OPERATING SYSTEMS

Scheduling:

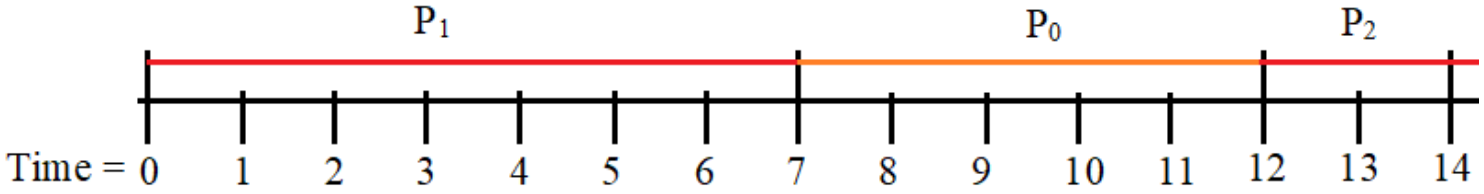
- FCFS
- SJF
- Round Robin
- SRF

First Come, First Serve:

- Jobs are executed on first come, first serve basis.
- It is a non-preemptive, pre-emptive scheduling algorithm.
- Easy to understand and implement.
- Its implementation is based on FIFO queue.
- Poor in performance as average wait time is high.

Ex:

	ST	BT
P0	2	5
P1	0	7
P2	3	2



So,

P1 = 0 – 0 = 0

P0 = 7 – 2 = 5

P2 = 12 – 3 = 9 The average would be (0 + 5 + 9) / 3

12 13 14

Shortest Job First:

- This is also known as **shortest job first**, or SJF
- This is a non-preemptive, pre-emptive scheduling algorithm.
- Best approach to minimize waiting time.
- Easy to implement in Batch systems where required CPU time is known in advance.
- Impossible to implement in interactive systems where required CPU time is not known.
- The processer should know in advance how much time process will take.
- Given: Table of processes, and their Arrival time, Execution time

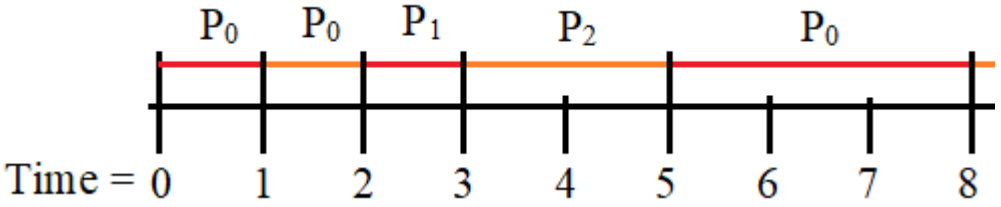
Ex:

	ST	BT
P0	0	5
P1	2	1
P2	3	2

At t = 0, P0 will be executed. At t = 1, again only P0 will be executed

At t = 2, P1 will be executed

At t = 3, P2 will be executed and after this, P0 will be executed



Terms:

(1) Starvation

Starvation occurs if a process is indefinitely postponed. This may happen if the process requires a resource for execution that it is never alloted or if the process is never provided the processor for some reason.

Some of the common causes of starvation are as follows −

- If a process is never provided the resources it requires for execution because of faulty resource allocation decisions, then starvation can occur.

- A lower priority process may wait forever if higher priority processes constantly monopolize the processor.
- Starvation may occur if there are not enough resources to provide to every process as required.
- If random selection of processes is used then a process may wait for a long time because of non-selection.

Some solutions that can be implemented in a system to handle starvation are as follows −

- An independent manager can be used for allocation of resources. This resource manager distributes resources fairly and tries to avoid starvation.
- Random selection of processes for resource allocation or processor allocation should be avoided as they encourage starvation.
- The priority scheme of resource allocation should include concepts such as aging, where the priority of a process is increased the longer it waits. This avoids starvation.

(2) Context Switching:

Context Switching involves storing the context or state of a process so that it can be reloaded when required and execution can be resumed from the same point as earlier. This is a feature of a multitasking operating system and allows a single CPU to be shared by multiple processes.

