EIE 418 LECTURE NOTES

INTERFACING: INTERFACES FOR SIMPLE COMPUTER SYSTEM AND TERMINAL TO TERMINAL.

What is an interface?

An interface is a device and/or set of rules to match the output of one device to send information to the input of another device. An interface usually requires: (i) a physical connection, (ii) the hardware (iii) rules and procedures and last (iv) the software.
Interfacing is the process of connecting devices together so that they can exchange information.

Why do we need an Interface?
The primary function of an interface is obviously to provide a communication path for data and commands between the computer and its resources. Interfaces acts as intermediaries between resources by handling part of the “bookkeeping” work and ensuring that the communication process flows smoothly.
Agreement regarding the signal type, how they should be converted and transmitted is not enough. Agreement is also required regarding the type of connector and the voltage levels they need to support. In other words, the physical and electrical interfaces are important. There is also a logical interface, which defines the significance of the signal. A protocol controls how the signals are built up, how communications are initiated, how they are terminated, the order of transmitting and sending, how to acknowledge a message, etc.
The physical interface defines how equipment is connected as well as the design of the connector. The electrical interface defines the electrical levels and what these denote (ones or zeros). The Logical interface defines what the signals signify.
In summary the following are reasons why we need an interface:

First Reason

First, even though the computer backplane is driven by electronic hardware that generates and receives electrical signals, this hardware was not designed to be connected directly to external devices. The electronic backplane hardware has been designed with specific electrical logic levels and drive capability in mind.
Exceeding the backplane hardware ratings will damage the electronic hardware.

Second Reason

Second, you cannot be assured that the connectors of the computer and peripheral are compatible. In fact, there is a good probability that the connectors may not even mate properly, let alone that there is a one-to-one correspondence between each signal wire's function.

Third Reason

Third, assuming that the connectors and signals are compatible, you have no guarantee that the data sent will be interpreted properly by the receiving device. Some peripherals expect single-bit serial data while others expect data to be in 8-bit parallel form.

Fourth Reason

There is no reason to believe that the computer and peripheral will be in agreement as to when the data transfer will occur; and when the transfer does begin the transfer rates will probably not match.

From the foregoing it is obvious that interfaces have a great responsibility to oversee the communication between the computer and its resources. Computer intefacing has some advantages as outlined below:

1. Advanced control applications need flexible processing power which is readily provided by the computer. Hence the computer does the complex control processing and sends signals to control the process through appropriate interfaces.
2. We always need to input and output control data. For example we need inputs from sensors (speed, accelaration, temperature, etc.) while we need to give out utput to actuators (motors, switches, valves). The computer can readily receive inputs and provide corresponding outputs once the right interface has been provided.
3. We are able to access the numerous advantages of using the computer for data acquisition and control such as in high speed proceesing, programming flexibility which is usually unavailable in hard wired logic, mass storage of data, data analysis and visualization and relatively low cost.

FUNCTIONS OF AN INTERFACE

The functions of an interface are shown in the block diagram of Figure1. An interface must ensure electrical and mechaninical compatibility, data compatibility, timing compatibility and some other additional functions as is required for data communication to take place between a computer and its peripherals.

