Classification of Processor Architectures

Agenda

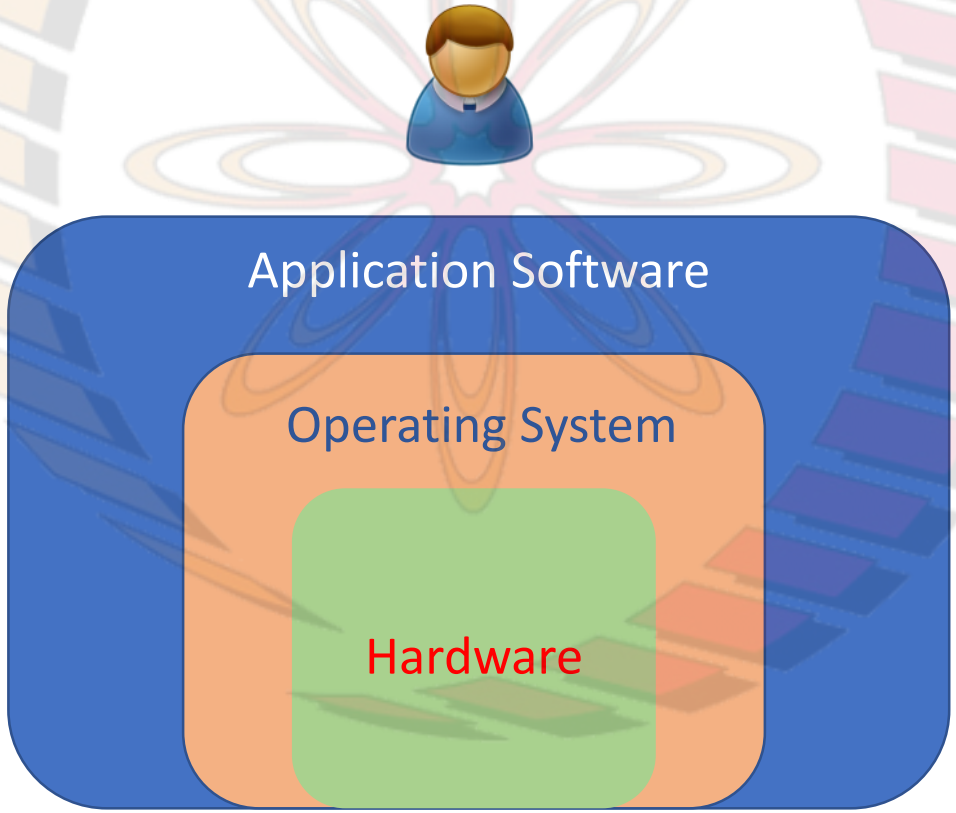
- Basics of Computing
- Flynn’s Taxonomy
- Basics of Process
- Parallelization Methodology
- Summary

Basics of Computing

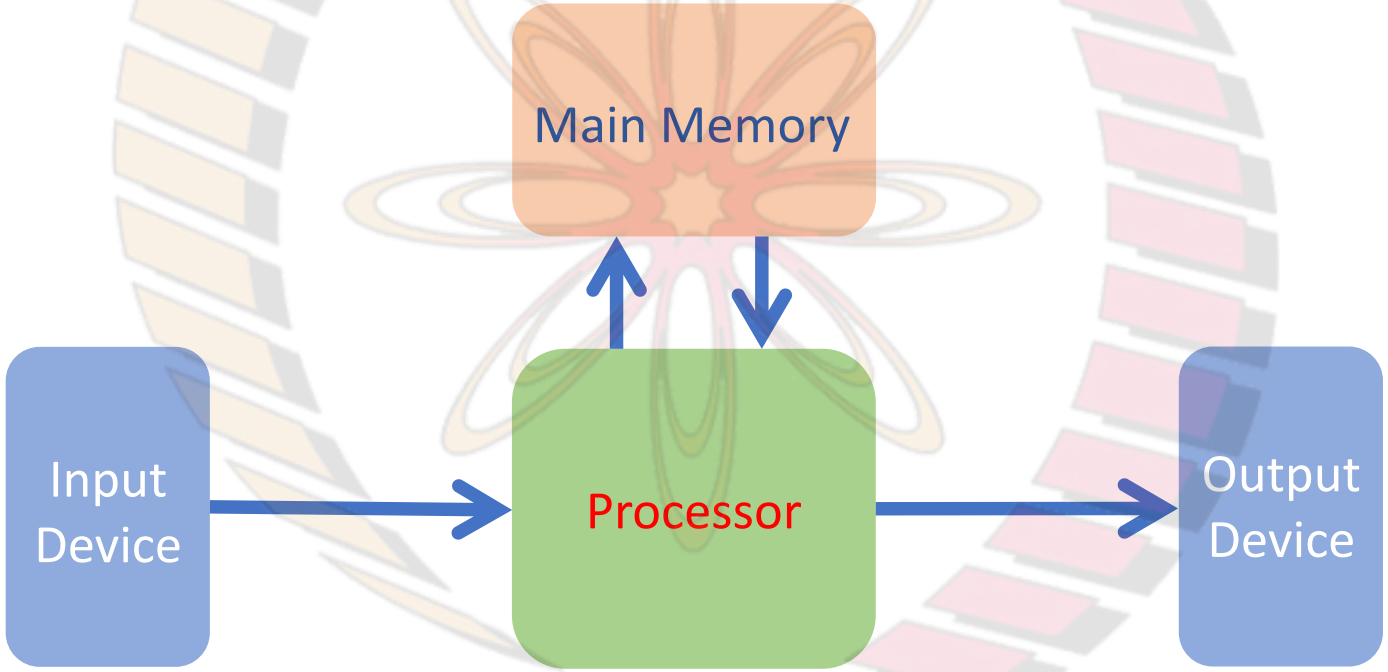
Types of Computers

Type	Used by	Access	Purpose	Use case
Desktop Computer	Single person	Keyboard-Display	Routine use	Home or office use
Embedded Computer	Appliance user	Customised	Control an appliance/device	In Car, Home appliance, found everywhere
Tablet	Single person	Touch Screen	Personal use	Entertainment
Server	Multiple users	Over network	Bigger workloads	File sharing, Banking, hosting website
Supercomputer	Multiple users	Job submission	Scientific applications	Solving grand challenge problems