Context Switching Triggers

There are three major triggers for context switching. These are given as follows −

- **Multitasking**: In a multitasking environment, a process is switched out of the CPU so another process can be run. The state of the old process is saved and the state of the new process is loaded. On a pre-emptive system, processes may be switched out by the scheduler.
- **Interrupt Handling**: The hardware switches a part of the context when an interrupt occurs. This happens automatically. Only some of the context is changed to minimize the time required to handle the interrupt.
- **User and Kernel Mode Switching**: A context switch may take place when a transition between the user mode and kernel mode is required in the operating system.

Context Switching Steps

The steps involved in context switching are as follows −

- Save the context of the process that is currently running on the CPU. Update the process control block and other important fields.
- Move the process control block of the above process into the relevant queue such as the ready queue, I/O queue etc.
- Select a new process for execution.
- Update the process control block of the selected process. This includes updating the process state to running.
- Update the memory management data structures as required.
- Restore the context of the process that was previously running when it is loaded again on the processor. This is done by loading the previous values of the process control block and registers.

Process Control Board (PCB):

Process Control Block is a data structure that contains information of the process related to it. The process control block is also known as a task control block, entry of the process table, etc.

It is very important for process management as the data structuring for processes is done in terms of the PCB. It also defines the current state of the operating system.

The following are the data items −

Process State: This specifies the process state i.e. new, ready, running, waiting or terminated.

Process Number: This shows the number of the particular process.

Program Counter: This contains the address of the next instruction that needs to be executed in the process.

Registers: This specifies the registers that are used by the process. They may include accumulators, index registers, stack pointers, general purpose registers etc.

List of Open Files: These are the different files that are associated with the process

CPU Scheduling Information:

The process priority, pointers to scheduling queues etc. is the CPU scheduling information that is contained in the PCB. This may also include any other scheduling parameters.

Memory Management Information

The memory management information includes the page tables or the segment tables depending on the memory system used. It also contains the value of the base registers, limit registers etc.

I/O Status Information

This information includes the list of I/O devices used by the process, the list of files etc.

Accounting Information

The time limits, account numbers, amount of CPU used, process numbers etc. are all a part of the PCB accounting information.

Location of the Process Control Block

The process control block is kept in a memory area that is protected from the normal user access. This is done because it contains important process information. Some of the operating systems place the PCB at the beginning of the kernel stack for the process as it is a safe location.

Whenever this context switching is happening, the information regarding at which point we have to continue our work.

It has the state of all the process in the operating system

Round Robin

- Round Robin is the preemptive process scheduling algorithm.
- Each process is provided a fix time to execute, it is called a quantum.
- Once a process is executed for a given time period, it is preempted and other process executes for a given time period.
- Context switching is used to save states of preempted processes.

Quanta = k = 2.

P0 2

P1 4

P2 8

It starts from P0 and will execute it for t = 0 to t = 2. After this, it will go come P1 and will execute it for 2 sec. Then it will execute P2 for 2 sec.

Once this cycle of 2 sec is done, then it will go back to P0, since P0 is for 2 sec and it is done, it will come to P1 and execute it for 2 sec. With this, P1 is completed. Then P2 is executed for 2 sec.

By now P0 and P1 are completed so P2 will be executed 2 times for 2 seconds.

RAM:

RAM (Random Access Memory) is the internal memory of the CPU for storing data, program, and program result. It is a read/write memory which stores data until the machine is working. As soon as the machine is switched off, data is erased.

Access time in RAM is independent of the address, that is, each storage location inside the memory is as easy to reach as other locations and takes the same amount of time. Data in the RAM can be accessed randomly but it is very expensive.

RAM is volatile, i.e. data stored in it is lost when we switch off the computer or if there is a power failure. Hence, a backup Uninterruptible Power System (UPS) is often used with computers. RAM is small, both in terms of its physical size and in the amount of data it can hold.

RAM is of two types −

- Static RAM (SRAM)
- Dynamic RAM (DRAM)

Any program in the OS will be stored in RAM, if we write any program in the form arrays in diff language (Java, Python, C++) will store in RAM but there is a possibility of corrupting the array or that particular memory i.e., security concern memory, it can be avoided by Operating system

In RAM there are blocks and each block has some address. If we are buying a 64 bit RAM then it would have so many addresses available. The higher the RAM more things can be run in the RAM. Previously, how did operating systems managed memory.

