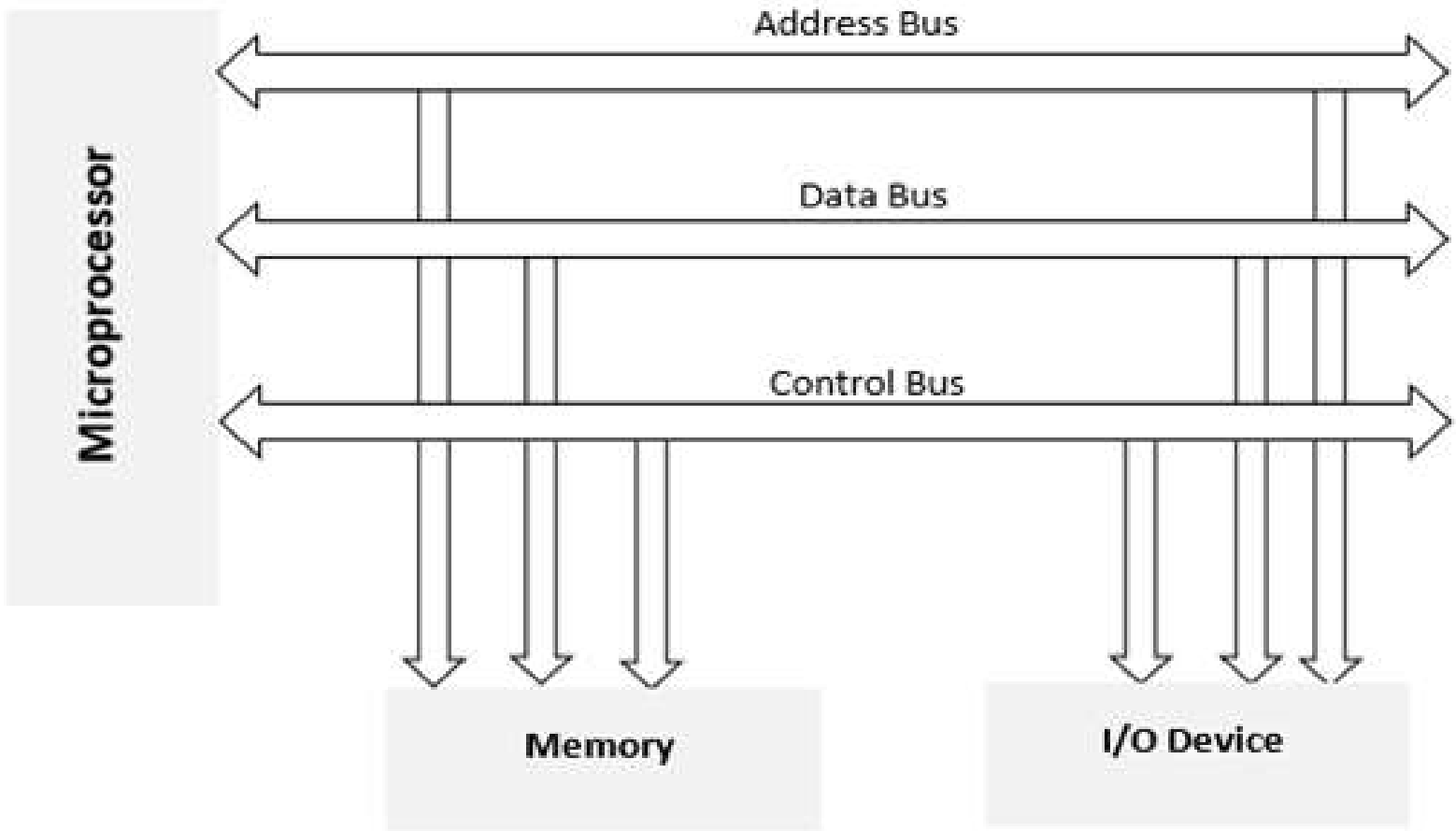
UNIT III:8086 Interfacing

- Semiconductor memories interfacing (RAM, ROM)
- Intel 8255 programmable peripheral interface
- Interfacing switches and LEDS
- Interfacing seven segment displays
- Stepper motor
- A/D and D/A converters
- Software and hardware interrupt applications
- Intel 8251 USART architecture and interfacing
- Intel 8237a DMA controller
- Need for 8259 programmable interrupt controllers

8086 Interfacing

- Interface refers to the path for communication between two components. Interfacing is of two types, memory interfacing, and I/O interfacing.
- Memory Interfacing occurs when we need the microprocessor to access the memory for reading instruction codes and the data stored in the memory.
- There are various communication devices like the keyboard, mouse, printer, etc. So, we need to interface the keyboard and other devices with the microprocessor by using latches and buffers. This type of interfacing is known as I/O interfacing.

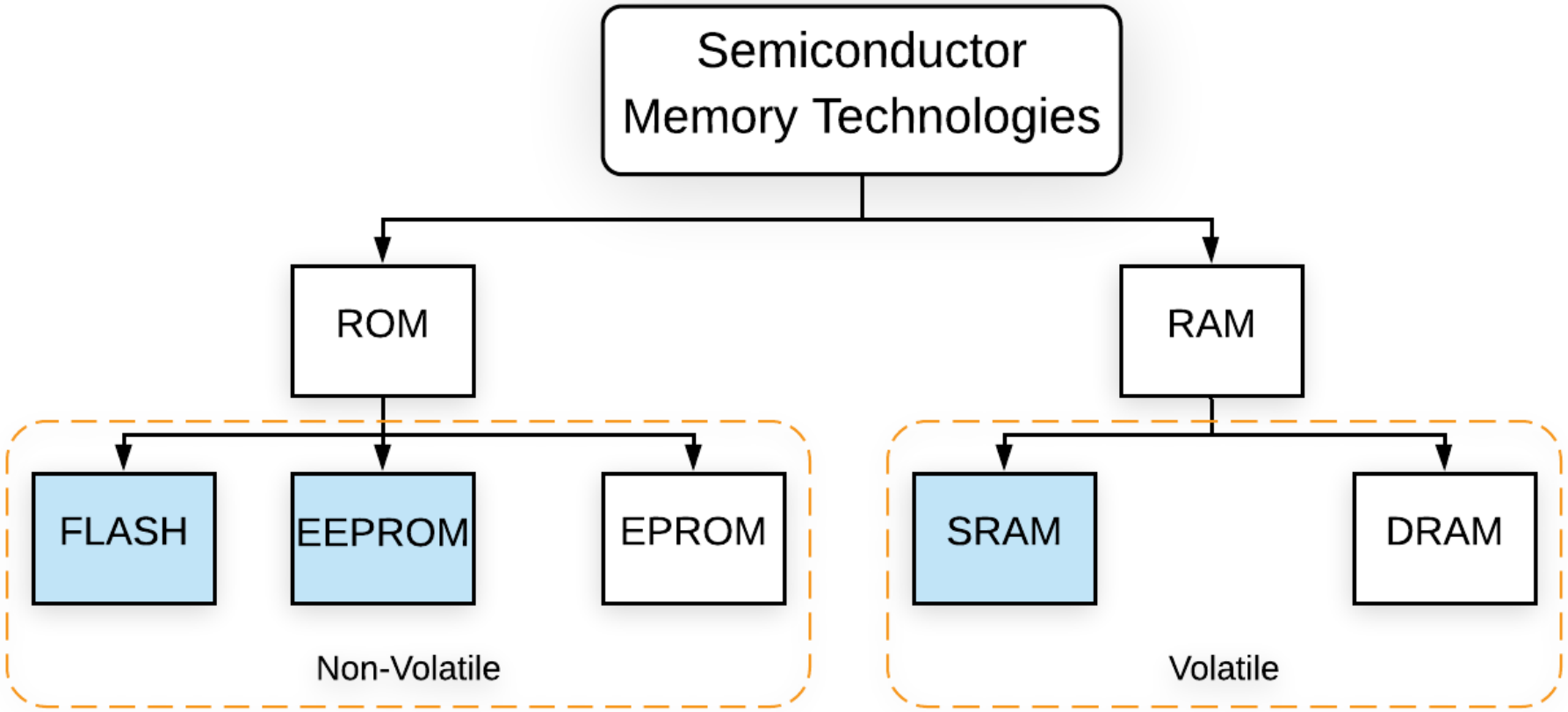
Block Diagram of Memory and I/O Interfacing

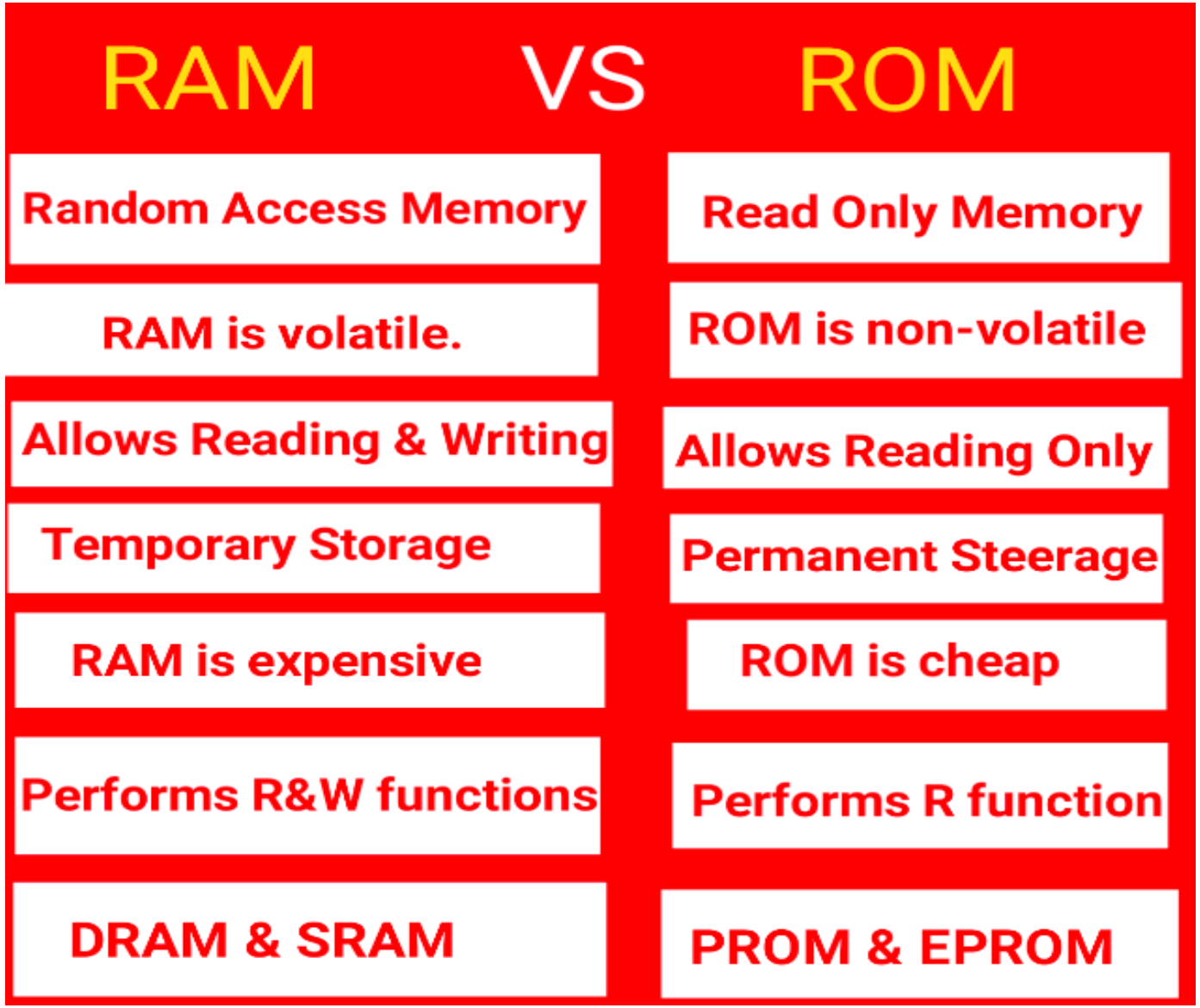


Semiconductor memories interfacing (RAM, ROM)

- Memory is simply a device that can be used to store the information i.e Programs and data.
- Semiconductor memory is a digital electronic semiconductor device used for digital data storage, such as computer memory.
- It typically refers to MOS memory, where data is stored within metal–oxide–semiconductor (MOS) memory cells on a silicon integrated circuit memory chip.

Types of Semiconductor Memories:




RAM
VS
ROM
Random Access Memory
Read Only Memory
RAM is volatile.
ROM is non-volatile
Allows Reading & Writing
Allows Reading Only
Temporary Storage
Permanent Steerage
RAM is expensive
ROM is cheap
Performs R&W functions
Performs R function
DRAM & SRAM
PROM & EPROM

Types of ROM

1. Programmable Read Only Memory (PROM)

- Empty of data when manufactured
- May be permanently programmed by the user

2. Erasable Programmable Read Only Memory (EPROM)

- Can be programmed, erased and reprogrammed
- The EPROM chip has a small window on top allowing it to be erased by shining ultra-violet light on it
- After reprogramming the window is covered to prevent new contents being erased
- Access time is around 45 – 90 nanoseconds

Types of ROM

3. Electrically Erasable Programmable Read Only Memory (EEPROM)

- Reprogrammed electrically **without** using ultraviolet light
- Must be removed from the computer and placed in a special machine to do this
- Access times between 45 and 200 nanoseconds

4. Flash ROM

- Similar to EEPROM
- However, can be reprogrammed while still in the computer
- Easier to upgrade programs stored in Flash ROM
- Used to store programs in devices e.g. modems
- Access time is around 45 – 90 nanoseconds

Advantages of ROM

- The advantages of ROM are as follows −
- Non-volatile in nature
- Cannot be accidentally changed
- Cheaper than RAMs
- Easy to test
- More reliable than RAMs
- Static and do not require refreshing
- Contents are always known and can be verified

What is RAM?

- Ram is volatile memory, meaning it does not retain data, when the electric power is turned off or fails, so if you turn off your computer, all memory stored in RAM is lost.
- When your computer is turned on again, the BIOS reads your operating system and related files from the hard disk and loads them back into RAM.

Semiconductor RAM types

S.No	SRAM	DRAM
1.	Transistors are used to store information in SRAM.	Capacitors are used to store data in DRAM.
2.	Capacitors are not used hence no refreshing is required.	To store information for a longer time, contents of the capacitor needs to be refreshed periodically.
3.	SRAM is faster as compared to DRAM.	DRAM provides slow access speeds.
4.	These are expensive.	These are cheaper.
5.	SRAMs are low density devices.	DRAMs are high density devices.
6.	These are used in cache memories.	These are used in main memories.

Types of Semiconductor Memories:

- Main difference between memory technologies:
  - RAM
    - Data can be read and written but data stored is *volatile*, i.e. need power to retain data
  - ROM
    - Data can only be read, and the data stored is not volatile, i.e. don't need power to retain data
  - Flash
    - Data can be read and written, and data stored is not volatile, i.e. don't need power to retain data

Semiconductor Memory Interfacing

- Semiconductor memories are of two types: RAM and ROM.
- The semiconductor memories are arranged as two dimensional arrays of memory locations.
- For example, 1K X 8 memory chip contains 1024 locations and each of them is one byte wide. i.e.1024 bytes of information can be stored in that chip.
- Each location should have an address. So, there has to be certain number of address lines on the memory chips.
- If we designate it as n, then n = log2N, where N is the number of locations that could be addresses in that chip and n is no.of address lines we need to use.

No.of address lines required:

- $n = \log_2 N$, where N is the number of locations that could be addresses in that chip and n is no.of address lines we need to use.

$$2^n = N$$

- For 1K chip n would be 10
- Available memory is 2K*8 means 8 bit data line and address line, $2K = 2^1 \times 2^{10} = 2^{11}$
- So 11 address lines A10,A9,⋯A0
- For 1K chip n would be 10, for 2K it is 11, for 4k it will be 12 etc.

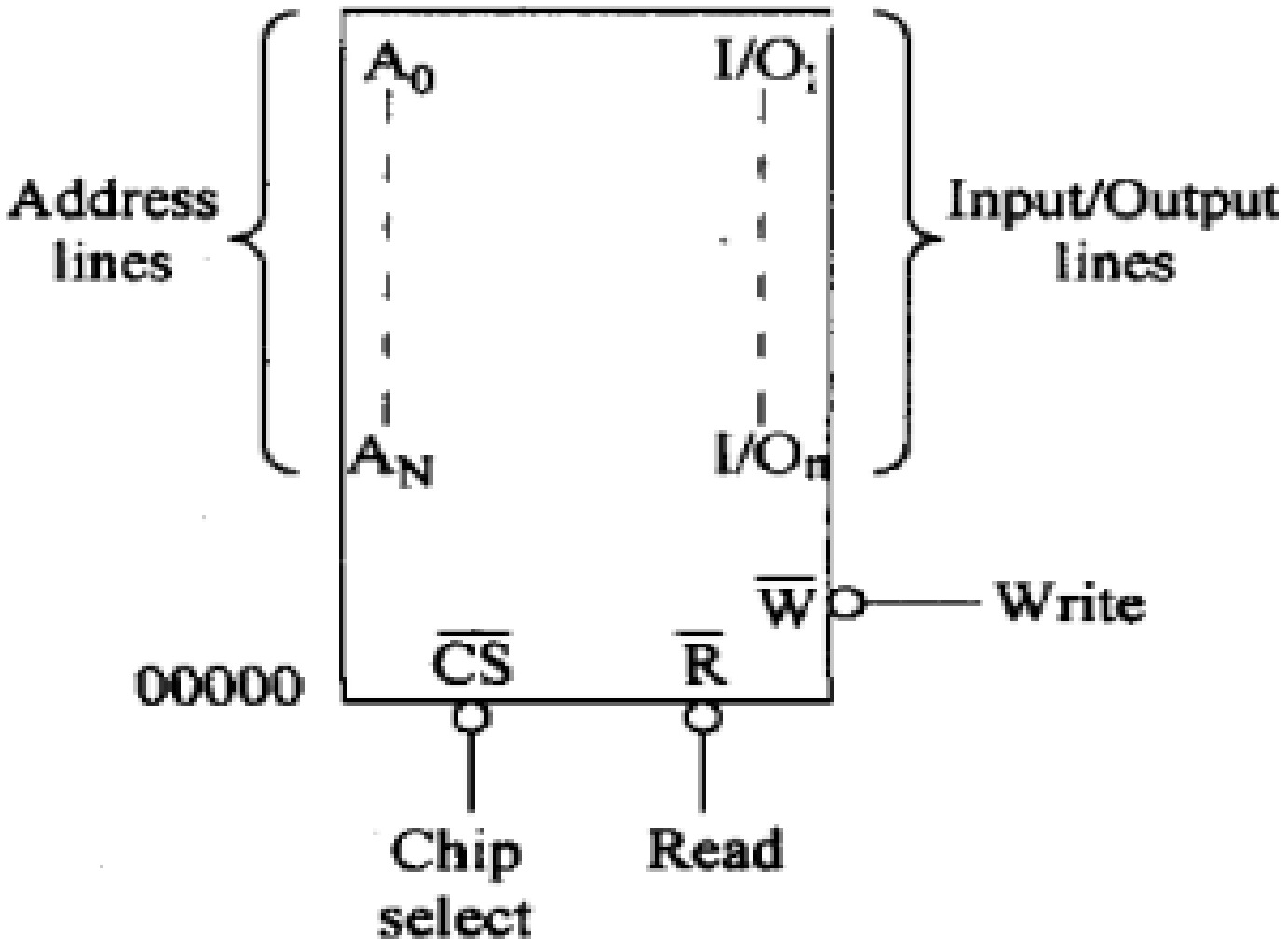


Figure 3.1.1 Schematic Representation of Memory

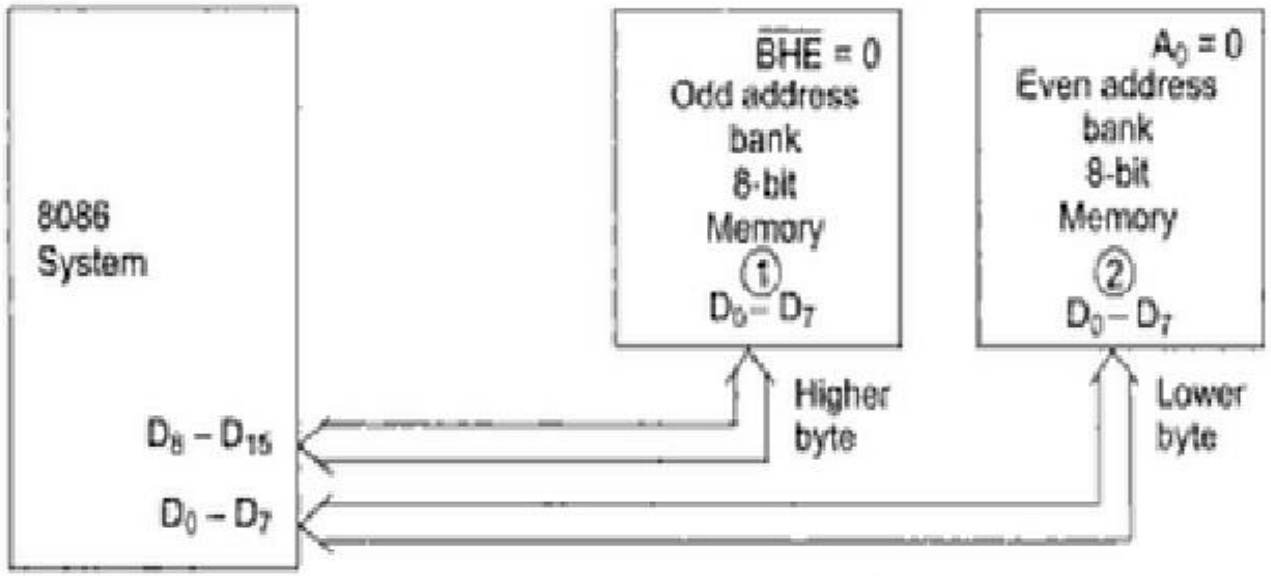


Fig. 1.7 *Physical Memory Organization*

$\overline{BHE}$	A_0	*Indication*
0	0	Whole word
0	1	Upper byte from or to odd address.
1	0	Lower byte from or to even address
1	1	None

The general procedure of static memory interfacing with 8086 is described as follows:

1. Arrange the available memory chips so that they form a 16-bit data bus width. The upper 8-bit bank called odd address memory bank and the lower 8-bit bank is called even address memory bank.

2. Connect available memory address lines with control signals like read, write of the microprocessor with those of memory chip.

3. Remaining address lines along with A0and BHE are used for decoding the required chip select signals for odd and even banks

Example:

Interface two 4K X 8 EPROMS and two 4K X 8 RAM chips with 8086.

- The address of the RAM may be selected anywhere in the 1 MB address space of 8086, but to make the address space continuous we would follow the given procedure.
- After reset the IP and CS are initialized to address FFFF0H.
- We must first calculate the total number of address lines required for 8K bytes of EPROM which is 13.

- **Problem:**

Interface two 4K×8 EPROMS and two4K×8 RAM chips with 8086.

