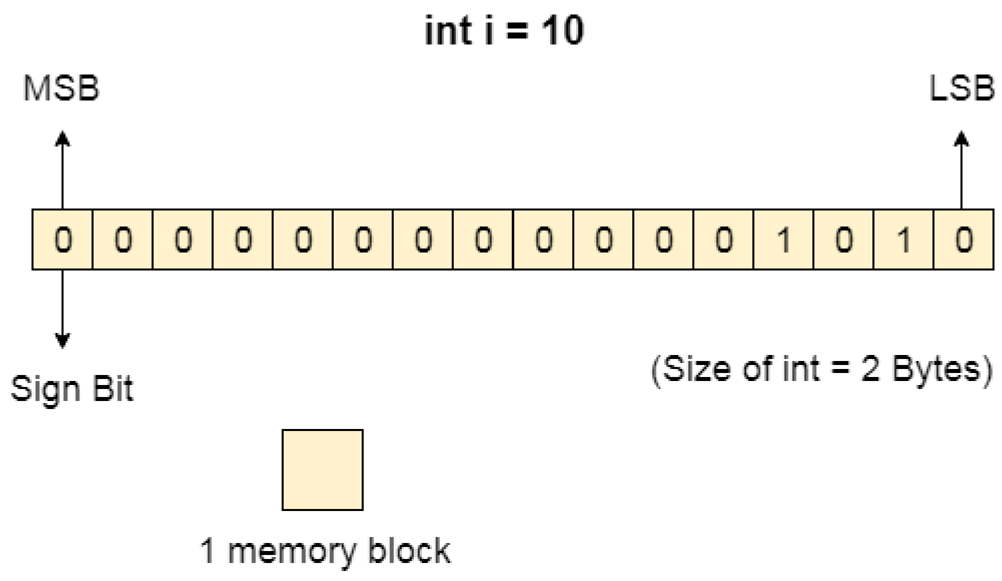
UNIT-IV

Memory Management:

A computer device that is capable to store any information or data temporally or permanently, is called storage device.

That means if we have a program line written as **int α = 10** then the computer converts it into the binary language and then store it into the memory blocks.

The representation of **inti = 10** is shown below.



How Data is being stored in a computer system?

In order to understand memory management, we have to make everything clear about how data is being stored in a computer system.

Machine understands only binary language that is 0 or 1. Computer converts every data into binary language first and then stores it into the memory.

Swapping in Operating System

Swapping is a memory management scheme in which any process can be temporarily swapped from main memory to secondary memory so that the main memory can be made available for other processes. It is used to improve main memory utilization. In secondary memory, the place where the swapped-out process is stored is called swap space.

The purpose of the swapping in operating system is to access the data present in the hard disk and bring it to RAM so that the application programs can use it. The thing to remember is that swapping is used only when data is not present in RAM.

Although the process of swapping affects the performance of the system, it helps to run larger and more than one process. This is the reason why swapping is also referred to as memory compaction.

The concept of swapping has divided into two more concepts: Swap-in and Swap-out.

- Swap-out is a method of removing a process from RAM and adding it to the hard disk.
- Swap-in is a method of removing a program from a hard disk and putting it back into the main memory or RAM.

Example: Suppose the user process's size is 2048KB and is a standard hard disk where swapping has a data transfer rate of 1Mbps. Now we will calculate how long it will take to transfer from main memory to secondary memory.

1. User process size is 2048Kb
2. Data transfer rate is 1Mbps = 1024 kbps
3. Time = process size / transfer rate
4. = 2048 / 1024
5. = 2 seconds
6. = 2000 milliseconds
7. Now taking swap-in and swap-out time, the process will take 4000 milliseconds.

Advantages of Swapping

1. It helps the CPU to manage multiple processes within a single main memory.
2. It helps to create and use virtual memory.
3. Swapping allows the CPU to perform multiple tasks simultaneously. Therefore, processes do not have to wait very long before they are executed.
4. It improves the main memory utilization.

Disadvantages of Swapping

1. If the computer system loses power, the user may lose all information related to the program in case of substantial swapping activity.
2. If the swapping algorithm is not good, the composite method can increase the number of Page Fault and decrease the overall processing performance.

CONTIGUOUS MEMORY ALLOCATION IN OS

What is Contiguous Memory Allocation in OS?

Contiguous memory allocation in the operating system is a memory allocation technique. But what is memory allocation? When a program or process is to be executed, it needs some space in the memory. For this reason, some part of the memory has to be allotted to a process according to its requirements. This process is called **memory allocation.**

One such memory allocation technique is contiguous memory allocation. As the name implies, we allocate contiguous blocks of memory to each process using this technique. So, whenever a process wants to enter the main memory, we allocate a continuous segment from the totally empty space to the process based on its size.

Contiguous Memory Allocation Techniques

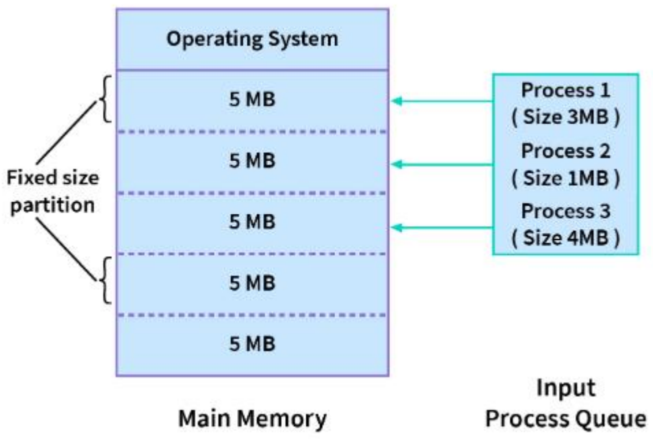
Whenever a process has to be allocated space in the memory, following the contiguous memory allocation technique, we have to allot the process a continuous empty block of space to reside. This allocation can be done in two ways:

1. Fixed-size Partition Scheme
2. Variable-size Partition Scheme

Let us look at both of these schemes in detail, along with their advantages and disadvantages.

Fixed-size Partition Scheme

In this type of contiguous memory allocation technique, **each process is allotted a fixed size continuous block in the main memory.** That means there will be continuous blocks of fixed size into which the complete memory will be divided, and each time a process comes in, it will be allotted one of the free blocks. Because irrespective of the size of the process, each is allotted a block of the same size memory space. This technique is also called **static partitioning.**



In the diagram above, we have 3 processes in the input queue that have to be allotted space in the memory. As we are following the fixed size partition technique, the memory has fixed-sized blocks. The first process, which is of size 3MB is also allotted a 5MB block, and the second process, which is of size 1MB, is also allotted a 5MB block, and the 4MB process is also allotted a 5MB block. So, the process size doesn't matter. Each is allotted the same fixed-size memory block.

It is clear that in this scheme, the number of continuous blocks into which the memory will be divided will be decided by the amount of space each block covers, and this, in turn, will dictate how many processes can stay in the main memory at once.

Note: The number of processes that can stay in the memory at once is called the degree of multiprogramming. Hence, the degree of multiprogramming of the system is decided by the number of blocks created in the memory.

