

BLAZEGUIDE

Rick Pratt

COLLABORATORS

	<i>TITLE :</i> BLAZEGUIDE		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY	Rick Pratt	August 10, 2022	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	BLAZEGUIDE	1
1.1	BlazeWCP	1
1.2	It's a floor wax, it's a dessert topping....	2
1.3	I'm a glutton for punishment...	2
1.4	300THz cpu with 512 GB mem...	3
1.5	If it blows up, it's your fault	3
1.6	You'll need a hammer	3
1.7	Errors	3
1.8	This program brought to you by MicroSof.....	4
1.9	Version history	4
1.10	Future	6
1.11	If I mention your name, pay up :)	6
1.12	You will be assimilated.....	6
1.13	Contacting Me	7
1.14	I feel so dirty....	7
1.15	Faster than a speeding snail....	8

Chapter 1

BLAZEGUIDE

1.1 BlazeWCP

BlazeWCP - Very fast C2P patch for the OS chunky pixel functions

Click

here

to see what I've found that does and doesn't

use these functions since this seems to be a point of confusion for some people.

What is it

Why another patch

NEW--->

Speed Tests

<---NEW

Requirements

Disclaimer

Installation

Error Codes

Bugs

Version History

Future stuff

People I'd like to thank

Other Stuff By Me

Please read WPATest guide before using WPATest.

Flames, bugs, congratulations...

find me here.

Find the latest version of BlazeWCP at <http://kingamiga.webjump.com>. ↩

1.2 It's a floor wax, it's a dessert topping....

BlazeWCP is a highspeed C2P patch for the OS chunky ↩
functions
WritePixelLine8(), WritePixelFormat8() and on OS 3.1+ WriteChunkyPixels().
BlazeWCP writes directly into the destination bitmap and uses 32bit C2P ↩
conversion
code for maximum speed. All of this makes it fast on 020+ cpus and REALLY ↩
fast on
machines with 32bit chipram. Using the included wpatest, a benchmark
of these functions written by Stephen Brookes (programs FBlit), I did some
speed/compared test. The results are
here
in this section.

1.3 I'm a glutton for punishment...

I made BlazeWCP after I got sick of how slow the other patches of this type were ↩
. PatchWPA8
was very slow on my 030 since it calls BltBitMapRastPort() after everyline it ↩
converts and
since it uses a single scratch bitmap and buffer for its functions it gets SLOW ↩
!! when more
then 1 program tries to use it's routines. NewWPA8 was faster but it only used ↩
16 bit C2P
code which is slower then it has to be when you have a 32bit cpu and 32bit ↩
chipram. NewWPA8
also has a bug that the author denies that causes line corruption with IBrowse ↩
since it calls
WritePixelLine8() to render to the screen. The corruption is caused by the fact ↩
that NewWPA8
builds the hook it passes to DoHookClipRects() using memory internal to the ↩
patch itself.
This works fine if only 1 program uses it's routines but when 2 or more use them ↩
the hook
gets overwritten with the new callers data. This causes invalid pointers to be ↩
passed to
DoHookClipRects() which causes the corruption me and others noticed. Since ↩
Ibrowse spawns
a task for every picture being rendered/decoded these routines DO get called by ↩
more then
one task and that's why some lines get corrupted.

/BEGIN RANT

I emailed Michael van Elst, the author of NewWPA8, about this corrupt lines ↵
problem about
2 years ago, before I knew what caused it. He sent me an email back claiming ↵
that
there was nothing wrong with NewWPA8 because it "worked for him". Mr. Elst (or ↵
van Elst),
any good programmer will tell you that just because something "works for you" ↵
doesn't mean
it works for everybody. Your complete lack of interest in a bug in your patch ↵
shows me
that you truly didn't care, or even bother to check. I guess I'm lucky you even ↵
bothered
to email me back at all.

/END RANT

1.4 300THz cpu with 512 GB mem...

A 020+
Some fast ram
Hopfully OS 2.0+, otherwise 3.0+. I need feedback on this one.

Recomended:

A 030 50Mhz
32bit chip ram
AGA chipset
FBlit

1.5 If it blows up, it's your fault

I take no responsibility for any death or destruction caused by this
patch. Use it at your own risk.....you've been warned.

