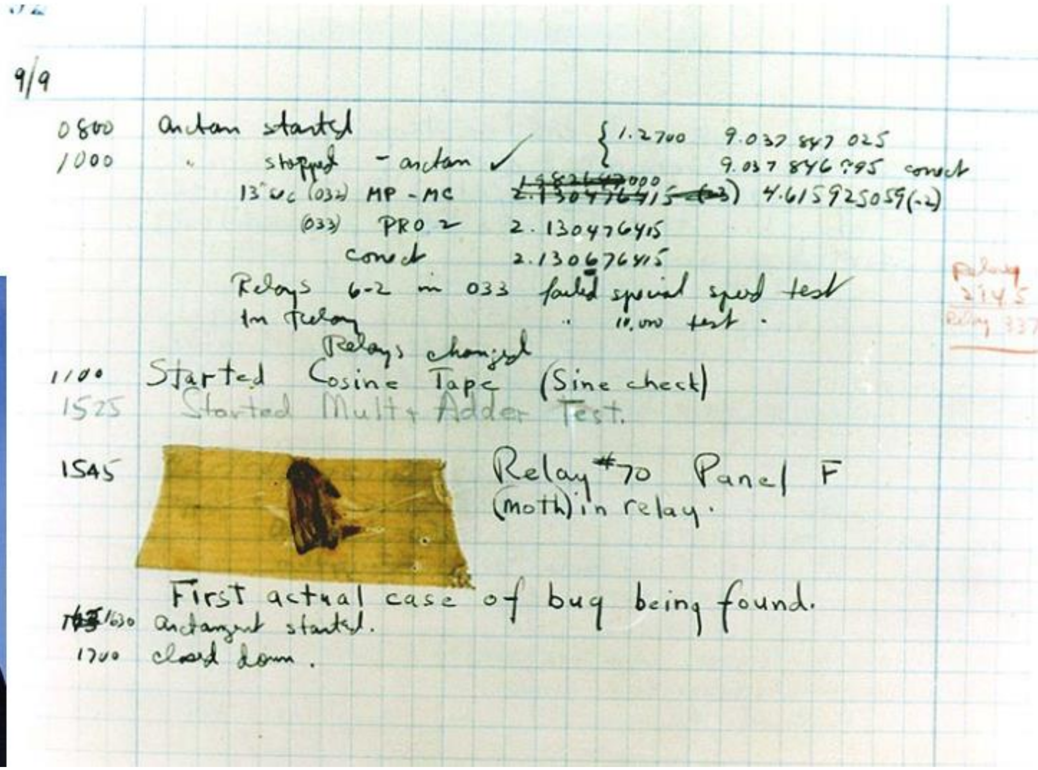


Alan Turing



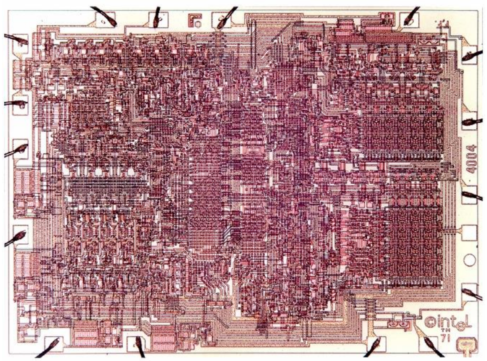
Interlude: Debug

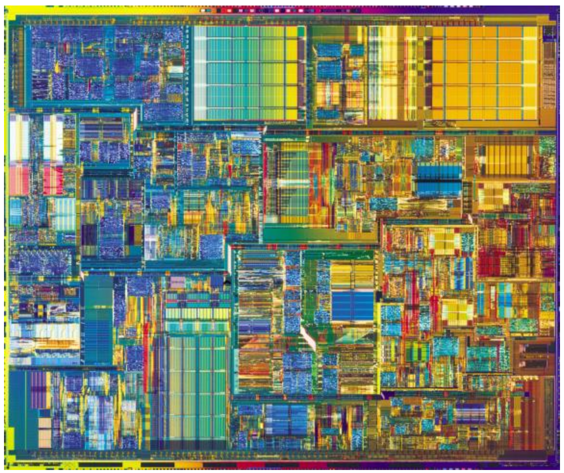
In 1947, Rear Admiral Grace Murray Hopper and associates was working on Mark II, the machine was experiencing problems. An investigation showed that there was a moth trapped in a relay. The operators removed the moth and affixed it to the log. The computer had been “debugged”.

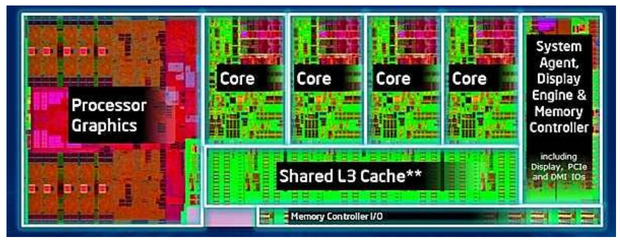


Development of Microprocessor

	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 M
Itanium 9560	2012	3 100 000 000	2.5 GHz	8	32 + 6 MB





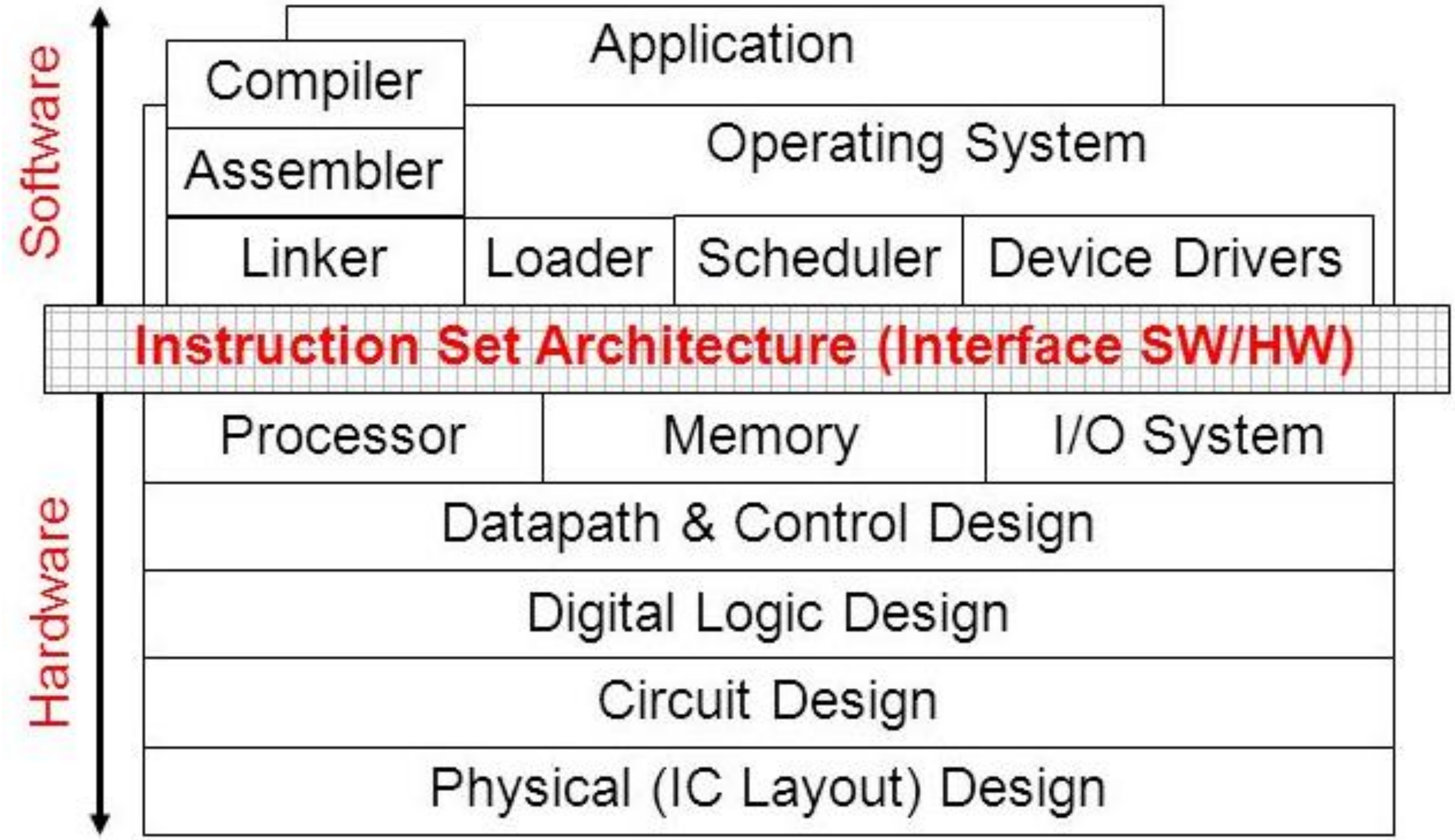


Outline

- Computers
- **Computer Architecture**
- This Course
- Trends
- Performance
- Quantitative Principles

The art of designing computers is based on engineering principles and quantitative performance evaluation

