>Remember

By Hugo Labrande

Issue #5 : Z-Machine interpreters

In the 1980s, Infocom published about three dozen of text adventures on most platforms of the era, from the TRS-80 to the Atari ST. Their method to spend minimal time on portability was to create a virtual machine, the Z-Machine, and compile their games into bytecode that could be read by the Z-Machine. All was needed was some software that could read this bytecode and execute it on a target machine: an interpreter.

To ensure that all interpreters behaved coherently, Infocom actually wrote specifications for the Z-Machine (version 3, 4, and 5 for text-only, and 6 for graphical adventures), which specified how the interpreter should behave; this was to make sure that the people tasked with writing a new interpreter would follow the same rules as the other ones. This specification was then reverse-engineered by amateurs as early as the end of the 1980s (Barry Boone on the TI-99/4A, the InfoTaskForce in Australia, etc.). In the 1990s, people added some features and fixed a few inconsistencies, and created version 8, which was basically the same as version 5 but allowed larger file sizes. This gave the Z-Machine standards, which are available online:

http://inform-fiction.org/zmachine/standards/z1point1/index.html

This means that if you'd like to implement a Z-Machine interpreter, all you need to do is follow this specification. Over the years, literally hundreds of Z-Machine interpreters have been written, for sometimes very exotic platforms. They sometimes have their own quirks, can fall out of fashion, or get forgotten; I cannot begin to retrace all the Z-Machine interpreters that have been written. However, this month, I would like to give you a taste of all the different platforms that a Z-Machine game can be played on; I hope this will give you some ideas and make you want to experiment.

In order to prepare this article, I scoured the Internet, Github, Sourceforge, the IF-Archive, etc. to find as many interpreters as possible. I'd be remiss if I didn't mention Stefan Vogt's help for the retro interpreters; Stefan spent countless hours finding the best interpreters on 1980s machines, then very generously gave all that information to me and helped me set them up for *Tristam Island*. The information on interpreters for 1980s machines is, thanks to him, of great quality!

A word of warning

Usually, running a game in an interpreter is straightforward, but not always. The procedure can be complicated; the interpreter is not necessarily open source, or can be hard to find; there can be a few bugs left, etc. I will attempt to describe some of the procedures that are required, to the best of my knowledge, to run the games, but writing full tutorials would take too much time. Do not hesitate to reach out if you have a question; having released *Tristam Island* on 36 of these platforms, I definitely know some of the finer details for these, at the very least!

All interpreters here run v3 games, and most run v5 games or more; I will attempt to mention this whenever possible.

And lastly, at the end of this document, I will mention a few open problems – as in, platforms that really could have a Z-Machine interpreter, but don't. If anyone wants to help, or knows about something I don't, please get in touch!

8-bit 1980s era

- **Amstrad CPC & PCW**: the Infocom interpreter is written in z80 assembly and works for both Amstrad CPC and PCW, and (I believe) supports the Z-Machine v3 and v5. Another possibility is the interpreter ZXZVM, written by John Elliott: he wrote a version that works for Amstrad PCW, available on his website, and it was recently ported on the Amstrad CPC by Kevin Thacker:
  http://www.seasip.info/ZX/zxzvm.html
  https://www.cpc-power.com/index.php?page=detail&num=16439
  Be warned, however, that its performance isn't as good as Infocom's interpreter.

- **Apple II**: Infocom published interpreters for v3 and v5, with the former fitting with a v3 game on one side of an Apple II disk; a very detailed writeup was published at KansasFest 2017, but be warned that Inform 6.34 (the newest compiler) should be used as the ability to compile to z3 has been restored since this article was published:
  https://www.kansasfest.org/wp-content/uploads/apple_ii_inform_paper.pdf

- **Atari 8-bit**: Infocom's interpreter runs on Atari 400, 800, XE and XL, and supports v3 only; since Atari 400/800 disks are 130kb (and a v3 file can be as large as 128kb), some versions of the interpreter (e.g. *Plundered Hearts*) load the interpreter in memory, then make you flip the disk to play.

