UNIT-3

8086 INTERFACING

- Semiconductor memories interfacing (RAM, ROM), Intel 8255 programmable peripheral interface, Interfacing switches and LEDs, Interfacing seven segment displays, Intel 8251 USART architecture and interfacing, stepper motor, A/D and D/A Converters.

---

1) Interface two 8K X 8 EPROMs (2764) and two 8K X 8 RAM chips with 8086 using logic gates, such that the memory address ranges from FC000H - FFFFFH and 00000H - 03FFFH respectively.

Soln Let us interface two 8K X 8 EPROMs with 8086 and the address ranges from OFC000H-FFFFFH

Memory addresses assigned to EPROM chips:-

A19	A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	
1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	- FC000H
1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	- FC001H
1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	- FC002H
1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	- FC003H
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	- FFFFFH

→ All the even addresses ie, FC 000H, FC002H, FC004H ... has A0 as '0' which are assigned to one 8K x 8 EPROM chip (2764-A) which acts as even bank.

→ All the odd address ie, FC001, FC003, ... has A0 = 1 which are assigned to another 8K x 8 EPROM chip (2764-B) which acts as odd bank

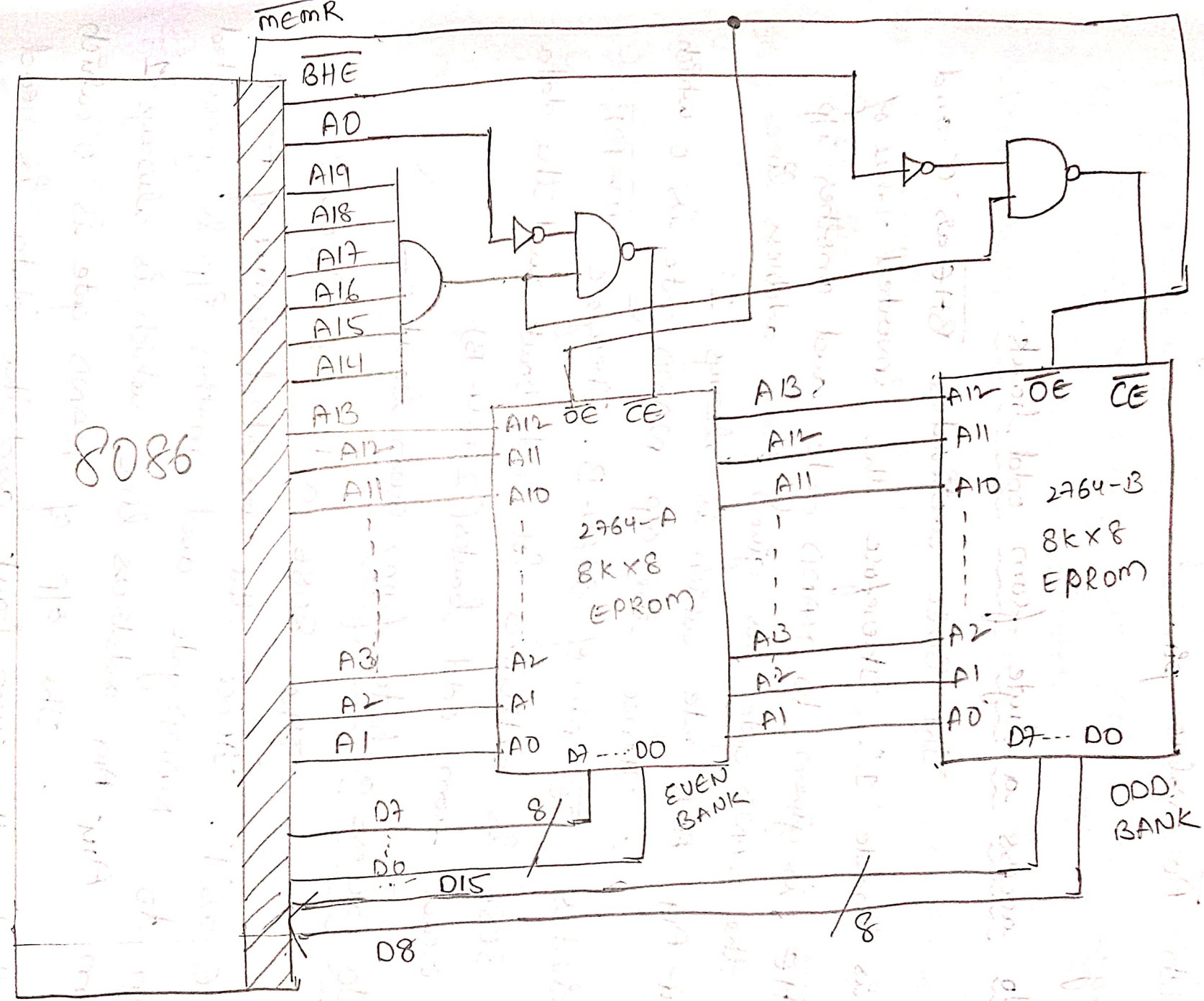
→ Since A0 = 0 for all even addresses it will be connected to chip select or chip enable signal ($\overline{CE}$) pin of 2764A along with higher order pins.

⇒ Similarly $\overline{BHE}$ is used for odd addresses ie, to connect $\overline{CE}$ of 2764B of 8086.

→ The given EPROM is 8K x 8 where 8K = 2^{13} ie, 13 address lines are required for each banks ie, 2764-A & 2764-B which are connected to A1 - A13 of 8086 with A0 - A12 of each banks

→ Remaining A14 - A19 of 8086 are used for address decoding.

→ Since all the address lines from A14 - A19 are 1, those are connected to an AND gate, and produce the o/p as '1'.


MEMR
BHE
A0
A19
A18
A17
A16
A15
A14
A13
A12
A11
A3
A2
A1
8086
A12 OE CE
A11
A10
2764-A
8k x 8
EPROM
A0
D7 --- D0
EVEN BANK
A13
A12
A11
A3
A2
A1
A12 OE CE
A11
A10
2764-B
8k x 8
EPROM
A2
A1
A0
D7 --- D0
ODD BANK
D7
D0
8
D15
D8
8

→ That AND gate o/p is given as one of the i/p's of NAND gate and another i/p of NAND gate is connected with $\overline{A_0}$ value pin.

Working:-

(i) To access a byte from odd bank.

The 8086 activates $\overline{BHE}$ as '0' and A_0 is made '1'. Therefore the inverted value of $\overline{BHE}$ is given to NAND gate and another i/p to the NAND gate is given from address line $A_{14}-A_{19}$ AND gate which is always '1'.

∴ Hence the o/p NAND gate is '0' which activates $\overline{CE}$ pin of 2764-B. Therefore $\overline{MEMR}$ of 8086 is activated for odd bank and the data is read from odd banks (2764-B).

(ii) To access a byte from even bank:-

The 8086 activates A_0 as 0 and $\overline{BHE}$ as '1'. Hence the inverted value of A_0 is given to NAND gate and another i/p is generated from $A_{14}-A_{19}$ address line which is always '1'.

∴ The o/p of NAND gate is '0' which activates the even bank and the data is read by activating $\overline{MEMR}$ of 8086.

(ii) To access 16 bit data from both the banks:-

To access both the banks, both A0 and $\overline{BHE}$ are made '0' which makes both the $\overline{CE}$ pin of 2764-A & 2764-B gets activated.

∴ 8086 activates $\overline{MEMR}$ signal and then reads the word from both banks.

* Interfacing of 2 8K X8 RAM chips with 8086 that have the address range 00000H – 03FFFH

A19	A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	Address
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0–00000H
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1–00001H
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	2–00002H
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	3–00003H
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	3FFEH
0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	3FFFH
0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	

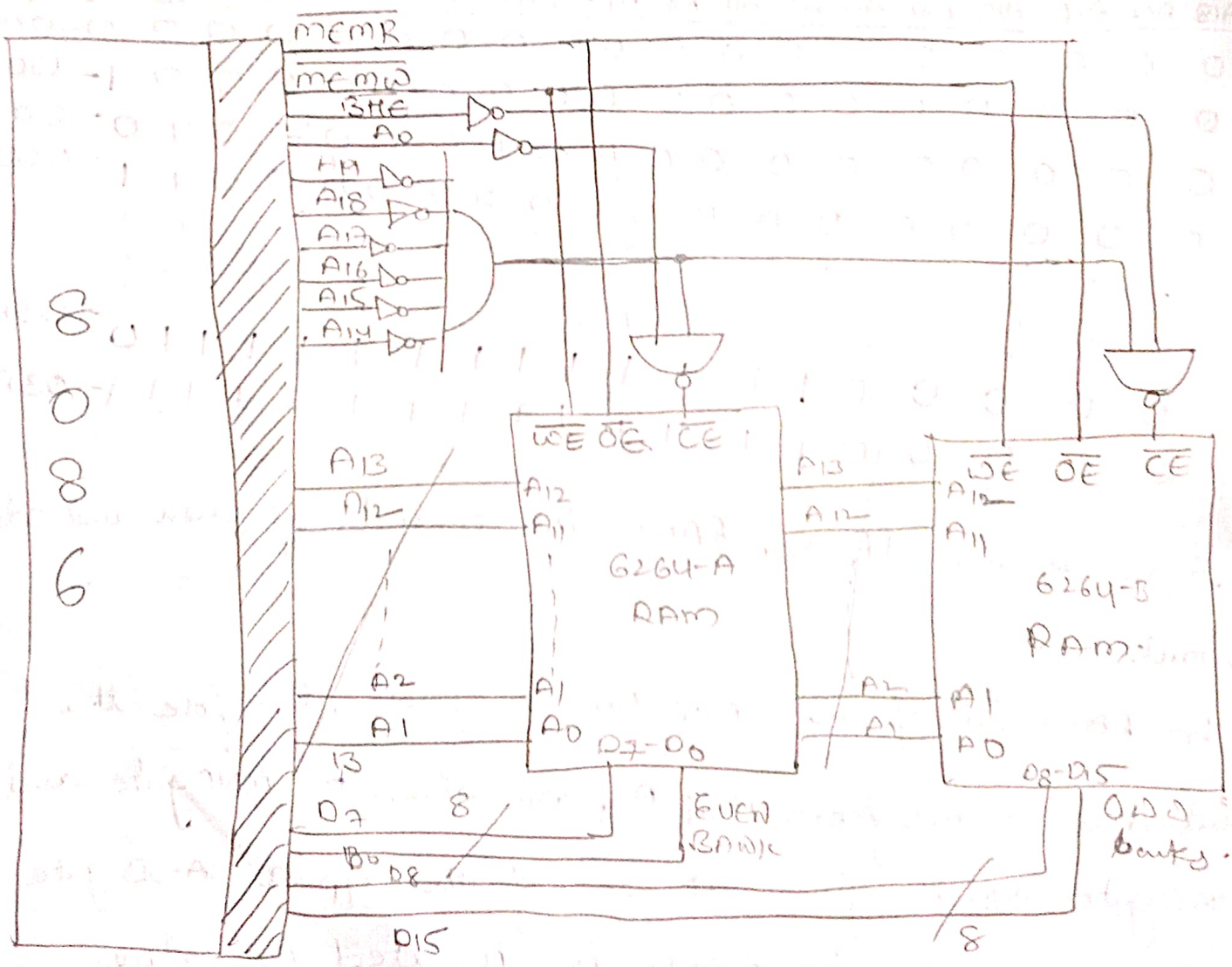
(1) Again as in EPROM, RAM also consists of even and odd banks

(2) In RAM interfacing A_{14}–A_{19} are zeroes, therefore the inverted values from A_{14}–A_{19} are given to AND gate and AND gate o/p is given as one of the i/p to NAND gate and another i/p to NAND is inverted A_0 value.

