Unit 7
Input / Output Interfaces

We know that keyboard and displays are used as communication channel with outside world. So, it is necessary that we interface keyword and displays with MP. This is called I/O interfacing.

Serial Communication

→ It is a method of conveying a single bit at a time i.e. only one bit of a word is transmitted at a time.
→ It is slower.
→ Hardware requirement is simple.
→ E.g. RS-232C

Modes of serial data transfer

i) Simplex mode
→ Data travel in only one direction. E.g. from computer to printer

ii) Half duplex mode
→ Data travel in both directions but not at the same time.

iii) Full duplex mode
→ Data travel in both directions at the same time.

Types of serial data transfer

i) Synchronous
→ Both transmitter and receiver are synchronized by same clock pulse.
→ It is also called clock-oriented data transmission.
→ Speed: >20 Kbps
→ Always implemented with hardware.

ii) Asynchronous
→ Both transmitter and receiver are synchronized by separate clock pulse.
→ It is also called character-oriented data transmission.
→ Speed: <20 Kbps
→ Always implemented with hardware and software.

Introduction to 8251A PCI (Programmable Communication Interface)

- The 8251A is a programmable serial communication interface chip designed for synchronous and asynchronous serial data communication.
- It supports the serial transmission of data.
- It is packed in a 28 pin DIP.
- It is also called USART (Universal Synchronous Asynchronous Receiver Transmitter).

Pin	Description
D_0-D_7	Parallel data
$C/\overline{D}$	Control register or Data buffer select
$\overline{RD}$	Read control
$\overline{WR}$	Write control
$\overline{CS}$	Chip Select
CLK	Clock pulse (TTL)
RESET	Reset
$\overline{TxC}$	Transmitter Clock
TxD	Transmitter Data
$\overline{RxC}$	Receiver Clock
RxD	Receiver Data
RxRDY	Receiver Ready
TxRDY	Transmitter Ready
$\overline{DSR}$	Data Set Ready
$\overline{DTR}$	Data Terminal Ready
SYNDET/ BRKDET	Synchronous Detect / Break Detect
$\overline{RTS}$	Request To Send Data
$\overline{CTS}$	Clear To Send Data
TxEMPTY	Transmitter Empty
V_{CC}	Supply (+5V)
GND	Ground (0 V)

Fig: Pin Description

Block Diagram:

The functional block diagram of 8251A consists of five sections. They are:

- Read/Write control logic
- Transmitter
- Receiver
- Data bus buffer
- Modem control.

The functional block diagram is shown in fig:

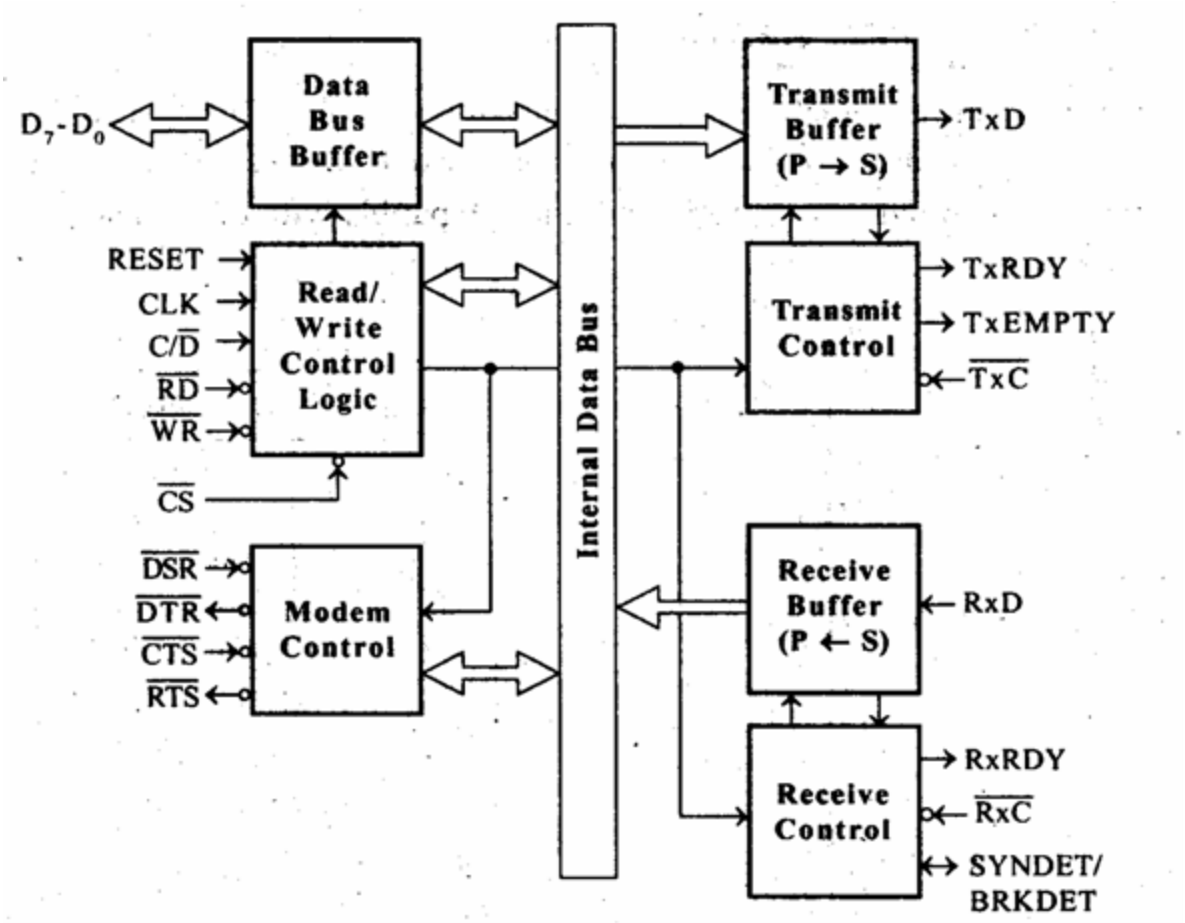


Fig: Functional block diagram of 8251A PCI

Read/Write control logic:

- The Read/Write Control logic interfaces the 8251A with CPU, determines the functions of the 8251A according to the control word written into its control register.
- It monitors the data flow.
- This section has three registers and they are control register, status register and data buffer.
- The eight parallel lines, D7-D0, connect to the system data bus so that data words and control/status words can be transferred to and from the device.
- The chip select (CS) input is connected to an address decoder so the device is enabled when addressed.
- The signals RD, WR, CS and C/D are used for read/write operations with these three registers.
- It has two internal addresses, a control address which is selected when C/D is high (1), and a data address which is selected when C/D input is low (0).
- When the RESET is high, it forces 8251A into the idle mode.
- The CLK (clock input) is necessary for 8251A for communication with CPU and this clock does not control either the serial transmission or the reception rate.

