➢Inter Process Communication

- Processes running into the system can be divided into two categories:

1. **Independent Processes:** - Process which do not shares data (i.e. resources) with any other process referred as an independent process. OR Process which do not affects or not gets affected by any other process referred as an independent process.
2. **Co-operative Processes:** - Process which shares data (i.e. resources) with any other process referred as co - operative process. OR Process which affects or gets affected by any other process referred as co-operative process.

❖Reasons for cooperating processes:

- Information sharing
- Computation speedup
- Modularity
- Convenience
- Cooperating processes need inter process communication (IPC)

Q. Why there is need of an IPC?

As concurrently executing co-operative processes shares common resources, so there are quite chances to occur conflictions between them and to avoid this conflictions there is a need of communication takes place between them.

Q. What is an Inter Process Communication?

An IPC is one of the important **service made available by the kernel, by using which co-operative processes can communicates with each other.**

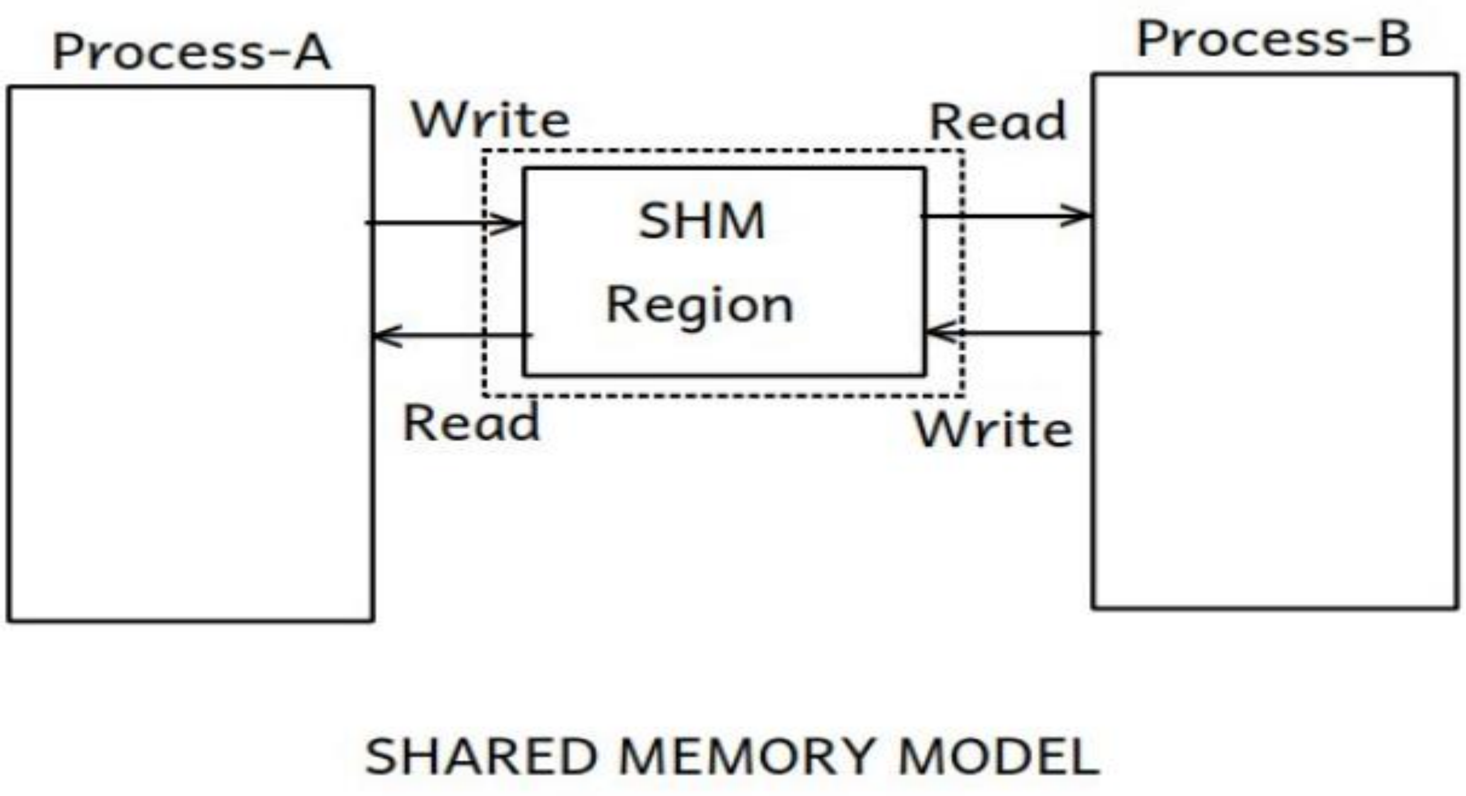
- Inter process communication takes place only **between co-operative processes.**
- Any process cannot directly communicates with any other process, hence there is a need of some medium, and to provide this medium is the job of an OS/Kernel.

❖There are **two techniques** by which IPC can be done/there are two IPC Models:

1. **Shared Memory Model:** under this technique, processes can communicates with each other by means of reading and writing data into the shared memory region (i.e. it is a region/portion of the main memory) which is provided by an OS temporarily on request of processes want to communicates.
2. **Message Passing Model:** under this technique, processes can communicates with each other by means of sending messages.

