9.5 Control Word Formats

A high on the RESET pin causes all 24 lines of the three 8-bit ports to be in the input mode. All flip-flops are cleared and the interrupts are reset. This condition is maintained even after the RESET goes low. The ports of the 8255 can then be programmed for any other mode by writing a single control word into the control register, when required.

9.5.1 For Bit Set/Reset Mode

Fig. 9.4 shows bit set/reset control word format.

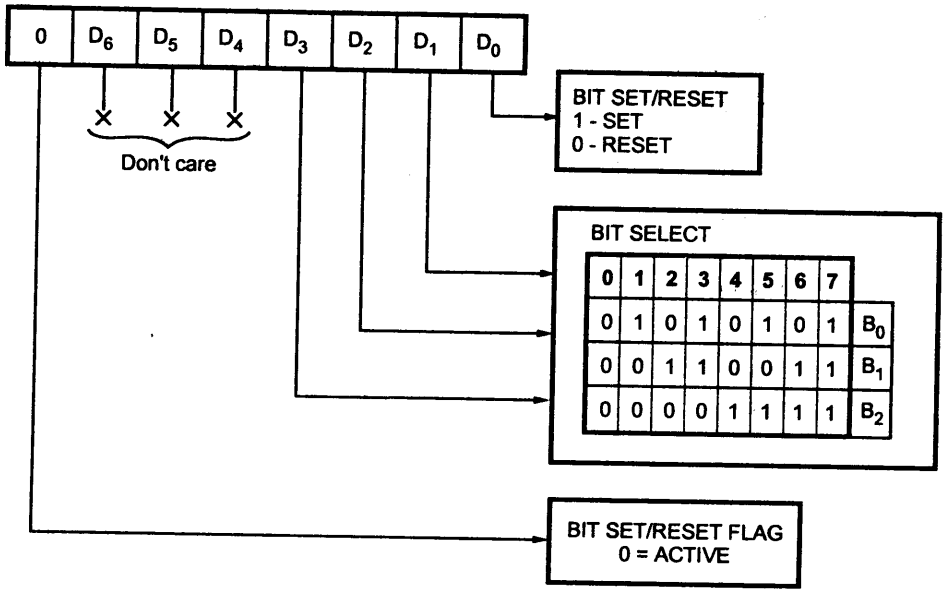


Fig. 9.4 Bit set/reset control word format

The eight possible combinations of the states of bits D_3 - D_1 (B_2 B_1 B_0) in the Bit Set-Reset format (BSR) determine particular bit in PC_0 - PC_7 being set or reset as per the status of bit D_0. A BSR word is to be written for each bit that is to be set or reset. For example, if bit PC_3 is to be set and bit PC_4 is to be reset, the appropriate BSR words that will have to be loaded into the control register will be, 0XXX0111 and 0XXX1000, respectively, where X is don't care.

The BSR word can also be used for enabling or disabling interrupt signals generated by Port C when the 8255 is programmed for Mode 1 or 2 operation. This is done by setting or resetting the associated bits of the interrupts. This is described in detail in next section.

9.5.2 For I/O Mode

The mode definition format for I/O mode is shown in Fig. 9.5. The control words for both, mode definition and Bit Set-Reset are loaded into the same control register, with bit D_7 used for specifying whether the word loaded into the control register is a mode definition word or Bit Set-Reset word. If D_7 is high, the word is taken as a mode definition word, and if it is low, it is taken as a Bit Set-Reset word. The appropriate bits are set or reset depending on the type of operation desired, and loaded into the control register.

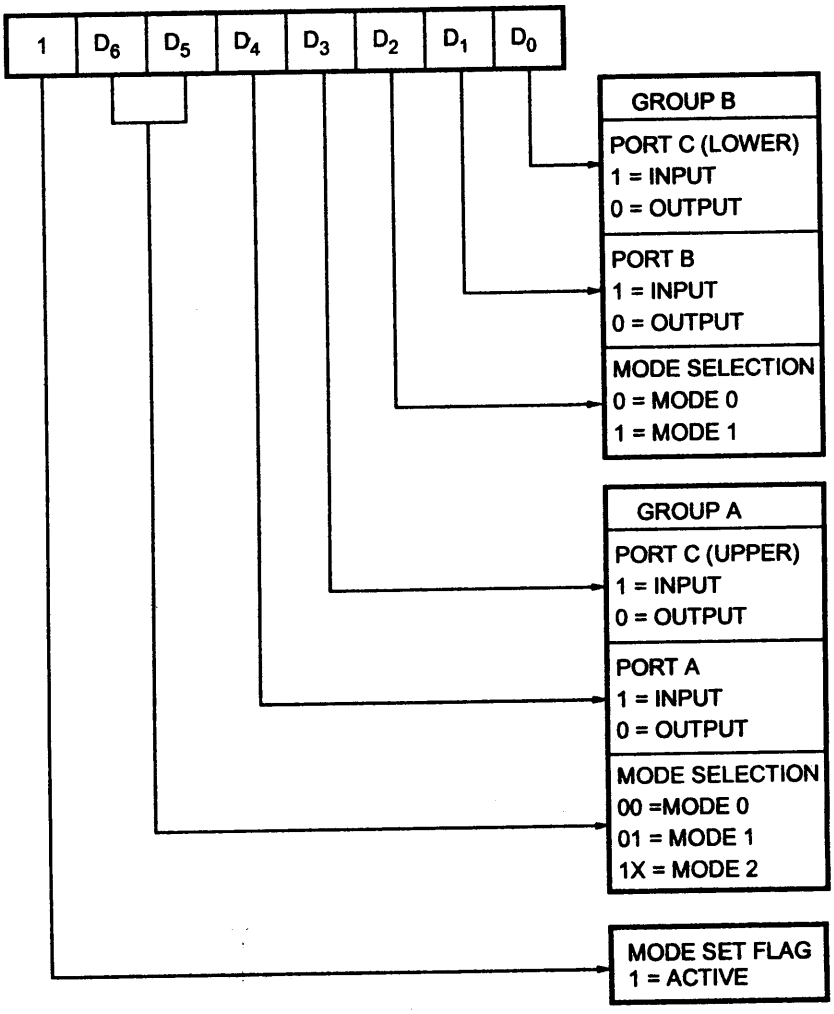


Fig. 9.5 8255 Mode definition format

Example 9.1: *Write a program to initialize 8255 in the configuration given below :*

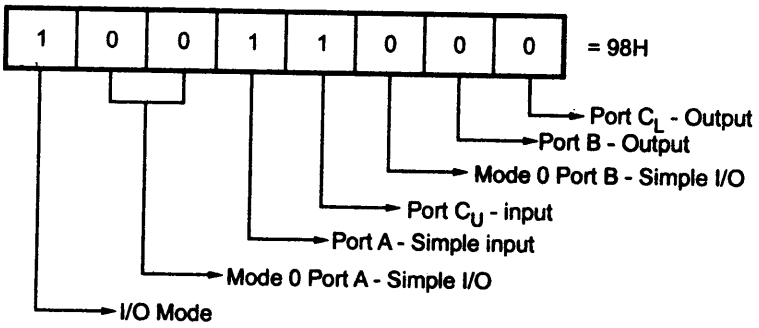
1. *Port A : Simple input*
2. *Port B : Simple output*

3. *Port* C_L *: Output*

4. *Port* C_U *: Input*

Assume address of the control word register of 8255 is 83H.

Solution :



Source program:

```
MOV AL,98H        ; Load control word
OUT 83H,AL        ; Send control word
```

Example 9.2 : *Write a program to initialize 8255 in the configuration given below :*

1. *Port A : Output with handshake*

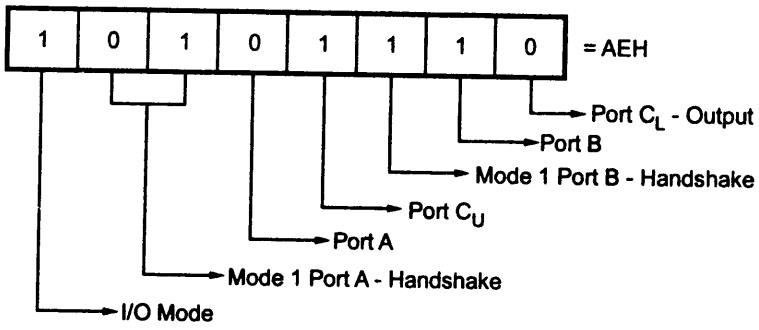
2. *Port B : Input with handshake*

3. *Port* C_L *: Output*

4. *Port* C_U *: Input*

Assume address of the control word register of 8255 is 23H.

Solution :



Source program:

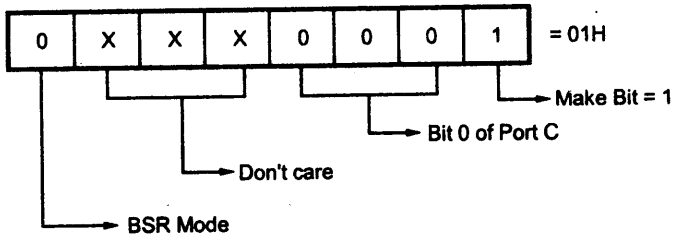
```
MOV AL,0AEH   ; Load control word
OUT 23H,AL    ; Send control word
```

Program : Blink port C bit 0 of 8255.

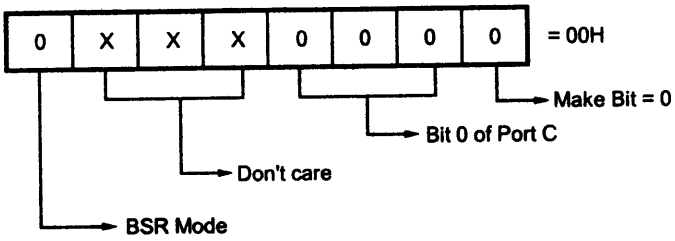
Program Statement :

Write a program to blink Port C bit 0 of the 8255. Assume address of control word register of 8255 is 83H. Use Bit Set/Reset mode.

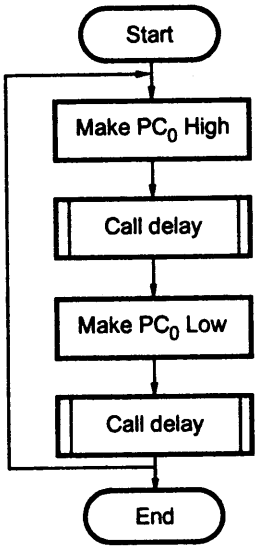
Solution : Control word to make Bit 0 high.



Control word to make Bit 0 low



Flow chart :



Source program :

```
BACK:   MOV AL,01H    ; Load bit pattern to make PC0 high
        OUT 83H,AL    ; Send it to control word register
        CALL DELAY    ; Call Delay subroutine
        MOV AL,00H    ; Load bit pattern to make PC0 Low
        OUT 83H,AL    ; Send it to control word register
        CALL Delay    ; Call Delay subroutine
        JMP BACK      ; Repeat
```

9.6 8255 Programming and Operation

9.6.1 Programming in Mode 0

The Ports A, B and C can be configured as simple input or output ports by writing the appropriate control word in the control word register. In the control word, D_7 is set to '1' (to define a mode set operation) and D_6, D_5 and D_2 are all set to '0' to configure all the ports in Mode 0 operation. The status of bits D_4, D_3, D_1 and D_0 then determine (refer to Fig. 9.5) whether the corresponding ports are to be configured as Input or Output.

