# EEE 421
# Microprocessor and Interfacing

# Chapter 9

# Digital Interfacing

# Programmable Parallel Ports and Handshake Input/output

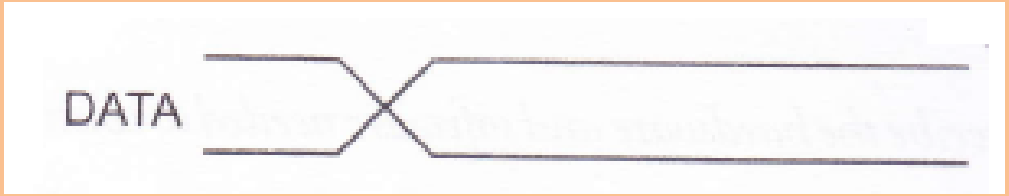
# Methods of parallel Data Transfer

❑SIMPLE INPUT AND OUTPUT:

- This method is applied to get digital data from a simple switch like thermostat, into a microprocessor.
- Example:

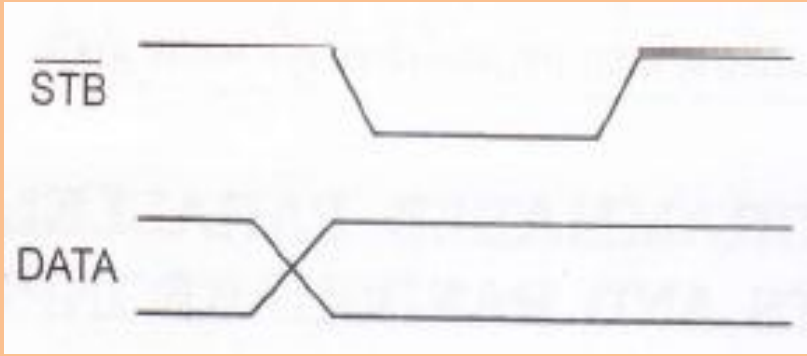
✓connect the switch to an input port line and read the port.

✓connect the input of the LED buffer on an output.



# Methods of parallel Data Transfer

- SIMPLE STROBE I/O:
- In many applications, valid data is present on an external device only at a certain time, so it must be read in at that time. For Example: ASCII keyboard.
- The sending device, such as a keyboard, output parallel data on the data lines, and then outputs an STB signal to let you know that valid data is present.



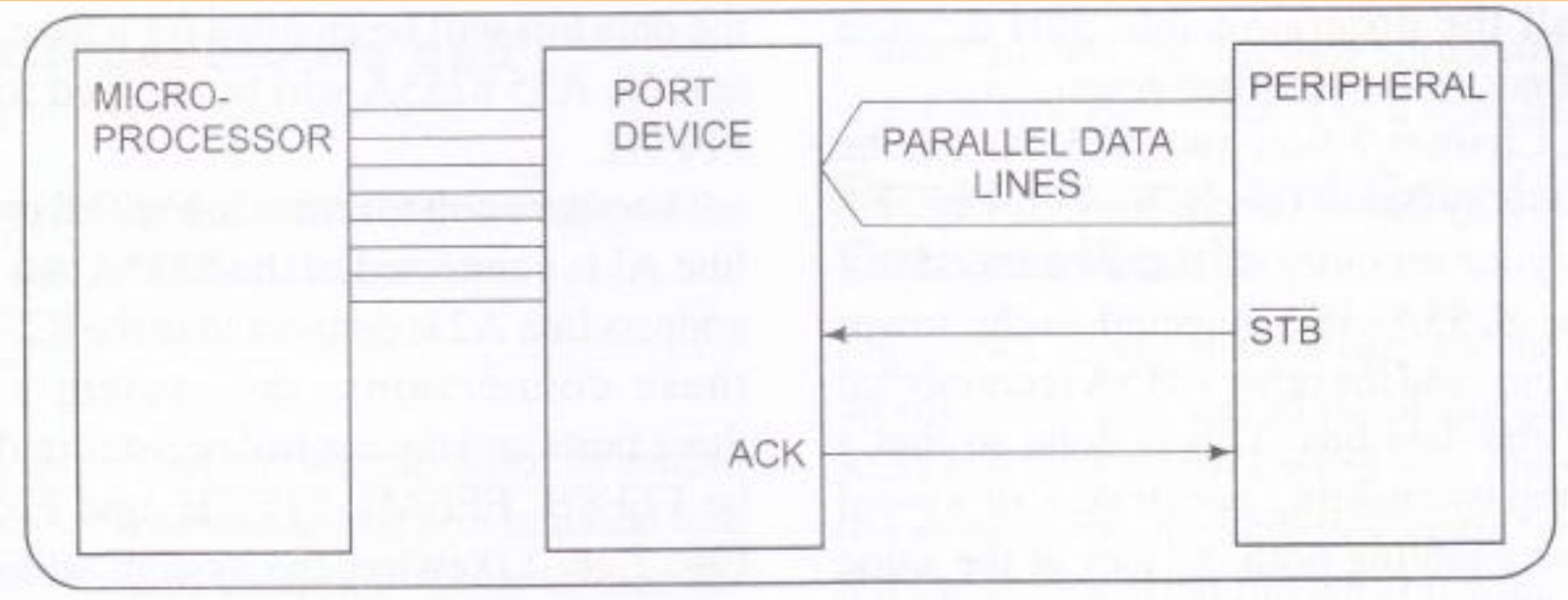
# Methods of parallel Data Transfer

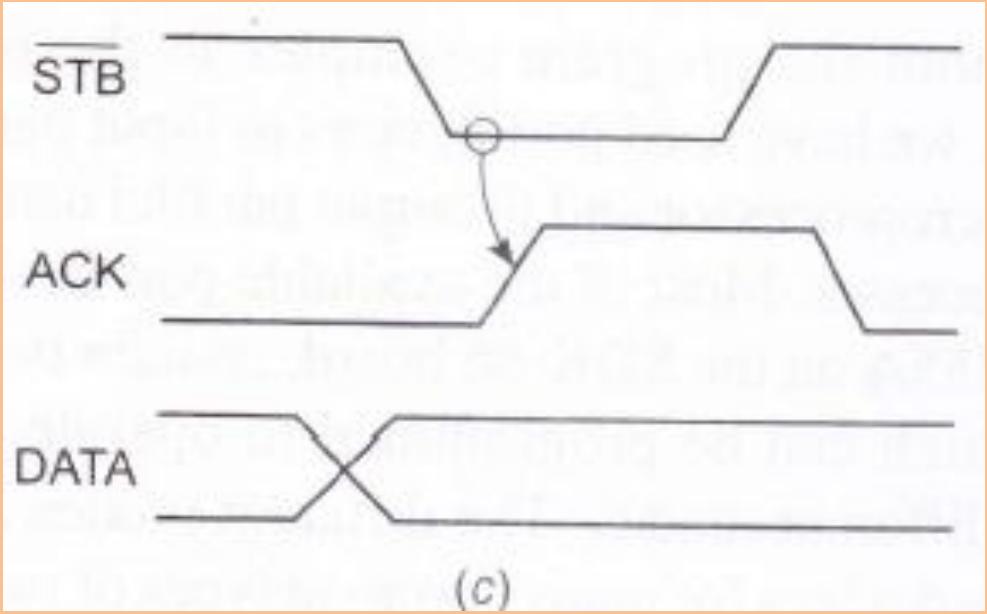
- For low rates of data transfer, such as from a keyboard to microprocessor this method is suitable.
- However, for higher speed data transfer this method does not work, because there is no signal which tells the sending device when it is safe to send the next data byte.

