

Lecture FOUR

Interfacing I/O Devices

REAL-TIME SYSTEMS

INPUT/OUTPUT INSTRUCTIONS

All x86 microprocessors, from the 8088 to the Pentium, can access external devices called *ports*. This is done using I/O instructions. The x86 CPU is one of the few processors that have I/O space in addition to memory space. While memory can contain both opcodes and data, I/O ports contain data only. There are two instructions for this purpose: “OUT” and “IN”. These instructions can send data from the accumulator (AL or AX) to ports or bring data from ports into the accumulator. In accessing ports, we can use an 8-bit or 16-bit data port. Since 8-bit data ports in the 8088 are the most widely used, we will concentrate on them and introduce 16-bit data ports only in the last section of this chapter.

8-bit data ports

The 8-bit I/O operation of the 8088 is applicable to all x86 CPUs from the 8088 to the Pentium. The 8-bit port uses the D0 - D7 data bus to communicate with I/O devices. In 8-bit port programming, register AL is used as the source of data when using the OUT instruction and the destination for the IN instruction. This means that to input or output data from any other registers, the data must first be moved to the AL register. Instructions OUT and IN have the following formats:

	<u>Inputting Data</u>	<u>Outputting Data</u>
Format:	IN dest, source	OUT dest, source
(1)	IN AL, port#	OUT port#, AL
(2)	MOV DX, port# IN AL, DX	MOV DX, port# OUT DX, AL

In format (1), port# is the address of the port and can be from 00 to FFH. This 8-bit address allows 256 input ports and 256 output ports. In this format, the 8-bit port address is carried on address bus A0 - A7. No segment register is involved in computing the address, in contrast to the way data is accessed from memory.

In format (2), the port# is also the address of the port, except that it can be from 0000 to FFFFH, allowing up to 65,536 input and 65,536 output ports. In this case, the 16-bit port address is carried on address bus A0 - A15, and no segment register (DS) is involved. This is the way Intel Corporation expanded the number of ports from 256 to 65,536 while maintaining compatibility with the earlier 8085 microprocessors. The use of a register as a pointer for the port address has an advantage in that the port address can be changed very easily, especially in cases of dynamic compilations where the port address can be passed to DX.

How to use I/O instructions

I/O instructions are widely used in programming peripheral devices such as printers, hard disks, and keyboards. The port address can be either 8-bit or 16-bit. For an 8-bit port address, we can use the immediate addressing mode. The following program sends a byte of data to a fixed port address of 43H.

```
MOV    AL,36H           ;AL=36H
OUT     43H,AL           ;send value 36H to port address 43H
```

The 8-bit address used in immediate addressing mode limits the number of ports to 256 for input plus 256 for output. To have a larger number of ports we must use the 16-bit port address instruction.

To use the 16-bit port address, register indirect addressing mode must be used. The register used for this purpose is DX. The following program sends values 55H and AAH to I/O port address 300H (a 16-bit port address). In other words, the program below toggles the bits of port address 300H continuously.

```
BACK:  MOV    DX,300H      ;DX = port address 300H
        MOV    AL,55H
        OUT     DX,AL      ;toggle the bits
        MOV    AL,0AAH
        OUT     DX,AL      ;toggle the bits
        JMP     BACK
```

Notice that we can only use register DX for 16-bit I/O addresses; no other register can be used for this purpose. Also notice the use of register AL for 8-bit data. For example, the following code transfers the contents of register BL to port address 378H.

MOV DX,378H	;DX=378 the port address
MOV AL,BL	;load data into accumulator
OUT DX,AL	;write contents of AL to port ;whose address is in DX

Example 11-1

In a given 8088-based system, port address 22H is an input port for monitoring the temperature. Write Assembly language instructions to monitor that port continuously for the temperature of 100 degrees. If it reaches 100, then BH should contain 'Y'.

Solution:

BACK:	IN	AL,22H	;get the temperature from port # 22H
	CMP	AL,100	;is temp = 100?
	JNZ	BACK	;if not, keep monitoring
	MOV	BH,'Y'	;temp = 100, load 'Y' into BH

Just like the OUT instruction, the IN instruction uses the DX register to hold the address and AL to hold the arrived 8-bit data. In other words, DX holds the 16-bit port address while AL receives the 8-bit data brought in from an external port. The following program gets data from port address 300H and sends it to port address 302H.

MOV	DX,300H	;load port address
IN	AL,DX	;bring in data
MOV	DX,302H	
OUT	DX,AL	;send it out

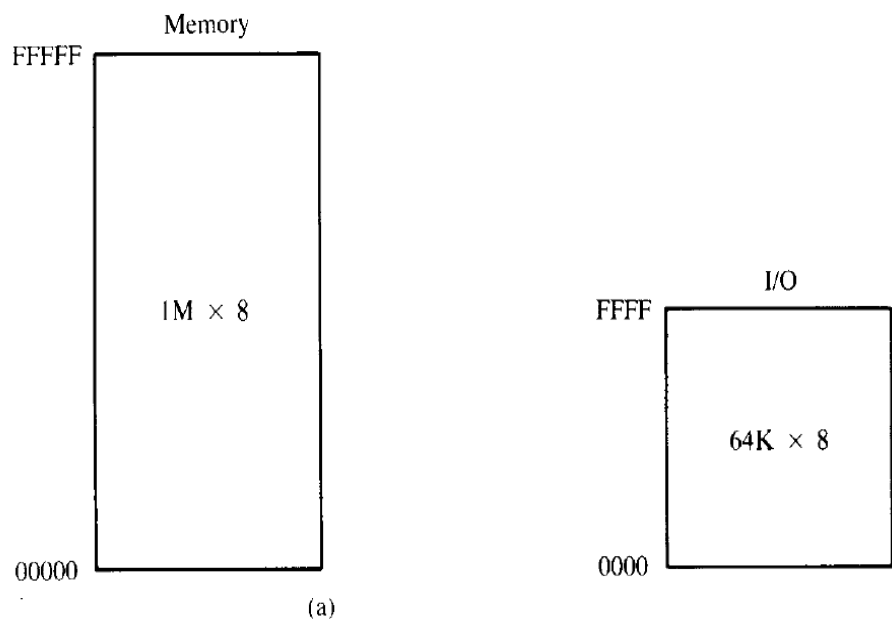
Isolated and Memory-Mapped I/O

There are two different methods of interfacing I/O to the microprocessor: **isolated I/O** and **memory-mapped I/O**. In the isolated I/O scheme, the IN, INS, OUT, and OUTS instructions transfer data between the microprocessor's accumulator or memory and the I/O device. In the memory-mapped I/O scheme, any instruction that references memory can accomplish the transfer. Both isolated and memory-mapped I/O are in use, so both are discussed in this text. The PC does not use memory-mapped I/O.

Isolated I/O. The most common I/O transfer technique used in the Intel microprocessor-based system is isolated I/O. The term *isolated* describes how the I/O locations are isolated from the memory system in a separate I/O address space. (Figure 11–1 illustrates both the isolated and memory-mapped address spaces for any Intel 80X86 or Pentium–Core2 microprocessor.) The addresses for isolated I/O devices, called ports, are separate from the memory. Because the ports are separate, the user can expand the memory to its full size without using any of memory space for I/O devices. A disadvantage of isolated I/O is that the data transferred between I/O and the microprocessor must be accessed by the IN, INS, OUT, and OUTS instructions. Separate control signals for the I/O space are developed (using $M/\overline{IO}$ and $\overline{W/R}$), which indicate an I/O read ($\overline{IORC}$) or an I/O write ($\overline{IOWC}$) operation. These signals indicate that an I/O port address, which appears on the address bus, is used to select the I/O device. In the personal computer, isolated I/O ports are used for controlling peripheral devices. An 8-bit port address is used to access devices located on the system board, such as the timer and keyboard interface, while a 16-bit port is used to access serial and parallel ports as well as video and disk drive systems.

