

Parallel Processing Paradigm

By
Smita Mande

Introduction Parallel Processing

➤ Serial Processing:

To solve a problem, an algorithm divides the problem into smaller instructions. These discrete instructions are then executed on the Central Processing Unit of a computer one by one. Only after one instruction is finished, next one starts.

Keypoints:

1. In this, a problem statement is broken into discrete instructions.
2. Then the instructions are executed one by one.
3. Only one instruction is executed at any moment of time.

Introduction Parallel Processing

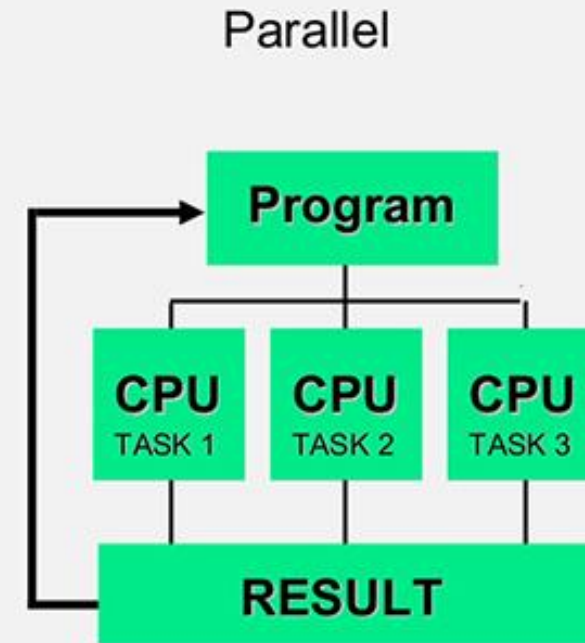
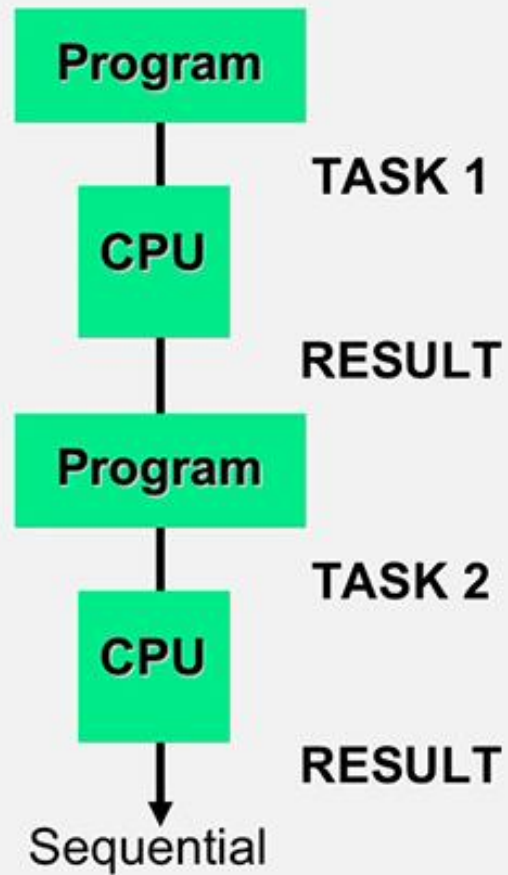
➡ Parallel Processing

It is the use of multiple processing elements simultaneously for solving any problem. Problems are broken down into instructions and are solved concurrently as each resource that has been applied to work is working at the same time.

Advantages of Parallel Computing over Serial Computing are as follows:

