HARDWARE HACKER

Sources of information on wire, the SCSI Standard, testing and interfacing of RS-232C serial data

By Don Lancaster

The main feature of this month's column discusses what anyone interested in computers and digital interfacing should know about the so-called RS-232C serial interface standards. But first a timely word about where you can get information on wire.

Where can I get tech details on wire and cable?

Check out the new *Master Catalog 885* by *Belden*. This beauty has a technical information section that gives you lots of details on solid and stranded magnet wire, details on all the major coaxial cable types, and charts of cable current capacities. Best of all is a list of some 54 ap notes and tech bulletins available free.

Of particular interest to *Modern Electronics* hardware hackers should be its bulletins on soldering (T/8-13), coax selection (T/8-24), and hi-fi cables (T/8-31). As always, you should make letterhead requests or direct telephone inquiries.

Speaking of wire in general, one of the best sources I have found for breadboarding jumpers is the *Squires Electronics* BK-1 wire kit. It's a steal at $9.95. Squires also sells such things as shrink tubing and precut wirewrap wire.

What is an SCSI interface?

SCSI stands for "Small Computer Standard Interface." This is an emerging standard way of connecting hard disks and streaming tape backup systems to personal computers. Since the SCSI or "scuzzy," interface is very good at rapidly transferring large amounts of data, you'll shortly see its use for such things as laser printers, image processing, document scanners, CD disks, and graphics workstations.

Much of the new interest in this standard centers on the new SCSI interface in the Macintosh Plus by *Apple Computer*. SCSI interfaces for the Apple IIe and IIc should also be available shortly.

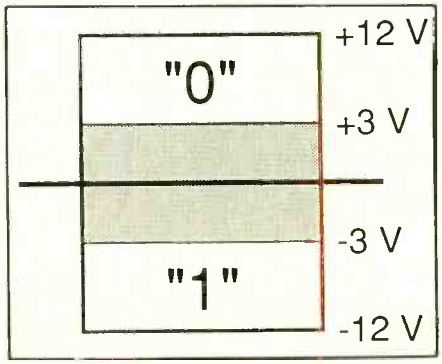

Fig. 1. The RS-232C signal levels.

The SCSI interface is fully documented in ANSC X3T9.2 SCSI Specification, available for $20 from *CBEMA*.

For magic chips that make SCSI go, check into the WD33C92/3 from *Western Digital*. WD also has lots of data sheets, ap notes, and listings available.

What is an RS-232C interface?

Who in their right mind would ever interconnect computers and peripherals with an interface that (a) uses weird supply voltages; (b) requires cumbersome and expensive connectors of too many pins; (c) is error prone; (d) works only over limited distances, (e) provides zero safety or current loop isolation; (f) is lousy at networking; and (g) is both very difficult to use and complex to debug? The answer, unfortunately, is just about everybody who uses a computer.

RS-232C is far and away the most common way of connecting a personal computer to a printer, plotter or modem. The DB-25 connectors used are now seen everywhere. Even older dot-matrix printers are pretty much being forced into serial RS-232C communications.

The "lets go serial" mania began when personal computer manufacturers discovered they could get FCC certification far easier with a stock and slow serial interface than with a fast parallel one.

At any rate, RS-232C is a standard, and incredibly awful, way of connecting computers and peripherals together. As many as 25 lines may be needed for a single serial interface.

Let's look at the key concepts of RS-232C. The data communication is asynchronous, meaning that any amount of time can go by between character or data transmissions. Each character or data word sent consists of a start bit, seven or eight data bits, an optional and seldom-used parity bit, and finally one or two stop bits.

Only seven data bits are needed to send a character, while eight are needed for an 8-bit data byte. Alternately, an 8-bit data byte can be broken up into a hex ASCII *pair* of characters that can be sent as two successive 7-bit characters. The Laserwriter uses this latter route, nicely letting you shove pictures and images through any old word processor. And nastily taking twice as long to send the characters over the interface.