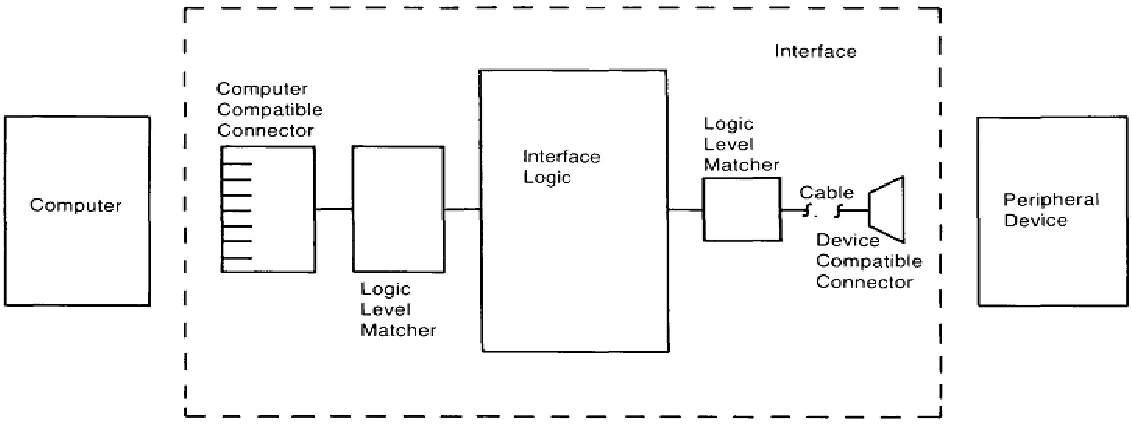


Figure 1: Functional Diagram of an Interface

Electrical and Mechanical Compatibility

Electrical compatibility must be ensured before any thought of connecting two devices occurs. Often the two devices have input and output signals that do not match; if so, the interface serves to match the electrical levels of these signals before the physical connections are made.

Mechanical compatibility simply means that the connector plugs must fit together properly. All of the 9826 interfaces have 100-pin connectors that mate with the computer backplane. The peripheral end of the interfaces may have unique configurations due to the fact that several types of peripherals are available that can be operated with the 9826. Most of the interfaces have cables available that can be connected directly to the device so you don't have to wire the connector yourself.

Data Compatibility

Just as two people must speak a common language, the computer and peripheral must agree upon the form and meaning of data before communicating it. As a programmer, one of the most difficult compatibility requirements to fulfill before exchanging data is that the format and meaning of the data being sent is identical to that anticipated by the receiving device. Even though some interfaces format data, most interfaces have little responsibility for matching data formats; most interfaces merely move agreed-upon quantities of data to or from computer memory. The computer must generally make the necessary changes, if any, so that the receiving device gets meaningful information.

Timing Compatibility

Since all devices do not have standard data-transfer rates, nor do they always agree as to when the transfer will take place, a consensus between sending and receiving device must be made. If the sender and receiver can agree on both the transfer rate and beginning point (in time), the process can be made readily.

If the data transfer is not begun at an agreed-upon point in time and at a known rate, the transfer must proceed one data item at a time with acknowledgement from the receiving device that it has the data and that the sender can transfer the next data item; this process is known as a "handshake". Both types of transfers are utilized with different interfaces and both will be fully described as necessary.

Additional Interface Functions

Another powerful feature of some interface cards is to relieve the computer of low-level tasks, such as performing data-transfer handshakes. This distribution of tasks eases some of the computer's burden and also decreases the otherwise-stringent response-time

requirements of external devices.

MODEM TERMINAL INTERFACES

The word "modem" is a contraction of the words **modulator-demodulator**. A modem is typically used to send digital data over a phone line. The sending modem **modulates** the data into a signal that is compatible with the phone line, and the receiving modem **demodulates** the signal back into digital data. **Wireless modems** convert digital data into radio signals and back. Modems came into existence in the 1960s as a way to allow terminals to connect to computers over the phone lines. A typical arrangement is shown below in Figure 2.

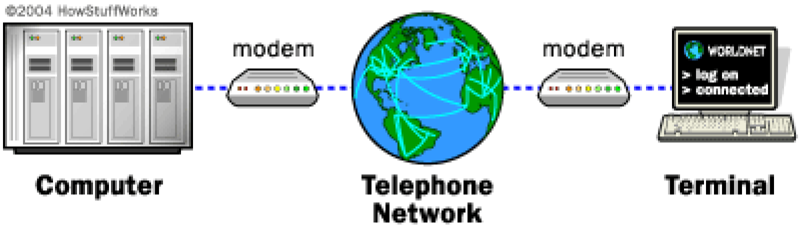


Figure 2: MODEM Interface

In a configuration like this, a **dumb terminal** at an off-site office or store could "dial in" to a large, central computer. The 1960s were the age of **time-shared** computers, so a business would often buy computer time from a time-share facility and connect to it via a 300-bit-per-second (bps) modem. A dumb terminal is simply a keyboard and a screen. A very common dumb terminal at the time was called the **DEC VT-100**, and it became a standard of the day (now memorialized in terminal emulators worldwide). The VT-100 could display 25 lines of 80 characters each. When the user typed a character on the terminal, the modem sent the ASCII code for the character to the computer. The computer then sent the character back to the terminal so it would appear on the screen.