External Fragmentation:

Total memory space is enough to satisfy a request or to reside a process in it, but it is not contiguous, so it cannot be used. External fragmentation can be reduced by compaction or shuffle memory contents to place all free memory together in one large block. To make compaction feasible, relocation should be dynamic.

First Fit

In the first fit approach is to allocate the first free partition or hole large enough which can accommodate the process. It finishes after finding the first suitable free partition.

Advantage	**Dis-advantage**
Fastest algorithm because it searches as little as possible.	The remaining unused memory areas left after allocation become waste if it is too smaller. Thus, request for larger memory requirement cannot be accomplished.

Best Fit

The best fit deals with allocating the smallest free partition which meets the requirement of the requesting process. This algorithm first searches the entire list of free partitions and considers the smallest hole that is adequate. It then tries to find a hole which is close to actual process size needed.

Advantage	**Dis-advantage**
Memory utilization is much better than first fit as it searches the smallest free partition first available.	It is slower and may even tend to fill up memory with tiny useless holes.

Worst fit

In worst fit approach is to locate largest available free portion so that the portion left will be big enough to be useful. It is the reverse of best fit.

Advantage	**Dis-advantage**
Reduces the rate of production of small gaps.	If a process requiring larger memory arrives at a later stage then it cannot be accommodated as the largest hole is already split and occupied.

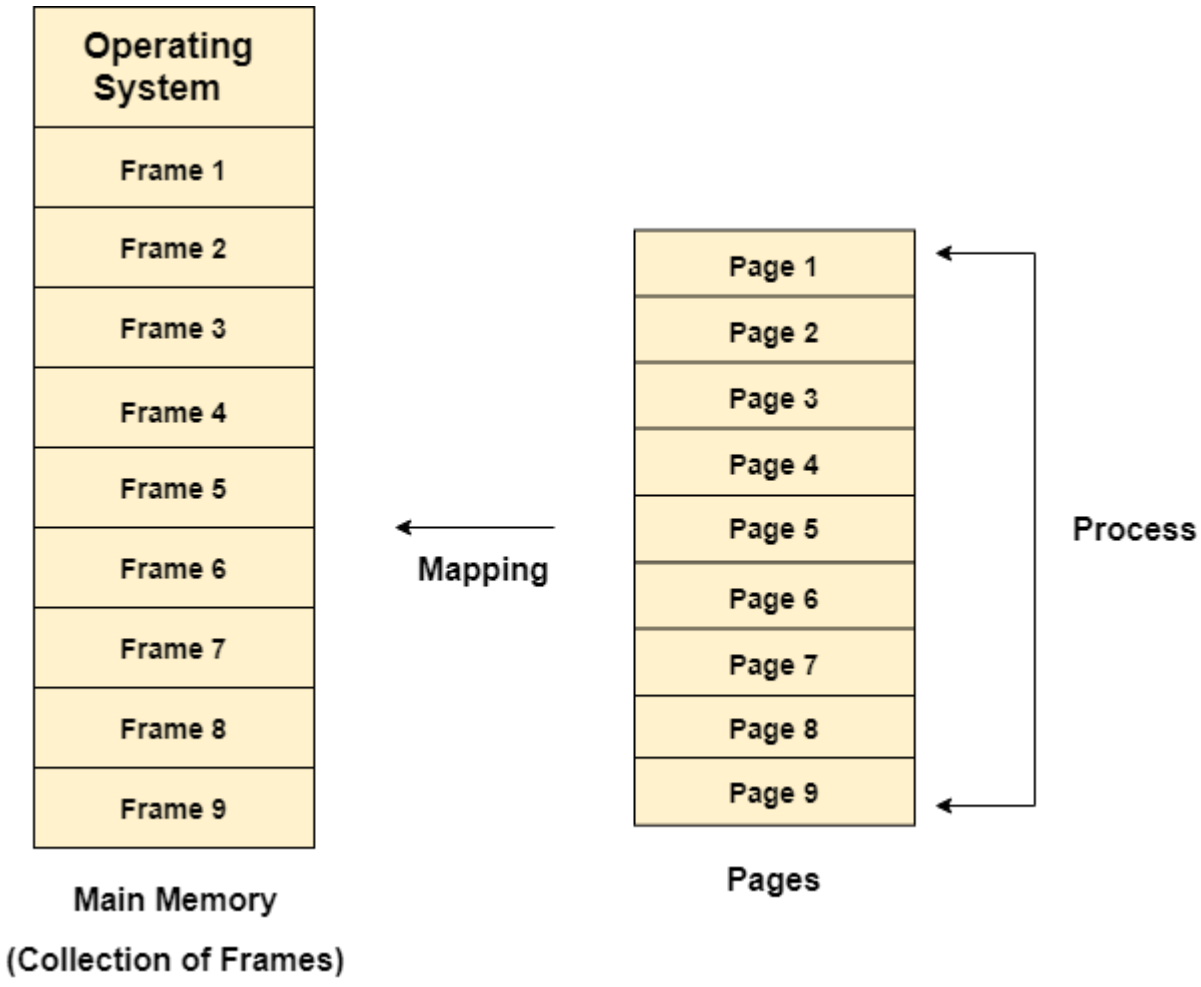
Using RAM to the fullest was not possible. So, what to do about it? Logical RAM (or Virtual Memory): We don't talk to the actual RAM.

