CSC 411

Computer Organization (Fall 2024)
Lecture 9: Computer systems

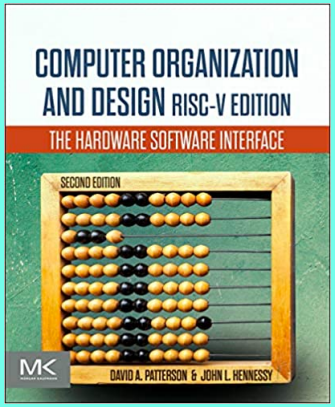
Prof. Marco Alvarez, University of Rhode Island

Disclaimer

Some figures and slides are adapted from:

Computer Organization and Design (Patterson and Hennessy)

The Hardware/Software Interface



Units of measure

SI prefixes

Factor	Name	Symbol	Factor	Name	Symbol
10^{1}	deca	da	10^{-1}	deci	d
10^{2}	hecto	h	10^{-2}	centi	c
10^{3}	kilo	k	10^{-3}	milli	m
10^{6}	mega	M	10^{-6}	micro	μ
10^{9}	giga	G	10^{-9}	nano	n
10^{12}	tera	T	10^{-12}	pico	p
10^{15}	peta	P	10^{-15}	femto	f
10^{18}	exa	E	10^{-18}	atto	a
10^{21}	zetta	Z	10^{-21}	zepto	z
10^{24}	yotta	Y	10^{-24}	yocto	y
10^{27}	ronna	R	10^{-27}	ronto	r
10^{30}	quetta	Q	10^{-30}	quecto	q

SI prefixes are a set of 24 prefixes used in the International System of Units (SI) to indicate multiples and submultiples of SI units. They are based on powers of 10, and each prefix has a unique symbol.

Binary prefixes

The SI prefixes refer strictly to powers of 10. They should not be used to indicate powers of 2 (for example, one kilobit represents 1000 bits and not 1024 bits). The names and symbols for prefixes to be used with powers of 2 are recommended as follows:

kibi	Ki	2^{10}
mebi	Mi	2^{20}
gibi	Gi	2^{30}
tebi	Ti	2^{40}
pebi	Pi	2^{50}
exbi	Ei	2^{60}
zebi	Zi	2^{70}
yobi	Yi	2^{80}

Decimal term	Abbreviation	Value	Binary term	Abbreviation	Value	% Larger
kilobyte	KB	10^3	kibibyte	KiB	2^{10}	2%
megabyte	MB	10^6	mebibyte	MiB	2^{20}	5%
gigabyte	GB	10^9	gibibyte	GiB	2^{30}	7%
terabyte	TB	10^{12}	tebibyte	TiB	2^{40}	10%
petabyte	PB	10^{15}	pebibyte	PiB	2^{50}	13%
exabyte	EB	10^{18}	exbibyte	EiB	2^{60}	15%
zettabyte	ZB	10^{21}	zebibyte	ZiB	2^{70}	18%
yottabyte	YB	10^{24}	yobibyte	YiB	2^{80}	21%
ronnabyte	RB	10^{27}	robibyte	RiB	2^{90}	24%
queccabyte	QB	10^{30}	quebibyte	QiB	2^{100}	27%

Practice

- A hard drive is advertised as 1 TB (terabyte)
  - calculate the difference in storage capacity (in bytes) between the SI and binary interpretations of this size
- A file size is given as 2.5 GiB (gibibytes)
  - express this in both gigabytes (GB) and mebibytes (MiB)

Computer systems

The computer revolution

- Progress in computer technology
  - underpinned by domain-specific accelerators
- Novel applications (not long ago considered fiction):
  - generative AI, automotive computers, smartphones, human genome project, web, search engines
- Although common hardware technologies …
  - different design requirements

```
Computers in all forms are now pervasive
```

Classes of computers

- Personal computers (desktops/laptops)
  - low cost, general purpose, variety of software
  - subject to cost/performance tradeoff
- Server computers
  - network based, large workloads
  - high capacity, performance, reliability
  - range from small servers to building sized supercomputers (high-end scientific and engineering calculations)
- Embedded computers
  - largest class by volume
  - hidden as components of systems (Internet of Things)
  - power/performance/cost constraints

