Fig 1. Typical RS232 Output Driver.

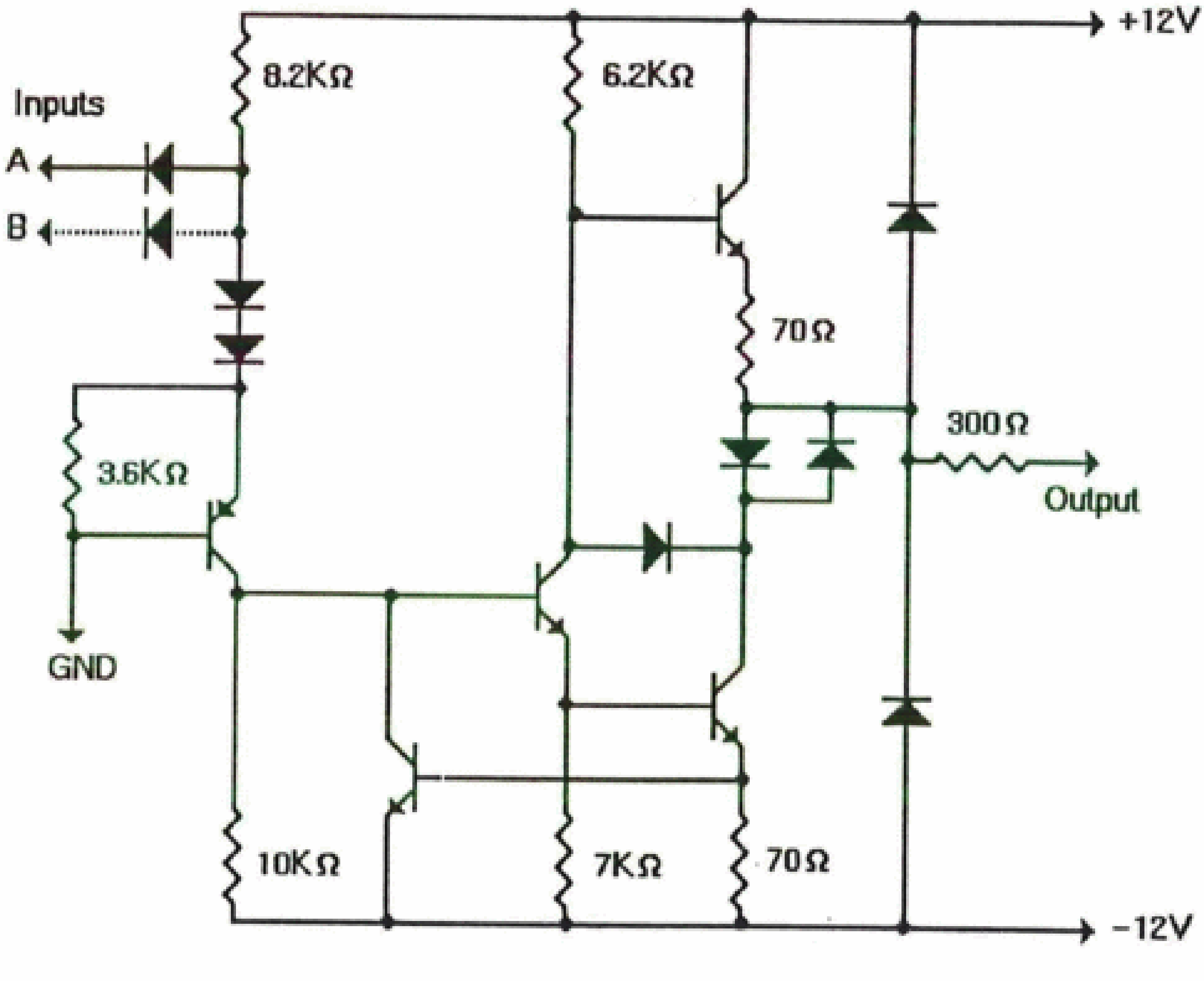
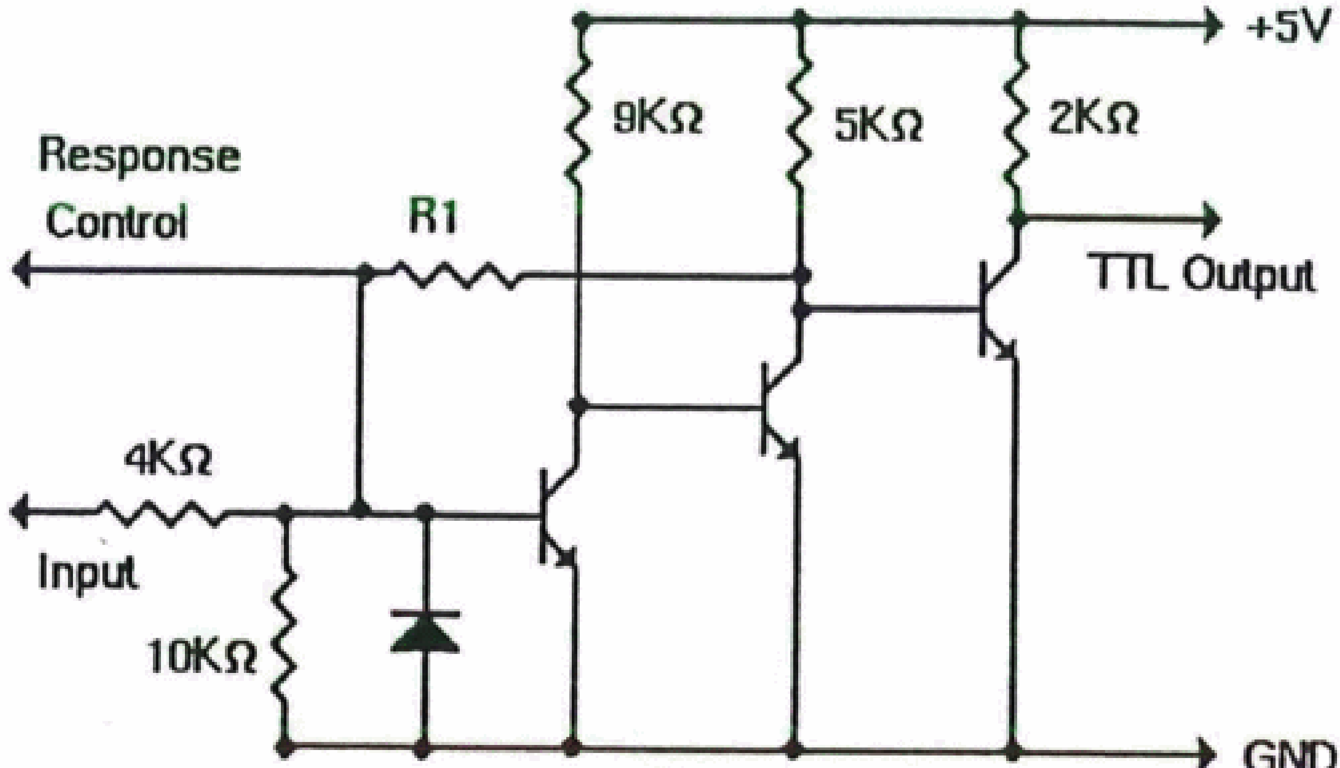
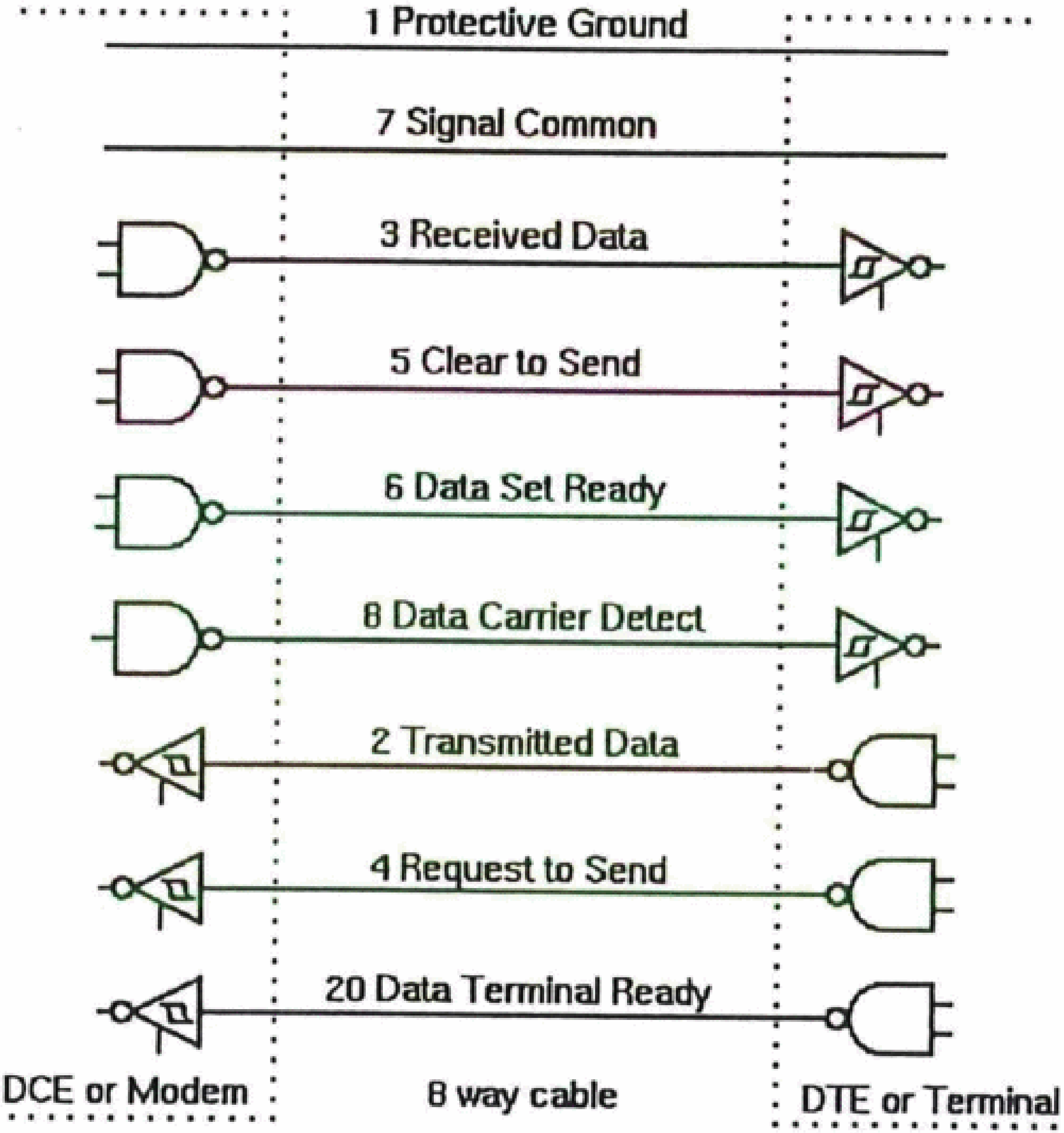


