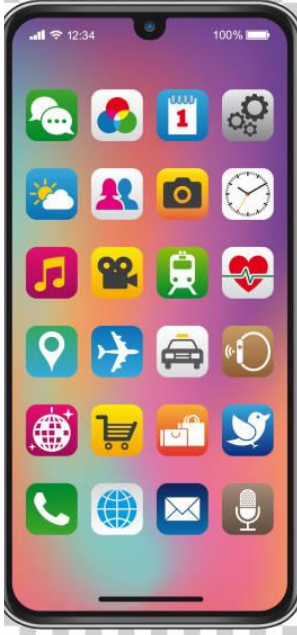
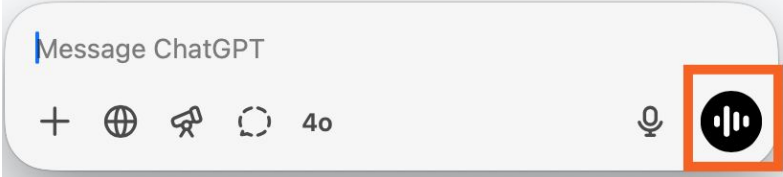


Introduction to Computer Architecture

How software runs on hardware

Rijurekha Sen

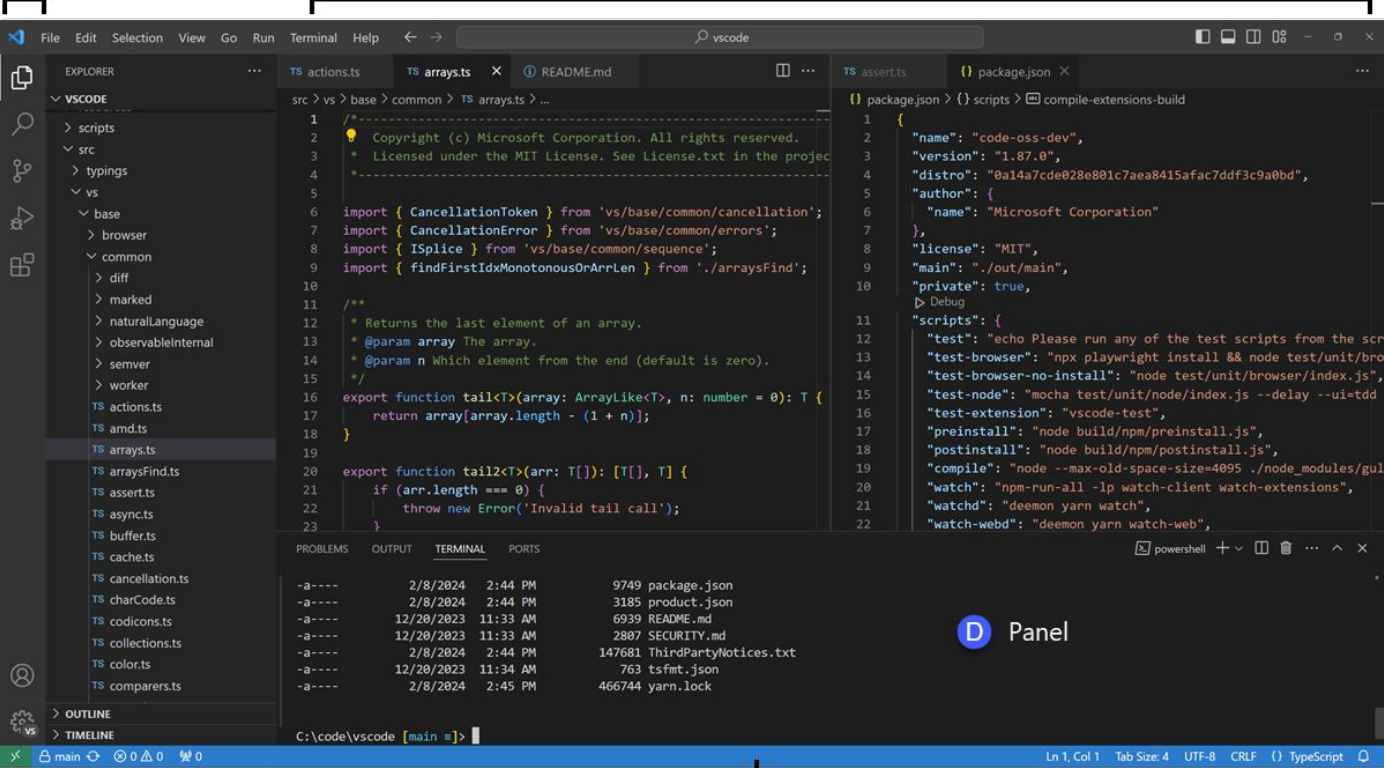
User interaction with computers through software



App software developer interactions with computer through IDE

A Activity Bar

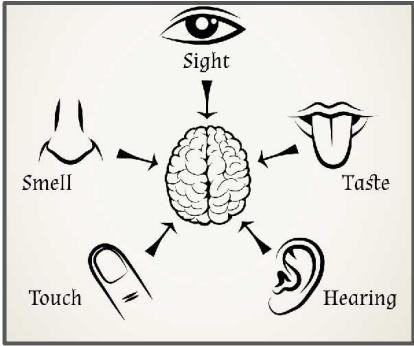
C Editor Groups



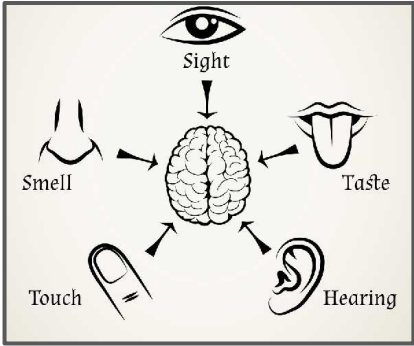
B Primary Side Bar

E Status Bar

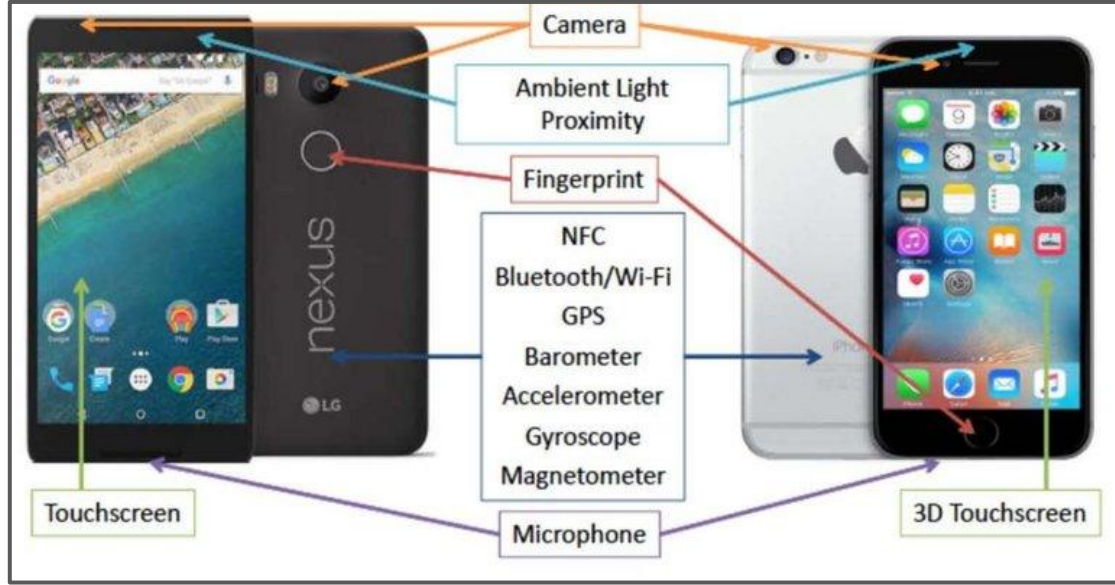
User interactions with computers through hardware



