**Scelbi Oscilloscope Board**

Paul Robson May 2015

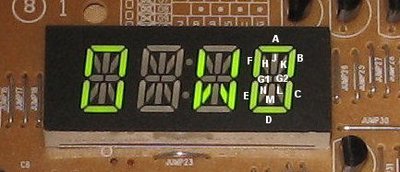
**Analogue Board**

This is relatively simple. A pair of voltage levels are maintained using the two Op-Amps integrators, one representing the X axis and one the Y axis. These can be changed using Xp/Xm/Yp/Ym and reset using Q1-Q4 which clears the position. The XpXmYpYm lines adjust X and Y in the minus or plus direction, setting these on for a period while the oscilloscope is not blanked will ‘draw a line’ in the same way a Vectrex works. As with the Vectrex, it needs to be continually refreshed.

The Y Axis is offset using the D-A convertor R26-R29 and R30 – this represents the current line (hence there are 16 vertical line positions)

**Digital Board**

The digital board allows the creation of an n x 16 character display – the exact values of n depend on R8 on the analogue board, which has to be code refreshed as it drives an oscilloscope – persistence of vision makes the display seem stable. Each of those characters is a ‘starburst’ character, a bit like an extended 7 segment character which allows the displaying of most of the standard character set.



(Note, it is not identical to this, the bottom vertical line is split into two, and there is a dot option in the centre of the character)

Unlike the Vectrex design, this cannot be generalised to (say) draw graphics on the scope – the hardware only allows the drawing of a starburst character.

Latched Input

Z1,Z9,Z10 and Z18 latch a 16 bit word from the output ports (driven using BB-BE – connections are down to the user). Of this 15 bits drive the starburst (see the Spreadsheet for the actual pattern). These lines are referred to as A0-A7 and B0-B7. The 16th line, B7 resets the whole display position when it is logic ‘0’ via the INIT output and by resetting the line counter Z7.

Counter

Each character is displayed over a period of 24 CPU cycles. These cycles start after the upper byte of the display latch is written (e.g. Z10,Z18) which sets the latch using Z14 pins 1,4,5.

Timing is done using the 7496 shift registers Z29-Z33. These do not operate as shift registers per se, but function like a 24 bit equivalent of the CMOS part 4017, e.g. only one bit is set and it shifts left to right across the diagram as you go. When the drawing is finished the character generation is stopped by Z33 pin D going high and stops until the next character is written.

Line Counter

The current line is maintained by Z7, a 7493 binary counter, which is clocked by the newline pattern and cleared by B7.

Segment Decoding

The guts of this part of the circuit is the 4 8 input NAND gates Z16,Z18,Z26 and Z35. Using the inverted outputs of the timing shift registers (the rows of inverters) these decide for each ‘pulse’ in the timing sequence (e.g. the outputs Z29-Z33 only one of which is ‘1’ at once) whether or not to output Xp/Xm/Yp/Ym in that position of the timing chain.

Sequence Adjusting

This is bizarre but simple ! There are 22 possible timing counts (Z29-Z32 A-E and Z33 A and B) but only 15 data bits (A0-A7,B0-B6). This ‘stretches’ this sequence so a different input from the latches is connected into Z4 pin 10 in each timing part (each input is a combination of a timing line from the shift register and a latch input, (hence the NAND gates Z2,Z11,Z19,Z26,Z3).

The effect of this is for each offset in the timing shift register (e.g. the 22 timing positions from the shift register) one of the selected outputs from the latches is displayed on the BLNK line. This sequences is the “control bits” row in the spreadsheet which shows for each timing entry (the S/R bit set) which input latch line is connected to BLANK and which adjustments are made.

When BLANK is zero and one of Xp, Xm, Yp, Ym (or two in fact) a line is drawn on the oscilloscope. Joining together all these lines (or a dot for B0) draws the segments of the starburst character.

After the character is drawn the ‘oscilloscope drawing position’ is put to the next horizontal position (done in S5B)

New Line

The circuitry at the bottom left (Z4,Z3,Z12) handles the new line character. When you put a certain value into Z1/Z9 (this is I think 0101 0101 binary) this sets the output of Z21/4 to logic ‘0’, which sets the BLNK high (so nothing is drawn) and (I you trace it) clocks Z7 which is a 7493 – this connects to the L1/L2/L4/L8 on the analogue circuitry, thus changing the drawing line. A spare gate from Z15 simultaneously resets the OpAmp circuitry (via BJ output, INIT) – this goes to 1 whenever either B7 is low or the pattern is 01010101.

Coding:

Output $00 into latches Z10/Z18 ; clears the screen

For each row on the screen:

For each character on the row:

Look up the bit pattern for the character ; bit 15 must be set

Output the lower part into Z1/Z9

Output the higher part into Z10/Z18 ; starts timing

Wait 24 CPU Cycles ; let it draw char

Put %01010101 into Z1/Z9 ; forces a new line

This can be optimised by removing the 24 CPU Cycle wait and making sure the bit pattern lookup takes at least 24 CPU Cycles (it almost certainly will !) and having a delay before the new line.