MODULE-3

SYLLABUS

IO and memory interfacing – Address decoding– interrupt structure of 8085. I/O ports- Programmable peripheral interface PPI 8255 - Modes of operation. Interfacing of LEDs, ADC and DAC with 8085

3.1 I/O INTERFACING

- Interfacing is the process of connecting devices together such that they can exchange information.
- There are various communication devices like keyboard, printer etc. So we need to interface these devices with microprocessor by using latches and buffers. This type of interfacing is known as I/O interfacing.

I/O Interfacing Techniques:

I/O devices can be interfaced with microprocessor by two ways:

- Memory mapped I/O and
- I/O mapped I/O or isolated I/O.

Memory mapped IO

- With memory mapped I/O there is a **single address space** for memory and I/O devices as shown in figure.
- The processor treats the I/O devices just like memory locations and **uses same instructions to access both memory and I/O devices**.
- **Single read line and single write line** are required on the bus to access both memory and I/O devices.

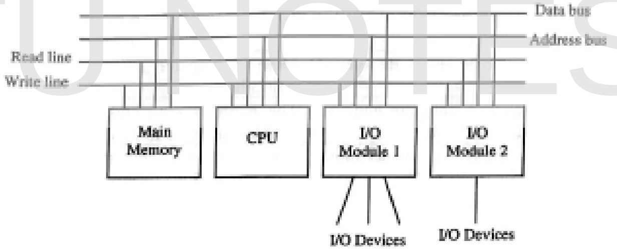


Figure 3.2: Structure of Memory Mapped I/O

I/O mapped I/O or isolated I/O:

- With I/O mapped I/O, the memory and I/O devices are **addressed separately**.
- Processor **uses different instructions for memory and I/O devices.** (ie, MOV for memory and IN,OUT for I/O)
- **Different read and write control signals are used for memory and I/O devices** as shown in the figure below.

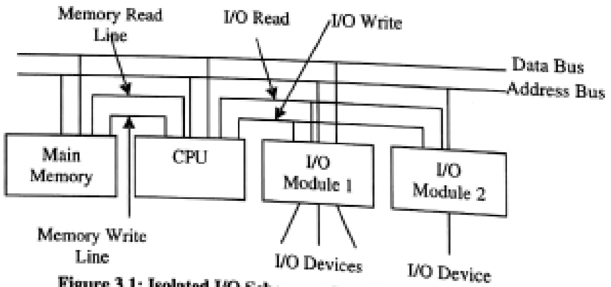


Figure 3.1: Isolated I/O Scheme or I/O Mapped I/O

3.2 MEMORY INTERFACING

- The process of interconnecting memory with microprocessor through buses and other hardware components is known as **memory interfacing.**

Address Decoding:

Address decoding is the way by which microprocessor decodes an address to select a memory location among the total available memory locations. Two types of address decoding techniques are there.

- Absolute Decoding or Full Decoding
- Partial Decoding or Linear Decoding

Absolute Decoding or Full Decoding:

- The decoding in which **all available address lines are used** for decoding to select a location is called absolute decoding technique.
- The figure below shows the absolute decoding technique.

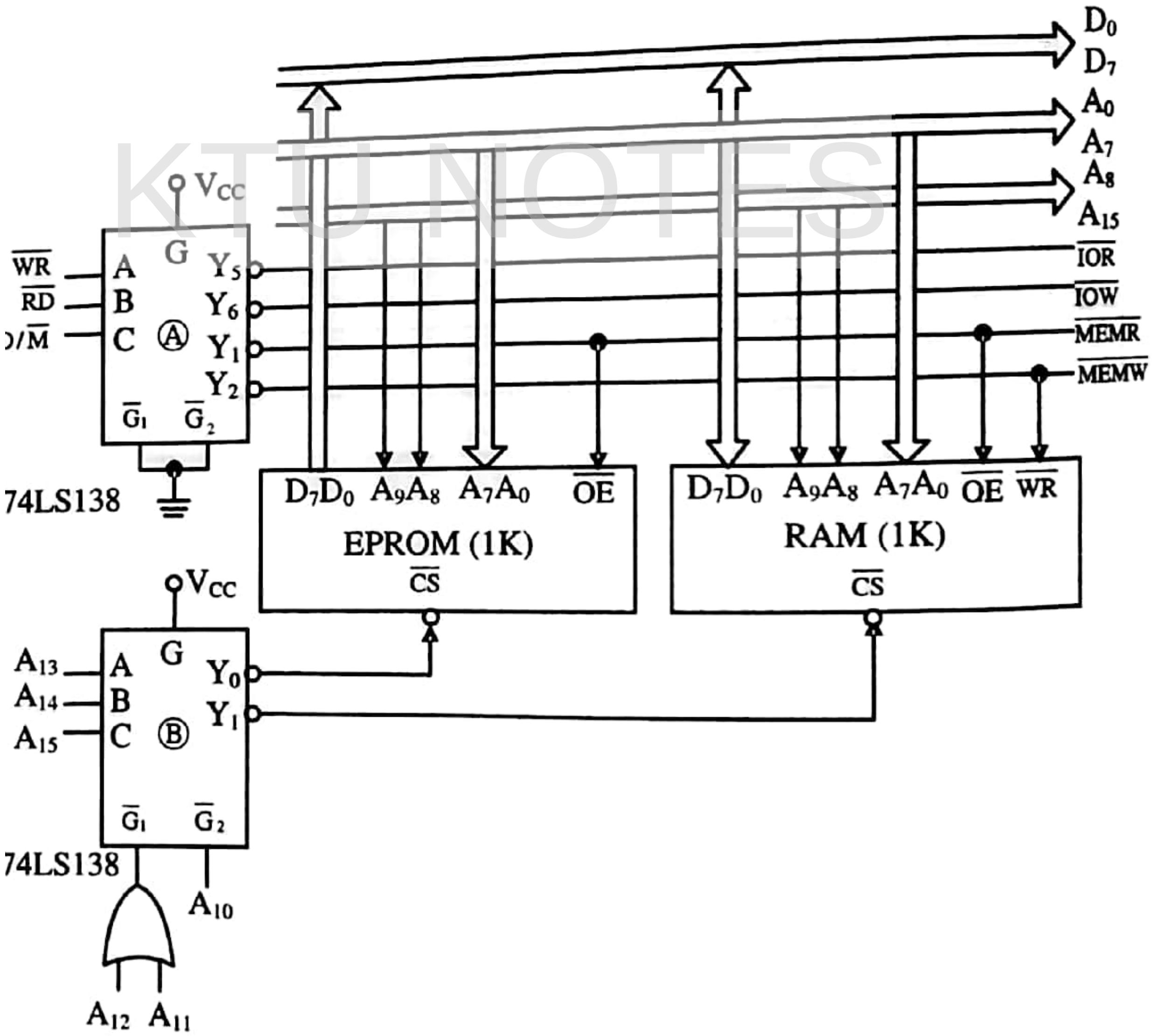


Figure 3.6: Absolute Decoding Technique

- This decoding technique is normally used in large memory systems.
- The memory map for absolute decoding is shown below.

Table 3.2: Memory Map

Memory ICs	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	Address
Starting address of EPROM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000H
End address of EPROM	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	03FFH
Starting address of RAM	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	2000H
End address of RAM	0	0	1	0	0	0	1	1	1	1	1	1	1	1	1	1	23FFH

Partial Decoding or Linear Decoding:

- The decoding in which **all available address lines are not used** for decoding to select a location is called partial decoding technique.
- The figure below shows the partial decoding technique.

