UNIT 6

INPUT/ OUTPUT INTERFACES

Parallel communication

- Method of conveying several binary bits simultaneously i.e. all bits of word are transferred at a time.
- Higher data rate/more parallel wires.
- Maximize bandwidth.
- Hardware requirement is complex.
- E.g. 8255A PPI

Synchronizing the computer with peripherals:

- The information exchanged between a microprocessor and an I/O interface circuit consists of input or output data and control information.
- The status signal enable the microprocessor to monitor the device and when it is ready then send or receive data.
- Control information is the command by microprocessor to cause I/O device to take some action.
- If the device operates at different speeds, then microprocessor can be used to select a particular speed of operation of the device.
- The techniques used to transfer data between different speed devices and computer is called ***synchronizing.***

Different techniques under synchronizing are:

1. **Simple I/O:**
   - The buffer switch and latch switches i. E. LED are always connected to the input and output ports.
   - The devices are always ready to send or receive data.
   - Here, ***cross line*** indicate the time for new valid data.

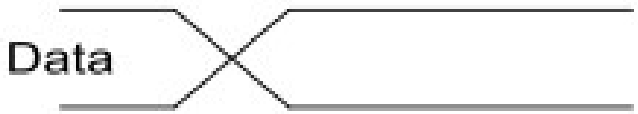


Fig: Simple I /O

2. Wait Interface(Simple strobe I/O):

- MP need to wait until the device is ready for the operation.
- 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.

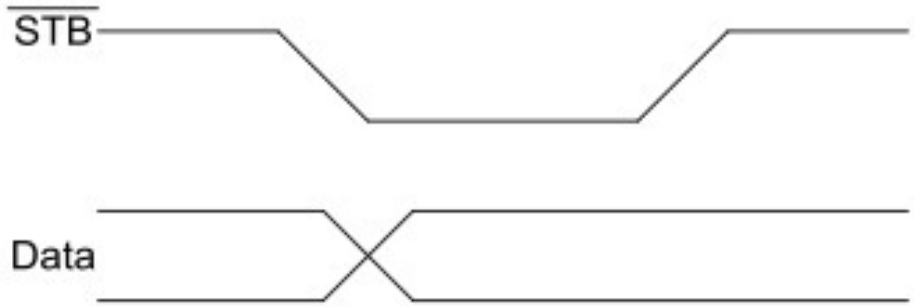


Fig: Simple Strobe I/O

Wait Interface(Simple strobe I/O): (cont...)

- 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.

3. Single Handshaking:

- The peripheral outputs some data and send signal to MP, “here is the data for you.”
- MP detects asserted signal, reads the data and sends an acknowledge signal (ACK) to indicate data has been read and peripheral can send next data. “I got that one, send me another.”
- MP sends or receives data when peripheral is ready.

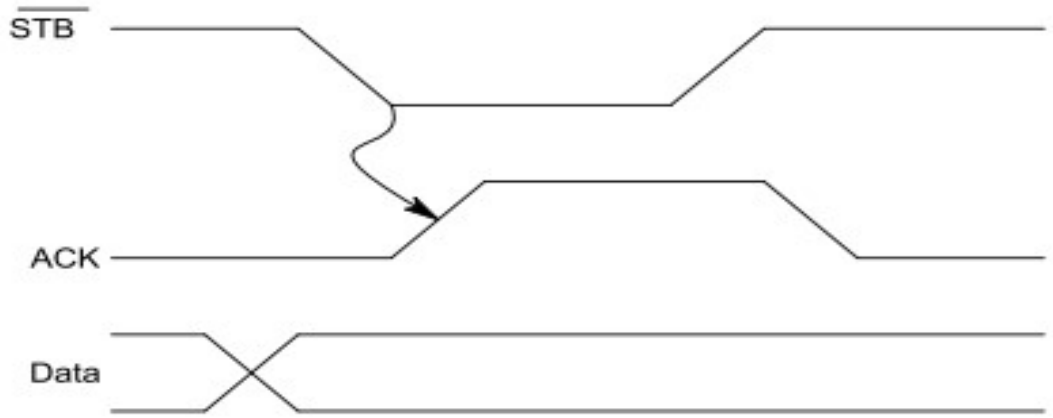


Fig: Single Handshaking

4. Double Handshaking:

- The peripheral asserts its line low to ask MP “Are you ready?”
- The MP raises its ACK line high to say “ I am ready”.
- Peripheral then sends data and raises its line low to say “Here is some valid data for you.”
- MP then reads the data and drops its ACK line to say, “I have the data, thank you, and I await your request to send the next byte of data.”