Computer abstraction levels



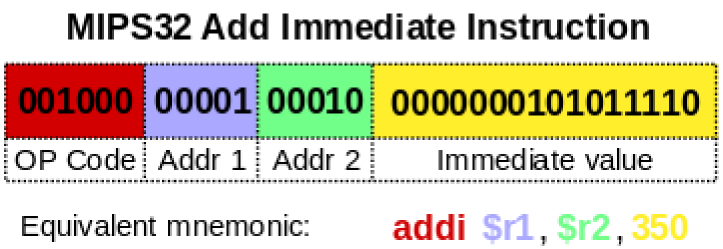
Computer Architecture

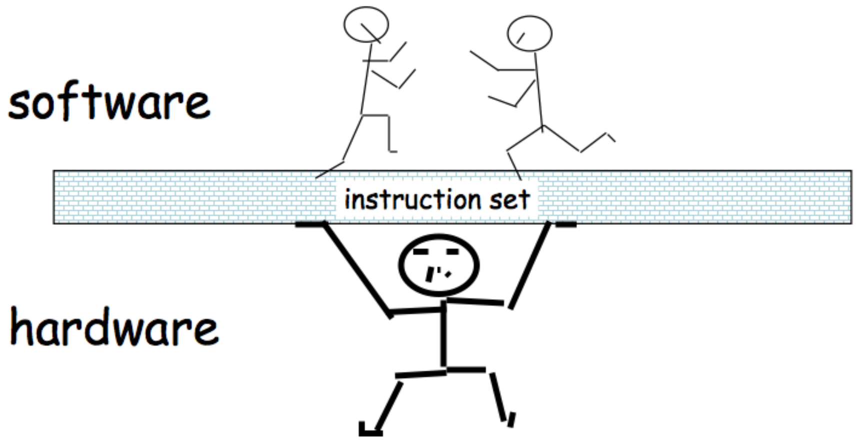
Computer architecture is a set of disciplines that describe the functionality, organization and implementation of computer systems.

- **ISA: Instruction-set architecture**
- **Computer orginization: micro architecture**
- **Specific implementation**

ISA

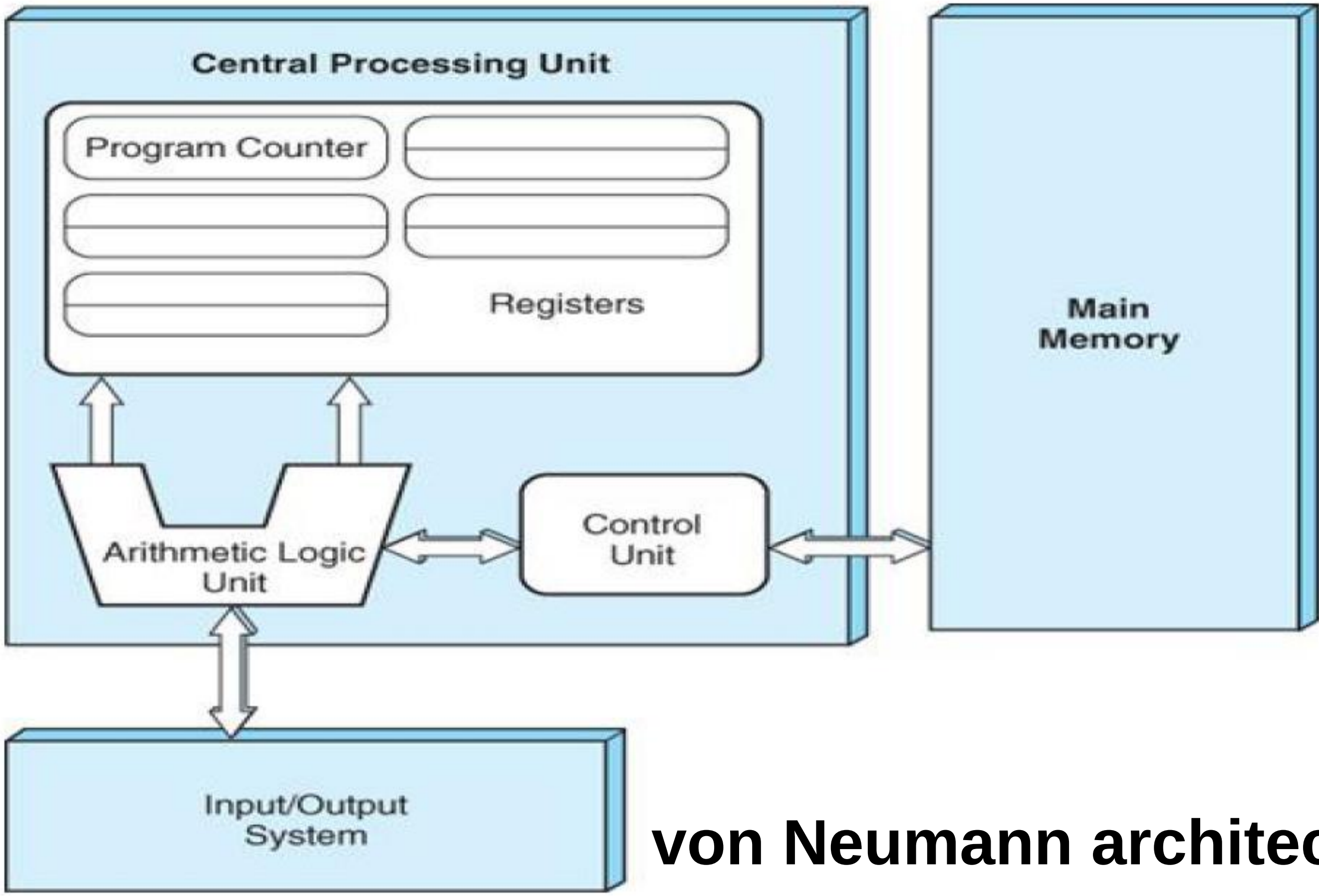
An instruction set architecture (ISA) is the interface between the computer's software and hardware and also can be viewed as the programmer's view of the machine.





Microarchitecture

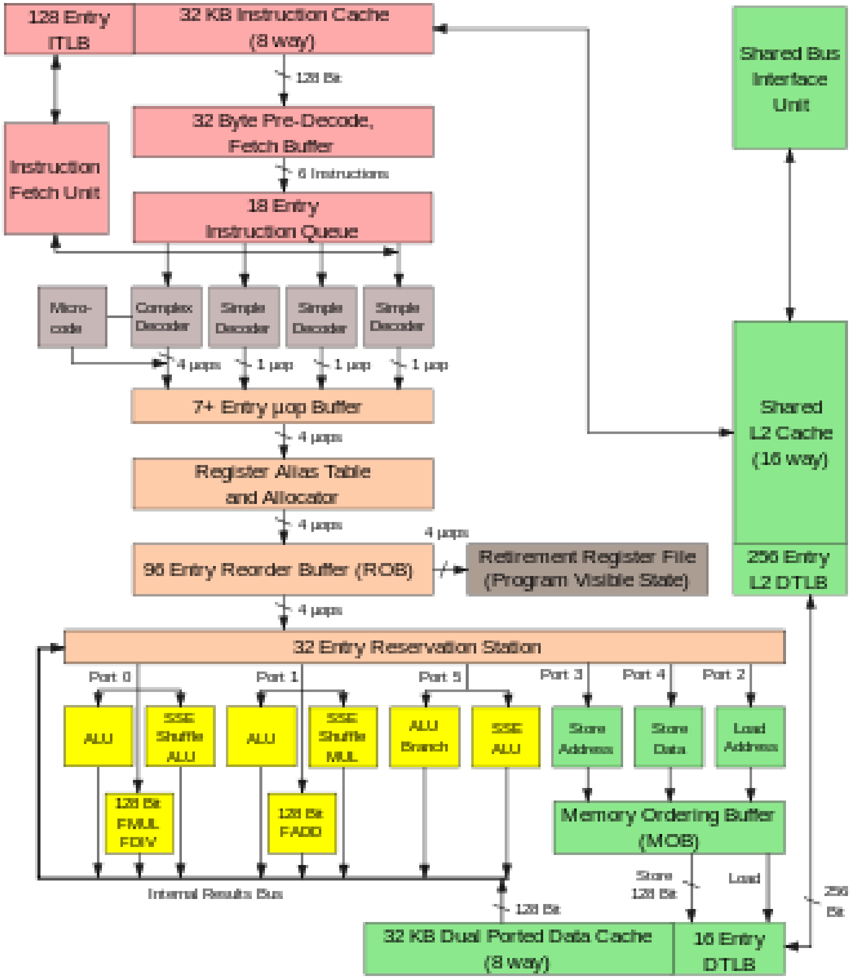
Microarchitecture is the way a given instruction set architecture (ISA) is implemented on a processor.