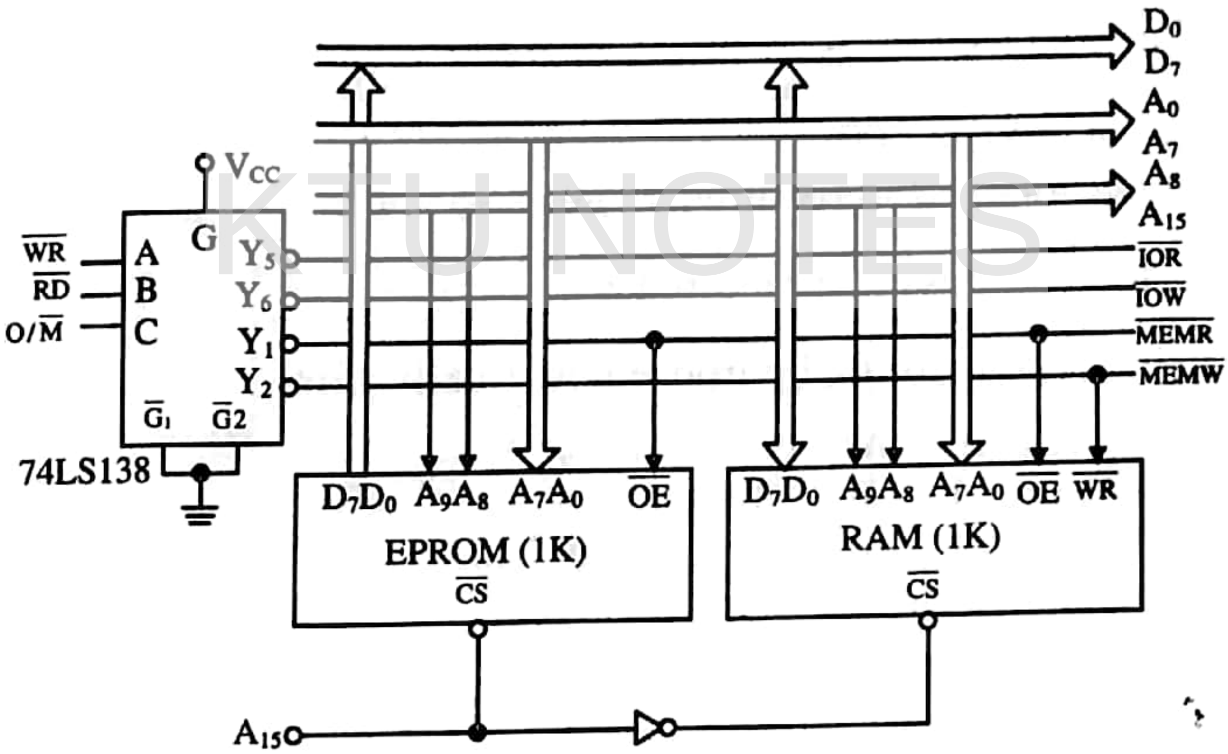
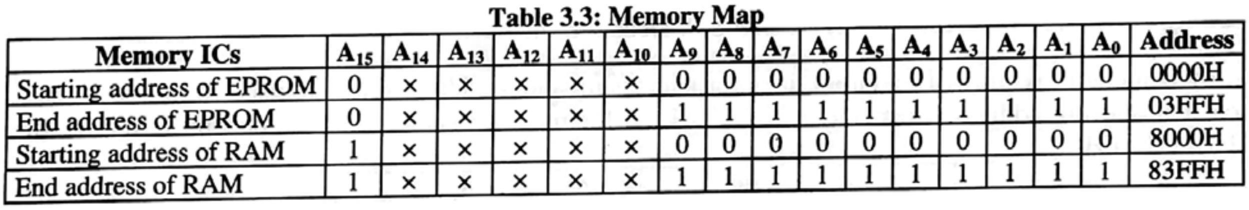


Figure 3.7: Linear Decoding

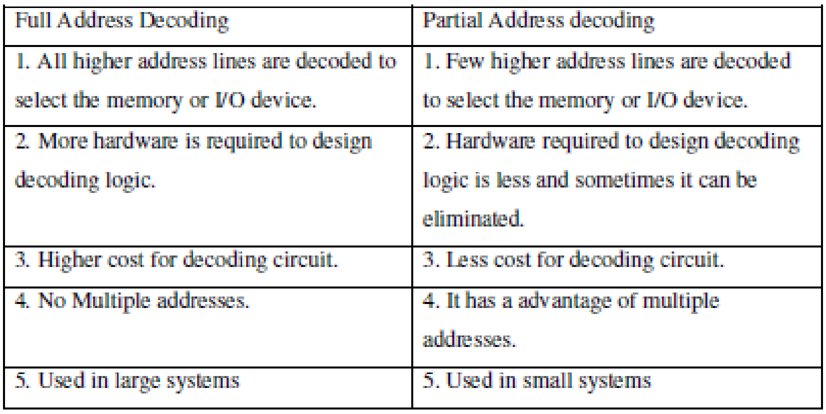
- In small systems, the hardware for decoding circuit can be eliminated by using individual high order address lines to select memory chips. This is the partial decoding technique.
- The cost of the decoding circuit can be reduced with this technique.
- Drawback is the possibility of having multiple addresses.
- The memory map for partial decoding technique is shown below.

Table 3.3: Memory Map



Memory ICs	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	Address
Starting address of EPROM	0	×	×	×	×	×	0	0	0	0	0	0	0	0	0	0	0000H
End address of EPROM	0	×	×	×	×	×	1	1	1	1	1	1	1	1	1	1	03FFH
Starting address of RAM	1	×	×	×	×	×	0	0	0	0	0	0	0	0	0	0	8000H
End address of RAM	1	×	×	×	×	×	1	1	1	1	1	1	1	1	1	1	83FFH

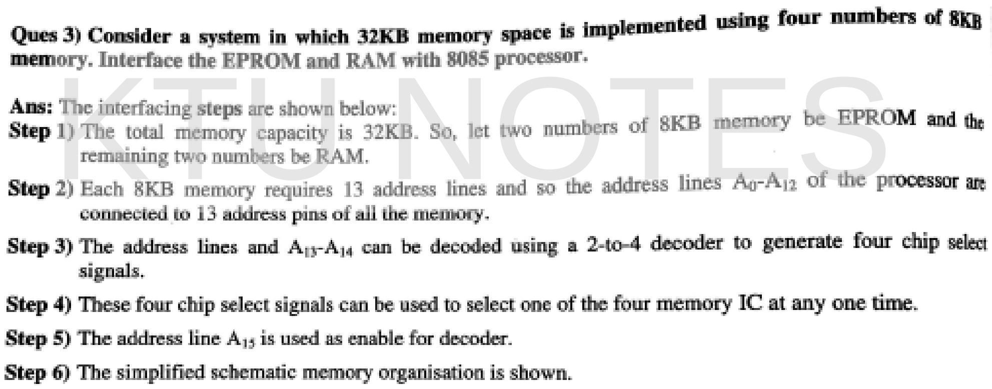
Comparison between Absolute and Partial Decoding techniques.



Full Address Decoding	Partial Address decoding
1. All higher address lines are decoded to select the memory or I/O device.	1. Few higher address lines are decoded to select the memory or I/O device.
2. More hardware is required to design decoding logic.	2. Hardware required to design decoding logic is less and sometimes it can be eliminated.
3. Higher cost for decoding circuit.	3. Less cost for decoding circuit.
4. No Multiple addresses.	4. It has a advantage of multiple addresses.
5. Used in large systems	5. Used in small systems

- UNIVERSITY MODEL QUESTION -

Memory Interfacing Problems: (For more problems, refer class notes)



Ques 3) Consider a system in which 32KB memory space is implemented using four numbers of 8KB memory. Interface the EPROM and RAM with 8085 processor.

