

SERIAL BACKGROUNDER (For Asynchronous Serial Communication)

INTRODUCTION

Serial data communication is the most common means of transmitting data from one point to another. In serial communication systems, the data or characters are sent bit by bit over some kind of transmission path. The receiving device recognizes the bits and reassembles them back into the original data word. Serial data communication systems can be divided into binary and character oriented systems. Binary systems are generally used to send high-speed data between computers and other devices. Binary transmission systems often include data, clock and sync or frame signals. Character oriented serial systems encode characters into bit patterns that can be read on a wide variety of computer terminals, teletypes, printers etc. Asynchronous characters include clocking and syncing as part of the character design. This application note describes asynchronous character serial communication since it is the type that we run into on a daily basis when communicating with PCs and other serial devices.

CHAPTER 1 - HISTORY

Asynchronous serial communication began as a way to communicate between early teletype machines. Characters were formed by mechanical rotors that scanned contacts when a key was depressed. The characters used a 5 bit 'Baudot' code with a start and two stop bits. Initial baud rates were slow, originally 33 baud. Data transmission used a 20 mA current path where a break in the current was a logic '0'. Special modems from Western Union were used to communicate over dedicated phone lines. Later improvements raised the baud rate to 110 baud and expanded the character format to 7 data bits plus a parity bit.

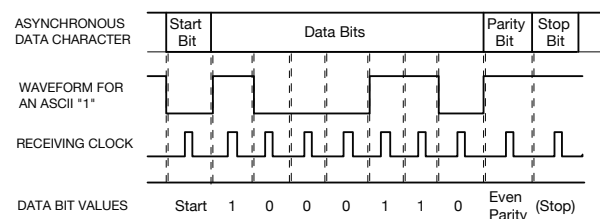
In 1963, the Electronic Industry Association (EIA) established the RS-232 standard to govern the interface between data terminal equipment (DTE) and data communication equipment (DCE) employing serial data. The RS-232 Standard converted the signaling method to a bipolar voltage. AT&T and other companies introduced 300 and 1200 baud modems that worked over the dial-up phone network. This made possible communication between remote computers and local terminals. In the 1980s, major modem advances made 9600 baud a common rate. A decade later, modem baud rates had increased to the 57.6 kbaud common today in every PC.

CHAPTER 2 - DATA TRANSMISSION

DATA FORMAT

Data format refers to the pattern the transmitter uses to send the data or characters so that the receiver will know how to recognize the pattern and reassemble the bits back into the original data. The most common format is called asynchronous characters because each character is sent one character at a time with a minimum amount of time between characters. Each asynchronous character has a low going Start Bit, a number of data bits, an optional Parity Bit and 1 or 2 high Stop Bits. The transmitter holds the transmission line in the Stop Bit (mark) level when it has no characters to transmit.

The receiver uses the Start Bit to synchronize its receive clock with the center of the data bit at the start of each character as shown in Figure 1. The receive clock stops after inputting the Stop Bit and restarts on the next Start Bit. Timing is derived from separate oscillators in the transmitter and in the receiver.



Note: Waveform drawn with bits shown high true

Figure 1 Asynchronous Data Character Waveforms

The data portion of the serial character contains 5 to 8 data bits and is transmitted least significant bit first. The original 5 data bits in the Baudot code were expanded to 7 data bits when the ASCII code chart with 128 characters was introduced. The Parity Bit was used as a way to detect faulty characters by forcing all characters to have an even or odd number of bits. Parity checking only detects single bit errors and has fallen out of use as data transmission became more reliable and people wanted to use the 8th bit as a data bit. Today, most computers and other serial devices use asynchronous characters with 8 data bits, no parity and one Stop Bit.

