HRTmon

Alain Malek

COLLABORATORS			
	TITLE : HRTmon		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Alain Malek	August 9, 2024	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

HRTmon

1.1 HRTmon documentation

HRTmon v1.0 Documentation

1.2 HRTmon/Introduction

HRTmon

Copyright © 1991 - 1995

All Rights Reserved

Written by

Alain Malek

Have you ever wanted to get an Action Replay on your A1200, A3000 or A4000 ?

If the answer is yes, then you should have a look at HRTmon.

HRTmon is a monitor for your Amiga, which doesn't use the libraries.

You can invoke the monitor at any point. Even if a game or a demo switched

off the interrupts.

Once in HRTmon you can watch all the memory, disassemble, edit, save, etc...

(Have a look at the commands list).

and continue, as if nothing happend.

You don't have to care about any picture address. ALL the CHIP-memory can

be edited transparently.

To work perfectly HRTmon requires a 'magic' level7 button.

(This button is recommanded but you can use HRTmon without it.)

NOTE:

The demo version lacks some features:

- saving of files.
- assemble.
- all trainer functions.

HRTmon is shareware software. If you find it useful, please register to get the full version. Please refer to the How to Register section of this documentation.

Please notice that you are not entitled to use the demoversion on a regular basis without registration. The demoversion is only for evaluation of the program. If you don't use it, delete it. If you use it, register.

Your name and a personal code will be inserted in the registered version of HRTmon.

1.3 HRTmon/Requirements

REQUIREMENTS

Any Amiga with Kickstart 2.04 (or greater).

68020+ processor.

A level7 interrupt button. (Recommanded)

Tested on:

o A1200

o MC68020,MC68030

o Kickstarts 39

o 2.0MB CHIPRAM, 0 - 4MB FASTRAM

If you find any bugs under your configuration, please contact me (address in the Disclaimer and Author Info section).

1.4 HRTmon/Installation

INSTALLATION

Simply copy these files in the same drawer.

hrtmon.data

HRTmonPrefs

HRTmonPrefs.info

HRTmon.guide

HRTmon.guide.info

Copy libs/reqtools.library to your libs: directory.

(If you don't have it already).

1.5 HRTmon/Usage

USAGE

HRTmonPrefs

Floppy disk launch

Enter HRTmon

The editor

The tracer

Commands list

Remove HRTmon

1.6 HRTmon/Usage/Tracer

Tracer

- You can enter the tracer by pressing the F7 key.

- To get out of the tracer select one of the editor pages by pressing

F8,F9 or F10

- To trace one single instruction press the right-arrow.

- To trace a BSR or a JSR instruction without tracing all the sub-routine

press the down-arrow. (The sub-routine will be executed but not traced.)

1.7 HRTmon/Usage/Editor

The Editor

General:

- When you enter HRTmon, you get directly in the editor.

- If the keyboard doesn't respond, just press your right-mouse button, to

reinit the CIA. (HRTmon doesn't do it automatically, to avoid modifying

some read-only registers.)

- The editor has 3 different pages. You can select one of these pages

by pressing the F8,F9,F10 function keys.

- To execute a command, you just have to type the command on any line

and press return to validate it.

- To repeat the same command without any parameter just press RETURN.

(Usefull when disassembling or dumping memory.)

- Use the arrows keys to move the cursor arround the screen.

- The editor can work in 2 modes: insert-mode ON or OFF.

When the insert-mode is OFF, you can still insert some blank spaces, by

pressing SHIFT+Del.

- To clear a single line, press SHIFT+backspace.

History:

- If you need to enter a command you used before, just press SHIFT+arrow-up

or SHIFT+arrow-down to get the last commands you have executed.

Expression:

- Each time you have to enter a number in a command, you can also enter

an expression.

- You can use a register in an expression.

Examples: h a0+4

d pc

- Use:

% for binary

$ for hexadecimal

# for decimal (default)

[..] for indirect addressing

Example: h [4] will display the ExecBase structure.

