FalconPatch

Ben Hutchings <benjamin.hutchings@worcester.oxford.ac.uk>

COLLABORATORS

	TITLE : FalconPatch		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Ben Hutchings <benjamin.hutchings@worcester.oxford.ac.uk>	August 6, 2022	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

FalconPatch

1.1 Contents

FalconPatch version 1.0 - 24 July 1997

by Ben Hutchings

FalconPatch fixes defects in the Falcon 68040 accelerator for the A1200

You need AmigaGuide version 40 to display this guide correctly. I recommend you also use AGPatch to fix some of the remaining bugs in version 40.

1.2 Legal Notices

Copyright

The FalconPatch package consists of the files:

ClearRes FalconPatch FalconPatch.guide FalconPatch.guide.info

(hereafter "the files").

This package is copyright © Ben Hutchings 1997. It may freely be distributed provided:

(i) no charge is made for this above the costs of duplication, distribution and the media used, (ii) all the files are distributed together, and (iii) none of the files are altered, except, if the distributor so wishes, by converting them into an archived or compressed form from which they can be retrieved unaltered using readily-available free software tools.

Software released under these conditions is often known as Freeware.

As far as I am aware, the NewIcons images used in the file and "FalconPatch.guide.info" is © 1996 The NewIcons Team and may freely be distributed on its own.

Disclaimer

The FalconPatch package is provided "as is", without warranty of any kind, either expressed or implied. The entire risk as to the quality and performance of the program is with you.

1.3 Introduction to FalconPatch

Introduction to FalconPatch

FalconPatch is a program intended to fix various problems with the Falcon 68040 accelerator board for the A1200. It also fixes a general problem with expansion memory on the A1200.

The A1200 has a bit of a problem with expansion memory in that although it is a 32-bit machine, the built-in CPU and the motherboard only have a 24-bit address bus. All 68030/040/060 accelerators, which have a full 32-bit address bus, map their memory outside the 24-bit addressing range. (It's only possible to fit 8M of Fast RAM in that range anyway - or 4M if a PCMCIA card is to be used at the same time.) This means that the memory cannot be auto-configured, so many system-critical structures will be stored in Chip memory instead, which seriously damages multi-tasking and disk performance. The 3.1 ROM improves on this situation by auto-configuring any memory starting at a particular address in the memory map, but I do not know of any A1200 accelerators that actually use this address.

The excellent program FastExec provides a partial solution to the above problem; however, I have recently found a case which it fails to address. Amiga International has started to release beta versions of some components of the next AmigaOS (version 43) on its public FTP site; in particular, IDE_scsi.device, which now recognises "IDE" (ATAPI) CD-ROM drives and has various other enhancements over the version in the A1200's ROM. These components have to be present when the computer boots, so they are loaded into memory and then the computer is rebooted. After a reboot, some memory may "disappear" until it is configured again. In fact, only the Chip RAM will stay just where it is. So the IDE_scsi.device will be loaded into Chip RAM - which makes it run very slowly indeed! FalconPatch ensures that the Falcon's Fast RAM re-appears almost immediately after a reboot, so it's safe to store new system components in Fast RAM. Correctly-written programs will automatically take advantage of this.

There are also a few bugs in the Falcon's on-board ROM (which is quite amazing, considering how short a program is stored there). First, it gives the Fast RAM a priority of 0, when it should be +40. Secondly, it gives the RAM a name of "Falcon RAM", but then frees the memory containing this name (so the name gets over-written). Thirdly, the test to work out how large a SIMM you have installed over-writes some data, which isn't good if you're ripping things out of memory. ;-) FalconPatch fixes all these problems.

Warning: FalconPatch relies heavily on undocumented features of the Kickstart ROM and the Falcon board. Although the greatest care has been taken to ensure that FalconPatch will refuse to run in an unexpected hardware configuration and will handle every expected hardware configuration correctly, I cannot guarantee that it will not cause damage to data or hardware.