Solution:

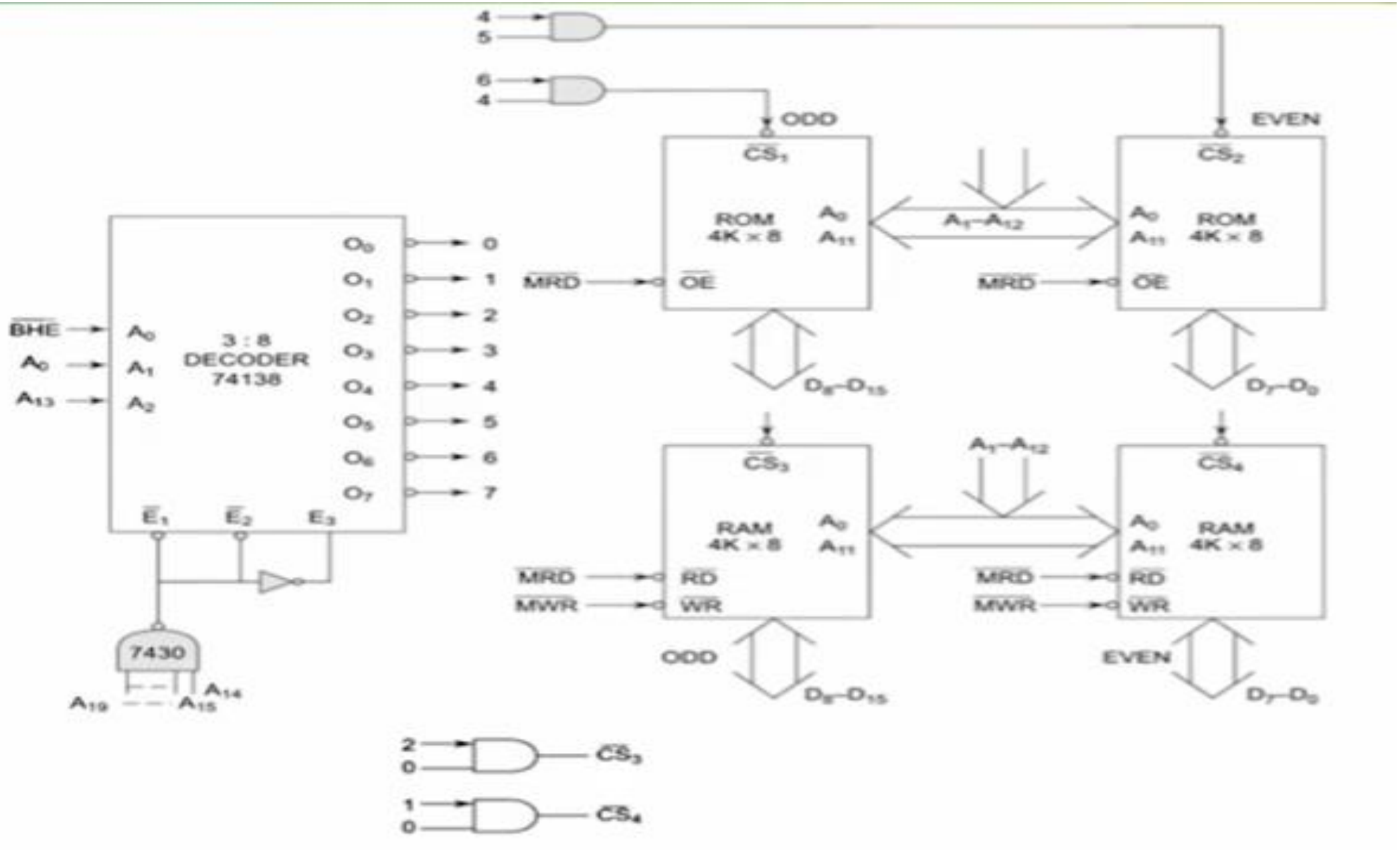
- First we have to write the memory map from the problem given.
- It will reveal the logic to be used for decoding circuit.
- Total 8K bytes of EPROM need 13 address lines A0 -A 12 (since 2^{13} = 4K).
- Since for n address lines, the number of memory locations able to address is 2^n.
- Address lines A 13 - A 19 are used for decoding to generate the chip select .

- The memory system in this example contains in total four 4K x 8 memory chips.
- The two 4K x 8 chips of RAM and ROM are arranged in parallel to obtain 16-bit data bus width.
- The BHE signal goes low when a transfer is at odd address or higher byte of data is to be accessed.
- Let us assume that the latched address. BHE and demultiplexed data lines are readily available for interfacing.
- Since the first instruction is fetched from FFFF0h after the microprocessor is reset, we will make that address to be present in EPROM and write the memory map as follows.
- And, to avoid windowing let us keep the locations to be present in the RAM as immediate addresses .
- Locations having addresses from FFFFFH to FE000H are allocated to EPROM 1 and 2.
- Immediate address map FDFFFH to FD000H is allocated to RAM1 and 2.

A 19	A 18	A 17	A 16	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
EEPROM																			
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	.	.	.	.	.	.	.	.	.	.	.	.	.
1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
EEPROM ADDRESS FFFFF – FE000																			
RAM																			
1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	0	.	.	.	.	.	.	.	.	.	.	.	.	.
1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RAM ADDRESS FDFFF – FC000																			

Memory Chip Selection

DECODER I/P	A2	A1	A0	SELECTION / COMMENT
ADDRESS / BHE'	A13	A0	BHE'	
Word transfer on D0- D15	0	0	0	Even and Odd address in RAM
Byte transfer on D0 – D7	0	0	1	Only Even address in RAM
Byte transfer on D8 – D15	0	1	0	Only Odd address in RAM
Word transfer on D0- D15	1	0	0	Even and Odd address in ROM
Byte transfer on D0 – D7	1	0	1	Only Even address in ROM
Byte transfer on D8 – D15	1	1	0	Only Odd address in ROM


4
5
6
4
ODD
EVEN
$\overline{CS_1}$
$\overline{CS_2}$
ROM
4K × 8
A_0
A_{11}
A_1–A_{12}
$\overline{MRD}$
$\overline{OE}$
D_8–O_{15}
D_7–O_0
$\overline{BHE}$
A_0
A_{13}
3 : 8
DECODER
74138
O_0
O_1
O_2
O_3
O_4
O_5
O_6
O_7
0
1
2
3
4
5
6
7
$\overline{E_1}$
$\overline{E_2}$
E_3
7430
A_{19}
A_{15}
A_{14}
$\overline{CS_3}$
$\overline{CS_4}$
RAM
4K × 8
$\overline{RD}$
$\overline{WR}$
$\overline{MWR}$
2
0
1
0
$\overline{CS_3}$
$\overline{CS_4}$

Memory

- Each memory device has at least one **chip select** (CS)or **chip enable** (CE) or **select** (S)pin that enables the memory device.
- –This enables read and/or write operations.
- Each memory device has at least one control pin.

–For ROMs, an **output enable** (OE) or **gate** (G) is present.

- The OE pin enables and disables a set of tristate buffers.

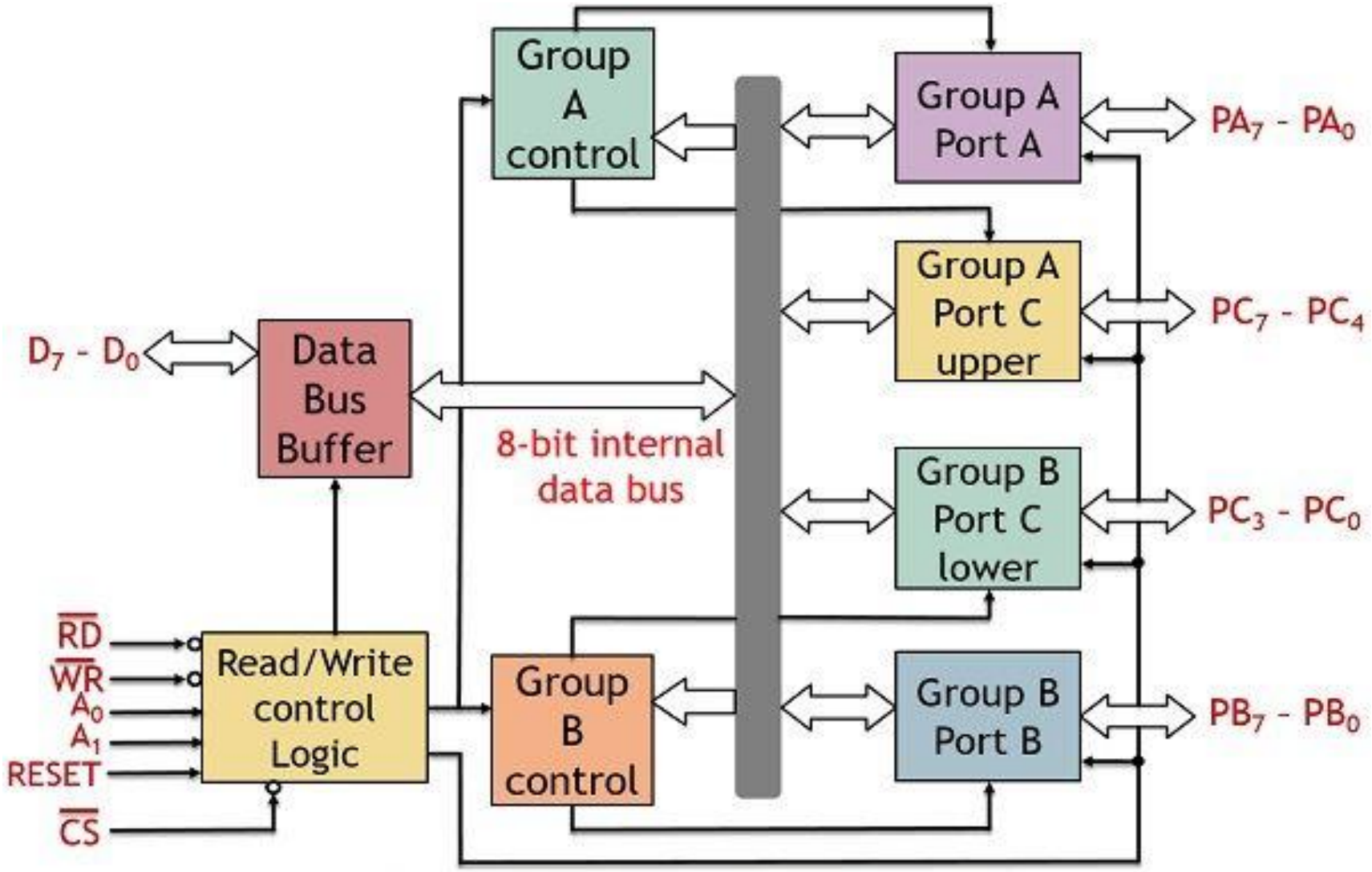
–For RAMs, a **read-write** (R/W) or **write enable** (WE) and **read enable**

- (OE) are present

Intel 8255 programmable peripheral interface

- 8255 is a programmable I/O device that acts as interface between peripheral devices and the microprocessor for parallel data transfer.
- 8255 PPI (programmable peripheral interface) can be programmed to transfer data under various conditions, from simple I/O to interrupt I/O.
- It is an important general purpose I/O device that can be used with almost any microprocessor.

Architecture of 8255



Architecture of 8255

It consists of Three sections :

1. Data Bus Buffer
2. Read/Write Control logic
3. Group A & Group B Ports-

Group A (Port A and Port C i.e., $PC_7 – PC_4$)

Group A (Port A and Port C Lower i.e., $PC_3 – PC_0$)

Data Bus Buffer

- This **three-state bi-directional 8-bit buffer** is used to interface the 8255 to the system data bus.
- Data is transmitted or received by the buffer upon execution of input or output instructions by the CPU.
- Control words and status information are also transferred through the data bus buffer.

- **Read/Write and Control Logic**
- The function of this block is to manage all of the internal and external transfers of both Data and Control or Status words. It accepts inputs from the CPU Address and Control busses and in turn, issues commands to both of the Control Groups.
- **(CS) Chip Select.** A "low" on this input pin enables the communication between the 8255 and the CPU.
- **(RD)** Read. A "low" on this input pin enables 8255 to send the data or status information to the CPU on the data bus. In essence, it allows the CPU to "read from" the 8255.
- **(WR)** Write. A "low" on this input pin enables the CPU to write data or control words into the 8255.
- RD is from CPU to 8255(input line) and WR is from 8255 to CPU(output Line)

- **(A0 and A1)** Port Select 0 and Port Select 1. These input signals, in conjunction with the RD and WR inputs, control the selection of one of the three ports or the control word register. They are normally connected to the least significant bits of the address bus (A0 and A1).

A1	A0	SELECTION
0	0	PORT A
0	1	PORT B
1	0	PORT C
1	1	CONTROL REGISTER

- **(RESET)** : It is an active high signal that shows the resetting of the PPI. A high signal at this pin clears the control registers and the ports are set in the input mode.

The table below shows the operation of the control signals:

A_1	A_0	RD'	WR'	CS'	Input/Output Operation
0	0	0	1	0	Port A - Data Bus
0	1	0	1	0	Port B - Data Bus
1	0	0	1	0	Port C - Data Bus
0	0	1	0	0	Data Bus - Port A
0	1	1	0	0	Data Bus - Port B
1	0	1	0	0	Data Bus - Port C
1	1	1	0	0	Data Bus - Control register

Group A & Group B Ports

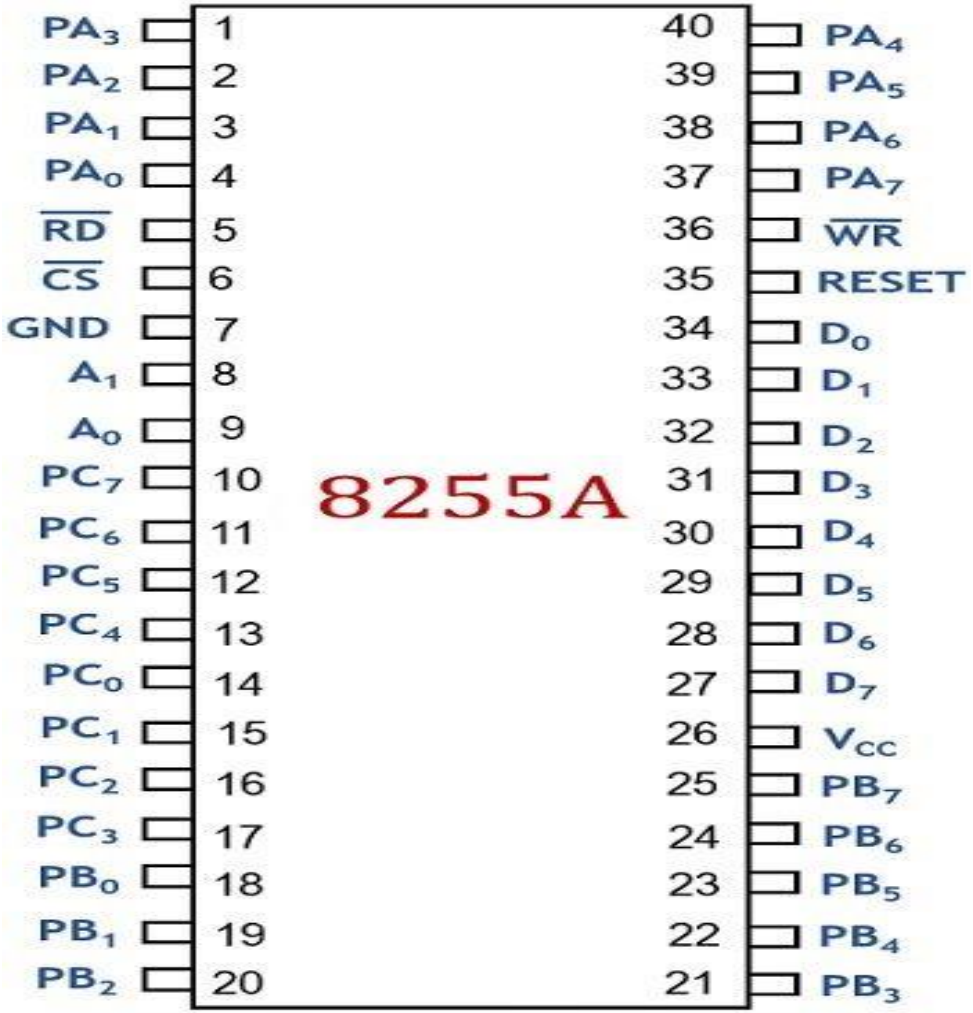
- It has Three ports **Port A, Port B and Port C** and as each port has 8 lines.
- Two Groups: 12-bits each

Group A - Port A and Port C Upper (i.e., $PC_7 – PC_4$)

Group A - Port A and Port C Lower (i.e., $PC_3 – PC_0$)

  - Port A – 8-bit buffered I/O Latch and can be programmed by mode0,mode1 & mode 2
  - Port B – 8-bit buffered I/O Latch and can be programmed by mode 0 and mode 1
  - Port C – 8-bit unlatched buffered I/O ,split into 2 parts and programmed by bit set/ reset mode

Pin diagram of 8255



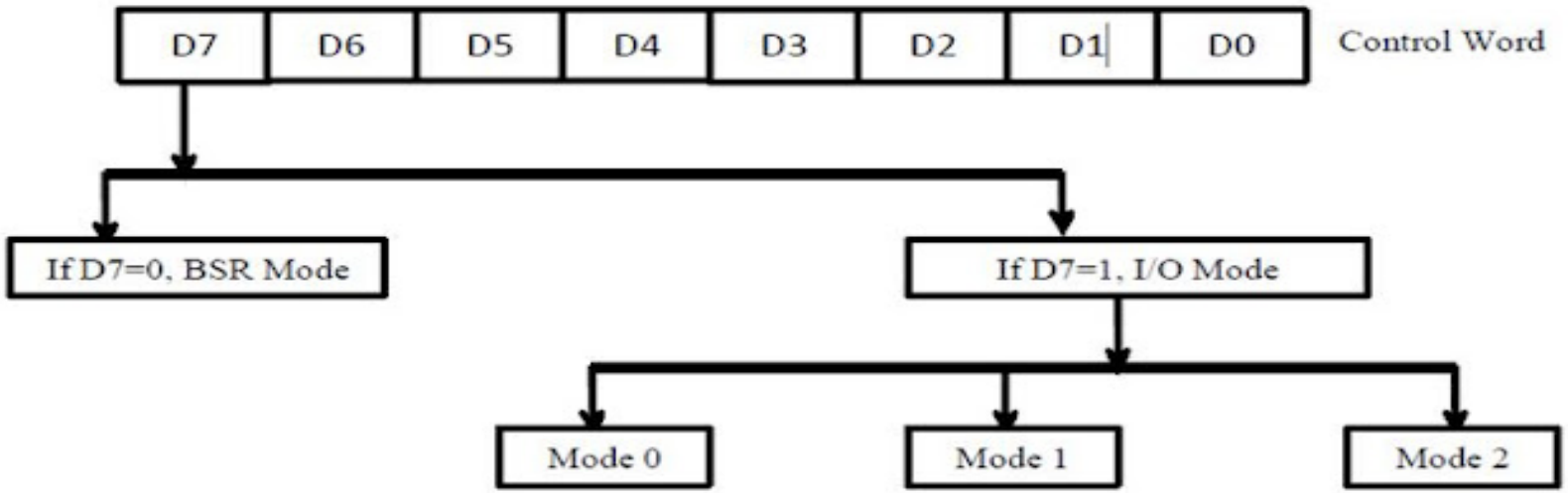
The external voltage required to drive the circuit is provided at pin number 26 i.e., VCC. Also pin number 7 is the ground connection of the circuit.

Out of 40 pins, 24 pins are allotted to I/O ports. Rest of the pins are allotted to the signals discussed above.

Modes of Operation-8255

The 8255A is primarily operated in two modes:

1. **The BSR (Bit-Set-Reset) mode**
2. **I/O (input-output) mode** : The I/O mode is further grouped into 3
   - Mode 0 - Simple I/O interfacing
   - Mode 1 – Handshaking I/O (Strobed I/O)
   - Mode 2 – Bi-Directional Handshaking I/O (Strobed bi-directional I/O)



When D7=0, BSR mode

- For port C
- No effect on I/O mode and functions of port A and B
- Individual bits of port C can be used for applications such as ON/OFF switch

When D7=1, I/O mode

i) **Mode 0**

- Simple I/O interfacing for port A, B and C

ii) **Mode 1**

- Interfacing with handshake signals for port A and B
- Port C bits are used for handshake

iii**) Mode 2**

- Bidirectional I/O interfacing for port A
- Port B: either in mode 0 or mode 1
- Port C bits used for handshake

(a) BSR mode :

The Bit Set/ Reset mode is used to Set / Reset the individual bits of PORT-C

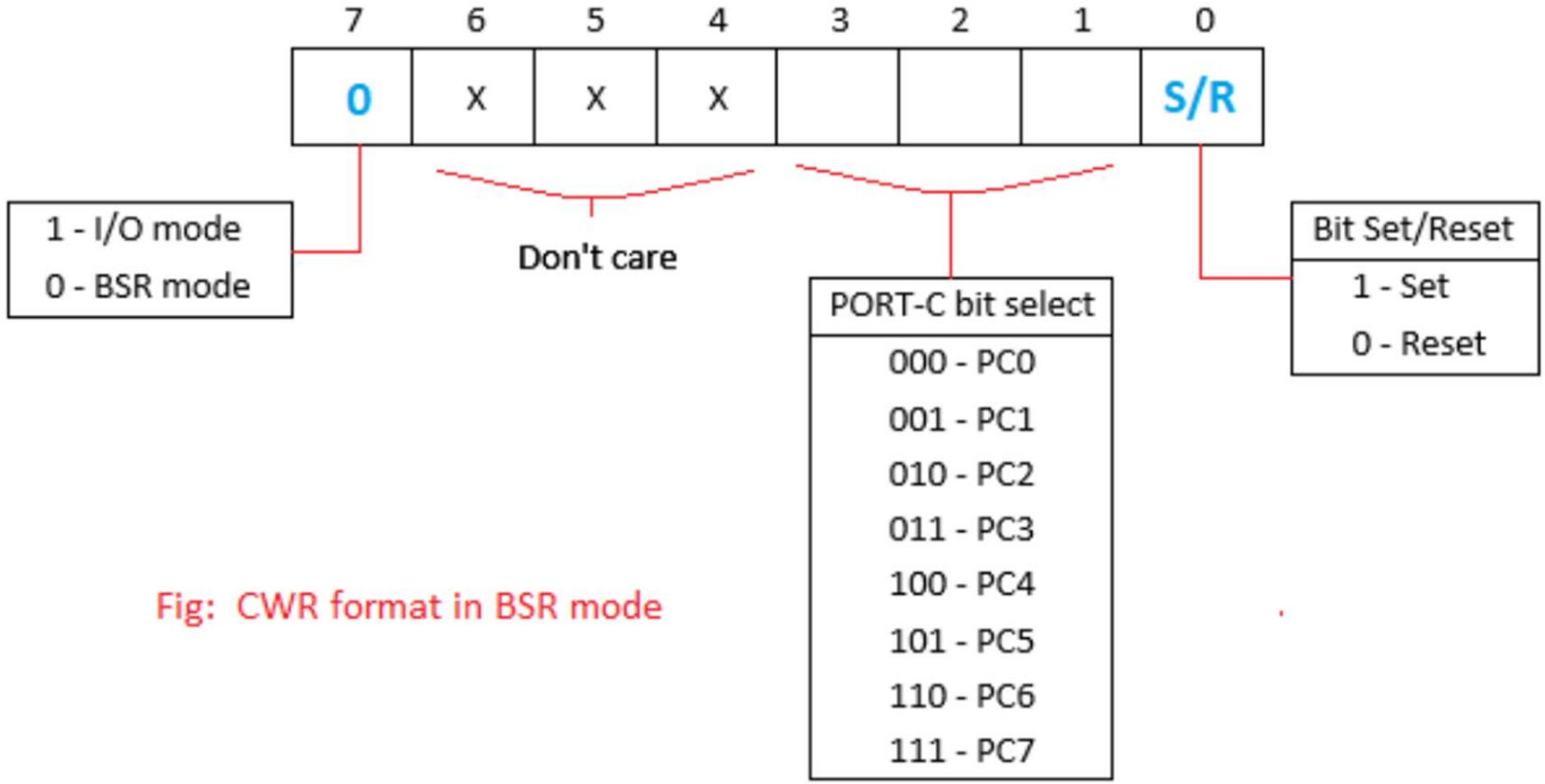


Fig: CWR format in BSR mode

(b) I/O mode :

- I/O mode is used for sending and receiving data from I/O devices
- In this mode each Port is configured as either input (or) output port