Fig 2. Typical RS232 Line Receiver.



R1 = 10KΩ for 1489, 2KΩ for 1489A

Fig 3. Typical Interconnection



Pin	Symbol	Description	DTE direction DCE
1	PG	Protective Ground/Chassis/Screen	DTE<-------->DCE
7	SG	Signal Ground/common return	DTE<-------->DCE
2	TD	Transmitted Data	-------->DCE
3	RD	Received Data	DTE<--------
4	RTS	Request to Send	-------->DCE
5	CTS	Clear to Send	DTE<--------
6	DSR	Data Set Ready	DTE<--------
20	DTR	Data Terminal Ready	-------->DCE
8	DCD	Data Carrier Detect	DTE<--------
22	RI	Ringing Indicator	DTE<--------
21	SQD	Signal Quality Detector	DTE<--------
23	CH	DTE Data Signal Rate Selector	-------->DCE
23	CI	DCE Data Signal Rate Detector	DTE<--------
15	TxCk	DCE Transmitter Clock	DTE<--------
17	RxCk	DCE Receiver Clock	DTE<--------
24	TxCk	DTE Transmitter Clock	-------->DCE
14	STD	Secondary Transmitted Data	-------->DCE
16	SRD	Secondary Received Data	DTE<--------
19	SRTS	Secondary Request to Send	-------->DCE
13	SCTS	Secondary Clear to Send	DTE<--------
12	SDCD	Secondary Data Carrier Detect	DTE<--------
9		Reserved for Data Set Testing	--------
10		Reserved for Data Set Testing	--------
11		Unassigned sometimes Busy/Status	-------->DCE
18		Unassigned	--------
25		Unassigned	--------

Fig.4. RS-232C Pin Assignments showing signal directions.

