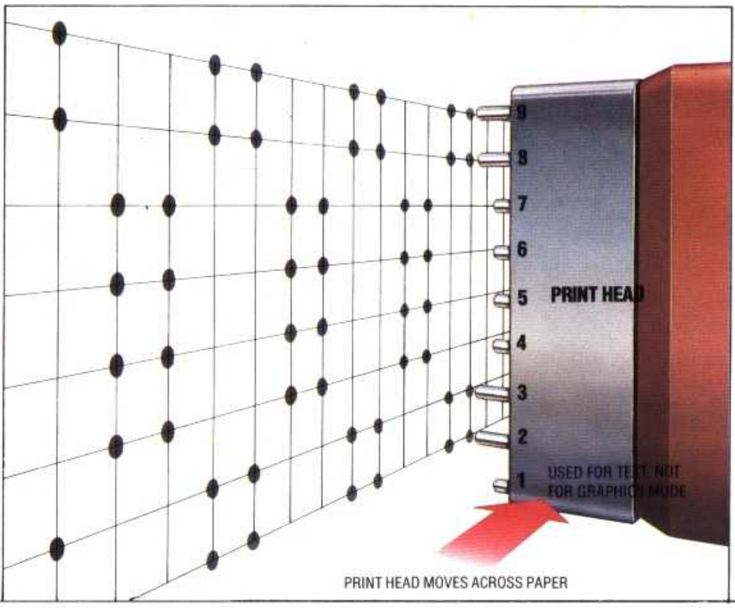
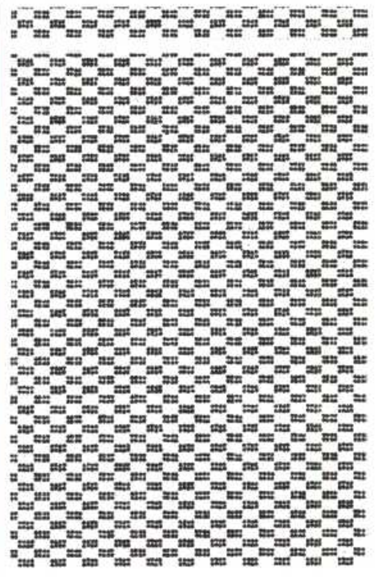
Pin Point
The pattern was produced on a dot matrix printer by sending alternate pairs of the decimal numbers 195 and 60 to the print head. The chart (below) shows how these numbers in binary are interpreted by the print head pins, (illustrated on the right). Controlled paper feed causes the next line to overprint the gap left by Pin 1

PIN NUMBER	PIN VALUE		
9	128	●	○
8	64	●	○
7	32	○	●
6	16	○	●
5	8	○	●
4	4	○	●
3	2	●	○
2	1	●	○
		195	60

● PIN FIRES
○ PIN DOES NOT FIRE


9
8
7
6
5
4
3
2
1
PRINT HEAD
USED FOR TEXT, NOT FOR GRAPHICS MODE
PRINT HEAD MOVES ACROSS PAPER



STEVE CROSS

numbers of each pin. This process is then repeated for each of the 480 s across the page.

In the illustration there are two pin patterns: CHR$(195) and CHR$(60). So to print the first four columns of the line pattern we type:

```
LPRINT CHR$(195);CHR$(195);CHR$(60);
  CHR$(60);
```

After four columns the pattern repeats, so a FOR...NEXT loop takes care of the rest of the line.

It is important to realise that CHR$(60) in the example does not instruct the printer to print the ASCII character with code 60 — it is a way of representing the data for the pins in the print head. The printer recognises it as such because we have previously transmitted the CHR$(27);"K" sequence to turn on the graphics mode.

This method of printing, known as *bit image printing*, is described for an Epson FX-80 printer; other printers use a similar method, but the exact details will vary. Producing graphics in this way is quite laborious, and only really suitable for patterns. A much better way of printing graphics is by means of a *screen dump*. This is a program that copies what is displayed on the monitor screen onto the paper.

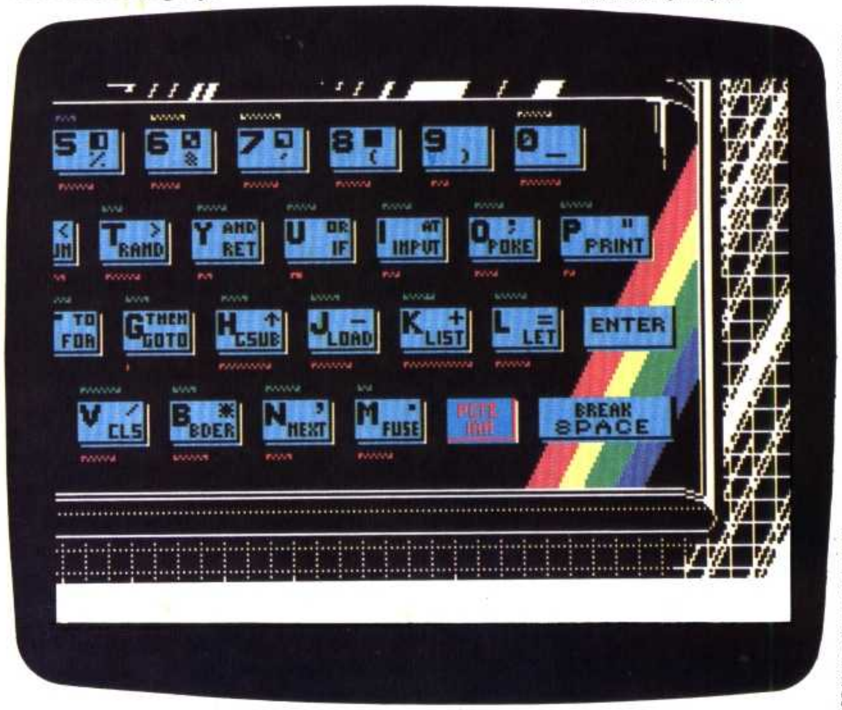
By scanning across and down the screen display, the program tests to see if the pixel is on at each position. If it is, then we want a pin in the print head to fire at the corresponding position on the paper. The scanning is done by using the POINT(x,y) function, or similar commands that are available on most micros; if a pixel is lit then the function POINT(x,y) will be 1; if it is unlit, the function is 0. The different screen resolutions of different micros mean that some adjustment might be necessary.

One problem that might have occurred to you is: how does a screen dump program handle colour displays? The usual solution is to use different dot patterns for each colour. A screen pixel that is black might be printed using four dots in the form of a square; one that is red might be represented by a two dots; and one that is white would not use any dots. The POINT(x,y) function produces a different number dependng on the colour of the pixel, and so can still be used.

Screen dump programs are usually added to the end of the program producing the picture, in the form of subroutines. To 'dump' the picture to the printer you might press the key 'P', and the program would then jump to the subroutine. A screen dump program written in BASIC tends to be quite slow, taking perhaps five minutes to print out a small picture. Machine code versions are slightly quicker.

As you can see, the graphics capabilities of dot matrix printers are reasonably advanced — if a little cumbersome to use. Once mastered however, the printed page can be as attractive as the screen display.

Splash Of Colour
This picture of the Spectrum keyboard was produced on a colour ink-jet printer — in effect this is a dot matrix printer in which the pins are replaced by ink jets



SCREEN DUMP BY DIMENSION GRAPHICS

GRAPHICS BY IAN McKINNELL