Properties of an Operating System

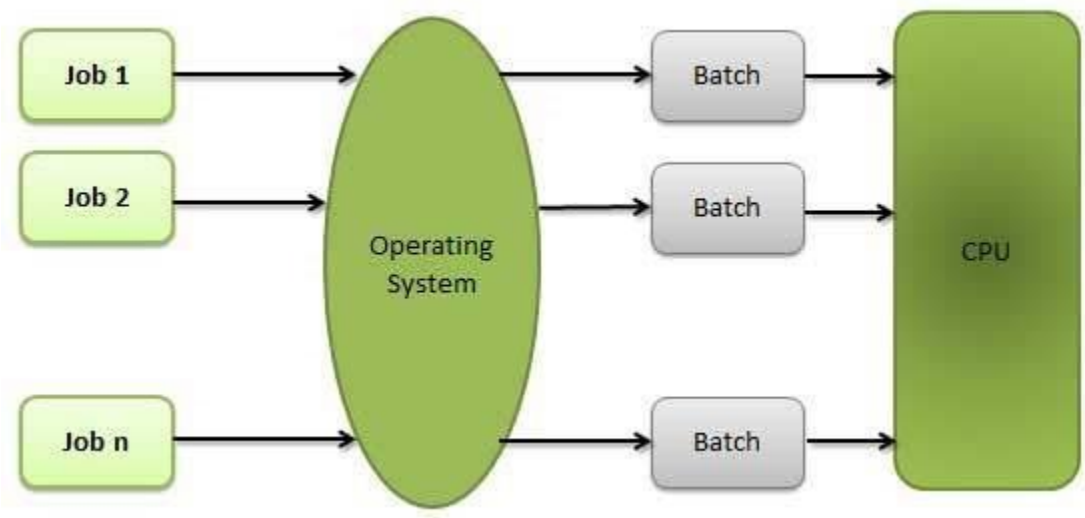
Following are the different properties of an Operating System. This tutorial will explain these properties in detail one by one:

1. Batch processing
2. Multitasking
3. Multiprogramming
4. Interactivity
5. Real Time System
6. Distributed Environment
7. Spooling

Batch processing

Batch processing is a technique in which an Operating System collects the programs and data together in a batch before processing starts. An operating system does the following activities related to batch processing –

- The OS defines a job which has predefined sequence of commands, programs and data as a single unit.
- The OS keeps a number a jobs in memory and executes them without any manual information.
- Jobs are processed in the order of submission, i.e., first come first served fashion.
- When a job completes its execution, its memory is released and the output for the job gets copied into an output spool for later printing or processing.



Advantages

- Batch processing takes much of the work of the operator to the computer.
- Increased performance as a new job get started as soon as the previous job is finished, without any manual intervention.

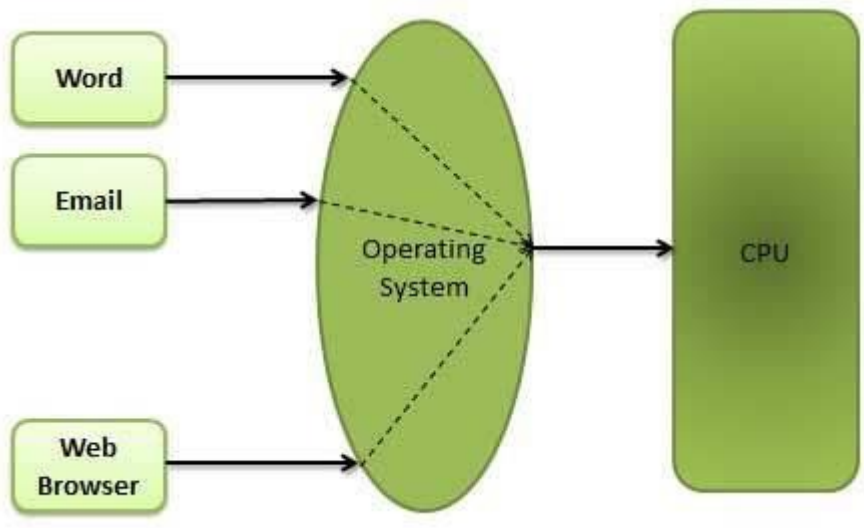
Disadvantages

- Difficult to debug program.
- A job could enter an infinite loop.
- Due to lack of protection scheme, one batch job can affect pending jobs.