- Any process cannot directly send message to any other process.
- Shared Memory Model is faster than Message Passing Model

•


Process-A
Process-B
Write
Read
SHM
Region
Read
Write
SHARED MEMORY MODEL

2. Message Passing Model: there are further different IPC techniques under message passing model.

I. **Pipe:** - By using Pipe mechanism one process can send message to another process, vice versa is not possible and hence **it is a unidirectional communication technique**.

- In this IPC mechanism, from one end **i.e. from write end one process can writes data into the pipe, whereas from another end i.e. from read end, another process can read data from it, and communication takes place.**

- There are two types of pipes:

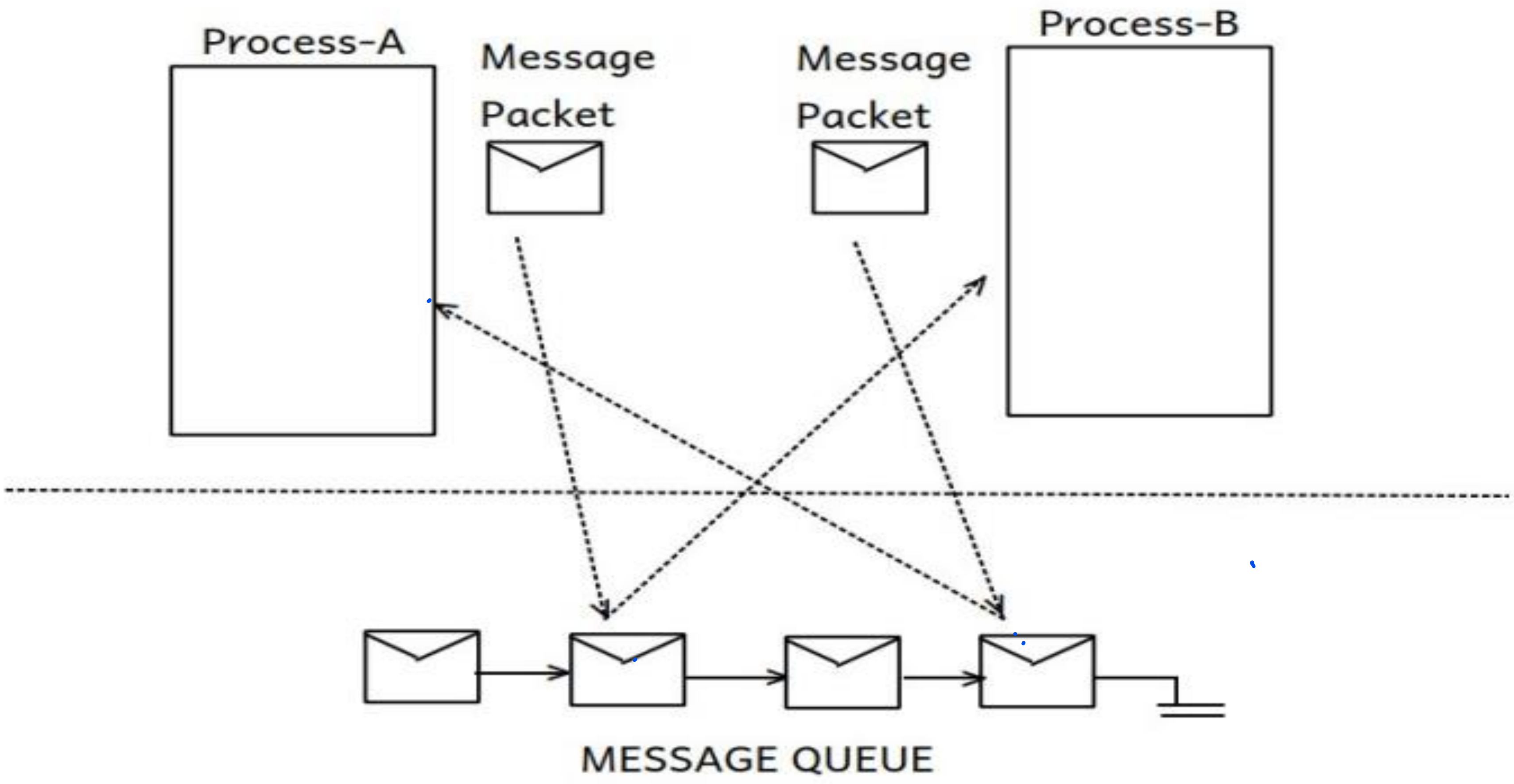
1. **unnamed pipe:** in this type of pipe mechanism, only related processes can communicates by **using pipe (|) command.**
2. **named pipe:** in this type of pipe mechanism, related as well as non-related processes can communicates by **using pipe() system call.**

- By using Pipe only processes which are running in the same system can communicates,

ii. Message Queue:

- By using message queue technique, processes can communicates by means of sending as well as receiving **message packets** to each other via message queue provided by the kernel as a medium, and hence it **is a bidirectional communication.**
- **Message Packet: Message Header(Information about the message) + Actual Message.**
- Internally an OS maintains message queue in which message packets sent by one process are submitted and can be sent to receiver process **and vice-versa**.
- By using message queue technique, only processes which are running in the same system can communicates.

Inter Process Communication



iii. Signals:

- Processes communicates by means of sending signals as well.
- One process can send signal to another process through an OS.
- An OS sends signal to any process but any another process cannot sends signal to an OS.

