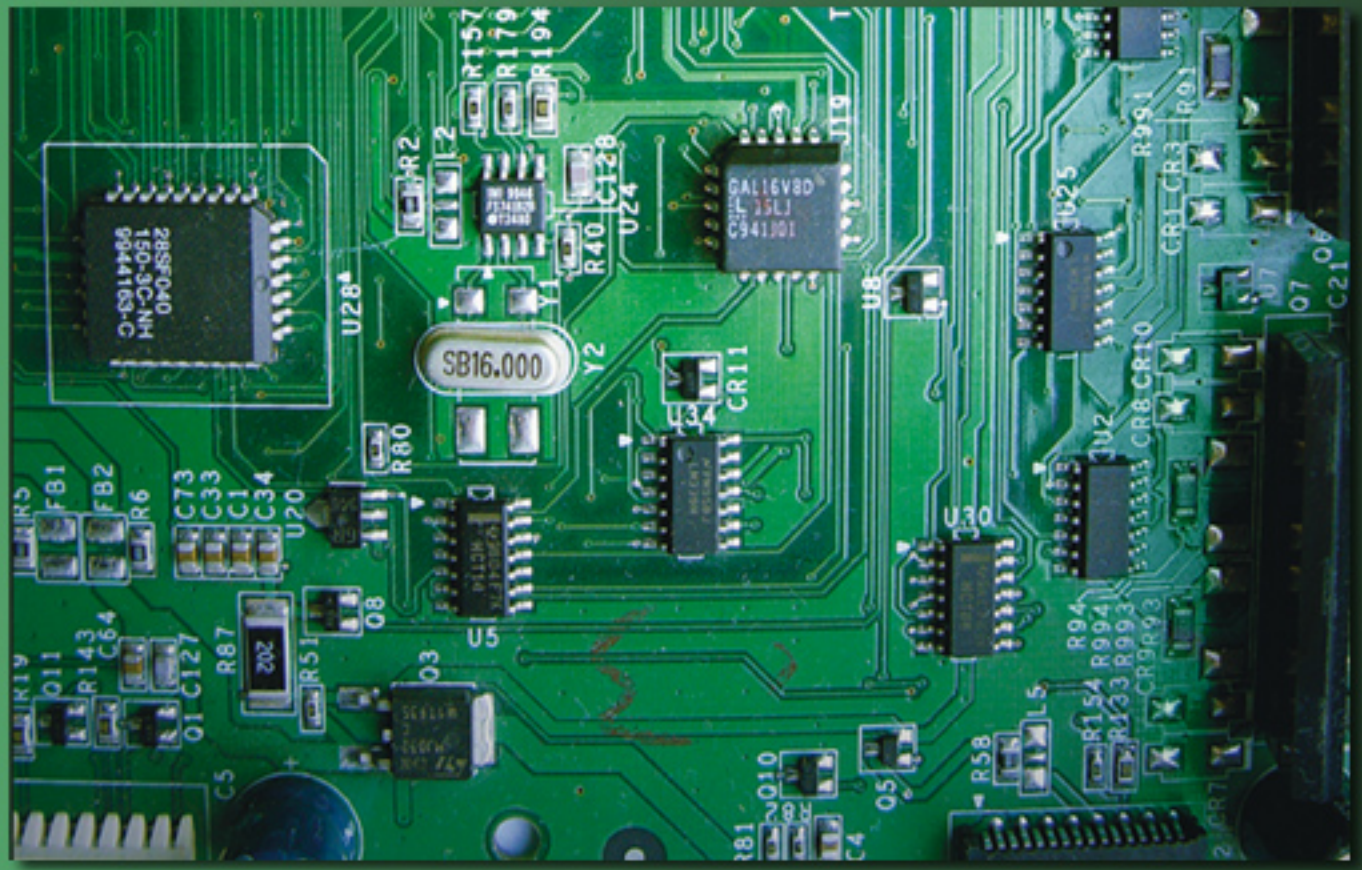
The Intel Microprocessors

8086/8088, 80186/80188, 80286, 80386, 80486 Pentium, Pentium Pro Processor, Pentium II, Pentium 4, and Core2 with 64-bit Extensions

Architecture, Programming, and Interfacing



EIGHTH EDITION

Barry B. Brey

PEARSON

Chapter 11: Basic I/O Interface

Introduction

- This chapter outlines some of the basic methods of communications, both serial and parallel, between humans or machines and the microprocessor.
- We first introduce the basic I/O interface and discuss decoding for I/O devices.
- Then, we provide detail on parallel and serial interfacing, both of which have a variety of applications.

Chapter Objectives

Upon completion of this chapter, you will be able to:

- Explain the operation of the basic input and output interfaces.
- Decode an 8-, 16-, and 32-bit I/O device so that they can be used at any I/O port address.
- Define handshaking and explain how to use it with I/O devices.
- Interface and program the 82C55 programmable parallel interface.

Chapter Objectives (*cont.*)

Upon completion of this chapter, you will be able to:

- Interface LCD displays, LED displays, keyboards, ADC, DAC, and various other devices to the 82C55.
- Interface and program the 16550 serial communications interface adapter.
- Interface and program the 8254 programmable interval timer.

Chapter Objectives (*cont.*)

Upon completion of this chapter, you will be able to:

- Interface an analog-to-digital converter and a digital-to-analog converter to the microprocessor.
- Interface both DC and stepper motors to the microprocessor.

11–3 THE PROGRAMMABLE PERIPHERAL

- 82C55 **programmable peripheral interface** (**PPI**) is a popular, low-cost interface component found in many applications.
- The PPI has 24 pins for I/O, programmable in groups of 12 pins and groups that operate in three distinct modes of operation.
- 82C55 can interface any TTL-compatible I/O device to the microprocessor.

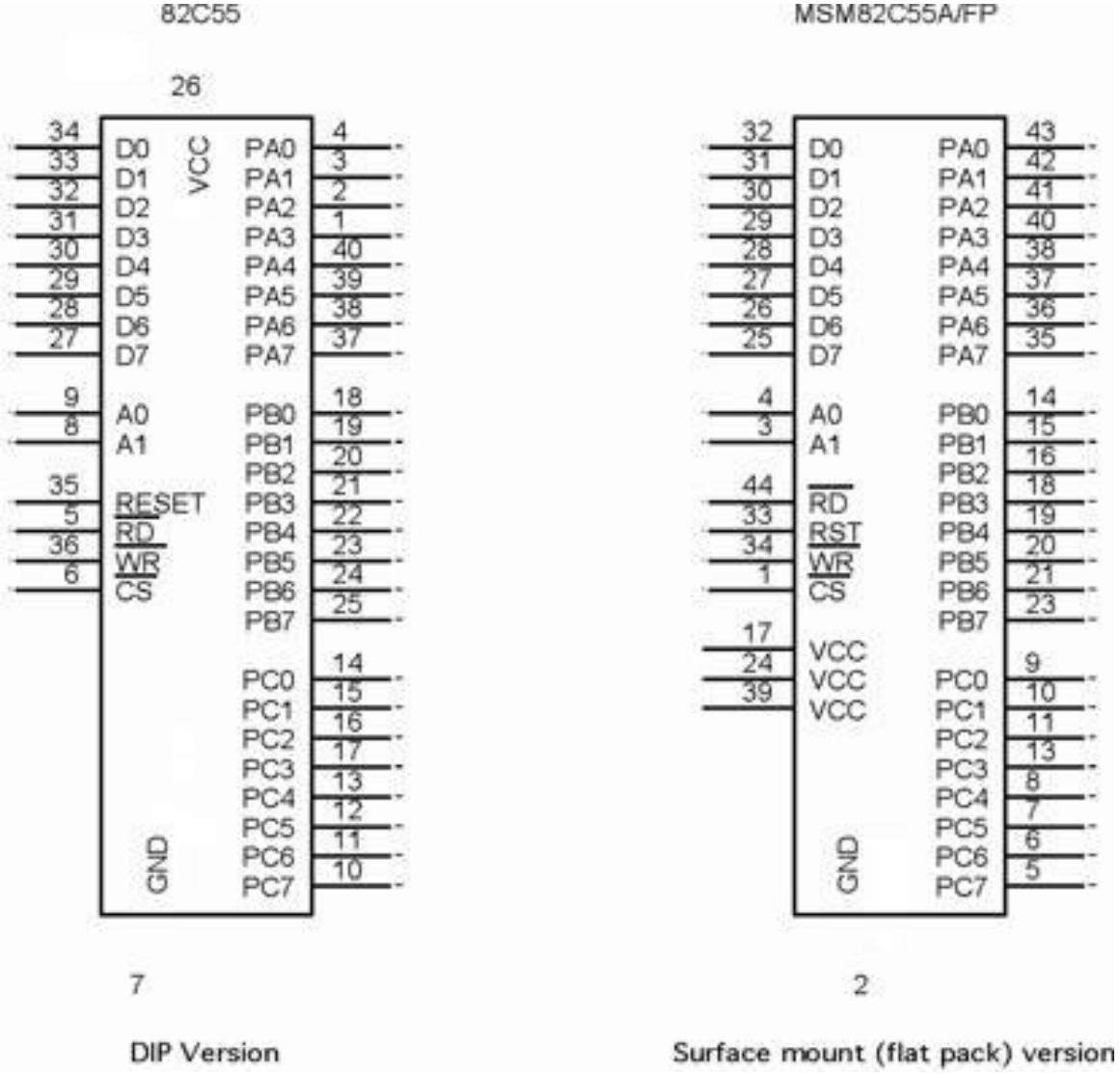
- The 82C55 (CMOS version) requires wait states if operated with a processor using higher than an 8 MHz clock.
- Because I/O devices are inherently slow, wait states used during I/O transfers do not impact significantly upon the speed of the system.
- The 82C55 still finds application even in the latest Core2-based computer system.

