What the heck is RS-232 anyway…

It just occurred to me that there is an entire generation of technicians in the work force that did not grow up in the days of the TRS-80 and Commodore 64 PC. Rather, they were brought up on broadband and Wi-Fi. To them that funny looking 9-Pin connector that we old folks cut our teeth on is a mystery full of uncertainty and doubt – with a little bit of fear mixed in for good measure. In this paper, I hope to dispel some of the mystery and give you the essentials of RS-232.

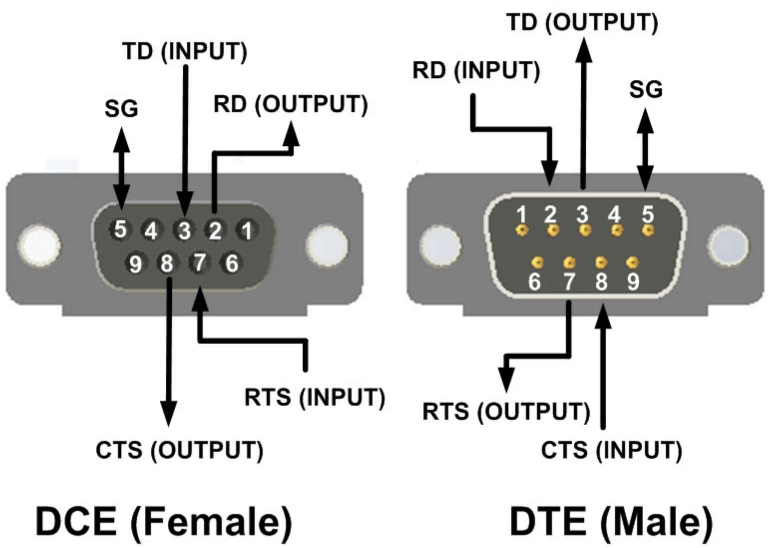
What is it?

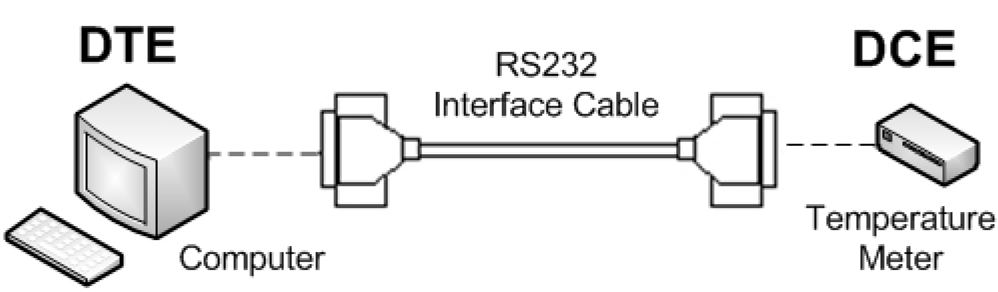
At one time, RS-232 / EIA-232 was the most widely used communication standard on the planet. It was defined and redefined many times. The "EIA" stands for "Electronic Industries Association" and the "RS" stands for "Recommended Standard." That being the case, it was always rather loose. The physical characteristics of the hardware include both a 25 pin and 9 pin D sub connector. RS-232 is capable of operating at data rates up to 20 Kbps and can push data about 50 ft. The absolute maximum data rate is difficult to nail down due the differences in the transmission line and cable length. It is possible to operate at some pretty high data rates if the distance is short. The voltage levels are defined as a range from -12 to +12 volts. RS-232 is also single ended. This means that a single electrical signal is compared to a common signal (ground) to determine binary logic states. A voltage of +12 volts (usually +3 to +10 volts) represents a binary 0 and -12 volts (-3 to -10 volts) is a binary 1. Also, RS-232 is an asynchronous serial protocol. This implies that the data word is transmitted as single bits to the receiver, which puts the word back together. To make this happen correctly, the data rate, hand shaking, start & stop bits, and error checking must all be pre-defined. RS-232 is also an unbalanced protocol. It uses a single conductor for each signal and a common ground. The signal level is relative to the common ground. This method is cheap and easy. However, it is also susceptible to noise and almost always requires a lower data rate then balanced protocols such as RS-422 or RS-485. The effective distance is roughly 50 feet or about 15 meters.

Connections

One of the confusing things about RS-232 is the cable pin-out and the dual nature of the signals. In some cases Transmit Data may be an input and in others it is an output. To those of us that have been around awhile, it is second nature. However, it can create problems to those that are unaware. RS-232 defines two types of equipment. The first one is the Date Terminal Equipment or DTE. The other is the Data Communication Equipment. These terms where introduced by IBM and are used to differentiate the different device types at each end of the cable. The DTE is the computer or terminal that serves as the data source or the data sink. It also provides the control functions. The DTE usually has a male DB connector. The DCE is the MODEM or peripheral device. It typically has a female DB connector. Hooking up a DTE to a DCE is easy. The cabling is straight through. Pin x goes to Pin x on the other side. You run into problems when you try to hook up two DTE’s or DCE’s to each other. In this case you need to use a “Null-Modem” cable which crosses transmit and receive lines. This makes more sense if you review the following chart and connector diagram.