- **Example**: When we shutdown the system, an OS sends **SIGTERM** signal to all processes, due to which processes gets terminated normally, but few processes can handle **SIGTERM** i.e. even after receiving this signal from an OS they continues execution, to such processes an OS sends **SIGKILL** signal due to which processes gets **terminated forcefully**.
- **e.g. SIGSTOP, SIGCONT, SIGSEGV etc...**

❑Important Signals

- **SIGINT (2):** When CTRL+C is pressed, INT signal is sent to the foreground process.
- **SIGKILL (9)**: During system shutdown, OS send this signal to all processes to forcefully kill them. Process cannot handle this signal.
- **SIGSTOP (19**): Pressing CTRL+S, generate this signal which suspend the foreground process. Process cannot handle this signal.
- **SIGCONT (18):** Pressing CTRL+Q, generate this signal which resume suspended the process
- **SIGSEGV (11):** If process access invalid memory address (dangling pointer), OS send this signal to process causing process to get terminated. It prints error message "Segmentation Fault".

- By using signal ipc technique, only processes which are running in the same system can communicates.

iv. Socket

- Limitation of above **all IPC techniques** is, **only processes which are running on the same system can communicates**, so to overcome this limitation **Socket IPC mechanism** has been designed.
- By using socket IPC mechanism, **process which is running on one machine can communicates with process running on another machine, whereas both machines are at remote distance from each other and provided they connected in a network (either LAN / WAN/Internet).**
- **Socket = IP Address + Port Number.** ← (DCN)

- e.g. chatting application.

➢ Process Coordination / Process Synchronization

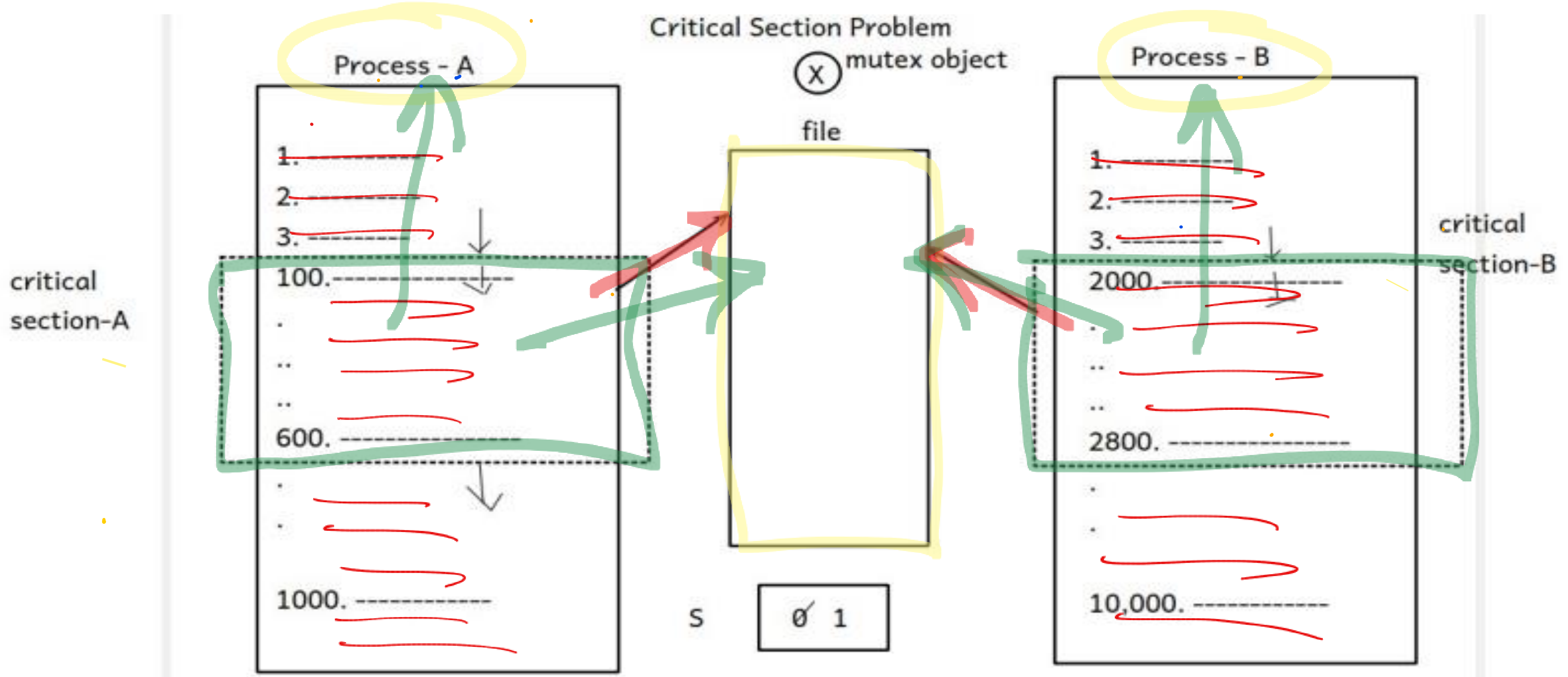
- Why Process Co-ordination/Synchronization?

  - If concurrently executing co-operative processes are accessing common resources, then conflictions may takes place, which may results into the problem of **data inconsistency**, and hence to avoid this problem coordination / synchronization between these processes is required.

