003a6898-0

COLLABORATORS

	TITLE : 003a6898-0		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		August 5, 2022	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

003a6898-0

1.1 No title

```
                                                  CpuControl V1

                          Copyright 1995-1996 by Oliver Goss

                               Release 1.06 / 07.07.1996


                            CPUCONTROL IS A CARDWARE PRODUCT

                 PLEASE READ THE 'DISTRIBUTION' SECTION FOR MORE DETAILS.
```

CONTENTS:

```
~Technical~Details~
     Some stuff for the programmers ...


~Compatibility~~~~~
     Successfully tested configurations




~Author's~notes~~~~
     Some remarks from me ?!


~Distribution~~~~~~
     Details about distribution


~How~to~contact~~~~
     If you feel the need ...




~History~~~~~~~~~~~
     Curriculum vitae


~Troubleshooting~~~
     If you have a problem with CpuControl


~To~do~...~~~~~~~~~
     Features that might be added some day
```

1.2 Installation

Installing CpuControl:

Installing CpuControl is really no big deal. Just copy the executable to your C: directory or any other directory you might find useful. If you want, you can change the filename into a shorter name. I recommend to replace the original CPU command (as it comes along with the OS 2.0+ distribution), since CpuControl offers the same functionality with somewhat improved capabilities.

1.3 Introduction

Introduction:

CpuControl started some time ago as a programming exercise. The idea was to create a program that uses the advanced features of Motorola’s MC68020 and (especially) MC68030 processors like Cache manipulation, MMU code and the enhanced addressing modes. So I decided to do some replacement for the CPU command as it comes along with the Workbench distribution. CPU is a very useful tool, but it could need some improvement, anyway. There were several aspects which I was not quite satiesfied with as the following examples may illustrate:

o CPU accepts any command line if only the keywords are valid. Things like CPU CACHE NOCACHE CACHE CACHE FASTROM NOFASTROM ... are processed without problems problems, since CPU executes every option one after another. So the Caches are acticated, disactivated, activated, and so on. This is not really a disadvantage, but I think a good command line parser should reject such crap.

o When the FASTROM feature is activated, CPU moves the VectorBase into FastRam in order to improve interrupt handling. This cannot be undone without restarting the machine.

o The MMU table used by the FASTROM option is not very efficient (it is larger than it needs to be, I mean).

But CpuControl was not intended as a simple CPU clone. Due to the hardware restrictions of my old A2000 I was forced (almost ...) to use the FASTROM feature ever since I got my A2630 accelerator board. This occupies 512K of my precious 32Bit RAM (4MB ain’t too much, especially when you have got a graphics board !). The memory loss cannot be helped, but it *should* be possible to use this 512K not only for a simple copy of my ROM contents. So I implemented a real nice KICK option that makes it possible to load and activate another KickStart version (see
Features
section for details). I’ve been looking for a MMU based SoftKicker for some time, but I found none for my hardware configuration. There’s SoftKick for A3000s and Set040 for 68040 based machines, but none for my shitty old A2000 ! The other SoftKickers like SKick and MKick couldn’t convince me, either (the Patchtables were not up to date for Kick 3.1).

1.4 Requirements

Requirements:

CpuControl is designed to be used with MC68020/68030 processors (Commodore’s [R.I.P.] accelerator boards A2620 and A2630 are perfectly fit). If it is started on a system with a 68000/68010 or 68040, it quits with an error message. At this point you might ask why your MC68040 processor is not supported. Well, the reason is that especially MMU coding is quite different from the ’030. Since I don’t own a 68040 equipped machine (yet) and the necessary documentation, I am not able to support this processor at the present time.

Here’s now what you need to use ALL features of CpuControl:

o OS 2.04 or better

o MC68020 or MC68030 (FPU 68881 or 68882 optional)

o MMU 68851 or internal 68030 MMU

o 2 MB 32Bit FastMem (the more, the better !)

o 512K ChipMemory (notice that a softkicked OS might want more !)

o any Chipset (OCS, ECS, AA)

Please note that a functional MMU is needed for full functionality. On 68020 systems without 68851 or on 68EC030 (EC = economy ?) systems you can not use the FASTROM and KICK options ! CpuControl is able to detect a missing or non-functional MMU and quits with a warning if you try to use this options.

