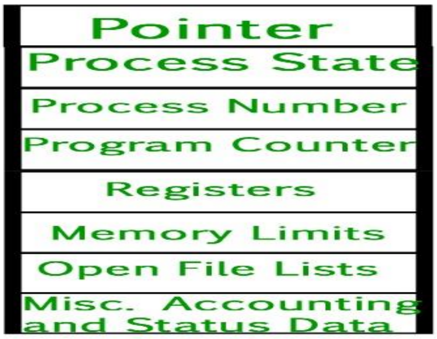
Chapter 2:- Processes and Threads

*** Process Table and Process Control Block (PCB)

- While creating a process, the operating system performs several operations. To identify the processes, it assigns a process identification number (PID) to each process. As the operating system supports multi-programming, it needs to keep track of all the processes. For this task, the process control block (PCB) is used to track the process's execution status. Each block of memory contains information about the process state, program counter, stack pointer, status of opened files, scheduling algorithms, etc.

  All this information is required and must be saved when the process is switched from one state to another. When the process makes a transition from one state to another, the operating system must update information in the process's PCB. A process control block (PCB) contains information about the process, i.e. registers, quantum, priority, etc. The process table is an array of PCBs, that means logically contains a PCB for all of the current processes in the system.



Process Control Block

1. **Pointer:** It is a stack pointer that is required to be saved when the process is switched from one state to another to retain the current position of the process.
2. **Process state:** It stores the respective state of the process.
3. **Process number:** Every process is assigned a unique id known as process ID or PID which stores the process identifier.
4. **Program counter:** It stores the counter,**:** which contains the address of the next instruction that is to be executed for the process.
5. **Register:** Registers in the PCB, it is a data structure. When a processes is running and it's time slice expires, the current value of process specific registers would be stored in the PCB and the process would be swapped out. When the process is scheduled to be run, the register values is read from the PCB and written to the CPU registers. This is the main purpose of the registers in the PCB.
6. **Memory limits:** This field contains the information about memory management system used by the operating system. This may include page tables, segment tables, etc.
7. **Open files list :** This information includes the list of files opened for a process.

Additional Points to Consider for Process Control Block (PCB)

- **Interrupt handling:** The PCB also contains information about the interrupts that a process may have generated and how they were handled by the operating system.
- **Context switching:** The process of switching from one process to another is called context switching. The PCB plays a crucial role in context switching by saving the state

of the current process and restoring the state of the next process.

- **Real-time systems:** Real-time operating systems may require additional information in the PCB, such as deadlines and priorities, to ensure that time-critical processes are executed in a timely manner.
- **Virtual memory management:** The PCB may contain information about a process's virtual memory management, such as page tables and page fault handling.
- **Inter-process communication:** The PCB can be used to facilitate inter-process communication by storing information about shared resources and communication channels between processes.
- **Fault tolerance:** Some operating systems may use multiple copies of the PCB to provide fault tolerance in case of hardware failures or software errors.

Advantages-

1. **Efficient process management:** The process table and PCB provide an efficient way to manage processes in an operating system. The process table contains all the information about each process, while the PCB contains the current state of the process, such as the program counter and CPU registers.
2. **Resource management:** The process table and PCB allow the operating system to manage system resources, such as memory and CPU time, efficiently. By keeping track of each process's resource usage, the operating system can ensure that all processes have access to the resources they need.
3. **Process synchronization:** The process table and PCB can be used to synchronize processes in an operating system. The PCB contains information about each process's

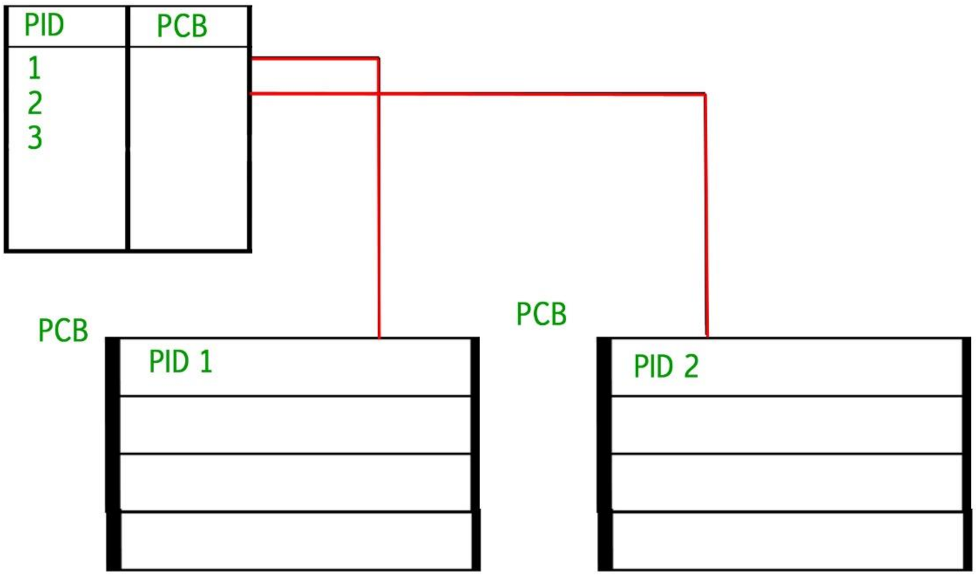
synchronization state, such as its waiting status and the resources it is waiting for.

