PROCESSES

Prof. Chetan K. Solanki

Process

- A process is a set of sequential steps that are required to do a particular task.
- A process is an instance of a program in execution.
- For e.g.: in Windows, if we edit two text files, simultaneously, in notepad, then it means we are implementing two different instances of the same program.
- For an operating system, these two instances are separate processes of the same application.

Process

- A process needs certain resources such as:
  - CPU Time
  - Memory Files
  - I/O Devices

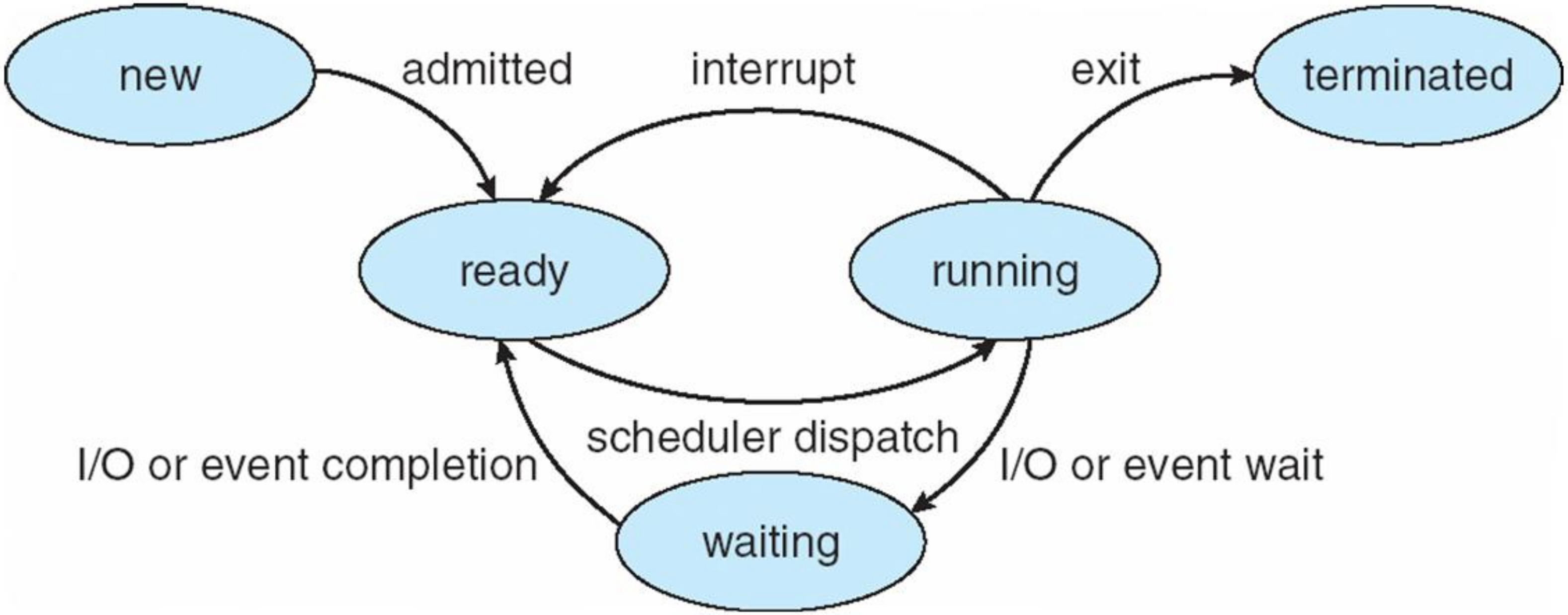
  to accomplish its task.
- These resources are allocated to the process either when it is created or while it is executing.

Process States

- A process goes through a series of process states for performing its task.
- As a process executes, it changes state.
- Various events can cause a process to change state.

Process States

- The various states of a process are:



Process States

- **New:**
  - A process that has just been created.
- **Ready:**
  - The process is ready to be executed.
- **Running:**
  - The process whose instructions are being executed is called running process.

Process States

- **Waiting:**
  - The process is waiting for some event to occur such as completion of I/O operation.
- **Terminated:**
  - The process has finished its execution.
- **Note:** Only one process can be *running* on any processor at any instant. However, there can be many processes in *ready* and *waiting* states.

Process Control Block (PCB)

- Process Control Block (PCB) is a data structure used by operating system to store all the information about a process.