- 82C55 is used for interface to the keyboard and parallel printer port in many PCs.
  - found as a function within an interfacing chip set
  - also controls the timer and reads data from the keyboard interface
- An experimentation board is available that plugs into the parallel port of a PC, to allow access to an 8255 located on the board.
- The 8255 is programmed in either assembly language or Visual C++ through drivers available with the board.

Basic Description of the 82C55

- Fig 11–18 shows pin-outs of the 82C55 in DIP and surface mount (flat pack) format.
- The three I/O ports (labeled A, B, and C) are programmed as groups.
  - group A connections consist of port A (PA_7–PA_0) and the upper half of port C (PC_7–PC_4)
  - group B consists of port B (PB_7–PB_0) and the lower half of port C (PC_3–PC_0)
- 82C55 is selected by its $\overline{CS}$ pin for programming and reading/writing to a port.

Figure 11–18 The pin-out of the 82C55 peripheral interface adapter (PPI).



- Table 11–2 shows I/O port assignments used for programming and access to the I/O ports.
- In the PC, a pair of 82C55s, or equivalents, are decoded at I/O ports 60H–63H and also at ports 378H–37BH.
- The 82C55 is a fairly simple device to interface to the microprocessor and program.
- For 82C55 to be read or written, the $\overline{CS}$ input must be logic 0 and the correct I/O address must be applied to the A_1 and A_0 pins.
- Remaining port address pins are *don't cares*.

- Fig 11–19 shows an 82C55 connected to the 80386SX so it functions at 8-bit addresses C0H (port A), C2H (port B), C4H (port C), and C6H (command register).
  - this interface uses the low bank of the I/O map
- All 82C55 pins are direct connections to the 80386SX, except the $\overline{CS}$ pin. The pin is decoded/selected by a 74ALS138 decoder.
- A RESET to 82C55 sets up all ports as simple input ports using mode 0 operation.
  - initializes the device when the processor is reset

Figure 11–19 The 82C55 interfaced to the low bank of the 80386SX microprocessor.

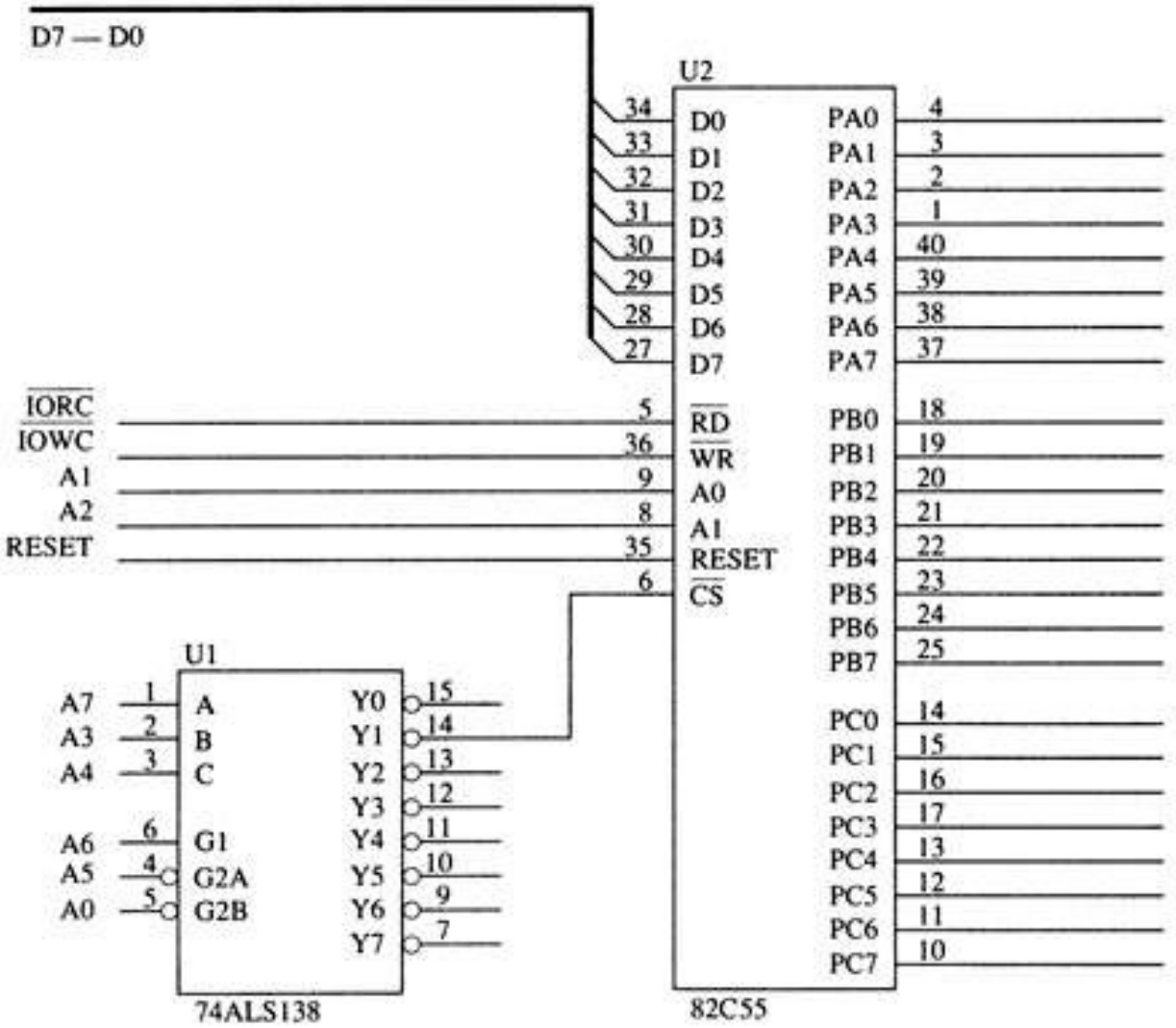


Table 11–2: I/O port assignments

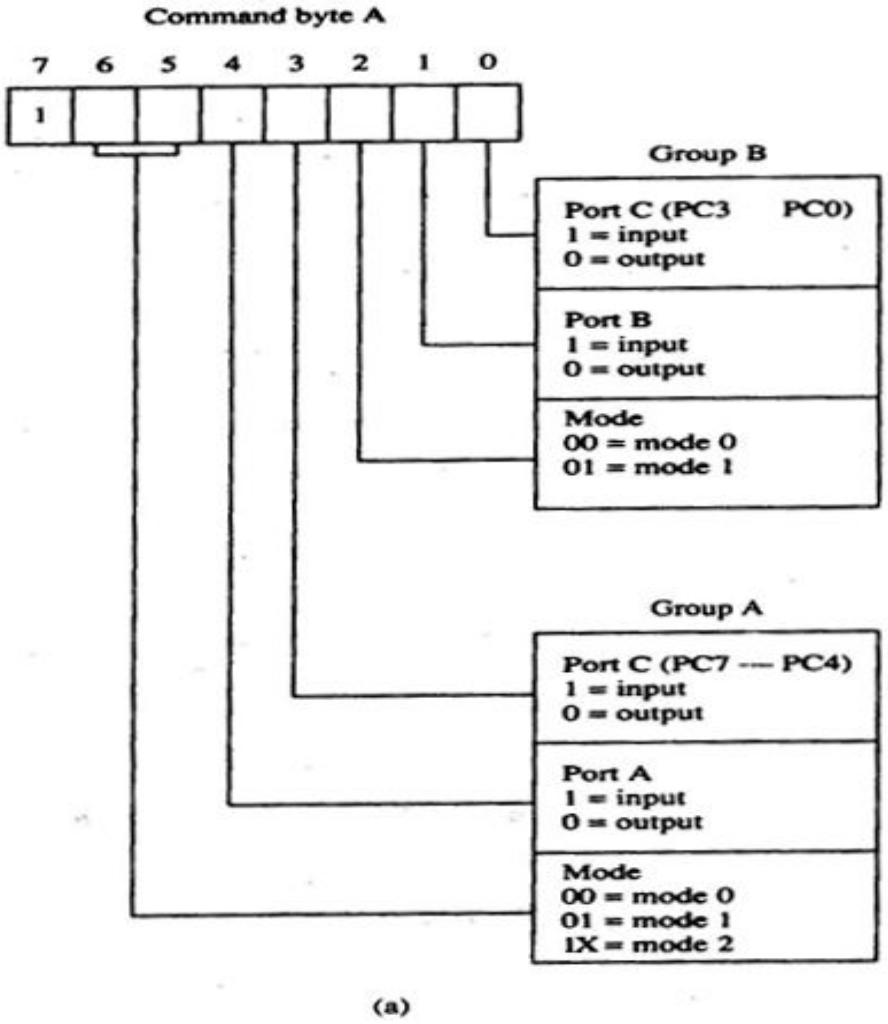
A_1	A_0	Function
0	0	Port A
0	1	Port B
1	0	Port C
1	1	Command Register

