

ZXAM

Antonio J. Rosselló

Copyright © 1993-95 WareSoft

COLLABORATORS

	TITLE : ZXAM		
ACTION	NAME	DATE	SIGNATURE
WRITTEN BY	Antonio J. Rosselló	December 2, 2024	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	ZXAM	1
1.1	ZXAM Guide	1
1.2	Distribution of ZXAM	1
1.3	Features of the emulator	3
1.4	Requirements for the emulator	4
1.5	The AGA Question	4
1.6	What's this thing?	5
1.7	Snapshot formats	5
1.8	How to run ZXAM	6
1.9	ToolType PATTERN	7
1.10	ToolType LOADPATH	7
1.11	ToolType SAVEPATH	7
1.12	ToolType DEFFORMAT	7
1.13	ToolType REXXPATH	8
1.14	ToolType ROMFILE	8
1.15	ToolType PRERUN	8
1.16	ToolType POSTRUN	8
1.17	ToolType LOADMODE	8
1.18	ToolType SAVEMODE	9
1.19	ToolType OUTPUT	9
1.20	ToolType AUDIO	9
1.21	ToolType AYEMULATION	9
1.22	ToolType SPEED	9
1.23	ToolType JOYSTICK	9
1.24	ToolType TASKPRI	9
1.25	ToolType DISPLAY	10
1.26	ToolType REFRESH	10
1.27	ToolType SCREENMODE	10
1.28	ToolType PUBSCREENNAME	10
1.29	The emulator's control window	10

1.30	Gadget Load Program	11
1.31	Gadget Reload	11
1.32	Gadget Save Program	11
1.33	Gadget Poke	11
1.34	The 'Save Format' Box	12
1.35	Gadget RUN	12
1.36	Gadget STOP	12
1.37	Gadget Run ARexx	12
1.38	Gadget Abort ARexx	12
1.39	String Address	12
1.40	String Data	12
1.41	Text Loaded File	13
1.42	Text Format	13
1.43	Emulation options	13
1.44	The chewing speccy keyboard	14
1.45	Supported ARexx commands	15
1.46	ZXAMGetReg	19
1.47	ZXAMSetReg	19
1.48	ZXAMPeek	19
1.49	ZXAMPoke	19
1.50	ZXAMDPeek	19
1.51	ZXAMDPoke	19
1.52	ZXAMGetMem	20
1.53	ZXAMPutMem	20
1.54	ZXAMFindByte	20
1.55	ZXAMRun	20
1.56	ZXAMQuit	20
1.57	ZXAMppLoadFile	20
1.58	ZXAMppSaveFile	21
1.59	ZXAMLoadRequester	21
1.60	ZXAMSaveRequester	21
1.61	ZXAMNameFormat	21
1.62	ZXAMActName	21
1.63	ZXAMActLoadPath	21
1.64	ZXAMActSavePath	22
1.65	ZXAMActFormat	22
1.66	ZXAMActSaveFormat	22
1.67	ZXAMProgVersion	22
1.68	ZXAMSaveFormat	22

1.69 ZXAMParseLoaded	22
1.70 ZXAMParseToSave	23
1.71 ZXAMJoinPathName	23
1.72 ZXAMGetScr	23
1.73 ZXAMClearNameFormat	23
1.74 ZXAMLoadPath	23
1.75 ZXAMSavePath	23
1.76 ZXAMFilePart	23
1.77 ZXAMPathPart	24
1.78 ZXAMPreRun	24
1.79 ZXAMActPreRun	24
1.80 ZXAMPostRun	24
1.81 ZXAMActPostRun	24
1.82 ZXAMPattern	24
1.83 ZXAMActPattern	24
1.84 ZXAMBASICToken	25
1.85 ZXAMReadAbort	25
1.86 ZXAMClearAbort	25
1.87 ZXAMNoReload	25
1.88 ZXAMEnableAbort	25
1.89 ZXAMDisableAbort	25
1.90 ZXAMDisassemble	26
1.91 ZXAMLoadMode	26
1.92 ZXAMSaveMode	26
1.93 ZXAMActLoadMode	26
1.94 ZXAMActSaveMode	27
1.95 ZXAMAbout	27
1.96 ZXAMFindBlock	27
1.97 ZXAMDoChecksum	27
1.98 ZXAMBreakPoint	27
1.99 ZXAMTrace	28
1.100ZXAMDoInt	28
1.101ZXAMBlockOfRegs	28
1.102Registers	29
1.103ZXAMAudio	29
1.104ZXAMAYEmulation	29
1.105ZXAMSpeed	30
1.106ZXAMJoystick	30
1.107ZXAMTaskPri	30

1.108ZXAMDisplay	30
1.109ZXAMRefresh	30
1.110ZXAMActAudio	30
1.111ZXAMActAYEmulation	31
1.112ZXAMActSpeed	31
1.113ZXAMActJoystick	31
1.114ZXAMActTaskPri	31
1.115ZXAMActDisplay	31
1.116ZXAMActRefresh	31
1.117ZXAMEmulToFront	32
1.118ZXAMControlToFront	32
1.119ZXAMActRun	32
1.120ZXAMAYBlockOfRegs	32
1.121ZXAMWriteROM	32
1.122ZXAMCheckSumString	32
1.123ZXAMCompareString	33
1.124ZXAMStop	33
1.125ZXAMTapeAction	33
1.126ZXAMTapeLoadName	33
1.127ZXAMTapeSaveName	33
1.128ZXAMTapeAddress	33
1.129ZXAMTapeSize	34
1.130ZXAMTapeFlag	34
1.131ZXAMTapeLoadOffset	34
1.132ZXAMTapeSaveOffset	34
1.133ZXAMTapeSetResult	34
1.134ZXAMTapeMark	35
1.135ZXAMTapeSetLoadName	35
1.136ZXAMTapeSetSaveName	35
1.137ZXAMTapeSetLoadOffset	35
1.138ZXAMTapeSetSaveOffset	35
1.139ZXAMTapeSetMark	35
1.140Tech info	36
1.141About this version....	36
1.142The worse part of any DOC....	38
1.143ZXAM versions	39
1.1441.0 ß	40
1.1451.1 ß	40
1.1461.2 ß	40

1.1471.3 B	41
1.1481.3b B	41
1.1491.4 B	42
1.1501.5	43
1.1511.6	44
1.1521.6b	45
1.1532.0	45
1.154Future of the emulator	49
1.155Thanks for all these people!	49
1.156Where I am?	50
1.157Message for registered users	51

Chapter 1

ZXAM

1.1 ZXAM Guide

ZXAM SPECTRUM EMULATOR v2.0
© 1993-95 WareSoft All rights reserved
by Antonio J. Pomar Rosselló

Distribution	Distribution and registration.
Features	What can ZXAM do ?
Introduction	About ZXAM...
Requirements	What is needed to run the emulator.
The AGA Question	But, why an AGA version ?
Spectrum program formats	Supported formats.
Running the emulator	Starting options.
Program's window	Use of the emulator.
Menus	Emulation options.
The Spectrum's keyboard	Those lovely chewing keys.
ARexx port	Complete flexibility.
Technical info	Curious thing.
Known bugs	Unwanted "features"...
About this version	Specific notes on this version.
History	ZXAM versions until now.
Future	What next ?
Acknowledgments	A lot of people.
Contact	How to reach the author.
Registered users	Important note for registered users.

1.2 Distribution of ZXAM

This program is SHAREWARE. This means that you can test the program during one month. After this period you must send \$ US 15 (only CASH) to the author. In return you will receive the latest version available (probably the 128k version will be available by march 1995). By doing this contribution, you assure the further development of the emulator. Thanks in advance.

This SHAREWARE version can be freely distributed provided that all the associated files are distributed with the program itself. Neither the program nor their associated files must be modified in any way. The complete distribution package can be compressed with LHA, LHarc or similar for distribution convenience. No charge must be made for use or distribution of this package. You can upload this package to any BBS, include it in a magazine's coverdisk, in a Public Domain library or in recopilation CDs of PD/Freeware/Shareware (like the excellent Aminet ones).

If you want to get the latest version available you can do a FileRequest (FidoNet) to Tanit BBS-Ibiza (+34-71-392829, USRobotics 28.800 baud) with the Magic Name 'ZXAM'. You can get it through terminal too (if you know spanish, of course).

The complete distribution package contains:

ZXAM_Spectrum	Generic version of ZXAM (with icon).
ZXAM_Spectrum_FASTER	Faster generic version (with icon).
ZXAM_SpectrumAGA	Special AGA version (with icon).
ZXAM_SpectrumAGA_FASTER	Faster AGA version (with icon).
ZXAM_REXX/	Drawer with several ARexx scripts.
ZXAM_Español.guide	Spanish GUIDE (with icon).
ZXAM_English.guide	The text that you are reading (with icon).
LEEME!!	Distribution notes, spanish (with icon).
README!!	Distribution notes, english (with icon).
INTERFACE.IFF	Diagram of the tape interface (with icon).
DIGIT_INTERFACE.IFF	Digitalization of my interface (with icon).
powerpacker.library	To load compressed spectrum programs.

The executables are compressed for distribution convenience of this package inside floppy disks. The whole package fits nearly perfect in a double density FastFileSystem disk. The executables uncompressed are 450Kb each (because the system can't join several DATA and BSS hunks in memory in a given order when loading), so the compression has been necessary.

This software is provided 'AS IS' without warranty of any kind, either expressed or implied. By using this package you agree to accept the entire risk as to the quality and performance of the program, or even of the DOC. The same applies to the tape interface. It is prohibited to build and sell the tape interface for profit. The interface must be build for personal use only.

All trademarks and Copyrights of products mentioned in this document are acknowledged by the author.

In the DOC of the Spectrum Emulator 1.7 I have read that AMSTRAD has given special permission for emulator authors to use the Spectrum's ROM code. Anyway the Spectrum's ROM is copyright to AMSTRAD.

The iff screen 'KeysHelp.iff' has been included with permission of the author (Mikael Ostergren). Thanks Mikael :-)

The nice MagicWB icons of the emulator where done by Richard Harris. Thanks Richard :-)

The Newicon icons of the emulator has been done by myself, so if you think they are awful, you know who must be blamed for it ;-)

The powerpacker.library is © Nico François.

The Sinclair ZX Spectrum Emulator is © Peter McGavin.

The Sinclair ZX Spectrum Emulator "Z80" is © Gerton Lunter.

The emulator ZX Spectrum (SPEC386) is © Pedro Gimeno.

1.3 Features of the emulator

- Emulation of a Sinclair ZX Spectrum 48k.
 - Stabilized 48k sound.
 - Load/save to tape, or redirected to disk or to ARexx scripts for use of complex tape formats, like .TAP.
 - ARexx port with 93 functions.
 - Fully multitasking.
 - Automatic speed adjustment.
 - Optimised CPU usage. The CPU time not necessary for the required speed is given to the others tasks in multitasking (a 1200/030/28MHz can run two copies of Manic Miner in multitasking at full speed, and 10% of CPU time is still unused).
 - Full emulation of the Spectrum 128k sound (AY-3-8912), with white noise, volume waves and 3 channels of 4 bit samles.
 - The stereo follows the ABC and ACB distributions used by the AY interfaces for Spectrum 48k (the left side contains the 48k sound and the A channel, and the right side contains the B and C channels).
 - Dynamic assignation of the noise channel for 48k or 128k sound.
 - The Spectrum's screen can be put inside a window over any public screen. This is possible even with RTG emulations, like the Retina one. The emulator tries to adapt to the colors available in the screen.
 - System friendly screens can be used for the Spectrum display, even when disabling multitasking.
 - Emulation of Kempston, Sinclair (Interface II) and Cursor (AGF/Protek) joysticks.
 - Includes a transfer to save to tape complete games.
 - Real border, but not synchronized with VBlank.
 - Flash emulated (only special AGA version).
 - FASTER version for more speed, but with a less accurate emulation.
 - The emulator adapts to the available hardware. If no sound channels or CIAs are available the emulator will disable automatically the options that uses that hardware.
 - The hardware needed for the emulation is allocated only when the emulation is running.
 - The windows of the emulator (both the control and display window) are appwindows, so you can drop snapshotsd inside to load.
 - Custom tape interface, much cheaper than a sampler.
 - You can run several copies of the emulator in multitasking. The only limit for the number of copies running is the available memory and the CPU power
-

- of the machine (or the patience of the user ;-)
- Step-by-step execution of Z80 code, Z80 disassembling and breakpoints.
- Full emulation of the R register.
- Full emulation of BCD.
- Emulated all the undocumented instructions of Z80.
- AGA version can work in B&W or Color mode.