Ans: The interfacing steps are shown below:

Step 1) The total memory capacity is 32KB. So, let two numbers of 8KB memory be EPROM and the remaining two numbers be RAM.

Step 2) Each 8KB memory requires 13 address lines and so the address lines A_0-A_{12} of the processor are connected to 13 address pins of all the memory.

Step 3) The address lines and A_{13}-A_{14} can be decoded using a 2-to-4 decoder to generate four chip select signals.

Step 4) These four chip select signals can be used to select one of the four memory IC at any one time.

Step 5) The address line A_{15} is used as enable for decoder.

Step 6) The simplified schematic memory organisation is shown.

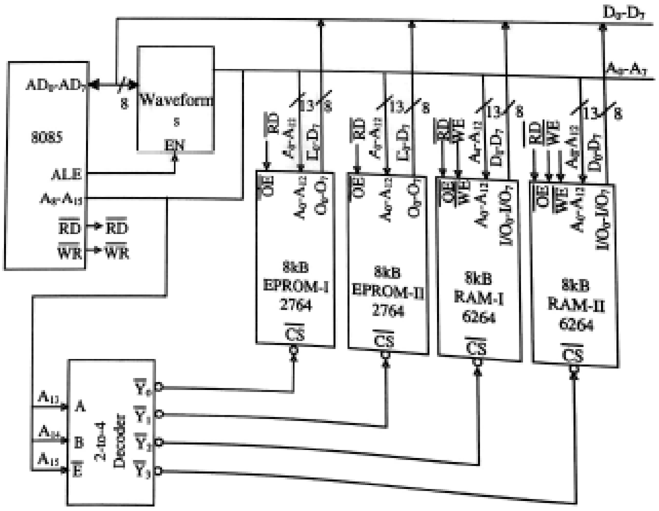


Figure 3.4: Interfacing 16KB EPROM and 16KB RAM with 8085

Ques 4) Consider a system in which the 64KB memory space is implemented using eight numbers of 8 KB memories. Interface the EPROM and RAM with 8085 processor.

Ans: The interfacing **steps** are shown below:

Step 1) The total memory capacity is 64KB. So, let 4 numbers of 8KB EPROM and 4 numbers of 8KB RAM.

Step 2) Each 8KB memory requires 13 address lines. So the address line A_0-A_{12} of the processor are connected to 13address pins of all the memory ICs.

Step 3) The address lines A_{13}, A_{14} and A_{15} are decoded using a 3-to-8 coder to generate eight chip select signals. These eight chip select signals can be used to select one of the eight memories at any one time.

Step 4) The memory interfacing is shown in following **figure 3.5**:

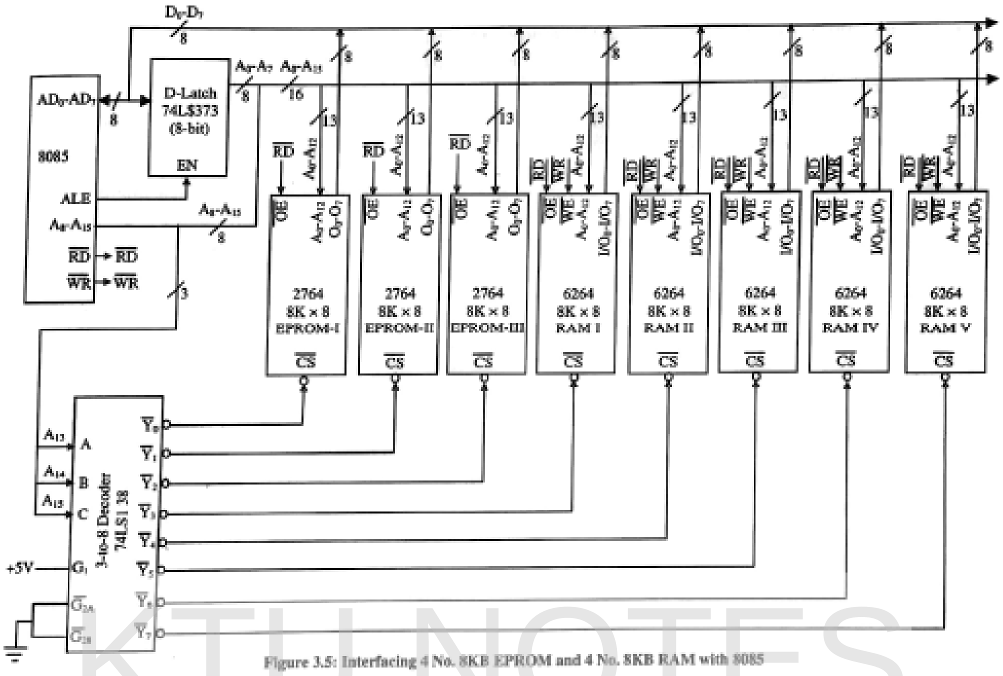


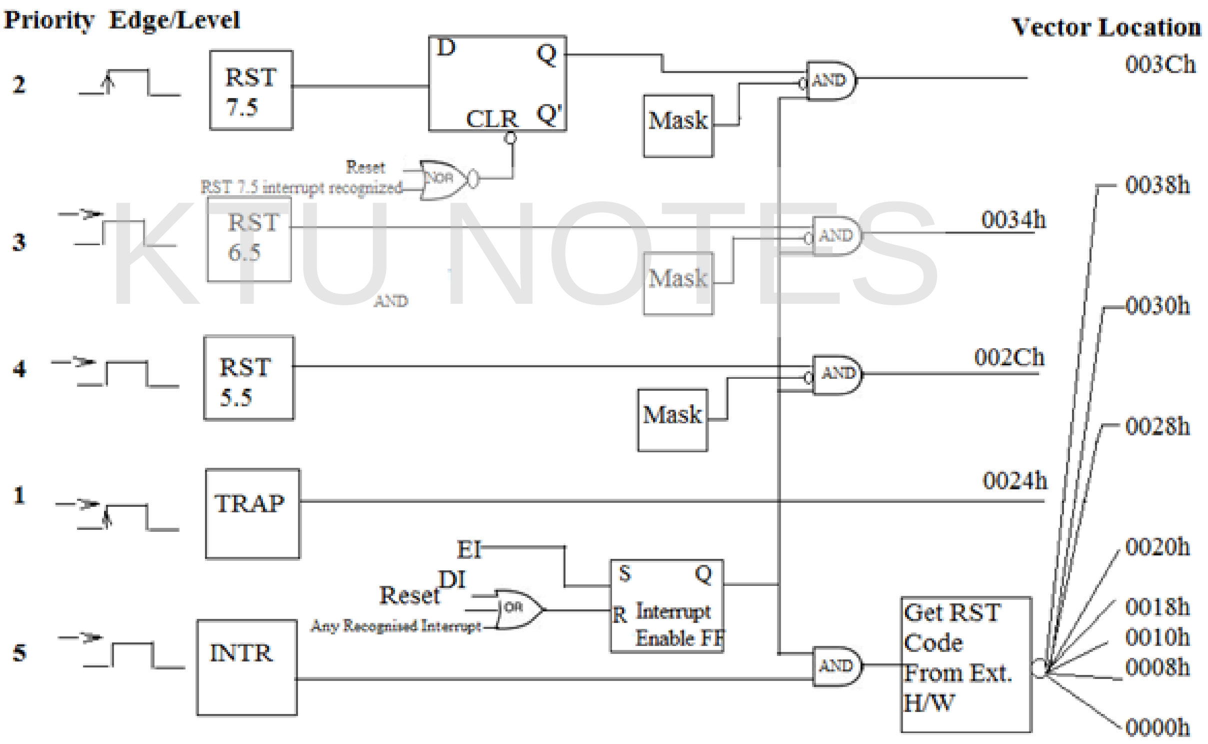
Figure 3.5: Interfacing 4 No. 8KB EPROM and 4 No. 8KB RAM with 8085

3.3 Interrupt Structure in 8085

- Interrupt is signal send by an external device to the processor, to request the processor to perform a particular task or work.

