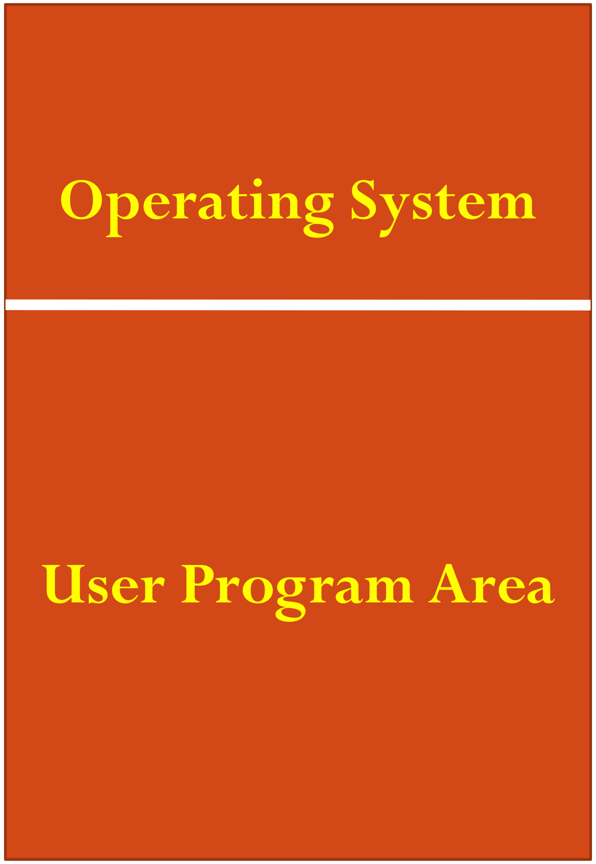
Types of Operating System

3. Batch Processing System

- The OS in the early computers was fairly simple.
- Its major task was to transfer control automatically from one job to the next.
- The OS was always resident in memory.
- To speed up processing, operators batched together jobs with similar requirement/needs and ran them through the computer as a group. Thus, the programmers would leave their programs with the operator.
- The operator would sort programs into batches with similar requirements and, as the computer became available, would run each batch.
- The output from each job would be sent back to the appropriate programmer.

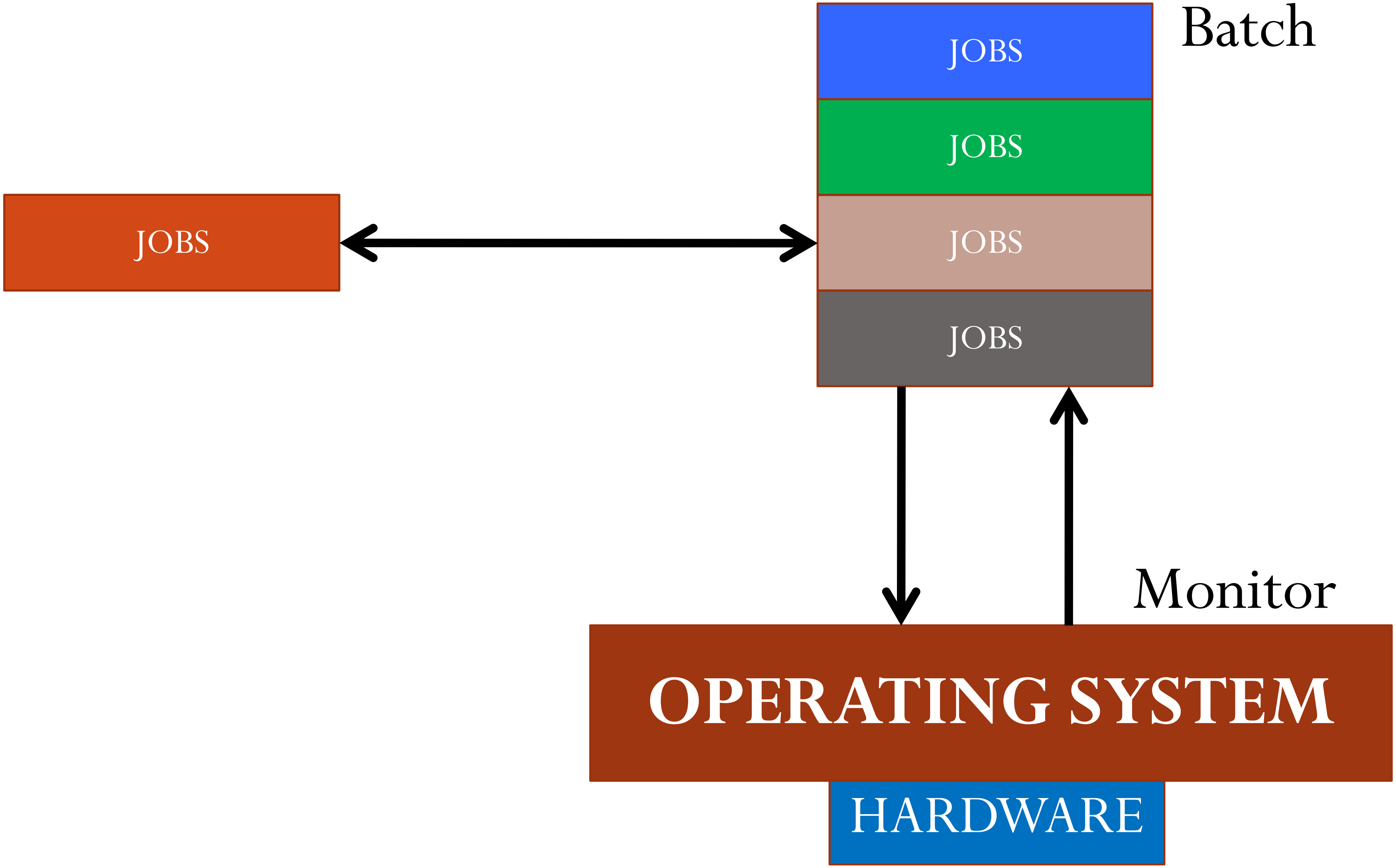
Memory layout of Batch System



Operating System

User Program Area

Batch Processing Operating System



Advantages:

- Batch processing system is particularly useful for operations that require the computer or a peripheral device for an extended period of time with very little user interaction.
- Increased performance as it was possible for job to start as soon as previous job is finished without any manual intervention.
- Priorities can be set for different batches.

Disadvantages:

- No interaction is possible with the user while the program is being executed.
- In this execution environment , the CPU is often idle, because the speeds of the mechanical I/O devices are intrinsically slower than are those of electronic devices.

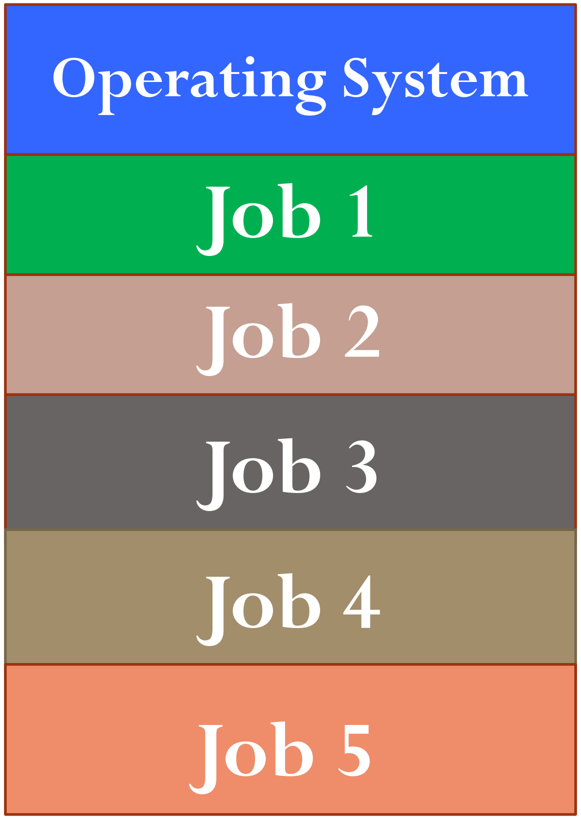
4. Multiprogramming System

- The most important aspect of job scheduling is the ability to multi-program.
- Multiprogramming increases CPU utilization by organizing jobs so that CPU always has one to execute.
- The idea is as follows: The operating system keeps several jobs in memory simultaneously. This set of jobs is a subset of the jobs kept in the job-pool. Since the number of jobs that can be in the job pool.
- The operating system picks and begins to execute one of the jobs in the memory.
- Eventually, the job may have to wait for some task, such as an I/O operation, to complete.
- In a multiprogramming system, the OS simply switches to, and executes another job. When that job needs to wait, the CPU is switched to another job and so on. Eventually the first job finishes waiting and gets the CPU back.
- As long as at-least one job needs to execute, the CPU is never idle.

Multiprogramming (Continued…)

- Multiprogramming is the first instance where the OS must make decisions for the users (Making this decision is Job Scheduling).
- All the jobs that enter the system are kept in the job pool. This pool consists of all processes residing on disk awaiting allocation of main memory.
- If several jobs are ready to run at the same time, the system must choose among them. Making this decision is job scheduling.
- Finally multiple jobs running concurrently require that their ability to affect one another be limited in all phases of the operating system, including process scheduling, disk storage and memory management.