1.4 Requirements for the emulator

The Generic version of ZXAM needs an Amiga computer with 68020 CPU (or higher) and version 2.04 of the operating system (or higher).

The special AGA version needs an Amiga computer with 68020 CPU (or higher), version 3.0 of the operating system (or higher) and AGA chipset. In short, an Amiga 1200 or 4000.

Is VERY convenient that you have 32 bits FAST memory. Without FAST memory the emulation will run really slow.

All the versions need the powerpacker.library, to load the compressed Spectrum programs, in LIBS: or in the same directory of the executable.

For the filerequester the asl.library is used, but this library has a problem. Each time the filerequester is opened, the whole directory has to be read. This can be very annoying when loading files from floppy disks, or when you load it from a harddisk directory with a lot of files (400 files, in my case). The best solution could be to use the reqtools.library filerequester, as this requester has a buffer that avoid the re-reading of the directory, but due to serious technical problems I couldn't do it yet. Meanwhile the best option is to use the RTPatch program (by Nico François), that patches the asl/req/arp libraries to use the reqtools filerequester instead. This way the emulator, and other programs that use asl/req/arp, will use the reqtools filerequester, that is much better than the original ones. Curiously with RTPatch 1.3 (the one that I use) the buffer works well, but with ReqChange 3.0 (another patch that does the same) the buffer doesn't work.

Another program worth installing is the system commodity 'MouseBlanker'. In multitasking modes can be annoying to have the mouse pointer in the screen while using the emulator. With that commodity the mouse pointer will disappear when pressing any key, and reappear when moving the mouse.

1.5 The AGA Question

But, what's AGA in the special version? Well, I use some special AGA capabilities, not present in the OCS/ECS chipset, to convert the Spectrum's pixels and attributes to Amiga bitplanes. These capabilities accelerate the access to the screen of the emulation, speeding up programs with massive graphic movement, like Uridium and demos. In most cases the difference isn't noticeable, so it's more recommendable to use the Generic version in most cases, and use the AGA version when you need the maximum graphic speed available.

The AGA version uses a graphic system that can't be used with multitasking enabled, and that be shown in PAL mode. This is the reason why the AGA version lacks all the multitasking capabilities of the Generic version. If you need multitasking, use any other monitor, like VGA, or run the emulator over a RTG card, you will need to use the Generic version. Another thing is that only the AGA version has a complete FALSH emulation. The generic version probably will include FLASH emulation in next versions, but right now the FLASH is emulated as INVERSE.

1.6 What's this thing?

Well, maybe you have guessed that this program is a Sinclair ZX Spectrum 48k emulator, with those lovely attributes and only one channel of 1 bit sound (sob...). Since version 2.0, ZXAM includes the emulation of the AY-3-8912 sound chip (the one used in 128k versions of Spectrum), but the emulation is still 48k.

1.7 Snapshot formats

Before you can understand the function of some of the gadgets of the program, you must learn something about the format of the Spectrum programs (usually called 'snapshots').

On the disk, a Spectrum program is simply a file that contains a copy of the 48k of RAM of the Spectrum, along with all the registers of the Z80 CPU (the CPU of the Spectrum).

The problem is that every Spectrum emulator (on any computer) uses a different format to save the memory and the registers. That means that a program saved with an emulator will not load into any other emulator.

Instead of create my own format for the Spectrum program I have decided to support several formats of snapshot. You can decide what format you want to use for the programs, or use the emulator as a format conversor too.

The four formats that I know, and that I have included in the emulator, are called PC, MIRAGE, Z80 and KGB. These formats are:

PC: This is the format used by a nice emulator for the IBM PC, and compatibles, programmed in Spain by Pedro Gimeno. The executable is called SPEC386.EXE and the version I know is the 0.99D B. This emulator needs a 386DX at 25MHz, with a 256k cache, to approach to the original Spectrum speed. These files are 49190 bytes long and have a .SP extension appended to the name.

MIRAGE: This is the format used by the Spectrum Emulator 1.7 by Peter McGavin. These files are 49179 bytes long and have the .snapshot extension appended to the name. I recommend that you change the extension to .mirage, because there are many Sepctrum emulators that use the .snapshot extension and this can cause confusion. The format version supported by the ZXAM emulator is the post-1.6 version, that stores correctly the border colour. This is the same

format used by the emulators (for PC) VGASPEC (*.SNA) and JPP (*.SP, by Arnt Gulbrandsen).

Z80: This format (implemented since version 1.6) is the one used by the Spectrum emulator (for PC computers) called 'Z80' by G.A.Lunter. ZXAM can load three Z80 formats (versions 1.45, 2.01 and 3.0) if the snapshot is a 48k one. If the snapshot is 128k or needs the emulation of Interface I or any other hardware not supported by ZXAM, the snapshot will not be loaded. This format is implemented for loading only.

KGB: This format (implemented since version 1.6b) is the one used by the KGB Spectrum emulator for Amiga computers. These files are always 49486 bytes long. I don't know who is the author as I have never seen any DOC file for this emulator. Probably is in disuse due to its incompatibility with version 2.0 and higher of AmigaDOS. This format is available for loading only.

In all the cases the snapshots can be crunched with powerpacker.

Notice that the emulator NEVER uses the filename extension for identify the format of the file. Instead, a series of checkings in the file structure are done for identifying it. You can use the extension you wish for the files, but if you don't include this extension in the pattern of filerequester (tooltype PATTERN in the emulator's icon) the files will not appear in it.

There is another format, the .TAP one. This is not a snapshot format. Instead, it is a "tape" file that contents a series of headers and data blocks. There is an ARexx script with the emulator to "explode" these .TAP files into the corresponding .header and .bytes files (a la Spectrum 1.7). This way you can load these files through the "Load mode -> Disk" option, or from Peter McGavin's Spectrum Emulator. Since ZXAM v2.0 there is an ARexx script to load automatically the .TAP files without having to explode them (just select 'Load mode -> ARexx'). That script can even create .TAP files from the 'Save mode -> ARexx' option.

If you know any other format (for any computer, like Amiga, Cray, MAC, Atari or PC) please send to me the information that you have, along with a disk with several games saved in that format. In the next version of the emulator I will support that format.

1.8 How to run ZXAM

You can run the emulator from WorkBench or SHELL. If you run it from the SHELL, it will "detach" from the SHELL window and will search the program's icon to read the ToolTypes. Moreover, it will load the snapshot file specified, if so, in the command line.

Any problem at startup will be reported by a requester.

You can put ZXAM as a default tool for the icon of a Spectrum snapshot and run the emulator from there. The snapshot will be loaded automatically.

Inside the emulator's icon you can put many ToolTypes to control the emulation options and the initial state of the menus. Available ToolTypes are:

LOADPATH
SAVEPATH
PATTERN
REXXPATH
ROMFILE
PRERUN
POSTRUN
OUTPUT
DEFFORMAT

Initial values for the 'Options' menu:

LOADMODE
SAVEMODE
AUDIO
AYEMULATION
SPEED
JOYSTICK
PUBSCREENNAME

ToolTypes NOT available in the special AGA version:

TASKPRI
DISPLAY
REFRESH
SCREENMODE

1.9 ToolType PATTERN

PATTERN= Here you can put the pattern (or 'filter') for the
 filerequester. Take a look at the user's manual of
 your Amiga to learn how to use these patterns.

1.10 ToolType LOADPATH

LOADPATH= The directory where the LOAD filerequester will be
 opened. This can be the directory where you have
 the Spectrum games in the HD.

1.11 ToolType SAVEPATH

SAVEPATH= The directory where the SAVE filerequester will be
 opened.

1.12 ToolType DEFFORMAT

DEFFORMAT= The default format used to save the programs (PC, PC_PP, MIRAGE or MIRAGE_PP). Anyway, you can change this with the suitable gadgets.

1.13 ToolType REXXPATH

REXXPATH= The directory where you have the ARexx scripts for the emulator. The script names must end with the '.zxm' extension.

1.14 ToolType ROMFILE

ROMFILE= The external ROM to load over the internal one. The ROM can be compressed with powerpacker. You must be careful about non-original ROMs, as they can produce incompatibility with certain games (not an emulator's fault ;-)

1.15 ToolType PRERUN

PRERUN= Sets a command line to be executed by AmigaDOS right before starting the emulation. For example, if "PRERUN=cpu nocache" the caches of CPU will be disabled. Before the PRERUN command line is executed the emulator will save the status of the caches for restore it when returning from emulation (with HELP key). This way you can have the caches disabled in the emulation but without having to disable the caches for the WorkBench too. In multitasking this command line is not executed.

1.16 ToolType POSTRUN

POSTRUN= This command line will be executed right after returning from the emulation. The status of caches is restored BEFORE the POSTRUN is executed. In multitasking this command line is not executed.

1.17 ToolType LOADMODE

LOADMODE= Sets the default mode for the ROM's LOAD routine (TAPE, DISK or AREXX).

1.18 ToolType SAVEMODE

SAVEMODE= Sets the default mode for the ROM's SAVE routine (TAPE, DISK, DISK_PP if you want that the data is crunched before SAVEing to the disk, and AREXX).

1.19 ToolType OUTPUT

OUTPUT= Specify an output file (usually a CON: window) as output for the PRERUN and POSTRUN commandlines, and ARExx scripts. This is a very useful option for debugging new ARExx scripts. If not specified, the default output is NIL:.

1.20 ToolType AUDIO

AUDIO= Sets the initial state of the 'Options -> Audio' menu. Possible values are OFF, FAST and CIASync.

1.21 ToolType AYEMULATION

AYEMULATION= Sets the initial state of the 'Options -> AY-3-8912' menu. Possible values are OFF, ON and SMART.

1.22 ToolType SPEED

SPEED= Sets the initial state of the 'Options -> Speed' menu. Possible values are FAST and AUTO.

1.23 ToolType JOYSTICK

JOYSTICK= Sets the initial state of the 'Options -> Joystick' menu. Possible values are KEMPSTON, SINCLAIRRIGHT, SINCLAIRLEFT and CURSOR.

1.24 ToolType TASKPRI

TASKPRI= Sets the initial state of the 'Options -> TaskPri' menu. Possible values are +1, 0 and -1.

1.25 ToolType DISPLAY

DISPLAY= Sets the initial state of the 'Options -> Display' menu. Possible values are SCREEN, WINDOW, NOSYSTEMSCREEN and NOSYSTEMPAL.

1.26 ToolType REFRESH

REFRESH= Sets the initial state of the 'Options -> Refresh' menu. Possible values are 1, 2, 4 and 8.

1.27 ToolType SCREENMODE

SCREENMODE= Here you can specify the screen mode to use when opening a 'legal' screen for the emulator (SCREEN and NOSYSTEMSCREEN modes). All available modes in the Screenmode preferences of the system can be used, IF the desired mode can support 16 colors. The name put here is exactly the same that appear in the 'sys:Prefs/screenmode' preferences.

If no screenmode is specified, the emulator will use a PAL screen (in 2.04 and 2.1 systems), or get the monitor used by the screen where the control window is open (in 3.0+ systems) using the BestModeID function. The emulator can be promoted externally (with NewMode or ModePro) only if the resulting screen is 256x192 pixels at 16 colors. The screen is opened with the name 'ZXAM Screen'.

Due to the way the emulator draws on screen, that screen can't be promoted to a RTG card. If you want to use the emulator over a RTG card you must use the 'Display -> Window' mode.

