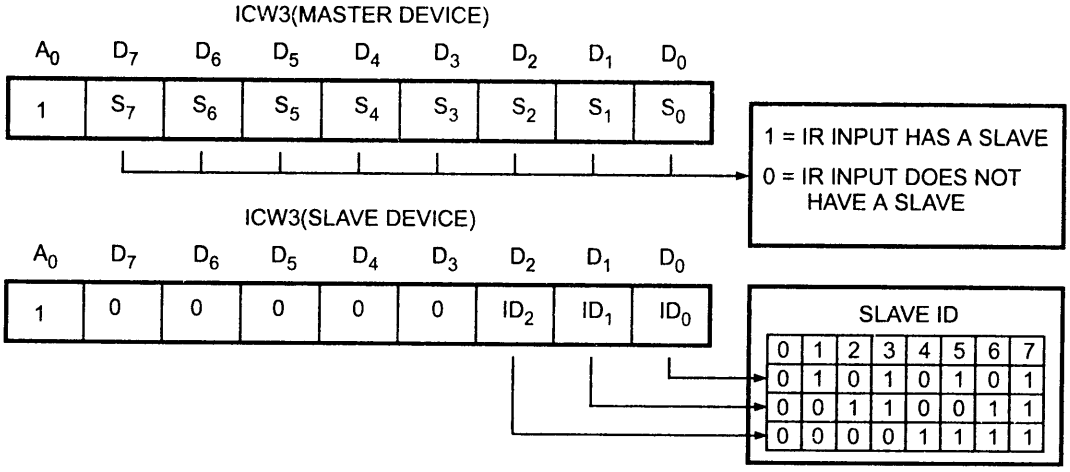


Fig. 6.10 Initialization command word 3 (ICW3)

Initialization Command Word 4 (ICW4)

It is loaded only if the D_0 bit of ICW1 (IC 4) is set. The format of ICW4 is shown in Fig. 6.11.

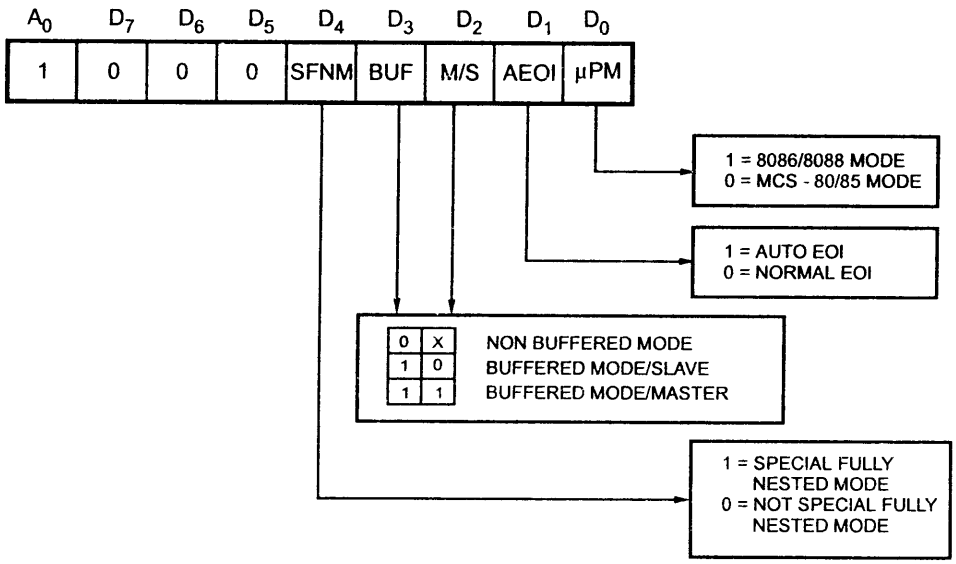


Fig. 6.11 Initialization command word 4 (ICW4)

It specifies.

1) Whether to use special fully nested mode or non special fully nested mode.
2) Whether to use buffered mode or non buffered mode.
3) Whether to use Automatic EOI or Normal EOI
4) CPU used, 8086/8088 or 80810.

After initialization, the 8259 is ready to process interrupt requests. However, during operation, it might be necessary to change the mode of processing the interrupts. Operation Command Words (OCWs) are used for this purpose. They may be loaded anytime after the 8259's initialization to dynamically alter the priority modes.

Operation Command Word 1 (OCW1)

A Write command to the 8259 with A_0 = 1 (after ICW2) is interpreted as OCW1. OCW1 is used for enabling or disabling the recognition of specific interrupt requests by programming the IMR.

M = 1 indicates that the interrupt is to be masked, and M = 0 indicates that it is to be unmasked as shown in Fig. 6.12.

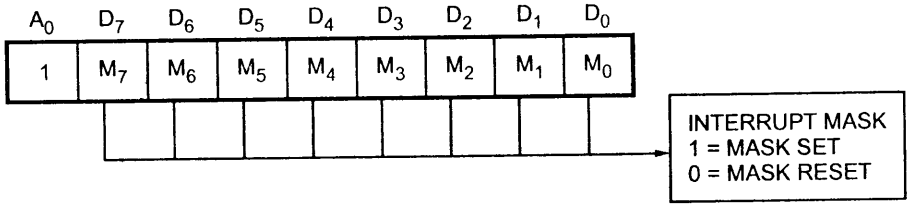


Fig. 6.12 Operation command word 1 (OCW1)

Operation Command Word 2 (OCW2)

A Write command with A_0 = 1 and D_4 D_3 = 00 is interpreted as OCW2. The R(Rotate), SL (Select-Level), EOI bits control the Rotate and End Of Interrupt Modes and combinations of the two. Fig. 6.13 shows the Operation Command Word format. L_2 -L_0 are used to specify the interrupt level to be acted upon when the SL bit is active.

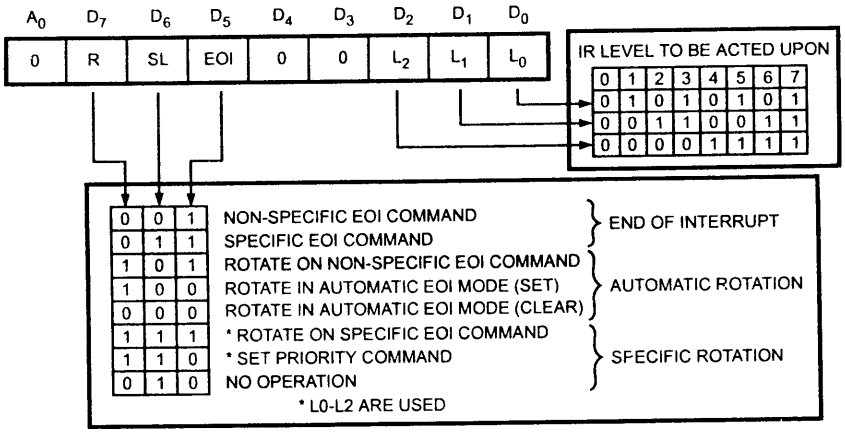


Fig. 6.13 Operation command word 2 (OCW2)

Operation Command Word 3 (OCW3)

OCW3 is used to read the status of the registers, and to set or reset the Special Mask and Polled modes. Fig. 6.14 shows format operational command word 3.

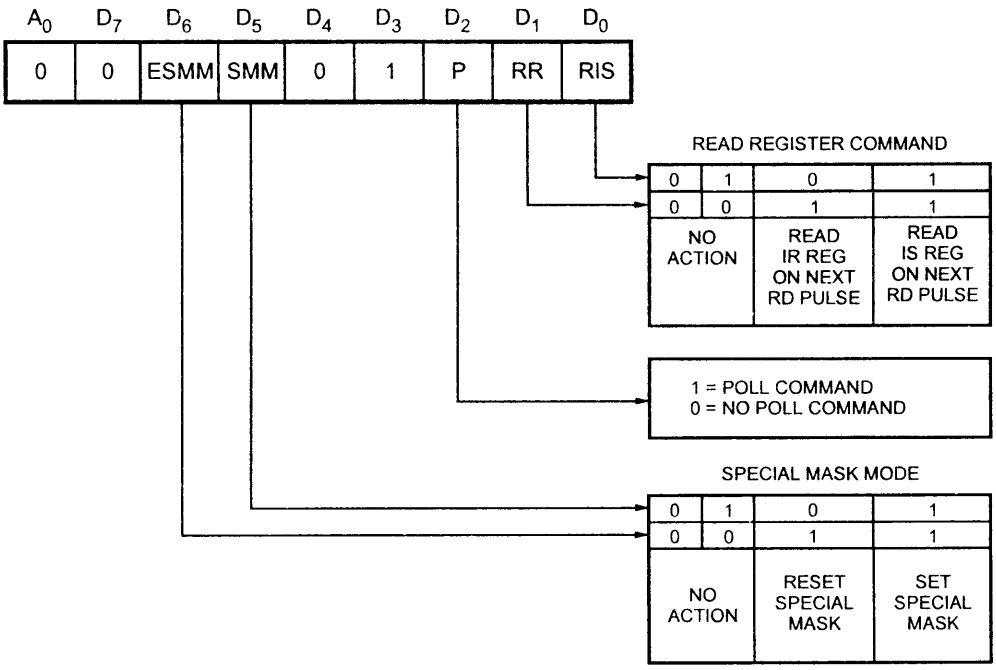


Fig. 6.14 Operation command word 3 (OCW3)

8259 Status Read Operations

The status of the Interrupt Request Register, the In-Service Register, and the Interrupt Mask Register of the 8259 may be read by issuing appropriate Read commands as described below.

IRR Status Read

An OCW3 with RR (Read Register) = 1 and RIS (Read ISR) = 0 set up the 8259 for a status read of the Interrupt Request Register.

When the 8259 is not in the Polled mode, after it is sets up for an IRR status read operation, all Read commands with A_0 = 1 cause the 8259 to send the IRR status word.

ISR Status Read

An OCW3 with RR = 1 and RIS = 1 sets up the 8259 for a status read of the In-Service Register. A subsequent read command issued to the 8259 will cause the 8259 to send the contents of the ISR onto the data bus.

IMR Status Read

A Read command issued to the 8259 with A_0 = 1 (with $\overline{RD}$, $\overline{CS}$ = 0) causes the 8259 to put out the contents of the Interrupt Mask Register. OCW3 is not required for a status read of the IMR.

