Speculator

William James

COLLABORATORS			
	TITLE : Speculator		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	William James	July 25, 2024	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

Speculator

1.1 Speculator Manual

Speculator'95

Version 1.03

A ZX Spectrum emulator

for Commodore-Amiga computers

1.2 Introduction

1.00 Introduction - overview of Speculator

Speculator is a 48K ZX Spectrum emulator for all Commodore Amiga computers with Kickstart 2.0 or greater. A fast processor is recommended. It is shareware and may be freely copied in an unaltered form. It is planned to release annual updates depending on the general response to this initial release. So the next will be Speculator 1996. Contributions and suggestions are very welcome. Speculator can use 48K MGT snapshots (images of Spectrum memory) and all standard Spectrum file types on any AmigaDOS directory device. It comes with a variety of example files already in AmigaDOS format. Amiga keyboard handling is fast and comprehensive.

Speculator is faster and easier to use than many other ZX emulators because it is system-friendly and key routines in the Spectrum 16K ROM have been re-written to run in the native 68000 code rather than interpretatively in the original Z80 code. This speeds up program editing, display output and other common routines, while also allowing direct access to Amiga devices. It is quite practical to develop ZX BASIC and machine code programs as if you were using a Spectrum, with full Amiga hardware at your disposal - real keyboard, floppy drives, hard drives, ram disk, rock steady screen unknown on a 'real' 48K Spectrum.

Speculator is also unusual in that it runs as a single task, updating the Amiga screen immediately whenever a program tries to write a change to the ZX screen area. This limits speed in some circumstances but may be faster and gives a smoother display than is possible with multi-tasking emulators that periodically copy from the ZX screen area to the Amiga display.

1.10 Reference material

1.20 Overview of the main features of Speculator

1.3 Reference material

1.10 Reference material

This documentation assumes familiarity with the Spectrum. The 'best book' is the original orange 48K or later Amstrad 128K Spectrum manual, but hundreds more are available from User groups, 'All Formats' shows and Microfairs, Radio Rallies, and even a few shops. The definitive Z80 reference is Rodnay Zaks 'Programming the Z80', ISBN 089588-057-1. The 'worst book' I have come across is the 'Spectrum Graphics Compendium' by David Durang (Pitman Publishing Limited ISBN 0-273-02170-2), this has to be seen to be believed. 1000's of Spectrum magazines have been published; old multi-format magazines often contain interesting listings which can be entered and modified. A wealth of ZX documentation, knowledge and software is available after more than a decade, often at very low prices.

1.4 Overview of the main features of Speculator

1.20 Overview of the main features of Speculator

o Speculator is OS friendly and runs on all versions of the 680X0.

o The Workbench 2.0/3.0 environment and AGA mode promotion is supported.

o Snapshot files in six different formats are supported transparently.

o Support for all the standard Amiga keys as well as the main 40 Spectrum ones.

o Rewritten Spectrum ROM Calls for increased BASIC speed.

o ROM extensions - CAT command and full BASIC file handling implemented.

o Hi Resolution display of menus and requesters, with mode promotion.

o Integrated utilities such as built-in Z80 Disassembler and screen to IFF saver.

o High speed - efficient Z80 engine and screen handling.

o ZA80 cross-assembler included for developing Z80 assembler programs.

o Speculator can be used transparently through project icons for snapshot programs.

o LPRINT, LLIST and output to stream #3 is re-directed to any Amiga device or file.

o Spectrum disk handler supplied for using MGT disks just like real Amiga disks.

o Optional built in Easy Keys feature to over come touch key entry problems.

o Various legal example programs including BetaBASIC3, AstroBall and Smashout.

o Cassette load with simple cable for all Amigas.

o Serial transfer to/from Spectrum.

1.5 Hardware and OS requirements

2.00 Hardware and OS requirements

Speculator requires a fast processor with fast RAM to achieve 100% real Spectrum speed. On an A1200 with a 25Mhz MC68030 Viper card + 32Bit Fast RAM, Speculator will run at around the 100% mark (as will "ZXAM" by Antonio J. Pomar Rosselló and Peter McGavin`s "Spectrum"). Speculator will run on any of the 68000 series processors, however, as it will detect processor type and patch any incompatible instructions automatically ("move sr,dn" and "move ccr,dn").

Speculator will work on a non-AGA machine, however, it will mode promote on AGA machines equipped with multiscan monitors for the best results. As default the tooltype HIRES=FALSE is set. For the best display results on AGA machines with multiscan monitors this should be set HIRES=TRUE.

Speculator requires Kickstart 2.0 or greater as it makes use of ASL requesters and Gadtools etc. I didn't try to make it 1.3 compatible since it is likely that anyone with a fast enough processor to make useful use of a Spectrum emulator has also got an up to date version of the operating system. If this is going to be a problem there are software bootable versions of Kickstart 2.0 available e.g. `A500+ emulator`.

1.6 Menu Options

3.00 Menu options

3.10 Project Menu

3.20 Options Menu

3.30 Utilities Menu

3.40 Settings Menu

1.7 Project Menu

3.10 Project Menu

3.11 Load Snapshot

3.12 Save snapshot

3.13 Reset Spectrum

3.14 About

3.15 Help

3.16 Quit

1.8 Load Snapshot

3.11 Load Snapshot

This gives a standard ASL File requester for loading snapshot files (Memory Images). These files can be in any of the five supported types which are detected automatically. However, G.A.Lunter's PC format .Z80 files will need the suffix .Z80 to be recognised correctly. This menu option will NOT load non-snapshot type files (standard Spectrum files like .BAS or .CDE), these must be loaded by typing LOAD "" (on the J key) from BASIC as you would on a normal Spectrum. (see 7.00 Loading Spectrum Files and Chapter 20 Page 139 of the ZX Spectrum BASIC Programming manual). The default drawer, filename and pattern can be set with the tooltypes;

SNP_DRAWER={path}

SNP_FILE={filename}

SNP_PATTERN={pattern}

1.9 Save Snapshot

3.12 Save snapshot

This will save the current state of the Spectrum as a snapshot file in compressed ZX82 format: the entire 48K of Spectrum RAM and all the Z80 registers plus information about the border colour, interrupt mode etc. (See Appendix A. Part 1. for details of the ZX82 file format.) The tooltypes for setting the default drawer, file and pattern from section (3.11) also apply to Save Snapshot.

1.10 Reset Spectrum

3.13 Reset Spectrum

This will reset the Spectrum to the just turned on state. It is equivalent to typing RANDOMIZE USR 0 from BASIC. This will not reset the entire Amiga, just the emulator.

1.11 About

3.14 About

Gives a series of requesters with information about the program such as SHAREWARE registration, Authorship, Accreditations. The MORE button will cycle through the requesters, ENOUGH will exit from them.

1.12 Help

3.15 Help

This will bring up a colourful help screen with information about the positioning of Spectrum BASIC Keywords and their mapping to the Amiga keyboard. (See Section 6.00 for more information about the Keyboard) This can be called up at any time and will pause the emulator. It does not affect the keyboard buffer or the operation of BASIC or Spectrum programs at all. You can exit this screen with a right or left mouse button click or any key press.

1.13 Quit

3.16 Quit

Exits Speculator. Takes you from the 80's back to the 90's.

1.14 Options Menu

3.20 Options Menu

1.15 IntRate

3.21 IntRate

Some Spectrum programs which make heavy use of interrupt code could end up locked in interrupt code if the emulator is running on a slow 680x0. This option allows the interrupt rate to be halved to avoid this problem. Its default is 50Hz and I have not yet encountered a program that needs the interrupt rate slowed on a 1200 with fast memory. However, just in case, I have left this option available.

1.16 IntMode

3.22 IntMode

This allows the user to change the interrupt mode between 1 and 2. This option is included for loading snapshot files which may have the interrupt mode set incorrectly (.KGB or .Mirage). It should normally be left alone. (See Section 4.00 Loading Spectrum files)

WARNING: This option can cause Spectrum programs to crash and Speculator to Reset.

1.17 FlashRate

3.23 FlashRate

Because Flash emulation is a fairly large overhead on a slow 68K and FLASH was a bit of a naff idea in the first place, I have made the Flash rate selectable and even turn-off-able. This should not be an issue on a reasonably speedy Amiga, but it is included for completeness from Speculator's QL incarnation.

1.18 Border

3.24 Border

Some of the supported snapshot file formats did/do not correctly save the colour of the border, so this option allows that to be corrected. (See Section 4.00 Loading Spectrum files)

1.19 RegInfo

3.25 RegInfo

Displays the current contents of all the Z80 registers and the interrupt mode. Useful when hacking, using the disassembler or if having trouble determining the interrupt mode of non-ZX82 format snapshot files as some snapshot formats do not store the interrupt mode correctly.

1.20 Utilities Menu

3.30 Utilities Menu

3.31 Disassemble

3.32 Poke Memory

3.33 Save Screen

1.21 Disassemble

3.31 Disassemble

This brings up the front end requester which interfaces to the built in labelling Z80 disassembler, written by Dave Barker. Disassembles Z80 machine code to an Amiga device.

Address Start address of disassembly.

Z80 PC Makes the start address the current Z80 PC.

Bytes The Number of bytes to disassemble.

Lines The number of lines to disassemble.

Set... Brings up an ASL file requester for disassembler output..

Disassemble Start disassembling to selected file. A busy pointer will be displayed until the diassembly has finished.

