Chapter 1: Measuring & Understanding Performance

JOSÉ Mª GONZÁLEZ LINARES

DEPT. DE ARQUITECTURA DE COMPUTADORES

UNIVERSIDAD
DE MÁLAGA

E.T.S.
INGENIERÍA
INFORMÁTICA

INDEX

INTRODUCTION

THE COMPUTER REVOLUTION

What is a computer?

A computer is a general purpose device that can be programmed to carry out a finite set of arithmetic or logical operations.

- We use programs, i.e., sequences of instructions, to solve problems.
- The Operating System help us to code and execute our programs.

Classes of Computers

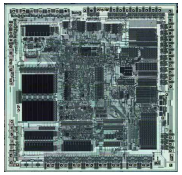
Desktop computers
- General purpose, variety of software, graphics display, keyboard and mouse
- Subject to cost/performance tradeoff

Server computers
- Network based
- High capacity, dependability, security

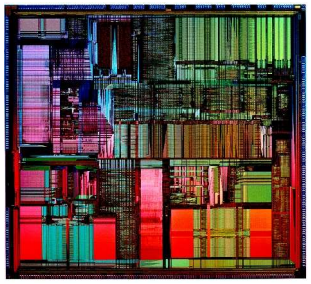
Supercomputers
- Hundreds to thousands processors, terabytes of memory, petabytes of storage
- High performance, cost

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

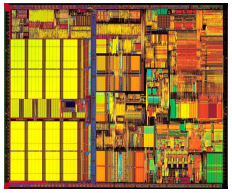
EVOLUTION OF PROCESSORS FOR DESKTOPS AND LAPTOPS



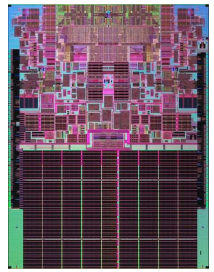
Intel 80286 (1982)
134×10^3 transistors
12 Mhz, 68,7 mm^2



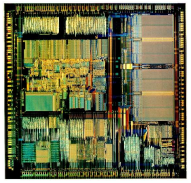
Intel Pentium (1993)
3.1×10^6 transistors
66 Mhz, 264 mm^2



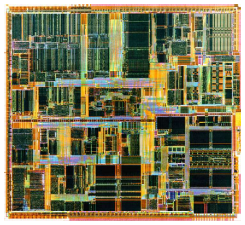
Intel Pentium III (1999)
28×10^6 transistors
733 Mhz, 140 mm^2



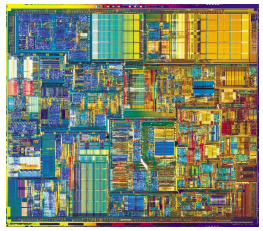
Intel Core (2006)
291×10^6 transistors
1 Ghz, 143 mm^2



Intel80386 (1985)
275×10^3 transistors
33 Mhz, 104 mm



Intel Pentium II (1997)
7.5×10^6 transistors
300 Mhz, 209 mm^2



Intel Pentium 4 (2000)
42×10^6 transistors
1.5 Ghz, 224 mm^2

CURRENT PROCESSORS FOR DESKTOPS AND LAPTOPS

Intel 12th Gen
Alder Lake

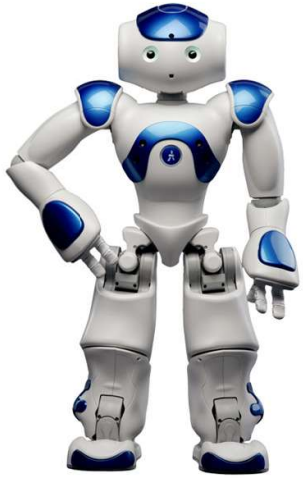
Process of 10nm and less, high-performance and power-efficient cores, Out-of-Order execution, TDP between 4 and 100 W, three levels of cache, integrated GPU, …

SUPERCOMPUTERS

The top500 list in June 2022

Rank	System	Cores	Rmax (PFlop/s)	Rpeak (PFlop/s)	Power (kW)
1	**Frontier** - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE DOE/SC/Oak Ridge National Laboratory United States	8,730,112	1,102.00	1,685.65	21,100
2	**Supercomputer Fugaku** - Supercomputer Fugaku, A64FX 48C 2.2GHz, Tofu interconnect D, Fujitsu RIKEN Center for Computational Science Japan	7,630,848	442.01	537.21	29,899
3	**LUMI** - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE EuroHPC/CSC Finland	1,110,144	151.90	214.35	2,942
4	**Summit** - IBM Power System AC922, IBM POWER9 22C 3.07GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband, IBM DOE/SC/Oak Ridge National Laboratory United States	2,414,592	148.60	200.79	10,096
5	**Sierra** - IBM Power System AC922, IBM POWER9 22C 3.1GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband, IBM / NVIDIA / Mellanox DOE/NNSA/LLNL United States	1,572,480	94.64	125.71	7,438

WHERE ELSE ARE EMBEDDED PROCESSORS FOUND?



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

Moore's Law

2017: Quad Core Kaby Lake with 3.2B transistors

2019: Epyc Rome with 39.5B transistors

2022: Apple M1 Ultra with 114B transistors

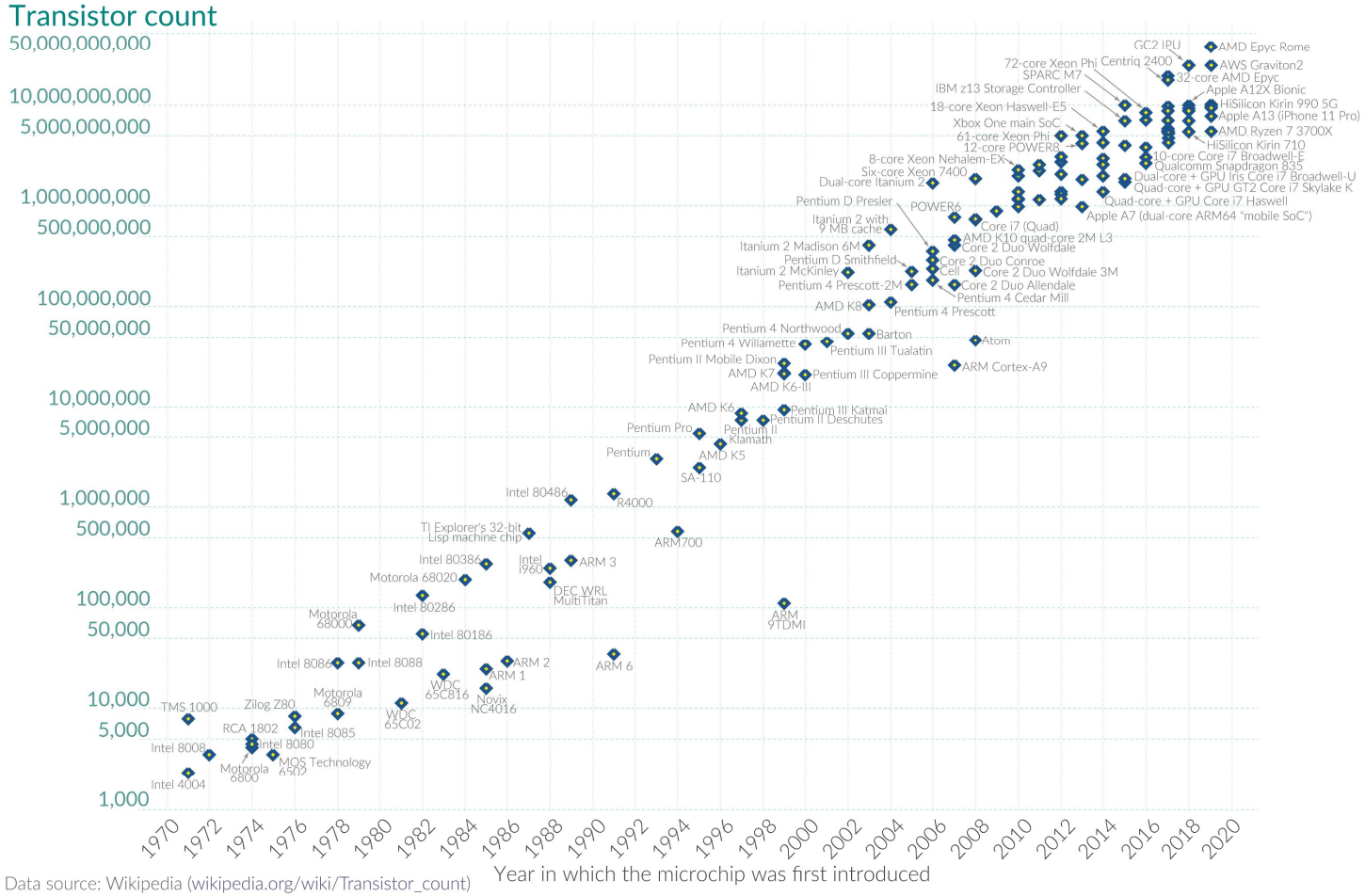
2020: GA100 Ampere Nvidia GPU with 54B transistors