- It is also known as Process Descriptor.

- When a process is created, the operating system creates a corresponding PCB.

Process Control Block (PCB)

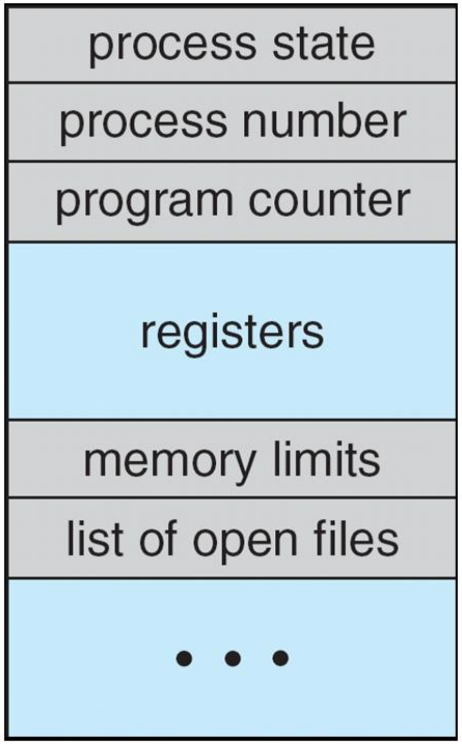
- Information in a PCB is updated during the transition of process states.

- When a process terminates, its PCB is released.

- Each process has a single PCB.

Process Control Block (PCB)

- The PCB of a process contains the following information:



process state

process number

program counter

registers

memory limits

list of open files

• • •

Process Control Block (PCB)

- **Process Number:** Each process is allocated a unique number for the purpose of identification.
- **Process State:** It specifies the current state of a process.
- **Program Counter:** It indicates the address of next instruction to be executed.

Process Control Block (PCB)

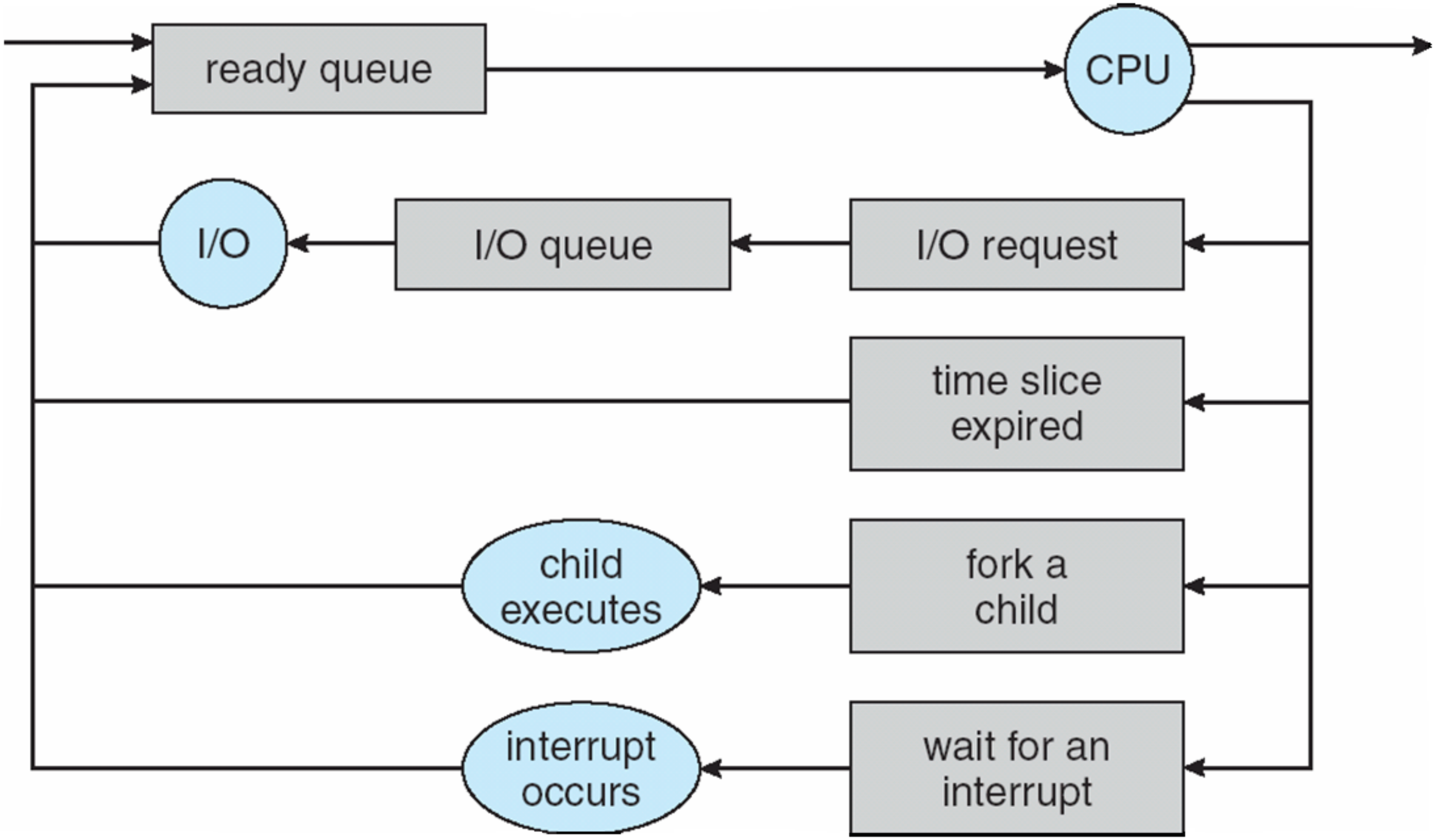
- **Registers:** These hold the data or result of calculations. The content of these registers is saved so that a process can be resumed correctly later on.
- **Memory Limits:** It stores the amount of memory units allocated to a process.
- **List of Open Files:** It stores the list of open files and there access rights.

