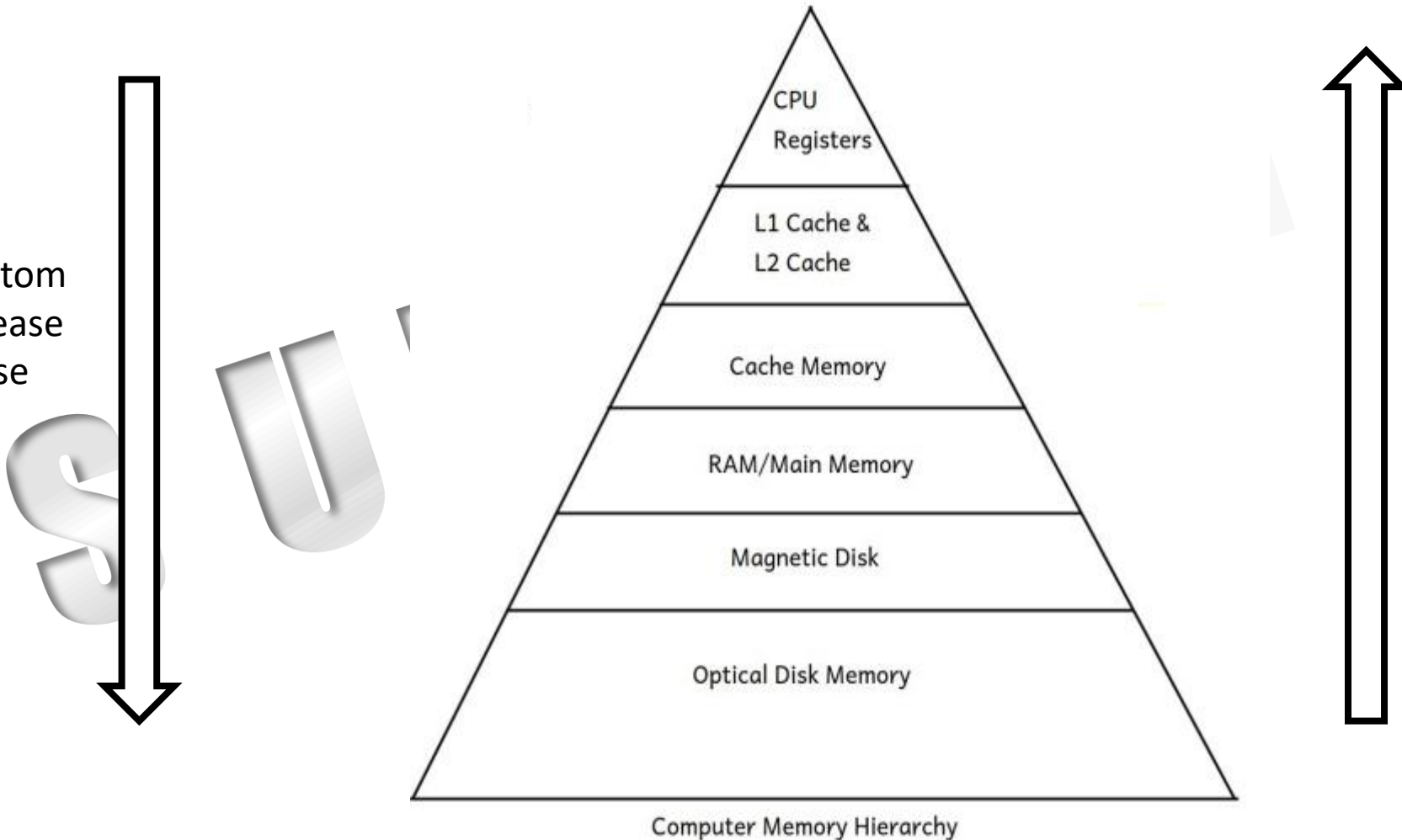# Memory Management

Computer Fundamentals and Operating Systems

➢ Computer Memory Technologies

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

CPU
Registers

L1 Cache &
L2 Cache

Cache Memory

RAM/Main Memory

Magnetic Disk

Optical Disk Memory

Computer Memory Hierarchy

Computer Fundamentals and Operating Systems

➢ 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.