2022: GH100 Hopper Nvidia with 80B transistors

Moore's Law: The number of transistors on microchips doubles every two years

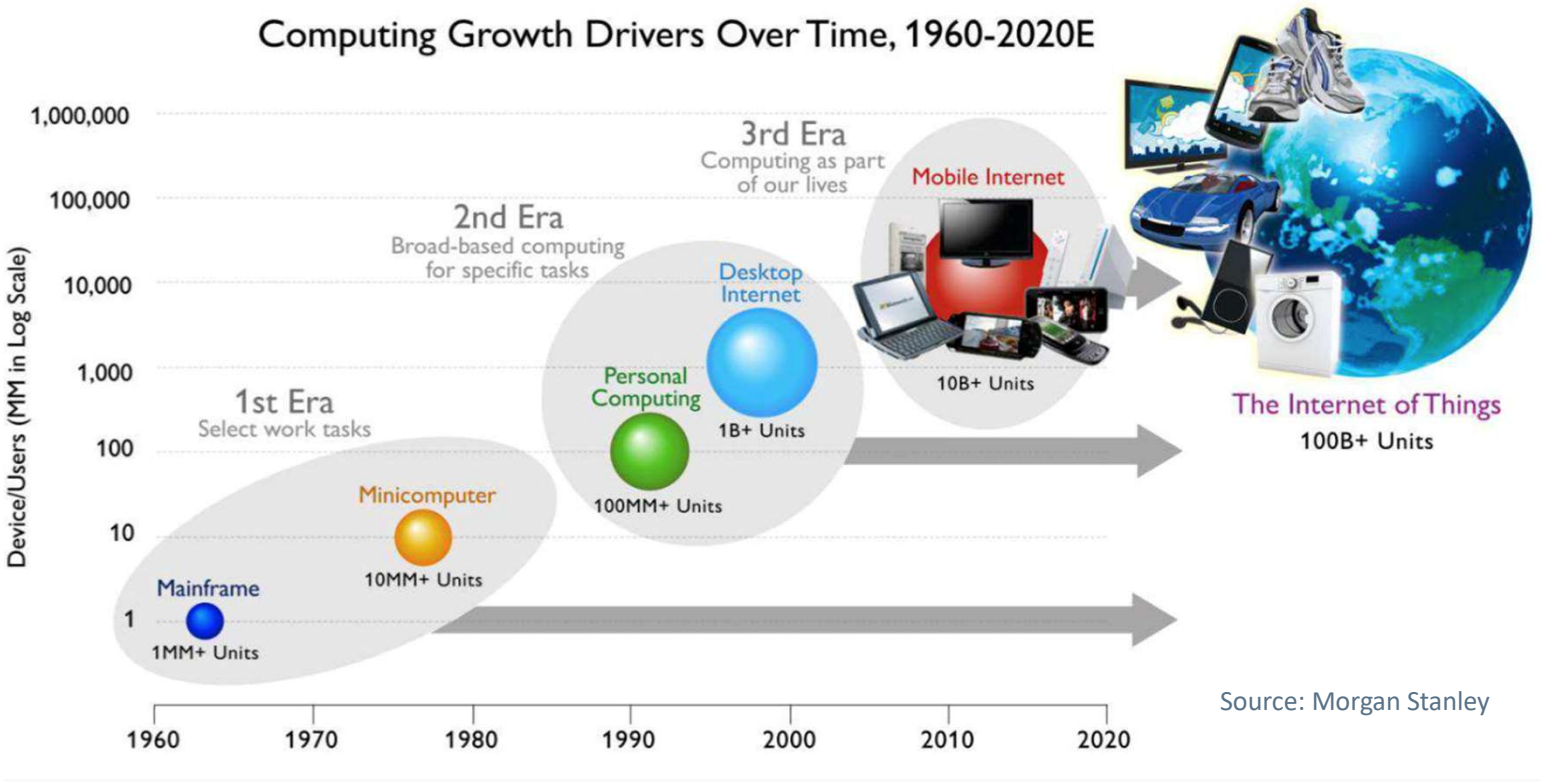
Our World in Data

Moore's law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years.
This advancement is important for other aspects of technological progress in computing – such as processing speed or the price of computers.



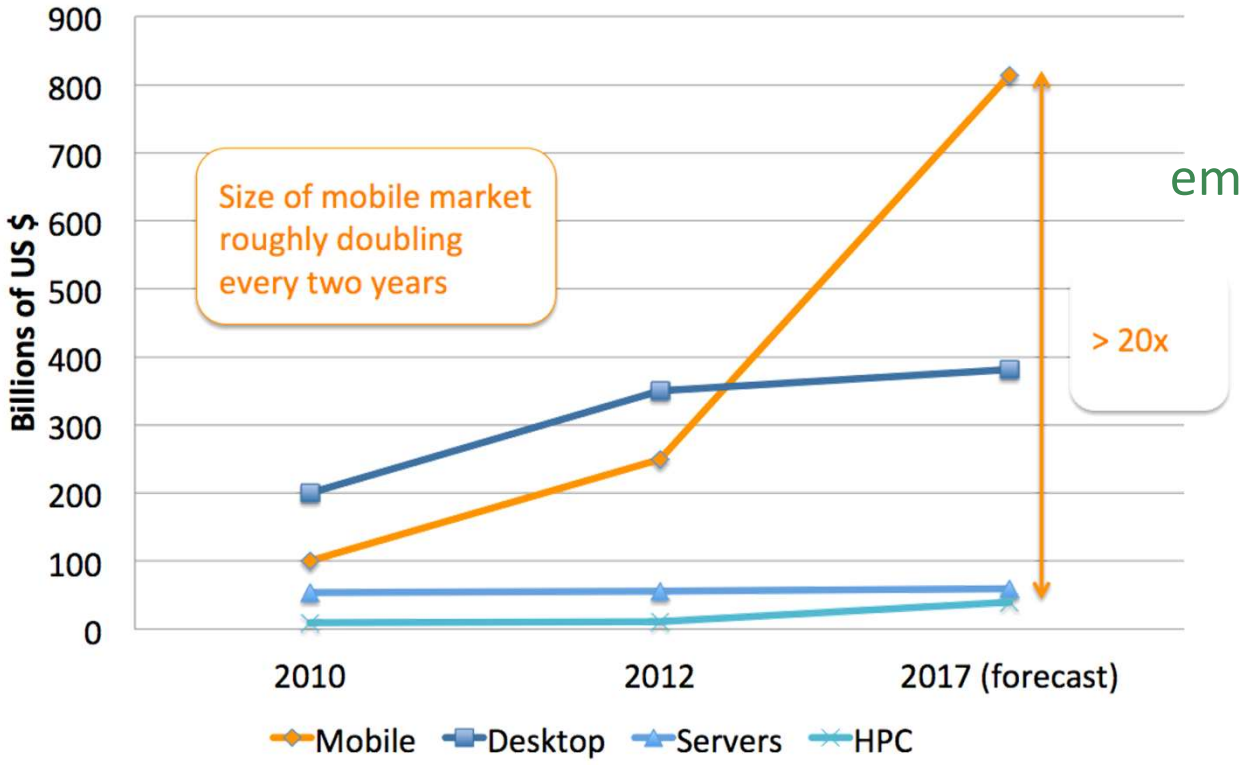
Data source: Wikipedia (wikipedia.org/wiki/Transistor_count)
OurWorldinData.org – Research and data to make progress against the world's largest problems.

