USEFUL POINTERS

We have examined ways of defining characters on the Commodore 64, BBC Micro and Sinclair Spectrum. Here we discuss possible program improvements, and concentrate on the problem of saving and loading our redefined character set into a specified area of the 64's memory.

Now that our character-defining programs are up and running (see pages 572 and 588), it's worth spending some time comparing the three versions and, perhaps, improving them. The Commodore version was written first, since it is the most difficult of the three to program. This version was then translated, virtually line by line, for the other two machines. Partly because of this translation, and partly because space was limited, the screen formatting is rudimentary, and no use is made of colour, sound or hi-res graphics. Improvements in all these areas can, therefore, obviously be made, but will not be discussed here.

Leaving aside questions of programming efficiency (not really vital in this program, since there are no speed-dependent tasks), we will concentrate on the user interface: instructions, help, command keys and facilities.

There are no instructions in the program, mainly because the listing had to be fitted onto a single page of the course. An instruction page could be printed on the screen at the start of the run, and there is probably room on the main screen display for some abbreviated reminders — a cursor movement display, perhaps, and one-word summaries of the command keys. This should largely remove the need for a help page.

The choice of command keys might be improved. On the BBC Micro and the Spectrum the cursor is moved around the window by the usual cursor control keys, whereas on the Commodore the unshifted function keys are used. This makes for very convenient programming in the Commodore version since the ASCII codes of the eight function keys are consecutive between 133 and 140, but the layout of the keys themselves is not exactly ergonomic and they do not repeat — unlike the BBC and Spectrum keys. This last can be changed on the Commodore by POKE 650,128, but the function keys themselves cannot be made easier to use, so you may wish to restore cursor control to the cursor keys.

Another possible improvement is the choice of cursor movement strategy. As written, this simply disallows as illegal any command that would move the cursor out of the window. The alternative is to 'wrap' the cursor in some fashion: if it is moved off the bottom of the window, it can wrap around to the top of the window, and vice versa, and similarly for horizontal wrapping. This is easily programmed, but requires more code than the simple tests used in subroutine 3500.

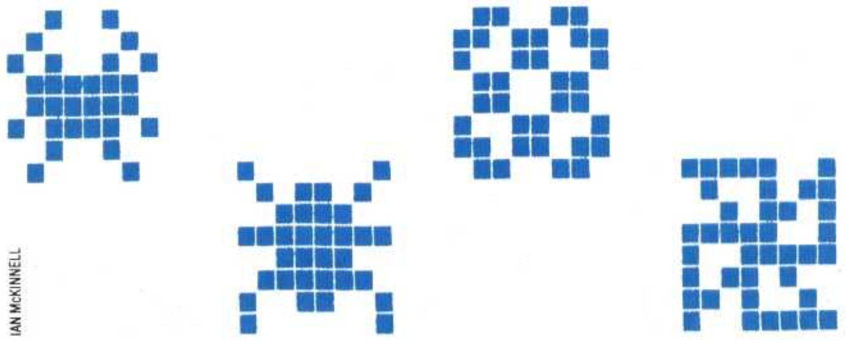
The commands provided are the minimum necessary, and could well be extended. On the Spectrum and BBC, SAVEing and LOADing character sets could be included as commands, and in all three versions it would be useful to be able to copy one character's definition to another character — so that CHR$(N) and CHR$(N+1) represented the same character, for example. You might want a hard copy of the new character set, so a printer option could also be added. The program's simple modular structure makes adding these commands reasonably straightforward.

COMMODORE 64 SAVE

A problem unique to Commodore BASIC is that the SAVE command seems to refer only to the entire BASIC program area, whereas the other two machines' BASICS allow you to specify the area of memory you wish to SAVE. The Commodore LOAD command, however, does allow the loading of files to any desired area, so if we can solve the SAVE problem then we can store and retrieve the new character sets.

The Commodore SAVE command is pointed at the BASIC program area by two address pointers — TXTTAB (at locations 43 and 44), and VARTAB (at 45 and 46). The first, TXTTAB, points to the start of the BASIC program area (usually at address 2048 onwards), while VARTAB points to the start of the BASIC variables area; since this starts where the BASIC program finishes, VARTAB effectively points to the end of the BASIC program area. If we change these pointers so that they indicate the start and finish of the new character set, and then issue a SAVE command, that should solve the problem.

Before we do this, however, we might reconsider the location of the character set itself. The subroutine at line 61000 (see page 573) copies the ROM character set to a two Kbyte block


IAN McKINNELL