- To enter a string with spaces use : ’

Example: s ’mod.blue light’ $40000 $50000

1.8 HRTmon/Usage/Floppy

Floppy

To launch HRTmon from a floppy disk you have to create a boot-disk

with HRTmonPrefs.

This boot-disk makes possible to debug games or demos using ’NON-DOS’

disks.

You have to do the following :

- Boot the HRTmon disk.

- When the screen flashs insert the disk you want to debug.

- Press your left-mouse button to enter HRTmon.

You are now debuging the boot-block of the disk you inserted.

1.9 HRTmon/Usage/HRTmonPrefs

HRTmonPrefs

Starting HRTmonPrefs:

Type HRTmonPrefs from the CLI or simply click on the HRTmonPrefs icon.

HRTmonPrefs options:

Address

Enter the address where you want HRTmon to be located.

This address must be entered in hexadecimal.

If you don't have any idea where to locate HRTmon then type in this

address: $1d0000

AllocABS

When HRTmon is installed the memory used will automatically be

allocated through the AllocABS function of exec.library.

This behaviour can be turned off by this switch.

Make Boot-Disk

Click this button to create a disk which will load HRTmon at

'address', and allows you to boot from another disk.

(See Floppy disk launch)

Right-Mouse Enter

Select this option if you want to enter HRTmon by pressing on

your right mouse button. If you don't select this option, the

only way to enter HRTmon will be the level7 button.

Install

After having checked the Address, and AllocABS options click on

this button to install HRTmon in memory.

Saving HRTmonPrefs options:

Select 'Save' in the preferences menu.

1.10 HRTmon/Usage/Enter

ENTER

To enter HRTmon, simply press your level7 button or

the right-mouse button if you have selected the Right-Mouse Enter in

HRTmonPrefs.

If the keyboard doesn't work in HRTmon press the right mouse button.

(It shouldn't happen very often.)

HINTS:

HRTmon uses the VBR register to get transparent.

If you can't enter a demo or game, than try to trace the startup-code

and modify every instruction concerning the VBR instruction.

The programm shouldn't modify the VBR register, and every read of the

VBR register should return zero. (HRTmon emulates a VBR register equal to

zero).

1.11 HRTmon/Usage/Commands

COMMANDS

List of commands :

r view or edit CPU registers

a assemble 680x0

d disassemble 680x0

h hex dump/modify memory

n ascII dump of memory

type type from memory

c copy memory

q compare memory

o fill memory

ce exchange memory

f find

fi find instruction

p view memory

b set or remove breakpoint (view all)

bj set or remove JSR breakpoint

bd remove all breakpoints

t trace

ta trace till address reached

debug switch debug mode on/off

drive change current drive

motor switch drive motor on

format format FFS disk

qformat quick format FFS disk

dir dir

cd change current directory

l load file

s save file on FFS disk

makedir makedir

rs read sectors from disk

ws write sectors to disk

ra read all

wa write all

tset set trainer buffer

ts trainer start

tf trainer find

mmuon switch MMU on

mmuoff switch MMU off

custmmu copy of custom registers

output redirect output to memory

af assemble 65816

df disassemble 65816

fif find instruction 65816

cop find copperlist

esc view/modify escape registers

setmap change keyboard

pal force PAL display

ntsc force NTSC display

led switch power led on/off

? evaluate expression

x exit HRTmon

List of special keys :

F1 clear screen

F2 toggle insert mode on/off

F4 toggle USA/CH keyboard

F6 toggle 65802/65816 CPU mode (for disassemble)

F7 enter tracer

F8 view page 1

F9 view page 2

F10 view page 3

ESC break the actual command

HELP show commands list

See also : The Editor

1.12 HRTmon/Usage/Commands/CMD_R

CMD_R : registers

Description:

- View the processor registers, or modify a register.

Syntax:

r

or r reg val

Parameters:

reg = name of register to modify.

val = new value for the register.

Notes:

- register RTS shows you the return address if a RTS is executed.

if the PC points on a RTE instruction, you will see the 'RTE address'.

1.13 HRTmon/Usage/Commands/CMD_A

CMD_A : assemble

Description:

- Assemble MC680X0 code into memory.