CONNECTIVITY DRIVING COMPUTING



Source: Morgan Stanley

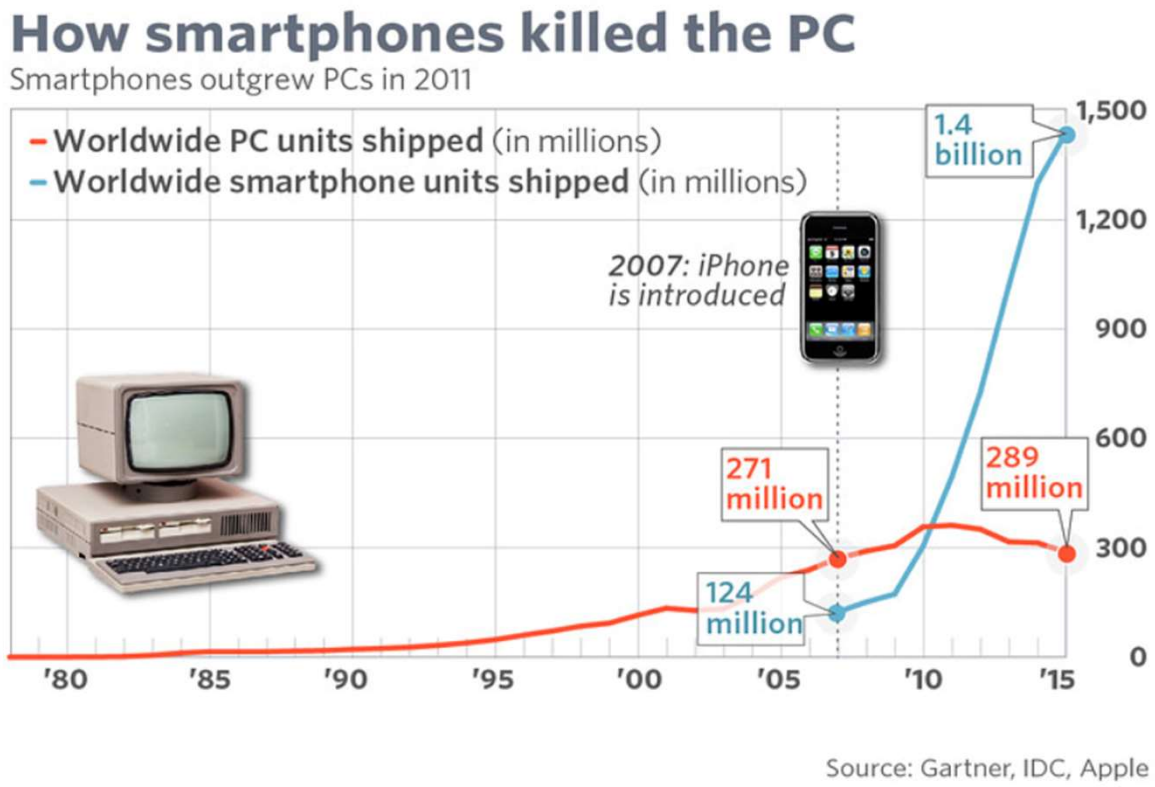
The Processor Market



embedded growth >> desktop growth

Source: IDC. HPC forecast Intersect360

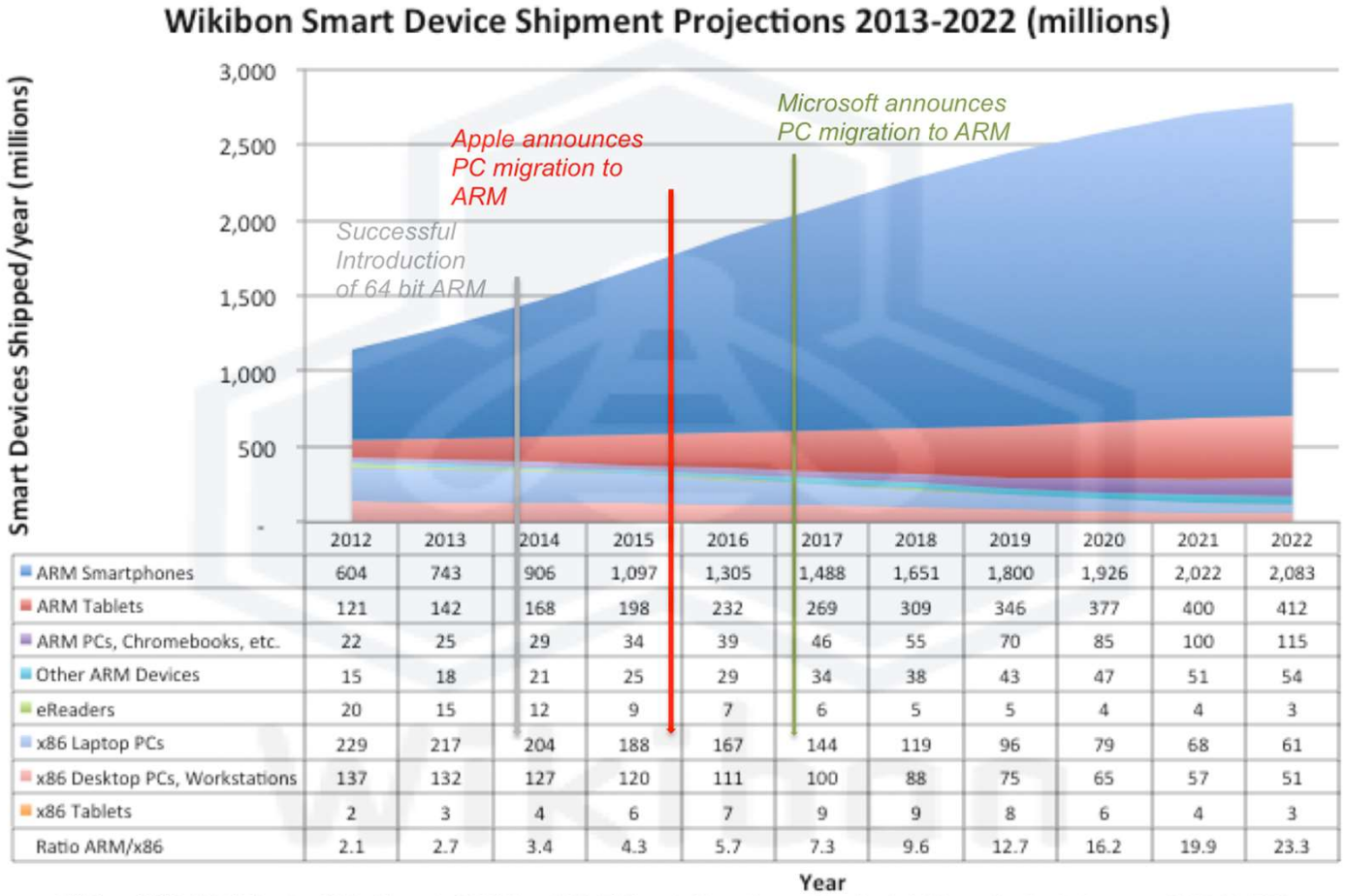
SMARTPHONE INDUSTRY EVOLUTION



4bn people buying
every 2 years
instead of 1.6bn
every 5 years

TECNOLOGY SHIFT

ARM has cannibalized the processor market



	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021	2022
ARM Smartphones	604	743	906	1,097	1,305	1,488	1,651	1,800	1,926	2,022	2,083
ARM Tablets	121	142	168	198	232	269	309	346	377	400	412
ARM PCs, Chromebooks, etc.	22	25	29	34	39	46	55	70	85	100	115
Other ARM Devices	15	18	21	25	29	34	38	43	47	51	54
eReaders	20	15	12	9	7	6	5	5	4	4	3
x86 Laptop PCs	229	217	204	188	167	144	119	96	79	68	61
x86 Desktop PCs, Workstations	137	132	127	120	111	100	88	75	65	57	51
x86 Tablets	2	3	4	6	7	9	9	8	6	4	3
Ratio ARM/x86	2.1	2.7	3.4	4.3	5.7	7.3	9.6	12.7	16.2	19.9	23.3

Source: Wikibon 2013, IDC & Gartner 2012 shipments & Wikibon 2013-2022 projections. Assumption: Apple & Microsoft migrate to successful 64-bit ARM.

What Will You Learn?

How programs are translated into the machine language
- And how the hardware executes them

The hardware/software interface

What determines program performance
- And how it can be improved

How hardware designers improve performance

How software designers improve performance
- And what is parallel processing

Is it useful?

Marcos Fajardo wrote from scratch a rendering system, Arnold, that runs natively on x86 CPUs, where it tries to take advantage of all available threads and SIMD lanes for optimal parallelism.

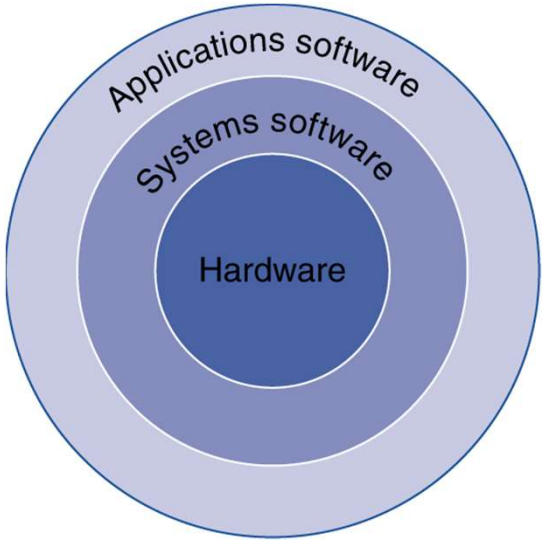
It is used in

- Thor
- Captain America
- X-Men: First Class
- The Avengers
- Gravity
- Guardians of the Galaxy
- Star Wars: The Force Awakens
- Game of Thrones
- …

BELOW YOUR PROGRAM

UNDER THE COVERS

Below Your Program



Application software

- Written in high-level language (HLL)

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, Input/Output (I/O) controllers