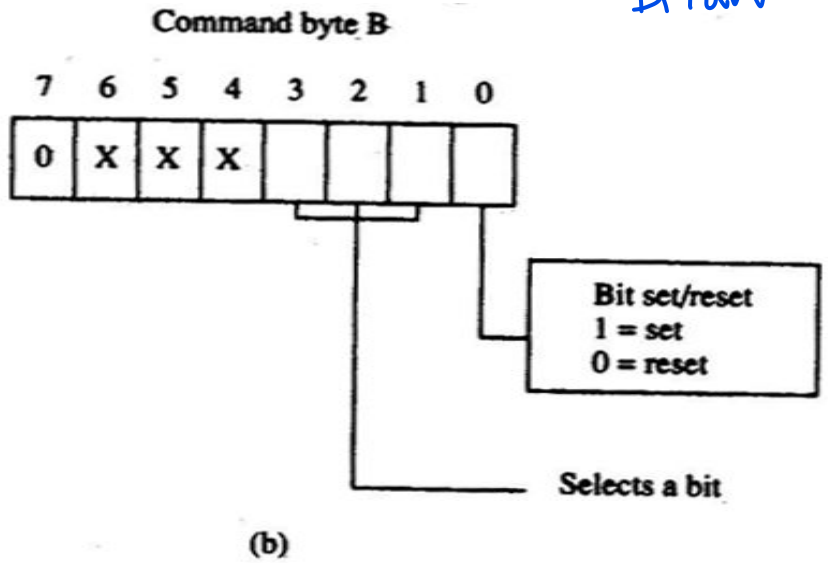
- After a RESET, no other commands are needed, as long as it is used as an input device for all three ports.
- 82C55 is interfaced to the PC at port addresses 60H–63H for keyboard control.
  - also for controlling the speaker, timer, and other internal devices such as memory expansion
- It is also used for the parallel printer port at I/O ports 378H–37BH.

Programming the 82C55

- 82C55 is programmed through two internal command registers shown in Figure 11–20.
- Bit position 7 selects either command byte A or command byte B.
  - command byte A programs functions of group A and B
  - byte B sets (1) or resets (0) bits of port C only if the 82C55 is programmed in mode 1 or 2
- Group B (port B and the lower part of port C) are programmed as input or output pins.

Figure 11–20 The command byte of the command register in the 82C55. (a) Programs ports A, B, and C. (b) Sets or resets the bit indicated in the select a bit field.





- group B operates in mode 0 or mode 1
- Mode 0 is basic input/output mode that allows the pins of group B to be programmed as simple input and latched output connections
- Mode 1 operation is the strobed operation for group B connections
- data are transferred through port B
- handshaking signals are provided by port C

- Group A (port A and the upper part of port C) are programmed as input or output pins.
- Group A can operate in modes 0, 1, and 2.
  - mode 2 operation is a bidirectional mode of operation for port A
- If a 0 is placed in bit position 7 of the command byte, command byte B is selected
- This allows any bit of port C to be set (1) or reset (0), if the 82C55 is operated in either mode 1 or 2.
  - otherwise, this byte is not used for programming

Mode 0 Operation

- Mode 0 operation causes 82C55 to function:
  - as a buffered input device
  - as a latched output device
- Fig 11–21 shows 82C55 connected to a set of eight seven-segment LED displays.
- These are standard LEDs.
  - the interface can be modified with a change in resistor values for an organic LED (OLED) display or high-brightness LEDs

Figure 11–21 An 8-digit LED display interfaced to the 8088 microprocessor through an 82C55 PIA.

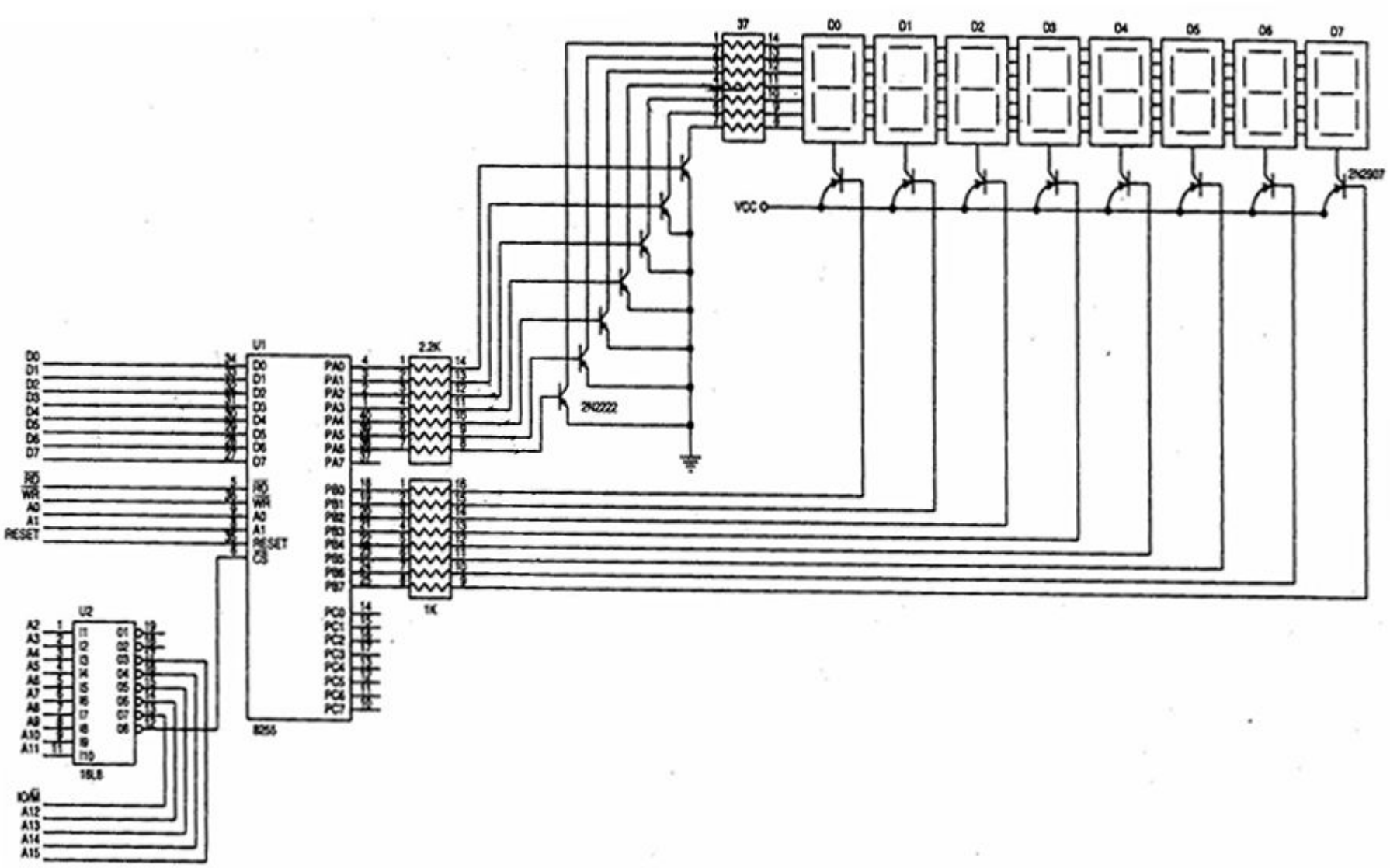
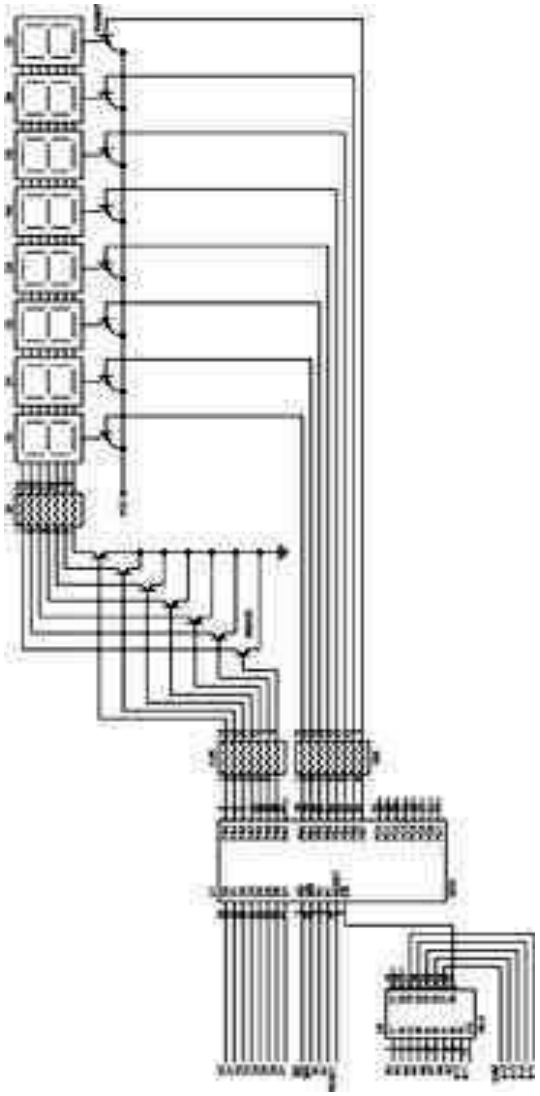


Figure 11–21 An 8-digit LED display interfaced to the 8088 microprocessor through an 82C55 PIA.