Memory-Mapped I/O. Unlike isolated I/O, memory-mapped I/O does not use the IN, INS, OUT, or OUTS instructions. Instead, it uses any instruction that transfers data between the microprocessor and memory. A memory-mapped I/O device is treated as a memory location in the memory map. The main advantage of memory-mapped I/O is that any memory transfer instruction can be used to access the I/O device. The main disadvantage is that a portion of the memory system is used as the I/O map. This reduces the amount of memory available to applications. Another advantage is that the $\overline{\text{IORC}}$ and $\overline{\text{IOWC}}$ signals have no function in a memory mapped I/O system and may reduce the amount of circuitry required for decoding.

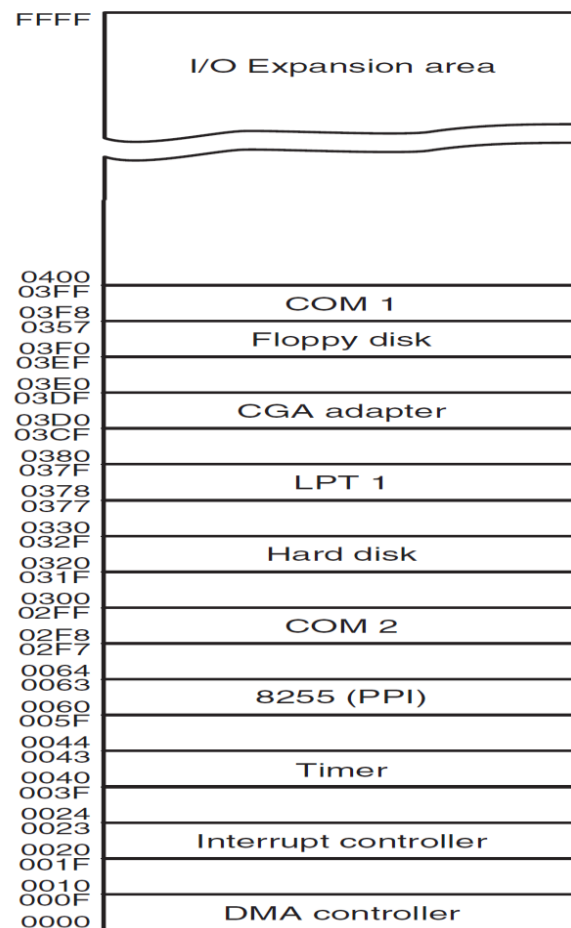
FIGURE 11-1 The memory and I/O maps for the 8086/8088 microprocessors. (a) Isolated I/O. (b) Memory-mapped I/O.



Personal Computer I/O Map

The personal computer uses part of the I/O map for dedicated functions. Figure 11–2 shows the I/O map for the PC. Note that I/O space between ports 0000H and 03FFH is normally reserved for the computer system and the ISA bus. The I/O ports located at 0400H–FFFFH are generally available for user applications, main-board functions, and the PCI bus. Note that the 80287 arithmetic coprocessor uses I/O address 00F8H–00FFH for communications. For this reason, Intel reserves I/O ports 00F0H–00FFH. The 80386–Core2 use I/O ports 800000F8–800000FFH for communications to their coprocessors. The I/O ports located between 0000H and 00FFH are accessed via the fixed port I/O instructions; the ports located above 00FFH are accessed via the variable I/O port instructions.

FIGURE 11–2 The I/O map of a personal computer illustrating many of the fixed I/O areas.



Handshaking

Many I/O devices accept or release information at a much slower rate than the microprocessor. Another method of I/O control, called **handshaking** or **polling**, synchronizes the I/O device with the microprocessor. An example of a device that requires handshaking is a parallel printer that prints a few hundred characters per second (CPS). It is obvious that the microprocessor can send more than a few hundred CPS to the printer, so a way to slow the microprocessor down to match speeds with the printer must be developed.

Figure 11-5 illustrates the typical input and output connections found on a printer. Here, data are transferred through a series of data connections (D_7 – D_0). BUSY indicates that the printer is busy. $\overline{STB}$ is a clock pulse used to send data to the printer for printing.

2: I/O ADDRESS DECODING AND DESIGN

In this section we show the design of simple I/O ports using TTL logic gates 74LS373 and 74LS244. For the purpose of clarity we use simple logic gates such as AND and inverter gates for decoders. It may be helpful to review the address decoding section for memory interfacing in the preceding chapter before you embark on this section. The concept of address bus decoding for I/O instructions is exactly the same as for memory. The following are the steps:

1. The control signals IOR and IOW are used along with the decoder.
2. For an 8-bit port address, A_0 - A_7 is decoded.
3. If the port address is 16-bit (using DX), A_0 - A_{15} is decoded.

Using the 74LS373 in an output port design

In every computer, whenever data is sent out by the CPU via the data bus, the data must be latched by the receiving device. While memories have an internal latch to grab the data, a latching system must be designed for simple I/O ports.

The 74LS373 can be used for this purpose. Notice in Figure 11-1 that in order to make the 74LS373 work as a latch, the OC pin must be grounded. For an output latch, it is common to AND the output of the address decoder with the control signal IOW to provide the latching action as shown in Figure 11-2.

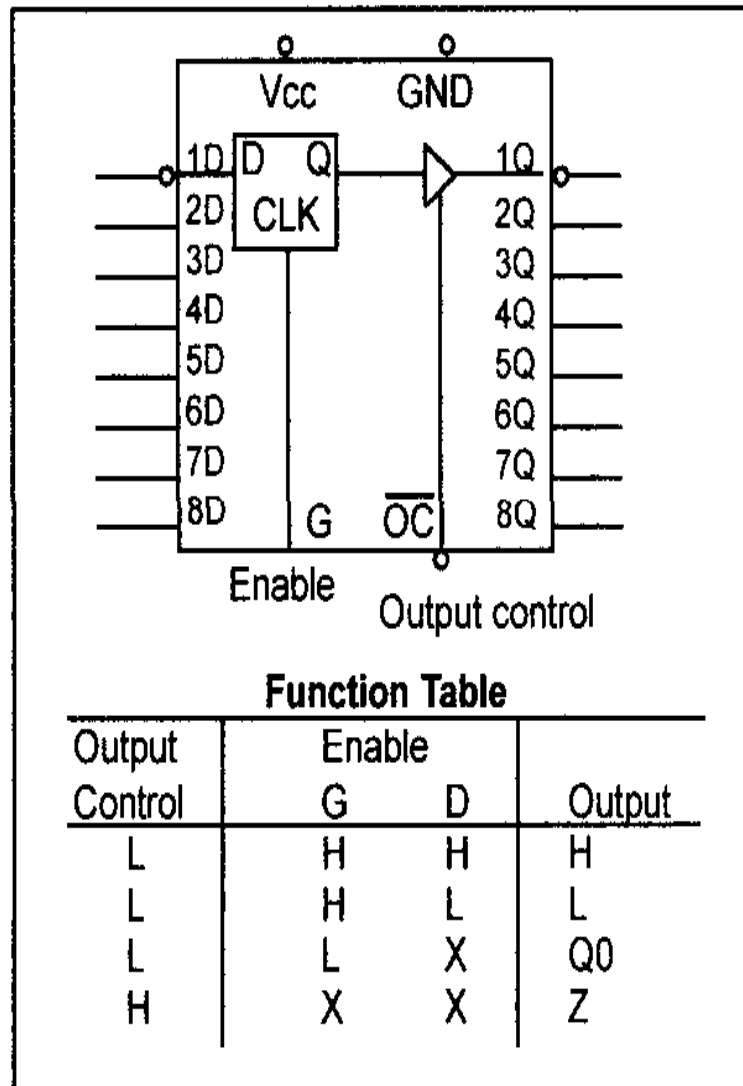


Figure 11-1. 74LS373 D Latch

(Reprinted by permission of Texas Instruments. Copyright Texas Instruments, 1988)

IN port design using the 74LS244

Likewise, when data is coming in by way of a data bus, it must come in through a three-state buffer. This is referred to as *tristated*, which comes from the term tri-state buffer ("tri-state" is a registered trademark of National

