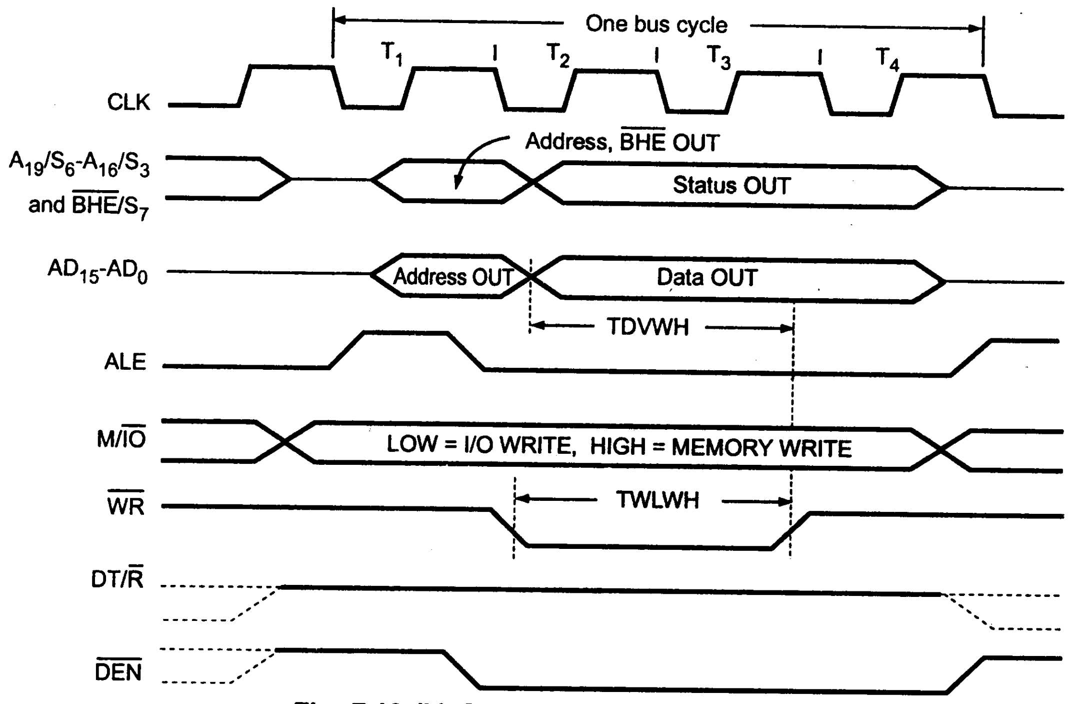


Fig. 7.13 (b) Output (write operation)

These are explained in steps.

1. When processor is ready to initiate the bus cycle, it applies a pulse to ALE during T_1. Before the falling edge of ALE, the address, $\overline{BHE}$, $M/\overline{IO}$, $\overline{DEN}$ and $DT/\overline{R}$ must be stable i.e. DEN = high and $DT/\overline{R} = 0$ for input or $DT/\overline{R} = 1$ for output.
2. At the trailing edge of ALE, ICs 74LS373 or 8282 latches the address.
3. During T_2 the address signals are disabled and S_3-S_7 are available on AD_{16}/S_3-AD_{19}/S_6 and $\overline{BHE}/S_7$. Also $\overline{DEN}$ is lowered to enable transceiver.
4. In case of input operation, $\overline{RD}$ is activated during T_2 and AD_0 to AD_{15} go in high impedance preparing for input.
5. If memory or I/O interface can perform the transfer immediately; there are no wait states and data is output on the bus during T_3.
6. After the data is accepted by the processor, $\overline{RD}$ is raised high at the beginning of T_4.
7. Upon detecting this transition during T_4, the memory or I/O device will disable its data signals.
8. For an output operation, processor applies $\overline{WR} = 0$ and then the data on the data bus during T_2.
9. In T_4, $\overline{WR}$ is raised high and data signals are disabled.

10. For either input or output operation, $\overline{DEN}$ is raised during T_4 to disable the transceiver. Also $M/\overline{IO}$ is set according to the next transfer at this time or during next T_1 state. Thus length of bus cycle in 8086 is four clock cycle. If the bus is to be inactive after completion of bus cycle, then the gap between the successive cycles is filled by ideal state clock cycles.

When the memory or I/O device is not able to respond quickly during transfer, wait states (T_W) are inserted between T_3 and T_4 by disabling the READY input of the 8086. The bus activity during wait state is same as during T_3.

7.6.3.2 HOLD Response Sequence

The Fig. 7.14 shows the HOLD and HLDA signal timings in minimum mode system. The HOLD pin is sampled at leading edge of each clock pulse. If it is sampled active by the 8086 before T_4 of the previous cycle or during T_1 state of the current cycle, the 8086 activates HLDA in the next clock cycle and for succeeding bus cycles. It relinquishes the control of all buses and the control of buses is handed over to the requesting master. The control of the bus is not regained by the 8086 until the requesting master does not inactivate the HOLD pin. After inactivation of HOLD signal, 8086 regains the control of buses and inactivates the HLDA signal.

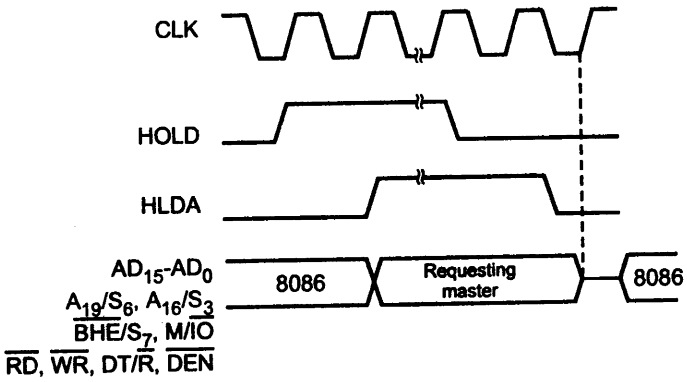


Fig. 7.14 HOLD and HLDA signal timings

7.7 Maximum Mode 8086 System and Timings

7.7.1 Maximum Mode Configuration

Fig. 7.15 shows the typical maximum mode configuration. In the maximum mode additional circuitry is required to translate the control signals. The additional circuitry converts the status signals ($\overline{S_2}$-$\overline{S_0}$) into the I/O and memory transfer signals. It also generates the control signals required to direct the data flow and for controlling 8282 latches and 8286 transceivers. The Intel 8288 bus controller is used to implement this control circuitry.

Fig. 7.16 shows that the 8288 is able to originate the address latch enable signal to the 8282's, the enable and direction signals to the 8286 transceivers, and the interrupt acknowledge signal to the interrupt controller. It also decodes the $\overline{S_2}$-$\overline{S_0}$ signals to generate $\overline{MRDC}$, $\overline{MWTC}$, $\overline{IORC}$, $\overline{IOWC}$, MCE/$\overline{PDEN}$, $\overline{AEN}$, IOB, CEN, AIOWC, and AMWC signals.

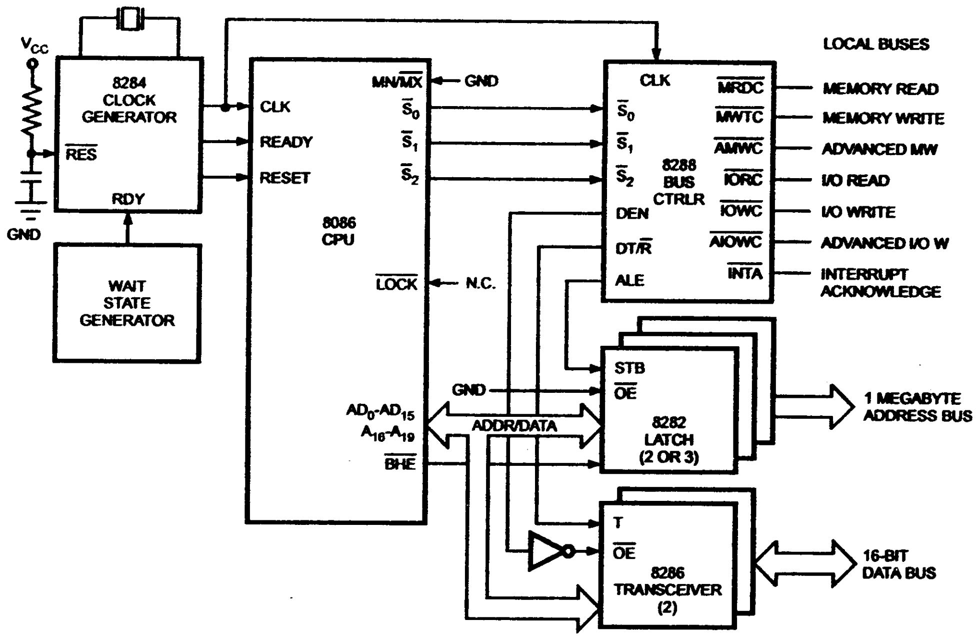


Fig. 7.15 Typical maximum mode configuration

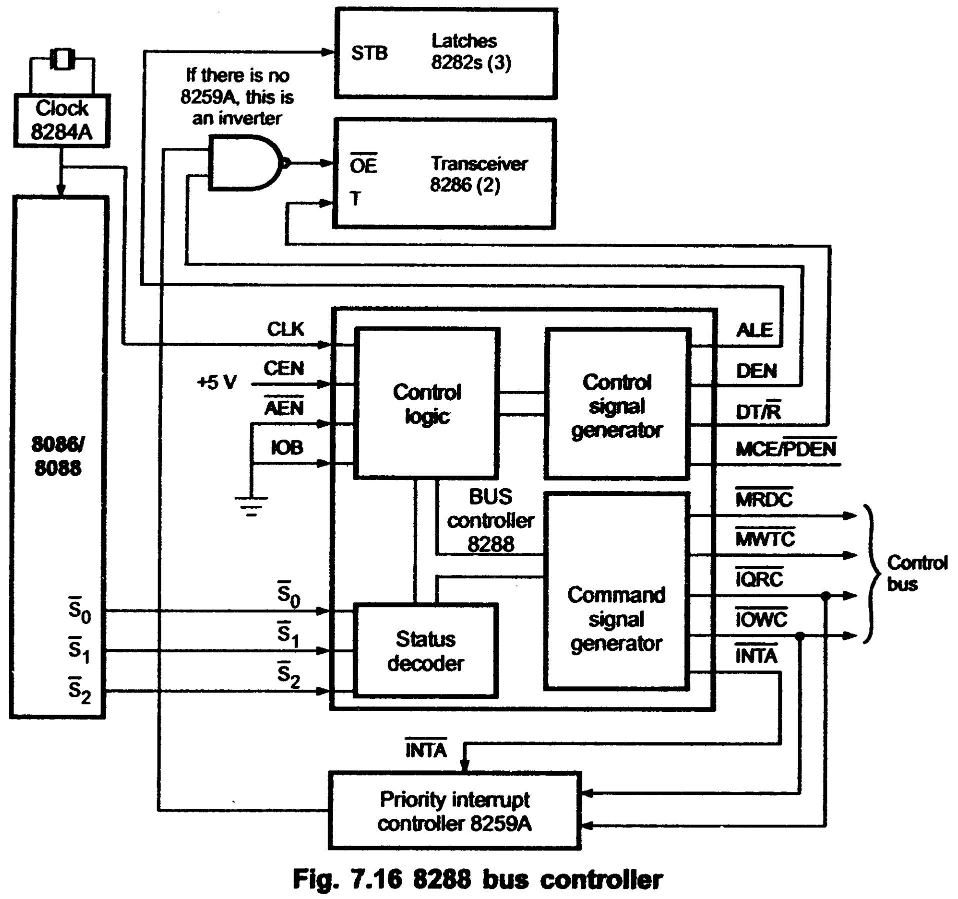


Fig. 7.16 8288 bus controller

$\overline{\text{MRDC}}$ (Memory Read Command) : It instructs the memory to put the contents of the addressed location on the data bus.

$\overline{\text{MWTC}}$ (Memory Write Command) : It instructs the memory to accept the data on the data bus and load the data into the addressed memory location.

$\overline{\text{IORC}}$ (I/O Read Command) : It instructs an I/O device to put the data contained in the addressed port on the data bus.

$\overline{\text{IOWC}}$ (I/O Write Command) : It instructs an I/O device to accept the data on the data bus and load the data into the addressed port.

MCE/$\overline{\text{PDEN}}$ (Master Cascade Enable/Peripheral Data Enable) : It controls the mode of operation of 8259. It selects cascade operation for 8259 (interrupt controller) if IOB signal is grounded and enables the I/O bus transceivers if IOB is tied high.

$\overline{\text{AEN}}$, IOB and CEN : These pins are used in multiprocessor system. With a single processor in the system, $\overline{\text{AEN}}$ and IOB are grounded and CEN is tied high. $\overline{\text{AEN}}$ causes the 8288 to enable the memory control signals. IOB (I/O bus mode) signal selects either the I/O bus mode or system bus mode operation. CEN (control enable) input enables the command output pins on the 8288.

$\overline{\text{AIOWC}}$/$\overline{\text{AMWC}}$ (Advance I/O Write Command/Advance Memory Write Command) : These signals are similar to IOWC and MWTC except that they are activated one clock pulse earlier. This gives slow interfaces an extra clock cycle to prepare to input the data.

7.7.2 Maximum Mode 8086 System

The Fig. 7.17 shows the typical maximum mode 8086 system. Here interfacing of memory and I/O devices are shown with the basic maximum mode configuration. The connections for memory and I/O devices are similar to that of minimum mode configuration. However, the generation of control signals from 8086 is done by external bus controller 8288.

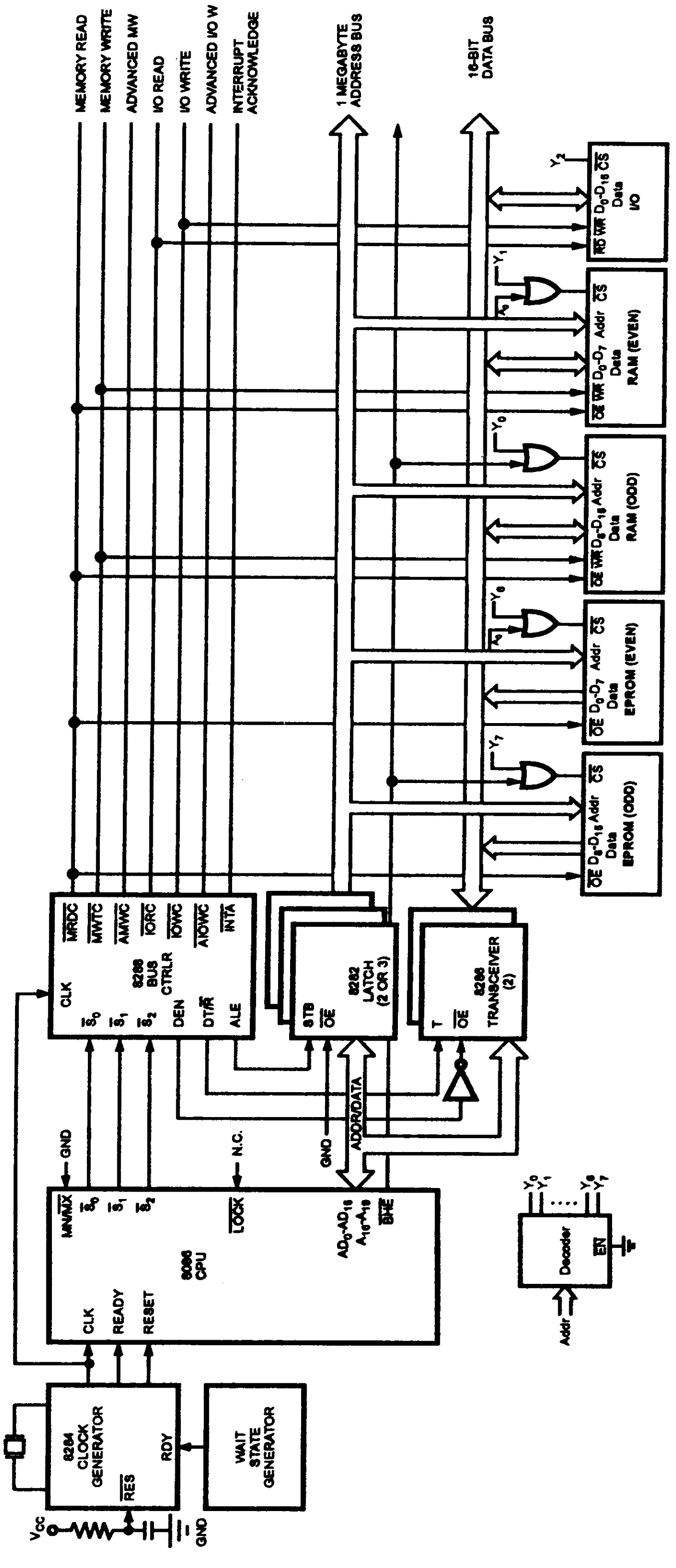


Fig. 7.17 Typical maximum mode 8086 system

7.7.3 Bus Timings for Maximum Mode

7.7.3.1 Timings for Read and Write Operations

The timing diagrams of input and output transfers are shown in the Fig. 7.18 (a) and (b) respectively.

