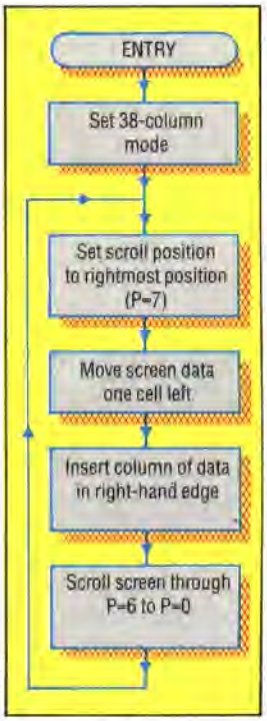

Steps Towards A Smooth Scroll
To achieve smooth scrolling, we can use a special facility offered by the Commodore 64's video control chip — VIC — which has special scrolling registers that allow the visible screen to move from its normal position relative to the border. Single pixels in the horizontal or vertical direction can be produced. Combining this effect with character copying in machine code, we can produce smooth scrolling on a reduced 38-column screen

A pointer to the memory area is initially set to point to the byte at the beginning of the memory area to be scrolled onto the screen. Once the first column has been scrolled on, then the pointer can be incremented by one to copy a second column onto the right-hand edge of the screen, from where it can be scrolled to the left. After this process has been repeated 40 times, a complete screen of data will have been scrolled on. The memory pointer should then be increased by 960 (1000-40) to point to the beginning of the next screen.

This process must be duplicated for the corresponding area of colour data. To simplify this, we should make the address of each byte in the colour map have a constant offset to the address of the corresponding byte in the screen data map. The process can be repeated for as many screens of data as have been designed and held consecutively in memory.

In order to use the scroll routine, several pieces of information must be passed before calling it. The routine needs to know:

1) The start address of the memory area where the screen data to be scrolled is held.
2) The offset to the corresponding colour data.
3) The number of screens of data to be scrolled on.
4) A delay value, used to slow down the smooth scroll operation.

This data should be POKEd to the locations set aside in the machine code program.

Basic Calling Program

```
10 REM ****************************
20 REM ****************************
30 REM **                        **
40 REM **  BASIC CALLING PROG    **
50 REM **  FOR SCROLL ROUTINE    **
60 REM **                        **
70 REM ****************************
80 REM ****************************
90 :
95 DN=0:REM FOR CASS DN=1
100 IFA=0THENA=1:LOAD"SCROLL.HEX",DN,1
110 POKE55,0:POKE56,32:CLR:REM LOWER MEMTOP
115 REMGOSUB1000:REM SET UP SIMPLE DISPLAY
120 :
130 LMEM  =49664:    REM START OF MEMORY
140 HMEM  =49665:    REM AREA
150 LCOFF =49666:    REM OFFSET TO COLOUR
160 HCOFF =49667:    REM MAP
170 NMSCR =49668:    REM NUMBER OF SCREENS
190 DELAY =49669:    REM DELAY VALUE
200 SCROLL=49670:    REM PROG START ADDRESS
210 :
220 REM PRINTCHR$(147):REM CLEAR SCREEN
230 INPUT"DECIMAL START ADDRESS";SA
240 HS=INT(SA/256):LS=SA-HS*256
250 POKELMEM,LS:POKEHMEM,HS
260 :
270 INPUT"NUMBER OF SCREENS";NS
290 POKE NMSCR,NS
300 :
310 INPUT"DECIMAL OFFSET TO COLOUR MAP";OS
320 HO=INT(OS/256):LO=OS-HO*256
330 POKELCOFF,LO:POKEHCOFF,HO
340 :
350 INPUT"DELAY VALUE < 256";DV
360 IF DV>255 OR DV<0 THEN 350
370 POKEDELAY,DV
380 :
390 SYS SCROLL
400 POKE53270,PEEK(53270)ORB
```

The program loads the machine code into memory and asks for the information required via INPUT statements. The program splits this information into LO-byte/HI-byte form where necessary and POKEs it to the storage spaces allocated at the beginning of the machine code program. The machine code routine is then called.