As described earlier, the sequence shown in flowchart (Fig. 6.4) must be followed to initialize 8259A. According to this flow chart an ICW1 and an ICW2 must be sent to any 8259A in the system. If the system has any slave 8259As (cascade mode) then an ICW3 must be sent to the master, and a difference ICW3 must be sent to the slave. If the system is an 8086, or if you want to specify certain special conditions, then you have to send an ICW4 to the master and to each slave. To have better understanding the initiation sequences for different specification are given in the next section.

Note : It is assumed that A_1 of the system bus is connected to the A_0 of the 8259A. So the internal addresses correspond to 0 and 2. It is also assumed that the base address of the device is 40H. So the two system addresses for the 8259A are 40H and 42H.

Example 6.1 : *Write the initialization instructions for 8259A interrupt controller to meet the following specifications :*
a) Interrupt type 32. b) Edge triggered, single and ICW4 needed.
c) Mask interrupts IR1 and IR3.

Solution :

ICW1

A_7	A_6	A_5	1	LTIM	ADI	SNGL	IC4
0	0	0	1	0	0	1	1

= 13H

Note : When used with an 8086, bit D_2, D_5, D_6 and D_7 are do not care, so we make them 0's for simplicity.

ICW2

In an 8086 system ICW2 is used to tell the 8259A the type number to send in response to an interrupt signal on the IR0 input.

B_7	B_6	B_5	B_4	B_3	B_2	B_1	B_0
0	0	1	0	0	0	0	0

= 20H = 32 Decimal

ICW2 for sending interrupt type 32 to the 8086 in response to an IR0 interrupt is 20H.

Note : For an IR1 input the 8259A will send 00100001 binary (33 decimal) and so on for the other IR inputs.

ICW3

Since we are not using a slave in our example, we do not need to send an ICW3.

ICW4

For our example, the only reason we need to send an ICW4 is to let the 8259A know that it is operating in an 8086 system. We do this by making bit D_0 of the ICW4 one.

OCW1

An OCW1 must be sent to an 8259A to unmask any IR inputs. For our example we want to mask IR1 and IR3, so we put 1's in these two-bits and 0's in the rest of the bits.

M_7	M_6	M_5	M_4	M_3	M_2	M_1	M_0	
0	0	0	0	1	0	1	0	= 0AH

Program :

```
MOV AL,13H    ; edge triggered, single, ICW4 needed
OUT 40H,AL    ; Send ICW1
MOV AL,20H    ; type 32 is first 8359A type
OUT 41H,AL    ; send ICW2
MOV AL,01H    ; ICW4, 8086 mode.
OUT 41H,AL    ; send ICW4
MOV AL,0AH    ; OCW1 to mask IR1 and IR3
OUT 41H,AL    ; send OCW1
```

Example 6.2 : *Write the initialization instructions for master and slave configuration to meet the following specifications :*

1) The INTR of slave is routed through IR2 of the master 8259A to the 8086.

2) Master and slave are both level triggered.

3) First interrupt types for master and slave are 32 and 64 respectively.

4) Modes : automatic rotation and auto end of interrupt.

5) Addresses of the master are 40H and 41H and the slave are 80H and 81H.

6) Buffers are not used.

Initialization command words for Master ICW1 (Master).

ICW1 (master)

A_7	A_6	A_5	1	LTIM	ADI	SNGL	IC4	
0	0	0	1	1	0	0	1	= 10H

ICW2 (master)

B_7	B_6	B_5	B_4	B_3	B_2	B_1	B_0	
0	0	1	0	0	0	0	0	= 20H

ICW3 (master)

S_7	S_6	S_5	S_4	S_3	S_2	S_1	S_0	
0	0	0	0	0	1	0	0	= 04H

ICW4 (master)

0	0	0	SFNM	BUF	M/S	AEOI	µPM
0	0	0	0	0	0	1	1

= 03H

Program :

```
MOV AL, 19H  ; level triggered, cascaded, ICW4 needed
OUT 40H, AL  ; send ICW1 (master)
MOV AL, 20H  ; type 32 is first 8259A type
OUT 41H, AL  ; send ICW2 (master)
MOV AL, 04H  ; slave at IR2
OUT 42H, AL  ; send ICW3 (master)
MOV AL, 03H  ; ICW4, 8086 mode, and set AEOI
OUT 41H, AL  ; send ICW4 (master)
MOV AL, 19H  ; level triggered, cascaded, ICW4 needed
OUT 80H, AL  ; send ICW1 (slave)
MOV AL, 40H  ; type 64 is first 8259A type
OUT 81H, AL  ; send ICW3 (slave)
MOV AL, 02H  ; ID for slave connected to IR2
OUT 81H, AL  ; send ICW2 (slave)
MOV AL, 01H  ; ICW4, 8086 mode
OUT 81H, AL  ; send ICW4
MOV AL, 80H  ; OCW2 (rotate in auto EOI mode set command)
OUT 80H, AL  ; send OCW2 (slave)
```

6.5.6 8259A Interfacing

Fig. 6.15 shows that how an 8259A can be interfaced with the 8086 microprocessor system in minimum mode. In case of 8088 microprocessor same interfacing diagram can be used except M/$\overline{IO}$ signal. In 8088, M/$\overline{IO}$ signal is represented by IO/$\overline{M}$ signal, therefore this signal is connected to G (active high) signal of decoder to interface 8259A in I/O mapped I/O mode.

Addressing of 8259A :

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
1	1	1	1	1	1	1	1	1	1	1	1	0	0	X	0	FFF0H
F				F				F				0/2				FFF2H

The 74LS138 address decoder will assert the $\overline{CS}$ input of the 8259A when an I/O base address is FFF0H or FFF2H on the address bus. The A_0 input of the 8259A is used to select one of the two internal addresses in the device. A_0 of the 8259A is connected to system line A1. So the system addresses for the two internal addresses are FFF0H and FFF2H. The data lines of an 8259A are connected to the lower half of the system data bus, because the 8086 expects to receive interrupt types on these lower eight data lines. $\overline{RD}$ and $\overline{WR}$ signals are connected to the system $\overline{RD}$ and $\overline{WR}$ lines. The interrupt request signal

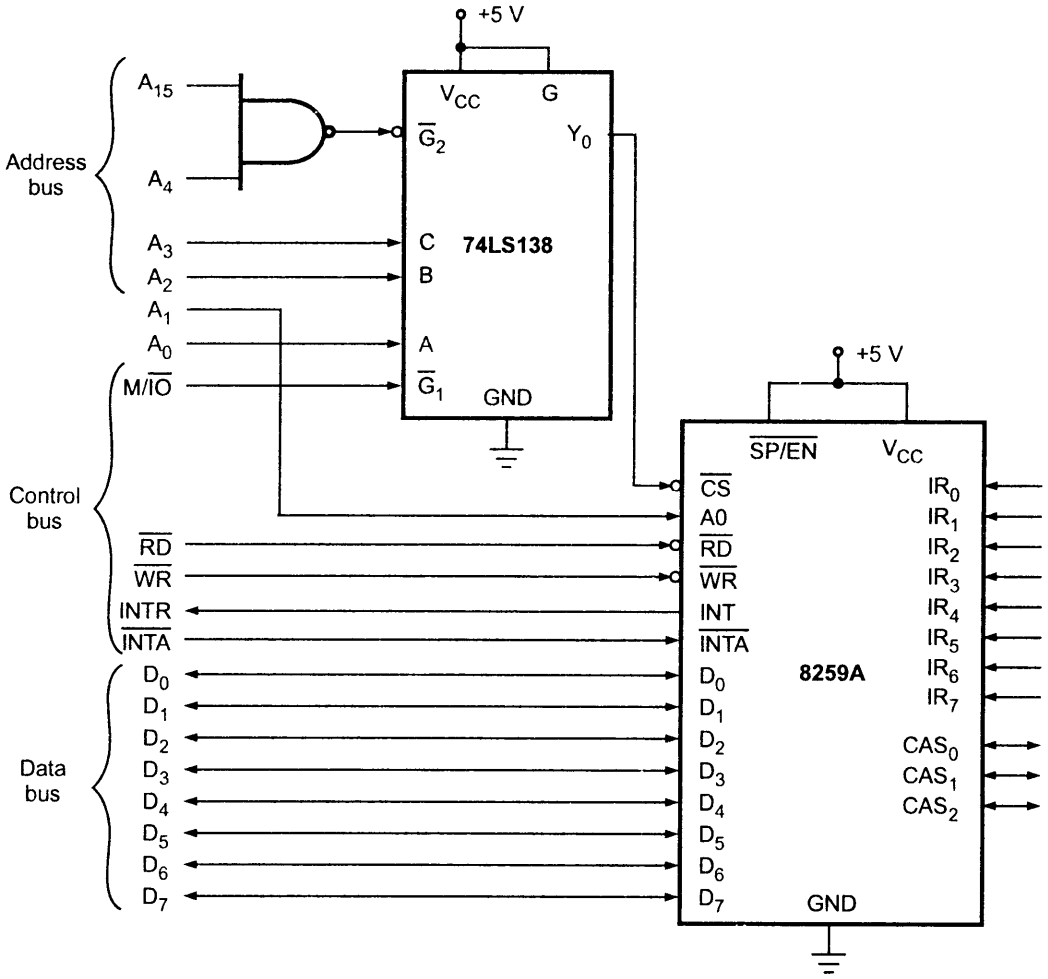


Fig. 6.15 8259A interface to 8086 system bus

INT from the 8259A is connected to the INTR input of the 8086 and $\overline{INTA}$ from the 8086 is connected to $\overline{INTA}$ on the 8259A. As we are using single 8259A in the system $\overline{SP}/\overline{EN}$ pin is tied high and CAS_0-CAS_2 lines are left open. The eight IR inputs are available for interrupt signals.

Note :

1. Unused IR inputs should be tied to ground so that a noise pulse cannot accidentally cause an interrupt.
2. In maximum mode RD and INTA signals of 8259A are connected to the IORC, IOWC and INTA lines of 8288 bus controller.