1.28 ToolType PUBSCREENNAME

PUBSCREENNAME= Name of the public screen where the control and emulation windows must be opened. If not specified, or the screen isn't found, the windows will be opened on the default public screen.

1.29 The emulator's control window

If you run the emulator you will see the 'control window', from which you can operate the emulator and control the options available. The gadgets that has a letter underlined (all the gadgets in the window) has a shortcut that is the corresponding key of that letter. The shortcut for the close window gadget is the Esc key, and the Help key shows the About requester. The TAB key can be

1.34 The 'Save Format' Box

"SAVE FORMAT" box .- Inside this box there are 4 gadgets to select the format you want to use to save the Spectrum programs. The formats are PC and MIRAGE, and the compressed forms PC_PP and MIRAGE_PP.

1.35 Gadget RUN

RUN .- Starts the emulation in the mode set in the 'Options' menu.

1.36 Gadget STOP

This gadget is named 'STOP' in the Generic version of ZXAM, and 'Reset & RUN' in the special AGA version, having a different function in each case:

Stop.- Stops the emulation, as if HELP had been pressed.

Reset & RUN.- Runs the emulator, but doing a RESET of the Spectrum.

1.37 Gadget Run ARexx

Run ARexx.- Opens a file requester to select the ARexx script to run. It will be opened on the path fixed by the REXXPATH ToolType. Only the files with the '.zxm' extension will be showed.

1.38 Gadget Abort ARexx

Abort ARexx.- This gadget is under control of the ARexx script. Is the script that must enable it and read it when needed. The emulator simply says to the script if the gadget has been pressed.

1.39 String Address

Poke address.- Here you must put the address where you want to do the POKE.

1.40 String Data

Poke data.- Here you must put the byte for the POKE.

1.41 Text Loaded File

Here is shown the name of the snapshot actually loaded.

1.42 Text Format

Here is shown the format (on disk) of the actually loaded snapshot.

1.43 Emulation options

The menu strip includes two pull down menus. The 'Project' menu just have the 'Load program', 'Save program', 'Run', 'About...' and 'Quit' items. The 'Options' menu is used to control the different emulation options. Here you have the 'Options' menu with their sub-menus:

Load mode	Controls the LOAD mode of the ROM patch.
Disk	Load from disk the .header and .bytes files (like Spectrum 1.7).
Tape	Load from tape through the ROM patch (needs a CIA).
ARexx	Run the ZXAM_TAPE.zxam ARexx script (see the ARexx section).

(the 'Tape' mode is available only in NON multitasking modes)

Save mode	Controls the SAVE mode.
Disk	Save to disk the .header and .bytes files (like Spectrum 1.7).
Disk Crunched	Save to disk, but compressing with powerpacker.library.
Tape	Save to tape (needs 'Audio -> CIA Sync').
ARexx	Run the ZXAM_TAPE.zxam ARexx script (see the ARexx section).

(the 'Tape' mode is available only in NON multitasking modes)

Audio	Controls the 48k sound.
Off	Disables the 48k sound.
Fast	Sound at maximum speed.
CIA Sync	Sound stabilized with a CIA.

(the 'CIA Sync' must be enabled in order to work right the Transfer and Save to tape)

AY-3-8912	Controls the 128k sound.
Off	Disables the 128k sound.
On	Enables the 128k sound.
Smart	'Smart' mode that rearrange the channels in order to do a correct mixing of tone and white noise (useful only when the Amiga audio is listened through a stereo system). Can be disabled in prevision of problems.

Speed	Estabilizador de velocidad.
Fast	Maximum possible speed (uses 100% of available CPU).
Auto	Automatic speed, using only the necessary CPU.

(In multitasking the timer.device is used to control the speed, but in NON multitasking modes a CIA must be allocated)

(The recommended mode for ALL computers is 'Auto', as it only slows down the emulation when the computer is too fast, but never will slow down the emulation when the computer is too slow. This is the reason why this mode is called 'Automatic')

Joystick Joystick emulated over the Amiga's one.
 Kempston Kempston type (used in most of games).
 Sinclair Right Sinclair type (Interface II) right side (keys 6 to 0).
 Sinclair Left Sinclair type (Interface II) left side (keys 1 to 5).
 Cursor Cursor type (also known as AGF and PROTEK).

(This option doesn't disable the Kempston joystick. It adds another emulation to the kempston. That means that in Sinclair mode the joystick can be read as Sinclair or as Kempston simultaneously)

(Multitasking related options, not available in the special AGA version)

TaskPri Priority of the emulation task.
 +1
 0
 -1

Display Display mode of the emulation.
 Screen Own screen in multitasking.
 Window Window over a public screen in multitasking.
 No System (Screen) Own (system friendly) screen, NON multitasking.
 No System (PAL) Custom copperlist, NON multitasking. This is the mode used by previous versions.

(see the Known bugs section for problems of the 'No System (Screen)' mode)

Refresh Time between redraws of the emulation window.
 1 Frame (only for really powerful machines)
 2 Frames (smooth movements, but power consuming)
 4 Frames (the best for general use and medium power machines)
 8 Frames (slow machines or static images, like the BASIC screen)

NOTE: The modes that need a CIA to work are 'Load Mode -> Tape', 'Audio -> CIA Sync' and 'Speed -> Auto' (the last only in NON multitasking). These modes doesn't need a CIA for each one. They share the same CIA, and thus a copy of ZXAM running will allocate only one CIA, or none if not available or these settings are not used.

1.44 The chewing speccy keyboard

In the emulation the keyboard is like the one of the Spectrum, with some 'special' keys:

<- Deletes a char (Shift+0)
 Ctrl Like pressing Caps+Symbol (E mode)
 Alt Like Symbol Shift
 Del Like Shift+1 (Edit)
 Tab Like Shift+9 (Graf)
 Help Returns to WorkBench

Esc RESET!!!

F1 Switches between colour/black&white modes. In CHIPMEM-only systems the DMA load of the memory slows down the CPU. If you switch to the black&white mode the DMA load is much smaller, thus the emulator will run about 10% faster and the sound will be A LOT better. (only special AGA version).

F2 In black&white mode inverts the image (only special AGA version).

F6 Starts the transfer (the Spectrum's memory will be saved to tape ready to be loaded in a real Spectrum or in the emulator itself). This only works in NON multitasking modes.

Cursor keys Like the original cursor keys of Spectrum+ (Shift+5 6 7 8).

, (key on the right of M) is the ',' (Symbol+N)

. (2^a on the right of M) is the '.' (Symbol+M)

' (3^a on the right of M) is the '"' (Symbol+P)

Ñ (key on the right of L) is the ':' (Symbol+Z)

; (2^a key on the right of L) is the ';' (Symbol+O)

(Symbols referent to the spanish Amiga 1200 keyboard)

The keypad works complete.

The joystick is emulated as selected in the 'Options -> Joystick' menu.

Maybe you will notice that you can't press simultaneously 2 keys in the same row of the keyboard. This is because you have an Amiga 1200, and its keyboard simply behaves in this very particular way. You can test it with any of those keyboard-playing Spectrum games (all the games!).

In multitasking modes the keys/joystick are read only when the emulation window/screen is selected with the mouse.

1.45 Supported ARexx commands

Since version 1.4 ß the emulator has an ARexx port. This port is for writing scripts to support new file formats, assemble, disassemble, and any other thing you can think. The port has the name 'ZXAM_REXX', but you only need the name to test if the emulator is present. More on this below.

With the emulator you have some example scripts (example but useful, I think). There is an example to load snapshots from ARexx, another to save the Spectrum's screen as IFF, for batch converting of formats, for listing the BASIC program inside the Spectrum memory, for extracting the ROM from the emulator, if you want to modify it for any reason, for disassemble the spectrum's memory and for discompose the .TAP files. Some of these scripts are simply examples of usage of the emulator's functions. A good sample of the possibilities of the ARexx port is the Monitor.zxam script (created by Leonardo Cocaña Galán). This script acts like a very simple disassembler, but can be greatly enhanced to be a full blown disassembler/monitor. Some of the scripts need some system commands (like requestchoice or multiview). If you doesn't have these commands you must modify the scripts to use the commands that you have.

The ARexx port of the emulator is an unusual one. Intead of being a Command Host it is a Function Host. That means that the parameters are fomatted inside parentheses, separated by commas. Even if the function doesn't need parameters

you must put an empty parentheses at the end. The advantages are that you can use the functions directly inside expressions, and you don't need to do ADDRESS ZXAM_REXX or OPTIONS RESULTS at the beginning of the script. All the functions begin with ZXAM to avoid interference with function's names of other Function Hosts and libraries. The names are case insensitive.

IMPORTANT NOTE: Now that the emulator can run in multitasking, many functions (like register access or load/save snapshots) can be executed only with the emulation stopped (if not the returned results could be completely wrong). These functions will return an error if you try to run them with the emulation running. With the ZXAMActRun() and ZXAMStop() functions you can read the actual state of the emulation, and stop it if you want. With the ZXAMRun() you can resume the execution of the emulator.

In the listing the functions marked with (*) NEED the emulation to be stopped in order to work, and the ones marked with (+) are NOT available in the AGA version.

The 93 functions implemented in version 2.0 (37.24 and 39.24) are:

Available since version 1.4 B (39.14):

- * ZXAMGetReg
- * ZXAMSetReg
- ZXAMPeek
- * ZXAMPoke
- ZXAMDPeek
- * ZXAMDPOke
- ZXAMGetMem
- * ZXAMPutMem
- ZXAMFindByte
- ZXAMppLoadFile
- ZXAMppSaveFile
- ZXAMRun
- ZXAMQuit
- ZXAMNameFormat
- ZXAMClearNameFormat
- ZXAMLoadRequester
- ZXAMSaveRequester
- ZXAMActFormat
- ZXAMActName
- ZXAMActLoadPath
- ZXAMActSavePath
- ZXAMSaveFormat

Available since version 1.5 (39.17):

- ZXAMActSaveFormat
- ZXAMProgVersion
- * ZXAMParseLoaded
- * ZXAMParseToSave
- ZXAMJoinPathName
- ZXAMGetScr
- ZXAMLoadPath
- ZXAMSavePath
- ZXAMPathPart
- ZXAMFilePart

ZXAMBASICToken
ZXAMActPreRun
ZXAMActPostRun
ZXAMPreRun
ZXAMPostRun
ZXAMActPattern
ZXAMPattern
ZXAMReadAbort
ZXAMClearAbort
ZXAMNoReload
ZXAMEnableAbort
ZXAMDisableAbort

Available since version 1.6 (39.20):

ZXAMDisassemble

Available since version 1.6b (39.21):

ZXAMLoadMode
ZXAMSaveMode
ZXAMActLoadMode
ZXAMActSaveMode
ZXAMAbout
ZXAMFindBlock
ZXAMDoChecksum
* ZXAMBreakPoint
* ZXAMTrace
* ZXAMDoInt
* ZXAMBlockOfRegs

Available since version 2.0 (37.24 y 39.24):

ZXAMAudio
ZXAMAYEmulation
ZXAMSpeed
ZXAMJoystick
+ ZXAMTaskPri
+ ZXAMDisplay
+ ZXAMRefresh
ZXAMActAudio
ZXAMActAYEmulation
ZXAMActSpeed
ZXAMActJoystick
+ ZXAMActTaskPri
+ ZXAMActDisplay
+ ZXAMActRefresh
ZXAMEmulToFront
ZXAMControlToFront
ZXAMActRun
* ZXAMAYBlockOfRegs
* ZXAMWriteROM
ZXAMChecksumString
ZXAMCompareString
ZXAMStop

Special scripts:

`ZXAM_INIT.zxam` This script is executed when starting ZXAM, right after the 'About...' requester.

`ZXAM_EXIT.zxam` This script is executed when quitting ZXAM.

`ZXAM_TAPE.zxam` This script is executed while in 'Load Mode -> ARexx' and 'Save Mode -> ARexx'. For example, when 'Load Mode -> ARexx' is selected and from Spectrum's BASIC you write 'LOAD ""' this script is executed to give to the emulator the desired blocks of data to load. The example script enables to load blocks from .TAP files without having to 'explode' them in .header and .bytes files.