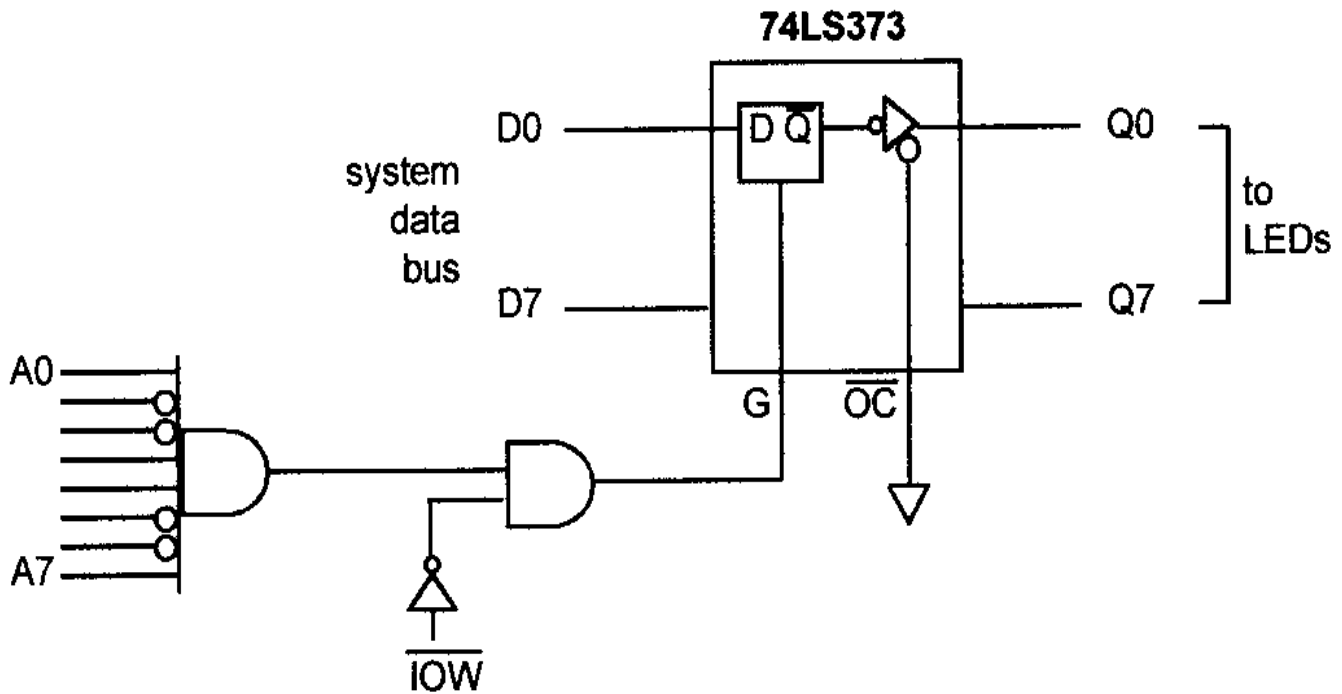
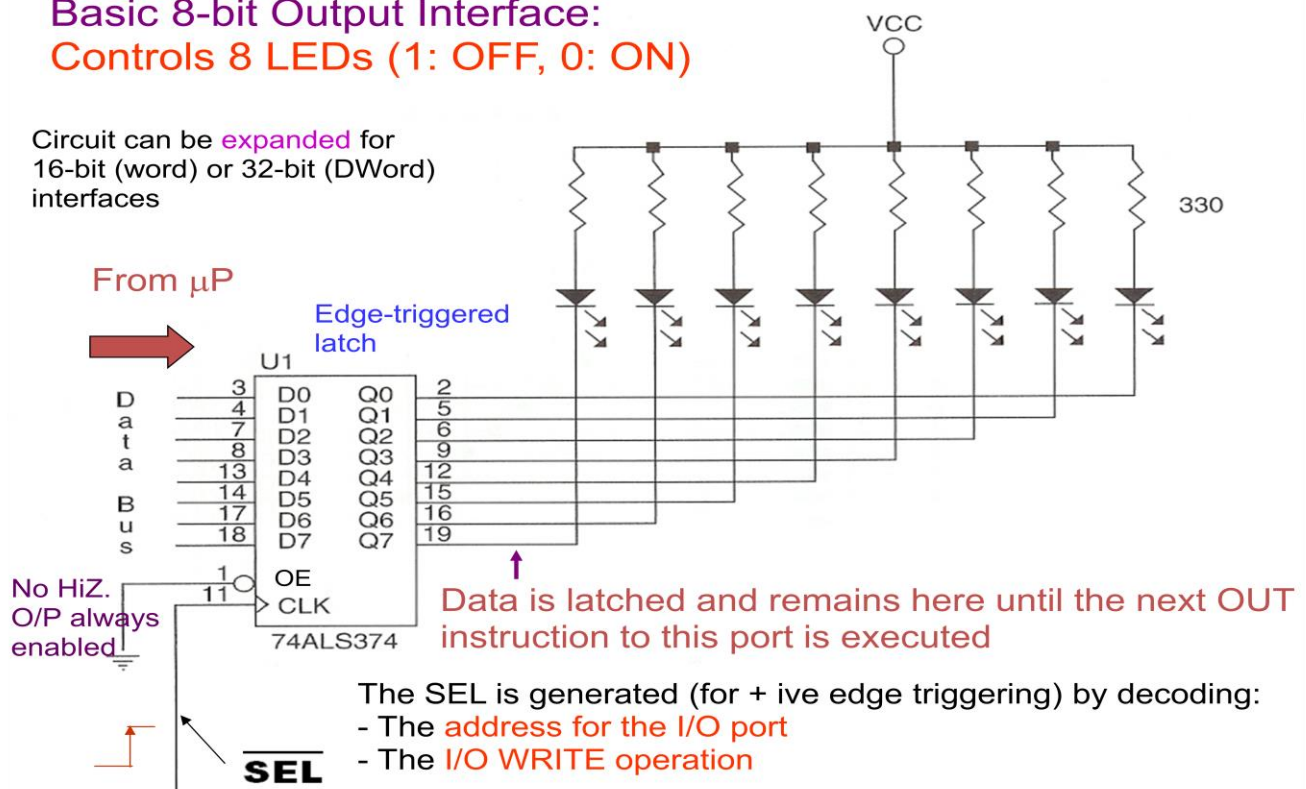


Figure 11-2. Design for "OUT 99H,AL"

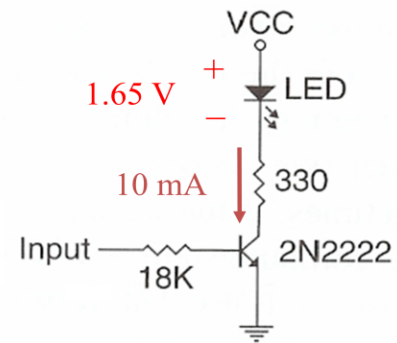
Basic 8-bit Output Interface:
Controls 8 LEDs (1: OFF, 0: ON)

Circuit can be expanded for 16-bit (word) or 32-bit (DWord) interfaces



Output Devices: The LED

- LED diodes are used in many systems as indicators and as sources of infrared beams
- LED must be forward biased to conduct and emit light
- When conducting, a typical (Red, Green, or Yellow) LED passes a current of about 10 mA with a voltage drop $V_{\text{diode}} \approx 1.65 \text{ V}$ (1.5 V - 2 V in practice)
- A Blue, White, or UltraBright LED passes more current at a larger voltage drop, e.g. 30 mA @ 2.5 V.

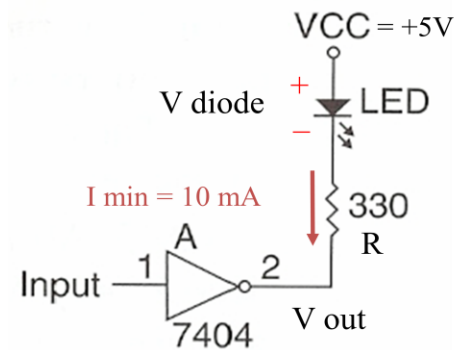


A bit of electronics!

Interfacing the LED directly to a TTL O/P

- 10 mA currents can be easily sunk by a TTL output at the low state (0)
(A standard 74 TTL O/P sinks up to 16 mA – lower values for lower power series e.g. 74LS sinks only 4 mA)
- $V_{\text{out, low}} \approx 0.1 \text{ V}$
- $V_{\text{diode}} \approx 1.65 \text{ V}$
- To ensure $I = 10 \text{ mA}$,

R should be



LED ON indicates 1 or 0 at Input?

