Chapter-14

Programmable Interface Devices:

❑ **8155 I/O and Timer**
❑ **8279 Keyboard/Display Interface**

A programmable interface device is designed to perform various input/output functions. Such a device can be set up to perform specific functions by writing an instruction (or instructions) in its internal register, called **control register**.

14.1 BASIC CONCEPTS IN PROGRAMMABLE DEVICES

In Chapter 5 we assumed that the I/O devices were always ready for data transfer. But in the situation to interface a printer (a slower device) to a microprocessor (the fastest unit) there must be some signals exchanged between them prior to sending and receiving data. These signals are called handshake signals. Otherwise, processor will read the same data again and again for a slow responding input device and overwrite the previous data by the next data for a slow responding output device.

14.1.1 Making the 74LS245 Transceiver Programmable

The 74LS245 is a bidirectional tri-state octal buffer, and the direction of data flow is determined by the signal of DIR. When DIR is high, data flow from S to B, and when it is low, data flow from B to A.

In fact, this is a hard-wired programmable device; the direction of the data flow is programmed through DIR. However, we are interested in a device that can be programmed by writing an instruction through the MPU. This can be accomplished by adding a register called the control register, as shown in Figure 14.2, and by connecting the DIR signal to bit D_o of the control register. When $D_o = 1$, data flow from A to B as output, and when $D_o = 0$, data flow in the opposite direction as input.

Example 14.1: Write instructions to initialize the hypothetical chip (Figure 14.2) as an output buffer.

Solution: **Instructions**
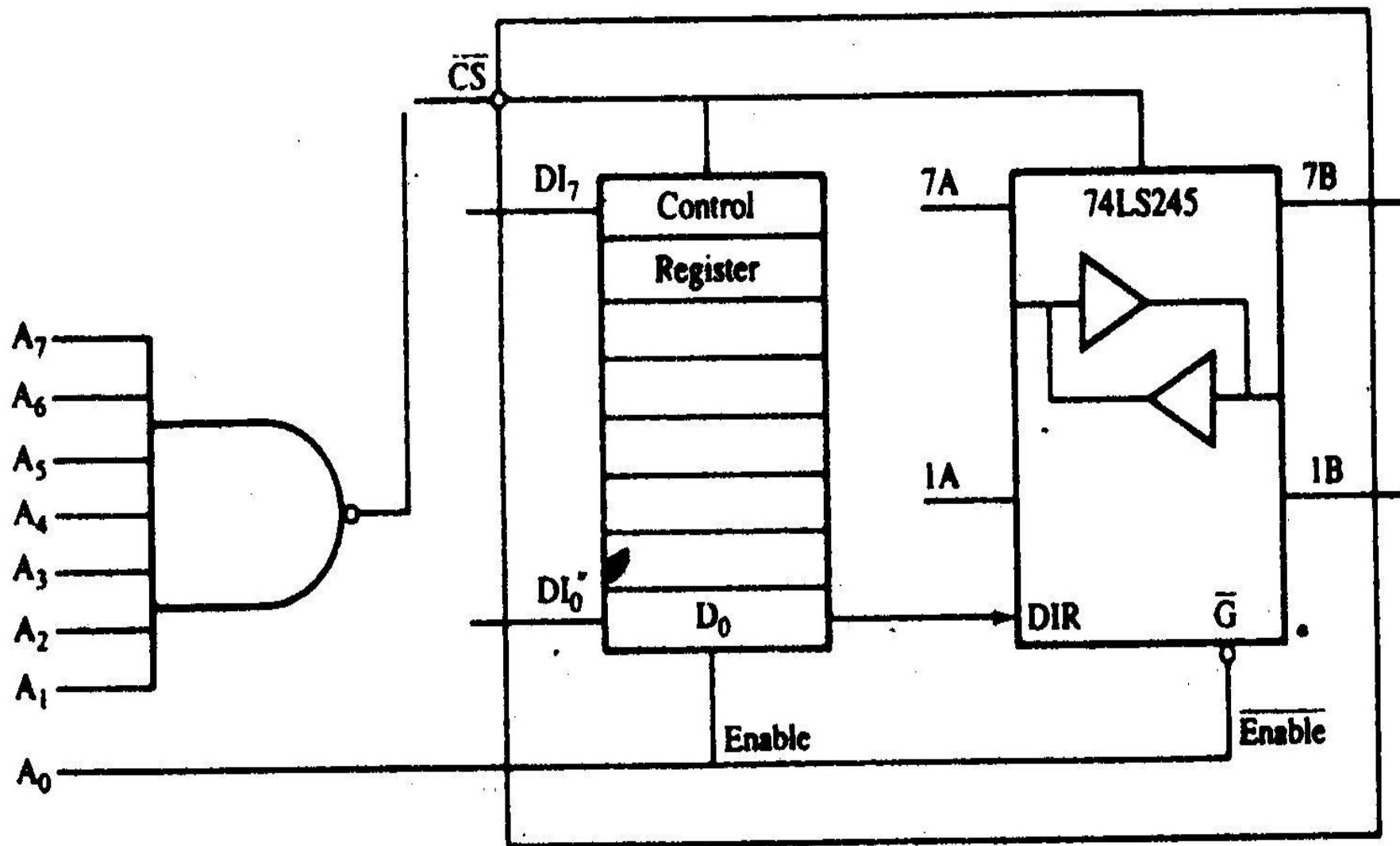
MVI A,01H
OUT FFH
MVI A, BYTE1
OUT FEH

FIGURE 14.2
Making 74LS245 Programmable

14.1.2 Programmable Device with a Statues Register

Figure 14.3 shows a hypothetical programmable device with a status register. There are three registers and can be accessed as ports. The port addresses and the functions of these registers can be summarized as follows:

Control register (output only) = FFH (A_1 and $A_0 = 1$)
   Status register (input only) = FFH (A_1 and $A_0 = 1$)
Data register (input or output) = FFH ($A_1 = 1$, $A_0 = 0$)

14.1.3 Programmable Device with a Handshake Signals

The MPU and peripherals operate at different speeds; therefore, signals are exchanged prior to data transfer between fast-responding MPU and slow-responding peripherals such as printers and data converters. These signals are called **handshake signals**. These signals are generally provided by programmable devices.

