

# **GFXMaster**

TIK/RETIRE

COLLABORATORS

	TITLE : GFXMaster		
ACTION	NAME	DATE	SIGNATURE
WRITTEN BY	TIK/RETIRE	August 6, 2024	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

# Contents

<b>1</b>	<b>GFXMaster</b>	<b>1</b>
1.1	GFX Master V1.02 Documentation . . . . .	1
1.2	General Info & Special Features . . . . .	1
1.3	History/Bugs . . . . .	2
1.4	Gadget Discription . . . . .	3
1.5	Load Gadget . . . . .	4
1.6	Load Cycle Gadget . . . . .	4
1.7	Save Gadget . . . . .	4
1.8	Save Cycle Gadget . . . . .	4
1.9	View Gadget . . . . .	5
1.10	Auto Show Gadget . . . . .	5
1.11	Screen Mode Lock Gadget . . . . .	5
1.12	Picture Width Gadget . . . . .	5
1.13	Picture Height Gadget . . . . .	5
1.14	Picture Color Gadget . . . . .	5
1.15	Screen Mode Gadget . . . . .	6
1.16	Screen Mode String Gadget . . . . .	6
1.17	Brush Save Gadget . . . . .	6
1.18	Brush Save Cycle Gadget . . . . .	6
1.19	Brush Integer Gadgets . . . . .	7
1.20	RawSave Gadget . . . . .	7
1.21	BlitWords Gadget . . . . .	7
1.22	ColorSave Gadget . . . . .	7
1.23	RawAlign Gadget . . . . .	7
1.24	Mask Pen Gadget . . . . .	8
1.25	Invert Mask Gadget . . . . .	8
1.26	Sprite Gadget . . . . .	8
1.27	ColorMode Gadget . . . . .	8
1.28	Palette Gadget . . . . .	9
1.29	Color Save Format Gadget . . . . .	9

---

1.30 Sprite Control Words Gadget . . . . .	9
1.31 Sprite Color Save Gadget . . . . .	9
1.32 RGB Chunky Save Mode Gadget . . . . .	9
1.33 RGB Chunky Transpose Gadget . . . . .	10
1.34 Saver Configuration . . . . .	10
1.35 Spare Page Gadgets . . . . .	10
1.36 Image Selection Gadget . . . . .	10
1.37 Frame Selection Gadget . . . . .	10
1.38 Overwrite Gadget . . . . .	10
1.39 ReqTools Gadget . . . . .	11
1.40 Iconify Gadget . . . . .	11
1.41 Message Box Gadget . . . . .	11
1.42 Slider Up Gadget . . . . .	11
1.43 Slider Down Gadget . . . . .	11
1.44 Window Close Gadget . . . . .	11
1.45 Window Zoom Gadget . . . . .	11
1.46 The Picture Screen . . . . .	12
1.47 Available Loaders . . . . .	12
1.48 Available Savers . . . . .	14
1.49 Multiselect Option . . . . .	17
1.50 File Suffix . . . . .	18
1.51 Preferences . . . . .	18

---

# Chapter 1

## GFXMaster

### 1.1 GFX Master V1.02 Documentation

ANOTHER FINE R E T I R E PRODUCTION WILL DEMOLISH YOUR HARDWARE

GFX MASTER V1.02, © 1994 by TIK/RETIRE

THE ULTIMATE CODER TOOL!

Yo, folx. This is another kind of GFX converter. Another one among dozens released in 1994. But a special one! Why? Use this wonderful piece of software and then you will find out the reasons --> ~Info/Features~~.

Choose one of the following topics for further information:

~History/Bugs~~~~	~Loaders~~~~~
~Window~Gadgets~~	~Savers~~~~~
~Picture~Display~	~Multiselect~~~~
~Preferences~~~~~	~File~suffix~~~~

This program is GIFTWARE which means if you use it send me a little donation (money & sources preferred). If you want the latest version or the source (partially) send me a disk AND a kind of gift. You want a special version with your own sub savers or anything else? No problem! Send some money and we will talk about it!!! New ideas and bug reports are also welcome!