4. **Process scheduling:** The process table and PCB can be used to schedule processes for execution. By keeping track of each process’s state and resource usage, the operating system can determine which processes should be executed next
.

Disadvantages-

1. **Overhead:** The process table and PCB can introduce overhead and reduce system performance. The operating system must maintain the process table and PCB for each process, which can consume system resources.
2. **Complexity:** The process table and PCB can increase system complexity and make it more challenging to develop and maintain operating systems. The need to manage and synchronize multiple processes can make it more difficult to design and implement system features and ensure system stability.
3. **Scalability:** The process table and PCB may not scale well for large-scale systems with many processes. As the number of processes increases, the process table and PCB can become larger and more difficult to manage efficiently.
4. **Security:** The process table and PCB can introduce security risks if they are not implemented correctly. Malicious programs can potentially access or modify the process table and PCB to gain unauthorized access to system resources or cause system instability.
5. **Miscellaneous accounting and status data –** This field includes information about the amount of CPU used, time constraints, jobs or process number, etc. The process control block stores the register content also known as execution content of the processor when it was blocked

from running. This execution content architecture enables the operating system to restore a process's execution context when the process returns to the running state. When the process makes a transition from one state to another, the operating system updates its information in the process's PCB. The operating system maintains pointers to each process's PCB in a process table so that it can access the PCB quickly.



Process table and process control block

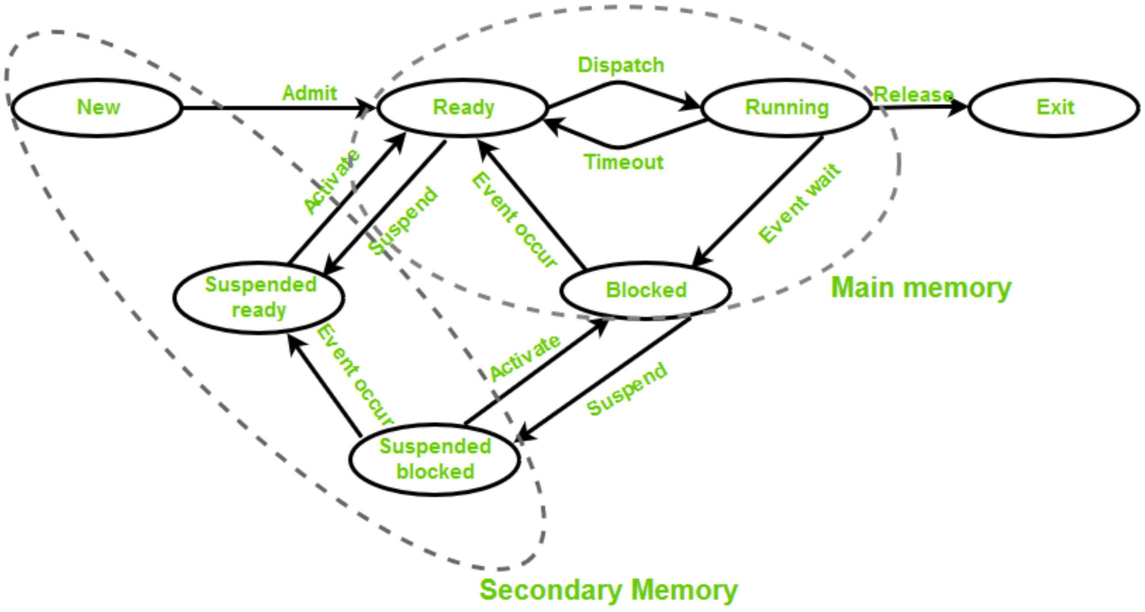
Process States in Operating System

The states of a process are as follows:

- **New (Create):** In this step, the process is about to be created but not yet created. It is the program that is present in secondary memory that will be picked up by OS to create the process.
- **Ready:** New -> Ready to run. After the creation of a process, the process enters the ready state i.e. the process is loaded into the main memory. The process here is ready to run and is waiting to get the CPU time for its execution. Processes that are ready for execution by the CPU are

maintained in a queue called ready queue for ready processes.

