ILAHIA COLLEGE OF ENGINEERING AND TECHNOLOGY

Department Of Computer Science and Engineering

Operating Systems (S4CSE)

Module II

Process Management: Process Concept – Processes-States – Process Control Block – Threads-Scheduling – Queues – Schedulers – Context Switching. Process Creation and Termination.

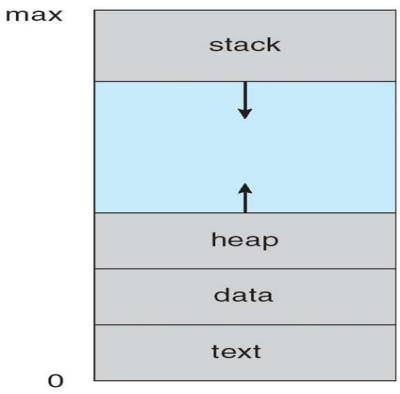
Inter Process Communication: Shared Memory, Message Passing, Pipes

1. Process Concept

A process is a program in execution. The execution of a process must progress in a sequential fashion. A process includes:

- program counter: current activity is represented by the value of program counter and the content of processor's registers
- stack: contain temporary data
- data section: contain global variables
- heap: memory that is dynamically allocated during process run time

Program is a passive entity and process is an active entity ,with a program counter specifying the next instruction to execute. A Program becomes process when executable file loaded into memory.



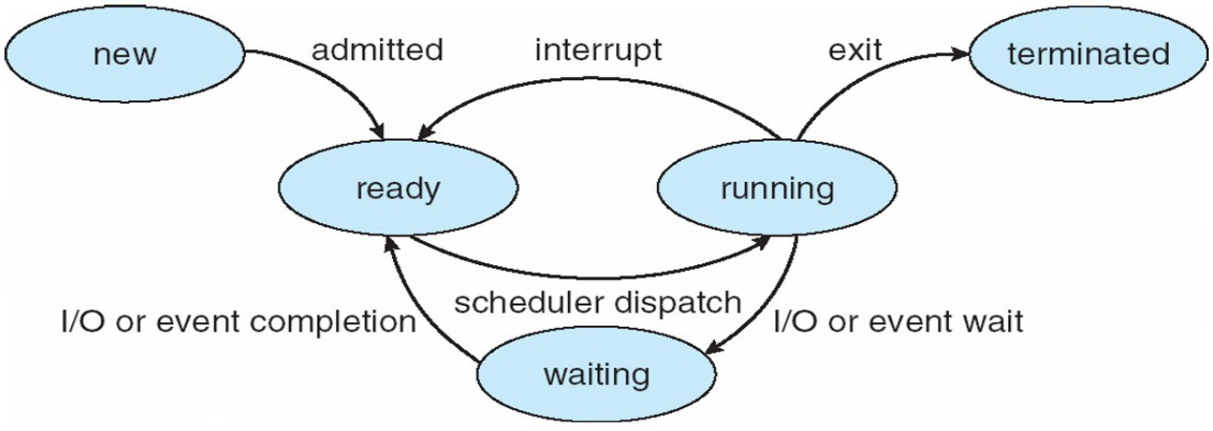
Process in memory

2. Process State

As a process executes, it changes state. The state of a process is defined as the current activity of the process. Process can have one of the following five states at a time.

- **New**: The process is being created.
- **Ready**: The process is waiting to be assigned to a processor. Ready processes are waiting to have the processor allocated to them by the operating system so that they can run.
- **Running**: Process instructions are being executed (i.e. The process that is currently being executed).
- **Waiting**: The process is waiting for some event to occur (such as the completion of an I/O operation).
- **Terminated**: The process has finished execution.

It is important to realize that only one process can be *running* on any processor at any instant. Many processes may be *ready* and *waiting*



3. Process Control Block, PCB

Each process is represented in the operating system by a process control block (PCB) also called a task control block. PCB is the data structure used by the operating system. PCB contains many pieces of information associated with a specific process includes:

ProcessState
Process state may be new, ready, running, waiting and so on.

ProgramCounter
Program Counter indicates the address of the next instruction to be executed for this process.

CPUregisters
CPU registers include general purpose register, stack pointers, index registers and accumulators etc. number of register and type of register totally depends upon the computer architecture.

