WarpPNG

Oliver Roberts

COLLABORATORS

	TITLE : WarpPNG	

ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Oliver Roberts	July 31, 2024	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

WarpPNG

1.1 WarpPNG.datatype 44.19

```
WarpPNG.datatype 44.19 - the fastest 24-bit PNG picture datatype!
                         (for 68k, PPC/WarpOS and PPC/MorphOS)
```


1.2 Description

As you've probably guessed, WarpPNG.datatype is a yet another PNG datatype. The original difference being that it was for owners of PPC cards, and was targetted specifically at WarpUp (not PowerUp). Even better, it is fast, compact, clean, well behaved and fast - a true plug'n'play PPC datatype. Now though, 68k optimized and MorphOS native versions have been added, which also benefit from my WarpDT engine. One of the key features is its superior speed, hence the name WarpPNG.

I originally decided to write this datatype because no other decent WarpOS native PNG datatype existed for the PPC, and it is was relatively easy to implement being based on the same engine that WarpJPEG.datatype uses.

1.3 Features

```
 - Supports both normal and interlaced PNG files

 - Gamma correction options allow the output image to be changed to
   suit your monitor/environment

 - Two different modes (combine or strip) for handling images that contain
   an alpha channel

 - Highly optimized datatype dispatch engine, and fast PNG decoder,
   resulting in a very efficient, compact and quick PNG datatype

 - Asynchronous file i/o and double buffering techniques (WarpOS only),
   which speeds up image decoding

 - Optimized versions for 68020, 030, 040 and 060

 - PowerPC support with native WarpOS and native MorphOS versions

 - Alter the pen allocation precision when images are remapped to an
   8-bit display

 - Specific support for the OS 3.5/3.9 picture.datatype, when available

 - The dithering feature of the OS 3.5/3.9 picture.datatype can be
   configured to your liking (e.g. disabled for 15/16-bit displays)

 - Based on libpng 1.0.12 and zlib 1.1.3
```

1.4 System Requirements

```
This datatype needs the following in order to work:

 - Kickstart 3.0 or higher
 - picture.datatype v43 or higher
   (i.e. either of the ones supplied with AmigaOS 3.5/3.9, P96 or CGraphX)

 68k version

 - 68020 processor or higher (optimized versions included)

 WarpOS version

 - PPC accelerator card + 68040/060
 - WarpUp Release 4 (powerpc.library V15) or higher

 MorphOS version

 - PPC accelerator card
 - MorphOS beta release 1 or higher

Note that a graphics card is not necessary - the P96 or OS 3.5/3.9
picture.datatype will automatically dither images down to your native Amiga
```

display requirements.

1.5 Installation

To install WarpPNG.datatype, simply run the provided installer script by double-clicking the icon. This will create a CPU specific version of the library for you, and install it to SYS:Classes/Datatypes and also installs the PNG descriptor file to DEVS:Datatypes. It does nothing else, so you needn't worry about it messing about with your system :)

1.6 Preferences Options

WarpPNG can be configured using the "Datatypes/WarpPNG.prefs" environment variable. The preferred way of altering the settings is via the WarpDTPrefs graphical user interface (requires OS3.5 or higher), available from http://www.nanunanu.org/~oliver/warpdtprefs.html or Aminet (util/dtype/WarpDTPrefs.lha).

However, it is also possible to alter the settings manually by changing the environment variable with setenv, according to the following template:

```
DITHER_OVERRIDE/K,DITHER_QUALITY/K,DITHER_DEPTH/K/N,PENS_OVERRIDE/K,
PENS_QUALITY/K,GAMMA_CORRECTION/K/T,SCREEN_GAMMA/K/N,IMAGE_GAMMA/K/N
ALPHA_MODE/K
```

OS3.5 DITHER CONTROL OPTIONS

The three dither options all interact with each other and apply only when at least picture.datatype V44 (OS3.5/3.9) is in use.

By default picture.datatype V44+ dithers images for all target displays less than 24-bit. It is debatable whether it's worth using dithering on 15/16-bit displays - for photographic images, the difference is usually not noticable, but for computer generated images containing smooth colour gradients dithering produces a noticably better output image.