So feel free and contact TIK of RETIRE:

Nico Schmidt
Möhlauer Str. 9
06773 Jüdenberg
GERMANY

Or send an EMail to: Nico.Schmidt@Student.Uni-Magdeburg.de

### 1.2 General Info & Special Features

---

To all normal people (non-coders):  
This tool is only a program which converts IFF (and other) pictures into the different formats used for creating games & demos and not a kind of GFX processing software like ADPRO where you can manipulate your mega-mighty true-color graphics.  
That's why this is a special coder tool.

But why another one? Try it and then you know the difference.  
And now some reasons for using THIS converter:

- Supports OCS/ECS/AGA chipset
- Style-guide user interface
- WB & CLI startup parameters
- Fully supported Drag 'n' Drop
- Always fast on 68000 Amigas
- Selectable ModeID of the GFX render screen
- Systemfriendly drawing functions (will work with graphic boards)
- Supports datatypes.library for loading "alien" pictures
- Multiple file loading feature
- Self defineable file extensions for each save format
- Advanced raw loader with auto-detect
- Special savers available, for example: RawBlit and Chunky
- Raw alignment for 2x/4x CAS mode
- Sophisticated sprite saver:
  - \* Supports 16/32/64 pixel sprites
  - \* Saves multiple sprites if brush size greater than sprite size
  - \* Saves sprite coordinates
  - \* Automatic sprite-control-word-calculation
- Selectable color bank and color offset in copper palette mode
- And many other cool things

But remember: The state-of-art software needs at least OS2.04!

## 1.3 History/Bugs

General limitations & bugs:

The detection of an 8-bit-per-gun color resolution CMAP isn't clear.  
The method I used is to check if EVERY CMAP byte has null as its low nibble. If so, the IFF pic contains (hope so) a 4-bit-per-gun CMAP.

Another fake operation is the identification of EHB/HAM pictures.  
I get these information from the lower word of the ModeID.  
RKM: "Never interpret these bits directly."  
But the original V37 display database knows nothing about EURO72, DBLPAL, ... and so it's impossible to get some info about pictures with an unknown ModeID. Unreasonable solution!!!  
But don't worry about this, there shouldn't be any problems.

It's possible to display a picture with 8-bit-per-gun CMAP on a 4-bit-per-gun graphics system (e.g. a 32 color AGA pic on ECS).  
That's why there may be some color faults!  
But this isn't a bug. It's a feature!!!

---

Note:

Do not open the ReqTools screen mode requester with reqtools.lib 38.1194. The requester probably crash!

I think the RT\_IntuiMSGFunc of rtScreenModeRequestA is buggy.

-----

History:

----- GFX Master V1.02 -----

- \* Released at THE PARTY IV in Herning on 28.12.94
- \* The first public release
- \* No special things to mention

----- GFX Master V1.10 -----

- \* Second public release on 27.5.95
- \* Completely new design
- \* Drag 'n' drop implemented
- \* Startup parameters possible
- \* Images now stored in FastMem -> speed!!!
- \* Some other features added
- \* Enhanced raw, palette & mask saver
- \* RGB chunky saver added
- \* HAM8 bug fix, modify planes are now plane 1 & 2
- \* And some minor bug fixes...

## 1.4 Gadget Discription

This is the list of all gadgets available in the main window.  
The corresponding key shortcut is shown in brackets.

Picture related gadgets:

~Load~~~~~ < L >	~LoadCycle~~~ < SHIFT L >
~Save~~~~~ < S >	~SaveCycle~~~ < SHIFT S >
~View~~~~~ < V >	~ScreenMode~~ < M >
~ModeString~~ < SHIFT M >	
~Show~~~~~	~PicWidth~~~~ < W >
~Config~~~~~ < O >	~PicHeight~~~ < H >
~Lock~~~~~	~PicColors~~~ < C >

Brush related gadgets:

~Save~~~~~ < A >	~SaveCycle~~~ < SHIFT A >
~Config~~~~~ < N >	~BrushValues~ < B >

Remaining gadgets:

~SparePage~~~	~MessageBox~~ < DEL >
~Image~~~~~	~SliderUp~~~~ < CursorUp >
~Frame~~~~~	~SliderDown~~ < CursorDown >

```
~Overwrite~~~                               ~CloseGadget~ < ESC >
~ReqTools~~~~~                             ~ZoomGadget~~ < I >, < RMB >
~Iconify~~~~~
```

## 1.5 Load Gadget

This function opens a file requester (what else?). Now you can select your picture or with the ~Multiselect~ feature a couple of pictures you want to load. Depending on the choice of the ~LoadCycle~ gadget the desired part of the picture will be loaded. For further descriptions look at ~Loaders~.