Memory layout for multiprogramming system



Operating System

Job 1

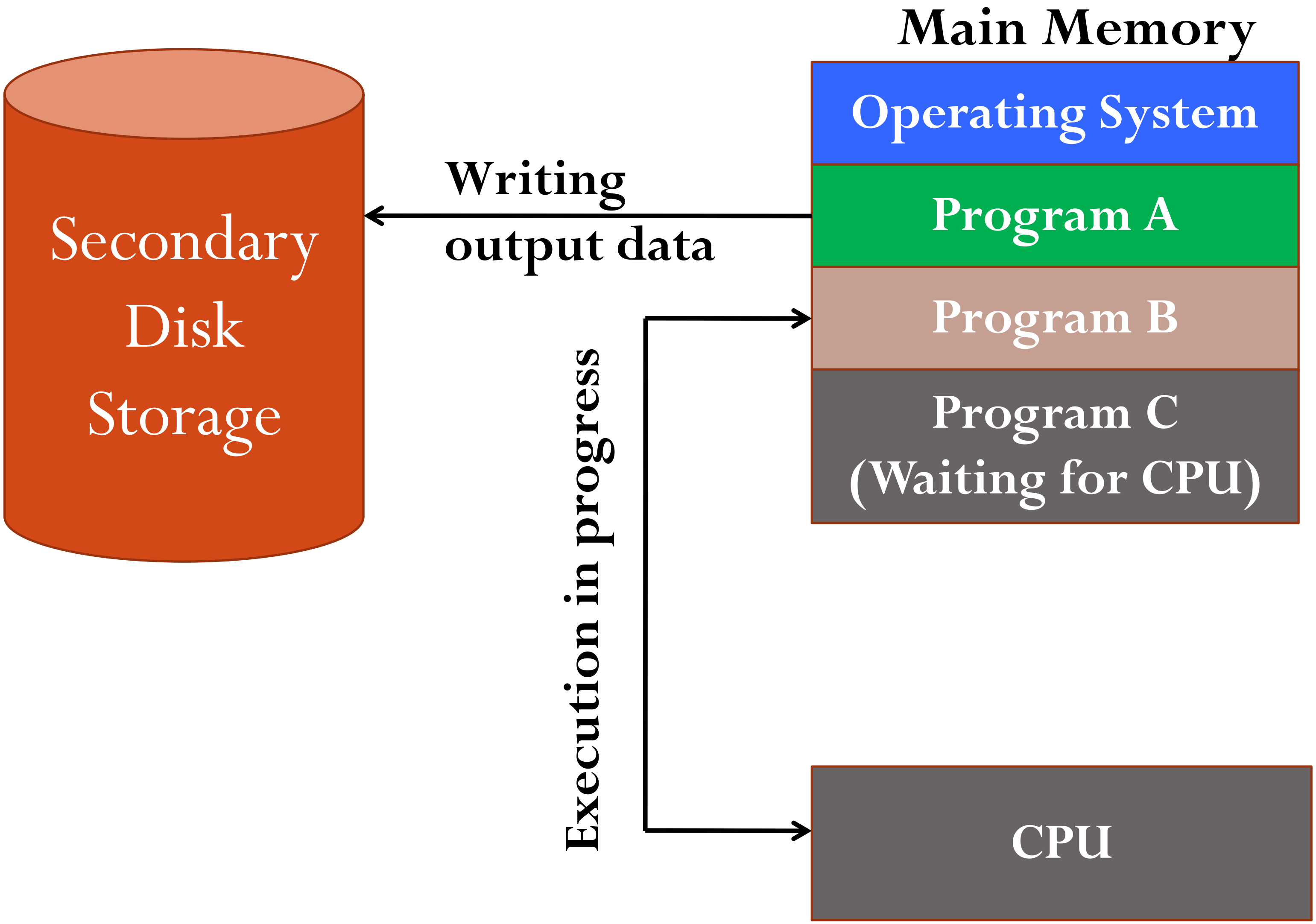
Job 2

Job 3

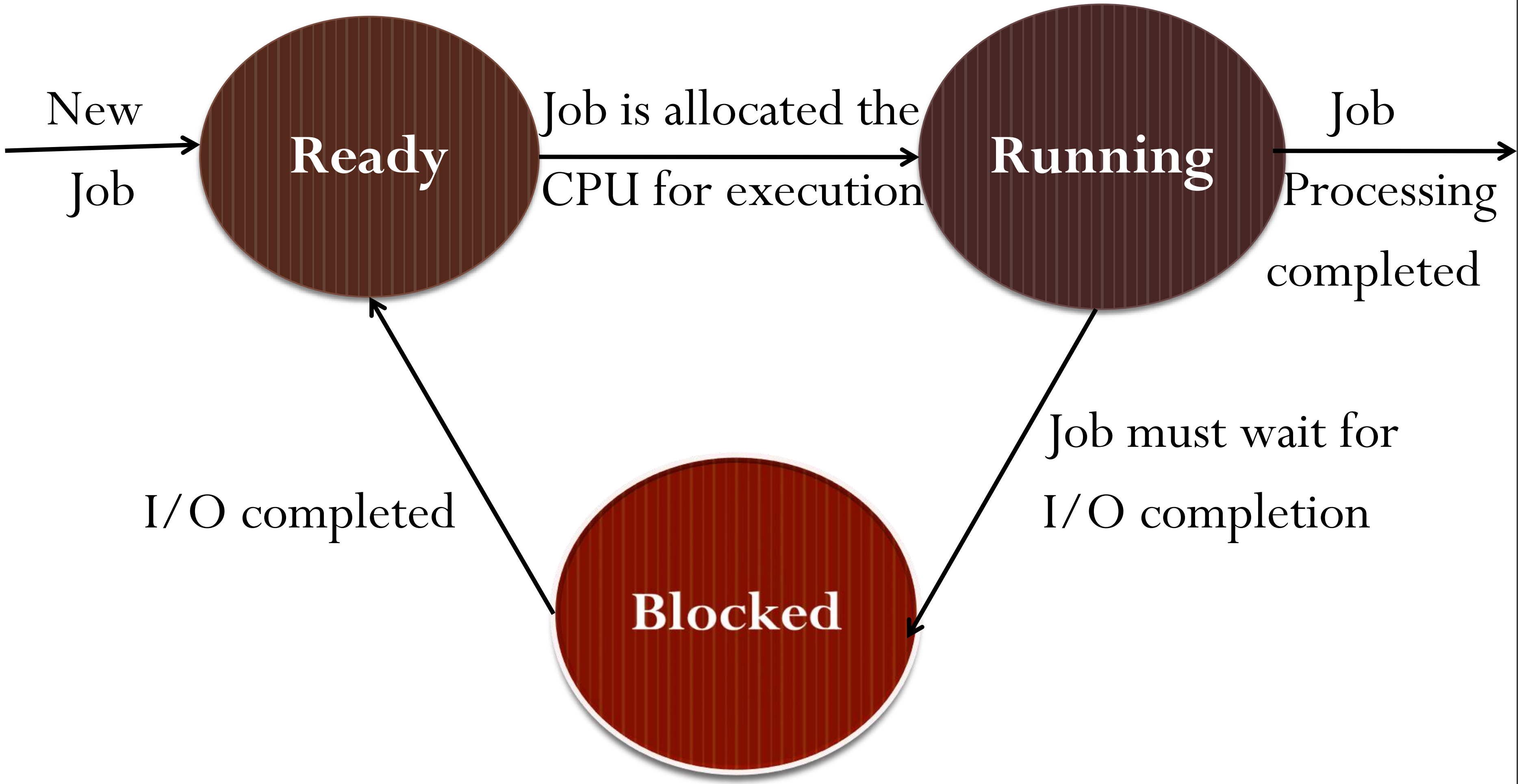
Job 4

Job 5

Memory layout for multiprogramming system



Three different states of jobs in multiprogramming



New Job

Ready

Job is allocated the CPU for execution

Running

Job Processing completed

Job must wait for I/O completion

Blocked

I/O completed

5) Multitasking System

- Technically, multitasking is same as multiprogramming.
- In a multitasking OS, a single user can execute multiple programs at the same time.
- Multitasking is the system's capability to work one or more than one job or process at the same time. It means that whenever a job needs to perform I/O operations, the CPU can be used for executing some other job or process that is also residing in the system and is ready to use the CPU.
- NOTE:

  The term **multiprogramming** is used for **multi-user systems** i.e., systems that are simultaneously used by many users such as mainframe and server class system.

  The term **multitasking** is used for **single user system** i.e., systems that are used by only one user at a time such as personal computer or a notebook.

Multitasking System (Continued...)

- Thus multitasking is the method of processing the multiple tasks concurrently in a single user system.
- For Example: A user is running separate program in 4 different windows at the same time. The program in window 1 could be printing a document, he program in window 2 could be displaying an e-mail of user, a spreadsheet program in window 3 could be preparing sales report and compilation of a program is in progress in window 4. in this manner, a user may work on many tasks at the same time. Thus, progress of different tasks can be viewed on different windows in a multitasking system.
- There are two types of multitasking:

1. Cooperative Multitasking
2. Pre-emptive Multitasking

Types of Multitasking System

1. **Cooperative Multitasking**. In Cooperative Multitasking, a program can acquire the CPU for the required amount of time. A program can share CPU with any other program that is executing simultaneously.
2. **Pre-emptive Multitasking**. In Pre-emptive Multitasking the OS allocates particular time to a program. The CPU is preempted if a higher priority job arrives in a system.