- **BBC Micro** (B, B+, and Master) and **Acorn Electron** users have a choice between two interpreters: an older one by Jon Welch ("BBC Micro Z-Machine v1.1", GPL licensed) and a port of Ozmoo by Steve Flintham. The latter is the better one, and development is still active on Github; you'll find in the repository everything you need, including the Python script that generates the disk image. Supports splash screens and v5 games.
  https://zornslemma.github.io/ozmoo.html

- **Commodore PET** and **VIC-20**: Edilbert Kirk (BitShifter) wrote a very nice 6502-assembly interpreter which works on the PET, the VIC-20, the Plus/4, the C64 and the C128. It can run v5 files and has accented character support. I'm not sure it's publicly available, but you can ask him (he is in various C64-related forums). Infocom disk downloads are at:
  http://petsd.net/petfood.php?lang=en

- **Commodore Plus/4, Commodore 64, Commodore 128**: on top of the aforementioned interpreter by Edilbert Kirk, there is Ozmoo, created by Johan Berntsson and Fredrik Ramsberg, which is a great interpreter; it is optimized for speed (with options for pre-loading data in memory), can use different fonts, supports a splash screen in C64 mode, and can read v3, v5, and z8 files. It can also generate tape files if your game is small enough (e.g. Mini-Zork II).
  https://github.com/johanberntsson/ozmoo

And I almost forgot to mention Zeugma, by Linus Akesson, which requires a RAM expansion unit but also supports v3, v5 and v8 files:
http://www.linusakesson.net/software/zeugma/index.php

- **CP/M systems** such as the **Osborne 1**, **DEC Rainbow**, **Kaypro**, **Intertec Superbrain**, etc. can use the Infocom interpreter; however, some customization might be needed so the interpreter behaves well on the target machine. Richard Loxley has documented the process for his Osborne 1 computer:
https://www.richardloxley.com/2018/04/28/osborne-restoration-part-17-text-adventure-games/

- **Famicom**: since the Famicom system had an official keyboard accessory in Japan, user Zzo38 has started (but not completed, I don't think) a Z-Machine interpreter for the Famicom. The code is available here:
http://wiki.nesdev.com/w/index.php/User:Zzo38/Famicom_Z-machine

- **IBM PC**: Infocom's games were released on the IBM PC line, which means that, much like other platforms, the interpreter can be recovered and the data file switched to play any game; v3 is a given, but I'm not sure about v5.

- **MSX 1 & 2**: some confusion exists about whether Infocom actually produced a MSX interpreter (some say they did after the Activition acquisition, but there is no trace of it). It might have been an adaptation of their CP/M interpreter. In any case, SLotman, a well-known Brazilian MSX hacker, has an interpreter for MSX 1 and MSX 2 on his website:
http://www.icongames.com.br/msxfiles/our-en.htm

- **Oric Atmos**: Chema Enguita and Fabrice Frances, two highly regarded Oric programmers, adapted the pinfocom interpreter for the Oric, giving Pinforic, available at the IF-Archive.
https://www.ifarchive.org/indexes/if-archiveXinfocomXinterpretersXoldXpinfocom.html
The interpreter comes on one disk, and the game file (v3 only) should be put on another disc (which is a special format for the Oric – use a tool to create a DOS disk image, then run it through the raw2mfm tool). They've actually done a little bit more work on it recently to make a version with everything on just one disc (and some magic to make this compatible with the Telestrat); they're talking about open-sourcing the latest version in a proper repository soon!

- **Spectrum +3**: there are two interpreters available here. The first one is ZXZVM, which we've talked about in the "Amstrad CPC & PCW" section; the second one is an interpreter by George Beckett and Stefan Vogt, which, from my understanding, they made from an Infocom article published around the time of *Deadline* about their CP/M interpreter. The latter interpreter isn't public yet (ask Stefan!), and requires a CP/M+ & Mallard BASIC disk image to work, which you can purchase at
https://locoscript-software.square.site/product/zx-spectrum-3-cp-m-plus-mallard-basic-dsk-image-/11

- **TI-99/4A**: the interpreter that is in use in the community is seemingly Barry Boone's interpreter, which descended from Infocom's by way of heavy hacking and reverse-engineering. It requires a custom format, for which I have written a conversion script:
https://github.com/hlabrand/retro-scripts

