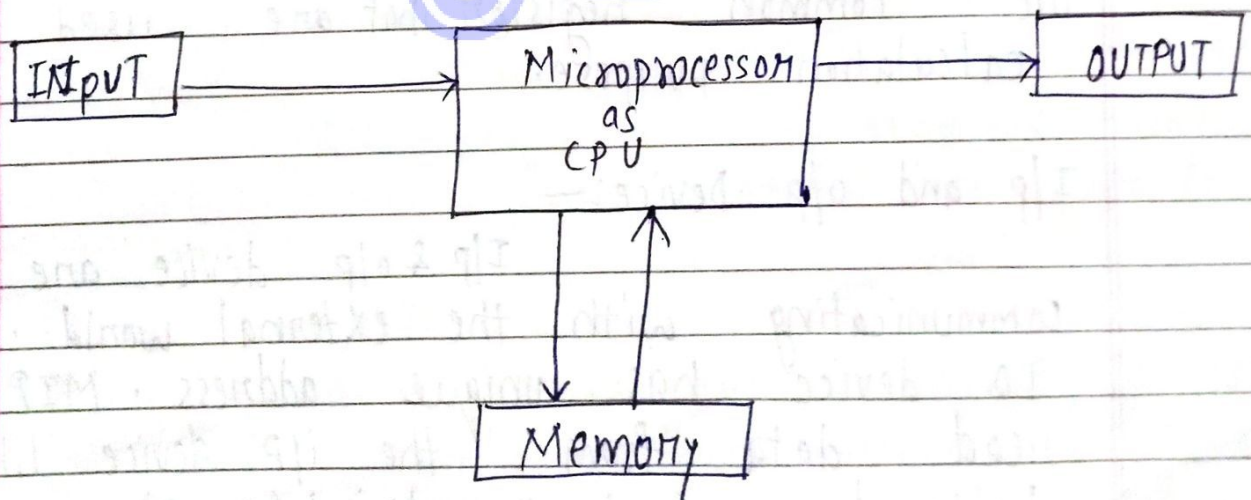


MICROPROCESSOR: —

A microprocessor is a device which can process the data in few micro seconds and is used to control the device. It is a semiconductor device of logic circuit manufactured by using VLSI technique.

Microprocessor is a multipurpose clock driven, register based electronic device that reads binary instruction from a storage device except binary data as input and process data according to the instruction and provide result as output.

As like a brain, MIP gets input from eye, ear and another sense organ and process information to the output such as other external part of our body.



The MIP can be divided into 3 segment:-

- (i) CPU:- It is used for fetch binary coded instruction from memory, decode the instruction & generate the control signal required to execute the instruction.
 - (ii) The ALU performs the arithmetic & logical operation such as addition, subtraction, OR, NOT, AND, NAND, NOR, X-OR & X-NOR etc in binary format.
 - (iii) CU:- Control unit provides the timing & control signal to all the operation in the microprocessor and microcontroller. It controls the flow of data b/w MIP memory & peripheral.
- Register Array:- It is used for the temporary storage of data during the execution of the program. A, B, C, D, E, H, L are the common registers that are used for calculation purpose.

I/p and o/p Device:-

I/p & o/p device are used for communicating with the external world. Each IO device has unique address. MIP can read data from the i/p device like keyboard, Analog to digital (A to D) converter & card-reader.

The input port receive the data from the i/p device. The output device are used to display or print the program, data or result. eg - printer, floater & D to A converter. IO device are connected with the port address and different type of port selected by combination of flip-flop.

HOW DO MICROPROCESSOR WORK?

We assume that the program & data are already entered in the read-write memory. The program include binary instruction to add given data and to display the answer at the 7 segment LED. When MIP is given a command to execute the program it reads and execute one instruction at a time and finally send result to the display. The sequence of the process is read interrupted and perform. The MIP fetches the instruction from the memory sheet and decode it and execute the instruction. The sequence of each instruction is fetch, decode & execute.

Micro controller :-

It is a 8 bit micro controller designed by INTEL. It includes several peripheral devices like timer, counter and memory element. The internal memory of 8051 is 4kb. It includes the

boolean processor feature. It is available in NMOS and PMOS designing architecture.

Microprocessor

Microcontroller

- | | |
|---|--|
| <p>(1) If ALU & CU is fabricated on a single chip is called as microprocessor.</p> <p>(2) It is general purpose device used in computer.</p> <p>(3) There is a internal PROM.</p> <p>(4) It has internal timer.</p> | <p>(1) If microprocessor + peripheral devices are fabricated on a single chip it is k/a microcontroller.</p> <p>(2) It is used for an embedded system.</p> <p>(3) There is no PROM.</p> <p>(4) It does not has internal timer.</p> |
|---|--|

Architecture of Microprocessor / 8085 \Rightarrow

8 bit , 40 pin IC fabricated on a single chip. It is an . It used data buses for read write and process 8 bit data on a single time.

Feature:-

- \rightarrow Address bus - 16 bit
- \rightarrow Data bus - 8 bit
- \rightarrow Memory Capacity - 64kb
- \rightarrow Input pins - 20
- \rightarrow Output pins - 20
- \rightarrow Clock frequency - 3 to 5 MHz
- \rightarrow Instructions - 74

The architecture of 8085 has the following blocks-

- (i) Register block
- (ii) ALU
- (iii) CU
- (iv) Interrupt control
- (v) Serial I/O block

① Register Block \Rightarrow 8085 has a group of predefined register which is used to store data & result.

(i) General purpose register:- It has 6 different types of 8 bit register where the data or the result can be store. Register are B, C, D, E, H, L these register are programmable by user to store/copy any data to perform different operation.

(ii) Temporary register:- These are used by MIP for internal operation to store the operand or any address of memory (W & Z)

(iii) Special purpose register:-

(a) Program Counter :- It is a 16 bit register is point to the memory address of the next instruction to be fetched and executed. It is automatically incremented by 1 to point the next ~~memory~~ ^{memory} location.

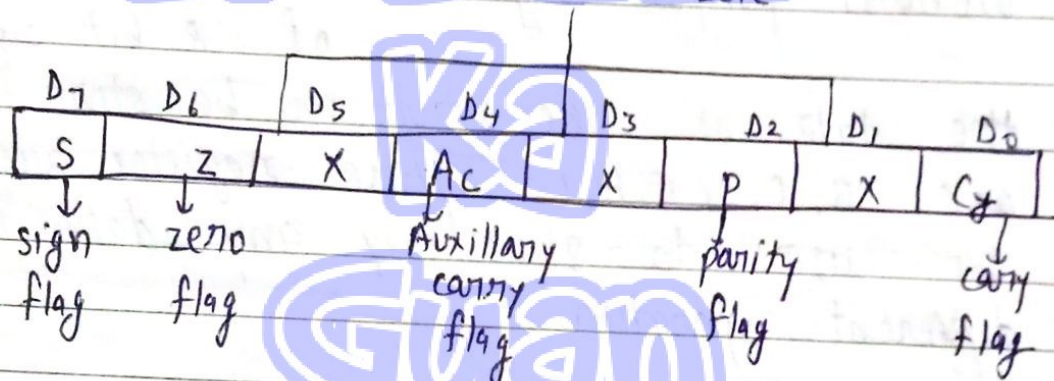
(b) **Stack pointer:-** It is a 16 bit register used as a memory pointer to point the memory location of the stack. It defines the starting address of the stack.

(c) **Increment / Decrement latch:-** It is 16 bit register which used to increment or decrement the content of stack pointer and program counter register.

(iv) ~~(d)~~

Flag register:-

Don't care



Carry Flag:- If any carry is generated from the operation the carry flag is set otherwise reset.

Parity Flag:- If the result has even no. of 1's then parity flag is set otherwise reset.

Auxillary Flag:- When a carry is generated by digital a_3 & carry forward to D_4 . The Ac is set otherwise reset.

Zero flag:- If the ALU operation result is zero then the flag is set otherwise reset.

Sign Flag:- After any arithmetic operation, if D₇ bit store in accumulator is 1 then sign flag is set otherwise reset.

Q. MVI A, 48H
MVI B, 6EH
ADD B
STA → 2060H
HLT

Ans- MVI A, 48 → 01001000

MVI B, 6E → 01101110

ADD B → A = A + B

S = 1

Z = 0

Ac = 1

P = 0

Cy = 0

For X=0 ⇒ 10010000 = 90H

For X=1 ⇒ 10111010 = BAH

$$\begin{array}{r} 01001000 \\ 01101110 \\ \hline 10110110 \end{array}$$

Ans

Q. MVI A, 48H

MVI B, 6EH

~~XOR~~ B

ST A → 2060H

HLT

Ans- A, 48 → 01001000, B, 6E → 01101110

A = A ⊕ B

⊕ 01101110

S = 0

Z = 0

Ac = 0 For X=0 ⇒ 00000000 = 00H

P = 0 For X=1 ⇒ 001001010 = 2AH

Cy = 0

Ans

$$\begin{array}{r} 01001000 \\ \oplus 01101110 \\ \hline 00100110 \end{array}$$

- ② Arithmetic & logic unit Block \Rightarrow
Accumulator:- It is an 8 Bit register loaded data from the internal data bus and can transfer data to the internal data bus. One of the operand and result of the ALU operation stored in the accumulator. It can be rotated and shifted as per our requirement.

Arithmetic & logic Unit \Rightarrow It is responsible for perform bitwise arithmetic and logical operation. The result of ALU operation placed in the accumulator after the execution of instruction.

- ③ Control Unit Block \Rightarrow It is important to control and synchronize all data transfer in microprocessor.

(a) Instruction Register (IR) \Rightarrow It is an 8 Bit register. The instruction operation code stored in memory and transfer to the instruction decoder.

(b) Instruction Decoder (ID) \Rightarrow Fetch opcode is sent to the instruction decoder which decodes the meaning of the opcode and given timing & control signal to register.

(c) Timing & Control unit \Rightarrow The timing & control unit use a synchronize clock to general the

signals. These signals are used to regulate the transformation of data.

④ Interrupt Block \Rightarrow The interrupt is the process which is initiated by the external device.

Microprocessor has 5 interrupts:—

- (i) TRAP (ii) RST 7.5 (iii) RST 6.5
(iv) RST 5.5 (v) INTR

Interrupt can be categorised in maskable and non-maskable interrupt.

TRAP is non maskable interrupt which can't be ignored. If Microprocessor receive TRAP interrupt by the external device then it has to respond for it. Remaining 4 are maskable interrupt and can be ignored.

⑤ Serial I/O control Block \Rightarrow

The 8085 is the only 8-bit microprocessor which has a separate pin for serial input and output operation. They are SID (Serial input data line) and SOD (Serial output data line). For the long distance communication 8085 used these data pins.