When personal computers started appearing in the late 1970s, **bulletin board systems** (BBS) became the rage. A person would set up a computer with a modem or two and some BBS software, and other people would dial in to connect to the bulletin board. The users would run **terminal emulators** on their computers to emulate a dumb terminal.

People got along at 300 bps for quite a while. The reason this speed was tolerable was because 300 bps represents about 30 characters per second, which is a lot more characters per second than a person can type or read. Once people started transferring large programs and images to and from bulletin board systems, however, 300 bps became intolerable. Modem speeds went through a series of steps at approximately two-year intervals:

- 300 bps - 1960s through 1983 or so
- 1200 bps - Gained popularity in 1984 and 1985
- 2400 bps
- 9600 bps - First appeared in late 1990 and early 1991
- 19.2 kilobits per second (Kbps)
- 28.8 Kbps
- 33.6 Kbps
- 56 Kbps - Became the standard in 1998
- ADSL, with theoretical maximum of up to 8 megabits per second (Mbps) - Gained popularity in 1999

300-bps Modems

We'll use 300-bps modems as a starting point because they are extremely easy to understand. A 300-bps modem is a device that uses **frequency shift keying** (FSK) to transmit digital information over a telephone line. In frequency shift keying, a different tone (frequency) is used for the different bits.

When a terminal's modem dials a computer's modem, the terminal's modem is called the **originate** modem. It transmits a 1,070-hertz tone for a 0 and a 1,270-hertz tone for a 1. The computer's modem is called the **answer** modem, and it transmits a 2,025-hertz tone for a 0 and a 2,225-hertz tone for a 1. Because the originate and answer modems transmit different tones, they can use the line simultaneously. This is known as **full-duplex** operation. Modems that can transmit in only one direction at a time are known as **half-duplex** modems, and they are rare.

Let's say that two 300bps modems are connected, and the user at the terminal types the letter "a." The ASCII code for this letter is 97 decimal or 01100001 binary. A device inside the terminal called a UART (universal asynchronous receiver/transmitter) converts the byte into its bits and sends them out one at a time through the terminal's **RS-232 port** (also known as a **serial port**). The terminal's modem is connected to the RS-232 port, so it receives the bits one at a time and its job is to send them over the phone line.

Faster Modems

In order to create faster modems, modem designers had to use techniques far more sophisticated than frequency-shift keying. First they moved to **phase-shift keying** (PSK), and then **quadrature amplitude modulation** (QAM). These techniques allow an incredible amount of information to be crammed into the 3,000 hertz of bandwidth available on a normal voice-grade phone line. 56K modems, which actually connect at something like 48 Kbps on anything but absolutely perfect lines, are about the limit of these techniques.

All of these high-speed modems incorporate a concept of **gradual degradation**, meaning they can test the phone line and fall back to slower speeds if the line cannot handle the modem's fastest speed.

The next step in the evolution of the modem was **asymmetric digital subscriber line** (ADSL) modems. The word *asymmetric* is used because these modems send data faster in one direction than they do in another. An ADSL modem takes advantage of the fact that any normal home, apartment or office has a **dedicated copper wire** running between it and phone company's nearest mux or central office. This dedicated copper wire can carry far more data than the 3,000-hertz signal needed for your phone's voice channel. If both the phone company's central office and your house are equipped with an ADSL modem on your line, then the section of copper wire between your house and the phone company can act as a purely digital high-speed transmission channel. The capacity is something like 1 million bits per second (Mbps) between the home and the phone company (*upstream*) and 8 Mbps between the phone company and the home (*downstream*) under ideal conditions. The same line can transmit both a phone conversation *and* the digital data.

The approach an ADSL modem takes is very simple in principle. The phone line's bandwidth between 24,000 hertz and 1,100,000 hertz is divided into 4,000-hertz bands, and a **virtual modem** is assigned to each band. Each of these 249 virtual modems tests its band and does the best it can with the slice of bandwidth it is allocated. The aggregate of the 249 virtual modems is the total speed of the pipe.

POINT TO POINT PROTOCOL

Today, no one uses dumb terminals or terminal emulators to connect to an individual computer. Instead, we use our modems to connect to an **Internet service provider** (ISP), and the ISP connects us into the Internet. The Internet lets us connect to any machine in the world. Because of the relationship between your computer, the ISP and the Internet, it is no longer appropriate to send individual characters. Instead, your modem is routing TCP/IP packets between you and your ISP.