Advantages

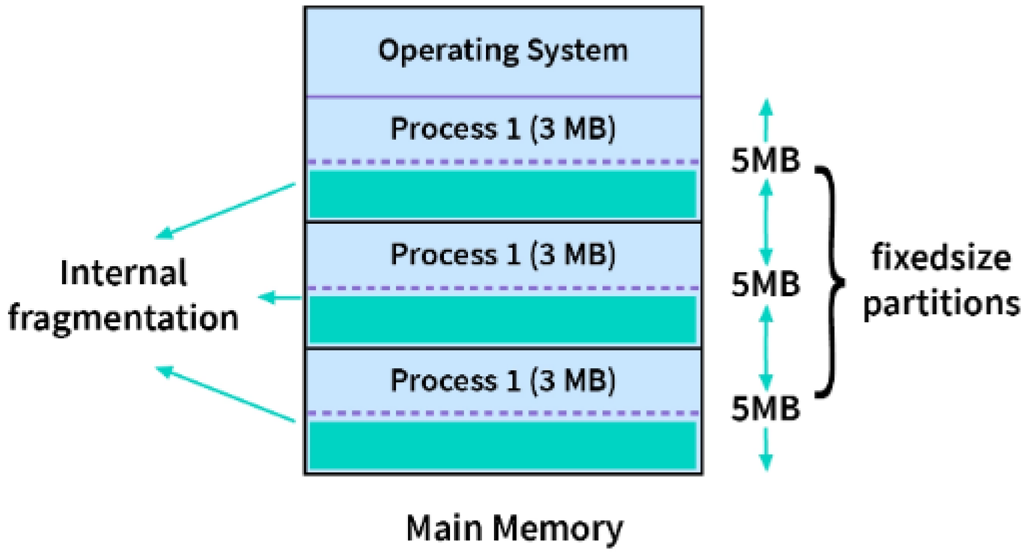
The advantages of a fixed-size partition scheme are:

1. Because all of the blocks are the same size, this scheme is simple to implement. All we have to do now is divide the memory into fixed blocks and assign processes to them.
2. It is easy to keep track of how many blocks of memory are left, which in turn decides how many more processes can be given space in the memory.
3. As at a time multiple processes can be kept in the memory, this scheme can be implemented in a system that needs multiprogramming.

Disdvantages

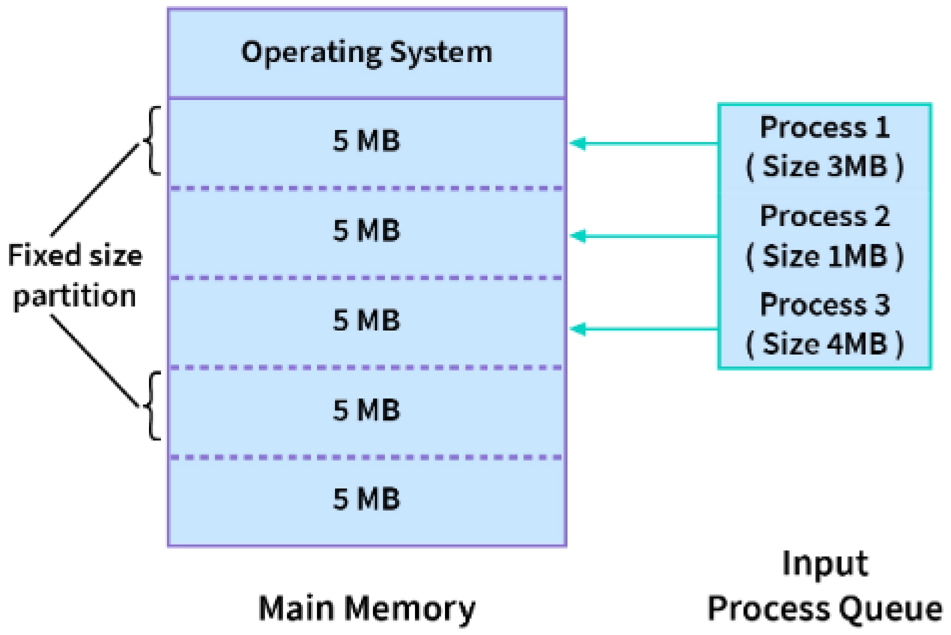
Though the fixed-size partition scheme has many advantages, it also has some disadvantages:

1. As the size of the blocks is fixed, we will not be able to allot space to a process that has a greater size than the block.
2. The size of the blocks decides the degree of multiprogramming, and only that many processes can remain in the memory at once as the number of blocks.
3. If the size of the block is greater than the size of the process, we have no other choice but to assign the process to this block, but this will lead to much empty space left behind in the block. This empty space could've been used to accommodate a different process. This is called internal fragmentation. Hence, this technique may lead to space wastage.



Variable-size Partition Scheme

In this type of contiguous memory allocation technique, **no fixed blocks or partitions are made in the memory.** Instead, each process is allotted a variable-sized block depending upon its requirements. That means, whenever a new process wants some space in the memory, if available, this amount of space is allotted to it. Hence, the size of each block depends on the size and requirements of the process which occupies it.



In the diagram above, there are no fixed-size partitions. Instead, the first process needs 3MB memory space and hence is allotted that much only. Similarly, the other 3 processes are allotted only that much space that is required by them.

As the blocks are variable-sized, which is decided as processes arrive, this scheme is also called Dynamic Partitioning.

Advantages

The advantages of a variable-size partition scheme are:

1. As the processes have blocks of space allotted to them as per their requirements, there is no internal fragmentation. Hence, there is no memory wastage in this scheme.
2. The number of processes that can be in the memory at once will depend upon how many processes are in the memory and how much space they occupy. Hence, it will be different for different cases and will be dynamic.
3. As there are no blocks that are of fixed size, even a process of big size can be allotted space.

Disadvantages

Though the variable-size partition scheme has many advantages, it also has some disadvantages:

1. Because this approach is dynamic, a variable-size partition scheme is difficult to implement.
2. It is difficult to keep track of processes and the remaining space in the memory.

Strategies Used for Contiguous Memory Allocation Input Queues

So far, we've seen the two types of schemes for contiguous memory allocation. But what happens when a new process comes in and has to be allotted a space in the main memory? How is it decided which block or segment it will get?

Processes that have been assigned continuous blocks of memory will fill the main memory at any given time. However, when a process completes, it leaves behind an empty block known as a hole. This space could also be used for a new process. Hence, the main memory consists of processes and holes, and any one of these holes can be allotted to a new incoming process. We have three strategies to allot a hole to an incoming process:

First-Fit

This is a very basic strategy in which we start from the beginning and allot the first hole, which is big enough as per the requirements of the process. The first-fit strategy can also be implemented in a way where we can start our search for the first-fit hole from the place we left off last time.

Best-Fit

This is a greedy strategy that aims to reduce any memory wasted because of internal fragmentation in the case of static partitioning, and hence we allot that hole to the process, which is the smallest hole that fits the requirements of the process. Hence, we need to first sort the holes according to their sizes and pick the best fit for the process without wasting memory.

Worst-Fit

This strategy is the opposite of the Best-Fit strategy. We sort the holes according to their sizes and choose the largest hole to be allotted to the incoming process. The idea behind this allocation is that as the process is allotted a large hole, it will have a lot of space left behind as internal fragmentation. Hence, this will create a hole that will be large enough to accommodate a few other processes.

PAGING AND SEGMENTATION

External fragmentation occurs because we allocate memory continuously to the processes. Due to this space is left and memory remains unused hence, cause external fragmentation. So to tackle this problem the concept of paging was introduced where we divide the process into small pages and these pages are allocated memory non-contiguously into the RAM. So, let's get started and learn more about it.

Non-Contiguous Memory Allocation Technique

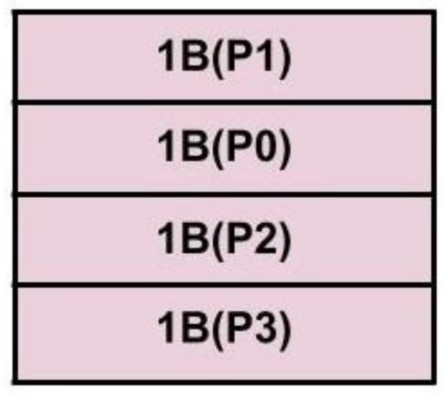
In the non-contiguous memory allocation technique, different parts of the same process are stored in different places of the main memory. Types:

1. Paging
2. Segmentation

Paging

Paging is a non-contiguous memory allocation technique in which secondary memory and the main memory is divided into equal size partitions. The partitions of the secondary memory are called pages while the partitions of the main memory are called frames . They are divided into equal size partitions to have maximum utilization of the main memory and avoid external fragmentation.

Example: We have a process P having process size as 4B, page size as 1B. Therefore there will we four pages(say, P0, P1, P2, P3) each of size 1B. Also, when this process goes into the main memory for execution then depending upon the availability, it may be stored in non-contiguous fashion in the main memory frame as shown below:



Main Memory

This is how paging is done.