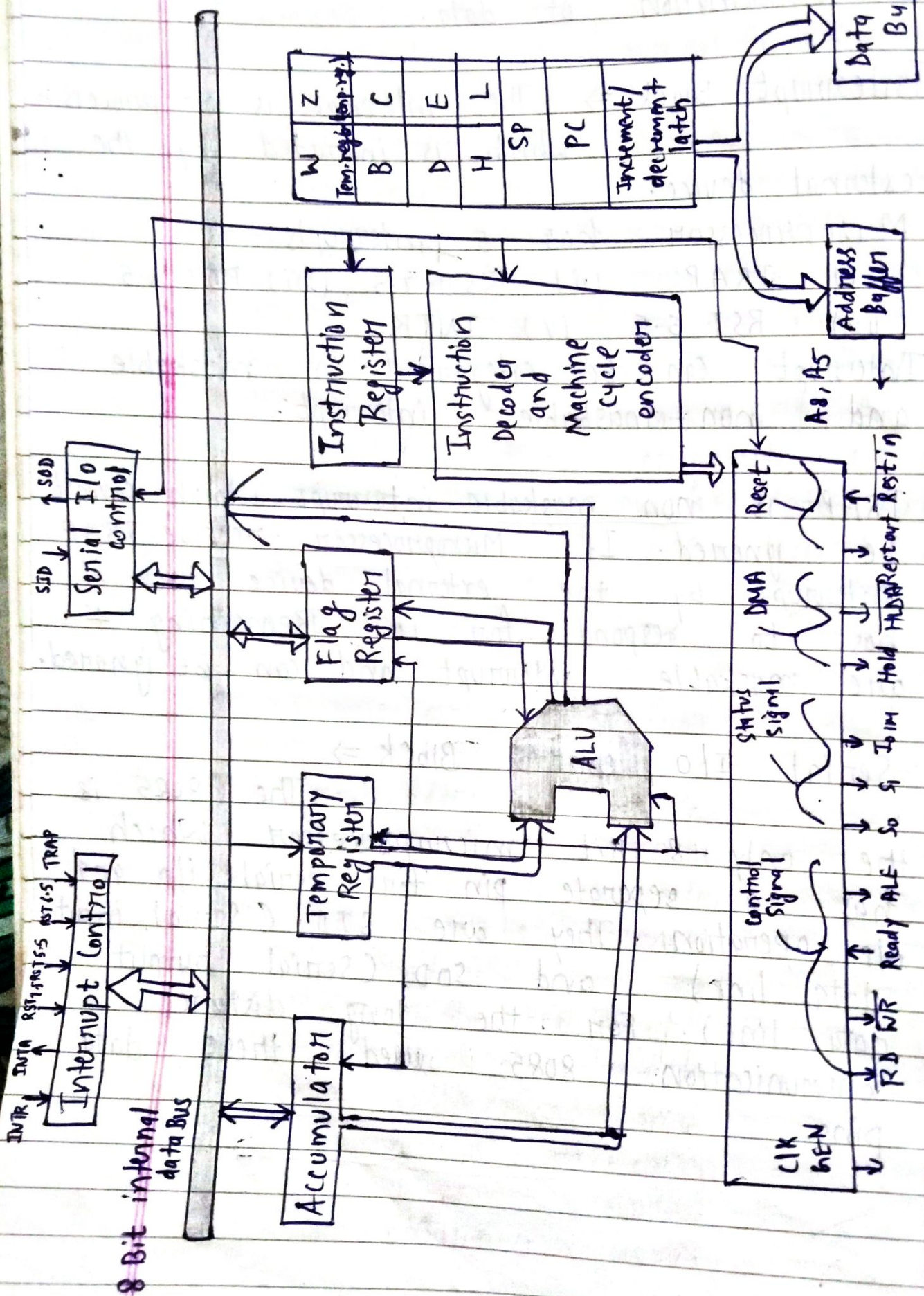
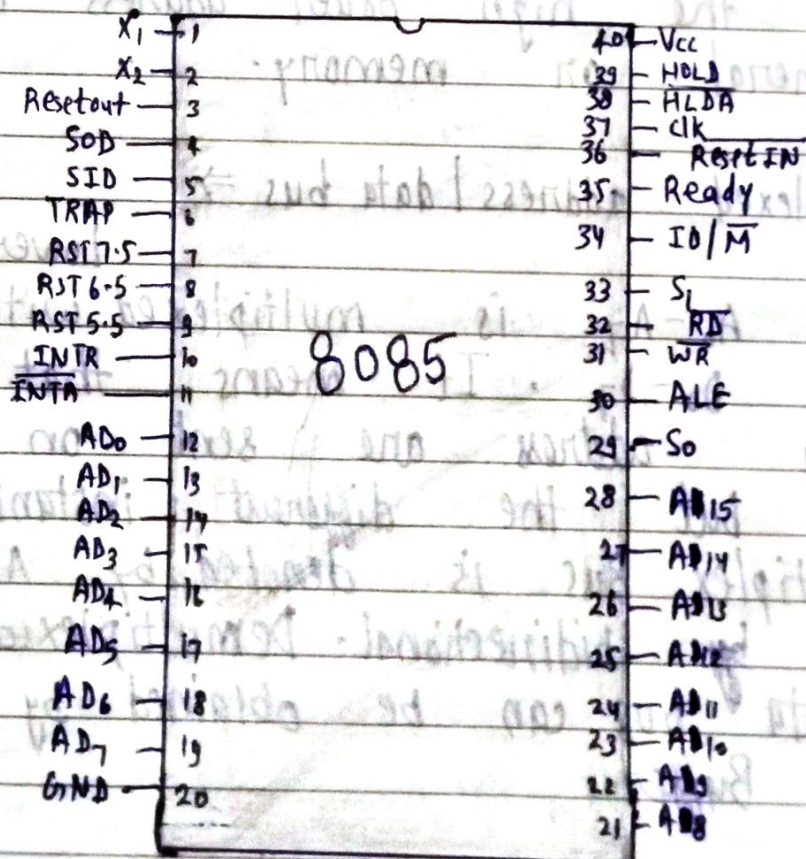
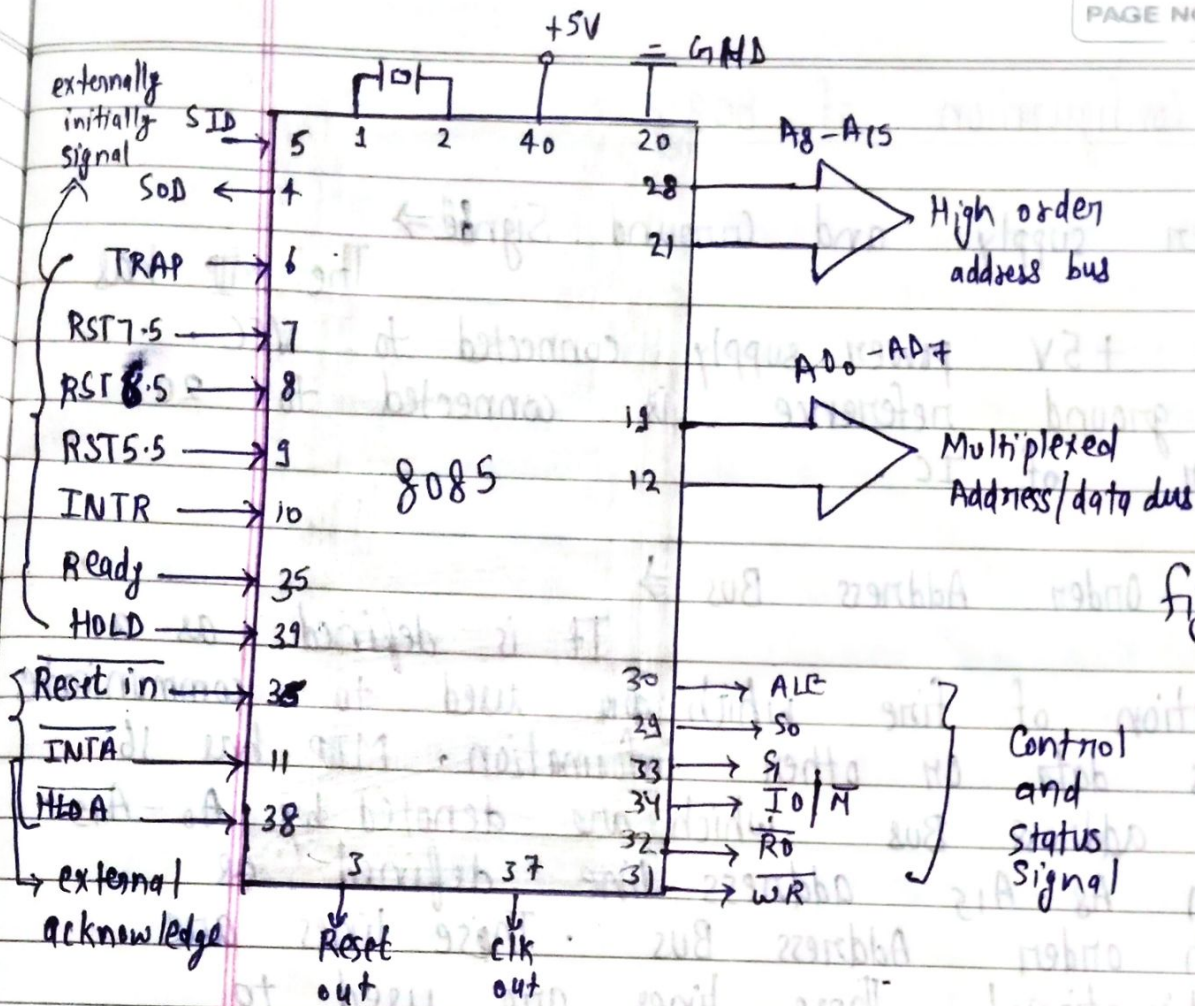


fig:- Architecture of 8085



Pin configuration of 8085 \Rightarrow

(i) Power supply and Ground Signal \Rightarrow

Single +5V power supply connected to VCC.
The ground reference is connected to 20.
number of IC.

(ii) High Order Address Bus \Rightarrow

It is defined as a collection of line which are used to communicate address data or other information. MIP has 16 bit address bus which are denoted by $A_0 - A_{15}$.
from $A_8 - A_{15}$ address line defined as high order address bus. These lines are unidirectional. These lines are used to send the high order address to the peripheral on memory.

(iii) Multiplexed address / data bus \Rightarrow

Lower order address bus $A_0 - A_7$ is multiplexed with 8 bit data bus $D_0 - D_7$. It means that the data and address are sent on the same line but at the different instant of time.
Multiplex bus is denoted by $AD_0 - AD_7$. These are bidirectional. Demultiplexed address & data bus can be obtained by using latch & buffer.

(iv) Control Signal \Rightarrow

(a) Read Bar (\overline{RD}) ~~control~~ : — It is an active low signal meaning that if

\overline{RD} value is zero the external device placed the data and microprocessor read the data.

(b) Write Bar (\overline{WR}) : — When \overline{WR} signal goes low

it read the data to ~~Read data~~ address bus. At a time of result execution the \overline{WR} signal value is zero.

(c) ALE (Address latch enable) : — It is the positive

generate by peripheral device for begin and execution. It indicates that the bit on multiplexed address data bus are address bus.

(v) Status Signal \Rightarrow

(a) $\overline{IO/M}$: — $\overline{IO/M}$ signal shows that the microprocessor

is communicating with I/O port or memory when it is high, it indicates memory operation and low, it indicates I/O operation.

The signal is combined with \overline{RD} and \overline{WR} and generate the control signal which is I/O read, I/O write, memory read & memory write.

(b) S_1 & S_0 :— These status signal are used to indicate the internal operation of a microprocessor.

$I/O/\overline{M}$	S_1	S_0	Operation
0	0	1	Memory Write
0	1	0	Memory Read
1	0	1	I/O write
1	1	0	I/O Read
0	1	1	opcode fetch
1	1	1	Interrupt Ack.
Z	0	0	Hold
Z	X	X	Reset

Z \equiv high impedance

X = Unspecified

(vi) Externally initiated ~~ted~~ Signal \Rightarrow

(a) Reset signal :—

(i) Reset IN :— It is an active low input signal when the signal on this pin is zero, the program counter is said to 0000H. Control unit transfers the content of program counter to the address latch and provide the address of the first instruction to be executed. The other register values are not cleared by resetting.

(ii) Reset out :- This signal is generated by MP in response of the reset in signal. When reset in logic 0, reset out is logic 1, it indicates that the MP is being reset.

(vii) Ready Signal: \Rightarrow It is an active high signal used by MP to send whether a peripheral is ready or not for data transfer. If it is high it means that the peripheral is ready for communication. If it is low then microprocessor will wait until it becomes high.

(viii) Interrupt Signal \Rightarrow 8085 has 5 interrupts. TRAP is non-maskable & highest priority interrupt. It is level or edge sensitive signal. It was used during critical situation such as power failure. RST 7.5, 6.5 & 5.5 are maskable interrupts. These are vector interrupts and transfer the program execution to the specific memory location. RST 5.5 has the lowest priority.

(ix) INTR is maskable and lowest priority among all the interrupts. It is non-vectorised interrupt.

(x) \overline{INTA} :- It is active low signal and made active by the processor. It is acknowledged that MP has recognised interrupt through INTR pin.

(xi) Direct Memory Access Signal :—

~~DMA~~ DMA is used to transfer data from memory to peripheral or from peripheral to memory without the intervention of the MP.

(a) HOLD :— Active high signal. One signal on this pin indicates that the other device is requesting for DMA operation. After receiving hold signal MP releases the address & data bus to be used by other device.

(b) HLDA :— It is the active low signal. The value of this signal is zero by the MP after the IO device has completed the DMA operation & makes the hold signal active.

(xii) Serial input output signal :—

(a) SID :— It is a data line for serial input. The data on this line is loaded into the 7th bit accumulator when RIM instruction is executed.

(b) SOD :— It is the data line for serial output. The 7 bit of accumulator is output on SOD line when SIM instruction is executed.

(xiii) Clock Signals \Rightarrow

(i) X_1 & X_2 :- These are terminal to be connected to an external crystal oscillator which is derived from internal circuitry of the MP.

(ii) Clock :- It is a clk output for user which can be used by other digital ICs.

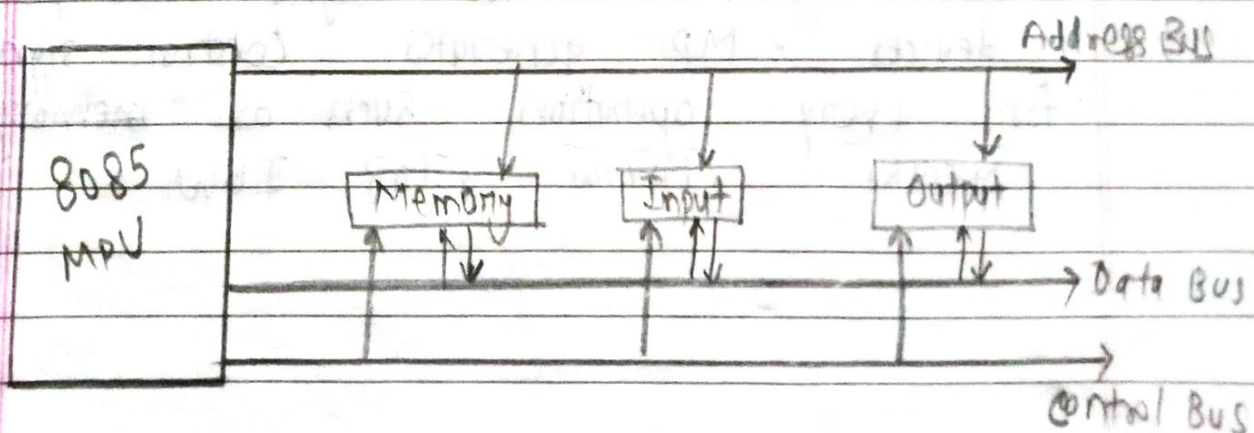
BUS Concept And Organisation \Rightarrow

To provide comm. b/w MP & other peripheral devices, Buses are required. The MP needs to perform these 3 steps:-

(i) Identify the peripheral (Memory location)

(ii) Transfer binary data

(iii) Provide Timing & Synchronised signal

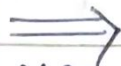


(i) Address Bus :— 8085 MPU address Bus consists of 16 signal line. Unidirectional signal lines communicate with microprocessor to peripheral device. The no. of address lines are used to determine the no. of memory location on input/output port. If CPU has n address lines then the total memory location are $2^n = 2^{16} = 65,536 = 16 \text{ KB}$. Address range from 0000 H to $FFFF \text{ H}$ these lines are represented by $A_0 - A_{15}$.

(ii) Data Bus :— 8085 has 8 bit address line these are bidirectional lines which range from 00 H to $FF \text{ H}$ and we define 256 different type of data we can assign. So that 8085 is also known as 8 bit microprocessor. It is represented by $D_0 - D_7$.

(iii) Control Bus :— These are the individuals lines they are in mixed direction carry synchronisation signal and deliver control signal to peripheral devices. MP generates control signal for every operation such as MEMRi, MEMW, IOR, IOW.

Concept to Multiplexing and Demultiplexing



MPU is responsible for both data transfer and data access. Lower order address bus line $A_0 - A_7$ is combined with $D_0 - D_7$ form the multiplexed data line. If we do not multiplex these lines then we have to add 8 more pins on MPU IC. Hence to reduce the size or area ~~multiplex~~ of the chip we multiplex these lines.

Need of Demultiplexing A_7 to A_0 arrive because some time these ~~the~~ line work as address bus and sometime they work as data bus. This is done by using the latch and buffer. The ALE signal goes high ~~when~~ during first clock pulse and lower address is passed through the latch. and At the second pulse the ALE signal goes Low and provide data for this multiplexed Bus.

The entire operation is performed the opcode fetch fetch decoding and executing.

ALE \uparrow $A_0 - A_7$

ALE \downarrow ~~$A_0 - A_7$~~

Need of Multiplexing and De-Multiplexing \Rightarrow

To concept the size of 8085 IC we multiplex the address line to the data line. We need demultiplexing because at the time of opcode fetch we required address of this signal so at the time of first clk-pulse ALE becomes high and provide the address of the data after first clk pulse ALE goes low and provide data on the particular address location.

Concept of static & Dynamic RAM \Rightarrow

Static RAM:—

- \rightarrow Data is stored in FF like structure.
- \rightarrow Faster or high speed memory
- \rightarrow Volatile memory
- \rightarrow More power dissipation
- \rightarrow Less density
- \rightarrow No refreshing
- \rightarrow High cost
- \rightarrow Less hardware required
- \rightarrow It will continue to hold & store information even when the power is cutoff.
- \rightarrow S-RAM is also known as cache-memory
- \rightarrow Each memory cell requires six transistors
- \rightarrow It has low density but high

Dynamic RAM:—

- \rightarrow Data is stored in MOS capacitor type structure
- \rightarrow It stores the bit in form of charge.