# Methods of parallel Data Transfer

- SINGLE HANDSHAKE I/O:

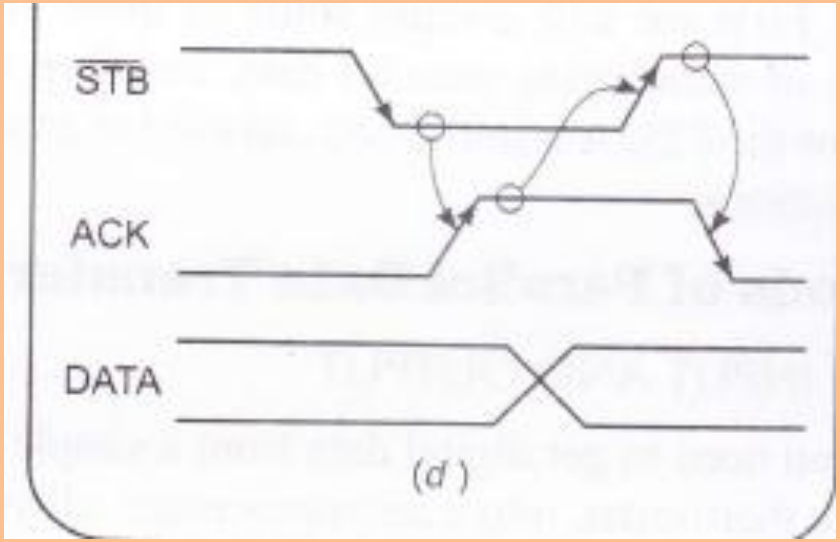
- The peripheral outputs some parallel data and sends an STB signal to the microprocessor. The microprocessor detects the asserted STB signal on a polled or interrupt basis and reads in the bytes of data.
- Then the microprocessor sends an Acknowledge signal (ACK) to the peripheral to indicate that the data has been read and that the peripheral can send the next byte of data.


MICRO-
PROCESSOR
PORT
DEVICE
PARALLEL DATA
LINES
PERIPHERAL
STB
ACK


STB
ACK
DATA
(c)

# Methods of parallel Data Transfer

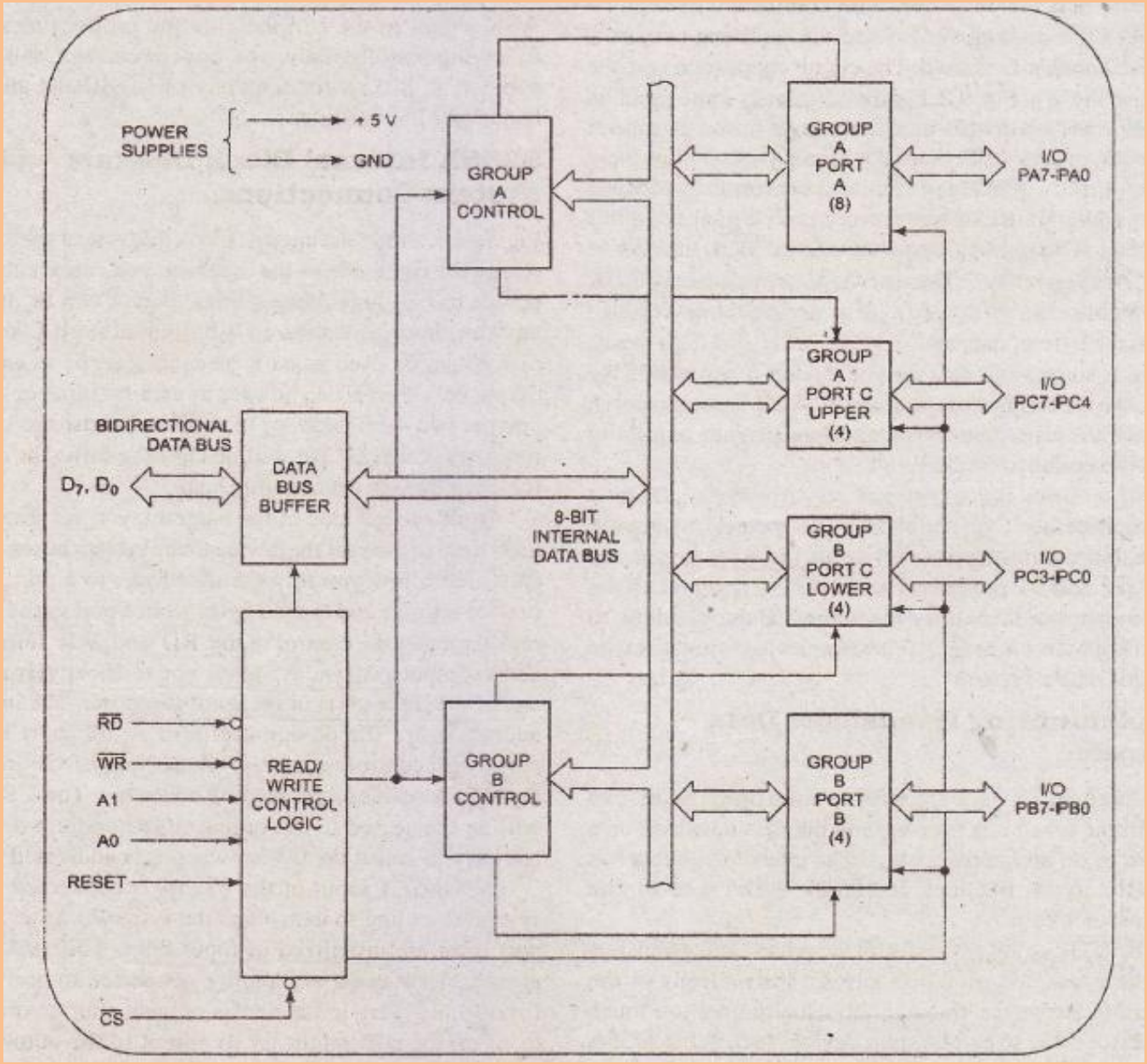
- Double Handshake Data Transfer
- Used in more coordinated system.
- $\overline{STB}$ low: Are you ready?
- ACK high: I am ready.
- The peripheral then send data
- $\overline{STB}$ high: some valid data for peripheral.
- ACK low: I have the data,



# Implementing Handshake data transfer

- 8255 is a port device ,which can be programmed to manage $\overline{STB}$ and ACK signal in proper time.

