# BCS-302

# Computer Architecture & Organisation (CAO)

Computer Architecture and organisation are two closely related to a computer system they referred to the different layers of abstraction in computers.

Computer Architecture:- Computer Architecture is a high level description of a subject.

2) It defines the functional behaviour design philosphy and performance capability of computer system.

3) In other words it is the conceptual design that provides the information about how the computer system should perform and how it should be structured.

4) following terminology can be considered in the computer architecture.

- Design and behaviour
- Performance & optimisation
- focus on end goals.

Examples of computer architecture :-

1) Von Neumann Architecture (for single memory for both data and instructions.

2) Harward Architecture (with separate memory for data and instructions).

Computer Organisation:- It deals with the operational abpects of the computer and how the various components are connected and were together.

2) It is more practical and lower level implementation of how hardware components commuunicate with each others.

## Important terminology

1) Micro-Architecture :- It involves how instructions are executed within the processor and the various execution units such as ALU

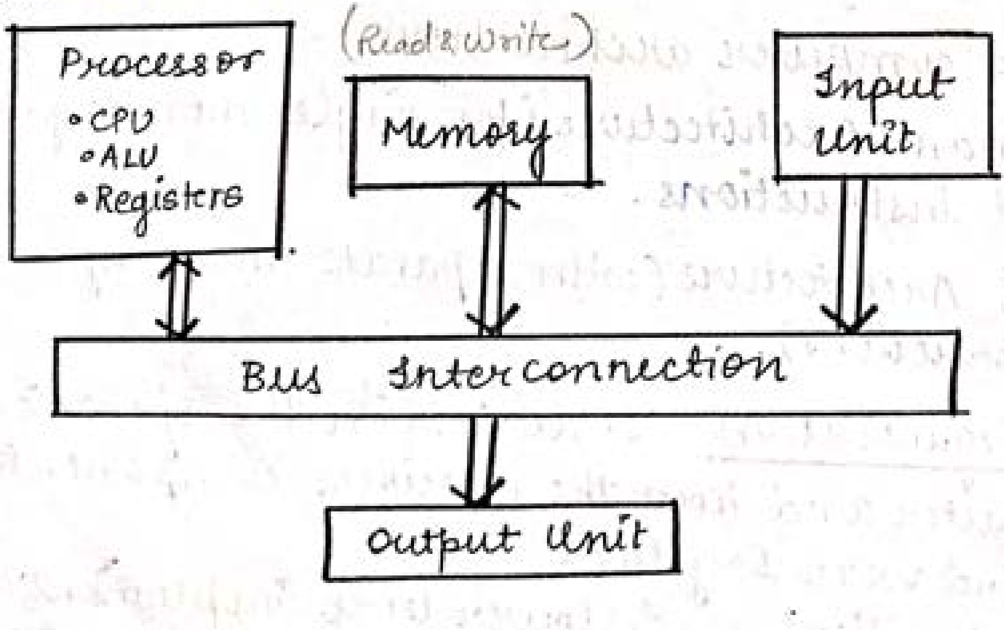
2) Hardware Details :- It specifies hardware components for Example multipliers, adder memory controller and clock signal

3) Resource Utilisation :- It deals with the deficient hardware resource also focus on low level details. such as transistors needed, power consumption and overall efficiency.

## Functional Units of Digital System

A digital system consists of several functional units, each responsible for performing specific task such as. data processing, storage and control operations. These units are interconnected to form a computing system capable of executing programs and performing various functions &

Following block diagram shows the functional units and there interconnections:-



## CPU (Central Processing Unit)

The CPU is the ~~third of t~~ heart of the digital system, responsible for executing instructions, performing arithmetic and logical operations and controlling other units.

1) **ALU**:- • It performs basic arithmetic operations (addition, subtraction, multiplication, division).
- It also executes logical operations (AND, OR, NOT etc)

2) **CU (Control Unit)**:- • It directs the operation of the processor.
- It decodes instructions and generate control signals to coordinate the execution of instructions.
- Manage the flow of data between CPU and other components

3) **Registers**:- • These are the high speed storage location in small size available within the CPU.
- Registers temporarily hold the data being processed or used in calculations (for ex- accumulator, instruction register and program counter).
↳ Instructions for the next task

4) **Memory**:- The memory unit stores the data and instruction
- It is divided into two main types -
  a) Primary Memory (RAM or main memory):-
  b) Secondary memory (External memory)

5) **Input unit**:- The input unit is responsible for accepting data and instructions from external devices & converting them into a format understand by the digital system.
For Eg. Keyboard, mouse, microphone etc.

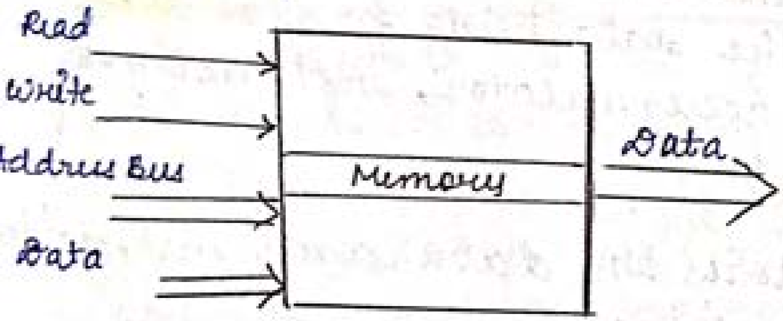
6) **Output unit**:- The output unit converts the processed data from the digital system into a form that can be understood by external devices.
For Eg:- Monitor, printer, speaker etc.

**Interconnection of functional Units.** A computer consists of a set of componints or modules.

- There are three basic types (•CPU, Memory, •i/o) module that communicate with each other.
- The functional components consists of bus architecture for communication.
- The connection of path connecting the various modules. is called the interconnection structure.

Following figures shows the type of exchange that are needed by indicating the major form for each module type:

**Memory module:-** Generally a memory consists of N-words of equal length.



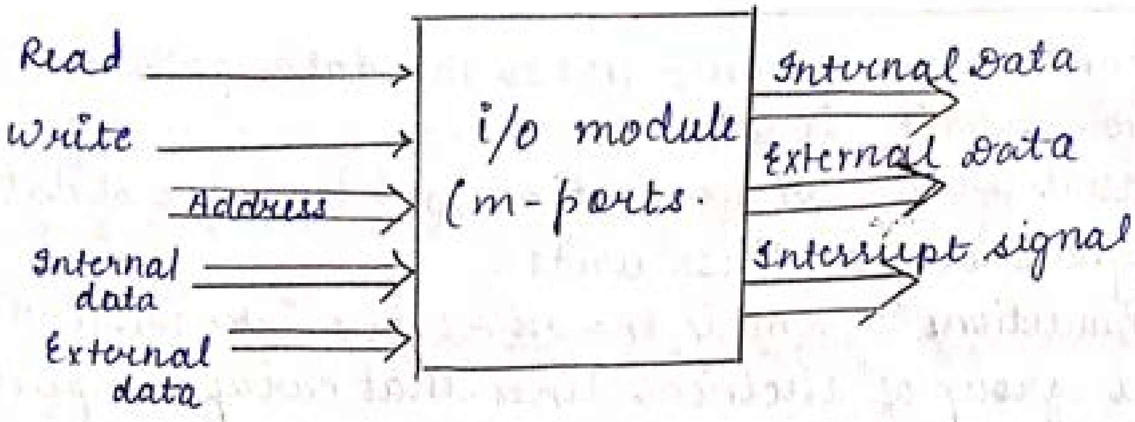
Each word is asigned a unit numerical address. A word of data can be read from memory or written into the memory.

- The nature of the operation is indicated by read and write control signals.
- The notation for the operation is specified by an address

**i/o module:-** From the intrial point of view computer system i/o module is functionally similar to memory.

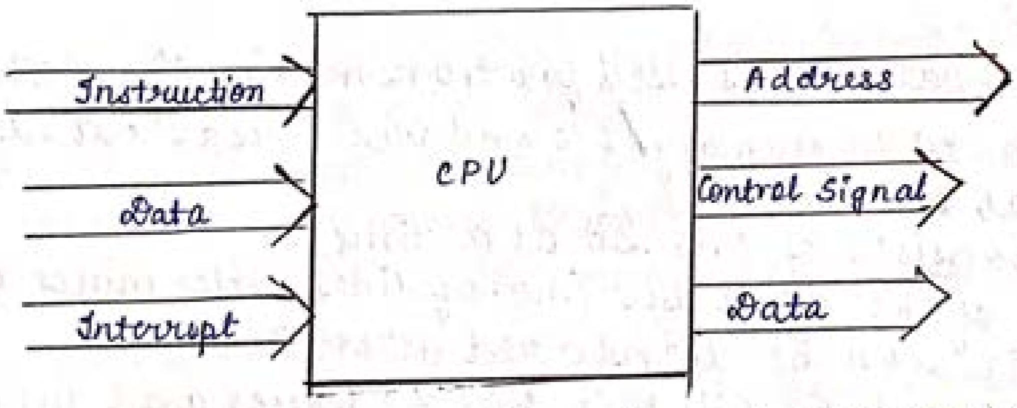
- There are two operations -(read and write)
- An i/o module may also control more than one external devices.
- We can consider each of the interface to an external device as a part and give each a unit address (m-ports)
- In addition to there are external data ports for the input and output of the data with an external devices.
- Finally an i/o module may be able to sent interrupt signals to the processor.