If you want to use the KICK option, you need a valid ROM image that meets the following conditions:

o the OS version must be greater or equal to OS 2.04

o the file must contain plain Kickstart data, i.e. no Superkickstart files or ZKick files are supported at the moment

o the Kickstart image MUST be fit for your particular system, i.e. you cannot use an A4000 kickfile on an A2000 ! This might change in future releases, although it is not very likely, since it requires heavy patches in the Kickstart code.

The easiest way to get a suitable kickfile is to run the 'SaveROM' tool that comes along with this distribution on an Amiga that is equipped with the required ROM version. This tool reads the contents of the installed ROM and writes it into a specified file. Be sure that no FASTROM option is active, because some software might patch the Kickstart in RAM without correcting the checksum (the software of the Piccolo gfx board does so when used with OS 2.xx in order to use 256 colors in the WB emulation). This causes a checksum error when softkicking the OS.

IMPORTANT NOTES: THE KICKSTART SOFTWARE IS COPYRIGHTED BY AMIGA TECHNOLOGIES. IT IS ILLEGAL TO USE COPIES OF THE KICKSTART SOFTWARE WITHOUT PERMISSION OF THE COPYRIGHT OWNER. USING ILLEGAL COPIES OF THE KICKSTART STUFF IS THEREFORE NOT ENCOURAGED BY ME, EVEN THOUGH CPUCONTROL OFFERS YOU THE POSSIBILITY TO DO SO ! THIS DISTRIBUTION DOES NOT CONTAIN ANY KICKSTART FILES.

As you may have noticed, CpuControl does not support Kickstart versions below 2.04. I thought about programming a version of CpuControl that works with Kickstart 1.x, but I received only ONE mail with a request for Kick 1.x support. Due to this I decided not to support these obsolete 1.x OS version any longer (sorry). Another reason for this is that my old Kick 1.3 ROM does not work with my Apollo accelerator board :(.

1.5 How to contact

How to contact me:

Here's my address for those who feel like contacting me:

Oliver Goss
Gnesener Str. 4
93057 Regensburg

Germany

By now I can also be reached directly by email !

Email: goss.oliver@rbgs344.rbg1.siemens.net

1.6 Usage

Usage:

CpuControl is a shell-based program, i.e. it cannot be started from the Workbench (you can use 'Execute Command' from the Workbench menu if you don't want to open a shell window).

CpuControl supports three different command line templates. Only one valid template can be used at one time, i.e. the arguments of one template must not be mixed with another template. If you use arguments that are not supported by your system, you will receive an error message in some cases, e.g. when the FASTROM option is used on a 68EC030 system. All Cache and Burst related arguments adapt automatically to your system.

TEMPLATE 1:

```
CpuControl CACHE/S,NOCACHE/S,DCACHE/S,NODCACHE/S,ICACHE/S,NOICACHE/S,
           BURST/S,NOBURST/S,DBURST/S,NODBURST/S,IBURST/S,NOIBURST/S,
           FASTROM/S,NOFASTROM/S,MOVEVBR/S,RESETVBR/S.
```

This command template is similar to the CPU command. It covers most of CpuControl's features.

Switches/Keywords:	Function:
no arguments	Reports some status information (Just try it).
?	Display some help.
CACHE	Enable CPU Caches.
NOCACHE	Disable CPU Caches.
DCACHE	Enable Data Cache.
NODCACHE	Disable Data Cache.
ICACHE	Enable Instruction Cache.
NOICACHE	Disable Instruction Cache.
BURST	Enable Bursts (Data and Instruction).

NOBURST	Disable Bursts
DBURST	Enable Data Burst.
NODBURST	Disable Data Burst.
IBURST	Enable Instruction Burst.
NOIBURST	Disable Instruction Burst.
FASTROM	Move Kickstart ROM into 32-Bit RAM (MMU required !)
NOFASTROM	Remove FASTROM.
MOVEVBR	Move VBR into 32-Bit RAM.
RESETVBR	Reset VBR to location $00000000.

Example: The command line ’CpuControl CACHE BURST FASTROM MOVEVBR’ enables all Caches and Bursts and moves both the Kickstart and the Vector Base into FastRAM, boosting the system performance to the maximum.

TEMPLATE 2:

```
CpuControl  KICK/A FILE/A
```

This command template is used to softkick a Kickstart file. Please note that the full path to the Kickfile has to be given. A functional MMU is required to use this option !