von Neumann architecture

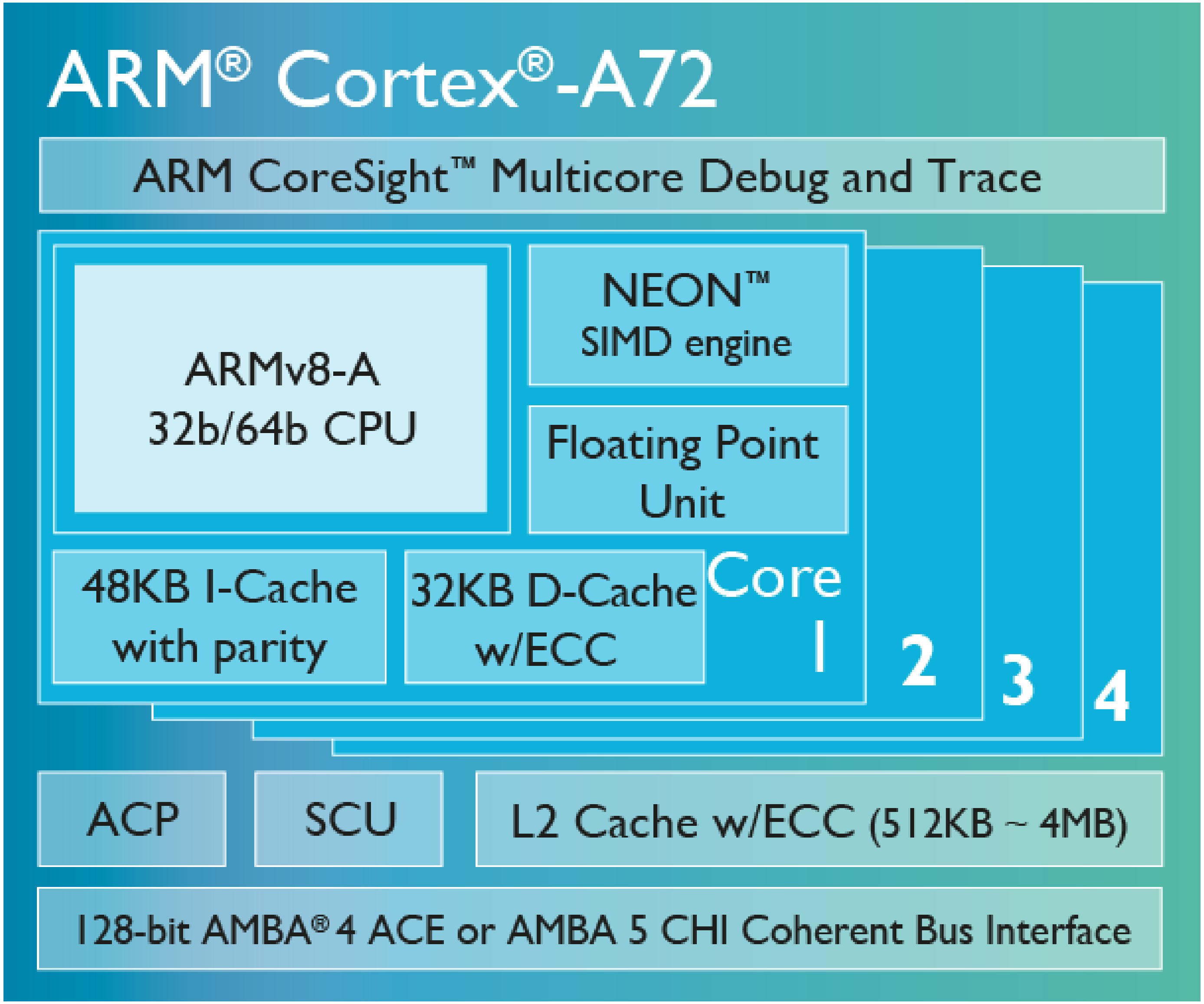
Microarchitecture

Microarchitecture is the way a given instruction set architecture (ISA) is implemented on a processor.

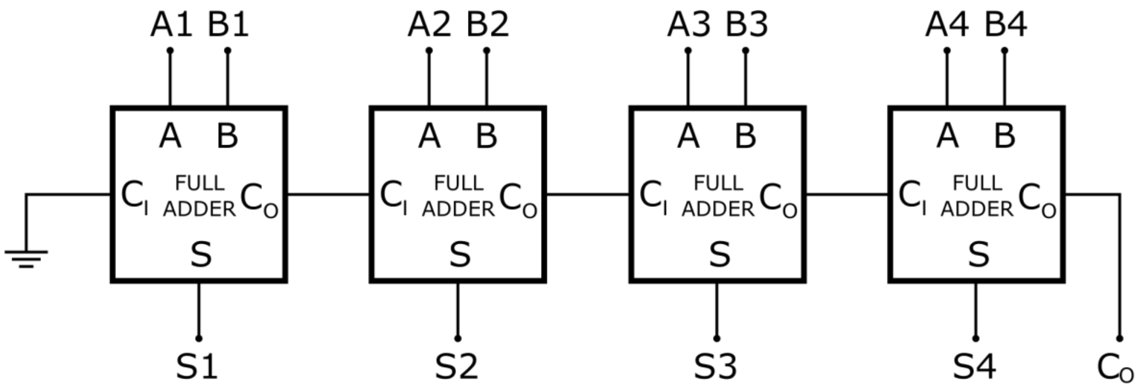


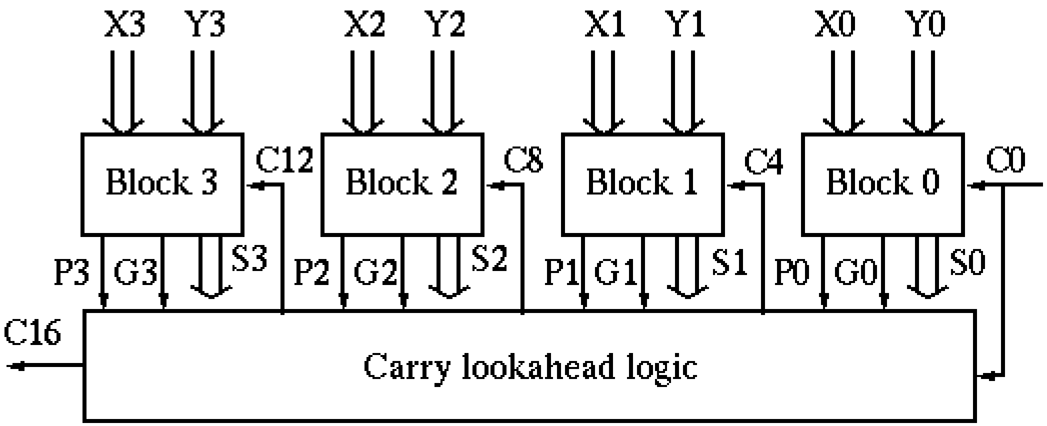
Intel Core 2 Architecture

Microarchitecture



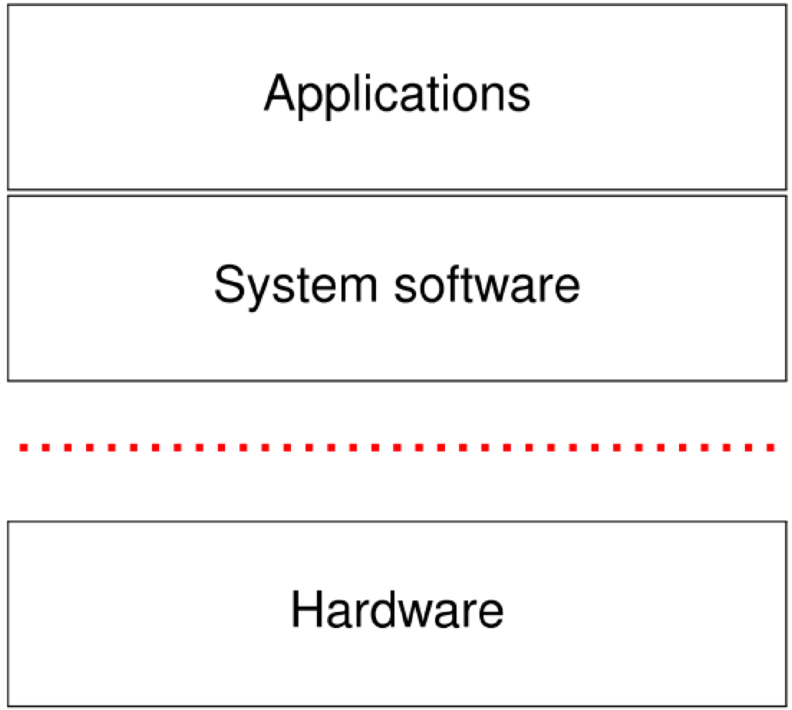
Implementation





The role of computer architecture?

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



Why computer architecture?

Understand how to evaluate and choose

- What do we mean ”one computer is faster than another”?
- How can Gene Amdahl help you decide which enhancement is the best?
- Is a larger cache better than higher clock frequency?
- Why is pipelining faster than combinational circuits?
- Different levels of caches - why?
- ...

Design your own specialized architecture

- Embedded special purpose processors
  - Axis Communications/Ericsson/Nokia/ARM/SAAB
  - ...

Write better program

What computer architecture?