Process Scheduling

- In multiprogramming, several processes are kept in main memory so that when one process is busy in I/O operation, other processes are available to CPU.
- In this way, CPU is busy in executing processes at all times.
- This method of selecting a process to be allocated to CPU is called Process Scheduling.

Process Scheduling

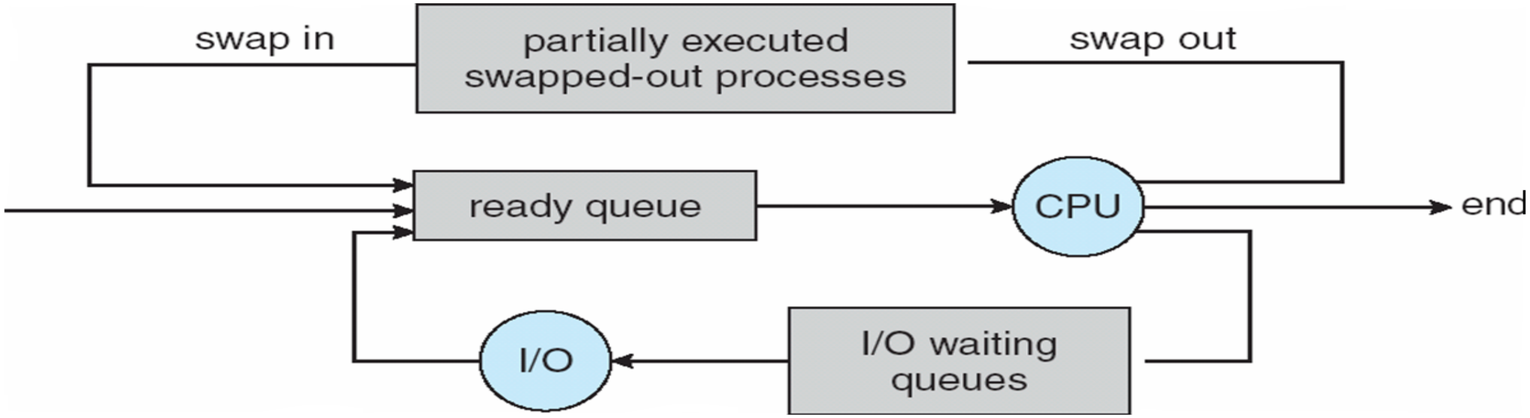
- Process scheduling consists of the following sub-functions:
  - **Scheduling:** Selecting the process to be executed next on CPU is called scheduling.
    - In this function a process is taken out from a pool of ready processes and is assigned to CPU.
    - This task is done by a component of operating system called **Scheduler.**


ready queue
CPU
I/O
I/O queue
I/O request
time slice expired
child executes
fork a child
interrupt occurs
wait for an interrupt

Process Scheduling

- **Dispatching:** Setting up the execution of the selected process on the CPU is called dispatching.
  - It is done by a component of operating system called **Dispatcher**.
  - Thus, a dispatcher is a program responsible for assigning the CPU to the process, that has been selected by the Scheduler.
- **Context Save:** Saving the status of a running process when its execution is to be suspended is known as context save.