Comparison between Multiprogramming and Multitasking

Sr. No.	Multiprogramming	Multitasking
1.	It is used for multi-user systems i.e., the systems that are used by many users at the same time.	It is used for single user systems i.e., the systems that are used only by one user a same time.
2.	It refers to concurrent execution of multiple jobs that may be of same user or different users.	It refers to concurrent execution of multiple jobs of same user.

5. Multiprocessing/ Parallel/ Tightly Coupled System

- Most systems to date are single-processor systems; that is they have only one main CPU. However, **multiprocessor systems** (also known as **parallel systems** or **tightly coupled systems**) are growing in importance.
- Such systems have **more than one processor in close communication**, sharing the **computer bus**, **the clock**, and sometimes **memory** and **peripheral devices**.
- Multiprocessor systems have three main advantages:

1. **Increased throughput** : By increasing the number of processors, we hope more work done in less time. The speed-up of N processors is not N; rather it is less than N. When multiprocessors cooperate on a task, a certain amount of overhead is incurred in keeping all the parts working correctly. This overhead, plus contention for shared resources, lowers the expected gain from additional processors.
2. **Economy of Scale**: Multiprocessor systems can save more money than multiple single processor systems, because they can share peripherals, mass storage, and power supplies . If several programs operate on same set of data, it is cheaper to store those data on one disk and to have all the processors share them, than to have many computers with local disks & many copies of the data.

Multiprocessing System (Continued…)

3. **Increased reliability**. If functions can be distributed properly among several processors, then the failure of one processor will not halt the system, only slow down it. If we have ten processors and one fails, then each of the remaining nine processors must pick up a share of the work of the failed processor. Thus, the entire system runs only 10 percent slower, rather failing altogether. This ability to continue providing service proportional to the level of surviving hardware is called **graceful degradation**. Systems designed for graceful degradation are also called **fault tolerant**.