  - **Race Condition:** if two or more processes are trying to access same resource at a time, race condition may occurs, and data inconsistency problem may takes place due to race condition.

\- **Race condition** can be avoided by an OS by

1. deciding order of allocation of resource for processes
2. whichever changes did by the last accessed process onto the resource remains final changes.



"data incosistency" problem occure in above case only when both the sections of process A & B are runniing at a same time, and hence these sections are reffered as critical section, and hence data inconsistency problem may occure when two or more processes are running in their critical sections at a same time, and this problem is also reffered as "critical section problem".

➢ Synchronization Tools:

1. **Semaphore:**
2. **Mutex :**

➢ Semaphore:

- Semaphore was suggested by Dijkstra scientist (dutch math)
- Semaphore is a counter

❑ On semaphore two operations are supported:

➢ wait operation: decrement op: P operation:

1. Semaphore count is decremented by 1.
2. If cnt < 0, then calling process is blocked(block the current process).
3. Typically wait operation is performed before accessing the resource.

➢ **signal operation: increment op: V operation:**

1. semaphore count is incremented by 1.
2. if one or more processes are blocked on the semaphore, then wake up one of the process.
3. typically signal operation is performed after releasing the resource.

- **There are Two types of semaphore**

i. Binary semaphore : can be used when at a time resource can be acquired by only one process.

- It is an integer variable having either value is 0 or 1.

ii. Counting / Classic semaphore : can be used when at a time resource can be acquired by more than one processes

Q. If sema count = -n, how many processes are waiting on that semaphore?

Answer: "n" processes waiting

2. Mutex Object:

- Can be used when at a time resource can be acquired by only one process.
- **Mutex object has two states : locked & unlocked, and at a time it can be only in a one state either locked or unlocked.**
- Semaphore uses signaling mechanism, whereas mutex object uses locking and unlocking mechanism

➢Semaphore vs Mutex

S: Semaphore can be decremented by one process and incremented by same or another process.

M: The process locking the mutex is owner of it. Only owner can unlock that mutex.

S: Semaphore can be counting or binary.

M: Mutex is like binary semaphore. Only two states: locked and unlocked.

S: Semaphore can be used for counting, mutual exclusion or as a flag.

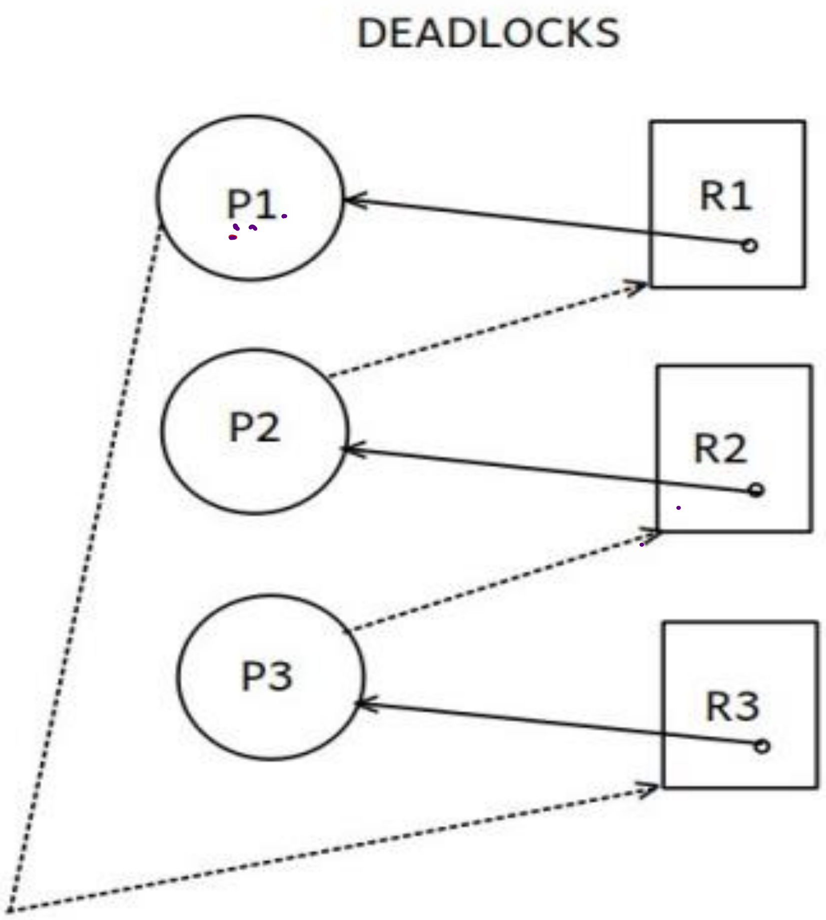
M: Mutex can be used only for mutual exclusion.

➢Deadlock:

- **There are four necessary and sufficient conditions to occur deadlock / characteristics of deadlock:**

1. **Mutual Exclusion:** at a time resource can be acquired by only one process.
2. **No Preemption:** control of the resource cannot be taken away forcefully from any process.
3. **Hold & Wait:** every process is holding one resource and waiting for the resource which is held by another process.
4. **Circular Wait:** if process P1 is holding resource and waiting for the resource held by another process P2, and process P2 is also holding one resource and waiting for the resource held by process P1.

