

TRS80Model1

COLLABORATORS

	<i>TITLE :</i> TRS80Model1	
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>
WRITTEN BY		April 12, 2022
<i>SIGNATURE</i>		

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	TRS80Model1	1
1.1	TRS80Model1	1
1.2	Requirements	2
1.3	Description	2
1.4	Why A TRS80 Model 1 Emulator?	2
1.5	Running the Emulation	3
1.6	Keyboard Mapping	3
1.7	Speed Regulation	4
1.8	Program Information	4
1.9	Planned Improvements	4
1.10	Bugs? What Bugs?	4
1.11	About the Author	6
1.12	Payment	6
1.13	Credits	6
1.14	History	7
1.15	Finding Software	7
1.16	Compatibility	7

Chapter 1

TRS80Model1

1.1 TRS80Model1

TRaSh 80
The only (?) Model 1 emulator for the Amiga
Copyright © 1997, by
The RED SKULL
All Rights Ignored

Requirements

Description

Why A TRS-80 Emulator?

Running the Emulation

Keyboard Mapping

Compatibility

Speed Regulation

Bugs?

Program Information

Shareware fee?

Program History

Getting Hold of some software

1.2 Requirements

REQUIREMENTS:

- o Amiga computer with Kickstart 2.0 or newer
- o A 68020+ CPU. Emulation WILL NOT WORK on a 68000 system.
- o About 512k free RAM (preferably most of it FAST RAM)
- o ASL.library is used for the File Requesters

1.3 Description

DESCRIPTION:

"TRaSh 80" is currently the only TRS80 Model 1 emulator (as far as I am aware) for the Amiga computer. At present it emulates the following configuration:-

- o Z-80 CPU (including a number of Undocumented Instructions)
- o Model 1 Keyboard at the hardware level.
- o 48k RAM
- o Load "/CMD" and execute directly from AmigaDOS
- o Write protection for the ROM area

Coming in the next version: (if there is sufficient interest in the project)

- o Load/Save Snapshot
- o 4 80Tk Single Sided Disk Drives (MAYBE DOUBLE/DEN)
- o Serial Port Support.
- o Printer Port Support. (print to an amiga disk file)
- o Virtual Cassette Support

The emulation also runs in a reasonably system friendly manner, multitasking properly with other programs. It does, however, bypass the graphics library calls in favour of directly writing to it's own screens Bitmap. This was done primarily for speed reasons. This may or may not be compatible with third party graphics cards.

It has been tried on a PICASSO 2 and worked fine.

1.4 Why A TRS80 Model 1 Emulator?

WHY A TRS 80 Model 1 EMULATOR?

Primarily it was the first computer that I owned. I remember the thrill of getting my Level1 4k machine home for the first time early in 1978. It was miles ahead of the single board 6502 machine I had been playing with up to that point.

Secondly I feel that it is much easier for a beginner to create a working program on one of these old 8-bit machines than on one of today's super powered (by comparison) home computers. Just look at myriad compile options on any of today's C-compilers!

Thirdly a lot of the old software is really worthwhile, despite the graphics limitations, and addictive to play.

Last but not least because I envied the IBM-PC crowd having a Model 1 emulator!

1.5 Running the Emulation

RUNNING THE EMULATION

Simply Click on the TRaSh-80 icon or run it from a CLI.

The gadgets are pretty self-explanatory.

1.6 Keyboard Mapping

The Keyboard of TRaSh 80 corresponds (pretty much) to the keys of the AMIGA

The <HELP> key is the TRaSh 80 <CLR> key.

The key is the TRaSh 80 <BREAK> key.

Both Left and Right <SHIFT> keys are read as a LEFT <SHIFT> by the Z-80 routines.

One slight problem exists with selection of shifted key combinations.

If a shifted key combination is pressed and the <SHIFT> key is released FIRST, then that key's ordinary function cannot be accessed until the previously shifted combination is again selected and the <SHIFT> key released LAST.

This shouldn't cause any problems, just be aware if certain key combinations don't seem to work this is probably why.

1.7 Speed Regulation

SPEED REGULATION

Come on! This is only a pre-release Beta version.

1.8 Program Information

TRaSH 80 is written in a mixture of C and 68020 Assembler

SAS 6.57 and PHXass (freeware) were used respectively in development, with CED V3.5 the editor of choice.

Gadget Handling, program loading/snapshot , keyboard emulation are in C. The Z-80 emulator and the Screen write routines are written in assembler and run as a separate process on their own.

The Z-80 core uses an "instruction pipelining" system, not dissimilar to that used in the excellent "APPLE2000" by Kevin Kralian. While tending to be rather memory hungry, speed wise it is very efficient.

ASL.library is used for the file requesters as nothing elaborate was required on the 1 Bitplane TRaSh 80 screen.

The Z-80 HALT instruction will currently terminate program operation.

1.9 Planned Improvements

PLANNED IMPROVEMENTS

1.10 Bugs? What Bugs?

Lets call them Program limitations.