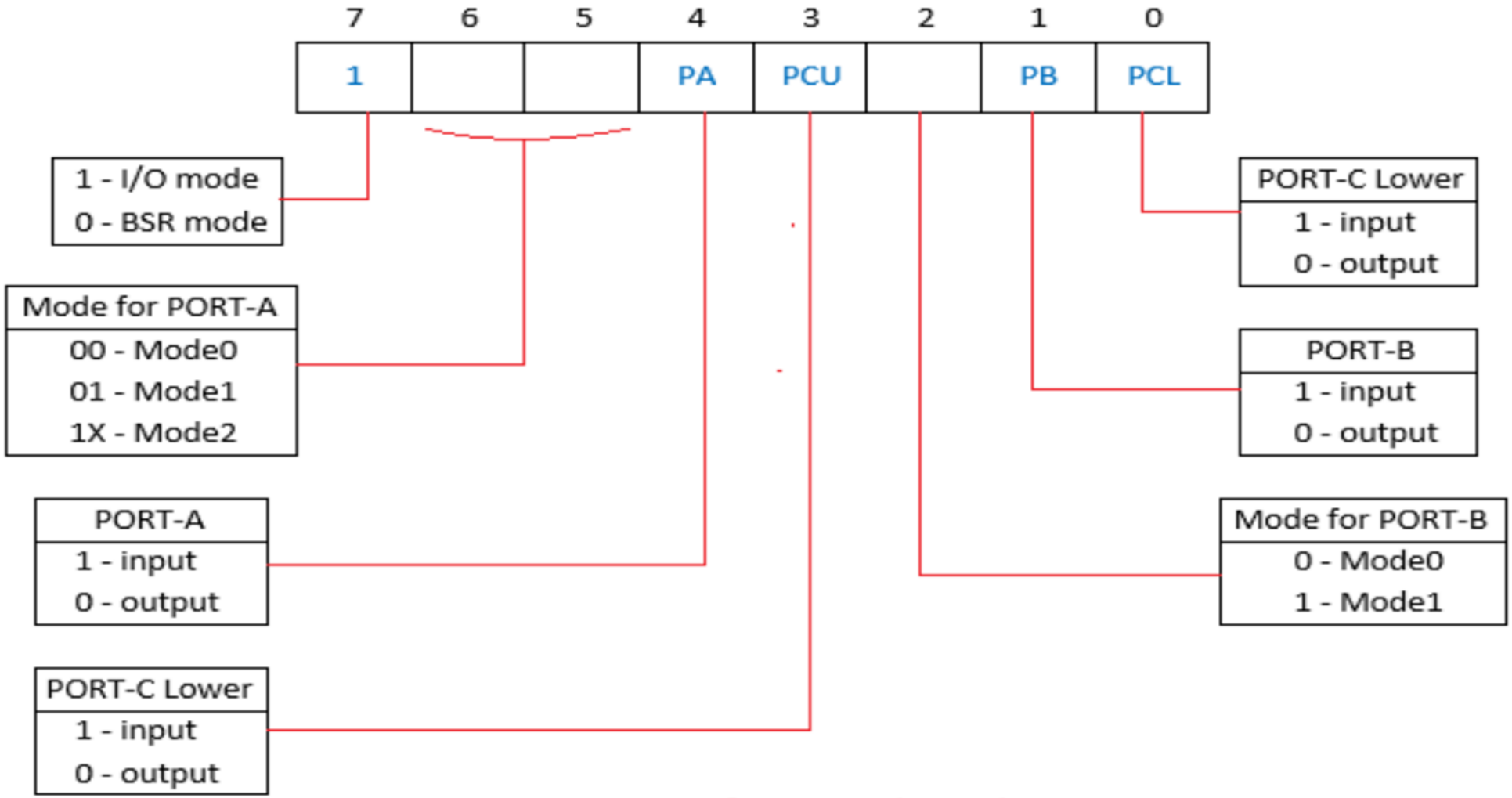
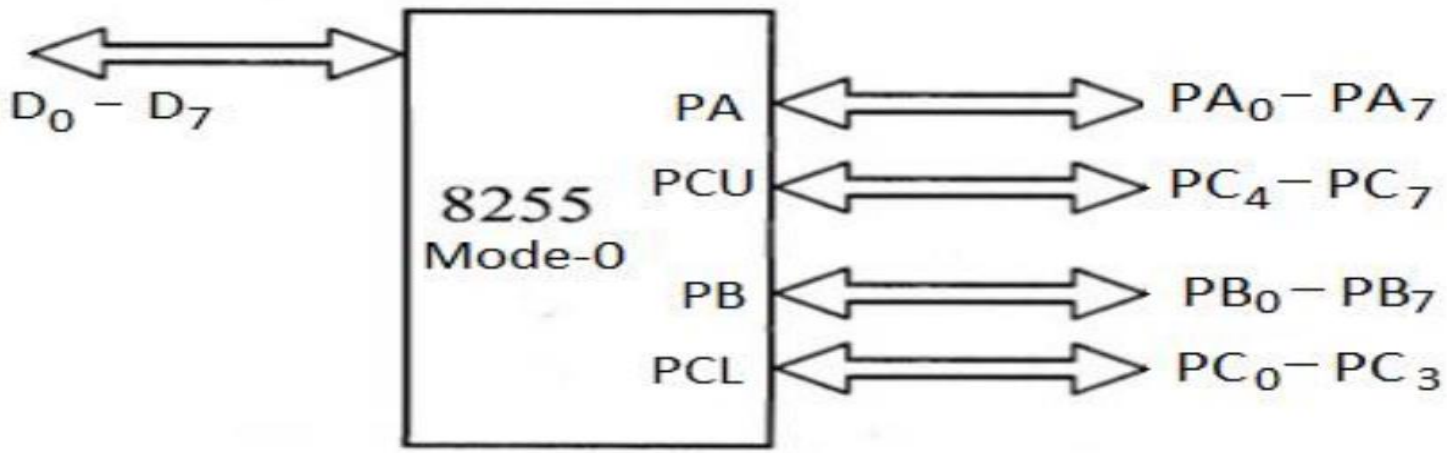


Fig: CWR format in I/O mode

I/O modes :

Mode-0 : Simple I/O → PORT-A, PORT-B & PORT-C
Mode-1 : Handshaking I/O → PORT-A, PORT-B
Mode-2 : Bi-directional Handshaking I/O → only PORT-A can be operated

(i) Mode-0 : Simple I/O



- This mode provides simple input and output capability.
- All 3- ports can be operated in Mode-0
  - PORT-A → 8-bit I/O port
  - POTR-B → 8-bit I/O port
  - PORT-C → Two 4-bit I/O ports → PORT-C upper & PORT-C lower
- Each port can be programmed to function as either input (or) output port
- This mode doesn't support handshaking and interrupt capability

Mode-1 : Handshaking I/O

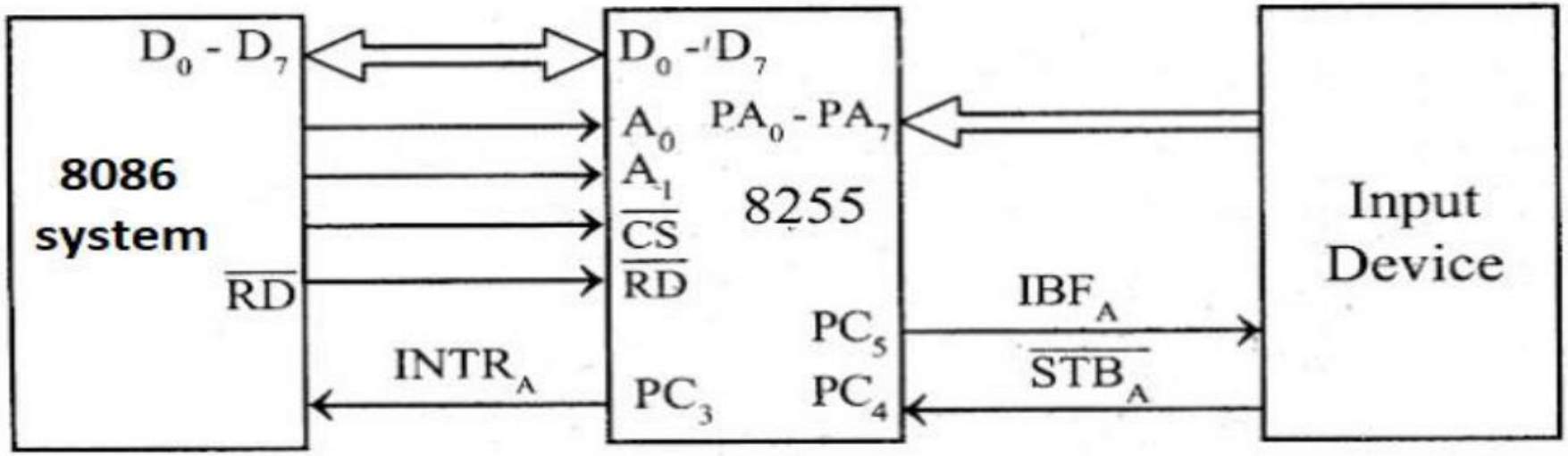
- In this mode, I/O data transfer is controlled by handshaking signals
- Only PORT-A & PORT-B can be operated in Mode-1
- Each port uses 3-lines from PORT-C as Handshaking signals.

  For PORT-B → PC_0 , PC_1 , PC_2 are used as Handshaking lines

  For PORT-A → input port → PC_3 , PC_4 , PC_5

  output port → PC_3 , PC_6 , PC_7
- The remaining 2-line of PORT-C can be used as I/O lines
- This mode supports Interrupt logic.

Input operation :

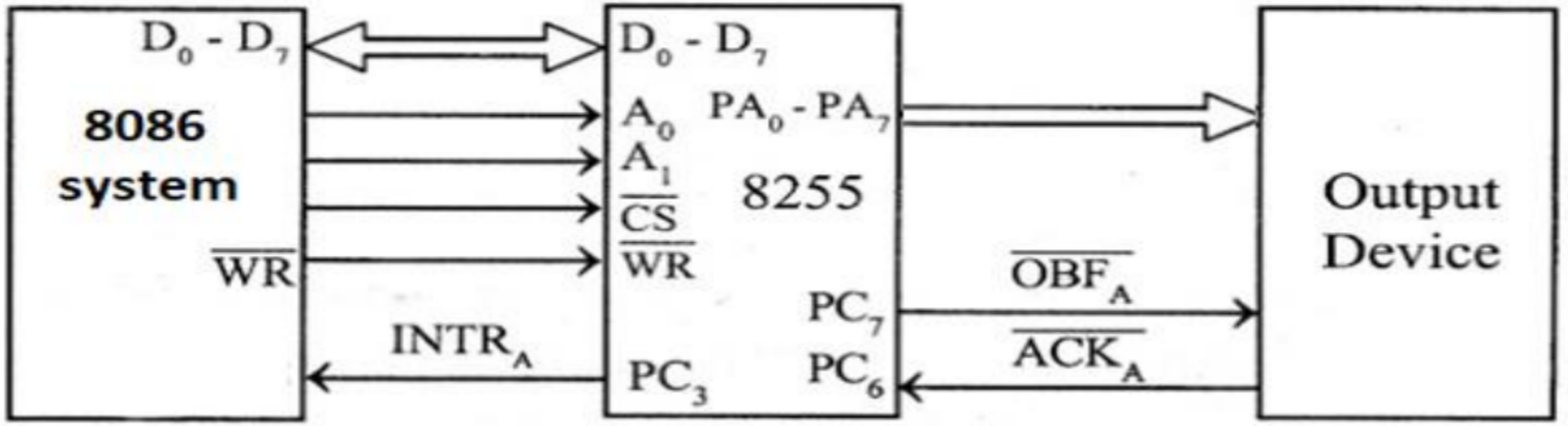


$\overline{\text{STB}}$ (Strobe) : It is active low input signal from input device.
It indicates the 8255 that the data is placed on the port lines.

IBF (Input Buffer Full) : It is active high output signal from 8255 to input device.
It indicates the input device that the input buffer is full and it is not ready to accept next byte from input device. This signal is deactivated when CPU reads data from input port of respective port, i.e., at the rising edge of RD signal

INTR (Interrupt Request) : It is active high output signal from 8255 to CPU.
It indicated the CPU that the data from input device is available in input buffer.
It is set when IBF is active and it is reset when RD signal is issued by CPU

Output operation:



$\overline{\text{OBF}}$ (Output Buffer Full) : It is active low output signal from 8255 to output device.
It indicates the output device that the data is available on the port lines.
Upon the activation of the OBF signal, the output device reads data from port lines

$\overline{\text{ACK}}$ (Acknowledge) : It is active low input signal from output device.
It indicates the 8255 that the data from port lines has been received by output device

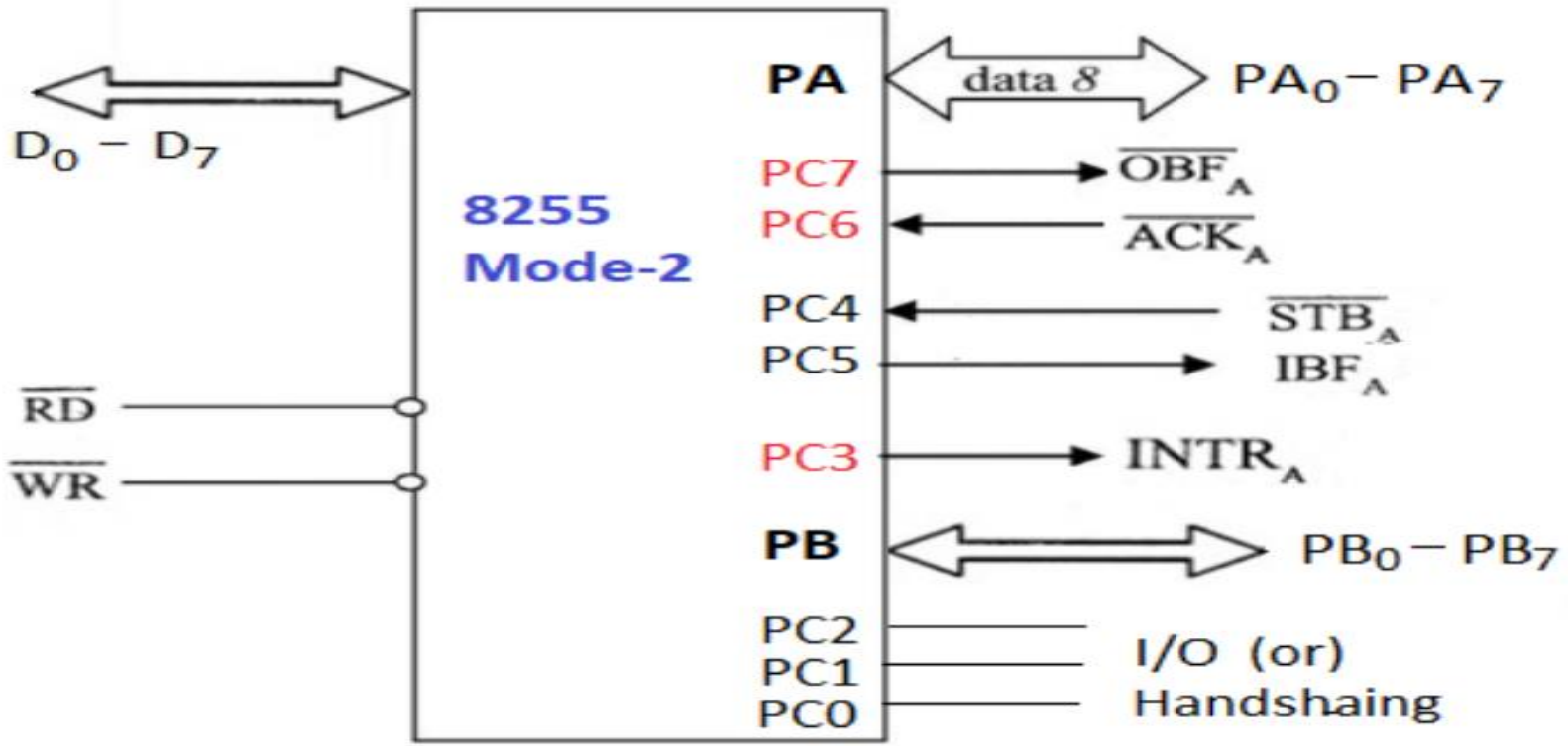
INTR (Interrupt Request) : It is active high output signal from 8255 to CPU.
It indicated the CPU that the output device is ready to accept next data byte.
It is set when ACK signal is active and it is reset when WR signal is issued by CPU

Mode-2 : Bi-directional Handshaking I/O

- Only PORT-A can be operated in Mode-2 as bi-directional handshaking I/O port
- The Five PORT-C lines PC_3 - PC_7 are used as Handshake signals

  PORT-A → input port → PC_3 , PC_4 , PC_5

  output port → PC_3 , PC_6 , PC_7
- The PORT-B can be operated in Mode-0 (or) Mode-1.
- When PORT-B is operated in Mode-1, the PORT-C lines PC_0 , PC_1 , PC_2 are used as Handshaking signals. Otherwise, these lines can be used as simple I/O lines



Input operation :

$\overline{\text{STB}}$ (Strobe) : It is active low input signal from input device.
It indicates the 8255 that the data is placed on the port lines.

IBF (Input Buffer Full) : It is active high output signal from 8255 to input device.
It indicates the input device that the input buffer is full and it is not ready to accept next byte from input device. This signal is deactivated when CPU reads data from input port of respective port, i.e., at the rising edge of RD signal

INTR (Interrupt Request) : It is active high output signal from 8255 to CPU.
It indicated the CPU that the data from input device is available in input buffer.
It is set when IBF is active and it is reset when RD signal is issued by CPU

Output operation:

$\overline{\text{OBF}}$ (Output Buffer Full) : It is active low output signal from 8255 to output device.
It indicates the output device that the data is available on the port lines.
Upon the activation of the OBF signal, the output device reads data from port lines

$\overline{\text{ACK}}$ (Acknowledge) : It is active low input signal from output device.
It indicates the 8255 that the data from port lines has been received by output device

INTR (Interrupt Request) : It is active high output signal from 8255 to CPU.
It indicated the CPU that the output device is ready to accept next data byte.
It is set when ACK signal is active and it is reset when WR signal is issued by CPU

To communicate with peripherals through 8255 three steps are necessary:

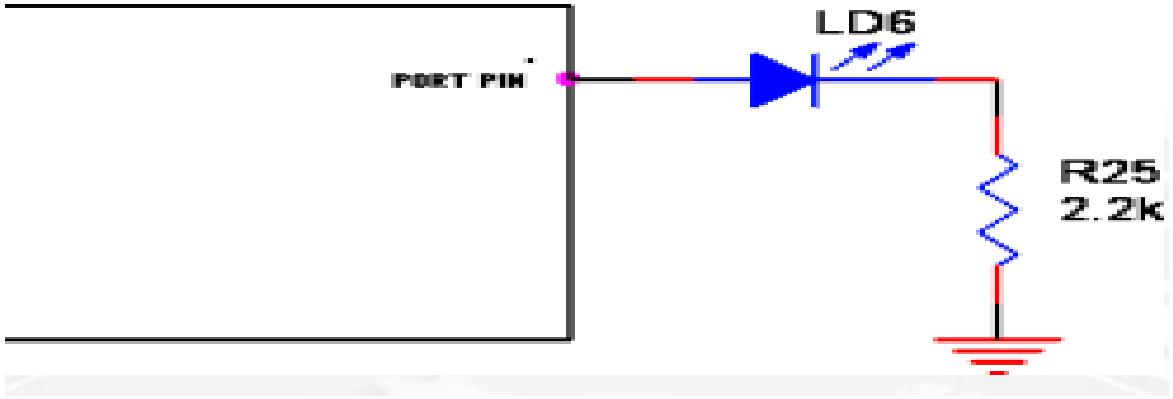
- Determine the addresses of Port A, B, C and Control register according to Chip Select Logic and the Address lines A0 and A1.
- Write a control word in control register.
- Write I/O instructions to communicate with peripherals through port A, B, C.

The common applications of 8255 are:

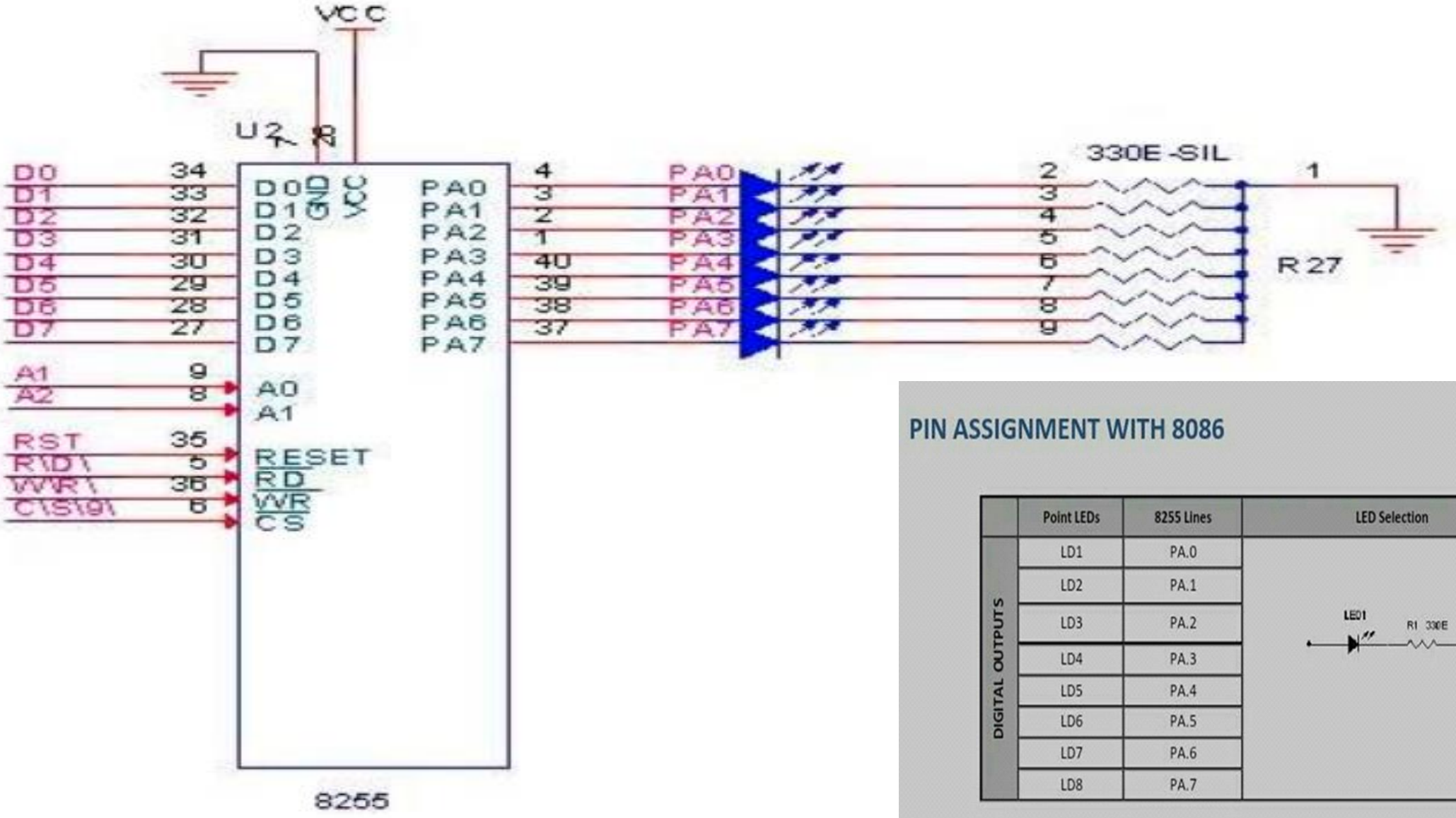
- Traffic light control
- Generating square wave
- Interfacing with DC motors and stepper motors

Interfacing LEDS

- Anode is connected through a resistor to GND & the Cathode is connected to the Microprocessor pin as shown in Fig.
- When the Port Pin is HIGH, the LED is OFF & when the Port Pin is LOW the LED is turned ON.
- We now want to flash a LED. It works by turning ON a LED & then turning it OFF & then looping back to START.
- A delay is generated between the flashing of LEDs.



Interfacing LEDS



PIN ASSIGNMENT WITH 8086

	Point LEDs	8255 Lines	LED Selection
DIGITAL OUTPUTS	LD1	PA.0	
	LD2	PA.1	
	LD3	PA.2	
	LD4	PA.3	
	LD5	PA.4	
	LD6	PA.5	
	LD7	PA.6	
	LD8	PA.7	

ASSEMBLY PROGRAM TO TURN ON AND OFF LEDS USING 8086

1	0	0	0	0	0	0	0

CWR=1000 0000 = 80H
PORT-A as OUTPUT PORT

```
MOV AL,80H
MOV DX,ADD-CWR
MOV DX,AL

UP: MOV AL,#00H
        MOV DX,ADD-PORTA
        MOV DX,AL
        CALL DELAY
        MOV AL,#FFH
        MOV DX,AL
        CALL DELAY
        JMP UP
DELAY: MOV CX,#FFFFH
P0: DEC CX
    JNZ P0
RET
```

Interfacing switches and LEDS

Problem.1

Interface an 8255 with 8086 to work as an I/O port. Initialize port A as output port, port B as input port and port C as output port. Port A address should be 0740H. Write a program to sense switch positions SW_0-SW_7 connected at port B. The sensed pattern is to be displayed on port A, to which 8 LEDs are connected, while the port C lower displays number of on switches out of the total eight switches.

Solution The control word is decided upon as follows:

B7	B_6	B_5	B_4	B_3	B_2	B_1	B_0	Control word
1	0	0	0	0	0	1	0	= 82H
I/O mode	Port A in mode 0		Port A,o/p	Port C,o/p	Port B,mode 0	Port B,i/p	Port C,o/p	

Thus 82H is the control word for the requirements in the problem. The port address decoding can be done as given below. The 8255 is to be interfaced with lower order data bus, i.e. D_0-D_7. The A_0 and A_1 pins of 8255 are connected to A_{01} and A_{02} pins of the microprocessor respectively. The A_{00} pin of the microprocessor is used for selecting the transfer on the lower byte of the data bus. Hence any change in the status of A_{00} does not affect the port to be selected, rather A_{01} and A_{02} of the microprocessor decide the port to be selected as they are connected to A_0 and A_1 of 8255. The 8255 port addresses are tabulated as shown below.