Deadlock: Resource Allocation Graph



Three deadlock handling methods are there:

1. **Deadlock Prevention:** deadlock can be prevented by discarding any one condition out of four necessary and sufficient conditions.
2. **Deadlock Detection & Avoidance:** before allocating resources for processes all input can be given to deadlock detection algorithm in advanced and if there are chances to occur deadlock then it can be avoided by doing necessary changes.

❖There are two deadlock detection & avoidance algorithms:

1. **Resource Allocation Graph Algorithm :**
   - OS maintains graph of resources and processes.
   - A cycle in graph indicate circular wait will occur. In this case OS can deny a resource to a process.
2. **Banker's Algorithm :**
   - A bank always manage its cash so that they can satisfy all customers.

3. Deadlock Recovery:

- System can be recovered from the deadlock by two ways:

1. **Process termination:** in this method randomly any one process out of processes causes deadlock gets selected and terminated forcefully to recover system from deadlock.
   - Process which gets terminated forcefully in this method is referred as **a victim process**.

2. **Resource preemption:** in this method **control of the resource taken away forcefully from a process** to recover system from deadlock.

❑Starvation:

- The process not getting enough CPU time for its execution.
- Process is in ready state/queue. Reason: Lower priority (CPU is busy in executing high priority process).

❑Deadlock:

- The process not getting the resource for its execution.
- Process is in waiting state/queue indefinitely. Reason: Resource is blocked by another process (and there is circular wait).

➢ System Calls

- **System Calls** are the functions defined in a C, C++ & Assembly languages, which provides interface of services made available by the kernel for the user (programmer user).
- If programmers want to use kernel services in their programs, it can be called directly through system calls or indirectly through set of library functions provided by that programming language.
- In Unix OS has 64 system calls.
- In Linux has 300+ system calls.
- **There are 6 categories of system calls:**

1. **Process Control System Calls**: e.g. fork(), _exit(), wait() etc...
   1. fork() : To create new processes
      - Creates a new process duplicating the calling process.
      - The new process is referred as child process, while the calling process is referred as parent process
   2. _exit() : To exit processes
   3. wait() : To hold/wait processes
   4. exec() : To load a new program (executable file) into current process's memory space.

2. **File Operations System Calls:** e.g. open(), read(), write(), close() etc...
    1. open() :
    2. read() :
    3. write() :
    4. close() :
3. **Device Control System Calls:** e.g. open(), read(), write(), ioctl() etc...
4. **Accounting Information System Calls**: e.g. getpid(), getppid(), stat() etc...
    1. **getpid() :** To get Process ID
    2. **getppid() :** To get Parent Process ID
    3. **stat()** : To get File Information
5. **Protection & Security System Calls**: e.g. chmod(), chown() etc..
    1. **chmod() :** Change user mode / permission
    2. **chown()** : get file owner info
6. **Inter Process Communication System Calls:** e.g. pipe(), signal(), msgget() etc...

Memory Management

❖Memory holds (digital) data or information.

-Bit = Binary Digit (0 or 1) => Internally it is an electronic circuit i.e. FlipFlop

-1 Byte = 8 Bits

-B, KB (2^10), MB (2^20), GB (2^30), TB (2^40), PB (2^50), XB (2^60), ZB (2^70)

