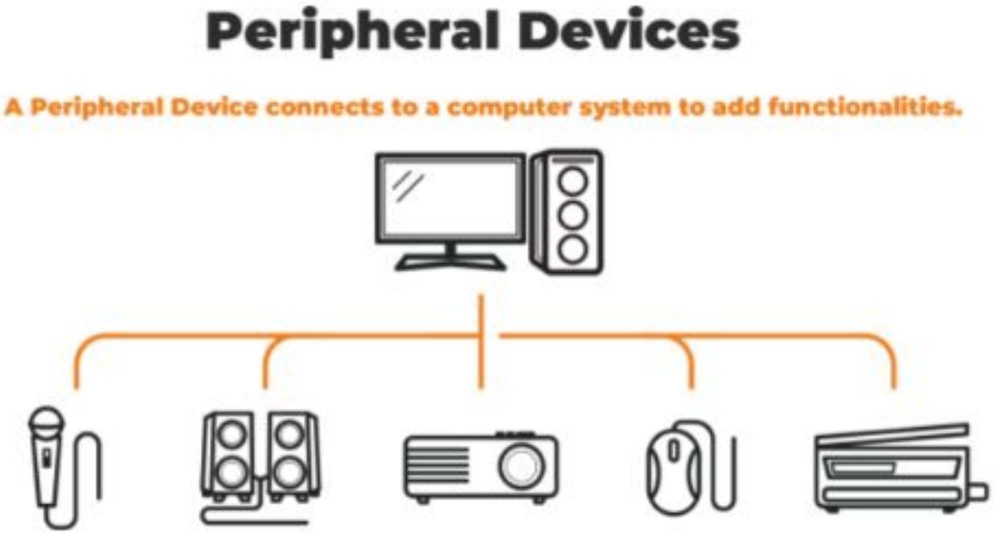
Input-Output Organization: Peripheral Devices



Peripheral Devices

A Peripheral Device connects to a computer system to add functionalities.

I/O bus and Interface module

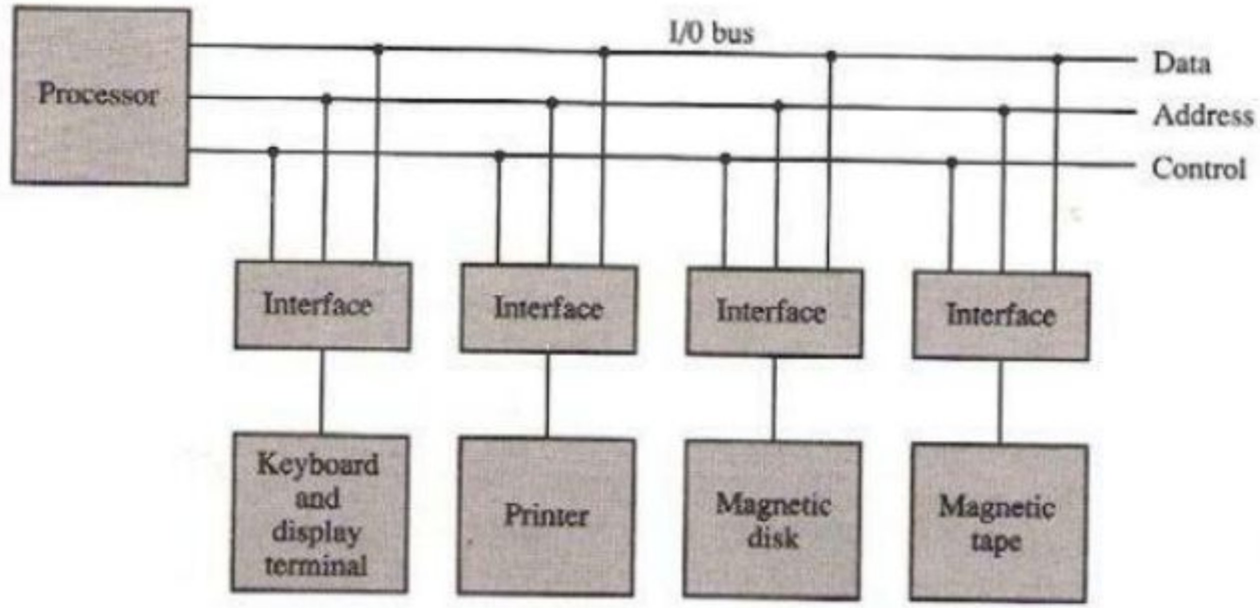


Fig: Connection of I/O bus to I/O devices

There are four types of I/O command

i) **Control command** The control command is issued to activate and inform the peripheral devices what they have to do.

ii) **Status command** The processor issues status command to test the status condition of interface and peripherals.

iii) **Output data command** It is issued to transfer data from system bus to one of storage register in I/O module.

iv) **Input data command** It is issued to transfer data from peripheral to one of its register in I/O module.

I/O and Memory bus

The CPU communicates with the memory and I/O. Both I/O and memory have data, address and control buses.

Three ways that computer buses can be used to communicate with memory and I/O.

i) Use two separate buses, one for memory and the other for I/O.

ii) Use one common bus for both memory and I/O but have separate control lines for each.

iii) Use one common bus for both memory and I/O with common control line

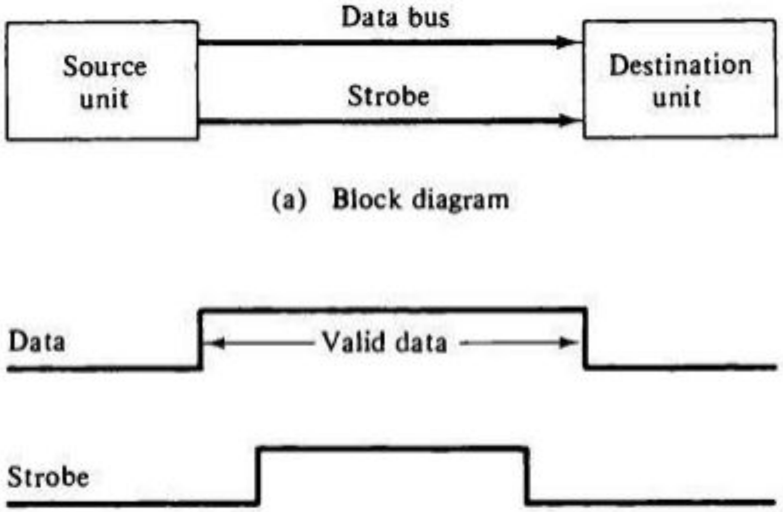
Types and advantages

Asynchronous Data Transfer

Asynchronous data transfer between two independent units requires that control signals be transmitted between the communicating units to indicate the time at which data is being transmitted.