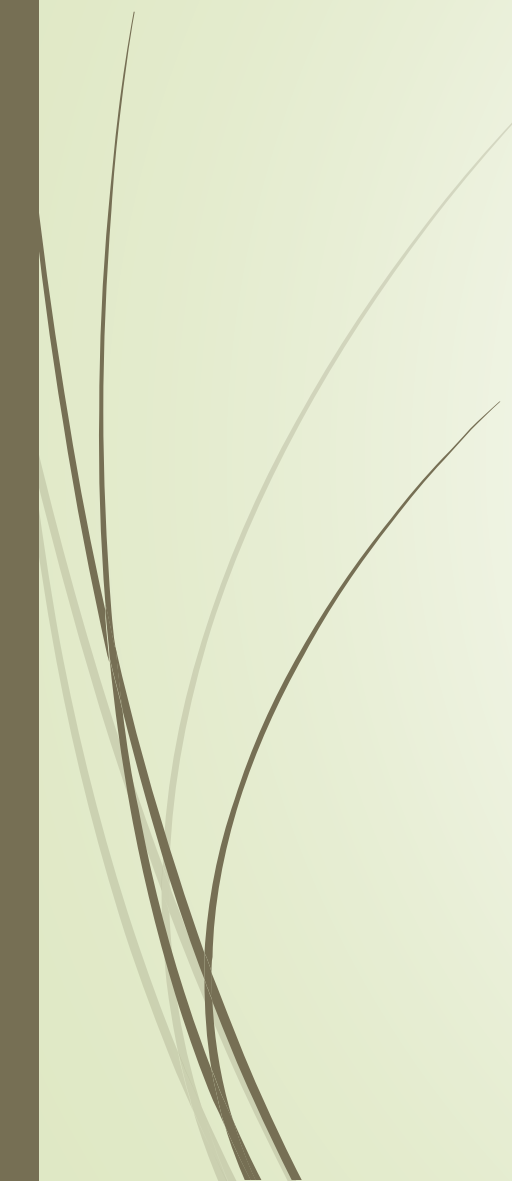
1. It saves time and money as many resources working together will reduce the time and cut potential costs.
2. It can be impractical to solve larger problems on Serial Computing.
3. It can take advantage of non-local resources when the local resources are finite.
4. Serial Computing 'wastes' the potential computing power, thus Parallel Computing makes better work of the hardware.

Sequential and parallel processing





Types of Parallelism

- Bit-level parallelism
 - Instruction-level parallelism
 - Task Parallelism
 - Data-level parallelism (DLP)
- 



► Bit-level parallelism

It is the form of parallel computing which is based on the increasing processor's size. It reduces the number of instructions that the system must execute in order to perform a task on large-sized data.

- *Example:* Consider a scenario where an 8-bit processor must compute the sum of two 16-bit integers. It must first sum up the 8 lower-order bits, then add the 8 higher-order bits, thus requiring two instructions to perform the operation. A 16-bit processor can perform the operation with just one instruction.

► Instruction-level parallelism

A processor can only address less than one instruction for each clock cycle phase. These instructions can be re-ordered and grouped which are later on executed concurrently without affecting the result of the program. This is called instruction-level parallelism.



■ **Task Parallelism –**

Task parallelism employs the decomposition of a task into subtasks and then allocating each of the subtasks for execution. The processors perform the execution of sub-tasks concurrently.