Simplified Model of Computer



Simplified View of the hardware

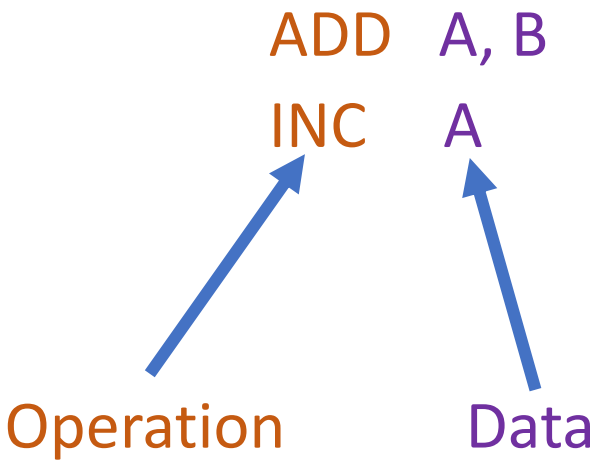


Basic Computing Model

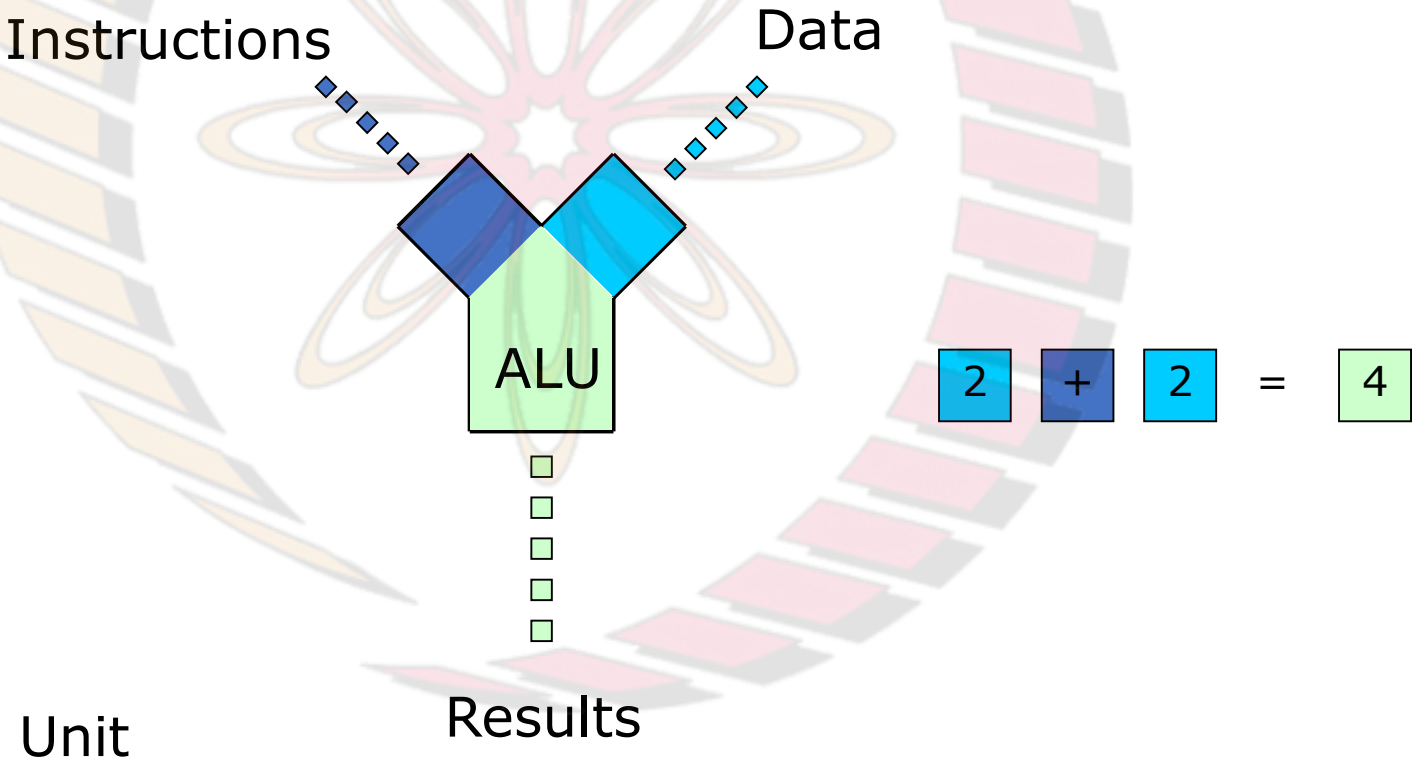
- Programmers use High-level Programming Language

a = a + b +1

- However, CPU understands Machine Language (or Assembly Language)
- High level language statement is converted to assembly language by system software tools
- Assembly language program comprises of a series of Instructions
- Each Instruction typically has an opcode and an operand
    - Opcode specifies the operation to be performed
    - Operand specifies the location of data