Interrupt Process in 8085:

- The processor will check the interrupts always at the last machine cycle of each instruction.
- If there is any interrupt, it accepts the interrupt and sends the $\overline{INTA}$ signal to the peripheral.
- The current value of program counter will be saved to stack and starting address (vector address) of interrupt service routine (ISR) will be copied to program counter.
- The processor executes the interrupt service routine (ISR).
- It returned to main program by last instruction of ISR (RET) after retrieving the address stored at the stack.



Interrupt Structure of 8085

Types of Interrupts:

It supports two types of interrupts.

- **Hardware**
- **Software**

Software interrupts:

- The software interrupts are program instructions. These instructions are inserted at desired locations in a program. While running a program, If a software interrupt instruction is encountered, then the processor executes an interrupt service routine (ISR)

- The 8085 has eight software interrupts from RST 0 to RST 7. The vector address for these interrupts can be calculated as follows.
- Interrupt number * 8 = vector address
- For eg:- for RST1, 1 * 8 = 08 = 08H So, Vector address for interrupt RST 5 is 0008H.

Hardware interrupts:

- An external device initiates the hardware interrupts by applying an appropriate signal at the interrupt pin of the processor.
- If the interrupt is accepted then the processor executes an interrupt service routine.
- The 8085 has five hardware interrupts

  (1) TRAP (2) RST 7.5 (3) RST 6.5 (4) RST 5.5 (5) INTR

1. **TRAP:**

- This interrupt is a non-maskable interrupt. It is unaffected by any mask or interrupt enable.
- TRAP is a vectored interrupt.
- TRAP bas the highest priority.
- TRAP interrupt is edge and level triggered.

2. **RST 7.5:**

- The RST 7.5 interrupt is a maskable interrupt.
- It has the second highest priority.
- It is edge sensitive.
- Maskable interrupt. It can be disabled by DI instruction.
- Enabled by EI instruction.

3. **RST 6.5 and 5.5:**

- The RST 6.5 and RST 5.5 both are level triggered.
- RST 6.5 and RST 5.5 are maskable interrupts. These can be disabled by DI instruction.
- The RST 6.5 has the third priority whereas RST 5.5 has the fourth priority.

4. **INTR:**

- INTR is a maskable interrupt. It can be disabled by DI instruction.
- It can be enabled by EI instruction.
- Non- vectored interrupt. After receiving $\overline{\text{INTA}}$ signal, it has to supply the address of ISR.
- It has lowest priority.
- It is a level sensitive interrupts.

The following sequence of events occurs when INTR signal goes high.

1. The 8085 checks the status of INTR signal during execution of each instruction.
2. If INTR signal is high, then 8085 complete its current instruction and sends an interrupt acknowledge signal $\overline{\text{INTA}}$ to the device that interrupted.
3. In response to the acknowledge signal, interrupted device places an RST instruction on the data bus.
4. On receiving the instruction, the 8085 save the address of next instruction on stack and execute received instruction.
5. Then the microprocessor executes the interrupt service routine.

- -

Priority of Interrupts:

The 8085 microprocessor has five interrupt inputs. They are TRAP, RST 7.5, RST 6.5, RST 5.5, and INTR. These interrupts have a fixed priority of interrupt service. If two or more interrupts go high at the same time, the 8085 will service them on priority basis. The TRAP has the highest priority followed by RST 7.5, RST 6.5, RST 5.5 and INTR. The priority of interrupts in 8085 is shown in the table.

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

- -

Maskable and Non-Maskable Interrupts:

Maskable Interrupt:

An Interrupt that can be disabled or ignored by the instructions of CPU is called as Maskable Interrupt.

Eg: RST 7.5, RST 6.5, RST 5.5 and INTR are maskable Interrupts.

Non-Maskable Interrupt: An interrupt that cannot be disabled or ignored by the instructions of CPU is called as Non-Maskable Interrupt.

Eg:TRAP interruptof 8085

- -

Vectored and Non-Vectored Interrupts:

Vectored interrupt
In vectored interrupts, the processor automatically branches to the specific address in response to an interrupt.
Eg: TRAP, RST 7.5, RST 6.5, RST 5.5

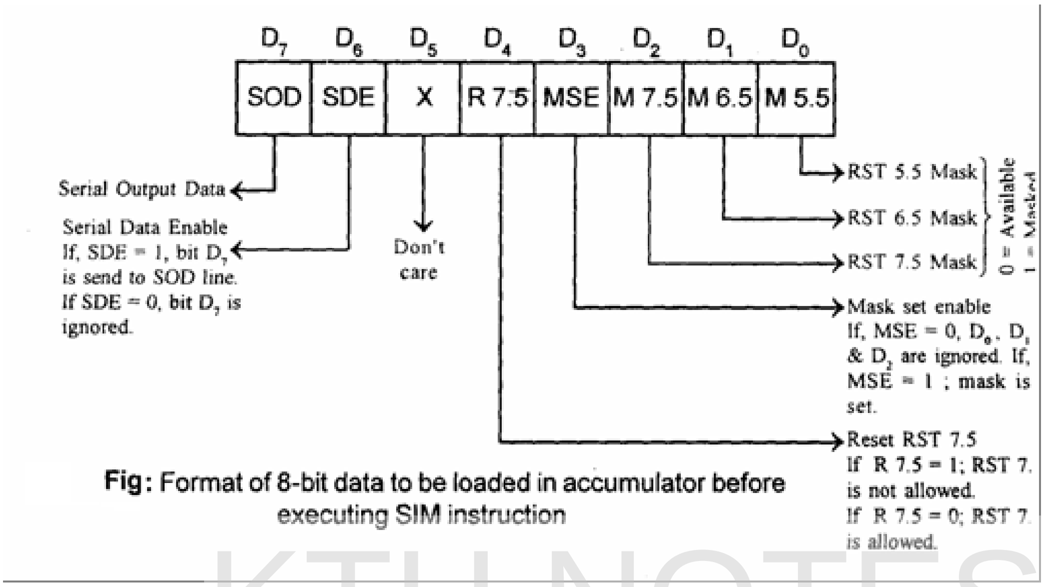
Non-vectored interrupt
But in non -vectored interrupts the interrupted device should give the address of the interrupt service routine (ISR).
Eg: INTR

- -

- -

SIM and RIM for interrupts:

- The 8085 provide additional masking facility for RST 7.5, RST 6.5 and RST 5.5 using SIM instruction.
- The status of these interrupts can be read by executing RIM instruction.
- The masking or unmasking of RST 7.5, RST 6.5 and RST 5.5 interrupts can be performed by moving an 8-bit data to accumulator and then executing SIM instruction.
- The format of the 8-bit data is shown below.