Syntax:

a address instruction

Parameters:

address = address where the instruction must be assembled.

instruction = MC68000-MC68040/FPU/MMU instruction.

Notes:

- You have to enter the instruction with the new syntax.

Example: move.l d0,(4,a0) correct

move.l d0,4(a0) incorrect

move.l d0,($40000) correct

move.l d0,$40000 incorrect

- You have to enter the instructions in their original forme.

Example: cmpi.w #4,d0 correct

cmp.w #4,d0 incorrect

- If the assemble command succeeds it will automatically print 'a nextaddress'

on the next line.

If you want to stop assembling just press return.

Otherwise type in the next instruction.

1.14 HRTmon/Usage/Commands/CMD_D

CMD_D : disassemble

Description:

- Disassemble MC680X0 code from memory.

Syntax:

d address

Parameters:

address = address of the 680x0 code to disassemble.

Notes:

- 8 instructions from 'address' will be disassembled and printed.

1.15 HRTmon/Usage/Commands/CMD_H

CMD_H : hex dump

Description:

- Dump memory in hex and ascII, or edit memory.

Syntax:

h address ;hex dump

or h address val.x val.x val.x ... ;modify memory

or h address 'string' ;modify memory

Parameters:

address = address of the memory to dump or to modify

val = new values to write in memory.

x = size of val = b,w,l (default: w)

Examples:

h $40000 10.b $140.w $12345678.l

h $50000 'HRTmon'

1.16 HRTmon/Usage/Commands/CMD_N

CMD_N : ascII dump

Description:

- Dump memory in ascII only.

Syntax:

n address

Parameters:

address = address of the memory to dump.

1.17 HRTmon/Usage/Commands/CMD_TYPE

CMD_TYPE

Description:

- Type a text in memory in the editor.

Syntax:

type address

Parameters:

address = address of the memory to type.

Notes:

- the memory will be printed in the editor.

A zero will stop the command.

1.18 HRTmon/Usage/Commands/CMD_C

CMD_C

Description:

- Copy a block of memory.

Syntax:

c start end dest

Parameters:

start = start address of the block to copy.

end = end address of the block to copy.

dest = destination address for the block.

1.19 HRTmon/Usage/Commands/CMD_Q

CMD_Q

Description:

- Compare two blocks of memory to see if they are equals.

Syntax:

q start end start2

Parameters:

start = start address of the block to compare.

end = end address of the block to compare.

start2 = start address of the 2nd block to compare with the first one.

Notes:

- If the two blocks are different, the address of the first byte different from the 2nd block will be printed.
- If the two areas are equal, the following message will be printed: 'Equal areas.'

1.20 HRTmon/Usage/Commands/CMD_O

CMD_O

Description:

- Fill the memory with the same byte.

Syntax:

o start end val

Parameters:

start = start address of the block to fill.

end = end address of the block to fill.

val = value used to fill the block.

Notes:

- 'val' can only be a byte. (If you enter a word or a long it will be truncated to a byte.)

1.21 HRTmon/Usage/Commands/CMD_CE

CMD_CE

Description:

- Exchange two blocks of memory.

Syntax:

ce start end dest

Parameters:

start = start address of the block to exchange.

end = end address of the block to exchange.

dest = destination address for the exchange.

Notes:

- The two areas will be exchanged.

1.22 HRTmon/Usage/Commands/CMD_F

CMD_F

Description:

- Find a sequence of bytes,words,longs in memory.

Syntax:

f start end val.x val.x ...

Parameters:

start = start address for the search.

end = end address for the search.

val = value to find.

x = size (b,w,l)

Examples:

f $f80000 $f90000 $4ef9.w $f800d2.l

1.23 HRTmon/Usage/Commands/CMD_FI

CMD_FI

Description:

- Search for a string in the disassemble (MC680X0) memory.

Syntax:

fi start end ’string’

Parameters:

start = start address for the search.

end = end address for the search.

string = string to find.

Notes:

- don’t forget to use the new Motorola syntax.

