MuFastRom

Thomas Richter

COLLABORATORS			
	TITLE : MuFastRom		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Thomas Richter	January 13, 2023	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

MuFastRom

1.1 MuFastRom Guide

´### ,#. ,#### ,### ####' #### ,#### ,###' ####' #### _____ ,#### ,###' | | ####' #### | ___ ___ | ___ ,#### ,###' ---- | | |
| | | |___ ####' #### | | | | | | | ,#####. ,###' . | |___| |___| |_ ___| ####'##. ,#### ,## ,#### ########' # ,##' ####' `####' `###'
,#### ##### © 1999-2002 THOR - Software, ####' Thomas Richter `##'

_____

MuFastRom Guide

Guide Version 1.09 MuFastRom Version 40.9

The Licence : Legal restrictions

MuTools : What is this all about, and what's the MMU library?

What is it : Overview

Installation : How to install MuFastRom

Synopsis : The command line options and tool types

History : What happened before

© THOR-Software

Thomas Richter

Rühmkorffstraße 10A

12209 Berlin

Germany

EMail: thor@math.tu-berlin.de

1.2 The THOR-Software Licence

The THOR-Software Licence (v2, 24th June 1998)

This License applies to the computer programs known as "MuFastRom" and the "MuFastRom.guide". The "Program", below, refers to such program. The "Archive" refers to the package of distribution, as prepared by the author of the Program, Thomas Richter. Each licensee is addressed as "you".

The Program and the data in the archive are freely distributable under the restrictions stated below, but are also Copyright (c) Thomas Richter.

Distribution of the Program, the Archive and the data in the Archive by a commercial organization without written permission from the author to any third party is prohibited if any payment is made in connection with such distribution, whether directly (as in payment for a copy of the Program) or indirectly (as in payment for some service related to the Program, or payment for some product or service that includes a copy of the Program "without charge"; these are only examples, and not an exhaustive enumeration of prohibited activities).

However, the following methods of distribution involving payment shall not in and of themselves be a violation of this restriction:

(i) Posting the Program on a public access information storage and retrieval service for which a fee is received for retrieving information (such as an on-line service), provided that the fee is not content-dependent (i.e., the fee would be the same for retrieving the same volume of information consisting of random data).

(ii) Distributing the Program on a CD-ROM, provided that

a) the Archive is reproduced entirely and verbatim on such CD-ROM, including especially this licence agreement;

b) the CD-ROM is made available to the public for a nominal fee only,

c) a copy of the CD is made available to the author for free except for shipment costs, and

d) provided further that all information on such CD-ROM is re-distributable for non-commercial purposes without charge.

Redistribution of a modified version of the Archive, the Program or the contents of the Archive is prohibited in any way, by any organization, regardless whether commercial or non-commercial. Everything must be kept together, in original and unmodified form.

Limitations.

THE PROGRAM IS PROVIDED TO YOU "AS IS", WITHOUT WARRANTY. THERE IS NO WARRANTY FOR THE PRO-GRAM, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT OF THIRD PARTY RIGHTS. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

IF YOU DO NOT ACCEPT THIS LICENCE, YOU MUST DELETE THE PROGRAM, THE ARCHIVE AND ALL DATA OF THIS ARCHIVE FROM YOUR STORAGE SYSTEM. YOU ACCEPT THIS LICENCE BY USING OR REDISTRIBUTING THE PROGRAM.

Thomas Richter

1.3 What's the MMU.library?

All "modern" Amiga computers come with a special hardware component called the "MMU" for short, "Memory Management Unit". The MMU is a very powerful piece of hardware that can be seen as a translator between the CPU that carries out the actual calculation, and the surrounding hardware: Memory and IO devices. Each external access of the CPU is filtered by the MMU, checked whether the memory region is available, write protected, can be hold in the CPU internal cache and more. The MMU can be told to translate the addresses as seen from the CPU to different addresses, hence it can be used to "re-map" parts of the memory without actually touching the memory itself.

A series of programs is available that make use of the MMU: First of all, it's needed by the operating system to tell the CPU not to hold "chip memory", used by the Amiga custom chips, in its cache; second, several tools re-map the Kickstart ROM to faster 32Bit RAM by using the MMU to translate the ROM addresses - as seen from the CPU - to the RAM addresses where the image of the ROM is kept. Third, a number of debugging tools make use of it to detect accesses to physically unavailable memory regions, and hence to find bugs in programs; amongst them is the "Enforcer" by Michael Sinz. Fourth, the MMU can be used to create the illusion of "almost infinite memory", with so-called "virtual memory systems". Last but not least, a number of miscellaneous applications have been found for the MMU as well, for example for display drivers of emulators.