1. **Strobe pulse** supplied by one of the units to indicate to the other unit when the transfer has to occur.
2. **Handshaking** The unit receiving the data item responds with another control signal to acknowledge receipt of the data

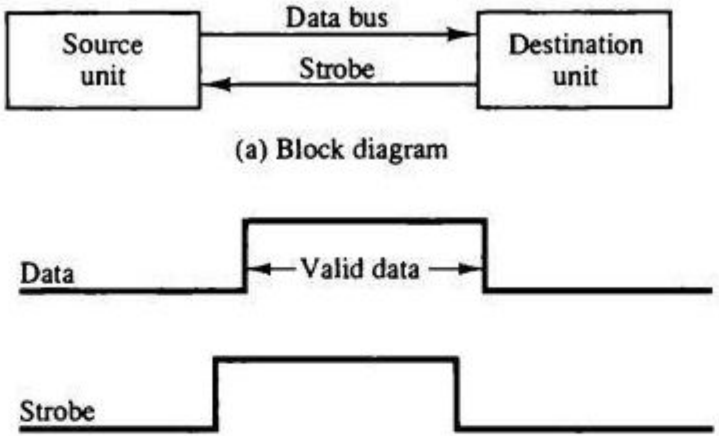
Strobe Control



(a) Block diagram

(b) Timing diagram

Fig: Source-initiated strobe for data transfer



(a) Block diagram

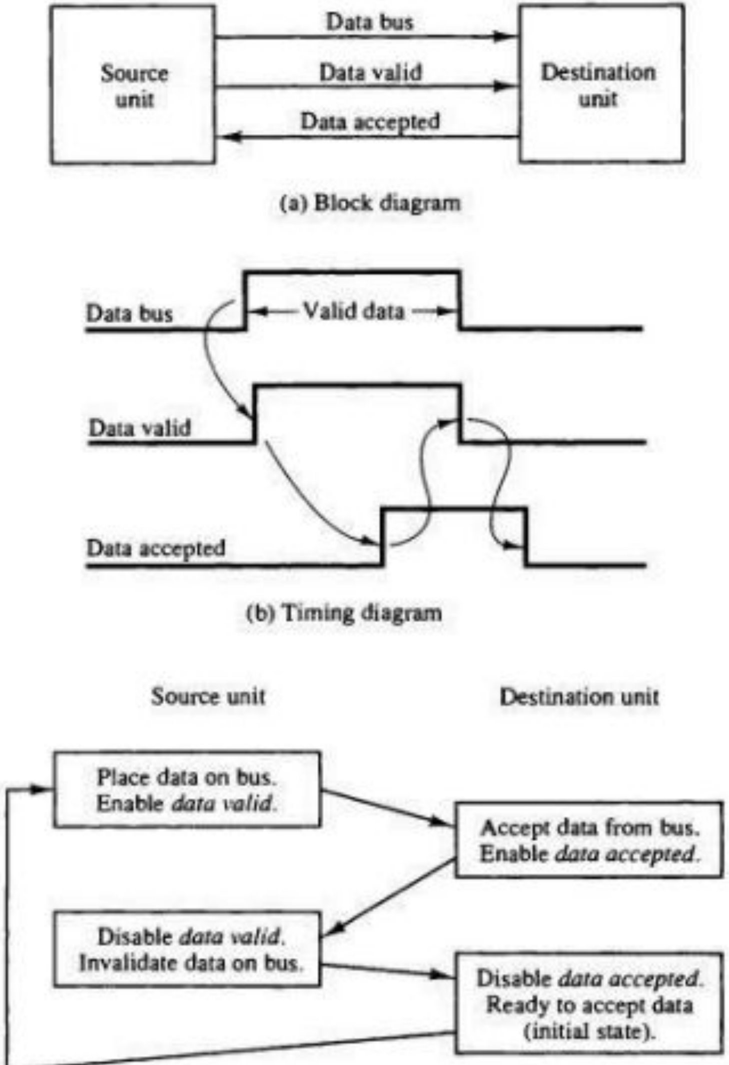
(b) Timing diagram

Fig: Destination-initiated strobe for data transfer

Handshaking

Source-initiated transfer

using handshaking



(a) Block diagram

(b) Timing diagram

Place data on bus. Enable *data valid*.