Following block diagram shows the i/o module:-



Processor (CPU) :-

- The Processor reads an instructions and data, write the data after processing and uses control signals to control the overall operation of the system
- It also receives the interrupt signal
- following figures shows the block diagram for the CPU module



Note :- The interconnection structure must support the following types transfer

1. Memory to processor
2. Processor to Memory
3. i/o to processor
4. Processor to i/o
5. i/o to Memory
6. Memory to i/o

for the last two operations an i/o module is allow to exchange the data directly with memory without ~~flowing~~ going through the processor by using the Direct Memory Access Controller (DMA or Controller

# Bus Interconnection System

- The interconnection system refers to the data path which link the functional units together.
- It is essential for the communication and transfer of data and control signals between these units.
- The interconnections system is known as bus interconnection.
- A bus is a group of electrical lines that carry computer signals or bits.
- A bus consists of vertical lines and each line is capable of transmitting signals represinting binary '1' or binary '0'.

On the basis of function performed buses can be classified into three types :-

a) Data bus

b) Address bus

c) Control bus

a) **Data bus** :- Data bus is used for transmitting the data/instructions from CPU to memory/I/O and vice-versa. Data bus is a bidirectional bus.

- Each line consists of one bit at a time.
- The width of the data bus (no. of lines determines the how much data can be transferred at once.
- For Example: A 32 bit bus has 32 wires and therefore it can transmit 32 bit of data at a time.
- ~~The width of the data bus determines how much data~~
- The wider the bus width faster will be data flow in the databus and better system performance.

**Address Bus** :- Address Bus is used to carry the address from CPU to memory / I/O devices.

- Address bus is a unidirectional bus.
- It is used to identify the particular location in memory. that may be source or destination.
- For Example If the processor wants to read or write a memory

word it puts the address of the desired word from the address line.

- The width of the address bus determines the amount of ~~phys.~~ digital memory addressable by the processor.
- Basically it determines the size of the memory that the computer can use.
- The wider the address bus more memory of a computer will be able to use.
- The addressing capacity of the system can be increased by adding more number of address lines.
- for Example If address bus consists of 16 bits therefore it can have

$$2^{16} = \underset{\substack{\downarrow \\ 1K}}{2^{10}} \cdot \underset{\substack{\downarrow \\ 64}}{2^{6}}$$

1K = 1024

$= 64$ K memory locations.

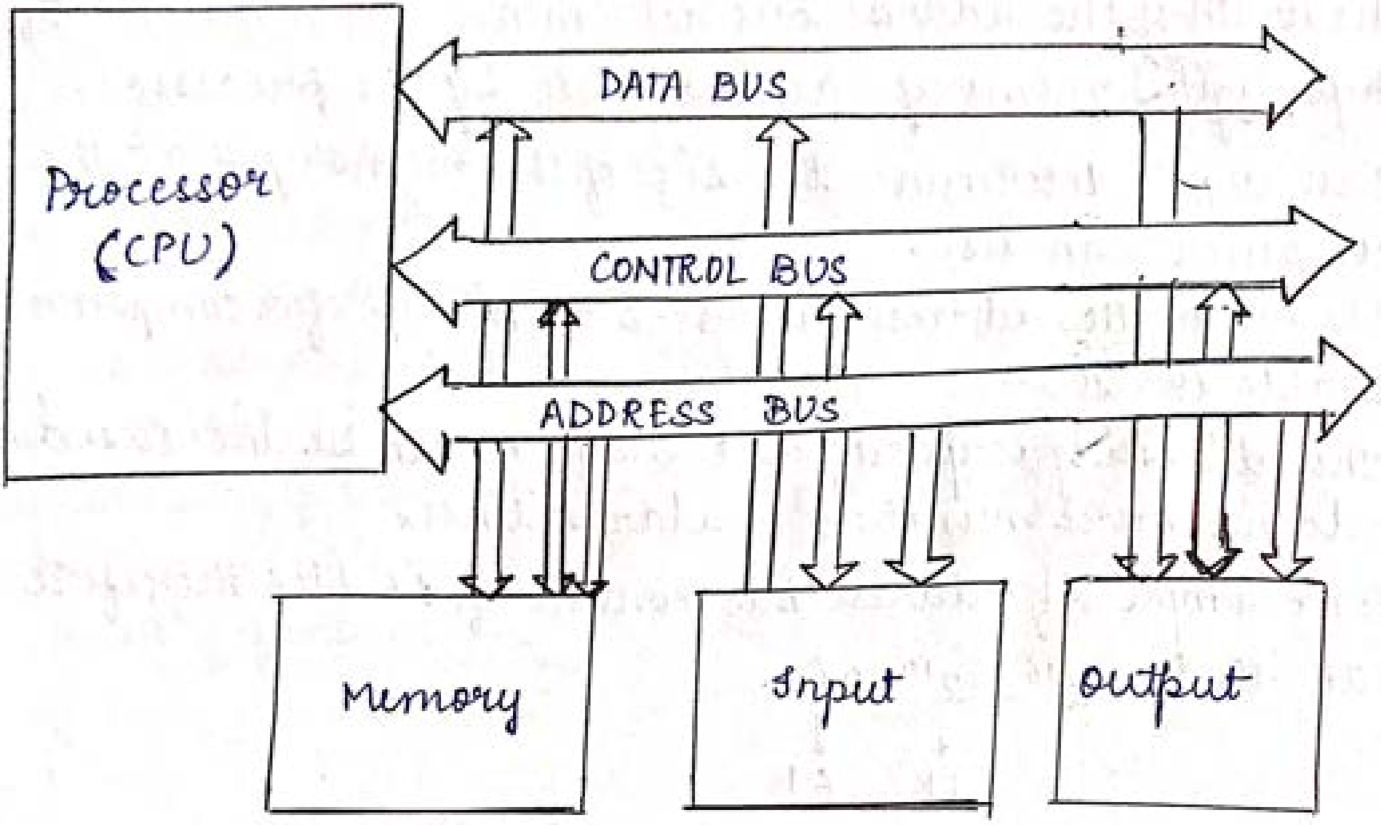
Controlbus :- The control lines are used to control the access and the use of the data and address lines.

- The control bus is used to transfer the control and timing signal from one component to other.
- The CPU uses control bus to communicate with the devices that are connected to the computer system.
- The control signals are generated in the control unit of the CPU.

Following are the basic control signals generated in the control unit:-

1. memory Read
2. Memory write
3. I/O read
4. I/O write
5. Transfer acknowledge
6. Bus Request
7. Bus Grant
8. Interrupt Request
9. Interrupt acknowledge
10. clock signal

- can be unidirectional or bidirectional based on the processor used.



<u>Bus Arbitration</u>:- Bus arbitration is a process of determining which module or device in a computer system gets access to a shared bus when multiple components want to use it simultaneously.

