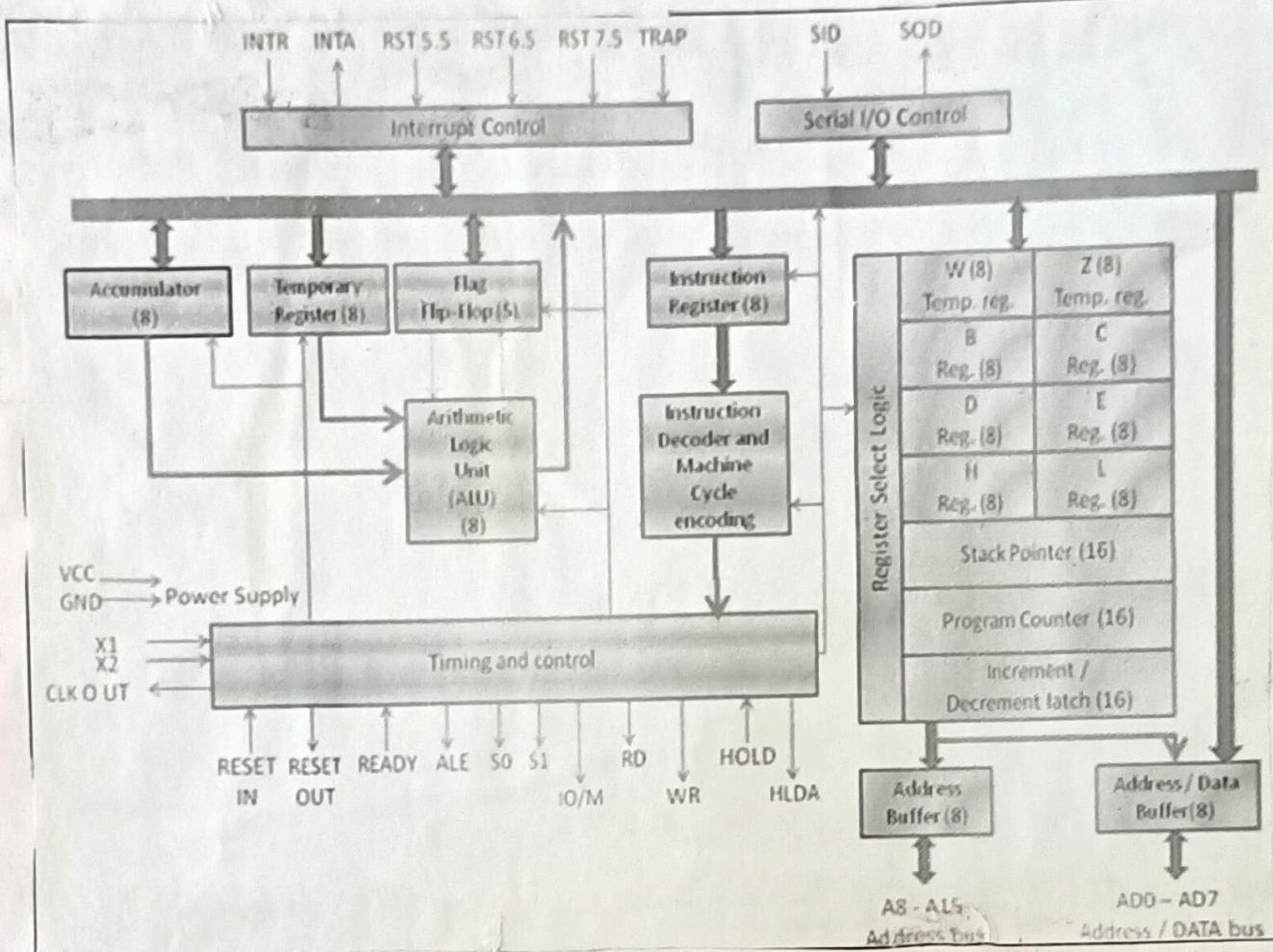


* Internal Architecture of 8085 ⑧

The diagram shows as functional block diagram of 8085



The internal architecture of 8085 contains ALU, Control unit (CU) & set of registers inside it.

* ALU : Arithmetic Logic Unit

It is responsible for performing arithmetic operations like addition, subtraction, multiplication & division.

It is also responsible for performing logical operations like AND, OR, X-OR, Complement, 8-state and shift.

These operations are performed by arithmetic circuits like half adder, full adder, half subtractor, full subtractor. These circuits are implemented by logic gates & gates are implemented by using transistors.

8085 has 8 bit ALU which can operate upon 8 bit & 16 bit numbers.

* Control Unit (CU): It acts as a brain of the processor. CU is responsible for generating necessary control signals at appropriate time.

The examples of control signals generated by control unit are - memory read signal (MEMR) memory write signal (MEMW) IO Read (IOR), IO Write (IOW) or Interrupt Acknowledge (INTA).

* Register File: 8085 has or it contains a set of internal registers. These registers are used for temp. storage of data. It means these registers are acting as internal memory of 8085. It is always better to have more no. of internal registers so that we get more internal memory so that we have to depend less on external memory. Also it is better to ~~have~~ have registers of larger size.

The various registers inside 8085 can be classified according to their size, use & availability to user.

Classification of Register Inside 8085

A] Classification According to size

- 8 bit - examples are A, B, C, D, E, H, L
- 16 bit - examples are Program Counter (PC)
Stack Pointer (SP)

B] Classification According to their Use

i) General Purpose Registers (GPR)

If the register is used only for temporary storage of data, then it is known as GPR.

examples are - B, C, D, E, H & L

ii) Special Function Registers (SFR)

If the register is used for only a specific function, then it is known as SFR.

ex. - Program Counter (PC), Stack Pointer (SP)

c) Classification of Registers based on their availability to User

i) Programmer Visible Registers: If the register is available to user while writing the programs then it is known as programmer visible register.

ex - A, B, C, D, E, H, L

ii) Programmer Invisible Registers: If the register is not available to user while writing the programs, but it is used internally by the processor (8085), then it is known as Programmer Invisible Registers

ex - H, Z.

* Detailed Explanation (Register Organization) of Registers

* Accumulator (A): 8085 contains a 16 bit register known as accumulator. It is denoted as 'A'. While performing any arithmetic operation (e.g. addition of two 8 bit nos), out of two operands (nos), one operand or number is always stored in A. After performing the desired operation, the result generated is stored by into Accumulator. It is a by default process.

* General Purpose Registers

8085 provides six general purpose register having size of 8 bit each.

These registers are named as B, C, D, E, H & L.

They are used for temporary storage of 8 bit data.

These register can paired together to store 16 bit numbers. But 8085 allows only limited pairing

Three possible register pairs are BC DE & HL

In these pairs we can store a 16 bit no. as size of pair is 16 bit.

W & Z Registers:

These two registers i.e. W & Z are of 8 bit each.

They are known as Programmer Invisible Registers.

These two registers are not available to the user while writing the programs, but they are used internally by 8085 itself.

Program Counter (PC) :

It is a ~~not~~ 16 bit Special Function Register (SFR).

It always contains 16 bit memory address.

It contains a specific address. i.e it contains 16 bit memory address of the next memory location which is to be executed.

Thus, PC is used to keep track of execution.

Stack Pointer (SP) :

It is a 16 bit Special function Register (SFR)

It always contains 16 bit memory address of stack memory.

Stack pointer (SP) always points towards ~~stack~~ top of the stack memory.

Flag Register:

8085 has a 8 bit flag register.

It contains 5 flags. (flip flops)

Each flag requires 1 bit from flag register. Hence total 5 bits from flag Register (8 bits) are used & the remaining three bits are ^{not} used or unused.

Flag Register gives us the status of the result generated by ALU.

The result generated by ALU, after performing the desired operation is stored in Accumulator (A) as well as it is given to flag register so that the respective flags are updated (i.e. they are set or resetted).

8085 has 5 flags & these are

- i) sign (S)
- ii) Zero (Z)
- iii) Carry (CY)
- iv) ~~Auxillary~~ Auxillary Carry (AC)
- v) Parity (P)

The format of flag register is as shown -

S	Z	-	AC	-	P	-	CY
---	---	---	----	---	---	---	----

Sign (S): If the result of arithmetic operation is negative (-), then this flag is set to One, otherwise it is reseted or zero.

