200-RS232 In Depth

RS-232 was created for one purpose, to interface between Data Terminal Equipment (DTE) and Data Communications Equipment (DCE) employing serial binary data interchange. So as stated the DTE is the terminal or computer and the DCE is the modem or other communications device.

HISTORY

The Electronic Industries Association (EIA) originally adopted RS-232 in 1960. The standard evolved over the years and in 1969 the third revision (RS-232C) was to be the standard of choice of PC makers. In 1987 a fourth revision was adopted (RS-232D also known as EIA-232D). For the most part this new revision added 3 additional test lines. In this document we will work with the original RS-232C standard, which is the one used in the PC world.

Most equipment using RS-232 serial ports use a DB-25 type connector even though the original documents didn't specify a connector. Many PCs today use DB-9 connectors since all you need in asynchronous mode is 9 signals. But take note that the document does specify the amount of pins and their assignment: 20 affected to different signals; 3 are reserved; and 2 are not affected. Normally the male connector is on the DTE side and the female connector is on the DCE side, but this is not always the case.

SERIAL COMMUNICATIONS

The concept behind serial communications is as follows; data is transferred from sender to receiver one bit at a time through a single line or circuit. The serial port takes 8, 16 or 32 parallel bits from your computer bus and converts it as an 8, 16 or 32 bit serial stream. The name serial communications comes from this fact; each bit of information is transferred in series from one location to another. In theory a serial link would only need two wires, a signal line and a ground to move the serial signal from one location to another. But in practice this doesn't really work over time. Some bits might get lost in the transmission and thus alter the ending result. If one bit is missing at the receiving end, all succeeding bits are shifted resulting in incorrect data when converted back to a parallel signal. So to establish reliable serial communications you must overcome these bit errors that can emerge in many different forms.

SERIAL TRANSMISSION METHODS

Two serial transmission methods are used to correct serial bit errors. One is synchronous communication, the sending and receiving ends of the communication are synchronized using a clock that precisely times the period separating each bit. By checking the clock the receiving end can determine if a bit is missing or if an extra bit (usually electrically induced) has been introduced in the stream.

An example of this method of communication: Let's say that on a conveyor belt a product is passing through a sensing device every 5 seconds. If the sensing device senses something in between the 5-second interval it assumes that whatever is passing is a foreign object of some sort and sounds an alarm. If on the 5-second interval nothing goes by, it assumes that the product is missing and sounds an alarm.

One important aspect of this method is that if either end of the communication loses its clock signal, the communication is terminated.

The alternative method is known as **asynchronous communication**. This is the method used most often in PCs. This method adds markers within the bit stream to help track each data bit. By introducing a start bit which indicates the start of a short data stream, the position of each bit can be determined by timing the bits at regular intervals. By sending

start bits in front of each 8-bit stream, the two systems don't have to be synchronized by a clock signal. The only important issue is that both systems must be set at the same port speed. When the receiving end of the communication receives the start bit it starts a short-term timer. By keeping streams short, there's not enough time for the timer to get out of sync. This method is known as **asynchronous communication** because the sending and receiving end of the communication are not precisely synchronized by the means of a signal line.

Each stream of bits is broken up in 5 to 8 bits called words. Usually in the PC environment you will find 7 or 8 bit words, the first is to accommodate all upper and lower case text characters in ASCII codes (the 127 characters) the latter one is used to exactly correspond to one byte. By convention, the least significant bit of the word is sent first and the most significant bit is sent last. When communicating the sender encodes the each word by adding a start bit in front and 1 or 2 stop bits at the end. Sometimes it will add a parity bit between the last bit of the word and the first stop bit, this used as a data integrity check. This is often referred to as a data frame.

Five different parity bits can be used:

The **mark parity** bit is always set at a logical 1.