Multitasking

Multitasking is when multiple jobs are executed by the CPU simultaneously by switching between them. Switches occur so frequently that the users may interact with each program while it is running. An OS does the following activities related to multitasking –

- The user gives instructions to the operating system or to a program directly, and receives an immediate response.
- The OS handles multitasking in the way that it can handle multiple operations/executes multiple programs at a time.
- Multitasking Operating Systems are also known as Time-sharing systems.
- These Operating Systems were developed to provide interactive use of a computer system at a reasonable cost.
- A time-shared operating system uses the concept of CPU scheduling and multiprogramming to provide each user with a small portion of a time-shared CPU.
- Each user has at least one separate program in memory.



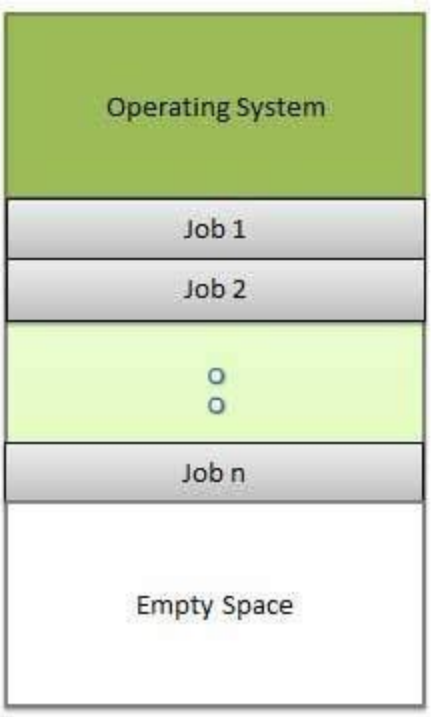
- A program that is loaded into memory and is executing is commonly referred to as a **process**.
- When a process executes, it typically executes for only a very short time before it either finishes or needs to perform I/O.
- Since interactive I/O typically runs at slower speeds, it may take a long time to complete. During this time, a CPU can be utilized by another process.
- The operating system allows the users to share the computer simultaneously. Since each action or command in a time-shared system tends to be short, only a little CPU time is needed for each user.

- As the system switches CPU rapidly from one user/program to the next, each user is given the impression that he/she has his/her own CPU, whereas actually one CPU is being shared among many users.

Multiprogramming

Sharing the processor, when two or more programs reside in memory at the same time, is referred as **multiprogramming**. Multiprogramming assumes a single shared processor. Multiprogramming increases CPU utilization by organizing jobs so that the CPU always has one to execute.

The following figure shows the memory layout for a multiprogramming system.



An OS does the following activities related to multiprogramming.

- The operating system keeps several jobs in memory at a time.
- This set of jobs is a subset of the jobs kept in the job pool.
- The operating system picks and begins to execute one of the jobs in the memory.
- Multiprogramming operating systems monitor the state of all active programs and system resources using memory management programs to ensures that the CPU is never idle, unless there are no jobs to process.

Advantages

- High and efficient CPU utilization.
- User feels that many programs are allotted CPU almost simultaneously.

Disadvantages

- CPU scheduling is required.
- To accommodate many jobs in memory, memory management is required.

Interactivity

Interactivity refers to the ability of users to interact with a computer system. An Operating system does the following activities related to interactivity –

- Provides the user an interface to interact with the system.
- Manages input devices to take inputs from the user. For example, keyboard.
- Manages output devices to show outputs to the user. For example, Monitor.

The response time of the OS needs to be short, since the user submits and waits for the result.

Real Time System

Real-time systems are usually dedicated, embedded systems. An operating system does the following activities related to real-time system activity.

- In such systems, Operating Systems typically read from and react to sensor data.
- The Operating system must guarantee response to events within fixed periods of time to ensure correct performance.

Distributed Environment

A distributed environment refers to multiple independent CPUs or processors in a computer system. An operating system does the following activities related to distributed environment –