- ports A & B are programmed as (mode 0) simple latched output ports
- port A provides segment data inputs port B provides a means of selecting one display position at a time for multiplexing the displays
- the 82C55 is interfaced to an 8088 through a PLD so it functions at I/O port numbers 0700H–0703H
- PLD decodes the I/O address and develops the write strobe for the $\overline{WR}$ pin of the 82C55

- Programming the 82C55 is accomplished by the short sequence of instructions listed in Example 11–9.
- Ports A and B are programmed as outputs.

EXAMPLE 11–9

```
;programming the 82C55 PIA

 MOV    AL, 10000000B

     MOV    DX, 703H ;address command

         OUT     DX, AL
```

EXAMPLE 11–10

```
;An assembly language procedure that multiplexes the 8-digit display.
;This procedure must be called often for the display
;to appear correctly
DISP PROC NEAR USES AX BX DX SI
        PUSHF
        MOV BX, 8               ;load counter
        MOV AH, 7FH             ;load selection pattern
        MOV SI, OFFSET MEM-1    ;address display data
        MOV DX, 701H            ;address Port B
```

```
;display all 8 digits
    .REPEAT
        MOV AL, AH   ;send selection pattern to Port B
        OUT DX, AL
        DEC DX
        MOV AL, [BX+SI] ;send data to Port A
        OUT DX, AL
        CALL DELAY  ;wait 1.0 ms
        ROR AH, 1     ;adjust selection pattern
        INC DX
        DEC BX    ;decrement counter
    .UNTIL BX == 0
    POPF
    RET
DISP ENDP
```

EXAMPLE 11–11

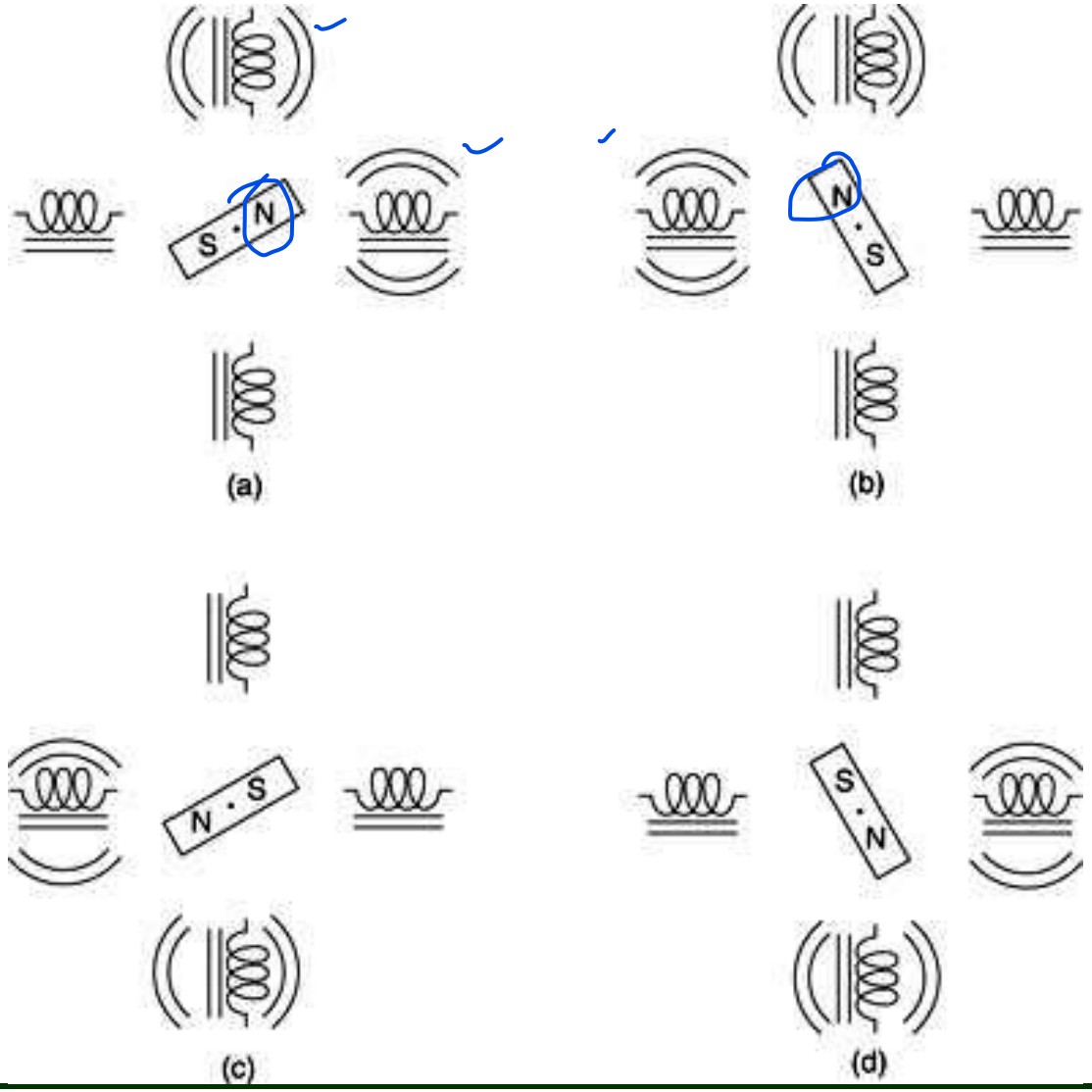
```
;equation for the delay
;
;               Delay Time
;XXXX = ------------
;               time for LOOP
;
DELAY PROC NEAR USES CX

      MOV CX, XXXX

D1:

      LOOP D1

      RET

DELAY ENDP
```

A Stepper Motor Interfaced to the 82C55.

- Another device often interfaced to a computer system is the **stepper motor**.
  - a digital motor because it is moved in discrete steps as it traverses through 360°
- An inexpensive stepper motor is geared to move perhaps 15° per step
- A more costly, high-precision stepper motor can be geared to 1° per step.

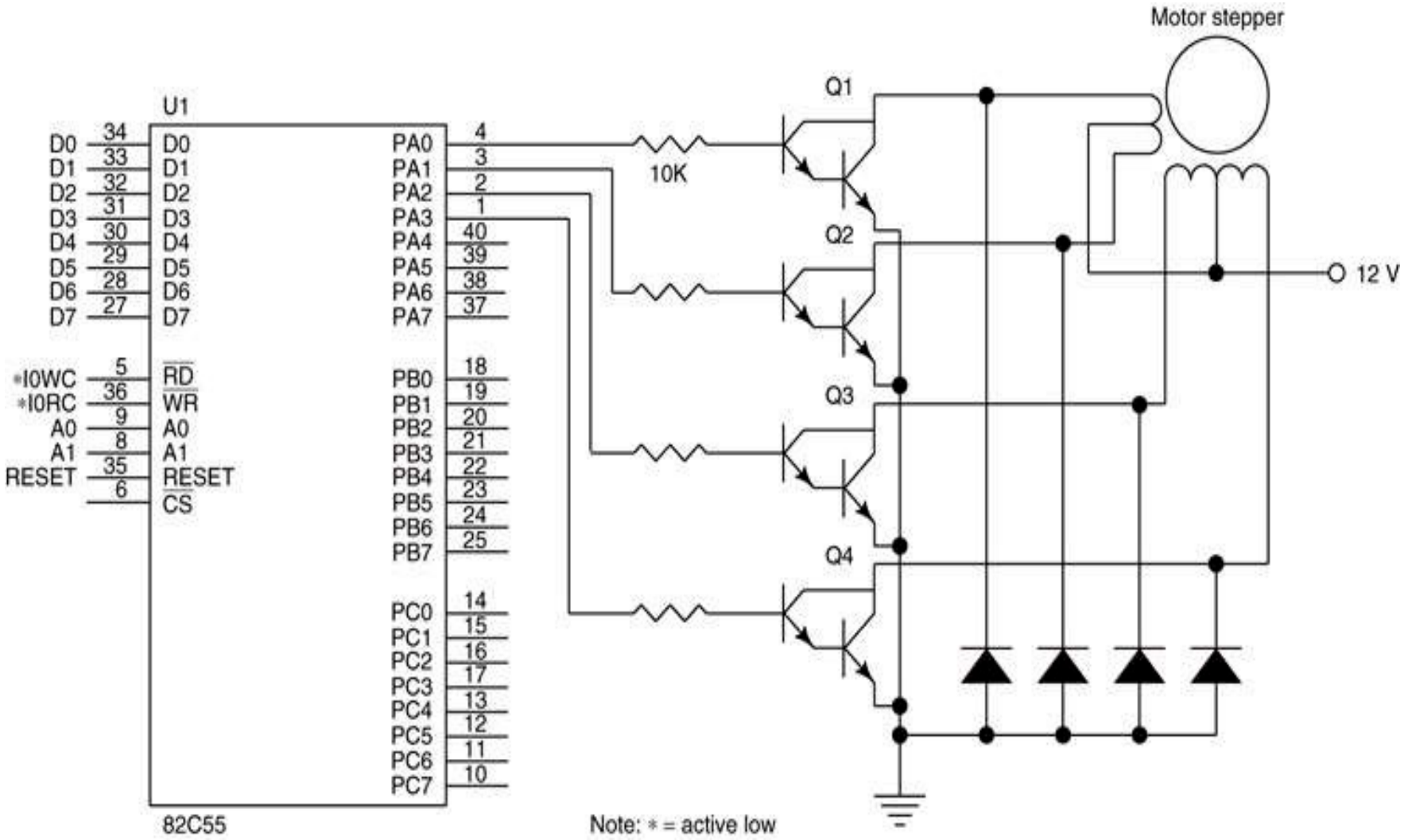
- In all cases, these steps are gained through many magnetic poles and/or gearing.
- Figure 11–23 shows a four-coil stepper motor that uses an armature with a single pole.
  - two coils are energized
- If less power is required, one coil may be energized at a time, causing the motor to step at 45°, 135°, 225°, and 315°.
- The motor is shown with the armature rotated to four discrete places, called full stepping.
  - accomplished by energizing the coils, as shown