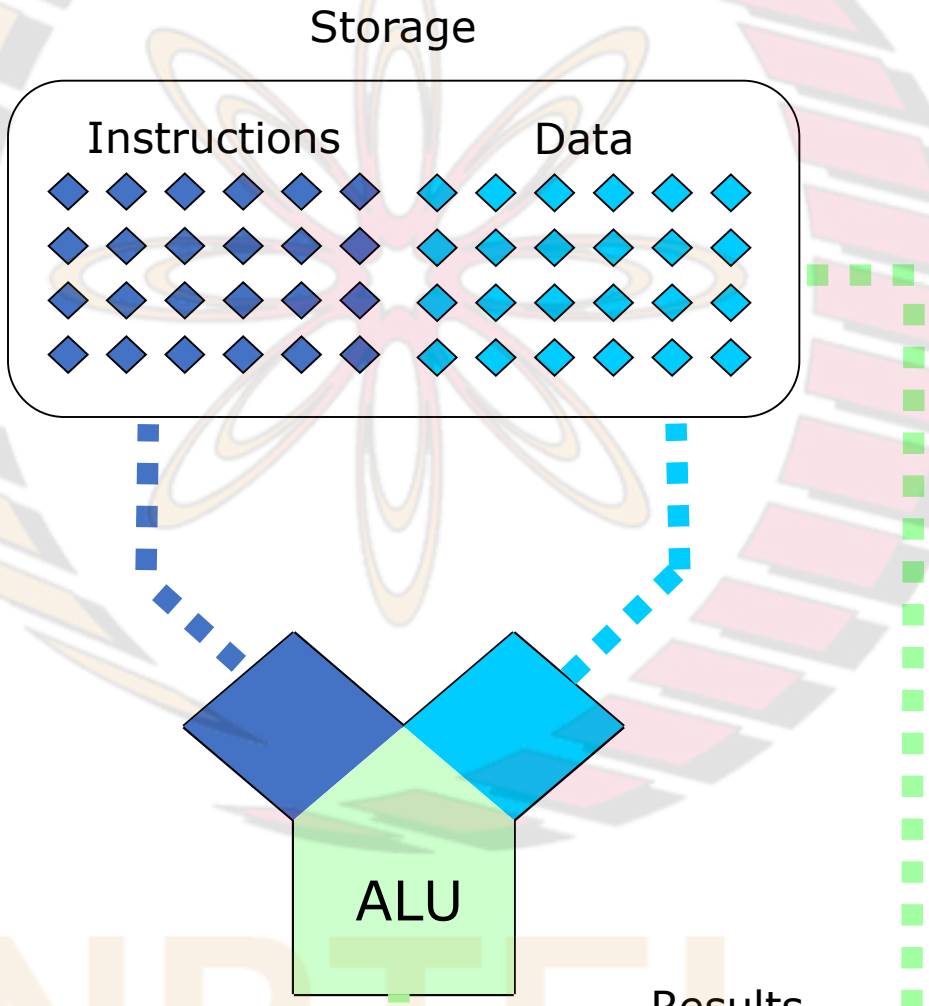


The heart of the processor - Simplified



ALU : Arithmetic Logic Unit

Instruction and Data

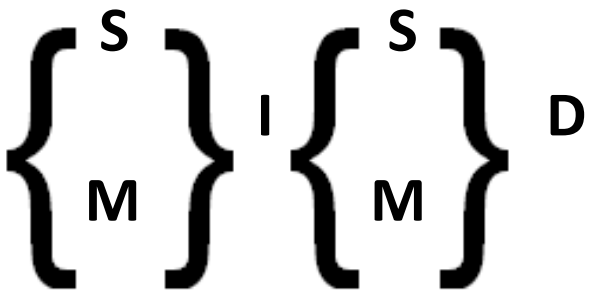


ALU : Arithmetic Logic Unit

Flynn’s Taxonomy

Flynn's Taxonomy

- In 1966, Michael J. Flynn proposed classification of computer architectures



Source: Wikipedia

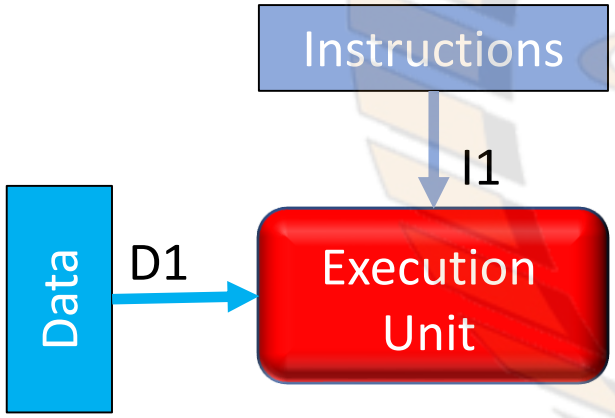
- SI: Single Instruction – All processors execute same Instruction

MI: Multiple Instruction – Different processors may be executing different instructions

- SD: Single Data – All processors are operating on the same Data

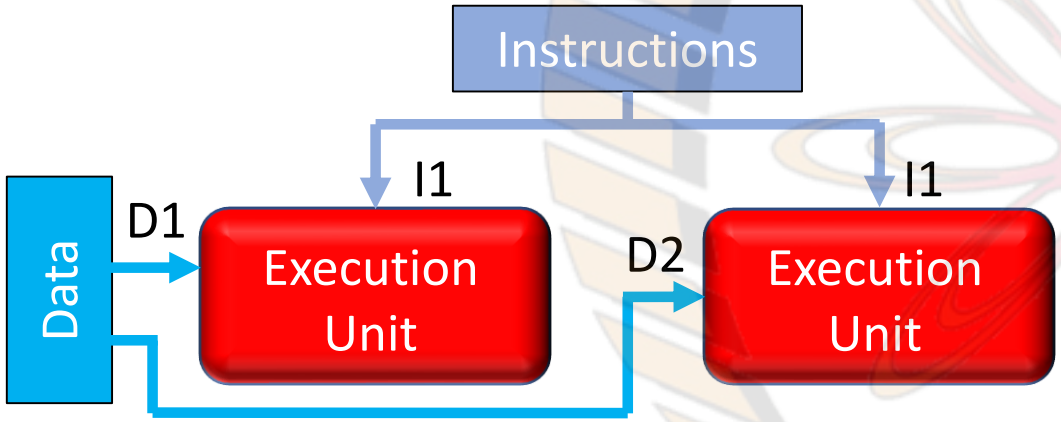
- MD: Multiple Data- Different processors operating on different data

Single Instruction Single Data



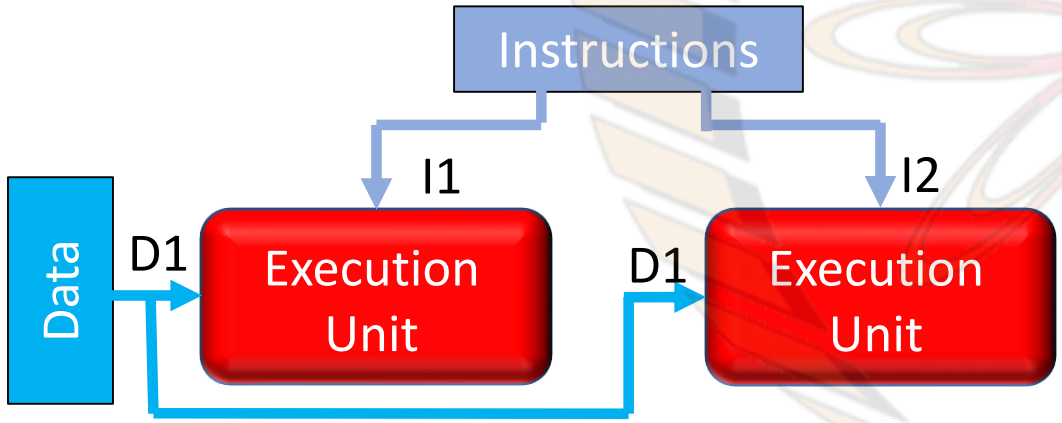
- A sequential computer which does not incorporate any parallelism
- Standard single CPU serial computer and program
- Building block for parallel computing systems