Figure 14.4(a) shows a programmable device in the input mode, with two handshake signals: STB (Strobe) and IBF (Input Buffer Full) and one interrupt signal (INTR). Figure 14.4(b) shows the programmable device in the output mode using the same handshake signals, except that they are labeled differently: OBF (Output Buffer Full) and ACK (Acknowledgement).
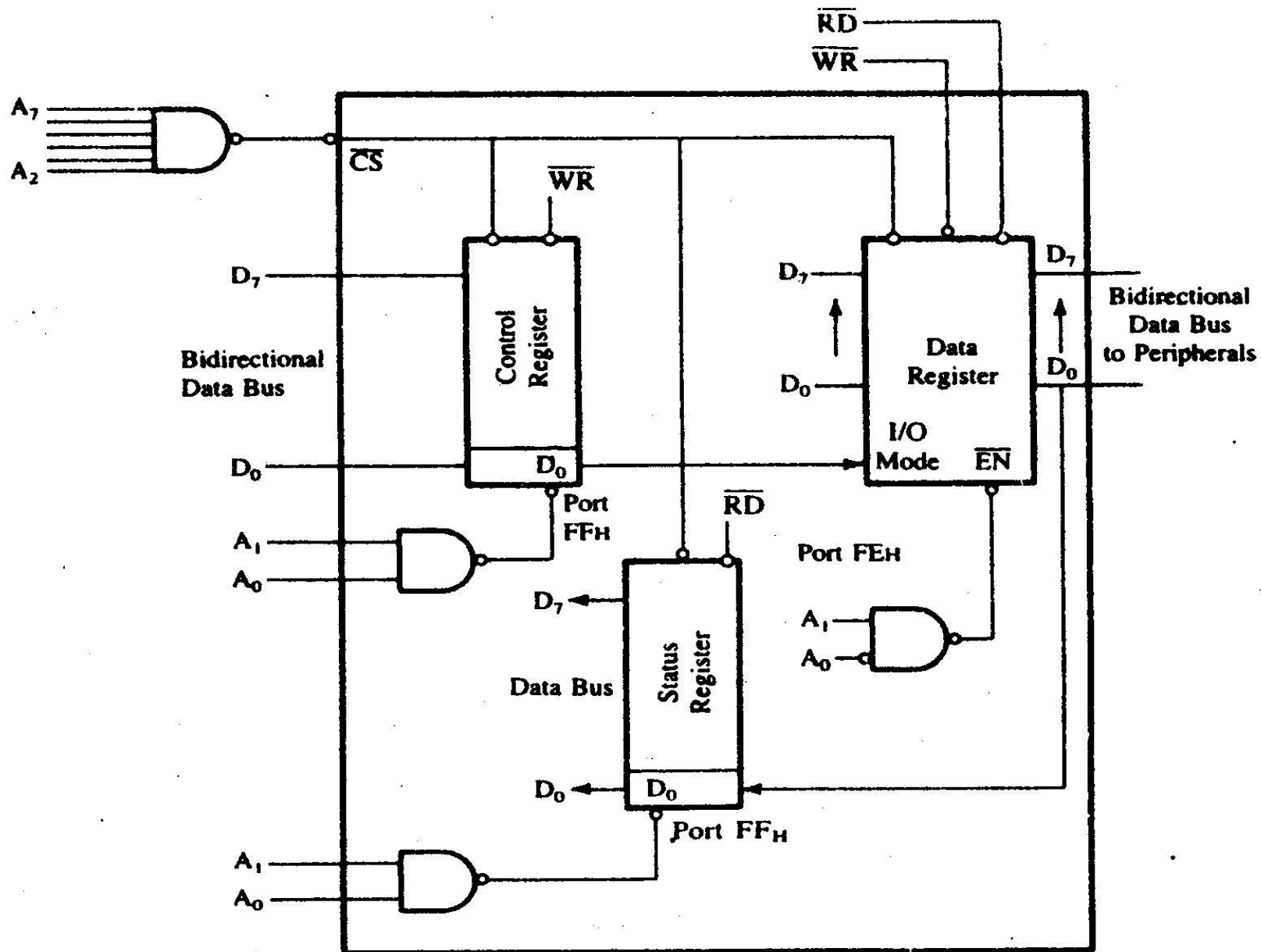
FIGURE 14.3
A Hypothetical Programmable Device with a Status Register
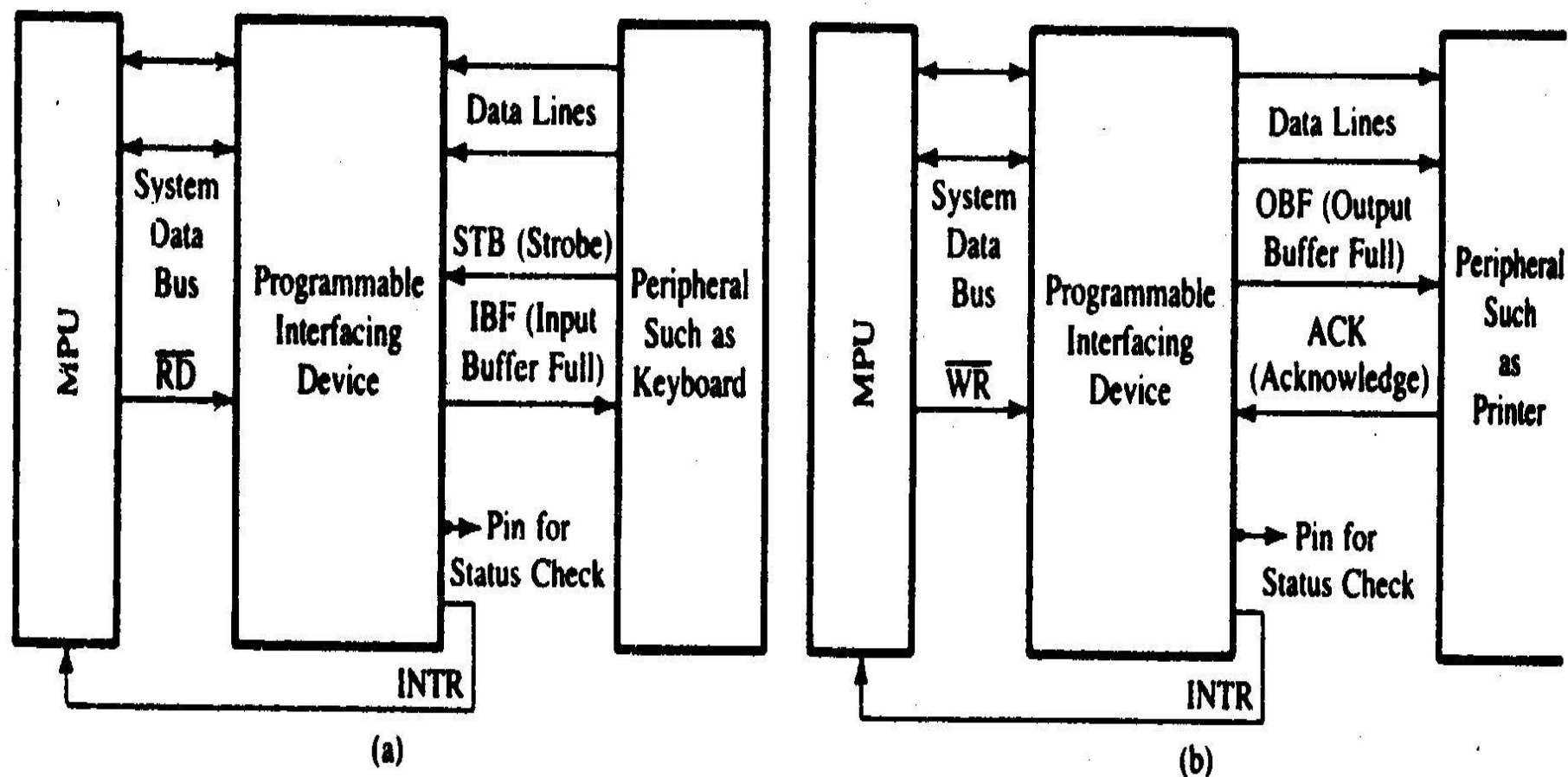
FIGURE 14.4

Interfacing Device with Handshake Signals for Data Input (a) and Data Output (b)

Examination of Figure 14.4 shows the following similarities among the handshake signals:

1. Handshake signals ACK and STB are input signals to the device and perform similar functions, although they are called by different names.

2. Handshake signals OBF and IBF are output signals from the device and perform similar functions (Buffer Full).

14.2 THE 8155: MULTIPURPOSE PROGRAMMABLE DEVICE

The 8155 is a multipurpose programmable device specially designed to be compatible with the 8085 microprocessor. The ALE, $IO/\overline{M}$, $\overline{RD}$, and $\overline{WR}$ signals from the 8085 can be connected directly to the device; this eliminates the need for external demultiplexing of the low-order bus $AD_7 - AD_0$ and generation of the control signals such as $\overline{MEMR}$, $\overline{MEMW}$, $\overline{IOR}$, and $\overline{IOW}$.