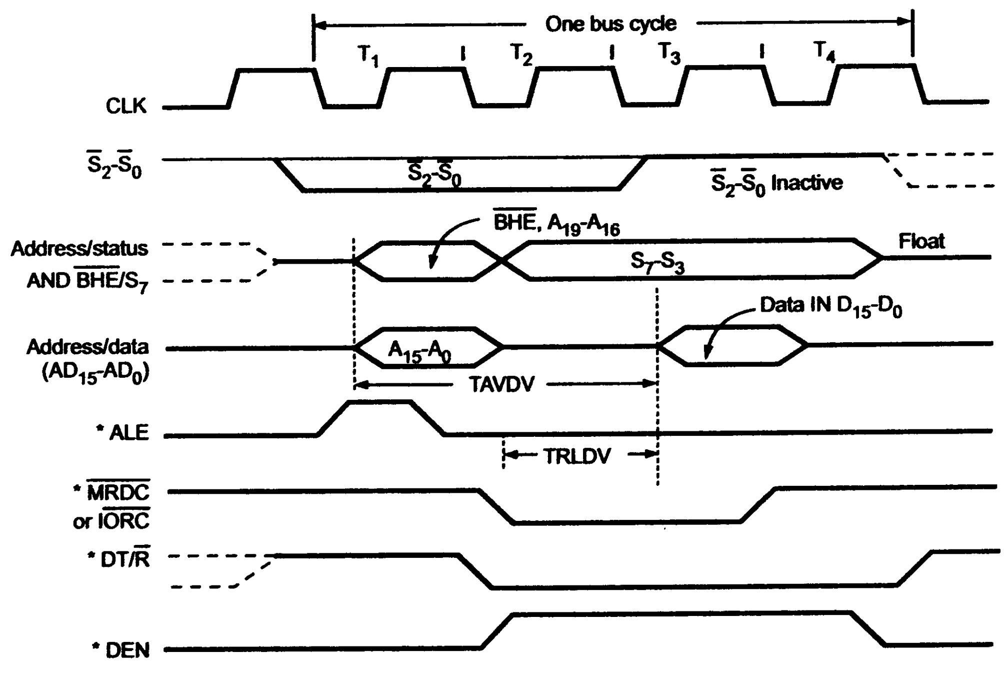

One bus cycle
T_1
T_2
T_3
T_4
CLK
$\overline{S}_2$-$\overline{S}_0$
$\overline{S}_2$-$\overline{S}_0$
$\overline{S}_2$-$\overline{S}_0$ Inactive
Address/status AND $\overline{BHE}/S_7$
$\overline{BHE}$, A_{19}-A_{16}
S_7-S_3
Float
Data IN D_{15}-D_0
Address/data (AD_{15}-AD_0)
A_{15}-A_0
TAVDV
* ALE
TRLDV
* $\overline{MRDC}$ or $\overline{IORC}$
* DT/$\overline{R}$
* DEN

Fig. 7.18 (a) Input (read operation)

These are explained in steps.

1. S_0, $\overline{S}_1$, $\overline{S}_2$ are set at the beginning of bus cycle. On detecting the change on passive state $\overline{S}_0 = \overline{S}_1 = \overline{S}_2 = 1$, the 8288 bus controller will output a pulse on its ALE and apply a required signal to its DT/$\overline{R}$ pin during T_1.
2. In T_2, 8288 will set DEN = 1 thus enabling transceiver. For an input, 8288 it will activates $\overline{MRDC}$ or $\overline{IORC}$. These signals are activated until T_4. For an output, the $\overline{AMWC}$ or $\overline{AIOWC}$ is activated from T_2 to T_4 and $\overline{MWTC}$ or $\overline{IOWC}$ is activated from T_3 to T_4.
3. The status bits $\overline{S}_0$ to $\overline{S}_2$ remain active until T_3, and become passive during T_3 and T_4.
4. If ready input is not activated before T_3, wait state will be inserted between T_3 and T_4.

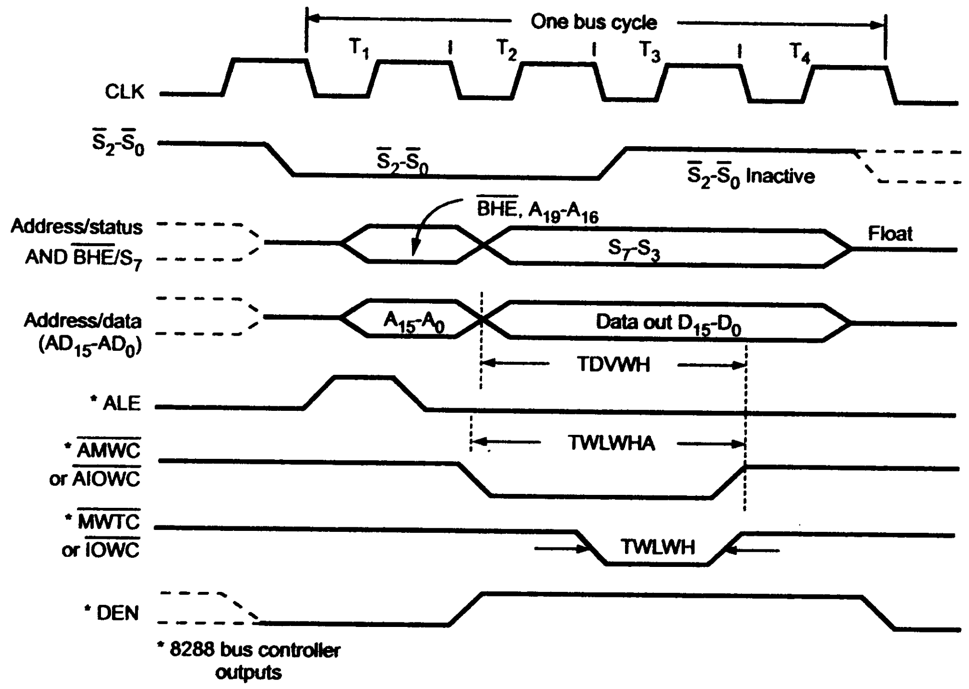

One bus cycle
T_1
T_2
T_3
T_4
CLK
$\overline{S_2}$-$\overline{S_0}$
$\overline{S_2}$-$\overline{S_0}$
$\overline{S_2}$-$\overline{S_0}$ Inactive
Address/status AND $\overline{BHE}/S_7$
$\overline{BHE}$, A_{19}-A_{16}
S_7-S_3
Float
Address/data (AD_{15}-AD_0)
A_{15}-A_0
Data out D_{15}-D_0
TDVWH
* ALE
* $\overline{AMWC}$ or $\overline{AIOWC}$
TWLWHA
* $\overline{MWTC}$ or $\overline{IOWC}$
TWLWH
* DEN
* 8288 bus controller outputs

Fig. 7.18 (b) Output (write operation)

7.7.3.2 Timings for $\overline{RQ}/\overline{GT}$ Signals

The Fig. 7.19 shows the timing diagrams for bus request and bus grant pins of 8086. A bus request, grant and release is accomplished by a sequence of three pulses. The $\overline{RQ}/\overline{GT}$ pins are sampled at the rising edge of each clock pulse and if a request is detected the 8086 will apply a grant pulse to the $\overline{RQ}/\overline{GT}$ if the following conditions are met.

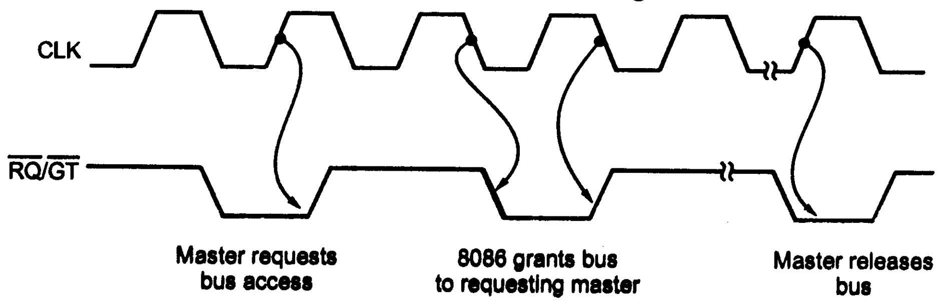

CLK
$\overline{RQ}/\overline{GT}$
Master requests bus access
8086 grants bus to requesting master
Master releases bus

Fig. 7.19 Timings for bus request and bus grant signals

1. The previous bus transfer was not the low byte of a word to or from an odd address if the CPU is an 8086. For an 8088, regardless of the address alignment, the grant signal will not be sent until the second byte of a word reference is accessed.
2. The first pulse of an interrupt acknowledgement did not occur during the previous bus cycle.
3. An instruction with a LOCK prefix is not being executed.

If condition 1 and 2 is not met, then the grant will not be given until the next bus cycle, and if condition 3 is not met, the grant will wait until the locked instruction is completed.

After activation of grant pulse, requesting master takes the control of buses. This master may control the buses for only one bus cycle or for several bus cycles. When it is ready to relinquish the buses it will send the processor release pulse over the same line that it made its request.

7.8 Minimum Mode and Maximum Mode 8088 Systems

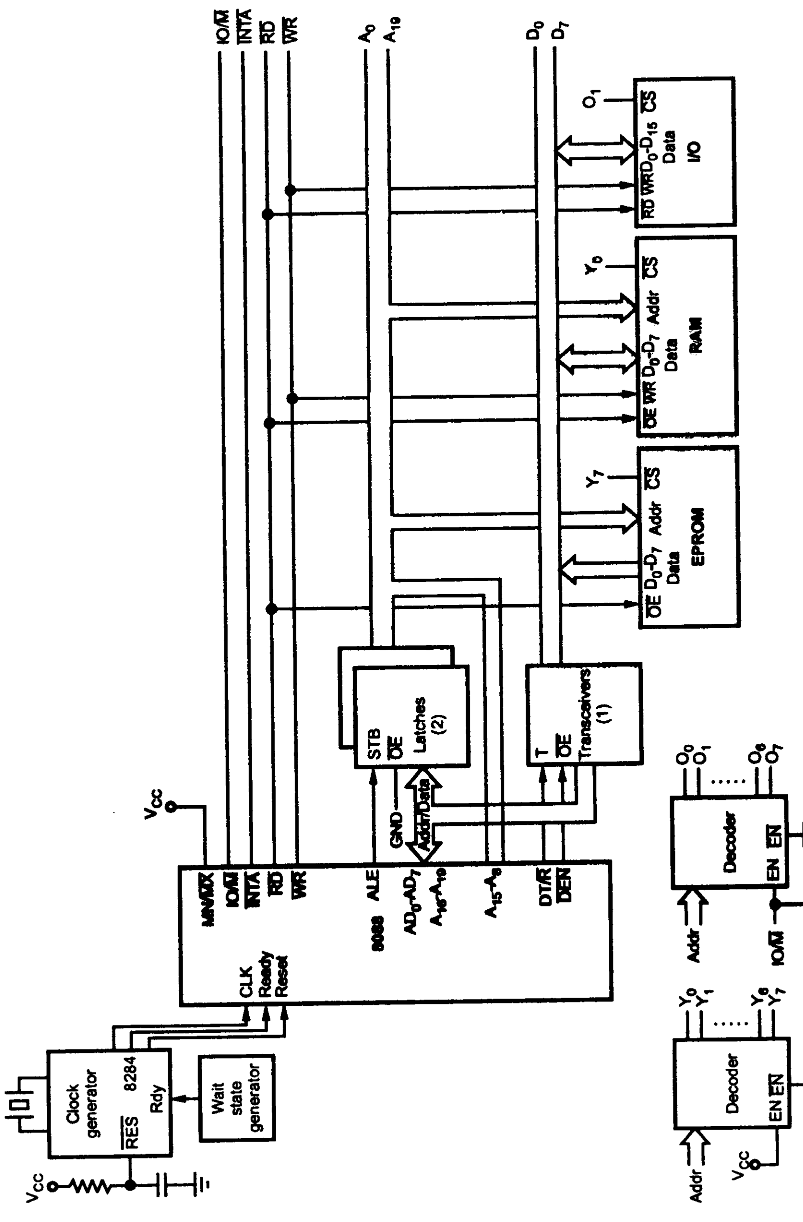


Fig. 7.20 (a) Minimum mode 8088 system

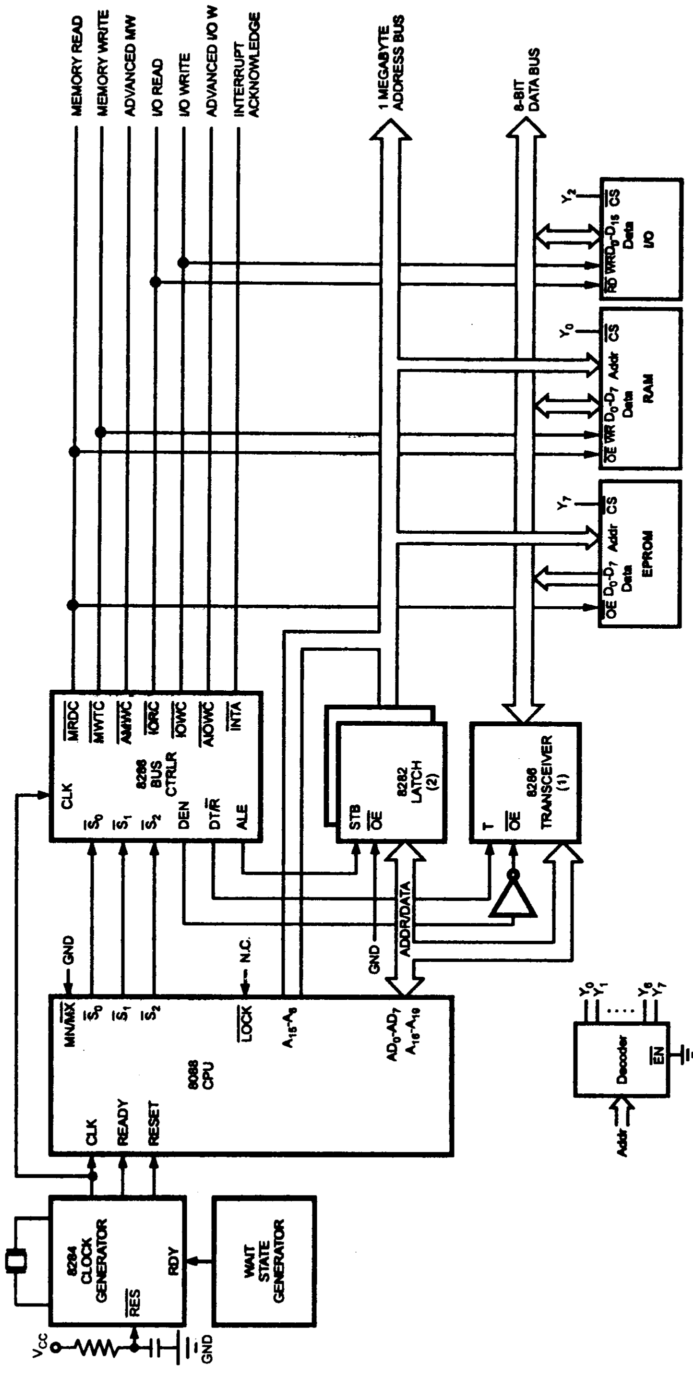


Fig. 7.20 (b) Maximum mode 8088 system

The Fig. 7.20 shows the minimum mode and maximum mode 8088 systems. Like 8086, 8088 has 20 address lines and hence its addressing capability is 1 Mbyte. However, since

8088 has 8-bit data bus, it is not to have separate odd and even memory banks like 8086. Rest of the interfacing is same as that of 8086 systems.