The execution environment of the script is very specific. When starting, the script must use the `ZXAMTapeAction()` function to know the operation desired (load, save or verify). The operation is expressed with a number that can be: 0=nothing, 1=load, 2=verify and 3=save. The script must decide what to do in each of the cases. If the operation is 0 that means that the script hasn't been executed in automatic mode, but in manual mode (from CLI or from the 'Run ARexx' gadget). The script must say to the user that it isn't a direct execution script and exit. For this script there are several specific functions, that are:

```
ZXAMTapeAction
ZXAMTapeLoadName
ZXAMTapeSaveName
ZXAMTapeAddress
ZXAMTapeSize
ZXAMTapeFlag
ZXAMTapeLoadOffset
ZXAMTapeSaveOffset
ZXAMTapeSetResult
ZXAMTapeMark
ZXAMTapeSetLoadName
ZXAMTapeSetSaveName
ZXAMTapeSetLoadOffset
ZXAMTapeSetSaveOffset
ZXAMTapeSetMark
```

When exiting, the script must use the `ZXAMTapeSetResult` function to give to the emulator the result of the operation: 2=error, 1=ok and 0=none. The 0 result is for scripts that doesn't support the desired action and want to pass the control to the internal .header and .bytes loader/saver (as if 'Load/Save Mode -> Disk' where selected). When result is 1, and operation is LOAD or VERIFY, a second result must be given to the emulator. This result is the path+name of the file to be loaded/verified (in .header/.bytes format). This is usually a temporal file that contains the extracted portion of the .TAP file to load.

All this seems a little confusing, but if you take a look at the included `ZXAM_TAPE.zxam` script you will understand the mechanics of this script. This script is actually prepared to LOAD files from .TAP, and to SAVE (create) new .TAP files. With a little patience you can, for example, modify/enhance the script to read .header/.bytes files from inside a .LHA archive, as if it were a compressed .TAP file.

1.46 ZXAMGetReg

ZXAMGetReg: Gets the contents of a register, 8 or 16 bits. The optional parameter 'format' sets the format of the value returned ('h' for hexadecimal and 'd' for decimal). The default format is decimal.

```
format:          value = ZXAMGetReg( register , [format] )
```

1.47 ZXAMSetReg

ZXAMSetReg: Puts an 8 or 16 bits value inside a Z80 register.

```
format:          ZXAMSetReg( register , value )
```

1.48 ZXAMPeek

ZXAMPeek: Reads a byte from the specified address.

```
format:          value = ZXAMPeek( address )
```

1.49 ZXAMPoke

ZXAMPoke: Writes a byte at the specified address.

```
format:          ZXAMPoke( address , byte )
```

1.50 ZXAMDPeek

ZXAMDPeek: Reads a 16 bit value from the Spectrum's memory. The value is read with the z80 format, that is, less significant byte first.

```
format:          value = ZXAMDPeek( address )
```

1.51 ZXAMDPeke

ZXAMDpoke: Writes a 16 bit value from the Spectrum's memory. The value is written with the z80 format, that is, less significant byte first.

```
format:          ZXAMDPeke( address , value )
```

1.52 ZXAMGetMem

ZXAMPutMem: Puts a string in the memory. Usually the string is a block of bytes, or the entire spectrum's memory.

format: ZXAMPutMem(address , string)

1.53 ZXAMPutMem

ZXAMGetMem: Copies the contents of the specified block to the result string.

format: string = ZXAMGetMem(address , size)

1.54 ZXAMFindByte

ZXAMFindByte: Searches the specified byte, starting from the address. If the top of the spectrum's memory is reached without finding the byte, a -1 will be returned. If the byte is found, the address of the byte is the result.

format: address = ZXAMFindByte(address , byte_to_search)

1.55 ZXAMRun

ZXAMRun: Starts the emulation.

format: ZXAMRun()

1.56 ZXAMQuit

ZXAMQuit: Quits the emulator.

format: ZXAMQuit()

1.57 ZXAMppLoadFile

ZXAMppLoadFile: Loads a file through the powerpacker.library and puts the complete file in the result string. If the file wasn't compressed, it will be loaded anyway.

format: string = ZXAMppLoadFile(path_&_name)

1.58 ZXAMppSaveFile

ZXAMPPSaveFile: Saves a block of bytes to a file, compressing it with `powerpacker.library`. If you don't want that the file be compressed, use the `ARexx` own file commands.

format: `ZXAMppSaveFile(path_&_name , string)`

1.59 ZXAMLoadRequester

ZXAMLoadRequester: Opens the `LOAD` `filerequester`. The title is for the top bar of the requester. The path for open the requester is optional. If no path is specified, the last path used with the requester will be used. If `'path_&_name'` is empty means that the `'cancel'` has been pressed.

format: `path_&_name = ZXAMLoadRequester(title , [path])`

1.60 ZXAMSaveRequester

ZXAMSaveRequester: Opens the `SAVE` `filerequester`. The title is for the top bar of the requester. The path for open the requester is optional. If no path is specified, the last path used with the requester will be used. If `'path_&_name'` is empty means that the `'cancel'` has been pressed.

format: `path_&_name = ZXAMSaveRequester(title , [path])`

1.61 ZXAMNameFormat

ZXAMNameFormat: Puts the strings `'name'` and `'format'` in the `'Loaded file'` and `'Format'` boxes of the emulator's window.

format: `ZXAMNameFormat(name , format)`

1.62 ZXAMActName

ZXAMActName: Returns the contents of the `'Loaded file'` box. If there is no name an empty string will be returned.

format: `name = ZXAMActName()`

1.63 ZXAMActLoadPath

ZXAMActLoadPath: Returns the last path used in the `LOAD` `filerequester`.

format: `path = ZXAMActLoadPath()`

1.64 ZXAMActSavePath

ZXAMActSavePath: Returns the last path used in the SAVE filerequester.

```
format:          path = ZXAMActSavePath()
```

1.65 ZXAMActFormat

ZXAMActFormat: Returns the contents of the 'Format' box. If there is no format an empty string will be returned.

```
format:          format = ZXAMActFormat()
```

1.66 ZXAMActSaveFormat

ZXAMActSaveFormat: Returns the format actually selected for SAVE. The result can be PC, MIRAGE, PC_PP and MIRAGE_PP.

```
format:          format = ZXAMActSaveFormat()
```

1.67 ZXAMProgVersion

ZXAMProgVersion: Returns the internal version of the emulator. This is for scripts that uses functions available only since certain version (if you want to know the version at which appears certain function, look at the HISTORY.

```
format:          version = ZXAMProgVersion()
```

1.68 ZXAMSaveFormat

ZXAMSaveFormat: Selects a format for SAVE. Can be PC, MIRAGE, PC_PP or MIRAGE_PP.

```
format:          ZXAMSaveFormat( formatname )
```

1.69 ZXAMParseLoaded

ZXAMParseLoaded: Tries to load in memory the block passed as string of bytes. If the format is known (PC, MIRAGE, Z80 or KGB, without compression) then the memory and registers will be loaded with the passed block. The function will return the name of the format of the block. If the format is unknown then the function will return an empty string, and nothing is modified.

```
format:          format = ZXAMParseLoaded( block )
```

1.70 ZXAMParseToSave

ZXAMParseToSave: Returns the contents of the memory and registers in the format selected for SAVE (without compression).

format: block = ZXAMParseToSave()

1.71 ZXAMJoinPathName

ZXAMJoinPathName: Joins the path and the name.

format: path_&_name = ZXAMJoinPathName(path , name)

1.72 ZXAMGetScr

ZXAMGetScr: Returns the actual Spectrum screen in interleaved format, 4 bitplanes, ready to be saved inside a IFF ILBM file.

format: block = ZXAMGetScr()

1.73 ZXAMClearNameFormat

ZXAMClearNameFormat: Deletes the 'Loaded file' and 'Format' boxes.

format: ZXAMClearNameFormat()

1.74 ZXAMLoadPath

ZXAMLoadPath: Modifies the path of the LOAD requester.

format: ZXAMLoadPath(path)

1.75 ZXAMSavePath

ZXAMSavePath: Modifies the path of the SAVE requester.

format: ZXAMSavePath(path)

1.76 ZXAMFilePart

ZXAMFilePart: Returns only the filename from a full path.

format: name = ZXAMFilePart(path_&_name)

1.77 ZXAMPathPart

ZXAMPathPart: Returns only the path from a full path.

format: path = ZXAMPathPart(path_&_name)

1.78 ZXAMPreRun

ZXAMPreRun: Modifies the PRERUN command line.

format: ZXAMPreRun(command_line)

1.79 ZXAMActPreRun

ZXAMActPreRun: Returns the actual PRERUN command line.

format: command_line = ZXAMActPreRun()

1.80 ZXAMPostRun

ZXAMPostRun: Modifies the POSTRUN command line.

format: ZXAMPostRun(command_line)

1.81 ZXAMActPostRun

ZXAMActPostRun: Returns the actual POSTRUN command line.

format: command_line = ZXAMActPostRun()

1.82 ZXAMPattern

ZXAMPattern: Modifies the pattern for the file requesters.

format: ZXAMPattern(pattern)

1.83 ZXAMActPattern

ZXAMActPattern: Returns the actual pattern for the file requesters.

format: pattern = ZXAMActPattern()

1.84 ZXAMBASICToken

ZXAMBASICToken: Returns the expanded token string equivalent to the character supplied. For example, `ZXAMBASICToken('ff'x)` will return 'COPY'. All characters from 0 to 255 are translated according to the Spectrum's character table.

format: `string = ZXAMBASICToken(character)`

1.85 ZXAMReadAbort

ZXAMReadAbort: Returns 0 or 1, depending if the 'Abort ARexx' gadget has been pressed.

format: `pressed = ZXAMReadAbort()`

1.86 ZXAMClearAbort

ZXAMClearAbort: Clears the flag that is set when 'Abort ARexx' is pressed.

format: `ZXAMClearAbort()`

1.87 ZXAMNoReload

ZXAMNoReload: Disables the 'Reload' gadget. This must be done when you leave permanently changed the contents of the 'Loaded File' box.

format: `ZXAMNoReload()`

1.88 ZXAMEnableAbort

ZXAMEnableAbort: Enables the 'Abort ARexx' gadget. This must be done if you want to read it (when running an ARexx script the gadget is disabled by default).

format: `ZXAMEnableAbort()`

1.89 ZXAMDisableAbort

ZXAMDisableAbort: Disables the 'Abort ARexx' gadget. If you don't want to read the gadget, you can disable it.

format: `ZXAMDisableAbort()`

1.90 ZXAMDisassemble

ZXAMDisassemble: Disassembles the Z80 instruction placed in the given address and returns a string formatted the following way:

- 1 ASCII character that says the number of bytes length of the instruction. This is useful to increment the memory pointer this number and thus point to the next instruction. This char must be excluded when printing the disassemble.
- the rest of the string is the typical Z80 disassemble in the usual 'address+object code+mnemonic' format. If the instruction disassembled is a relative jump (JR, DJNZ, etc) a comment will be appended to the line with the destination address of the jump. The illegal instructions will be presented as '---'.