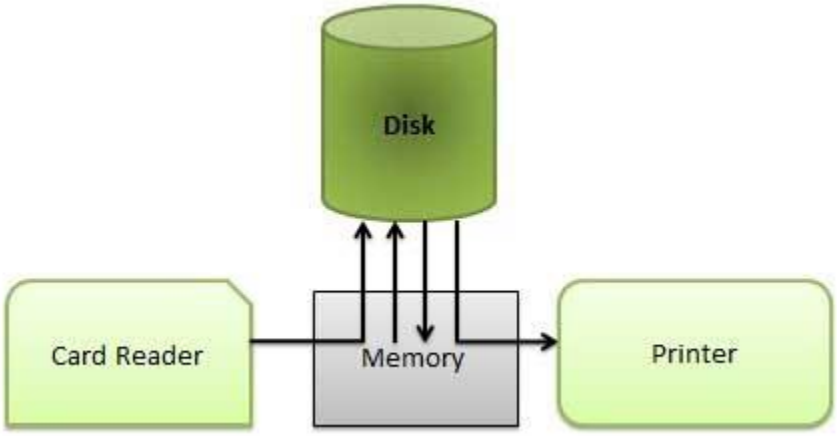
- The OS distributes computation logics among several physical processors.
- The processors do not share memory or a clock. Instead, each processor has its own local memory.
- The OS manages the communications between the processors. They communicate with each other through various communication lines.

Spooling

Spooling is an acronym for simultaneous peripheral operations on line. Spooling refers to putting data of various I/O jobs in a buffer. This buffer is a special area in memory or hard disk which is accessible to I/O devices.

An operating system does the following activities related to spooling –

- Handles I/O device data spooling as devices have different data access rates.
- Maintains the spooling buffer which provides a waiting station where data can rest while the slower device catches up.
- Maintains parallel computation because of spooling process as a computer can perform I/O in parallel fashion. It becomes possible to have the computer read data from a tape, write data to disk and to write out to a tape printer while it is doing its computing task.



Advantages

- The spooling operation uses a disk as a very large buffer.
- Spooling is capable of overlapping I/O operation for one job with processor operations for another job.

greekforgreeks & javatpoint

Difference between Multiprogramming, multitasking, multithreading and multiprocessing

1. **Multiprogramming –** Multiprogramming is known as keeping multiple programs in the main memory at the same time ready for execution.
2. **Multiprocessing –** A computer using more than one CPU at a time.
3. **Multitasking –** Multitasking is nothing but multiprogramming with a Round-robin scheduling algorithm.
4. **Multithreading** is an extension of multitasking.

1. Multi programming –

In a modern computing system, there are usually several concurrent application processes which want to execute. Now it is the responsibility of the Operating System to manage all the processes effectively and efficiently. One of the most important aspects of an Operating System is to multi program. In a computer system, there are multiple processes waiting to be executed, i.e. they are waiting when the CPU will be allocated to them and they begin their execution. These processes are also known as jobs. Now the main memory is too small to accommodate all of these processes or jobs into it. Thus, these processes are initially kept in an area called job pool. This job pool consists of all those processes awaiting allocation of main memory and CPU. CPU selects one job out of all these waiting jobs, brings it from the job pool to main memory and starts executing it. The processor executes one job until it is interrupted by some external factor or it goes for an I/O task. **Non-multi programmed system's working –**

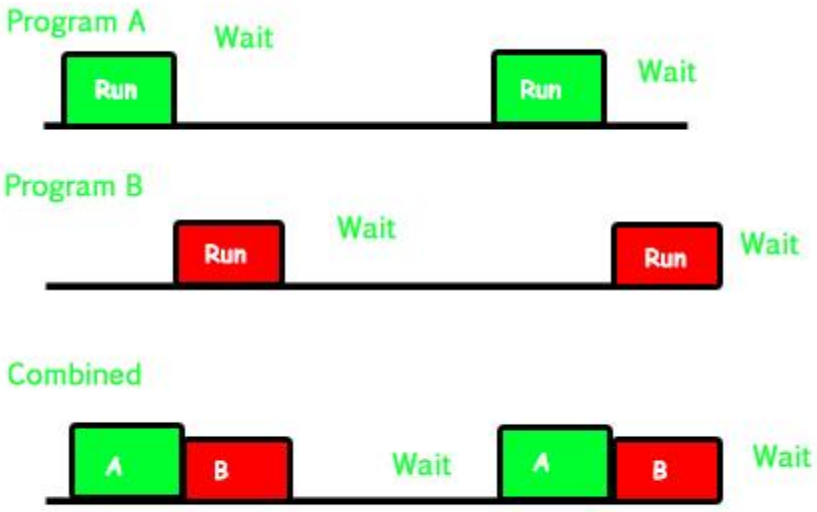
- In a non multi programmed system, As soon as one job leaves the CPU and goes for some other task (say I/O), the CPU becomes idle. The CPU keeps waiting and waiting until this job (which was executing earlier) comes back and resumes its execution with the CPU. So CPU remains free for all this while.
- Now it has a drawback that the CPU remains idle for a very long period of time. Also, other jobs which are waiting to be executed might not get a chance to execute because the CPU is still allocated to the earlier job. This poses a very serious problem that even though other jobs are ready to execute, CPU is not allocated to them as the CPU is allocated to a job which is not even utilizing it (as it is busy in I/O tasks).