- you can use the * character to replace any character.

Examples:

fi $f80000 $f90000 ’9a,a*)’

will find for example:

MOVE.W #$7FFF,($9A,A4)

1.24 HRTmon/Usage/Commands/CMD_P

CMD_P

Description:

- View the memory as a bitplane. If you want you can edit the bitplane.

Syntax:

p address

Parameters:

address = address of the top left corner of the memory to display.

Notes:

- press F10 to exit this mode.

- use arrows keys to move in memory.

press left Alt/Amiga to decrease/increase scroll speed.

- press ’M’/’N’/’,’ to increase/decrease/reset modulo.

- press F8 to get into edit mode. (a mouse cursor will be displayed)

In edit mode: Press left mouse button to set a pixel.

Press right mouse button to clear a pixel.

1.25 HRTmon/Usage/Commands/CMD_B

CMD_B

Description:

- Set or remove an Illegal breakpoint.

Syntax:

b address ;set,remove breakpoint

or b ;view all breakpoints

Parameters:

address = address of the breakpoint to set or remove.

1.26 HRTmon/Usage/Commands/CMD_BJ

CMD_BJ

Description:

- Set or remove a JSR breakpoint.

Syntax:

bj address ;set,remove JSR breakpoint

or bj ;view all breakpoints

Parameters:

address = address of the breakpoint to set or remove.

Notes:

- Will enter the monitor with a JSR instruction.

Don't forget that it will modify 6 bytes in memory to put a JSR.

- This feature was implemented for Amigas without a VBR register.

(Now HRTmon needs at least a 68020, so this command is obsolet.)

It doesn't need to modify the illegal instruction exception vector.

Use b command instead.

1.27 HRTmon/Usage/Commands/CMD_BD

CMD_BD

Description:

- Remove all breakpoints set by {"b " Link cmd_b} or {"bj " Link cmd_bj}.

Syntax:

bd

Notes:

- All breakpoints will be removed.

1.28 HRTmon/Usage/Commands/CMD_T

CMD_T

Description:

- Trace one or more instructions.

Syntax:

t ;trace 1 instruction

or t nbsteps ;trace nbsteps instructions

Parameters:

nbsteps = nb of instructions to trace without entering in the monitor.

1.29 HRTmon/Usage/Commands/CMD_TA

CMD_TA

Description:

- Trace till a specific address is reached.

Syntax:

ta address ;trace 1 instruction

Parameters:

address = when the processor reaches this address, it will activate the

monitor.

Notes:

- This function is usefull when you want to put a breakpoint in a ROM.

- Warning ! This function will slow-down your computer, as each instruction

will be traced until the address is reached.

- Be carfull when the processor switchs from User mode, to Supervisor !

1.30 HRTmon/Usage/Commands/CMD_DEBUG

CMD_DEBUG

Description:

- Intercept some exceptions.

Syntax:

debug ;switch on/off debug mode

Notes:

- When in debug mode. The monitor will be invoked when one of the following

exceptions is raised.

Bus Error

Address Error

Illegal Instruction

Divide by Zero

LINE-A

LINE-F

1.31 HRTmon/Usage/Commands/CMD_DRIVE

CMD_DRIVE

Description:

- View or set the actual drive number.

Syntax:

drive ;view actual drive no

or drive no ;change drive no

Parameters:

no = no of the drive to use for disk operations. (0-3)

Notes:

- All disk operations will take place on that drive.

1.32 HRTmon/Usage/Commands/CMD_MOTOR

CMD_MOTOR

Description:

- Switch the drive motor on.

Syntax:

motor

1.33 HRTmon/Usage/Commands/CMD_FORMAT

CMD_FORMAT

Description:

- Format a floppy disk in FFS.

Syntax:

format diskname

Parameters:

diskname = name for the disk. Default = HRTmon

Notes:

- The disk will be formatted in FFS.

1.34 HRTmon/Usage/Commands/CMD_QFORMAT

CMD_QFORMAT

Description:

- Quick format a floppy disk.

