0301304 FUNDAMENTAL OF OPERATIONG SYSTEM

UNIT	MODULES	WEIGHTAGE
1	INTRODUCATION TO OPERATING SYSTEM	20 %
2	PROCESS MANAGEMENT	20 %
3	PROCESS COMMUNICATION AND SYNCHRONIZATION	20 %
4	MEMORY MANAGEMENT	20 %
5	FILE MANAGEMENT , DISK MANAGEMENT , SECURITY AND PROTECTION	20 %

UNIT -2 Process Management

- Fundamentals of Process Management
- Relationship between Processes
- Life Cycle of Process
- Process Control Block
- Schedulers
  - Long Term Scheduler
  - Short Term Scheduler
  - Medium Term Scheduler

UNIT -2 Process Management

- Scheduling Algorithms
  - First Come First Serve (FCFS)
  - Priority Scheduling
  - SJN
  - Round Robin
  - SRT

UNIT -2 Fundamental of Process Management

- Program
- Job
- Task
- Process

Program = Job = Task = Process

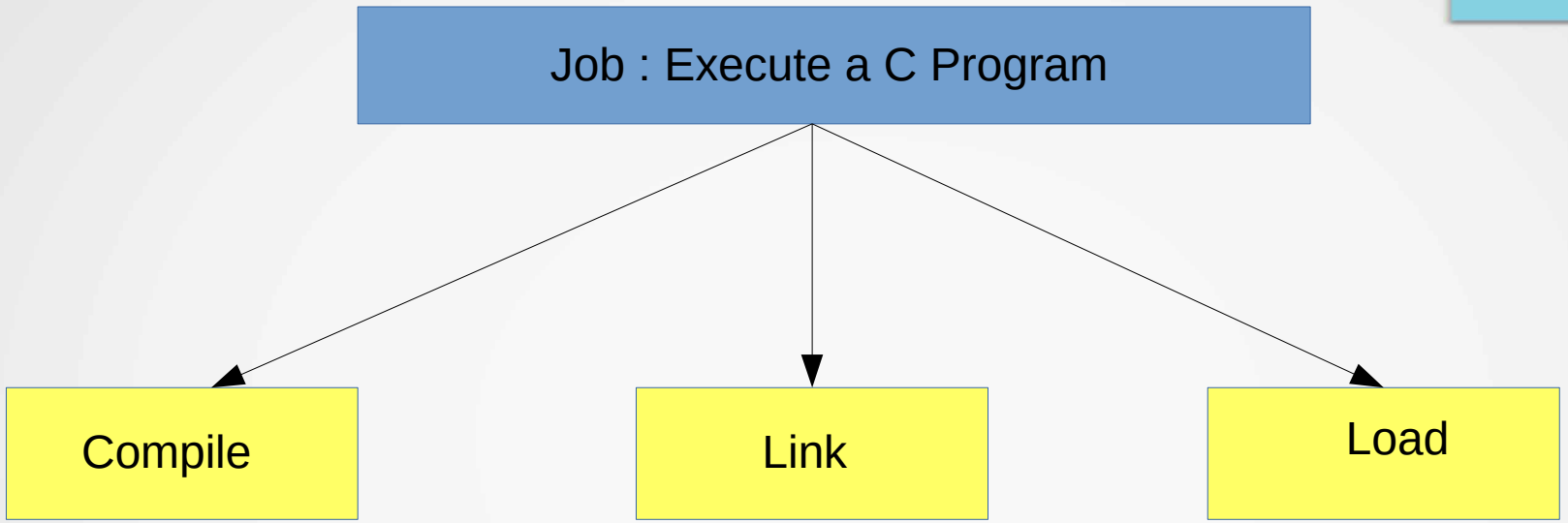
UNIT -2 Program, Job & Task

- To perform a computation on the computer system, we must have a unit of work or execution for the user computation.
- **Program**
  - A program can be **considered a set of instructions** in the form of modules.
  - For examp – C program

UNIT -2 Program, Job & Task

- **Job**
  - In the Batch system, there was a requirement to load and unload the magnetic tapes for various activities such as
    - **Compiling**
    - **Linking**
    - **Loading**
  - Job was used for performing the execution of a program through the execution of those activities.
  - **Job is a sequence of single programs**

UNIT -2 Program, Job & Task



Job as a sequence of programs

UNIT -2 Program, Job & Task

- **Task**
  - The term task was used **when there was a need to have concurrent execution on a single processor.**
  - More than one program of a single user is called Task.
  - In window environment opening multiple window.
  - **Multi Programming is also called Multi Tasking**

UNIT -2 Program, Job & Task

- **Process**
  - The term **'Process' is different from the term 'Job' or 'Program'.**
  - A **Program is a set of instructions** the user has written and stored somewhere.
  - It means a program **is passive entity** and continues to exist at a place.
  - **When a program is ready for the execution, it becomes active and is known as a process.**

UNIT -2 Program, Job & Task

- **Process**
  - When a program is ready for execution or in other words, when it becomes a process, **it means that now it can compete for resources.**
  - The resources are :
    - CPU time
    - Memory
    - I/O devices, etc.

UNIT -2 Fundamental of Process Management

- Process **is a basic term to** understand the operating sytem.
- There are **number of user and system processes,** so their is need to manage them.
- Any given time, **any running process may be interrupted.**
- Due to **this interruption, the processes are not in the same state forever.** They change state according to an event in the system.