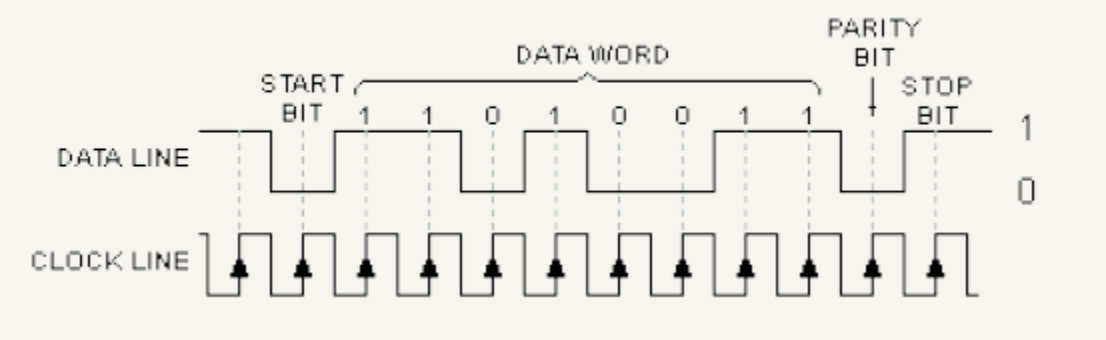
The **space parity** bit is always set at a logical 0.

The **even parity** bit is set to logical 1 by counting the number of bits in the word and determining if the result is even.

In the **odd parity** bit, the parity bit is set to logical 1 if the result is odd.

The latter two methods offer a means of detecting bit level transmission errors.

A **non parity bit frame** uses no parity bits, thus eliminating 1 bit in each frame.



Asynchronous Serial Data Frame (8E1)[fig.1]

In the example above you can see how the data frame is composed of and synchronized with the clock signal. This example uses an 8-bit word with even parity and 1 stop bit also referred to as an 8E1 setting.

BIT RATES

Another important part of every asynchronous serial signal, is the bit rate at which the data is transmitted. The rates at which the data is sent is based on the minimum speed of 300 BPS (bits per second), you may find some slower speeds of 50, 100 and 150 BPS, but these are not used in today's technology. Originally, faster speeds were all based on the 300 BPS rate. You merely doubled the preceding rate, so the rates were as follows, 600, 1200, 2400, 4800, 9600, 19200 and 38400. In the late 1990's the maximum communication rate by standard dial-up modem was 56,000 BPS. Because of this the doubling rule was violated to create the 57,000 rate between PCs and modems that supported the 56K rate. Prior to this the 38,400 rate was the fastest speed supported by today's BIOS's. Note that a few years ago the fastest speed was of 19200 BPS, due to the strain exercised on the CPU because of the software control used to control the serial

port. Today with the new Micro Channel, EISA, VL Bus and PCI motherboards, the new systems take advantage of bus mastering DMA control, which have pushed rates up to 38,400 by eliminating microprocessor overhead. By bypassing the BIOS all together and controlling the hardware directly, you can obtain greater speeds.

Here is the list of all signals specified in the RS232C standard. Each signal is identified by its letters, V.24 equivalent (CCITT), pin

number on a DB-25 and DB-9 connector and its signal name. The circuit letters associated to each signal are devised by the following:

If the first letter is A, this is a common circuit

If the first letter is B, this is a signal circuit

If the first letter is C, this is a control circuit

If the first letter is D, this is a timing circuit

If the letters are preceded by an S, this is a secondary channel

CIRCUIT	V.24 Circuit #	DB-25 Pin #	DB-9 Pin #	Signal Name
AA	101	1	-	Protective Ground
AB	102	7	5	Signal Ground
BA	103	2	3	Transmitted Data
BB	104	3	2	Received Data
CA	105	4	7	Request to send
CB	106	5	8	Clear to send
CC	107	6	6	Data set ready
CD	108	20	4	Data terminal ready
CE	125	22	9	Ring detector
CF	109	8	1	Carrier detect
CG	110	21	-	Signal quality detect
CH/CI	111/112	23	-	Data signal rate selector
DA	113	24	-	Transmitter signal timing (DTE)
DB	114	15	-	Transmitter signal timing (DCE)
DD	115	17	-	Receiver signal timing
SBA	118	14	-	Secondary TX
SBB	119	16	-	Secondary RX
SCA	120	19	-	Secondary RTS
SCB	121	13	-	Secondary CTS
SCF	122	12	-	Secondary CD
		9	-	Reserved Positive test
		10	-	Reserved Negative test
		11	-	N/C
		18	-	N/C
		25	-	N/C