- **Run:** The process is chosen from the ready queue by the CPU for execution and the instructions within the process are executed by any one of the available CPU cores.
- **Blocked or Wait:** Whenever the process requests access to I/O or needs input from the user or needs access to a critical region(the lock for which is already acquired) it enters the blocked or waits state. The process continues to wait in the main memory and does not require CPU. Once the I/O operation is completed the process goes to the ready state.
- **Terminated or Completed:** Process is killed as well as PCB is deleted. The resources allocated to the process will be released or deallocated.
- **Suspend Ready:** Process that was initially in the ready state but was swapped out of main memory(refer to Virtual Memory topic) and placed onto external storage by the scheduler is said to be in suspend ready state. The process will transition back to a ready state whenever the process is again brought onto the main memory.
- **Suspend wait or suspend blocked:** Similar to suspend ready but uses the process which was performing I/O operation and lack of main memory caused them to move to secondary memory. When work is finished it may go to suspend ready.



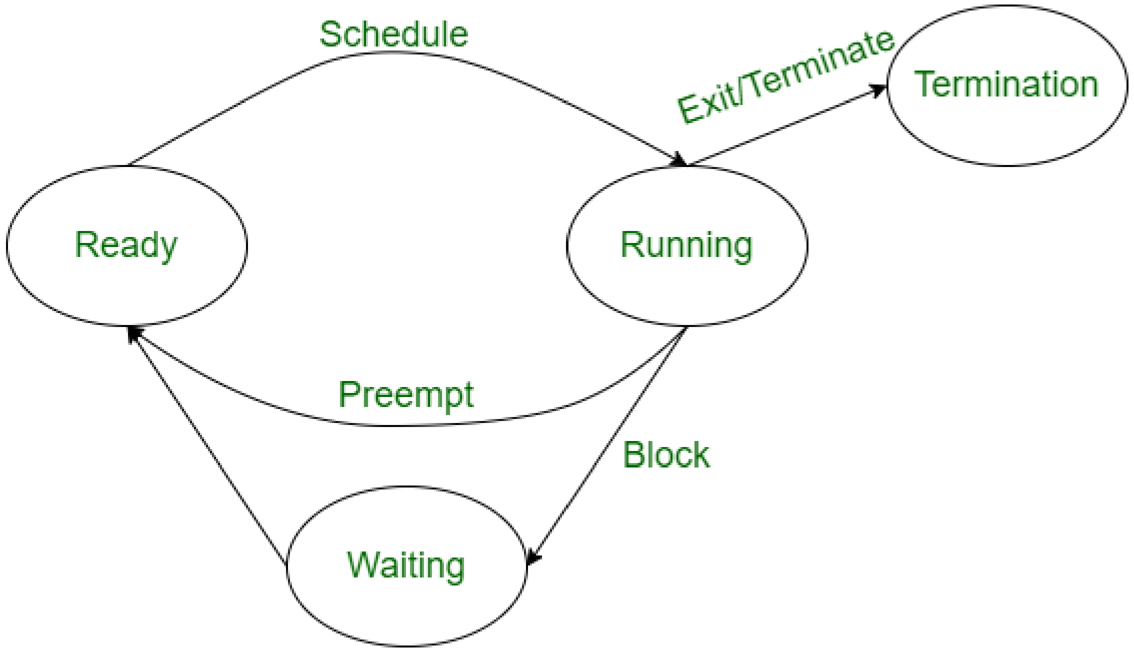
Types of Schedulers

1. **Long-term – performance:** Decides how many processes should be made to stay in the ready state. This decides the degree of multiprogramming. Once a decision is taken it lasts for a long time which also indicates that it runs infrequently. Hence it is called a long-term scheduler.
2. **Short-term – Context switching time:** Short-term scheduler will decide which process is to be executed next and then it will call the dispatcher. A dispatcher is a software that moves the process from ready to run and vice versa. In other words, it is context switching. It runs frequently. Short-term scheduler is also called CPU scheduler.
3. **Medium-term – Swapping time:** Suspension decision is taken by the medium-term scheduler. The medium-term scheduler is used for swapping which is moving the process from main memory to secondary and vice versa.

The swapping is done to reduce degree of multiprogramming.

Operation on a Process:-

The execution of a process is a complex activity. It involves various operations. Following are the operations that are performed while execution of a process:



Creation

This is the initial step of the process execution activity. Process creation means the construction of a new process for execution. This might be performed by the system, the user, or the old process itself. There are several events that lead to the process creation. Some of the such events are the following:

1. When we start the computer, the system creates several background processes.

2. A user may request to create a new process.
3. A process can create a new process itself while executing.
4. The batch system takes initiation of a batch job.