Pagination:

In Operating Systems, Paging is a storage mechanism used to retrieve processes from the secondary storage into the main memory in the form of pages. The main idea behind the paging is to divide each process in the form of pages. The main memory will also be divided in the form of frames.

One page of the process is to be stored in one of the frames of the memory. The pages can be stored at the different locations of the memory but the priority is always to find the contiguous frames or holes. Pages of the process are brought into the main memory only when they are required otherwise, they reside in the secondary storage.

Different operating system defines different frame sizes. The sizes of each frame must be equal. Considering the fact that the pages are mapped to the frames in Paging, page size needs to be as same as frame size.

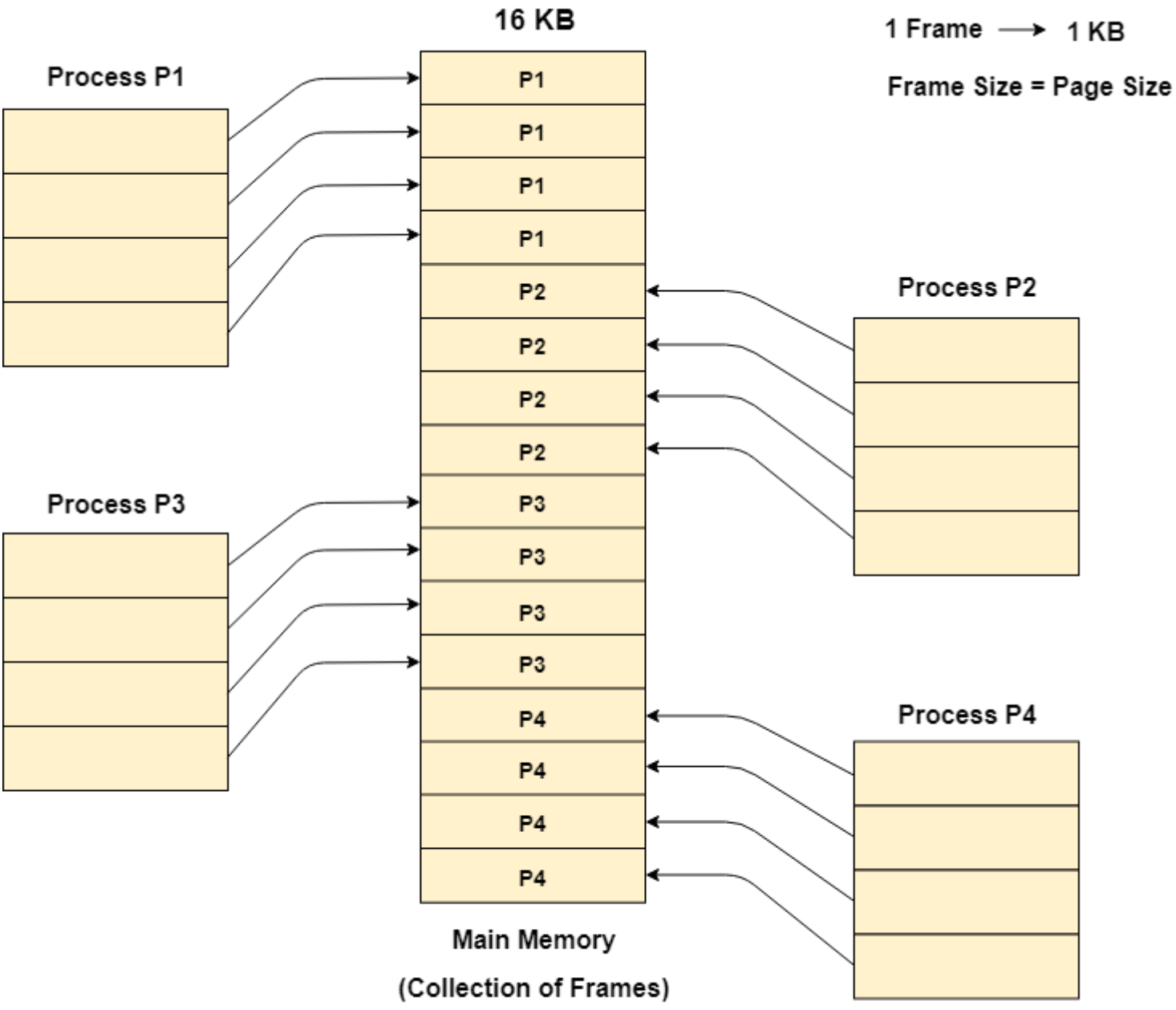


Example

Let us consider the main memory size 16 Kb and Frame size is 1 KB therefore the main memory will be divided into the collection of 16 frames of 1 KB each.

There are 4 processes in the system that is P1, P2, P3 and P4 of 4 KB each. Each process is divided into pages of 1 KB each so that one page can be stored in one frame. Initially, all the frames are empty therefore pages of the processes will get stored in the contiguous way.

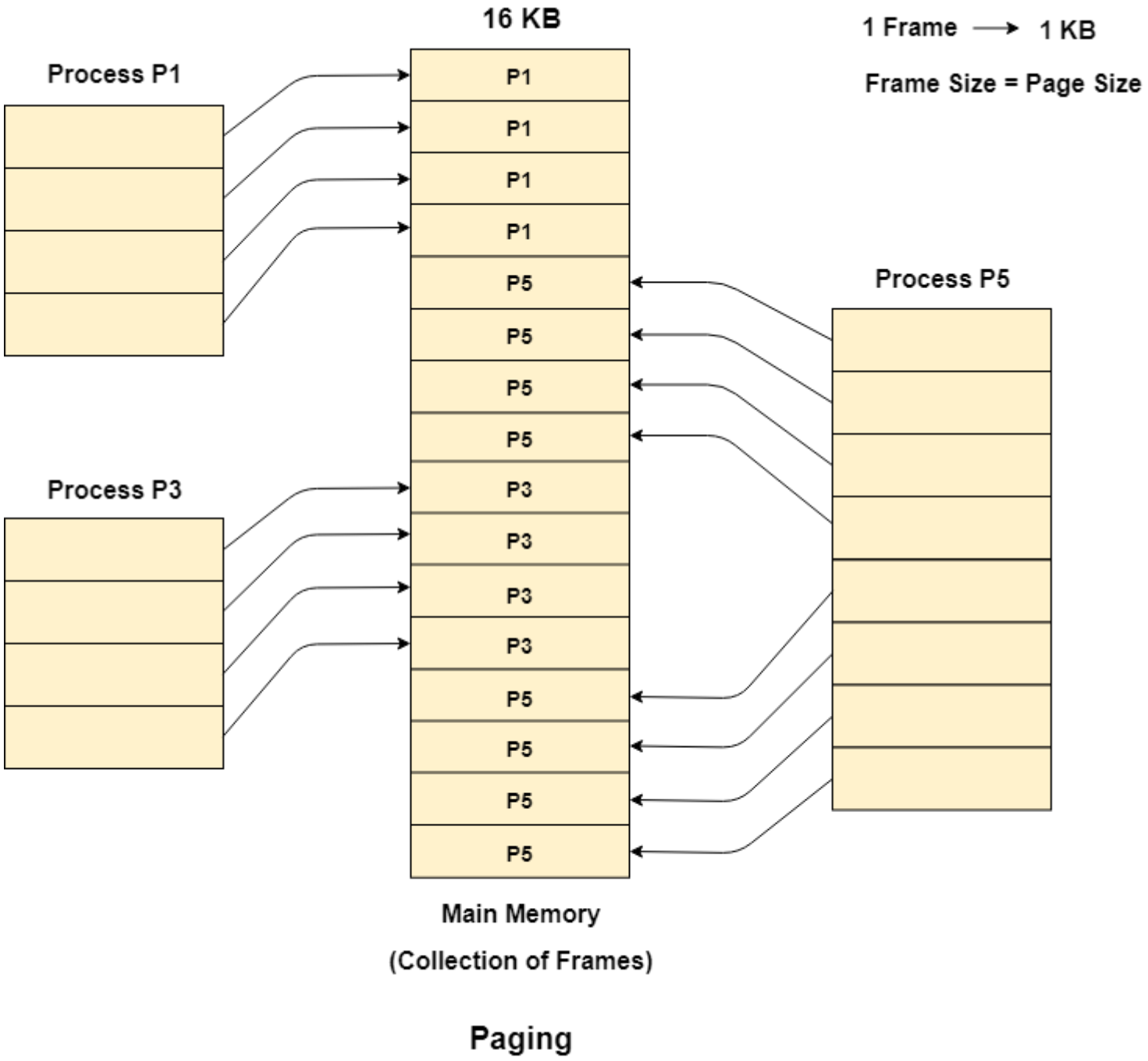
Frames, pages and the mapping between the two is shown in the image below.