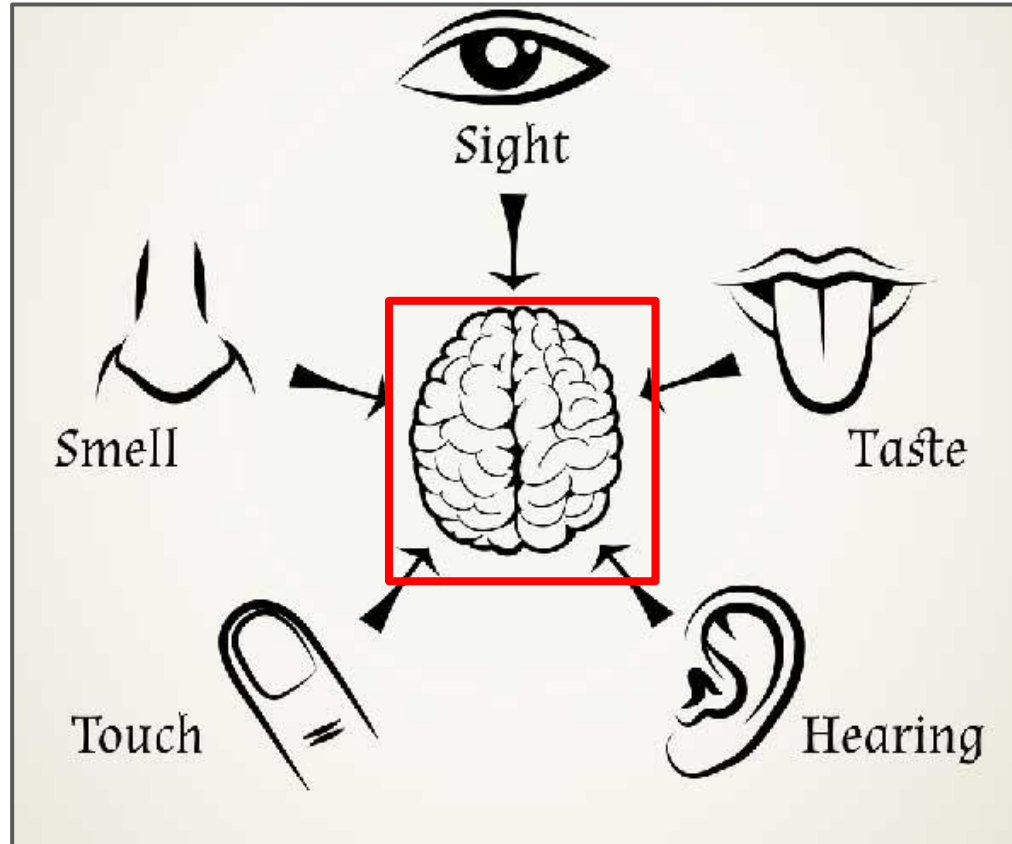
User interactions with computers through hardware



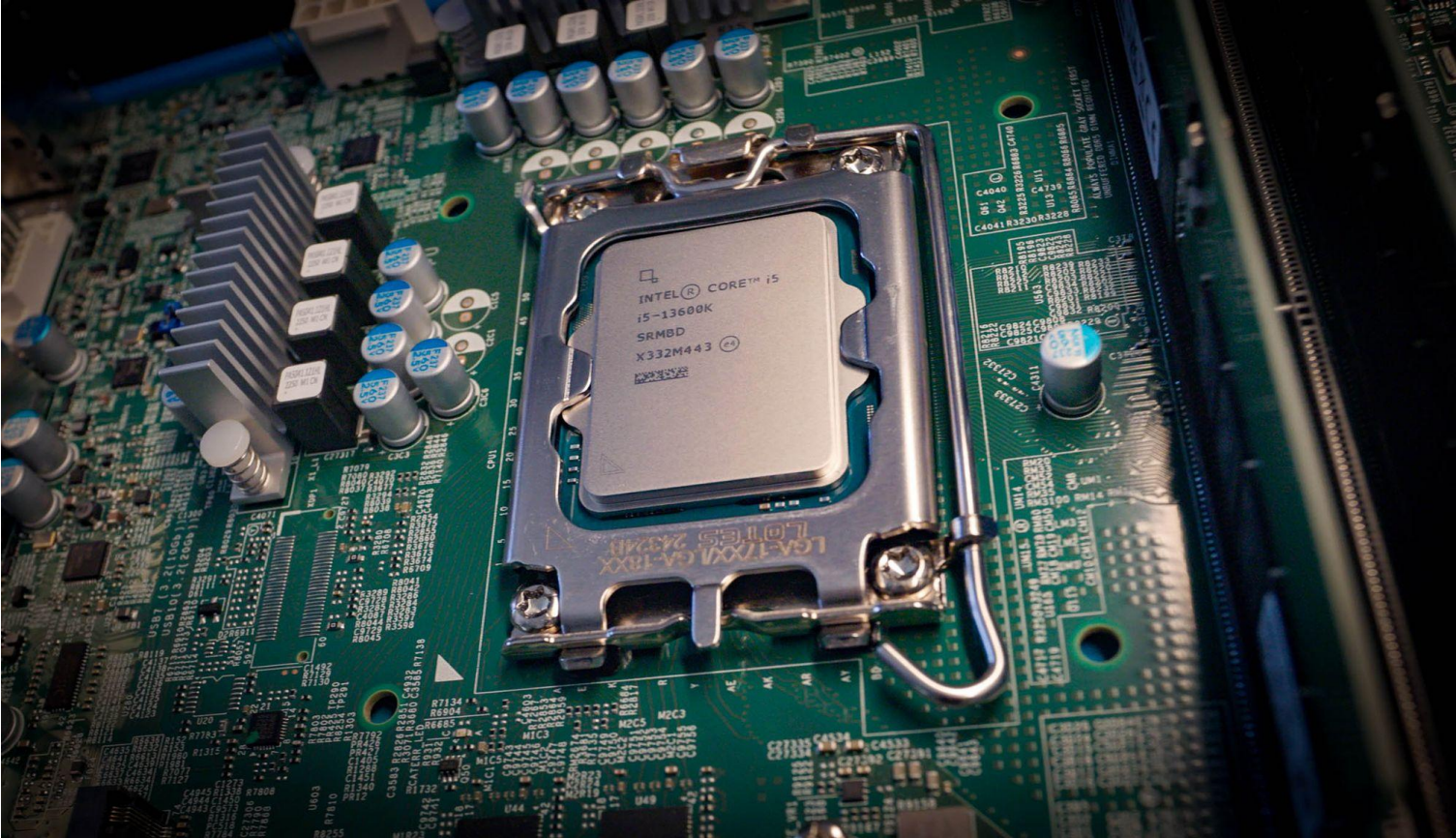
input Device			Output Device	
Modem	Camera	Mouse	Screen	Plotter
Light pen	Touch screen	CD/DVD	Printer	Speaker
Pendrive	Web camera	Keyboard	Head phone	Earphone
BCR	Joystick	Scanner		



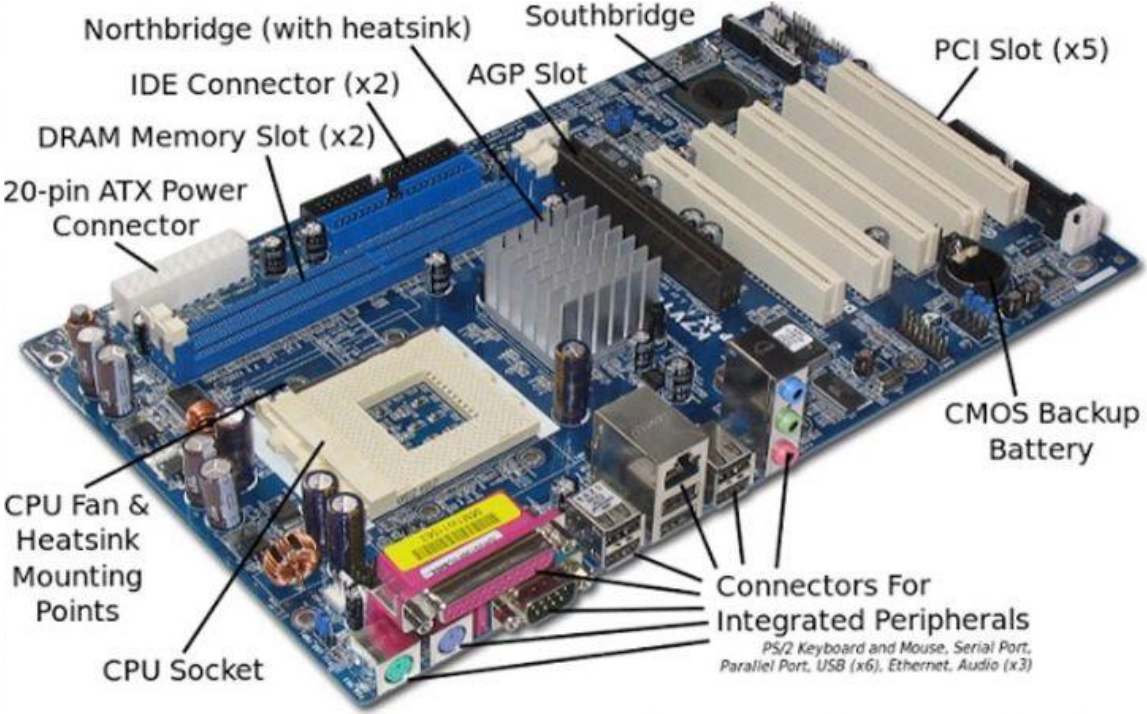
But where/how does the “compute” actually happen?