7.9 Typical 8088 System Timing Diagram

The Fig. 7.21 shows the typical 8088 system timing diagram. As shown in the Fig. 7.21, the address/data bus of 8088 system is split into three parts : 1. The lower 8-bit multiplexed address and data bus (AD_7-AD_0) 2. Middle 8-bit address bus (A_{15}-A_8) 3. The upper 4-bit multiplexed address and status bus (A_{19}/S_6 - A_{16}/S_3). Like 8086, each bus cycle of 8088 contains at least four clock cycles : T_1, T_2, T_3 and T_4. During T_1, 8088 sends address on the address bus and activates ALE signal. The ALE signal is used to activate latches and thus to latch the address. The data transfer takes place during T_3 and T_4.

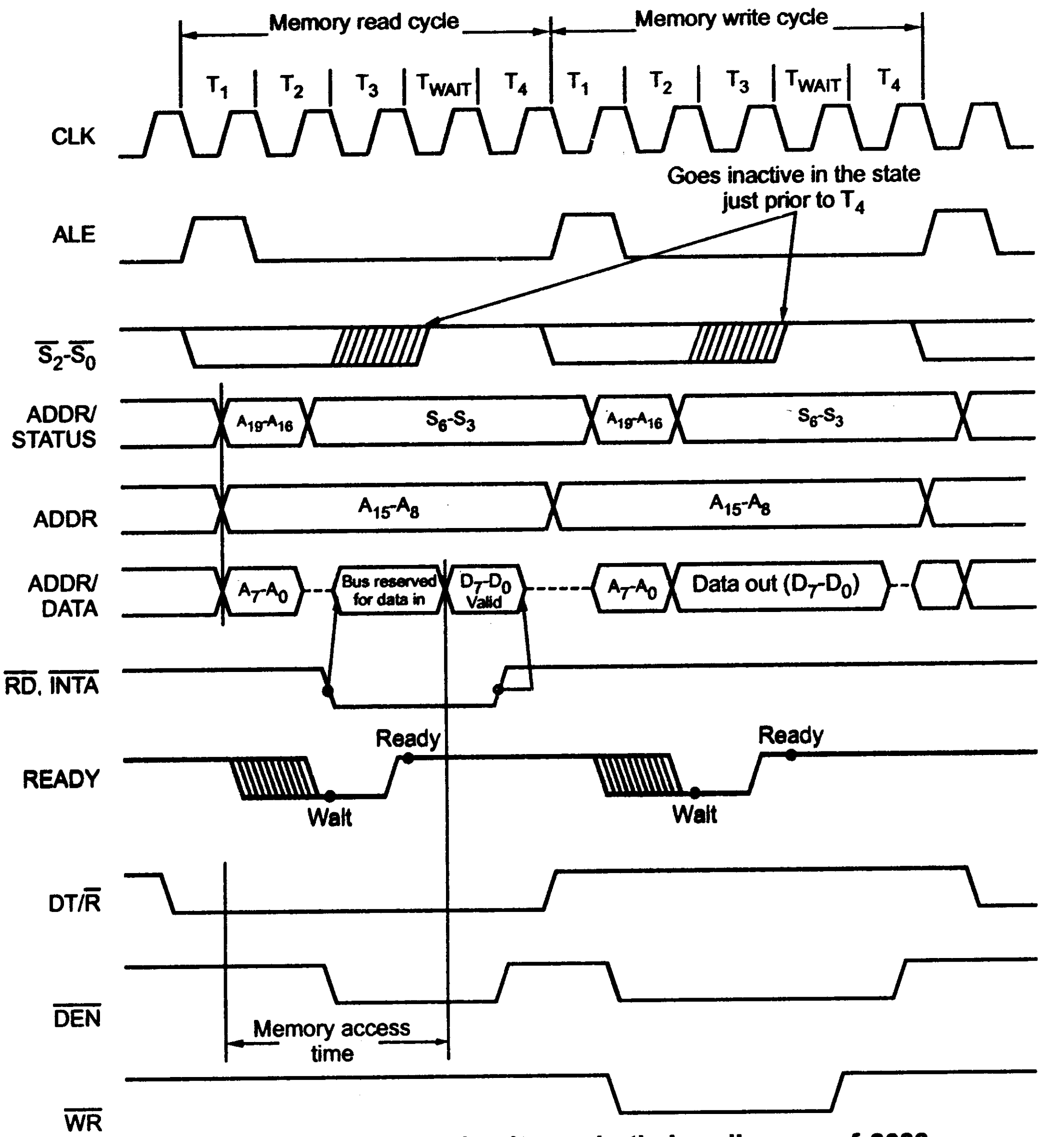


Fig. 7.21 Memory read and write cycle timing diagram of 8088

The time interval T_2 is used primarily for changing the direction of the bus during read operation. Ready signal is sampled during T_3. If Ready signal is not active, the wait state is inserted in the bus cycle. $\overline{RD}$ and $\overline{WR}$ signals are activated in their respective machine cycles. If a system is large enough to need data bus buffers, the 8088 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 8088 $\overline{DEN}$ signal will enable the buffers at the appropriate time in the machine cycle, as shown in the Fig. 7.21.

Review Questions

1. *Explain the function of following pins in 8086.*
   i) NMI ii) MN/$\overline{MX}$ iii) $\overline{TEST}$ iv) $\overline{BHE}$ v) DT/$\overline{R}$ vi) $\overline{DEN}$ vii) QS_0-QS_1.
2. *Explain the maximum mode signals of 8086.*
3. *Explain the minimum mode signals 8086.*
4. *With the help of block diagram explain memory interfacing with 8086 and explain why two bus cycles are required to access odd address word ?*
5. *Draw and explain the memory map for 8086.*
6. *Explain the I/O addressing capabilities of 8086.*
7. *Draw and explain the I/O map of 8086.*
8. *Explain the general bus operation of 8086 with the help of timing diagram.*
9. *Explain the purpose of Ready, $\overline{DEN}$ and DT/$\overline{R}$ signals.*
10. *With the help of block schematic diagrams explain the operation of 8284 clock generator and 8286 transceiver.*
11. *Sketch block diagram showing basic 8086 minimum mode system. Explain functions of 8282 latches and 8286 transceiver.*
12. *Define bus cycle, and explain the minimum mode read and write bus cycle with proper timing diagram.*
13. *Explain the HOLD response sequence in the minimum mode of 8086 with the help of timing diagram.*
14. *Draw and explain a block diagram showing 8086 in maximum mode configuration.*
15. *Draw and explain the timing diagrams of input and output transfers of 8086 in maximum mode.*
16. *Indicate the signals which are different when 8086 in minimum mode and in maximum mode.*
17. *Explain the operation of bus request and bus grant signal with the help of timing diagram.*
18. *Draw and explain the minimum mode system with 8088 processor.*
19. *Draw and explain the maximum mode system with 8088 processor.*
20. *Draw and explain the typical 8088 system timing diagram.*

8 Memory and I/O Interfacing

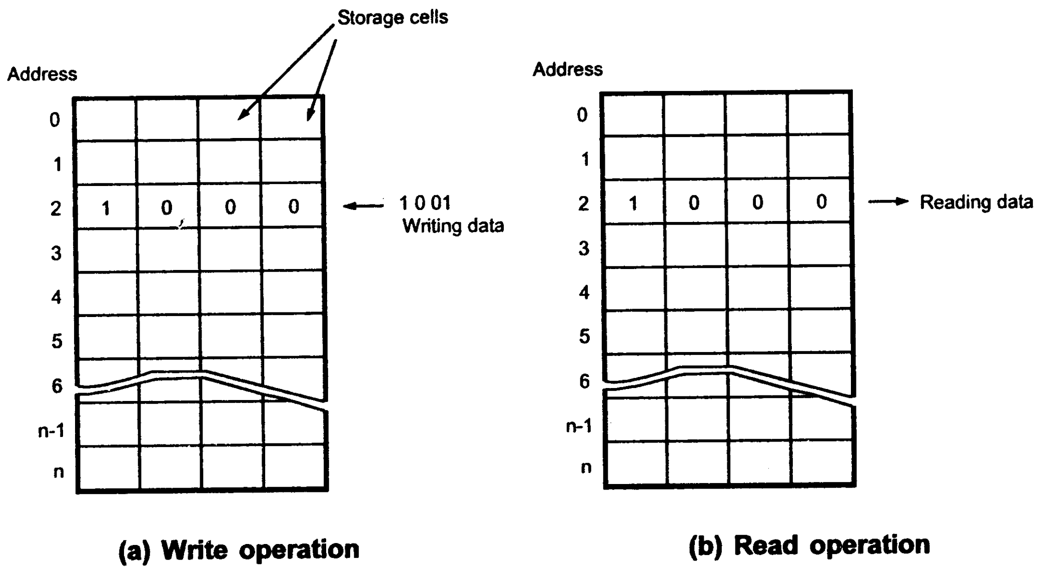
Memory is an integral part of a microprocessor system, and in this chapter, we will discuss how to interface a memory device with the microprocessor. The memory interfacing circuit is used to access memory quite frequently to read instruction codes and data stored in memory. This read/write operations are monitored by control signals. The microprocessor activates these signals when it wants to read from and write into memory. In this chapter we will see memory structure and its requirements, concepts in memory interfacing and interfacing examples. The later part of this chapter explains the basic concept in I/O interfacing.

8.1 Terminology and Operations

Memories are made up of registers. Each register in the memory is one storage location. Each location is identified by an **address**. The number of storage locations can vary from a few in some memories to hundreds of thousand in others. Each location can accommodate one or more bits. Generally, the total number of bits that a memory can store is its **capacity**. Most of the types the capacity is specified interms of bytes (group of eight bits)

Each register consists of storage elements (flip-flops or capacitors in semiconductor memories and magnetic domain in magnetic storage), each of which stores one bit of data. A storage element is called a **cell.**

The data stored in a memory by a process called **writing** and are retrieved from the memory by a process called **reading**. Fig. 8.1 illustrates in a very simplified way the concept of write, read, address and storage capacity for a generalized memory.

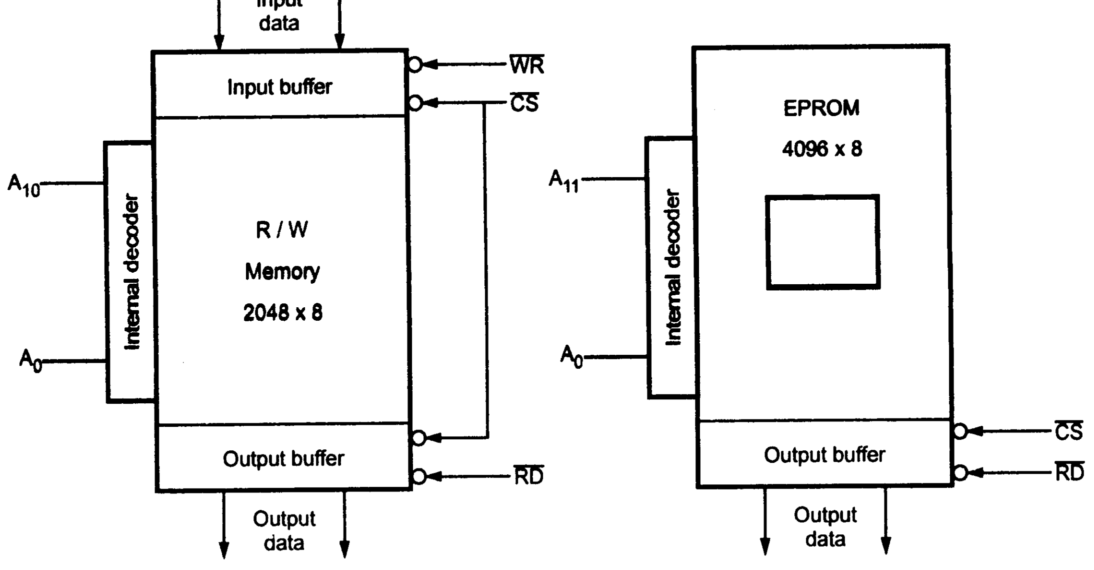


(a) Write operation (b) Read operation

Fig. 8.1

8.2 Memory Structure and its Requirements

Read/write memories consist of an array of registers, in which each register has unique address. The size of the memory is N x M as shown in Fig. 8.2 (a) where N is the number of registers and M is the word length, in number of bits.



(a) Logic diagram for RAM (b) Logic diagram for EPROM

Fig. 8.2

Example 1 : If memory is having 12 address lines and 8 data lines, then

Number of registers/memory locations = $2^N = 2^{12}$

$= 4096$

Word length = M bit

= 8-bit

Example 2 : If memory has 8192 memory locations, then it has 13 address lines.

The Table 8.1 summarizes the memory capacity and address lines required for memory interfacing.

Memory Capacity	Address Lines Required
1 K = 1024 memory locations	10
2 K = 2048 memory locations	11
4 K = 4096 memory locations	12
8 K = 8192 memory locations	13
16 K = 16384 memory locations	14
32 K = 32768 memory locations	15
64 K = 65536 memory locations	16

Table 8.1

As shown in the Fig. 8.2 (a) memory chip has 12 address lines A_0-A_{11}, one chip select ($\overline{CS}$), and two control lines, read ($\overline{RD}$) to enable output buffer and write ($\overline{WR}$) to enable the input buffer. The internal decoder is used to decode the address lines. Fig. 8.2 (b) shows the logic diagram of a typical EPROM (Erasable Programmable Read-Only Memory) with 4096 (4 K) registers. It has 12 address lines A_0-A_{11}, one chip select ($\overline{CS}$), one Read control signal. Since EPROM is a read only memory, it does not require the ($\overline{WR}$) signal.

8.3 Basic Concepts in Memory Interfacing

For interfacing memory devices to microprocessor 8086/88 following important points are to be kept in mind.

1. Microprocessor 8086/88 can access 1 Mbytes memory since address bus is 20-bit. But it is not always necessary to use full 1 Mbytes address space. The total memory size depends upon the application.
2. Generally EPROM (or EPROMs) is used as a program memory and RAM (or RAMs) as a data memory. When both, EPROM and RAM are used, the total address space 1 Mbytes is shared by them.

3. The individual capacities of program memory and data memory depend on the application.
4. It is not always necessary to select 1 EPROM and 1 RAM. We can have multiple EPROMs and multiple RAMs as per the requirement of application.
5. We can place EPROM/RAM anywhere in full 1 Mbytes address space. But program memory (EPROM) should be located at last memory page so that the starting address FFFF0H will lie within the program memory range. To provide facility to set addresses in the interrupt vector table we must provide RAM at page 0 of memory. So that the interrupt vector table lie with the read/write memory range.
6. It is not always necessary to locate EPROM and RAM in consecutive memory addresses. However, it is advised to do that.
7. While interfacing memory to 8086 we have to provide odd and even banks of memory. Even bank is selected when $A_0 = 0$ and odd bank is selected when $\overline{BHE} = 0$. Odd and even banks are not required for interfacing memory to 8088.

The memory interfacing requires to :

- Select the chip
- Identify the register
- Enable the appropriate buffer.

Microprocessor system includes memory devices and I/O devices. It is important to note that microprocessor can communicate (read/write) with only one device at a time, since the data, address and control buses are common for all the devices. In order to communicate with memory or I/O devices, it is necessary to decode the address from the microprocessor. Due to this each device (memory or I/O) can be accessed independently.

8.3.1 Address Decoding Techniques

1) Absolute Decoding

2) Linear Decoding

3) Block Decoding.

1) Absolute Decoding : In absolute decoding technique the memory chip is selected only for the specified logic level on the address lines; no other logic levels can select the chip. Fig 8.3 shows the memory interface with absolute decoding. Two 8 K EPROMs (2764) are used to provide even and odd memory banks. Control signals $\overline{BHE}$ and A_0 are used to enable outputs of odd and even memory banks respectively. As each memory chip has 8 K memory locations, thirteen address lines are required to address each locations, independently. All remaining address lines are used to generate an unique chip select signal. This addressing technique is normally used in large memory systems.