Paging

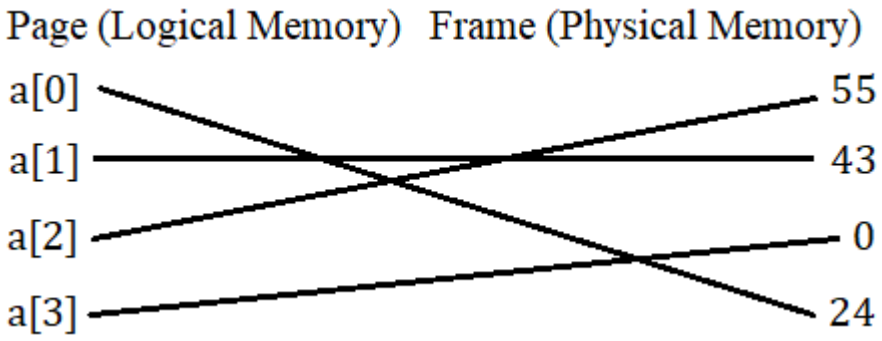
Let us consider that, P2 and P4 are moved to waiting state after some time. Now, 8 frames become empty and therefore other pages can be loaded in that empty place. The process P5 of size 8 KB (8 pages) is waiting inside the ready queue.

Given the fact that, we have 8 non contiguous frames available in the memory and paging provides the flexibility of storing the process at the different places. Therefore, we can load the pages of process P5 in the place of P2 and P4.



Paging

Pagination Table: (Page table)



Page matches the any exact block in frame based on memory required.

It matches the exact block based on the memory needed to avoid the wastage of storage

The logical memory blocks called page, Physical memory blocks called as frame,

Logical memory mapped to physical memory through page table and can store the data in any block of physical memory.

TLB: (Translation ***Lookaside*** Buffer) –

A page table stored in register and cache to access search faster.

A Translation look aside buffer can be defined as a memory cache which can be used to reduce the time taken to access the page table again and again. It is a memory cache which is closer to the CPU and the time taken by CPU to access TLB is lesser then that taken to access main memory. In other words, we can say that TLB is faster and smaller than the main memory but cheaper and bigger than the register.

TLB follows the concept of locality of reference which means that it contains only the entries of those many pages that are frequently accessed by the CPU.