Translation of logical Address into physical Address

As a CPU always generates a logical address and we need a physical address for accessing the main memory. This mapping is done by the MMU(memory management Unit) with the help of the page table . Lets first understand some of the basic terms then we will see how this translation is done.

- Logical Address: The logical address consists of two parts page number and page offset.

1. Page Number: It tells the exact page of the process which the CPU wants to access.

2**. Page Offset:** It tells the exact word on that page which the CPU wants to read.

Logical Address = Page Number + Page Offset

- **Physical Address**: The physical address consists of two parts frame number and page offset.

1. **Frame Number:** It tells the exact frame where the page is stored in physical memory.

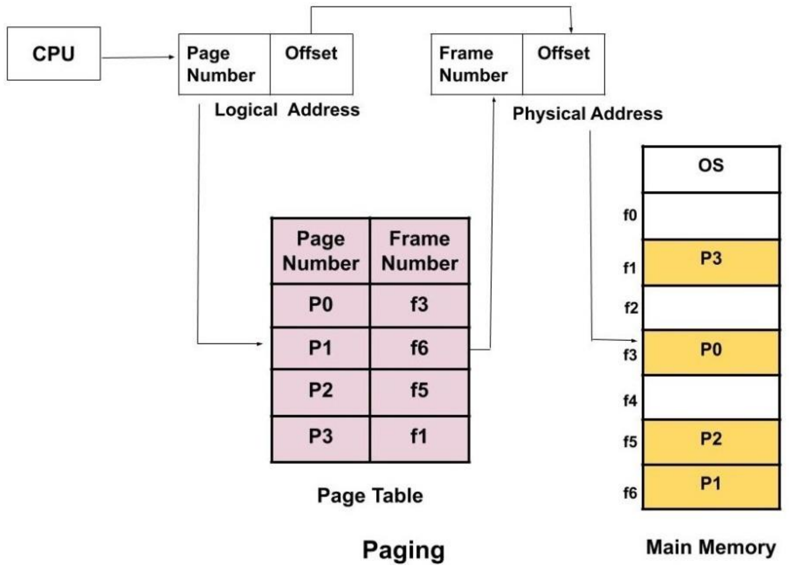
2. **Page Offset:** It tells the exact word on that page which the CPU wants to read. It requires no translation as the page size is the same as the frame size so the place of the word which CPU wants access will not change.

Physical Address = Frame Number + Page Offset

- Page table: A page stable contains the frame number corresponding to the page number of some specific process. So, each process will have its own page table. A register called Page Table Base Register(PTBR) which holds the base value of the page table.

How is the translation done?

The CPU generates the logical address which contains the page number and the page offset . The PTBR register contains the address of the page table. Now, the page table helps in determining the frame number corresponding to the page number. Now, with the help of frame number and the page offset the physical address is determined and the page is accessed in the main memory.



Advantages of Paging

1. There is no external fragmentation as it allows us to store the data in a non-contiguous way.
2. Swapping is easy between equal-sized pages and frames.

Disadvantages of Paging

1. As the size of the frame is fixed, so it may suffer from internal fragmentation. It may happen that the process is too small and it may not acquire the entire frame size.
2. The access time increases because of paging as the main memory has to be now accessed two times. First, we need to access the page table which is also stored in the main memory and second, combine the frame number with the page offset and then get the physical address of the page which is again stored in the main memory.
3. For every process, we have an independent page table and maintaining the page table is extra overhead.

SEGMENTATION

In paging, we were blindly diving the process into pages of fixed sizes but in segmentation, we divide the process into modules for better visualization of the process. Here each segment or module consists of the same type of functions. For example, the main function is included in one segment, library function is kept in other segments, and so on. As the size of segments may vary, so memory is divided into variable size parts.

Translation of logical Address into physical Address

As a CPU always generates a logical address and we need a physical address for accessing the main memory. This mapping is done by the MMU(memory management Unit) with the help of the segment table .

Lets first understand some of the basic terms then we will see how this translation is done.

- Logical Address: The logical address consists of two parts segment number and page offset.

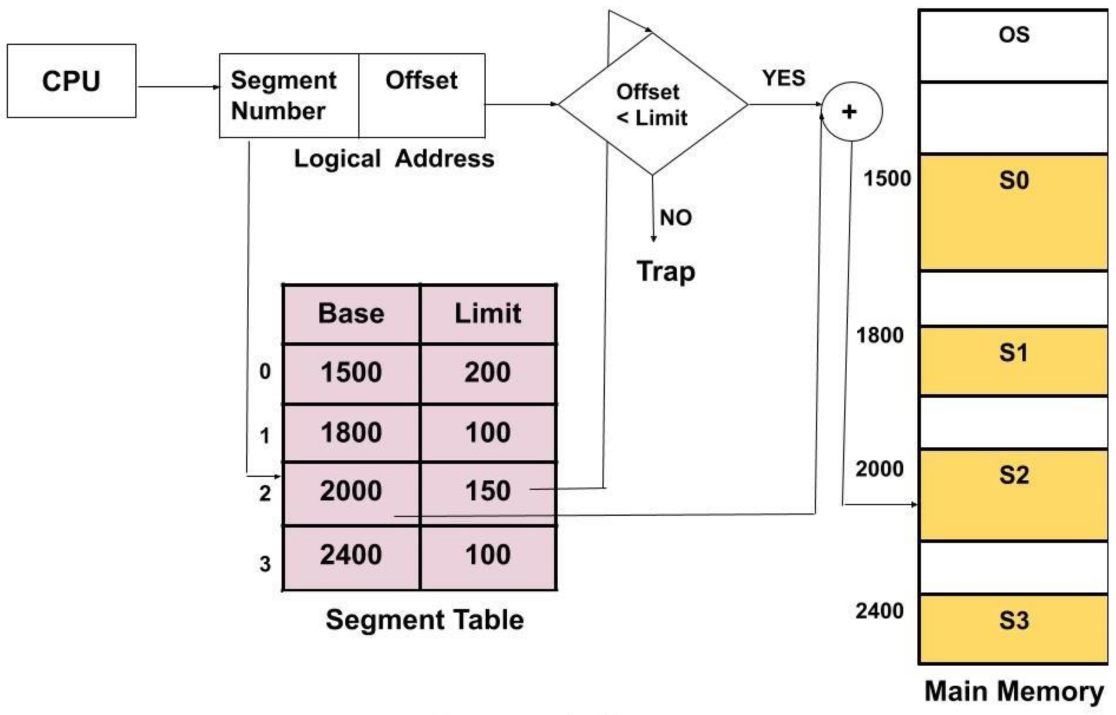
1. Segment Number: It tells the specific segment of the process from which the CPU wants to read the data.
2. Segment Offset: It tells the exact word in that segment which the CPU wants to read.

Logical Address = Segment Number + Segment Offset

- Physical Address: The physical address is obtained by adding the base address of the segment to the segment offset.
- Segment table: A segment table stores the base address of each segment in the main memory. It has two parts i.e. Base and Limit . Here, base indicates the base address or starting address of the segment in the main memory. Limit tells the size of that segment. A register called Segment Table Base Register(STBR) which holds the base value of the segment table. The segment table is also stored in the main memory itself.

How is the translation done?

The CPU generates the logical address which contains the segment number and the segment offset . STBR register contains the address of the segment table. Now, the segment table helps in determining the base address of the segment corresponding to the page number. Now, the segment offset is compared with the limit corresponding to the Base. If the segment offset is greater than the limit then it is an invalid address. This is because the CPU is trying to access a word in the segment and this value is greater than the size of the segment itself which is not possible. If the segment offset is less than or equal to the limit then only the request is accepted. The physical address is generated by adding the base address of the segment to the segment offset.



Segmentation

Advantages of Segmentation

1. The size of the segment table is less compared to the size of the page table.
2. There is no internal fragmentation.

Disadvantages of Segmentation

1. When the processes are loaded and removed (during swapping) from the main memory then free memory spaces are broken into smaller pieces and this causes external fragmentation.
2. Here also the time to access the data increases as due to segmentation the main memory has to be now accessed two times. First, we need to access the segment table which is also stored in the main memory and second, combine the base address of the segment with the segment offset and then get the physical address which is again stored in the main memory.