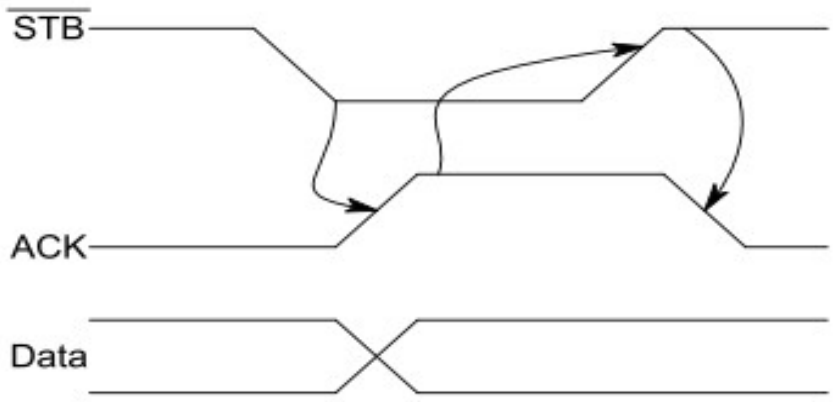


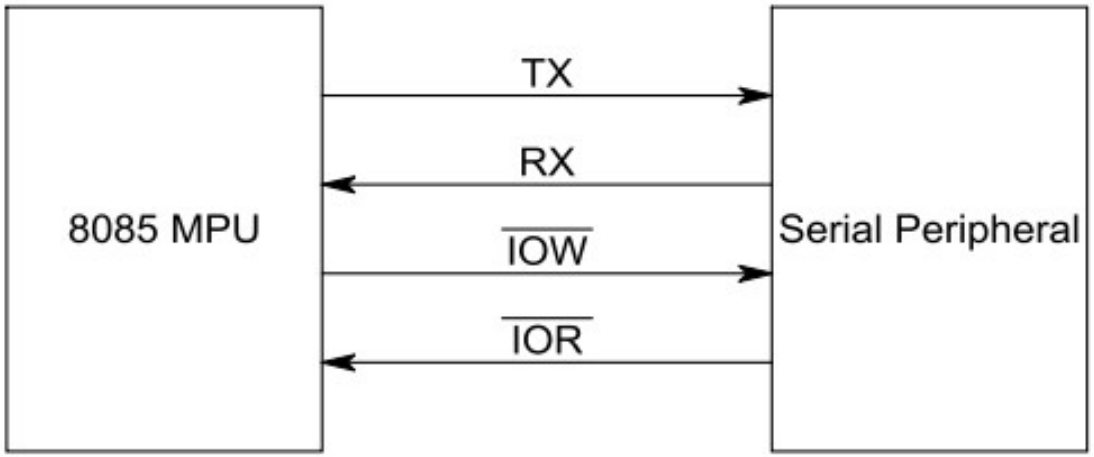
Fig: Double Handshaking

Serial Communication

- Data are transferred serially one bit at a time starting from Least Significant bit.
- Slow due to single communication link but inexpensive to implement.
- It uses clock to separate consecutive bits.
- Its function is to deal with the data on the bus in the parallel mode and communicate with the connected device in serial mode.
- Its data bus has n data lines, the serial I/O interface accepts n bit of data simultaneously from the bus and n bits are sent one by one at a time thus requiring n time slots.
- Not suitable for fast operation, needed microprocessor.

Serial Communication (cont..)

- Hardware requirement is simple.
- E.g. RS-232C



- **Serial I/O transfer** is **more common** than the **parallel I/O**. The two major forms of serial data transmission are:
  1. **Synchronous data transfer**
  2. **Asynchronous data transfer**

Serial Communication (cont..)

1. **Synchronous**
   - Both transmitter and receiver are synchronized by same clock pulse.
   - It is also called clock-oriented data transmission.
   - More complex interface (high data rates supported up to ~ 10 Gbps)
   - Always implemented with hardware.
2. **Asynchronous**
   - No clock sent (Tx & Rx have own clocks)
   - It is also called character-oriented data transmission.
   - Requires start and stop bits which provides byte timing and increases overhead
   - Parity often used to validate correct reception.
   - Always implemented with hardware and software.
   - Simple interface (limited data rate, typically < 64 kbps)
   - Used for connecting: Printer, Terminal, Modem, home connections to the Internet

