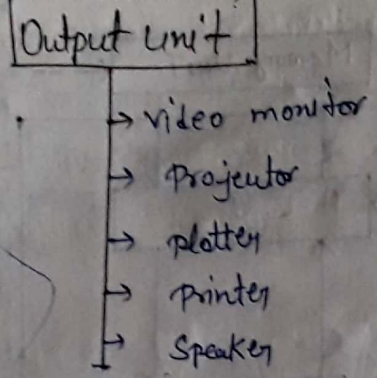
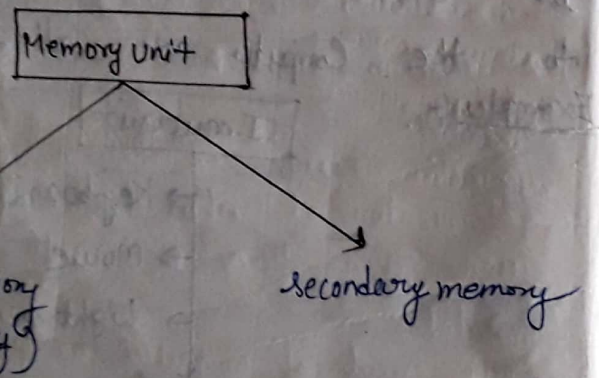


Memory Unit

⇒ The memory unit is used to store the program or data.

There are two types of memory devices are used in Computer System.



⇒ Primary Memory ⇒ The primary memory is called as main memory. the programe and data stored in main memory at the time of an execution.

⇒ The main memory is a semiconductor memory.

⇒ It consist of the large no of semiconductor cells which is capable of storing one bit of information.

⇒ These cells are group to store a word of information.

⇒

Secondary memory ⇒ Secondary storage devices are used to store large amount of data.

⇒ Example ⇒ Compact, disk, flopy disk, Hard disk., magnetic tape.

Arithmetic & Logical Unit ⇒ The Arithmetic & Logic Unit (A.L.U) is responsible for performing arithmetic operations such as (+) addition, substraction (-), multiplication (*), division & logical operations (AND, OR, Negation(!), ~) etc. to perform these operations operants from the main memory are brought into the high speed storage elements called Registor of the Processor.

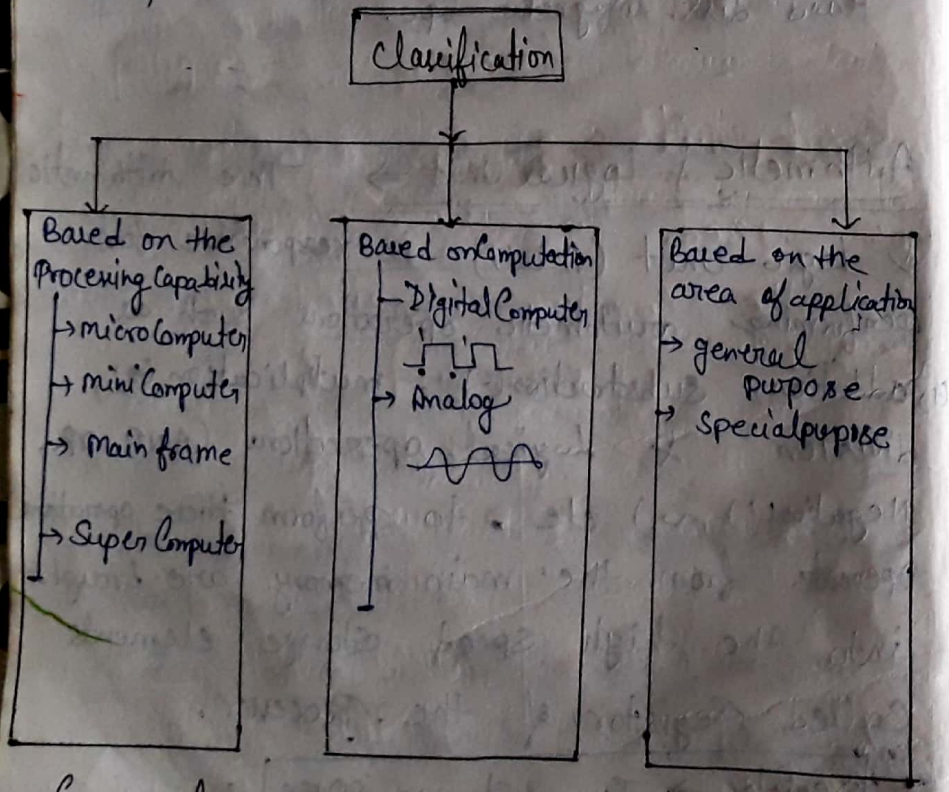
Registor → Temp-Storage space

Control Unit ⇒ The control unit coordinates and control the activities among the functional units. the basic function of

Control unit fatch the instruction from the main memory, identifie the operations and the devices involved in it and generate control signals.

Computer Types & Clasification

Computer are clasified in to 3 Categary



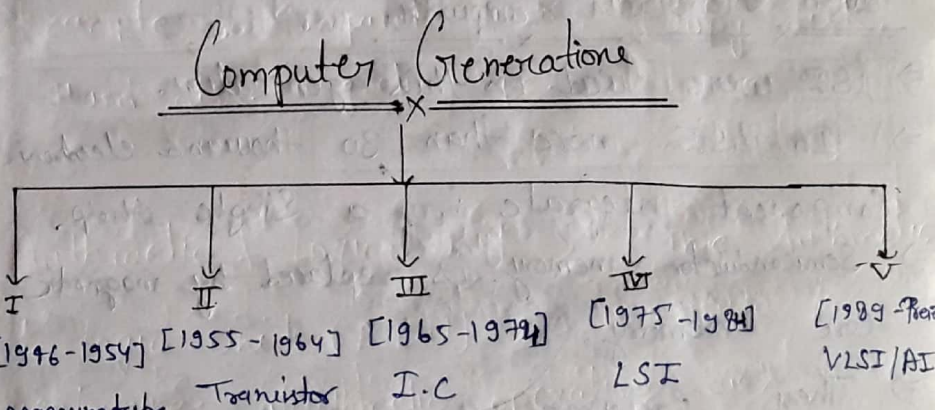
General purpose ⇒ The computers that are used for any type of appliation are called general pupose Computors.

⇒ general Computer are used to solve abstract problem or can solve mathmetical equations.

Special purpose Computers ⇒ The Computers that are used for specific purpose are known as Special purpose Computer.

Ex :— Computer used in weather forcasting, Rocket launching,

Computer Generations



First generation Computer

⇒ First generation Computer works were to bulky insize hence required large room for installation.

⇒ They used thousand of vaccum tube hence can assume more power.

Second generation

⇒ Capacitor and tranistors are used instead of vaccum tubes.

⇒ They were much smaller in size.

⇒ They were less expensive to produce.
⇒ They have less hardware failure.

Third generation Computer

⇒ They consume less power and emit less heat.
⇒ They have faster and larger memory made of semiconductor.

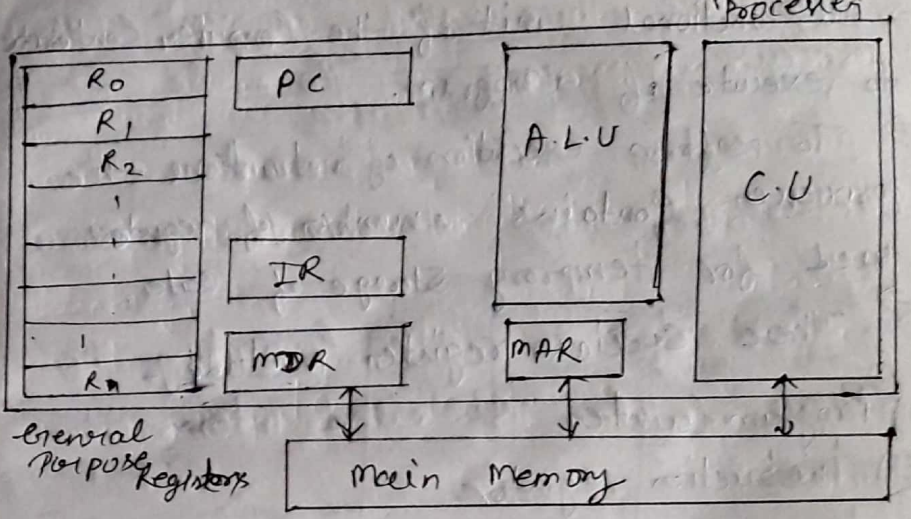
Fourth Generation Computer

⇒ LSI were used in this generation.
⇒ In LSI more than 30 thousand electronic component integrate in a single chip.
⇒ Semiconductor memory is replaced by magnetic core memory.
⇒ They consume less power and more reliable.

fifth generation Computer

⇒ In this generation VLSI and ULSI technology are used.
⇒ They have faster and larger storage memory.
⇒ Laptop, Notebook, Smart phones are much smaller and handy to allow user to use Computing while travelling

Basic Operation Concept :-



PC — Program Counter
IR — Instruction Register
MDR — Memory Data Register
A.L.U — Arithmetic & logical unit
C.U — Control unit
MAR — Memory address register

⇒ The Basic function is to execute program instruction are store in the Computer memory.
⇒ These instruction are executed to process data which is loaded input Computer memory throw input unit
⇒ After Processing the data the result is either store back into the Computer memory for further refrence

⇒
or it is send to outside word throw the output unit
⇒ All functional unit of the Computer Contribute to execute of a program.
⇒ To perform execution of instructions the processer Contains a number of registors used for temprory storage of data.
⇒ There special register include -

(1) Program Counter
(2) instruction registor
(3) memory addres registor
(4) memory data register.

Program Counter (P.C)

⇒ A program is a series of instructions store in the memory.
⇒ There instructions tell the C.P.U how to get the desired result.
⇒ It is important that there instructions must be executed in a proper ~~model~~ order.

to get Correct result.

⇒ The sequence of instructions is monitored by the program Counter.

⇒ Keep track of which instructions is being executed and what will be the next one.

Instruction Registers (IR) :—

⇒ It is use to ~~hold~~ the instructions that is currently executed.

memory data register (MAR) :—

⇒ There registers are use to handle the data Transfer between the main memory and the processer.
⇒ The memory address register hold the address of main memory to or which from data is to be transfer.

Memory Data Register (MDR) :—

⇒ It is some time calls memory buffer registers Contains the data to be written (~~return~~) into or read from the address word of the main memory.

General purpos registers (G.P.R) :—

⇒ There registors are used for whall the operant from arithmatic & logic operations.
⇒ Thae registors are also use for frequently data storage.

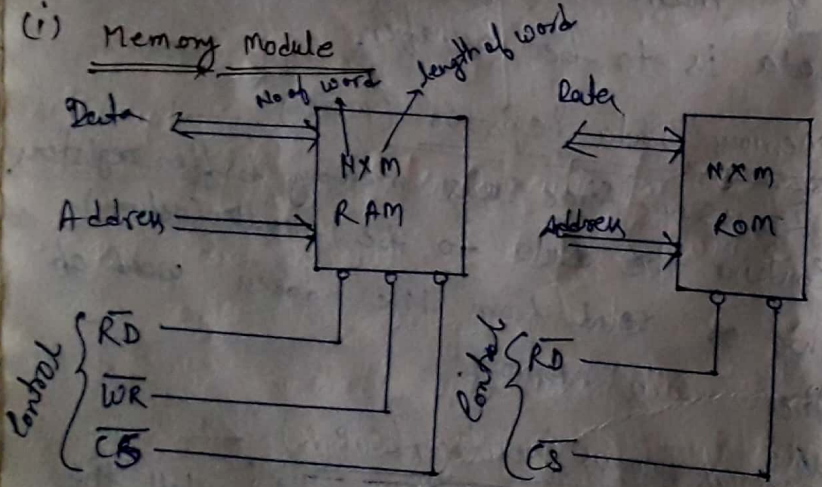
Inter Connection Structure :-

⇒ There are three basic Component (I/O) Procesor, memory & I/O module, of a Computer System that Communicate with each other are interconnected in various Path.

⇒ a Collection of Path Connecting the various module is Called interconnection Structure.

⇒ Thre module are disscus as follow :-

(1) Memory Module



⇒ Memory is specified as N x M where 2^N is the no. of words that can be stored & M is the length of words

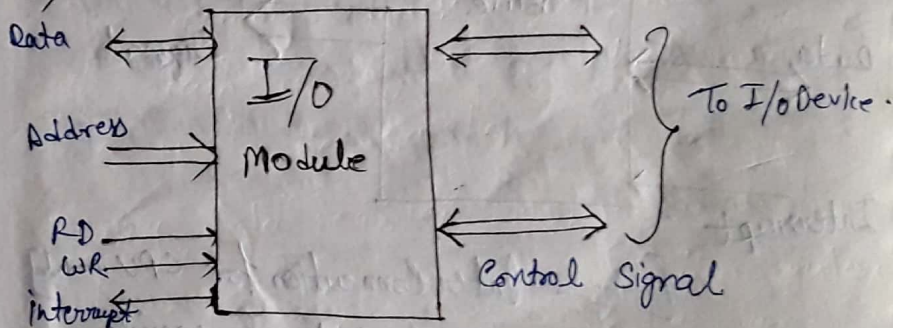
⇒ Each word is an array with an unique address (0, 1, 2, ----- 2^N)

⇒ with the help of Control Signal (RD, WR & CS) memory read or write operation can be indicated

⇒ for ROM WR signal is not required.

(2) I/O Module

⇒



[fig :- Interconnection Structur for I/O module]

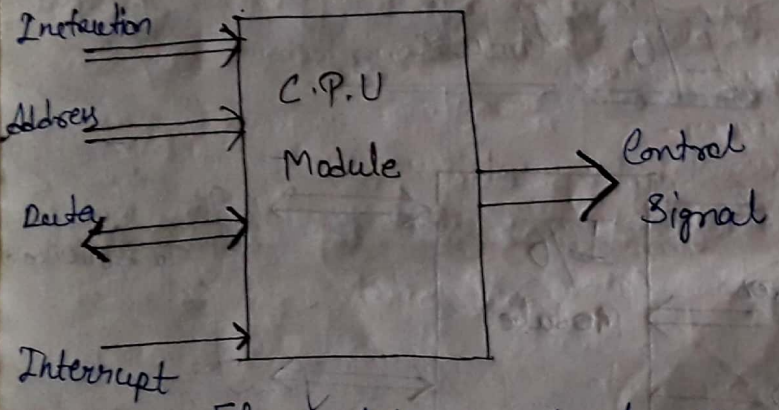
⇒ The CPU read Data from input Device and write Data input Output Devices and each device has an unique address.

⇒ The input/output operation off I/O Module is Contol by read (RD) write (WR) (CS).

⇒ I/o module can send interrupt signal to the C.P.U.

(3) C.P.U Module

⇒



[fig:- Inter Connection for CPU module]

⇒ C.P.U Read instruction & Data & write Data after processing, generate CS to Control the overall operations of the System.

⇒ It also recieves interrupt signal from the i/p output module.

[Buses] (Architecture)

⇒ A group of wires called bus.

⇒ It is used to transmition information. it used to necessary signal for Communication b/w modules.

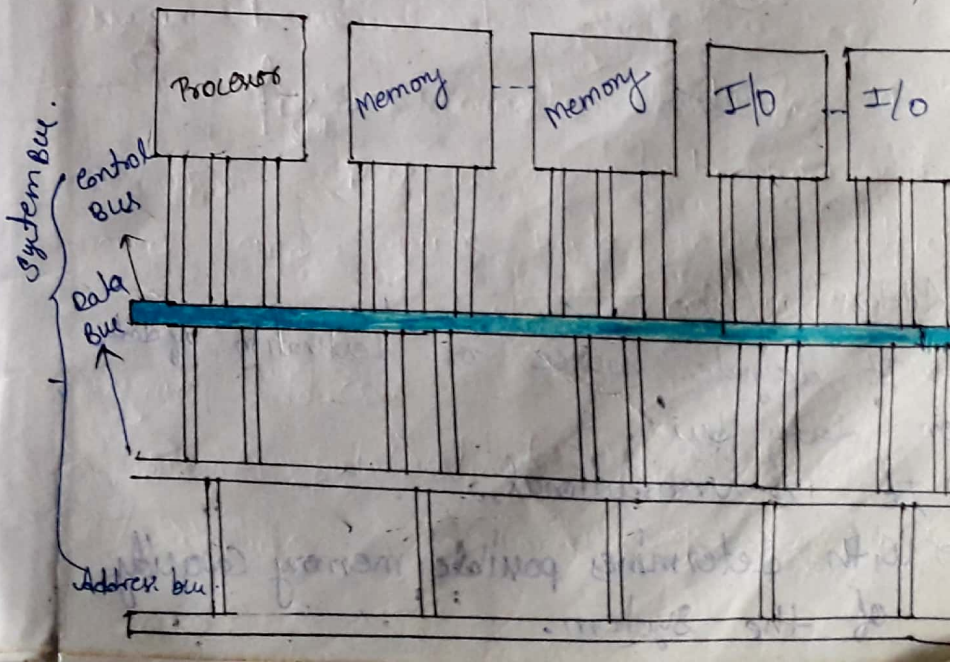
⇒ A bus is a shared transmition medium, must only we used by one device at a time.

⇒ when we to Connect major Computer Components

Ex - C.P.U, Memory - is called a System bus

⇒ a System bus is seprated into functional groups (i.e) Data bus, address bus, and Control bus.

[fig:- Bus Interaction Scheme]



Data Bus/line :-

⇒ Move Data between System modules, The data bus lines are bidirectional (i.e) C.P.U can read data on these lines from memory from memory or from port as well as send data out on these lines to way memory location or to a port.

⇒ with is a key factor. it determines no of bytes that can be transfer in one cycle and hence the overall System performance.

Addres Bus/line

⇒ It designate source or destination of data on data bus.

⇒ It is unidirectional.

⇒ With determines possible memory capacity of the System.

⇒ It is also used to address input/output port.

⇒ The higher order bit select a particular module and lower order bit select a memory location. for input output port within the module.

fo

Control Bus/line

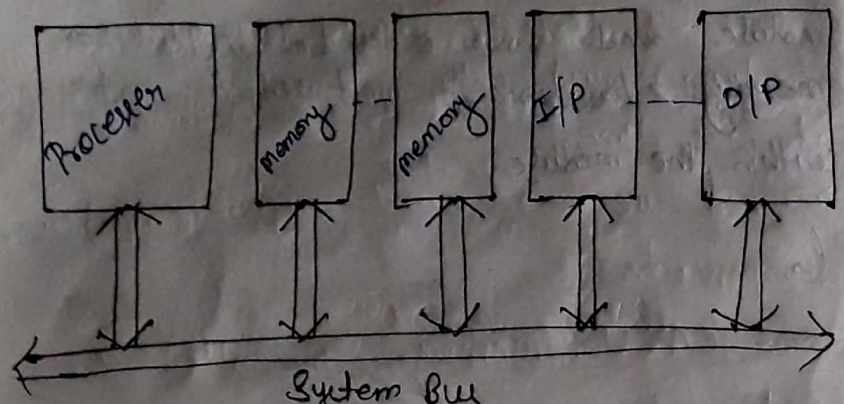
⇒ Control access to and use of data and address lines.

⇒ Contral lines input

- Memory read & memory write
- I/o read & I/o write
- Transfer Ack
- Bus request and bus grant
- Interupt request and interupt Ack
- Clock
- Reset.

According to Structure :- There are two types of bus :-

(1) Single Bus Structure:-



[fig:- Single Bus]

⇒ Data bus, address bus, Control bus are Represented by a single bus. hence such interconnection bus Structure is called Single bus Structure.

⇒ In a Single bus Structure all uniques are Connected to Common bus called system Bus.

⇒ However. with single bus Only two units Can Communicate with each other. at a time.

⇒ The bus Control line are used to are ~~Orbidoxy~~ ~~Category~~ multiple request for use of the bus.

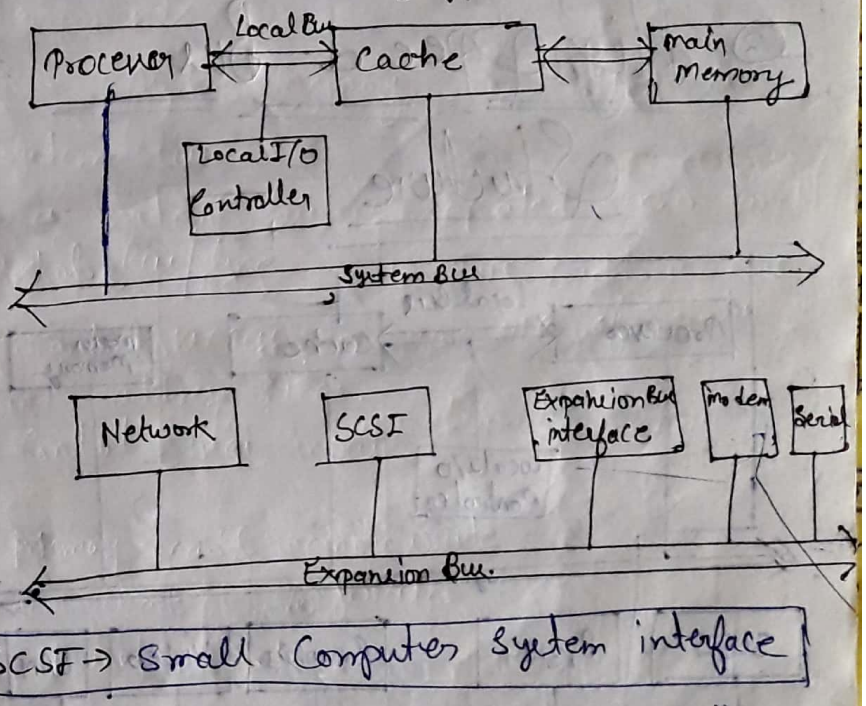
⇒ Advantage of single bus structure :-

① It is low cost.

② Flexibility for any Peripheral devices.

② Multiple Bus Structure

[fig:- Traditional Bus structure]



SCSI → Small Computer System interface

⇒ The Traditional Bus Connection uses three busses. Local Bus, System Bus, Expansion Bus.

⇒ The use of Cache Structure provide C.P.U. to frequent access to memory.

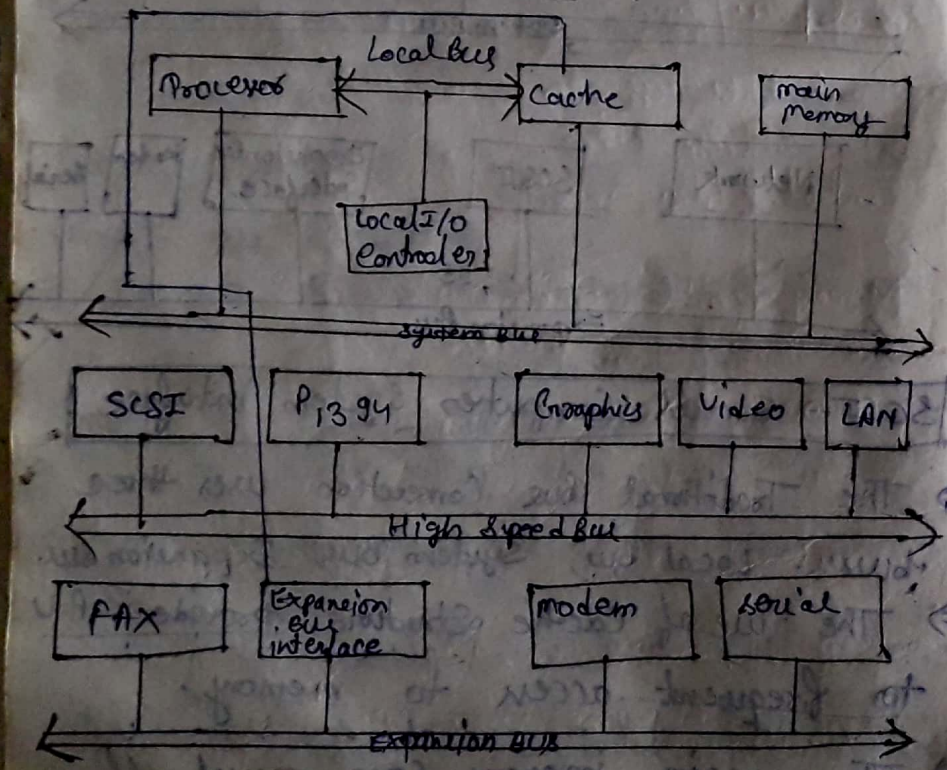
⇒ The main memory can moved off local bus to a System Bus.

⇒ The Expansion bus interface -

① Buffer Data transfer between System and I/O controller on expansion bus.

② Insulate memory to processor traffic to I/O Traffic.

③ High Speed Bus Structure



⇒ It is specifically design to support High Capacity input Output devices.

⇒ It uses High speed bus along with the three buses we in the traditional bus Connection

⇒ Cache Controller is use to a Provide high speed with the speed bus.

⇒ This Bus support Connection to high speed lan (local area Network) such as FDDI (Fiber distributed data interface), video & graphics workstations Controller as well as interface Controller to local peripheral Devices including SCSI (Small Computer System interface.) & P1394

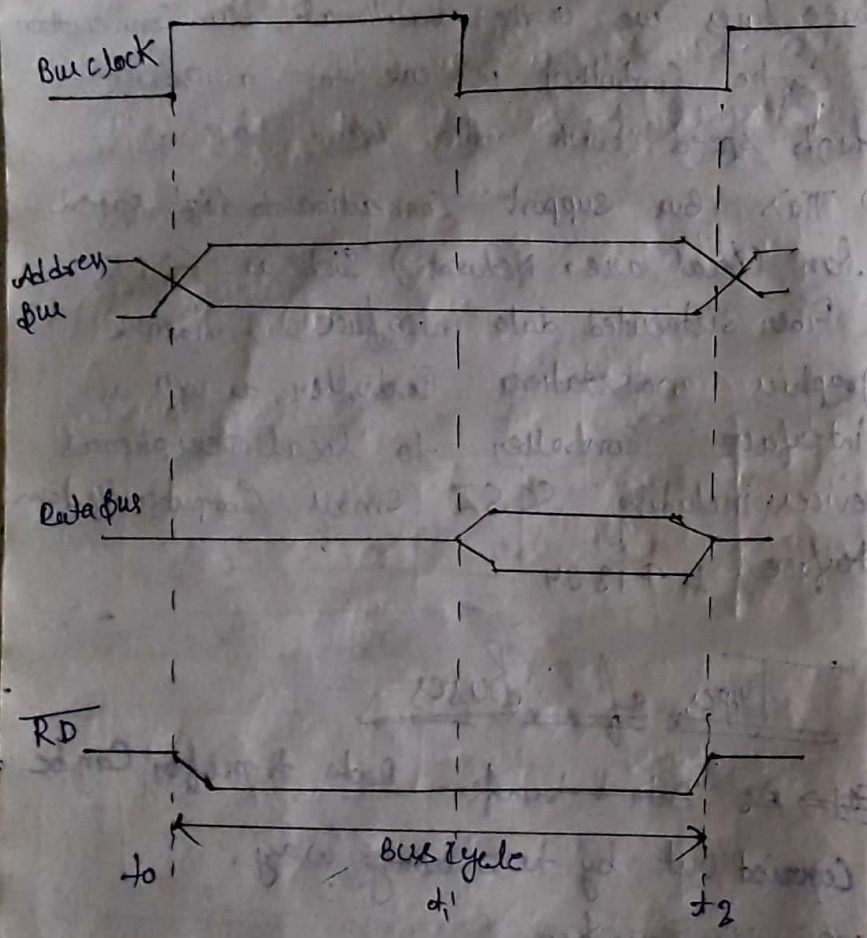
Types of Buses

Data transfer Can be Carried out by two ~~ways~~ way.

(1.) Synchronous bus

(2.) Asynchronous bus

(1) Synchronous Bus ⇒ At time t=0 that t node this processor place the device address on the place line address line of the system bus & set the Control line, to perform I/O operation.

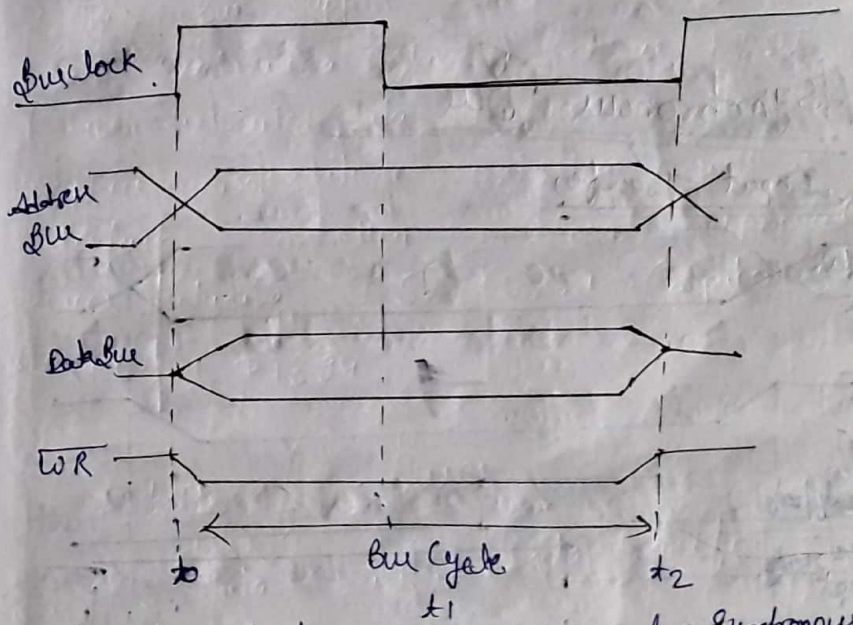


[(a) :- Input :- Timing Diagram for Synchronous Input transfer]

⇒ during time t node address device get address & it recognized that input operation is requested.

⇒ At time t_1 address devices Caes its Data on the Data bus.

⇒ At the end of the Bus Cycle that is, at time t_2 the processor reads the data line and load the data from Data bus into its input Buffer.



[(b) for output :- Timing Diagram for Synchronous O/p transfer]

⇒ Processor places Output Data on the Data line when it transmit the address end Control line information.

⇒ at time t_1 the address device reads the data line and load the Data into Data Buffer

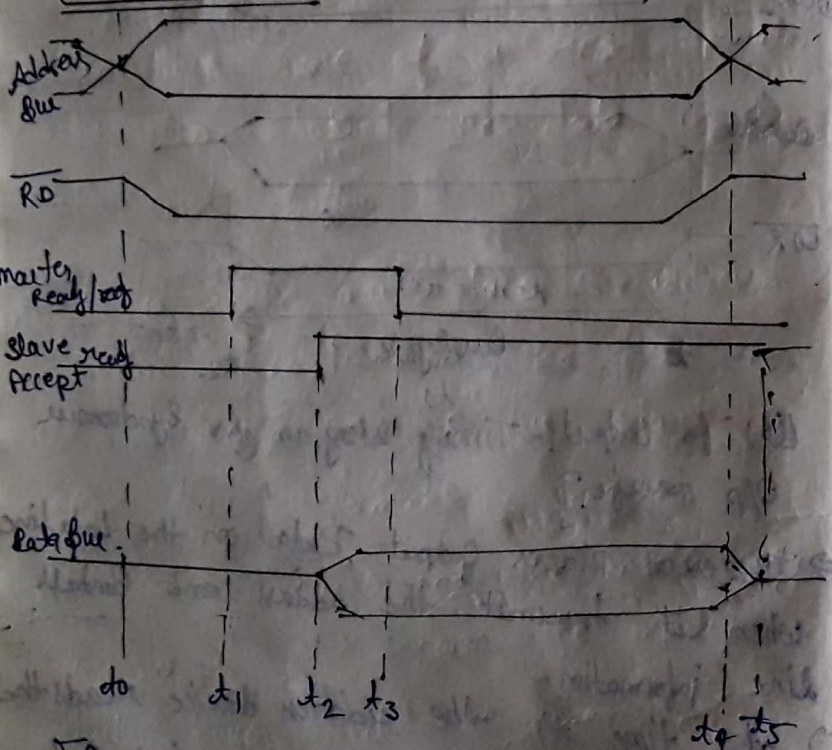
⇒ The processor simply assume that at time t_2 the Output Data have been recieved by the I/O Devices.

And cycle end.