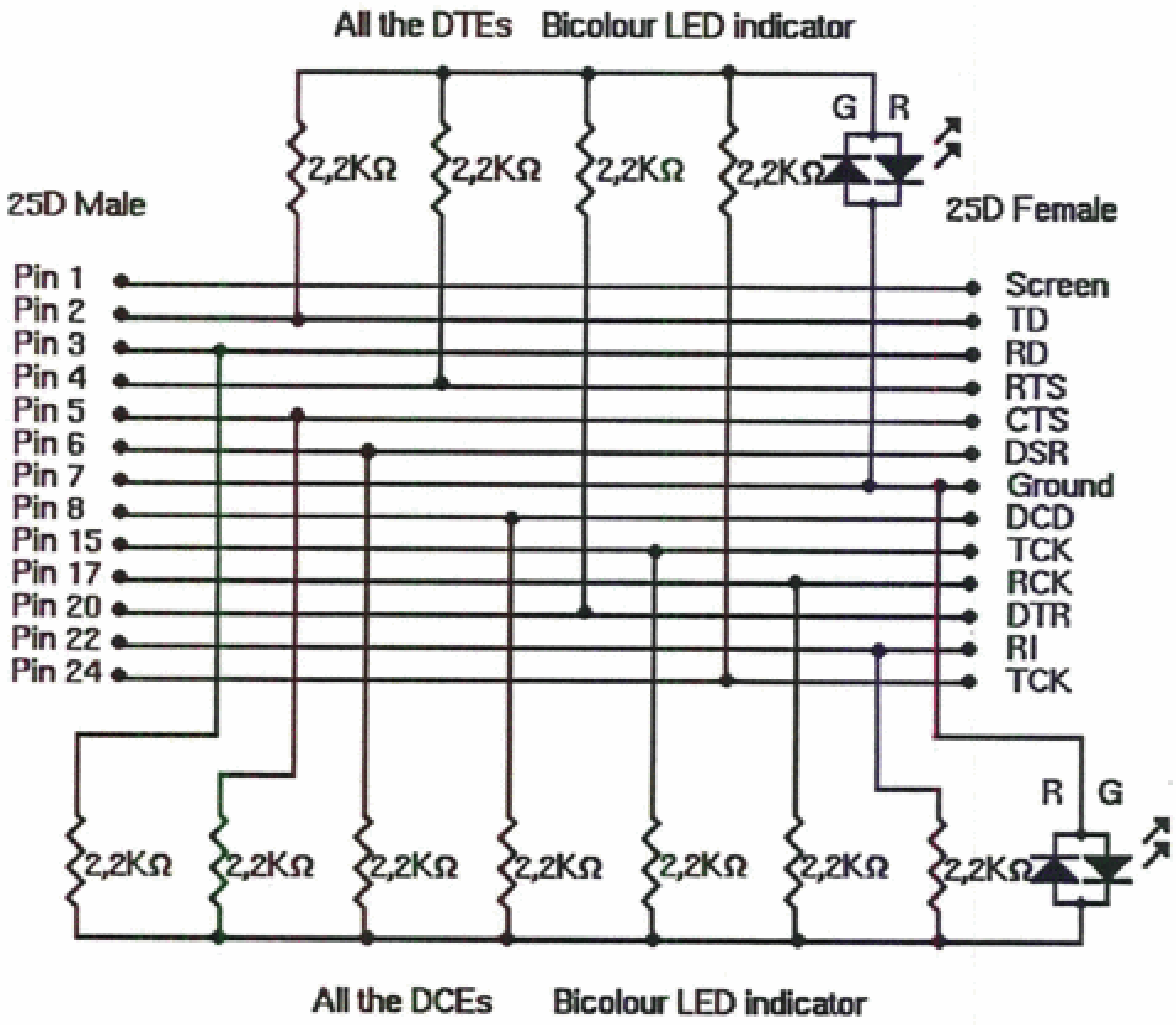


Fig 6. Diagnostic Tool for all normal RS232 signal lines.

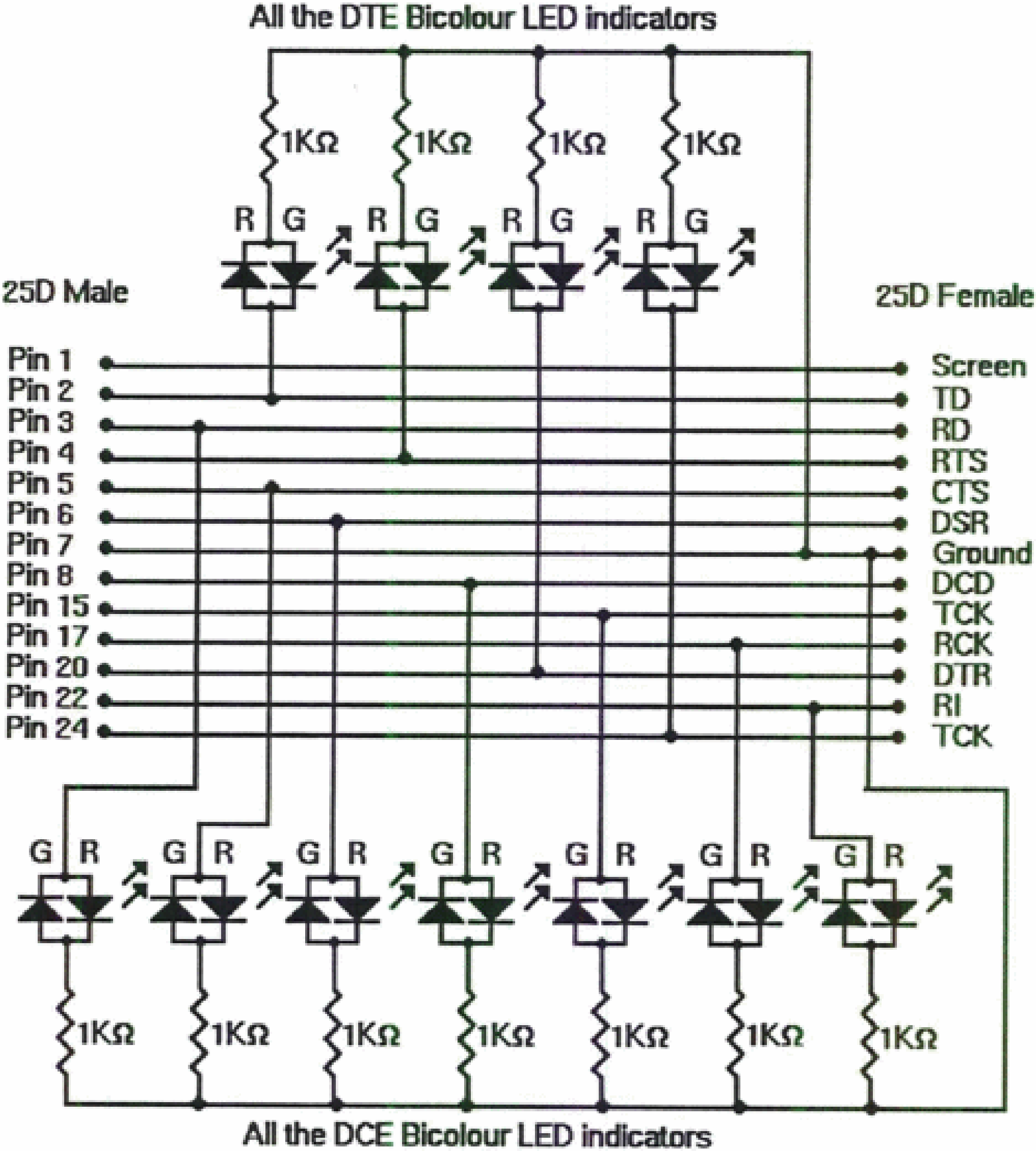


Fig 7. Diagnostic Tool for all normal RS232 signal lines.

RS232RS.WS4   (= RS-232C article by Richard Stevens)

- "RS232 Really Made Simple"
  Richard Stevens, 14 May 1987

(Retyped by Emmanuel ROCHE.)

There are a large number of wealthy people who got that way by writing books about RS232, and selling a vast assortment of RS232 gadgets. All of these are claimed to be "marvellous", "clever", "indispensable", and so on. It seems that the list is limited only by the advertising copywriters imagination. Most of these euphemisms describe what is usually no more than a connector or two, and a few simple components. There are a few notable and valuable exceptions, but their rarity is highlighted by the piles of overpriced junk, which is normally used only the once!

What is RS232?

The name refers to an American invention, which evolved to a state where it was adopted as the "Electronics Industries Association RS-232-C Standard" (Ref. 1).

Alas, it has continued to evolve and grow in complexity. as everyone tried to make it the universal system for serial data communication. Names and numbers have been altered, and so have the protocols, but the manufacturers keep coming down to the same basic simple configuration in the vast majority of microcomputers.

In general, there are a few aspects that are common to all implementations:

a) the data is transmitted serially
b) there is a form of control such as "hadshaking"
c) the signals have a certain power level
d) there are male and female plugs and/or sockets in the way
e) there is a speed limit
f) it never works as supplied (first corollary of Murphy's law)
g) it always works after it has been blessed.

This may seem to be a somewhat bald summary, but, when one considers the variety of schemes chosen, in order to make one buy accessories for Fruity-Super-Whizzos, obtainable only from the manufacturer, it is amazing how they manage it! Nevertheless, given the knowledge of what is happening down there in the copper and gold flashing, anyone can cut out the sharks and do their own thing. Only when time is lots of money, should discretion be the better course.

The electrical bit

For those who like to know what is really going on inside, the basic signal MUST be compatible with what follows.

Sending

a) Usually, it comes out of a device named 1488 (or 75188).