Zero (Z): If the result of arithmetic operation is zero then this flag is set to One, otherwise it is reseted or becomes '0'

Carry (CY): While performing addition of two 8 bit nos. if carry is generated from MSB (Most Significant Bit) CY flag is set to One, otherwise reseted (0)

** There is no separate flag for borrow (subtraction)

The same flag is used subtraction operation also.

While performing subtraction of two 8 bit nos. if borrow is generated from MSB, then also CY flag is set to one, otherwise reseted (0)

Auxillary Carry (AC):

This flag is used in case of BCD operations.

BCD - Binary Coded Decimal.

While performing addition of two BCD nos. if carry is generated from D₃ bit to D₄ bit (i.e lower nibble to upper nibble), this flag is set (1), otherwise it is reseted (0)

Parity (P)

If the 8 bit result generated in accumulator (A) contains even no. of one's then this flag is set to one, otherwise it is reseted or becomes zero (0)

Temporary Register

It is an 8 bit register. It is used to hold (store) the second operand (no) temporarily.

While performing any arithmetic or logical operation inside ALU, one no. is provided to ALU by Accumulator & the second no. is provided by Temporary Register.

Instruction Register (IR)

It is an 8 bit register

It is used to hold Opcode (short code) of the instruction which is to be executed.

Apart from all these registers there are some other blocks present in architecture diagram.

Instruction Decoder & Machine Cycle Encoding Unit

Opcode of the instruction which is to be executed is first stored into Instruction Register (IR) & then it is given to this block.

This block will decode the instruction so that 8085 will come to ^{know} exact meaning of the instruction (i.e. what to do with this instruction) & then it will calculate the amount of time required for execution of this instruction. The time required for execution of inst. depends upon the clock frequency.

Serial I/O Control

This block will control the serial communication facility in 8085. SOD pin will be used to send out the data from 8085 in bit by bit manner & SID is used to receive the data in bit by bit manner.

Interrupt Control:

This block will control the five H/W interrupts of 8085 i.e TRAP, RST 7.5, RST 6.5, RST 5.5 & INTR. Also if INTR interrupt is received, it will generate acknowledge signal INTA.

All the blocks are connected together by using the internal bus as shown in the architecture diagram.

Concept of memory, memory locations, memory address

In any microprocessor including 8085 the size of main memory (RAM) is decided by number of address lines present in that processor. The formula used is

$$2^N = \text{Size of Main memory (RAM)}$$

where $N = \text{No. of address line.}$

In 8085, there are 16 address lines, hence size of main memory (RAM) will be $2^{16} = 64 \text{ Kbytes.}$

$$\begin{aligned} \text{In computers } 1 \text{ Kbyte} &= 2^{10} \text{ bytes} \\ &= 1024 \text{ bytes.} \end{aligned}$$

$$\begin{aligned} \text{Hence in 8085, } 64 \text{ Kbytes} &= 64 \times 1 \text{ Kbytes} \\ &= 64 \times 1024 \text{ bytes} \\ &= 65,536 \text{ bytes.} \end{aligned}$$

Thus in 8085 RAM i.e main memory size is 64 Kilo bytes or 65,536 bytes.

The main memory (RAM) of 8085 has 65,536 memory locations.

Each location is capable of storing 8 bit data or 1 byte i.e storage capacity of each location is 8 bit or 1 byte

In total there are 65,536 locations. Hence each memory location is identified by unique address & known as memory address.

The memory address of each memory location is 16 bit as there are 16 address lines in 8085.

* With above discussion we have to remember two things

- 1] Each memory location can store only 8 bit (1 byte) data
- 2] Each location is identified by 16 bit unique address known as memory address.

65,535	FFFF H
65,534	FFFE H
3	0003 H
2	0002 H
1	0001 H
0	0000 H

- * In above diagram 64K memory is divided into 65,536 locations.
- * The first mem. location is numbered as '0' hence last will be 65,535
- * Each location is capable of holding 8 bit data or 1 byte
- * Each location will have 16 bit memory address.

The address of 0th Location = 0000 0000 0000 0000

All 16 address lines will have value zero.

The address of 1st location = 0000 0000 0000 0001

The address of 2nd location = 0000 0000 0000 0010

3rd location = 0000 0000 0000 0011

:

:

:

The address of last or 65536th = 1111 1111 1111 1111
location

But it is quite difficult to remember these 16 bit binary addresses. Hence this 16 bit address is divided into 4 groups each of bits. When 4 binary bits are combined together

it gives its hexadecimal Equivalent.

For ex. add. of 0th location = 0000 0000 0000 0000

When these 16 bits are grouped into groups of 4 bit each we get

$$\begin{array}{cccc} \underline{0000} & \underline{0000} & \underline{0000} & \underline{0000} \\ 0 & 0 & 0 & 0 \end{array}$$

Thus address of 0th location will be 0000H in hex

Address of 1st location will be 0001H in hex

$$\begin{array}{cccc} \underline{0000} & \underline{0000} & \underline{0000} & \underline{0001} \\ 0 & 0 & 0 & 1 \\ : & : & : & \end{array}$$

Similarly,

The add of last location (65,536) will be ~~00~~ FFFFH in hex

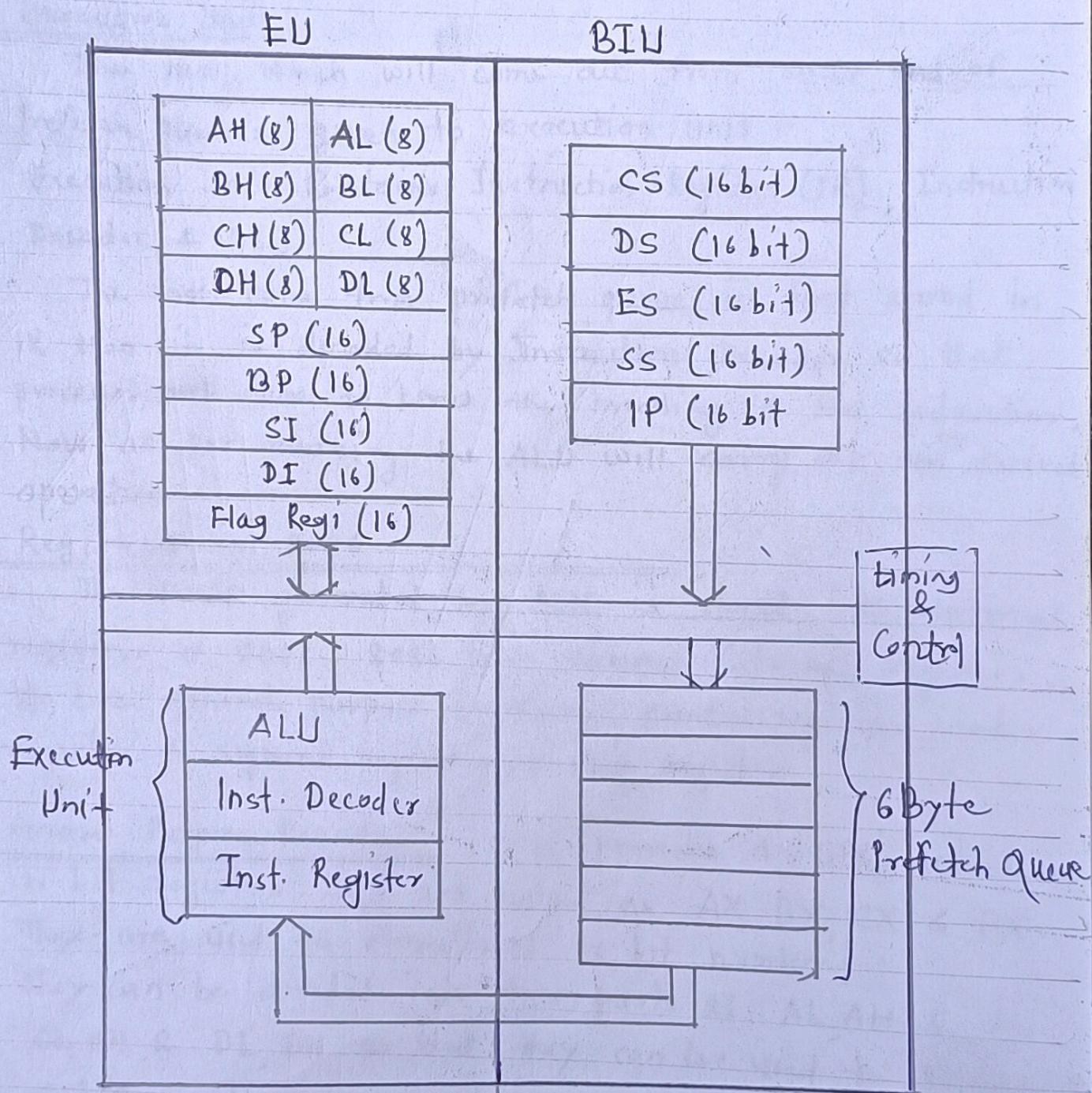
$$\begin{array}{cccc} \underline{1111} & \underline{1111} & \underline{1111} & \underline{1111} \\ F & F & F & F \end{array}$$

- * For simplicity memory addresses are always expressed in Hex form.
- * In this way the starting address of 8085 memory is 0000H & the ending or last address is FFFFH.