NOTE ⇒ In Synchronous Bus all device drive timing information from a common clock signal

Asynchronous Bus

Input Transfer



[Fig:- Timing Diagram for Asynchronous input transfer]

⇒ In Asynchronous Bus a common clock is eliminated and the data

Transfer on the system Bus is achieved by the handshake between the processor & device being address. here the clock is replace by two control signal ready and accept.

⇒ The Data transfer on the bus is based on the use of handshake between master & slave.

⇒ The common clock is replaced by to timing control line (i.e) master ready & slave ready.

⇒ The first is accedrate to indicate that it is ready for dimension an second is for the response from the slaved

⇒ In principle. a Data transfer controll by a handshake Protocal Proceed as follows -- the master place & address Command information on the bus than it indicate to all devices than it has done so by activatie the master ready line.

⇒ This Causes all devices on the Bus to decode the address

⇒ The selected slave perform the required operation & inform the processor that it has done so by activating Slave Ready line.

⇒ The master wait for slave ready to become a certain before it removes its signal from the Bus.

⇒ In this case of a read operation it also strobes the data input its input buffer.

At t_0 ⇒ The master places the address & Command information on the bus and all devices on the bus start to decode this information.

At t_1 ⇒ A master set the master Ready line to 1 to inform the input/output devices that the address & Command information is Ready.

At t_2 ⇒ The Selected slave having decoding the address and Command information perform the required

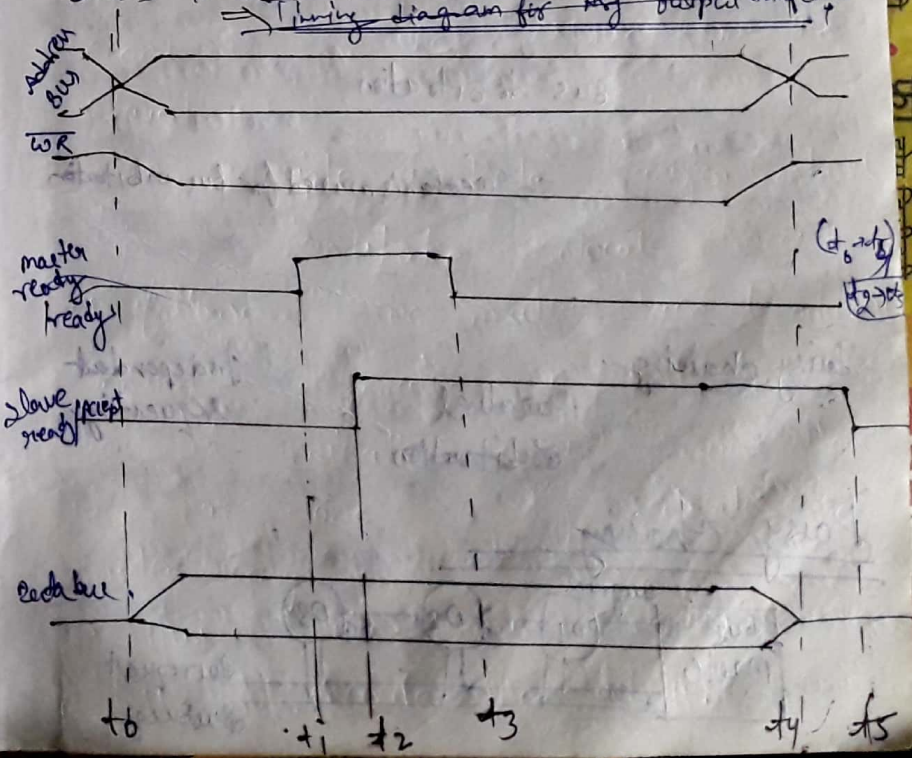
Input operation by placing the data from its data request on the data Bus line.

At t_3 ⇒ The slave ready signal arrived at t master, indicating that the input data are available on the Bus.

At t_4 ⇒ A master remove the address & Command information from the bus.

At t_5 ⇒ when the devices interface recieve the 1 to 0 transaction of the master ready signal it removes the data and slave ready signal from the bus. This Complete the input transfer.

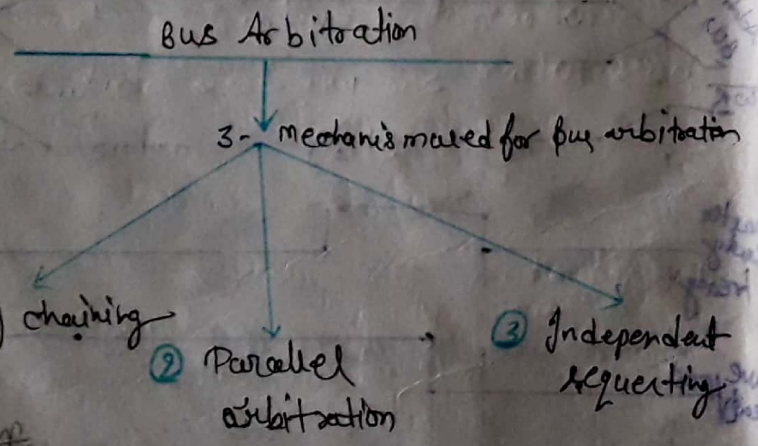
⇒ Timing diagram for Any output transfer:



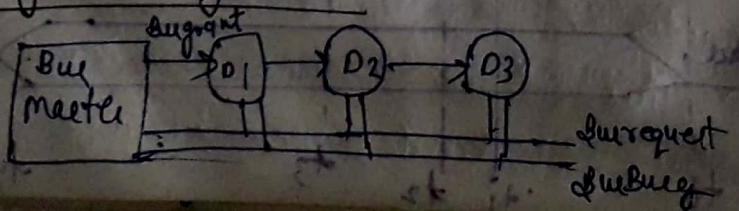
Bus Arbitration

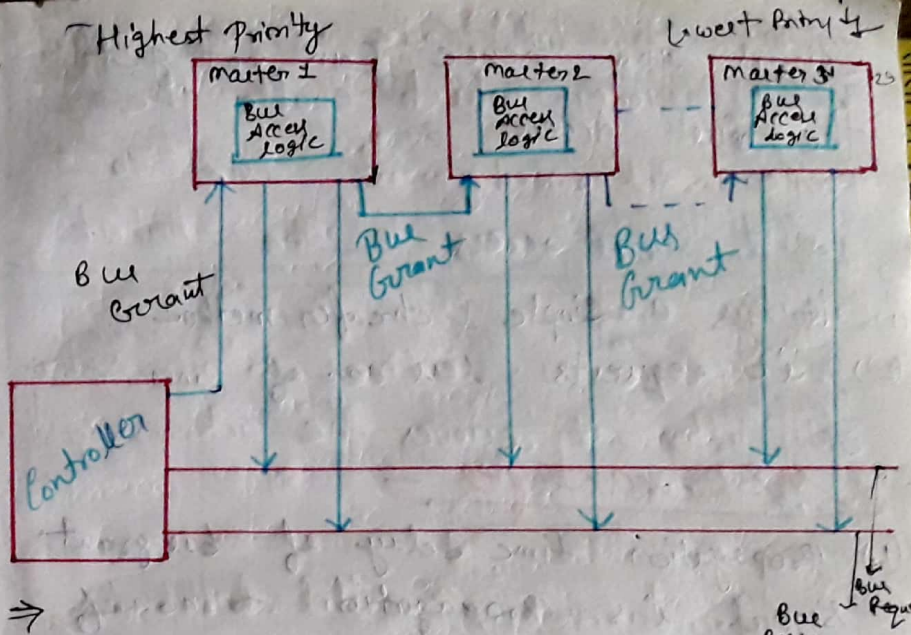
⇒ It is a machanism which decides the Selection of Current master to access the bus is Knownas Bus arbitration.

Bus Master — The device than allow to inciate Data transfer On the Bus at any given time is Called Bus Master.



Daisy chaining





⇒ Daisy chaining metho d is Simple & achieved method.

⇒ all matere make use of Same line for Bus request

⇒ The bus Grant Signal Serially propagate (propogate) through each master until it an Counter the first one that is requesting access to the bus,

⇒ This Master Blocks the propagation of the bus grant Signal, activates the Busy line. & Controls of the bus.

⇒ Therfore any other requesting module will not recieve the grant

Signal and hence can not get the bus Access.

Advantages :-

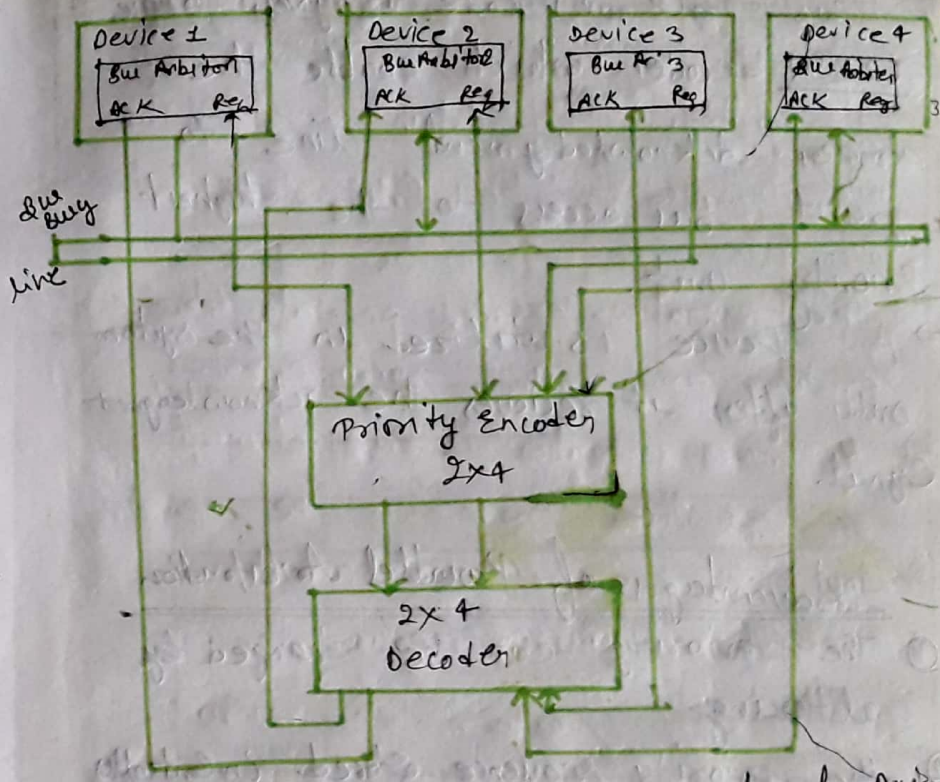
(1) it is a simple & cheafer method
(2) it requests less no. of lines.

Disadvantages :-

(1) Propagation time delay of bus grant signal is propagation to no. of masters in the System.
(2) This makes arbitration time slow.
(3) The priority master is fixed by physical location of master.
(4) failure of any one master causes whole System to fail.

Parallel Arbitration

=> It Consist of Priority Encoder & Decoder. In this machanism each bus arbitor has a bus request line and a Acknowledgment line
=> each arbiter enable the request line when its device [Processor]



[Fig:- Parallel Arbitration for four Device]

=> each requesting for the access to the System bus.
=> Each arbiter bus request output line is Connected to the input of Priority I/Input ~~Input~~ Encoder.
=> The output of the Encoder generates ~~bit~~ two code which represents the highest priority among those requesting the bus.

⇒ This to(2 bit) code is given to the (2×4) decoder which enable the Proper Acknowledgment line to grant bus access to the highest Priority Unit.

⇒ A Device is utilized in the system only after it recieves the acknowledgment Signal.

Advantages of Parallel Arbitration

① The priority can be changed by altering

② the priority sequence stored in Controller

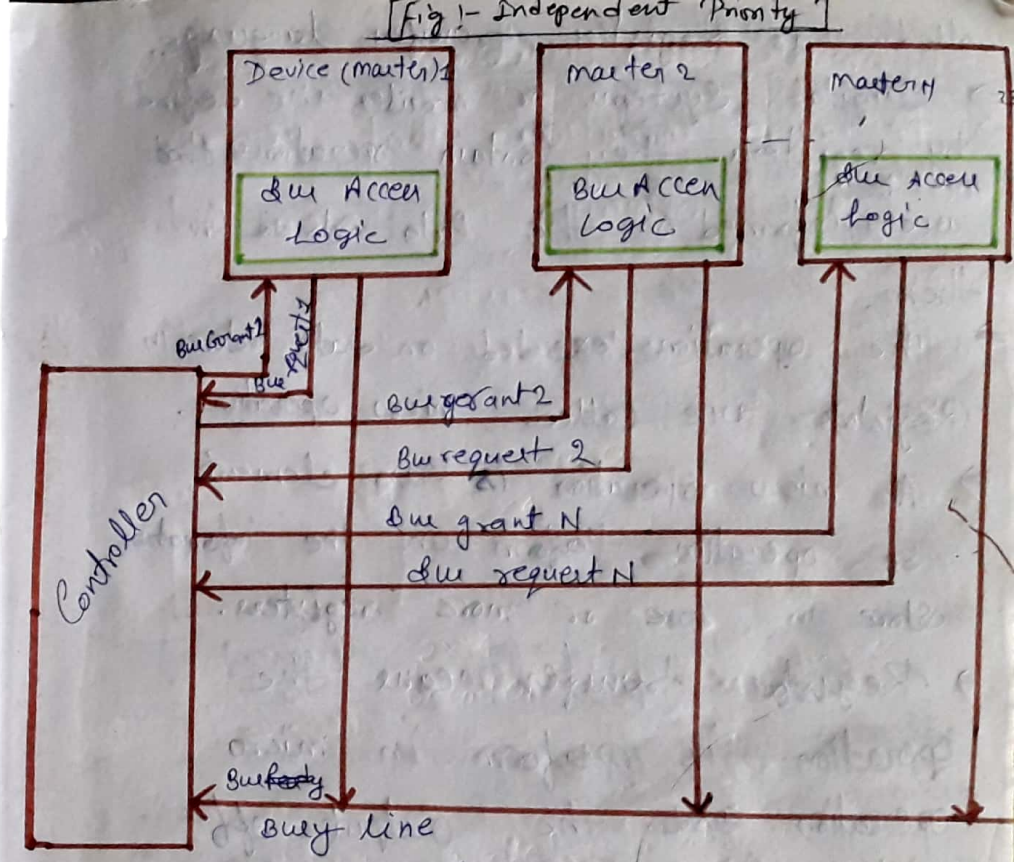
③ if one module fails entire System does not fail.

Independent Priority

⇒ In this scheme each master has a Seprate Pair off Bus request and Bus grant lines.

⇒ And each pair has a priority sign to it the build in priority encoder within the Controller select the highest priority and sign Corresponding

[Fig :- Independent Priority]



bus grant signal.

B= { Register, Bus, Memory } Transfer

→ Register :- There are the fast Processing unit we to Store temporary Information.

Register
↓
Cache
↓
MM
↓
mag. Disk
↓
mag. tape.

Register Transfer Language :- The Symbolic Notation we to describe the operation transfers among the registor is

called as Register tranfer language.

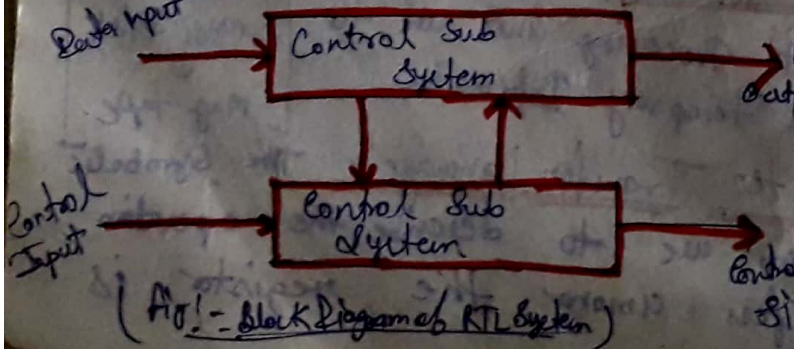
⇒ Digital System or modules are defind by registers they Contain operations that are performed on the Data Stored in them.

⇒ The operations executed on data store in Registor are called micro operations.

⇒ A micro operation is an element be operation Perform on the information store in one or more registers.

⇒ Register tranfer means the operation is perform in micro operation and the output off micro operation is tranfer into Same or another registor.

⇒ The danguge is a procesor for writing Symbols to verify specific given Computitional process.



(Fig!- Block Diagram of RTL System)

Microoperations

- Inter-register tranfer micro operations ($R \rightarrow R$) ($R \rightarrow R$)
- Arithmatic micro operation ($+, -, *, \div$)
- Logic micro-operation ($\wedge, \vee, \oplus, \sim$)
- Shift Micro-operation (shl, shr)

★ The operations performed on the data sub system are data tranfer & data move etc

★ A micro operation is a simple operation that can be performed during one clock period.

Eg:- Shift, move, count, add and load etc.

(1) Inter-Registor Tranfer micro-operations

⇒ It is wese to tranfer content of one Registor to another register.

⇒ The information tranfer for one register to another reg register is represented by Symbolic form by wing the operator

⇒ The Statement of Simple Register tranfer is given by $B \leftarrow A$

Content of A is transfer to B. $A \rightarrow B$

(2) Arithmatic Micro operation

⇒ The micro operations that perform arithmatic operation on Numeric Data. Store in the register.

⇒ addition (+), Subtraction (-), 1's Compliment, 2's Compliment.

A ⟷ B+C →

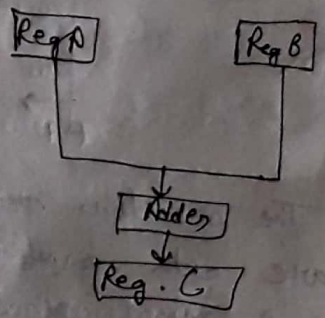
B ⟷ A-C

C ⟷ $\bar{A}$

C ⟷ $\bar{B}+1$

A ⟷ A+1

B ⟷ B-1



(3) Logic Micro operation

⇒ The micro operations that perform bit manipulation operation on the non numeric Data. in the register is called logic micro operation.

AND Function. $A \leftarrow B \wedge C$

OR Function. $A \leftarrow B \vee C$

NOT Function $A \leftarrow \bar{B}$

(4) Shift micro operation:

⇒ This is micro operation transfers binary bit b/w the registers serially.

→ Registers can be shift to the left or to the right.

⇒ The one bit shift is represented by Symbolic representation in the form of Statement

1-bit shift is represented by

A ← shl B → 1 bit shift to the left of register B

C ← shr A → 1-bit shift to the right of register A

Shif left → (A ← shl A)

shift right → (B ← shr B)

Circular shift left → (C ← cil C)

circular shift Right → (C ← cir C)

Arithmatic shift left → (B ← ashl B)

Arithmatic shift right → (C ← ashr C)

Ex:- B = 1 0 0 1 0 1 0

B ← shl B

⇒ 1 0 0 1 0 1 0

Ex 2 ⇒ shr B

⇒ B = 1 0 0 1 0 1 0 ← 0

⇒ 0 1 1 0 0 1 0 1 A

Circular shift

Ex: B = 1100 1010

B ← CirB

⇒ 01100 10 11

B ← CirB

B ⇒ 1100 1010

⇒ 100 1010 1

Ex 1 → Sh(2) → Circular.

Bus Transfer

⇒ A group of wires that connect several devices to convey data or information is called as a bus.

Bus transfer :- The Data transfer between various blocks connected to the common bus is called bus transfer.

⇒ A digital computer has many registers and it is necessary to provide data path between them to transfer information from one register to another. if separate lines are used to each register there will be accen no of wires and controlling of these wires makes circuit complex.

⇒ Therefore n multiple configuration a common bus system is used to transfer information b/w registers.

⇒ A common bus consist of a set of common lines, one for each bit of a register through which binary information transfer one at a time.

→ Control signal are use to determine which is the source register and which is the destination register.

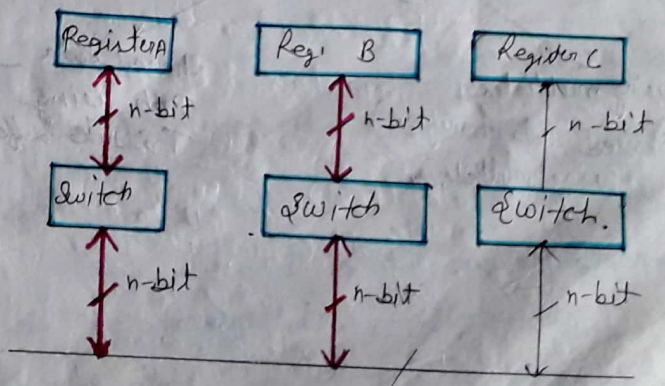
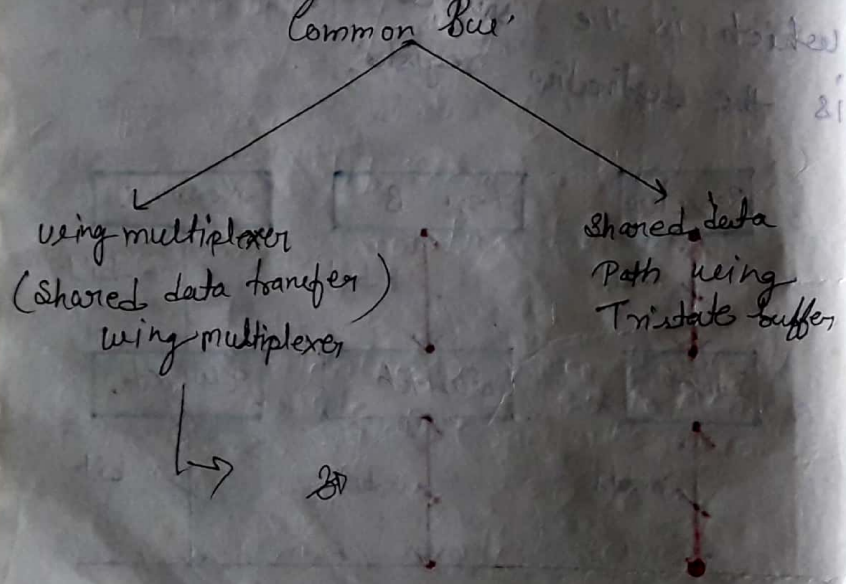


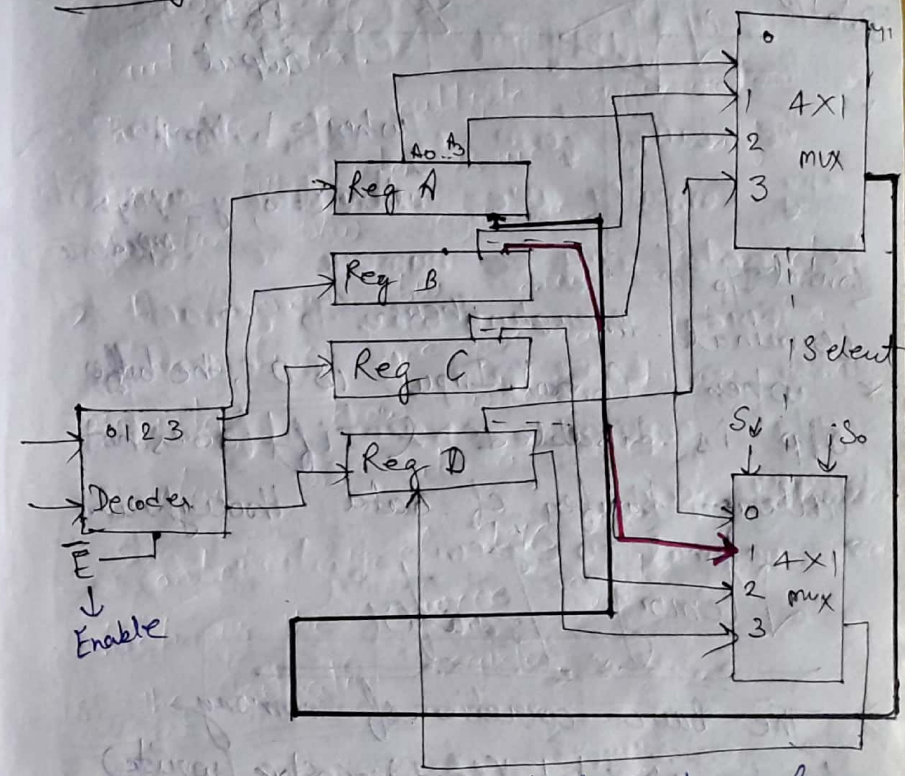
Fig:- Shared common bus system

- **Switches** ⇒ Switches are required path to be shared. these switches are implimented by multiplexer. or tristate non-inverting buffer.
- one Common control signal is used to control all the switch, the switches are close (Connection enabled) or open (Connection established)

Common bus ⇒ Common bus scheme can be implimented in two way



Using mux

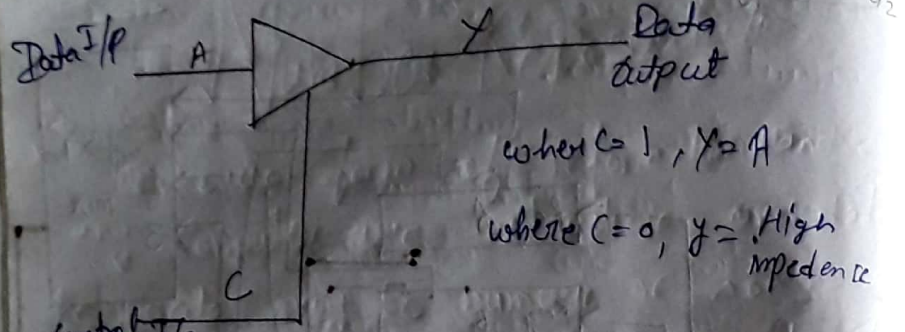


⇒ The transfer of information from the bus into the destination register is by activating the load control of the register.

⇒ if the decoder is not enable no information will be transfer b/w the register.

Shared data Path using Tristate buffer

⇒ when Controll Input is 1, the buffer output is enabled and buffer allow to transfer the data from Input to output



when C=1, Y=A

where C=0, y= High impedence

terminal through them.

* when Cordol Input is 0, the buffer o/p is disabled & ift does not allow tranefer of data throughout.

Memory Tranefer

⇒ The Basic operation of memory tranefer is fatch (Read) or store (write)

⇒ The read operation tranefer a Copy off Content from memory location to C.P.U.

fatch (Read) → memory to C.P.U

write → C.P.U to memory

⇒ The store operation tranefer the word information from the CPU To a Specific memory location, destroying Previous Content of their location.

⇒ The memory coord is represented by M

➡ The required information is selected from the memory location by address, It is stored in the address register. Symbolised by AR

⇒ The data tranefer to another register or data register Symbolized by DR

Read operation

Read : DR → M[AR]

- This tranefer information into data Register (DR) from memory word M selected by the address reg. (AR)

Write operation

- The write operation tranefer the content of data register to a memory word M selected by the address.

write : M[AR] ← B

- The Content of B is tranfered to selected memory location.

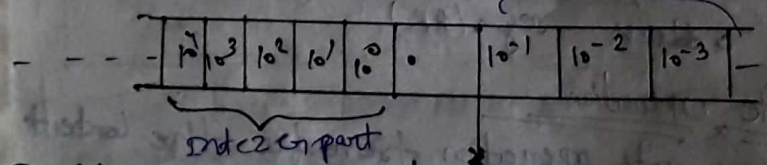
M[AR] → B

CHAPTER-2

Number Representation

* The Basic form of information Handle by any Computer are instructions and data the data may be numeric or non numeric
⇒ There are two way to represent information either in binary form or in decimal form the digital Computer represent information in binary form

➡ ① Decimal no system.



② Binary NO System

Ex:- Decimal NO System

467.34

⇒ $4\times10^2+6\times10^1+7\times10^0+3\times10^{-1}+4\times10^{-2}$

⇒ $(467.34)_{10}$

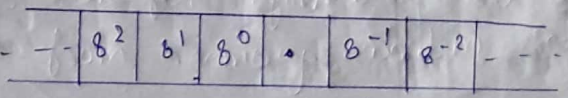
Binary :-

Name	Size (bits)
Bit	1
Nibble	4
Byte	8
word	16
Double word	32

Octal Number

Base → 8 ⇒ 2^3

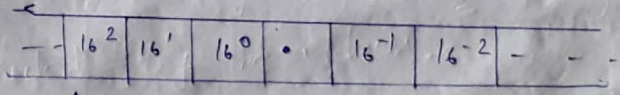
Represented by → 3bit (0 - - 7)



Hexadecimal :-

Base 16 → 2^4

represented by 4 bits (0 - - 15)



10 → A
B → 11
C → 12
D → 13
E → 14
F → 15

Number Representation

- Fixed point Representation
  - Signed magnitude representation
  - 1's Complement
  - 2's Complement
- Floating point representation → Ieee Standard for floting point representation (24.5)

(24)

⇒ Based on the no system two Basic Data types are implimented in the Computer System.

(1) Fixed Point number

(2) Floating Point number

⇒ In binary number system a number can be represented as an integer or a fraction

Fixed Point Number

⇒ A number having a fixed length, is called as fixed point number. these numbers are represented into two form that is signed integer & unsigned integer.

⇒ Unsigned integer number represent positive numbers.

⇒ To represent negative numbers various technics are used because a Computer don't have any provision to represent negative sign.

① Signed Binary Numbers

⇒ A Binary number can be have only two Symbols 0 & 1, these Symbols are used to indicate positive binary numbers. and negative binary numbers.

⇒ In the MSB bit of sign number one bit is reserve to indicate the sign of the remaining bits are magnitude. (n+1) → magnitude. n-bit → 1 sign

Sign magnitude Representation

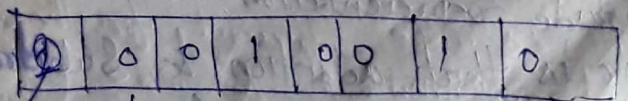
⇒
- 0 ⇒ The given no is +ve
- 1 ⇒ The given no is -ve

Eg:- +18 & -18 are represented at 8-bit

18 equivalent binary no — 10018 → 5it

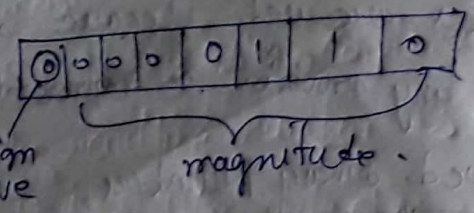
+18 =

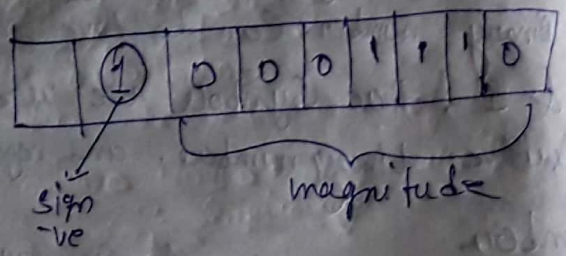
-18 =

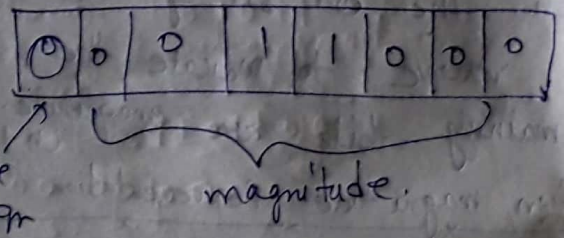


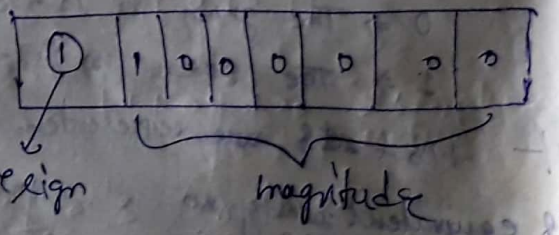
-ve

Eg:- 6 → 0110 → 8 bit

+6 = 

-14 ⇒ 

+24 = 

⊕-64 = 

Draw backs :—

⇒ There are several drawbacks of sign magnitude representation.

(i) One is that addition & Subtraction required Consideration of both the sign of the number and there relative magnitude in Order to [illegible]

the required operation.

(ii) There are two representation for 0

⇒ +0 = 00000000
⇒ -0 = 10000000 ⇐

⇒ This is inconvenient because it is difficult to test for zero. the sign magnitude representation is not normally used because this circuit implimentation is more complex then the other system.