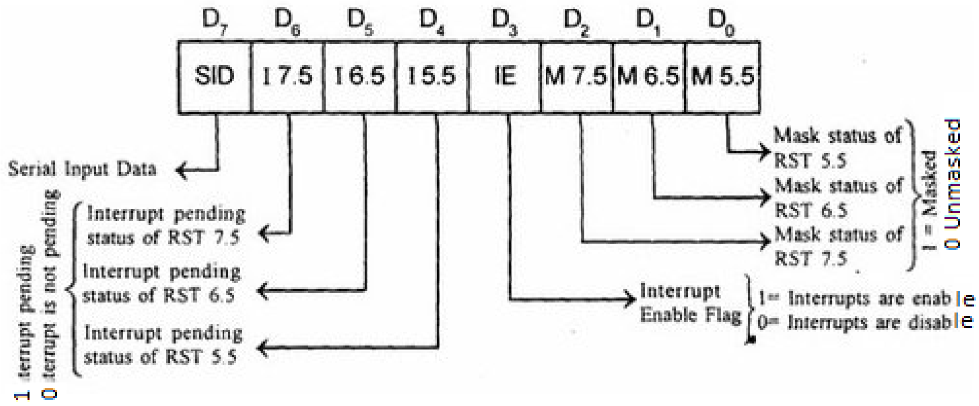


D7 D6 D5 D4 D3 D2 D1 D0
SOD SDE X R 7.5 MSE M 7.5 M 6.5 M 5.5
Serial Output Data
Serial Data Enable
If, SDE = 1, bit D7 is send to SOD line.
If SDE = 0, bit D7 is ignored.
Don't care
RST 5.5 Mask
RST 6.5 Mask
RST 7.5 Mask
0 = Available
1 = Masked
Mask set enable
If, MSE = 0, D0, D1 & D2 are ignored. If, MSE = 1 ; mask is set.
Reset RST 7.5
If R 7.5 = 1; RST 7. is not allowed.
If R 7.5 = 0; RST 7. is allowed.

Fig: Format of 8-bit data to be loaded in accumulator before executing SIM instruction

- The status of pending interrupts can be read from accumulator after executing RIM instruction.
- When RIM instruction is executed an 8-bit data is loaded in accumulator, which can be interpreted as shown in fig.

RIM Instruction



D7 D6 D5 D4 D3 D2 D1 D0
SID I 7.5 I 6.5 I 5.5 IE M 7.5 M 6.5 M 5.5
Serial Input Data
Interrupt pending status of RST 7.5
Interrupt pending status of RST 6.5
Interrupt pending status of RST 5.5
1 nterrupt pending
0 nterrupt is not pending
Mask status of RST 5.5
Mask status of RST 6.5
Mask status of RST 7.5
1 = Masked
0 Unmasked
Interrupt Enable Flag
1= Interrupts are enable
0= Interrupts are disable

- -

3.4 PERIPHERAL IC INTERFACING

1. PROGRAMMABLE PERIPHERAL INTERFACE (PPI)-8255

Features:

- It is a programmable device.
- It has 24 I/O programmable pins.

Function of pins:

- Data bus(D_0-D_7):These are 8-bit bi-directional buses, connected to 8086 data bus for transferring data.
- CS: This is Active Low signal. When it is low, then data is transfer from 8085.
- Read: This is Active Low signal, when it is Low read operation will be start.
- Write: This is Active Low signal, when it is Low Write operation will be start.
- Address (A_0-A_1):This is used to select the ports. like this

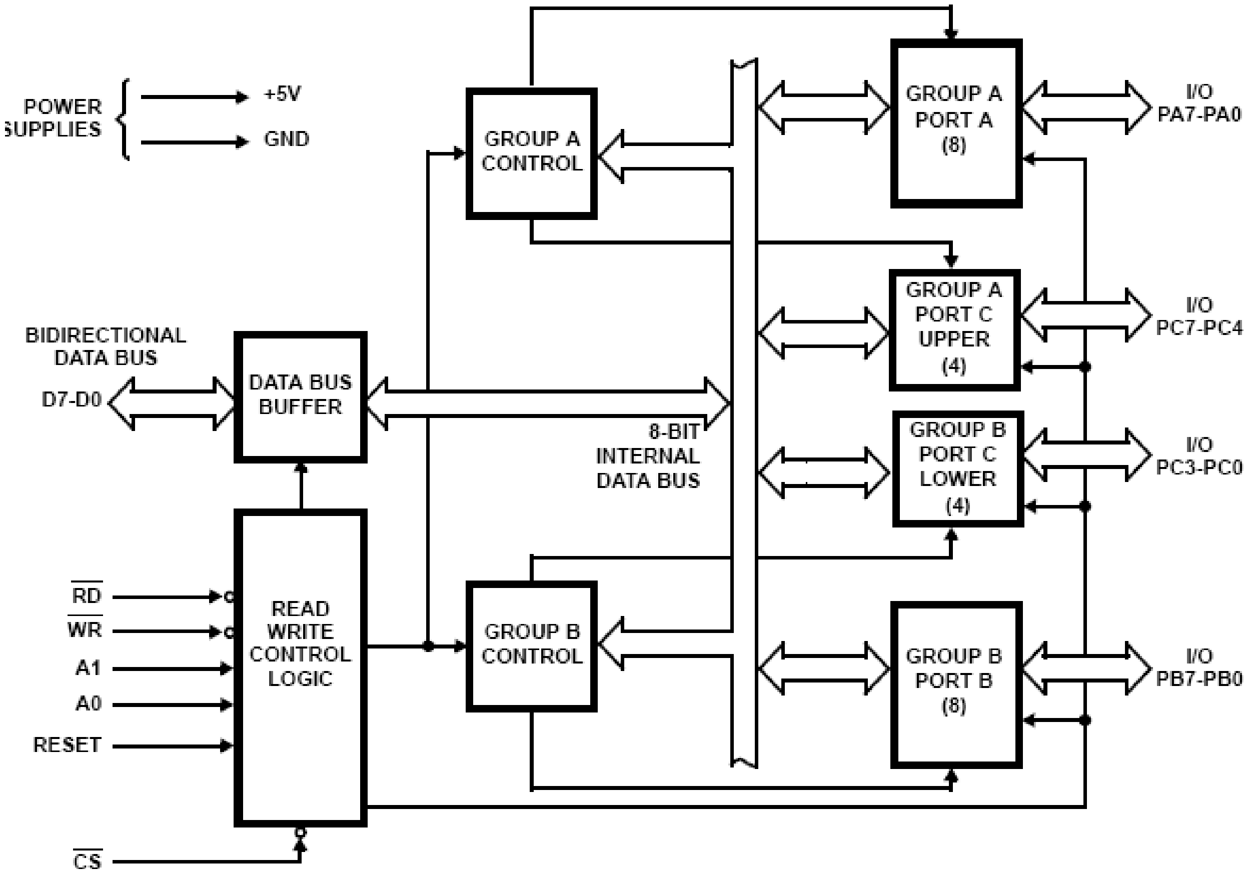
A1	A0	Select
0	0	PA
0	1	PB
1	0	PC
1	1	Control reg.

- RESET: This is used to reset the device. That means clear control registers.
- PA_0-PA_7:It is the 8-bit bi-directional I/O pins used to send the data to peripheral or to receive the data from peripheral.
- PB_0-PB_7:Similar to PA
- PC_0-PC_7:This is also 8-bit bidirectional I/O pins. These lines are divided into two groups.

1. PC_0 to PC_3(Lower Groups)
2. PC_4 to PC_7 (Higher groups)

These two groups can be worked separately.

Block Diagram:



Data Bus buffer:

- It is a 8-bit bidirectional Data bus.
- Used to interface between 8255 data bus with system bus.
- The internal data bus and Outer pins D_0-D_7 pins are connected in internally.
- The direction of data buffer is decided by Read/Control Logic.

Read/Write Control Logic:

- This is getting the input signals from control bus and Address bus
- Control signal are RD and WR.
- Address signals are A0,A1,and CS.
- 8255 operation is enabled or disabled by CS.

Group A and Group B control:

- Group A and B get the Control Signal from CPU and send the command to the individual control blocks.
- Group A send the control signal to port A and Port C (Upper) PC7-PC4.
- Group B send the control signal to port B and Port C (Lower) PC3-PC0.