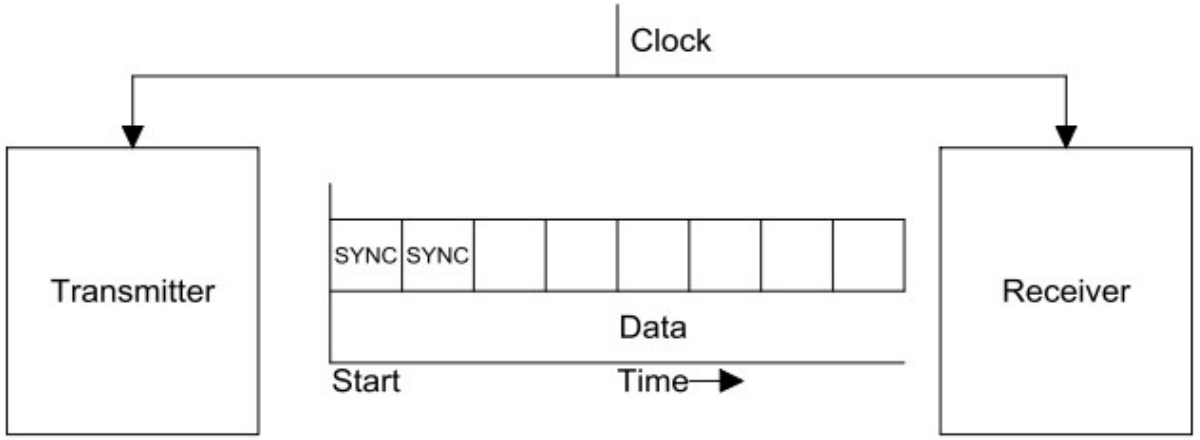


Fig: Synchronous Serial Transmission Format

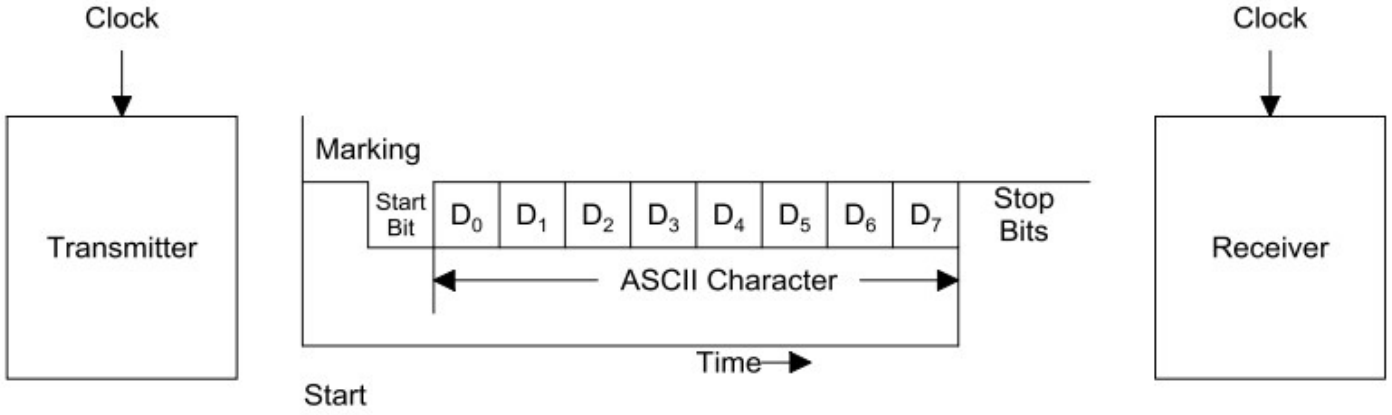


Fig: Asynchronous Serial Transmission Format

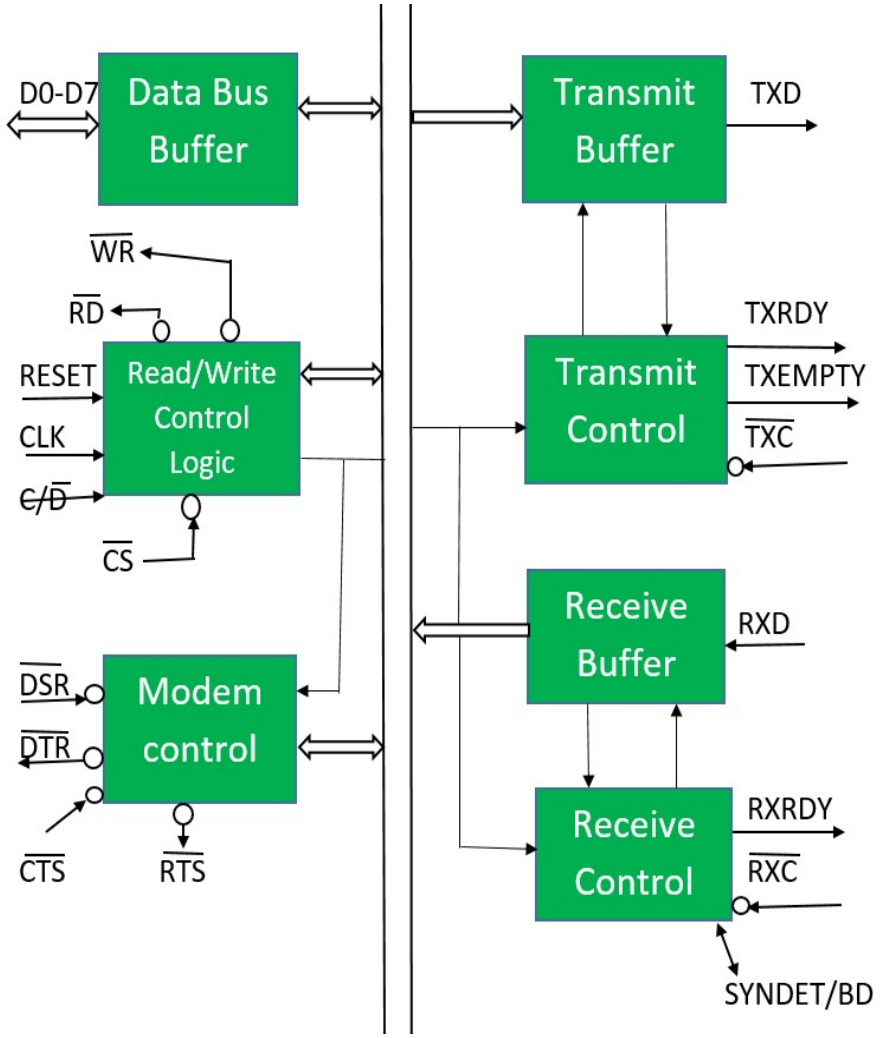
Serial Communication (cont..)

Modes of serial data transfer

1. **Simplex mode**
   - Data travel in only one direction. E.g. from computer to printer
2. **Half duplex mode**
   - Data travel in both directions but not at the same time.
3. **Full duplex mode**
   - Data travel in both directions at the same time.

Introduction to Programmable Communication Interface 8251

- Programmable serial communication interface chip designed for synchronous and asynchronous serial data communication.
- It is packed in a 28 pin DIP.
- It is also called USART (Universal Synchronous Asynchronous Receiver Transmitter).
- It takes data serially from peripheral (outside devices) and converts into parallel data.
- After converting the data into parallel form, it transmits it to the CPU.
- Similarly, it receives parallel data from microprocessor and converts it into serial form.
- After converting data into serial form, it transmits it to outside device (peripheral).