Architecture of 8086 Processor.

(10)

8086 is successor of 8085. 8086 is the first processor in which first time concept of prefetching (inst. pipeline) is used. Also first time main memory is divided into number of smaller parts called segments. The internal architecture of 8086 is as shown in the diagram. The diagram is divided into two parts 1) Execution Unit (EU)
2) Bus Interface Unit (BIU)



BIU: It is responsible for all the communication with outside world (memory or I/o devices). (1)

The BIU prefetches the instructions from main memory by using system bus.

Prefetch Queue: The instructions prefetched by BIU will enter into prefetch queue. In 8086 the length of prefetch queue is 6 bytes (i.e it can store 6 bytes / 6 instructions) The queue follows first in first out (FIFO) policy.

Execution Unit :

The inst. which will come out from other end of Prefetch queue is given to execution unit.

Execution Unit Contains Instruction Register (IR), Instruction Decoder & ALU.

The inst. came from prefetch queue is first stored in IR then it is decoded by Instruction Decoder so that processor ~~will~~ come to know the meaning of the instruction. Now as per meaning the ALU will carry out the desired operation.

Registers in 8086

The result generated by ALU is stored into internal registers of 8086. 8086 have various internal registers, like general purpose registers, pointer registers, index register & segment registers, & flag register

General Purpose Registers: 8086 provides 4 GPR's which are 16 bit registers. They are named as AX, BX, CX & DX. They are used to store/hold 16 bit numbers. They can be divided into two parts as AL, AH, BL, BH, CL, CH & DL, DH so that they can be used to store 8 bit numbers.

Pointer Registers

8086 provides two pointer registers 1) SP 2) BP

- 1) Stack Pointer SP (16 bit) : It is a 16 bit SFR which always points towards top of the stack memory
- 2) Base Pointer BP (16 bit) : It is a 16 bit SFR which always points towards normal memory.

Index Registers

8086 provides two 16 bit index register SI & DI

- 1) Source Index (SI) - 16 bit
- 2) Destination Index (DI) - 16 bit

These two index registers are used in case of string or array related operations for autoincrement/decrement

Flag Register

8086 provides 16 bit flag register

8086 has 9 flags. Each flag requires one bit from FR

The 9 flags are

- 1) Sign (S)
- 2) ~~Carry~~ Zero (Z)
- 3) Carry (CY)
- 4) Auxiliary Carry (AC)
- 5) Parity (P)
- 6) Overflow (OV)
- 7) Direction Flag (DF)
- 8) Interrupt Flag (IF)
- 9) Trap flag (TF)

The first 5 flags are same as 8085 & they are known as status flags & the remaining 4 flags are known as control flags.

Segment Registers:

8086 provides 4 segment registers each of 16 bit

They are named as CS, DS, ES & SS

They are used to give starting address of the segments

Instruction Pointer

(13)

8086 provides a 16 bit Instruction Pointer which gives offset value (distance to be travelled within the segment).

Segmentation

* Dividing the larger size of memory into number of smaller modules is called as segmentation.

* 8086 has 20 Address Lines

$$\therefore \text{Size of Main Memory} = 2^{20} = 1 \text{M}$$

This 1 MB memory is divided into 16 equal parts or segments,

~~Each~~ Each segment is of size 64 KB

Out of 16 segments only 4 segments are active at a time. The active segments are named as —

- 1] Data Segment
- 2] Code Segment
- 3] Extra Segment
- 4] Stack Segment

Control Unit Design:

(14)

Generally a digital system is separated into two parts 1) Data processing Unit
2) Control Unit.

- * The data processing unit is collection of functional units capable of performing certain operations on data.
- * While the control unit issues control signals to the data processing part to perform operations on data.
- * These control signals selects the functions to be performed at specific times & route the data through appropriate functional units.

We can say that the data processing unit is logically reconfigured by the control unit & it is intimately involved in the sequencing & timing of the data processing unit.

* Control Unit Design Methodology:

There are two approaches in control unit design

- 1] Hardwired Control Unit Design
- 2] Micro-programmed Control Unit Design

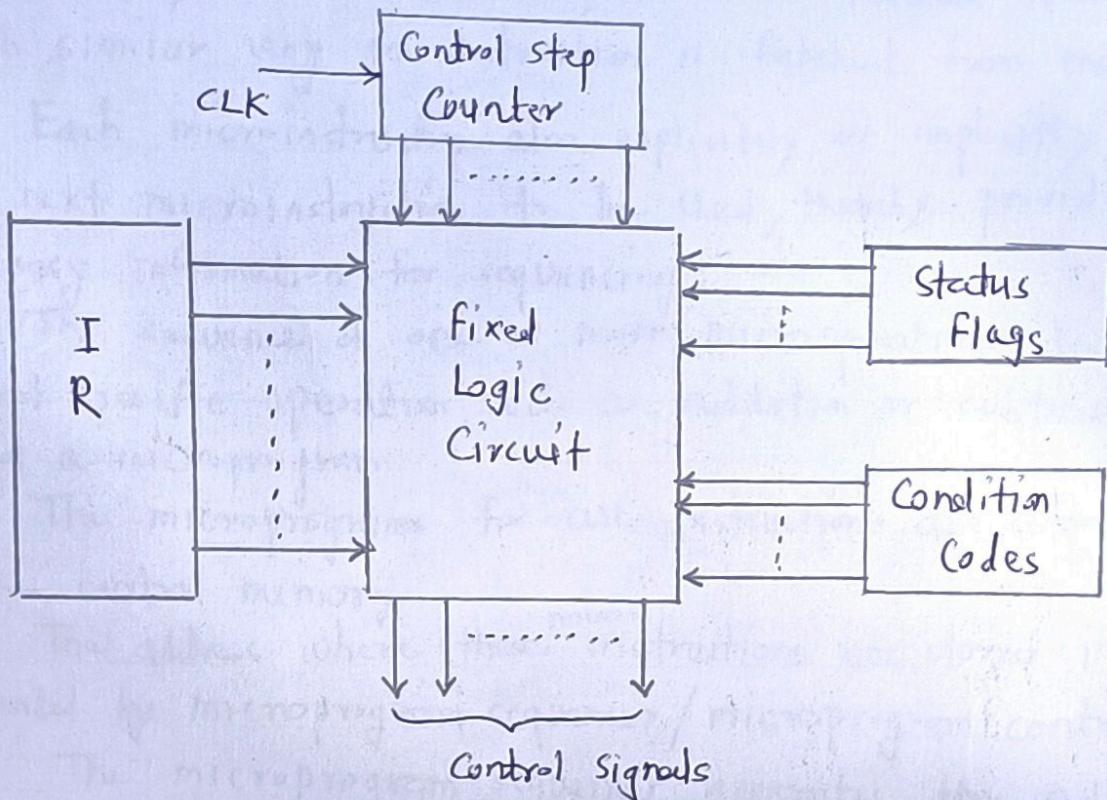
* Hardwired Control Unit Design : In the hardwired implementation the control units use fixed logic circuits to interpret instructions and generate control signals from them.

The hardwired control unit design involves various trade-offs between the amount of hardwired used, the speed of operation and the cost of the design process itself. In hardwired ~~the~~ control unit design there are different methods

- 1] State Table method
- 2] Delay Element method
- 3] Sequence counter method

4] Programmable Logic Array (PLA) method.

The fig. shows block diagram of a typical hardwired control unit



* Microprogrammed Control Unit

In this approach the control unit uses microprograms to interpret and execute instructions.

