Microprocessor

Unit 5:
Basic I/O, Memory R/W and Interrupt Operations

Course Outline

- Parallel Communication – Introduction and Applications
- Serial Communication
  - Introduction and Applications
  - Introduction to Programmable Communication Interface 8251
  - Basic Concept of Synchronous and Asynchronous Modes
- Simple I/O, Strobe I/O, Single handshake I/O, Double handshake I/O
- 8255A and it’s Working
  - Block Diagram
  - Modes of Operation
  - Control Word
- RS-232 – Introduction, Pin Configuration (9 pin and 25 pin) and function of each pin, Interconnection between DTE-DTE and DTE-DCE

Input/Output interface

- User communicates with the computer via standard peripheral devices such as keyboard, mouse, display, printer, etc.
- The I/O interface is required to enable the interface between the microprocessor and the peripheral devices.
- The peripheral devices are connected on a microprocessor system through I/O ports.
- The I/O interface provides the following:
  - Isolation between the buses and the peripheral devices.
  - Address decoding
  - Synchronization/Latching

Input/Output interface

- Peripherals connected to a computer need special communication links for interfacing them with the MPU.
- The purpose of communication link is to resolve the differences that exist between the central computer and each peripheral.
- The Major Differences are:-
  - Peripherals are electromechnical and electromagnetic devices and CPU and memory are electronic devices. Therefore, a conversion of signal values may be needed.
  - The data transfer rate of peripherals is usually slower than the transfer rate of CPU and consequently, a synchronization mechanism may be needed.
  - Data codes and formats in the peripherals differ from the word format in the CPU and memory.
  - The operating modes of peripherals are different from each other and must be controlled so as not to disturb the operation of other peripherals connected to the CPU.

Input/Output interface

- To Resolve these differences, computer systems include special hardware components between the MPU and Peripherals to supervises and synchronizes all input and out transfers
- These components are called **Interface Units** because they interface between the processor bus and the peripheral devices.