Unfortunately, the Amiga Os does not provide ANY interface to the MMU, everything boils down to hardware hacking and every program hacks the MMU table as it wishes. Needless to say this prevents program A from working nicely together with program B, Enforcer with FastROM or VMM, and other combinations have been impossible up to now.

THIS HAS TO CHANGE! There has to be a documented interface to the MMU that makes accesses transparent, easy and compatible. This is the goal of the "mmu.library". In one word, COMPATIBILITY.

Unfortunately, old programs won't use this library automatically, so things have to be rewritten. The "MuTools" are a collection of programs that take over the job of older applications that hit the hardware directly. The result are programs that operate hardware independent, without any CPU or MMU specific parts, no matter what kind of MMU is available, and programs that nicely co-exist with each other.

I hope other program authors choose to make use of the library in the future and provide powerful tools without the compatibility headache. The MuTools are just a tiny start, more has to follow.

1.4   What's the job of MuFastRom?

MuFastRom is a mmu.library compatible ROM remapper. It creates a mirror image of the Kickstart ROM in (hopefully faster) RAM and hence speeds up your system. The mirrored image gets write-protected to prevent programs from overwriting the "ROM" in RAM.

1.5   Installation of MuFastRom

Installation is pretty simple:

- First, install the "mmu.library": Copy this library to your LIBS: drawer if you haven't installed it yet. It's contained in this archive.

- Copy "MuFastROM" wherever you want.

- Remove all other ROM remappers from your startup-sequence and add the following line:

MuFastRom ON

to enable ROM re-mapping. That's all.

1.6   Command line options and tooltypes

MuFastROM can be started either from the workbench or from the shell. In the first case, it reads its arguments from the "tooltypes" of its icon; you may alter these settings by selecting the "MuFastRom" icon and choosing "Information..." from the workbench "Icon" menu. In the second case, the arguments are taken from the command line. No matter how the program is run, the arguments are identically.

MuFastRom ON=FASTROM/S,OFF=NOFASTROM/S,HEAD/S,PROTECT/S,NOPROTECT/S

_____

ON=FASTROM

A simple switch. If present, ROM re-mapping is enabled. MuFastRom will complain if it is already installed, or the ROM is already re-mapped by a different tool. This is the default if "OFF" is missing.

_____

OFF=NOFASTROM

Another switch. If given, ROM re-mapping is disabled again and the mirror image is released.

_____

HEAD

If this switch is present, MuFastRom allocates the memory for the ROM mirror image at the lower end of the fast memory pool, i.e. at lower addresses. The default is to allocate the memory from the upper end of the pool to avoid fragmentation. This option might be useful if exec merged two memory blocks, slower 16 bit memory and faster 32 bit memory into one block and the ROM should go to the fastest possible memory.

_____

_____

PROTECT

Enables a protection mechanism for the re-mapped ROM and for the mirror RAM from getting overwritten. This mechanism is a defensive silent protection by default, unless a debugging tool already selected a more aggressive protection which will cause "hits" if a write into the "ROM" is attempted.

By default, MuFastROM uses whatever protection was turned on in first place.

_____

NOPROTECT

Disables any protection of the mirror image, e.g. to allow debuggers to set breakpoints in the "ROM".

By default, MuFastROM uses whatever protection was turned on in first place.

_____

When started from the workbench, MuFastRom knows one additional tooltype, namely:

WINDOW=<path>

where <path> is a file name path where the program should print its output. This should be a console window specification, i.e. something like

CON:0/0/640/100/MuFastRom

This argument defaults to NIL:, i.e. all outputs will be thrown away.

_____

1.7 History

Release 40.2:

This is the first official release. I would call "MuFastROM" pretty final because there's not much this program has to do. All the tough tricks are run by the mmu.library anyways. However, the "mmu.library" is still in beta state.

Release 40.3:

Added the HEAD,PROTECT and NOPROTECT options.

Release 40.4:

Fixed PROTECT and NOPROTECT options, used to be a bit messed up.

Release 40.5:

I don't know what drove me crazy to remove the PROTECT option in 40.4. Fixed!

Release 40.6:

The ROM remapper will now copy over the cache settings from the re-mapping source to the ROM image. This is especially important if the ROM mirror is placed in non-cacheable Z-II memory.

Release 40.7:

"ON" is now the default option if no other option was given on the command line invocation.

Release 40.8:

MuFastRom did not unload itself if run from the workbench. Fixed. The MuFastRom code is now a bit more error tolerant.

Release 40.9:

MuFastRom is now able to write-protect the ROM mapped by MuMapRom.