PORT A:

- This is a 8-bit buffered I/O latch.
- It can be programmed by mode 0 , mode 1, mode 2 .

PORT B:

- This is a 8-bit buffer I/O latch.
- It can be programmed by mode 0 and mode 1.

PORT C:

- This is a 8-bit Unlatched buffer Input and an Output latch.
- It is splitted into two parts, Port C Upper and Port C Lower.
- It can be programmed by bit set/reset operation.

Modes of Operation :

- Basically, there are two modes, BSR mode and I/O mode.

6. BIT SET/RESET MODE:

- Any one of the 8-bits of PORT C can be Set or Reset depending upon the select bits on control word register.

7. I/O MODE:

3 types of I/O modes are there, mode0, mode1 and mode2.

MODE 0(Simple input / Output):

- In this mode , two 8 bit ports(port A and port B) and two 4 bit ports(port C upper and port C lower) are available for I/O operations.
- Features:
- Any port can be used as an input or output port.
- Ports do not have Handshake or interrupt capability.

MODE 1 :(Strobed I/O mode)

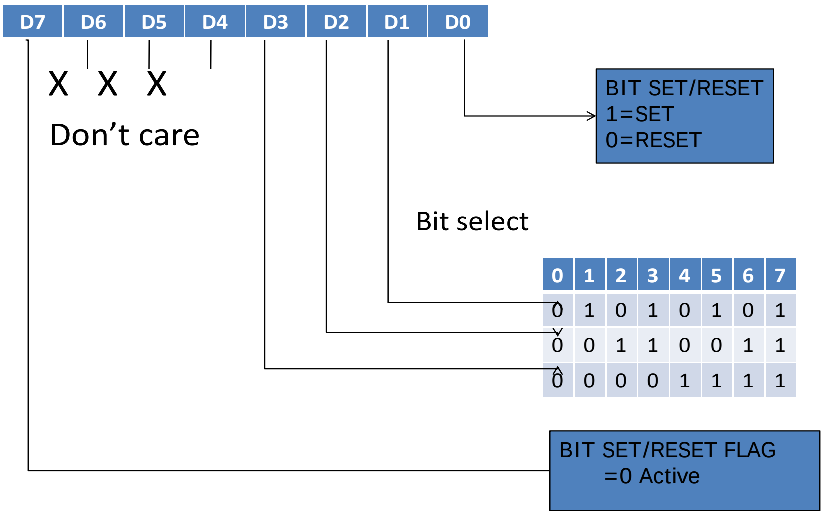
- Two groups-group A and group B are available for strobed data transfer.
- Each group contains one 8-bit data I/O port and one 4 bit control port.
- The 8 bit data port can be either used as input or output port.
- Out of 8 bit port C,PC3-PC5 are used to generate control signals for port A and PC0-PC2 are used to generate control signals for port B. The lines PC6 and PC7 may be used as independent I/O lines.

MODE 2: Strobed bi-directional I/O mode:

- This mode allows bidirectional data transfer over a single 8-bit data bus (port A) using handshake signals.
- Port A is working as 8-bit bidirectional.
- 5 bit control port PC3-PC7 is used for generating/accepting handshaking signals of port A.
- Here, port B and three lines of port C (PC2-PC0) may be used in either simple I/O mode or strobed mode.

Control word register format.

For BSR **mode**:



0	1	2	3	4	5	6	7
0	1	0	1	0	1	0	1
0	0	1	1	0	0	1	1
0	0	0	0	1	1	1	1

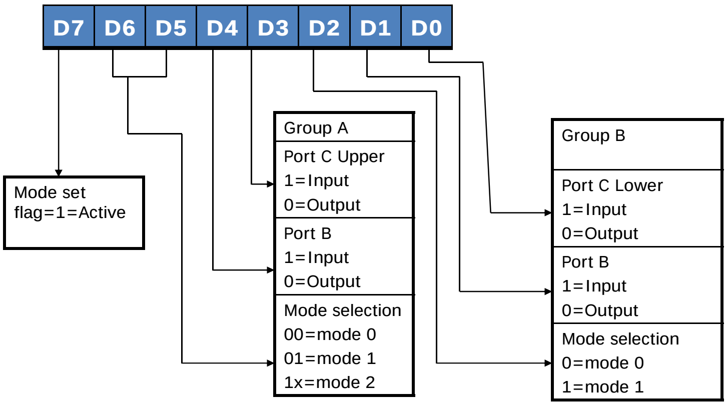
- PC0-PC7 is set or reset as per the status of D0.

Example:

- To Set PC3, control register will be 0XXX0111.
- To reset PC4, control register will be 0XXX01000.
- X is a don't care.

- FOR I/O MODE:

The mode format for I/O as shown in figure



For Example: Write the 8 bit control word to configure 8255 with the following requirement.
PORTA as output port, PORTB as input port and PORTC as input port.

Answer: D7 D6 D5 D4 D3 D2 D1 D0
1 0 0 0 1 0 1 1

- UNIVERSITY QUESTION -

What are handshaking signals?

Handshaking signals are dedicated signals to coordinate data transfer between two devices with different speed. The following are the commonly used handshaking signals:

$\overline{\text{STB}}$ (Strobe Input) :

This signal is generated by a peripheral device that it has transmitted a byte of data.

IBF (Input buffer full) :

This signal is an acknowledgement by the 8255 to indicate that the input latch has received the data byte.

$\overline{\text{OBF}}$(Output Buffer Full) :

This is an output signal that goes low when the microprocessor writes data into the output latch of the 8255.

$\overline{\text{ACK}}$ (Acknowledge) :

This is an input signal from a peripheral that must output a low when the peripheral receives the data from the 8255 ports.

- -

3.5 INTERFACING LED WITH 8085

- LEDs are interfaced with 8085 using 8255 PPI.
- 8255 PPI is interfaced using address lines, data lines and control line RD and WR.
- LEDs are connected to any one of the port by configuring the port in simple IO mode.
- An interfacing diagram with 8 LEDs is shown below.
- LEDs are connected to the ground through current limiting resistors.
- When logic 1 is send to any pin of Port A, the LED will conduct the current and it will glow.
- When logic 0 is send to any pin of Port A, then the LEDs will be off.