For example in mode 0, if Port A and Port B are to operate as output ports with Port C lower as input, and Port C upper as output, the control word that will have to be loaded into the control register will be as follows.

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
1	0	0	0	0	0	0	1	= 81H

As mentioned earlier, this mode provides simple input and output operations for each of the three ports. No handshaking is required, data is simply written to or read from a specified port.

INPUT MODE : Fig. 9.6 shows the timing diagram for mode 0 input mode.

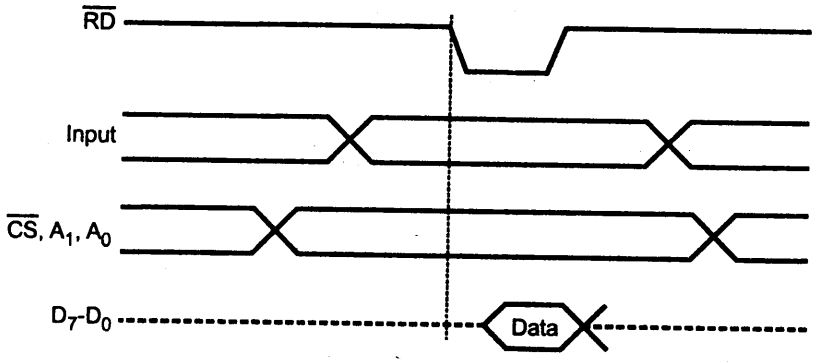


Fig. 9.6 Timing diagram for mode 0 input mode

After initialization of 8255 in the input mode 0, CPU can read data through the input port by initiating read command with proper port address. Read command activates $\overline{RD}$ signal. Upon activation of $\overline{RD}$ signal CPU reads the data from the selected input port into the CPU register.

OUTPUT MODE :

Fig. 9.7 shows the timing diagram for mode 0 output mode.

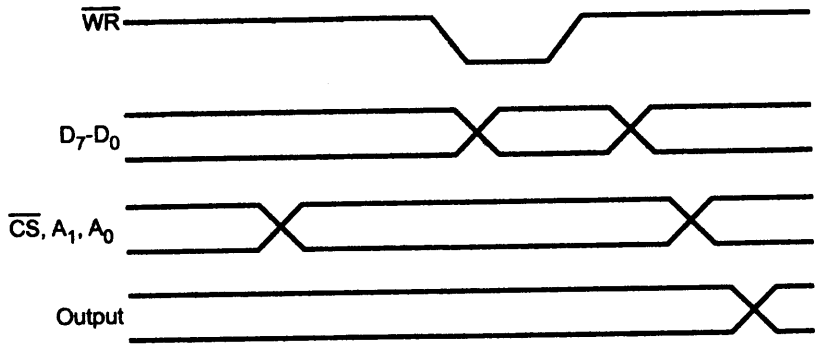


Fig. 9.7 Timing diagram for mode 0 output mode

After initialization of 8255 in the output mode 0, CPU can write data into the output port by initiating write command with proper port address. CPU sends data on the data bus and upon activation of $\overline{WR}$ signal, data on the data bus gets latched on the selected output port.

Mode 0 Configurations :

A		B		GROUP A			GROUP B	
D_4	D_3	D_1	D_0	PORT A	PORT C (Upper)	#	PORT B	PORT C (Lower)
0	0	0	0	OUTPUT	OUTPUT	0	OUTPUT	OUTPUT
0	0	0	1	OUTPUT	OUTPUT	1	OUTPUT	INPUT
0	0	1	0	OUTPUT	OUTPUT	2	INPUT	OUTPUT
0	0	1	1	OUTPUT	OUTPUT	3	INPUT	INPUT
0	1	0	0	OUTPUT	INPUT	4	OUTPUT	OUTPUT
0	1	0	1	OUTPUT	INPUT	5	OUTPUT	INPUT
0	1	1	0	OUTPUT	INPUT	6	INPUT	OUTPUT
0	1	1	1	OUTPUT	INPUT	7	INPUT	INPUT
1	0	0	0	INPUT	OUTPUT	8	OUTPUT	OUTPUT
1	0	0	1	INPUT	OUTPUT	9	OUTPUT	INPUT
1	0	1	0	INPUT	OUTPUT	10	INPUT	OUTPUT
1	0	1	1	INPUT	OUTPUT	11	INPUT	INPUT
1	1	0	0	INPUT	INPUT	12	OUTPUT	OUTPUT
1	1	0	1	INPUT	INPUT	13	OUTPUT	INPUT
1	1	1	0	INPUT	INPUT	14	INPUT	OUTPUT
1	1	1	1	INPUT	INPUT	15	INPUT	INPUT

9.6.2 Programming in Mode 1 (Input / Output with Handshake)

Both Group A and Group B can operate in Mode 1, either together, or individually, with each port containing an 8-bit latched Input or Output data port, and a 4-bit port which is used for control and status of the 8-bit port.

When Port A is to be programmed as an input port, PC_3 , PC_4 and PC_5 are used for control. PC_6 and PC_7 are not used and can be Input or Output, as programmed by bit D_3 of the control word. When Port A is programmed as an output port, PC_3 , PC_6, and PC_7 are used for control and PC_4 and PC_5 can be Input or Output, as programmed by bit D_3, of the control word.

When port B is to be programmed as an input or output port, PC_0 , PC_1 and PC_2 are used for control.

Mode 1 Input Control Signals :

1. $\overline{STB}$ (Strobe Input) :

This is an active low input signal for 8255 and output signal for the input device. The input device activates this signal to indicate CPU that the data to be read is already sent on the port lines of 8255 port. Upon activation of this signal 8255 loads the data from the input port lines into the input buffer of that port.

2. IBF (Input Buffer Full) :

This is an active high output signal for 8255 and an input signal for input device. This signal is generated by 8255 in response to $\overline{STB}$ signal as an acknowledgment to input device. It also indicates to the input device that the input buffer is full and it is not ready to accept next byte from the input device. Therefore input device sends data on the port lines only when IBF signal is not active. The IBF signal is deactivated when CPU reads the data from input buffer of the respective port by activation of $\overline{RD}$ signal.

3. INTR (Interrupt Request) :

This is an active high output signal generated by 8255. A 'high' on this output can be used to interrupt the CPU when an input device is requesting service. The 8255 sets the INTR when $\overline{STB}$ signal is 'one', IBF signal is 'one' and INTE is 'one', indicating CPU that the data from the input device is available in the input buffer. This signal is reset by the falling edge of the $\overline{RD}$ signal i.e. immediately after reading the data from the input buffer.

INTE (Interrupt Enable) flip-flop is used to enable or disable INTR (Interrupt request) signal. If INTE flip-flop is set, the interrupt request is generated depending on the status of $\overline{STB}$ and IBF signals. If INTE flip-flop is reset, the interrupt request is not generated, allowing masking facility for the interrupt.

Mode 1 : Port A Input Operation

Fig. 9.8 (a) shows Port A as an input port along with the control word and control signals (for handshaking with a peripheral). When the control word (as in Fig. 9.8 (a) is loaded into the control register, Group A is configured in Mode 1 with Port A as an input port. Port A can accept parallel data from a peripheral (like a keyboard) and this data can be read by the CPU. The peripheral first loads data into Port by making the $\overline{STB}_A$ input low. This latches the data placed by the peripheral on the common data bus into Port A. Port A acknowledges reception of data by making IBF_A (Input Buffer Full) high. IBFA is set when the $\overline{STB}_A$ input is made low, as shown in Fig. 9.8 (b).

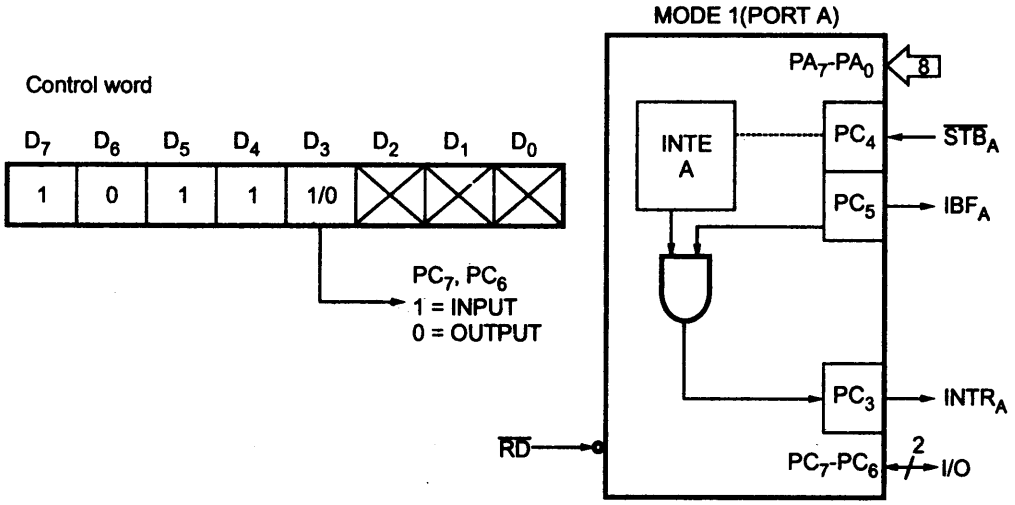


Fig. 9.8 (a) Port A in mode 1

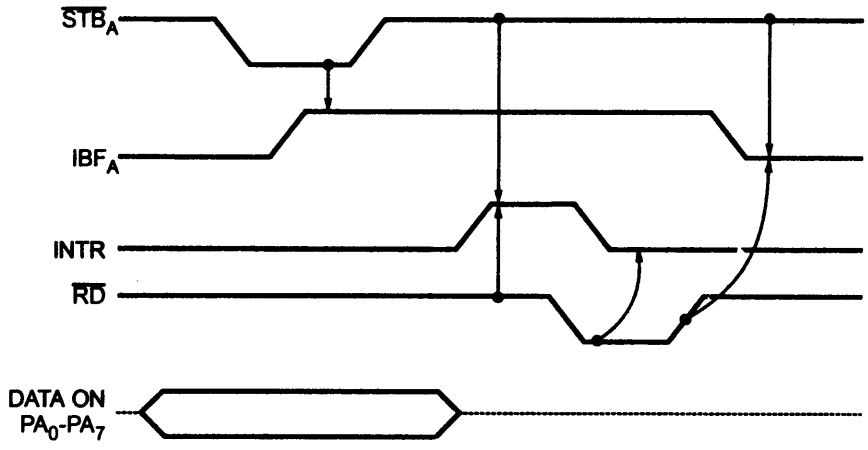


Fig. 9.8 (b) Timing diagram for port A in mode 1

$INTR_A$ is an active high output signal which can be used to interrupt the CPU so that the CPU can suspend its current operation and read the data written into Port A by the

peripheral. $INTR_A$ can be enabled or disabled by the INTEA flip-flop which is controlled by Bit Set-Reset operation of PC_4. $INTR_A$ is set (if enabled by setting the $INTE_A$ flip-flop) after the $\overline{STB}_A$ has gone high again, and if IBF_A is high.