Types of multiprocessor systems

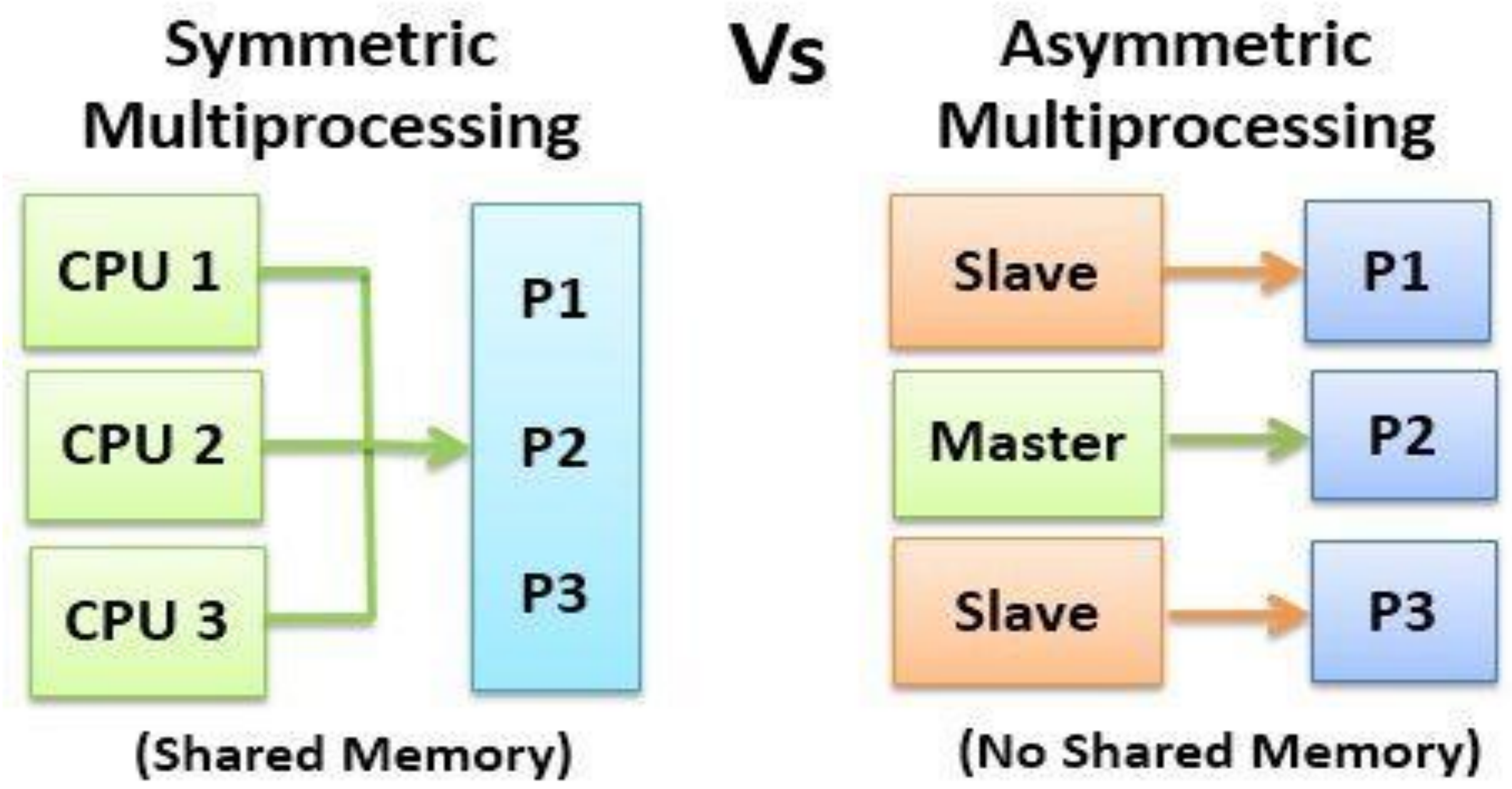
1. Symmetric Multiprocessing
2. Asymmetric Multiprocessing