1.4 What you need to run FalconPatch

What you need to run FalconPatch

· An Amiga 1200 · A Falcon 68040 accelerator · Some Fast RAM · A hard drive · Kickstart version 39.106 or 40.68 (select About in Workbench to check this) · A caffeinated or alcoholic beverage (this is optional but strongly recommended; if you have some to spare, I can find a good home for it)

Hardware I recommend you avoid

· Squirrel SCSI card (what, you want multitasking?) · Falcon accelerator (sucks completely, but you already know that) · Amiga 1200 (more trouble than it's worth) · Amiga 3000 (too slow, too cramped, several chips are buggy) · Amiga 4000 (too expensive, memory performance sucks) · Any older Amiga (obsolete)

Can you tell that I'm fed up with Amiga hardware at the moment?

1.5 Installing FalconPatch

Installing FalconPatch

There is no automatic installation routine for FalconPatch, because it would be rather complicated to make it work with all possible software configurations. You must carry out installation yourself as follows:

1. If you are using an early version of FastExec that always reboots the machine when it is run, you must remove it before continuing. You can do this by loading your startup-sequence into a text editor and putting a semi-colon (;) at the beginning of the line that runs FastExec. Then reboot before continuing.

2. Run ClearRes (included in this package) and then FalconPatch, and see whether it produces an error message. If your computer meets the requirements and yet you still get an error message, please contact me with the details of your machine's configuration.

3. If you are using a later version of FastExec that you didn't remove at step 1, remove it now in the same way.

4. Reboot the computer and check that it boots correctly. If it doesn't, please contact me with details - and change your startup-sequence back! If you're using a tower system with Zorro slots, you may find that the screen turns cyan when you reboot. This indicates that the Falcon is not the first board in the system, which is problematic for FalconPatch. In this case, you should reboot again and contact me to see what can be done about making it support your system.

5. Copy FalconPatch to the C directory of your boot partition.

6. Load your startup-sequence into a text editor again.

7. If you have any other programs that install code to run immediately after a reboot, such as LoadV43Module, you must insert the line "C:FalconPatch" before the first of them. Otherwise, add the line "C:FalconPatch REBOOT" immediately after "C:SetPatch QUIET". See the examples below if you're confused.

7. Save the startup-sequence.

Examples

The standard OS 3.1 startup-sequence looks like this:

; $VER: Startup-Sequence_HardDrive 40.2 (25.2.93) ; Startup-Sequence for hard drive systems

C:SetPatch QUIET

C:Version >NIL: C:AddBuffers >NIL: DF0: 15 FailAt 21 ... To install FalconPatch, you would add the line "C:FalconPatch REBOOT" immediately after the line "C:SetPatch QUIET".

A few days ago, my startup-sequence looked like this:

c:loadv43module devs:a300.ld.strip REBOOT

c:setpatch quiet

;If a temporary custom startup exists, execute and delete it if exists s:temp-startup copy s:temp-startup ram: delete s:temp-startup quiet execute ram:temp-startup delete ram:temp-startup quiet quit endif

mountenv

c:patchcontrol ... I added "C:FalconPatch" as the first command.

Note: If FalconPatch is already in memory when you run it, it will return 5 (WARN). You might want to test for this in your startup-sequence, though I doubt it. ;-)

1.6 Technical Details

Interaction with the OS

FalconPatch installs a ColdCapture vector that carries out a number of critical procedures involved in starting the Amiga, configures and adds the Falcon's RAM, and then jumps back into ROM. At the time a ColdCapture vector is called, it is not normally possible to add RAM to the system. Exec's memory list does not get initialised until slightly later. Therefore FalconPatch's code duplicates what the ROM does up to the point where RAM gets added to the system, at which point it adds the Falcon's RAM. Unfortunately, it is not possible to do this without using a large number of absolute ROM addresses - which is why FalconPatch only works with Kickstart versions 39.106 and 40.68.

FalconPatch also installs a ROMTag that restores the ColdCapture vector (which gets cleared after every time it is called) and tells expansion.library about the Falcon board using the AddConfigDev() call.