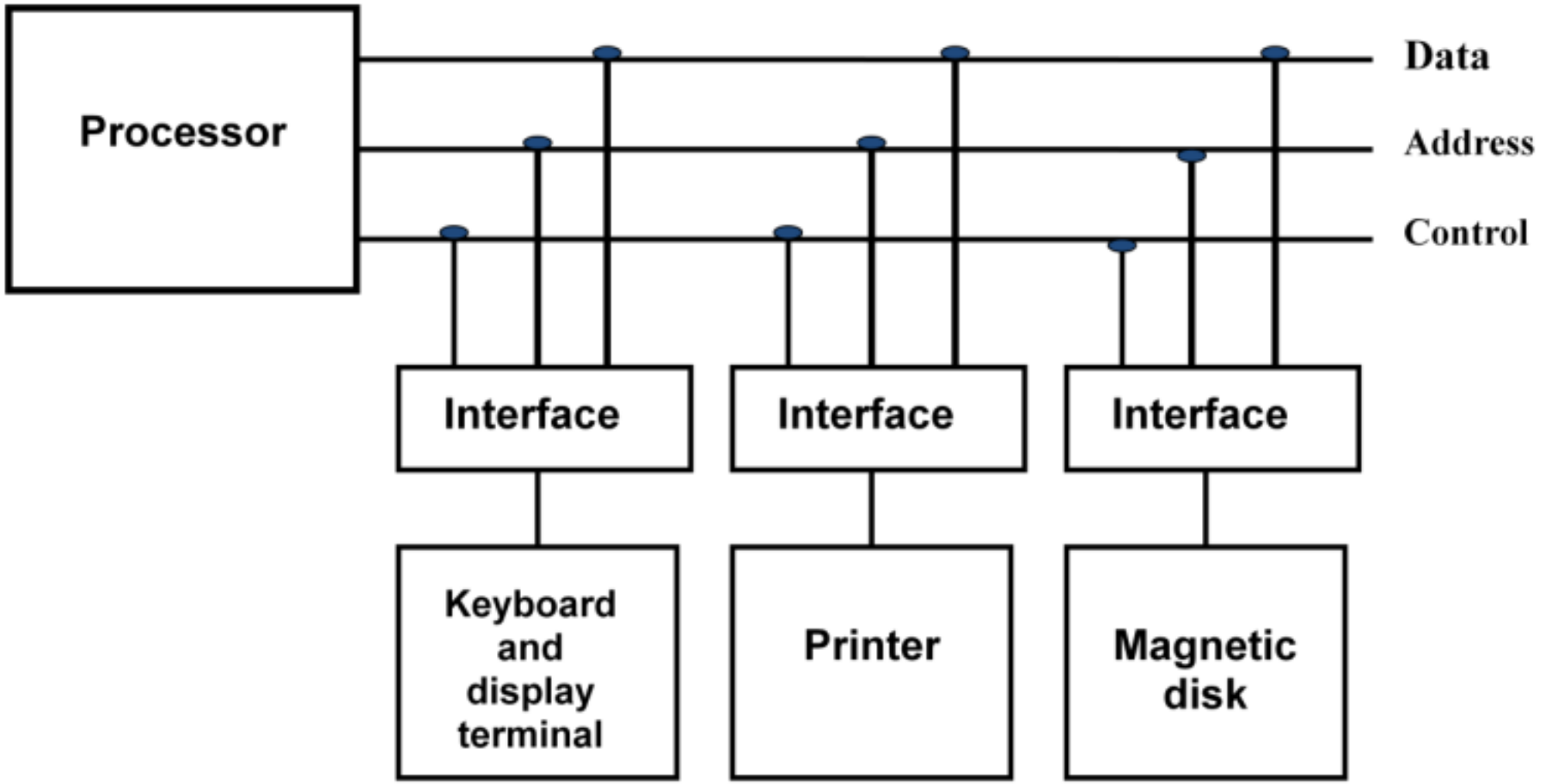
I/O BUS and Interface Module

- It defines the typical link between the processor and several peripherals.
- The I/O Bus consists of data lines, address lines and control lines.
- The I/O bus from the processor is attached to all peripherals interface.
- To communicate with a particular device, the processor places a device address on address lines.
- Each Interface decodes the address and control received from the I/O bus, interprets them for peripherals and provides signals for the peripheral controller.
- It is also synchronizes the data flow and supervises the transfer between peripheral and processor.
- Each peripheral has its own controller.

I/O command

- The control lines are referred as I/O command.
- Control command
  - A control command is issued to activate the peripheral and to inform it what to do.
- Status command
  - A status command is used to test various status conditions in the interface and the peripheral.
- Data Output command
  - A data output command causes the interface to respond by transferring data from the bus into one of its registers.
- Data Input command
  - The data input command is the opposite of the data output

Connection of bus to I/O devices



Function of Input/Output interface

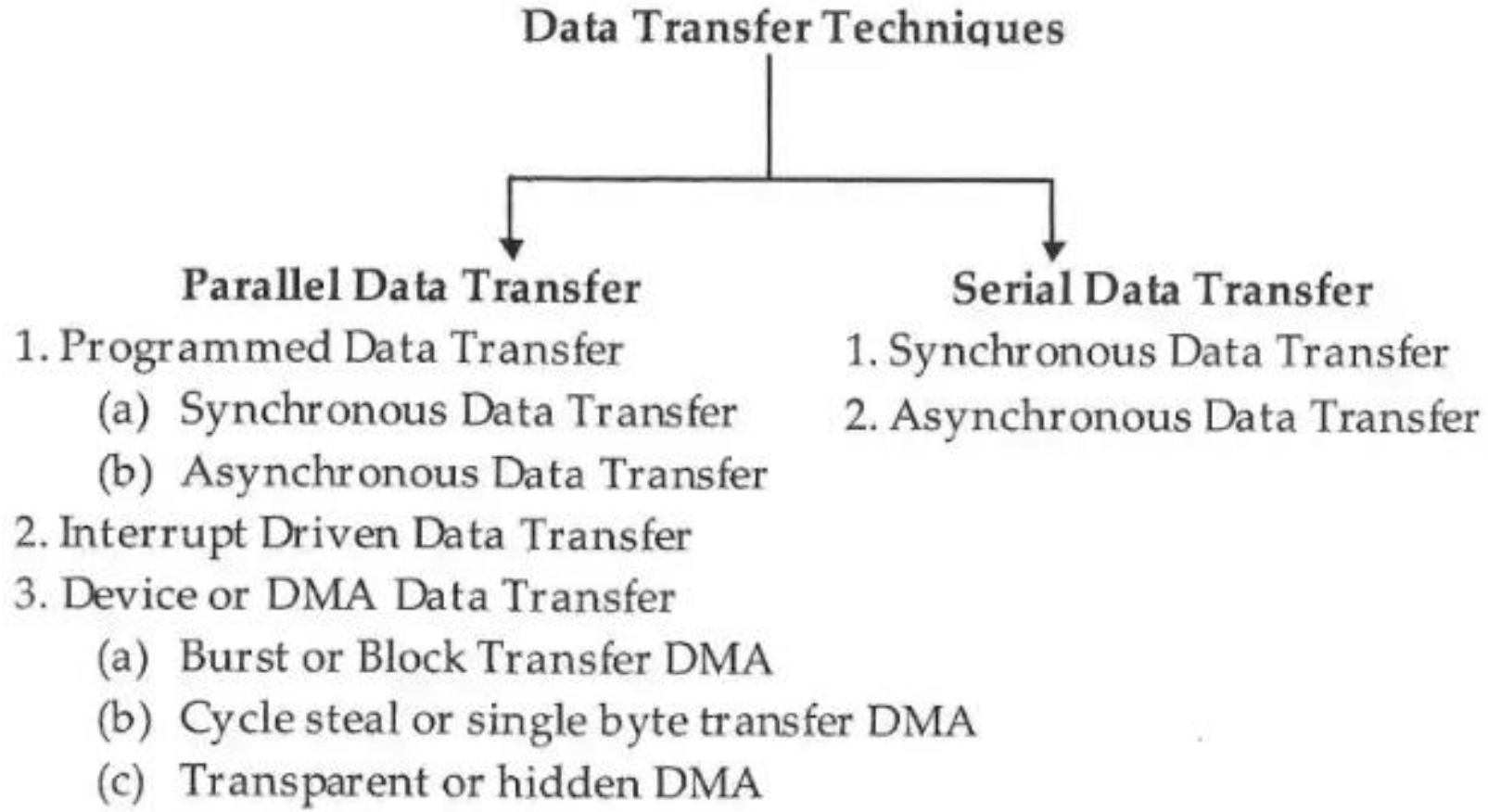
- It is used to synchronize the operating speed of CPU with respect to input-output devices.
- It selects the input-output device which is appropriate for the interpretation of the input-output device.
- It is capable of providing signals like control and timing signals.
- In this data buffering can be possible through data bus.
- There are various error detectors.
- It converts serial data into parallel data and vice-versa.
- It also convert digital data into analog signal and vice-versa.