Levels of Program Code

High-level language

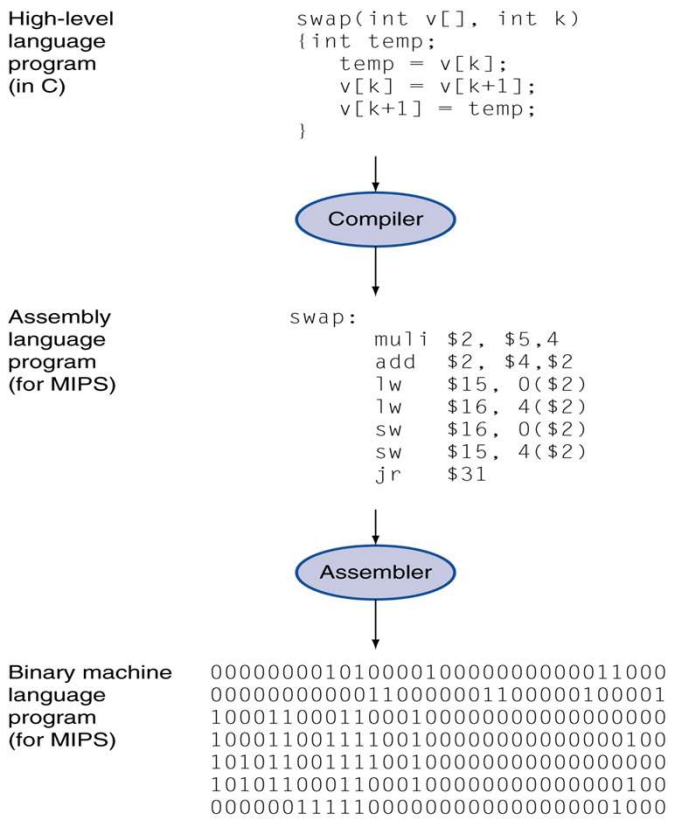
- Level of abstraction closer to problem domain
- Provides for productivity and portability

Assembly language

- Textual representation of instructions

Hardware representation

- Binary digits (bits)
- Encoded instructions and data



Abstraction

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

Under the Covers

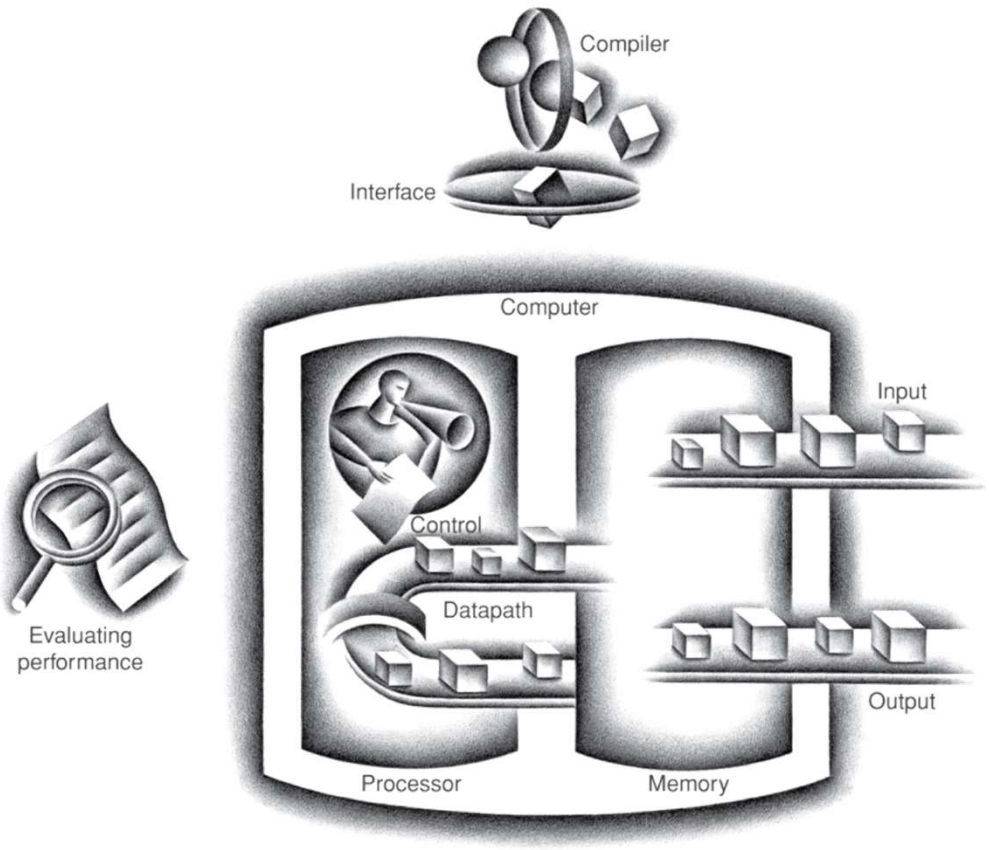
Same components for all kinds of computer

- Five classic components of a computer – input, output, memory, datapath, and control

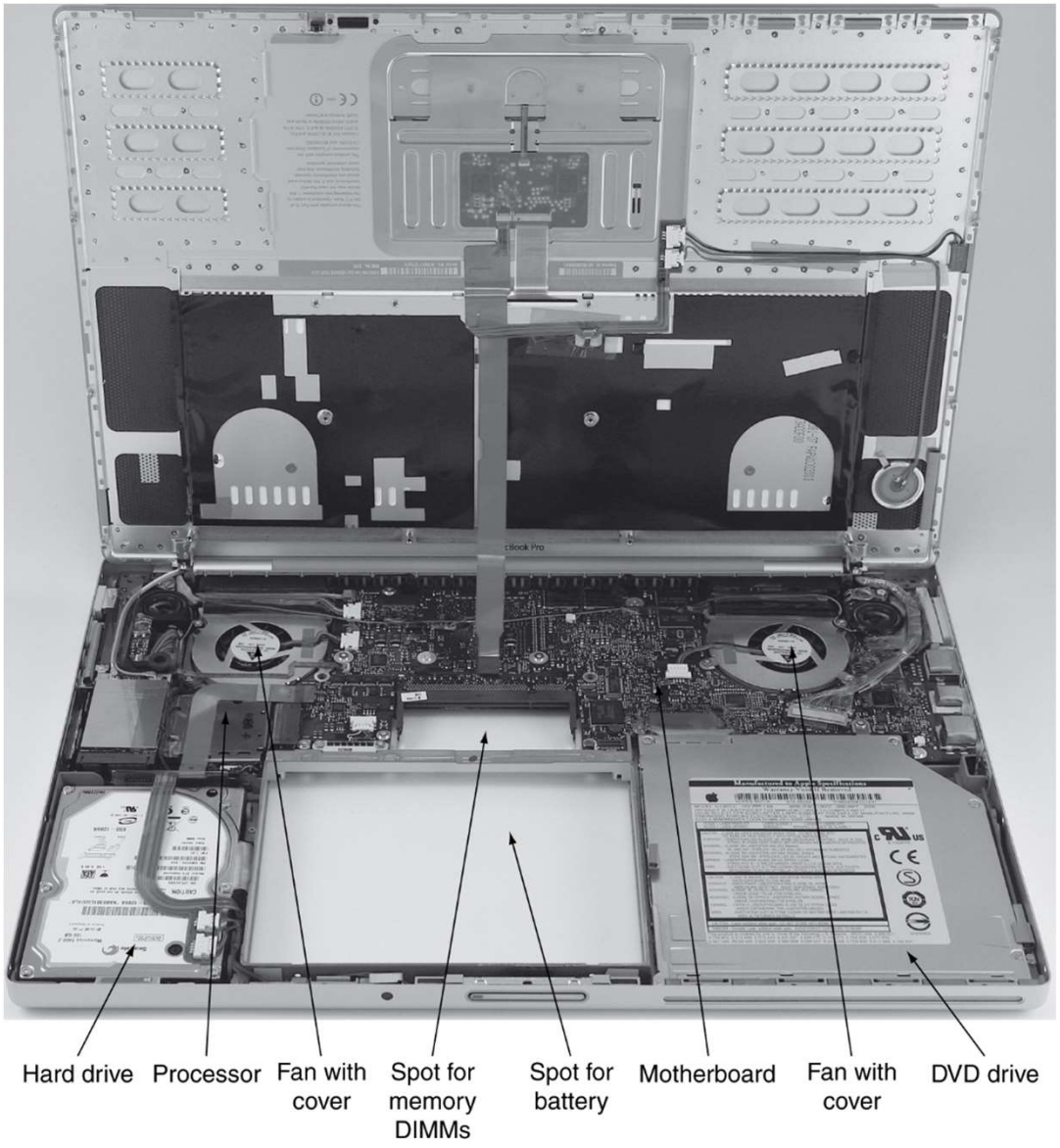
datapath + control = processor (CPU)

Input/output includes

- User-interface devices
  - Display, keyboard, mouse
- Storage devices
  - Hard disk, CD/DVD, flash
- Network adapters
  - For communicating with other computers