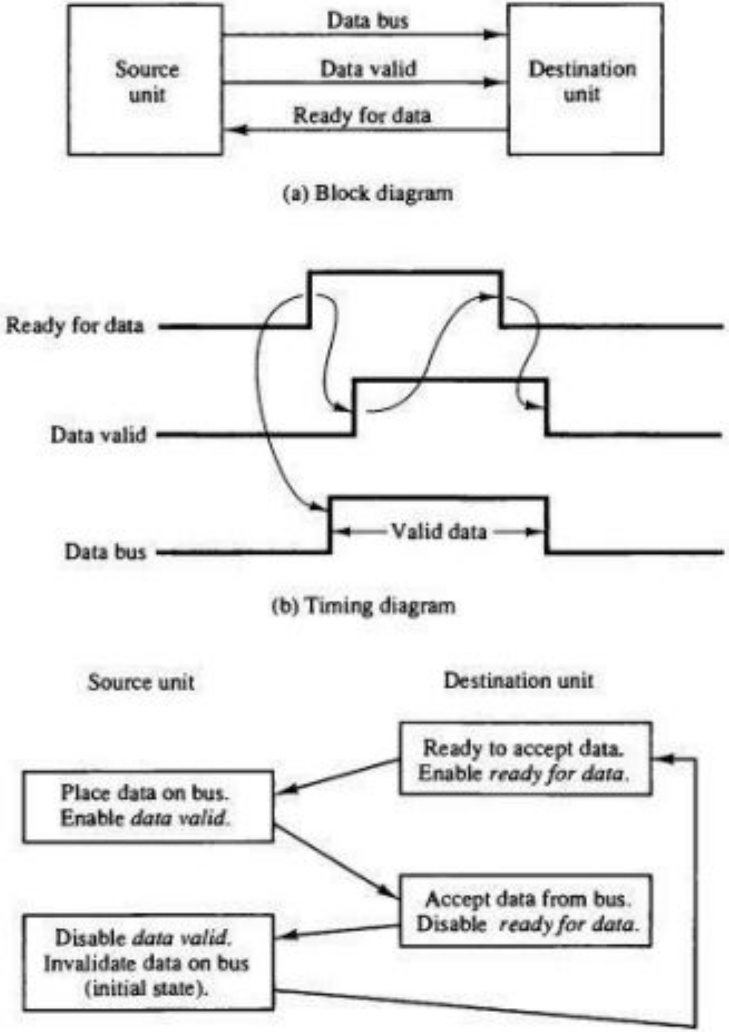
Accept data from bus. Enable *data accepted*.

Disable *data valid*. Invalidate data on bus.

Disable *data accepted*. Ready to accept data (initial state).

Destination-initiated transfer

using handshaking



(a) Block diagram

(b) Timing diagram

Ready to accept data. Enable *ready for data*.

Place data on bus. Enable *data valid*.

Accept data from bus. Disable *ready for data*.

Disable *data valid*. Invalidate data on bus (initial state).

Modes of Transfer

Interrupt

When a Process is executed by the CPU and when a user Request for another Process, then this will create disturbance for the Running Process. This is also called as the Interrupt.

Types of Interrupts

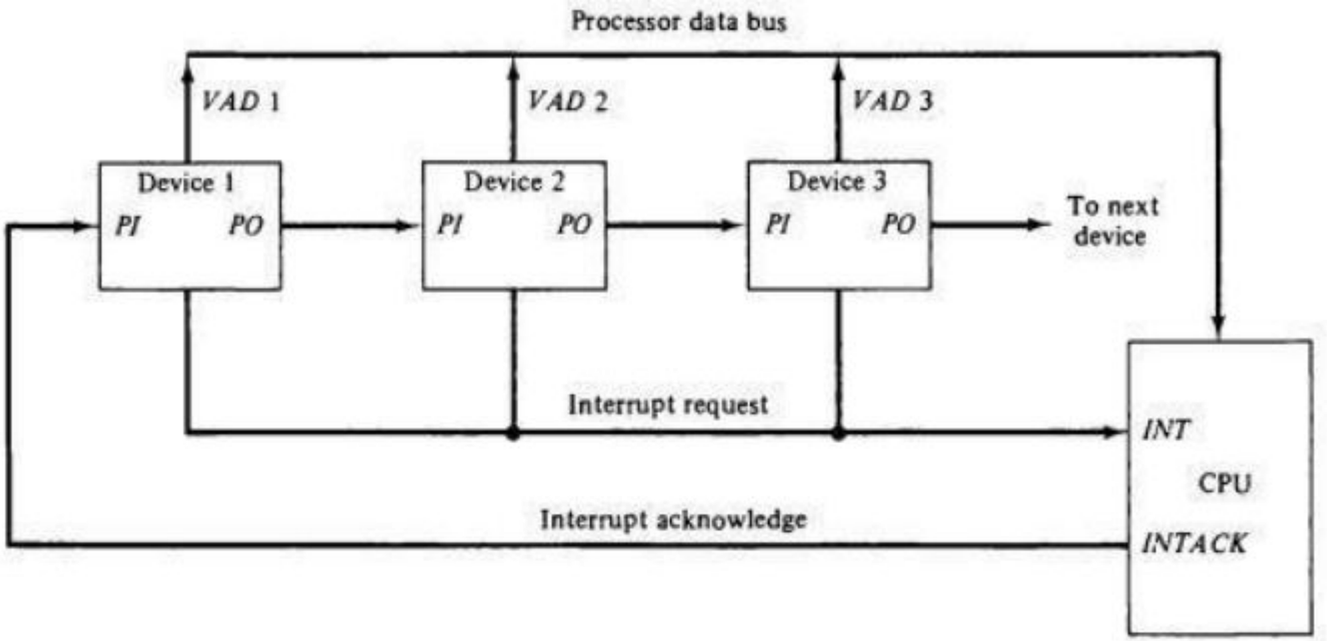
a) Internal Interrupt :The Internal Interrupts are those which are occurred due to some problem in the execution

b) Software Interrupt: The software interrupts are those which are made some call to the system.