DATA SPEED

Serial data speed is referred to as 'Baud Rate'. A baud is defined as a signaling bit, which includes data bits as well as start/stop framing, parity or any other bits that make up the data format. Typical computer baud rates and their uses are:

- 110, 300, 1200, 2400, 9600 - for low speed devices, teletypes and older modems
- 14400, 28,800, 38.4K, 57.6K - for higher speed devices and newer modems
- 57.6K, 64K, and 115.2K - for device-to-device and network communication

As data speeds have become faster, the term 'baud' has become dated and has been replaced in many instances by the abbreviation 'kbs'. kbs stands for thousand bits per second. The terms are interchangeable, just remember it takes 1000 baud for 1 kbs. i.e. 9600 baud is 9.6 kbs or 9.6 kbaud.

Bytes per second or characters per second is another measure of data rate. Today it is common to see the term kbytes/sec. The 57.6 kbaud rate divided by 10 bits/character becomes 5.76 kbytes/sec.

CLOCKING METHODS

In asynchronous communication systems, each device uses its internal clock to transmit and receive data characters. The receive clock starts on the falling edge of the start bit and generates read strobes in the middle of each bit. The receiver's oscillator only has to be accurate to $\pm 0.1\%$ so it can accurately strobe in the data. In actual practise, most oscillators are accurate to $\pm 0.02\%$ or better.

In isochronous transmission systems, all characters are generated from a master clock so that their bit transitions occur at known times. The master clock is normally supplied by the transmission system or by the modems at a 1X or 16X times bit rate. Isochronous transmission is commonly used with high speed systems that multiplex many different serial data links together into a higher speed serial data link such as a T1 line.

Synchronous data transmission is form of character oriented serial communication where the characters do not have start/stop bits and are sent continuously without spaces between characters. Messages typically start with a unique header or preamble that allows the receiver to synchronize to the message. Voids between messages are filled by predetermined fill characters which are discarded by the receiver. The transmitter's clock is sent with the data so the receiver does not have to generate a receive clock from its internal oscillator. Synchronous characters are typically used for network communication systems.

TRANSMISSION MEDIUM

Although serial data can be transmitted over any medium, most of today's computer systems use metallic cable for distances up to 200 feet. Beyond 200 feet, the signal can be converted in to a differential signal that can go up to 12,000 feet on twisted-pair cable. When data must be transmitted over long distances, it is typically sent over the phone company's direct dial network (DDN) as shown in Figure 2. Modems are used to convert the serial data bits into tones that will pass through the telephone system's 300 to 3000 Hz voice band. Early modems converted low speed data bits (110 to 1200 baud) into two tones (frequency switched keying) that the receiving modem recognized and converted back to data bits. Today's high speed modems use multiple tones and phase encoding techniques to stay within the telephone system's 3 kHz bandwidth. These modems operate at speeds of 9,600, 14,400 and up to 57,600 baud. Data compression techniques can extend the effective baud rate up to 115.2 kbaud. The 56K modem used in today's PC computers typically communicates at rates from 20,000 to 40,000 baud. The data rate is a function of the quality of the switched telephone line.

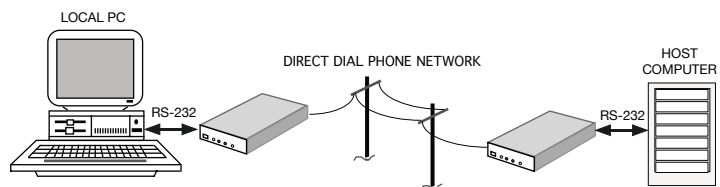


Figure 2 Long Distance Transmission Over Telephone Lines

FLOW CONTROL

Another aspect of serial data transmission is the control of the transmitter to avoid data overrun at the receiver. Serial transmission systems can work in a full-duplex mode where either device can transmit at the same time or in a half-duplex mode where only one device transmits at a time. The three common methods used to control data transmission are: control signals, X-on/X-off characters and fixed transmission protocols.

