COMPUTER ORGANIZATION

SYLLABUS:-

UNIT-I Flip-flops-Registers and shift registers-counters-decodersMultiplexers-PLDs-sequential circuits.Basic Structure of Computers. Functional UNITs:Basic operational concepts-Bus structures-performanceMultiprocessors and Multi computers-Historical Perspective.

UNIT-II Addressing Methods and Machine Program Sequencing: Basic Concepts:Memory locations and address, Main Memory operations, Instructions and Instruction Sequencing-Addressing Modes.

UNIT-III Input *I* Output organization:Accessing I/0 Devices-InterruptsDirect Memomry Access-1/O Hardware-Standard I/O Interface.

UNIT-IV MemorySystemConcepts:-SemiconductorRAM Meories-Read only memories-Cache Memories-PerformanceConsiderations-VirtualMemories:Memory Management Requirements,Arithmetic:-Addition and subtraction of sign members-Design of fast adders-Multiplication of positive members-Signed operand multiplication-Fast multiplicationInteger division-Floating point numbers and operations.

UNIT-V Basic Processing UNIT: Concepts-execution of a complete instructionMultiple-Bus organization-Hardwarorganization- hardware control-Micro Programmed Control. Pipelining: Concepts-Data hazards-Instruction hazards- Influence on Instruction sets-data path and control constructions.

LECTURE NOTES

UNIT-I

Flip-Flop:

A flip-flop in digital electronics is a circuit with two stable states that can be used to store binary data. The stored data can be changed by applying varying inputs. Flip-flops and latches are fundamental building blocks of digital electronics systems used in computers, communications, and many other types of systems. Both are used a data storage elements.

There are basically 4 types of flip-flops:

1. SR Flip-Flop
2. JK Flip-Flop
3. D Flip-Flop
4. T Flip-Flop

Registers and shift registers:

Registers-

A register may hold an instruction, a storage address, or any kind of data (such as a bit sequence or individual characters). Some instructions specify registers as part of the instruction. For example, an instruction may specify that the contents of two defined registers be added together and then placed in a specified register.

Registers are a type of computer memory used to quickly accept, store, and transfer data and instructions that are being used immediately by the CPU. The registers used by the CPU are often termed as Processor registers.

A processor register may hold an instruction, a storage address, or any data (such as bit sequence or individual characters).

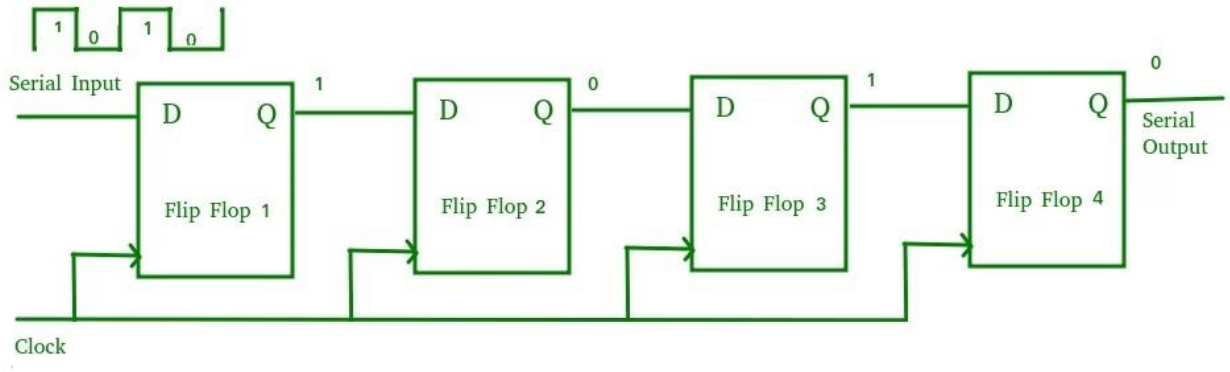
- program counter.
- memory address register (MAR)
- memory data register (MDR)
- current instruction register (CIR)

- accumulator (ACC)

Shift registers-

shift register is **a digital memory circuit found in calculators, computers, and data-processing systems**. Bits (binary digits) enter the shift register at one end and emerge from the other end. The two ends are called left and right. Flip flops, also known as bistable gates, store and process the data.

- Serial-in/serial-out.
- Parallel-in/serial-out.
- Serial-in/parallel-out.
- Universal parallel-in/parallel-out.
- Ring counter.



Counters:

In digital logic and computing, a **counter** is a device which stores (and sometimes displays) the number of times a particular event or process has occurred, often in relationship to a clock. The most common type is a sequential digital logic circuit with an input line called the *clock* and multiple output lines. The values on the output lines represent a number in the binary or BCD number system. Each pulse applied to the clock input increments or decrements the number in the counter.

Counters are used not only **for counting but also for measuring frequency and time ; increment memory addresses** . Counters are specially designed synchronous sequential circuits, in which , the state of the counter is equal to the count held in the circuit by the flip flops.

Decoders:

A decoder is **a combinational circuit that modifies binary data from n input lines to a maximum of 2n unique output lines**. An encoder creates the binary code corresponding to the input activated. A decoder gets a set of binary inputs and activates only the output that complements that input number.

A decoder is a circuit that **changes a code into a set of signals**. It is called a decoder because it does the reverse of encoding, but we will begin our study of encoders and decoders with decoders because they are simpler to design.

Multiplexer:

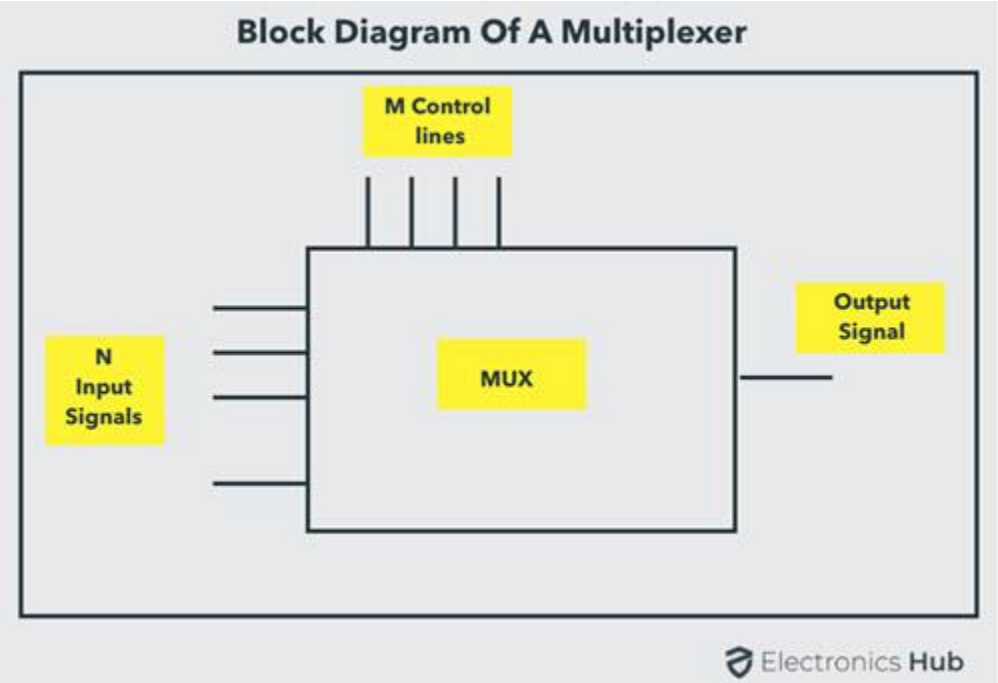
Multiplexing, or *muxing*, is a way of sending multiple signals or streams of information over a communications link at the same time in the form of a single, complex signal. When the signal reaches its destination, a process called *demultiplexing*, or *demuxing*, recovers the separate signals and outputs them to individual lines.

Multiplexing is a method used by networks to consolidate multiple signals -- digital or analog -- into a single composite signal that is transported over a common medium, such as a fiber optic cable or radio wave. When the composite signal reaches its destination, it is demultiplexed, and the individual signals are restored and made available for processing.

It is **the process in which multiple signals coming from multiple sources are combined and transmitted over a single communication/physical line.**

The advantage of multiplexing is that **we can transmit a large number of signals to a single medium**. This channel can be a physical medium like a coaxial, metallic conductor or a wireless link and will have to handle multiple signals at a time. Thus the cost of transmission can be reduced.

PLD:



Programmable Logic Devices PLDs are the integrated circuits. They contain an array of AND gates & another array of OR gates. There are three kinds of PLDs based on the type of arrays, which has programmable feature. Programmable Read Only Memory. Programmable Array Logic.

PLDs provide specific functions, including **device-to-device interfacing, data communication, signal processing, data display, timing and control operations**, and almost all works will perform.

Logic gates, multiplexers, demultiplexers, arithmetic circuits, etc., are some examples. Sequential logic devices such as flip-flops, counters, registers, etc., to be discussed in the following chapters, also belong to this category of logic devices.

Sequential circuits:

The sequential circuit is a special type of circuit that has a series of inputs and outputs. The outputs of the sequential circuits depend on both the combination of present inputs and previous outputs. The previous output is treated as the present state. So, the sequential circuit contains the combinational circuit and its memory storage elements. A sequential circuit doesn't need to always contain a combinational circuit. So, the sequential circuit can contain only the memory element.

The sequential circuits can be **event driven, clock driven and pulse driven**.

Basic structure of Computers:

The basic structure of computers is the components that are basic to the performance and functioning of the computer. It is a simple concept that describes how any data is entered into the central processing unit with the help of input devices, such as a mouse, scanner, keyboard, joystick, and others, and printed on the screen by the output unit. There are three components of the basic structure of a computer. These components are the control processing unit (CPU), an input unit, and an output unit. The memory units and a control unit also form the basic structure of a computer. The memory unit stores the data, and the control unit sends the commands.

A general-purpose computer has four main components: **the arithmetic logic unit (ALU), the control unit, the memory, and the input and output devices** (collectively termed I/O). These parts are interconnected by buses, often made of groups of wires.

Functional unit:-

Basic Operational Concepts:

A Computer has five functional independant units like Input Unit, Memory Unit, Arithmetic & Logic Unit, Output Unit, Control Unit.

Input Unit :-

Computers take coded information via input unit. The most famous input device is keyboard. Whenever we press any key it is automatically being translated to corresponding binary code & transmitted over a cable to memory or proccessor.

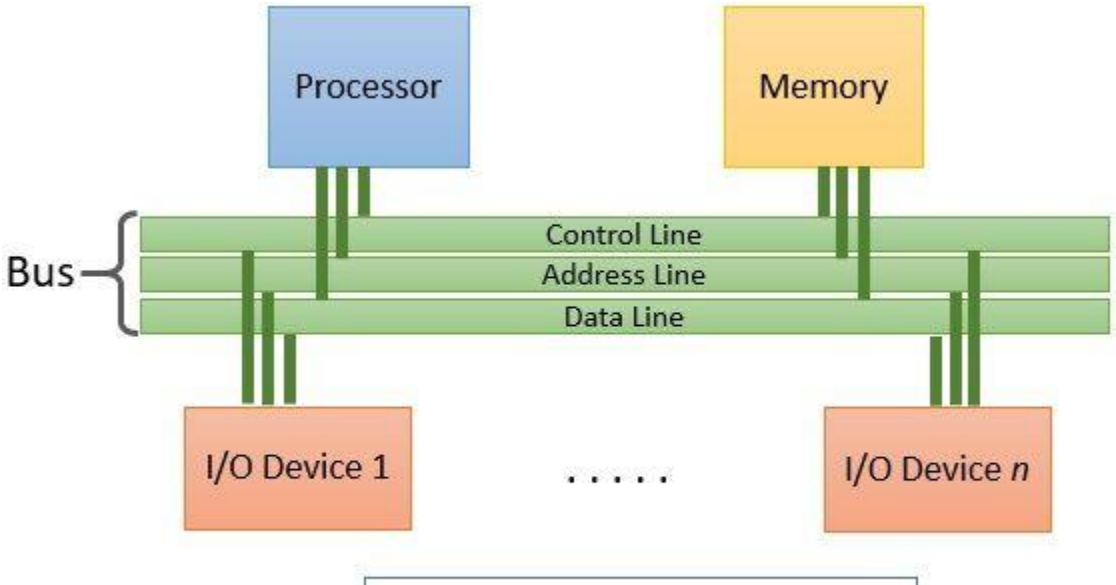
Basic Operational Concepts

- Instructions take a vital role for the proper working of the computer.
- An appropriate program consisting of a list of instructions is stored in the memory so that the tasks can be started.
- The memory brings the Individual instructions into the processor, which executes the specified operations.
- Data which is to be used as operands are moreover also stored in the memory.

Bus Structures:

The bus in the computer is the **shared transmission medium**. This means multiple components or devices use the same bus structure to transmit the information signals to each other a time only one pair of devices can use this bus to communicate with each other successfully.

A system bus has typically from fifty to hundreds of distinct lines where each line is meant for a certain function. These lines can be categories into three functional groups i.e., data lines, address lines, and control lines. Let us discuss them one by one each.



Bus Structure

Performance of Functional Units:

Functional units are a part of a CPU that **performs the operations and calculations called for by the computer program**. Functional units of a computer system are parts of the CPU (Central Processing Unit) that performs the operations and calculations called for by the computer program.

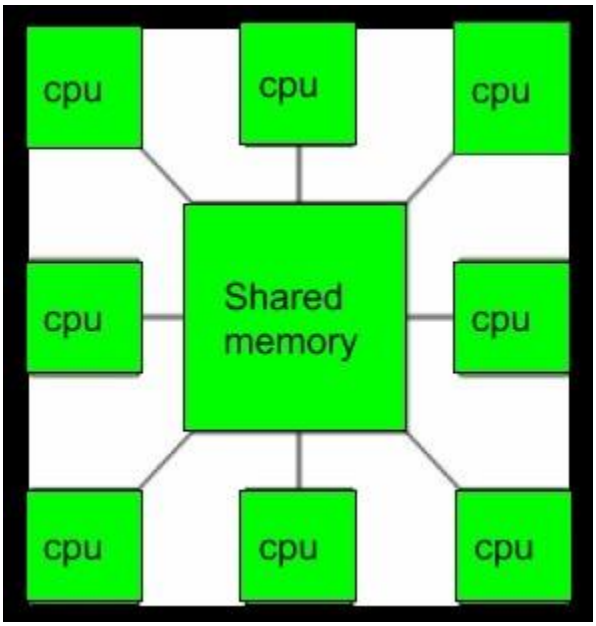
Computer performance metrics (things to measure) include availability, response time, channel capacity, latency, completion time, service time, bandwidth, throughput, relative efficiency, scalability, performance per watt, compression ratio, instruction path length and speed up. CPU benchmarks are available.

Multi-Processors and Multi-Computers:

Multi-Processors-

A Multiprocessor is a computer system with two or more central processing units (CPUs) share full access to a common RAM. The main objective of using a multiprocessor is to boost the system's execution speed, with other objectives being fault tolerance and application matching.

There are two types of multiprocessors, one is called shared memory multiprocessor and another is distributed memory multiprocessor. In shared memory multiprocessors, all the CPUs shares the common memory but in a distributed memory multiprocessor, every CPU has its own private memory.