32 bit and 64 bit Operating System

In computing, a byte is the unit of data, and processing is generally denoted as bit processing. In general, there exist two types of processors, namely a 32-bit processor and a 64-bit processor. This type of processor tells us how much memory a processor can have access from a CPU register.

- A 32-bit system can access 2^{32} memory addresses, i.e., 4 GB of RAM or physical memory; ideally, it can also access more than 4 GB of RAM.
- A 64-bit system can access 2^{64} memory addresses, i.e., actually 18-Quintillion bytes of RAM. In short, any amount of memory greater than 4 GB can be easily handled by it.

With an increase in the availability of 64-bit processors and larger RAM capacities, Microsoft and Apple both have upgraded versions of their operating systems designed to take full advantage of the new technology. The first fully 64-bit operating system was Mac OS X Snow Leopard in 2009. Meanwhile, the first Smartphone with a 64-bit chip (Apple A7) was the iPhone 5s.

What is 32-Bit Operating System?

It is a CPU architecture type that holds the capacity to transfer 32 bits of data. It refers to the amount of data and information that your CPU can easily process when operating. A majority of the computers produced in the early 2000s and 1990s were 32-bit machines.

One bit in the register can typically reference an individual byte. Thus, the 32-bit system is capable of addressing about 4,294,967,296 bytes (4 GB) of RAM. Its actual limit is less than 3.5 GB (usually) because a portion of the register stores various other temporary values apart from the memory addresses.

What is 64-Bit Operating System?

The 64-bit microprocessor allows computer systems to process information, data, and memory addresses represented by 64 bits. Such a system can typically reference 16 exabytes (17,179,869,184 GB), or 18,446,744,073,709,551,616 bytes of memory.

A 64-bit system (a computer with a 64-bit processor) can access more than 4 GB of RAM. It is numerous million times more than what an average workstation would require to access. It means that if a computer has 8 GB of RAM, it requires a 64-bit processor. Or else, the CPU will be inaccessible to at least 4 GB of the memory.

Advantages of 64-bit over the 32-bit operating system

Below are the following advantages of a 64-bit operating system over the 32-bit operating system, such as:

Addressable Memory
Available Resources
Computer Performance
Software Performance
Multi-Tasking

1. **Addressable memory:** 32-bit operating systems can address a maximum of 4 GB of RAM. But 64-bit operating system can address up to 17,179,869,184 GB (16 exabytes). That's a lot more than 4GB of memory that a 32-bit operating system can handle.
2. **Available Resources:** The 64-bit operating system can make full use of available system resources compared to a 32-bit system. To simplify, installing more RAM on a system with a 32-bit OS doesn't impact performance. However, upgrade that system with excess RAM to the 64-bit version of Windows, and you'll notice a difference.
3. **Computer Performance:** The system can perform more calculations per second using a 64-bit system with a 64-bit processer. As a result, it increases the processing power and makes a computer run faster. This is limited in the case of 32-bit operating systems.
4. **Software performance:** More software's are written to leverage the benefit of a 64-bit operating system fully. If you are using a 64-bit operating system and install software of 64 bit, you can up-front notice the increase in performance. It becomes even more critical when performing a huge operation that requires the system to access more memory. An increase in software performance leads results in an increase in overall efficiency.
5. **Multitasking:** Using 64-bit, users can do various things in multitasking at the same time. Users can easily switch between various applications without any windows hanging problems.

Difference between 32-bit and 64-bit OS

Parameters	32-bit Processors	64-bit Processors
Handling of Data and Storage	As its name suggests, the 32 bit OS can store and handle lesser data than the 64 bit OS. More specifically, it addresses a maximum of 4,294,967,296 bytes (4 GB) of RAM.	The 64 bit OS, on the other hand, can handle more data than the 32 bit OS. It means that it can address a total of 264 memory addresses, which is 18-Quintillion GB of RAM.
Architecture	The 32-bit system has general computing, including IBM System/360 and IBM System/370, the DEC VAX, the Motorola 68000 Family, the Intel IA-32, and the 32-bit version of x86 architecture different versions. These are architectures that are used for embedded computing and include 68000 families.	The registers are divided into different groups like integer, floating, control and often for addresses of various uses and names like address, index or base registers. The size of these registers is dependent on the amount of addressable memory.
Compatibility of System	A 32-bit processor system could properly run a 32-bit OS, but it cannot run the 64-bit OS at its full capability.	A 64-bit processor system can run either a 32-bit or 64-bit version of an installed operating system (OS).
Performance	The factor of performance in a 32-bit processor is less efficient than the 64-bit processor.	It exhibits a higher performance than the 32-bit processor.
Application Support	The 64-bit programs and applications won't work.	The 32-bit programs and applications will work with no hassle.
Addressable Space	It has an addressable space of 4 GB.	These have an addressable space of 16 GB.
Calculation per second	32-bit systems have dual-core and quad-core versions available.	64bit systems can come with dual-core, quad-core, six-core, and eight-core versions. Having these multiple cores available has increased its speed of calculations per second.
Multitasking Support	The 32-bit system is not an ideal option for multitasking and stress-testing.	For multitasking and stress testing, the 64-bit processor is better. It also works well for the execution of other heavy applications.
OS Support	It needs a 32-bit operating system.	This one can run on both 32-

		bit and the 64-bit operating system.
OS and CPU Requirements	The 32-bit applications and operating systems require 32-bit CPUs.	The 64-bit operating system needs a 64-bit CPU, and the 64-bit applications require a 64-bit CPU and OS.
Systems Available	These support Windows 7, Windows XP, Windows Vista, Windows 8, and Linux.	These support Windows XP Professional, Windows 7, Windows 8, Windows 10, Windows Vista, Linux, and Mac OS X.
Limits in Memory	A 32-bit system has a limit of 32 bit Windows 3.2 GB of RAM. The limit in its addressable space doesn't allow you to use the entire physical memory space of 4GB.	A 64-bit system enables its users to store up to 17 Billion GB of RAM.

ARM ARCHITECTURE

Advanced RISC Machine or Acorn RISC Machine is the architecture with different computing architectures set to be used in different environments. 32-bit and 64-bit can be used here in different computer processors. It was developed by Arm Holdings and the architecture is updated in between. This architecture is specified to be used with CPU, different chips in the system, and in different registers. Reduced Instruction Set Computing helps in creating instructions for the system to be used for several purposes. Smartphones, microcomputers, and embedded devices also use ARM architecture for the instruction set in the registers.

What is ARM Architecture?

Many Arm-based devices are used all over the world and are one of the popular architectures within the devices. The architecture can be divided into A, R, and M profiles. A profile is mainly for applications, R profile is for real-time and M profile is for Microcontroller. A profile helps to maintain high performance and is designed to run the complex system in Linux or Windows. R profile checks for systems with real-time requirements and is found in networking equipment or embedded control systems. M profile is used in IOT devices and can be synchronized with small and high-power devices.

Components of ARM Architecture

The instruction set in the architecture describes the function of each instruction and explains the representation of the instruction in the memory through encoding. With the help of these instructions, it is easy to manage the architecture and the microprocessors.

Priority encoders help to load the instruction and to store it in the specified register in order to manage the files. This helps to identify the registers and instructions easily in the architecture.

Multiplexers are used in the architecture to manage the operation of processor buses. The components are instructed to work in behavioral mode and the components are implemented as an entity. The architecture of the entity is optimized depending on the application of the processor and helps to construct and maintain the design.

32-bit inputs are used in Arithmetic and Logic Unit which comes from register files and shifter. The outputs are modified in register flags. There are C flag, V flag, S flag and Z flag. 16 opcodes can be implemented with the help of 4-bit function bus in the microprocessor. V-bit output goes to the V flag and C output to the C flag and so on.

Booth Multiplier Factor has 32-bit inputs to manage from the register file. The output given is also 32-bit and the multiplication starts when the input is actively working with the processor.

