when the keys are pressed in conjunction with the Shift key, *FX227 for when the keys are pressed in conjunction with the Control key, and *FX228 works when both the Shift and Control keys are pressed with the function keys.

Video functions and VDU drivers: Most of the control of display in the BBC Micro is performed by writing values to the VDU drivers. However, there are a couple of OSBYTE calls that are helpful in this context.

*FX19: Although this has no parameters, it is useful when you're programming moving graphics. Once executed, it causes the computer to wait until the next frame of the display is drawn before continuing. This results in moving graphics that are less 'flickery'. The call must be made whenever the pause is required. As the display is redrawn 50 times per second, it can be used to generate time delays or to provide interrupts to the CPU.

*FX218: This is a read/write OSBYTE call that informs us of the length of the 'VDU queue'. We have already explained how some VDU codes, when sent through the VDU drivers, expect other codes to follow them (see page 897). The number of bytes that the VDU drivers are still waiting for at any time is the number of bytes in the VDU queue. On the whole, this call is best used to read information back, and the result will be returned in the X register.

*FX20: This call enables us to redefine characters in the ASCII range 32 to 255, rather than the more usual and limited range of user-defined characters. We can thus redefine, in Modes 0 to 6, the characters that are accessed from the normal keyboard. To redefine the characters with ASCII codes in the range 32 to 128 we must set aside memory from our BASIC workspace by setting PAGE to a higher value than usual. The redefinition of these characters is done with the VDU23 call. Characters are redefined in blocks of 32, each block requiring 256 bytes of memory. The only parameter used is passed over in the X register. Full details of this technique are given in the user guide.

Sound: *FX210,n is a useful call for those plagued by noisy 'blast-the-aliens' games. It allows you to disable the sound effects. *FX210,0 enables sound to be heard in the usual fashion, but any other value as the X parameter kills the sound.

*FX211 to *FX214: These calls control the sound generated by CTRL-G or VDU7. They are read/write calls. *FX211,n controls the channel of the sound chip on which the VDU7 sound is generated. *FX12,n controls the volume of the sound generated, or the envelope used to generate the sound. The volume is coded in the same fashion as the amplitude parameter in the BASIC SOUND command — that is, -15 is very loud, 0 is inaudible, and positive numbers are envelope numbers. The value passed over to the X register in the FX call is given by (n-1)*8, where n is the parameter. *FX213,n controls the pitch of the tone generated by VDU7; *FX214,n controls the time for which the tone generated by VDU7 is played.

Escape key: *FX229,n enables the Escape key to be 'turned off'. *FX229,1 disables the Escape key, and *FX229,0 enables normal Escape key action.

*FX220,n enables the user to set up a key to generate the Escape event — n being the ASCII code of the key that you wish to act as Escape. The normal Escape key is disabled. Thus, *FX220,65 causes the Escape event to occur whenever the A key is pressed.

Buffers: *FX21,n flushes — or empties — a buffer. This operation removes any bytes from that buffer. For example, flushing the keyboard buffer removes any keypresses that might have accumulated while a program has been running. If you try a GET command when there are some unprocessed 'keys' in the keyboard buffer, the GET command will fetch a key from the buffer instead of waiting for a keypress to occur. Flushing a sound buffer finishes sound generation on that particular channel, even if there are sound commands waiting to be processed by the sound chip. The buffer to be operated on depends upon the X parameter (see table in the margin).

*FX138,n,m: This inserts the value m into the buffer numbered n. Buffer numbers are the same as for *FX21. Thus, *FX138,0,65 will insert the letter A into the keyboard buffer.

X Parameter	Buffer Operated On
0	Keyboard
3	Printer
4	Sound channel 0
5	Sound channel 1
6	Sound channel 2
7	Sound channel 3

All these OSBYTE calls can, of course, be made by passing the relevant parameters in the A, X and Y registers and making the subroutine call as described earlier.

We have covered all the major functions that come under the control of OSBYTE calls. There are many more that we could examine in detail — you will find them listed in the BBC Micro user guide — but the techniques involved in using them are the same as those described here.

ERRATUM

The User-Defined Commands (USERV) program listing on page 898 was not given in full. Here is the complete listing:

```
5  DIM C 100
10 USERV=&200
20 ?USERV=&00
25 ?(USERV+1)=&0C
30 FOR I%=0 TO 2
   STEP 2
40 P%=&C00
50 [ OPT I%
60     CMP #0
70     BNE notcode
80     TXA
90   .loop
95     JSR &FFE3
100    DEY
110    CPY #0
120    BNE loop
130    RTS
140  .notcode
145    RTS
150 ]: NEXT I%
160 .
200 FOR rep=1 to 10
210 FOR asc=33 TO 48
220 asc$=STR$(asc)
230 rep$=STR$(rep)
240 code$="*CODE
      "+asc$+","+rep$
250 $C=code$
260 X%=C MOD 256
270 Y%=C DIV 256
280 CALL &FFF7
290 NEXT: NEXT
```