CPU Scheduling information:

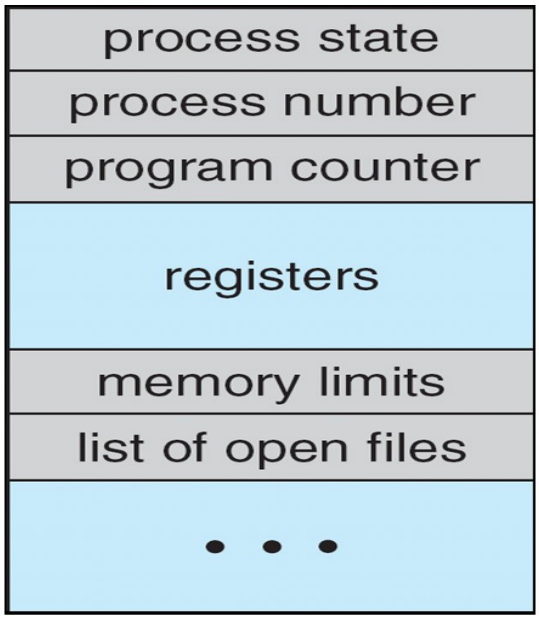
This information includes a process priority,pointers to scheduling queues and any other scheduling information

Memory Management Information
This information may include the value of base and limit registers, the page tables, or the segment tables depending on the memory system used by the operating system.This information is useful for deallocating the memory when the process terminates.

AccountingInformation
This information includes the amount of CPU and real time used, time limits, job or process numbers, account numbers etc.

i/o status information

This information includes the list of i/o devices allocated to the process, a list of open files and so on



Threads

a process is a program that performs a single **thread** of execution. For example, when a process is running a word-processor program, a single thread of instructions is being executed. This single thread of control allows the process to perform only one task at a time. The user cannot simultaneously type in characters and run the spell checker within the same process. On a system that supports threads, the PCB is expanded to include information for each thread. Other changes throughout the system are also needed to support threads.

A thread is a basic unit of CPU utilization; it comprises a thread ID, a program counter, a register set, and a stack. It shares with other threads belonging to the same process its code section, data section, and other operating-system resources, such as open files and signals. A traditional (or ***heavyweight***) process has a single thread of control. If a process has multiple threads of control, it can perform more than one task at a time.