File File name to send output to. The default output can be set with tooltypes;

DIS_DRAWER={path}

DIS_FILE={filename}

DIS_PATTERN={pattern}

Exit Exit the requester.

Options

address Show Z80 instruction addresses.

ascii Show ASCII characters.

data bytes Show data bytes.

dfb RST $8 is a 'report an error' routine on the Spectrum, with the next byte containing the error number. RST $28 is the Spectrum's 'floating point calculator' routine. FPC instructions appear in following data bytes until an end-floating-point code $38 is sent.

hex Show all numbers as hexadecimal (base 16).

lower case Show all letters in lower case.

labels 2-Pass Labels. This may cause a slight delay before any there is any output. Labels that were assigned but 'missed' (probably due to some data bytes) are shown with a small 'l'. The maximum number of labels is 32767.

tabs Use tabs and not spaces. This will result in memory being saved. Tabs are of size 8.

1.22 Poke Memory

3.32 Poke Memory

A simple requester provided for poking the Hacks and Cheats supplied in various Spectrum Games magazines. Enter the poke address (16384 to 65535) and the new value to poke, 0 to 255.

1.23 Save Screen

3.33 Save screen

Allows the current Spectrum screen to be saved out as an IFF ILBM file.

1.24 Settings menu

3.40 Settings menu

1.25 Save Icons?

3.41 Save Icons?

Determines whether icons are saved by Speculator. This can be set as a tooltype in the information field of the main Speculator icon or in the project icon of a ZX82 type Snapshot;

CREATEICONS=TRUE|FALSE

Snapshots saved with this option can be run directly from Workbench by double clicking on their project icon just as if they were native Amiga games. You must ensure they have the correct path to the Default Tool (i.e. Speculator). To set the default tool, select the Icon and then select Information from the Icons menu of Workbench.

1.26 HiRes

3.42 HiRes

This setting changes between displaying menus and requesters in 640x256 or 640x512. This is to enable people with multiscan monitors to make optimum use of the screen display. I have been informed that I should have called this option Interlaced, but I find this a bit misleading as it is unlikely that anyone in their right mind would attempt to use this mode on a PAL monitor and a multiscan monitor certainly does not interlace the picture. This can be set as a tooltype in the information field of the main Speculator icon or in the project icon of a ZX82 type Snapshot;

HIRES=TRUE|FALSE

1.27 Tape Load

3.43 TapeLoad

This selects Loading from Tape or Loading from disk; the default is off which means typing LOAD would use disk. However this can be changed by setting tool type;

TAPELOAD=TRUE|FALSE

1.28 Compression

3.45 Compression (of ZX82 files)

You can turn off compression with this option if you have a big hard drive and you want to show off to your friends, however, it is not recommended as it is unlikely to impress them and more likely to make them think you are sad and shallow. This option is included in case you wish to convert a ZX82 files into another emulator format and do not want the hassle of writing a byte run decompression subroutine. (See Appendix A-1 for details about the ZX82 file format and compression technique). The default for this option is on but again it can be set on or off with the tooltype;

COMPRESSION=TRUE|FALSE

1.29 AutoRun

3.45 AutoRun

This switch can prevent BASIC programs from auto running if you need to break into them. For example, if you need to change the LOAD "" command in a BASIC program that expects to be loading from tape.

AUTORUN=TRUE|FALSE

1.30 Volume

3.46 Volume

This enables the volume of the Spectrum sound to be adjusted from very-annoying through the various degrees of annoyance down to completely and satisfyingly off. This can be set as a tooltype in the information field of the main Speculator icon or in the project icon of a ZX82 type Snapshot;

VOLUME=OFF|QUIET|NORMAL|LOUD

The actual levels of these settings can also be adjusted to suite your system if you want with three further tooltypes;

QUIET=1-64

NORMAL=1-64

LOUD=1-64

1.31 Print To

3.47 Print To

Textual output to the Spectrum printer (channel #3) can be redirected to any Amiga device or file with this setting. (Graphics are not currently supported). The Set button will give an ASL File Requester for selecting the device. The default can be set with the Tooltype

PRT_FILE={file or device name}

1.32 Loading and Saving snapshots

4.00 Loading and saving snapshot files

Snapshot files contain a copy of the entire 48k of Spectrum RAM, and the current value of all the Z80 registers, including the state of the interrupts. This means a any program can be saved onto a disk and then reloaded in exactly the same state. These types of files have become the most common way Spectrum programs are distributed and there are several formats used by different emulators, the most common being .Z80 format by G.A.Lunter.

1.33 Loading Snapshot Files

4.10 Loading snapshot Files

Snapshot files can be loaded via the Load item of the project menu. Snapshots in ZX82 format can also be loaded from ZX BASIC with the normal LOAD command. To make it easy to obtain Spectrum software without having to use a conversion utility, Speculator supports many of the snapshot file conventions of other Spectrum emulators on several other machines. All these file specifications are listed in detail in Appendix A. But briefly;

1.34 ZX82 File Format

4.11 .ZX - ZX82 type Amiga Speculator file.

Amiga Speculator, William James, 1988-1995, UK.

The file extension is not needed as an internal identifier "ZX82" is used to recognise this file type.

Full specification

1.35 .Mirage, .SNA, and .snapshot type.

4.12 .Mirage .SNA and .snapshot type.

- Spectrum, Peter McGavin, Amiga, NewZealand.

- JPP, Arnt Gulbrandsen, PC, Norway.

- ZXAM Antonio J. Pomar Rosselló, Amiga, Spain.

This file format is recognised by it's fixed file length of 49179.

NOTE: Only snapshots are supported not 'Tape' files.

Full specification

1.36 .KGB type

4.13 .KGB type

- KGB, Amiga, sorry don't know the author's name.

This file format is recognised by its fixed file length of 49486.

Full specification

1.37 .Z80 type

4.14 .Z80 type 48k & some 128K snapshots (old and new formats).

- G.A.Lunter, PC, Netherlands.

- SPECTATOR, Carlo Delhez, QL, Netherlands.

- Ergon developments emulators, QL, Italy.

- ZX, Andrew Lavrov, QL.

- Mac Spectacle, Guenter Woigk, Germany, 1995.

- ZXAM, Antonio J. Pomar Rosselló, Amiga, Spain.

NOTE: These files MUST have the correct file extension (i.e. '.Z80') as it is not easy for Speculator to automatically detect which type they are. This restriction may be removed in future versions.

Full specification

1.38 .ZX type

4.15 .ZX type

- QL Speculator '93, William James, QL, 1993, UK.

Only snapshots not standard (BASIC) Spectrum files are supported. This file format is recognised by its fixed file length of 49174. This is included mainly for my own benefit, however if you're interested in how an emulator works, my QL Version of Speculator is available including the source code and is in the public domain. I will not be releasing the source code for Amiga Speculator. QL Speculator'93 is available on a QL format disk from me (William James) at the address supplied in section 9. Please send a disk and a stamped addressed envelope (along with any feedback about Amiga Speculator).

Full specification

1.39 .SP type

4.16 .SP type

- Spectrum, Pedro Gimeno, PC, Spain

- ZXAM, Antonio J. Pomar Rosselló, Amiga, Spain.

This file format is recognised by it's internal identifier "SP" and its fixed length of 49190.

Full specification

1.40 Saving Snapshots

4.20 Saving snapshots

Snapshots are only saved out in Speculator ZX82 format. This restriction seems a bit mean in retrospect and future versions of Speculator may save out Snapshots in the other formats as well. You can take a snapshot at any time by selecting the Project/Save option, selecting a file name and clicking OK.

1.41 Converting Snapshots

4.30 Converting snapshots to ZX82 format

Although Speculator will transparently load several file formats listed in section 4.1 you may wish to convert snapshots into ZX82 format to save disk space, set the border and interrupt modes correctly and to allow games to be started directly from Workbench with their icons. This can be achieved by multiple menu selection. With the right mouse button held select the project menu and then use the left mouse button to select LOAD and then SAVE before releasing the right mouse button. The "Load snapshot" file requester will arrive, select the snapshot file to be converted and hit OK. The "Save snapshot" file requester will now arrive immediately without any Z80 emulation occurring and the new file name can be selected. The saved file will be exactly the same as the old file except that it is in ZX82 format and now has an icon (if SaveIcons is enabled). A similar process can be used to set a new border colour or to change an incorrect interrupt mode. i.e. multiple menu select 'Project/Load', 'Options/Border', 'Project/Save' or 'Project/Load', 'Options/Interrupt mode', 'Project/Save'. NOTE: Do not try to change the interrupt mode unless you are sure it was not set correctly in the old snapshot file as you will probably crash as soon as it starts to execute.

1.42 Snapshot error alerts

4.40 Snapshot error alerts

There are several Error/Warning alerts associated with Snapshot loading:

1.43 Not a Snapshot file

4.41 Error: Not a Snapshot file.

This will occur if you attempt to load a non-Snapshot ZX82 file (BAS, CDE, DAT, DT$) from the Load Snapshot menu item. You must load these sort of files from ZX BASIC by typing LOAD "" to get an ASL file requester or by typing LOAD "filename". (See Section 7.00 for more information.)

1.44 Not a recognised file type

4.42 Error: Not a recognised file type.

Either this is not a Spectrum file at all or it is an unrecognised Spectrum emulator file format. If you find a Spectrum emulator file format that you would like future versions of Speculator to support, please send me details and some example files on a disk.