(3) And o/p of NAND gate is given to $\overline{CE}$ pin of EVEN bank RAM

(4) Similarly $\overline{BHE}$ will be connected to odd bank through a NAND gate with $\overline{CE}$ pin of odd bank.

(5) $\overline{MEMR}$ signal of 8086 is connected to $\overline{OE}$ (output enable) i/p of $\overline{MEMW}$ signal of 8086 is connected to $\overline{WE}$ (write enable) operation.



INTEL 8255 PROGRAMMABLE PERIPHERAL INTERFACE

* 8255 programmable peripheral interface (PPI) is a popular i/o chip which perform both digital I/O operations from the processors.

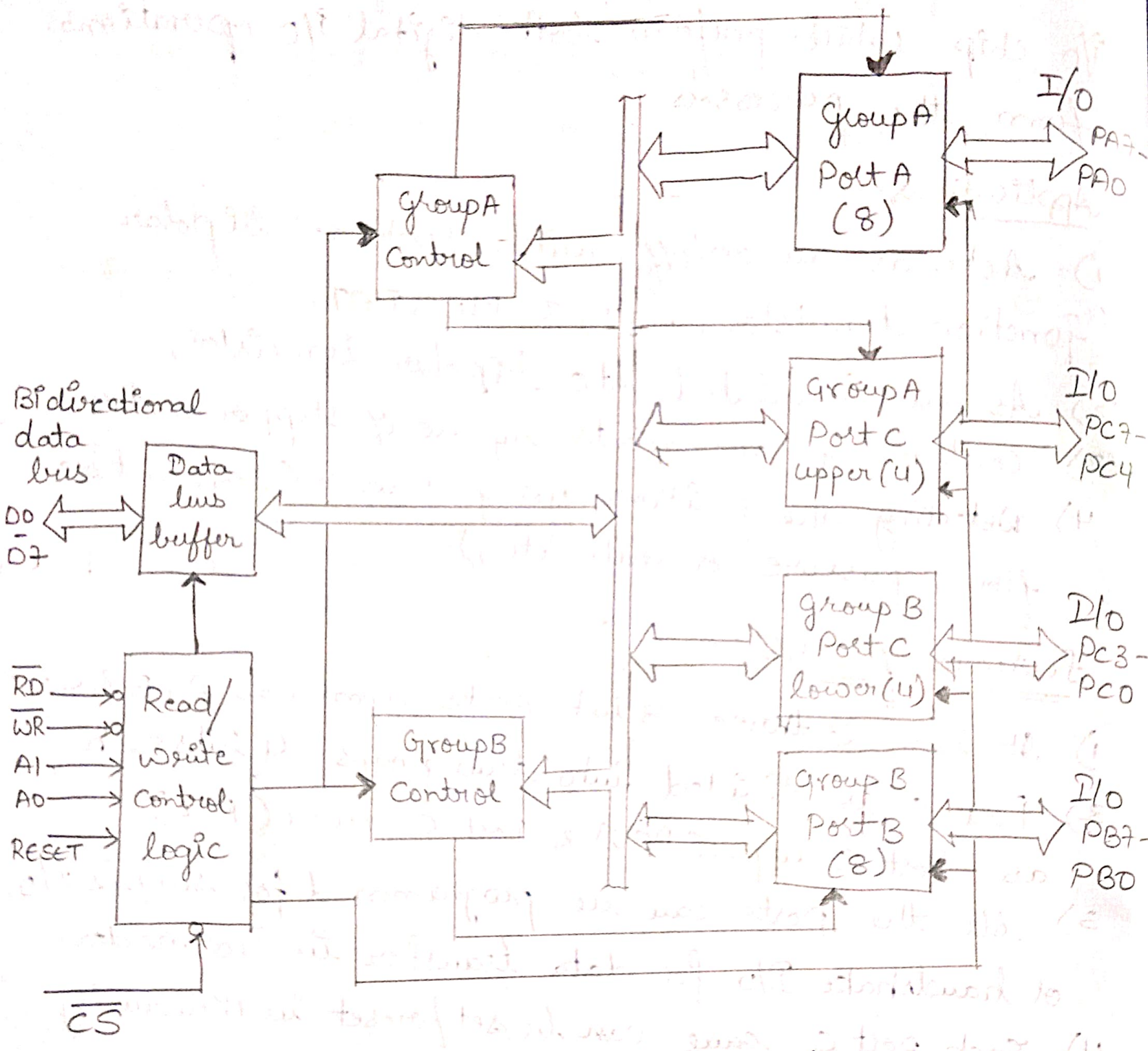
Applications :-

1) Acts as an on/off switch such as bipolar junction transistor (BJT) & MOSFET.
2) As an insulated gate bipolar transistor,
3) Controlling movements by use of stepper motors.
4) Detecting the position using sensors (temperature, flow, pressure or level etc.,)

Features of 8255 :-

1) It has 3 three 8 bit ports named as A, B & C.
2) Port C is divided into two groups 4 bits each as Port C upper (PCU) & Port C lower (PCL)
3) All the ports can be programmed for simple I/O, or handshake I/O for data transfer in I/O modes.
4) Each port C ~~can~~ can be set/reset individually in bit set/reset (BSR) mode.
5) Port A and PCU are grouped as group A
6) Port B and PCL are grouped as group B.
7) Along with three ports there will be another register called control register which decides the operating mode.

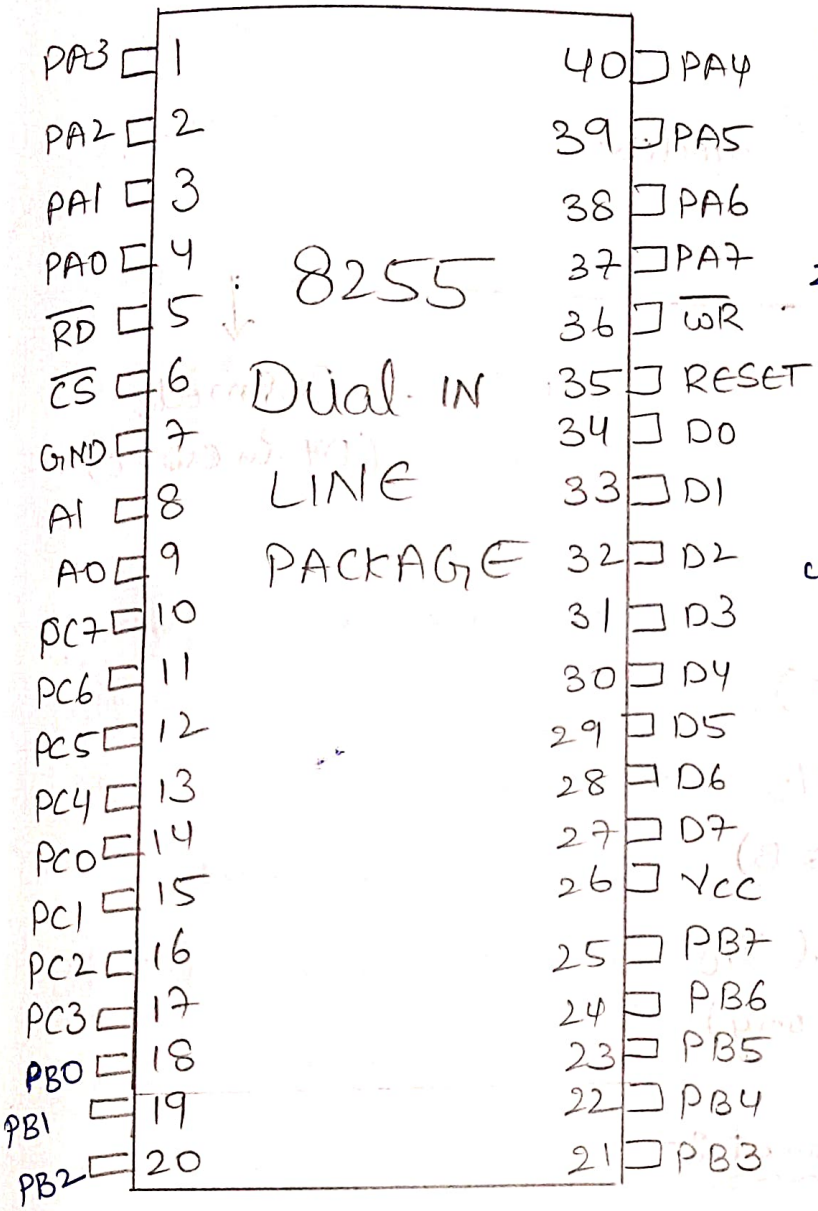
BLOCK DIAGRAM of 8255 :-



To identify or select the ports and control register A0 & A1 are used as below.

A1	A0		Register selected
0	0	—	Port A
0	1	—	Port B
1	0	—	Port C
1	1	—	Control register

The pin diagram of 8255 IC :-

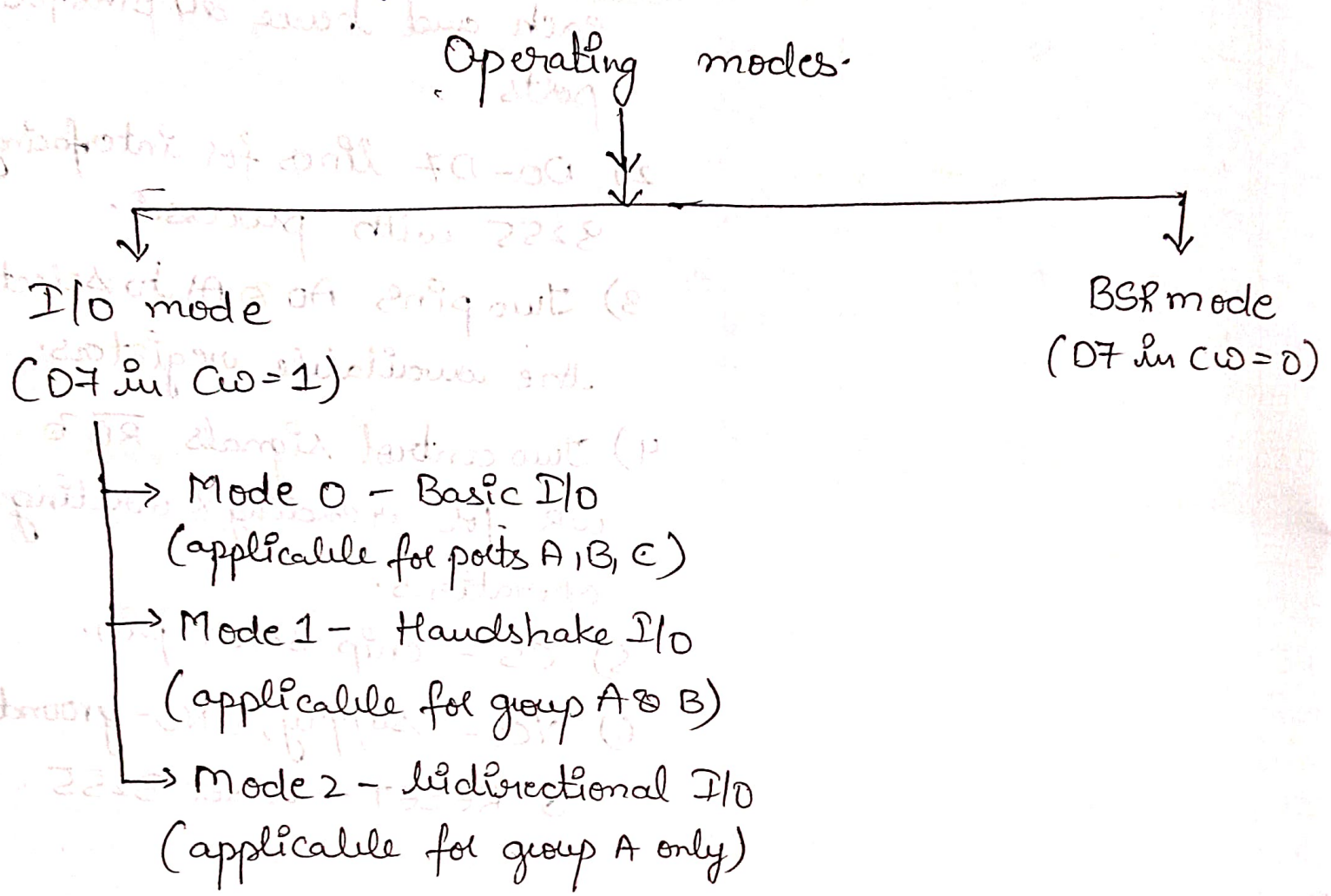


1) Three ports require 8 lines each and hence 24 pins for ports.
2) D0-D7 lines for interfacing 8255 with processor.
3) Two pins A0 & A1 to select the available registers.
4) Two control signals $\overline{RD}$ & $\overline{WR}$ for reading & writing operations.
5) $\overline{CS}$ - chip select pin.
6) Vcc - supply, GND - Ground & RESET - reset 8255.

