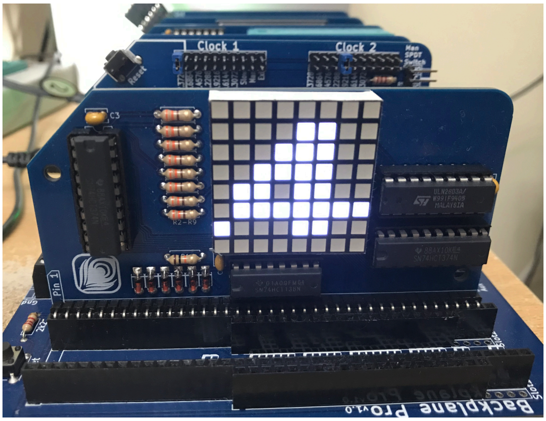
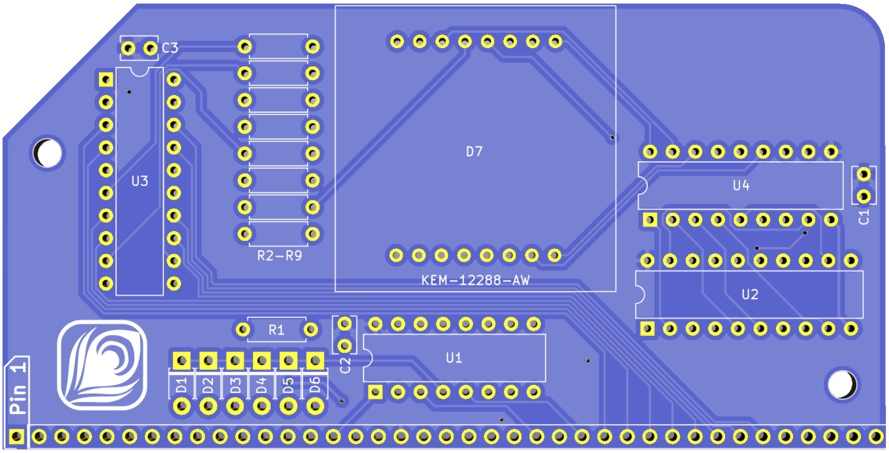
8x8 LED matrix display module - designed for RC2014

This module allows you to control a small 8 pixel by 8 pixel display. You can use it to display scrolling messages, patterns and animations, run John Conway's Life and even play games!

The display's 64 LEDs are arranged in a matrix. The package has 16 pins, so it can be controlled by two 8-bit ports. The module listens for writes to ports 0 and 2 (in line with other RC2014 modules such as the digital i/o and bubble LED display).



Assembly



Parts list

	PCB			1
D7	LED Matrix, 12288-A	(common cathode)		1
U1	74HCT138			1
U2,U3	74HCT374			2
U4	ULN2803A			1
R1	resistor		10k	1
R2-R9	resistors		330r	8
D1-D6	diodes		1N4148	6
C1-C3	capacitors		100nf	3
	IC socket		16 pin	1
	IC socket		18 pin	1
	IC socket		20 pin	2
	right-angle header		40 pin	1

Assembly notes

The matrix has a part number on one edge. Orientate the part so that this number is down (closest to the number printed on the board).

If you supply your own 8x8 LED matrix, note that the board is designed for a 32mm x 32mm (1.2”) common cathode matrix. If you’ve bought a kit then the matrix should be an equivalent even if the part number doesn’t match the number on the silkscreen.

Take care with the orientation of the diodes. The silkscreen shows a line at one end which corresponds with the line printed on the diodes.

Also take care with the orientation of the ICs and sockets. These traditionally have a notch and/or a spot marking pin 1. The notch is marked on the silkscreen.

The LEDs can be intense, depending how you’re driving them (brightness can be controlled by the proportion of time that they’re on - ‘duty cycle’) and the colour of the LEDs.   Feel free to use larger resistor values for R2-R9

The diffuser is optional. Feel free to use the display with or without, whichever you prefer.

Programming the display

The matrix has 8 rows which are driven by port 0 and 8 columns which are driven by port 2. This means that it's possible to light any single pixel, or a row or column of pixels, by sending values to those two ports. However, you can't light any combination of pixels you like across the whole display. Therefore your program must send the pixels for each row in turn. As long as this is repeated fast enough, persistence of vision makes the LEDs appear to be on constantly. This is known as multiplexing.

When you switch on and reset your RC2014 you may see a line sweep down the matrix, the display may be completely lit or not lit, or show random data.

In BASIC,

```
        out 0,1
        out 2,255
```

should light the top row only

then,

```
        out 2,1
```

should light only the top-right pixel.

Try this BASIC program:

```
        10 FOR I=0 TO 7
        20 READ R,C
        30 OUT 2,0 : OUT 0,R : OUT 2,C
        40 NEXT I
        50 RESTORE
        60 GOTO 10
        1000 DATA 1,60, 2,66, 4,169, 8,169
        1010 DATA 16,133, 32,185, 64,66, 128,60
```

It'll be a little flickery; BASIC is not quite fast enough to make the image appear to be solid. But at least this demonstrates the multiplexing technique: we visit each row in turn by sending a value of 1, 2, 4, 8, 16, 32, 64 and 128 to port 0 and at the same time set the value of the pixels in that row as a value between 0-255. Setting port 2 to 0 before changing port 0 (line 30) avoids 'ghosting' of the pixels from the row above.

The example code in assembly and forth uses these principles. <insert address here> They usually also maintain a buffer to store the pixel data.