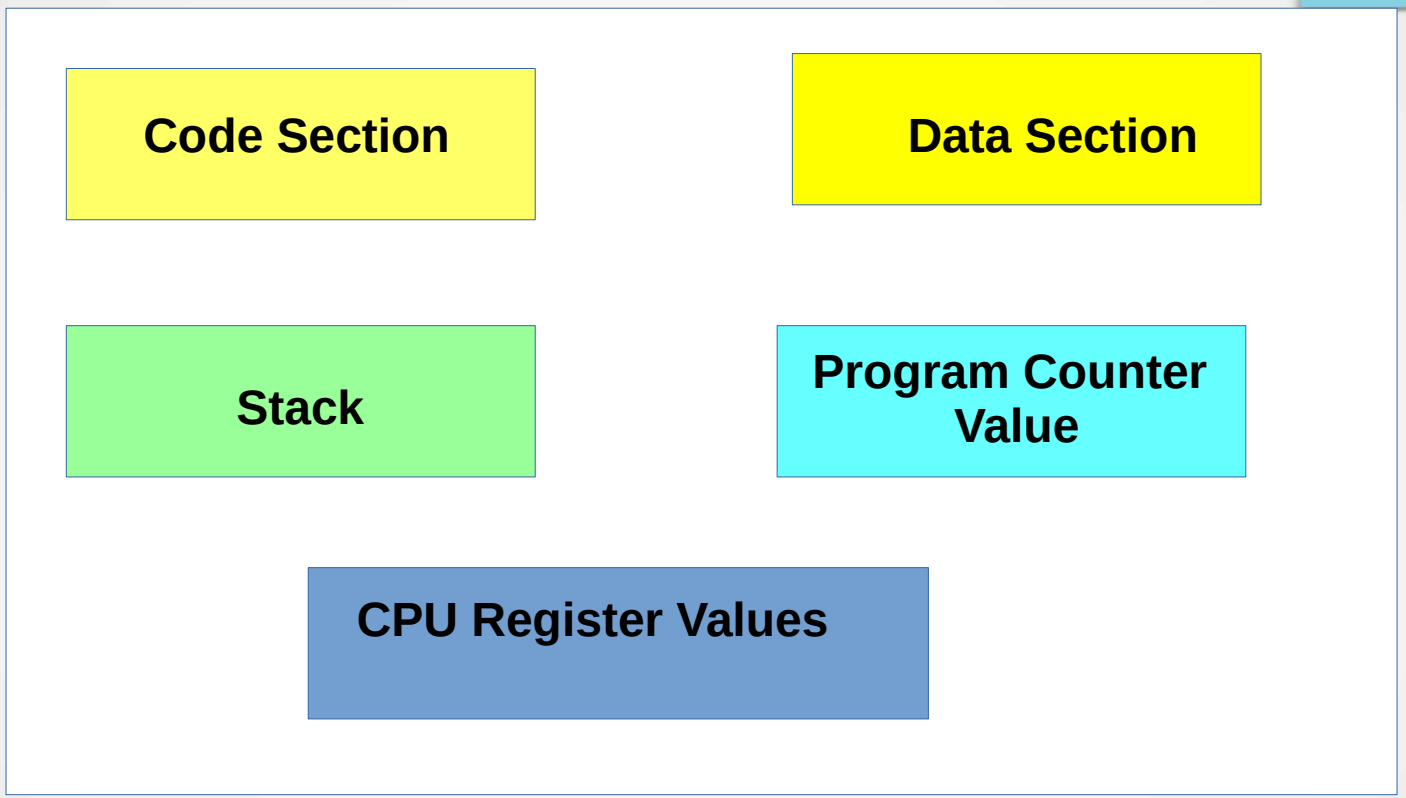
UNIT -2 Fundamental of Process Management

- If a process is interrrupted, another process is scheduled to be dispatched to processor for execution.
- Addition to this, processes also need to communicate and synchronize with each other.
- It is critical to manage the processes in the system from the view point of their **state change, scheduling, dispatching, process switching, communication, synchronization and so on.**

UNIT -2 Program, Job & Task

- **Process**
  - When a program need to execute and when it get the CPU time, it has :
    - ***Program Counter (PC)*** value – used for moving to the next instruction while executing along with
    - ***Program Code / Code Section***
    - ***Data Section and stack*** also allocated to process along with other resources.
    - At the time of exection process may store data at **CPU Registers.**

UNIT -2 Process Environment



Process Environment

UNIT -2 Program V/s Process

Program	Process
Passive / Static	Active / Dynamic
Can’t compete for resources	Competes for resources
Has a code section	Has a code section, data section, stack, and program counter

UNIT -2 Relationship between Processes

- The program may consist of set of procedures or functions, but each of them executes in a sequence.
- This progam when executed becomes a single process consisting of all procedures inside the program.
- In some case, some procedures or functions in a program can be executed simultaneously.
- That is, there is no sequence of order of execution between them.
- **In this case, the program when executed consists of many processes, this type of program is known as concurrent program.**

UNIT -2 Relationship Between Processes

- In general, there are some processes (created out of a program or concurrent program) whose execution overlap in time. **This processes are known as concurrent processes.**
- Concurrent processes may be indepedent of each other and are known as **independent processes.**
- It may happen that, they interact with each other, share data, send a message to each other or send signals to coordinate their activites. **These are known as interacting or cooperating processes.**

UNIT -2 Life Cycle of a Process

- A Process is an active entity and change its state with time.
- A Process from its creation to termination passes through various states.
- A process when created is in a ***new state as a program/job.***
- When new job is entered into the system, it is stored in the ***job pool / queue*** mainted in the hard disk.

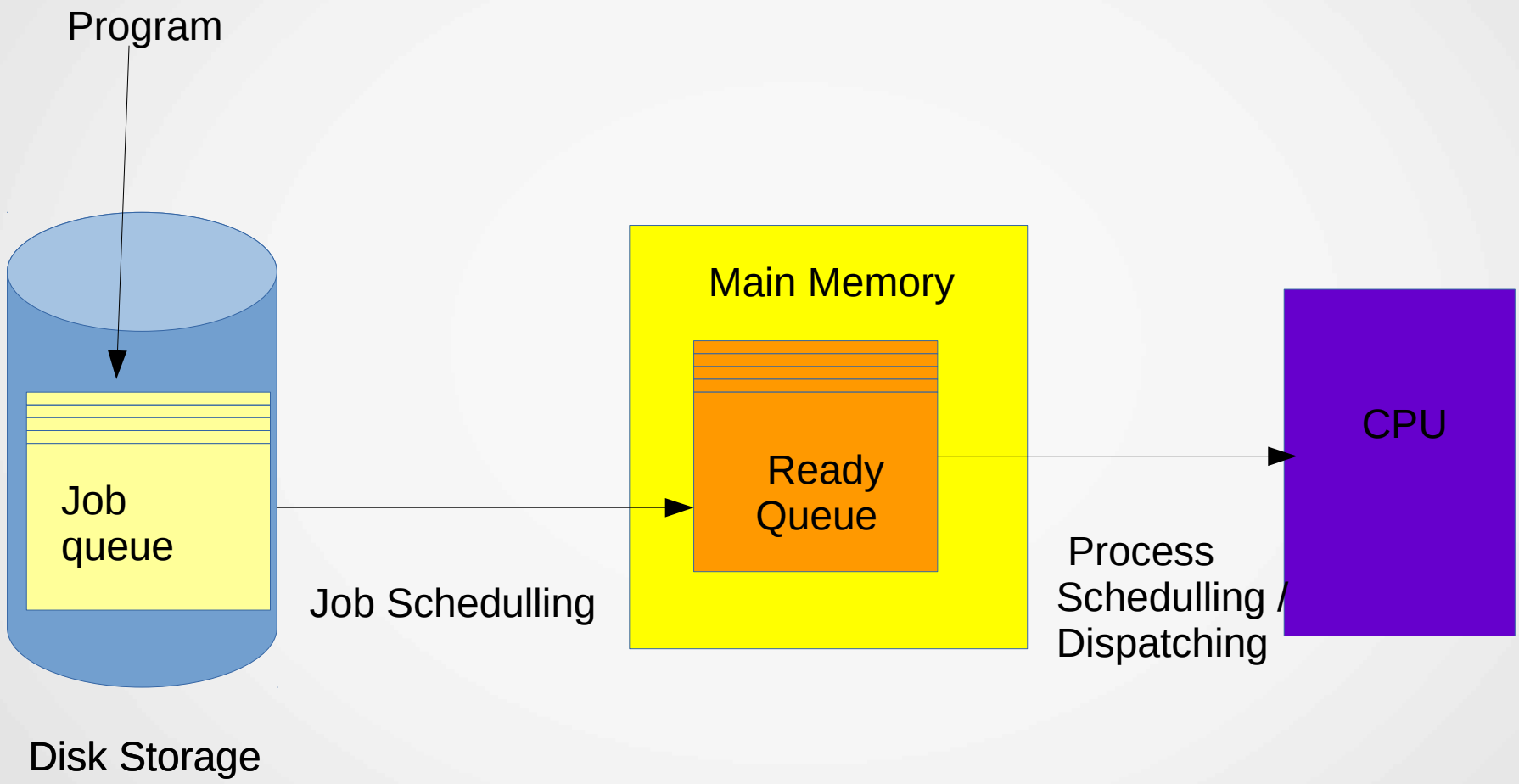
UNIT -2 Life Cycle of a Process

- In the job queue, a job waits for its turn to be moved to the main memory.
- When a job is selected to be brought into the main memory, it is ***called job scheduling.***
- When a job is loaded into the memory, it becomes a ***process*** and is stored in a waiting queue where all processes wait for their turn to be sent to the CPU for execution.
- This waiting queue is called ***ready queue.***
- A Process in ready queue becomes ready for exeution as it can compete for the CPU and resources.