Operating mode of 8255 and control word.

1) The software programming of 8255 is done by writing a control word (CW) to the control register of 8255.
2) Control word contains information such as mode, bit set/bit reset etc.,
3) The operating modes of 8255 is ~~divided~~ having two configurations ie., input/output mode (I/O mode) & BSR mode

3) I/O consists of 3 different modes for the ports as in below fig:



(i) I/O control word format:-

CWR (control word register) is of 8 bits from D0-D7 as given below.

D7	D6	D5	D4	D3	D2	D1	D0
1 (1=I/O mode)	Group A mode select 00 - mode 0 01 - mode 1 1x - mode 2		PortA 1- i/p 0- o/p	Port C upper 1- i/p 0- o/p	Group B mode select 0- mode 0 1- mode 1	Port B 1- i/p 0- o/p	Port C lower 1- i/p 0- o/p

Bits D6 & D5

D6 & D5 are used to select the operating modes of group A

(i) mode 0 (D6 & D5 are both 0's) – Basic I/O operations – Port A, B, C are operated as i/p as well as o/p.

(ii) Mode 1 (D6 & D5 are 0 & 1 respectively) – Strobed or handshaking operation:
– Port A controls the data transfer and port C upper is used for handshaking.

iii) Mode 2 (D6 & D5 are 1 & X respectively) – bidirectional bus
– Port A is bidirectional bus & Port C is used for handshaking.

Bit D4 – It decides whether port A is for i/p (= 1) or o/p (= 0)

Bit D3 – It decides whether port C upper (PC4–PC7) is used for i/p (= 1) or o/p (= 0).

Bit D2 :– selects the mode of group B. It operates only on two modes. ie.,

(i) Mode 0 :– Basic I/O for group B if D2 = 0

(ii) Mode 1 :– Strobed or handshaking operation.
If D2 = 1 Port B is in mode 1 configuration & port C lower are used for handshaking and control of data transfer.

Bit 1 :– Decides the data direction of port B
If it is 1 = acts as i/p
0 = acts as o/p.

Bit D0 :- Decides the data direction of Port C lines
If it is 1 - acts as i/p.
0 - acts as o/p.

BSR mode (To operate in BSR mode D7 bit should be 0)

Bit format of CWR is as follows.

D7	D6	D5	D4	D3	D2	D1	D0
0 (0-BSR mode)	X (Don't care)	X	X	B2	B1	B0	Bit set/Reset 1 = Set 0 = Reset

Bit select to select one of the 8 pins of port C.

BSR mode is to control port C pins individually.

Interfacing Switches & LED's :-

(1) Interface an 8255 with 8086 to work as an I/O port. Initialize port A as o/p port, port B as i/p and port C as o/p port. Port A address should be 0740H. Write a program to sense switch position SC_0-SC_7 connected at port B. The sensed pattern is to be displayed on port A, to which 8 LED's are connected, while port C lower displays total no of ON switches out of eight.

Sol :-

Given statements are 1) 8255 works as I/O mode
2) Port A — o/p port
3) Port B — i/p port

4) Port C upper as — o/p port
5) Port C lower as — o/p port

∴ The control word register (CWR) is

D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	0	0	0	1	0
I/O mode	mode 0		Port A o/p	Port C upper o/p	mode 0	Port B i/p	Port C lower o/p

∴ Control word = 82H.

Given port A address should be 0740H.

∴ Ports	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 (Port select)	A_1 (Port select)	A_0 (Port select)	
Port A	0	0	0	0	0	1	1	1	0	1	0	0	0	0	0	0	— 0740H
Port B	0	0	0	0	0	1	1	1	0	1	0	0	0	0	1	0	— 0742H
Port C	0	0	0	0	0	1	1	1	0	1	0	0	0	1	0	0	— 0744H
CWR	0	0	0	0	0	1	1	1	0	1	0	0	0	1	1	0	— 0746H.

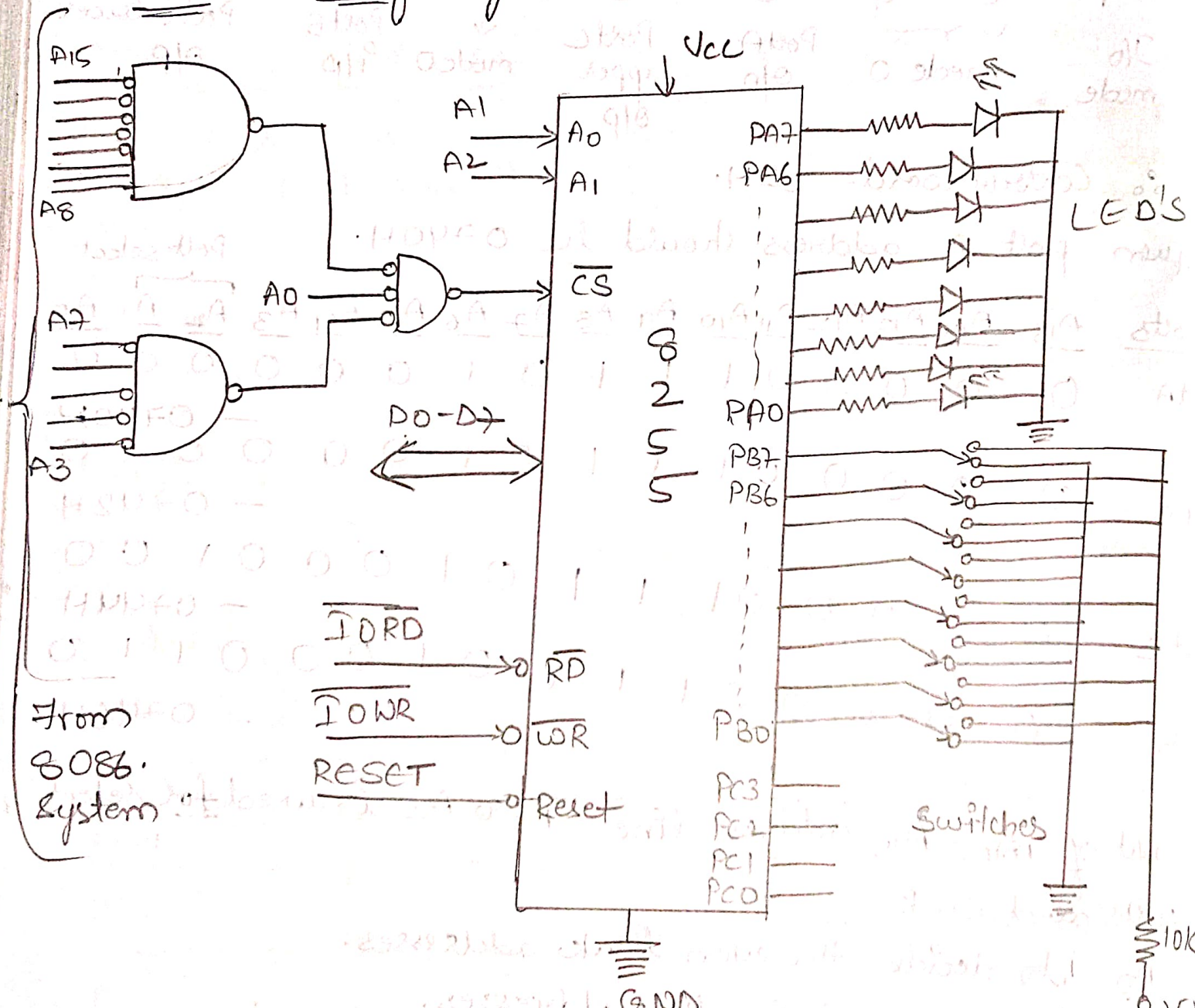
- Out of A_0 – A_{15} address line A_1 & A_2 is used for selecting ports and CWR
- A_0 to decide the even bank addresses.
- A_3 – A_{15} for deciding the addresses.

ALP for the program :—

```
MOV   DX, 0746H      # Initialize CWR
MOV   AL, 82H
OUT   DX, AL
```

```
SUB   DX, 04    # Port B
IN    DX, AL
SUB   DX, 02    # Port A
OUT   DX, AL.
```

8255 interfacing with 8086 :-



INTERFACING SEVEN-SEGMENT DISPLAYS :-

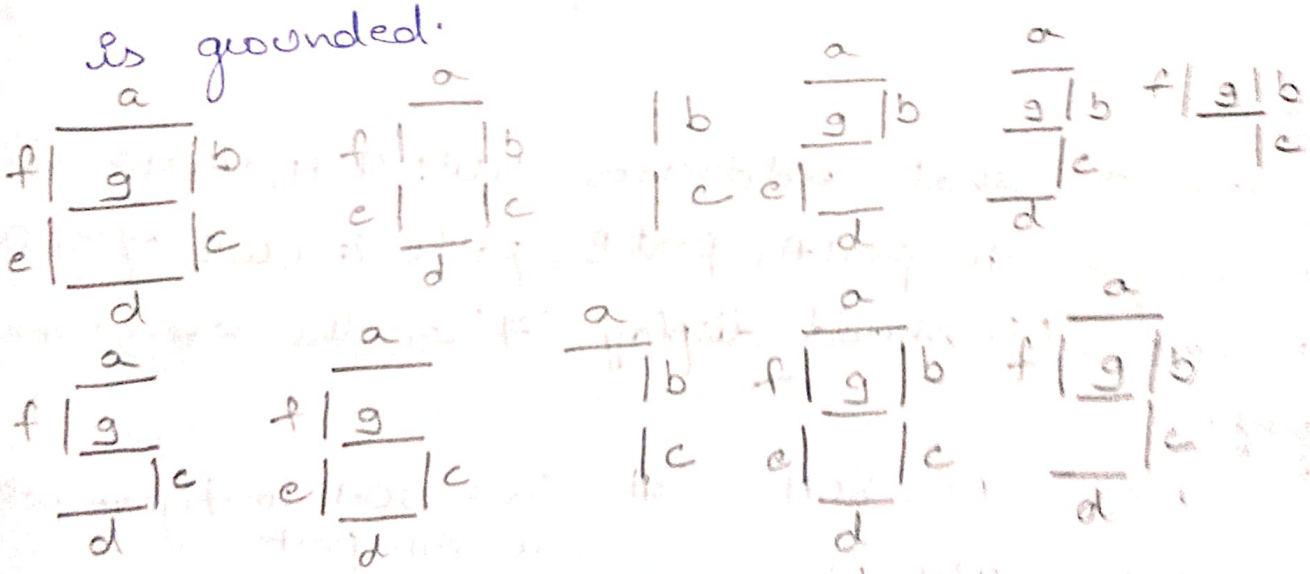
* Seven segment display consists of 7 LED segments.
* There is one pin for each segment named as a, b, c, d, e, f. One more LED for decimal point (dp). Another pin for power supply.
* Seven segment display comes in two types,

(i) Common anode :- a) the anodes of all segments LED are connected together first.

b) To illuminate a segment from a to f common anode is connected to supply and segment (a to f) inputs are connected to voltage low or logic '0'.