Opening the Box



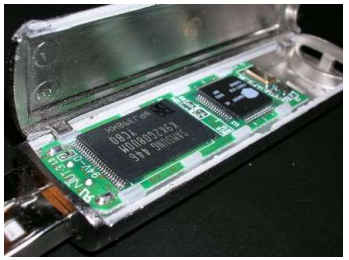
A Safe Place for Data

Volatile main memory

- Loses instructions and data when power off

Non-volatile secondary memory

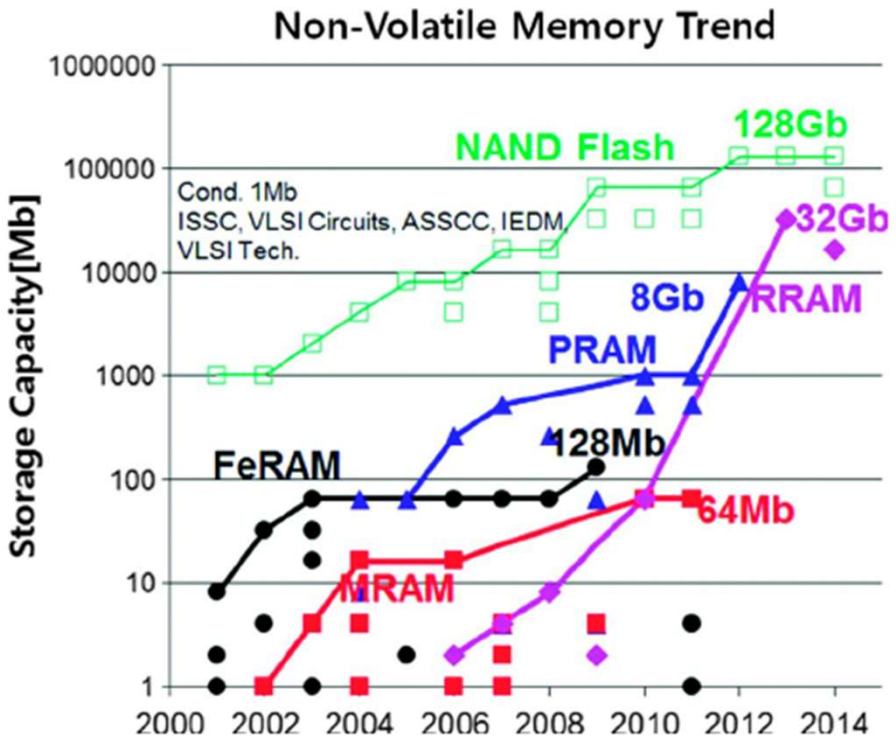
- Magnetic disk
- Optical disk (CDROM, DVD)
- SSD
- Flash memory



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
2005	Ultra large scale IC	6,200,000,000

NETWORKS

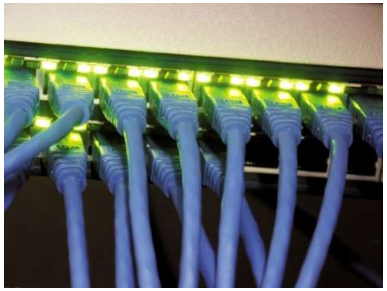
Communication and resource sharing

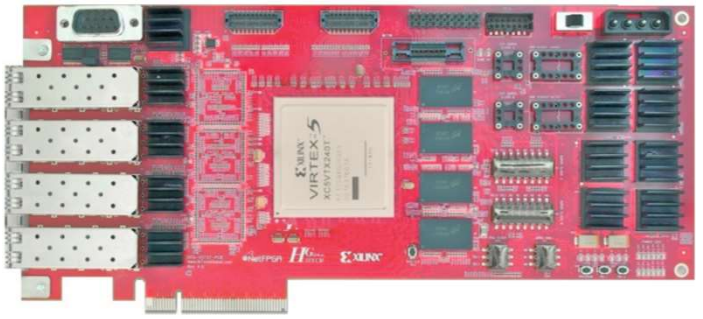
Local area network (LAN): Ethernet

- Within a building

Wide area network (WAN): the Internet

Wireless network: WiFi, Bluetooth





NetFPGA 10 Gigabit Ethernet card

Technology Scaling Road Map (ITRS)

Year	2006	2008	2010	2012	2014	2017
Feature size (nm)	65	45	32	22	14	10
Intg. Capacity (BT)	0.3	0.7	1.1	1.4	1.9	>3.2

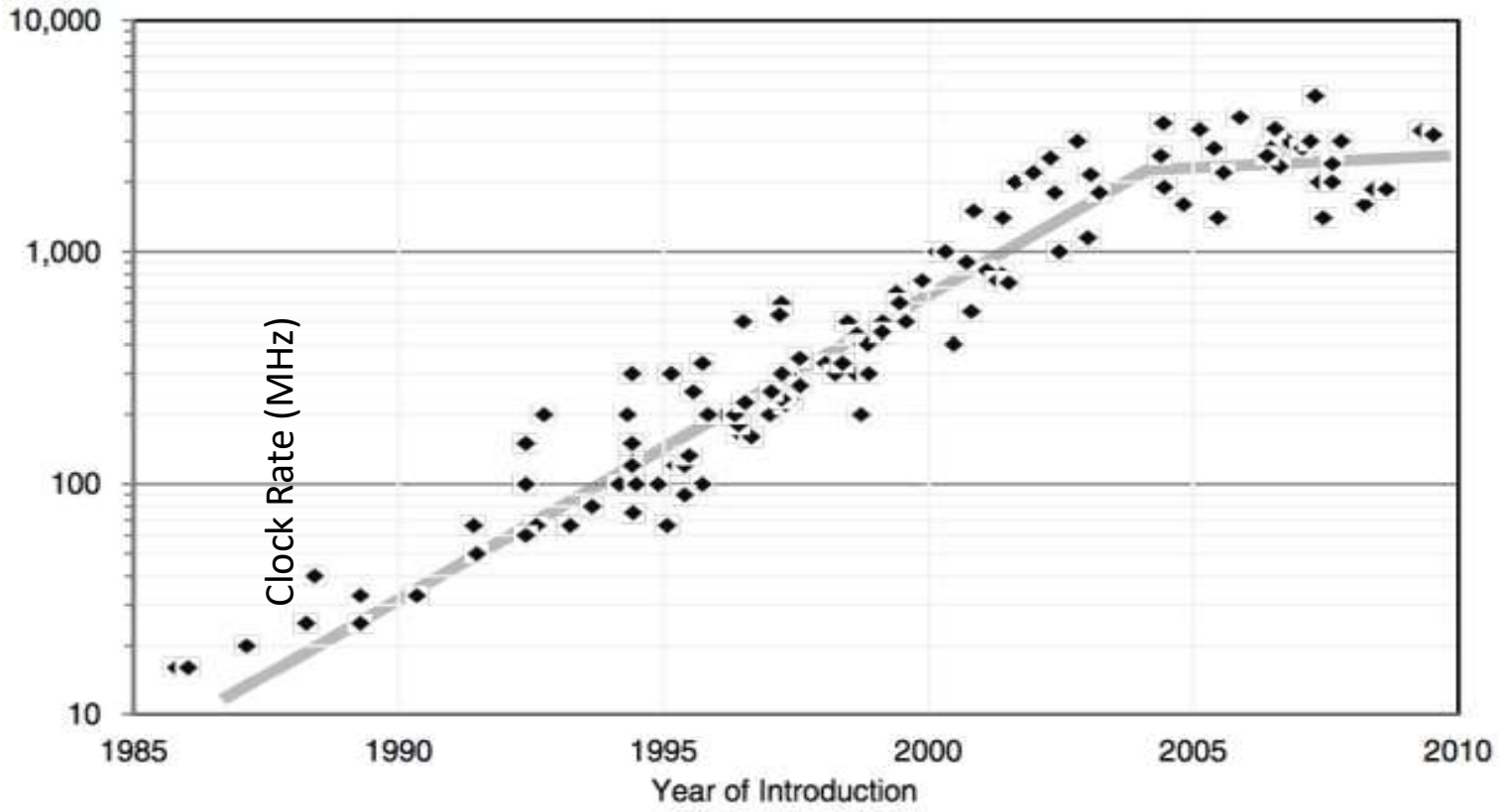
Fun facts about 45nm transistors

- 30 million can fit on the head of a pin
- You could fit more than 2,000 across the width of a human hair