The 8155 includes 256 bytes of R/W memory, three I/O ports, and a timer. Figure 14.5 shows the pin configuration and block diagram of the 8155 programmable device. The programmable I/O sections of this device are illustrated in the following sections:
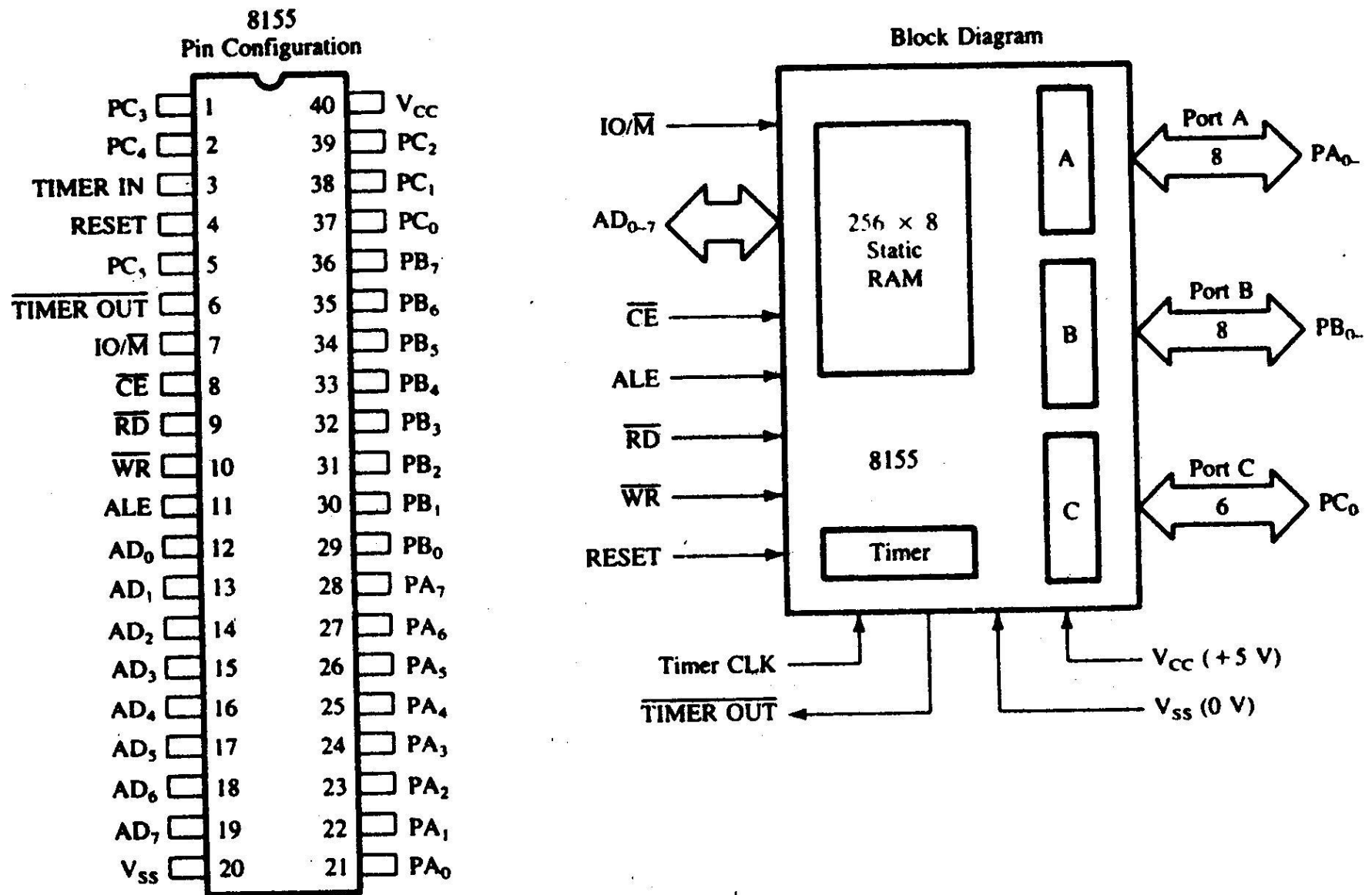
8155
Pin Configuration

Pin		Pin	
PC_3	1	40	V_{CC}
PC_4	2	39	PC_2
TIMER IN	3	38	PC_1
RESET	4	37	PC_0
PC_5	5	36	PB_7
$\overline{\text{TIMER OUT}}$	6	35	PB_6
$IO/\overline{M}$	7	34	PB_5
$\overline{CE}$	8	33	PB_4
$\overline{RD}$	9	32	PB_3
$\overline{WR}$	10	31	PB_2
ALE	11	30	PB_1
AD_0	12	29	PB_0
AD_1	13	28	PA_7
AD_2	14	27	PA_6
AD_3	15	26	PA_5
AD_4	16	25	PA_4
AD_5	17	24	PA_3
AD_6	18	23	PA_2
AD_7	19	22	PA_1
V_{SS}	20	21	PA_0

Block Diagram

$IO/\overline{M}$

AD_{0-7}

$\overline{CE}$

ALE

$\overline{RD}$

$\overline{WR}$

RESET

256×8 Static RAM

8155

Timer

A — Port A 8 — PA_{0-}

B — Port B 8 — PB_{0-}

C — Port C 6 — PC_0

Timer CLK

$\overline{\text{TIMER OUT}}$

V_{CC} (+5 V)

V_{SS} (0 V)

FIGURE 14.5

8155 Pin Configuration and Block Diagram

Source: Intel Corporation, *Embedded Microprocessors* (Santa Clara, Calif.: Author, 1994), pp. 1–31.

14.2 The 8155 Programmable I/O Ports and Timer

The 8155 is a device with two sections: the first is a 256 bytes R/W memory, and the second is a programmable I/O. Functionally, these two sections can be viewed as two independent chips.

The I/O section includes two 8-bit parallel I/O ports (A and B), one 6-bit port (C), and a timer (Figure 14.5). All the ports can be configured simply as input/output ports. Ports A and B also can be programmed in the handshake mode, each port using three signals from port C. The timer is a 14-bit down-counter and has four modes.

THE 8155 I/O PORTS

The I/O section of the 8155 includes a control register, three I/O ports, and two registers for the timer (Figure 14.6).

To communicate with the peripherals through the 8155 the following steps are necessary:

1. Determine the address (port numbers of the registers and I/Os) based on the Chip Enable logic and address lines AD_0, AD_1, and AD_2.
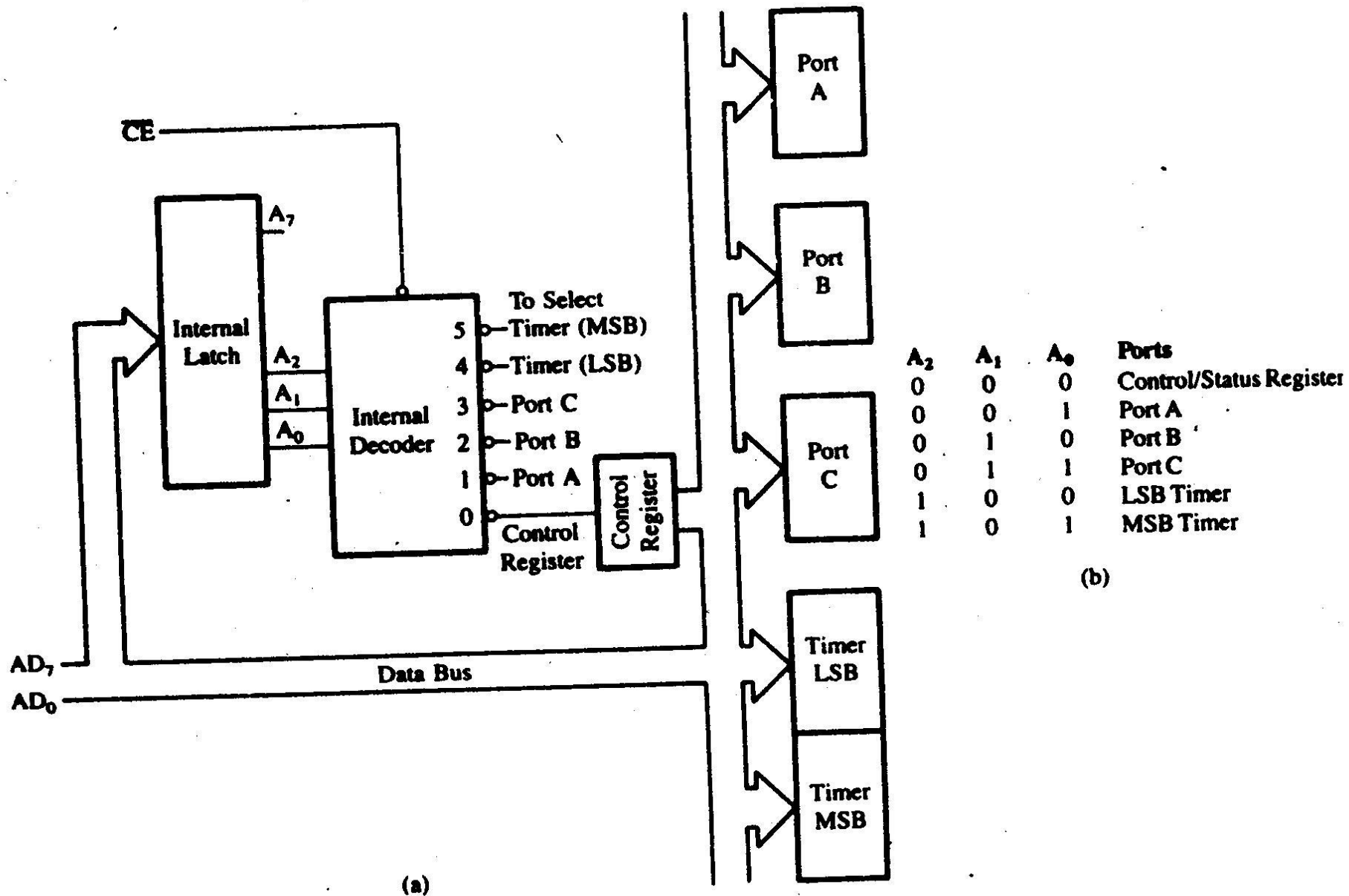
FIGURE 14.6
Expanded Block Diagram of the 8155 (a) and Its I/O Address: Selection (b)

2. Write control word in the control register to specify I/O functions of the ports and the timer characteristics.

3. Write I/O instructions to port addresses to communicate with peripherals.

4. Read the status register, if necessary, to verify the status of the I/O ports and the timer. In simple applications, this step is not necessary.