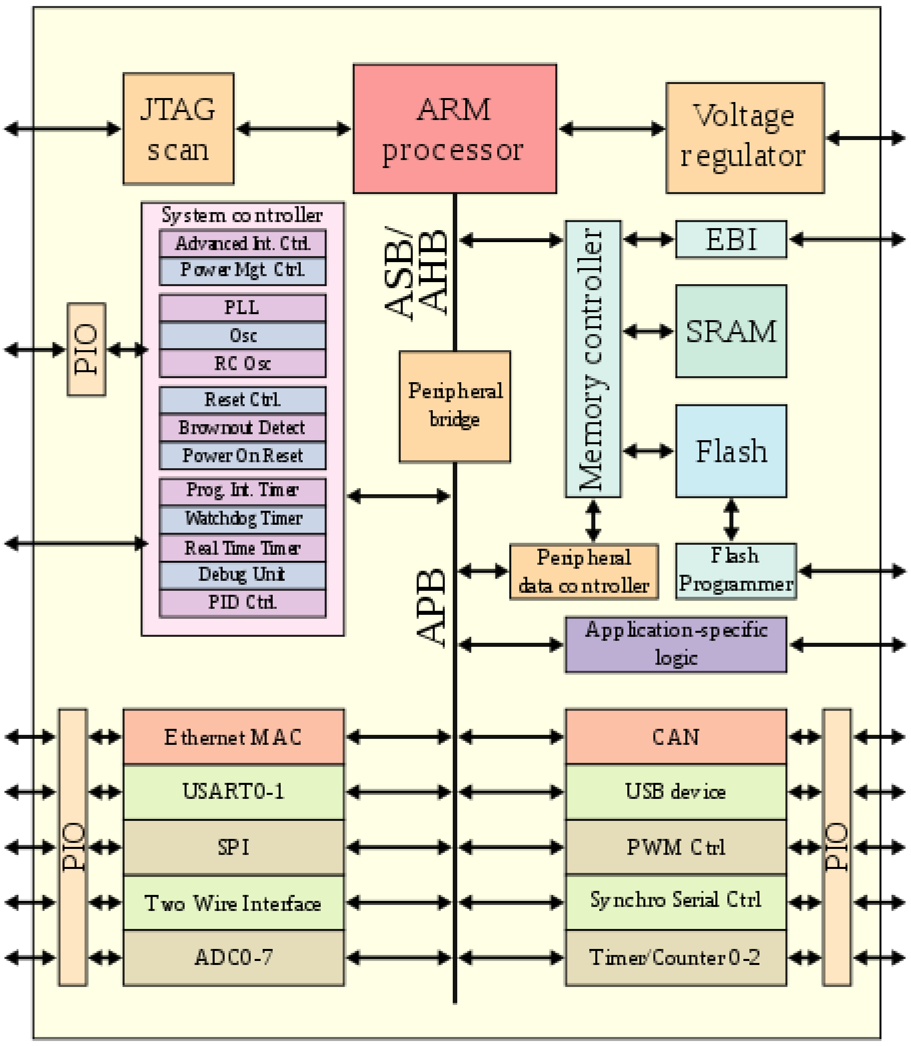
Also, another multiplication rule is used in the system for 2’s complement numbers. The Booth algorithm manages both positive and negative numbers with the same importance and faster multiplication is performed by ignoring 0’s and 1’s in the process. 16 clock cycle manages the multiplication cycle in the system.

32-bit input shifting is represented with the help of Barrel Shifter. The input is either from the register file or from the data given by the user for faster output. Shifter manages these with the help of different controls and also with the functionalities of the system. The logical or arithmetic operation to be performed is indicated in the shift field of the barrel shifter. The quantity is mentioned in the instruction field and it can be as low as 6 bits in the register. It allows up to 32-bit file and responds to left, right, arithmetic right, and rotate right shifts. This works well with multiplexers in the microprocessor.

The Control unit controls the entire process of the architecture and manages the system operation. It can be made of circuits or can be the combination of functions and circuits in the design. Timing is managed in the control unit and works with a combination of a state machine in the processor. The operation in the processor is managed and controlled with the help of signals from the control unit as these are connected with different components in the system.

There is a register set in the architecture to help in managing the registers with their corresponding time in the structure. It is important to check for the number of registers and the size of the same for the proper functioning of the system. The function of registers is to manage the files and keep them in the proper place and check for their initial state in the processor.

The exception model in the architecture helps to know the different levels of access provided in the system and the types of exceptions in the system. When an exception is accepted, the changes undergone in the system are also noticed. Breakpoints are noticed and information is captured with the help of tracepoints.



Benefits

Energy consumed is very low in the architecture and their design is such that with high processing power, low energy to be consumed. Hence devices with high performance prefer this architecture in the device.

Cost of the architecture is very less when compared with others. This makes ARM architecture popular among devices. Portable devices prefer this architecture for the performance and low-cost model in the system.

The pipeline used is simple and at a low power envelope, normal loads are managed. The dies are smaller and this helps the system to manage the defects with better outputs in the system. A single wafer manages many dies and this helps to manage the costs of the processor.

The security offered in the architecture is incomparable with others. Also, they came up with virtualization first in the structure. This helps the system to manage the processes in the architecture easily in portable devices and also in CPU-managed devices.

VIRTUAL-MEMORY MANAGEMENT:

Virtual Memory is a storage allocation scheme in which secondary memory can be addressed as though it were part of the main memory. The addresses a program may use to reference memory are distinguished from the addresses the memory system uses to identify physical storage sites, and program-generated addresses are translated automatically to the corresponding machine addresses.

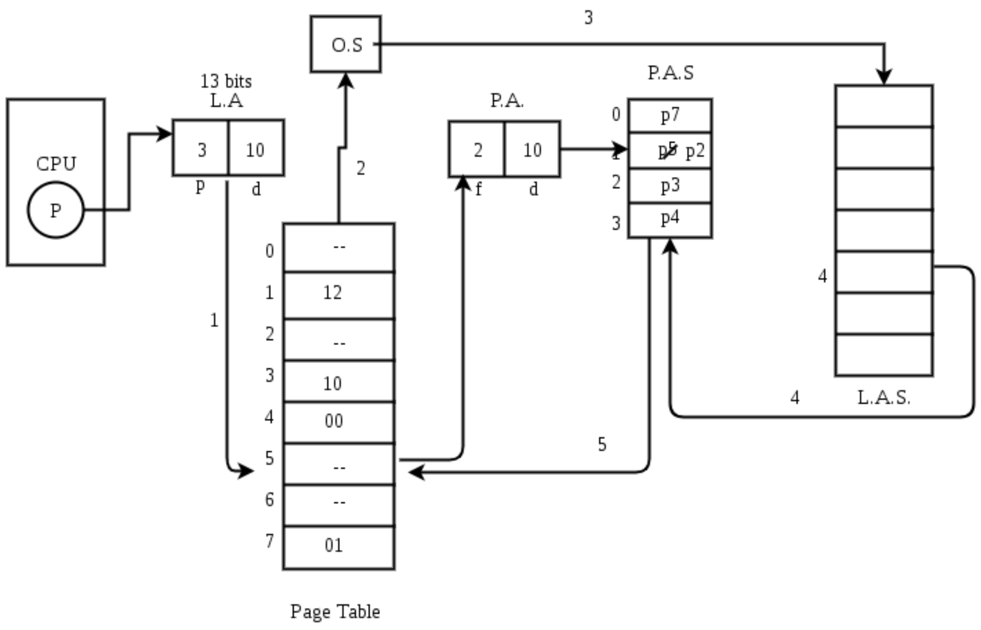
The size of virtual storage is limited by the addressing scheme of the computer system and the amount of secondary memory is available not by the actual number of the main storage locations.

It is a technique that is implemented using both hardware and software. It maps memory addresses used by a program, called virtual addresses, into physical addresses in computer memory.

1. All memory references within a process are logical addresses that are dynamically translated into physical addresses at run time. This means that a process can be swapped in and out of the main memory such that it occupies different places in the main memory at different times during the course of execution.
2. A process may be broken into a number of pieces and these pieces need not be continuously located in the main memory during execution. The combination of dynamic run-time address translation and use of page or segment table permits this.