- MAR, MBR, IOAR, IOBR, PC, SP, Accumulator etc...

➢ 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 etc...

Computer Fundamentals and Operating Systems

➢ Computer Memory Technologies:

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

   Secondary Memory.

o **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

o **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.

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

➢ Why there is a need of cache memory?

o 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.

o 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 ?

o 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.

o There are two types of RAM:

1. **DRAM (Dynamic RAM):** memory cells are made up of capacitors

   - Main memory is as example of DRAM.

2. **SRAM (Static RAM):** memory cells are made up of capacitor and transistor.

   - Cache Memory is an example of SRAM

Computer Fundamentals and Operating Systems

o **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.

o 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.

o 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.

o **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.

o Disk cache technique is used to reduce speed mismatch between the CPU and Secondary memory

# Computer Fundamentals and Operating Systems

Operating Systems Concepts
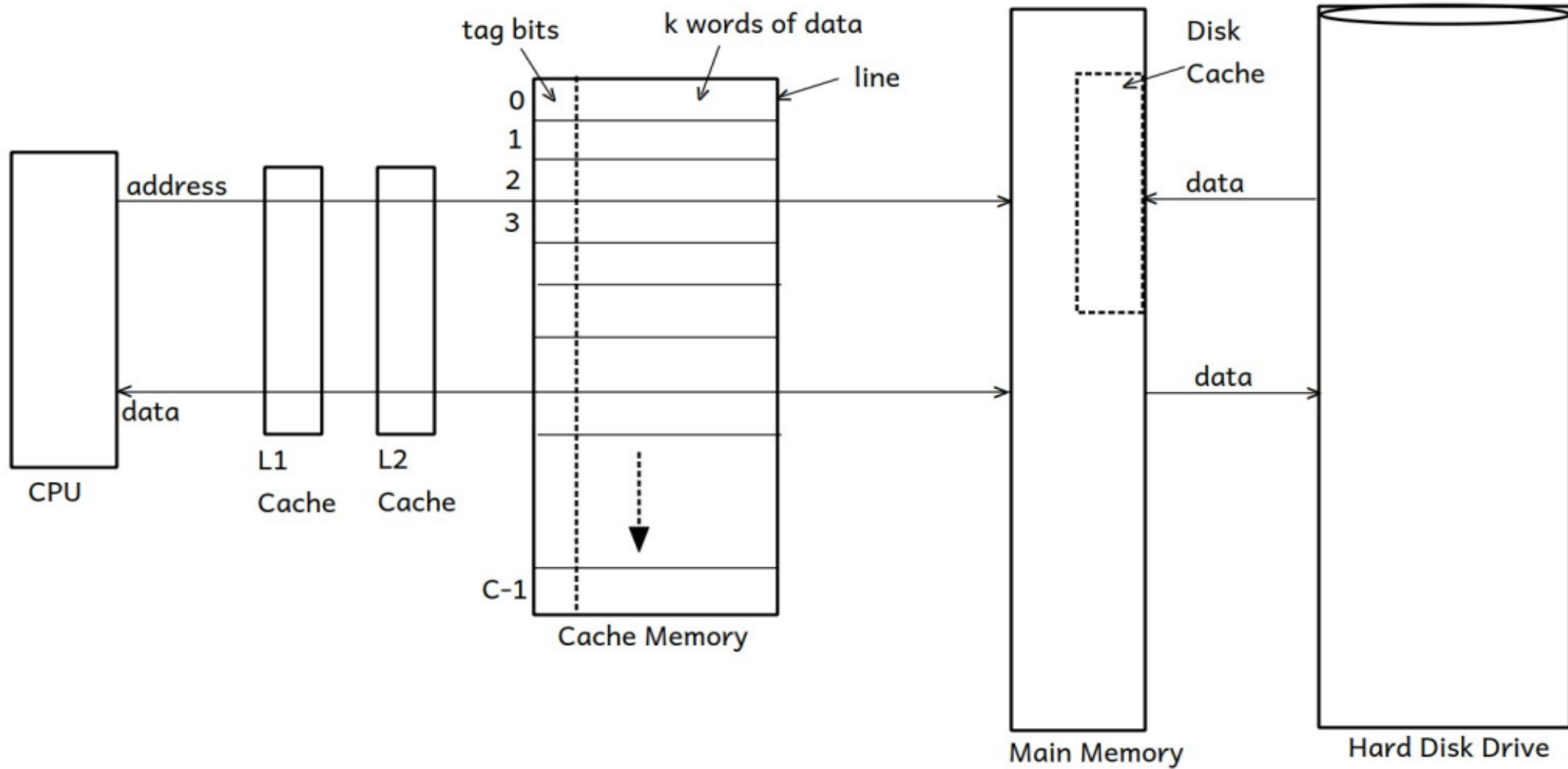
➤ Memory Management