■ **Data-level parallelism (DLP) –**

Instructions from a single stream operate concurrently on several data

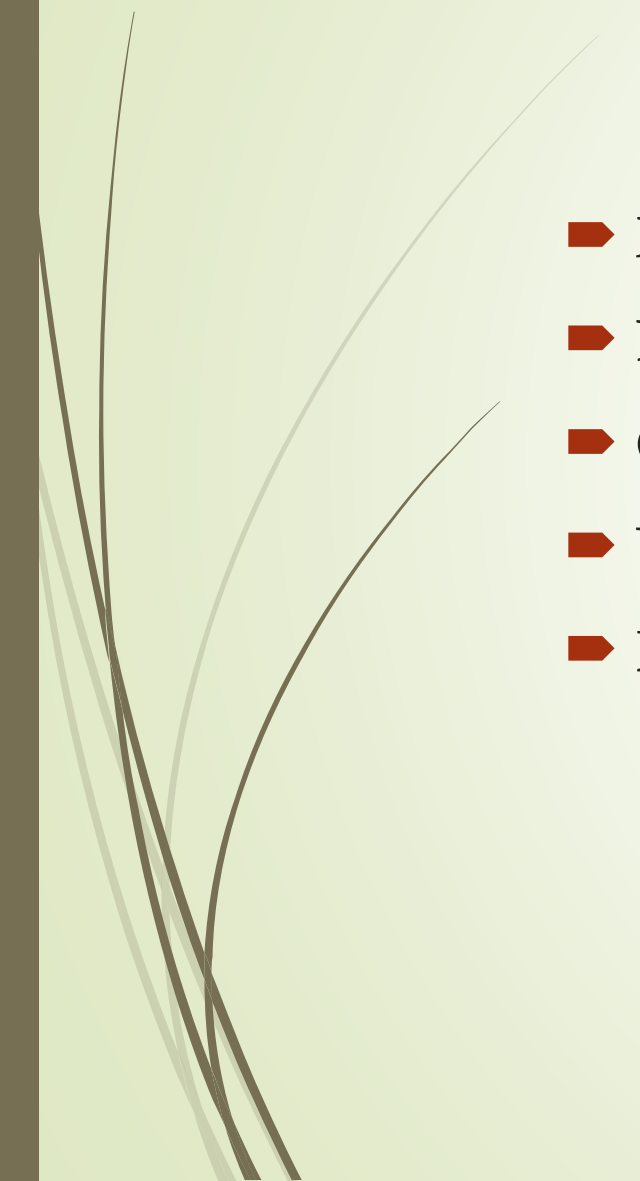


Parallelism in Uni processor system

- A uniprocessor is a system with a single processor which has three major components that are main memory i.e. the central storage unit, the central processing unit i.e. CPU, and an input-output unit like monitor, keyboard, mouse, etc.
- Parallelism in a uniprocessor means a system with a single processor performing two or more than two tasks simultaneously. Parallelism can be achieved by two means hardware and software.
- Parallelism increases efficiency and reduces the time of processing.



Hardware Approach for Parallelism in Uniprocessor

- Multiplicity of Functional Unit
 - Parallelism and Pipelining within CPU
 - Overlapped CPU and I/O Operation
 - Use Hierarchical Memory System
 - Balancing of Subsystem Bandwidth
- 



1. Multiplicity of Functional Unit

In earlier computers, the CPU consists of only one arithmetic logic unit which used to perform only one function at a time.

This slows down the execution of the long sequence of arithmetic instructions.

To overcome this the functional units of the CPU can be increased to perform parallel and simultaneous arithmetic operations.

2. Parallelism and Pipelining within CPU

Parallel adders can be implemented using techniques such as carry-lookahead and carry-save.

The multiplier can be recoded to eliminate more complex calculations.

3. Overlapped CPU and I/O Operation

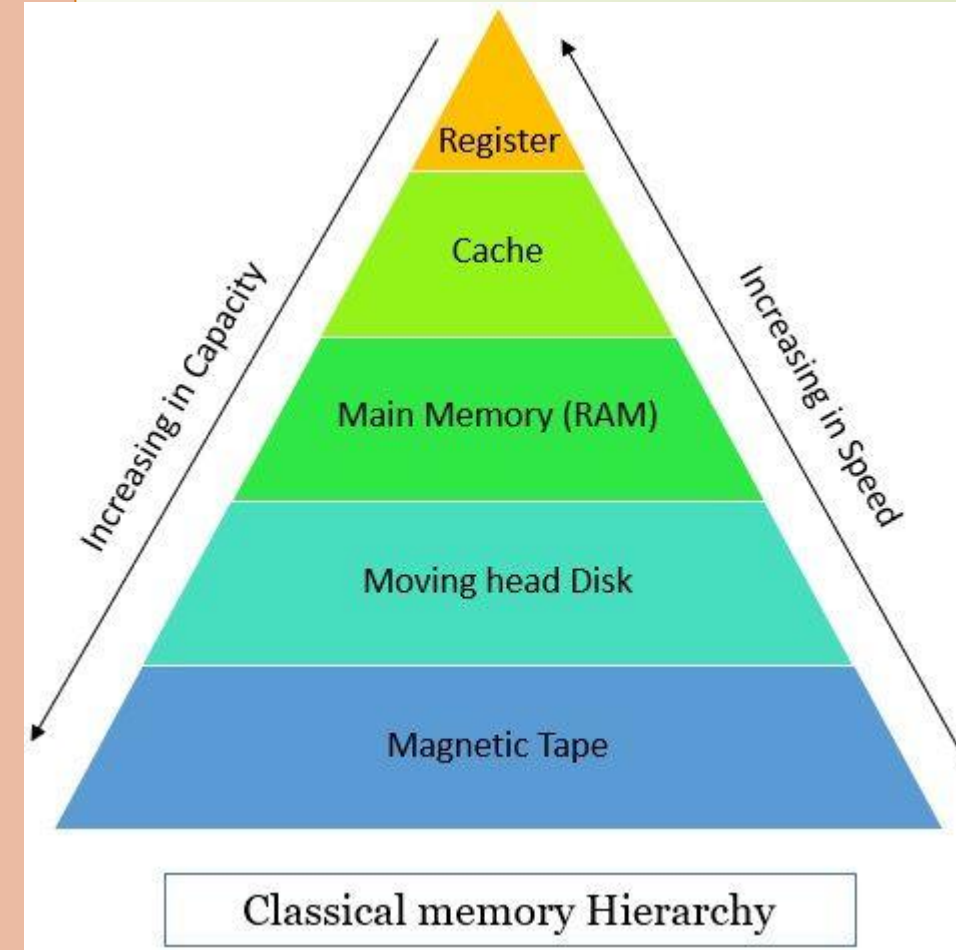
To execute I/O operation parallel to the CPU operation we can use I/O controllers or I/O processors.

