decimal value of bits 0-3	bits 3,2,1,0	location pointed to
0	0000	0
2	0010	2048
4	0100	4096
6	0110	6144
8	1000	8192
10	1010	10240
12	1110	12288
14	1110	14336

The value of bit 0 in this register is unimportant, while bits 4 to 7 control other functions, and must be left unchanged. We use 11110000 (240 decimal) as an AND-mask for this purpose, and 00001110 (14 decimal) as an OR-mask to make the register point to 14336 — the start location of our character set:

```
POKE 53272,(PEEK(53272) AND 240) OR 14
```

Now, using a FOR..NEXT loop we can perform the actual copying.

While a program is copying the ROM character set into RAM, the CPU cannot deal with I/O device interrupts. The keyboard, for example, interrupts the CPU every sixtieth of a second, causing it to scan the keyboard for a keypress. These interrupts are triggered by the system timer. If the CPU were interrupted by an I/O device while the character set was occupying the I/O ROM space (as is the case during copying), then the system would probably crash, and only turning the power off and on would reset the machine. Fortunately, we can disable the interrupt mechanism by setting bit 0 of location 56334 to zero; the other bits of this location must be left unchanged, so the following logical POKE command should be used:

```
POKE 56334, PEEK(56334) AND 254
```

Once the interrupts are disabled, and the CPU is forced to look at the character set in ROM, then copying can begin.

We wish to copy, say, the 64 characters from '@' to '?' — screen codes 0 to 63 — so we must copy the 512 locations (8×64=512) starting at 53248 into a suitable block of RAM. Various areas can be used, and our choice here is a block starting at location 14336. This would normally be in the BASIC program area but we can protect it by lowering the top-of-memory pointer in location 56, thus:

```
POKE 56, 32
```

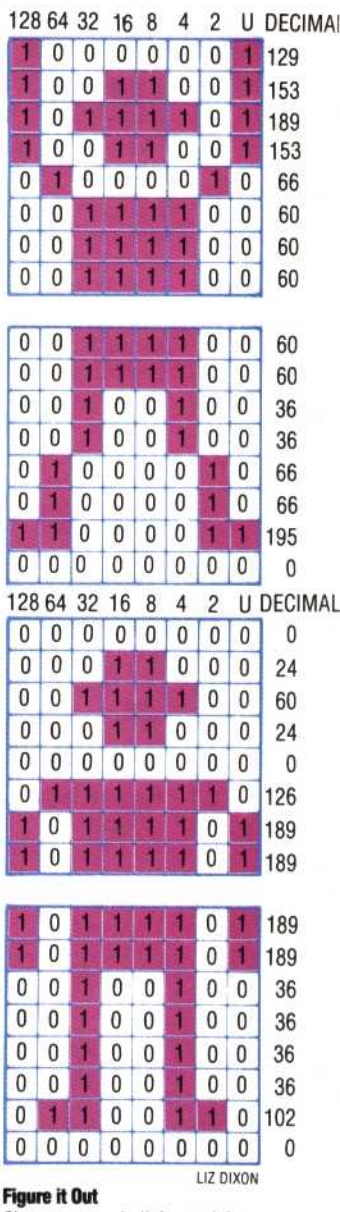
DESIGNING NEW CHARACTERS

Each character in the Commodore set is designed on an eight by eight dot matrix. Each row of the matrix is interpreted as a binary number (dots that are illuminated count as one, dots that do not show up on the screen count as zero) and so requires a byte of storage, and the entire eight-row character requires eight consecutive locations in memory. The starting location of the bytes describing any character can be calculated from the start address of the entire block, and the screen code of the particular character, thus:

Start of character = 14336+8×(screen code)

Once the locations of the bytes defining a character are known, we can POKE new values into those bytes, thus changing the dot patterns appearing on the screen when that character is printed. As long as the one character set is enabled, pressing the appropriate key will cause the new character to appear on the screen. In the demonstration program, we redefine the characters [,],£, and ↑ (codes 27-30) as parts of a figure, and achieve animation by printing and overprinting different versions of the figure.

```
130 :
140 REM**** COPY ROM CHAR SET ****
150 POKE56,32
   :REM LOWER TOP OF MEMORY
160 POKE56334,PEEK(56334)AND254
   :REM TURN OFF INTERRUPT TIMER
165 POKE1,PEEK(1)AND251
   :REM FLIP TO CHAR ROM
170 FOR I=0 TO 511
   :REM COPY
180 POKE14336+I,PEEK(53248+I)
   :REM 64
190 NEXTI
   :REM CHARACTERS
200 POKE1,PEEK(1)OR4
   :REM FLIP BACK TO I/O
210 POKE56334,PEEK(56334)OR1
   :REM TURN ON INTERRUPT TIMER
220 POKE 53272,(PEEK(53272)AND240)
OR14:REM SET CHAR POINTER
230 REM****  COPY COMPLETE  ****
240 :
250 :
300 REM**** ATHLETIC ARTHUR ****
310 FOR I=14552TO14552+31
   :REM READ
320 READ A:POKEI,A:NEXT I
   :REM CHAR DATA
330 PRINTCHR$(147)
   :REM CLEAR SCREEN
340 POKE55338,14:POKE55378,14
   :REM COLOUR CHRCTR. LIGHT BLUE
350 POKE1066,27:POKE1106,28
   :REM ARMS UP
360 FORI=1TO500:NEXT I
   :REM DELAY LOOP
370 POKE1066,29:POKE1106,30
   :REM ARMS DOWN
380 FORI=1TO500:NEXT I
   :REM DELAY LOOP
390 GOTO350
480 :
490 :
500 REM**** ARMS UP DATA ****
510 DATA129,153,189,153,66,60,60,60
520 DATA60,60,36,36,66,66,195,0
530 REM**** ARMS DOWN DATA ****
540 DATA0,24,60,24,0,126,189,189
550 DATA189,189,36,36,36,36,102,0
```



LIZ DIXON

Figure it Out
Characters are built in an eight by eight dot matrix, described in eight consecutive bytes. Each row of the character is interpreted as a single-byte binary number, dots representing ones and spaces representing zeros. For use in Commodore 64 programs, these binary numbers must then be converted to decimal