- * Every instruction in a CPU is implemented by a sequence of one or more sets of concurrent micro-operations.
- * Each micro-operation is associated with a specific set of control lines which, when activated, causes the micro-operation to take place.
- * Since the number of instructions & control lines is often in hundreds the complexity of h/w control unit is very high & hence it is costly & difficult to design.
- * furthermore h/w control unit is relatively inflexible because it is difficult to change the design, if one wishes to correct design error or modify the inst. set.

Microprogramming is a method of control unit design in which the control signal selection and sequencing information is stored in a ROM or RAM called as control memory.

The control signals to be activated at any time are specified by microinstruction, which is fetched from CM in much similar way an instruction is fetched from main memory

Each microinstruction also explicitly or implicitly specifies the next microinstruction to be used, thereby providing the necessary information for sequencing.

The sequence of one or more microoperations designed to control specific operation, such as addition or subtraction is called a microprogram.

The microprograms for all instructions are stored in the control memory.

The address where these micro-instructions are stored in CM is generated by microprogram sequencer / microprogram controller

The microprogram sequencer generates the address for microinstruction according to the instruction stored in the IR

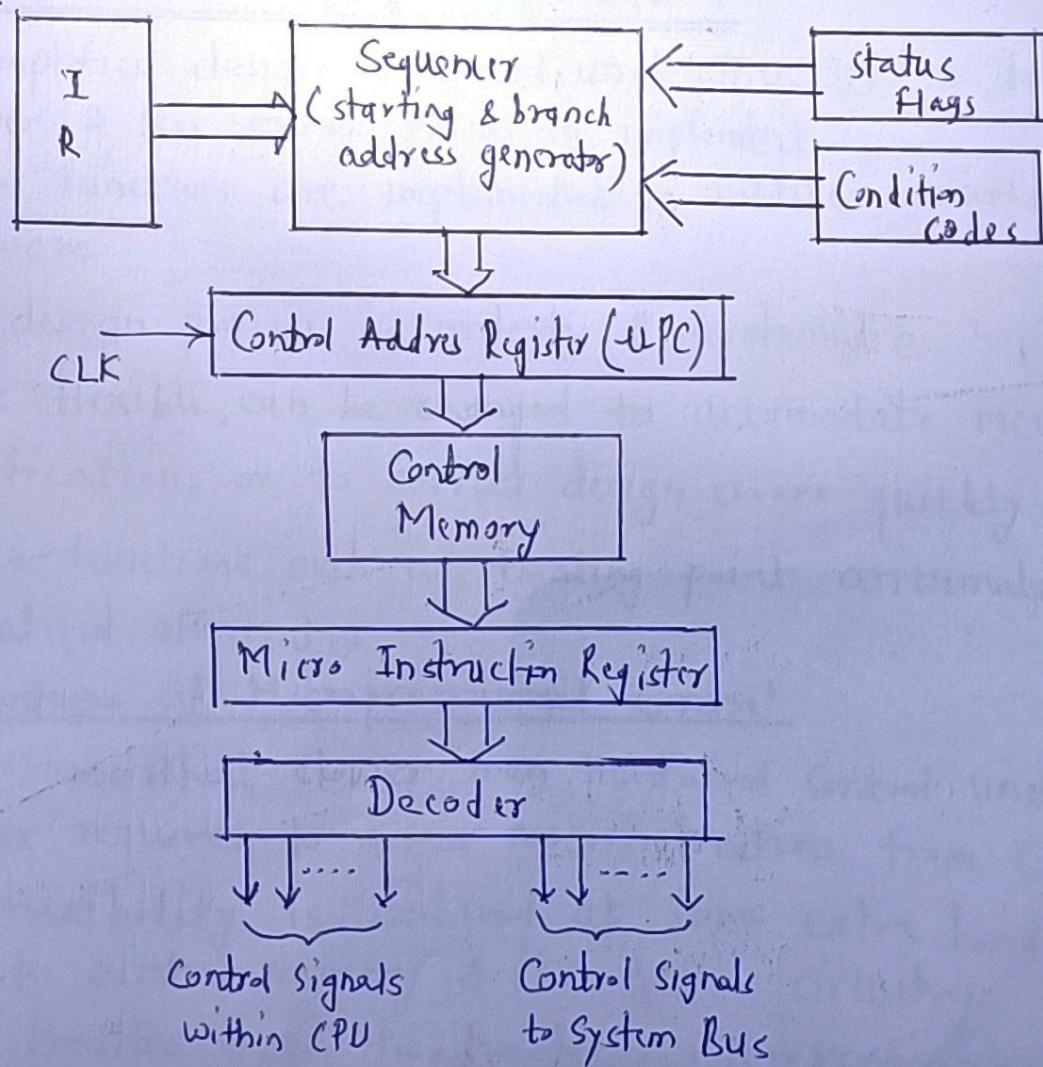


Fig shows the organisation of microprogrammed CU. (17)
consist of control memory, control address register, microinstruction register and microprogram sequencer.

This CU works as follows

- 1] The control address register holds the address of the next microinstruction to read.
- 2] When address is available in control address register, the sequencer issues READ command to the control memory.
- 3] After issue of READ command, the word from addressed location is read into microinstruction register.
- 4] Now the content of microinstruction register generates control signals & next address information for the sequencer.
- 5] The sequencer loads a new address into the control address register based on the next address information.

Advantages of Microprogrammed Control

- 1] It simplifies design of control unit. Thus it is both cheaper & less error prone to implement.
- 2] Control functions are implemented in software rather than hardware.
- 3] The design process is orderly & systematic.
- 4] More flexible, can be changed to accommodate new system specifications or to correct design errors quickly & cheaply.
- 5] Complex functions such as floating point arithmetic can be realised efficiently.

Disadvantages of Microprogrammed Control

- 1] It is somewhat slower than hardwired control unit, because time is required to access microinstructions from CM.
- 2] The flexibility is achieved at some extra hardware cost due to control memory & its access circuitry.

Besides these disadvantages, microprogramming is the dominant technique for implementing control units.

18

Difference between Hardwired CU & Microprogrammed CU

Hardwired Control	Microprogrammed Control
1] It is faster	1] It is slower
2] It is not flexible, to accommodate new ^{sys} specifications or new instructions	2] It is more flexible, to accommodate new system specifications or new instructions
3] It is difficult to handle large/complex instruction sets	3] It is easier to handle large or complex instruction sets
4] Design process is somewhat complicated	4] Design process is orderly & systematic
5] Very difficult to support operating system and diagnostic features.	5] Very easy to support operating system & diagnostic features.
6] It is used mostly in RISC microprocessors	6] It is used in mainframes & some microprocessors
7] ROM is not used	7] ROM of size 2K-10K by 20-400 bit microinst. is used
8] It uses least chip area	8] It uses more chip area.

I/O Module & It's functions

- 19

A computer communicates with outside world by means of input output (I/O) system.