For direct information transfer between the I/O device and the main memory, direct memory access (DMA) can be used.

4. Use Hierarchical Memory System

We all are aware of the fact that the processing speed of the CPU is 1000 times faster than the memory accessing speed which results in slowing the processing speed.

To overcome this speed gap hierarchical memory system can be used



5. Balancing of Subsystem Bandwidth

The processing and accessing time of CPU, main memory, and I/O devices are different.

$$t_d > t_m > t_p$$

where t_d is the processing time of the device,
 t_m is the processing time of the main memory,
and t_p is the processing time of the central processing unit.

To put a balance between the speed of CPU and memory a fast cache memory can be used which buffers the information between memory and CPU.

To balance the bandwidth between memory and I/O devices, input-output channels with different speeds can be used between main memory and I/O devices.

Software Approach for Parallelism in Uniprocessor

➤ Multiprogramming

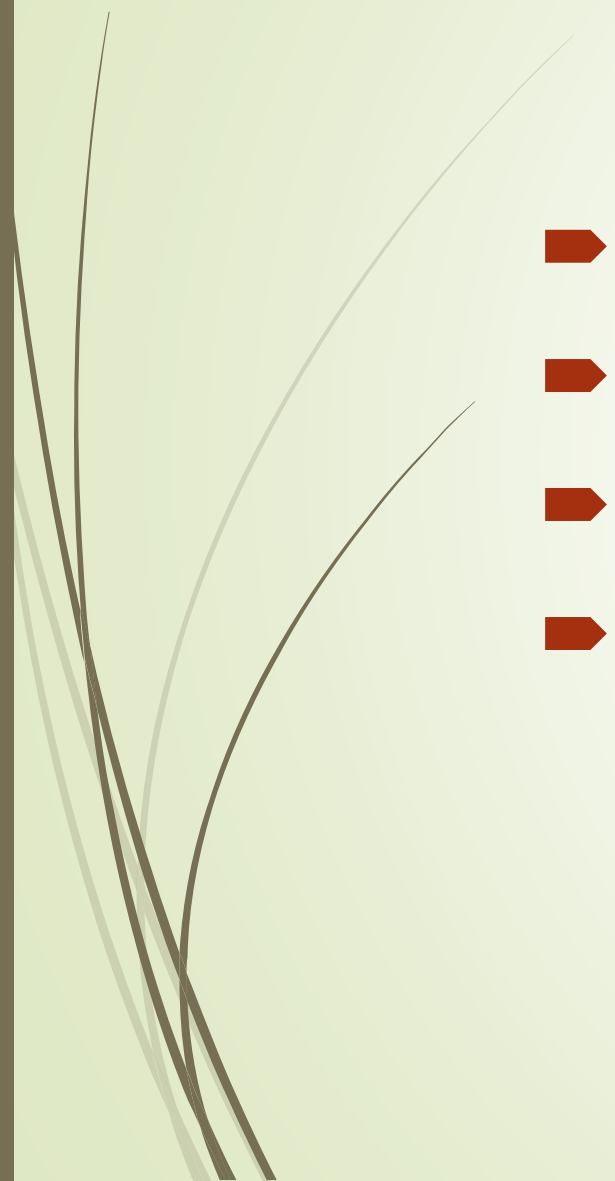
Program interleaving can be understood as when a process P_1 is engaged in I/O operation the process scheduler can switch CPU to operate process P_2 . This led process P_1 and P_2 to execute simultaneously. This interleaving between CPU and I/O devices is called multiprogramming.

➤ Time-Sharing

Multiprogramming is based on the concept of time-sharing. The CPU time is shared among multiple programs. Sometimes a high-priority program can engage the CPU for a long period starving the other processes in the computer.

