Dotting The I's

The 'character generator' is the section of your computer's memory that defines the way characters look on the screen. On some systems it is possible to design your own symbols

We have already seen in the Basic Programming course (see page 214) that all alphanumeric characters — and graphic symbols if your computer has them — are stored in RAM memory in the form of eight-bit codes (usually ASCII), so that one character occupies one byte.

When information is printed on the screen, the codes for each character are put into a reserved area of memory called the 'video RAM'. If, for example, the letter A is printed in the top left-hand corner of the screen, the first byte of video RAM will contain the code 65 (ASCII for A). If a C is printed below the A, and the computer has a 40-column screen, then the value 67 will be found in the 41st location of video RAM, and so on. How does the computer convert the value 65 into the pattern of dots that makes up the character A on the screen? The answer lies in a device called a 'character generator'.

A character generator is simply a collection of patterns stored as bits in memory. Home computers have the character generator stored in ROM — thus permitting an immediate display of characters when the machine is switched on. The character generator may be incorporated into the ROMs that contain the BASIC interpreter and operating system, or it may be on its own ROM chip. Where the latter is the case, you will often find independent suppliers offering replacement ROMs that will produce a foreign character set, or a range of specialist symbols for, say, engineering or mathematics. However, an increasing number of machines permit the character generator to be transferred into RAM, which allows the programmer to design his own characters and symbols.

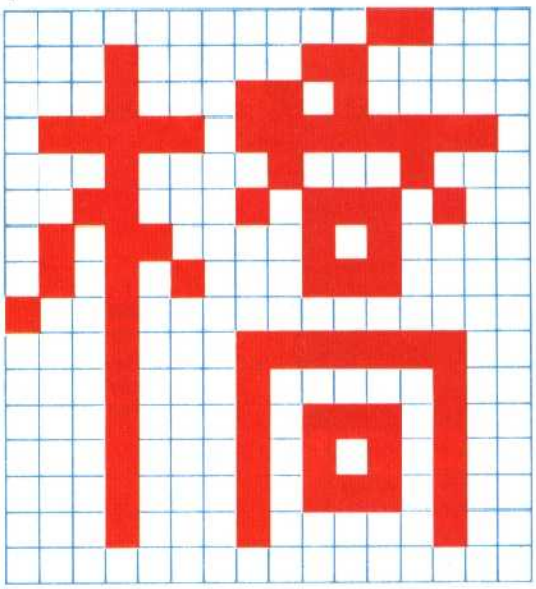
All characters are constructed on a matrix of dots, which on most home computers is eight by eight, although a larger matrix would result in greater legibility and a wider range of displayable characters. Characters are designed by filling in squares on the grid. They are defined by representing each filled square with a 1, and blank squares with a 0, giving a total of 64 bits, or eight bytes for each character.

The first byte in the character generator would represent the bit pattern for the top line of the first character in the table. If the computer can display only the ASCII characters with codes from 0 to 127, then the character generator will require 128×8 bytes (1 Kbyte of memory).

The difficulty for the computer is that when the raster scan on the television screen is generating the topmost line of the display, it needs to produce the topmost line of dots for the character positioned at the top left of the screen, followed by the topmost line of the character positioned to the right of it, and so on across the screen. Then, when the raster scan starts on its second pass, it must find and display the second line for each of the characters on the top row of the screen.

The video circuitry achieves this by having two independent counters. One counter keeps track of which location in the video memory corresponds to the point that the raster scan is passing over. The other counts the raster scan lines, starting at zero for the first line and reaching seven for the eighth, and then resetting to zero on the ninth, and so on. Thus, the computer looks up the ASCII or display code from the video memory, multiplies it by eight and adds on the current value of the line counter. This gives it an address in the character generator for the eight-bit pattern that corresponds to the correct row of the character it is currently scanning.

Let's take the example of generating the character A, which has an ASCII code of 65. We can calculate that the first line of the character will be stored in byte number 520 (65×8+0); the second line in byte 521 (65×8+1); the third line in byte 522 (65×8+2); and so on. All that remains is for the video circuitry to convert those eight bits into a sequence of voltages that will switch the scanning electron beam on and off to display the pattern on the screen.



Complex Characters
Japanese characters can be quite complicated, and the usual 8×8 matrix cannot display enough detail to make them readable. The character for 'bridge', shown here, is just readable on a 16×16 matrix. Better clarity is obtained by using a 24×24 dot matrix