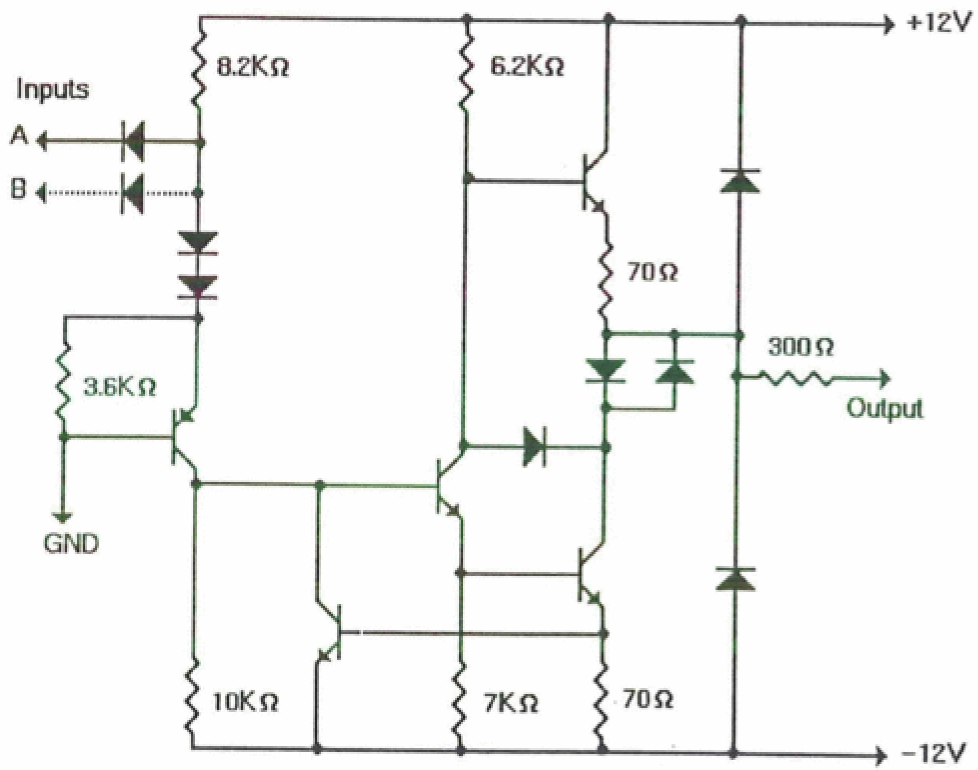
b) It is Bipolar, i.e., ordinary hot transistors in a spider.

c) The source impedance is a 300-Ohm resistor in the chip, with up to 2200 pF of external capacitance to ground to slow it down AND kill noise.

d) The voltage swing is usually between +10 Volts and -10 Volts, coming from + and -12 Volt supplies, and the current is usually limited to 10-mA by the chip (References 4 and 5).

See Figure 1 for a detail of one output of the 1488.

Fig 1. Typical RS232 Output Driver.



Receiving
----

a) The destination is usually a chip called 1489 (or 75189).

b) Again, it is Bipolar.

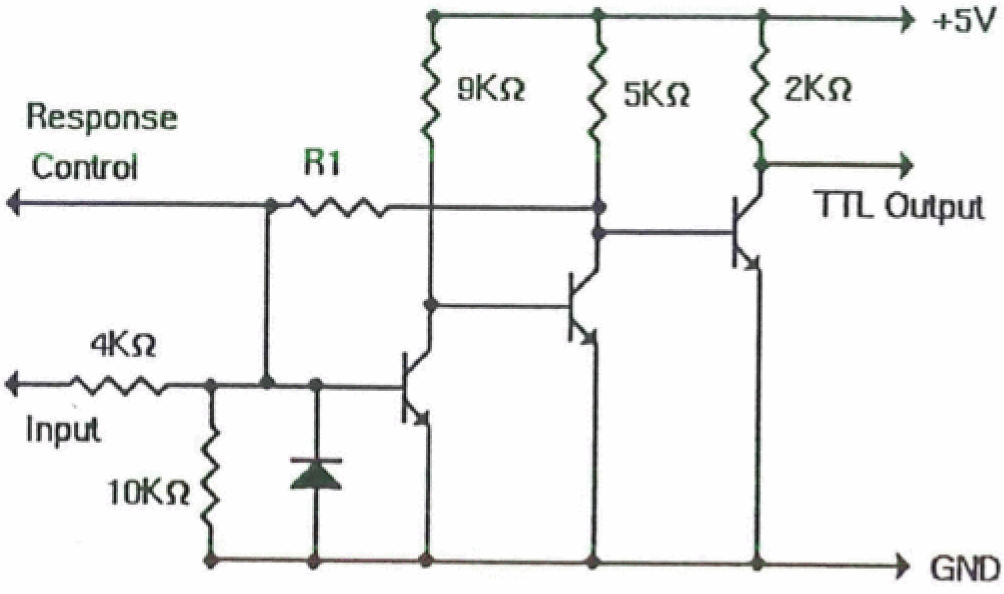
c) The input load impedance is usually more than 3K Ohms.

d) The signal level must be greater than +2 Volts when on a positive transition, and less than +1 Volt for a negative transition, while not exceeding +25 or -25 Volts!

e) There is also a pin connection for a small filter capacitor (often 330 pF), to limit the speed and kill noise. Here also, can be connected a resistor to alter the input thresholds.

See Figure 2 for a detail of one input of the 1489.

Fig 2. Typical RS232 Line Receiver.



R1 = 10KΩ for 1489, 2KΩ for 1489A

In between
----

a) For short runs (5 feet, say) use any screened multicore cable having enough wires!

b) For long runs (over 20 feet, say) use one twisted pair for each signal, with the returns all connected to signal ground, and an overall screen. Expensive, but worthwhile.

c) A number of 25-pin DB miniature Dee connectors.