Processor, Memory, Storage, Bus



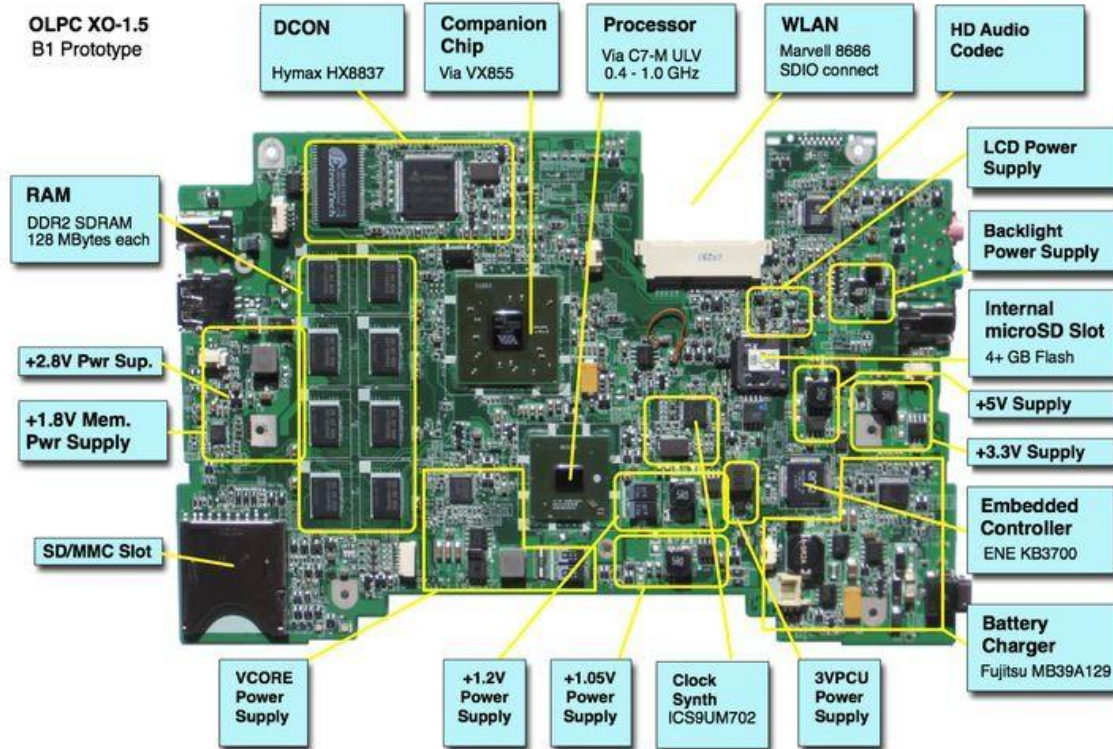
Desktop motherboard



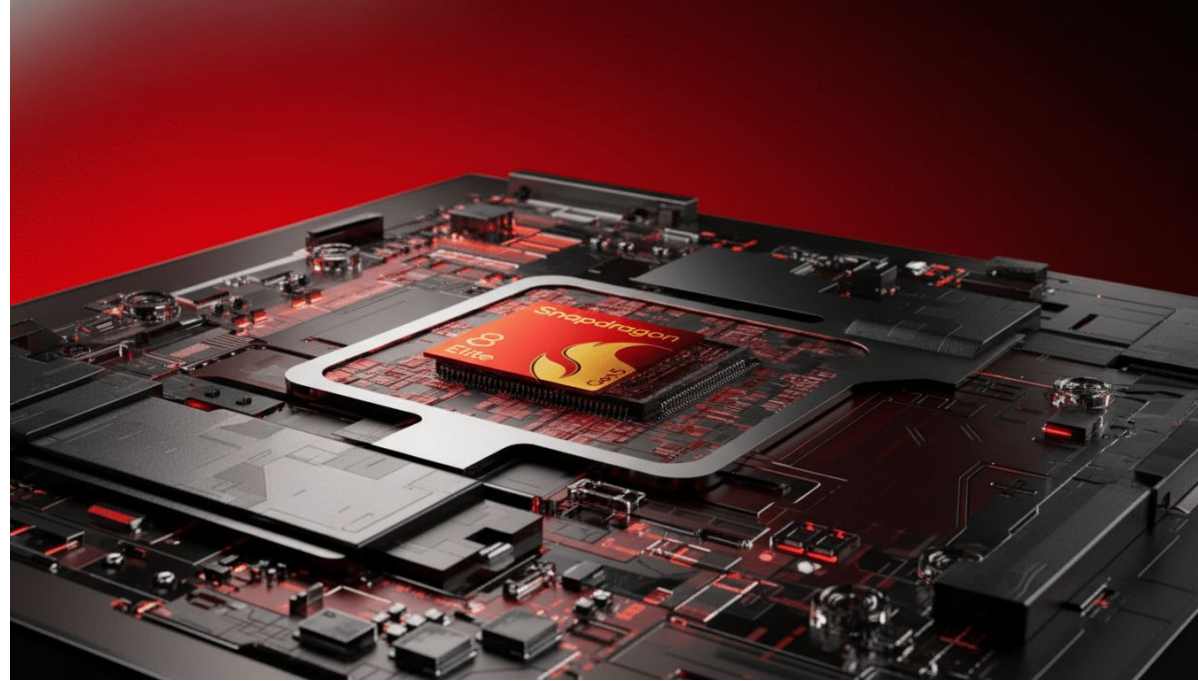
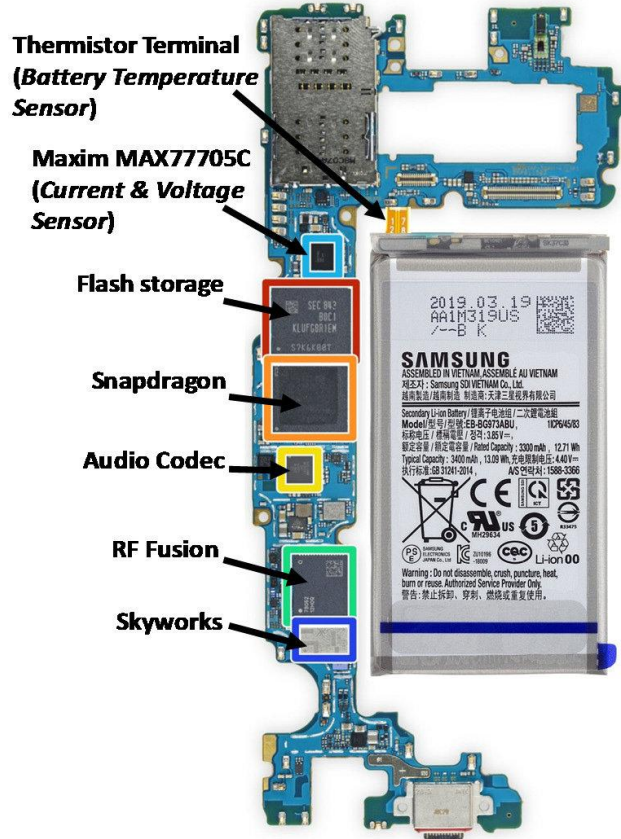
Similar system architecture - smartwatch to data centers



