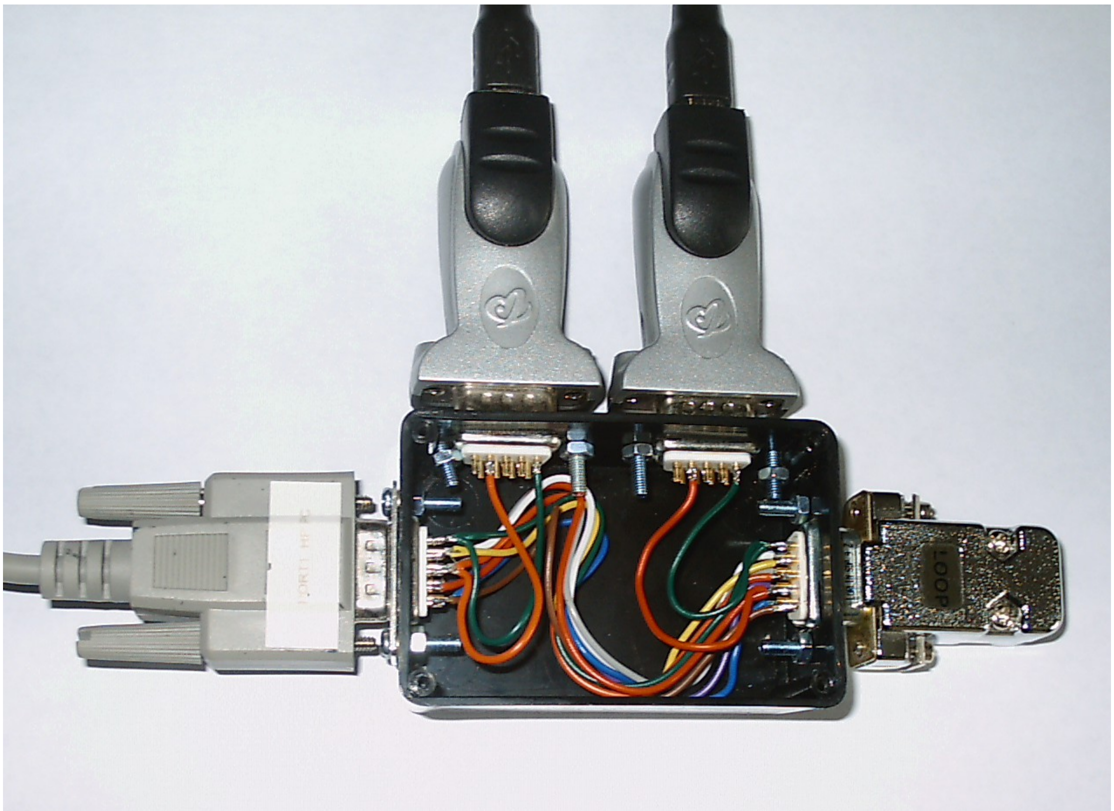
RS232 Monitor Project

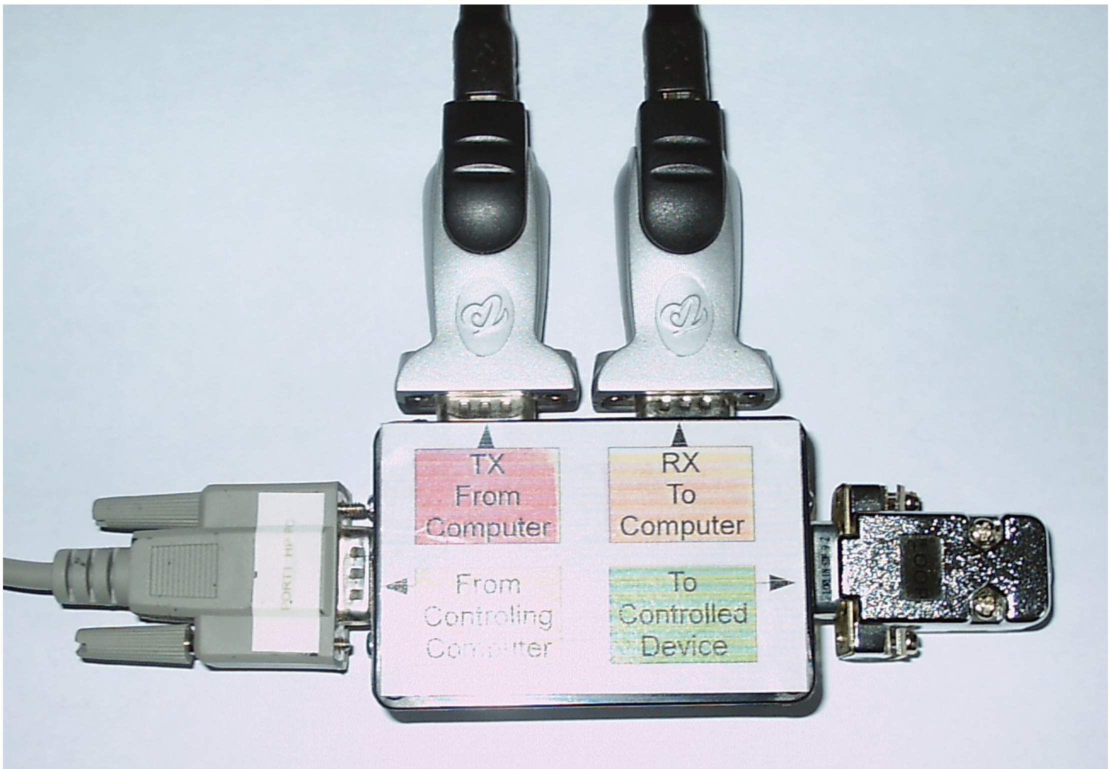
Originally Published in QL Today Vol 15, Issue 2, Dec-Jan 2010/2011