The optional parameter "format" sets the format of the number printing of address and data. 'H' means hexadecimal and 'D' means decimal. The default is hexadecimal.

format: ZXAMDisassemble(address , [format])

1.91 ZXAMLoadMode

ZXAMLoadMode: Sets the mode for the LOAD patch (that means, sets the state of the LOAD MODE menu). The mode is a string ('TAPE', 'DISK' or 'AREXX').

format: ZXAMLoadMode(mode)

1.92 ZXAMSaveMode

ZXAMSaveMode: Sets the mode for the SAVE patch (that means, sets the state of the SAVE MODE menu). The mode is a string ('TAPE', 'DISK', 'DISK_PP' or 'AREXX').

format: ZXAMSaveMode(mode)

1.93 ZXAMActLoadMode

ZXAMActLoadMode: Returns the actual mode of the LOAD patch (that means, the status of the LOAD MODE menu). The result is a string ('TAPE', 'DISK' or 'AREXX').

format: mode = ZXAMActLoadMode()

1.94 ZXAMActSaveMode

ZXAMActSaveMode: Returns the actual mode of the SAVE patch (that means, the status of the SAVE MODE menu). The result is a string ('TAPE', 'DISK', 'DISK_PP' or 'AREXX').

format: mode = ZXAMActSaveMode()

1.95 ZXAMAbout

ZXAMAbout: Shows the 'About...' requester.

format: ZXAMAbout()

1.96 ZXAMFindBlock

ZXAMFindBlock: Search for a series of bytes in the Spectrum's memory. The first argument is the starting address for searching, and the second is the bytes for search in a string. That means, for searching 'HELLO' we must do `ZXAMFindBlock(0,'HELLO')`, and for searching the bytes 0F 45 7B we must do `ZXAMFindBlock(0,'0F457B'x)`. If the hexadecimal string is in a variable, then you must use `x2c(variable)` in the second argument. This way ARExx will translate the hexadecimal string into the corresponding bytes. The result is the address where the bytes have been found, or -1 if they haven't been found.

format: address = ZXAMFindBlock(address , block_for_search)

1.97 ZXAMDoChecksum

ZXAMDoChecksum: Does a checksum like the one that does the ROM's SAVE routine, that means, does a XOR (exclusive OR) of all the bytes of a block of bytes. The parameters are the starting address of the block and the length (number of bytes) of this block. The result is the XOR of all those bytes.

format: result = ZXAMDoChecksum(address , number_of_bytes)

1.98 ZXAMBreakPoint

ZXAMBreakPoint: Puts a breakpoint at the given address of the Spectrum's memory. If while the execution of code the Z80 arrives to the breakpoint, the emulator returns to the WorkBench (like if the 'HELP' key has been pressed) and the breakpoint will disappear. Only ONE breakpoint can be in memory. The execution of `ZXAMBreakpoint()` deletes the previous breakpoint. If the given address is 0 then the previous breakpoint will be deleted, and no new breakpoint will be put. The breakpoint is an instruction. That means that if the program modifies the memory address where the breakpoint is put, it will disappear. The breakpoint only survives to one entering into the emulator. As

soon as you press 'HELP' to exit to the workbench, the breakpoint will disappear (the exit for load/save to disk doesn't delete the breakpoint).

```
format:          ZXAMBreakPoint( address )
```

1.99 ZXAMTrace

ZXAMTrace: Executes the instruction addressed by the Program Counter (without disabling the system). At the return, the function returns the disassembling of the executed instruction (in the same format than ZXAMDisassemble()). The 'format' parameter sets if the disassembling is decimal ('d') or hexadecimal ('h'). It's optional, and the default mode is hexadecimal. Neither the CIAs nor the audio channels need to be allocated to run step-by-step code.

```
format:          diassem = ZXAMTrace( [format] )
```

1.100 ZXAMDoInt

ZXAMDoInt: Does an interruption (like if the INT signal has arrived to the Z80 processor), that means, puts PC in the stack, and sets the PC to the value appropriate to the actual Interrupt Mode (\$38 if IM1, and the vector pointed by I if IM2). If the interrupts are disabled the function does nothing.

```
format:          ZXAMDoInt()
```

1.101 ZXAMBlockOfRegs

ZXAMBlockOfRegs: Returns a block of 27 bytes that contains all the registers and internal status of the Z80 processor. The block's structure is:

Pos	Size	Contents
0	1	A
1	1	F
2	2	BC
4	2	DE
6	2	HL
8	1	A'
9	1	F'
10	2	BC'
12	2	DE'
14	2	HL'
16	2	IX
18	2	IY
20	2	SP
22	2	PC
24	1	Interrupt Status (0=disabled / 1=enabled)
25	1	Interrupt Mode (0, 1 or 2)
26	1	Border Colour (0 to 7)

The 2 bytes registers are in 68000 format, that means, the most significant

byte comes FIRST. This function is for passing the whole register set to other programs, like a disassembler/monitor, instead of having to get the registers one by one.

format: block = ZXAMBlockOfRegs()

1.102 Registers

REGISTERS: the registers for use with ZXAMGetReg and ZXAMSetReg are:

-8 bit registers: write his name (A, d, etc...). Even the indivisible 16 bit registers can be accessed by halves. For example, SPH and SPL are the two halves of SP.

-16 bit regiters: write his name, like HL or PC. AF isn't implemented.

-inverted 16 bit registers: swapping the letters of the name of a 16 bit register you will read/write on it but swapping the upper and lower bytes.

-special "registers":

INT : status of interrupts (0=OFF, 1=ON)

IM : interrupr mode (0, 1 or 2)

BOR : actual border colour (0 to 7)

-128k's sound chip registers (AY-3-8912): the 16 registers of the AY chip are available for read/write unader the names AY0, AY1... ..AYE and AYF (as you can see, the last character ofthe name is the register number, in hexadecimal). There is the additional register AXX, that contains the last byte sent to the \$FFFD port, that is used by the spectrum to indicate the AY register at which it want to read/write.

in all the cases you can append a '2' to the register name for accessing to the alternative set of registers (hl' = hl2 and a' = a2). The registers are case insensitive. All the registers can be read and written.

1.103 ZXAMAudio

ZXAMAudio: Sets the state of the 'Options -> Audio' menu. Possible values are OFF, FAST and CIASYNC.

format: ZXAMAudio(new state)

1.104 ZXAMAYEmulation

ZXAMAYEmulation: Sets the state of the 'Options -> AY-3-8912' menu. Possible values are OFF, ON and SMART.

format: ZXAMAYEmulation(new state)

1.105 ZXAMSpeed

ZXAMSpeed: Sets the state of the 'Options -> Speed' menu.
Possible values are FAST and AUTO.

format: ZXAMSpeed(new state)

1.106 ZXAMJoystick

ZXAMJoystick: Sets the state of the 'Options -> Joystick' menu.
Possible values are KEMPSTON, SINCLAIRRIGHT, SINCLAIRLEFT and CURSOR.

format: ZXAMJoystick(new state)

1.107 ZXAMTaskPri

ZXAMTaskPri: Sets the state of the 'Options -> TaskPri' menu.
Possible values are '+1', '0' y '-1'.

format: ZXAMTaskPri(new state)

1.108 ZXAMDisplay

ZXAMDisplay: Sets the state of the 'Options -> Display' menu.
Possible values are SCREEN, WINDOW, NOSYSTEMSCREEN and NOSYSTEMPAL.

format: ZXAMDisplay(new state)

1.109 ZXAMRefresh

ZXAMRefresh: Sets the state of the 'Options -> Refresh' menu.
Possible values are 1, 2, 4 and 8.

format: ZXAMRefresh(new state)

1.110 ZXAMActAudio

ZXAMActAudio: Reads the state of the 'Options -> Audio' menu.
Return values are OFF, FAST and CIASYNC.

format: state = ZXAMActAudio()

1.111 ZXAMActAYEmulation

ZXAMActAYEmulation: Reads the state of the 'Options -> AY-3-8912' menu. Return values are OFF, ON and SMART.

```
format:          state = ZXAMActAYEmulation()
```

1.112 ZXAMActSpeed

ZXAMActSpeed: Reads the state of the 'Options -> Speed' menu. Return values are FAST and AUTO.

```
format:          state = ZXAMActSpeed()
```

1.113 ZXAMActJoystick

ZXAMActJoystick: Reads the state of the 'Options -> Joystick' menu. Return values are KEMPSTON, SINCLAIRRIGHT, SINCLAIRLEFT and CURSOR.

```
format:          state = ZXAMActJoystick()
```

1.114 ZXAMActTaskPri

ZXAMActTaskPri: Reads the state of the 'Options -> TaskPri' menu. Return values are '+1', '0' y '-1'.

```
format:          state = ZXAMActTaskPri()
```

1.115 ZXAMActDisplay

ZXAMActDisplay: Reads the state of the 'Options -> Display' menu. Return values are SCREEN, WINDOW, NOSYSTEMSCREEN and NOSYSTEMPAL.

```
format:          state = ZXAMActDisplay()
```

1.116 ZXAMActRefresh

ZXAMActRefresh: Reads the state of the 'Options -> Refresh' menu. Return values are 1, 2, 4 and 8.

```
format:          state = ZXAMActRefresh()
```

1.117 ZXAMEmulToFront

ZXAMEmulToFront: Brigs to front the emulation screen.

format: ZXAMEmulToFront()

1.118 ZXAMControlToFront

ZXAMControlToFront: Brings to front the screen where the control window is open.

format: ZXAMControlToFront()

1.119 ZXAMActRun

ZXAMActRun: Reads the state of the emulation. Resturns a 0 if the emulation is stopped, and a 1 if is running.

format: estado = ZXAMActRun()

1.120 ZXAMAYBlockOfRegs

ZXAMAYBlockOfRegs: Returns a 16 bytes string containing all 16 register of the 128k's sound chip. The first byte is register 0.

format: string = ZXAMAYBlockOfRegs()

1.121 ZXAMWriteROM

ZXAMWriteROM: Writes a new ROM over the internal one. The new ROM must be supplied a s a 16384 bytes string.

format: ZXAMWriteROM(new ROM)

1.122 ZXAMChecksumString

ZXAMChecksumString: Returns the checksum of a string, that means, the XOR of all the bytes of the string. This is the kind of checksum used by the tape routines of Spectrum's ROM.

format: checksum = ZXAMChecksumString(string)

1.123 ZXAMCompareString

ZXAMCompareString: Compares a string with a portion of spectrum's memory. The returned value can be 0 if they are identical, or -1 if not.

format: result = ZXAMCompareString(address , string)

1.124 ZXAMStop

ZXAMStop: Stops the emulation.

format: ZXAMStop()

1.125 ZXAMTapeAction

ZXAMTapeAction: Returns a value that is the operation that must be done by the ZXAM_TAPE.zxam ARexx script. Possible operations are 0=nothing, 1=load, 2=verify y 3=save.

format: operation = ZXAMTapeAction()

1.126 ZXAMTapeLoadName

ZXAMTapeLoadName: This function returns a string previously stored with ZXAMTapeSetLoadName. These functions are used to store a string of up to 255 characters. In the supplied script ZXAM_TAPE.zxam are used to store the path+name of the actual .TAP file used for LOAD. When doing a Reset of the Spectrum this string is deleted, and the functions returns an empty string.

format: string = ZXAMTapeLoadName()

1.127 ZXAMTapeSaveName

ZXAMTapeSaveName: The same that ZXAMTapeLoadName, but the string is another one. The complementary function is ZXAMTapeSetSaveName. Used in ZXAM_TAPE.zxam to store the path+name of the actual .TAP file used to SAVE.

format: string = ZXAMTapeSaveName()

1.128 ZXAMTapeAddress

ZXAMTapeAddress: This function returns a 16 bit address, that is the address inside the Spectrum's memory of the block that must be SAVED by ZXAM_TAPE.zxam. When doing LOAD this address is supplied too, but is not relevant.

format: address = ZXAMTapeAddress()

1.129 ZXAMTapeSize

ZXAMTapeSize: Returns the size of the block to SAVE though ZXAM_TAPE.zxam. When doing LOAD this size is supplied, and can be used to foresee possible LOADING errors, or to search the correct block. The supplied ZXAM_TAPE.zxam ignores this as it accesses to the .TAP file sequentially, like with a real tape.

format: size = ZXAMTapeSize()

1.130 ZXAMTapeFlag

ZXAMTapeFlag: Returns the 8 bit flag for the block to SAVE. When doing LOAD this is the requested flag of the block to load.

format: flag = ZXAMTapeFlag()

1.131 ZXAMTapeLoadOffset

ZXAMTapeLoadOffset: Returns a number previously stored with ZXAMTapeSetLoadOffset. Created to store an offset inside a file (like ZXAM_TAPE.zxam does), but can be used for any purpose. When doing Reset of Spectrum this number is cleared to 0.

format: offset = ZXAMTapeLoadOffset()

1.132 ZXAMTapeSaveOffset

ZXAMTapeSaveOffset: Just like ZXAMTapeLoadOffset but with another, separated, number. The corresponding function is ZXAMTapeSetSaveOffset.

format: offset = ZXAMTapeSaveOffset()

1.133 ZXAMTapeSetResult

ZXAMTapeSetResult: This function must be used to return the result of the operation done by ZXAM_TAPE.zxam. The first parameter is the result code: 0=none, 1=OK and 2=error. 0 is the value returned when the script doesn't support the requested operation and wants to pass the operation to the internal .header/.bytes requester (as if the mode were 'Disk' instead of 'ARexx'). 1 is the result that indicates that the operation has been completed successfully. 2 is to return an error, that will do a BREAK inside the emulation.