Another thing FalconPatch does is to set the attributes, priority and name of the Falcon's RAM correctly when it is first run - they are set to MEMF_KICK|MEMF_FAST|MEMF_PUBLIC, 40 and "expansion memory". This ensures that other programs that install ROMTags will allocate them in Falcon RAM before the first reboot. It is usually worth forcing a reboot during the first pass through startup-sequence - both FastExec and FalconPatch cut my boot time from 39 to 28 seconds - but only the last program that installs a ROMTag should do this, obviously!

Interaction with the Falcon board

FalconPatch searches for a board with manufacturer ID 18260 (MacroSystem) and product ID 253, which is the product ID of my Falcon board. It also checks that the serial number is $18051995 - which you may notice is actually a date.

Falcon RAM maps to addresses $1000000-$18000000. DRAM chips are given addresses in two parts - a row address and a column address. These parts have the same number of bits or else differ by 1. So different sizes of chip need the addresses to be split up in different ways:

SIMM size Chip "area" Row bits Column bits 128 Mbyte 32 Mbit 12 13 64 Mbyte 16 Mbit 12 12 32 Mbyte 8 Mbit 11 12 16 Mbyte 4 Mbit 11 11 8 Mbyte 2 Mbit 10 11 4 Mbyte 1 Mbit 10 10

A register on the Falcon board, mapped to address $20000000, selects how many row bits to use. The Falcon's ROM code only writes long words to this address, so FalconPatch does the same. Writing $30000000 selects 12 row bits; $50000000 selects 11; $90000000 selects 10. I think it's also possible to select 9 row bits. It's safe to select an incorrect number of row bits while testing the memory size! After a reset, the board seems to default to 9 row bits. This could cause problems for FalconPatch if its code was stored in the Falcon's RAM, so it is actually stored in Chip RAM. This isn't a significant performance hit, as the Falcon code only runs for a moment during a reset. It also seems to me that there could also be a problem with reading ExecBase immediately after a reset when it is stored in the Falcon's RAM - but there isn't, thankfully!

Currently, FalconPatch only works if the Falcon board is the only or the first installed Zorro II board. If you are using a tower-converted A1200 with Zorro II slots, FalconPatch probably won't work - in this case, you will get a cyan screen when you reboot after running FalconPatch. It may be possible to fix this, but I don't know whether it's worth it and I don't have a system to test on. If you have such a system, please contact me.

1.7 Potential problems

Potential problems

1. FalconPatch is not and probably never will be compatible with soft-kickers. I'll consider integrating an MMU-soft-kicker with FalconPatch at a later date.

2. If you have a tower-converted A1200 with Zorro expansion slots, FalconPatch may not work. See step 4 of the installation procedure for more information.

3. Any virus checker you run is likely to warn about the ColdCapture, KickMemPtr and KickTagPtr vectors being set to unusual values. Do not clear these vectors, as this will prevent FalconPatch from working.

1.8 Acknowledgements

Acknowledgements

· Power Computing in the UK, for selling me this horrible board in the first place. I intended to buy the Turbo 1240 that they announced they would be selling in August 1996. This announcement was a characteristically "optimistic" bit of PR - when I phoned them, I was told they wouldn't actually be selling it at that time, but that the Falcon was "better than the Turbo 1240". Obviously they have a quite unique understanding of the term "better". The Turbo 1240 costs less, has a clock on board, and has a SCSI module that isn't a figment of Power Computing's collective imagination. But if I had bought a better accelerator, I wouldn't have written this program, you see?

· Torbjörn Andersson, for writing FastExec. Although I didn't look at the source of FastExec while writing FalconPatch, it was certainly an inspiration - I probably wouldn't have thought this patch was possible if I hadn't seen his source.

· Carl Sassenrath, for writing such a modular OS kernel that a low-level patch like this is possible even when the OS has ill-advisedly been stored on a ROM.

1.9 How to contact me

How to contact me

Please write or email me if you have any comments, suggestions, bug reports or programming hints to make.

snail Ben Hutchings 43 Harrison Close Reigate Surrey RH2 7HS email benjamin.hutchings@worcester.oxford.ac.uk WWW http://users.ox.ac.uk/~worc0223 (for support pages, append /amiga/freeware) IRC womble or wombleII (usually in #amigacafe on EFnet/Undernet)