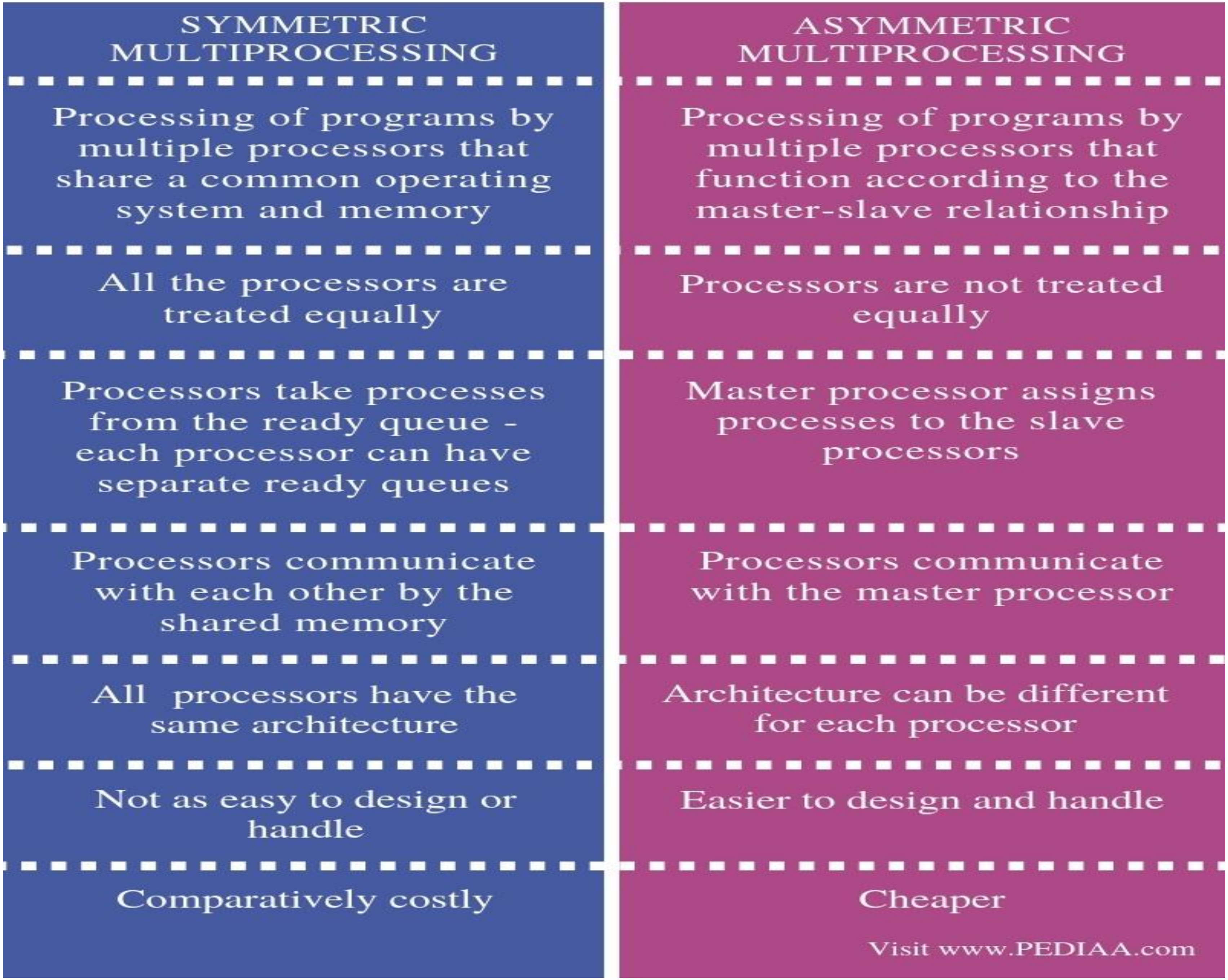
Types of multiprocessor systems

VImp Q. Differences between Symmetric and Asymmetric Multiprocessing

Sr. No.2	Symmetric Multiprocessing	2. Asymmetric Multiprocessing
1	In SMP, each processor runs an identical copy of the OS, and these copies communicate with one another as needed.	In ASMP, each processor is assigned a specific task.
2	SMP means that all processors are peers; no master-slave relationship exists between processors.	A master processor controls the system; the other processors either look to the master for instruction or have predefined tasks. This scheme defines master-slave relationship.
3	Each processor concurrently runs a copy of the operating system.	The master processor schedules and allocates work to the slave processors.

Differences between Symmetric and Asymmetric Multiprocessing

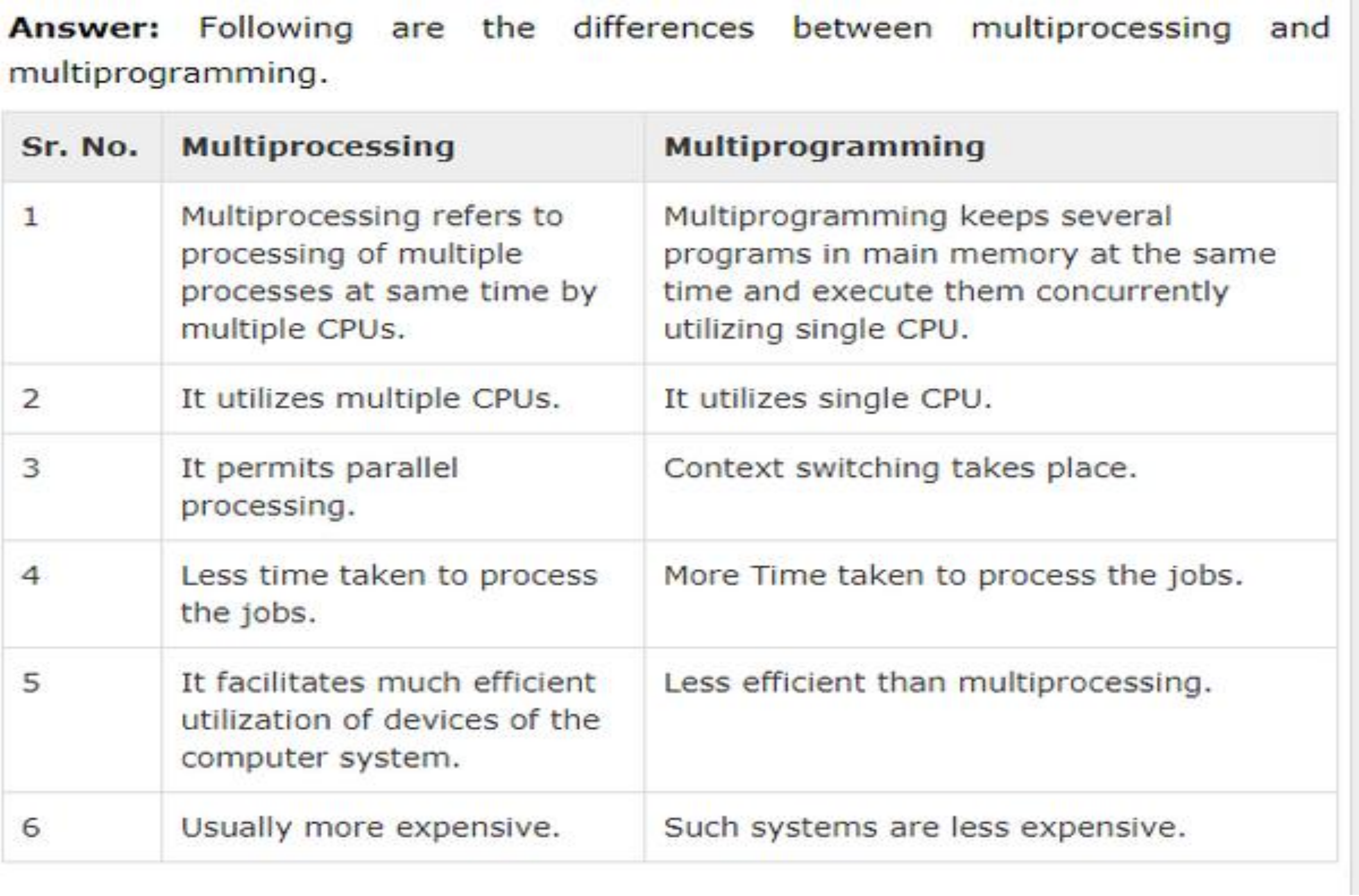




SYMMETRIC MULTIPROCESSING	ASYMMETRIC MULTIPROCESSING
Processing of programs by multiple processors that share a common operating system and memory	Processing of programs by multiple processors that function according to the master-slave relationship
All the processors are treated equally	Processors are not treated equally
Processors take processes from the ready queue - each processor can have separate ready queues	Master processor assigns processes to the slave processors
Processors communicate with each other by the shared memory	Processors communicate with the master processor
All processors have the same architecture	Architecture can be different for each processor
Not as easy to design or handle	Easier to design and handle
Comparatively costly	Cheaper

Visit www.PEDIAA.com

Comparison between Multiprogramming and Multiprocessing



Answer: Following are the differences between multiprocessing and multiprogramming.