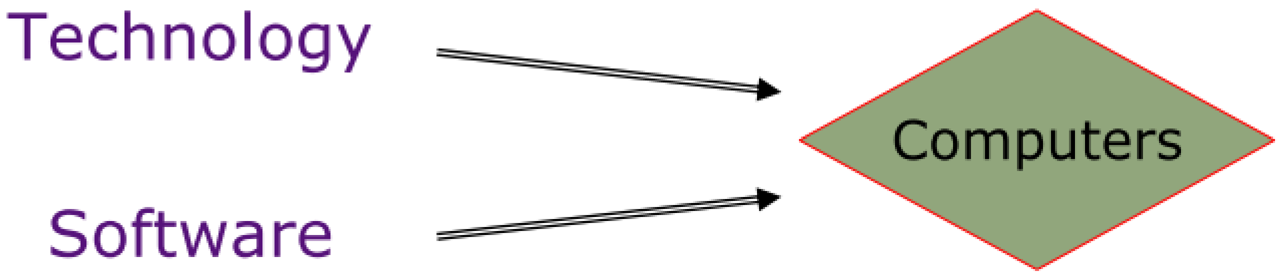
Design and analysis

- ISA
- Orgnization (microarchitecture)
- Implementation

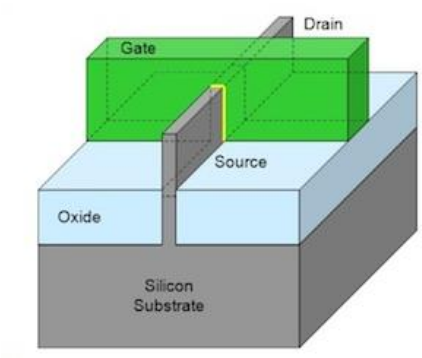
To meet requirements of

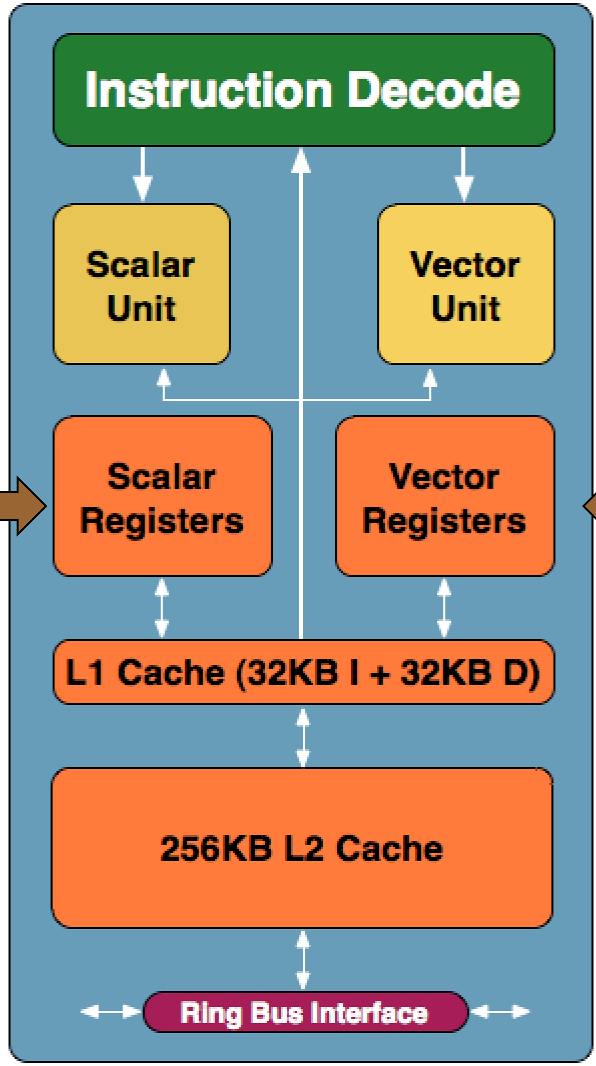
- Functionality (application, standards…)
- Price
- Performance
- Power
- Reliability
- Dependability
- Compatability
- ..

What affect computer architecture?



X86 Architecture



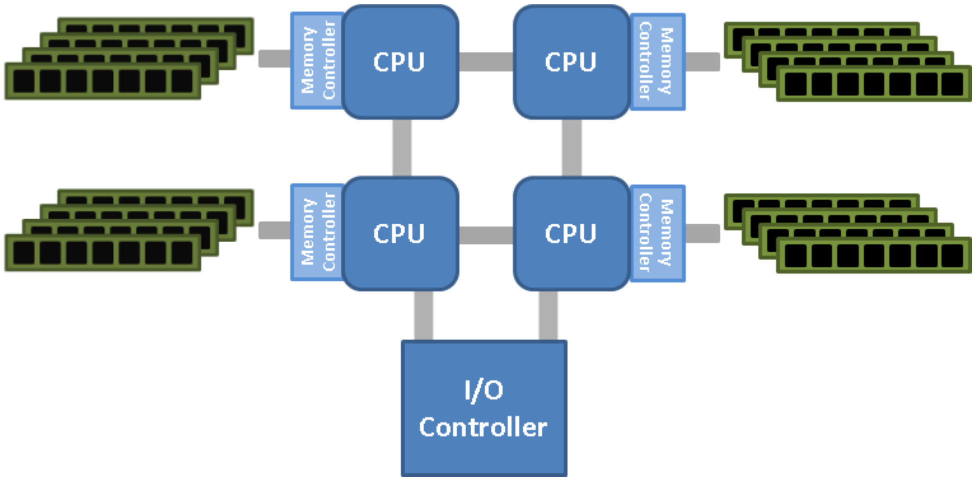


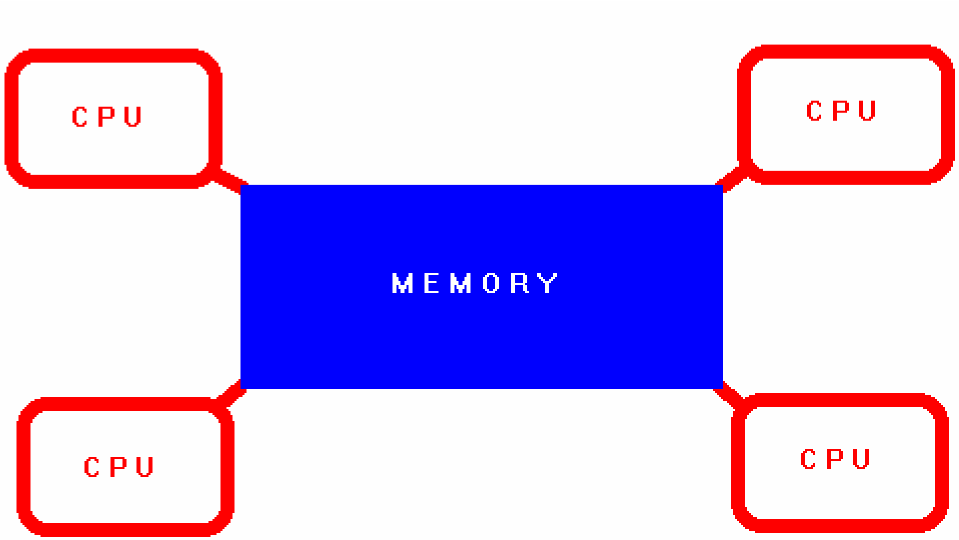
Architecture change due to new applications

ARM

?

Architecture change due to new applications





Outline

- Computers
- Computer Architecture
- **This Course**
- Trends
- Performance
- Quantitative Principles

Course Objectives

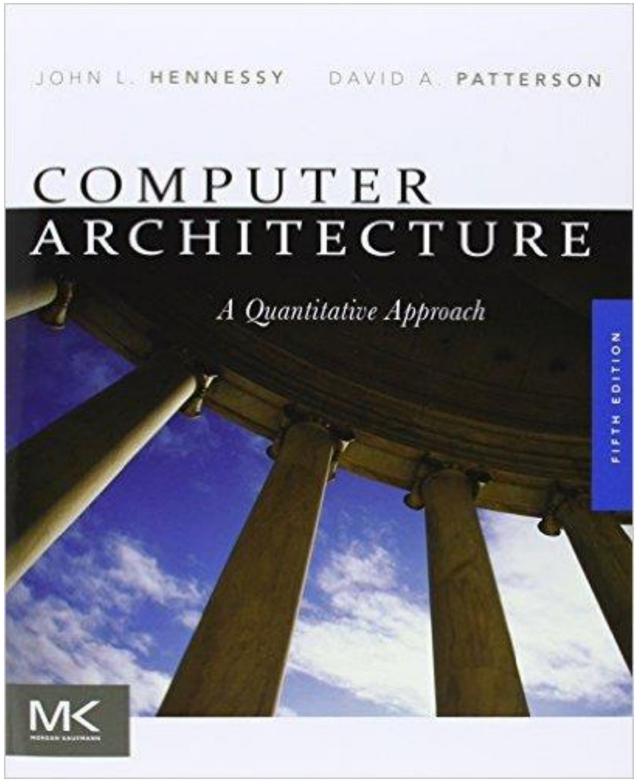
After this course, you will...