(ii) Common cathode :- a) All the cathodes of 7 LEDs are connected together first.

b) To illuminate the required segment it is to be connected to logic '1' and common cathode is grounded.



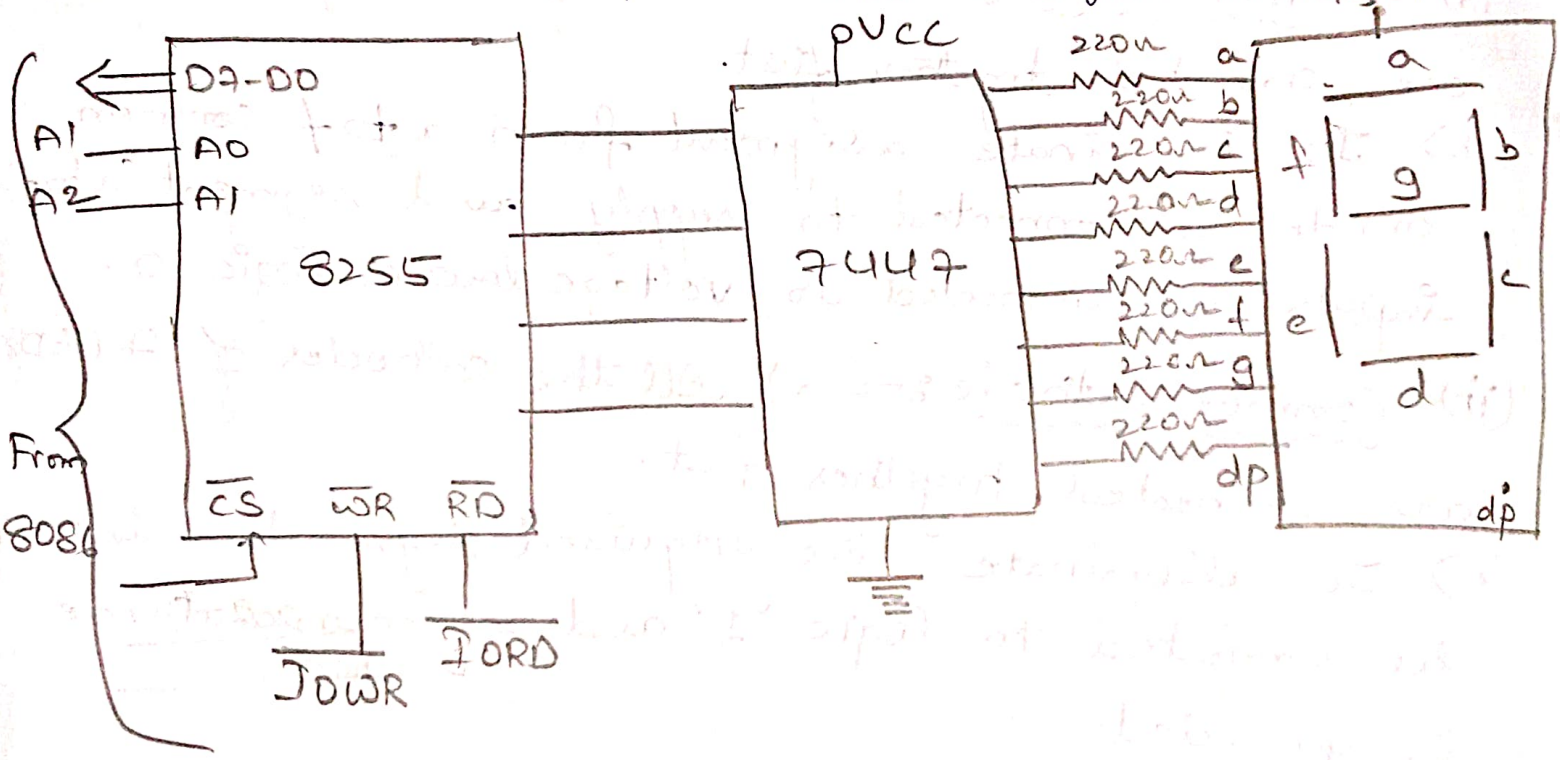
LED arrangements for 7 segments :-

1) The IC 7447 is a BCD to seven segment display converter.

2) IC 7447 generates active low o/p's (logic 0) to illuminate the segments from a to f.

3) Therefore IC 7447 is an active common anode display.

The complete circuit diagram is as follows



Program:-

Consider that addresses 40H, 42H, 44H & 46H are assigned to port A, port B, port C & CWR of 8255. And the LED should display '7' on the seven segment display.

Prog:-

```
MOV   AL, 80H    # CW = 80H configure port A as o/p port
OUT   46H, AL    # CW register is initiated.
MOV   AL, 07H    # Number that is to be displayed is 7
```

where PA0–PA3 are connected to 7447 IC

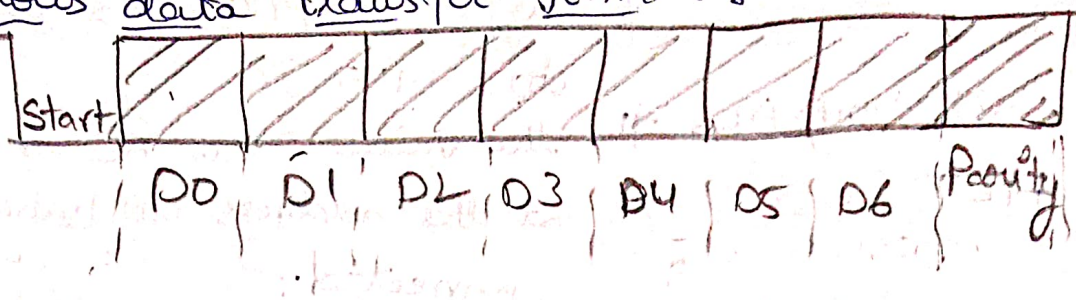
MOV OUT 40H, AL # the value '7' in AL is given to port A at the address 40H where display is connected.

HLT

SERIAL COMMUNICATION :–

* Serial communication is sending and receiving information bit by bit.
* It is preferred for long range communications as it is easy to implement using a single wire.
* Serial data systems are done in three forms
  1) Simplex – data flows only in one direction Eg:- TV
  2) Half duplex – communication takes place in both the direction but not at an instant of time. Eg:- walky talky
  3) Full duplex :– Communication takes place in both the direction at an instant of time.
     Eg:- Telephones.
* Serial data can be sent either in synchronous or asynchronous modes.
  (a) Synchronous transmission :– data is sent in blocks with constant rate (frequency of transmision and reception)
  (b) Asynchronous transmission :– data is sent in asynchronized form randomly. In this each data character has a bit format to identify its start & two or more bits to identify its end.

Asynchronous data transfer format :-



* Baud rate :- It is the rate (speed) at which the serial data is being transferred. It is measured in bits/second.

$$\text{Baud rate} = \frac{1}{\text{time b/w signal transitions}}$$

Eg :- Data changes every 1.67 ms.

$$\therefore \text{Baud rate} = \frac{1}{1.67\text{ ms}} = \frac{1}{1.67 \times 10^{-3}} = 600 \text{ Bd.}$$

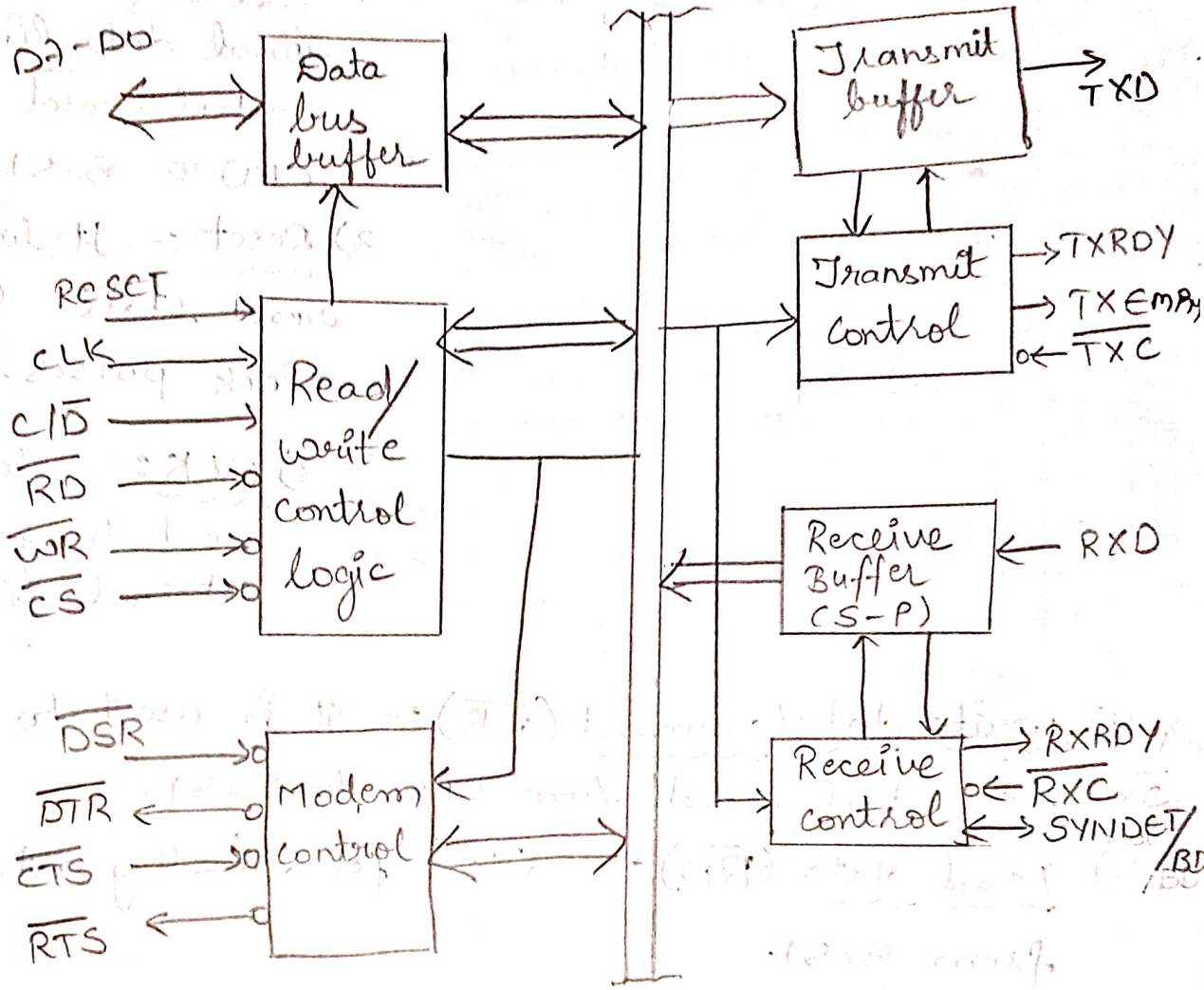
∴ The common baud rates are 600, 300, 1200, 2400, 4800, 9600, 19200.

Features of 8251 USART :-

8251 is a universal synchronous asynchronous receiver transmitter (USART) used for serial data communication.