If these characteristics are present then, it is not necessary that all the pages or segments are present in the main memory during execution. This means that the required pages need to be loaded into memory whenever required. Virtual memory is implemented using Demand Paging or Demand Segmentation.

Demand Paging :
The process of loading the page into memory on demand (whenever page fault occurs) is known as demand paging.
The process includes the following steps :



1. If the CPU tries to refer to a page that is currently not available in the main memory, it generates an interrupt indicating a memory access fault.
2. The OS puts the interrupted process in a blocking state. For the execution to proceed the OS must bring the required page into the memory.
3. The OS will search for the required page in the logical address space.
4. The required page will be brought from logical address space to physical address space. The page replacement algorithms are used for the decision-making of replacing the page in physical address space.
5. The page table will be updated accordingly.
6. The signal will be sent to the CPU to continue the program execution and it will place the process back into the ready state.

Hence whenever a page fault occurs these steps are followed by the operating system and the required page is brought into memory.

Advantages :

- More processes may be maintained in the main memory: Because we are going to load only some of the pages of any particular process, there is room for more processes. This leads to more efficient utilization of the processor because it is more likely that at least one of the more numerous processes will be in the ready state at any particular time.
- A process may be larger than all of the main memory: One of the most fundamental restrictions in programming is lifted. A process larger than the main memory can be executed because of demand paging. The OS itself loads pages of a process in the main memory as required.
- It allows greater multiprogramming levels by using less of the available (primary) memory for each process.

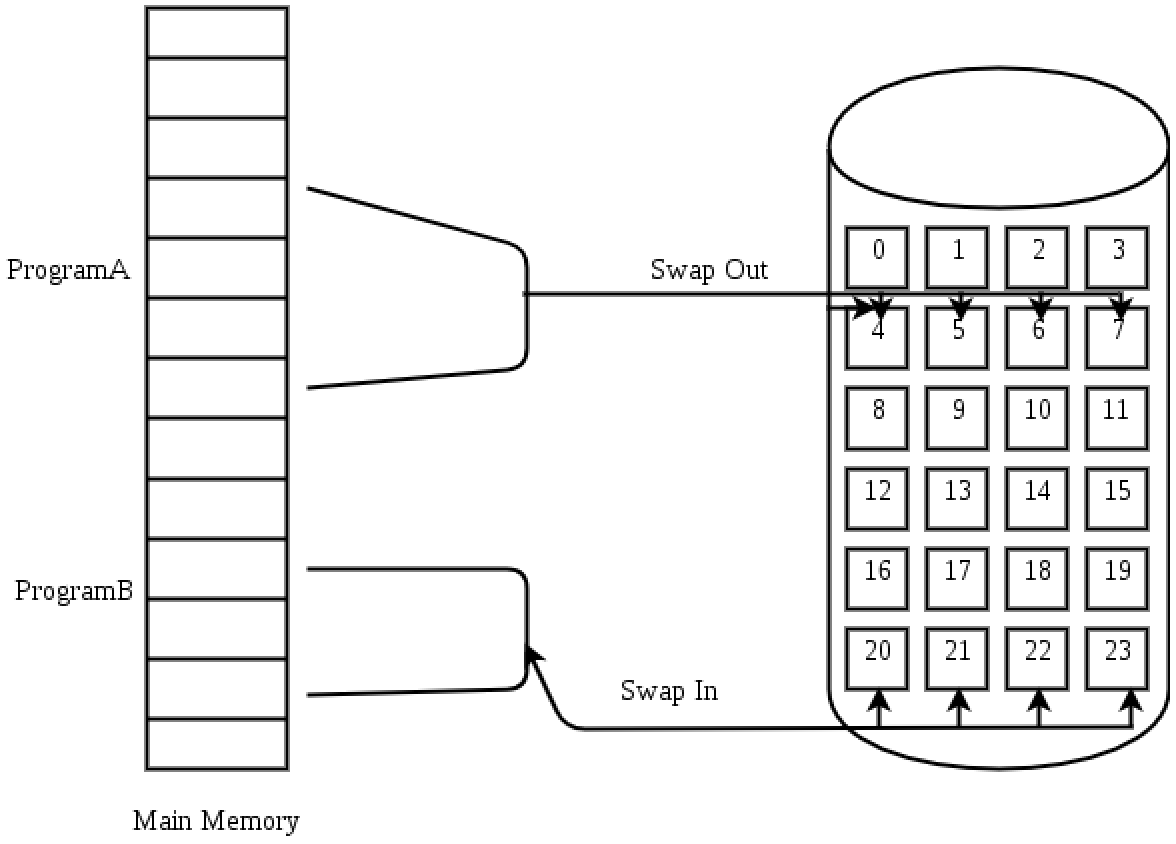
Page Fault Service Time :
The time taken to service the page fault is called page fault service time. The page fault service time includes the time taken to perform all the above six steps.
Let Main memory access time is: m

Page fault service time is: s

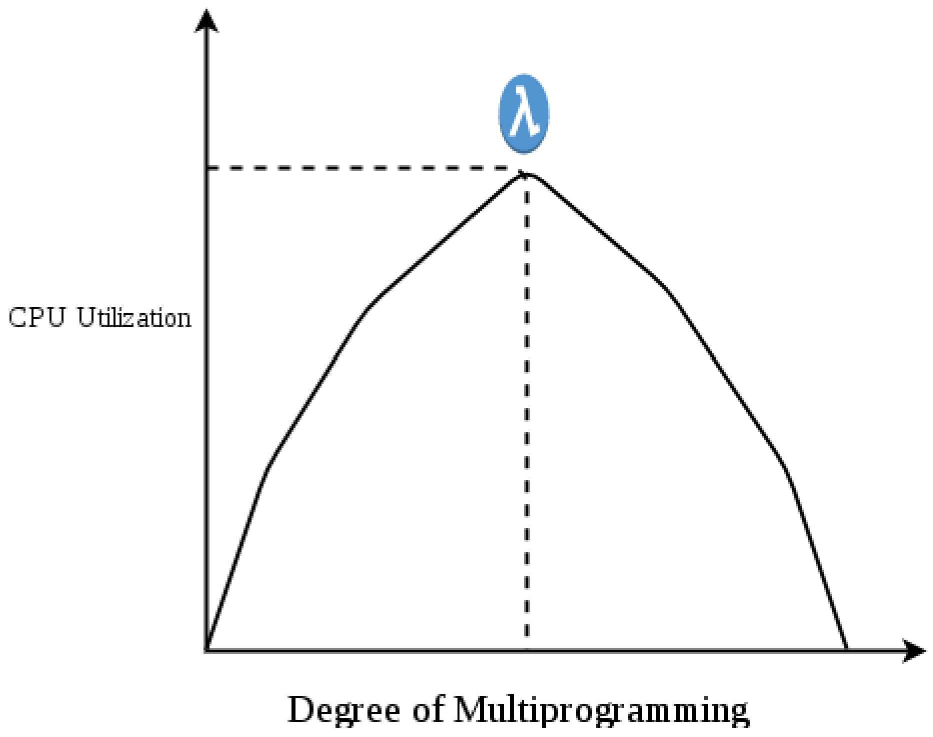
Page fault rate is : p

Then, Effective memory access time = (p*s) + (1-p)*m

Swapping:
Swapping a process out means removing all of its pages from memory, or marking them so that they will be removed by the normal page replacement process. Suspending a process ensures that it is not runnable while it is swapped out. At some later time, the system swaps back the process from the secondary storage to the main memory. When a process is busy swapping pages in and out then this situation is called thrashing.



Thrashing :



At any given time, only a few pages of any process are in the main memory and therefore more processes can be maintained in memory. Furthermore, time is saved because unused pages are not swapped in and out of memory. However, the OS must be clever about how it manages this scheme. In the steady-state practically, all of the main memory will be occupied with process pages, so that the processor and OS have direct access to as many processes as possible. Thus when the OS brings one page in, it must throw another out. If it throws out a page just before it is used, then it will just have to get that page again almost immediately. Too much of this leads to a condition called Thrashing. The system spends most of its time swapping pages rather than executing instructions. So a good page replacement algorithm is required.