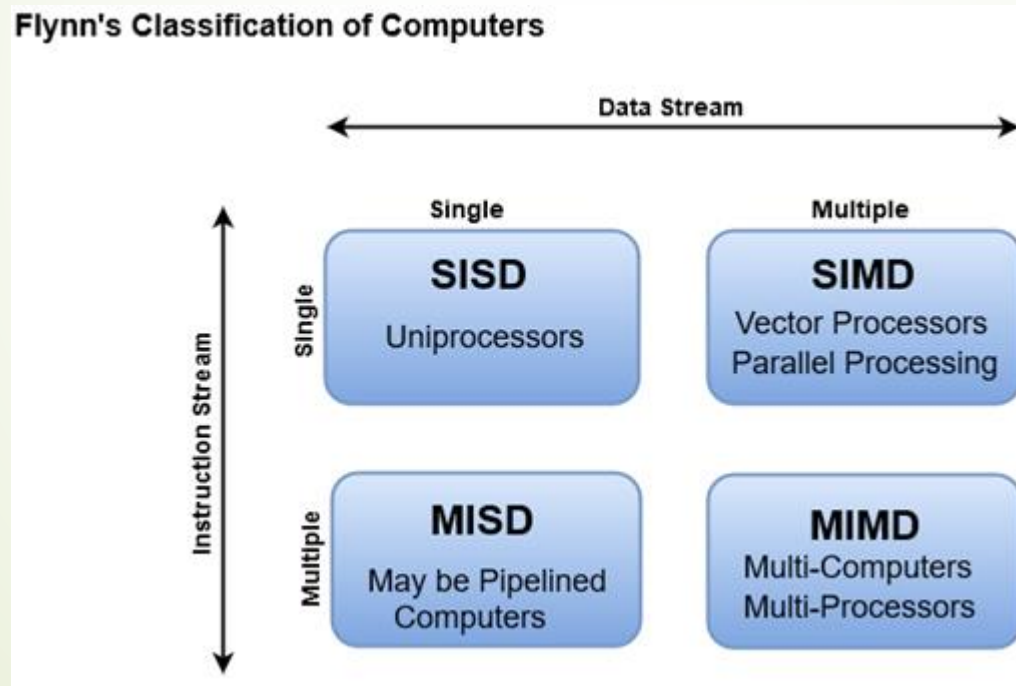


Classification based on Architectural schemes

- Flynn's classification
 - Shores classification
 - Feng's classification
 - Handle's classification
- 

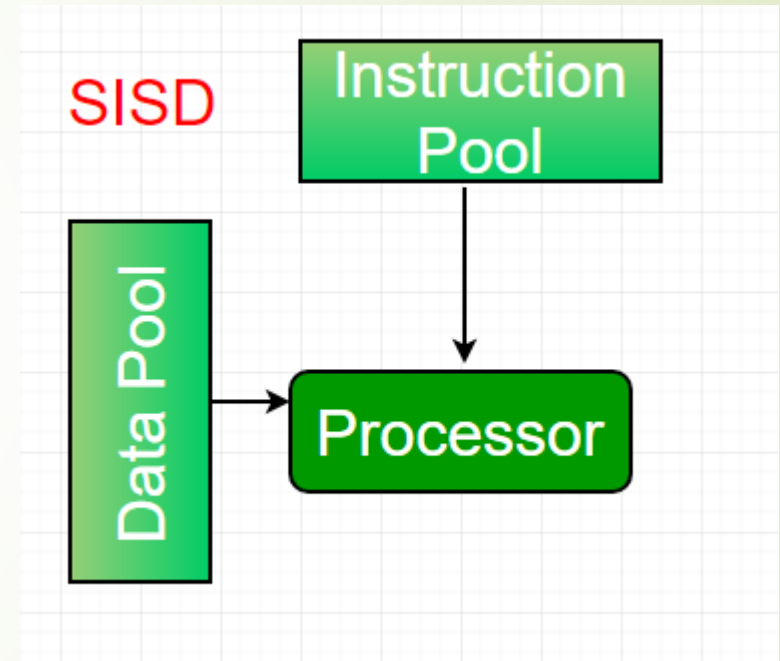
Flynn's classification

- According to **Flynn's** there are four different classification of computer architecture,



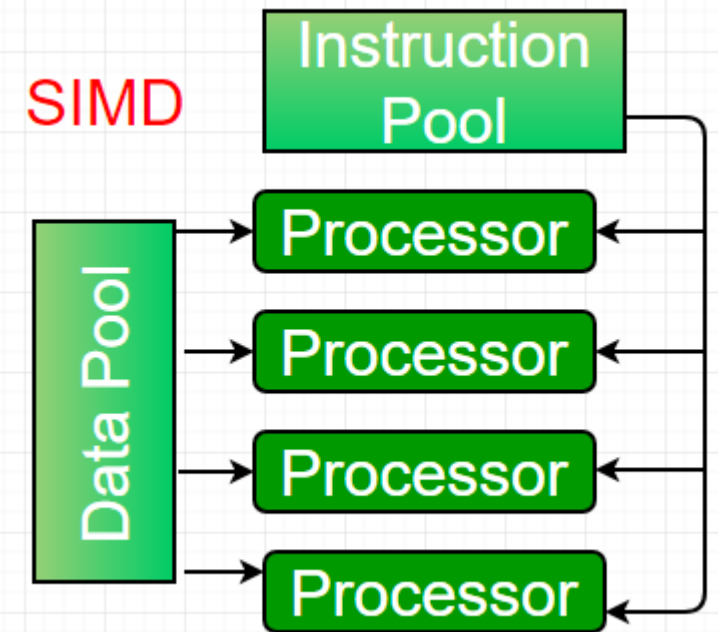
Single-instruction, single-data (SISD) systems

