SPRITE MACHINE

INSTRUCTION MANUAL

Programmed by H. MYHRE/DATA LOFTET and JOHN ANDERSEN

© 1985 ANIROG SOFTWARE/SCANDOMATIC (NORWAY)

THE SPRITE MACHINE

A SPRITE is a user designed graphic which can be moved around the screen independently of the rest of the display. For example, a ship at sea could be made of one sprite in which case it will have minimal details. However, it could be made up of up to eight sprites (max. for CBM 64) showing a magnificent detailed ship in full sail.
Using the SPRITE MACHINE program, you can create sprites quickly and efficiently. You can choose the colour of the sprite and the background, or design it in multi-colour. The colours can be mirrored, rotated, scrolled and animated. You can choose the speed of the animation. If you have designed a horse, you can take it through it's paces from a canter to a full gallop.
Designing a big object is easy with this program. A ship in full sail can be designed using all the eight sprites. These sprites can be viewed together to give a full view of the complete ship. Any of the sprites making up the ship can be edited, scrolled or rotated while the whole ship is in view, so that you can see the effect of the change as you make it.
A 'DEMO' program is included in the package to show the versatility of the program. An absolute beginner will find the SPRITE MACHINE easy to use and can experiment with several sprites included in the program. The powerful set of commands together with a handy reference card will be very attractive to a professional programmer. The program contains many useful hints, tips and information about the working of the video chip in the 64, its memory maps and memory blocks, Sprite pointers and how to incorporate the sprites in your own program; in fact everything that a beginner or professional will need to know, saving many hours of work, looking for this invaluable information.
SPRITE MACHINE, with its powerful graphics manipulation, display and overlay facilities, is the most advanced sprite editor available.

THE PACKAGE INCLUDES:

- ★ THE SPRITE MACHINE PROGRAM.
- ★ WITH SEVERAL READY TO USE SPRITES.
- ★ COMPREHENSIVE MANUAL, AND A REFERENCE CARD - LISTING ALL THE COMMANDS.
- ★ A DEMONSTRATION PROGRAM.

LOADING SPRITE MACHINE

To Load: Type LOAD and press the RETURN key.

DISK

Insert the disk into the disk drive.
To load the main program, type:

```
LOAD"S*",8,1 [RETURN]
```

To load the demo, type:

```
LOAD"D*",8,1 [RETURN]
```

When you have studied this manual thoroughly, and you have learned to use the commands, you will only need the reference card, which is included in this manual. Instead of searching through the manual, just consult the card.

THE MODES AND COMMANDS

All the commands are one key operation. The first letter of the command has been used as far as possible. However some commands can destroy previously made sprites. As a safeguard against accidental erasure, these commands require use of SHIFT key. The keys are listed in the header of each command. ROTATE "R", means that you must press R to perform the rotate command. SAVE SPRITES [SHIFT] + "S" means that you have to hold down the SHIFT key and press S to save your sprites.

[ ].
Everything included in these brackets is a special key. [RETURN] means the RETURN-key. [SHIFT] + "A" means that you have to hold down the shift-key and press the a-key.

NORMAL MODE

This mode is for editing high-resolution sprites. The sprite is made up of 24★21 pixels. You are only allowed to have one sprite colour in addition to the background colour.

MULTI-COLOUR MODE. "M"

To enter multi-colour mode, press M. To get back to normal mode, press M once again. In multi-colour mode, you can have three different colours plus the background colour and resolution is 12*21 pixels.
Two pixels in normal mode correspond to one pixel in multi-colour mode.

Normal mode	**Multi-colour mode**
blank - blank	background colour
filled - blank	sprite colour
blank - filled	multi-colour ♯ 1
filled - filled	multi-colour ♯ 2

This is useful to know for transferring sprites between normal and multi-colour mode.