8251 converts serial data from outside and send the parallel data to CPU and receives parallel data from CPU and sends serial data to peripherals. It works in full duplex mode with variable baud rate in both synchronous and asynchronous form.

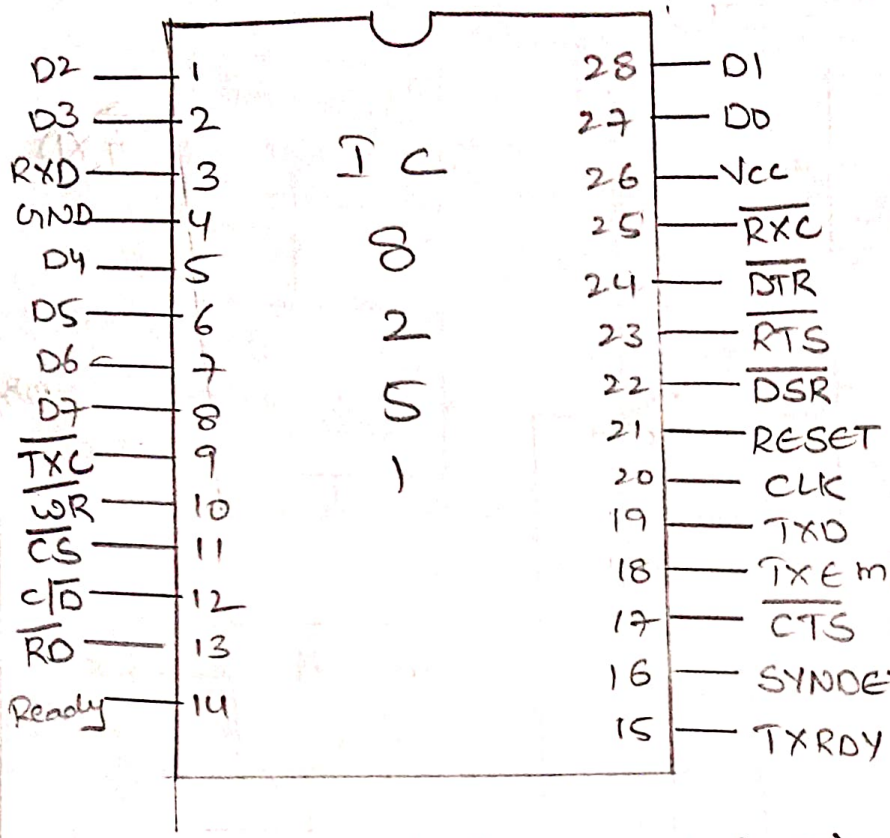
The internal block diagram of 8251 is shown in below fig;



Basic operations of 8251 and related control signals-

$\overline{CS}$	C/$\overline{D}$	$\overline{RD}$	$\overline{WR}$		Function
1	x	x	x	—	Chip not selected, data bus in high Impedance state.
0	x	1	1	—	data bus in high impedance state
0	1	0	1	—	Status word read by CPU from status register.
0	1	1	0	—	Control word written into control register by CPU
0	0	0	1	—	Data read by CPU from data register
0	0	1	0	—	Data written into data register by CPU

PIN DIAGRAM OF 8251 :-



1) D0-D7 :- 8 bit data bidi-tional data lines used fo control word transfer b/w CPU & 8251

2) Reset :- It brings 8251 reset state. It requires clock pulses to reset 82

3) CLK :- Clock signal used to generate internal device timing

4) write data/command ($\overline{WR}$) :- It is used to write dat or control word from CPU to 8251.

5) Read data ($\overline{RD}$) :- Used for reading data and statu from 8251.

6) C/$\overline{D}$ (Control/Data) :- If C/$\overline{D}$ = 1 command wo or status word is accessed.

If C/$\overline{D}$ = 0, data are accessed.

7) $\overline{CS}$:- It selects 8251 for CPU accesses.

8) Transit data line (TXD) :- It is an output signal for transmitting serial data from 8251.

9) Transmitter ready (TXRDY) :- It is an o/p signal which indicates that 8251 is ready to accept a data character.

Transmitter Empty (TXEMPTY):- It is an o/p signal that indicates that 8251 is ~~ready~~ has no data for transmission.

Transmitter clock ($\overline{TXC}$):- Clock i/p signal determines the transfer speed of transmitted data or also called as baud rate.

Receive data (RXD):- A signal line that receives serial data.

Receiver ready (RXRDY):- 1) This signal indicates that 8251 is ready with a character that is to be read by the CPU.

2) If CPU reads the data, RXRDY will be reset.

3) If the CPU is reading the data characters and the next one has arrived. Then the preceding data will be lost. Such an error is called overrun error flag status.

Receiver clock ($\overline{RXC}$):- It determines the transfer speed of received data or baud rate of reception.

Sync detect/Break detect (SYNDET/BD):- It is a active high o/p signal.

(1) In asynchronous mode, it is used to indicate break of data.

2) In synchronous mode it indicates the correct receipt of data.

The following signals are for modem for hand-shaking and establishing of connection.

1) Data set ready ($\overline{DSR}$):- It is an i/p signal that indicates 8251 from the modem interface. $\overline{DSR}$ indicates

that modem is powered up.

2) Data terminal ready ($\overline{DTR}$) :- It is an o/p signal from 8251 from modem interface. It indicates that 8251 is powered up.

3) Clear to send ($\overline{CTS}$) :- It is an i/p signal to 825 from modem interface. If it set to low level, it indicates that data is transmittable if TXE is in enable status.

4) Request to send ($\overline{RTS}$) :- It is an active low o/p signal sent to modem by 8251, that it is ready to send data.

⇒ Control word :-

8251 operations are initialized by send control word and then the operatio are carried out.

There are two types of control word

(i) mode instruction (setting of function).

2) command instruction (setting of operation).

(i) mode command word :-

After resetting of 8251 mode instructio is recognized first.

The functions of mode instruction are follows.

Mode instruction – asynchronous mode.

D7 D6	D5 D4	D3 D2	D1 D0
Frame control stop bit length 00- inhibit 01- 1 stop bits 10- 1.5 stop bits 11- 2 stop bits	Parity check X0 - Disable 01 - odd parity 11 - even parity	Character length 00 - 5 bits 01 - 6 bits 10 - 7 bits 11 - 8 bits.	Baud rate select bits 0 0-SYN mode 0 1- 1X clock 1 0- 16X clock 1 1- 64X clock

If D1 & D0 in this format is "00" then 8251 works in synchronous mode. And its mode instruction format is as follows.

D7	D6	D5 D4	D3 D2	D1 D0
No of sync characters 0- 2 characters 1- 1 characters	Synchronous mode 0- internal synchronization 1- External synchronization	Parity check X0 - disable 01 - Odd 11 - Even	Character length 00 - 5 bits 01 - 6 bits 10 - 7 bits 11 - 8 bits	X X no operation

2) Serial command word :-

Command word format is followed by writing its mode word.

The function are as follows along with its format and explanation.

Mode instruction – asynchronous mode.

D7 D6	D5 D4	D3 D2	D1 D0
Frame control stop bit length 00- inhibit 01- 1 stop bits 10- 1.5 stop bits 11- 2 stop bits	Parity check X0 - Disable 01 - odd parity 11 - even parity	Character length 00 - 5 bits 01 - 6 bits 10 - 7 bits 11 - 8 bits.	Baud rate select bits 0 0-SYN mode 0 1- 1X clock 1 0- 16X clock 1 1- 64X clock

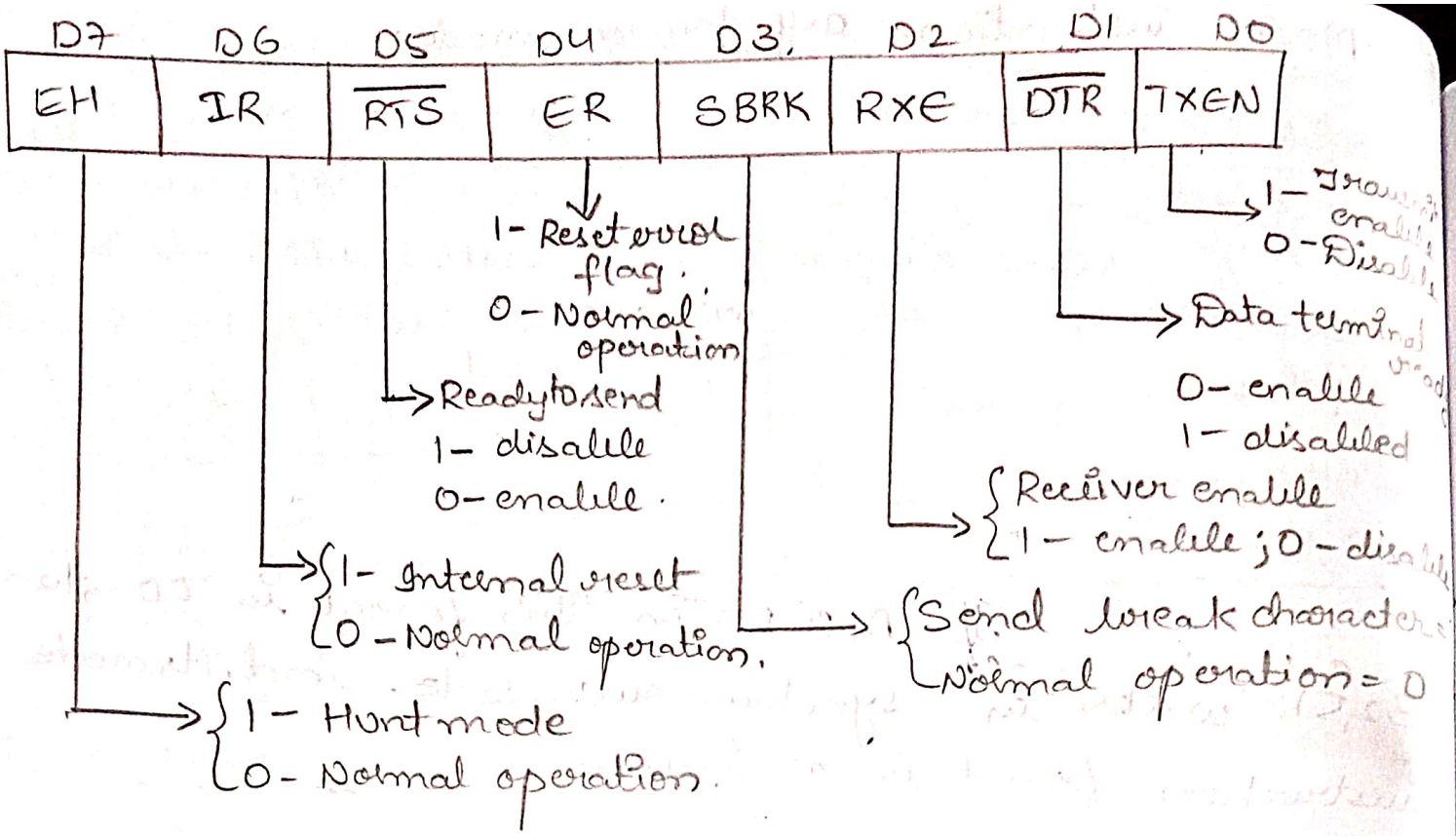
If D1 & D0 in this format is "00" then 8251 works in synchronous mode. And its mode instruction format is as follows.