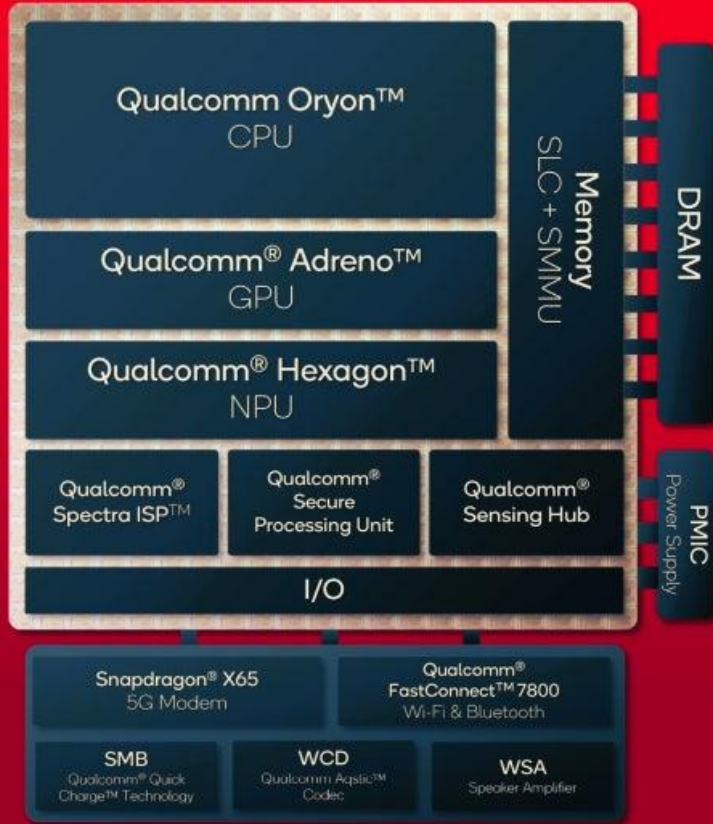
Laptop motherboard



Smartphone motherboard

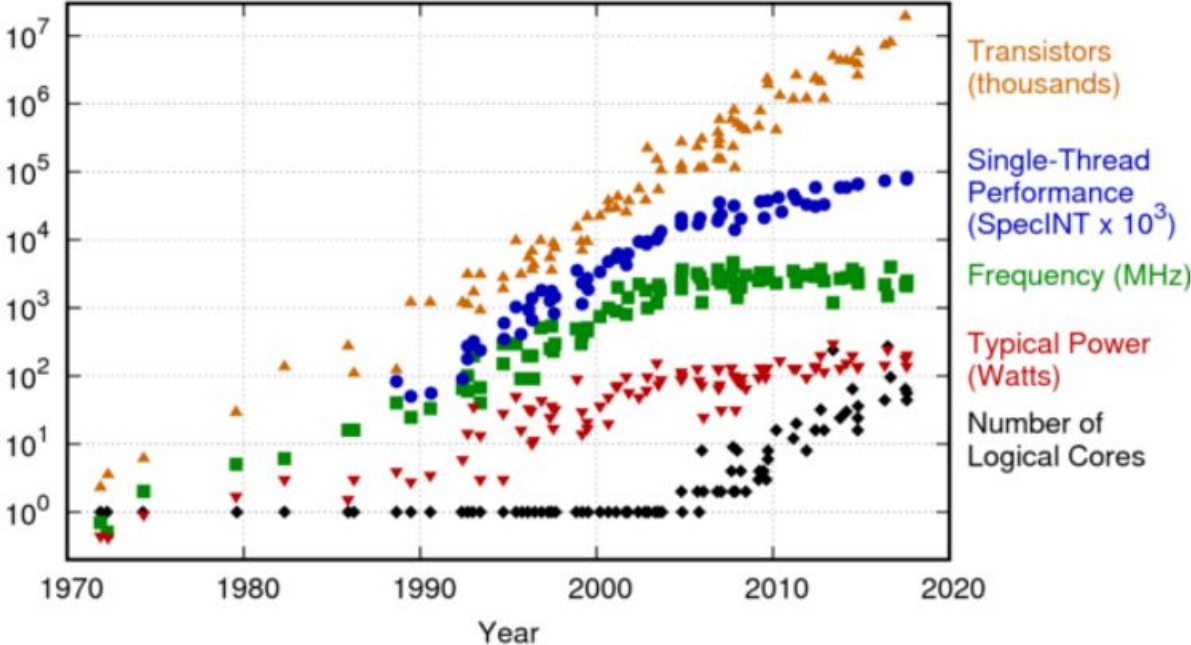


Integrated System on Chip (SOC)



Processor performance

42 Years of Microprocessor Trend Data



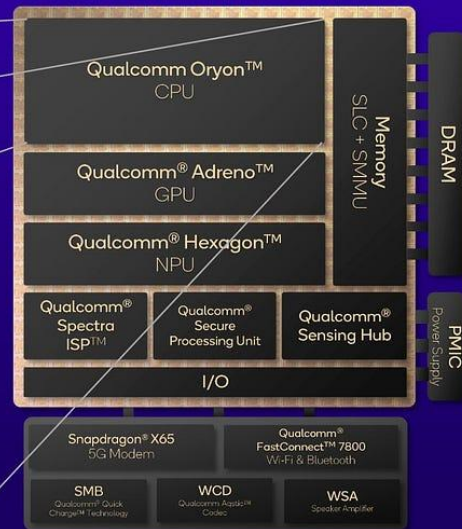
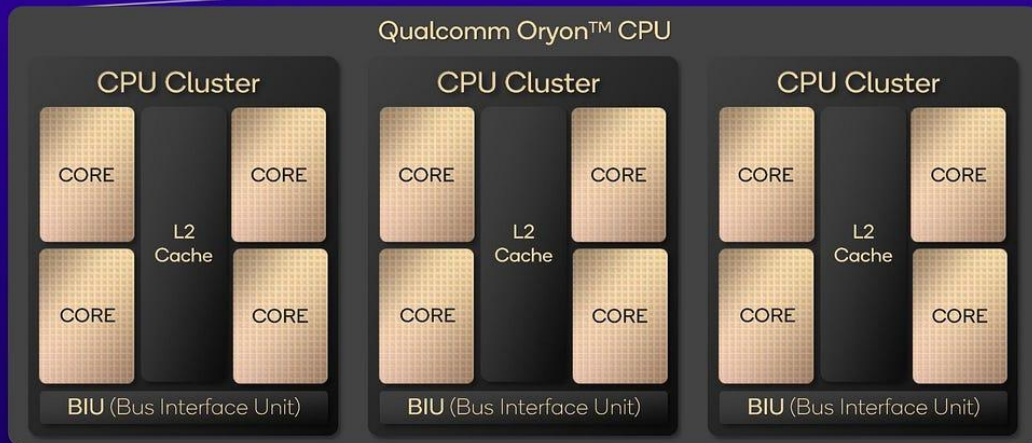
Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2017 by K. Rupp