- It cannot happen that one job is using the CPU for say 1 hour while the others have been waiting in the queue for 5 hours. To avoid situations like this and come up with efficient utilization of CPU, the concept of multi programming came up.

The main idea of multi programming is to maximize the CPU time. **Multi programmed system's working –**

- In a multi-programmed system, as soon as one job goes for an I/O task, the Operating System interrupts that job, chooses another job from the job pool (waiting queue), gives CPU to this new job and starts its execution. The previous job keeps doing its I/O operation while this new job does CPU bound tasks. Now say the second job also goes for an I/O task, the CPU chooses a third job and starts executing it. As soon as a job completes its I/O operation and comes back for CPU tasks, the CPU is allocated to it.
- In this way, no CPU time is wasted by the system waiting for the I/O task to be completed. Therefore, the ultimate goal of multi programming is to keep the CPU busy as long as there are processes ready to execute. This way, multiple programs can be executed on a single processor by executing a part of a program at one time, a part of another program after this, then a part of another program and so on, hence executing multiple programs. Hence, the CPU never remains idle.

In the image below, program A runs for some time and then goes to waiting state. In the mean time program B begins its execution. So the CPU does not waste its resources and gives program B an opportunity to run.



2. Multiprocessing –

In a uni-processor system, only one process executes at a time. Multiprocessing is the use of two or more CPUs (processors) within a single Computer system. The term also refers to the ability of a system to support more than one processor within a single computer system. Now since there are multiple processors available, multiple processes can be executed at a time. These multi-processors share the computer bus, sometimes the clock, memory and peripheral devices also. **Multi processing system's working –**

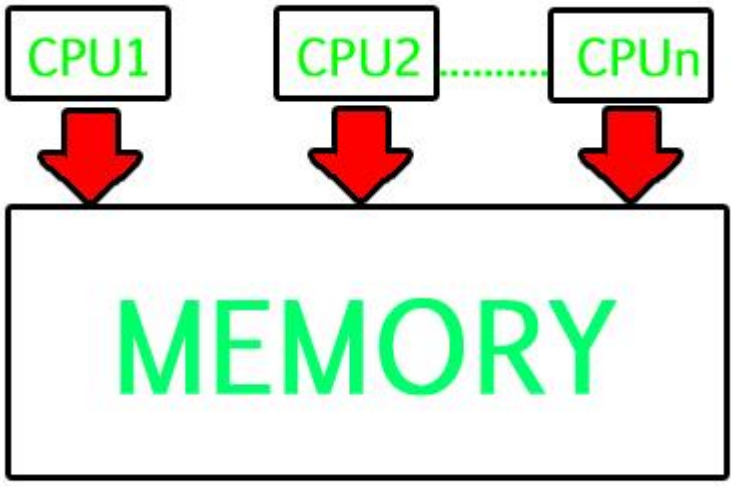
- With the help of multiprocessing, many processes can be executed simultaneously. Say processes P1, P2, P3 and P4 are waiting for execution.

Now in a single processor system, firstly one process will execute, then the other, then the other and so on.

- But with multiprocessing, each process can be assigned to a different processor for its execution. If its a dual-core processor (2 processors), two processes can be executed simultaneously and thus will be two times faster, similarly a quad core processor will be four times as fast as a single processor.

Why use multi processing –

- The main advantage of multiprocessor system is to get more work done in a shorter period of time. These types of systems are used when very high speed is required to process a large volume of data. Multi processing systems can save money in comparison to single processor systems because the processors can share peripherals and power supplies.
- It also provides increased reliability in the sense that if one processor fails, the work does not halt, it only slows down. e.g. if we have 10 processors and 1 fails, then the work does not halt, rather the remaining 9 processors can share the work of the 10th processor. Thus the whole system runs only 10 percent slower, rather than failing altogether.



