Jin-Soo Kim
(*jinsoo.kim@snu.ac.kr*)

Systems Software &
Architecture Lab.

Seoul National University

Fall 2022

Introduction to Computer Architecture

Chap. 1.1 – 1.5, 1.7 – 1.8

Computer Systems













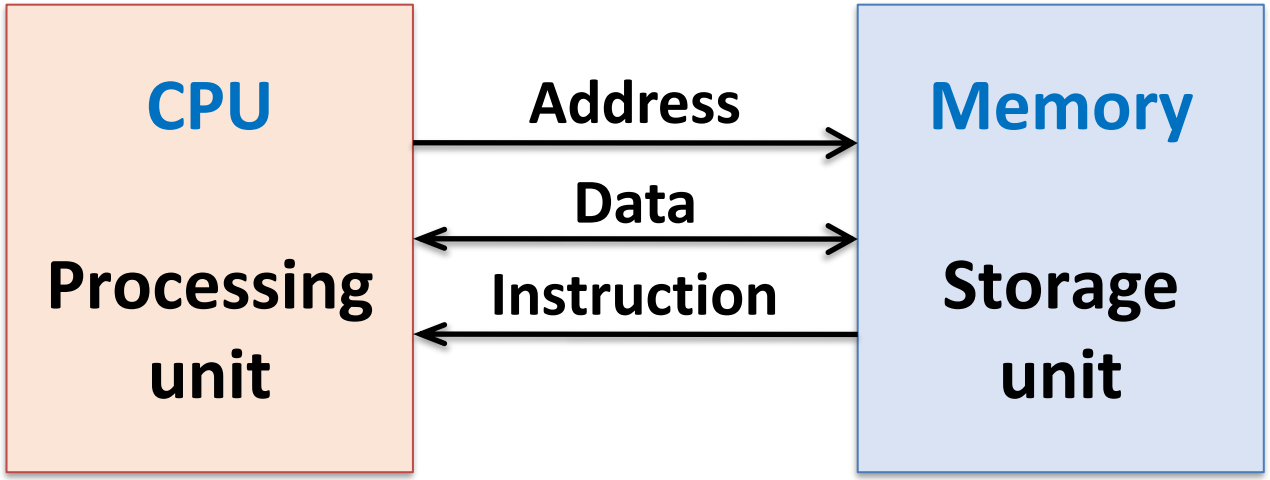

Classes of Computers

- Personal computers
  - General purpose, variety of software
  - Subject to cost/performance tradeoff
- Supercomputers
  - High-end scientific and engineering calculations
  - Highest capability but represent a small fraction of the overall computer market
- Server computers
  - Network-based
  - Range from small servers to large data centers
  - High performance, capacity, reliability
- Embedded computers
  - Hidden as components of systems
  - Stringent power/performance/cost constraints

A computer is a  machine.

von Neumann Architecture

- By John von Neumann, 1945

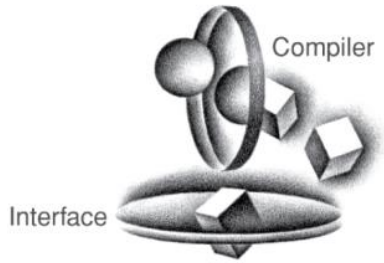


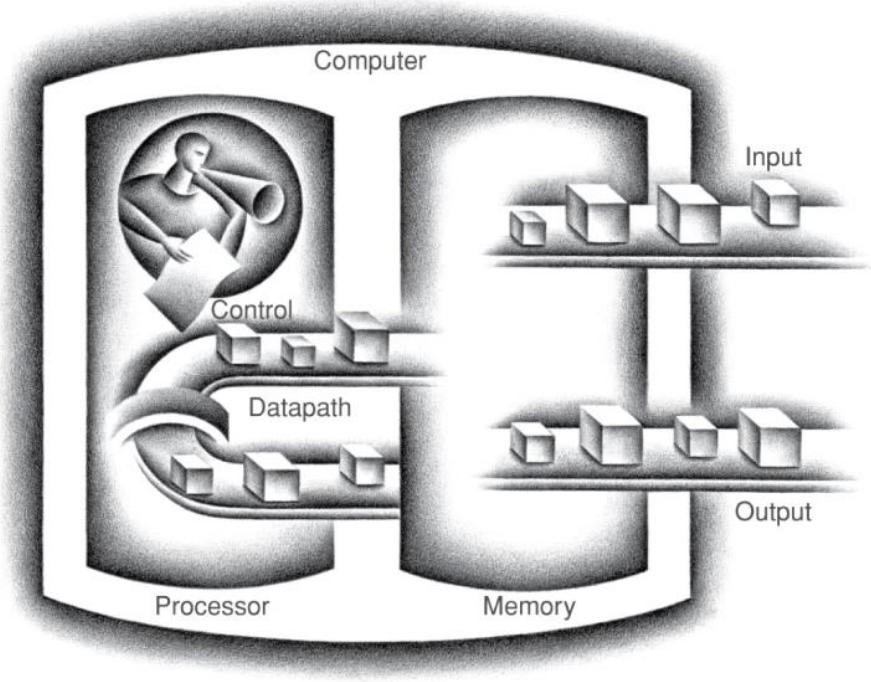
Data movement
Arithmetic & logical ops
Control transfer

Byte addressable array
Code + data (user program, OS)
Stack to support procedures

Components of a Computer

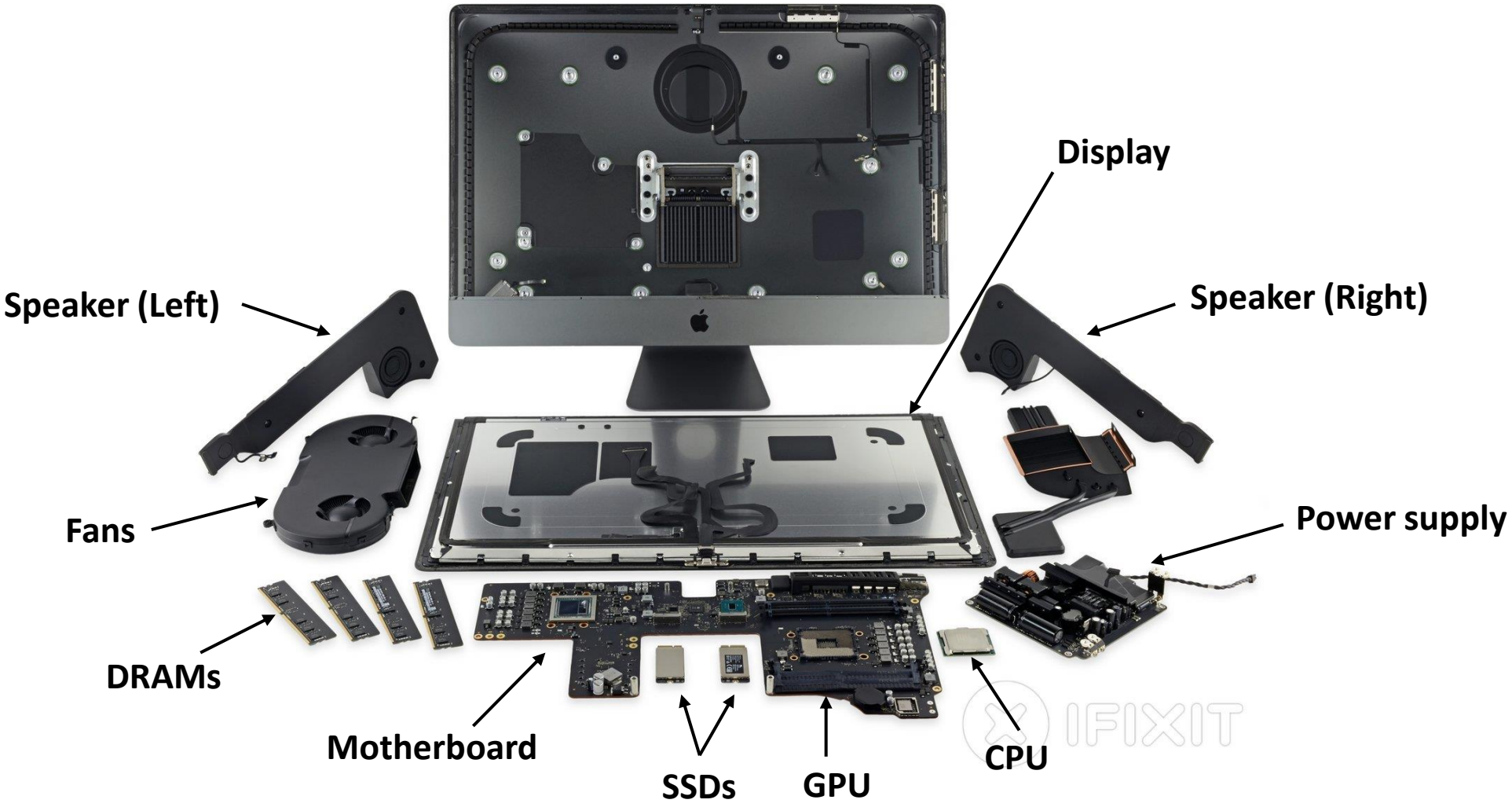
- CPU: Control + Datapath
- Memory
- I/Os
  - GPUs
  - User-interface devices:
    Display, keyboard, mouse, sound, …
  - Storage devices:
    HDD, SSD, CD/DVD, …
  - Network adapters:
    Ethernet, 3G/4G/5G, WiFi, Bluetooth, …





- Same components for all kinds of computer

Opening the Box (iMac Pro)