The post PC era

- Personal Mobile Device (PMD) — overtaking PCs
  - battery operated, internet-connected
  - smart phones, tablets, electronic glasses
- Cloud computing — overtaking traditional servers
  - Warehouse Scale Computers (WSC) — e.g. Amazon AWS
  - Software as a Service (SaaS)
    - portion of software run on a PMD and a portion run in the Cloud

Accelerators and heterogeneous computing

- Multiple cores in one chip
- System-on-a-Chip (SoC) design
- Graphics processing units (GPUs)
  - throughput-oriented multicore processors
  - great for gaming and machine learning

```
Focus on performance and energy efficiency

   critical in all classes of computers
```

High performance computing

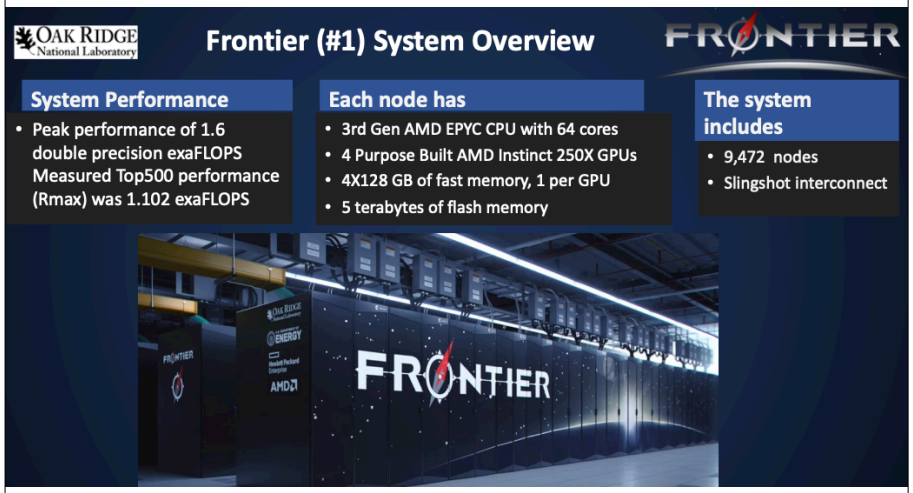
Top 500 project

- 500 most powerful computers in the world
  - exascale applications
- Updated twice a year
  - ISC'xy in June, Germany and SC'xy in November, U.S.

	NOVEMBER 2023	Manufacturer	Computer	Country	Cores	Rmax [Pflops]	Power [MW]
1	Oak Ridge National Laboratory	HPE	Frontier HPE Cray EX235a, AMD EPYC 64C 2.0GHz, Instinct MI250X, Slingshot-11	USA	8,730,112	1,102	21.1
2	Argonne National Laboratory	HPE	Aurora* HPE Cray EX Xeon CPU Max 9470 52C 2.4GHz, Intel Data Center GPU Max, Slingshot-11	USA	4,742,808	585.3	24.6
3	Microsoft Azure	Microsoft	Eagle Microsoft NDv5 Xeon Platinum 8480C 48C 2GHz, NVIDIA H100, NVIDIA Infiniband NDR	USA	1,123,200	561.2	
4	RIKEN Center for Computational Science	Fujitsu	Fugaku Supercomputer Fugaku, A64FX 48C 2.2GHz, Tofu interconnect D	Japan	7,630,848	442.0	29.9
5	EuroHPC / CSC	HPE	LUMI HPE Cray EX235a, AMD EPYC 64C 2.0GHz, Instinct MI250X, Slingshot-11	Finland	2,069,760	309.1	6.0