Cascading :

The 8259A can be easily interconnected to get multiple interrupts. Fig. 6.16 shows how 8259A can be connected in the cascade mode. In cascade mode one 8259A is configured in Master mode and other should be configured in the Slave mode. In this figure 8259A-1 is in the master mode and others are in slave mode. Each slave 8259A is identified by the number which is assigned as a part of its initialization. Since the 8086 has only one INTR input, only one of the 8259A INT pins is connected to the 8086 INTR pin. The 8259A connected directly into the 8086 INTR pin is referred as the master. The INT pins from other 8259A are connected to the IR inputs of the master 8259A. These cascaded 8259As are referred as slave. The INTA signal is connected to both master and slave 8259A.

(See Fig. 6.16 on next page.)

The cascade pins CAS_0 to CAS_2 are connected from the master to the corresponding pins of the slave. For the master these pins function as outputs, and for the slave these pins function as inputs. The $\overline{SP}/\overline{EN}$ signal is tied high for the master. However it is grounded for the slave.

Each 8259A has its own addresses so that command words can be written to it and status bytes read from it.

Addresses for 8259As :

No	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
8259A-1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	X	0	FFF0H
	F				F				F				0/2				FFF2H
8259A-2	1	1	1	1	1	1	1	1	1	1	1	1	0	1	X	0	FFF4H
	F				F				F				4/6				FFF6H
8259A-3	1	1	1	1	1	1	1	1	1	1	1	1	1	0	X	0	FFF8H
	F				F				F				8/A				FFFAH

Master and slave operation :

When the slave receives an interrupt signal on one of its IR inputs, it checks mask condition and priority of the interrupt request. If the interrupt is unmasked and its priority is higher than any other interrupt level being serviced in the slave, then the slave will send an INT signal to the IR input of a master. If that IR input of the master is unmasked and if that input is a higher priority than any other IR inputs currently being serviced, then the master will send an INT signal to the 8086 INTR input. If the INTR interrupt is enabled, the 8086 will go through its INTR interrupt procedure and sends out two $\overline{INTA}$ pulses to both the master and the slave. The slave ignores the first interrupt acknowledge pulse but the master outputs a 3-bit slave identification number on the CAS_0-CAS_2 lines. Sending the 3-bit ID number enables the slave. When the slave receives the second $\overline{INTA}$ pulse from the 8086, the slave will send the desired type number to the 8086 on the eight data lines.

If an interrupt signal is applied directly to one of the IR inputs of the master, the master will send the desired interrupt type to the 8086 when it receives the second $\overline{INTA}$ pulse from the 8086.

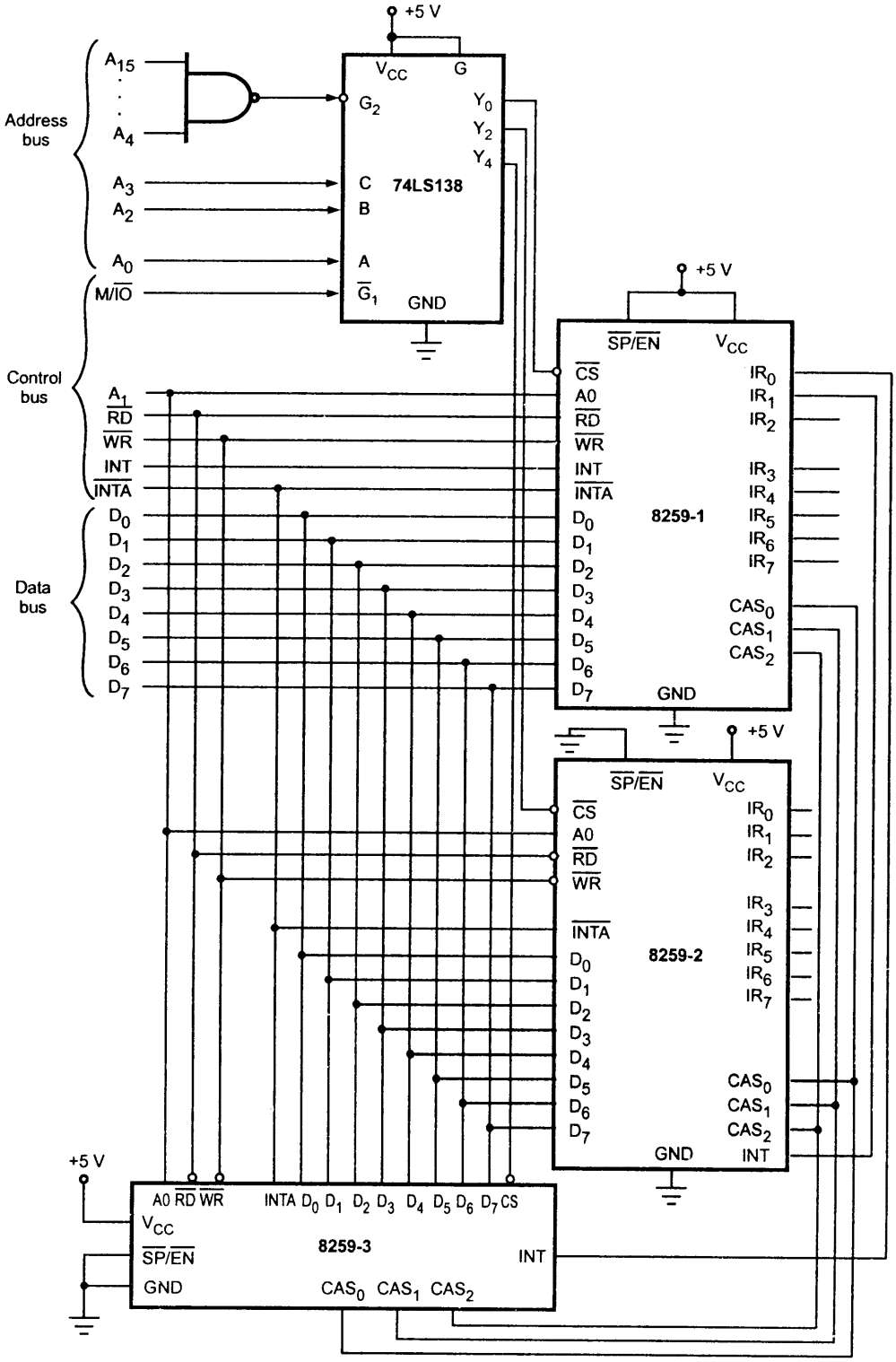


Fig. 6.16 Three 8259 in the cascade mode

6.6 Interrupt Example

There are several reasons for writing interrupt service routine. However, to write such a interrupt service routine we have set the address of our interrupt service routine in the interrupt vector table. To set an interrupt vector to a specified address, (starting address of interrupt service routine) there are two ways :

1. Using function 25H of INT21H.
2. Without using any DOS function.

1. INT21H, Function 25H : Set Interrupt Address

To set a new interrupt address, load the required interrupt number in the AL and the new address in the DX :

```
MOV AH, 25H       ; Request interrupt address
MOV AL, int #     ; Interrupt number
LEA DX, newaddr   ; New address for interrupt
INT 21H
```

The above program replaces the present address of the interrupt with the new address. In effect, then, when the specified interrupt occurs, processing links to resident program, rather than to the normal interrupt address.

2. Without using any DOS Function

The DOS function discussed above do nothing more than getting address of interrupt vector corresponding to an interrupt number and loading two words (segment address and offset address of the interrupt service routine) into it. The address of interrupt vector can be obtained by multiplying the interrupt number by 4. Once we get the address of the Interrupt vector table we have load the segment address and offset address of the interrupt service routine.

Example 6.3 : *Generate a real time clock by generating a periodic interrupt request signal on the $\overline{NMI}$ input of 8086.*

Solution : Hardware : The Fig. 6.17 shows simple circuit that generates interrupt request after every 0.5 sec.

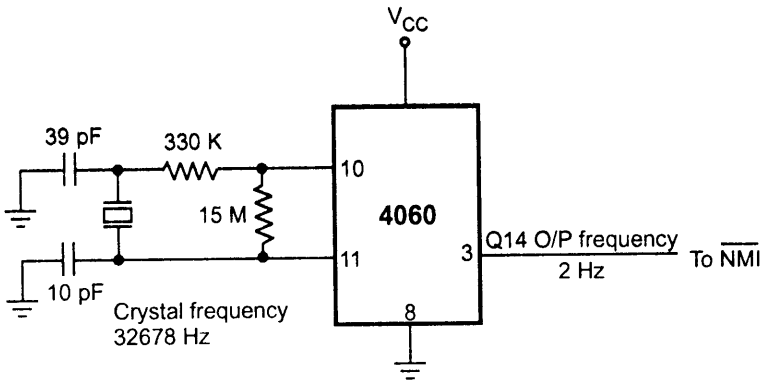


Fig. 6.17 Interrupt generation circuit

Software :

```
; Main Program
.MODEL SMALL
.DATA
     SEC DB ? ; Store seconds
     MIN DB ? ; Store minutes
     HR  DB ?; Store hours
.CODE
START:   MOV AX,@DATA
         MOV DS,AX     ; Initialize data segment
         MOV SEC,00H
         MOV MIN,00H
         MOV HR,00H    ; set present time
         CALL DISPLAY ; Displays the SEC, MIN and HR
         MOV AH,00H    ; set counter
; Get the address of int type 2 (NMI)
         SUB AX,AX     ; clear AX
         MOV ES,AX     ; set ES to bottom of memory 0000H
         MOV DI,2      ; put the interrupt number in DI
         SHL DI,1      ; multiply by 2
         SHL DI,1      ; multiply by 2 again
; Now load the address of routine in the vector table.
     MOV ES:[DI],OFFSET TIMES
     MOV ES:[DI]+2,SEG TIMES
HERE:    JMP HERE
TIMES:   PROC NEAR
         PUSH SI
         INC AH
         CMP AH,02
         JNZ DONE      ; check count = 2
         MOV AH,00H    ; reset counter
         MOV AL,SEC
         INC AL
         DAA           ; BCD adjust SECOND COUNT
         MOV SEC,AL
         CMP AL,60H    ; check if sec = 60
         JNZ DONE
         MOV SEC,00H   ; Reset SEC = 00
         MOV AL,MIN
         INC AL
         DAA
         MOV MIN,AL    ; BCD Adjust Minute Count
         CMP AL,60H    ; check if MINUTE = 60
         JNZ DONE
         MOV MIN,00H   ; Reset MIN = 00
         MOV AL,HR
         INC AL
         DAA
         MOV HR,AL     ; BCD Adjust HR COUNT
         CMP AL,24H    ; check if HR = 24
```

```
        JNZ DONE
        MOV HR,00H    ; Reset HR = 00
DONE:   POP SI        ; restore registers
        MOV AH,00
        IRET
        TIMES ENDP
        END START
        END
```