Source: https://www.ifixit.com/Teardown/iMac+Pro+Teardown/101807

What Happened:

	1997	2017
104 cabinets (76 computes, 8 switches, 20 disks)		
9298 cores	ASCI Red at Sandia	Cavium ThunderX2
150m²	1.3 TF/s, 850 KW	~ 1.1 TF/s, ~ 0.2 KW

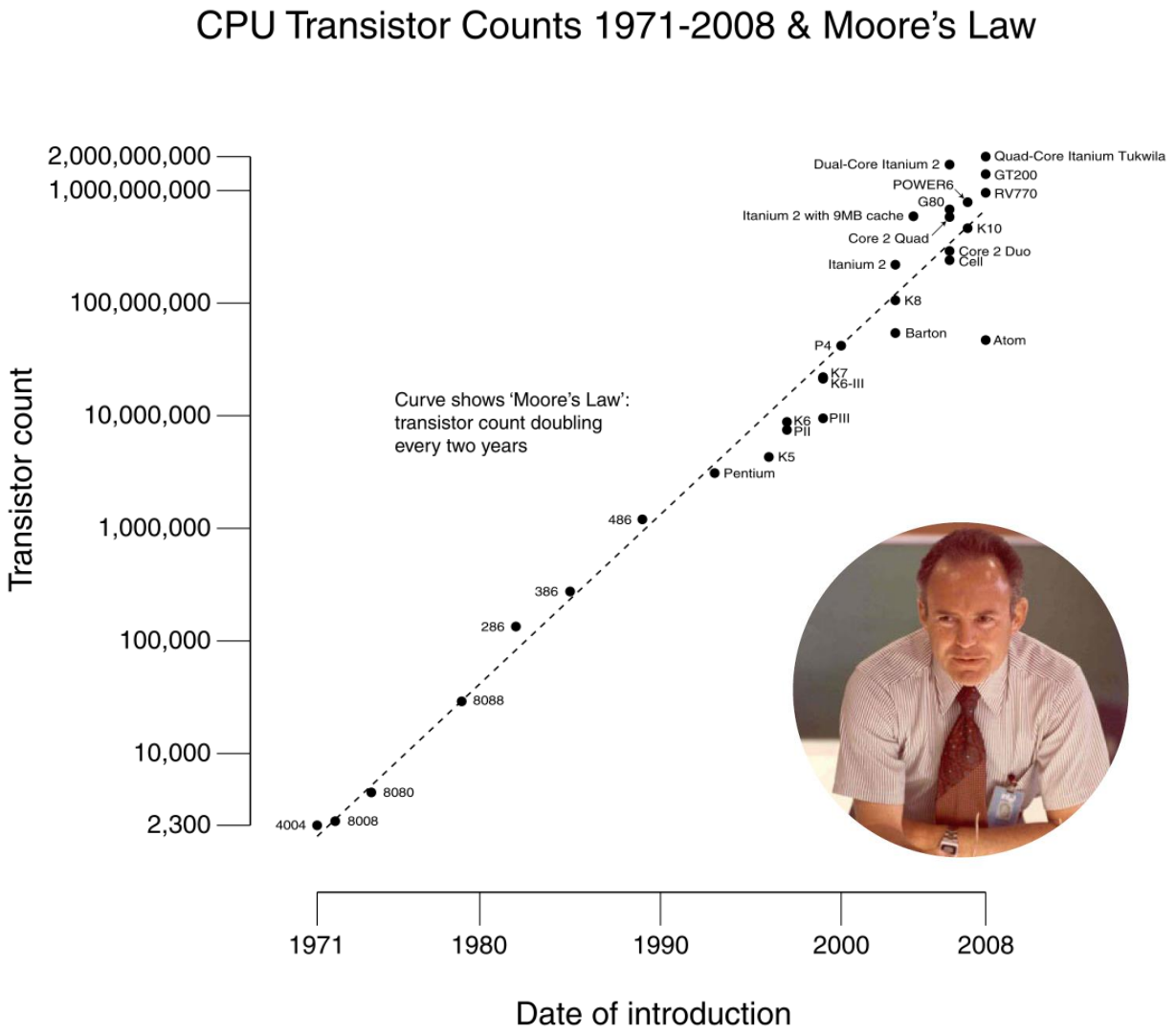
3.5 orders of magnitude

Source: David Keyes, "Algorithmic Adaptations to Extreme Scale Computing," ATPESC, 2018.

The Moore's Law

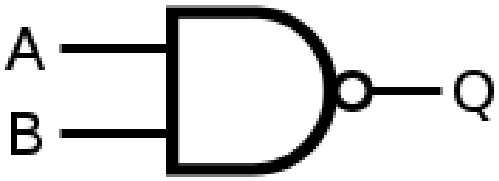
- *"The number of transistors incorporated in a chip will approximately double every 24 months"*
  (Gordon Moore, Intel Co-founder, 1965)

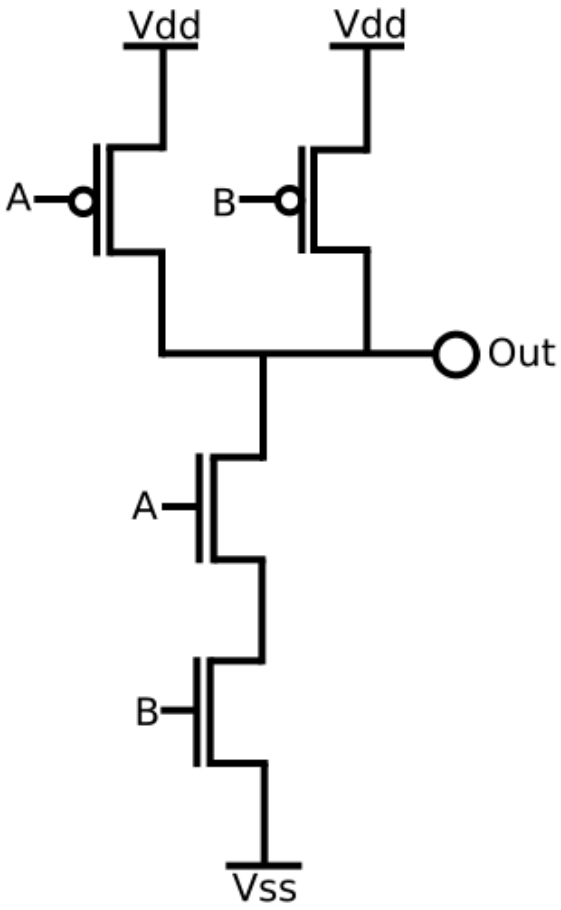
- Makes novel applications feasible
  - WWW, search engines
  - Smartphones, VR/AR
  - AI, Self-driving cars
  - Human genome project, …
- Computers are pervasive