Transmitter section:

- The transmitter section accepts parallel data from CPU and converts them into serial data.
- The transmitter section is double buffered, i.e., it has a ***buffer register*** to hold an 8-bit parallel data and another register called ***output register*** to convert the parallel data into serial bits.
- When output register is empty, the data is transferred from buffer to output register. Now the processor can again load another data in buffer register.
- If buffer register is empty, then TxRDY goes high.
- If output register is empty then TxEMPTY goes high.
- The clock signal, TxC controls the rate at which the bits are transmitted by the USART.
- The clock frequency can be 1, 16 or 64 times the baud rate.

Receiver Section:

- The receiver section accepts serial data and convert them into parallel data
- The receiver section is double buffered, i.e., it has an ***input register*** to receive serial data and convert to parallel, and a ***buffer register*** to hold the parallel data.
- When the RxD line goes low, the control logic assumes it as a START bit, waits for half a bit time and samples the line again.
- If the line is still low, then the input register accepts the following bits, forms a character and loads it into the buffer register.
- The CPU reads the parallel data from the buffer register.
- When the input register loads a parallel data to buffer register, the RxRDY line goes high.
- The clock signal RxC controls the rate at which bits are received by the USART.
- During asynchronous mode, the signal SYNDET/BRKDET will indicate the break in the data transmission.
- During synchronous mode, the signal SYNDET/BRKDET will indicate the reception of synchronous character.

MODEM Control:

- The MODEM control unit allows to interface a MODEM to 8251A and to establish data communication through MODEM over telephone lines.
- This unit takes care of handshake signals for MODEM interface.

Parallel Communication

→ It is a method of conveying multiple binary digits (bits) simultaneously i.e. all bits of word are transferred at a time.

→ It is faster.

→ Hardware requirement is complex.

→ E.g. 8255A PPI

Methods of parallel data transfer

i) Simple I/O

→ When you need to get digital data from simple switch, such as thermostat, into microprocessor, all you have to do is connect the switch to an I/O port line and read the port.

→ Likewise, when you need to output data to simple display device, such as LED, all you have to do is connect the input of the LED buffer on an output port pin and output the logical level required to turn on the light.

Fig: Simple I/O

→ The timing waveform represents the situation.

→ The crossed lines on the waveform represent the time at which a new data byte becomes valid on the output lines of the port.

→ The absence of other waveforms indicates that this output operation is not directly dependant on any other signal.

ii) 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.

→ E.g. the ASCII-encoded keyboard. When a key is pressed, circuitry on the keyboard sends out the ASCII code for the pressed key on eight parallel data lines, and then sends out a strobe signal on another line to indicate that valid data is present on the eight data lines.

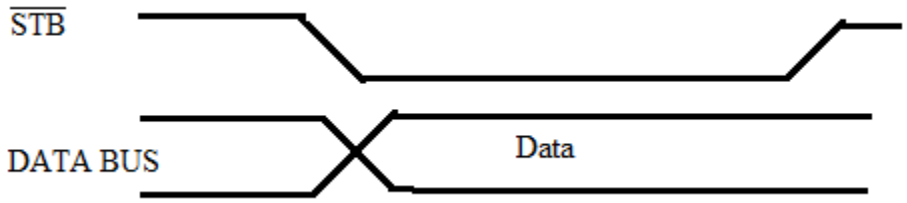


Fig: Strobe I/O

→ This timing waveform represents strobe I/O.
→ The sending device, such as a keyboard, outputs a parallel data on the data lines, and then outputs an STB signal to let you know that valid data is present.
→ For low rates of data transfer, such as from a keyboard to a MP, a simple strobe transfer works well.
→ 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.
→ In other words, the sending system might send data bytes faster than the receiving system could read them.
→ To prevent this problem, a handshake data transfer scheme is used.

iii) Single handshake I/O

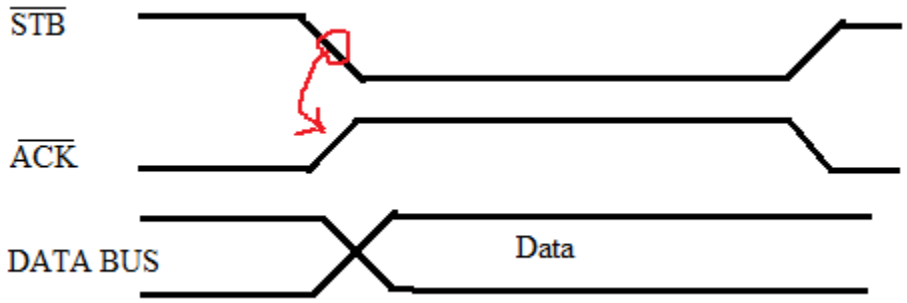


Fig: Single Handshake I/O

→ It shows the timing waveform for a handshake data transfer from a peripheral device to a MP.
→ The peripheral outputs some parallel data and sends an STB signal to the MP.
→ The MP detects the asserted STB signal on a polled or interrupts basis and reads in the bytes of data.
→ Then, the MP sends ACK (acknowledge) signal to the peripheral to indicate that the data has been read and that the peripheral can send next byte of data.
→ The point of this method is that the sending device or system is designed so that it does not send the next byte until the receiving device or system indicates with an ACK signal that it is ready to receive the next byte.

iv) Double handshake I/O

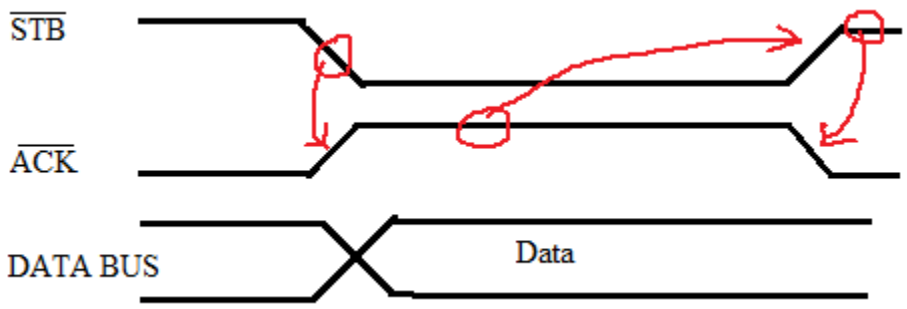


Fig: Double Handshake I/O

→ For data transfer where even more coordination is required between the sending system and the receiving system, a double handshake is used.
→ The sending (peripheral) device asserts its STB line low to ask the receiving device whether it is ready or not for data reception.
→ The receiving system raises its ACK line high to indicate that it is ready.
→ The peripheral device then sends the byte of data and raises its STB line high to assure that the valid data is available for the receiving device (MP).
→ When MP reads the data, it drops its ACK line low to indicate that it has received the data and requests the sending system to send next byte of data.