❖. Why there is a need of memory (main memory)management ?

o As main memory is must for an execution of any program and it is a limited memory, hence an OS manages main memory.

o To achieve maximum CPU utilization, an OS must support **multitasking, and to support multi-tasking multiple processes** must be submitted into the system at a time i.e. it must support multiprogramming, but **as main memory is limited** to support multiprogramming an **OS has to do memory management to complete an execution of all submitted processes.**

o **Memory space of one process should gets protected from another process**.

Operating Systems Concepts

SWAPPING: MEMORY MANAGER

Operating Systems Concepts
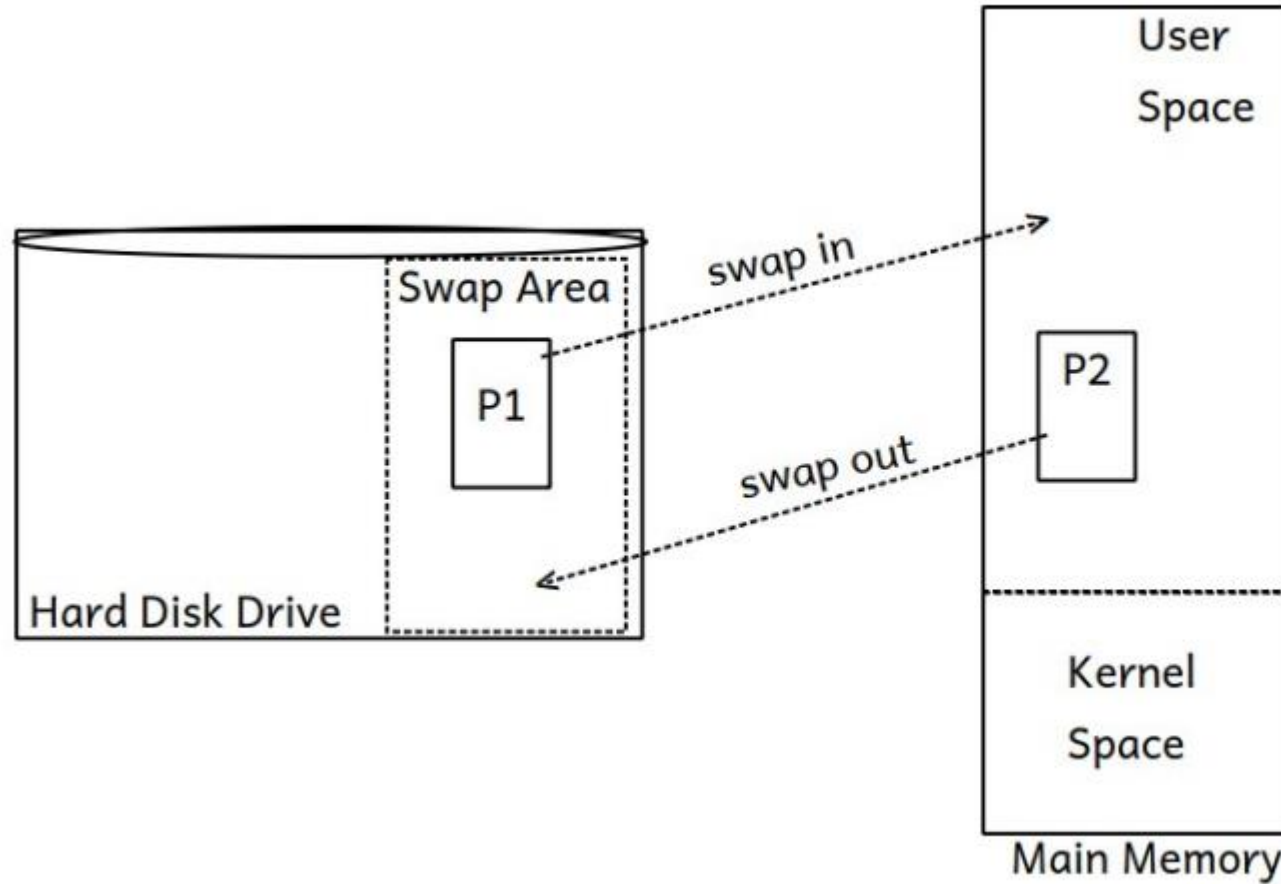
➢ Swapping:

o **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.

o 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.

o **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.

➤ Swapping:

o 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.

o There are two variants of swapping: **swap in & swap out.**

Operating Systems Concepts

o Addresses generated by compiler (i.e. compiler + linker) are referred as **logical addresses.**

o Addresses which can be see by the process when it is in the main memory referred as **physical addresses**.

o **MMU (Memory Management Unit):** which is a hardware unit converts logical address into physical address.

o **MMU is a hardware** contains adder circuit, comparator circuit, base register and limit register. Values of base register and limit registers gets change during context-switch, and memory space **of one process gets protected from another process**.

o CPU always executes program in its **logical memory space**.

➢ Memory Allocation:

o When a process is requesting for the main memory, there are two methods by which memory gets allocated for any process

1. Contiguous Memory Allocation

2. Non - contiguous Memory Allocation

1.   Contiguous Memory Allocation:

o Under this method, process can complete its execution only if memory gets allocated for it in a contiguous manner.

o There are two methods by which memory gets allocated for process under contiguous memory allocation method.

1. Fixed Size Partitioning

2. Variable Size Partitioning

1. Fixed Size Partitioning:

o In this method, physical memory i.e. main memory (user space) is divided into fixed number of partitions and size of each partition is remains fixed.

o If any process is requesting for the memory it can be loaded into main memory in any free partition in which it can be fit.

❖Advantages:

o This method is simple to implement.

❖Disadvantages:

o **Internal fragmentation**: if memory remains unused which is internal to the partition.

o **Degree of multi-programming** is limited to the number of partitions in the main memory.

o **Maximum size of a process is limited** to max size partition in the main memory.

o To overcome limitations/disadvantages of fixed size partitioning method, variable/dynamic size partitioning method has been designed.

2. Variable/Dynamic Size Partitioning:

- In this method, initially whole user space i.e. physical memory is considered as a single free partition, and processes gets loaded into the main memory as they request for it.

- Size of partition and number of partitions are not fixed in advanced, it gets decided dynamically.

❖Advantages:

- There are very less chances of **internal fragmentation** - Degree of multi-programming is not limited/fixed

- Size of the process is not also limited, any size process may get load into the main memory.

❖ Disadvantages:

- **External fragmentation:** due to loading and removing of processes into and from the main memory, main memory is fragmented i.e. gets divided into used partitions and free partitions.

2. Variable/Dynamic Size Partitioning:

- In this method, initially whole user space i.e. physical memory is considered as a single free partition, and processes gets loaded into the main memory as they request for it.

- Size of partition and number of partitions are not fixed in advanced, it gets decided dynamically.

❖Advantages:

- There are very less chances of **internal fragmentation** - Degree of multi-programming is not limited/fixed

- Size of the process is not also limited, any size process may get load into the main memory.

❖ Disadvantages:

- **External fragmentation:** due to loading and removing of processes into and from the main memory, main memory is fragmented i.e. gets divided into used partitions and free partitions.