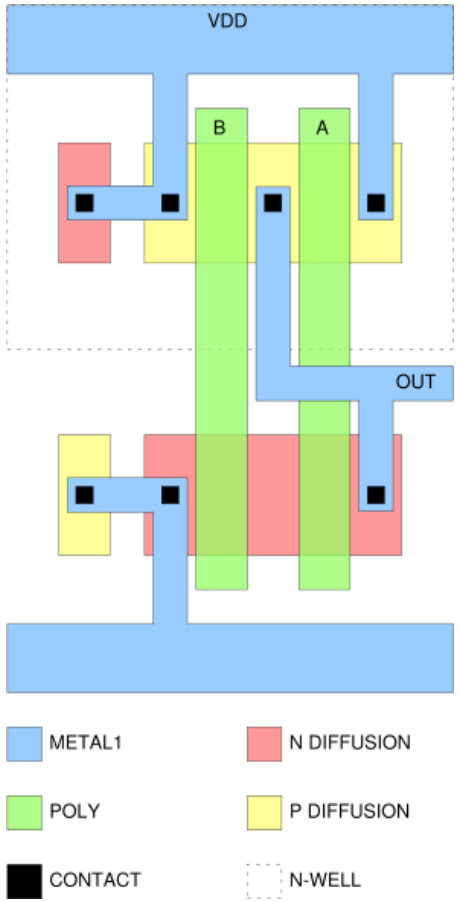


Transistors and Logic Gates

- NAND logic built with CMOS technology

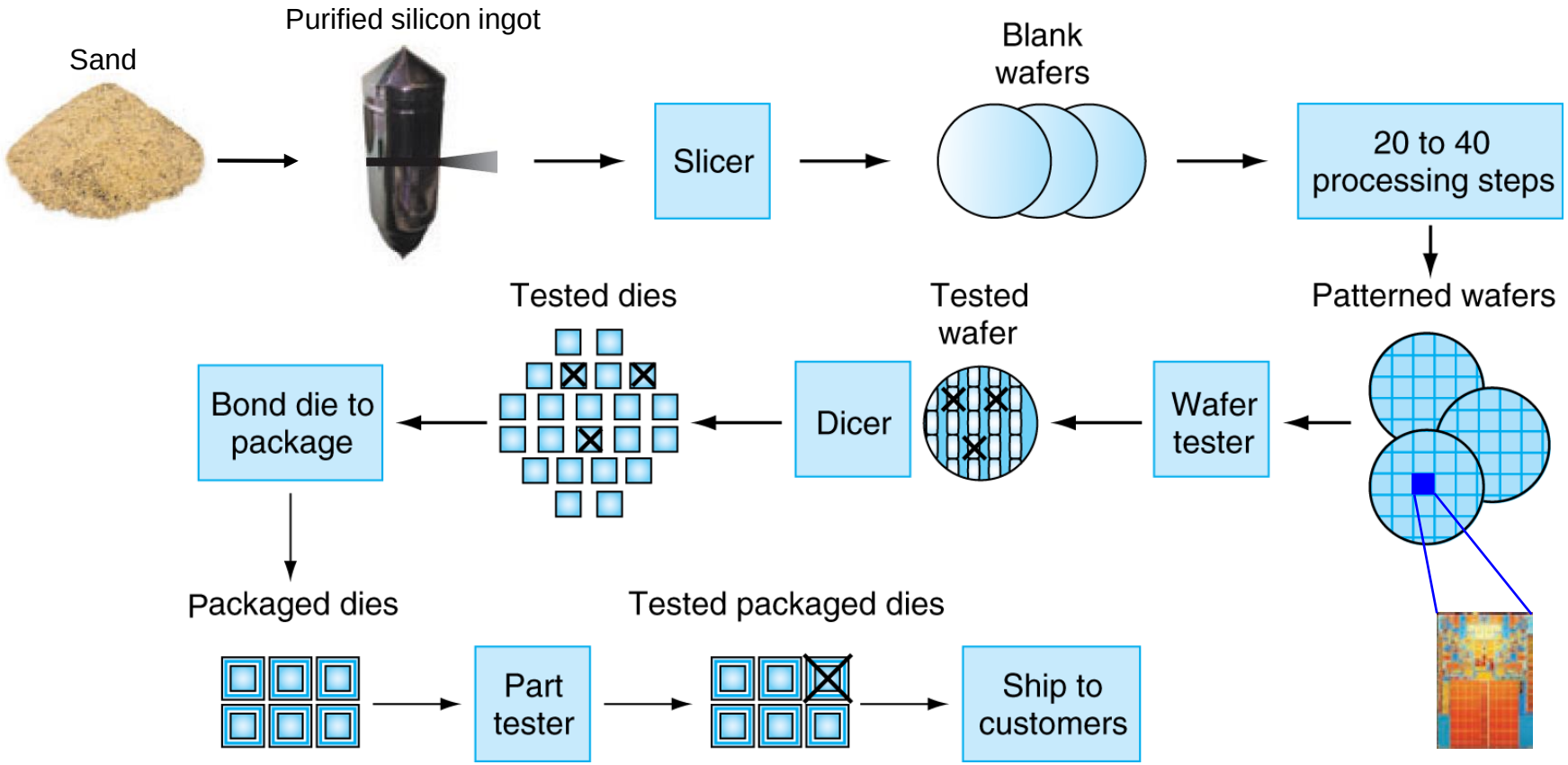






Source: https://en.wikipedia.org/wiki/NAND_gate

From Sand to Circuits



Yield: proportion of working dies per wafer

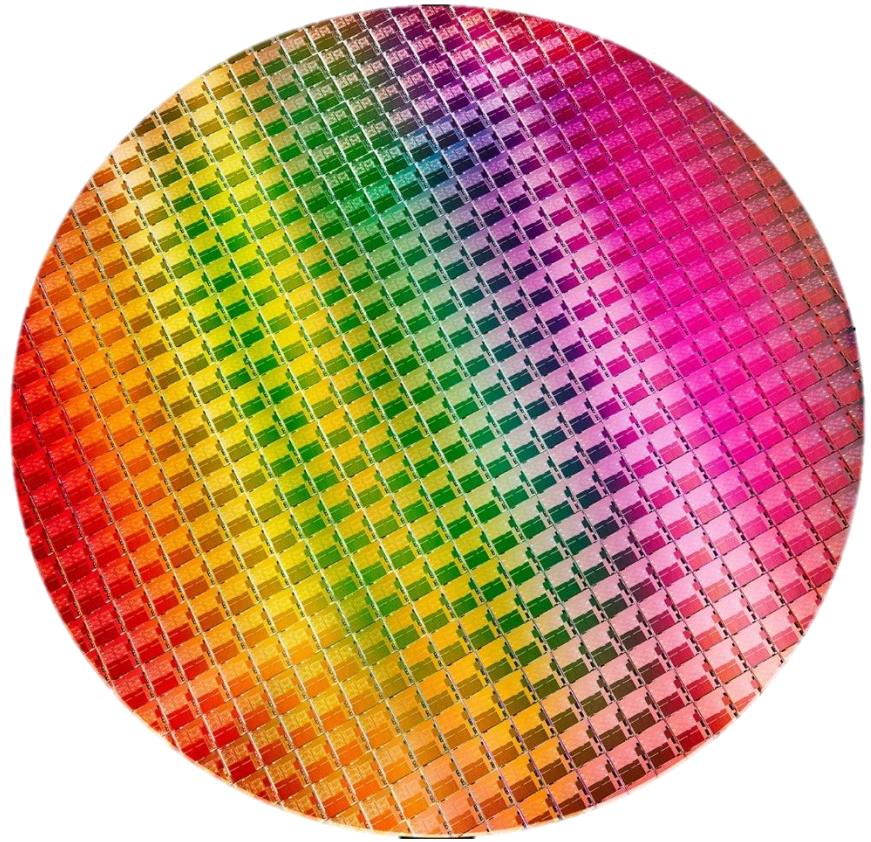
Intel Core 10th Gen. (Ice Lake)

- 12-inch (300mm) wafer, 506 chips, 10nm technology
- Each chip is 11.4 x 10.7 mm

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

$$Dies\ per\ wafer \approx Wafer\ area / Die\ area$$

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



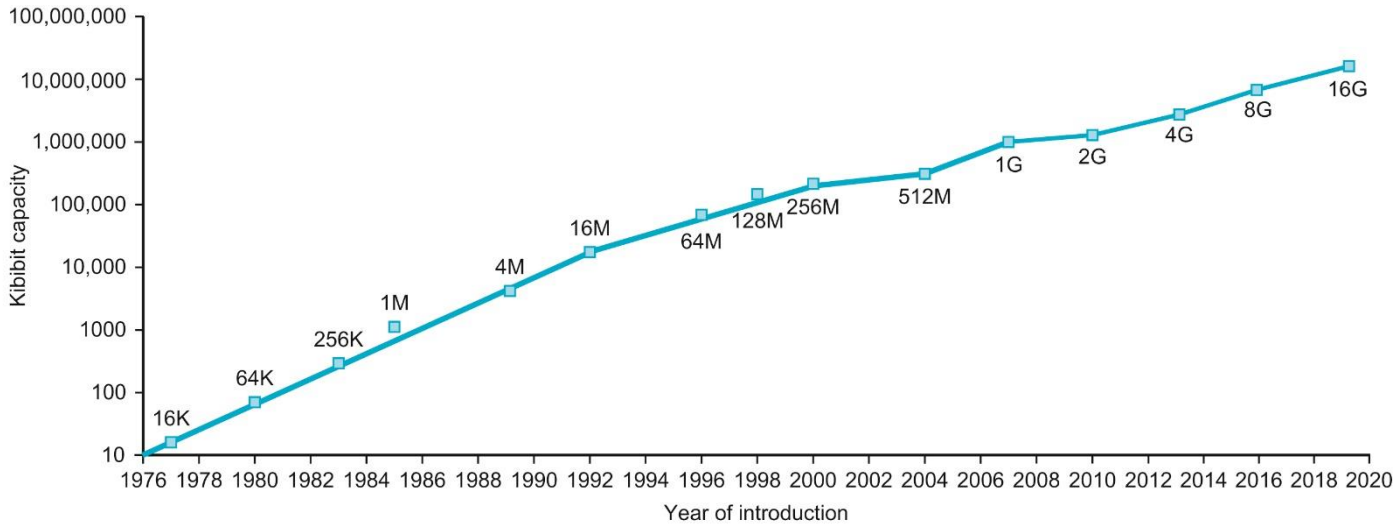
- Wafer cost and area are fixed
- Defect rate determined by manufacturing process
- Die area determined by architecture and circuit design

Technology Trends

- CPU
  - Logic capacity: ~ 30% / year
  - Clock rate: ~ 20% / year

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

- Memory
  - DRAM capacity: ~ 60% / year (4x every 3 years)
  - DRAM speed: ~ 10% / year



Dennard Scaling (RIP ~2005)