Figure 11–23 The stepper motor showing full-step operation: (a) 45° (b) 135° (c) 225° (d) 315°.



- The motor is driven by NPN Darlington amp pairs to provide a large current to each coil.
- A circuit that can drive this stepper motor is illustrated in Fig 11–24.
  - with the four coils shown in place
- This circuit uses the 82C55 to provide drive signals used to rotate the motor armature in either the right- or left-hand direction.
- A simple procedure that drives the motor is listed in Example 11–16.

Figure 11–24 A stepper motor interfaced to the 82C55. This illustration does not show the decoder.



EXAMPLE 11–16

```
PORT EQU 40H
STEP PROC NEAR USES CX AX
    MOV AL, POS    ;get position
    OR CX, CX      ;set flag bits
    IF !ZERO?
        .IF !SIGN?    ;if no sign
            .REPEAT
                ROL AL, 1  ;rotate step left
                OUT PORT, AL
                CALL DELAY ;wait 1 ms
            .UNTILCXZ
```

```
        .ELSE
            AND CX, 7FFFH  ;make CX positive
            .REPEAT
                ROR AL, 1  ;rotate step right
                OUT PORT, AL
                CALL DELAY  ;wait 1 ms
                .UNTILCXZ
        .ENDIF

    .ENDIF

    MOV POS, AL
    RET

STEP ENDP
```

Key Matrix Interface

- Keyboards come in a variety of sizes, from standard 101-key QWERTY keyboards to special keyboards that contain 4 to 16 keys.
- Fig 11–25 is a key matrix with 16 switches interfaced to ports A and B of an 82C55.
  - the switches are formed into a 4 × 4 matrix, but any matrix could be used, such as a 2 × 8
- The keys are organized into four rows and columns: (ROW_0–ROW_3) (COL_0–COL_3)

Figure 11–25 A 4 × 4 keyboard matrix connected to an 8088 microprocessor through the 82C55 PIA.

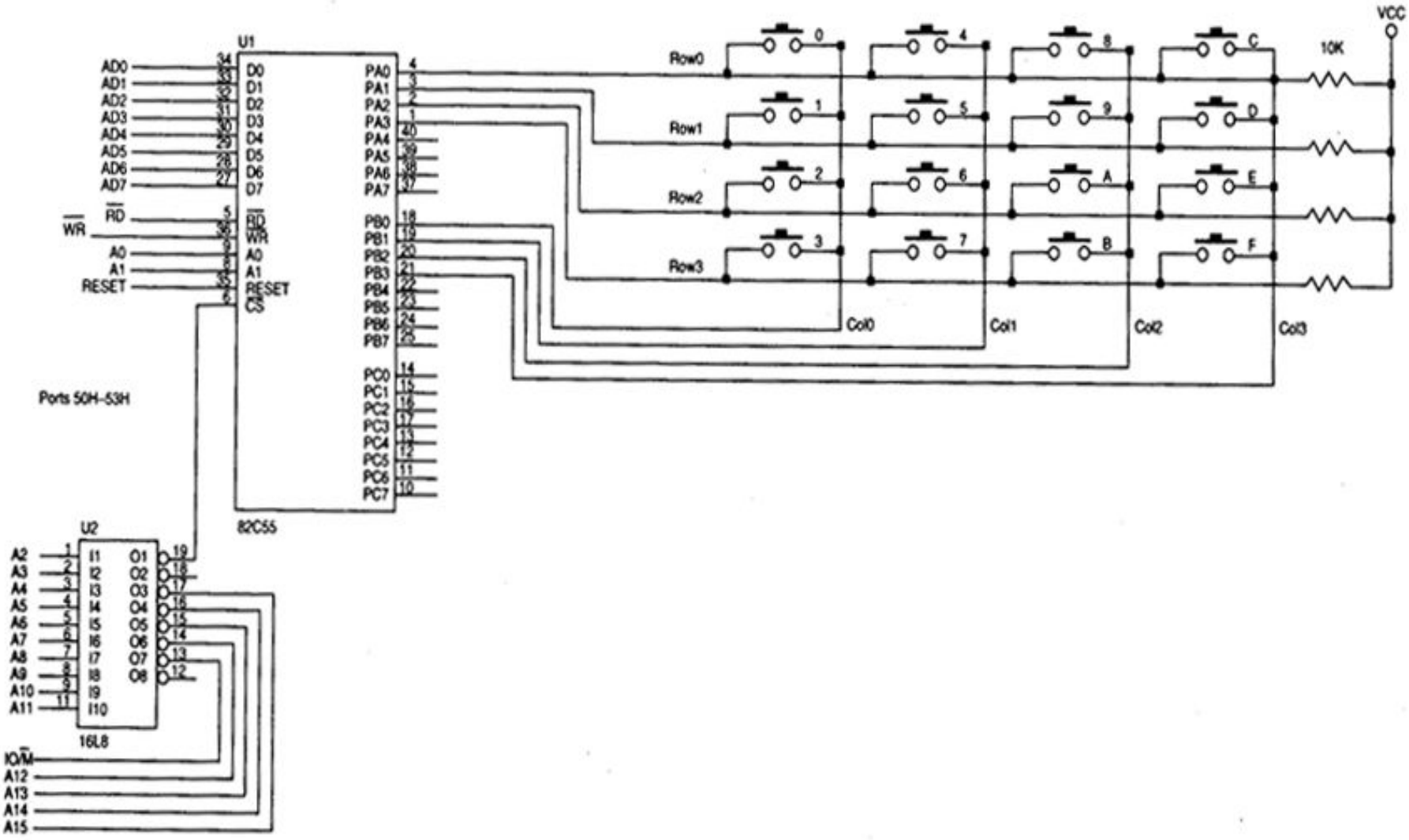
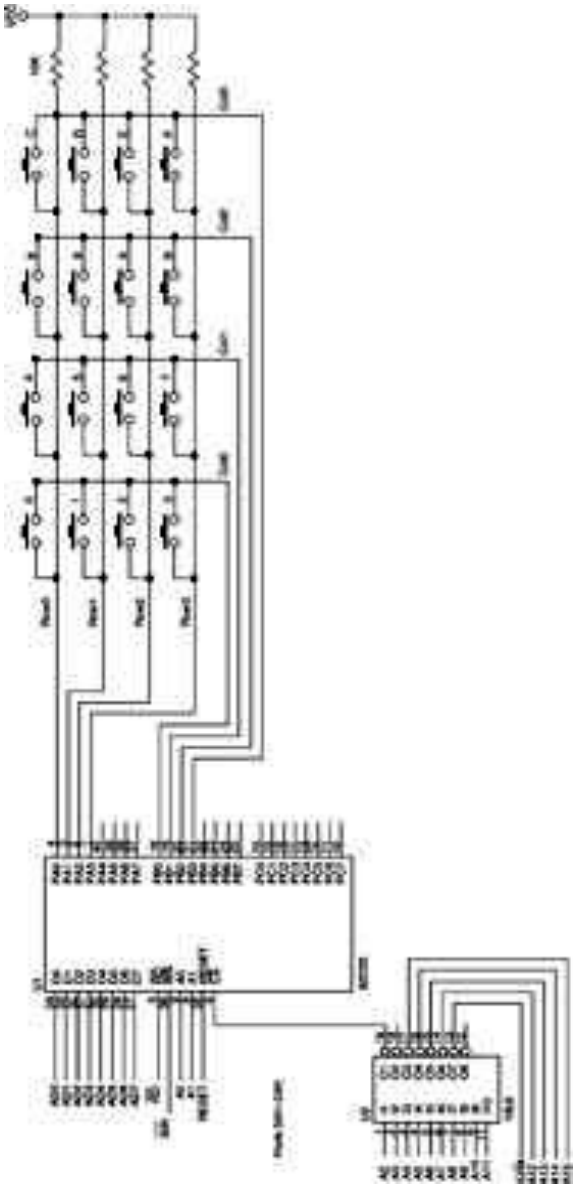
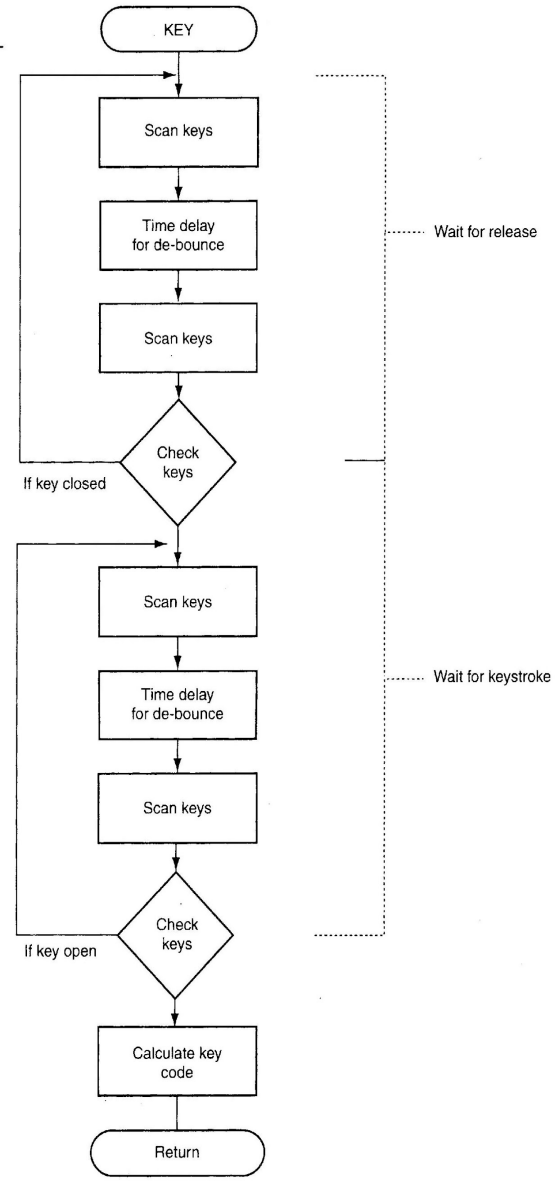