DITHER_OVERRIDE

Selects what our dither quality setting should override:

```
NOTHING    - do not change the dither quality in any circumstances
DEFAULTS   - only override picture.datatype's default dither quality,
             which is used when the application does not specify it's
             own dither quality value
APPS       - override the dither quality that an application may set
EVERYTHING - always use your custom dithering settings
```

DITHER_QUALITY

Select the dither quality to use when this datatype decides that changes to the dither quality are required, according to your dither override and

depth settings:

  POOR - simple colour remapping and no dithering (fastest)
  GOOD - slower, higher quality dithered output (default)
  BEST - similar to GOOD, with maybe slightly higher quality output

DITHER_DEPTH

Use this option to select which display depths should use the above quality setting. When the target display is deeper than the given depth value, the above dither quality will be applied to that image. For example setting this to 8 will cause the dithering settings to be used for 15/16-bit target displays, but left untouched when the target display is <= 8-bit.

PEN SELECTION OPTIONS

The pen selection options are only relevant to 8-bit displays, and adjust the precision to which pens/colours are allocated, which affects the image quality and number of pens that will get used.

PENS_OVERRIDE

Selects what our pen/colour quality setting should override:

  NOTHING   - do not change the pen precision under any circumstances
  DEFAULTS   - only override picture.datatype's default pen precision
               quality, which is used when the application does not
specify
               it's own pen precision value
  APPS   - only override the pen precision that an application has set
  EVERYTHING - always use your pen precision settings

PENS_QUALITY

Changes the pen precision quality to use when remapping an image to an 8-bit display, in situations according to your pens otherride setting:

  POOR - use only a small amount of unique colours, at the expense of
         image quality
  GOOD - allocate enough pens to ensure a reasonable output image
         quality, whilst not hogging all pens for a single image (default)
  BEST - allocate as many pens as possible/necessary, resulting in the
         best image quality, but when displaying multiple images on the
         same screen, the quality of every image might not be so good

GAMMA CORRECTION OPTIONS

The following options allow you to control the gamma correction feature:

GAMMA_CORRECTION

  ON - enable gamma correction (default)
  OFF - disable gamma correction support completely

SCREEN_GAMMA

Specify the gamma of your display/screen - all images will have their gamma corrected to match this option. Generally, this value should be altered to match your monitor and display environment (e.g. bright room, sunlight, dark room, etc). If you set the screen gamma too low, images will appear too dark - likewise, images will appear too light if you set it too high. The default gamma value is 2.2, although the actual value you must give is the gamma value multiplied by one million - e.g. set this option to 2200000 for a gamma of 2.2.

IMAGE_GAMMA

PNG files may well contain a gamma value, which defines the gamma of that image. In this case, the file gamma will always be used. But, for images that do not define a gamma value, you can use this option to change the fallback image gamma. The default value of 0.45455 should be sufficient in most cases. As with the screen gamma, the gamma value must be multiplied by one million (e.g. 0.45455 -> 454550).

MISCELLANEOUS OPTIONS

ALPHA_MODE

For images which contain an alpha channel, specify how this should be handled:

  USE - the alpha channel will be combined with the background and image, which is the standard way of handling the alpha channel (default)
  DISCARD - completely ignore the alpha channel, allowing you to see the image as-is. This is useful for PNG images containing a fully transparent alpha channel, that have been created with buggy programs, which would otherwise result in a blank image being shown. Of course, images that contain a meaningful alpha channel will not be shown properly using this option.

1.7 Speed

The fastest PNG datatype?

The table below shows the time (in seconds) it took to decode 14 different images on my A1200 603e/240MHz 060/50MHz, with BVision and CGX picture.datatype, with other tested datatypes configured as close as possible to WarpPNG's internal settings. However, I should emphasize that the figures above are for comparison purposes only - the actual values are likely to be slightly different (faster or slower) on your system, but I'm confident that the WarpOS version of WarpPNG is noticably faster than all alternative PPC datatypes.

The figures speak for themselves... WarpPNG for WarpOS is at least 75% faster than other PPC datatypes in most cases (100-300% faster with palette based and/or smaller files), and around 3 times faster than 68k datatypes on a 060 (the speed difference will be even greater on 040 systems with a fast PPC). Maybe you don't believe these claims - my answer to that is simple... Try WarpPNG for yourself and you will see the difference! :)

