Operating System

Prof. Vibhuti Patel, Assistant Professor
Parul Institute of Computer Application

Parul®
University

DIGITAL LEARNING CONTENT

CHAPTER-1

Operating System Overview & Processes

What is Operating System?

- An operating system (OS) is a collection of software that manages computer hardware resources and provides general services to computer programs.
- The operating system is an important part of the system software in a computer system.
- Application programs usually require the operating system to work.
- In simple terms it is the executive manager, the part of the computing system that manages all the hardware and all the software.
- It provides a basis for application programs and acts as an intermediary between the computer user and the computer hardware.

Computer Components

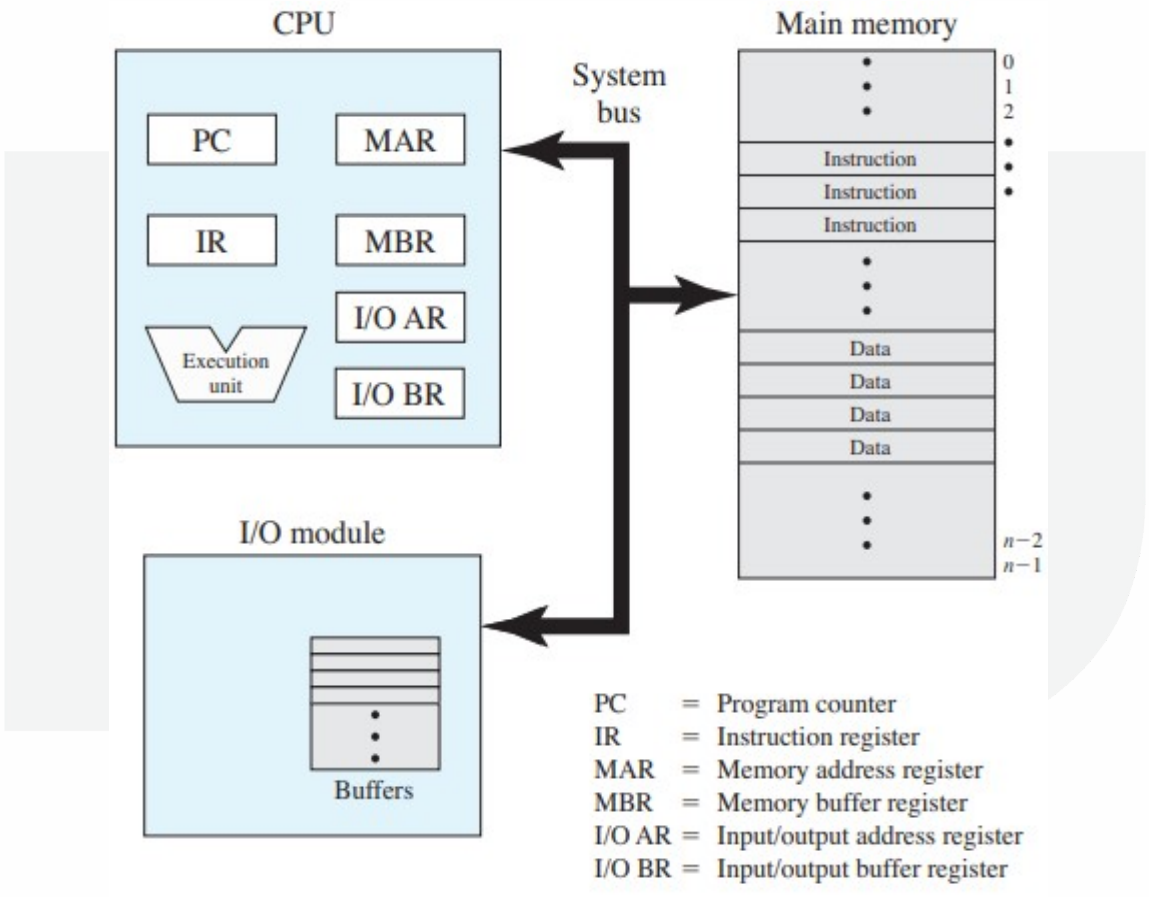


Image source : Book

Basic Elements

Processor:

Internal Registers:

Memory address register (MAR):

- Specifies the address for further reading or writing.

Memory buffer register (MBR):

- Contains data written in memory or reads data from memory.

External registers:

I / O address register (I / OAR)

- specifies a particular I / O device.

I / O buffer register (I / OBR)

- Used to transfer data between the I / O module and the processor.

Image source : Google

Parul® University

DIGITAL LEARNING CONTENT

Basic Elements

Main Memory (RAM)

- Volatile
- Refers to actual memory or primary memory
- Storage of data and programs.
- when the computer is shut down, the contents of the memory are lost.

Basic Elements

I / O Modules:

- Secondary Memory Devices
- Communications Tools
- Terminals
- Contains a variety of devices
- Disks

Basic Elements

System bus

Communication between processors, main memory and I / O modules.

OS Functions

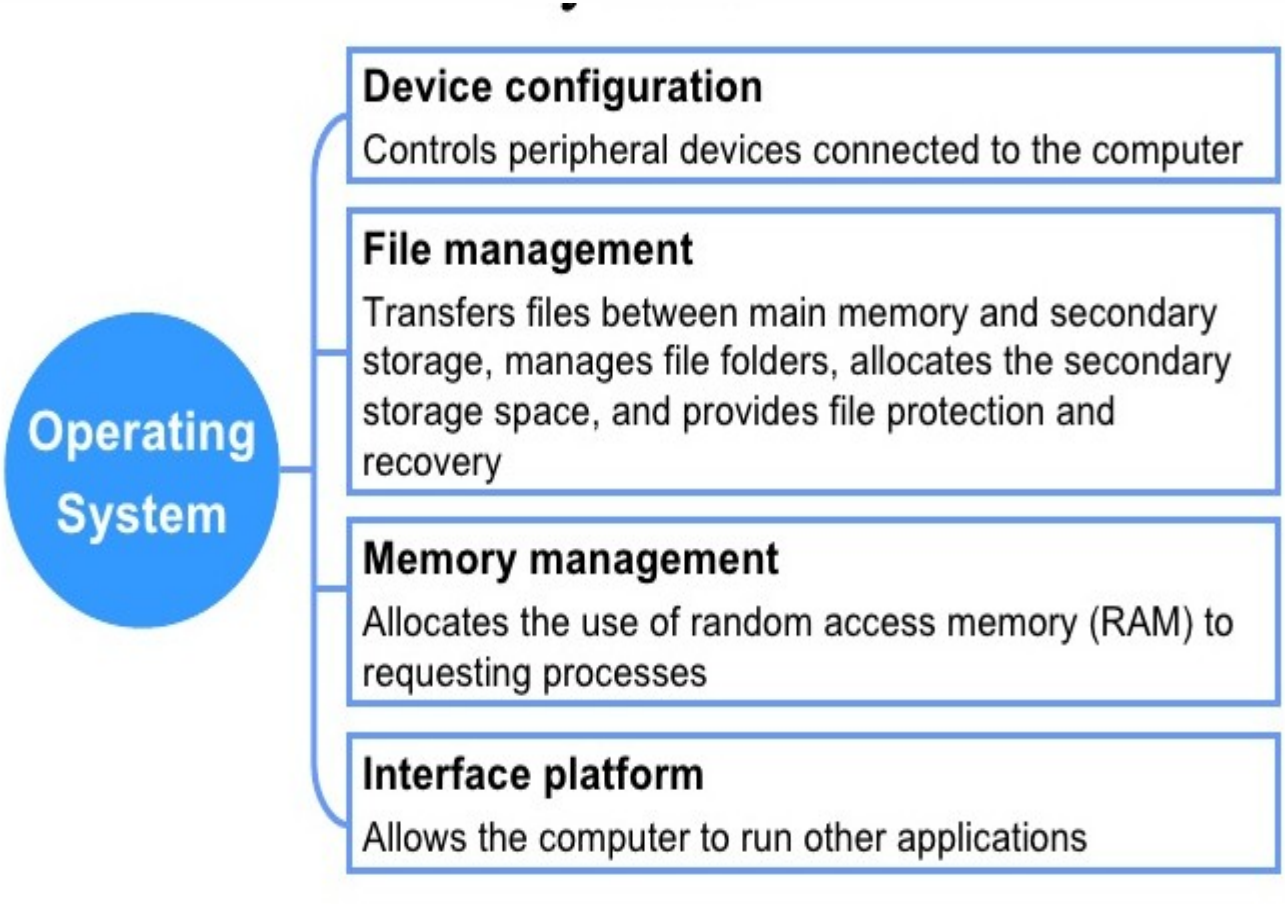


Image source : Google

Parul® University

DIGITAL LEARNING CONTENT

Other OS Functions

- Proper utilization of computer resources
- Provide background for running user programs
- Error Demonstrate and manage errors
- Control the selection and operation of IP peripherals
- Users act as a communication link between users and hardware
- Defense system protection, i.e. security