This is a very simple project to monitor the communications across a RS232 link. This could be between two computers. Alternately between a computer and some other RS232 controlled device. So checking the communications both ways at the same time, between the controlling computer and the controlled device. So you can see the data being sent to the controlled device and the data returned from the controlled device back to the computer. It can also be used for checking or researching the protocol between devices. In fact the reason I developed this project was, I wanted to check the communications between my PC running QPC2 and an AOR7030 communications receiver. Which does not have the simplest protocol. I could use this to send commands to the receiver and check the returns and that receiver had done what I told it to do. It saved a lot of time fault finding my code. Since it permitted me to run my development program and at the same time monitor the communication across the RS232 link.

So what is required. If you using original QL type hardware then you will need two computers. Since the black box original QL has only 2 RS232 ports. However if you have a SuperHermes then this will be different since you could use the extra RS232 ports this option provides. So this could be a new use for the QL sat in the cupboard. When you use two computers, one can be running your application, and second one to monitor the RS232 link. However, if you are say, using QPC2, then only one computer may be required. This would depend on your software, it is possible to run your application and a RS232 monitoring program at the same time. Or have the monitoring routines as part of your application. Either way you will require the breakout box, detailed below. Depending on the PC computer hardware you are using, you may need USB to RS232 adapters. Even if your PC computer has an RS232 port, which is becoming more and more rare. If it has, it will be most likely only be one and you will most likely need to provide more RS232 ports using USB to RS232 adapter(s). To provide the main port for controlling your device, plus two further ports for the monitoring. I run this on both my desk top PC and my Asus Eee PC computers with no problem.



Finished box open.



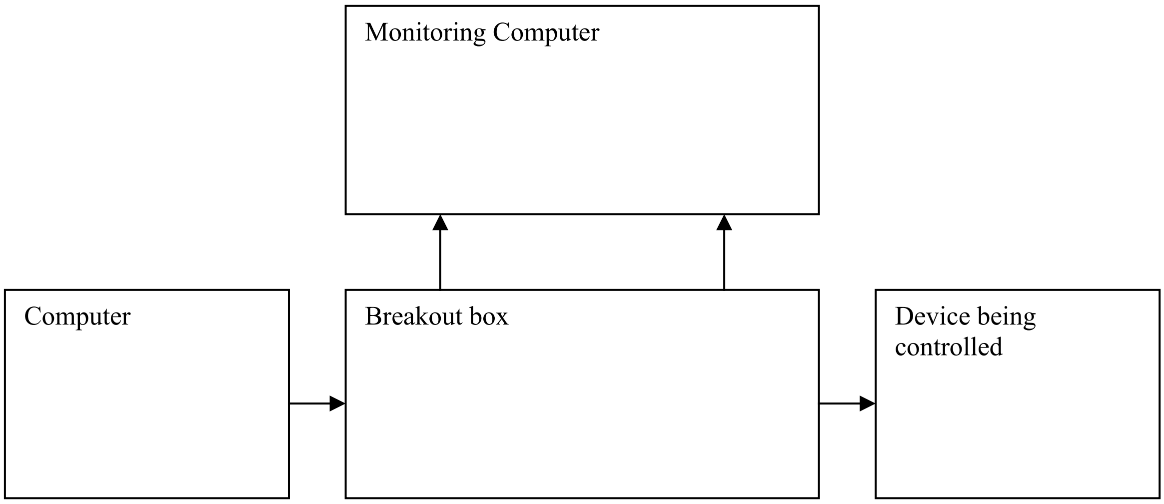
Finished box closed

Part required and approximate cost

1 x D-Type 9 way plug, Maplin code RK60Q		£1.39
3 x D-Type 9 way socket, Maplin code RK61R	£1.39	£4.17
1 x T3 box, Maplin code KC92A	£2.19	£2.19
3M Nuts and Bolts		
Hook up wire		
2 x USB to Serial Adapters, Maplin code ZP43W	£14.99	£29.98
Total cost approximately		£37.73

The above information correct at time of writing.