Work on the interpreter is still under way, with the latest developments chronicled in an AtariAge thread below. The interpreter now supports native z3 format (z5 is just impossible on this machine), accented letter support, and even a splash screen! If you're interested, ask InsaneMultitasker:
https://atariage.com/forums/topic/310898-ti-99-infocom-interpreter-dev

- **TRS-80 Model III**: to the best of my knowledge, the only interpreter there is is Infocom's, and they stopped supporting it in 1984 (meaning there could be some bugs left). But Hugh Steers (yes, *that* Hugh Steers!) figured out how to run *Tristam Island* on it, which might give you an idea of the process:
https://intfiction.org/t/new-retro-game-tristam-island/47194/12

- **TRS-80 Color Computer**: again, the Infocom interpreter is the only one there is for it; however, this one is particular, since it's the only Infocom interpreter with source code available, thanks to Brian Moriarty!
https://retrotinker.blogspot.com/2018/02/z-intepreter-source-for-coco-recovered.html
And see the process on how to build disk images on John W. Linville's blog:
https://retrotinker.blogspot.com/2017/11/building-infocom-disk-images-for-coco.html
And apparently, someone named "The Zippster" has figured out a way to run the interpreter in 64 column mode for the CoCoVGA configurations. I haven't been able to, if anyone wants to help me...
https://thezippsterzone.com/2018/05/11/64-column-infocom-interpreter/

16- and 32-bit era

- **Acorn Archimedes**: the Archimedes had Z-Machine ports starting very early (it was when the InfoTaskForce interpreter was made available on that platform that Graham Nelson started playing around with it, which gave birth to Inform). The ITF interpreter is available on the IF-Archive, as well as a later interpreter (ZIP2000):
https://www.ifarchive.org/indexes/if-archiveXinfocomXinterpretersXoldXitf.html
https://www.ifarchive.org/indexes/if-archiveXinfocomXinterpretersXzip2000.html

- **Amiga**: there are a few interpreters on Amiga, including AmigaZIP (a port of Zip by Werther Pirani that requires Workbench 1.2 or later) and Frotz (version 2.43, requires Workbench 2.0 or later). Both support v3, v5 and v8, and Frotz also supports v6 games.
https://www.ifarchive.org/indexes/if-archiveXinfocomXinterpretersXzip.html
http://aminet.net/package/game/text/frotz.lha

- **Apple IIGS**: there are a few interpreters available for the AppleIIGS, including a port of Zip. A more recent interpreter is "On Beyond", by famed Apple II hacker 4am, which can be found on Github:
https://inform-fiction.org/zmachine/appleII.html
https://github.com/a2-4am/pitch-dark/tree/master/src/onbeyond

- **Atari ST**: there seems to be a couple of interpreters on that platform, such as "atarizip", "zorkmachine", "itf" (a port of the InfoTaskForce interpreter) and a port of Pinfocom, on top of the Infocom interpreter, of course (which runs v3 and v5). I use a port of JZIP, John Holder's interpreter, which runs v3 and v5 files; the port to the Atari ST was made by Dancer in 1995. I resurrected this

codebase to hack at it and have proper accented character support; you'll find the binary on the IF-Archive, and the code on my Github:
https://github.com/hlabrand/jzip-atariST

- **DOS**: there are a few interpreters for DOS, but most notable is Frotz, which is still updated and compiled for 16-bit MS-DOS with each of its releases. It can be found on the IF-Archive:
  https://www.ifarchive.org/indexes/if-archiveXinfocomXinterpretersXfrotz.html

- **Macintosh**: the MaxZip interpreter, a port of Mark Howell's Zip by Andrew Plotkin, is very nice, with a clean interface, v8 support, and good (but not perfect) support for accented characters. It requires System 7 and 1600kb of free memory:
  https://www.ifarchive.org/indexes/if-archiveXinfocomXinterpretersXzip.html

Software ecosystems and operating systems

- **BeOS**: there are two interpreters, a port of Frotz and a port of JZIP, with the latter having nice features like font selection and a GUI. This website has versions that are later than the ones on the IF-Archive:
  https://www.zophar.net/beos/zmachine.html

- **Java**: these two interpreters were really big in the 2000s, and for a while were the only way to play Z-Machine games in a browser:
  https://sourceforge.net/projects/zmpp/
  https://sourceforge.net/projects/zplet/