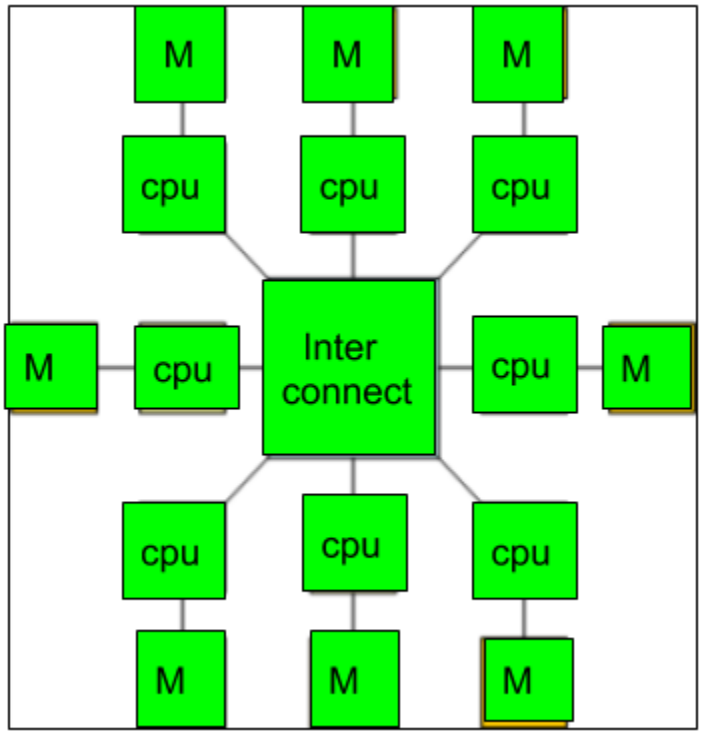


Multi-Computers-

A multicomputer is **a system consists of several individual computers connected in a network and each individual computer has its own computing resources such as I/O and memory devices**. In a multicomputer, the computers execute the streams of separate instructions and have their own memory.

A Multi-Computer system is a computer system with multiple processors that are connected together to solve a problem. Each processor has its own memory and it is accessible by that particular processor and those processors can communicate with each other via an interconnection network.

As the multicomputer is capable of messages passing between the processors, it is possible to divide the task between the processors to complete the task. Hence, a multicomputer can be used for distributed computing. It is cost effective and easier to build a multicomputer than a multiprocessor.



Historical Perspectives:

The history of the computer **dates back to several years**. There are five prominent generations of computers. Each generation has witnessed several technological advances which change the functionality of the computers. This results in more compact, powerful, robust systems which are less expensive.

While the conceptual idea behind a computer was developed in the 19th century, **the first electronic computer was developed in the 1940s**. Early computers used mechanical relays and vacuum tubes, which were replaced by transistors and later by integrated circuits, which led to the microprocessors we use today.

Important Questions:-

Short Questions:

1. What is shift registers, explain?
2. What is meant by Multiplexer?
3. Explain types of Bus Structures?
4. Explain differences between multi-processor and multi-computers?

Long Questions:

1. What is registers and shift registers?
2. Explain about Counters, what is meant by decoders?

3. What is meant by PLD?
4. Describe about basic operational concepts with historical perspectives in computer organization?

UNIT- II

Addressing Methods and Machine Program Sequencing:

Addressing Methods-

The addressing modes help us specify the way in which an operand's effective address is represented in any given instruction. Some addressing modes allow referring to a large range of areas efficiently, like some linear array of addresses along with a list of addresses. The addressing modes describe an efficient and flexible way to define complex effective addresses.

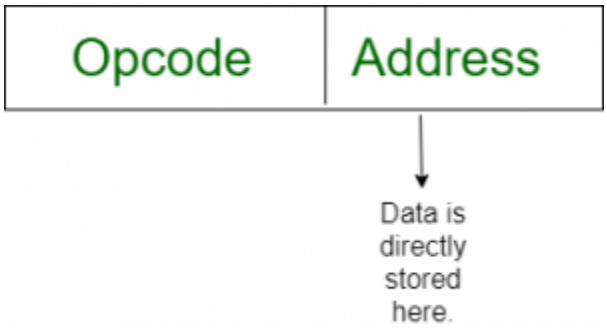
The programs are generally written in high-level languages, as it's a convenient way in which one can define the variables along with the operations that a programmer performs on the variables.

- Implied Mode.
- Immediate Mode.
- Register Mode.
- Register Indirect Mode.
- Autodecrement Mode.
- Autoincrement Mode.
- Direct Address Mode.
- Indirect Address Mode.

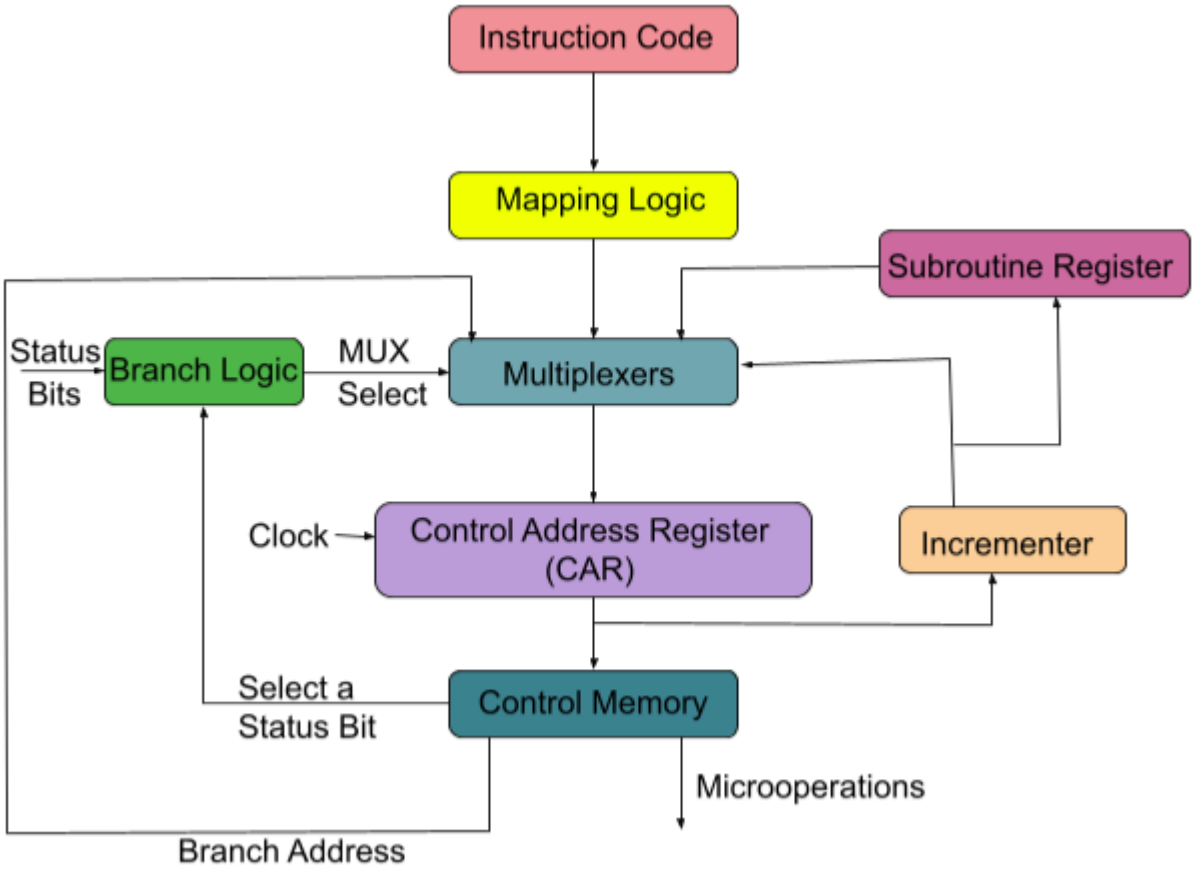
Immediate addressing mode (symbol #): In this mode data is present in address field of instruction .Designed like one address instruction format.

Note: Limitation in the immediate mode is that the range of constants are restricted

by size of address field.



Machine Program Sequencing-



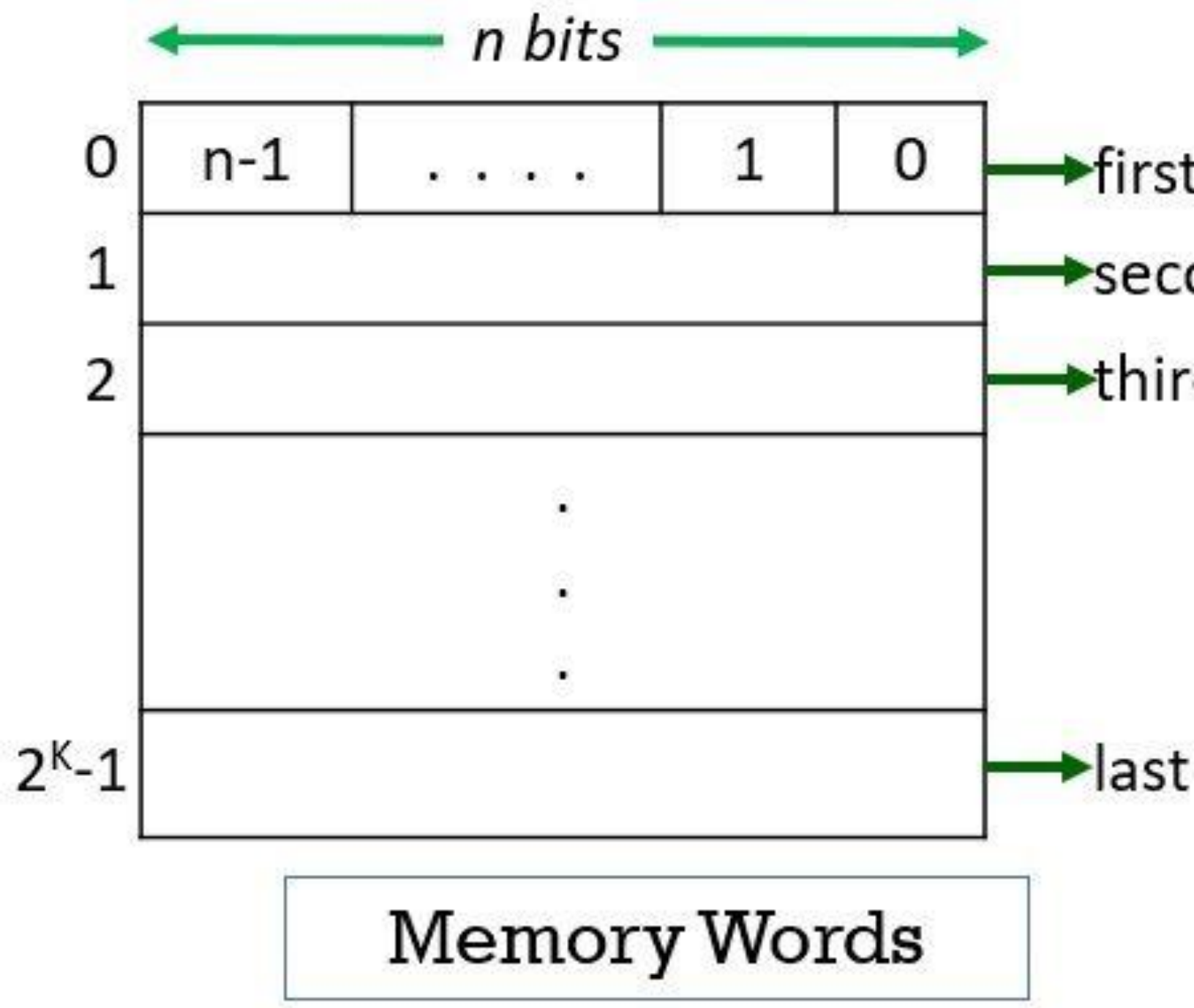
Machine Instructions are commands or programs written in machine code of a machine (computer) that it can recognize and execute.

- A machine instruction consists of several bytes in memory that tells the processor to perform one machine operation.
- The processor looks at machine instructions in main memory one after another, and performs one machine operation for each machine instruction.
- The collection of machine instructions in main memory is called a **machine language program.**

Memory Locations and Address:

Memory locations and addresses determine how the computer's memory is organized so that the user can efficiently store or retrieve information from the computer. The computer's memory is made of a silicon chip which has millions of storage cell, where each storage cell is capable to store a *bit* of information which value is either 0 or 1.

The group of n bit is termed as ***word*** where n is termed as the *word length*. The word length of the computer has evolved from 8, 16, 24, 32 to 64 bits. General-purpose computers nowadays have 32 to 64 bits. The group of 8 bit is called a *byte*.


n bits
0
n-1
. . . .
1
0
first
1
2
2K-1
last
Memory Words

Memory Organisation in Computer Architecture

The memory is organized in the form of a cell, each cell is able to be identified with a unique number called address. Each cell is able to recognize control signals such as “read” and “write”, generated by CPU when it wants to read or write address. Whenever CPU executes the program there is a need to transfer the instruction from the memory to CPU because the program is available in memory. To access the instruction CPU generates the memory request.

Memory Request:

Memory request contains the address along with the control signals. For Example, When inserting data into the stack, each block consumes memory (RAM) and the number of memory cells can be determined by the capacity of a memory chip.

Word Size:

It is the maximum number of bits that a CPU can process at a time and it depends upon the processor. Word size is a fixed size piece of data handled as a unit by the instruction set or the hardware of a processor.

Main memory Operations:

Both program instructions and data operands are stored in the memory. To execute an instruction, the processor control circuits must cause the word (or words) containing the instruction to be transferred from the memory to the processor. Operands and results must also be moved between the memory and the processor. Thus, two basic operations involving the memory are needed, namely, Read and Write.

• Two memory operations are:

1) Load (Read/Fetch) &

2) Store (Write).

Steps for Load operation:

1) Processor sends the address of the desired location to the memory.

2) Processor issues „read“ signal to memory to fetch the data.

3) Memory reads the data stored at that address. 4) Memory sends the read data to the processor.

• The Store operation transfers the information from the register to the specified memory-location. This will destroy the original contents of that memory-location.

• **Steps for Store operation are:**

1) Processor sends the address of the memory-location where it wants to store data.

2) Processor issues „write" signal to memory to store the data.

3) Content of register(MDR) is written into the specified memory-location.

RAM is the main memory of a computer. Its objective is to store data and applications that are currently in use. The operating system controls the usage of this memory. It gives instructions like when the items are to be loaded into RAM, where they are to be located in RAM, and when they need to be removed from RAM.

Instructions and Instruction Sequencing:

The tasks carried out by a computer program consist of a sequence of small steps, suchas adding two numbers, testing for a particular condition, reading a character from the keyboard, or sending a character to be displayed on a display screen.

A computer must have instructions capable of performing 4 types of operations:

1) Data transfers between the memory and the registers (MOV, PUSH, POP, XCHG).
2) Arithmetic and logic operations on data (ADD, SUB, MUL, DIV, AND, OR, NOT).
3) Program sequencing and control(CALL.RET, LOOP, INT).
4) I/o transfers (IN, OUT).

instruction sequencing The order in which the instructions in a program are carried out. Normally the sequence proceeds in a linear fashion through the program, and the address of the instructions is obtained from the program counter in the control unit. This sequence is interrupted when a branch instruction is executed; at such a time the address field of the branch instruction is inserted into the program counter and the process continues.

The processor control circuits use information in PC to fetch & execute instructions one at a time in order of increasing address.

INSTRUCTIONS AND INSTRUCTION SEQUENCING

1. Data transfer between memory and processor registers.
2. Arithmetic & logic operations on data.
3. Program sequencing & control.
4. I/O transfers.

Instruction Type	Syntax	Example	Description	Instructions for Operation C<-[A]+[B]
Three Address	Opcode Source1,Source2,Destination	Add A,B,C	Add the contents of memory-locations A & B. Then, place the result into location C.	
Two Address	Opcode Source, Destination	Add A,B	Add the contents of memory-locations A & B. Then, place the result into location B, replacing the original contents of this location. Operand B is both a source and a destination.	Move B, C Add A, C
One Address	Opcode Source/Destination	Load A	Copy contents of memory-location A into accumulator.	Load A Add B Store C
		Add B	Add contents of memory-location B to contents of accumulator register & place sum back into accumulator.	
		Store C	Copy the contents of the accumulator into location C.	
Zero Address	Opcode [no Source/Destination]	Push	Locations of all operands are defined implicitly. The operands are stored in a pushdown stack.	Not possible

Addressing Modes:

Addressing Modes– The term addressing modes refers to the way in which the operand of an instruction is specified. The addressing mode specifies a rule for interpreting or modifying the address field of the instruction before the operand is actually executed.