- lower speed of operation
- lower power consumption.
- It requires refreshing to change the data stored in MOS capacitor.
- Cheaper than the S-RAM
- Complicated system design because extra h/w is required to control refreshing.
- It is a volatile memory all the available data are lost when power is switched off.

Types of Memory OR Type of ROM ⇒

- (i) Mask ROM:- In this ROM, bit pattern is permanently recorded by the manufacturer during the fabrication. User can not change any memory location.
- (ii) Programmable ROM (PROM):- In PROM, user can store the program or data only once and reprogramming is not possible. This memory has nitron + nickel + chlorine or polycrystalline wire arranged in a matrix form. These wire can be functionally made as diode or fuse. The user can program this memory with a special PROM programmer that selectively burn the fuse according to the bit pattern stored in the accumulator. Universal PROM programmer

is also available which is more power
ful and provide greater facility.

(iii) EPROM [Erasable programmable ROM] :-

It is Erasable PROM. Erasing can be done by high intensity shortwave UV light and programming can be done by through electrical signal. 10 to 20 minutes are required to erase the content. It is not possible to erase selective information when we erased PROM the entire information is lost.

(iv) EEPROM [Electrical Erasable programmable ROM] :-

EEPROM is program through the electrical signal & erased by using electrical signal. Erasing process is 10 millisecond for erasing the content.

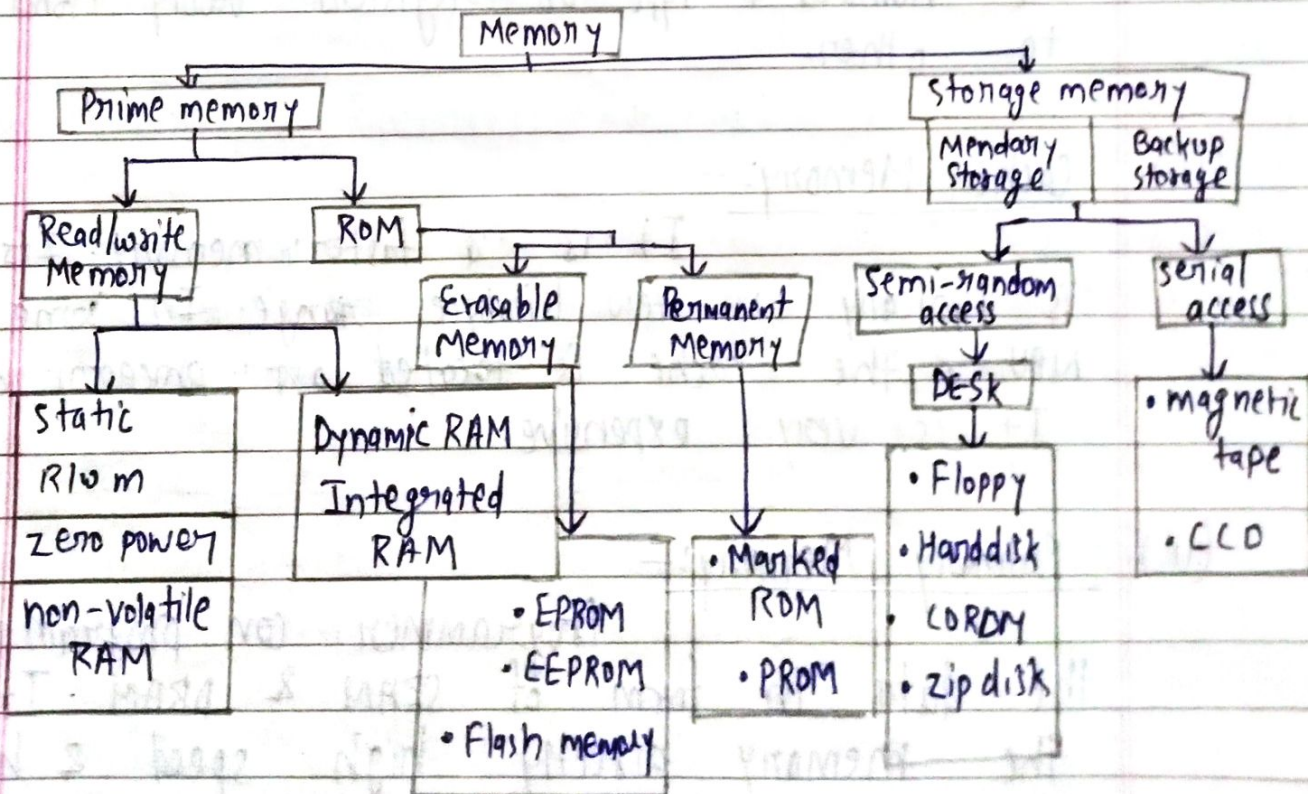
In this memory user can change any particular location or group of location without affecting other. It is expensive memory compare to the EPROM. Erasing time is quite small as compare to the time required to erase PROM. It has high cost, low density and low reliability.

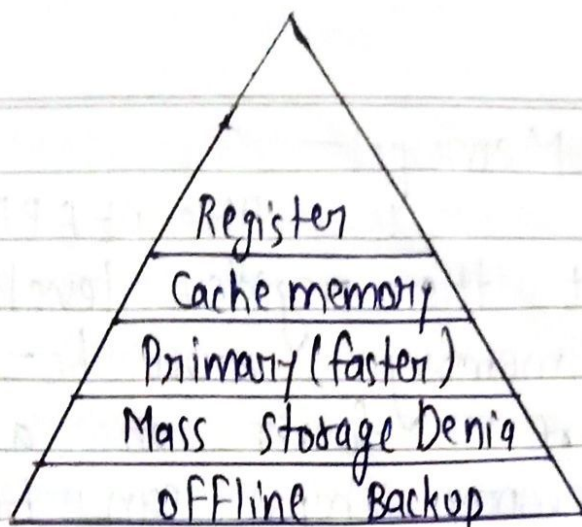
(v) Flash Memory:-

The EEPROM can be erased at the register level but the flash memory must be erased either in entire form or a sector level. These memory chips can be erased and program million times. It provides high packing density, Low cost and fast programming technique.

(vi) Hybrid MOSFET \Rightarrow

MPU uses high speed, high performance and high density, n-channel mos to implement logic gates. But due to greater power consumption is less efficient than N-MOS and P-MOS





Semi-conductor memories are used for storing information in microprocessor based system. The memory type listed towards the top of the pyramid are faster and those listed over the lower end is stored.

(i) Register:-

There are the parts of microprocessor. The amount of register storage is limited from a few hundred bits to a few thousand bits. The number & type of registers vary from one MPU to other.

(ii) Cache Memory:-

It is a faster memory. Its size is typically in few kilobyte range. In some advance MPU, the cache is located on processor chip. It is very expensive.

(iii) Primary Memory:-

Programmer can program and store the data in form of SRAM & DRAM. It access the memory directly high speed & high cost.

(iv) Mass storage:-

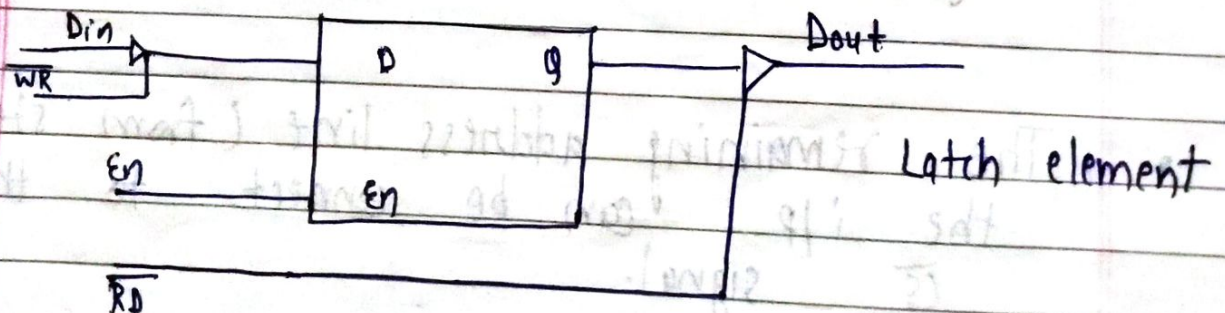
Several programs & data are needed to assign in system so that they can be loaded for execution into the primary memory without delay. Holding these programs & data require several megabytes of memory. These programs and data may not be accessed frequently. These one or more mass storage device is used for store this information.

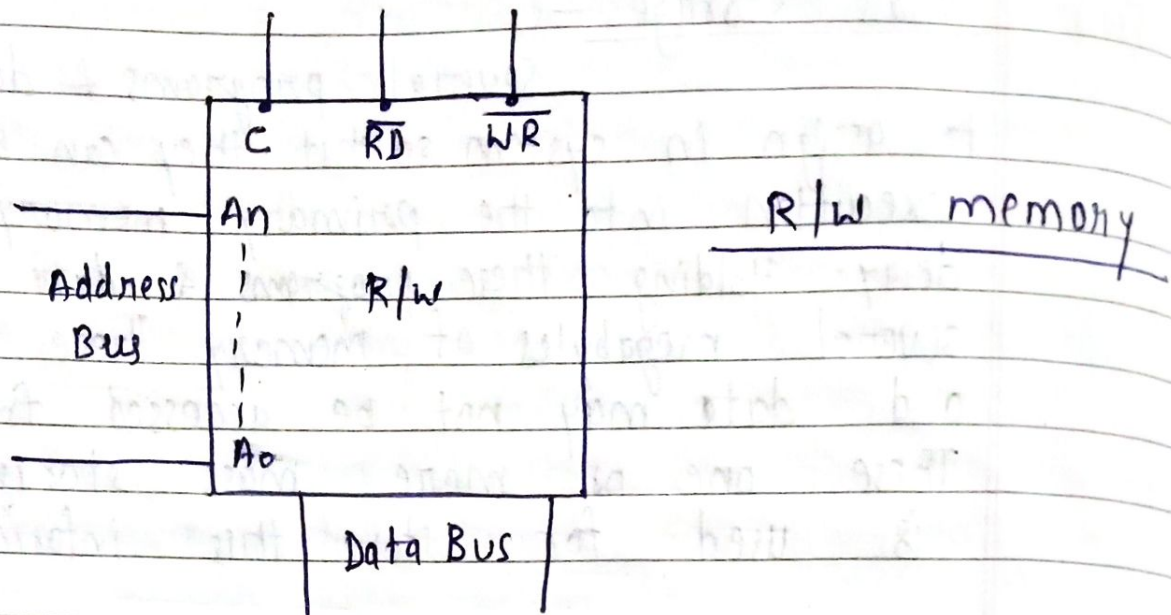
(v) Offline Backup:-

It is a type of removable storage device. When all the mass storage available in a system get used up then offline backup play important role. Removable hard disk and a tab drive are example of offline backup. By using this the user can perform any operations.

Memory Element \Rightarrow

It is a circuit that can store high or low bit as per the applied signal. It can also provide read & write related operation.





Requirement for the memory chip \Rightarrow

- A memory chip require address line to identify a memory register.
- The number of address line required is determined by the number of register in a chip.
($2^n = \text{number of register where } n \text{ is the number of address line}$)
- A memory chip require, a chip select signal (\overline{CS}) to enable the chip.
- The remaining address line (from step 1) of the chip can be connect to the \overline{CS} signal.

- The address line connected to \overline{CS} select the chip & the address line connected to the address lines of memory chip select the register.

- The control signal \overline{RD} enable the output buffer & the data from the selected register are made available on the output lines.

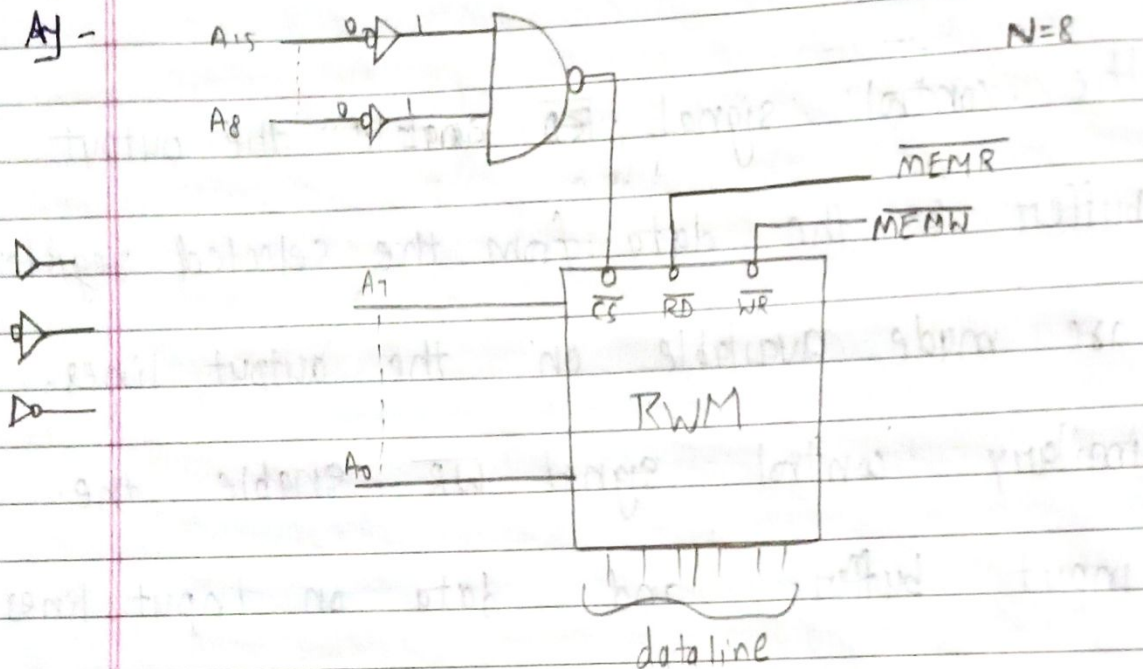
Similar control signal \overline{WR} enable the input buffer and data on input lines are written into memory cells.