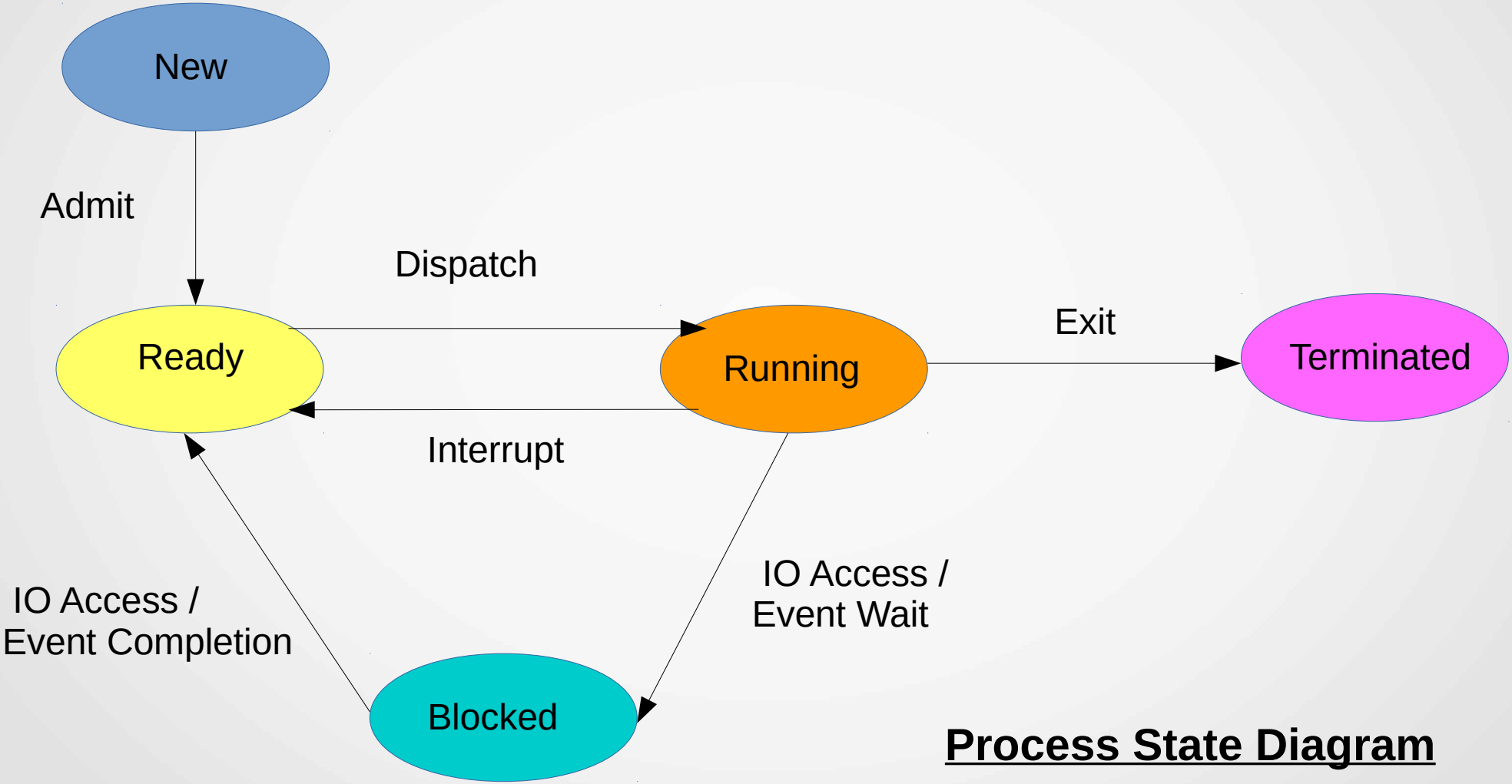
UNIT -2 Life Cycle of a Process

- The processes in ready queue are then selected for the next execution called ***process scheduling or CPU scheduling.***
- The selected process is sent for execution called ***process dispatching.***
- After getting the CPU time, the running process executes its full code and terminates.
- This scheduling and dispatching functions are performed by the ***scheduler*** and ***dispatcher.***

Job Scheduling & Process Scheduling



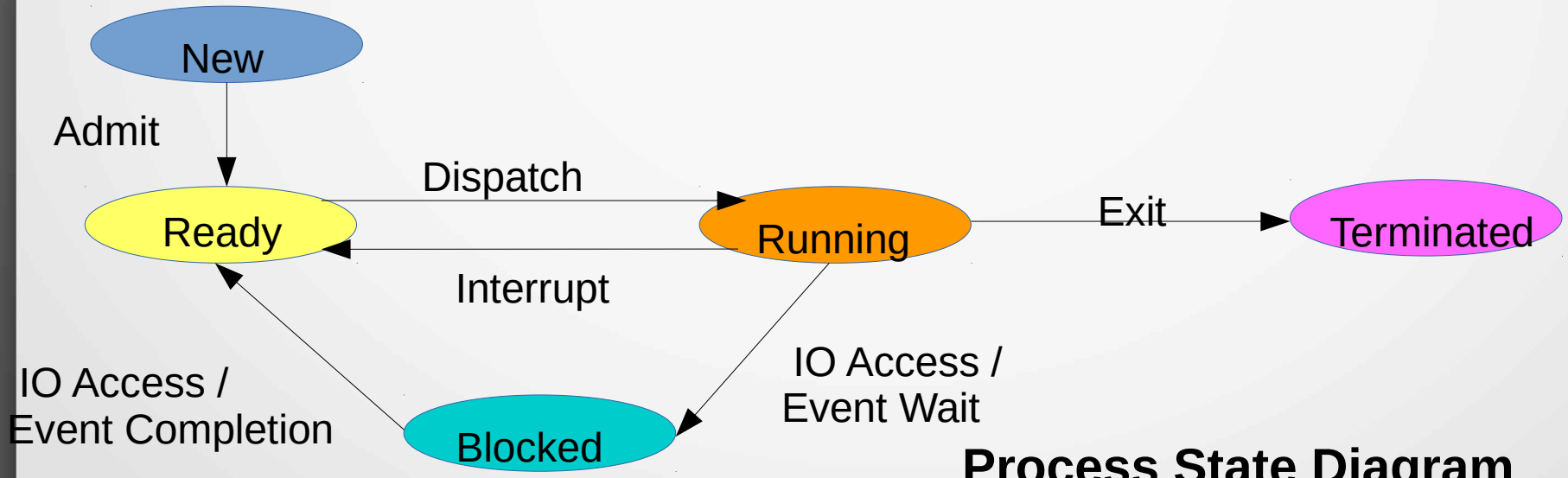
UNIT -2 Process States and State Transitions