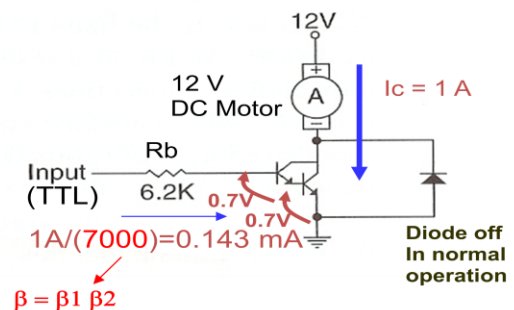
$$R = \frac{V_{\text{CC}} - V_{\text{diode}} - V_{\text{out}}}{10 \text{ mA}}$$

$$= \frac{5 - 1.65 - 0.1}{10} = 0.325 \text{ K}\Omega \approx 330 \Omega$$

Nearest standard resistor value

Driving larger currents and voltages; e.g. DC motors, mechanical relays, etc.

- Large current loads such as motors or large relays require a **Darlington pair** in place of the transistor driver
- Two transistor gains in cascade, ($\beta = \beta_1 \beta_2$), so smaller base currents from TTL for large load currents
- Can use 12 V or higher supply
- Select a transistor that meets both the voltage and current requirements for the load
- The diode is used to prevent the transistor from being destroyed by the inductive kickback current that appears when the field collapses suddenly in the coil



Current can not change instantly through an inductor. If it suddenly drops from I_1 to 0, a negative current $-I_1$ is generated which gradually decays to 0. The diode provides a safe path for this current away from the transistor

$$R_b = \frac{V_{out, high, min} - 0.7 - 0.7}{0.143mA}$$

$$= \frac{2.4 - 1.4}{0.143} = 7 K\Omega \approx 6.8 k\Omega$$

Input Devices. Input devices are already TTL and compatible, and therefore can be connected to the microprocessor and its interfacing components, or they are switch-based. Most switch-based devices are either open or connected. These are not TTL levels—TTL levels are a logic 0 (0.0 V–0.8 V) or a logic 1 (2.0 V–5.0 V).

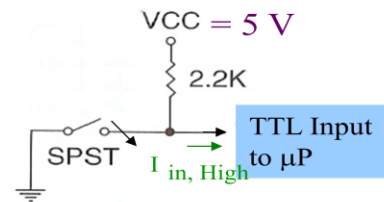
For a switch-based device to be used as a TTL-compatible input device, some conditioning must be applied. Figure 11-6 shows a simple toggle switch that is properly connected to function as an input device. Notice that a pull-up resistor is used to ensure that when the switch is open, the output signal is a logic 1; when the switch is closed, it connects to ground, producing a valid logic 0 level. The value of the pull-up resistor is not critical—it merely assures that the signal is

As was the case for memory chips, such a tri-state buffer is internal and therefore invisible. For the simple input ports we use the 74LS244 chip. See Figure 11-4 for the internal circuitry of the 74LS244. In Figure 11-4, notice that since G1 and G2 each control only 4 bits of the 74LS244, they both must be activated for the 8-bit input. Examine Figure 11-5 to see the use of the 74LS244 as

The 74LS244 not only plays the role of buffer, but also provides the incoming data with sufficient driving capability to travel all the way to the CPU. Indeed, the 74LS244 chip is widely used for buffering and providing high driving capability for unidirectional buses. The 74LS245 is used for bidirectional buses,

Input Devices: The Switch

- The most basic input device
- Switches are passive- do not produce a voltage, easily present a logic 0 to an I/P by grounding it.
- To make a switch TTL compatible, a pullup resistor is used as shown. The value should be between 1K and 27K Ω .
- Why not, say, 400 K Ω ?
- Switches bounce... and this is often undesirable.



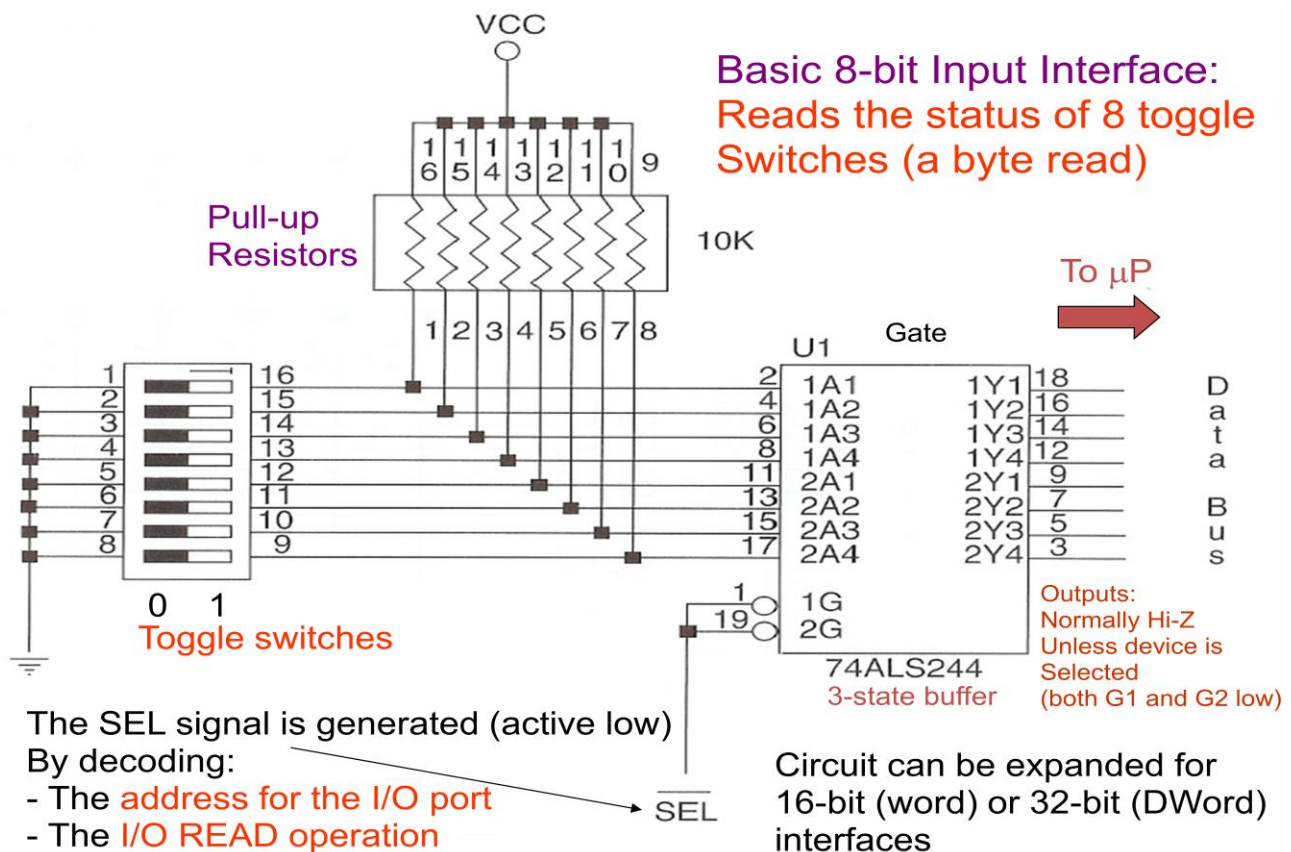
(Single Pole Single Throw)