Isolated Vs Memory-mapped I/O

- Isolated I/O
  - I/O locations are separate from memory locations
  - Special I/O instructions are used
  - The most common technique for Intel microprocessors
  - Advantage: More space for memory
  - Disadvantage: Additional control signal (IO/$\overline{M}$) & instructions increase complexity
- Memory-mapped I/O
  - I/O devices are treated as memory locations in the memory map
  - Any memory transfer instruction can be used (MOV, LDA, STA, etc)
  - Advantage: Simpler decoding circuitry, no special instructions required
  - Disadvantage: A portion of the memory system is used as the I/O map, reducing the memory available to applications

Parallel Vs Serial Communication

Data Transfer Technique



Serial Communication

- In serial communication the data bits are transmitted serially over a common communication link one after the other.
- Basically it does not allow simultaneous transmission of data because only a single channel is utilized.
- Thereby allowing sequential transfer rather than simultaneous transfer.
- It is highly suitable for long distance signal transmission as only a single wire or bus is used.
- So, it can be connected between two points that are separated at a large distance with respect to each other.
- But as only a **single data bit is transmitted per clock pulse** thus the transmission of data is a quiet time taking process.

Parallel Communication

- In parallel communication the various data bits are simultaneously transmitted using multiple communication links between sender and receiver.
- Here, despite using a single channel between sender and receiver, various links are used and each bit of data is transmitted separately over all the communication link.
- Here, as we can see that for the transmission of 8-bit of data, 8 separate communication links are utilized.
- And so rather following a sequential data transmission, simultaneous transmission of data is allowed. This leads to a faster communication between the sender and receiver.
- But for connecting multiple lines between sender and receiver multiple connecting unit are to be present between a pair of sender and receiver. And this is the reason why parallel communication is not suitable for long distance transmission, because connecting multiple lines to large distances is very difficult and expensive.

Serial Vs Parallel Transmission

S.NO	Serial Transmission	Parallel Transmission
1.	In serial transmission, data(bit) flows in bi-direction.	In Parallel Transmission, data flows in multiple lines.
2.	Serial Transmission is cost-efficient.	Parallel Transmission is not cost-efficient.
3.	In serial transmission, one bit transferred at one clock pulse.	In Parallel Transmission, eight bits transferred at one clock pulse.
4.	Serial Transmission is slow in comparison of Parallel Transmission.	Parallel Transmission is fast in comparison of Serial Transmission.
5.	Generally, Serial Transmission is used for long-distance.	Generally, Parallel Transmission is used for short distance.
6.	The circuit used in Serial Transmission is simple.	The circuit used in Parallel Transmission is relatively complex.

Synchronous and Asynchronous Modes

Synchronous Mode

- This communication methods is mostly used when large amounts of data needs to be transferred from one location to the other.

Synchronous Mode

- Advantages
  - Lower overhead
  - Greater throughput compared to asynchronous mode
- Disadvantages
  - Slightly more complex
  - Hardware is more expensive

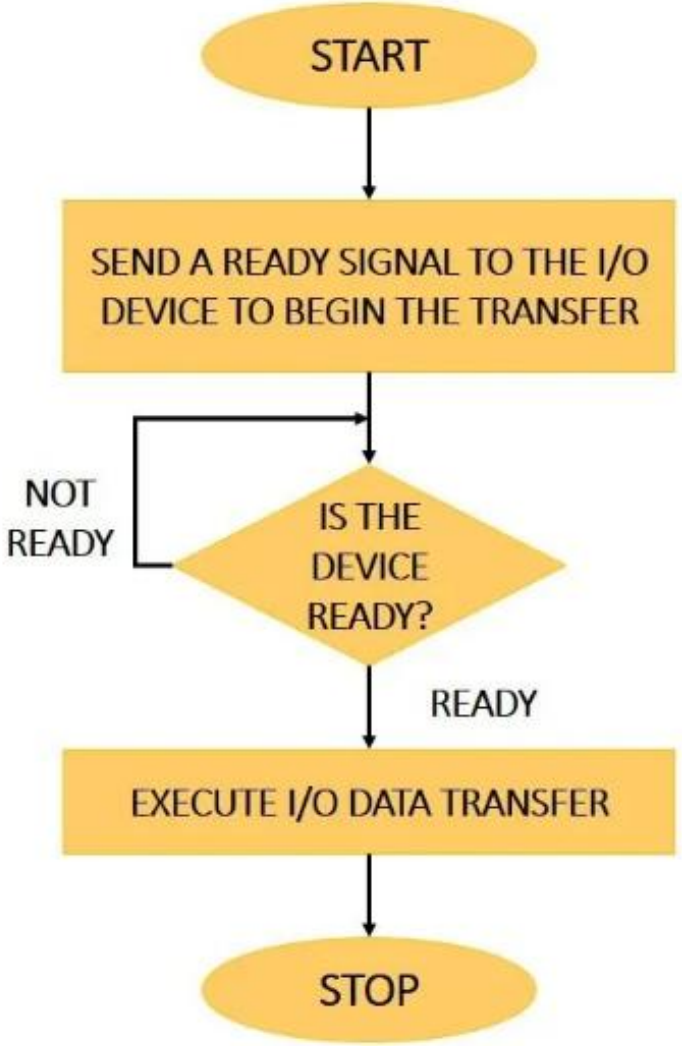
Asynchronous Mode

- It does not use a clock to synchronize data between the source and destination.