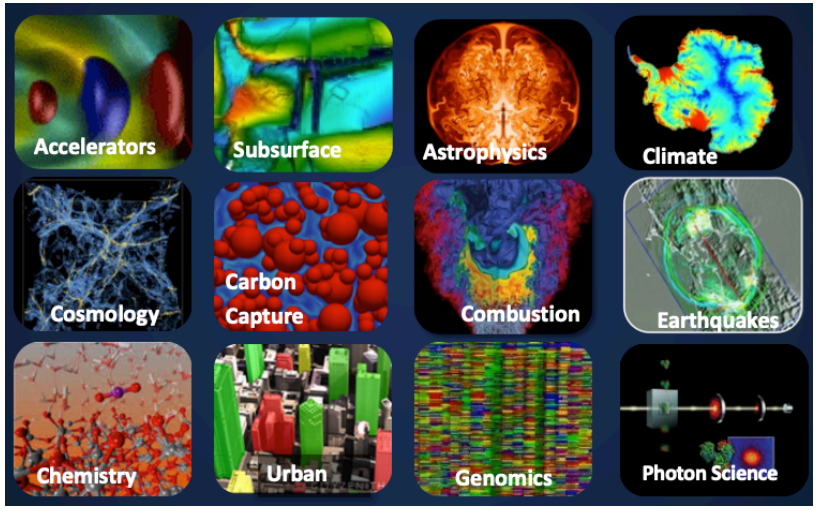
credit: Berkeley's CS267 lectures

Top computer in the world



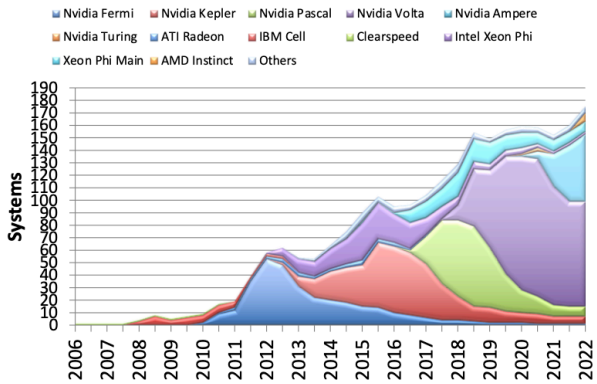
credit: Berkeley's CS267 lectures

Exascale applications at LBNL



credit: Berkeley's CS267 lectures

Heterogenous devices



TOP500

2022: Over 1/3rd of all systems have **accelerators or co-processors**

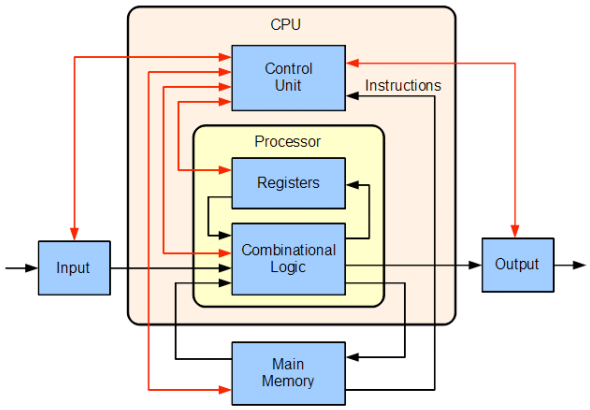
The actual "performance share" is higher

credit: Berkeley's CS267 lectures

Abstractions

The Von-Neumann model

- High-level organization of computer hardware



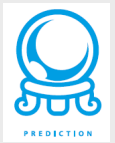
Same model for all kinds of computers: desktop, server, embedded

Input/output includes: display, keyboard, mouse, hard disk, CD/DVD, network adapters, …

https://en.wikipedia.org/wiki/Computer_architecture

Seven great ideas

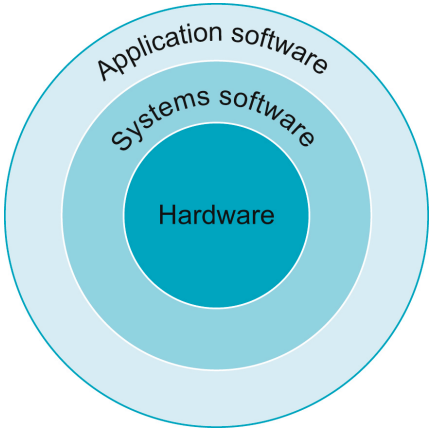
- Use **abstraction** to simplify design
- Make the **common case fast**
- Performance via **parallelism**
- Performance via **pipelining**
- Performance via **prediction**
- **Hierarchy** of memories
- **Dependability** via redundancy



Abstraction layers (computing system)

	Problem
	Algorithm
	Program / Language
	System Software (OS, VM)
ISA (Instruction Set Architecture)	SW / HW Interface
Implementation of the ISA	Microarchitecture
Building blocks (gates)	Logic
	Devices
	Electrons

