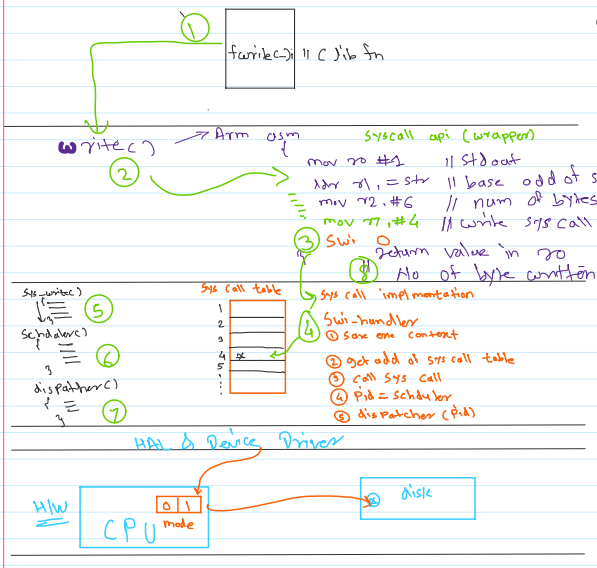
Day-3

18 November 2023 08:43



CPU Modes : 2 (System / Kernel / Privilege ...)

Kernel mode → all instruction can be execute
all register can be accessed

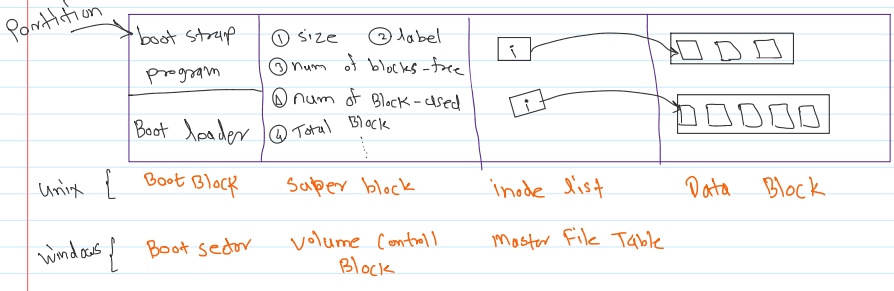
CPU mode : 1
user mode → special instruction are not allowed
(unprivelege) eg. In/out, intr control.....
few regr are not accesible
eg. MMU register, timer, PIC.

→ When user code is running mode = 1 & when kernel code is running mode = 0.
→ mode is changed from 1 to 0, when an intr/exception arrives - mode chang back to 1 when intr/exception handler returns.
→ During sys call, this is done by sw intr
→ Actual CPU mode depends CPU arch.
→ ARm Corton-A : 7 modes
svc, irqn, fiq, undef, abt, system, user
privilege mode

*If You Want To make your Own Sys call

1st
① Download linux kernel source code
② Implement your systemcall in some .C file under kernel source tree (.codes)
③ Create/ modify makefile to compile your system code.
④ make your syscall entry into arch specific syscall table.
⑤ Recompile the linux kernel & deploy it (in boot dir) also modify boot loader (grub) to make entry of your kernel.
⑥ Reboot system into your compiled kernel

2nd
① Implement your sys call appln wrapper using assembly lang or using syscall funct
② Call your sys call api from your code
i.e. main()



file system is created by formating tool / utility
→ windows : format. exe
→ linux : mkfs.