8255	*I/O Address lines*																*Hex. Port*
Ports	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_{09}	A_{08}	A_{07}	A_{06}	A_{05}	A_{04}	A_{03}	A_{02}	A_{01}	A_{00}	*Addresses*
PortA	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

8255	*I/O Address lines*																*Hex. Port*
Ports	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_{09}	A_{08}	A_{07}	A_{06}	A_{05}	A_{04}	A_{03}	A_{02}	A_{01}	A_{00}	*Addresses*
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

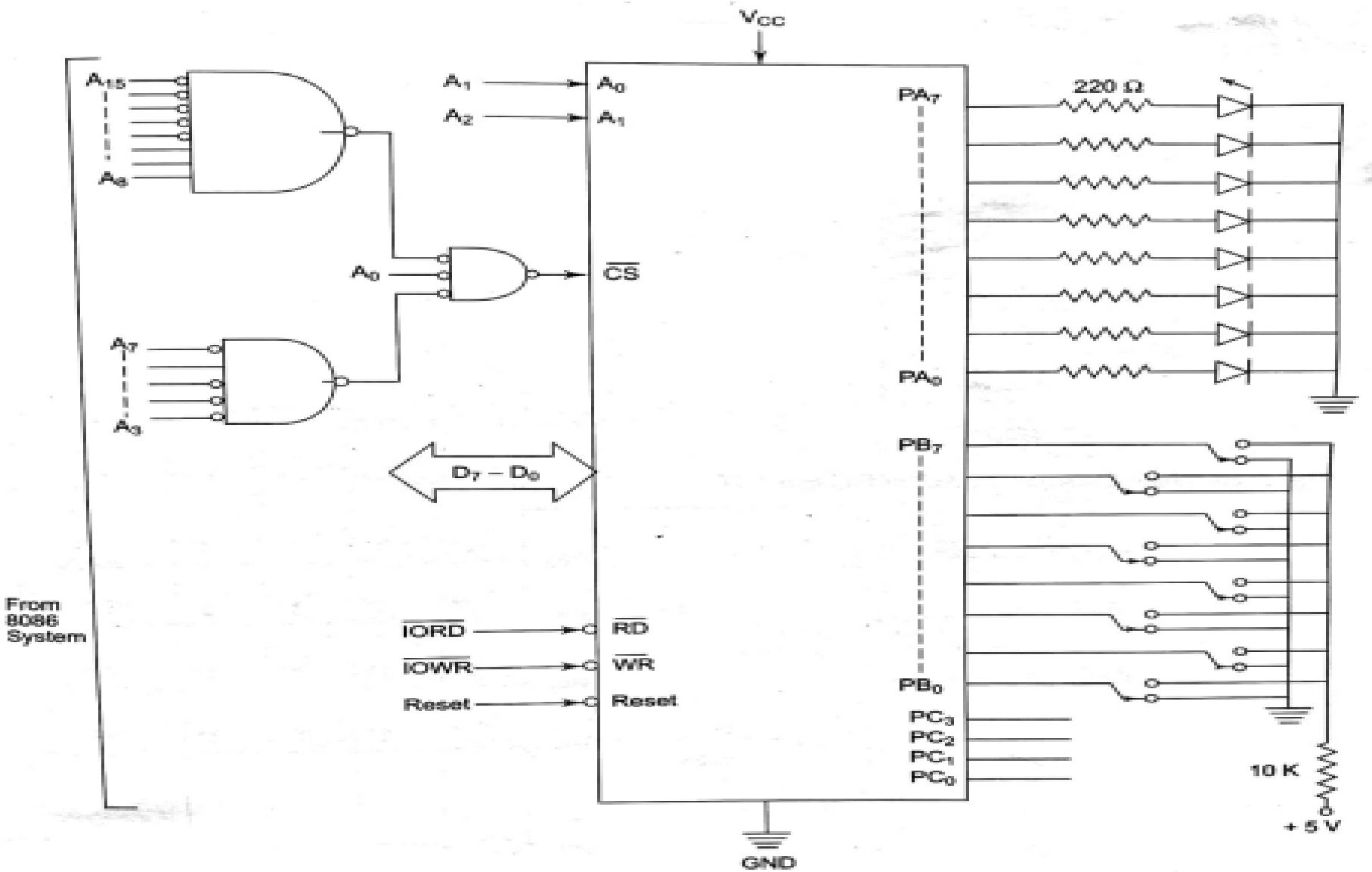


Fig. 5.19 *8255 Interfacing with 8086 for Problem 5.10*

```
        MOV DX, 0746 H  ; Initialise CWR with
        MOV AL, 82 H    ; control word 82H
        OUT DX, AL      ;
        SUB DX,04       ; Get address of port B in DX
        IN AL, DX       ; Read port B for switch
        SUB DX,02       ; positions in to AL and get port A address
                        ; in DX.
        OUT DX, AL      ; Display switch positions on port A
        MOV BL, 00 H    ; Initialise BL for switch count
        MOV CH, 08H     ; Initialise CH for total switch number
YY:     ROL AL          ; Rotate AL through carry to check,
        JNC XX          ; whether the switches are on or
        INC BL          ; off, i.e. either 1 or 0
XX :    DEC CH          ; Check for next switch. If
        JNZ YY          ; all switch are checked, the
        MOV AL, BL      ; number of on switches are
        ADD DX, 04      ; in BL.Display it on port C
        OUT DX,AL       ; lower.
        HLT             ; Stop
```

Interfacing seven segment displays

Seven Segment Displays are used in a number of systems to display the numeric information.

The seven segments can display one digit at a time.

Thus the no. of segments used depends on the no. of digits in the number to be displayed.

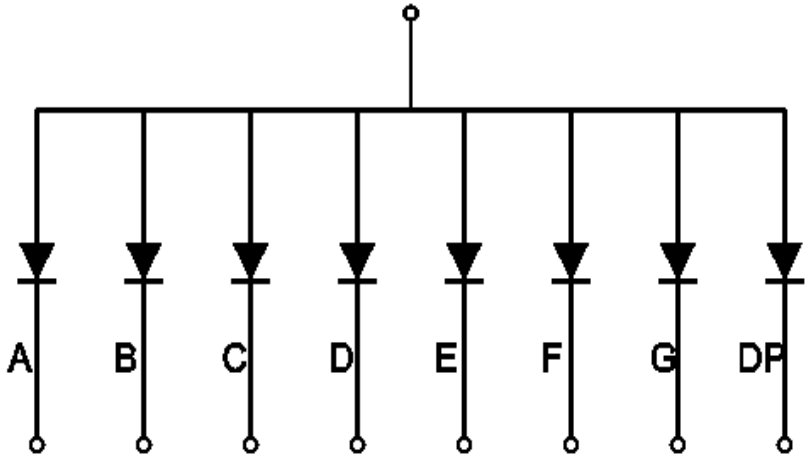
A seven segment consists of eight LEDs which are aligned in a manner so as to display digits from 0 to 9 when proper combination of LED is switched on.

Seven segment uses seven LED’s to display digits from 0 to 9 and the eighth LED is used for the dot.

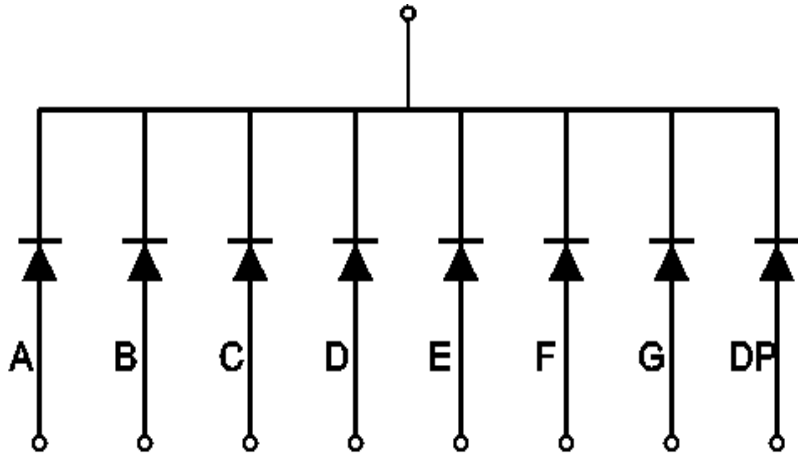
A typical seven segment looks like as shown in the figure below

- **Seven-segment display** is a form of electronic display used for displaying alpha-numeric characters. A seven-segment display is a set of seven LEDs elements, arranged to form a figure of 8. Each of the LEDs is turned ON and OFF and the combination of LEDs which are ON forms a character. If all elements are activated, the display shows a numeral 8. Numbers from 0-9 and few alphabets can be displayed.

  The working of 7 segment display is on the similar lines as that of and LED as the 7 segment displays is eventually made up of 7 LEDs and an LED for the decimal point. The use of the decimal point is to display decimal numbers like 3.1 or 7.5. The 7 segment display are of two types viz. Common Anode display and Common Cathode display.



common anode



common cathode

All of the cathodes (negative terminals) or all of the anodes (positive terminals) of the segment LEDs are connected and brought out to a common pin; this is referred to as a "common cathode" or "common anode" device. Hence a 7 segment plus decimal point package will only require nine pins. Common cathode implementations require logic low (0) to turn on a segment, common anode implementations require logic high (1) to turn on a segment.

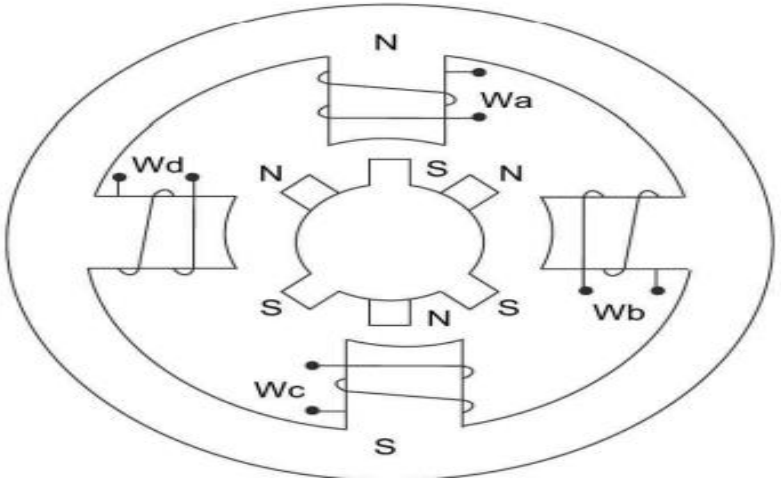
Stepper Motor Interfacing

- A stepper motor is a device used to obtain an accurate position control of rotating shafts
- It employs rotation of its shaft in terms of steps rather than continuous rotation as in case of AC or DC motors
- To rotate the shaft of the stepper motor, a sequence of pulses is applied to the windings of the stepper motor in a proper sequence
- The number of pulses required for one complete rotation of the shaft of the stepper motor is equal to its number of internal teeth on the rotor
- The stator teeth and the rotor teeth lock with each other to fix a position of the shaft.
- With a pulse applied to the winding input, the rotor rotates by one

tooth position or an angle x

- $X=360^0$/no. of rotor teeth

- After the rotation of the shaft through angle x, the rotor locks itself with the next tooth in the sequence on the internal surface of stator
- Fig below shows the internal schematic of a four winding stepper motor.
- Stepper motors have been designed to work with digital circuits.
- Binary level pulses of 0-5v are required at its winding inputs to obtain the rotation of shafts
- The sequence of pulses can be decided, depending upon the required motion of the shaft.



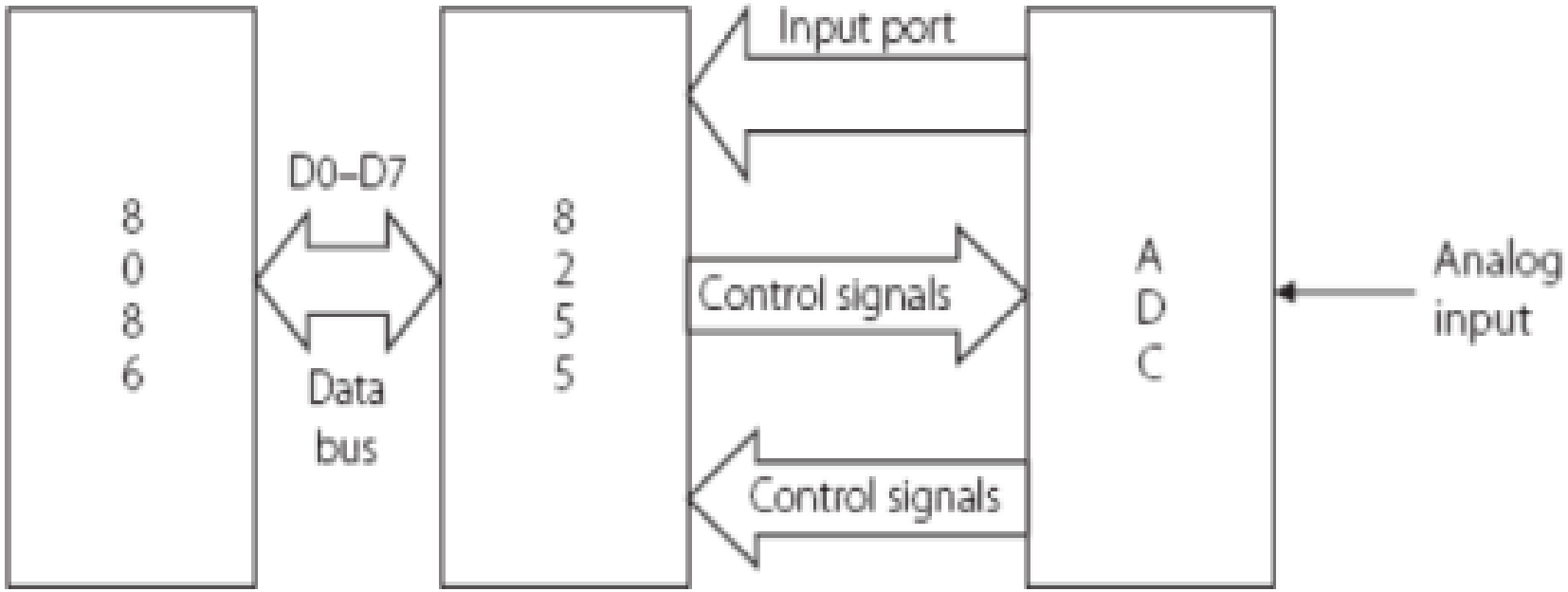
- Design a stepper motor controller and write an ALP to rotate shaft of a 4-phase stepper motor
- (i) in clockwise 5 rotations
- (ii) in anticlockwise 5 rotations
- The 8255 port A address is 0740h. The stepper has 200 rotor teeth
- The port A bit PA0 drives winding Wa, PA1 drives Wb and so on.
- The stepper motor has an initial delay of 10 msec. Assume that the routine for this delay is already available

A/D CONVERTERS

- In most of the cases, the PIO 8255 is used for interfacing the analog to digital converters with microprocessor.
- The analog to digital converters is treaded as an input device by the microprocessor, that sends an initializing signal to the ADC to start the analogy to digital data conversation process.
- The start of conversation signal is a pulse of a specific duration.
- The process of analog to digital conversion is a slow process, and 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.

ADC (Analog-to Digital Converter)

Interface with 8086



Control Signals:

SOC: The start of conversation signal

EOC : End of conversion signal

ADC INTERFACING

- A general algorithm for ADC interfacing contains the following steps :

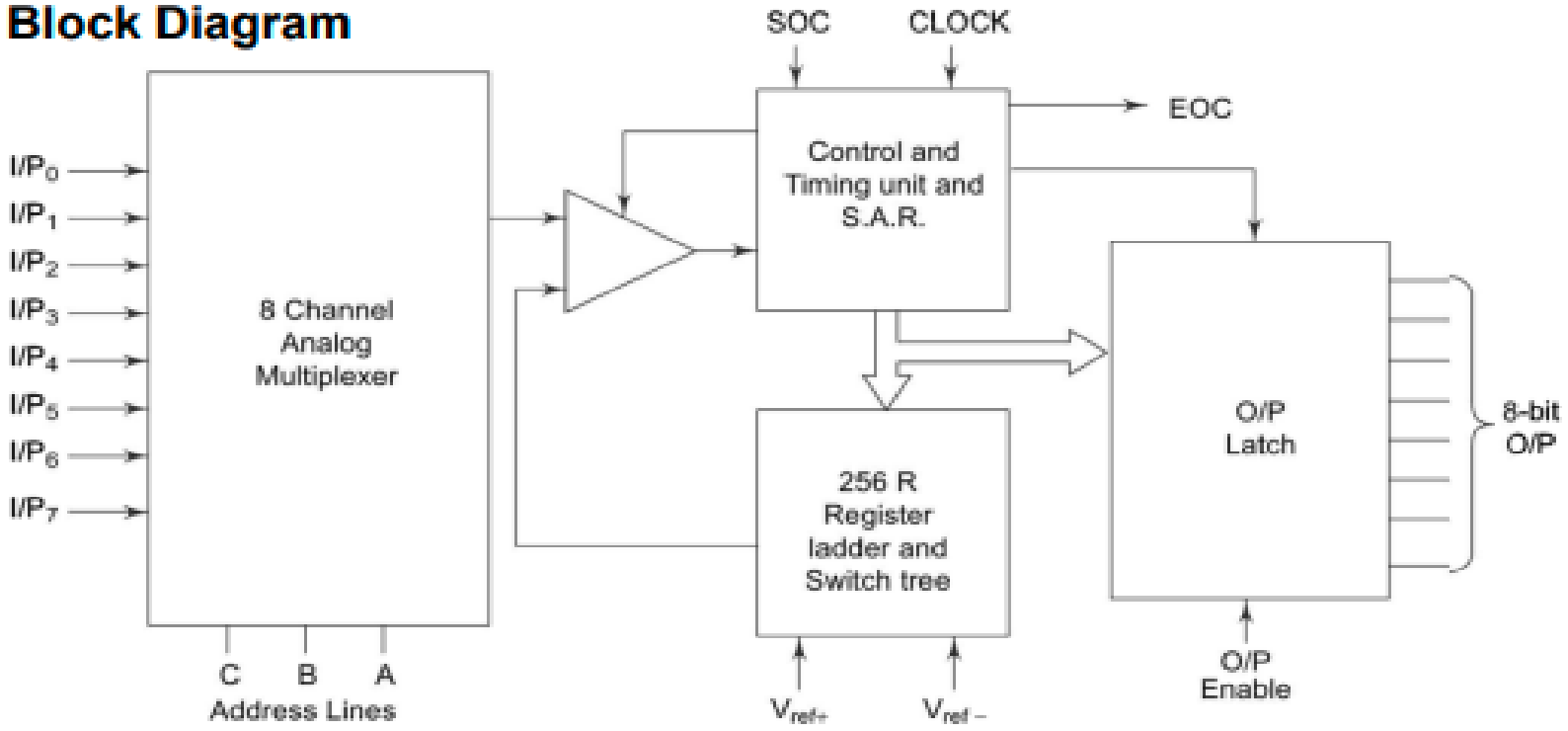
1. Ensure the stability of analog input, applied to the ADC.
2. Issue start of conversion (SOC) pulse to ADC.
3. Read end of conversion (EOC) signal to mark the end of conversion process.
4. Read digital data output of the ADC as equivalent digital output.

ADC 0808/0809

- These are unipolar analog to digital converters, i.e. they are able to convert only positive analog input voltages to their digital equivalents.
- These chips do not contain any internal sample and hold circuit.
- The analog to digital converter chips 0808 and 0809 are 8-bit CMOS, successive approximation converters.
- The conversion delay is 100 μs at a clock frequency of 640 kHz, which is quite low as compared to other converters.

ADC 0808 / 0809

Block Diagram



Low-cost ADC, Power 15 mW,

Compatible with a wide range of microprocessors. Power Supply 5 V

Moderate speed 100 μs

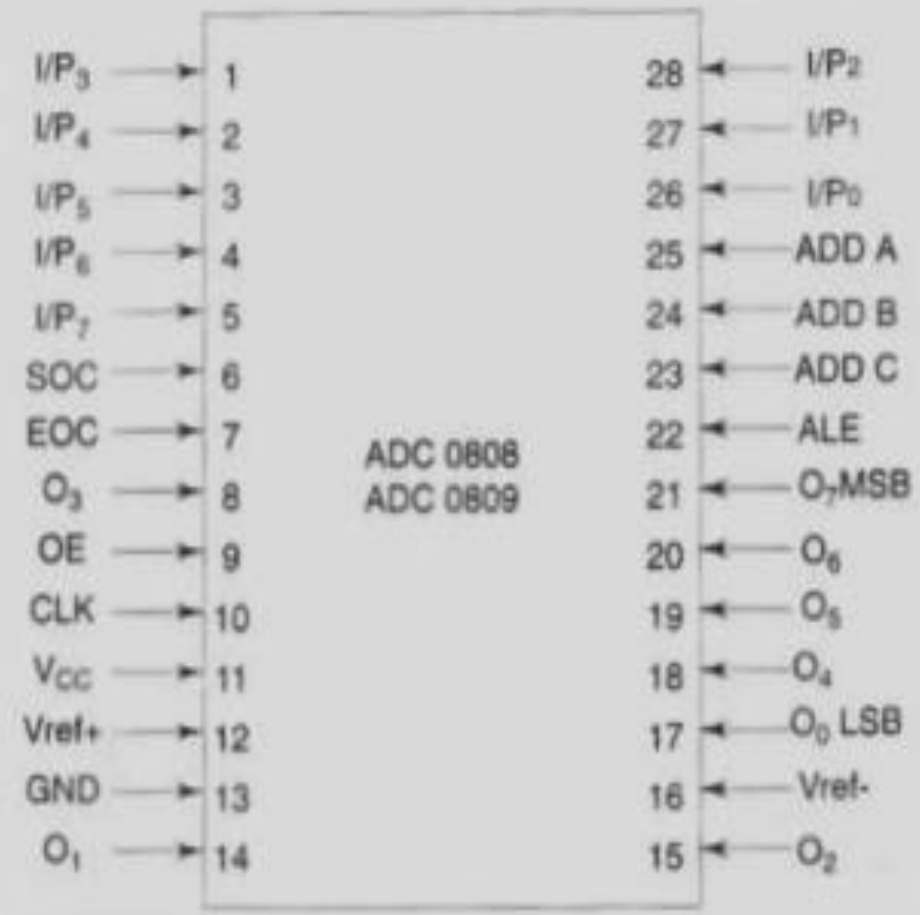
Moderate accuracy Error $\pm$ LSB

ADC 0808/0809

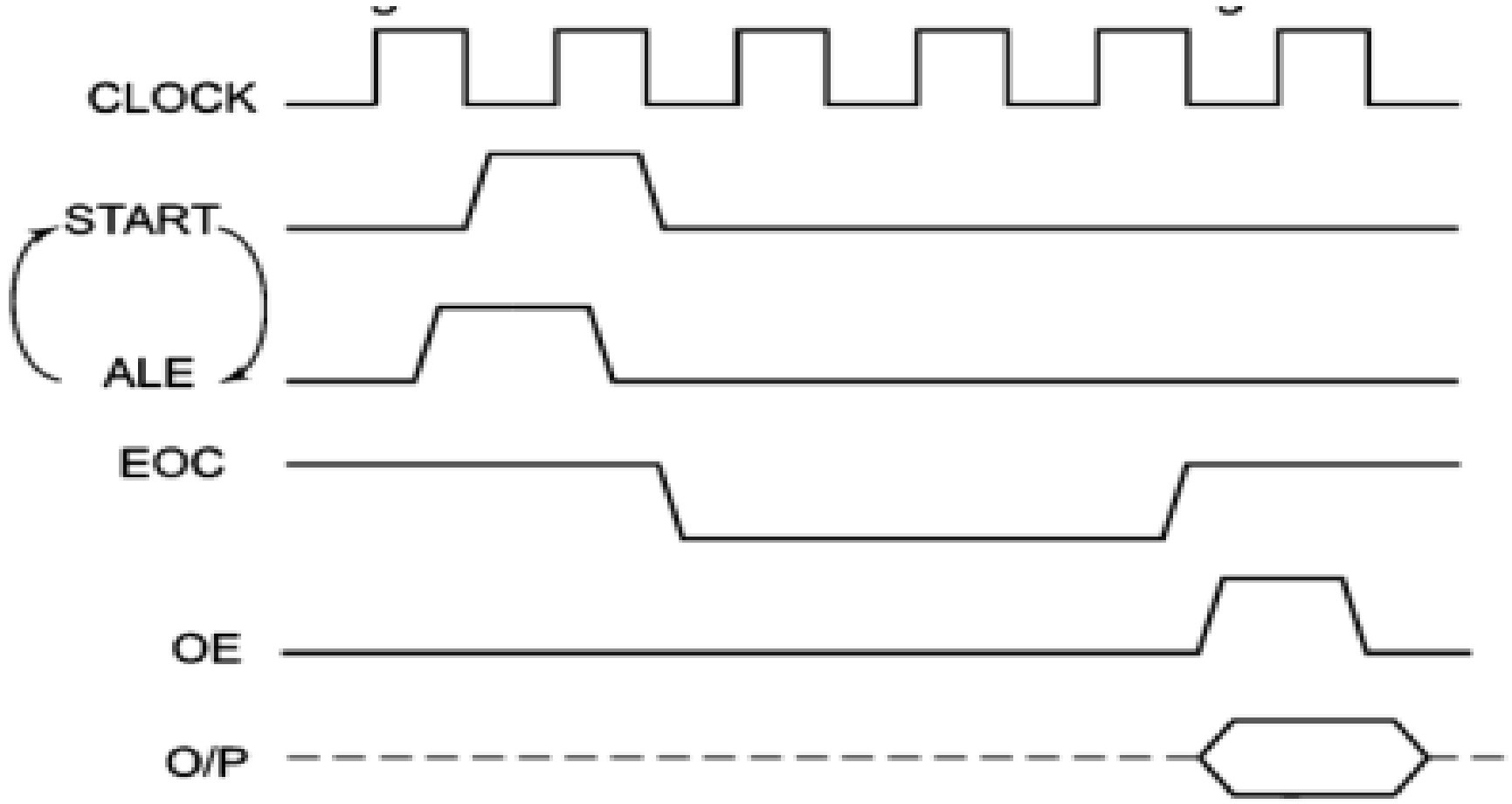
- These converters internally have a 3:8 analog multiplexer so that at a time eight different analog inputs can be connected to the chips.
- Out of these eight inputs only one can be selected for conversion by using address lines ADD A, ADD B and ADD C, as shown.
- Using these address inputs, multichannel data acquisition systems can be designed using a single ADC.