Addressing modes for 8086 instructions are divided into two categories:

1) Addressing modes for data

2) Addressing modes for branch

The different modes/methods that are used for specifying the address of the operand in the given instructions are called the addressing modes.

...

Addressing Modes Types

- Implied Mode.
- Immediate Mode.
- Register Mode.
- Register Indirect Mode.
- Autodecrement Mode.

- Autoincrement Mode.
- Direct Address Mode.
- Indirect Address Mode.

Types of Addressing Modes

- Implied Mode − In this mode, the operands are specified implicitly in the definition of the instruction. ...
- Instruction format with mode field.
- Immediate Mode − In this mode, the operand is specified in the instruction itself.

Important Questions:

Short Questions:

1. What is machine program sequencing?
2. What are addressing methods?
3. Explain types of main memory operations?
4. What is meant by addressing mode?

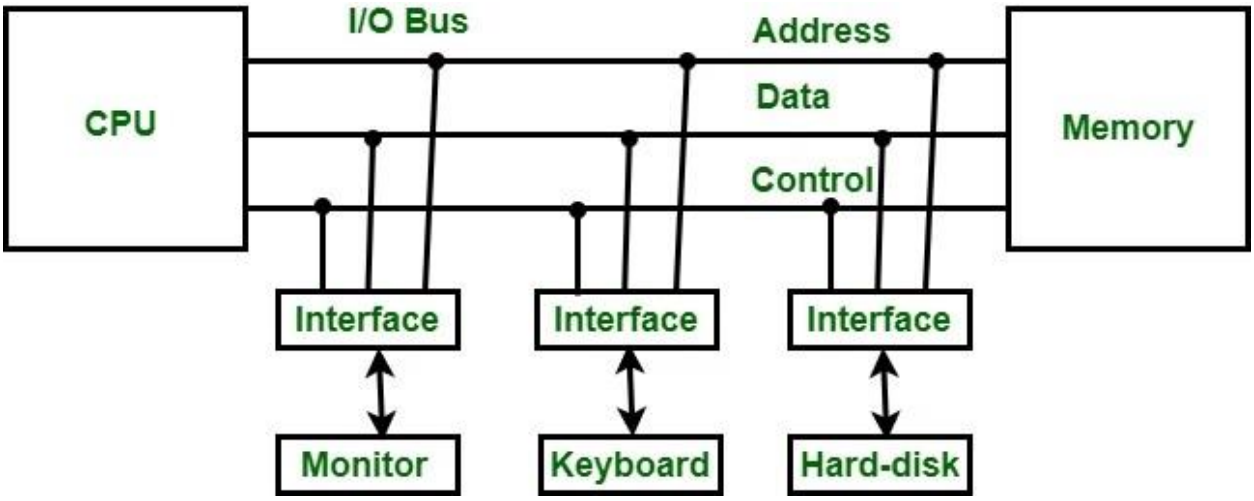
Long Questions:

1. What are the different types of addressing methods and machine program sequencing?
2. Describe instructions and instructions sequencing?
3. What are the main memory operations and addressing modes types?

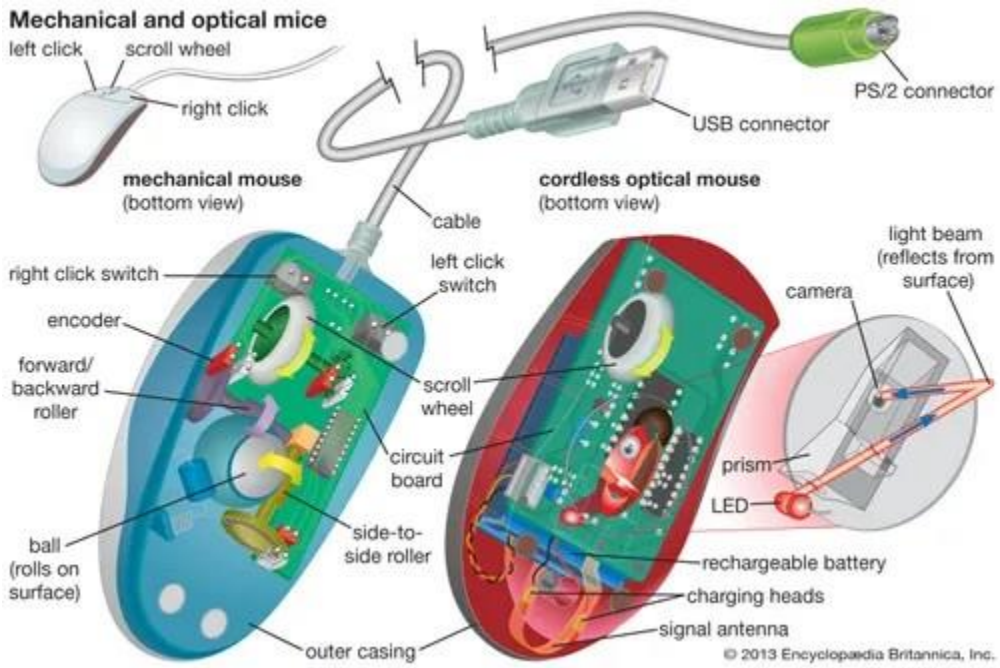
UNIT-III

Input/Output Organization (I/O):

An I/O interface is used as an method which helps in transferring of information between the internal storage devices i.e. memory and the external peripheral device . A peripheral device is that which provide input and output for the computer, it is also called Input-Output devices. For Example: A keyboard and mouse provide Input to the computer are called input devices while a monitor and printer that provide output to the computer are called output devices. Just like the external hard-drives, there is also availability of some peripheral devices which are able to provide both input and output.



Peripheral device, also known as **peripheral, computer peripheral,input-output device,** or **input/output device**, any of various devices (including sensors) used to enter information and instructions into a computer for storage or processing and to deliver the processed data to a human operator or, in some cases, a machine controlled by the computer. Such devices make up the peripheral equipment of modern digital computer systems.

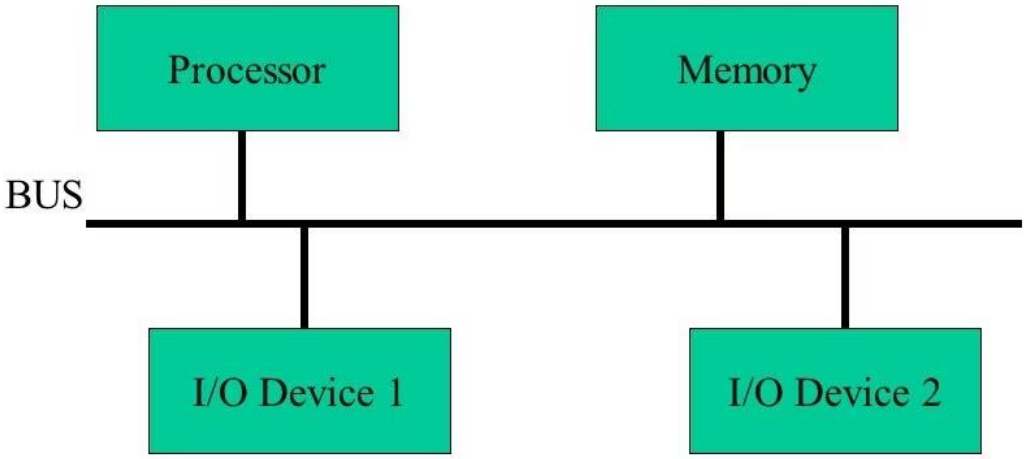


Accessing I/O Devices:

Accessing I/O Devices.: In computing, input/output, or I/O, refers to **the communication between an information processing system (computer), and the outside world**. Inputs are the signals or data received by the system, and outputs are the signals or data sent from it.

A simple arrangement to connect I/O devices to a computer is to **use a single bus arrangement**. The bus enables all the devices connected to it to exchange information. Typically, it consists of three sets of lines used to carry address, data, and control signals. Each I/O device is assigned a unique set of addresses.

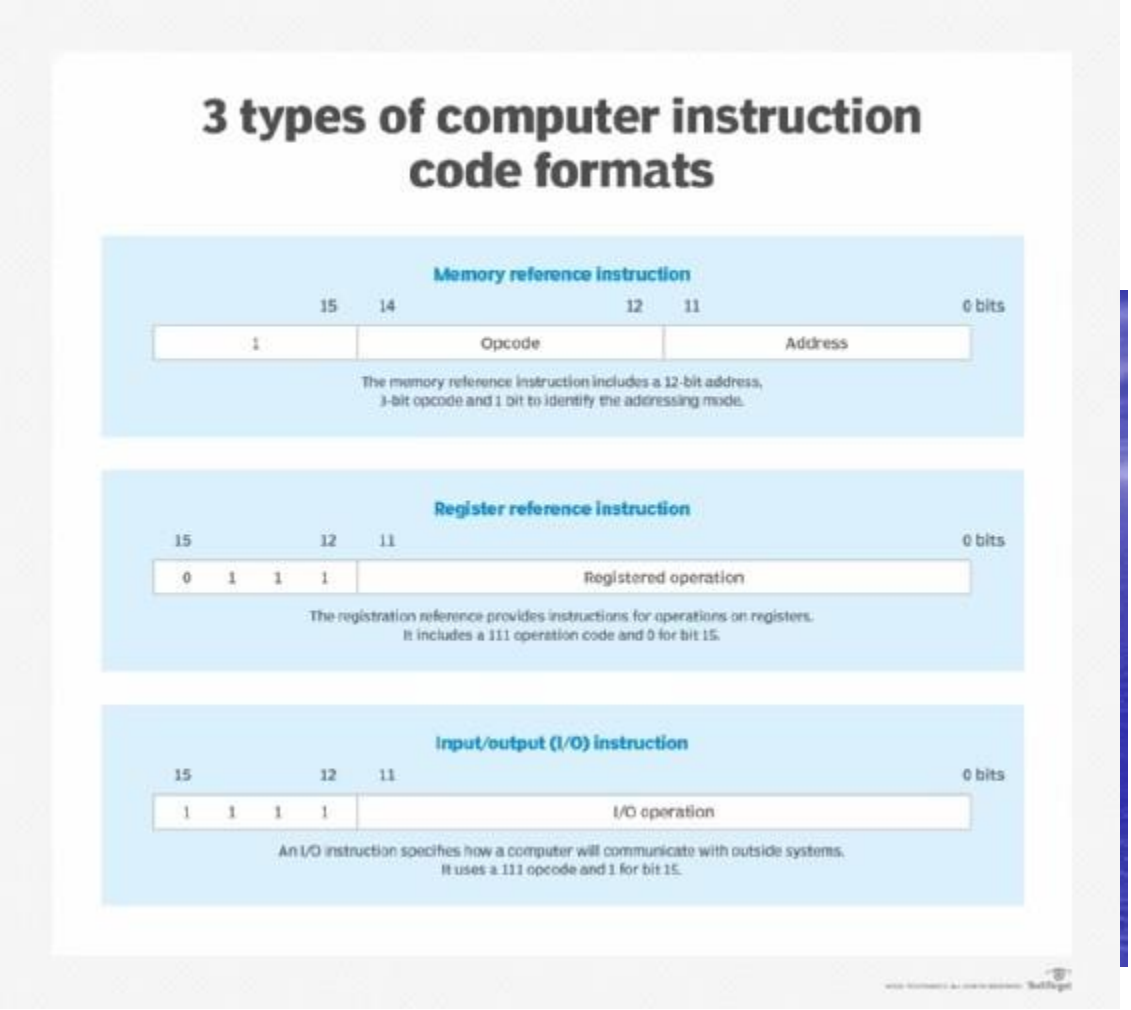
Accessing I/O Devices



Interrupts in Computer Organization:

An interrupt is a signal emitted by a device attached to a computer or from a program within the computer. It requires the operating system (OS) to stop and figure out what to do next. An interrupt temporarily stops or terminates a service or a current process. Most I/O devices have a bus control line called Interrupt Service Routine (ISR) for this purpose.

An interrupt signal might be planned (i.e., specifically requested by a program) or it may be unplanned (i.e., caused by an event that may not be related to a program that's currently running on the system).



I. Hardware interrupt

A hardware interrupt is an electronic signal from an external hardware device that indicates it needs attention from the OS. One example of this is moving a mouse or pressing a keyboard key. In these examples of interrupts, the processor must stop to read the mouse position or keystroke at that instant.

II. Software interrupts

A software interrupt occurs when an application program terminates or requests certain services from the OS. Usually, the processor requests a software interrupt when certain conditions are met by executing a special instruction. This instruction invokes the interrupt and functions like a subroutine call. Software interrupts are commonly used when the system interacts with device drivers or when a program requests OS services.

There are three types of hardware interrupts:

- Maskable interrupts. In a processor, an internal interrupt mask register selectively enables and disables hardware requests. ...
- Non-maskable interrupts. In some cases, the interrupt mask cannot be disabled so it does not affect some interrupt signals. ...
- Spurious interrupts.

Direct Memory Access (DMA) :

Direct memory access (DMA) is a method that allows an input/output (I/O) device to send or receive data directly to or from the main memory, bypassing the CPU to speed up memory operations.

The process is managed by a chip known as a DMA controller (DMAC).

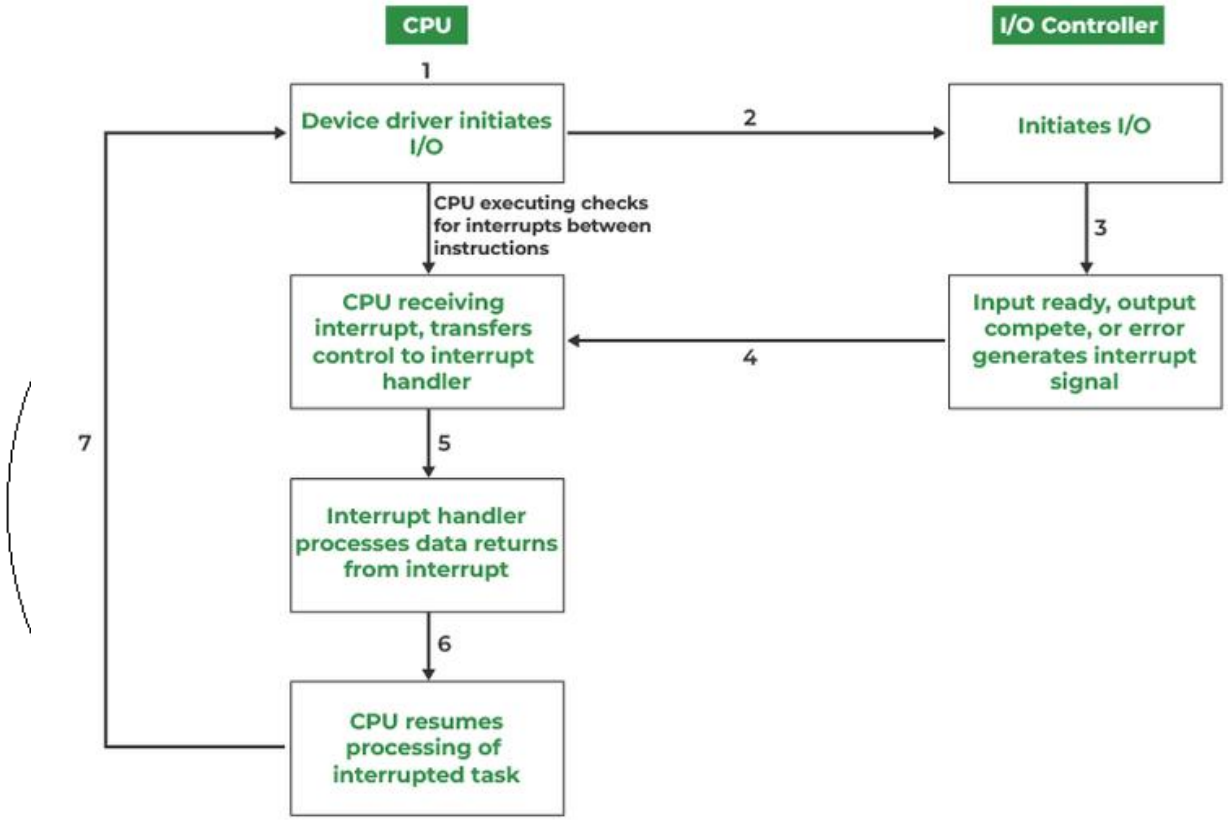
A computer's system resource tools are used for communication between hardware and software. The four types of system resources are:

- I/O addresses.
- Memory addresses.
- Interrupt request numbers (IRQ).
- Direct memory access (DMA) channels.
- **Direct memory access** (**DMA**) is a feature of computer systems that allows certain hardware subsystems to access main system memory independently of the central processing unit (CPU).
- Without DMA, when the CPU is using programmed input/output, it is typically fully occupied for the entire duration of the read or write operation, and is thus unavailable to perform other work. With DMA, the CPU first initiates the transfer, then it does other operations while the transfer is in progress, and it finally receives an interrupt from the DMA controller (DMAC) when the operation is done.