CHIP ENABLE LOGIC AND PORT ADDRESSES

Address lines $AD_2 - AD_0$ also shown as $A_2 - A_0$ after internal demultiplexing, select one of the registers, as shown in Figure 14.6(b). Address lines $A_3 - A_7$ are don't care lines; however, the logic levels on the corresponding high-order lines, $A_{11} - A_{15}$, will be duplicated on lines $A_3 - A_7$.

Example 14.2: Determine the addresses of the control/status register, I/O ports, and timer registers in Figure 14.7.

Solution: To select the chip, the output line of the 8205 (3-to-8) decoder (Figure 14.7) should go low:

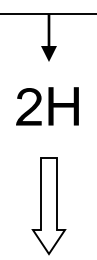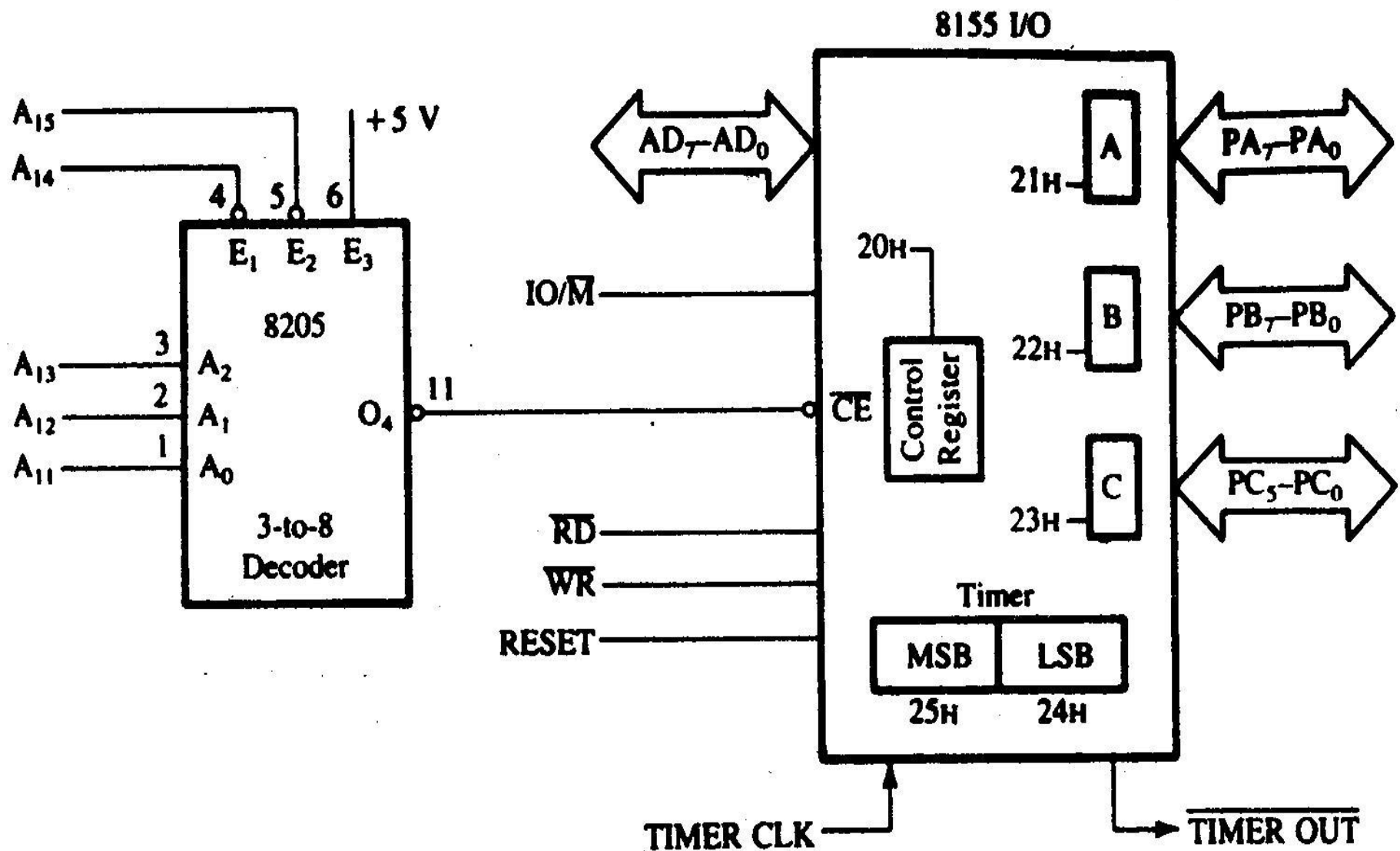| A_{15}	A_{14}	A_{13}	A_{12}	A_{11}
0	0	1	0	0

FIGURE 14.7

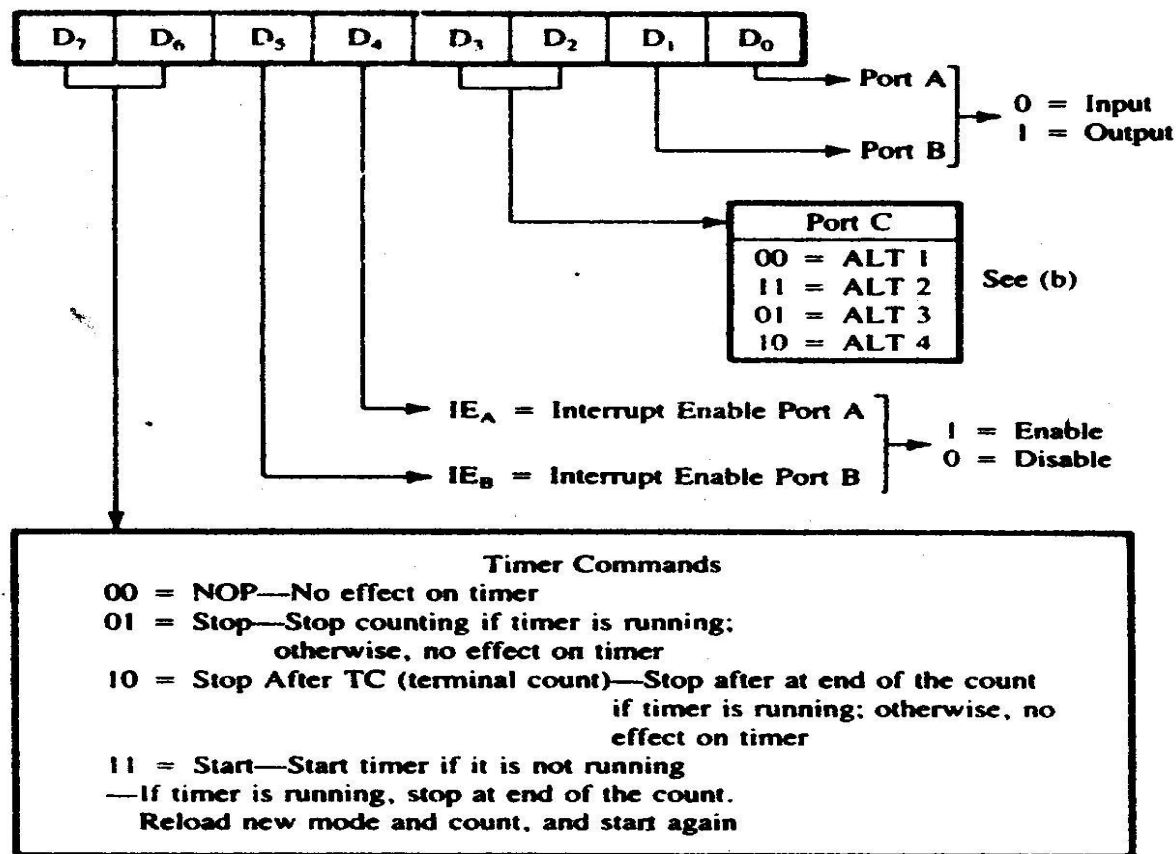Interfacing 8155 I/O Ports (Schematic from the SDK-85 System)