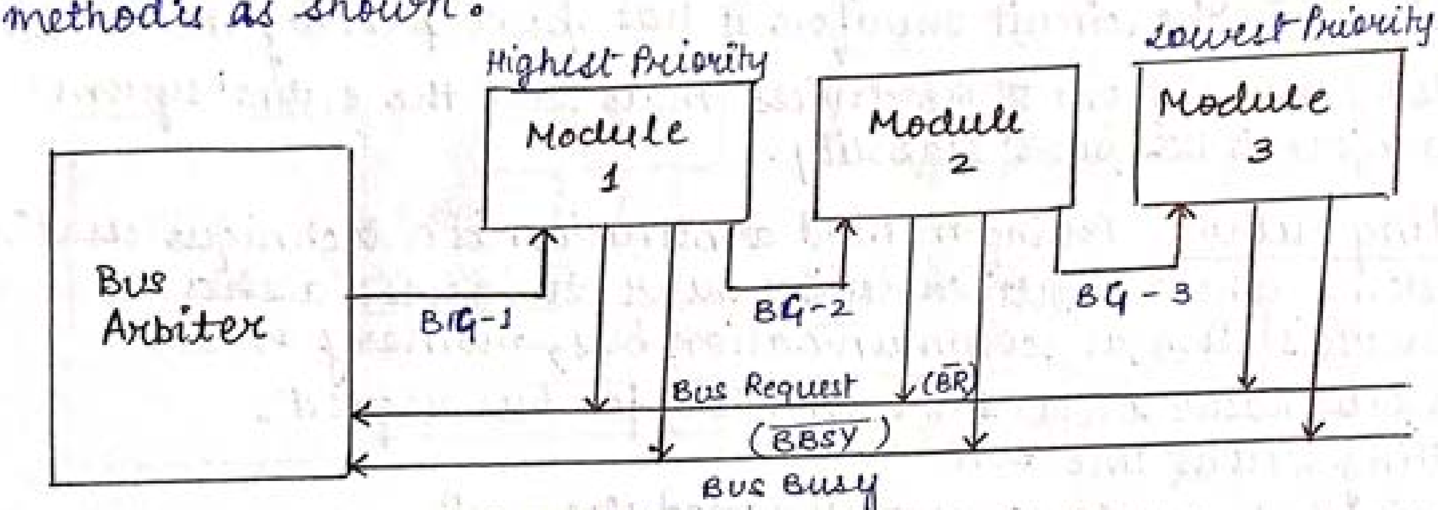
- The device that allow to initiate data transfer on bus at any given time is called <u>bus master</u>.
- There can be only one bus master at any given time.
- So bus arbitration is a process by which next device to become the bus master is selected. and bus mastership is transferred to it the
- The bus arbitrator decides who will become the current bus master
- There are two approaches to bus arbitration.
  a) Centralised bus arbitration
  b) Distributed bus arbitration

<u>Centralised bus arbitration</u>:- In centralised bus arbitration a central bus ~~arbi~~ signal bus arbitrator or bus controller performs the required arbitration.

- The bus arbitrator may be the processor or a separate controller connected to the bus.

- There are three different arbitrations scheme that use the. centralised bus arbitration approach.
- These schimes are
  a) Daisy Chaining
  b) Podding method
  c) Indipendent Request

Daisy Chaining arbitration The system connections for daisy chaining method's as shown.



- A single bus arbiter performs the required arbitration.
- Normally the processor is the bus master unless it grants bus mastership to one of the module.
- Daisy chaining is a bus arbitration method used to manage or acess the bus in a sequential manner.
- In this method devices are connected in a chain and only one device at a time has control over the bus.
- Any module indicates that it needs to become bus master by activating Bus Request line.
- When $\overline{BR}$ is activated the processor activates bus grant signal BG-1 Indicating to module 1 that may use the bus when it becomes free.
- If module 1 is requesting the bus it block propogation of bus grant signal to other device otherwise it passes the bus grant signal to next module.
- Current bus master indicates to all devices that it is using the bus by activating the bus busy line.
- Arbiter circuit ensures that only one request is granted at any given time according to the predefined priority scheme.

Advantages:- 1) The design is simple.

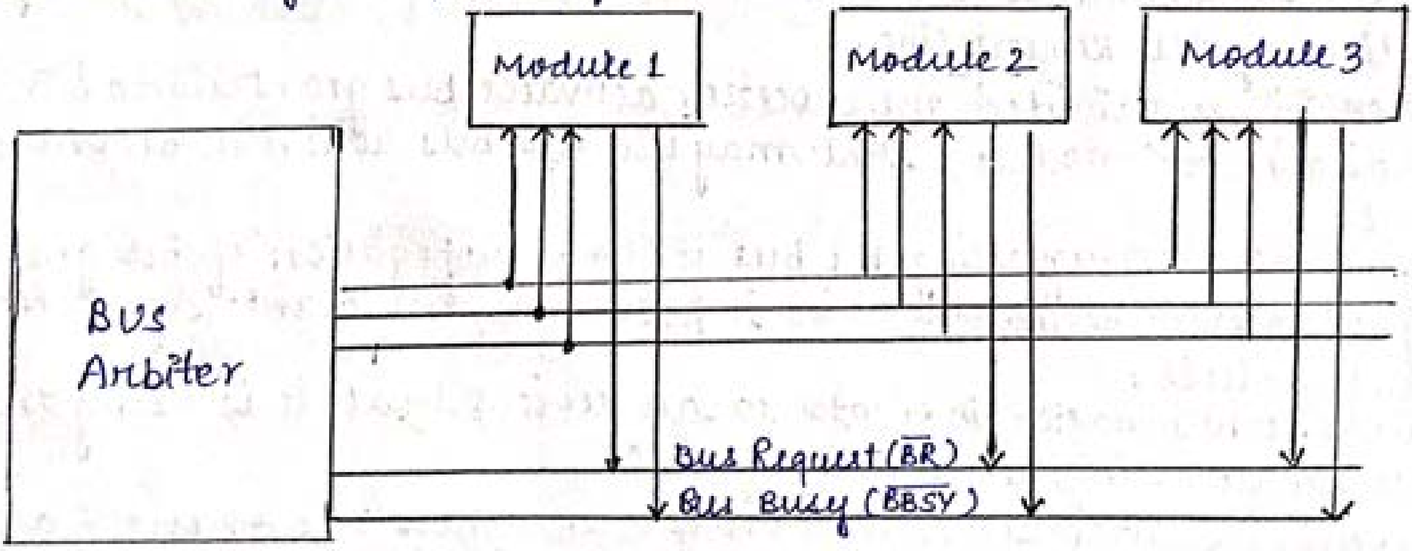
2) The number of control line is less.

Disadvantages:- 1) The priority of the bus master is rigid and depends on the physical proximity with the bus master with the bus arbiter that is the module that is near to the bus arbiter gets highest priority. therefore it has poor performance.

- The bus is granted serially and hence a propogation delay is induced in the circuit Therefore it has poor priority mechanism.
- Also failure of one of the devices may fail the entire system. Therefore it has poor stability.

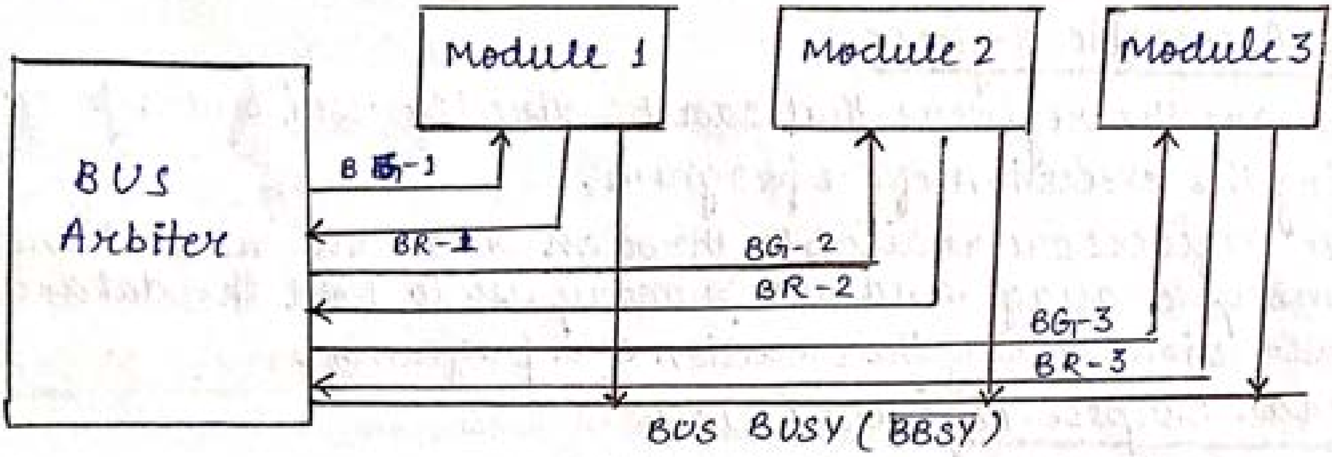
Polling Method Polling method arbitration is a technique used in systems where multiple devices near to access a shared resources such as communication bus, memory or CPU.

- All bus master have the same line for bus request.
- Polling count line = $n$
- Maximum number of connected modules = $2^n$
- If bus busy line is inactive then bus controller gives the bus grant addresses based on the bus request.
- Bus ~~expt~~ Controller polls the system in well define order as per priority.
- Priority is set by the bus controller.
- Once system receives its own address it will active the bus busy signal and take control of system bus.
- The block diagram for the following method is as shown:-



## Independent Request Arbitration:-

- In independent request bus arbitration each device that wants access to the bus has a dedicated request line going to the bus arbiter.
- In other words, all bus master have Individual bus request line.
- Individual module can request through the bus controller by its own bus request line and the bus arbiter can grant the bus through the module through which requested the bus.