- **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 microarchitecture of a processor
- **Be able to evaluate design alternatives towards design goals using quantitative evaluation methods**
- **Side effects...**
  - Better digital IC designer
  - Better understanding of compiler, operating system, high-performance programming

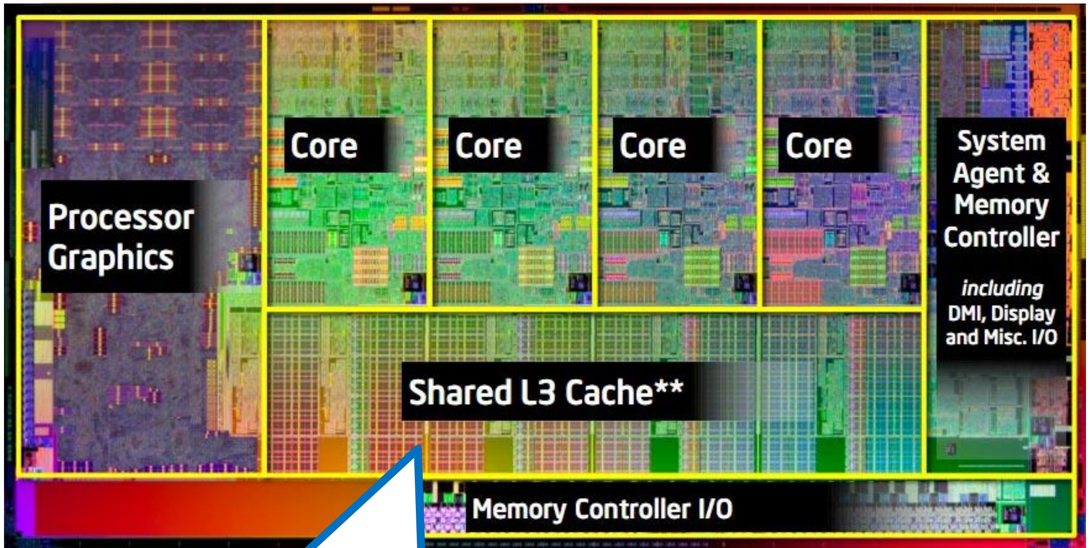
Book Recommendation

Computer Architecture – A Quantitative Approach

- Hennessy, Patterson
- 5th Edition



Course Content & Schedule



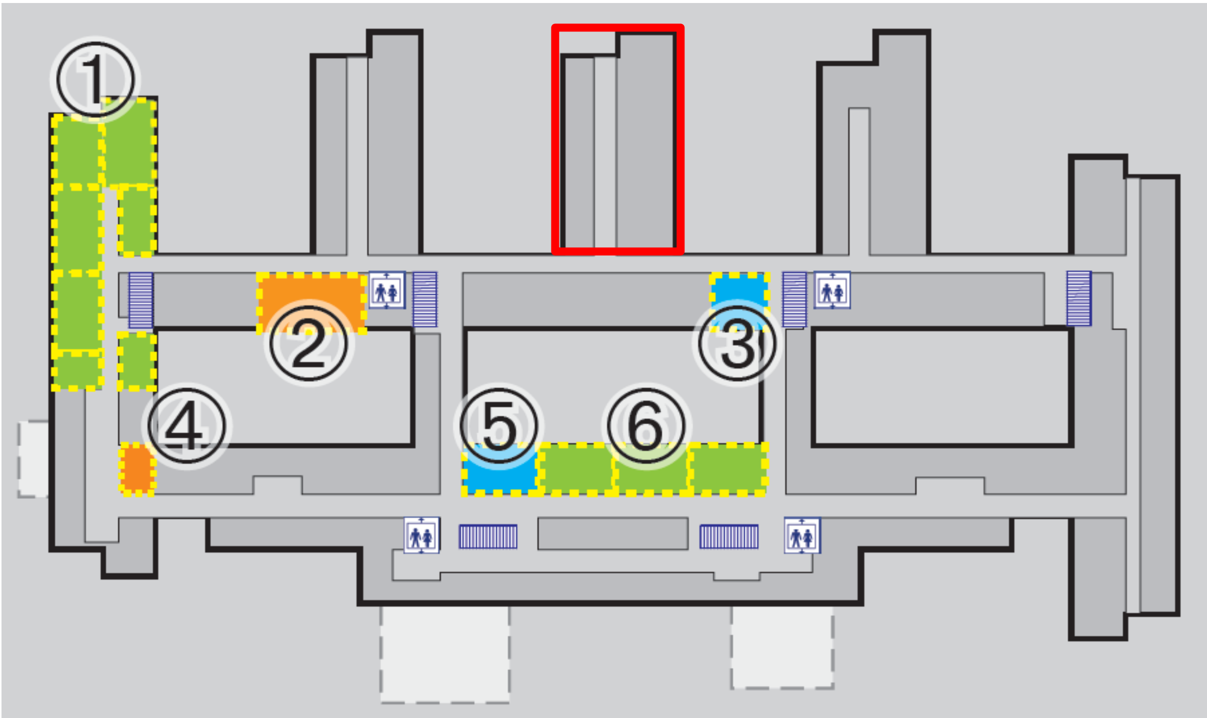
- Concept &Theory
- Assignment & Project
- Exams

- Overview
- Instruction set architecture
- Pipeline
- Memory System
- Storage System
- I/Os
- Embedded & application specific processing

Teachers

Lecture

- Liang Liu, Assistant Professor
- Email: liang.liu@eit.lth.se
- Room: E2342
- Homepage: http://www.eit.lth.se/staff/Liang.Liu



Teachers

Lecture

- Liang Liu, Assistant Professor
- Email: liang.liu@eit.lth.se
- Room: E2342
- Homepage: http://www.eit.lth.se/staff/Liang.Liu

Teaching Assistants

- Michal Stala

Michal Stala

Teachers

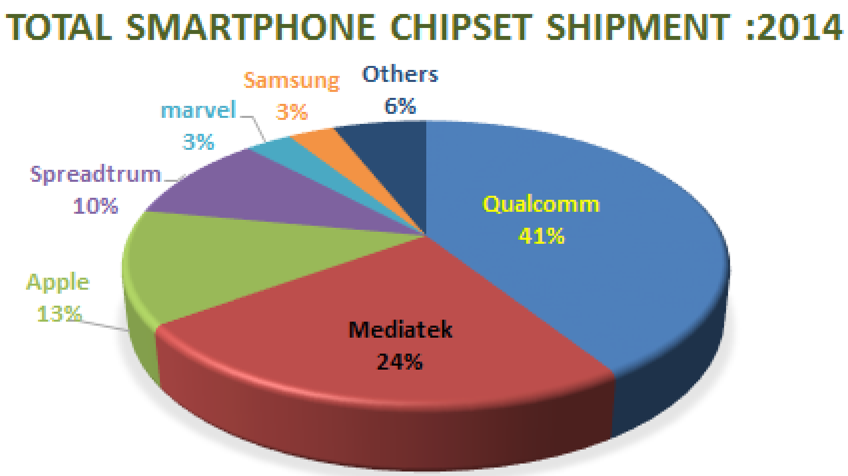
Lecture

- Liang Liu, Assistant Professor
- Email: liang.liu@eit.lth.se
- Room: E2342
- Homepage: http://www.eit.lth.se/staff/Liang.Liu

Teaching Assistants

- Michal Stala

Invited Lectures



Lectures and Labs

Lectures (10)

- Tuesday : 13:15-15:00 **E:1406**
- Thursday: 08:15-10:00 **E:1406**
- Covers design principles and analysis methodology
- Read the literature before each lecture
- Does not cover all of the literature
- Ask many questions!

Labs (4)

- Tuesday: 08:15-12:00 **E:4118-E:4119**
- Friday: 08:15-12:00 **E:4118-E:4119**
- 2 students/group
- 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**)
- Finish Lab before **DEADLINE**

Examination (Written)

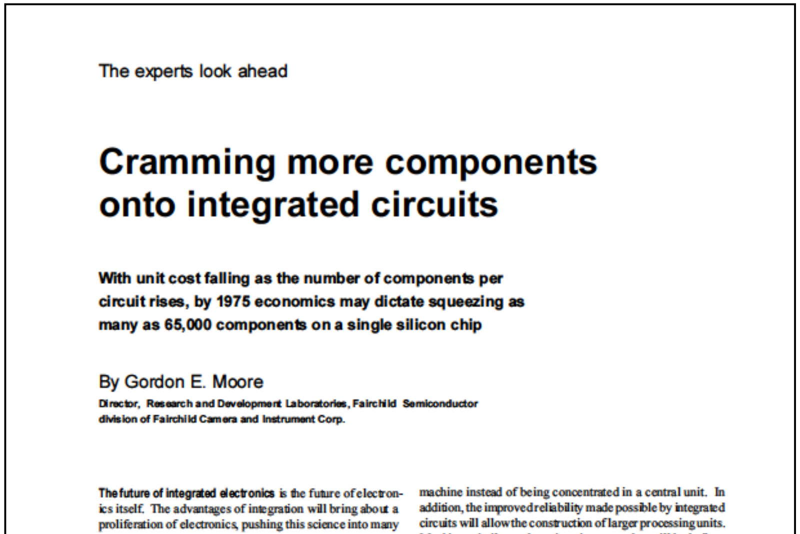
- **Anonymous exam**
- **Pass all labs to be able to attend written exam**
- **Five problems**
  - Highly lab related
  - Problem solving
  - Descriptive nature

Questions?