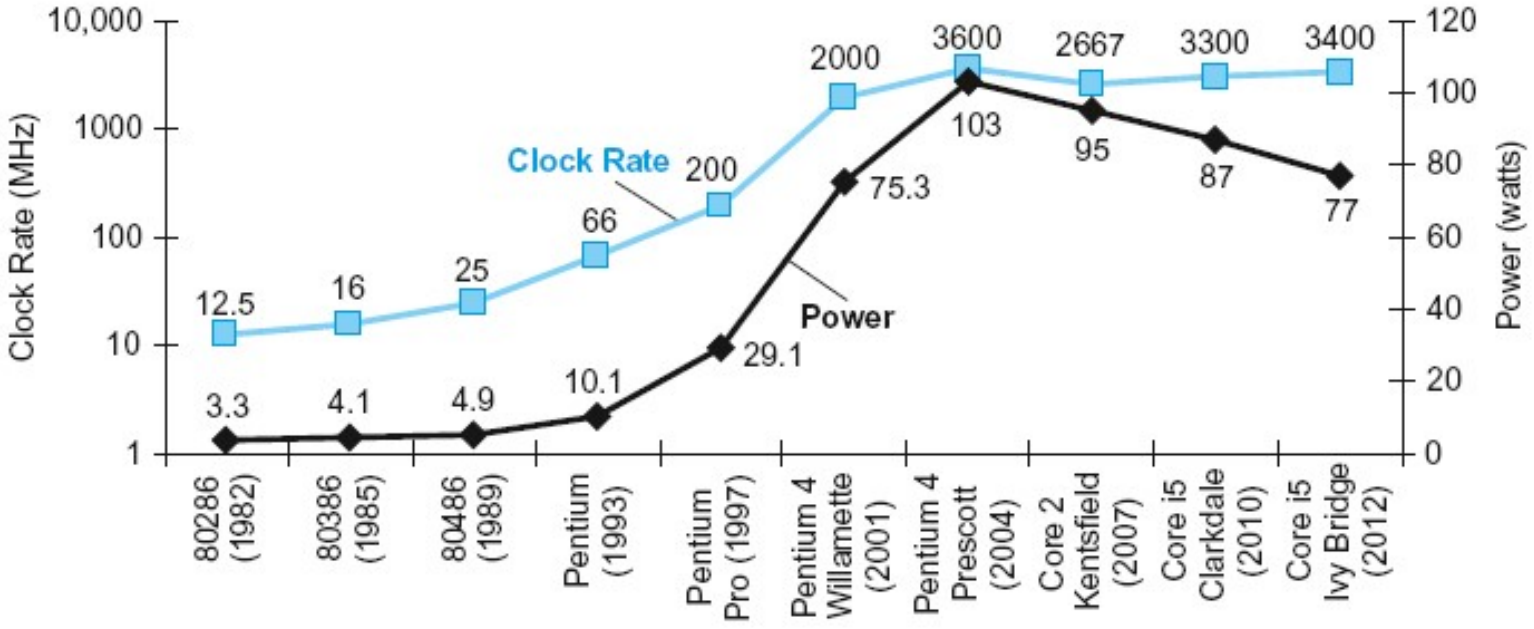
If car prices had fallen at the same rate as the price of a single transistor has since 1968, a new car today would cost about 1 cent

PROCESSOR FREQUENCY

Why processor clock rate have almost not increased in the last ten years?



Power Trends



In CMOS IC technology

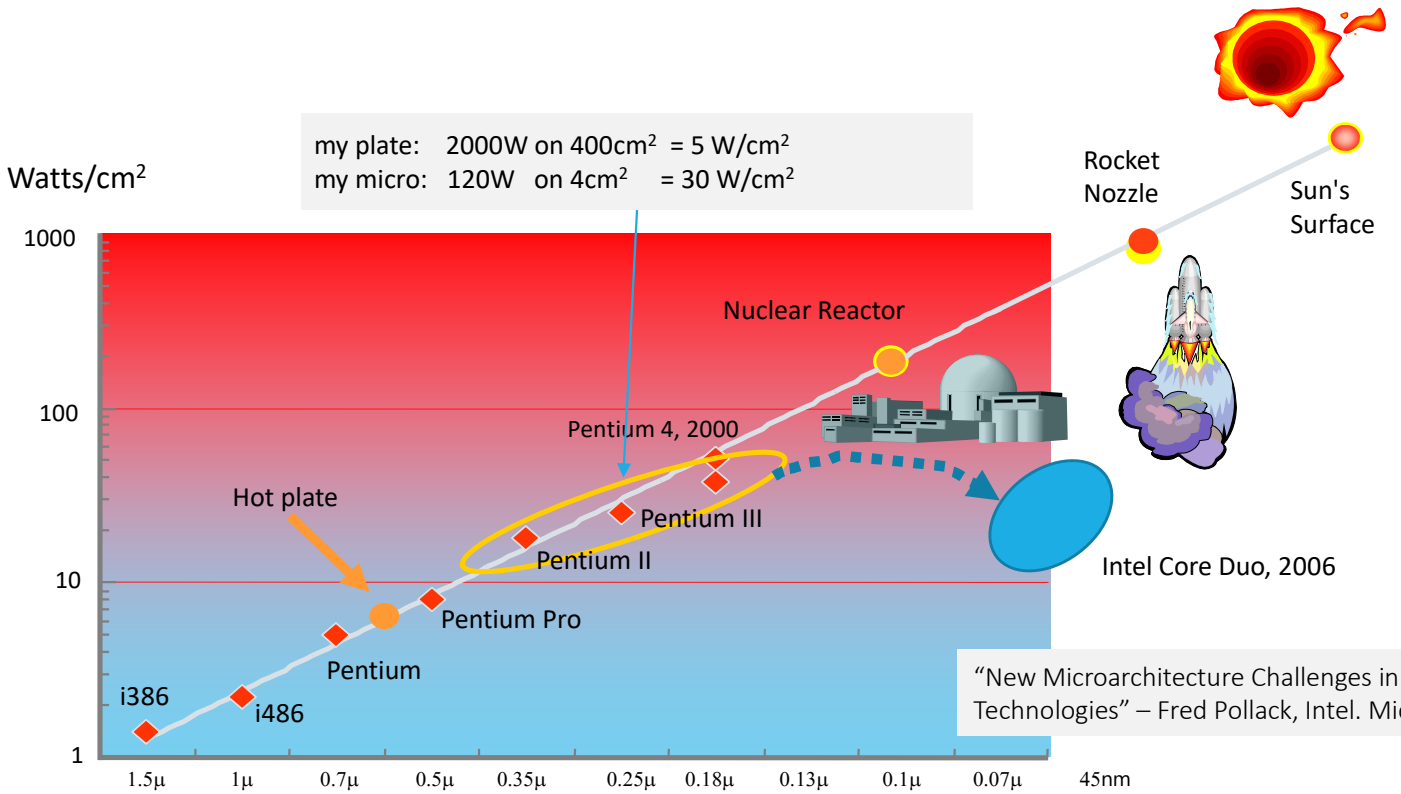
$$\text{Power} = \text{Capacitive load} \times \text{Voltage}^2 \times \text{Frequency}$$

×30

5V → 1V

×1000

What Happened to Clock Rates and Why?



"New Microarchitecture Challenges in the Coming Generations of CMOS Process Technologies" – Fred Pollack, Intel. Micro32 Conf. keynote. 1999.

Reducing Power

Suppose a new CPU has

- 85% of capacitive load of old CPU
- 15% voltage and 15% frequency reduction

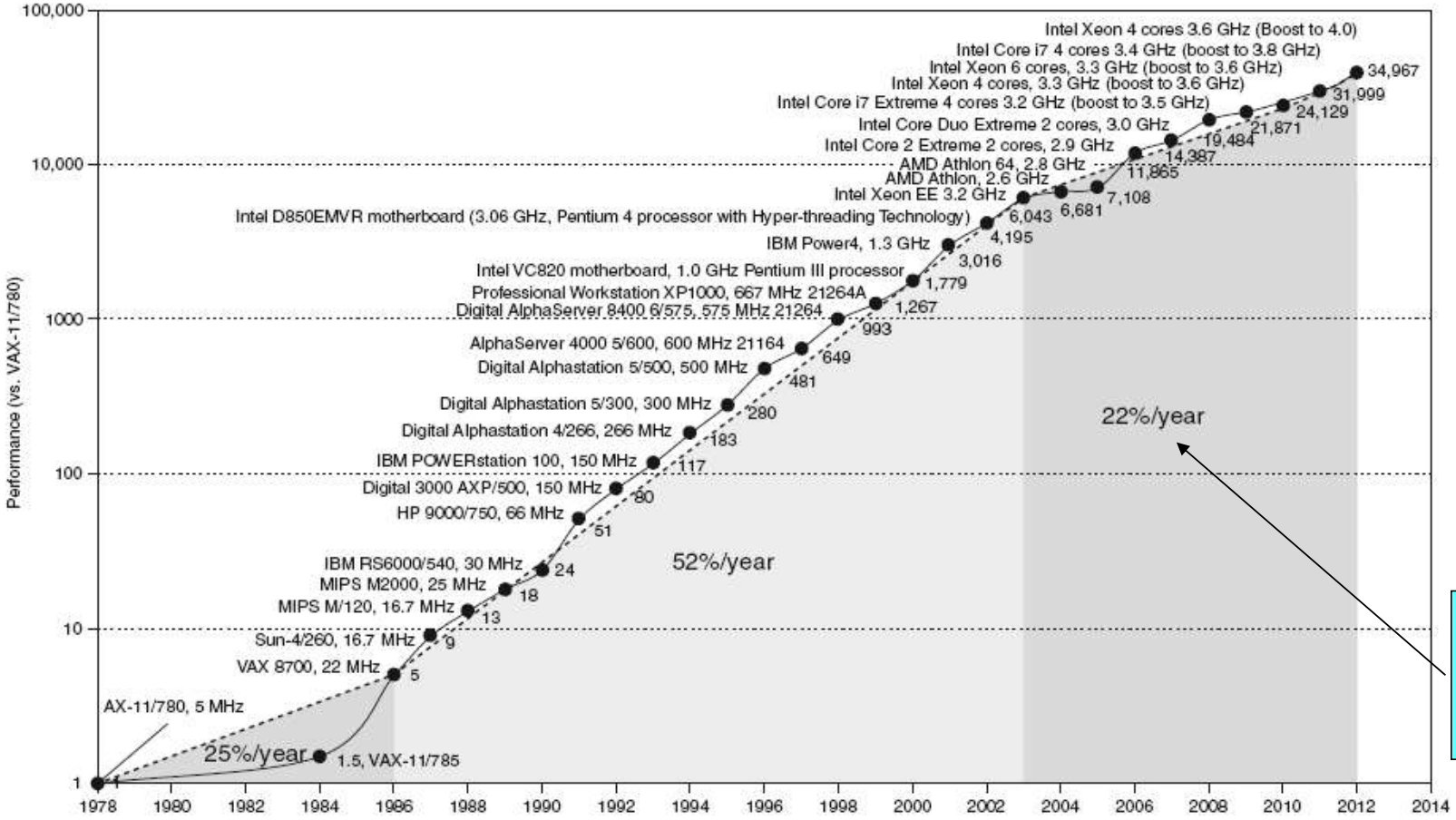
$$\frac{P_{new}}{P_{old}} = \frac{C_{old} \times 0.85 \times (V_{old} \times 0.85)^2 \times F_{old} \times 0.85}{C_{old} \times V_{old}^{\ 2} \times F_{old}} = 0.85^4 = 0.52$$