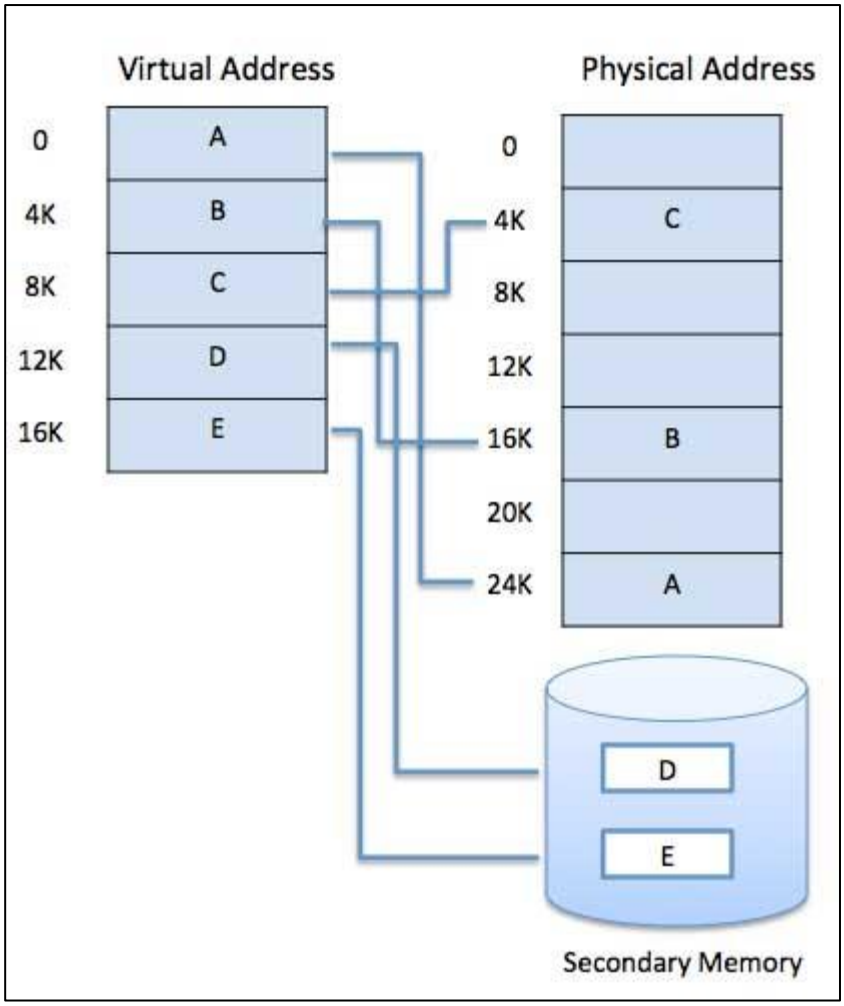
Difference between storing something in

HDD, RAM, Cache and Register.

HDD: costly to retrieve the data
RAM: 10times faster than HDD to retrieve the data
Cache: 10 times faster than RAM
Register: 10 times faster than Cache
SSD: solid state dries, faster than HDD and slower than RAM
Page table will store in cache or register to search faster

Virtual Memory:

Some space of HDD is used by RAM this is called Virtual memory. When the RAM is overloaded then it uses virtual memory. A computer can address more memory than the amount physically installed on the system. This extra memory is actually called virtual memory and it is a section of a hard disk that's set up to emulate the computer's RAM. The main visible advantage of this scheme is that programs can be larger than physical memory. Virtual memory serves two purposes. First, it allows us to extend the use of physical memory by using disk. Second, it allows us to have memory protection, because each virtual address is translated to a physical address.



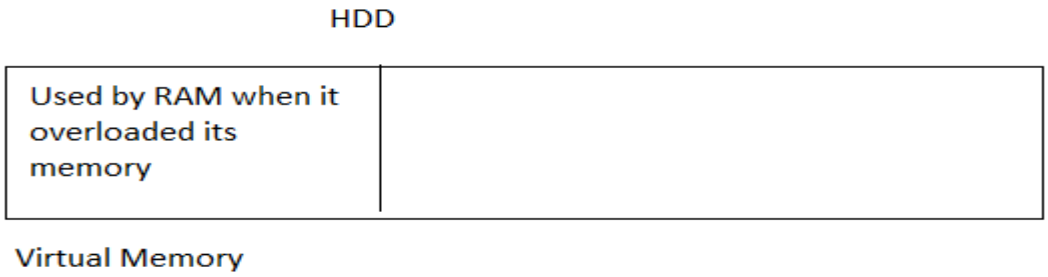
Modern microprocessors intended for general-purpose use, a memory management unit, or MMU, is built into the hardware. The MMU's job is to translate virtual addresses into physical addresses. A basic example is given to the right –

Following are the situations, when entire program is not required to be loaded fully in main memory.