Pin	Description
D0-D7	Parallel data
C/$\overline{D}$	Control reg./Data Buffer reg.
$\overline{RD}$	Read Control
$\overline{WR}$	Write Control
$\overline{CS}$	Chip Select
$\overline{CTS}$	Clear to send
$\overline{RTS}$	Request to send
TXD	Transmit data
RXD	Receive data
RESET	Reset
$\overline{DSR}$	Data set ready
$\overline{DTR}$	Data terminal ready
SYNDET/BD	Synchronous detect/Break detect
$\overline{RXC}$	Receiver clock
$\overline{TXC}$	Transmitter clock
RXRDY	Receiver ready
TXRDY	Transmitter ready
TXEMPTY	Transmitter empty

Block diagram explanations:

- The functional block diagram of 8251A consists of following sections.
  - Data bus buffer
  - Read/Write control logic
  - Transmitter
  - Receiver
  - Modem control.

- **Data bus buffer:**
  - This block helps in interfacing the internal data bus of 8251 to the system data bus.
  - The data transmission is possible between 8251 and CPU by the data bus buffer block.

- **Read/Write control logic:**
  - ➢ It is a control block for overall device.
  - ➢ It controls the overall working by selecting the operation to be done.
  - ➢ The operation selection depends upon input signals as:

$\overline{CS}$	C/$\overline{D}$	$\overline{RD}$	$\overline{WR}$	Operation
1	X	X	X	Invalid
0	0	0	1	data CPU< ----- 8251
0	0	1	0	data CPU ----- > 8251
0	1	0	1	Status word CPU < ----------8251
0	1	1	0	Control word CPU---------- > 8251

- **Transmit buffer:**
  - This block is used for parallel to serial converter that receives a parallel byte for conversion into serial signal and further transmission onto the common channel.
    - **TXD:** It is an output signal, if its value is one, means transmitter will transmit the data.
- **Transmit control:**
  - This block is used to control the data transmission with the help of following pins:

1. **TXRDY:** It means transmitter is ready to transmit data character.
2. **TXEMPTY:** An output signal which indicates that TXEMPTY pin has transmitted all the data characters and transmitter is empty now.
3. **TXC:** An active-low input pin which controls the data transmission rate of transmitted data.

- **Receive buffer:**
  - This block acts as a buffer for the received data.
    - **RXD:** An input signal which receives the data.
- **Receive control:**
  - This block controls the receiving data.
  1. **RXRDY:** An input signal indicates that it is ready to receive the data.
  2. **RXC:** An active-low input signal which controls the data transmission rate of received data.
  3. **SYNDET/BD:** An input or output terminal. External synchronous mode-input terminal and asynchronous mode-output terminal.

- **Modem control (modulator/demodulator):**
  - A device converts analog signals to digital signals and vice-versa and helps the processor to communicate over telephone lines or cable wires.
  - The following are active-low pins of Modem:
    - **DSR:** Data Set Ready signal is an input signal.
    - **DTR:** Data terminal Ready is an output signal.
    - **CTS:** It is an input signal which controls the data transmit circuit.
    - **RTS:** It is an output signal which is used to set the status RTS.

8255A and it's Working

- **Programmable Peripheral Interface (PPI) - 8255A:**
  - The INTEL 8255 is a 24 pin device used to transfer parallel data between processor and slow peripheral devices like ADC, DAC, keyboard, 7-segment display, LCD, etc.
  - The 8255 has three ports: ***Port-A, Port-B and Port-C***.
    - ***Port-A*** can be programmed to work in any one of the three operating modes mode-0, mode-1 and mode-2 as input or output port.
    - ***Port-B*** can be programmed to work either in mode-0 or mode-1 as input or output port.
    - ***Port-C (8-pins)*** has different assignments depending on the mode of port-A and port-B.

- If port-A and B are programmed in mode-0, then the port-C can perform any one of the following functions:
  - As 8-bit parallel port in mode-0 for input or output.
  - As two numbers of 4-bit parallel ports in mode-0 for input or output.
  - The individual pins of port-C can be set or reset for various control applications.
- If port-A is programmed in mode- 1/mode-2 and port-B is programmed in mode-1 then some of the pins of port-C are used for handshake signals and the remaining pins can be used as input/ output lines or individually set/reset for control application.