Review Questions

1. *What do you mean by interrupt ?*
2. *What is interrupt service routine ?*
3. *What are the sources of interrupts in 8086 ?*
4. *What is interrupt vector table ?*
5. *Draw and explain the IVT for 8086.*
6. *Briefly describe the conditions which cause the 8086 to perform each of the following types of interrupts : Type 0, Type 1, Type 2, Type 3 and Type 4.*
7. *Explain interrupt structure of 8086.*
8. *What are software interrupt ? How 8086 responds to software interrupts ?*
9. *Draw and explain the interrupt acknowledge cycle of 8086.*
10. *Describe the response of 8086 to the interrupt coming on pin.*
11. *What do you mean by interrupt priorities ?*
12. *State the interrupt priorities for 8086 interrupts.*
13. *What are advantages of using 8259 ?*
14. *List the features of 8259.*
15. *Explain the operating modes of 8259.*
16. *Draw and explain the interfacing of 8259 with 8086.*
17. *Draw and explain the interfacing of cascaded 8259s with 8086.*
18. *Explain the procedure of interrupt programming.*

❑❑❑

7 8086/8088 Configurations

7.1 Introduction

Unlike 8085, 8086 and 8088 can be operated in two modes : minimum mode and maximum mode. In this chapter we study the topics related to minimum mode and maximum mode operation of 8086. Topics include clock generation, bus buffering, bus latching, timings, minimum mode operation and maximum mode operation. Let us begin with signal description of 8086.

7.2 Signal Description of 8086

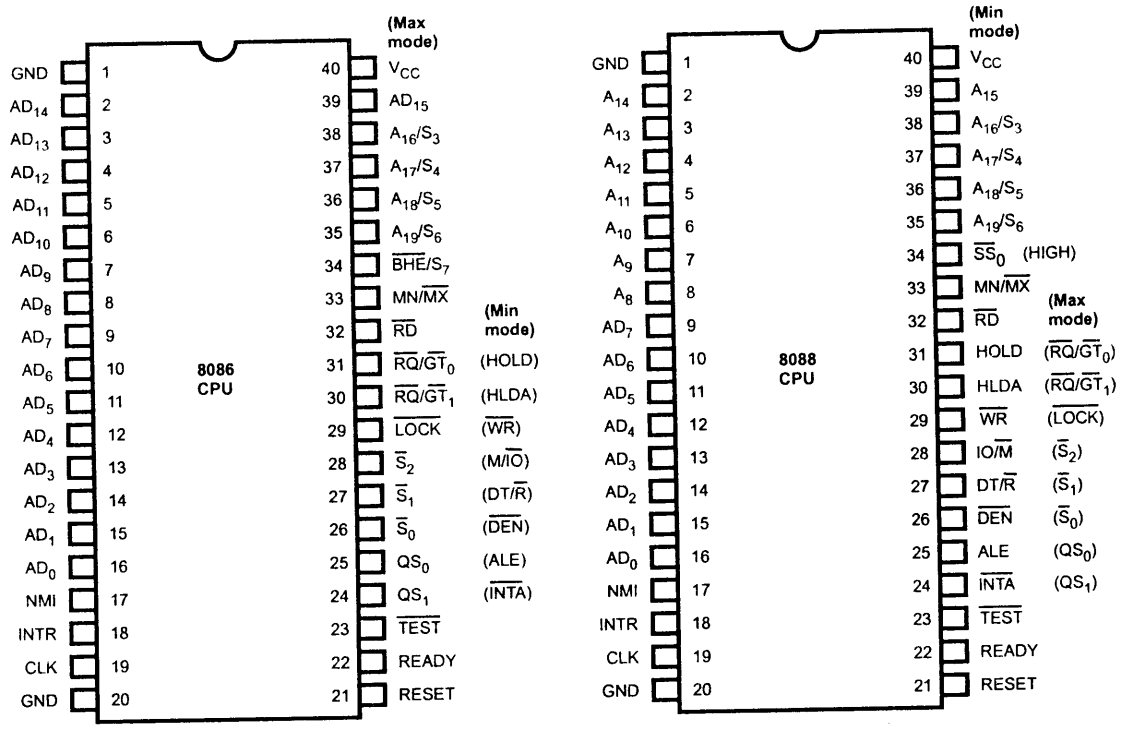
In order to implement many situations in the microcomputer system the 8086 and 8088 has been designed to work in two operating modes :

1. Minimum mode
2. Maximum mode.

The minimum mode is used for a small systems with a single processor and maximum mode is for medium size to large systems, which often include two or more processors. Fig. 7.1 shows the pin diagram of 8086 and 8088 in minimum as well as maximum mode. As a close comparison reveals, there is no much difference between two microprocessors - both are packaged in 40-pin dual-in-line package (DIPs). As mentioned in section 1.1, the 8086 is a 16-bit microprocessor with a 16-bit data bus, and the 8088 is a 16-bit microprocessor with an 8-bit data bus. The pin-out shows, the 8086 has pin connections AD_0-AD_{15}, and the 8088 has pin connections AD_0-AD_7. There is one more minor difference in one of the control signals. The 8086 has an $M/\overline{IO}$ pin, and the 8088 has an $IO/\overline{M}$ pin. The only hardware difference appears on pin 34 of both chips : on the 8086 it is a $\overline{BHE}/S_7$ pin, while on the 8088 it is a $\overline{SS}_0$ pin.

The 8086 signals can be categorised in three groups.

- Signals having common functions in both minimum and maximum modes.
- Signals having special functions for minimum mode.
- Signals having special functions for maximum mode.



(a) Pin diagram of 8086

(b) Pin diagram of 8088

Fig. 7.1

7.2.1 Signals with Common Functions in both Modes

1. AD_{15}-AD_0 : Acts as address bus during the first part of machine cycle and data bus for the remaining part of the machine cycle.
2. A_{19}/S_6-A_{16}/S_3 : During the first part of machine cycle these are used to output upper 4-bits of address. During remaining part of the machine cycle these are used to output status, which indicates the type of operation to be performed in that cycle. S_3 and S_4 indicate the segment register being used as follows :

S_4	S_3	Register
0	0	ES
0	1	SS
1	0	CS or none
1	1	DS

S_5 gives the current setting of the interrupt flag (IF) and S_6 is always zero.

3. $\overline{BHE}/S_7$: BHE (Bus High Enable) : Low on this pin during first part of the machine cycle, indicates that at least one byte of the current transfer is to be made

on higher order byte AD_{15}-AD_8; otherwise the transfer is made on lower order byte AD_7-AD_0.

$\overline{BHE}$	A_0	Data accesses
0	0	Word
0	1	Upper byte from odd address
1	0	Lower byte from even address
1	1	None

Status S_7 is output during the later part of the machine cycle, but, presently, S_7 has not been assigned a meaning.

4. **NMI** : It is a positive edge triggered nonmaskable interrupt request.
5. **INTR** : It is a level triggered maskable interrupt request. It is sampled during the last clock cycle of each instruction to determine if the processor should enter into an interrupt service routine.
6. **CLK** : 8086 requires clock signal (with 33 % duty cycle) from some external, crystal controlled generator to synchronize internal operations. Clock frequency depends on the version of 8086.

Processor	Required clock signal
8086	5 MHz
8086-2	8 MHz
8086-1	10 MHz

7. **RESET** : It clears PSW, IP, DS, SS, ES, and the instruction queue. It then sets CS to FFFFH. This signal must be high for at least 4 clock cycles. When RESET is removed, 8086 will fetch its next instruction from physical address FFFF0H.
8. **READY** : If this signal is low the 8086 enters into wait state. This signal is used primarily to synchronize slower peripherals with the microprocessor.
9. **$\overline{TEST}$ (Input)** : This signal is only used by the WAIT instruction. The 8086 enters into a wait state after execution of the WAIT instruction until a LOW signal on the $\overline{TEST}$ pin. $\overline{TEST}$ signal is synchronized internally during each clock cycle on the leading edge of the clock cycle.
10. **$\overline{RD}$ (Output)** : $\overline{RD}$ is low whenever the 8086 is reading data from memory or an I/O device.
11. **MN/$\overline{MX}$ (Input)** : The 8086 can be configured in either minimum mode or maximum mode using this pin. This pin is tied high for minimum mode.

7.2.2 Signal Definitions (24 to 31) for Minimum Mode

$\overline{\text{INTA}}$ (Interrupt Acknowledge) Output : This indicates recognition of an interrupt request. It consists of two negative going pulses in two consecutive bus cycles. The first pulse informs the interface that its request has been recognized and upon receipt of the second pulse, the interface is to send the interrupt type to the processor over the data bus.

ALE (Address Latch Enable) Output : This signal is provided by 8086 to demultiplex the AD_0-AD_{15} into A_0-A_{15} and D_0-D_{15} using external latches.

$\overline{\text{DEN}}$ (Data Enable) Output : This signal informs the transceivers that the CPU is ready to send or receive data.

DT/$\overline{\text{R}}$ (Data Transmit/Receive) Output : This signal is used to control data flow direction. High on this pin indicates that the 8086 is transmitting the data and low indicates that the 8086 is receiving the data.

M/$\overline{\text{IO}}$ Output : It is used to distinguish memory data transfer, (M/$\overline{\text{IO}}$ = HIGH) and I/O data transfer (M/$\overline{\text{IO}}$ = LOW).

$\overline{\text{WR}}$: Write Output : $\overline{\text{WR}}$ is low whenever the 8086 is writing data into memory or an I/O device.

HOLD Input, HLDA Output : A HIGH on HOLD pin indicates that another master (DMA) is requesting to take over the system bus. On receiving HOLD signal processor outputs HLDA signal HIGH as an acknowledgment. At the same time, processor tristates the system bus. A low on HOLD gives the system bus control back to the processor. Processor then outputs low signal on HLDA.