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

ADC 0808/0809 PIN DIAGRAM

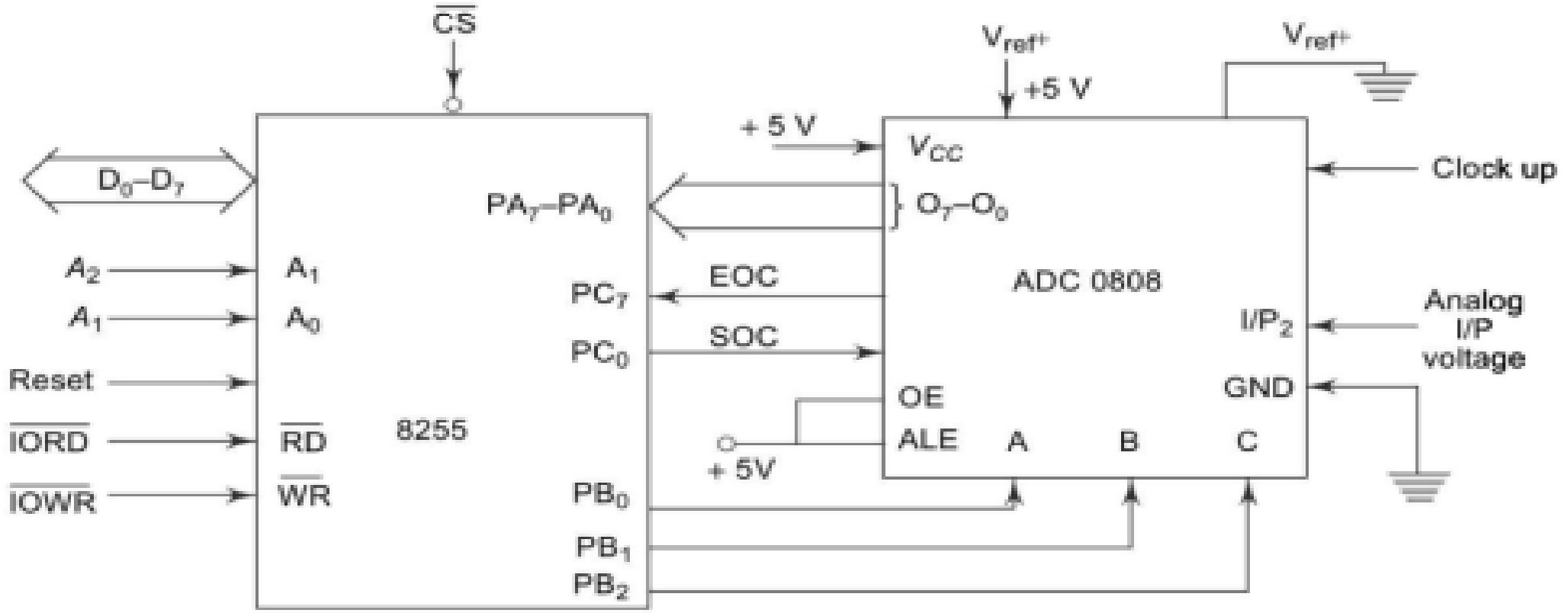


Timing Diagram



ADC 0808 / 0809

Interface ADC 0808 with 8086 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/P_2 of the ADC and a clock input of suitable frequency is available for ADC. Draw the schematic and write required ALP.



INTERFACING ADC 0808 WITH 8086 USING 8255-STEPS

- The analog input I/P2 is used & therefore address pins A,B,C should be 0,1,0 respectively to select I/P2.
- The OE (Out put latch Enable) & 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 ADC.
- Port A acts as a 8-bit input data port to receive the digital data output from the ADC.

Interface ADC 0808 with 8086 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/P_2 of the ADC and a clock input of suitable frequency is available for ADC. Draw the schematic and write required ALP.

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Control word
1	0	0	1	1	0	0	0	= 98 H

D7=1; I/O Mode.
D6=0 and D5=0; Port A Mode
.
D4=1; Port A is input port
D3=1; Port C (Upper) is input
D2=0; Port B Mode 0.
D1=0; Port B is output
D0=0; Port C (Lower) is output

The required ALP is given as follows:

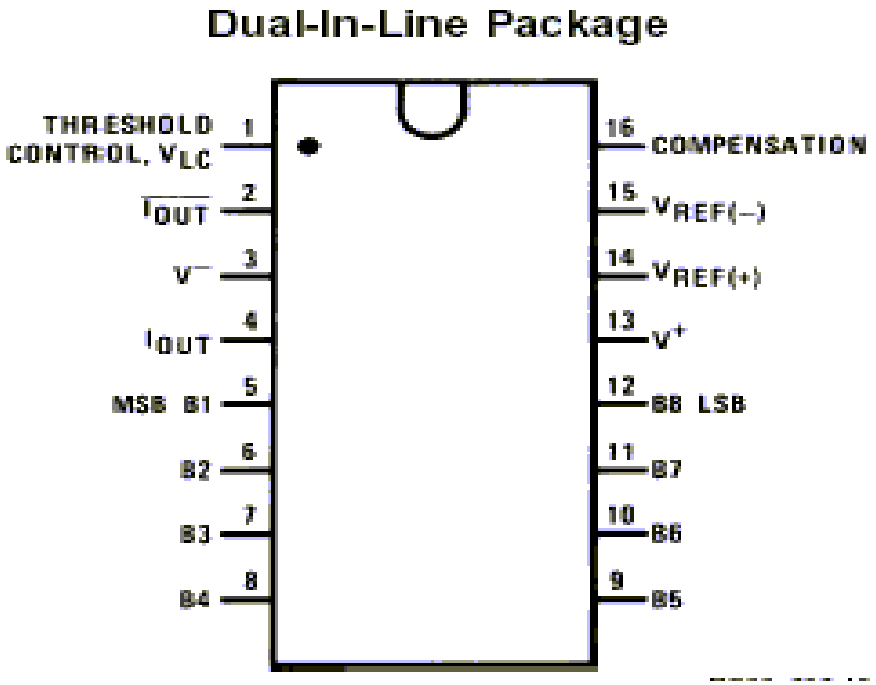
```
        MOV AL,98 H          ; Initialise 8255 as
        OUT CWR,AL           ; discussed above
        MOV AL,02H           ; Select I/P2 as analog
        OUT PORT B,AL        ; input
        MOV AL,00H           ; Give start of conversion
        OUT PORT C,AL        ; pulse to the ADC.
        MOV AL,01 H          ;
        OUT PORT C,AL        ;
        MOV AL,00H           ;
        OUT PORT C,AL        ;
WAIT :  IN AL,PORTC          ; Check for EOC by
        RCL                  ; reading port C upper and
        JNC WAIT             ; rotating through carry.
        IN AL,PORTA          ; If EOC, read digital equivalent in
                                 AL
        HLT                  ; Stop
```

DAC Interface

- DAC is an acronym used for Digital to Analog Converter and DAC interface is used to generate analog output by converting the digital signal obtained from the microprocessor into equivalent analog form.
- More simply, DACs are devices that perform digital to analog conversion however, this conversion requires a reference value on the basis of which the conversion takes place.

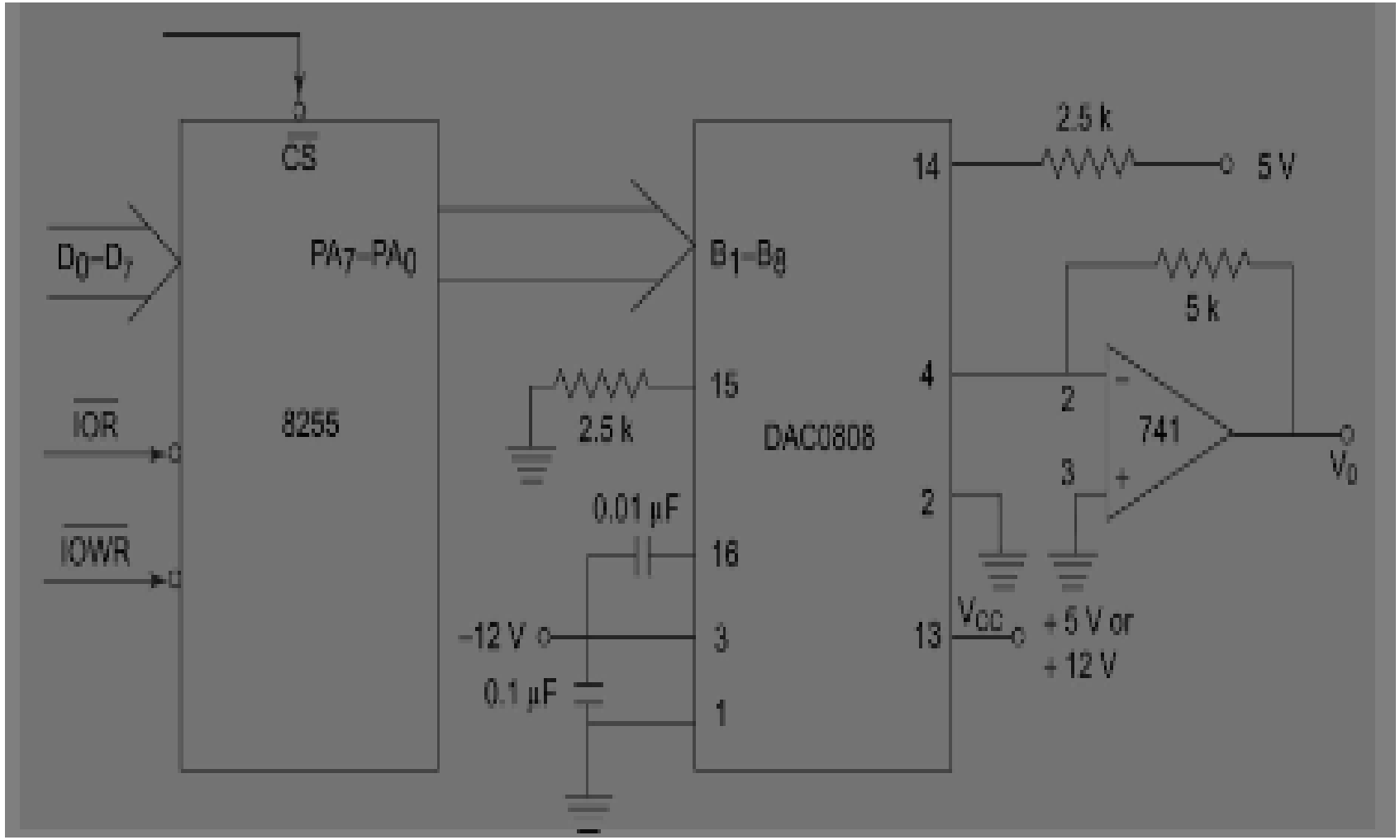
- In general, it is known that a microprocessor generates only digital signal i.e., in the form of binary values as its output.
- However, in some applications like in order to control analog devices, analog signals are needed.
- Due to this reason, DAC interfacing is necessary as using a DAC the digital output of the microprocessor can be converted into analog form.

DAC – 0800



As it is an 8-bit DAC thus it contains 8 digital input lines. Other than the input lines we have two dedicated pins that provide current output and complement of current output which is given the name IOUT and I'OUT. There are a positive and a negative supply voltage pin along with a pin dedicated for compensation voltage. Here VLC represents the threshold control pin and there are two pins for positive and negative values of reference voltage denoted by VREF (+) and VREF (-).

It is an 8-bit DAC that offers high speed and is a current output type of DAC that means it provides analog current as its output. The conversion time offered by this DAC is around 100 ns. The current output signal obtained from it can be converted into voltage by making use of a resistor.



Interfacing DAC0800 with 8086

VOUT = [VREF *(Decimal Input Number)]/8

Program to generate Square wave

```
MOV AL, 80H
MOV DX, 0FFC6H
OUT DX, AL

MOV DX, 0FFC0
again: MOV AL, 00H
OUT DX, AL
CALL delay
MOV AL, 0FFH
OUT DX, AL
CALL delay
JMP again

delay:    MOV CX,00FF
back: LOOP back
RET
```

Program to generate Triangular wave

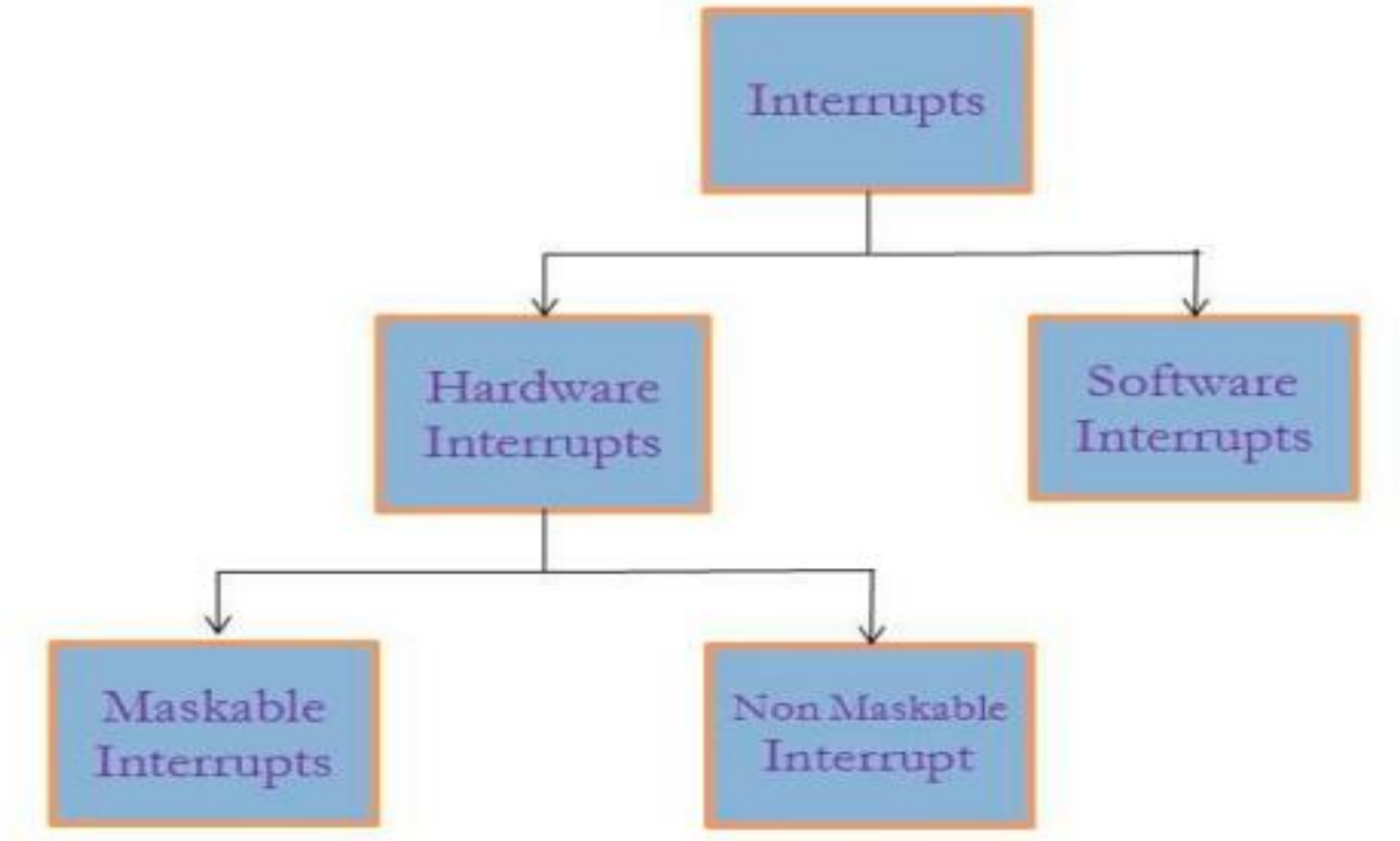
```
MOV AL, 80H
MOV DX, 0FFC6H
OUT DX, AL

MOV DX, 0FFC0H
again: MOV AL, 00H
 up1:OUT DX, AL
INC AL
CMP AL, 0FFH
JNE up1
up2: OUT DX, AL
DEC AL
 CMP AL, 00H
JNE up2
JMP again
```

Interrupts in 8086 microprocessor

- An interrupt is a condition that halts the microprocessor temporarily to work on a different task and then returns to its previous task. An interrupt is an event or signal that requests the CPU's attention.
- This halt allows peripheral devices to access the microprocessor. Whenever an interrupt occurs, the processor completes the current instruction and starts the implementation of an Interrupt Service Routine (ISR) or Interrupt Handler.
- ISR is a program that tells the processor what to do when the interrupt occurs. After the ISR execution, control returns to the main routine where it was interrupted.

Interrupt Types



Hardware Interrupts

- The interrupts initiated by external hardware by sending an appropriate signal to the interrupt pin of the processor is called hardware interrupt.
- The 8086 processor has two interrupt pins INTR and NMI. The interrupts initiated by applying appropriate signal to these pins are called hardware interrupts of 8086.

Hardware Interrupt Sources

The primary sources of interrupts, however are the

- PCs timer chip,
- keyboard,
- serial ports,
- parallel ports,
- disk drives,
- CMOS real- time clock,
- mouse,
- sound cards,
- and other peripheral devices.

Software Interrupts Types 0 through 255

- The software interrupts are program instructions. These instructions are inserted at desired locations in a program.
- While running a program, if software interrupt instruction is encountered then the processor initiates an interrupt.
- The 8086 processor has 256 types of software interrupts.
- The software interrupt instruction is INT n, where n is the type
- number in the range 0 to 255.

Interrupt Priorities

Interrupt	Priority
Divide Error , INT n, INTO	Highest
NMI	
INTR	
Single Step	Lowest

Example

- When divide-by-zero error interrupt and INTR interrupt comes simultaneously the 8086 will do a Divide error (type0) interrupt response first.
- When NMI interrupt and divide-by-zero error interrupt comes simultaneously the 8086 will do an NMI (type2) interrupt response first.

Interrupt Applications

Hardware Interrupt Applications

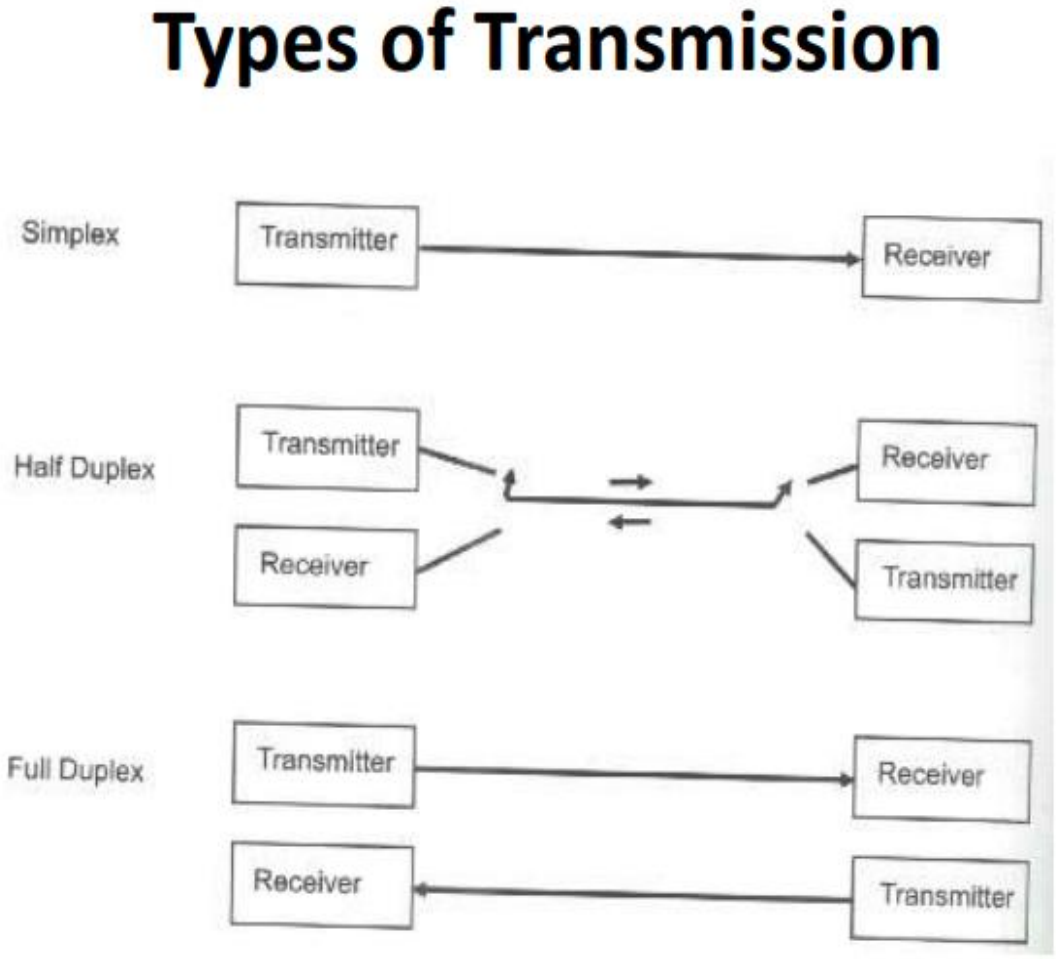
1. Simple Interrupt data input.
2. Counting Applications.
3. Timing Applications.