## 1.6 Load Cycle Gadget

With this button you can choose the part of the picture you want to load:

Picture - the entire picture (BODY + CMAP)

Palette - only the colors (CMAP)

Body - only the bitmap (BODY)

## 1.7 Save Gadget

If you press this button a file requester appears. Now choose a filename or if you are in ~Multiselect~ mode choose a destination directory. Depending on the choice of the ~SaveCycle~ gadget the picture will be saved in the desired format (see also ~Savers~). Saving of an additional palette file is also possible (~ColorSave~).

## 1.8 Save Cycle Gadget

Here you can choose the appropriate picture ~Saver~:

IFF - the picture will be saved as IFF ILBM file

Palette - a palette file will be generated depending on ~ColorMode~ and ~Palette~ gadget

Raw - raw picture data will be saved depending on ~RawSave~, ~RawAlign~, ~BlitWords~ and ~ColorSave~ gadget

Mask - a picture mask will be generated depending on ~RawSave~, ~RawAlign~ and ~BlitWords~ gadget

---



xRGB - RGB chunky data will be saved according to ~SaveAs~ and ~TransposeImage~ gadget

## 1.9 View Gadget

If your hardware is powerful enough a screen will be opened and the loaded picture will be displayed. Now select your brush, look at your lousy picture or do whatever you want.  
For more information look at ~PicDisplay~.

## 1.10 Auto Show Gadget

Enable this option and every picture you load will be displayed immediately.

## 1.11 Screen Mode Lock Gadget

It's easy to explain:  
Enable this gadget and choose your favourite screen mode.  
Whenever this gadget is checked all pictures are displayed with your selected ModeID.

## 1.12 Picture Width Gadget

Use this gadget to change your current picture width.

But remember: Pixels are added/removed without scaling!

## 1.13 Picture Height Gadget

Use this gadget to change your current picture height.

But remember: Pixels are added/removed without scaling!

## 1.14 Picture Color Gadget

Use this gadget to change the picture depth/color format.  
Valid inputs are: \* 2, 4, 8, 16, 32, 64, 128, 256  
\* HAM6, HAM8  
\* EHB (correctly EHB to 64 color conv.)

But remember: Planes are added/removed from picture without any color manipulation routines.

---

## 1.15 Screen Mode Gadget

A screen mode requester appears and you can choose the ModeID for the picture. With the width and height values you specify the page size for the picture. These values will be truncated to values between nominal and max overscan screen size of the selected screen mode.

## 1.16 Screen Mode String Gadget

This gadget displays the screen mode of the picture. It is displayed as a namestring (if any found in the Display DataBase) or as ModeID.

It's also possible to change the screen mode by typing-in the namestring (case sensitive!) or just "ModeID xxxxxxxx". (xxxxxxx is the ModeID defined in 'graphics/modeid.i')

## 1.17 Brush Save Gadget

This gadget works like the ~Save~ gadget but all actions are brush related.

## 1.18 Brush Save Cycle Gadget

Here you can choose the appropriate brush ~Saver~:

- IFF - the selected brush will be saved as IFF ILBM file
  - Raw - raw brush data will be saved depending on ~RawSave~, ~RawAlign~, ~BlitWords~ and ~ColorSave~ gadget
  - Mask - a brush mask will be generated depending on ~RawSave~, ~RawAlign~ and ~BlitWords~ gadget
  - Sprite - the selected brush will be saved as sprite data depending on ~Sprite~ and ~SpriteCoords~ gadget, if the brush is wider than current sprite width several sprites will be generated
  - SpColor - a sprite palette file will be generated depending on ~ColorMode~ and ~Palette~ gadget
  - xRGB - RGB chunky data will be saved according to ~SaveAs~ and ~TransposeImage~ gadget
-

## 1.19 Brush Integer Gadgets

The first two integer gadgets specify the top-left position of the brush. The other two gadgets are used for brush width and height.