Asynchronous Mode

- Advantages
  - Simple and doesnot require much synchronization on both communication sides
  - Timing is not critical as in synchronous mode
  - Hardware can be made cheaper
- Disadvantages
  - Large relative overhead
  - High proportion of the transmitted bits are uniquely for control purposes and thus carry no useful information.

Handshaking Protocol



Handshaking Protocol

- Under the handshaking method, the microprocessor and I/O device exchange a few signals before beginning the transfer of data between them.
- Let us understand the flowchart of the handshake protocol.
  - First, the microprocessor raises a ready signal and sends it to the I/O device, which is connected to it.
  - The status of the I/O device is continually checked to see it is prepared for data transferring.
  - If the device is not ready, it is checked again and again until the device signals it is ready.
  - Once the device is ready, the data transfer begins from the microprocessor to the I/O device.
- But how does the I/O device inform the microprocessor that it is ready for data transfer?
- This is done by the ‘ACK’ (**Acknowledge**) Signal. Also known as the handshake signal.

How synchronous Mode works?

- Separate clocking lines used when the distance between the data terminal equipment (DTE) and data communications equipment (DCE) is short.
- This method uses a clocking electrical system at both transmitting and receiving stations. This ensures that the communication process is synchronized.
- Devices that communicate with each other Synchronously use either separate clocking channels.

How asynchronous Mode works?

- Asynchronous communication is eased by two bits, which is known as start bit as ‘0’ and stop bit as ‘1.’
- You need to send ‘0’ bit to start the communication & ‘1’ bit to stop the Transmission.
- There is a time delay between the communication of two bytes.
- The transmitter and receiver may be function at different clock frequencies.

Synchronous Vs Asynchronous

- Synchronous data transfer
  - Sender and receiver use the same clock signal
  - Supports high data transfer rate
  - Needs clock signal between the sender and the receiver
  - A master (or one of the senders) should provide the clock signal to all the receivers in the synchronous data transfer.
- Asynchronous data transfer
  - For asynchronous data transfer, there is no common clock signal between the sender and receivers.
  - Sender provides a synchronization signal to the receiver before starting the transfer of each message.
  - The sender and the receiver first need to agree on a data transfer speed.
  - Slower data transfer rate.

Methods of Parallel Data Transfer

Simple I/O, Strobe I/O, Single handshake I/O, Double handshake I/O

Simple I/0

- See yourseld

Strobe I/0

- See yourself

Single Handshaking

- In single handshake, a peripheral device first sends a "Strobe signal" to the microprocessor to indicate that it is ready to send data.
- The microprocessor , upon detecting the strobe signal, opens up its input port and receives the data.
- After receiving data, it sends an "Acknowledge signal" to the peripheral to indicate that transmission has been completed.
- A transmission session has been completed.

Double Handshaking

- In double handshake, first the peripheral device sends a strobe signal, the microprocessor, sends the acknowledge signal to indicate that it is ready to receive data.
- After which data is received.
- After sending data, the peripheral sends a strobe signal to indicate data transmission completion, due to which, the microprocessor drops its acknowledge signal and a session has been completed.

Single Vs Double Handshaking

- The only difference in the two is that,
  - in double handshake, the peripheral is informed about the microprocessor's readiness to receive data.
  - This is doesn't happen in single handshake.
  - So the name follows "double handshake", literally meaning "double confirmation".

Serial Data Transmission modes

Serial Data Transmission modes

- Simplex
- Half Duplex
- Full Duplex

USART

USART

- Universal Synchronous Asynchronous Receiver Transmitter
- The USART module is a full duplex, serial I/O communication peripheral.
- It is packed in a 28 pin DIP.
- It contains all shift registers, clock generators and data buffers needed for serial communication.
- It can work in synchronous mode or in asynchronous mode.
- The USART uses two I/O pins to transmit and receive serial data.
- Both transmission and reception can occur at the same time i.e. full duplex operation.