D7	D6	D5 D4	D3 D2	D1 D0
No of sync characters 0-2 characters 1-1 characters	Synchronous mode 0-internal synchronization 1- External synchronization	Parity check X0 - disable 01 - Odd 11 - Even	Character length 00 - 5 bits 01 - 6 bits 10 - 7 bits 11 - 8 bits	X X no operation

2) Serial command word :-

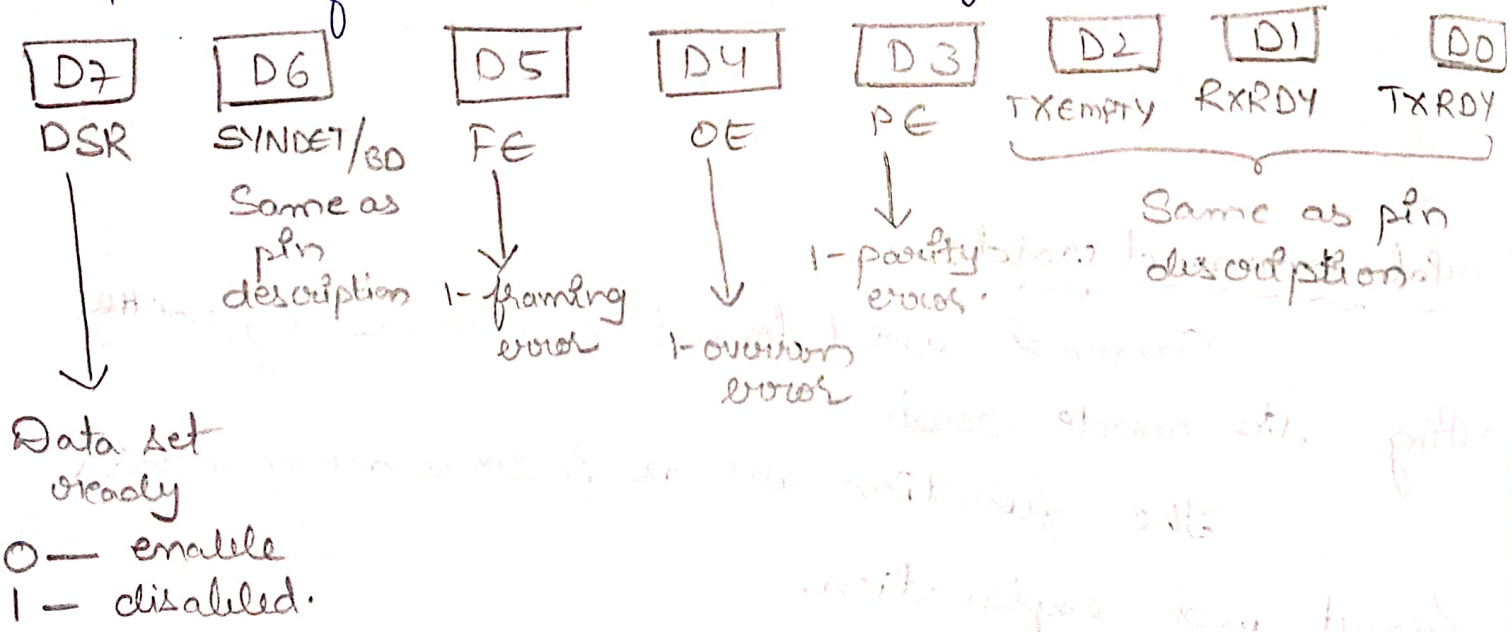
Command word format is followed by writing its mode word.

The function are as follows along with its format and explanation.



Status read instruction format:-

It is possible to see the internal status of 8251 just by reading a status word.



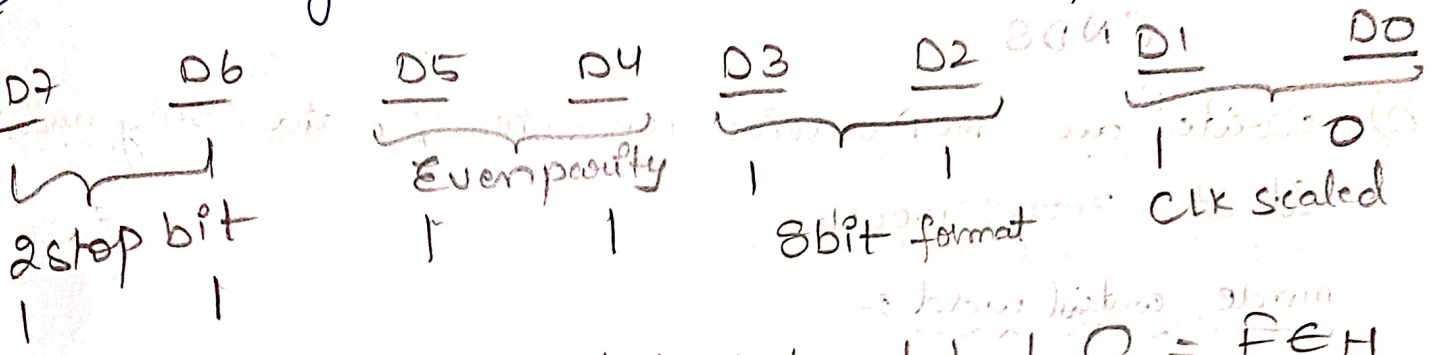
Problem:- Interfacing of 8251 with 8086.

- Design a hardware interface circuit for interfacing 8251 with 8086. Set 8251A in asynchronous mode as

transmitter and receiver with even parity enabled, 2 stop bits. 8 bit character length, frequency 160KHz and baud rate 10K.

a) write an ALP to transit 100 bytes of data string starting at location 2000H : 5000H.

Sol^n^ For asynchronous mode control word format is



Control word = 1 1 1 1 1 1 1 0 = FEH

Program :-

```
Assume   CS: code
code segment
Start :  mov   AX, 2000H
         mov   DS, AX
         mov   SI, 5000H
         mov   CL, 64H
         mov   AL, 0FEH
         OUT   0FEH, AL
         mov   AX, 11H
         OUT   0FEH, AX
WAIT :   IN    AL, 0FEH
         AND   AL, 01H
         JZ    WAIT
         mov   AL, [SI]
```

```
INC   SI
DEC   CL
JNZ   WAIT
INT   03H
Code  ENDS
END   START
ENDS
```

b) Write an ALP receive 100 bytes of data storing and it at 3000:4000H.

mode control word :-

D7	D6	D5	D4	D3	D2	D1	D0
0	1	1	1	1	1	1	0 = 07E

one stop bit

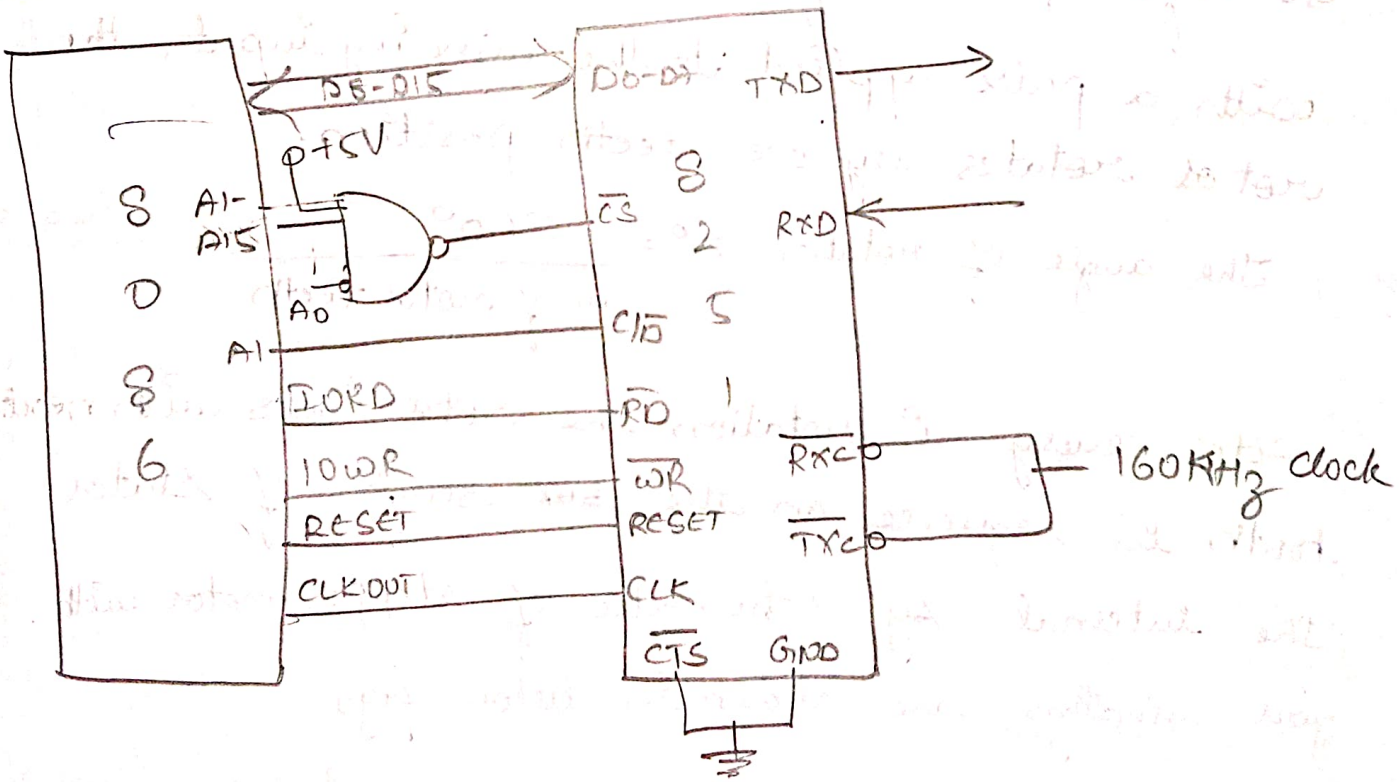
Command control word:-

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	1	0	1	0	0 = 14H
Normal operation	Normal operation	Ready to send	Reset error flags	Normal	RxE	DTR ready	No Tx

```
Assume  CS: Code
Code    segment
Start:  MOV  AX, 3000H
        MOV  DS, AX
        MOV  SI, 4000H
        MOV  CL, 64H
        MOV  AL, 7EH
        OUT  0FEH, AL
        MOV  AL, 14H
        OUT  0FEH, AL
```

```
NXTBT:  IN   AL, 0FEH
        AND  38H          00111000=38H
        JZ   READY
        MOV  AL, 14H
        OUT  0FEH, AL
READY:  IN   AL, 0FEH
        AND  02H          00000010=02H
        JZ   READY
        IN   AL, 0FCH
        MOV  AL, [SI]
        INC  SI
        DEC  CL
        JNZ  NXTBT
        CODE ENDS
        END START
        ENDS
```

INTERFACING DIAGRAM:-



INTERFACING STEPPER MOTORS :-

→ Stepper motor is a device used to obtain an accurate position control of rotating shafts.

→ It rotates its shafts in terms of steps, rather than continuous rotations in case of AC or DC motors.

→ To rotate the shafts in a direction, sequence pulses are applied to the windings in a proper sequence.

→ No of pulses required for complete rotation of the shaft = no of internal teeth on its rotor.

→ The stator teeth and the rotor teeth lock each other to a fix position of the shaft.

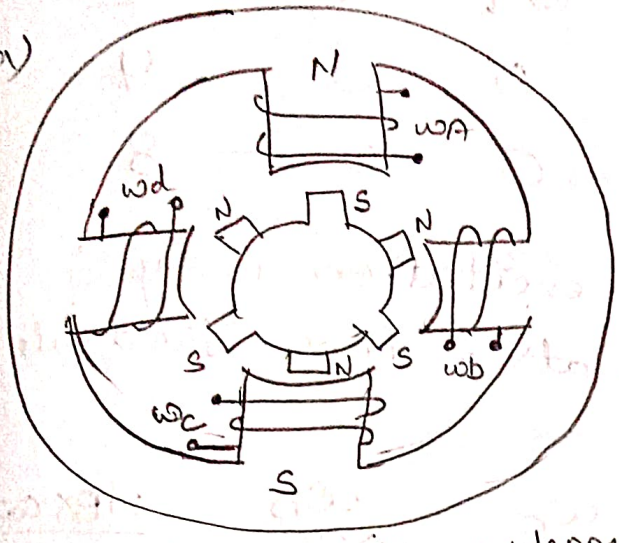
→ with a pulse applied to the winding input, the rotor rotates by one teeth position.