- **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



Scheduling Queues

- In multiprogramming, several processes are there in ready or waiting state.
- These processes form a queue.
- The various queues maintained by operating system are:
  - Job Queue
  - Ready Queue
  - Device Queue

Scheduling Queues

- **Job Queue:**
  - As the process enter the system, it is put into a job queue. This queue consists of all processes in the system.

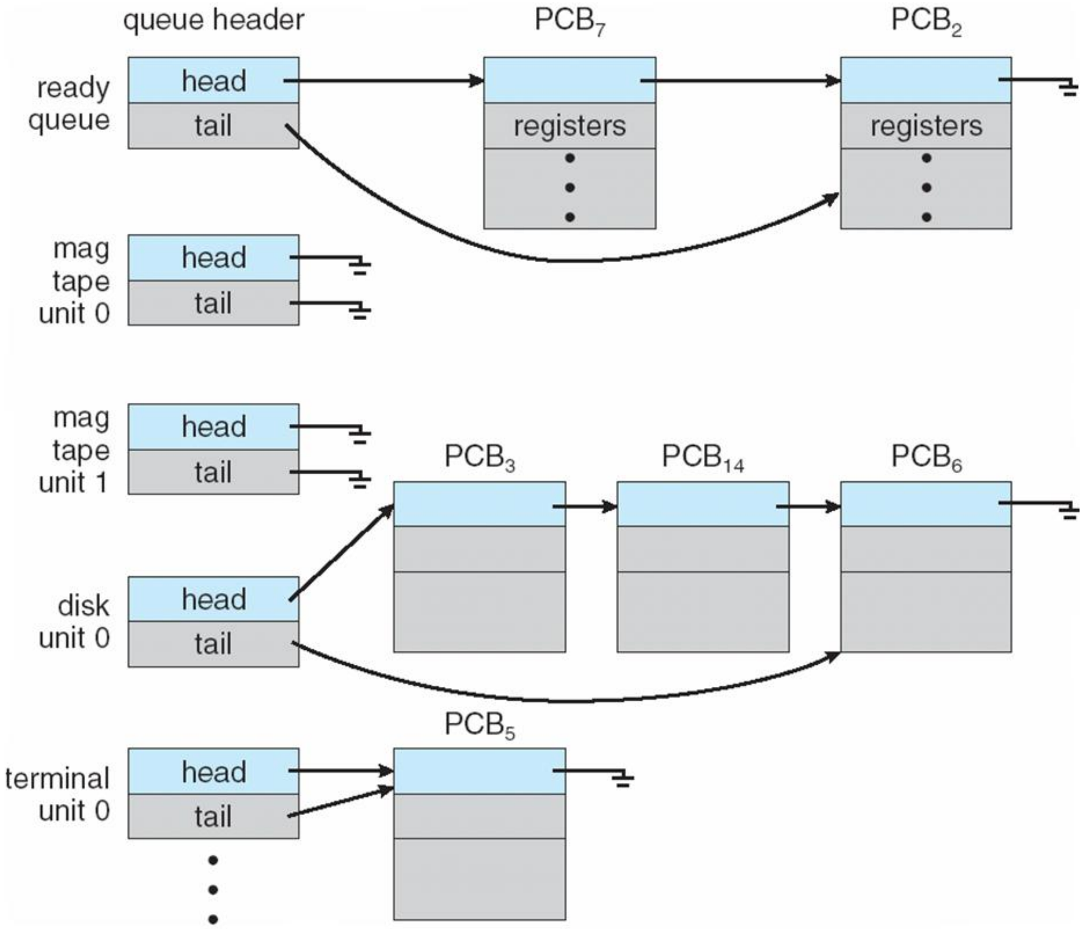
- **Ready Queue:**
  - It is a doubly linked list of processes that are residing in the main memory and are ready to run.