DISPLAY MODE. "D"

Display mode is provided to make it easier to create a picture containing multiple sprites. As in multi-colour mode, press D to get into display mode and back again. This mode can be entered in both normal and multi-colour mode. When you enter display mode, a "D" will appear right of the mode indicator and the sprite disappears from the sprite window. Above the sprite window, the sprites from 40 to 47 are displayed, ready to be positioned. Display mode simply means that several sprites can be displayed together. The sprite can be edited in this mode. You can freely move between normal and multi-colour mode in display mode. The Oops-command is not available in this mode, because the sprites are updated each time a key is pressed.

POSITION. "P"

Position can only be used in display mode. Pressing P causes the program to ask for the sprite to be positioned. Only sprites in location 40 to 47 can be displayed. Therefore to use display mode, you must store the sprites to be displayed in those locations. Use the cursor-keys or the joystick to position the sprite in the sprite window. Press X to change horizontal size, Y to change vertical size, or C to change the colour of the sprite. Press RETURN when finished. By pressing P again and choosing another number, a new sprite can be positioned in the sprite window. In this way, you can have up to 8 sprites in the sprite window. Among the sprites included in this package, is a viking-ship. The sprites for the ship are, in the sprite-blocks from 40 to 47. You may assemble them together and experiment with them. When the sprites are displayed, they can be edited as normal.

OVERLAY MODE. "O"

Overlay mode can only be entered from normal mode. In overlay mode you can place up to 8 different sprites upon each other. The resolution is 21*24 as in normal mode, but you can have up to eight different colours plus the background colour. When you enter this mode, the mode indicator displays "OVERLAY 32". This means that you are in overlay mode and the number is a constant showing which sprite-blocks are being used for overlay mode. This number is only used in animation, as you will see later. Below the mode indicator, the colours for the sprites will be displayed. The colour codes are the standard for the commodore machine. "COLOR 2: 5" means that the colour of the second sprite is colour number five, which is green.

THE CURSOR

The cursor is shaped like a "diamond" in normal mode, and like a "cross" in overlay mode. In multicolour mode, it's twice as wide and is made up by "〈〉". The cursor can be moved around with the cursor keys.

RETURN. [RETURN]

Pressing the return key causes the cursor to move to next row, first position.

HOME. [CLR/HOME]

This command returns the cursor to the home position, first column, first row.

CLEAR. [SHIFT] + [CLR HOME]

Clears the whole sprite and places the cursor at the home position.

DELETE. [INST/DEL]

Like in the basic editor, it deletes the pixel left of the cursor and moves the remaining pixels to the left.

INSERT. [SHIFT] + [INST/DEL]

The opposite of delete. All the pixels from the cursor to the right edge will be shifted to the right and the pixel under the cursor will be cleared. This command will not work if the cursor is at the last position of the line.

FILLING IN THE PIXELS

Each of the three main modes uses a different way to fill in pixels and clear them.

NORMAL MODE

Fill pixel ".'' (Full Stop)
Clear pixel - Space Bar
The cursor will advance to the next position in both cases.

MULTI-COLOUR MODE

To clear a pixel, use the space bar. The keys from 1 to 3 are used to fill pixels.

"1" : Sprite colour.
"2" : Multi-colour # 1
"3" : Multi-colour # 2

OVERLAY MODE

To clear a pixel, use the space bar. The keys from 1 to 8 are used to fill pixels.

"1" : Colour of sprite 1
"2" : Colour of sprite 2
"3" : Colour of sprite 3
"4" : Colour of sprite 4
"5" : Colour of sprite 5
"6" : Colour of sprite 6
"7" : Colour of sprite 7
"8" : Colour of sprite 8

JOYSTICK

JOYSTICK

A joystick can be plugged into port 2. It can be used for moving the cursor in all modes. Pressing the fire-button has different effects. In normal mode, it flips the pixel, so that a blank pixel is filled and the other way around. In multi-colour mode, it changes between the four colours. In overlay mode, it changes between the eight sprite colours. The joystick can also be used in the **position** command.

CHANGING COLOURS

CHANGING THE SCREEN COLOURS

F 2 : Change border colour.
F 4 : Change background colour.
F 6 : Change character colour.
F 8 : Change sprite colour.

CHANGING THE SPRITE COLOURS

NORMAL MODE

F 1 : Change background colour.
F 3 : Change sprite colour.

