Spooky Sprites

Johan Karlsson
Internet: d92jk@efd.lth.se

Version 4
August 11, 1995

Contents

1 Short Introduction

With this program you can create true colour sprites for the Atari Falcon. The stuff you require are a F030 with at least 4 megs of memory. It's now compatible with both screen enhancers, virtual memory and strange monitors. It still doesn't work with mtos, but then again what does?

If you have any questions, bug reports or suggestions for improvement then please send me an e-mail at this address d92jk@efd.lth.se.

2 How to use this program

2.1 Choosing colours

To pick a colour press the right button on the colour you want. You can do this anywhere but on the buttons. To create new colours double click with the left button on the palette you don't want anymore and you will come to the palette creation screen. The program uses all 65000 colours.

2.2 Put me down

When you have made your sprite you can put it down into the sprite storage area by pressing the put button (arrow down on the top row). To retrieve a stored sprite use the numeric keyboard keys 4 and 6 to scroll until the correct sprite is in the middle of the screen, then press the get button.

2.3 Saving sprites

Sprites may be saved in either uncompressed or compressed format. The sprites may also be saved as both formats at once. This is chosen per sprite so you might save some of your sprites uncompressed and some compressed and the rest both. If you want to change all sprites to the same format this can be done in the preferences meny. Selecting format for single sprite is done with the symbol in the bottom right corner of the button panel. If it says 1 then the current sprite will be saved in an uncompressed format, if it says 2 it will be saved in a compressed format and if it says 1&2 it will be saved in both formats. Notice that only sprites that have been put and show up in the bottom part of the screen will be saved.

The advantages of saving the sprites in compressed format is that they are drawn faster and if they contain lots of black they will require less disc space and memory than uncompressed sprites. The drawback is that if clipping is required the sprites they will take longer time to draw and if they don't contain any black then they will require more memory and disc space as well. Since the save method is chosen per sprite you can save the most important sprites in both formats e.g. the shots in a shoot-em-up game.

The sprites may also be saved as raw data. This makes the data very easy to use, but a serious drawback is that raw data cannot be reloaded because no information about sprite sizes etc. is saved. When saving sprites as raw data compression is never used.

2.4 Animation

To animate a sprite use the keys 4 and 6 to scroll to the first sprite in your animation sequence. Then click on the pacman button and your sprite will be animated. If the animation sequence isn't the one you want then make a right button click on the pacman and choose Change animation sequence.

2.5 Grabbing sprites from a picture

The grab a sprite from a picture you must first load a picture. When you have done that you may grab a sprite from it. When grabbing you shall click on the top left corner of the part you want to grab. In the preferences you might turn on a grid with an optional offset to make grabbing easier. This is all made from the file menu. The following picture formats are currently supported.

- **TGA** 24 bit type two Targa pictures.
- **TRP** 16 bit Falcon true colour picture.
- **Raw data** 16 bit Falcon true colour raw data.
- **NEO** 16 colours ST Neochrome picture.
- **PI1** 16 colours ST Degas Elite picture.
- **TRC** Don't know what this is, but it seems to work ok!
- **GOD** Godpaint 16 bit true colour picture.

All picture formats except Targa can be Ice packed (2.40). If you load raw data you have to enter the size of picture and the offset of the picture data from the start of the file in bytes.

3 How do I use the sprites in my own program

If you program in assembler then you can find a sprite drawing routine in falclib6. If you are using STOS then you could use Anthony Jacques' (jacquesa@cs.man.ac.uk) Falcon STOS extension, because it supports true colour Spooky Sprites. The assembler sprite drawing program is written in position independent code and could easily be used in other languages like GFA-Basic with very small modifications.

4 Keyboard commands

Note that the copy and paste commands can only be used from the keyboard. Flood fill may also be used by right clicking on the Fill button.

q	quit
p	preferences
x	fx
s	save sprites
l	load sprites
L	file menu
z	change size
c	copy
v	paste
V	set paste mode
b	paste onto picture
B	paste onto last grabbed place on the picture
a	animate
A	set animation mode
Undo	undo
4	select sprite to the left
6	select sprite to the right
1	select colour to the left
3	select colour to the right
r	grab sprite form picture
e	replace colour
f	fill
F	flood fill
d	dictionary
space	views picture

5 Sprite and picture formats

5.1 TRS, True Colour Sprites (version 1)

5.1.1 File header

4 bytes Ascii header 'TCSF'.
1 word Number of sprites.
1 word File version. This should be 1.
1 word Xres that the packed sprites are calculated for.

5.1.2 Sprite locations

This section is repeated number of sprites times.

1 byte Width in pixels.
1 byte Height in pixels.
1 long Offset to unpacked data from start of file or 0.
1 long Offset to packed data from start of file or 0.

5.1.3 Sprite data

This section is repeated number of sprites times.

If unpacked data is available then it's here.
width*height words of raw true colour data
If packed data is available then it's here
1 word Number of chunks
Start of compressed chunk (the chunk is repeated number of chunks times)
1 word Screen offset. Add this value to the screen pointer.
1 word Nbr of data words -1.
1*(nbr of data words) words of raw true colour data
End of chunk.

5.2 TRP, True Colour Picture

4 bytes ASCII header 'tru?'.
1 word picture width in pixels.
1 word picture height in pixels.
width*height words of picture data

5.3 TRE, Run length encoded True Colour Picture

4 bytes ASCII header 'tre1'.
1 word picture width in pixels.
1 word picture height in pixels.
1 long number of chunks.
This is followed by all the data chunks. The first chunk is a raw data chunk.

5.3.1 raw data chunk

1 byte Number of raw data pixels. If this byte is 255 the number of pixels will be the following word+255
Number of pixels words of raw data.
This chunk is followed by a rle chunk.

5.3.2 rle chunk

1 byte Number of rle pixels. If this byte is 255 the number of rle pixels will be the following word+255. This is the number of times you should draw the previous colour again.
This chunk is followed by a raw data chunk.

6 History

6.1 Version 4

The largest change is that you can use the sprite editor in double pixel mode (768*240) if you use a rgb monitor or tv. The video mode is changed in the preferences menu. Switching between ntsc and pal on rgb now works as it's supposed to do (you cannot do this on vga monitors). Doubling and halving of sprite width is now implemented. This allows you to convert medium resolution sprites to low res and vice versa. New paste mode Merge. Pictures can now be saved in a run length encoded format. Keyboard shortcut 'e' for replace colour. Faster sprite loading. Flood fill added. Can be installed as TRS application and load sprite files automatically. You may now create a picture containing all sprites. Some multitasking during the most time consuming operations. Lots of minor bugs removed.

6.2 Version 3

Some new file formats supported e.g. Neochrome and Degas Elite. A couple of bugs have been removed. New keyboard commands 'r' grabs a sprite from a picture and 'space' views a loaded picture.

6.3 Version 2

A new improved sprite grabbing. You can now load pictures that are larger than the screen and scroll around. Loading pictures doesn't crash the virtual memory any more. You can now paste sprites to pictures and save the pictures. Now it's possible to save the sprites as raw data and to load raw data pictures.

6.4 Version 1

The quite buggy first version.