On receipt of the interrupt, the CPU can be forced to read Port A. The falling edge of the $\overline{RD}$ input resets IBFA and it goes low. This can be used to indicate to the peripheral that the input buffer is empty and that data can again be loaded into it.

Mode 1 : Port B Input operation

Fig. 9.9 shows Port B as an input port (when in Mode 1). The timing diagram and operation of Port B is similar to that of Port A except that it uses different bits of Port C for control. $INTE_B$ is controlled by Bit Set/Reset of PC_2.

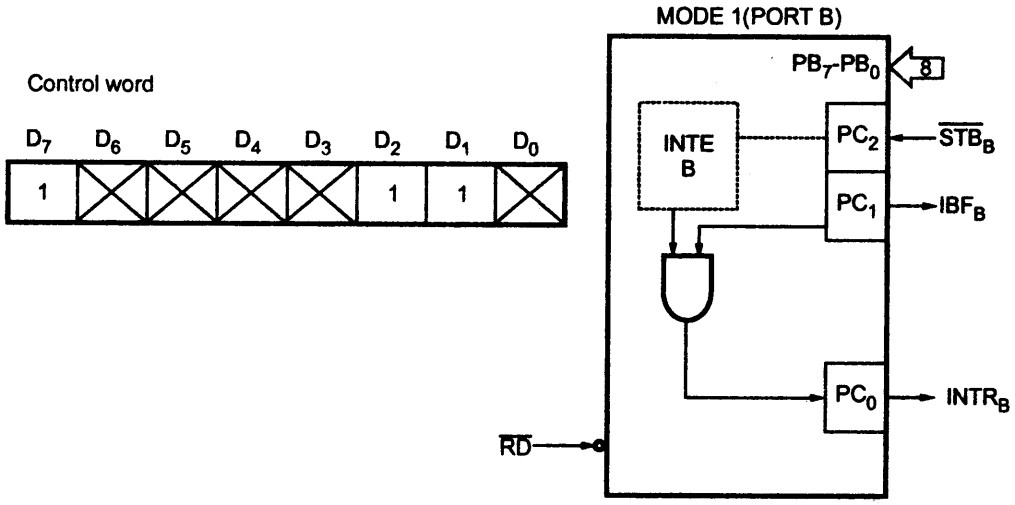


Fig. 9.9 Port B in mode 1

If the CPU is busy with other system operations, it can read data from the input port when it is interrupted. This is often called Interrupt driven I/O. However, if the CPU is otherwise not busy with other jobs, it can continuously poll (read) the status word to check for an IBF_A. This is often called Program Controlled I/O. The status word is accessed by reading Port C (A_1 A_0 must be 10, $\overline{RD}$ and $\overline{CS}$ must be low). The status word format when Ports A and B are input ports in Mode 1, is shown in Fig. 9.10.

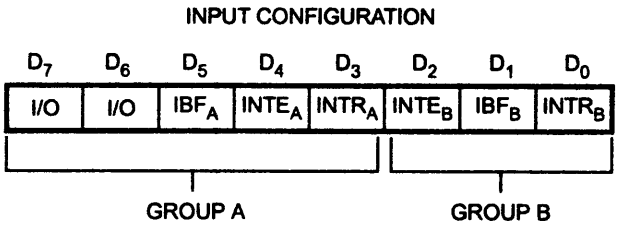


Fig. 9.10 Mode 1 status word (Input)

Mode 1 : Output control signals

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

This is an active low output signal for 8255 and input signal for the output device. The 8255 activates this signal to indicate output device that data is available on the output port. Upon activation of $\overline{OBF}$ signal, output device reads data from the output port and acknowledges it by $\overline{ACK}$ signal. The $\overline{OBF}$ signal is activated at the rising edge of the $\overline{WR}$ signal and de-activated at the falling edge of the $\overline{ACK}$ signal.

2. $\overline{ACK}$ (Acknowledge Input) :

This is an active low input signal for 8255 and output signal for the output device. The output device generates this signal to indicate 8255 that the data from port A or Port B has been accepted.

3. INTR (Interrupt Request) :

This is an active high output signal generated by 8255. A 'high' on this output can be used to interrupt the CPU when an output device has accepted data transmitted by the CPU. The 8255 sets the INTR when $\overline{ACK}$ signal is 'one', $\overline{OBF}$ is 'one' and INTE is 'one', indicating that the output device is ready to accept next data byte. This signal is reset by the falling edge of the $\overline{WR}$ signal i.e. immediately after sending the data to the output port.

INTE (Interrupt Enable) flip-flop is used to enable or disable INTR (Interrupt Request) signal. If INTE flip-flop is set, the interrupt request is generated depending on the status of $\overline{ACK}$ and $\overline{OBF}$ signals. If INTE flip-flop is reset, the interrupt request is not generated, allowing masking facility for the interrupt.

Mode 1 : Port A output operation

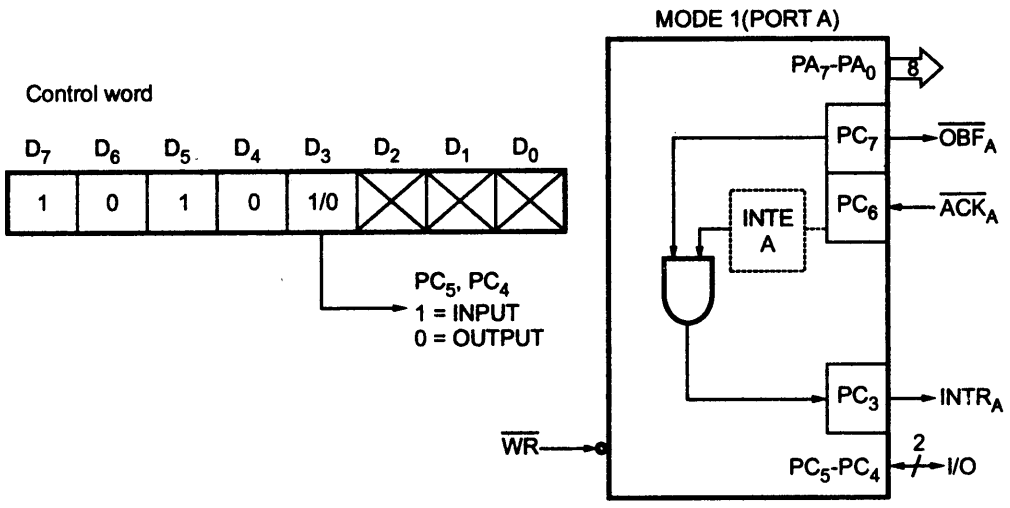


Fig. 9.11 (a) Port A in mode 1

Fig. 9.11 (a) shows Port A configured as an output port (When in Mode 1) along with the control word and control signals (for handshaking with a peripheral). When the control word (as in Fig. 9.11 (a)) is loaded into the control register, Group A is configured in Mode 1 with Port A as an output port. The CPU can send data to a peripheral (like a display device) through Port A of the 8255.

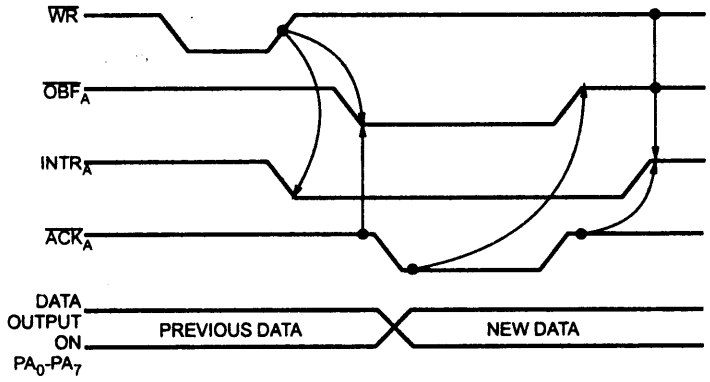


Fig. 9.11 (b) Port A in mode 1 (Output)

The $\overline{OBF}_A$ output (Output Buffer Full) goes low on the rising edge of the $\overline{WR}$ signal (when the CPU writes data into the 8255). The $\overline{OBF}_A$ output from 8255 can be used as a strobe input to the peripheral to latch the contents of Port A. The peripheral responds to the receipt of data by making the $\overline{ACK}_A$ input of the 8255 low, thus acknowledging that it has received the data sent by the CPU through Port A. The $\overline{ACK}_A$ low sets the OBF_A signal, which can be polled by the CPU through OBF_A of the status word to load the next data when it is high again.

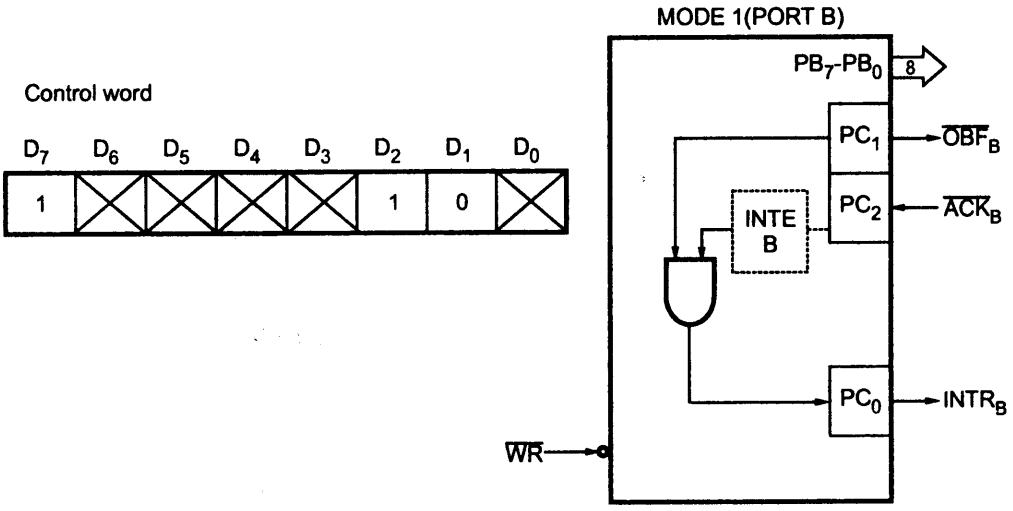


Fig. 9.12 Port B in mode 1 (Output)

$INTR_A$ is an active high output of the 8255 which is made high (if the associated $INTE_A$ flip-flop is set) when $\overline{ACK}_A$ is made high again by the peripheral, and when $\overline{OBF}_A$

goes high again (see timing diagram in Fig. 9.11). It can be used to interrupt the CPU whenever the output buffer is empty. It is reset by the falling edge of $\overline{WR}$ when the CPU writes data onto Port A. It can be enabled or disabled by writing a '1' or a '0' respectively to PC_6 in the BSR mode.