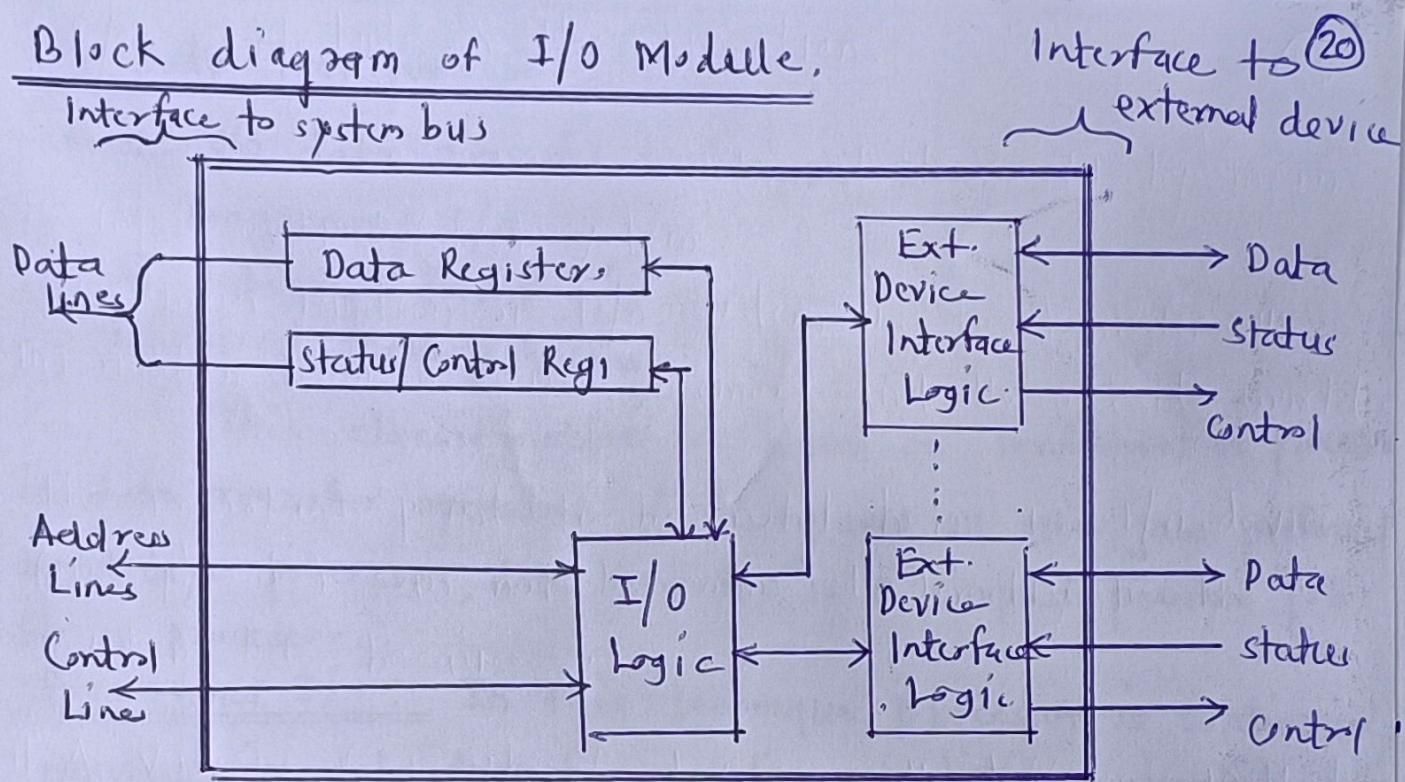
The main function of I/O system is to transfer the information between CPU or memory and the outside world.

The important point to be noted here is I/O devices (peripherals) can not be connected directly to the system bus because —

- 1] A variety of peripheral devices with different methods of operation are available. So it would be impractical to incorporate the necessary logic within the CPU to control a range of devices.
- 2] The data transfer rate of peripherals is often much more slower than that of the CPU or memory. So it is impractical to use a high speed system bus to communicate directly with the peripherals.
- 3] Generally, the peripherals used in a computer system have different ~~methods~~ data formats and word lengths than that of CPU used in it.
- 4] Generally CPU & Memory they are in IC form i.e (semiconductor technology) & most of the peripherals are electromechanical device & hence they differ in speed of operation.

So to overcome all these difficulties, it is necessary to use an I/O Module in between CPU and I/O device.

Block diagram of I/O Module.



Functions of an I/O Module

- ① Control & Timing: The I/O module coordinates the flow of traffic between internal resources and external devices by providing necessary timing & control requirements.
- ② Processor Communication: I/O module communicates with up processor. It accepts commands from processor. It exchanges data with processor. It also performs the status reporting job.
- ③ Device Communication: I/O Module also communicates with devices. This communication involves commands, status information & data.
- ④ Data Buffering: It also performs data buffering job. Data taken out memory is buffered into I/O module or data which is to be stored in memory is buffered into I/O module.
- ⑤ Error Detection: I/O module is also responsible for detection of errors & subsequently reporting errors to the processors. (e.g. paper jam, bad disk track)

I/O Data Transfer

(21)

Various I/O data transfer techniques are

- I] Programmed I/O
- II] Interrupt driven I/O
- III] Direct Memory Access

This classification is based on involvement of processor in data transfer operation. As compared all other jobs performed by today's processor, data transfer is a simplest possible job for a processor.

Programmed I/O: In this technique processor is continuously involved in data transfer operation which is a simplest job for any processor. Processor will not perform any other job while doing data transfer. It means valuable time of processor is utilised for doing simplest possible job. Data transfer is done by using IN & OUT instructions

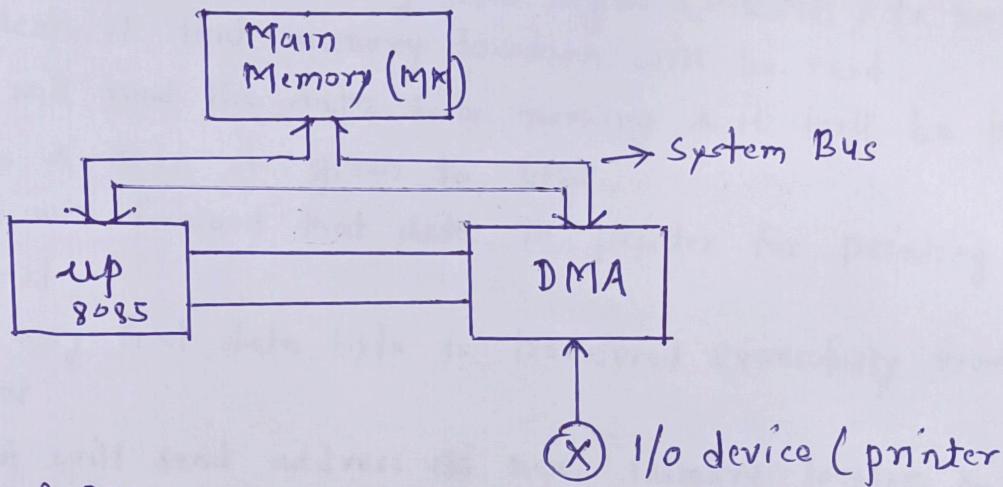
Interrupt driven I/O: In this technique processor is not continuously involved but it is partially involved in data transfer operation. Whenever any I/O device connected to processor wants to perform data transfer, it will interrupt the processor. The processor will stop ^{the} current activity & it give go to do the data transfer & when the data transfer is over it will rejoin the ~~at~~ original activity which he was doing initially. Thus, in this method processor is not continuously involved in data transfer but it goes or performs data transfer only when it is interrupted & once it is over, it goes back to rejoin the original activity which it was performing

Thus processor is partially involved in data transfer & hence this method is better than programmed I/O method.

* DMA : Direct Memory Access

(22)

In this method processor is not at all involved in data transfer operation. It is required only at the beginning (for getting control of the system bus) & at the end of data transfer operation (to transfer back control of the system bus)



Operation of DMA

- * I/O Device (printer) is not directly connected to 8085 but it is connected to DMA
- * DMA, 8085 (8085) & Main memory (MM) are connected to each other by using system bus.
- * When such a system is made ON 8085 (8085) is bus Master & DMA is slave. ~~8085~~ 8085 (8085) is master means system bus controlled or owned by 8085 (8085)

Steps involved in data transfer from Memory to I/O device (printer)

- 1] Printer will make request to DMA for data transfer
- 2] As DMA is slave, it will forward that request to 8085 (8085) by activating HOLD signal.

HOLD: It is the request made by DMA controller to 8085 (8085) for getting control of the system bus

- 3] Upon received HOLD signal 8085 (8085) will generate HLDA (hold acknowledge) signal & it is given to DMA
- 4] When this HLDA is received by DMA, it becomes master & 8085 (8085) becomes slave [Now the control of bus is with DMA]

(23)

DMA will inform the requesting device (printer) about acceptance of its request

- i] Now DMA will address of the first memory which is to be read to memory by using system bus
- ii] Now DMA will send Memory Read Signal (MEMR) to memory so that contents of that memory location will be read.
- iii] Now, he will read the data from memory & it will be placed on data bus & then it given to DMA.
- iv] Now DMA will forward that data to printer for printing by activating TOW signal.
- v] In this way first data byte is transferred successfully from memory to printer.
- vi] Now DMA will send address of next memory location & it will generate MEMR signal to read data
- vii] The data will be brought into DMA & then given to printer for printing by activating TOW signal.
- viii] This process is repeated till all the data bytes from memory gets transferred.
- ix] When all the bytes are over, the DMA will deactivate HOLD signal so that 8085 becomes master & DMA becomes slave. When DMA is doing data transfer by using system bus, at that 8085 will not keep quite or remain idle but it will perform all other jobs which are not requiring the system bus.
- * Thus in DMA operation, DMA is performing data transfer at the same time processing is also doing some other jobs simultaneously. Thus concurrent operations are possible & hence DMA is the best method of doing the data transfer.
- DMA is most preferred method for data transfer of large data files.