MULTI-COLOUR MODE

F 1 : Change background colour.
F 3 : Change sprite colour.
F 5 : Change multi-colour # 1
F 7 : Change multi-colour # 2

OVERLAY MODE

F 1 : Change background colour.

[SHIFT] + "1" : Change sprite colour 1.
[SHIFT] + "2" : Change sprite colour 2.
[SHIFT] + "3" : Change sprite colour 3.
[SHIFT] + "4" : Change sprite colour 4.
[SHIFT] + "5" : Change sprite colour 5.
[SHIFT] + "6" : Change sprite colour 6.
[SHIFT] + "7" : Change sprite colour 7.
[SHIFT] + "8" : Change sprite colour 8.

THE COMMANDS

SCROLLING SINGLE LINES

Any pixels shifted out of one edge will come in at the other edge. Position the cursor at the line to be scrolled and press the right key.

":" : Scroll line left. (Colon)
";" : Scroll line right. (Semi-Colon)
"↑" : Scroll line up. (Arrow Up)
"=" : Scroll line down. (Equal)

SCROLL. "S"

When you press S, the following message will appear at the top line, "COMMAND: SCROLL". Now use the cursor keys to scroll the sprite. The pixels will wrap around. When you are finished, and the sprite have been positioned, press the space bar to get back to normal operation.

ROTATE. "R"

This command rotates the sprite 90 degrees to the right. By rotating the sprite three times, you will achieve the same as a rotate to the left. There is a little problem. The grid in normal and overlay mode is 21★24 pixels. This means 24 pixels across and 21 pixels down. Only the 21 first pixels in each row is used. This leaves the last 3 untouched, since a rotation must have the same number of columns as rows. A good advice when you want to rotate a sprite, is to first scroll it as far to the left as possible, then rotate it and scroll it back in place.

TURN. "U" and "T" (MIRROR IMAGES)

The sprite can be turned vertically and horizontally.

"U" : Turns the sprite vertically.
"T" : Turns the sprite horizontally.

REVERSE. "I"

This command reverses the sprite. This command is not possible in overlay mode. It can be used in multi-colour mode, which changes the colours, and has the same effect as moving the sprite to normal mode, reversing it, and moving it back to multi-colour mode.

CENTER. "C"

This centres the cursor.

CHANGE SPRITE SIZE. "X" and "Y"

This changes the size of the sprite displayed in the sprite window.

"X" : Change vertical size.
"Y" : Change horizontal size.

THE SPRITE STORAGE

A maximum of 336 sprites can be stored. The sprite storage is divided into two memory-blocks of 16Kb, called block zero and block one. Each have capacity of 256 sprites, but some of the sprite-blocks are occupied. In block zero, the sprite-blocks from 0 to 31 are used for the screen and as a work-area. 32 to 39 are used for overlay mode. 40 to 47 are used for display mode. 48 to 255 are free to be used. The blocks from 32 to 47 can be used if you only intend to use normal and multi-colour mode. In block one, 0 tc 127 are free and 128 to 255 are occupied by the program. Trying to access a sprite-block below 32 in memory-block zero or over 127 in memory-block one will not work.

Memory-block zero

0-15 : Work area. (Not available to the user).
16-31 : Screen memory. (Not available to the user).
32-39 : Overlay mode storage.
40-47 : Display mode storage.
48-255 : Free for the user.

Memory-block one

0-127 : Free for the user.
128-255 : Used by the program. (Not available for the user).

NOTE: The video-chip, which produces the screen image, can't "see" the sprite-blocks from 64 to 127 in memory-block zero. The sprites are still there, but the video-chip thinks it "sees" the character-set. When you are working on a sprite in normal or multi-colour mode, it will be moved into a work-area, and the video-chip is told to display it from there. When you move to another sprite the old sprite is moved from the work-area to the sprite-block, and the new sprite is moved into the work-area. Also see the oops command. But when you are animating in overlay mode, the sprites can not be moved into a work-area, therefore the sprites from 64 to 127 should not be used in overlay animation.