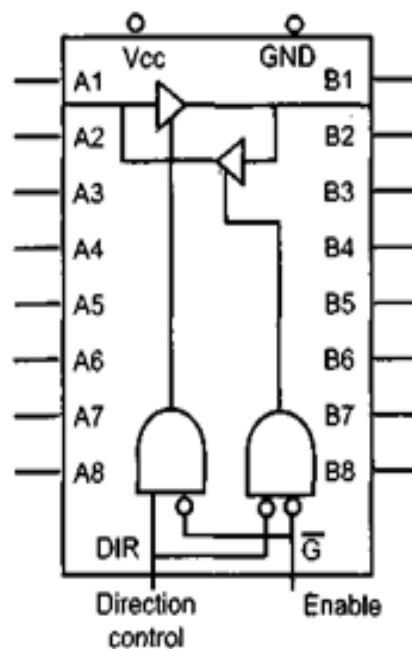
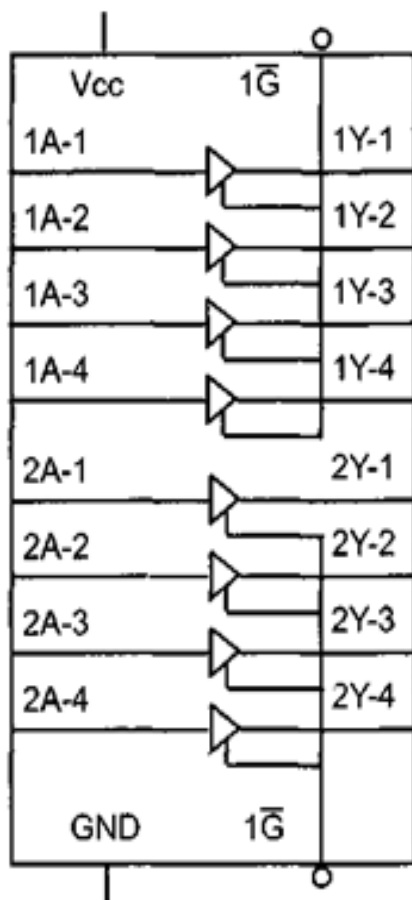
The pullup resistor does the **conditioning** required to allow the switch to produce a TTL compatible input to the processor

Here, processor senses the switch effect directly- hence **bouncing** is a problem, especially if input is used as a clock

Wrong switch State Can be read Here !

Initial Posn Final Posn

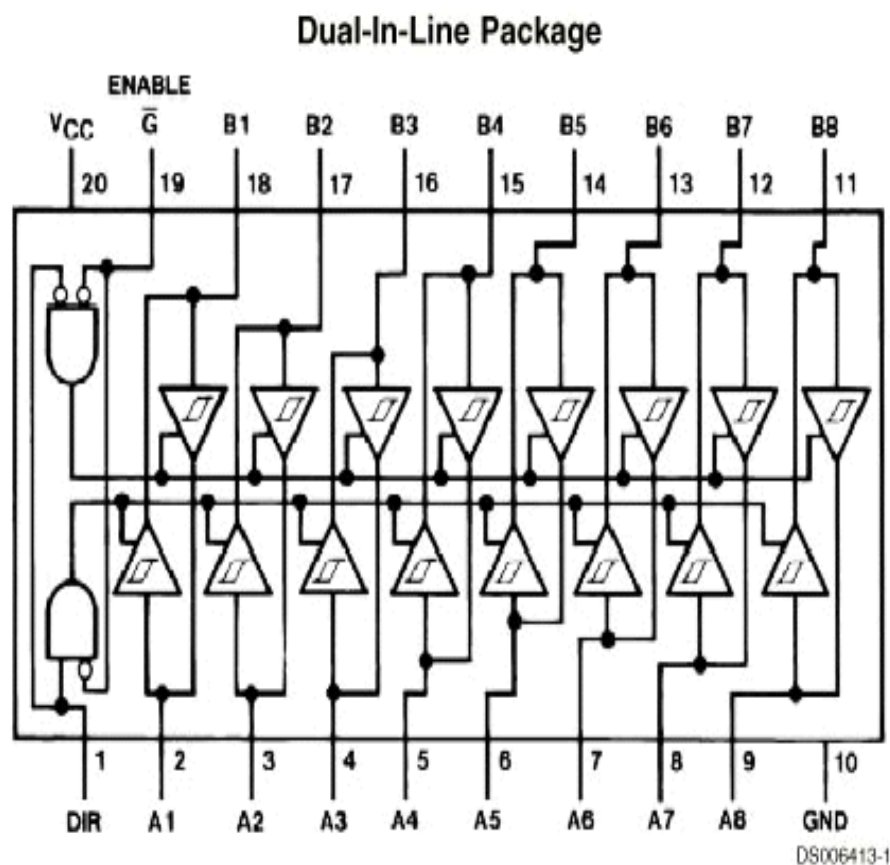




Function Table

Enable $\overline{G}$	Direction control DIR	Operation
L	L	B Data to A Bus
L	H	A Data to B Bus
H	X	Isolation

Connection Diagram



Order Number 54LS245DMQB, 54LS245FMQB, 54LS245LMQB,
DM54LS245J, DM54LS245W, DM74LS245WM or DM74LS245N
See Package Number E20A, J20A, M20B, N20A or W20A

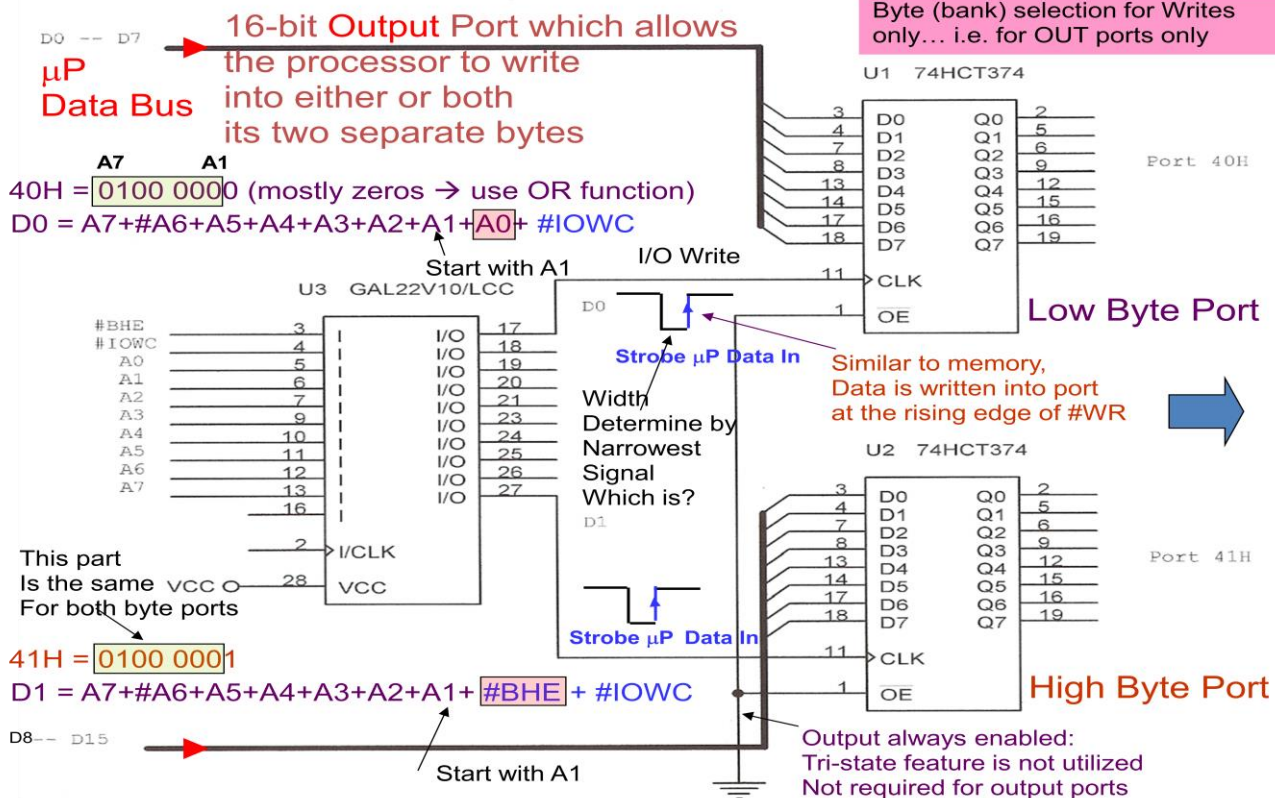
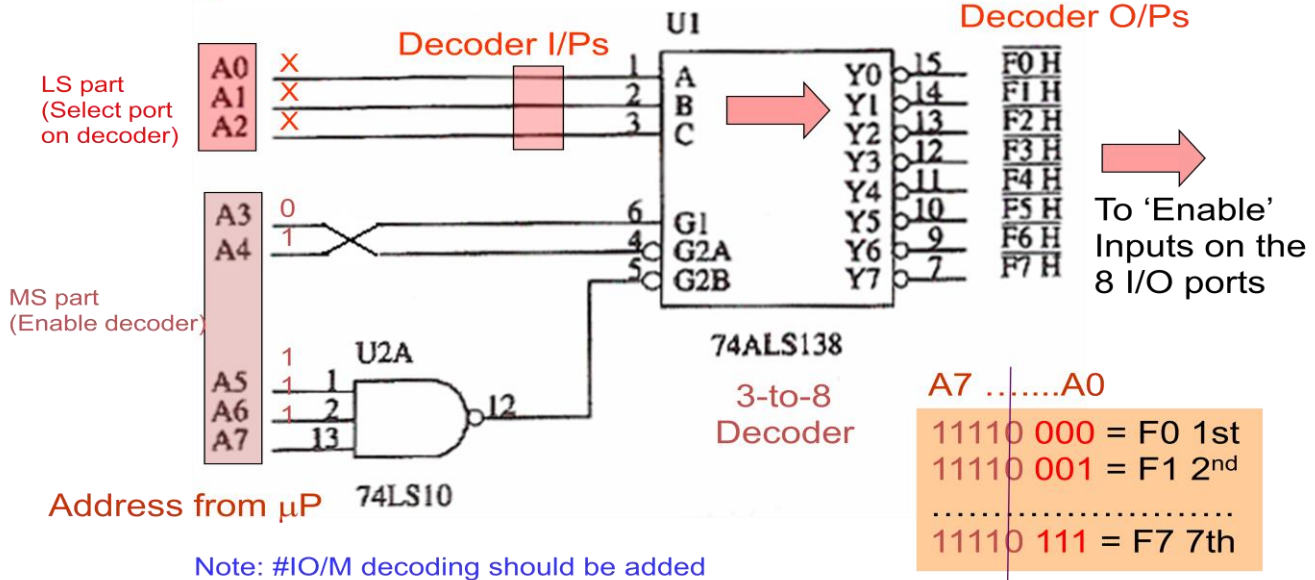
Function Table