Additional hardware for power control and heat dissipation

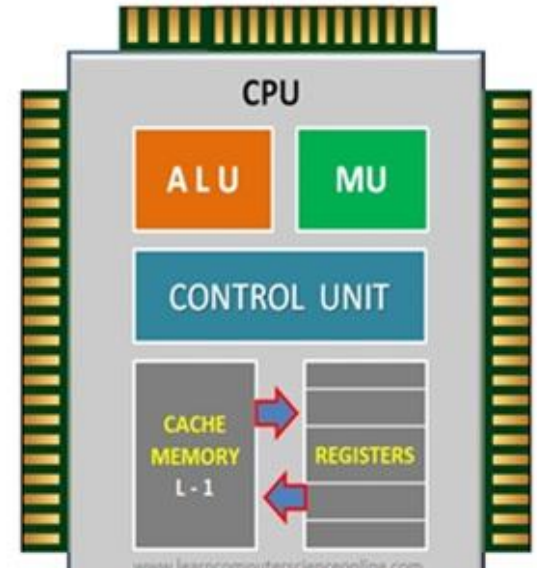
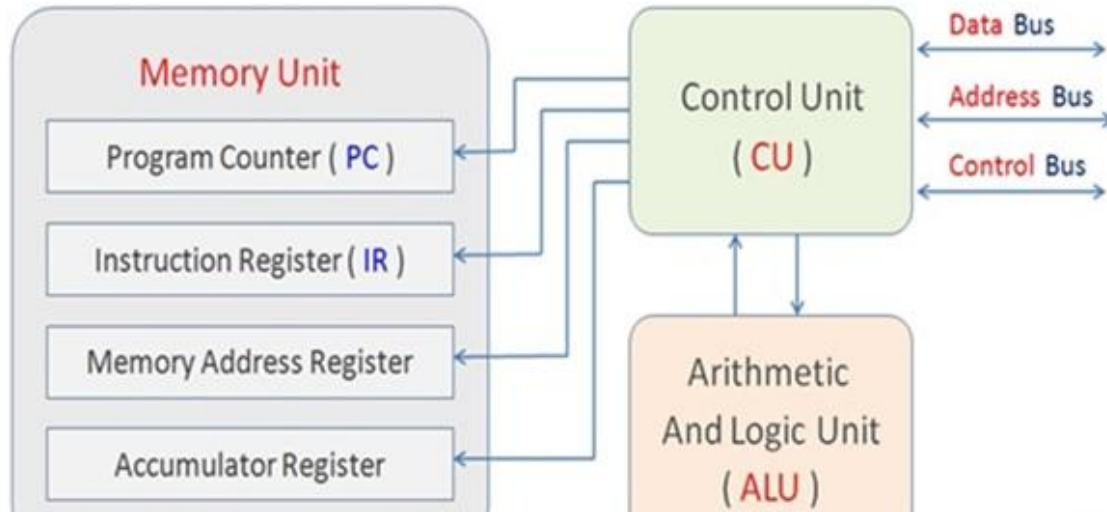


Multicore CPU

12 brand-new custom CPU cores power the SoC



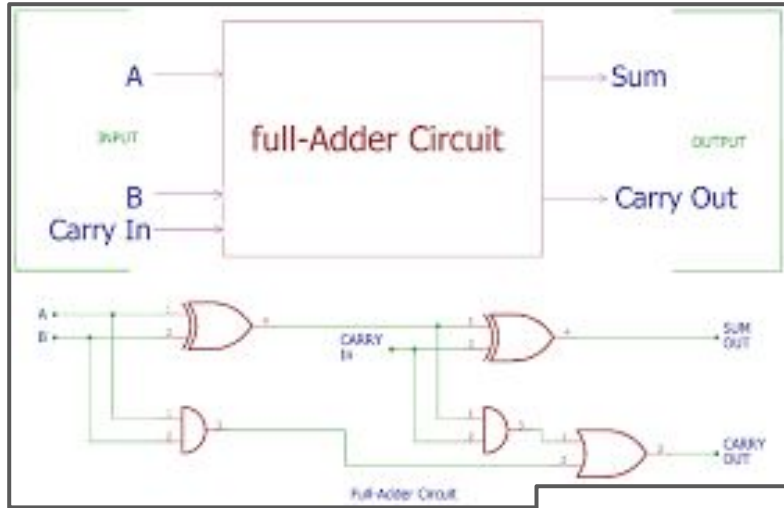
Inside each CPU



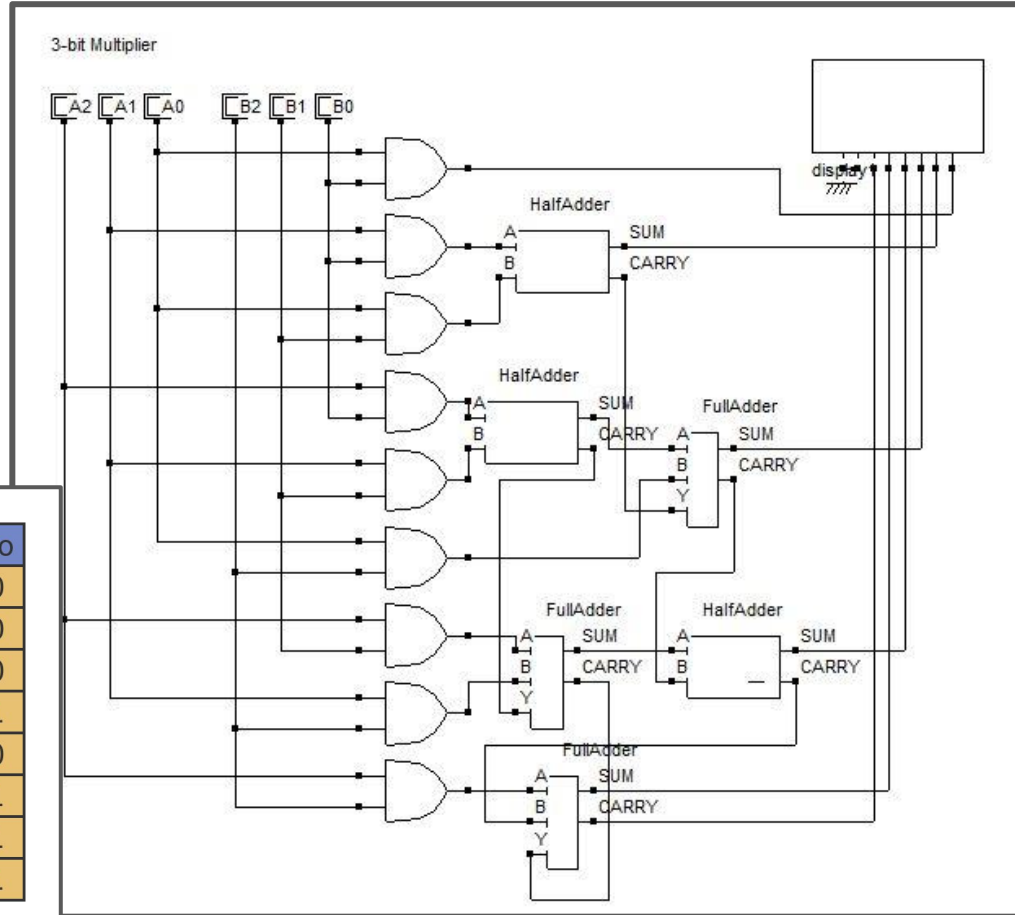
Typical Computer Architecture book for UG students

1. ALU design (digital logic course)

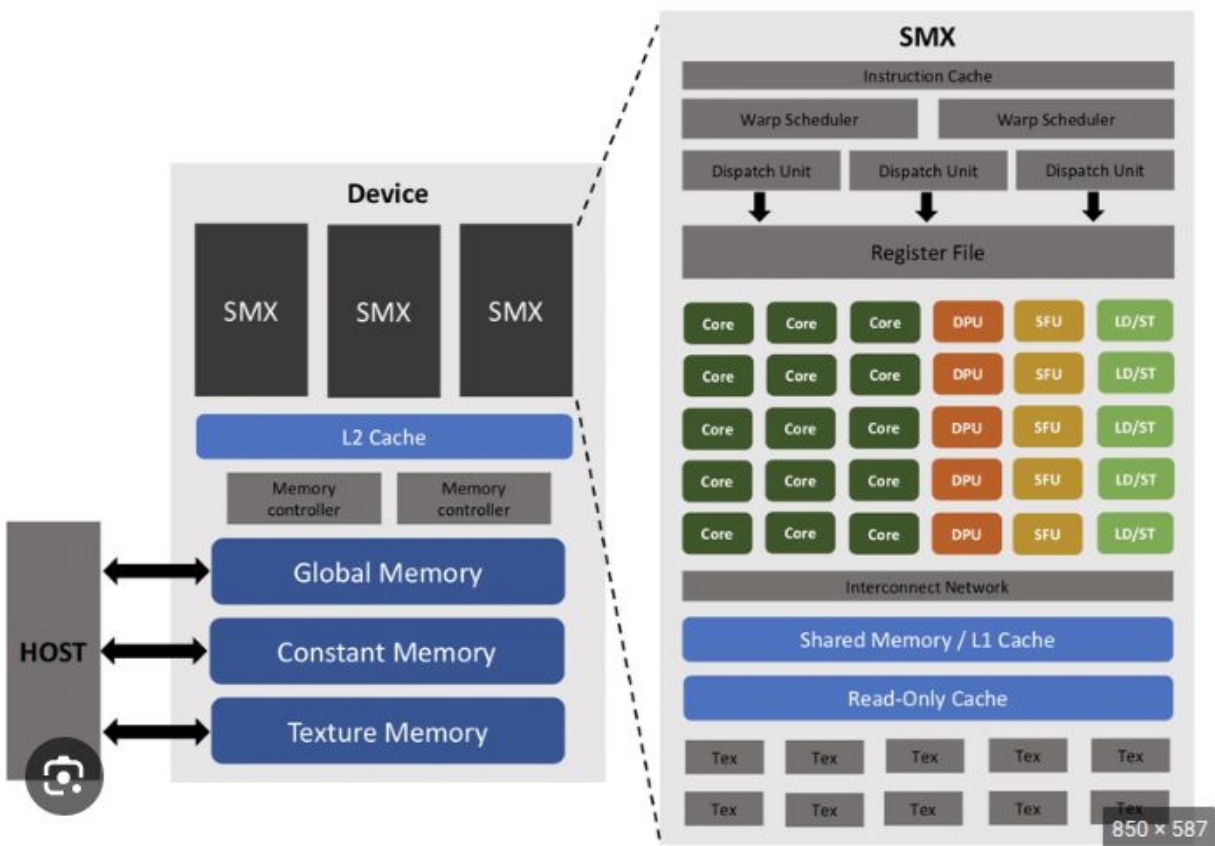
Inside the Arithmetic Logic Unit (ALU)