NEXT SPRITE. "+"

Advances to the next sprite-block.

PREVIOUS SPRITE. "—"

Go to the previous sprite-block.

GET A SPECIFIC SPRITE. "G"

Pressing G will make the program ask for the sprite you have. Enter the number and press the return key.

COPY SPRITES. [SHIFT] + "C"

This command copies sprites from one place to another. You will be asked for the starting sprite to be moved and the last sprite to be moved. If you only want to copy one sprite, enter that sprite number as the first and the last sprite. After these two numbers are entered you will be asked for the first sprite-block to move to. The status will be displayed. See saving and loading. Sprites can only be copied within the same memory-block.
An example:
If you want to move the sprites from 145 to 152 into the sprite-blocks from 32 to 39 (The overlay area), you enter:
145 [RETURN] 152 [RETURN] 32 [RETURN] 39 [RETURN]

SAVE SPRITES. [SHIFT] +"S"

After pressing S, the program displays "TAPE/DISK". Press T, if you want to save the sprites to tape, or press D, if you want to save them to disk. The selected device is then displayed over the previous message. The program then asks for the first and the last sprite to be saved. The memory-block, is the block you are currently in. Finally, you are asked for the filename. It can be up to 16 characters long. A blank filename is only allowed for tape. After the program has been saved, the status is reported in the upper line, waiting fro you to press a key. If it says "ERROR", something went wrong. Perhaps you entered the wrong data, or asked for the impossible.
An example:
If you intend to save the sprites from 128 to 159 to a diskette, with the name "SPRITE-FILE", you enter:

```
D 128    [RETURN]  159  [RETURN]  SPRITE-FILE  [RETURN]
```

LOAD SPRITES. [SHIFT] + "L"

After pressing L, press D or T for disk or tape. Then enter the starting sprite-block, which the sprites will be loaded into. Finally, the file-name must be entered. The rules are the same as for saving with the status displayed after loading.
An example:
If you want to load some sprites into the sprite-blocks from 80 and further, from tape, saved with the filename "SPRITES", you enter:

```
T  80  [RETURN]  SPRITES  [RETURN]
```

SPECIAL COMMANDS

OOPS! [SHIFT] + "O"

Each time you get a new sprite on the screen, a copy is made of it, and put in the work area. This command retrieves that sprite. If you accidentally cleared the sprite, or messed it up, you can get the sprite back again with this command.

NOTE: The copy is updated if you perform one of these commands:
Get next sprite.
Get previous sprite.
Get specific sprite.
Copy sprites.
Save sprites.
Force. (Of course!)
Animation.
Make data-statements.

FORCE. "F"

This command forces the current sprite picture into the work-area as the copy. Later, when you performs oops, this picture will be put back again. If you have made a sprite, but want to experiment with it, use the force command to keep a copy of it. It can then later be brought back again with the oops command. Remember to stay within the same sprite-block. If you move to another one, then the new sprite will be put in the work-area.

ANIMATION. "A"

NORMAL AND MULTI-COLOUR MODE

You will be asked the animation speed. 254 is the slowest and 1 is the fastest. Then it will ask for the first sprite and the last sprite. After they have been entered, the program waits for you to press a key. Each time you press a key, except the RETURN key, which terminates the command, a new sprite will be displayed. If you hold down one of the keys, the sprite will change at the speed you entered. When the program comes to the last sprite, it starts all over again from the beginning. Press [RETURN] when you are finished.

OVERLAY MODE

Input commands are almost as in normal and multi-colour mode, but in addition, you are asked, how many sprites you want to display at the same time.
Remember that you can't use the sprite-blocks from 64 to 127 in memory-block zero.
An example:
If you have two horses and riders, you put the horse first in the first sprite-block, the rider in the next, the second horse in the one after, and the second rider in the last one. Then set the colours for the two first sprites as you normally do in overlay mode. If the sprites have been placed in the sprite-blocks from 128 to 131, and you want a fast animation, you enter:
1 [RETURN] 128 [RETURN] 131 [RETURN] 2 [RETURN]

MAKE DATASTATEMENTS. [SHIFT] + "D"

If you program in BASIC, it is often easier to have the sprites in data-statements. When you press [SHIFT] and D, you will be asked for the starting sprite and the ending sprite. Then the prompt "COMMENT:" will appear, waiting for you to enter the name you choose to give to the sprite. Before each block of data, a line is inserted with the comments you have entered. This makes it easier for you to remember what kind of sprite the data is for. Finally the status message will be displayed, waiting for you to press a key.