Introduction to 8255A PPI (Programmable Peripheral Interface)

8255A is widely used programmable parallel I/O device. It can be programmed to transfer data under various conditions, from simple I/O to interrupt I/O. It is flexible, versatile and economical (when multiple I/O ports are required), but somewhat complex. It is an important general purpose I/O device that can be used with almost any microprocessor.

The 8255A has 24 input output pins that can be grouped primarily in two 8 bits parallel ports: A and B, with the remaining 8 bits as port C. The 8 bits of port C can be used as individual bits or be grouped in two four bits ports: C_{upper} (C_U) and C_{lower} (C_L) as in figure (a). The functions of these ports are defined by writing a control word in the control registers.

Block Diagram

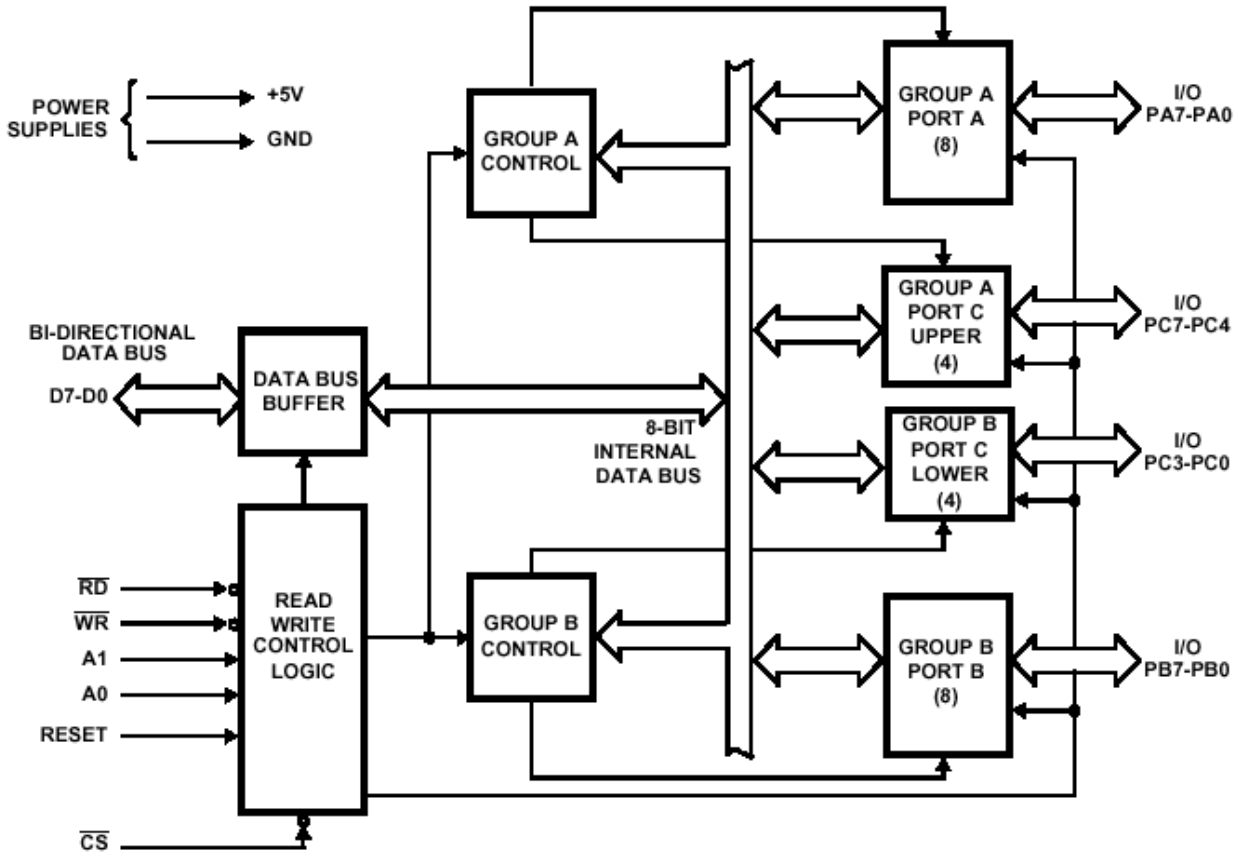


Fig (a): Internal Block Diagram of 8255A PPI

Data Bus Buffer
This three-state bi-directional 8-bit buffer is used to interface the 8255 to the system data bus. Data is transmitted or received by the buffer upon execution of input or output instructions by the CPU. Control words and status information are also transferred through the data bus buffer.
Read/Write and Control Logic
The function of this block is to manage all of the internal and external transfers of both Data and Control or Status words. It accepts inputs from the CPU Address and Control busses and in turn, issues commands to both of the Control Groups.
(CS) Chip Select. A "low" on this input pin enables the communication between the 8255 and the CPU.
(RD) Read. A "low" on this input pin enables 8255 to send the data or status information to the CPU on the data bus. In essence, it allows the CPU to "read from" the 8255.
(WR) Write. A "low" on this input pin enables the CPU to write data or control words into the 8255.
(A0 and A1) Port Select 0 and Port Select 1. These input signals, in conjunction with the RD and WR inputs, control the selection of one of the three ports or the control word register. They are normally connected to the least significant bits of the address bus (A0 and A1).
(RESET) Reset. A "high" on this input initializes the control register to 9Bh and all ports (A, B, C) are set to the input mode.

A1	A0	SELECTION
0	0	PORT A
0	1	PORT B
1	0	PORT C
1	1	CONTROL REGISTER

Group A and Group B Controls
The functional configuration of each port is programmed by the systems software. In essence, the CPU "outputs" a control word to the 8255. The control word contains information such as "mode", "bit set", "bit reset", etc., that initializes the functional configuration of the 8255. Each of the Control blocks (Group A and Group B) accepts "commands" from the Read/Write Control logic, receives "control words" from the internal data bus and issues the proper commands to its associated ports.
Ports A, B, and C
The 8255 contains three 8-bit ports (A, B, and C). All can be configured to a wide variety of functional characteristics by the system software but each has its own special features or "personality" to further enhance the power and flexibility of the 8255.
Port A One 8-bit data output latch/buffer and one 8-bit data input latch. Both "pull-up" and "pull-down" bus-hold devices are present on Port A.
Port B One 8-bit data input/output latch/buffer and one 8-bit data input buffer.
Port C One 8-bit data output latch/buffer and one 8-bit data input buffer (no latch for input). This port can be divided into two 4-bit ports under the mode control. Each 4-bit port contains a 4-bit latch and it can be used for the control signal output and status signal inputs in conjunction with ports A and B.