# 8255 internal Block Diagram



# 8255 Pin Configuration

- Port A ,B can be used in 8 bit input or output.
- Port C can be used in 8 bit input or output

or

- Two 4 bit input or output port

or

- Produce hand shake signal for port A or B.
- $\overline{WR}$ allows to write in port or status register.
- $\overline{RD}$ allows to read from port or status register.
- A0-A1:select address
- A1A0= 00:Port A
- A1A0= 01:Port B
- A1A0= 10:Port C
- A1A0= 11:Control Register

# 8255 Pin Configuration

- $\overline{CS}$: helps to enable chip at specified address.
- RESET: connected to system reset. When system is reset, All ports are initialized as input to prevent destruction of circuitry.
- 7.8 (Sheet 5) the addresses are
- Port A: FFF8H
- Port B:FFFAH
- Port C:FFFCH
- Control Register: FFFEH

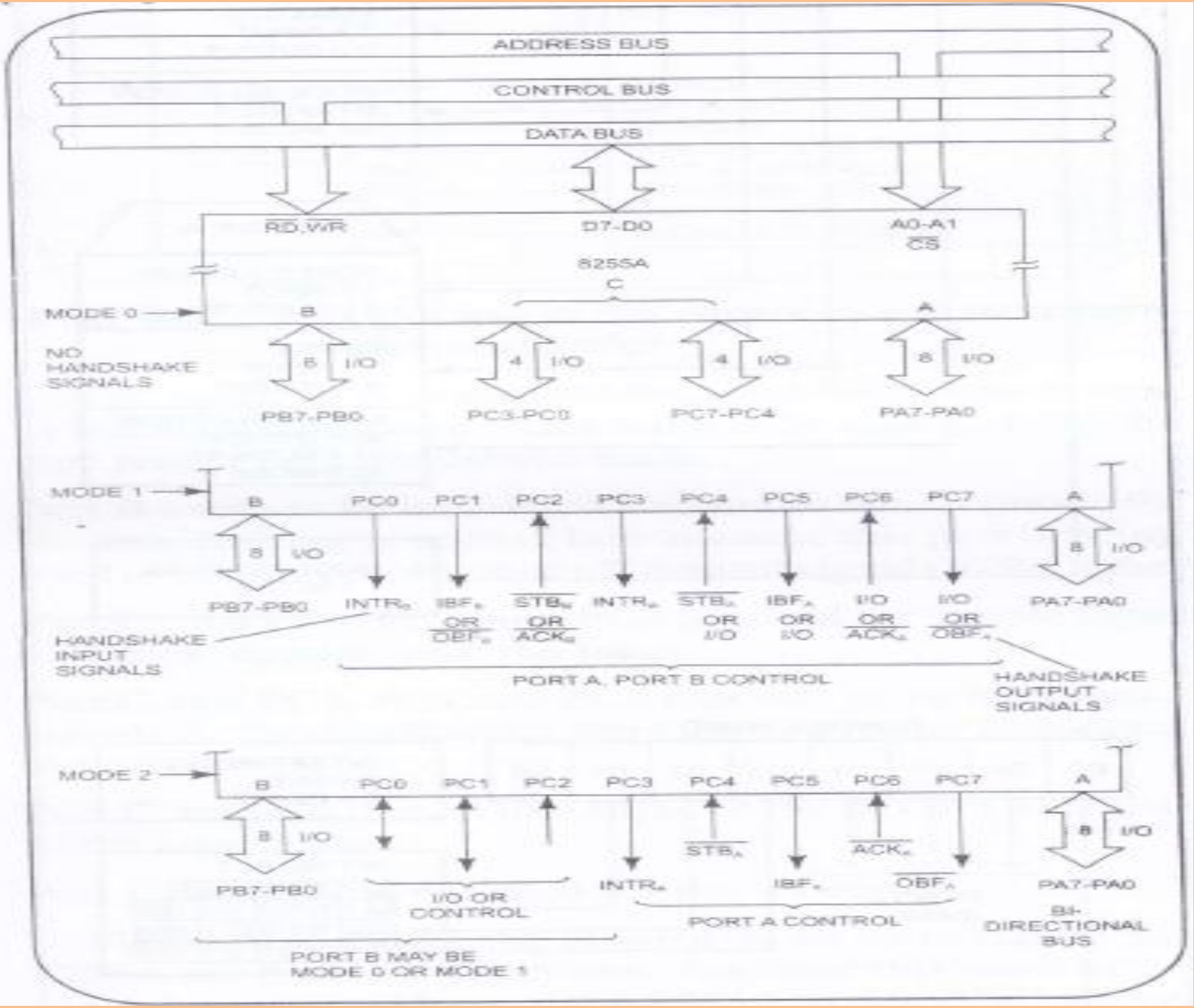
# 8255 Operation Modes and Initialization

- ***Mode 0:***
- Both port A and B are normally initialized and no handshake.
- Port C is used 8 bit port or independent two 4 bit port.
- ***Mode 1:***
- Both port A or port B can be used as handshake (strobed) input or output operation.
- Pins PC0,PC1, and PC2 function as handshake lines for port B if it is initialized in mode 1.
- Pins PC3, PC4 and PC5 function as handshake lines for port A if it is initialized as input port in mode 1.
- Pins PC6 and PC7 are available for use as input lines or output lines.
- If port A is initialized as a handshake output port, then port C pins PC3, PC6 and PC7 function as handshake signals.
- Port C pins PC4 and PC5 are available for use as input or output lines.

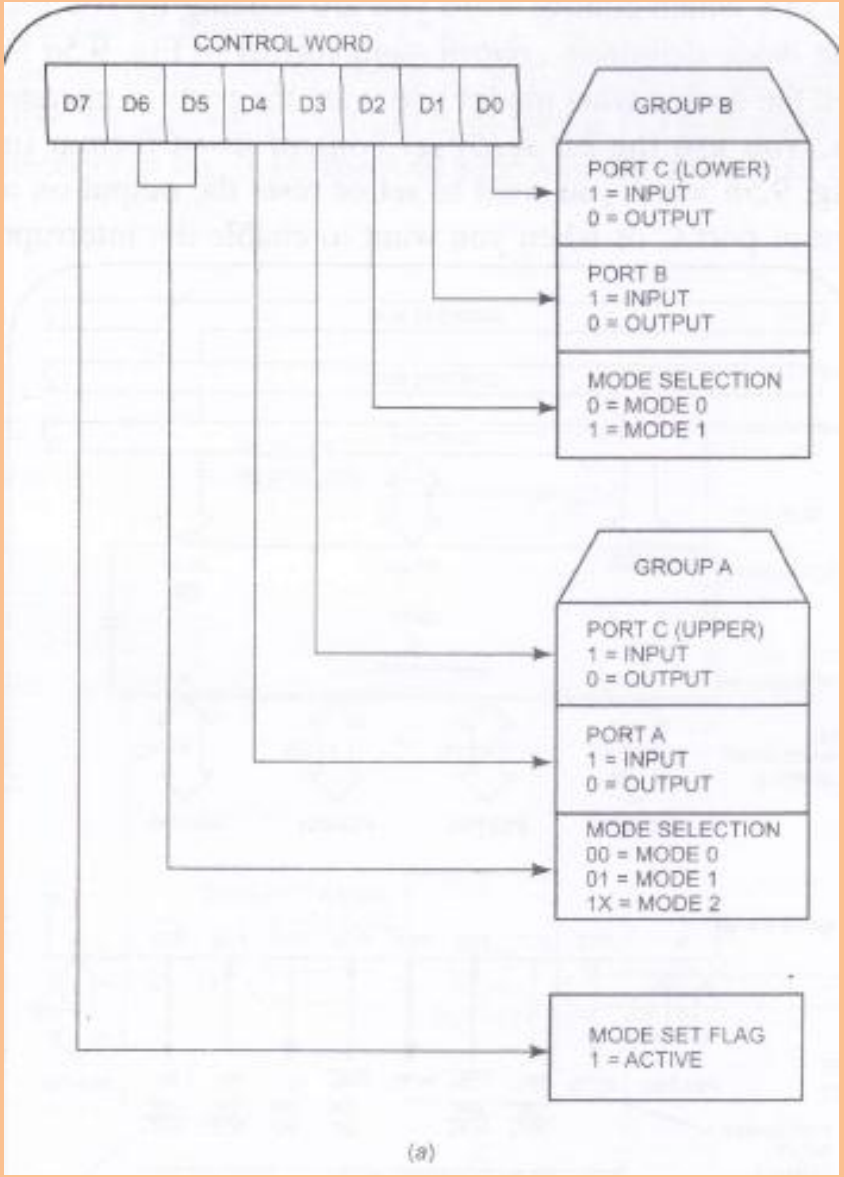
# 8255 Operation Modes and Initialization

❑ ***Mode 2:***

- Only port A can be initialized in mode 2 and bidirectional handshake data transfer.
- if port A is initialized in mode 2, then pins PC3 to PC7 used as handshake lines for port A.
- The other three pin PC0 to PC2 used as I/O port if port B is in mode 0.
- The three pins will be used for port B handshake lines if port B is initialized in mode1.