NEW. [SHIFT] + "N"

This command erases the data storage, and sets the number of free blocks to 48.

FREE. [SHIFT] + "F"

This gives you the amount of free blocks left for data-statements. Maximum is 48 blocks.

SAVE DATA-STATEMENTS. [SHIFT] + "Z"

The prompt "TAPE/DISK/ABORT" will be displayed. If you press T, tape will be selected, if you press D, disk will be selected, or if you press A, you return to normal operation. Then you are asked for the filename. Remember that a blank filename is not allowed for disk. After the file has been saved, the status message will be displayed. Press any key to exit. If you get screen message "ERROR", some of the causes can be:
Full disk.
Ending sprite less than starting sprite.
Disk drive not turned on.
Tape player not connected.
The saving has been interrupted.
Write protect on.

An example:
If you want to save the statements to tape, with "DATA" as filename, you enter:
T DATA (RETURN)

LOAD DATA-STATEMENTS. [SHIFT] + "X"

Choose disk or tape, then enter the filename. Read the status message and press a key. (See: save data-statements).

An example:
If you want to load the statements from disk, and the filename was "SPRITEFILE", you enter:
D SPRITEFILE [RETURN]

WARNING!
Trying to load programs without data-statements, or to load programs larger than 16 Kb, can crash the program.

BASIC SUPPORT

THE DATA STORAGE

You can make data-statement of up to 48 sprites, and then save them for later use. The saved program can be loaded in from BASIC. When you save your data, the sprite machine program automatically gives the line number starting from 50,000. Your own basic program line numbers therefore, must not exceed 50,000. This facility is provided so that the data statement can be easily managed with your own basic program.

An example:

The following example shows how you can create a sprite using the SPRITE MACHINE and then load your sprite into your programme and display it on the screen.

First you convert a sprite to data statements by pressing 'SHIFT' and 'D'. Then enter the number of the sprite you want, as the starting and ending sprite. SAVE the data statements by pressing 'SHIFT' and 'Z'.
Switch the machine off and then on. Enter the following basic lines:

```
 10  SPRITE = 11 : REM SET SPRITE TO 11
 20  FOR I = 0 TO 63 : READ A : POKE (SPRITE * 64 + I), A
     NEXT : REM READ THE SPRITE
 30  POKE 2040,SPRITE : REM SET SPRITE POSITION FOR FIRST SPRITE
 40  XCO = 53248 : YCO = 53249 : REM POKE ADDRESSES FOR SPRITE'S
     POSITION
 50  POKE XCO,250 : POKE YCO,200 : REM SET SPRITE POSITION
 60  ENABLE = 53269 : REM USED TO TURN THE SPRITES ON AND OFF
 70  POKE ENABLE,1 : REM TURN FIRST SPRITE ON
 80  COLOUR = 53287 : POKE COLOUR, 0 : REM SET COLOUR OF FIRST
     SPRITE TO BLACK
 90  XSIZE = 53277 : POKE XSIZE,1 : REM EXPAND VERTICAL SIZE
100  YSIZE = 53271 : POKE YSIZE, 1 : REM EXPAND HORIZONTAL SIZE
```

Now, use the MERGE-ROUTINE, which is explained below to link your saved data-statements to this program. When this is done, you can RUN the program, and your sprite will appear in black in the lower right corner.
The lines are numbered from 50000, and can be added to your main program. Unfortunately, the Commodore BASIC lacks a command to merge two programs. There are a large number of programs available on the market with a merge command. A simple MERGE program is listed below.