- In CMOS IC technology,

$$\boldsymbol{Power = Capactive\ Load \times Voltage^2 \times Frequency}$$

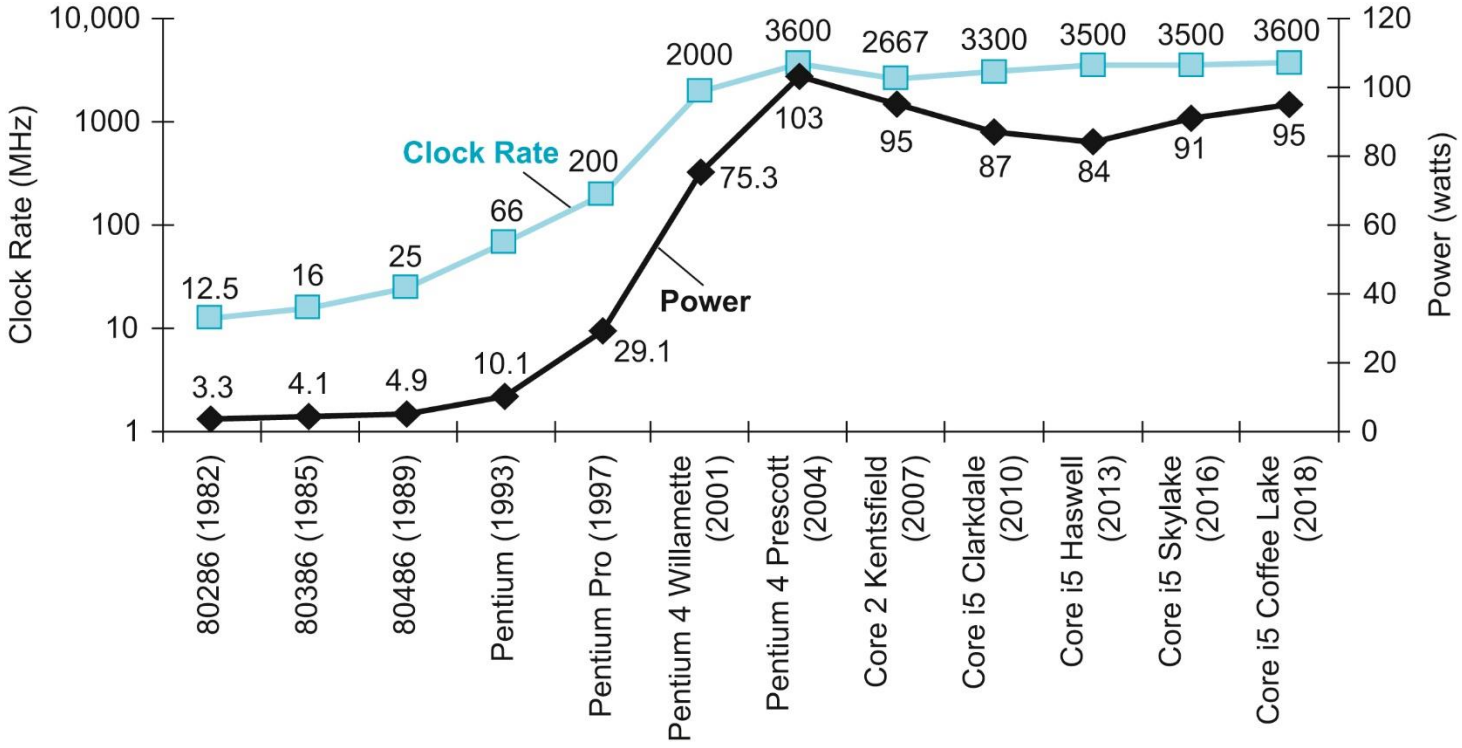
- As transistors get smaller, their power density stays constant
  - Transistor dimensions scaled by 0.7x
  - Capacitance reduced by 0.7x
  - Frequency increased by 1.4x (due to reduced delay)
  - Voltage reduced by 0.7x

$$\frac{\boldsymbol{P_{new}}}{\boldsymbol{P_{old}}} = \frac{\boldsymbol{C_{old} \times 0.7 \times (V_{old} \times 0.7)^2 \times F_{old} \times 1.4}}{\boldsymbol{C_{old} \times V_{old}^2 \times F_{old}}} = \boldsymbol{0.48}$$

Doubled transistors with faster performance at constant power!

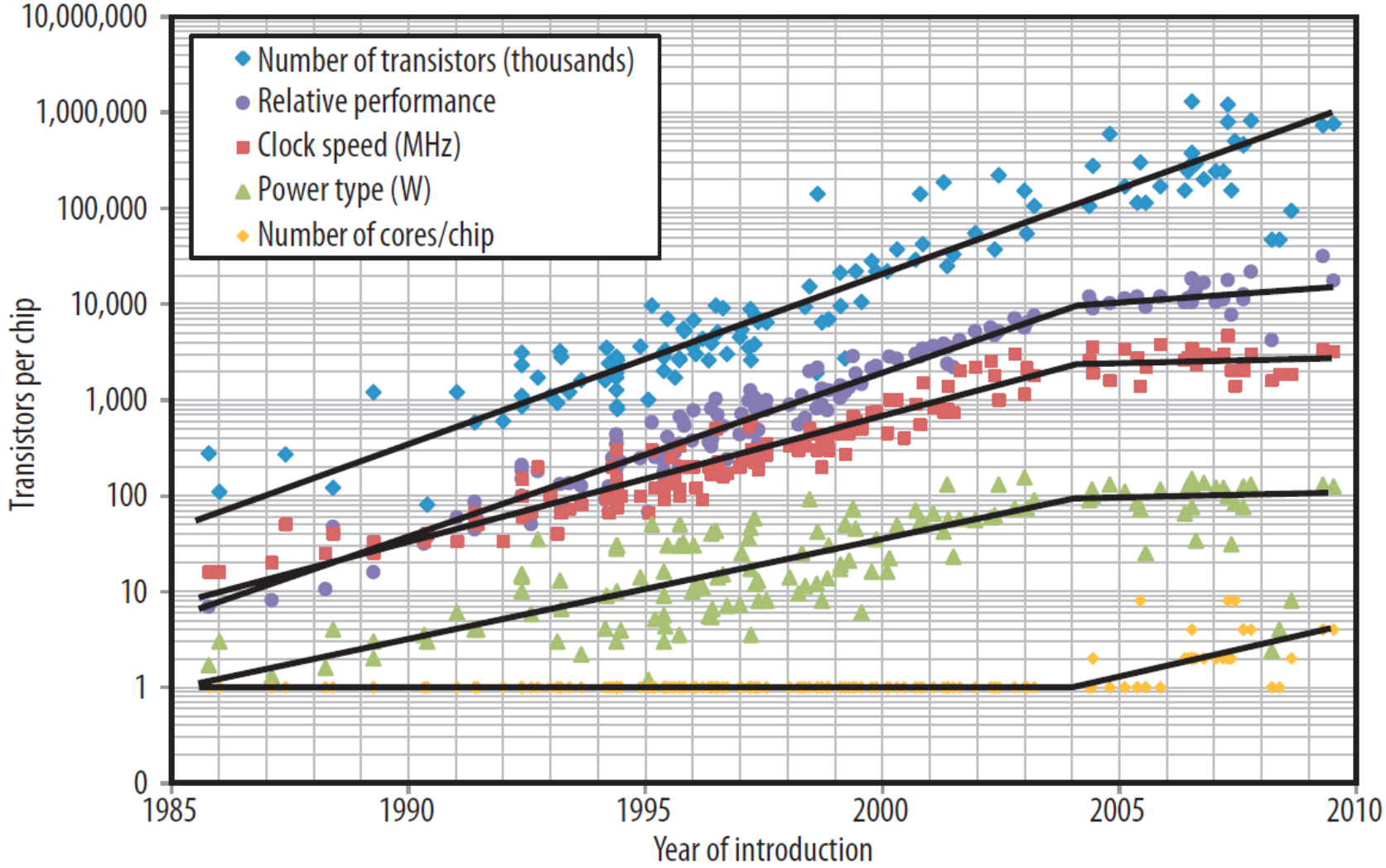
Power Trends

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



The Shift to Multicores

- ILP wall
  - Control dependency
  - Data dependency
- Memory wall
  - Memory latency improved by 10% / year
  - Cache shows diminishing returns
- Power wall

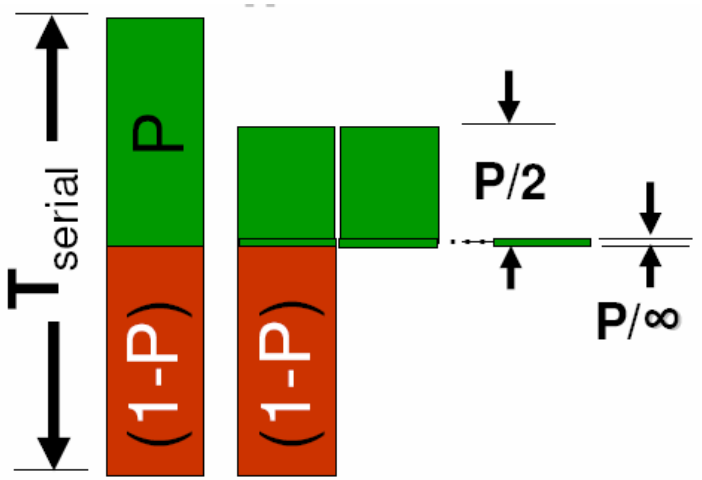


Source: IEEE Computer, 2011

Amdahl's Law

- The theoretical upper limit of speed up is limited by the serial portion of the code