Any start address, offset and number of screens may be specified, although the results will not be very meaningful if you don't put any screen designs in the memory area specified. You can test your program by loading and running the short BASIC program that sets up two simple screens of data starting at location 8192. The offset to the colour data area is 3,000 bytes. To scroll this data area onto the screen, the following information must be given in response to the prompts from the calling program:

1) Decimal start address:	8192
2) Colour offset:	3000
3) Number of screens:	2
4) Delay:	255

Basic Loader

```
10 REM ************************
15 REM **  BASIC LOADER      **
20 REM ** FOR HORIZONTAL    **
30 REM ** SCROLL ROUTINE    **
40 REM ************************
50 :
60 FOR I=49670 TO 49945
70 READ A:POKEI,A
80 CC=CC+A
90 NEXT
92 READ CS:IF CS<>CC THEN PRINT"CHECKSUM ERROR":STOP
100 DATA173,22,208,41,247,141,22,208
110 DATA174,4,194,180,40,138,72,152,72
120 DATA173,22,208,41,248,24,105,7,141
130 DATA22,208,169,0,133,251,169,4,133
140 DATA252,169,0,133,253,169,216,133
150 DATA254,162,3,160,1,177,251,136
160 DATA145,251,200,177,253,136,145
170 DATA253,200,200,208,241,230,252
180 DATA230,254,177,251,198,252,136
190 DATA145,251,200,177,253,198,254
200 DATA136,145,253,230,252,230,254
210 DATA202,208,213,160,1,177,251,136
220 DATA145,251,200,177,253,136,145
230 DATA253,200,200,192,232,144,239
240 DATA173,0,194,133,253,173,1,194
250 DATA133,254,169,39,133,251,169,4
260 DATA133,252,32,241,194,173,0,194
270 DATA24,109,2,194,133,253,173,1,194
280 DATA109,3,194,133,254,169,39,133
290 DATA251,169,216,133,252,32,241,194
300 DATA162,0,173,22,208,41,248,141,22
310 DATA208,138,24,109,22,208,141,22
320 DATA208,172,5,194,136,208,253,202
330 DATA16,231,173,0,194,24,105,1,141
340 DATA0,194,173,1,194,105,0,141,1
350 DATA194,104,168,104,170,136,240,3
360 DATA76,19,194,173,0,194,24,105,192
370 DATA141,0,194,173,1,194,105,3,141
380 DATA1,194,202,240,3,76,17,194,96
390 DATA162,25,160,0,177,253,145,251
400 DATA202,240,29,165,251,24,105,40
410 DATA133,251,165,252,105,0,133,252
420 DATA165,253,24,105,40,133,253,165
430 DATA254,105,0,133,254,76,245,194
440 DATA96
450 DATA40227:REM*CHECKSUM*
```

Set Up Display Routine

```
1000 REM **** SET UP DISPLAY ****
1010 CL=3000:REM OFFSET TO COLOUR MAP
1020 SS=8192:REM START OF DISPLAY MAP
1030 FORI=SS TO SS+479
1040 POKEI,1:REM SCREEN CODE FOR 'A'
1050 POKEI+CL,1:REM WHITE
1060 POKEI+480,2:REM SCREEN CODE FOR 'B'
1070 POKEI+CL+480,14:REM LIGHT BLUE
1080 NEXT
1085 FORI=SS+960TOSS+999
1090 POKEI,3:REM SCREEN CODE FOR 'C'
1100 POKEI+CL,3:REM CYAN
1110 NEXT
1999 :
2020 SS=9192:REM NEXT SCREEN START
2030 FORI=SS TO SS+479
2040 POKEI,3:REM SCREEN CODE FOR 'C'
2050 POKEI+CL,5:REM GREEN
2060 POKEI+480,4:REM SCREEN CODE FOR 'D'
2070 POKEI+CL+480,0:REM BLACK
2080 NEXT
2085 FORI=SS+960TOSS+999
2090 POKEI,5:REM SCREEN CODE FOR 'E'
2100 POKEI+CL,2:REM RED
2110 NEXT
```