The standard technique for routing these packets through your modem is called the **Point-to-Point Protocol** (**PPP**). The basic idea is simple -- your computer's TCP/IP stack forms its TCP/IP datagrams normally, but then the datagrams are handed to the modem for transmission. The ISP receives each datagram and routes it appropriately onto the Internet. The same process occurs to get data from the ISP to your computer.

MODEMS AND ROUTERS

Modems and **routers** are both involved in connecting your home PCs to the Internet. The modem encodes and decodes data so that it can pass between your home network and your Internet Service Provider (ISP). The router, on the other hand, directs the information collected by the modem to devices within that network. The modem brings the information in, and the router distributes (or "routes") it to different devices like computers and phones.

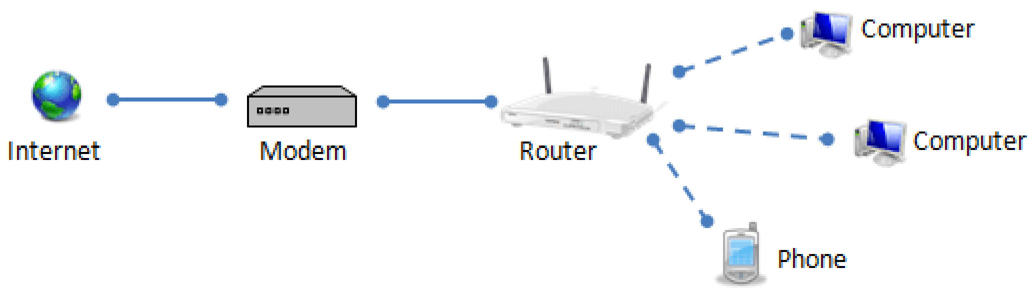
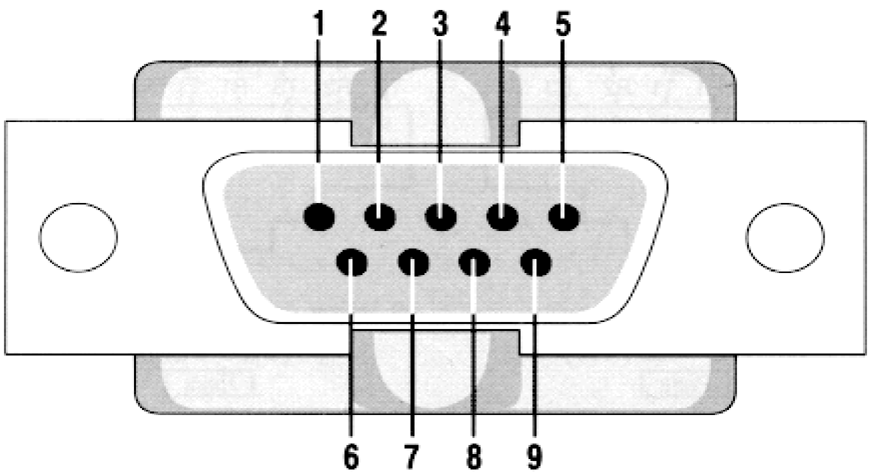


Fig3: Modem and Router

RS 232 Standard

And it's Variants Including RS232C, RS232D, V24, V28 and V10.

The RS-232 serial interface communications standard has been in use for very many years and is one of the most widely used standards for serial data communications because it is simple and reliable. The RS232 serial interface standard still retains its popularity and remains in widespread use. It is still found on some computers and on many interfaces, often being used for applications ranging from data acquisition to supplying a serial data communications facility in general computer environments. The long term widespread use of the RS232 standard has meant that products are both cheap and freely available, and in these days of new higher speed standards, the reliable, robust RS232 standard still has much to offer. The following figures shows RS232 connections



Pin	Signal	Pin	Signal
1	Data Carrier Detect	6	Data Set Ready
2	Received Data	7	Request to Send
3	Transmitted Data	8	Clear to Send
4	Data Terminal Ready	9	Ring Indicator
5	Signal Ground		

Fig 4: RS 232 Interface (9 pins)