Multiprocessing refers to the hardware (i.e., the CPU units) rather than the software (i.e., running processes). If the underlying hardware provides more than one processor then that is multiprocessing. It is the ability of the system to leverage multiple processors' computing power. **Difference between Multi programming and Multi processing –**

- A System can be both multi programmed by having multiple programs running at the same time and multiprocessing by having more than one physical processor. The difference between multiprocessing and multi programming is that Multiprocessing is basically executing multiple processes at the same time on multiple processors, whereas multi programming is keeping several programs in main memory and executing them concurrently using a single CPU only.
- Multiprocessing occurs by means of parallel processing whereas Multi programming occurs by switching from one process to other (phenomenon called as context switching).

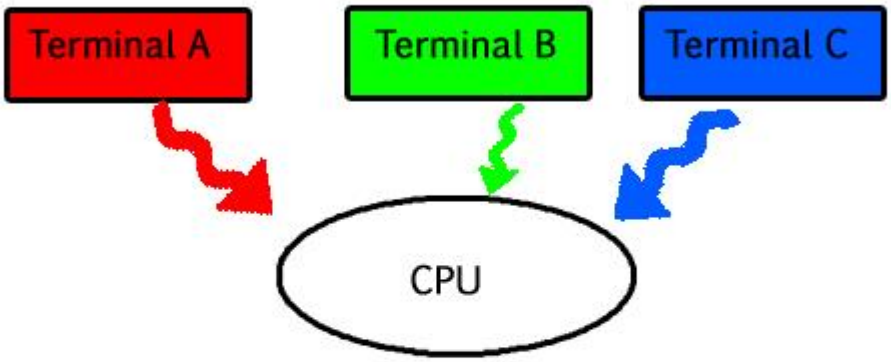
3. Multitasking –

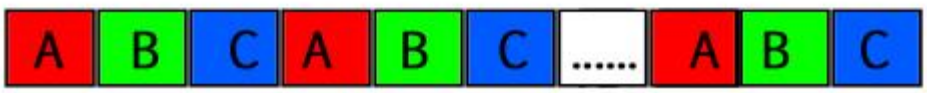
As the name itself suggests, multitasking refers to execution of multiple tasks (say processes, programs, threads etc.) at a time. In the modern operating systems, we are able to play MP3 music, edit documents in Microsoft Word, surf the Google Chrome all

simultaneously, this is accomplished by means of multi tasking. Multitasking is a logical extension of multi programming. The major way in which multitasking differs from multi programming is that multi programming works solely on the concept of context switching whereas multitasking is based on time sharing alongside the concept of context switching. **Multi tasking system's working –**

- In a time sharing system, each process is assigned some specific quantum of time for which a process is meant to execute. Say there are 4 processes P1, P2, P3, P4 ready to execute. So each of them are assigned some time quantum for which they will execute e.g time quantum of 5 nanoseconds (5 ns). As one process begins execution (say P2), it executes for that quantum of time (5 ns). After 5 ns the CPU starts the execution of the other process (say P3) for the specified quantum of time.
- Thus the CPU makes the processes to share time slices between them and execute accordingly. As soon as time quantum of one process expires, another process begins its execution.
- Here also basically a context switch is occurring but it is occurring so fast that the user is able to interact with each program separately while it is running. This way, the user is given the illusion that multiple processes/ tasks are executing simultaneously. But actually only one process/ task is executing at a particular instant of time. In multitasking, time sharing is best manifested because each running process takes only a fair quantum of the CPU time.

In a more general sense, multitasking refers to having multiple programs, processes, tasks, threads running at the same time. This term is used in modern operating systems when multiple tasks share a common processing resource (e.g., CPU and Memory).





- As depicted in the above image, At any time the CPU is executing only one task while other tasks are waiting for their turn. The illusion of parallelism is achieved when the CPU is reassigned to another task. i.e all the three tasks A, B and C are appearing to occur simultaneously because of time sharing.

- So for multitasking to take place, firstly there should be multiprogramming i.e. presence of multiple programs ready for execution. And secondly the concept of time sharing.