I/O Hardware:

In order to manage and control the various I/O device attached to computer, I/O system requires some hardware and software components. I/O devices commonly use certain hardware devices. **These are:** system bus and ports.

- **Ports** are the plugs used to connect I/O devices to the computer.
- **Bus** is a set of wires to which these ports and I/O controllers are connected and through which signals are send for 1/O command



I/O units (Keyboard, mouse, printer, etc.) typically consist of a mechanical component and an electronic component where electronic component is called the device controller.

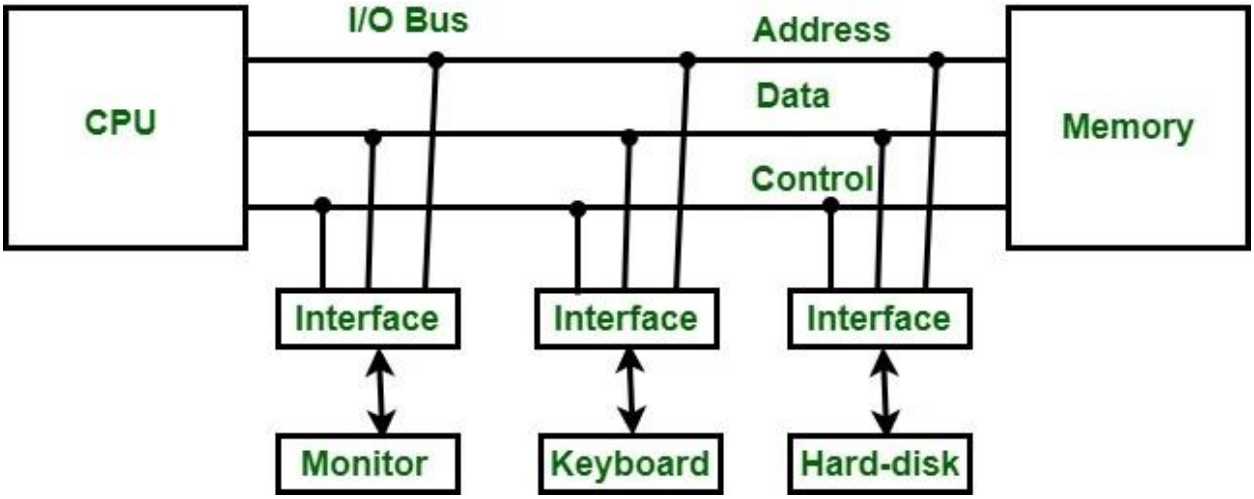
There is always a device controller and a device driver for each device to communicate with the Operating Systems.

/O Hardware is a set of specialized hardware devices that help the operating system access disk drives, printers, and other peripherals. These devices are located inside the motherboard and connected to the processor using a bus. They often have specialized controllers that allow them to quickly respond to requests from software running on top of them or even respond directly to commands from an application program. This post will discuss in detail I/O Hardware basics such as daisy chain expansion bus controller memory-mapped I/O Direct Memory Access (DMA)

Standard I/O Interface:

The I/O interface supports a method by which data is transferred between internal storage and external I/O devices. All the peripherals connected to a computer require special communication connections for interfacing them with the CPU

It is used as an method which helps in transferring of information between the internal storage devices i.e. memory and the external peripheral device . A peripheral device is that which provide input and output for the computer, it is also called Input-Output devices. For Example: A keyboard and mouse provide Input to the computer are called input devices while a monitor and printer that provide output to the computer are called output devices. Just like the external hard-drives, there is also availability of some peripheral devices which are able to provide both input and output.



An interface receives any of the following four commands −

- **Control** − A command control is given to activate the peripheral and to inform its next task. This control command depends on the peripheral, and each peripheral receives its sequence of control commands, depending on its mode of operation.
- **Status** − A status command can test multiple test conditions in the interface and the peripheral.
- **Data Output** − A data output command creates the interface counter to the command by sending data from the bus to one of its registers.
- **Data Input** − The data input command is opposite to the data output command. In data input, the interface gets an element of data from the peripheral and places it in its buffer register.

Important Questions:

Short Questions:

1. What is I/O Organization?
2. Define about Interrupts?
3. What is meant by DMA?

Long Questions:

1. Describe about Direct Memory Access?
2. Explain about I/O Hardware and explain about it's importance?
3. Explain Interrupts, explain about standard I/O interface?

UNIT – IV

Memory System Concepts:

A memory is just like a human brain. It is used to store data and instructions. Computer memory is the storage space in the computer, where data is to be processed and instructions required for processing are stored. The memory is divided into large number of small parts called cells. Each location or cell has a unique address, which varies from zero to memory size minus one

Cache Memory

Cache memory is a very high speed semiconductor memory which can speed up the CPU. It acts as a buffer between the CPU and the main memory. It is used to hold those parts of data and program which are most frequently used by the CPU. The parts of data and programs are transferred from the disk to cache memory by the operating system, from where the CPU can access them.

Primary Memory (Main Memory)

Primary memory holds only those data and instructions on which the computer is currently working. It has a limited capacity and data is lost when power is switched off. It is generally made up of semiconductor device. These memories are not as fast as registers. The data and

instruction required to be processed resides in the main memory. It is divided into two subcategories RAM and ROM.

Secondary Memory

This type of memory is also known as external memory or non-volatile. It is slower than the main memory. These are used for storing data/information permanently. CPU directly does not access these memories, instead they are accessed via input-output routines. The contents of secondary memories are first transferred to the main memory, and then the CPU can access it. For example, disk, CD-ROM, DVD, etc.

Semi-conductor RAM memories:

Random Access Memory is **a form of semiconductor memory technology that is applied for reading and writing data in any order**. It is used for such purposes as the computer or processor memory where variables and others are stored and are needed on a random basis.

Semiconductor memory is a type of semiconductor device tasked with storing data. There are two electronic data storage mediums that we can utilize, magnetic or optical.

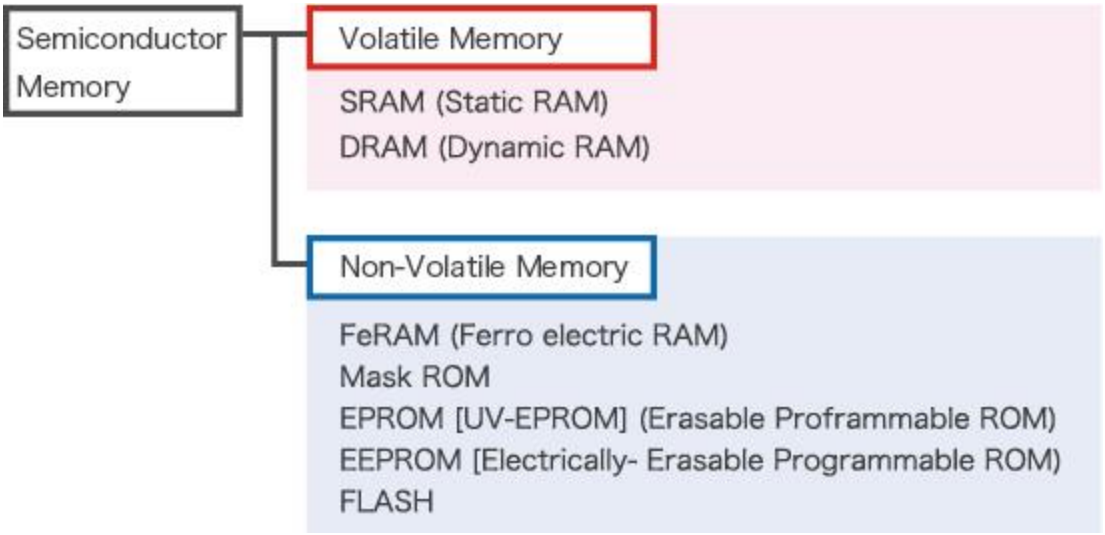
Magnetic storage:

- Stores data in magnetic form.
- Affected by magnetic fields.
- Has high storage capacity.
- Doesn't use a laser to read/write data.
- Magnetic storage devices are; Hard disk , Floppy disk, Magnetic tape etc.

Optical storage:

- Stores data optically, uses laser to read/write.
- Not affected by magnetic fields.
- Has less storage than a hard disk.
- Data accessing is high, compared to a floppy disc.
- Optical storage devices are; CD-ROM,CD-R, CD-RW, DVD etc.

Semiconductor Memory Types



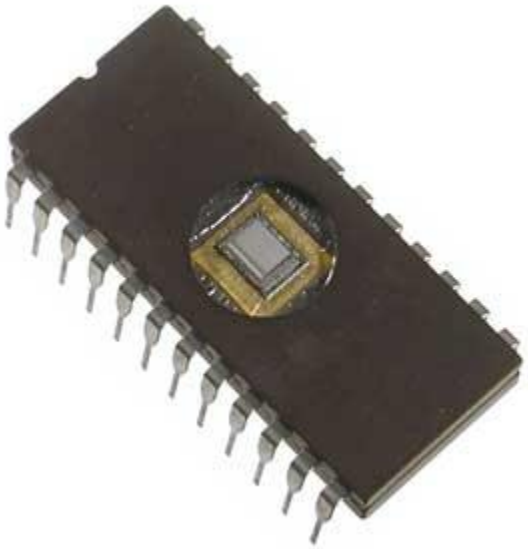
* **RAM** (Random Access Memory) : Enables Read/Write of stored contents

* **ROM** (Read Only Memory) : Allows only Read operation.

Read Only Memories (ROM):

Read-only memory, or ROM, is a type of computer storage containing non-volatile, permanent data that, normally, can only be read, not written to. ROM contains the programming that allows a computer to start up or regenerate each time it is turned on. ROM also performs large input/output (I/O) tasks and protects programs or software instructions. Once data is written on a ROM chip, it cannot be removed.

ROM stands for **Read Only Memory**. The memory from which we can only read but cannot write on it. This type of memory is non-volatile. The information is stored permanently in such memories during manufacture. A ROM stores such instructions that are required to start a computer. This operation is referred to as **bootstrap**. ROM chips are not only used in the computer but also in other electronic items like washing machine and microwave oven.



MROM (Masked ROM)

The very first ROMs were hard-wired devices that contained a pre-programmed set of data or instructions. These kind of ROMs are known as masked ROMs, which are inexpensive.

PROM (Programmable Read Only Memory)

PROM is read-only memory that can be modified only once by a user. The user buys a blank PROM and enters the desired contents using a PROM program.

EPROM (Erasable and Programmable Read Only Memory)

EPROM can be erased by exposing it to ultra-violet light for a duration of up to 40 minutes. Usually, an EPROM eraser achieves this function. During programming, an electrical charge is

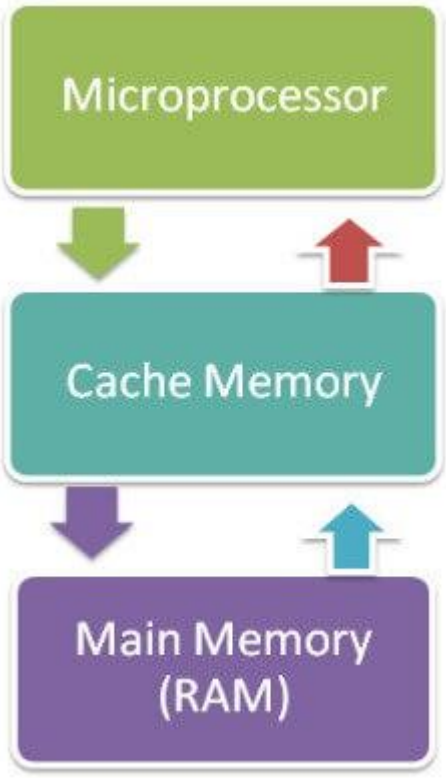
trapped in an insulated gate region. The charge is retained for more than 10 years because the charge has no leakage path.

EEPROM (Electrically Erasable and Programmable Read Only Memory)

EEPROM is programmed and erased electrically. It can be erased and reprogrammed about ten thousand times. Both erasing and programming take about 4 to 10 ms (millisecond). In EEPROM, any location can be selectively erased and programmed. EEPROMs can be erased one byte at a time, rather than erasing the entire chip. Hence, the process of reprogramming is flexible but slow.

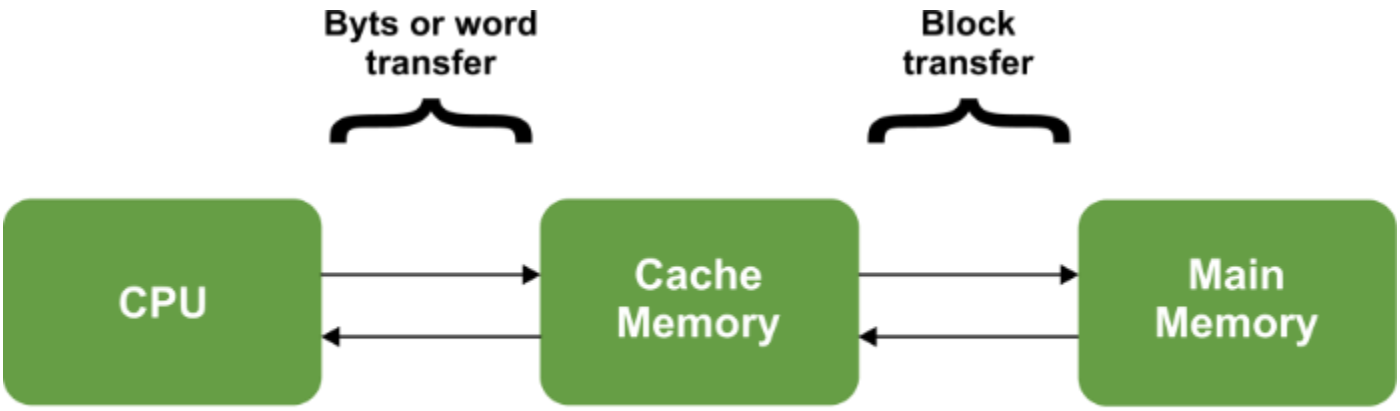
Cache Memories:

cache memory, also called **cache**, supplementary memory system that temporarily stores frequently used instructions and data for quicker processing by the central processing unit (CPU) of a computer.
The cache augments, and is an extension of, a computer's main memory. Both main memory and cache are internal random-access memories (RAMs) that use semiconductor-based transistor circuits. Cache holds a copy of only the most frequently used information or program codes stored in the main memory. The smaller capacity of the cache reduces the time required to locate data within it and provide it to the CPU for processing.
Cache memory is a high-speed memory, which is small in size but faster than the main memory (RAM). The CPU can access it more quickly than the primary memory. So, it is used to synchronize with high-speed CPU and to improve its performance.