In the control signal method, extra wires are provided in the cable for handshake signals that enable or inhibit data flow. The common control or handshake signal pairs are:

- Request-to-Send / Clear-to-Send Used by the DTE to request the DCE to transmit. DCE replies by raising Clear-to-Send when ready to accept data.
- Data-Terminal-Ready / Data-Set-Ready Used by DTE and DCE to indicate to the other unit that they are on and buffers are ready to accept data.

All control signals must be high to enable data transmission. The receiving device drops the control line to stop data transmission when its buffer is full or it is busy with the last message. This technique can be used with any kind of serial data. Modems (DCEs) provide a third signal called Data Carrier Detect to enable the DTE's data transmission when two modems are receiving each other's data carrier.

The second method of controlling the data flow is to imbed X-on/X-off characters in the data message. At turn on, both devices are initially in the X-on state. When one device becomes full, it sends the other device an X-off character to inhibit future data transmission. An X-on character is then sent to restart the data transmission when there is room in the receiving device's buffer for additional data. This technique only works with ASCII data since binary characters can be mistaken for the X-on/X-off characters. This technique is also useful when the serial transmission link does not have extra wires to handle flow control signals.

Known message protocols can be used to eliminate the possibility of overrunning the receiving device's buffer. Examples are the command-response sequences used for industrial devices, sensors, external data acquisition devices and instruments. These devices only transmit when queried and the command/response messages have limited lengths that will not overflow the receiving device's serial receive data buffer.

CHAPTER 3 - SERIAL STANDARDS

To ensure compatibility among serial products, manufacturers have adopted interface standards so they are electrically compatible. The more popular standards are:

- RS-232 Original standard for office machines and computer systems. Uses single ended bipolar signals. Cable length limited to 50 foot.
- RS-423 New single ended signal standard.
- RS-422 and RS-485 New high-speed standard for longer distance transmission. Uses differential signals and transmits up to 1,200 meters.
- RS-449 New connector standard for RS-422 and RS-423 signals. Uses DC shell.
- RS-530 New connector standard that combines RS-232 and RS-422/RS-485 signals on the same connector. Uses DB shell.

Devices employing the same interface standard can usually be connected together but the user should verify each devices' signal requirements before plugging them together.

CHAPTER 2 - RS-232 STANDARD

The RS-232 standard specifies:

1. Electrical and Mechanical characteristics of the interface
2. A number of interchange circuits with descriptions of their functions
3. The relationship of interchange circuits to standard interface types
4. A maximum cable length of 50 foot baud rate limit of 20,000.

Functionally, the RS-232 specification established two types of devices, DCE and DTE, that mate together with a pin-to-pin cable. The Data Communication Equipment (DCE) was designated as the device that connects to the communication line. A modem is a DCE device. The Data Terminal Equipment (DTE) was designated as the device that connects to the DCE. A terminal, a printer with a serial interface, computers and most serial devices are designed as DTE devices. DTE devices can be mated to DTE devices by a special 'null-modem' cable that crosses the data and handshake signals of one device with the same signals on the other device.

The RS-232 Standard specifies a 25-pin male connector (DB-25P) for the data terminals (DTEs) and a 25-pin female connector (DB-25S) for the data communications units (DCEs). IBM compatible PCs popularized the use a 9-pin male connector (DE-9P) for their second RS-232 serial port. Nine-pin serial connectors are also used on a number of other serial devices due to their smaller size.

The major signals and their definitions are:

Table 1 RS-232 Signals

RS-232 Signal	Name	Function
BA	Transmit Data	Transmit data to DCE (TxD)
BB	Receive Data	Received data from DCE (RxD)
CA	Request-to-Send	Transmit request to modem (RTS)
CB	Clear-to-Send	Modem ready response (CTS)
CC	Data Set Ready	Modem on and ready
CD	Data Terminal Ready	Terminal on and ready (DTR)
CF	Carrier Detected	Modem receiving carrier (DCD)
AB	Signal Ground	Signal Ground

The RS-232 Standard also defined 14 additional signals for a secondary data channel, for data transmission clocks and for device testing. These signals are not used by the COM ports on PCs or in most modern serial devices.