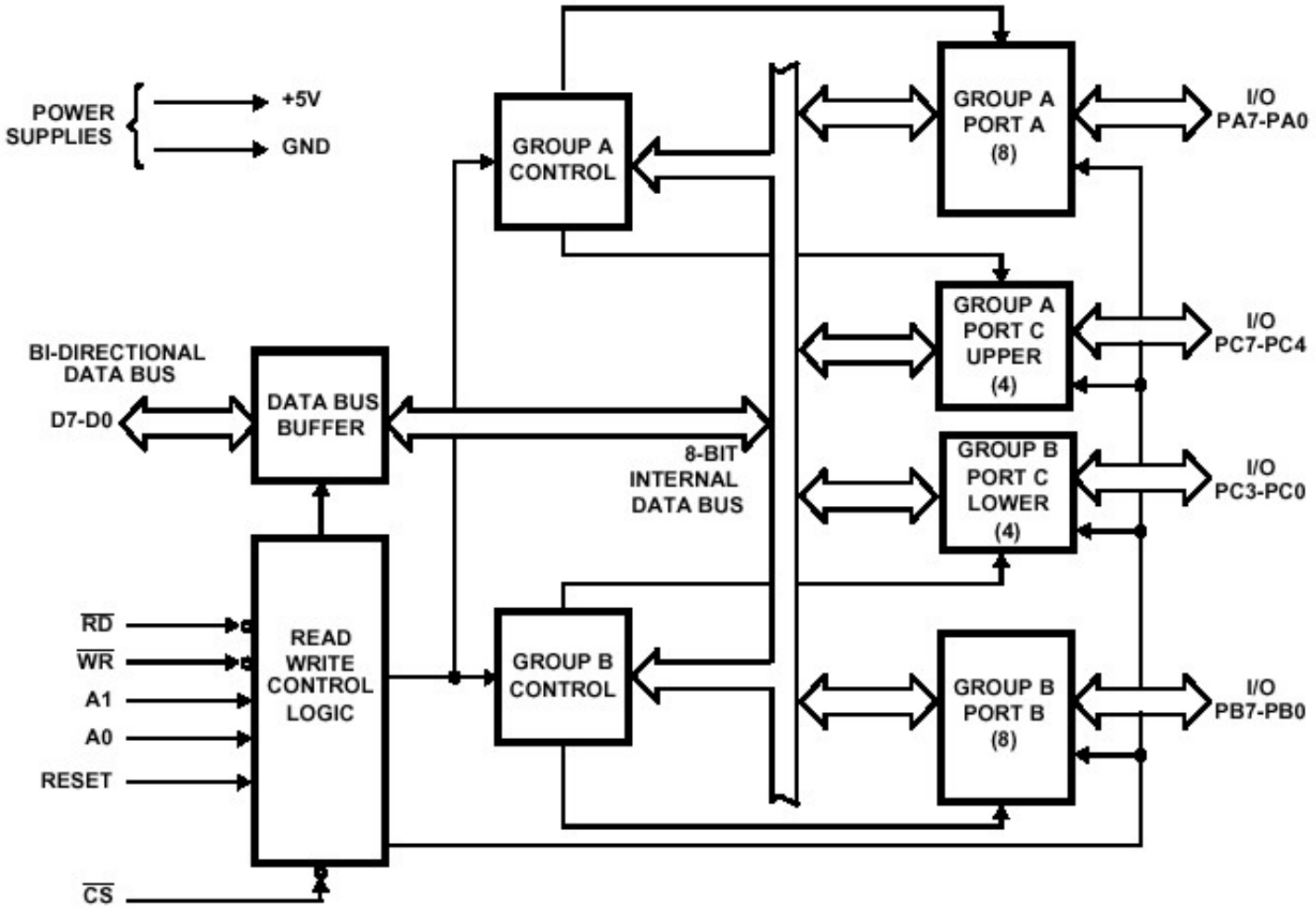


Fig: Internal Block Diagram of 8255A PPI

Features

- ***Mode 0:*** *Ports A and B operate as either inputs or outputs and Port C is divided into two 4-bit groups either of which can be operated as inputs or outputs.*
- ***Mode 1:*** *Same as Mode 0 but Port C is used for handshaking and control.*
- ***Mode 2:*** *Port A is bidirectional (both input and output) and Port C is used for handshaking. Port B is not used.*
- **Read/write 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.
    - ➢ RD (low): This control signal enables the read operation. When this signal is low, the microprocessor reads data from a selected I/O port of the 8255A.
    - ➢ WR (low): This control signal enables the write operation. When this signal goes low, the microprocessor writes into a selected I/O port or the control register.

- RESET: This is an active high signal. It clears the control register and set all ports in the input mode.
- CS (low), A0 and A1: These are device select signals.

Internal Devices	A_1	A_0
Port A	0	0
Port B	0	1
Port C	1	0
Control Register	1	1

- **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.
- **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.

Modes of Operation:

- The 8255A is primarily operated in **two modes**:
  1. I/O (input-output) mode and
  2. BSR (Bit-Set-Reset) mode.
- The I/O mode is further grouped into:
  1. Mode 0 (Simple I/O interfacing):
     - All ports function as simple I/O ports.
  2. Mode 1 (Interfacing with handshake signals):
     - Ports A and ports B use bits from port C as handshake signals.

     Two types of data transfer can be implemented: ***status check and interrupt.***
  3. Mode 2 (Bidirectional I/O interfacing):
     - 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.
- The BSR mode is used to set or reset the bits in port C.