(2) 1's Compliment Representation

1 OR 1 ↓ 0

⇒ In the 1's Compliment Representation negative no are obtain by Complimenting each bit of the Corresponding number (i,e) by changing 0 to 1 & 1 to 0

Eg:- 0100

1's Compliment → 1011

(2) 20 = 0 0 1 0 1 0 0 (+ve) ⇒ 00 010100

-20 = 1 0 1 0 1 1 (-ve) ⇒ 11101011

(3). 2's Compliment

⇒ If one is added to the 1's Compliment of a binary number the resulting number is known as 2's Compliment of the binary number.

Ex:- 20 → 0 0 0 1 0 1 0 0

+1

0 0 0 1 0 1 0 1

2's Compliment

1's Compliment = 1 1 0 1 0 1 1

+1

1 1 0 1 1 0 0

20 → 0 0 0 1 0 1 0 0

1's Compliment → 1 1 1 0 1 0 1 1

+1

1 1 1 0 1 1 0 0 2's Compliment.

Advantages of 2's Compliment

① For 4-bit numbers, the value -8 is represented only in the 2's Compliment and not in other representation.

② It is more effecient for the logic circuit implimentation. for more often used in Computer for addition & Subtraction operation.

Ex:- -8 = 1 0 0 0

1's = 0 1 1 1

+1

2's = 1 0 0 0

Floating Point numbers

- These Containe the binary point variable in its position, and hence called floating point numbers.
- These no's has 3-fiedds.
  - sign
  - mantisa
  - Exponent
- General structure of floating point numbers.

$$\pm M \times B^{\pm E}$$

E → Exponent

M → Significant digit (mantisa)

B → Scaling factor (Base)

Eg:-

⇒ Let us Conider a number 1 1 1 0 1 1 1. 1 0 0 1 1 0 to be represented in the floating point number. ~~to repres~~ ~~torepraunt~~ in floating point formate, the first binary point is shifted to left to the sign bit and the number is multipled by correct scaling factor to get the same value. After shifting.

⇒ 1.1 1 0 1 1 1 1 0 0 1 1 0 × 2 ↳ Exponent

Significant or ↓ Base (Scaling factor)

Here sign = 0 (+ve no. with 0)

Mantissa = 11011100110

Exponent = 6

⇒ A floating point no bias value is added to the two Exponent.

⇒ Due to this the magnitude of two number can be Compare by doing arithmatic on the Exponent Part.

⇒ The range of floating point-

$$-1 \le f \le 1-r^{(n-1)}$$

if n = 32 bit

$$-1 \le f \le 0.999$$

IMP

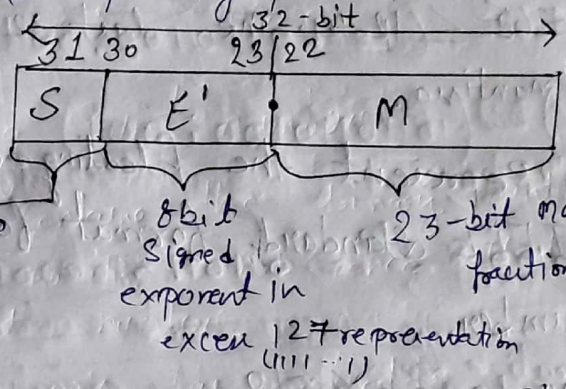
IEEE Standard for Representation floating point no. :-

[IEEE → Institute of Electrical & electronics Engg]

IEEE 754

- 32-bit (single precision)
- 64-bit (Double precision)

Single precision ⇒ The 32-bit IEEE Standard floating point representation is –



- value Represented = $\pm M \times 2^{E'-127}$

① The sign of a number is given in the first bit followed by the representation for the exponent of the scaling factor.

② Instead of the sign exponent E the value actually stored in the exponent field is $E' = E + \text{bias}$

③ In the 32-bit floating point system the bias is 127, Hence $E' = E + 127$

④ This representation of Exponent is called the Excess 127 formate.

⑤ The end value of E' (0 to 255) are used to indicate the floating point value of exact (0) or (∞) respectively in single pricision.

⑥ Thus E' is in range $0 \le E' \le 255$.

⑦ The actual exponent E is in range $-126 \le E \le 127$. The last 32 bits represent mantissa.

Example: Represent the floating point in the IEEE standard format for the given number.

$1.000\,10100 \times 2^{-10}$

⇒ Sign = 0

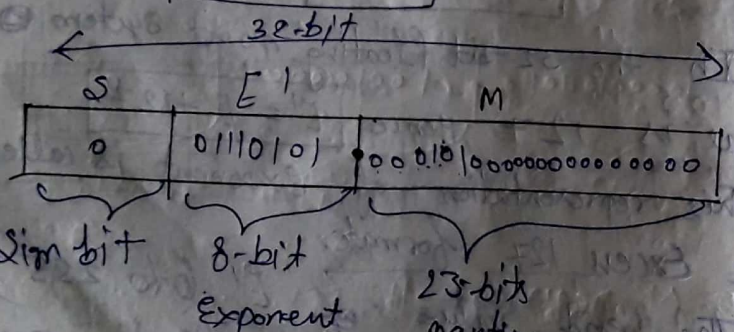
Mantissa = 00010100 — 0 (23 bits)

Actual Exponent $E = -10$

$E' = E + 127$

$= -10 + 127 = 117$

$E' = 1110101$



Q⇒ 1.00101×2^{-87} represent this no, in IEEE single precision.

Sign = 0

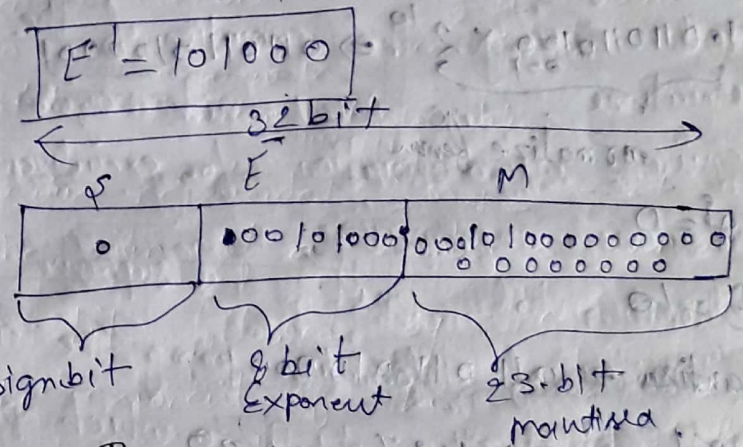
mantissa = 00101 — 0

Actual Exponent $E = -87$

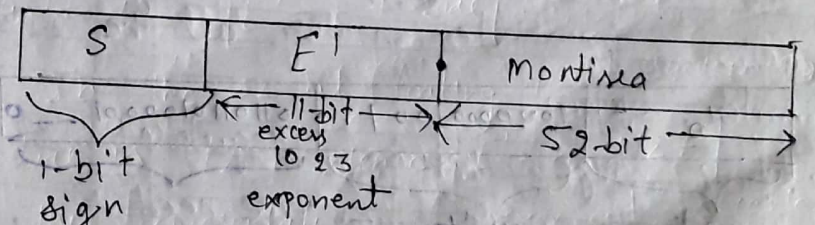
$E' = -87 + 127$

$= 40$

$E' = 101000$



Double Precision ⇒



$E' = E + 1023 \Rightarrow -1022 \le E \le 1023$

Represent

Q. $(1466 - 125)_{10}$ to Double precision

Ans $1460 = (10110110100.001)$

$(1460.125)_{10} = (10110110100.001)$

$1.0110110100001 \times 2^{10} \Rightarrow 01101101000001$

normalize form

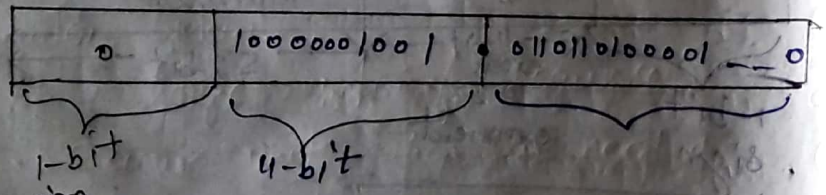
$qs = 0$

$E = 10$

mantisa $= 01101101000001$

$E' = 10 + 1023 \Rightarrow 1033$

$= 10000001001$



Processor Organization ⇒ A processor is a main part of Computer, CPU is same times Heffers to as Data Sub system.

⇒ The Execution unit Containe a set of Registor for storing Data and

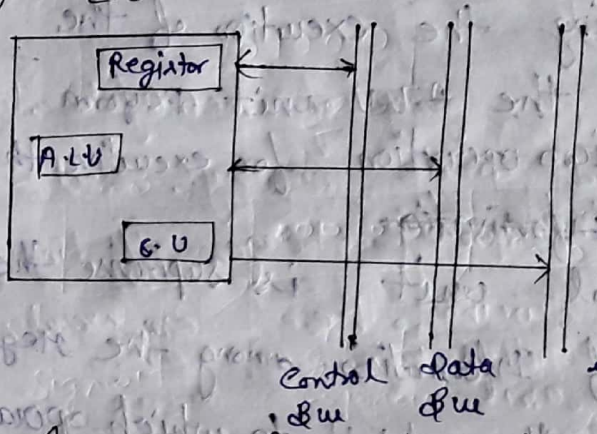
A.L.U unit for execution of Arithmatic & logic operation.

⇒ The processor has 3-main part

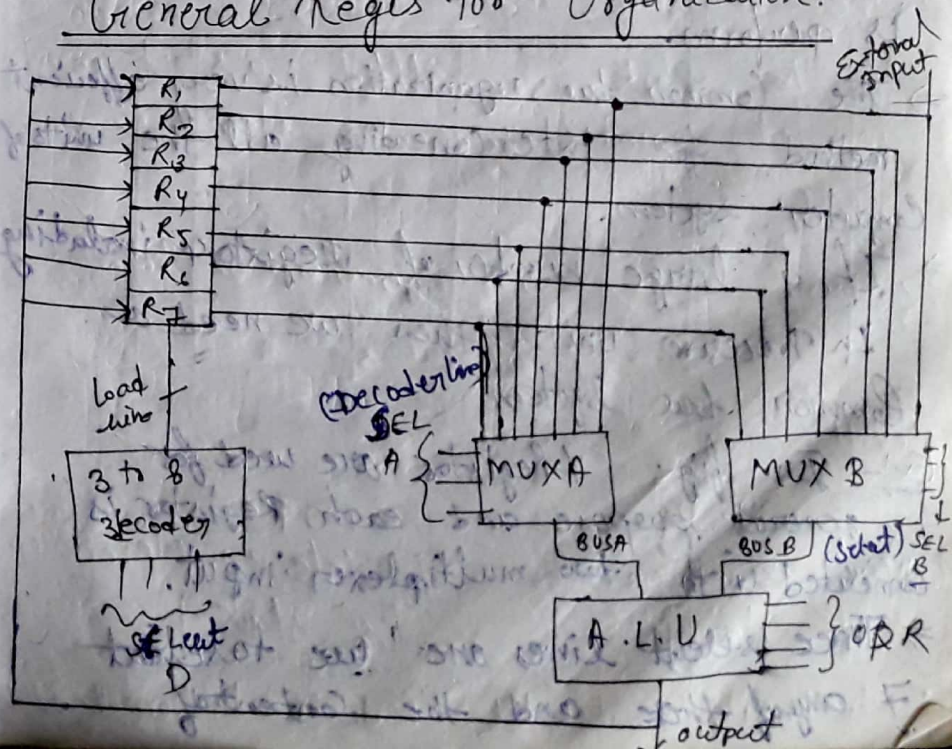
(1) Registors

(2) A.L.U

(3) C.U



General Registor Organization.



⇒ C.P.U
The part of Computer which perform the Data procesing operation is called C.P.U
⇒ The C.P.U is made of 3 Parts (i,e) Registor, C.U, A.L.U unit.
⇒ The Registor set stores intermediate data used during the execution of the instructions. the A.L.U unit perform required micro operation for executive the instructions. information
⇒ The Control unit is supervise the transfer of Information among the registor and instructs the A.L.U to which oporation is perform.
⇒ The Common bus organization is very effecient method for interconecting all the units of Computer System.
⇒ If a large number of registors including In Processor unit then we need a Common bus System

Ex: In fig. 7 Registor are used for general purpose and each Registor is connected with two multiplexer input.
⇒ Three select lines are use to select 7 any of three and the Content of selected Registor are supplied to the A.L.U unit.
⇒ The Bus A & B are use to form the I/P to the Common A.L.U. the operation to be perform is selected in the A.L.U is determine arithmatic & logic micro operation why using function Select line (OPR)
Ex. The result of the micropperation is available for all the registors as a I/P

Eg ⇒ To perform the following addition.

$R_3 \leftarrow R_1 + R_2$

- MUX A select (SEL A): to place the Content of R_1 in to Bus A
- MUX B select (SEL B): to place the Content of R_2 in to Bus B
- A.L.U operation (OPR): to provi A+B
- Decoder destination selector (SELD)
  1. To Transfer the Content of the output Bus in the R_3

Stack Organization

⇒ Stack is a storage structure that stores information in such the way that the last item stored is the first item retrieve.

⇒ It is the LIFO structure (Last in first out)

⇒ It is a group of memory locations in the read write memory. that is used for temporary storage of binary information during the ten execution of a Program.

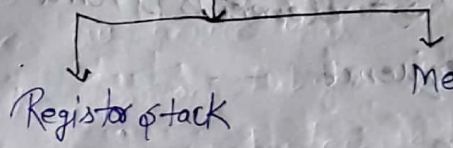
⇒ The Registor that hold the address of the stack is called a stack pointer and it point the top item in the stack.

⇒ There are two Operations on the stack

PUSH
[insert an item on top of stack]
• store/write the data in stack

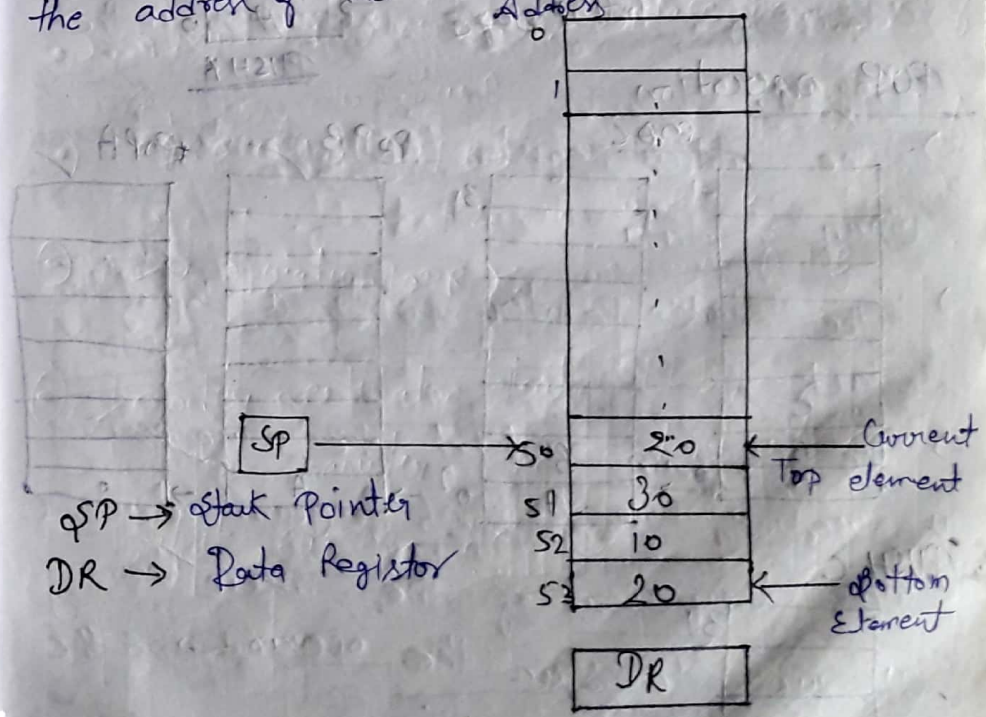
POP
[Delete an item on top of stack]
• Read the data from the stack

Implimentation of stack



(i) Registor stack ⇒ A stack can be organized as a collection of registors that are used to store temporary Information during the execution of a Program.

⇒ The stack pointer is a registor that hold the address of the top element of the stack



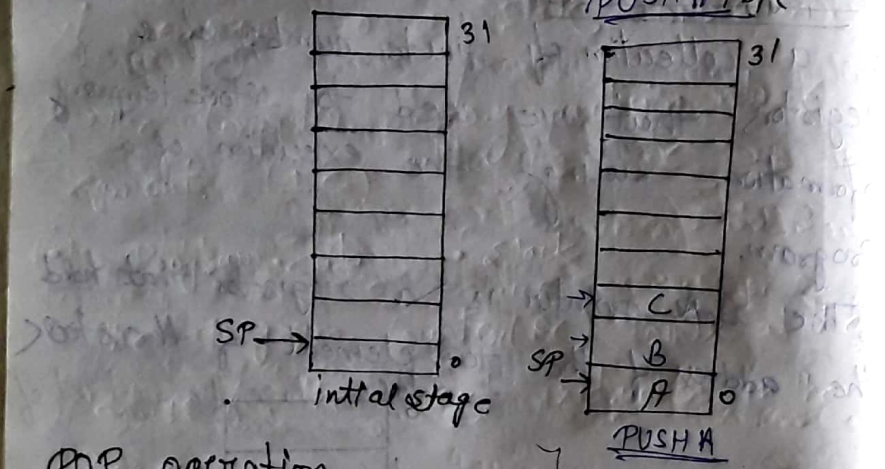
SP → Stack Pointer
DR → Data Registor

[Fig:1 - Organization of 64 @ 53 word stack]

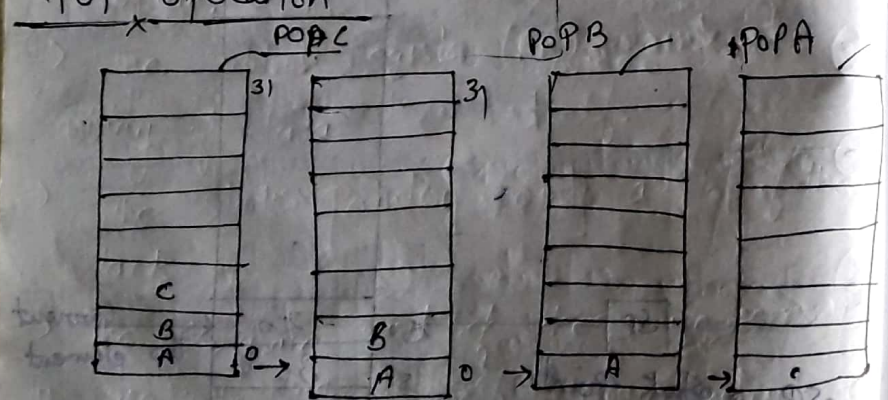
PUSH & POP operation

A, B, C are 3- elements / items

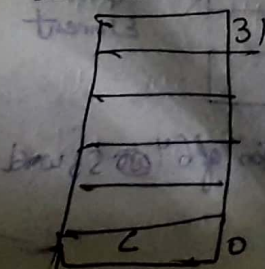
⇒ instruction sequence - -



POP operation



PUSH C



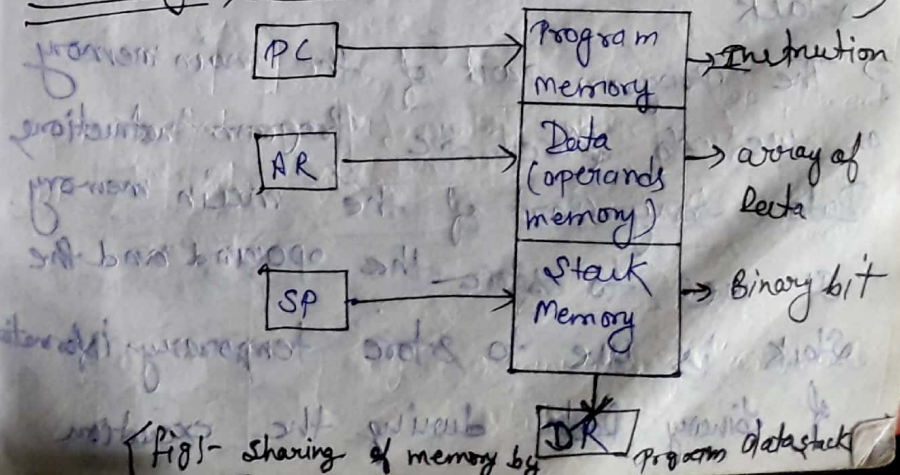
⇒ n-32 word stak the stack pointer is a 5-bit registor because $32 = 2^5$ where 5 represents the bits.

⇒ Initially the stak is empty, when the Data element pushed on the stack the stack pointer increament so that it points to the next higher level registor address.

⇒ And The Data element is written in the registor pointed by the stack pointer

⇒ if stack pointer reach at 31 & one more data element insert then stack Pointer is increamented to 0 now there is no more empty registor in the stack and the stack is ~~send~~ set to full.

Memory Registor (Memory Organization)



Figs- sharing of memory by

⇒ The stack can be implimented a Ram attached to a C.P.U.

⇒ The implimentation of a stack in the C.P.U is done by asigning a porson of memory to a stak operation.

⇒ The Computer memory partision into 3-segment. (i.e)

(i) Program (ii) Data (iii) stack

⇒ The program Counter (PC) Points at the address of the next instruction in the Program,

⇒ The addres Registor (AR) points at array of data.

⇒ The stack Pointer points a top of the stack

⇒ The Program part of the main memory as used to stored the Program instructions Data Array Part of the main memory as we to store the operand and the stack is we to store temporary information of binary Data during the execution of a program.

The Data is inserted by PUSH operation as follow →

$SP \leftarrow SP-1$

$M[SP] \leftarrow DR$

POP opern –

$DR \leftarrow M[SP]$

$SP \leftarrow SP+1$

NOTE (Blank Note Book kealyfai)

Addresing Mode

1. Immediate Addresing mode.
2. Direct Addresing mode
3. In Direct Addresing mode
4. Registor Direct Addres mode
5. Registor In direct Mode
6. Index Addresing mode
7. Relative Addresing mode

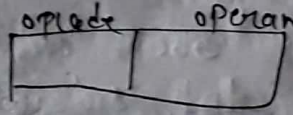
(i) ~~Immediate~~ Addresing mode

⇒ The various formats of specified the operands are called addressing mode.

⇒ The operation field of an instruction specify the operation to be perform.

⇒ This operation must be executed on Some data stored in Computer Register or Memory word.

⇒ The various method of accessing data is called as the addressing mode.

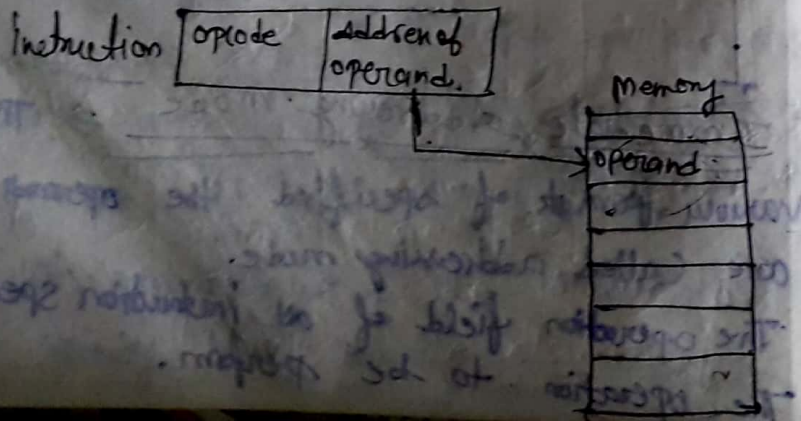


(1.) Immediate addressing mode ⇒

In this method the require Data is directly loaded to require register or memory location.

⇒ The data to be loaded or move to a register or a memory location in this mode is provided as a ⇒ part of instruction itself.

(2) Direct Addressing mode ⇒

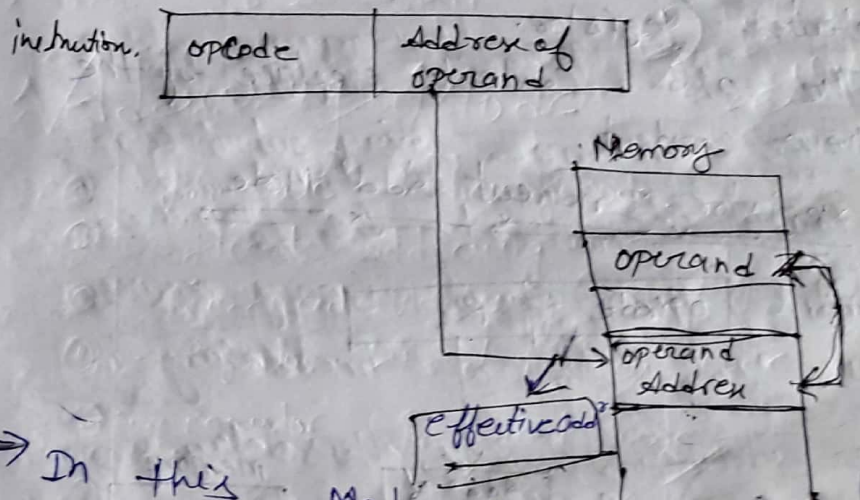


⇒ In this mode the operand is in memory location.

⇒ The address of this location is given in the Instruction.

⇒ The address is given directly by address field of instructions.

(3) Indirect Addressing mode ⇒



⇒ In this Mode the address field of the instruction gives the address where the effective address is stored in memory.

⇒ Control unit fetch the instruction from memory and uses its address path to access memory again to read the effective address.

Effective add^{r} = address part of instruction + Content of C.P.U register

⇒(4) Registor Direct addr mode ⇒

(4) instruction | opcode | R_1 → Registor

R_1 → operand

⇒ In this mode the operand are kept a registors that reciev data in the C.P.U : if a content is loaded to selected Registor.

⇒ The old content replace by the new content.

(5) Registor Indirect Addr Mode

instruction | opcode | R_1

R_1 → Addr of operand → operand (memory)

⇒ when the addr of the operand is in a registor the mode is refer to as registor Indirect mode.

⇒ This addressing mode is use to sequencially access the element of an array stored in memory.

⇒ The starting address of the array is stored in a Registor, an access mode can the registor is increamented to point to the next element.

(6) Indexed or Displacement Addressing mode

⇒ instruction | opcode | R_1 | 4

operand Adder → memory → operand

R_2

⇒ In this mode the effective addr of the operand is generated by adding a Constant value to the content of a Registor.

⇒ The effective address of the operand is given by

$$A_{eff} = X + [R]$$

X → Constant

R → Registor.

(7) Relative Addressing Mode ⇒

instruction | opcode | 4 → Constant

operand Adder → memory → operand

PC

⇒ It is similar to Indexed mode, But the

(b) Base Add, is held in program Counter (PC) rather than in another register.

⇒ This allow the storage of memory operand at a fixed offset from the Current Instruction.

Address of operand = PC + Constant

Look ahed Carry adder

⇒ The sum and Carry output of any Stage Can not be produced until the input Carry occurs, this leads to a time delay in the addition Process, this delay Knowns Carry propagation delay.

Ex:-

```
  1 1 0
  0 1 0 1
+ 0 0 1 1   ⟶ LSB
-----------
  1 0 0 0
```

- In this example the addition of LSB [least significant bit] position produced the Carry bit to the second position bit.

⇒ This Carry when added to bit of second position produced a Carry into 3rd position. the latter Carry when added to the bits of the 3rd position produced a Carry in the last position.

- The Key thing notice in this exm. is that the ~~sum bits generate the~~ last position of a bit is depend on the Carry that was generated by the addision in the previous position.

⇒ This meane that Adder will not produced Coreut result until LSB Carry has Propagated thorow intermediate for adder.

⇒ This represent a time delay that depends on the propagation delay produce is in full Adder.

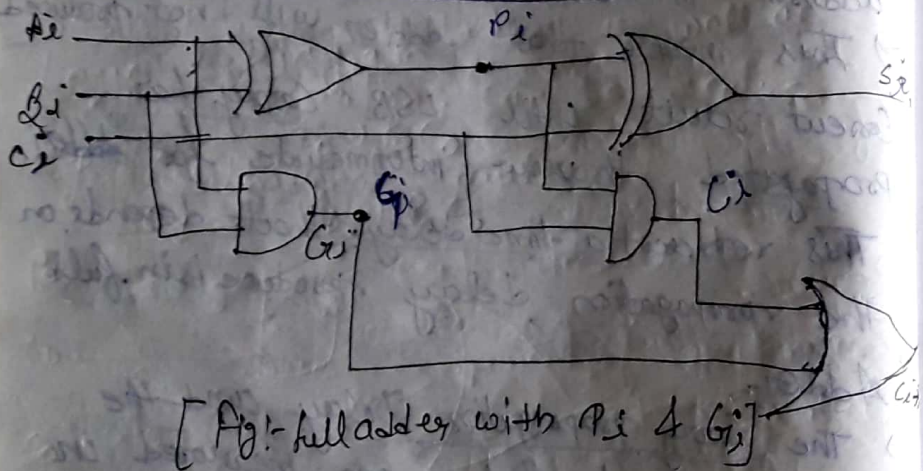
⇒ The Carry propogation Delay and the Sum propogation Delay are measured in terms of gate delay.

⇒ one method of speeding up this Process why eliminating outer stage Carry delay is called look ahed Carry addition.

⇒ This method utilize logic gates to

look at the lower order bits of augend & addend to see if a higher order carry is to be generated. In wes function Carry generated and Carry propogated.

⇒ Carry Generate → G_i
Carry propogate → P_i



[Fig:- full adder with P_i & G_i]

$P_i = A_i \oplus B_i$

$G_i = A_i \cdot B_i$

for 4-bit $i = 0, 1, 2, 3$

$P_i = A_0 \oplus B_0$

$G_i = A_0 \cdot B_0$

How to calculate 4 bit

$i = 0, 1, 2, 3$

$S_i = P_i \oplus C_i$

$C_{i+1} = G_i + P_i C_i$

for $S_i = P_i \oplus C_i$

$S_0 = P_0 \oplus C_0$

$C_{0+1} = G_0 + P_0 C_0$

$C_1 = G_0 + P_0 C_0$

$i = 1$

$S_1 = P_1 \oplus C_1$

$C_2 = G_1 + P_1 C_1$

$C_2 = G_1 + P_1 G_0 + P_1 P_0 C_0$

$i = 2$

$S_2 = P_2 \oplus C_2$

$C_3 = G_2 + P_2 C_2$

$i = 3$

$S_3 = P_3 \oplus C_3$

$C_4 = G_3 + P_3 C_2$

for $i = 2$

$S_2 = P_2 \oplus C_2$

$C_3 = G_2 + P_2 (G_1 + P_1 G_0 + P_0 P_1 C_0)$

$C_3 = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_0 P_1 C_0$

$C_2 = G_1 + P_1 (G_0 + P_0 C_0)$

$C_2 = G_1 + P_1 G_0 + P_0 P_1 C_0$

$i = 3$

$S_3 = P_3 \oplus C_3$

$C_{i+1} = G_3 + P_3 C_3$

$= G_3 + P_3 (G_2 + P_2 G_1 + P_1 P_2 G_0 + P_0 P_1 P_2$

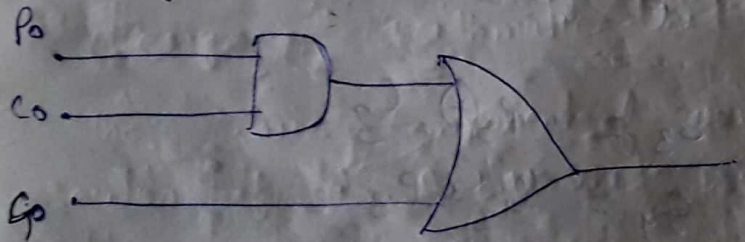
$C_4 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_1 P_3 P_2 G_0 + P_3 P_0 P_1 P_2$
G_0

C_2, C_3, C_4 execut on one time.