1.45 Attempting to load a 128K snapshot

4.43 Warning: Attempting to load 128K snapshot.

This is displayed when loading 128K .Z80 snapshots. This is because Speculator does not currently emulate a 128K Spectrum. However, a lot of 128K games do not in fact make use of the extra memory, just the sound chip and will run on Speculator. Speculator will attempt to load the file but cannot garantee it will not crash the emulator. If it does load successfully it is recommended that a new snapshot is taken in ZX82 format which will avoid the requester being displayed in future, save on disk space and allow the snapshot to be run directly from Workbench by double clicking its icon.

1.46 Loading from Workbench

4.5 Loading snapshots directly from the Workbench

Snapshot files in ZX82 format that have an icon can be used just like any native Amiga program. Double clicking the snapshot`s icon will run Speculator, and then load and execute the snapshot. You must ensure that the icon has its default tool set to "Speculator" and has the correct path to find it. You can also set the various tool types in the projects which will over-ride any tooltypes set in the main Speculator icon.

1.47 Setting initial file path

4.6 Setting the initial file path for snapshots

The default drawer, filename and match pattern for the load and save snapshot requesters can be set up with the following three tooltypes;

SNP_FILE={Filename}

SNP_DRAWER={Drawer}

SNP_PATTERN={Pattern}

For a full list of Speculator's tooltypes, see Appendix B.

1.48 Changing the default Icon

4.7 Changing the default icon saved with snapshots

Snapshots saved out from Speculator in ZX82 format can be run by simply double clicking their icons. Icons will be saved out with the snapshot file if the menu item "Settings/Create icons?" is turned on. The icon to be used should be stored as;

system:prefs/Env-Archive/Speculator/def_ZX82

I have supplied an workbench 2/3 compliant icon which can be used. However, I am no artist and you may wish to design another. You can set tooltypes in the project icons which will override the tooltypes set in the main Speculator icon. This means each project (snapshot) can have its own set of preferences if you wish. Speculator does not automatically save the current preferences out as tooltypes in this version.

1.49 Tape Loading

5.00 Tape Loading

I have not had a great deal of luck with tape loading code for various reasons. Currently, I have most success loading tapes that have saved out from a real Spectrum using the same tape recorder. Also most of the later (and better) games for the Spectrum used some form of hyper-load. As Speculator is not a real-time emulator there is not much hope of loading these types of files from cassette.

The tape loader code uses a CIA timer to provide exact timings of the incoming tape pulses and works as well on a slow machine as it does on a fast one. It has been tested on everything from a standard A500+ to an A4000 with a Cyberstorm'060 accelerator. The CTS of the serial port is used to read the in coming tape signal. This is designed to take from -12v up to +12v and so should be quite safe to use with cassette recorders. Please note, I do not accept any responsibility for damage to your serial ports. You use this feature at your own risk. Please be careful when making up a lead that you wire up the correct pins. If you are not a competent person (if you tend to break things), get someone who is to do it for you. You will need a 25 way female D-type socket, a 3.5 mm miniature jack plug and a length of screened audio cable. Connect the screen to GND (pin 7) and the signal to CTS (pin 5).

The operating system is switched off during tape loading. I am not happy at all about doing this, but it is the only way I could get the routine to work as timing is very critical. As a result, if you want to quit while loading you must use the left mouse button

as 'BREAK'. This is because the keyboard will not work throughout the loading process. Another consequence of turning the operating system off is that when it gets turned back on I get a series of unwanted key presses. I haven't been able to solve this problem to date.

Getting the volume level correct is very critical - try to get the colour bands in the border area about the same width if you can. It may take two or three attempts. Any more than that and I would give up.

Sad plea for help: If anyone out there has already written tape loading code that is better than mine, I would be grateful if they could get in touch with me about including it with the next version of Speculator.

1.50 Keyboard Emulation

6.00 The keyboard emulation

I have put a lot of effort into the Keyboard handling on Speculator. Firstly, I have made sure that all the symbols on the Amiga keyboard function as they should using the standard Amiga shift keys. In addition, using the Ctrl, L-Alt, L-Amiga and R-Alt keys as the Spectrum's various shift keys, the original Spectrum's functionality has been retained.

A Help screen with a map of the Amiga keyboard showing the positions of the Spectrum keywords and the additional function keys is displayed while the Help key is pressed (or from the help menu item).

The Amiga shift and function keys have been assigned as follows;

Ctrl Extended Mode Symbol Shift & Caps Shift together

L-Shift Standard Amiga Shift

L-Alt CAPS SHIFT

L-Amiga Extended Mode Exactly the same as Ctrl

R-Amiga As on standard Amiga Menu select or pointer control

R-Alt SYMBOL SHIFT

R-Shift Standard Amiga Shift

F1 EDIT Equivalent to L_Alt & 1

F2 CAPS SHIFT Equivalent to L_Alt & 2

F3 TRUE VIDEO Equivalent to L_Alt & 3

F4 INVERSE VIDEO Equivalent to L_Alt & 4

F5 GRAPHICS Equivalent to L_Alt & 9

F10 NMI Causes a Non-maskable interrupt

Left Arrow Left Equivalent to L_Alt & 5

Down Arrow Down Equivalent to L_Alt & 6

Up Arrow Up Equivalent to L_Alt & 7

Right Arrow Right Equivalent to L_Alt & 8

Help Give help screen

When programming note that <= <> and => are keywords and should be typed with combination keys (R-Alt Q, W and E) rather than as two separate symbols.

The Spectrum keyword entry system means that some key presses are interpreted as BASIC commands and functions, depending on context. Two common examples follow; you can look up more on the Help Screen. Don't be afraid to experiment - it takes three days of puzzlement to learn the key positions, then you should be able to type ZX BASIC as fast as you can type programs in full - faster, if you avoid obscure combinations! Press Ctrl (equivalent to both SHIFT keys on a Spectrum, or EXTEND on a Spectrum Plus) to select "E" cursor mode, then R-Alt & "9" for CAT; add a device name in quotes, like "DH0:" to see the catalogue of ZX files on the device. Press "J" for LOAD, then a file name in double quotes to load the chosen file.

The EASYKEYS facility has been included for people who have trouble with the Spectrums obscure touch key entry system (See section 8.30).

Beta BASIC 3 has improved keyboard input and editing which lets you mix keywords and spelt-out commands. Just type a space to move from keyword mode if you prefer to spell out commands in full (See section 8.50).

6.10 A1200 Keyboard bug

1.51 A1200 Keyboard bug

6.10 A1200 keyboard bug

With the release of the A1200 a new and completely unnecessary incompatibility was introduced into the Amiga Keyboard controller. This bug is not on the A4000 or any earlier models of the Amiga. The A1200 keyboard can't deal with multiple key presses, which has resulted in problems with may games and applications such as MED. This has also made it IMPOSSIBLE to emulate the Spectrum keyboard accurately. Several Spectrum programs that use the keyboard are not emulateable on the A1200 (for example QuickSilver's "Ant Attack"). I will make sure Amiga Technologies UK are aware of the problem at the next developer conference, and hopefully it will be fixed on future A1200's. Until then... it's not my fault. If you have problems try using a joystick if the program has a Kempston joystick option. Alternatively, if the program allows you to redefine the keys, try using weird combinations of the shift keys. The ultimate solution is to replace your 1200 keyboard or upgrade to an A4000 if you have lots of spare money.

1.52 Standard Spectrum file handling

7.00 Standard Spectrum file handling

Unlike a lot of other Spectrum emulators, Speculator has quite extensive support for standard Spectrum file handling in addition to snapshots. These are the normal file types used by Spectrum BASIC from a tape. There are four different types;

0: BASIC

1: Data array

2: String array

3: Code (or screen)

The ZX ROM has been patched in several places enable Speculator to use any Amiga storage device for loading or saving, as well as cassette tapes for loading.

First the file handling commands are intercepted (at $0634) during BASIC parsing allowing file names of up to 32 characters to be used or if no file name is supplied an ASL requester is used for file name selection.

The load and save data block routines are intercepted in the ROM (at $0562 and $0970 respectively) which implement disk handling using the file name set during BASIC parsing.

For programs other than BASIC which may call the ROM routines directly without the file names being picked up during parsing there is a further patch (at $0984) in the save data block code which will give an ASL requester. If no name has been picked up during passing for a ROM call to Load data block, an ASL requester is also used.

Lastly, the error messaging has also been patched to allow AmigaDOS error messages to be returned.

For more details on how the ROM patching works see section 10.

Normal tape loading is also implemented and can be switched on or off from the preferences menu (see section 5 for tape loading).

7.10 Standard Spectrum files from disk

7.20 The CAT command

7.30 File paths

7.40 Obtaining files standard ZX82 file format

7.50 Loading non-ZX82 files from ZX BASIC

1.53 Standard Spectrum files from disk

7.10 Standard Spectrum files from disk

The emulator will read and write files in ZX82 format by name from any AmigaDOS directory or device (e.g. RAM:, DH0:, PC0:, SP0: etc.).

Speculator patches the ZX ROM to give standard AmigaDOS Error Messages.

Speculator extends the 48K ZX Spectrum by implementing CAT "" and allowing LOAD and SAVE on Amiga directory devices. You currently need to return to Workbench or Shell to rename files, format devices, or create directories.

If a file name is already in use the old file is deleted and the newly saved one replaces it. The ZX file type and details are held in the head of the ZX82 file (See Appendix A section 1 for full description).