- **Javascript**: Parchment is a really nice interpreter with support for most of the specification (timed events, unfortunately, aren't handled) and v3, v5 and v8 compatibility; it powers the iplayif.com website, but you can install it on any website of yours, and it is bundled with Inform 7. There is also Encrusted, an interpreter written in Rust and compiled in WebAssembly; it looks very nice and has very interesting features (such as live mapping and narration), but is limited to v3. (I also haven't been able to compile it from source, and it looks like it's been abandoned.)
  https://github.com/curiousdannii/parchment
  https://sterlingdemille.com/encrusted/

- **PDP-11**: an interpreter named Zemu (not to be confused with the one for TI-calculators) is available for PDP-11:
  http://www.ifarchive.org/indexes/if-archiveXinfocomXinterpretersXzemu.html

- **Python**: there are a number of interpreters in Python, on the IF-Archive or in Github. Python code is pretty legible, so it might be a good start if you want to look into how an interpreter works:
  https://github.com/DFillmore/viola
  https://github.com/theinternetftw/xyppy
  https://github.com/sussman/zvm

- **RiscOS**: a good resource, gathering quite a few interpreters, is this page:
  https://inform-fiction.org/zmachine/riscos.html

- **OS/2**: Frotz and Zip have both been ported on OS/2, though I don't know which one is the better one:
  https://inform-fiction.org/zmachine/os2.html

- **TOPS-20**: Adam Thornton has a code repository attempting to adapt the command-line version of Frotz for the TOPS-20 operating system, meaning Frotz could run Zork on a PDP-10 again, just like in the Infocom days. It's on Github:
  https://github.com/athornton/tops20-frotz

- **TempleOS**: a port of Frotz is available for this operating system:
  http://www.ifarchive.org/indexes/if-archiveXinfocomXinterpretersXfrotz.html

Handhelds

- **Archos PMA430**: Frotz has been ported to this platform, but I don't have much more information than that:
  http://www.ifarchive.org/indexes/if-archiveXinfocomXinterpretersXfrotz.html

- **Apple Newton**: to the best of my knowledge, the only interpreter seems to be YAZI, written by George Madrid and Sanjay Vakil, and based on the Zip interpreter by Mark Howell. It reads v3 files only, and the only known package is a shareware package with pared-down features (for instance, any save made after 50 turns cannot be restored, unless you pay the 25$ registration fee). Some people online report having been able to contact Sanjay Vakil (even recently) and he generated a registration key for them, but there's not really an automated method that works yet. You can find the interpreter here:
  https://www.ifarchive.org/indexes/if-archiveXinfocomXinterpretersXoldXyazi.html

- **Android**: there are many interpreter apps on Android, and some are taken down or never updated anymore, etc. A recent and very nice one is Fabularium, which has a clean interface (and even allows you to run I6 code); I'm a big fan of the interface of Text Fiction, and I also like Jfrotz for its hardware keyboard (which takes less space than the regular Android keyboard).

- **Blackberry**: it appears like the creator of Jfrotz for Android started out with a project of a Z-Machine interpreter for the Blackberry. I'm really not sure how to make it work at this point, but here is the link anyway:
  https://sites.google.com/site/zaxmidlet/

- **Game Boy** and **Game Boy Color**: just because there is no keyboard doesn't mean that the Game Boy doesn't have an interpreter! The interpreter's name is INFGMB, and was written by Martin Korth (alias noca$h, a prolific writer of emulators and other tools). The latest version is on Martin's website, and has script that'll allow you to generate the package from your story file. Be warned: the interpreter only supports v3 and is not exempt of bugs (for instance when you use more than 64 abbreviations). Of course, you usually

need a special device (a linker) that emulates a cartridge but has a microSD slot in order to run this interpreter on real hardware.
https://problemkaputt.de/infgmb.htm

- **Game Boy Advance**: I believe the only interpreter is GBAFrotz, a port of Frotz 2.41 by Jonas Minnberg. It runs v3, v5 and v8 files, and a lot of thought and care went into its usability; the buttons are mapped to common verbs, you can still select letters from a virtual keyboard with a combination of buttons, and there is the possibility to grab words from the screen to put them in your input. Minnberg's old website is down, but still accessible through the Wayback Machine, including a full source download. The executable (and scripts to create a GBA file) are on the IF-Archive; of course, you'll need a linker to play this on real hardware:
https://www.ifarchive.org/indexes/if-archiveXinfocomXinterpretersXfrotz.html

- **GP2X**: Frotz has been ported to this Linux-based handheld device:
https://www.ifarchive.org/indexes/if-archiveXinfocomXinterpretersXfrotz.html

- **iPhone**: again, quite a few interpreters on the App Store, and I wouldn't be able to say which is the nicest, but I've heard a lot of people like Frotz.