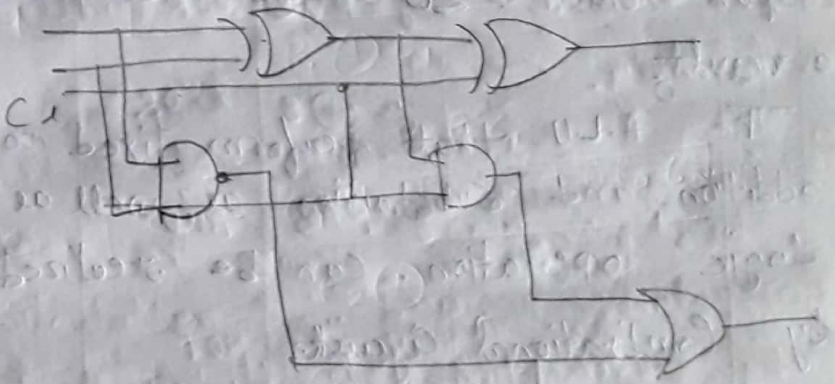
Logic Diagram

For $C_0 = G_0 + P_0 C_0$

$C_1 = G_0 + P_0 C_0$



for C_1, C_2, C_3, C_4



Arithmatic & logic unit Design

75

⇒ The A.L.U unit perform all the necessary addition, substraction, shifting & logical operation

② It require one or two operands upon which to operate and produce a result.

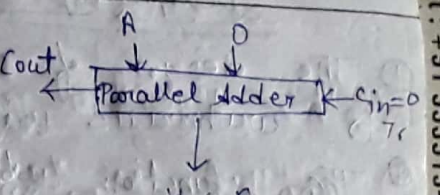
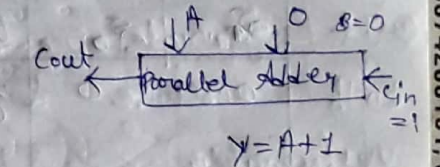
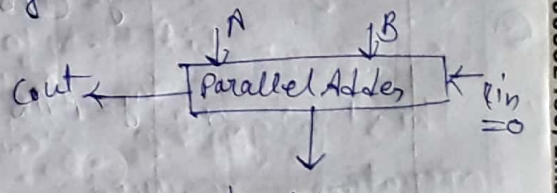
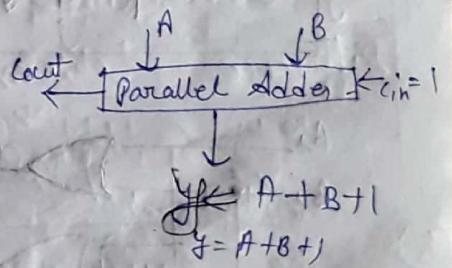
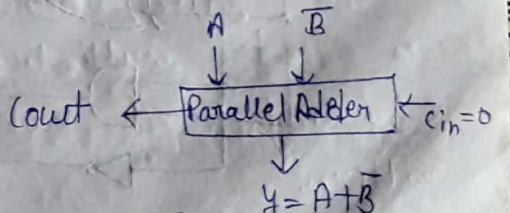
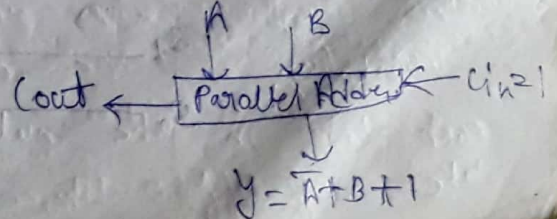
③ The A.L.U that perform fixed point addition and substraction as well as logic operation. Can be realized by Combinational Circuit.

Design of arithmatic Unit

① The Basic Component of arithmatic section of a A.L.U unit is parallel adder.

② Parallel adder is use to perform Arithmatic unit operations.

③

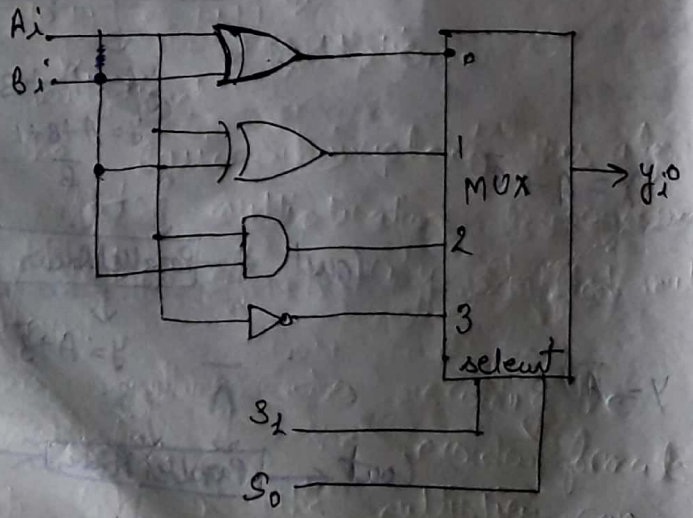
S.No	function	Implimentation
1.	$y \leftarrow A$ (Data transfer)	
2.	Implement $y=A+1$	
3.	Addition without carry $y=A+B$	
4.	Addition with Carry $y=A+B+1$	
5.	$y=A+\overline{B}$	
6.	$y=\overline{A}+B+1$	

Design of logic Unit

① The design of logic unit is simple compare to arithmatic unit.

② The function table & logic diagram are as follow -

S_1	S_0	Output	function
0	0	$y = A \vee B$	OR
0	1	$y = A \oplus B$	X-OR
1	0	$y = A \wedge B$	AND
1	1	$y = \bar{A}$	NOT

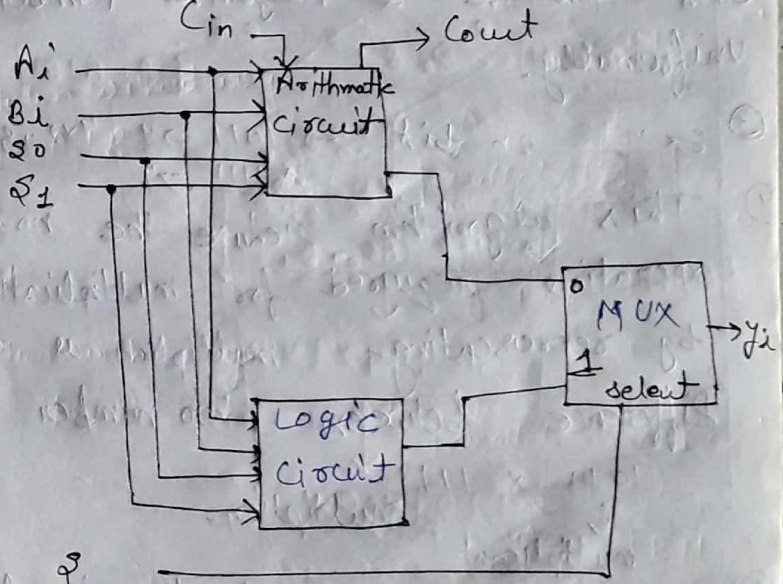


③ The logic Diagram is repeated n times for n-bit logic Circuit.

④ Therefore input A, B & Output y are shown with subscript i

where $i = 1, 2, 3 \ldots n$

Combining Arithmatic & Logic unit



S → S is Control Signal

when [S = 0, [Arithmatic Circuit is activated]
S = 1, [Logic Circuit is activated]

Multiplication

Booth's Algorithm for signed numbers

① Booth's algorithm is a powerful Algorithm for sign number multiplication.

② It generate 2n-bit product and treat both positive & Negative Number. Uniformally.

③ Ex:- 2n bit ⇒ 1011 → 4 4×2=8 bit
1111 → 4

④ This algorithm reduce the no of operation required for multiplication. by representing multiplier as a Difference between two number

1011 × 1111
↓ multiplicant — multiplier

⑤ The Basic principle of this algorithm is that multiplication is performed by addition of properly shifted multiplicant pattern.

⑥ This algorithm involves recording the multiplier. In the recorder format. each bit in the multiplier can take any of three values (i.e 0, +1, & -1

⑦ In general form the recording can be as follows

Scanned bit of multiplier		Recoding Bit
0	0	0
0	1	+1
1	0	-1
1(Q)	1(Q-1)	0

Ex:- Record the multiplier 0, 1, 1 0 1 0 1 1 0 0 1 for Booth algorithm

Record ⇒ according to the table & value of Q, (Q-1)

(Q) (Q-1)
0 1 1 0 0 1 0

+1 0 -1 0 +1 -1

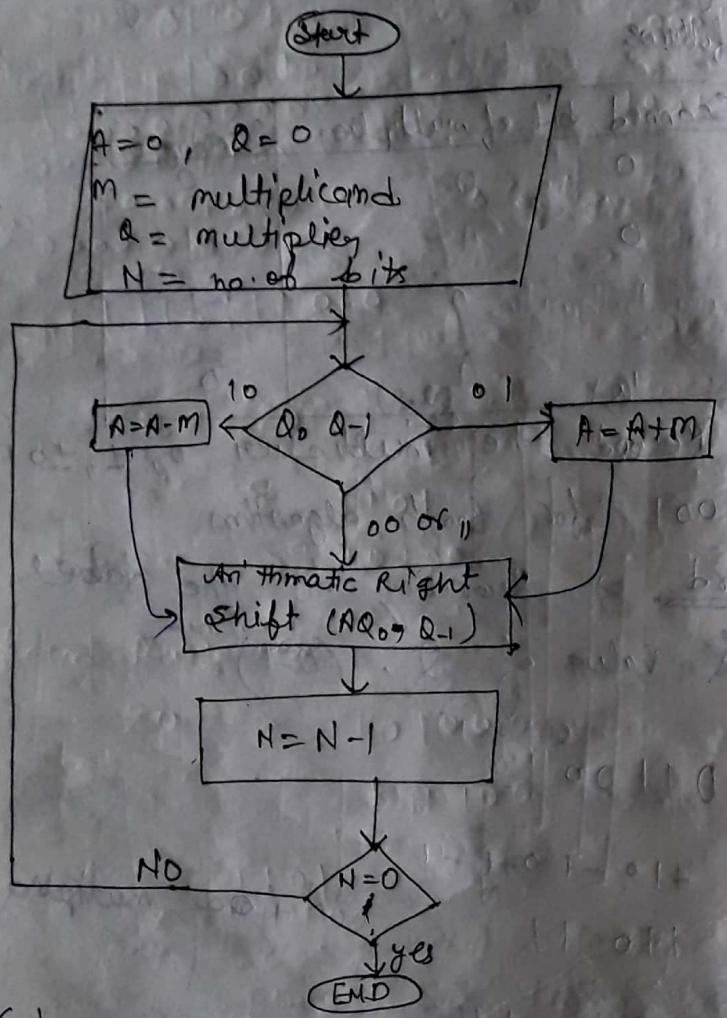
Record multiplier

1 1 0 1 1

1 0 1 1 0

0 -1 +1 0 -1

Flow chart



Ex:- multiply +14 × -5 by Booth algo.

+14 → 0110 (multiplicand)

-5 → 11011 (multiplier)

Recod multiplier:-

11011[0)

⇒ 0 -1 +1 0 -1

0 1 1 1 0 (multiplicand) ⇒ 2 n bit = 2×5 = 10 bit o/p

× 0 -1 1 0 1 (recoded) — no 5

NOTE

⇒ firstly multicand write 2's compliment

⇒ when multiplier (-1) write 2's Compliment of multiplicand

0 1 1 1 0 → 1's Compliment → 1 0 0 0 1

2's Compliment +1 → 1 0 0 1 0

because output coc count 10 bit

this.

again.

0 0 0 0 0 1 0 0 1 0

1 1 1 1 1 1 1 0

0 1 1 1 0 → 0 0 0 0 0 1 0 0 0 1

+1

0 1 1 1 0 → 0 0 0 0 0 0 1 1 1 0

1's Compliment → 1 1 1 1 1 1 0 0 0 1

2's Compliment +1

1 1 1 1 1 1 0 0 1 0

0 1110
× 0 0101

→ 01110
10001
+1
10010

11111 00010
00000 0000x
00001 110xx
11100 10xxx —2's Complement
0000 00xxxx

1 1110111010

Q -13 × -20 in 6-bit

-13 → 1101
-20 → 10100

Recod Mutiplier

-20 → 10100 0
-1+1-100

⇒ -1101
× -1+1-100

Q 15 × -13

15 → 01111
-13 → 1011 → 2's Complement
⇒ 1011 → -1010-1

01111
× +1010-1

11111 0001
0 0000000x
1 11111 1xx
0 000 000 xxx
0 1000 xxxx

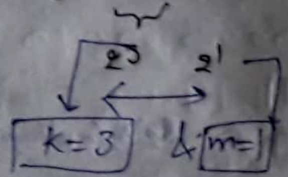
1 1 0011110 1 — Ans

Booth's Algorithm

① Booth's Algorithm gives a procedure for multiply binary integer in sign to 2's compliment representation.

② It operate on the fact that string of zero's (0's) require no addition (in multiplier) but just shifting and string of (1's) from bit 2^k to weight 2^n can be treated as $2^{k+1} - 2^m$

Example → 001110 (+14)



This no. can be represented as $2^{k+1} - 2^n$

i.e. $2^{3+1} - 2^1 = 16 - 2 \Rightarrow 14$

∴ M×14 done by $M \times 2^4 - M \times 2^1$ product can be obtained by shifting the M 4-times to left & subtracting M sifted M left once

③ where M = multiplicand

- Before shifting subtract multiplicand M may be added or substract from partial order or unchange. according to following

Rules. if

Q_0	Q_{-1}	
0	0	(Ashr) (Arithmatic shift right)
1	1	(Ashr) (Arithmatic shift right)
0	1	→ Add M
1	0	→ Subtract M

Ashr

1 0 0 0 1 1 0 0 1 1 0 → Shr (equme zero hai) ⓪1 0 0 0 1 1 0 0 1 1

same → 1 1 0 0 0 1 1 0 0 1 1

Find Ashr.

Simple right shift → 0 1 0 0 0 1 1 0 0 1 1

Ashr → ①1 0 0 0 1 1 0 0 1 1

क्योंकि Ashr Que me 1 है।

Ex:- 5 × −4 (in 4-bit)

5 → 0101

4 → 0100 1's compliment → 1011
+1
1100 2's Comp

0101
1010
+1
1011

operation	A	Q_0	Q_{-1}	N (No. of bit)
Initial	0000	1100	0	4
Ashr	0000	0110	0	3
Ashr	0000	0011	0	2
A+2'(m) m+multiplicand(s)	+1011	[A-M]		(n-1) (1-1)
	1011	0011	0	
Ashr	1011	0011	0	
Ashr	1101	1001	1	1
Ashr	1110	1100	1	0

Answer.

⇒ if 1 to than A - M y A+2'(m)

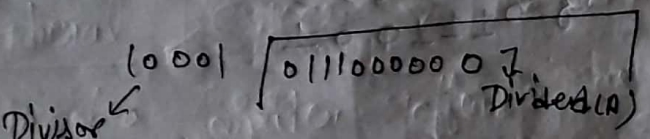
⇒ again Ashr process

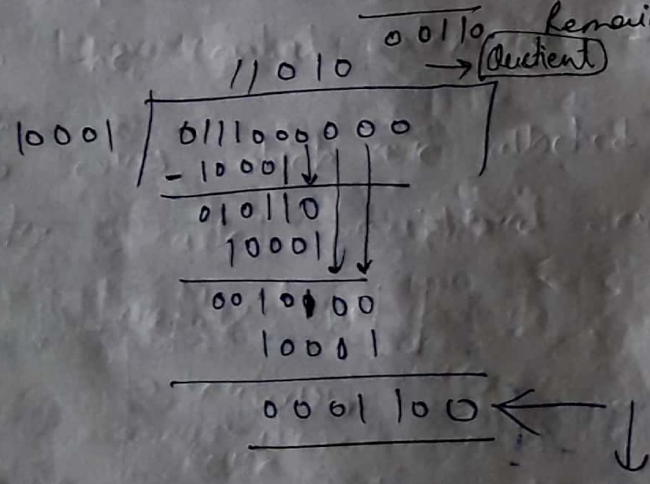
⇒ and n bit every process on Ashr (n-1) than (0').

Binary Division Algorithm

⇒ Binary Division is simpler then Decimal because quotient digits are either 0 or 1. and there is no need to estimate how many times the dividend/partial remainder fits into the divisor.

Ex:-





⇒ In the Division process first the bits of the divident are examine from left to right, until the set of bits examine represent a number greater than or equal to Divider.

⇒ Until this condition occure zero's are placesed in qotient from left to right.

⇒ when the condition is satisfied (1) is placed in quotient and Divisor is substract from Dividend.

⇒ from this point onward the division process is repeated until the last right digit occurs.

⇒

Array Multiplication

⇒ The multiplication procest for binary number is similar to decimal number.

⇒ Binary multiplication is simple then Decimal multiplication. because it involved only two bit 0 & 1.

Rule for multiplication

0 * 0 = 0
0 * 1 = 0
1 * 0 = 0
1 * 1 = 0

⇒ it uses n shift & add to multiplyn bits binary no.

⇒ The combinational logic circuit implemt to perform this multiplication and called Combinational multiplier or /Array multiplier

UNIT - II

I/O Organization

Peripheral Devices → The Input output subSystem of a Computer reffered to as Input/output, Provides a Communication between the central system and I/O devices.

⇒ I/O Devices which are attached to Computer is called peripheral devices.

Ex:- Phreripheral unit
- → keyboard
- → Mouse
- → Joystick
- → Printer
- → Display unit
- → Magnetic tape
- → Magnetic disk

★ I/O Interface

★ The C.P.U, memory and Input/output Devices are the main part of a Computer System.

★ The C.P.U fetch instructions from the Memory, Processed and store the result in the memory.

★ The main function of I/O interfaces is to tranfer information b/w C.P.U & memory.

★ ~~Dma~~

I/O Module or Port

★ There are two major function of I/O unit.

(1.) Interface to C.P.U & Memory via System Bus.

(2.) Interface to one to more I/O device tailored data links.

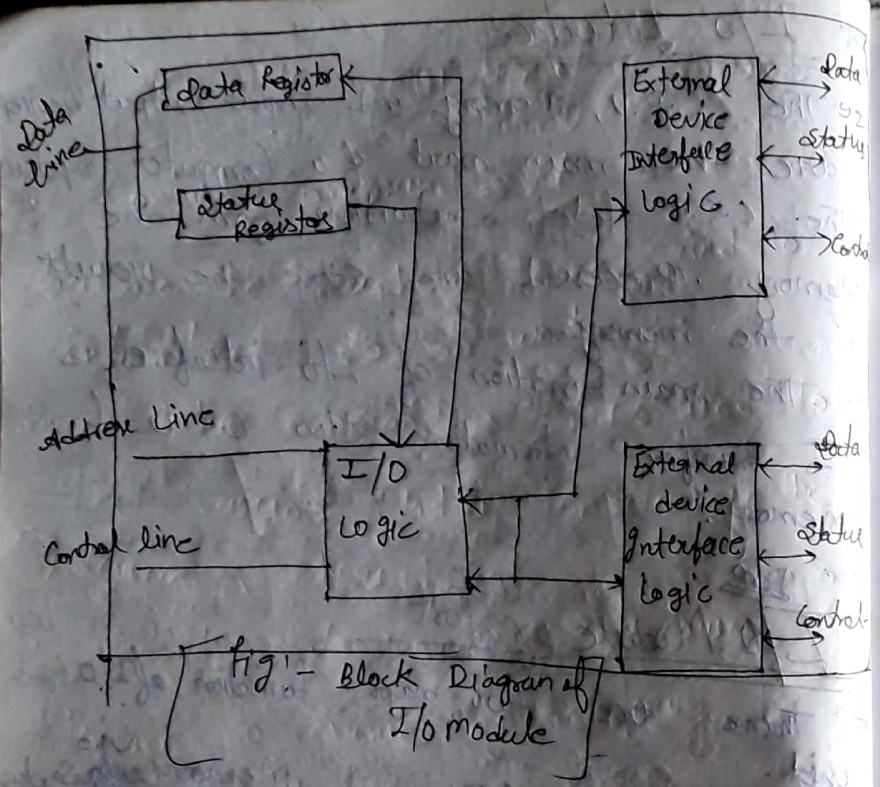


Fig:- Block Diagram of I/O Module

Major Components of I/O Module

1. Control & Timing :- To coordinate the flow of traffic b/w internal resources & External resources
2. C.P.U Communication :- It involves diferent
3. Device signals such as Control
4. Data Buffering signal, exchang of data. between C.P.U & I/O module over data bus. And Check whether the weather Pheripheral device is ready or not for data tranefer.

C.P.U ⇒ Device ⇒ It involves Commands, status information & Data.

Data Buffering ⇒ Data tranefer rate of Pheripheral devices are quite high than C.P.U & memory. Data is buffered in I/O module.

Error detection ⇒ To check different types of error such as mechanical, electrical malfunction, bit pattern, transmission error etc

Input/output Bus

★ The Input/output Bus Consist of Datalines, Address lines and Control lines. it act as an Communication link b/w Processor and Several Peripheral Devices.

★ It Comprises of magnetic tape, magnetic disk, printer, and terminals of each interface decodes.

★ The address & Control recieves from Input Bus.

★ The Input/output Bus Processor Provide a function code to Control line which is reffer to as Input/output Command.

There are four type of Command

1. Contol Command ⇒ It is used to activate the pheripheral Device. And to inform it to do work.

2. Status Command ⇒ It is used to test various status Condition in the Interface & pheripheral.

3. Output Data Command ⇒ It causes to Interface to response by transfering the Data from bus into one of it. Registors

4. Input Data Command ⇒ It is the apposite of output Data Command, the interface recieve from pheripheral and placed is Into buffer Registor.

Interrupt

→ An Interrupt is a signal to the proceror immeadiate by hardware & software. indicating an event that need immediate attention.

(2) An interrupt alert the Proceror requiring the interruption of Correct Code that the proceror is exegcted.

• Baically stoped something for few moment

Types of Interrupts

(1) External interrupt
(2) Internal Interrupt
(3) software interrupt

(1) Caused by I/o devices, from timing devices are from other external fource, it is also caused by Power failure.

Internal Interrupt ⇒ There are caused from illegal or error ness use of instruction, Data. they are also know as Traps Ex:- overflow Condition in Registors due to some Arithmatic micro operation.

Software interrupt ⇒ The interrupt that occurs due to the software is called software interrupt

Priority interrupt

⇒ When interrupt request arrived from two or more devices Simetaneously, the Proceror has to

decide which request should be execute first. and which one should be ~~delete~~ delayd.

⇒ The processor takes the decision with the help of interrupt priority. It accept the request which having the highest priority.

⇒ A Priority interupt is a system that established a priority over the various sources to determine which condition is to be service first / Execute first when two or more request arrive simtaneously.

DMA (Direct Memory Access)

★ Generally the Data tranfer from Memory → I/p Device I/o → memory can be achieved by only processor.

⇒ when the Data tranfer from memory to I/p Device, from I/o Device to memory

⇒ first the Processor send address & Control Signal to memory.. to read the data from memory. or to I/p / o/p device to read data from Input/output Device respectively ;-

⇒ Then the processor send address and Control Signal to Ip/op device. Or send the address & Control signal to memory to write data in to I/o device or memory respectively.

Data tranfer

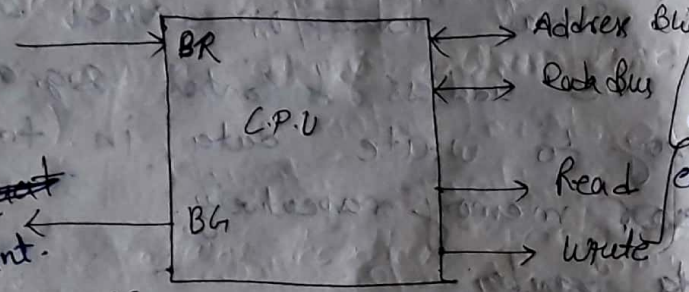
① I/o device → Memory → Processor send Address & Control signal to I/o device to write Data into memory.

② Memory → I/o device → Proces send address & Controll signal to memory read data from memory & write to I/o device.

This process is inconvenient because processor Control signal to request simultenously Hence → A Scheme called Direct memory Access (DMA) has been devoloped in which I/o, o/p device can access memory for Data tranfer.

○ DMA is an input/output technic used for high speed data tranfer.

② For Direct transfer a dedicated Hardware device is used called DMA Controller.



[Fig:- DMA Controller with signals]

⇒ During DMA Transfer the CPU is idle and has no Control of the memory buses.

⇒ The DMA Controller takes over the buses to manage the transfered directly between I/p & o/p Devices and memory.

⇒ The C.P.U may be placed in idel state in a varity of ways.

⇒ One Common method used in micro processor is to disable the buses through special Control signal.

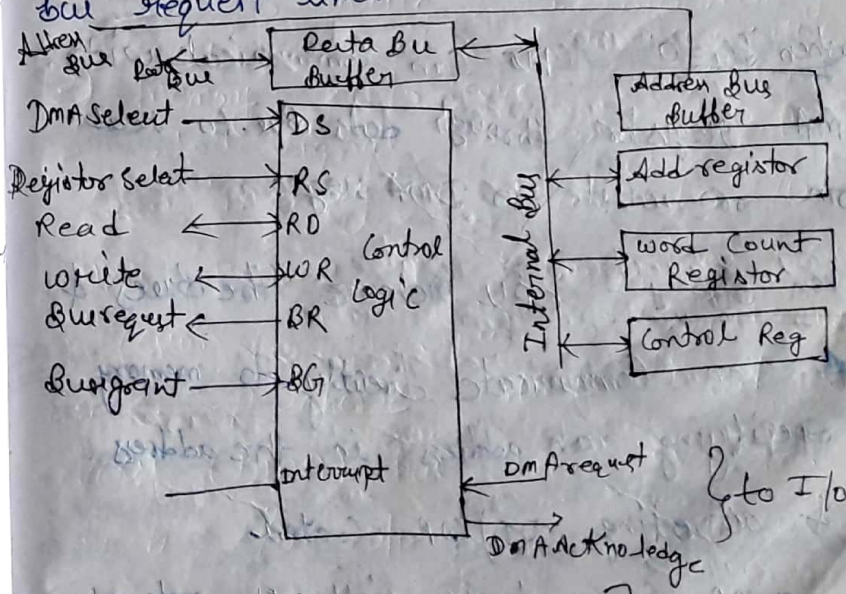
* Bus request :- ⇒ BR input is used by DMA Controller to request the C.P.U to release Control of the buses.

⇒ when this input is active the C.P.U Terminates the execution of current instructions & places address bus, Data Bus, Read & write lines into a high impedence state.

BG (Bus Grant) ⇒ The C.P.U activate BG to inform external DMA that buses are in high impedence state.

⇒ The DMA that intiate request now can take Control of the buses to conduct memory transfer. (without Processor intervention)

⇒ when DMA terminates the transfer, it disable bus request line.



[fig:- DMA Controller]

⇒

DMA Controller Needs the usual circuit of an interface to communicate with the C.P.U & Input/output devices. It need three registers, Address Register a word count register and a set of address line (control Registor).

⇒

① Add. Reg. — Contains the address specify memory location

② Contol Reg

③ Word Count Reg → Specify mode of transfer. No. of words to be transfered

✱ when $BG=0$ - CPU Can Communicate with DMA registor through data Bus. to read from or write to DMA reg.

• when $BG=1$ - CPU release the Buses & DMA Can Communicate directly to memory by specifying an address in the address Bus & activating RD/WR Control

NOTE - DMA Communicates with the external peripheral through the request & Acknowledge lines by handshaking procedure.

Modes of Tranfer

- Programmed I/O
- Interrupt initiated I/O
- DMA (Direct memory Accer)

⇒ Data transfer between the central computer and Input/output devices may be Handled in a variety of modes.

② Some modes use the C.P.U AS an intermediate path, others transfer the Data directly to the and from the memory unit.

③ There are three possible modes to transfer Data

(i) Programmed I/O → (C.P.U) —

(ii) Interrupt intiated I/O —

(iii) DMA (Direct Memory Access)

Programmed I/O ⇒ Programed I/O operations are the result of input/output Instruction written in the Computer Program.

⇒ Each Data item transfer each initiated by an instruction in the programe.

② I/O operation will mean a data transfer between an Input/output Device

and memory or between the input/output device and processor

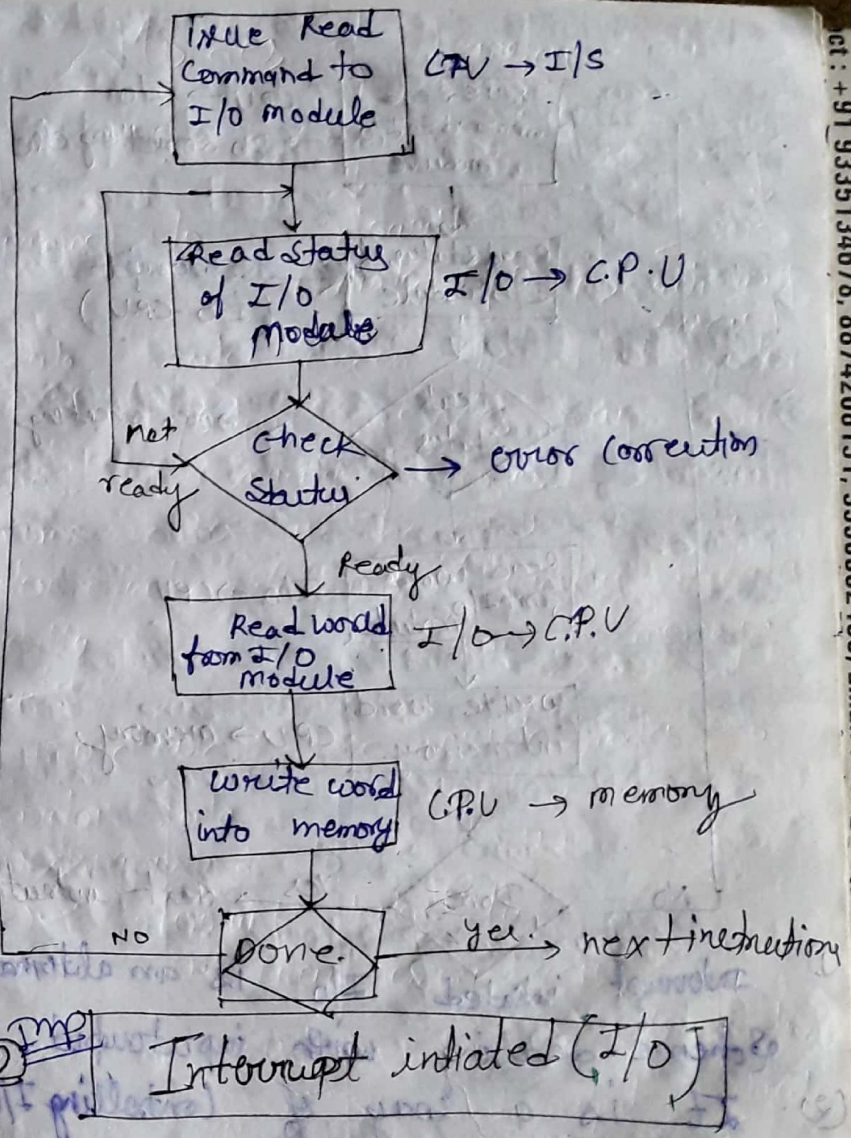
③ If on any computer system input/output operations are completly controlled by the processor, then the system is said to be using programe I/O.

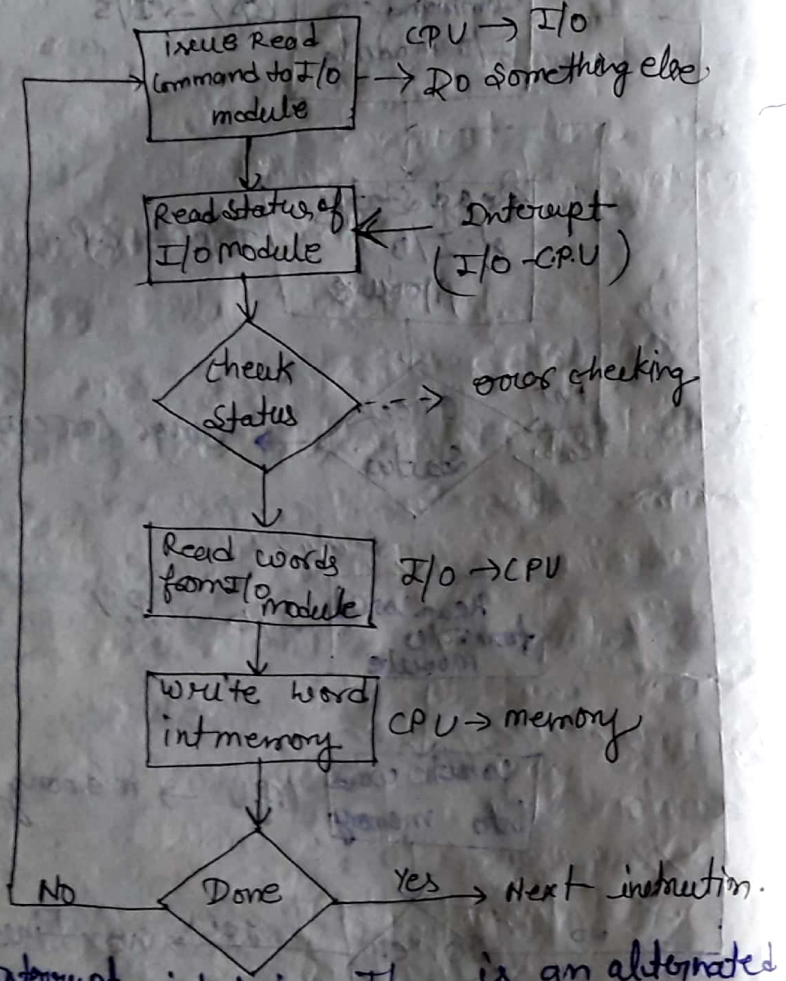
④ When such A teotniques is wed, Processor executes programme that intiate, Direct and terminate the input/output operation, including sensing device status, sending the read or write command And toanefering the Data.