1.54 The CAT command

7.20 The CAT command

The CAT command has been implemented to enable viewing of header information such as file type, compression, file length etc. It has a different format to the Interface 1 implementation which was;

CAT {drive number 1 - 8}

To see all the ZX82 Spectrum files in an Amiga directory, use CAT instead of DIR., with the Amiga device name in quotes (or a string variable, or expression):

CAT "DH0:" (Left Amiga Key then R-Alt & 9 to get the CAT keyword)

CAT "" will use the current BASIC directory set by the last BASIC LOAD or SAVE. CAT "DF0:" will set the BASIC directory to DF0: and give a catalogue of its contents. This can be very slow from floppy disk as each file must be opened and its header examined before the any printing occurs. CAT "ZX:programs" would set the current BASIC directory and display it's contents. CAT cannot be redirected to the printer as it can with interface 1. If the selected directory does not exist an extended error message will occur;

S: Object not found , 0:1

If the volume is not found a DOS requester will appear;

Please insert volume ... in any drive

If it is cancelled you get this extended error message;

S: Device (or volume) is not mounted ,0:1

Example output;

ZX:BASIC/ZX82/

Atic BAS C 182 10

Atic1 CDE U 6912 16384

Atic2 CDE U 16221 1

Calander BAS U 16221 1

EasyKeys BAS U 108 100

EasySwitch3 BAS U 665 22

EasyCode CDE U 301 31000

Scrabble1 CDE U 6912 16384

This displays the name (truncated to 14 characters), type (e.g. CDE for CODE, BAS for BASIC, SNP for SNAPSHOT (MGT 48K memory image), DTA or DT$ for DATA ARRAYS), compression, size in bytes and "start": an address for CODE and an optional line number for ZX BASIC files.

The size field is always the size of the uncompressed file, a 'C' or 'U' is used to indicate whether compression has been used. To get the actual size of a compressed file you will have to use LIST or Dir from a Shell or icons/information from the workbench menu.

CAT will only display the names of files in ZX82 format. If you need to see all files in a directory you will need to go to Workbench or a Shell and use List or Dir.

7.21 File types recognized by CAT

1.55 File types recognized by CAT

7.21 File types recognized by CAT

The directory type is equivalent to the MGT directory type decremented by one as follows;

BAS Type 0 - BASIC file

DTA Type 1 - Data Array

DT$ Type 2 - String Array

CDE Type 3 - CODE file

SNP Type 4 - 48k SnapShot file

MDV Type 5 - microdrive file

SCR Type 6 - screen

SPC Type 7 - special code

128 Type 8 - 128k Snapshot

FILE Type 9 - file

EXE Type 10 - execute code

UNI Type 11 - uni directory

CRC Type 12 - Create code

Type 13 - Unknown

Type 14 - Unknown

SBAS Type 15 - sam_basic

SDTA Type 16 - sam_num_array

SDT$ Type 17 - sam_char_array

SCDE Type 18 - sam_code

SSCR Type 19 - sam_screen

SDIR Type 20 - sam_directory

Type 21 - 255 are unknown

1.56 File paths

7.30 File paths

The initial file name and path for LOAD, SAVE and CAT can be set in the .info file with the Tooltypes;

STD_FILE={file name}

STD_DRAWER={file path}

STD_PATTERN={match pattern}

The directory is changed by selecting a new file path with LOAD or SAVE, so, for example, these successive commands both use RAM:, and leave it selected as the default.

LOAD "RAM:AticPic" SCREEN$ (Ctrl R-Alt K gives SCREEN$)

LOAD "AticCode" CODE (Ctrl I gives CODE)

Use the normal Spectrum keys to save (press S in K mode) or load (press J in `K` mode), giving a standard Amiga file path in quotes.

If no filename is specified then an ASL file requester is used to select a file or directory.

e.g.

LOAD "" SCREEN$

or

SAVE "" LINE 10

From this requester you can select a Volume, Directory and Filename as usual.

The volume and directory path will become current and any further file operations will relative to this.

This is very helpful if you are loading a Spectrum game for example which usually have a short BASIC loader program that LOADs other files. You may want to modify the BASIC loader program to use file names, but you needn't specify the full directory path for all of them.

If you do not modify the original BASIC loader program which often didn't specify particular filenames (as tape loading is sequential) you may find you will get an ASL requester asking for each file name in turn.

NOTE:

This File requester is not the same as the LOAD or SAVE a SNAPSHOT file requester from the project menu. This is for LOADing and SAVEing standard Spectrum files, although you can also load ZX82 format snapshots from this requester if you want.

1.57 Obtaining files standard ZX82 file format

7.40 Obtaining files standard ZX82 file format

Standard BASIC type files are not as commonly available as snapshot files. "Spectrum" by Peter McGavin uses a different format where the header and the data blocks are saved as separate files. There is currently no utility for converting this type of file into ZX82 format. If you wish to transfer BASIC type files from McGavin`s emulator you could Snapshot them, load them into Speculator and then re-save them out as standard files from BASIC. The best way of getting these files is to use the supplied Spectrum handler to read the files directly from an MGT format disk. You can also load files from cassette and then re-save them from BASIC, but this requires a little knowledge of the Spectrum.

1.58 Loading non-ZX82 files from ZX BASIC

7.50 Loading non-ZX82 files from ZX BASIC

Speculator will load any files that aren't in ZX82 format as if they were CODE type files as long as the load address is specified. Files of longer than the available Z80 memory will be truncated.

for example;

CLEAR 32767

LOAD "README.TXT" CODE 32768

This would load the README text file supplied with Speculator into Z80 memory as if it were a code file. This feature can also be useful for programmers in conjunction with the ZA80 cross assembler supplied with Speculator. ZA80 produces Z80 binary files in a raw format (with no Spectrum header information) but they can be loaded and executed (or saved out again in ZX82 format) with this feature. See section 8.10 for more details on ZA80.

1.59 Supplied programs, drivers, utilities, demos

8.00 Supplied programs, drivers, utilities, demos

@{ " 8.10 ZA80 Assembler " LINK 8.10}

@{ " 8.20 Disassembler " LINK 8.20}

@{ " 8.30 EasyKeys " LINK 8.30}

@{ " 8.40 MGT Disk Handler " LINK 8.40}

@{ " 8.50 BetaBASIC " LINK 8.50}

@{ " 8.60 Demo Games " LINK 8.60}

1.60 ZA80 assembler

8.10 ZA80 assembler V1.14 D.J.H.Hartley, 1989,1990.

This is an excellent Amiga Z80 cross assembler which will produce raw binary format files. These files can be loaded into Speculator from BASIC as if they are CODE type files as long as the start address is supplied. For full information about ZA80 see the text file ZA80.doc and example source files, some of which show how a ZX82 header can be generated automatically.

1.61 Z80 Disassembler

8.20 Z80 Disassembler V1.00 Dave Barker 1993

This labelling Z80 disassembler was originally part of a PD QL program called QSpec2. Dave has kindly allowed me to include the disassembler directly into Speculator with a new Amiga style front end. The Disassembler requester is fully explained in Section 3.31. Used in conjunction with ZA80 it becomes quite a powerful Z80 software development system. This may be extended into a full Z80 monitor on future versions of Speculator.

1.62 EasyKeys

8.30 EasyKeys - based on a program by Glen Kendall

EASY KEYS is a program that allows ZX keywords to be typed in full. It works well as long as you remember to use CAPITALS for keywords, a space after the initial line number and include a space in GO TO and GO SUB. The program, elegantly written by Glyn Kendall, runs in Z80 code, intercepting the ROM syntax checker. This program has now been integrated into Speculator. It has been modified and reassembled with ZA80 so that it now resides in the spare space in ZX ROM at address 14446. Easykeys can be activated by setting the tooltype EASYKEYS=TRUE in Speculator's icon. This will make Speculator patch Easykeys into the ROM and run it on start-up allowing keywords to be typed in full. If the Easykeys tooltype is omitted or if it is set EASYKEYS=FALSE the ROM patch will not be made. I recommend that EASYKEYS is not patched unless you intend to do some BASIC programming as it is possible it may cause incompatibility with games that use interrupt mode 2 and use the ROM as an interrupt vector table.

NOTE: With Easykeys all keywords must be types in capitals, and it is important to put spaces in "GO TO" and "GO SUB" and after initial line numbers. It also has trouble if you don't put spaces after keywords like "INT " or "RND ". When active EasyKeys will use a flashing chevron instead of a "K" as a cursor.

1.63 MGT disk handler

8.40 MGT disk handler SPFileSystem V2.0 Frank Swift 1995

This disk handler was written by Frank Swift and it allows MGT format disks to be used directly on the Amiga. Disks can be read written and even formatted. It has been adapted to use Speculator's ZX82 file format, however, it doesn't currently support byte-run compression. If you have problems with hard disk space then you may wish to compress your files by loading them into Speculator from the MGT disk and then saving them out again onto your hard drive with compression turned on.

The handler is installed from Workbench as follows;

a) Open the drawer "Speculator/MGT-Handler/L" and drag the program "SPFileSystem" to your system "L:" drawer.

b) Then drag the icons SP0: and SP1: from "Speculator/MGT-Handler/Devs/DOSDrivers" to your system "Storage/DOSDrivers" drawer.

The handler is activated by double clicking the SP0: or SP1: icon. If a MGT format disk is inserted now a disk icon called "Disciple" will arrive on the Workbench after a short delay.