A	B	C _i	S	C _o
0	0	0	0	0
0	1	0	1	0
1	0	0	1	0
1	1	0	0	1
0	0	1	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1



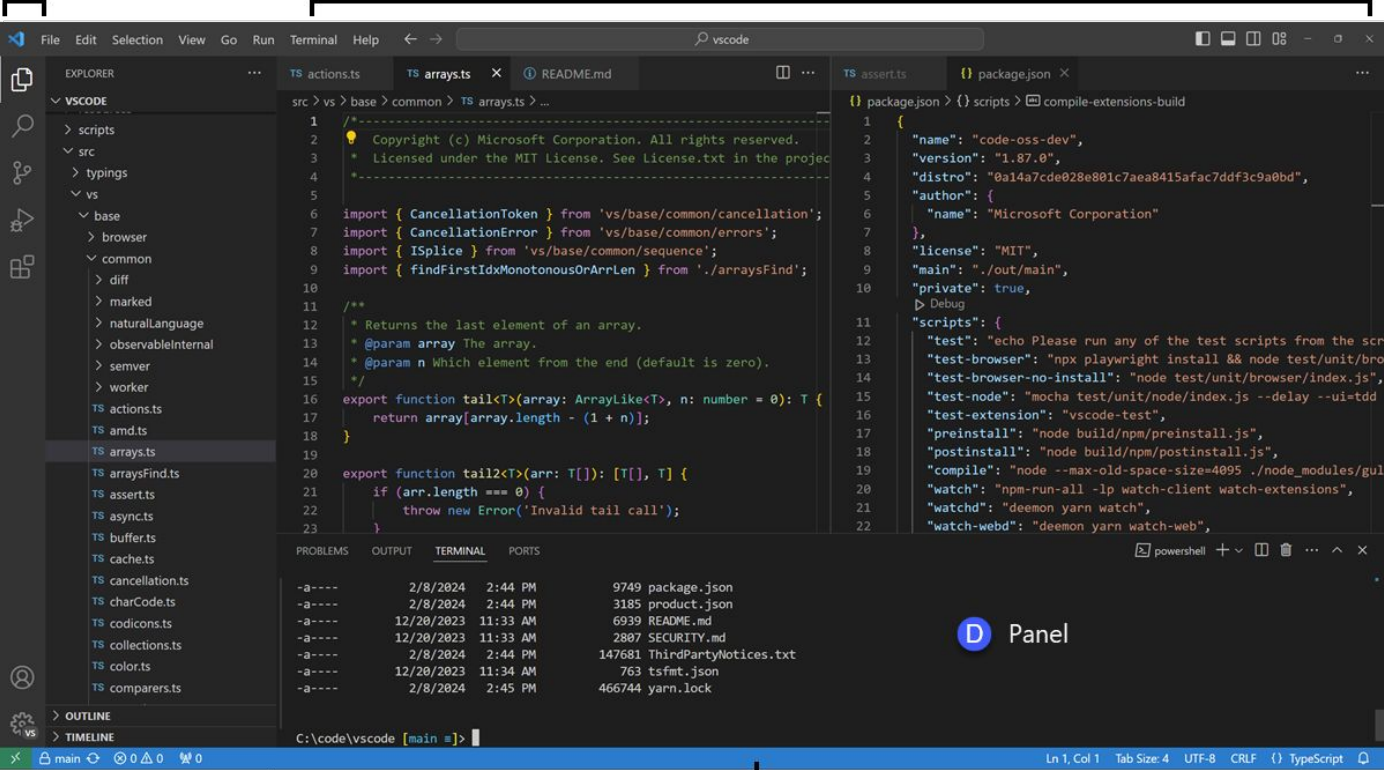
GPU



App software developer interactions with computer through IDE

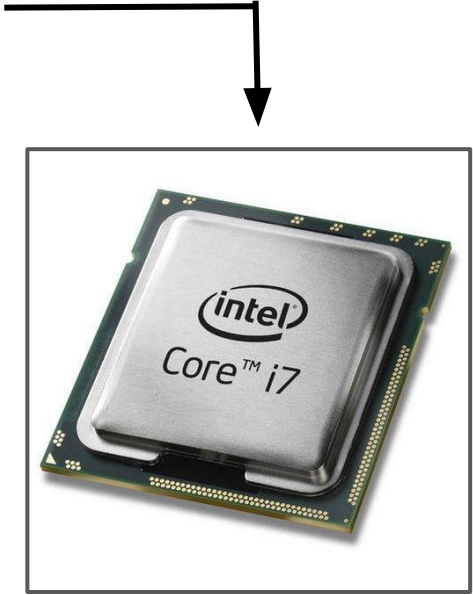
A Activity Bar

C Editor Groups

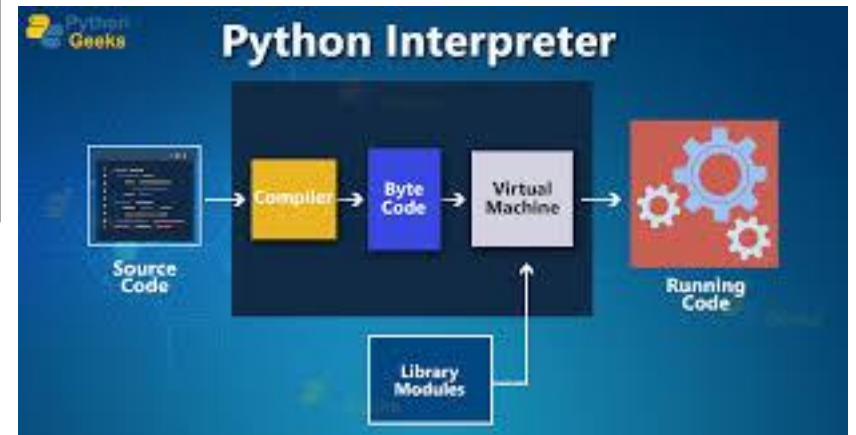
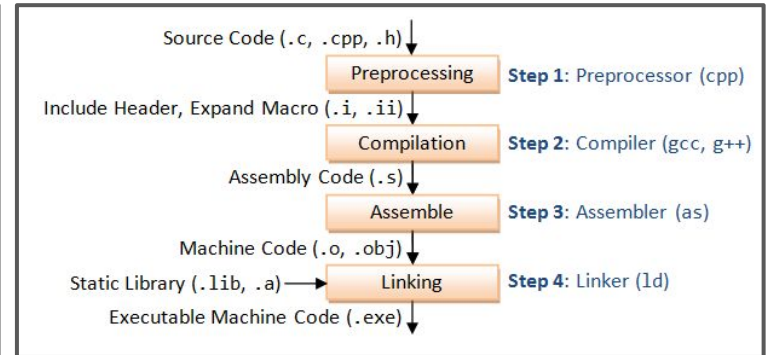
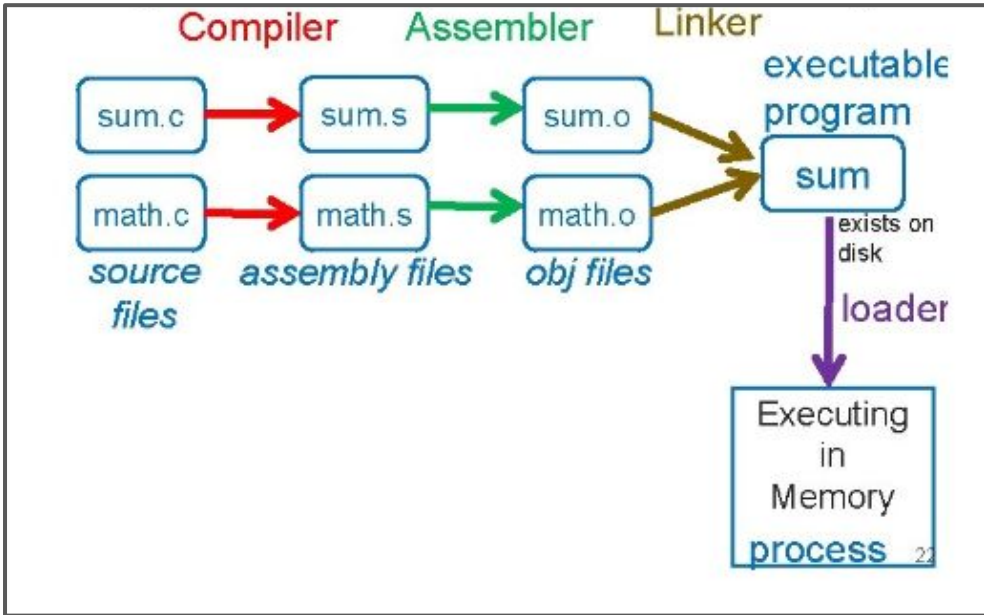