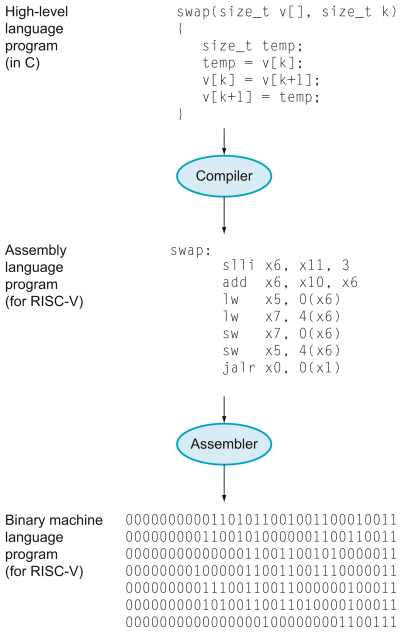
Below your program

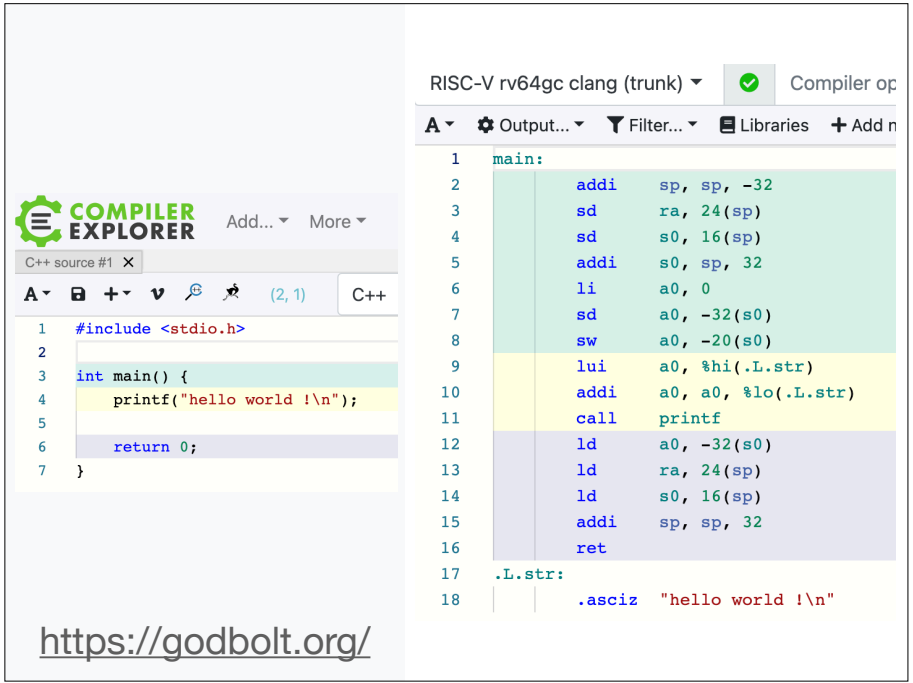


- Application software
  - written in high-level language
- System software
  - compiler: translates HLL code to machine code
  - operating system: service code
    - handling input/output
    - managing memory and storage
    - scheduling tasks & sharing resources
- Hardware
  - processor, memory, I/O 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





https://godbolt.org/

Looking into an executable

```c
#include <stdio.h>

int main() {
    printf("hello world !\n");

    return 0;
}
```

```
$ objdump -d
```

Instruction set architecture

"the words of a computer's language are called *instructions*, and its vocabulary is called an *instruction set*" [Computer Organization and Design, P&H]

`the basic job of a CPU is to execute instructions`

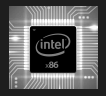
Definitions

- Instruction set
  - the **repertoire** of instructions of a computer
  - **different computers have different instruction sets**
    - … with many aspects in common
- Instruction set architecture (ISA)
  - a **contract** between hardware and software
  - defines instruction format, syntax, and semantics
- Microarchitecture
  - implementation of the architecture, i.e., how the processor physically executes the instructions
  - e.g., cache sizes, core frequency, pipelining, branch prediction
  - different microarchitectures can use the same ISA (e.g., Intel vs AMD for x86)