But a power wall has been reached

- We can't reduce voltage further
- We can't remove more heat

How else can we improve performance?

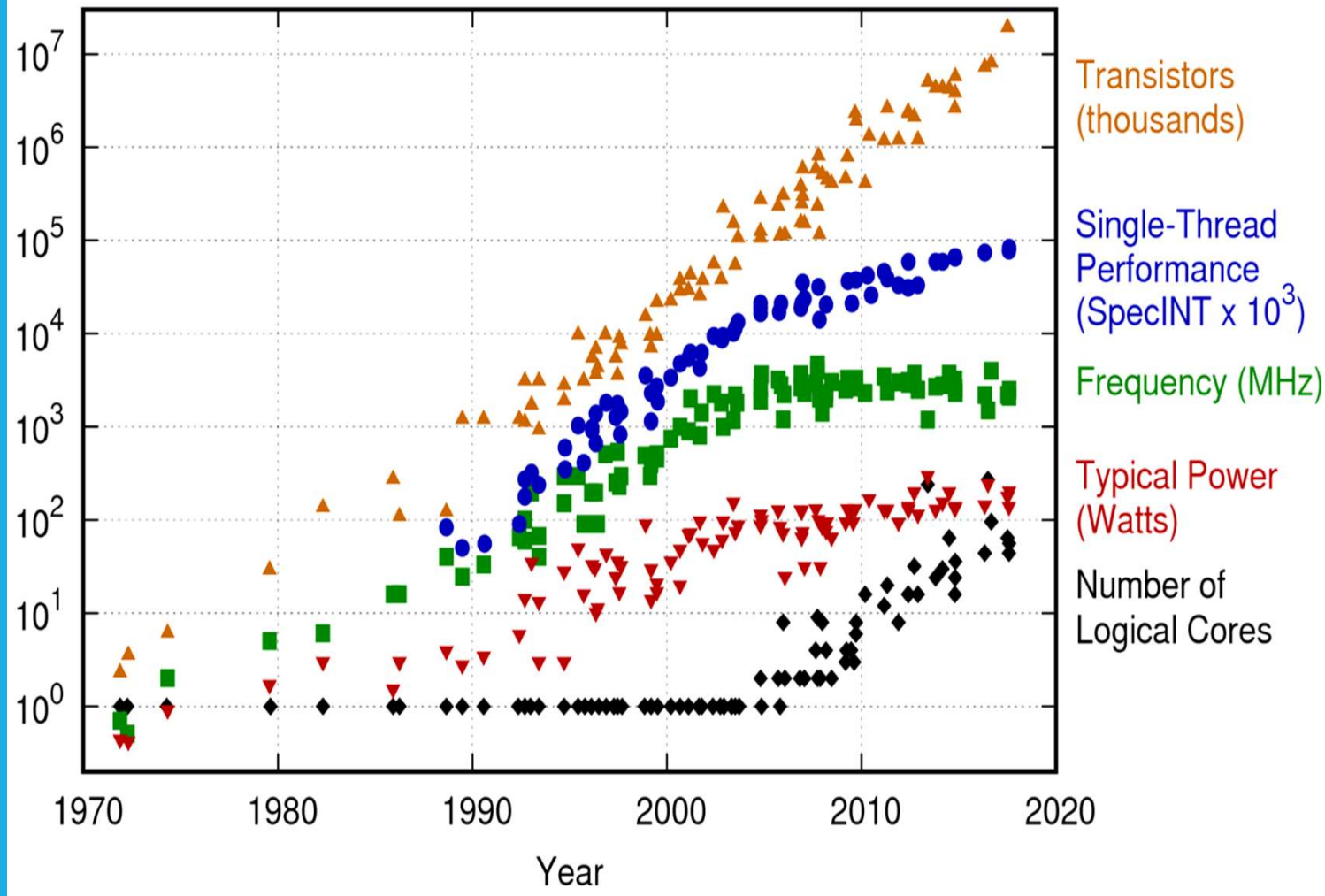
UNIPROCESSOR PERFORMANCE



Constrained by power, instruction-level parallelism, memory latency

THE SEA CHANGE: MULTIPROCESSORS

42 Years of Microprocessor Trend Data (Feb. 15, 2018)



Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2017 by K. Rupp

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

The Sea Change: multiprocessors

The power challenge has forced a change in the design of microprocessors

- Since 2002 the rate of improvement in the response time of programs on desktop computers has slowed from a factor of 1.5 per year to less than a factor of 1.2 per year

As of 2006 all desktop and server companies are shipping microprocessors with multiple processors – cores – per chip

Product	AMD Ryzen	Intel Kaby Lake	Samsung Exynos Octa	Qualcomm Snapdragon
Cores per chip	4-16 + GPU	4 + GPU	4 + 4	4 + GPU +DSP
Clock rate	2.0-3.8 GHz	1.8-3.0 GHz	1.3-1.8 GHz	2.3 GHz
Power	~12-95 W	~4.5-91W	< 10 W	< 10 W

Plan of record is to double the number of cores per chip per generation (about every two years)

UNDERSTANDING PERFORMANCE

POWER, THROUGHPUT AND RESPONSE TIME

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

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?

RELATIVE PERFORMANCE

Define performance as $^{1}/_{ExecutionTime}$

X is n times faster than Y means

$$\frac{Performance_X}{Performance_Y} = \frac{ExecutionTime_Y}{ExecutionTime_X} = n$$

Example: time taken to run a program

- 10s on A, 15s on B
- $\frac{ExecutionTime_Y}{ExecutionTime_X} = \frac{15s}{10s} = 1.5$
- So A is 1.5 times faster than B