If you want the handlers to be activated automatically on start-up then they should be put in your system "Devs:DOSDrivers" draw, however this is not recommended unless you intend to use Disciple disks regularly as you may get contention with other DOS Drivers in the system slowing down disk access, and it can also clutter your Workbench screen unnecessarily.

NOTE: Early versions of the MGT Disciple have a snapshot bug which affects some programs - they fail to save the value of the alternate F register.

1.64 BetaBASIC3

8.50 BetaBASIC3 by Dr. Andy Wright, BETASOFT.

See the file BetaNotes.doc for a brief summary of this program.

1.65 Demo Games

8.60 Demo Games

It is planned to compile a full disk of ZX PD for Speculator, any offerings are very welcome.

8.61 Demo Snapshots

8.62 BASCI Programs

1.66 Demo Snapshots

8.61 Demo Snapshots

SNAIL QUEST is one of over 100 Spectrum adventure games available from the Guild of Adventure Software, 760 Tyburn Road, Erdington, Birmingham B24 9NX, UK. Most cost under £1 each on tape! Control the game by typing simple sentences, such as GO NORTH, TIE SHEETS TO BED or HELP! Type J "SNAILQUEST" in ZX BASIC to load the 48K snapshot.

ASTROBALL is a 48K demo written by Balor Knight. Bounce the ball between the platforms to collect the coins and reach the top of the screen. Thanks to Revelation Software of PO Box 114, Exeter, Devon EX4 1YY for this demo. Press 1 to select key control with Q up, A down, O left and P right. Press 2 for Kempston joystick control, emulated by the arrow keys or joystick. Press SPACE to start after examining the map. If you win you are rewarded with a small game of Space Invaders!

SMASHOUT is a Breakout type game, with some rather nice sound effects. This game was written by Chris Pile and is copyright of Digital Image 1985. It is included with permission of the author. Use keys Z for left, X for right, H for halt and P for fire.

UltimateWarrior an impressive 3-D Knightlore type game Copyright 1988 Mark Howlett.

1.67 BASIC programs

8.62 BASIC programs

Several simple games in compiled BASIC are also included as they are much shorter than full 48K game snapshots, come with source and can be tweaked to run at reasonable speed on slow hardware. HiSoft BASIC is available on MGT disk from Hisoft, The Old School, Greenfield, Bedford MK45 5DE, UK. ZIP comes from Betasoft, 24 Wyche Avenue, Kings Heath, Birmingham B14 6LQ, UK.

RUNAWAY is a ZX BASIC game designed by Simon N Goodwin, first published in the launch issue of Games Computing. RUNAWAY C is an accelerated version, compiled with Hisoft's ZX BASIC compiler. Use cursor arrows or Q, A, O and P to steer the Runaway Robot around the computer-generated mazes. It takes practice as the robot goes haywire whenever it hits a wall. One or two people may play.

STAR BASE 5 is a demonstration of the ZIP compiler by the same author. Again corresponding compiled code is in STAR BASE 5C. Press 6 or 7 to aim the gun, and 0 to fire.

SpeedTest is a simple BASIC benchmark program for comparing the performance of Speculator with the real thing or other emulators. You should be careful when comparing different emulators as some of them (not Speculator) have inaccurate interrupt rates which can give misleading results. For example on my machine ZXAM (the full version) reports about 103% the speed of a real Spectrum when in fact it is only about 79% when timed with a stopwatch. For ease of comparison I have supplied this program in .Z80 format which will load into most emulators. This program was originally by Ergon Development for their QL range of emulators. I have modified some its textual content to reflect the Amiga timings rather than the QL.

1.68 Disclaimer, Shareware registration and distribution

9.00 Disclaimer, Shareware registration and distribution

1.69 Disclaimer

9.10 Disclaimer

No warranties are made. All use is at your own risk. No liability or responsibility is assumed. Speculator is not licensed or recommended for controlling life support devices or systems intended for surgical implant into the body or intended to support or sustain life... even plant life.

1.70 Shareware registration

9.20 Shareware registration

Speculator is shareware which, in this case, means you can use it freely for an evaluation period of one month. After this period, if you wish to continue using Speculator, you should send £10 (or the equivalent in your currency) to the address given below. In return you will receive updates of Speculator as and when they become available until I stop developing it. By making this contribution, you assure the further development of Speculator (see section 11.00) and assure yourself easy passage into the after life with a clear conscience. Pass on copies of Speculator to anyone who wants it but please make sure they get everything included in the archive including this manual. I would also welcome any feedback, comments, ideas, criticism, bug reports, or job offers (Amiga software or hardware development).

My address is;

William James

54 Victor Road

Newtown

Colchester

Essex

CO1 2LX

England

or e-mail me at;

w.james@philinet.demon.co.uk

(Thanks to Phil Batts for lending me this e-mail address.)

1.71 Distibution

9.30 Distribution

1.72 Third Party Copyrighted Material

9.40 Third Party Copyrighted Material

Some of the files supplied with Speculator are third party PD. software and have different restrictions and copyright details.

The Sinclair ROM image is copyright and used by permission of Amstrad, posted by Cliff Lawson (Amstrad 75300,1517) on Compuserve on 19th March 1993:

"I have given Amstrad's permission to the authors of Spectrum emulators to use our code because it is no longer in commercial competition to our own Spectrum product and I am kind of keen for the memory of that ground breaking machine to continue."

ZA80 - Z80 cross assembler V1.14 Copyright (c) D.J.H.Hartley, 1989,1990. This program is distributable in accordance with the details given in the supplied file ZA80.txt.

Z80 Disassembler V1.00 by Dave Barker Although this has been incorporated into Speculator the original program QSpec2 is PD and available with source code with Speculator'93.

MGT disk handler (SPFileSystem) - by Frank Swift V2.05.08 (c) July 1995 This program is PD and is freely distributable.

The various demo Spectrum files have been supplied with permission from their respective authors are listed along with their copyrights in section 8.

1.73 How the emulator works

10.00 How the emulator works

The whole idea behind Speculator was to do all this in an operating system friendly way and to incorporate the Spectrum as naturally as possible into the Amiga environment as outlined by the "User interface style guide". I wanted to retain accurate Spectrum emulation but also the option of enhanced use-ability, through menus, file requesters, clear screen display, and proper keyboard handling. There are still people out there who like to program in Spectrum BASIC. There are also a lot of people who still use the Spectrum to play Adventure games. Hopefully, Speculator will give these games a new lease of life.

Speculator can be considered as emulating two almost separate things, first the Z80 processor and secondly the ZX Spectrum hardware so I have split this discussion into two halves.

10.10 The Z80 Emulator - Warning: a bit technical

10.20 The Spectrum hardware/software emulation

1.74 The Z80 Emulator

10.10 The Z80 emulator (Warning: a bit technical)

The Z80 emulator is a software model of a real Z80. It works by simulating each of the Z80's 1000+ instructions with a separate routine in 68000. So for every Z80 instruction there are several 68000 ones and speed of execution becomes a real problem. Add to this issues like Motorola and Zilog using different endian formats for storing numbers, (Z80 = Lo-Hi and 68K = Hi-Lo) and things start to get slower. The 68K is also fussy about reading and writing words on odd boundaries because of 16/32 bit memory where as Z80's, being 8-bit machines, don't mind at all. These factors cause the emulator routines to look far from optimal at a first glance.

For example the Z80 instruction call NN (CDNN) translates to something like;

* call nn

Z8CD moveq.l #0,d0

move.b (a6)+,d0 ; Get the low byte of the address

lsl.w #8,d0 ; (I wish there was a swap.b instruction)

move.b (a6)+,d0 ; Get the high byte of the address

rol.w #8,d0

suba.l a4,a6 ; Work out current Z80 PC

move.w a6,WrkSpc1(a2) ; Put it onto the Z80 stack

move.b WrkSpc1(a2),-(a3) ; byte...

move.b WrkSpc2(a2),-(a3) ; by byte (yawn...)

lea.l (a4,d0.l),a6 ; jump to the address

DoNextInstruction ; A macro for the fetch execute loop

Where; d0 is a scratch register a2 is a pointer to the Z80 variables and registers in memory a3 is the Z80 stack pointer (absolute) a4 is a pointer to the Z80 64k memory block a6 is the Z80 PC (absolute).

This is just intended for illustration of some of the ideas for people who are used to 68K assembler.

The routines for each Z80 instruction are arranged in a 64k block of memory spaced every 256 bytes from 80-FF, 00-7F. The routines address can then be calculated from the instruction byte quite quickly relative to a pointer to the centre (00 = NOP). The NOP instruction is effectively a fetch/execute loop as it does nothing apart from read the next instruction byte, increment the Z80 PC and then jump to the correct instruction routine.

* nop - used as Fetch/Execute loop

Z800 move.b (a6)+,d0 ; Fetch next Z80 code byte and increment Z80 PC

lsl.w #8,d0 ; multiply by 256

jmp (a1,d0.w) ; jump to (execute) the routine relative to 'Z800'

where; d0=scratch, a1=Z800, a6=Z80PC

The prefixed instructions ED,DD,FD,CB are also interlaced into this 64k block to avoid having five separate 64k blocks.