By combining five high-order address lines with three low-order address lines (A2 – A0), the port numbers in Figure 14.7 will range from 20H to 25H shown below:

A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	AD_2	AD_1	AD_0 = Address Ports
0	0	1	0	0	0	0	0 = 20H – CR or SR
					0	0	1 = 21H – Port A
		2H			0	1	0 = 22H – Port B
					0	1	1 = 23H – Port C
					1	0	0 =24H–Timer(LSB)
A_7	A_6	A_5	A_4	A_3	1	0	1 =25H–Timer(MSB)

N.B. For I/O-mapped I/O (or port-addressed I/O or peripheral I/O) the low-order address and high-order address bus carry the same information.

CONTROL WORD

The I/O ports and the timer can be configured by writing a control word in the control register. The control register bits are defined as shown in Figure 14.8.

Timer Commands

$00 = $ NOP—No effect on timer
$01 = $ Stop—Stop counting if timer is running; otherwise, no effect on timer
$10 = $ Stop After TC (terminal count)—Stop after at end of the count if timer is running; otherwise, no effect on timer
$11 = $ Start—Start timer if it is not running
—If timer is running, stop at end of the count. Reload new mode and count, and start again

(a)

Port C
$00 = $ ALT 1
$11 = $ ALT 2
$01 = $ ALT 3
$10 = $ ALT 4
See (b)

$IE_A = $ Interrupt Enable Port A
$IE_B = $ Interrupt Enable Port B
$1 = $ Enable
$0 = $ Disable

Port A
Port B
$0 = $ Input
$1 = $ Output

Table: ALT 1–ALT 4: Port C Bit Assignments, Defined by Bits D_3 and D_2 in the Control Register

ALT	D_3	D_2	PC_5	PC_4	PC_3	PC_2	PC_1	PC_0
ALT 1	0	0	I	I	I	I	I	I
ALT 2	1	1	O	O	O	O	O	O
ALT 3	0	1	O	O	O	$\overline{STB}_A$	BF_A	$INTR_A$
ALT 4	1	0	$\overline{STB}_B$	BF_B	$INTR_B$	$\overline{STB}_A$	BF_A	$INTR_A$

$I = $ Input, $STB = $ Strobe, $INTR = $ Interrupt Request
$O = $ Output, $BF = $ Buffer Full, Subscript A $= $ Port A
B $= $ Port B

(b)

FIGURE 14.8
Control Word Definition in the 8155 (a) and Table of Port C Bit Assignments (b)

Bit D_2 and D_3 determine the functions of port C; their combination specifies one of the four alternatives, from simple I/O to interrupt I/O, as shown in Figure 14.8(b). Bits D_4 and D_5 are used only in the interrupt mode to enable or disable internal flip-flops of the 8155. These bits do not have any effect on the Internal Enable (INTE) flip-flop of the MPU.

14.2.2 Illustration: <u>Interfacing Sven Segment LED Output Ports Using the 8155</u>

<u>PROBLEM STATEMENT</u>

1. Design two seven-segment-LED displays using ports A and B of the 8155.
2. Write initialization instructions and display data bytes at each port.

<u>HARDWARE DESCRIPTION</u>

The decode logic is the same as that used in the previous discussion; therefore, the port addresses are as follows:

        Control Register = 20H
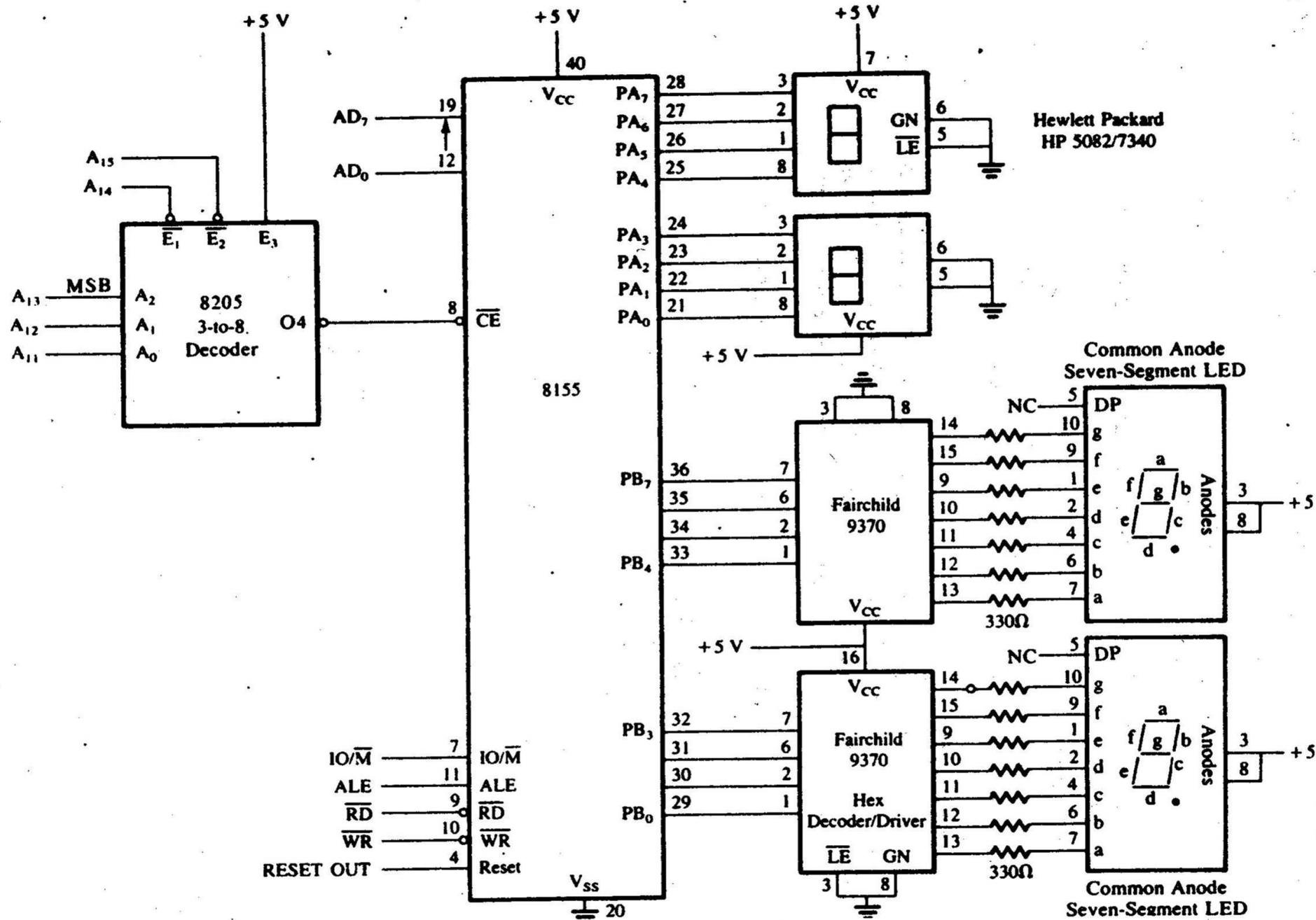   Port A   = 21H
        Port B   = 22H

FIGURE 14:9
Interfacing 8155 I/O Ports with Seven-Segment LEDs

CONTROL WORD

To configure ports A and B as outputs, the control word is as follows:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0		
0	0	0	0	0	0	1	1	=	03H

PROGRAM

```
MVI A, 03H
OUT 20H
MVI A, BYTE1
OUT 21H
MVI A,BYTE2
OUT 22H
HLT
```

14.2.3 <u>The 8155 Timer</u>

The timer section of the 8155 has two 8-bit registers; 14 bits are used for the counter, two bits for the timer mode, and it requires a clock as an input. This 14-bit down counter provides output in four different modes, as described in Figure 14.(b).
The timer can be stopped either in the midst of counting or at the end of a count (applicable to Modes 1 and 3).