The data or contents of the main memory that are used frequently by CPU are stored in the cache memory so that the processor can easily access that data in a shorter time. Whenever the CPU needs to access memory, it first checks the cache memory. If the data is not found in cache memory, then the CPU moves into the main memory.

Cache memory is placed between the CPU and the main memory. The block diagram for a cache memory can be represented as:



Cache Memory

The cache is the fastest component in the memory hierarchy and approaches the speed of CPU components.

Performance considerations:

Computer performance is the amount of work accomplished by a computer system. The word performance in computer performance means "How well is the computer doing the work it is supposed to do?". It basically depends on response time, throughput and execution time of a computer system.

Response time is the time from start to completion of a task. This also includes:

- Operating system overhead.
- Waiting for I/O and other processes
- Accessing disk and memory
- Time spent executing on the CPU or execution time.

- **Throughput** is the total amount of work done in a given time.
- **CPU execution time** is the total time a CPU spends computing on a given task. It also excludes time for I/O or running other programs. This is also referred to as simply CPU time.
- **Throughput** is the total amount of work done in a given time.
- **CPU execution time** is the total time a CPU spends computing on a given task. It also excludes time for I/O or running other programs. This is also referred to as simply CPU time.

Reporting performance is generally understood to be **the speed and efficiency with which reports are generated by the system when end-users perform a query**. Performance depends on a

variety of factors, including system bandwidth, number of concurrent users, volume of data to be presented, etc.

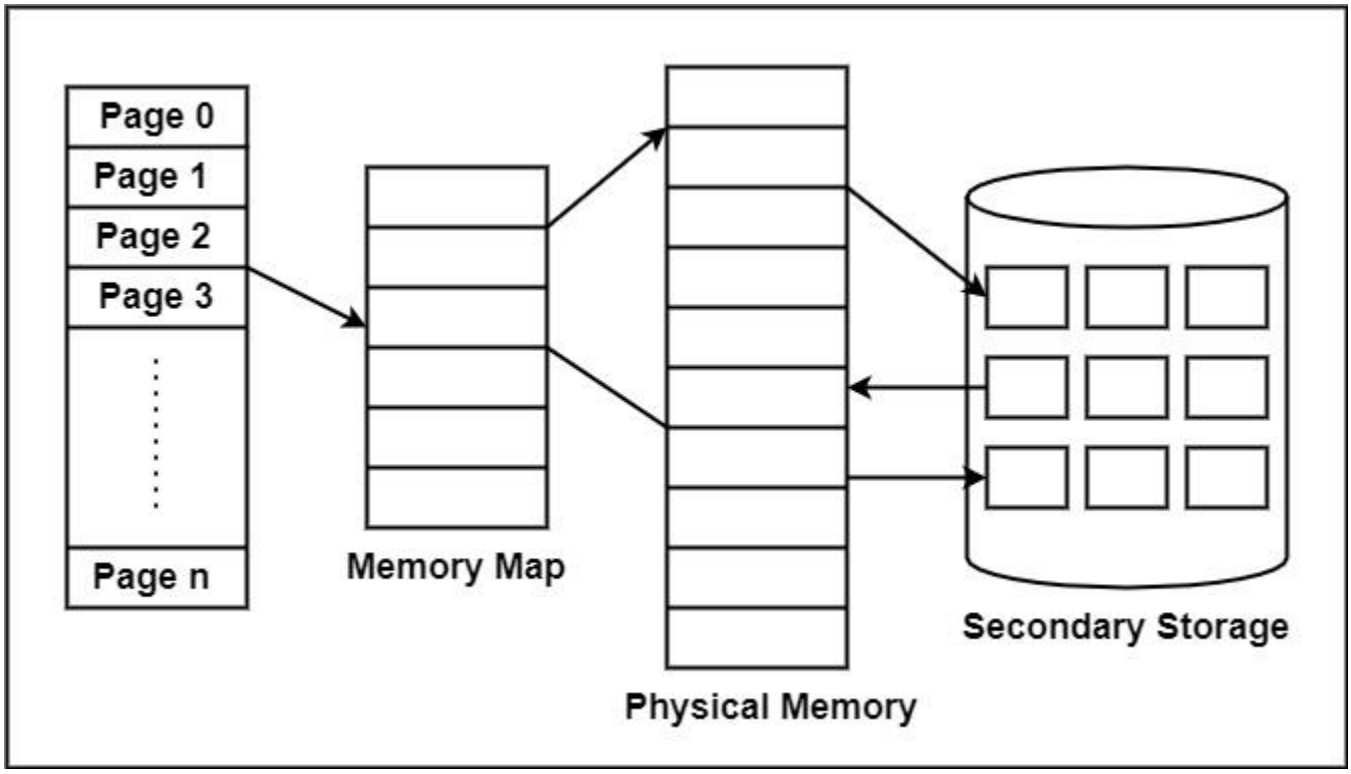
Virtual Memory:

Virtual memory is a memory management technique where secondary memory can be used as if it were a part of the main memory. Virtual memory is a common technique used in a computer's operating system (OS).

Virtual memory uses both hardware and software to enable a computer to compensate for physical memory shortages, temporarily transferring data from random access memory (RAM) to disk storage. Mapping chunks of memory to disk files enables a computer to treat secondary memory as though it were main memory

Virtual memory is the partition of logical memory from physical memory. This partition supports large virtual memory for programmers when only limited physical memory is available.

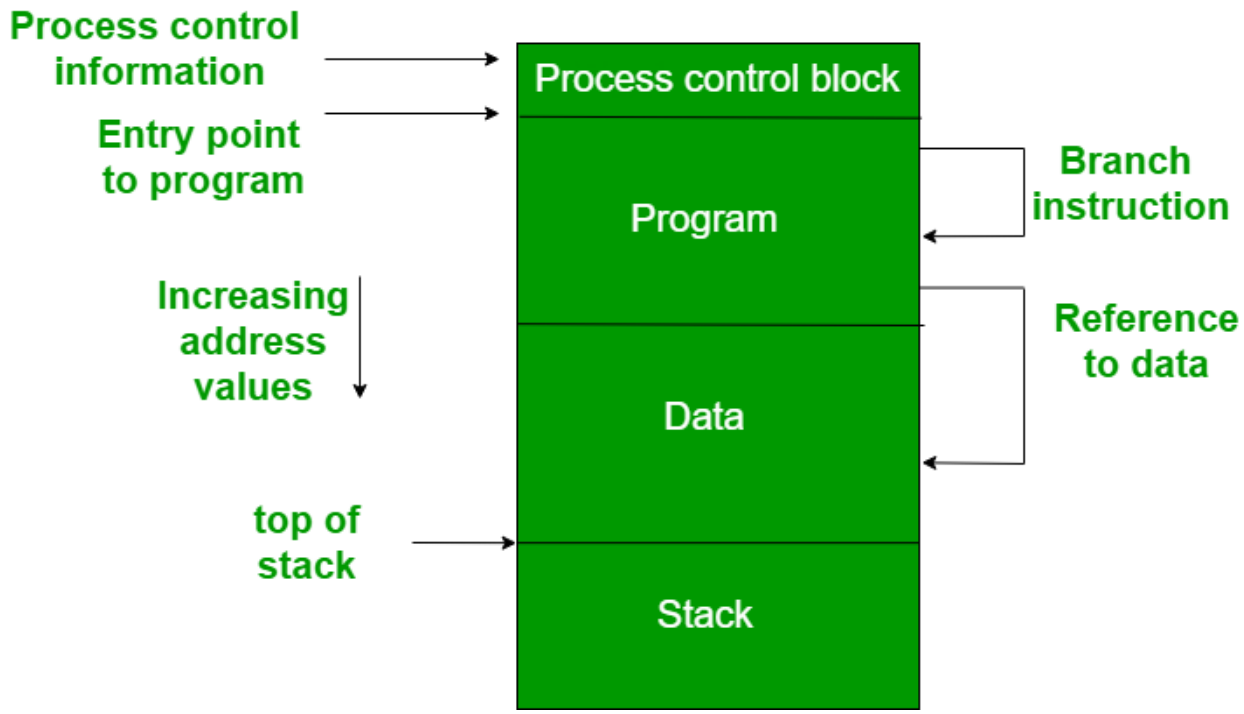
Virtual memory can give programmers the deception that they have a very high memory although the computer has a small main memory. It creates the function of programming easier because the programmer no longer requires to worry about the multiple physical memory available.



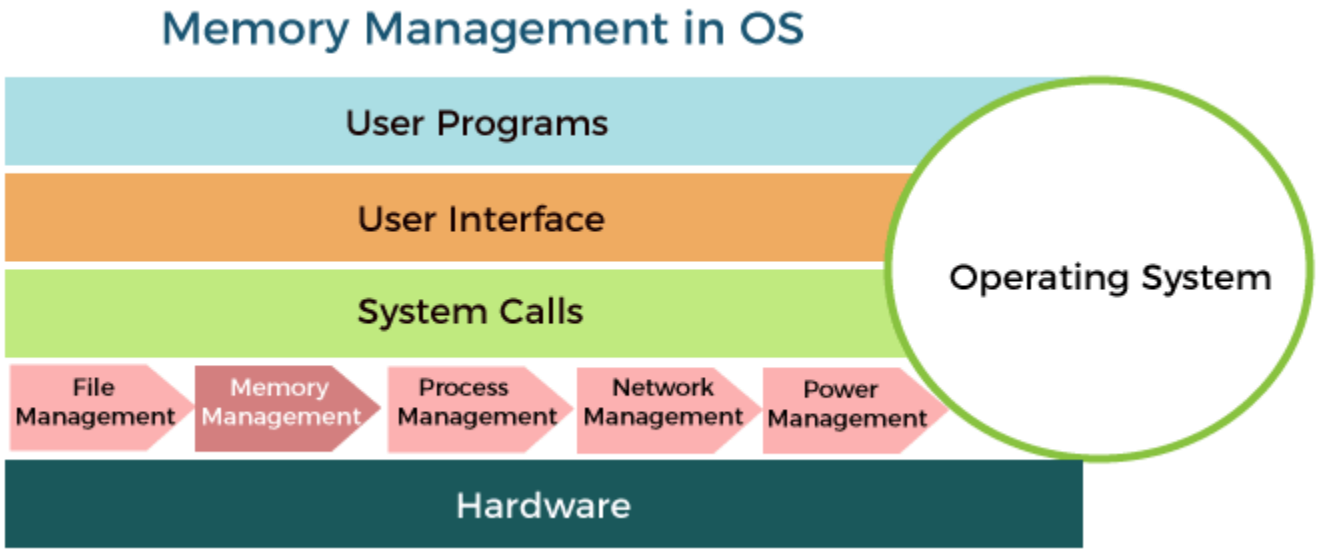
Memory Management Requirements:

Memory management keeps track of the status of each memory location, whether it is allocated or free. It allocates the memory dynamically to the programs at their request and frees it for reuse when it is no longer needed. Memory management meant to satisfy some requirements that we should keep in mind.

These Requirements of memory management are:



Memory is the important part of the computer that is used to store the data. Its management is critical to the computer system because the amount of main memory available in a computer system is very limited. At any time, many processes are competing for it. Moreover, to increase performance, several processes are executed simultaneously. For this, we must keep several processes in the main memory, so it is even more important to manage them effectively.



Addition and Substraction Of Sign Members:

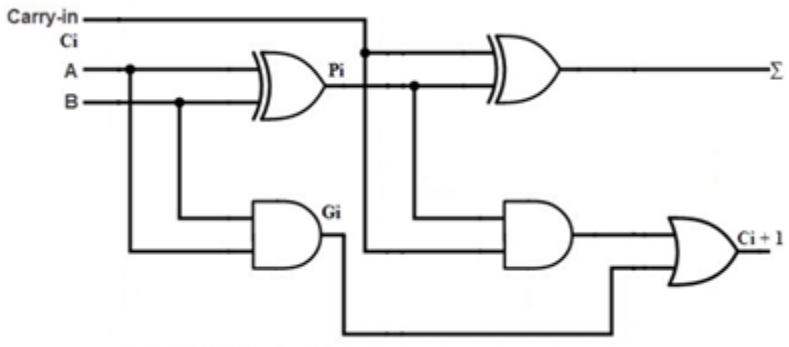
In a computer, the basic arithmetic operations are Addition and Subtraction. Multiplication and Division can always be managed with successive addition or subtraction respectively. However, hardware algorithms are implemented for Multiplication and Division.

It is to be recollected that computers deal with binary numbers unless special hardware is implemented for dealing with other number systems. Although instructions may be available for treating signed and unsigned operations, the programmer must deal with the numbers and handling of the result. The hardware assists the programmer by way of appropriate instructions and flags.

The hardware circuit which executes this addition is called **Adder**. There are two types of adders namely **Half adder** and **Full adder**. Basic adder circuit does 1-bit addition and is extended for n-bit addition. The adder circuit characteristics are detailed by a circuit, a truth table, Formula and a block symbol. The adder circuits are constructed from logic gates which satisfy the formula as per truth table. These are also called combinational logic. A Combinational logic output reflects the input without cclock

Design Of Fast Adders:

- carry-look ahead adder (CLA) or fast adder is a type of adder used in digital logic.
- A carry-look ahead adder improves speed by reducing the amount of time required to determine carry bits.
- It can be contrasted with the simpler, but usually slower, ripple-carry adder (RCA), for which the carry bit is calculated alongside the sum bit, and each stage must wait until the previous carry bit has been calculated to begin calculating its own sum bit and carry bit.
- The carry-lookahead adder calculates one or more carry bits before the sum, which reduces the wait time to calculate the result of the larger-value bits of the adder
- A carry-Lookahead adder is a fast parallel adder as it reduces the propagation delay by more complex hardware, hence it is costlier.
- This method makes use of logic gates so as to look at the lower order bits of the augend and addend to see whether a higher order carry is to be generated or not.



carry-lookahead adder (CLA) or fast adder is **a type of electronics adder used in digital logic**. A carry-lookahead adder improves speed by reducing the amount of time required to determine carry bits.

Carry Select Adder (CSA) is known to be the fastest adder among the conventional adder structures. It is used in many data processing units for realizing faster arithmetic operations. In this paper, we present an innovative CSA architecture. It employs a novel incrementer circuit in the interim stages of the CSA.