1. Stack Pointer (SP).

Due to the way the Z-80 is emulated the STACK PTR (SP) can point to mem locations outside of the 64k address space. Consider the following fragment.

```
31 FE FF ; LD SP,$FFFE
E1      ; POP HL (SP NOW $10000)
E1      ; POP HL (SP NOW $10002)
```

Unfortunately SP is now pointing outside memory that TRaSh 80 has

AllocMem'ed from the EXEC memory pool. The GURU awaits.....

The SP cannot however be explicitly loaded with a value outside of the 64k address space. Any instructions like INC SP will NOT send it out of the 64k range, only POP's and PUSH's.

This could on the other hand be a usefull feature. Would anyone like to see support for a stack that sits outside the main 64k memory? For that matter would anyone like support for block moves into and out of the 64k address space?

2. Undocumented Instruction.

A number of the more popular undocumented instructions are currently implemented. The next couple of versions of TRaSh 80 will have more.

3. Screen Update Routines.

Currently all instructions that write into the Screen Memory area (\$3C00-\$3FFF) will cause an on-screen change, with the following exceptions:-

All Stack operations. Let's face it, code that tries to write to the screen by PUSH's and POP's must be CRAP!

4. Memory Protection.

Memory locations < \$3C00 (start of the screen) will not be affected by instructions that write into system memory, with the exception of:-

All stack operations. Again only bad code would create such a situation.

This means that the ROM area (\$0000-\$2FFF) is protected from spurious writes by misbehaving programs.

5. Index Register Operation.

The index register READ instructions (IX+d) and (IY+d) currently do NOT wrap-around when they index past \$FFFF or past \$0000 in the other direction. This shouldn't cause many problems except for software that is already buggy.

Index register WRITE operations function correctly and will wrap at the 64k boundaries. Indexed writes to the ROM area are intercepted.

6. Several of the Z-80 instructions do not yet operate correctly. Notably the DDCB and FDCB opcodes. However they do not seem to be used by the ROM to run BASIC. This is being worked on as you read this.

Any bug reports or suggestions for improvements please contact me at <layling@intercoast.com.au>

1.11 About the Author

ABOUT THE AUTHOR

TRaSh 80 was written over the course of about 6 months.

Spurred on by John Fehr's MODEL 3 program, and the abundance of TRS80 software appearing on the InterNet, I decided to go ahead with the task.

My interests include HAM radio (VK2LR) , computer hardware design, organic farming and leading a self-sufficient lifestyle.

My next project will be an emulator for the OHIO Scientific "Superboard II" / "C1P" computers. For those unfamiliar with it, it was kind of a predecessor to the Apple][. It had 8k BASIC in ROM, keyboard integrated into the main board, with a 6502 processor. It found wide acceptance in engineering and scientific applications as a result of the high quality (and extensive range) of the peripheral cards that became available.

However information is sparse, so any mem-maps, monitor routine entry points, etc. would be greatly appreciated.

If you like this program take the trouble to e-mail me , it's always nice to know who is using your programs. <layling@intercoast.com.au>

1.12 Payment

This program is being released as FREeware.

That means you are free to COPY, USE, and ENJOY this software for personal use, provided that the distributed archive is copied intact.

I would be pleased to receive any e-mails from people who are using this program and would like to see it further developed.

However if you wish to distribute this program on CD or diskette as part of a collection, I insist on receiving a copy. Also no more than a reasonable fee for duplication must be charged.

The Z-80 emulation core of TRaSh 80 is rather zippy. If you would like to include it as part of another emulator project feel free to contact me at <layling@intercoast.com.au>

If you have a commercial project that you think may benefit from the Z-80 core, likewise feel free to contact me.

Naturally users of this program accept all responsibility for it's use, suitability for any purpose, etc, etc.....Legal Babble....

1.13 Credits

CREDITS

I owe lots of thanks to lots of people.

1.14 History

HISTORY:

V0.8 - 14/08/97

First AMINET release.

1.15 Finding Software

FINDING YOURSELF SOME SOFTWARE

There are 2 main sources on the NET to the best of my knowledge.

The "TRS-80 Page" at [HTTP://WWW.KJSL.COM/](http://WWW.KJSL.COM/) includes a lot of general information as well as a link to their FTP site.

The other is IRA GOLDKLANG's TRS80 pages at [HTTP://WWW.INFONORTH.COM/~GOLDKLANG](http://WWW.INFONORTH.COM/~GOLDKLANG) he has many Model 1 and also Model3 programs available for http download.

1.16 Compatibility

COMPATIBILITY

All TRS-80 programs for the Model 1 should work straight away.

Problems are likely in the following areas:

1. Assembler programs using "Undocumented Instructions".

A number of undocumented instructions that affect the HIGH and LOW bytes of IX and IY exist in the DD and FD ranges.

TRaSh 80 implements some of these instructions, however the next release will implement more still.

2. Programs trying to "LPRINT". Since printer support is not yet active, programs trying to output to the printer port will simply hang.
-

3. Likewise the Serial Port.

4. " " the Disk Drives.