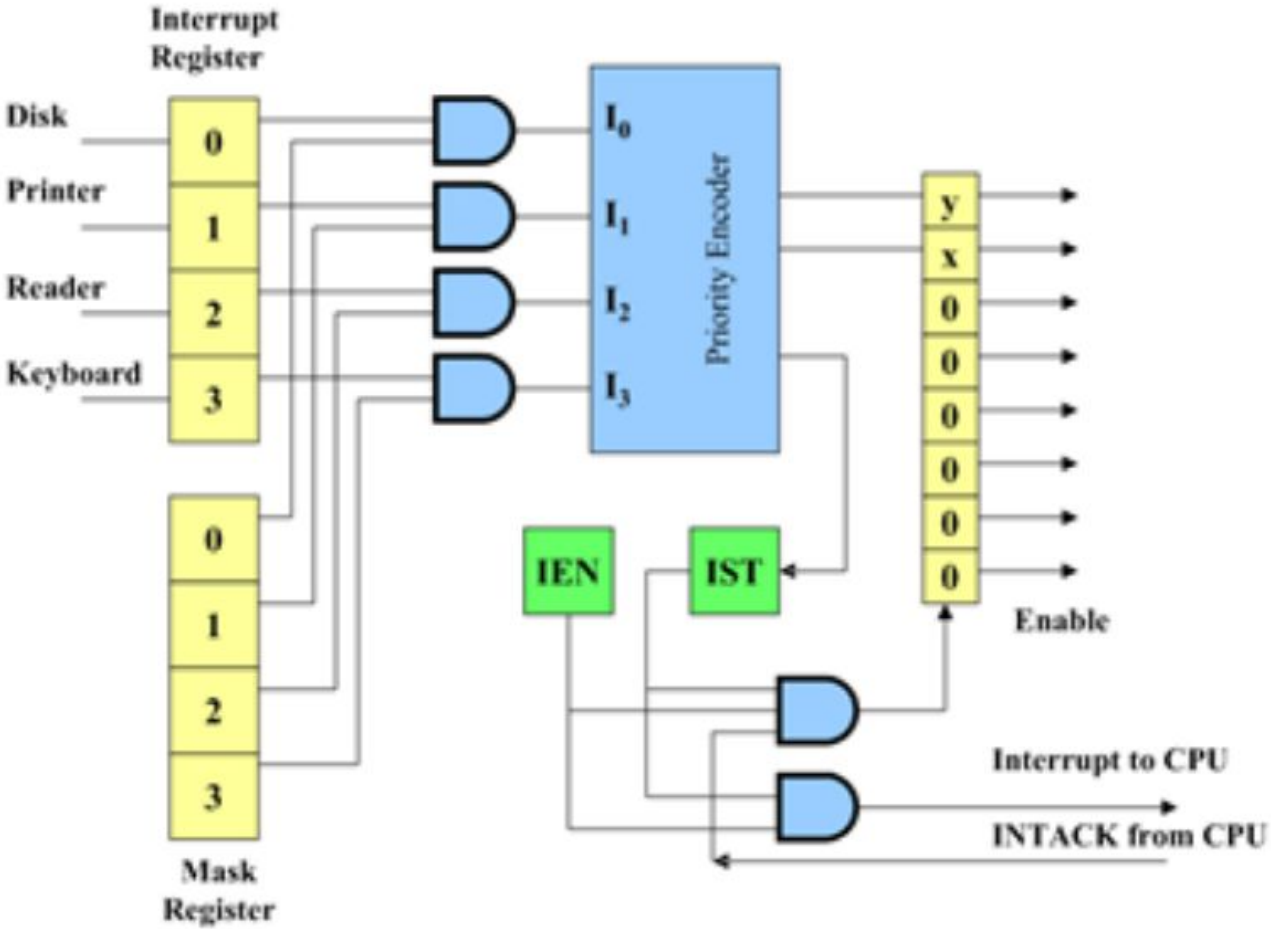
c) External Interrupt: The External Interrupt occurs when any input and output devices request for any operation and the CPU will execute those instructions first.