B Primary Side Bar

E Status Bar



From source code to execution



Typical Computer Architecture book for UG students

1. ALU design (digital logic course)

2. Instruction set architecture

Instruction Set Architecture (ISA) - hardware software interface

Application software

`a[i] = b[i] + c;`

↓ Compiler

Systems software
(OS, compiler)

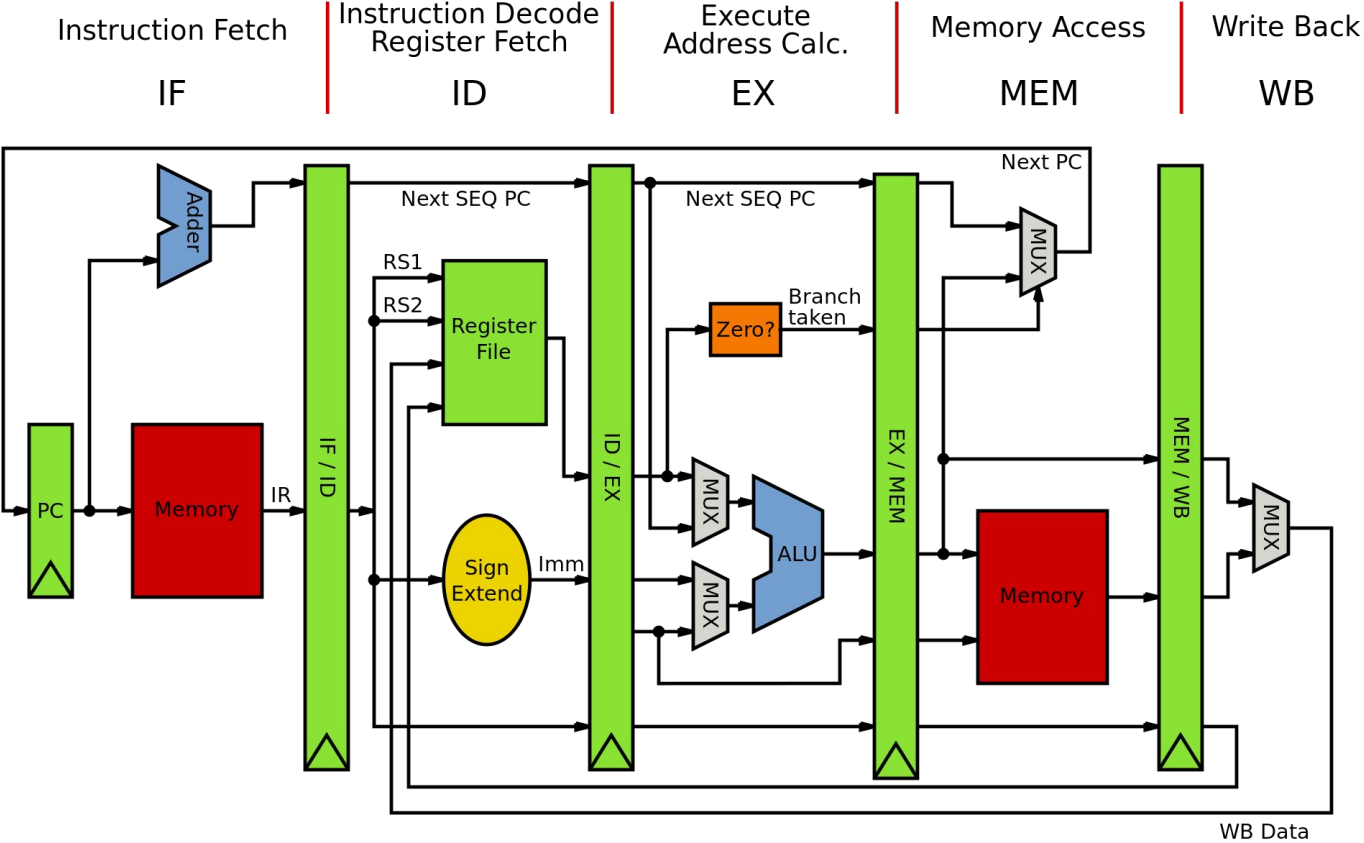
```
lw  $15, 0($2)
add $16, $15, $14
add $17, $15, $13
lw  $18, 0($12)
lw  $19, 0($17)
add $20, $18, $19
sw  $20, 0($16)
```

↓ Assembler

Hardware

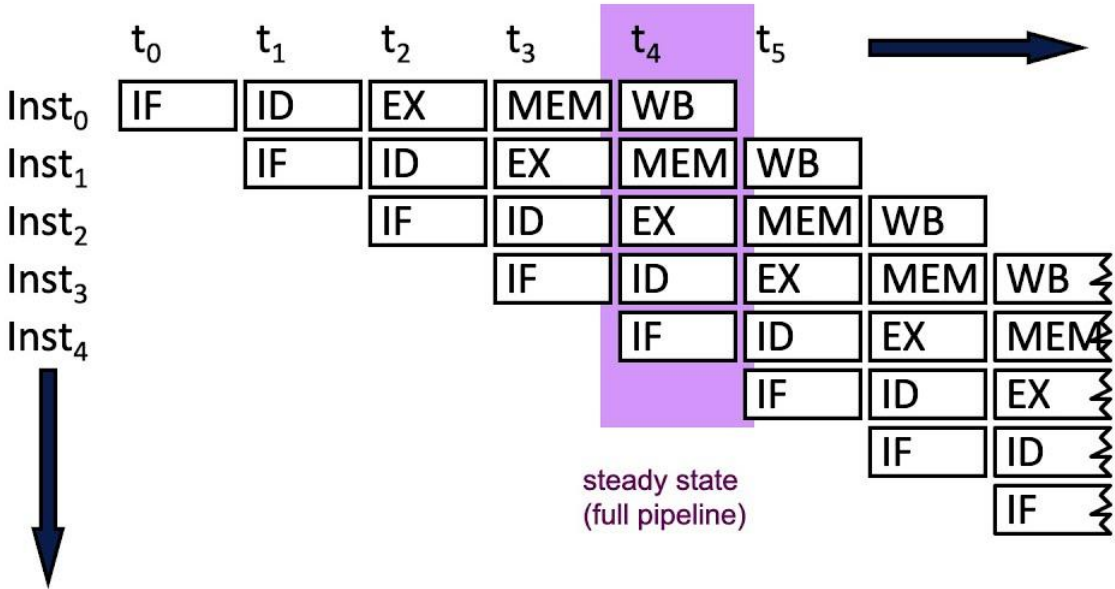
```
000000101100000
110100000100010
...
```

How does a single instruction run on CPU?



How to improve performance with instruction pipelining?

Illustrating Pipeline Operation: Operation View



Hazard detection and handling

Consider the following set of instructions in a 5-stage pipeline.
Operands are read in ID.