## Register:-

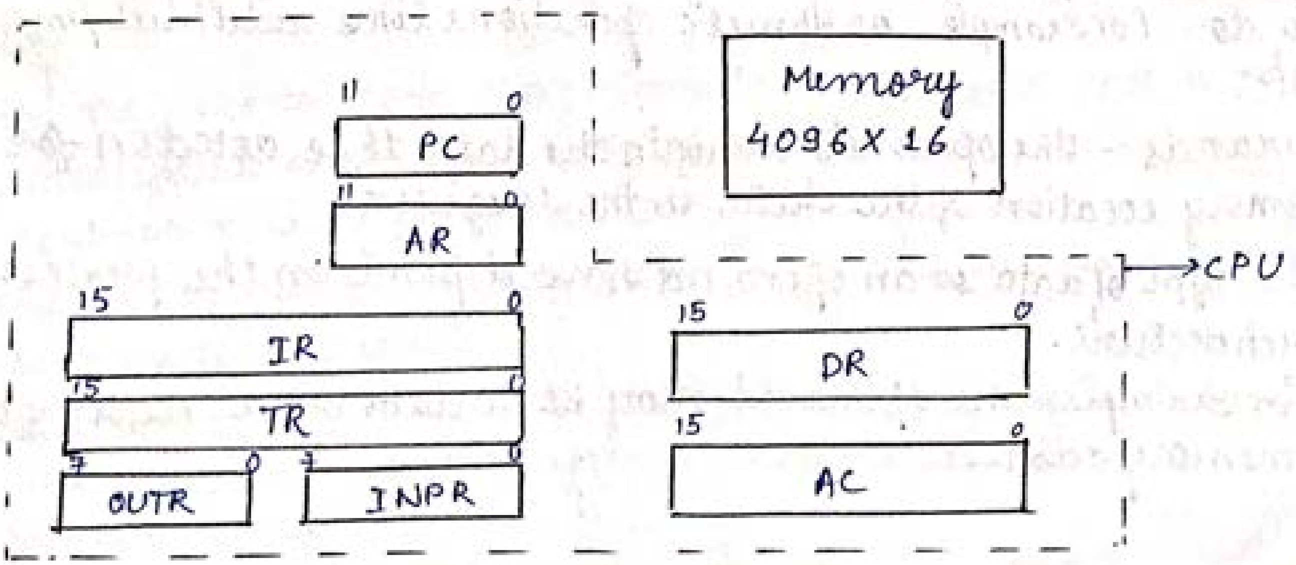
→ Registers are used to store data temporarily in fast memory location.

→ Registers are built in mimory location In the processor.

→ Registers can store various types of information such as data values, memory address or instructions.

→ Registers are used as a intermediate storage for data during arithmetic and logical operations and other processing operations.

→ following figure shows the registers in basic computer system:-



PC - Program Counter
AR - Address Register
IR Instructions Register
TR - Temporary Register
OUTR - Output Register
INPR - Input Register
DR - Data Register
AC - Accumulator

User Accessable Register :-

→ These are the registers that can be directly used by the programmer during the execution of the program.

→ These registers are available through machine instructions or assembly language and are primary use to hold the data, address or instructions during the execution of a program.

General Purpose Registers (GPR)

→ GPR can be used by the program the

→ They can stored both data and memory address and are used for performing arithmetic, logical and data transfer operations

→ These are numbered as $R_0, R_1, R_2 \ldots, R_{n-1}$.

→ These are used to store temporarily data during any operations

→ GPR contains operands for any opcode (operational code).

Note :- Opcode and operands are the two parts of machine language instruction that specifies what the processor should do and what data should it act on.

Opcode :- Opcode / Instruction code that tells the processor what to do. For Example Arithmetic operations like add / sub / multiply / div etc.

operands :- The operands contain the data to be acted on for the memory location of the data in the register.

• The type of data ~~or~~ an operand have depends on the processor architecture.

• For Example :- An operand may be a data or a register or a memory address.

Accumulator:- The accumulator register is located inside the AL

- It is used during arithmetic and logical operations of ALU.
- It store intermediate result during ~~turbul~~ calculations.
- Basically It is a register used to hold one of the operand for ALU operation and it store the result of these operations.

(8bit)
① MOV A, Register:- It moves the content of another register into the accumulator. where R is a general purpose register.

② MVI A, Data:- Loads the intermediate 8 bit data value directly into the accumulator.

③ LDA, 2010:- (Load Data Accumulator)
LDA is the opcode and 2010 is the memory location
- LDA 2010 command moves the content of the 2010 memory location to the accumulator.

PROGRAM COUNTER

Program counter has a main role in the control flow of a program by heap tracking of the next instructions to be executed.
- The program counter holds the memory address of the next instructions that the CPU will execute.
- After each instructions is fetched from the memory, the program counter is automatically incremented to point to the next instructions in sequence.
- It ensures that instructions are executed in the correct order and it can be modified to jump to different parts of the program ( For eg:- loop, function and branching.
- The size of the PC (no. of bits) depends on the addressable memory space for Ex:- In a 16 bit system bus 8085 microprocessor ) The PC is 16 byte

Data Register:-
- Data Registers are used to store temporary data that the CPU needs to process, manipalate and transfered within the CPU.
- It can be used for various purposes such as I/O operations.
- It generally stores specific type of data. for Ex input data, output data or intermediate results.

operation of a data Register The operations performed by a data register usually depends on system architecture but the common operation are -

1) Loading Data:- Data is loaded into the register from memory, i/o devices or another ~~e~~ register

2) Holding Data:- Temporarily holds the data that is to be processed or transferred to an other components.

3) Data Transfer:- ~~s~~ transfer the data between CPU and memory or between i/o devices & memory.

4) Data Output:- It send the data from the register to the external components.

ADDRESS REGISTER

An address register is used in different addressing modes like direct, indirect or index addressing.

- It stores the location of the data or instructions in memory but does not directly ~~w~~ interact with the memory system.
- The CPU uses address ~~s~~ register to specify the memory location that it wants to access.
- These registers play an important role in accessing memory during program execution by holding the data or instructions that the processor needs to fetch.

Basic operations:-

1) Loading Address:- The address register can be loaded with a memory address that points the data or instruction to be accessed.

For Ex:- MOV AR, 2000

Load address 2000 in the address register

2) Increment or decrement:- During loops or when accessing the array the address stored in the register is generally incremente or decremented.

for Ex:- INC AR

Increment the value of the address register.

- The width of the address register is generally determined by the architecture address bus width that define the amount of memory that the CPU can address.

For Ex:- In a 32 bit system the address register is usually 32 bits wide.

Instruction Register (IR):-

- The instruction register holds or store the instruction that is currently in execution.
- The CPU mainly uses IR to get the Instruction, decode it and decide which operations have to be perform to execute the instruction.

Operation:-

1) Fetch/~~fetch~~ cycle:- During the fetch cycle the CPU retrives the next instructions from memory and store it in the instruction register.
   - The address of the instruction to be fetched is held by the PC and the instruction is fetched from that memory address.
2) Decode :- Once the instruction is loaded into the IR it is decoded to determine what operation the CPU needs to perform
   - The decoding process Involves breakdown the instructions into opcode and operands.
3) Execute:- After decoding the CPU uses the information from the instruction register to execute the necessary operation.
   - The Instruction register itself does not perform the execution but holds the instructions during the entire process.

TEMPORARY REGISTER (TR):

The Intermediate result of the calculations is stored in these registers. ~~there~~

- There is not any specific purpose for temporary register so they can be used for any purpose by the CPU.
- Because of this sometimes they refer as GPR.

INPUT REGISTER (IR):-

The input register is used to store the input data that is given to the CPU from an external device ~~of~~ or source like mouse or Keyword.

