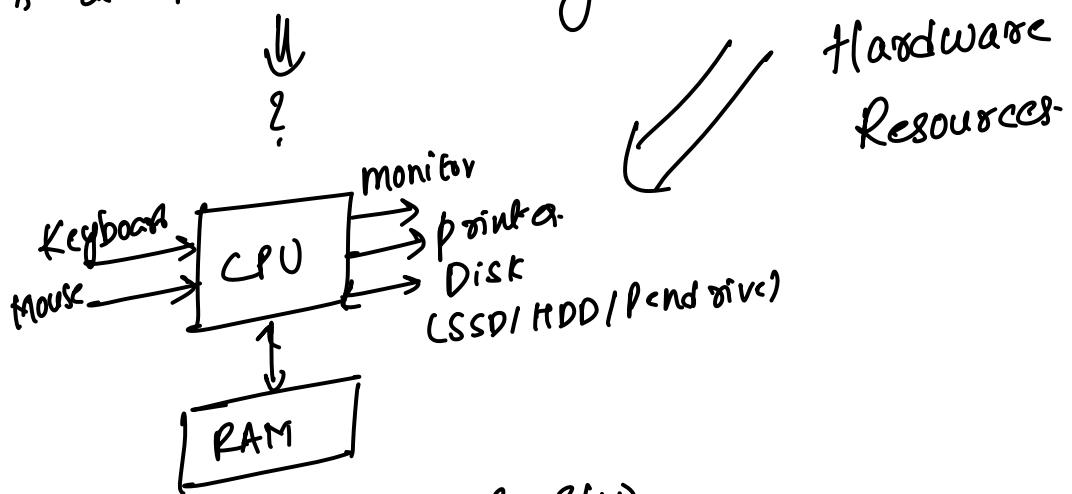


Operating
Systems

P. Sailesh
Kumar

Operating Systems

1) It is a Resource manager.



Resources can be ff(w & S/W)

Make the life easy for S/W engineer

Make the life easy for S/W engineer

(like how pointer works internally)

(like how engineer doesn't need to know)

the S/W engineer doesn't need to know)

* OS can be broadly considered as a resource manager.

* OS can be broadly considered as a resource manager.

Eg Browser runs on our S/W.

Eg Search a file, OS provides this utility, basically

it talks to hardware-

* OS is a piece of software (Most OS's are implemented in C/C++)

A program can be broken into multiple processes

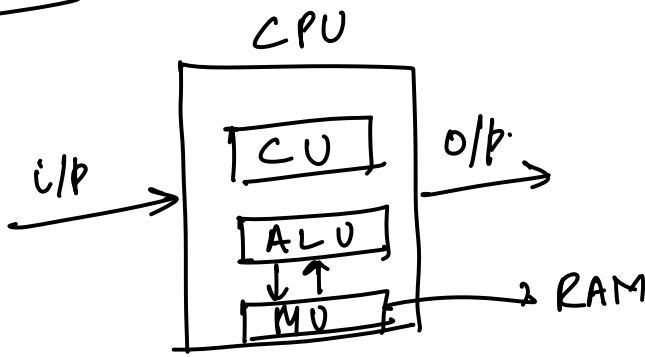
each process uses CPU's resources.

We can see list of processes in our task manager (Windows)

Important topics

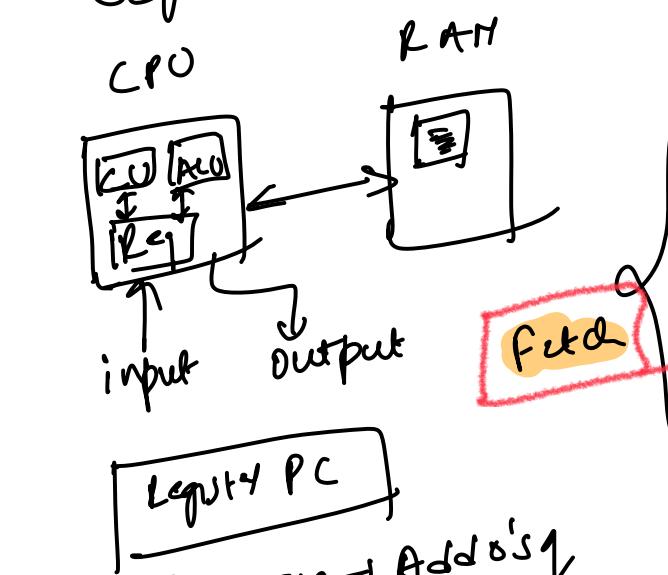
- 1) Process management
- 2) Memory management.
- 3) File Systems & Device management.
- 4) Protection & Security Mechanisms

Let's understand Architecture first



Any program we want to execute, it is stored in RAM.

↓
Sequence of instructions.



PC → Program → Address of current instruction.

MAR → Memory address register

MDR → Memory data register

CR → Current instruction register

Instruction cycle

- 1) Address in PC is copied to MAR
- 2) Increment PC
- 3) Inst. found at MAR is copied to MDR
- 4) Inst. in MDR is copied to CIR
- 5) CU takes over now & decodes the CIR
- 6) CU sends signal to ALU

Execute

RAM
 $(02 : ADD C1) O((21))$
 $\quad\quad\quad (14 \quad 13)$

Take val stored at 120
 (21)
add them & store back
at 120

Interrupts:

main()

{
int i;

int j = 3 + 2;

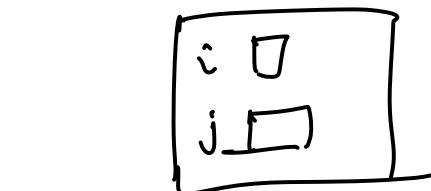
j = j * 2;

scanf("1-d", &i); // keyboard input

i = i / j;

printf("1-d", i);

}



Now CPU is waiting for keyboard & hence it is a interrupt

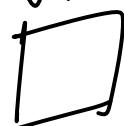
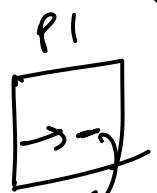
CPU is in idle state

Print to monitor

Uni VS Multi-programming

Multi
Multiple programs executed simultaneously.

Uni → one program executed at a time
huge wastage of resources



Interrupted

CPU is idle now

P2 is not executed

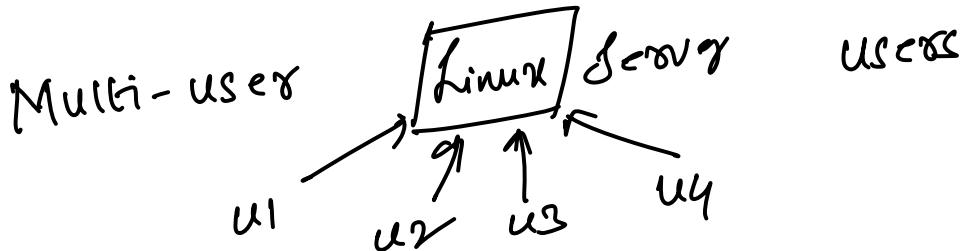
We want P2 to load in RAM when P1 is not in execution

Multi-programming vs Multi-Tasking

(Unix Terminology)

(Windows)
Terminology

At the root both
are same:



Types of multi-programming

1) Preemptive
(forcefully removing)

→ Force a process
to give away control
of resources in CPU

Eg win10, Linux, Android

OS can interrupt the
current program from
running at any given
point of time to let another
program use the CPU

2) Non-preemptive

process has to giveaway control
itself.

↳ completion

↳ I/O event

Eg win3.0 & win3.1

Non-preemptive OS lets a program
run, until it finishes or gives
up voluntarily.