- An SISD computing system is a uniprocessor machine which is capable of executing a single instruction, operating on a single data stream.
- In SISD, machine instructions are processed in a sequential manner and computers adopting this model are popularly called sequential computers.
- Most conventional computers have SISD architecture.
- All the instructions and data to be processed have to be stored in primary memory.



Single-instruction, multiple-data (SIMD) systems

- An SIMD system is a multiprocessor machine capable of executing the same instruction on all the CPUs but operating on different data streams.
- Machines based on an SIMD model are well suited to scientific computing since they involve lots of vector and matrix operations. So that the information can be passed to all the processing elements (PEs) organized data elements of vectors can be divided into multiple sets (N-sets for N PE systems) and each PE can process one data set.

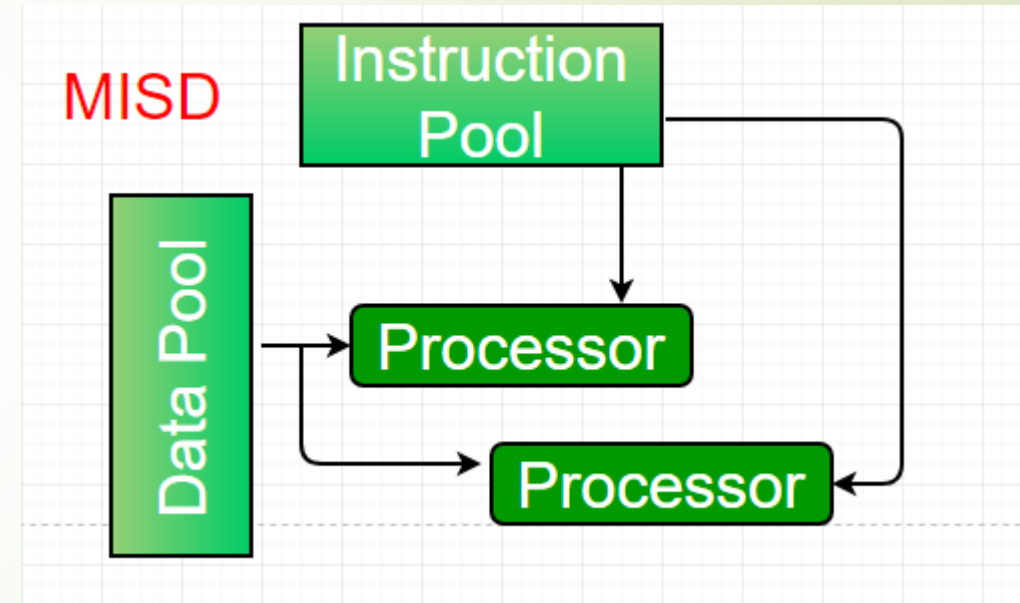


Multiple-instruction, single-data (MISD) systems

- An MISD computing system is a multiprocessor machine capable of executing different instructions on different PEs but all of them operating on the same dataset .