Q. Design a memory address range of the chip with 256 bytes of memory. Explain how the range can be changed by modifying the hardware of chip select line.

$$256 = 2^N = 2^8$$

$$N=8$$

Ans -

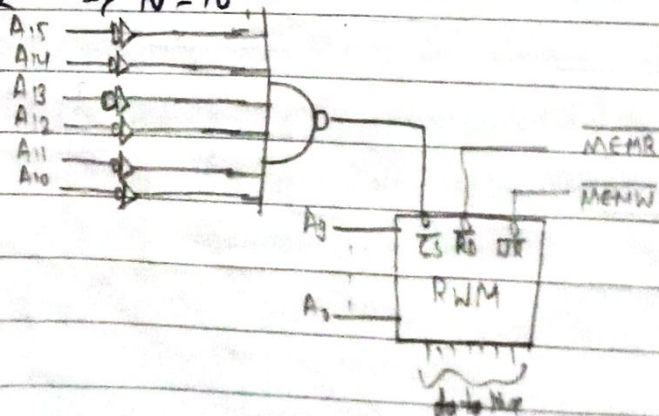


A ₁₅	A ₁₄	A ₁₃	A ₁₂	A ₁₁	A ₁₀	A ₉	A ₈	A ₇	A ₆	A ₅	A ₄	A ₃	A ₂	A ₁	A ₀
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
								1	1	1	1	1	1	1	1

0000H - 00FFH

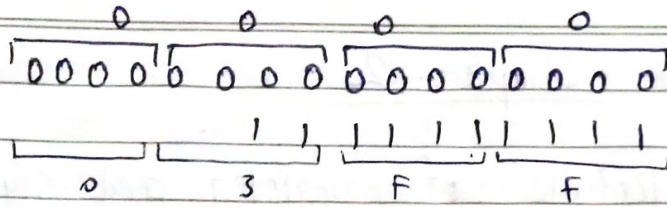
Q. Explain the memory address range of 1K by designing the memory element. Also change the memory address into the range of 2000.

$$1K = 2^{10} \Rightarrow N=10$$

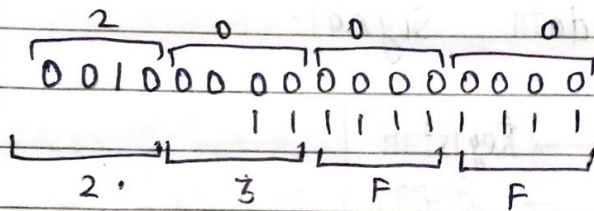
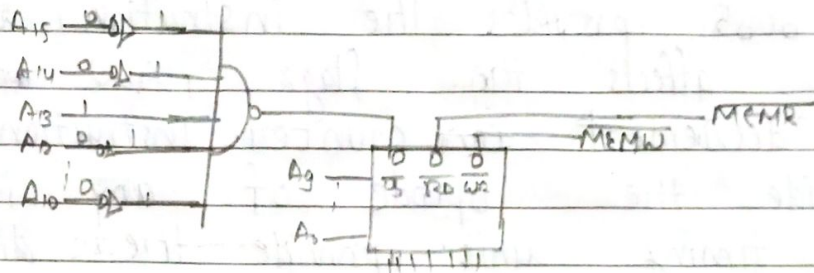


DATE : / /

PAGE NO. :



0000H - 03FFH



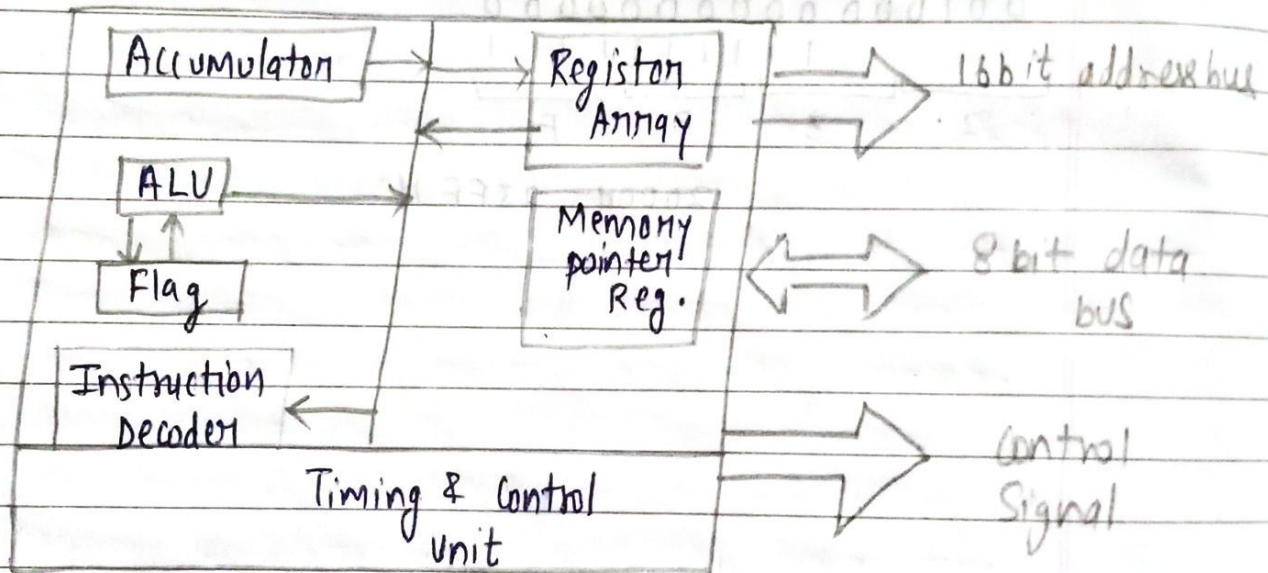
2000H - 23FFH

ing

Chapter - 2

Software Architecture of register and signal \Rightarrow

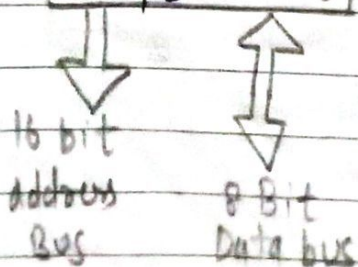
The 8085 provides the instruction which are really affects the flags, ALU and increment and decrement counter. Instruction decoder provide the opcode of any instruction and timing unit provide the different synchronising signal and data signal.



A	8	FR	8
B	8	C	8
D	8	E	8
H	8	L	8
SP		16	
PC		16	

Er Sahil
Ka
Gyan

for:- Register used by UP



Instruction is a set of rule containing opcode and operand whenever an instruction is applied microprocessor perform a specific task. And instruction is a command given to the computer to perform a specified operation on given data.

Classification of instruction \Rightarrow

(i) According to word size:—

(a) 1 Byte instruction:— It is an instruction contain only register & register pair. It is known as 1 Byte instruction.

Eg - `MOV A, B`

Eg - `HLT`

Eg - `ADD B`

(b) 2 Byte instruction:— If an instruction contain 8 bit data then this become

2 Byte instruction.

Eg - `MVI A, 10H`

Eg - `ADI 08H`

(c) 3 Byte instruction:— If an instruction contain 16 Bit address or no. then this become 3 byte instruction.

Eg - `LXI H 3050H`

Eg - `LXLD 3051`

Eg - `SHLD 3000`

Eg - `STA 3020H`

(ii) According to operation perform by instruction:-

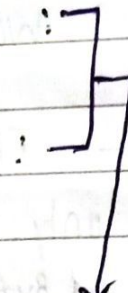
(a) Data transfer instruction :-

→ $\text{mov } R_0, R_5$
eg- $\text{mov } A, B$
 $R_0, R_5 \rightarrow A, B, C, D, E, H, L$

→ $\text{MVI } R, \text{ 8 bit data}$
 $\text{MVI } B, 06H$

→ $\text{IN } \text{ 8 bit address}$
 $\text{IN } 04H$

→ $\text{OUT } \text{ 8 bit address}$
 $\text{OUT } 02H$



Er Sahil
Ka
Gyan

(b) When instruction IN & OUT execute the data at 8 bit address will store in accumulator and data of 8 bit address available at the output.

→ $\text{LXI } R, \text{ 16 bit address}$:- Load in register

→ $\text{LDA } \text{ 16 bit add.}$:- Load in accumulator

→ $\text{LHLD } \text{ 16 bit add.}$:- Load in H & L

→ $\text{STA } \text{ 16 bit add.}$:- store accumulator value in given add.

→ $\text{SHLD } \text{ 16 bit add.}$:- store HL value in give add.

→ $\text{LDAX } R_p$:- Load accumulator indirect. Accumulator is stored from the memory address which is specified in the given register pair.

→ $\text{STAX } R_p$:- Store accumulator value at the memory address which is specified on store in register pair.
eg- $\text{STAX } H$

(b) logical group instruction: —

- (i) OR A R :- OR accumulator with register
- (ii) OR A M :- _____ memory
- (iii) OR I 8bit data :- OR instruction with 8bit data
- (iv) AN A R :- And accumulator with register.
- (v) AN A M :- And _____ memory
- (vi) AN I 8bit data :- And instruction with 8bit data.
- (vii) XRA A R :-
- (viii) XRA M :-
- (ix) XRI 8bit data :-
- (x) CMA :-
- (xi) CMC :- Complement the carry status.

Q. MVI A, 18H
CMA
HLT

⇒ 00011000 → 18H
→ 11100111
E7 H

(xii) STC :- Set carry status
The status carry flag is set and no other values are affected.

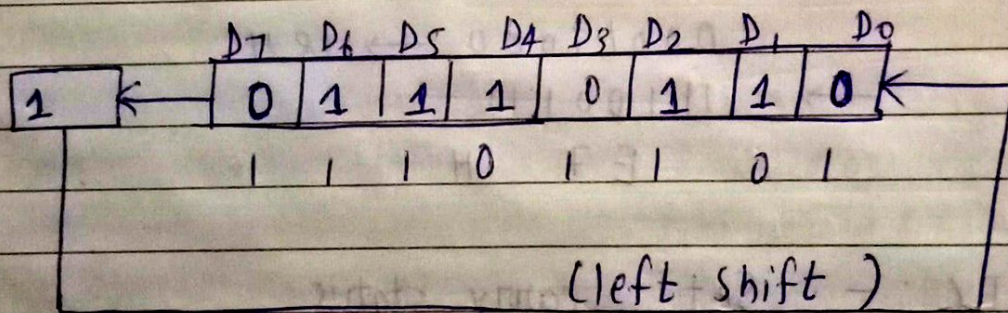
- (xiii) CMP R :- Compare R
- (xiv) CMP M :- Compare memory
- (xv) CMI 8bit data

CMP R:- The content of register is subtracted from the accumulator and status flag are set according to the result of the subtraction. But the result is discarded and content of the accumulator remains unchanged.

(c) Rotate Instructions

① **RAL:-**

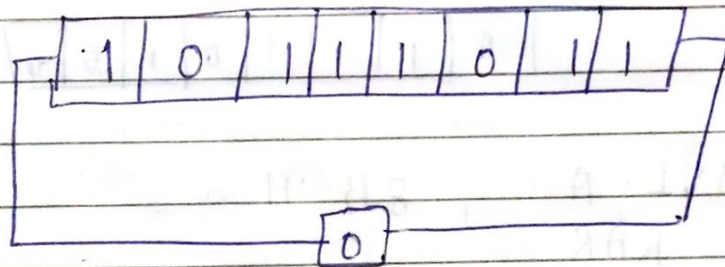
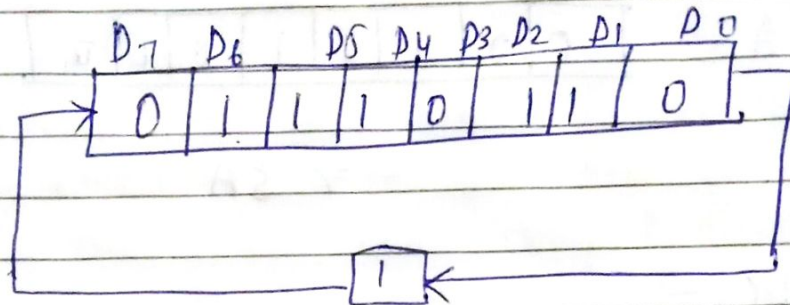
Rotate accumulator left with carry. The content of the accumulator is rotated left 1 bit through carry. The 7th bit of the accumulator is move to carry bit and carry bit is move to D₀ bit.



② **RAR:-**

(Rotate accumulator right). The content of the accumulator is rotate right by one position through the carry flag. Bit D₀ is placed in the carry flag and the bit in the

carry flag is placed in the most significant position D7.



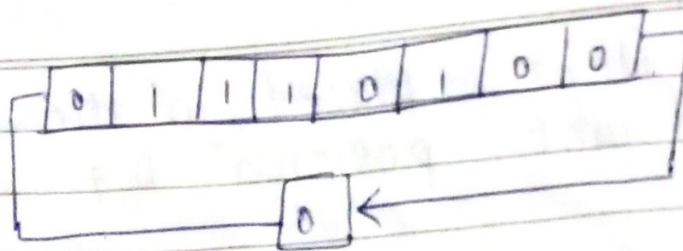
⑤

RRC: -

Each binary bit of accumulator is rotate right by one position bit D0 is placed in the position of D7 as well as in the carry flag.

Q. MVI A, 74H, carry flag = 0
 RAR
 RAR
 HLT
 A = ?

Ans. 74 = 01110100



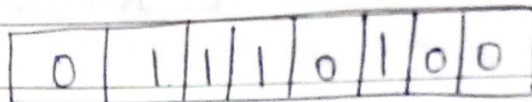
A =

0	0	1	1	1	0	1	0
---	---	---	---	---	---	---	---

⇒ 3A

Ans

Q. (ii) RLC:-



0	1	1	1	0	1	0	0
---	---	---	---	---	---	---	---

 = 74

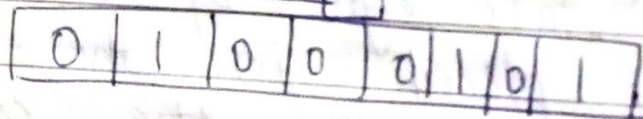
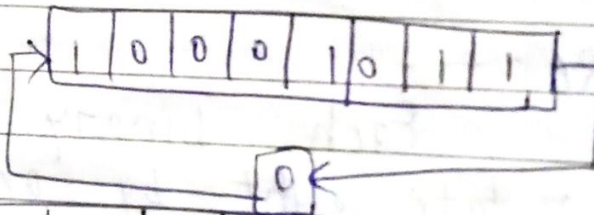
Q. (iii) MVI A, 8B H

RAR

RRC

Ans -

8B =



RAR ⇒ 45 H

RRC:-

101 00010

Arithmetic instruction : —

- (1) ADD R :- Add contain of register with accumulator & store in it.
- (2) ADD M :- Add contain of memory with accumulator & store in it.
- (3) ADI 8bit data :- Add 8 bit data with accumulator & store in result in accumulation
- (4.) ACI 8bit data :- Add 8bit data with accumulator and carry & store in accumulator.
- (5.) DAD Rp :- The contained of the registered pair BCD H, L are added to the contain of HL and sum is stored in HL pair.
- (6.) SUB R :- Subtract contain of register with accumulator & store in it.
- (7.) SUB M :- Subtract contain of memory ^{location} with accumulator and store in it.
- (8.) SUI 8bit data :- subtract 8bit data with accumulator & store in accumulator and carry
- (9.) SBB R :- Subtract contain of register with accumulator and store in it.
- (10.) SBB M :- ~~1~~ contain of memory location and carry states subtract from the accumulator & store in it
- (11.) SBI 8bit data :- Subtract carry data & 8 bit data with accumulator and store in it.
- (12.) INR R :- Increase the value of register,
- (13.) DNR R :- Decrease the value of register.
- (14.) INX Rp :- Increase the value of pair of register.
- (15.) ~~DNX~~ RP :- Decrease the value of pair of register.