Common circuits

AA: Protective ground

This line is connected to the power ground of the serial adapter. It should not be used as signal ground. Connect this line to the screen of the lead wire (if applicable). By connecting this line on both sides you make sure that no large currents flow through the signal ground in case of an insulation defect or other defect on either side. On the other side, when great distances separate two devices you may not wish to use this signal, because of different ground potential and it is possible that it may carry a substantial current as a ground loop. If it is great enough, it may cause electrical interference.

AB: Signal ground

This is the logical ground, which is used as a point of reference for all signals received or transmitted. This signal is very important and must be present for all communications.

Signal circuits

BA: Transmitted data

This line is used to transmit data from the DTE to the DCE. It is maintained at a logical 1 state when nothing is transmitted. The terminal will start to transmit when a logical 1 is present on all of the following lines:

Clear To Send

Data Terminal Ready

Data Set Ready

Data Carrier Detect

The standard specifies the output levels as being -5 to -15 Volts for logical 1 and +5 to +15 Volts for logical 0, and the input levels as being -3 to -15 Volts for logical 1 and +3 to +15 Volts for logical 0. This ensures data bits to be read correctly even at maximum lengths between DTE and DCE, which is specified as 50 feet although you could probably go much greater distances. As you may have noticed, logical 1 are represented by a negative tension and vice versa. There's no particularly good reason for the inversion except that it's the way things have always been done, why change when it works!

BB: Received data

This circuit is used to receive data from the DCE to the DTE. The terminal will start to transmit when a logical 1 is present on all of the

following lines:

Request To Send

Data Terminal Ready

Data Set Ready

Data Carrier Detect

Control circuits

CA: Request To Send

On this line, the DTE will send a signal when it wants to receive data from the DCE.

CB: Clear To Send

Here the DCE will send a signal when it's ready to receive data from the DTE. (ex. When your local modem connects to an other modem via telephone lines).

CC: Data Set Ready

At a logical level of 1, this line indicates to the DTE that the DCE is ready to send data. (ex. When a modem has established a connection with a remote modem and is in transmission mode).

CD: Data Terminal Ready

When a logical level 1 is sent from the DTE the DCE can start to send and receive data. When this line passes to logical level 0 the DCE will stop all communications. (ex. A modem would stop all communications and would disconnect from the line, you will often see "DROP DTR" in communication programs).

CF: Data Carrier Detect

On this line the DCE indicates to the DTE that it has established a carrier with a remote device.

CE: Ring Indicator

This line is used mostly by communications software when the modem is not in "auto answer" mode and will indicate to the software that a remote device is calling. This is signal is optional when not using software that will answer a phone call automatically.

CG: Signal quality

This line although rarely used serves to indicate to the DTE that the quality of the signal is poor or just not good enough to keep a good connection.

CH: Data signal rate selector

In the case where a modem able of multiple connection rates, the DTE could choose the speed at which it is connected. Usually this line is kept a logical level 0, which selects the highest speed.

CI: Data signal rate selector

This signal is the same as CH but in this case the modem selects the speed at which the DTE communicates.

Timing circuits

In synchronous mode, it is necessary to have some way to exchange clock signals, here are three timing circuits used in the RS-232 protocol.

DA & DB: Transmitter signal timing

DA: DTE towards DCE (clock part of the DTE).

DB: DCE towards DTE (clock part of the DCE).

These two circuits are used to synchronize the flow of data. Timing is given by the DTE or DCE, but never from both at the same time. Usually data is transmitted to the modem or it's own clock control on the DB circuit.

DD: Receiver signal timing DCE

DD: DCE towards DTE (clock part of the DCE).

This circuit is used to synchronize data received from the DTE. The clock signal received on this line indicates to the DTE at which instant to sample the received data on the BB line.