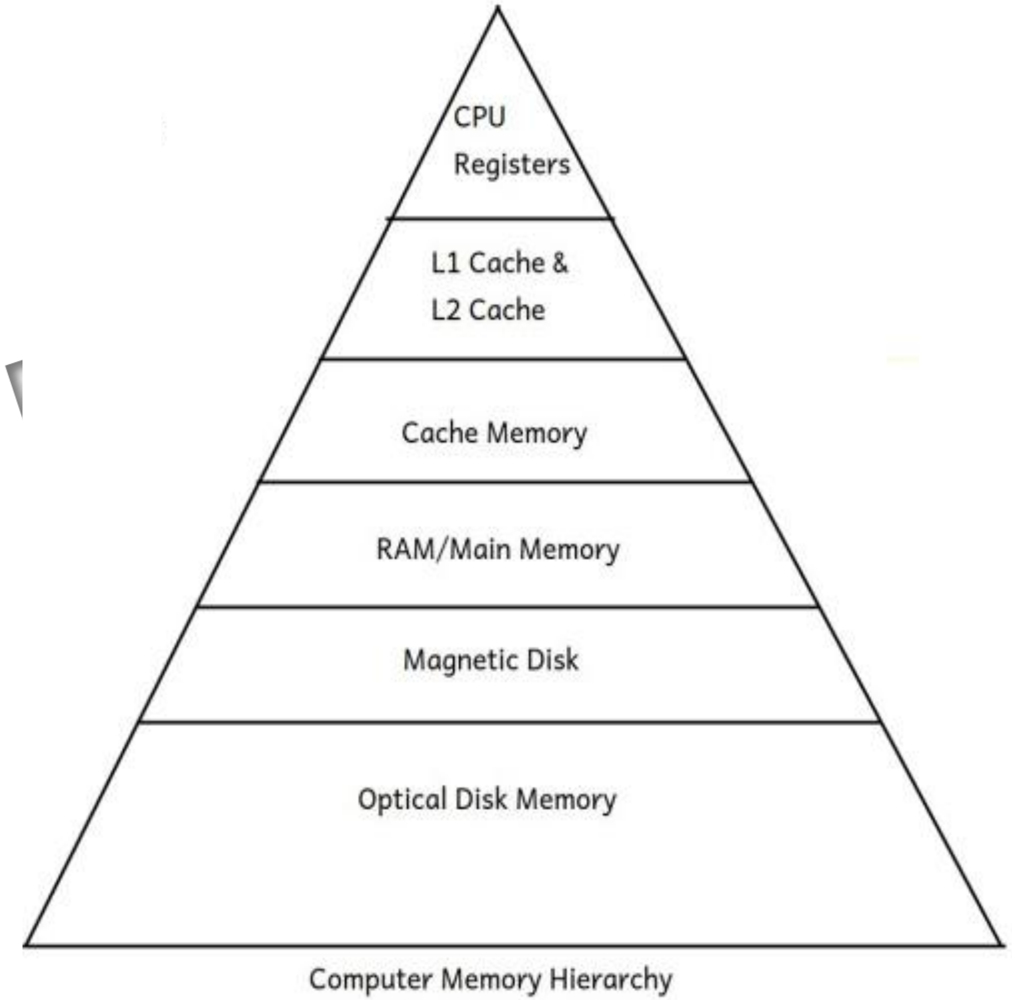
➢Memory Technologies

- ❖ There are four methods by which data can be accessed from the computer memory:

1. **Sequential Access**: e.g. Magnetic Tape
2. **Direct Access**: e.g. Magnetic Disk
3. **Random Access**: e.g. RAM Memory
4. **Associative Access**: e.g. Cache Memory

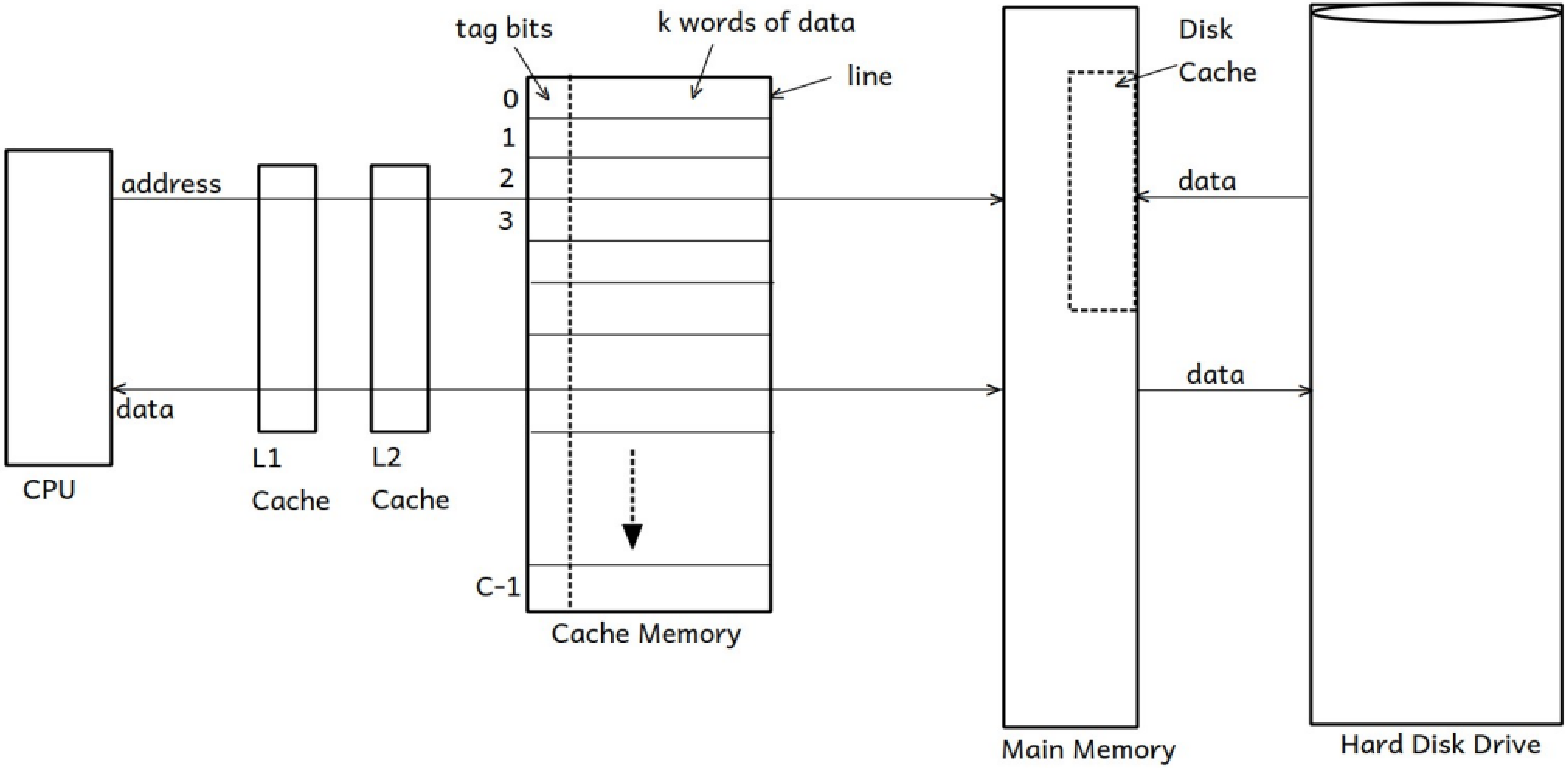
➢Computer Memory Technologies

As we go Top to Bottom
1.Acses speed decrease
2.Coast also decrease
3.Capacity increase



