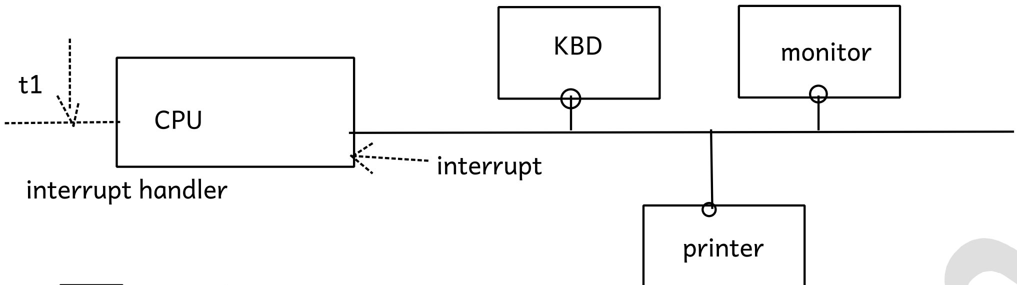
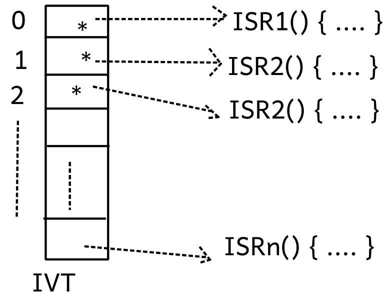
# Interrupt Handling:

+ Interrupt is a signal sent by any i/o device to the CPU, due to which CPU stops execution of current process/job/task and executes ISR associated with that interrupt. Once execution of ISR is finished, suspended task can be resumed.



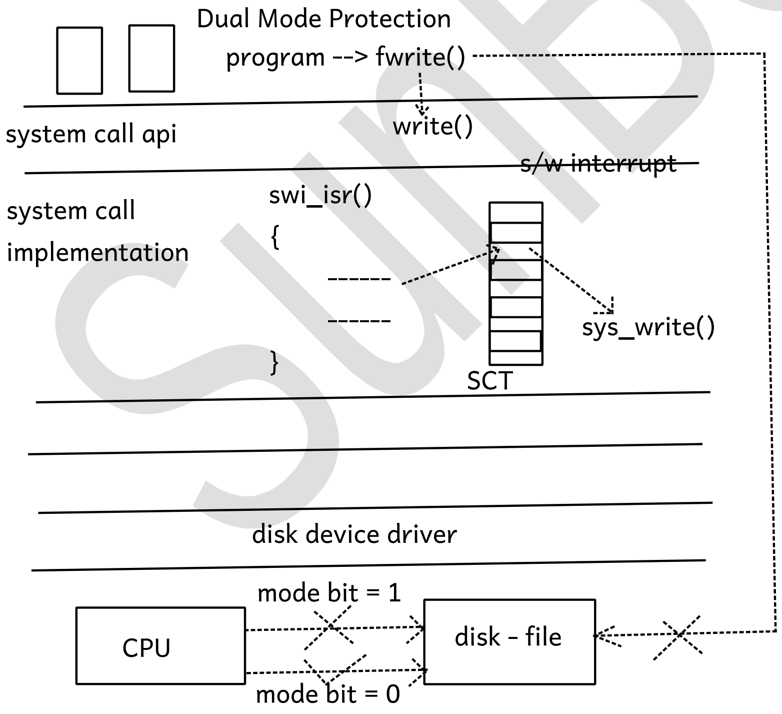


ISR: Interrupt Service Routine

IVT: Interrupt Vector Table:

array of function pointers, each entry contains addr of ISR associated with a device