Syntax:

qformat diskname

Parameters:

diskname = name for the disk. Default = HRTmon

Notes:

- The disk will be quick formatted in FFS.

- This command won't work properly if the disk as never been formatted

in any Amiga format. (First use format)

1.35 HRTmon/Usage/Commands/CMD_DIR

CMD_DIR

Description:

- View the contents of the actual directory.

Syntax:

dir

or dir name

Parameters:

name = name of the directory to list.

Notes:

- use cd command to change the current directory.

1.36 HRTmon/Usage/Commands/CMD_CD

CMD_CD

Description:

- Change current directory.

Syntax:

cd name

Parameters:

name = name of future current directory.

Notes:

- To go to the parent directory type : cd /

1.37 HRTmon/Usage/Commands/CMD_L

CMD_L

Description:

- Load a binary file into memory.

Syntax:

l name address

Parameters:

name = name of the file to load from disk.

address = destination address for the file.

1.38 HRTmon/Usage/Commands/CMD_S

CMD_S

Description:

- Save a block of memory to disk.

Syntax:

s name start end

Parameters:

name = name of the file to save to disk.

start = start address of the block to save.

end = end address of the block to save.

1.39 HRTmon/Usage/Commands/CMD_MAKEDIR

CMD_MAKEDIR

Description:

- Create a new directory.

Syntax:

makedir name

Parameters:

name = name of the new directory to create.

1.40 HRTmon/Usage/Commands/CMD_RS

CMD_RS

Description:

- Read some sectors from disk.

Syntax:

rs address startsec nbsec

Parameters:

address = address where you want to load the sectors.

startsec = first sector you want to load.

nbsec = nb of sectors from startsec you want to load.

1.41 HRTmon/Usage/Commands/CMD_WS

CMD_WS

Description:

- Write some sectors to disk.

Syntax:

ws address startsec nbsec

Parameters:

address = address from where you want to save.

startsec = first sector of the disk to save.

nbsec = nb of sectors from startsec you want to save.

1.42 HRTmon/Usage/Commands/CMD_RA

CMD_RA

Description:

- Read all. (See wa)

Syntax:

RA

Notes:

- You will be asked to enter the 3 disks you used for wa.

- As all registers can't be read. This function won't work 100%.

- To get the best results do the following.

1st load the programm you want to restore.

2nd when you are in the programm use the RA command.

By this way all custom registers will be already initialized.

1.43 HRTmon/Usage/Commands/CMD_WA

CMD_WA

Description:

- Write all. (Save memory, registers, etc... to disks.)

It is usefull if you want to save a game which doesn't have a 'save game'

option.

Syntax:

WA

Notes:

- You will be asked to enter 3 disks.

- As all registers can't be read. This function won't work 100%.

- See ra command to see how to restore the programm.

- Warning ! Only the CHIP memory from $0 to $200000 will be saved.

So, if you want to use this command you will have to switch off your

FAST-RAM.

1.44 HRTmon/Usage/Commands/CMD_TSET

CMD_TSET

Description:

- Set the trainer buffer used for the ts and tf commands.

Syntax:

tset start end

Parameters:

start = start of the memory block you want to use for the trainer option.

end = end of the memory block.

Notes:

- 64K should be enough for most cases.

1.45 HRTmon/Usage/Commands/CMD_TS

CMD_TS

Description:

- Start the trainer process.

It will save all address containg a specific value (the actual number of

lives you have) in the trainer buffer.

Syntax:

ts start end val.x

Parameters:

start = start of the memory block you want to search.

end = end of the memory block.

val = the value you want to search. (number of lives)

x = b or w or l

Notes:

- After having used this command, continue your game and loose one life.

Then enter HRTmon and use the tf command.

1.46 HRTmon/Usage/Commands/CMD_TF

CMD_TF

Description:

- Contine the trainer process.

It will check if the address contained in the trainer buffer

have the value given in parameter.

If it's the case the address will be printed.

Only the printed address will remain in the trainer buffer, so you

can repeat the same procedure, to track step by step the exact address.