Parul®
University

DIGITAL LEARNING CONTENT

Features of Operating System

- Multi - Tasking
- Multi Programming
- Parallel Processing
- Buffering

Features of Operating System

Multi - Tasking: Conducting two or more programs at the same time from the understanding of a single user.

•The CPU can only do one task at a time, however, it is very fast and can do two or more tasks at the same time.

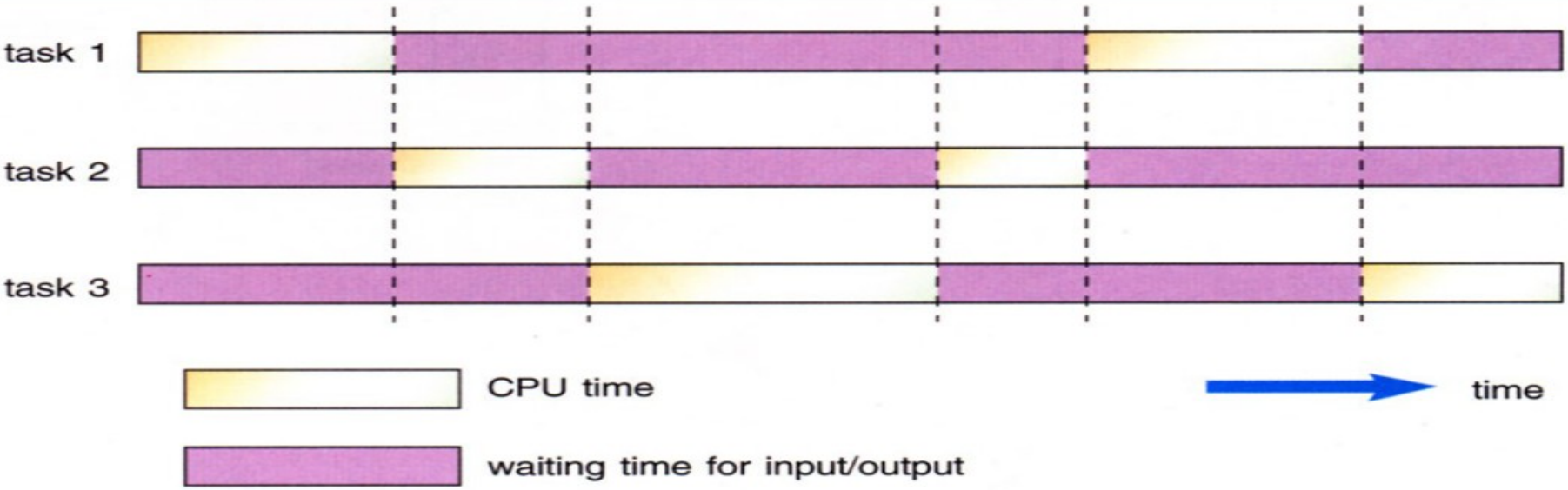


Image source : BOOK

Features of Operating System

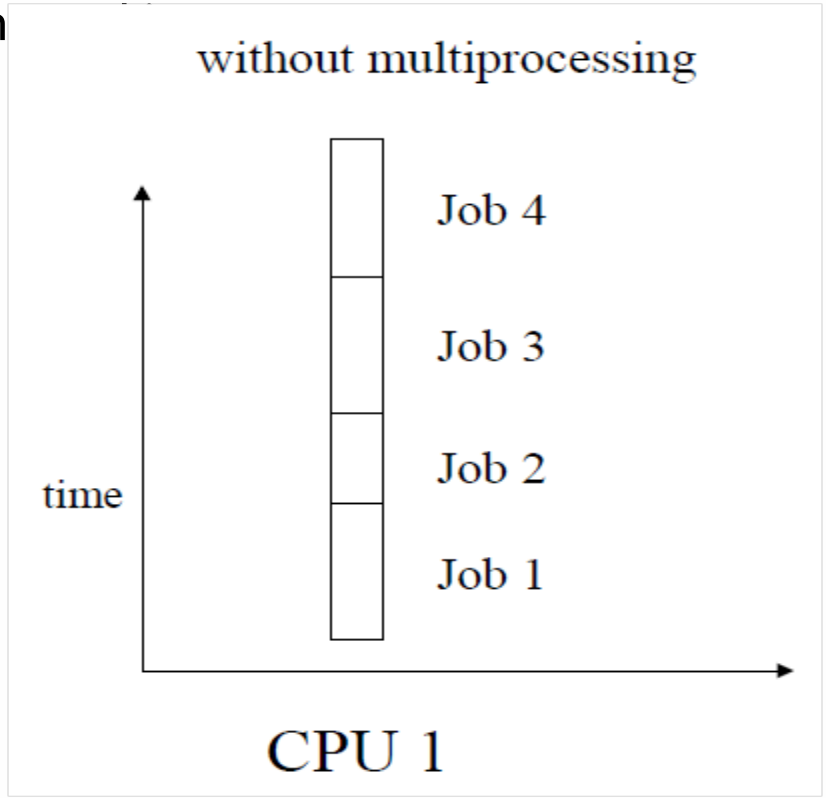
Multi Programming: Two or more programs are stored in main memory at the same time.

- When the job has to wait (e.g. I / O operation), the CPU switches to another job to run.
- When the first job expectation is completed, the first job to be executed is returned to the CPU.

Image source : Google

Features of Operating System

Parallel Processing: Use two or more CPUs to manage working computer n



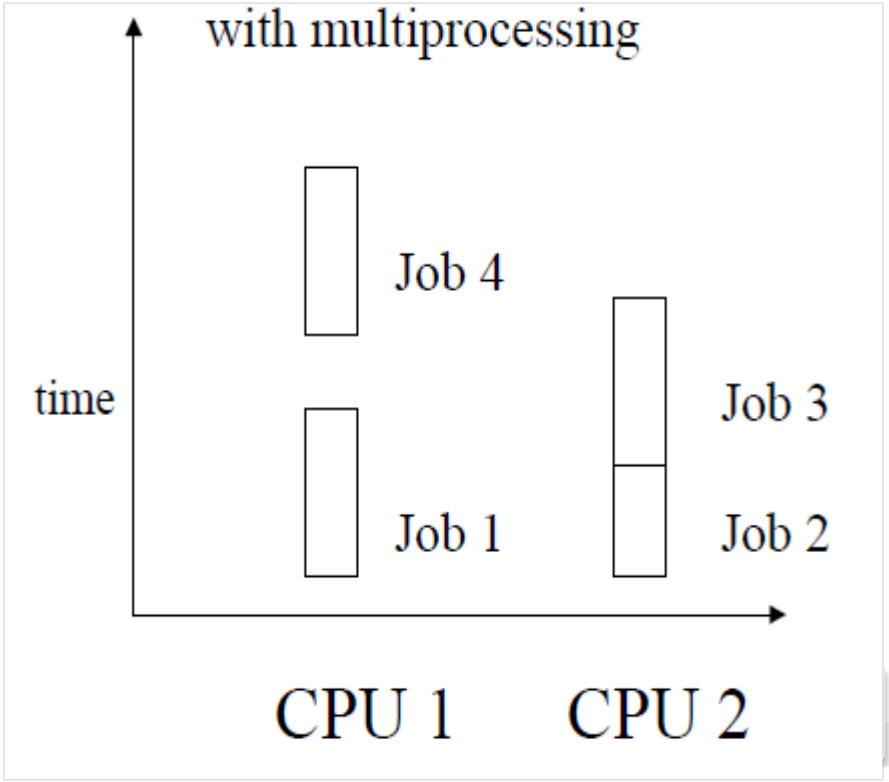


Image source : Google

Features of Operating System

Buffering: Temporary storage area (buffers) for reading data from the input device or sending data to the output device.