Process State Diagram

UNIT -2 Process States and State Transitions

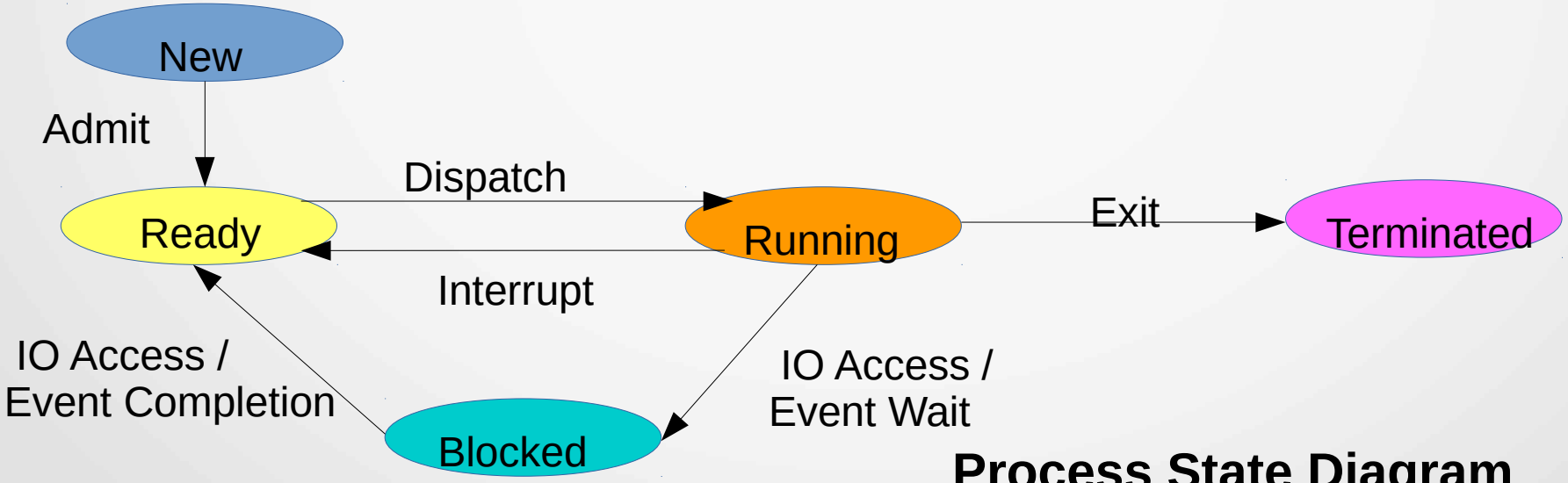
- **New State**
  - Whenever a job/program enters the system, it is put into a **job queue**, the process is in its ***New state.***
  - It means that the process is still **in the secondary storage** as a program.



Process State Diagram

UNIT -2 Process States and State Transitions

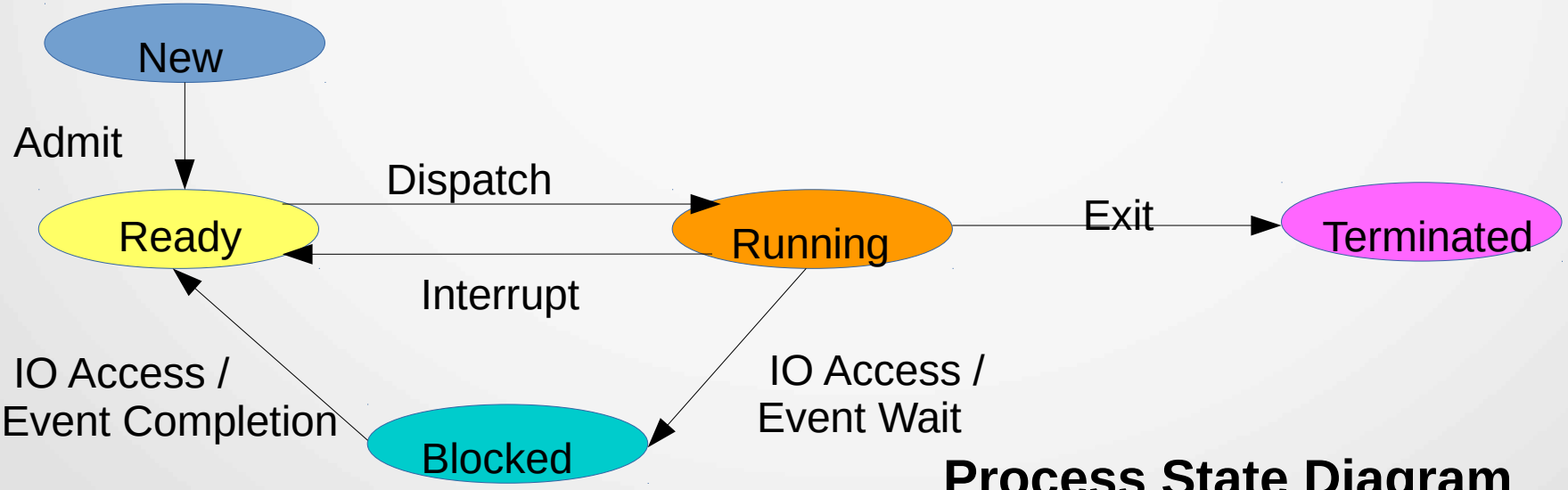
- **Ready State**
  - When program in job queue is scheduled and brought to the main memory in **the ready queue**, the state of the process is ***ready.***
  - The process in ready state is called ready because now it is ready for execution but not executing.



Process State Diagram

UNIT -2 Process States and State Transitions

- **Running State**
  - A process in the ready queue when **selected by the scheduling mechanism for execution and dispatched to the CPU becomes a running process.**
  - The CPU executes the instruction in the code of process.
  - A process while running does not mean that it will hold the CPU until it terminates.