ADDRESS BUS
CONTROL BUS
DATA BUS
RD, WR
D7-D0
A0-A1
CS
8255A
C
MODE 0
B
A
NO HANDSHAKE SIGNALS
8 I/O
4 I/O
4 I/O
8 I/O
PB7-PB0
PC3-PC0
PC7-PC4
PA7-PA0
MODE 1
PC0
PC1
PC2
PC3
PC4
PC5
PC6
PC7
INTR_B
IBF_B
OR
OBF_B
STB_B
OR
ACK_B
INTR_A
STB_A
OR
I/O
IBF_A
OR
I/O
I/O
OR
ACK_A
I/O
OR
OBF_A
HANDSHAKE INPUT SIGNALS
PORT A, PORT B CONTROL
HANDSHAKE OUTPUT SIGNALS
MODE 2
I/O OR CONTROL
STB_A
ACK_A
INTR_A
IBF_A
OBF_A
PORT A CONTROL
BI-DIRECTIONAL BUS
PORT B MAY BE MODE 0 OR MODE 1

# Constructing and Sending 8255A Control Words



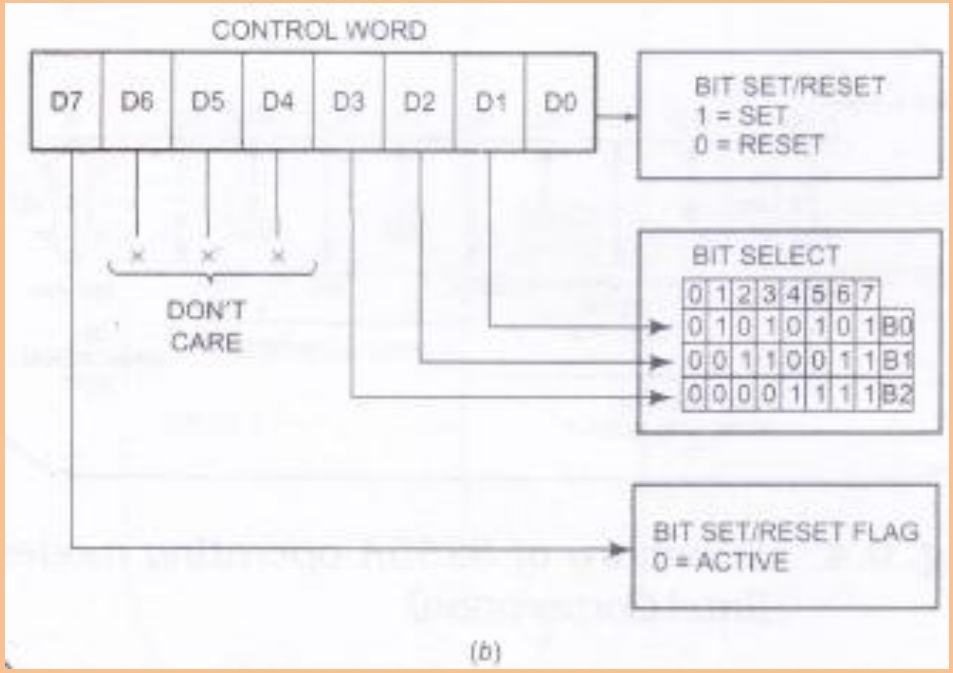




**Fig. 9.5** *8255A control word formats, (a) Mode-set control word, (b) Port C bit set/reset control word.*

# Sending Control words

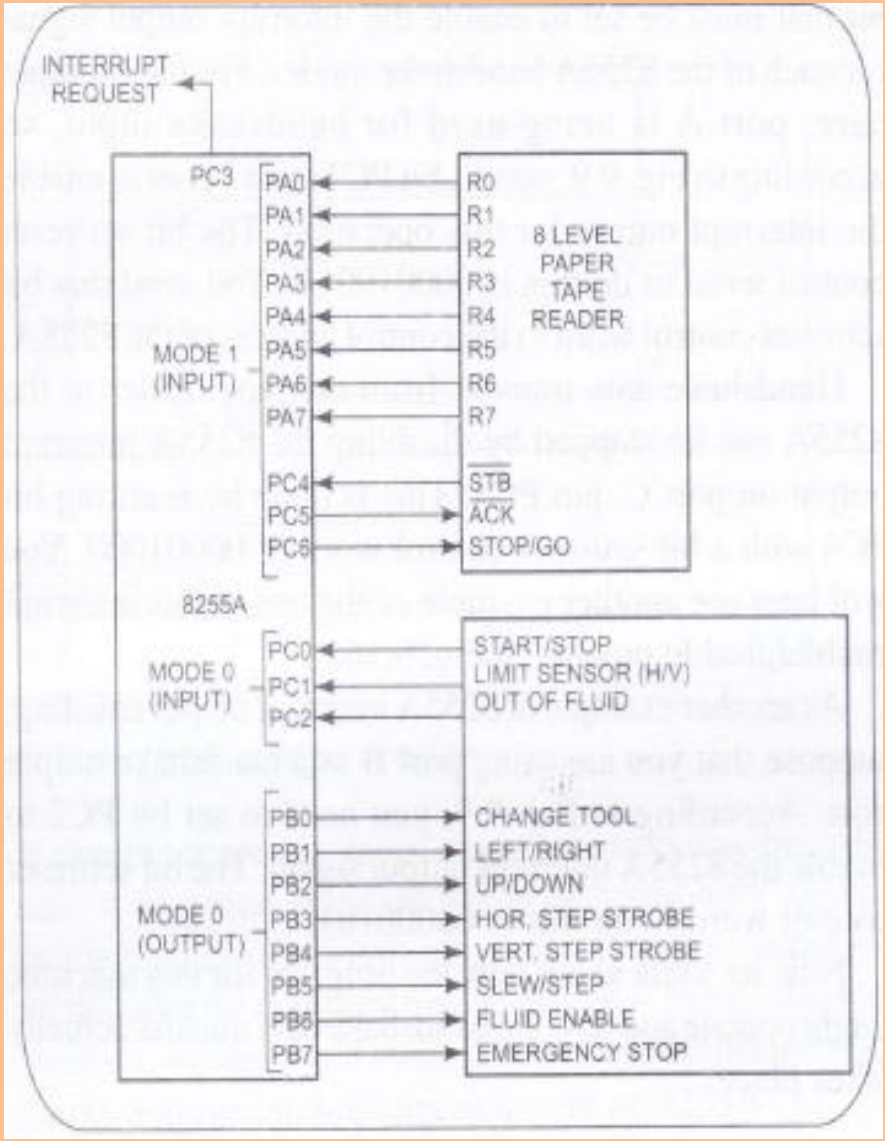
❑***Coding:***

- MOV AL, 10001110B ;Load control word.