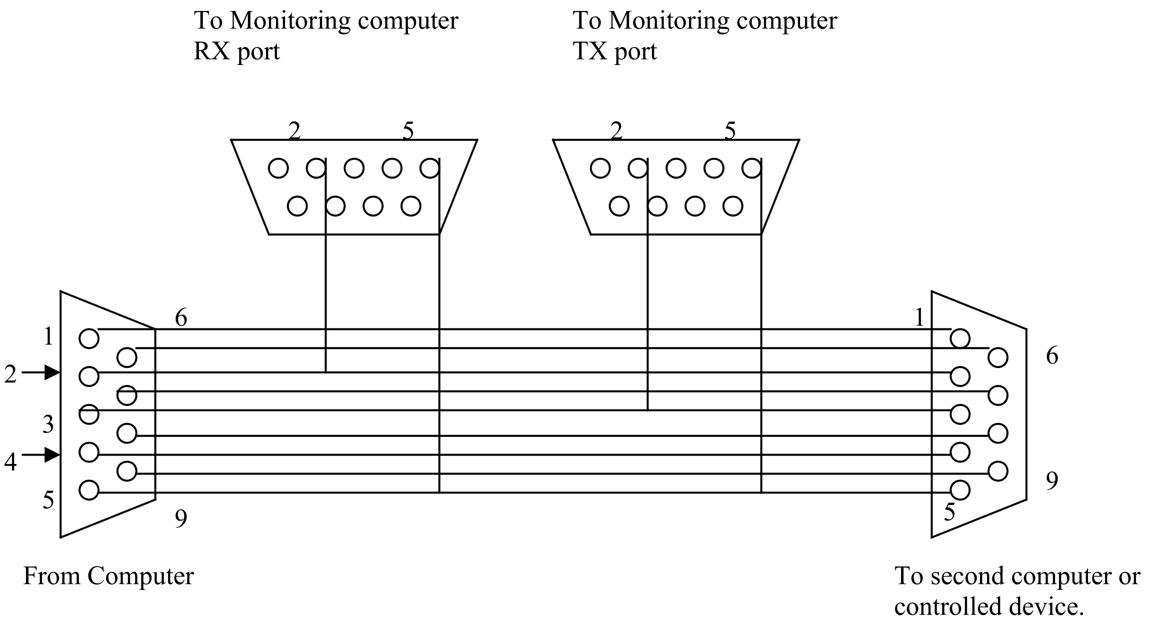
As can be seen this is as simple as is can get.



An example of how the box can be connected up is shown above. This example uses separate computers for controlling and monitoring, but this need not be the case if you have more than 3 RS232 ports on your computer.

As can be seen from the circuit diagram, all connections between the computer and the controlled device, with a spur for the serial data transmit and receive lines. This means if the hardware hand shake is used between the computer and controlled device this is not affected.

So how does this work. Very simple, we monitor pin 2 [RXD(Received Data)] and pin 3 [TXD(Transmitted Data)] of the main path from the controlling computer to the controlled device, on two RS232 ports, one monitoring the RXD and the other TXD. So we just use the RXD on the two RS232 monitoring connectors.
The break out box has 4 x 9 pin D-Type connectors. 3 female 9 pin D-Types and one male 9 pin D-Type. One male and one of the female connectors are connected together pin for pin. So that is pin one to pin one, pin two to pin two and so on. Pin five is the ground pin. Pin 2 is the receive data pin and pin 3 is the transmit data pin. The two remaining female 9 pin D-Types have ground connected to pin 5, and one connector pin2 is connect to pin 2 on the main pass connector and the second remaining male 9 pin D-Type pin 2 is connect to pin 3 on the main pass connector. See the diagram below.