# Dual Mode Operation/Dual Mode Protection:



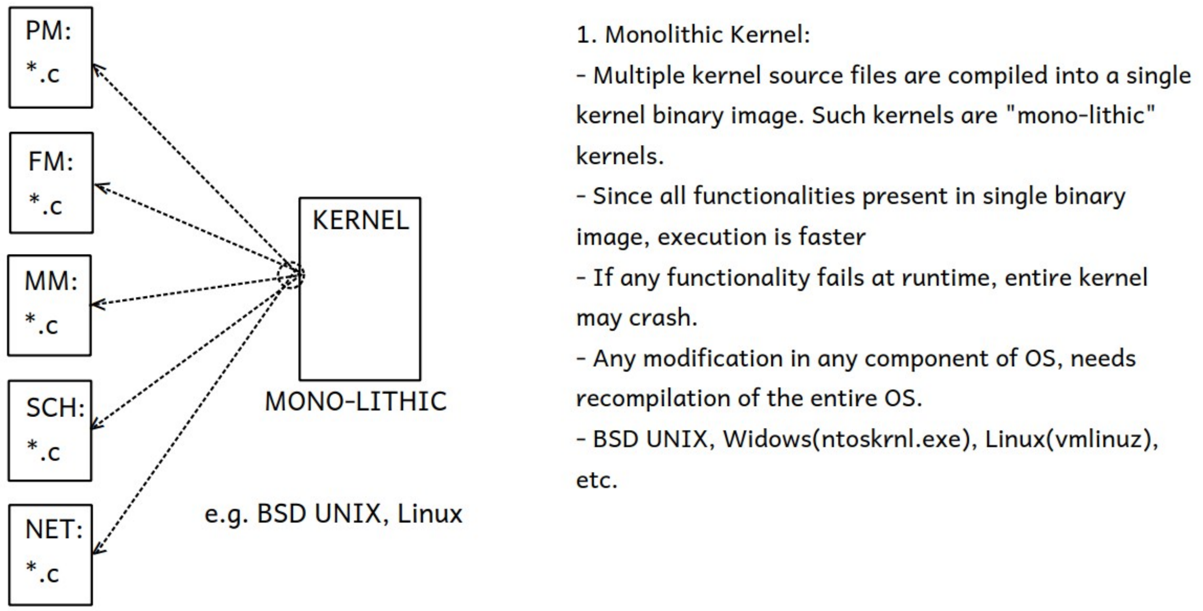
- When the CPU executes user defined code instructions system runs in "user mode", whereas when the CPU executes system code instructions system runs in "system mode", and this is reffered as "dual mode operation".

- system mode is reffered as "kernel mode"/"super visor mode"/"priviledged mode"/"monitor mode"
- user mode is also reffered as "non-priviledged mode".

    mode bit = 0 --> kernel mode/system mode
    mode bit = 1 --> user mode
    - value of mode bit is maintained by an OS/kernel
    - when system call (i.e. a s/w interrupt) occures value mode bit gets clear,
    i.e. value of mode bit will becomes 0.

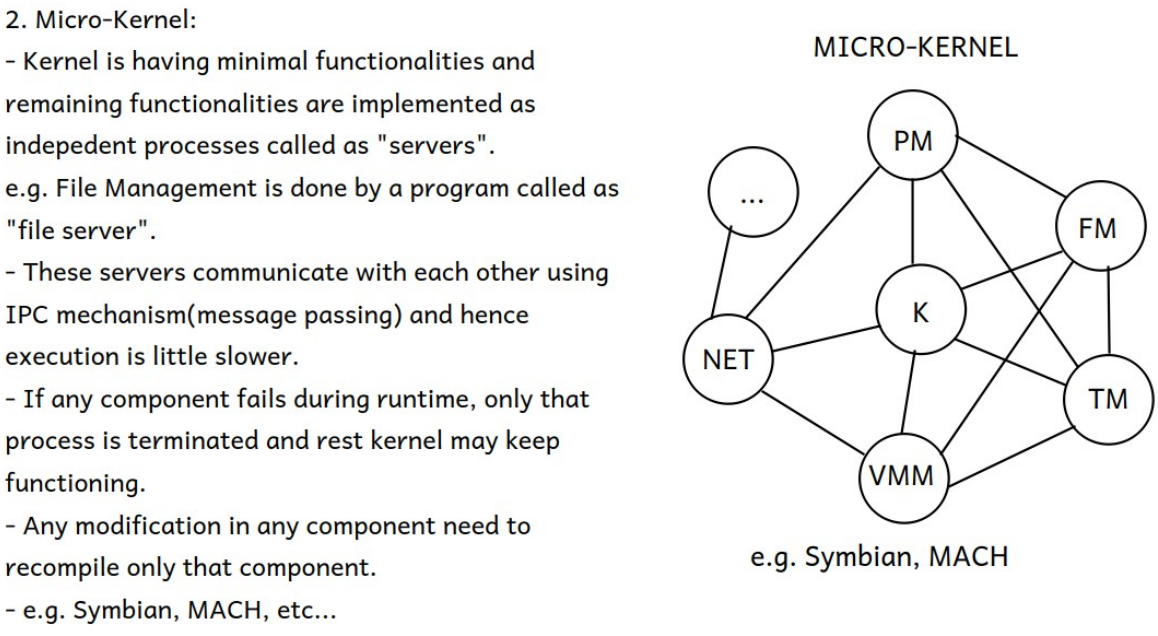
# Types of kernel:

1. Monolithic Kernel:



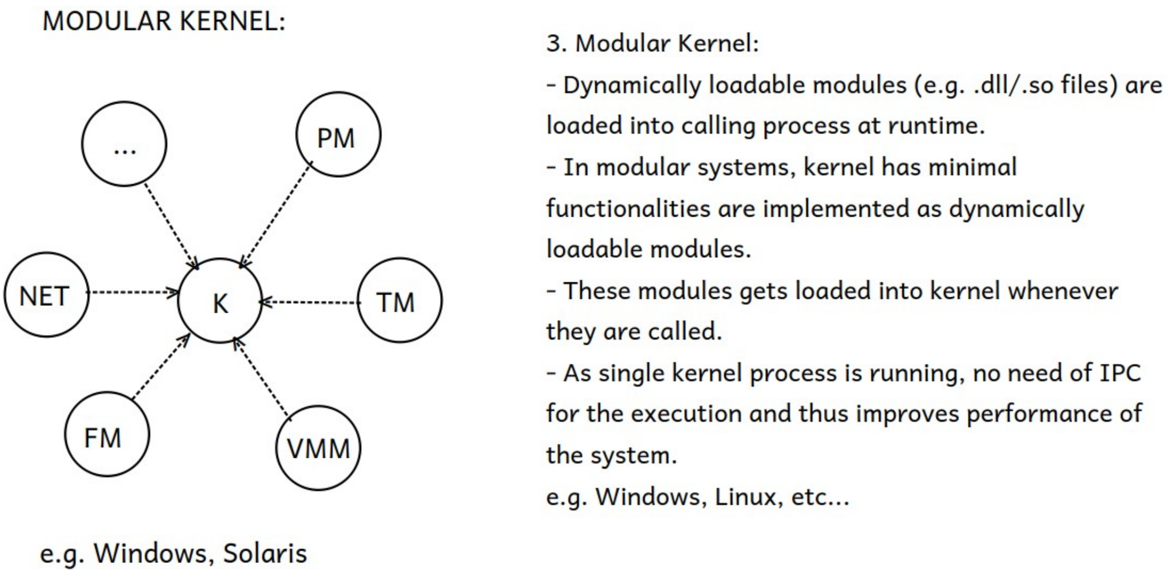
1. Monolithic Kernel:
- Multiple kernel source files are compiled into a single kernel binary image. Such kernels are "mono-lithic" kernels.
- Since all functionalities present in single binary image, execution is faster
- If any functionality fails at runtime, entire kernel may crash.
- Any modification in any component of OS, needs recompilation of the entire OS.
- BSD UNIX, Widows(ntoskrnl.exe), Linux(vmlinuz), etc.

2. Micro Kernel:



2. Micro-Kernel:
- Kernel is having minimal functionalities and remaining functionalities are implemented as indepedent processes called as "servers".
e.g. File Management is done by a program called as "file server".
- These servers communicate with each other using IPC mechanism(message passing) and hence execution is little slower.
- If any component fails during runtime, only that process is terminated and rest kernel may keep functioning.
- Any modification in any component need to recompile only that component.
- e.g. Symbian, MACH, etc...