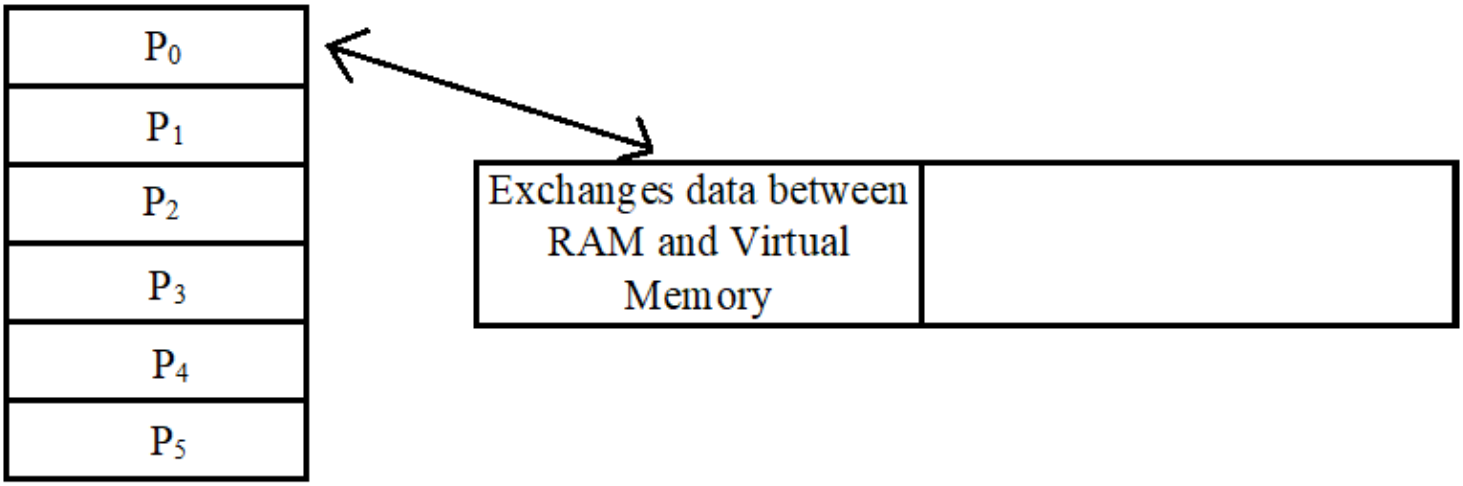
- User written error handling routines are used only when an error occurred in the data or computation.
- Certain options and features of a program may be used rarely.
- Many tables are assigned a fixed amount of address space even though only a small amount of the table is actually used.
- The ability to execute a program that is only partially in memory would counter many benefits.
- Less number of I/O would be needed to load or swap each user program into memory.

- A program would no longer be constrained by the amount of physical memory that is available.
- Each user program could take less physical memory, more programs could be run the same time, with a corresponding increase in CPU utilization and throughput.

Virtual memory is commonly implemented by demand paging. It can also be implemented in a segmentation system. Demand segmentation can also be used to provide virtual memory.



Swap Memory: when a processor needs to retrieve the data in Virtual memory then it Swaps the data from Virtual memory to RAM and performs the operation.



When OS wants to process data in ‘a’ it directly can not use it, so it swaps data from virtual memory to RAM and then it uses it. RAM and VM swaps their data to utilize it.

Page Eviction: It helps in swapping memory, it checks which data should swap from RAM to Virtual memory to replace the virtual memory data to RAM, it can be done in 2 ways

1.LRU: Least recently used – it says swap the least used page in RAM
2.LFU algorithms: least frequently used, it says swap the frequently used page in RAM and send to virtual memory and receives the page/data from virtual memory.

https://www.tutorialspoint.com/operating_system/index.htm