SIGNAL LEVELS

The RS-232 Standard specifies a bi-polar signal whose transmit signal levels are -5 to -25 volts for a low and +5 to +25 volts

for a high. The +5 to -5 volt band is the undecided area. A minimum of ±3 volts is required at the receiver input. Older electronic devices outputted data with typical voltage swings of ± 12 to ±18 volts. As technology improved and integrated circuits became common, signal swings were reduced to ±9 and even ±6 volts. Receiver sensitivity has improved from ±3 volts to ±30 mV but the RS-232 Standard still requires ±3 volts at the receiver.

Data and control signals were given opposite signal levels.

Voltage	Data	Control Signals
+5 to +25	Logic '0' (Space)	Logic '1' (On)
-5 to -25	Logic '1' (Mark)	Logic '0' (Off)

The sending device always holds its data transmitter in the '1' or Mark level when not transmitting. Control or handshake lines are held at their appropriate levels.

SERIAL PIN ASSIGNMENTS

The 25-pin connectors have different signal-pin assignments than do the 9-pin connectors. Signal directions depend upon the device designation as a DTE or DCE device. Pinouts are:

Table 2 RS-232 Pinouts

RS-232 Signal	Name	Pin Numbers		Signal Direction
		25-pin	9-pin	
BA	Transmit Data	2	3	From DTE
BB	Receive Data	3	2	To DTE
CA	Request-to-Send	4	7	From DTE
CB	Clear-to-Send	5	8	To DTE
CC	Data Set Ready	6	6	To DTE
CD	Data Terminal Ready	20	4	From DTE
CF	Carrier Detected	8	1	To DTE
AB	Signal Ground	7	5	-

Signal direction is listed for a DTE type device since most of us work with PC COM ports or similar serial devices that are DTE devices.

RS-232 CABLES

The RS-232 Standard specifies that the male connector is associated with DTE devices and that female connectors are associated with DCE devices. DTE and DCE devices can normally be connected with the standard pin-to-pin RS-232 cable shown in Figure 3.

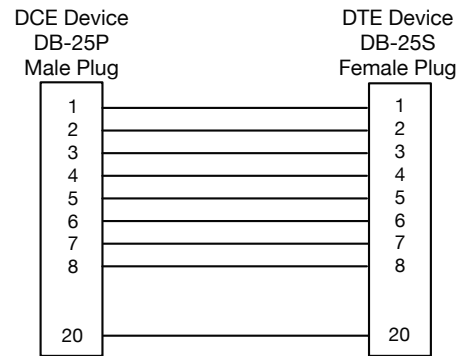


Figure 3 RS-232 Straight Cable

DTE devices can be connected to each other with a null-modem cable that crosses the data and handshake lines as shown in Figure 4.

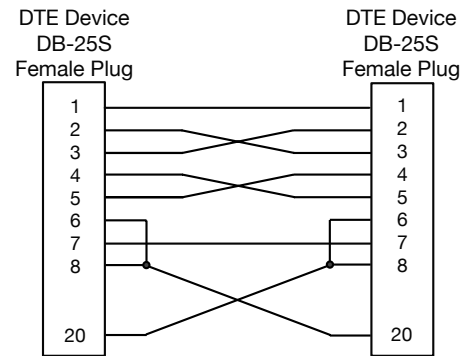


Figure 4 RS-232 Null Modem Cable

If you do not have a null-modem cable, you can use a straight through cable and a null-modem adapter.

If you are making your own cable, you can take advantage of the fact that most RS-232 devices only use Transmit Data, Receive Data and Ground. The remaining handshake lines can be jumpered back as shown in Figure 5 or ignored if they are not used by your serial devices. Select plug types to mate with your serial devices.