Multiplication of Positive Numbers:

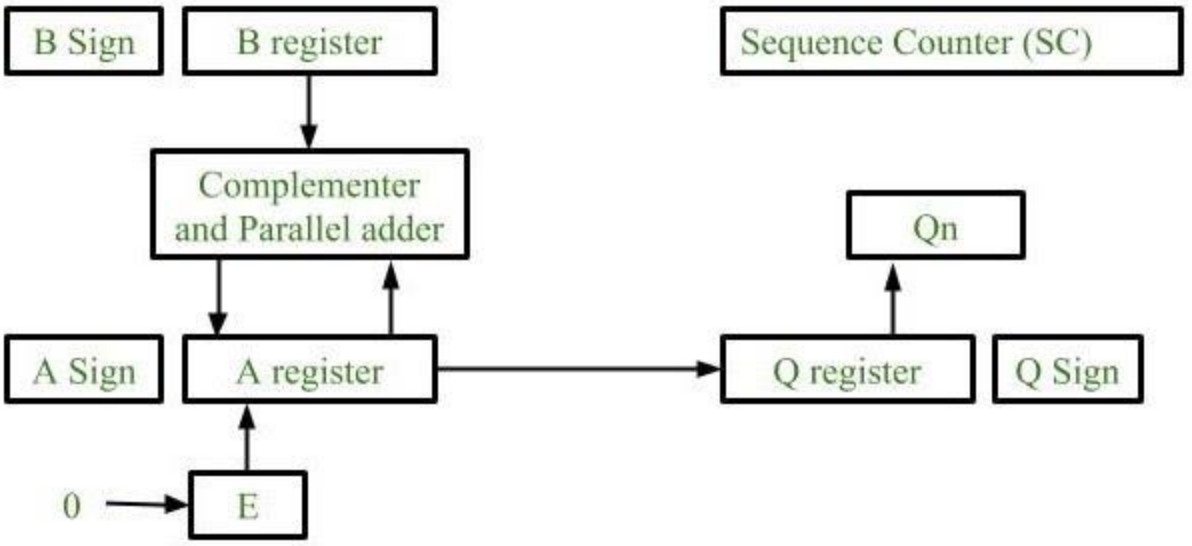
Multiplication Algorithm in Signed Magnitude Representation

Multiplication of two fixed point binary number in *signed magnitude representation* is done with process of *successive shift* and *add operation*.

```
   10111 (Multiplicand)
x  10011 (Multiplier)
  ------
   10111
   10111
  00000
 00000
10111
---------
011011010    (Product)
---------
```

In the multiplication process we are considering successive bits of the multiplier, least significant bit first.

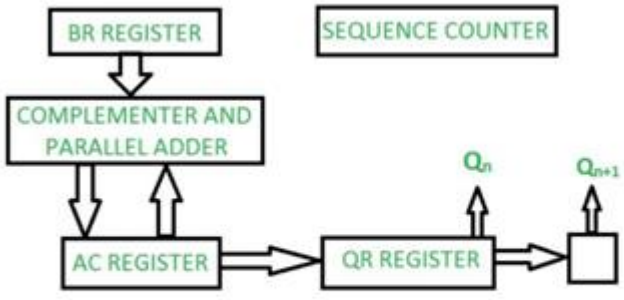
Hardware Implementation :
Following components are required for the *Hardware Implementation* of multiplication algorithm :



The booth algorithm is a multiplication algorithm that allows us to multiply the two signed binary integers in 2's complement, respectively. It is also used to speed up the performance of the multiplication process. It is very efficient too. It works on the string bits 0's in the multiplier that requires no additional bit only shift the right-most string bits and a string of 1's in a multiplier bit weight 2^k to weight 2^m that can be considered as $\mathbf{2^{k+1} - 2^m}$.

Signed Operand Multiplication:

It is **a multiplication algorithm that multiplies two signed binary numbers in two's complement notation**. It generates a 2n bit product for two n bit signed numbers. Flowchart of Booth's ALgorithm: Multiplying (-6) and (2) using Booth's Algorithm: (6)10=(0110)2.



Fast Multiplication:

Booth's algorithm is a multiplication algorithm that multiplies two signed binary numbers in 2's complement notation. Booth used desk calculators that were faster at shifting than adding and created the algorithm to increase their speed. Booth's algorithm is of interest in the study of computer architecture.

The performance of a fast multiplier depends on **generating fewer partial products**. The lower number of partial products means high speed of multiplication. There are two methods to reduce the number of partial products which are. Booth's multiplication algorithm. Implementing larger multiplier by smaller ones

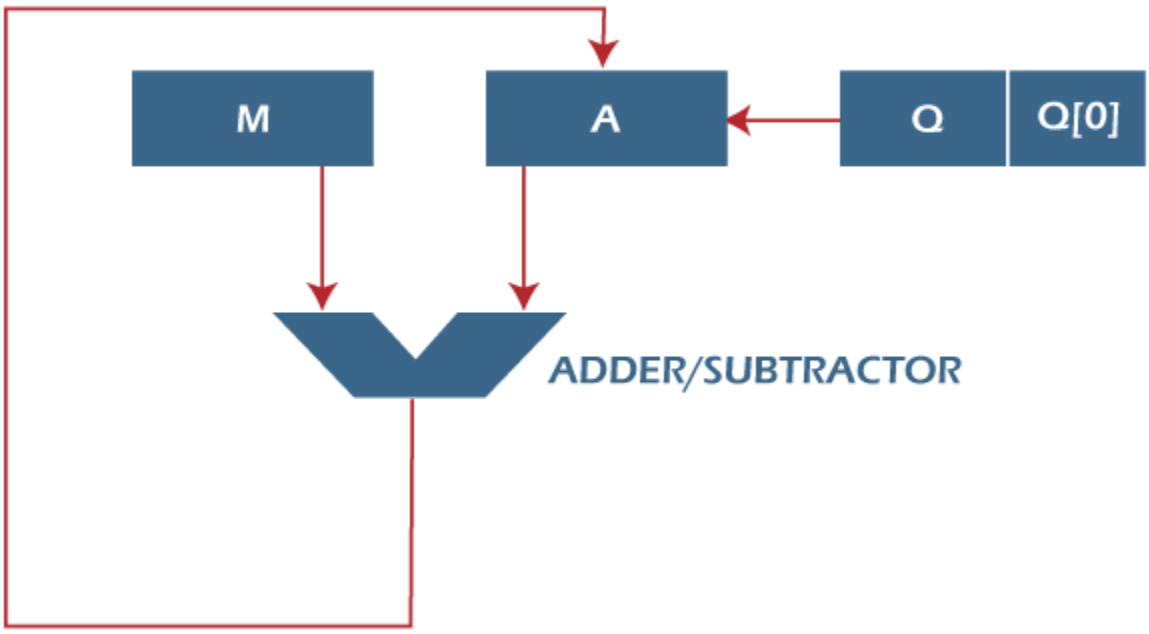
The **Karatsuba algorithm** is a fast multiplication algorithm that uses a divide and conquer approach to multiply two numbers. It was discovered by Anatoly Karatsuba in 1960 and published in 1962.

Integer Division:

The **% (integer divide) operator** divides two numbers and returns the integer part of the result. The result returned is defined to be that which would result from repeatedly subtracting the divisor from the dividend while the dividend is larger than the divisor. During this subtraction, the absolute values of both the dividend and the divisor are used: the sign of the final result is the same as that which would result from regular division.

The result returned has no fractional part (that is, no decimal point or zeros following it). If the result cannot be expressed as a whole number, the operation is in error and will fail—that is, the result must not have more digits than the current setting of NUMERIC DIGITS.

Compared to other arithmetic operations, division works very poorly on x86 and computers in general. Both floating-point and integer division is notoriously hard to implement in hardware. The circuitry takes a lot of space in the ALU, the computation has a lot of stages, and as the result, `div` and its siblings routinely take 10-20 cycles to complete, with latency being slightly less on smaller data type sizes.



Floating Point Numbers and Operations:

Arithmetic operations on floating point numbers consist of **addition, subtraction, multiplication and division**. The operations are done with algorithms similar to those used on sign magnitude integers (because of the similarity of representation) — example, only add numbers of the same sign.

The floating-point representation can implement operations for high range values. The numerical evaluations are carried out using floating-point values. It can create calculations easy, scientific numbers are described as follows −

The number 5,600,000 can be described as 0.56 * 107.

Therefore, 0.56 is the mantissa and 7 is the value of the exponent.

floating-point (FP) number is a kind of fraction where the radix point is allowed to move. If the radix point is fixed, then those fractional numbers are called fixed-point numbers. The best example of fixed-point numbers are those represented in commerce, finance while that of floating-point is the scientific constants and values.

FLOATING POINT OPERATIONS

The scientific notation has a single digit to the left of the decimal point. A number in scientific notation that has no leading 0s is called a normalized number, which is the usual way to write it. Foating point - Computer arithmetic that represents numbers in which the binary point is not fixed. Floating-point numbers are usually a multiple of the size of a word.

The representation of a MIPS floating-point number is shown below, where s is the sign of the floating-point number (1 meaning negative), exponent is the value of the 8-bit exponent field (including the sign of the exponent), and fraction is the 23-bit number. This representation is called sign and magnitude, since the sign has a separate bit from the rest of the number.

Floating Point Representation – Basics

There are posts on representation of floating point format. The objective of this article is to provide a brief introduction to floating point format.

The following description explains terminology and primary details of IEEE 754 binary floating-point representation. The discussion confines to single and double precision formats.

Usually, a real number in binary will be represented in the following format,

$$I_m I_{m-1} \ldots I_2 I_1 I_0 . F_1 F_2 \ldots F_n F_{n-1}$$

Important Questions:

Short Questions:

1. What is semi conductor RAM memories?
2. Explain ROM?
3. Define Cache Memory?
4. Describe what is meant by Design Of Fast Adders?
5. What are floating point numbers?

Long Questions:

1. Describe performance considerations with cache memory?
2. What are the different types of memory system concepts in computer organization?
3. Difference between cache and virtual memories?
4. Define memory management requirements, what are the multiplication of positive numbers?

UNIT-V

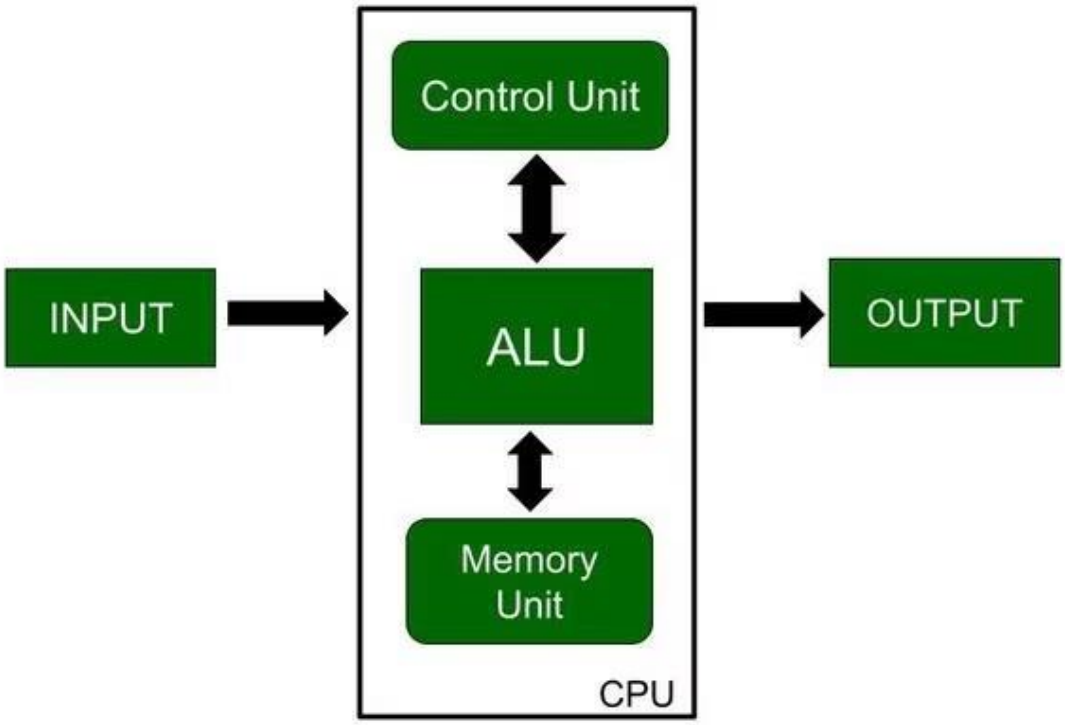
Basic Processing Units:

Concepts:

CPU is the brain of the computer. All types of data processing operations and all the important functions of a computer are performed by the CPU. It helps input and output devices to communicate with each other and perform their respective operations. It also stores data which is input, intermediate results in between processing, and instructions.

Now, the CPU consists of 3 major units, which are:

1. Memory or Storage Unit
2. Control Unit
3. ALU(Arithmetic Logic Unit)



- The processor fetches one instruction at a time and performs the operation specified.
- Instructions are fetched from successive memory locations until a branch or a jump instruction is encountered.λ

- The processor keeps track of the address of the memory location containing the next instruction to be fetched using Program Counter (PC).
- Instruction Register (IR)

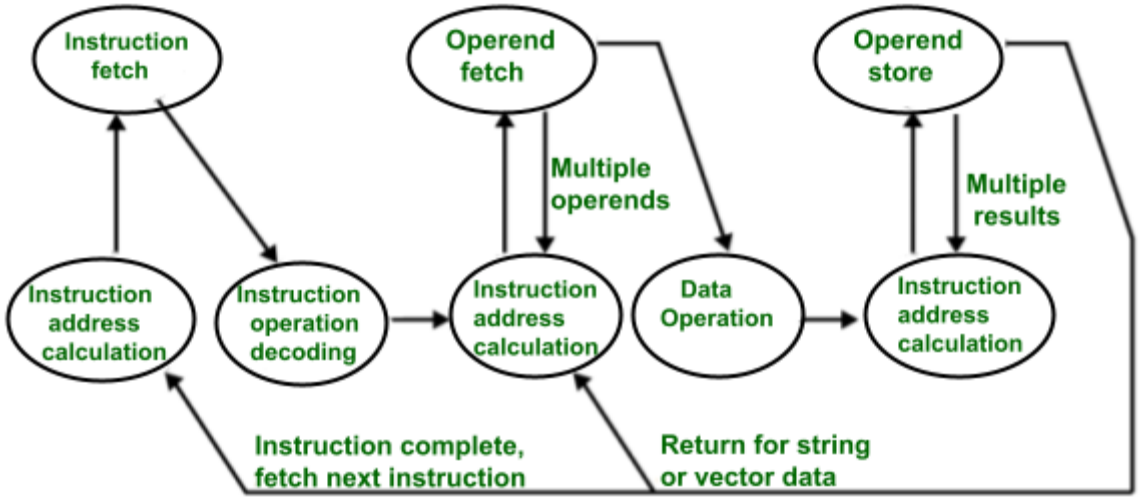
Execution of complete Instructions:

1. Fetch information from memory to CPU.
2. Store information to CPU register to memory.
3. Transfer of data between CPU registers.
4. Perform arithmetic or logic operation and store the result in CPU registers.

Internal steps for executing an instruction

- Five-stage pipeline
- (non-accumulator
- based-processor)
- Fetch the instruction.
- Decode the instruction and get the source operand.
- Get the destination operand.
- Perform the operation.

As instructions are a part of the program which are stored inside the memory, so every time the processor requires to execute an instruction, for that the processor first fetches the instruction from the memory, then decodes the instruction and then executes the instruction. The whole process is known as an instruction cycle.