  MOV DX, FFFEH ;Load control register.

  OUT DX, AL
- Procedure is similar for sending both control word.

# 8255A Handshake Application Examples




- Port A needs to be initialized for handshake input (mode1) because instruction codes have to be read in from the tape reader on a handshake basis.
- Port B needs to be initialized for simple output (mode 0).
- Port C, bits PC0, PC1 and PC2 are used for simple input of sensor signals from the lathe.
- Port C bits PC3, PC4, and PC5 function as the handshake signals for the data transfer from the tape reader connected to port A.
- Port C, bit PC6 is used for output of the STOP/GO signal to the tape reader.
- Port C, bit PC7 is not used for this example.

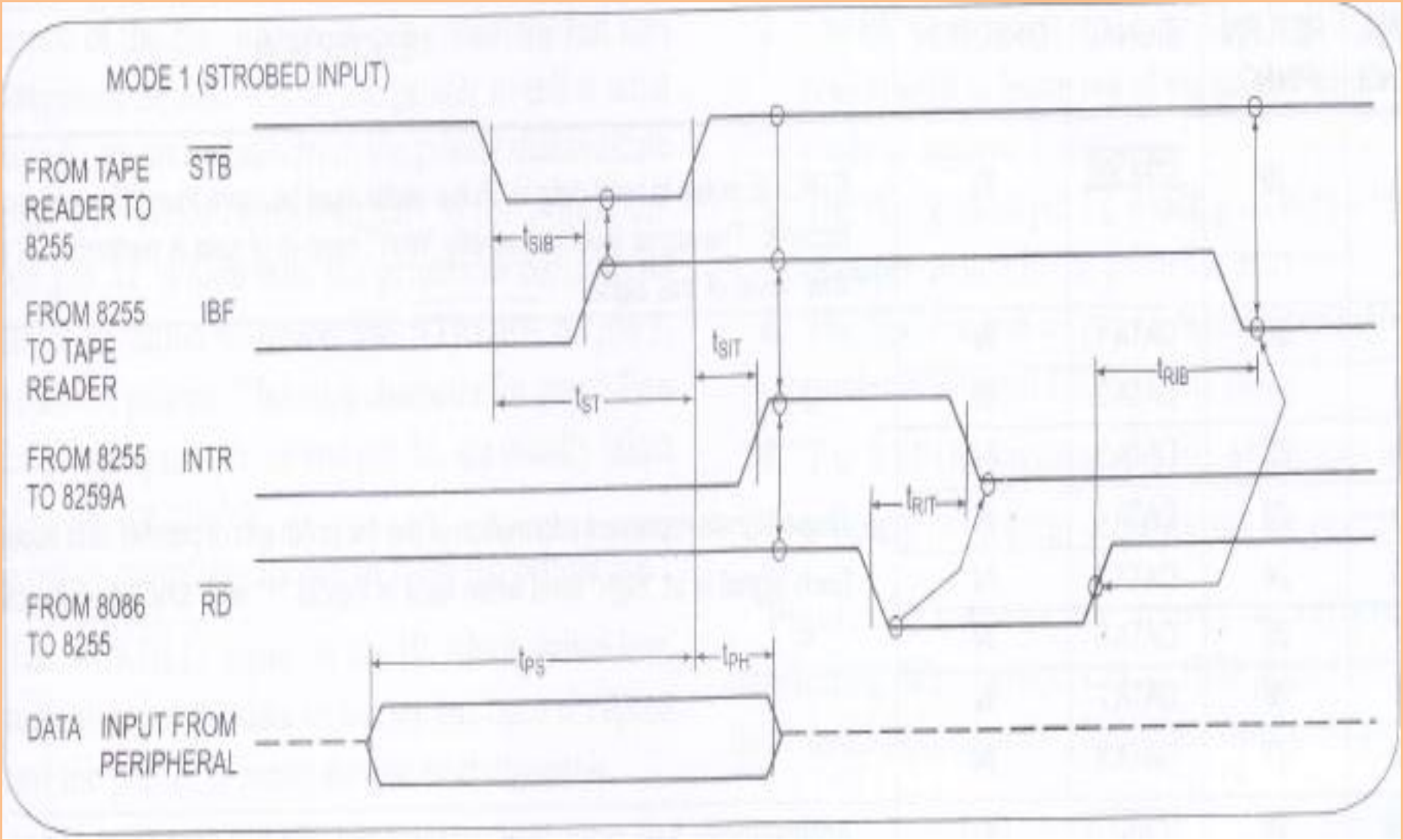
# Data reading from tape recorder

- Tape recorder sends data to port A of 8255.
- Tape recorder also sends $\overline{STB}$ signal to 8255 to tell that *a new data is sent.*
- 8255 responds with IBF (Input Buffer Full) signal to be high to tell *8255 is ready for data.*
- When IBF is high $\overline{STB}$ is high again. It causes two effects on 8255.
- 1st:8255 latches data and tape can remove it and load new one.
- 2nd:Enables interrupt of 8086 by PC3 of 8255.

# Data reading from tape recorder

- Microprocessor are ready to read data.
- $\overline{RD}$ goes low and read from port A.
- 8255 automatically reset interrupt signal so that the second interrupt cannot cause the same data transfer.
- When RD goes high, IBF (PC5) will low again automatically by 8255, which indicate that the data transfer is complete.

# Timing Diagram



# Interfacing a Microprocessor to Keyboards

- Keyboard Types
  - Mechanical Key switches
  - Membrane Key switches
  - Capacitive Key switches
  - Hall Effect Key switches