7.2.3 Signal Definitions (24 to 31) for Maximum Mode

1. **QS_1, QS_0 (output) :** These two output signals reflect the status of the instruction queue. This status indicates the activity in the queue during the previous clock cycle.

QS_1	QS_0	Status
0	0	No operation (queue is idle)
0	1	First byte of an opcode
1	0	Queue is empty
1	1	Subsequent byte of an opcode

2. **$\overline{S}_2$, $\overline{S}_1$, $\overline{S}_0$ (output) :** These three status signals indicate the type of transfer to be take place during the current bus cycle.

$\overline{S_2}$	$\overline{S_1}$	$\overline{S_0}$	Machine cycle
0	0	0	Interrupt Acknowledge
0	0	1	I/O Read
0	1	0	I/O Write
0	1	1	Halt

$\overline{S_2}$	$\overline{S_1}$	$\overline{S_0}$	Machine cycle
1	0	0	Instruction fetch
1	0	1	Memory read
1	1	0	Memory write
1	1	1	Inactive-Passive

3. $\overline{LOCK}$: This signal indicates that an instruction with a LOCK prefix is being executed and the bus is not to be used by another processor.
4. $\overline{RQ}/\overline{GT_1}$ and $\overline{RQ}/\overline{GT_0}$: In the maximum mode, HOLD and HLDA pins are replaced by $\overline{RQ}$ (Bus request)/$\overline{GT_0}$ (Bus Grant), and $\overline{RQ}/\overline{GT_1}$ signals. By using bus request signal another master can request for the system bus and processor communicate that the request is granted to the requesting master by using bus grant signal. Both signals are similar except the $\overline{RQ}/\overline{GT_0}$ has higher priority than $\overline{RQ}/\overline{GT_1}$.

7.3 Physical Memory Organisation

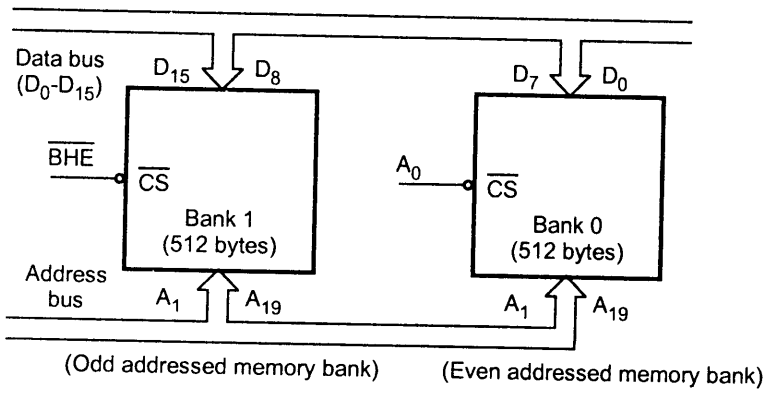


Fig. 7.2 Memory interfacing

Most of the memory ICs are byte oriented i.e. each memory location can store only one byte of data. The 8086 is a 16-bit microprocessor, it can transfer 16-bit data. So in addition to byte, word (16-bit) has to be stored in the memory. This is stored by using two consecutive memory locations, one for least significant byte and other for most significant byte. The address of word is the address of least significant byte. To implement this, the entire memory is divided into two memory banks : bank0 and bank1. Fig. 7.2 shows the interfacing diagram to these memory banks. Bank0 is selected only when A_0 is zero and Bank1 is selected only when $\overline{BHE}$ is zero. A_0 is zero for all even addresses. So Bank0 is usually referred as **even addressed memory** bank. $\overline{BHE}$ is used to access higher order memory bank, referred to as **odd addressed memory** bank.

Together $\overline{BHE}$ and A_0 tell the interface how the data appears on bus. Four possible combinations are shown in the table.

Sr. No.	Operation	$\overline{BHE}$	A_0	Data Lines Used
1.	Read/Write a byte at an even address	1	0	D_7 - D_0
2.	Read/Write a byte at an odd address	0	1	D_{15} - D_8
3.	Read/Write a word at an even address	0	0	D_{15} - D_0
4.	Read/Write a word at an odd address	0	1	D_{15}-D_0 in first operation byte from odd bank is transferred.
		1	0	D_7-D_0 in second operation byte from even bank is transferred.

Note : To access odd addressed word two bus cycles are required.

Every microprocessor based system has a memory system. Almost all systems contain two basic types of memory, read-only memory (ROM) and random access memory (RAM) or read/write memory. Read only memory contains system software and permanent system data such as lookup tables, while Random Access Memory contains temporary data and application software. ROMs/PROMs/EPROMs are mapped to cover the CPU's reset address, since these are non-volatile. When the 8086 is reset, the next instruction is fetched from memory location FFFF0H. So in the 8086 systems, the location FFFF0H must be ROM location.

The Fig. 7.3 shows memory map for 8086. Certain locations in 1 Mbyte memory are reserved and some are dedicated for specific CPU operations. Locations from FFFF0H to

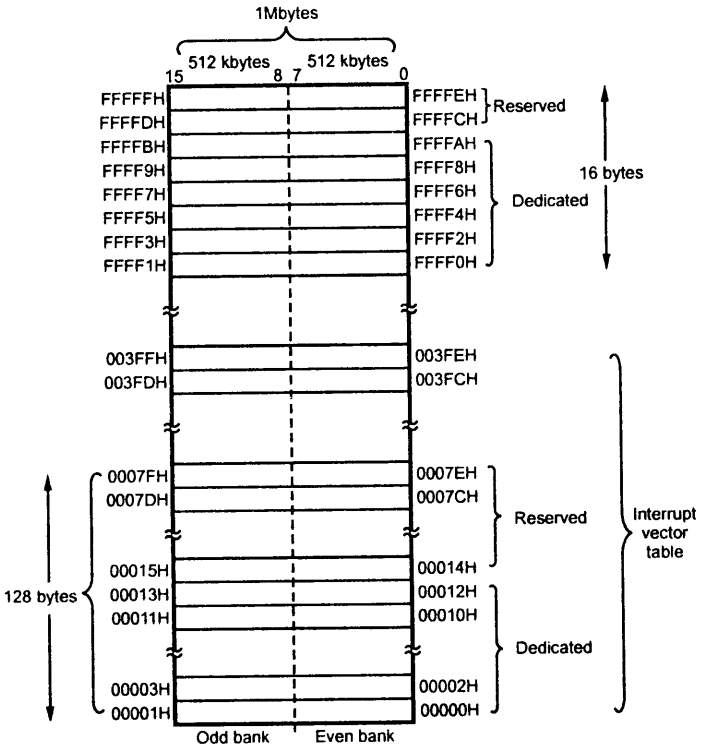


Fig. 7.3 Memory map for 8086

FFFF5H are dedicated to the initialization procedure of the 8086, while locations FFFF6H to FFFFBH are dedicated to the initialization procedure of the 8089 input/output processor. Locations 00000H to 00013H are dedicated to store the vector addresses of the dedicated interrupts. The dedicated locations are used for processing of specific system initialization, interrupt and reset function.

Intel has also reserved several locations for future hardware and software products. Locations from 00014H to 0007FH and locations from FFFFCH to FFFFFH are reserved locations. The locations from 00000H to 003FFH are used for interrupt vector table (IVT). The interrupt vector table provides the starting location/address of the interrupt service routine for the interrupt supported by 8086. The detail description of interrupt vector table is given in sections 6.2 and 6.3.

7.4 I/O Addressing Capability

The 8086 can generate 16-bit of I/O address. Thus it can address upto 64 kbyte I/O locations or 32 K word I/O locations. The 16-bit I/O address appears on A_0 to A_{15} address lines; A_{16} to A_{19} lines are at logic 0 during the I/O operations. The 16-bit DX register is used as 16-bit I/O address pointer to address upto 64 K devices in indirect addressing mode. The I/O instructions with direct addressing mode can directly address one or two of the 256 I/O byte locations in page 0 of the I/O address space. See Fig. 7.4.

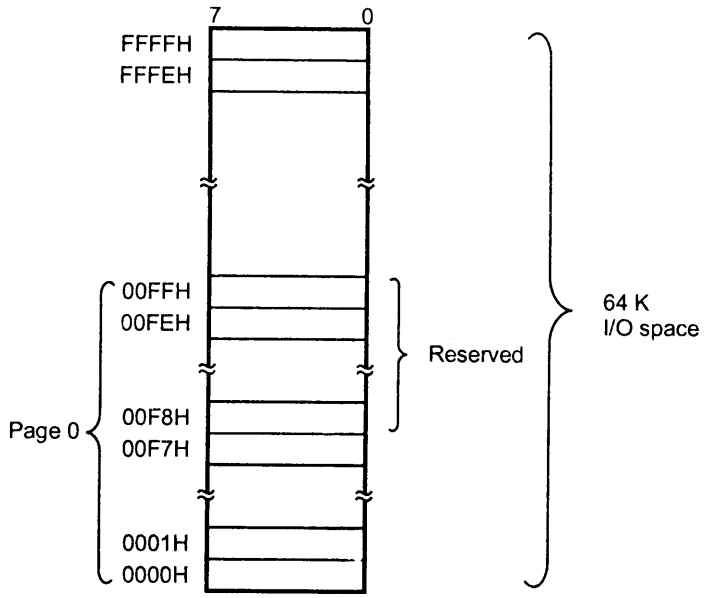


Fig. 7.4 I/O map for 8086

I/O ports are addressed in the same manner as memory locations. Even addressed bytes are transferred on the D_7-D_0 bus lines and odd addressed bytes on D_{15}-D_8. Care must be taken to assure that each register within an 8-bit peripheral located on the lower portion of the bus be addressed as even. In the I/O space, Intel has reserved 00F8H to 00FF locations.

7.5 General 8086 System Bus Structure and Operation

The 8086 has a common address and data bus. The address and data are time multiplexed, i.e. address and data appear on common bus at different time intervals. Thus bus is commonly known as multiplexed address and data bus. The multiplexed address and data bus provides the most efficient use of pins on the processor while permitting the use of a standard 40-lead package. This multiplexed address and data bus has to be demultiplexed externally with the use of latches and the ALE signal provided by 8086, as shown in Fig. 7.5.