Fig 5: RS232C diagram

Fig 6: RS232

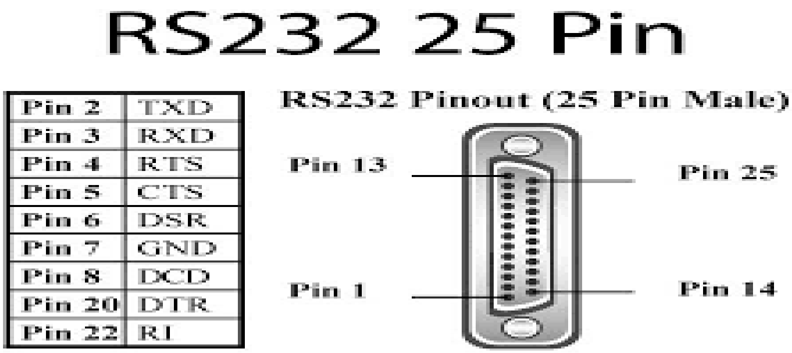


Fig 7: RS232; (25 Pins)

RS-232 Interface Basics

The interface is intended to operate over distances of up to 15 metres. This is because any modem is likely to be near the terminal. Data rates are also limited with a maximum of 19.2 Kbits per second for RS-232C. However slower rates are often used. In theory it is possible to use any baud rate, but there are a number of standard transmission speeds in bps used as follows: 50, 75, 110, 150, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 76800 bps.

Note that speeds up to 19200 bits per second are normally used. Above this speed, noise that is picked up, especially over long cable runs can introduce data errors. Where high speeds and long data runs are required then standards such as RS422 may be used.

RS-232 connections

The RS-232C specification does not include a description of the connector to be used. However, the most common type found is the 25 pin D-type connector.

RS232 signal levels

The voltage levels are one of the main items in the specification. For RS232 data signals a voltage of between -3V and -25V represents a logic 1. The logic 0 is represented by a voltage of between +3V and +25V. Control signals are in the "ON" state if their voltage is between +3V and +25V and "OFF" if they are negative, i.e. between -3V and -25V.

The data is sent serially on RS232, each bit is sent one after the next because there is only one data line in each direction. This mode of data transmission also requires that the receiver knows when the actual data bits are arriving so that it can synchronise itself to the incoming data. To achieve this, a logic 0 is sent as a start bit for the synchronisation. This is followed by the data itself and there are normally seven or eight bits. The receiver obviously has to know how many data bits to expect, and there are often small dual in line switches either on the back of the equipment or inside it to set this information.

Data on RS232 is normally sent using ASCII (American Standard Code for Information Interchange). However other codes including the Murray Code or EBCDIC (Extended Binary Coded Decimal Interchange Code) can be used equally well. After the data itself, a parity bit is sent. Again this requires setting because it is optional and it can be even or odd parity. This is used to check the correctness of the received data and it can indicate whether the data has an odd or even number of logic ones. However there is no facility for error correction unlike what is the case for many systems these days. Finally a stop bit is sent. This is normally one bit long and is used to signify the end of a particular byte. Sometimes two stop bits are required and again this is an option that can often be set on the equipment.

RS232 data transmission is normally asynchronous. However transmit and receive speeds must obviously be the same. A certain degree of tolerance is allowed. Once the start bit is sent the receiver will sample the centre of each bit to see the level. Within each data word the synchronisation must not differ by more than half a bit length otherwise the incorrect data will be seen. Fortunately this is very easy to achieve with today's accurate bit or baud rate generators.

Lines and their usage

There are four types of lines defined in the RS232 specification. They are Data, Control, Timing and Ground. Not all of them are required all the time. It is possible to set up a very simple communication using very few lines. When looking at the lines and their functions it is necessary to remember that they are defined for a connection between a modem (the data set or communications equipment) and a terminal or computer (data terminal equipment) in mind. All the lines have directions, and when used in this way a one to one cable operates correctly.

The most obvious lines are the data lines. There are two of these, one for data travelling in each direction. Transmit data is carried on pin 2 and the receive data is carried on line three (see Figure 7). The most basic of the control circuits is Data Carrier Detected (DCD). This shows when the modem has detected a carrier on the telephone line and a connection appears to have been made. It produces a high, which is maintained until the connection is lost.