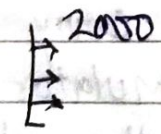
(16.) DAA :— The instruction is used in the program after the execution of any instruction. It provides the decimal value of the hexadecimal number. It used the carry & auxiliary carry for decimal system. 6 is added to the LSB of the contain of the accumulation if the value lie b/w A to F.

Branch Instruction \Rightarrow The instruction of this group change the normal sequence of the program. There are two type of branch instruction:—

(i) Conditional (ii) Unconditional

(i) Conditional BI:— It transfers the program to the specify level when certain condition is satisfied.

(ii) Unconditional BI:— It transfers the program to the specify level unconditionally.

Eg— JMP add.

 2000 JMP 3000
 3000 ADD B

Conditional Jump:—

After the execution of the conditional jump instruction the program jumps to the instruction specified by the address if the specified

condition is full-fil.

(A) (a) JC 16 bit address :— Jump if there is carry
on $\boxed{Cy = 1}$

(B) JNC 16 bit add. :— Jump if there is no carry
i.e. $\boxed{Cy = 0}$

(C) JP :— Jump if result is ~~plus~~ pluse. $\boxed{S = 0}$

(d) JM :— Jump if result is minus. $\boxed{S = 1}$

(e) JZ :— Jump if result is zero. $\boxed{Z = 1}$

(f) JNZ :— Jump if the result is not zero. $\boxed{Z = 0}$

(g) JPE :— Jump if result is even parity. $\boxed{P = 1}$

(h) JPO :— Jump if result is odd parity $\boxed{P = 0}$

(B) CALL :— Call instruction is used to call a subroutine before the control is transferred to the subroutine the address of the next instruction of the main program is stored in the stack. The content of the stack pointer is decremented by 2 to indicate the new stack top.

CC :— call subroutine if carry status = 1

CNC :— call subroutine if $Cs = 0$

CZ :— call subroutine if result is zero $\boxed{Z = 1}$

CNZ :— If result is not zero $\boxed{Z = 0}$

CP :- If result is +ve
 CM :- If result is minus
 CPE :- If result has even parity
 CPO :- If result has odd parity

S=0

S=1

P=1

P=0

RET :- Return instruction is used at the end of a sub-routine and before the execution of sub-routine the address of next instruction of the main program is saved in the stack. The execution of return instruction brings back the same address from the stack to the program counter.

RC, RNC, RZ, RNZ, RP, RM, RE, RPO

(c) RST n :- Restart is a 1/3 Call instruction. It restarts the program from the given memory location. The instructions are generally used in conjunction with interrupt & inserted using external h/w. These can be used as s/w instructions in a program to transfer program execution to one of the 8 locations.

RST 0 0000 H

RST 1 0008 H

RST 2 0010 H

RST 3 0018 H

RST 4 0020 H

RST 5 0028 H

RST 6 0030 H

RST 7 0038 H

(D) PCHL :- The content of the HL pair transfer to the program counter. The content of register H move to the high order 8 bit of register program counter. The content of register L are low order 8 bit of register program counter.

~~(E)~~ STACK, i/o & Machine control group instruction :-

(a) HLT :- The execute of Hold stop the microprocessor

(b) EI :- Enable interrupt instruction is used for enabling the interrupt at the time of execution.

(c) DI :- ^{When} Disable interrupt is executed the interrupt are disable.

(d) XTHL :- The content of the register L are exchanged with the byte of STACK TOP. The content of register H exchanged with byte below the stack top.

(e) SPHL :- The content of the H & L pair are transferred to the stack pointer register.

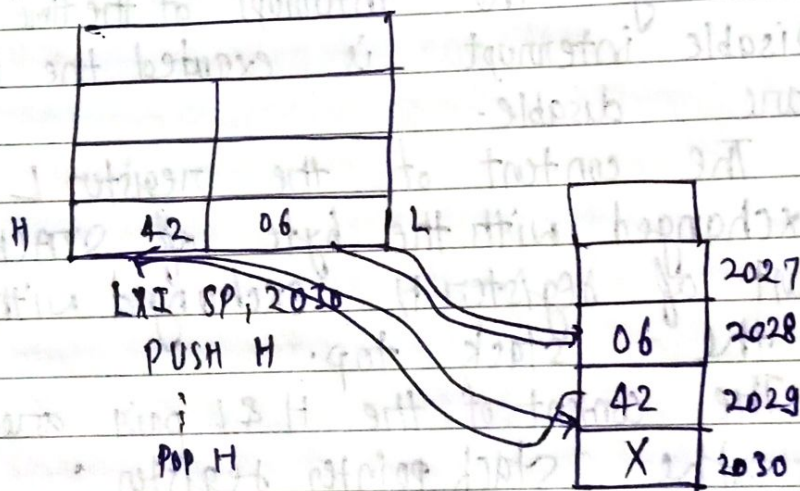
(f) NOP :- It is performed when we required that the microprocessor don't execute any instruction.

(g) STACK :- STACK in an 8085 MPU can be described as a set of memory location in the read write memory. The beginning of the stack is defined in the program by using the instruction LXI SP, data byte in the register pair's MPU can be stored on the stack in the reverse order by using

the instruction PUSH.

The data byte can be transferred from the stack to respective register by using the instruction POP.

The SP register track the storage and retrieve the information because two data bytes are being stored at a time. 16 bit memory address in the SP register is decremented by 2. When data bytes are retrieved the SP register is incremented by 2.



PSW (Program status word):—

Push and POP PSW instruction save the contain of the accumulator and flags and after execution of the program retrieve the content.

(H) SUB ROUTINE:—

A sub routine is a group of instruction return separately from the main

program to perform the function that occurs repeatedly in the main program.

8085 has to instruction to implement sub-routine. The call instruction is used in the main program call a sub-routine and return instruction is used at the end of the sub-routine to return to the main program.

MVI A, 36

ADD B

INR C

2000 Call 2070

2003

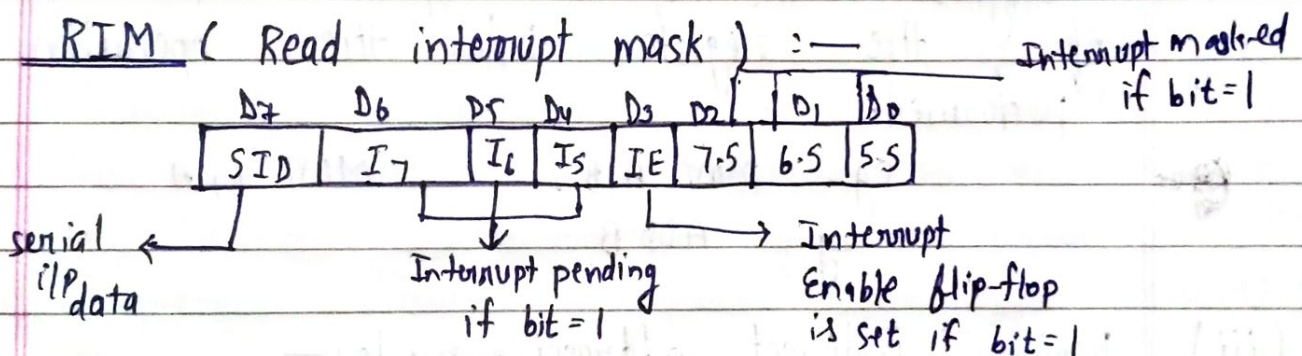
INX H

DCR C

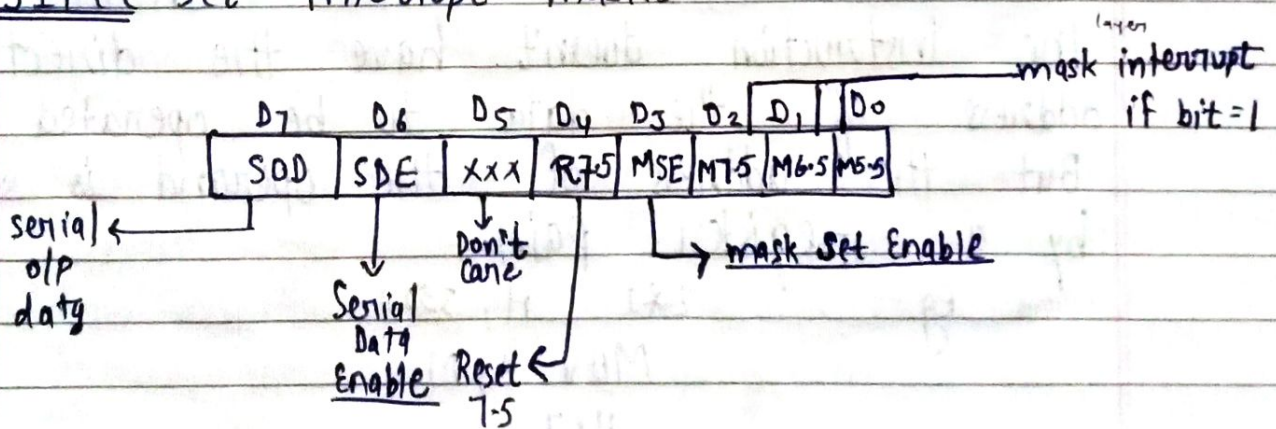
MVI C, 00

RET

RIM (Read interrupt mask) :-



SIM (Set interrupt mask) :-



Addressing mode \Rightarrow

Each instruction is required on which it has to operate. There are

various technique to specify data for instruction. These techniques are called addressing mode.

(i) Direct addressing mode :-

In this the operand is given by direct address where the data is present in the given memory location

eg - LDA 3500
STA 2560 H

(ii) Register addressing mode :-

In register addressing mode, data is in the general purpose register. The opcode specifies the address of the register into the operation to be performed.

eg - MOV A, B , MOV C, A
eg - ADD B

(iii) Register indirect addressing mode :-

In this mode, the instruction doesn't have the direct address of the data to be operated on. But the address of the operand is specified by a register pair.

eg - LXI H, 3500
MOV A, M
HLT

(iv) Immediate addressing mode :-

In this mode, the data or operand to be used immediately given in the

instruction itself.

eg-
 MOV A, 18H
 ANI 08H
 LXI H, 5600H

(v) Implicit addressing mode :-

There are certain instructions which operand is the content of the accumulator. Such instructions do not require the address of the operand.

eg- RAL, RAR, CMA

Assembly Language Programming And Debugging ⇒

A user can interact with the microprocessor through the assembly language instruction. These instructions are the command to the MPU to be executed in the given sequence to complete task. Assembly language instructions are also called Mnemonics.

Programs can have two type of structure

(i) Sequential program :-

The instruction of the sequential program executed one after one without any change in the sequence. It means that there is no branch instruction in the program.

(ii) Iterative program :-

The program in which a segment of program is repeated

to complete task or to solve a difficult problem.
It can be classified into 2 groups

(i) Unconditional :— In such type of program, the segment of program is repeated without any condition.
eg - Call, Jump instruction doesn't check any condition and move further to another memory location.

(ii) Conditional :— In this program, the segment of program is repeated after checking the condition. If condition is satisfied repetition is done otherwise not.

Debugging of The Program \Rightarrow

the program is similar to Debugging of trouble shooting h/w but it is much more difficult.

Some clue give alert to the programmer to know about the error made by him. Either the program work or doesn't

The position of the program is known by debugging of program. It can be divided into 2 process:—

(i) Static technique :— In Static debugging, examiner do visual inspection of h/w by a paper and pencil and check the flowchart or machine code. Some clue for finding error in the program.

- (a) Selecting a wrong machine code
- (b) Specify the wrong group instruction
- (c) Giving wrong order of high and low order bytes.
- (d) Failure to set the flag before using jump instruction.
- (e) Failure in indexing and counting
- (f) Failure to clear the accumulator when it is used in arithmetic operations
- (g) Failure to clear register when it is used to crack ~~flag~~ some data like carry or auxillary carry.

(ii) Dynamic technique:— The tool of dynamic debugging are

(a) Single step:— This key on the keyboard on single board microcomputer allow to execute one instruction at a time and after that programmer can check for the expected result by examining memory or register.

(b) Break point:— It is provided by software routine that allow the programmer to execute a program in a section. Break point inserted in the program by using RST instruction.

(iii) Programming technique:— PT are used to perform the repeated task. A loop is set-up by instructing the MPU to change the sequence of

execution and performed task again on the next data.

(a) **Indexing**:- Indexing means we made a point which point to the stored data. According to this programming technique, a register pair is used to store the address of first data. When the task had been completed on the pointed data the pointer is incremented by 1. And the task is repeated again from the new pointed data.

(b) **Counting**:- This programming tech. allow the programmer to counter how many times the instruction on a segment of program is to be executed. For that number, a counter is set. The value of counter decremented each time whenever one iteration is completed. The process is repeated till the counter reach to 0.

(c) **Branching**:- After the task is completed at the one time the same instruction is repeated with new data to repeat the execution of the instruction again and again it is required that the last instruction should branch the sequence of execution to the first instruction of the loop.

Instruction Format \Rightarrow

Each instruction contains 2 specific information field. One is the operation code which specify the operation to be performed. The operation is specified by binary code. Hence the name of this is operation code. Second is operand which say about the data on which task is to be performed. First field of the instruction is always opcode but the second ~~field~~ field may be source/destination operand, source/destination operand address, 8 bit data with register or next instruction address opcode.

The length of the instruction may be 1, 2 or 3 byte.

Opcode format:- For an assembly language program, opcode is operation code which is represented by an English like word.
Eg - ADD, SUB, MOVE,

The opcode is unique for each instruction & contain the information about operation and, register or memory operation to be used.

8085 has 8 bit opcode which identify all operation, register and flags with this specific course.

B	000
C	001
D	010
E	011
H	100
L	101
M	110
A	111

BC	00
DE	01
HL	10
A Forse	11

Timing \times Operation \Rightarrow

The program is a set of instructions written in a sequence and perform predefined task. To execute an instruction the μ PU need to perform various operation related to memory or IO devices.

Instruction Cycle :-

ICy is defined as a time required to complete the execution of an instruction. It may include one to 6 operation (opcode fetch, memory read, memory write, IO read, IO write & stack) to complete the execution of 1 instruction.

Machine Cycle :-

Machine cycle is defined as a time required to complete the one operation. The operation can be accessing the memory I/O on ack. the external signal.

T-state :-

T-state is defined as 1 sub-division of the operation perform in one clock period. It is the unit in which time of any operation is measured. These sub divisions are internal status synchronizes with the internal clock. Each machine cycle consists the of 3 to 6 state.