Major ISAs

- Major ISAs
  - X86 (Intel/AMD)
    - desktop computers, laptops, servers
  - ARM
    - widely used on embedded, phones, recently on laptops, servers
  - RISC-V
    - fast-growing, used in embedded systems, servers, personal computers
- Even more microarchitectures
  - Apple/Samsung/Qualcomm have their own microarchitecture (implementation) of ARM
  - Intel/AMD have different microarchitectures for x86

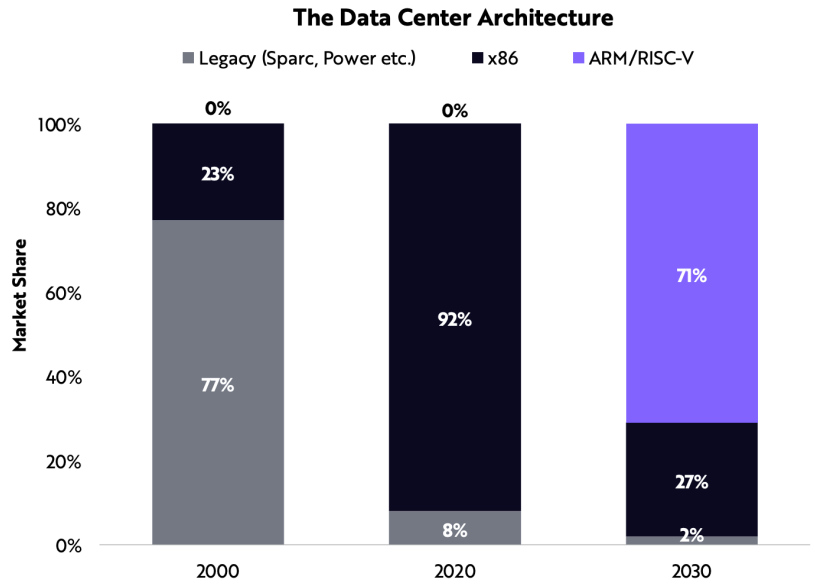
```c
int sum(int *p, int n) {
    int i, sum = 0;
    for (i = 0 ; i < n ; i ++) {
        sum += p[i];
    }
    return sum;
}
```



```
sum(int*, int):
        push    rbp
        mov     rbp, rsp
        mov     qword ptr [rbp - 8], rdi
        mov     dword ptr [rbp - 12], esi
        mov     dword ptr [rbp - 20], 0
        mov     dword ptr [rbp - 16], 0
.LBB0_1:
        mov     eax, dword ptr [rbp - 16]
        cmp     eax, dword ptr [rbp - 12]
        jge     .LBB0_4
        mov     rax, qword ptr [rbp - 8]
        movsxd  rcx, dword ptr [rbp - 16]
        mov     eax, dword ptr [rax + 4*rcx]
        add     eax, dword ptr [rbp - 20]
        mov     dword ptr [rbp - 20], eax
        mov     eax, dword ptr [rbp - 16]
        add     eax, 1
        mov     dword ptr [rbp - 16], eax
        jmp     .LBB0_1
.LBB0_4:
        mov     eax, dword ptr [rbp - 20]
        pop     rbp
        ret
```

```
sum(int*, int):
        sub     sp, sp, #16
        str     r0, [sp, #12]
        str     r1, [sp, #8]
        mov     r0, #0
        str     r0, [sp]
        str     r0, [sp, #4]
        b       .LBB0_1
.LBB0_1:
        ldr     r0, [sp, #4]
        ldr     r1, [sp, #8]
        cmp     r0, r1
        bge     .LBB0_4
        b       .LBB0_2
.LBB0_2:
        ldr     r0, [sp, #12]
        ldr     r1, [sp, #4]
        ldr     r1, [r0, r1, lsl #2]
        ldr     r0, [sp]
        add     r0, r0, r1
        str     r0, [sp]
        b       .LBB0_3
.LBB0_3:
        ldr     r0, [sp, #4]
        add     r0, r0, #1
        str     r0, [sp, #4]
        b       .LBB0_1
.LBB0_4:
        ldr     r0, [sp]
        add     sp, sp, #16
        bx      lr
```

```
sum(int*, int):
        addi    sp,sp,-48
        sw      ra,44(sp)
        sw      s0,40(sp)
        addi    s0,sp,48
        sw      a0,-36(s0)
        sw      a1,-40(s0)
        sw      zero,-24(s0)
        sw      zero,-20(s0)
        j       .L2
.L3:
        lw      a5,-20(s0)
        slli    a5,a5,2
        lw      a4,-36(s0)
        add     a5,a4,a5
        lw      a5,0(a5)
        lw      a4,-24(s0)
        add     a5,a4,a5
        sw      a5,-24(s0)
        lw      a5,-20(s0)
        addi    a5,a5,1
        sw      a5,-20(s0)
.L2:
        lw      a4,-20(s0)
        lw      a5,-40(s0)
        blt     a4,a5,.L3
        lw      a5,-24(s0)
        mv      a0,a5
        lw      ra,44(sp)
        lw      s0,40(sp)
        addi    sp,sp,48
        jr      ra
```