The second parameter must be specified only when the requested operation has been LOAD or VERIFY and the result is 1 (OK). This parameter is the path+name of the file that must be loaded in the spectrum's memory (usually is a temporal file extracted from a .TAP file).

format: ZXAMTapeSetResult(result , [result file])

1.134 ZXAMTapeMark

ZXAMTapeMark: Returns a number previously stored by `ZXAMTapeSetMark`. Created to store a 'mark' (an offset) over a .TAP file to 'rewind' to that mark when needed.

format: offset = ZXAMTapeMark()

1.135 ZXAMTapeSetLoadName

ZXAMTapeSetLoadName: This function is used to store a string, that can be retrieved with `ZXAMTapeLoadName()`.

format: ZXAMTapeSetLoadName(string)

1.136 ZXAMTapeSetSaveName

ZXAMTapeSetSaveName: This function is used to store a string, that can be retrieved with `ZXAMTapeSaveName()`.

format: ZXAMTapeSetSaveName(string)

1.137 ZXAMTapeSetLoadOffset

ZXAMTapeSetLoadOffset: This function is used to store a number, that can be retrieved with `ZXAMTapeLoadOffset()`.

format: ZXAMTapeSetLoadOffset(number)

1.138 ZXAMTapeSetSaveOffset

ZXAMTapeSetSaveOffset: This function is used to store a number, that can be retrieved with `ZXAMTapeSaveOffset()`.

format: ZXAMTapeSetSaveOffset(number)

1.139 ZXAMTapeSetMark

ZXAMTapeSetMark: This function is used to store a number, that can be retrieved with `ZXAMTapeMark()`.

format: ZXAMTapeSetMark(number)

1.140 Tech info

Program written in assembler from the first lines to the full 17750 actual lines (151023 lines when expanding MACROs!!!). In my Amiga 1200 with hard disk and 68030/28MHz the Devpac 3.02 assembles the emulator in 51 secs (3 min 50 secs with a bare A1200, and 4 min 53 secs for the last version assembled with my Amiga 500 Plus). Well, if programming is fun... BIG programming is BIG fun!

This is my third program in assembler. I began to program it in my Amiga 500 Plus with 2 Meg of RAM and 3 floppy drives. As soon as I can I bought an Amiga 1200 and converted the emulator to AGA-only and 68020 code.

I began the emulator to experiment with the 68020 code. At the beginning I was programming conditionally in 68000 and 68020 code, and testing the emulator in the Amiga 4000/030 of a friend. With the new Amiga 1200 I have experimented with the AGA chips, and discarded completely the 68000 code.

1.141 About this version....

Firts of all, this emulator it is no longer named Spectrum Emulator AGA. This version is the 'ZXAM Spectrum Emulator v2.0', that includes two main versions, the Generic one for all Amigas with 68020+ and AmigaOS 2.04+ (including AGA machines), and the 'Special AGA' version, that has a much faster graphic engine, but at the expense of losing all the multitasking capabilities of the Generic version.

But, why this version? and the promised 128k version?

I suppose someone is thinking this right now....well, I will try to explain the release of this version.

I never had any intention to make multitasking the emulator....but, well, I never had intention to make the OCS/ECS version too, so seems that my previsions never had been very good....

The multitasking thing started when I begun the work to convert to 128k the emulation, and started to restructure nearly the whole emulator (even rewriting from scratch some parts). When the graphic system of the ECS version where the next 'target' I decided to modify it to work with system friendly screens, and solve the problem of people that use monitors that can't work in PAL mode. Once this was done, I passed to the step-by-step code execution. This code (implemented in version 1.6) was simply a way of executing Z80 instructions without disabling the system (from ARexx), and thus, it was fairly multitasking yet. Well, the thing is that I had 'legal screens' and 'multitasking Z80 emulation', so with some slight(ish) modifications it was fairly easy to have the emulator running in multitasking...

Well, here is the result of all that story. The multitasking has been much more difficult to implement than what I though initially, but I think it has been worth the effort. I hope you think so too.

There are still several things to do in this version, but the 128k emulation has been delayed too much, so I decided to release this version in its actual state, and retake the 128k work. While I am at the hard work of developing the

128k version you can enjoy the multitasking, the little speccy in a window, the 128k sound, the automatic speed adjustment, etc...

This is the list of 48k games and demos with 128k sound that I have. All of them sound quite well with ZXAM 2.0:

180\textdegree{}

ATF

Blazing Thunder

Bobsleigh

Brat Attack

Captain Fizz

Carlos Sainz

Complete Bastard

Crazy Cars II

Crazy Cars

Cybernoid

Double Dash

Egg Head II

Enduro Racer

Exolon

Goody

IK+

Loopz

Los Inhumanos

Mega Phoenix

Motos

Mountain Bike Racer

Mutant Zone (1)

Mutant Zone (2)

Plotting Right speed even with a bare 1200

Pulsator

Rex (1)

Rex (2)

Shoot Out

Silk Worm

Skateboard Kidz

Stormlord II

Stormlord

Tetris

The A Team (1)

The A Team (2)

The Fury

The Last Mission

(Demos)

Antares II

Carminadle 3

Digi Synth 2

Gallery

Gemini Even 4 bit samples

Geography Demo

Judgment Day 3

LCD Demo

Noname 2

TFA Demo

Even 4 bit samples

1.142 The worse part of any DOC....

First of all I want to explain that this is nearly a β version (I was tempted to release it as β) because it hasn't been fully tested yet, and the modifications done are more than enough to create lots of bugs. If you find a bug, please report it to me as soon as possible, and I will try to fix it.

* Problems with the multitasking priority of the emulation:

Is not recommendable to raise the emulation priority to +1. If so, any ARexx script that you try to run will get blocked because REXXMAST creates a task with priority 0 to run the script, and the emulation running at +1 could potentially get all the CPU time, so the script never will get the necessary CPU to run. If this happens, you will have to select the display screen/window and press HELP to stop the emulation. Then the script will run as expected.

Anyway, putting the emulation priority to +1 the speed up will not be noticeable, unless you have several priority 0 tasks 'eating' CPU time. In that case, the emulator will get more CPU time, but the tasks with less priority will get completely blocked. It's much better to use a priority 0, or even -1.

Another possible problem is that when running several copies of the emulator in multitasking you never must set their priorities to different values. If you set their priorities to different values, the copies with the higher priority could block completely the other copies, unless you have enough processing power.

* Border color in NOSYSTEMSCREEN mode:

In the NOSYSTEMSCREEN mode (that means, NON multitasking, with system friendly screen) the border color can't be permanently changed (it's always black). This is due to the impossibility to change the 0 color of a legal screen while multitasking is switched off. I hope to solve this problem in the next version. In NOSYSTEMPAL mode the border can be changed without problems.

The best solution is to use NON multitasking modes only for LOAD/SAVE to tape. For general use it's better to use the multitasking modes. The speed difference between SCREEN and NOSYSTEMSCREEN is minimal, unless there are other tasks using too much CPU time.

Another thing is the border lines of the LOAD/SAVE to tape routines. If you use a screenmode different than PAL, the height of the border lines will be different to the original lines of the Spectrum. This is due to the different horizontal frequency of the mode. It is harmless, and the tape routines will work as usual.

The last detail is that in NOSYSTEMSCREEN mode, being a legal screen, is affected by the borderblanking if you use it (with BBlank command or any other). This will not let you to see the border, being a real problem when

using the tape LOADING routines, or the transfer. To see the border you can do (in the icon) PRERUN=BBlank and POSTRUN=BBlank. This way the borderblanking will be disabled when starting the emulation, and enabled when stopping it. This will work only in NOSYSTEMSCREEN mode, not in multitasking SCREEN mode.

* No 'turbos':

There isn't still possibility to do a 'real' loading, for turbos and other protected games. The timing necessary to do this is extremely accurate, and the emulator isn't still prepared to give this kind of precision. This possibility will be seriously investigated in future versions.

* Problems with Enforcer:

During the tests with Enforcer I have noticed that the emulation is quite slower when Enforcer is running. I don't know exactly the causes of this behaviour, but I suspect that could be because of the control over the CPU caches that Enforcer does. If you want the emulation at full speed you must disable enforcer.

* Automatic speed with the display window on a public screen:

The 'Display -> Window' mode slows down slightly the automatic speed system, even when the processing power is more than enough to run at the right speed. This problem comes from the system, not from the emulator. Seems that the problem is some kind of inaccuracy of the timer.device when redrawing the display window. Right now I haven't found a solution to this problem, but the speed difference is very little, so you can use this mode without worrying about it.

* Problems with 48k sound in some games:

In some games (like Alien 8) the 48k sound suddenly disappears, or strange noises come from the speakers. This is due to the bad programming practices of some spectrum coders, sending strange data to unused I/O ports that sometimes are the same that later where used by the 128k spectrum to access to the AY chip. In the case of Alien 8, it sends a byte to the AY chip that enables the noise channel, that shares the audio channel with the 48k sound, disabling automatically the later. The only solution to this problem is to disable the AY emulation (menu 'AY-3-8912 -> Off').

1.143 ZXAM versions

Estas son la versiones que ha habido, hasta ahora, del ZXAM:

- 1.0 β First distribution (AGA only).
- 1.1 β (non-public version).

- 1.2 B Created FULL and FAST versions.
- 1.3 B Included LOAD/SAVE from tape.
- 1.3b B Bug fix and transfer.
- 1.4 B ARexx Port (non-public version).
- 1.5 Created OCS/ECS version.
- 1.6 Z80 disassembler and trace.
- 1.6b Bug fix and more formats.

.....a big step forward.....

- 2.0 Multitasking and 128k sound.

1.144 1.0 B

- 1.0 B 39.00 (11-March-94)
 - First public release.
- (distributed as ZXAM Spectrum Emulator AGA 1.0 B)

1.145 1.1 B

- 1.1 B 39.01 (2-April-94) (renamed ZXAM_SureCrash)
 - Audio filter off when emulating. Restored at exit.
 - Disabled nearly all of the interruptions. The sound is now a lot better.
 - The system keyboard handler 'freezes' after a while. I have to use this handler because of the new keyboard hardware of the Amiga 1200 (snort!).
 - 39.02 (19-April-94)
 - Added the ZOOM gadget to the window.
 - The #&\$&! system keyboard handler is still 'freezing'.
- (never distributed, of course)

1.146 1.2 B

- 1.2 B 39.03 (26-January-94)
 - ;At last! Fixed the problem with the system keyboard handler.
 - Now, if you press the Caps Lock key the system will be informed correctly about its state.
 - Available the FULL (complete) and FAST (faster) versions. The FAST version is about 10% faster, but it is a less detailed emulation (no BCD, no R register, etc...).
- 39.04 (27-January-94)
 - The emulator 'detachs' from the SHELL window.
 - The icon ToolTypes will be read even if the emulator runs from the SHELL.

39.05 (29-January-94)

-No more audio problems. If a player (like Delitracker) is playing when starting the emulation, it will be stopped until return to the WorkBench (at least, with the players that I have).

39.06

-The Timer-B of CIA-B is allocated.

-Preparatory version with some synchronizations with CIA Timer for future implementations of LOAD/SAVE to tape.