The CPU is busy because I / O performance is slow.

Parul®
University

DIGITAL LEARNING CONTENT

Essential Managers of Every Operating System

1. Memory Management
2. Processor Management
3. Device Management
4. File Management

Essential Managers of Every Operating System

1. **Memory Management:** Memory management refers to primary memory or primary memory management. Main memory provides fast storage with direct access to the CPU. So to run a program, it must be in main memory.

- The operating system performs the following functions for memory management.

1. Memory keeps track of the primary memory, i.e. which part is in use and which part is not in use.
2. In multiprogramming, the OS determines when and how much memory to process.
3. The request process allows memory when requested.
4. This process de-allocates when memory is not needed.

Image source : Google

Essential Managers of Every Operating System

2. Process Management:

- In multiprogramming environment, OS decides which process gets the processor when and how much time. This function is called process scheduling.

- The operating system performs the following functions for processor maintenance.

1. Process Tracks the processor and the status of the process. The program responsible for this task is called the traffic controller.
2. Allocates processor (CPU) to a process.
3. Processor de-allocations when the processor is no longer needed.

Essential Managers of Every Operating System

3. Device Management:

- The OS controls device communication through related drivers.
- The operating system performs the following functions for device maintenance.

1. Tracks all devices.
2. Determines when and what process the device will receive.
3. Allocate the device
4. De-Allocate devices

Essential Managers of Every Operating System

4. File Management :

- The file system is usually organized as directories for navigation and use. These directories may contain files.

- The operating system performs the following functions for file management.

1. Tracks information, location, uses, status, etc.
2. Decides who gets the resources.
3. Allocates files to directories.
4. De-allocates files to directories.

Types of Operating System

1. Batch Operating System
2. Time-sharing Operating System (Interactive System)
3. Real-Time Operating System
4. Multi Threading Operating System

Types of Operating System

1. Batch Operating System:

- BOP has two or more functions:

1. CPU time
2. I / O time (input / output operations)

- Users of the batch operating system do not interact directly with the computer.
- To speed up processing, work together as a group of tasks with similar requirements.

Image source : Google

Types of Operating System

Batch Operating System:

- This type of operating system does not interact directly with the computer. There is an operator who takes similar jobs with similar requirements and classifies them into batches. It is the operator's responsibility to classify tasks with similar requirements.

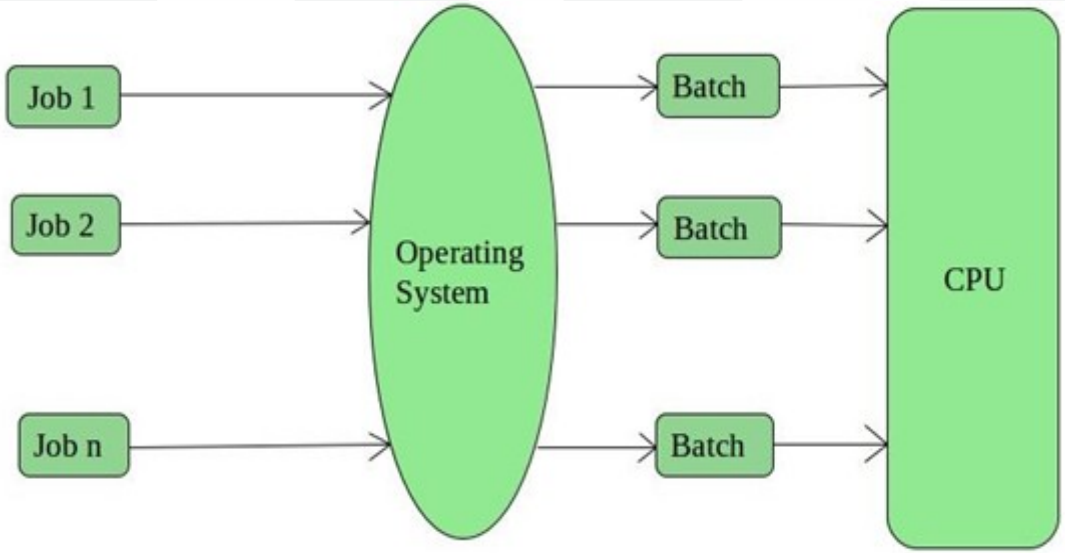


Image source : Google

Types of Operating System

Benefits of Batch Operating System:

- It is very difficult to know the time required to complete any job. Batch system processors know how long it takes to work while queuing.
- Users Multiple users can share batch systems
- The idle time for the batch system is very short
- In batch systems it is easy to perform large tasks repeatedly.

Disadvantages of Batch Operating System:

- Computer operators must be familiar with batch systems
- Batch systems are difficult to debug
- It is sometime expensive
- If any job fails, other jobs will have to wait for an unknown time

Examples of Batch based Operating System: Payroll, Bank Statements etc

Types of Operating System

2. Time Sharing Operating System :

- Time sharing system sharing allows multiple users to share computer resources.
- In other words, time sharing refers to the allocation of computer resources to several programs at once in a time slot.
- Sharing Time Sharing system is an operating system that gives some time to each task and time to all tasks so that all processes run smoothly without any problems.