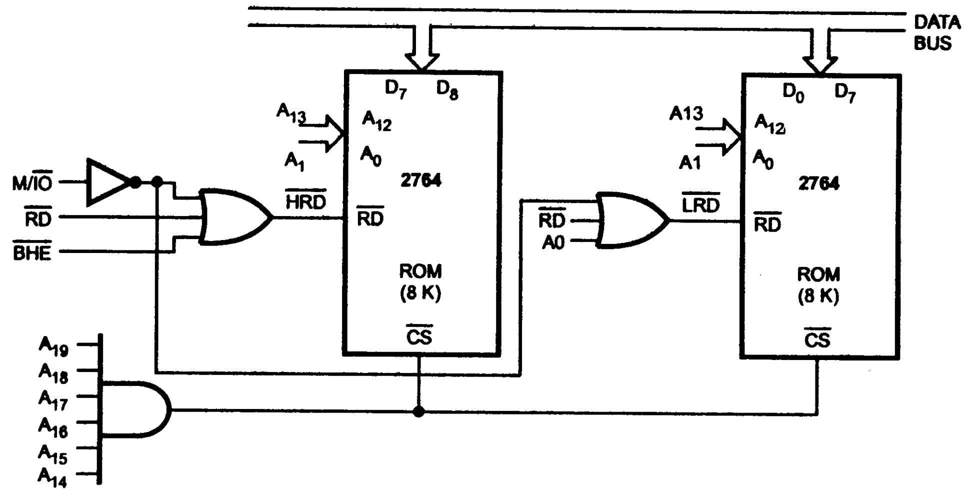


Fig. 8.3 Absolute decoding

2) Linear Decoding :

In small systems, hardware for the decoding logic can be eliminated by using only required number of addressing lines (not all). Other lines are simply ignored. This technique is referred as **linear decoding** or **partial decoding**. Fig 8.3 shows the addressing of 16 K RAM (6264) with linear decoding. Control signals $\overline{BHE}$ and A_0 are used to enable odd and even memory banks, respectively. The address line A_{19} is used to select the RAM chips. When A_{19} is low, chip is selected, otherwise it is disabled. The status of A_{14} to A_{18}

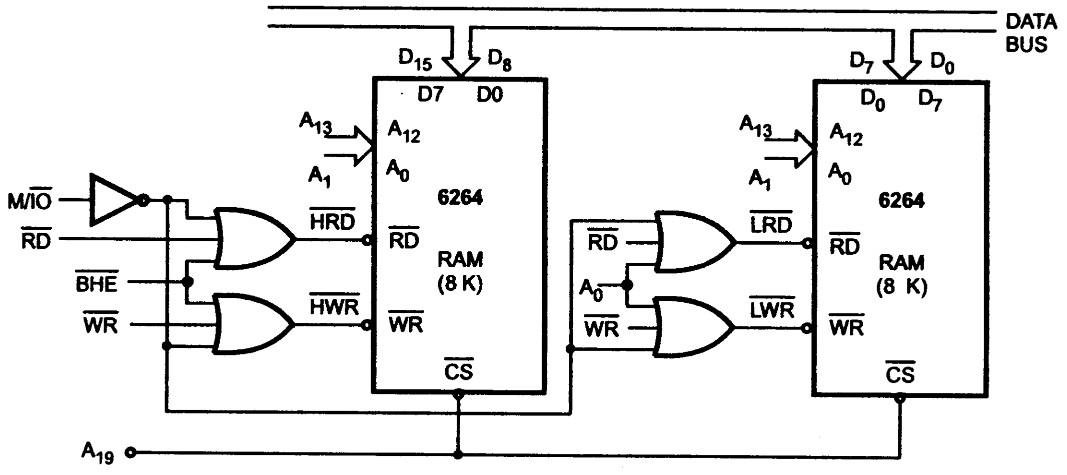


Fig. 8.4 Linear decoding

does not affect the chip selection logic. This gives you multiple addresses (shadow addresses). This technique reduces the cost of decoding circuit, but it has drawback of multiple addresses.

3) Block Decoding : In a microcomputer system the memory array is often consists of several blocks of memory chips. Each block of memory requires decoding circuit. To avoid separate decoding for each memory block special decoder IC is used to generate chip select signal for each block. Fig. 8.5 shows the block decoding technique using 74138, 3:8 decoder.

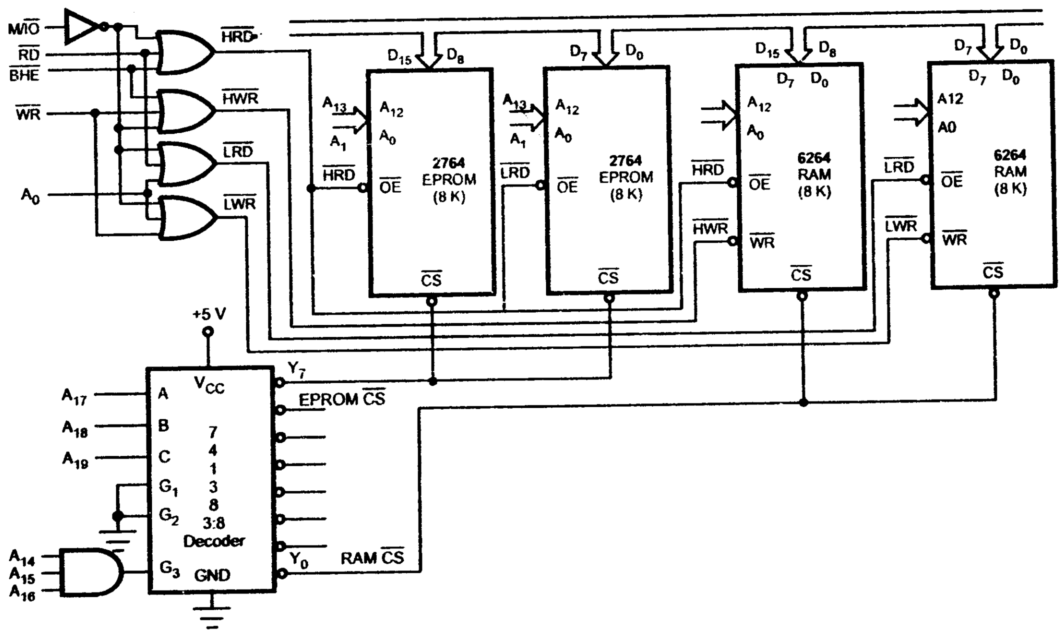


Fig. 8.5 Block decoding

8.3.2 Applications

Example 8.1 : *Design an 8088 based system with the following specifications*
i) 8088 in minimum mode ii) 8 kbyte EPROM iii) 8 kbyte RAM
Draw the complete schematic of the design indicating all the address selected, decoders used, etc.

Solution : As RAM and EPROM are 8 kbyte, 13 address lines are required from A_0 to A_{12}. The remaining address lines are used for address decoding circuitry. The address lines A_0 to A_7 and A_{16} to A_{19} are latched using IC 74373 (octal latch) and ALE signal. Fig. 8.6 shows the interface between 8088 and two memory chips.

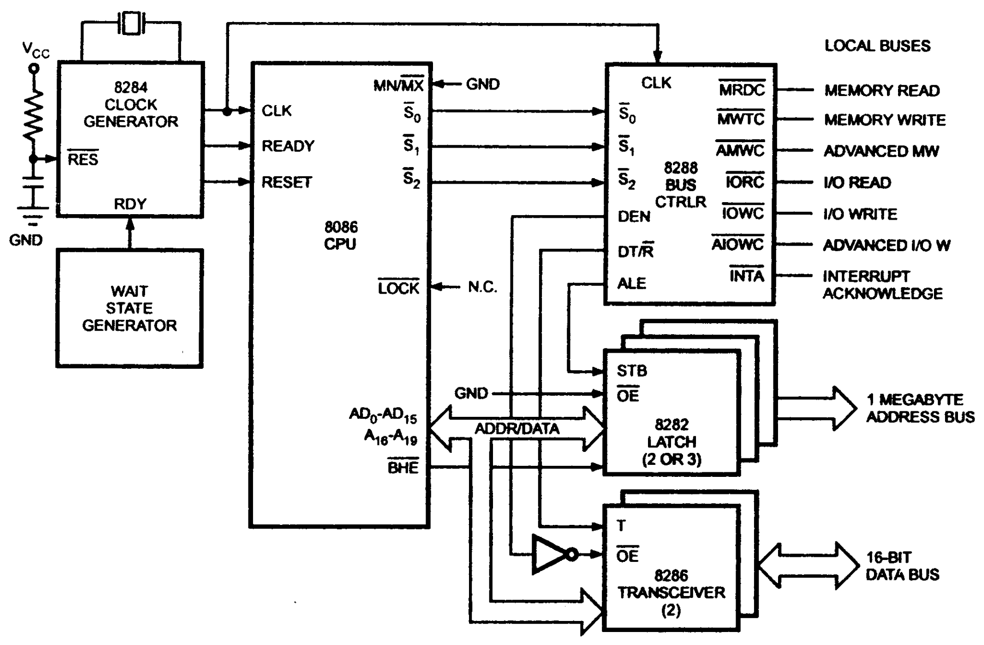


Fig. 7.15 Typical maximum mode configuration

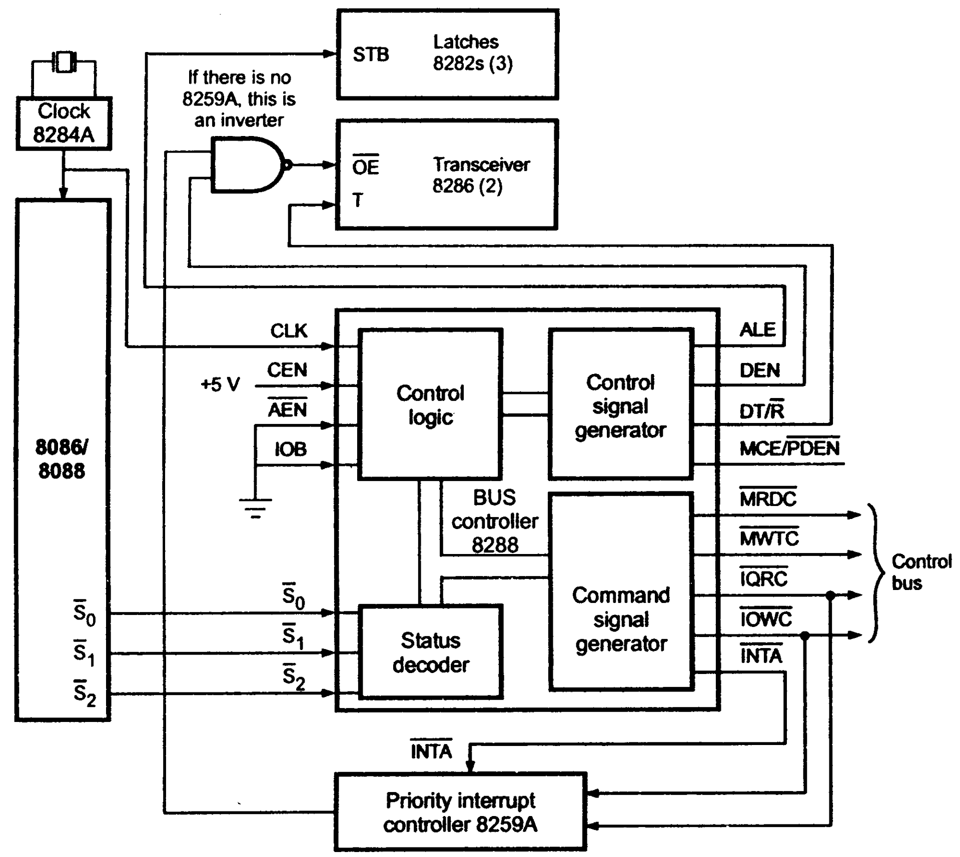


Fig. 7.16 8288 bus controller

$\overline{\text{MRDC}}$ (Memory Read Command) : It instructs the memory to put the contents of the addressed location on the data bus.

$\overline{\text{MWTC}}$ (Memory Write Command) : It instructs the memory to accept the data on the data bus and load the data into the addressed memory location.

$\overline{\text{IORC}}$ (I/O Read Command) : It instructs an I/O device to put the data contained in the addressed port on the data bus.

$\overline{\text{IOWC}}$ (I/O Write Command) : It instructs an I/O device to accept the data on the data bus and load the data into the addressed port.

MCE/$\overline{\text{PDEN}}$ (Master Cascade Enable/Peripheral Data Enable) : It controls the mode of operation of 8259. It selects cascade operation for 8259 (interrupt controller) if IOB signal is grounded and enables the I/O bus transceivers if IOB is tied high.

$\overline{\text{AEN}}$, IOB and CEN : These pins are used in multiprocessor system. With a single processor in the system, $\overline{\text{AEN}}$ and IOB are grounded and CEN is tied high. $\overline{\text{AEN}}$ causes the 8288 to enable the memory control signals. IOB (I/O bus mode) signal selects either the I/O bus mode or system bus mode operation. CEN (control enable) input enables the command output pins on the 8288.

$\overline{\text{AIOWC}}$/$\overline{\text{AMWC}}$ (Advance I/O Write Command/Advance Memory Write Command) : These signals are similar to IOWC and MWTC except that they are activated one clock pulse earlier. This gives slow interfaces an extra clock cycle to prepare to input the data.

7.7.2 Maximum Mode 8086 System

The Fig. 7.17 shows the typical maximum mode 8086 system. Here interfacing of memory and I/O devices are shown with the basic maximum mode configuration. The connections for memory and I/O devices are similar to that of minimum mode configuration. However, the generation of control signals from 8086 is done by external bus controller 8288.

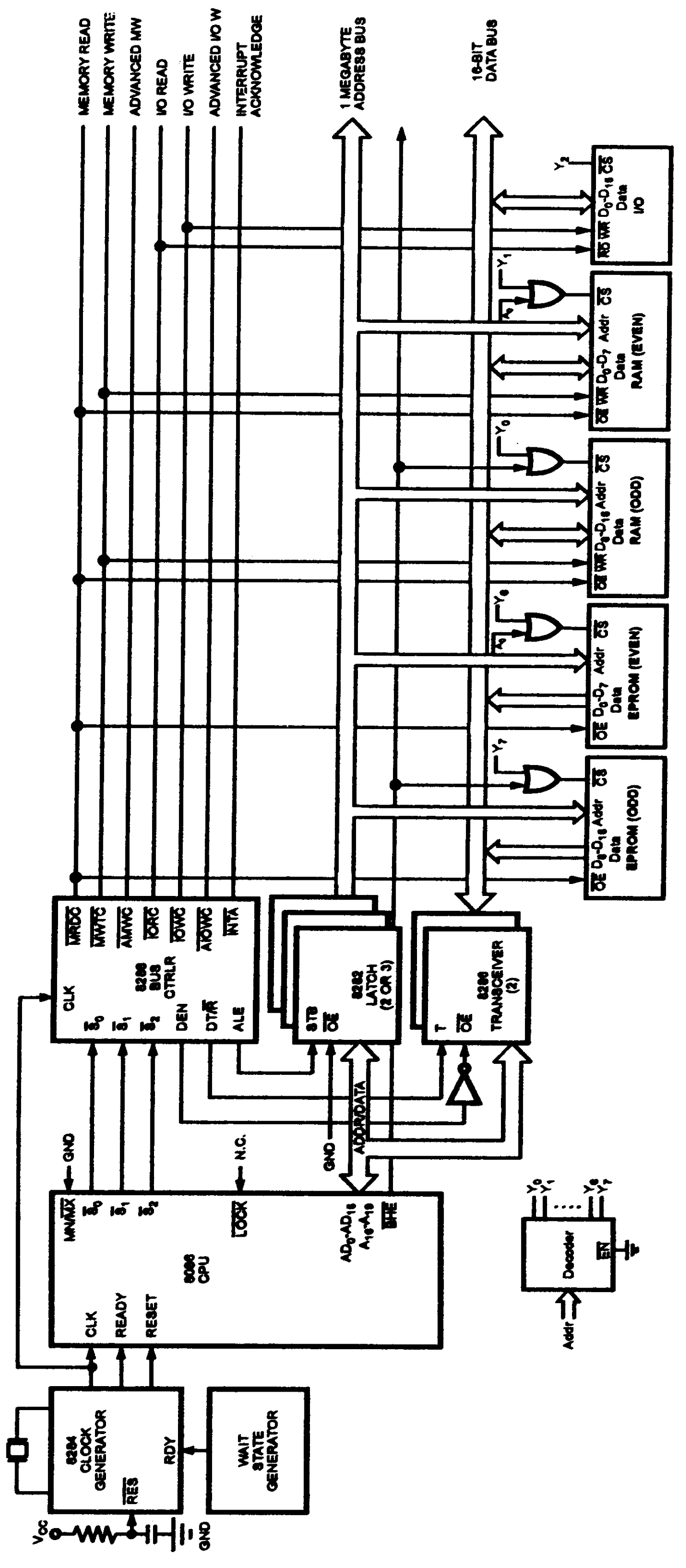
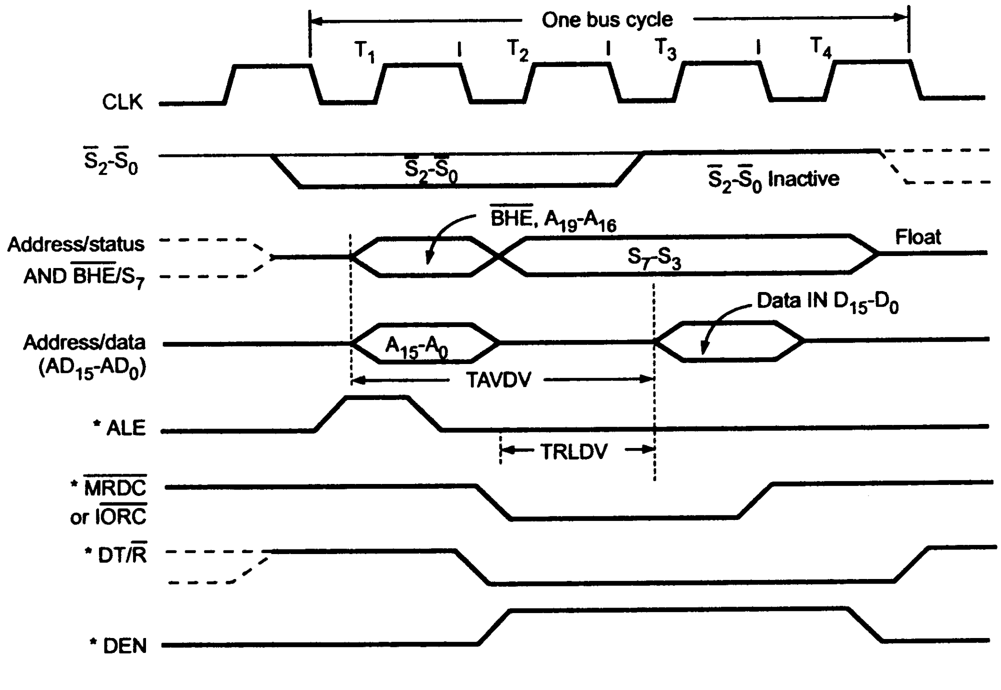


Fig. 7.17 Typical maximum mode 8086 system

7.7.3 Bus Timings for Maximum Mode

7.7.3.1 Timings for Read and Write Operations

The timing diagrams of input and output transfers are shown in the Fig. 7.18 (a) and (b) respectively.