Check back into last month's "Hardware Hacker" to see how these bits are generated and received using hardware or software UARTs.

Character transmission rate is set by the baud rate, which in turn sets the speed at which the bits go flying over the interface. Standard baud rates are 300 or 1200 for modems, 300 or 9600 for printers.

Note that many personal computers cannot scroll their screens faster than 1200 baud. The number of characters per second is roughly equal to one-tenth the baud rate. Thus, 300 baud translates to approximately 30 characters per second.

Figure 1 shows the transmitted voltage levels. A logic zero is anything from +3 to +12 volts. A logic one is anything from −3 to −12 volts. Note that this negative logic is "upside down" from what you might expect. From −3 to +3 volts is usually assumed to be noise.

Figure 2 shows the most important and most-often-used RS-232C pins and what they do.

There are two ground lines, the signal ground on pin 7, and the separate safety ground on pin 1. While the signal ground is always used, it is very important to prevent any ground loop currents or system noise from going over this line.

Pin 2 is the data-out line, while pin 3 is the data-in line, referenced to the host computer.

Now the fun begins. If you are driving a modem, then pin 2 at the host connects

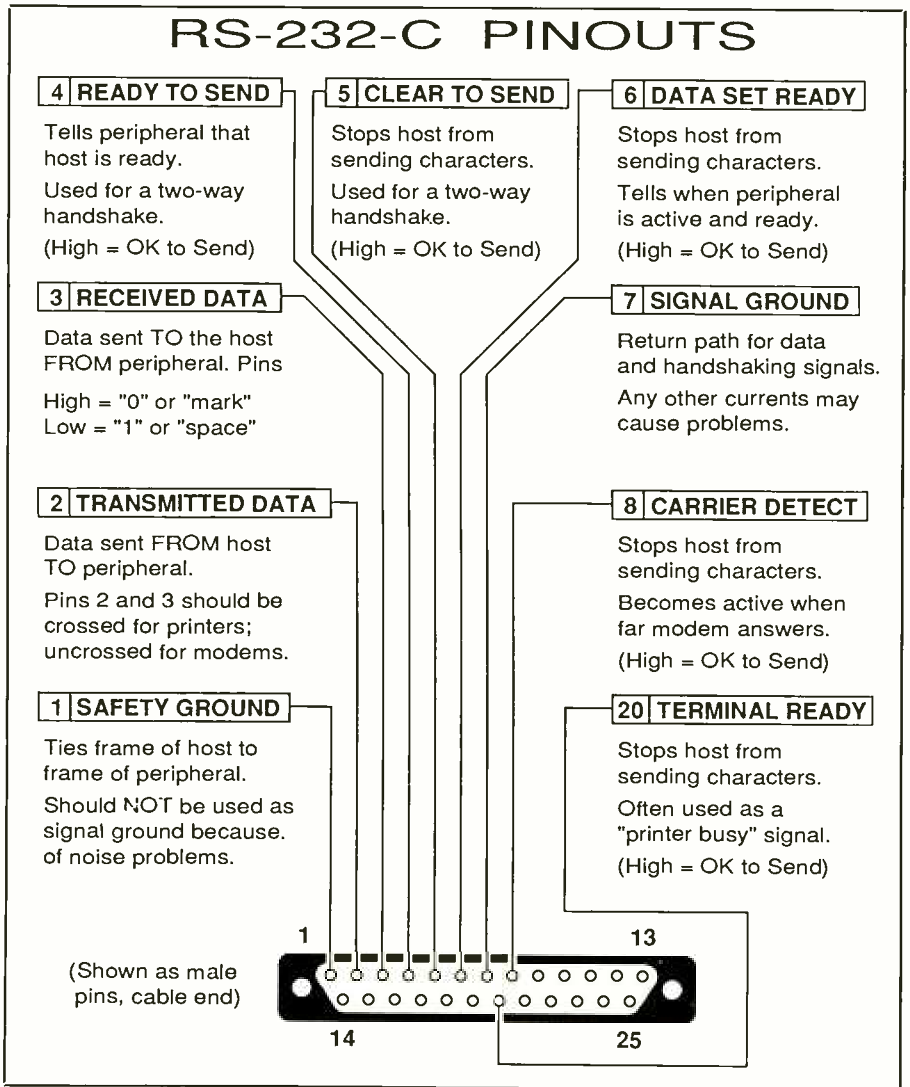

Fig. 2. These are the most-often-used RS-232C connector pinouts.