- **Kindle Touch**: Kindle Touch / Paperwhite e-Readers have a port of Frotz; I don't really know how usable it is, but since these devices have a touchscreen, it shouldn't be too bad:
http://www.ifarchive.org/indexes/if-archiveXinfocomXinterpretersXfrotz.html

- **Kobo**: these e-Readers also have their own port of Frotz:
http://www.ifarchive.org/indexes/if-archiveXinfocomXinterpretersXfrotz.html

- **Nintendo DS**: there are two interpreters available on Nintendo DS, and I believe both are based on Jonas Minnberg's work for the GBA and the NDS. The first one is an interpreter that was based on Glkpogo, by Jonas Minnberg, which allowed to port Frotz easily; binaries are available here (select the glk_pogo one), and it should work by putting the story files in the right folders:
http://errabes.free.fr/pogo2/
The second one is DSFrotz, by PapaFuji, and is actually a full-fledged application that will also display metadata, cover art, organise your collection, etc; it even has stylus support, meaning you can draw each letter to add it to the input. Its core is based on GBAFrotz by Jonas Minnberg; the author has also released some bundles of free games, including the Scott Adams and Infocom games, on that format. The website is still up with the latest version, but an older version (with sources) is also available on the IF-Archive:
http://gugusse.central.free.fr/papafuji/DSFrotz.html
http://www.ifarchive.org/indexes/if-archiveXinfocomXinterpretersXfrotz.html

- **Nokia Communicator**: there seems to be an interpreter descending from the Zip interpreter on these devices:
https://inform-fiction.org/zmachine/nokia.html

- **Palm OS**: you actually have quite a few choices for Palm OS devices! There's a port of Frotz, but also Frobnitz (which claims to run on devices without

expanded memory), and Kronos, which bundles in a Magnetic Scrolls interpreter. All three run all kinds of Z-Machine games except v6 games:
https://inform-fiction.org/zmachine/palmos.html

- **PlayStation Portable**: there seems to be a port of Frotz on the PSP, though I have never tried to use it. (However, the huge number of emulators available on the PSP means that you could in theory run another interpreter inside an emulator!)
https://www.brewology.com/downloads/download.php?id=11932&mcid=1

- **PocketPC**: the IF-Archive has a port of Zip for PocketPC:
http://www.ifarchive.org/indexes/if-archiveXinfocomXinterpretersXzip.html

- **Psion**: there are two interpreters on Psion, a port of Frotz, and a port of the old InfoTaskForce interpreter. A friend of mine was able to confirm that Frotz still runs on a Psion. That's all I know!
https://inform-fiction.org/zmachine/epoc.html

- **Sharp Wizard OZ-7XX** : the only interpreter there seems to be is "Z-Machine 1.2", an adaptation of the Zip interpreter by Mark Howell, written by Shawn Roberts and Matthew Holladay, which runs only v3 games; it is on the IF-Archive:
https://www.ifarchive.org/indexes/if-archive/infocom/interpreters/zip/

- **Texas Instruments TI-83** and **TI-84** : those z80-based calculators (with a very cool homebrew scene) have recently gotten a Z-Machine v3 interpreter by Nicholas Hunter Mosier, who worked some more on it for Tristam Island; however, he does not have time to work on it anymore. The code is on Github, and a conversion script is on my Github:
https://github.com/nmosier/zemu
https://github.com/hlabrand/retro-scripts

- **TI-89** and **TI-92**: the Foblub interpreter is a port of the Pinfocom interpreter (which can only do v3), extended to get v4 and v5 compatibility on these TI calculators. The links you'll find online are broken, but thankfully the Wayback Machine has rescued the file:
https://web.archive.org/web/20061208153113/http://home.tele2.fr/grinstelbes/foblub-0.3beta.tar.gz