Process State Diagram

UNIT -2 Process States and State Transitions

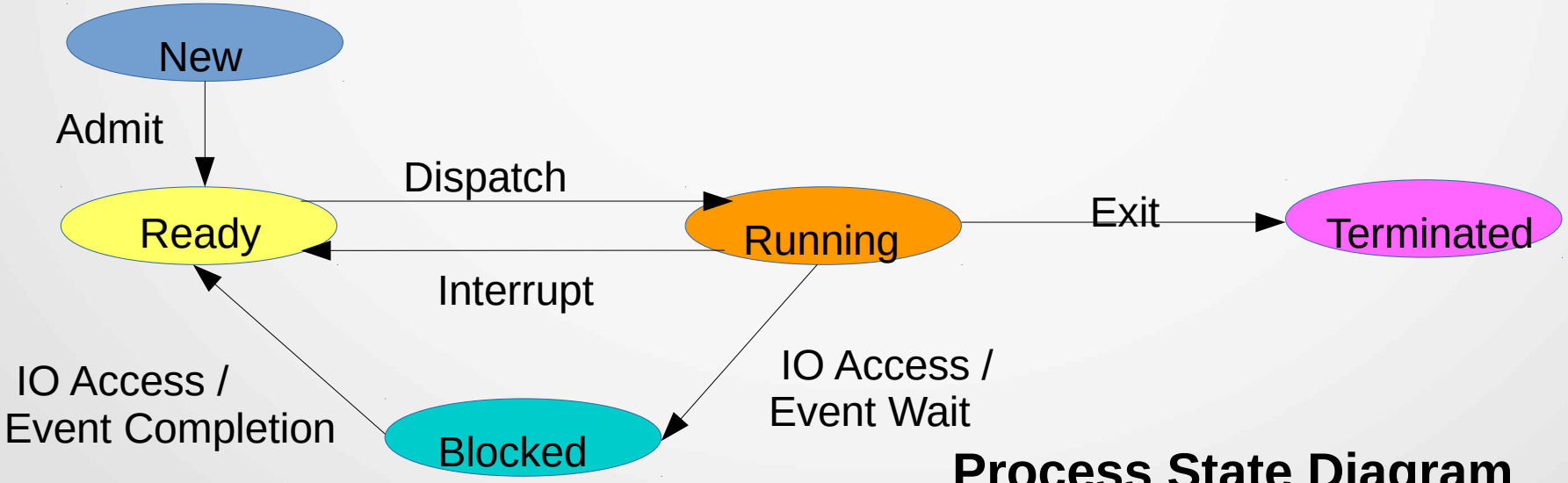
- **Blocked State**
  - A process while executing may reach an instruction where it has to wait for **some I/O devices or some other event.**
  - In this case, the processor will be taken away from the running process and may be given to another ready process.
  - Therefore, **the current running process becomes a blocked process.**
  - The blocked process waits for the event only in the main memory but in a separate queue know as **blocked queue.**

UNIT -2 Process States and State Transitions

- **Blocked State**
  - When the **I/O access or the other event is over, the process is now again ready to execute further.**
  - However now it cannot be given the processor because some other process may be executing at that time.
  - Therefore, **the blocked process after its wait will move again to the ready queue for its turn to execute.**

UNIT -2 Process States and State Transitions

- **Terminated State**
  - A process executed completely till its end and terminates and becomes a ***terminated process.***



Process State Diagram

UNIT -2 Process States and State Transitions

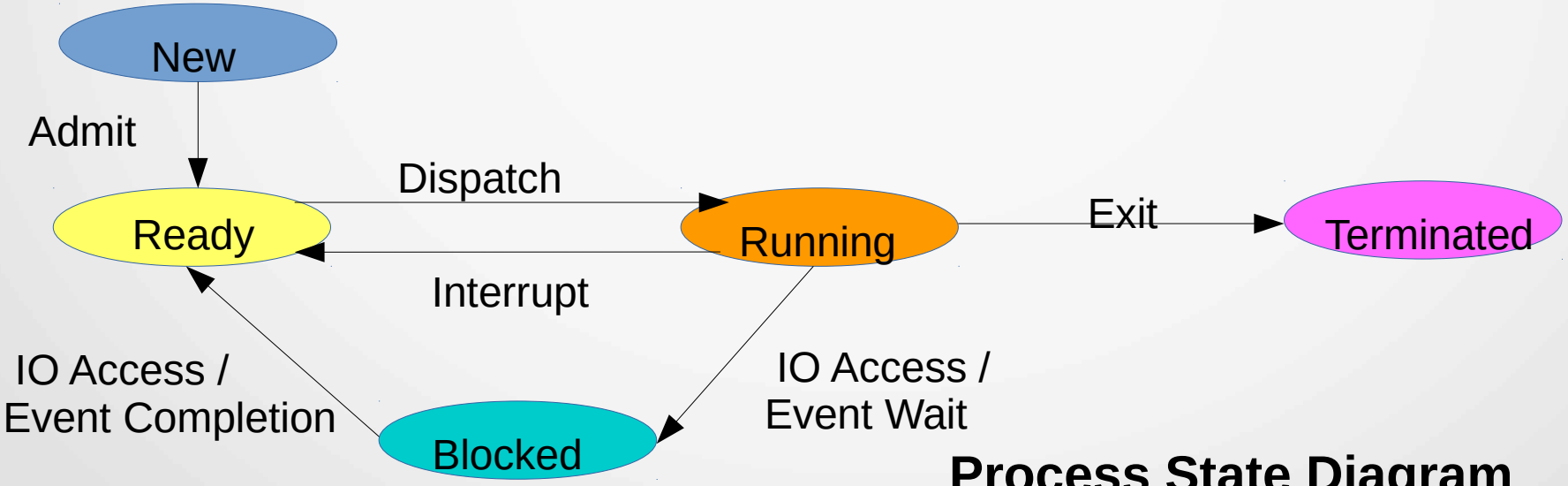
- The **process changes its state when there is an event causing a state transition.** The events can be of the following types:
  - A new process is created.
  - The process makes a resource request.
  - Resource is released.
  - The process requests an I/O device.
  - An I/O device is released after access
  - The alocated time slice for a process is over. In this case, system timer sends a timer interrupt.

UNIT -2 Process States and State Transitions

- The **process changes its state when there is an event cusing a state transition.** The events can be of the follwing types:
  - A higher – priority process appears in the ready queue.
  - The process reaches its end of execution or is aborted.
  - Any hardware interrupt generated.
  - An error or exception condition is generated in the current running process.

UNIT -2 Process States and State Transitions

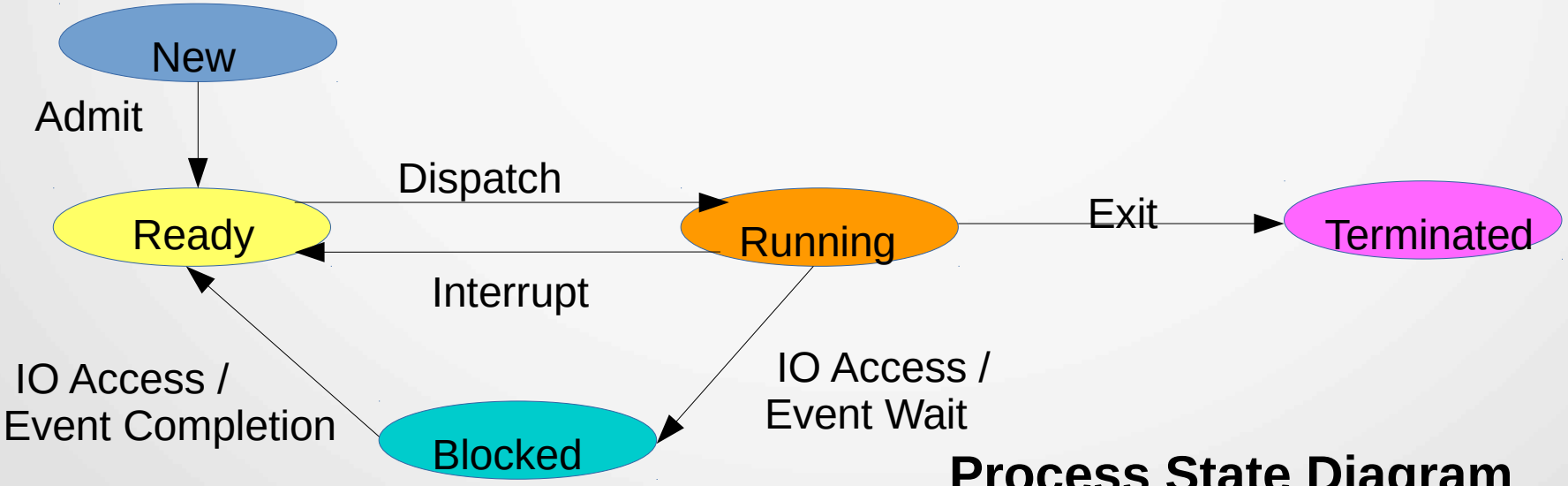
- **Event become the source of state transitions** of the processes and help in managing the processes.
  - **Admit** (New -> Ready)
  - **Dispatch** (Ready -> Running)
  - **Exit** (Running -> terminated)