Operating Systems Concepts

o **External fragmentation**: due to loading and removing of processes into and from the main memory, main memory gets fragmented i.e. it gets divided into used partitions and free partitions. In such case, if any new process is requesting for the memory and even if the requested size of memory is available, but due to unavailability of the memory in a contiguous manner request of that process cannot be accepted, this problem is referred as an external fragmentation.

o External fragmentation is the biggest problem under contiguous memory allocation, and hence there are two solutions on this problem:

o **1. Compaction**

o **2. Non- contiguous Memory Allocation**

Operating Systems Concepts

1. Compaction:

- shuffling of main memory can be done in such a way that all used partitions can be shifted to one side and all free partitions can be shifted to other side and contiguous large free partition will be made available for the new processes.
- Compaction is practically not feasible as there is need to do recalculations of addresses every time.

2. Non- contiguous Memory Allocation:

- Under this method, process can complete its execution even if memory gets allocated for it in a non - contiguous manner, and it can be achieved by two memory management techniques:

    1. **Segmentation**

    2. **Paging**

- So by using segmentation & paging techniques, process can complete its execution even after memory gets allocated for it in a **non- contiguous manner.**

1. Segmentation

o In this technique, process in its logical memory is divided into small size segments **like stack segment, heap segment, data segment, bss segment , rodata segment, code segment etc...,** and when process is requesting for memory it is not requesting memory contiguously for whole process, memory gets allocated contiguously only for small size segments, and segments of one process may get load into the memory randomly at any locations, i.e. for a process memory gets allocated **in a non - contiguous manner**, and then only an execution of a process can be completed.

o As segments of a one process gets loaded randomly into the main memory, and in a system thousands processes are running at a time, hence to keep track on all the segments of each process, an OS maintains one table per process referred as **a segmentation table in which information about all the segments of that process can be kept**.

o Using segmentation an **external fragmentation can be reduced but cannot be completely avoided => to overcome this limitation paging technique has been designed**.
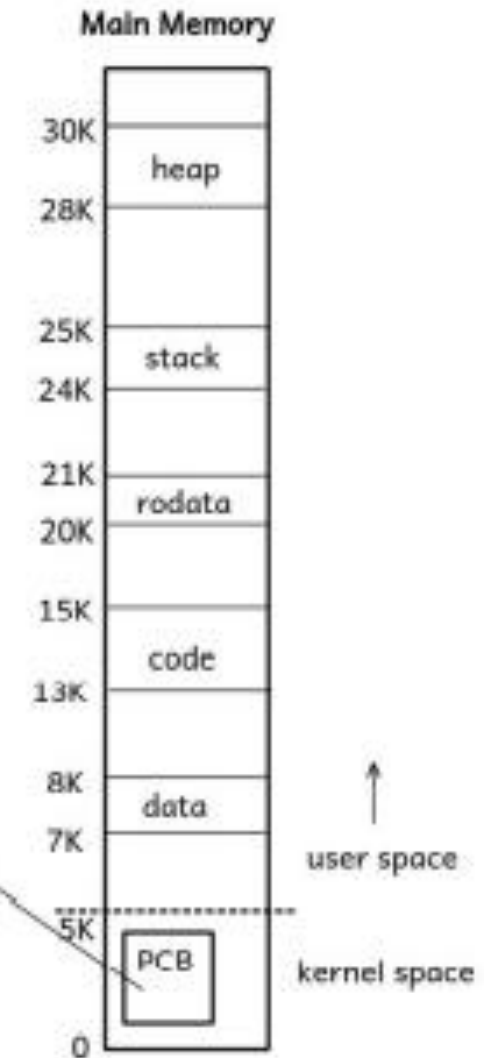
Operating Systems Concepts

Segmentation

Big size process gets divided logically into small size segments

Process: Size 7 K

0	stack	1K
1	heap	1K
2	rodata	1K
3	data	2K
4	code	2K

kernel space

segment table

seg addr	limit	base
0	1K	24000
1	1K	28000
2	1K	20000
3	2K	7000
4	2K	13000
5	null	null

