System calls

- Functions exposed by OS to be used by user to do some task.

m=1

```
fp = fopen("team1.txt", "r");
```

- library function
- System call API (C library)

```
fd = open("team1.txt", O_RDONLY);
{
    - do necessary setting
    - Generate S/W interrupt (Trap)
}
```

Mode - x86
- 00 - Ring 0
- 01 - Ring 1
- 10 - Ring 2
- 11 - Ring 3 - un priviledge

} priviledge

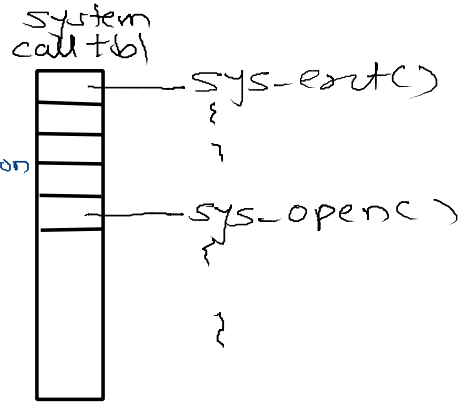
Dual mode protection

m=0

```
swi_handler()
{
```

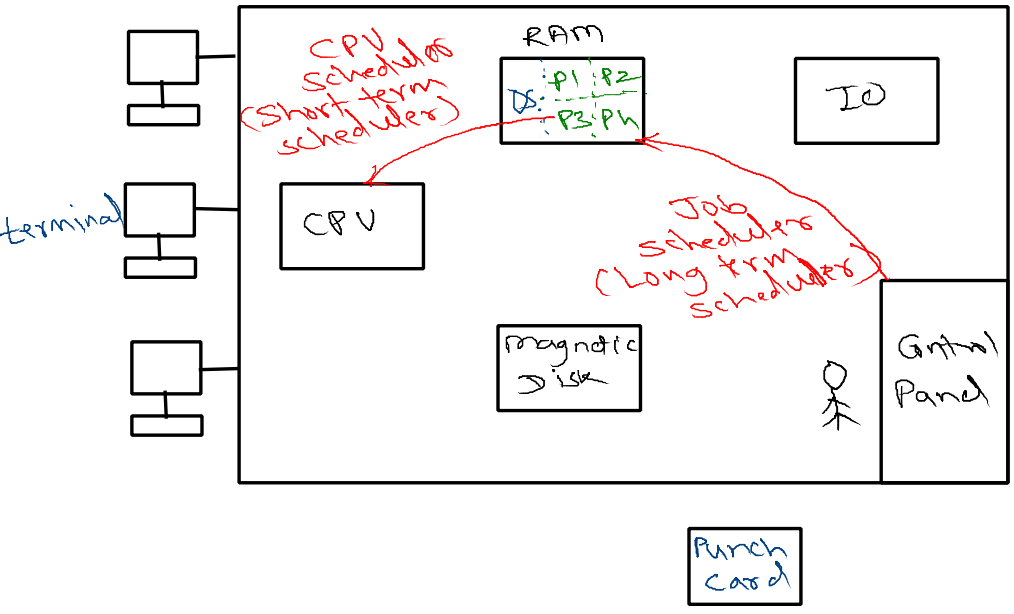
1. Save execution context
2. find ISR from IVT
3. call ISR
4. get system call implementation from system call tbl using system call number
5. call system call impl^n
6. restore execution context

```
}
```



} System call impementations

CPU mode
- 0 - system/kernel/priviledge mode
- 1 - user/unpriviledged mode

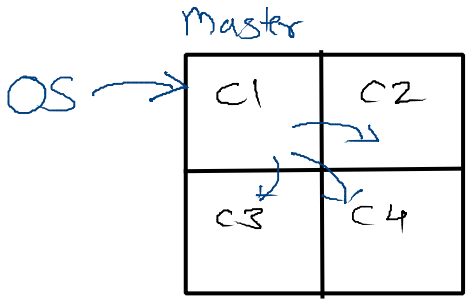


1. Resident Monitor
2. Batch System
3. Multiprogramming System
   - multiple programs are loaded into RAM
   - Degree of multiprogramming
     - no. of programs loaded into RAM
   - CPU time/burst - time spent on CPU
   - IO time/burst - time spent on IO
   - CPU burst > IO burst - CPU bound
   - IO burst > CPU burst - IO bound
   - mixture of CPU bound & IO bound programs are loaded into RAM
4. Time Sharing system / Multitasking systems
   Response time < 1sec
   1. Process based Multitasking
   2. Thread based Multitasking / Multithreading
5. Multi User system
   - multiple users can operate same system