		PowerPC/68K (dual-cpu)		
		WarpOS		PowerUp
		WarpPNG 44.19	akPNG 44.112 PPCLibEmu	PowerUp native
	bytes			
1024x768 24-bit,	833721	1.22	2.06	2.26
1024x768 24-bit int,	1037008	1.59	2.44	2.74
779x767 24-bit,	946564	1.06	1.82	2.11
473x639 24-bit,	377518	0.52	0.91	0.98
473x639 24-bit int,	459869	0.66	1.06	1.17
368x463 24-bit,	279896	0.36	0.64	0.70
251x400 24-bit,	136856	0.23	0.43	0.42
718x425 grey,	157890	0.25	0.49	0.51
718x425 grey int,	182522	0.29	0.55	0.57
1024x768 256 col,	324599	0.40	0.77	0.84
779x767 256 col,	404245	0.39	0.78	0.81
473x639 256 col,	163397	0.23	0.46	0.44
368x463 256 col,	115942	0.17	0.34	0.32
251x400 256 col,	57056	0.11	0.27	0.22

(int = interlaced image)

The MorphOS version is round about the same speed as the WarpOS version (above), but a little bit faster in some cases. It's much faster than the PowerUp version of akPNG running under emulation in MorphOS - around 50-80% faster in the case of 24-bit images, and 3-5 times faster in the case of 8-bit greyscale and paletted images. No benchmarks for 68k versions are included because they were found to be many times slower.

		PowerPC only (no 68K)	
		MorphOS	
		WarpPNG 44.19	akPNG 44.112 PowerUp emulation
	bytes		
1024x768 24-bit,	833721	1.24	1.77
1024x768 24-bit int,	1037008	1.39	2.19
779x767 24-bit,	946564	1.08	1.66
473x639 24-bit,	377518	0.52	0.83
473x639 24-bit int,	459869	0.57	0.99
368x463 24-bit,	279896	0.36	0.63
251x400 24-bit,	136856	0.22	0.44
718x425 grey,	157890	0.24	0.93
718x425 grey int,	182522	0.27	0.99
1024x768 256 col,	324599	0.37	1.77
779x767 256 col,	404245	0.38	1.58
473x639 256 col,	163397	0.21	0.93

368x463 256 col,	115942	0.15 0.67
251x400 256 col,	57056	0.11 0.49

(int = interlaced image)

The results for the 68k version are much tighter, with WarpPNG only being 5-10% faster than Gunther Nikl's datatype on non-interlaced images. For interlaced images, WarpPNG is 50% faster for 24-bit images, and 70% faster for 8-bit images.

		M68K		
		WarpPNG	PNGdt	akPNG
		44.19	G.Nikl	44.112
1024x768 24-bit,	833721 bytes	3.56	3.76	4.62
1024x768 24-bit int,	1037008 bytes	4.22	6.26	7.96
779x767 24-bit,	946564 bytes	3.18	3.33	4.31
473x639 24-bit,	377518 bytes	1.38	1.49	1.97
473x639 24-bit int,	459869 bytes	1.67	2.51	3.30
368x463 24-bit,	279896 bytes	0.97	1.03	1.43
251x400 24-bit,	136856 bytes	0.55	0.59	0.88
718x425 grey,	157890 bytes	0.67	0.71	1.17
718x425 grey int,	182522 bytes	0.81	1.39	2.64
1024x768 256 col,	324599 bytes	0.91	0.91	1.66
779x767 256 col,	404245 bytes	0.90	0.94	1.72
473x639 256 col,	163397 bytes	0.46	0.48	0.94
368x463 256 col,	115942 bytes	0.29	0.32	0.69
251x400 256 col,	57056 bytes	0.16	0.19	0.45

(int = interlaced image)

Note: all tests were performed, multiple times, using Visage, with the following command line: "visage test.png nopng time test".

WarpOS version still too slow!

Despite these facts, I'm still disappointed with the relatively slow performance advantage offered by my 603e over my 060 (approximately only 3 times faster). The main problem is that PPC datatypes still have to use the 68k for reading the data from disk and for creating / writing to the bitmap time lost for file i/o is negligable as WarpPNG uses double buffered asynchronous i/o (supports DMA controllers).

The largest bottleneck is that the DTM_WRITEPIXELARRAY method of the picture.datatype has to be used to write the image data from WarpPNG into the image bitmap. As this process is done via picture.datatype, it can only currently be performed by the 68k. To give you some idea of how much of a problem this is for WarpPNG, typically, half of the overall decode time is used by the PPC to decode the whole image, and the other half is used by DTM_WRITEPIXELARRAY on the 68k. And that's on a graphics card - the time used by DTM_WRITEPIXELARRAY will probably be even greater on systems using native Amiga graphics. It doesn't take a genius to see that this is slowing the datatype down, and is the main reason why WarpPNG will still be faster on a 060 than a 040.