Data Terminal Ready (DTR) and Data Set Ready (DSR) are the main control circuits. They convey the main information between the terminal and modem. When the terminal is ready to start handling data it flags this on the DTR line. If the modem is also ready then it returns its signal on the DSR line. These circuits are mainly used for telephone circuits. After a connection has been made the modem will be connected to the line by using DTR. This connection will remain until the terminal is switched off line when the DTR line is dropped. The modem will detect this and release the telephone line.

Sometimes pin 20 is not assigned to DTR. Instead another signal named, Connect Data Set To Line (CDSTL) is used. This is virtually the same as DTR, but differs in that DTR merely enables the modem to be switched onto the telephone line. CDSTL commands the modem to switch, despite anything else it may be doing. A further two circuits, Request To Send (RTS) and Clear To Send (CTS) are also used. These pair of circuits are used together. The terminal equipment will flag that it has data to send. The modem will then return the CTS signal to give the all clear after a short delay.

This signalling is used particularly when switched carriers are used. It means that the carrier is only present on the line when there is data to send. It finds its uses when one central modem is servicing several others at remote locations.

Secondary lines

There are two types of lines that are specified in the RS-232 specification. There are the primary channels that are normally used, and operate at the normal or higher data rates. However, there is also provision for a secondary channel for providing control information. If it is used it will usually send data at a much slower rate than the primary channel. As the secondary lines are rarely used or even implemented on equipment, manufacturers often use these connector pins for other purposes. In view of this it is worth checking that the lines are not being used for other purposes before considering using them. When the secondary system is in use, the handshaking signals operate in the same way as for the primary circuit.

Grounding

The ground connections are also important. There are two. First the protective ground ensures that both equipment are at the same earth potential. This is very useful when there is a possibility that either equipment is not earthed. The signal ground is used as the return for the digital signals travelling along the data link. It is important that large currents that are not part of the signalling do not flow along this line otherwise data errors may occur.

The RS-232 specification is still widely used. Although faster specifications exist, it is likely to remain in use for many years to come. One of the reasons for this is the fact that it is found on most of today's personal computers. Although the parallel "LPT" ports are used almost universally for printers, it still used for many other purposes, including connecting the computer to a modem.

The RS 232 standard has been used in many areas, well beyond its original intended applications. As a result, this has led to uncertainty in the way some applications use the RS232 standard. However the RS 232 standard operates very reliably when correctly set up and for many years it has provided one of the main forms of serial data transmission. Even though many other standards are available for data transmission these days, the RS 232 standard is still widely used, and is likely to remain so for many years to come.

Development of the RS 232 standard

The RS 232 standard for data communications was devised in 1962 when the need to be able to transmit data along a variety of types of lines started to grow. The idea for a standard had grown out of the realisation in the USA that a common approach was required to allow interoperability. As a result the Electrical Industries Association in the USA created a standard for serial data transfer or communication known as RS232. It defined the electrical characteristics for transmission of data between a Data Terminal Equipment (DTE) and the Data Communications Equipment (DCE). Normally the data communications equipment is the modem (modulator/demodulator) which encodes the data into a form that can be transferred along the telephone line. A Data Terminal Equipment could be a computer.

The RS 232 standard underwent several revisions, the C issue known as RS232C was issued in 1969 to accommodate the electrical characteristics of the terminals and devices that were being used at the time. The RS 232 standard underwent further revisions and in 1986 Revision D was released (often referred to as RS232D). This revision of the RS 232 standard was required to incorporate various timing elements and to ensure that the RS 232 standard harmonised with the CCITT standard V.24, while still ensuring interoperability with older versions of RS 232 standard. Further updates and revisions have occurred since then and a newer version is TIA-232-F issued in 1997 under the title: "Interface Between Data Terminal Equipment and Data Circuit-Terminating Equipment Employing Serial Binary Data Interchange." The name of the RS 232 standard has changed during its history, several times as a result of the sponsoring organisation. As a result it has variously been known as EIA RS-232, EIA 232, and most recently as TIA 232.

Variations of the RS 232 standard

There are number of different specifications and standards that relate to RS 232. The RS 232 standard is often referred to by the other related standards and in particular V.24 which is the ITU / CCITT designation for the serial data communications standard. A description of some of the RS 232 standards and the various names and references used is given below:

- ***EIA/TIA-232:*** This reference to the RS 232 standard includes the names of the first and current sponsoring organisations, the Electronic Industries Alliance (EIA) the Telecommunications Industry Alliance (TIA).