Process State Diagram

UNIT -2 Process States and State Transitions

- **Event become the source of state transitions** of the processes and help in managing the processes.
  - **Interruput** (Running - > Ready)
  - **I/O or Event Wait** (Running -> Blocked)
  - **I/O or Event Wait Completion** (Blocked -> Ready)



Process State Diagram

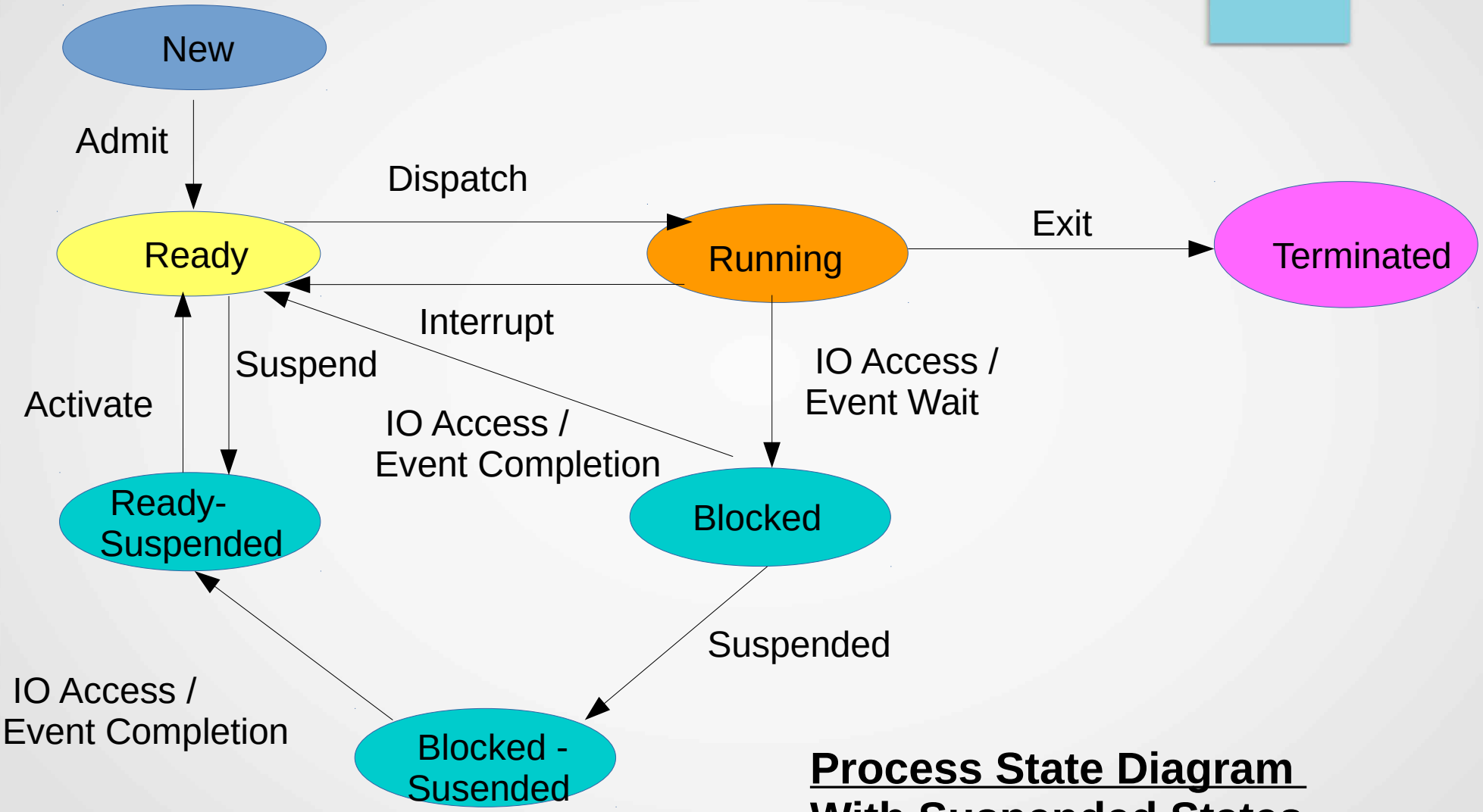
UNIT -2 Suspended Processes & Their State Transitions

- Any process **change their states when there is an event.**
- When a process waits for an I/O device, that is, when the process is blocked, another process from the ready queue is scheduled.
- However, there may be the case that **all the processes need I/O devices, that is all the processes at a particular instant of time are blocked** and are waiting for some event to happen and no process is under execution.
- In this case, **no useful work is being done by the processor.**

UNIT -2 Suspended Processes & Their State Transitions

- Therefore, it is n**ecessary to bring in some process that is ready for execution.**
- However, **there may be a situation that there is no space so that a new process may be swapped in.**
- Therefor, we need to create the memory space for this purpose.
- Since blocked processes cannot be executed unless their I/O devices are released, **some blocked process may be swapped out.**
- The swapped out process is known as **suspended process** and the **queue where it waits is called suspended queue** in the secondary storage such **as hard disk.**