How to make the datatype faster

Is there anything that can be done about this? Well, yes, there are a few patches that you can install which should make things faster:

- NewWPA8 (util/boot/NewWPA8.lha on Aminet) should provide a notable speed increase on native Amiga graphics - probably won't make any difference if you use a graphics card.

- If you use a graphics card and CyberGraphX, you may want to make sure you are using the supplied v43 picture.datatype, as this will be faster than the P96 and OS3.5/3.9 picture.datatype on your system.

Of course, any other general speed-up patches should help too.

1.8 Distribution Conditions

WarpPNG.datatype is public domain with the copyright remaining with the author and may be freely distributed legally providing:

(1) None of the distributed files are changed in any way
(2) It is not sold for profit and it is not included on any disks that are sold solely for profit (includes magazine coverdisks)
(3) The distribution contents remain complete (see list below)

If this software is to be sold for profit, permission must be obtained from me, the author.

Aminet, Amiga Format and Amigactive have been granted permission to distribute WarpPNG.datatype on their CDs.

The following files must be present in their original and unchanged form in any copies of this software:

Classes/Datatypes/WarpPNG.datatype.020
Classes/Datatypes/WarpPNG.datatype.030.pch
Classes/Datatypes/WarpPNG.datatype.040.pch
Classes/Datatypes/WarpPNG.datatype.060.pch
Classes/Datatypes/WarpPNG.datatype.wos
Classes/Datatypes/WarpPNG.datatype.elf
Devs/Datatypes/PNG
Devs/Datatypes/PNG.info
WarpPNG.guide
WarpPNG.guide.info
Install_WarpPNG
Install_WarpPNG.info
spatch

1.9 Disclaimer

This software is provided "as is", without warranty of any kind, either expressed or implied, statutory or otherwise. By using the archive and its contents, you accept the entire risk as to its quality and performance.

Neither Oliver Roberts nor any other party involved in the creation,
production or delivery of the archive and its contents shall be liable for
any direct, indirect, special, consequential or incidental damages,
including without limitation damages for loss of profits, loss of use or
loss of anticipated costs, expenses or damages, and any data or information
which may be lost or rendered inaccurate, even if Oliver Roberts is advised
of the possibility of such damages.

Do not attempt to tamper with the supplied files. Doing so will cause
problems and you may find things start going wrong!

1.10 Acknowledgements

PNG support based on the libpng link library by the PNG Development Group,
and the zlib link library by Jean-loup Gailly and Mark Adler.

The WarpOS version was made possible by VBCC, which was used to build and
compile the datatype. Thanks to Volker Barthelmann and Frank Wille, for
their support and help.

The Spanish installer translation is by Dámaso D. Estévez.

The French installer translation is by Philippe Bovier.

Thanks also to Sam Jordan for WarpOS and helping me out with various
queries regarding it.

Finally, thanks to the OS 3.5 development team - now everyone has access to
a 24-bit picture.datatype, I don't need to bother messing about adding
dithering routines :)

1.11 Support & the Future

Some things that may appear in the future:

 - If I can squeeze any more speed out of the datatype, I'll do so :)

If you have any other suggestions, please let me know.

Future releases of WarpPNG.datatype will be available from either Aminet
(util/dtype/WarpPNGdt.lha) or its webpage:

  http://www.nanunanu.org/~oliver/warppng.html

If you would you like to know when WarpPNG is next updated, then you may
want to subscribe to my announcement list to receive an e-mail informing
you of the changes as soon as a new versions of any of my products are
released. To subscribe, send a blank e-mail to

  futaura-announce-subscribe@yahoogroups.com

```
or go to

  http://groups.yahoo.com/subscribe/futaura-announce
```

1.12 About the author

```
If you have any problems with this software, or if you have any
suggestions/queries, please contact me and I will do my best to sort any
bugs out as soon as possible:

e-mail: oliver@futaura.co.uk
   www: http://www.nanunanu.org/~oliver/
   icq: 34640231
```

