CH-1 Computer System Overview

11/09/25

Q1) What is an Operating System?

→ It acts as an ~~interpreter~~ interface between hardware & software
→ It is also known an resource manager.
→ It acts as a governance of a body to manage the resources.
→ There are 2 types of interpreter:-
  (1) Text based OS (shell)
  (2) Graphical User Interface

→ Booting Process
1) Power on: CPU starts
2) POST: BIOS checks hardware (RAM, CPU, Keyboard)
3) BIOS/UEFI: Finds bootable device (HDD/SSD/USB)
4) Boot loader: Loads OS Kernel → GRUB/GRUB2/LILO
5) Kernel: Initialize memory, devices & system functions.
6) Services & UI: GUI/CLI starts
7) Login screen

→ Von-neuman Architecture

This architecture follows Store Program Concept whose execution is taking place in primary memory.
→ This consists of 2 primary units: i.e. CU & ALU
→ Control Unit is responsible for generation of control & timing signals during the execution of micro-operations.

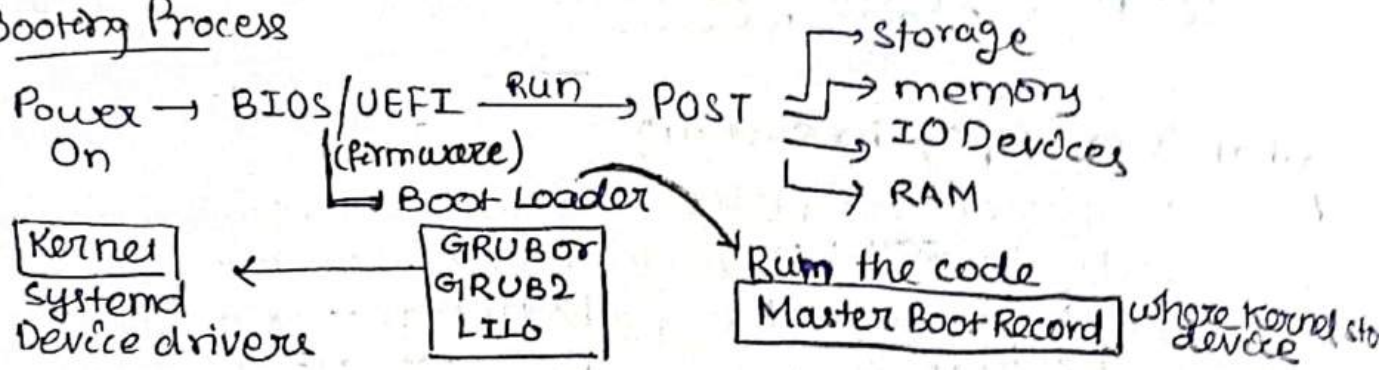
• Micro Operation:-
Operation which performs the data stored in the register during 1 clock cycle is known as Micro Operation.

12/09/25

Firmware:- Firmware is a special type of software which is permanently programmed into Hardware device.
→ It provides low level control of hardware & helps to work or run.