## 1.20 RawSave Gadget

With this knob you can select the mode, how planes of raw pictures/brushes should be saved:

Normal - RawBlitt - Chunky - Image => look at ~Savers~

## 1.21 BlitWords Gadget

This option will add a word to the left or right side of each row in Raw/RawBlit mode, if you want to shift the planes with the blitter. This option will be ignored by the chunky saver. The raw alignment (if any) will be done after this operation.

## 1.22 ColorSave Gadget

If you want to save a raw picture/brush with this gadget you can select how the palette information should be saved:

None - nothing special happens

Extern - a separate palette file will be generated with the appropriate ~FileSuffix~

Before - palette will be linked with raw data, before all planes

Behind - palette will be linked with raw data, behind all planes

## 1.23 RawAlign Gadget

This button is only useful, if you save raw picture/brush data for AGA machines. If you want to display a picture in a "Fast-DMA" fetch mode (2x/4x CAS), your planes must be long/double aligned. Here you can select, how long a row should be. If necessary some zero bytes will be added to each row.

This works also in Chunky mode but it isn't so useful (only for performance reasons => long/double mem access).

Byte - rarely used, but should be useful for chunky images

Word - standard, each row will be an even number of bytes long,

this is also true for the chunky saver!!!

Long - each row will consists of a # of bytes even by 4

Quad - each row will consists of a # of bytes even by 8

## 1.24 Mask Pen Gadget

Normally this value is set to -1 to save a 'standard' mask which means all colors different from the background color are taken into account while saving the mask plane.

Choose another pen (0-255) to save your specific mask.

## 1.25 Invert Mask Gadget

Set this flag to invert the mask plane. This may be useful if you want to use the CPU for mask operations instead of the blitter.

## 1.26 Sprite Gadget

Here you can select the save mode of your sprite data:

ASM 16/32/64 - 16/32/64 pixel sprite as assembler source code

BIN 16/32/64 - 16/32/64 pixel sprite as binary data

For further details look at ~Savers~.

## 1.27 ColorMode Gadget

Here you can specify the format of the separate palette file:

IFF - a DPaint like IFF ILBM palette

Copper - palette for copper usage

Data - pure data palette information

LoadRGB - LoadRGB palette (look at graphics autodoc)

If you need more information about file formats read ~Savers~.

---

## 1.28 Palette Gadget

This gadget allows you to select the precision of your color tab:

OCS/ECS - 12 bit per RGB, max. 32 colors

AGA 12 - 12 bit per RGB, upto 256 colors

AGA 24 - 24 bit per RGB, upto 256 colors

## 1.29 Color Save Format Gadget

Toggle between ASM source and binary saver.  
But remember: IFF is always binary!!!

## 1.30 Sprite Control Words Gadget

Select sprite control word generation:

Empty - empty control words are generated

Auto - see below

None - no control words are generated

Automatic sprite control word calculation:

Sprite coordinates will be calculated and saved with each sprite. If you select the brush via the integer gadgets the top-left (X,Y) position will be used as sprite coords. If you select the brush from the ~PicDisplay~ you may move the brush to the position you want. The sprite coords resolution depends on the screen page size, e.g. if page size 640x512 the sprite X coord will be saved as a highres pixel value. The sprite position for a "top-left" sprite on a screen with nominal size (320x256, 640x512, ...) will be X=\$40, Y=\$2C (for SHRES X=4\*\$40). If you use a screen with overscan page size the sprite coords will be adjusted. If you save several sprites from one brush, the X coord will be increased depending on the current sprite size and X resolution (page size).

## 1.31 Sprite Color Save Gadget

Enable this gadget to save a separate sprite color file with the appropriate ~FileSuffix~.

## 1.32 RGB Chunky Save Mode Gadget

Toggle between 12 and 24 bit RGB chunky mode.

---

### 1.33 RGB Chunky Transpose Gadget

Enable this gadget to save your RGB chunky pixels in vertical order (picture matrix will be transposed).