Software Interrupt Applications

1. To test various ISPs.
2. To insert break points in program for debugging.
3. To call BIOS procedures in an IBM PC type compute

Data Communications

- Data communications refers to the ability of one computer to exchange data with another computer or a peripheral
- Physically, the data comm. path may be a short, 5 to 10 feet ribbon cable connecting a microcomputer and parallel printer; or it might be a high speed telecommunications port connecting two computers thousands of miles apart.
- Standard data communication interfaces and standards are needed
- Centronic's parallel printer interface
- RS-232 defines a serial communications standard
- We focus on serial I/O this week
- 8251 USART (Universal Synchronous/Asynchronous Receiver/Transmitter) is the key component for converting parallel data to serial form and vice versa
- Two types of serial data communications are widely used
  - Asynchronous communications
  - Synchronous communications



1. Simplex

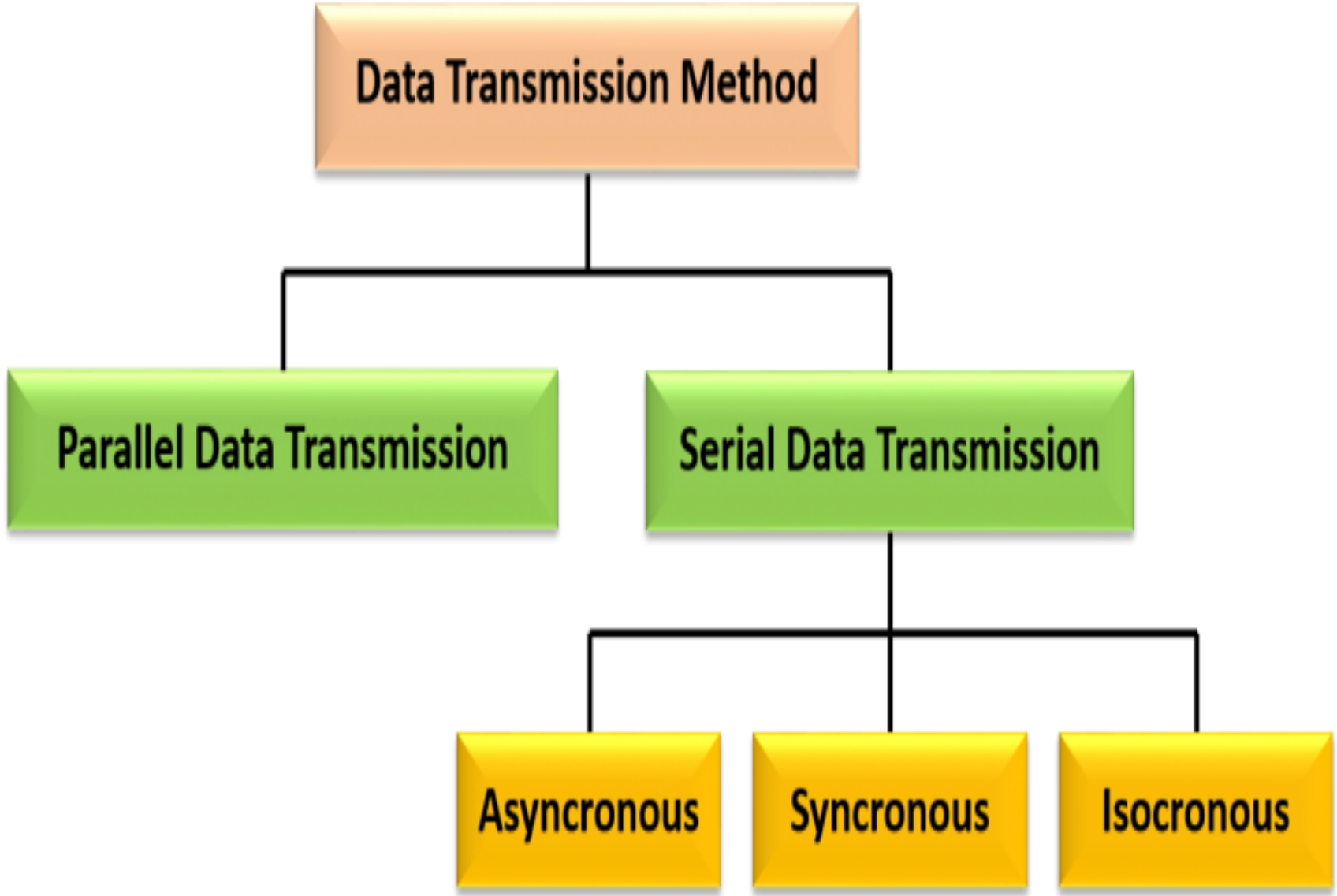
In simplex, the hardware exists such that data transfer takes place only in one direction. There is no possibility of data transfer in the other direction. A typical example is transmission from a computer to the printer.

2. Half Duplex

The half duplex transmission allows the data transfer in both directions, but not simultaneously. A typical example is a walkie-talkie.

3. Full Duplex

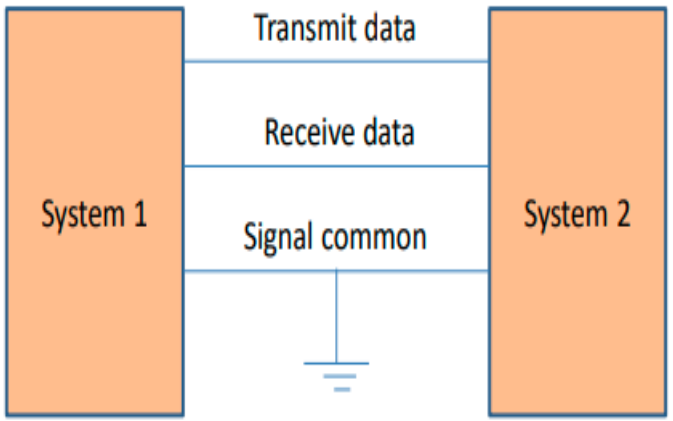
The full duplex transmission allows the data transfer in both direction simultaneously. The typical example is transmission through telephone lines.


Data Transmission Method
Parallel Data Transmission
Serial Data Transmission
Asyncronous
Syncronous
Isocronous

Serial Transmission	Parallel Transmission
Data flows in two directions, bit by bit	Data flows in multiple directions, 8 bits at a time
Cost is economical	Cost is expensive
Number of bits transferred per clock pulse is 1 bit	Number of bits transferred per clock pulse is 8 bits.
Speed is slower	Speed is faster
Used for long distance communication	Used for short distance communication
Computer to computer	Computer to printer

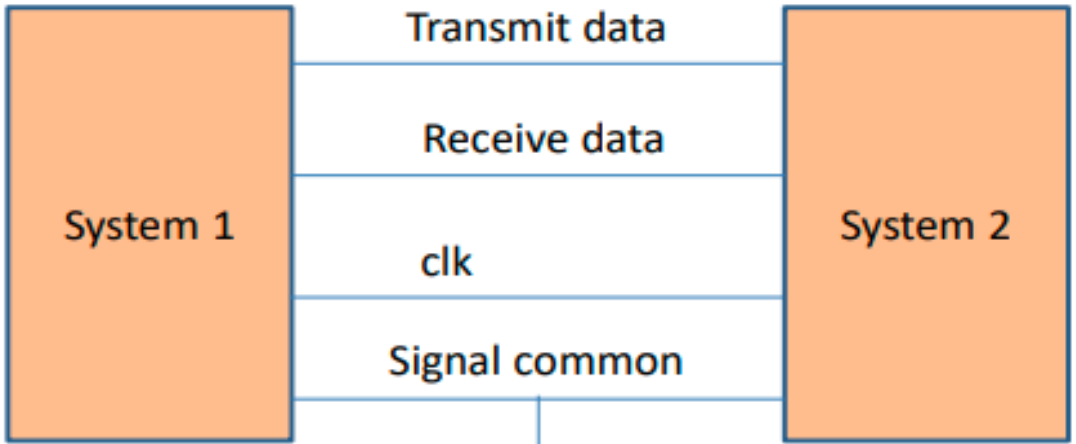
Asynchronous Communications

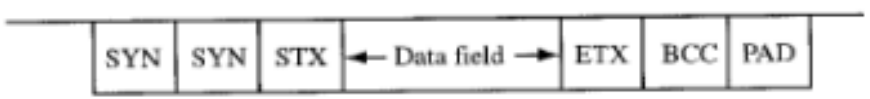
- Eliminates the need for a clock signal between two microprocessor based systems



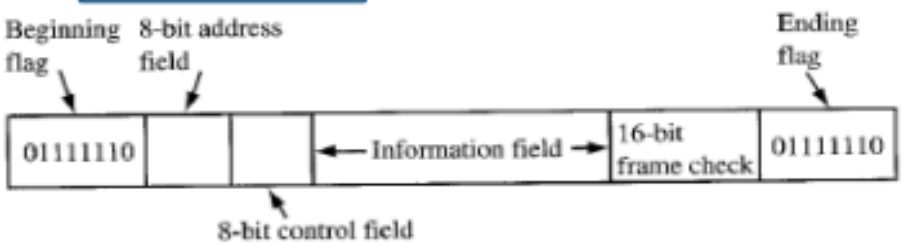
Data to be transmitted is sent out one character at a time and the receiver end of the communication line synchronization is performed by examining synchronization bits that are included at the beginning and at the end of each character

Synchronous Communications





BISYNC: Each block of data has synch characters. The size of block data can be 100 or more bytes. BCC checks for errors.



Serial Data Link Control: Developed by IBM used for computer networking (Token Ring). After Flag byte the network address is sent. Control Byte stores information about sequence of data etc. Data is thousands of bits. 16 bit field is used for error checking.

INTEL 8251 USART ARCHITECTURE AND INTERFACING

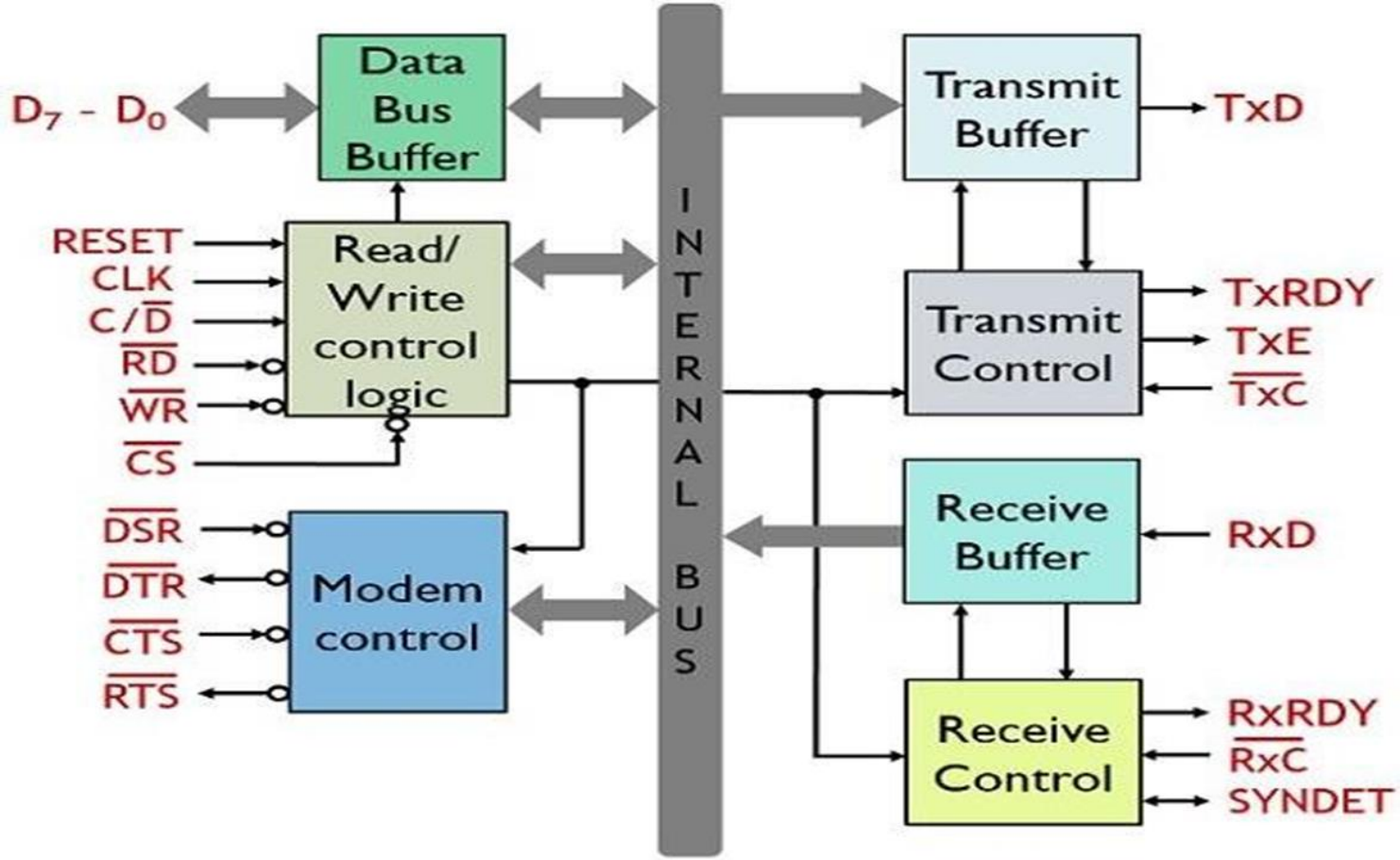
8251 Universal Synchronous and Asynchronous Receiver Transmitter (USART) or Programmable Communication Interface (PCI) acts as a mediator between microprocessor and peripheral to transmit serial data into parallel form and vice versa.

1. It takes data serially from peripheral (outside devices) and converts into parallel data.
2. After converting the data into parallel form, it transmits it to the CPU.
3. Similarly, it receives parallel data from microprocessor and converts it into serial form.
4. After converting data into serial form, it transmits it to outside device (peripheral).
5. Also, it allows both synchronous and asynchronous transmission and reception thus is called so.

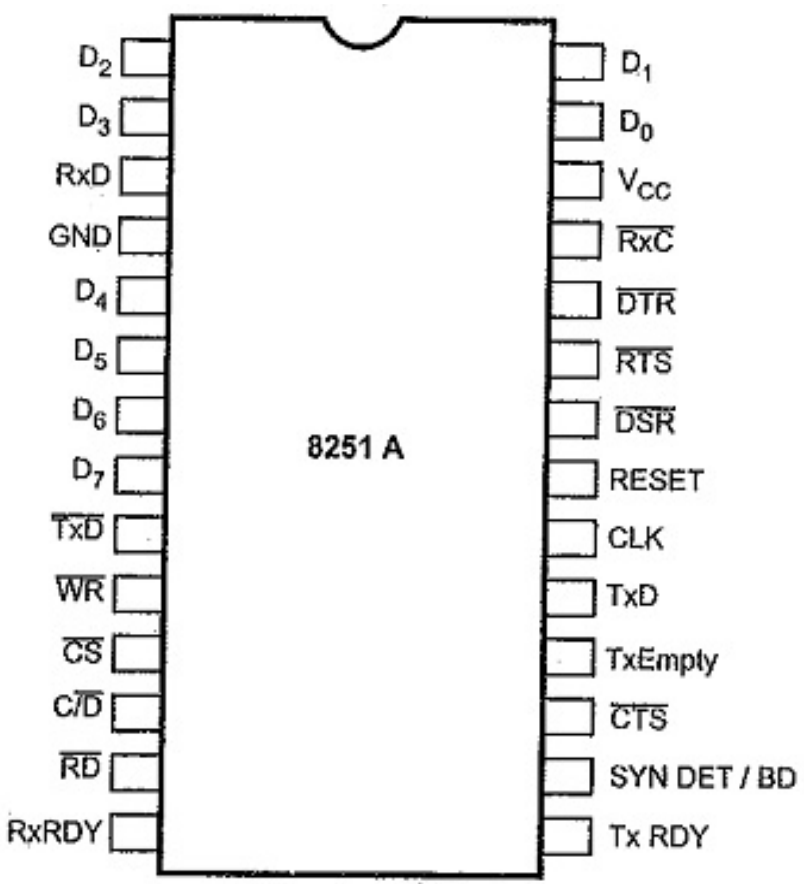
Features of USART

- It is an universal synchronous and asynchronous communication controller.
- It supports standard asynchronous protocol wit
  a. 5 to 8 bit character format
  b. odd, even or no parity generation and detection
  c. Automatic break detect and handling
- It is built in baud rate generator
- It allows full duplex transmission
- It provides error detection logic, when detects parity, overrun and framing errors.
- It has 28 pins.

USART 8251 ARCHITECTURE



USART 8251 Pin Diagram



There are 5-BLOCKS IN 8251:

1. **Data bus buffer**
2. **Read/Write control logic**
3. **Modem control (modulator/demodulator)**
4. **Transmitter section**
    1. **Transmit buffer**
    2. **Transmit control**
5. **Receiver section**
    1. **Receive buffer**
    2. **Receive control**

1.Data bus buffer –
This block helps in interfacing the internal data bus of 8251 to the system data bus.

- The data bus buffer has **8-bit** bidirectional data bus that allows the transfer of data bytes, status or command word between the processor and external devices.

2. **Read/Write control logic –**
It is a control block for overall device. It controls the overall working by selecting the operation to be done.

- ***CS***: It is chip select. A low signal at this pin shows that processor has selected 8251 in order to communicate with the peripheral devices.
- ***C/D***: As the system has control, status and data register. So, when a high signal is present at this pin then control or status register is addressed. While in case of low signal data register is addressed.
- ***RD*** and ***WR***: Both read and write are active low signal pins. A low signal at RD shows that the processor is reading the control, status or data bytes from the 8251. While at WR indicates the write operation over the data bus of 8251.
- ***CLK*** and ***RESET***: CLK stands for clock and it produces the internal timing for the device. While an active high signal at the RESET pin puts the 8251 in the idle mode.

3.Modem control (modulator/demodulator) –

A device converts analog signals to digital signals and vice-versa and helps the computers to communicate over telephone lines or cable wires. The following are active-low pins of Modem.

1. **DSR:** Data Set Ready signal is an input signal.
2. **DTR:** Data terminal Ready is an output signal.
3. **CTS: Clear to send-** It is an input signal which controls the data transmit circuit.
4. **RTS: Ready to send-** It is an output signal which is used to set the status RTS.

4. Transmitter Section

It has 2 blocks.

a. Transmit buffer –

This block is used for parallel to serial converter that receives a parallel byte for conversion into serial signal and further transmission onto the common channel.

1. **TXD:** It is an output signal, if its value is one, means transmitter will transmit the data.

b. Transmit control –

This block is used to control the data transmission with the help of following pins:

1. **TXRDY:** It means transmitter is ready to transmit data character.
2. **TXEMPTY:** An output signal which indicates that TXEMPTY pin has transmitted all the data characters and transmitter is empty now.
3. **TXC:** An active-low input pin which controls the data transmission rate of transmitted data.

5.Receiver Section:

It has 2 blocks.

a. Receive buffer –

This block acts as a buffer for the received data.

1. **RXD:** An input signal which receives the data.

b. Receive control –

This block controls the receiving data.

1. **RXRDY:** An input signal indicates that it is ready to receive the data.
2. **RXC:** An active-low input signal which controls the data transmission rate of received data.
3. **SYNDET/BD:** An input or output terminal. External synchronous mode-input terminal and asynchronous mode-output terminal.

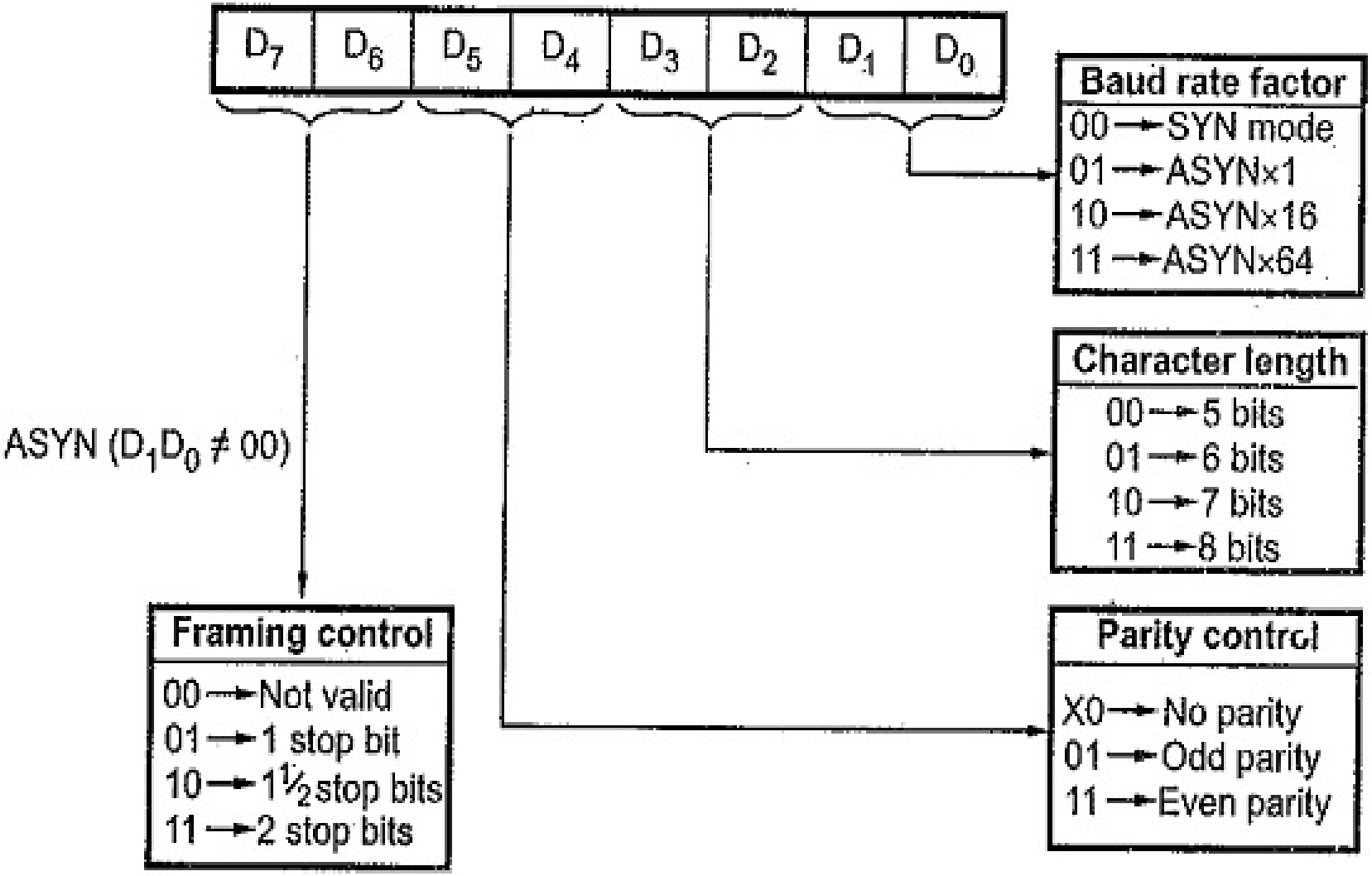
Control Word of 8251:

The Control Word of 8251 defines the complete functional definition of 8251 Block Diagram in Microprocessor and they must be loaded before any transmission or reception. The control words of Block Diagram of 8251 Microcontroller are split into two formats