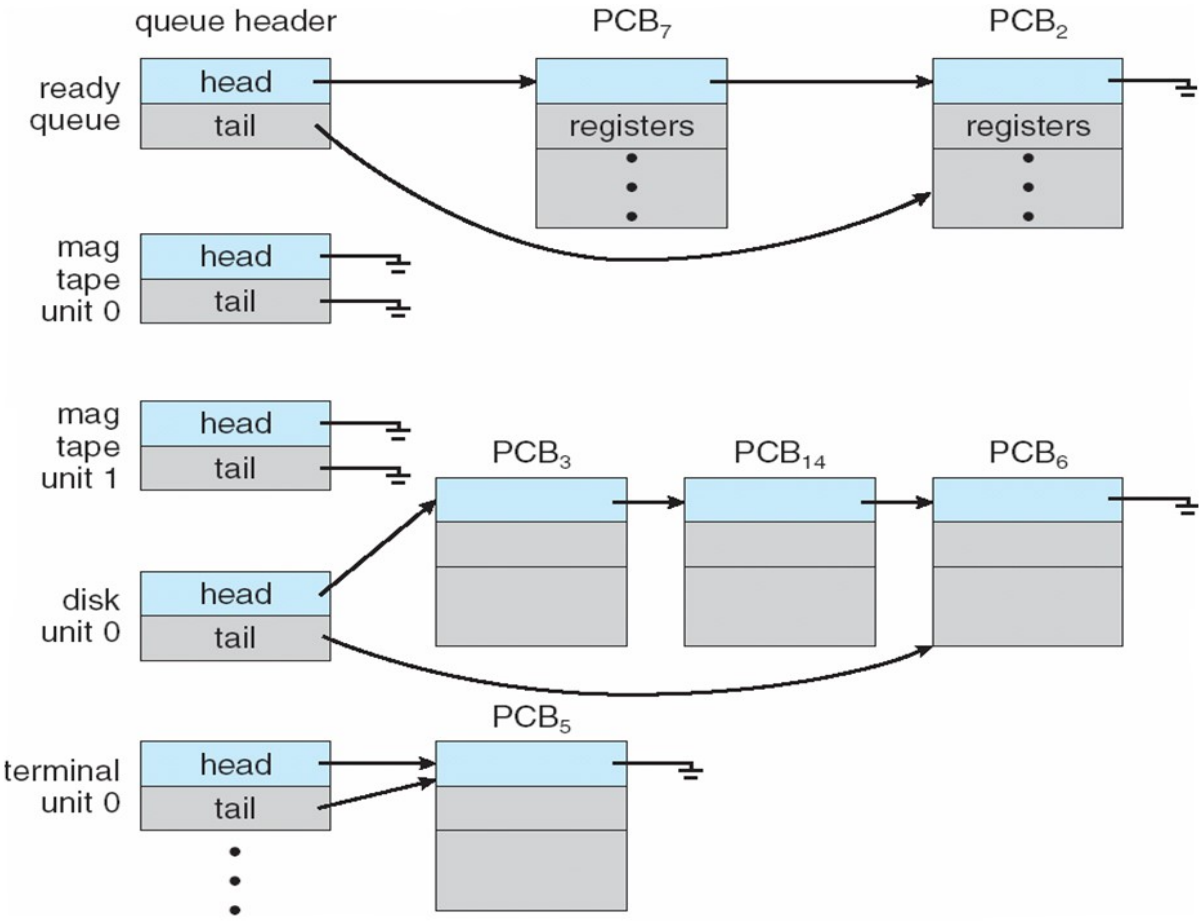
4. Process Scheduling

The objective of multiprogramming is to have some process running at all times, to maximize CPU utilization. When there is more than one process ready to execute with the processor, a selection decision needs to be made to pick a process for execution from the queue of ready processes. This activity is called ***Process Scheduling***.

4.1 Scheduling queues

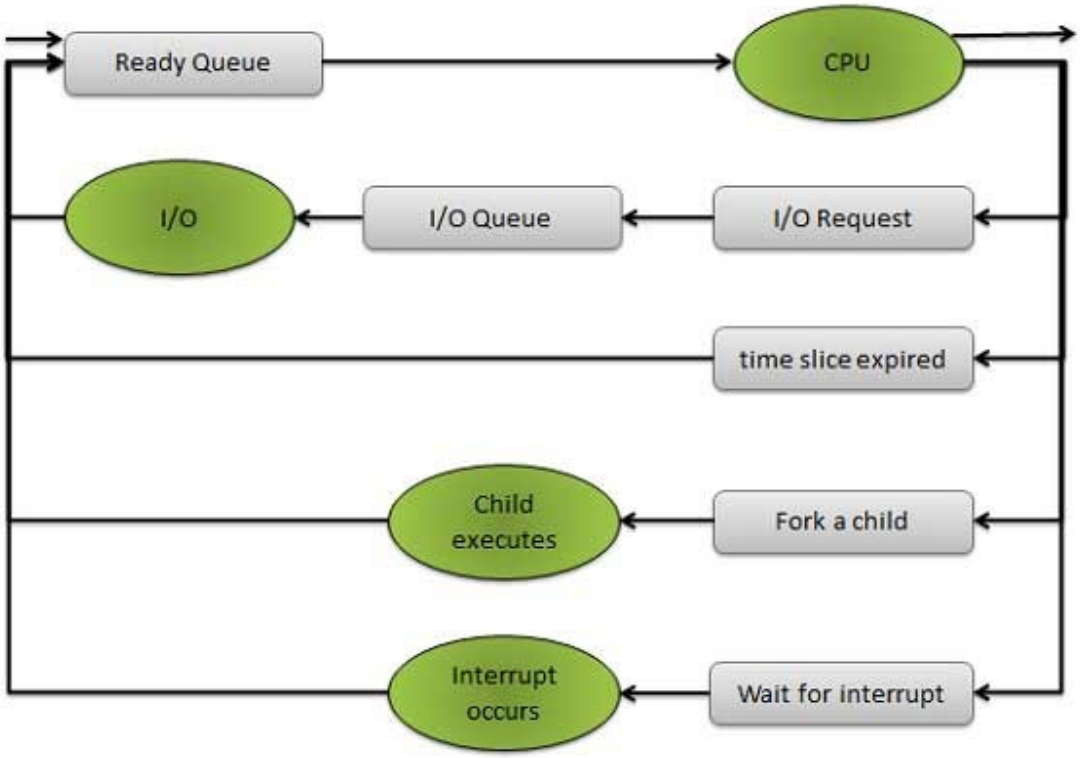
There are three types of scheduling queues:

- **Job queue**: When the process enters into the system, they are put into a job queue, which consist of all processes in the system.
- **Ready queue** : The processes that are residing in main memory and are ready and waiting to execute are kept on a list called the ready queue**.T**his queue is generally stored as a linked list.
- **Device queue:** set of processes waiting for an I/O device



These queuses are generally stored as a linked list.A ready queue header contains pointers to the first and final PCBs in the list. Each PCB includes a pointer field that points to the next PCB in the ready queue.