Outline

- Computers
- Computer Architecture
- This Course
- **Trends**
- Performance
- Quantitative Principles

Moore’s Law



The experts look ahead

Cramming more components onto integrated circuits

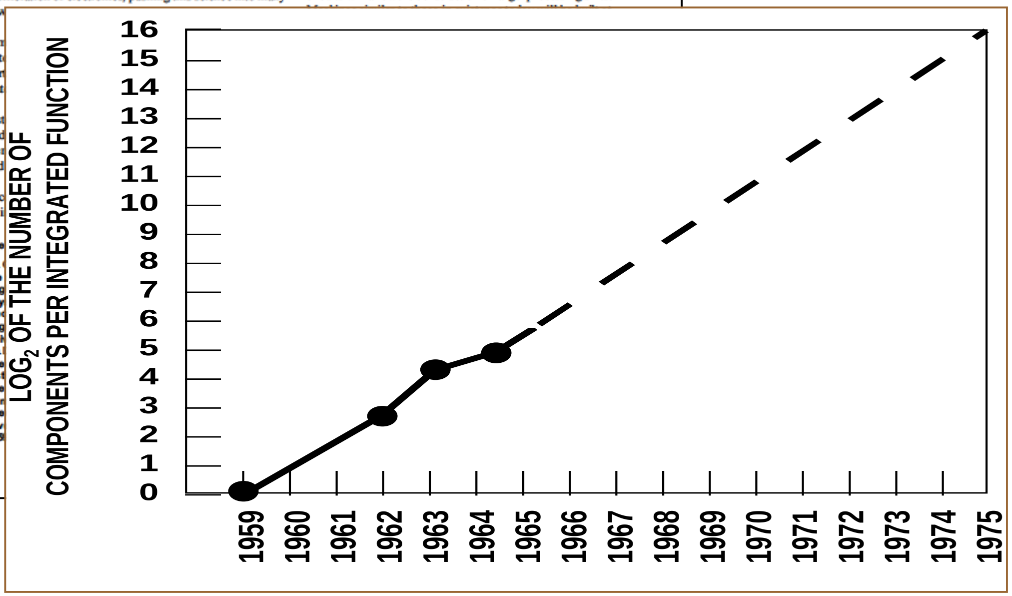
With unit cost falling as the number of components per circuit rises, by 1975 economics may dictate squeezing as many as 65,000 components on a single silicon chip

By Gordon E. Moore

Director, Research and Development Laboratories, Fairchild Semiconductor division of Fairchild Camera and Instrument Corp.

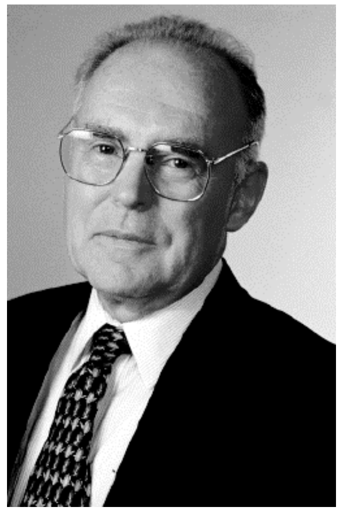
The future of integrated electronics is the future of electronics itself. The advantages of integration will bring about a proliferation of electronics, pushing this science into many

machine instead of being concentrated in a central unit. In addition, the improved reliability made possible by integrated circuits will allow the construction of larger processing units.



- **Electronics, Apr. 19, 1965**

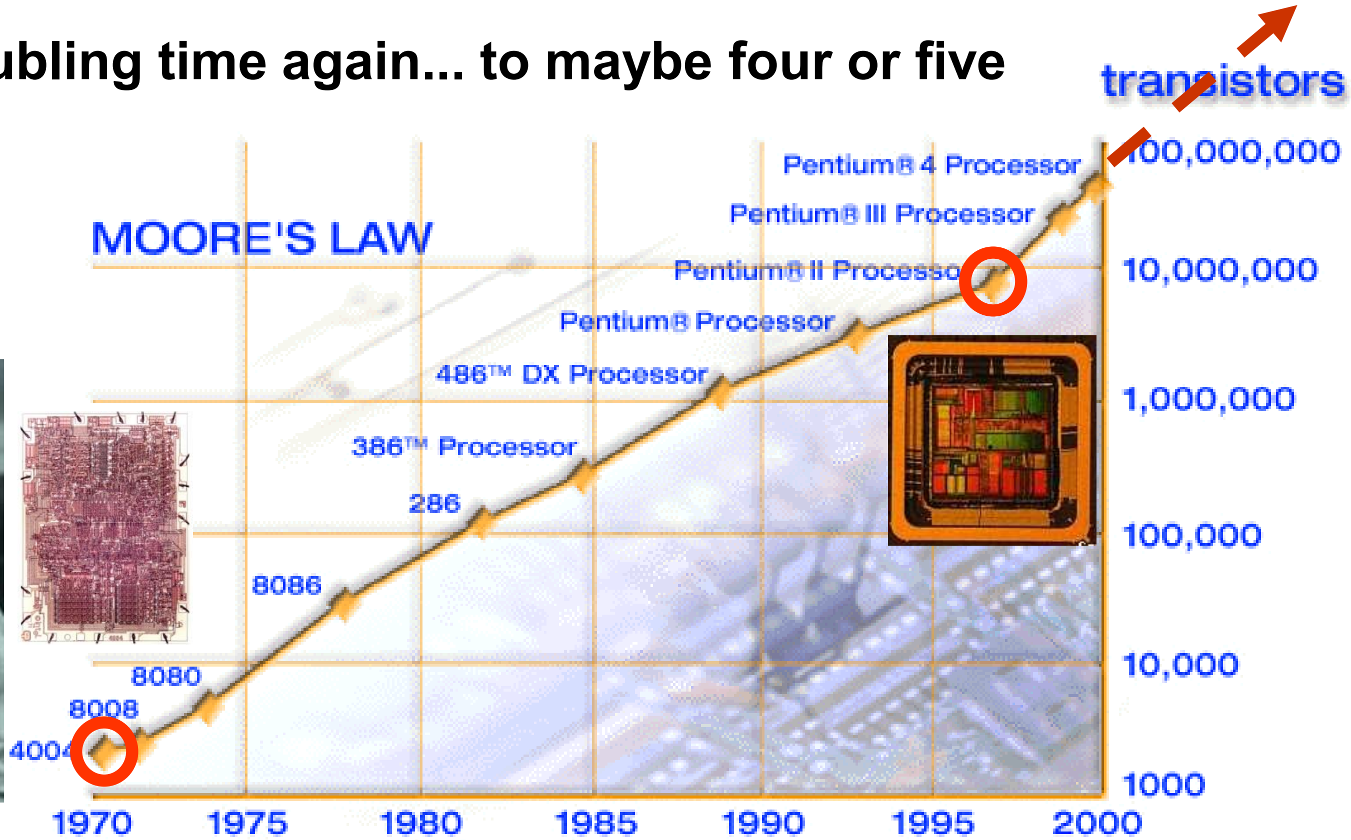
Gordon Moore (co-founder of Intel) described a doubling every year in the number of components per integrated circuit