Example: The command line ’CpuControl KICK Devs:Kickstarts/Kick3.1’ tries to load the specified Kickstart and resets the machine afterwards. It is not possible to load the same OS version as it is found in the ROM. If you try to do so, CpuControl just quits, thus giving you the possibility to use it as the first command in your startupsequence, i.e. there’s no need to check if the desired OS has already been softkicked.

TEMPLATE 3:

```
CpuControl  UNKICK/A
```

This command template is used to remove a softkicked OS. All other RomTags remain untouched. Any residents hooked in via ColdCapture or CoolCapture are removed, though.

Example: The command line ’CpuControl UNKICK’ kills the KICK Romtag and reboots the computer with the ROM Kickstart.

1.7 Compatibility

Compatibility:

CpuControl has been successfully tested on the following configurations. Please note that some features are not available on certain systems.

My old system:

- A2000, 1M Chip memory, 2 x 3.5" floppy drives, Kick 1.3 & 2.04
- A2630 with 4M 32-Bit memory, tuned 68882 with 30 MHz
- Piccolo Graphics Board with 2M memory
- Nexus SCSI with Quantum LP105S and LP52S drives
- Flickerfixer (DeInterlace Card)
- Vortex GoldenGate 386SX PC-Emulator with 4.5M RAM,external 3.5" HD Floppy, Seagate ST3243A harddisk, TARGA SVGA board and serial card.

My upgraded system:

- A2000, 1M Chip memory, 2 x 3.5" floppy drives, Kick 1.3 & 2.04
- Apollo A2030 68030/68882/50/50 with 16M 32-Bit memory and Apollo SCSI-Controller
- External SCSI-Tower with Toshiba XM-3701B CD-ROM, IBM DPES31080S Deskstar XP1080MB & Quantum LP105S harddisks
- Piccolo Graphics Board with 2M memory
- Flickerfixer (DeInterlace card)
- Vortex GoldenGate 386SX PC-Emulator with 4.5M RAM, internal 5.25" HD diskdrive, external 3.5" HD diskdrive, Seagate ST3243A harddisk, TARGA SVGA board and Multi-IO board.

CpuControl seems to work fine with the following MMU based Software:

- GigaMem
- Enforcer

1.8 Distribution

Disclaimer:

No warranties of any kind are made as to the funtionality of this program. You are using it at your own risk, i.e. the author can not be held liable for data loss or any other kind of damage caused by the use of CpuControl.

Copyright:

All files in this distribution are Copyright © 1995 by Oliver Goss. The author retains all rights to the program.

CpuControl is distributed as CARDWARE. If you find the program useful, you are encouraged to send me a nice postcard. Please note that CpuControl is definitely NOT Public Domain Software, i.e. you are not allowed to alter the contents of this distribution in any way (see below !).

Distribution:

CpuControl may be freely distributed in any way, as long as the following conditions are met:

1. No fees may be charged for its distribution, except a reasonable charge for media and shipping, etc.

2. The distribution must contain all files as provided by me in the original archive.

3. The files contained in this distribution MUST NOT be changed nor be used (complete or in part) in other products. It is permitted to add faithful translations of the documentation to the archive as long as the original documentation is enclosed.

4. CpuControl must not be used in commercial products.

If you have a question about the distribution or copyright stuff that is not covered by the above lines, please feel free to contact the
author
:) .

1.9 Features

Features:

CpuControl offers you the following features (random order):

o Complete Cache and Burstmode control

o FastROM option

o Ability to move (and remove) the Vector Base into FastRAM

o Smarter Command Line interface compared to stock CPU command

o Comprehensive error messages

o Nice status message

o Ability to softkick any OS version from 2.xx to 3.xx

o ColdCapture and CoolCapture vectors are not used

o 100% handcrafted 68020+ assembler code

1.10 Author's notes

Author's Notes:

I developed CpuControl to meet my personal needs and thus it might be completely useless to you. However, this piece of software has been released in the hope that some Amiga users have a need for it. So if you use CpuControl from time to time, please drop me postcard.

Special thanks go to the following persons:

- Christian Rotter for providing his email address

- Stefan Thielscher and Peter Simons for their articles 'Generationswechsel' in the AmigaPlus magazine. Especially the MMU related articles were indispensable for the development of CpuControl.

- Daniel Zenchelsky for ZKick V3.01. The sourcecode was a great help when I created the KICK option.