Modes of Operation
The 8255A is primarily operated in two modes: I/O (input-output) mode and the BSR (Bit-Set-Reset) mode. The I/O mode is further grouped into Mode 0 (Simple I/O interfacing), Mode 1 (Interfacing with handshake signals) and Mode 2 (Bidirectional I/O interfacing).
Figure (b) shows all the functions of 8255A, classifying according to two modes: the Bit Set-Reset (BSR) mode and Input Output (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, mode 1 and mode 2. In mode 0, all ports function as simple I/O ports. Mode 1 is a handshake mode whereby ports A and ports B use bits from port C as handshake signals. In the handshake mode, two types of data transfer can be implemented: status check and interrupt. In mode 2, port A can be set up

for bidirectional data transfer using handshake signal from port C, and port B can be set up either in mode 0 or mode 1.

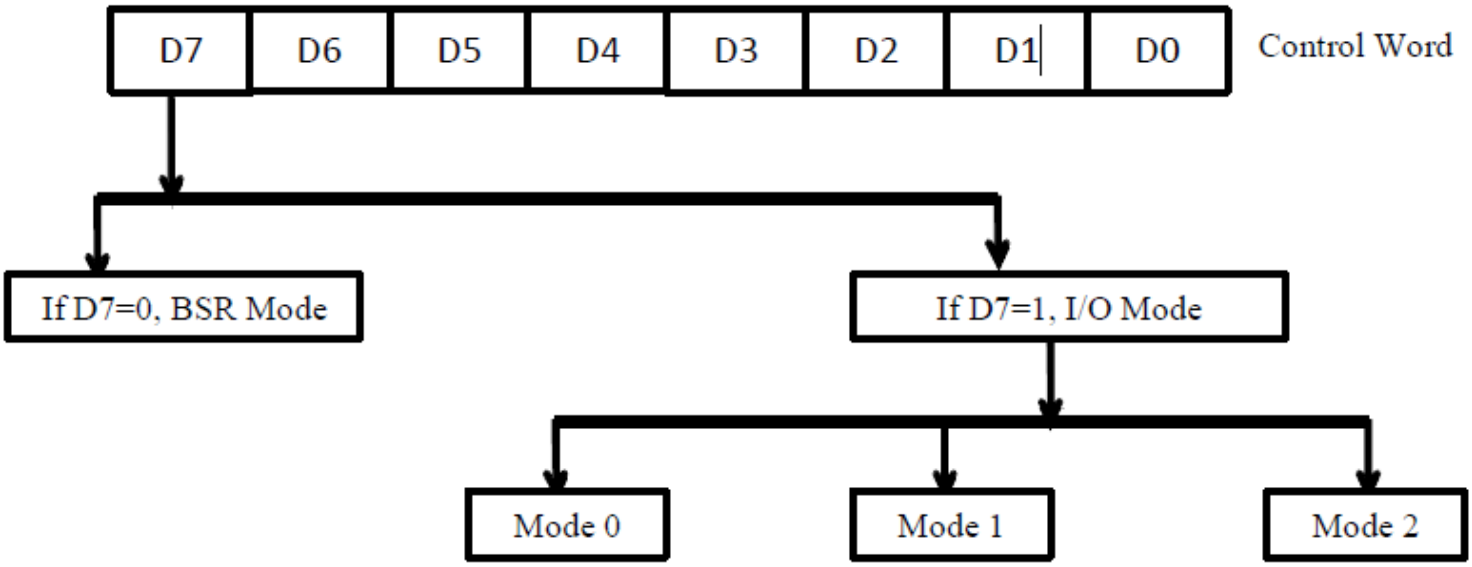


Fig (b): Control word specifying various modes

When D7=0, BSR mode

→ For port C

→ No effect on I/O mode and functions of port A and B

→ Individual bits of port C can be used for applications such as ON/OFF switch

When D7=1, I/O mode

i) Mode 0

→ Simple I/O interfacing for port A, B and C

ii) Mode 1

→ Interfacing with handshake signals for port A and B

→ Port C bits are used for handshake

iii) Mode 2

→ Bidirectional I/O interfacing for port A

→ Port B: either in mode 0 or mode 1

→ Port C bits used for handshake

Control Word

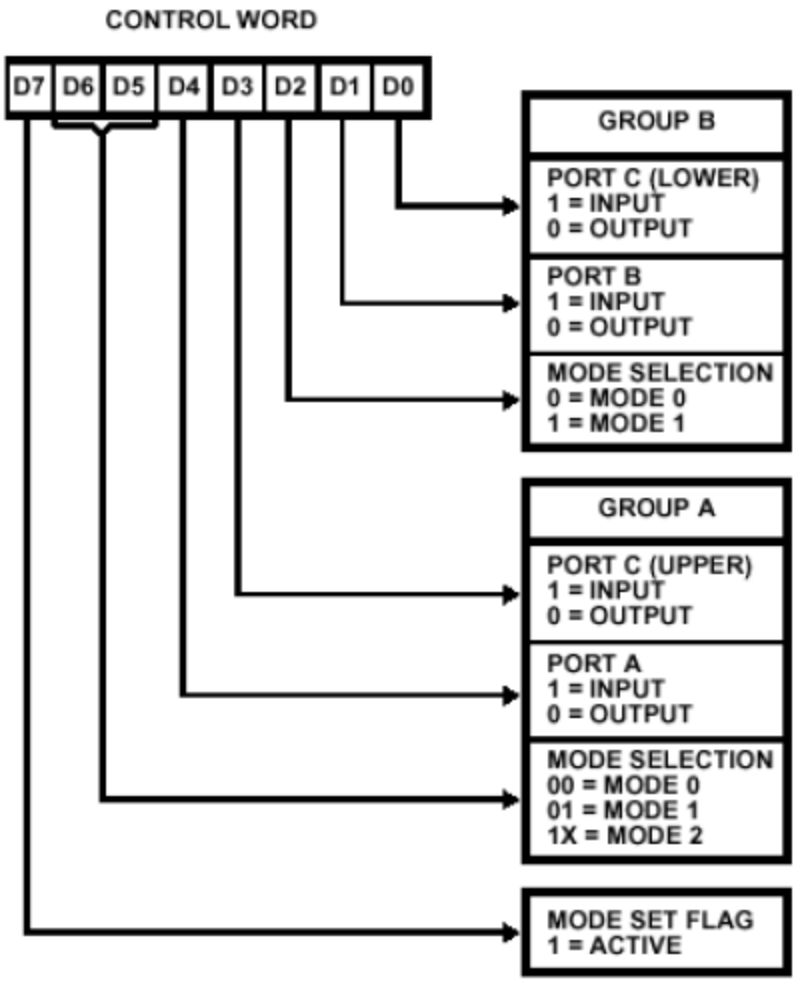


Fig: I/O Mode Definition Control Word Format

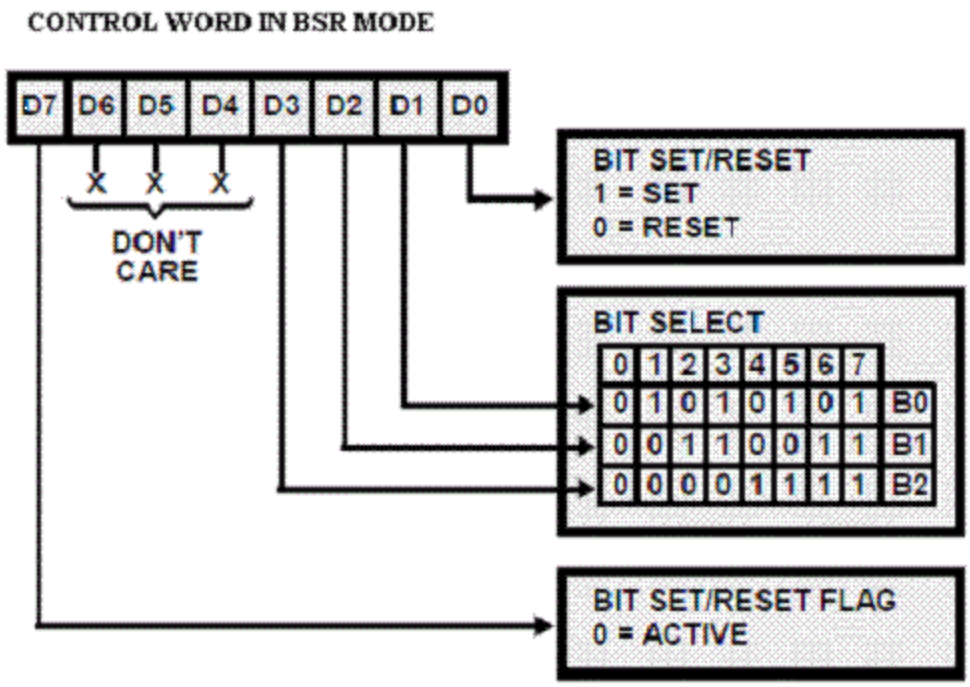


Fig: BSR Mode

- The content of control register is called control word specifying an input output functions for each port.
- The register can be accessed to write a control word when A_0 and A_1 are at logic 1. The register is not accessible for read operation.