Mode 1 : Port B output operation

Fig. 9.12 shows Port B as an output port when in Mode 1. The operation of Port B is similar to that of Port A. $INTR_A$ is controlled by writing a '1' or a '0' to PC_2 in the BSR mode.

The status word is accessed by issuing a Read to Port C. The format of the status word when Ports A and B are Output ports in Mode 1 is shown in Fig. 9.13.

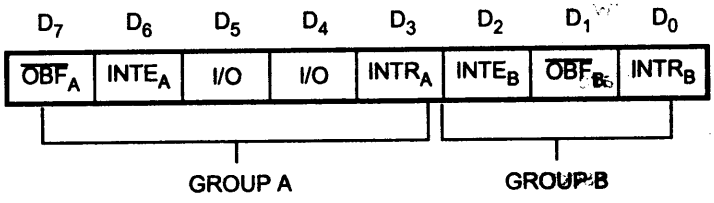


Fig. 9.13 Mode 1 status word (Output)

9.7 Programming in Mode 2 (Strobes Bi-directional Bus I/O)

When the 8255 is operated in Mode 2 (by loading the appropriate control word), Port A can be used as a bi-directional 8-bit I/O bus using for handshaking. Port B can be programmed in Mode 0 or in Mode 1. When Port B is programmed in mode 1, PC_0 - PC_2 lines of Port C are used as handshaking signals.

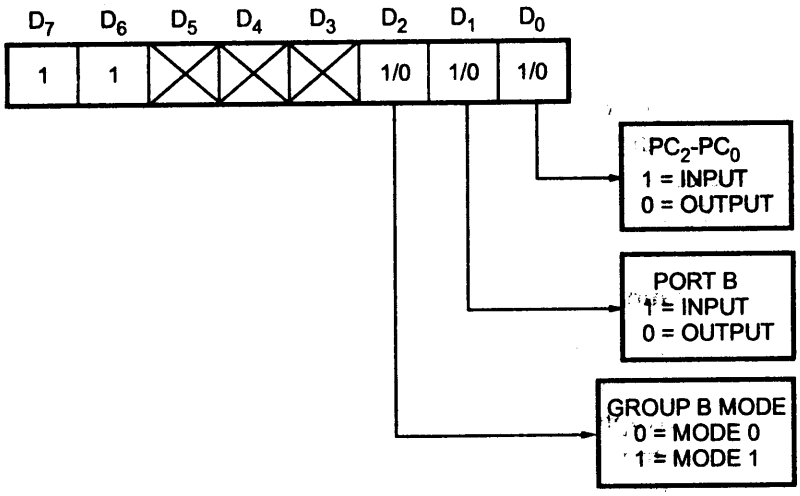


Fig. 9.14 Mode 2 control word

Fig. 9.14 shows the control word that should be loaded into the control port to configure 8255 in Mode 2.

Mode 2 : Control signals

INTR (Interrupt Request) : A 'high' on this output can be used to interrupt the CPU for input or output operations.

Output Control Signals :

$\overline{OBF}_A$ (Output Buffer Full)

This is an active low output which indicates that the CPU has written data into Port A.

$\overline{ACK}_A$ (Acknowledge)

This is an active low input signal (generated by the peripheral) which enables the tri-state output buffer of Port A and makes Port A data available to the peripheral. In Mode 2, Port A outputs are in tri-state until enabled.

INTE 1

This is the flip-flop associated with Output Buffer Full. INTE 1 can be used to enable or disable the interrupt by setting or resetting PC_6 in the BSR Mode.

Input Control Signals :

$\overline{STB}$ (Strobe Input)

This is an active low input signal which enables Port A to latch the data available at its input.

IBF (Input Buffer Full Flip-Flop)

This is an active high output which indicates that data has been loaded into the input latch of Port A.

INTE 2

This is an Interrupt enable flip-flop associated with Input Buffer Full. It can be controlled by setting or resetting PC_4 in the BSR Mode.

Mode 2 : Port A operation.

Fig. 9.15 shows Port A and associated control signals when 8255 is in Mode 2. Interrupts are generated for both output and input operations on the same $INTR_A$ (PC_3) line.

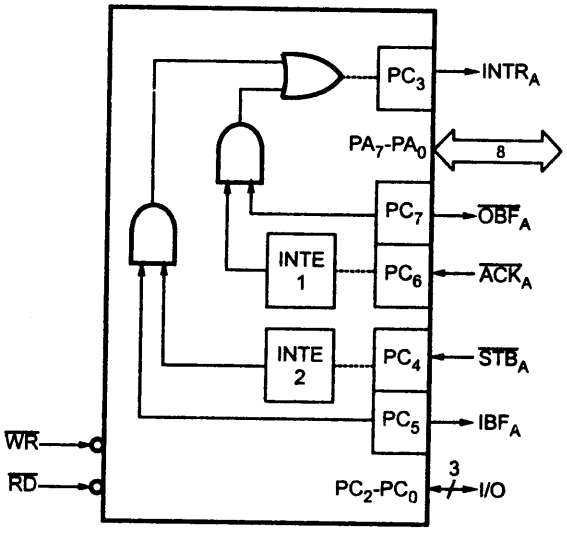


Fig. 9.15 Mode 2 operation

Status Word In Mode 2

The status word for Mode 2 (accessed by reading Port C) is shown in Fig. 9.15. D_7 - D_3 of the status word carry information about $\overline{OBF}_A$, $INTE_1$, IBF_A, $INTE_2$, $INTR_A$. The status of the bits D_2 - D_0 depend on the mode setting of Group B. If B is programmed in Mode 0, D_2 - D_0 are the same as PC_2 - PC_0 (simple I/O); however if B is in Mode 1, D_2 - D_0 carry information about the control signals for Port B (as in Fig. 9.10, or Fig. 9.13), depending upon whether Port B is an Input port or Output port respectively.

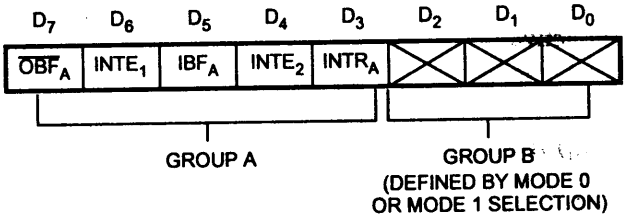


Fig. 9.16 Status word for mode 2

Mode Definition Summary

	MODE 0	
	IN	OUT
PA_0	IN	OUT
PA_1	IN	OUT
PA_2	IN	OUT
PA_3	IN	OUT
PA_4	IN	OUT
PA_5	IN	OUT
PA_6	IN	OUT
PA_7	IN	OUT
	MODE 0	
	IN	**OUT**
PB_0	IN	OUT
PB_1	IN	OUT
PB_2	IN	OUT
PB_3	IN	OUT
PB_4	IN	OUT
PB_5	IN	OUT
PB_6	IN	OUT
PB_7	IN	OUT
PC_0	IN	OUT
PC_1	IN	OUT
PC_2	IN	OUT
PC_3	IN	OUT
PC_4	IN	OUT
PC_5	IN	OUT
PC_6	IN	OUT
PC_7	IN	OUT

MODE 1	
IN	OUT
IN	OUT
IN	OUT
IN	OUT
IN	OUT
IN	OUT
IN	OUT
IN	OUT
MODE 1	
IN	**OUT**
IN	OUT
IN	OUT
IN	OUT
IN	OUT
IN	OUT
IN	OUT
IN	OUT
IN	OUT
$INTR_B$	$INTR_B$
IBF_B	OBF_B
STB_B	ACK_B
$INTR_A$	$INTR_A$
STB_A	I/O
IBF_A	I/O
I/O	ACK_A
I/O	OBF_A

MODE 2
GROUP A ONLY
↔
↔
↔
↔
↔
↔
↔
↔
MODE 2
GROUP A ONLY
–
–
–
–
–
–
–
–
I/O
I/O
I/O
$INTR_A$
STB_A
IBF_A
ACK_A
OBF_A

Mode 0 or Mode 1 Only

9.8 Interfacing 8255 to 8086 in I/O Mapped I/O Mode

The 8086 has four special instructions IN, INS, OUT, and OUTS to transfer data through the input/output ports in I/O mapped I/O system. M/$\overline{IO}$ signal is always low when 8086 is executing these instructions. So M/$\overline{IO}$ signal is used to generate separate addresses for, memory and input/output. Only 256 (2^8) I/O addresses can be generated when direct addressing method is used. By using indirect address method this range can be extended upto 65536 (2^{16}) addresses.

Fig. 9.17 shows the interfacing of 8255 with 8086 in I/O mapped I/O technique. Here, $\overline{RD}$ and $\overline{WR}$ signals are activated when M/$\overline{IO}$ signal is low, indicating I/O bus cycle. Only lower data bus (D_0 - D_7) is used as 8255 is 8-bit device. Reset out signal from clock generator is connected to the Reset signal of the 8255. In case of interrupt driven I/O INTR signal (PC_3 or PC_0) from 8255 is connected to INTR input of 8088.

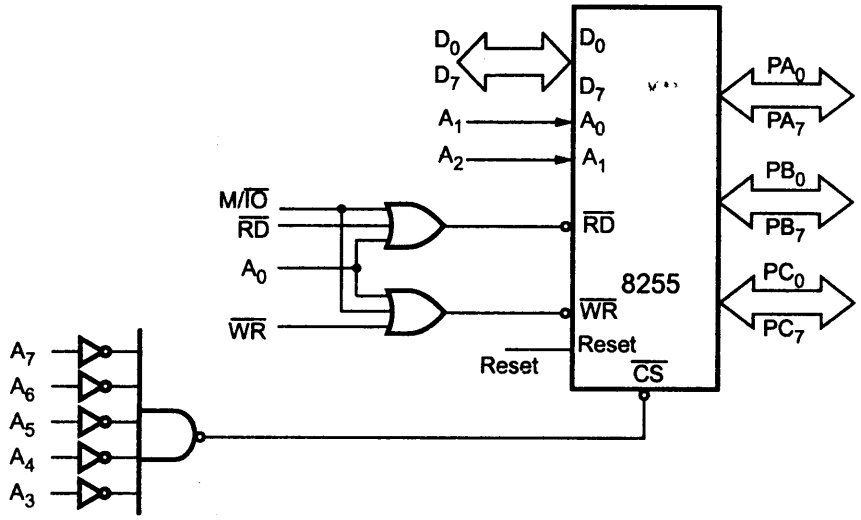


Fig. 9.17 I/O mapped I/O

I/O Map :

Port / control Register	Address lines								Address
	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	
Port A	0	0	0	0	0	0	0	0	00H
Port B	0	0	0	0	0	0	1	0	02H
Port C	0	0	0	0	0	1	0	0	04H
Control register	0	0	0	0	0	1	1	0	06H

Note : It is assumed that the direct addressing is used.