UNIT -2 Process States and State Transitions



Process State Diagram With Suspended States

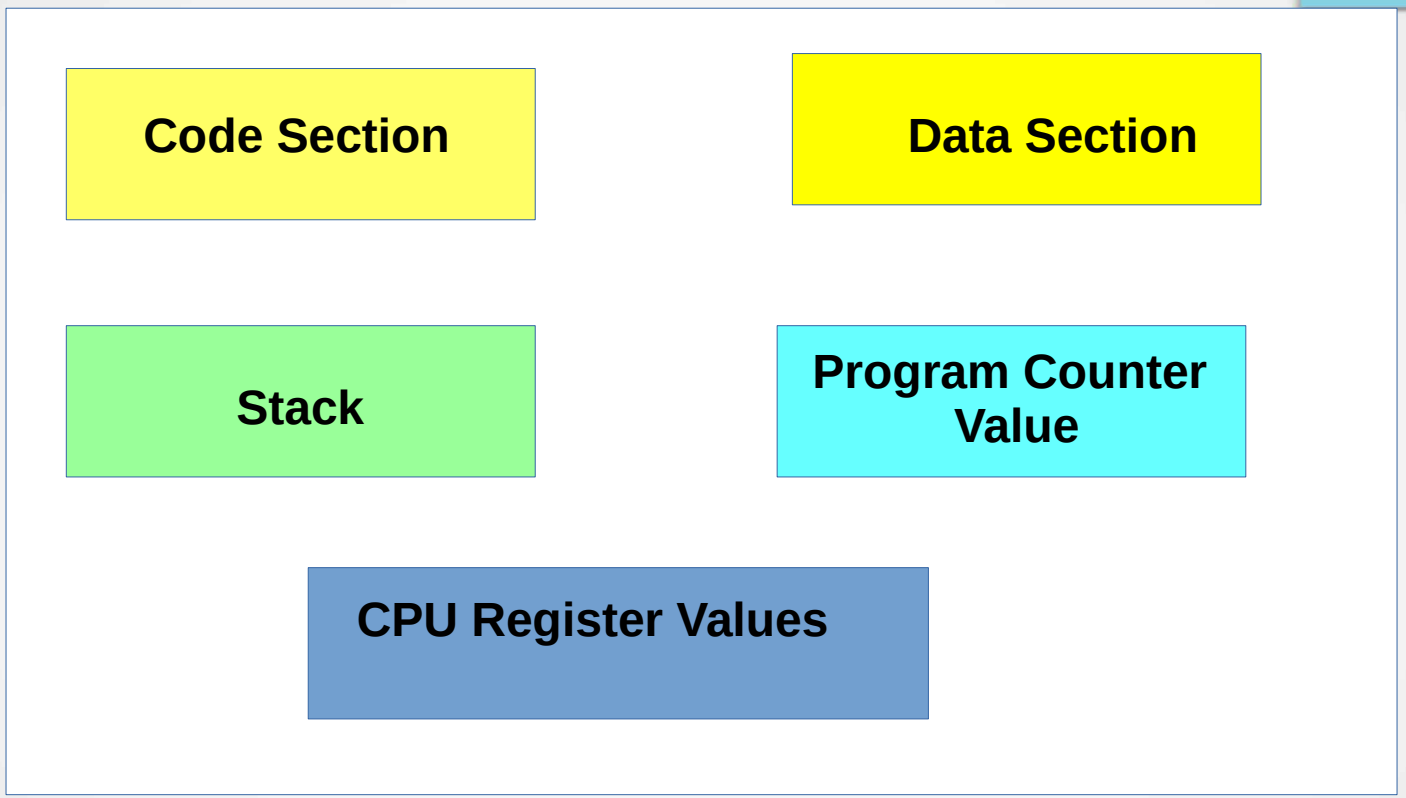
UNIT -2 Suspended Processes & Their State Transitions

- **Blocked Suspended State**
  - The b**locked process waiting in blocked queue in the memory is suspended and moved to suspended queue in disk.**
  - The state of the process is called **blocked suspended.**
- **Ready Suspended State**
  - When the event for which the blocked s**uspended process was waiting has occurred, its state changes.**
  - The state is r**eady because now it is ready to be executed.**
  - However yet it cannot be executed as it is still in the disk.
  - Therefore, its state is **called ready Suspended.**

UNIT -2 Suspended Processes & Their State Transitions

- **Suspend** (Blocked -> Blocked Suspend)
- **I/O or Event Wait Completion** (Blocked Suspend -> Ready suspend)
- **Activate** (Ready Suspend -> Ready)
- **Suspended** (Ready -> Ready Suspend)

UNIT -2 Process Environment



Process Environment

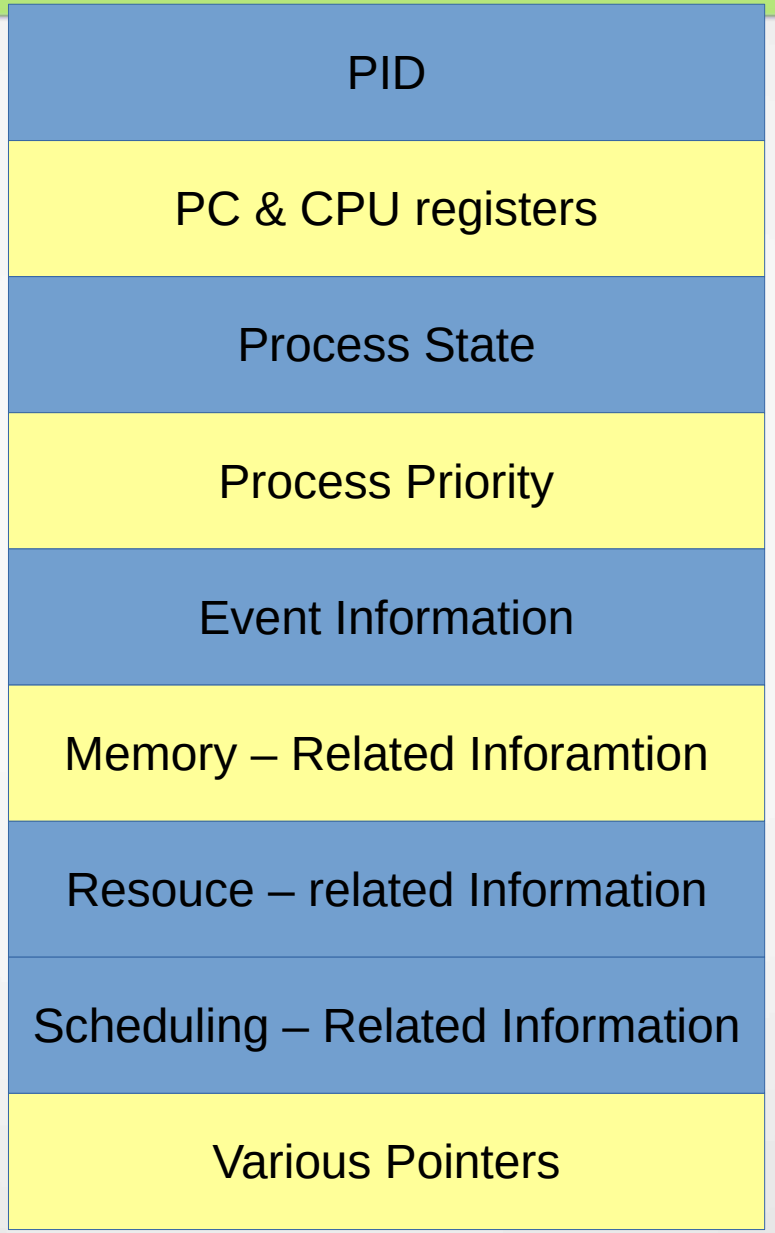
UNIT -2 Process Control Block (PCB)

- Process environment consists of program code, data section and stack.
- But **these are not sufficient to control a process**.
- The **OS needs some attributes associated with the processes to implement and control them.**
- The attributes are stored in the data structure known as **Process Control Blcok (PCB).**

UNIT -2 Process Control Block (PCB)

- The **PCB is created for every process** whenever a new process is created in the system.
- It also **deleted as the process is terminated.**
- The **PCB contains all the information about the process needed for its control.**
- For e.g – for the identification of process, it must have PID.
  - Current state of Process and its PC value
  - Process priority
  - The information related to resources held by the process and accounting information such as CPU time used.