### 1.34 Saver Configuration

This button is used to displays the configuration window according to the selected saver:

IFF saver	-	no configuration
Palette saver	-	~SaveMode~ ~Palette~~~~ ~SaveFormat~
Raw saver	-	~SaveMode~ ~BlitWords~~ ~ColorSave~~ ~Alignment~
Mask saver	-	~MaskPen~~ ~InvertMask~
Sprite saver	-	~SaveMode~ ~CtrlWords~~ ~ColorSave~~
SprCol saver	-	~SaveMode~ ~Palette~~~~ ~SaveFormat~
xRGB saver	-	~SaveAs~~~ ~Transpose~~

### 1.35 Spare Page Gadgets

This is an internal image buffer. May be used in future for composing two images (May be?!? Or not?!? Or what?!?).

To	-	copy current picture into buffer
From	-	restore buffered picture
Swap	-	toggle between buffered & current picture
Clear	-	trashes the buffered picture
		hold down < SHIFT > to clear buffer AND current picture

### 1.36 Image Selection Gadget

Not yet implemented.

### 1.37 Frame Selection Gadget

Not yet implemented.

### 1.38 Overwrite Gadget

If this option is enabled every existing file will be overwritten. If not, a requester will appear with 3 choices: New name/Overwrite/Skip. An existing directory can't be deleted, so you have only 2 choices.

### 1.39 ReqTools Gadget

Choose the requester you prefer (ASL/ReqTools).

### 1.40 Iconify Gadget

Here you can choose your preferred iconify mode:

Window - use shrink window  
Icon - use AppIcon  
Menu - use AppMenuItem

### 1.41 Message Box Gadget

Here are all information displayed you need. Press the < DEL > button to clear the list.

### 1.42 Slider Up Gadget

Nothing to explain. Use < SHIFT CursorUp > to scroll faster.

### 1.43 Slider Down Gadget

Nothing to explain. Use < SHIFT CursorDown > to scroll faster.

### 1.44 Window Close Gadget

Exit or what?! Yo, but if you want to save the prefs hold down < SHIFT > while exiting.

### 1.45 Window Zoom Gadget

Press the zoom gadget to shrink the window.  
The I- or Right-Mouse-Button turns the program into iconify mode according to the ~Iconify~ gadget.

---

## 1.46 The Picture Screen

Well, on this screen your picture will be displayed. What else?

Brush selection:

And now you can select your desired brush. Take your mouse and select the brush via the left button (DPaint like). If a brush is already selected you can move the border to a new position. Use the left mouse button to fix/unfix the brush position. This is useful for cutting several brushes with the same size. Use the right button to unselect current brush. To make the cutting easier hold down < SHIFT >. If you vertically move your mouse the horizontal movement is disabled and vice versa. Another cool option is the color selection of the brush border and the "target" lines. Try < CursorLeft > and < CursorRight >.

Sprite coordinates:

If you exit from the pic display and the brush position is fixed the sprite coords are equal to the top/left brush coords. Unfix the brush (left button) and move it to the desired position. Now it is unfixed (don't press left button again!). On exit these coords will be used as sprite coords. Remember: The X coordinate depends on the current page size. This means shires coords on a screen with  $x \geq 1024 - 128$ , hires on a screen with  $x \geq 512 - 64$  (converted to shires for saving) and lores coords on smaller screens ( $\times 4$  to get shires for saving). On overscan screens the sprite coords will be automatically adjusted. The nominal 0-position for sprites will be \$100,\$2C (shires).

The pointer:

It's used for displaying the brush position & size. The first 2 numbers represent the current pointer position or the top/left position of a fixed brush counted in pixels (x,y). Number 3/4 will display the current brush width/height. If you hold down < ALT > the 4th number displays the current word-width and the 3rd the "pixel-rest". Press < CAPS LOCK > to toggle between original pointer colors and colors used by the picture. Another (DPaint like) feature is the function of < DELETE >. Use < SHIFT > in addition to get all possible pointer states.

By the way, close this screen with < SPACE >, < ESC > or middle mouse button.

## 1.47 Available Loaders

Currently are 3 different loaders available plus loading via startup parameters and drag 'n' drop:

IFF ILBM loader:

The loader scans all IFF files for appropriate ILBM chunks. This means you can also load the 1st frame of an IFF animation. The usage of the `iffparse.library` and the unpacking into chip

---



mem will slow down the loading operation. But this is the most compatible way to load IFF files without a large amount of mem. If your picture isn't an IFF file the next loader will be called.