9.9 Interfacing 8255 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. 9.18 shows the interfacing of 8255 with 8086 in memory mapped I/O technique. Here $\overline{RD}$ and $\overline{WR}$ signals are activated when M/$\overline{IO}$ signal is high, indicating memory bus cycle. Address lines A_0 - A_1 are used by 8255 for internal decoding. To get absolute address, all remaining address lines (A_3 - A_{19}) are used to decode the address for 8255. Other signal connections are same as in I/O mapped I/O.

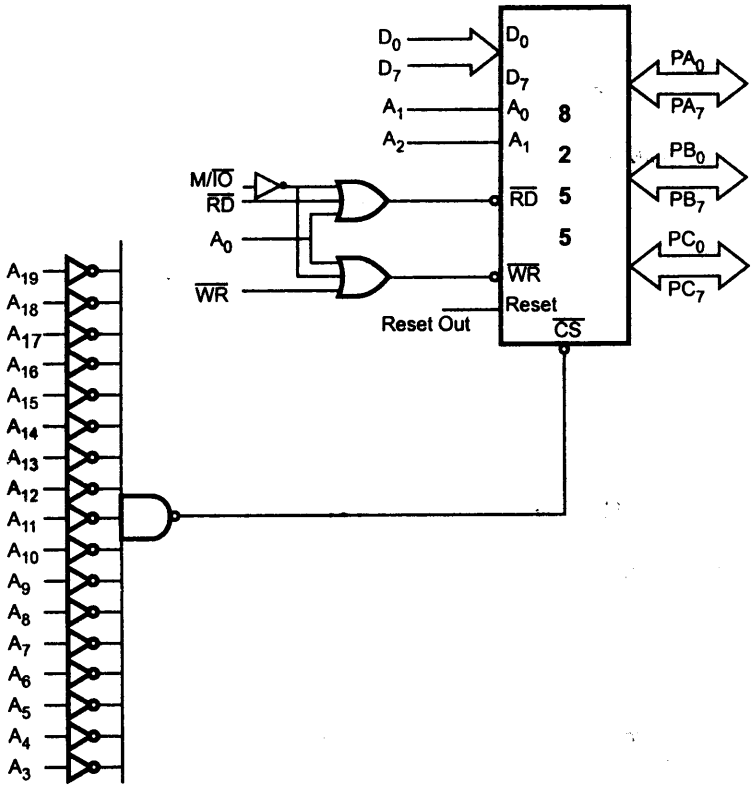


Fig. 9.18 Memory mapped I/O

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
Port A	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	00000H
Port B	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	00002H
Port C	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	00004H
Control register	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	00006H

9.10 Interfacing 8255 to 8088 in I/O Mapped I/O Mode

Fig. 9.19 shows the interfacing of 8255 with 8088 in I/O mapped I/O technique. Here, $\overline{RD}$ and $\overline{WR}$ signals are activated when IO/$\overline{M}$ signal is high, indicating I/O bus cycle. Address lines A_0 and A_1 are connected to A_0 and A_1 lines of 8255. The reset out signal from clock generator is connected to the reset input of the 8255.

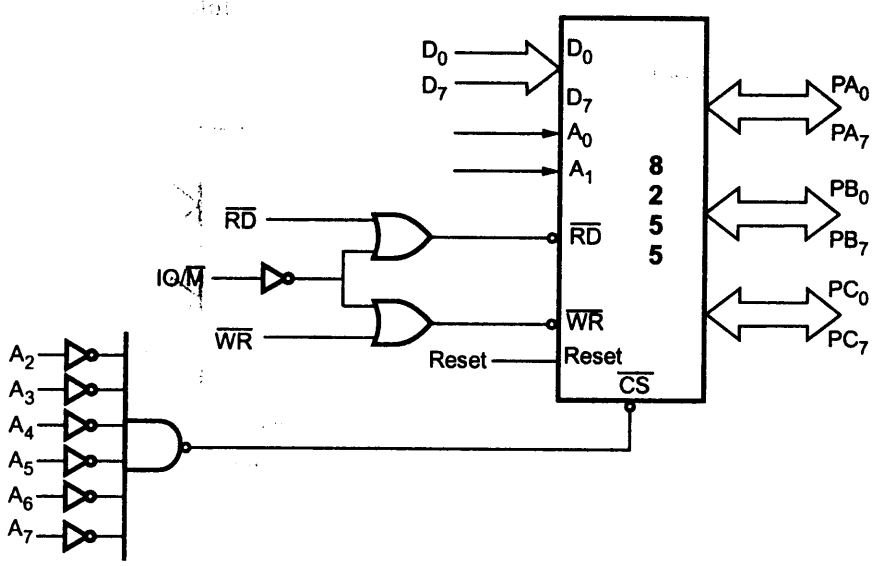


Fig. 9.19 Interfacing 8255 to 8088 in I/O mapped I/O

In case of interrrupt driven I/O, INTR signal (PC_3 or PC_0) from 8255 is connected to INTR input of 8089.

I/O map :

Port / Control register	Address lines								Address
	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	
Port A	0	0	0	0	0	0	0	0	00H
Port B	0	0	0	0	0	0	0	1	01H
Port C	0	0	0	0	0	0	1	0	02H
Control Register	0	0	0	0	0	0	1	1	03H

Note : it assumed that the direct addressing is used.

9.11 Interfacing 8255 to 8088 in Memory Mapped I/O

In this type of I/O interfacing, the 8088 uses 20 address lines to identify an I/O device; an I/O device is connected as if it is a memory register. The 8088 uses same control signals and instructions to access I/O as those of memory. Fig. 9.20 shows the interfacing of 8255 with 8088 in memory mapped I/O technique. Here $\overline{RD}$ and $\overline{WR}$ signals are activated when IO/$\overline{M}$ signal is low, indicating memory bus cycle. Address lines A_0 - A_1 are used by 8255 for internal decoding. To get absolute address, all remaining address lines (A_2 - A_{19}) are used to decode the address for 8255. Other signal connections are same as in I/O mapped I/O.

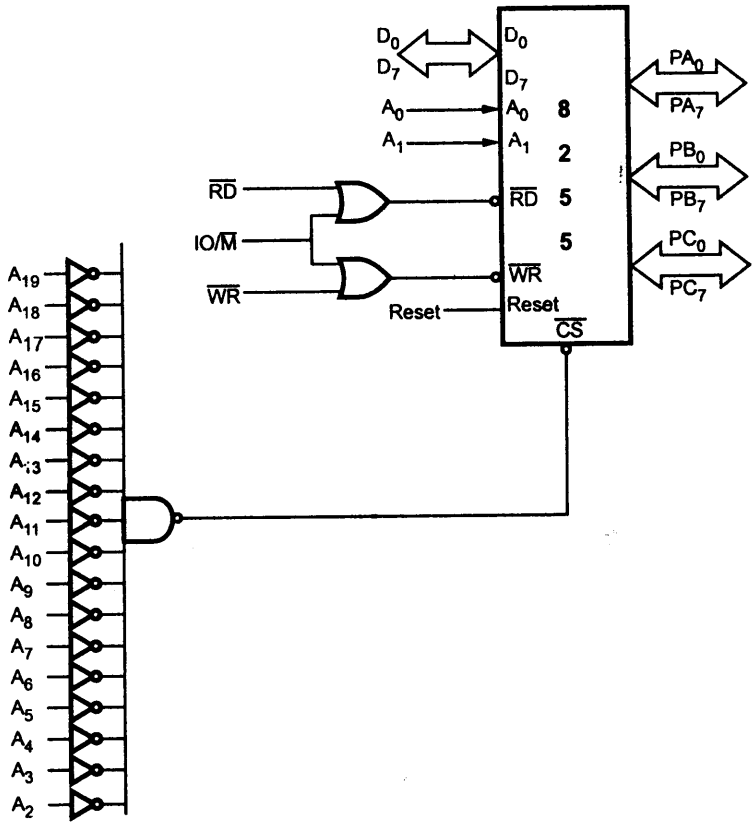


Fig. 9.20 Memory mapped I/O

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
Port A	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	00000H
Port B	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	00001H
Port C	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	00002H
Control Register	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	00003H

9.12 Centronics Parallel Printer Interface

As explained earlier, handshaking signals are required to transfer data between two devices whose speeds are not same. This centronics protocol is a printer protocol, gives standards for printer interface.

It has 36 pins. The Fig. 9.21 shows the pin definitions for centronics interface. The ASCII characters are sent to the printer through eight data lines. Each data line has individual ground to reduce the change of picking up electrical noise in the lines.

Signal Pin No.	Reutrn Pin No.	Signal	Direction	Description
1	19	$\overline{\text{STROBE}}$	IN	STROBE pulse to read data in. Pulse width must be more than 0.5 ms at receiving terminal. The signal level is normally "high": read-in of data is performed at the "low" level of this signal.
2	20	DATA 1	IN	
3	21	DATA 2	IN	
4	22	DATA 3	IN	These signals represent information of the 1st to 8th this of parallel data respectively. Each signal is at "high" level when data logical "I" and "low" when logical "0".
5	23	DATA 4	IN	
6	24	DATA 5	IN	
7	25	DATA 6	IN	
8	26	DATA 7	IN	
9	27	DATA 8	IN	
10	28	$\overline{\text{ACKNLG}}$	OUT	Approximately 5 ms pulse; "low" indicates that data has been received and the printer is ready to accept other data.
11	29	BUSY	OUT	A "high" signal indicates that the printer cannot receive data The signal becomes "high" in the following cases: 1. During data entry. 3. In "office" state. 2. During printing operation.4. During printer error status.
12	30	PE	OUT	A "high" signal indicates that the printer is out of paper.
13	--	SLCT	OUT	This signal indicates that the printer is in the selected state.
14	--	$\overline{\text{AUTO FEED XT}}$	IN	When this signal being at "low" level, the paper is automatically fed one line after printing. (The signal level can be fixed to "low" with DIP SW pin 2-3 provided on the control circuit board).
15	--	NC		Not used.
16	--	OV		Logic GND level.
17	--	CHASIS-GND	--	Printer chasis GND. In the printer, the chasis GND and the logic GND are isolated from each other.
18	--	NC	--	Not used.
19-30	--	GND	--	"Twisted-Pair Return" signal; GND level.
31	--	$\overline{\text{INIT}}$	IN	When the level of this signal becomes "low" the printer controller is reset to its initial state and the print buffer is cleared. This signal is normally at "high" level and its pulse with must be more then 50 μs at the receiving terminal.

32	--	$\overline{\text{ERROR}}$	OUT	The level of this signal becomes "low" when the printer is in "Paper End" state. "Offline" state and "Error" state.
33	--	GND	--	Same as with pin numbers 19 to 30.
34	--	NC	--	Not used.
35				Pulled up to + $5V_{dc}$ through 4.7 K-ohms resistance.
36	--	$\overline{\text{SLCT IN}}$	IN	Data entry to the printer is possible only when the level of this signal is "low". (Internal fixing can be carried out with DIP SW 1-8. The condition at the time of snpment is set "low" for this signal).