- Bit D_7 of the control register specifies either I/O functions or Bit Set-Reset function as classified in figure (b).
- If bit D_7=1, bit D_6-D_0 determine I/O function in various modes as shown in figure (b).
- If bit D_7=0, port C operates in Bit Set-Reset mode.
- The BSR control word does not affect the function of port A and port B.

RS-232C

In 8251A (USART), we discussed how serial communication takes place. The TTL signals output by a USART, however, are not suitable for transmission over long distances, so these signals are converted to some other form to be transmitted. In this section we discuss device used to send serial data signals over long distances.

→ RS232 is the most widely used serial I/O interfacing standard.

→ However the I/O voltage levels are not TTL compatible. In the RS232, a 1 is represented by –3 to –25 V, while 0 bit is +3 to +25 V, making –3 to +3 undefined.

→ For this reason voltage converter such as MC1488 and MC1489 are used to convert the TTL logic levels to the RS232 voltage levels and vice versa. See Figure below.

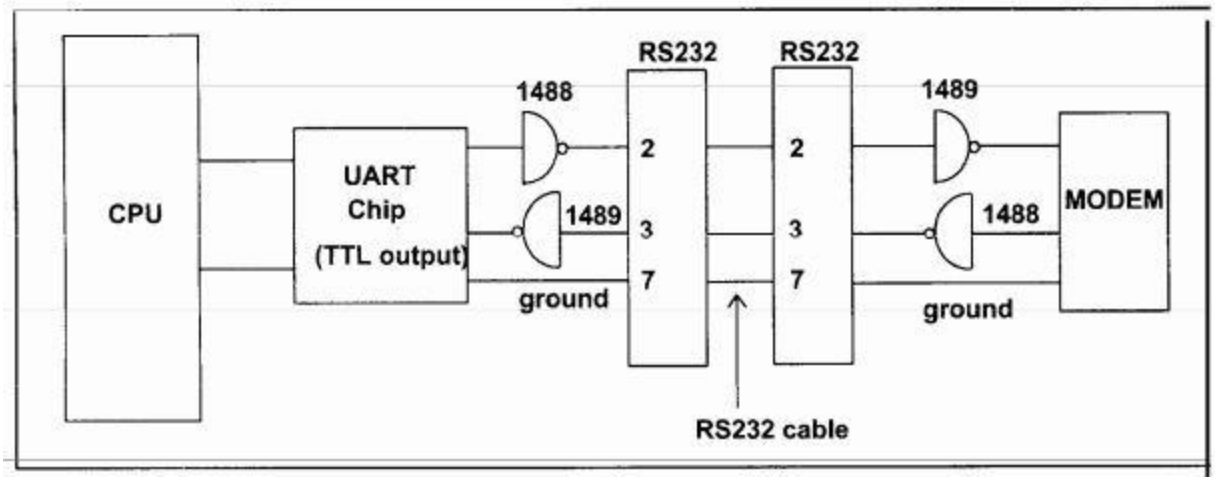


Fig: RS232 connection

RS-232 stands for Recommend Standard number 232 and C is the latest revision of the standard. The serial ports on most computers use a subset of the RS-232C standard. The full RS-232C standard specifies a 25-pin "D" connector of which 22 pins are used. Most of these pins are not needed for normal PC communications, and indeed, most new PCs are equipped with male D type connectors having only 9 pins.

DCE and DTE Devices

Two terms you should be familiar with are DTE and DCE. DTE stands for Data Terminal Equipment, and DCE stands for Data Communications Equipment. These terms are used to indicate the pin-out for the connectors on a device and the direction of the signals on the pins. Your computer is a DTE device, while most other devices are usually DCE devices.