Types of Operating System

2. Time Sharing Operating System :

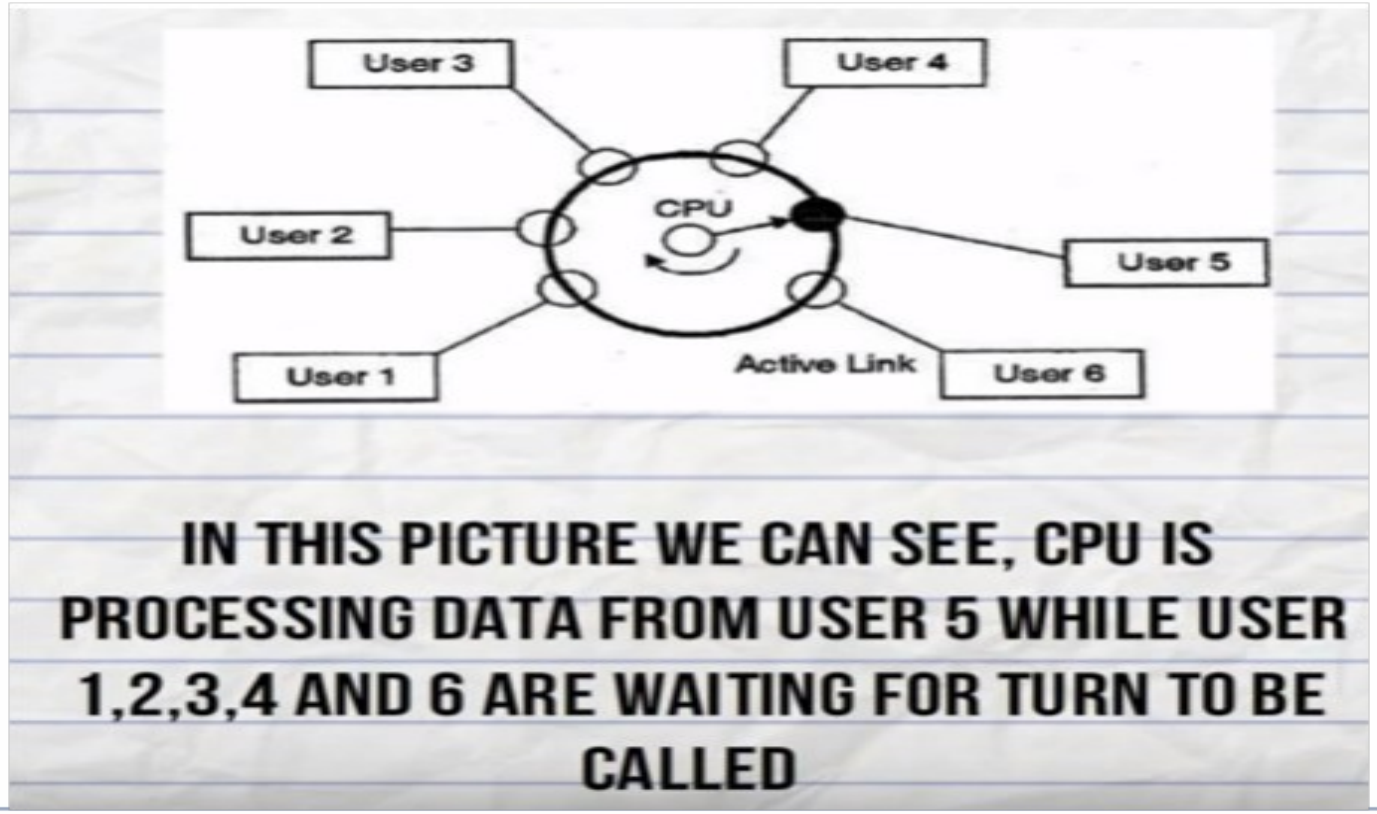


Image source : Google

Types of Operating System

2. Time Sharing Operating System :

- In the second image example, we see that the CPU controls a wide variety of data released by the computer, central storage and shared printer.
- This time the sharing system simplifies personal tasks, making it easier to store data.

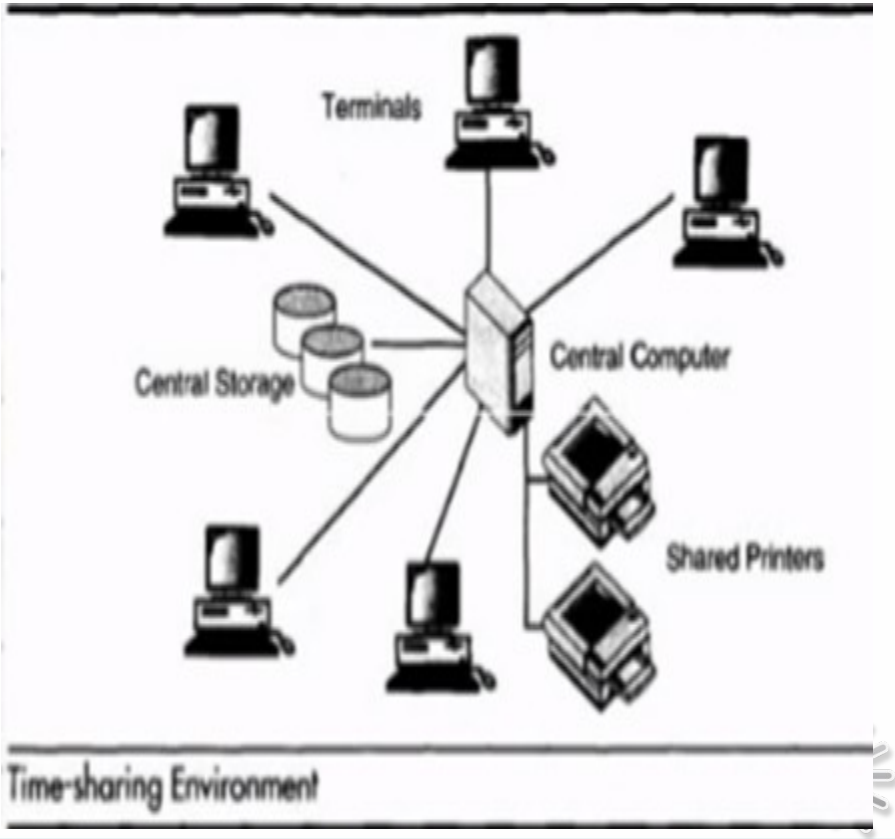


Image source : Google

Types of Operating System

Benefits of Time Sharing Operating System :

- Assigns a time limit for each task
- Users can access one terminal simultaneously
  - Multiprogramming

Disadvantages of Time Sharing Operating System :

- Lack of protection against the file
- Communication problem with data

Types of Operating System

3. Real Time Operating System:

- RTOS is the brain of a real time system its response to input immediately.
- In RTOS, the task will be completed by specifically timed its responses in a predictable way to un-predictable way.
- The time taken by the system to respond to an updated input and display the required updated information is called response time.

- **Examples include scientific experiments, medical imaging systems, industrial control systems, weapons systems, robots, home controllers, and air traffic control systems.**

Types of Operating System

There are two types of Real Time Operating Systems:

Hard Real-Time Systems:

- This OS allows for applications where time limits are very strict & even the shortest possible delay is not acceptable. These systems are designed to save life, such as automatic parachutes or air bags which are readily available in case of any accident. Virtual memory is never appearing on these systems.

Soft Real-Time Systems:

- These OSs are for those applications where there is less time-constraint.

Types of Operating System

Benefits of Real Time Operating System:

- **Maximum Utilization:** Maximum utilization of equipment and system, thereby maximizing output from all resources
- **Task Shifting:** Time assigned for shifting tasks in these systems are very less. For example, in older systems it takes about 10 microseconds to transfer one task to another and in latest systems it takes 3 microseconds.
- **Focus on Application:** Focus on applications that are running and less importance to applications which are in queue.
- **Real time operating system in embedded system:** Due to small size of programs, RTOS can also be used in embedded systems such as transport and others.
- **Error Free:** These types of systems are error-free.
- **Memory Allocation:** Memory allocation is best managed on these type of systems.

Types of Operating System

Disadvantages of Real Time Operating System:

- **Limited Tasks:** Very few tasks run at the same time, their concentration is very low on some applications to avoid errors.
- **Use heavy system resources:** Sometimes system resources are not only good, but also expensive.
- **Complex Algorithms:** Algorithms are very complex & difficult for designer to write.
- **Device driver and interrupt signals:** Specific device drivers are required and interrupt signals to response earliest to interrupts.
- **Thread Priority:** It is not advisable to set thread priority as these systems are less prone to switching tasks.

Examples of Real Time Operating Systems are: Scientific experiments, medical imaging systems, robots and air traffic control systems

Types of Operating System

4. Multi Threading Operating System:

- Thread is a part of process. Hence it is called as light weight.
- Threads provide a way to improve application performance through parallelism.
- Threads refers to a software approach to improving operating system performance by minimizing the overhead thread.

What is Process?

- A program in execution
- An instance of a program running on a computer
- An entity that can be allocate to and execute on a processor
- Unit of activity characterized by the execution of a series of instructions,
- A current state, and an associated set of system resources
- A process is basically a program in execution. The execution of a process must progress in a sequential manner.
- A process is an instance of a program, or part of a program in memory. Processes are executed by the operating system to perform complex tasks: play a movie or video, play a game, or run an editor used to write this text.

