3/25/24

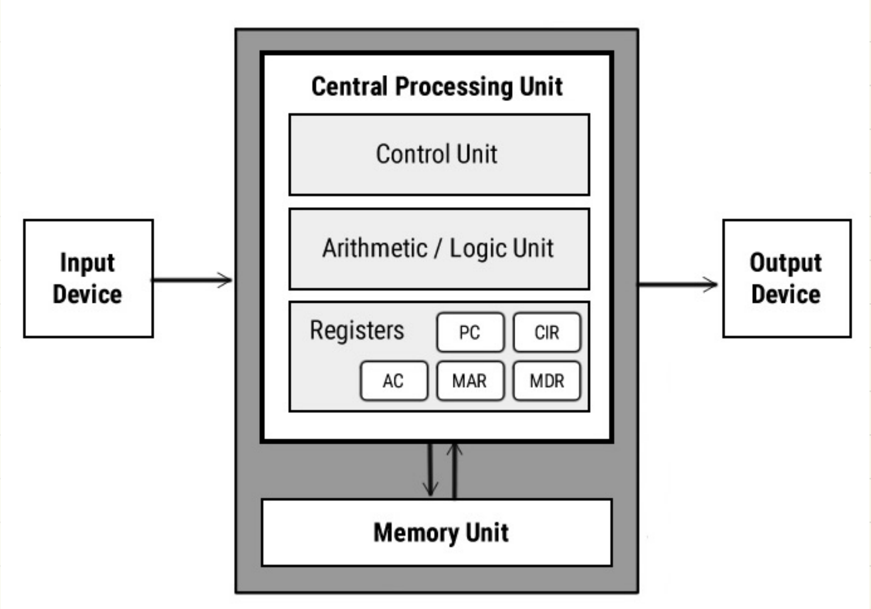
Welcome to 451!

OS: a program that abstracts & manages hardware resources.
for user program.

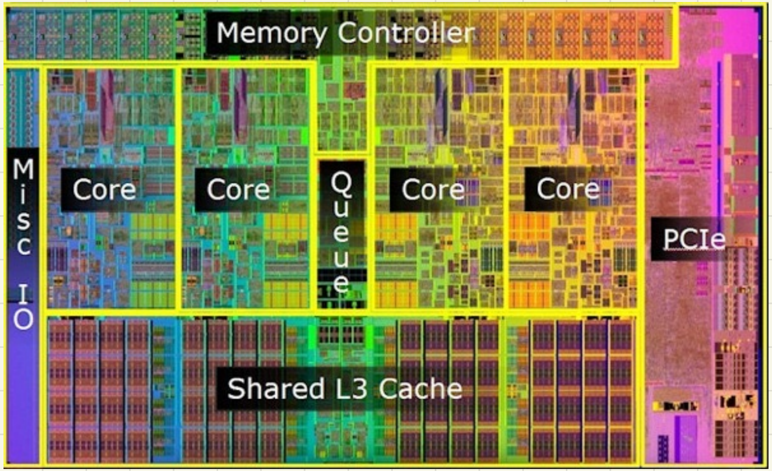
what are ⤴

in this class we care about
CPU, DRAM, & storage devices

→ CPU.



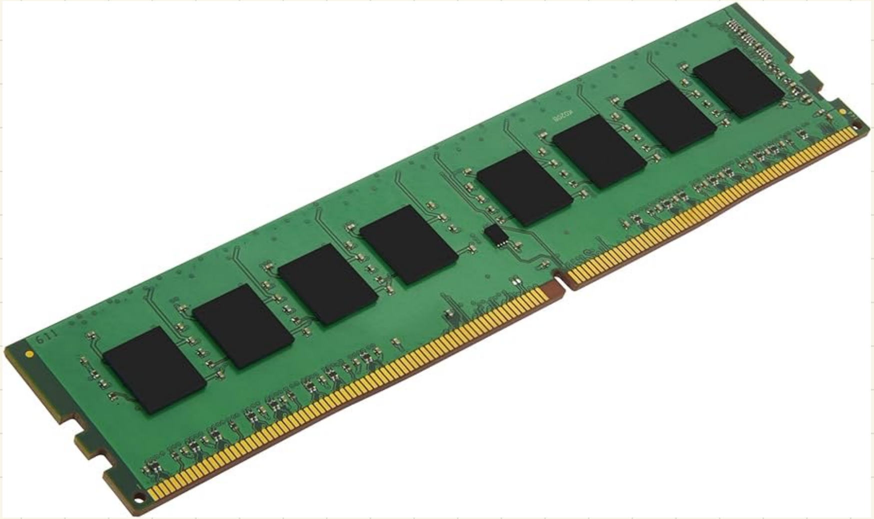
single core



multicore

- → executes instr.
- → %rip holds address of next instr. to execute.
- → OS sets up the %rip for a new process.

- → Physical memory
  - → byte-addressable, slower than CPU (why we have caches)



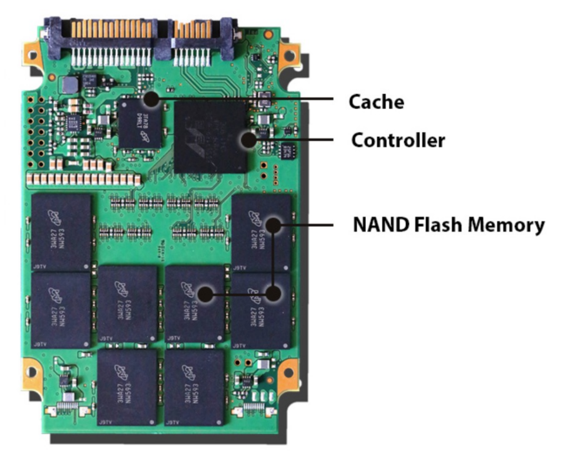
volatile = data does not last through a power cycle

→ Storage Devices

- → persistent, large capacity (TBs)
  (non-volatile)
- → block-addressable, way slower than memory

performance also differs based on access patterns (sequential or random)

hard drive
sector size 512 bytes



solid state drive (SSD)
page size 4096 bytes.

Other I/O Devices

Input : mouse , keyboard , webcam , microphone

Output : monitor , headphones , speakers

How does the OS abstract these resources?

CPU => process

DRAM => virtual memory

Storage => Filesys (files, directories)

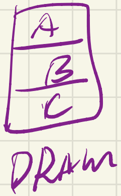
Network => Network stack (TCP/IP)

Why provide them?

→ Ease of use [Illusionist]

→ simplified services

- Storage ⇒ filesys : named files, folder organization
- DRAM ⇒ virtual memory : a process owns the entire address space



→ mask hw limitations

- filesys: file handles bytes (hides blocks)
- virtual memory: allows process to use more than physically available

→ Common interface [glue]

→ allows processes to share & communicate

→ programs portable across hw

→ managed access [referee].

- → resource management
  - → schedule processes onto a single CPU (saves & restore each process's state)
  - → sharing of physical memory
- → isolation (or OS)
  - → a process can't read other processes' memory
- → managed sharing
  - → explicitly requested shared memory

How does OS provide these abstractions?

What this class is about!