- ***RS-232C:*** This was the designation given to the release of RS 232 standard updated in 1969 to incorporate many of the device characteristics.
- ***RS-232D:*** This was the release of the RS 232 standard that occurred in 1986. It was revised to incorporate various timing elements and to ensure that the RS 232 standard harmonised with the CCITT standard V.24.
- ***RS-232F:*** This version of the RS 232 standard was released in 1997 to accommodate further revisions to the standard. It is also known as TIA-232-F.
- ***V24:*** The International Telecommunications Union (ITU) / CCITT (International Telegraph and Telephone Consultative Committee) of the ITU developed a standard known as ITU v.24, often just written as V24. This standard is compatible with RS232, and its aim was to enable manufacturers to conform to global standards and thereby allow products that would work in all countries around the world. It is entitled "List of definitions for interchange circuits between data terminal equipment (DTE) and data circuit-terminating equipment (DCE)."
- ***V28:*** V.28 is an ITU standard defining the electrical characteristics for unbalanced double current interchange circuits, i.e. a list of definitions for interchange circuits between data terminal equipment (DTE) and data circuit-terminating equipment (DCE).
- ***V10:*** V.10 is an ITU standard or recommendation for unbalanced data communications circuits for data rates up to 100 kbps that was first released in 1976. It can inter-work with V.28 provided that the signals do not exceed 12 volts. Using a 37 pin ISO 4902 connector it is actually compatible with RS423.

RS-232 Applications

The RS-232 standard has come a long way since its initial release in 1962. Since then the standard has seen a number of revisions, but more importantly, RS232 has been used in an ever increasing number of applications. Originally it was devised as a method of connecting telephone modems to teleprinters or teletypes. This enabled messages to be sent along telephone lines - the use of computers was still some way off.

As computers started to be used, links to printers were required. The RS-232 standard provided an ideal method of connection and therefore it started to be used in a rather different way. However its use really started to take off when personal computers were first introduced. Here the RS-232 standard provided an ideal method of linking the PC to the printer.

The RS-232 standard provided an ideal method of linking many other remote items to computers and data recorders. As a result, RS-232 became an industry standard, used in a host of applications that were never conceived when it was first launched in 1962.

The RS 232 standard is very widely used and is probably the most widely used standard for serial data communications over distances. The RS 232 standard has stood the test of time, and being introduced in 1962 it has been in use for well over 45 years.

Serial Port for PC

Standard PC serial ports come in to versions: 9 pin and 25 pin one The functions of those both version are exactly the same, only different kind of connectors and different pinout PC serial port is nowadays usually used for interfacing PC to modem or mouse. Original PC serial port was designed to operate up to 19.2 kbit/s (maximum speed defined in RS-232C standard) but nowadays they can typically go up to 115.2 kbit/s (some special cards can do even faster than that). PC serial port sends and receives data in serial format.   In serial, asynchronous data transfer the individual bits which comprise each data byte are sent one after the other over a single line. In this context, asynchronous means that the clock information is not included with the transmission, so that frequent re-synchronization using start/stop bits is required.

The maximum length specified by RS-232 is only 50 feet (around 15 meters), however much longer lengths are possible with proper shielding on the cable. Generally you can run 9600 bps communication up to 250 feet (80 meters) over shielded data cable or unshielded twisted pair cable in good environment. When using shielded cable and slower data rate longer lengths are possible (up to hundreds of meters in good conditions)

RS449 Basics, Interface and Pinout

The basics of RS449 data communications standard, what it is, the RS449 pinout and the RS 449 interface.

The RS449 or RS-449 interface is a further enhancement of RS232 and RS423. It is aimed at catering for very fast serial data communications at speeds up to 2 Mbps. In order to achieve this RS449 makes some changes when compared to RS232 to the way in which the signals are referenced, while still being able to retain some compatibility with RS232.

The RS499 standard which has now been discontinued is also known by the references EAI-449, TIA-449 and ISO 4902

RS449 interface

One of the ways in which the RS449 data communications standard is able to send at high speeds without stray noise causing interference is to use a differential form of signalling. Earlier data communications standards such as RS232 used signalling that was referenced to earth and while this was easier to implement and cheaper to cable, it introduced limitations into the system.