RS-232 Function	Pin Number DB25	DB9	Input to DTE	Input to DCE
Shield	1			
Transmit Date (TD)	2	3		
Receive Data (RD)	3	2		
Request to Send (RTS)	4	7		
Clear to Send (CTS)	5	8		
DCE Ready (DSR)	6	6		
Signal Ground (SG)	7	5		
Received Line Signal Detector (DCD)	8	1		
DTE Ready (DTR)	20	4		
Ring Indicator (RI)	22	9		





Control Signals

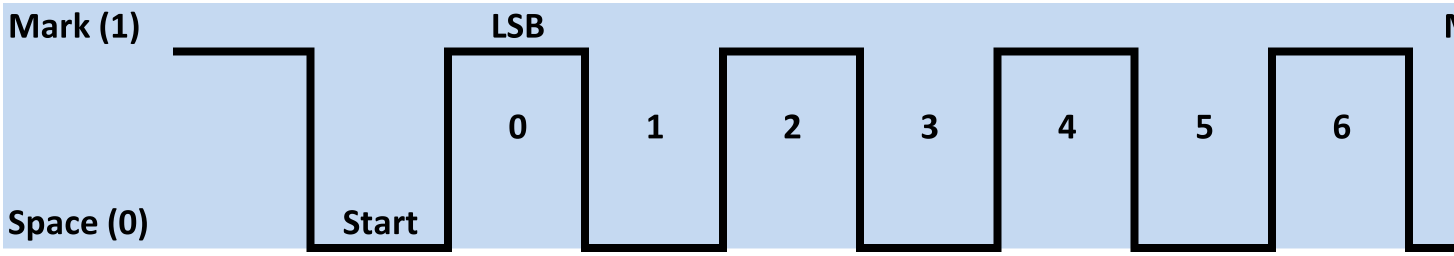
The hardware control signals are also confusing to some people. For the most part, we only use the actual data (TD and RD). In some cases, the hardware may be required to act as a traffic cop by asserting or RTS and CTS. Here is a list of the available control signals and what they do.

RS-232 Function	Purpose
Transmit Date (TD)	Carries data from DTE to DCE
Receive Data (RD)	Carries data from DCE to DTE
Request to Send (RTS)	DTE requests DCE to prepare to receive data
Clear to Send (CTS)	Indicates DCE is ready to accetp data
DCE Ready (DSR)	DCE is ready to receive commands or data
Received Line Signal Detector (DCD)	DCE is connected to the line
DTE Ready (DTR)	Indicates presence of DTE to DCE
Ring Indicator (RI)	DCE has detected and incomming ring signal on the line

In some applications RS-232's use of the RTS and CTS lines is asymmetric: The DTE asserts RTS to indicate a desire to transmit to the DCE, and the DCE asserts CTS in response to grant permission. This allows for half-duplex modems that disable their transmitters when not required, and must transmit a synchronization preamble to the receiver when they are re-enabled. Most of the time though, the flow control is handled by the software and the RTS and CTS lines do not play a significant role in communications. However, these lines do present a handy but unplanned feature. Instead of letting the power available on these lines go to waste, they can be used to power a serial converter or other device. This is often called “port powering.” Be careful though. Some manufacturers provide an externally sourced regulated 5 VDC power on Pin 1. This is used to power devices that have a higher current draw than can be supplied by the RS-232 port.

Protocol

The communications speed is measured in baud. Simply put this is a measure of bits transferred per second. For example, 19200 baud is 19200 bits per second.



Typical RS-232 Waveform, 1 Start Bit, 8 Data Bits, NO Parity, 1 Stop Bit

Data bits are a measurement of the actual data bits in the word. For example, this diagram shows an 8 bit word. The word is a single byte transfer, including Start/Stop bits, Data bits and Parity. If you are transferring standard ASCII (0 to 127) seven data bits are enough. If it is an extended ASCII code (128 to 255), then 8 data bits are needed.

Parity is a simple way to error-check. Parity is either Even, Odd, Mark or Space. You can also use no parity. For Even and Odd parity, the serial port sets the parity bit (the last bit after the data bit) to a value to ensure that the data packet has an Even or Odd number of logic-high bits. For example, if the data is 10010010, for Even parity, the serial port sets the parity bit as 1 to keep the number of logic-high bits Even. For Odd parity, the parity bit is 0 so that the number of logic-high bits is Odd. Mark parity simply sets the parity bit to logic-high and Space sets the parity bit to logic-low, so that the receiving party can determine if the data is corrupted.

Stop bits are used to signal the end of a communication packet. This also helps to synchronize different clocks on the serial devices.

Summary

In summary, RS-232 is not nearly as mysterious as it seems. If you have this basic knowledge, you should be able to utilize this valuable communications port now and in the future. As far as the demise of the RS-232 port goes, who knows? The prognosticators have been threatening us with that for years but billions of ports remain active in the industrial world. I think this old work horse will be with us awhile longer. I hope this takes some of the mystery out of RS-232. If you have any questions, please contact our support group. They will be happy to help solve any and all RS-232 problems you may run into.