Process State Diagram

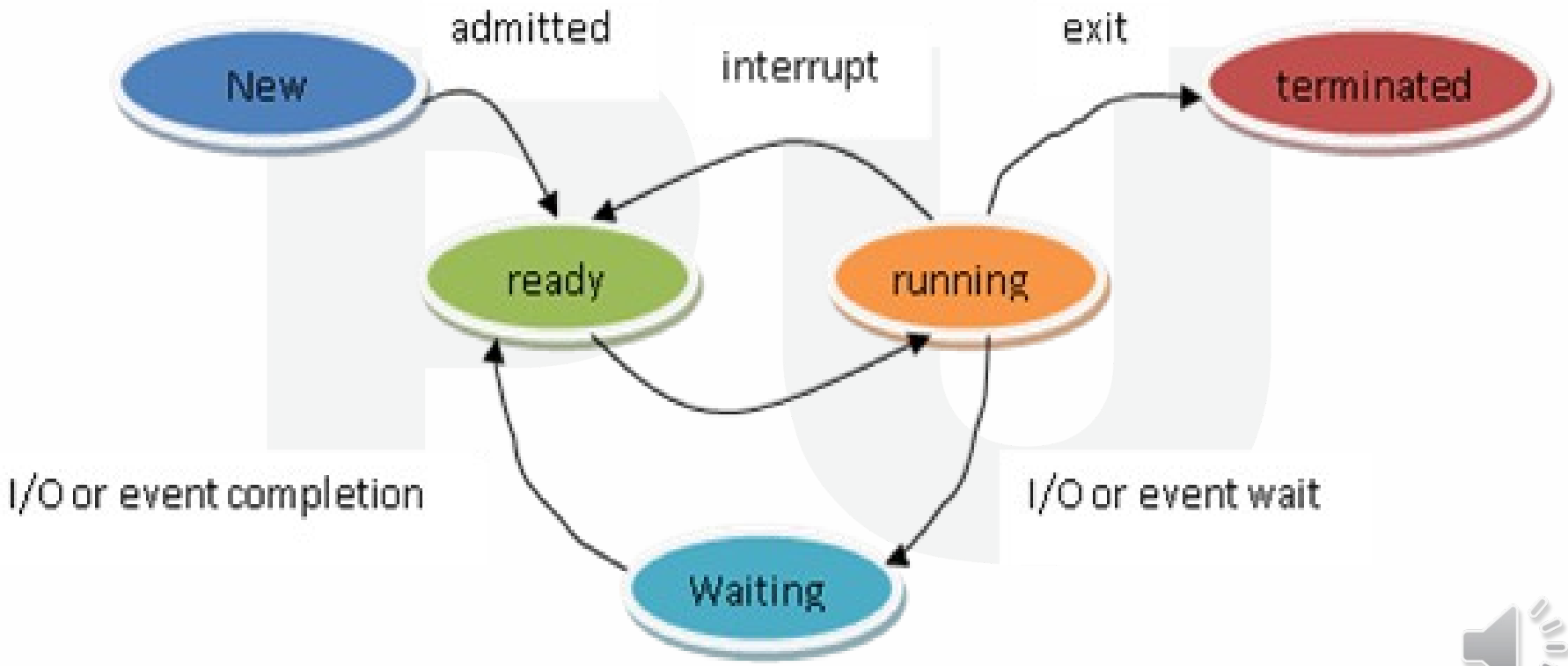


Image source : Google

Process State Diagram

The process changes its state during execution. Each process can be in one of the following states:

- **New:** When creating a new process.
- **Running:** A process is said to be in running while instructions are being executed.
- **Waiting:** The process is waiting for some event to happen (such as an I / O operation).
- **Ready:** The process is waiting for processor.
- **Terminated:** Process finished its execution.

Process Control Block

- Operating system maintains a unique data structure called the PCB (Process Control Block).
- All the information related to each process is stored in the PCB & maintained by OS.

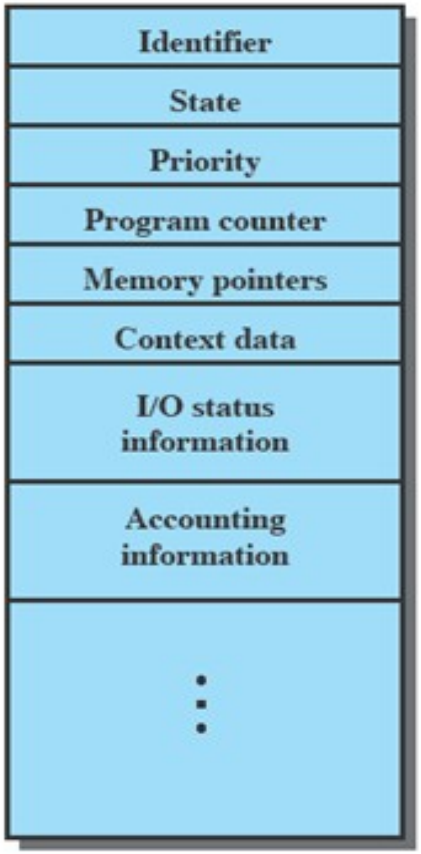


Image source : Google

Process Control Block

- **Identifier:** A unique identifier associated with this process, to distinguish it from all other processes.
- **State:** If the process is currently running, it is in running state.
- **Priority:** Priority level compared to other processes.
- **Program counter:** The address of the next instruction in the program to be executed.
- **Memory pointers:** Add pointers to the program code & data associated with this process, plus any memory blocks shared with other processes.
- **Context data:** This is the data that are present in processor registers while the process is running.

Process Control Block

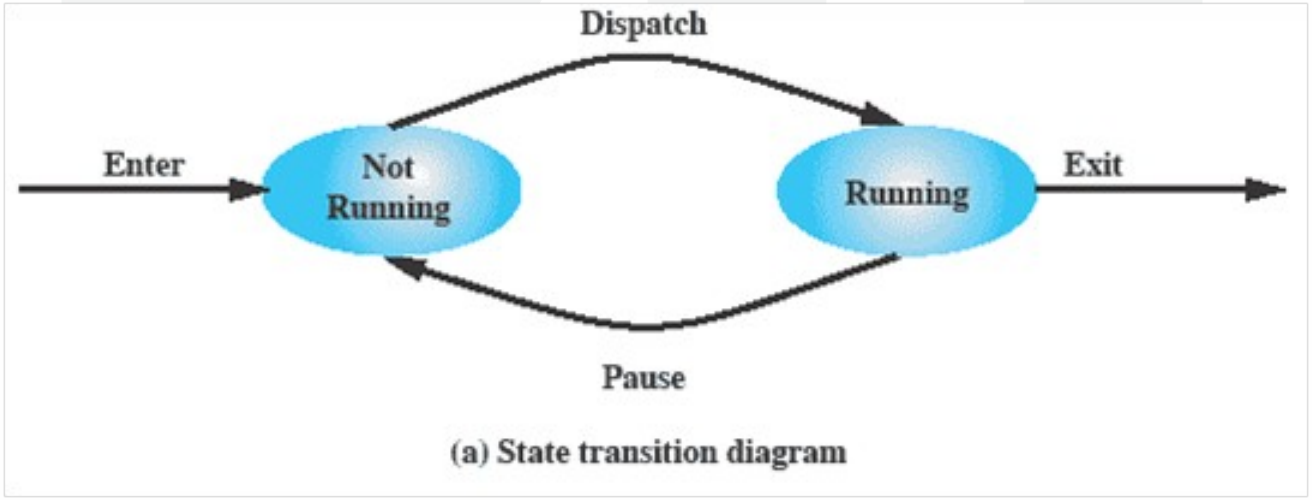
- **I / O status information:** It contains I / O requests, I / O devices (eg, tape drives) assigned to this process, a list of files in use by the process.
- **Accounting information:** It may contain processor time and clock time, time limits, account numbers, and so on.

Two State Process Block

Process may be in one of two states:

- Running
- Not Running

In the simplest model, a process is either running, or it is not.