Before loading the data statement tape, type:

```
POKE251,PEEK(43) : POKE252,PEEK(44)
A=PEEK(45)+PEEK(46)*256-2
H=INT(A/256) : L=A—256*H
POKE43,L : POKE44,H
```

Now, load the program:

```
From disk:  LOAD"FILENAME",8
From tape:  LOAD"FILENAME"
```

After loading, type:

```
POKE43,PEEK(251) : POKE44,PEEK(252)
```

The two programs are merged and you can save the whole program on tape or disk.

```
To disk:  SAVE"FILENAME",8
To tape:  SAVE"FILENAME"
```

HINTS, TIPS AND INFORMATION

THE MEMORY-BLOCKS

Everything that is displayed on the screen, is controlled by the Video Interface Chip, often called the VIC. It gets its information from the data stored in the memory. The Commodore 64 has 64 Kb of ram, but the memory is divided into 4 memory-blocks of 16 Kb and can only "see" one of them at the same time. This means that all the data must be stored in the same block. When you turn the power on, block 0 is automatically chosen.
But that isn't all, because the 64 doesn't always "see" it as you do. When block 0 is chosen it thinks it "sees" the character-set from 4096 to 8191, regardless of what is really there. The same thing happens in block 2, from 36864 to 40959.
To tell the 64 which block to get data from, you use:

```
POKE 56578 . PEEK ( 56578 ) OR 3
POKE 56576 , ( PEEK (56576 ) AND 252 ) OR X
```

Where X can be:

```
X=0 : Block 3 : 49152 - 65535
X=1 : Block 2 : 32768 - 49151
X=2 : Block 1 : 16384 - 32767
X=3 : Block 0 :         0 - 16383
```

THE MEMORY MAP

BLOCK 0:	
0 - 703	System variables.
704 - 767	Free. (Sprite-block 11).
768 - 831	System variables.
832 - 1023	Only used during tape operation. (Sprite-block 13-15).
1024 - 2047	Normally used for screen and sprite pointers.
2048 - 16383	Free. Used for your BASIC program.
BLOCK 1:	
16384 - 32767	Free. Used for your BASIC program.
BLOCK 2:	
32768 - 40959	Free. Used for your BASIC program.
40960 - 49151	BASIC-ROM.
BLOCK 3:	
49152 - 53247	Free.
53248 - 57343	I/O-Chip.
57344 - 65535	KERNAL-ROM.

THE SPRITE POINTERS

Each sprite is made up of 63 bytes of data. In 16 Kb, there is room for 256 sprites. The sprite-blocks are numbered from 0 to 255. The formula for their starting address is: SPRITE-BLOCK*64.
Up to 8 sprites can be displayed at the same time. To tell the VIC which sprite to display, you POKE the number of the sprite-block into the sprite pointer.
The sprite-pointers are normally positioned from 2040 to 2047. 2040 is sprite 1, 2041 is sprite 2, and so on. If you move the screen around, the sprite pointers will follow. Their new address will be from: SCREEN+1016.
An example:
If you have stored the data from 704 to 766, and want to use sprite 2, you type:

```
POKE 2041, 704/64
```

USING YOUR SPRITES

If you have saved your sprites directly from memory, you just load them in with:
From tape: `LOAD"",1,1`
From disk: `LOAD "FILENAME",8,1`

To display them, simply POKE the sprite pointers with the sprite-blocks.

If you have saved your sprites as data-statements, you must first merge them to your program. (Explained under data statements). Then, in the program, they must be read from the data-statements into memory.
To read a sprite into a sprite-block, use:

```
FOR X = TO 62 : READ A : POKE SPRITE-BLOCK*64 + X, A : NEXT
```

MAKING ROOM FOR YOUR SPRITES

The easiest way to get more room for sprites, is to move the start of BASIC. To move it to the beginning of memory-block 1, you use:

```
POKE 43 , 1 : POKE 44 , 16*4 : POKE 45 , 3 POKE 46 , 16*4 [RETURN]
POKE 16*1024 , 0 : CLR [RETURN]
```

THIS WILL LEAVE THE SPRITE BLOCKS FREE.