By using twisted wire pairs for the data lines, any unwanted noise will be picked up by both wires together. As the RS449 receivers use a differential input, and they are not referenced to ground, any noise that is picked up does not affect the input. This means that higher levels of noise can be tolerated without any degradation to the performance to the data communications system.

For the RS449 interface, ten additional circuits functions have been provided when compared to RS232. Additionally three of the original interchange circuits have been abandoned.

In order to minimise any confusion that could easily occur, the circuit abbreviations have been changed. In addition to this the RS449 interface requires the use of 37 way D-type connectors and 9 way D-type connectors, the latter being necessary when use is made of the secondary channel interchange circuits.

RS449 Primary connector pinout and interface connections

The RS449 primary connector, which is used the one that is used as standard uses a 37 way D-type connector. The pinout and connections are given in the table below:

Pin	Signal Name	Description
1		Shield
2	SI	Signal Rate Indicator
3	n/a	Unused
4	SD-	Send Data (A)
5	ST-	Send Timing (A)
6	RD-	Receive Data (A)
7	RS-	Request To Send (A)
8	RT-	Receive Timing (A)
9	CS-	Clear To Send (A)
10	LL	Local Loopback
11	DM-	Data Mode (A)
12	TR-	Terminal Ready (A)
13	RR-	Receiver Ready (A)
14	RL	Remote Loopback
15	IC	Incoming Call
16	SF/SR+	Signal Freq./Sig. Rate Select.
17	TT-	Terminal Timing (A)
18	TM-	Test Mode (A)
19	SG	Signal Ground
20	RC	Receive Common
21	n/a	Unused
22	SD+	Send Data (B)
23	ST+	Send Timing (B)
24	RD+	Receive Data (B)
25	RS+	Request To Send (B)
26	RT+	Receive Timing (B)
27	CS+	Clear To Send (B)
28	IS	Terminal In Service
29	DM+	Data Mode (B)
30	TR+	Terminal Ready (B)
31	RR+	Receiver Ready (B)
32	SS	Select Standby
33	SQ	Signal Quality

34	NS	New Signal
35	TT+	Terminal Timing (B)
36	SB	Standby Indicator
37	SC	Send Common

RS449 primary connector pinout and connections

Within the RS449 interface a number of differential connections are defined. In the pinout table above they are labelled as either "A and B" or "+" and "-". When setting up a connection, it is necessary to ensure that the correct polarities are used. As twisted pairs are used for the A and B connections, it is often possible to mix them. If this happens the interface will not work.

RS449 Auxilliary connector

A second connector is defined for use when the secondary channel interchange circuits are needed.. This connector uses a 9 way D-type connector.

Pin	Signal Name	Description
1		Shield
2	SRR	Secondary Receive Ready
3	SSD	Secondary Send Data
4	SRD	Secondary Receive Data
5	SG	Signal Ground
6	RC	Receive Common
7	SRS	Secondary Request to Send
8	SCS	Secondary Clear to Send
9	SC	Send Common

RS449 secondary connector

The RS449 data communications interface is an interface standard that is able to provide data communications with speeds of up to 2 Mbps. Retaining some similarities to RS232, it is a more comprehensive interface capable of greater speeds and operation with greater levels of data integrity.


TTC
EIA-530 / MIL-188C
RS-449 / MIL-188-114
RS-449 / MIL INTERFACE ADAPTOR
TX DATA
TX CLK OUT
EXT TX CLK IN
MODEL 41400
RS-449 INTERFACE
SIGNAL DESIGNATION
PIN NUMBER
PIN NUMBER
SIGNAL DESIGNATION
Receive Common 20
21
Send Data 22
Send Timing 23
Receive Data 24
Request to Send 25
Receive Timing 26
Clear to Send 27
Terminal in Service 28
Data Mode 29
Terminal Ready 30
Receiver Ready 31
Select Standby 32
Signal Quality 33
New Signal 34
Terminal Timing (B) 35
Standby/Indicator 36
Send Common 37
1 Shield
2 Signaling Rate Indicator
3
4 Send Data
5 Send Timing
6 Receive Data
7 Request to Send
8 Receive Timing
9 Clear to Send
10 Local Loopback
11 Data Mode
12 Terminal Ready
13 Receiver Ready
14 Remote Loopback
15 Incoming Call
16 Select Frequency
17 Terminal Timing
18 Test Mode
19 Signal Ground

Fig: RS 449