Figure 11–25 A 4 × 4 keyboard matrix connected to an 8088 microprocessor through the 82C55 PIA.



- the 82C55 is decoded at I/O ports 50H–53H for an 8088
- port A is programmed as an input port to read the rows
- port B is programmed as an output port to select a column
- a flowchart of the software required to read a key from the keyboard matrix and debounce the key is illustrated in Fig. 11–26

Figure 11–26 The flowchart of a keyboard-scanning procedure.

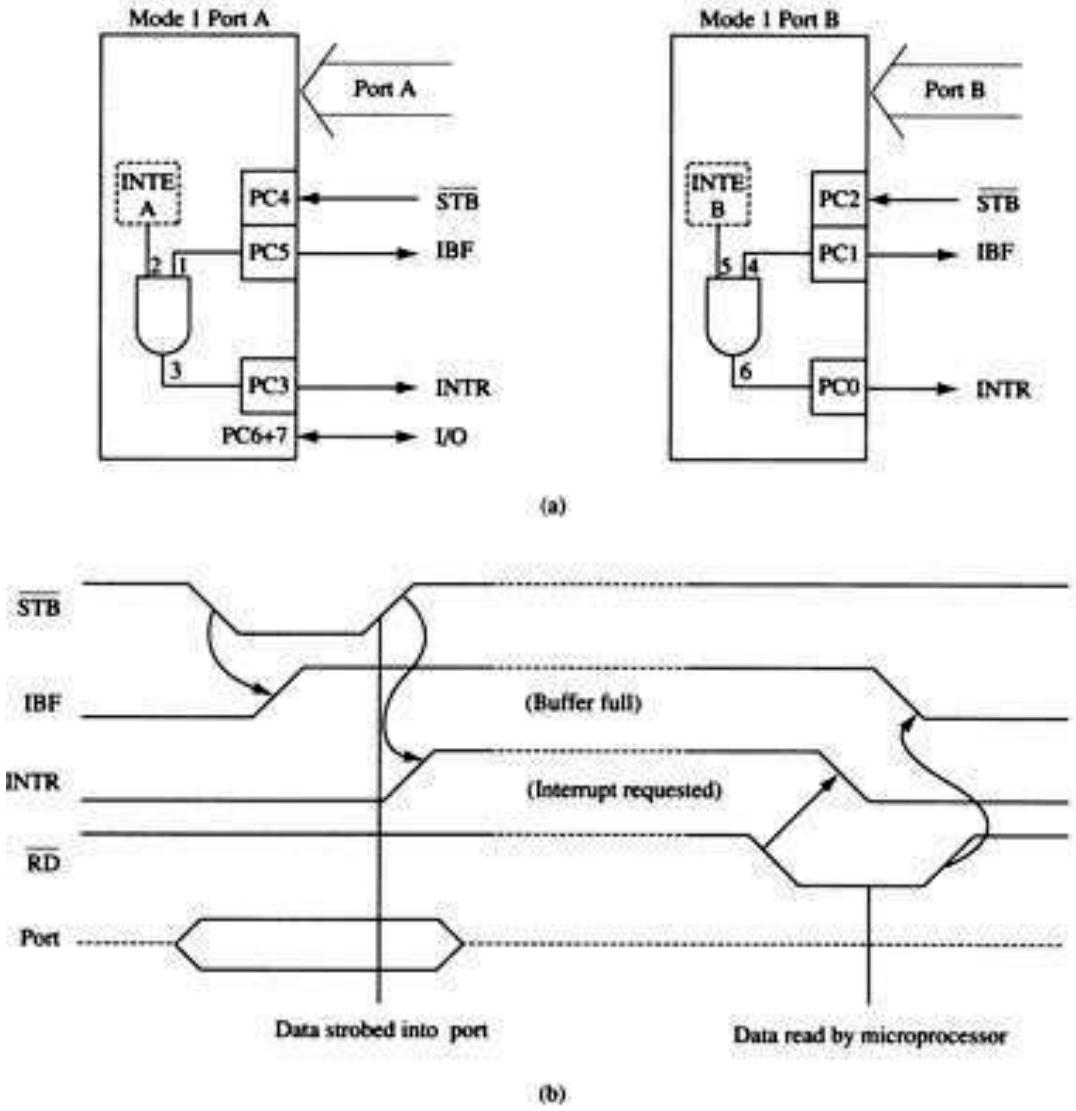


- keys must be debounced, normally with a time delay of 10–20 ms
- the software uses a procedure called SCAN to scan the keys and another called DELAY10 to waste 10 ms of time for debouncing
- the main keyboard procedure is called KEY and appears in Example 11–17
- the KEY procedure is generic, and can handle any configuration from a 1 × 1 matrix to an 8 × 8 matrix.

Mode 1 Strobed Input

- Causes port A and/or port B to function as latching input devices.
  - allows external data to be stored to the port until the microprocessor is ready to retrieve it
- Port C is used in mode 1 operation—not for data, but for control or handshaking signals.
  - to help operate either or both port A and B as strobed input ports
- Fig 11–27 shows how both ports are structured for mode 1 strobed input operation.

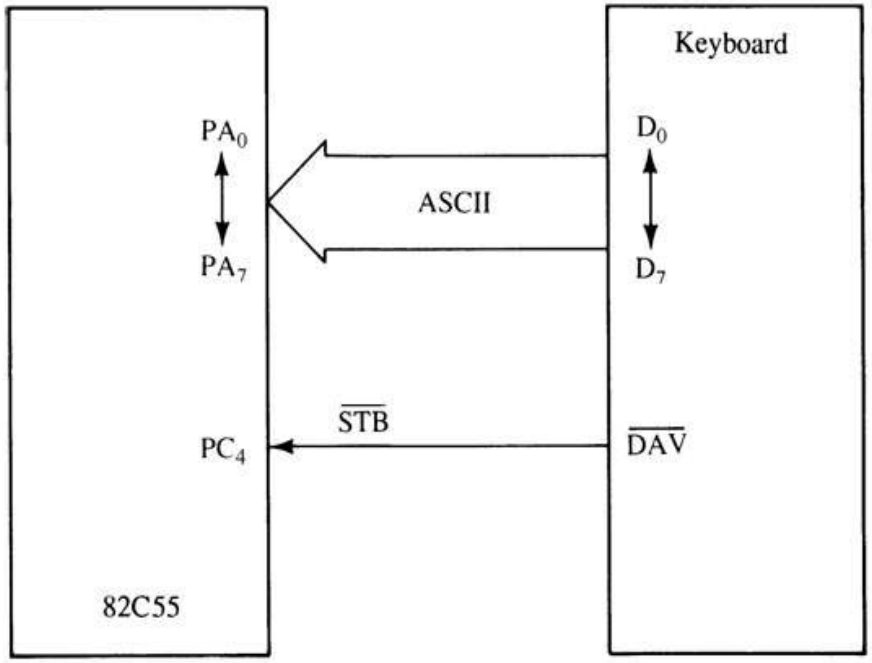
Figure 11–27 Strobed input operation (mode 1) of the 82C55. (a) Internal structure and (b) timing diagram.