Notes :

1. "Direction" refers to the direction off signal flow as viewed from the printer.
2. "Return" denotes "Twisted-Pair Return" and is to be connected at signal-ground level.

   When wikring the interface, be sure to use a twisted-pair cable for each signal and never fail to complete connection on the return side to prevent noise effectively, these cables should be shielded and connected to the chassis of the system unit.
3. All interface conditions are based on TTL level. Both the rise and fall times of each signal must be less than 0.2 μs.
4. Data transfer must not be carried out by ignoring the ACKNLG or BUSY signal. (Data transfer to this printer can be carried out only after confirming the ACKNLG signal or when the level of the BUSY signal is "low".

Fig. 9.21 Pin definitions for centronics interface

The other signals fall into two categories, signals sent to the printer to tell it which operation to do and signals from the printer that indicate its status. These signals are as follows :

a) Input signals for printer :

1. INIT : This signal when activated tells the printer to perform its internal initialization sequence.
2. STROBE ($\overline{\text{STB}}$) : This signal when activated tells the printer that valid data is available on the data bus.

b) Status signals output from printer :

1. $\overline{\text{ACK}}$: This signal when low indicates that the data character has been accepted and the printer is ready for the next data.
2. BUSY : This is active high signal. It goes high when printer is not ready to receive a character.
3. PE : This active high signal goes high when printer is out of paper.
4. SLCT : This signal goes high if the printer is selected for receiving data.
5. ERROR : This active low signal goes low for variety of problem conditions in the printer.

Fig. 9.22 shows the timing waveforms for transfer of data characters to an IBM printer using the basic handshake signals.

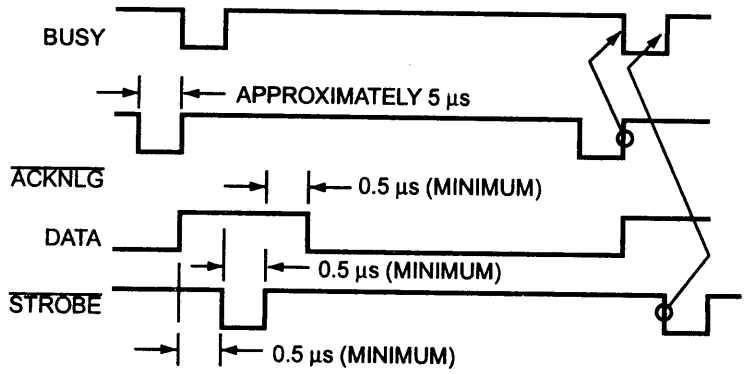


Fig. 9.22 Timing waveforms for transfer of data characters to an IBM printer

Communication between Computer and Printer

Computer sends the INIT pulse for atleast 50 µs, to initialize the printer. Computer then checks for BUSY low to confirm whether the printer is ready to receive data or not. If BUSY signal is low (not busy), computer sends an ASCII code on eight parallel data lines and after at least 0.5 µs, it also sends $\overline{STB}$ signal to indicate, valid data is available on the data bus. Computer activates this $\overline{STB}$ signal for at least 0.5 µs and it also ensures that valid data is present on the data bus for at least 0.5 µs after the $\overline{STB}$ signal is disabled. When the printer is ready to receive the next character, it asserts its $\overline{ACK}$ signal low for about 5 µs. The rising edge of the $\overline{ACK}$ signal tells the computer that it can send the next character. The rising edge of the $\overline{ACK}$ signal also resets the BUSY signal from the printer. When computer finds busy low, it sends the next character along with strobe and the sequence is repeated till the last character transfer.

Centronics Printer Interface using 8255

Fig. 9.23 shows the circuit for interfacing centronix type parallel input printer to 8255A. Port A is used to send 8-bit data to the printer. It is used in mode 1 so PC_6

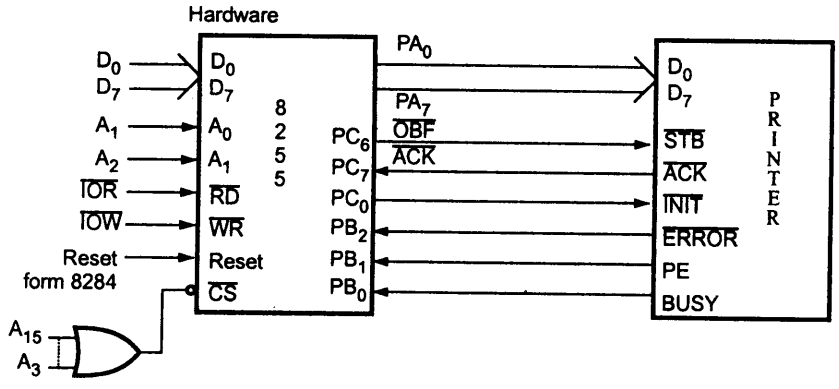


Fig. 9.23 Interfacing centronix printer to 8255A

($\overline{OBF}$ signal) is used as $\overline{STB}$ signal to tell the printer that valid data is available on the data bus and PC_7 is used as an $\overline{ACK}$ signal. BUSY, PE and $\overline{ERROR}$ signals are connected to the PB_0 to PB_2 port lines.

In the next section we will see flowchart and program required to print a message.

Flowchart :

Fig. 9.24 Flowchart for printer interface.

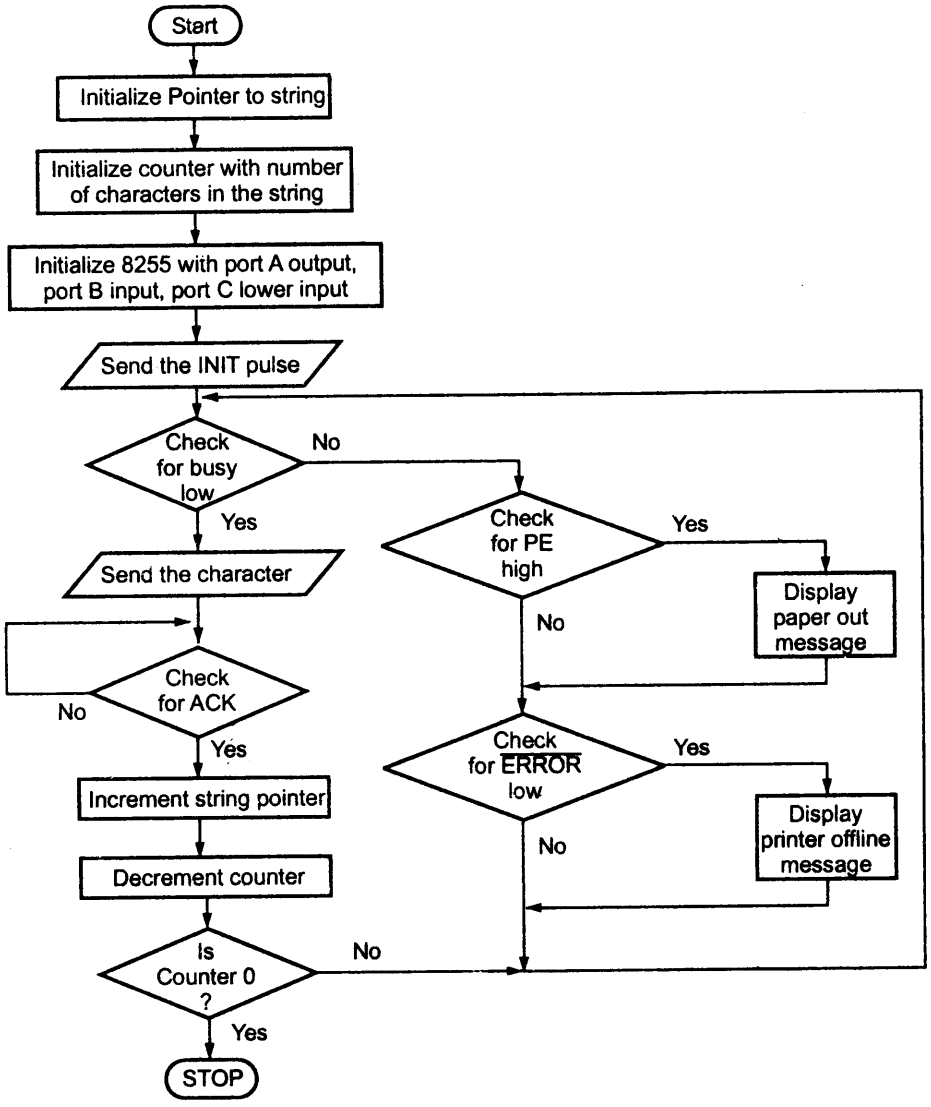


Fig. 9.24 Flowchart for printer interface

In the program it is necessary to initialize 8255 as follows

Port	Input/output	Mode
Port A	Output	1
Port B	Input	0
Port Upper	—	—
Port Lower	Output	—

Control word :

I/O	Mode A		PA	PC_U	Mode B	PB	PC_L
1	0	1	0	X	0	1	0

= A2H

I/O map :

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	Port
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000 H	Port A
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0002 H	Port B
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0004 H	Port C
0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0006 H	CR

Program :

```
                .MODEL  SMALL
                PortA           EQU     0000
                PortB           EQU     0002
                PortC           EQU     0004
                CR              EQU     0006
                .DATA
                    Mes1        DB  'Printer Paper Out', 10, 13, '$'
                    Mes2        DB  'Printer Offline', 10, 13, '$'
                    Mes3        DB  'Printing Over', '$'
                    Mes4        DB  'This is to be print'
                    COUNT       DB  15
                .CODE
START:          MOV     AX,@DATA        ; Initialize data segment
                MOV     DS,AX
                LEA     BX,MES4         ; Initialize pointer to string
                MOV     DX,CR           ; CR is control register address
                MOV     AL,0A2H         ; Load control word
                OUT     DX,AL
                MOV     AL,07           ; Make INTE_A high to enable INTR_A

                OUT     DX,AL
                MOV     AL,00
                OUT     DX,AL           ; Make PC_0 low (BSR mode)
                                        ; to give INIT low
                MOV     CX,0FFFH
```

```
BACK:   DEC     CX          ; Wait for more than 50 µs
        LOOP BACK
        MOV     AL,01
        OUT     DX,AL       ; Make INIT HIGH
NEXT:   MOV     DX,PortB
        IN      AL,DX
        MOV     AH,AL       ; Save status information
        AND     AL,01H
        JNZ     CHECK       ; Check for BUSY if high goto
                            ; check
        MOV     AL,[BX]
        MOV     DX,PortA
        OUT     DX,AL       ; Send the character
        MOV     DX,PortC
AGAIN:  IN      AL,DX       ; Check for ACK by
        AND     AL,08       ; checking INTR_A line high
        JZ      AGAIN
        INC     BX          ; Increment string pointer
        MOV     AL,COUNT
        DEC     AL
        MOV     COUNT,AL    ; Decrement counter
        JNZ     NEXT        ; Check for counter = 0
        JMP     LAST
CHECK:  MOV     AL,AH
        AND     AL,02
        MOV     AL,AH       ; Save printer status
        JZ      CHECK1
        LEA     DX,MES1
        MOV     AH,09H      ; Call for DOS interrupt
        INT  21H            ; to display MES1
CHECK1: AND     AL,04
        JNZ     NEXT
        LEA     DX,MES2
        MOV     AH,09H      ; Call for DOS interrupt to
        INT     21H         ; display MES2
        JMP     NEXT
LAST :  LEA     DX,MES3     ; Call for DOS interrupt to
        MOV     AH,09H      ; display MES3
        INT     21H
        MOV     AH,4CH      ; Terminate program
        INT     21H
        END     START
        END
```

9.13 Stepper Motor Interfacing

A stepper motor is a digital motor. It can be driven by digital signal. Fig. 9.25 shows the typical 2 phase motor interfaced using 8255. Motor shown in the circuit has two phases, with center-tap winding. The center taps of these windings are connected to the 12 V supply. Due to this, motor can be excited by grounding four terminals of the two windings. Motor can be rotated in steps by giving proper excitation sequence to these windings. The lower nibble of port A of the 8255 is used to generate excitation signals in the proper sequence.

