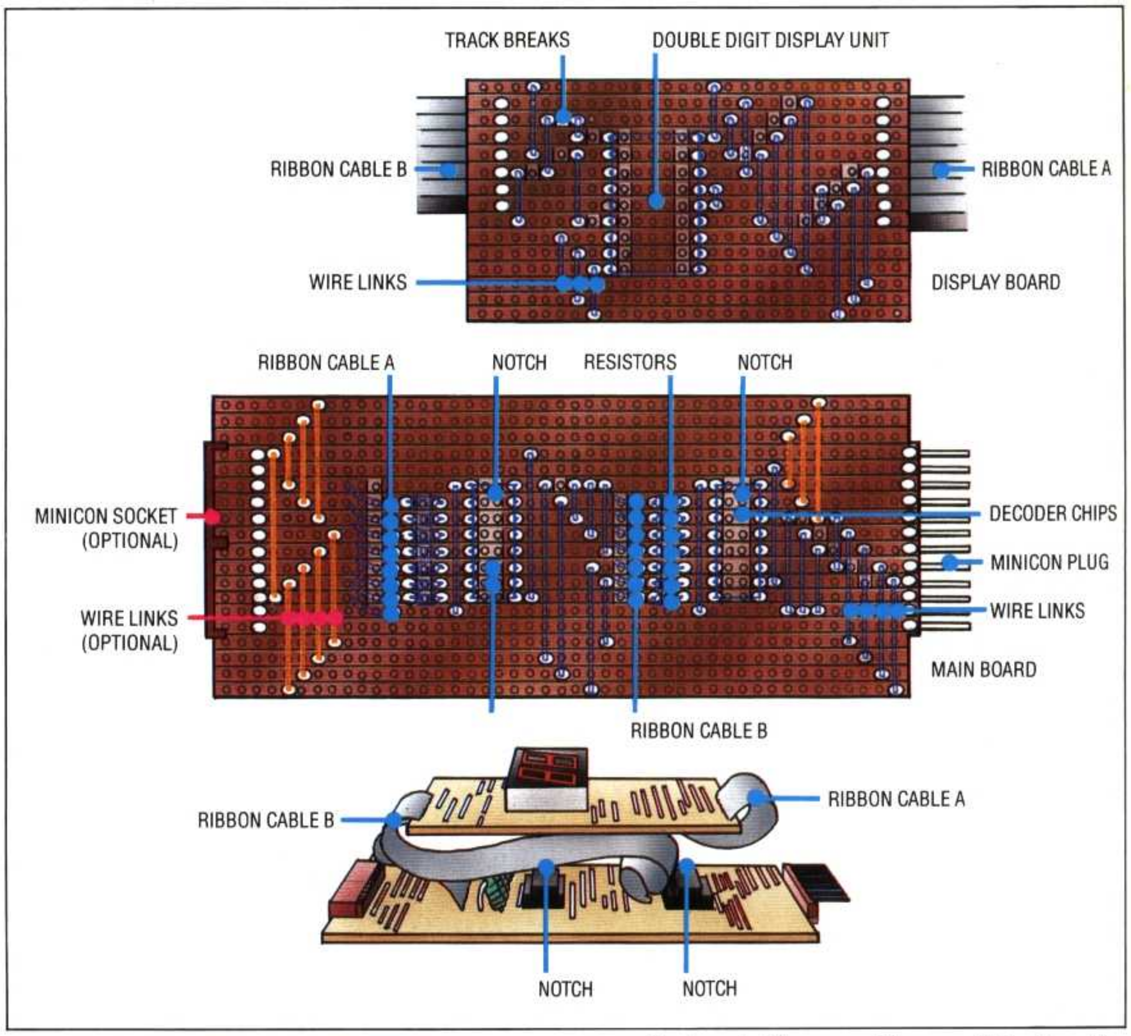
Laying It Out
Cut the veroboard to the two sizes required (19 tracks of 46 holes; 15 tracks of 28 holes). Cut the track breaks first on both boards. Solder the two chip sockets in place first, then the wire links and the resistors. If the bus extension socket is not required, then omit the red-coloured links in the illustration. Fit the minicon plug and (optional) socket to the main board, and solder the display unit in place — dots towards the socket end of the board. Solder the connection ribbon cables in place, so that they go straight from board to board without twisting. Now plug in the chips — make sure they are oriented as shown



```
160 IF INKEY(-99)=-1 THEN PROCdelay
170 ENDPROC
180 :
190 DEF PROCdelay
200 FOR I =1 TO 500:NEXT
210 ENDPROC
```

```
10 REM CBM 64 MULTIPLEXING
30 DDR=56579:DATREG=56577
40 POKEDDR,255
50 LB=15*16:RB=15
60 INPUT"DATA TO BE MULTIPLEXED";DT
70 POKEDATREG,DT+LB
80 GOSUB1000:REM SLOWER
90 POKE DATREG,DT*16+RB
100 GOSUB1000:REM SLOWER
110 GOTO70
120 :
1000 REM SLOWER S/R
1010 GETA$
1020 IFA$=" "THENGOSUB2000:REM DELAY
1030 RETURN
1999 :
2000 REM DELAY S/R
2010 FORI=1 TO 250:NEXT
2020 RETURN
```

A simple application is to use the twin seven-segment displays as a hexadecimal counter. This displays a count of the number of pulses input to the user port from a simple make-or-break switch connected to one port line. On first glance this seems a trivial task until you realise that all eight user port lines are needed for the display, leaving none for input. If we specify one of the lines for input, say line 0, then the computer's I/O system will always hold this line high, no matter what number may be present in the data register. If 128 (10000000 in binary) were to be placed in the data register, then this would instantly be changed to 129 (10000001) because line 0 is held high for input. This would obviously give incorrect count values on the displays. The solution lies in using a technique similar to multiplexing. If we set line 0 to accept input for a short time only and set all lines to output for a longer time, then the displays will appear to glow continuously with the correct counter value, with only a flicker of the incorrect value caused by setting line 0 for input.

```
10 REM BBC COUNTER
20 DDR=&FE62:DATREG=&FE60
50 count=0
55 :
60 REPEAT
70 PROCinput
72 PROCadd
73 FORI=1TO40
75 PROCdisplay
77 NEXT I
80 UNTIL count>255
90 END
999 :
1000 DEF PROCadd
1010 IF flag =1 THEN count=count+1
1050 ENDPROC
```