OUTPUT REGISTER :-

Output register will store the result or the output from the CPU. to an external device such as printer or monitor

# Bus & Memory Transfer

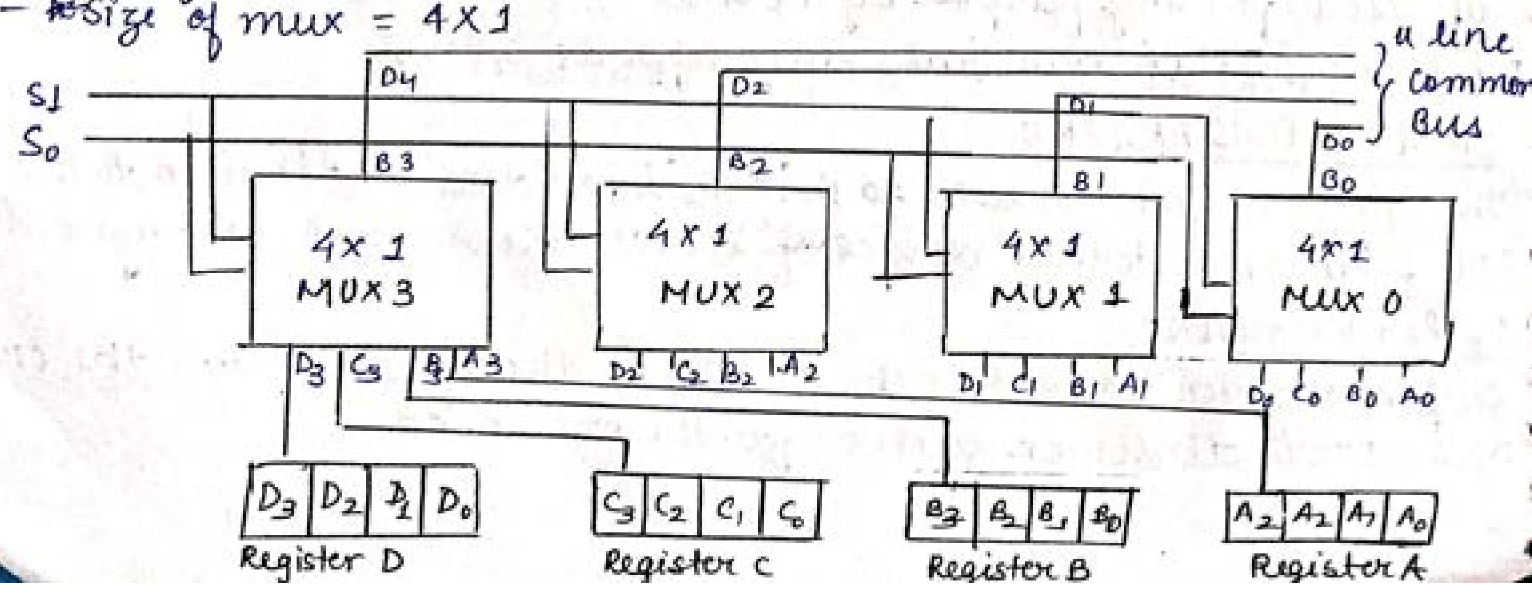
A more efficient method for transferring information between registers in a multiple register configuration is a common bus system.

- Generally the digital computer has many registers.
- A path must be provided to transfer information from one register to another.
- A common bus consists of set of common lines, one for each bit of a register.
- Control signals determine which register is selected by the bus during each particular register transfer.
- common bus system may be design with –
  1) By using multiplexer
  2) By using tri-state buffer.

## Designing of bus system using multiplexer

The multiplexer select the source regester ~~use~~ whose binary Information is then placed on the bus.

- consider the operation for four registers.
- In general –
  ⓐ No. of Multiplexers (MUX) = Size of register
  ⓑ Size of Multiplexers = No. of Registers
- So if we consider four register operation and each register is of 4 bit. therefore, we need
  – No. of mux = 4
  – Size of mux = 4×1



- The bus consists of four 4×1 Mux Multiplexer each having four data inputs from 0 to 3 and two selection inputs $S_0$ and $S_1$
- The diagram shows that the bites in the same significant position in each register are connected to the data inputs of one multiplexers to form one line of the bus.
- Therefore multiplexers 0 multiplex the 4 zero bits of the register. mux 1 multiplex the 4 one bites of the register and similarly for the other 2 bits.
- The 2 selection line $S_0$ & $S_1$ are connected to the selection inputs of all four multiplexers.
- The selection line choose the 4 ~~f~~ bits of one register and transfer them into the 4 line common bus.

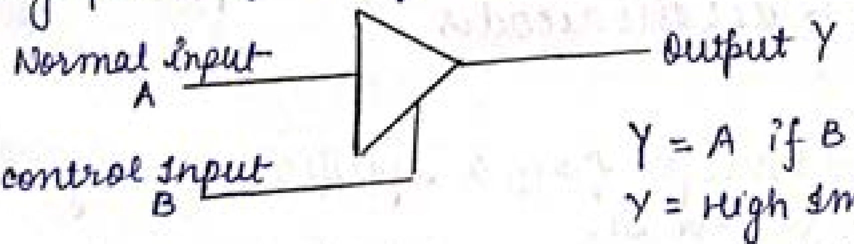
following table shows the relation between selection line and register selected.

| $S_0$ | $S_1$ | Register selected |
|---|---|---|
| 0 | 0 | A |
| 0 | 1 | B |
| 1 | 0 | C |
| 1 | 1 | D |

**By using tri state buffer or three-state bus Buffer**

(Application =

- A bus system can be constructed with three state gates instead of multiplexer
- The tri-state gate is a digital circuit that have three states.
- The most commonly used three state gate in case of bus system is a buffer ~~or~~ gate.
- The graphical symbol of a three state buffer gate can be represented as -

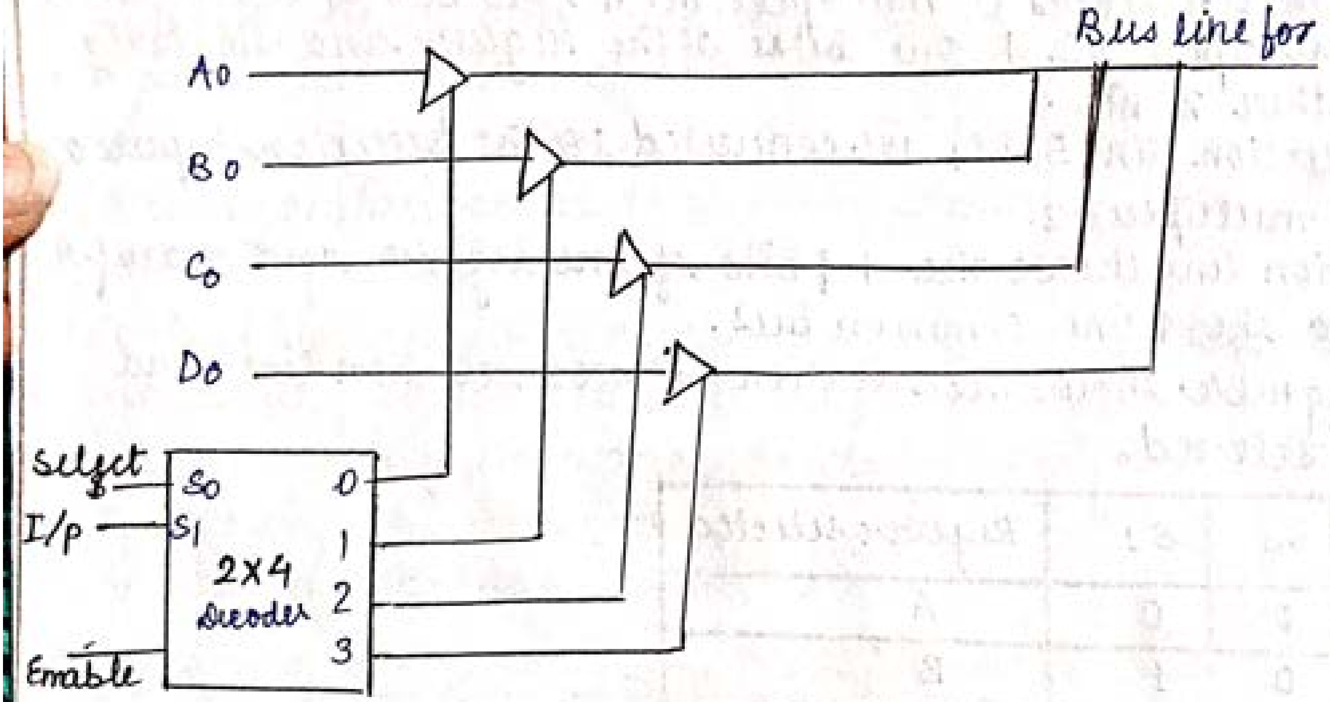


Y = A if B = 1

Y = High impedance if B = 0

- The control input determines the output state. when the control input is one the output is enabled and the gate ~~ts~~ behaves like any conventional buffer with the output = Normal input.