- Greg Tibbs for SoftBoot V3.31. Again the sourcecode gave me a lot of inspiration ;) .

NOTE:

CpuControl was in part inspired by sourcecodes from Greg Tibbs, Daniel Zenchelsky and Stefan Thielscher. The code it
self is my own design, i.e. no parts have been copied. Nevertheless some resemblance may occur in certain cases.

1.11 Technical Details

Technical details:

This paragraph is intended for the programmers among you. It offers some background information over the implementation of the FASTROM option, the KICK option and some other stuff. The CACHE/BURST/... options are quite simple, so I decided not to bore you with an explanation about how to disable the processor caches etc. ;) . However, this text does not offer much information about how to program the MMU or something like that. That would have been beyond the scope of this documentation.

One important feature of CpuControl is its ability to detect the missing or non-functional MMU in an 68EC030 processor. This is not too easy to achieve, since the MC68EC030 has a functional TC register although the MMU itself may be less than functional.

Another possibility that has to be considered is a 68020 based system without MMU (as it is the case with most 68020 boards or in an Amiga 1200). The MMU detection code must therefore be able to handle the LINE-F exception caused by the execution of a MMU instruction like PMOVE.L TC,(A0) on 68020 systems without MMU. This is achieved by a small exception handler that redirects the LINE-F vector temporarily to some code that catches the exception and sets an appropriate flag to FALSE. The return address on the supervisor stack is then modified so that the instruction that caused the exception is skipped when the handler returns to the normal program flow. After that the LINE-F vector is restored with its former contents.

At this point we know whether MMU instructions can be executed or not. But the fact that a MMU instruction can be executed without raising an exception does not necessarily mean that a functional MMU is available. So we have to perform another test to be sure that the MMU is really functional. Therefore CpuControl sets up a very simple MMU table that maps the whole address space of the processor (4 Gigabyte), using two Early Termination descriptors. Any memory access *should* now set the Used Bit in at least one of the descriptors. If this is the case, we can be (almost) sure that the MMU is functional. However, this test may not be 100% reliable. (The idea for this test has been taken from Stefan Thielscher's FastROM program as published on the AmigaPlus Disk 9/94. Hope you don't mind, Stefan ;) .) I have received some mails which reported strange crashes when starting CpuControl (Guru #80000038). Under normal conditions this error did not occur on my (old A2630) system. But when I got my Apollo board I noticed the same Guru especially during a coldstart. I couldn't trace the reason for this behaviour completely, but it was obviously caused by the MMU detection routine. As far as I can tell it was some sort of a caching problem, because after I disabled the caches during the MMU test, no more crashes happened ;-) .

If these initial tests have been successfully passed, you are able to use the FASTROM and the KICK option, as long as your MMU is not yet occupied by another program.

The principle of the FASTROM option is not very difficult to understand. CpuControl just tries to allocate a sufficient buffer in your FastMem area and copies the ROM contents into this buffer. The buffer is received via AllocMem with the MEMF_REVERSE flag set, i.e. AllocMem tries to allocate the buffer at the end of your FastMem area in order to reduce memory fragmentation since the buffer MUST be aligned to a 32 KByte boundary due to the used pagesize in the MMU table.

(NOTE: Ralph Babel writes in his Guru Book that the MEMF_REVERSE option does not work reliably on 2.xx systems. I didn't discover any difficulties with this option, but if you do so, please let me know.)

OK, let's go on. The next thing to do is to allocate a small buffer for the MMU table and to generate the necessary translation tree. I used the maximum pagesize of 32 KByte here to get a small MMU table. After that the translation tree is activated and any access to the ROM area at $F80000 is now redirected to the FastMem buffer with the ROM image. Please note that the FASTROM option is useful ONLY on Amigas that are originally based on the 68000 processor like the A500(+), A2000, CDTV. The newer Amiga models already have 32 Bit access to the ROM. Well, there's now only one thing more to say about the FASTROM option. The FASTROM options of CpuControl and the stock CPU command are NOT compatible, i.e. you can not remove a CPU FASTROM with a CPUCONTROL NOFASTROM ! CpuControl recognizes foreign MMU tables since all of its own MMU tables contain a special identification longword.