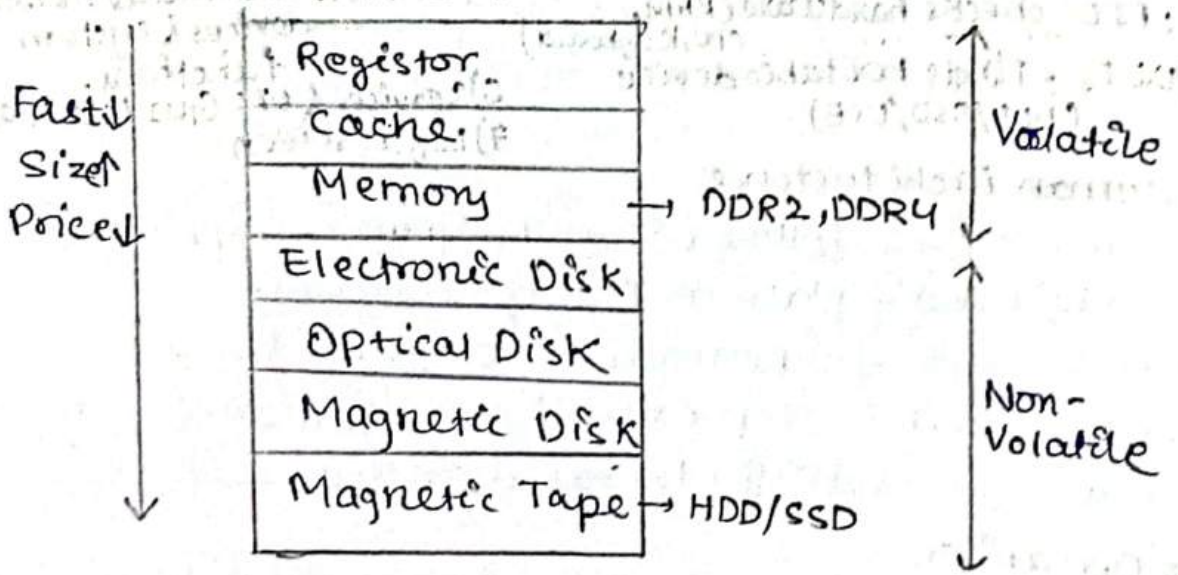
Boot Loader:-
(1) Locate the OS Kernel on a disc
(2) Load the Kernel into computer's memory
(3) Start running the Kernel code

Booting Process



Storage Structure :- General purpose computers executes most the program from the main memory also known as rewrita memory (RW memory)

Main memory is implemented in semiconducted technolog known as DRAM.



In computer operating system the resources are divided into 2 categorie

(i) HardWare Resource : IO devices, Processors

(ii) Software Resource : Device Drivers, Pipes, sdma4

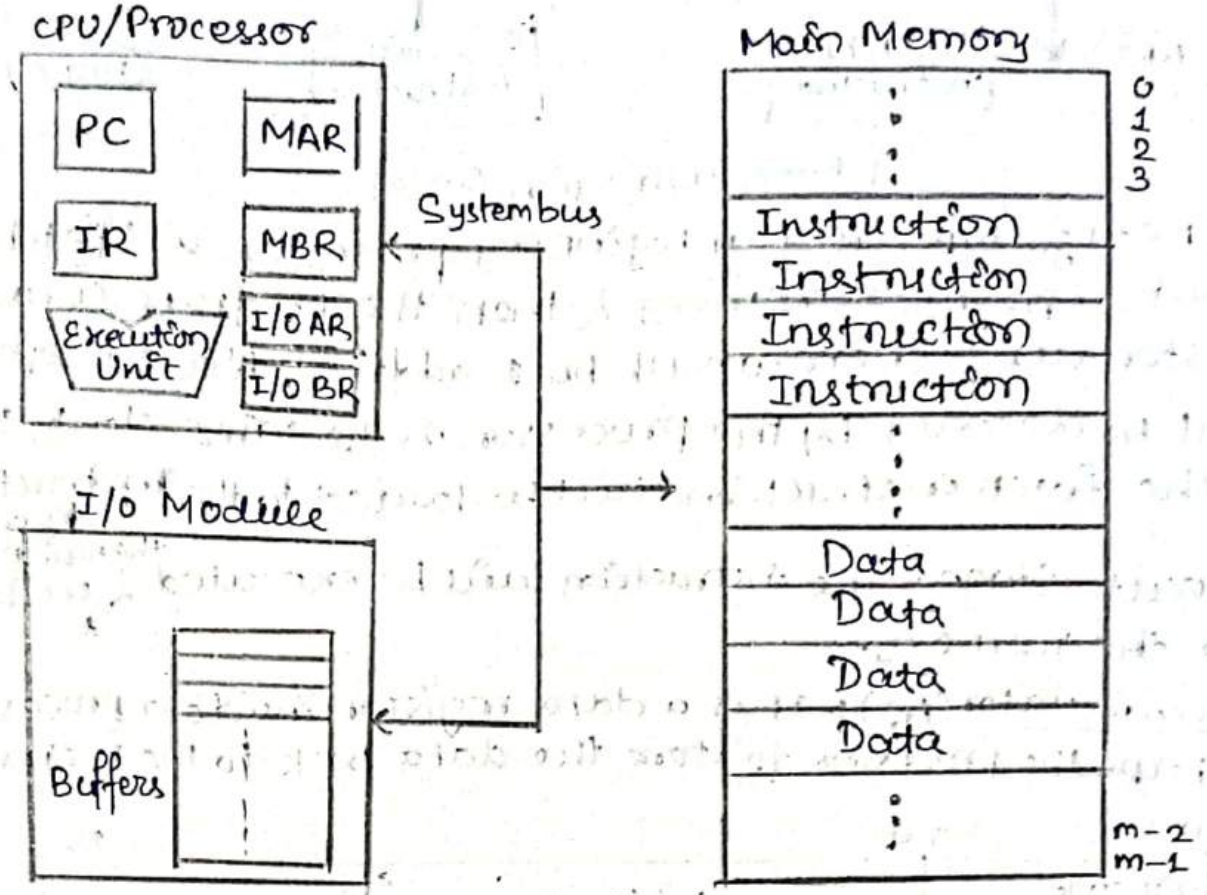
→ Resource is directly under control of Operating System