MEM is memory Write for result; RW is register Write for result

ADD X3, X6, X5 - Result to be written in X3

SUB X4, X3, X5 - X3 has one of the operand

OR X6, X3, X7 - X3 has one of the operand

AND X8, X3, X7 - X3 has one of the operand

XOR X12, X3, X10 - X3 has one of the operand

X3 is accessed
in READ mode;
expect the result
of ADD to be
available in X3

But result of
ADD written
in X3 at t5

	t1	t2	t3	t4	t5	t6	t7	t8	t9
ADD X3, X6, X5	IF	ID	IE	MEM	RW X3	--	--	--	--
SUB X4, X3, X5	--	IF	ID X3	IE	MEM	RW	--	--	--
OR X6, X3, X7	--	--	IF	ID X3	IE	MEM	RW	--	--
AND X8, X3, X9	--	--	--	IF	ID X3	IE	MEM	RW	--
XOR X12, X3, X11	--	--	--	--	IF	ID X3	IE	MEM	RW

Note when
each instruction
is accessing X3

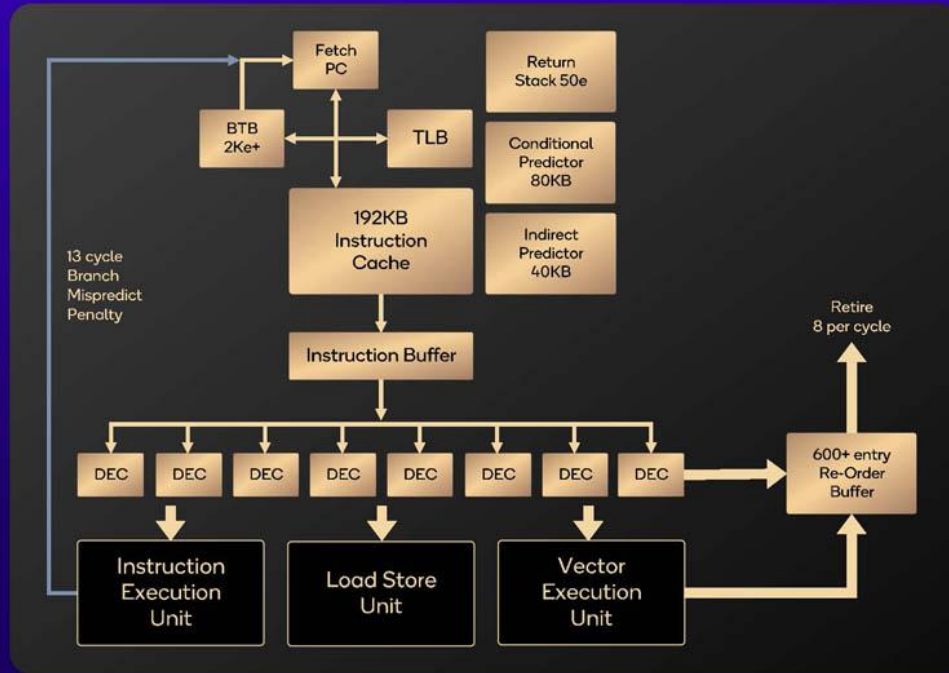
Typical Computer Architecture book for UG students

1. ALU design (digital logic course)
2. Instruction set architecture
- 3. Processor design**

Out of order execution, branch prediction, superscalar,
vector processors

Processor design - micro-architecture

Fetch Pipeline, including Decode, Rename, and Retire



Typical Computer Architecture book for UG students

1. ALU design (digital logic course)
2. Instruction set architecture
3. Processor design
4. **Memory hierarchy and caching**

Memory hierarchy

A Typical Memory Hierarchy

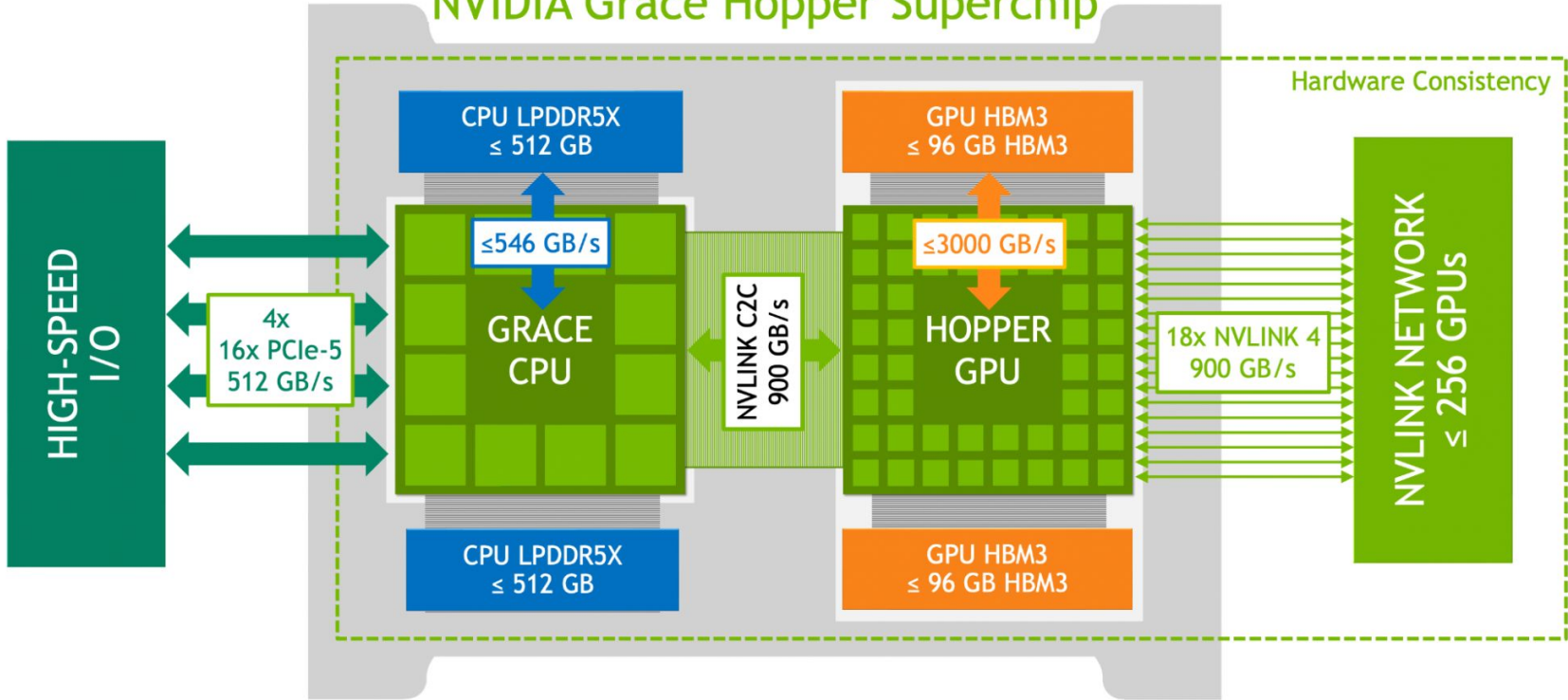
- Everything is a cache for something else...

The diagram illustrates a memory hierarchy with levels from top to bottom: Registers, Level 1 Cache, Level 2 Cache, Level 3 Cache, Main Memory, Flash Drive, and Hard Disk. Dashed horizontal lines separate the levels into three categories: 'On the datapath' (Registers), 'On chip' (Level 1-3 Caches), and 'Mechanical devices' (Hard Disk). 'Other chips' (Main Memory and Flash Drive) are also indicated. A table to the right provides quantitative data for each level.

	Access time	Capacity	Managed By
On the datapath	1 cycle	1 KB	Software/Compiler
Level 1 Cache	2-4 cycles	32 KB	Hardware
Level 2 Cache	10 cycles	256 KB	Hardware
On chip	40 cycles	10 MB	Hardware
Other chips	200 cycles	10 GB	Software/OS
Flash Drive	10-100us	100 GB	Software/OS
Mechanical devices	10ms	1 TB	Software/OS

CPU-GPU systems

NVIDIA Grace Hopper Superchip



Follow-up systems courses after Computer Architecture

1. Operating Systems
2. Parallel Programming
3. Compilers
4. Database systems
5. Networking