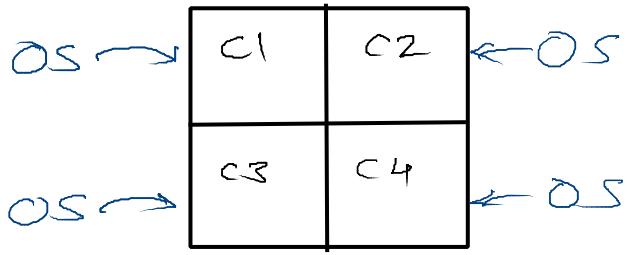
Multiprocessor / Multicore

↳ Multi processing system (Parallel systems)

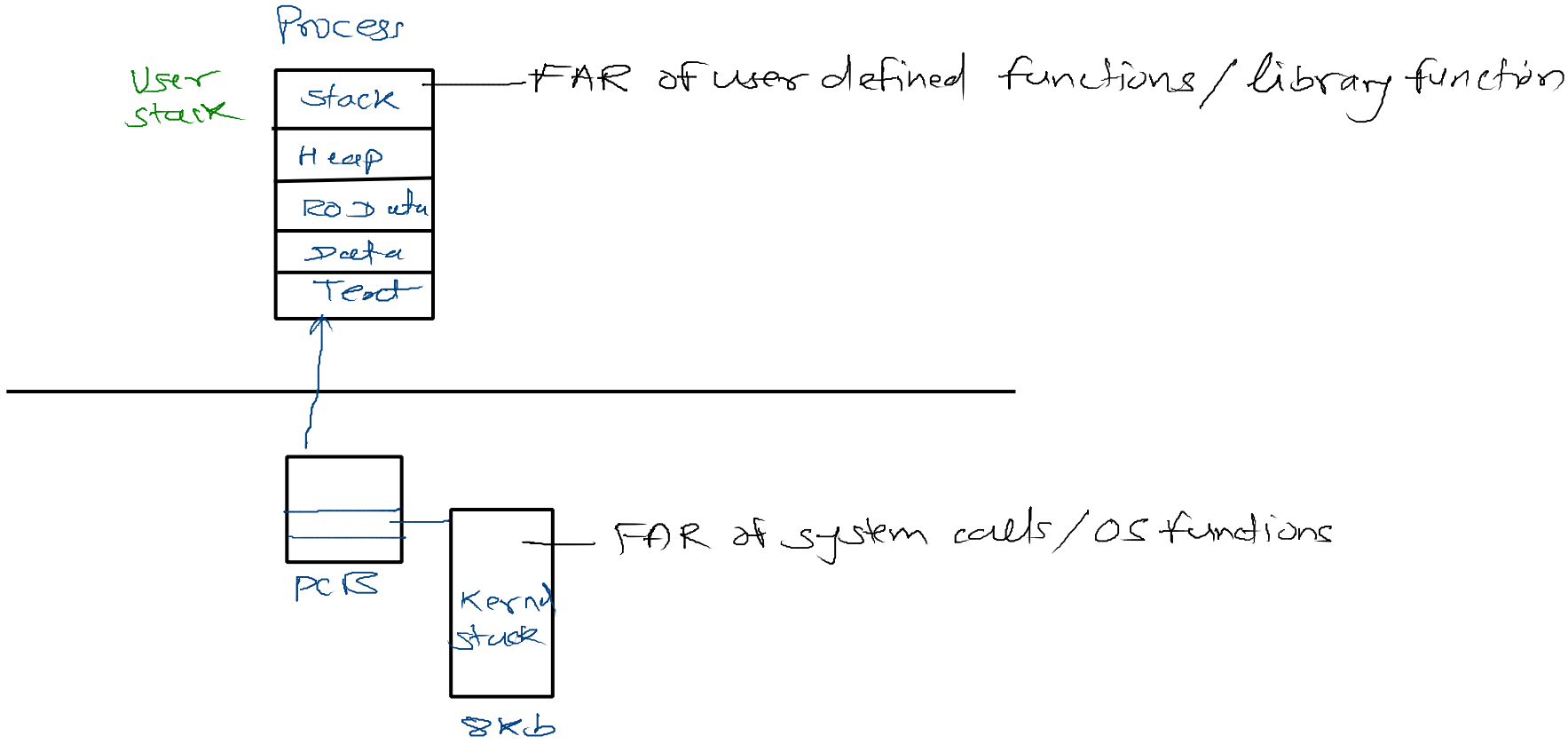
Asymmetric Multiprocessing



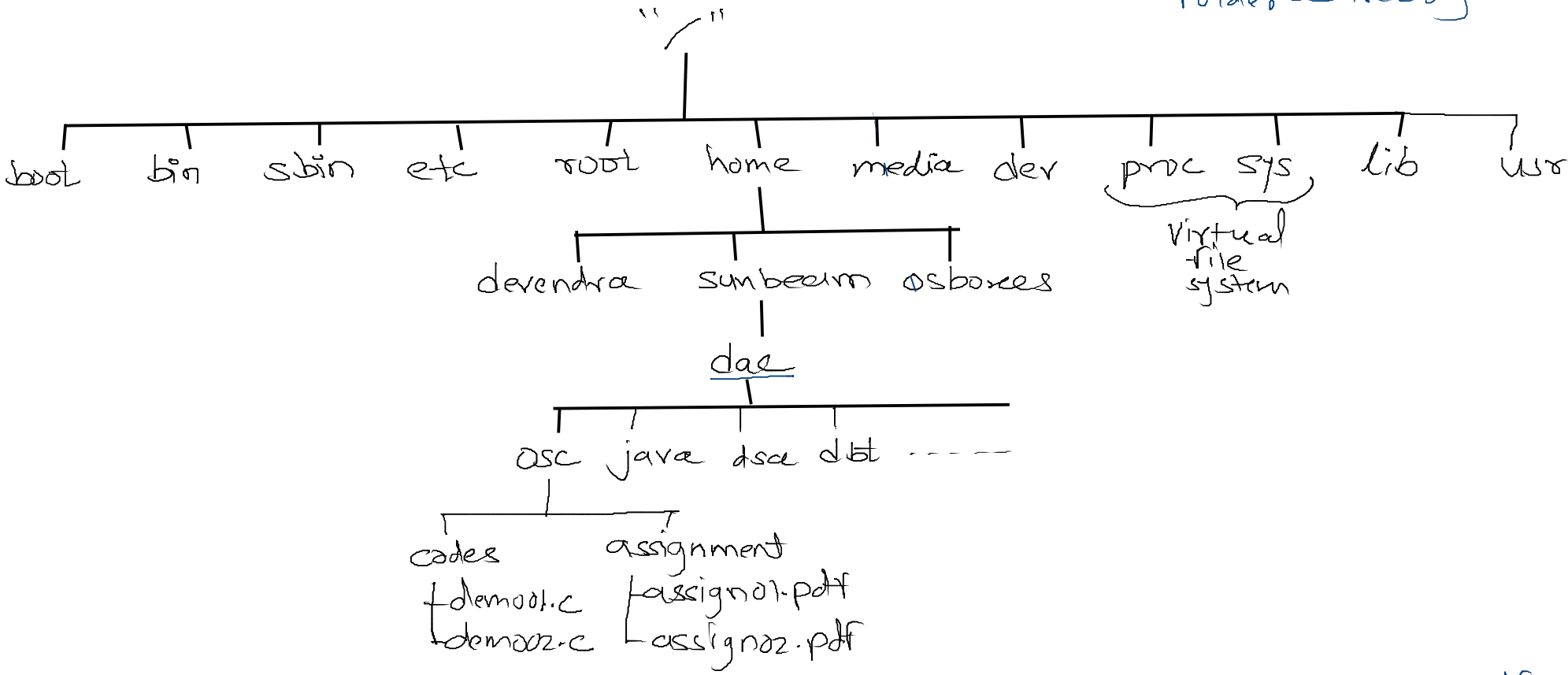
Symmetric Multiprocessing



- Windows Vista
- Linux 2.5 +


Process
User stack
Stack
Heap
RO Data
Data
Text
FAR of user defined functions / library function
PCB
Kernel stack
8Kb
FAR of system calls / OS functions

file - file
Folder - Directory



Absolute path – /home/sunbeam/dac/osc/assignment/assign02.pdf
Realative Path – osc/assignment/assign02.pdf