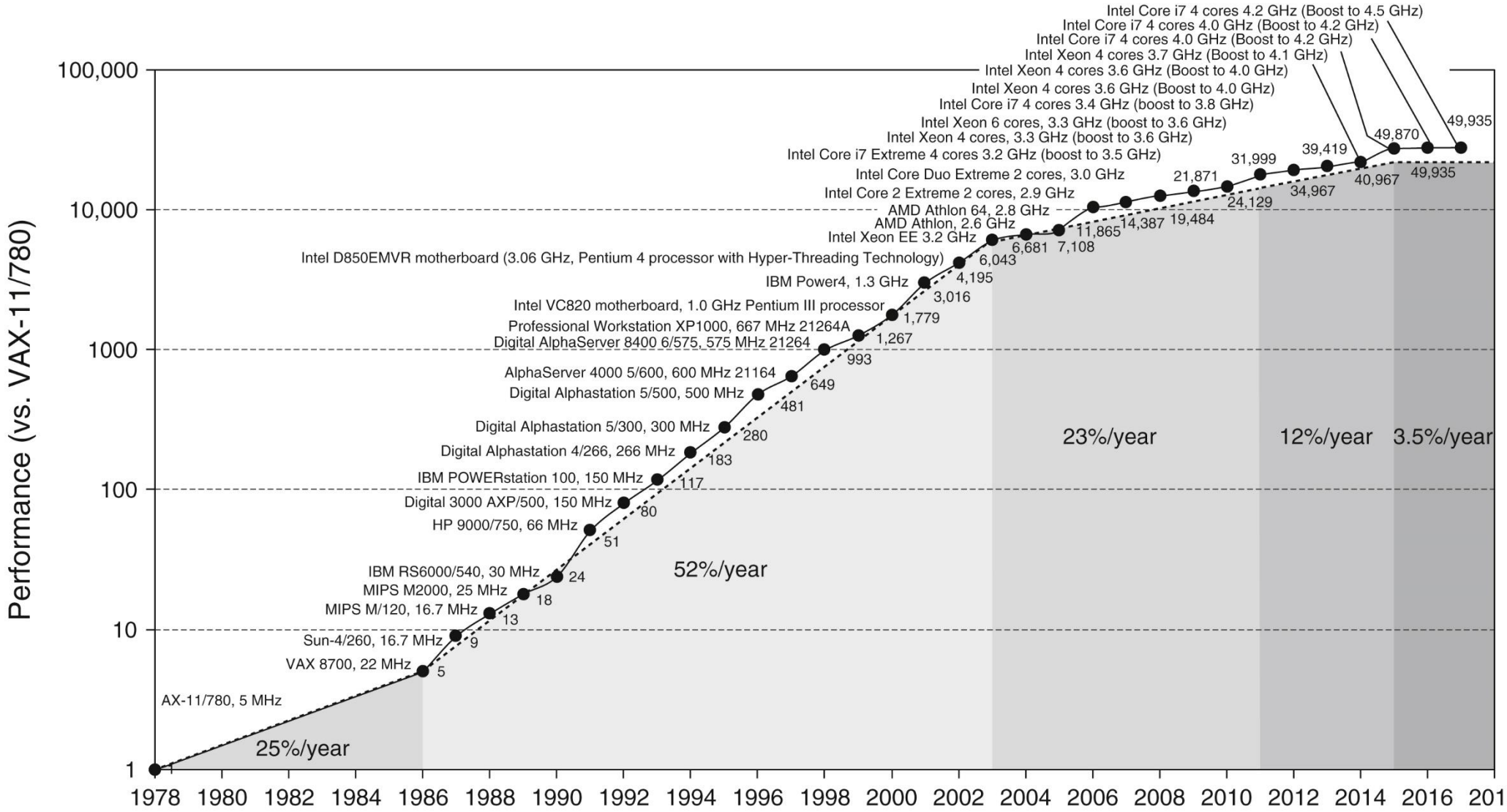
$$Speedup = \frac{1}{(1-P) + \frac{P}{n}}$$



  - $S = (1 - P)$: the time spent executing the serial portion
  - n: the number of processor cores

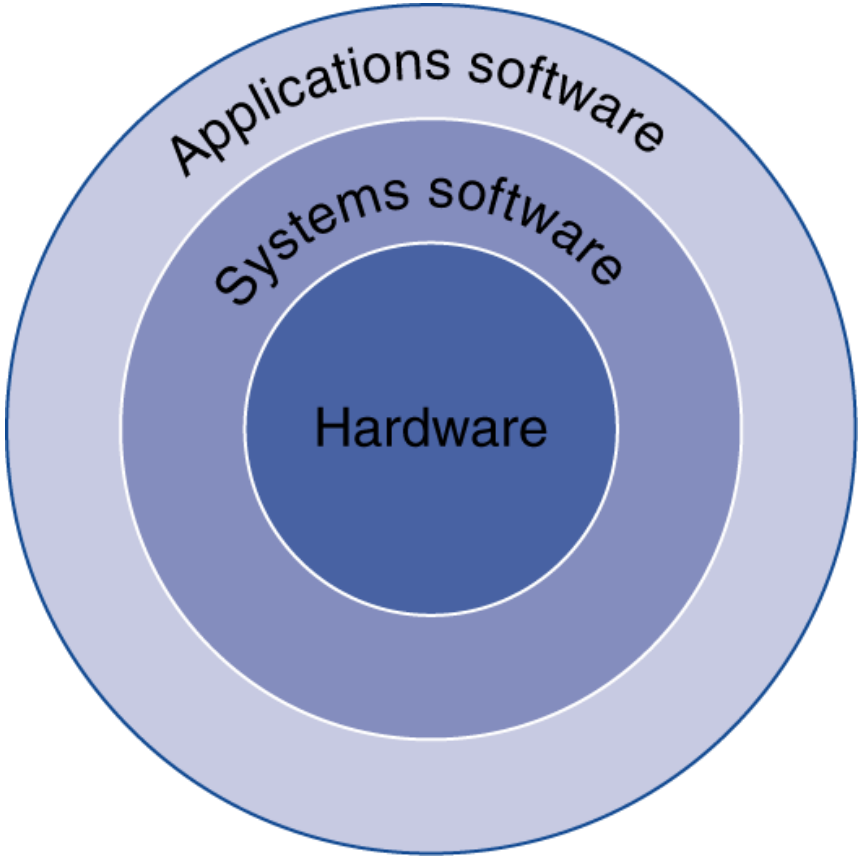
- Corollary: make the common case fast

Uniprocessor Performance

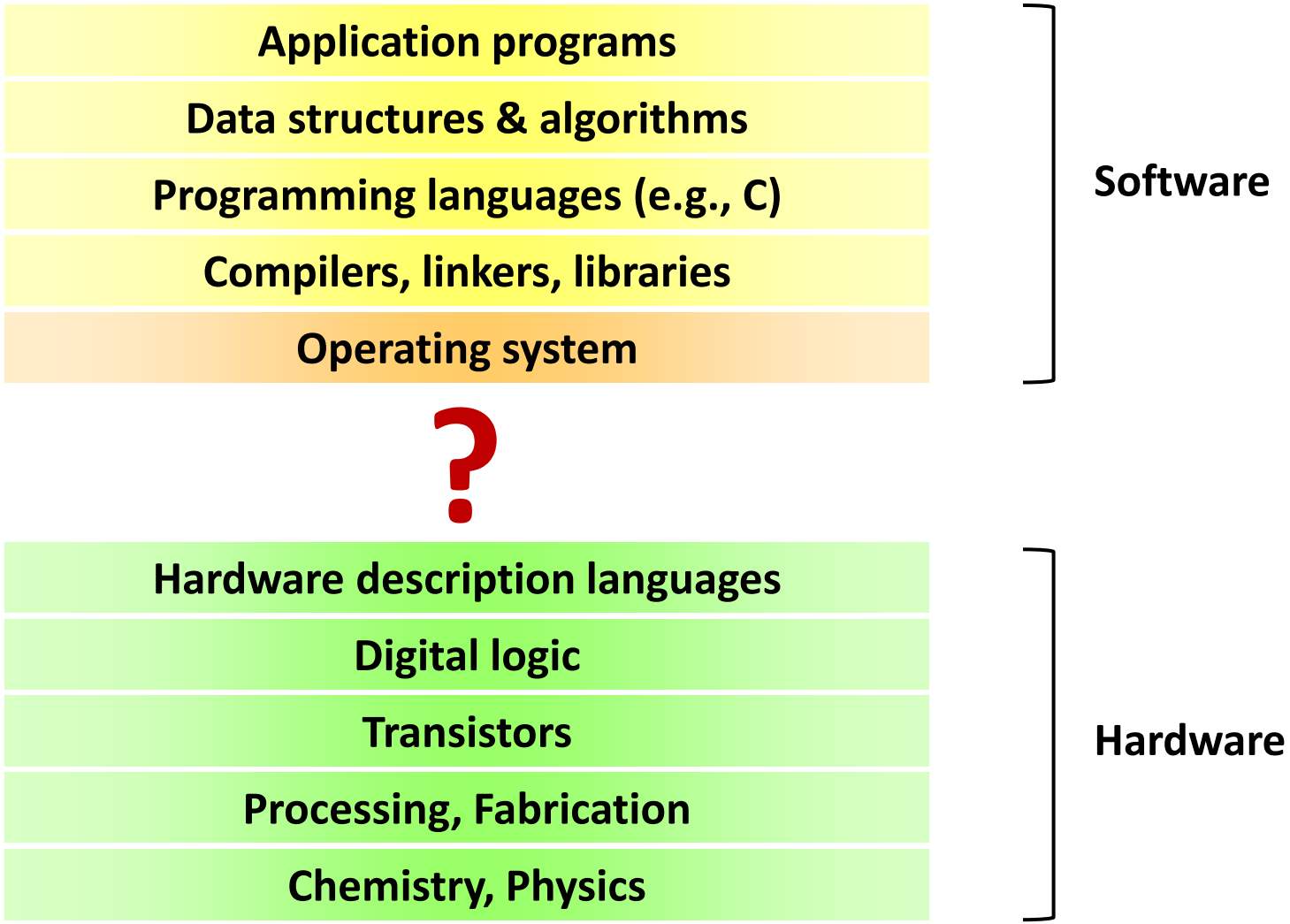


Software: Below Your Program

- 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



How to Run Your Program?