Steps of Progammed I/O

[Basic steps for program type]

1. CPU request I/O operation
2. I/O module perform operation
3. I/O modul perform set status Bits.
4. CPU check Status bit periodically.
5. I/O module doesnot inform CPU directly
6. I/O modul doesnot interrupt C.P.U
7. CPU may wait or Comeback later.





① Interrupt initiated I/O is an alternated scheme dealing with input output.

② It is a way of controlling I/O activities where by a peripheral or terminal that need to make or recieve a data tranfer sends the signal.

③ This will causes a program interrupt to be set at a time appropriate to the priority level of the I/O interrupt, relative to the total interrupt system. the processor enter an interrupt service routine.

④ The function of the routine will depend upon the system or interrupt levels and priorities that is implimented in the processor.

② [Basic operations of Interrupt initiated I/O]

⇒ C.P.U ~~waiting~~ change read Command

→ I/O module gets Data from peripheral while C.P.U does other works.

⇒ I/O module interrupt C.P.U.

⇒ C.P.U request Data.

⇒ Input/output module tranfer Data.

③ Interrupt Processing

⇒ A device driver initiate an Input/output request on behalf of a processor.

⇒ The device driver signals Input/output controller for the proper device, which initiates the requested Input/output (hensrate)

⇒ The device signals the I/O Controller that is ready to retreve input

, the output is complete or that an error has been generated

⇒ The C.P.U receives the interrupt signal on the interrupt request line and tranfer controll over the interrupt handler routine.

⇒ The interrupt handler determines the cause of the interrupt, perform the neccesary procesing and execute a return from interrupt instruction.

⇒ The C.P.U returns to the execution state prior to the interrupt being signal

⇒ The C.P.U continuess processing until the cycle begains again

Advantage of Interupt initiated I/O

① Fast & effecient.

Disadvantage

① Can be tough interrupt to get various Peices of one well to gether.

I/O Channel

➔ A channel is an Independent hardware Component that cordinate input output device to a set of controller.

➔ Computer syetem that we I/O channel have special hardware Component that handle all input output operations.

➔ channels are seprate, Independent and low cost processor for its functioning which are called channel processor.

➔ channel processor are simple but Contains suffecient memory to Handle all input output Task.

➔ when input output transfer Complete or an error detective the channel controller Communicate with the C.P.U using an interrupt and inform to C.P.U about the error or the Task Completion.

➔ Each channel support work or mode controllers & devices.

Types of I/O channel

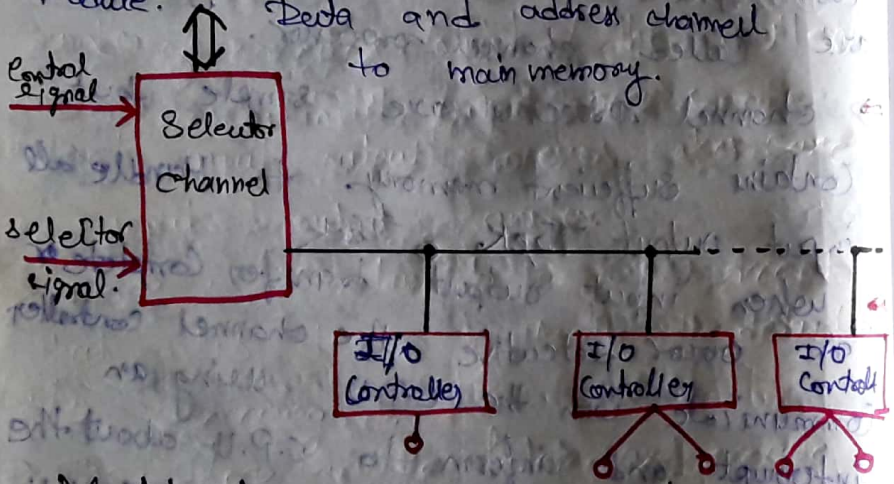
→ selector I/O channels

→ multiplexer. I/O channel

Selector I/O channel → This channel can handle only one Input/output operation at a time. and use to Control one high speed device at a time.

⇒ It select one device at a time and does the Data transfer.

⇒ Each device or a small set of devices is Handle by a Controller or Input/output Module. Data and address channel to main memory.



Multiplexer → (multiplexer I/O channel)

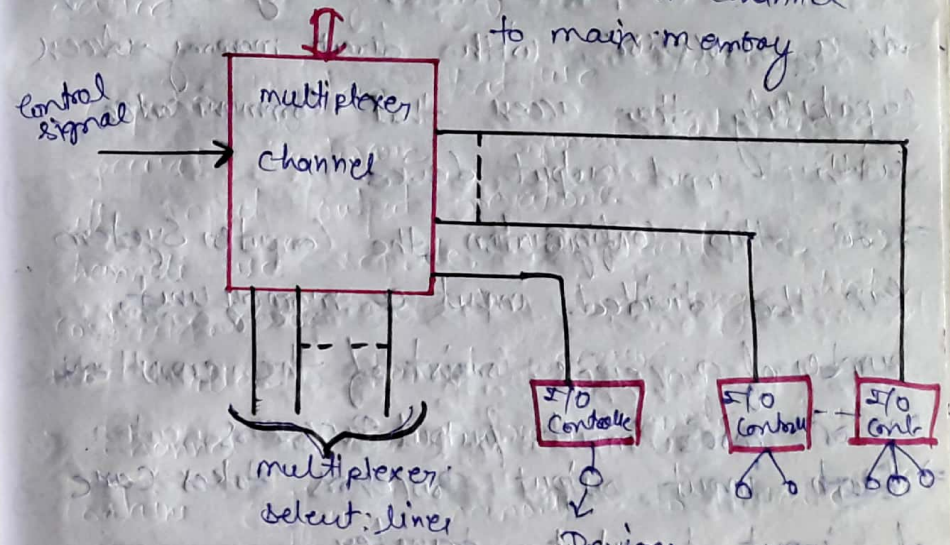
⇒ This can be Connected to a number of slow and medium speed devices.

⇒ It is capable of operating number of Input/output devices simultaneously.

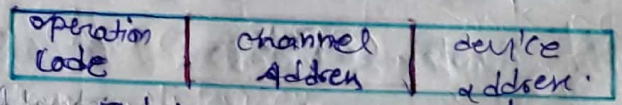
⇒ An multiplexed channel different devices can have different speed.

⇒ However it is best suited for low speed devices.

Data and Address channel to main memory



NOTE → The Computer system may have no of channels and each is assign one address each channel may be connected to several devices then each device is assign an address.



[Fig: I/O instruction format]

Imp

[I/O Processors]

→ Instead of having each interface Communicate with the C.P.U a Computer may have

one and more external Processor and assign them the Task of Communicating directly with all input/output Devices.

⇒ Input output Processors may be classified As a processor with direct memory Access Capability that access that Communicates with input output Device.

⇒ In this Configuration the Computer System Can be divided input & memory unit, number of Processors. Consist of the C.P.U and one or more input/output Processors.

⇒ Each input output Processors takes care of input & output transfer.

⇒ I/OP is similar to a C.P.U Accept that it is design to Handle the detail of I/O Processing.

⇒ The I/OP Can be fetch and execute its on instructions.

⇒

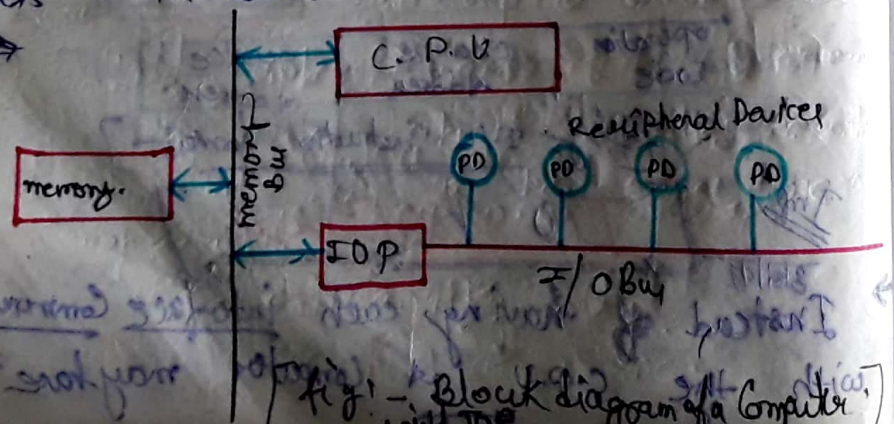


fig:- Block diagram of a Computer

[Features of I/O Processors]

1.) Input/output Processors can fetch and execute its on instructions.

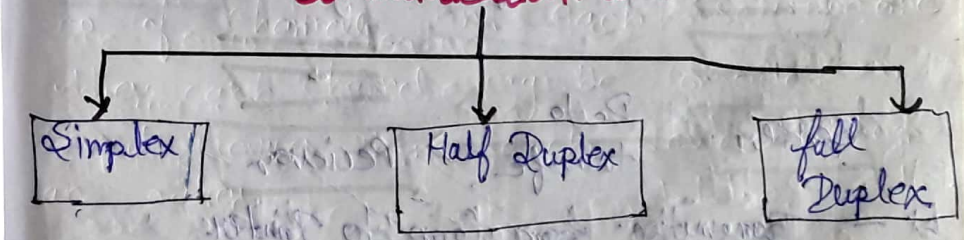
(2.) instructions are specially design for I/o processing.

(3.) I/o Processor also can perform other task such as arithmatic, logic, Branching and code translation.

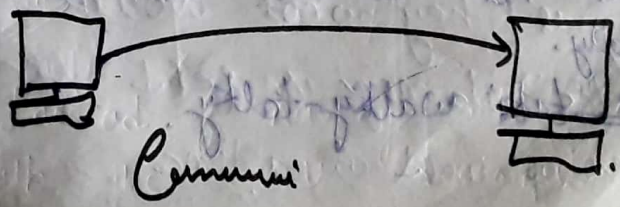
4.) It can transfer Data from 8-bit source to 16-bit destination and vice-versa.

(5.) The I/O Processors support multiprocessing environment.

Communication Mode



Communication ⇒ Transfer of information from one place to another place.

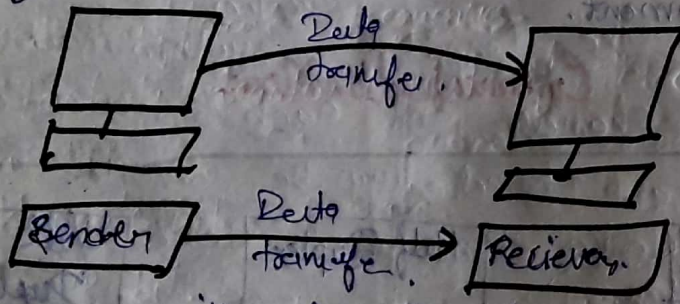


→ The communication is defind as unidirectional or Bidirectional transfer of Data from one port to another point. through a medium.

⇒ There are three types of Communication Mode.

1. Simplex mode
2. H. Duplex mode
3. F. Duplex mode.

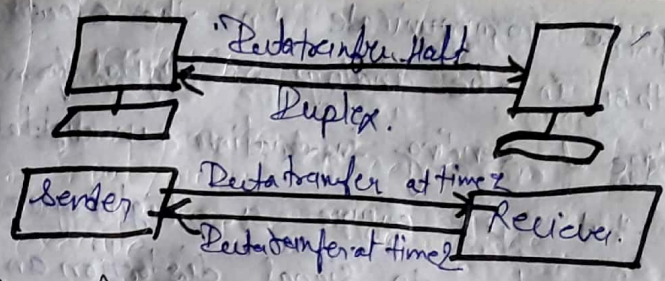
(1) Simplex Mode ⇒ It is simple Communication mode in which Data is transmitted over a singal channel in one direction only.



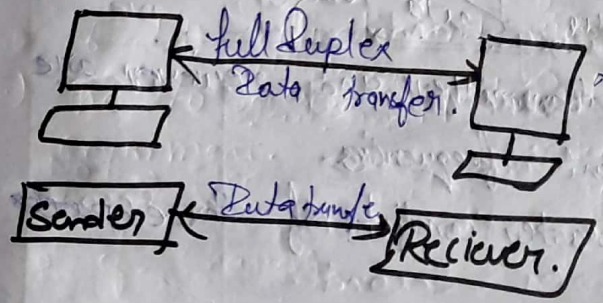
Ex:- Transmition from Comp to Printer.

(2) Half Duplex Mode ⇒ It is bidirectional in this mode Data is transmitted in both direction but not at the same time or simantaneously.

Ex:- ~~Walki-talki~~ walky-talky.



(3) Full Duplex Mode ⇒ In this mode Data is transmitted in both direction at the same time.



Methods of Communication (Transmitter format)

Synchronous

Asynchronous

• Data may be communicate in the electrical Domain either in the analog or digital form the advantage of sending information in digital form is that it ensure a high degree of ⓔ immunity to signal distortion and noise which made drastically alter the information being transmitted.

⇒ The information is tanfer through a channel

Asynchronous Data Transfer

⇒ In this transmitter and reciever are used different frequencies.

⇒ Each character consist of three parts

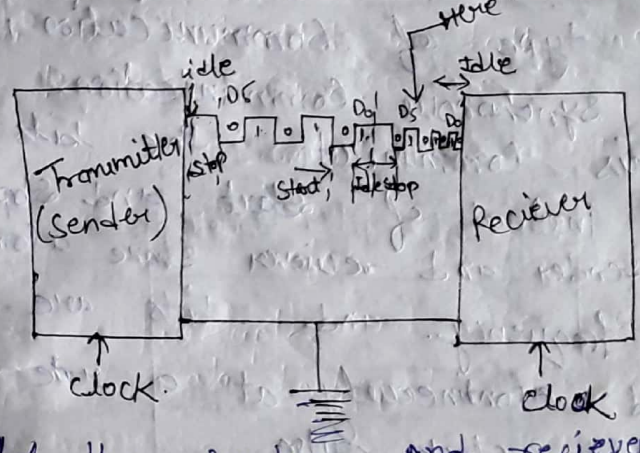
(1) Start bit
(2) character bit
(3) Stop bit.

⇒ In this format the bits of a character ⇒ are set at a Constant rate, but characters can come at any ways as long as they do not overlap.

The Data neg is send one character or words at a time.

when no data is send over the line it is maintain at and ideal value (i,e) logical (1)

Start bit ⇒ The Start bit is 0 each character is preceeded by Start bit (पीछे वाला)



Both the sender and reciever used a Separate clocking signal.

⇒ It is not essential that they both be off exactly the same frequency

⇒ The Start bit is indicate the biegining of the character

Stop bit ⇒ Stop bit indicate end of character. the combination of Start bit, character bit and Stop bit is knowas a frame.

Synchronous Transfer

⇒ In Synchronous transfer the system is used a common clock and the transfer of Data from sender to reciever with in a same frequency.

⇒ This type of Communication is called synchronous Communication.

⇒ In this Communication the bits are inserted instead of start and stop bit

⇒ The sender and reciever share a common clock frequency. and bits are transmitted continuously at a rate dictated by clock pulse

NOTE: Data Rate ⇒ No of bits per second or character per second

bits/sec → broad rate.

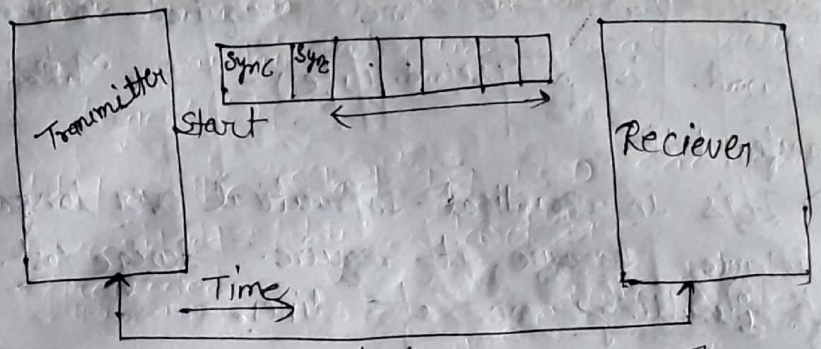
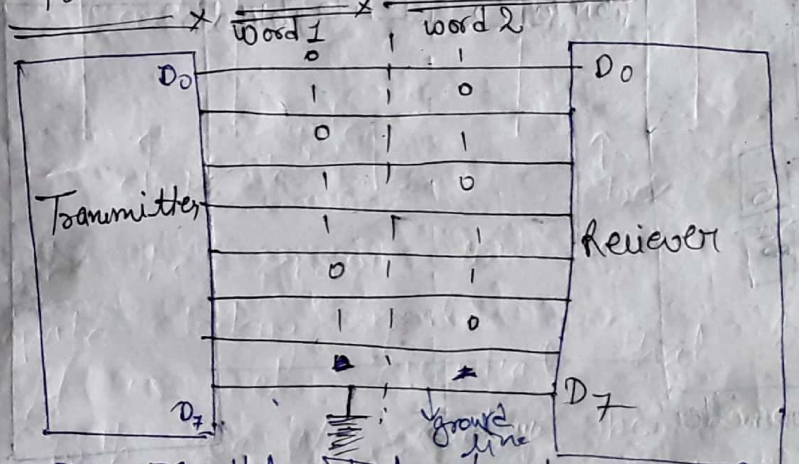


fig:- Synchronous Transfer

Parallel Data transfer



⇒ In Parallel Data transfer each word are send at same time.

⇒ when a word of n bits is to be transmitted in parallel each bit is transmitted on a separate line along with a common ground line.

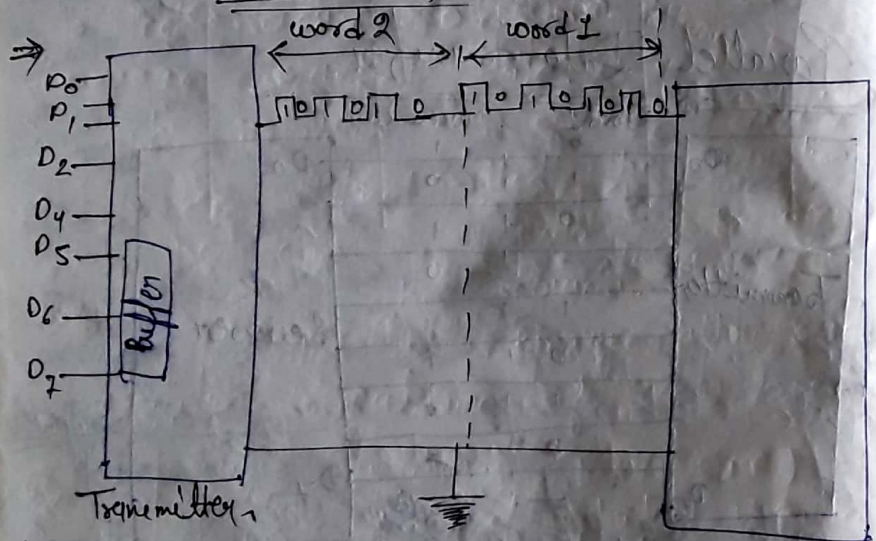
⇒ Thus a channel required n+1 lines.

⇒ If the time between succesive samples is 5 second, n = 8 bit and 2 succesive word word1 (01011101) & word2 (10100110) it can be

Seen that the time required to tranfer 1 word is 8 second (1 e) = time taken to tranfer 1 bit.

⇒ It is in ~~practical~~ impractical over long distance because it require large cost for installing no of lines.

Serial Data Tranfer



⇒ In serial Data tranemition each bit of the word is sent in serially, 1 at a time over single pair of wires.

⇒ A parallel to serial Convertor is used to convert the incoming parallel Data to a serial form.

⇒ If the bit rate is retained after the parallel to serial Conversion, the time

taken to tranemit a word in serial Data tranemition will be n times more than the time taken in parallel Data tranmition

UNIT - III

Date 04/09/2018

Instruction Set →

Instruction ⇒ Instruction is a command to the processor to perform a given task from on specified Data.

⇒ each instruction has two part 1 is task & to be perform called the operation and the second is the Data to be operated upon is called the operands.

⇒ And instruction is a Binary pattern Based on the architechure of C.P.U to perform A specific function.

⇒ The entire group of instruction is called the instruction set.

define performoperation.

Instruction | opcode | operands

Elements of an Instructions

- opcode
- Source operand reference (other)
- Destination operand reference
- Next instruction address

opcode	Source operand ref	Destination operand ref

[Fig:- A simple instruction format]

opcode ⇒ Specifies the particular operation that is to be perform

Operand field ⇒ Specifies where to get the source & destination operand

Types of perform operation.

⇒ types of operation performed by an instruction

1. → Data tranefer b/w memory & C.P.U register
2. → Arithmatic and logic operation on data
3. → perform sequencing and control
4. → I/O module

Instruction Representation

⇒ Each instruction is represented by a sequence of bits.

⇒ The instruction is divided into field corresponding to the elements of the instructions.

⇒ Binary Representation of instruction is difficult

So that Symbolic representation of machine instruction is used.

Ex:- D = B+C

Load B
Add C
More D, BC

⇒ Any program written in High level language must be translated into machine language to get executed, So the instruction set of the Processor must be sufficient to express any of instruction from High level language into machine language

Instruction format

⇒
- Zero Add. Instruction (PUSH A)
- One - Add^r Instruction (Load D, Add C)
- Two - address instruction (MOV X, A)
- Three - Address - instruction (SUB Y, A, B / Add T, A, B)

Ex:-

⇒ Instruction types of instruction format are depends on noumber of address specified in an instruction. there are 4 types of

instruction format

(1). Zero Add^r Instruction ⇒ Zero Add^r instruction are applicable only in stack memory organizato

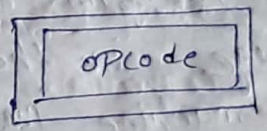
⇒ A stack is lifo (LIFO) set of locations

⇒ Ex:- PUSH, POP

OP3 = OP1 + OP2

PUSH OP1

PUSH OP2
Add
POP OR3



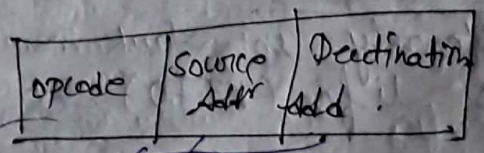
(2) One Add^r instruction ⇒ This instruction has the add^r of operand only. The second operand will be of a proceso Register called Accumulator.

⇒ The result of Accumulator move to main memory by another instructions.

Ex:-
Load D
Add C

(3) Two - Add^r instruction ⇒ It can be obtain by storing the result of operation into the memory add^r

of one of these operand.

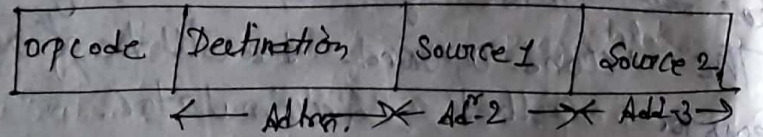


Ex. move Y, A

(3) Three add^r instruction ⇒ A machine instruction with program counter but No operand storage in the C.P.U is called 3-add^r machine instruction.

⇒ It consist of following part

- operation code.
- Address of operand called Add1 & Add2.
- Add^r of memory locations where the result of operation is to be stored.



Eg:- ADD R_1, A, B ($R_1 \leftarrow m[A] + m[B]$)

ADD R_2, C, D ($R_2 \leftarrow m[C] + m[D]$)

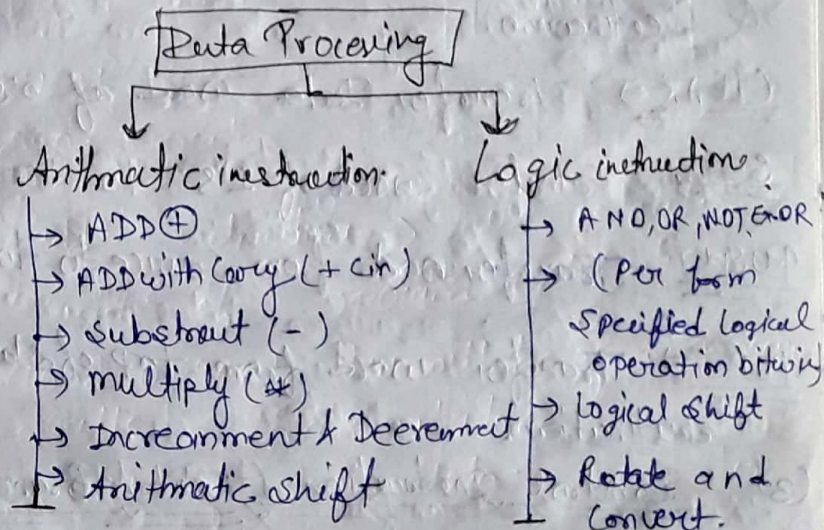
Instruction of types

- Data processing (Arithmatic & logic instruction)
- Data movement (Data transfer instruction)
- Control instruction (Test & Branch)
- I/O instruction

(1) Data Processing ⇒ Data processing instructions include arithmatic & logic instructions.

⇒ The arithmatic instruction provide computational capability for processing numeric data and logic instruction provide capabilities of performing logical operations on the bit of a word.

Ex:⇒



(2) Data Movement ⇒ These instruction are the subgroup of Data storage instruction.

⇒ These instructions are use to copy information from one location to another either in the procesor internal registor or in the External main memory.

⇒ These instructions are as follows →

Ex:- Move, Load.

MOVE :- Copy word or block from source to destination.

LOAD :- Copy word from memory to processor register.

STORE :- Copy word from processor to memory

SWAP [Exchange] ⇒ Interchange the content of source & Destination.

CLEAR ⇒ Transfer word of 0's to destination

SET ⇒ Transfer word of 1's to destination.

PUSH ⇒ Transfer word from source to top of stack

POP ⇒ Transfer word from top of stack to destination.

(3) Control Instruction (Test & Branch)

⇒ There are two type of control Instruction (that is test & Branch instruction.

⇒ Test ⇒ The Test instructions are use to Test the value of Computation that is the instruction verifies the result and transfers Control based on results.

Branch ⇒ The Branch instructions are used depending on the decision

⇒ These instructions change the sequence in which programs are executed

Jump / Branch :- Unconditional transfer, load PC with specified address ~~Jump Condi~~

Jump conditional :- Test specified condition; if True load PC with specified address.

Jump to Subroutine :- place current program Control information including.

Branch and Link ⇒ PC is known location. eg:- top of stack; jump to specified address.

Execute: fetch operand from specified location & Execute inst.

RETURN :- Restore Current program Control information

SKIP Conditional ⇒ Test specified condition, if True, increament PC to the next Instruction.

TRAP (Software interrupt) ⇒ Enter supervisor mode.

I/O Instruction ⇒ This type of instruction is for cause information to be tanfered between the processor or main memory and External Input/output devices.

⇒ INPUT (Read) ⇒ Copy Data from specified I/O port to destination

OUTPUT (write) ⇒ Copy data from source to I/O port

START I/O ⇒ Tranfer instruction to IOP to initiat I/O operation.

TEST I/O ⇒ Tranfer status information to I/O System. to destination

HALT I/O ⇒ Tranfer instruction to to IOP to terminate an I/O operation.

Instruction Cycles & Subcycles

⇒

Fetch
↓
Decode
↓
Execute

⇒ A program residing in the memory unit of the computer consist of a sequence of instructions.

⇒ The program is executed in the computer by going through a cycle for each instructions.

⇒ Each instruction cycle is divided into ⇒ Subcycles or Phases.

⇒ Each instruction cycle consist off following phases.

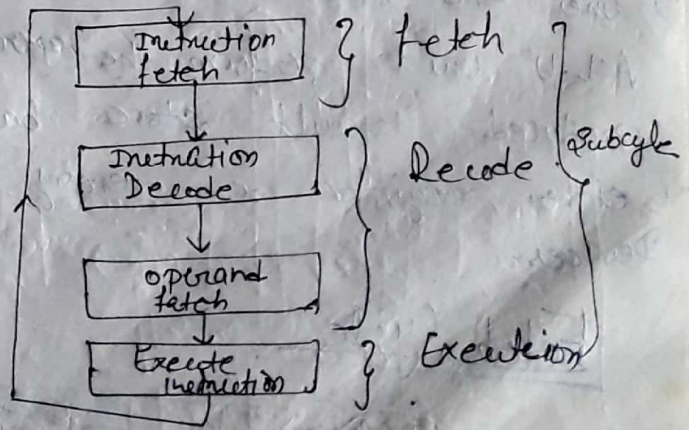
(1) ~~Phase~~ fetch ~~fetch~~ instruction from the memory.

(2) Decode the instruction

(3) Read the effective address of memory if the instruction has an indirect address.

(4) Execute the instructions.

↓ Subcycles



[fig:- instruction cycle]

Subcycle ⇒ One Subcycles have 3 Cycles & Steps have 4.

⇒ At the beginning of Program execution the processor fetch an instruction from the memory.

⇒ The Program Counter Hold the address of the instruction to be fetched.

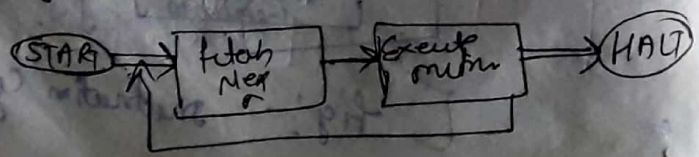
⇒ And incrementated to point the Next instruction

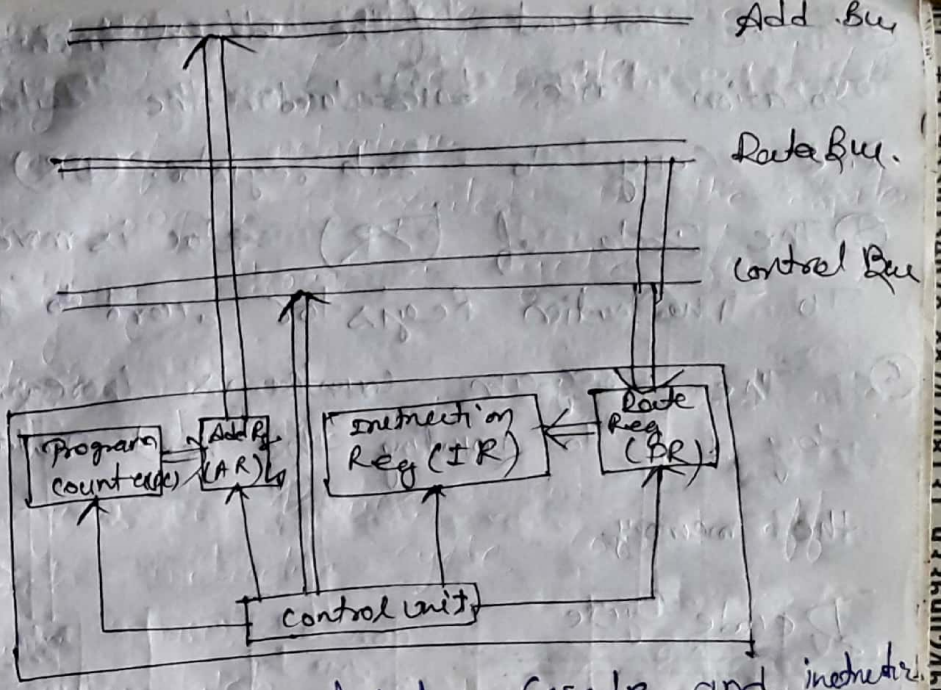
⇒ The Control unit decode the instruction to find the required operation, no of operand and the place where the operands are store.