Moore’s Law

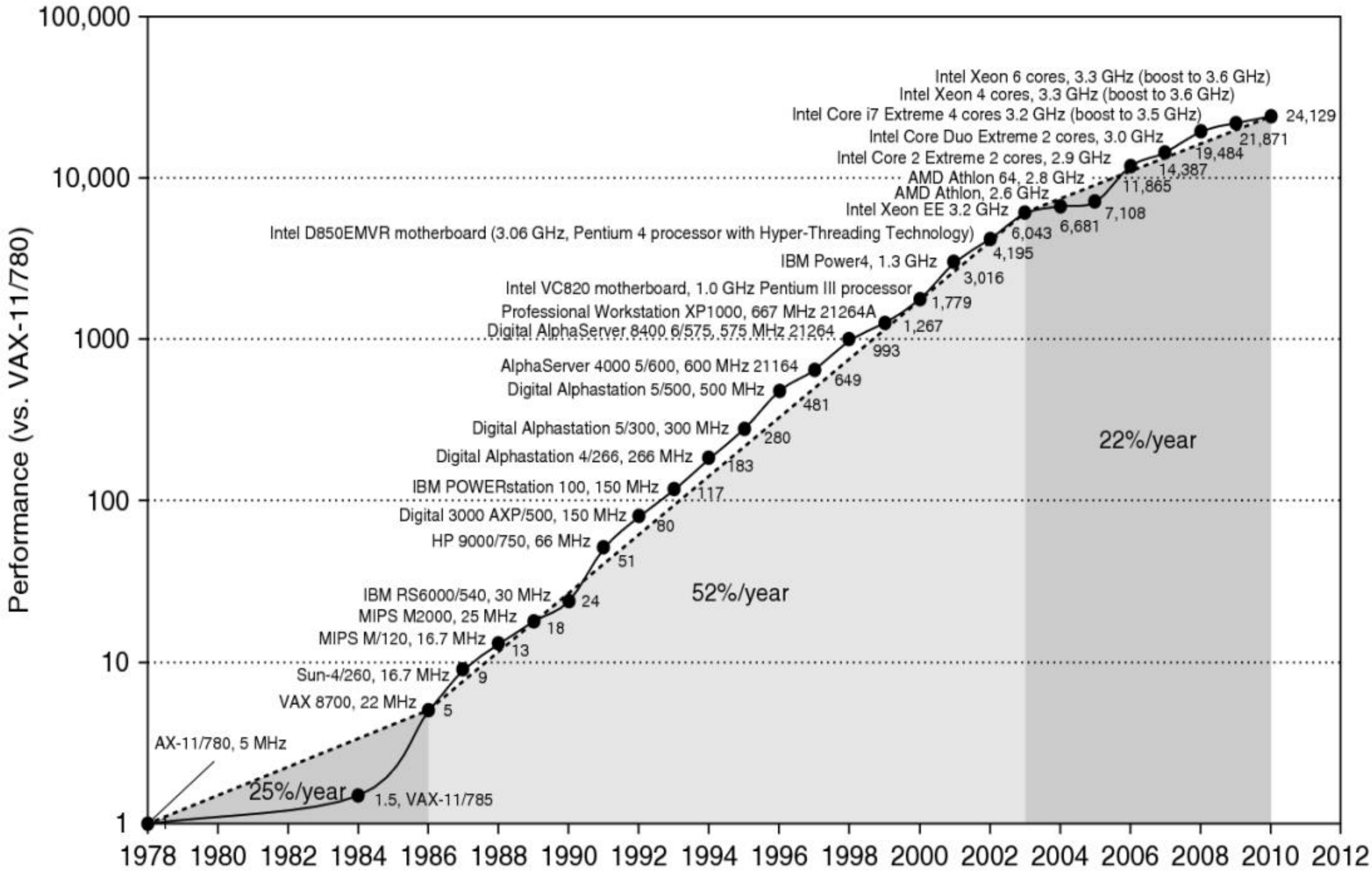
Moore reformulates to a doubling every 2 years. (1975)
Interview 2000:
”...change the doubling time again... to maybe four or five years.”

Gordon Moore
Co-founder of Intel



ca. 1 billion transistors in 2007

Performance of Microprocessor



Performance of Car 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.

Bill Gates: "if GM had kept up with the technology like the computer industry has, we would all be driving $25.00 cars that got 1,000 miles to the gallon."

If automobile speed had increased at the same speed as clock frequency, you could now drive from San Francisco to New York in about 13 seconds!

Performance of Car Compared

In response to Bill's 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 in the road, you would have to buy a new car.

3. Occasionally your car would die on the freeway for no reason. You would have to pull over to the side of the road, close all of the windows, shut off the car, restart it, and reopen the windows before you could continue.
For some reason you would simply accept this.

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 "CarNT", but then you would have to buy more seats.

…

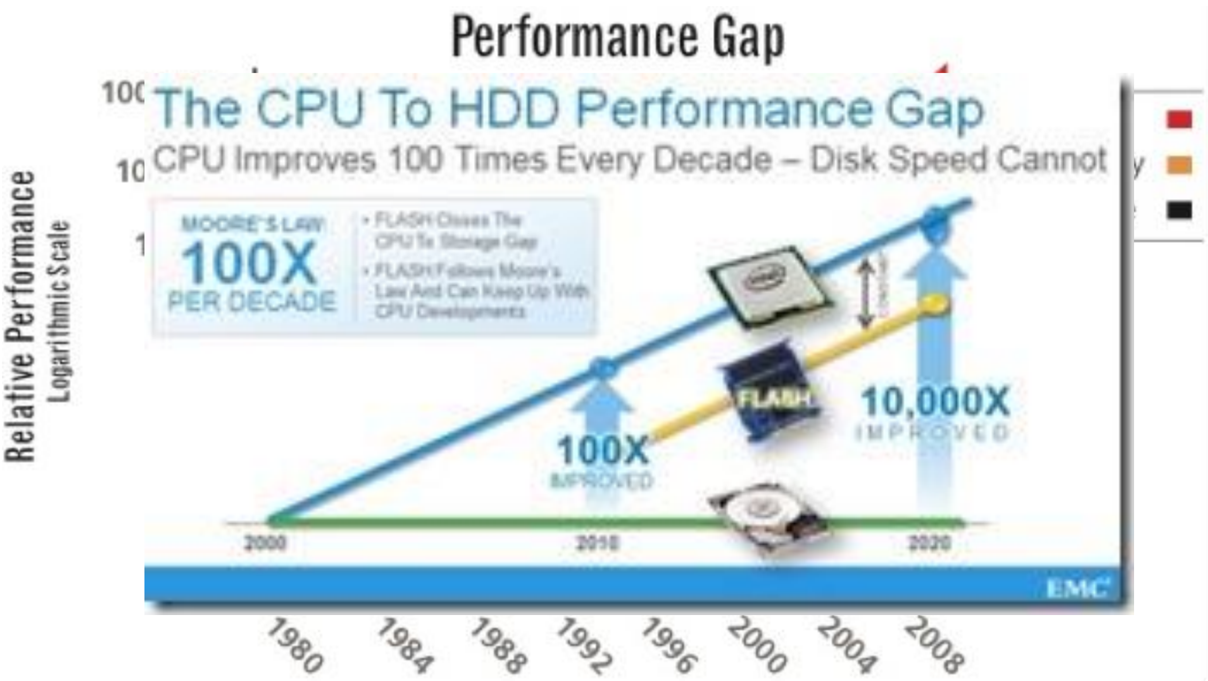
9. The airbag system would ask, "are you sure?" before deploying.

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 grabbed hold of the radio antenna.

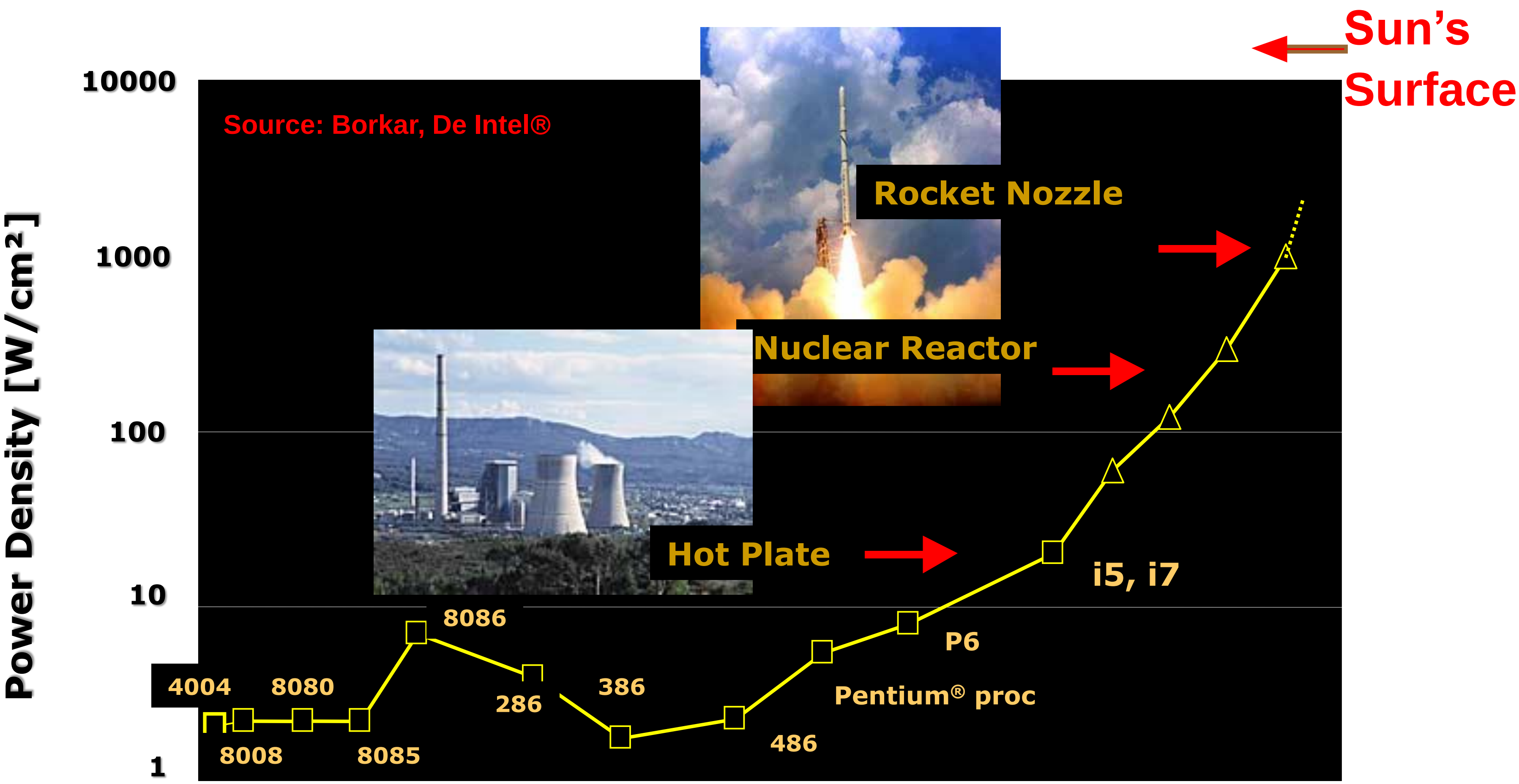
12. Every time GM introduced a new car, car buyers would have to learn to drive all over again because none of the controls would operate in the same manner as the old car.