Priority Interrupt

Daisy-Chaining Priority

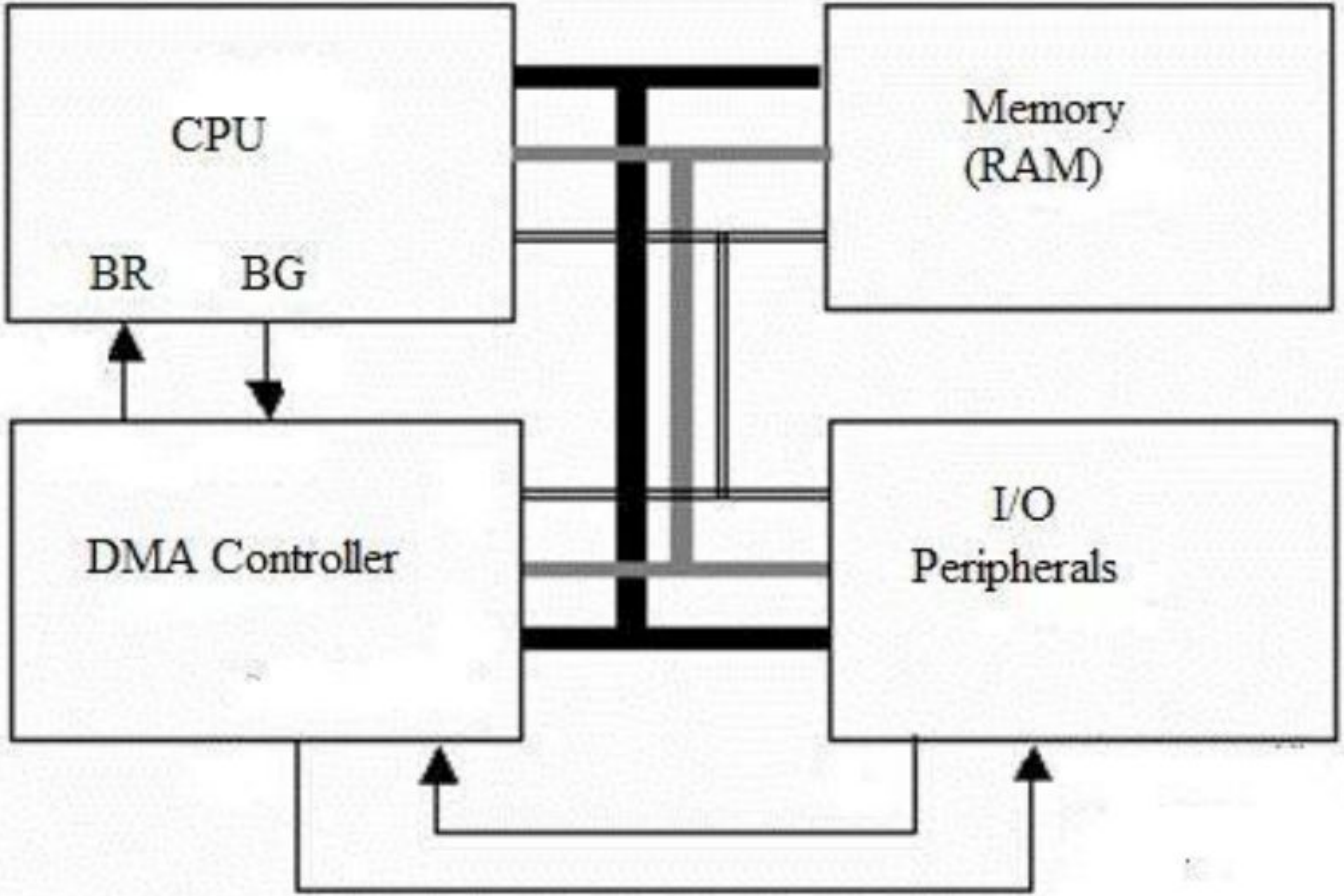


Parallel Priority Interrupt



Direct Memory Access (DMA)

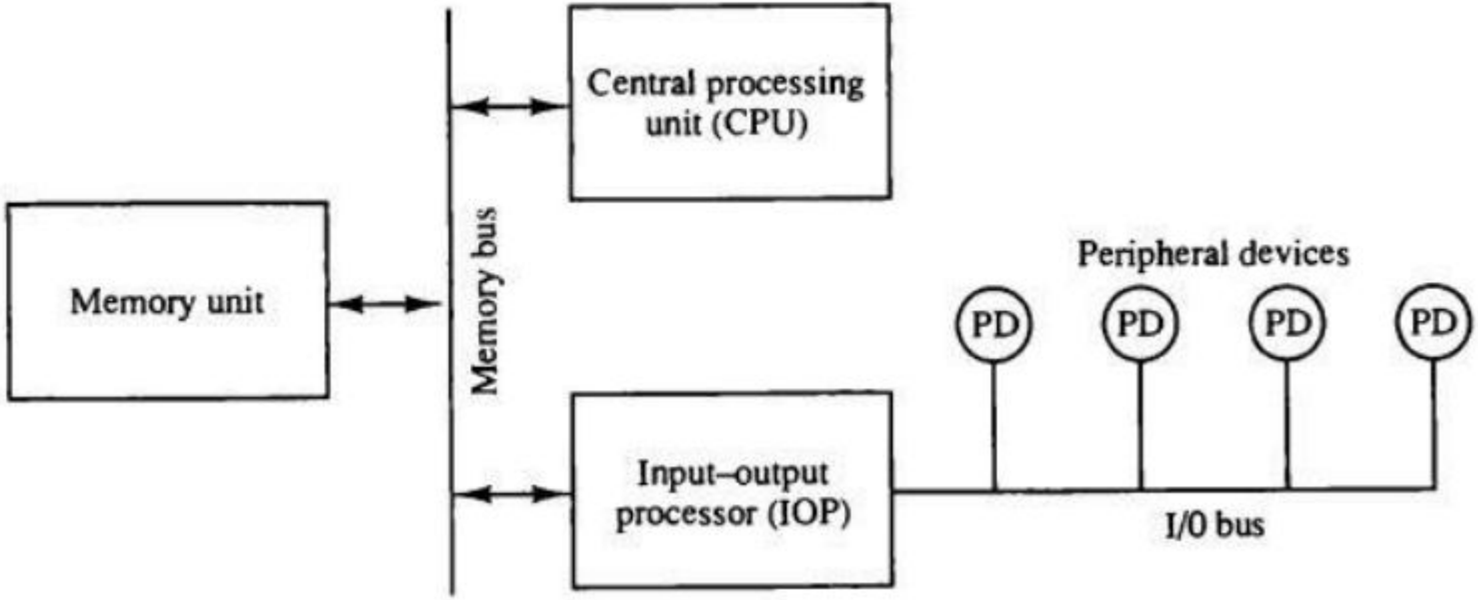
DMA is a sophisticated I/O technique in which a DMA controller replaces the CPU and takes care of the access of both, the I/O device and memory, for fast data transfers. Using DMA you get the fastest data transfer rates possible.


CPU
BR
BG
Memory
(RAM)
DMA Controller
I/O
Peripherals

Bus Request (BR): used by DMA controller to request the CPU for buses. When this input is active, CPU terminates the execution of the current instruction and places the address bus; data bus and read & write lines into high impedance state.

Bus Grant (BG): CPU activates BG output to inform DMA that buses are available (in high impedance state). DMA now take control over buses to conduct memory transfers without processor intervention. When DMA terminates the transfer, it disables the BR line and CPU disables BG and returns to normal operation.

Input-Output Processor (IOP)



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.

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.