S_1 S_0 \rightarrow full opcode
 1 1 \rightarrow read
 1 0 \rightarrow read
 0 1 \rightarrow write

DATE : / /

PAGE NO. :

g-	M/C	T	operation
(i) MOV A, B	1	4	F
(ii) MOV A, M	2	7 (4+3)	FR
(iii) LXI H, 16 bit add.	3	10	FRR
(iv) STA \rightarrow 16 bit add.	4	13	FRRW
(v) SHLD 16 bit add.	5	16	FRRWW
(vi) SBB M	2	7	FR
(vii) RAL	1	4	F
(viii) STAX H	2	7	FW
(ix) INR M	3	10	FRW
(x) STACK related	3	12	SWW

Timing Diagram \rightarrow

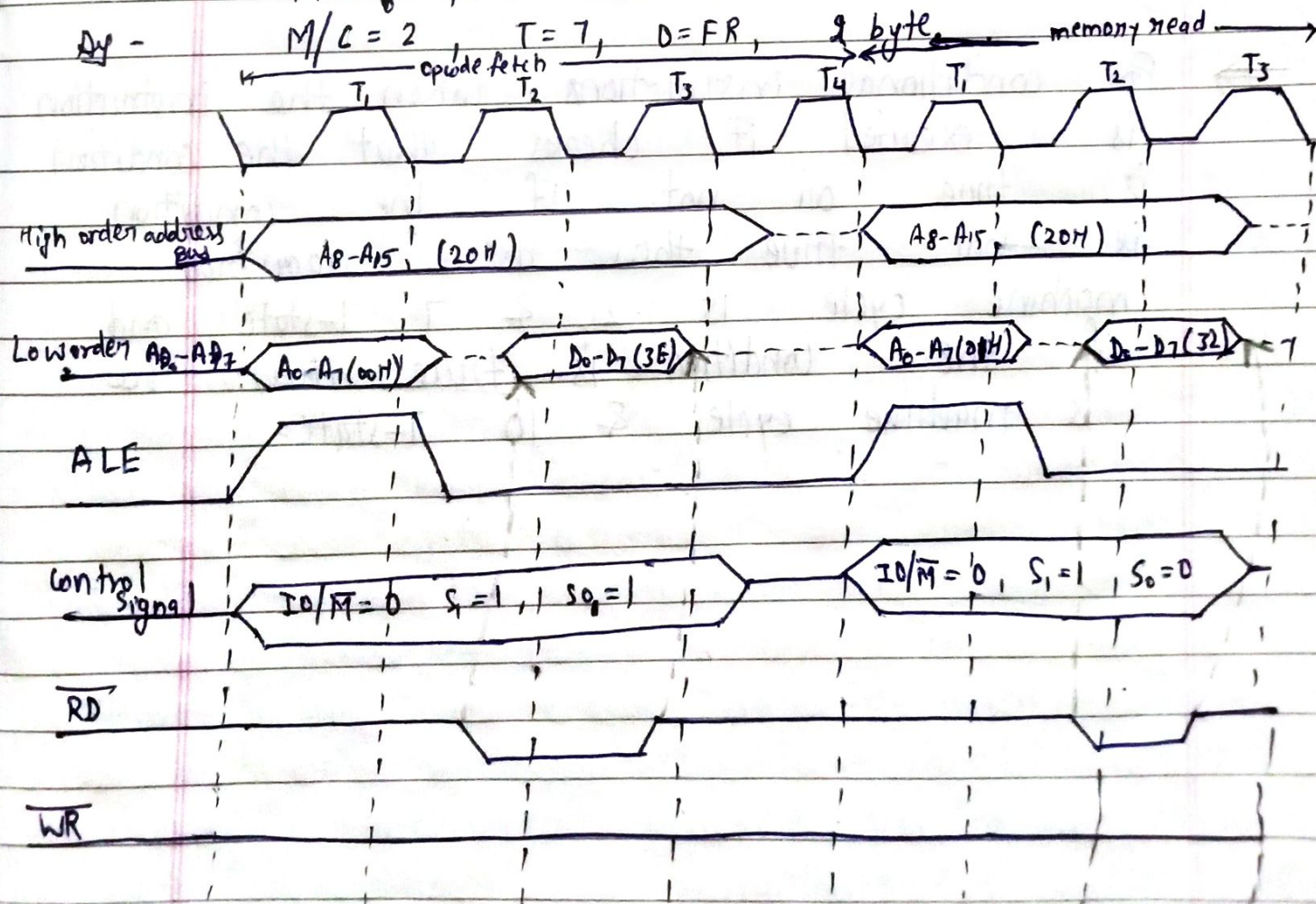
Q.

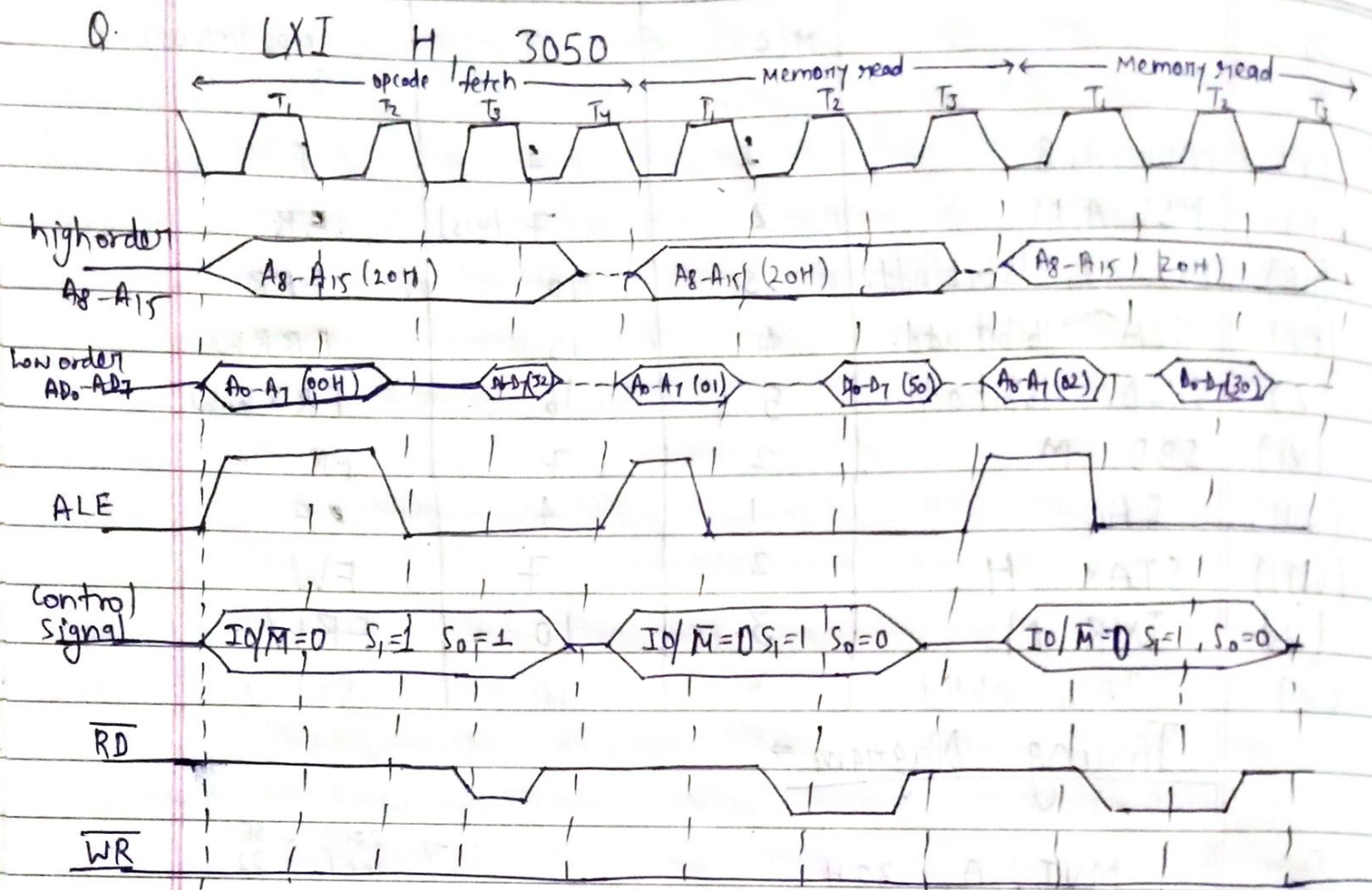
MVI, A, 32H

$\Rightarrow 2000 = 3E$
 $2001 = 32$

Ans -

M/C = 2, T = 7, D = FR, 2 byte, memory read





⇒ For conditional instructions when the instruction is executed it checks that the condition is true or not. If the condition is not true then the required machine cycle is 2 & 7 T-state and if the condition is true then the 3 machine cycle & 10 T-state.

Advance assembly language Programs

Advance assembly language programming is used in the subroutine program or we use stack for programming or to make program easy. In this programming the program is divided into small program called subprogram or subroutine. The stack is data structure initialised through stack pointer in read-write memory that is used for temporary storage of information. Stack stores the data in LIFO manner.

→ The application of this programming are Traffic signal, digital clock, process control, serial data transfer & counter.

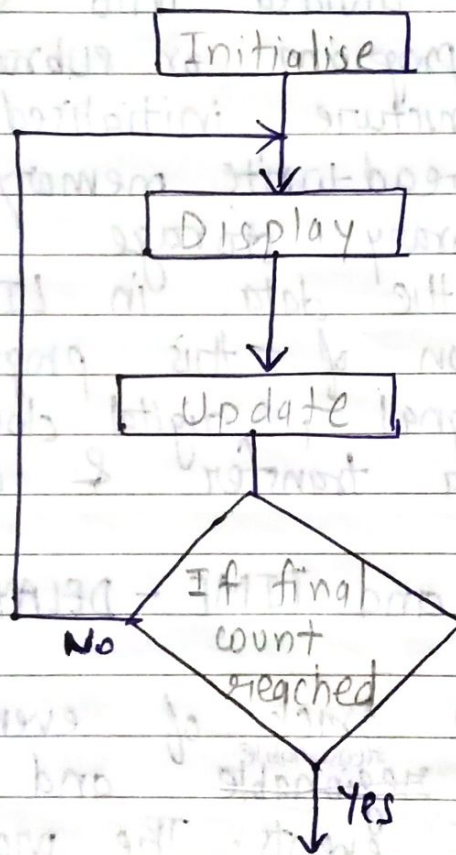
COUNTER and TIME - DELAY ⇒

Counters are used to keep track of event and setting up the ^{reasonable} ~~reasonable~~ and accurate timing b/w two events. The process of designing counter and time delay using slw instruction is more flexible and less time consuming than the design process using hlw.

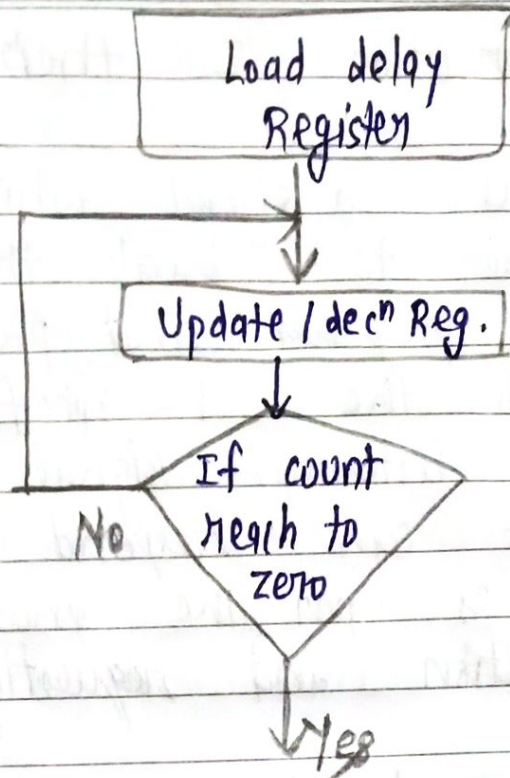
Counter:-

A Counter is designed simply by loading an appropriate number into one of the register and using the increment or decrement instructions. A loop is designed

Who to update the counter & each count is checked to determine that it has reached the final number or not. If the final counter is not reached the loop is repeated. To observe the counting there must be an appropriate time delay b/w instructions.



Time-delay : — The processor is used to design a specific delay is similar to that used to setup a counter. A register is loaded with a number depending on the time delay require and then the register is decremented until it reach zero by setting up a loop with the conditional jump instruction. If loop cause the delay it count by the MPU.



The counter defines the maximum value of the process. In counter, we set the maximum time period from lowest to highest value

eg-

MVI C, FFH	7
loop: DCR C	4
JNZ loop	10/7

clock frequency = 2 MHz

$$\text{Time period} = \frac{1}{F} = 0.5 \mu\text{s}$$

$$T = 0.5 \mu\text{s}$$

$$\text{Time delay for MVI} = 7 \times 0.5 = 3.5 \mu\text{s}$$

$$\text{Time delay for loop}$$

$$= 14 \times 0.5 \times 4 + 11 \times 0.5 \times 1 = 33.5$$

$$\text{Total time delay for program} = 3.5 + 33.5 = 37.0 \mu\text{sec}$$

Type of instruction & their uses \Rightarrow

The interrupt is defined as a signal which is generated by peripheral to break the sequence of the main program and program execution jump through the T-specified memory location. The interrupt signal is generated by peripheral it can respond or discard the request as per the priority b/w current going operation and requesting operation.

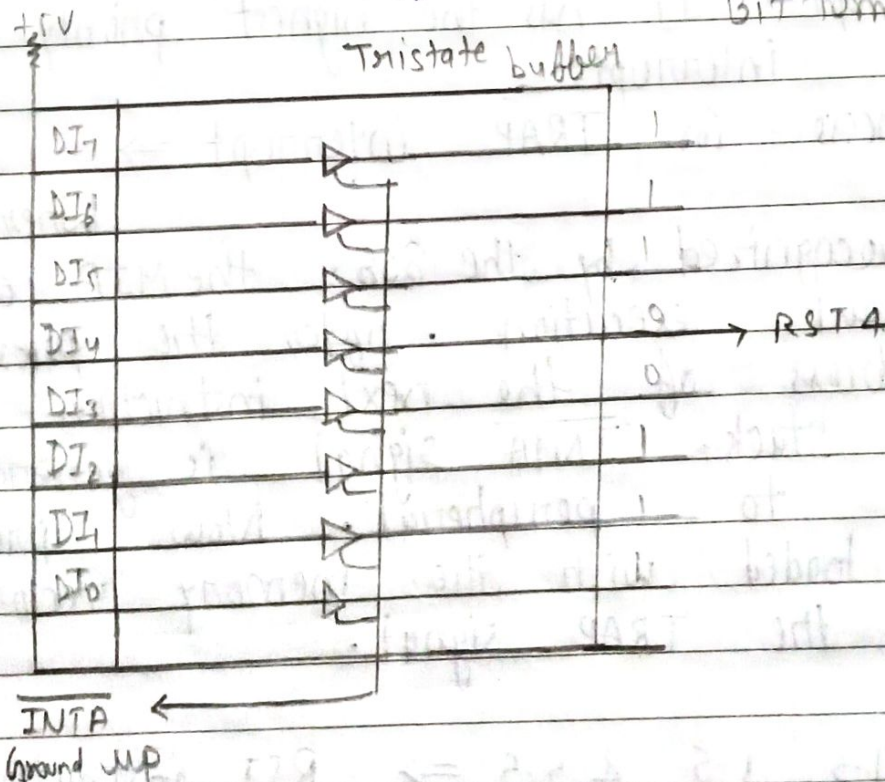
(*) SLW interrupt \Rightarrow

The interrupt cost by executing special interrupt instruction called slw interrupt. After execution of this instruction microprocessor complete the execution of the instruction which is currently executing and transfer the program control to the sub-routine program. After the completing the execution of sub-routine the program return to the main program. 8 RST instruction are executed with the instruction given by MIP. Whenever RST instruction is execute the address in program counter is stored into the stack before the program execution is transferred to the RST call location. When return instruction is find the program control return back to the main program.