Sr. No.	Multiprocessing	Multiprogramming
1	Multiprocessing refers to processing of multiple processes at same time by multiple CPUs.	Multiprogramming keeps several programs in main memory at the same time and execute them concurrently utilizing single CPU.
2	It utilizes multiple CPUs.	It utilizes single CPU.
3	It permits parallel processing.	Context switching takes place.
4	Less time taken to process the jobs.	More Time taken to process the jobs.
5	It facilitates much efficient utilization of devices of the computer system.	Less efficient than multiprocessing.
6	Usually more expensive.	Such systems are less expensive.

7. Real-Time Operating Systems (RTOS)

- A **real-time operating system** (RTOS) is an **operating system** (**OS**) intended to serve **real-time** applications that process data as it comes in, typically without buffer delays. Processing **time** requirements (including any **OS** delay) are measured in tenths of seconds or shorter increments of **time**.
- Real time operating systems are generally special-purpose ones designed to run embedded or specialized systems.
- These systems often operate in environments where there are significant constraints on hardware design and often narrow tasks that need to be done with great accuracy and speed.
- Real time systems are used in military, engineering, medical, and data communications systems (among others).
- Example : Typical **examples** of **real-time systems** include Air Traffic Control **Systems**, Networked Multimedia **Systems**, Command Control **Systems** etc.

Advantages:

- Better Throughput
- Time taken for response is less

Disadvantages

- Very Costly
- Large memory required

6. Distributed/ Loosely coupled Systems

- **Distributed Operating System** is one of the important type of operating system.
- Multiple central processors are used by Distributed systems to serve multiple real-time applications and multiple users. Accordingly, Data processing jobs are distributed among the processors.
- Processors communicate with each other through various communication lines (like high-speed buses or telephone lines). These are known as **loosely coupled systems** or distributed systems. Processors in this system may vary in size and function. They are referred as sites, nodes, computers, and so on.

Advantages

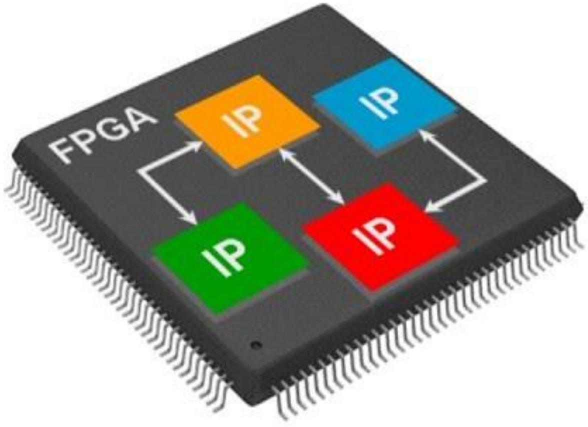
- With resource sharing facility, a user at one site may be able to use the resources available at another.
- Speedup the exchange of data with one another via electronic mail.
- Failure of one site in a distributed system doesn’t affect the others, the remaining sites can potentially continue operating.
- Better service to the customers.
- Reduction of the load on the host computer.
- Reduction of delays in data processing.

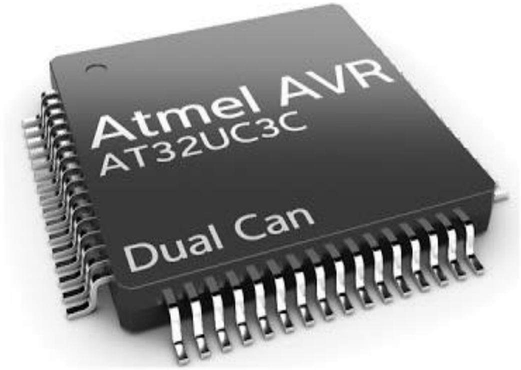
EMBEDDED OPERATING SYSTEM
A Real-Time Examples
USED IN REGULAR LIFE
EES
ELYSIUM EMBEDDED SCHOOL

- **Embedded System:**

  "Embedded" means hidden inside so one can't see it.

  "System" means multiple components interfaced Together for a common purpose.