- When the control input is zero the output is disabled and the gate goes to a high impedance state regardless the value in the normal input.
- The construction of a bus system will tri-state buffer is as shown:-

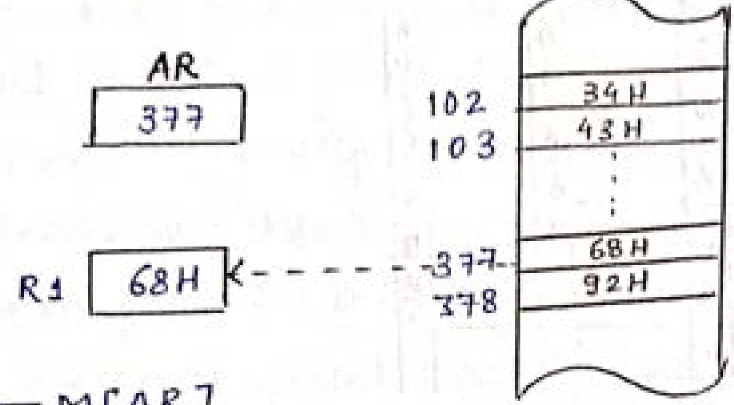


- The outputs of ~~fore~~ four buffer are connected together to form a single bus line.
- The control inputs to the buffer determines which of the four normal inputs will communicate with the bus line.
- No more than one buffer may be in the active state at any given time.
- The connected buffers must be controlled so that only one tri-state buffer has access to the bus line while all other buffers are maintained in a high impedance state.
- one way to ensure that no more than one control input is active at any given time is to use the decoder.

## Memory Transfer

- Memory transfer means to transfer from a specific memory location to a register and vice-versa.
- A transfer from memory to register is called READ operation.
- A transfer from register to memory is called WRITE operation.

**Memory READ**:- A memory READ operation first search for a location in the memory and then transfer the values located at that location to a register.

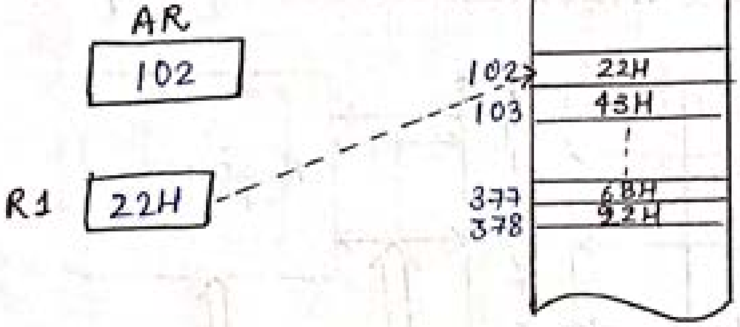


R1 ← M[AR]

Memory

M[AR] = Memory location specified by the address register

**Memory WRITE**:- A memory write operation simply write data from register to a memory location specified by the address register.



Memory

## Processor Organization

- The processor consists of ALU, CU and registers.
- The processor organization refers to how the components within a CPU are structured and how they interact to perform the operation.
- A processor does the following things:-

ⓐ **Fetch Instructions**:- The processor reads an instructions from memory (main memory, register, cache memory).

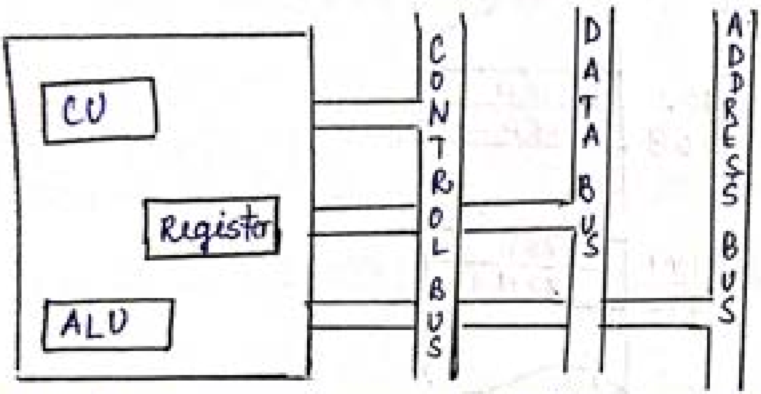
ⓑ **Interpret Instructions**:- Instruction is decoded to determine what action is required.

ⓒ **Fetch Data**:- The execution of an instruction may require reading data from memory or an i/o module.

ⓓ **Process Data**:- The execution of an instruction may require performing some arithmetic or logical operation on the data.

ⓔ **Write Data**:- The result of an execution may require writing data to memory or i/o module.

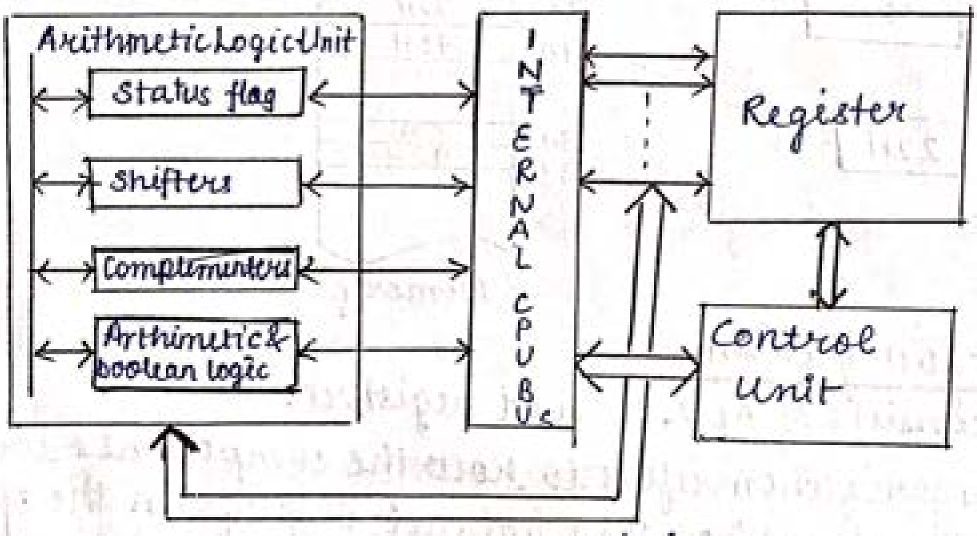
Following figure shows a simplified view of a processor indicating its connection to the rest of the system via system bus.



ALU does the actual computation of processing the data.

- The control unit control the movement of data & instructions into and out of the processor and controls the operation of the ALU.
- It also consists of minimal internal memory consisting of a set of storage locations called registers.

Following figure shows the more detail view of the processor.



The data transfer and ~~othrcontrol~~ logic control path are indicated by the internal processor bus.

- It is required to transfer data between the various registers and the ALU. Because the ALU operates only on the data in the internal processor memory.

Types of processor Organization

Processor organisation can be categorised into different types depending on the how instructions are executed and how the processor interacts with memory.

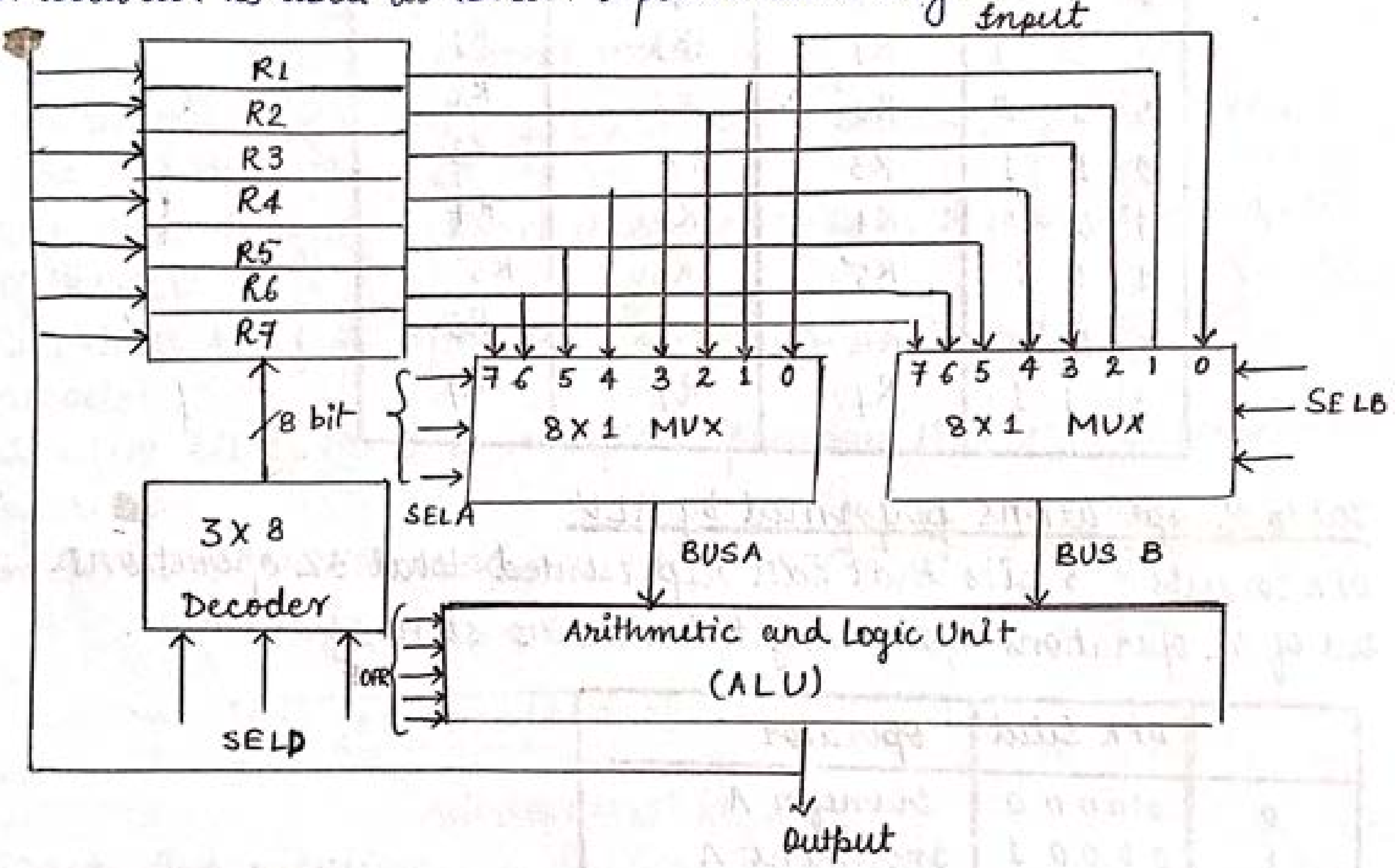
1) General purpose Register Organisation
2) Accumulator based Organisation