→ The language understable by the H/w is binary language & the language understable by the user is High Level langu So, to convert high-level language to low-level langua we need a interface is known as OS.

→	DDR2	DDR4
Speed	400-1066 MT/s	1600-4800 MT/s
Power	1.8V	1.2V
capacity	2-4GB	16-32GB
Performance	Slower	Faster
Usage	Old PCs	New PCs

	HDD	SSD
Speed	50-150MB/s	500-7000 M
Durability	Fragile	Durable
Noise	Noisy	Silent
Cost	Cheaper	Costlier
capacity	20 TB	4TB

15/09/25



Computer Component Top-level View

→ At a top level a computer consists of a processor, main memory, & I/O components, & these components are interconnected in some way to a bus called system bus.

(i) Processor controls the operation of computer & performs its data processing function.

(ii) Main memory stores data & program to be executed inside the processor, this is typically volatile in nature.

(iii) I/O Module move the data btw computer & its external environment. The external environment consists of various devices including secondary memory devices, communication equipments & terminals.

Components of ~~Registers~~ Processor

(i) PC is a special type of register which keeps track of address of nxt instruction to be executed inside the processor.

(ii) MAR specifies the address in memory for next read/write.

(iii) MBR contains the data to be written into memory or receives the data from the memory.

(iv) IR register specifies the fetch instruction ready for execution.

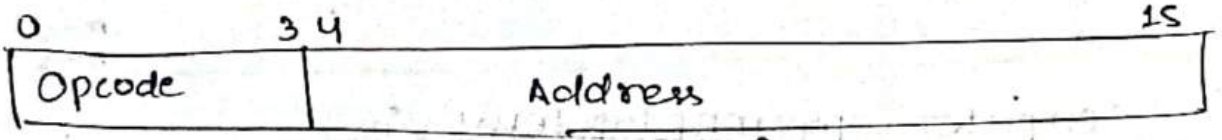
18/09/25

(Basic Instruction Cycle)

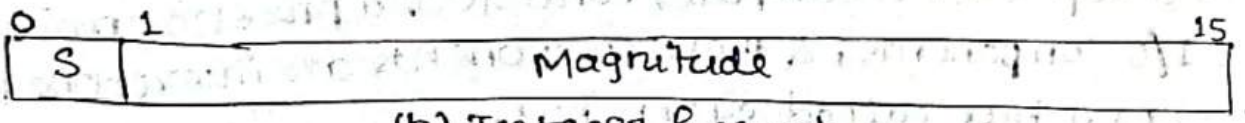
→ Fetch stage typically at beginning processor will fetch the instruction from the memory & then the program counter registor will increament by 1 address. Also the opcode will be decoded by the processor in the same clock cycle. The fetch instruction will be loaded to the Instruction Registor (IR)

→ Execute Stage :- The instruction will be executed & the result will be stored back on the memory.