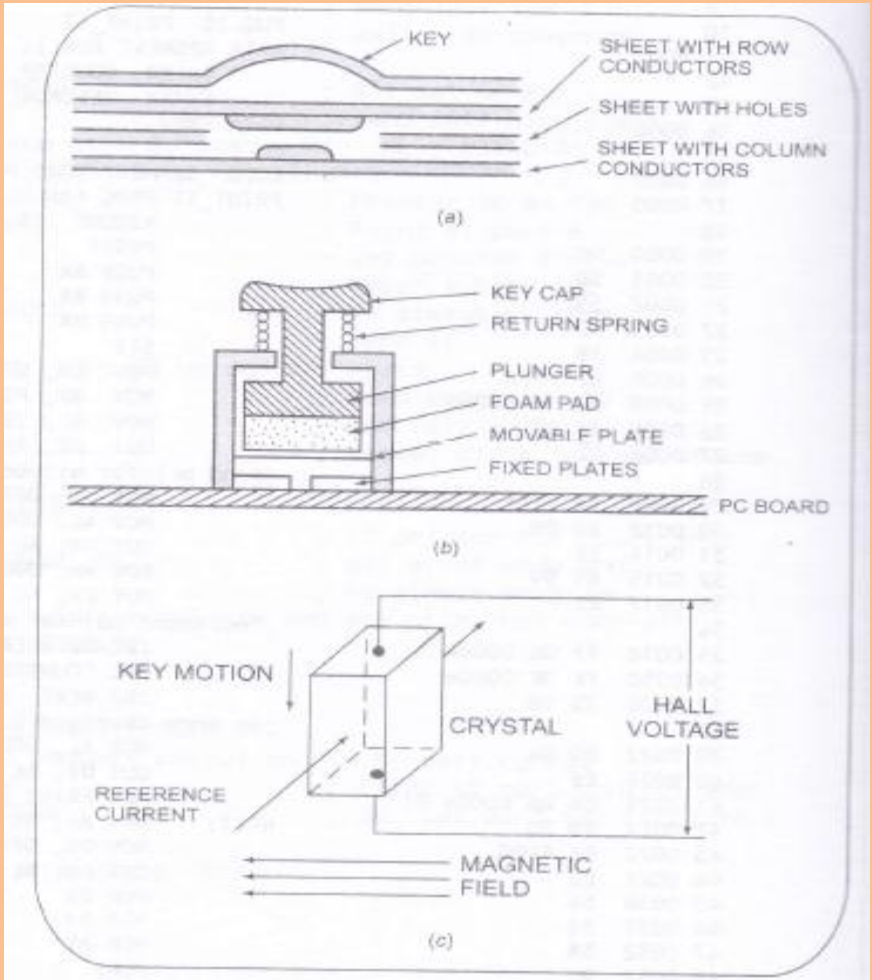



**Fig. 9.18** *Keyswitch types. (a) Membrane. (b) Capacitive. (c) Hall effect.*

# Interfacing a Microprocessor to Keyboards

- Keyboard Circuit Connections and Interfacing
- In most keyboards, the key switches are connected in a matrix of rows and columns
- Getting meaningful data from a keyboard requires the following three major tasks:

✓ 1. Detect a key Press.

✓ 2. Debounce the key Press.

✓ 3. Encode the key press (produce a standard code for the Pressed key).

# Software Key board Interfacing

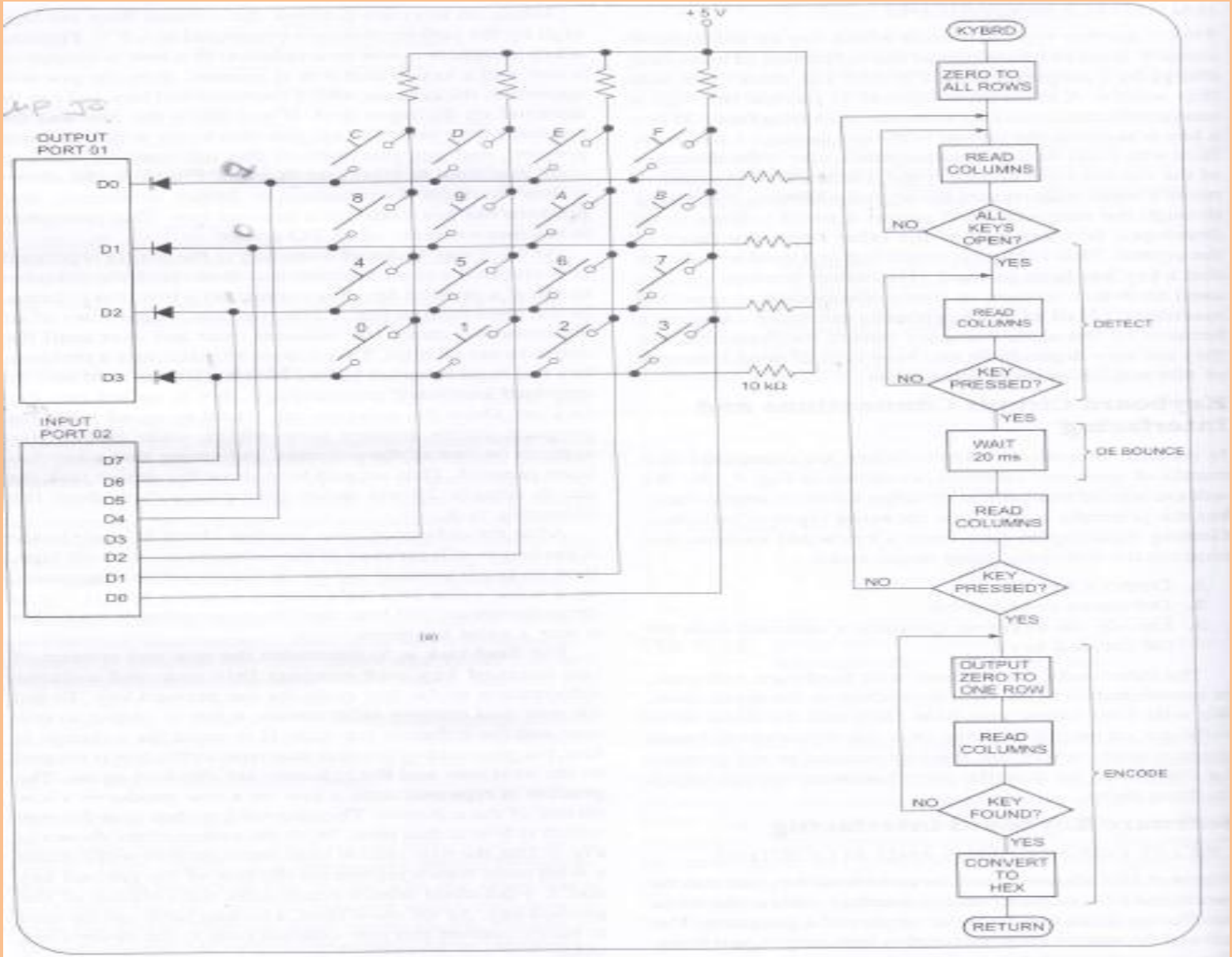



**Fig. 9.19** *Detecting a matrix keyboard keypress, debouncing it, and encoding it with a microcomputer (a) Port connections. (b) Flowchart for procedure.*

# Keyboard Interfacing with Hardware

- For systems where the CPU is too busy in other major tasks in software, an external device is used to do them.
- One example of a MOS device, which can do this is the General Instrument AY5-2376,
- The AY5 -2376 independently detects a key press by cycling a low down through the rows and checking the columns just as we did in software.
- The AY5-2376 has a feature called two-key rollover.
- This means that if two keys are pressed at nearly the same time, each key will be detected, debounced, and converted to ASCII.

# Dedicated Microprocessor Keyboard Encoders

- The 8048 microprocessor used here contains an 8-bit CPU, a ROM, some RAM, three ports and a programmable timer/counter.
- A program stored in the on-chip ROM performs the three keyboard tasks and sends the code for a pressed key out to the computer.
- The key code is sent out in serial form rather than in parallel form.
- One of the major advantages of using a dedicated microprocessor to do the three keyboard tasks is programmability.
- Special-function keys on the keyboard can be programmed to send out any code desired for a particular application.