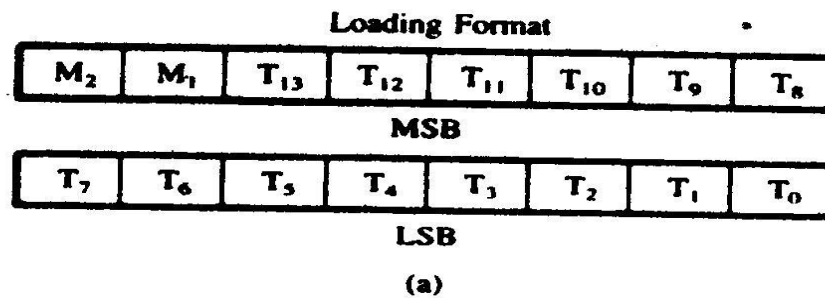### 14.2.4 Illustration: <u>Designing a Square-Wave Generator Using the 8155 Timer</u>

<u>PROBLEM STATEMENT</u>

Design a square-wave generator with a pulse width of 100 µs by using the 8155 timer. Set up timer in Mode 1 if the clock frequency is 3 MHz. Use the same decode logic and the port addresses as in Example 14.2 (Figure 14.7).

<u>PROBLEM ANALYSIS</u>

Timer Count: The pulse width required is 100 µs; therefore, the count should be calculated for the period of 200 µs. The timer output stays high for only half the count.
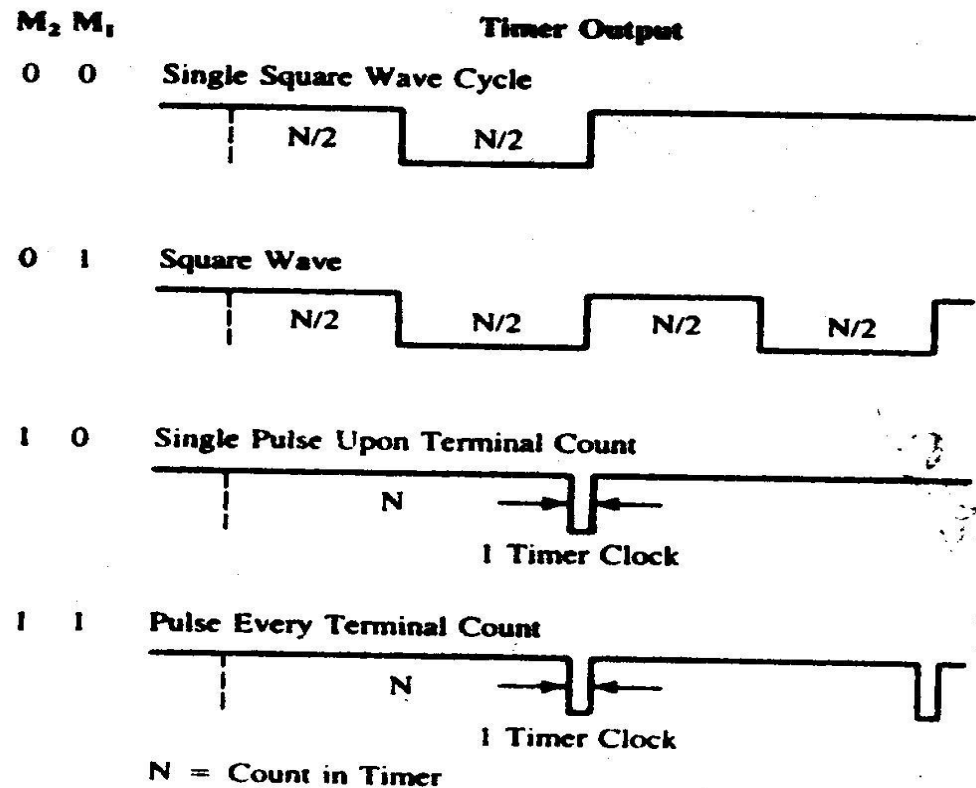
Loading Format

M_2	M_1	T_{13}	T_{12}	T_{11}	T_{10}	T_9	T_8

MSB

T_7	T_6	T_5	T_4	T_3	T_2	T_1	T_0

LSB

(a)

Modes | M_2 M_1 | **Timer Output**

Mode 0: In this mode, the timer output remains high for half the count and goes low for the remaining count, thus providing a single square wave. The pulse width is determined by the count and the clock frequency.

0 0 Single Square Wave Cycle

Mode 1: In this mode, the initial timer count is automatically reloaded at the end of each count, thus providing a continuous square wave.

0 1 Square Wave

Mode 2: In this mode, a single clock pulse is provided at the end of the count.

1 0 Single Pulse Upon Terminal Count

Mode 3: This is similar to Mode 2, except the initial count is reloaded to provide a continuous wave form.

1 1 Pulse Every Terminal Count

N = Count in Timer

(b)

FIGURE 14.10
Timer Loading Format (a) and Modes (b)

Clock Period = 1/f = 1/3 x 10^6 = 330 ns

Timer Count = Pulse Period / Clock Period = 200 x 10^{-6} / 330 x 10^{-9} = 606

Count = 025EH

The port addresses for the timer registers are

Timer LSB = 24H

Timer MSB = 25H

The least significant byte, 5EH (of the count 025EH), should be loaded in the timer register with address 24H. The most significant byte is determined as follows:

M_2	M_1	T_{13}	T_{12}	T_{11}	T_{10}	T_9	T_8	
0	1	0	0	0	0	1	0	= 42H

Therefore, 42H should be loaded in the timer register with the address 25H.

CONTROL WORD

Ports A and B are used as output port; therefore, the control word is:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
1	1	0	0	0	0	1	1	= C3H

Initialization Instructions

MVI A, 5EH
OUT 24H
MVI A,42H
OUT 25H
MVI A, C3H
OUT 20H
HLT

14.2.5   The 8155 I/O Ports in Handshake Mode

In the handshake mode, data transfer occurs between the MPU and peripherals using control signals called handshake signals. Two I/O ports of the 8155, A and B, can be configured in the handshake mode; each uses three signals from port C as control signals (Figure 14.11). Another alternative (ALT3 in the table in Figure 14.8) available in the 8155 is to configure port A in the handshake mode with three control signals from port C, configure port B as simple I/O, and configure the remaining three bits from port C as outputs.

CONTROL SIGNALS IN HANDSHAKE MODE

When both ports A and B are configured in the handshake mode, port A uses the lower three signals of port C (PC_0, PC_1, and PC_2), and port B uses the upper three signals (PC_3, PC_4, and PC_5), as shown in Figure 14.11. The function of these signals are:

❑ **STB (Strobe Input):** The low signal on this pin informs the 8155 that data are strobed into the input port.

❑ **BF (Buffer Full):** The high signal on this pin indicates the presence of a data byte in the port.