#### Datatypes loader:

This loader needs the datatypes.library. That's why it doesn't work on systems without OS 3.0.

Now you have to install the desired picture datatypes and you are able to load all wanted pictures. If the loader doesn't recognize the filetype the last loader will be called.

#### Raw loader:

This loader is called if you try to load a unknown picture type.

Now you have to decide to load this as raw picture data.

But first you have to set up some picture attributes:

Width - The desired picture width

Height - Height of pic do you want

Offset - Number of bytes to skip in file before fetching data for CMAP and planes (limited to file size)

Modulo - Number of bytes to skip at the end of each row before fetching data for the next row

Depth - The number of planes the pic contains and special modes  
EHB: 6 planes, HAM6/8: 6/8 planes

Note: If you load a chunky raw picture (8 bit/pixel)  
here you can select the number of planes for the  
destination picture (other bits are truncated)

RawMode - Normal planes/Interleaved planes/Chunky data

CMAP - If your raw picture contains palette information choose the appropriate mode

Note: Chunky pic: always 256 colors

HAM6/8: 16/64 colors unlike some other converters

EHB: 32 colors, what else? 64 is quite nonsense!

Palette - Select the size of each color entry (1 or 2 words)

The appropriate key shortcuts of the gadgets are underlined.

The "Calculated Size" reflects your settings. So it's easier to find the correct settings for an "unknown" raw picture.

Don't worry if size of your selected picture greater than file size. The last planes or the CMAP behind will be filled with 0's.

Dependent on the ~LoadCycle~ gadget the desired part of the picture will be loaded. That's why it's possible to load only a CMAP from a raw picture, or only the planes, or both.

If you save a raw picture/brush or a binary palette file you haven't to set up the attributes. It's done automatically by the loader. The ModeID, number of planes, CMAP and all the other things are coded and added to the file comment.

#### Startup parameters:

---

CLI start - add all pictures you want to load to the command line,  
 pattern matching is supported  
 WB start - use multiselection on startup, or use GFX Master as  
 the default tool, or ...

Drag 'n' drop:

All Workbench drag 'n' drop features are now implemented.  
 Checkt it out, it works!!!

## 1.48 Available Savers

Here comes a list with all available savers and save formats.

Picture/brush IFF ILBM saver:

I think it's clear. The IFF file will consist of a BMHD, CMAP, CAMG and BODY chunk. In HAM mode the CMAP contains "only" 16 or 64 (HAM8) colors. More aren't neccessary! I decided to save 64 colors in EHB mode, 'cause some PC ILBM viewers crashes with 32 color CMAP (did't know anything about EHB, or what?). All pictures will be saved with an 8-bit-per-gun CMAP. The BODY data is compressed via ByteRun1 algorithm.

Picture/brush RAW saver:

Currently are 4 different raw savers implemented. Choose the appropriate saver with the ~RawSave~ gadget. The ~RawAlign~ and ~BlitWords~ gadgets may influence the saving operation. The storage of the palette information depends on the the ~ColorSave~ gadget. Look at "Palette saver" for a detailed description.

Normal raw saver: row1plane1, row2plane1, row3plane1, ...  
 row1plane2, row2plane2, row3plane1, ...

RawBlitt saver: row1plane1, row1plane2, row1plane3, ...  
 row2plane1, row2plane2, row2plane3, ...

Chunky raw saver: chunkyrow1, chunkyrow2, chunkyrow3, ...

A chunky row consists of an EVEN number of bytes. Each byte represents a 256-color-pixel. This is useful for machines with hardware chunky2planar converter (like the AKIKO of the CD32) for fast texture mapping.

Image saver: An Intuition Image structure will be generated before the plane data (normal raw data).  
 But remember: Byte alignment isn't possible!

Picture/brush MASK saver:

This saver generates a picture/brush mask. Mask generation depends on the settings of the ~MaskPen~ & ~InvertMask~ gadgets. Also check out the ~RawSave~, ~RawAlign~ and ~BlitWords~ gadgets.

Normal raw mask: row1, row2, row3, ...

```
RawBlitt mask:    row1, row1, row1, ... n times
                  row2, row2, row2, ... n times
                  row3, row3, row3, ... n times, n-picture depth
```

Image mask: Normal raw mask with an Intuition Image header

Saving a chunky mask is a senseless operation.