Measuring Execution Time

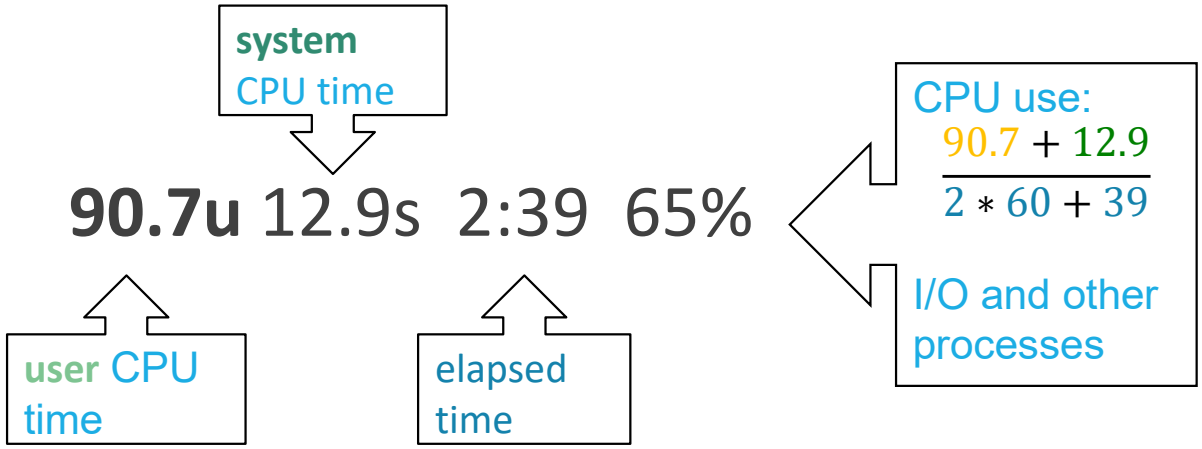
Elapsed time

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

CPU time

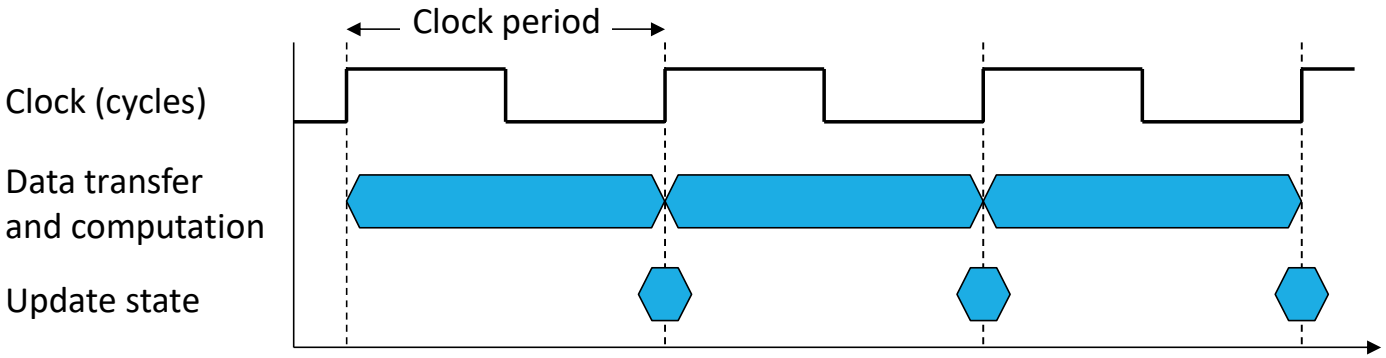
- Time spent processing a given job
  - Discounts I/O time, other jobs' shares
- Comprises user CPU time and system CPU time
- Different programs are affected differently by CPU and system performance

EXAMPLE: 'TIME' IN UNIX



CPU Clocking

Operation of digital hardware governed by a constant-rate clock



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

CPU TIME

$$CPUTime = CPUClockCycles \times ClockCycleTime$$

$$= \frac{CPUClockCycles}{ClockRate}$$

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

$$ClockCycles = InstructionCount \times \underbrace{CyclesPerInstruction}_{\textbf{CPI}}$$

$$CPU\ Time = InstructionCount \times CPI \times ClockCycleTime$$

$$= \frac{InstructionCount \times CPI}{ClockRate}$$

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: ClocCycleTime = 250ps, CPI = 2.0

Computer B: ClockCycle Time = 500ps, CPI = 1.2

Same ISA, which is faster, and by how much?

$$\text{CPU Time}_A = \text{Instruction Count} \times \text{CPI}_A \times \text{ClockCycleTime}_A$$

$$= I \times 2.0 \times 250\text{ps} = I \times 500\text{ps}$$

A is faster…

$$\text{CPU Time}_B = \text{Instruction Count} \times \text{CPI}_B \times \text{ClockCycleTime}_B$$

$$= I \times 1.2 \times 500\text{ps} = I \times 600\text{ps}$$

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

…by this much

CPI in More Detail

If different instruction classes take different numbers of cycles

$$ClockCycles = \sum_{i=1}^{n} (CPI_i \times InstructionCount_i)$$

The weighted average CPI is

$$CPI = \frac{ClockCycles}{InstructionCount} = \sum_{i=1}^{n} \left(CPI_i \times \frac{InstructionCount_i}{InstructionCount} \right)$$

Relative frequency

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

$$\text{CPU Time} = \underbrace{\frac{\text{InstructionCount}}{\text{Program}}}_{\text{IC}} \times \underbrace{\frac{\text{Clock Cycles}}{\text{Instruction}}}_{\text{CPI}} \times \underbrace{\frac{\text{Seconds}}{\text{ClockCycleTime}}}_{\text{CC } (=T_c)}$$

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

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: if multiply accounts for 80s of100s,

- How much improvement in multiply performance is needed to get 5× overall?
- $\frac{100s}{5} = 20s = \frac{80}{n} + 20$ →can´t be done!

TOTAL SPEED-UP

How much speed-up is obtained by improving a part?

$$S = \frac{T_{CPU\ original}}{T_{CPU\ improved}} = \frac{1}{\frac{F_e}{S_e} + (1 - F_e)}$$

where F_e is the fraction of enhancement and S_e is the factor of enhancement.

TOTAL SPEED-UP

How much speed-up is obtained by improving a part?

$$S = \frac{T_{CPU\ original}}{T_{CPU\ improved}} = \frac{1}{\frac{F_e}{S_e} + (1 - F_e)}$$

where F_e is the fraction of enhancement and S_e is the factor of enhancement.

The opportunity for improvement is affected by how much time the modified event consumes

- Corollary 1: make the common case fast

The performance enhancement possible with a given improvement is limited by the amount the improved feature is used

- Corollary 2: the bottleneck will limit the improvement

Pitfall: MIPS as a Performance Metric

MIPS: Millions of Instructions Per Second

$$MIPS = \frac{InstructionCount}{ExecutionTime \times 10^6} = \frac{ClockRate}{CPI \times 10^6}$$

Doesn´t account for

- Differences in ISAs between computers
- Differences in complexity between instructions

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)

$$\sqrt[n]{\prod_{i=1}^{n} ExecutionTimeRatio_i}$$

SPEC CPU 2017

Since July 2017 a new benchmark is available

- A new metric, SPECrate, is added
  - Former metric is called SPECspeed
- It includes 4 new suites:
  - Pairs of int/fp with speed/rate
  - Base and peak metrics can be chosen

Name	Description	IC×10^9	CPI	Tc (ns)	Tcpu(s)	Ref time(s)	SPECratio
perl	Interpreted string processing	2118	0.75	0.40	637	9777	15.3
bzip2	Block-sorting compression	2389	0.85	0.40	817	9650	11.8
gcc	GNU C Compiler	1050	1.72	0.40	722	8050	11.1
mcf	Combinatorial optimization	336	10.00	0.40	1345	9120	6.8
go	Go game (AI)	1658	1.09	0.40	721	10490	14.6
hmmer	Search gene sequence	2783	0.80	0.40	890	9330	10.5
sjeng	Chess game (AI)	2176	0.96	0.40	835	12100	14.5
libquantum	Quantum computer simulation	1623	1.61	0.40	1047	20720	19.8
h264avc	Video compression	3102	0.80	0.40	993	22130	22.3
omnetpp	Discrete event simulation	587	2.94	0.40	690	6250	9.1
astar	Games/path finding	1082	1.79	0.40	773	7020	9.1
xalancbmk	XML parsing	1058	2.70	0.40	1143	6900	6.0
Geometric mean							11.7

High cache miss rates

CINT2006 FOR OPTERON X4 2356

Description	Name	Instruction Count x 10^9	CPI	Clock cycle time (seconds x 10^{-9})	Execution Time (seconds)	Reference Time (seconds)	SPECratio
Interpreted string processing	perl	2252	0.60	0.376	508	9770	19.2
Block-sorting compression	bzip2	2390	0.70	0.376	629	9650	15.4
GNU C compiler	gcc	794	1.20	0.376	358	8050	22.5
Combinatorial optimization	mcf	221	2.66	0.376	221	9120	41.2
Go game (AI)	go	1274	1.10	0.376	527	10490	19.9
Search gene sequence	hmmer	2616	0.60	0.376	590	9330	15.8
Chess game (AI)	sjeng	1948	0.80	0.376	586	12100	20.7
Quantum computer simulation	libquantum	659	0.44	0.376	109	20720	190.0
Video compression	h264avc	3793	0.50	0.376	713	22130	31.0
Discrete event simulation library	omnetpp	367	2.10	0.376	290	6250	21.5
Games/path finding	astar	1250	1.00	0.376	470	7020	14.9
XML parsing	xalancbmk	1045	0.70	0.376	275	6900	25.1
Geometric mean	–	–	–	–	–	–	25.7

CINT2006 FOR INTEL CORE I7 920

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
∑ssj_ops/ ∑power		493

SPECpower_ssj2008 for Xeon X5650

Target Load %	Performance (ssj_ops)	Average Power (Watts)
100%	865,618	258
90%	786,688	242
80%	698,051	224
70%	607,826	204
60%	521,391	185
50%	436,757	170
40%	345,919	157
30%	262,071	146
20%	176,061	135
10%	86,784	121
0%	0	80
Overall Sum	4,787,166	1,922
Σssj_ops/Σpower =		2,490

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

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