Instruction execution :
Instruction execution needs the following steps, which are

- PC (program counter) register of the processor gives the address of the instruction which needs to be fetched from the memory.
- If the instruction is fetched then, the instruction opcode is decoded. On decoding, the processor identifies the number of operands. If there is any operand to be fetched from the memory, then that operand address is calculated.
- Operands are fetched from the memory. If there is more than one operand, then the operand fetching process may be repeated (i.e. address calculation and fetching operands).
- After this, the data operation is performed on the operands, and a result is generated.
- If the result has to be stored in a register, the instructions end here.
- If the destination is memory, then first the destination address has to be calculated. Then the result is then stored in the memory. If there are multiple results which need to be stored inside the memory, then this process may repeat (i.e. destination address calculation and store result).
- Now the current instructions have been executed. Side by side, the PC is incremented to calculate the address of the next instruction
- The above instruction cycle then repeats for further instructions

Multiple Bus Organization:

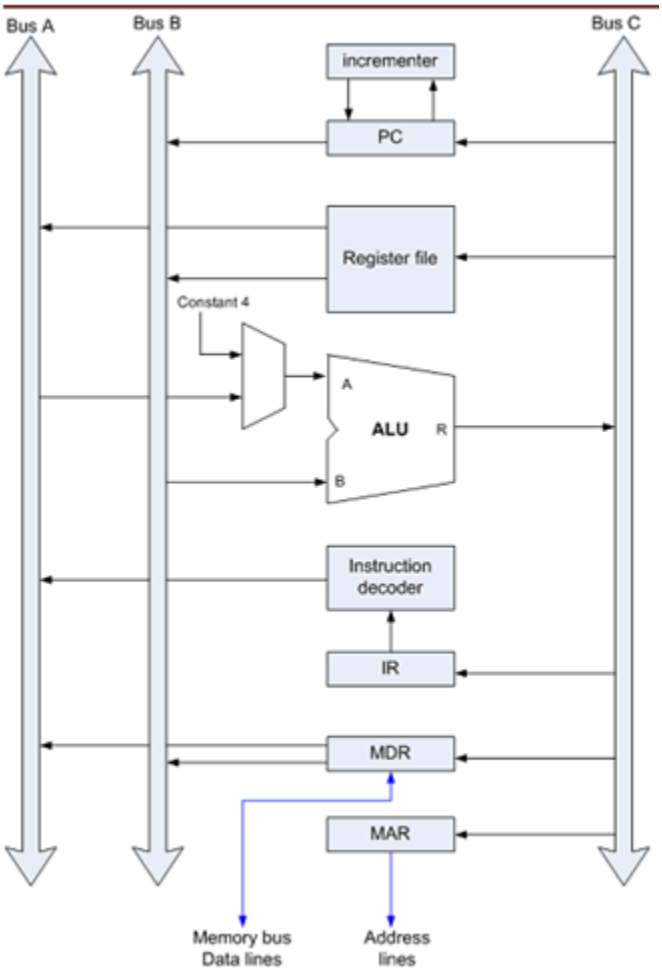
Notes for Multiple Bus Organization: This is the best article to access all the lecture notes and study materials for Multiple Bus Organization topics.

The topic is an important part of the subject of Computer Organization and architecture. To understand it well, a student must also properly understand all the concepts of the main subject.

Multiple Bus Organization also has many purposes in the global industries. This article will provide the students with everything that will briefly discuss their various uses and applications. For the future life of the students too, this article will be immensely beneficial.

In single bus organization, only one data item can be transferred over the bus in a clock cycle. To reduce the number of steps needed, most commercial processors provide multiple internal paths that enable several transfers to take place in parallel.

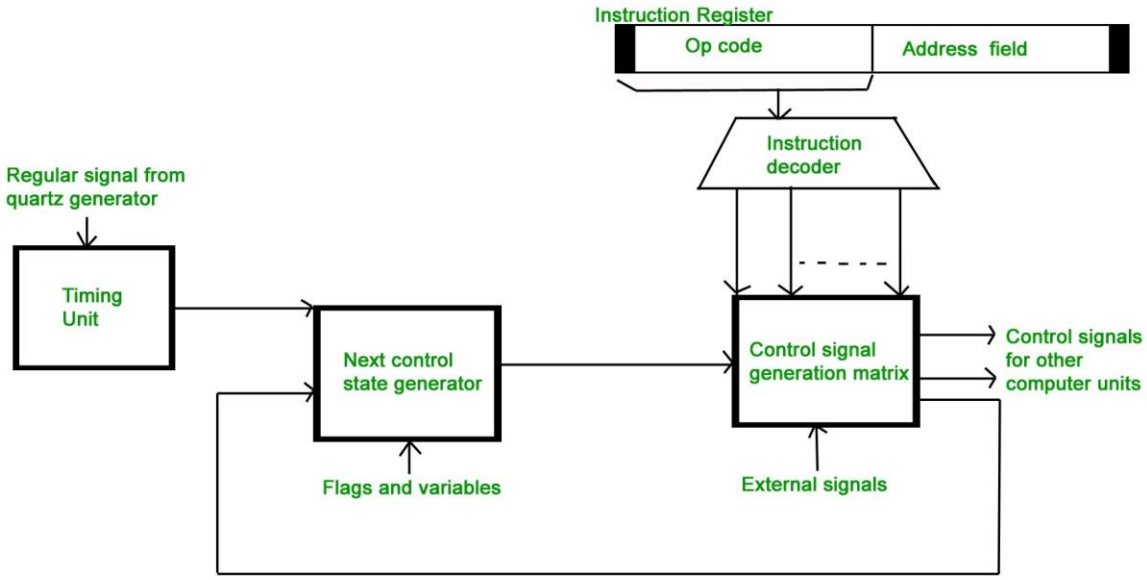
In a multiple bus system many processors may try to access the shared memory simultaneously. To deal with this problem, a policy might be implemented that allocates the available buses to the processors making requests to memory. In particular case, the policy might deal with the case when the number of processors exceeds from the B. For performance point of view this allocation has to be performed by hardware arbiters which, as we will see, add significantly to the difficulty of the multiple bus interconnection networks.



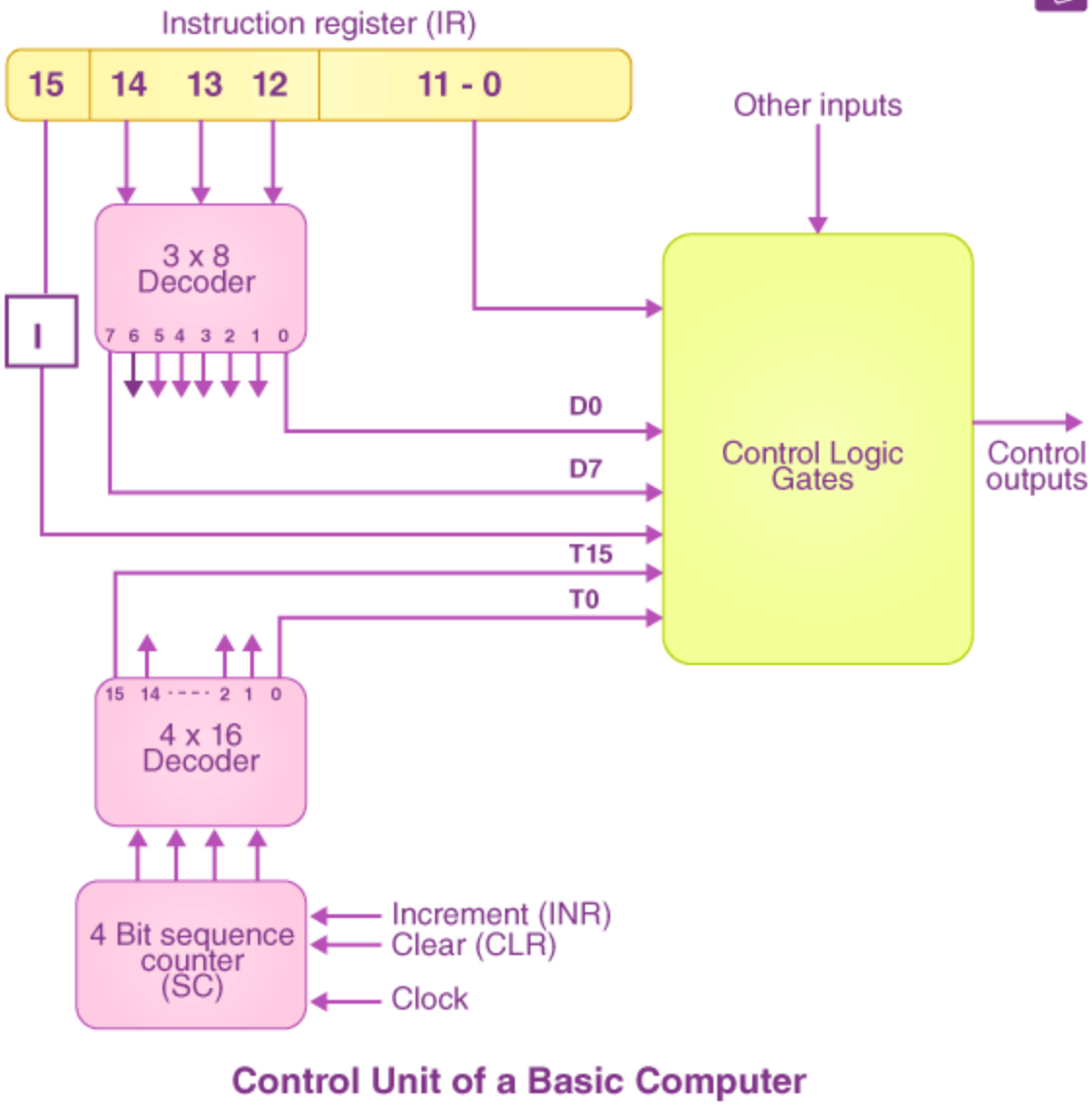
Hardware Control:

Hardwired Control Unit: The control hardware can be viewed as a state machine that changes from one state to another in every clock cycle, depending on the contents of the instruction register, the condition codes, and the external inputs. The outputs of the state machine are the control signals. The sequence of the operation carried out by this machine is determined by the wiring of the logic elements and hence named “hardwired”.

- Fixed logic circuits that correspond directly to the Boolean expressions are used to generate the control signals.
- Hardwired control is faster than micro-programmed control.
- A controller that uses this approach can operate at high speed.
- RISC architecture is based on the hardwired control unit



hardwired control is a method of generating control signals with the help of Finite State Machines (FSM). The control signals that are necessary for instruction execution control in the Hardwired Control Unit are generated by specially built hardware logical circuits, and we can’t change the signal production mechanism without physically changing the circuit structure.



Control Unit of a Basic Computer

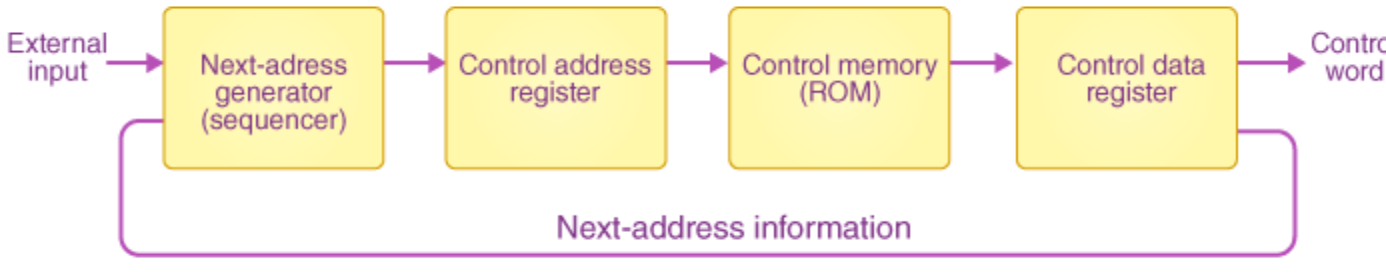
Micro-Programmed Control:

microprogrammed control unit is a control unit that saves binary control values as words in memory. By creating a certain collection of signals at every system clock beat, a controller generates the instructions to be executed. Each one of these output signals causes a single micro-operation, such as register transfer. As a result, defined micro-operations that can be preserved in memory are formed from the sets of control signals.

The programming approach is used to implement a microprogrammed control unit. A program made up of microinstructions is used to carry out a series of micro-operations. The control unit's control memory stores a microprogram composed of

microinstructions. The creation of a set of control signals is dependent on the execution of a microinstruction.

The block diagram of this type of organisation is shown below:



Microprogrammed control unit of a Basic Computer

Characteristics of Micro-programmed Control Unit

- The microinstruction address is specified in the control memory address register.
- All the control information is saved in the control memory, which is considered to be a ROM.
- The microinstruction received from memory is stored in the control register.
- A control word in the microinstruction specifies one or multiple micro-operations for a data processor.
- The next address is calculated in the circuit of the next address generator and then transferred to the control address register for reading the next microinstruction when the micro-operations are being executed.
- Because it determines the sequence of addresses received from control memory, the next address generator is also known as a microprogram sequencer.

Pipeling Concepts:

Pipelining is the process of storing and prioritizing computer instructions that the processor executes. The pipeline is a "logical pipeline" that lets the processor perform an instruction in multiple steps. The processing happens in a continuous, orderly, somewhat overlapped manner.

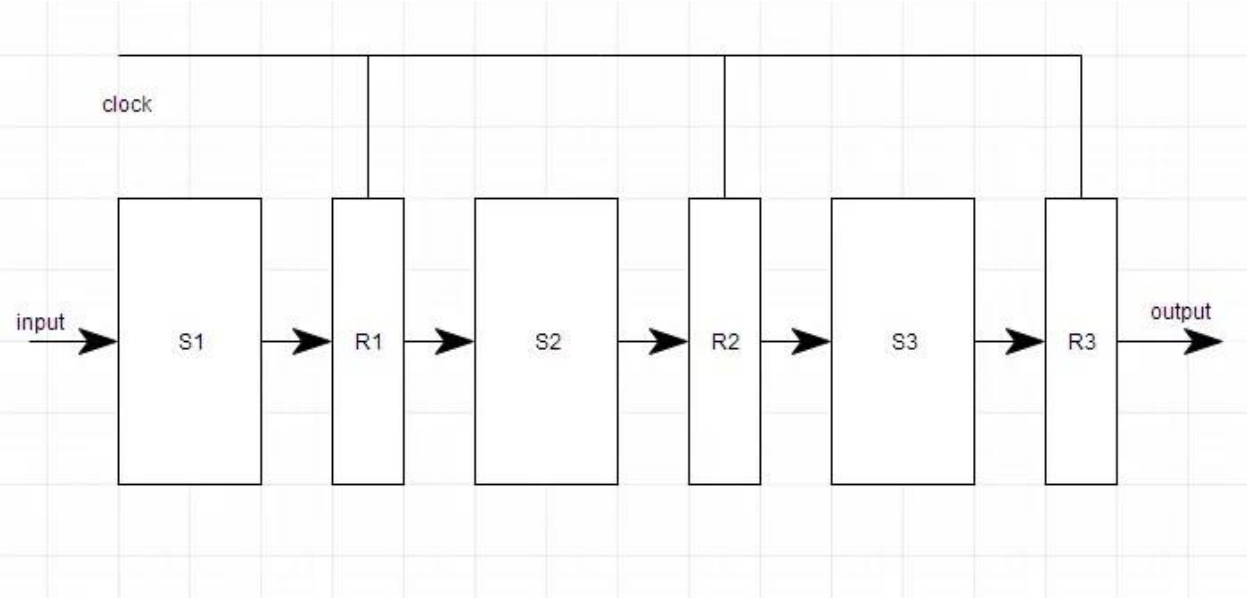
In computing, pipelining is also known as *pipeline processing.* It is sometimes compared to a manufacturing assembly line in which different parts of a product are assembled simultaneously, even though some parts may have to be assembled before others. Even if there is some sequential dependency, many operations can proceed concurrently, which facilitates overall time savings.

Pipelining is a process of arrangement of hardware elements of the CPU such that its overall performance is increased. Simultaneous execution of more than one instruction takes place in a pipelined processor. Let us see a real-life example that works on the concept of pipelined operation. Consider a water bottle packaging plant.

Pipelining is the process of accumulating instruction from the processor through a pipeline. It allows storing and executing instructions in an orderly process. It is also known as **pipeline processing**.

Pipelining is a technique where multiple instructions are overlapped during execution. Pipeline is divided into stages and these stages are connected with one another to form a pipe like structure. Instructions enter from one end and exit from another end.

Pipelining increases the overall instruction throughput.



Data Hazards:

Data hazard, structural hazard, and control hazard are three categories of common hazards. Hazards in the domain of central processing unit (CPU) design are problems with the instruction pipeline in CPU micro-architectures when the next instruction cannot execute in the next clock cycle, which might possibly result in inaccurate calculation results.