to pin 2 at the modem. Pin 3 at the host connects to pin 3 at the modem. On the other hand, if you are connecting to a printer or peripheral with some smarts, then pin 2 at the host must connect to pin 3 at the peripheral, and vice versa.

So, sometimes you have to cross the data path to get both ends speaking to each other. Data path crossing can be done inside the cable (watch this detail *very* carefully!) by DIP switches or with port configuration blocks. The rule is: modems do not cross; printers cross once and only once.

Pins 4 (Ready To Send) and 5 (Clear to Send) are not used too often, but they give you a secondary way of handshaking. As before, these pins do not cross with modems and do cross with printers that use these lines.

Pin 6 is an output signal from the host called I/O Ready. This one tells the printer or whatever that the computer is running and ready to either send or receive data. Pin 8 is called the Carrier Detect. It is intended mostly for modem use, whenever the modem wants to tell the computer that it found the necessary tones on the phone line to start communication.

Finally, pin 20 is the very important Data Terminal Ready line, otherwise known as the printer busy line. The printer uses this line to stop characters or data being sent whenever it already has more data than it can use. It is this line that is supposed to stop transmission whenever a printer's buffer is full.

Unfortunately, there is no standard rule of how many pins are used which way. Some printers can live with only four wires, namely signal ground, transmitted data, received data, and printer busy. Other times, all 25 of the pins end up in use doing exotic things.

One baseline rule for printers is to: cross 2 and 3; cross 4 and 5; and jumper 6, 8, and 20 together. But there are many other variations and possibilities.

There's a gadget called a modem eliminator that can plug between two RS-232 cables to let two personal computers directly talk to each other. A null modem often crosses 2 and 3 and jumpers 6, 8, and 20. If the null modem has no handshake, pin 20 on each end goes to its own pin 6 and 8. If the null modem has full handshaking, then pin 20 on each end goes to pins 6 and 8 on the opposite end.

Another gadget is the sniffer. A sniffer may be used to monitor whatever data is passing over the RS-232C interface, without interfering in any way with that data transmission and without any handshaking of any sort. This ties a pair of connectors together with all lines straight through. A separate male connector has only pin 7 tied to the others. Pin 3 of this connector is switchable to either the main pin 2 line or pin 3 line. Often a series resistor, say 3000 ohms, is placed in the sniffer lead to minimize any loading.

What is handshaking?

Handshaking is any technique that makes sure a character or a data value gets used once only. The simplest form of handshaking is to send out characters or data so slowly that they can never be missed. While simple and cheap, this is often very wasteful of both your time and that of your personal computer.

For "real" handshaking, you have the choice of a direct, or hardwire connection, or by sending text control commands over the serial interface.

Personal computers have long favored a direct-wire handshake, often coming from peripheral to the computer by way of RS-232C line 20, which is often used as a "printer busy" signal.

Dino computers and minicomputers have traditionally used text character commands. Note that a hardwire connection is simple and cheap and can immediately stop character transmission. On the other hand, the hardwire busy signal cannot be routed through a modem or a satellite link.

One fairly uncommon text control command handshaking is called ETX/ACK. Whenever an ETX or hex $03 is found in a printer's buffer, an ACK or hex $06 is sent back to the host computer. In effect, the computer says "let me know when you have gotten so far." The peripheral answers later with "I just got there." The computer is "boss" in this transaction.