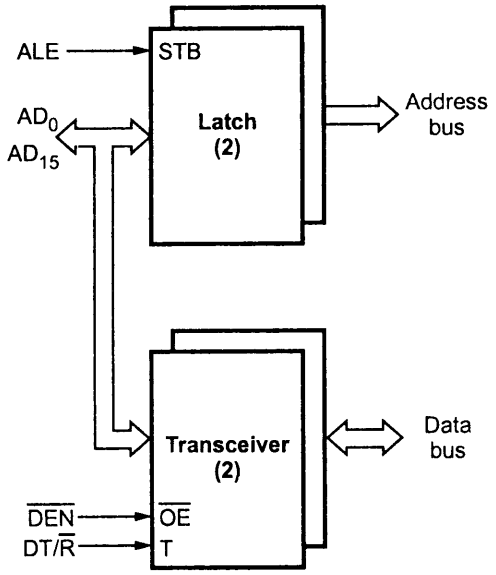


Fig. 7.5 Demultiplexing of address and data bus

Each processor bus cycle consists of atleast four clock cycles. These are referred to as T_1, T_2, T_3 and T_4. Refer Fig. 7.5. During T_1, processor sends address on the address bus and activates ALE signal. The ALE signal is used activate latches and thus to latch the address. The data transfer occurs on the bus during T_3 and T_4. The time interval T_2 is used primarily for changing the direction of the bus during read operations. Ready signal is sampled during T_3. The slower peripheral devices use this signal to indicate that the device is not ready to send the desired data within specified time. In the event that a "NOT Ready" indication is given by the slower peripheral device, 'WAIT' states (T_w) are inserted between T_2 and T_3 to give enough access time for the slower peripheral device.

Each WAIT state is of the same duration as a clock cycle. During a WAIT state, the signals on the buses remain the same as they were at the start of the WAIT state. If the Ready input is made high during wait state, then after WAIT state the 8086 will go on with the regular T_4 of the machine cycle. However, if the 8086 Ready input is still low at the end of a WAIT state, then the 8086 will insert another WAIT state. The 8086 will continue inserting WAIT states until the Ready input is made high again.

The status bits $\overline{S}_0$, $\overline{S}_1$ and $\overline{S}_2$ are used, in maximum mode, by the bus controller to identify the type of bus transaction according to the table given below.

$\overline{S}_2$	$\overline{S}_1$	$\overline{S}_0$	Machine cycle
0	0	0	Interrupt acknowledge
0	0	1	I/O read
0	1	0	I/O write
0	1	1	Halt

$\overline{S}_2$	$\overline{S}_1$	$\overline{S}_0$	Machine cycle
1	0	0	Instruction fetch
1	0	1	Memory read
1	1	0	Memory write
1	1	1	Inactive-Passive

Status bits S_3 through S_7 are multiplexed with high-order address bits and the $\overline{BHE}$ signal. These bits are also demultiplexed using latch and the ALE signal during T_1. Therefore, the status bits S_3 through S_7 are valid during T_2 through T_4. Status bits S_3 and S_4 indicate which segment register was used for this bus cycle in forming the address, according to the following table.

S_4	S_3	Characteristics
0	0	Alternate data (extra segment)
0	1	Stack
1	0	Code or None
1	1	Data

Out of remaining status bits, S_5 is a reflection of the interrupt enable bit of the flag register, S_6 is always 0 and S_7 is a spare status bit.

If a system is large enough to need data bus buffers, then the 8086 DT/$\overline{R}$ signal connected to these buffers will set them for input during a read operation or set them for output during a write operation. The 8086 $\overline{DEN}$ signal will enable the buffers at the appropriate time in the machine cycle, as shown in the Fig. 7.6.

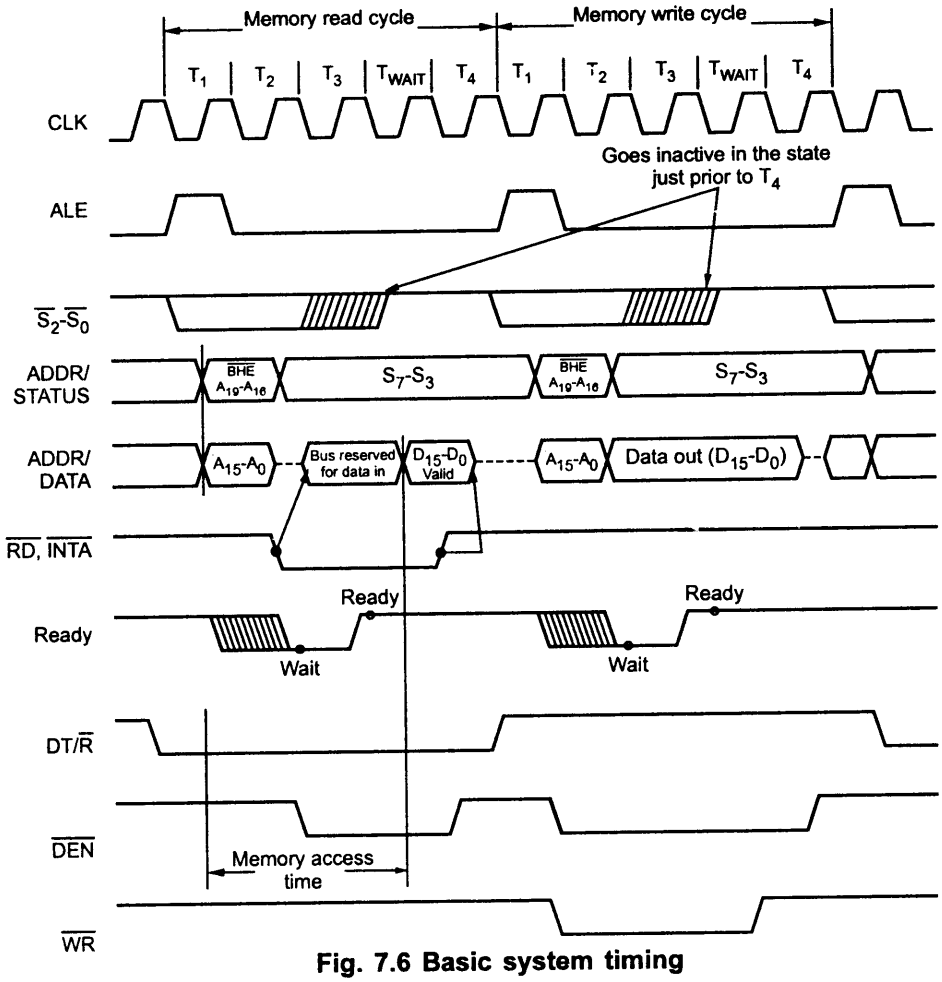


Fig. 7.6 Basic system timing

7.6 Minimum Mode 8086 System and Timings

7.6.1 Minimum Mode Configuration

Latching

Fig. 7.7 shows the typical minimum mode configuration. As shown in the figure, AD_0-AD_{15}, A_{16}/S_3-A_{19}/S_6, and $\overline{BHE}/S_7$ signals are multiplexed. These signals are demultiplexed by external latches and ALE signal generated by the processor. This is accomplished by using three latch ICs (Intel 8282/8283), two of them are required for a 16-bit address and three are needed if a full 20-bit address is used. In case of 8088, only two external latches are required. One for demultiplexing AD_0-AD_7 and other for demultiplexing A_{16}/S_3 and AD_{19}/S_6. Fig. 7.8 shows the internal block diagram of 8282/8283 latches. The 8282 provides noninverting outputs while the 8283 version inverts the input data. In addition to their demultiplexing function, these chips also buffer the address lines, providing increased output driving capability. The output low level is specified as 0.45 V maximum with a sink current of 32 mA maximum. The high level is specified as 2.4 V minimum while supplying a 5 mA maximum high level load current.