FIGURE 14.11
8155 with Handshake Mode
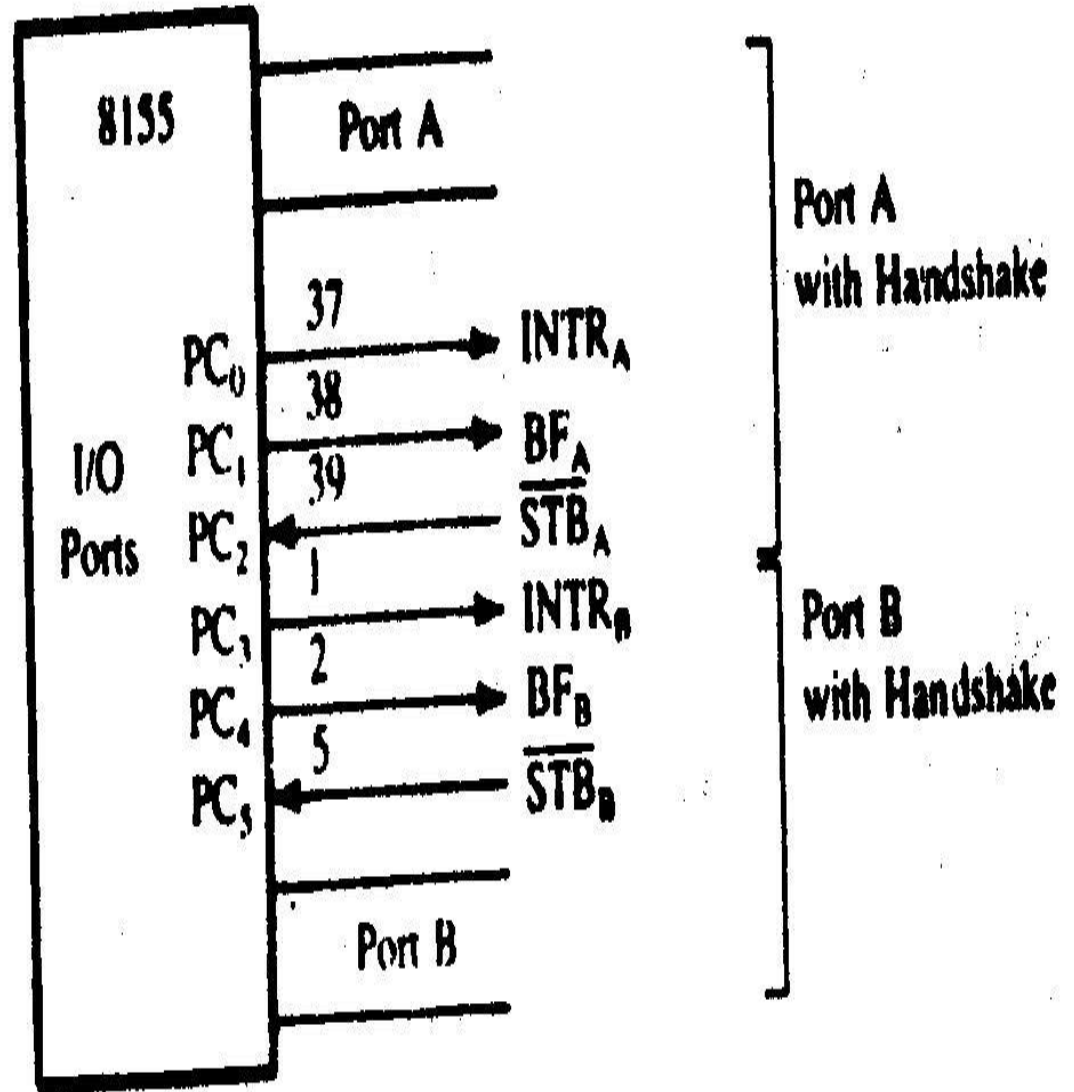
❑ **INTR (Interrupt Request):** This signal interrupts the processor to read an existing data from its (8155) input port or write a new data to its output port.

❑ **INTE (Interrupt Enable) :** This is an internal flip-flop used to enable or disable the interrupt capability of the 8155. The interrupts for port A and B are controlled by bits D_4 and D_5 respectively, in the control register.

These control signals can be used to implement either interrupt I/O or status check I/O.

INPUT

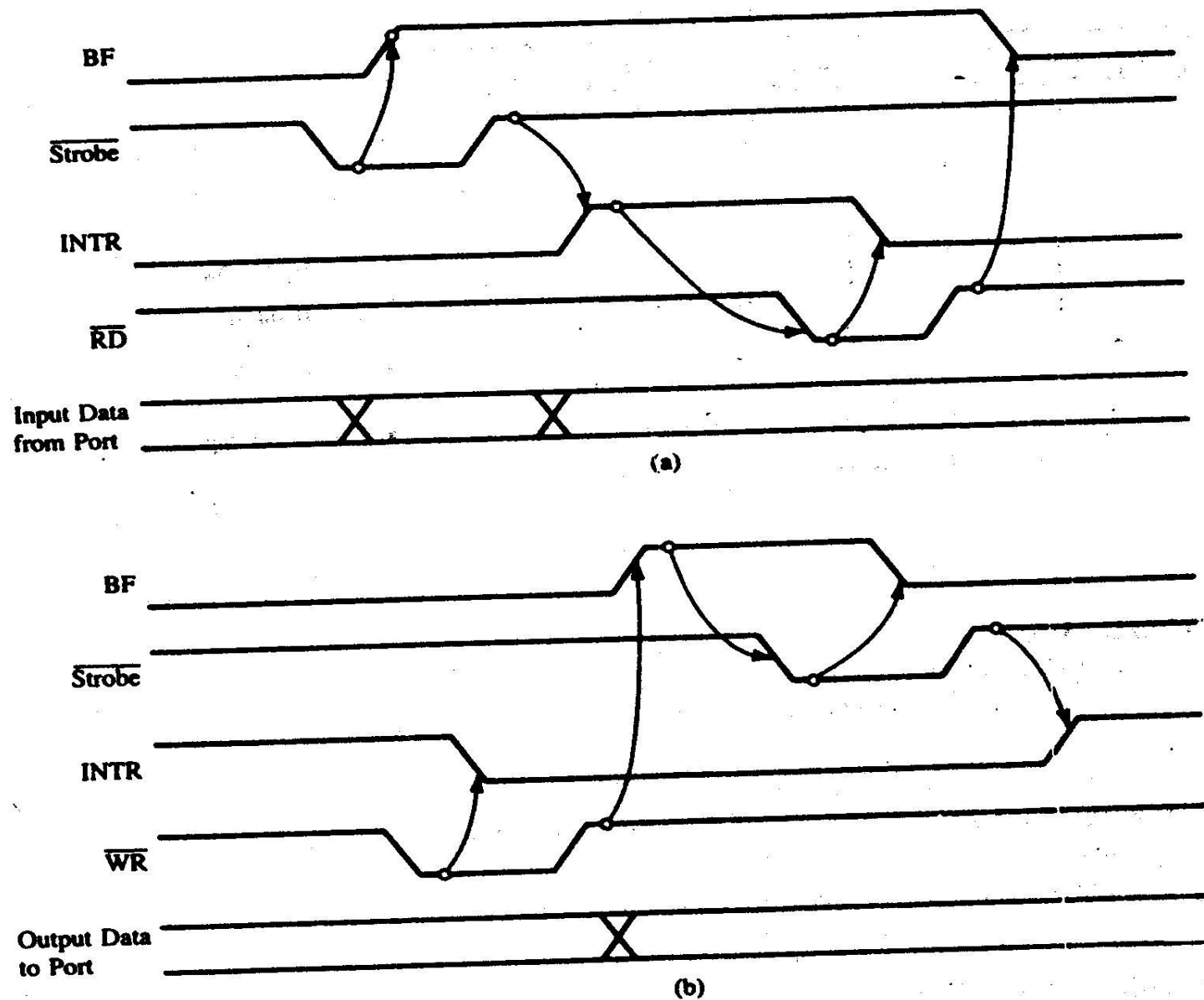Figure 14.12 (a) shows the sequence of events and timing in data input to the 8155.

1.
2.
3.
4.

FIGURE 14.12
Timing Waveforms of the 8155 I/O Ports with Handshake: Input Mode (a) and Output Mode (b)

Embedded Microprocessors, (Santa Clara, Calif.: Author, 1994), pp. 1–43.

<u>OUTPUT</u>

Figure 14.12 (b) shows the sequence of events and timing in data output to the 8155.

1.
2.
3.
4.