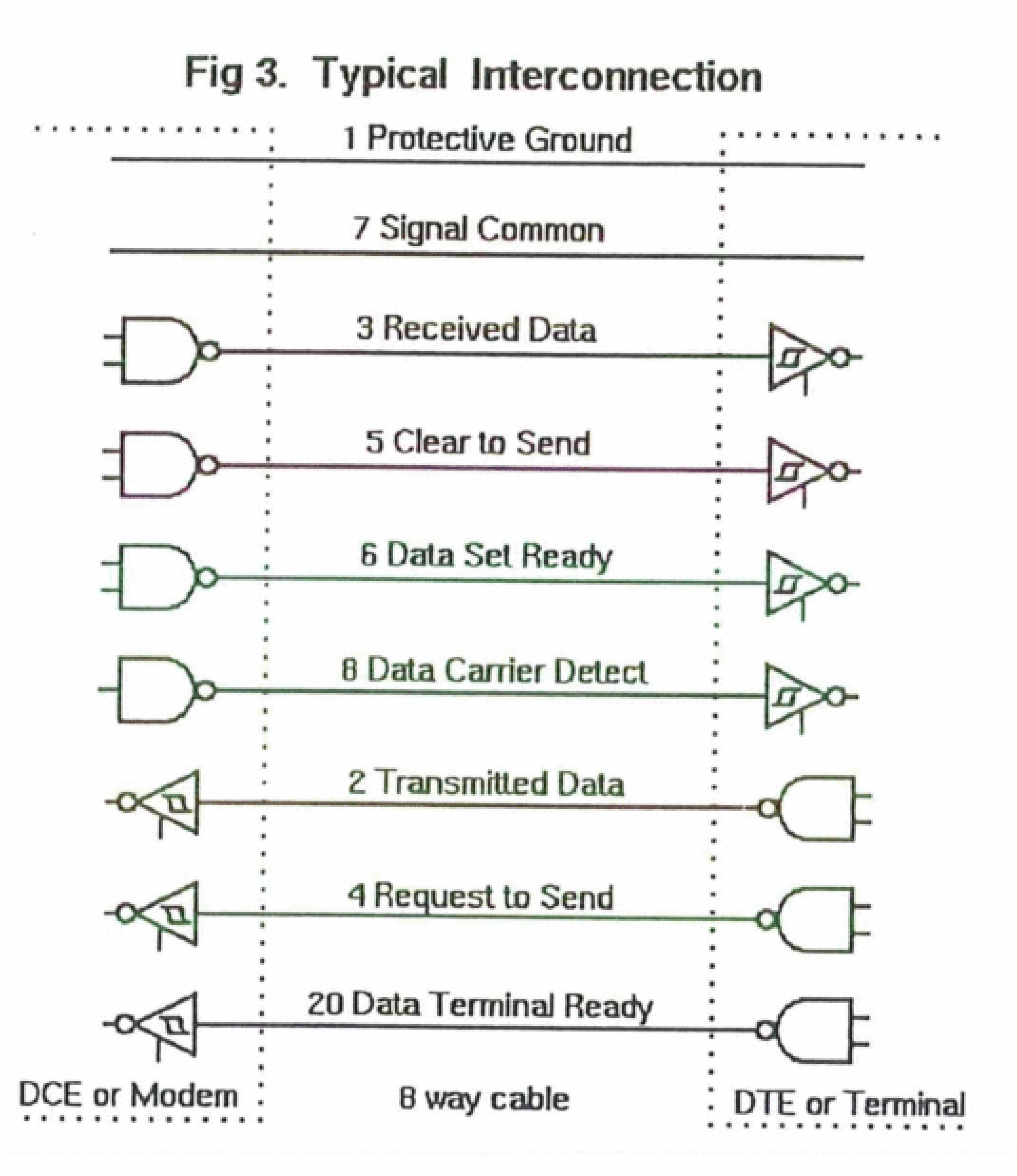
Signal sense
----

a) All the signals are "True" when positive (high), and used but unconnected inputs are usually tied to an output that is always going to stay high. This keeps the UART happy. (what's a UART? See below.)

b) There is a speed limit of 20,000 bits per second, or an upper frequency limit of 10-KHz. It is usually 19,600 bits/sec for practical reasons (the UART, again). The filter capacitors are chosen with the speed limit in mind, and are usually near the values mentioned above. These depend on the trade-off between noise rejection, speed, and attenuation because of cable length.

c) When in Synchronous mode, it is possible to go to 76KBaudwith a USART, but with no filters, and shorter runs.

See Figure 3 for a typical connection scheme of the DTE output to the DCE. The return signals DCE to DTE will require another pair of chips and four more conductors in the cable.



So, how does it work?

The simplest system

This is just a one-way signal, down a single pair of wires, or perhaps a Coaxial cable. A keyboard needs only this. Usually, however, there is power coming back from the terminal or computer, and a small cheap connector, and suddenly it does not look like RS232. But the signal usually still is!

How can the information be sorted out, especially if it is just now and then? The secret lies in the way we specify the form of the data. First, it is always in a packet of eight bits called a BYTE (hence the magazine). Eight bits allow 256 different codes which amply covers the alphabet and numbers, also a lot of other useful things!

Computers are stupid, and will believe anything you tell them. If noise changes even one bit of the data, the computer will act on the error. So as to prevent this, there is usually a PARITY bit, or two, which allows the computer to check the data.

This leads to the terms ODD and EVEN PARITY. Odd parity exists when the number of high bits odd is kept odd by making the parity bit high or low to suit. When two parity bits are used, the byte is usually split, so the parity is checked in different ways. Error control is a deep subject and, if you want to know more, read J.R. Watkinsons "Wireless World" articles on Disc systems with special reference to the error checking! (References 6 and 7)

In continuous data streams, the unit could lose track of the start of a byte if a noise spike got in. So, start and stop bits are added to the byte and parity. Start bit is normally high, and stop bit is normally low. We now have a distinct word, in a rigid format that even a computer cannot ognore. However, the eight bit word has grown by up to 50% (2 parity plus start and stop bits!).

This means that, even if one is sending data at 2400 Baud, the actual Byte rate may be as slow as 200 bytes per second, and that is ignoring CRC checks and block markers.

Now a bit more complex

Two-way communication is built into the system as standard. Two pairs of wires, one headed in each direction, would suffice if that was all that was wanted. Most computers and terminals have better things to do than just talk to VDU (Visual Dislay Unit) and listen to Keyboard. In fact, they cannot even do both of those together, that needs multi-tasking!! Normally, a buffer is emptied out to the VDU, and the keyboard infrequently interrupts proceedings, and fills its own buffer (in a good system). All the rest of the time, the CPU (Central Processing Unit) is busy earning its keep, or playing games.

So, how to arbitrate? RS232 has the answer. Two more signals are used to allow handshaking to occur. These are called RTS (**R**EQUEST **T**O **S**END) and CTS (**C**LEAR **T**O **S**END), and are paired up with the two data streams.

When a Terminal (DTE, or **D**ATA **T**ERMINAL **E**QUIPMENT) sees a high signal on its CTS control line, it will go ahead and send the data down the line TD (**T**RANSMITTED **D**ATA). When CTS goes low, the DTE will stop until allowed to continue.

When a Modem or DCE (**D**ATA **C**OMMUNICATION **E**QUIPMENT) sees an RTS on the control line from the Terminal (DTE), it will send to the Terminal the information it has collected, i.e., the **R**ECEIVED **D**ATA (RD).

We now have a simple system where the two equipments can talk to each other, if care is taken that they will not both try and talk at once. This can be handled in the software, and is a common setup. It does have some severe limitations, however. For example, what if one of the devices (a DCE or Modem?), which may be talking elsewhere and relaying information back to the DTE, loses the outside line, and then just sits there and sulks? It knows that there is more to come, and holds on to the DTE, refusing to release it for other tasks. Now, the software could check this with a timer. But, if it times out too soon, and the line is restored after the terminal gives up, this could be embarassing. So, more signals are used!

When the Modem (as DCE) realises that someone is ringing up, it pulls up a line called RI (**R**ING **I**NDICATOR), informing the computer that the phone is ringing. Nothing happens until the computer (DTE) replies with the DTR (**D**ATA **T**ERMINAL **R**EADY) line. The Modem then takes the call and, when the link is established, sets DCD (**D**ATA **C**ARRIER **D**ETECTOR) high. Then, the DCE checks the other end of the link. It looks for the far end DTR and, if OK, acknowledges this with DSR (**D**ATA **S**ET **R**EADY). Then, all the other signals can operate as before. If the line is lost, the Modem pulls down DCD, so the terminal knows the problem and can follow appropriate instructions already in the programme.

Naturally, it is not that simple, since there are yet more signals available. Ther is worse to come, as some of these are a duplicate set of most of the signals already mentioned, with similar names and the prefix "Secondary". These are used in "Four Wire" and "Full Duplex" operation, and the General Post Office rubs hands in glee.

As a final wrinkle, there is a feature not yet mentioned. So far, the system described has been operating in the Asynchronous mode. It is possible to have higher speeds with less error if timing signals are supplied to go with the data and, of course, there are three of them, plus controls.

This still leaves a few spare pins on the "RS-232-C Standard DB-25 Connector". So, naturally, they are filled in with such exotic functions as Signal Quality Detector, etc. Confused? Don't worry. For nearly all requirements, only eight of the 25 pins on the connector are used. Which is why so many "manufacturers" don't use an expensive 25-pin connector, when they can charge the same for a cheaper 9-in one, or even a 6-pin DIN!