1.6 You'll need a hammer

Just copy BlazeWCP to your C directory and add C:BlazeWCP to your
startup-sequence AFTER SetPatch BUT BEFORE any gfxcard support
software like P96 or cybergfx. I doubt you'll be using this patch
if you have a gfxcard tho....lucky you :)

1.7 Errors

The patch will return error code 10 if it can't allocate memory for the patch, 20 if you don't have a 020 or better and 30 if it can't open graphics.library v36+.

1.8 This program brought to you by MicroSof.....

None known, email me if you find some.

1.9 Version history

0.x - Internal testing versions, buggy and not optimized, not released.

1.0 - Was supposed to be the initial release.

1.1 - Sped up 8bit and 4bit conversion routines a little.

- Sped up the main loop a bit.
- Added support for rastports with masked planes.
- Initial Release.

1.2 - Streamlined the c2p code a little, constance are only loaded into registers once instead of by both routines of a conversion >4 planes. Makes the code smaller and possibly a little quicker.

- Now calculates # pixels plotted by adding up the pixels plotted for every rect the hook is called for. Should make the #s returned by WPL8 and WPA8 100% accurate....WCP now actually returns # pixels plotted too even tho it doesn't have to.
- 2nd Release.

1.3 - Reorderd register usage a little. This saves 12 bytes per routine and freed up a register which is now used to defer a chipram write for the 4/8 bit conversion routines instead of defering it to the stack.

- The above optimization made me think a little (not too much, i'd hurt myself ;) and allowed me to see a way to keep the other chipram write that was defered to the stack in the 4/8 bit routines in a register. Now all the conversion routines run completely within the registers and are just complex mem copy loops, as they

should be.

- 3rd Release

1.4 - Removed some unneeded code that didn't hurt anything, but wasn't helpful either. ↔

- Ooops, due to the way masks and alignment code is generated BlazeWCP would do a conversion twice if it was smaller then 31 pixels wide and didn't end exactly at a 32 bit boundry. This didn't cause any problems but slowed these small conversions down when they didn't have to be. ↔

- Removed a very bad bug in the code that generates displacement into the chunky buffer. The code used the width of the operation instead of the BytesPerRow that gets passed to WriteChunkyPixels() to move thru the chunky buffer. The only program I know of that even shows a problem because of this is ImageDesk v3.04. Thank you very much to Stephen Brookes for finding this bug for me and to Andras Gabor for reporting it. How this patch worked correctly all this time is really beyond me :) ↔

- 4th Release

1.5 - Fixed the masked rastport code since I broke it in the last release. Just made things look funny is all. Not that big a deal since the only thing I know of that calls these functions with masked rastports is wpatest. ↔

- Added a very small speedup for 020/030 cpus that only affects WritePixelArray8() and WriteChunkyPixels(). ↔
- Patches now install with Forbid()...CacheClearU()....Permit(). You never know what might try and use functions while they're being patched.... ↔

- 5th release

1.6 - Combined the loops that build the temp. stack bitmap and initial destination bitmap pointers. This saves a little space. ↔

- Filled in that saved space making the patch about 40% bigger by adding c2p code written specifically for the first/last longword masking. This is faster on my 030 by only ~6% so it may not stay. ↔
- 6th release
-

1.65 - Fixed the corruption caused when the new alignment routines were called to render into a small (16 pixel or smaller) interleaved bitmap. It was caused by a little memory access goof-up on my part. Thanks to everyone that reported this bug.

- 7th release

1.7 - Inlined code that takes the place of the DoHookClipRects() call. This removes the need for OS 3.0 and so this patch should work from OS 2.0 and up.

- Made some other changes that eek out a tiny bit more speed.

- 8th release

1.10 Future

- More speed.....that's a tough one....

- Sugestions welcome.

1.11 If I mention your name, pay up :)

- Iain Barclay: for 8n1.device, getting me into asm programming and putting up with stupid questions ;)

- Stephen Brookes: for FBlit and for the help you've given me. Also for responding to my emails even when you're swamped and I'm sure couldn't care less about them at the time.

- Michael Kalms: for the lightning fast chunky 2 planar routines that this patch uses.

- Michael van Elst: for adamantly denying the existance of a blatanly obvious bug in NewWPA8, which aside from that is a really good patch.

I'd also like to thank everyone for the feedback I've gotten :)

1.12 You will be assimilated.....

Other stuff by me on Aminet

FText V1.7 - Speeds up text by rendering to fastram instead of chip. Needs ↔
FBlit V3+.

Get the latest FBlit from <http://www.tpec.u-net.com>

QBC V1.1 - A cpu only BlitClear() patch.