Now we come to the most advanced feature of CpuControl, namely the KICK option. This option required a lot of time and headscratching to be implemeted. It is strongly inspired by the sourcecodes of ZKICK by Dan Zenchelsky and SOFTBOOT by Greg Tibbs, but the implementation is my very own and no code has been copied from the named sources. The KICK option is in part similar to the FASTROM option. The great difference that has to be taken care of is that it is not sufficient to just load another Kick version into a buffer and to redirect the ROM accesses to this buffer as it is done by the FASTROM option. Here we must rebuild *all* system structures according to the requirements of the new Kickstart. To achieve this, we have to reset the computer and force it to enter the restart code in the loaded Kick image. This is far more difficult as it sounds, as I must discover. The greatest problem is that it is not possible to perform a proper warmboot (reset) when the MMU is activated. You may have noticed this when you tried to use exec's ColdReboot routine with an activated FASTROM. I tried to find the reason for this behaviour, but I'm not sure whether my explanation is correct or not. Let me know if you have a better one.

The critical instruction in this case is the CPU's RESET command. Whenever this instruction is executed while the MMU is active, the computer crashes. At first I thought that the RESET instruction also disables the MMU, but this is not so according to the Motorola User Manuals. But why does the dratted machine crash when the MMU actually remains activated ? I pondered this for a while and my explanation is the following: The RESET instruction is necessary to reset all expansion boards in the computer. It triggers the _hardware_ autoconfig process that renders the expansion devices temporarily unaccessible until the autoconfig process has been completed. Unfortunately the 32 Bit memory on the A2630 board is an autoconfig memory device and thus the MMU table as well as the loaded ROM image vanishes into thin air for a short period of time during the reset process !

So what can be done to prevent this ? The easiest way is to just leave out the RESET instruction in the restart routine and to jump directly into the ROM restart code. This seems to work, BUT the expansion devices are not reconfigured when the RESET instruction is left out, i.e. there's no FastRam, SCSI, Gfx board etc. available when the computer reboots into the new OS. This cannot be borne.

After some more pondering I devised the following solution: CpuControl hooks a RomTag structure into the system that contains some special reboot code. The RomTags are gathered and executed by the system's boot code after the expansion devices have been reconfigured (this is not so when you use a KickStart version below 2.0x. See Ralph Babel's Guru Book for further details !!! [This is also a reason why there is no Kick 1.x version of CpuControl]). Before CpuControl restarts the computer it gathers some vital information concerning the installed expansion devices via the expansion.library. This information is stored in the protected RomTag area in ChipMem (ChipMem is non-autoconfig memory, i.e. it is ALWAYS acessible). After that CpuControl restarts the machine. At this point of time the MMU is NOT activated, so that the important RESET instruction can be executed without crashing the computer. Thus your expansion devices are properly reconfigured (by hardware). After that the RomTag is executed for the first time. It sets up a MMU translation tree and activates the address translation to replace the ROM with the loaded ROM image. The RomTag code then jumps directly into the reboot code of the ROM image and restarts the machine once again (without RESET instruction, mind you !). Now the RomTag becomes executed once more. It uses the stored information about the expansion devices to link them into the system again by the use of the expansion.library. This routine also adds all memory boards to the system's MemList.

The final step in the versions < V1.06 was then to re-allocate the memory containing the ROM image via 'AllocAbs'. But when I got my Apollo accelerator board, I noticed that the re-allocation of the memory failed on this new environment. The machine always crashed during execution of the 'startup-sequence'. The reason for this is quite easy: the Apollo board's memory is not located inside the Zorro II address space and thus it is non-autoconfig memory which has to be added to the system by the use of a special driver in the Apollo startup software (located in an EPROM). This kind of drivers is activated during the execution of Kickstart's resident module named 'diag.init' AFTER the 'expansion.library' has been configured. So my own RomTag had never a chance to find this memory ! The remedy for this is quite simple. I only had to hook in another RomTag which is executed after 'diag.init' and then things worked well again. (This also is a big compatibility enhancement, since all boards that configure their non Zorro II memory in this way (GVP ?) should work now correctly). After this, the second RomTag gives back control to the system.

Now the computer starts up with the new OS and all expansion devices are available again ! The only aftertaste that remains is the ExecBase structure being left in Chip memory. This lowers the system's performance considerably ;) (about 0.5% on my machine).

The Restart routines do not use the ColdCapture and CoolCapture vectors and the RomTag is linked into the system in a compatible way, i.e. foreign RomTag structures are recognized and properly queued into the list of KickTag pointers.