8255 Control Word

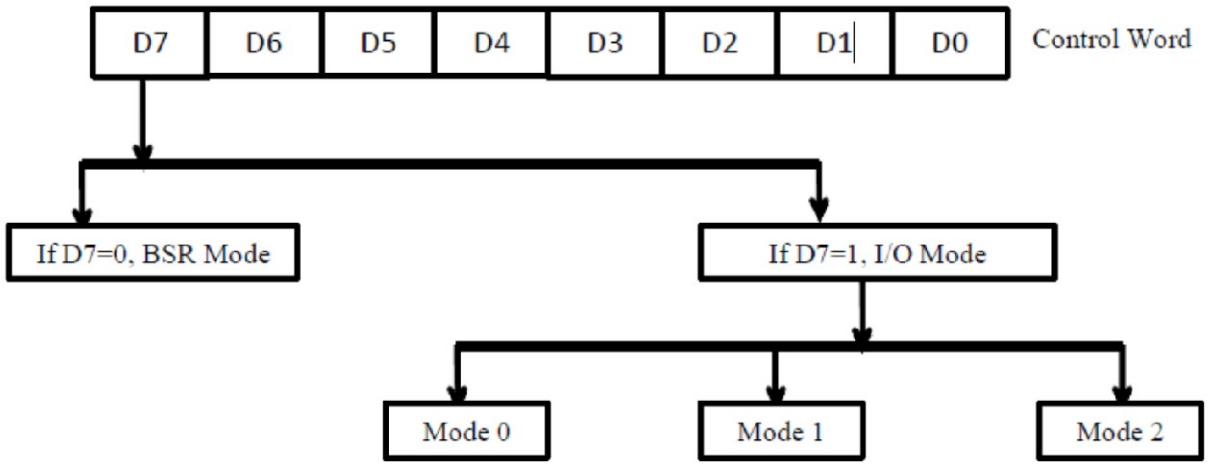


Fig: Control word specifying various modes

Figure shows all the functions of 8255A, classifying according to two modes: the Bit Set -Reset (BSR) mode and Input Output (I/O) 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 A0 and A1 are at logic 1. The register is not accessible for read operation.

- Bit D7 of the control register specifies either I/O functions or Bit Set-Reset function as classified in figure.