Instruction	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	Call memory add ⁿ	Hex code
RST 0	1	1	0	0	0	1	1	1	0000 H	C7 H
RST 1	1	1	0	0	1	1	1	1	0008 H	CF H
RST 2	1	1	0	1	0	1	1	1	0010 H	D7 H
RST 3	1	1	0	1	1	1	1	1	0018 H	DF H
RST 4	1	1	1	0	0	1	1	1	0020 H	E7 H
RST 5	1	1	1	0	1	1	1	1	0028 H	EF H
RST 6	1	1	1	1	0	1	1	1	0030 H	F7 H
RST 7	1	1	1	1	1	1	1	1	0038 H	FF H

The RST instruction is executed by using the external h/w and the signal ~~interrupt~~ INTA in response to a INTR 8085 send INTA which is used to enable the buffer and transferring the program contain to the define memory location.

Bit format



(b) H/w interrupt \Rightarrow In 8085 mip. pins are used to receive interrupt request are called h/w interrupt. It has 5 interrupt TRAP, RST 7.5, RST 6.5, RST 5.5, INTR. Whenever any interrupt pin of the MIP is activated by the peripheral, the execution of program jump to the predefined memory location. The predefined location is the memory address of the service routine for the respective interrupt.

(i) TRAP :- TRAP is a non-maskable interrupt which is also called NMI. It is unaffected by any mask on interrupt enable. It is both +ve logic and level triggered interrupt. It has the highest priority among the all interrupt.

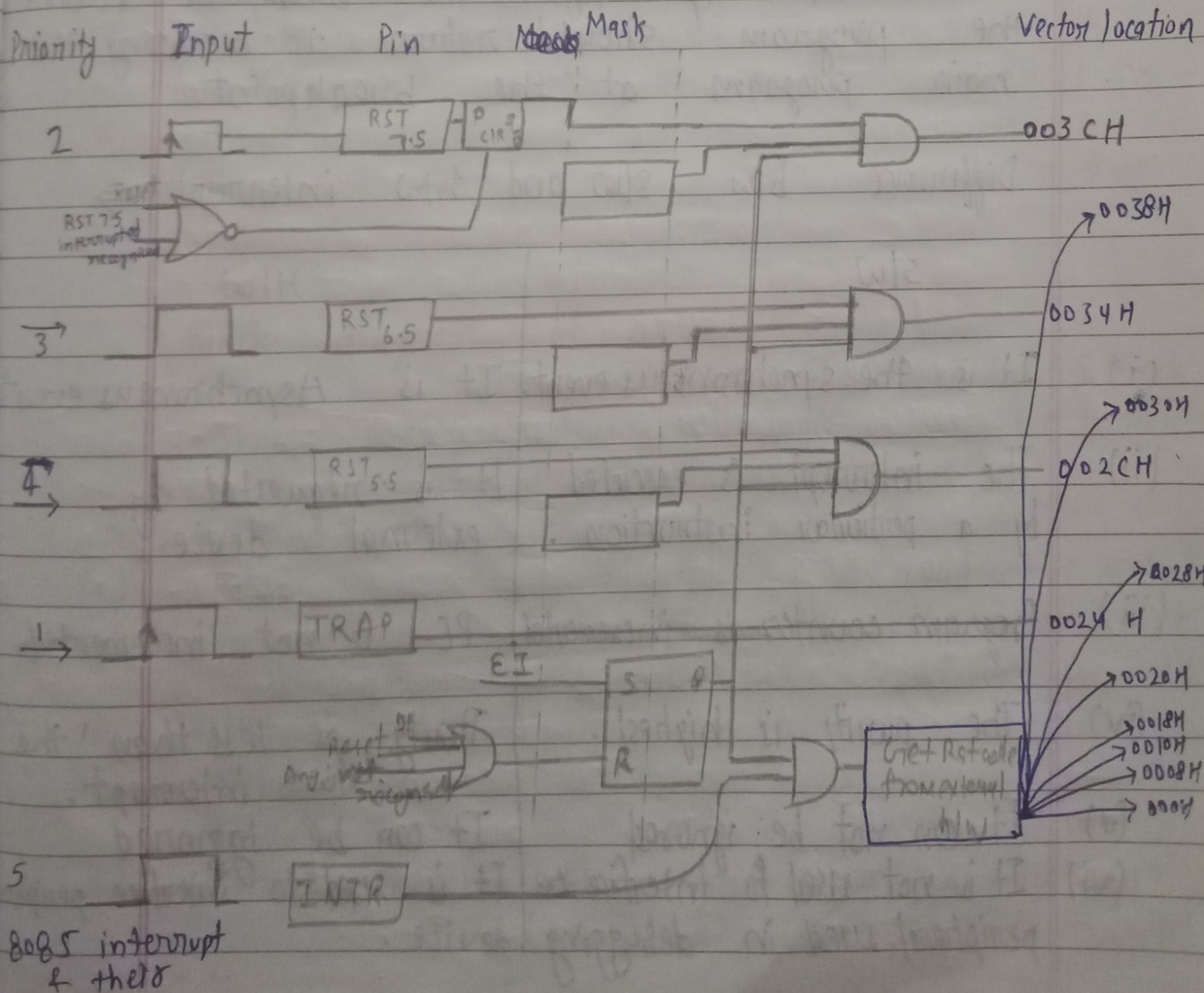
Process for TRAP interrupt \Rightarrow

When trap signal is recognized by the 8085 the MIP complete the current execution. After the present instruction, address of the next instruction store into the stack. INTA signal is generated and sent to peripheral. Now program counter is loaded with the memory vector address of the TRAP signal.

(ii) RST 7.5, 6.5 & 5.5 \Rightarrow RST instructions are +ve edge trigger signal all are maskable interrupts. These are mask through appropriate signal & h/w technique.

The μP send ack. to the peripheral whenever these interrupts are received. After that it is stored the address of next instruction into the stack and jump to the vector address.

(iii) INTR :- It is non-vectorized maskable interrupt additional h/w is required to determine the address of the interrupt service routine. In response to the $\overline{\text{INTA}}$ signal the additional h/w place on instruction opcode on the data bus.



RST instructions and their use \Rightarrow

RST instructions are commonly used to set a slow breakpoint as a debugging technique. A breakpoint is a re-start instruction in a program where a execution of the program stop temporary and program control is transferred to the RST location.

The program should be transferred from RST location to the breakpoint service routine to allow the user for examine the register & the memory. After the breakpoint routine the program should return to executing the main program at the breakpoint.

Difference b/w slow and h/w interrupt \Rightarrow

slow	h/w
(i) It is the synchronous event.	It is Asynchronous event.
(ii) The interrupt is executed by a particular instruction.	It is requested by external device.
(iii) Program counter is incremented.	PC is not incremented.
(iv) The priority is highest.	Priority is less than the slow interrupt.
(v) It can not be ignored.	It can be ignored.
(vi) It is not used to interface the peripheral, used in debugging device.	It is used to interface peripheral.

8259 Programmable Interrupt Control \Rightarrow

The 8259 work with 8050 for controlling the interrupt operation of 8050. It can

- (i) Manage 8 interrupt according to the instruction written into its control register.
- (ii) It can vector an interrupt request anywhere in the memory map
- (iii) Resolve 8 level of interrupt priority in a variety of mode such as fully nested mode, automatic notation mode & specific rotation mode.
- (iv) Mask each interrupt request separately.
- (v) Read the status of pending interrupt in service interrupt and mask interrupt.
- (vi) It can be worked with 8085 as well as 8086

Block diagram of 8259 :-

It includes 8 block control logic, read-write logic, data bus buffer and 3 register (~~IRR~~ , ISR , IMR).

- (i) Read write logic \Rightarrow It is a read write control logic. When the address line A_0 is at logic zero the controller is selected to write a command or read a status.
- (ii) Control logic \Rightarrow This block has 2 pins INT and \overline{INTA} . The INT is connected

to the interrupt pin of the MPU. Whenever a valid interrupt is generated the signal goes high. The INTA is the interrupt ack. signal from the MPU.

(iii) Interrupt registers & priority Resolver \Rightarrow The

IRR (Interrupt Request Register) has 8 i/p lines $IR_0 - IR_7$. When these lines go high the request are stored in the register. The ISR (Interrupt in service register) store all the level that are currently being serviced. And the IMR (Interrupt Mask Register) store the masking bit of the interrupt line to be mask. The PR (priority resolver) examine these 3 register and determine that the INT should be sent to the MPU or not.

Cascade

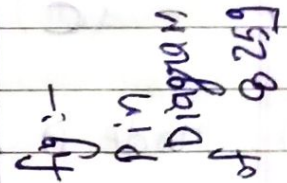
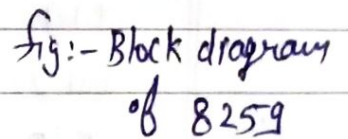
(iv) ~~Cascade~~ Buffer / Comparater \Rightarrow The block is used to expand the no. of interrupt level by cascading two or more 8259.

WORKING of 8259 \Rightarrow

\rightarrow The ISR stores the request.

\rightarrow The PR check 3 register.

\rightarrow The IRR for interrupt request
IMR for masking Bit.



- ISR for interrupt request is being served. It resolves the priority and set \overline{INT} high when ^{appropriate}.
- The MPU ack. the interrupt by sending \overline{INTA}
- After the \overline{INTA} is received the appropriate priority bit in the ISR is set to indicate which interrupt level is being served.
- The Call instruction placed on the databus & it places to \overline{INTA} signal on the databus.
- During a transition of \overline{INTA} signal, call address

of the memory location is placed in the control register.

→ During the 3rd \overline{INTR} signal the ISR Bit is reset by the priority mode.

→ The program sequence is transferred to the next memory location specified by the call instruction.

8255A:— 8255A has 24 I/O port & 8 Bit bidirectional I/O

Operating mode of 8255:—

(i) Input/output mode

(ii) Bit set/reset mode (BSR mode)

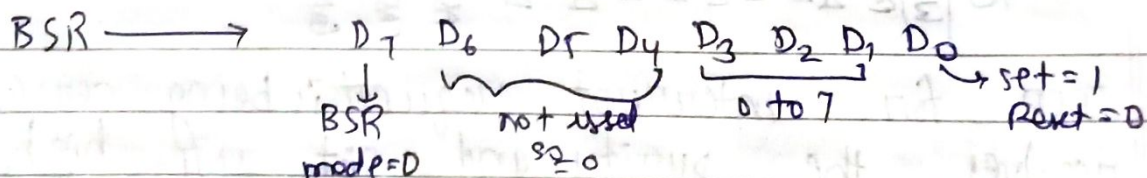
(i) I/O mode again classified into 3 types

— (A) mode 0

(B) mode 1

(C) mode 2

(A) Mode 0:— This functional configuration provides simple input & output.



(B) Mode 1:— handshake signals are exchanged b/w the microprocessor & peripherals prior to data transfer. Each port uses 3 lines from port C as handshakes.

8254 programmable interval timer: — For generate clock pulse, for delay or take some time before pulse & for counting signals.

Mode 0 — Interrupt on terminal Count

Mode 1: — H/w Retriggerable one-shot

Mode 2: — Rate Generator (Count pulse)

Mode 3: — Square wave mode

Mode 4: — S/w triggered strobe

Mode 5: — H/w Triggered Strobe [Retriggerable]

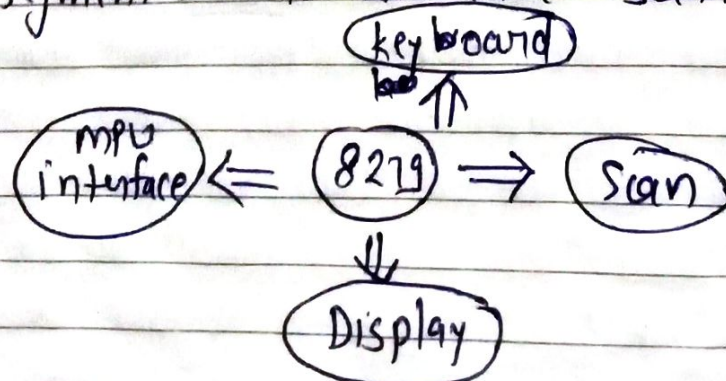
Application of 8254 →

Real time clock, event counter, digital one-shot, programmable rate generator, square wave generator, Binary rate multiplier, complex waveform generator, complex motor control.