1.Mode instruction

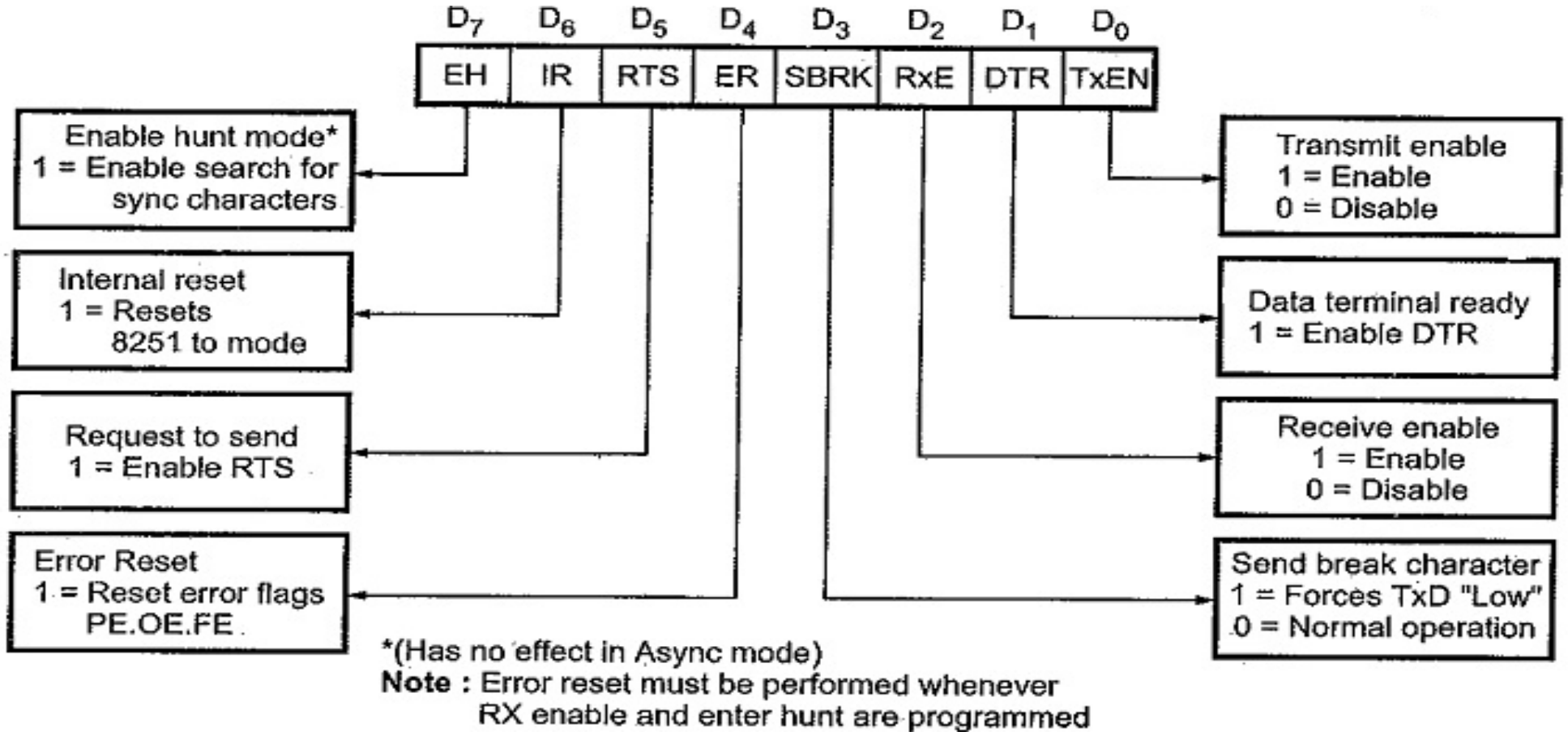
2.Command instruction

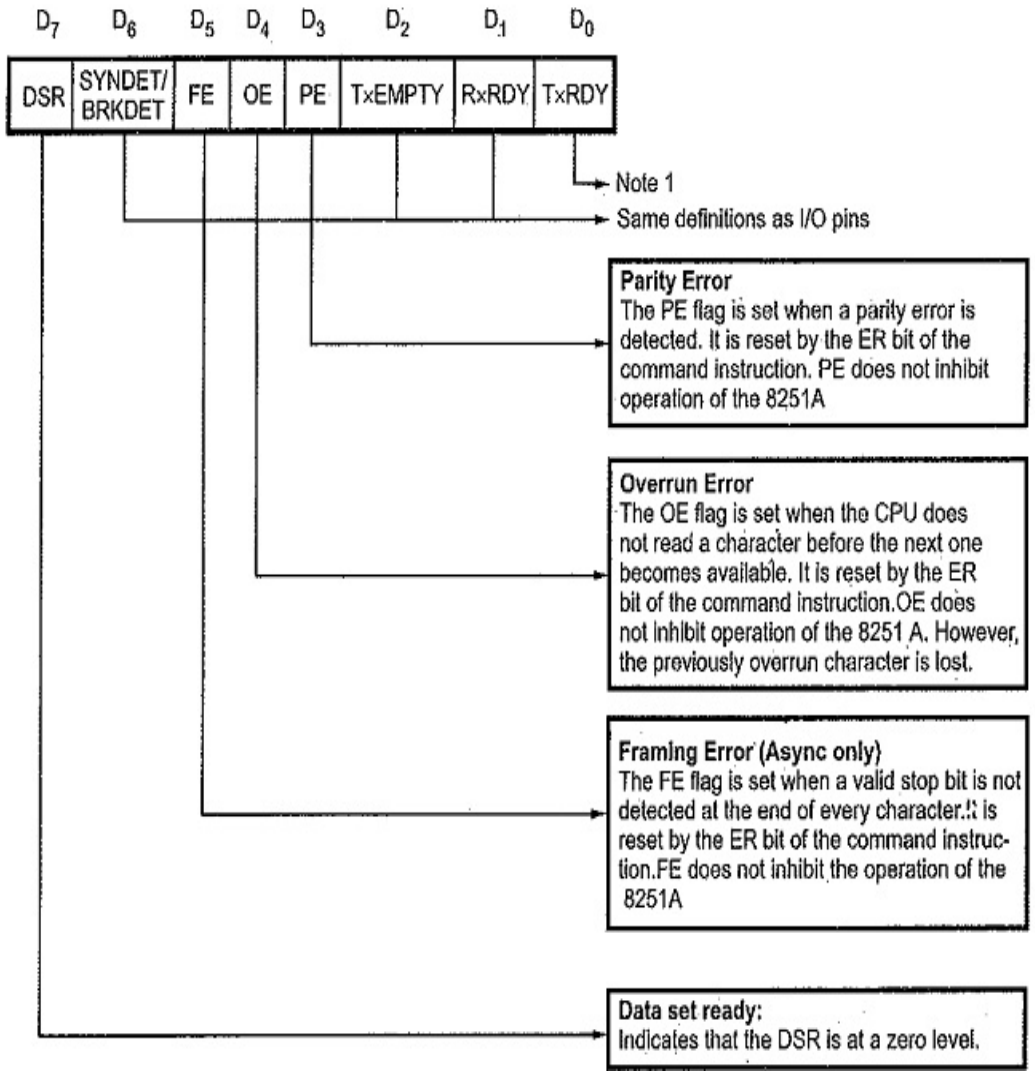
Mode Instruction : Fig. shows the mode instruction format of 8251.



Command Instruction:

After the mode instruction, command character should be issued to the USART. It controls the operation of the USART within the basic frame work established by the mode instruction. Fig. shows command instruction format.





8251A Status Word:

In the data communication systems it is often necessary to examine the “status” of the transmitter and receiver. It is also necessary for CPU to know if any error has occurred during communication. The 8251 Block Diagram in Microprocessor allow the programmer to read above mentioned information from the status register any time during the functional operation. Fig. shows the format of status register.

Interfacing 8251A to 8086 in I/O Mapped I/O Mode:

Fig. 14.46 shows the interfacing of 8251 with 8086 in I/O mapped I/O technique. Here, RD and WR signals are activated when M/IO signal is low, indicating I/O bus cycle. Only lower data bus (D0 – D7) is used as 8251 is 8-bit device. Reset out signal from clock generator is connected to the reset signal of the 8251.

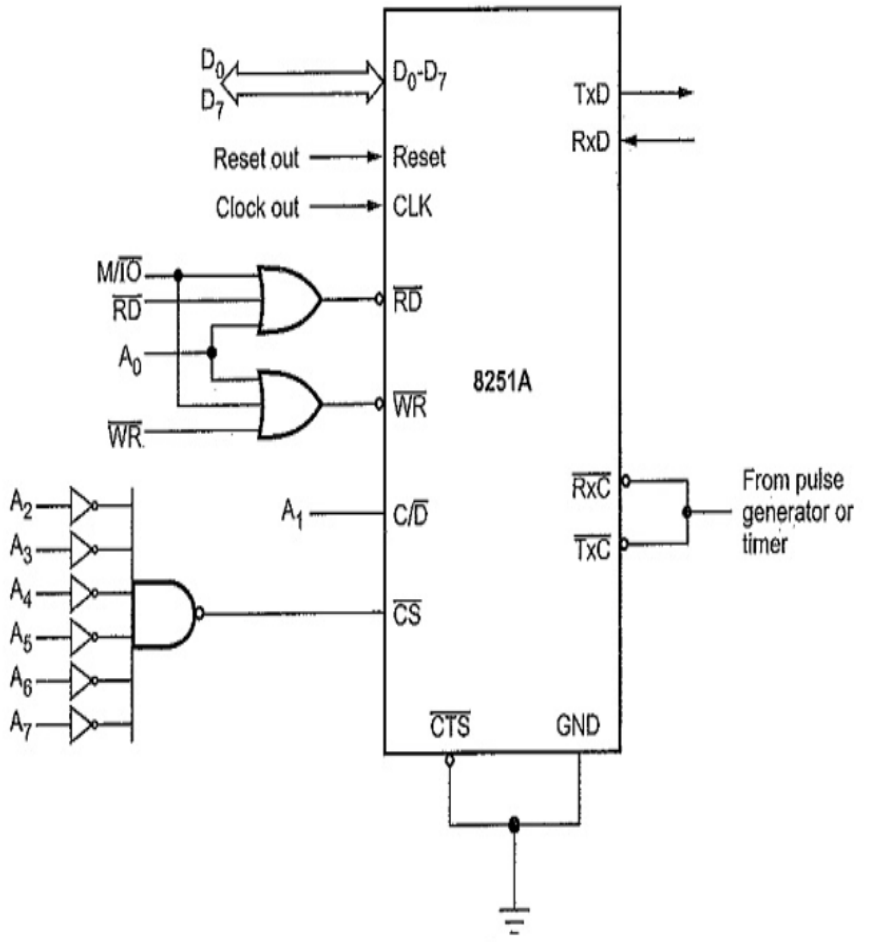


Fig. 14.46 Interfacing of 8251A with 8086 in I/O mapped I/O

I/O Map :

Register	Address lines								Address
	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	
Data Register	0	0	0	0	0	0	0	0	00H
Control Register	0	0	0	0	0	0	1	0	02H

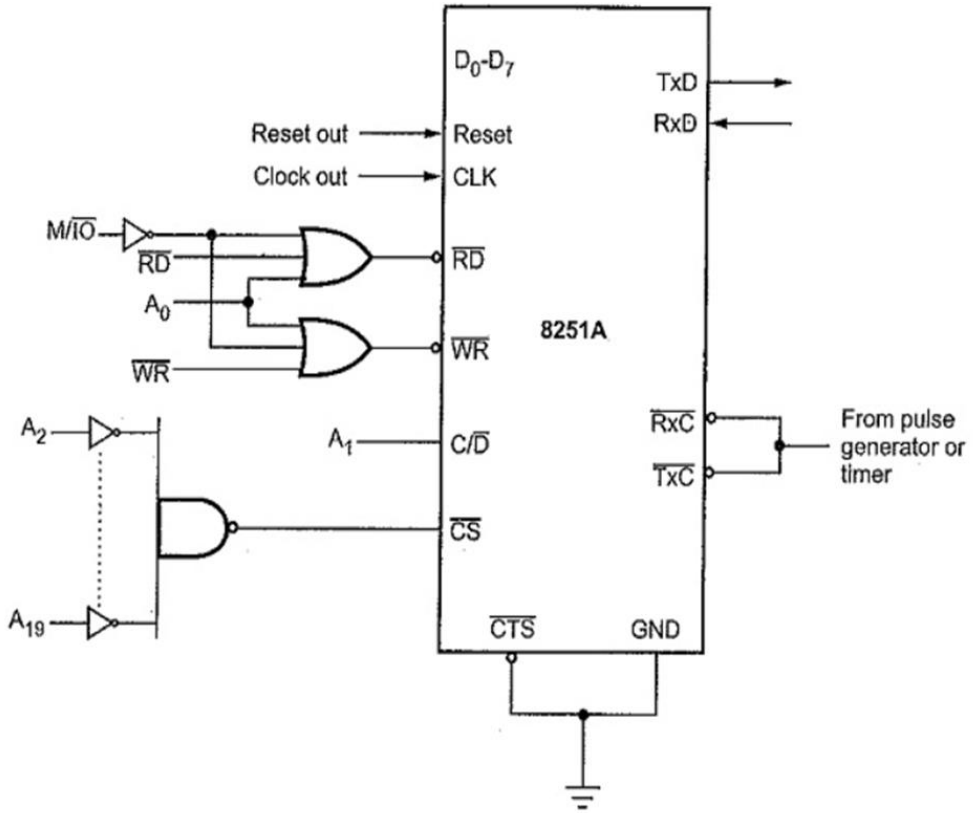


Fig. 14.47 Interfacing of 8251A with 8086 in memory mapped I/O

Interfacing 8251 to 8086 in Memory Mapped I/O:

In this type of I/O interfacing, the 8086 uses 20 address lines to identify an I/O device; an I/O device is connected as if it is a memory register. The 8086 uses same control signals and instructions to access I/O as those of memory. Fig. 14.47 shows the interfacing of 8251 with 8086 in memory mapped I/O technique. Here, RD and WR signals are activated when M/IO signal is high, indicating memory bus cycle. Address line A1 is used to select either data register, or control register. The remaining address lines A2-A19 are used to decoder the addresses for 8251.

I/O Map :

Register	A_{19}	A_{18}	A_{17}	A_{16}	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
Data Register	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	00000H
Control Register	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	00002H

MODES OF DATA TRANSFER:

- We store the binary information received through an external device in the memory unit.
- The information transferred from the CPU to external devices originates from the memory unit.
- Although the CPU processes the data, the target and source are always the memory unit.

We can transfer this information using three different modes of transfer.

1.Programmed I/O

2.Interrupt- initiated I/O

3.Direct memory access(DMA)

Direct memory access

- Direct memory access (DMA) is a process in which an external device takes over the control of system bus from the CPU.
- DMA is for high-speed data transfer from/to mass storage peripherals, e.g. harddisk drive, magnetic tape, CD-ROM, and sometimes video controllers.
- The basic idea of DMA is to transfer blocks of data directly between memory and peripherals.
- The data don’t go through the microprocessor but the data bus is occupied.

Basic process of DMA – Minimum Mode

- The **HOLD** and **HLDA** pins are used to receive and acknowledge the hold request respectively.
- Normally the CPU has full control of the system bus.
- In a DMA operation, the peripheral takes over bus control temporarily.

DMA controller

- A DMA controller interfaces with several peripherals that may request DMA.
- The controller decides the priority of simultaneous DMA requests communicates with the peripheral and the CPU, and provides memory addresses for data transfer.
- DMA controller commonly used with 8086 is the 8257/8237 programmable device.
- The 8257/8237 is a 4-channel device.
- Each channel is dedicated to a specific peripheral device and capable of addressing 64 K bytes section of memory.

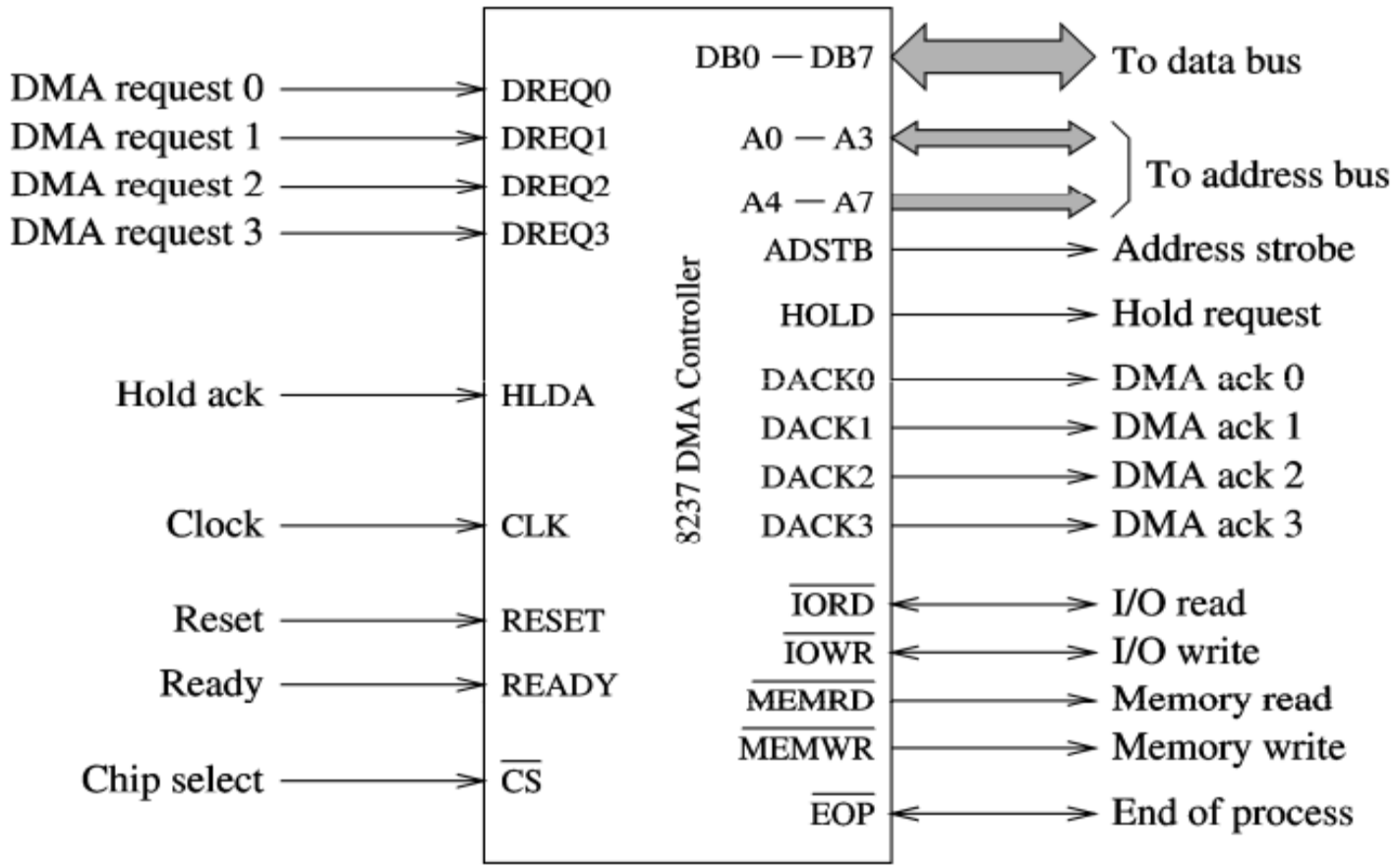
Basic process of DMA – Maximum Mode

- The **RQ/GT1** and **RQ/GT0** pins are used to issue DMA request and receive acknowledge signals.
- Sequence of events of a typical DMA process:
  1. *Peripheral asserts one of the request pins, e.g. RQ/GT1 or RQ/GT0 (RQ/GT0 has higher priority)*
  2. *8086 completes its current bus cycle and enters into a HOLD state.*
  3. *8086 grants the right of bus control by asserting a grant signal via the same pin as the request signal.*
  4. *DMA operation starts.*
  5. *Upon completion of the DMA operation, the peripheral asserts the request/grant pin again to relinquish bus control.*

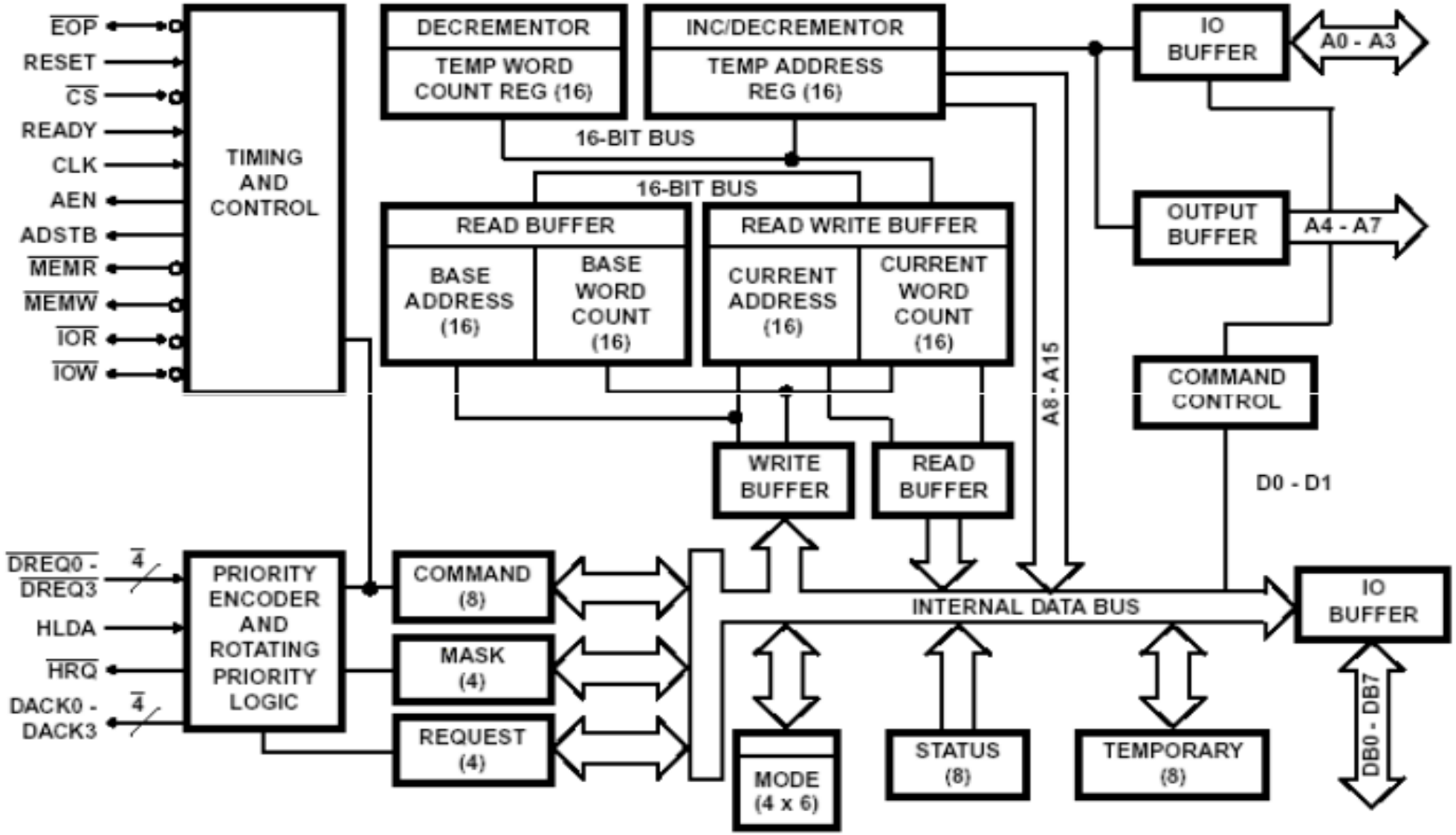
Features of 8237

- 4 independent DMA channel
- Enable and disable control of individual requests
- Memory – memory transfer
- Address increment or decrement
- Cascading

8237 - DMA Controller



Block Diagram



8237 Registers

1. Current word register
2. Command register
3. Mode register
4. Request register
5. Mask register
6. Status register
7. Temporary register
8. Current address register

8237 Registers

1.Current address register

- One 16-bit register for each channel
- Holds address for the current DMA transfer

2.Current word register

- Keeps the byte count
- Generates terminal count (TC) signal when the count go from zero to FFFFH

3.Command register

- Used to program 8257

4.Mode register

- Each channel can be programmed to
  - Read or write
  - Autoincrement or autodecrement the address
  - Autoinitialize the channel

5.Request register

- For software-initiated DMA

6.Mask register

- Used to disable a specific channel

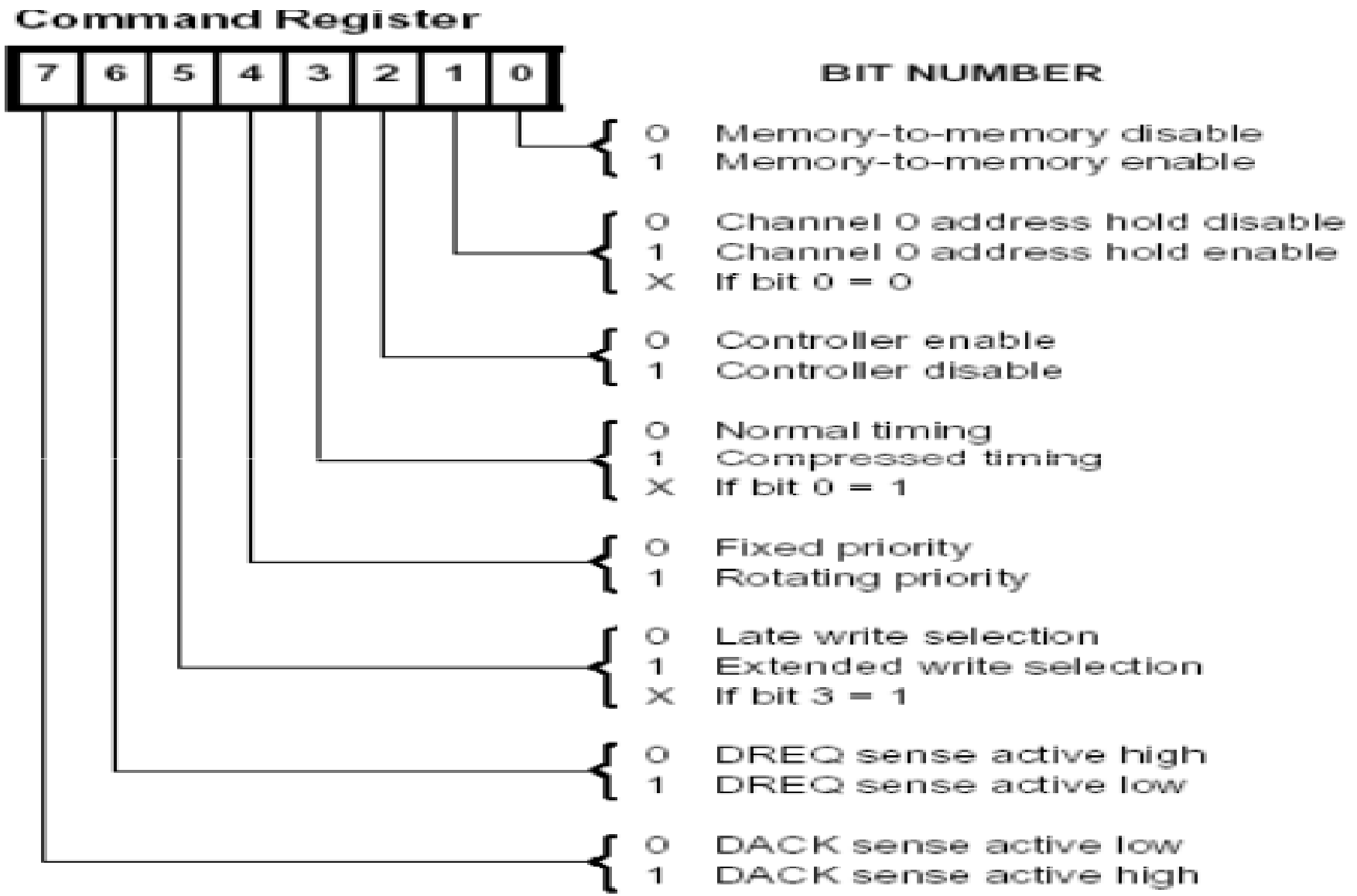
7.Status register

8.Temporary register

- Used for memory-to-memory transfers

Types of data transfer

- 8237 supports **four** types of data transfer
  1. **Single cycle transfer**
     - Only single transfer takes place
     - Useful for slow devices
  2. **Block transfer mode**
     - Transfers data until TC is generated or external $\overline{\text{EOP}}$ signal is received
  3. **Demand transfer mode**
     - Similar to the block transfer mode
     - In addition to TC and EOP, transfer can be terminated by deactivating DREQ signal
  4. **Cascade mode**
     - Useful to expand the number channels beyond four

