BBC DEMO 5

This is the fifth demonstration in a series to show off the speed of machine code graphics on the BBC Micro and Electron. You will need a sample Mode 5 screen but if you haven't got one, program B.SCREEN will generate one for you. There is a short pause while the code is assembled and then just can sit back and enjoy the show. It looks like someone is pouring grains of sand from the top of the screen, and the picture is gradually built up as each grain lands at the bottom.

The program makes use of a hidden screen, or screen *cache*. Various manipulations can be carried out on the cache before copying it to normal area of memory occupied by the screen RAM.

The Mode 5 screen takes up &2800 bytes of memory, starting at &5800 and running up to &7FFF. These values are different in other Modes, but let's concentrate on Mode 5. We can store a complete copy of the Mode 5 screen at &3000, i.e. &2800 bytes below the start of the screen RAM.

DEMO5 starts by loading in the screen file using OSCLI with the X and Y registers pointing to a command line at *COMMAND*.

It proceeds by copying the very last row of pixels in the screen cache to every row in the screen RAM. The address of the last row of pixels in the cache is given

GRAPHICS

Listing DEMO5

```
 10 REM BBC DEMO 5
 20 REM C.R.Dewhust 03/12/02
 30 :
 40 MODE 5
 50 HIMEM=&3000
 60 S%=&3000
 70 width=40
 80 depth=256
 90 bpr=&140
100 sourcerow=S%+width*(depth-8)+7
110 targetrow=&7EC7
120 PROCassem
130 VDU 23,1,0;0;0;0;
140 VDU 19,3,4;0;
150 CALL&900
160 END
170 :
180 DEFPROCassem
190 oscli=&FFF7
200 osbyte=&FFF4
210 FOR N%=0 TO 2 STEP2
220 P%=&900
230 [OPT N%
240 \Load screen into cache
250 LDX #command MOD256
260 LDY #command DIV256
270 JSR oscli
280 \Set the start row
290 LDA #sourcerow MOD256:STA &7C
300 LDA #sourcerow DIV256:STA &7D
310 \Set the finish row
320 LDA #targetrow MOD256:STA &7A
330 LDA #targetrow DIV256:STA &7B
340  \depth pointer
350 LDA #depth AND&FF:STA &7E
360 .l3
370 \Make copies of start & finish
380 LDA &7A:STA &76
390 LDA &7B:STA &77
400 LDA &7C:STA &78
410 LDA &7D:STA &79
420 \Copy depth counter
430 LDA &7E:STA &75
440 .l2
450 \Count 40 columns
460 LDA #width:STA &74
470 \Copy copies of start & finish
480 LDA &76:STA &70
490 LDA &77:STA &71
500 LDA &78:STA &72
510 LDA &79:STA &73
520 \copy 1 row to the screen
530 LDY #0
540 .l1:LDA (&72),Y:STA (&70),Y
550 \Move across to next column
560 TYA:CLC:ADC #8:TAY
570 \Increase high byte of screen and
cache addresses if low bytes became zero
580 BNE s1:INC &71:INC &73
590 .s1:DEC &74:BNE l1
600 \Move target row up the screen
610 LDA &76:AND #7:BEQ top
620 \Result was zero if at top of char
 row
630 DEC &76:JMP cont
640 \Move to start of char row above
650 .top:LDA &76:SEC:SBC #(bpr-7) MOD2
56:STA &76
660 LDA &77:SBC #(bpr-7) DIV256:STA &7
7
670 .cont
680 DEC &75:BNE l2
690 \Move source row up screen
700 LDA &7A:AND #7:BEQ top2
710 DEC &7A:JMP cont2
720 .top2:LDA &7A:SEC:SBC #(bpr-7) MOD
256:STA &7A
730 LDA &7B:SBC #(bpr-7) DIV256:STA &7
B
740 .cont2
750 \Move source row up cache
760 LDA &7C:AND #7:BEQ top3
770 DEC &7C:JMP cont3
780 .top3:LDA &7C:SEC:SBC #(width*8-7)
 MOD256:STA &7C
790 LDA &7D:SBC #(width*8-7) DIV256:ST
A &7D
800 .cont3
810 DEC &7E:BEQ exit:JMP l3
820 .exit
830 \Wait to see if a key is pressed
840 LDA #129:LDX #0:LDY #5:JMP osbyte
850 :
860 .command
```

SPECIMEN