8279 : — It is used to provide interface b/w keyboard & display

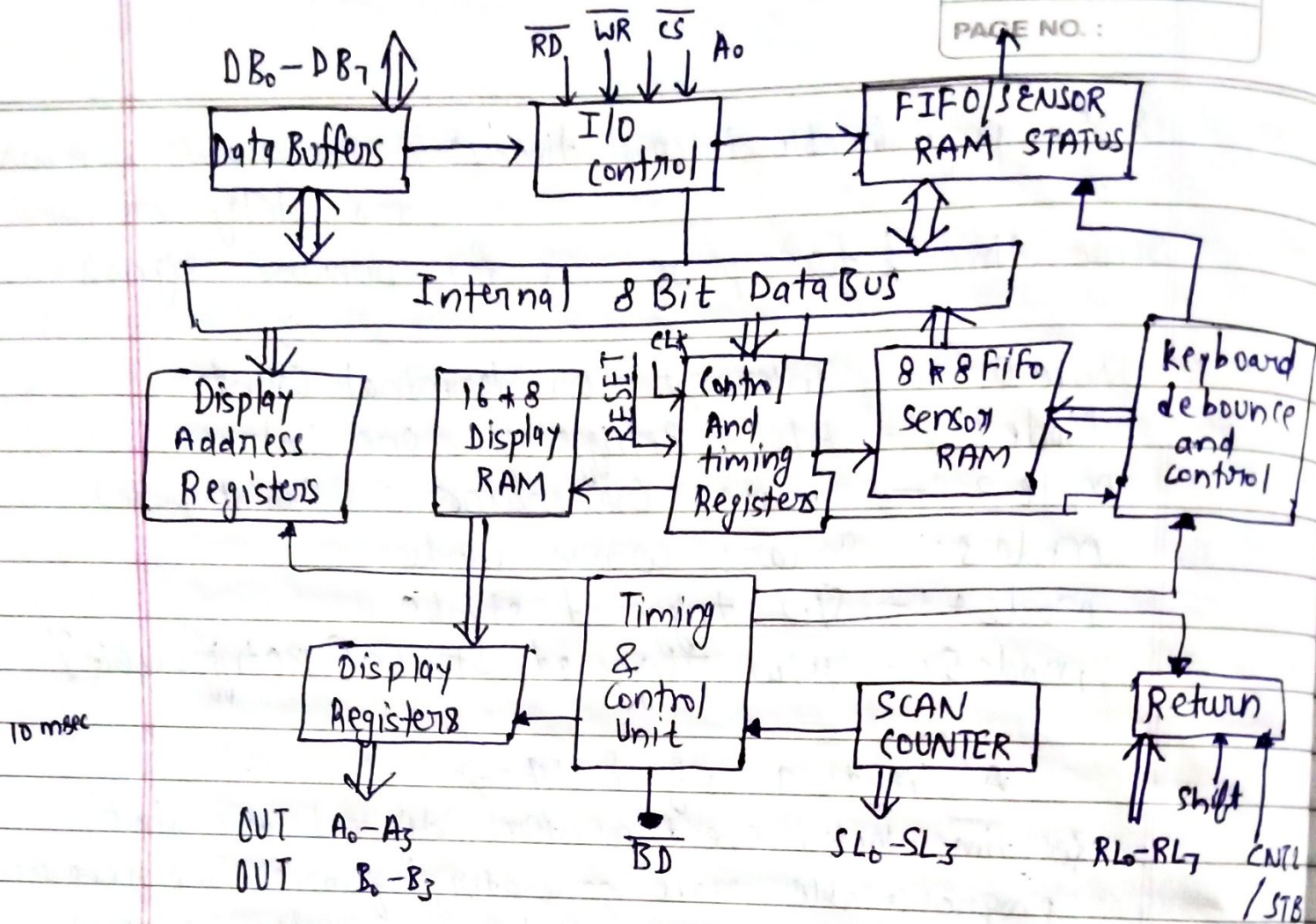
Keyboard segment → connected to 64 contact key matrix keyboard entries & debounced & stored in FIFO. Interrupt signal is generated with each entry.

Display segment: — 16 character scanned display.



DATE : / /

PAGE NO. :



8279 Internal Architecture

Q.1 Explain various programming modes of 8279 keyboard & Display controller?

Ans The INTEL 8279 is specially developed for interfacing keyboard and display devices to 8085/8086/8088 up based system. The important features of 8279 are:

- Simultaneous keyboard & display operations
- Scanned keyboard mode
- Scanned serial mode
- 8-character keyboard FIFO
- 16-character display
- Right or left entry 16-byte display RAM
- Programmable scan timing

* The 4 major sections of 8279 are keyboard, scan, display & CPU interface.

(i) keyboard Interface

- ⇒ The keyboard section consists of eight return lines $RL_0 - RL_7$ that can be used to form the columns of a keyboard matrix.
- ⇒ It has two additional input: shift and control
- ⇒ In the 2 key lockout mode
- ⇒ The keyboard section also have an 8x8 FIFO RAM.

(ii) Display section

- ⇒ It has 8 output lines divided into 2 groups A_0-A_3 & B_0-B_3 .
- ⇒ The cathodes are connected to scan lines through driver transistors.
- ⇒ The display can be blanked by $RD(\text{low})$ line.

(iii) Scan Section

- ⇒ The scan section has a scan counter & 4 scan lines SL_0 to SL_3 .
- ⇒ In decoded scan mode, the b/w of scan lines will be similar to a 2 to 4 decoder.
- ⇒ The scan lines are common for keyboard & display.

(iv) CPU interface section

- ⇒ The CPU interface takes care of data transfer b/w 8279 & the processor.
- ⇒ This section has 8 bidirectional data lines DB_0 to DB_7 for data transfer b/w 8279 & CPU.
- ⇒ The control signals $WR(\text{low})$, $RD(\text{low})$, $CS(\text{low})$ & A_0 are used for read/write to 8279.

Q.2 Explain all operation mode of 8255 in brief?

Ans Operation Mode of 8255 are:—

⇒ There are 2 main operational modes of 8255—

- (i) Input / Output mode
- (ii) Bitset / reset (BSR) mode

I/O mode again classified into 3 types:-

(a) Mode 0 : Simple I/O

→ This functional configuration provides simple input & output operations for each of three ports.

→ No 'handshaking' is required, data is simply written to or read from a specified port.

(b) Mode 1 : I/O with handshake

→ In mode 1; handshake signals are exchanged b/w the MP & peripheral prior to data transfer.

→ Features: Two ports (A & B) functions as 8 Bit I/O ports. They can be configured either as i/p or o/p ports.

(c) Mode 2 : Bidirectional operation

→ This functional configuration provides a means for communicating with a peripheral device or structure on a single 8 Bit bus for both transmitting & receiving data.

→ Interrupt generation & enable/disable functions are also available.

(d) BSR mode:-

→ The BSR mode is concerned only with the 8 Bit of port C, which can be set or reset by writing an appropriate

control word in the control register.

- A control word with bit $D_7 = 0$ is recognized as a BSR control word.
- It doesn't alter any previously transmitted control word with bit $D_7 = 1$.
- The I/O operation of ports A & B are not affected by a BSR control word.

Q.3 Explain the control word of 8254 PIT with example?

Ans - The format of control word of 8254 is:

SC_1	SC_0	RW_1	RW_0	M_2	M_1	M_0	BCD/Binary
--------	--------	--------	--------	-------	-------	-------	------------

Here by using the value of SC_1 & SC_0 we select a specific counter:

SC_1	SC_0	Selection
0	0	C_0
0	1	C_1
1	0	C_2
1	1	Read back status

The value of RW_1 & RW_0 are used to decide the read-write operation

RW_1	RW_0	Selection
0	0	Counter latch command
0	1	R/W lower byte
1	0	R/W higher byte
1	1	R/W lower byte followed by higher byte

The value of M_2, M_1 & M_0 are used to decide the operating mode of 8254

M_2	M_1	M_0	Operation Mode
0	0	0	Mode 0
0	0	1	Mode 1
X(0/1)	1	0	Mode 2
X(0/1)	1	1	Mode 3
1	0	0	Mode 4
1	0	1	Mode 5

The LSB of control word is used to select whether the counter is binary or BCD. If the bit is 0, it works as binary counter & if its value is 1 it works as BCD counter.

Q.4 Design a square wave generator with a pulse width of $150 \mu s$

Ans \rightarrow In this continuous square wave with period equal to count is generated.
 \rightarrow The frequency of square wave = frequency of clock divide by count

If count(N) is odd pulse stay high for $(N+1)/2$ & low for $\frac{N-1}{2}$

Given: - pulse width = 150×10^{-6} sec

then

CW=16

LSB=4

 \overline{WR}

CLK

WATE

Out

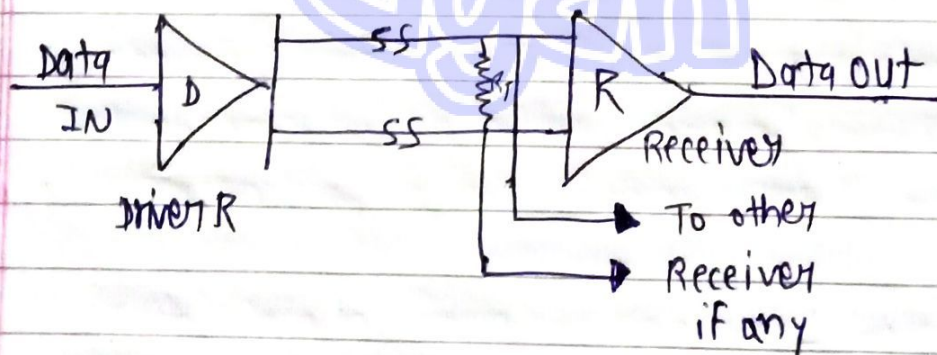
N	N	N	N	N	N	N	N	N	N	N	N	N
---	---	---	---	---	---	---	---	---	---	---	---	---

Er Sahil
Ka
Gyan

Q.1 Write brief technical note on bus standard RS 422A?

Ans -

RS 422A:- High speed data transmission b/w computer system components & peripheral over very long distance, under high noise condition is very difficult with single ended drivers and receivers as an improvement over single ended driver and receivers. Standard achieves transmission rates from 100kbs to 100Mbps. Standard introduced RS 422A which uses low impedance signals. RS-422A is used for balanced transmission. It supports transmission distance which is greater than 1000m. RS 422A is a low impedance signal which is differential signal.



RS-422A
Interface
Circuit

Q.2 Explain interfacing of 20 key matrix keyboard using port B & port C of PPI 8255 with help of schematic diagram.

Q - Interfacing Matrix keyboard using 8255 -

Fig. shows a matrix keyboard with 20 keys, the keyboard has 5 rows & 4 columns. The 1st 16 keys in a sequence will represent data 0 to F is Hex and the remain 4 will represent function such as store & execute.

The circuit shows that the rows are connected to port C & column are connected to port B.

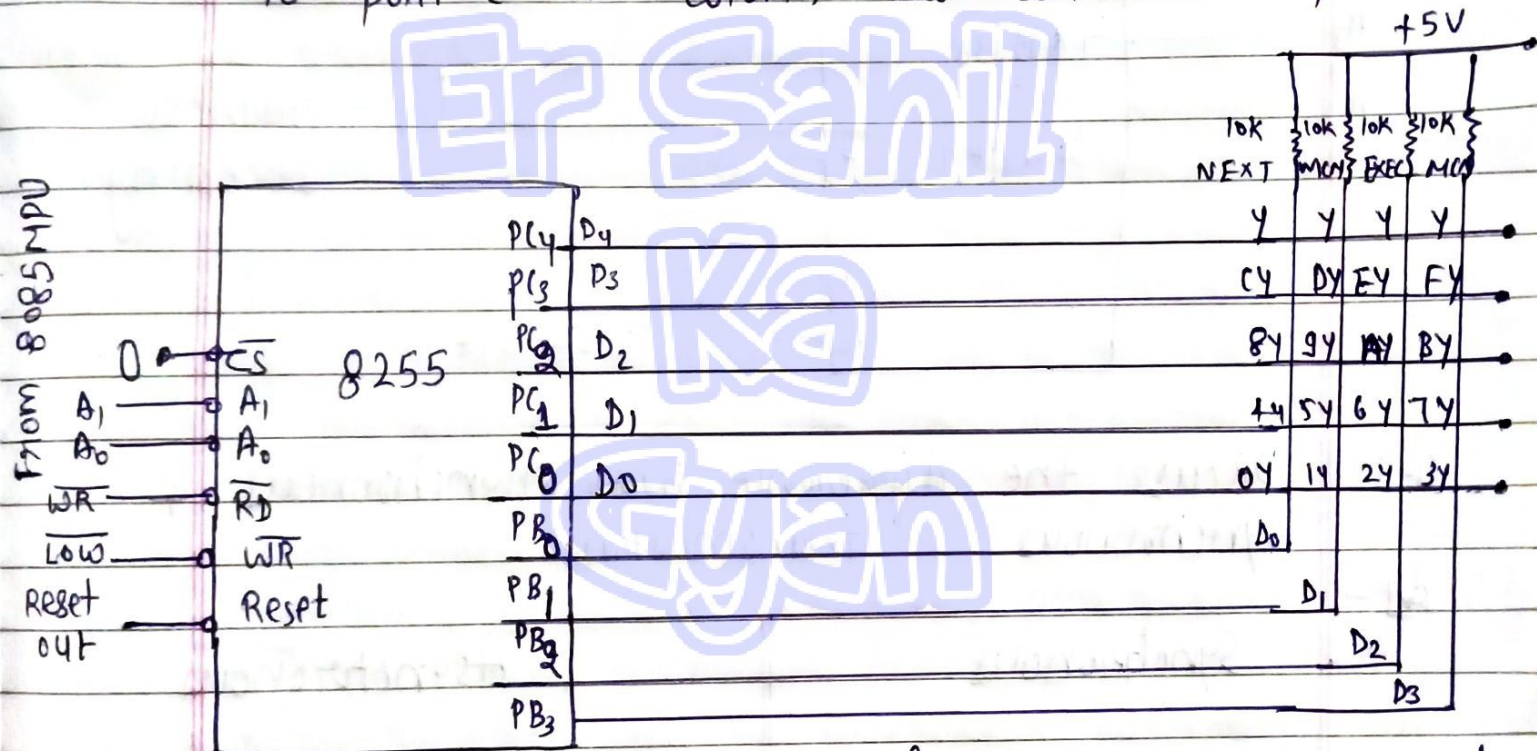


Fig:- Interfacing a Matrix keyboard

In this fig, the column & rows make contact only when key is passed, otherwise they remain high (+5V). When a key is passed, the key must be identified by column & row & the intersection of column & row must go high to low. This can be accomplished as explained in following:-

(i) Ground all rows by sending logic 0 through output port.

Crash b/w
data

Time
Interval

Implemented
by

Eg -

Does not exist

constant

H/w & s/w

Chatrooms,
video conferencing,
Telephonic conversation,
etcetera

Exist

Random

H/w only

Letters, emails,
forums, etcetera.

Q.4

Ans -

Discuss about MPU & explain the features of MPU?

A MPU is a device that implements the elements of computer system on a single integrated circuit or as a few integrated circuits operating as a cohesive unit, designed for processing digital data.

Modern MPU typically incorporate the functionality of a clock, control processing unit, ALU, Floating point unit (FPU), CU, memory management, interrupts, I/O interfaces and cache.

MPU are instruction set processor (ISPs) meaning they operate on a predefined set of instruction. In broadcast, their basic functionality is to continuously send in digital data consisting of instruction & possibly times.

The features of MPU →

- (i) Memory
- (ii) Decision making power based on previously

entered values

- (3.) Repeatability of reading
- (4.) Digital read out & interactiveness
- (5.) Parallel processing
- (6.) Time sharing & multiprocessing
- (7.) Data storage, retrieval & transmission
- (8.) Effective control of multiple equipment in time sharing basis.
- (9.) MPU are extremely used where a lot of processing is required.

Er Sahil
Ka
Gyan