A kicked ROM image survives crashes and resets as long as the execbase structure is not damaged or destroyed.

Well, if you think you need more information about this stuff then contact the

author.

1.12 To do ...

To do ... :

This paragraph contains a list with features I want(ed) to add to CpuControl. It depends on the feedback I get from YOU if these features become implemented in future releases. Any suggestions of functions you want to have added are always welcome.

To-Do list (random order):

- FORCE switch for KICK option to override an already activated MMU

- Kickstart patching to remove errors etc.

- enhanced command line parser

- GUI-based version (maybe)

- 68040/68060 support

- relocate Execbase to Fastmem

- messages have to be localized

- FLUSH command to clean up memory

```
- some special fixes for Apollo boards

and some more I cannot recall at the moment.


Some words concerning 68040/68060 support. To implement this I  need  some  more
information  about  68040/68060.  So  if  YOU  have  Motorola's  manuals I would
appreciate if you sent me some copies of the  pages  concerning  Cache  and  MMU
handling !!!
```

1.13 Troubleshooting

```
                Troubleshooting:

This paragraph is designed to shed some  light  on  common  problems  you  might
discover whilst using CpuControl.

First of all a list of CpuControl's error messages with hints about what  is  to
be done when you stumble upon such a message:

Message: 'Only 68020 and 68(EC)030 CPUs are supported'

Meaning: CpuControl discovered a 68000 or 68040+ CPU.

Remedy:  CASE 68000,68010 -=> buy a turboboard
         CASE 68040,68060 -=> use Softboot



Message: 'Not enough memory for FASTROM option'

Meaning: Just what it says. You'll need at least a 512K block (non-fragmented !)
         FastMem  plus  some  600 Bytes for the MMU table. I think it's not much
         use to activate the FASTROM option with less than 2 Megs of FastRam.

Remedy:  Plug in more memory.




Message: 'Cant allocate sufficient memory for KICK option'

Meaning: See above.

Remedy:  See above.




Message: 'Cant allocate sufficient memory for the RomTag'

Meaning: There's not enough ChipMem available for the RomTag-Code. Less than  2K
         of  ChipMem are needed to install the RomTag, so this message shouldn't
         appear too frequently.
```

Remedy: Close some windows or screens, stop playing sound modules or samples,
         remove any backdrop pictures on your WB screen, etc.

Message: 'Not enough FastMem for option MOVEVBR available'

Meaning: Just what it says. 1028 Bytes are required.

Remedy: Stop some running programs.

Message: 'Too many arguments'

Meaning: What could that mean ??? Ah, yes, you gave too many arguments on the
         command line.

Remedy: RTFM to figure out how to provide a decent command line.

Message: 'Invalid argument(s)'

Meaning: Obviously some crap was discovered while parsing the command line

Remedy: Check your typo !

Message: 'Contradictory arguments'

Meaning: You used a nonsense combination of arguments like CACHE NOCACHE.

Remedy: Re-think your comand line.

Message: 'Missing filename for argument KICK'

Meaning: No need to explain ...

Remedy: Provide a valid filename (+path) when using the KICK-option.

Message: 'You need a functional MMU to use FASTROM'

Meaning: Seems like you have a MC68EC030 CPU ! Or your MMU/CPU is defective (or

```
         you’ve discovered a bug in CpuControl’s recognition code)

Remedy:  Replace your CPU or buy another accelerator board.



Message: ’A MMU is required to use the KICK option’

Meaning: Like above ...

Remedy:  Like above ...



Message: ’Your MMU is already in use’

Meaning: Some other program has already installed a MMU translation tree.  Check
         out if the stock CPU command has been executed with the FASTROM keyword
         or if some other MMU related programs like GigaMem, Enforcer,VMem, etc.
         have been activated.

Remedy:  Start CpuControl before any other program that uses the MMU.



Message: ’Cant execute NOFASTROM’

Meaning: CpuControl discovered that your MMU is activated, but  the  translation
         table  was not built by CpuControl’s FASTROM code. This message appears
         when the stock CPU command has installed its FASTROM.

Remedy:  Try to remove the FASTROM with CPU NOFASTROM.



Message: ’Cant access specified file’

Meaning: The file you named for the KICK option can not be found.

Remedy:  Check filename and path.



Message: ’Invalid KickStart file’

Meaning: The file you want to softkick is obviously not a valid ROM image.

Remedy:  Get a VALID image using the SaveROM program.
```

Message: 'Cant open file'

Meaning: The file you named for the KICK option can not be opened.