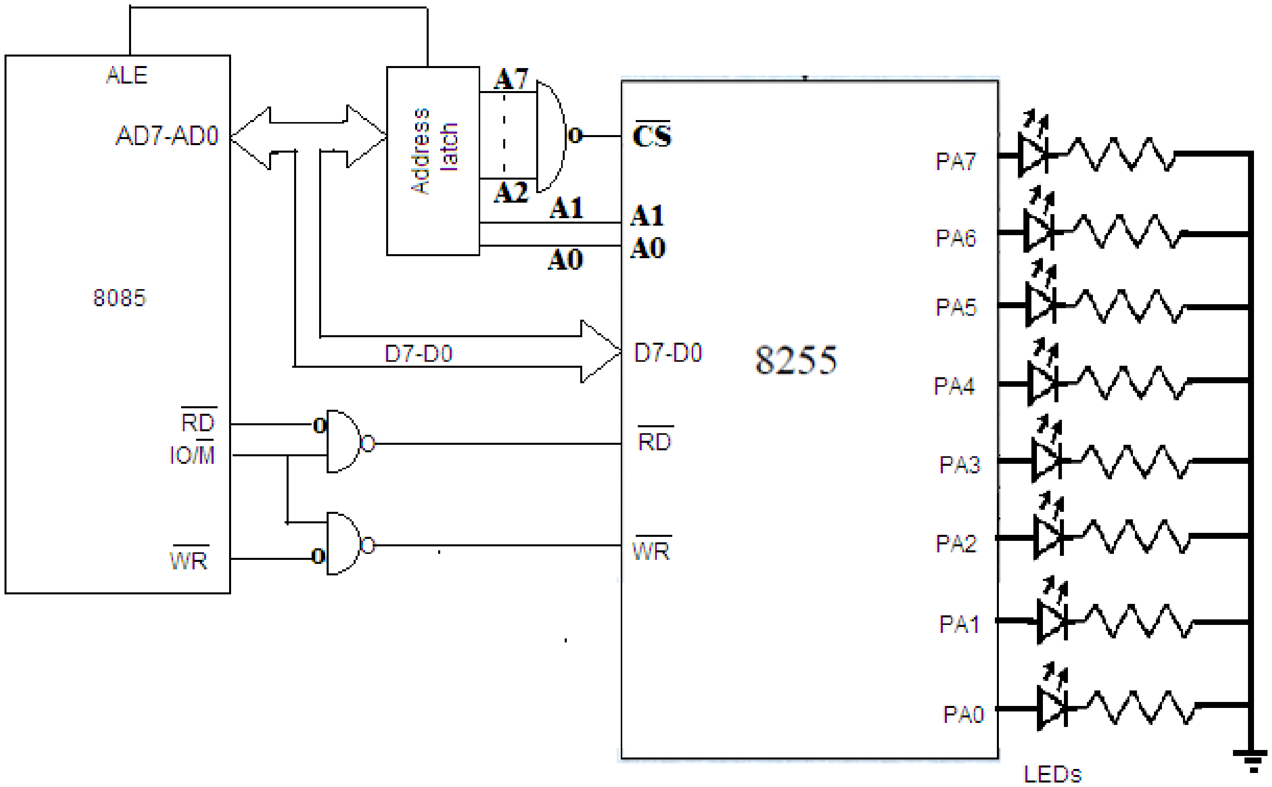


Figure: Interfacing LEDs with 8085

Example program:- Program to blink LEDs continuously

In order to configure the 8255 with Port A as output port, the control word to be written in Control Word Register should be 10000000 ie, 80H.

The 8255 control word is written as follows:

```
D7 D6 D5 D4 D3 D2 D1 D0
 1  0  0  0  0  0  0  0
```

- The required ALP is as follows:

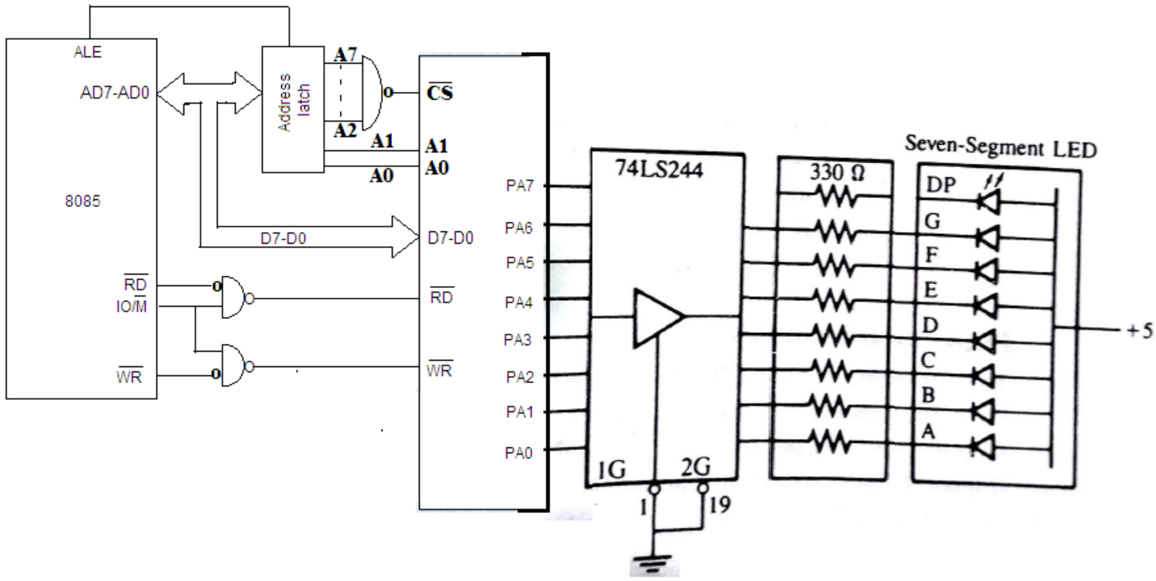
```
Main Program:
        MVI A, 80           ;initialize 8255 as discussed above.
        OUT CWR
BACK:   MVI A, FF           ;To send 1 to all LEDs at port A.
        OUT Port A
        CALL DELAY
        MVI A, 00           ;To send 0 to all LEDs at port A.
        OUT Port A
        CALL DELAY
        JMP BACK

Delay Subroutine:

DELAY:  MVI C, FF
REP:    DCR C
        JNZ REP
        RET
```

- UNIVERSITY QUESTION -

Explain interfacing of seven segment LED display with 8085.



- Seven segment LED display is interfaced with 8085 using 8255 PPI.
- 8255 PPI is interfaced using address lines, data lines and control line RD and WR.
- Seven segment LED display is connected to any one of the port by configuring the port in simple IO mode.
- Seven segment LED display is connected to the ground through 74LS244 driver and current limiting resistors.
- Seven segment LED display used here is a common anod type, hence When logic 0 is send to any pin of Port A, the LED will conduct the current and it will glow.
- When logic 1 is send to any pin of Port A, then the LEDs will be off.

- -

3.6 INTERFACING ANLOG TO DIGITAL CONVERTER (ADC) WITH 8085

- The analog to digital converter is treated as an input device by the microprocessor.
- ADC 0808 is an ADC chip with 8 input channels which is the commonly used one.

During the analog to digital conversion process,

- Initially, microprocessor sends an initializing signal (start of conversion-SOC) to the ADC to start the analog to digital data conversion process. The start of conversion signal is a pulse of a specific duration.
- The microprocessor has to wait for the digital data till the conversion is over. After the conversion is over, the ADC sends end of conversion EOC signal to inform the microprocessor that the conversion is over and the result is ready at the output buffer of the ADC.
- These tasks of issuing an SOC pulse to ADC, reading EOC signal from the ADC and reading the digital output of the ADC are carried out by the CPU using 8255 I/O ports.

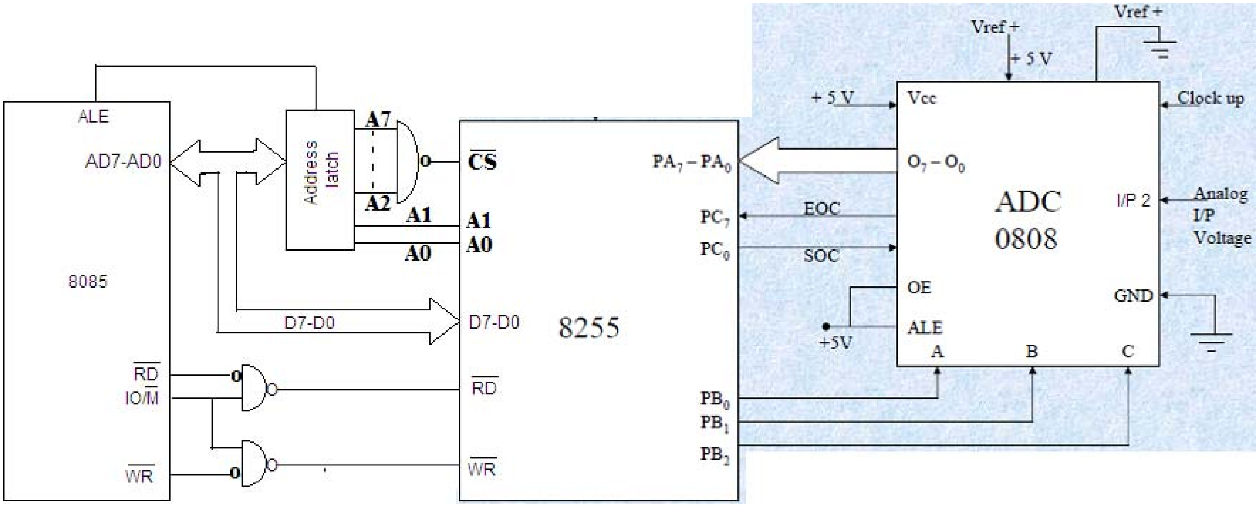


Figure: Interfacing ADC with 8085

Example: Interfacing ADC 0808 with 8085 using 8255 ports. Use port A of 8255 for transferring digital data output of ADC to the CPU and port C for control signals. Assume that an analog input is present at I/P2 of the ADC and a clock input of suitable frequency is available for ADC.