(24)

Interrupt structure of 8085

Interrupt: It is ~~an~~ disturbance to the normal execution of program in any processor/computer.

Classification of Interrupts in 8085

- 1] Hardware Interrupts and software interrupts
- 2] Maskable Interrupts and Non-maskable interrupts
- 3] Single Level Interrupt system & Multilevel interrupts
- 4] Vectored, Non-vectored & Auto Vectored Interrupts

* 8085 provide 5 Hardware & 8 Software Interrupts

Hardware Interrupt: The interrupt generated ~~from~~ by an external device connected to processor from outside is called as hardware interrupt.

8085 has 5 Hardware interrupts & these are

1. TRAP
2. RST 7.5
3. RST 6.5
4. RST 5.5
5. INTR

These interrupts will be received on Pin No. 6, 7, 8, 9 & 10 of 8085 respectively. The details of these interrupts are -

Type of Interrupt	Priority	Maskable / Non Maskable	Type of Triggering	Vector Address
TRAP	1 st (highest)	NM	+ Ve Edge	0024 H
RST 7.5	2 nd	M	+ Ve Edge & Level	003CH
RST 6.5	3 rd	M	Level	0034H
RST 5.5	4 th	M	Level	002CH
INTR	Last or least	M	Level	-

NM - Non-maskable

M - Maskable.

Software Interrupts

Interrupts which are generated by execution of any interrupt related instruction are called as software interrupts. They are generated internally.

There are 8 software interrupts in 8085 & these are .

Interrupt	Vector Address
RST 0	0000H
RST 1	0008H
RST 2	0010H
RST 3	0018H
RST 4	0020H
RST 5	0028H
RST 6	0030H
RST 7	0038H

Difference Between Hardware & Software Interrupts

Hardware Interrupts	Software Interrupts
1] It is an asynchronous event which may or may not occur with respect to time	1] It is a synchronous event which always occurs with respect to time
2] It is generated by external devices (peripherals) connected to microprocessor	2] It is generated internally by execution of any interrupt related instruction.
3] They have lower priority	3] They have highest priority
4] They are used for interfacing peripheral Devices	4] They are used for program debugging.
5] CPU may or may not generates interrupt acknowledge signal	5] CPU never generates interrupt acknowledge signal.
6] Ex. TRAP, RST 7.5, RST 6.5 RST 5.5 & INTR	6] Ex. RST 0, RST 1, RST 2, RST 3 RST 4, RST 5, RST 6, RST 7

Maskable Interrupts: Interrupts which can be avoided or made pending are known as maskable interrupts.

* Generally hardware interrupts are maskable.

Non-Maskable Interrupts: Interrupts which can not be avoided or made pending are known as non-maskable interrupts.

All software interrupts are non-maskable.

Difference Between Maskable & Non-maskable Interrupts

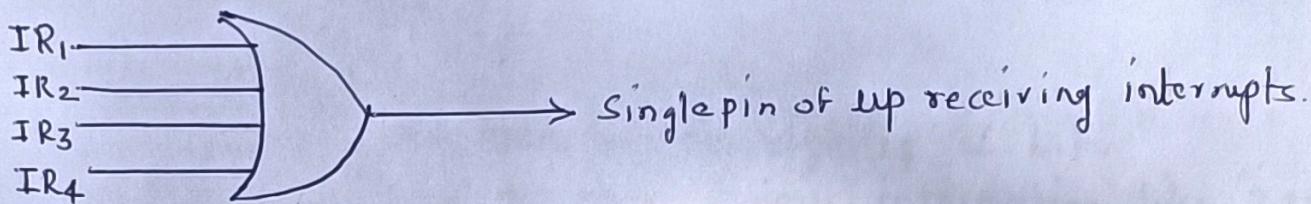
Maskable Interrupts	Non-Maskable Interrupts
1] Interrupts which can be avoided or made pending are known as maskable interrupts	1] Interrupts which can not be avoided or made pending are known as non-maskable interrupts
2] They have lower priority	2] They have highest priority
3] They are used for interfacing peripheral devices	3] They are used for emergency situations like power failure, smoke detection
4] Generally they are level triggered h/w interrupt are maskable	4] Software interrupt are always they are always edge triggered interrupts maskable
5] CPU may or may not generates interrupt acknowledge signal	5] CPU never generates interrupt acknowledge signal.
6] Ex. RST 7.5, RST 6.5, RST 5.5 INTR	6] Ex. TRAP, RST 0, RST 1, RST 2, RST 3, RST 4, RST 5, RST 6, RST 7

* Single Level & Multi Level Interrupt System

Single Level Interrupt System

If only one pin is provided in the processor to receive the interrupts, then it is known as single level interrupt system.

If multiple interrupts are coming to such processor then in that case these interrupts are logically ORed together & the output of OR gate is connected to that single pin on up.



Multilevel Interrupt System

If more than one or multiple pins are provided for on processor to receive the interrupts then it is called as multilevel interrupt system.

e.g. In 8085, five separate pins (Pin No 6-10) are provided to receive 5 interrupts (TRAP, RST 7.5, RST 6.5, RST 5.5 & INTR). Hence it is a multilevel interrupt system.

* Steps Involved in providing service to a interrupt :-

When processor is executing its main program & in between any interrupt comes then -

- ① up will finish execution of current instruction (i.e. instruction in hand)
- ② Before going to provide service to interrupt, up will store add. of the next instruction from main program (called as Return Address) on stack memory by performing PUSH operation.
- ③ Now the program control is transferred from main program to Interrupt Service Routine (ISR).
- ④ up will start executing instructions from ISR & the last inst of ISR will be RET
- ⑤ When RET is executed, the program control is transferred back to main program & the return address is obtained back from stack memory by performing POP operation.
- ⑥ This return address is loaded into PC so that up will start executing remaining instruction from main program.

... --> data & hence they

* Memory :

It is a storage medium or device.

It is used to store information.

The information can be data, programs, intermediate results or final results.

* Unit of Memory :

The smallest possible unit of memory is 1 bit.

The memory is used to store binary information like '0' or '1'.

Group of four binary bits is called as Nibble

e.g. 1100

Group of eight binary bits is called as 'byte'

e.g. 11001100

The storage capacity of memory is always expressed in byte, Kilo byte, Mega bytes, Giga bytes etc.

$$1 \text{ KBytes} = 2^{10} = 1024 \text{ bytes}$$

$$1 \text{ MBytes} = 2^{20} = 1,05,536 \text{ bytes}$$

$$1 \text{ GByte} = 2^{30} \text{ Bytes.}$$

* Classification of Memory :

- ① Internal Memory & External Memory
- ② Primary (main) memory & Secondary Memory
- ③ Based on technology used — Magnetic Memory
Optical Memory
Semiconductor memory
- ④ Based on what type facility is available in memory

RAM

ROM

EPROM

EEROM

Internal Memory : If the memory is present within the CPU (processor) it is called as internal memory

e.g. Every processor is having a set of registers (register file) inside it. These registers are used for temporary storage of data & hence th

is called as internal memory of computer

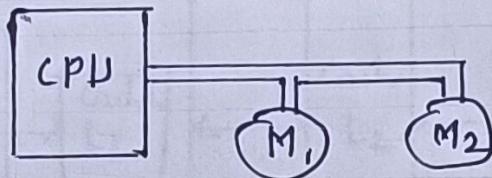
Apart from this now a day all the processor's (CPU) have inbuilt or internal cache memory.

As it is inbuilt or internally available hence it is called as internal memory or internal Cache.

The internal cache is also called as Level 1 or L₁ Cache.

External Memory:

If the memory device is connected externally to CPU (processor), then is called as external ~~per~~ memory



As shown M₁, M₂ are external memories. They are connected to CPU are using external busses.

Primary or Main Memory:

Primary memory is also known as Main Memory of computer (RAM)

It is used to store programs which are active in use. Basically it is used to store user programs. i.e. programs written by we people.

The have higher speed but their storage capacity is less as compared to secondary memory.