Data hazard arises if an instruction accesses a register that a preceding instruction overwrites in a future cycle. Unless we decrease data hazard, pipelines will produce erroneous outputs.

Data hazard in pipelining arises when one instruction is dependent on the results of a preceding instruction and that result has not yet been calculated. Whenever two distinct instructions make use of the same storage. The location must seem to be run sequentially.

In other words, a Data hazard in computer architecture occurs when instructions with data dependency change data at several stages of a pipeline. Ignoring possible data hazards might lead to race conditions (also termed race hazards).

Types of Data Hazard in Pipelining

The data hazard or dependency exists in the instructions in the three types that are as follows:

- Read after Write (RAW)
- Write after Read (WAR)
- Write after Write (WAW)

Handling Data Hazards :
These are various methods we use to handle hazards: Forwarding, Code recording, and Stall insertion.

These are explained as follows below.

1. **Forwarding :**
   It adds special circuitry to the pipeline. This method works because it takes less time for the required values to travel through a wire than it does for a pipeline segment to compute its result.
2. **Code reordering :**
   We need a special type of software to reorder code. We call this type of software a hardware-dependent compiler.
3. **Stall Insertion :**
   it inserts one or more installs (no-op instructions) into the pipeline, which delays the execution of the current instruction until the required operand is written to the register file, but this method decreases pipeline efficiency and throughput.

- *Structural hazards:*
  - *As mentioned above, structural hazards are those that occur because of resource conflicts.*

- *Most common type: When a functional unit is* **not fully pipelined** .
  - *The use of the functional unit requires more than one clock cycle.*
  - *If an instruction follows an instruction that is using it, and the second instruction also requires the resource, it must stall.*

- *Pipelining changes the relative timing of instructions by overlapping them in time.*
- *This introduces possible hazards by reordering accesses*

- *To the register file (data hazards.)*
- *To the program counter (control hazards.)*
    - *Consider the code:*

```
ADD R1, R2, R3
SUB R4, R5, R1
AND R6, R1, R7
OR R8, R1, R9
XOR R10, R1, R11
```

    - *All of the instructions after ADD use the result of the ADD instruction.*

Instruction Hazards:

Instruction hazards: **Pipeline execution of instructions will reduce the time and improves the performance**. Whenever this stream is interrupted, the pipeline stalls, as figure 3.7 illustrates for the case of a cache miss. A branch instruction may also cause the pipeline to stall.

Pipeline hazards are **conditions that can occur in a pipelined machine that impede the execution of a subsequent instruction in a particular cycle for a variety of reasons**.

Control hazards are caused by control dependences. **An instruction that is control dependent on a branch cannot be moved in front of the branch, so that the branch no longer controls it** and an instruction that is not control dependent on a branch cannot be moved after the branch so that the branch controls it.

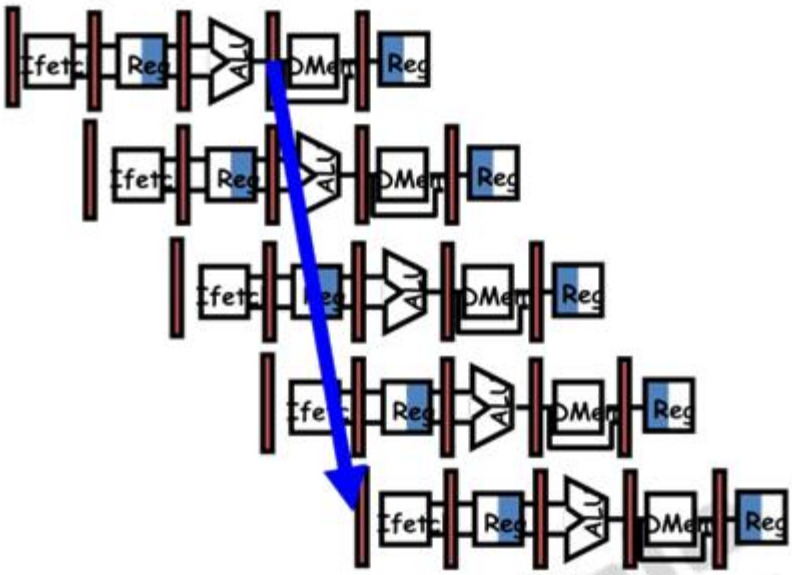


Figure 13.1

Data hazard, structural hazard, and control hazard are three categories of common hazards. Hazards in the domain of central processing unit (CPU) design are problems with the instruction pipeline in CPU micro-architectures when the next instruction cannot execute in the next clock cycle, which might possibly result in inaccurate calculation results.

Influence on Instruction Set::

Influence on Instruction Sets: **Some instructions are much better suited to pipelined execution than other instructions**. For example, instruction side effects can lead to undesirable data dependencies. The machine instructions are influenced by addressing modes and condition code flags.

An Instruction Set Architecture (ISA) is part of the abstract model of a computer that defines how the CPU is controlled by the software. The ISA acts as an interface between the hardware and the software, specifying both what the processor is capable of doing as well as how it gets done.

The ISA provides the only way through which a user is able to interact with the hardware. It can be viewed as a programmer's manual because it's the portion of the machine that's visible to the assembly language programmer, the compiler writer, and the application programmer.

The processor's architecture and instruction set determine how many cycles, or ticks, are needed to execute a given instruction. In other words, **some instruction sets are more efficient than others, enabling the processor to do more useful work at a given speed**.

The **instruction set**, also called **ISA** (**instruction set architecture**), is part of a computer that pertains to programming, which is more or less machine language. The instruction set provides commands to the processor, to tell it what it needs to do. The instruction set consists of addressing modes, instructions, native data types, registers, memory architecture, interrupt, and exception handling, and external I/O.

Examples of instruction set

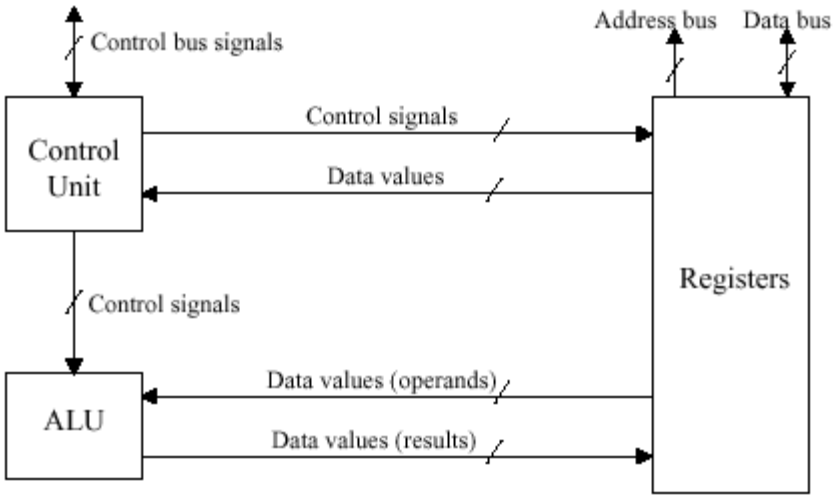
- **ADD** - Add two numbers together.
- **COMPARE** - Compare numbers.
- **IN** - Input information from a device, e.g., keyboard.
- **JUMP** - Jump to designated RAM address.
- **JUMP IF** - Conditional statement that jumps to a designated RAM address.
- **LOAD** - Load information from RAM to the CPU.
- **OUT** - Output information to device, e.g., monitor.
- **STORE** - Store information to RAM.

Data Path and control constructions:

It is worthwhile to further discuss the following components in Figure 4.1:

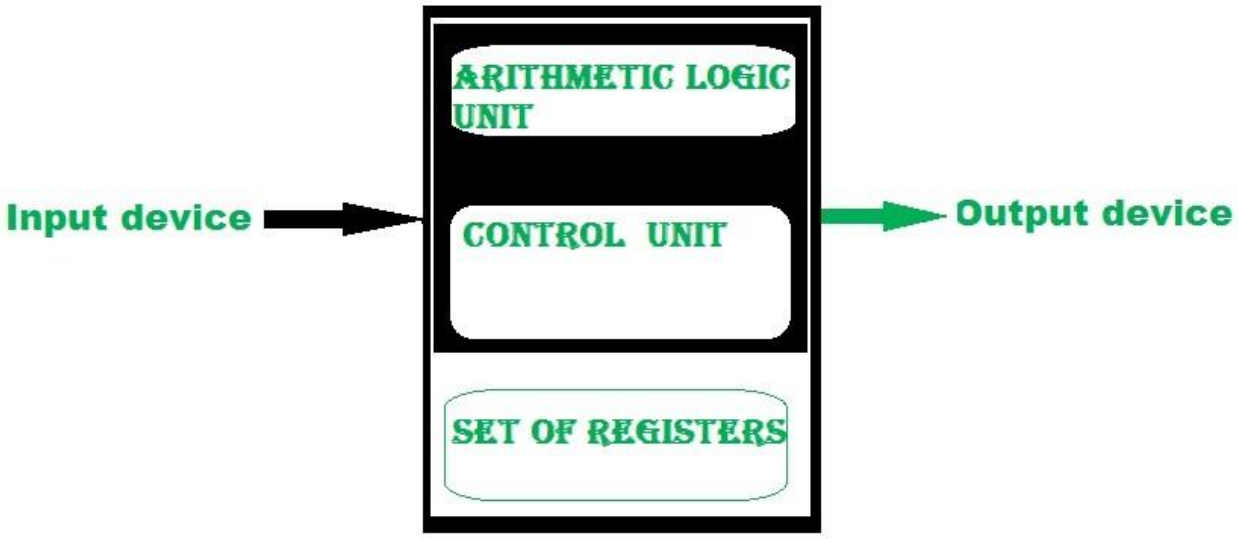
- *Processor (CPU)* is the active part of the computer, which does all the work of data manipulation and decision making.
- *Datapath* is the hardware that performs all the required operations, for example, ALU, registers, and internal buses.
- *Control* is the hardware that tells the datapath what to do, in terms of switching, operation selection, data movement between ALU components, etc.

The processor represented by the shaded block in Figure 4.1 is organized as shown in Figure 4.2. Observe that the ALU performs I/O on data stored in the register file, while the Control Unit sends (receives) control signals (resp. data) in conjunction with the register file.



datapath is **a collection of functional units such as arithmetic logic units or multipliers that perform data processing operations, registers, and buses**. Along with the control unit it composes the central processing unit (CPU). A larger datapath can be made by joining more than one datapaths using multiplexers.

Representing and storing numbers were the basic operation of the computers of earlier times. The real go came when computation, manipulating numbers like adding, multiplying came into the picture. These operations are handled by the computer's **arithmetic logic unit (ALU)**. The ALU is the mathematical brain of a computer. The first ALU was INTEL 74181 implemented as a 7400 series is a TTL integrated circuit that was released in 1970.



Central Processing Unit

Datapath Elements

Essentially a DATAPATH consists of the following elements.

ALU – one or more to carry out the computation. ALU is not only used in data operations but also in address calculation too.

Instruction Register and decoder – to decode what instruction to be executed and how to execute the instruction.

Program Counter – Always points to the next instruction to be executed and manages the flow of instructions.

Memory – Instruction memory is mostly read-only in the fetch phase. Data memory is required to access the operand and result writing. The memory is accessed over a bus from the CPU. To access memory, the address of the memory location is required in addition to Read/Write of data. The Memory Address Register (MAR) holds the address of the memory location to be accessed.

Registers – Registers are in physical proximity and internal to the CPU. These are ultra-fast than Memory. Most of the times the operands are brought from memory and kept in registers. Rather these registers are used as a workspace and rough space for workout.

Register files – These are multiport register set enabling faster and parallel access to the register set.

Internal Registers – Instruction Register, Memory Address Register, Memory Data Register. These are not accessible to the programmer.

Multiplexers – Anything is reachable with these. These allow what is to be allowed out based on the selection input.

Internal bus – which connects all these elements.

Control unit – the master which manages the data path elements.

Important Questions:

Short Questions:

1. What is meant by basic processing unit?
2. What is hazard control?
3. What is instruction hazard?
4. What is data path and it's control?

Long Questions:

1. What is Bus Organization, explain multiple bus organization with examples?
2. How to execute a complete instruction?
3. What is Hazard, data hazard and instruction hazards?
4. Describe in brief about data path and control construction?

Question papers for sample:

S.V.U COLLEGE OF COMMERCE MANAGEMENT AND COMPUTER SCIENCE :: TIRUPATI

DEPARTMENT OF COMPUTER SCIENCE

TIME:2hours INTERNAL EXAMINATION-1 MAX MARKS:30

SECTION:A

Answer any five from the following 5*2=10m

1.Explain about Address.

2.explain about plds and counters.

3.define multiprocessors.

4.what is instruction sequencing?

5.what are addressing modes?

6.explain main memory operations.

7.define memory locations.

SECTION-B

ANSWER ANY ONE QUESTION FROM EACH UNIT 2*10=20M

UNIT-1

8. a) Explain NAND Gate implementation.

b) what is flipflops?

(OR)

9.Explain block diagram of simple computer & Functional Units.

Unit-2

10. a) write about addressing modes and types.

b) Explain about instruction format and its types.

(OR)

11.Explain Locations and Addresses(Byte Addressability & Big Endian -Little Endian).

S.V.U COLLEGE OF COMMERCE MANAGEMENT AND COMPUTER SCIENCE :: TIRUPATI

DEPARTMENT OF COMPUTER SCIENCE

TIME:2hours INTERNAL EXAMINATION-2 MAX MARKS:30

SECTION:A

Answer any five from the following 5*2=10m

1.Explain about Machine Instructions.

2.What is Optical memory?

3.Define Universal Gates.

4.Explain cache memory and virtual memory.

5.what is Floating Point?

6.Derive DMA.

7.what is n-bit parallel address?

SECTION:B

ANSWER ANY ONE QUESTION FROM EACH UNIT 2*10=20M

UNIT-1

8.a) what is Interrupt and Handling Interrupt requests?

b). Discuss about Strobe control.

(OR)

9.a) Explain Bus Synchronous.

b) Design Fast adder.

Unit-2

10.a) what is virtual memory?

b) What is Super Scalar Processor?

(OR)

11.a) Briefly Explain Pipelines and Hazards.

b) Explain about Ultra Sparc-II Processor.

MASTER OF COMPUTER APPLICATIONS DEGREE EXAMINATION

FIRST SEMESTER

PAPER MCA 103: COMPUTER ORGANISATION

(Under C.B.C.S Revised Regulations W.e.f.2021-2023)

(Common paper to University and all Affiliated Colleges)

Time:3 hours Max.Marks:70

PART-A

(Compulsory)

Answer any five of the following questions each question carries 2 marks (5*2=10)

1.a) What is Flipflops?

b) Derive PLDs.

c) write about Addressing Methods?

d) What are main memory Operations?

e) explain about Interrupts.

f) explain about Cache memory.

g) write about Standard I/O Interface.

h) Explain about Virtual Memories.

i) Write about Bus Organisation.

j) Define RAM.

PART-B

Answer any ONE question from each unit

Each question carries 12 Marks (5*12=60)

UNIT-1

2.Explain Block Diagram of Simple computer & its Functional Units.

(OR)

3.Difference between Multiprocessors and Multi computers.

UNIT-2

4.Explain about Addressing Modes and Its types.

(OR)

5.What is Micro Instruction Sequencing.

UNIT-3

6.What are working principles of DMA.

(OR)

7.what is Interrupt and Handling Interrupts.

UNIT-4

8.what is mapping function and various Mapping Techniques?

(OR)

9.what is Virtual Memory? How virtual Memory is Translated?

Unit-5

10.Briefly explain pipelines and Hazards and four Stages of pipelines.

(OR)

11.Explain about Microprogrammed control Unit.