# Dedicated Microprocessor Keyboard Encoders

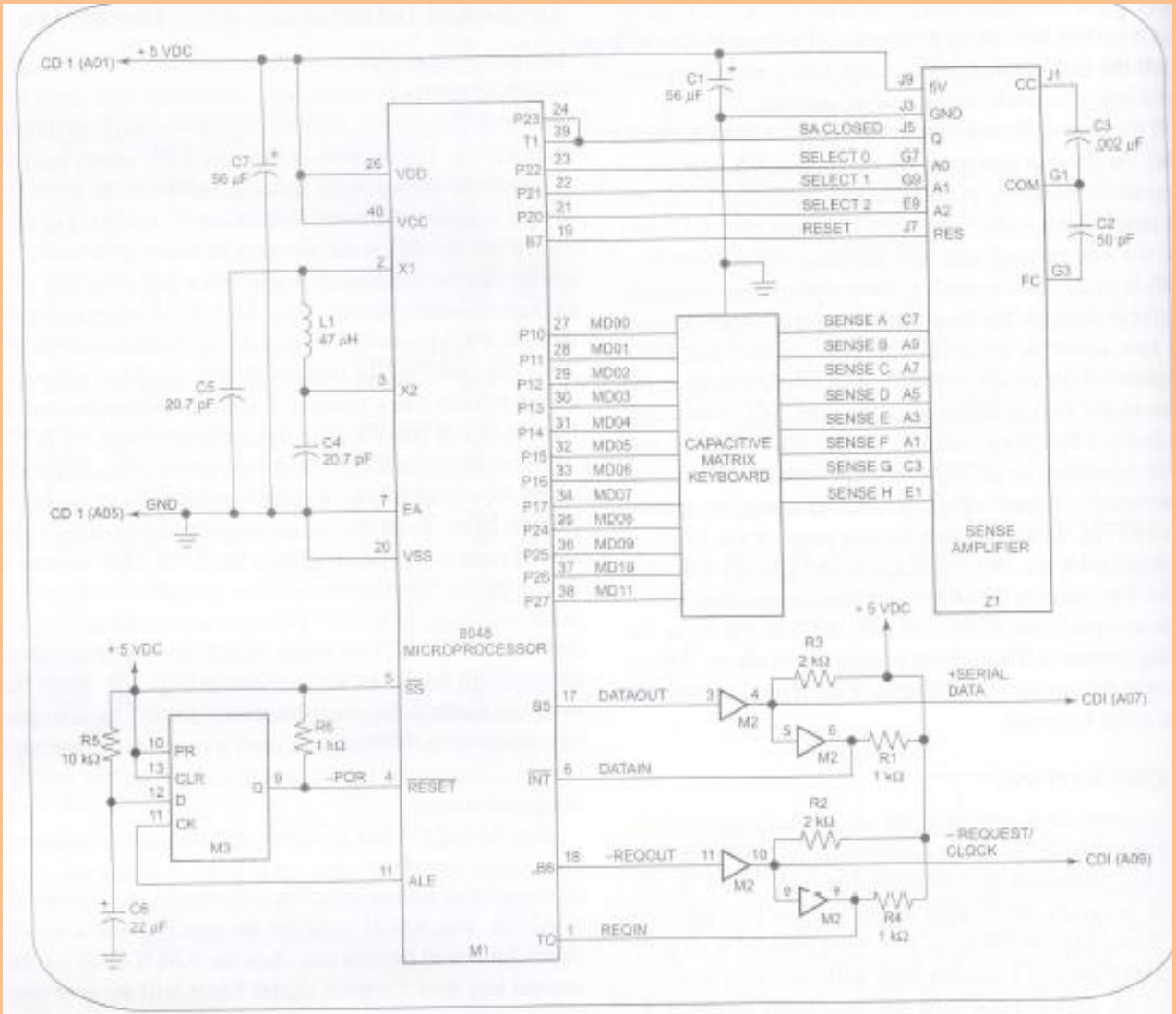
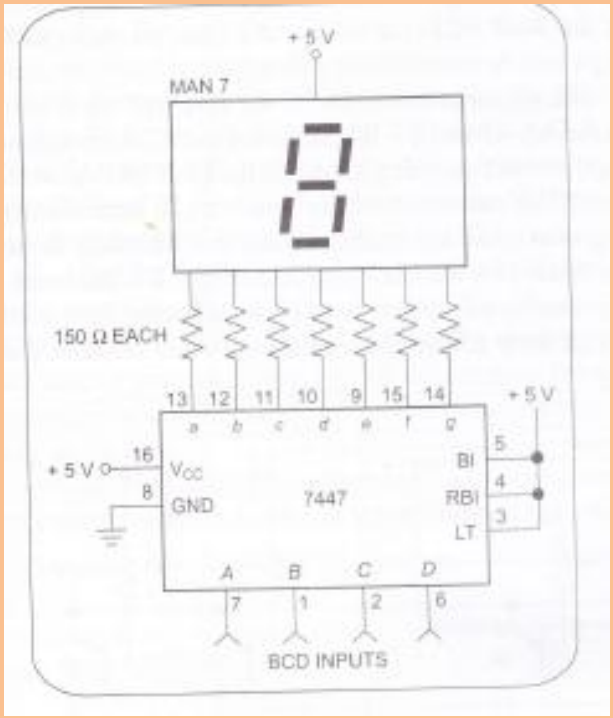
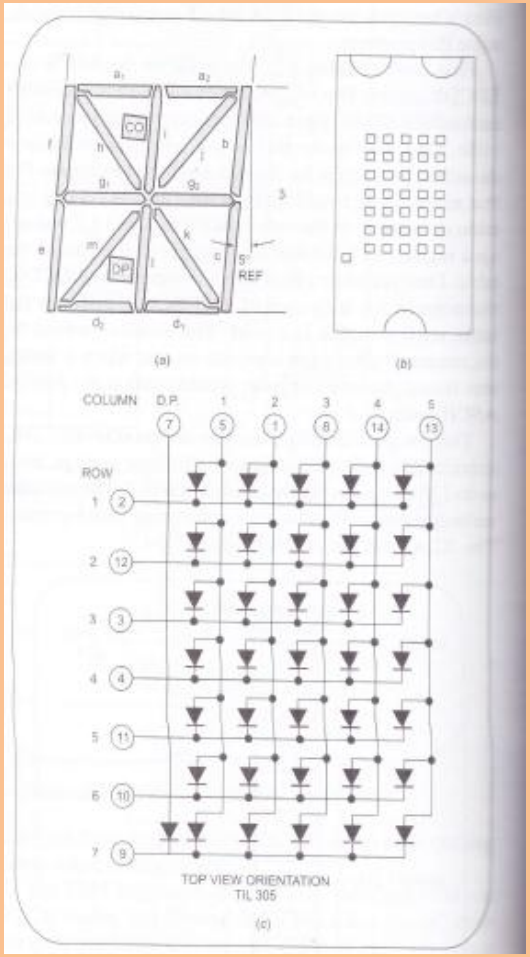