Single Instruction Multiple Data



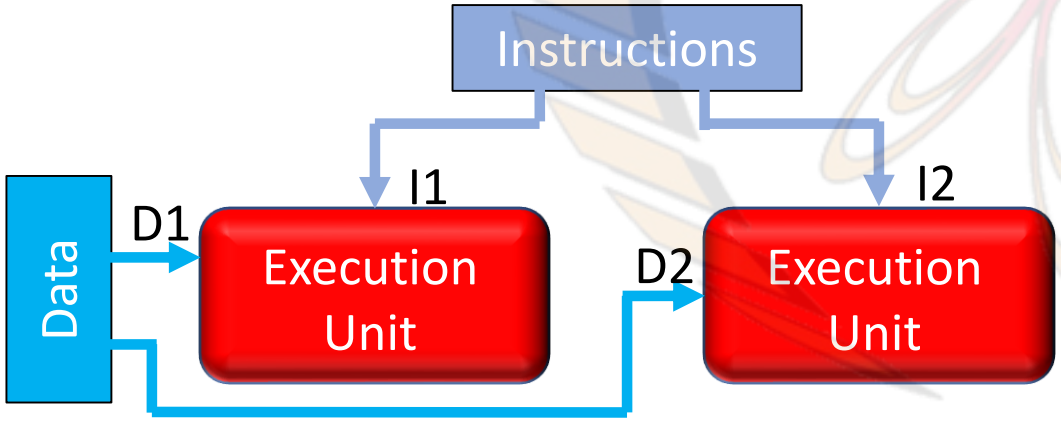
- A computer which operates on multiple data streams using a single instruction stream
- The first use of SIMD was in vector supercomputers like Cray platforms
- SIMD instructions are executed by ALU in modern processors
- Useful for applications that handle streaming data. For example Audio or Video
- GPUs use this concept

Multiple Instruction Single Data



- Multiple instructions operate on a single data stream.
- Rarely implemented architecture
- Possible uses
  - for fault tolerance
  - Multiple frequency filters
  - Cracking messages coded using cryptographic techniques

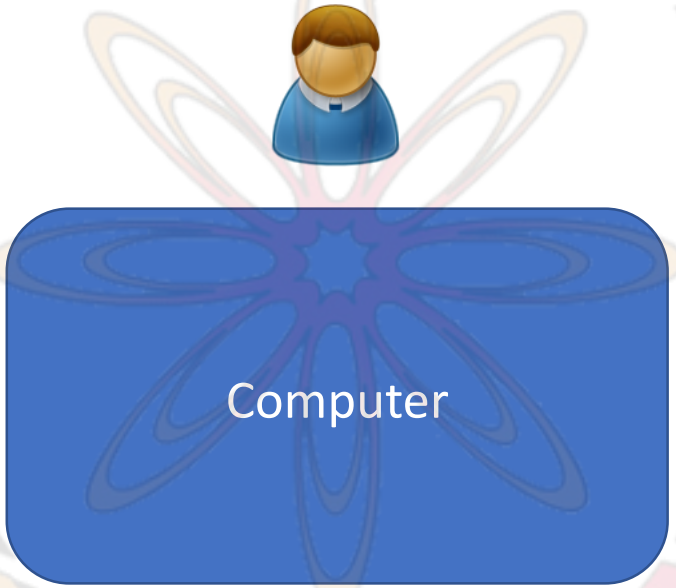
Multiple Instruction Multiple Data



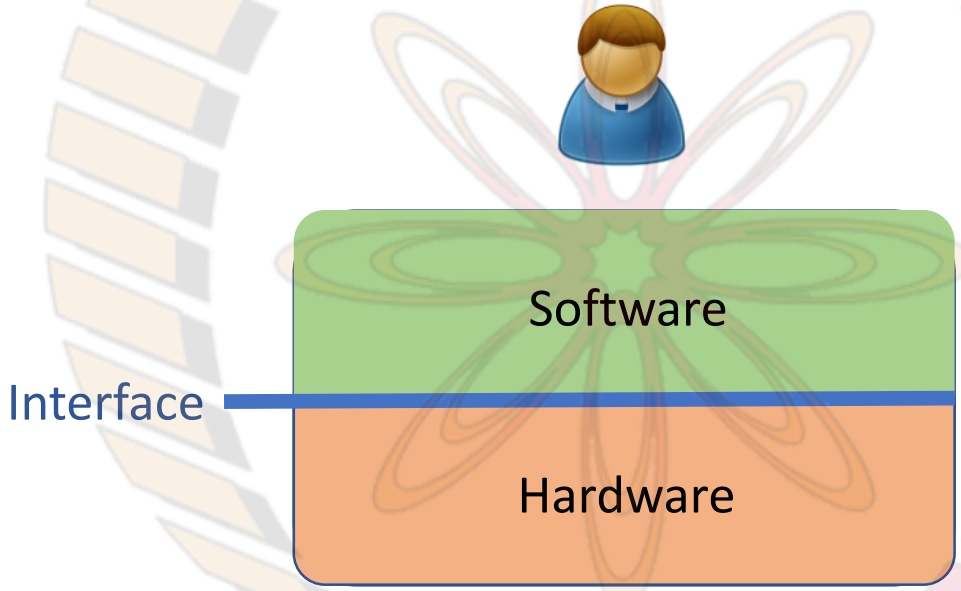
- Multiple independent processors concurrently executing different instructions on their respective data
- Most of the parallel computers are of this type
- Two varieties of implementation
  - Shared Memory
  - Distributed Memory