(distributed as ZXAM Spectrum Emulator AGA 1.2 B)

1.147 1.3 B

1.3 B 39.07 (7-July-94)

-First attempt of patching the ROM LOAD routine. Now you can LOAD from tape even in colour mode, with an Amiga 1200 with no FAST memory, and even with the FLASH working during the LOAD (!!). The patch seems to be very good, and works perfectly over the Amiga 1200 and Amiga 4000/030, but needs testing in other configurations.

-Synchronized the OUT instruction. With this we have REAL TIME SAVE (not patched!!), and a pure tone BEEP. Some games now sounds A LOT better, but other have slowed down a little.

39.08 (11-July-94)

-REAL border! Now, if there is enough speed, you can see the fancy border effects that some programs do.

-Readjusted the proportion of bright added to the colours when BRIGHT 1. Now looks much more like the original Spectrum.

-Readjusted the LOAD patch. Still loads very well, but now reacts better to the BREAK, HELP and ESC keys.

-At last! I have managed to do that the lines counter of Devpac 3.02 "turns around" for the second time. Now the assembled lines (with the macros expanded) are 133637 (!!).

39.09 (16-July-94)

-Fixed the problem of the LOAD patch and OUT instruction with the border. Now the border is nearly perfect. Some border changes are missed because of interferences between the CPU and the copperlist. To see the border lines while LOADING or SAVEing is more than enough.

(distributed as ZXAM Spectrum Emulator AGA 1.3 B)

1.148 1.3b B

1.3b B 39.10 (23-July-94)

-The emulator now includes a "transfer", that can pass Spectrum programs from disk to tape. Initially included for my personal use for the speed comparisons between

- my Amiga 1200 and my old Spectrum +2A.
- When the program is saved to tape, it will be saved with the name used to load it from disk.
- The transfer don't uses his own LOAD routine. It uses the ROM load routine. That means that even the emulator itself can load "tranferred" programs.
- The transfer can be stopped with the HELP key.
- The transfer saves even de border colour.
- When you do "Save Program" the new name is passed to the "Loaded file" box. This way we can do "Reload" of the last saved program.
- The SAVE requester stores a different path that the Load requester. This way is easier to do multiple format conversions.
- If no program is loaded, the transfer saves the program with the name LOADER.

39.11 (31-July-94)

- Some adjustments done on the emulator synchronization. The sound is still stabilized, but some programs (like Manic Miner) have speeded up a little.
- Fixed a bug with the transfer.
- Fixed the problem of Kempston joystick with Panama Joe and Commando.
- The border changes are now even faster.

(distributed as ZXAM Spectrum Emulator AGA 1.3b ß)

1.149 1.4 ß

1.4 ß 39.12 (2-September-94)

- Fist attempt to add an ARexx port to the emulator. It works very well. Impemented functions GetReg, SetReg, Peek, Poke and Putmem.

39.13 (10-September-94)

- Now the menus are NewLook.
- Added several ARexx functions. Implemented functions DPeek, DPoke, FindByte, and the reversed access to registers.
- Added code to run ARexx scripts, but there isn't gadget for that thing yet.
- Written some scripts.

39.14 (16-September-94)

- Added internal version to About...
 - Added the ROMFILE ToolType, to load external ROMs. The ROM can be compressed with powerpacker.
 - The name of the last loaded/saved program now appears in the SAVE filerequester, but with the extension of the actually seleted SAVE format.
 - Added more ARexx functions: ProgVersion, ActSaveFormat, SaveFormat, ParseToSave, ParseLoaded, JoinPathName, ClearNameFormat, GetScr, FilePart, PathPart, LoadPath and SavePath.
 - Modified the scripts to use the "powerpacked" functions.
-

- Added the 'Run ARexx' gadget. The gadget opens a filerequester to select the script you want to run.
- Added ToolType REXXPATH to specify the path where open the 'Run ARexx' filerequester. Extension is always '.zxam'.
- When you run it from the CLI, it will do CD to the path where the executable is placed. Every path must be relative to the executable's directory, or must supply a full path.

(never distributed)

1.150 1.5

- 1.5 39.15 (28-September-94)
- Added the option to save the programs compressed with powerpacker.library.
 - Reused the 'Save format' buttons. Now there are 2 buttons for each format (raw and compressed).
 - Added my FidoNet address to the About ;-)
 - Added more ARexx functions: ProgVersion, ActSaveFormat, SaveFormat, ParseToSave, ParseLoaded, JoinPathName, ClearNameFormat, GetScr, FilePart, PathPart, LoadPath, SavePath, PreRun, ActPreRun, PostRun, ActPostRun, Pattern, ActPattern and BASICToken.
 - If the ARexx port can't be created successfully the 'Run ARexx' gadget is disabled.
 - Added the option to add a title to the LOAD and SAVE filerequesters when used from ARexx.
 - Now the emulator returns properly the memory when there is no WB 3.0+, 68020+ or AGA chipset.
 - The window is an AppWindow. That means that you can drop icons inside the window and it will be loaded. If you drop a disk/drawer the filerequester will be opened in that disk/drawer.
 - Added ToolTypes PRERUN y POSTRUN for running of commands right before the emulation is started, and before the emulation is stopped.
 - Before PRERUN the caches status is stored, and is restored before POSTRUN.
 - The emulator is no longer ß (BETA). That means that the interface is complete and the emulator is stable. The precision of emulation isn't included in the ß consideration, because if it were included the emulator could remain ß forever!! This is NOT a resignation to improve the emulation ;-)
- 39.16 (12-October-94)
- Added ARexx functions: ClearAbort, ReadAbort and NoReload.
 - Added the 'Abort ARexx' gadget, to stop the ARexx scripts. Anyway, the script is the one that must read the gadget and decide the appropriate action.
- 39.17 (27-October-94)
- Modified some instructions that uses the P/V flag.
 - Created an OCS/ECS version of the emulator. This version don't have FLASH attributes (instead they are
-

inverted) and the colour emulation is slower than the AGA version. These are the only differences between the OCS/ECS version and the AGA version.

(distributed as ZXAM Spectrum Emulator 1.5)

1.151 1.6

1.6 39.18 (14-November-94)
-Now the emulator allocates correctly the audio channel 0. If the channel isn't available, a requester is showed and the emulation will not run.

39.19 (6-December-94)
-Added the ZXAMDisassemble ARexx function, that disassembles Z80 code from the spectrum's memory.
-Fixed a problem with keyboard and some games (like the DINAMIC ones).
-Added a menu to enable/disable the audio-CIA sync.
-Added the CIASYNC tooltype.
-Added a menu to select the mode of the ROM's SAVE routine. Can be redirected to TAPE, DISK or DISK crunching the data with powerpacker.library. Pressing CANCEL a BREAK will be performed inside the emulator.
-Added a menu to select the mode of the ROM's LOAD routine, that can be redirected to TAPE or DISK.
-The CIA reading is better.
-Now the emulator can use any CIA available. Before the emulator needed specifically the TIMER-B of CIA-B.
-The CIA timer is allocated only during the emulation.
-Fixed a problem of the FLASH with processors with data cache (030 and 040).
-Added support for loading the Z80 snapshots.

39.20 (22-December-94)
-Fixed a problem with Arexx scripts and paths with spaces.
-Added the LOADMODE and SAVEMODE tooltypes.
-Fixed a problem with the filename extension when doing 'save program'.
-The games can be loaded by their 'default tool' o doing multi-selection.
-Added to the Disassemble function the possibility of printing addresses/data in decimal or hexadecimal.
-Added to Disassemble the printing of destination address of relative jumps.
-Now the transfer is no more affected by the Save Mode menu, and always does the SAVE to tape.
-Added support for loading NEW Z80 format (2.01). Older versions only can load the OLD Z80 format (1.45).
-Modified the ZXAMParseLoaded() ARexx function to load and parse Z80 snapshots.
-Removed the menu that enabled to enable/disable the audio sync with CIA. Removed the CIASYNC ToolType too.
-Fixed a minor bug when saving in PC_PP format.
-The emulator now allocates the 4 audio channels, for a

forthcoming AY-3-8912 emulation.
-The emulator is now crunched with Imploder 4.0. Before it was crunched with powerpacker, but the decrunching header caused some gurus in low memory conditions.

(distributed as ZXAM Spectrum Emulator 1.6)

1.152 1.6b

1.6b 39.21 (19-January-95)
-Fixed a bug in the loading of Z80 snapshots. The emulator refused to load old (v1.45) Z80 snapshots if they were compressed. This bug was due to wrong information in the "Z80" v1.45 manual :-)
-Added the possibility of loading (but not saving) snapshots in KGB format.
-Modified some instructions. Now these instructions increments correctly the R register.
-Some enhancements in the flags emulation.
-Added an ARexx script for disassembling, BASIC listing searching of pokes, etc... It's a kind of monitor (Made by Leo Cocaña. Thanks Leo! :-)
-Fixed two bugs in the embedded disassembler (in the 'ld hl,(nn)' and 'ld (nn),de' instructions).
-Added the ARexx functions LoadMode(), SaveMode(), ActLoadMode() and ActSaveMode(). This way the new menu 'options' can be controlled from ARexx too.
-Added the ARexx function About()
-Modified the ARexx function ParseLoaded() for parse the KGB snapshots too.
-Now 'Reload' works after a 'Save program'.
-After 'Reset & Run' the 'Reload' shortcut is disabled (like the gadget).
-Embedded a 1 second pause at the end of the ROM's SAVE routine. This is for programs that SAVE headers and blocks without pause between them (like the GENS assembler).
-Added ARexx functions FindBlock() and DoChecksum().
-Added ARexx functions BreakPoint(), Trace(), DoInt() and BlockOfRegs(). With these functions you can execute Z80 code step-by-step, and put breakpoints.
-Modified the ARexx function GetReg(). Now it has a second parameter (optional) that sets the format of the result (decimal or hexadecimal).
-Now, when an ARexx script is running, the 'close window' events aren't queued anymore, and the menus are disabled.
-Fixed a bug in the 'Reload' routine after doing SAVE in PC_PP format.

(distributed as ZXAM Spectrum Emulator 1.6b)

1.153 2.0

- 2.0 39.22 (6-february-95)
- After several weeks of modifications and restructurations of teh sources, seems that the emulator runs again.
 - Bug found...in Devpac!!
 - Added support for 'legal' screens in the ECS version.
 - Added a simple multitasking to the ECS version.
 - Added automatic speed ajustement for too fast machines.
 - Fixed incompatibility with 3D-Tennis, Amaurote and Aquaplane.
 - Added a handler for the input.device to read the keyboard in multitasking.
 - Added support for multitasking in a screen or inside a window over the workbench (ECS version).
 - Used a completely system friendly code to redraw the display window. This way the emulation can be used with future versions of harware, and even with RTG cards.
 - Added a color adaptation system (only kickstart 3.0).
 - The display window is an AppWindow, so you can drop icons inside the window to load the snapshots.
 - The keys are read only when the display screen/window is selected.
 - Added support to disable the sound.
 - The pull down menus no longer use the topaz font. Now use the default font for the actual screen. The control window still uses topaz.
 - The name of the actual snapshot is shown in the titlebar of the display window.
 - The height of the window when 'zoomed' in adjusted for the font used in that titlebar.
 - Now the TAB key performs the function of the 'Graf' key (Shift+9).
 - The snapshots can be loaded while the emulation is running. These snapshots will be run automatically.
 - Enhanced the Reset key to avoid lockups of the emulation.
 - Fixed a bug in the BREAK emulation when the CANCEL is pressed in a LOAD/SAVE requester.
 - Fixed a problem with the loading routine for Mirage snapshots. Some bad saved snapshots wasn't correctly identified because of strange values stored in the 'border' byte.
 - Included emulaion for SINCLAIR (Interface 2) and CURSOR (AGF/PROTEK) joysticks.
 - The joystick is now read like the keyboard, only when the display window/screen is selected.
 - Modified the copperlist in the ECS version to use the additional bandwidth of the AGA chipset.
 - Implemented a very basic emulation of 128k sound (the AY-3-8912 chip).
 - Added the AUDIO, AYEMULATION, SPEED, JOYSTICK, TASKPRI, DISPLAY and REFRESH tooltypes, to control the new options.
 - Added the AY chip registers to the registers accessible from ARexx. The registers are AY0, AY1.....AYE, AYF, and the extra AYX register, that is the latest byte sent o the \$FFFFD port.
 - Added the loading code for Z80 v3.0 snapshots.
 - LOAD/SAVE to tape are available only in multitasking.
 - Now the main task priority is always one point higher than the priority of the emulation task, to avoid the blocking of the window by the emulation.
 - Now the audio channels and the CIA are allocated only when are
-

necessary.