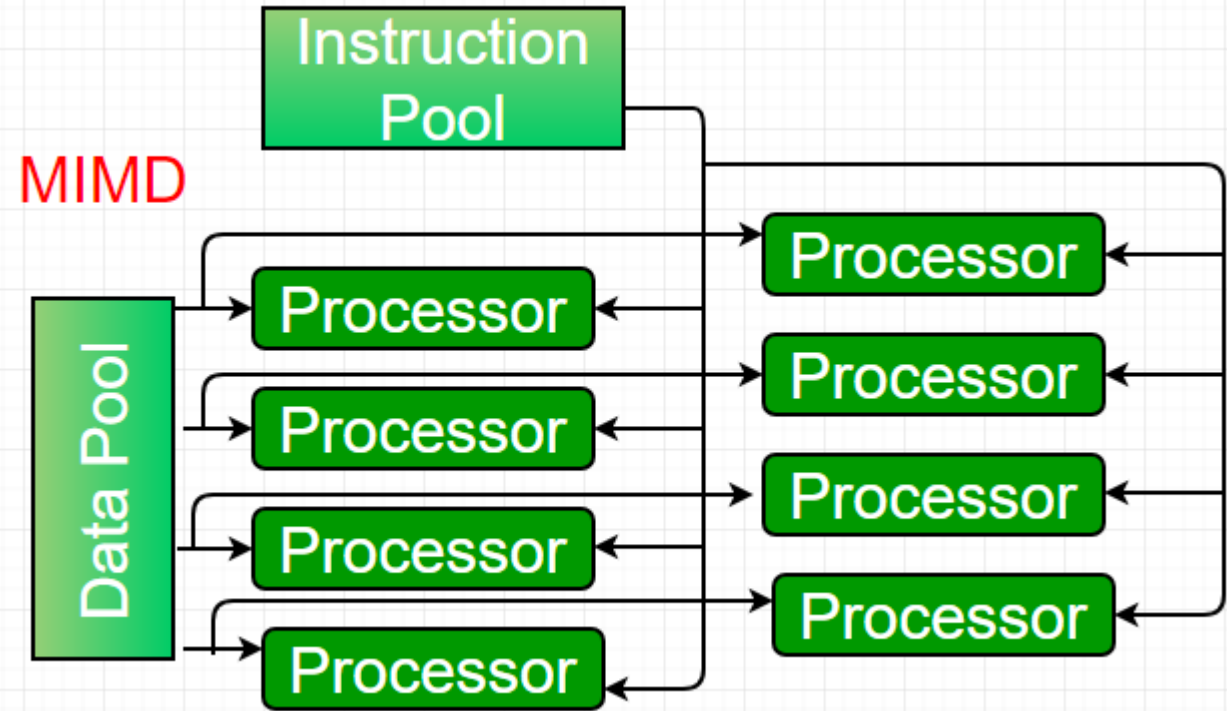
Example $Z = \sin(x) + \cos(x) + \tan(x)$

- The system performs different operations on the same data set.



Multiple-instruction, multiple-data (MIMD) systems

- An MIMD system is a multiprocessor machine which is capable of executing multiple instructions on multiple data sets.
- Each PE in the MIMD model has separate instruction and data streams; therefore machines built using this model are capable to any kind of application.
- Unlike SIMD and MISD machines, PEs in MIMD machines work asynchronously.







MIMD machines are broadly categorized into

1. Shared-memory MIMD

2. Distributed-memory MIMD

- In the **Shared memory MIMD** model (tightly coupled multiprocessor systems), all the PEs are connected to a single global memory and they all have access to it. The communication between PEs in this model takes place through the shared memory, modification of the data stored in the global memory by one PE is visible to all other PEs.
- In **Distributed memory MIMD** machines (loosely coupled multiprocessor systems) all PEs have a local memory. The communication between PEs in this model takes place through the interconnection network (the inter process communication channel, or IPC).


- 
- 
- The shared-memory MIMD architecture is easier to program but is less tolerant to failures and harder to extend with respect to the distributed memory MIMD model.
 - Failures in a shared-memory MIMD affect the entire system, whereas this is not the case of the distributed model, in which each of the PEs can be easily isolated.
 - Moreover, shared memory MIMD architectures are less likely to scale because the addition of more PEs leads to memory contention. This is a situation that does not happen in the case of distributed memory, in which each PE has its own memory.
 - As a result of practical outcomes and user's requirement , distributed memory MIMD architecture is superior than other existing models.



Need and basics of Multicore architecture

A multi-core CPU's objective is to obtain great performance. It was designed to overcome physical limitations of a single-core CPU.

Some applications of the multicore processor are as follows:

- Games with high graphics, such as Overwatch and Star Wars Battlefront, as well as 3D games.
 - The multicore processor is more appropriate in Adobe Premiere, Adobe Photoshop, iMovie, and other video editing software.
 - Solid works with computer-aided design (CAD).
 - High network traffic and database servers.
 - Industrial robots, example, are embedded systems.
- 