1.13 Contacting Me

Send praise, suggestions, bug reports, flames to:

rickprat@usit.net

or find me on DalNet channel #AmIRC nick: Kingamiga

Check out my poor excuse for a website at <http://kingamiga.webjump.com>

1.14 I feel so dirty....

I added this little section because it seems that some people are confused as to what BlazeWCP should affect. I know this will miss most of the people it's aimed at since they don't read docs but I'm getting tired of responding to emails like "BlazeWCP doesn't help my SysSpeed scores" or "My workbench text scrolls slower now". Sorry folks, BlazeWCP has no effect on either of these things but I will add a list of things that DO use BlazeWCPs functions and things that DON'T use them. I can only list things that I have, feel free to email me about any programs you have that use BlazeWCPs funtions.

The programs that use the patched functions only use them to render images so the functions will not:

- Make text scroll/display quicker (I've another patch for that ;)
- Make windows move/diplay quicker.
- Make window contents scroll quicker.
- Speed up icon rendering.
- Speed up/ slow down benchmark scores that don't test these functions.
- Speed up your games unless the game gives you the option to use these ↔
functions

or uses them already.

- Give your Amiga more gfx power then your PC.
- ...etc....etc...you get the idea....I hope..

To put another way, BlazeWCP is faster then other patches of this type but it can only speed up things that use the patched functions. It won't make your cpu faster or your HD give 100mb/s transfers....sorry.

Things I have that DO use BlazeWCPs functions:

```

IBrowse      V1.0 - V1.12  (V1.2 and newer use there own c2p code)
jfif.datatype using tower jpeg.codec
wpatest
P96Speed
DoomAttack with -rtg tooltype set
ImageDesk   V3.04
Eagleplayer

```

Most Mpeg/Quicktime/AVI players let you set them to use WritePixelFormat8() using a tooltype.

Things I have that DON'T! use BlazeWCP

```

Voyager
SysSpeed (sorry people, but BlazeWCP doesn't get used by SysSpeed)
WorkBench

```

I've other things that don't use these functions (most everything not listed as using them) but I didn't see the point in listing those.

User reports are welcomed.

1.15 Faster than a speeding snail....

I've added this section to move all the speed stats, etc. to. The guide was getting a bit messy the way it was before.

I ran a test using aMiPEG V1.0 from aminet and get the following results. I used a 208x176 mpeg using only the DITHER gray option on a 256 color 320x256 lo-res PAL screen. My specs are:

```

A1200  OS 3.0 030/882 40MHz 8megs Fast 2megs chip

```

```

WPA8 Function  fps avg.
used:

```

```

OS 3.0      3.439371 fps
PatchWPA8  5.906459 fps
NewWPA8    5.912143 fps
BlazeWCP   10.791788 fps  <- not bad for a 030 40MHz ;)

```

Using an 8bit (256 color) 640 x 512 hires laced PAL screen and WPATest to benchmark I get the following results: (#'s are thousands of pixels/s so 100k is 100,000 pixels/s)

Please read the wpatest guide before using it. It's a little buggy. The pause after hitting the Speed button is normal.

```

not 32bit aligned      32bit aligned

```

```

OS 3.0      111k pixels/s      272k pixels/s
PatchWPA8   152k pixels/s      593k pixels/s <- FBlit's why it's this fast on <-
  this test
NewWPA8     200k pixels/s      569k pixels/s
BlazeWCP    482k pixels/s      2072k pixels/s

```

Speeds Reported by users, (non-aligned) - (aligned) in thousand pixels/s
 Test Screen - PAL 8bit (256 color) hires laced 640 x 512:

	WPL8	WPA8	WCP	
030 40MHz		286 - 1462	482 - 2072	482 - 2072
030 50MHz		xxx - xxxx	547 - 2478	xxx - xxxx
040 28MHz		453 - 2027	636 - 2490	634 - 2490
060 50MHz		xxx - xxxx	876 - 4124	xxx - xxxx
060 50MHz		852 - 4067	1076 - 4624	1072 - 4620

I also had a report from a man with an ECS equipped A2000 that BlazeWCP <-
 works very well.
 His results were:

```

030 40MHz      92 - 430      157 - 667      156 - 663

```

Of course he was only able to run the test at 16 colors on a 640 x 512 PAL <-
 screen
 but he reported a speedup of 4x for the non-aligned test and 10x for the <-
 aligned test
 compared to the OS routines.