The angle of rotation $x^\circ = \frac{360^\circ}{\text{no of rotor teeth}}$.

→ After every x° rotation the rotor locks with next teeth in sequence on the suf surface of Stator.

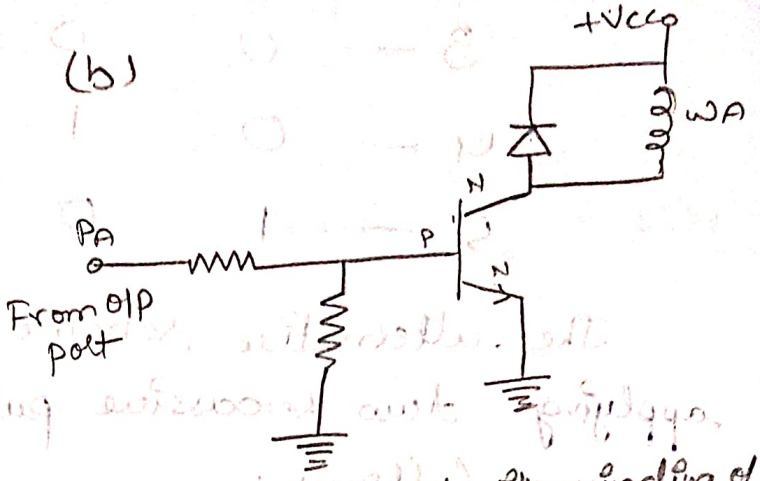
→ The internal sy schematic of stepper motor with four windings are shown in below fig;

a)



Internal schematic of stepper motor with four windings.

(b)



Interfacing 8051 with winding of stepper motor.

=> A typical stepper motor has 200 no of teeth (rotor)

∴ the angle $x = 360°/200 = 1.8°$ ... step angle.

=> The scheme of rotating a shaft of stepper motor is called wave scheme.

=> To rotate in clock wise direction the voltage pulses are applied in a cyclic fashion. ie., firstly to winding wA then to wb, wc, wd.

Clock wise →  → wA → wB → wC → wd → wa
1 2 3 4 5

For anticlockwise → → wA → wD → wC → wB → wA.

The successive excitation sequence is as follows.

motion	Step	wA	wB	wC	wD	Hexcode
Clockwise →	1	1	0	0	0	8
	2	0	1	0	0	4
	3	0	0	1	0	2
	4	0	0	0	1	1
	5	1	0	0	0	8
	1	1	0	0	0	8
Anticlock →	2	0	0	0	1	1

3 —	0	0	1	0	— 2
4 —	0	1	0	0	— 4
5 —	1	0	0	0	— 8

The alternative scheme of excitation is by applying two successive pulses at a time. The sol... is as follows.

motion	Step	W_A	W_B	W_C	W_D	HEX
Clockwise	1 —	1	1	0	0 —	0C
	2 —	0	1	1	0 —	06
	3 —	0	0	1	1 —	03H
	4 —	0	0	0	1 —	01H
	5 —	1	1	0	0 —	0CH
Anticlockwise	1 —	0	0	1	1 —	03H
	2 —	0	1	1	0 —	06H
	3 —	1	1	0	0 —	0CH
	4 —	1	0	0	0 —	08H
	5	0	0	1	1 —	03H

Problem:- Design a stepper motor controller and write an ALP to rotate shaft of a 4 phase stepper motor:

(1) Clockwise 5 rotations

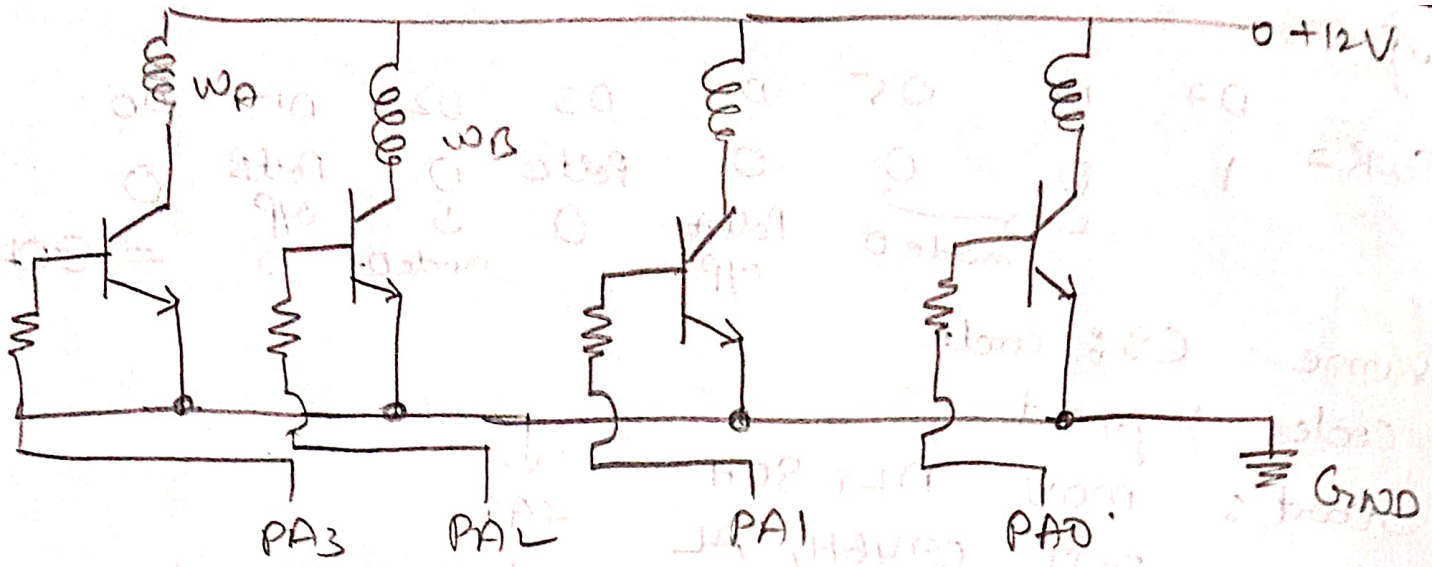
(2) Anticlockwise 5 rotations.

The 8255 port A address is 0740H. Stepper motor has 200 teeth rotor. Port A bit PA0 drives W_A, PA1 drives W_B and so on. Initial delay of 10msec (Assume that delay is already available)

Program:-

	D7	D6	D5	D4	D3	D2	D1	D0	
CWR =	1	0	0	0	PortC	0	PortB	0	
		mode 0		PortA o/p	0	mode 0	o/p 0		= 80H

```
Asume   CS: code
code    segment
Start:  MOV   AL, 80H
        OUT   0746H, AL
        MOV   AL, 88H
        MOV   CX, 1000      # count of rotation = 200x5
                                = 1000
L1:     OUT   PortA, AL
        CALL  Delay.
        ROL   AL, 01
        DEC   CX
        JNZ   L1
        MOV   AL, 88
        MOV   CX, 1000
L2:     OUT   PortA, AL
        CALL  Delay.
        ROR   AL, 01
        DEC   CX
        JNZ   L2
        INT   03H
CODE    ENDS
        END   START
        ENDS.
```



ADC (Analog to digital converter) :-

1) ADC is treated as i/p device by the micro processor.
2) It is interfaced with 8086 with 8255 as an o/p port.
3) Conversion starts with a pulse of duration called
   a) SOC - start of conversion pulse to ADC.
   And the conversion ends with a signal edge o
   b) EOC → End of conversion signal, which marks the end of conversion process.
   c) Then the digital data can be read out of ADC as equivalent digital o/p.

The time taken by the ADC from the activ edge of SOC pulse till the edge of EOC signal is called "conversion delay" of ADC

The most popular ADC's techniques are successive approximation and dual slope integration techniques.

Now we study about successive approximation technique which has IC 0808 ie:, ADC 0808, which is an 8-bit IC.

The general algorithm for conversion is

1) Stabilize analog input applied to ADC.
2) Issue start of conversion (SOC) pulse to ADC
3) Read End of conversion (EOC) signal.
4) Read digital data o/p of the ADC.

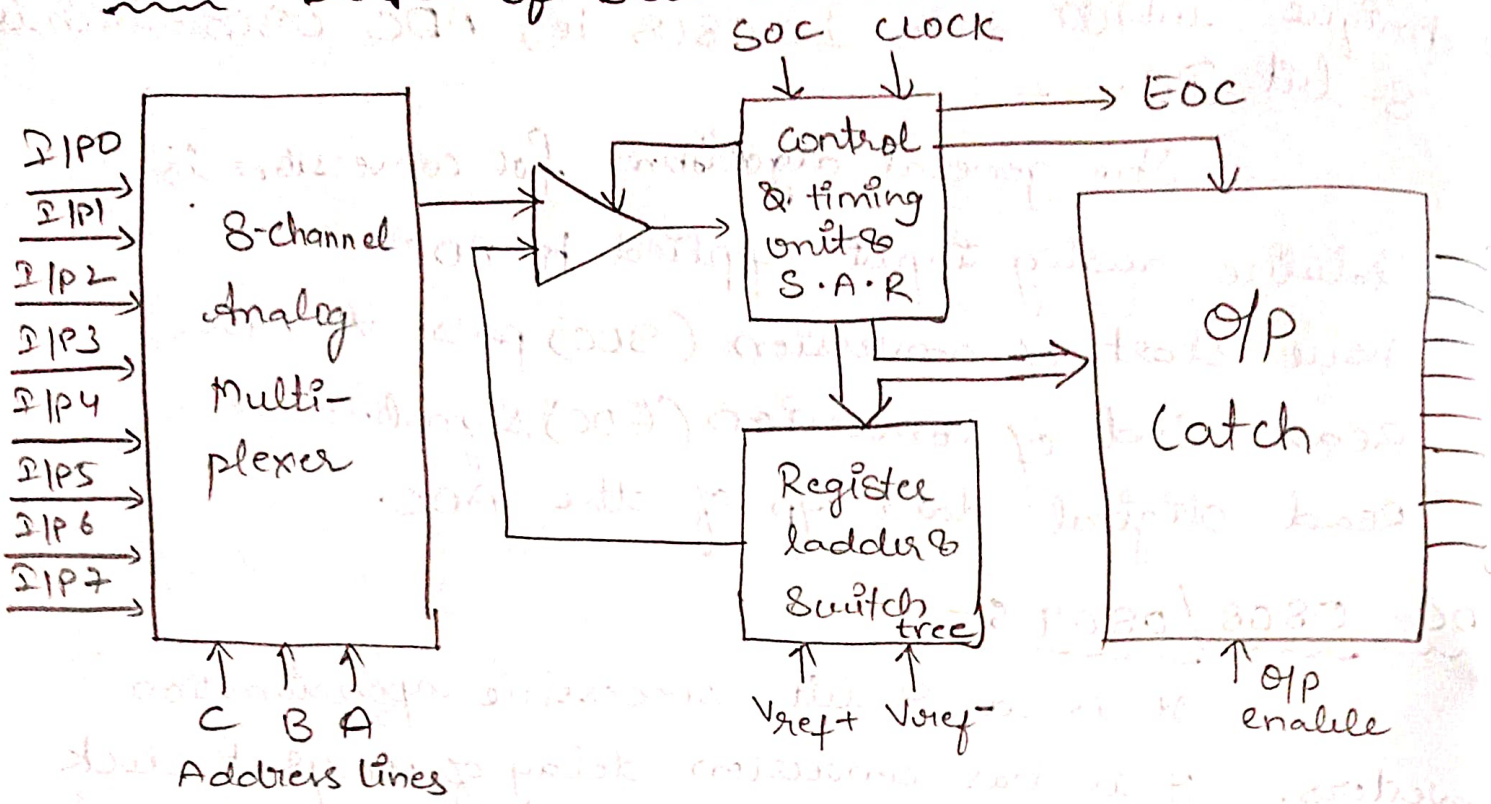
ADC 0808/0809 :-

It is a 8-bit, successive approximation converters. It is has conversion delay of 100μs at clock frequency of 640kHz.