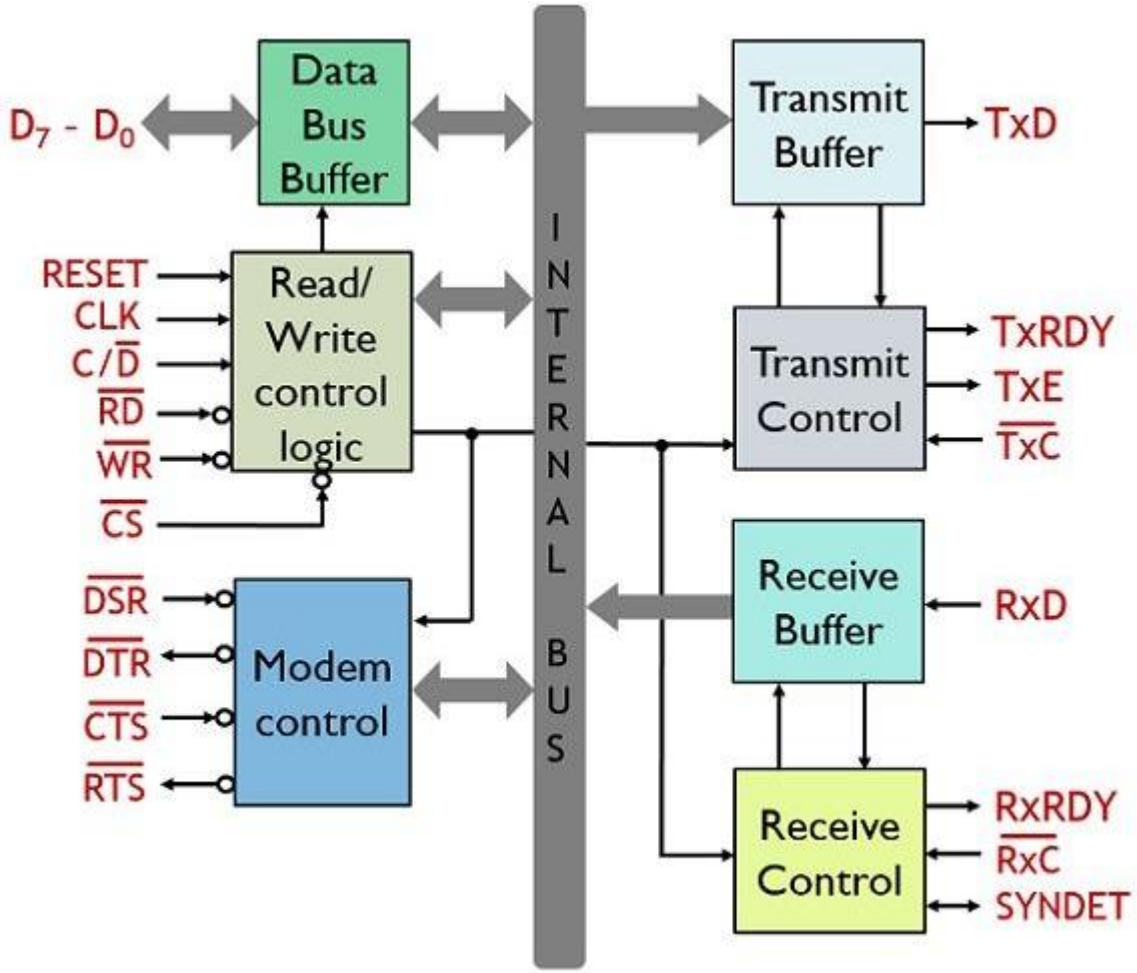
USART

- To send a byte, the application writes the byte to the transmit buffer.
  - The USART then sends the data, bit by bit in the requested format, adding STOP, Start, and parity bits as needed.
- In a similar way, the USART stores received bytes in the receiver buffer
  - Then the USART can generate an interrupt to notify the processor to find out if data has arrived.

8251 USART

- 8251 is a USART for serial data communication.
- Programmable peripheral designed for synchronous/asynchronous serial data communication, packaged in 28-pin DIP
- Receives parallel data from the processor and transmit serial data after conversion.
- Also receives serial data from the outside and transmit parallel data to the CPU after conversion.
- To reduce the overall cost of the system despite parallel data communication between the processor and peripheral devices, the serial transfer of data is permitted.
- Hence for this purpose, USART acts as a mediator between the processor and peripheral devices so, that the parallel data from the processor can be converted into serial data and efficiently transferred to the peripheral devices.
- And viceverca

Architecture of 8251 USART



Functional Blocks of 8251

- Data bus buffer
- Read/Write control logic
- Modem control
- Transmit buffer
- Transmit control
- Receive Buffer
- Receive control

Data Bus Buffer

- It basically interfaces the 8251 with the internal system buses of the processor.
- The data bus buffer has **8-bit** bidirectional data bus that allows the transfer of data bytes, status or command word between the processor and external devices.
- The data transmission is possible between 8251 and CPU by the data bus buffer block.

Read/Write Control Logic

- This functional unit generates a control signal for the operation of 8251 according to the signal present in the control bus of the processor.
- It is a control block for overall device.
- It controls the overall working by selecting the operation to be done.
- Basically, it performs decoding operation of the control signal produced by the processor, so that respective operation can be performed by the USART.
- The control formats for system operation is stored in control and command word registers present in the read/write logic unit.

Read/Write Control Logic (contd…)

- Signals handled by the read/write control logic
  - ***CS***: It is chip select. A low signal at this pin shows that processor has selected 8251 in order to communicate with the peripheral devices.
  - ***C/D***: As the system has control, status and data register. So, when a high signal is present at this pin then control or status register is addressed. While in case of low signal data register is addressed.
  - ***RD*** and ***WR***: Both read and write are active low signal pins. A low signal at RD shows that the processor is reading the control, status or data bytes from the 8251. While at WR indicates the write operation over the data bus of 8251.
  - ***CLK*** and ***RESET***: CLK stands for clock and it produces the internal timing for the device. While an active high signal at the RESET pin puts the 8251 in the idle mode.

Read/Write Control Logic (contd…)

- Signals handled by the read/write control logic

$\overline{CS}$	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 unit is used to change the parallel data received from the CPU into serial data by inserting the necessary framing information.
- Once the data is transformed into serial form, then in order to transmit it to the external devices, it is provided to the TxD pin of the 8251.
- This unit consists of 2 registers. These are as follows:
  - ***Buffer register***:
  - ***Output register***:

Transmit Buffer (contd…)

- ***Buffer register***:
  - Basically the data provided by the processor is stored in the buffer register.
  - As we know that initially, the CPU provides parallel data to 8251.
  - So, the processor loads the parallel data to the buffer register.
  - Further, this data is fed to the output register.
- ***Output register***:
  - The parallel data from the buffer register is fed to the empty output register.
  - This register changes the 8-bit parallel data into a stream of serial bits.
  - Then further the serial data is provided at the TxD pin so as to have its transfer to the peripheral device.
  - **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.
- And it does so by accepting and sending signals both externally and internally.
- The various control signal generated by this unit are as given below:
  - **TXRDY:** It means transmitter is ready to transmit data character.
  - **TXEMPTY:** An output signal which indicates that TXEMPTY pin has transmitted all the data characters and transmitter is empty now.
  - **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.
  - This unit takes the serial data from the external devices, changes the serial data into the parallel form so that it can be accepted by the processor.
- It consists of 2 registers: receiver input register and buffer register.