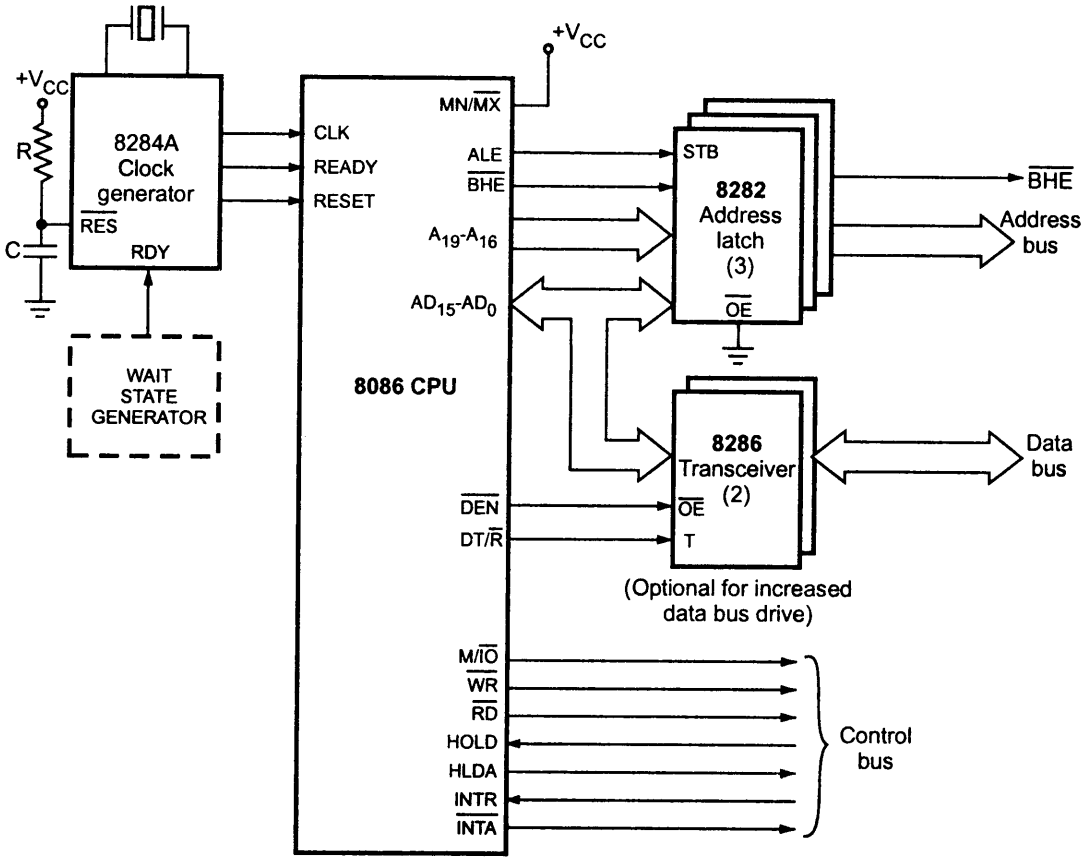


Fig. 7.7 Typical minimum mode configuration

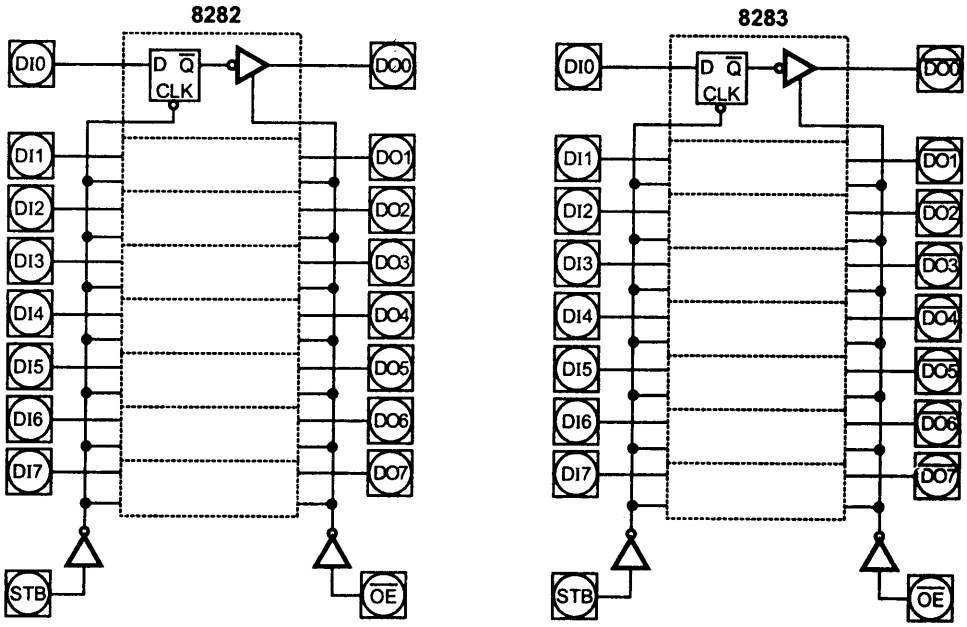


Fig. 7.8 Internal diagram of 8282 and 8283

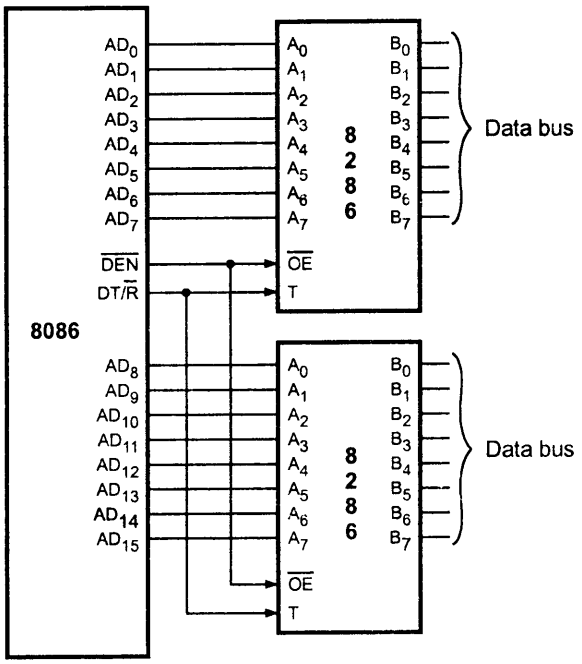


Fig. 7.9 Connection details of 8286

Buffering

If a system includes several interfaces then to increase current sourcing/sinking capacities it is necessary to use drivers and receivers (transceiver) for data bus also. The Intel 8286 device is used to implement the transceiver block shown in Fig. 7.7. The 8286 contains 16 tristate elements, eight receivers, and eight drivers. Therefore two 8286s are required to service 16 data lines of 8086. Fig. 7.9 shows the detailed connections of 8286.

$DT/\overline{R}$ signal is connected to the T input, which controls the direction of the data flow. When this signal is low, receivers are enabled, so that 8086 can read data from memory or input device. To write data into memory or output device, the 8086's $DT/\overline{R}$ signal goes high. Due to this drivers are enabled to transfer data from 8086 to the memory or the output device. At the time of data transfer, to enable output of transceiver its $\overline{OE}$ should be low. This is accomplished by connecting $\overline{DEN}$ signal of 8086 to the $\overline{OE}$ pin of 8286, since $\overline{DEN}$ signal goes low when CPU is ready to send or receive data.

Clock generator

The third component, other than the processor that appears in Fig. 7.7 is an 8284 clock generator. The 8284 clock generator does the following functions :

- Clock generation
- RESET synchronization
- READY synchronization
- Peripheral clock generation.

The Fig. 7.10 shows the internal logic diagram of 8284.

The top half of the logic diagram represents the clock and reset synchronization section of the 8284 clock generator. As shown in the logic diagram, the crystal oscillator has two inputs : X_1 and X_2. If a crystal is attached to X_1 and X_2, the oscillator generates a square-wave signal at the same frequency as the crystal. The output of oscillator is fed to an AND gate and also to an inverter buffer that provides the oscillator output signal. The $F/\overline{C}$ signal selects one of the oscillator inputs. When $F/\overline{C}$ input is 1, the EFI input determines the frequency; otherwise oscillator determines the frequency. When EFI input is

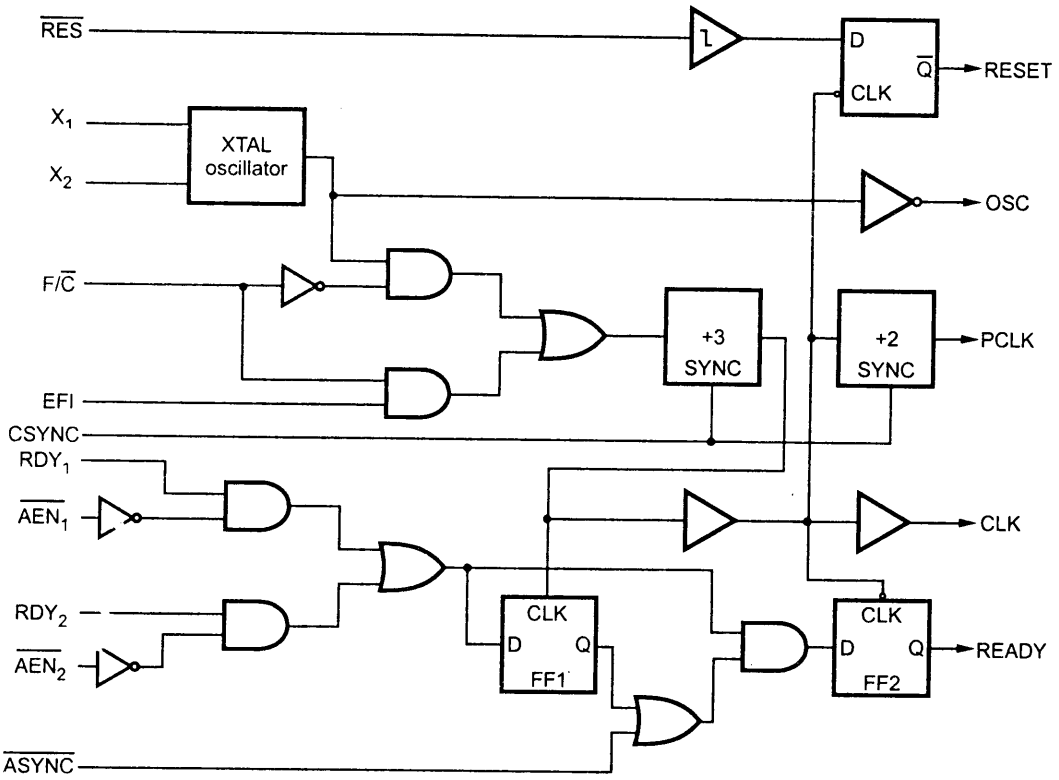


Fig. 7.10 The internal logic diagram of 8284

used, CSYNC signal is used for multiple processor system synchronization. If the internal crystal oscillator is used, CSYNC signal is grounded. In both the cases the output clock frequency is one third of the input frequency. The CLK signal is also buffered before it leaves the clock generator. As shown in the Fig. 7.10, the output of the divide-by-3 counter generates the timing for ready synchronization, a signal for another counter (divide-by-2), and the CLK signal to the 8086/8088 microprocessors. The two cascaded counters (divide-by-3 and divide-by-2) provide the divide-by-6 output at PCLK, which can be used to provide clock input for peripherals. The address enable pins, $\overline{AEN_1}$ and $\overline{AEN_2}$ are provided to qualify the bus ready signals, RDY_1 and RDY_2, respectively.

The reset circuit of 8284 consists of a schmitt trigger buffer and a single D flip-flop circuit. The D flip-flop ensures that the timing requirements of the 8086/8088 RESET input are met. This circuit applies the RESET signal to the microprocessor on the negative edge (1 to 0 transition) of each clock. The 8086/8088 microprocessors sample RESET at the positive edge (0 to 1 transition) of the clocks; therefore, this circuit meets the timing requirements of the 8086/8088.

The Fig. 7.11 shows the circuit connection for 8284 clock generator. The RC circuit provides a logic 0 to the $\overline{RES}$ input pin when power is first applied to the system. After a short time, the $\overline{RES}$ input becomes a logic 1 because the capacitor charges toward + 5.0 V through the register. A push button switch allows the microprocessor to be reset by the operator.