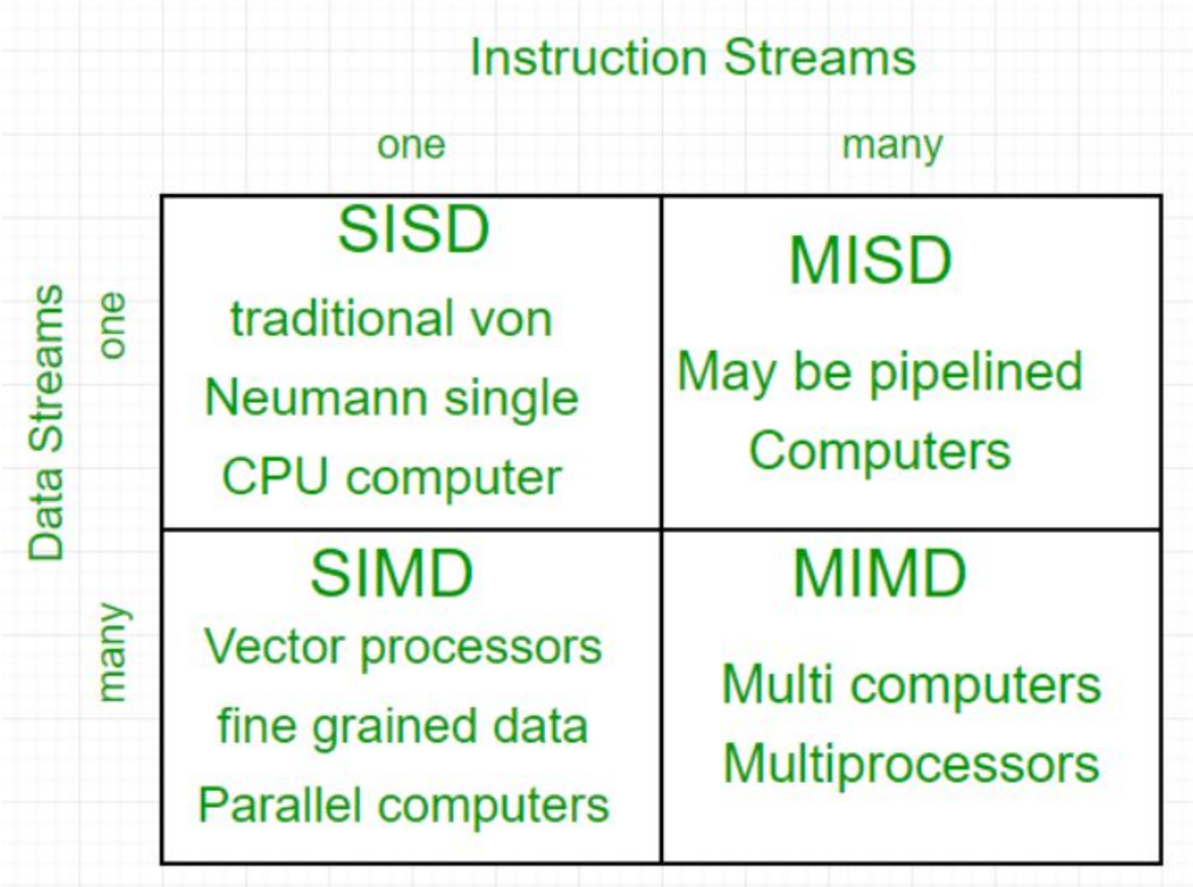
Advantages of Parallel Computing over Serial Computing are as follows:

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.

Types of Parallelism

- **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.
- **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 –** Limited by non-regular data manipulation patterns and by memory bandwidth

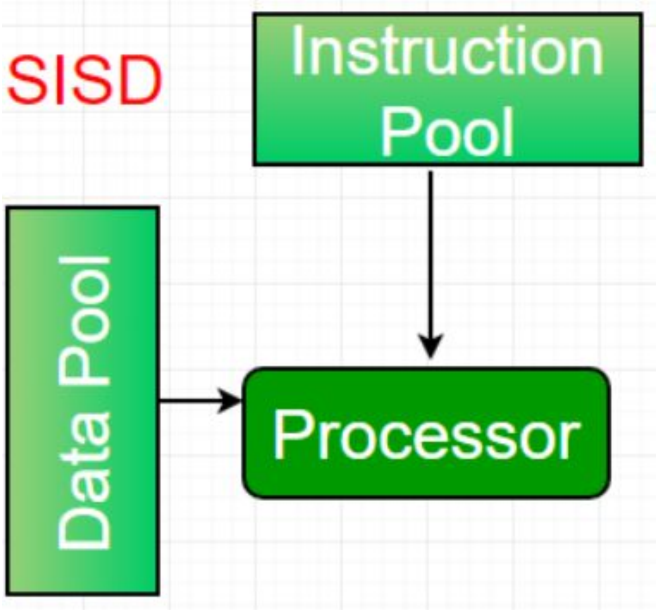
Flynn’s taxonomy



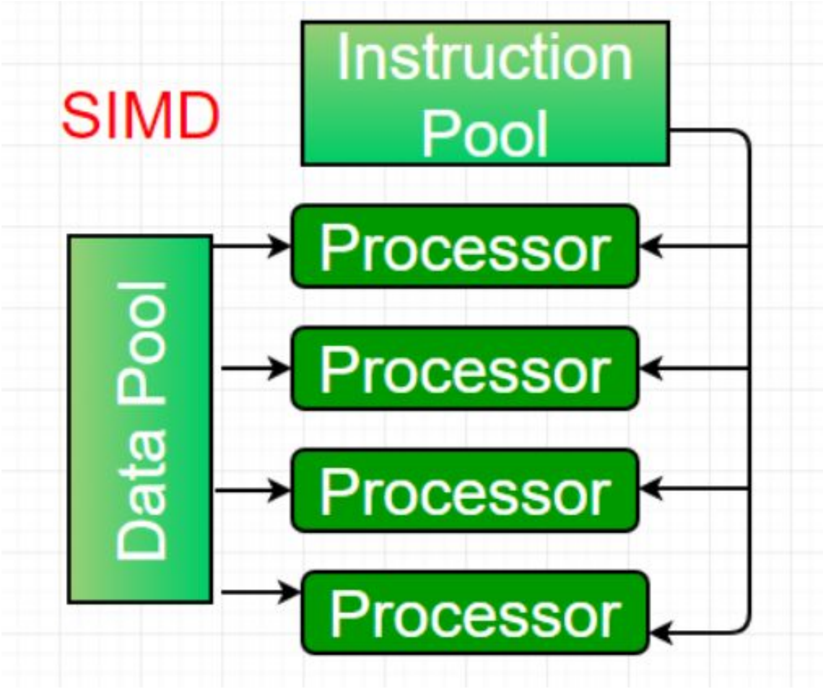
		Instruction Streams	
		one	many
Data Streams	one	SISD traditional von Neumann single CPU computer	MISD May be pipelined Computers
	many	SIMD Vector processors fine grained data Parallel computers	MIMD Multi computers Multiprocessors