A common representation of process scheduling is a **queueing diagram**



- ✓ Each rectangular box represents a queue. Queues are of two types
    - Ready queue
    - set of Device queues
- ✓ The circles represent the resources that serve the queues.
- ✓ The arrows indicate the process flow in the system.

A newly arrived process is put in the ready queue. Processes waits in ready queue for allocating the CPU. Once the CPU is assigned to a process, then that process will execute. While executing the process, any one of the following events can occur.

- The process could issue an I/O request and then it would be placed in an I/O queue.
- The process could create new sub process and will wait for its termination.
- The process could be removed forcibly from the CPU, as a result of interrupt and put back in the ready queue.

A process continues this cycle until it terminates at which time it is removed from all queues and has its PCB and resources be allocated

4.2 Schedulers

A process migrates among various scheduling queues throughout its lifetime. The OS select some processes from these queues and this selection process is done by a scheduler.There are two types of schedulers:

- **Long-term scheduler** (or job scheduler) – selects which processes should be brought into the ready queue
- **Short-term scheduler** (or CPU scheduler) – selects which process should be executed next and allocates CPU
- **Medium term scheduler**

Long-term scheduler

It is also referred to as ***Job Scheduler or Admission Scheduler***. It selects processes from the disk and loads them into memory for execution. *Execution frequency* of this scheduler *is less* in comparison to others. It controls the ***degree of multiprogramming***. It is invoked only when a process leaves the system, so as to keep the degree of multiprogramming stable i.e. average rate of process creation = average departure rate of processes. generally the processes can be described in 2 ways:

1. i/o bound

   i/o bound processes spend more time for doing i/o than doing computations

2. CPU bound

   CPU bound processes spend more time for doing computations than doing i/o

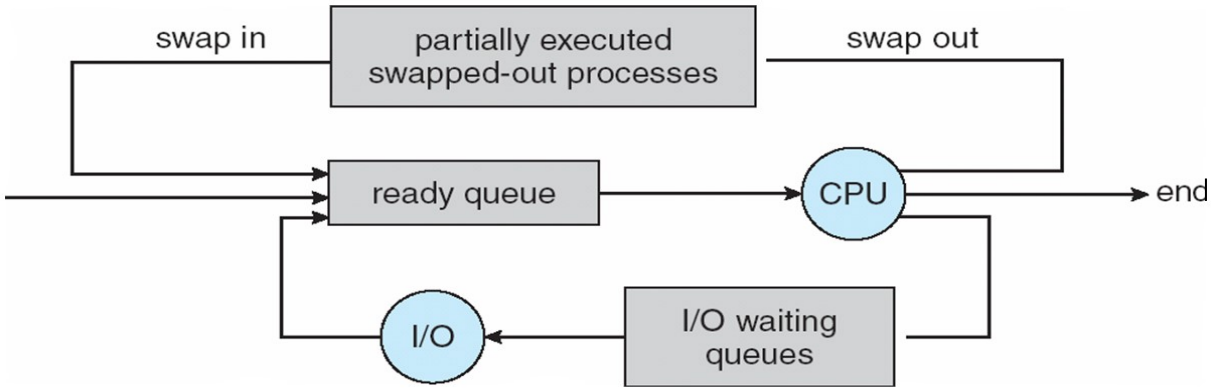
Long term scheduler select a good *mix of CPU-Bound and I/O-Bound processes*

Short-term scheduler

It is also referred to as ***CPU Scheduler***. It selects a process from among the processes in the ready queue for allocation of the CPU. Its frequency if execution is more compared to other schedulers. It is invoked each time the CPU requires a new process for execution. It executes at least once in every 10 milliseconds, provided, it takes 1 millisecond for decision-making. It comprise of two important modules: Dispatcher and Context Switch.

Medium term scheduler

Time sharing system introduce an additional intermediate level of scheduling called medium term scheduler.



It removes the processes from the memory. It reduces the degree of multiprogramming. Later ,the process can be introduced into memory and its execution can be continued where it left off. This scheme is called swapping. The process is swapped out, and is later swapped in, by the medium term schedulerThe medium term scheduler is in-charge of handling the swapped out-processes.