⇒ The operands may reside in the register or can be taken from input Devices

⇒ Once the operands are available the A.L.U unit perform the operation and produced result store back into either register or memory or output Devices.

Fetch Cycle





⇒ During the fetch cycle and instruction opcode as read from memory.

⇒ The following sequence of operation take place During this cycle.

① The address of an instruction to be fetched from memory is in Program Counter (PC). This address is moved to Address register.

② The content of address register is placed on the address Bus.

⇒③ The Control unit generate the Control signals to read the content of the memory

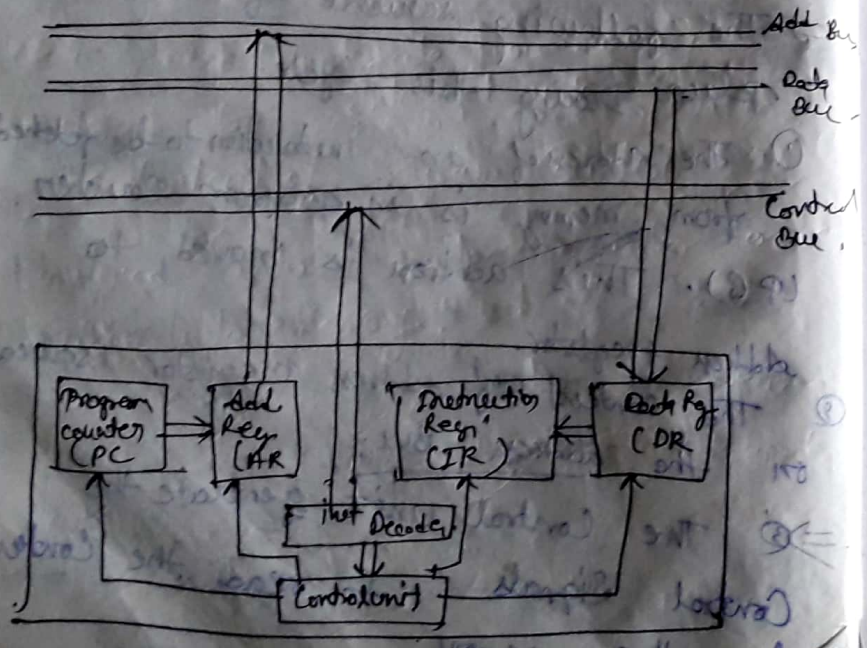
④ The Content of the address memory location are ~~loaded~~ placed on the Data Bus and Copy to Data register. (DR)

⑤ The Data of (DR) ~~data~~ is move To instruction Register acord to fig.

⑥ The program counter is increamented by 1 to pointing the next Data in the memory

Decode Cycle

⇒



⇒ In this cycle instruction register content [IR] [(opcode) (operation code)] are decoded by instruction Decoder.

⇒ During Decoding process it is determine that the operands are present in C.P.U Registors.

⇒ The Data Register content is the address refrence.

⇒ The control unit genrate the control Signal to read the address of the operand in the Data Register.

[Execution Cycle] →

⇒ In this Cycle the actual operation is ~~operat~~ perform.

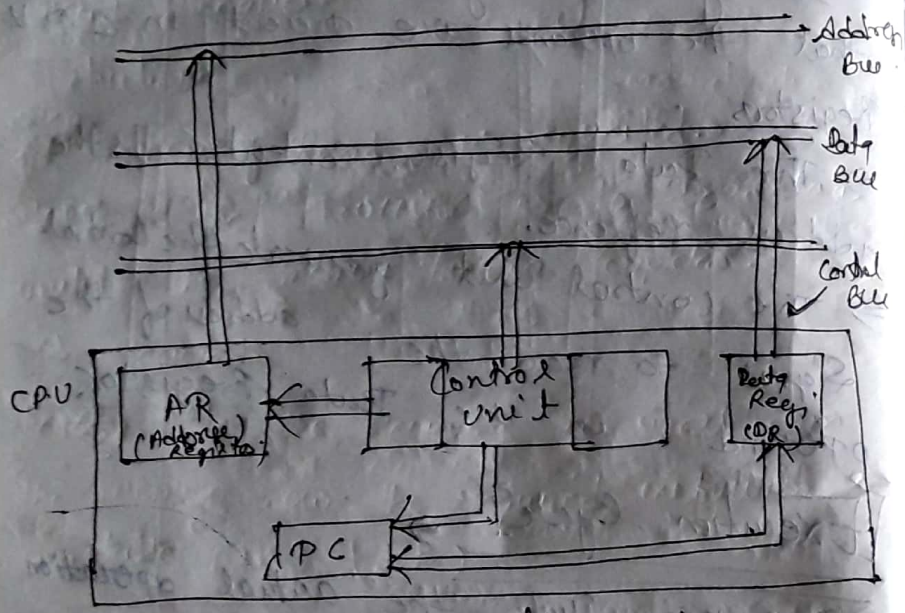
⇒ The cycle may involve Registors, Read & write from memory or Input/output & Arithmatic & logical operation of A.L.U

[Interrupt Cycle]

• During interrupt Cycle the Content of Program Counter

is save so that the C.P.U can resume normal activity after servicing the interrupt.

⇒



[fig:- Data flow during Interrupt cycle]

- The Content of Program Counter ⇒ is Transfered into Data Register to be written into memory.
- The special memory location is reserve for this purpose. for Example stack.

Eg:- Stack.

- This memory location is loaded into Address Register from the control unit. It can be a stack pointer.
- The interrupt services routine address is loaded into the program counter, so the next instruction cycle with start by fetching first instruction from the interrupt services routine

Execution of a Complete Instruction

⇒ The Execution of instruction requires the following steps operation

- fetch the instruction
- feth the operand (content of memory location pointed at by the address field of the instruction
- perform operation.
- Load the result into Register

Execution Time ⇒ It is a time require to Complete one instruction called as Execution Time

⇒ Instruction Execution required following steps to proceed Execution.

[1.] step The instruction fetch operation is initiated by loading the content of program counter into (AR) and sending read Request to memory to read the instruction.

step [2] → memory Read is requested.

Step 3 → The processor is to be delay until the memory function completed. Signal is Recieved.

Step 4 → After the recieving the signal from the memory now the required instruction is available in (IR).

Step 4 → The content of memory are tanfered by memory read operation.

Step 5 → The Content of registor are Tranfered to the input off (A.L.U) unit.

* now the operand are available in A.L.U unit.

⇒ Step 6 → The main operation is perform.

Step 7 → The result is tranfered to registors

⇒ The end Signal in S7 indicate that this is the last step of the current instruction, and it causes a new cycle to begin by returning to step1

⇒ The sequence of step required to perform three steps use single Bus organizatin of C.P.U

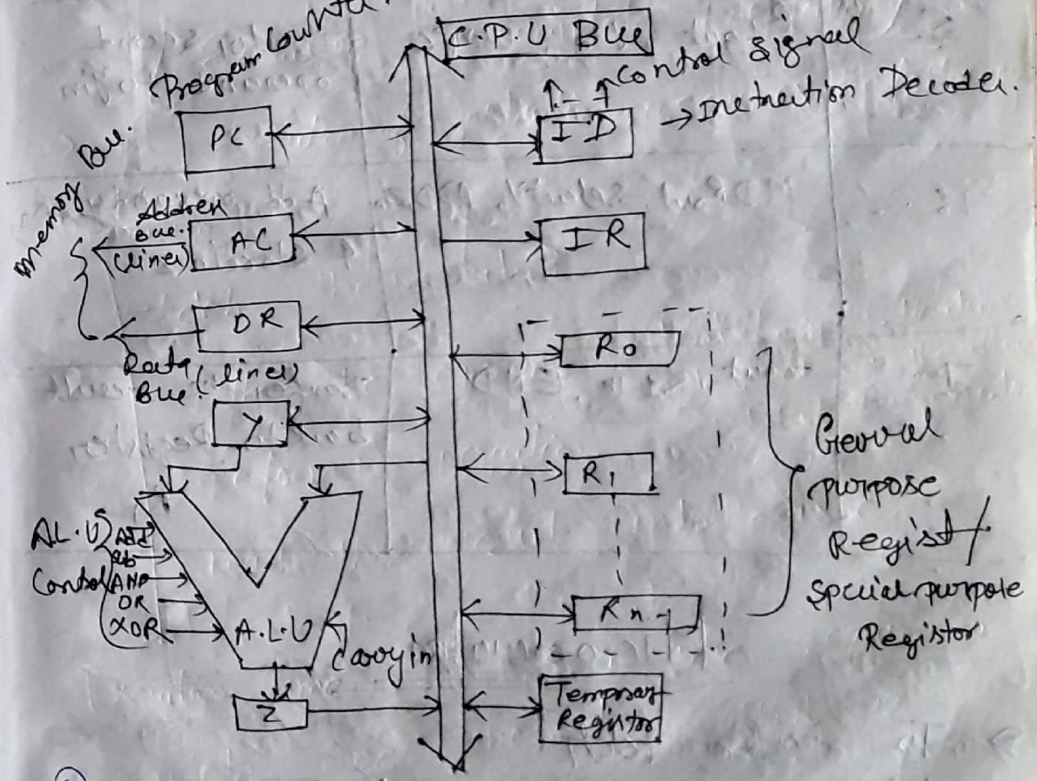


Fig:- C.P.U BUS

Fig:- Single Bus Organgotin

ADD R1,[R2]

Step	Action	Comment
1	P_{cout}, MARin, Read, select 4	→ Load PC in MAR & Read to memory.
2	Zout, PCin, Yin, WMFC	→ Load PC with next address, wait until memory respond
3	MDR_{out}, IRin	→ load instruction from MDR to IR
4	R_2 out, MARin, Read	Load Read & it operand pointed by R2
5	R_1out, Yin, WMFC	Transfer second operand to Yin
6.	MDRout, selectY, ADD, Zin	Add operands & R, & store in Z
7.	zout, R1 in, END	Transfer the result back to Register R_1

Microoperation

⇒ To perform fetch

⇒ decode and execute operations

⇒ processor unit perform microoperation

- X, O Y, Z 2 temporary reg → used only by C.P.U for temporary storage during execution [Programmer cannot access these Registers]

- DR, ALU and interconnecting bus is reffered to a data path.
- Register R_0 --- R_{n-1}, includes general purpose and special purpose Register. (Stack, Index, reg. etc.)
- 2 - options for I/P of the A.L.U' –
  1. mux is used to select one input.
  2. it select either output of y reg (when I/P = 1) or a constant no. as an I/P for the A.L.U.
- According to select I/P
  - → select constant no. when I/P = 0
  - → constant no. used to increament PC

[Fetching word from memory]

⇒ opcode fetch cycle gives the opcode of fetching a word from memory to C.P.U.

⇒ opcode specifies address of memory location where the word is stored.

⇒ C.P.U Transfer address of word to address register, which is connected to address

line of memory bus.

⇒ To read word C.P.U activate Read signal.

⇒ Memory copy Data from Address Register to Data Bus.

⇒ C.P.U then read Data from Data Register and load to specified Register.

⇒ The MFC (Memory fetch Completed) all control signal for memory transfer. 1 → Transfer Completed, 0 → No Transfer

⇒ if MFC = 1 (it means content have been read and available on the bus)

MFC = 0 (it means content have not been read and not available on the bus)

MFC = 1 → Ex:- Address of memory location to be accessed in R_1.

⇒ Memory content to be Loaded in R_2

- AR ← $[R_1]$
- Read signal
- wait for MFC signal
- R_2 ← (DR)

Storing A word In Memory

⇒ To store word in memory the C.P.U ⇐ perform opcode fetch cycle and operand write cycle.

⇒ C.P.U invoke operand write cycle address is loaded into (AR).

⇒ Data word first loaded to (DR) and Right Command issue to memory.

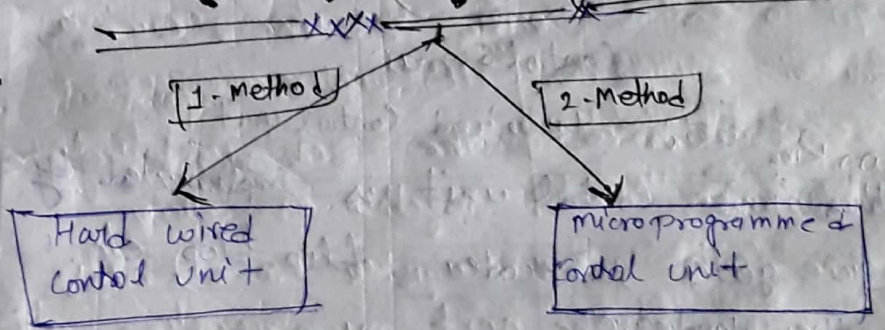
⇒ On Recieving write command Data word is written into memory.

Eg:- Data word to be store in R_2 and memory address in R_1

- AR ← $[R_1]$ · Address place in AR
- DR ← $[R_2]$ write: Data placed in DR and send write signal to memory
- MFC = 1 : data written to specified address.

Design of Control Unit (Book)

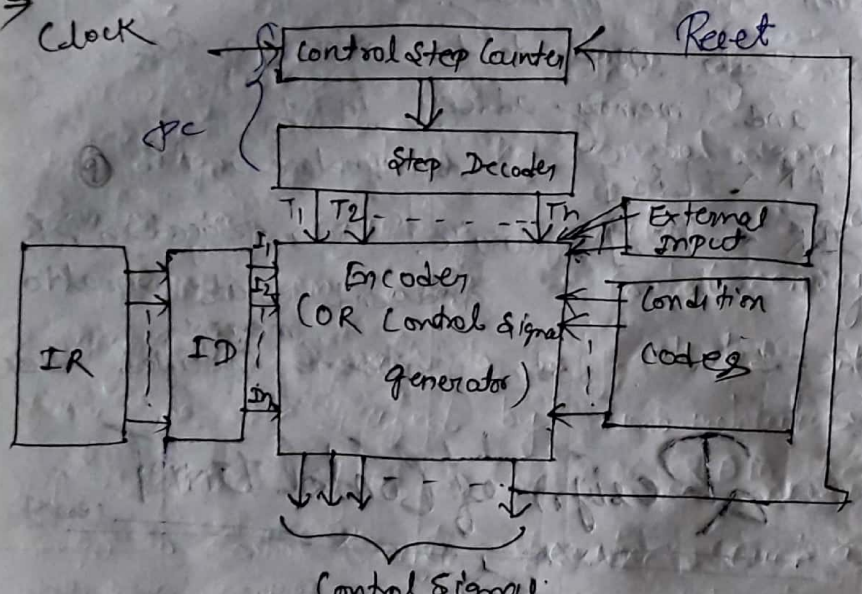
⇒



⇒ Control unit generates Timing and control signals for the operations of the Computer System

⇒ The Control unit Communicate with A.L.U and main memory it is also Control the Transmuttion between Processor, main Memory and various peheripheral Devices.

[Hard wired Control unit]



[Fig!- Hardwired Control unit]

① working

⇒ Hardwired control unit is implemented by logic gates, Decoder, flip flop etc. in the Hardware.

⇒ The Control signals are generated as an output of logic gates.

⇒ The instruction Decoder Decodes the Instruction loaded in the IR. (IR)

⇒ if the IR is an 8-bit Register then instruction Decodeder generates 28

⇒ 256 lines, 1 for each instructions.

⇒ According to code in IR only one line is activated at a time.

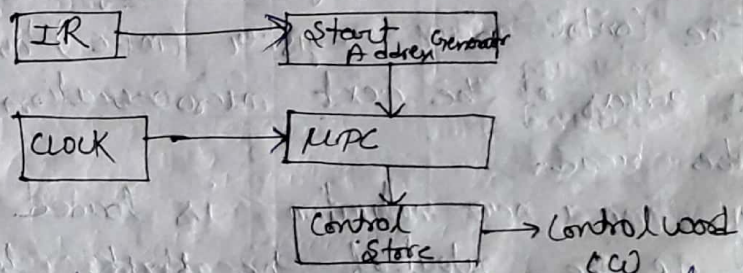
⇒ The Step Decoder Provide a separate line for each step or a time slot in a control sequence.

⇒ The Encoder gets the input from the ID, step Decoder, external Input and Contional codes.

⇒ And generate a individual Control Signal.

⇒ After execution of each instruction the Control signal is reset Control step counter and ~~may~~ make ready for next instruction.

[Microprogrammed Control Unit]



➔ In this machine Control Signals are generated by a Program similar to the machine language Programme.

⇒ Control word ⇒ It is word for which individual bit represent various Control Signals each of the Control

step in the control sequence of an instruction. Defines a unique combination of 0's & 1's in the control word.

⇒ microprograming is method of control unit design in which they control signals selection and sequencing information is stored in ROM or RAM called a control memory.

⇒ They control signals to be activated at any time are specified by micro instructions.

⇒ A sequence of one or more microoperation design to control specific operation such as addition, multiplication etc is called a microprogrammed. ⇒ The Component of the control unit work together as follows.

(i) The control address register (MPC) holds the address of the next microoperation instruction to be read.
Every time a new instruction is loaded into the IR, the output of the block level starting address generation is loaded in MPC.

(ii) when address is available in control address register the sequencer issue read command.

(3) After issue the read command the word from address location is read into the micro-Instruction register.

(4) The MPC is automatically increamented by clock.

(5.) The control word issue the control signal in the form of 1's & 0's. In this word individual bits represent the various control signals.

Advantage of Micro-program Control :-

1.) It simplifies the design of control unit
2.) control function are impliment in Software.
3.) It is more flexible and can be changed to acomidate new system specification or to correct the devision error quickly.
4.) Complex function such floating point arithmatic can be realize efficiently

Disadvantages :-

1) It is slower then hardware control unit
2) It require extra hardware cost to achieve flexibility.

Difference b/w Hardwired & Microprogram

Hardwired	Microprogrammed
① Speed is fast	① Speed is slow
② Implement in hardware	② Software
③ Ability to handle large & complex instruction is difficult	③ Easier
④ Designing is difficult for more operation	④ Easy
⑤ Ability to support operating system and diagnostic features is very difficult	⑤ Easy
⑥ No memory used	⑥ RAM or ROM used
⑦ Used in RISC (Reduced instruction set Computing)	⑦ Used in CISC (Complex instruction set Computing)

Microprogram Sequencing

- **Micro instruction →** A micro instruction contains the control signals and sequencing information
⇒ A simple way to structure micro instruction is to assign one bit position to each control signal (i,c) required in the C.P.U.

- **Micro program sequencer ⇒** The basic component of micro program control unit are the control memory and circuit that select the next address.
⇒ The address selection part is called micro program sequencer.
→ A micro program sequencer is a general purpose building block for micro program control unit.
⇒ The main purpose of micro program sequencer is to present an address to the control memory so that micro instruction can be read and executed.
⇒

⇒ The task of micro program sequencing is done by micro program sequencer.
⇒ There are 2 factors that must consider while designing micro program sequencing.

① ⇒ **Size of micro instruction ⇒**

② The address generation time

The size of micro instruction ⇒ should be minimum show that the size of control memory required to store micro instruction is also less. This reduced the cost of control memory.

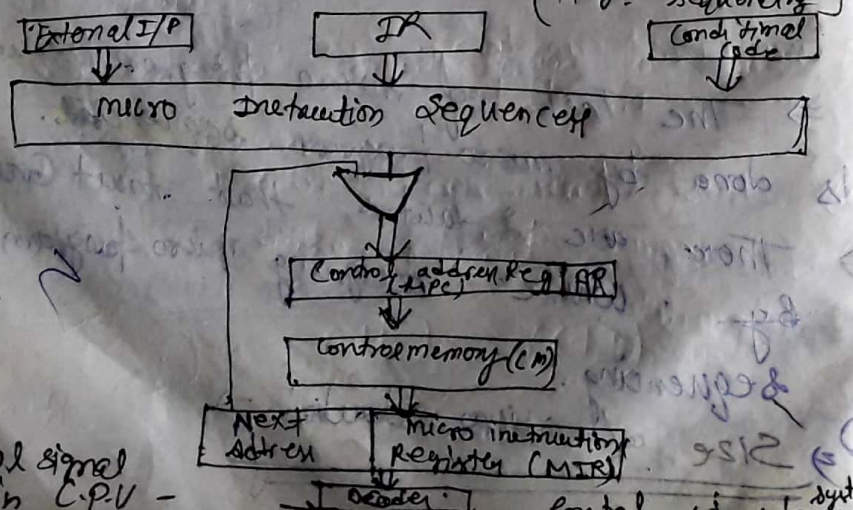
NOTE ⇒ with less address generation time the micro instruction can be executed in less time, resulting better output.

⇒ during execution of microprogram the address of next micro instruction to be executed as 3 sources.

(1) Determine by IR ⇒ occurs only once per Instruction Cycle.

(2) Next sequential address ⇒ store add of separate micro instruction

(3) Branch (Condition code) ⇒ micro program sequencing



⇒ ② The fig shows a Block diag of Control memory and Associative hardware needed for selecting the next micro instruction address.

⇒ ③ The micro instruction in Control memory Contains a set of bits to initiate micro operations in computer Register and other bits to specify the method by which the next address obtained

⇒ ④ The next address bits are fed through the (selector) or gate to the micro add^r Register or μPC show that the address can be modified on the Basis of Data in the IR.

⇒ ⑤ The MIR is loaded micro instruction and is Decoded to produce the required control signals.

⇒ ⑥ A micro instruction Cycle can be executed faster then instruction cycle because micro instructions are stored in the C.P.U, whereas instruction must be fetches from an external memory.

Horizontal and Vertical microprogramming

⇒ Horizontal M.P

1⇒ Horizontal Approach has each bit for each signal that is required to Control all microprogramming level Components.

2. This means that programmes Need to use every bit in the Micro instruction for each Controll.

3. Each Control Signal is represented by a bit in micro instruction.

Vertical M.P

1. Vertical Approch only we the opcode that represent micro instruction that it needs.

2. To understand the opcode a decoder is required.

3. It needs a small storage to Store micro programmes.

4.

Ex:- Instruction format.

X = (A+B) : (C+D) into

3-add, 2-~~Add~~, 1-Add

UNIT-III Memory organi

Memory Organization

⇒

Reg
↓
Cache
↓
M.M
↓
S.M

MEMORY ➡ Memory is an evential Component of Digital Computer we to Store Programmes and Data.

⇒ There two types of Memory.

Memory
- Main Memory (Primary Memory)
- Secondary Memory

⇒ **Main Memory** ➡ The main memory is a fast memory.

⇒ It stores programmes along with data which are to be executed.

⇒ It also store necessary Programmes of system software, which are required to execute users programmes.

⇒ It is a volatile memory. It means all the data loss their content when the power is switch off.

Ex:- RAM

Secondary Memory

⇒ Secondary memory is permanent storage use to store programmes and Data.

⇒ The size of secondary memory is very large.

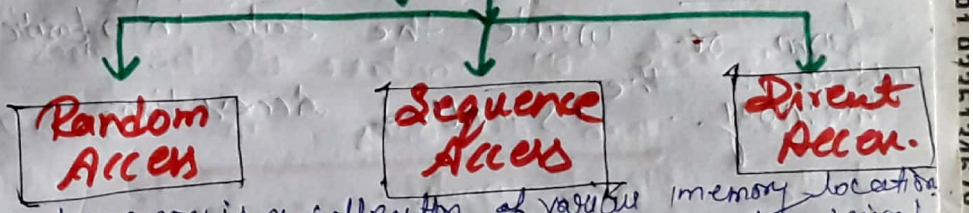
Ex:- Hard disk, Compact disk, magnetic tape, magnetic disk, optical disk etc.

Cache Memory

⇒ It is very fast and expensive memory use to remove the mismatch of speed between processor & main memory.

⇒ It exist between the C.P.U & Main memory

Memory Access Method



- Each memory is a collection of various memory location. Accessing the memory means finding and reaching desired location and then reading the information from that memory location.
- There are three access method

Random Access ⇒ In this method each memory location has a unique address using the unique address each memory location can be address independently. In any order in equal amount of time.

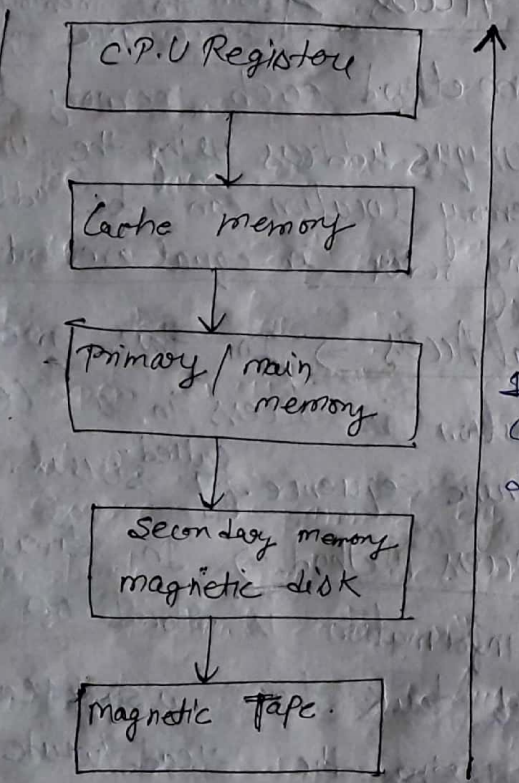
Sequencial Access ⇒ In this method memory location is access in a certain predetermine sequence called serial.

Direct Access Method ⇒ In direct Access method information is stored on tracks and each track has a separate read/write head. Using the read/write head information of that track would be Accesit sequentially.

This features makes it a semi random mode and generally used in magnetic disk.

Access time ⇒ The time required to read from or write the Data into particular address in the memory is called Access time.

Memory Hierarchy



Memory Classification

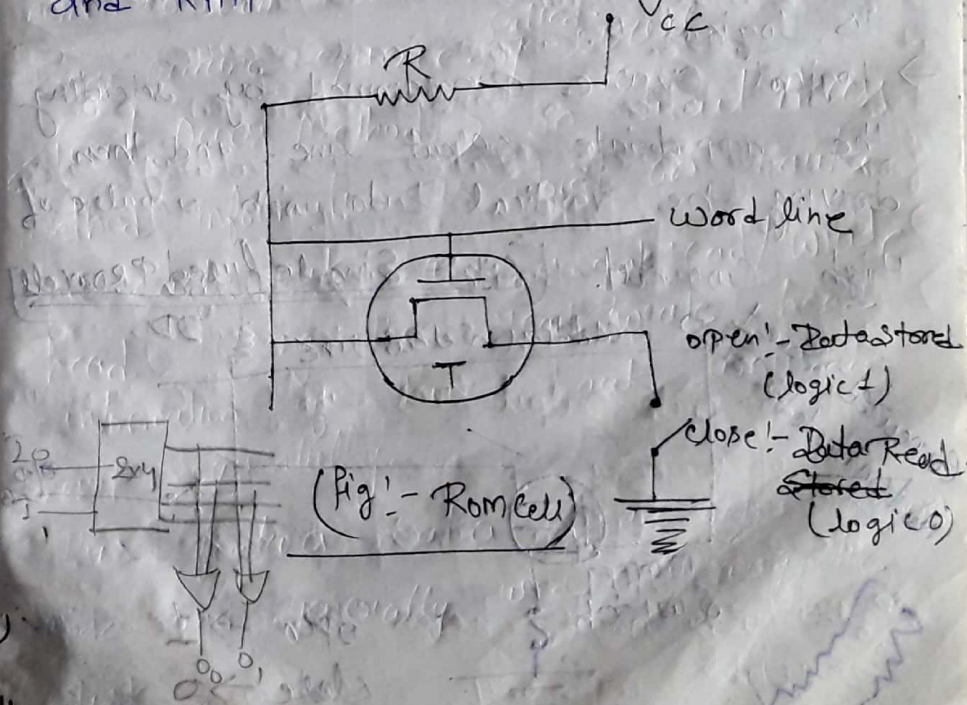
- RAM (Random Access Memory)
  - Static RAM (SRAM)
  - Dynamic RAM (DRAM)
- ROM (Read only memory)
  - PROM (Programmable)
  - EPROM (Erasable)
  - EEPROM (Electrically)

ROM

⇒ This is Read only memory the Data can't be write in this memory it is non volatile that is it can hold Data if power is switch off.

⇒ Rom is use to store Binary codes for the sequence of instructions in the computer and the Data. the size of Rom is large.

⇒ And it is cheaper then Cache memory and RAM



(Fig! - Rom cell)

PROM [Programmable Read only Memory]

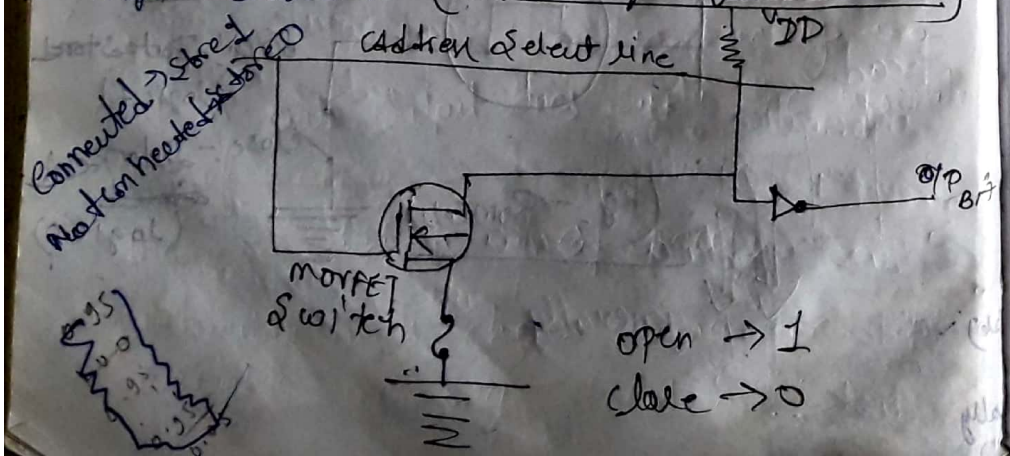
→ Programmable ROM contains all the fuses intact given all ones in the bits of the stored words.

⇒ It provide programmable facility

⇒ Each address select Data line instruction and has on mosFET.

⇒ when the fuse is intact the memory cell is configur as logic 1 and when the fuse is Blow down that is open circuit the memory cell is logical 0.

⇒ Logical 0's are programmed by selecting the appropriate select line. and then driving the vertical Data with a pulse of High current. [Fig:- Single fused PROM cell]



→ EPROM:- In EPROM once fixed pattern is permanent and can't be altered or changed * the erasable PROM can be Restructure to the intial state even though it has been previously programme.

⇒ when EPROM is placed under A special Ultra-violet life for a given period of time the data are erased.

⇒ After erasen the data EPROM return to it intial state and can be programme to a new set of values.

⇒ # EEPROM

⇒ Electically EPROM also used mosfet circuit and very similar to the EPROM.

⇒ Data is stored as charge or No charge on an insulated layer.

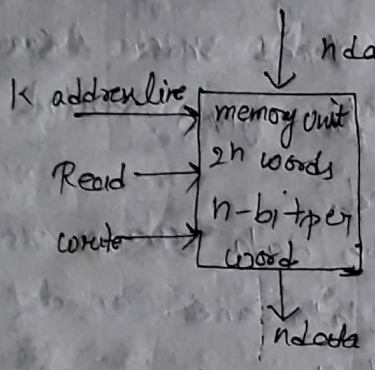
⇒ In EEPROM programmed connection can be erased with an electrical signal instead of ultra violet light.

⇒ EEPROM allows selected erasing instead of erasing all informations, since the information can be change by using electrical signal.

→

RAM (Random Access Memory)

RAM ⇒



[Fig:- Block Diagram of Ram]

➔ RAM is a semiconductor memory.

⇒ It is also known as read/write memory it is called Random Access because the time it takes to transfer information to or from any desired location is always same.

⇒ It is a volatile memory.

⇒ The n data input lines provide the information to be stored in memory and n-data output lines supply the information out of memory.

⇒ The k-address line specify the particular words chosen among the many available memory location.

⇒ To control signal that write and read is use to transfer the data into the memory and Read the Data from the memory.

➔ Types of RAM

- Static RAM (SRAM)
- Dynamic RAM (DRAM)

SRAM

⇒ The Static RAM retains its content as long as power is supply. However Data is lost when the Power is switched off. Due to the nature of volatile. SRAM does not have to periodically refereced.