- 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

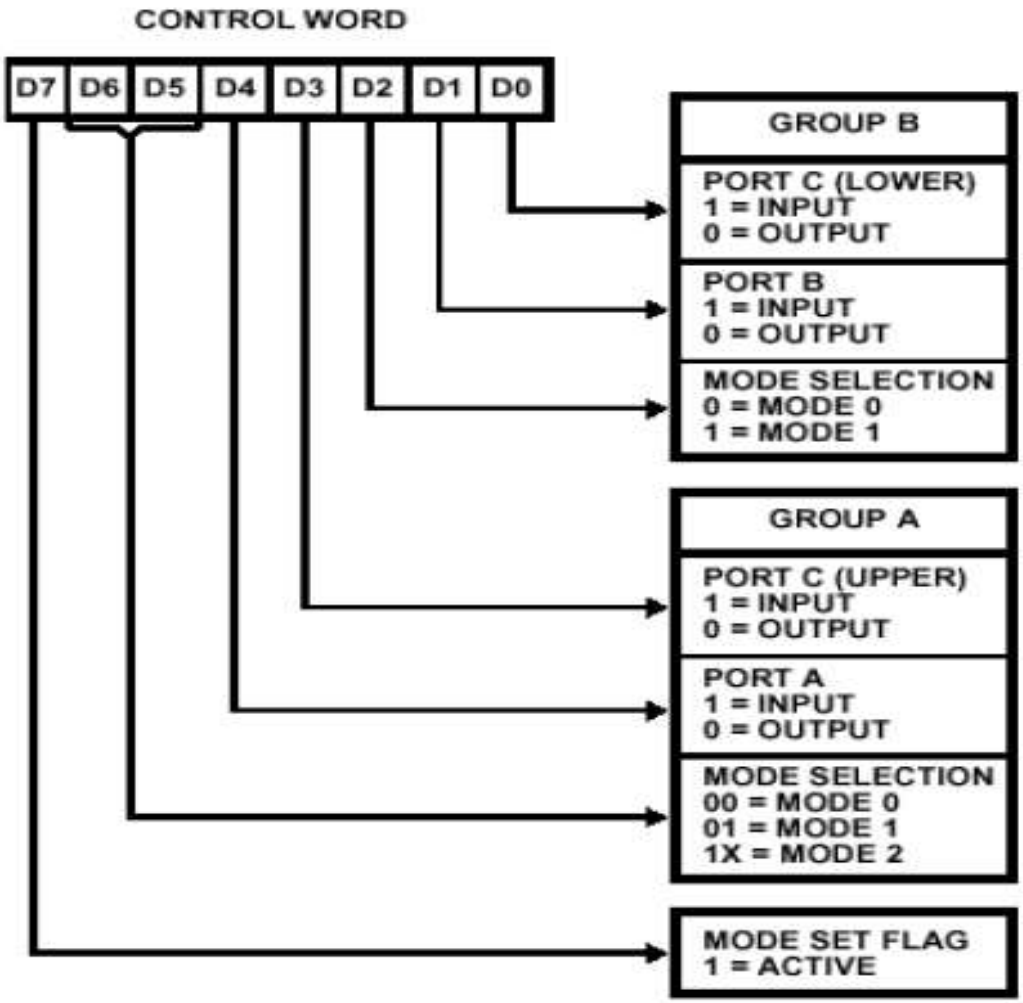


Fig: I/O Mode Definition Control Word Format

CONTROL WORD IN BSR MODE

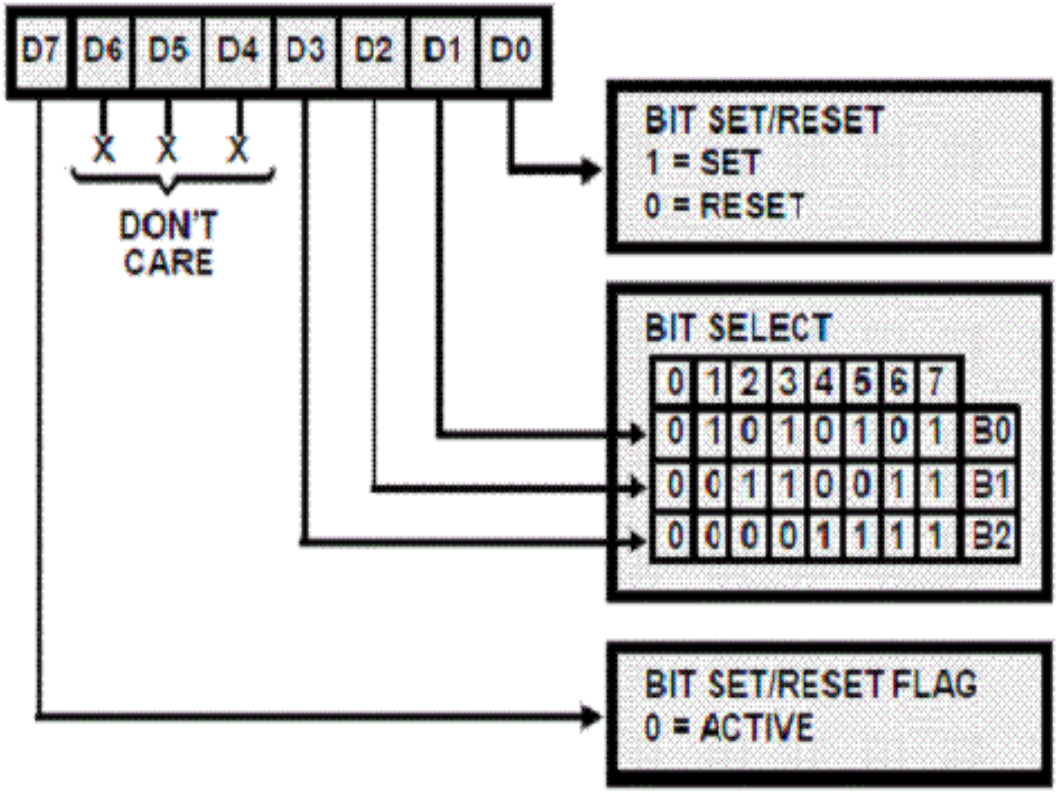


Fig: BSR Mode

RS-232

- 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.
- Device used to send serial data signals over long distances.
- RS232 is the most widely used serial I/O interfacing standard.
- RS-232C is an interface developed to standardize the interface between data terminal equipment (DTE) and data communication equipment (DCE) employing serial binary data exchange.
- Modem and other devices used to send serial data are called data communication equipment (DCE).
- The computers or terminals that are sending or receiving the data are called data terminal.

- 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.
- It describes the voltage levels, impendence levels, rise and fall times, maximum bit rate and maximum capacitance for all signal lines.
- It also specifies that DTE connector should be male and DCE connector should be female.
- It can send 20kBd for a distance of 50 ft.

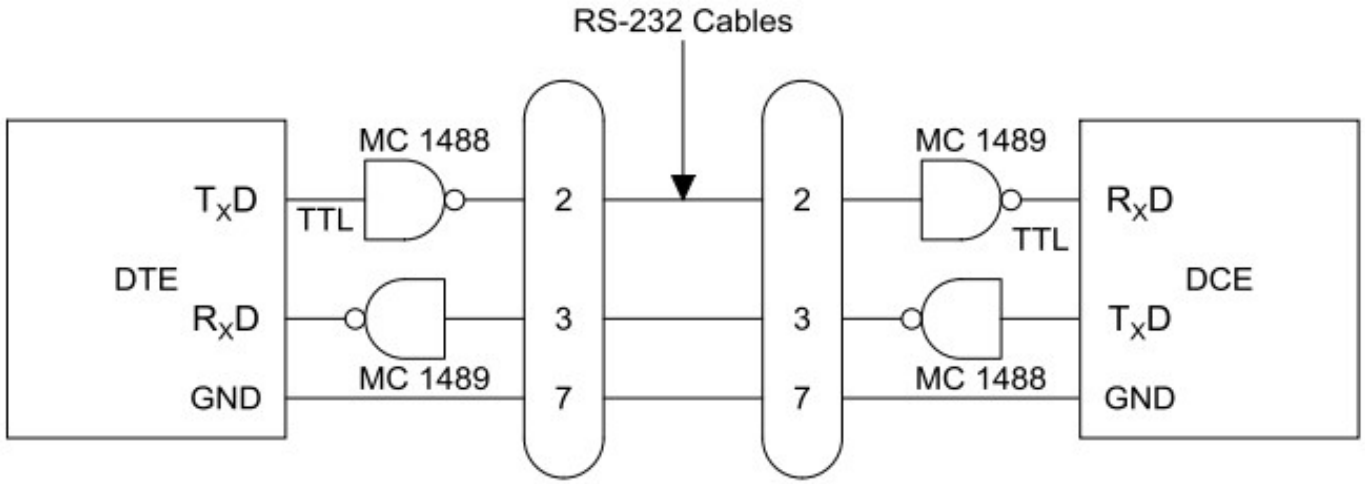


Fig: Connection of DTE and DCE through RS-232C Interface

- The I/O voltage levels are not TTL compatible.
- **1 (HIGH)** is represented by _**–3 to –25V**, while **0 (LOW)** by **+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.