Scheduling Queues

- **Device Queue:**
  - It contains all those processes that are waiting for a particular I/O device.
  - Each device has its own device queue.

- Diagram on the next slide shows the queues.

Scheduling Queues



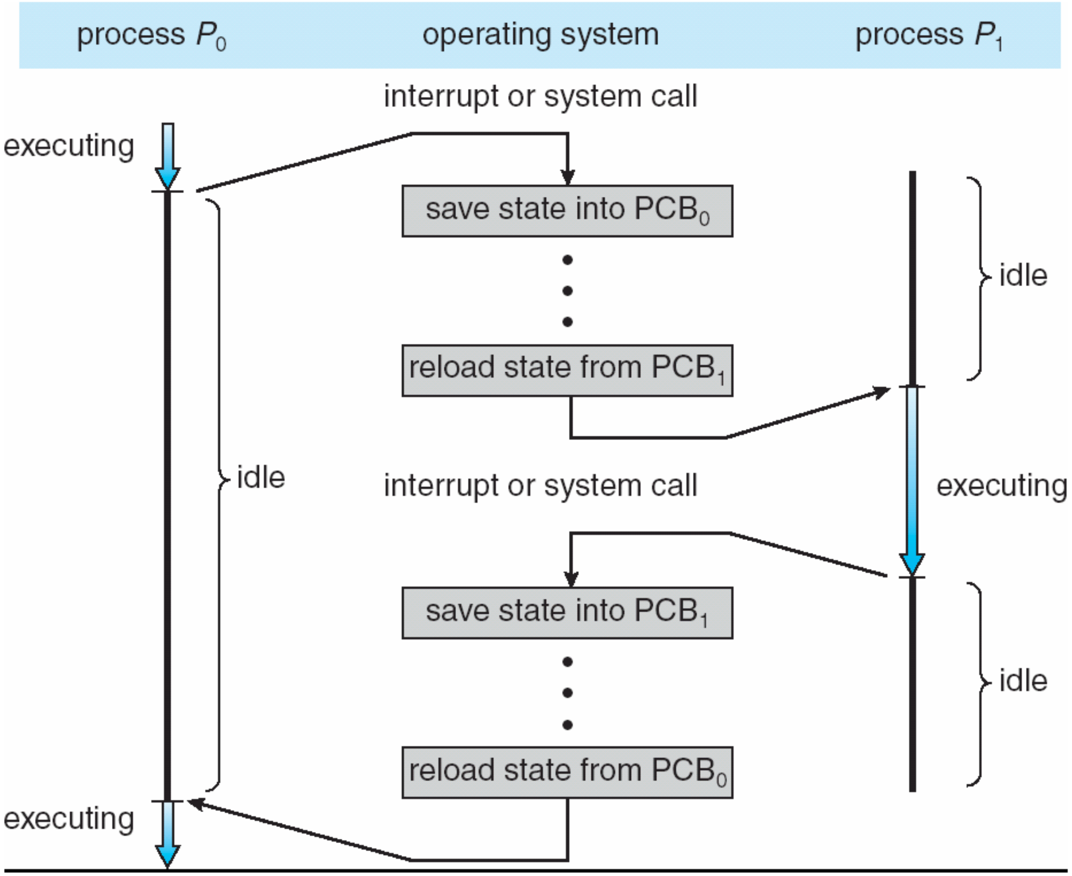
Schedulers (Cont)

- Short-term scheduler is invoked very frequently (milliseconds) ⇒ (must be fast)
- Long-term scheduler is invoked very infrequently (seconds, minutes) ⇒ (may be slow)
- The long-term scheduler controls the *degree of multiprogramming*
- Processes can be described as either:
  - **I/O-bound process** – spends more time doing I/O than computations, many short CPU bursts
  - **CPU-bound process** – spends more time doing computations; few very long CPU bursts

Context Switch

- Switching the CPU from one process to another process requires saving the state of old process and loading the saved state of new process.
- This task is known as **Context Switch.**
- When context switch occurs, operating system saves the context of old process in its PCB and loads the saved context of the new process.

Context Switch



process P_0

operating system

process P_1

interrupt or system call

executing

save state into PCB_0

idle

idle

reload state from PCB_1

interrupt or system call

executing

save state into PCB_1

idle

reload state from PCB_0

executing

Process Scheduling