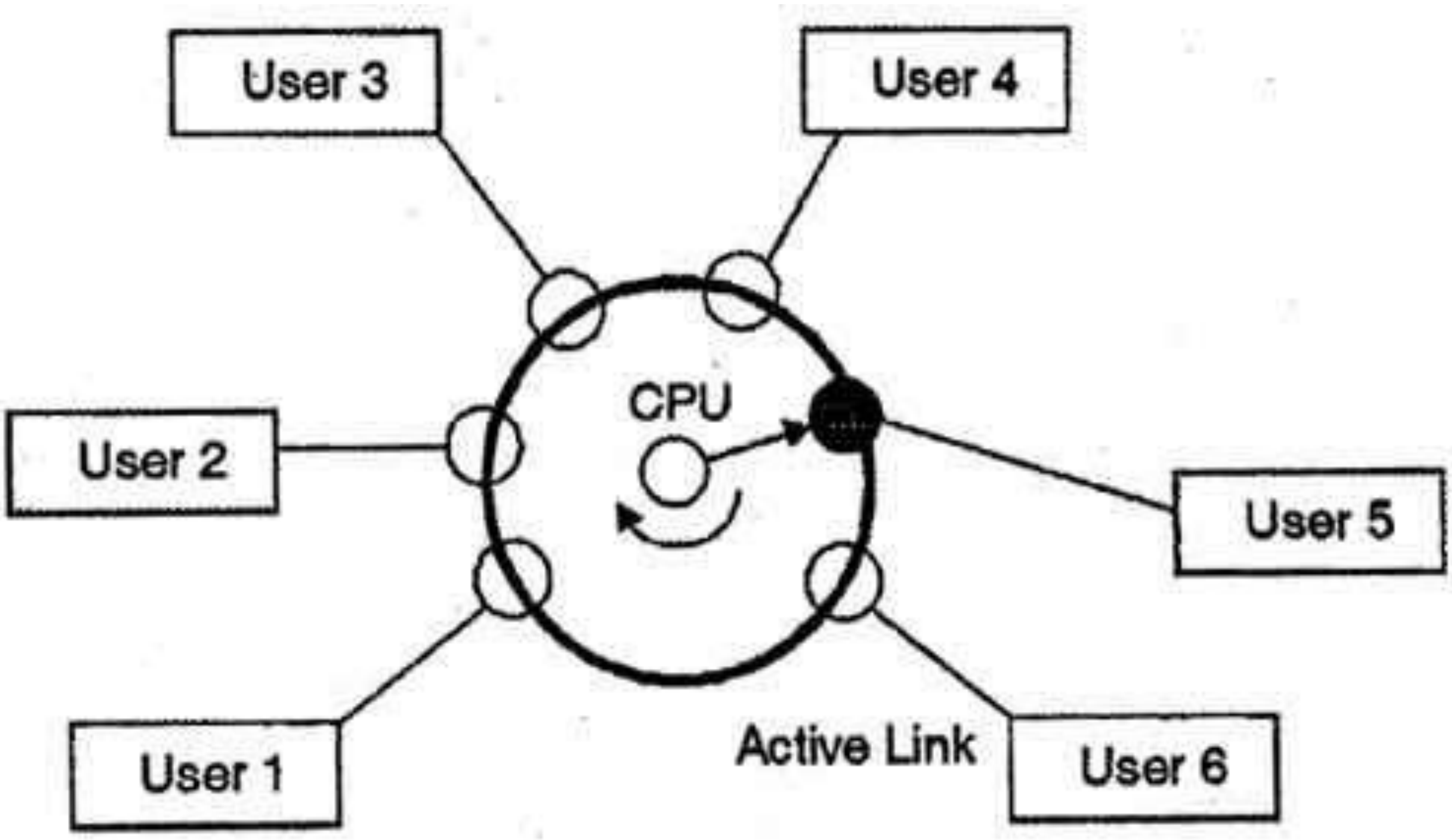




8. Time Sharing Operating System

- A **time sharing system** allows many users to share the computer resources simultaneously. In other words, time sharing refers to the allocation of computer resources in time slots to several programs simultaneously. For example a mainframe computer that has many users logged on to it.
- Each user uses the resources of the mainframe -i.e. memory, CPU etc. The users feel that they are exclusive user of the CPU, even though this is not possible with one CPU i.e. shared among different users.
- The time sharing systems were developed to provide an interactive use of the computer system. A time shared system uses CPU scheduling and multi-programming to provide each user with a small portion of a time-shared computer. It allows many users to share the computer resources simultaneously. As the system switches rapidly from one user to the other, a short time slot is given to each user for their executions.
- The time sharing system provides the direct access to a large number of users where CPU time is divided among all the users on scheduled basis. The OS allocates a set of time to each user. When this time is expired, it passes control to the next user on the system. The time allowed is extremely small and the users are given the impression that they each have their own CPU and they are the sole owner of the CPU. This short period of time during that a user gets attention of the CPU; is known as a *time slice or a quantum*. The concept of time sharing system is shown in figure.

- The time-shared systems are more complex than the multi-programming systems. In time-shared systems multiple processes are managed simultaneously which requires an adequate management of main memory so that the processes can be swapped in or swapped out within a short time.



Handheld Systems

- Handheld systems include Personal Digital Assistants (PDA's), such as Palm OS, cellular telephones with connectivity to a network such as the internet.
- Developers of handheld systems and applications face many challenges most of which are due to the limited size of such devices.
- Ex: PDA
- First issue faced by programmer: Due to this limited size, most handheld devices have a small amount of memory, include slow processors, and feature small display screens.
- Many handheld devices have been limited size of memory.
- As a result, the OS and applications must manage memory efficiently. This includes returning all allocated memory back to the memory manager once the memory is no longer being used.
- Smartphones combine a **mobile phone** and a **handheld** computer into a single **device**. **Examples** of smartphones over the years have included the Apple **iPhone**, Samsung Galaxy, Microsoft and Nokia Lumia, Sony Ericsson etc.,
- Second issue faced by programmer: Speed of processor
- Processor for most handheld devices often run at a fraction of the speed of a processor in a PC.
- Third issue faced by programmer: Displaying the content in web pages in we clipping where only a small subset of web page is delivered and displayed on a handheld device.