Fortunately, in most cases, the only lines used are the TD, RD, CTS, and RTS.

The other outputs, although being active (favourites: DTR, DSR), are just serving as pull-ups to indicate the equipment is switched on! Sometimes, the DCD is used as a BUSY indicator, but more often the BUSY/STATUS output from a DTE-connected printer is on pin 11.

See Figure 4 for a table of the pin assignments.

Pin	Symbol	Description	DTE direction DCE
1	PG	Protective Ground/Chassis/Screen	DTE <--------> DCE
7	SG	Signal Ground/common return	DTE <--------> DCE
2	TD	Transmitted Data	--------> DCE
3	RD	Received Data	DTE <--------
4	RTS	Request To Send	--------> DCE
5	CTS	Clear To Send	DTE <--------
6	DSR	Data Set Ready	DTE <--------
20	DTR	Data Terminal Ready	--------> DCE
8	DCD	Data Carrier Detect	DTE <--------
22	RI	Ring Indicator	DTE <--------
21	SQD	Signal Quality Detector	DTE <--------
23	CH	DTE Data Signal Rate Selector	--------> DCE
23	CI	DCE Data Signal Rate Detector	DTE <--------
15	TxCk	DCE Transmitter Clock	DTE <--------
17	RxCk	DCE Receiver Clock	DTE <--------
24	TxCk	DTE Transmitter Clock	--------> DCE
14	STD	Secondary Transmitted Data	--------> DCE
16	SRD	Secondary Received Data	DTE <--------
19	SRTS	Secondary Request To Send	--------> DCE
13	SCTS	Secondary Clear To Send	DTE <--------
12	SDCD	Secondary Data Carrier Detect	DTE <--------
9		Reserved for Data Set Testing	--------
10		Reserved for Data Set Testing	--------
11		Unassigned, sometimes Busy/Status	--------> DCE
18		Unassigned	--------
25		Unassigned	--------

Figure 4. RS-232C pin assignments showing signal directions

UART, USART, you must have one!

So, what is a UART, or a USART for that matter? The UART stands for Universal Asynchronous Receiver Transmitter, and the S in USART means it can also handle Synchronous transfer of data. There are a variety of these devices, but they all perform the same basic functions:

a) convert the incoming serial data into 8-bit parallel

b) the converse when transmitting

c) check parity as needed

d) check and operate the control lines under guidance from the processor

e) they need a reference clock, othen 16 times the bit rate, for reading or sending the serial data, and this sets the limit for asynchronous operation

f) some units provide their own programmable clock generator

g) with an external synchronous clock, the USART is often capable of handling data speeds up to 200,000 bits per second, as used in the RS-422 standard.

Is this what happens in real life?

The above is all very well in theory. If one has what is often called a (dumb) Terminal, it would expect to be wired as a DTE. So, then, when the terminal is talking directly to a computer, the computer should be wired as a DCE. What then happens if the computer (DCE) talks directly to a modem (DCE) as well?

Also, printers are usually DTE, since historically they often sat on a Modem. Bear in mind that most computer manufacturers make their own serial ports identical to enable "better interchangeability".

The chances are (especially considering Murphy's law) that the units at both ends of the cable are wired identically, so that an output is feeding up another output, and the inputs are friutlessly searching other inputs. Very nasty, but not catastrophic since there is protection built in. To the rescue comes a much maligned/vaunted device called the "Null-Modem". It is a bit of cable with the wires swopped over. Much money has been made, and time wasted, in the pursuit of Null-Modems. The basic Null-Modem cable has, as cross-wired pairs, 2 and 3, 4 and 5, 20 and 6, and a strap to 8 from an adjacent 6 or 5. But see also Figure 5 for typical arrangements that will work on most systems that are properly implemented.

```
  Pin  DTE  direction  DCE   Pin
 ┌────┬──────────────────────┬─────┐
 │  1 │ DTE <----PG---> DCE  │  1  │
 │  7 │ DTE <----SG---> DCE  │  7  │
 ├────┼──────────────────────┼─────┤
 │  2 │       ----TD---> DCE │  2  │
 │  3 │ DTE <----RD---       │  3  │
 │  4 │       ---RTS---> DCE │  4  │
 │  5 │ DTE <---CTS---       │  5  │
 │  6 │ DTE <---DSR---       │  6  │
 │ 20 │       ---DTR---> DCE │ 20  │
 │  8 │ DTE <---DCD---       │  8  │
 │ 22 │ DTE <----RI---       │ 22  │
 ├────┼──────────────────────┼─────┤
 │ 17 │ DTE <---RxC---       │ 17  │
 │ 24 │       ---TxC---> DCE │ 24  │
 └────┴──────────────────────┴─────┘
```

Figure 5A. DTE-DCE 10-way direct

Most systems are not properly implemented. So, we need to know just what has been done. Notice also that, so far, there has been no mention of Sex. Now, it comes. All connectors have male or female gender, some have both, but that's another story. Supposedly, DCE is Female and DTE is Male, however... Most equipment tends to have female sockets on the panel, since this prevents inadvertent shorting of the pins by foreign bodies, reduces bending damage, and also helps to keep out static from exposed contacts. Thus, most, but not all, interconnect cables have a male plug on each end, but differently wired! Extension cables, of course, usually have both a male plug and a female plug, and simply MUST be wired pin to pin. Usually, there are eight cores, wired to pins 2 through 8 and 20, and a screen wired to pin 1.

The answer!

to help sort the confusion, people have such devices as "Breakout Boxes". These allow one to identify a unit as being DCE or DTE, and also which pins are actually carrying signals. Then, with a forest of little wires and plugs, one can link up the appropriate lines in the cables. The snag is that they never work for more than a day at a time, and are amazingly expensive.

A short answer

Figure 6 shows a very simple device to allow one to identify DCE or DTE. Some diodes perform a logical OR on the normal connections from a DTE to a Green LED. A second set allows a Red LED to indicate if the DCE is active.