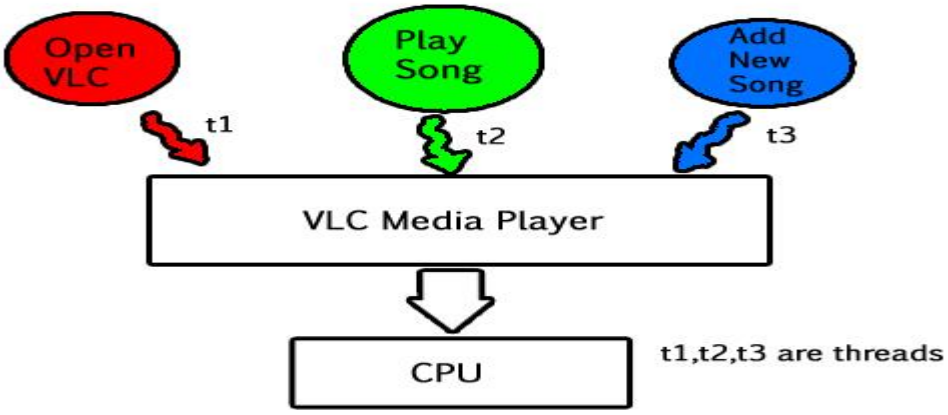
4. Multi threading –

A thread is a basic unit of CPU utilization. Multi-threading is an execution model that allows a single process to have multiple code segments (i.e., threads) running concurrently within the "context" of that process. e.g. VLC media player, where one thread is used for opening the VLC media player, one thread for playing a particular song and another thread for adding new songs to the playlist. Multi threading is the ability of a process to manage its use by more than one user at a time and to manage multiple requests by the same user without having to have multiple copies of the program. **Multi threading system's working – Example 1 –**

- Say there is a web server which processes client requests. Now if it executes as a single threaded process, then it will not be able to process multiple requests at a time. Firstly one client will make its request and finish its execution and only then, the server will be able to process another client request. This is really costly, time consuming and tiring task. To avoid this, multi threading can be made use of.
- Now, whenever a new client request comes in, the web server simply creates a new thread for processing this request and resumes its execution to hear more client requests. So the web server has the task of listening to new client requests and creating threads for each individual request. Each newly created thread processes one client request, thus reducing the burden on web server.

Example 2 –

- We can think of threads as child processes that share the parent process resources but execute independently. Now take the case of a GUI. Say we are performing a calculation on the GUI (which is taking very long time to finish). Now we can not interact with the rest of the GUI until this command finishes its execution. To be able to interact with the rest of the GUI, this command of calculation should be assigned to a separate thread. So at this point of time, 2 threads will be executing i.e. one for calculation, and one for the rest of the GUI. Hence here in a single process, we used multiple threads for multiple functionality.



The image below completely describes the VLC player example:

Advantages of Multi threading –

- Benefits of Multi threading include increased responsiveness. Since there are multiple threads in a program, so if one thread is taking too long to execute or if

it gets blocked, the rest of the threads keep executing without any problem. Thus the whole program remains responsive to the user by means of remaining threads.

- Another advantage of multi threading is that it is less costly. Creating brand new processes and allocating resources is a time consuming task, but since threads share resources of the parent process, creating threads and switching between them is comparatively easy. Hence multi threading is the need of modern Operating Systems.

Go through the following link for more precise understasnding: https://www.javatpoint.com/multiprogramming-vs-multiprocessing-vs-multitasking-vs-multithreading

Difference between 32-bit and 64-bit operating systems

In computing, there are two types of processors existing, i.e., 32-bit and 64-bit processors. These types of processors tell us how much memory a processor can access from a CPU register. For instance,

A 32-bit system can access 2^{32} different memory addresses, i.e 4 GB of RAM or physical memory ideally, it can access more than 4 GB of RAM also.
A 64-bit system can access 2^{64} different 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.

Most computers made in the 1990s and early 2000s were 32-bit machines. The CPU register stores memory addresses, which is how the processor accesses data from RAM. One bit in the register can reference an individual byte in memory, so a **32-bit** system can address a maximum of 4 GB (4,294,967,296 bytes) of RAM. *The actual limit is often less than around **3.5 GB** since part of the register is used to store other temporary values besides memory addresses*. Most computers released over the past two decades were built on a 32-bit architecture, hence most operating systems were designed to run on a 32-bit processor.

A **64-bit** register can theoretically reference 18,446,744,073,709,551,616 bytes, or 17,179,869,184 GB (16 exabytes) of memory. This is several million times more than an average workstation would need to access. What's important is that a 64-bit computer (which means it has a 64-bit processor) can access more than 4 GB of RAM. If a computer has 8 GB of RAM, it better has a 64-bit processor. Otherwise, at least 4 GB of the memory will be inaccessible by the CPU.

A major difference between **32-bit processors and 64-bit processors** is the number of calculations per second they can perform, which affects the speed at which they can complete tasks. 64-bit processors can come in dual-core, quad-core**,** six-core**, and** eight-core **versions** for home computing. Multiple cores allow for an increased number of calculations per second that can be performed, which can increase the processing power and help make a computer run faster. Software programs that require many calculations to function smoothly can operate faster and more efficiently on the multi-core 64-bit processors, for the most part.