Remedy: Check the file.

Message: 'Could not read from file'

Meaning: There were errors while reading from the file.

Remedy: Check the file for errors.

Message: 'Cannot reset VBR'

Meaning: The VBR table cannot be removed from FastMem since it was not moved by CpuControl's MOVEVBR command.

Remedy: Check if some other program was executed that moves the VBR into FastMem (e.g. CPU with FASTROM option)

Message: 'There is already a kickfile installed'

Meaning: A kickfile has already been softkicked using CpuControl.

Remedy: Kill the softkicked ROM image using CpuControl's UNKICK option and try again.

Message: 'Cant open expansion.library'

Meaning: Just what it says. Shouldn't happen at all, though.

Remedy: ???

Common problems with CpuControl:

* Can't remember any at the moment

Please report any misbehaviour and malfunctions to one of the given addresses

.

1.14 History

```
History:

This part contains an chronological overview of CpuControl's development process
with information about changes, revisions, removed and discovered bugs, etc.



V0.01     Date:   07/02/95

          This revision and all later revisions up to  V1.00 remained unreleased
          due to the fact that they were under constant development.



V1.00     Date:   27/04/95

          First version  with  all  functions  completely  implemented.  Kicking
          another  OS  version  works  at least on my machine. Nevertheless this
          release needs extensive testing and a rework in some aspects.

          This version already works fine with GigaMem and all of  my  currently
          installed expansion hardware (quite a lot).

          Known Bugs: - Any program  calling  exec's  'ColdReboot'  leads  to  a
                        system  deadlock in the reboot process. A keyboard reset
                        revives the system in this  case  *without*  losing  the
                        kicked OS !

                      - A similar problem occurs when the Bootmenu is  activated
                        (by pressing both mousebuttons during a reset). A system
                        deadlock occurs after the Menu has been closed.  Another
                        keyboard reset revives the system with the manipulations
                        done in the menu taking effect.



V1.01     Date:   01/05/95

          Slightly improved version.

          Changes:    - 'LVOColdReboot' is set to my own routine when the RomTag
                        is executed. Any program using 'Coldreboot' now triggers
                        a proper reset  without  system  deadlock.  The  library
                        vector  is  changed  in a (more or less) system friendly
                        manner via 'SetFunction'.

          Known Bugs: - The Bootmenu still doesn't work correctly. This might be
                        solved by patching the ROM image.
```

```
V1.02      Date:   02/05/1995

           Added options MOVEVBR and RESETVBR to  enhance  the  performance  when
           handling  interrupts  and  exceptions.  Improved  program  info string
           (option '?').

           Known Bugs: - Performance drops sharply  (due  to  SysInfo)  when  the
                         CACHE  option is used. Obviously the Freeze-Bits are set
                         by accident, rendering the caches useless (Aaarrrrrgh).



V1.03      Date:   03/05/1995

           Performance loss when  using  CACHE  option  has  been  corrected.  No
           further Bugs have been corrected :( .



V1.04      Date:   04/05/1995

           'MMU Status' string now displays information  about  the  current  MMU
           user.  Only  CpuControl's own MMU tables are currently recognized, all
           others are titled 'unknown user'. Again not much improvement :( .



V1.05      Date:   08/05/1995

           Changes:    - Patching of _LVOColdReboot via  'SetFunction'  has  been
                         removed  since  it  didn't  work  at  all when using the
                         Bootmenu. Now the ColdReboot code  in  the  Kickfile  is
                         manipulated  to  meet  our needs. Result: Bootmenu works
                         fine now ;) .

           To do:      - Command line parsing while using templates 1 &  2  needs
                         some  rework.  At  least one additional switch has to be
                         implemented for the KICK option.



V1.06      Date:   11/03/1996

           Changes:    - As I got my Apollo  68030/68882/50/50  board  some  time
                         ago,  I noticed that the memory containing the ROM image
                         was not re-allocated when resetting the machine, leading
                         to  crashes  during execution of the 'startup-sequence'.
                         Added a second RomTag to fix this (see documentation).

                       - Another big bug fixed: When  NO  Zorro  II  FastMem  was
                         found, the memory for the ROM image wasn't re-allocated.
                         This has been fixed with the  above  mentioned  measure.
```