If you have trouble keeping the two straight then replace the term "DTE device" with "your PC" and the term "DCE device" with "remote device" in the following discussion.

The RS-232 standard states that DTE devices use a 25-pin male connector, and DCE devices use a 25-pin female connector. You can therefore connect a DTE device to a DCE using a straight pin-for-pin connection. However, to connect two like devices, you must instead use a null modem cable. Null modem cables cross the transmit and receive lines in the cable, and are discussed later in this chapter. The listing below shows the connections and signal directions for both 25 and 9-pin connectors.

25 Pin Connector on a DTE device (PC connection)

Male RS232 DB25 

Pin Number	Direction of signal:
1	Protective Ground
2	Transmitted Data (TD) Outgoing Data (from a DTE to a DCE)
3	Received Data (RD) Incoming Data (from a DCE to a DTE)
4	Request To Send (RTS) Outgoing flow control signal controlled by DTE
5	Clear To Send (CTS) Incoming flow control signal controlled by DCE
6	Data Set Ready (DSR) Incoming handshaking signal controlled by DCE
7	Signal Ground Common reference voltage
8	Carrier Detect (CD) Incoming signal from a modem
20	Data Terminal Ready (DTR) Outgoing handshaking signal controlled by DTE
22	Ring Indicator (RI) Incoming signal from a modem

9 Pin Connector on a DTE device (PC connection)

Male RS232 DB9 

Pin Number	Direction of signal:
1	Carrier Detect (CD) (from DCE) Incoming signal from a modem
2	Received Data (RD) Incoming Data from a DCE
3	Transmitted Data (TD) Outgoing Data to a DCE
4	Data Terminal Ready (DTR) Outgoing handshaking signal
5	Signal Ground Common reference voltage
6	Data Set Ready (DSR) Incoming handshaking signal
7	Request To Send (RTS) Outgoing flow control signal
8	Clear To Send (CTS) Incoming flow control signal
9	Ring Indicator (RI) (from DCE) Incoming signal from a modem

Interconnection between DTE-DTE and DTE-DCE

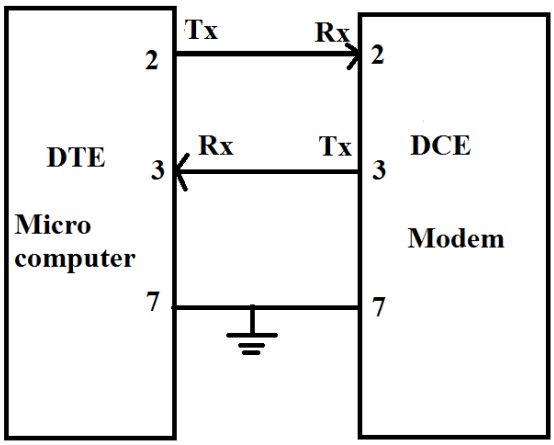


Fig (a):-DTE to DCE

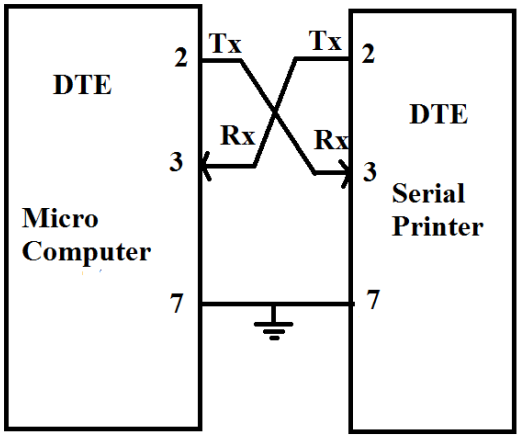


Fig (b):-DTE to DTE (null modem)

- The minimum interface between a computer and a peripheral requires 3 lines: pins 2, 3 and 7 as shown in figure.
- These lines are defined in relation to DTE; the terminal transmits on pin 2 and receives on pin 3. On the other hand, the DCE transmits on pin 3 and receives on pin 2.
- Now the dilemma is: how does a manufacturer define the role of its equipment? For example, the user may connect its microcomputer to serial printer configured as DTE. Therefore, to remain compatible with the defined signals of RS-232C, the RS-232C cable must be reconfigured as shown in figure (b) above.
- In figure (a), the microcomputer is defined as a DTE, and it can be connected to the modem defined as a DCE, without any modification in RS-232C cable.
- However, when it is connected to the printers, the transmitted and received lines must be crossed as shown in figure (b). This is also known as Null modem connection.

RS 232 Handshaking

In order that data can be exchanged on an RS 232 link, the control signals must indicate that the equipment at either end of the link is ready to send the data and ready to receive the data. This can be achieved in a number of ways, but one of the more common is to use the RTS, CTS, and DTR lines.

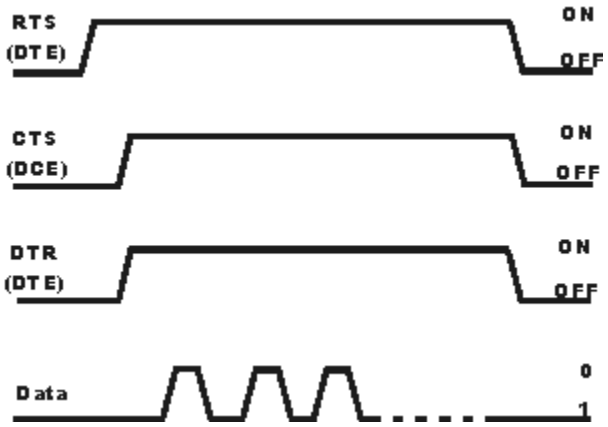
These lines are found in the Data Terminal Equipment, DTE and Data Communications Equipment, DCE as follows:

LINE ABBREVIATION	LINE NAME	EQUIPMENT
RTS	Request to Send	DTE
CTS	Clear to Send	DCE
DTR	Data Terminal Ready	DTE

The handshaking exchange to start the data flow is quite straightforward and can be seen as a number of distinct stages:

1. RTS is put in the ON state by the DTE.
2. The DCE then put the CTS line into the ON state.
3. The DTE then responds by placing the DTR line into the ON state.
4. The DTR line remains on while data is being transmitted.

At the end of the transmission, DTR and RTS are pulled to the OFF state and then the DCE pulls the CTS line to the OFF state. This series of handshake controls was devised to allow the DTE to request control of the communications link from the related modem, and then to let the modem inform the terminal equipment that the control has been acquired. In this way the communications will only take place when both ends of the link are ready.