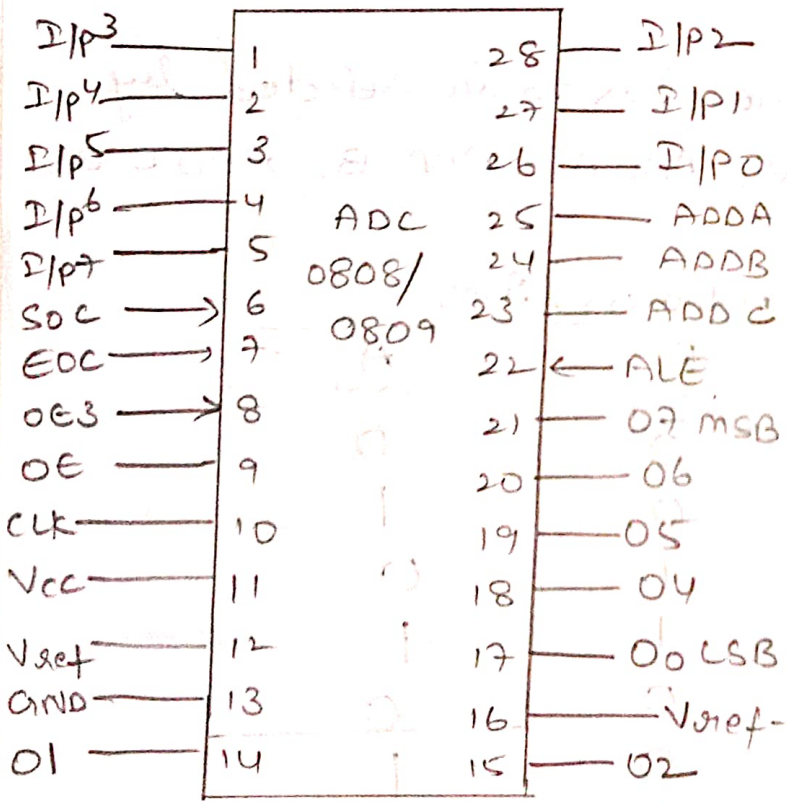
- These converters internally have a 3:8 analog multiplexers.
- Out of eight i/p's only one has to be selected by using 3 address lines ie, ADD A, ADD B, ADD C as shown below table.

Analog i/p selected	Address lines C	B	A
I/p 0	0	0	0
I/p 1	0	0	1
I/p 2	0	1	0
I/p 3	0	1	1
I/p 4	1	0	0
I/p 5	1	0	1
I/p 6	1	1	0
I/p 7	1	1	1

Block diagram of ADC 0808/0809 :-



PIN DIAGRAM OF ADC 0808/0809 :-



(i) IP0-IP7 – i/p lines anal

(ii) ADD-A,B,C – Address lines fo selecting i/p's.

(iii) $O_7 - O_0$ – Digital 8 bit op.

(iv) SOC – start of conversion Si

v) EOC – End of conversion signa

(vi) OE – o/p latch enable

vii) CLK – clock signal.

viii) Vcc, GND – Supply +5V & gro

ix) Vref+, Vref- – reference volta
+ve & +5V)
negative reference V

Vref – (0V)

* Interface ADC 0808 with 8086 using 8255 ports. Use port A of 8255 for transferring digital data o/p of ADC to the CPU and port C for control signals. Assume that an analog i/p is present at I/P2 of the ADC and a clock i/p of suitable frequency is available for ADC. Draw schematic and write the ALP.

Sol:- (1) Initialize port A as i/p data from 0808 ADC to the CPU.

(2) i/p2 is selected ∴ A, B, C address lines values are A = 0 ; B = 1 ; C = 0.

3) OE & ALE pins are enabled with +5V.

4) Port C upper act as i/p port to receive 'EOC'

5) Port C lower act as o/p port to send 'SOC' to ADC

SOC – start of conversion.

EOC – End of conversion.

∴ CWR value is

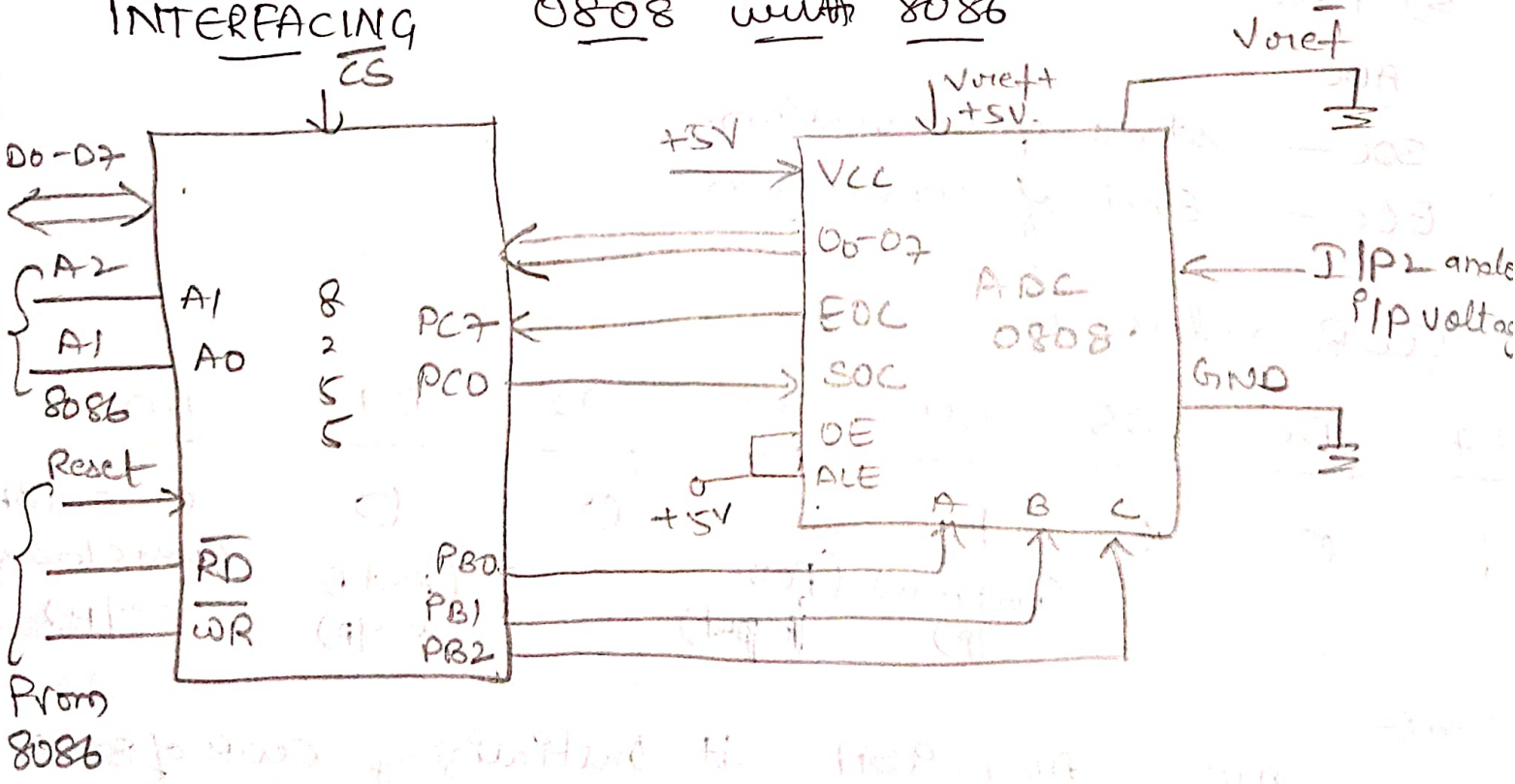
D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	1 (Port A as i/p)	1 (PC upper input)	0	0 (port B as o/p)	0 = 98H (port C lower as o/p)

Program:-

```
MOV   AL, 98H   # Initializing CWR of 8255
OUT   CWR, AL
MOV   AL, 02H   # i/p2 is selected as analog input in ADC
OUT   Port B, AL
MOV   AL, 00
```

```
        OUT   Port C, AL      # Give SOC to AD
        MOV   AL, 01H
        OUT   Port C, AL
        MOV   AL, 00
        Rout  Port C, AL
wait:   IN    AL, Port C      # Check for EOC by reading port C upper
        RCL
        JNC   wait
        IN    AL, Port A      # If EOC appears, read equivalent 8 bit digital o/p data through port A.
        HLT
```

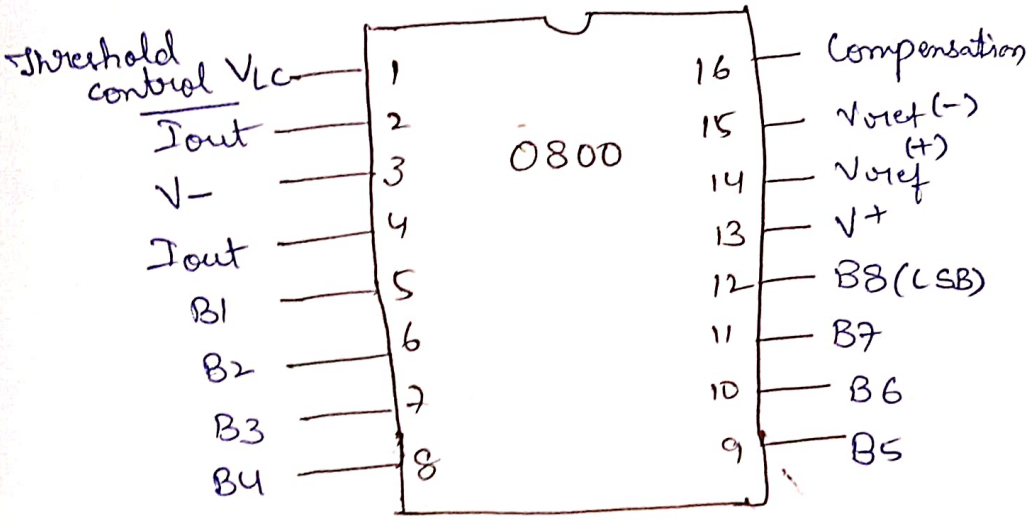
INTERFACING 0808 with 8086



INTERFACING DIGITAL TO ANALOG CONVERTERS:-

Digital to analog converts binary numbers into their analog equivalent voltages.

* The most commonly used DAC is DAC 0800 that can be easily interfaced to 8086 through 8255.
* 8255 acts as an o/p port to give data from the processor to DAC chip.
* One port of 8255 is enough to interface an 8-bit DAC.
* The internal pin diagram of 0800 DAC is as follows.



B1 - B8 - digital input data lines

I_{out} - Current output

$\overline{I_{out}}$ - complement of current o/p

V^- - negative supply voltage

$V+$ - positive supply voltage.

Comp - compensation voltage.

V_{LC} = threshold control.

V_{ref}^+ = positive reference voltage, V_{ref}^- - negative reference voltage

Interfacing with 8086