You will see from the basic program shown below, that we open two channels one for each monitoring port. Read the inputs on these ports and then prints out the data. That is it for a very basic monitoring operation. In some applications you may wish to show the data in Hex or Binary form which is very easy to do.

```
20 OPEN#4;ser1:REMark open serial port, change port number for your system
30 BAUD#4,9600
40 OPEN#5;ser2: REMark open serial port, change port number for your system
50 BAUD#5,9600
60 REPeat loop
70 q$=INKEY$
80 IF q$=="q" THEN EXIT loop:REMark quit loop and program
90 t$=INKEY$(#4)
100 PRINT#1;t$;
110 r$=INKEY$(#5)
120 PRINT#2;r$;
130 END REPeat loop
140 CLOSE#4
150 CLOSE#5
160 STOP
```

Monitoring system set-up

The following program is a more advanced monitoring routine with two windows displaying both the TXD and RXD data, in decimal and hex form. This program also has some error trapping as well.

```
10 REMark RS232 monitor Feb 2010
20 setup_serial_ports
30 setup_screen
40 run_monitor
50 CLOSE#3
60 CLOSE#4
70 STOP
1000 DEFine PROCedure setup_screen
1010 xsize=SCR_XLIM
1020 ysize=SCR_YLIM
1030 WINDOW#0;xsize,ysize,0,0
1040 PAPER#0;0:INK#0,7:CLS#0
1050 WINDOW#1;xsize-20,(ysize/2)-20,10,10
1060 PAPER#1;2:INK#1;7:CLS#1:BORDER#1,2,255
1070 WINDOW#2;xsize-20,(ysize/2)-20,10,(ysize/2)+10
1080 PAPER#2;3:INK#2,7:CLS#2:BORDER#2,2,255
1090 AT#0;0,2:PRINT#0;"Data transmitted by the computer to the controlled
device"
1100 AT#0;38,2:PRINT#0;"Data received by the computer from the controlled
device"
1110 END DEFine setup_screen
1200 DEFine PROCedure setup_serial_ports
1210 INPUT#0;"Input serial port number that is monitoring transmitted data
from computer to controlled device :";portt
1220 portt$="ser"&portt
```

```
1230 PRINT#0;"Input baud rate for port ";portt;" ,this the rate being transmitted from the computer :";:INPUT#0;"";porttb
1240 test_baud_rate porttb: IF baud_ok=0 THEN GO TO 1230
1250 INPUT#0;"Input serial port number that is monitoring received data to computer from the controlled device    :";portr
1260 portr$="ser"&portr
1270 PRINT#0;"Input baud rate for port ";portt;" ,this the rate being received from the controlled device :";:INPUT#0;"";portrb
1280 test_baud_rate portrb: IF baud_ok=0 THEN GO TO 1270
1290 BAUD portt;porttb
1295 OPEN#3;portt$
1300 BAUD portr;portrb
1310 OPEN#4;portr$
1330 END DEFine setup_serial_ports
1400 DEFine PROCedure test_baud_rate (check_baud)
1410 baud_ok=0
1420 IF check_baud=300 OR check_baud=600 OR check_baud=1200 OR check_baud=2400
OR check_baud=4800 OR check_baud=9600 OR check_baud=19200 THEN baud_ok=1
1430 IF baud_ok=0 THEN PRINT#0;"Incorrect baud rate, baud rate must any of the following supported rates, 300, 600, 1200, 2400, 4800, 9600 or 19200"
1490 END DEFine test_baud_rate
1500 DEFine PROCedure run_monitor
1510 REPeat main_loop
1520 q$=INKEY$
1530 IF q$=="q" THEN EXIT main_loop
1540 t$=INKEY$(#3)
1550 IF t$<>"" THEN PRINT#1;t$;
1555 IF t$<>"" THEN INK#1;0:PAPER#1;4:PRINT#1;"(";CODE(t$);")";:INK#1;7:PAPER#1;2
1556 IF t$<>"" THEN INK#1;0:PAPER#1;5:PRINT#1;"[";HEX$((CODE(t$)),8);"]";:INK#1;7:PAPER#1;2
1560 r$=INKEY$(#4)
1570 IF r$<>"" THEN PRINT#2;r$;:r=CODE(r$)
1575 IF r$<>"" THEN INK#2;0:PAPER#2;4:PRINT#2;"(";CODE(r$);")";:INK#2;7:PAPER#2;3
1576 IF r$<>"" THEN INK#2;0:PAPER#2;5:PRINT#2;"[";HEX$((CODE(r$)),8);"]";:INK#2;7:PAPER#2;3
1580 END REPeat main_loop
1590 END DEFine run_monitor
32000 DEFine PROCedure UPDATE
32010 SAVE win1_RS232Monitor_RS232Monitor_bas
32020 PRINT "Update Complete"
32030 END DEFine UPDATE
```

Next time I will look at how you can get a I2C port running on your QPC2.