RS232 handshaking signals

The RS 232 data communications standard is a reliable for of data communications which has been used for many years and shows every sign of being used for many years to come. In order that it is able to communicate satisfactorily the RS 232 signals and voltage levels must be able to ensure that the line receivers are able to decode the data with no errors and that the communications protocols are adhered to. Once these are all established, data can be exchanged reliably and efficiently.

Keyboard and Display Controller: Introduction to 8279

The INTEL 8279 is specially developed for interfacing keyboard and display devices to 8085/8086/8088 microprocessor based system. The important features of 8279 are:

- Simultaneous keyboard and display operations.
- Scanned keyboard mode.
- Scanned sensor mode.
- 8-character keyboard FIFO.
- 16-character display.
- Right or left entry 16-byte display RAM.
- Programmable scan timing.

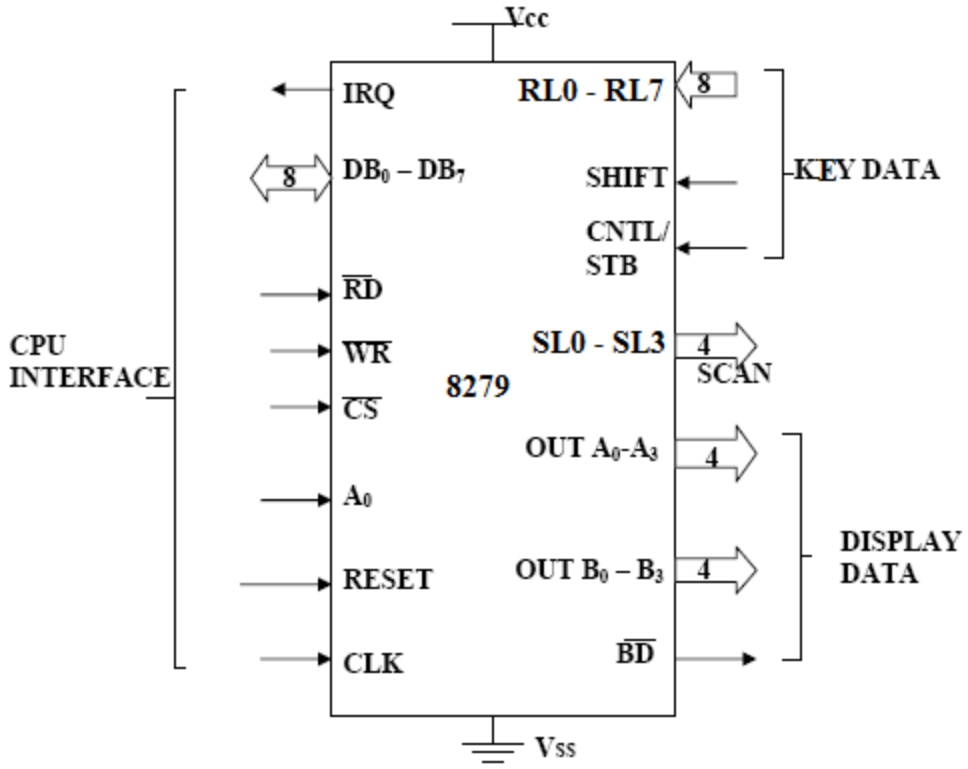


Fig: Pins of 8279

Block diagram of 8279

- The functional block diagram of 8279 is shown.
- The four major sections of 8279 are keyboard, scan, display and CPU interface.

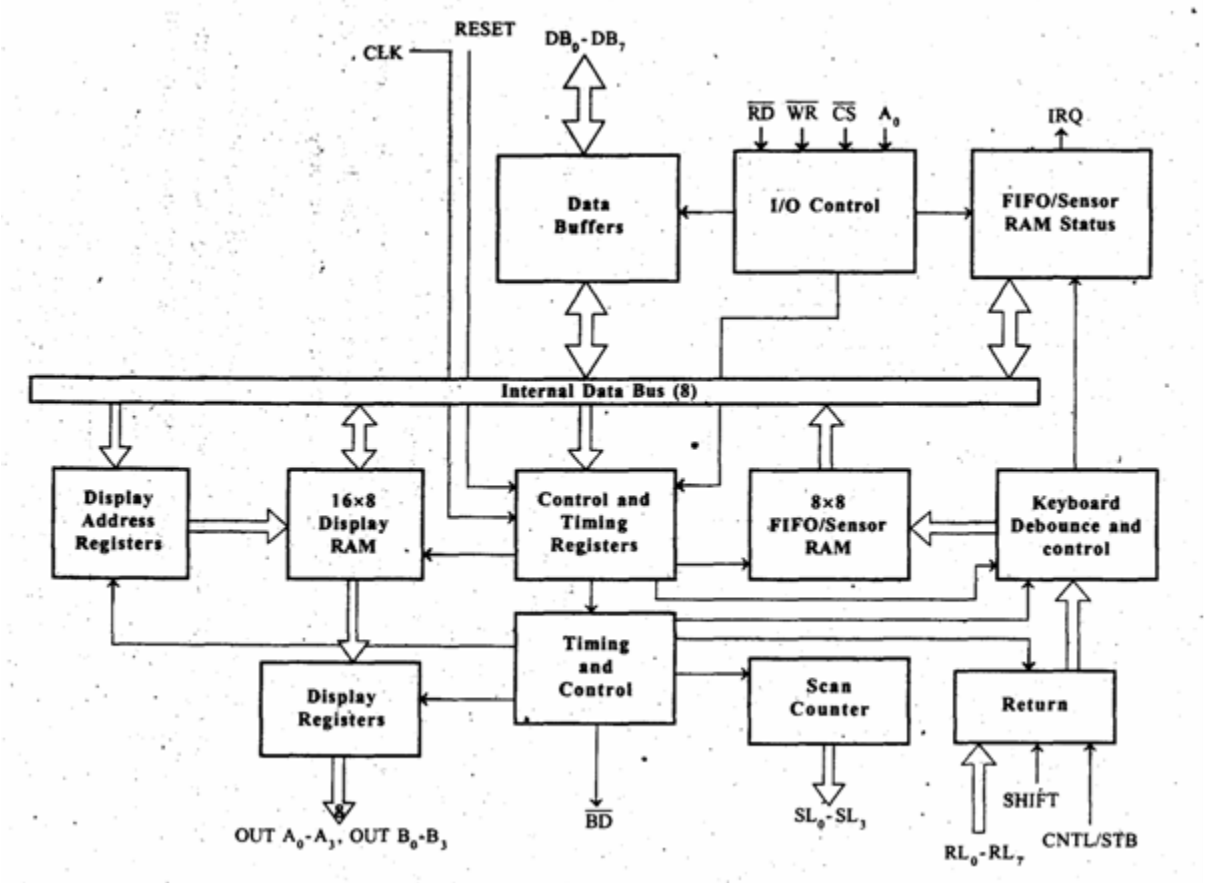
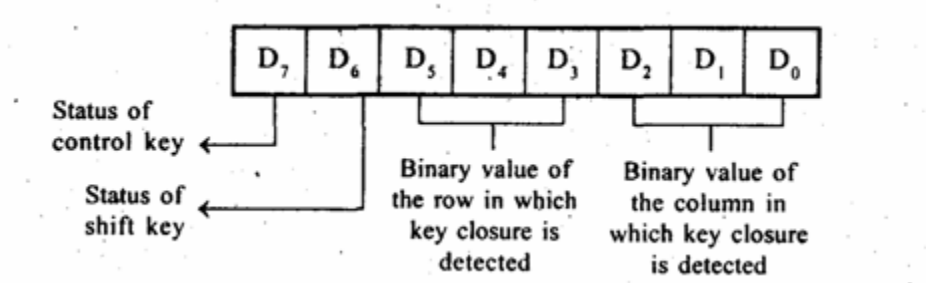