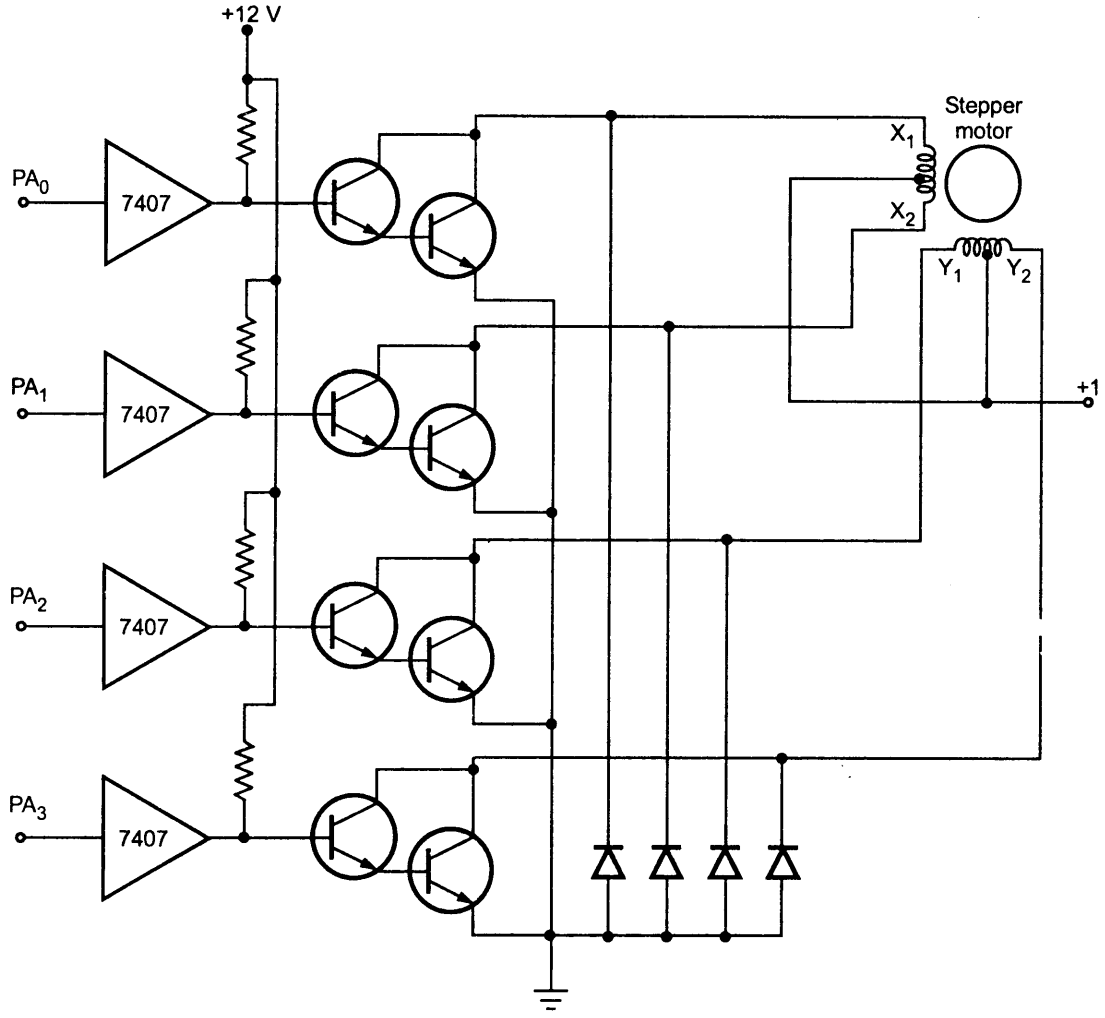


Fig. 9.25 Stepper motor interface

The Table 9.2 shows typical excitation sequence. The given excitation sequence rotates the motor in clockwise direction. To rotate motor in anticlockwise direction we have to excite motor in a reverse sequence. The excitation sequence for stepper motor many change due to change in winding connections. However, it is not desirable to excite both the ends of the same winding simultaneously. This cancels the flux and motor winding may damage. To avoid this, digital locking system must be designed. Fig. 9.26 shows a simple digital locking system. Only one output is activated (made low) when properly excited; otherwise output is disabled (made high).

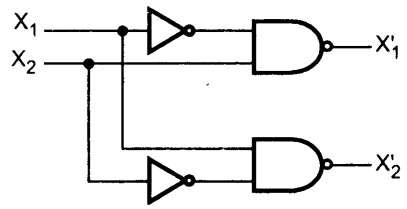


Fig. 9.26 Digital locking system

Step	X_1	X_2	Y_1	Y_2
1	0	1	0	1
2	1	0	0	1
3	1	0	1	0
4	0	1	1	0
1	0	1	0	1

Table 9.2 Full step excitation sequence

The excitation sequence given in Table 9.2 is called **full step sequence**. In which excitation ends of the phase are changed in one step. The excitation sequence given in Table 9.3 takes two steps to change the excitation ends of the phase. Such a sequence is called **half step sequence** and in each step the motor is rotated by 0.9°.

Step	X_1	X_2	Y_1	Y_2
1	0	1	0	1
2	0	0	0	1
3	1	0	0	1
4	1	0	0	0
5	1	0	1	0
6	0	0	1	0
7	0	1	1	0
8	0	1	0	0
1	0	1	0	1

Table 9.3 Half step excitation sequence

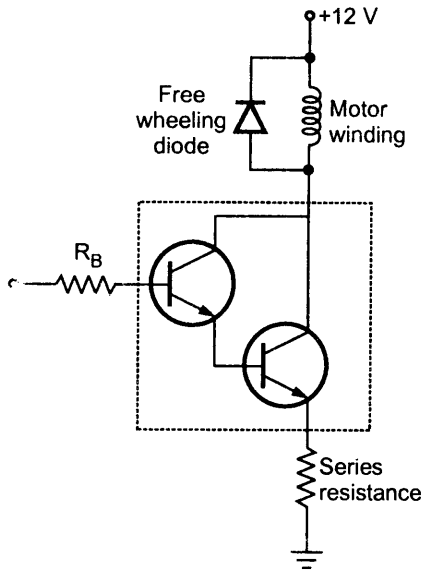


Fig. 9.27 Excitation circuit with series resistance

We know that stepper motor is stepped from one position to the next by changing the currents through the fields in the motor. The winding inductance opposes the change in current and this puts limit on the stepping rate. For higher stepping rates and more torque, it is necessary to use a higher voltage source and current limiting resistors as shown in Fig. 9.27. By adding series resistance, we decrease L/R time constant, which allows the current to change more rapidly in the windings. There is a power loss across series resistor, but designer has to compromise between power and speed.

Example 9.3 : *Interface stepper motor to the 8086 microprocessor system and write an 8086 assembly language program to control the stepper motor.*

Solution : Hardware : Fig. 9.28 shows the typical 2 phase motor rated 12V/.67A/ph interfaced with the 8086 microprocessor system using 8255. Motor shown in the circuit has two phases, with center-tap winding. The center taps of these windings are connected to the 12 V supply. Due to this, motor can be excited by grounding four terminals of the two windings. Motor can be rotated in steps by giving proper excitation sequence to these windings. The lower nibble of port A of the 8255 is used to generate excitation signals in the proper sequence. These excitation signals are buffered using driver transistors. The transistors are selected such that they can source rated current for the windings. Motor is rotated by 1.8° per excitation.

Software : As port A is used as an output port, control word for 8255 is 80H.

```
TITLE Stepper Motor Control Program
.MODEL SMALL
.DATA
    PORTA     EQU 00H        ; Port number
    CR        EQU 06H        ; Control Register Address
    Excite_code DB 03H,06H,09H,0CH ; code sequence
                                   ; for clockwise rotation
```

```
.CODE
; Procedure to rotate a stepper motor clockwise by 90°
STEP PROC NEAR
        MOV AX,@DATA          ; [ Initialise
        MOV DS,AX             ;   data segment ]
        MOV AL,80H            ; [ Initialise
        OUT CR,AL             ;   8255 ]
        MOV CL,32H            ; Set repetition count to 50₁₀
```