Secondary Memory

They are connected externally to CPU but they have huge storage capacity & they are slower in speed as compared to main memory (RAM)

The storage capacity of secondary memory is always in GB's or TB's

for eg. Hard disk.

This memory is used to store operating system or system programs & large data file which are not frequently used.

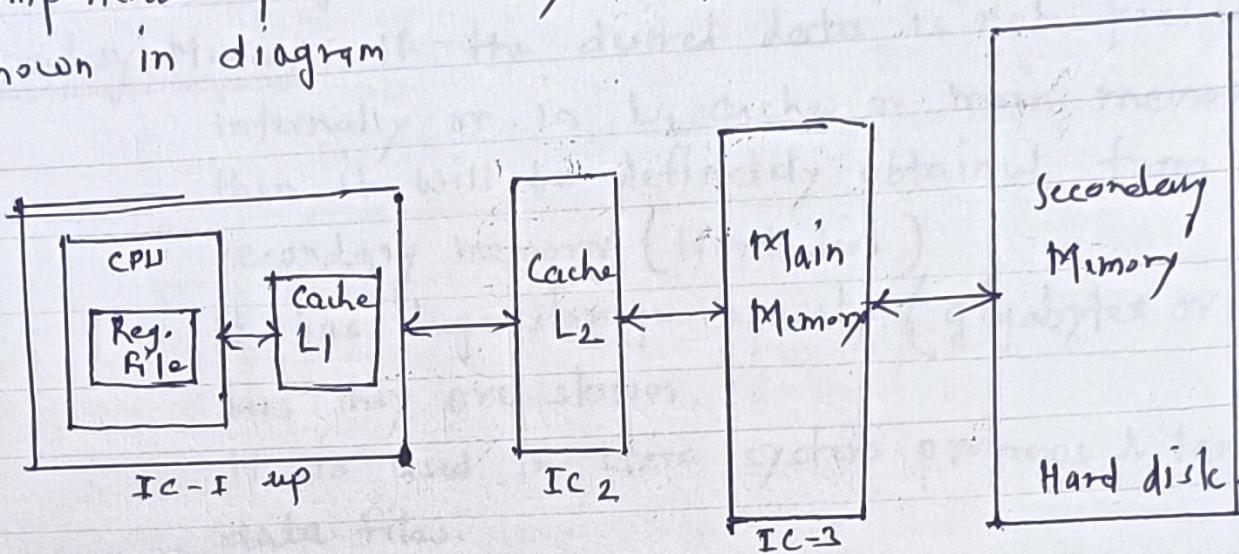
Type of Technology

Magnetic - Flappy, Hard disk, magnetic tape

Optical - CD

Semiconductor - RAM, ROM (in IC form)

Memory Hierarchy in Computer : The various memory components present in any computer system are as shown in diagram



» Internal Registers: They are present within CPU. They are used to store the data temporarily.
It is always better to have ~~more~~ no. of registers with larger in size.

They are the closest memory elements to CPU.

» Internal Cache (L₁): It is present within or inside CPU
If the desired data is not present in L₁, cache processor will search it in internal cache (L₁)

The search from internal registers or L₁ cache is much more faster as it is internal communication.

» External Cache (L₂): It is a high speed SRAM placed between CPU & main memory. If desired data is not available in internal registers or L₁ cache then CPU will try to get it from external cache or L₂ cache.

Main memory: If the desired data is not available internally or in external cache (L_2) the CPU will get it from external main memory.

Main memory (RAM) have generally larger storage capacity & it is used to store the programs which are active in use.

Secondary Memory: If the desired data is not present internally or in L_2 cache or main memory then, it will be definitely obtained from secondary memory (Hard disk).

It has huge storage capacity (gigabytes or Terabytes) but they are slower.

It is used to store system programs & large data files.

Memory Characteristics / Properties to be considered while Selecting a memory device.

① Cost of the memory:

Cost of the memory is not only the cost of storage medium but it includes the cost of accessing mechanism (Read/write heads)

For e.g. in case of floppy, the cost of memory is cost of floppy disk + cost of floppy drive.

OR

~~In~~ In case of CD, the cost of memory is cost of CD + cost of CD drive.

The cost of memory is always expressed in dollars/bit

$$C = \frac{C}{S} \text{ dollars/bit}$$

C = cost of the complete memory system

S = storage capacity (total no. of bits)

c = cost of the memory,

It is always preferable to have lower cost of the memory

* Performance of Memory: It is measured in terms of speed. The speed of memory is measured in terms of "access time". Read Access Time: The time required to read specific amount of information from memory is called Read Access Time.

Less is the access time faster is the memory

* Write Access Time: The time required to write specific amount of information in memory is called Write Access time.

Less is the access time faster is the memory
It is not necessary that read access time & write access time must be same. It depends on the type of storage medium (magnetic/optical) as well as type of accessing mechanism.

* Accessing Modes: It is the order or sequence in which information is accessed or stored in memory.

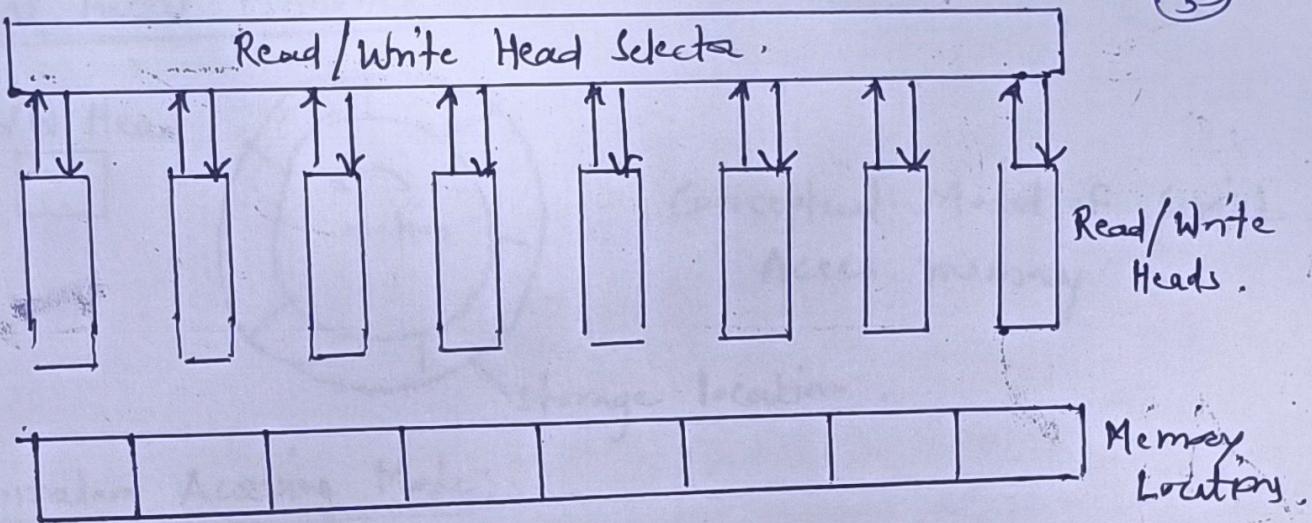
Random Accessing Mode:

If the information or locations can be accessed in any order and access time is independent of the location being accessed, then it is called as Random Access mode.

Semiconductor memories (memories in the form of IC)
i.e. RAM, ROM, they are random Access memory.

Each storage location of RAM can be accessed independently, for this purpose a separate access mechanism i.e. Read-Write head is required for each location.

Since every location has its own access mechanism, the memory will be costlier but it will be faster.



Conceptual Model of RAM.

Serial or Sequential Access Memory

Memories whose locations can be accessed only in certain predetermined sequence are called as serial or sequential access memories.

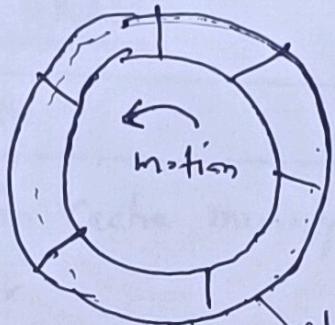
Magnetic disk, magnetic tape & optical memories are examples of serial access memories.

In serial access memories the access mechanism i.e. (Read Write Heads) is shared by storage locations & must be assigned to different locations at different time by moving the stored information or the read write Heads or both.

In most of serial memories the storage locations are moving or rotating around a closed path or track. A particular location can be accessed only when it passes the fixed R/W head. Hence the time required to access a particular location depends on its position relative to the R/W head.