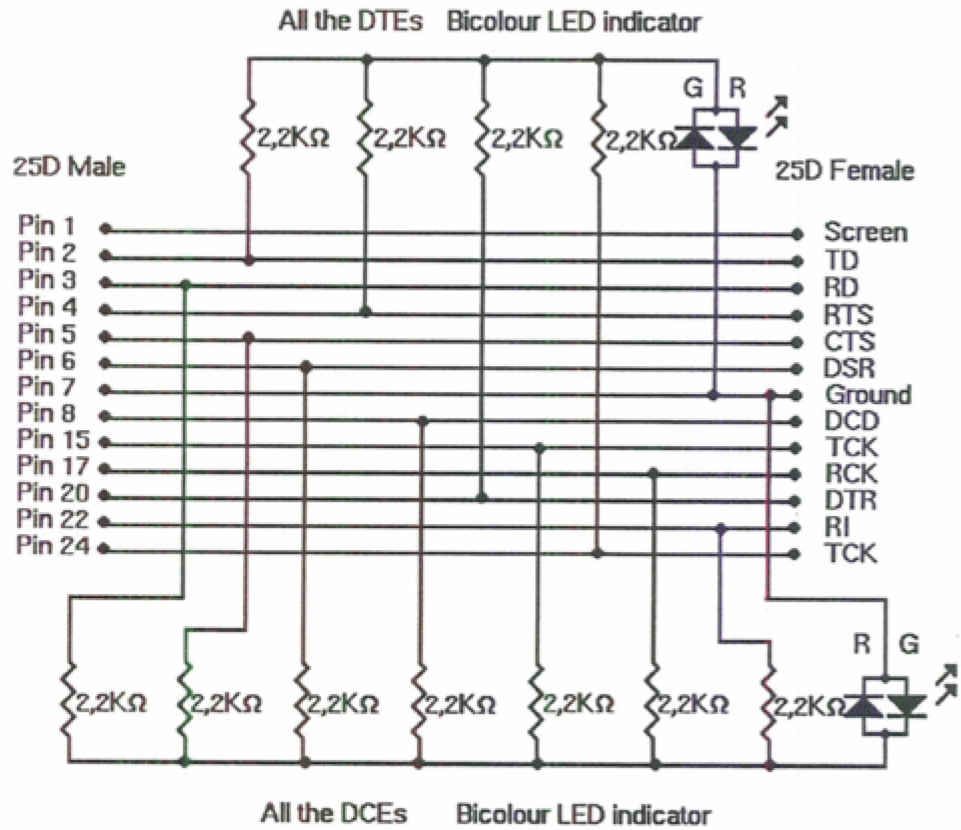


Fig 6. Diagnostic Tool for all normal RS232 signal lines.

A fuller answer

Figure 7 is more complicated but will also tell you which lines are active, if there is enough signal, and if it is truly symmetrical. A Bi-Colour LED is connected to each signal line through a series resistor, so as not to reduce the voltage too much, and will indicate as follows:

- Red if the signal is High,
- Green if Low,
- Yellow if a data stream, and
- OFF if no connection at the sending end.
- Brightness tells an experienced eye the signal strength!

With such a device costing from £10 to £30 in components, one can design the exact null-modem cable one needs in seconds. Wiring, naturally, takes longer, and then it doesn't work!

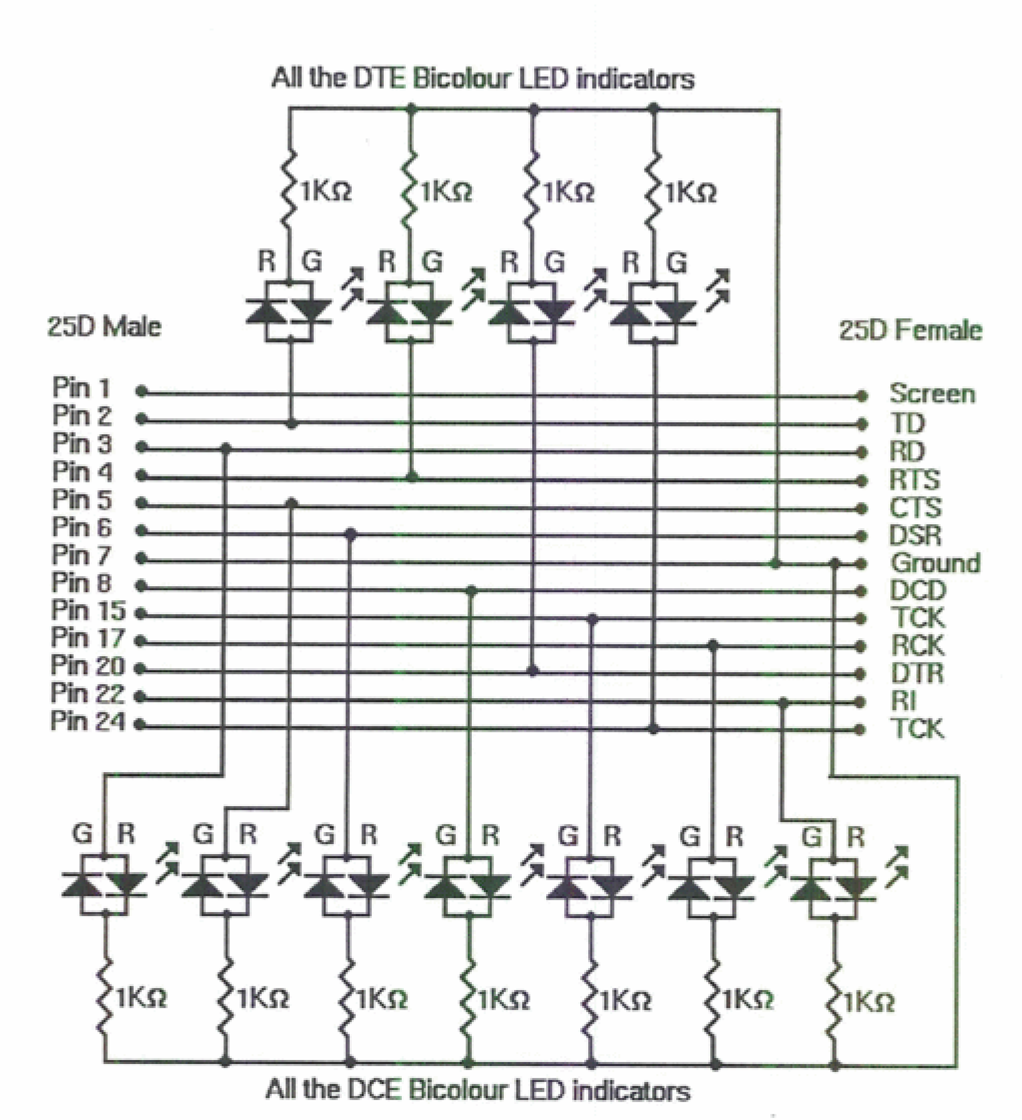


Fig 7. Diagnostic Tool for all normal RS232 signal lines.

Usage of the box is simple

1) Plug it into the computer first, switch on or reset, and see what develops. Make a note of which LEDs lit. If nothing happened, try sending a file to the port in question, at least one LED will light and give the clue as to DCE or DTE connection.

2) Now, plug the box into the "peripheral" instead, and switch it on. At least one light must come on, if not then it is Duff! Peripherals are humble servants, always ready and waiting!

3) Having identified DCE and/or DTE, connect a lead, having all normal connections (pins 1 through 8 and pin 20) as direct or null-modem (Figure 5B), with the box also in circuit. Now, this is the other reason for both sex of connector on the box! Try to send a file to the peripheral, and watch what happens! Make a note of all the lights that show, even if only briefly.

Pin	DTE direction DTE	Pin
1	DTE <----PG---> DTE	1
7	DTE <----SG---> DTE	7
2	----TD---> DTE	3
3	DTE <----RD---	2
4	---RTS---> DTE	5
5	DTE <---CTS---	4

```
      |  6 | DTE <)--DSR---          | 20 |
      |  8 | DTE <)        (> DTE    |  8 |
      | 20 |       ---DTR--(> DTE    |  6 |
      |----+-------------------------+----|
      | 15 | DTE <)--RxC---          | 24 |
      | 17 | DTE <)        (> DTE    | 15 |
      | 24 |       ---TxC--(> DTE    | 24 |
      +----+-------------------------+----+
```

Figure 5B. DTE-DCE 8-way null-modem

4) If this works, try the other direction if possible (not much comes back from a printer). If this also works, nothing more need be done, except remove the box and tidy up! Label Cable!