UNIT -2 Process Control Block (PCB)



UNIT -2 Process Control Block (PCB)

- **PID**
  - It is a **unique identification** number of the process
- **PC**
  - Indicates the **address value at which the next instruction of the process will be executed** by the processor.
- **Registers**
  - CPU **registers are used for the execution of a process.** While the process is in execution, **data registers, address registers, control and status registers are used for executing and controlling the process.**
  - The **registers information must be saved when there is a state change of the process** so that it may resume its execution when its next turn comes.

UNIT -2 Process Control Block (PCB)

- **State**
  - A process **has a number of states in its life. For scheduling the processes, the current state of a process must be known.**
- **Priority**
  - The **priority number can be assigned to a process** to give preference to it over other.
- **Event Information**
  - This is the **event for which a blocked process is waiting. If the awaited event is over, the information regarding this event must be stored in this field so that the status of the blocked process is changed to ready.**

UNIT -2 Process Control Block (PCB)

- **Memory – Related Information**
  - Memory management component of the **OS uses many registers and tables. The information regarding all this memory related information linked to a process** is also mentioned in the PCB.
- **Resource Related Information**
  - **The resources allocated to a process are listed here.** For e.g all files opened by this process are listed in this filed.

UNIT -2 Process Control Block (PCB)

- **Scheduling – Related Information**
  - A process will be executed according to a scheduling algorithm. **The scheduling related information of a process is also stored such as the time the process has waited, the amount of time the process executed the last time it was running.**
- **Pointer to Parent Process**
  - If a process is a child process, then the pointer to its parent process is stored here.

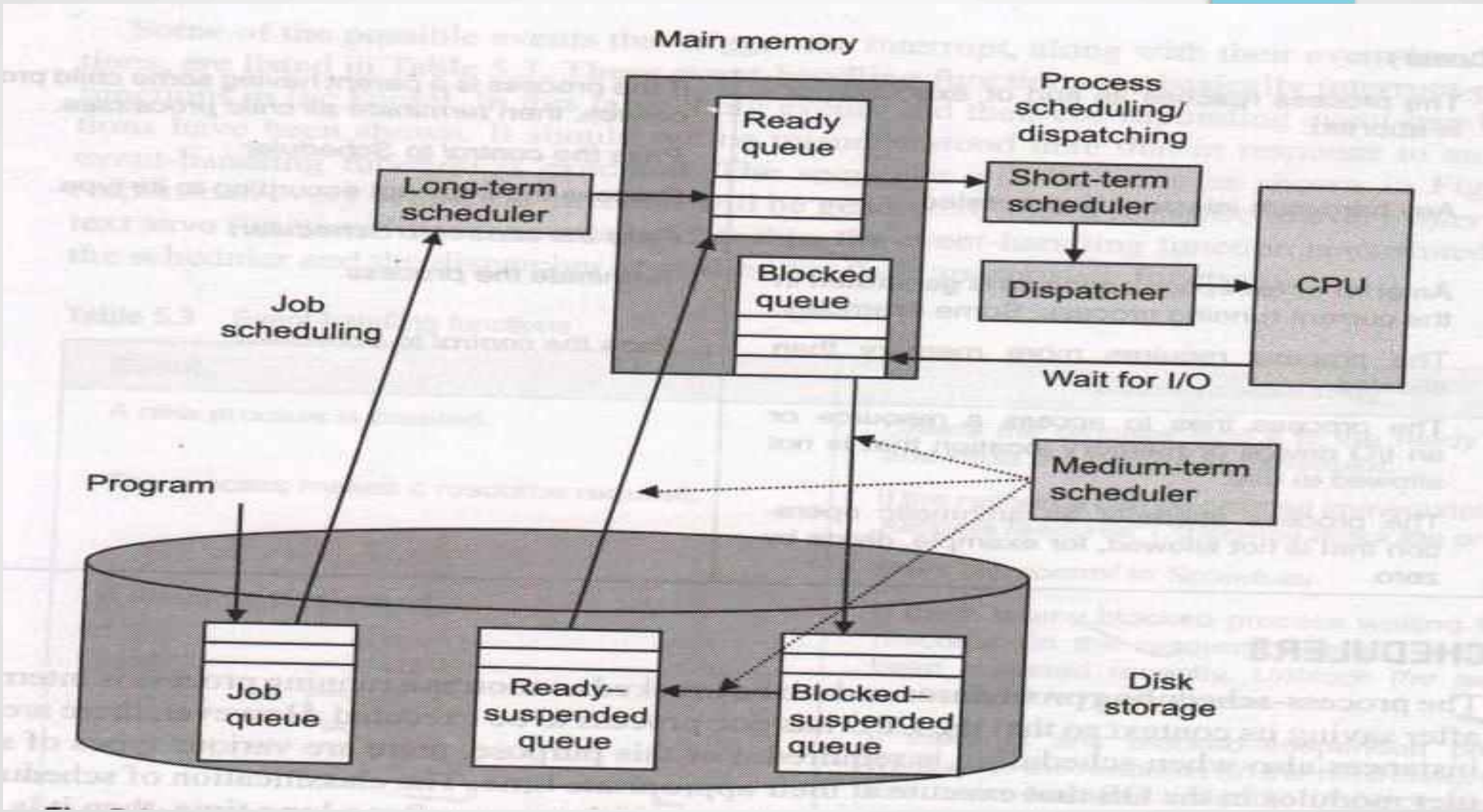
UNIT -2 Process Control Block (PCB)

- **Pointer to Child Process**
  - If a process has some child processes, then the pointer to its child processes is stored here.
- **Pointer to Address Space of the Process**
  - This is the pointer to the process data and instructions in the memory.

UNIT -2 Schedulers

- **The process scheduling procedure needs to be invoked as soon as a process changes its state.**
- The classification of schedulers is based on the frequency of their use in the system.
  - **Long – Term Schedulers**
  - **Short – Term Schedulers**
  - **Medium – Term Schedulers**

UNIT -2 Schedulers



UNIT -2 Long – Term Schedulers

- **This schedulers is invoked when there is a need to perform job scheduling, that is, when a job from the job pool is selected to be sent to the ready queue.**
- Since a job entering the system needs to be process in the ready queue, this scheduler is invoked whenever there is a need to increase the degree of multi proramming in the system.

UNIT -2 Short – Term Schedulers

- This scheduler is invoked when there is a need to perform process scheduling, **that is when a process from the ready queue is to be selected for dispatching to the CPU.**
- There are various instances in the system when this type of scheduling is needed.
- Whenever **there is an interruupt, the running process stops, and the short term scheduler is invoked every time to select another process for execution.**
- That **is why this scheduler is called a short term scheduler.**