#### Picture/brush RGB chunky saver:

Here is a saver for generating 12/24 bit chunky truecolor pictures. Often used for 'Copper'-Rotators and other 'chunky' effects. Choose the depth (12/24) with the ~SaveAs~ gadget.

Switch on the 'transpose' mode to save the pixel data in vertical order (~TransposeImage~).

```
12 bit:  DC.W    RGB01, RGB02, RGB03, ...
```

```
24 bit:  DC.W    RGB01, rgb01, RGB02, rgb02, RGB03, rgb03, ...
```

#### SPRITE saver:

The selected brush will be saved as sprite data depending on the ~Sprite~ gadget. If your brush width larger than current sprite width several sprites are generated: upto 8 sprites, in 16 color mode max. 4 sprites. Automatic sprite color generation is switched on with the ~ColorSave~ gadget. Other settings will be ignored.

Note: Sprite saving works only in 4/16 color mode.

```
16 pixel:  ctrl1, ctrl2                ctrl1/2 - sprite control words
           dat1a, dat1b                datxa  - data word, plane1
                                   datxb  - data word, plane2
           $0000, $0000
```

```
32 pixel:  ctrl1, ctrl2, ctrl2, $0000
           dat1a, dat2a, dat1b, dat2b
           $0000, $0000, $0000, $0000
```

```
64 pixel:  ctrl1, ctrl2, ctrl2, $0000, ctrl2, $0000, $0000, $0000
           dat1a, dat2a, dat3a, dat4a, dat1b, dat2b, dat3b, dat4b
           $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000
```

For a 16 color sprite 2 structures are generated. The second holds plane 3 and 4 and the attach bit of ctrl2 is set.

It's possible to save sprites with the belonging coordinates.

Enable these option with the ~SpriteCoord~ gadget.

How to set up these coordinates? Look at ~Picture~Display~.

The control words will be automatically calculated. If you save sprites as ASM source you can easily change the coords.

The generated sprite source looks like this:

```
X          SET $0100                (x coordinate, superhires)
Y          SET $002C                (y coordinate)
CW1        SET $FFFF&(Y<<8+X>>3)
```

```

CW2      SET $FFFF&(Y+$0001)<<8+68&(Y>>6*9)+34&((Y+$0001)>>7*9)
                                                +4&X/4+3&X*8
DC.W     CW1,CW2
DC.W     dat1a,dat1b
DC.W     $0000,$0000

```

#### Picture/brush PALETTE saver:

If you save raw pictures/brushes it's possible to save an additional (external) palette file automatically. Another possibility is to link a (binary/image) palette with the raw data, before or behind the planes. All what you have to do is to select the desired option with the ~ColorSave~ gadget.

Choose your desired palette format with the ~ColorMode~ and the ~Palette~ gadget.

Use the ~SaveFormat~ gadget to save ASM source or binary files.

Note: The number of colors depends on the number of bitplanes.

Exceptions: EHB - 32 colors

HAM6/8 - 16/64 colors (NOT 256 colors!)

Special ECS Productivity/Superhires color format not supported! Who need this? (My Worbench on A2000)

IFF palette: This is the standard DPaint palette file:

BMHD, CMAP & CAMG chunk

Note: Always saved with 8-bit-per-gun!

#### ECS/OCS copper palette:

```

CO      SET $0180                                (e.g. 32 colors)

DC.W    CO+00, RGB01, CO+02, RGB02, CO+04, RGB03, ...
DC.W    ...           , CO+60, RGB31, CO+62, RGB32

```

#### AGA 12 bit copper palette:

```

CO      SET $0180
CB      SET $0                                (e.g. 64 colors)

DC.W    $0106, (CB+0)<<13+$000
DC.W    CO+00, RGB01, CO+02, RGB02, CO+04, RGB03, ...
DC.W    ...           , CO+60, RGB31, CO+62, RGB32
DC.W    $0106, (CB+1)<<13+$000
DC.W    CO+00, RGB33, CO+02, RGB34, CO+04, RGB35, ...
DC.W    ...           , CO+60, RGB63, CO+62, RGB64

```

#### AGA 24 bit copper palette:

```

CO      SET $0180
CB      SET $0                                (e.g. 64 colors)

DC.W    $0106, (CB+0)<<13+$000
DC.W    CO+00, RGB01, CO+02, RGB02, CO+04, RGB03, ...
DC.W    ...           , CO+60, RGB31, CO+62, RGB32
DC.W    $0106, (CB+1)<<13+$000
DC.W    CO+00, RGB33, CO+02, RGB34, CO+04, RGB35, ...
DC.W    ...           , CO+60, RGB63, CO+62, RGB64
DC.W    $0106, (CB+0)<<13+$200
DC.W    CO+00, rgb01, CO+02, rgb02, CO+04, rgb03, ...