Single-instruction, single-data (SISD) systems

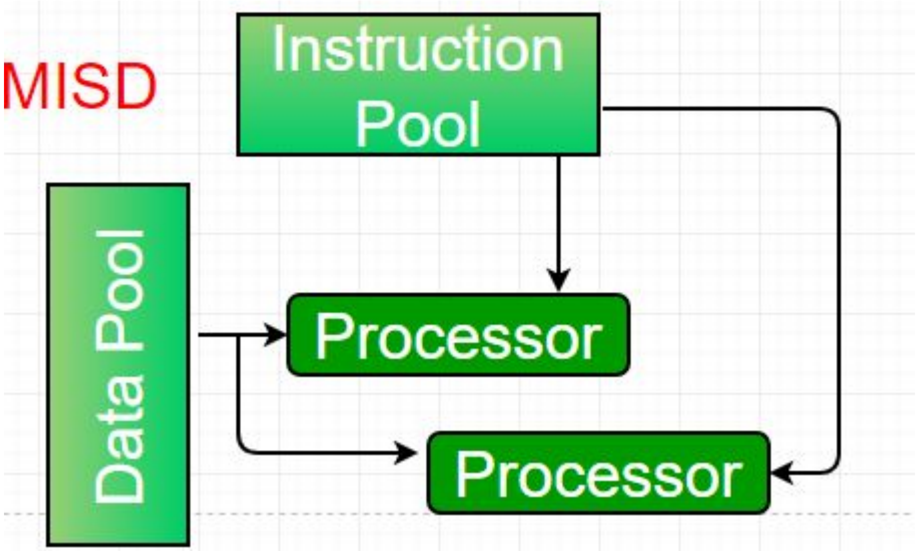
An SISD computing system is a uniprocessor machine which is capable of executing a single instruction, operating on a single data stream. In SISD, machine instructions are processed in a sequential manner and computers adopting this model are popularly called sequential computers.



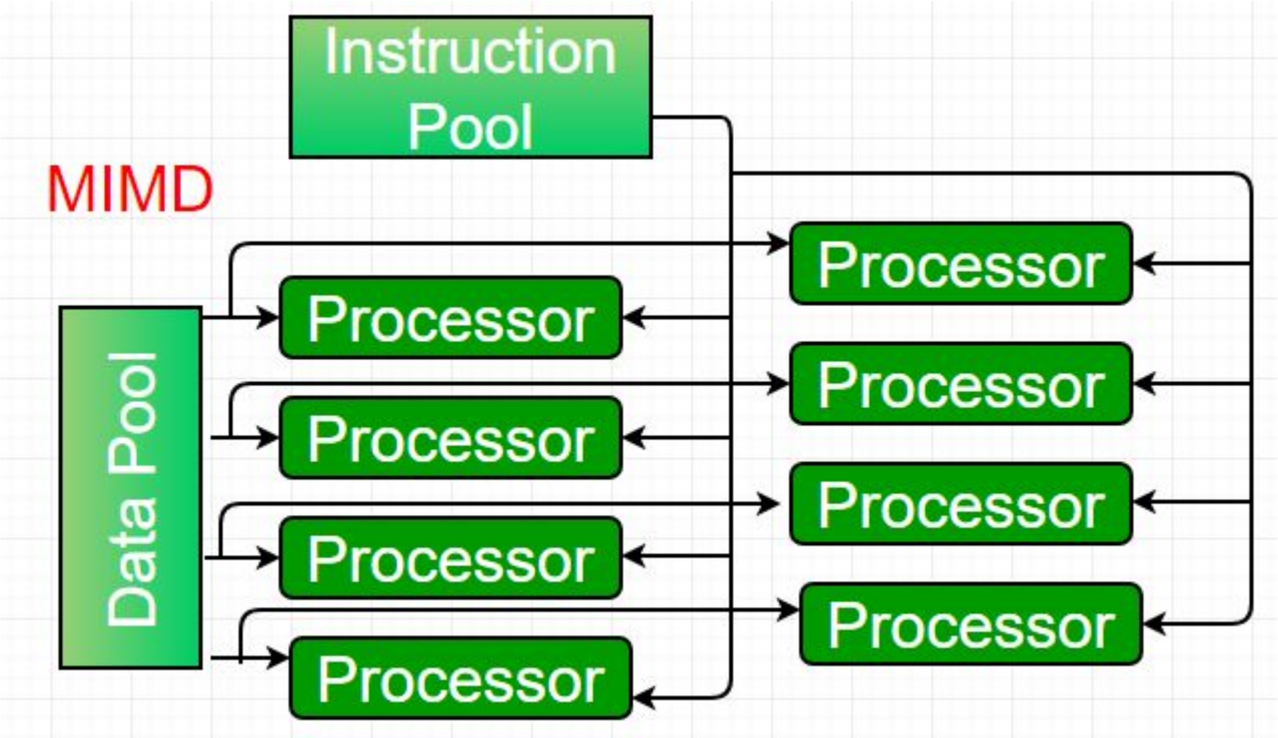
Single-instruction, multiple-data (SIMD) systems



