Go for your guns

Create some great on-screen effects on the CPC464 with these programs by ***Brian Cadge***

The Amstrad's video display is produced by two pieces of hardware, the Video Gate Array and the CRT controller. The VGA controls the screen mode (resolution), and handles the loading of colours into palette memory. It is also used by the Amstrad to control the bank switching of ROMs.

The second part of the video hardwre is the CRT (Cathode Ray Tube) Controller chip. The Amstrad uses the 6845 CRTC chip to control the positioning and size of the display on the monitor. Although the firmware uses some features of the 6845 hardware, principally for hardware scrolling the screen, there are many other useful features of this chip which the firmware does not support.

The 6845 has a total of 16 internal registers, most of which are Write-Only, that is to say we can output values to them, but we cannot read the current values from them. Two Z80 I/O port addresses are used to control the 6845, these are *BCXX* and *BDXX*. The former is the address port and the latter the data port.

Values are sent to a peripheral chip, such as the 6845, using the *Out* command in Basic. To send a value to one of the 6845 registers we first of all send out the registers number to the adress port *BCXX* to tell the chip which register we want to change. The actual data value to be put in the particular 6845 register is then output to the data port. So to send the value 40 to register one in the 6845 we would use the following Basic commands: *Out &BC00,1 : Out &BD00,40.*

Although the address port will keep the value one, it is always advisable to output the register number immediately before the data just in case the firmware has since updated the address port.

So just what are the 6845 registers used for? The table below lists the more useful ones. Those missing are either not implemented by the Amstrad hardware, or are only partly implemented.

Register	Function
0	Horizontal Total
1	Horizontal Displayed
2	Horizontal Sync Position
3	Sync Widths
4	Vertical Total
5	Vertical Total Adjust
6	Vertical Displayed
7	Vertical Sync Position
8	Interlace and Skew
9	Maximum Raster Address
12	Start Address (high)
13	Start Address (low)

It is generally not a good idea to alter any register involved with Sync functions, as this can lead to the loss of picture and you will have to reset the computer. The useful registers have the following functions: Register 0 controls the overall 'height' of the physical monitor screen. Program Four uses this feature to give a 'neon lights' effect that can be very useful for title displays, etc. An interesting feature of changing this register is that it controls when the frame flyback pulse occurs. Making the screen longer (as Program Four does) means it takes longer for the CRT to scan one 'screen' and so the interrupt occurs less frequently.

Register One controls the number of columns displayed on the screen. The Amstrad sets this to 40 (note this value applies to all modes), but this can be changed to display more or less of the border. Similarly, Register Six controls how many text lines are displayed, initially 25. Register Two controls how far left the display starts and Register Seven controls how far down the display starts. Program One uses these positioning registers to allow you to move the text screen around on the monitor using the cursor keys.

Program Two uses the positioning registers and the size registers to repostion the screen in the top left hand corner of the monitor and then expand it to 24 columns by 34 lines in Mode 0. The border is completely concealed as the text screen now covers the whole of the monitor screen. In practice, if you change the size of the screen it will be necessary to check that the new display uses no more than 16K of memory or else wrap-round will occur. The firmware graphics and text routines will not operate as normal on different sizes of screen.

The position of text screen can be moved in all four directions in steps of one Mode 1 character. Register Five allows fine adjustment of the vertical position; this has many uses. For example, hardware smooth scrolling of graphics and text could be implemented. Program Three uses this register to give an 'earthquake' effect that can be used in games. The screen appears to 'shake' as in an earthquake. Try adding a *Call &BD19* in Lines 75 and 105 to see an example of smooth scrolling. This firmware routine waits for the current screen scan to complete before returning and can be used to tidy up screen updating.