One bus cycle
T_1
T_2
T_3
T_4
CLK
$\overline{S_2}$-$\overline{S_0}$
$\overline{S_2}$-$\overline{S_0}$
$\overline{S_2}$-$\overline{S_0}$ Inactive
$\overline{BHE}$, A_{19}-A_{16}
Address/status AND $\overline{BHE}/S_7$
S_7-S_3
Float
Data IN D_{15}-D_0
Address/data (AD_{15}-AD_0)
A_{15}-A_0
TAVDV
* ALE
TRLDV
* $\overline{MRDC}$ or $\overline{IORC}$
* DT/$\overline{R}$
* DEN

Fig. 7.18 (a) Input (read operation)

These are explained in steps.

1. S_0, $\overline{S_1}$, $\overline{S_2}$ are set at the beginning of bus cycle. On detecting the change on passive state $\overline{S_0} = \overline{S_1} = \overline{S_2} = 1$, the 8288 bus controller will output a pulse on its ALE and apply a required signal to its DT/$\overline{R}$ pin during T_1.
2. In T_2, 8288 will set DEN = 1 thus enabling transceiver. For an input, 8288 it will activates $\overline{MRDC}$ or $\overline{IORC}$. These signals are activated until T_4. For an output, the $\overline{AMWC}$ or $\overline{AIOWC}$ is activated from T_2 to T_4 and $\overline{MWTC}$ or $\overline{IOWC}$ is activated from T_3 to T_4.
3. The status bits $\overline{S_0}$ to $\overline{S_2}$ remain active until T_3, and become passive during T_3 and T_4.
4. If ready input is not activated before T_3, wait state will be inserted between T_3 and T_4.

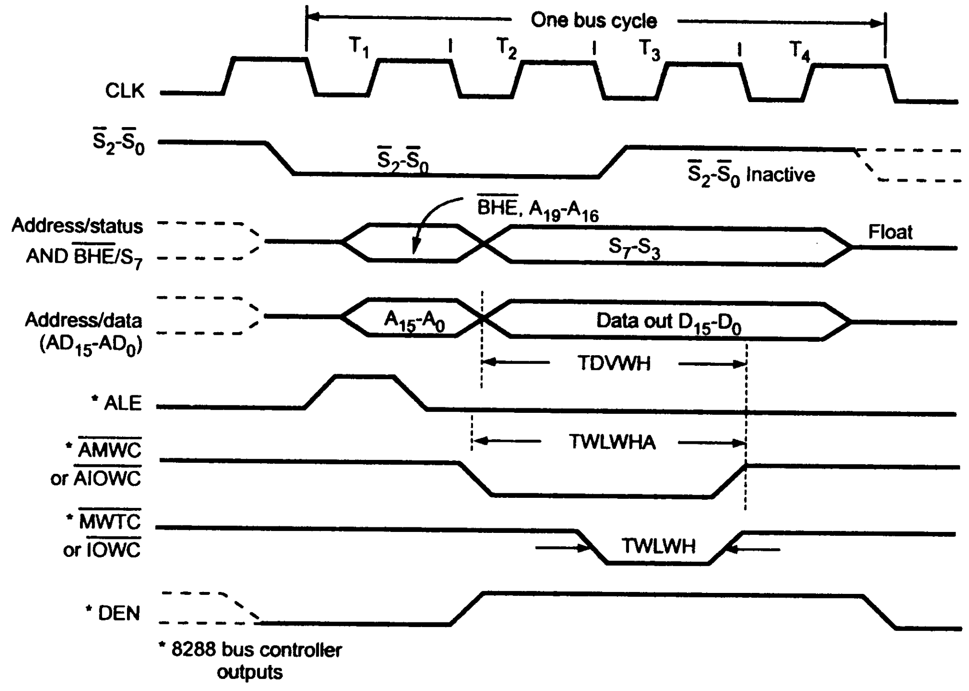

One bus cycle
T_1
T_2
T_3
T_4
CLK
$\overline{S_2}$-$\overline{S_0}$
$\overline{S_2}$-$\overline{S_0}$
$\overline{S_2}$-$\overline{S_0}$ Inactive
Address/status AND $\overline{BHE}/S_7$
$\overline{BHE}$, A_{19}-A_{16}
S_7-S_3
Float
Address/data (AD_{15}-AD_0)
A_{15}-A_0
Data out D_{15}-D_0
TDVWH
* ALE
TWLWHA
* $\overline{AMWC}$ or $\overline{AIOWC}$
* $\overline{MWTC}$ or $\overline{IOWC}$
TWLWH
* DEN
* 8288 bus controller outputs

Fig. 7.18 (b) Output (write operation)

7.7.3.2 Timings for $\overline{RQ}/\overline{GT}$ Signals

The Fig. 7.19 shows the timing diagrams for bus request and bus grant pins of 8086. A bus request, grant and release is accomplished by a sequence of three pulses. The $\overline{RQ}/\overline{GT}$ pins are sampled at the rising edge of each clock pulse and if a request is detected the 8086 will apply a grant pulse to the $\overline{RQ}/\overline{GT}$ if the following conditions are met.

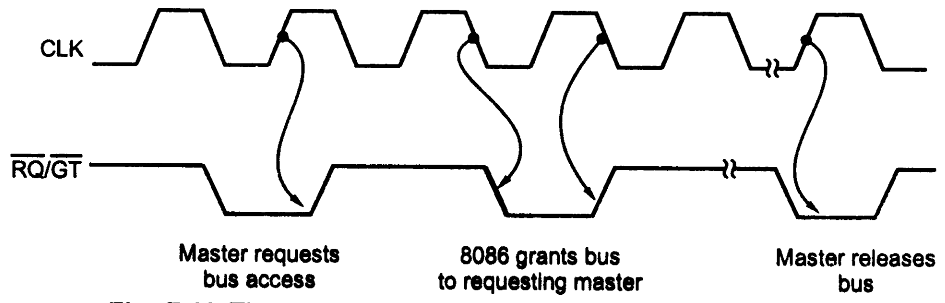

CLK
$\overline{RQ}/\overline{GT}$
Master requests bus access
8086 grants bus to requesting master
Master releases bus

Fig. 7.19 Timings for bus request and bus grant signals

1. The previous bus transfer was not the low byte of a word to or from an odd address if the CPU is an 8086. For an 8088, regardless of the address alignment, the grant signal will not be sent until the second byte of a word reference is accessed.
2. The first pulse of an interrupt acknowledgement did not occur during the previous bus cycle.
3. An instruction with a LOCK prefix is not being executed.

If condition 1 and 2 is not met, then the grant will not be given until the next bus cycle, and if condition 3 is not met, the grant will wait until the locked instruction is completed.

After activation of grant pulse, requesting master takes the control of buses. This master may control the buses for only one bus cycle or for several bus cycles. When it is ready to relinquish the buses it will send the processor release pulse over the same line that it made its request.

7.8 Minimum Mode and Maximum Mode 8088 Systems

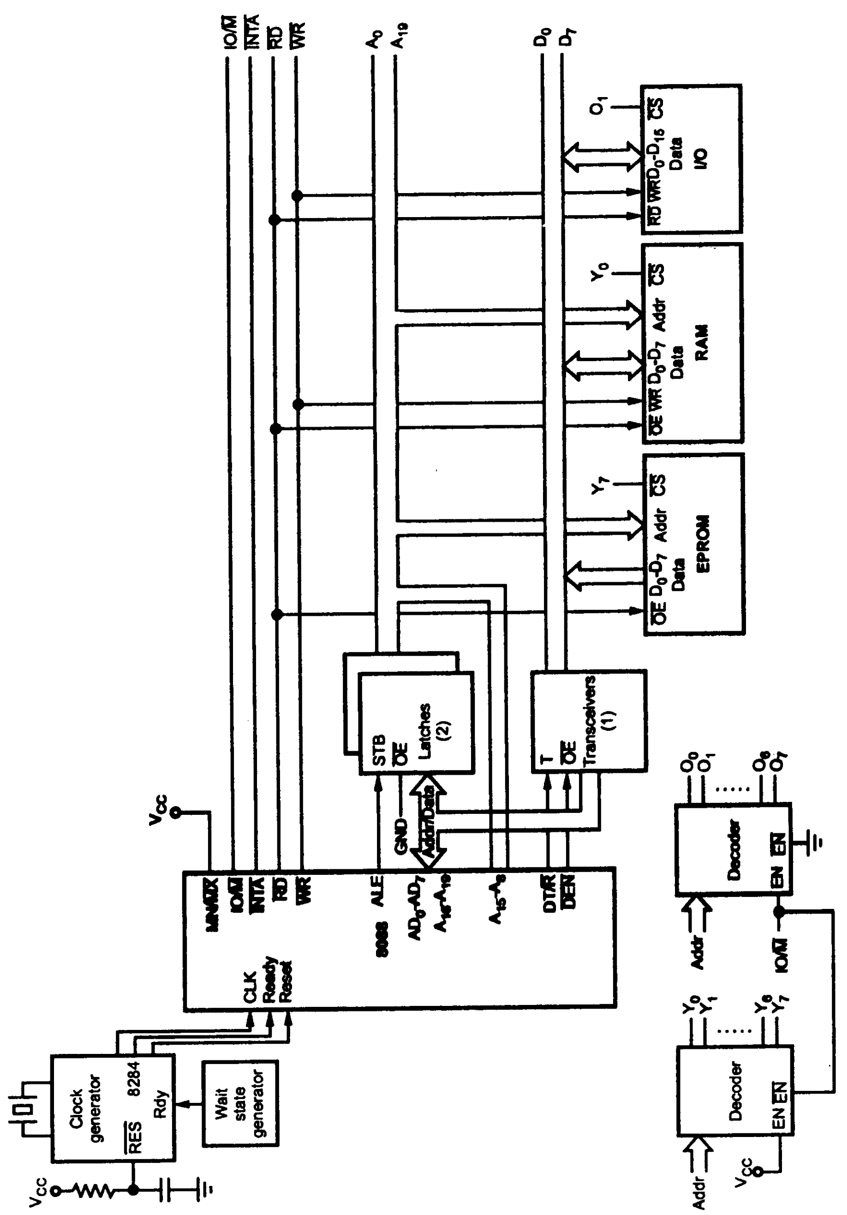


Fig. 7.20 (a) Minimum mode 8088 system

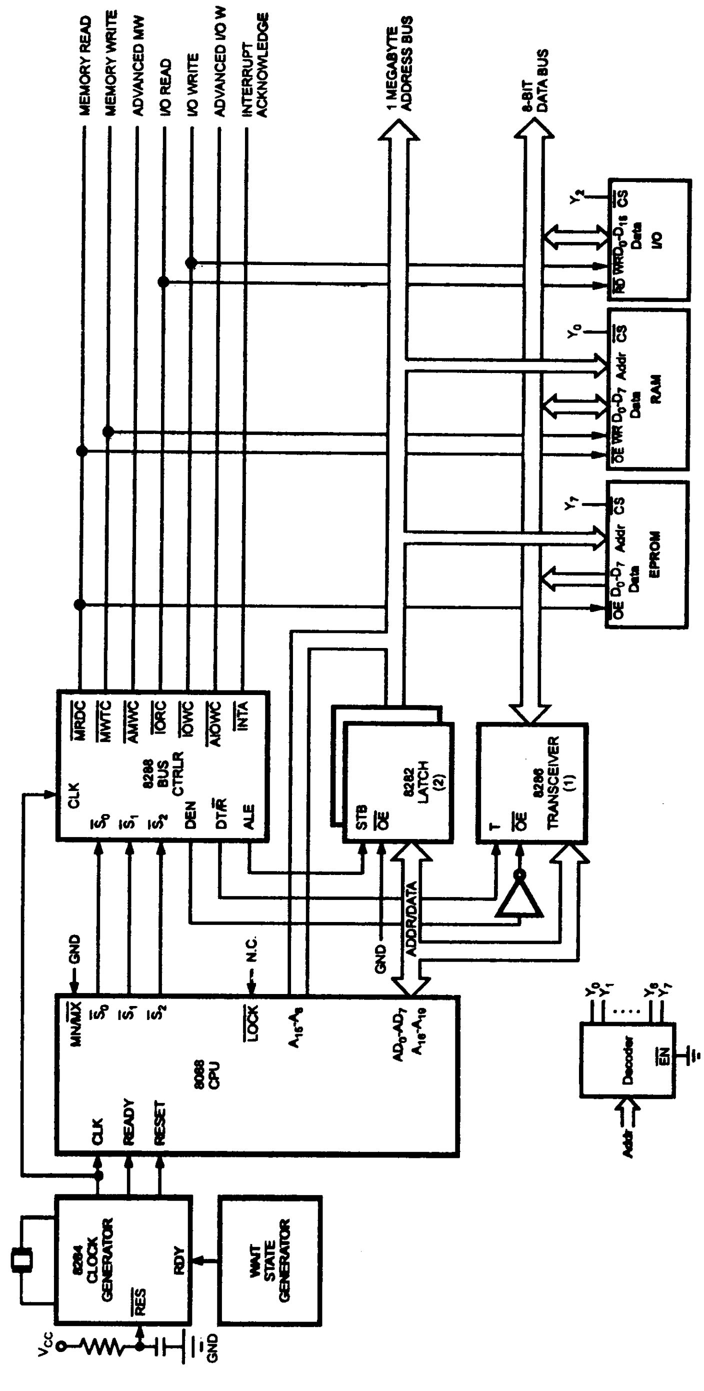


Fig. 7.20 (b) Maximum mode 8088 system

The Fig. 7.20 shows the minimum mode and maximum mode 8088 systems. Like 8086, 8088 has 20 address lines and hence its addressing capability is 1 Mbyte. However, since

8088 has 8-bit data bus, it is not to have separate odd and even memory banks like 8086. Rest of the interfacing is same as that of 8086 systems.