Advantages of 64-bit over 32-bit

- Using 64-bit one can do a lot in multi-tasking, user can easily switch between various applications without any windows hanging problems.

- Gamers can easily play High graphical games like Modern Warfare, GTA V, or use high-end software like Photoshop or CAD which takes a lot of memory since it makes multi-tasking with big software, easy and efficient for users. However, upgrading the video card instead of getting a 64-bit processor would be more beneficial.

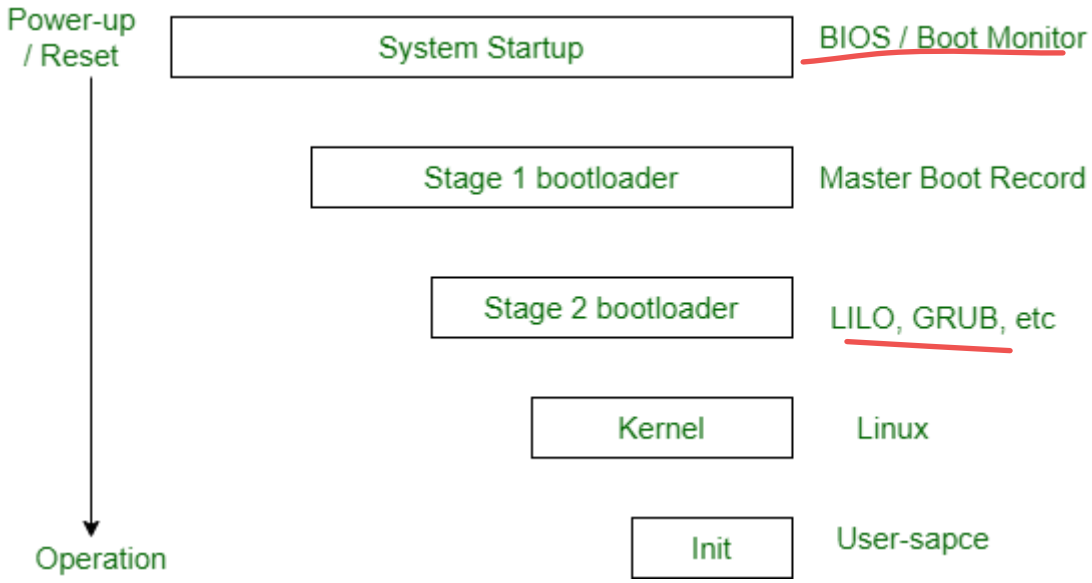
Note:

- A computer with a 64-bit processor can have a 64-bit or 32-bit version of an operating system installed. However, with a 32-bit operating system, the 64-bit processor would not run at its full capability.
- On a computer with a 64-bit processor, we can't run a 16-bit legacy program. Many 32-bit programs will work with a 64-bit processor and operating system, but some older 32-bit programs may not function properly, or at all, due to limited or no compatibility.

What happens when we turn on a computer?

A computer without a program running is just an inert hunk of electronics. The first thing a computer has to do when it is turned on is to start up a special program called an operating system. The operating system's job is to help other computer programs work by handling the messy details of controlling the computer's hardware.

An overview of the boot process



The boot process is something that happens every time you turn your computer on. You don't really see it, because it happens so fast. You press the power button and come back a few sec (or minutes if on slow storage like HDD) later and Windows 10, or Windows 11, or whatever Operating System you use is all loaded.