* ed prefix

Z8ED move.b (a6)+,d0 ; Get next Z80 instruction byte and increment Z80 PC lsl.w #8,d0 ; multiply by 256

move.b #EDops,d0 ; Add an offset to ED routine

jmp (a1,d0.w) ; jump to routine relative to 'Z800' + ED offset

where; d0=scratch, a1=Z800, a6=Z80PC

The most crucial part of the emulator is this fetch/execute loop, which will be called between each and every Z80 instruction, thousands of times a second. Any inefficiency here can have dramatic effects on the speed of emulation. To increase the speed of the emulator I have tried to avoided using such a loop as each jump back to the loop has a branch penalty on the 68K. I use threaded code, where possible. This means each instruction fetches the next reducing the branch penalty overhead. Most routines use three instructions to fetch the next instruction directly;

.. routine ...

move.b (a6)+,d0 ; Fetch the next instruction and increment the PC

lsl.w #8,d0 ; Execute next instruction directly by jumping

jmp (a1,d0.w) ; relative to the middle of the 64k code block

where; d0=scratch, a1=pointer to the middle of the emulator, a6=Z80PC

However, if all the instructions did this it would be very difficult to intercept the execution every 50th of a second in order to do an interrupt. For this reason, all 'program control' instructions (jmp, jr, ret, call etc.) use the 'NOP' despatch. These instructions will always occur more than 50 times a second and so it is guaranteed an interrupt can occur.

The vertical blanking interrupt server is used to self modify the first instruction of the "NOP" routine. This causes the emulator to jump directly to Z80 interrupt handling code rather than the next instruction. I don't like using self-modifying code, but it does improve speed dramatically in this case. I make sure the address of the modification gets flushed from the instruction cache.

Apart from the above, the rest of the emulator works by careful coding of routines to do exactly what the Z80 did. It can be very difficult to track down an erroneous routine. Flags are emulated by having lookup tables which convert 6800x0 flags into the Z80 flags. The block move and copy instructions (ldir, ddir, cpir, cpdr) work differently to the Z80 in that they do not allow interrupts between each loop.

10.11 Unimplemented Z80 Instructions

10.12 The Z80 H Flag And the DAA instruction

10.13 Undocumented Z80 instructions

10.14 Real time execution

1.75 Unimplemented Z80 instructions

10.11 Unimplemented Z80 instructions

The Z80 instructions IND, INI, OUTD, OUTI are not implemented as they are not useful on standard Spectrum hardware, so the emulator reports 'not implemented'. These instructions should never appear in real Spectrum programs - if you find one that uses them, please tell!

1.76 The Z80 H Flag And the DAA instruction

10.12 The Z80 H Flag And the DAA instruction

The half carry flag of the Z80 is set or reset depending on the carry and borrow status between bits 3 and 4 of an 8-bit arithmetic operation. This flag is used by the decimal adjust accumulator instruction DAA to correct the result of a packed BCD add or subtract operation. This flag is very hard to emulate on a 680x0 and so I have not tried to implement it. This caused problems with implementing the DAA instruction. This has been solved by storing the value of the accumulator before any operations that would have affected the h flag in a meaningful way. When a DAA instruction is encountered the last arithmetic operation can be undone and then redone with the 680x0 BCD instructions ABCD and SBCD. A bit of a kludge but it works acceptably.

1.77 Undocumented Z80 instructions

10.13 Undocumented Z80 instructions

The Spectrum Z80A processor is capable of executing some 'undocumented' machine-code instructions. A few programmers use these to confuse hackers or manipulate registers in non-standard ways. Over 150 such instructions are now recognised by Speculator. All other unused instructions will cause a 'Nonsense Op code' requester. This requester will tell you the current value of the Z80 program counter, warn you that the Z80 has probably crashed, and offer you the chance to do a Z80 reset or to continue with emulation.

1.78 Real time execution

10.14 Real time execution

This current version the Z80 emulator does not attempt to emulate in real time. That is, it is running as fast as it can instead of trying to make each instruction take the same amount of time (T-states) as they would on a real Z80. This is because until recently is has always been a problem trying to get up to 100% real speed of a 3.5MHz Z80. Now that faster 680x0 processors are becoming available the problem has become trying to slow the emulator down. This is not as straightforward as it might sound. For example, if the Z80 was made to wait for a while every 50th of a second to keep its average speed accurate, sound and tape loading would become distorted and inaccurate. However, I do have a solution and I hope to implement it in a future release of Speculator. This should make loading of hyperload cassette tapes a real possibility, as well as improving sound emulation and game playability.

1.79 The Spectrum hardware/software emulation

10.20 The Spectrum hardware/software emulation

The screen is emulated by checking at the end of every Z80 instruction that does a write to memory. If the write was to the Spectrum screen area the Amiga screen is updated immediately. For screen writes this is 4 Amiga screen writes, one for each colour (including bright). For an attribute this is 4*8 writes to the Amiga Screen for the whole attribute square. This is potentially much slower than on a real Spectrum but the screen code has been highly optimised using a separate routine for each possible value written, and a double buffer to prevent updating when nothing has changed. It has an advantage over some other methods in that it only updates the screen when it needs to be updated rather than 50 times a second.

IO instructions (in and out) are also intercepted and emulated including low level keyboard access, border and beeper and the Kempston joystick port. Speculator keeps a model of the Spectrum key matrix in memory. This is updated every time a raw key message is received at the IDCMP port. When a read of IO port $FE is made the current state of each of the key rows are ANDed together depending on the bits set in the high byte of the address. The Rawkey values have been carefully remapped so that most of the Amiga keys do something meaningful, for example the arrow keys simulate shifted 5,6,7 & 8 of the Spectrum etc. Even keyboard phantoming has been emulated for multiple keyboard presses - but that was all a waste of time due to the keyboard bug on the A1200 (See section 5).

At the higher level, the interrupt 1 keyboard handling routine of the Spectrum ROM has been rewritten in pure 68000 and gives a much higher degree of compatibility. Here all the normal operation of the "Qwerty" keyboard has been retained, e.g. if you type from BASIC <shift & 2> you get quotes <">. You can also get quotes as you would on a real Spectrum by pressing <Symbol Shift & P> (where R-Alt. is Symbol Shift).

The sound is emulated at a low level by toggling the audio channel 0 between $8080 and $7F7F, after disabling its interrupt bit and setting its output frequency to the maximum. This works fairly well, however, the sound can be a little distorted because of the operating system multitasking. The ROM routine for Beep is intercepted and uses the audio device to achieve much better results.

10.21 Patches to the ZX ROM

10.22 In Port $FF

10.23 A few incompatabilities

1.80 Patches to the ZX ROM

10.21 Patches to the ZX ROM

A few genuinely-unused ED prefixed Z80 op codes codes are used to intercepted Z80 execution and call fast 68000 routines. These op codes are patched into the ZX ROM in several very carefully chosen places that could not affect normal Spectrum operation. Only two byte instructions have been patched so that any calls to ROM will not occur half way through the patch. The addresses of the patches have also been checked to make sure they could not have possibly been used by any program as interrupt mode 2 vectors offsets.

A list of all non-standard Z80 op codes, where they go and what they do;

$ED00 at $15F4 Intercepts output to the ZX printer and sends it to an Amiga device.

$ED01 at $0E44 Speeds up the CLS routine with optimised 68000 code.

$ED02 at $0E00 Speeds up the scroll routine with optimised 68000 code.

$ED03 at $0BB7 Speeds up printing of all characters

$ED04 at $0984 Intercepts ROM calls to the "SaveBlock" routine and gives an ASL requester.

$ED05 at $0562 Intercepts ROM calls to "loadblock" from BASIC and direct calls

$ED06 at $11DA Speeds up RAMCheck - yes well...

$ED07 at $0970 Intercepts save routine if called from BASIC.

$ED08 at $1346 Expands error messaging routine to deal with AmigaDOS IO errors.

$ED09 at $0634 Gets a file name during BASIC parsing of LOAD, SAVE and CAT. Allows file names of more than 10 characters or uses an ASL requester if no name is supplied.

$ED0A at $xxxx Allows 68000 to call Z80 routines and return safely (smart).

$ED0B at $03B5 Replaces Beep routine using Amiga Audio device instead.

A few other minor changes are made to the ROM;

$0B at $1B14 Change command class of 'CAT'

$17 at $0D2C Correct scroll bug

$19 at $0A33 Correct cursor bug

$C9 at $257D Correct SCREEN$ bug

$DA at $3200 Correct maths bug

$28 at $006D Correct NMI bug

1.81 In Port $FF

10.22 In Port $FF

A few programs read Spectrum port $FF to synchronise their output with the Spectrum attributes. Speculator returns a cyclical value, 0 to 255 if this port is read, so such programs do not get stuck but displays may flicker.

1.82 A few incompatibilities

10.23 A few incompatibilities

There are a few bugs in the emulator which cause problems with a few games; Dan Dare tends to run out of time too quickly, however, the scrolly demo at the beginning works fine (this doesn't show up on some other popular emulators). Arkanoid sprites tend to flicker quite a lot and I don't know why. Hall of the Things keys don't seem to work.

These are a few examples, if anyone has any explanations or finds any more games that have problems, please get in touch. I hope to iron out many of these problems by the next releases.

Note: This is the first official release of Amiga Speculator. Although it has been tested on several different machines, there are bound to be a few unforseen bugs and problems. These will be sorted out as and when I am made aware of them although new releases will be kept to a minimum. Look out for the next release some time toward the end of summer 1996.