7.9 Typical 8088 System Timing Diagram

The Fig. 7.21 shows the typical 8088 system timing diagram. As shown in the Fig. 7.21, the address/data bus of 8088 system is split into three parts : 1. The lower 8-bit multiplexed address and data bus (AD_7-AD_0) 2. Middle 8-bit address bus (A_{15}-A_8) 3. The upper 4-bit multiplexed address and status bus (A_{19}/S_6 - A_{16}/S_3). Like 8086, each bus cycle of 8088 contains at least four clock cycles : T_1, T_2, T_3 and T_4. During T_1, 8088 sends address on the address bus and activates ALE signal. The ALE signal is used to activate latches and thus to latch the address. The data transfer takes place during T_3 and T_4.

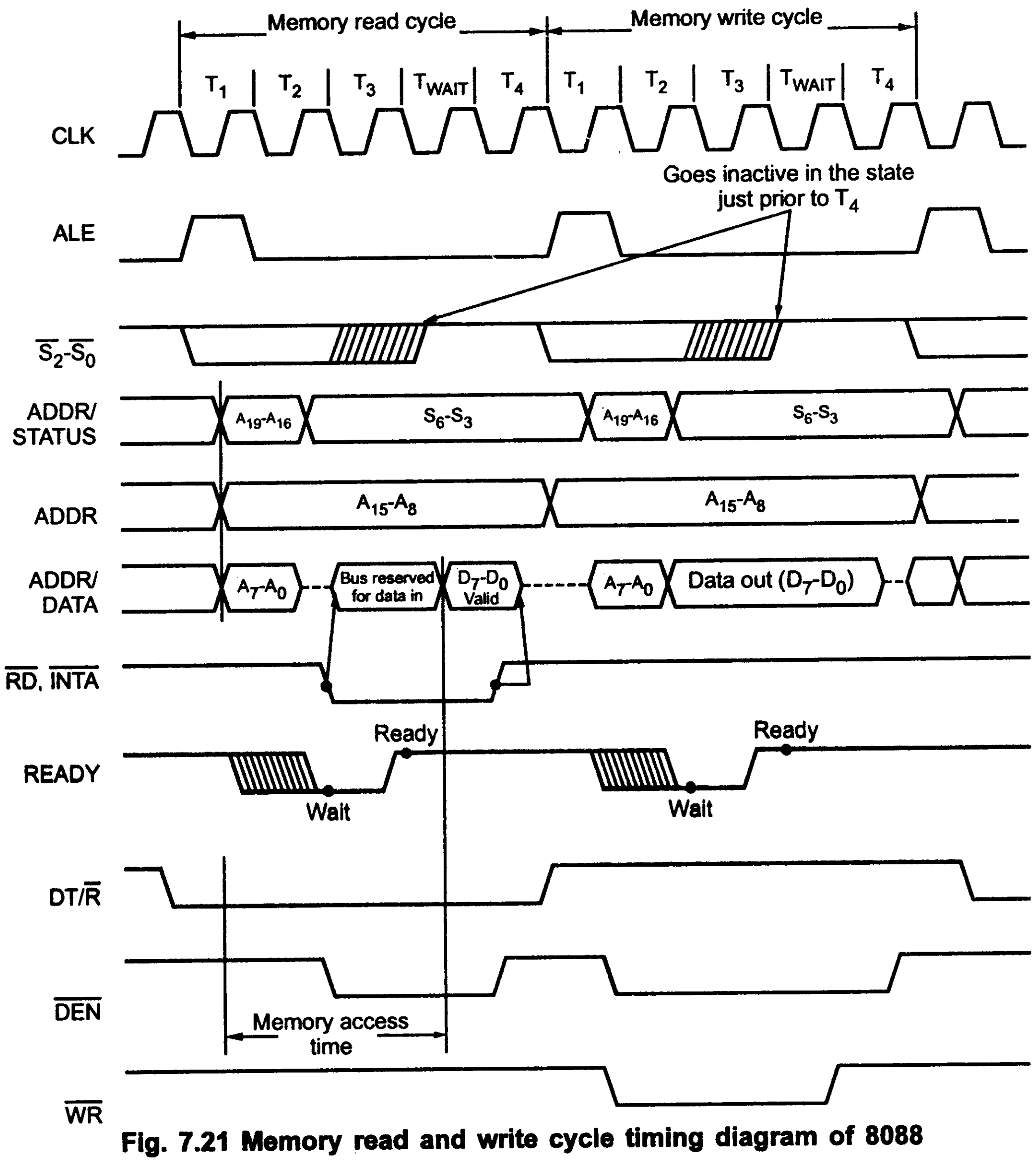


Fig. 7.21 Memory read and write cycle timing diagram of 8088

The time interval T_2 is used primarily for changing the direction of the bus during read operation. Ready signal is sampled during T_3. If Ready signal is not active, the wait state is inserted in the bus cycle. $\overline{RD}$ and $\overline{WR}$ signals are activated in their respective machine cycles. If a system is large enough to need data bus buffers, the 8088 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 8088 $\overline{DEN}$ signal will enable the buffers at the appropriate time in the machine cycle, as shown in the Fig. 7.21.

Review Questions

1. *Explain the function of following pins in 8086.*
   i) NMI ii) MN/$\overline{MX}$ iii) $\overline{TEST}$ iv) $\overline{BHE}$ v) DT/$\overline{R}$ vi) $\overline{DEN}$ vii) QS_0-QS_1.
2. *Explain the maximum mode signals of 8086.*
3. *Explain the minimum mode signals 8086.*
4. *With the help of block diagram explain memory interfacing with 8086 and explain why two bus cycles are required to access odd address word ?*
5. *Draw and explain the memory map for 8086.*
6. *Explain the I/O addressing capabilities of 8086.*
7. *Draw and explain the I/O map of 8086.*
8. *Explain the general bus operation of 8086 with the help of timing diagram.*
9. *Explain the purpose of Ready, $\overline{DEN}$ and DT/$\overline{R}$ signals.*
10. *With the help of block schematic diagrams explain the operation of 8284 clock generator and 8286 transceiver.*
11. *Sketch block diagram showing basic 8086 minimum mode system. Explain functions of 8282 latches and 8286 transceiver.*
12. *Define bus cycle, and explain the minimum mode read and write bus cycle with proper timing diagram.*
13. *Explain the HOLD response sequence in the minimum mode of 8086 with the help of timing diagram.*
14. *Draw and explain a block diagram showing 8086 in maximum mode configuration.*
15. *Draw and explain the timing diagrams of input and output transfers of 8086 in maximum mode.*
16. *Indicate the signals which are different when 8086 in minimum mode and in maximum mode.*
17. *Explain the operation of bus request and bus grant signal with the help of timing diagram.*
18. *Draw and explain the minimum mode system with 8088 processor.*
19. *Draw and explain the maximum mode system with 8088 processor.*
20. *Draw and explain the typical 8088 system timing diagram.*

❑❑❑

8

Memory and I/O Interfacing

Memory is an integral part of a microprocessor system, and in this chapter, we will discuss how to interface a memory device with the microprocessor. The memory interfacing circuit is used to access memory quite frequently to read instruction codes and data stored in memory. This read/write operations are monitored by control signals. The microprocessor activates these signals when it wants to read from and write into memory. In this chapter we will see memory structure and its requirements, concepts in memory interfacing and interfacing examples. The later part of this chapter explains the basic concept in I/O interfacing.

8.1 Terminology and Operations

Memories are made up of registers. Each register in the memory is one storage location. Each location is identified by an **address**. The number of storage locations can vary from a few in some memories to hundreds of thousand in others. Each location can accommodate one or more bits. Generally, the total number of bits that a memory can store is its **capacity**. Most of the types the capacity is specified interms of bytes (group of eight bits)

Each register consists of storage elements (flip-flops or capacitors in semiconductor memories and magnetic domain in magnetic storage), each of which stores one bit of data. A storage element is called a cell.

The data stored in a memory by a process called **writing** and are retrieved from the memory by a process called **reading**. Fig. 8.1 illustrates in a very simplified way the concept of write, read, address and storage capacity for a generalized memory.

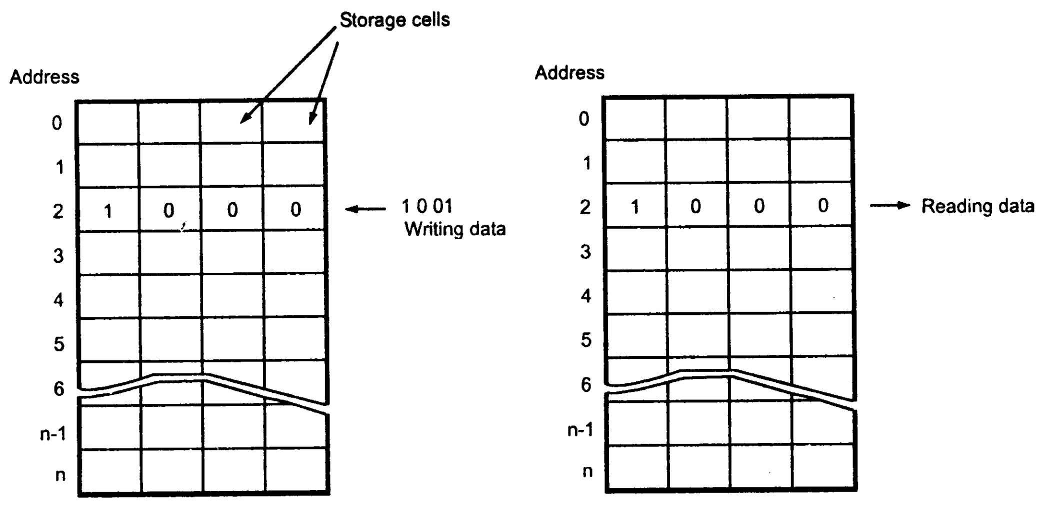
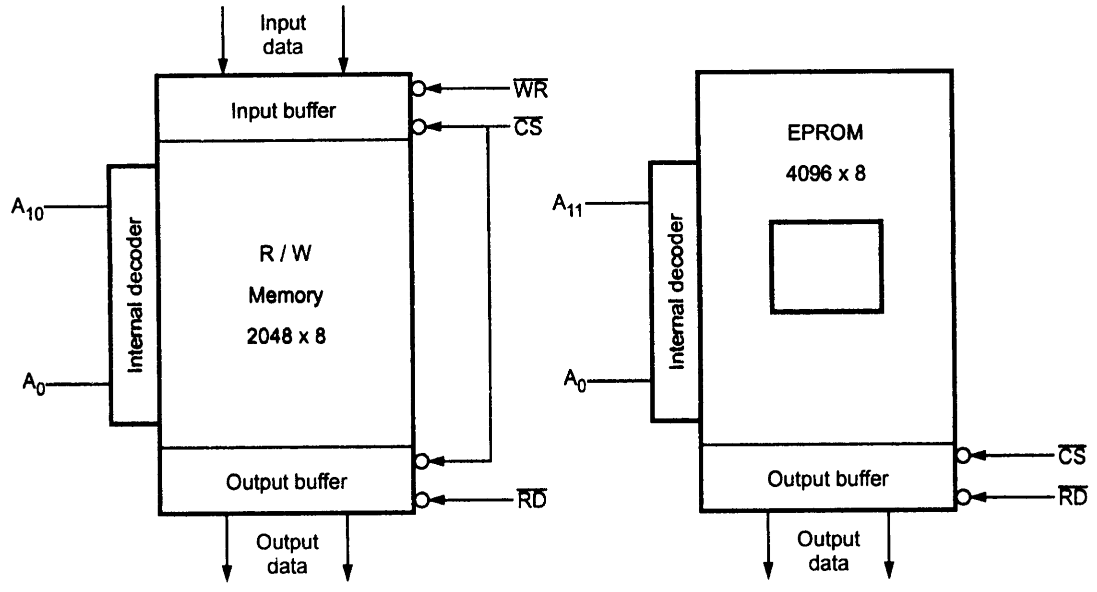


Fig. 8.1

8.2 Memory Structure and its Requirements

Read/write memories consist of an array of registers, in which each register has unique address. The size of the memory is N x M as shown in Fig. 8.2 (a) where N is the number of registers and M is the word length, in number of bits.



(a) Logic diagram for RAM (b) Logic diagram for EPROM

Fig. 8.2

Example 1 : If memory is having 12 address lines and 8 data lines, then

Number of registers/memory locations = $2^N = 2^{12}$

= 4096

Word length = M bit

= 8-bit

Example 2 : If memory has 8192 memory locations, then it has 13 address lines.

The Table 8.1 summarizes the memory capacity and address lines required for memory interfacing.

Memory Capacity	Address Lines Required
1 K = 1024 memory locations	10
2 K = 2048 memory locations	11
4 K = 4096 memory locations	12
8 K = 8192 memory locations	13
16 K = 16384 memory locations	14
32 K = 32768 memory locations	15
64 K = 65536 memory locations	16

Table 8.1

As shown in the Fig. 8.2 (a) memory chip has 12 address lines A_0-A_{11}, one chip select ($\overline{CS}$), and two control lines, read ($\overline{RD}$) to enable output buffer and write ($\overline{WR}$) to enable the input buffer. The internal decoder is used to decode the address lines. Fig. 8.2 (b) shows the logic diagram of a typical EPROM (Erasable Programmable Read-Only Memory) with 4096 (4 K) registers. It has 12 address lines A_0-A_{11}, one chip select ($\overline{CS}$), one Read control signal. Since EPROM is a read only memory, it does not require the ($\overline{WR}$) signal.

8.3 Basic Concepts in Memory Interfacing

For interfacing memory devices to microprocessor 8086/88 following important points are to be kept in mind.

1. Microprocessor 8086/88 can access 1 Mbytes memory since address bus is 20-bit. But it is not always necessary to use full 1 Mbytes address space. The total memory size depends upon the application.
2. Generally EPROM (or EPROMs) is used as a program memory and RAM (or RAMs) as a data memory. When both, EPROM and RAM are used, the total address space 1 Mbytes is shared by them.

3. The individual capacities of program memory and data memory depend on the application.
4. It is not always necessary to select 1 EPROM and 1 RAM. We can have multiple EPROMs and multiple RAMs as per the requirement of application.
5. We can place EPROM/RAM anywhere in full 1 Mbytes address space. But program memory (EPROM) should be located at last memory page so that the starting address FFFF0H will lie within the program memory range. To provide facility to set addresses in the interrupt vector table we must provide RAM at page 0 of memory. So that the interrupt vector table lie with the read/write memory range.
6. It is not always necessary to locate EPROM and RAM in consecutive memory addresses. However, it is advised to do that.
7. While interfacing memory to 8086 we have to provide odd and even banks of memory. Even bank is selected when $A_0 = 0$ and odd bank is selected when $\overline{BHE} = 0$. Odd and even banks are not required for interfacing memory to 8088.

The memory interfacing requires to :

- Select the chip
- Identify the register
- Enable the appropriate buffer.

Microprocessor system includes memory devices and I/O devices. It is important to note that microprocessor can communicate (read/write) with only one device at a time, since the data, address and control buses are common for all the devices. In order to communicate with memory or I/O devices, it is necessary to decode the address from the microprocessor. Due to this each device (memory or I/O) can be accessed independently.

8.3.1 Address Decoding Techniques

1) Absolute Decoding

2) Linear Decoding

3) Block Decoding.

1) Absolute Decoding : In absolute decoding technique the memory chip is selected only for the specified logic level on the address lines; no other logic levels can select the chip. Fig 8.3 shows the memory interface with absolute decoding. Two 8 K EPROMs (2764) are used to provide even and odd memory banks. Control signals $\overline{BHE}$ and A_0 are used to enable outputs of odd and even memory banks respectively. As each memory chip has 8 K memory locations, thirteen address lines are required to address each locations, independently. All remaining address lines are used to generate an unique chip select signal. This addressing technique is normally used in large memory systems.