In the given diagram, the initial degree of multiprogramming up to some extent of point(lambda), the CPU utilization is very high and the system resources are utilized 100%. But if we further increase the degree of multiprogramming the CPU utilization will drastically fall down and the system will spend more time only on the page replacement and the time is taken to complete the execution of the process will increase. This situation in the system is called thrashing.

Causes of Thrashing :

1. **High degree of multiprogramming** : If the number of processes keeps on increasing in the memory then the number of frames allocated to each process will be decreased. So, fewer frames will be available for each process. Due to this, a page fault will occur more frequently and more CPU time will be wasted in just swapping in and out of pages and the utilization will keep on decreasing.
   For example:
   Let free frames = 400
   Case 1: Number of process = 100
   Then, each process will get 4 frames.
   Case 2: Number of processes = 400
   Each process will get 1 frame.
   Case 2 is a condition of thrashing, as the number of processes is increased, frames per process are decreased. Hence CPU time will be consumed in just swapping pages.

2. **Lacks of Frames**: If a process has fewer frames then fewer pages of that process will be able to reside in memory and hence more frequent swapping in and out will be required. This may lead to thrashing. Hence sufficient amount of frames must be allocated to each process in order to prevent thrashing.

Recovery of Thrashing :

- Do not allow the system to go into thrashing by instructing the long-term scheduler not to bring the processes into memory after the threshold.
- If the system is already thrashing then instruct the mid-term scheduler to suspend some of the processes so that we can recover the system from thrashing.

COPY-ON-WRITE

Copy-on-Write(CoW) is mainly a resource management technique that allows the **parent and child process** to share the same pages of the memory initially. If any process either parent or child modifies the shared page, only then the page is copied.

The CoW is basically a technique of efficiently copying the data resources in the computer system. In this case, if a unit of data is copied but is not modified then **"copy"** can mainly exist as a **reference to the original data.**

But when the **copied data is modified**, then at that time its copy is created(where new bytes are actually written)as suggested from the name of the technique.

The main use of this technique is in the implementation of the fork system call in which it shares the virtual memory/pages of the Operating system.

Recall in the **UNIX(OS)**, the fork() system call is used to create a **duplicate process of the parent process** which is known as the child process.

- The **CoW** technique is used by several Operating systems like Linux, Solaris, and Windows XP.
- The **CoW** technique is an efficient process creation technique as only the pages that are modified are copied.

Free pages in this technique are allocated from a pool of **zeroed-out pages.**

The Copy on Write(CoW) Technique

The main intention behind the **CoW** technique is that whenever a parent process creates a child process both parent and child process initially will share the same pages in the memory.

- These shared pages between parent and child process will be marked as copy-on-write which means that if the parent or child process will attempt to modify the shared pages then a copy of these pages will be created and the modifications will be done only on the copy of pages by that process and it will not affect other processes.

Now its time to take a look at the basic example of this technique:

Let us take an example where Process A creates a new process that is Process B, initially both these processes will share the same pages of the memory.

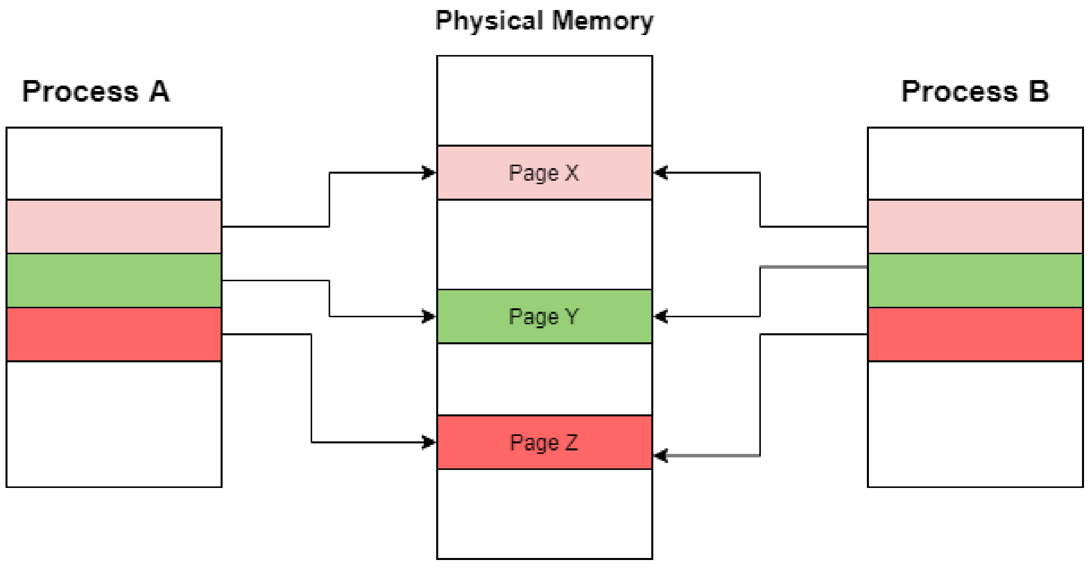
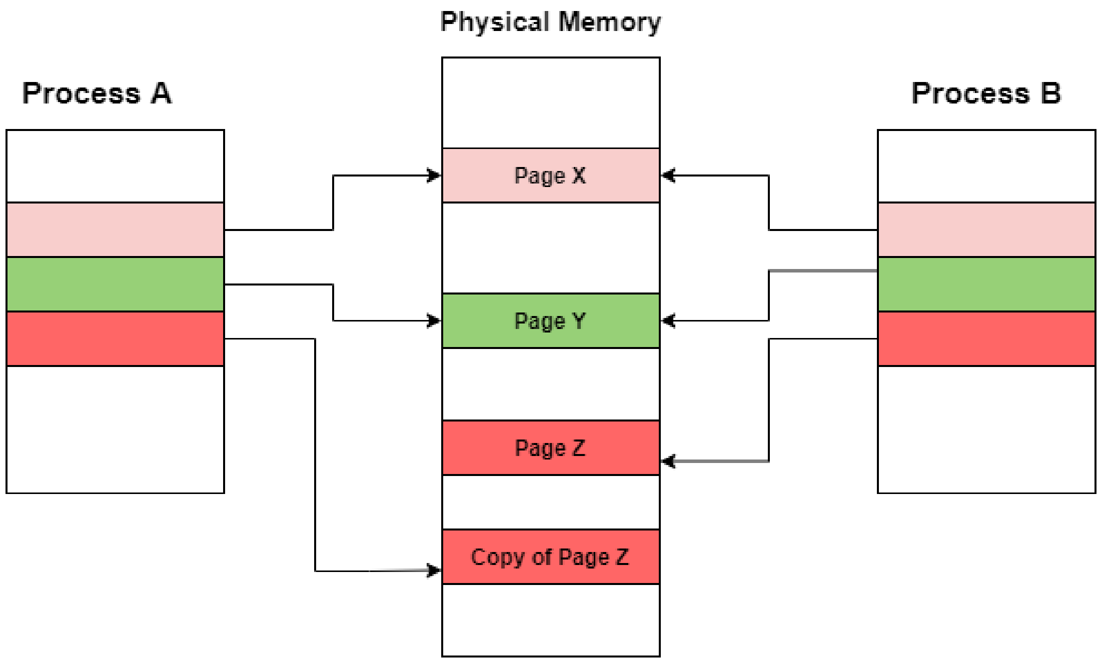


Figure: Above figure indicates parent and child process sharing the same pages

Now, let us assume that process A wants to modify a page in the memory. When the **Copy-on-write(CoW)** technique is used, only those pages that are modified by either process are copied; all the unmodified pages can be easily shared by the parent and child process.