Scheduling/Dispatching

The event or activity in which the state of the process is changed from ready to run. It means the operating system puts the process from the ready state into the running state. Dispatching is done by the operating system when the resources are free or the process has higher priority than the ongoing process. There are various other cases in which the process in the running state is preempted and the process in the ready state is dispatched by the operating system.

Blocking

When a process invokes an input-output system call that blocks the process, and operating system is put in block mode. Block mode is basically a mode where the process waits for input-output. Hence on the demand of the process itself, the operating system blocks the process and dispatches another process to the processor. Hence, in process-blocking operations, the operating system puts the process in a 'waiting' state.

Preemption

When a timeout occurs that means the process hadn't been terminated in the allotted time interval and the next process is ready to execute, then the operating system preempts the process. This operation is only valid where CPU scheduling supports preemption. Basically, this happens in priority scheduling where on the incoming of high priority process the ongoing process is preempted. Hence, in process preemption operation, the operating system puts the process in a ‘ready’ state.

Process Termination

Process termination is the activity of ending the process. In other words, process termination is the relaxation of computer resources taken by the process for the execution. Like creation, in termination also there may be several events that may lead to the process of termination. Some of them are:

1. The process completes its execution fully and it indicates to the OS that it has finished.
2. The operating system itself terminates the process due to service errors.
3. There may be a problem in hardware that terminates the process.
4. One process can be terminated by another process

What is Thread in Operating Systems?

In a process, a thread refers to a single sequential activity being executed. these activities are also known as thread of execution or thread control. Now, any operating system process can execute a thread. we can say, that a process can have multiple threads

Components of Threads

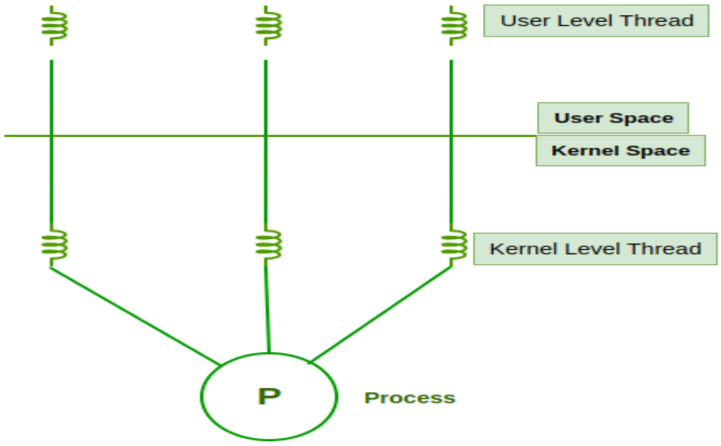
These are the basic components of the Operating System.

- Stack Space
- Register Set
- Program Counter.

Types of Thread in Operating System

Threads are of two types. These are described below.

- User Level Thread
- Kernel Level Thread



1. User Level Threads

User Level Thread is a type of thread that is not created using system calls. The kernel has no work in the management of user-level threads. User-level threads can be easily implemented by the user. In case when user-level threads are single-handed processes, kernel-level thread manages them. Let’s look at the advantages and disadvantages of User-Level Thread.

Advantages of User-Level Threads

- Implementation of the User-Level Thread is easier than Kernel Level Thread.
- Context Switch Time is less in User Level Thread.
- User-Level Thread is more efficient than Kernel-Level Thread.

- Because of the presence of only Program Counter, Register Set, and Stack Space, it has a simple representation.

Disadvantages of User-Level Threads

- There is a lack of coordination between Thread and Kernel.
- In case of a page fault, the whole process can be blocked.

2. Kernel Level Threads

A kernel Level Thread is a type of thread that can recognize the Operating system easily. Kernel Level Threads has its own thread table where it keeps track of the system. The operating System Kernel helps in managing threads. Kernel Threads have somehow longer context switching time. Kernel helps in the management of threads.

Advantages of Kernel-Level Threads

- It has up-to-date information on all threads.
- Applications that block frequency are to be handled by the Kernel-Level Threads.
- Whenever any process requires more time to process, Kernel-Level Thread provides more time to it.

Disadvantages of Kernel-Level threads

- Kernel-Level Thread is slower than User-Level Thread.
- Implementation of this type of thread is a little more complex than a user-level thread.

*****************Thank You************************

Prepared by Prof.Y.J.Patangade
Department of BSC.Computer Science
Hutatma Rajgurumahavidyalay

ref. https://www.geeksforgeeks.org and TYBCS OS-I Text book