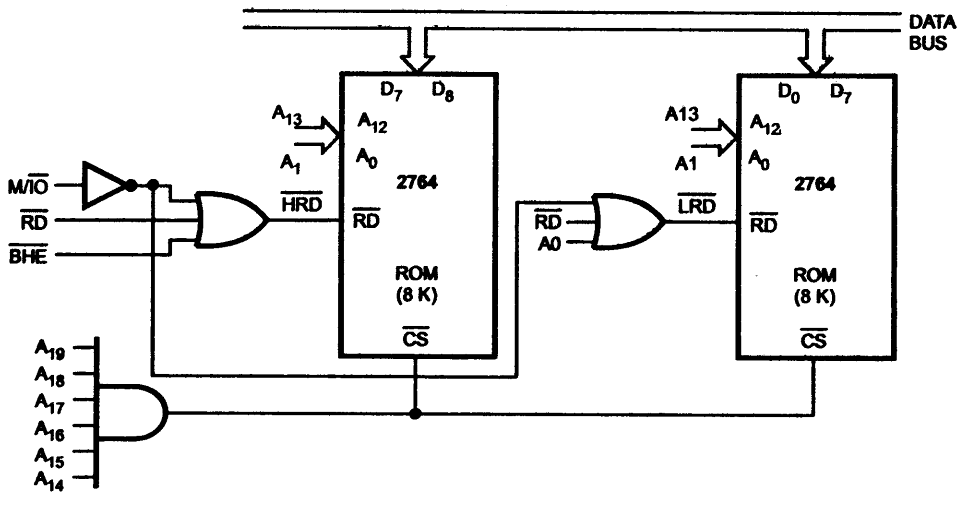


Fig. 8.3 Absolute decoding

2) Linear Decoding :

In small systems, hardware for the decoding logic can be eliminated by using only required number of addressing lines (not all). Other lines are simply ignored. This technique is referred as **linear decoding or partial decoding.** Fig 8.3 shows the addressing of 16 K RAM (6264) with linear decoding. Control signals $\overline{BHE}$ and A_0 are used to enable odd and even memory banks, respectively. The address line A_{19} is used to select the RAM chips. When A_{19} is low, chip is selected, otherwise it is disabled. The status of A_{14} to A_{18}

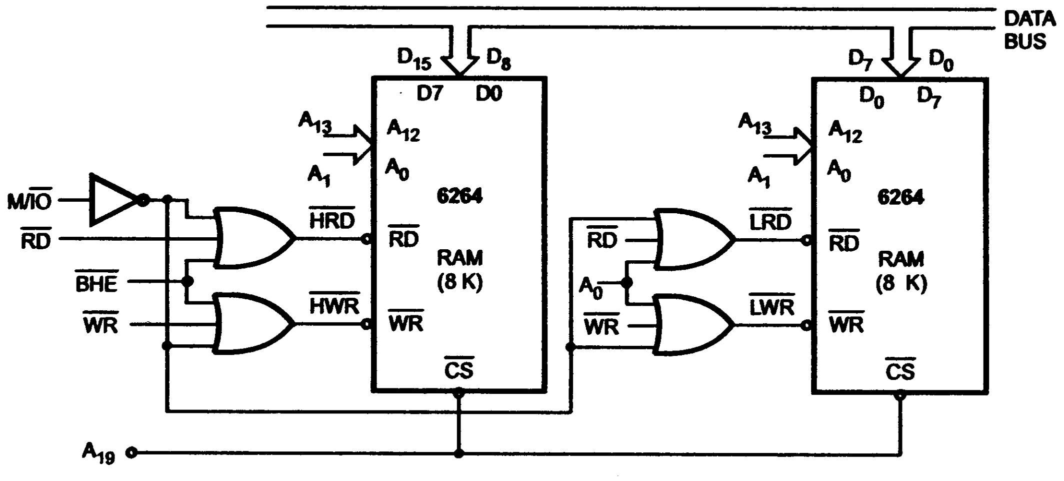


Fig. 8.4 Linear decoding

does not affect the chip selection logic. This gives you multiple addresses (shadow addresses). This technique reduces the cost of decoding circuit, but it has drawback of multiple addresses.

3) Block Decoding : In a microcomputer system the memory array is often consists of several blocks of memory chips. Each block of memory requires decoding circuit. To avoid separate decoding for each memory block special decoder IC is used to generate chip select signal for each block. Fig. 8.5 shows the block decoding technique using 74138, 3:8 decoder.

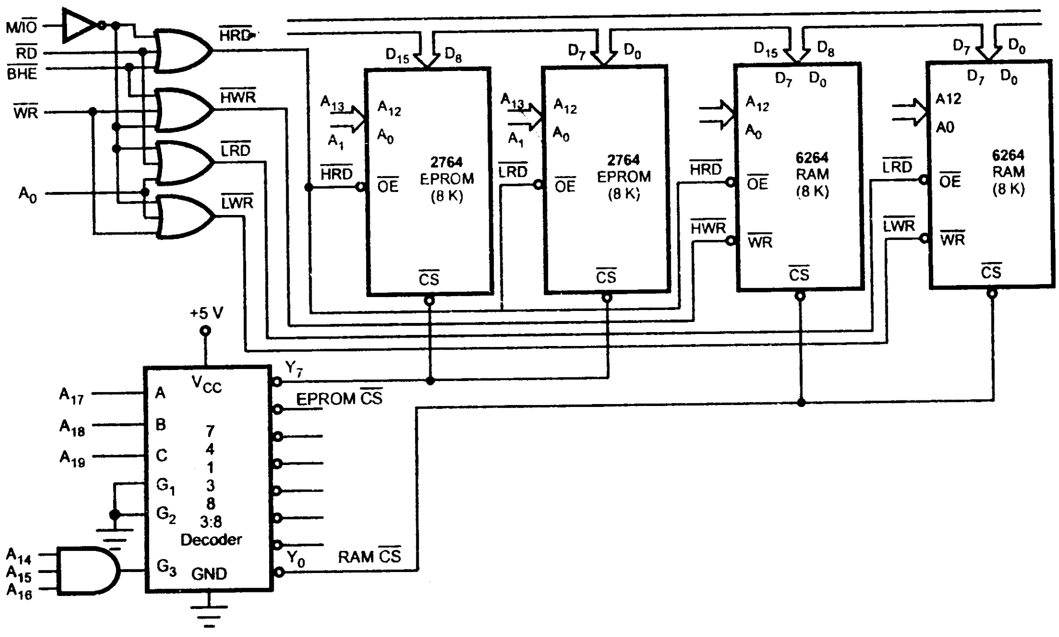


Fig. 8.5 Block decoding

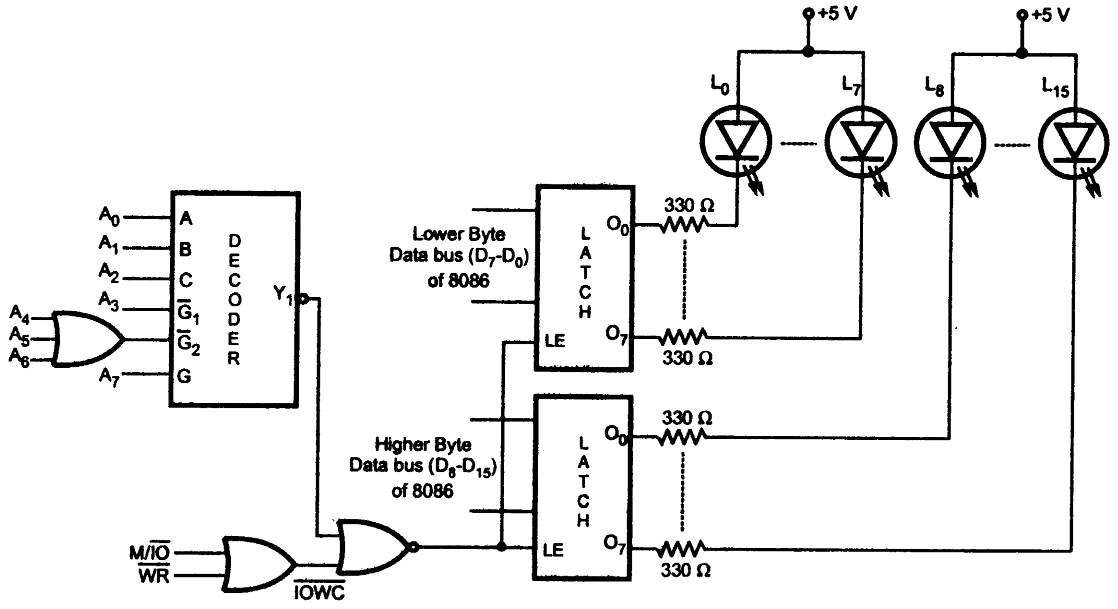
8.3.2 Applications

Example 8.1 : *Design an 8088 based system with the following specifications*
i) 8088 in minimum mode ii) 8 kbyte EPROM iii) 8 kbyte RAM
Draw the complete schematic of the design indicating all the address selected, decoders used, etc.

Solution : As RAM and EPROM are 8 kbyte, 13 address lines are required from A_0 to A_{12}. The remaining address lines are used for address decoding circuitry. The address lines A_0 to A_7 and A_{16} to A_{19} are latched using IC 74373 (octal latch) and ALE signal. Fig. 8.6 shows the interface between 8088 and two memory chips.

8.4.1.7 Interfacing 16-bit Output Device

Fig. 8.20 shows the circuit diagram to interface 16-bit output port (latch) which is used to send the signal for glowing 16 LEDs. LED will glow when output pin status is low. The IC 74LS138 and OR gate are used to generate device select signal. The latch enable signal is active high. So NOR gate is used to generate latch enable signal, which goes high when Y_2 and $\overline{IOWC}$ both are low.



Decoder select					Decoder input			
A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	
1	0	0	0	0	0	1	0	= 82 H

Fig. 8.20 Circuit diagram to interface 16-bit output port

The following program glows the LEDs L_{14}, L_{11}, L_8, L_5, L_2 and L_0.

L_{15}	L_{14}	L_{13}	L_{12}	L_{11}	L_{10}	L_9	L_8	L_7	L_6	L_5	L_4	L_3	L_2	L_1	L_0	
1	0	1	1	0	1	1	0	1	1	0	1	1	0	1	0	= B6DAH

The code (data) B6DAH must be sent on the latch to glow LEDs L_{14}, L_{11}, L_8, L_5, L_2 and L_0.

Program:

```
MOV AX,0B6DAH      ; Loads the data in the accumulator.
OUT 82H,AX         ; sends the data on the latch.
```

Note : In the previous examples we have 8-bit port addresses so only lower eight address lines are used for address decoding. For 16-bit port addresses we have to decode lower sixteen address lines to generate device select signal. The remaining part of the circuit is same.

The Fig. 8.21 shows the combined circuit for I/O interfacing. For this circuit the address of input port is 80H and address of output port is 82H. The following program displays the status of switches on the LEDs.

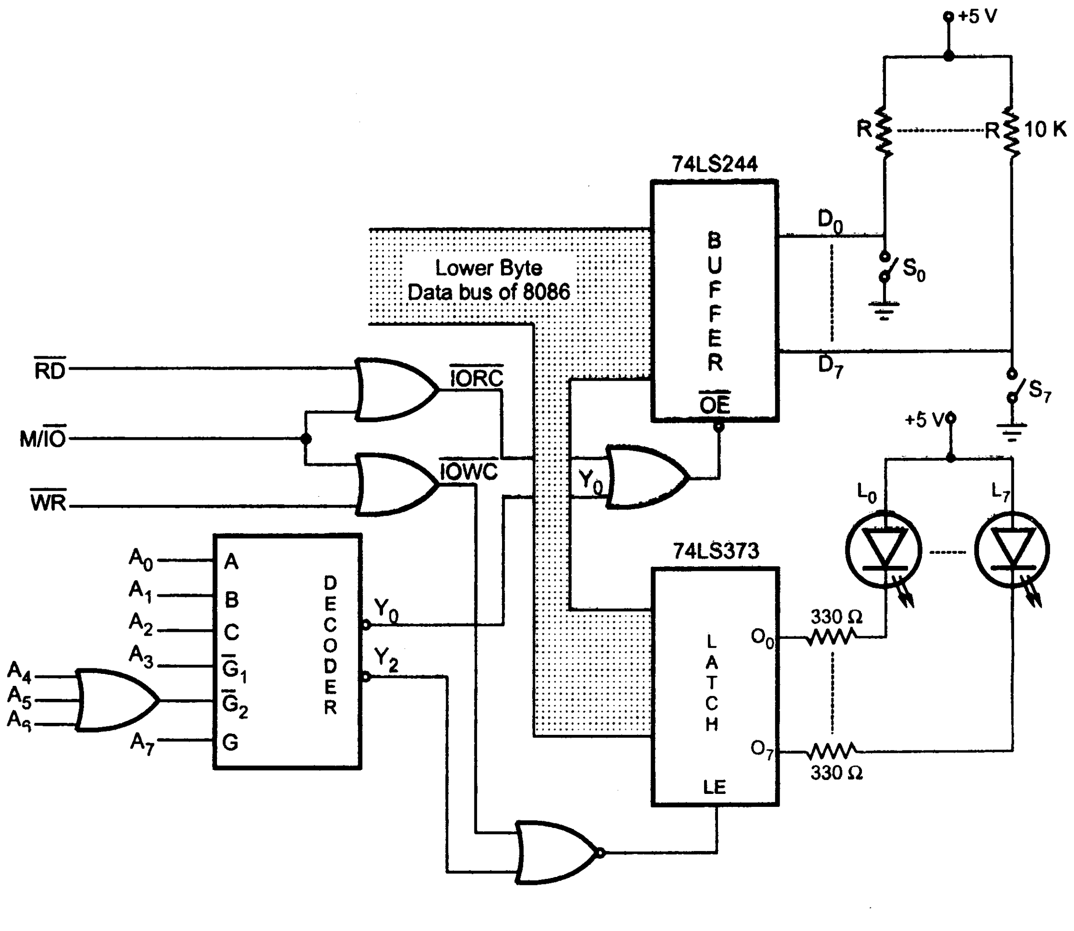


Fig. 8.21 I/O interfacing using I/O mapped I/O

Program :

```
IN   80H        ; Read status of all switches.
OUT  82H        ; send status on the output port.
```

Example :

Refer Fig. 8.21 and write a program that will check the switch1 status and do accordingly.

i) $S_1 = 0$: Blink Lower nibble LEDs.

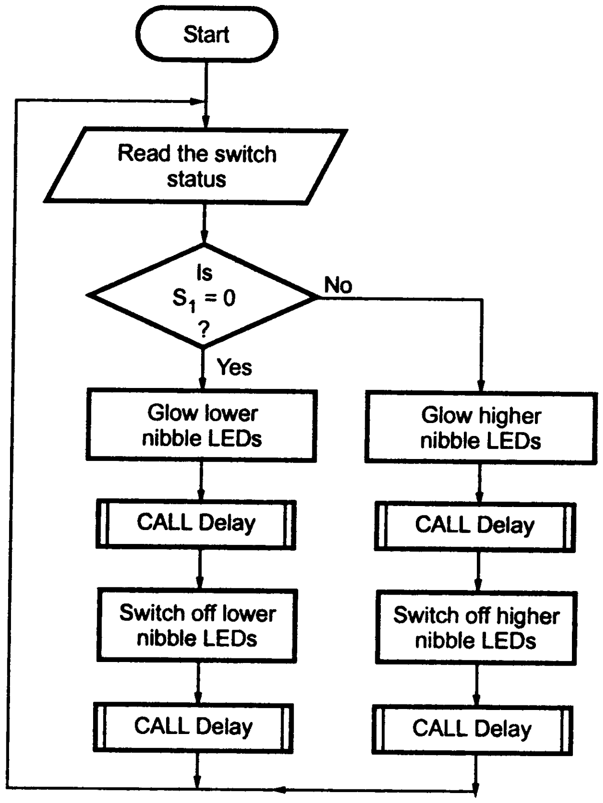
ii) $S_1 = 1$: Blink Higher nibble LEDs.

Assume delay routine is available.

Solution : Input port address = 80H

Output port address = 82H

Flow chart :



Source Program :

```
START:   IN AL,80H      ; Read status of switches
         AND AL,01H     ; Masks Bit 1 to Bit 7
         JNZ HIGHER     ; If swl is not pressed goto blink
                        ; higher nibble
         MOV AL,0F0H    ; Load bit pattern to glow lower nibble LEDs
         OUT 82H,AL     ; Send it to output port
         CALL Delay     ; Call delay subroutine
         MOV AL,0FFH    ; Load bit pattern to switch off all LEDs
         OUT 82H,AL     ; Send it to output port
```

```
          CALL Delay     ; Call delay subroutine
          JMP  START     ; JUMP to START
HIGHER:   MOV AL,0FH     ; Load bit pattern to glow higher
                         ; nibble LEDs
          OUT 82H,AL     ; Send it to output port
          CALL Delay     ; Call delay subroutine
          MOV AL,0FFH    ; Load bit pattern to switch off all LEDs
          OUT 82H,AL     ; Send it to output port
          CALL Delay     ; Call delay subroutine
          JMP START      ; JUMP to START
```

8.4.1.8 I/O Interfacing with 16-bit Port Address

As mentioned earlier, for 16-bit port address we have to decode lower sixteen address lines to generate device select signal. Fig. 8.22 shows the interfacing of 8-bit input port (buffer) and 8-bit output port (latch) having 16-bit address. The address of input port is 0080H and address of output port is 0082H.