A Case for Robot Vacuum Cleaner

- Commands (or instructions)
  - Go one step forward `00`
  - Go one step backward `01`
  - Turn left `10`
  - Turn right `11`

- A simple program

```
00 00 11 00 11
00 00 10 00 10
00 00 11 00 11
00 00 10 00 10  ...
```

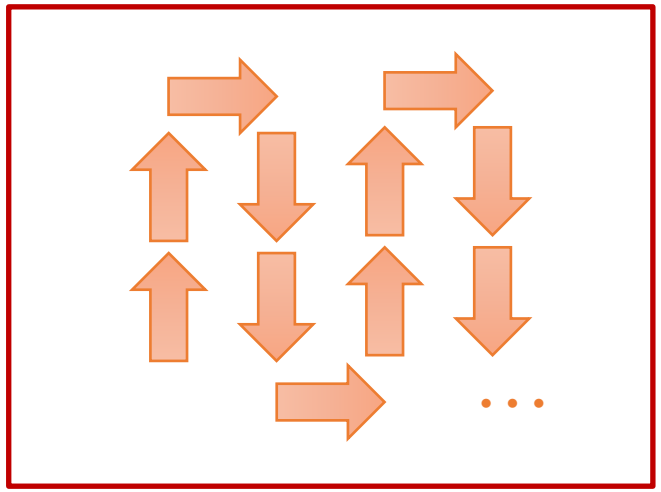
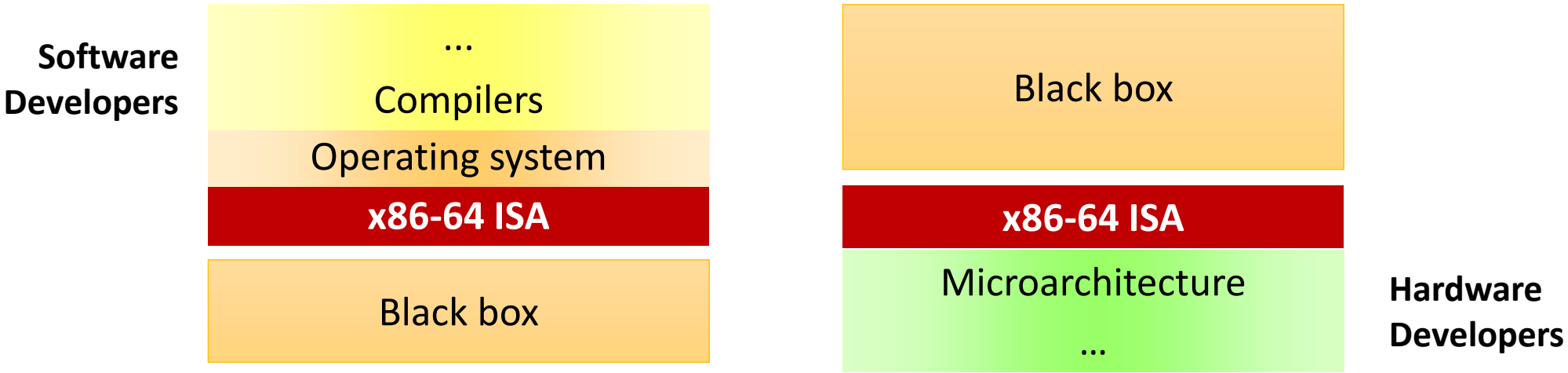


Image from https://www.lge.co.kr/vacuum-cleaners

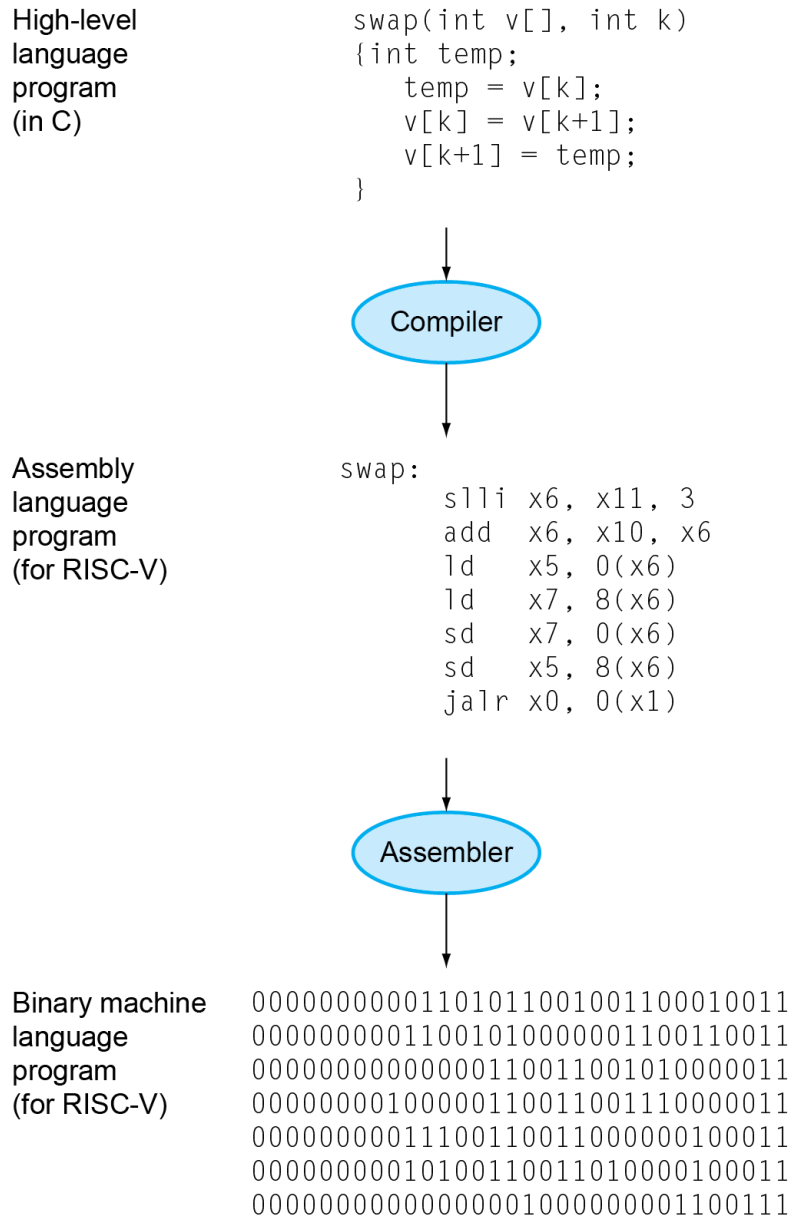
Instruction Set Architecture (ISA)

- The hardware/software interface
  - Hardware abstraction visible to software (OS, compilers, …)
  - Instructions and their encodings, registers, data types, addressing modes, etc.
  - Written documents about how the CPU behaves
  - e.g., All 64-bit Intel CPUs follow the same x86-64 (or Intel 64) ISA