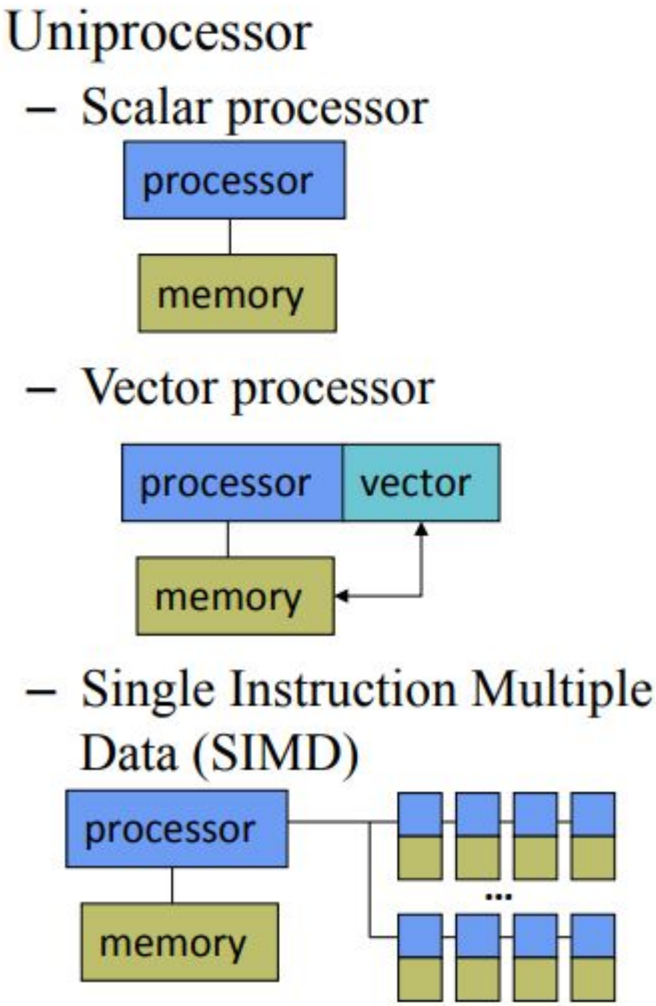
Multiple-instruction, single-data (MISD) systems



Multiple-instruction, multiple-data (MIMD) systems



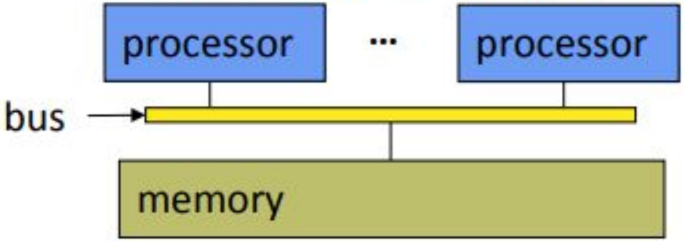
Parallel Architecture Types



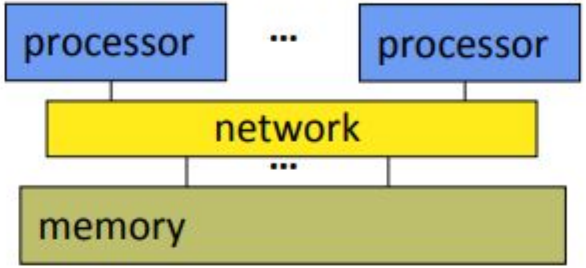
Uniprocessor

- Scalar processor
- Vector processor
- Single Instruction Multiple Data (SIMD)

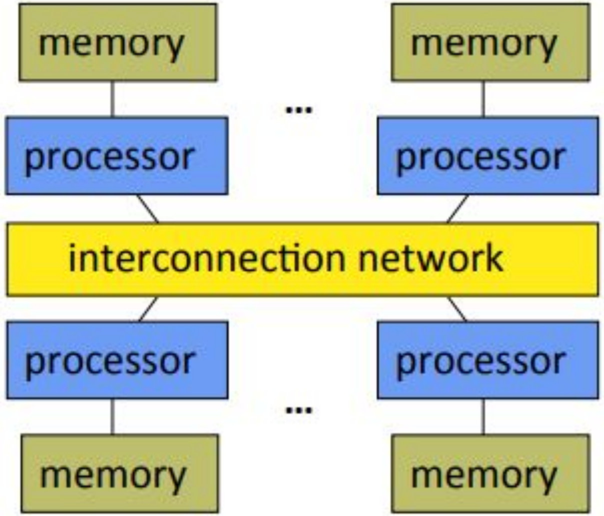
- Shared Memory Multiprocessor (SMP)
  - Shared memory address space
  - Bus-based memory system



  - Interconnection network

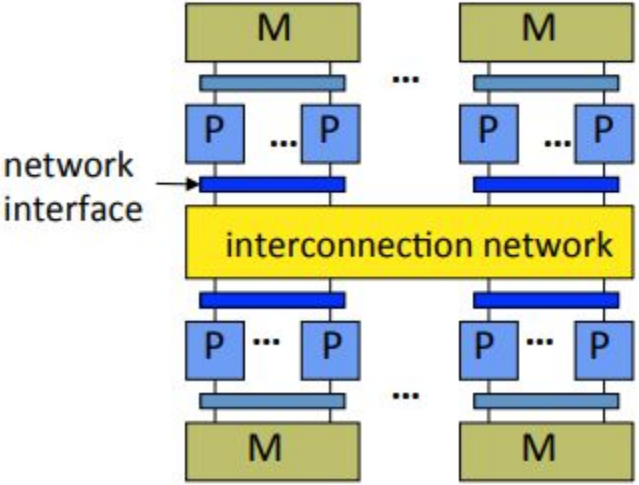


- Distributed Memory Multiprocessor
  - Message passing between nodes



  - Massively Parallel Processor (MPP)
    - Many, many processors

- Cluster of SMPs
  - Shared memory addressing within SMP node
  - Message passing between SMP nodes



  - Can also be regarded as MPP if processor number is large