The most-used text control commands are called XON/XOFF, otherwise known as DC1/DC3. Whenever a printer buffer gets fairly full, an XOFF command, or a hex $11 is sent to the computer. The computer stops sending any characters for a while. Meanwhile, the printer is using up the characters in its internal buffer. When the printer or other peripheral has used up most of the characters left, it sends an XON command, or a hex $14. The character stream begins anew.

With XON/XOFF, the printer says to the computer "Hey, hold up sending me stuff for a while." Later on, the printer says "OK, I can now use more data." The printer is temporarily the "boss" with this type of handshaking.

Note that the printer or whatever never runs out of characters and never has to stop. On the other hand, a fairly large buffer is needed for the "slop" at both ends for continuous character usage.

More and more personal computers are switching to the XON/XOFF route. The Laserwriter laser printer demands this type of handshaking, as do most modems. XON/XOFF is directly available by the Super Serial Card used in an Apple IIe. Older Aple IIc computers lacked a ready-to-go XON/XOFF ability, but this has been cured in the brand new "3.5 ROM" monitor release. See your dealer for the upgrade details.

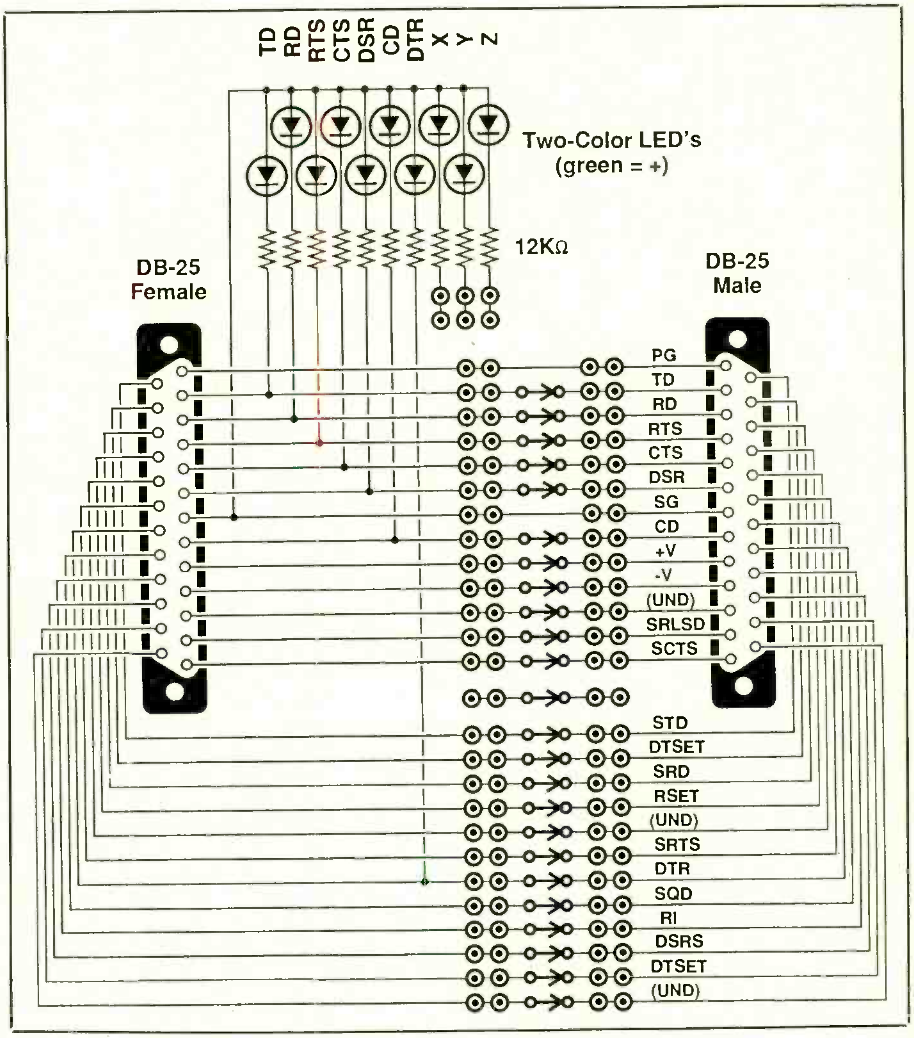

Fig. 3. Schematic diagram of a typical RS-232C breakout box.