Strobed Input Example

- An example of a strobed input device is a keyboard.
- The keyboard encoder debounces the key switches and provides a strobe signal whenever a key is depressed.
  - the data output contains ASCII-coded key code
- Figure 11–28 illustrates a keyboard connected to strobed input port A.

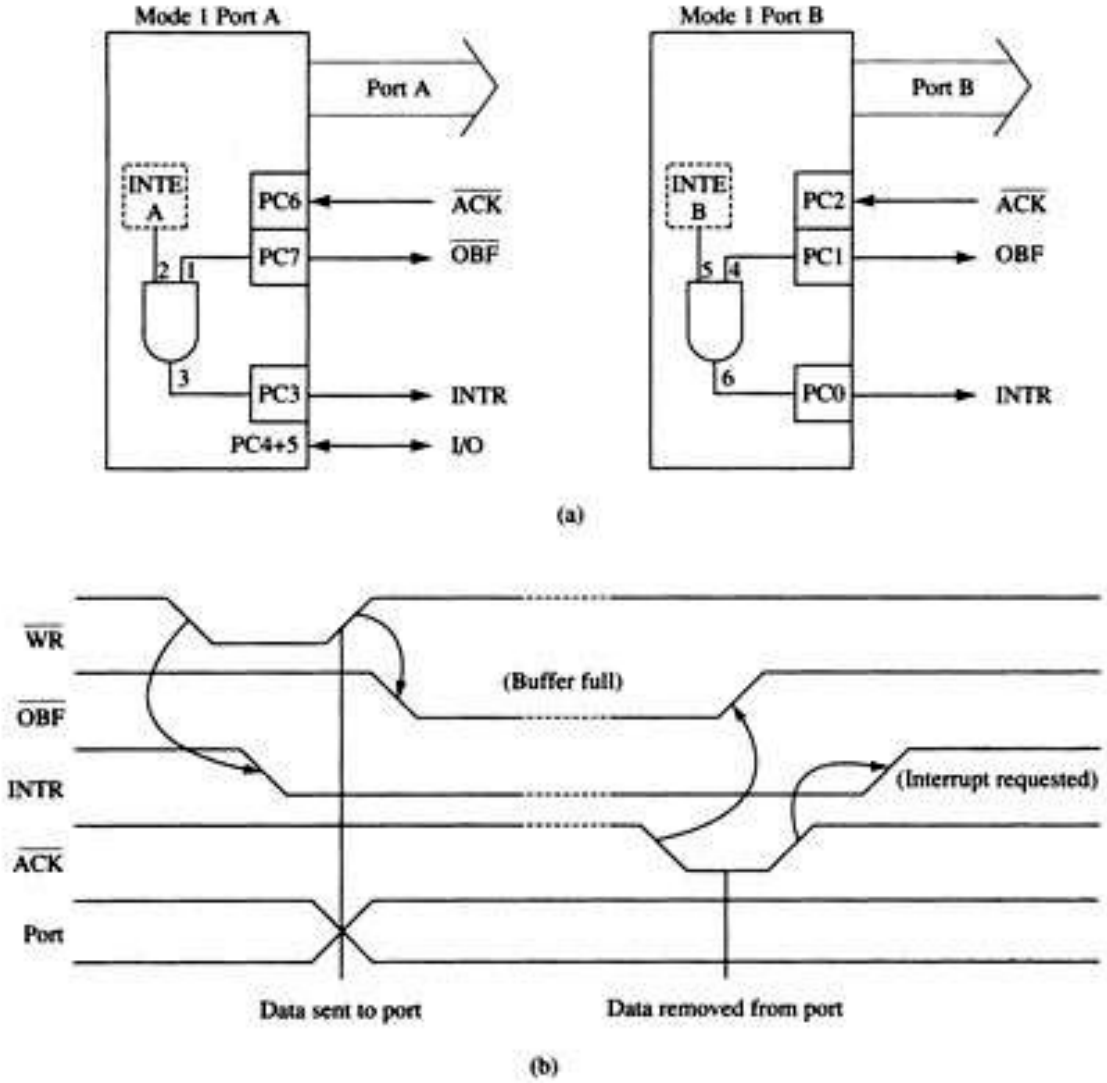
Figure 11–28 Using the 82C55 for strobed input operation of a keyboard.



Mode 1 Strobed Output

- Fig 11–29 shows the internal configuration and timing diagram of 82C55 when operated as a strobed output device under mode 1.
- Strobed output operation is similar to mode 0 output operation.
  - except control signals are included to provide handshaking
- When data are written to a strobed output port, the **output buffer full** signal becomes logic 0 to indicate data are present in the port latch.

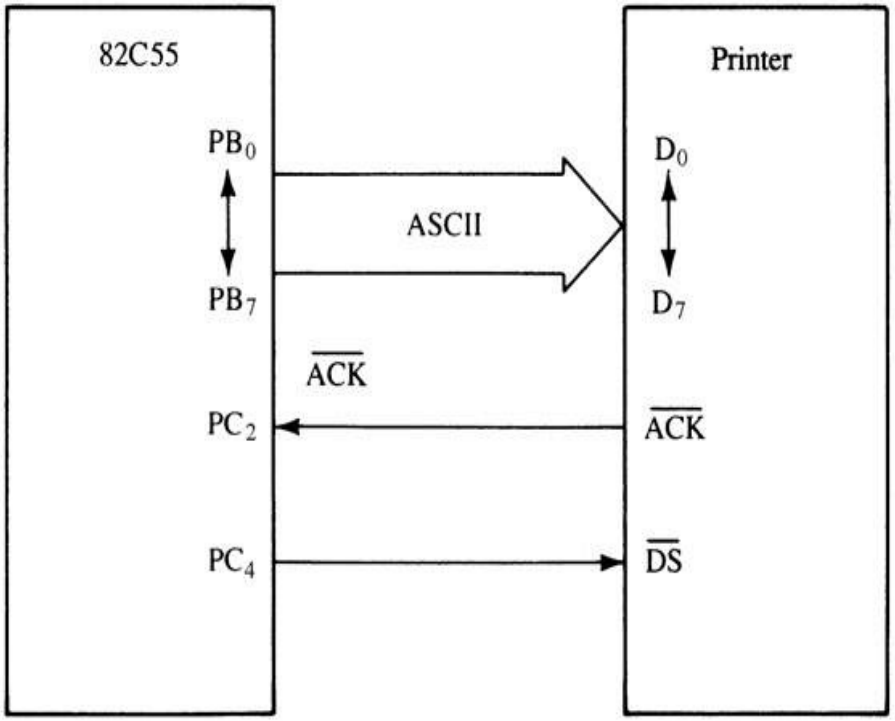
Figure 11–29 Strobed output operation (mode 1) of the 82C55. (a) Internal structure and (b) timing diagram.



Strobed Output Example

- The printer interface demonstrates how to achieve strobed output synchronization between the printer and the 82C55.
- Figure 11–30 illustrates port B connected to a parallel printer, with eight data inputs for receiving ASCII-coded data, a $\overline{DS}$ (**data strobe**) input to strobe data into the printer, and an $\overline{ACK}$ output to acknowledge the receipt of the ASCII character.