→ Accumulator (AC) :- It is a data registor inside a processor which helps the processor to store the data back to the memory.



(a) Instruction format



(b) Integer format

Program Counter (PC) = Address of instruction
Instruction Registor (IR) = Inst^r executed
Accumulator (AC) = Tempor storage

1 - 0001 = Load AC from memory
2 - 0010 = Store AC to memory
5 - 0101 = Add the AC from memory

19/09/25

Classes of Interrupts

Program	Result of an instruction execution such as arithmetic overflow, division by zero attempt to execute an illegal instruction.
I/O	Generated by an I/O controller, to signal normal completion of an operation or to signal a variety of error conditions.
Timer	Generated by timer within the processor.
Hardware failure	Due to failure such as power failure.

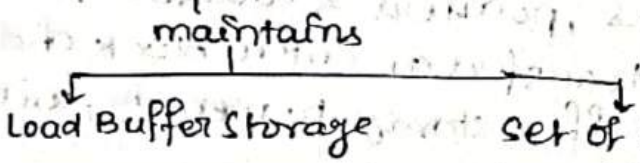
Basic of Operating System (I/O structure)

→ Storage is one of many types of I/O devices within a computer.

→ A large portion of operating system code is dedicated to managing I/O, both because of its importance to the reliability & performance of a system & because of varying natured the devices.

→ A general purpose computer system consists of CPU & multiple device controllers that are connected through a common bus.

→ Each device controller is in charge of a specific type of data

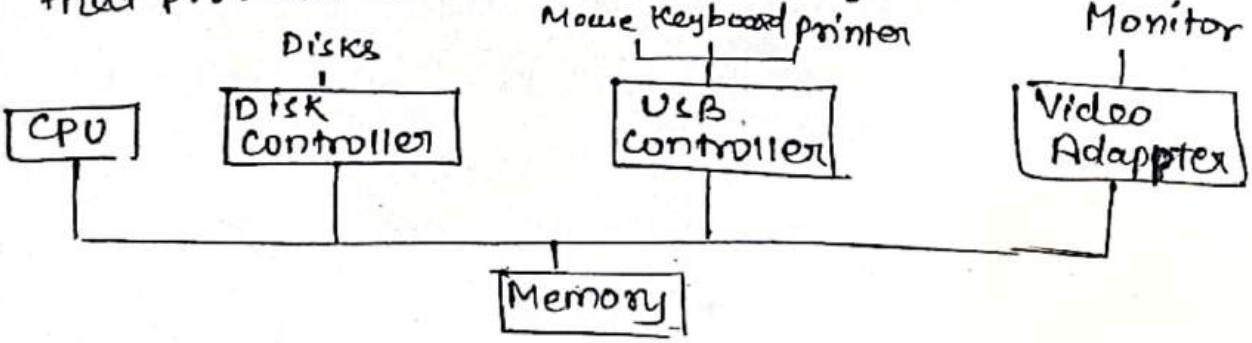


→ Typically, OS have a device driver for each device controller

→ This device driver understands the device controller & presents a uniform interface to the device to the rest of the OS.

- How OS works:-

→ A modern general purpose computer system consists of 1 or more CPUs & no. of device controllers connected to a common bus that provides access to shared memory.



Working of an I/O Operation :- → (Interrupt Driven I/O)

→ To start I/O operation, the device loads the appropriate register within the device controller.

→ The device controller, in turn, examines the contents of those registers to determine what action to take.

→ The controller starts the transfer of data from the device to its local buffer.

→ once transfer of data is complete, the device controller informs the device driver via an interrupt that it has finished its operation.

→ The device driver then returns control to the OS.

Different I/O techniques :-

(i) Programmed Interrupt
(ii) Interrupt Driven Interrupt
(iii) Direct Memory Access (DMA)

→ This form of interrupt-driven I/O is fine for moving small amounts of data but can produce high overhead when used for bulk data movement.

To solve this problem, DMA is used.

→ After setting up buffers, pointers & counters for the I/O device the device controller transfers an entire block of data directly to or from its own buffer storage to memory, with no intervention by the CPU.