How do I service an RS-232 serial data system?

As with most troubleshooting techniques, "divide and conquer" works well here. The best rule is to attack the big lumps first!

First, make sure the cables are in fact connected and routed between the systems in the way they are intended. Then check to be sure that pins 2 and 3 are *not* crossed for modem use, and *are* crossed *once and only once* for printer use. This crossing can be done with programming blocks, slide switches, or physical interchanging of cable pins.

Second, make sure that the data rates and formats are the same at both ends. Very often, a low initial baud rate of 110 or 150 bits per second is a good starting point, for it eliminates subtle handshaking problems from your initial setup. Seven or eight data bits, one or two stop bits, and no parity are often chosen. These are sometimes settable with slide switches, sometimes with system software. Repeating, it is *extremely important* that both ends of the system are running at the same baud rate, have the same parity, and have the same number of data and stop bits.

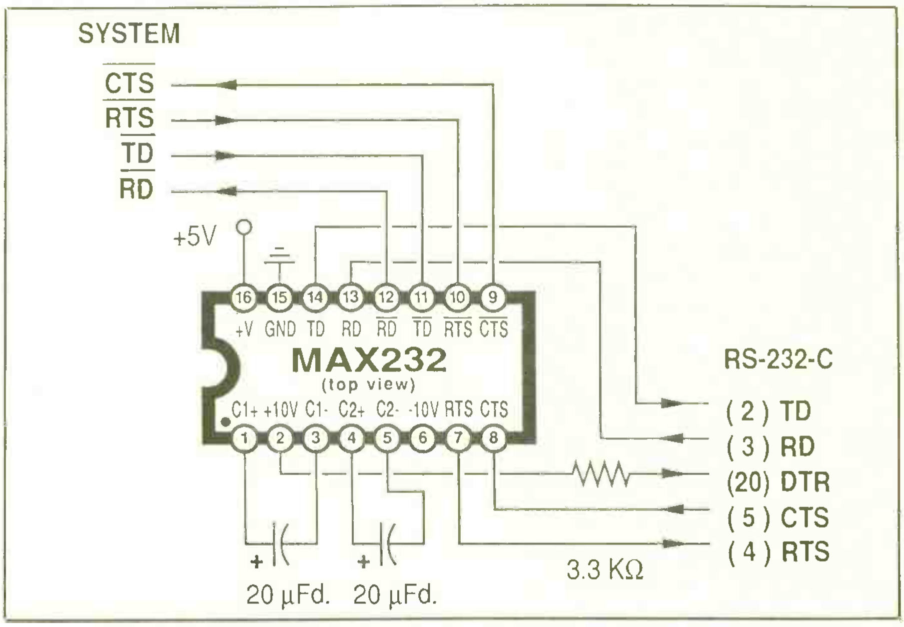

Fig. 4. An RS-232C interface that needs only a single 5-volt supply.

Do not go beyond step two until such time as the printer or whatever is trying to do something. If you get incorrect characters, very often a study of an ASCII character table can tell you what is really happening. Pay attention to any pins that could hold up sending of the characters.

Third, get something sent over the interface so you can handle the more subtle problems. If the printer is running over the same line or is double spacing, then a "CR after LF" switch or two is set wrong. If the printer insists on only printing, say 80 columns, when it is a wider printer, chances are an interface card or serial interface circuit is throwing in its own carriage returns where it thinks the line should end. In the Apple world, a "[I] 80 N" command will often set the interface to an 80-column line, while a "[I] 255 N" will often set the interface so wide that only the output text or the printer will force its own carriage returns.

At this point, you should get at least a paragraph or so of correct text or data on each try.