Enable $\overline{G}$	Direction Control DIR	Operation
L	L	B data to A bus
L	H	A data to B bus
H	X	Isolation

H = High Level, L = Low Level, X = Irrelevant

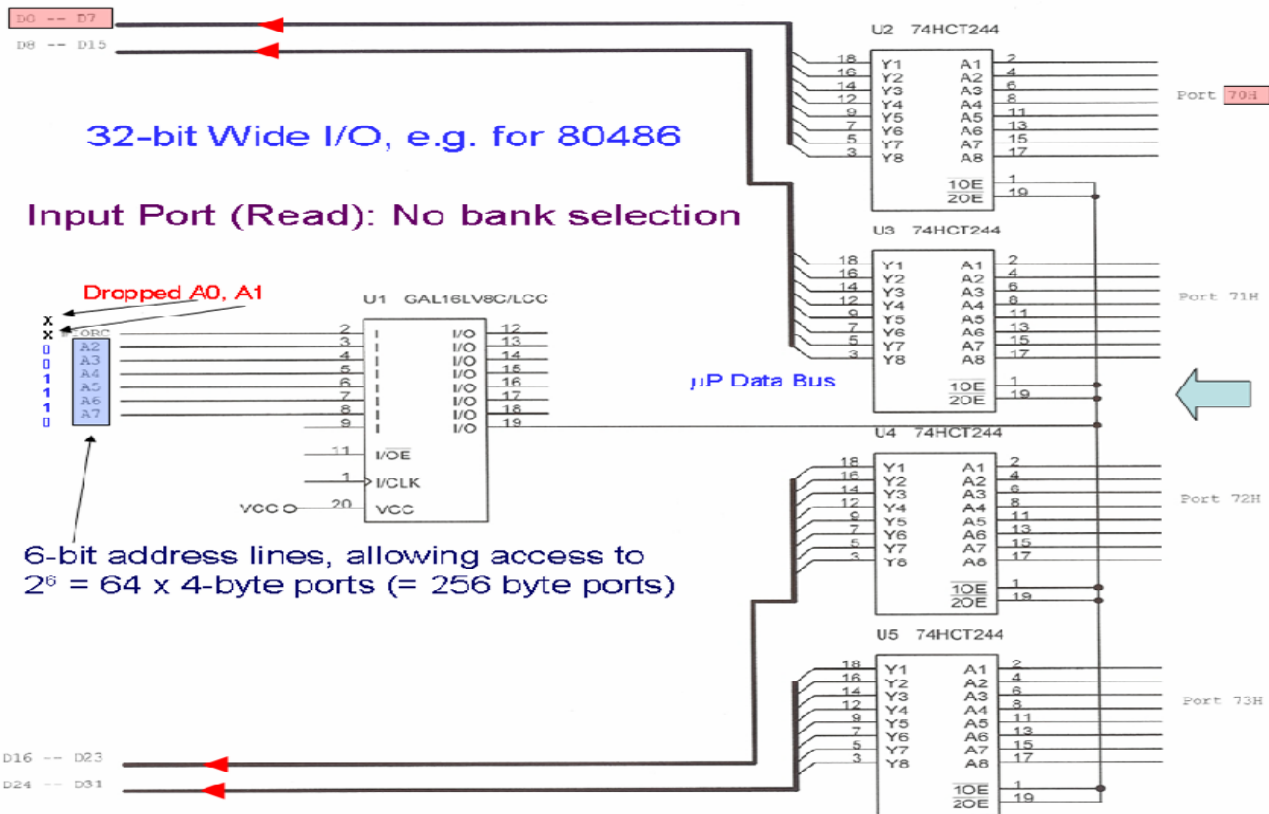
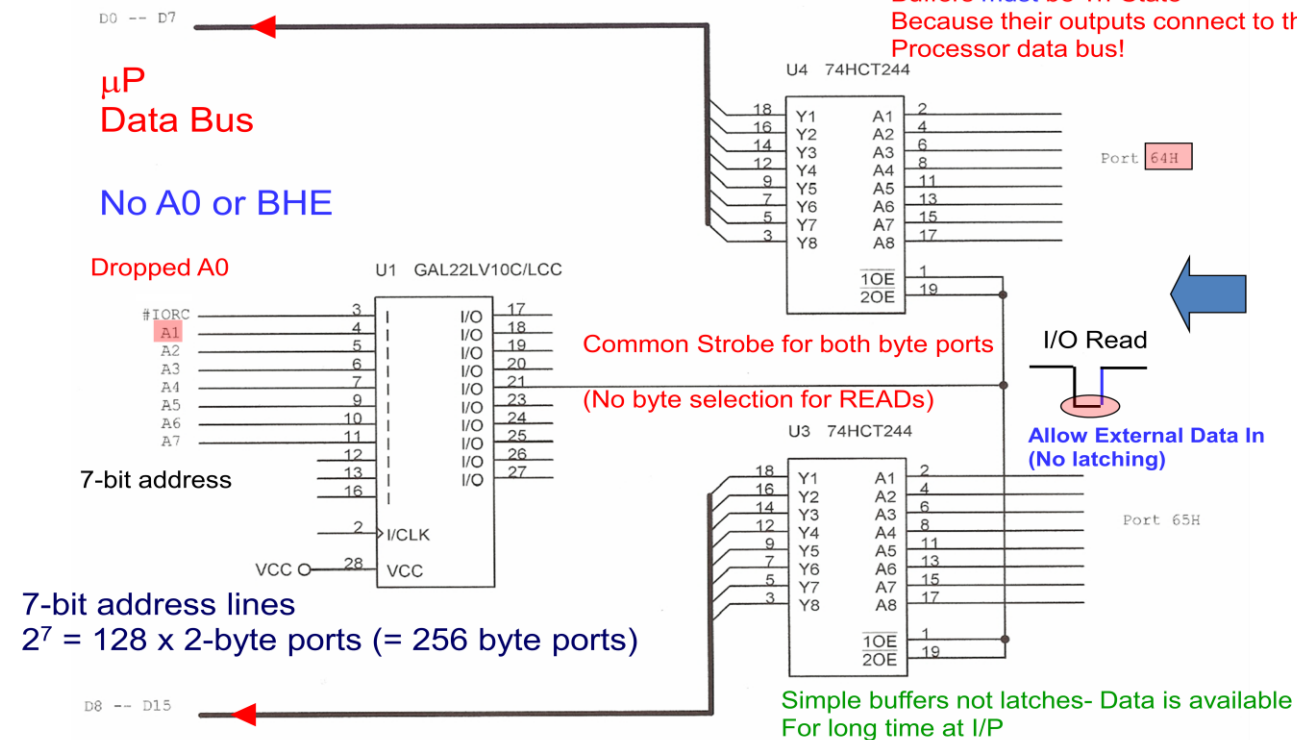
Address Decoding for 8 I/O Ports: F0 to F7 for the 8088 using 8-bit I/O address

a. Using a Decoder IC



16-bit Input Port: No use of BHE or A0 (Processor knows which byte it wants)