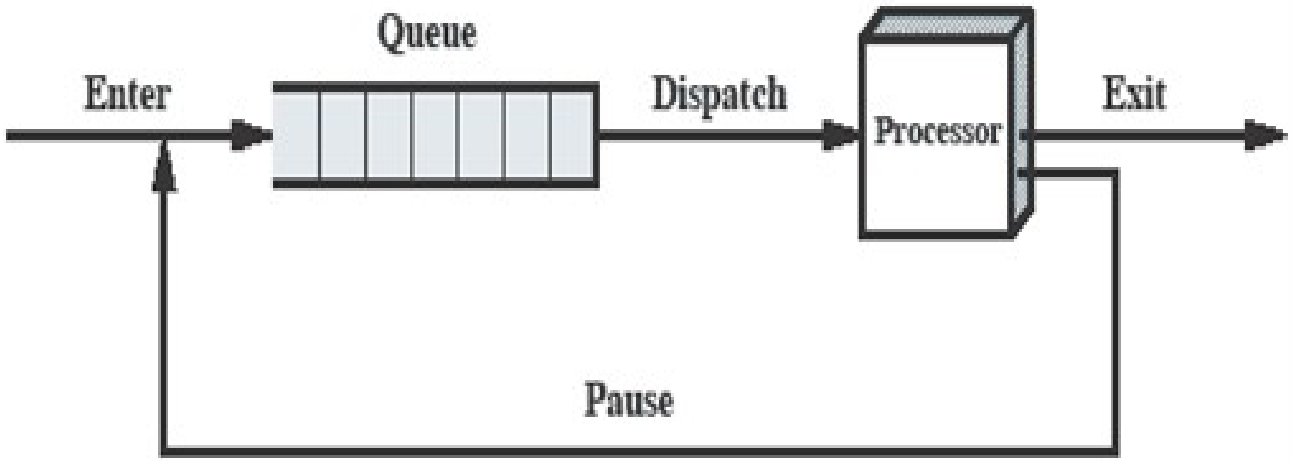


(a) State transition diagram

Image source : Book

Queuing Diagram

Processes moved by the dispatcher of the OS to CPU then back to the queue until the task is competed.



(b) Queuing diagram

Image source : Book

Five State Process Model

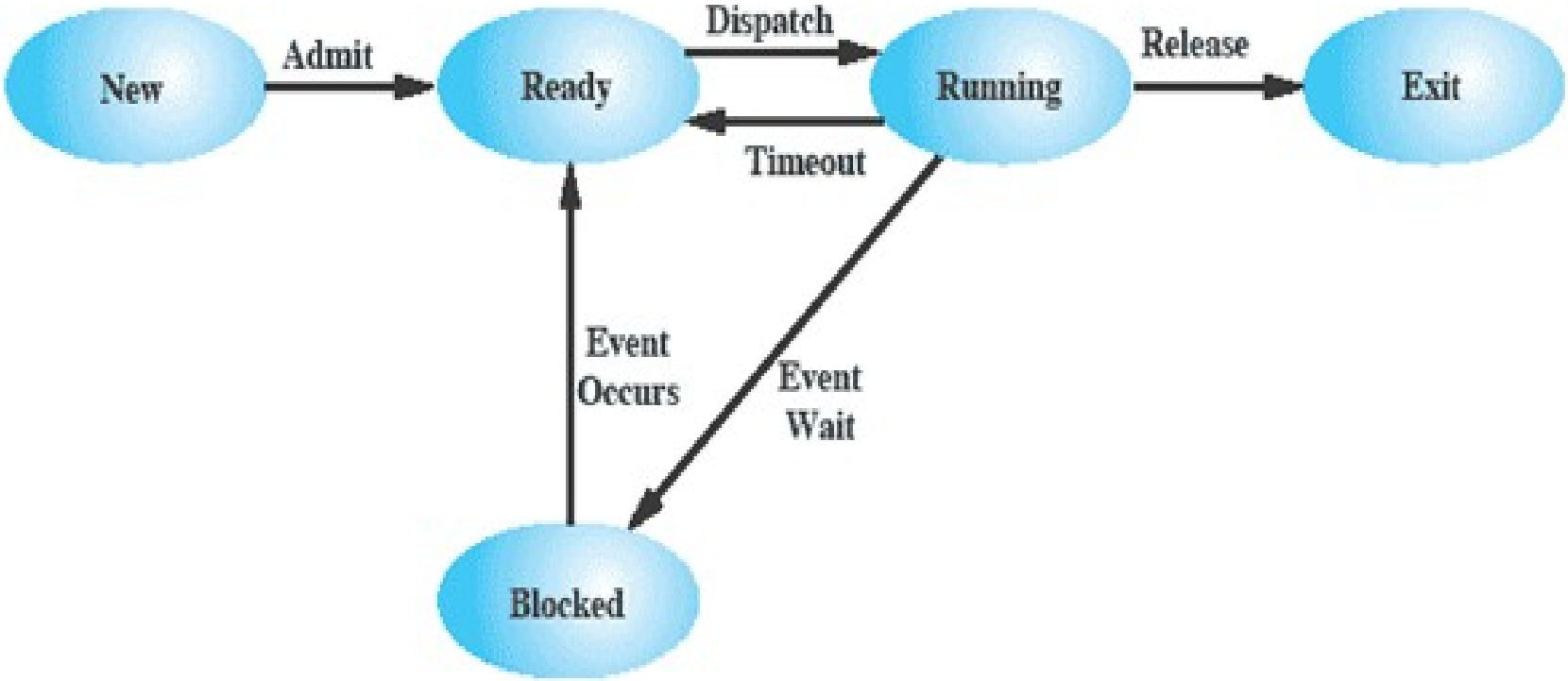


Image source : Book

Five State Process Model
k

- **New:** A process that has just been created but has not yet been entered to the pool of executable processes by the OS.
- **Ready:** A process that is ready to run when given the opportunity.
- **Running:** The process that is currently being executed.
- **Blocked / Waiting:** A process that cannot execute until some event occurs, such as the completion of an I / O operation.
- **Exit:** The process that has been released from the pool of executable processes by the OS, either because it halted or because it aborted for some reason.