Main Memory

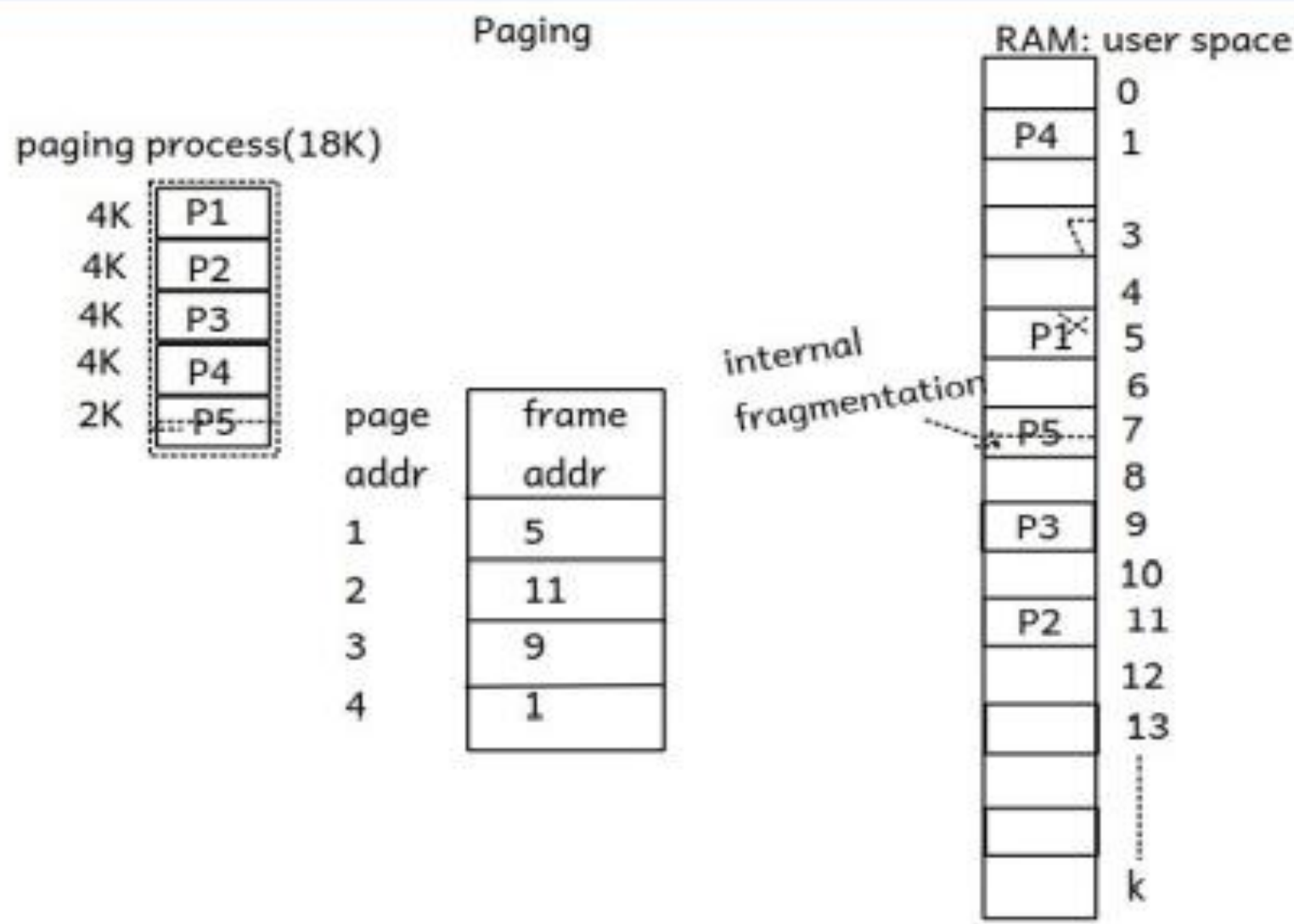|	
30K	heap
28K	
25K	stack
24K	
21K	rodata
20K	
15K	code
13K	
8K	data
7K	
5K	
	PCB
0	

user space

kernel space

2. Paging :

o In this technique, physical memory (i.e. user space of a main memory) is divided into fixed size of blocks referred as **frames**, and process's logical memory space is divided into same size of blocks referred as **pages,** whereas maximum size of page must be equal to size of frame, i.e.