Does not Apply to All

- 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



Moore's Law: power density



Heating (power) is a issue

TECHNOLOGY

Qualcomm Snapdragon 820 Faces Heating Problems: Report

Qualcomm might run into more trouble, after reports surfaced that the next-gen Snapdragon chipset has overheating issues

- Huawei uses in-house Kirin processors in its flagship devices.
- LG is said to be working on its own in-house ARM processors.
- Samsung covers its high-end needs on its own.
- HTC and Sony are turning to MediaTek in low- and mid-range devices.
- Many Chinese vendors use MediaTek processors in value devices, and some like Meizu use MT chips in flagships.
- MediaTek is introducing increasingly powerful SoCs that can end up in even more flagship designs.

Technology | Mon Jul 20, 2015 6:46pm EDT

Qualcomm preparing to lay off several thousand employees: tech website

Outline

- Computers
- Computer Architecture
- This Course
- Trends
- **Performance**
- Quantitative Principles

What is Performance?

Plane	DC to Paris	Speed
Boeing 747	6.5 h	980 km/h
Concorde	3 h	2160 km/h

- **Time to complete a task (T_{exe})**
  - Execution time, response time, latency
- **Task per day, hour…**
  - Total amount of tasks for given time
  - Thoughput, bandwidth
- **Speed of Concorde vs Boeing 747**
- **Throughput of Boeing 747 vs Concorde**

Performance

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

"X is n times faster than Y" means:

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

How to define execution time?

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

Program to evaluate performance

- **Real programs: e.g. TeX, spice, SPEC benchmarks, ...**
- **Kernels - small, key pieces of real applications**
- **Toy programs - sort, prime number generation**
  - Something 100-line programs
- **Synthetic benchmarks - “The average program”**
  - Fake programs to mathc the behaviour of real applications
  - Real programs are the only true measurement objects
- **SPEC benchmarks will be used here (*plus some toy programs*)**
  - Real programs modified to be portable and ti minimize the effect of IO

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 test**

Which Computer is Faster?

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**

Which Computer is Faster?

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

- 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

- Computers
- Computer Architecture
- This Course
- Trends
- Performance
- **Quantitative Principles**

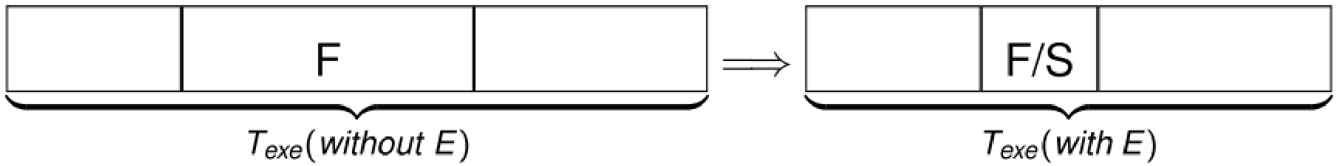
Quantitative Principles

This is intro to design and analysis

- Take advantage of parallelism
  - ILP, DLP, TLP, ...
- Principle of locality
  - 90% of execution time in only 10% of the code
- Focus on the common case
  - In makeing a design trade-off, favor the frquent case ove the infrequent case
- Amdahl’s Law
  - The performance improvement gained from uisng faster mode is limited by the fraction of the time the faster mode can be used
- The Processor Performance Equation

Amdahl's Law

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



Speedup due to enhancement E:

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

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

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

Best you could ever hope to do:

$$Speedup_{maximum} = \frac{1}{(1 - Fraction_{enhanced})}$$

Amdahl's Law: example

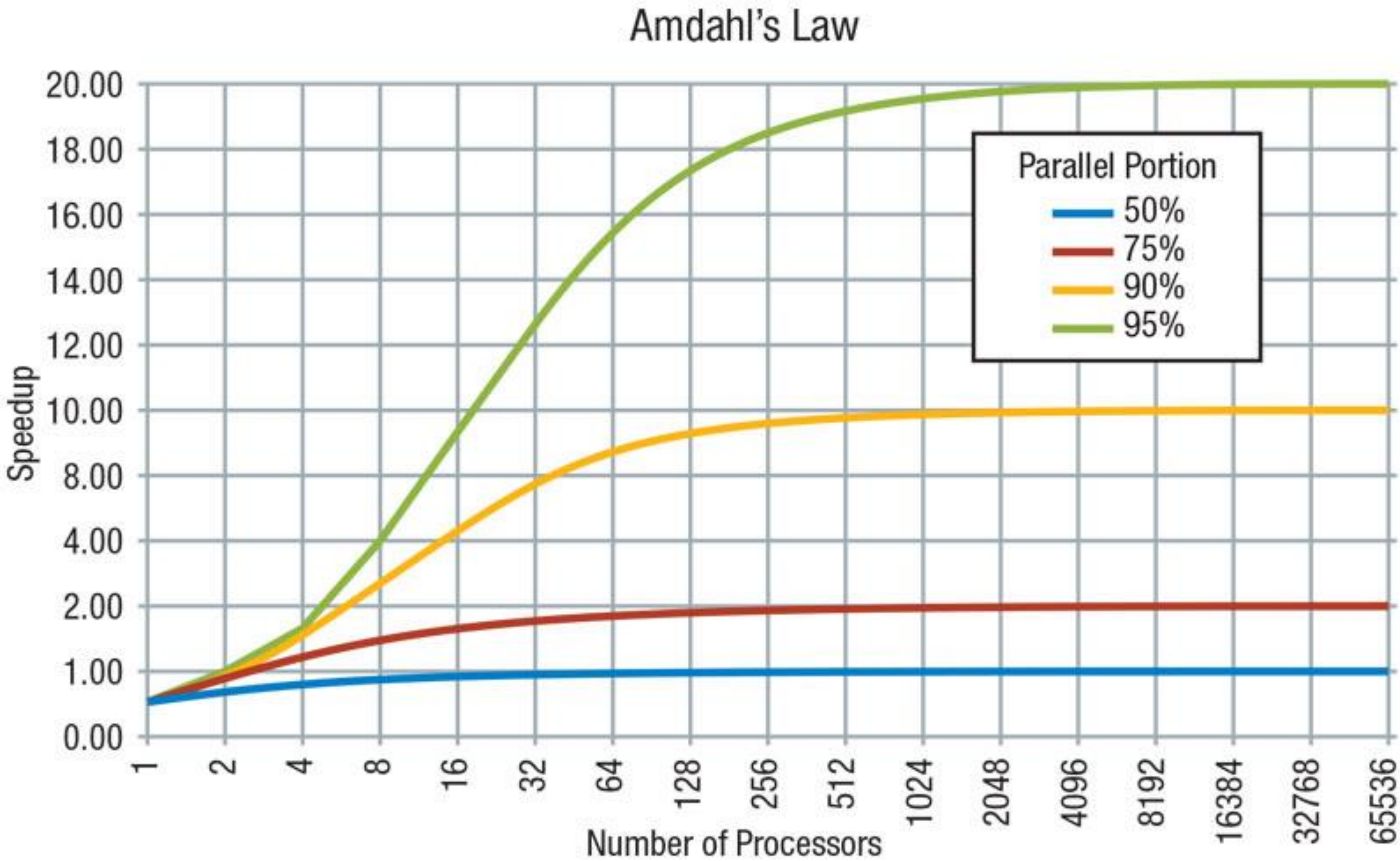
- New CPU is 10 times faster!
- 60% for I/O which remains almost the same…

$$\text{Speedup}_{\text{overall}} = \frac{1}{\left(1-\text{Fraction}_{\text{enhanced}}\right)+\dfrac{\text{Fraction}_{\text{enhanced}}}{\text{Speedup}_{\text{enhanced}}}}$$

$$= \frac{1}{\left(1-0.4\right)+\dfrac{0.4}{10}} = \frac{1}{0.64} = 1.56$$

Apparently, its human nature to be attracted by 10X faster, vs. keeping in perspective its just 1.6X faster

Amdahl's Law: example



http://www-inst.eecs.berkeley.edu/~n252/paper/Amdahl.pdf

Aspect of CPU performance

$$\text{CPUtime} = \text{Execution time} =$$
$$seconds/program =$$

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

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

Instructions are not created equally

"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"

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

Average CPI: example

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 %)

$\overline{CPI} = 1.5$

Invest resources where time is spent!