- Mc1488 converts:
  - logic 1 to -9V
  - Logic 0 to +9v
- Mc1485 converts RS – 232 to TTL

25 Pin Connector on a DTE device (PC connection)

Male RS232 DB25

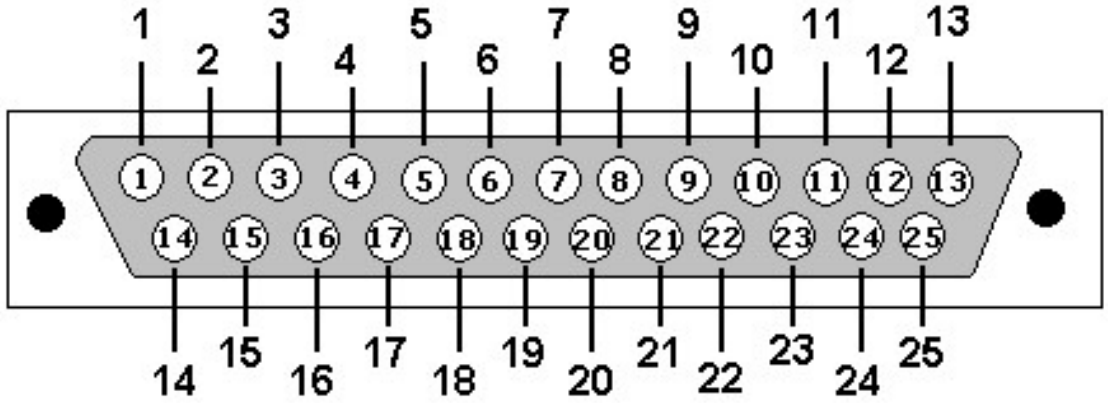


Fig: Male RS232 DB25

RS232 on DB25 (RS-232C)

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

Connector on a DTE device (PC connection)

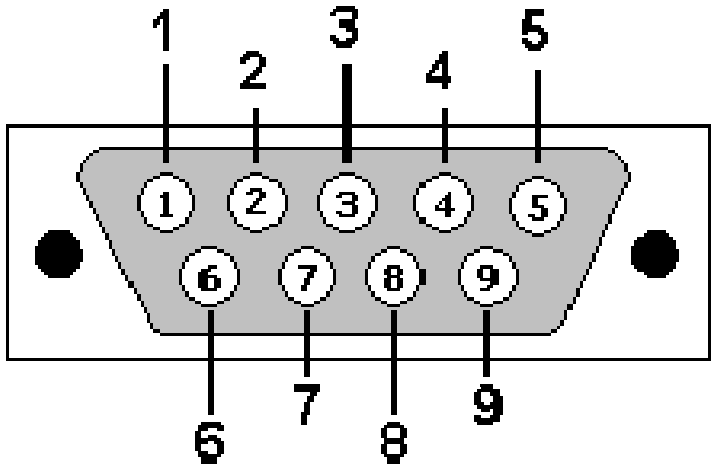


Fig: Male RS232 DB9

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

- The minimum interface between a computer and a peripheral requires 3 lines: pins 2, 3 and 7 as shown in figure.

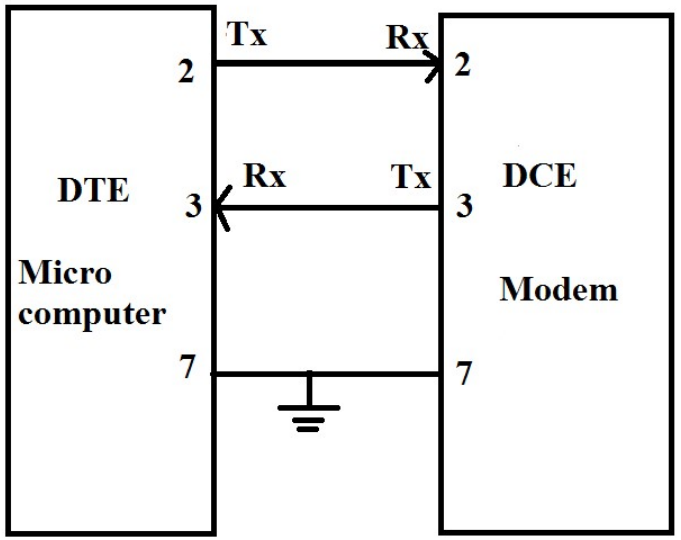


Fig (a):-DTE to DCE

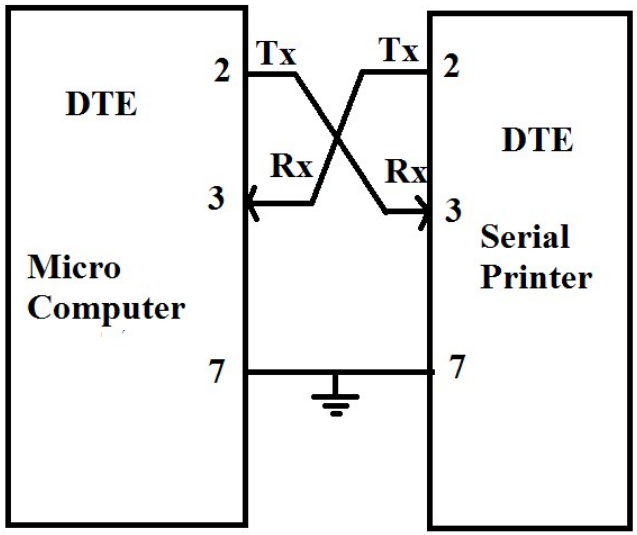


Fig (b):-DTE to DTE (null modem)

- 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), also known as ***Null modem connection.***