- **TI-nSpire**: it looks like someone has attempted a port of Frotz for these calculators, but I don't know if it works, and which devices (the first of the line, or one of the later models with a color screen?) it supports. Please report!
https://github.com/hoffa/nFrotz

- **V-Tech Helio**: this brand of PDAs have, you guessed it, their own port of Frotz:
https://inform-fiction.org/zmachine/helio.html

- **Windows Phone**: there a couple of interpreters specifically for that platform, although I would guess the Windows Store has (had?) more:
http://www.ifarchive.org/indexes/if-archiveXinfocomXinterpretersXzax.html
https://github.com/ActiveNick/TalkingAdventures

Bots for forums, Discord, IRC, etc

- **Discourse**: this is a forum platform, but you can run your own bot so you can play interactive fiction on a forum!
  https://github.com/merefield/discourse-frotz

- **Discord**: there are quite a few Z-Machine interpreter projects for Discord on Github, and I have not tested them. The French IF Discord server used a Python-based bot created by Nathanaël Marion (Natrium729), that you can find on his Gitlab:
  https://github.com/aeolingamenfel/discord-text-adventure-bot
  https://github.com/sponge/discord-frotz
  https://gitlab.com/Natrium729/xyzzanie

- **Facebook Messenger**: in theory, you can write and run bots for Facebook Messenger, but the rules have been tightened in the last few years, and I'm not sure this still works. In any case, here's a repository by AlexYucs:
  https://github.com/AlexYucs/ZFB

- **Lync / Skype for Business**: you can build bots for Lync, the live calling/messaging app, according to this repository:
  https://github.com/ericrrichards/LyncFrotz

- **ifMUD**: ClubFloyd is a fairly regular event gathering people on ifMUD to play games together in a MUD environment; more information here:
  http://www.ifwiki.org/index.php/ClubFloyd

- **IRC**: there are quite a few IRC bots you can find around – some written in Python, some in Rust, some in NodeJS... "grue" seems to be an old one (which means it potentially was used more), and there's a v3 one on Github:
  http://grue.sourceforge.net/
  https://github.com/ttencate/rustyknife

- **Slack**: I have found two repositories on Github to make your own Slack bot to play interactive fiction with:
  https://github.com/dmurvihill/frotzlack
  https://github.com/darryldunkin/slack-frotz

- **Telegram**: there are at least two Telegram bots that will allow you to play Z-Machine games, though I don't know how you set them up:
  https://github.com/eig114/frotzbot
  https://github.com/lbenini/node-frotz-telegram-bot

Other platforms

- **Dreamcast**: a port of Frotz was done by Sam Steele under the name FrotzDC. Various files are available online, but the very latest version (1.1) was, I believe, unpublished until I poked Sam about it. They can be found in the "releases" section of the Github repository:
  https://github.com/c99koder/FrotzDC/
  This interpreter supports v3, v5, and v8 files, and can save to the VMU; it has several color schemes, and the font is a C64 font that can be customised to your needs. I hacked it so latin-1 characters displayed properly, and it is available in that same repository. In order to create a disc file in the proper format, you need a few tools, and to follow the instructions at
  https://dreamcast.wiki/Creating_a_bootable_Dreamcast_disc

- **Embedded hardware**: what we think of "small embedded devices" has a lot of processing power, more than most computers of the 1980s. Someone made a v3 interpreter for such devices, and has a cool demonstration of it running in a keyboard – that's right, the keyboard's processor runs the interpreters, and fakes keystrokes to display the output in, say, Notepad. Cool stuff!
  https://github.com/daniel5151/embcrusted

- **FPGA / Verilog**: yes, there are hardware implementations of the Z-Machine! Should we still call it a virtual machine, or not?
  http://www.ifarchive.org/indexes/if-archiveXinfocomXinterpretersXhardware.html

- **LiveScribe pens**: these were "smart pens" that could digitize handwriting, and the video below caused quite a stir when it was published. I prodded the author of the code, and he kindly provided the source code on Github. It is of limited interest, since the "app store" for these has been disabled when the maker stopped producing pens.
  https://www.youtube.com/watch?v=scn3YdcFD60
  https://github.com/charcole/LivescribeZork

- **Mega65**: Ozmoo, by Johan Berntsson and Fredrik Ramsberg, is also available on Mega65 computers thanks to help from Paul Gardner-Stephen:
  https://github.com/johanberntsson/ozmoo

- **Spectrum Next**: the Next actually comes preloaded with the ZXZVM interpreter, thanks to Gary Lancaster, who ported it and fixed a few bugs to make sure it runs very smoothly. The distribution repository (which contains ZXZVM) is at:
  https://gitlab.com/thesmog358/tbblue

Open problems

Here are a few platforms which really could have a Z-Machine interpreter, but for some reason, don't.