Receive Control

- This block controls the receiving data.
- It manages the data reception, along with that it also detects the presence of false start bit, error in parity bit, framing errors etc.
- ***RxRDY***: It stands for receiver ready. When this signal goes high then it indicates that the receiver buffer register is holding the data and is ready to transfer it to the processor. Once the CPU reads the data sent by the 8251 then this pin is reset.
- ***RxC***: It stands for receiver clock. This clock signalling controls the rate at which the 8251 receives the data in the synchronous mode of operation. It is provided by the modem and is equal to the baud rate. While asynchronous mode offers the clock rate as 1, 16 or 64 times of the baud rate as it is programmable.
- **SYNDET/BD:** An input or output terminal. External synchronous mode-input terminal and asynchronous mode-output terminal.

Modem Control

- A device converts analog signals to digital signals and vice-versa and helps the computers to communicate over telephone lines or cable wires.
- The control circuitry for handing various signals is provided by the modem control unit. It includes DTS, RTS, DTR and CTS.
  - **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

8255

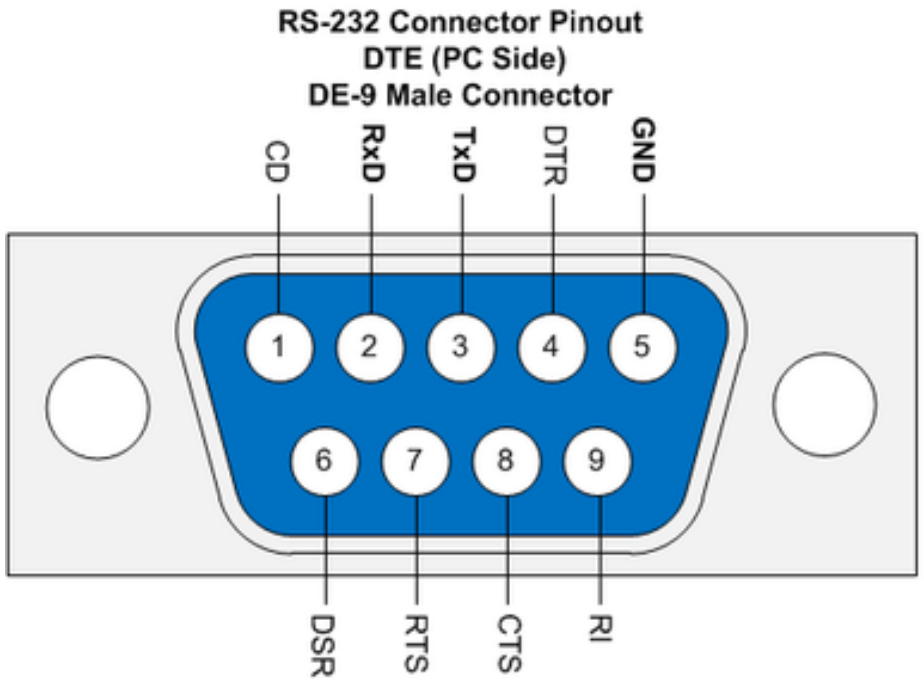
See Yourself

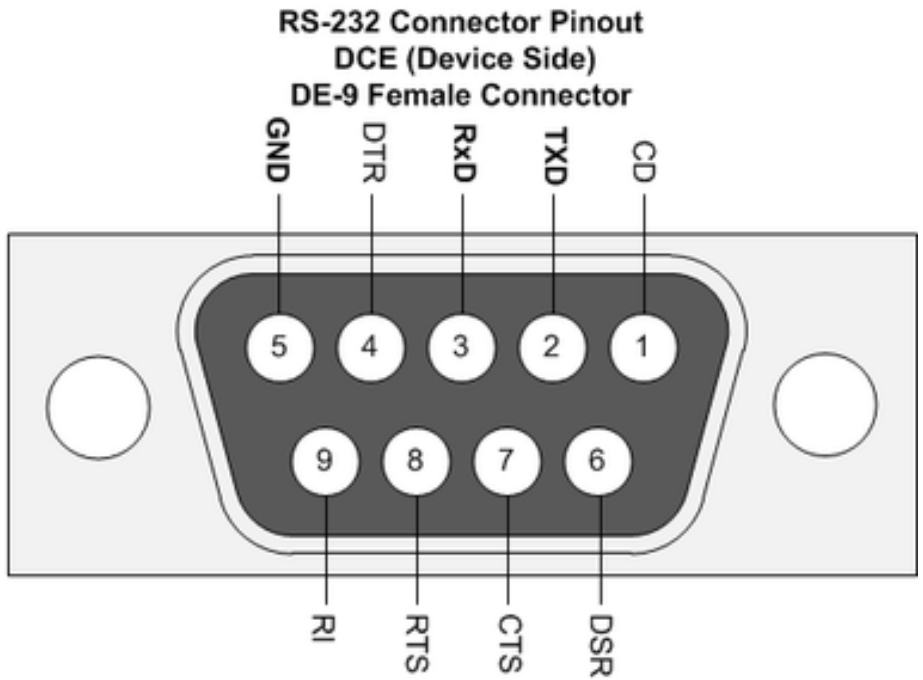
RS-232

RS-232

- The RS-232(X) is a serial communication protocol, commonly used for transferring and receiving the serial data between two devices.
- It is basically an interface standard.
- The logic 1 is represented by the -12V and logic 0 is represented the +12V.
- It supports both synchronous and asynchronous data transmissions.
- The RS-232 cable works at different baud rates like 9600 bits/s, 2400bits/s, 4800bits/s etc.
- The RS-232 cable has two-terminal devices namely Data Terminal Equipment and Data communication Equipment.
- Both devices will send and receives signals. The data terminal equipment is a computer terminal and data communication Equipment is modems, or controllers, etc.