❖Memory Access

- CPU <----> Memory
- **Bus**
    - set of wire to use connection / data transfer.
- **Address bus**
    - Unidirectional from CPU to the memory
    - Address represent location of the memory to read/write
    - Number of lines = number of locations
- **Data bus**
    - Bi-directional from/to CPU to/from the memory
    - Carries the data
    - Number of lines = width of data unit
- **Control bus**
    - Read/Write operation
- **CPU <---> Cache <---> RAM <---> Disk**


tag bits
k words of data
line
0
1
2
3
C-1
address
data
CPU
L1
Cache
L2
Cache
Cache Memory
Disk
Cache
data
data
Main Memory
Hard Disk Drive

- Computer memory can be categorized into two categories as per its **location:**
  - **Internal Memory & External Memory.**
    - **Internal Memory**: memory which is internal to the motherboard is referred as an internal memory.
      - e.g. CPU registers, L1 & L2 cache Cache memory, RAM.
    - **External Memory:** memory which is external to the motherboard is referred as an external memory.
      - e.g. Magnetic disk, Optical disk, magnetic tape , Pen Drive etc...
  - **Volatile vs Non-volatile memory**

    - **Volatile memory**: The contents of memory are lost when power is OFF.

    - **Non-volatile memory**: The contents of memory are retained even after power is OFF.

- Computer memory can also categorized into two categories: **Primary Memory & Secondary Memory**.

1. **Primary Memory:** memory which can be accessible directly by the CPU is referred as primary memory, i.e. memory which can accessible by the CPU with the help of instruction set having with the CPU.
   - e.g. CPU registers, L1 & L2 Cache, Cache Memory, RAM
2. **Secondary Memory:** memory which cannot be accessed directly by the CPU is referred as secondary memory.
   - e.g. Magnetic Disk, CD/DVD, PD etc..
   - If the CPU want to access disk contents, first it gets fetched into the RAM and then it can be accessed by the CPU from RAM.

- As for an execution of every program RAM memory is must and hence **RAM is also called as Main memory.**

Computer Memory Technologies:

- **CPU Registers:**
  - memory which is very close to the CPU are registers which is at the top in a computer memory hierarchy.
  - Instructions and data currently executing by the CPU can be kept temporarily into the CPU registers.

Why there is a need of cache memory?

- As the rate at which the CPU can execute instructions is faster than the rate at which data can be accessed from the main memory, so even the CPU is very fast, with the same speed data do not gets fetched from the main memory for execution, hence due to this speed mismatch overall system performance gets down.
- To reduce speed match between the CPU and the main memory Cache memory (hardware) can be added between them and system performance can be increases by means **reducing speed mismatch.**

➢What is Cache Memory ?

- Cache memory is faster memory, which is a type RAM i.e. SRAM, in which most recently accessed main memory contents can be kept/stored in an associative manner i.e. in a key-value pairs.
- **There are two types of RAM:**

1. **DRAM (Dynamic RAM):** memory cells are made up of capacitors and transistor.
   - Main memory is as example of DRAM.
2. **SRAM (Static RAM):** memory cells are made up of transistor.
   - Cache Memory is an example of SRAM

- **Cache Memory** has C no. of lines, whereas each line is divided into two parts, each line contains k words of data (recently accessed main memory contents) and its main memory addresses can be kept in few tag bits.

1. **First part of a line** : few tag bits contains main memory addresses of k words of data in that line
2. **Second part :** of a line contains k words of data.

- When the CPU want to fetch data from the main memory it requests for its address, and this requested address gets searched into the cache memory first, if requested addr is found in the cache memory then data also found in a cache memory, it is referred as **cache hit**, whereas if the requested address and hence data is not found in a cache memory then it is referred as a **cache miss,** in that data gets fetched from main memory and gets transferred to the CPU via cache memory only.

- Even after adding cache memory between the CPU and main memory, the rate at which the CPU can execute instructions is faster than the rate at which at which data can be accessed from cache memory, and hence to reduce speed mismatch between the CPU and cache memory one or more levels of cache memory i.e. L1 cache & L2 cache can be added between them.
- **Disk Cache:** it is purely a software technique in which portion of the main memory can be used as a cache memory in which most recently accessed disk contents can be kept in an associative manner, so whenever the CPU want to access data from hard disk drive it first gets searched into the disk cache.
- Disk cache technique is used to reduce speed mismatch between the CPU and Secondary memory

➢ROM (Read Only Memory)

- Read-only memory (Not writable).
- This type of memory is non-volatile.
- The information is stored permanently.
- Programs (executable instructions) stored in ROM are called -- Firmware. e.g. BIOS, Bootstrap loader, POST, ...