- **Camputer Lynx**: it doesn't look like this 8-bit 1980s computer has an interpreter yet! It's based on a Z80 processor, which means the source for ZXZVM (for CPC and Spectrum) or ZEMU (for TI-84) could be useful.

- **Dragon 32 & 64**: these Welsh computers are supposed to be mostly compatible with the TRS CoCo, but I've had no luck running the Infocom interpreter on a Dragon emulator. Anyone wants to try?

- **FM-7** and **FM-Towns**: these two Japanese systems are respectively a 6809-based computer (which means that the source code for the Infocom TRS CoCo interpreter could be useful) and its successor, a x86/DOS-based computer, for which interpreters exist. If anyone with a fondness for old Japanese computers wants a side-project...

- **Intellivision with ECS**: the Intellivision had an extension that made it work like a home computer, and in theory it could run a Z-Machine interpreter. Somebody asked on the AtariAge forum:

https://atariage.com/forums/topic/268643-z-machine-wacky-daydream/

- **Matra Alice**: this is a rather odd computer, which sold in the early 80s in France; it is based on the 6803 processor, which makes it stand out from others, and only has 32 KB of RAM. I'm not even sure it's possible!

- **Super Nintendo**, **Genesis/Megadrive**: let's reignite the console wars of the 1990s! Sure, they don't have a keyboard, but it doesn't matter – the GameBoy doesn't have any either! And there is quite a bit of documentation available for making your own Genesis games...

- **Symbian / Nokia N-Gage**: there are a few homebrews for this, and who hasn't dreamed of playing interactive fiction on a T9 keyboard?

- **PC-88**: this platform technically has a Z-Machine interpreter: Infocom's own CP/M interpreter, which they used to release Zork (1, 2, and 3) on this platform. However, since the PC-88 is not a CP/M-based machine, they also had to reimplement CP/M on the PC-88 and bundle it with their game. I'm really not sure how to pry this out, and I haven't even been able to run a PC-88 emulator (the Zork disk image can be found on archive.org, however). If anyone has any ideas...

- **Sam Coupé**: a rather obscure machine, but a Z80-based machine nonetheless, so there is source code around that could be helpful!

- **Sharp X1**: a Z80-based machine for the Japanese market, but I believe it is rather obscure, unfortunately.

- **Sharp X68000**: this Japanese 16-bit home computer could easily run a Z-Machine interpreter, but there doesn't appear to be any so far. I know at least one person on Twitter who is eagerly expecting such a port!

- **Sinclair QL**: if it can run *The Pawn*, it could run Infocom's games, no?

- **Tape-based Spectrums**: this would be necessarily limited, since v3 games can be up to 128kb in size (and you have to add the interpreter on top); you wouldn't be able to run games that get too close to the limit, such as *Plundered Hearts*. But it might still be worthwhile: lots of games are smaller than 120kb, and might be played on a Spectrum 128K, for instance.

- **Thomson MO / TO series**: these were very popular in France, and are 6809-based machines, which means one could take inspiration from the source code of the TRS CoCo interpreter's source mentioned above. The latter models had disk peripherals, and there's quite a bit of RAM on a TO7/70 for instance; an interpreter would be feasible.

- **ZX80, ZX81**: this is highly speculative, since the memory limitations were tight, but apparently you could go up to 64kb of RAM with the ZX81. For the ZX80, however...

Phew! This was a very big article for ">REMEMBER"! Don't get used to it, I can't write 10 pages every month for you ;)
I hope this was interesting, and please report back with any experiments you do on any device! Send me a tweet at @hlabrande with a picture and I'll gladly retweet it!
See you next month!