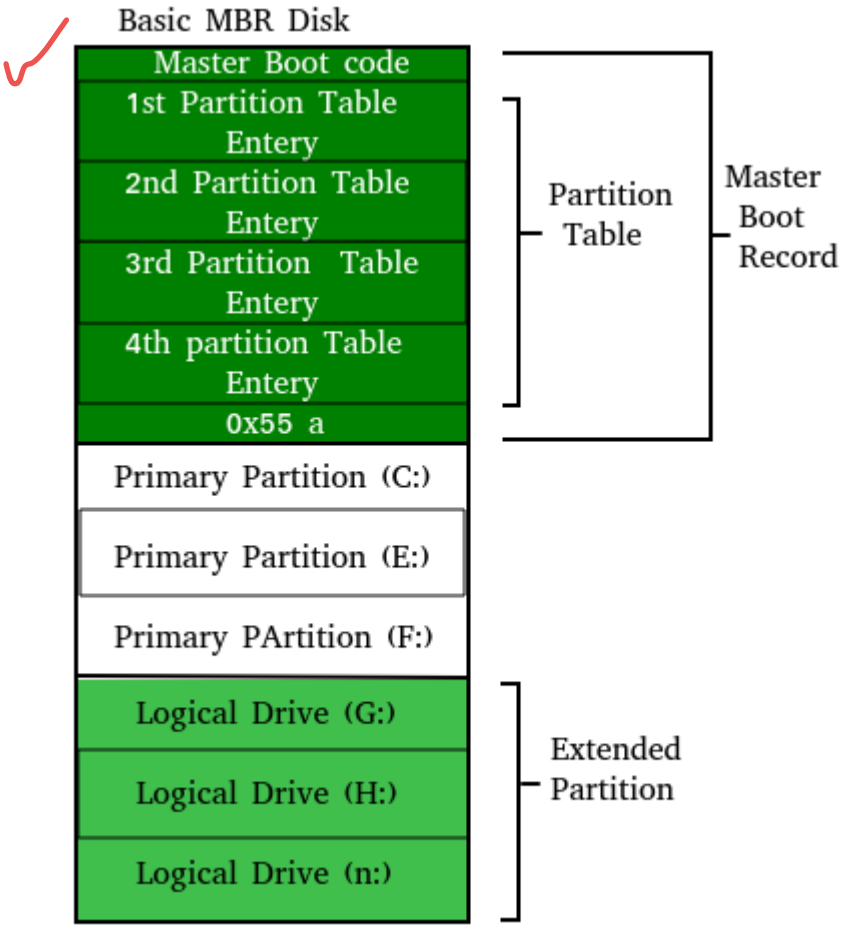
The BIOS chip tells it to look in a fixed place, usually on the lowest-numbered hard disk (the boot disk) for a special program called a boot loader (under Linux the boot loader is called Grub or LILO). The boot loader is pulled into memory and started. The bootloader's job is to start the real operating system.

Functions of BIOS

1. POST (Power On Self Test): The Power On Self Test happens each time you turn your computer on. It sounds complicated and that's because it kind of is. Your computer does so much when it's turned on and this is just part of that.

- It initializes the various hardware devices.
- It is an important process to ensure that all the devices operate smoothly without any conflicts. BIOSes following ACPI create tables describing the devices in the computer.
- The POST first checks the bios and then tests the CMOS RAM.
- If there is no problem with this then POST continues to check the CPU, hardware devices such as the Video Card, and the secondary storage devices such as the Hard Drive, Floppy Drives, Zip Drive, or CD/DVD Drives.
- If some errors are found then an error message is displayed on the screen or a number of beeps are heard.
- These beeps are known as POST beep codes.

2. Master Boot Record: The Master Boot Record (MBR) is a special boot sector at the beginning of the disk. The MBR contains the code that loads the rest of OS, known as bootloader. This complicated process (called the Boot Process) starts with the POST (Power On Self Test) and ends when the Bios searches for the MBR on the Hard Drive, which is generally located in the first sector, first head, first cylinder (cylinder 0, head 0, sector 1).
A typical structure looks like this:



The bootstrap loader is stored in the computer's EPROM, ROM, or another non-volatile memory. When the computer is turned on or restarted, it first performs the power-on-self-

test, also known as POST. If the POST is successful and no issues are found, the bootstrap loader will load the operating system for the computer into memory. The computer will then be able to quickly access, load, and run the operating system.

3. init: init is the last step of the kernel boot sequence. It looks for the file */etc/inittab* to see if there is an entry for *initdefault*. It is used to determine the initial run level of the system. A run-level is used to decide the initial state of the operating system.
Some of the run levels are:

- **Level 0:** System Halt.
- **Level 1:** Single user mode.
- **Level 2:** Full multiuser mode without network.
- **Level 3:** Full multiuser mode with network.
- **Level 4:** user definable.
- **Level 5:** Full multiuser mode with network and X display manager.
- **Level 6:** Reboot.

The above design of init is called SysV- pronounced as System five. Several other implementations of init have been written now. Some of the popular implementations are systemd and upstart. Upstart is being used by ubuntu since 2006. More details of the upstart can be found here.
The next step of init is to start up various daemons that support networking and other services. X server daemon is one of the most important daemons. It manages the display, keyboard, and mouse. When X server daemon is started you see a Graphical Interface and a login screen is displayed.