3) Stack based organisation
1) Load-Storage organisation

## General Register Organization

In this organization the registers communicate with each other not only for direct data transfer but also used ~~by~~ while performing various micro operations. Generally CPU has seven general registers following figure show the register organization how register are selected and how data flow between register & ALU.

A decoder is used to select a particular register.



| 3 | 3 | 3 | 5 |
|---|---|---|---|
| SEL A | SEL B | SEL D | OPR |

Control word

The output of each register is connected to multiplexers from buses A and B. The selection line in each multiplexer selects one register or input for the particular bus.

- The buses A & B from the input to a common ALU the
- The operation selected in the ALU determine the arithmetic or logic micro operation that is to be performed is done by operation

select line.

- The result is available for output data and goes into the input of all registers.
- Register for output is selected by a decoder.
- The decoder activates one of the register load inputs.

## Register Selection

| Binary codes | SEL A | SEL B | SEL D |
|---|---|---|---|
| 0 0 0 | Input | Input | Minu |
| 0 0 1 | $R_1$ | $R_1$ | $R_1$ |
| 0 1 0 | $R_2$ | $R_2$ | $R_2$ |
| 0 1 1 | $R_3$ | $R_3$ | $R_3$ |
| 1 0 0 | $R_4$ | $R_4$ | $R_4$ |
| 1 0 1 | $R_5$ | $R_5$ | $R_5$ |
| 1 1 0 | $R_6$ | $R_6$ | $R_6$ |
| 1 1 1 | $R_7$ | $R_7$ | $R_7$ |

## Table of operations performed by ALU

OPR consists of 5 bits that can represented total 32 operations. Out of 32 operations following table shows some of them.

| | OPR Select | Operator |
|---|---|---|
| 0 | 00000 | Transfer A |
| 1 | 00001 | Increment A |
| 2 | 00010 | Add A + B |
| 5 | 00101 | Subtract A - B |
| 6 | 00110 | Decrement A |
| 8 | 01000 | AND A, B |
| 10 | 01010 | OR A, B |
| 12 | 01100 | XOR A, B |
| 14 | 01110 | Complement A |
| 16 | 10000 | Shift Right A |
| 24 | 11000 | Shift Left A |

## control word

The combined value of binary selection input specifies the control word

- It consists of four fields -

-SEL A - select A
-SEL B - select B
-SEL D - select D

that contains three bit each and ORR field that contains 5 bit therefore the total bits in the control word are 14 bits.

| 3 | 3 | 3 | 5 |
|---|---|---|---|
| SEL A | SEL B | SEL D | OPR |

control word

- The three bit of (select A) SEL A select a source register of the first input of the ALU.
- The three bit of SEL B select a source register of the second input of the ALU.
- The three bit of SEL D select a ~~source~~ destination register by using the decoder.
- The five bit of the OPR select the operation that is to be performed by ALU.

For Ex :- The control word for the following operation is given as:-

$$R_2 \longleftarrow R_1 + R_3$$

| 001 | 011 | 010 | 00010 |
|---|---|---|---|
| R1 | R3 | destination ($R_2$) | operation - A+B |

## Stack Organisation

A stack is an ordered set of elements only one of which can be accessed at a time.

- Basically a computer system follows a memory stack organization.
- A portion of memory is assigned to a stack operation to impliment the stack in CPU.
- A stack is a storage device that stores information in such a manner that the item stored last is the first out from stack known as LIFO ~~manner~~ order (Last in first out).

- In the computer, stack is a memory unit with an address register that can count the address only.
- The register that holds the address for the stack is called stack pointer (SP), it always points at the top items or element of the stack.
- A stack is organization allows for efficient management of data and return address during program execution.

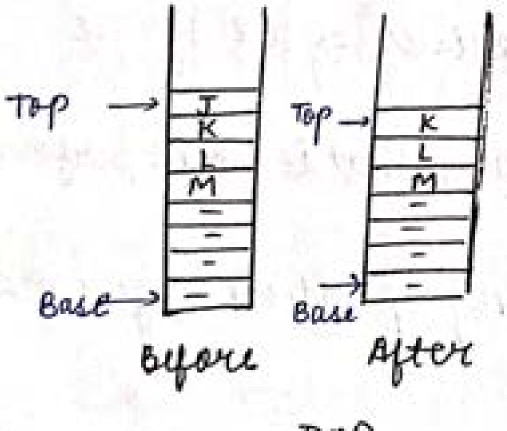
Operations

There are two operations in stack

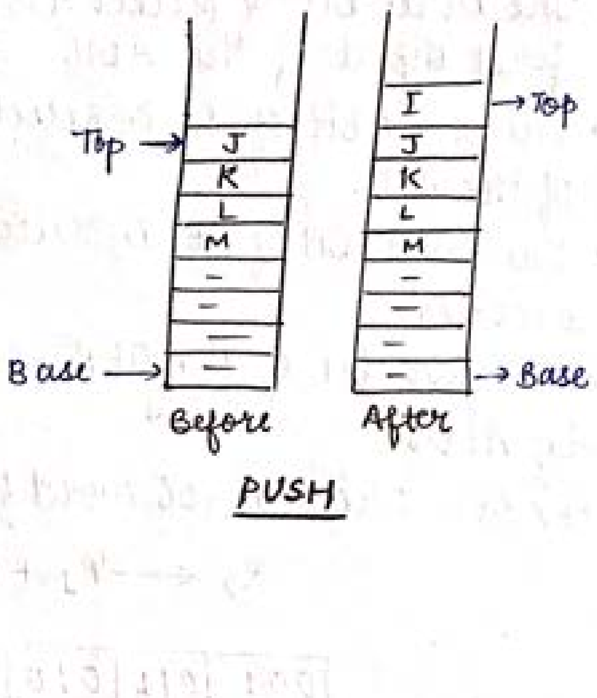
① PUSH ② POP

① PUSH :- To insert/add an element to the top of the stack

② POP :- In pop operation it removes or delete an element from the stack.



POP



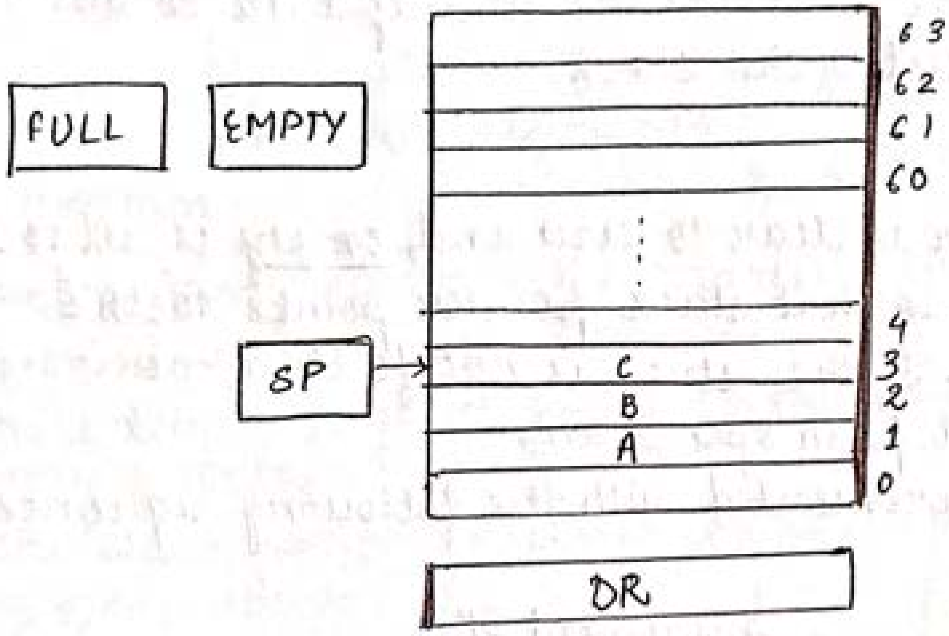
PUSH

Types of Stack

① Register stack ② Memory stack

Register stack :- A stack can be ~~placed~~ placed in a portion of the large memory or it can be organised as ~~a~~ ~~to~~ collection of a finite number of memory words or registers.