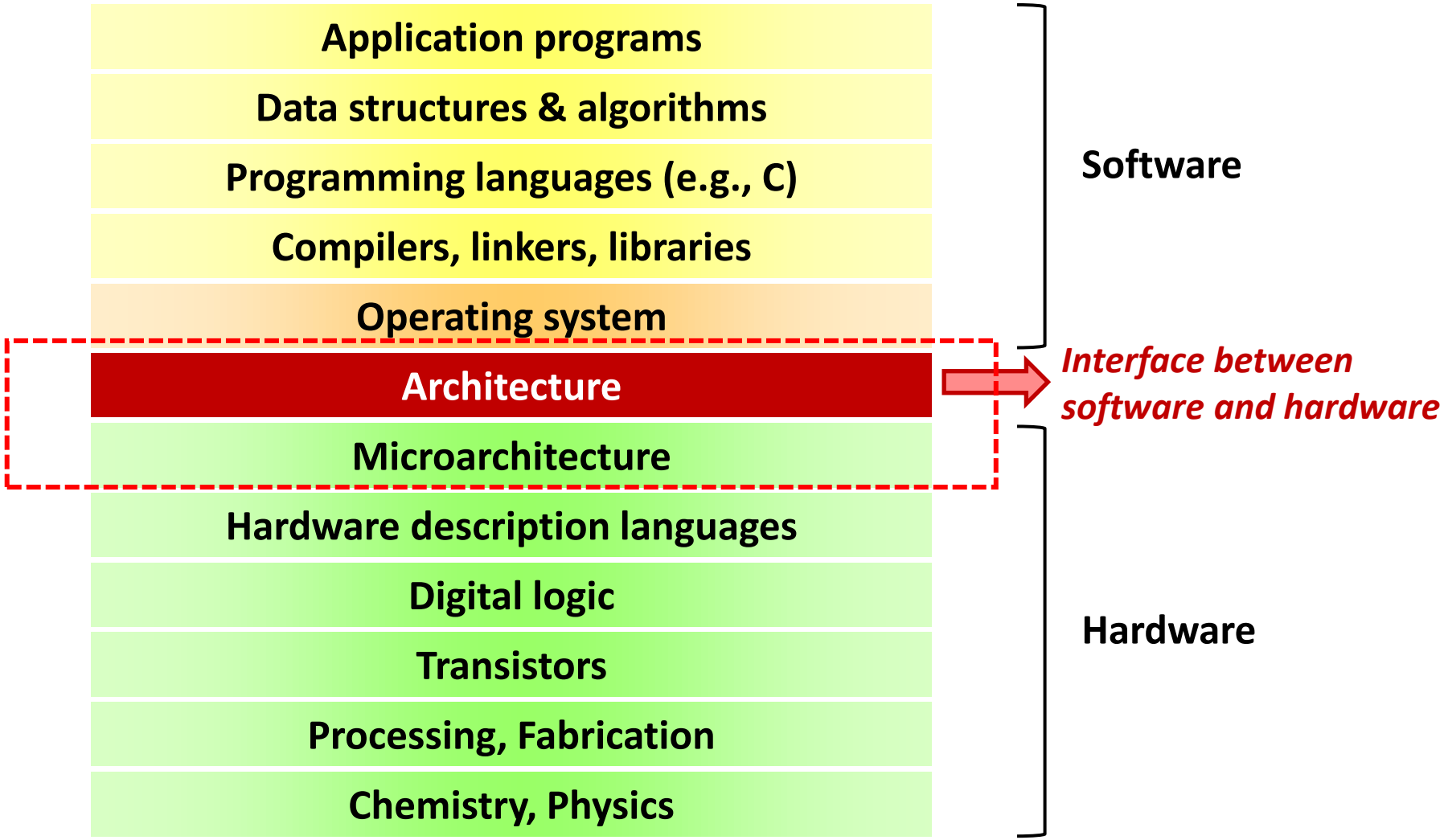


Levels of Program Code

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

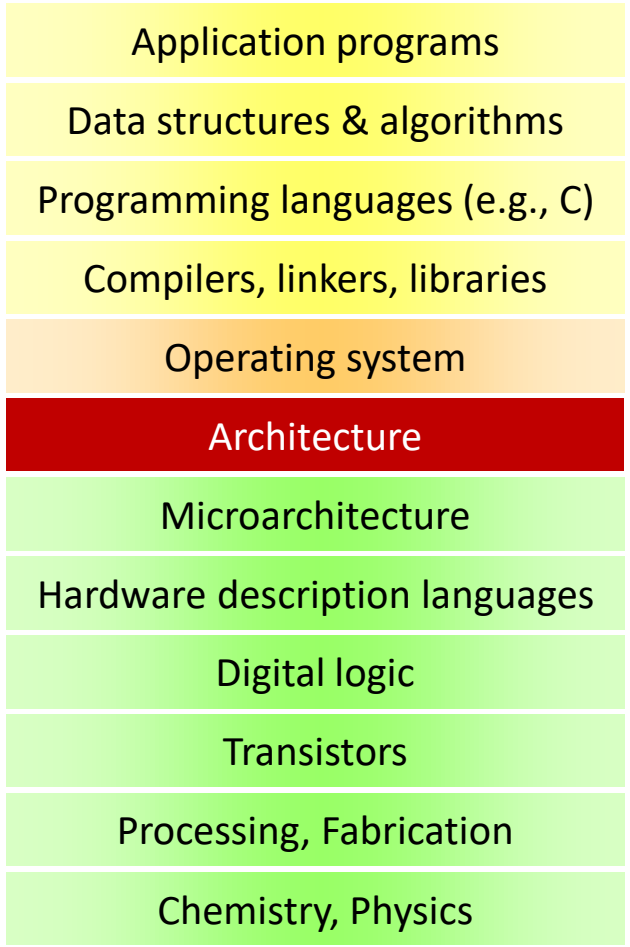


Full Levels of Abstraction



Abstraction is Good, But …

- Abstraction helps us deal with complexity
  - Hide lower-level details
- These abstractions have limits
  - Especially in the presence of bugs
  - Need to understand details of underlying implementations
- What is the right place to solve the problem?
- This is why you should take this course seriously even if you don't want to be a computer architect!



Topic 1: How To Design Interface?

- Choices critically affect both the software programmer and hardware designer
- Example: Copying *n* bytes from address *A* to *B*

x86_64 (CISC)

```
movq    A, %rsi
movq    B, %rdi
movq    n, %rcx
REP MOVS
```

RISC-V (RISC)

```
la      a0, A
la      a1, B
li      a2, n
add     a3, a0, a2
```

```
L0:
  lbu     a4, 0(a0)
  sbu     a4, 0(a1)
  addi    a0, a0, 1
  addi    a1, a1, 1
  bne     a0, a3, L0
```

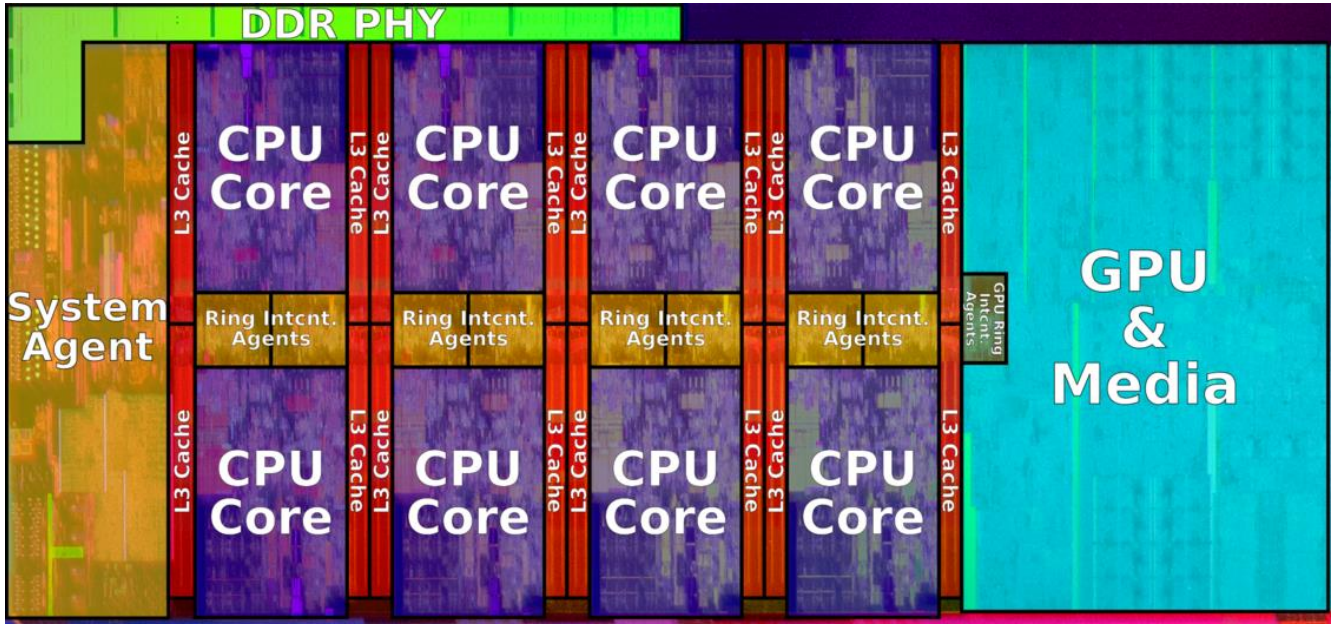
- Trade-offs: code size, compiler complexity, operating frequency, number of cycles to execute, hardware complexity, energy consumption, etc.

Topic 2: How To Implement?

- Microarchitectures: Where should you spend transistors to run your program faster with conforming to the given interface?

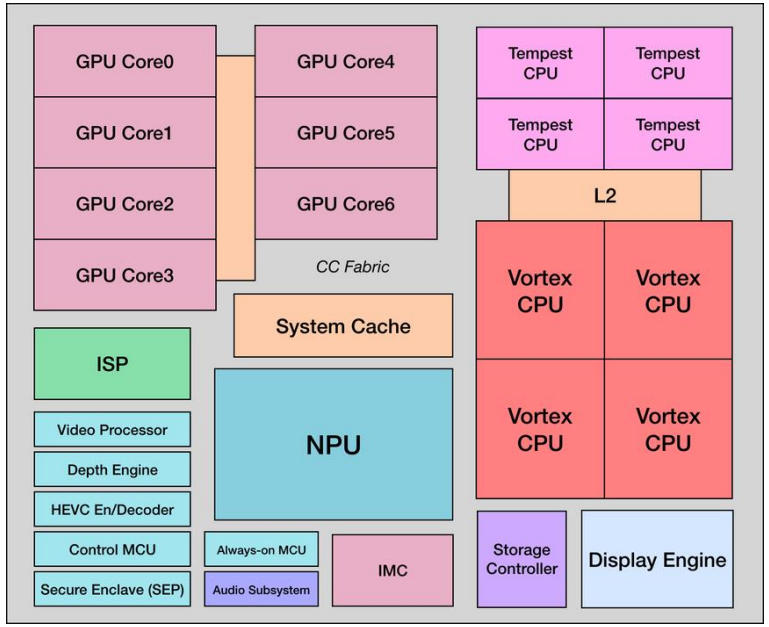
Intel Core i9-9900K (Coffee Lake, 2018)

Transistors: ~ 3B (14nm), Die size: ~ 177mm^2



Apple A12X Bionic (2018)