Basics of Process

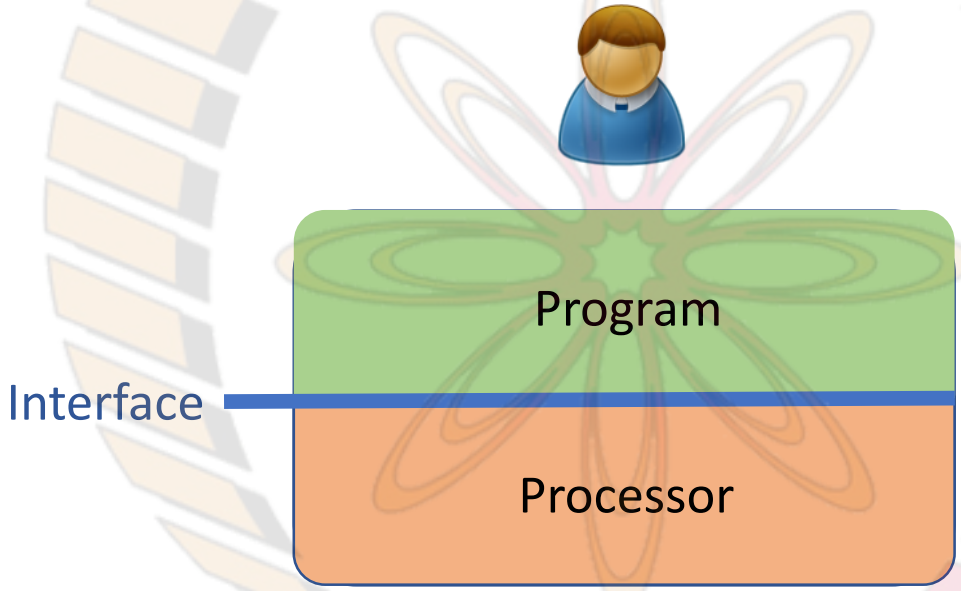
Basic Computing Model



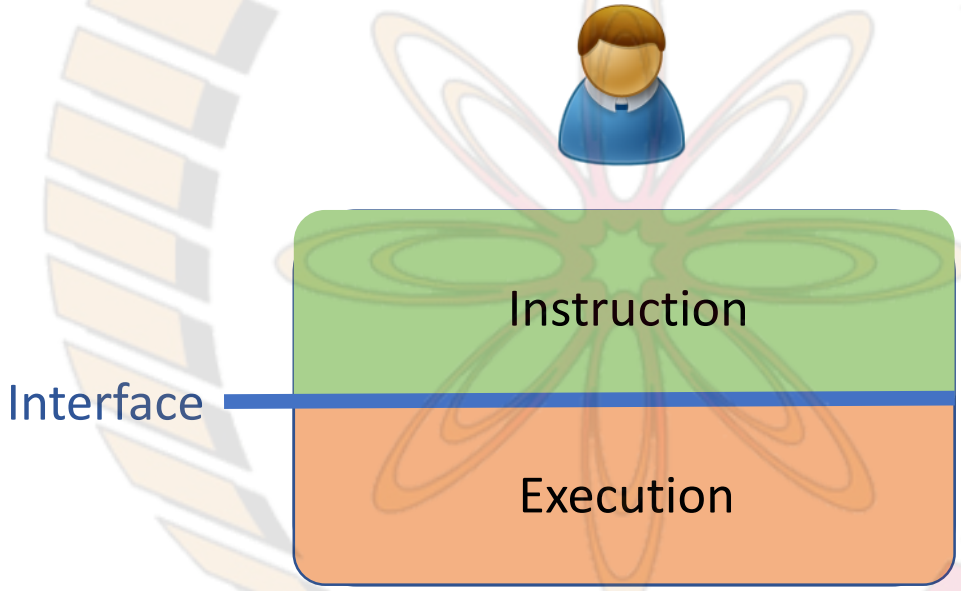
Basic Computing Model



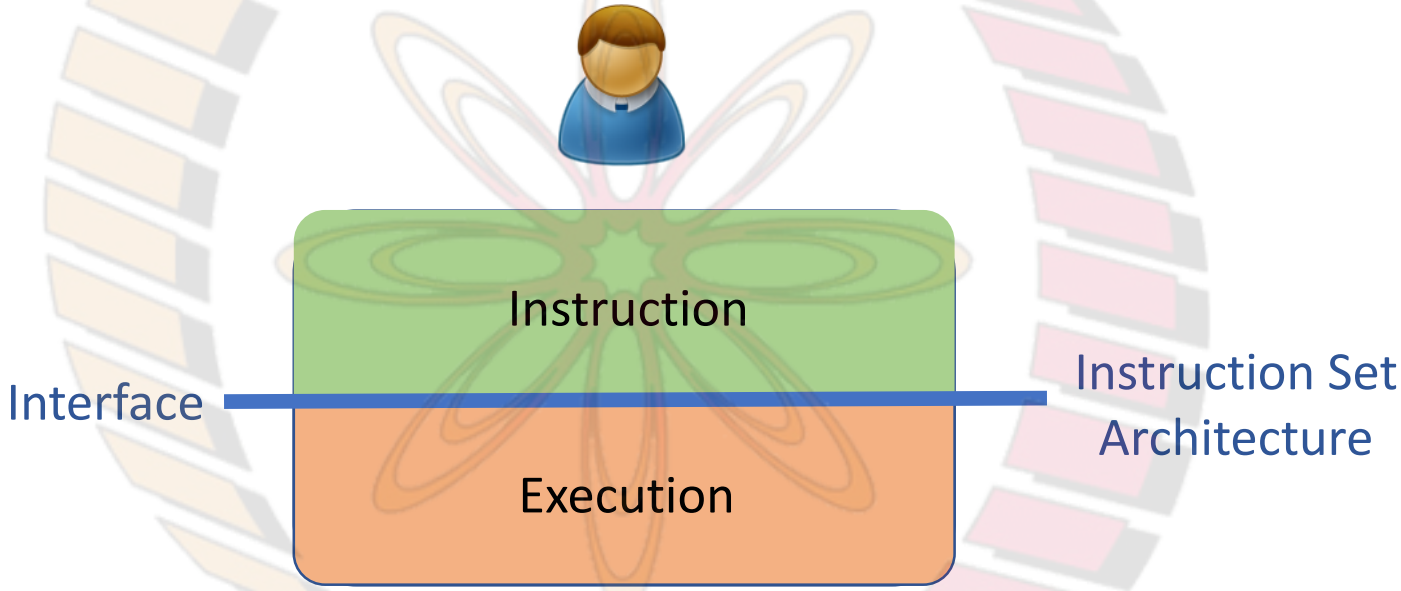
Basic Computing Model



Basic Computing Model

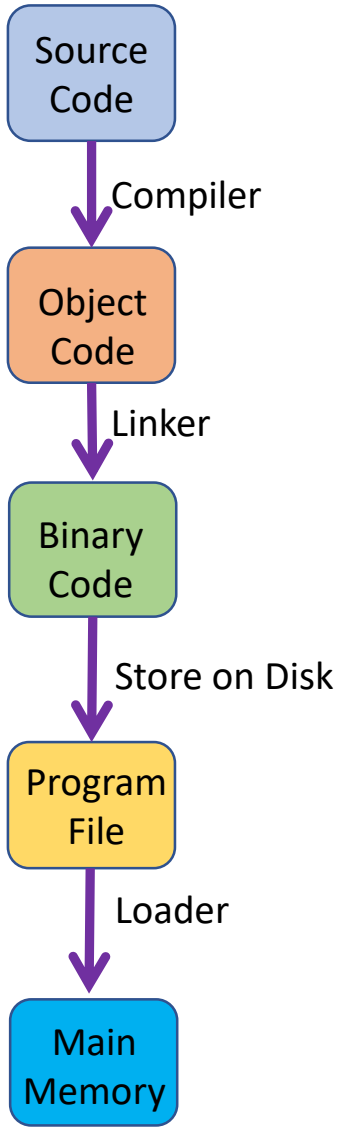


Basic Computing Model



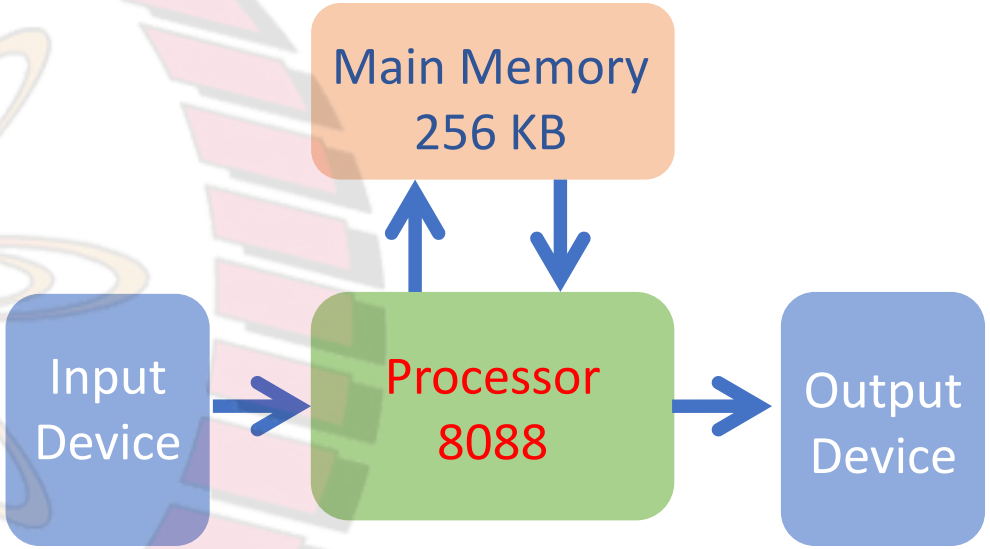
Process Basics

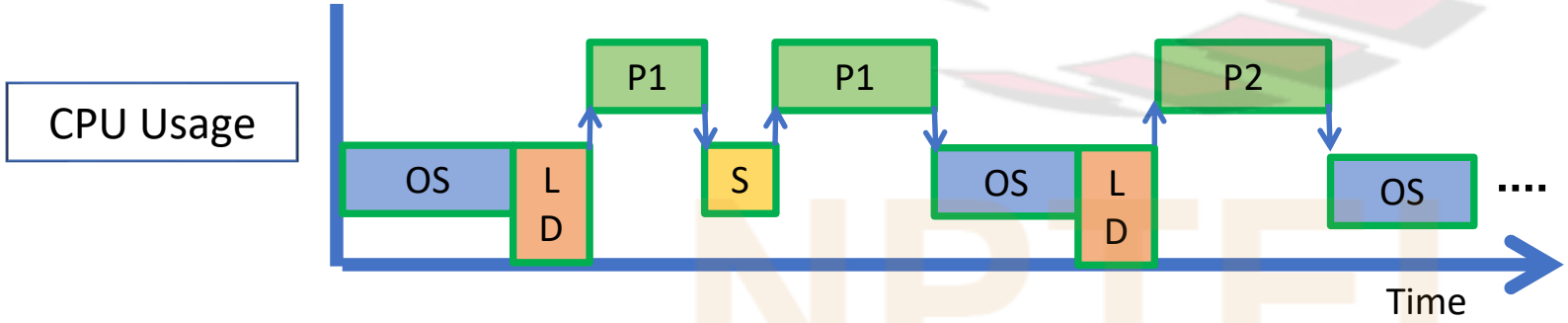
- Executable file is stored on secondary storage
- An user runs this program by typing the file name at command prompt
- The loader loads the program in main memory and runs it
- When a program gets into execution it is called a process
- To accomplish what it wants to do, while running a process needs resources like,
  - CPU time
  - Additional memory (usually data memory)
  - Files on hard disk
- Resources are allotted by the OS to a process either when it is created or while it is executing



Operating system and program

- Early computer systems allowed only one program to be executed at a time
- Consider example of original IBM PC
  - CPU - Intel 8088 at 4.77 Mhz
  - Memory - 256 KB
  - Operating System – MS DOS
- When running, the process has complete control of the system and use all its resources