```

```

DC.W      ...      , CO+60, rgb31, CO+62, rgb32
DC.W      $0106, (CB+1)<<13+$200
DC.W      CO+00, rgb33, CO+02, rgb34, CO+04, rgb35, ...
DC.W      ...      , CO+60, rgb63, CO+62, rgb64

```

RGBxx - color word, upper 4 bit from 8-bit-per-gun color  
 rgbxx - color word, but the lower nibbles

If you want to shift your palette to another color position the only thing you have to do is to change the color offset (CO). e.g. You want to set the colors from \$188 to \$196: Create an 8 color copper palette and set the color offset to \$188. It's also possible to put colors into the desired color bank (only AGA, what else?). Change the CB (color bank) value.

Pure Data palette:

```

12 bit:  DC.W      RGB01, RGB02, RGB03, ...
24 bit:  DC.W      RGB01, rgb01, RGB02, rgb02, RGB03, rgb03, ...

```

This kind of palette is also used to link with the raw picture/brush data, before or behind the planes.

Note: The linked CMAP of a CHUNKY raw picture contains ALWAYS 256 colors (12 or 24 bit), regardless of the source pic depth!

LoadRGB palette:

```

12 bit:  LoadRGB4  format
24 bit:  LoadRGB32 format

```

Look at the Autodocs for further information.

SPRITE COLOR saver:

It's almost the same like the picture/brush palette, but:

- the saver works only in 4/16 color mode
- the default CO value in copper mode is \$1A0 (instead of \$180)

Hint: Use the copper save mode. So it's easy to arrange the sprite colors for odd & even sprites on AGA machines with the appropriate CO/CB (color offset & base) settings.

## 1.49 Multiselect Option

It's possible to process several files. At first select all files you want to load. The first picture will be loaded. On error the next picture is loaded. Now you have to set up the appropriate gadgets. At least you have to press the save button you want. A directory requester appears where you have to choose the destination directory for the processed files. And now the processing starts. It's possible to stop the action. But beware: This cancels the multiselection! You have to select all remaining files again.

## 1.50 File Suffix

All output files will be saved with a file extension. It's possible to select your own extension. Append your desired file extension at the filename in the save file requester. But remember: Only 3 chars! If your extension contains more or less chars no file extension for this filetype will be used. To save files without a extension, simply remove it.

Now you have to save the prefs.

Note: All 3-char-extensions will be automatically removed from the loadname. So it could be a problem if your loadname is "Test.123". Your savename will be, for example, "Test.raw".

The default suffixes:	IFF picture	- .pic
	IFF brush	- .bru
	Raw picture/brush	- .raw
	Raw picture/brush mask	- .msk
	Sprite	- .spr
	Palette files	- .col
	RGB chunky	- .rgb

## 1.51 Preferences

If you exit this wonderful tool while pressing the < SHIFT > key the following settings will be saved:

- current main window position
- settings of the cycle/checkbox gadgets from main window
- file/screenmode requester position & size
  - Note: You have to specify the position/size with the ASL requesters (impossible with ReqTools). After that you can switch back to ReqTools (or not).
- contents of the pattern gadget for:
  - \* load file req
  - (max. 63 chars)      \* pic save file req
  - \* brush save file req
  - \* color save file req
  - \* raw save file req
  - \* mask save file req
  - \* sprite save file req
  - \* xRGB save file req
- file suffixes for:
  - \* pictures
  - (0 or 3 chars)      \* brushes
  - \* raw pictures/brushes
  - \* raw masks
  - \* sprites
  - \* color files
  - \* xRGB files
- settings of raw load window
- locked ModeID
- load/save path