- Added the SCREENMODE tooltype, to specify the mode that must be used for the 'legal' screen.
- Removed all VBlank synchronization, as in multisync/VGA screen it has no sense.

39.23

- Now the display screen is centered over the Overcan preferences of the system.
- Added menu for the new mode NON multitask but with 'legal' screen.
- Now the automatic speed control returns to the system the unused CPU time, for the other tasks.
- Now when the display is on a window, only the available bitplanes are processed. This speeds up the emulation in 8/4/2 color screens.
- Simplified the color adaptation system. Now runs over kickstart 2.04 too, but the results can be a little strange if the screen colors are too different of the original spectrum colors.
- Now several copies of ZXAM can be running in multitasking without interferences.
- Now in multitasking the automatic speed doesn't need a CIA. Instead it uses the timer.device. This way all the copies of ZXAM running in multitasking can be stabilized.
- Tested succesfully the emulation over an emulated workbench, with a Retina Z2 card :-)
- Added the PUBSCREENNAME tooltype, to specify the public screen where to open the control and emulation windows.
- Added a warning requester that is shown when the emulation code is running over CHIP memory.
- In NON multitasking modes the audio chanel's are always allocated, to avoid the disabling of the system while a player (like delitracker) is running.
- Now when the display window is opened over a public screen, the screen is moved to front.
- The control window itlebar now shows if the emulation is running or is stopped.
- Under kickstart 3.0 the BestModeID function is used to get the monitor used for the workbench if no screenmode is specified.
- The menu strip is now atatched to the display window too.
- The emulation is stopped when the load/save snapshot requester is open.
- Now a CLI parameter can be supplied, that is the name of a snapshot to load.
- The main (control window) task has always a priority +2, to avoid blocking of the window because of other copies of emulation running at high priorities.
- Enhanced slightly the use of the instruction cache of 020+.
- Added the white noise channel to the 128k sound emulation.

39.24 (y 37.24)

- Finished the AY chip emulation. Now includes the volume waves and a dynamic rearrangement of channels for a better mixing of tone+noise.
 - When running several copies of ZXAM, only the first one has ARexx port.
 - Added ARexx functions Audio(), AYEmulation(), Speed(),
-

- Joystick(), TaskPri(), Display(), Refresh(), ActAudio(), ActAYEmulation(), ActSpeed(), ActJoystick(), ActTaskPri(), ActDisplay(), ActRefresh(), WriteROM(), AYBlockOfRegs(), ActRun() and CheckSumString().
 - Now the display window is redrawn when a Poke is done from the control window.
 - When the ARexx port hasn't been created, all the ARexx options are completely disabled.
 - Added the ARexx control of load/save. Now the ROM load/save can be redirected to the ARexx script ZXAM_TAPE.zxam, to use complex formats like .TAP. Added the ARexx functions: TapeAction(), TapeLoadName(), TapeSaveName(), TapeAddress(), TapeSize(), TapeFlag(), TapeLoadOffset(), TapeSaveOffset(), TapeSetResult(), TapeSetLoadOffset(), TapeSetSaveOffset(), TapeSetLoadName() y TapeSetSaveName().
 - Added the execution of the ZXAM_INIT.zxam ARexx script when loading the emulator.
 - Added the execution of the ZXAM_EXIT.zxam ARexx script when quitting the emulator.
 - Docs converted to AmigaGuide (a lot of time required for this).
 - Added menu AY-3-8912 -> Smart, and the adequate tooltypes and ARexx functions to control this menu.
 - The ECS and AGA version now have different internal version numbers, so the ARexx scripts can know what version is running.
 - Added the OUTOUT ttoltype to give an output channel to the ARexx sccripts. Very useful for debugging the ARexx scripts.
 - Added ARexx functions CompareString(), ControlToFront(), EmulToFront(), TapeMark() and TapeSetMark().
 - Now when doing a Reset, the TapeLoadName and TapeSaveName are cleared, and the offsets put to 0.
 - Now the emulator will run even if the window couldn't be converted to AppWindow. This is for running the emulator without workbench loaded.
 - The transfer works again (only NON multitasking modes).
 - The 'Load/Save mode -> ARexx' options can't be selected when the ARexx port isn't available.
 - Fixed a bug in the transfer that trashed the ROM when stopped. Can be stopped with the HELP and the F6 key.
 - The transfer disables the 128k sound during the transference.
 - Completely lost the control of the lines counter of Devpac. I don't know the resulting lines at assembly time.
 - Added the 'Project -> Load Program' and 'Project -> Save Program' menus, with the adequate keyboard shortcuts. This way is possible to load/save snapshots from the emulation window.
 - The FASTER versions work again.
 - Adjusted the 'Speed -> Fast' mode to avoid an excessive speed with some programs (like the Silk Worm menu music).
 - Fixed a bug in the .Z80 (v3.0) loading routine.
 - Now the AY-3-8912 registers are loaded from the .Z80 snapshots (2.0 and 3.0).
 - Fixed a bug in the white noise routine of the AY emulation.
 - Fixed a bug in the audio allocation routine. Sometimes the 48k sound where generated even when the allocation failed.
 - Fixed a bug in the ARexx port. The second copy running had the ARexx port disabled, but when quitted it disabled the ARexx port of the first copy sunning.
-

- Readjusted the color table, as some dark colors where too dark.
- Fixed a bug in the transfer. When the snapshot name included more than one period ('.') the first one where supposed to be the extension separator. Now is the last period that is used as extesion separator.
- Now the AY-3-8912 emulation supports three channels of 4 bit samples, used by some demos.
- Added the Stop() ARexx function.
- Now the Kempston joystick is always active. With the 'Joystick' menu you can choose additional emulations, with the Kempston emulation still active.
- Now there are two Sinclair modes, the 'Sinclair Right' that emulates the right side joystick (keys 6, 7, 8, 9 and 0), and the 'Sinclair Left' that emulates the left side joystick (keys 1, 2, 3, 4 and 5).
- Fixed the ARexx functions BasicToken() and Trace().
- Now, when the emulation is running in multitasking, the following ARexx functions returns an error: GetReg(), SetReg(), ParseLoaded(), ParseToSave(), BreakPoint(), Trace(), DoInt(), BlockOfRegs(), AYBlockOfRegs(), WriteROM(), PutMem(), Poke() and DPoke().
- Disabled many functions, menus and tooltypes in FASTER and AGA versions, because they where of no use in these versions.
- Made some adjustments in the color comparation calculations. Now the color adaptation seems to be much better.
- Added a 'security' system that avoids the possibility that some graphics 'dissapear' from the display window due to the lack of colors in the screen. Now the emulator can be used even in 2 color screens.
- Included the ARexx script to load snapshots from DirOpus.
- Added the 'RUN' menu item.

(distributed as ZXAM Spectrum Emulator 2.0)

1.154 Future of the emulator

Easy, the future is the 128k emulation !

1.155 Thanks for all these people!

I wish to thank to:

- Fco Javier Cocaña Galán, Leonardo Cocaña Galán and Juan A. Estela Valín for their friendship and suggestions.
- Commodore-Amiga Inc for create the very best computer in the world.
- Motorola for their powerful and flexible microprocessors.
- Nico François for create the reqtools.library and powerpacker.library.
- Jan van den Baard for the incredibly useful GadToolsBox.
- IBM and Atari for create so bad computers, making very easy my choosing for the Amiga.
- Miguel Barnosi, Sysop of TANIT-BBS, for sending the emulator to several BBS of Spain, and for initiating me to the world of comms. And now for being the best FidoNet Boss in the Balearic islands ;-)
- Alberto Ordoñez Tellez, Raúl Ureña Sánchez, Arturo Rubio Pavón, Javier

López Cosialls and Sergi Martinez Colldeforn for encourage me and for their suggestions.

- Andrew Pointon (England) for sending to me some .Z80 snapshot (without it I couldn't manage to implement the Z80 format) and some .TAP files.
- Flávio Massao Matsumoto (Brazil) for sending to me the "ZX Spectrum FAQ" of Internet. Without it I couldn't manage to implement the KGB snapshot format.
- Jordy Mejias for passing to me the KBG disk, that I deleted a long time ago.
- Richard Harris (Southafrica) for the nice MagicWB icons that he sent to me for the emulator.
- Juan Gomez and Juan Jordana for uploading the 1.6b version to Aminet.
- All the registered users, for thinking that my work is worth the registration fee. LOTS OF THANKS.
- Rafat Krawczyk (MuabDib/ESI, Poland) for sending me the 128k music modules with the AYPlater for Delitracker II, and the speccy DEMOS, that where the main reason that encouraged to me to emulate the AY chip much earlier than my initial prevision.
- Alberto M. Ordoñez Tellez (Namek), for being a very perseverant games tester.
- All the people in the Amiga areas of FidoNet R34, where I feel like at home ;-)
- All the people that wrote to me from inside and outside Spain, for their encouragements and suggestions.

1.156 Where I am?

If you have any information about file formats, please, send it to me along with a disk with some programs saved in that format. If you find some bug (NAH!) report me the conditions in that it was found (processor, memory, etc..). If the problem is with a Spectrum program, send it to me.

In this moment I am searching for:

- 48k programs with 128k sound (to test the AY emulation and fix all possible bugs).
- Information about te Fuller joystick, to emulate it.
- Information about Spectrum mouses (AMX, Star, etc...) to emulate them.
- Spectrum Demos, and the tracker for the AY chip.
- A color comparison algorithm, as I don't know of a good one and the color adaptation of the emulation window isn't the best one.
- The format of modules for the AYPlayer (the 128k music player for Delitracker II).

For sending suggestions, bugs, information interchange, etc., contact me at the following address:

Antonio J. Pomar Rosselló
C/ Alférez Cerdá nº 13-bajos
Palma de Mallorca 07014
Balears (Spain)

You can send me e-mail through FidoNet: Toni Pomar (2:347/11.3 and 2:343/119.80)
or AmigaNet: Toni Pomar (39:195/1.3 and 39:190/1.80).

(if you have access to both FidoNet and AmigaNet, please send the messages through AmigaNet)

Through FidoNet you can get the latest ZXAM version available by doing FileRequest to Tanit BBS-Ibiza (+34-71-392829, USRobotics 28.800 baud) with the magic name 'ZXAM'.

or InterNet: tpomar@penedes.mazanet.es

(the old tpomar@penedes.mazanet.encomix.com address is out of order)

(my Internet access is limited and slow, so if you send me a message, prepare for a delay of 10 or 15 days to get a reply)

Palma de Mallorca 8-June-1995

NOTE: my knowledge of English is VERY limited, and this text is a good sample of this fact. If you know spanish, take a look at the spanish Guide. Maybe there are much less typos and mistakes there.

1.157 Message for registered users

First of all, I would like to apologize because of the really HUGE delay in answering the letters of the registered users. I had lots of time, and programming, problems.

After this I would like to explain that this is not a full blown release. Instead it's a half way stage of the emulator towards the 128k version (in fact this version is 'nearly B'), but that I have distributed to show the actual stage of development, and that I am not sleeping all the day :-)

Because of this 'transitional' stage the emulator still hasn't a key system for the registered users (future releases will use a key system).

ALL the registered users will receive the next 128k version, as soon as possible, along with the correspondent key. I just ask for a little patience, as there is still a LOT of work to do in order to have a really good 128k emulation. The initial tests had been discarded, as the resulting emulation where too slow. Now I have a nicer system for the 128k emulation, that promises good results, but is much more complex and requires a lot of time to implement.

I hope to have the 128k version ready for the autumn...