PAGE REPLACEMENT ALGORITHMS IN OPERATING SYSTEMS

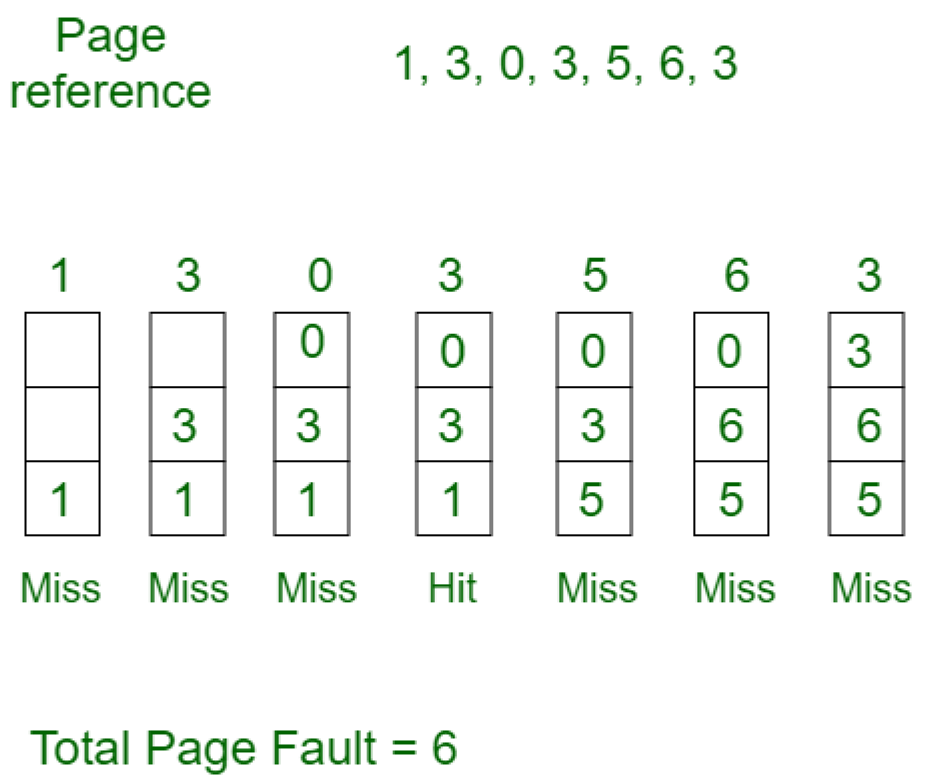
In an operating system that uses paging for memory management, a page replacement algorithm is needed to decide which page needs to be replaced when a new page comes in.

Page Fault: A page fault happens when a running program accesses a memory page that is mapped into the virtual address space but not loaded in physical memory. Since actual physical memory is much smaller than virtual memory, page faults happen. In case of a page fault, Operating System might have to replace one of the existing pages with the newly needed page. Different page replacement algorithms suggest different ways to decide which page to replace. The target for all algorithms is to reduce the number of page faults.

Page Replacement Algorithms:

1. First In First Out (FIFO): This is the simplest page replacement algorithm. In this algorithm, the operating system keeps track of all pages in the memory in a queue, the oldest page is in the front of the queue. When a page needs to be replaced page in the front of the queue is selected for removal.

Example 1: Consider page reference string 1, 3, 0, 3, 5, 6, 3 with 3 page frames. Find the number of page faults.

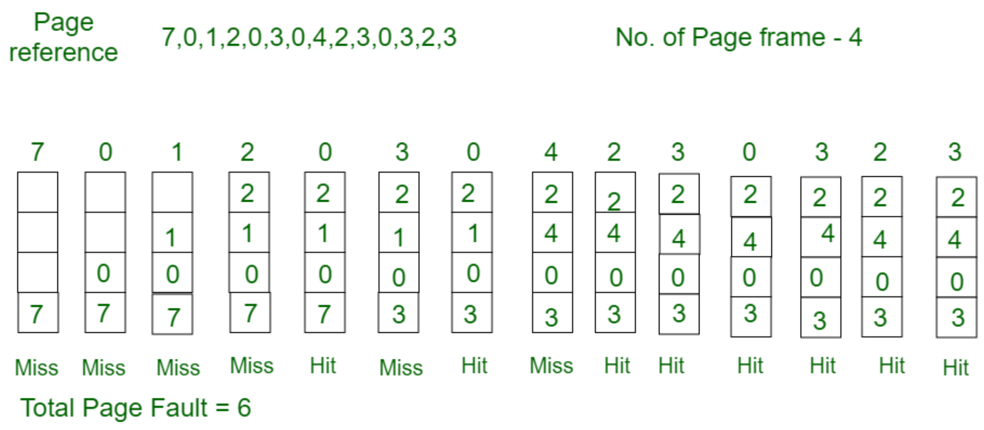


Initially, all slots are empty, so when 1, 3, 0 came they are allocated to the empty slots —> **3 Page Faults.**
when 3 comes, it is already in memory so —> **0 Page Faults.** Then 5 comes, it is not available in memory so it replaces the oldest page slot i.e 1. —>**1 Page Fault.** 6 comes, it is also not available in memory so it replaces the oldest page slot i.e 3 —>**1 Page Fault.** Finally, when 3 come it is not available so it replaces 0 **1 page fault.**

Belady's anomaly proves that it is possible to have more page faults when increasing the number of page frames while using the First in First Out (FIFO) page replacement algorithm. For example, if we consider reference strings 3, 2, 1, 0, 3, 2, 4, 3, 2, 1, 0, 4, and 3 slots, we get 9 total page faults, but if we increase slots to 4, we get 10-page faults.

2. Optimal Page replacement: In this algorithm, pages are replaced which would not be used for the longest duration of time in the future.

Example-2: Consider the page references 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 3 with 4 page frame. Find number of page fault.



Initially, all slots are empty, so when 7 0 1 2 are allocated to the empty slots —> **4 Page faults**

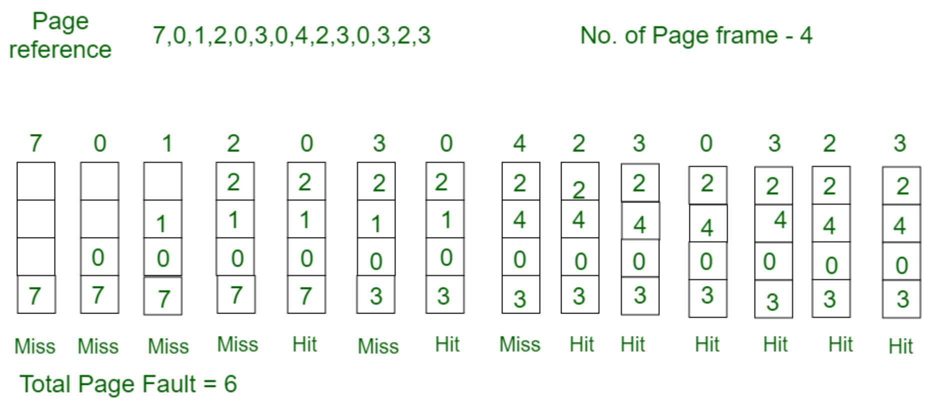
0 is already there so —> **0 Page fault.** when 3 came it will take the place of 7 because it is not used for the longest duration of time in the future.—>**1 Page fault.** 0 is already there so —> **0 Page fault.** 4 will takes place of 1 —> **1 Page Fault.**

Now for the further page reference string —> **0 Page fault** because they are already available in the memory.

Optimal page replacement is perfect, but not possible in practice as the operating system cannot know future requests. The use of Optimal Page replacement is to set up a benchmark so that other replacement algorithms can be analyzed against it.

3. Least Recently Used: In this algorithm, page will be replaced which is least recently used.

Example-3: Consider the page reference string 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 3 with 4 page frames. Find number of page faults.



Here LRU has same number of page fault as optimal but it may differ according to question.

Initially, all slots are empty, so when 7 0 1 2 are allocated to the empty slots —> **4 Page faults**

0 is already their so —> **0 Page fault.** when 3 came it will take the place of 7 because it is

least recently used —>**1 Page fault**
0 is already in memory so —> **0 Page fault**.
4 will takes place of 1 —> **1 Page Fault**
Now for the further page reference string —> **0 Page fault** because they are already available in the memory.

4. Most Recently Used (MRU): In this algorithm, page will be replaced which has been used recently. Belady’s anomaly can occur in this algorithm.