SPRITE-BLOCK	ADDRESS
11	704 - 767
32 - 63	2048 - 4095
128 - 255	8192 - 16385

Now you have 24 Kb left for your BASIC program, which is sufficient in most cases. Also see the memory map.

OVERLAY COLOURS

Try to avoid using the same colour for two or more sprites in overlay mode. It is a good idea to pick out your 8 favourite colours and your favourite background, and try to keep them each time you edit sprites in overlay mode. This makes it easier for you to find the right colours for your previously saved sprites.

OVERLAY PRIORITIES

Sprite 39 has the lowest priority, and sprite 32 has the highest priority. You will not notice this in overlay mode, but in combination with normal mode, this is useful to know. Sprites can be copied into the overlay work-area, from 32 to 39. An example:

If you want to make some horses with saddles, for animation in overlay mode, it is recommended to make the horses in normal mode first, then move one of them into sprite-block 39, and clear the sprite-blocks from 32 to 38. If you enter overlay mode, the horse will be there as sprite 8. Make the saddle upon the horse as sprite 7. Go back to normal mode and examine sprite 38 to 39. The saddle will be in 38, and the horse will be in 39. If you move one of the other horses into 39, and enter overlay mode, you will see that the new horse has the saddle on it. This is because the saddle, number 38, has a higher priority than the horse, number 39.

I SHOULD NOT HAVE DONE THAT

Sooner or later you will accidently press the wrong key. A lot of the commands can easily be recovered from. For example the rotate command. Pressing R three more times restores your sprite. Several commands require parameters. By entering a number larger than 255, or just pressing [RETURN], the command will be aborted and no harm has happened. The same will happen with a blank filename for disk-operation.

THE COLOURS

There are 16 different colours.

0	Black
1	White
2	Red
3	Cyan
4	Purple
5	Green
6	Blue
7	Yellow
8	Orange
9	Brown
10	Light red
11	Dark grey
12	Light grey
13	Light green
14	Light blue
15	White grey

Each time you change the colour of something, the next colour is selected. This means that if you have a black sprite, number 0, and change the sprite-colour, the new colour will be white, number 1. After number 15, white grey, it will change to number 0, black.

THE JOYSTICK

The joystick must be plugged into port 2. Most people will prefer editing with a joystick in normal mode. Each time you press the button the pixel will be reversed. By holding the button depressed and moving in any direction, you will draw a line. In the position command, you can use the joystick for fast positioning. It can also be used for editing in multi-colour and overlay mode, but this is only recommended for joystick fanatics.

REFERENCE

M	Enter multi-colour mode
D	Enter display mode
O	Enter overlay mode
P	Position sprite
SPACE	Clear pixel
.	Fill pixel - normal
1-3	Fill pixel - multi
1-8	Fill pixel - overlay
F'2	Border colour - screen
F'4	Background colour - screen
F'6	Character colour - screen
F'8	Sprite colour - screen
F'1	Background colour - sprite
F'3	Sprite colour - sprite
F'5	Multi-colour ♯ 1
F'7	Multi-colour ♯ 2
SH. 1-8	Sprite colour - overlay
RETURN	Next line, first position
HOME	First line, first position
CLEAR	Clear sprite and home
DELETE	Delete pixel
INSERT	Insert pixel
UP	Move cursor up
DOWN	Move cursor down
LEFT	Move cursor left
RIGHT	Move cursor right

	:	Scroll line left
	;	Scroll line right
	↑	Scroll line up
	=	Scroll line down
	S	Scroll
	R	Rotate
	T	Turn horizontally
	U	Turn vertically
	I	Reverse
	C	Centre
	X	Vertical size
	Y	Horizontal size
	+	Next sprite
	—	Previous sprite
	G	Get sprite
SHIFT	C	Copy sprites
SHIFT	S	Save sprites
SHIFT	L	Load sprites
SHIFT	O	Oops!
	F	Force
	A	Animation
SHIFT	D	Make data-statements
SHIFT	N	New
SHIFT	F	Free
SHIFT	Z	Save data-statements
SHIFT	X	Load data-statements