Fig: Functional block diagram of 8279

Keyboard section:

- The keyboard section consists of eight return lines RL0 - RL7 that can be used to form the columns of a keyboard matrix.
- It has two additional inputs: shift and control/strobe. The keys are automatically debounced.
- The two operating modes of keyboard section are 2-key lockout and N-key rollover.
- In the 2-key lockout mode, if two keys are pressed simultaneously, only the first key is recognized.
- In the N-key rollover mode, simultaneous keys are recognized and their codes are stored in FIFO.
- The keyboard section also has an 8x8 FIFO (First-In-First-Out) RAM.
- The FIFO can store eight key codes in the scan keyboard mode. The status of the shift key and control key are also stored along with key code. The 8279 generate an interrupt signal when there is an entry in FIFO. The format of key code entry in FIFO for scan keyboard mode is,



- In sensor matrix mode, the condition (i.e., open/close status) of 64 switches is stored in FIFO RAM. If the condition of any of the switches changes, then the 8279 asserts IRQ (interrupt request) as high to interrupt the processor.

Display section:

- The display section has eight output lines divided into two groups A0-A3 and B0-B3.
- The output lines can be used either as a single group of eight lines or as two groups of four lines, in conjunction with the scan lines for a multiplexed display.
- The output lines are connected to the anodes through driver transistor in case of common cathode 7-segment LEDs.
- The cathodes are connected to scan lines through driver transistors.
- The display can be blanked by BD (low) line.
- The display section consists of 16x8 display RAM. The CPU can read from or write into any location of the display RAM.

Scan section:

- The scan section has a scan counter and four scan lines, SL0 to SL3.
- In **decoded scan mode**, the output of scan lines will be similar to a 2-to-4 decoder.
- In **encoded scan mode**, the output of scan lines will be binary count, and so an external decoder should be used to convert the binary count to decoded output.
- The scan lines are common for keyboard and display.
- The scan lines are used to form the rows of a matrix keyboard and also connected to digit drivers of a multiplexed display, to turn ON/OFF.

CPU interface section:

- The CPU interface section takes care of data transfer between 8279 and the processor.
- This section has eight bidirectional data lines DB0 to DB7 for data transfer between 8279 and CPU.
- It requires two internal address A =0 for selecting data buffer and A = 1 for selecting control register of 8279.
- The control signals WR (low), RD (low), CS (low) and A0 are used for read/write to 8279.
- It has an interrupt request line IRQ, for interrupt driven data transfer with processor.

- The 8279 require an internal clock frequency of 100 kHz. This can be obtained by dividing the input clock by an internal prescaler.
- The RESET signal sets the 8279 in 16-character display with two-key lockout keyboard modes.

Notes:

Synchronous Serial Communication

Synchronous communication requires common timing signals and common clock for synchronization. This means that as well as the data, the clock signals must also be transmitted along. In synchronous method a block of data at a time is transmitted.

Asynchronous Serial Communication and data framing

The data coming in the receiving end of the data line in a serial data transfer is all 1s and 0s; it is difficult to make sense of data unless the sender and receiver agree on a set of rules, a protocol on how the data is packed, how many bits constitute a character, and when the data begins and ends.

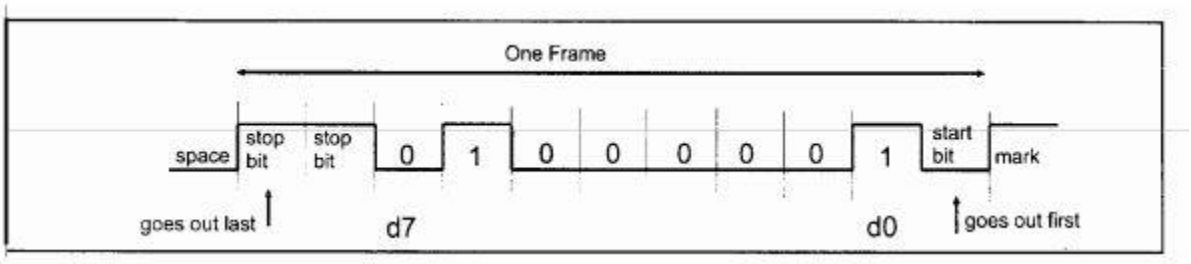
Start and Stop bits

→ Asynchronous serial data communication is widely used for character-oriented transmission, and block-oriented data transfers use the Synchronous method.

→ In the Asynchronous method, each character is put between a start and stop bits. This is called **framing**. In data framing, for asynchronous communications, the data such as ASCII characters are packed in between a start bit and stop bit.

→ The start bit is always one bit, but the stop bit(s) can be one or two bits. The start bit is always a **0 (low)** and the stop bit(s) is **1 (high)**.

→ The example below gives the framing of the ASCII character "A", where, binary 0100 0001, is framed in between the start bit and two stop bits. Notice that **the LSB is sent out first**.



Framing of ASCII "A" (41H)

→ Notice that the transmission begins with a start bit followed by D0, the LSB, and then the rest of the bits until the MSB (D7), and finally, the 2 stop bits indicating the end of character "A".

→ In asynchronous serial communication, peripheral chips and modems can be programmed for data that is 5, 6, 7 or 8 bits wide.

→ In some systems in order to maintain the data integrity, the **parity bit** of the character byte is included in the data frame.