LD – Loader

S – System Call

P1, P2 - Process

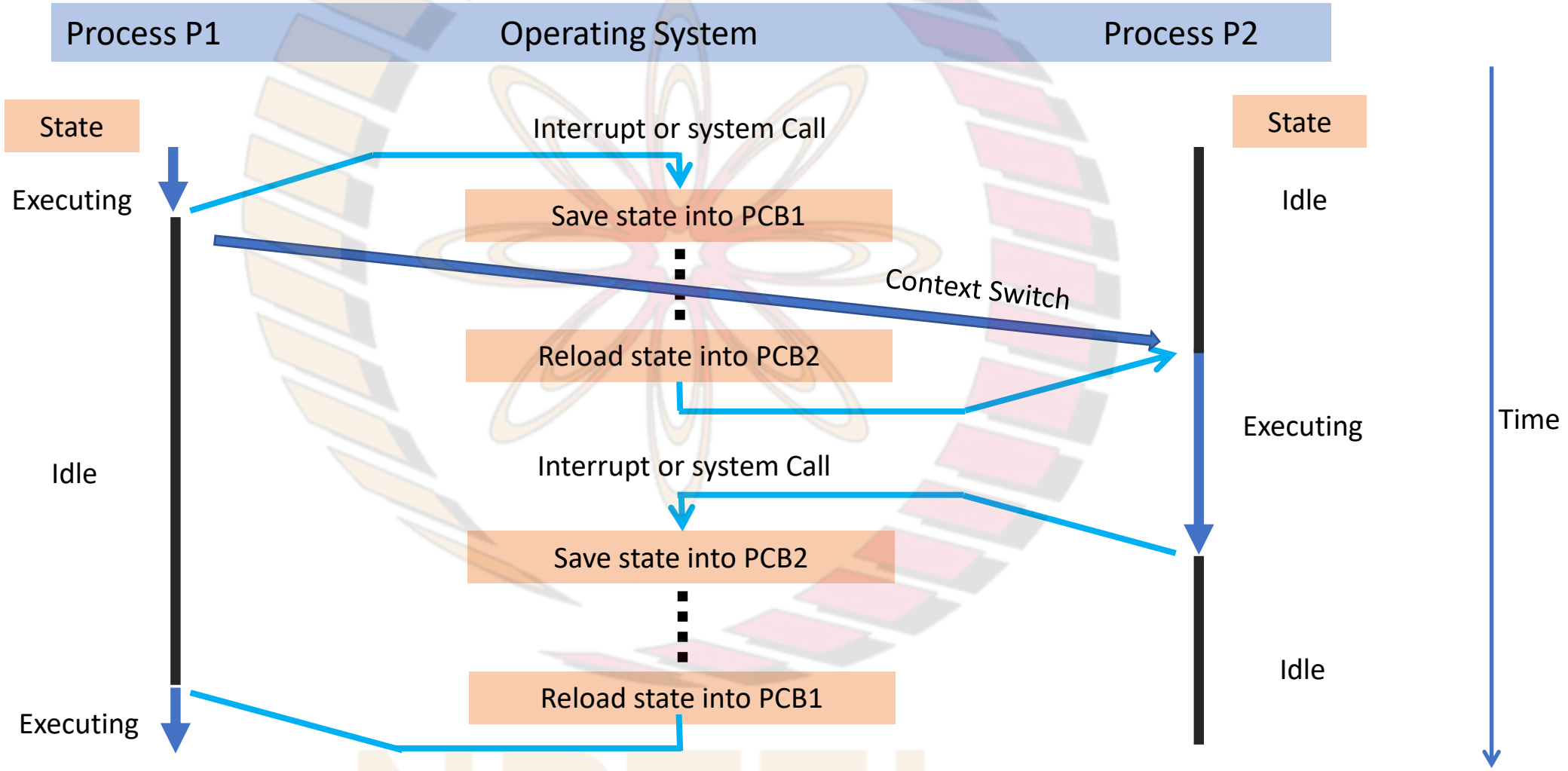
Multi-programming for optimal CPU usage

- CPU scheduling is the fundamental characteristics of multi-programming
- Single processor systems can run only one process at a time
- Other processes have to wait till the CPU becomes free
- The objective of multi-programming is keeping the CPU always busy
  - Some process has to be running at all the times
- A process keeps running till it has to wait, typically for an I/O request to complete
  - In a simple computing system like IBM PC with DOS, CPU just sits idle
- Multi-programming tries to use this idle time, in a productive manner
  - The OS takes the CPU from the waiting process and gives it to another process

Multitasking on a single CPU

- On a single CPU system operating system runs multiple programs (processes) on the CPU in time sliced manner
- Hardware support inside the CPU is needed to accomplish this
  - Not present in 8088, present in 80286 and later processors from Intel
- Each processes gets a slice of time to run on CPU
- When slice is over, the OS assigns another process to run on the CPU
- The OS can switch to another process, when a running process waits for I/O operation to complete or an event to take place
- Thus the OS runs multiple processes in quick succession to give an impression that all of these processes are running continuously

Multitasking between two processes on a single CPU



PCB: Process Control Block

Source: OS Principles - Galvin et al

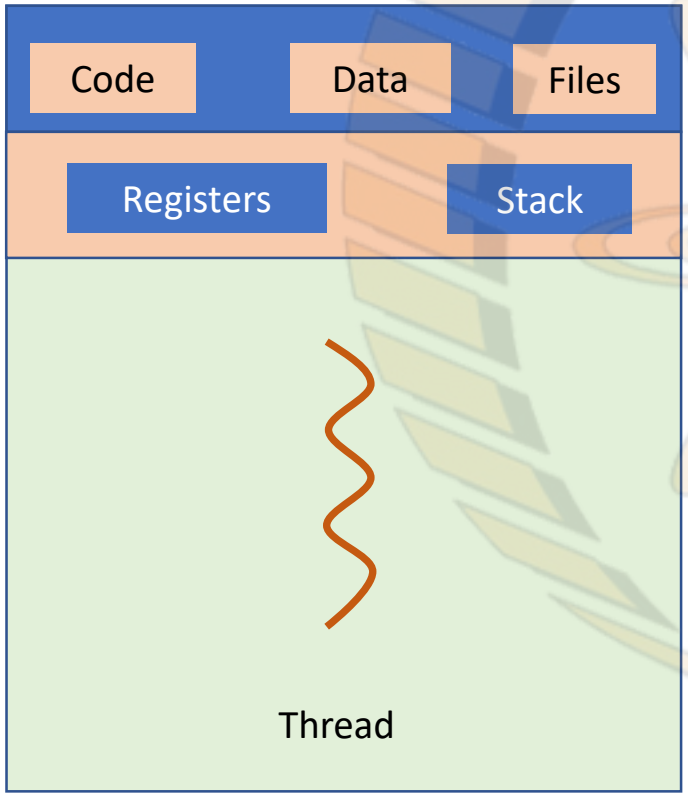
Threads

- Thread is a light weight process, which is a basic unit of CPU utilization
- The traditional process has a single thread
- Modern process consists of multiple threads