RS-232 DB-9: PIN Diagram





RS-232 9-pin: PIN Diagram

PIN No.	Pin Name	Pin Description
1	CD (Carrier Detect)	Incoming signal from DCE
2	RD (Receive Data)	Receives incoming data from DTE
3	TD (Transmit Data)	Send outgoing data to DCE
4	DTR (Data Terminal Ready)	Outgoing handshaking signal
5	GND (Signal ground)	Common reference voltage
6	DSR (Data Set Ready)	Incoming handshaking signal
7	RTS (Request to Send)	Outgoing signal for controlling flow
8	CTS (Clear to Send)	Incoming signal for controlling flow
9	RI (Ring Indicator)	Incoming signal from DCE

Handshaking

- Before the actual data transfer, signals are transmitted from DTE to DCE in order to make connections by a process known as handshaking. Following is the sequence of signal handshaking:
- Initially, the computer activates RTS signal to modem when a data is transferred from computer to modem.
- Modem in turn activates the DCD and then the CTS gets activated.
- Computer then sends data on TXD. After the data transmission is completed, the computer deactivates the RTS which causes the modem to deactivate CTS.

Handshaking: Types

- **No Handshaking**:
- If there is no handshaking, then DCE reads the already received data while DTE transmits the next data.
- All the received data stored in a memory location known as receiver's buffer.
- This buffer can only store one bit so receiver must read the memory buffer before the next bit arrives.
- If the receiver is not able to read the stored bit in the buffer and next bit arrives then the stored bit will be lost.
- As shown in below diagram, a receiver was unable to read the 4^{th} bit till the 5^{th} bit arrival and this result overriding of 4^{th} bit by 5^{th} bit and 4^{th} bit is lost.