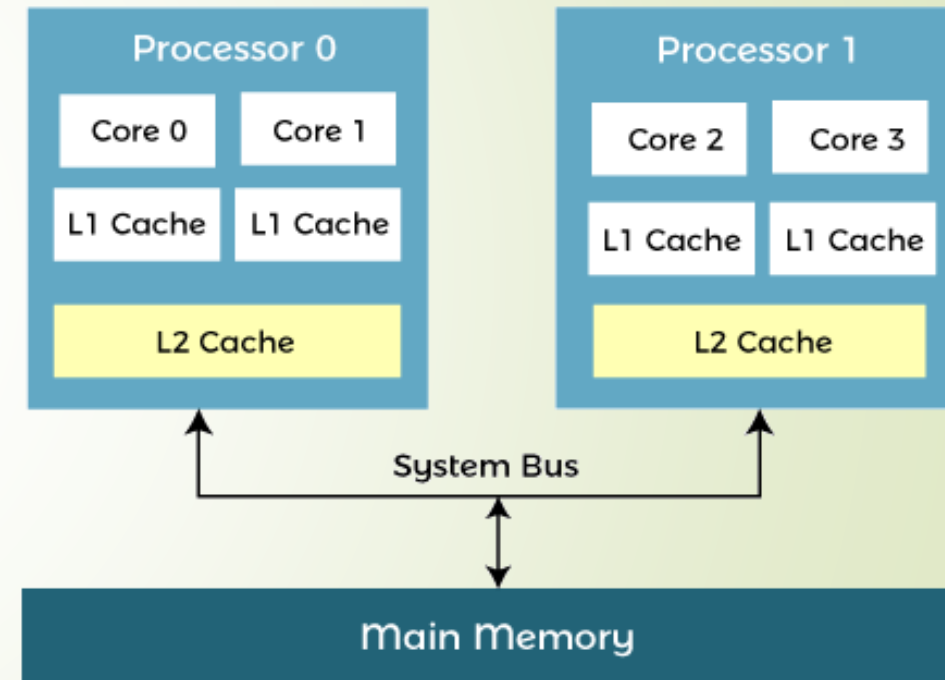


Multicore Processors

- A multi-core processor is an integrated circuit with two or more processors connected to it for faster simultaneous processing of several tasks, reduced power consumption, and for greater performance
- on a single chip, a multi-core processor comprises numerous processing units, or "Cores," each of which has the potential to do distinct tasks. For instance, if you are performing many tasks at once, such as watching a movie and using WhatsApp, one core will handle activities like watching a movie while the other handles other responsibilities like WhatsApp.
- A dual-core configuration is comparable to having several different processors installed on the same computer, but the connection between them is faster because the two CPUs are plugged into the same socket. Several instructions in parallel may be executed by individual cores, boosting the speed of software built to make use of the architecture's unique features.

Architecture of Multicore Processors

- A multi-core processor's design enables the communication between all available cores, and they divide and assign all processing duties appropriately.
- The processed data from each core is transmitted back to the computer's main board (Motherboard) via a single common gateway once all of the processing operations have been finished.
- This method beats a single-core CPU in terms of total performance.



Advantages of Multi-Core Processor

- **Performance**

A multi-core CPU, by nature, can do more work as compared to a single-core processor.

- **Reliability**

In multi-core CPUs, the software is always assigned to different cores. When one piece of software fails, the others remain unaffected. Whenever a defect arises, it affects only one core. As a result, multi-core CPUs are better able to resist faults.

- **Software Interactions**

Even if the software is running on multiple cores, it will communicate with one another.

- **Multitasking**

An operating system can use a multi-core CPU to run two or more processes at the same time, even if many programs may be executed at the same time.

- **Power Consumption**

Multitasking with a multi-core CPU, on the other hand, requires less power. Only the part of the CPU that generates heat will be used. The power consumption is eventually minimized, resulting in less battery utilization.

❑ Some other benefits of Multicore Processor:

- When compared to single-core processors, a multicore processor has the potential of doing more tasks.
- Low energy consumption when doing many activities at once.
- Data takes less time to reach its destination since both cores are integrated on a single chip.
- Detecting infections with anti-virus software while playing a game is an example of multitasking.

Disadvantages of Multi-Core Processors

- **Resource Sharing**

A multi-core processor shares a variety of resources, both internal and external. Networks, system buses, and main memory are among these resources. Consequently, any program running on the same core will have a higher chance of being interrupted.

- **Software Interference**

Due to resource sharing, software interference can cause problems. The presence of more cores implies a greater number of interference routes.

❑ **Some other key points of limitations of Multicore Processor:**

- Although it contains several processors, it is not twice as fast as a simple processor.
- The task of managing is more complicated as compared to managing a single-core CPU.
- The performance of a multi-core processor is entirely dependent on the tasks that users execute.
- If other processors demand linear/sequential processing, multi-core processors take longer to process.
- In comparison to a single-core processor, it is more expensive.