Syntax:

tf start end val

Parameters:

start = start of the memory block you want to search.

end = end of the memory block.

val = the value you want to search. (new number of lives)

Notes:

- If there are too many address, continue your game and loose one life.

Then enter HRTmon and use the tf command again.

1.47 HRTmon/Usage/Commands/CMD_MMUON

CMD_MMUON

Description:

- Switch the MMU on. (Only for full 68030)

Syntax:

mmuon

1.48 HRTmon/Usage/Commands/CMD_MMUOFF

CMD_MMUOFF

Description:

- Switch the MMU off. (Only for full 68030)

Syntax:

mmuoff

1.49 HRTmon/Usage/Commands/CMD_CUSTMMU

CMD_CUSTMMU

Description:

- Switch on/off a copy of the $dff000-$dff200 writes.

Syntax:

custmmu ;Switch function on/off

Notes:

- You need an Amiga 3000 with a 68030 with MMU to use this function.

(Will try to make it compatible with 68040 and any Amiga for a next release.)

- You can view the copy, at the address CUST.

For example type:

h cust+$80

to view the address of the copper-list.

- If the programm you are debugging is making an intensive use of the custom register you computer may slow-down. (Not very often...)

1.50 HRTmon/Usage/Commands/CMD_OUTPUT

CMD_OUTPUT

Description:

- Makes a copy of all outputs of HRTmon in memory.

Syntax:

output address ;start output

or output ;stop output

Parameters:

address = address where you want to redirect the output of HRTmon.

1.51 HRTmon/Usage/Commands/CMD_AF

CMD_AF

Description:

- assemble 65816 code in memory.

Syntax:

af address string

Parameters:

address = address where the code must be assembled.

string = code to assemble.

Parameters:

- I don't remember if this function was 100% finished.

As I never use it, I don't care... (Ask me if you really need it)

1.52 HRTmon/Usage/Commands/CMD_DF

CMD_DF

Description:

- disassemble 65816 code from memory.

Syntax:

df address

Parameters:

address = address to disassemble

Notes:

- Use the F6 key to toggle the 8/16 bit mode.

1.53 HRTmon/Usage/Commands/CMD_FIF

CMD_FIF

Description:

- Search for a string in the disassemble (65816) memory.

Syntax:

fif start end ’string’

Parameters:

start = start address for the search.

end = end address for the search.

string = string to find.

Notes:

- you can use the * character to replace any character.

1.54 HRTmon/Usage/Commands/CMD_COP

CMD_COP

Description:

- Find the actual copper-list.

Syntax:

cop start end

Parameters:

start = start the range to search.

end = end of the range to search.

Notes:

- This command can take some time if the copper-list is very long...

1.55 HRTmon/Usage/Commands/CMD_ESC

CMD_ESC

Description:

- View of modify the values written to some register at the exit from HRTmon.

(Some register can’t be read, and are modified by HRTmon, so you may

have to use this command.)

Syntax:

esc

or esc number value

Parameters:

number = no of the register to modify.

value = new value for the register.

1.56 HRTmon/Usage/Commands/CMD_SETMAP

CMD_SETMAP

Description:

- Modify the keyboard. (USA or CH)

Syntax:

setmap USA

or setmap CH

Notes:

- See also : F4

1.57 HRTmon/Usage/Commands/CMD_?

CMD_?

Description:

- Evaluate an expression.

Syntax:

? expression

Parameters:

expression = expression to evaluate.

Notes:

- use the '$' character to enter an hexadecimal value.

- use the '%' character to enter a binary value.

- use the '#' character to enter a decimal value.

- you can use any register in the expression.

Example:

? a0+d0*4

- use the '[' and ']' characters to enter an indirect address.

1.58 HRTmon/Usage/Commands/CMD_X

CMD_X

Description:

- Exit HRTmon

Syntax:

x

Notes:

- You can also press your left mouse button to exit HRTmon.

1.59 HRTmon/Usage/Commands/CMD_B

CMD_H

Description:

Syntax:

h address

Parameters:

1.60 HRTmon/Usage/Commands/

This function doesn't need any comments !