3. Modular Kernel:



MODULAR KERNEL:

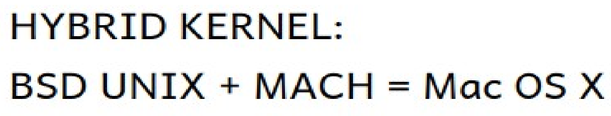
e.g. Windows, Solaris

3. Modular Kernel:

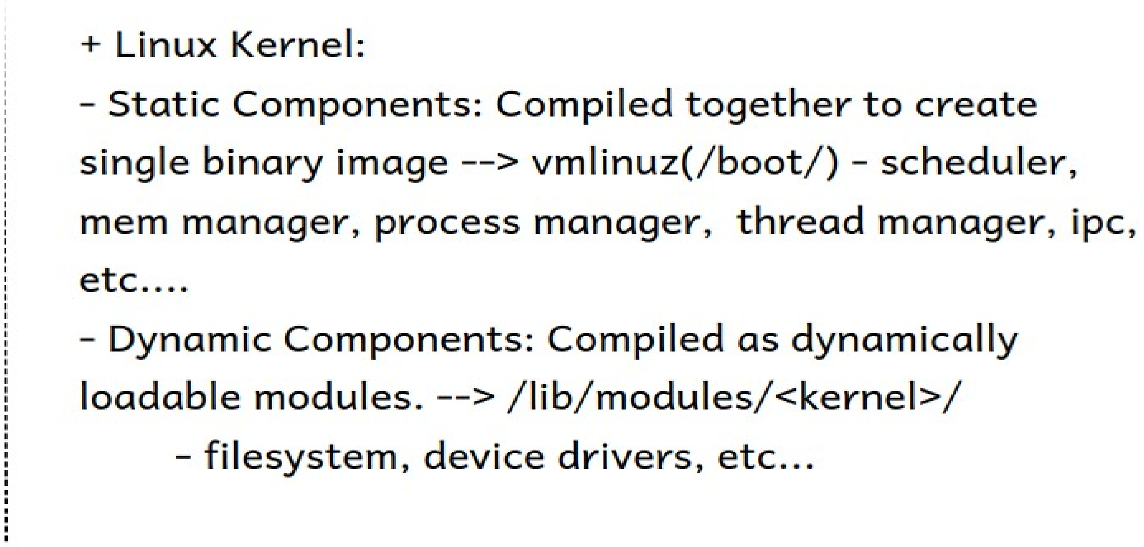
- Dynamically loadable modules (e.g. .dll/.so files) are loaded into calling process at runtime.
- In modular systems, kernel has minimal functionalities are implemented as dynamically loadable modules.
- These modules gets loaded into kernel whenever they are called.
- As single kernel process is running, no need of IPC for the execution and thus improves performance of the system.

e.g. Windows, Linux, etc...

4. Hybrid Kernel:



HYBRID KERNEL:
BSD UNIX + MACH = Mac OS X



+ Linux Kernel:
- Static Components: Compiled together to create single binary image --> vmlinuz(/boot/) - scheduler, mem manager, process manager, thread manager, ipc, etc....
- Dynamic Components: Compiled as dynamically loadable modules. --> /lib/modules/<kernel>/
    - filesystem, device drivers, etc...

+ Features of an OS:
1. multi-programming
2. multi-tasking
3. multi-threading
4. multi-processor
5. multi-user

- When the PCB for a process gets created into the kernel space then we can an execution of a program/process has been started or process has been submitted.

PCB gets created for a process --> process has been submitted/an execution of a program started.

- **"multi-programming":** system in which more than one programs i.e. multiple programs can be submitted at a time, and no. of programs that can be submitted into the system at a time is reffered as **"degree of multi-programming".**

- multi-tasking
- multi-threading
- multi-processor
- multi-user