Big Ideas 2021, ARK Investment Management LLC

CISC vs RISC

- Complex Instruction Set Computer (CISC)
  - more complex instructions, can directly manipulate memory
  - for example:
    - X86, X86_64 (Intel and AMD, desktop/laptop/server)
      - X86* internally are still RISC
- Reduced Instruction Set Computer (RISC)
  - simpler instructions, load-store architecture
  - for example:
    - ARM (smartphone/pad)
    - **RISC-V** (free ISA, closer to MIPS than other ISAs)
    - Others: Power ISA, SPARC, etc

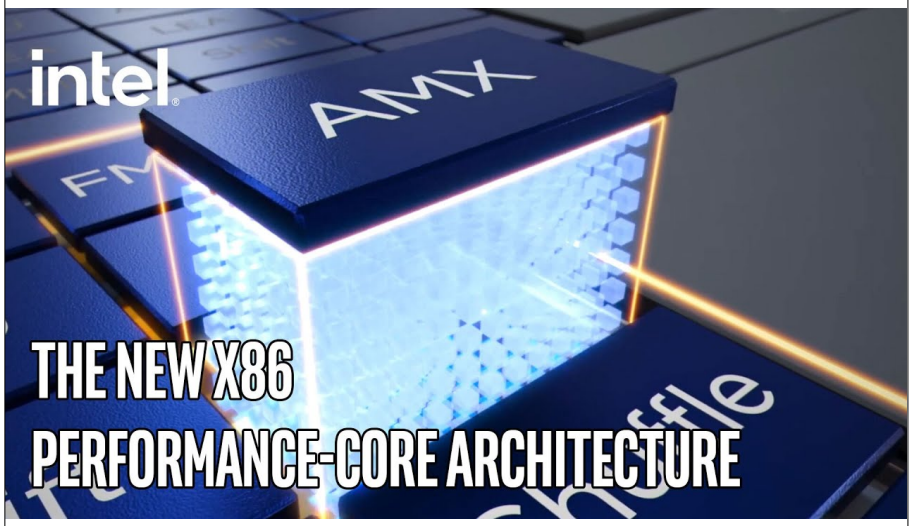
CISC vs RISC

- CISC
  - early trend was to add more and more instructions
  - VAX architecture had an instruction to multiply polynomials!
  - operations may directly manipulate memory
- RISC philosophy
  - keep the instruction set **small and simple**
  - makes it easier to build fast hardware.
  - let software do complicated operations by composing simpler ones

ACM has named **John L. Hennessy**, former President of Stanford University, and **David A. Patterson**, retired Professor of the University of California, Berkeley, recipients of the **2017 ACM A.M. Turing Award** for pioneering a systematic, quantitative approach to the design and evaluation of computer architectures with enduring impact on the microprocessor industry.