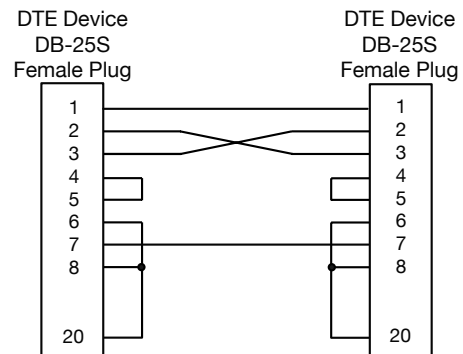


Figure 5 RS-232 Null-Modem Cable with Jumpers

Nine-pin RS-232 cables look like the 25-pin cables shown in the Figures 3 to 5 except for their pin numbers. Use Table 2 to convert 25-pin pin numbers to 9-pin pin numbers. Change the DB-25 connectors to DE-9 connectors.

CHAPTER 5 - THE RS-422 STANDARD

In 1978, the EIA adopted the RS-422 Standard to overcome the distance and noise problems associated with the single-ended RS-232 signals. The RS-422 standard specified low voltage, two-wire differential signals instead of the RS-232's bipolar signals.

RS-422 differential signals have the advantage of higher speed (up to 2Mbs), longer distance capability (up to 1,200 meters) and greatly increased noise rejection. These improvements let users route the serial lines over long distances and through high noise areas such as factory floors or by fluorescent lights that introduced errors into RS-232 signals. RS-422 receivers are specified to have a ± 0.2 V sensitivity, 4 Kohm minimum input impedance and be capable of withstanding a maximum input of ± 10 volts. Cable terminators and transmitter wave shaping may be required to minimize cross talk.

The RS-422 Standard was designed for network and for point-to-point applications. An RS-422 transmitter can drive up to seven RS-422 loads. The RS-422 Standard does not define a connector type.

RS-422 logic levels are:

Signal	+2 to +6V	-2 to -6V
Data A/B	0 (Space)	1 (Mark)
Control A/B	1 (On)	0 (Off)

The differential transmitter output terminal that is negative with respect to the other terminal for the logic '1' data signal is designated the A terminal. The positive terminal is designated the B terminal. All voltage measurements are made by connecting a voltmeter between the A and B terminals.

In network applications, one device, the Network Controller, may transmit to one or more devices. The maximum number of devices on an RS-422 network is 8. Each device is assigned an address and only the addressed device responds to the message or query. Each message from the Network Controller contains the device address. Network messages can be simple ASCII strings with a device address prefixed to the front of the message up to complex packet protocols. There are two types of network circuits, four-wire and two-wire.

The four-wire circuit shown in Figure 6 is often referred to as a full-duplex connection because data can flow in both directions at the same time. The Network Controller's transmitter can stay on all the time since nothing conflicts with it. However, the devices can only assert their transmitters when they are transmitting. The devices must de-assert their transmitters

when not transmitting to avoid blocking another device. A termination network is normally placed on the receiving line to bias the lines in the mark condition when not transmitting. The termination network also terminates the line in its characteristic impedance to minimize reflections.

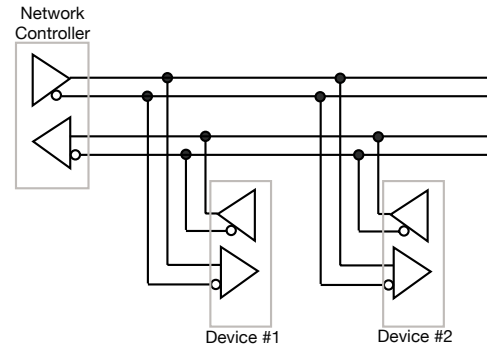


Figure 6 Four-wire Network

The two-wire circuit shown in Figure 7 is often referred to as a half-duplex circuit since data can only flow in one direction at a time. Here all transmitters must be de-asserted when not transmitting and only asserted when transmitting. Each device has to have some echo cancellation logic to prevent receiving its own message.