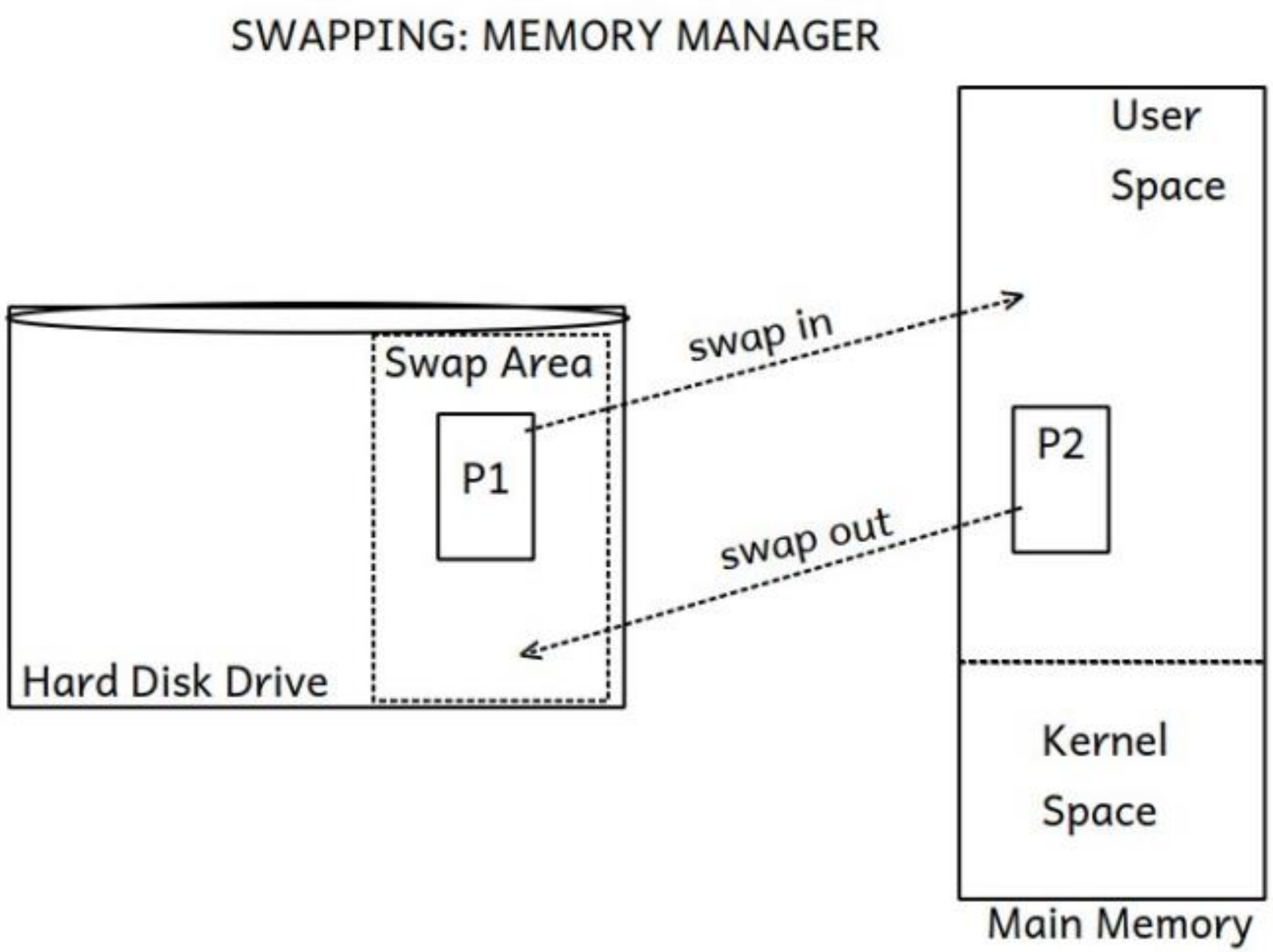
• Types of ROM

- **Masked ROM (MROM)** -- contents are fixed while manufacturing
- **Programmable ROM (PROM)** -- one time writable
- **Erasable Programmable ROM (EPROM)** -- written multiple times with special circuit
  - **Ultra-Violet EPROM (UV-EPROM)** -- all contents erased using UV rays
  - **Electrical EPROM (E-EPROM)** -- erase selected bytes using high electric current
- **Flash (like E-EPROM)** -- erase selected blocks - high speed

❖Virtual memory / Swap area

➢ **Swap area**: it is a portion of the hard disk drive (keep reserved while installation of an OS) can be used by an OS as an **extension of the main memory** in which **inactive running programs can be kept temporarily** and as per request processes can be **swapped in** and **swapped out** between swap area and the main memory by system program named as memory manager.

- In Linux swap area can be maintained in the form of swap partition, whereas in Windows swap area can be maintained in the form of swap files.
- **Conventionally size of the swap area should be doubles the size of the main memory,** i.e. if the size of main memory is 2 GB then size of swap area should be 4 GB, if the size of main memory is 4 GB then size of swap area should be 8 GB and so on.
- **Virtual memory advantages:**
  - Can execute more number of programs.
  - Can execute bigger sized programs.


SWAPPING: MEMORY MANAGER
User
Space
Swap Area
P1
swap in
P2
swap out
Hard Disk Drive
Kernel
Space
Main Memory

➢ Swapping:

- Swapping done by the system program of an OS named as **Memory Manager**, it swap ins active running programs into the main memory from swap area and swap outs inactive running programs from the main memory and keep them temporarily into the swap area.
- There are two variants of swapping: **swap in & swap out.**

Thank you!

Kiran Jaybhave

email – kiran.jaybhave@sunbeaminfo.com