o **if e.g. size of frame = 4K, then maximum size of each page must be 4K, size of page may be less than 4K.**

o As process is divided into pages, so when it is requesting for memory, pages of one process may gets loaded into the main memory at any free frames, and for a process memory gets allocated in a non - contiguous manner.

o As pages of a one process gets loaded randomly into the main memory, and in a system thousands processes are running at a time, so to keep track on all the pages of each process, an OS maintains one table per process referred as **a page table** in which information about all the pages of that process can be kept.

o There is no external fragmentation in paging.

o Internal fragmentation may exists in paging when the size of page is less than size of frame.

Operating Systems Concepts

Paging

paging process(18K)

4K	P1
4K	P2
4K	P3
4K	P4
2K	P5

page addr	frame addr
1	5
2	11
3	9
4	1

internal fragmentation

RAM: user space

	0
P4	1
	3
	4
P1	5
	6
P5	7
	8
P3	9
	10
P2	11
	12
	13
	k

Operating Systems Concepts

➢ Virtual Memory Management:

o As we seen an OS does memory management for completing an execution of multiple submitted processes at once.

o An OS also able to complete an execution of such a process having size bigger than size of main memory itself, and to achieve this an OS manages swap area memory as well with main memory and hence it is referred **as virtual memory** management.

o As an OS manages such a memory which is physically not a main memory and hence it is **referred as virtual memory management**.

o Virtual memory management can **be implemented by using Paging + Swapping**.

o In this technique for a process to complete its execution it is not mandatory it must exists wholly in a main memory, even its part is their into the main memory then execution of such process can be **completed part by part.**

o Big size process is divided into pages and when a process is requesting for memory few pages gets loaded into the main memory and few pages can be kept into the swap area, and as per the request **pages of that process can swapped in and swapped out between main memory and swap area.**

Operating Systems Concepts

o **Demand Paging:** any page of a process gets loaded into the main memory **only after requesting by that process i.e. on demand and hence referred as demand paging**, page which is never requested never gets loaded into the main memory and hence it also called **as pure demand paging**.

o If a process is requesting for any page and if that page is not exists in the main memory, then it is referred **as page fault.**

o As the size of process may be bigger than size of main memory itself, and in a system multiple processes are running at a time, hence no. of pages are more than no. of frames, so there are quite good chances that all frames becomes full, and in this case if any process is requesting for a page which does not exists in the main memory at that time there is need to remove any one page from the main memory so that into that free frame requested page will get loaded.

o So there is need to decide which page should get removed and requested page gets replaced in that frame, to do this there are certain algorithms referred as **page replacement algorithms.**

Operating Systems Concepts

➤ **Page Replacement Algorithms:**

1. **FIFO Page Replacement:** page which was inserted first gets replaced by the requested page.

2. **Optimal Page Replacement**: page which will not get used in a near future gets replaced by the requested page

3. **LRU(Least Recently Used) Page Replacement:** least recently used page gets replaced by the requested page.

4. **LFU(Least Frequently Used) Page Replacement:** least frequently used page gets replaced by the requested page.

5. **MFU(Most Frequently Used) Page Replacement:** most frequently used page gets replaced by the requested page.

o Algorithms 1, 2 & 3 uses stack based approach, whereas algorithms 4 & 5 uses counting based approach.

o Conceptually Optimal Page Replacement is the most efficient page replacement algorithm, as no. of page faults in this algorithm are very less, but as its practical implementation is not feasible hence LRU is the most efficient algorithm (implementation wise)

➢ Thrashing:

o If any process spends more time on paging rather than execution, then this high paging activity is referred as thrashing.

Thank you!

Kiran Jaybhave

email – kiran.jaybhave@sunbeaminfo.com