1.83 Development history of Speculator

11.00 Development history of Speculator

Speculator was entirely written from scratch by me (William James), including the Z80 emulator. It was started in the summer of 1988 on a Sinclair QL, and was running demonstrably, if not reliably, by November of that year. It was an unusual program to write in that incremental testing was not possible. Over 15,000 lines of assembler were written before it could be executed for the first time and the "© 1982 Sinclair Research Ltd" message arrived rather slowly on the screen. I believe it to be one of the first Spectrum emulators to be written. However, I rapidly lost interest in the whole thing after a software house offered me £xxx for it. I then realised I was writing for the wrong computer. Once I had bought an Amiga it took a good while to get the hang of the operating system. I was also at Essex University doing a Systems architecture degree. Since then I have sporadically worked on it when I had other things to do. The whole thing would never have seen the light of day without the encouragement and help of Simon N Goodwin, who did a truly astounding amount of finishing off on the QL version before releasing it as PD. in 1993. Between then and 1995 most of my time has been spend working on another Amiga related project and Speculator had to be put on hold. In the last few months I have been working hard ironing out as may bugs as possible and doing all the boring bits I had put off for so long (like this documentation).

Anyway, to cut a long story short, yes, I know it's 1995 now, I know Spectrums are now getting more interesting to archaeologists than anyone else. I am also aware there are about 40 other Spectrum emulators available for various machines. In particular "ZXAM" by Antonio J. Pomar Rosselló and Peter McGavin`s "Spectrum" on the Amiga, both of which are excellent emulators. However, I think that Speculator has enough going for it to make it worth releasing. I will be pleased if there is just one person out there who likes it enough to send me a tenner. I would probably do a little dance and sing a song if there were ten.

For the revision history of Speculator see the file README.txt which will be updated as it becomes necessary.

11.10 Future releases of Speculator

1.84 Future releases of Speculator

11.10 Future releases of Speculator

Depending on how Speculator is received I still have plenty of ideas for further improvements. I am also open to suggestions at the address supplied in section 9. Here is a list of some of the things I am intending to implement in version 2 of Speculator.

o Speeded up instruction despatch method for the Z80 emulator. Although it is already pretty efficient, there is a better method which is being worked on already.

o Real time instruction execution on fast enough machines. At the moment Speculator is trying to run as fast as it can. In future it will execute each instruction at the correct rate on a fast processor. This will mean improved sound emulation and better tape loading including hyper load. The theory of operation has already been worked out.

o 128K Spectrum emulation including the Ay-3-8910 sound chip.

o Built in Z80 monitor.

o Load and run multiple projects (snapshots)

o More control over palette settings etc.

o Improved EasyKeys that doesn't care about spaces in GOTO and GOSUB

I am open to any ideas or suggestions for improving Speculator. Please write to the address given in section 9.

1.85 Acknowledgements

12.00 Acknowledgements

Speculator was designed and written by William James, with help from:

Simon N Goodwin - helped in so many areas I can't even begin to list them.

Simon Jenkins - helped on optimising screen handling.

DJH Hartley - for allowing me to include ZA80 with Speculator.

Dave 'Halls' Barker - for letting me include his disassembler into Speculator.

Malcom Lear - for giving me the original idea back in 1988.

Frank Swift - for writing his MGT handler and supporting ZX82 format.

Dennis Briggs - for demonstrating the original Speculator to people at QL shows and for introducing me to Simon Goodwin.

HiSoft - for writing Devpac/Genam Amiga.

Amstrad - for giving permission to emulator writers to use the ZX Spectrum ROM.

Also, Glyn Kendall, Balor Knight, Revelation, Spectrum Adventurer's Guild, BETASOFT, Dr. Andy Wright, Chris Pile, Mark Howlett & anyone else I have forgotten to mention for giving me permission to distribute their Spectrum software with Speculator.

Finally, I would like to dedicate this program to the memory of my father, Erik James, who died on the 7th of August 1994 aged 51.

1.86 Appendix A - File formats

13.00

13.10 ZX82 File Format

13.20 Mirage and JPP Snapshot file format

13.30 KGB file format

13.40 PC Z80 file format

13.50 Sinclair QL Speculator file format

13.60 .SP file format

1.87 Amiga Speculator 'ZX82' file format

13.10 Amiga Speculator 'ZX82' file format

Amiga Speculator has its own file format which I have called ZX82 format due to the fact it contains a file identifier in the first four bytes consisting of the ASCII characters "ZX82". The format has a 12 byte header which contains the normal Spectrum type file information like length, type, start etc. as well as a compression flag which is set if the file is byte run compressed. Snapshot files have a further 32 bytes of register values and border colour information. Listed below are the offset definitions taken from the Speculator source code in case you need to write a conversion utility. All registers and other values are in Motorola format (High, Low). I have defined everything in bytes to avoid any possible confusion.

* The Standard ZX82 Header

ZX_ID rs.l 1 Identifier for a Speculator file "ZX82"

ZX_Type rs.b 1 0:BASIC 1:Numeric 2:String 3:Code 4:Snapshot

ZX_Comp rs.b 1 Is data block byte run compressed ? $00=No $FF=Yes

ZX_Length_H rs.b 1 File length up to 64k (ELINE-PROG for BASIC)

ZX_Length_L rs.b 1

ZX_Start_H rs.b 1 Start address for code (AUTOSTART for BASIC)

ZX_Start_L rs.b 1

ZX_ProgLen_H rs.b 1 Array name (VARS-PROG for BASIC)

ZX_ProgLen_L rs.b 1

ZX_ZXHdrLen rs.b 0 Length of ZX file header

ZX_ZXData rs.b 0 Start of Data block for standard ZX file

* The extended Snapshot ZX82 Header

ZX_Border rs.b 1 Border colour

ZX_IntMode rs.b 1 IntMode over-ride (0=use i_reg, 1=im1 and 2=im2)

ZX_Registers rs.b 0 Z80 register values for Snapshot Files

ZX_iy_H_reg rs.b 1 (High then Low i.e. Motorola format)

ZX_iy_L_reg rs.b 1

ZX_ix_H_reg rs.b 1

ZX_ix_L_reg rs.b 1

ZX_de_H_reg rs.b 1

ZX_de_L_reg rs.b 1

ZX_bc_H_reg rs.b 1

ZX_bc_L_reg rs.b 1

ZX_hl_H_reg rs.b 1

ZX_hl_L_reg rs.b 1

ZX_af_H_reg rs.b 1

ZX_af_L_reg rs.b 1

ZX_de_H_alt rs.b 1

ZX_de_L_alt rs.b 1

ZX_bc_H_alt rs.b 1

ZX_bc_L_alt rs.b 1

ZX_hl_H_alt rs.b 1

ZX_hl_L_alt rs.b 1

ZX_af_H_alt rs.b 1

ZX_af_L_alt rs.b 1

ZX_sp_H_reg rs.b 1

ZX_sp_L_reg rs.b 1

ZX_if_H_reg rs.b 1

ZX_if_L_reg rs.b 1

ZX_rf_H_reg rs.b 1

ZX_rf_L_reg rs.b 1

ZX_pc_H_reg rs.b 1

ZX_pc_L_reg rs.b 1

ZX_SnpHdrLen rs.b 0 Length of Snapshot file header

ZX_SnpData rs.b 65496 Start of data block for Snapshot type file

The ZX_Type field is derived from the MGT diciple directory MGT_Type-1, so further file types may be supported in this way in the future.

The compression used is the standard byte run compression as used by ILBM IFF files. The whole 48k data block is compressed as if it were one long row. See Amiga ROM Kernel Reference Manual: Devices Third Edition, Appendix A - IFF Specification (P347), Appendix C - Example Packer C code (P538).

1.88 Mirage and JPP Snapshot file format

13.20 Mirage and JPP Snapshot file format

This format is based on the format used by the Mirage Microdriver "Dump" command. Snapshot files are always 49179 bytes long. Note that in the table, the byte offset starts from 0, not 1.

Byte Description

0 i register

1 l' register

2 h' register

3 e' register

4 d' register

5 c' register

6 b' register

7 f' register

8 a' register

9 l register

10 h register

11 e register

12 d register

13 c register

14 b register

15 iy low register

16 iy high register

17 ix low register

18 ix high register

19 bit 2 is set if interrupts are enabled

20 r register

21 flags register

22 a register

23 sp low register

24 sp high register

25 interrupt mode (0, 1 or 2)

26 border colour in low 3 bits

27..49178 48 kbytes ram dump

The JPP Spectrum emulator for 80386-based PC's by Arnt Gulbrandsen uses exactly the same snapshot format as does VGASPEC, another PC ZX emulator.

1.89 KGB file format

13.30 KGB file format

Author: KGB (?)

Program: Spectrum

machine: Amiga

Country: ?

Extension: .KGB

Length: 49486 bytes

Byte Length Description

0 - 131 132 132 Pad bytes

132 - 49283 49152 48k data block

49284 - 49415 132 Another 132 pad bytes

49416 2 $000A (?)

49418 2 <> $0000 (?)

49420 2 Int Mode Stuff 1

49422 2 Int Mode Stuff 2

49424 2 $0001 ?

49426 2 $0100 ?

49428 2 Colour/BW mode

49430 2 $0001 ?

49432 2 $007D ?