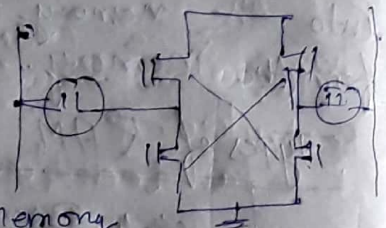
⇒ A flip-flop is use as a memory cell in the SRAM. So the flip-flop are Constructed by using MOS Transistors

⇒ Static RAMs chips use a matrix of 6-transistors.

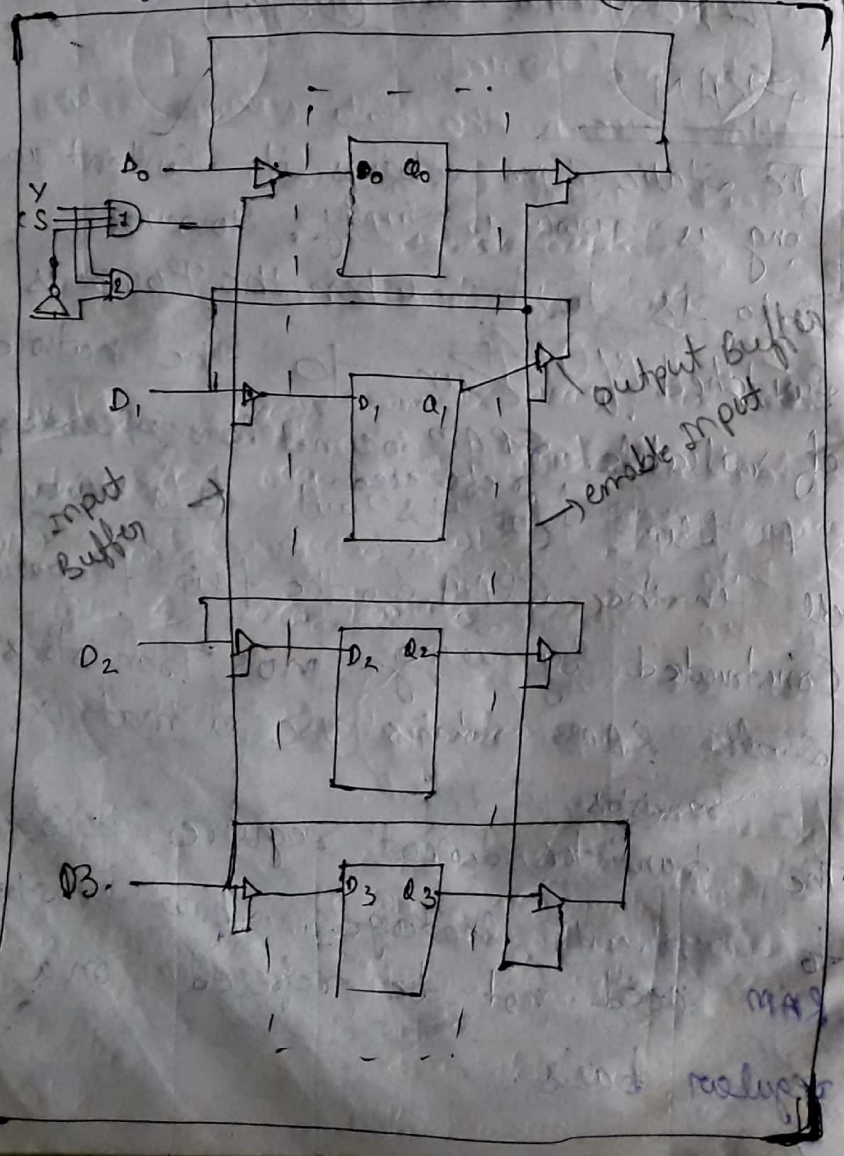
The transistors do not require power to prevent leakage, show the static RAM need not be refreed on a regular basis.

Characteristics of SRAM (Static RAM)

(1) faster
(2) Expensive
(3) Use transistors
(4) used in Cache Memory
(5) high speed Consumption



(Basic Structure of SRAM)



1.) The Basic Structure of Static Ram has 4 D flip-flop for read & write operation and the Data Bus is represented D_0 to D_3 that Carry the Data

write operation

- Gate 1 Enable - when chip select CS = 1 and $R/\bar{W} = 0$
- O/P of gate → to enable i/p of all i/p buffer
- Y (address select line) → select the specified register (has 4-bit data for each register)
- Data is loaded onto D-flip-flop.

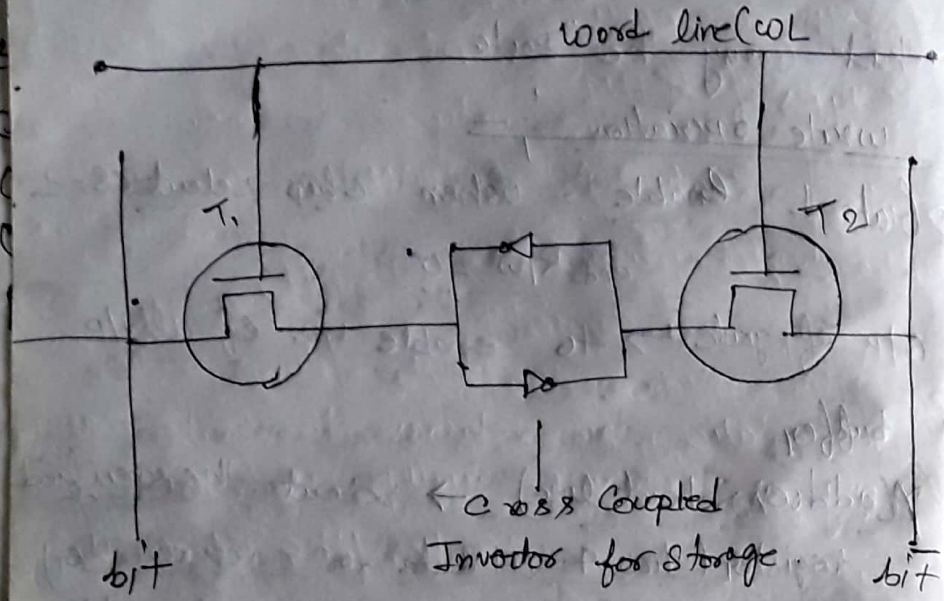
Read operation

- Gate 2 Enable - when chip select CS = 1 and $R/\bar{W} = 1$
- O/p of gate 2 is Connected to Enable i/p of output buffer
- The 4-bit data is read from the specified register

Advantage
① Refreshed is not require

Disadvantage
① high cost
② need more current

6-T cell :-



T_1 & T_2 ⇒ Access transistor, access to stored data for read and write.

WL ⇒ Control Access [WL=0, hold operator]
[WL=1, read/write operation]

3-operation

Hold
↓
WL=0 access transistor
Data core off
• Data holding latch

write
• WL=1, Access transistor on
• New Data applied to bit
• Data in latch overwrite with new value

Read
• WL=1, Access transistor
• bit
Sense amp
operation

[D-RAM]

Date 26/09/16
Page

① Dynamic RAM → Dynamic RAM is a semiconductor main memory.

② ⇒ It retains Data for only few mill seconds and must be refreshed periodically.

③ ⇒ It stores each bit of data on a small Capacitor within the memory cell.

④ ⇒ The Capacitor can be either charge or Discharged and this provides two states '1' & '0'. for the memory cell.

⑤ ⇒ It Contains 100.0 of memory cells.

⑥ ⇒ It requires only a single transistor and provide much higher level of memory density.

Advantages of D-RAM

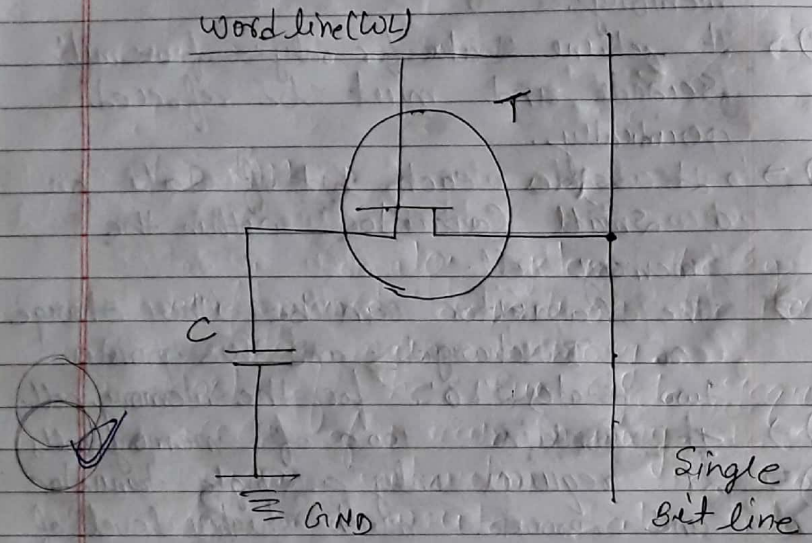
(1) High dense
(2) Low cost
(3) Simple memory cell structure
(4) Used in main memory.
(5) Large size

Disadvantages of D-RAM

(1) Data requires refreshing
(2) Slow operation speed
(3) Complex manufacturing process
(4) volatile

Date / /
Page Ayaki

DRAM cell:-



- T - used to controll access to cells
- C - Capacitor, used to store charg(1), discharge(0)

⇒ Two lines are connected to each Dynamic Ram cell that is word line (WL), single Bit line. So that the required cell can have Data read or write to or it.

⇒ **To write :-**

- The value into cell (0 or 1) is placed on single bit line & word line is enable.

Date / /
Page

- This charge the capacitor if '1' to be stored and discharge if a '0' is to be store.

To Read :-

- To read the value of Capacitor is examine
- when the address line is selected, the transistor is open and the charge stored on the capacitor is fled out on to Bit line

~~Bit line~~

⇒ **Difference between Static RAM & Dynamic RAM**

	Static RAM	Dynamic RAM
1.	It contains less memory cell per unit area	1. Contains 1000 of memory cell per unit area
2.	faster access, less access time	2. slow access, more access time
3.	high Cost	3. Cost is Low
4.	Use flip flops	4. Use Capacitor
5.	Used in cache memory	5. Used in main memory
6.	Refreshing is not required	6. Refreshing is required after few millisecond
7.	memory is static	7. Dynamic

Date / /
Page

RAM Organization

3 method to organize the memory.

1-DRAM 2-DRAM 2½-DRAM

1-D RAM →

For a Computer System the System performance depends on the processor and the main memory.

⇒ Most of the Bus Cycle require access of main memory to read Program instruction and to read or write Data into the memory.

⇒ The Ram memory can be organize By three 3 method.

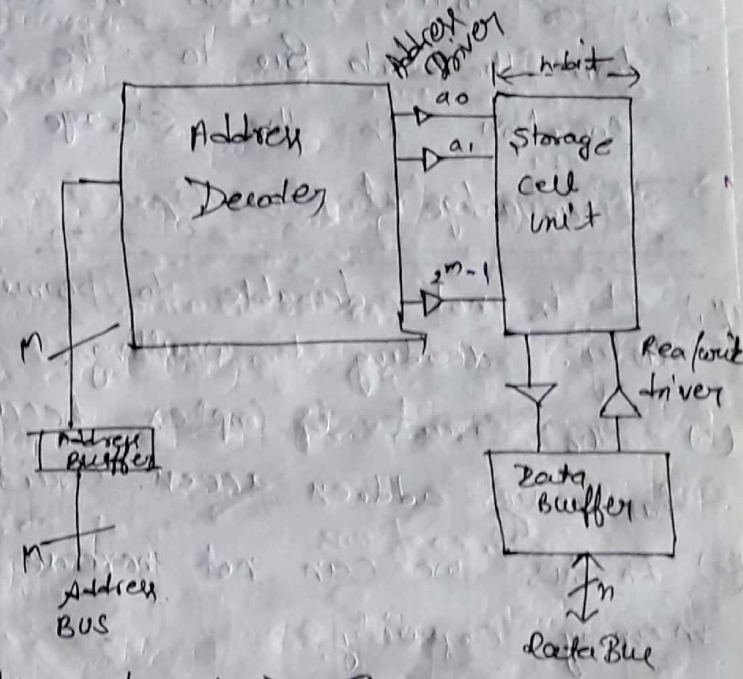
1. 1-DRAM
2. 2-DRAM
3. 2½-DRAM

1-Dimensional Memory (RAM)

⇒ The Dynamic RAM is the main Component of main memory because it has more memory cells per unit area Compare to Static RAM.

⇒ It has 2^m Storage location, each address store n bit word.

⇒ Capacity of RAM is = $2^m \cdot n$ (address lines × no of bits per address)



Operations of 1-D RAM

1⇒ Finding the address of target location to be accessed is transfer by address Bus to the Ram address Buffer (Address Decoder).

2. This address is then Processed by addr Decoder. which select the required location in the storage cell.

3⇒ For Read operation the content of addressed location are transfer from the storage cell unit to the Data Buffer & then to the Data Bus.

4⇒ for write operation the coord is transfer from the Data Bus to the selected location in the storage cell unit.

Disadvantage of 1-D RAM

(1.) It is not desirable to permit simultaneously reading & writting.

(2.) The Circuit is Complex because it has only address accen Circuit.

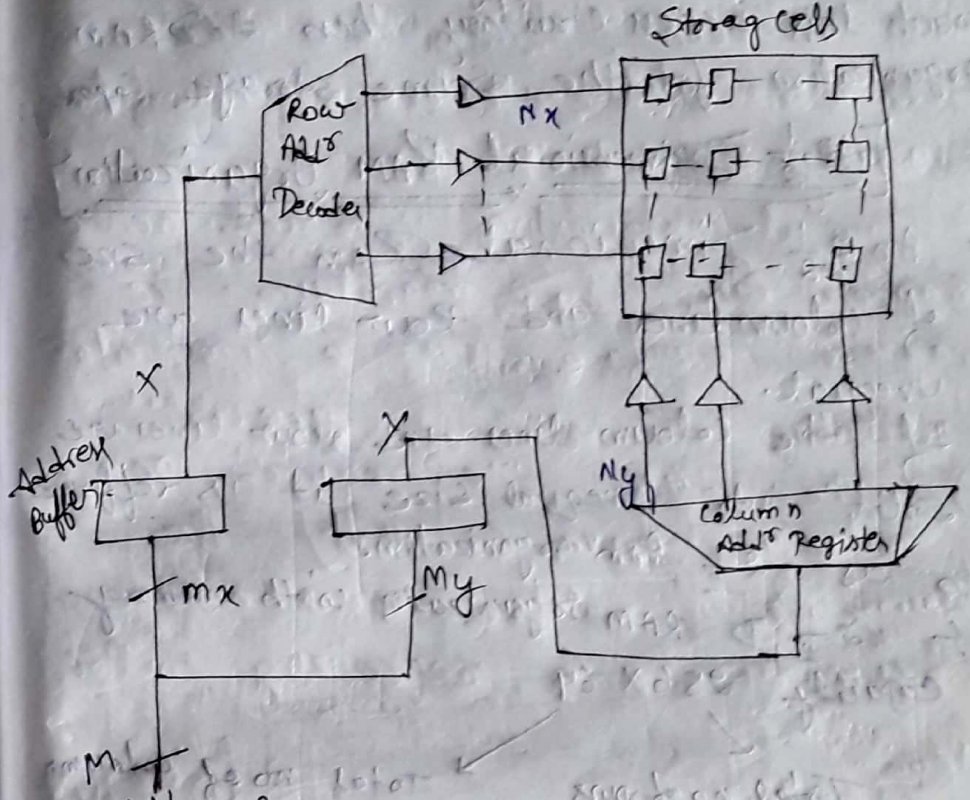
(3.) In this we can not increase the memory capacity.

(4.) To Solution this's problems are 2-D RAM.

2-D RAM

⇒ The 2-D RAM is suitable for large memory Capacity organization.

⇒ For large Capacity memory the storage cell is divided into Row & column address.



⇒ Here the m bit address coord is divided into two part x & y for row address selection & column address selection respectively.

⇒ The cells are arranged in a rectangular array of $N_x \leq 2^{mx}$ rows & $N_y \leq 2^{my}$ columns.

mx & my → No of bits for row address and Column address.

⇒ So the total no memory cells in 2-D Memory organization → $N_x \cdot N_y$

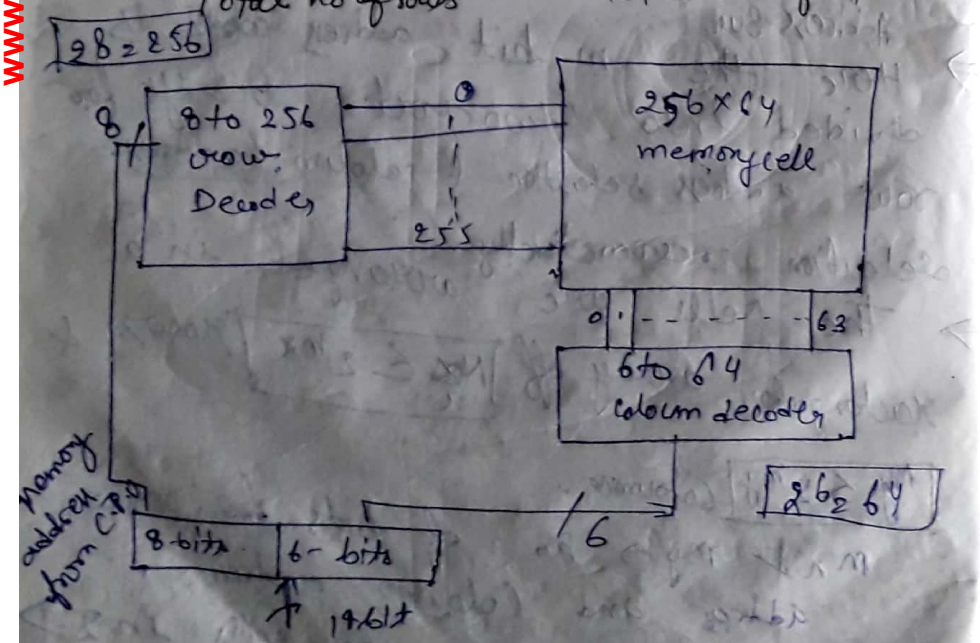
⇒ The 2 Dimensional organization Requires much less access circuitary then 1-D RAM organization for the same storage capacity.

[2½ Dimensional RAM Organization]

⇒ An 2½ Dimensional RAM the size of column line and Row line are unequal.

⇒ If the column line and row line are split into unequal size it is reffered to as 2½ RAM organization.

Eg:- 2½ D RAM organization with memory capacity 256 × 64



⇒ Two Decoders are use to implement this organization.

⇒ 1-Decoder is called Row Decoder and other 1 is called Column Decoder.

⇒ The memory consist of 256 Row and 64 Colum.

⇒ Each Row is selected by activating any one row at a time by using row Decoder.

⇒ Similarly Colum Decoder select 1-colum at a time.

Auxilary Memory

⇒ Large storage requirement of a Computer System is fulfilled by Auxilary Memory.

⇒ The most Common Auxilary Memories are ① magnetic disk ② magnetic tape.

Magnetic drums, optical disk are also auxilary Memory but it is not used frequently.

⇒ A Hard disk & floppy disk are important Component in all micro Computer and mini Computers System.

(i) Magnetic Disk ⇒ A magnetic Disc is a thean, Circular metal plate.

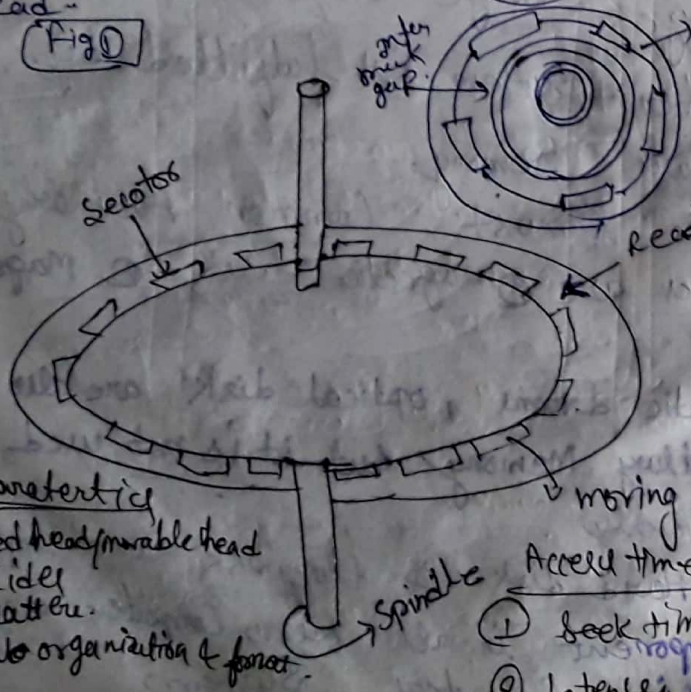
⇒ It is coated with thin magnetic film, usually on both side.

⇒ Digital information is stored on the magnetic disc by magnetizing the magnetic surface in a particular Direction.

⇒ The disk are mounted on a rotary drive so that the magnetized surface moves in close proximity to magnetizing coil or Head.

⇒ Fig 1

Fig 2



Characteristic

① fixed head/movable head
② Sides
③ platters.
④ Data organization & format.

Access time on Disk

① Seek time
② Latency time

Seek Time ⇒ Position the head on a specific track.

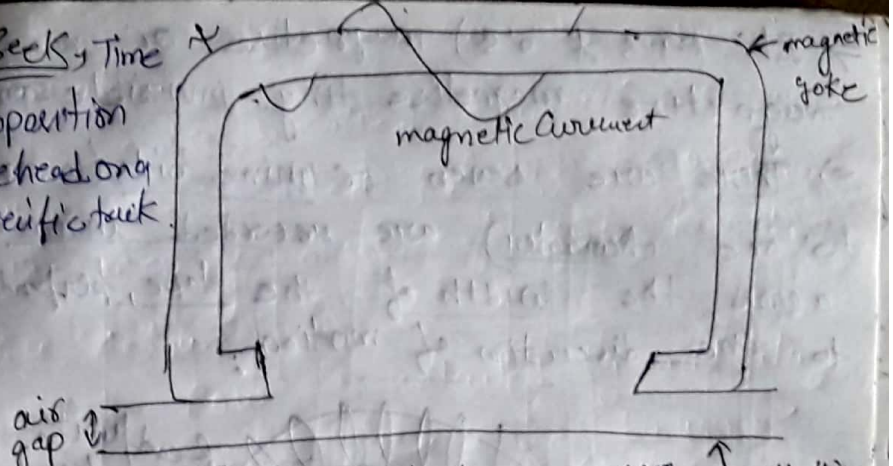
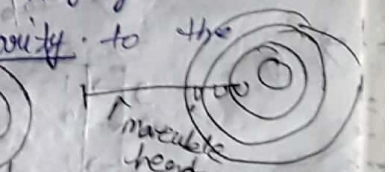


Fig:- Read/write Head details

⇒ The Head Consist of a magnetic yoke (frame) and magnetizing coil.

⇒ The digital information can be stored on magnetic film by applying current pulse of suitable polarity to the magnetizing coil

(ii) magnetic tape



⇒ magnetic tape is the most popular storage medium for large Data that are sequencially access at procesed.

⇒ The tape is formed by depositing magnetic film on the tape.

⇒ In magnetic tape iron oxide is used on a magnetizing material.

⇒ The information is recorded on the tape with the help of read write head.

⇒ It (1's & 0's) magnetise or non-magnetise magnetise tiny invisible spot.

⇒ There are seven or nine bits (corresponding to one character) are recorded in parallel across the width of the tape, perpendicular to the direction of motion.

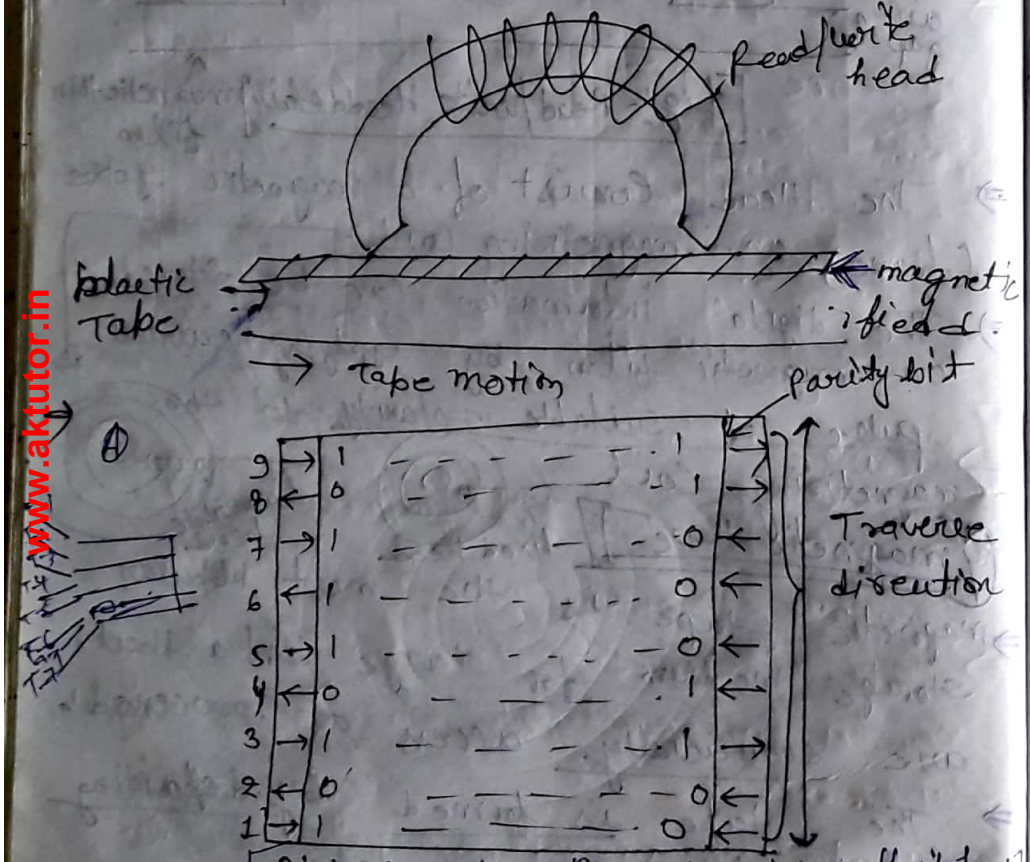
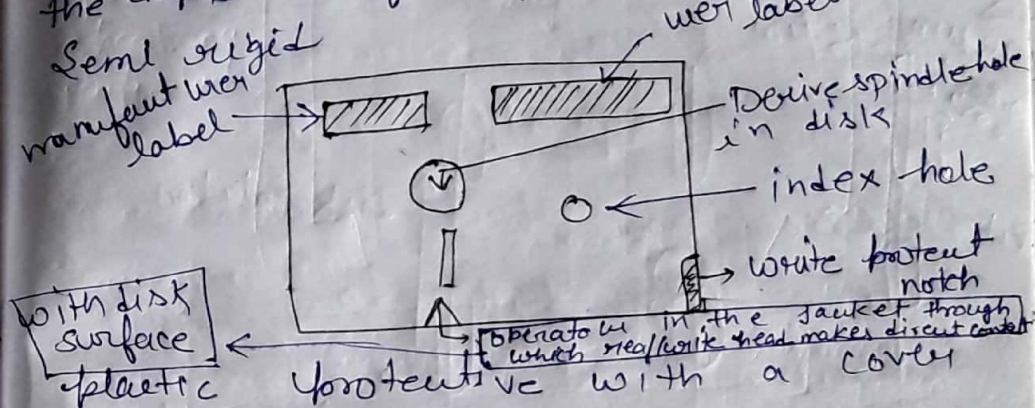


Fig:- Magnetic Recording with read/write head

⇒ A separate read/write head is provided for each bit position on the tape so that all bits of character can be read and write in the parallel spot.

Floppy Disk ⇒ A floppy disk is made by the deposition of magnetic material on semi rigid manufacturer label



as show in fig.

⇒ The read/write opening provides access for the read/write head and the index access hole allow the use of photosensor to stablize a reference position. when the write protected notch is covered, the data can not be recorded on the disk, prevending accidental loss of data.

Optical Memory

- 1983
  - Compact disk(CD)
  - 60 min audio Media
  - Non-erasable
- Types:
  - CD-ROM
  - WORM
  - Erasable

Optical Memory (1983)

- CD-ROM
  - CD
  - Compact
  - non-erasable.
- WORM (Write once Read many)
- Erasable optical Disk

⇒ The first optical memory device is the compact disk CD. The CD is a non erasable disk that can store more than 60 min of audio/video information.

There are three types of optical memory.

CD-ROM

⇒ CD-Rom is a similar to the compact disk it we similar technology as we'd in compact disk.

⇒ The Binary data is store in the form off 0.1 μm wide pits.

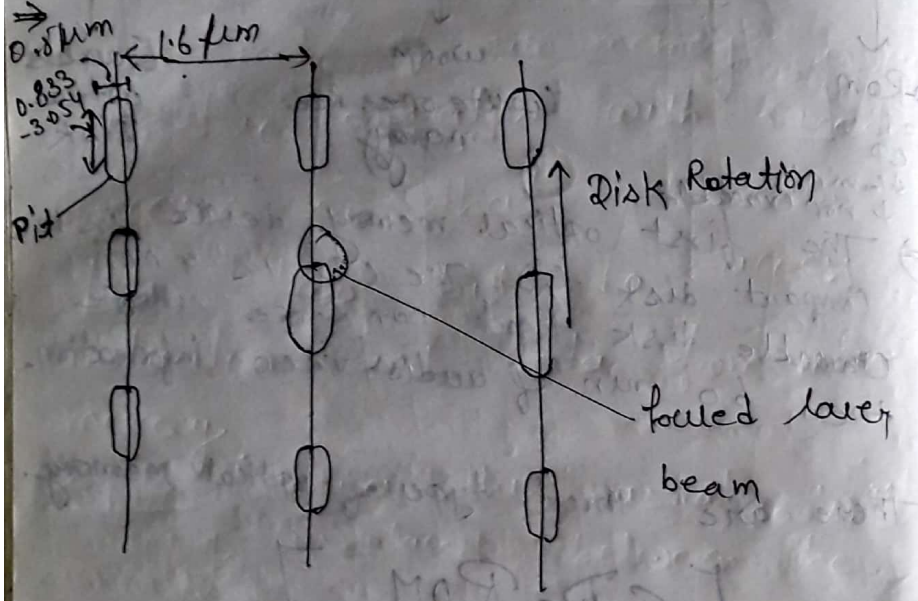
⇒ In a circular track on a plastic substrate.

⇒ A laser beam scan the tracks to read & write the data

⇒ A standard CD-Rom has capacity of 600 mb.

⇒ The access time is about 100 millisecond

and the data is transfered from the disk at a speed of 3.6 mb/s → mbps.



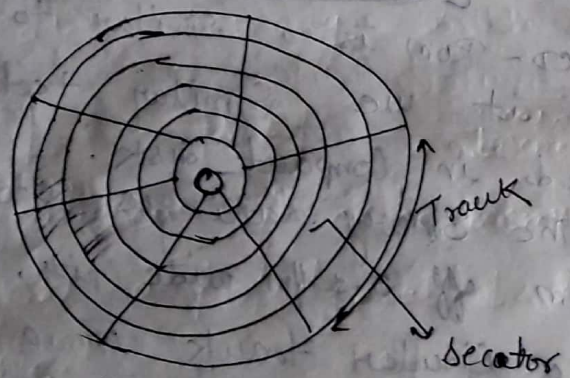


Fig:- Recorded portion of showing pits along with tracks

WORM (Write Once Read Many)

⇒ In Application where 1 or a small number of Copy of data is needed the WORM CD is used.

WORM storage mobiles one time writing provide and unlimited Reading of the data. the data can'nt be over written or erased but can be updated by writting new information into a file at another location. The new file is then linked to the original file by a Software.

Erasable optical disk

⇒ These storage media are used in application where store Data requires frequent changes.

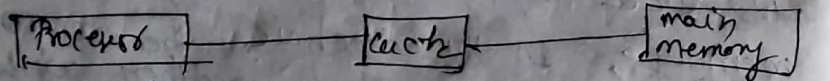
⇒ In these magnet optic technology is used to produce Read/write memory, In this technic the memory of laser beam is used to gather with a magnetic field to record and erase the Data or information.