4.3 Context Switch

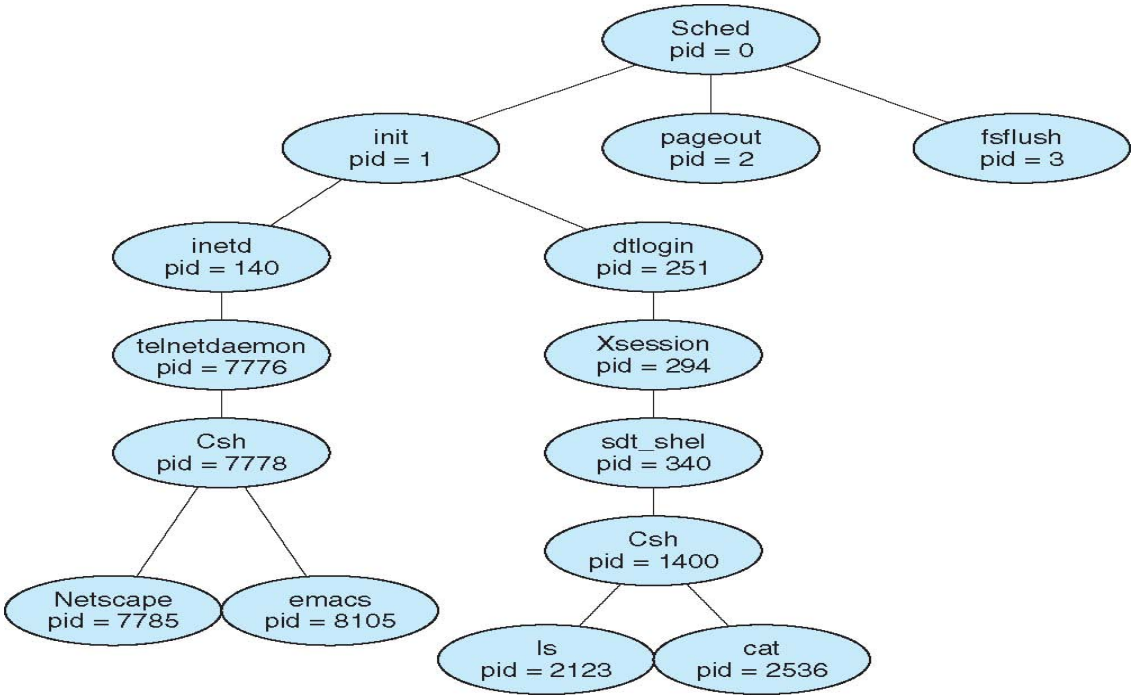
When CPU switches to another process, the system must save the state of the old process and load the saved state for the new process via a context switch. Context of a process represented in the PCB. Actually Context-switch time is overhead, because the system cannot do any useful work while switching. Its speed varies depending on the memory speed,number of registers etc.

4.4.Operations on processes

The processes in most systems can execute concurrently, and they may be created and deleted dynamically. Thus these systems must provide a mechanism for process creation and termination

Process creation

A process may create several new processes, through a create-process system call, during the course of execution. The creating process is called a parent process, and the new processes are called the child processes. **Parent** process create **children** processes, which, in turn create other processes, forming a tree of processes. Generally, most OS identify processes via **a process identifier** (**pid**).Figure below illustrates a typical process tree for the Solaris OS, showing the name of each process and its pid.



In general , a process will need certain resources to accomplish its task

Resources shared in one of the following way

- Parent and children share all resources
- Children share subset of parent's resources
- Parent and child share no resources

When a process creates a new process, two possibilities exist in terms of execution

- Parent and children execute concurrently
- Parent waits until children terminate

There are two possibilities in term of the address space of the new process

- Child duplicate of parent
- Child has a program loaded into it

Process termination

A process terminates when it executes last statement and asks the operating system to delete it by using exit() system call. At that point , the process may

- ✓ Output data from child to parent (via wait)
- ✓ Process' resources are deallocated by operating system

Parent may terminate execution of children processes (abort), for a variety of reasons, such as

- ✓ Child has exceeded allocated resources
- ✓ Task assigned to child is no longer required
- ✓ If parent is exiting

Some operating systems do not allow child to continue if its parent terminates.In such systems, if a process terminates, then all its children must also be terminated. This phenomenon is calied **cascading termination**

4.5 Inter process communication(IPC)

Concurrent processes executing in the system may be **independent** or **cooperating.**Independent processes cannot affect other processes executing in the system. ie,these processes doesn't share any data with other processes.Cooperating process can affect or be affected by other processes, including sharing data

Cooperating processes allows:

Information sharing

Computation speedup

Modularity

Convenience

Cooperating processes need **interprocess communication** (**IPC**)

Two models of IPC

- Shared memory
- Message passing