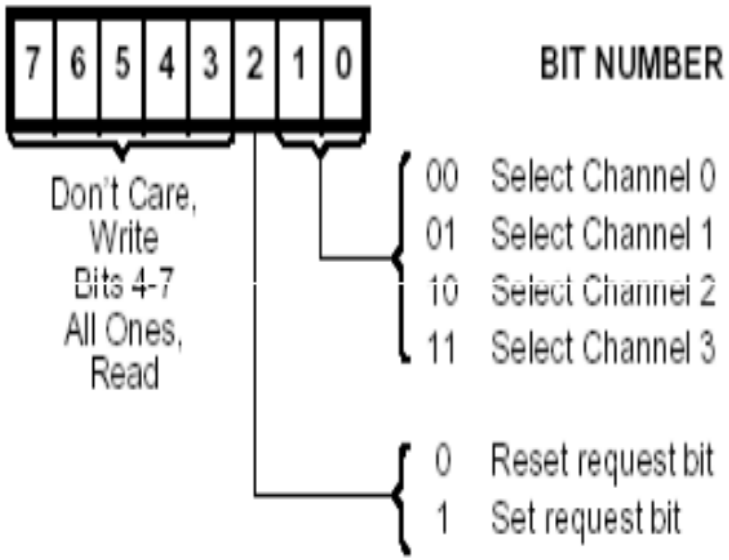

Command Register
7 6 5 4 3 2 1 0
BIT NUMBER
0 Memory-to-memory disable
1 Memory-to-memory enable
0 Channel 0 address hold disable
1 Channel 0 address hold enable
X If bit 0 = 0
0 Controller enable
1 Controller disable
0 Normal timing
1 Compressed timing
X If bit 0 = 1
0 Fixed priority
1 Rotating priority
0 Late write selection
1 Extended write selection
X If bit 3 = 1
0 DREQ sense active high
1 DREQ sense active low
0 DACK sense active low
1 DACK sense active high

Mode Register

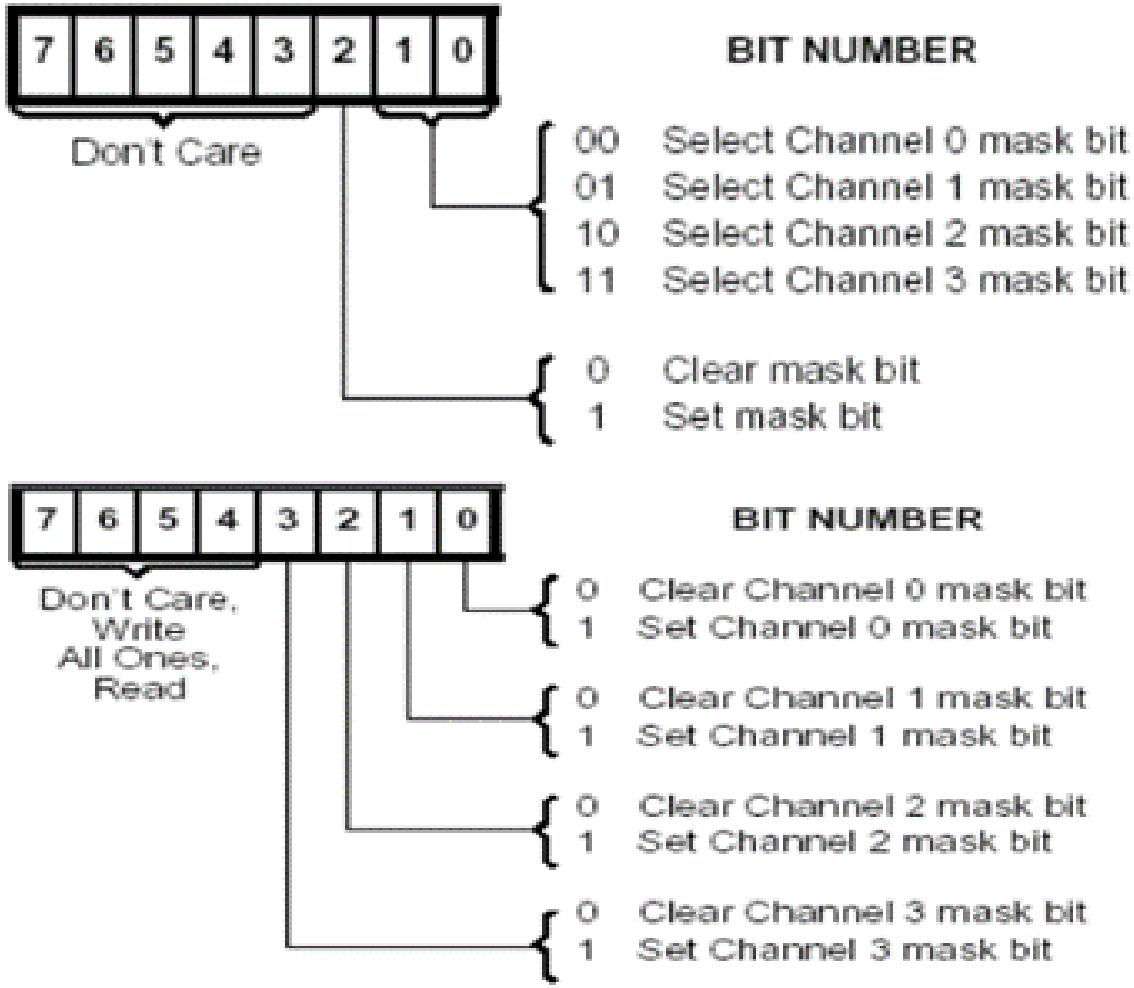
BIT NUMBER

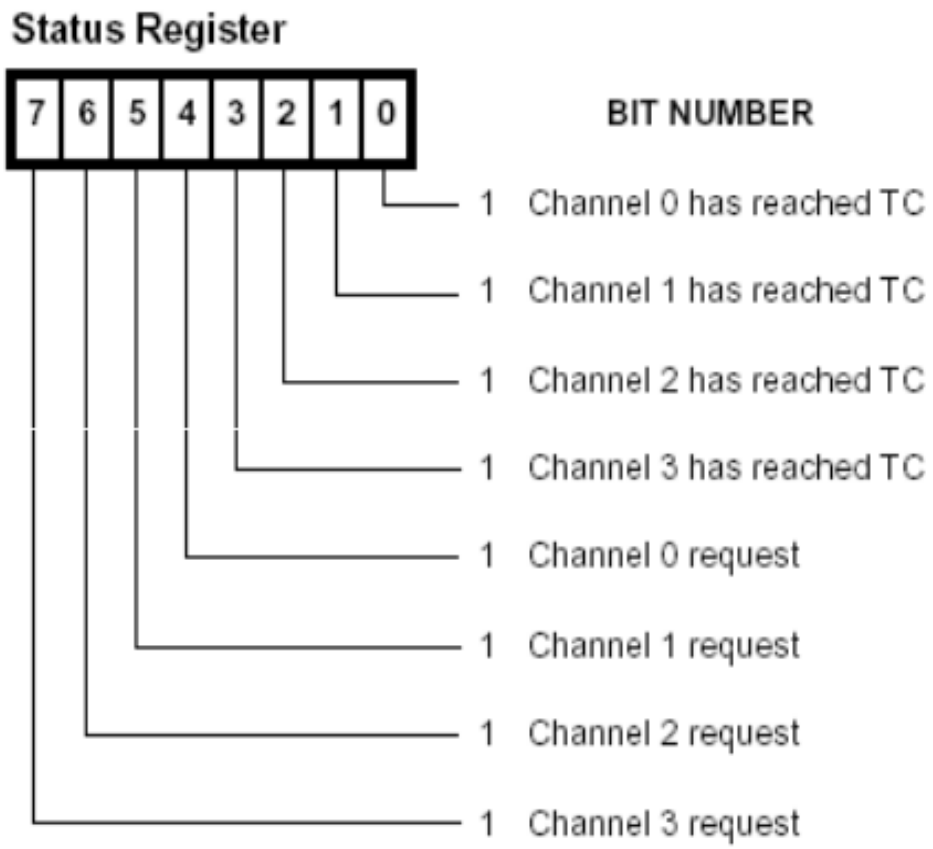
00	Channel 0 select
01	Channel 1 select
10	Channel 2 select
11	Channel 3 select
XX	Readback
00	Verify transfer
01	Write transfer
10	Read transfer
11	!!lega!
XX	If bits 6 and 7 = 11
0	Autoinitialization disable
1	Autoinitialization enable
0	Address increment select
1	Address decrement select
00	Demand mode select
01	Single mode select
10	Block mode select
11	Cascade mode select

Request Register



Mask Register





Advantages of DMA

- Fast memory transfer of data
- CPU and DMA run concurrently under cache mode
- DMA can trigger an interrupt, which frees the CPU from polling the channe

8259-Programmable Interrupt Controller (PIC)

The 8259 is known as the Programmable Interrupt Controller (PIC) microprocessor. In 8085 and 8086 there are five hardware interrupts and two hardware interrupts respectively. By adding 8259, we can increase the interrupt handling capability. This chip combines the multi-interrupt input source to single interrupt output. This provides 8-interrupts from IR0 to IR7.

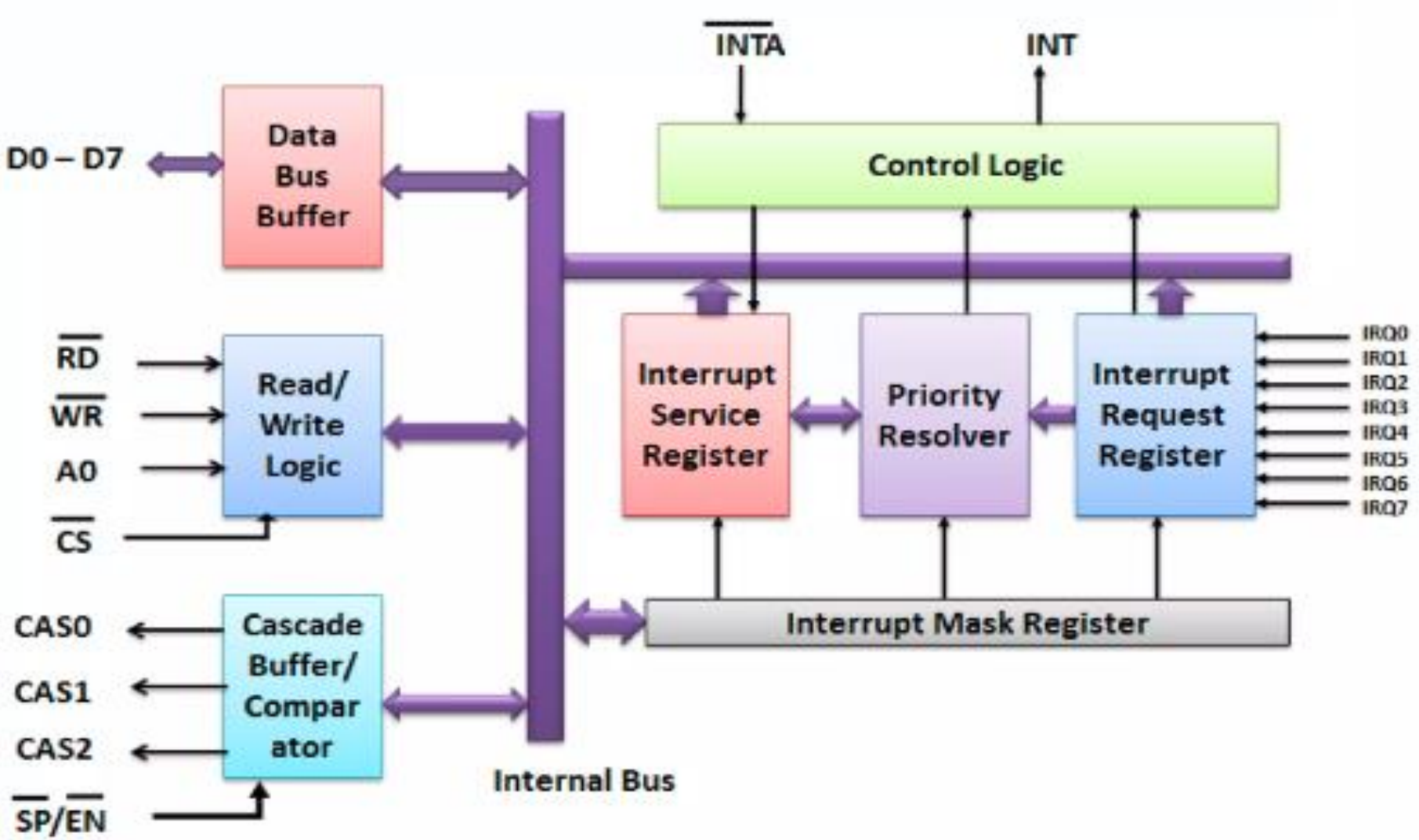
Features of 8259

- Supports 8 interrupt inputs from the peripherals and issues a single interrupt signal.
- Supports cascading of eight 8259ICs and multiplexes 64 interrupt sources to 1.
- Set priorities for the interrupts and provide different interrupt vector addresses.
- No clock is required

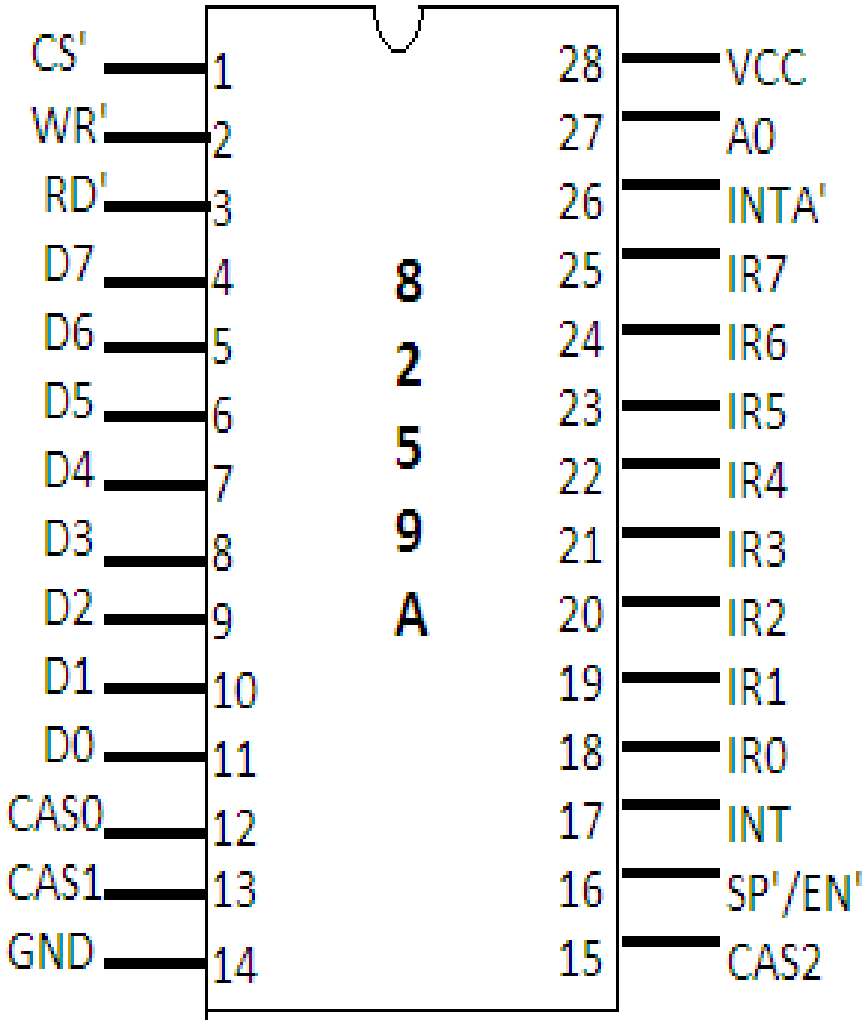
Need for 8259-PIC

- Interrupts are used to handle the routines.
- **Types**:
  - Maskable & Non Maskable
- For interrupts from **multiple sources**, hardware can use an external device called *Programmable interrupt controller* or *Priority Interrupt controller*.
- Main purpose of 8259 is to do the task of calling the ISR based on interrupt priority.
- It acts as **multiplexer** as it combines multiple interrupt input sources into a single interrupt request

8259- PIC Architecture



8259 Pin diagram



DATA BUS BUFFER

- This 3- state, bidirectional 8-bit buffer is used to interface the 8259 to the system data bus.
- Control words and status information are transferred through the data bus buffer.

READ/WRITE & CONTROL LOGIC

- The function of this block is to accept OUTPUT commands from the CPU.
- It contains the initialization command word (ICW) register and operation command word (OCW) register which store the various control formats for device operation.
- This function block also allows the status of 8159Ato be transferred to the data bus.

INTERRUPT REQUEST REG. (IRR)

- IRR stores all the interrupt inputs that are requesting service.
- Basically, it keeps track of which interrupt inputs are asking for service.
- If an interrupt input is unmasked, and has an interrupt signal on it, then the corresponding bit in the IRR will be set.

INTERRUPT MASK REGISTER (IMR)

- The IMR is used to disable (Mask) or enable (Unmask) individual interrupt inputs.
- Each bit in this register corresponds to the interrupt input with the same number. The IMR operation on the IRR.
- Masking of higher priority input will not affect the interrupt request lines of lower priority. To unmask any interrupt the corresponding bit is set '0'.

IN SERVICE REGISTER (ISR)

- The in service registers keeps tracks of which interrupt inputs are currently being serviced.
- For each input that is currently being serviced the corresponding bit will be set in the in service register.
- Each of these 3-reg can be read as status reg.

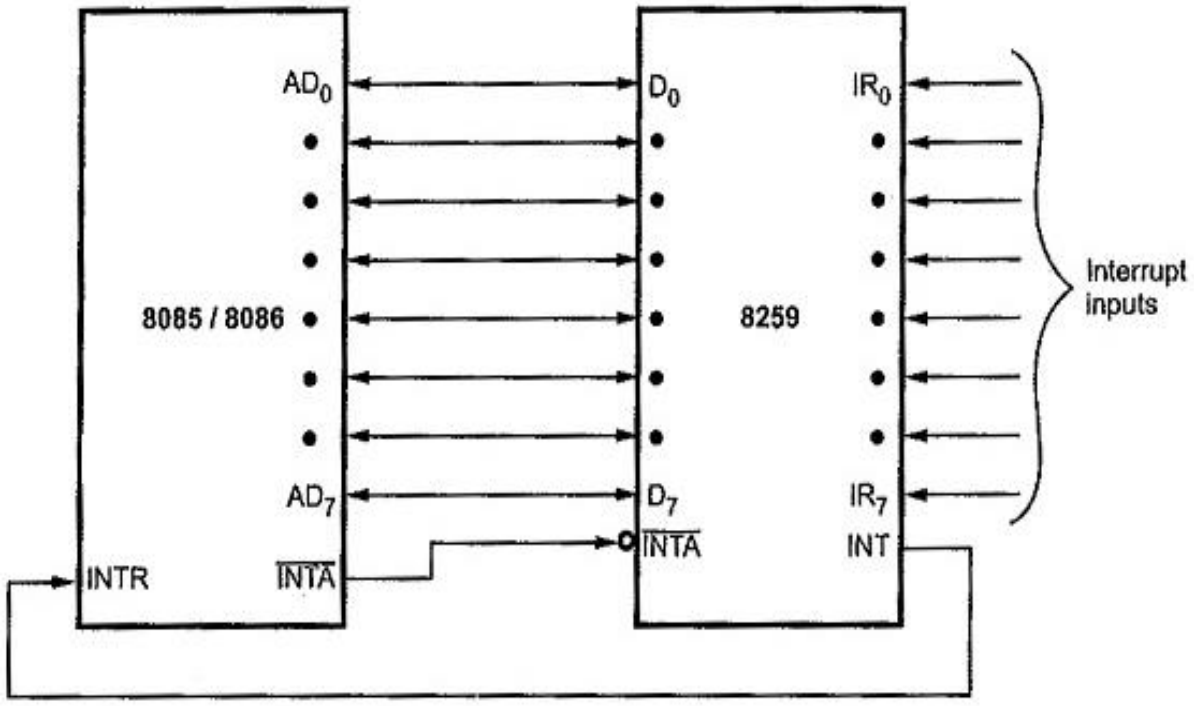
PRIORITY RESOLVER

- This logic block determines the priorities of the set in the IRR.
- The highest priority is selected and strobed into the corresponding bit of the ISR during pulse.

CASCADE BUFFER/COMPARATOR

- This function blocks stores and compare the IDS of all 8259's in the reg. The associated 3-I/O pins (CAS0-CAS2) are outputs when 8259 is used a master.
- Master and are inputs when 8259 is used as a slave. As a master, the 8259 sends the ID of the interrupting slave device onto the cas2-cas0.
- The slave thus selected will send its pre-programmed subroutine address on to the data bus during the next one or two successive pulses.

Connection between 8085/8086 and 8259A



- The Fig. shows the connection between 8085/8086 and 8259A.
- The 8259A is a commonly used priority interrupt controller, which is specifically designed for use with interrupt signals INTR and INTA of Intel series.
- It is packaged in a 28 pin DIP. It uses NMOS technology and requires a single + 5V supply.

10 M Questions

1. With neat functional block diagram, explain the 8255 programmable peripheral interface and its operating modes?
2. Draw and discuss the internal architecture of USART 8251.
3. Explain the operation of 8259 Programmable Interface Controller with a neat diagram.
4. (a) Explain different types of semiconductor memories.

   (b) Interface two 4K×8 EPROMS and two4K×8 RAM chips with 8086.

5. With a neat diagram, explain in detail about DMA controller.
6. (a) With neat block diagram explain ADC interfacing with 8086.

   (b) With neat block diagram explain DAC interfacing with 8086.

7. Interface an 8255 with 8086 to work as an I/O port. Initialize port A as output port, port B as input port and port C as output port. Port A address should be 0740H. Write a program to sense switch positions SW_0-SW_7 connected at port B. The sensed pattern is to be displayed on port A, to which 8 LEDs are connected, while the port C lower displays number of on switches out of the total eight switches.

8. (a) Explain How stepper motor is interfacing with 8086.

   (b) Write an ALP to rotate Stepper motor clock wise and ani-clock wise direction.

2Maks Questions

1. What is the difference between serial and parallel communication?
2. Write different types of data transfer methods?
3. List the Applications Hardware and software interrupts?
4. What is the need of PIC?
5. What is the difference between RAM and ROM?
6. What is DMA and how DMA initiated?
7. Difference between Asynchronous and Synchronous communication?
8. Define baud-rate?
9. What are the features of 8255.
10. What is USART? List Features of 8251
11. Features of PIC microcontroller.
12. Features of 8237
13. List the operating modes of 8255.