Ex:- pen drive

CACHE MEMORY

⇒ The speed of main memory is very low in comparation with speed of Processor So that Processor has two bit to access instruction and data from main memory.

⇒ To reduces this mismatch a fast memory is placed between the processor and main memory, is called Catch memory



⇒ The effectiveness of Cache memory is based on a property called locality of refrence.

[1.] Program locality ⇒ In Cache memory The prediction of memory location for the next address is essential.

⇒ This is possible because Computer System access memory from Conjugative location this prediction is known as Program locality.

[2.] Locality of Refrence ⇒ Analysis of a Program shows that most of the execution time spend on routines in which many instructions are exected repeatly.

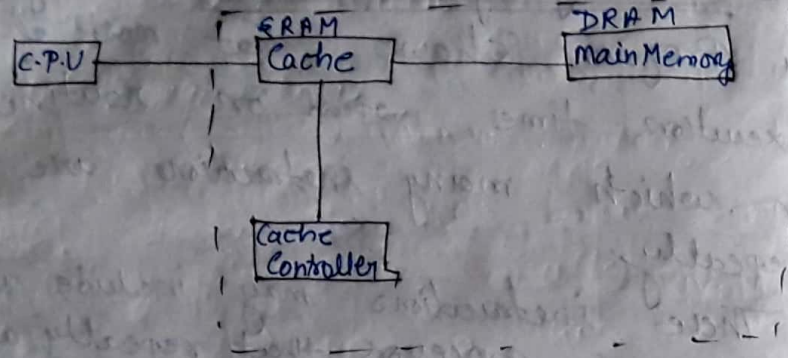
⇒ These instructions may include a loop or procesors that repeatly calls each other SubRoutine

⇒ If the active segements of the Program can be placed in a fast memory then total execution time can be reduced significantly.

⇒ The Correspondence between main memory blocks and Cache block is specified by a mapping function.

⇒ when cache is full and a memory word that is not in the Caches is refrenced, the cache control hardware must decide which block should be remove to create space for new block.

Cache memory system



Hit Rate ⇒ (or Hit Ratio) $\frac{\text{no of hits}}{\text{no of read/write bus cycle}} \times 100\%$.

→ The % of access where the processor find the data in the cache memory is called hit rate.

→ The Hit rate is normally greater then 90%.

→ ~~levels of~~

[Levels of Cache / Types of Cache]

L1-cache
(Primary or internal cache)
Built in processor use
Capacity:- 8KB – 128KB.
Speed:- Same as Processor

L2-cache
- Called secondary cache
- Placed after the primary cache
- 64KB – 256KB
- Slow then L1

L3-cache
- Seprate from Processor on the motherboard or chip board.
- Size upto 8MB
- Speed slower than L1 & L2

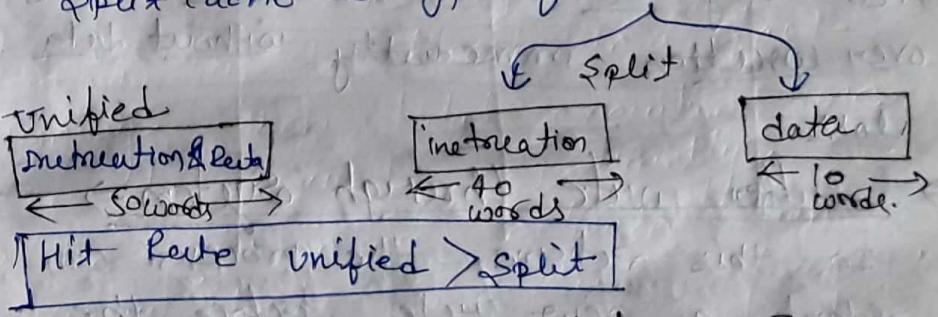
[Unified and Split Cache]

Unified Cache.

- Stores instruction and data at same place.
- Unified Cache is a type of cache.

Split Cache

- Split Cache is seprated into two part 1 part can only use ~~that~~ Store instruction and second part is use to store Data.
- Split Cache is type of Cache



Hit Rate unified > Split

[Write Policy (Cache updating)]

⇒ In Cache System the Copies of Save Data can exist at a time one in Cache Cache and other one in main memory.

⇒ If 1 copy is altered and the other is not, to different sets of Data become associated with the same address.

to prevent this the Cache System is updated such as

(1) write through policy.
(2) Buffer write through system
(3) Write back policy

Write through policy

⇒ In this scheme the cache Controller copies data to the main memory immediatly after it is written to the cache memory. ⇒ due to this the main memory always Contain a valid Data and any block in the Cache Can be over written immediatly without data loss.

Buffer write through System ⇒

In this scheme the procesor can start a new cycle before the write Cycle to the main memory is Complited. This means that the write accesses to the main memory are buffered.

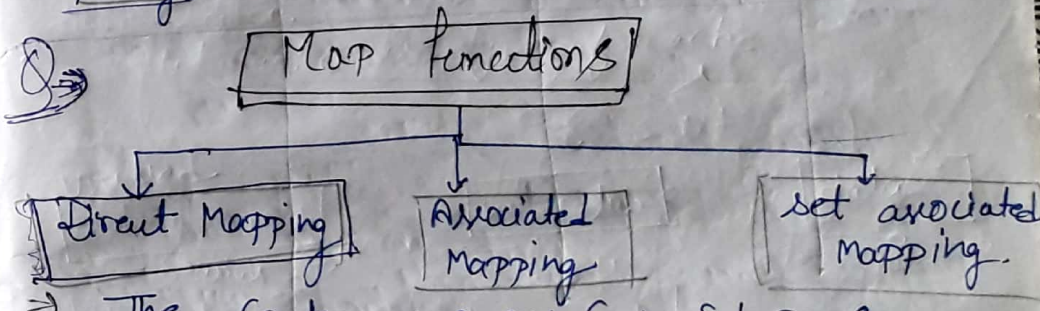
⇒ in such System the Cache location & main memory location are Updated Simanteanously.

Write back policy

LRU left optimal right

⇒ In this method only the Cache location is updated during the write operation.
⇒ the location is then mark updated with an associated flag bit often called modified bit.
⇒ The main memory block will be updated letter when this block will be remove, this is called write back policy.

Q ⇒



⇒ The Cache memory Can store a reasonable number of blocks at any given time. but this no is Small Compare to the blocks in the main memory.
⇒ Thu a mapping function is use to relate the main memory blocks and Cache memory blocks.
⇒ There are Three 3 mapping function show in above fig! —

Page replacement algorithm

⇒ Consider a case of cache memory consisting 128 blocks 16 word each for a total of 2048 words (128×16)

⇒ Assume that main memory is addressable by 16 bit address so main memory has 64K words.

Memory	Mem Capacity	No of blocks	No of words in a block
1. Cache	2048 (2K)	128	16
2. Main memory	64K	4K	16

⇒ The Group of 128 blocks forma Page in main memory

Direct Mapping

⇒ The main memory blocks can be placed in one and only one place in the cache memory, this mapping is called direct mapping.

⇒ It is the simplex mapping technic In this technic each block from main memory has only one possible location (fixed place) in the cache memory

⇒ The block j of main memory map to the j mod 128 of the cache.

⇒ The placement of a block in the cache is determine by the address of main memory.

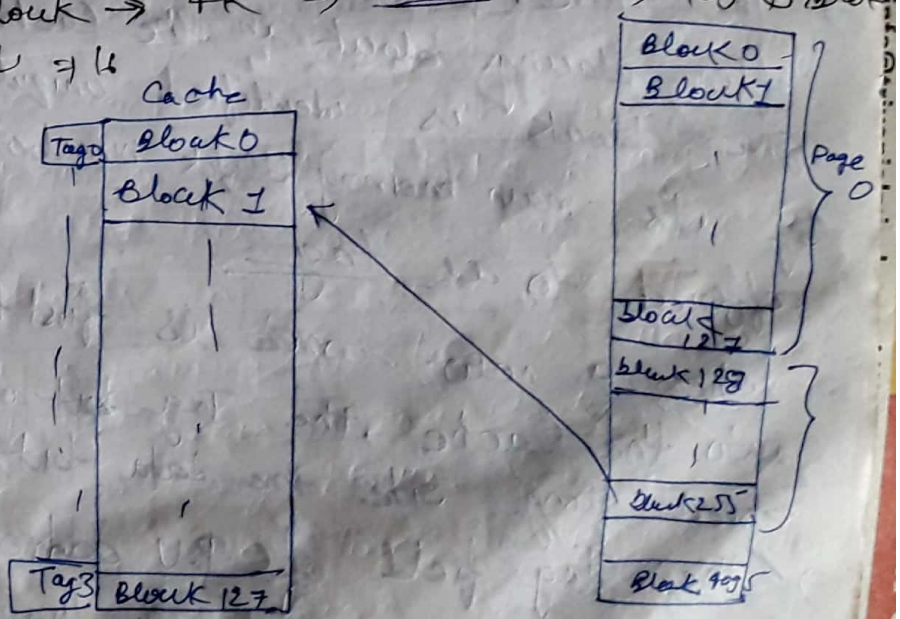
⇒ This address is divided into 3 field. that is

Tag	Block	Word

Main Memory → 64K

block ⇒ 4K ⇒ 4×1024 ⇒ 409 blocks

word ⇒ 16



Cache → 2048

block → 128

word → 16

2^{16} ⇒ 16 - 7 - 4 ⇒ 5

main Memory Add

Tag	Block	word
5	7	4

5-bit $= 2^5 \Rightarrow 32$

word field

⇒ The lower order 4-bit select one of the 16 words in a block. This field is known as word field.

⇒ The second field is known as block field and use to distinguish between blocks of the memory or the block from the other block.

⇒ The third field is Tag field - after placing block in the cache a given block is identified uniquely by its main memory block number it is reffered to as tag

⇒ when a new word is first brought into the cache, the Tag bit are stored along site the data bits.

⇒ The Tag field of C.P.U address is Compared with the tag in the word read from the Cache if - if the two tags match there is a Hit. (read from cache memory) if there is no match then it is miss (Data is not available in the cache, Data read from main memory).

Associated Mapping

⇒ This technic faster and flexible then Direct mapping.

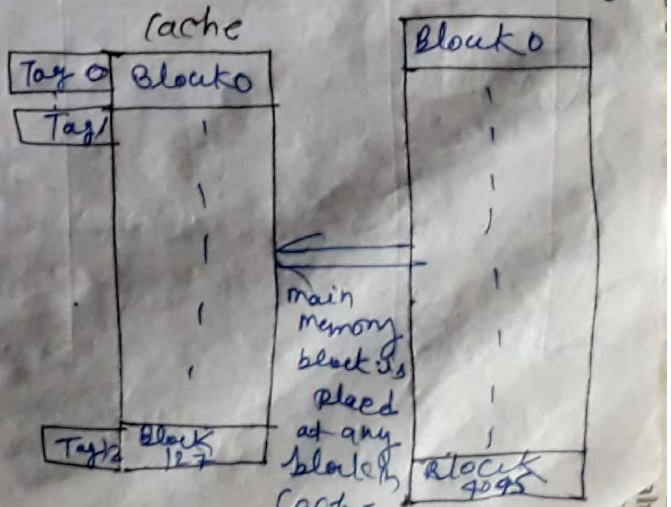
⇒ In this a main memory block can be placed into any block in cache.

As there is no fixed block, the memory address has only two fields (i.e) word field and Tag field.

⇒ This technic is also called fully associated Technique.

Memory add

Tag	word
12	4



Q1 A direct map cache consist 256 slots. Main memory contains 32K blocks of 16 words. find Tag, block & word bits

Ans block → 256 → 2^8 → 8

word → 16 → 2^4 → 4

Size of main memory → 32×2^{10}

→ $2^5 \times 2^{10}$

→ 256×32

→ $2^8 \times 2^5$ → 213

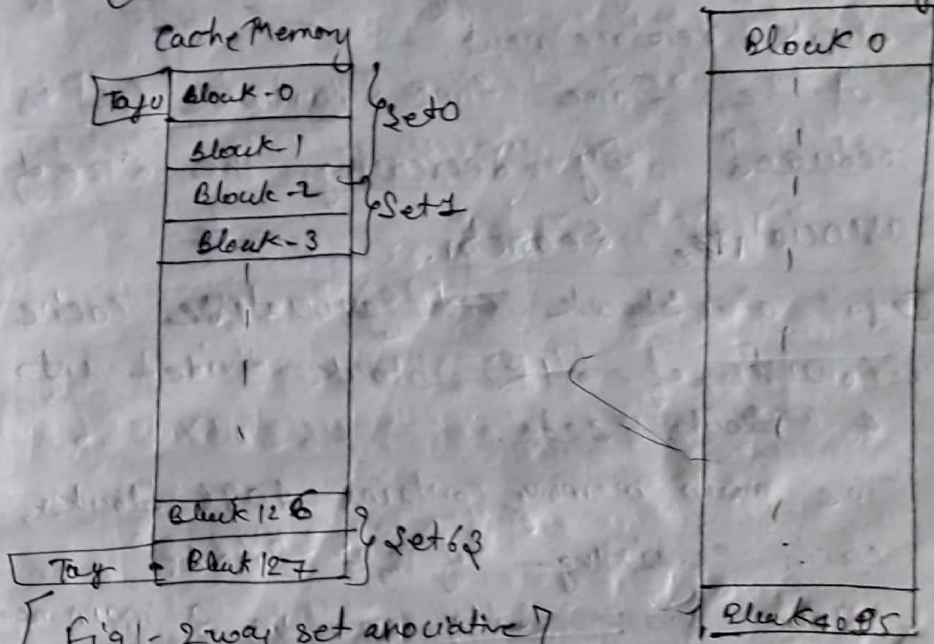
→ $2^{13} \times 16 \times 4$ → $2^{13} \times 2^6$

→ 2^{19}

⇒ 19 - 8 - 4

7	8	4

Set Associated Mapping



[Fig:- 2 way set associative]

Tag	set	word
6	6	4 bit

⇒ The set associated mapping is a combination of both direct mapping and associated mapping.

⇒ It takes advantage of both direct and associated mapping.

⇒ The block of the cache memory are grouped into sets. and the mapping allow a block to reside in any block of a specific set.

⇒ So the problem of direct mapping is remove by having some choice for block replacement.

⇒ at the same time hardware cost is reduced by decreasing the size of associative search.

Ex: a block set associative cache consists of 64 block divided into 4-blocks sets.

The main memory contains 4096 blocks, each consisting of 128 words.

(i) How many bits are in a main memory addres.

(ii) How many bits are there in Tags & words bits

(iii) what is the size of cache memory

Ans: size of main memory = 4096 × 128

= $2^{12} \times 2^7$

⇒ 2^{19}

①

Total bits for main memory address = 19 bit

⇒ ① words = 128 ⇒ 2^7

Total bit = 7 bit

⇒ ② Sets = 64 ⇒ $\frac{64}{4}$ ⇒ 16 ⇒ 2^4

No of bit ⇒ 4-bit.

block = 64

Tag

Tag	blocks	words
8	4	7

19 bit (memory address)

19 - 4 - 7 ⇒ 8

19, 4096

② Size of cache ⇒ 19 bit.

Q A set associative cache consist of 64 slots divided into 4 slots sets the main memory contains 4K blocks of 128 words each. So the format of main memory address.

Ans Tag ⇒ ?

word = 7

(set) block = $\frac{64}{4}$ ⇒ 16 block ⇒ 2^4

words = 128 no. of bit

size of main memory ⇒ 128 × 4096 ⇒

⇒ $2^7 \times 2^7$

⇒ 2^{19}

Total bits of main memory address = 19 bit

words ⇒ 4

block ⇒ 128 ⇒ 2^7 ⇒ 7

19 - 7 - 4 ⇒ 8

Tag	block	words
8	7	4

Eg A block set associative memory 128 blocks divided into 4 block sets & the main memory consist of 16,384 blocks and each block contain 256-8 bit word ① How bits are required for addressing main memory

② Tag/Set/word

Any Cache $\Rightarrow \frac{128}{4} \times 4 \Rightarrow 32$

$\Rightarrow$ SET 2

main memory block $\Rightarrow 16384 \times 256$

$\Rightarrow 16384 \times 2^8$

$\Rightarrow 2^{14} \times 2^8$

$\Rightarrow 2^{22}$

Blocks $\Rightarrow 32 \Rightarrow 2^5$

Words $\Rightarrow 256 \Rightarrow 2^8$ $\Rightarrow 22-5-8$

Tag	set	word
9	5	8

Virtual Memory

$\Rightarrow$ Virtual memory is a capability of operating system [memory management capability] that uses Hardware & Software to allow a computer to compensate for physical memory shortage by temporary transfering Data from main memory to Disk Storage.

$\Rightarrow$ Computer have a finite amount of RAM so memory can run out, specially RAM multiple programmes run at the same time.

$\Rightarrow$ A system using virtual memory can load larger programmes or multiple programmes running at the same time, allowing each & without having to purchase more RAM.

$\Rightarrow$ The virtual address space is increase using active memory in RAM and in active memory in hard disk to form contiguos addresses that hold both the application & Data.

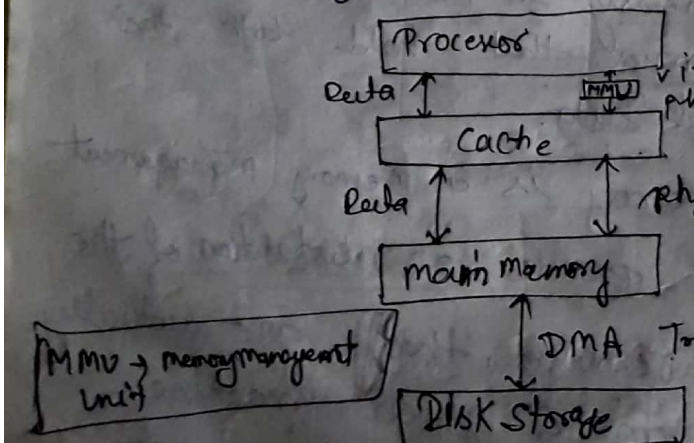
$\Rightarrow$ Virtual memory is a memory management technic that provides an abstraction of the Storage resources that are actually available on a given machine, which creates the illusion to user of a very large main memory.

⇒ When the programes does not Completely fit into main memory then it is divided into Segements.

⇒ The Segements which are Currently being executed are kept in main memory and the remaining segements are stored in the secondary storage Device.

⇒ (1) If an executing Program needs a Segements which is not Currently in the main memory, the required segement is Copied from the secondary storage Device.

(2) When a new segement of a Program is to be Coopied into main memory it must replace another segement already in the memory.



[Fig:- virtual memory organization]

Logical address Space ⇒ The address generated by the processor is called logical address or virtual address.

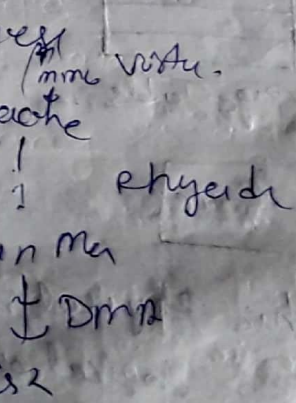
⇒ The set of such address is called logical address Space.

Physical Address Space ⇒ The physical Address Space is the range of Physical addresses that can be recognize by the memory.

⇒ The physical address is the address that represent the Data in physical memory.

⇒ The memory management unit (MMU) controlls the virtual memory System and it translate the logical address into physical Address.

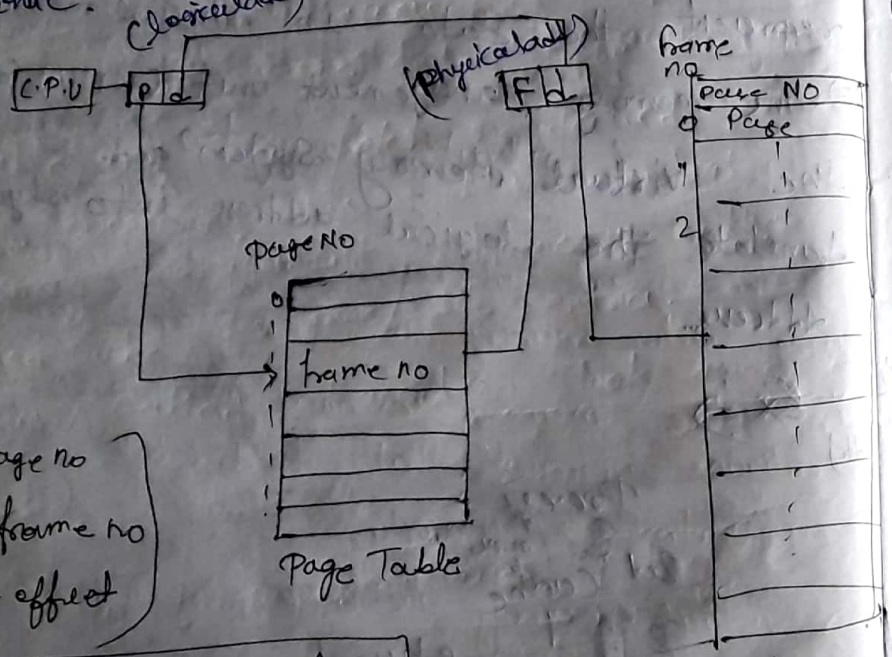
Ex ⇒



Pagging & Address translation

⇒ The mapping of logical Address to physical Address is done by the memory management unit which is a Hardware device. This mapping is known as pagging technique.

⇒ The translation of logical address to physical address is done by pagging Technic.



P = Page no
F = frame no
d = offset

Page No = Size of frame / Size

⇒ In pagging technic Physical memory is divided into frames and logical address is divided into pages. and all mapping information of pages & frames store page map table.

⇒ The logical address space is divided into fixed size blocks ~~each~~

⇒ Page No ⇒ The no of bits required to represent the pages in the logical address space.

frame No ⇒ The no of bits required to represent the frames. ~~for particular word of frame size~~ of in physical address space.

(d) Offset ⇒ No of bits required to represent a particular word in a page or particular word in frame.

Eg:- if logical add = 31 bit
logical address space = 2^{31} words

PAS space byte, bits
log AS = word

⇒ No of Page = logical add space / Page Size

⇒ No of frames = phy add space / frame size

Ex ⇒ Physical address space = 4K word & logical address space = 8K words find the logical address & physical address & No of Pages & frames ⇒ Assume page size = 1K.

Physical ⇒ 4×10^{10} ⇒ 2
P-A ↑ 212 bit
⇒ 12-bit

logical Addr ⇒ 8K × $2^3 \times 2^{10}$

⇒ 2^{13}

logical_add = 13-bit

Page size = 1K = frame size

$1K = 2^{10}$

⇒ Page NO ⇒ $\frac{8K}{1K}$ ⇒ 8

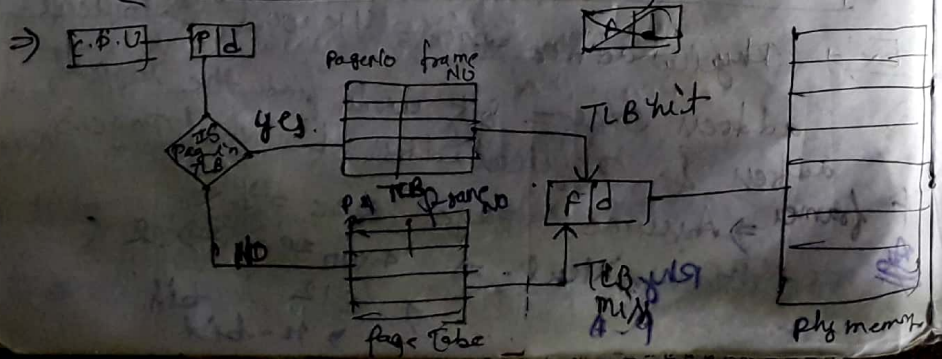
frame no = $\frac{4K}{1K}$ ⇒ 4

NOTE ⇒ The paging Technic is a time Consuming technic because it access first appropriate page Table and then fetch the desired Data.

⇒ To overcome this problem a virtual memory management Scheme we a High Speed Cache for page table entries.

⇒ This is Called Translation look aside Buffer (TLB).

TLB implementation



⇒ TLB ⇒ TLB Contains page table Entries that to have been most recently used

A TLB Contain following entries.

① Virtual page no.

② control bit

③ page frame no in the memory.

⇒ The address translation proceed as follows. given a virtual address the processor looks in the TLB for the refrenced page. if the page Table entry for this page is found in TLB, The physical addres is obtained immediatly (This is called TLB hit).

⇒ If the page entry is not found in the page Table TLB it is called TLB miss then the required entry is obtained from page table. this in the main memory and The TLB is updated. Then it is called TLB Miss.

Segementation

⇒ A memory management technic in which memory is divided into variable size chunks which can be allocated to Processes.

⇒ These memory chunks are called Segments a table stores the information about the all the segement that is called Segement Table.

⇒ The Segementation allow the program to view memory as consisting of multiple address space of Segements.

Segement Table

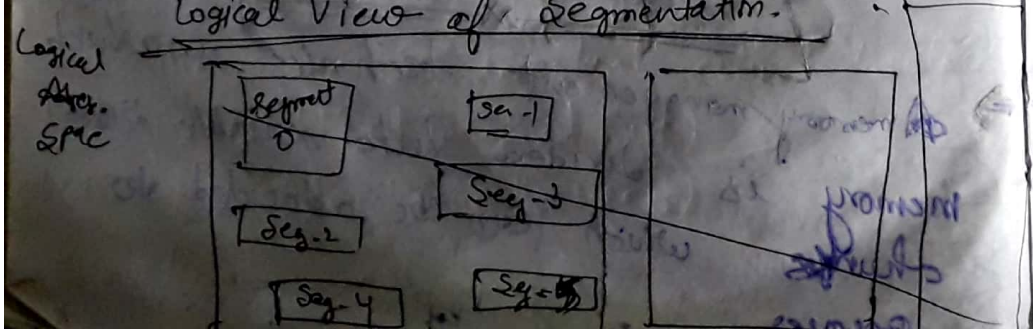
⇒ The Segement Table map two Dimensional logical address into 1-Dimensional physical address.

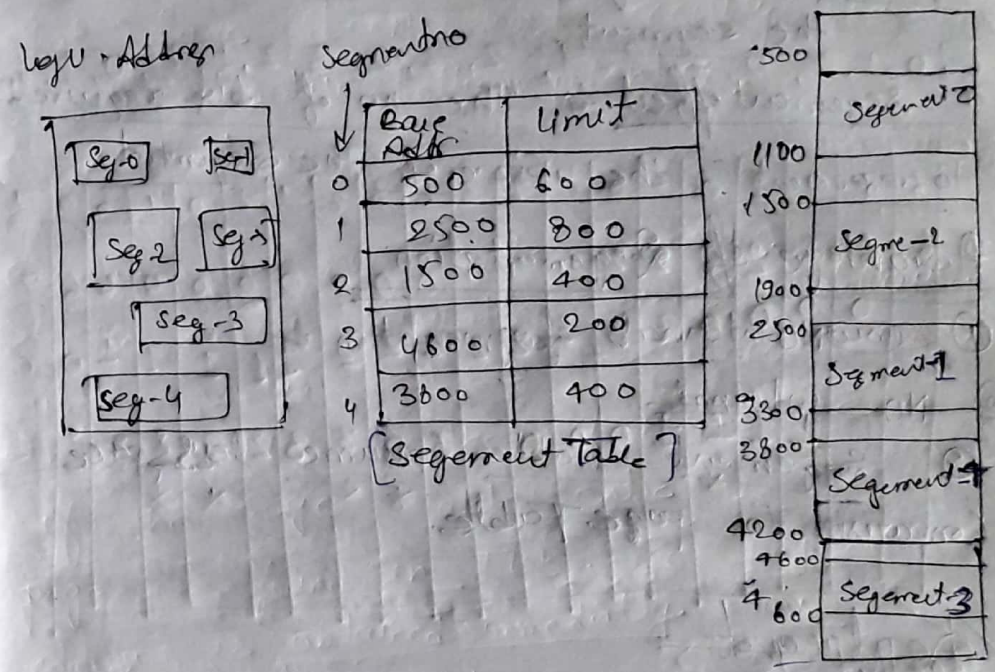
⇒ It consist of Base address and limit

⇒ Base address contains the Starting Physical address where the Segements reside in memory.

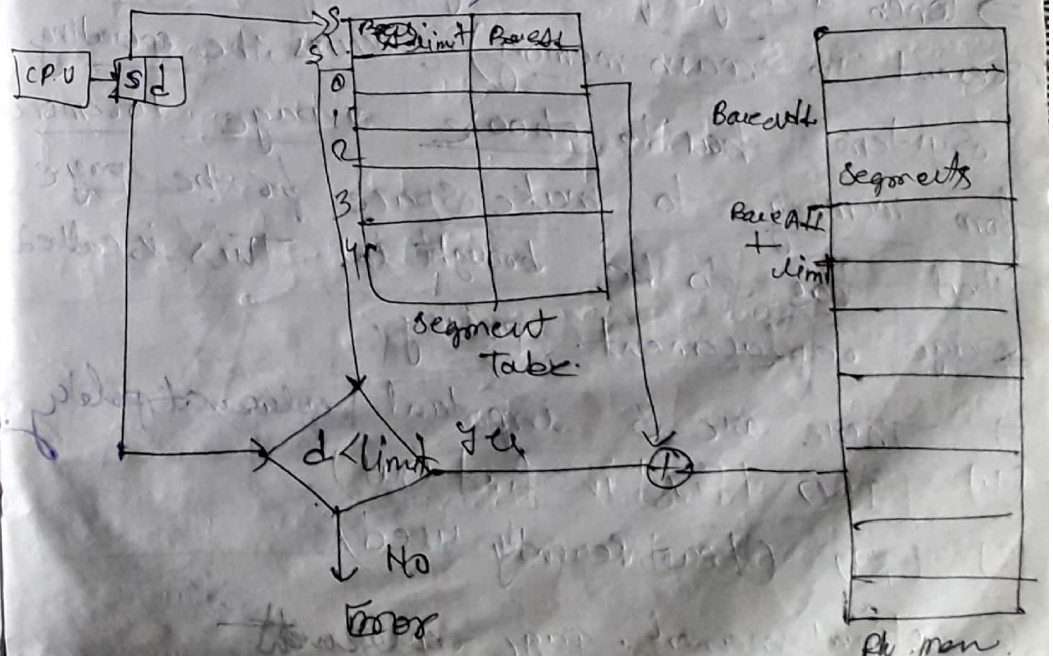
⇒ The limit specific the length of the Segement.

Logical View of Segmentation.





⇒ Translation of 2D-logical Add. into 1-D Phy Ad.



↳ Segment no ⇒ No of bits required to represent the segment.

↳ Segement offset ⇒ The no of bit required to represent the size of the Segement.

Advantage of Segementation

① No internal fragmentation

② ~~No internal~~

③ Segement table Consume less space in Comparision to page Table

~~Replacement~~

Page Replacement Policy

⇒ When a page fault (page being accessed not present in main memory) occurs, the operating system has to choose a page to remove from memory to make space for the page that has to be brought in. This is called page replacement policy

⇒ There are 3 important replacement policy

(1) FIFO (First in first Out)

(2) LRU (Least Recently used)

③ optimal merof page replacement

FIFO (Left)

⇒ FIFO select for replacement the page least recently loaded in memory. This is simple policy and easy to implement the page which come first replaced out first.

Ex: a memory Consist of 3 frame and during execution of a Program the following Pages are referenced in the sequence

2, 3, 2, 1, 5, 2, 4, 5, 3, 2, 5

Frame	2	3	2	1	5	2	4	5	3	2	5
1	2	2	2	2	5	5	5	5	3	3	3
2		3	3	3	3	2	2	2	2	2	5
3				1	1	1	4	4	4	4	4
			*					*		*	

* (page hit)

Page Hit = 3

No of Page = 11

Fault page = 11 - 3 ⇒ 8

Ex: Frame Size = 3

6, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 6, 0, 1

Pages	6	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	6	0	1
1	6	6	6	2	2	2	2	4	4	4	0	0	0	0	0	0	0	6	6	6
2		0	0	0	0	3	3	3	2	2	2	2	2	1	1	1	1	1	0	0
3			1	1	1	1	0	0	0	3	3	3	3	3	2	2	2	2	2	1
					*							*	*			*	*			

Page hit = 5

Total No of Page = 20

= 20 - 5 = 15