1.61 HRTmon/Usage/Remove

REMOVE

To remove HRTmon just run HRTmonPrefs.

HRTmon will be automatically located and removed.

1.62 HRTmon/Level7

LEVEL7 BUTTON

HRTmon needs a special hardware to work properly.

(If you don't have it you can still use HRTmon with the 'right-mouse enter'

option of HRTmonPrefs)

You will have to add a level7 button to your Amiga (if you don't have it

already). This button will also work with other software as ASMOne.

This button will generate a level7 (Non maskable interrupt) on you Amiga.

When you press this button, you will immediatly enter HRTmon.

Don't be afraid if you are not an electronics specialist ! It's very easy

if you know soldering.

(If you can't do it, just ask some friends. I'm sure you know someone

who will be able to do it.)

Material needed :

- 1 button.

- some wires.

- 3 diodes 1N4148 or any other equivalent.

Here is the schematics of this button:

IPL0 -------- >| ------\

|

IPL1 -------- >| -------------- button ------- GND

|

IPL2 -------- >| ------/

>| = diode.

IPL0 IPL1 IPL2 signals can be found on pin 83-82-81 of the expansion

port of the A1200.

1.63 HRTmon/Disclaimer and Author Info

HRTmon Information

The registered version is copyrighted, and is a personal license only. The demoversion is freely distributable as long as all of the files are included in their original form without additions, deletions or modifications of any kind, and only a nominal fee can be charged for the distribution. This software is provided "AS IS" without warranty of any kind, either expressed or implied. By using HRTmon, you agree to accept the entire risk as to the quality and performance of the program.

Please send your comments, wishes and bug reports for HRTmon to:

Alain Malek

Bonne-Esperance 1

1006 Lausanne

SWITZERLAND

Or by email:

amalek@di.epfl.ch

Please remember to state the version number you are using, in all correspondance.

1.64 HRTmon/How To Register

How To Register

To register, send me an errorfree double-density disk and USD $20 (SFR 25, DM 25, or an equal amount in any currency). Cash only, please. This will earn you the latest full version plus any number of free upgrades.

If you've got the possibility of receiving the package through email, send only money (no disk), and let me know how to contact you.

If you're a registered user and want to upgrade; send money to cover for a disk, packing and postage (one 2DD disk is approx. $1 USD, packing approx. $1 USD and postage is approx. $1.5 inside Europe and approx. $2 USD outside, which should sum up to ~$3.5 - $4 USD). This is not necessary if you've got an email adress where you can receive the upgrade, as they will be emailed to you free of charge upon request.

Notice that the demoversion is only an evaluation version. If you use it regularly, you should register.

Mail to:

Alain Malek

Bonne-Esperance 1

1006 Lausanne

SWITZERLAND

Email:

amalek@di.epfl.ch

Also contact me for questions, ideas for future releases, bugreports.

Please support Amiga shareware, don't spread the registered version, nor use a pirated version. Increased support means better and more software for the Amiga range of computers.

1.65 HRTmon/Thanks to...

Thanks to ...

- Nico Francois for his great reqtools library.

- Carnivore/BeerMacht for the assembler and disassembler functions.

First I did my own routines, but only for 68000, and I was too lazy to

add the 68020-68040,FPU,MMU instructions. So I took his routines from

BeerMon and adapted them to HRTmon.

- Roberto Marra for testing and giving me some ideas.

1.66 HRTmon/Future

The Future

- Better disk operation. (Add delete command.)

- Fix MMU functions for all machines. (I still have a 68EC030 so I won't

do this before I get a MMU ;)

- Add save-picture command.

- Add play-sound command.

- Add breakpoint option in the tracer.

- Any good idea from you.

1.67 HRTmon/History

History

version 1.01 10 April 1995

- Removed a bug when entering HRTmon. (Sometimes it crashed...)

- No more crash when exiting HRTmon, after having pressed the right-mouse

button in HRTmon (only in 'right-mouse enter' mode.)

- Now the debug command should work correctly.

version 1.00 2 April 1995

- First public release