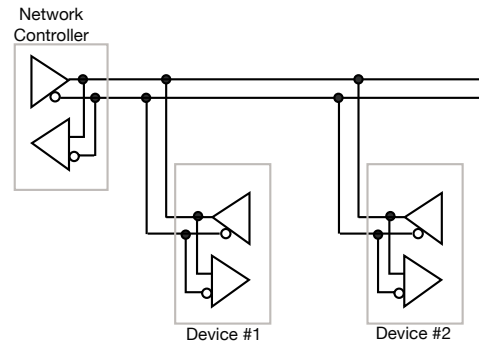


Figure 7 Two-wire Network

CHAPTER 6 - THE RS-485 STANDARD

RS-485 signals are differential signals that are similar to RS-422 signals except that their transmitters are capable of driving up to 32 receivers and distances of 10,000 meters. The RS-422 and RS-485 receiver characteristics are the same. Because of the increased power in the RS-485 transmitter, most differential serial interfaces are implemented with RS-485 circuits. The manufacturer then can identify the serial interface as being 'RS-422 and RS-485 compatible'.

RS-485 circuits can be used in two or four-wire networks. Often people mistakenly refer to the four-wire circuit shown in Figure 6 as an 'RS-422 Network' and to the circuit shown in Figure 7 as an 'RS-485 Network'. They make this assumption because most devices with RS-485 interfaces are

only two-wire interfaces. These references are wrong as RS-422 and RS-485 circuits can be used in both network configurations.

The EIA RS-485 Standard does not specify a connector type.

CHAPTER 7 - OTHER STANDARDS

RS-423 STANDARD

In 1978, the EIA adopted the RS-423 standard as an adjunct to the RS-422 differential signal standard. By 1978, receiver circuits had become all integrated circuits and 12 to 24 volt power was no longer available in many of the newer products. The RS-423 standard specified a ± 5 volt bi-polar signal swing vs the ± 25 volt swing in the older RS-232 Standard. RS-423 signals were intended to be used as the control and status signals for RS-422 data systems using the RS-449 Connector Standard. The RS-449 Connector Standard never became popular due to its awkward size (37 pin DC shell connector) and large number of signals.

RS-499 STANDARD

The EIA created the RS-499 Standard as a connector for a complex serial interface that utilized RS-422 and RS-423 signals. The RS-499 Serial Interface provided for many more circuits than used in today's systems. The RS-449 Connector Standard never became popular due to its awkward size (37 pin DC shell connector) and large number of signals.

RS-530 PINOUTS

In 1987, the EIA released a new standard, RS-530, for 25 pin connectors that combines RS-232 single-ended signals and RS-422/RS-485 differential signals on a single 25 pin connector. When the device is set to its RS-232 mode, its single-ended signals use the standard RS-232 pins. When the device is set to the RS-485 mode, its differential signals use the new RS-530 pin assignments. An example RS-530 device is ICS Electronics's Model 2363 Serial-to-Digital Interface that has both RS-232 and RS-485 signals on a 25 pin connector.

CHAPTER 8 - TROUBLESHOOTING

The following hints will help you track down serial problems. Most RS-232 problems are quickly solved by inserting an RS-232 Troubleshooter in the circuit. The RS-232 Troubleshooter is a low cost module with red-green LEDs that instantly shows you the state of the RS-232 signals and whether you have two transmit signals connected together. A Green LED on identifies a minus voltage such as Transmit Data held in its Mark state or an off Control Line. A red LED on identifies a positive voltage such as an on Control Line or a data line transmitting logic '0' bits. See <http://www.icsdatacom.com> for more information about RS-232 Troubleshooters.

COMMON PROBLEMS

Things to look for are:

1. Correct pin numbers - Verify transmit and receive data direction and pin numbers. Typically these are pins 2 and 3. DTE devices mate directly with DCE devices while DTE to DTE signal connections need to be crossed as shown in Figures 4 and 5.