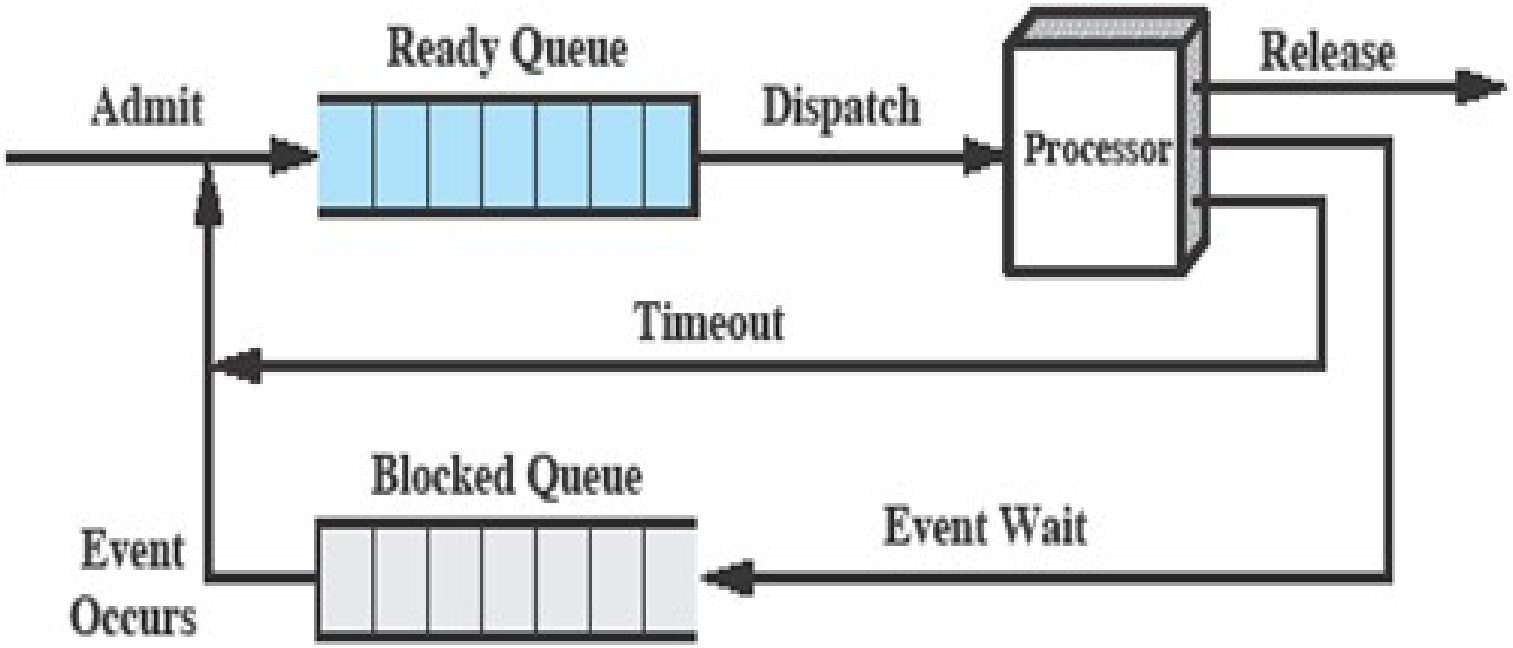
Five State Process Model

- **Null -> New:** A new process has been created to execute a program.
- **New -> Ready:** The OS will move a process from the New state to the Ready state when it is ready to perform an additional process.
- **Ready -> Running:** When it is time to select a process to run, the OS will select one of the processes in the Ready state. This is the job of the scheduler or dispatcher.
- **Running -> Exit:** The currently running process is terminated by the OS if the process indicates that it has completed, or if it aborts.
- **Running -> Ready:** The most common reason for this transition is that the running process has reached the maximum allowed time to run continuously.

Five State Process Model

- **Running -> Blocked:** A process is put in the Blocked state if it requests something for which it must wait.
- **Blocked -> Ready:** A process in the Blocked state is moved to the Ready state when waiting event occurs.

Using Two Queue



(a) Single blocked queue

Image source : Book

Thread

- Thread is a one part of process. Hence it is called as light weight.
- A thread is a basic unit of CPU utilization, that contains a program counter, a stack, and a set of registers, (and a thread ID.)
- Traditional (heavyweight) processes have a single thread of control - The program counter, and one series of instructions can be carried out at any time.

Single Threaded & Multithreaded Process

- As shown in the Figure, multi-threaded applications can have multiple threads in a single process, each with its own program counter, stack and set of registers, but sharing common code, data, and certain structures such as open files.

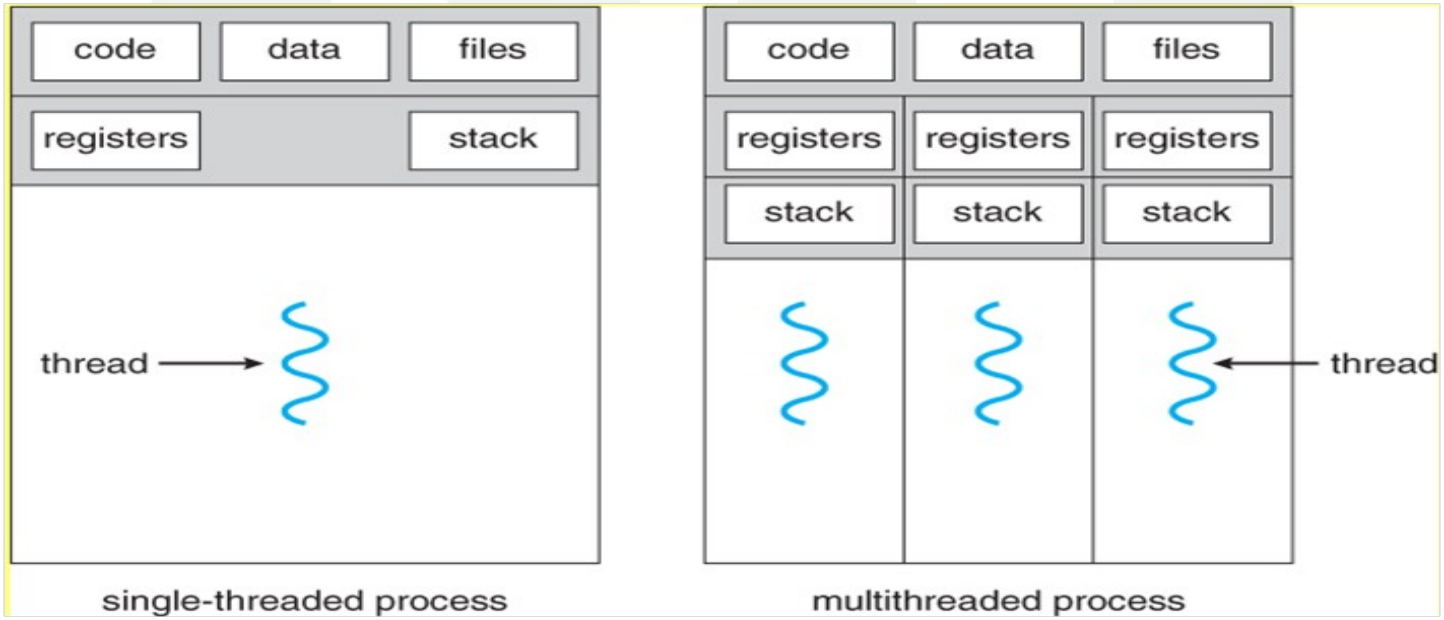


Image source : Book

Single Thread Approach

- MS-DOS supports a single user process and a single thread.
- Some UNIX, support multi-user processes, but only support one thread per process.

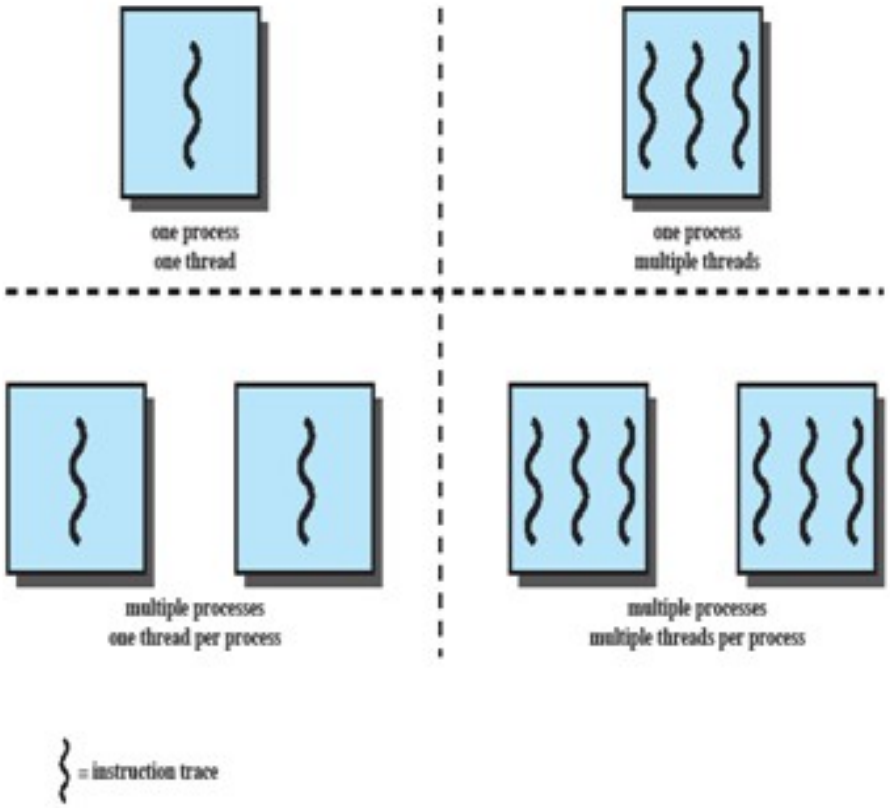


Image source : Book

Multithreading

- The ability of an OS to support multiple, concurrent paths of execution in a single process.
- Java runtime environment is a single process with multiple threads.
- Multiple processes and threads are appears in Windows, Solaris, and many modern versions of UNIX.