**Fig. 9.21** *IBM PC Keyboard scan circuitry using a dedicated microprocessor. (IBM Corporation)*

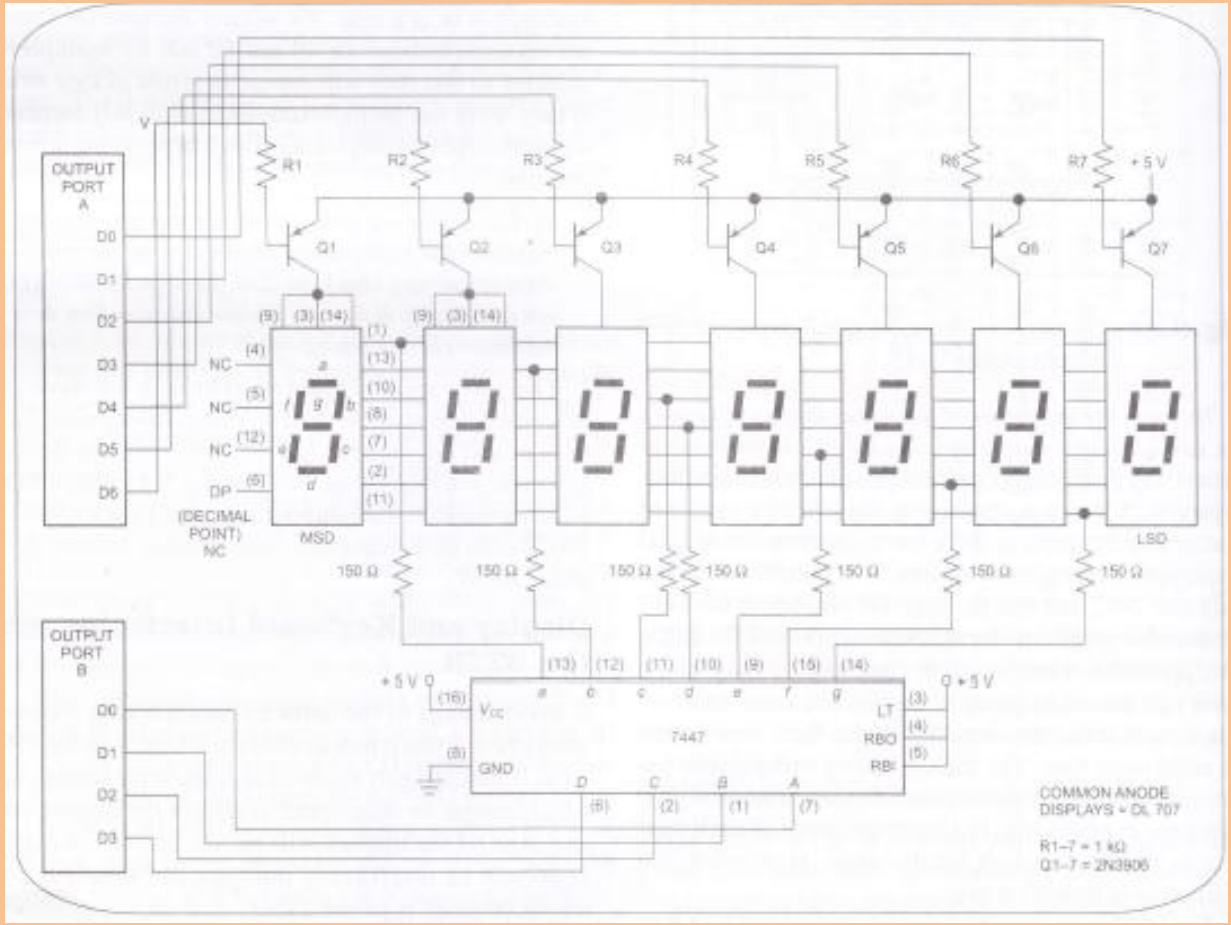
# Interfacing LED Displays to Microcomputers

- Figure shows a circuit that you might connect to a parallel port on a microcomputer to drive a single7-segment, common-anode display.



# Software-multiplexed Led Displays




- The immediately obvious advantages of multiplexing the displays are that only one 7447 is required, and only one digit is lit at a time

# Liquid-Crystal Display Interfacing.

- Figure 9.34 shows how an Intersil ICM7211M can be connected to drive a -digit, non -multiplexed, 7-segment LCD display.
- To display a character on one of the digits, you simply put the 4-bit hex code for that digit in the lower 4 bits of the AL register and output it to the system address that digit.
- The ICM7211M converts the 4-bit hex code r the required 7-segmenr code.

# Liquid-Crystal Display Interfacing.

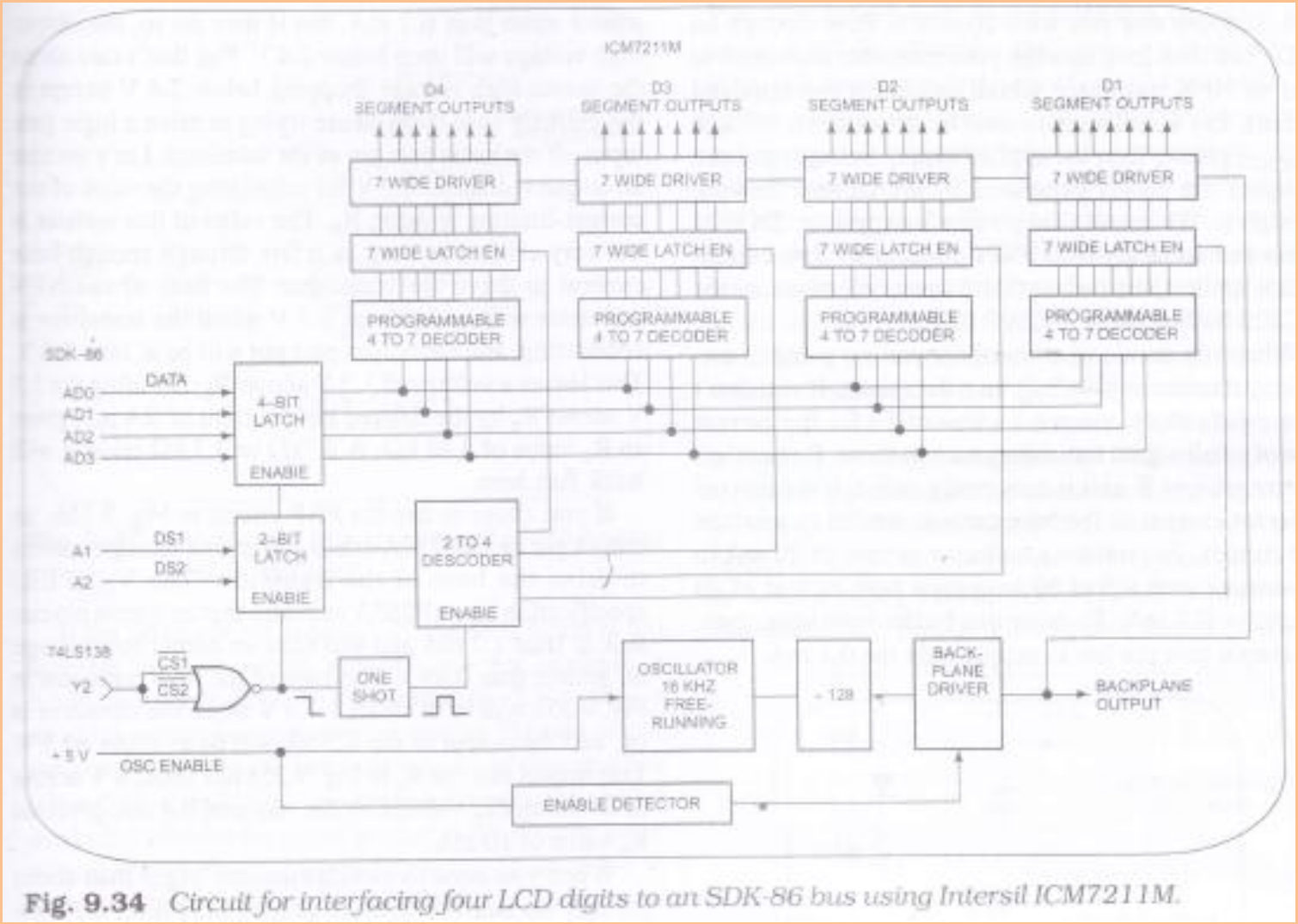



**Fig. 9.34** *Circuit for interfacing four LCD digits to an SDK-86 bus using Intersil ICM7211M.*

# CHAPTER

# ENDS