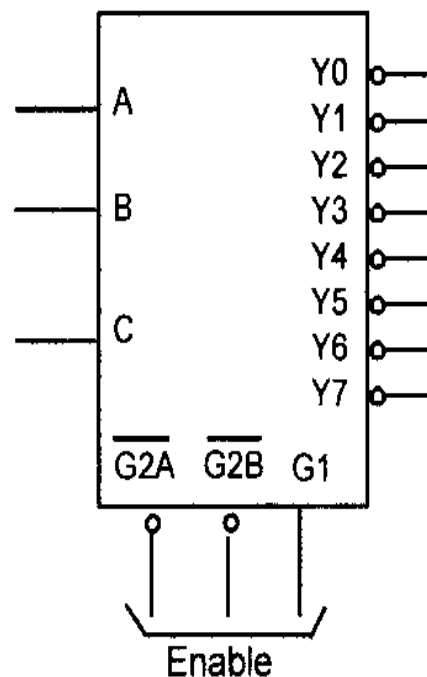
Buffers must be Tri-State
Because their outputs connect to the
Processor data bus!



Use of 74LS138 as decoder

In current system board design, CPLD (complex programmable logic device) chips are used for supporting logics such as decoders. In the absence of CPLD, one could use NANDs, inverters, and 74LS138 chips for decoders as we saw in the preceding chapter for memory address decoding. The same principle applies to I/O address decoding. Figure 11-8 shows the 74LS138. As an example of the use of a 74LS138 for an I/O address decoder, examine Figure 11-9. Notice how each Y output can control a single device. Figure 11-9 shows the address decoding for an input port located at address 304H. The Y4 output, together with the IOR signal, controls the 74LS244 input buffer. Alternatively, Y0 along with the IOW control signal could be used to control a 74LS373 latch. In other words, each Y output controls a single I/O device. Contrast that with Figure 11-7. The 74LS138 is much more efficient than the use of simple logic gates as decoders.

Block Diagram



Function Table

Inputs									
Enable	Select	Outputs							
G1G2	C B A	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
X H	X X X	H	H	H	H	H	H	H	H
L X	X X X	H	H	H	H	H	H	H	H
H L	L L L	L	H	H	H	H	H	H	H
H L	L L H	H	L	H	H	H	H	H	H
H L	L H L	H	H	L	H	H	H	H	H
H L	L H H	H	H	H	L	H	H	H	H
H L	H L L	H	H	H	H	L	H	H	H
H L	H L H	H	H	H	H	H	L	H	H
H L	L L L	H	H	H	H	H	H	L	H
H L	H H H	H	H	H	H	H	H	H	L

Figure 11-8. 74LS138 Decoder

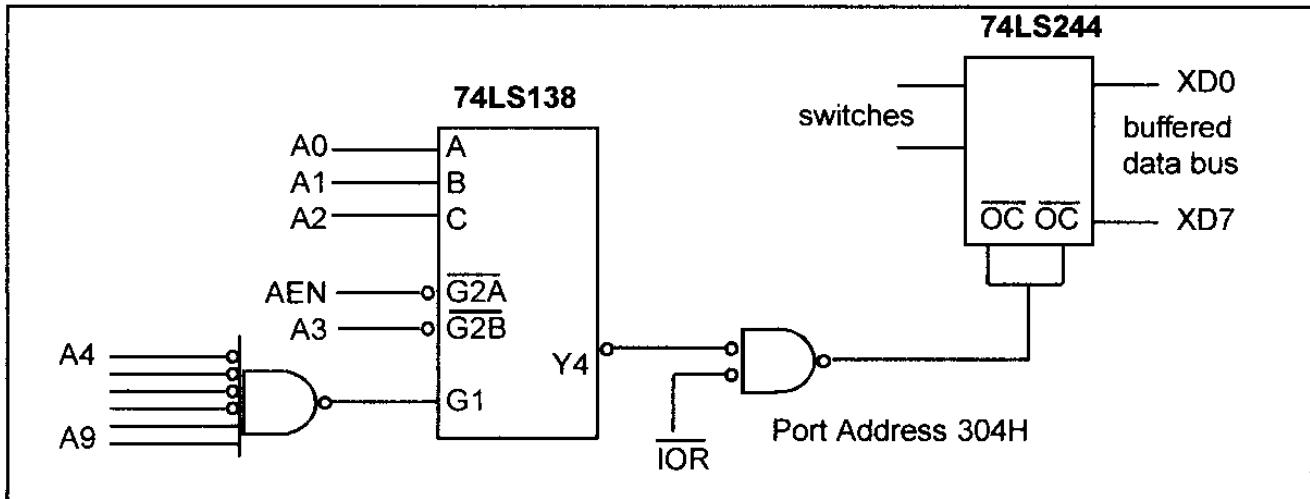
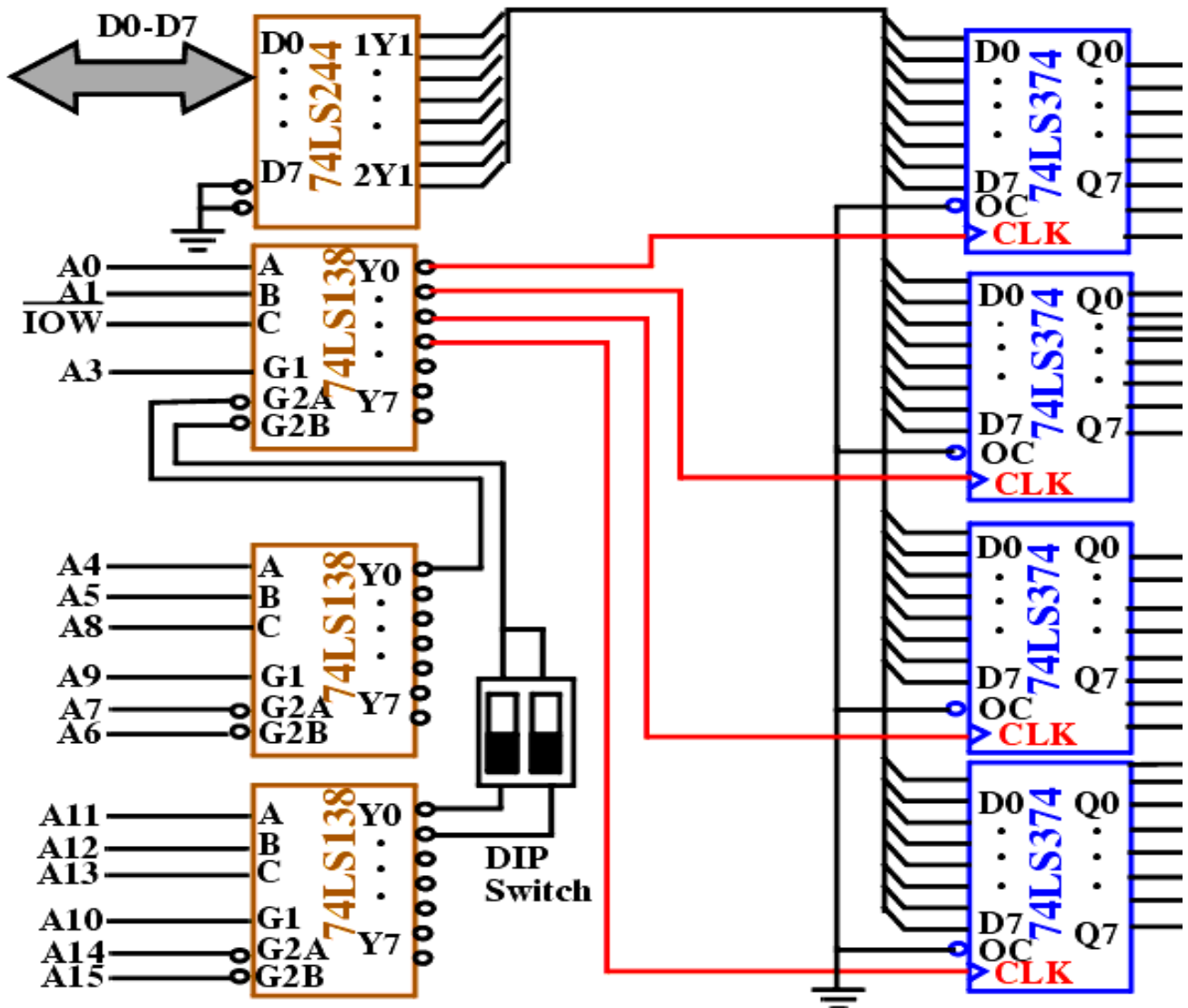
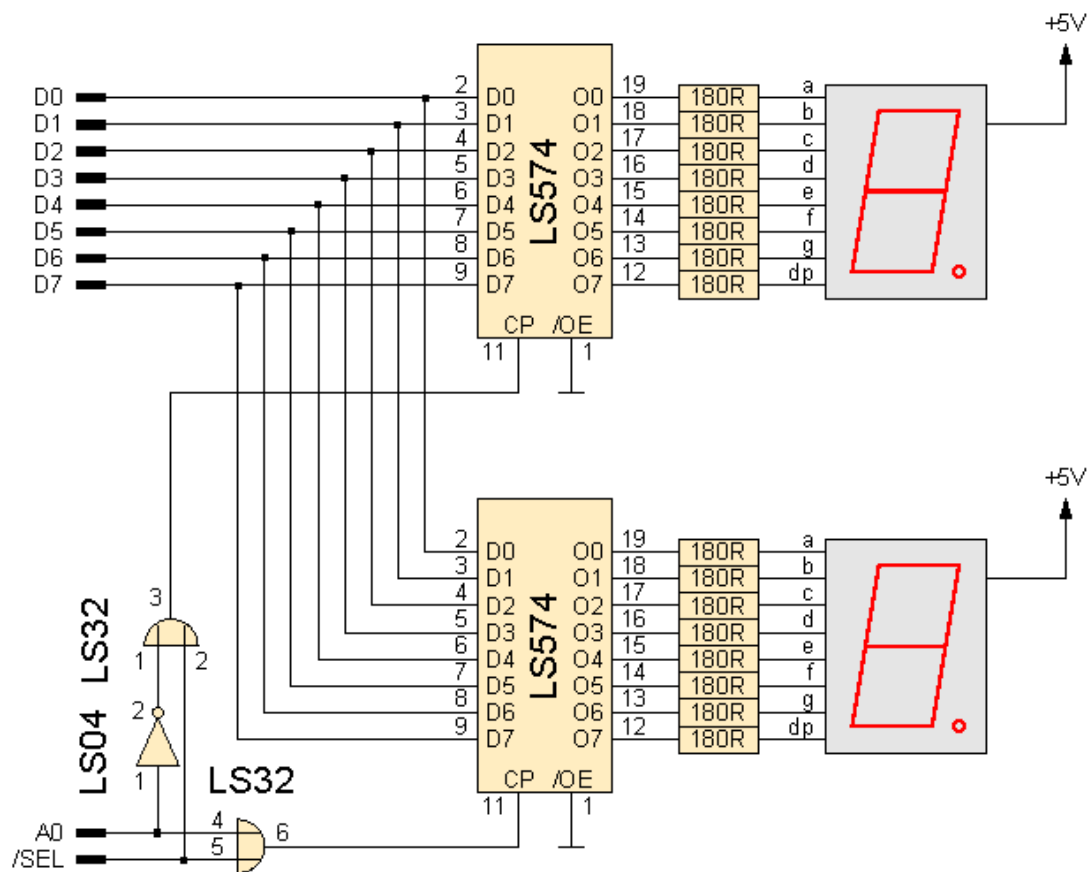