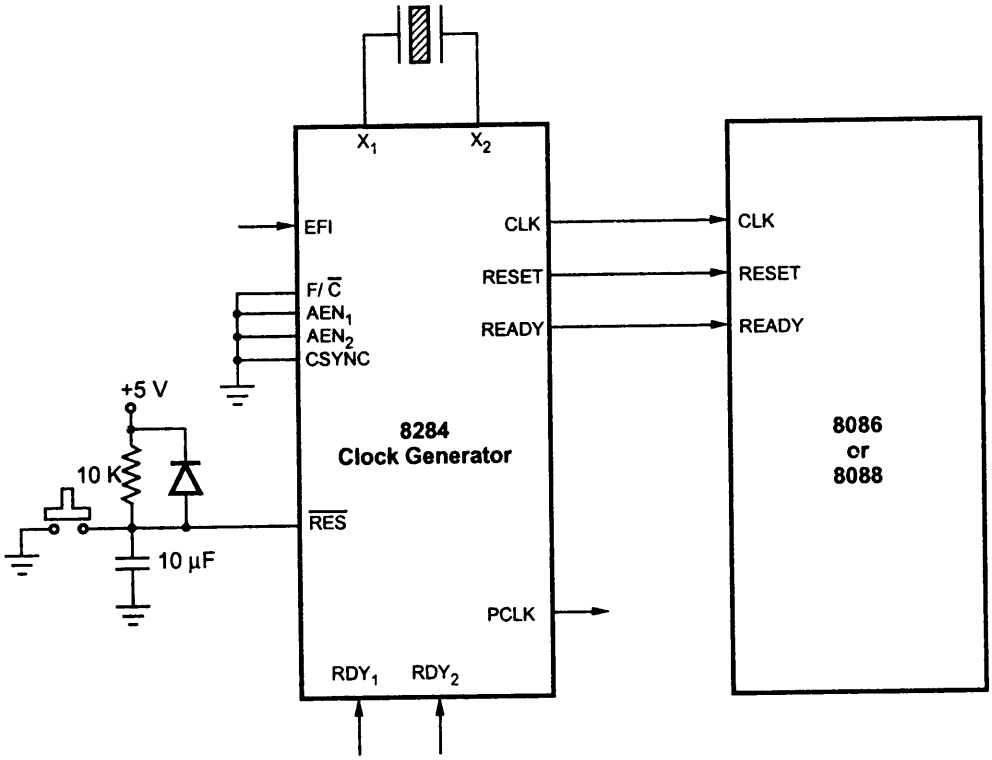


Fig. 7.11 Interfacing of 8284 clock generator with 8086 or 8088

Other signals

The status on the M/$\overline{IO}$, $\overline{RD}$, and $\overline{WR}$ lines decides the type of data transfer, as listed in the Table 7.1.

M/$\overline{IO}$	$\overline{RD}$	$\overline{WR}$	Operation
0	0	1	I/O read
0	1	0	I/O write
1	0	1	Memory read
1	1	0	Memory write

Table 7.1

HOLD and HLDA signals are used to interface other bus masters like DMA controller. Interrupt request (INTR) and interrupt acknowledge ($\overline{INTA}$) are used to extend the interrupt handling capacity of the 8086 with the help of interrupt controller.

7.6.2 Minimum Mode 8086 System

The Fig. 7.12 shows the typical minimum mode 8086 system. Here, interfacing of memory and I/O devices are shown with the basic minimum mode 8086 configuration.

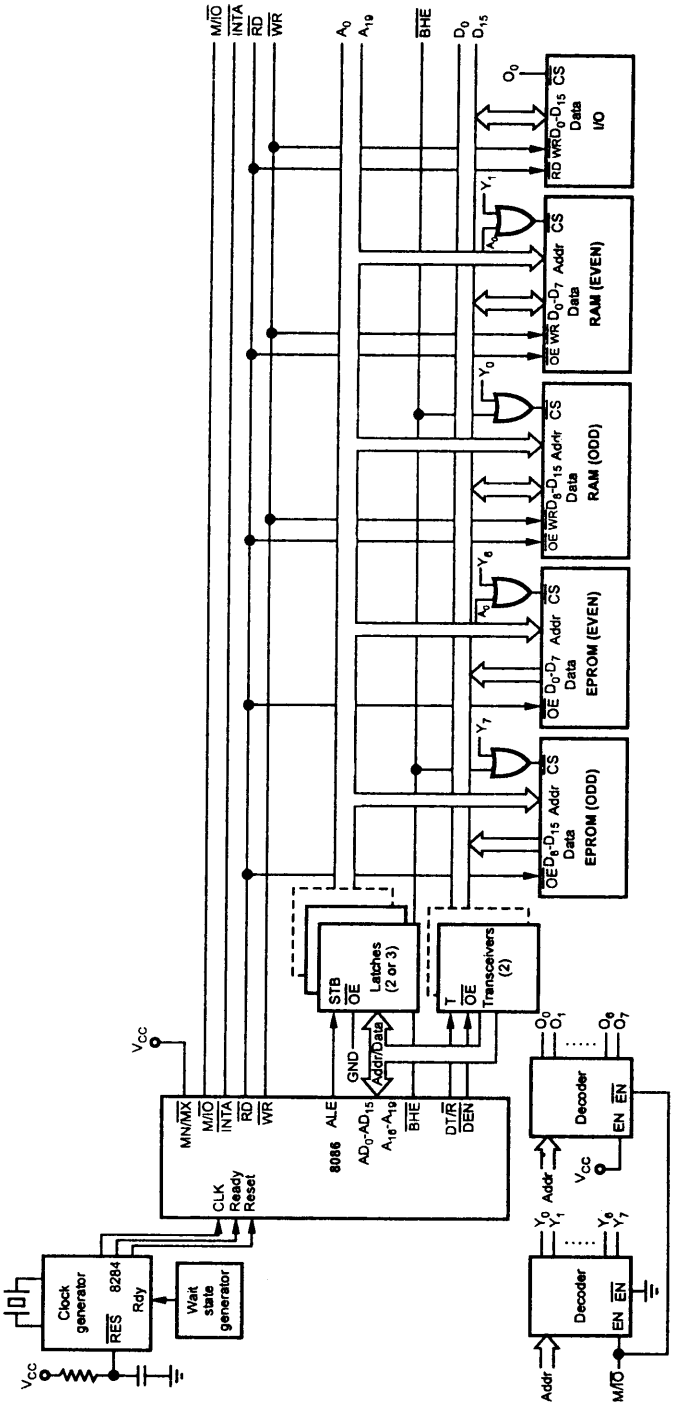


Fig. 7.12 Minimum mode 8086 system

For interfacing memory module to 8086, it is necessary to have odd and even memory banks. This is implemented by using two EPROMs and two RAMs. Data lines D_{15}-D_8 are connected to odd bank of EPROM and RAM, and data lines D_7-D_0 are connected to even bank of EPROM and RAM. Address lines are connected to EPROM and RAM as per their capacities. $\overline{RD}$ signal is connected to the output enable ($\overline{OE}$) signals of EPROMs and RAMs. $\overline{WR}$ signal is connected to $\overline{WR}$ signal of RAMs. Two separate decoders are used to generate chip select signals for memory and I/O devices. These chip select signals are logically ORed with either $\overline{BHE}$ or A_0 to generate final chip select signals. For generating final chip select signals for odd bank decoder outputs are logically ORed with $\overline{BHE}$ signal. On the other hand to generate final chip select signals for even bank decoder outputs are logically ORed with A_0 signal.

The 16-bit I/O interface is shown in the Fig. 7.12. $\overline{RD}$ and $\overline{WR}$ signals are connected to the $\overline{RD}$ and $\overline{WR}$ signals of I/O device. Data lines D_{15}-D_0 are connected to the data lines of I/O device. The chip select signal for I/O device is generated using separate decoder whose output is enabled only when M/$\overline{IO}$ signal is low.

7.6.3 Bus Timings for Minimum Mode

7.6.3.1 Timings for Read and Write Operations

The timing diagrams of input and output transfers for 8086 minimum mode are shown in the Fig. 7.13 (a) and (b) respectively.

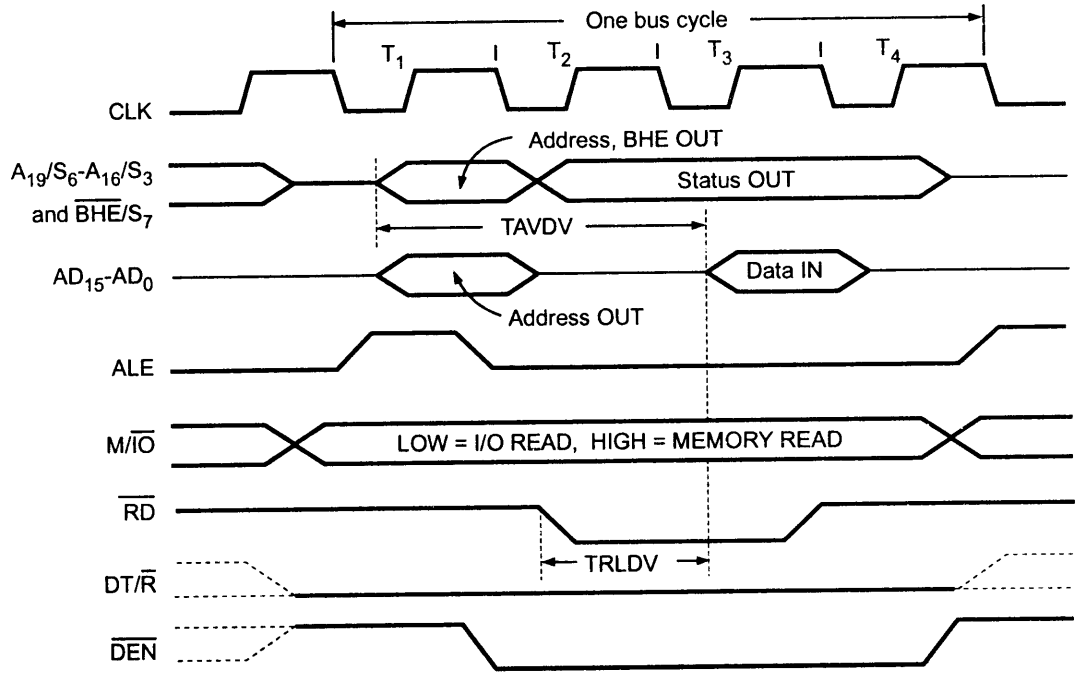


Fig. 7.13 (a) Input (read operation)