Serial access memories are much more slower in speed & also their cost is less as they are using a single R/W head.

R/W Head
□



Conceptual Model of Serial Access memory

storage location.

* Semirandom Accessing Mode:

Memory device like magnetic hard disk & CD-ROM contain many rotating tracks.

If each track has its own R/W head then tracks can be accessed randomly, but access within the track is serial. In such case the accessing mode is semirandom.

* Data Transfer Rate or Bandwidth of memory

The maximum amount of information that can be transferred to or from the memory per unit time is called data transfer rate or bandwidth.

It is measured in bits per second or words per second.

Kbyte, Mbytes or Gbytes these are the units used.

* Reliability:

It indicates how safe is data stored in memory.

Reliability is measured in terms of mean time before failure (MTBF)

Memories with no moving parts have higher reliability than memories having moving parts.

Even in memories where there are no moving parts, the reliability problem arises, particularly when very high storage density or data transfer rates are used.

Error detecting & error correcting codes can increase the reliability of the memory.

Memory Retention:

(25)

① SRAM & DRAM

SRAM	
1) It is used in Cache memory	1) It is used in main memory
2) It is faster	2) It is slower
3) It is costlier	3) It is cheaper
4) Storage capacity is less	4) Storage capacity is larger
5) Physical size is larger.	5) Physical size is smaller.
6) Power consumption is more	6) Power consumption is less.
7) flip-flops are used to store information	7) Capacitors are used to store information.
8) Refreshing circuitry is not required	8) Refreshing circuitry is required.

② Volatile & Non volatile memories:

Magnetic Tapes

Historically, **magnetic tape** was the first kind of secondary memory. A computer tape drive is analogous to a home tape recorder: a 2400-ft-long tape is wound from the feed reel past a recording head to the take-up reel. By varying the current in the recording head, the computer can write information on the tape in the form of little magnetized spots.

Figure 2-15 shows how information is organized on a magnetic tape. On a computer with 8-bit bytes, each frame contains 1 byte, plus an extra, redundant bit, called a parity bit, to improve reliability. Typical recording density is 1600 frames (bytes) per inch (denoted as 1600 bpi), which means that a frame takes up less than 1/1000 of an inch. Other common densities are 800 bpi and 6250 bpi. After a tape drive has finished writing a **physical record** (a sequence of frames), it leaves a gap on the tape while slowing down. If the program writes short physical records on the tape, most of the space will be wasted in the gaps. By writing physical records that are much longer than the gap, the tape utilization can be kept high.

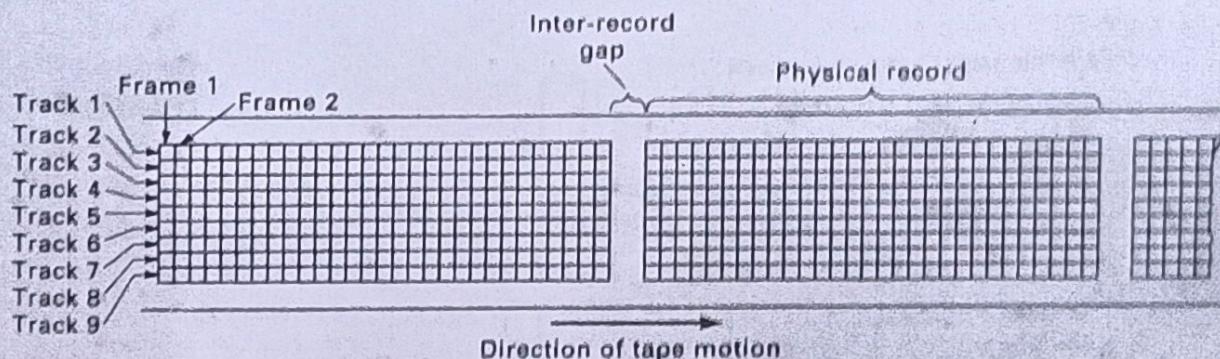


Fig. 2-15. Information on a magnetic tape is recorded as a sequence of rectangular bit matrices.

Magnetic tapes are sequential-access devices. If the tape is positioned at the beginning, to read physical record n , it is first necessary to read physical records 1 through $n - 1$, one at a time. If the information desired is near the end of the tape, the program will have to read almost the entire tape, which may take several minutes. Forcing a CPU that can execute millions of instructions per second to wait 200 sec while a tape is advanced is wasteful. Tapes are most appropriate when the data must be accessed sequentially.

Magnetic Disks

A **disk** is a piece of metal, ranging from about 5 to 10 inches in diameter, to which a magnetizable coating has been applied at the factory, generally on both sides (see Fig. 2-16). Information is recorded on a number of concentric circles, called **tracks**. Disks typically have between 40 and a few hundred tracks per

surface. Each disk drive has a movable head that can be moved closer to, or farther from, the center. The head is wide enough to read or write information from exactly one track. A disk drive often has several disks stacked vertically about an inch apart. In such a configuration the arm will have one head next to each surface, all of which move in and out together. The radial position of the heads (distance from the spindle) is called the **cylinder**. A disk drive with n platters will have $2n$ heads, hence $2n$ tracks per cylinder.

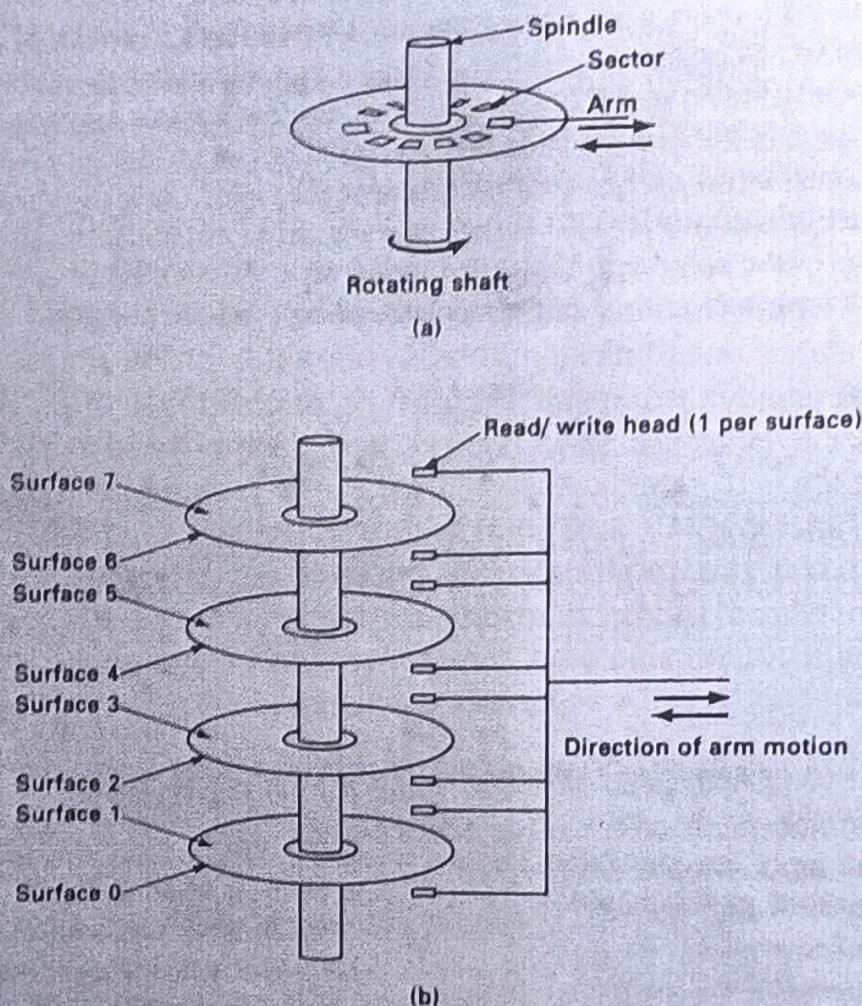


Fig. 2-16. (a) A disk with one platter. (b) A disk with four platters.

Tracks are divided into **sectors**, normally between 10 and 100 sectors per track. A sector consists of a certain number of bytes, usually 512.

To specify a transfer, the program must provide the following information: the cylinder and head, which together specify a unique track, the sector number where the information starts, the number of words to be transmitted, the main memory address where the information comes from or goes to, and whether information is to be read from the disk into memory or written from memory to the disk.