https://www.youtube.com/watch?v=3LVeEjsn8Ts

Intel's x86 core



https://www.youtube.com/watch?v=ijTRvIQV7bE

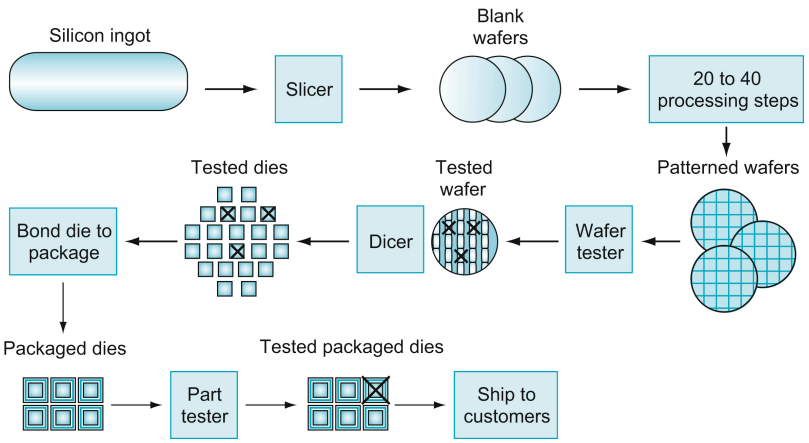
Chip Manufacturing

Semiconductor technology

- Conductors
  - materials that easily conduct
- Insulators
  - materials that do not conduc

- Semiconductors
  - materials with conductivity between conductors and insulators
  - those properties can be controlled by doping (adding/ removing electrons)
  - Silicon (base material for computer technology)

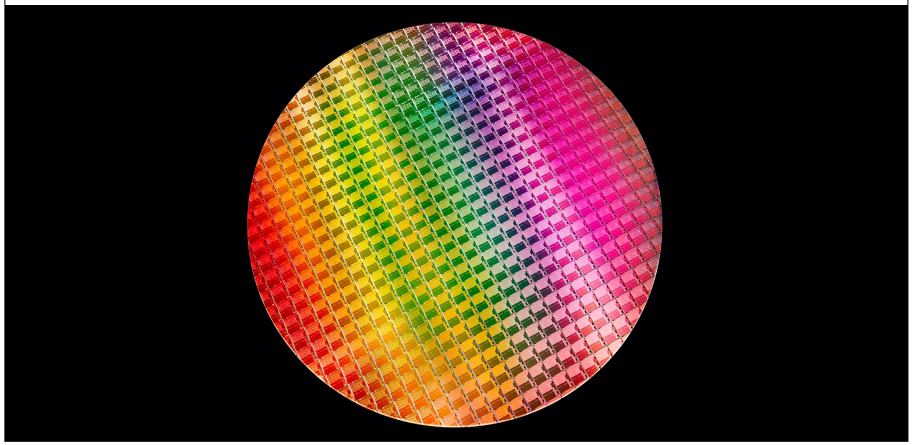
Manufacturing integrated circuits



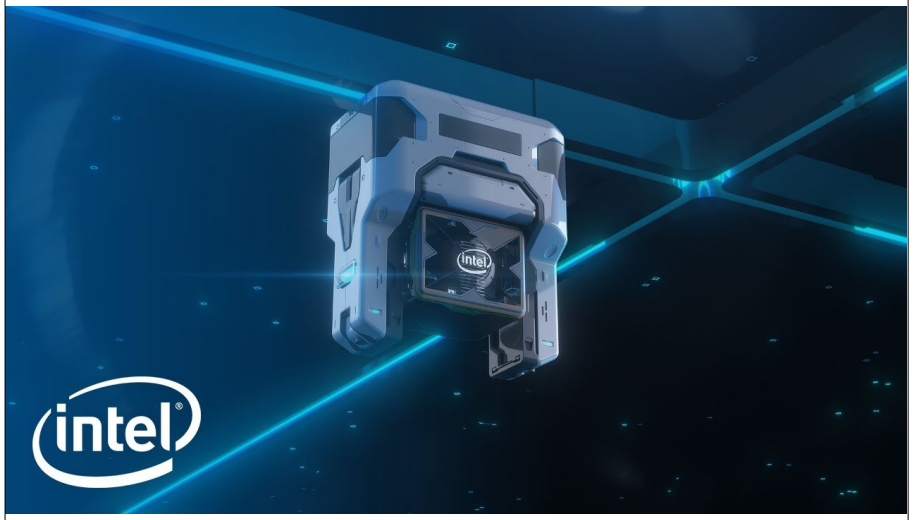
Yield: proportion of working dies per wafer

Intel® Core 10th Gen

- 300mm wafer (almost 1 foot), 506 chips, 10nm technology
- Each chip is 11.4 x 10.7 mm (a bit less than .5 in)



From sand to silicon



https://www.youtube.com/watch?v=_VMYPLXnd7E