Fourth, attack the handshaking. Speed up the baud rate to whatever you really want, remembering that modems often have maximum limits of 300 or

NAMES AND NUMBERS

Apple Computer
20525 Mariani Avenue
Cupertino, CA 95014
(408) 966-1010

Black Box Inc.
Box 12800
Pittsburgh, PA 15241
(412) 746-5530

Belden Wire and Cable
Box 1890
Richmond, IN 47375
(317) 983-5200

CBEMA
311 First Street NW S500
Washington, DC 20001
(202) 737-8888

Maxim
510 Pastoria Avenue
Sunnyvale, CA 94086
(408) 737-7600

Motorola Semiconductor
Box 20912
Phoenix, AZ 85036
(602) 244-6900

Squires Electronics
503 North 13th Avenue
Cornelius, OR 97113
(503) 357-7132

Western Digital
2445 McCabe Way
Irvine, CA 92714
(714) 851-1221

1200 baud, and that few computer screens can scroll even at a 1200 baud rate. Very often, 9600 baud is a good choice for a one-way printer trip, if the printer is set up to handle this rate.

One sure sign of handshake problems is when printing is normal for half a page or more and then gets violently ill. What has happened is the printer buffer or whatever overflowed, plowing the works.

Handshaking problems often consist of a jumper or slide switch not connecting the "busy" line from the printer or other peripheral to the "stop sending" line of the interface.

An oscilloscope can be used to solve any RS-232C problem, but it is both expensive and usually not capable of monitoring all interface lines at the same time. Instead, a device called a breakout box is used that can tell you an awful lot in a hurry. The breakout box goes between an RS-232C connector and a cable.

Figure 3 shows the typical features of a breakout box. Two-color light-emitting diodes will tell you if any line is high or low. It is important to limit the LED current to one milliamp or less so it does not swamp and overload any signal line. The extra test points let you connect an oscilloscope or voltmeter onto suspect leads. The jumper blocks let you break the "straight through" connections and cross leads, or do whatever to force a working interface.

One source of breakout boxes, RS-232C cables, and stuff like this is *Black Box Inc.* Their Black Box Catalog also has lots of useful technical information in it.

Show me an RS-232C circuit

You need a line driver to get from regular logic levels to RS-232C. Similarly, you need a line receiver to get from the RS-232C levels back into logic levels compatible with your usual computer circuits.

The usual way to do this was with a pair of old warhorse chips, the *Motorola* MC1488 and MC1489. The 1488 has four drivers, while the 1489 is a quad receiver. Motorola has recently upgraded and improved this product into the MC145406, a single package solution that provides three drivers and three receivers.

Both the old and new versions suffer from a serious flaw. Besides the usual +5 volts, you also need a +12-volt and a −12-volt supply. In today's modern computer circuits, it may turn out that the RS-232C drivers are the *only* thing that needs these oddball supply voltages. Thus, using these chips can add power supply cost and complexity.

Its *Maxim* to the rescue. This company's stunning new MAX232 chip is a pair of RS-232C drivers and a pair of receivers that—get this—works entirely off a single +5-volt supply. To do this neat trick, there is a pair of voltage converter circuits built into the chip. The first converter doubles the +5 to +10 volts; the second one changes the +10 into −10 volts. Figure 4 shows how to connect this exciting new circuit. The cost is under $5 in singles, and you save even more by not having to mess around with oddball power supply voltages.

In Parting

I have just freshly reprinted both my *Micro Cookbook,* Volume 1 and *Enhancing Your Apple II and IIe,* Volume 1, so be sure to write or call if you are interested in picking up either of these.

Remember that this is your column and you can get free technical assistance by writing or calling per the box at the end of this column. At present, I am several hundred letters behind, so you will get the fastest results by calling, rather than writing. ME

NEED HELP?

Phone or write your **Hardware Hacker** questions directly to:

Don Lancaster
Synergetics
Box 809
Thatcher, AZ 85552
(602) 428-4073