Figure 11-9. Using 74LS138 for I/O Address Decoder



EX: Design 8086 MPU interfacing circuit has TWO 7- segment at the address A0=0 and A0=1 then write the instructions to display NO. 98 on it?



MOV DX,0100

MOV AL,20

OUT DL,AL

MOV AL,00

OUT DH,AL

INT 3

s	BCD	G	F	E	D	C	B	A
0	0000	0	1	1	1	1	1	1
1	0001	0	0	0	0	1	1	0
2	0010	1	0	1	1	0	1	1
3	0011	1	0	0	1	1	1	1
4	0100	1	1	0	0	1	1	0
5	0101	1	1	0	1	1	0	1
6	0110	1	1	1	1	1	0	1
7	0111	0	0	0	0	1	1	1
8	1000	1	1	1	1	1	1	1
9	1001	1	1	0	1	1	1	1

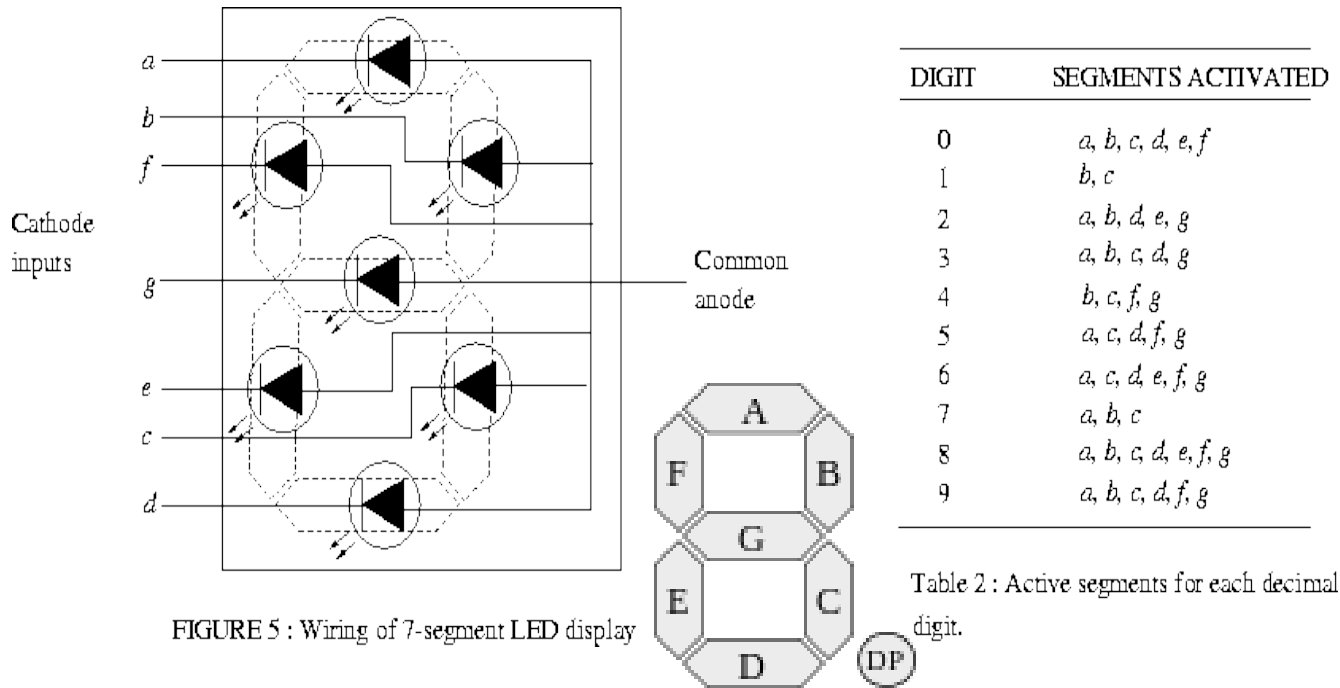


Table 2 : Active segments for each decimal digit.