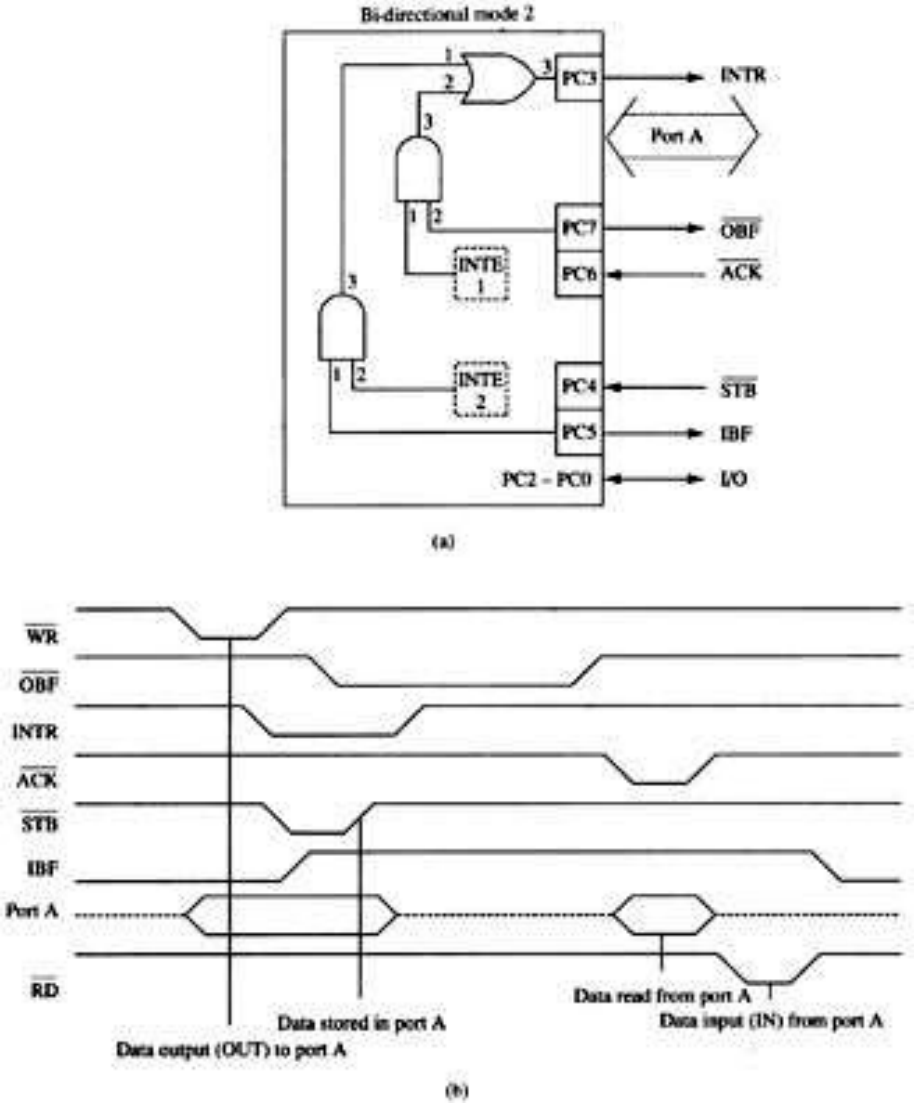
Figure 11–30 The 82C55 connected to a parallel printer interface that illustrates the strobed output mode of operation for the 82C55.



Mode 2 Bidirectional Operation

- Mode 2 is allowed with group A only.
- Port A becomes bidirectional, allowing data transmit/receive over the same eight wires.
  - useful when interfacing two computers
- Also used for IEEE-488 parallel high-speed GPIB (**general- purpose instrumentation bus**) interface standard.
- Figure 11–31 shows internal structure and timing for mode 2 bidirectional operation.

Figure 11–31 Mode 2 operation of the 82C55. (a) Internal structure and (b) timing diagram.



The Bidirectional Bus

- The bidirectional bus is used by referencing port A with the IN and OUT instructions.
- To transmit data through the bidirectional bus, the program first tests to determine whether the output buffer is empty.
  - if so, data are sent to the output buffer via OUT
- The external circuitry also monitors the signal to decide whether the microprocessor has sent data to the bus.

- To receive data through the bidirectional port A bus, IBF is tested with software to decide whether data have been strobed into the port.
  - if IBF = 1, data is input using IN
- The external interface sends data to the port by using the $\overline{\text{STB}}$ signal.
  - the IBF signal becomes logic 1 and data at port A are held inside the port in a latch
- When the IN executes, the IBF bit is cleared and data in the port are moved into AL.
- See Example 11–20 & Example 11–21 for a procedure.

EXAMPLE 11-20: A procedure transmits AH through the bi-directional bus

```
BIT7 EQU 80H
PORTC EQU 62H
PORTA EQU 60H
TRANS     PROC   NEAR
     .REPEAT
          IN  AL, PORTC
          TEST AL, BIT7
     .UNTIL !ZERO?
     MOV AL, AH
     OUT PORTA, AL
     RET
PRINT     ENDP
```

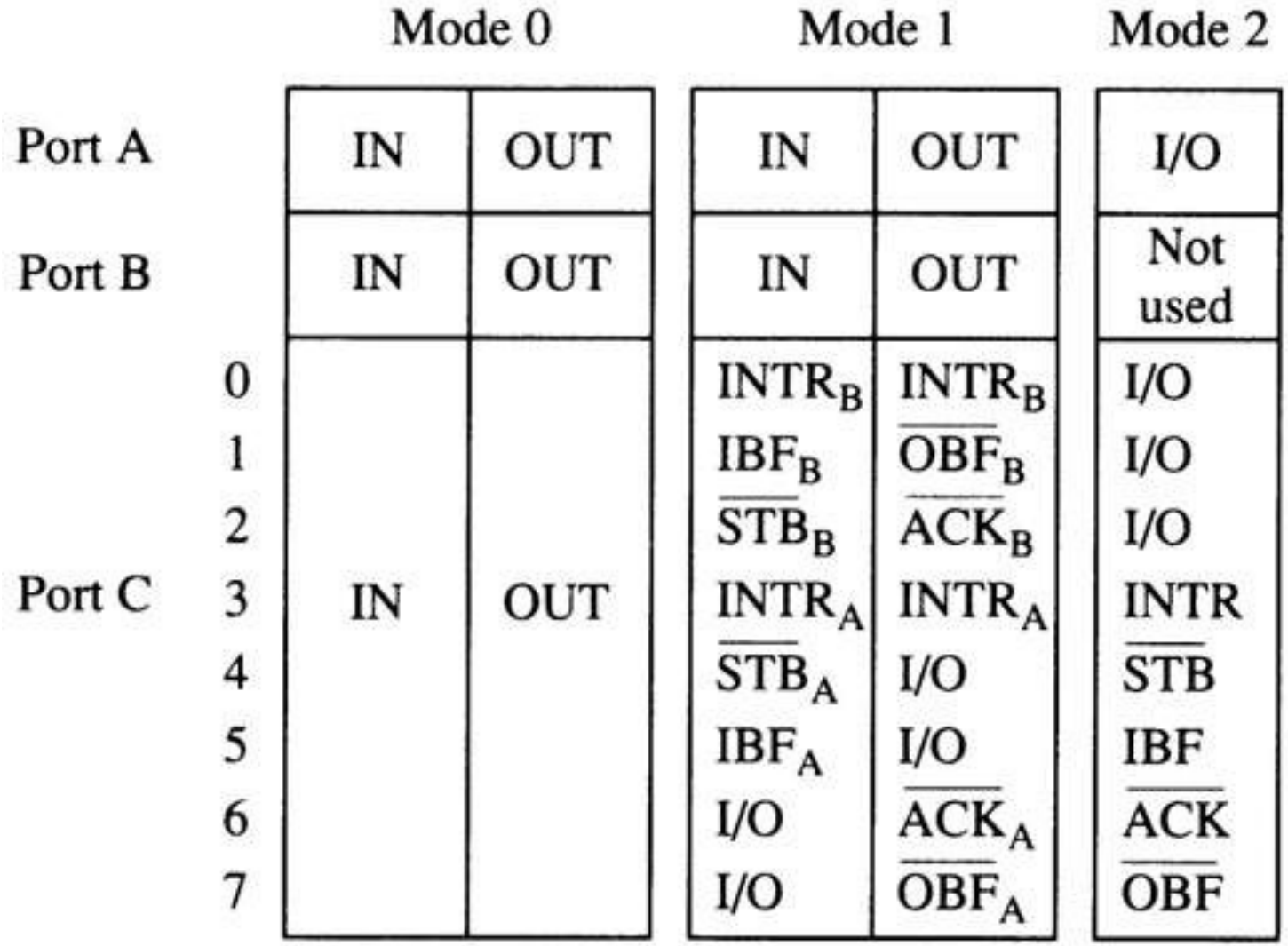
EXAMPLE 11-21: A procedure that reads data from the bi-directional bus into AL

```
BIT5 EQU 20H
PORTC EQU 62H
PORTA EQU 60H
READ  PROC   NEAR
     .REPEAT
          IN   AL, PORTC
          TEST AL, BIT5
     .UNTIL !ZERO?
     IN AL, PORTA
     RET
PRINT      ENDP
```

82C55 Mode Summary

- Figure 11–32 shows a graphical summary of the three modes of operation for the 82C55.
- Mode 0 provides simple I/O.
- Mode 1 provides strobed I/O.
- Mode 2 provides bidirectional I/O.
- These modes are selected through the command register of the 82C55.

Figure 11–32 A summary of the port connections for the 82C55 PIA.



		Mode 0		Mode 1		Mode 2
Port A		IN	OUT	IN	OUT	I/O
Port B		IN	OUT	IN	OUT	Not used
Port C	0	IN	OUT	$INTR_B$	$INTR_B$	I/O
	1			IBF_B	$\overline{OBF}_B$	I/O
	2			$\overline{STB}_B$	$\overline{ACK}_B$	I/O
	3			$INTR_A$	$INTR_A$	INTR
	4			$\overline{STB}_A$	I/O	$\overline{STB}$
	5			IBF_A	I/O	IBF
	6			I/O	$\overline{ACK}_A$	$\overline{ACK}$
	7			I/O	$\overline{OBF}_A$	$\overline{OBF}$

SUMMARY (*cont.*)

- The 82C55 is a programmable peripheral interface (PIA) that has 24 I/O pins that are programmable in two groups of 12 pins each (group A and group B).
- The 82C55 operates in three modes: simple I/O (mode 0), strobed I/O (mode 1), and bidirectional I/O (mode 2).

SUMMARY

(*cont.*)

- When the 82C55 is interfaced to the 8086 operating at 8 MHz, we insert two wait states because the speed of is faster than the 82C55 can handle.