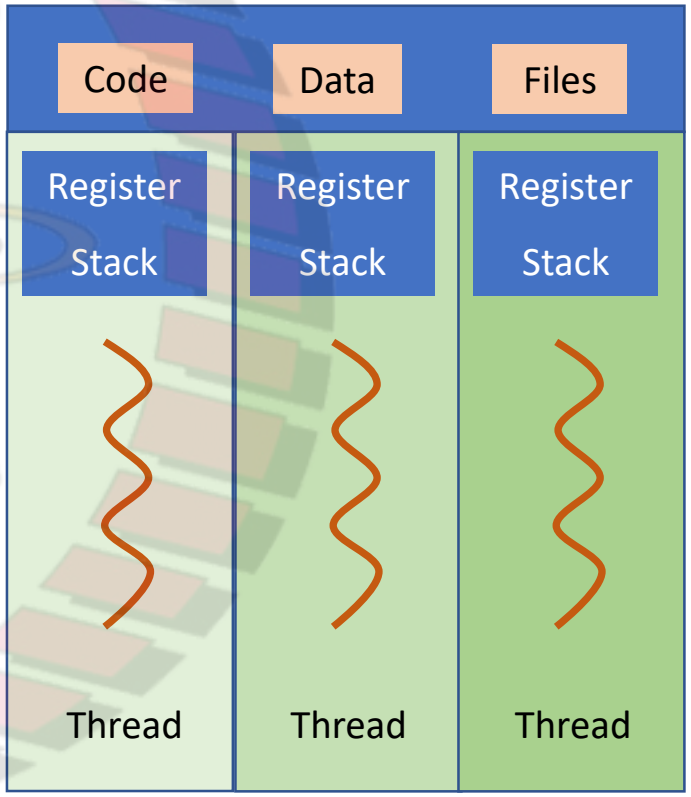
Unique to a Thread	What it shares with other threads of same process
Thread id	Code Section
Program Counter	Data section
Register set	OS resources like open files
Stack	

- The reason for having multiple threads is, the process can perform multiple tasks
- Sometimes a single application is required to perform several similar tasks
    - Webserver accepting multiple client requests

Single thread and Multi-thread process



Single Threaded Process



Multi Threaded Process

Source: OS Principles - Galvin et al

Benefits of Threads

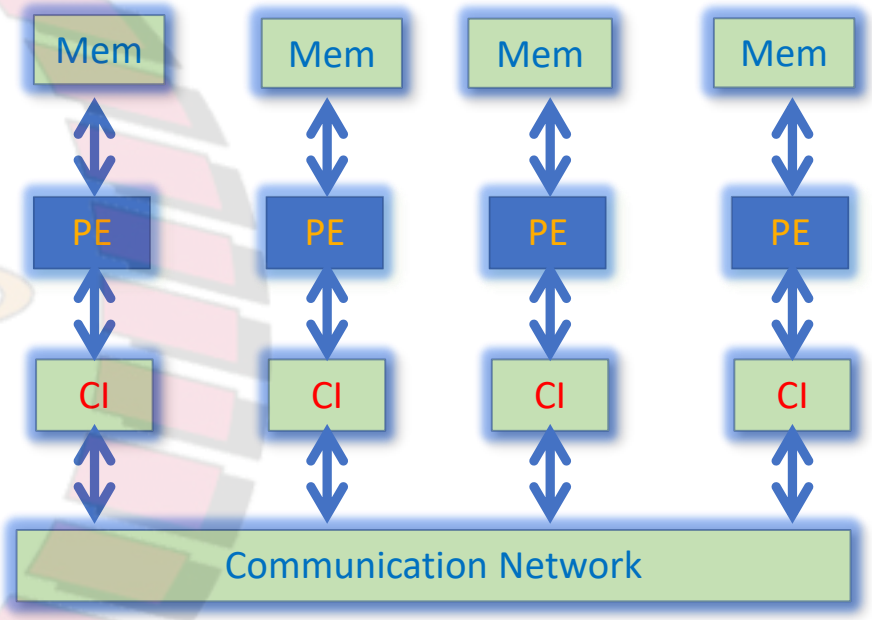
- Responsiveness : Even if part of a program (one of its thread) is blocked or executing lengthy operation, other part (other threads) can respond to user
- Resource Sharing: This allows an application to have serval different activities in the same address space
- Economy (of time): It is faster to create and switch context of threads than that of process
- Useful for Multiprocessor Architecture: The benefits of multi-threading is greatly increased when used with multiprocessor architecture

Scheduling Criteria

- CPU Utilization : Keep CPU as busy as possible. In real systems, the value typically varies between 40% and 90%
- Throughput: Measure of work done, in terms of number of processes that are completed in unit time
- Turnaround Time: for a process, is the time from the time of submission of process to the time, the process is completed
- Waiting time: Sum of the periods spent, waiting in ready queue

Multiprocessor Scheduling

- In systems having more than one processor, it becomes possible to share the load of processes / threads to be run
- However, the task of scheduling becomes complex
- In asymmetric multiprocessing the master server takes scheduling decisions
- In symmetric multiprocessing each processor does self-scheduling, picking up tasks from ready queue



Mem : Local Memory
PE: Processing Element
CI : Communication Interface

Intel's Hyper Threading technology

- Allows more than one thread to run on each core.
- The CPU exposes two execution contexts per physical core.
- One physical core is like two "logical cores" that can handle different threads
  - Duplicates the architectural state (takes 5% more die area)
  - Shares one set of execution resources
  - No loss in performance if single thread is run
- Two logical cores can run tasks more efficiently than a single-threaded core.
  - Overcoming idle time when waiting for one tasks to complete
- Intel claims this improves CPU throughput by up to 30% in server applications
- Hyperthreading is enabled through BIOS setting
- Hyperthreading is recommended to be disabled for HPC workloads

Summary

- Basic architecture is same across all types of computing platforms and consists of CPU, Memory and Input-Output
- Flynn’s taxonomy classifies parallel computing systems based on instruction streams and data streams
- The early implementations of CPUs were capable of running only one program at a time
- To make optimum use of CPU, techniques like multi-programming, multi-tasking, multi-threading etc. were introduced.
- Hyper-threading allows one physical CPU core to be viewed as two logical CPU cores, further improving the efficiency of CPU for certain types of workloads

Thank You