49434 2 BC register

49436 2 BC alt register

49438 2 DE register

49440 2 DE alt register

49442 2 HL register

49444 2 HL alt register

49446 2 IX register

49448 2 IY register

49450 1 I register

49451 1 R register

49452 1 Pad byte

49453 1 Flags in 68K format

49454 1 Pad byte

49455 1 a alt register

49456 1 Pad byte

49457 1 a register

49458 1 Pad byte

49459 1 Flags alt in 68K format

49460 2 Pad word

49462 2 pc register

49464 2 Pad word

49466 2 sp register

49468 - 49485 18 18 pad bytes

Note: All word values are stored in motorola (hi-lo) format. Also the flags themselves are stored as if they were 680x0 flags which means that only the carry, sign and zero flags are saved correctly in this format. I hope this information is correct as I can`t find the origional documentation.

1.90 PC .Z80 File Format

13.40 PC Z80 file format

The old .Z80 snapshot format (for version 1.45 and below) looks like this:

Byte Length Description

0 1 A register

1 1 F register

2 2 BC register pair (LSB, i.e. C, first)

4 2 HL register pair

6 2 Program counter

8 2 Stack pointer

10 1 Interrupt register

11 1 Refresh register (Bit 7 is not significant!)

12 1 Bit 0 : Bit 7 of the R-register

Bit 1-3: Border colour

Bit 4 : 1=Basic SamRom switched in

Bit 5 : 1=Block of data is compressed

Bit 6-7: No meaning

13 2 DE register pair

15 2 BC' register pair

17 2 DE' register pair

19 2 HL' register pair

21 1 A' register

22 1 F’ register

23 2 IY register (Again LSB first)

25 2 IX register

27 1 Interrupt flipflop, 0=DI, otherwise EI

28 1 IFF2 (not particularly important...)

29 1 Bit 0-1: Interrupt mode (0, 1 or 2)

Bit 2 : 1=Issue 2 emulation

Bit 3 : 1=Double interrupt frequency

Bit 4-5: 0=Normal

1=High video synchronisation

2=Normal

3=Low video synchronisation

2=Normal

Bit 6-7: 0=Cursor/Protek/AGF joystick

1=Kempston joystick

2=Sinclair 1 joystick

3=Sinclair 2 joystick

Because of compatibility, if byte 12 is 255, it has to be regarded as being 1. After this header block of 30 bytes the 48K bytes of Spectrum memory follows in a compressed format (if bit 5 of byte 12 is one).

The compression method is very simple: it replaces repetitions of at least five equal bytes by a four-byte code ED ED xx yy, which stands for "byte yy repeated xx times". Only sequences of length at least 5 are coded. The exception is sequences consisting of ED’s; if they are encountered, even two ED’s are encoded into ED ED 02 ED. Finally, every byte directly following a single ED is not taken into a block, for example ED 6*00 is not encoded into ED ED ED 06 00 but into ED 00 ED ED 05 00. The block is terminated by an end marker, 00 ED ED 00.

That’s the format of .Z80 files as used by versions up to 1.45. Since starting from version 2.0 the program emulates the Spectrum 128 too, there was a need for a new format. The first 30 bytes are almost the same as the old versions’ header.

In the flag byte, bit 4 and 5 have got no meaning anymore, and the program counter (bytes 6 and 7) are zero to signal a version 2.0 .Z80 file. So loading a new style .Z80 file into an old emulator will cause an error or a reset at the most.

After the first 30 bytes, an additional header follows:

Byte Length Description

30 2 Length of additional header block (contains 23)

32 2 Program counter

34 1 Hardware mode: 0=Spectrum 48K,

1=0+interface I,

2=SamRam,

3=Spectrum 128K,

4=3+interface I.

35 1 If in SamRam mode, bitwise state of 74ls259.

For example, bit 6=1 after an OUT 31,13 (=2*6+1)

If in 128 mode, contains last OUT to 7ffd

36 1 Contains 0FF if Interface I rom paged

37 1 Bit 0: 1 if R register emulation on

Bit 1: 1 if LDIR emulation on

38 1 Last OUT to fffd (soundchip register number)

39 16 Contents of the sound chip registers

Hereafter a number of memory blocks follow, each containing the compressed data of a 16K block. The compression is according to the old scheme, except for the end-marker, which is now absent.

The structure of a memory block is:

Byte Length Description

0 2 Length of data (without this 3-byte header)

2 1 Page number of block

3 [0] Compressed data

The pages are numbered, depending on the hardware mode, in the following way;

Page In ’48 mode In ’128 mode In SamRam mode

0 48K rom rom (basic) 48K rom

1 Interf. I rom Interf. I rom Interf. I rom

2 - rom (reset) samram rom (basic)

3 - page 0 samram rom (monitor,..)

4 8000-bfff page 1 Normal 8000-bfff

5 c000-ffff page 2 Normal c000-ffff

6 - page 3 Shadow 8000-bfff

7 - page 4 Shadow c000-ffff

8 4000-7fff page 5 4000-7fff

9 - page 6 -

10 - page 7 -

In 48K mode, pages 4,5 and 8 are saved. In SamRam mode, pages 4 to 8 are saved. In 128 mode, all pages from 3 to 10 are saved. This version saves the pages in numerical order. There is no end marker.

The .Z80 file format is also used by;

- G.A.Lunter, PC, Netherlands.

- SPECTATOR, Carlo Delhez, QL, Netherlands.

- Ergon Development emulators, QL, Italy.

- ZX, Andrew Lavrov, QL.

- Mac Spectacle, Guenter Woigk, Germany, 1995.

- ZXAM, Antonio J. Pomar Rosselló, Amiga, Spain.

These notes were taken directly from G.A. Lunter`s documentation.

1.91 Sinclair QL Speculator file format

13.50 Sinclair QL Speculator file format

Speculator supports standard ZX file types 0-3 by mapping them onto unused Qdos file-types 3-6, adding ZX-specific information in the Qdos file header. SNAP files have Qdos type 11.

ZX Qdos

0 3 ZX BASIC program (may include variables)

1 4 ZX BASIC numeric array

2 5 ZX BASIC character array

3 6 ZX BASIC CODE or SCREEN$ file

5 11 GDOS 48K SNAP

The length of the ZX file can be inferred from the length in the QL header -64, the number of bytes used for the header itself. The normally unused 'extra' field at byte offset 10 in the Qdos file header holds the remaining information from the ZX file header: Bytes 10 and 11 hold the start address for CODE or the start line for BASIC, in Z80 order, LSB first. If the start line exceeds 9999 no auto-start is performed. The name of a saved array is encoded at byte offset 11 of the file header. To decode this, AND the value with 31 and add 64 to convert to ASCII capitals. File type 11 is a Qdos equivalent of the Spectrum G(+)DOS SNAP 48K file-type, with register information added at the end of the 48K Spectrum memory image:

Offset Length Contents

0 48K ZX Memory image

49152 word IY register (LSB first)

49154 word IX register (LSB first)

49156 word DE' register (order: E',D')

49158 word BC' register (order: C',B')

49160 word HL' register (order: L',H')

49162 byte F' register (not always set by GDOS)

49163 byte A' register

49164 word DE register (order: E,D)

49166 word BC register (order: C,B)

49168 word HL register (order: L,H)

49170 byte F register; P/V flag (bit 2) signals IFF2 (DI/EI)

49171 byte I register; IM 2 if I<>63, otherwise IM 1

49172 word SP, stack pointer, LSB first

49174 end of file

Once the snapshot is loaded into the Spectrum memory, the remaining registers can be found on the machine stack:

(SP+0) byte F register (P/V flag signals interrupts ON or OFF)

(SP+1) byte R register

(SP+2) byte F register (this is the valid one!)

(SP+3) byte A register

(SP+4) word PC, Program Counter, LSB first

When these values are fetched, the SP must be increased by 6 before re-starting Z80 code emulation.

1.92 .SP file format

13.60 .SP file format

Author: Pedro Gimeno

Program: Spectrum

machine: PC

Country: Spain

Extension: .SP

Length: 49190 bytes

38 bytes of header information as follows

Byte Description

0 "S" File Identifier

1 "P" File Identifier

2 Data length low

3 Data length high

4 Load address low

5 Load address high

6 c_reg

7 b_reg

8 e_reg

9 d_reg

10 l_reg

11 h_reg

12 f_reg

13 a_reg

14 ix_L

15 ix_H

16 iy_L

17 iy_H

18 c_alt

19 b_alt

20 e_alt

21 d_alt

22 l_alt

23 h_alt

24 f_alt

25 a_alt

26 r_reg ?

27 i_reg

28 sp_L

29 sp_H

30 pc_L

31 pc_H

32 unknown

33 unknown

34 border colour

35 unknown

36 Interrupt bits

bit 1 = ints on/off

bit 2 = intmode (0=1 1=2)

37 unknown

38 48K Data block starts here

Note: I had to work this one out myself so there are a number of unknown bytes.

1.93 Appendix B - List of tooltypes

14.00 List of tooltypes

SNP_DRAWER={file path}

SNP_FILE={file name}

SNP_PATTERN={pattern}

STD_DRAWER={file path}

STD_FILE={file name}

STD_PATTERN={pattern}

DIS_DRAWER={file path}

DIS_FILE={file name}

DIS_PATTERN={file pattern}

PRT_FILE={printer device or file}

EASYKEYS=TRUE|FALSE

HIRES=TRUE|FALSE

COMPRESSION=TRUE|FALSE

CREATEICONS=TRUE|FALSE

AUTORUN=TRUE|FALSE

TAPELOAD=TRUE|FALSE

VOLUME=OFF|QUIET|NORMAL|LOAD

QUIET={0 - 64}

NORMAL={0 - 64}

LOUD={0 - 64}