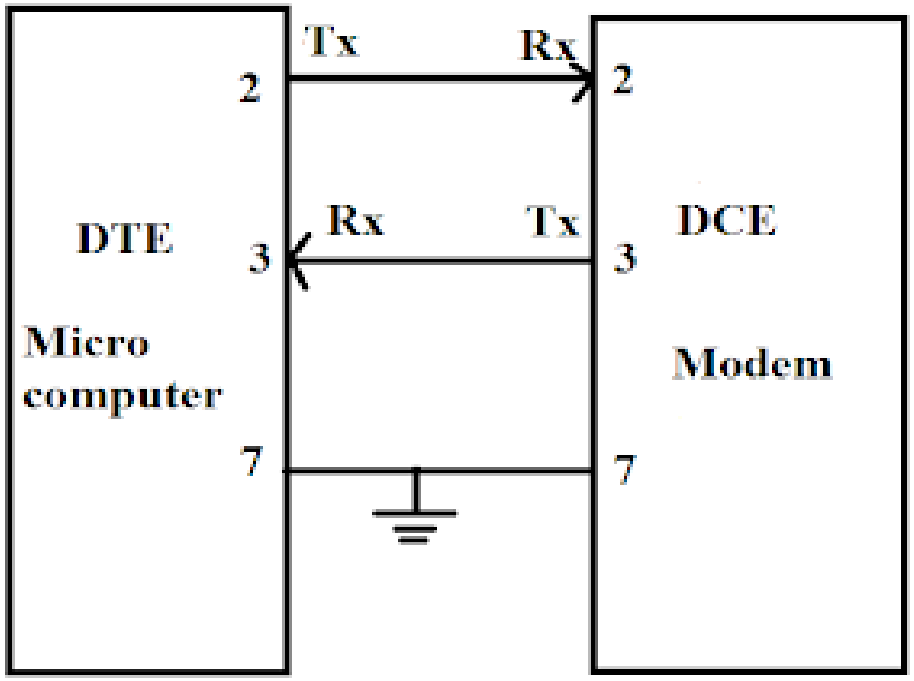
Handshaking: Types

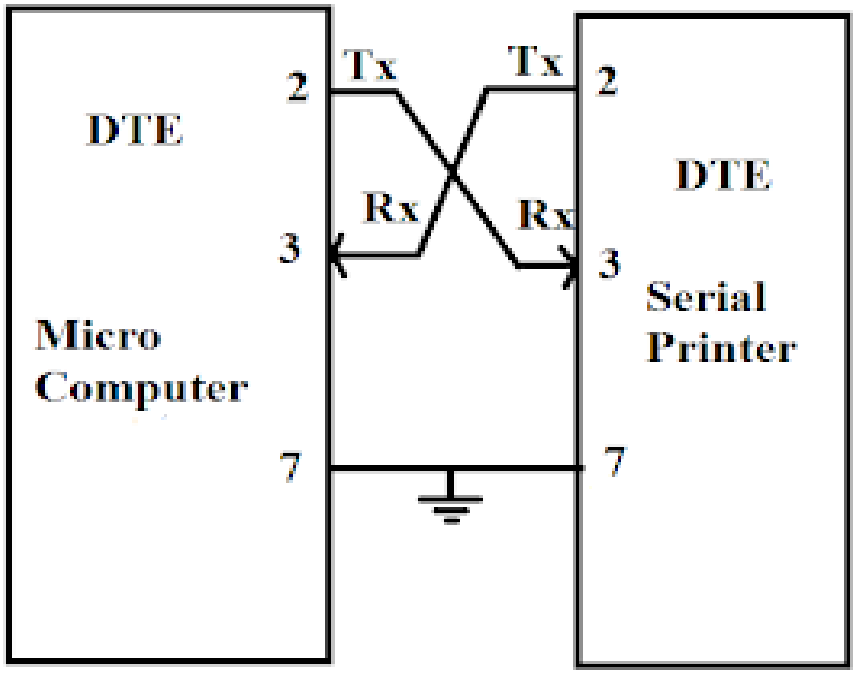
- **Hardware Handshaking**:
- It uses specific serial ports, i.e., RTS & CTS to control data flow.
- In this process, transmitter asks the receiver that it is ready to receive data then receiver checks the buffer that it is empty, if it is empty then it will give signal to the transmitter that I am ready to receive data.
- The receiver gives the signal to transmitter not to send any data while already received data cannot be read.
- Its working process is same as above described in handshaking.

Handshaking: Types

- **Software Handshaking**:
- In this process, there are two forms, i.e., X-ON & X-OFF. Here, ‘X’ is the transmitter.
- X-ON is the part in which it resumes the data transmission.
- X-OFF is the part in which it pauses the data transmission.
- It is used to control the data flow and prevent loss during transmission.

Interconnection between DTE-DTE and DTE-DCE

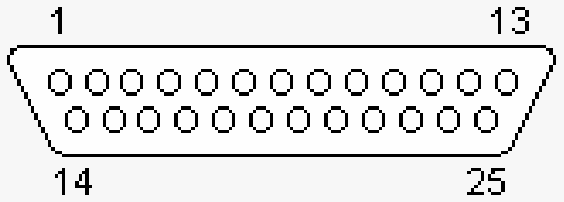




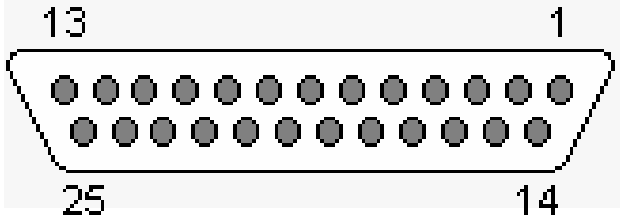
Interconnection between DTE-DTE and DTE-DCE

- The minimum interface between a computer and a peripheral requires 3 lines: pins 2, 3 and 7 as shown in figure.
- 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 DTE to DCE connection, 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 DTE to DTE. This is also known as Null modem connection.

RS-232 (DB-25 PIN)



DB25 pin D-SUB male at the DTE (Computer)



DB25 pin D-SUB female at the DCE (Modem)

RS-232 DB25 PIN

Pin	Name	Direction	Description
1	GND		Shield Ground
2	TXD	—»	Transmit Data
3	RXD	«—	Receive Data
4	RTS	—»	Request to Send
5	CTS	«—	Clear to Send
6	DSR	«—	Data Set Ready
7	GND		System Ground
8	CD	«—	Carrier Detect
9	-	-	RESERVED
10	-	-	RESERVED
11	STF	—»	Select Transmit Channel
12	S.CD	«—	Secondary Carrier Detect
13	S.CTS	«—	Secondary Clear to Send

Pin	Name	Direction	Description
14	S.TXD	—»	Secondary Transmit Data
15	TCK	«—	Transmission Signal Element Timing
16	S.RXD	«—	Secondary Receive Data
17	RCK	«—	Receiver Signal Element Timing
18	LL	—»	Local Loop Control
19	S.RTS	—»	Secondary Request to Send
20	DTR	—»	Data Terminal Ready
21	RL	—»	Remote Loop Control
22	RI	«—	Ring Indicator
23	DSR	—»	Data Signal Rate Selector
24	XCK	—»	Transmit Signal Element Timing
25	TI	«—	Test Indicator

Advantages

- The **advantages of RS232** make it as a standard serial interface for system to system communication and also for the following benefits.
- Simple protocol design.
- Hardware overhead is lesser than parallel communication.
- Recommended standard for short distance applications.
- Compatible with DTE and DCE communication.
- Low cost protocol for development.

Limitation of RS-232

- In order to operate RS232 needs a common platform between the transmitter and the receiver.
  - That is why short cables are used between DTE and DCE in RS232 Protocol.
- If Baud Rate increases along with the length of the cable, there is high possibility of cross talk being held by the capacitance between the cables.
- The signal in the line is extremely receptive to noise, which can be internal as well as external.
- The voltage levels of RS232 are not adaptable with the modern system of TTL. For this, an external level converter is required.

Application of RS-232

- RS232 communication is used in different applications. Some of them are:
  - Telephones lines
  - Teletypewriter devices.
  - Demodulator applications.
  - PC COM port interfacing.
  - In embedded system for debugging.
  - Modems and printers.
  - Handheld equipment.
  - CNC controllers, Software debuggers etc.
  - Barcode scanners and Point of Sales (POS) terminals.