- Following figure shows a Organization of a 64-word register stack



[SP - Stack pointer]

- The stack pointer register contains a binary number whose value is equal to the address of the word is currently on top of the stack.
- Three items are placed in the stack A,B,C in that order.
- In the above figure C element is on the top of the stack so that the content of SP is three.
- for removing the top item the stack is poped by reading the memory word at address 3 and dicrement the content of stack SP. Now the top of the stack is B so that the content of SP is 2.
- similarly for inserting the new item the stack is pushed by incrementing SP and writing a word in the next higher location in the stack.
- In a 64-word stack the stack pointer contains 6 bits $(2^6)$.
- Since SP has only 6 bits therefore it cannot exceed a no greater than 63
- When 63 is incremented by 1 the result is zero since.

$$\begin{array}{r} 1\ 1\ 1\ 1\ 1\ 1 \\ +\ 1 \\ \hline \underline{1}\ 0\ 0\ 0\ 0\ 0\ 0 \\ \hline \end{array}$$

[LSB - Least significant bits]

The SP can accomodate only 6 LSB bits.

- Then one bit register FULL is. set to 1 when the stack is full.
- Similarly when the stack pointer is 0 0 0 0 0 0 and decremented by 1 then 1 bit register Empty is set to 1.

DR is a data register that ~~for~~ holds the binary data to be written into or read out of the stack.

Push operation

Initially the stack pointer is clear to zero and, empty is set to 1. and full is clear to 0. so that stack pointer points to the word at address zero. If the stack is not full a new item is Inserted with a push operation.

- The PUSH operation is incremented with the following sequence of microoperation

$SP \leftarrow SP + 1$     Increment SP

$M[SP] \leftarrow DR$     write item on top of the stack

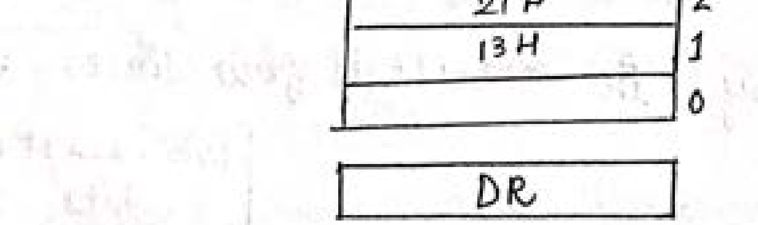
The stack pointer is incremented so that it points to the address of ~~the~~ next higher word.

- A memory write operation inserts the word from data register to the top of the stack.

POP operation

A new item is deleted from the stack if the stack is not empty

- A pop operation consists of the following sequence of microoperation



Read item from the stac

$DR \leftarrow M[SP]$

$SP \leftarrow SP - 1$

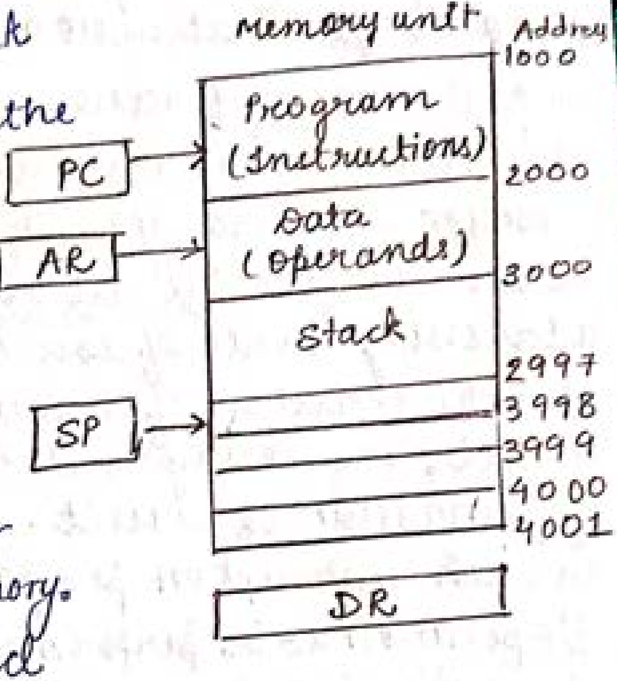
Decrement SP

The top item is read form the stack into Data Register the stack pointer is then decremented

Memory stack

Following figure shows a portion of computer memory partition into three segments

1) Program 2) Data 3) Stack



- The program counter at the address of the next instructions in the program.
- The address register points at an array of data.
- The stack pointer SP points at the top of the stack.
- All these three registers are connected to a common address bus and either one can provide an address for memory.
- PC is used during fetch cycle to read an instruction.
- AR is used during the execute phase to read an operand.
- SP is used to push or pop the elements into or from stack.
- The initial value of SP is 4001 and stack goe grows with decreasing the addresses.
- The elements in the stack communicate with a data register.
- A new Item is inserted with the push operation as follows.

$$\left.\begin{array}{l} SP \leftarrow SP - 1 \\ M[SP] \leftarrow DR \end{array}\right\} PUSH$$

- A memory write operation inserts the word from data register into the top of stack.
- A new Item or element is deleted with the pop operation as follows:-

$$\left.\begin{array}{l} DR \leftarrow M[SP] \\ SP \leftarrow SP + 1 \end{array}\right\} POP$$

- The top item is read from the stack into the data register.
- The stack pointer is then decremented to point at the next item in the stack.

## Addressing Modes

The term addressing modes refers to the way in which the operand of an instruction is specified.

- To perform any operation using microprocessor we have to give the corresponding instructions for executing this instruction processor will need 8/16 bit data Therefore the method by which we have to give the address of source of data is known as the addressing mode of source.
- After the execution of insertion the processor will obtain the result, for storing this result we have to give the address of destination of result.
- In each instruction program has to specify three things -
  a) operation to be performed
  b) Address of source of data
  c) Address of destination of result.

Types of Addressing modes

① Immediate Addressing modes
② Direct addressing modes
③ Register direct addressing Mode/Regular Addressing Mode
④ Register Indirect addressing mode
⑤ Implicit /Implied addressing Mode

### Immediate Addressing Mode

- If 8/16 bit data required for executing the instruction is directly given along with instruction then such type of instruction are called immediate addressing mode instructions.
- In most of the immediate addressing mode instruction the last alphabet is I

For Example:- ① MVI A, 75H [Move 75H in accumulator]

- MVI is the operation.
- 75H is the immediate data (source).
- A is the destination

② LXI B, 4565H [16 bit]

③ MVI M, 30H

It move 8 bit immediate data 30H into the memory location pointed by HL register pair

## Direct Addressing mode.

If 8/16 bit data require for executing the instruction is present in memory location and the address of memory location is given along with the instruction then such type of instruction are called direct addressing mode Instruction.

For Example ① LDA 2500 H

- Load Accumulator from the given address data.
- LDA is the operation
- 2500 is the address of source.
- Accumulator is the destination.

[HL stores memory address]

② STA 4065H

Store accumulator at given address

[Removes the data from accumulator and tansfer to memory]

## Register Direct Addressing Mode.

If 8/16 bit data is required for executing the instruction present in registor for 8 bit & for register pair (16 bit) and the name of the registor/register pair is given along with the instruction then such type of instructions are called register direct addressing mode.

Ex:-

MOV C, D. [Moves the data from D to C registers]

## Register Indirect Addressing Mode.

If 8/16 bit data require for executing the Instruction present in memory location and the address of that memory location is present in register pair & the name of that register pair is given along with instruction then such type of instruction are called register Indirect addressing mode instruction.

Ex:- MOV A, M

- Move data from memory location specified by HL pair to accumulator.
- MOV is the operation, • M is the memory location specified by HL register pair

- A is the destination.

## Implicit Addressing Mode

If the address of source of data as well as the destination of result is fixed then there is no need to give the operand in the instruction and such type of instructions are known as implicit addressing mode instruction.

for Ex:- ① CMA (Complement the accumulator) - 1's complement
② CMC (Complement carry flag)
③ STC (set carry flag)
④ RAL (Rotate accumulator left with carry)
⑤ RLC (Rotate accumulator left without carry)