5) If operation is not satisfactory, try some of the possibilities shown in the rest of Figure 5, bearing in mind the active lines observed so far (See also References 2 and 3).

```
 Pin  DTE  direction  DTE   Pin         Pin  DTE  direction  DCE   Pin
+----+---------------------+----+      +----+---------------------+-----+
|  1 | DTE <----PG---> DTE |  1 |      |  1 | DTE <----PG---> DCE |  1  |
|  7 | DTE <----SG---> DTE |  7 |      |  7 | DTE <----SG---> DCE |  7  |
|----+---------------------+----|      |----+---------------------+-----|
|  2 |     ----TD---> DTE  |  3 |      |  2 | DCE <----TD---      |  3  |
|  3 | DTE <----RD---      |  2 |      |  3 |     ----RD---> DCE  |  2  |
|  4 |     ---RTS---> DTE  |  8 |      |  4 | DCE <---RTS---      |  8  |
|  5 | DTE <)              |    |      |5or6|     ---CTS---> DCE  | 20  |
|  6 | DTE <)--DSR---      | 20 |      | 20 | DTE <---DSR---      |5or6 |
| 20 |     ---DTR--(> DTE  |  6 |      |  8 |     ---DTR---> DCE  |  4  |
|    |             (> DTE  |  5 |      |    |                     |     |
|  8 | DTE <---DCD---      |  4 |      |    |                     |     |
+----+---------------------+----+      +----+---------------------+-----+

Figure 5C. DTE-DTE 6-way null-modem   Figure 5D. DCE-DCE 6-way null-modem


 Pin  DTE  direction  DCE   Pin         Pin  DTE  direction  DCE   Pin
+----+---------------------+----+      +----+---------------------+-----+
|  1 | DTE <----PG---> DCE |  1 |      |  1 | DTE <----PG---> DTE |  1  |
|  7 | DTE <----SG---> DCE |  7 |      |  7 | DTE <----SG---> DTE |  7  |
|----+---------------------+----|      |----+---------------------+-----|
|  2 |     ----TD---> DCE  |  2 |      |  2 |     ----TD---> DTE  |  3  |
|  3 | DTE <----RD---      |  3 |      |  3 | DTE <----RD---      |  2  |
|  4 |      )              |    |      |  5 | DTE <)              |     |
|  5 | DTE <)              |    |      |  6 | DTE <)--DSR---      |20/4 |
|  6 | DTE <)              |    |      |  8 | DTE <)              |     |
|    |             (> DCE  |  4 |      |    |            (> DTE   |  5  |
| 20 |     ---DTR--(> DCE  | 20 |      |20/4|    ---DTR--(> DTE   |  6  |
|  9 | DTE <---DCD---      |  8 |      |    |            (> DTE   |  8  |
+----+---------------------+----+      +----+---------------------+-----+

 Figure 5E. DTE-DCE 4-way direct      Figure 5F. DTE-DTE 4-way null-modem


 Pin  DCE  direction  DCE   Pin         Pin  DTE  direction  DCE   Pin
+----+---------------------+----+      +----+---------------------+-----+
|  1 | DCE <----PG---> DCE |  1 |      |  1 | DTE <----PG---> DCE |  1  |
|  7 | DCE <----SG---> DCE |  7 |      |  7 | DTE <----SG---> DCE |  7  |
|----+---------------------+----|      |----+---------------------+-----|
|  2 | DCE <----RD---      |  3 |      |  2 |     ----TD---> DCE  |  2  |
|  3 |     ----TD---> DCE  |  2 |      |  3 | DTE <----RD---      |  3  |
|  4 | DCE <)--RTS---      |8,5 |      |4,20|      )              |     |
| 20 | DCE <)              |or 6|      |  5 | DTE <)       (> DCE |  4  |
|8,5 |     ---DCD--(> DCE  |  4 |      |  6 | DTE <)       (> DCE | 20  |
|or 6|             (> DCE  | 20 |      |  8 |              (  DCE |8,5, | 4
+----+---------------------+----+      +----+---------------------+-----+

Figure 5G. DCE-DCE 4-way null terminal  Figure 5H. DTE-DCE 2-way direct


 Pin  DTE  direction  DCE   Pin         Pin  DCE  direction  DTE   Pin
+----+---------------------+----+      +----+---------------------+-----+
|  1 | DTE <----PG---> DCE |  1 |      |  1 | DCE <----PG---> DTE |  1  |
|  7 | DTE <----SG---> DCE |  7 |      |  7 | DCE <----SG---> DTE |  7  |
|----+---------------------+----|      |----+---------------------+-----|
|  2 |     ----TD---> DCE  |  2 |      |  3 |     ----TD---> DTE  |  3  |
|  5 | DTE <)              |    |      |  4 | DCE <)---RD--(      |4,20 |
|  6 | DTE <)              |    |      | 20 | DCE <)--RTS--(      |  5  |
|  8 | DTE <---DCD--       |8,5,| 6    |    |             (> DTE  |  6  |
|    | DTE <---DSR---      |  6 |      |    |             (> DTE  |  8  |
```

```
 │       │                     (> DCE │  20 │      │       │                    │      │
 │       │                     (      │5,6, │8    │       │                    │      │
 └───────┴────────────────────────────┴─────┘    └───────┴────────────────────┴──────┘
```

Figure 5I. DTE-->DCE 2-way direct   Figure 5J. DTE-->DCE 2-way direct

```
  Pin    DTE  direction   DTE  Pin
┌───────┬─────────────────────┬─────┐
│   1   │ DTE <----PG---> DTE │  1  │
│   7   │ DTE <----SG---> DTE │  7  │
├───────┼─────────────────────┼─────┤
│   2   │      ----TD---> DTE │  3  │
│   3   │ DTE <----RD---      │  2  │
│   5   │ DTE <)              │     │
│   6   │ DTE <)        (     │20,4 │
│   8   │ DTE <)        (> DTE│  5  │
│ 20,4  │     )         (> DTE│  6  │
│       │               (>DTE │  8  │
└───────┴─────────────────────┴─────┘
```

Figure 5K. DTE-DTE 2-way nul-modem

6) When all else fails, read the manual. This normally gives the pinout. But not always; Apple, IBM, Epson are all a bit naughty in that, often, all the information one actually needs is only in the hardware manual, available at extra cost! Cultivate your dealer assiduously, give him presents, bring him other customers, make friends with him. Then, maybe you might get lucky!

What next?

This is only the start. I suggest the next stage is to find out how the protocol work, what MODEM7 and KERMIT do. Much can be learned here by getting the relevant CP/M Software Library discs. Kermit is on UK33 (doc) and UK34,35 (sources). The ?MODEM?.??? discs are too numerous to mention, so get the catalogue. If one is really keen, then burrowing into the hardware will be helped by the excellent book written by Sol Libes and Mark Garetz (Reference 6).

I hope someone will do a nice simple article, on RS232 software drivers, which is deep enough to have working examples for the Z80-SIO, 8251, and 64180. My system, alas, uses the simplest possible routine we could get away with, i.e., 2 data, 2 control, 1 clock option. The hardware can do the full protocol, but not my software. Z80-PIO Parallel ports seem much easier!

References

1. EIA Standard RS-232C.
   Electronics Industries Association.
   Washington D.C., 20006, 1969.

2. "RS232 Made Easy"
   Seyer
   1984

3. "The Amstrad CP/M Plus", pages 90-97
   Clarke & Powys-Libbe
   1986

4. "The Interface Circuits Data Book"
   Texas Instruments, Inc.
   1981

5. "Interface, Bipolar, (etc) Databook"
   National Semiconductor Corporation
   1983

6. "Interfacing to S-100/IEEE-696 Microcomputers", Chapter 11
   Libes & Garetz
   1981

7. "Compact Disc Articles"
   Jan 1985 to Feb 1986, May 1986, pp.27-28, June pp.80-81
   Watkingson
   Electronics & Wireless World
   1985,86

```
8. "Disc Drive Articles"
   Mar 1982 to Feb 1983, Feb pp.44-48
   Watkingson
   1982,83
```


EOF