<u>STATUS WORD</u>

The MPU can read the status register to check of the timer. The control register and the status register have the same port address; they are differentiated only by RD and WR signals. The status register bits are defined in Figure 14.13.
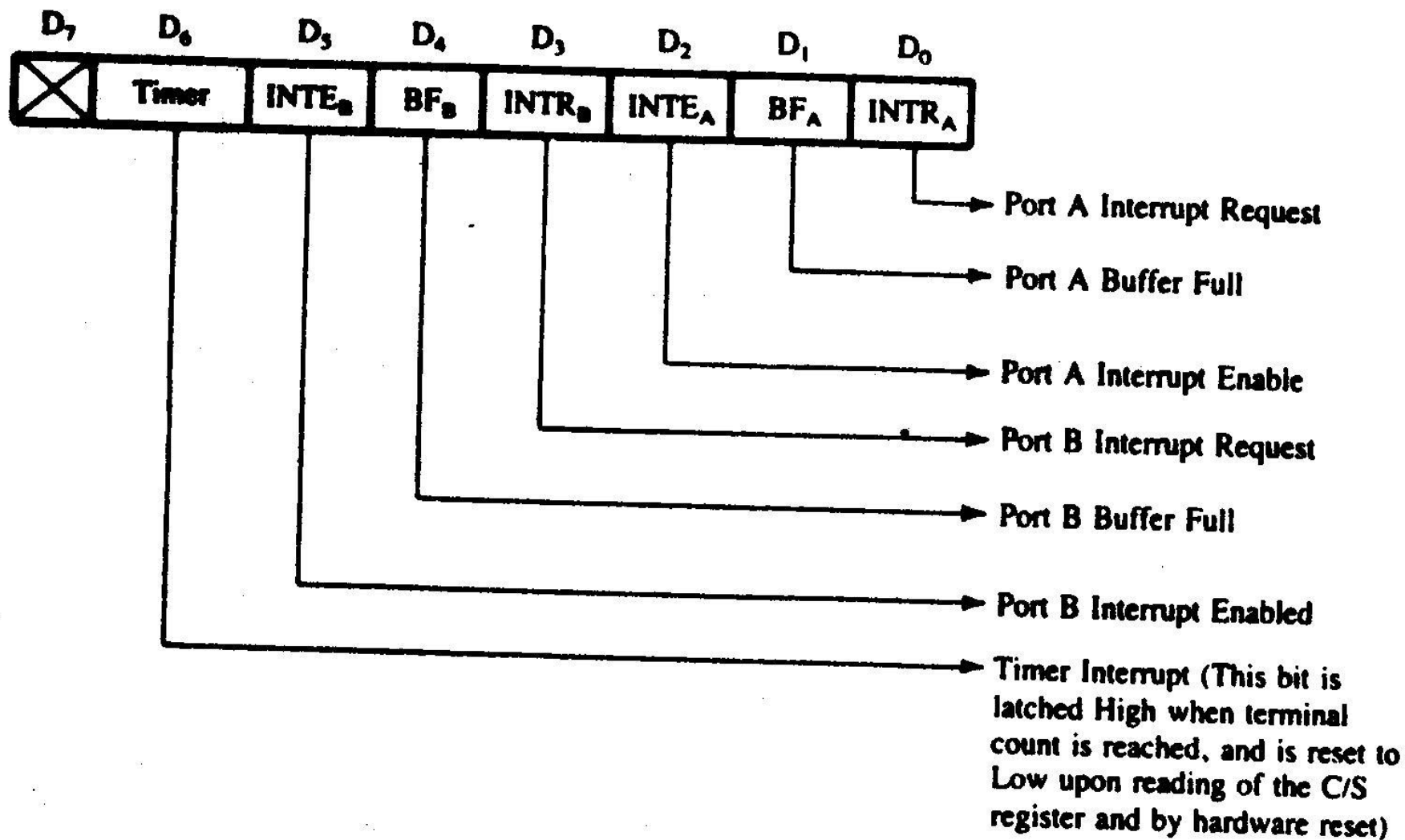
FIGURE 14.13

Status Word Definition

14.2.6 <u>Illustration</u>: <u>Interfacing I/O Ports in Handshake Mode using the 8155</u>

<u>PROBLEM STATEMENT</u>

Design an interfacing circuit using the 8155 read and display form an A/D converter to meet the following requirements:

1. Set up port A in the handshake mode to read data from A/D converter.
2. Set up port B as an output port to display data at seven-segment LEDs.
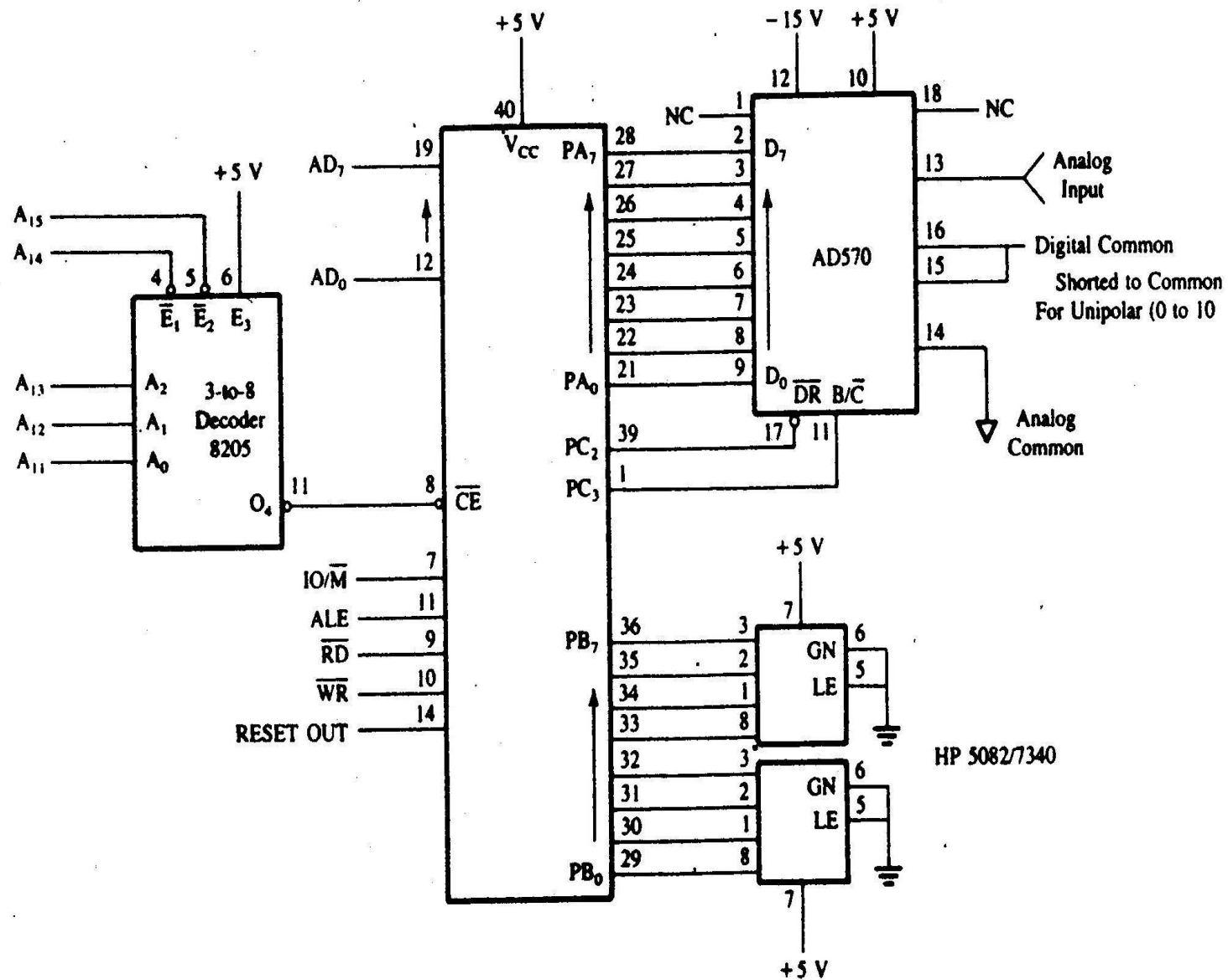3. Use line PC_3 from port C to initiate a conversion.

FIGURE 14.14
Interfacing the A/D Converter AD570 in the Handshake Mode

PROBLEM ANALYSIS

The circuit shows that the INTR signal (bit PC_o) is not being used. This suggests that port A is configured for status check and not for interrupt I/O. Therefore, the control word:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
0	0	0	0	0	1	1	0	= 06H

8155 Timer

To start the timer, the control word:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
1	1	0	0	0	1	1	0	= C6H

To stop the timer, the control word:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
0	1	0	0	0	1	1	0	= 46H