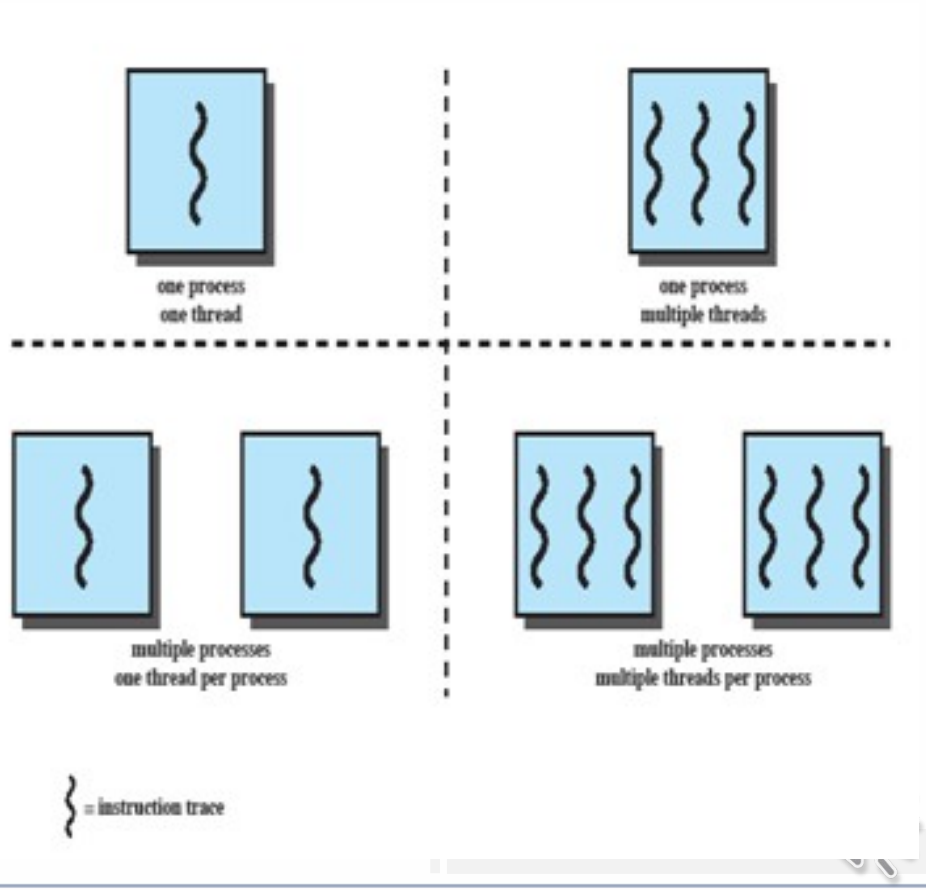


Image source : Book

Parul® University

DIGITAL LEARNING CONTENT

Benefits of Thread

- Takes less time to create a new thread than a process
- Less time to complete a thread than a process
- Switching between two threads takes less time that switching processes
- Threads can communicate with each other without using the kernel

Parul®
University

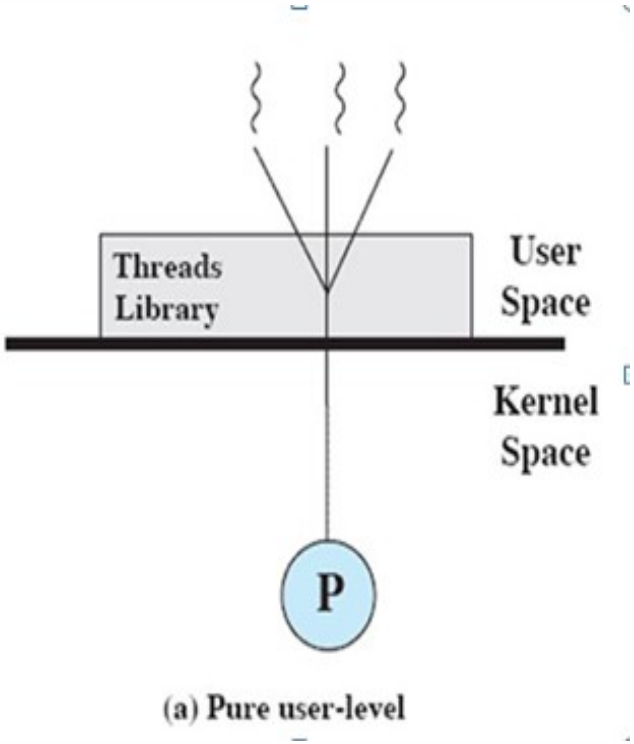
DIGITAL LEARNING CONTENT

Types of Thread

1. User Level Threads
2. Kernel Level Threads

1. User Level Thread

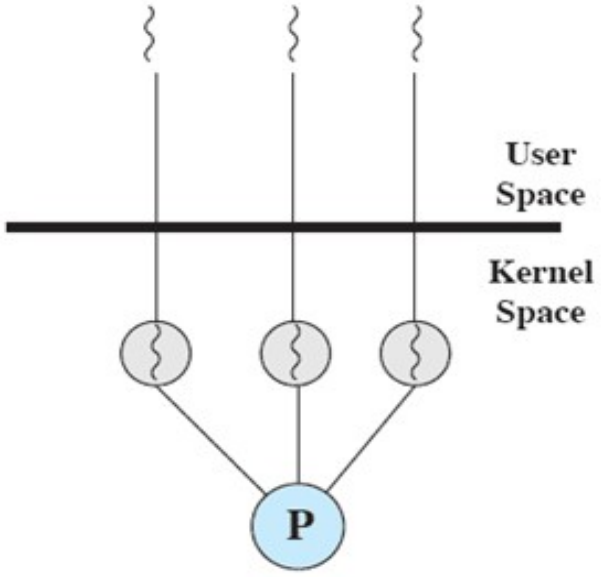
- All thread management is done by the application
- The kernel is unaware of the existence of user level threads.
- Therefore, all the user level threads handling, scheduling, context switching etc. is a part of process code, which is done without kernel environment.
- Therefore, user level threads are less overhead and much faster to create, manage and execute.



(a) Pure user-level

2. Kernel Level Thread

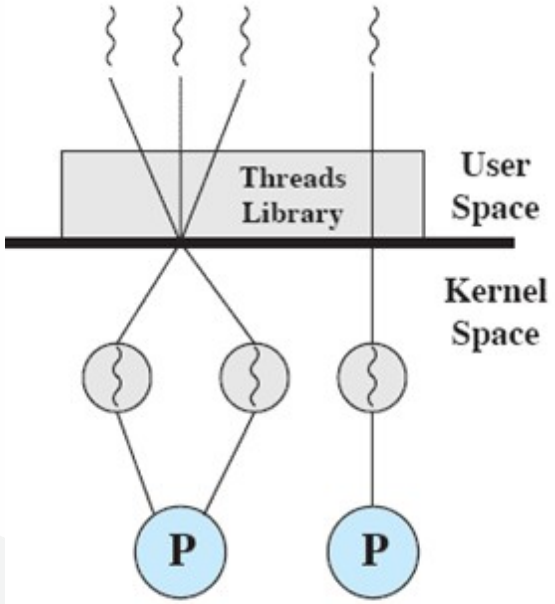
- Kernel maintains contextual information for process and threads
- No thread management done by application
- The schedule is done on a thread basis
- Windows is an example of this approach
- The kernel maintains thread table to keep a track of all threads in system.
- Kernel use system calls to create and manage threads.
- Kernel maintains all information about all the threads, so thread scheduling and managing can be done more efficiently.



(b) Pure kernel-level

3. Combined Approach

- Thread creation done in user space
- Bulk of scheduling and synchronization of threads by application
- Example: Solaris OS



(c) Combined

Difference between User Level & Kernel Level Thread

S.N.	User-Level Threads	Kernel-Level Thread
1	User-level threads are faster to create and manage.	Kernel-level threads are slower to create and manage.
2	Implementation is by a thread library at the user level.	Operating system supports creation of Kernel threads.
3	User-level thread is generic and can run on any operating system.	Kernel-level thread is specific to the operating system.
4	Multi-threaded applications cannot take advantage of multiprocessing.	Kernel routines themselves can be multithreaded.

DIGITAL LEARNING CONTENT

Parul® University

www.paruluniversity.ac.in