If you are not sure which pin is Transmit Data, you can use a DVM or an RS-232 Troubleshooter to locate it. Measure from ground (pin 7 on 25-pin connectors, pin 5 on 9-pin connectors) to pins 2 and 3 while the device is powered on. A reading of -6 to -12 volts indicates the data transmit pin. A reading of 0 ± 2 indicates the Receive Data input line. Check both serial devices to be sure that the Transmit Data of one device does go to the Receive Data line on the other device. The RS-232 Troubleshooter shows crossed data lines as one green LED on and the other data line as both LEDs off.

For RS-422 signals, use the DVM to find the positive and negative transmit pins by measuring between the two signals. If the transmitter is enabled, the pin with the negative voltage is the 'A' or '-' lead.

2. Needed control lines - Some devices need signal inputs on their control lines before they can send or receive while other devices have internal pullup resistors and can function with open inputs. If in doubt, add jumpers from a known 'on' signal such as the device's DTR or DSR output signal to the open input signal. For a 25 pin connector jumper pins 4 to 5 and pin 20 to pins 6 and 8 as shown in Figure 5.

3. Same baud rates - Both devices need to be set to the same baud rate. Different baud rates result in garbled data as shown below.

i.e. *!1- *|

Check the baud rate settings on both devices and retest.

4. Same character formats - It may be obvious but often the character formats and parity settings may not be set correctly. Both devices need the same character format. The symptom of an incorrect parity setting is half good-half bad characters. Check the characters with a numeric or alphabetical sequence like 1 2 3 4 or A B C D.

i.e. The sequence '12345' appears as '12*4*' where characters '1', '2', '4' are good and characters '3' and '5' are bad.

Using one or two stop bits is normally not a problem in receiving data. However some devices check for the correct number of stop bits and will not receive the character if it doesn't match the expected setting. When in doubt, set both devices to the same number of stop bits.

5. Receiving bad data between messages - In RS-422 and RS-485 systems, there may be time between data transmissions when no device is transmitting and the signal lines are allowed to float. Some receivers may see the floating lines as noise inputs or as additional characters. This problem often shows up as garbled characters at the start of a message with the remainder of the message being good.

The solution is to add a termination network to the data lines. The recommended location is to put the termination network at the last device on the data line. Long lines (> 1,000 feet) may need two termination networks, one at each end of the line. The general method is to use a 1 to 2 kohm pullup resistor from the +TX/+RX line to the plus supply voltage (VCC) and another 1 to 2 kohm pulldown resistor from the -TX/-RX line to GND. The two resistors will bias the system to a mark state when no transmitters are enabled. Place a 120 ohm or similar value load resistor across the two lines unless there is a load resistor in the adjacent serial device.

6. Receiving ones own messages - Could be caused by the receiving device set to echo back any message that it receives. Check the receiving device's settings and turn off the echo function. In half-duplex RS-422 or RS-485 systems, the receiver may listen to its own transmitted message as well as messages from the other devices. Change the device's program to discard its own transmitted message.

7. Bad signal levels - Could be caused by multiple load resistors on the network. This is a network issue where multiple serial devices may all have a load resistors. Remove all but except the load resistors at each end of the network.

ADVANCED TROUBLESHOOTING

Really difficult problems can often be solved by spying on the serial transmission line. Rig a spare CRT terminal or PC running Hyperterminal as a test monitor. Connect its receiver input to the transmit line from the device that sends first. Do not connect the monitor's transmit line to anything. Verify that your monitor shows the serial message that you are expecting to transmit. If not, fix it. Then connect your monitor's receiver input to the receive line. See if it shows the data you expected. If not fix it.

SUMMARY

This Application Note has provided an overview of asynchronous serial communication that is common between PCs and other serial devices. This note covers RS-232 and RS-422/RS-485 systems and includes troubleshooting techniques for solving the more common serial communication problems.