1.13 Program History

```
44.19 (20.8.2001)

 - Added option to allow the alpha channel to be ignored, for images
   which have one.
 - Now supports the PDTA_GetNumPictures and PDTA_WhichPicture tags.
 - Discovered a bug in the CyberGraphX picture.datatype, concerning
   error numbers which would have resulted in an unknown error being
   reported by application software if bitmap allocation failed - added
   workaround.

44.18 (25.7.2001)

 - Faster decoding of interlaced images - 80% faster for 8-bit images
   on 68k and 20% faster on PowerPC.  50% faster for 24-bit images on
   68k and 10% faster for PowerPC.
 - Updated with libpng 1.0.12.
 - PowerPC versions recompiled with VBCC 0.8.
 - Added French strings to the installer.

44.17 (2.5.2001)

 - Updated with libpng 1.0.11 (this should have been in yesterday's
   update - blame my ISP's webcache :).

44.16 (1.5.2001)

 - Updated with libpng 1.0.10.

44.15 (10.4.2001)

 - The built-in image gamma value (used when no value is defined in the
   prefs) was incorrect - fixed.
 - Gamma correction did not work in the 68040 and 060 versions, causing
   images to be mostly black or white if the default gamma values were
   not used.  This was due to a non-working FPU pow() implementation -
   mathieeedoubtrans.library is now used instead.  Does not affect speed.
```

```
44.14 (4.4.2001)

 - Enabled the gamma correction feature of libpng, and provided some
   preferences options to control it.
 - Added options to control the precision/quality of pen allocations for
   when an image is displayed on a pallette-based display (8-bit).
 - Unfortunately, the all versions are now fractionally (less than 1%)
   slower than before, although interlaced images are now decoded a bit
   faster in the 68k versions.
 - Added Spanish strings to the installer.

44.13 (15.3.2001)

 - Added preferences options to configure the OS3.5/3.9 dithering control
   feature.
 - The MorphOS version got broken in v44.11 - fixed.

44.12 (19.2.2001)

 - The dithering disabler feature (for when the OS3.5/3.9 picture.datatype
   is in use on hi/true colour displays) was flawed, since it only worked
   with images that were eventually attached to windows (e.g. Multiview).
   Now dithering will be disabled in all other cases too (e.g. Workbench /
   DOpus backdrops).

44.11 (1.2.2001)

 - Updated with libpng 1.0.9.

44.10 (12.1.2001)

 - Fixed a bug in the new startup code, which caused images not to be
   loaded in some cases - e.g. IPrefs failing to load WB backdrops.

44.9 (9.1.2001)

 - Recoded much of the startup code, with libraries now not being opened
   on the ramlib context (after advice from H&P).
 - Modified internal structure of the WarpOS version, to totally safeguard
   against cache conflicts (may have been causing rare random crashes).
 - Changed floating point usage of the 68k versions - the 040 and 060
   version now use the FPU directly, while the 020 and 030 versions
   continue to use the mathieee libs (in a safer manner than before).
 - All WarpOS benchmarks recomputed, since WarpOS V5 is a little bit
faster
   than V4 (this datatype is a bit faster as a result).
 - The 68k versions are now fractionally faster than before.
 - Added benchmarks for MorphOS.

44.8 (7.9.2000)

 - Added a MorphOS native version and 68k version (in 020, 030, 040 and
   060 flavours) to accompany the original WarpOS version.
 - Minor optimizations (code size only - no extra speed :(

44.7 (2.8.2000)
```

```
 - Around 10% faster decoding of most images!
 - Recompiled with VBCC 0.7d beta, which gives a slight performance
   increase and smaller program binaries.
 - Tweaked compiler optimization options and increased disk i/o buffer for
   enhanced performance.
 - Recomputed all benchmarks, to ensure correctness after a change in my
   test environment (test files are now on a different HD, due to a
   semi-fatal HD crash on the old partition).

44.6 (26.7.2000)

 - Updated with libpng 1.0.8.

44.5 (4.7.2000)

 - Updated with libpng 1.0.7.
 - Slightly faster (upto 2%) in some cases.

44.4 (27.4.2000)

 - Updated with libpng 1.0.6.
 - Tweaked compiler options and made some optimizations, which gives a
   slight performance increase when decoding 24-bit and greyscale
   images (around 3% faster - 8% faster for interlaced images).
 - Reworked transparency handling.

44.3 (22.1.2000)

 - Fixed possible deadlock case (caused everything to freeze/lock-up)
   which usually only occurred when lots of images were being decoded
   simultaneously.

44.2 (9.1.2000)

 - When the OS3.5 picture.datatype is in use, dithering is now switched
   off (only) if the image is to be rendered to a hi/true colour screen,
   resulting in much higher performance, with negligable quality loss.
 - Dispatcher now performs some extra functions, which should quash
   possible stability problems.
 - Corrected akPNG WarpUp benchmarks.

44.1 (3.1.2000)

 - Initial release.
```