The four programs presented with this article only show a few of the effects possible by direct manipulation of the CRTC chip. Although some of the effects may look strange, you cannot damage your monitor or computer by changing the register contents – if you lose the display just turn the computer on and off. You'll probably find lots more uses for the 6845 by experimenting with the registers yourself. But a word of warning, restrict all addresses in the *Out* command to *BC00* and *BD00* unless you know what you're doing or you might just lose the program in memory.

```
10 ' VDU PROGRAM #1
20 '
30 ' This program allows you to alter
40 ' position of the screen on the
50 ' monitor, using the cursor keys.
60 '
70 horz.pos=46:vert.pos=30
80 horz.adr=2:vert.adr=7
90 BORDER 1:INK 0,3:INK 1,26:PEN 1:PAPER 0:MODE 1
100 PRINT"Press the cursor keys to move the screen"
110 PRINT:PRINT"Press ESC ESC to finish...."
120 i$=INKEY$:IF i$="" THEN 120
130 IF i$=CHR$(&F2) AND horz.pos<60
    THEN horz.pos=horz.pos+1
140 IF i$=CHR$(&F3) AND horz.pos>32
    THEN horz.pos=horz.pos-1
150 IF i$=CHR$(&F1) AND vert.pos>20
    THEN vert.pos=vert.pos-1
160 IF i$=CHR$(&F0) AND vert.pos<40
    THEN vert.pos=vert.pos+1
170 CALL &BD19
180 OUT &BC00,horz.adr:OUT &BD00,horz.pos
190 OUT &BC00,vert.adr:OUT &BD00,vert.pos
200 GOTO 120
```

```
10 ' VDU PROGRAM #2
20 '
30 'This program demonstrates using
40 'all of the monitor screen for text.
50 'Giving 24 columns and 34 lines
60 'in mode 0.
70 '
80 ON BREAK GOSUB 240
90 MODE 0:BORDER 0:INK 0,3:INK 1,12:INK
   2,20:INK 3,18,3:PEN 1:PAPER 0
100 horz.adr=2:horz.cols=1:ver4.
   adr=7:vert.lines=6
110 OUT &BC00,horz.adr
120 OUT &BD00,50     'Move screen left 4
130 OUT &BC00,vert.adr
140 OUT &BD00,35     'Move screen up 5
150 OUT &BC00,horz.cols
160 OUT &BD00,48     'Display 48 columns
170 OUT &BC00,vert.lines
180 OUT &BD00,34     'Display 34 lines
190 FOR i=1 TO 520:PRINT"*";:NEXT
200 LOCATE 1,19:PEN 2
210 PRINT STRING$(24,154)+CHR$(15)+CHR$
   (3)+"POPULAR COMPUTING WEEKLY"+
CHR$(15)+CHR$(2)+STRING$(24,154);
220 PEN 1
230 WHILE INKEY$="":WEND
240 OUT &BC00,horz.adr
250 OUT &BD00,46
260 OUT &BC00,vert.adr
270 OUT &BD00,30
280 OUT &BC00,horz.cols
290 OUT &BD00,40
300 OUT &BC00,vert.lines
310 OUT &BD00,25
320 MODE 1:CALL &BC02  'reset default
   colours
```

```
10 ' VDU PROGRAM #3
20 '
30 ' This program can be used as a
40 ' subroutine to give an earthquake
50 ' effect in games.
60 '
70 FOR z=0 TO 2
80 OUT &BC00,5:OUT &BD00,z
90 NEXT z
100 FOR z=2 TO 0 STEP -1
110 OUT &BC00,5:OUT &BD00,z
120 NEXT z
130 IF INKEY$="" THEN 70
```

```
10 ' VDU PROGRAM #4
20 '
30 ' "Neon Signs" title program
40 '
50 MODE 0:INK 0,0:BORDER 0
60 a$=" POPULAR COMPUTING"
70 OUT &BC00,0:OUT &BD00,127
80 LOCATE 1,1:FOR x=1 TO LEN(a$):
   PRINT MID$(a$,x,1);
90 FOR d=1 TO 50:NEXT d,x
100 p=INT(RND(1)*15+1)
110 IF INKEY$="" THEN PEN p:GOTO 80
120 OUT &BC00,0:OUT &BD00,63:PEN 1:MODE 1
```