- **Solution**:
  - The analog input I/P2 is used and therefore address pins A,B,C should be 0,1,0 respectively to select I/P2.
  - The OE and ALE pins are already kept at +5V to select the ADC and enable the outputs.
  - Port C upper acts as the input port to receive the EOC signal while port C lower acts as the output port to send SOC to the ADC. Port A acts as a 8-bit input data port to receive the digital data output from the ADC.

The 8255 control word is written as follows:

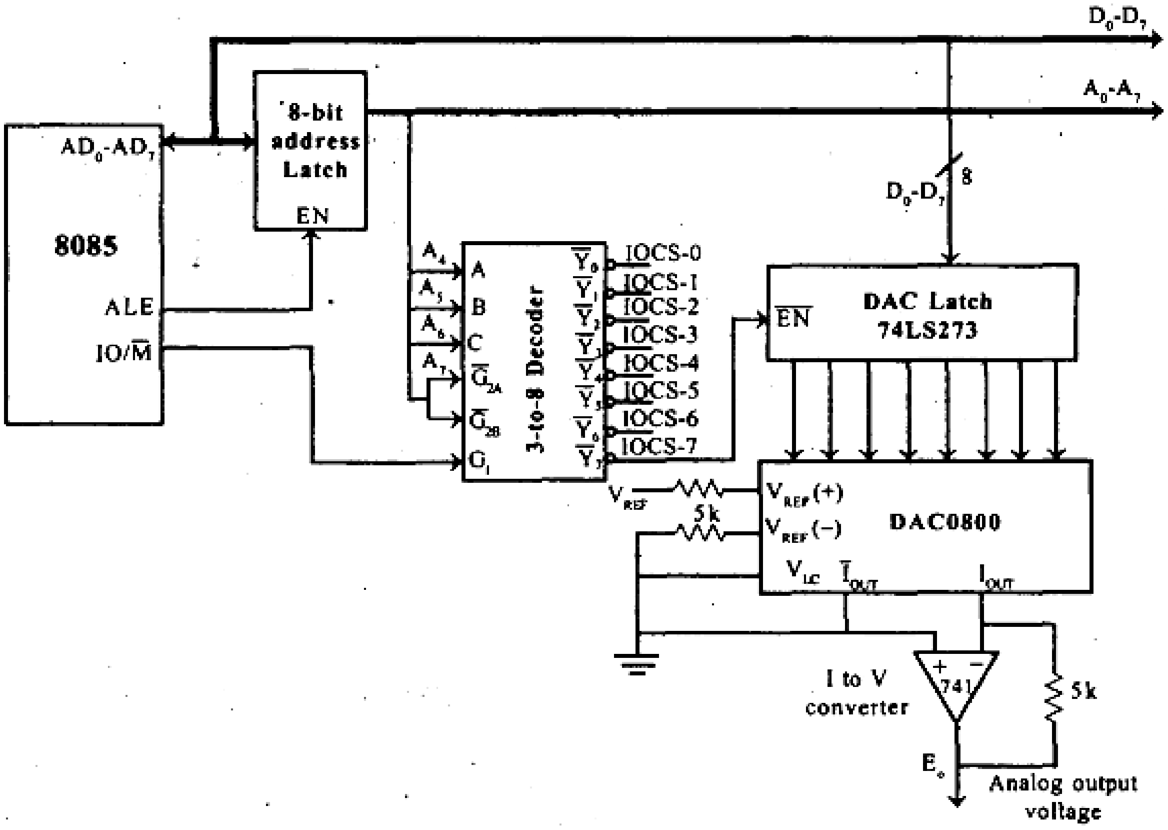
D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	1	1	0	0	0

- The required ALP is as follows:

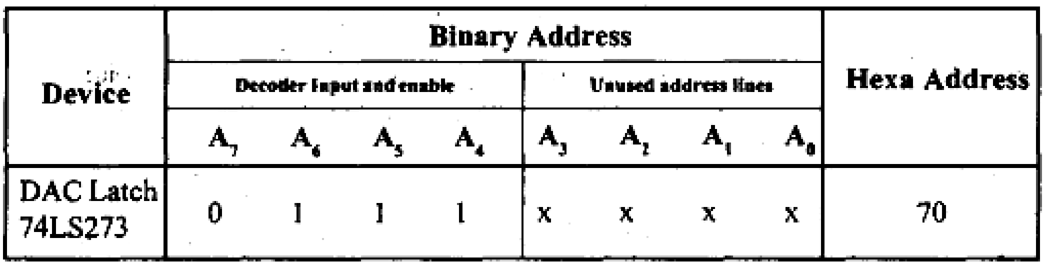
```
         MVI A, 98          ;initialize 8255 as discussed above.
         OUT CWR
         MVI A, 02          ;Select I/P2 as analog input.
         OUT Port B
         MVI A, 00         ;Give start of conversion pulse to the ADC
         OUT Port C
         MVI A, 01
         OUT Port C
         MVI A, 00
         OUT Port C
WAIT:    IN  Port C        ;Check for EOC by reading port C upper and rotating through CY
         RAL
         JNC WAIT
         IN  Port A        ;If EOC, read digital equivalent in AL
         HLT               ;Stop.
```

3.7 INTERFACING DIGITAL TO ANALOG CONVERTER (DAC) WITH 8085

- To convert the digital signal to analog signal a Digital-to-Analog Converter (DAC) has to be employed.
- The DAC will accept a digital (binary) input and convert to analog voltage or current.
- The DAC0800 can be interfaced to 8085 system bus by using an 8-bit latch and the latch can be enabled by using one of the chip select signal generated for I/O devices.
- A simple schematic for interfacing DAC0800 with 8085 is shown below.



- In this schematic the DAC0800 is interfaced using an 8-bit latch 74LS273 to the system bus.
- The 3-to-8 decoder 74LS 138 is used to generate chip select signals for I/O devices.
- The address lines A4, A5 and A6 are used as input to decoder.
- The address line A7 and the control signal IO/M (low) are used as enable for decoder.
- The decoder will generate eight chip select signals and in this the signal IOCS-7 is used as enable for latch of DAC.
- The I/O address of the DAC is shown in table.



Device	Binary Address								Hexa Address
	Decoder Input and enable				Unused address lines				
	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	
DAC Latch 74LS273	0	1	1	1	x	x	x	x	70

- In order to convert a digital data to analog value, the processor has to load the data to latch.
- The latch will hold the previous data until next data is loaded.
- The DAC will take definite time to convert the data. The software should take care of loading successive data only after the conversion time.
- The DAC 0800 produces a current output, which is converted to voltage output using I to V converter.