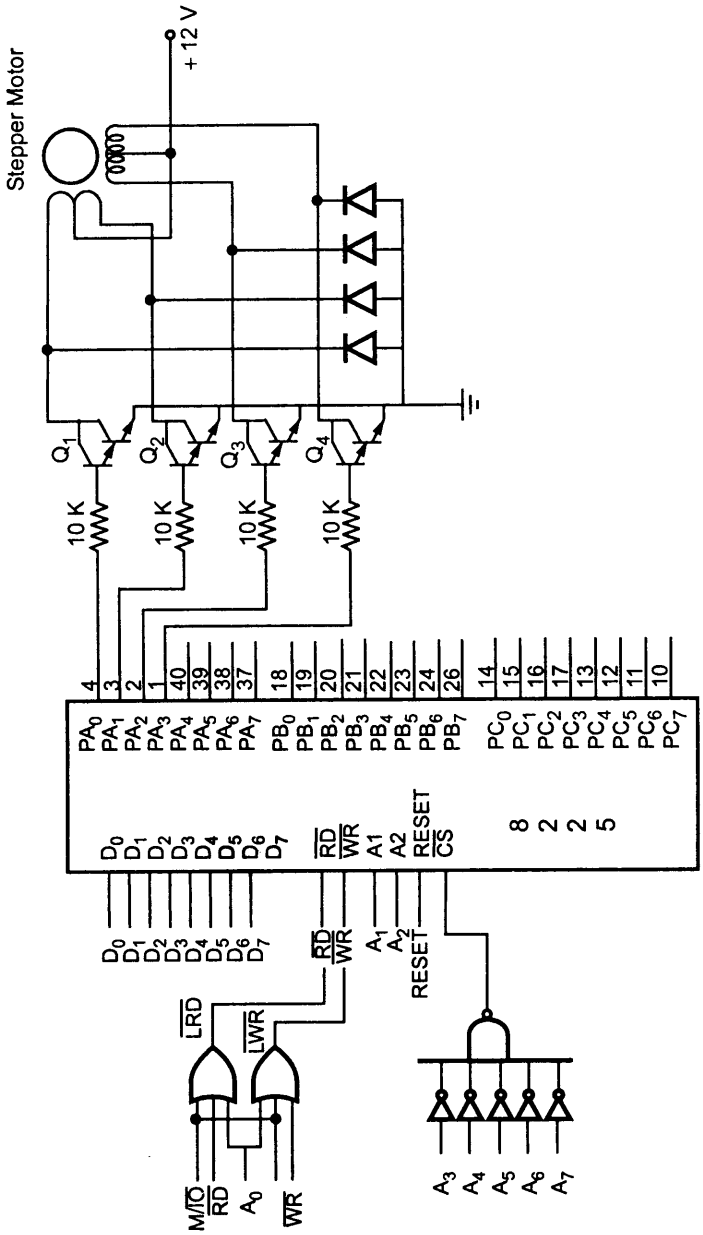


Fig 9.28 Stepper motor interface

```
START:   MOV AH,04H              ; Counts excitation sequence
         LEA BX,Excite_code      ; Initialize pointer
         MOV AL,[BX]
; Point to element in Excite_code
BACK1:   OUT PORTA,AL            ; Send Excite_code code
         CALL DELAY              ; Wait
         INC BX                  ; increment pointer
         DEC AH                  ; Repeat 4 times
         JNZ BACK1
         DEC CL                  ; Repeat 50 times
         JNZ START
         RET
STEP     ENDP
```

9.14 Control of High Power Devices using 8255

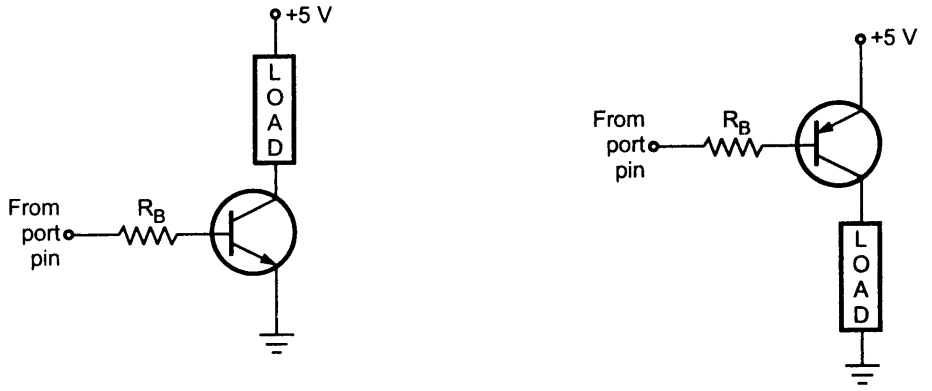
The output port pins of programmable devices can sink current about to 2 mA and source current about few tenth of milliamperes. Thus to interface high power devices such as lights, heaters, solenoids and motors we have to use driver circuit between port pin and the high power device. This section describes various options available to interface high power devices with suitable driver circuits.

9.14.1 Integrated Circuit Buffers

One way to increase current sinking and sourcing capacity is to use integrated circuit buffers such as 7406 hex inverting buffer and 7407 hex noninverting buffer. These buffers are of open collector type, therefore their current sourcing capacity is decided by external register. These buffers can sink current as much as 40 mA. This current is sufficient to interface LED displays.

9.14.2 Transistor Buffers

The Fig. 9.29 shows some buffer circuits using transistors. In these circuits, transistor is used as a switch. It can be switch ON or OFF with logic 0 or logic 1 on port pin depending on the application. To make transistor ON with logic 0 at port pin we have to use pnp transistor otherwise we have to use npn transistor. When transistor is ON, its collector current drives the load. To determine component values and transistor we have to check maximum collector current of the transistor (I_{Cmax}), maximum h_{fe} that transistor can provide (h_{femax}), maximum collector to emitter breakdown voltage (V_{BCEO})and a maximum power dissipation (p_{dmax}) allowed by the transistor.



(a) npn transistor (b) pnp transistor

Fig. 9.29 Transistor buffer circuits

Let us assume that the load current is 200 mA and maximum sourcing current of port pin is 1 mA. Then transistor should have

$$I_{Cmax} > 200 \text{ mA}$$

$$h_{femin} > \frac{I_C}{I_{Bmax}}$$

$$> \frac{200 \text{ mA}}{1 \text{ mA}}$$

$$\therefore \quad h_{femin} > 200$$

$$P_{dmax} > V_{CEsat} \times I_{Cmax}$$

$$> 0.2 \times 200 \text{ mA}$$

$$> 40 \text{ mW}$$

Assuming output high voltage of port pin equal to 4.5 V we have

$$R_B = \frac{4.5 - 0.8}{1 \text{ mA}}$$

$$= 3.7 \text{ k}\Omega$$

When load current requirements are more, one transistor is not sufficient to provide sufficient current gain. In such situations two transistors in darlington connection are used to provide the necessary current gain. The Fig. 9.30 shows the darling transistor pair used to drive relay or solenoid coil.

Here, the output port pin supplies the base current to transistor Q_1. This base current produces β times large collector current of Q_1. The collector current of Q_1 becomes the base current of Q_2 and is amplified by the current gain of Q_2, i.e. β_2. Therefore, overall current amplification is $\beta_1 \times \beta_2$. Usually this amplification is greater than 1000. Many such

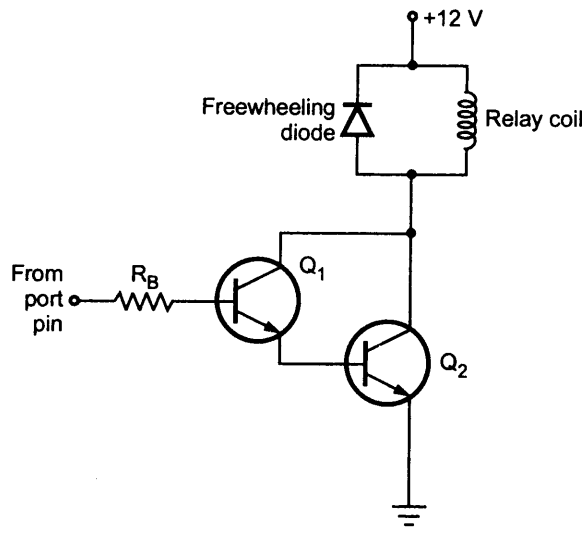


Fig. 9.30 Use of darlington transistor to drive relay or solenoid coil

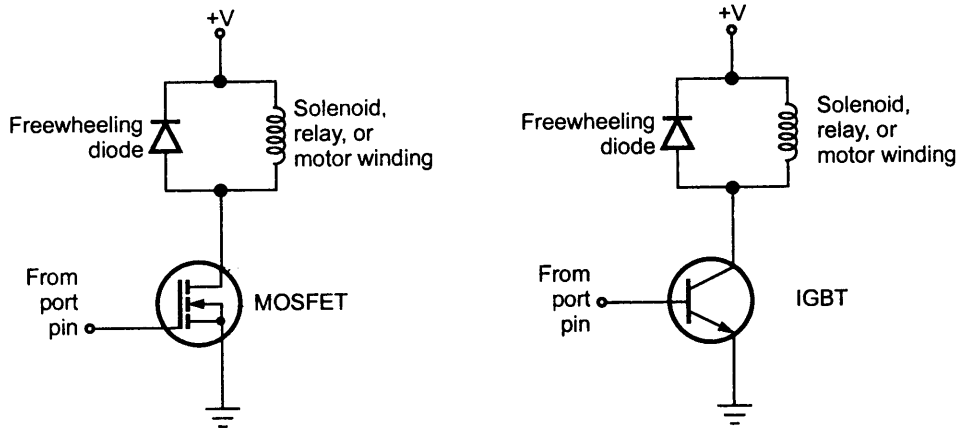
darlington pairs are available. We can use these darlington pair to drive loads upto about 1 A load currents.

As shown in the Fig. 9.30, free wheeling diode is purposely connected when load is inductive such as solenoid, relay and motor. We know that, inductor opposes change in current. When we switch off the transistor, the current in the inductor does not reach to zero immediately; it flows for a while. However, this current cannot flow through transistor, because it is off. Instead, this current develops a voltage across inductor with reverse polarities, i.e. – ve at the top and + ve at in the bottom in given circuit. Usually, this reverse voltage is large enough to break down the transistor. The free wheeling diode provides the path for induced current and clamps the reverse voltage across inductor at 0.7 V.

The Fig. 9.31 shows a power driver circuits using MOSFET and IGBT to drive solenoids, relay or motor windings. Power MOSFETS are somewhat more expensive than darlington transistors, but they have the advantage that they require only a voltage to drive them. The IGBT has high input impedance and fast switching speed. They also have low voltage drop and high current carrying capacity.

9.14.3 Isolation Circuits

To control 220 V or 440 V ac devices it is not desirable to have a common power circuit ground and control circuit ground. This is very important because if the 220 V ac line gets shorted to the V_{CC} line of a microcomputer, it usually bakes most of the microcomputer ICs. To avoid this, the control circuit and power circuit is electrically isolated by separating their individual ground lines.



(a) Power MOSFET circuit (b) IGBT circuit

Fig. 9.31 Circuits for driving solenoid, relay or motor winding

9.14.3.1 Electromagnetic Relays

To control ON/OFF operation of ac devices we can use electromagnetic relay. This relay has both normally open and closed contacts. When a current is passed through the coil of the relay, the switch arm is pulled down, opening the top contact and closing the bottom contact, as shown in the Fig. 9.32.

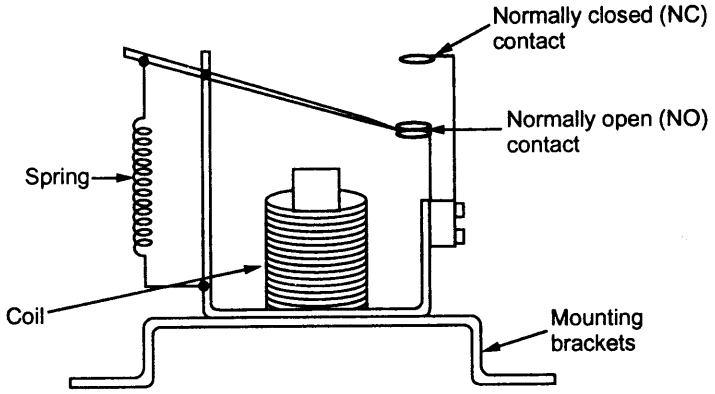


Fig. 9.32 Electromagnetic relay

The relay contacts are rated for maximum current of about 20A - 25A. These relays are also called mechanical relays because mechanical contact makes the circuit to ON or OFF. The mechanical relays having higher current ratings are sometimes called **contactors.**