Transistors: ~ 10B (7nm), Die size: ~ 122mm^2



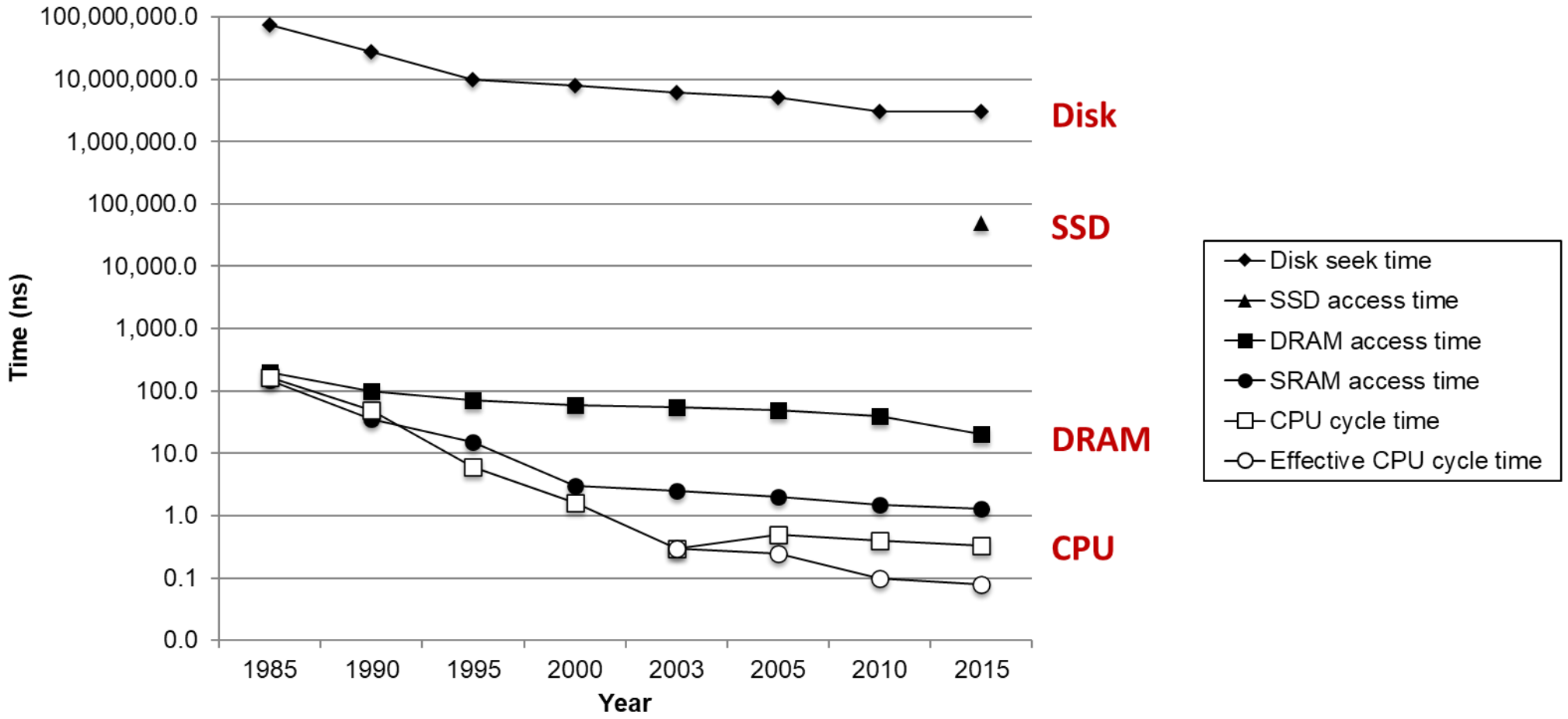
Source: https://en.wikichip.org/wiki/intel/core_i9/i9-9900k, https://en.wikichip.org/wiki/apple/ax/a12x

Exploiting Parallelism

- Instruction level parallelism (ILP)
  - Pipelining
  - Superscalar
  - Out-of-order execution
  - Branch prediction
  - VLIW (Very Long Instruction Word)
- Data level parallelism (DLP)
  - SIMD / Vector instructions
- Task level parallelism (TLP)
  - Simultaneous multithreading (Hyperthreading)
  - Multicore

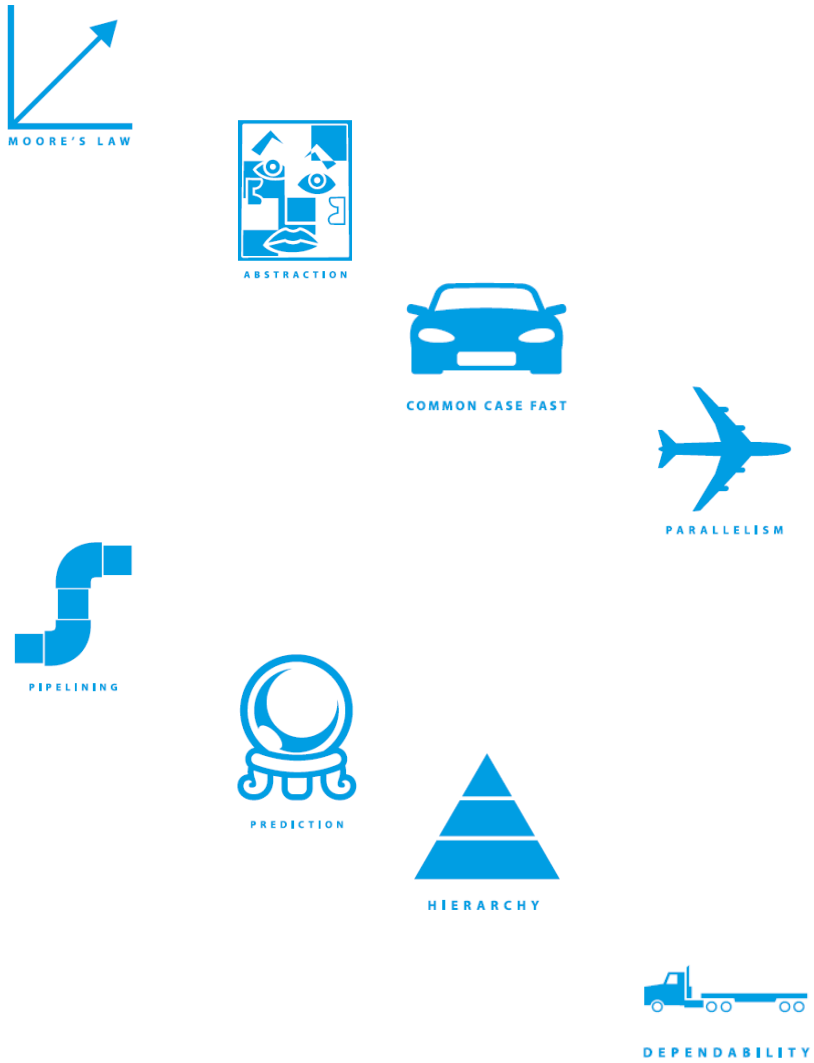
Topic 3: What About the Memory?

- It's just too slow!

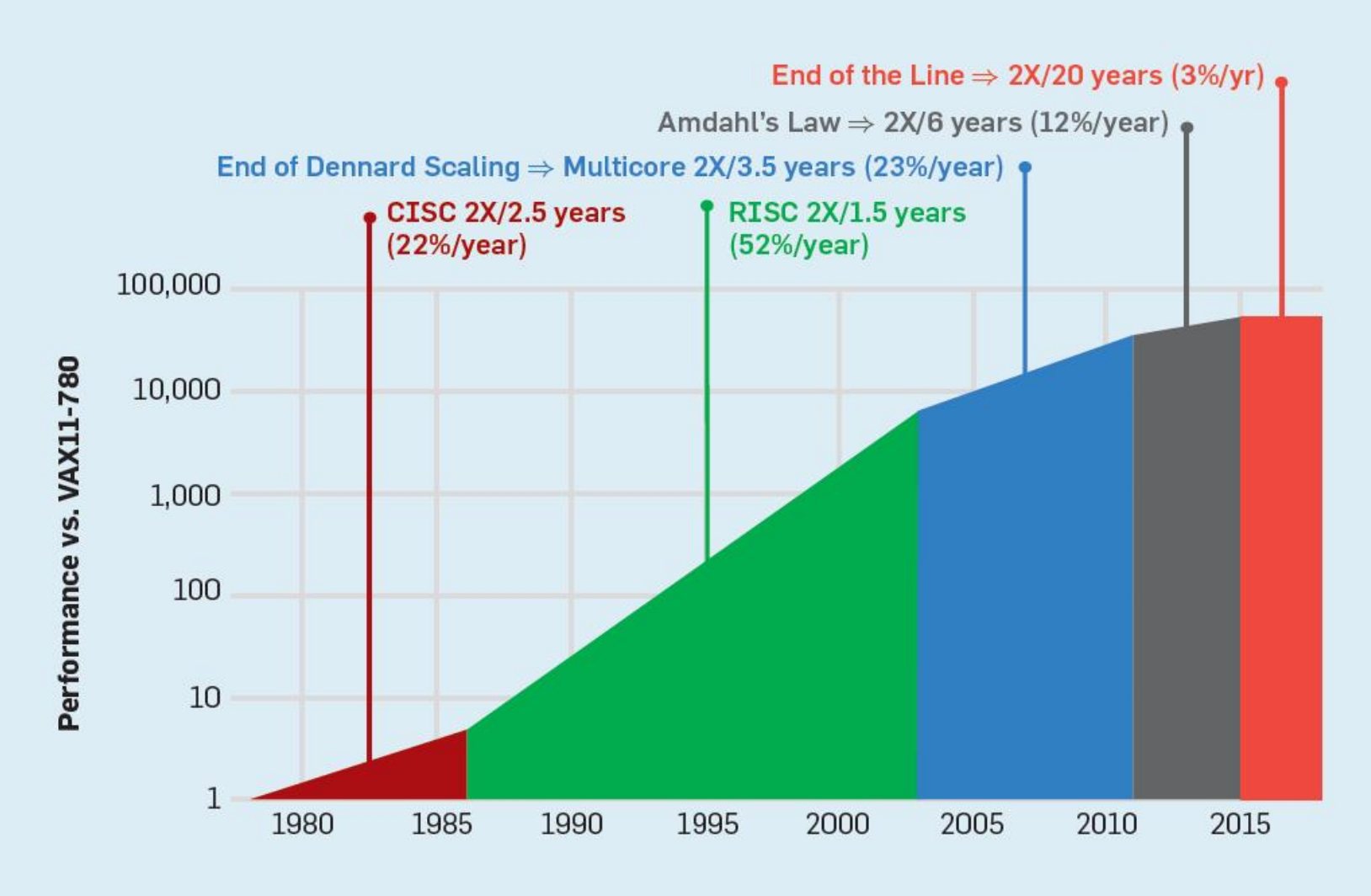


Eight Great Ideas in Computer Architecture

- 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



Summary: A New Golden Age?



End of Moore's law

A New Golden Age for Computer Architecture (CACM, Feb. 2019)

Source: J. L. Hennessy and D. A. Patterson, "A New Golden Age for Computer Architecture", CACM, 2019.