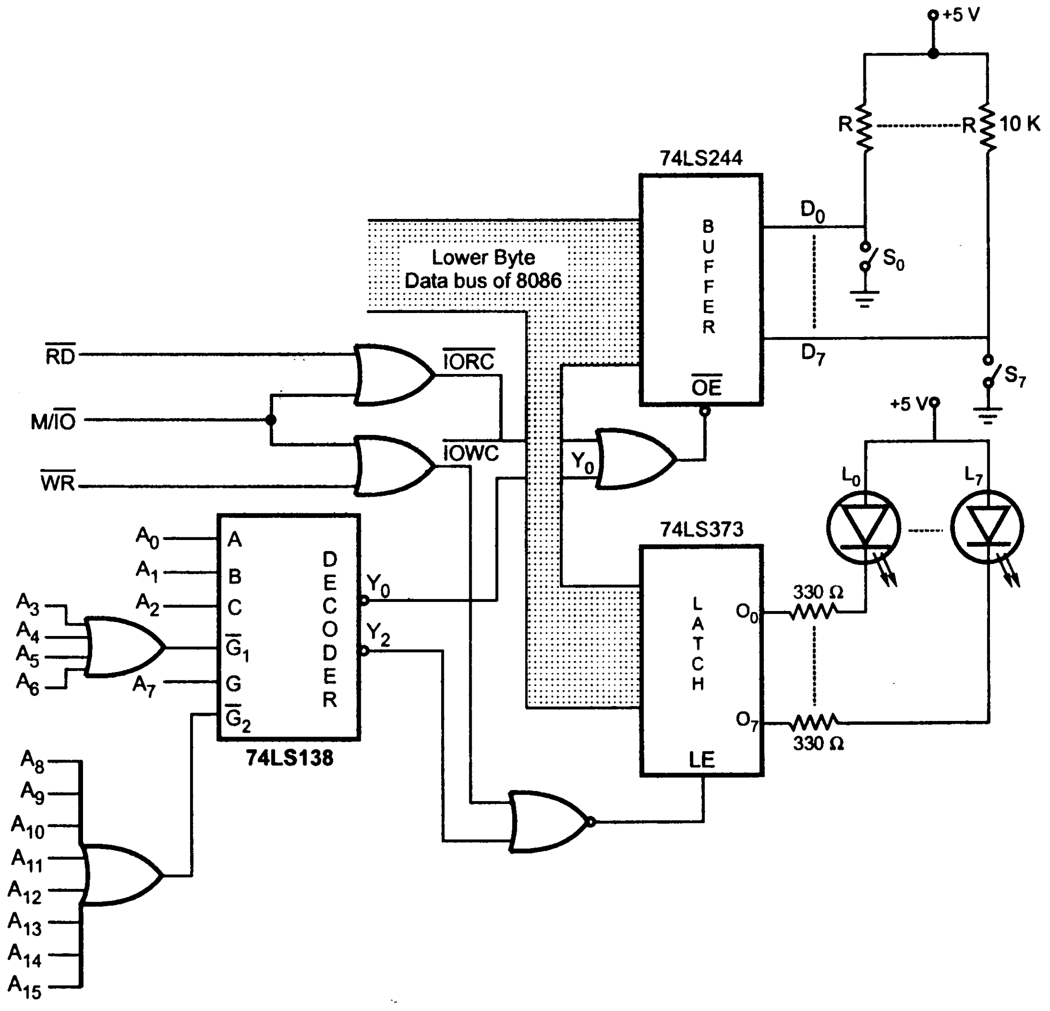


Fig. 8.22 Interfacing of 8-bit input port (buffer) and 8-bit output port (latch)

Example :

Refer Fig. 8.9 and write a program to display the status of switches on the LEDs.

Program :

```
MOV DX,0080H ; Load 16-bit address of the input port in DX
IN AL,DX     ; Read byte from input port in AL
MOV DX,0082H ; Load 16-bit address of the output port in DX
OUT DX,AL    ; Send byte to output port from AL
```

8.4.1.9 I/O Interfacing with Memory Mapped I/O

As mentioned earlier, in memory mapped 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. Fig. 8.23 shows the I/O interfacing with memory mapped I/O. As shown in the Fig. 8.23, the 8086 uses same control signals and instructions to access I/O as those of memory. Here $\overline{RD}$ and $\overline{WR}$ signals are activated when $M/\overline{IO}$ signal is high, indicating memory bus cycle.

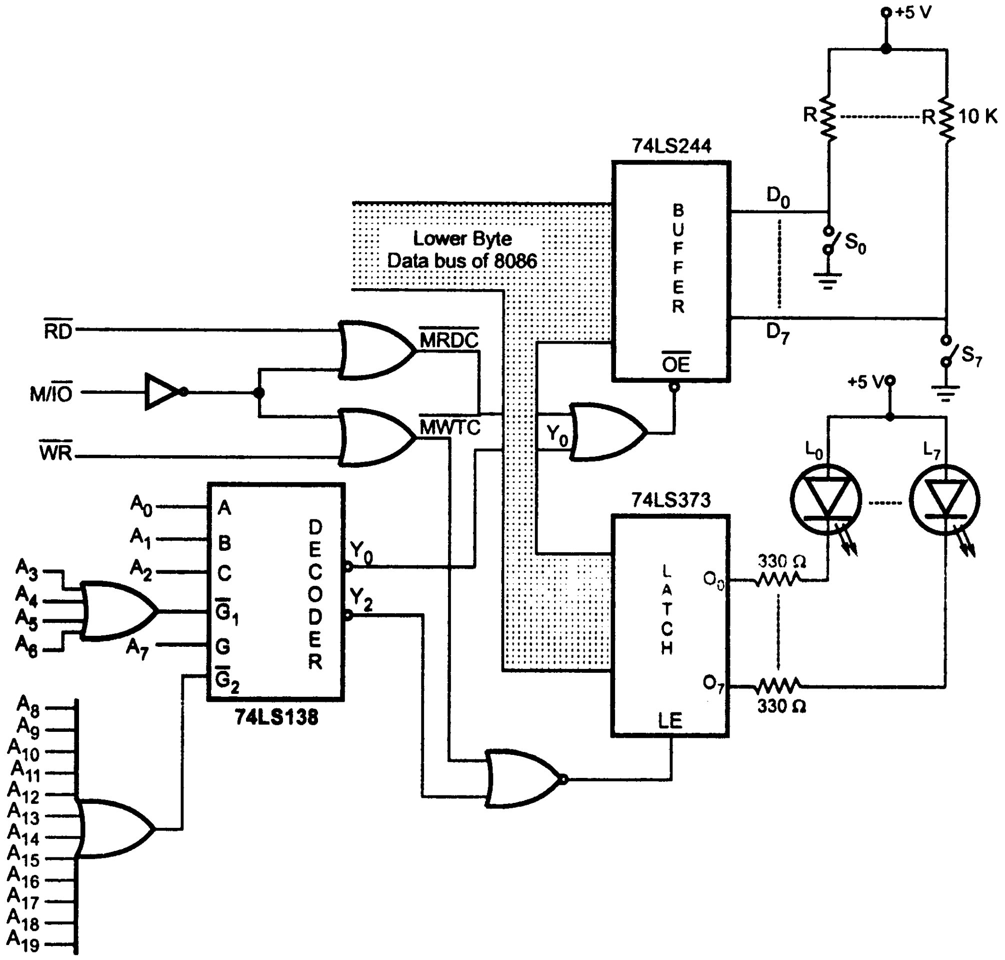


Fig. 8.23 I/O Interfacing with memory mapped I/O

Example :

Refer Fig. 8.23 and write a program to display the status of switches on the LEDs.

Program :

```
MOV AL,[0080H]; Read byte from input port in AL
MOV [0082],AL ; Send byte to output port from AL
```

8.4.2 Comparison between Memory Mapped I/O and I/O Mapped I/O

Sr. No.	Memory Mapped I/O	I/O Mapped I/O
1.	In this, the address of I/O device is 20-bit. The I/O device is connected as if it is a memory register.	In this the address of I/O device is 8-bit or 16-bit. It is 8-bit for direct addressing and 16-bit for indirect addressing.
2.	All memory related instructions can be used to access I/O device	In this, IN, INS, OUT and OUTS instructions are used to transfer data through the input/output ports.
3.	In 8086 system, I/O device is activated/selected when $M/\overline{IO}$ signal is high.	In 8086 system, I/O device is activated/selected when $M/\overline{IO}$ signal is low.
4.	In 8088 system, I/O device is activated/selected when $IO/\overline{M}$ signal is low.	In 8088 system, I/O device is activated/selected when $IO/\overline{M}$ signal is high.
5.	Maximum number of I/O devices are 1 Mbyte (theoretically).	Maximum number of I/O devices are 256 for direct addressing and 65536 for indirect addressing.
6.	Requires decoding of 20 address lines and hence more hardware involved.	Requires decoding of either 8 address lines or 16 address lines and hence comparatively less hardware involved.

Review Questions

1. *What is address ?*
2. *What is memory capacity.*
3. *Explain the memory structure and its requirements.*
4. *How much address lines are required to interface 4 kbytes of memory.*
5. *Explain various addressing decoding techniques.*
6. *Draw the interfacing diagram for 8086 based system (minimum mode) with the following specifications :*

   i) 16 kB RAM ii) 8 kB EPROM.

   Also show the required latches, buffers and decoder. Draw the memory map for the above interface.

7. *Draw the interfacing diagram for 8086 system in minimum mode with the following specifications :*

   i) 32 kB RAM   ii) 8 kB EPROM

   Also show the required latches, buffers and decoder. Draw the memory map for the above.

8. *What do you mean by input port ?*
9. *What do you mean by output port ?*
10. *Explain the I/O data transfer using I/O ports with the help of neat diagram.*
11. *Explain the I/O interfacing techniques :*

    a) I/O mapped I/O   b) Memory mapped I/O.

12. *Draw and explain the interfacing of*

    a) an 8-bit input port   b) an 16-bit input port

    c) an 8-bit output port   d) an 16-bit output port

    with microprocessor 8086.

13. *Explain various I/O data transfer techniques.*
14. *Give comparison between memory mapped I/O and I/O mapped I/O.*

❑❑❑

9 Programmable Peripheral Interface 8255

In this chapter, we are going to study programmable peripheral interface (PPI), 8255, designed by Intel. This chapter gives information about the block diagram, pin diagram, operating modes and interfacing requirements of 8255.

The 8255 is a general purpose programmable I/O device used for parallel data transfer. It has 24 I/O pins which can be grouped in three 8-bit parallel ports : Port A, Port B and Port C. The eight bits of port C can be used as individual bits or be grouped in two 4-bit ports : C_{upper} (C_U) and C_{lower} (C_L).

The 8255, primarily, can be programmed in two basic modes : Bit Set/Reset (BSR) mode and I/O mode. The BSR mode is used to set or reset the bits in port C. The I/O mode is further divided into three modes :

Mode 0 : Simple Input/Output

Mode 1 : Input/Output with handshake

Mode 2 : Bi-directional I/O data transfer

The function of I/O pins (input or output) and modes of operation of I/O ports can be programmed by writing proper control word in the control word register. Each bit in the control word has a specific meaning and the status of these bits decides the function and operating mode of the I/O ports.

9.1 Features of 8255A

1. The 8255A is a widely used, programmable, parallel I/O device.
2. It can be programmed to transfer data under various conditions, from simple I/O to interrupt I/O.
3. It is compatible with all Intel and most other microprocessors.
4. It is completely TTL compatible.

5. It has three 8-bit ports : Port A, Port B, and Port C, which are arranged in two groups of 12 pins. Each port has an unique address, and data can be read from or written to a port. In addition to the address assigned to the three ports, another address is assigned to the control register into which control words are written for programming the 8255 to operate in various modes.
6. Its bit set/reset mode allows setting and resetting of individual bits of Port C.
7. The 8255 can operate in 3 I/O modes : (i) Mode 0, (ii) Mode 1, and (iii) Mode2.
   a) In Mode 0, Port A and Port B can be configured as simple 8-bit input or output ports without handshaking. The two halves of Port C can be programmed separately as 4-bit input or output ports.
   b) In Mode 1, two groups each of 12 pins are formed. Group A consists of PortA and the upper half of Port C while Group B consists of Port B and the lower half of Port C. Ports A and B can be programmed as 8-bit Input or Output ports with three lines of Port C in each group used for handshaking.
   c) In Mode 2, only Port A can be used as a bidirectional port. The handshaking signals are provided on five lines of Port C (PC_3 - PC_7). Port B can be used in Mode 0 or in Mode 1.
8. All I/O pins of 8255 has 2.5 mA DC driving capacity (i.e. sourcing current of 2.5 mA).

9.2 Pin Diagram

Fig. 9.1 shows the pin diagram of 8255. (See Fig. 9.1 on next page)

Pin Symbols	Function
D_0-D_7 (Data Bus)	These bi-directional, tri-state data bus lines are connected to the system data bus. They are used to transfer data and control word from microprocessor (8085) to 8255 or to receive data or status word from 8255 to the 8085.
PA_0-PA_7 (Port A)	These 8-bit bi-directional I/O pins are used to send data to output device and to receive data from input device. It functions as an 8-bit data output latch/buffer, when used in output mode and an 8-bit data input buffer, when used in input mode.
PB_0-PB_7 (Port B)	These 8-bit bi-directional I/O pins are used to send data to output device and to receive data from input device. It functions as an 8-bit data, output latch/buffer when used in output mode and an 8-bit data input buffer, when used in input mode.
PC_0-PC_7	These 8-bit bi-directional I/O pins are divided into two groups PC_L (PC_3-PC_0) and PC_U (PC_7-PC_4). These groups individually can transfer data in or out when programmed for simple I/O, and used as handshake signals when programmed for handshake or bi-directional modes.
$\overline{RD}$ (Read)	When this pin is low, the CPU can read the data in the ports or the status word, through the data buffer.
$\overline{WR}$ (Write)	When this input pin is low, the CPU can write data on the ports or in the control register through the data bus buffer.
$\overline{CS}$ (Chip Select)	This is an active low input which can be enabled for data transfer operation between the CPU and the 8255.

Reset	This is an active high input used to reset 8255. When RESET input is high, the control register is cleared and all the ports are set to the input mode. Usually Reset Out signal from 8085 is used to reset 8255.
A_0 and A_1	These input signals along with $\overline{RD}$ and $\overline{WR}$ inputs control the selection of the control/status word registers or one of the three ports. Table. 9.1 summarizes the status of A_0, A_1, $\overline{CS}$, $\overline{RD}$ and $\overline{WR}$ to access the control word/ports. A_0 and A_1 are generally connected to the A_0, A_1 pins of the address bus; the 8255 therefore occupies four consecutive locations in the I/O space.

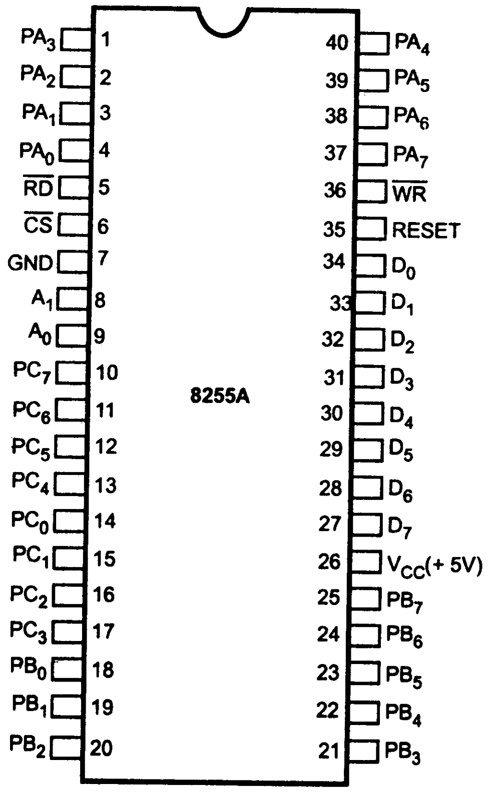


Fig. 9.1 Pin diagram of 8255A

A_1	A_0	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	Operations
					Input (Read) Operation
0	0	0	1	0	Port A to Data Bus
0	1	0	1	0	Port B to Data Bus
1	0	0	1	0	Port C to Data Bus
					Output (Write) Operation
0	0	1	0	0	Data Bus to Port A
0	1	1	0	0	Data Bus to Port B
1	0	1	0	0	Data Bus to Port C
1	1	1	0	0	Data Bus to Control Register