Programming the ZX Spectrum Next

Personal experience of RCL/VVG

Speccy.pl Party 2025

Some context

- I am RCL of Virtual Vision Group (and also of Suspend on PC)
- Have been programming ZX Spectrum since 1994
  - Assembly programmer since 1995
  - …with some big gaps (coded for it in 1994-1999, 2005, 2012, 2024-...?)
- Grew up using the Spectrum clones of a “PC wannabe” kind
  - Didaktik Skalica, Profi+
  - TR-DOS
  - 1MB RAM
- Left the platform sometime in 1999 (and returned in 2024)
  - But haven’t quit demoscene and programming in general on other platforms

More context

- ZX Spectrum Next is a ZX Spectrum successor
  - I have seen many such attempts since the late 1990s, but this one is successful
- FPGA… but this is still a hardware platform
- 2MB RAM
  - Older models with 1MB are still around, albeit they are rare
- 28 Mhz
  - Optionally, but who wouldn’t turn this on?
- “Extended” Z80
  - MUL !!!   and some other convenient new instructions
- DMA, CTC, TurboSound (3x AY), sprites, tiles, 256 colors…

What is this talk about?

- I want to familiarize people with the platform
  - Assuming certain knowledge of Z80 and/or ZX Spectrum programming
- I want to share my personal experience
  - You should not treat this as an official programming guide
  - I have not yet mastered everything, I am not doing things “properly”
- But why?
  - To make ZX Spectrum Next releases numerous enough that they have their own compo instead of Wild :)
    - Thank you, Tygrys, for making a separate Next Demo compo at Speccy.pl Party!
  - I like the platform and want to see more sceners support it
    - On the other hand, do I need competition?...

What is out of scope for the talk?

- Why do people invent new things?
- Why Next and not XYZ (TS-Config, MB-03 etc)?
- Is FPGA a hardware?
- Is Next a retro platform?
- Who paid you to do this talk and how much?

Executable formats

- What you can ship
  - .NEX
    - Games, demos
  - .DOT
    - Intros
- There is more, but I did not need them
  - https://wiki.specnext.dev/File_Formats

- Next as a platform is **exceptionally friendly to the assembly programmers**
- It has two main formats for its executables
  - .NEX - similar to .SNA or .Z80
  - .DOT - similar to .COM from PC or CP/M
    - Max 8KB of code without any headers that is loaded under the address $2000 (8192) and executed
- You do not need any BASIC loaders (although BASIC programs are supported on the platform)
- .DOT format is just perfect for the size-limited intros (like .COM was under DOS)

NEX format

- For demos
- Easy to deal with
- Unsuitable for size coding

- NEX is more advanced than SNA (ZX Spectrum snapshot format)
  - It can contain a loading screen or binary data that is appended to the end and intended to be read manually
  - There is no shame in distributing your Next demo in .NEX, unlike distributing a ZX Spectrum demo in .SNA
  - Sjasm+ supports it starting with some early 2020s version

```
; start and stack addresses are given here
SAVENEX OPEN "helloworld.nex", START, $FF40
; core version (good to be backwards compat)
SAVENEX CORE 3, 0, 0
; border color and flags
SAVENEX CFG 7, 0, 1, 0
SAVENEX AUTO ; save all touched memory
SAVENEX CLOSE
```

- Read more here: https://z00m128.github.io/sjasmplus/documentation.html#c_savenex

- **HUGE FLAW**: minimal size is about 16KB
  - You want a smaller binary? Use .DOT

Compatibility

- By default, Next starts with ZX Spectrum screen and ZX Spectrum color palette
- Using ZX Spectrum bank switching (port $7ffd) is supported
- Next registers are accessible via ports, without needing extended Z80N instructions

- What's the easiest way to start programming for the Next?
- Replace DEVICE ZXSPECTRUM128 with DEVICE ZXSPECTRUMNEXT in your Speccy asm program and save your binary as .NEX
- If you are not using any ROM routines, it will work the same
- Video RAM layout is exact same (by default)
- Moreover, a regular Speccy program can detect that it is run on the Next and start using the new functionality (like 28Mhz):
  - See my example of doing this on github: https://github.com/RCL/ZXSpectrumNextEtudes/
- Porting regular ZX Spectrum (assembly) software to the Next is **very easy**!

ZX Spectrum Next software does not have to use new Next features and video modes.

It can also serve as a more convenient Pentagon successor with a true 50Hz refresh rate (or even a 60hz one).

New capabilities - memory

- Instead of $7FFD you can use 8 Next registers (available via ports too) to map 8 KB address space windows to banks
- Mapping change does not require trashing BC reg pair (nor even any register whatsoever if you know the bank number in advance).
- Interrupt vector table should be only 32 byte aligned and does not have to take 256 bytes

- It is even easier to work with the memory!

- You can map memory using 8 KB windows
  - You can leave a “kernel” and a stack in $E000-$FFFF addresses and switch the lower 56KB
  - Or vice versa, you can map video RAM to $E000, using the lower 56KB for your code and data
  - You can map various tables starting at $0000 (to simplify the lookup)

- Switching banks is fast and easy
  - nextreg <window_register>, <bank_number>
  - nextreg <window_register>, a

- There is *a lot* more memory (at least 1MB, but you can also target 2MB as it is the most common config these days)

- There is generally no slow or special memory - just like on the Pentagon

- Placement of the interrupt vector table is way less restricted

New capabilities - CPU

- Instructions to calc video address
- Ability to add 16-bit immediates to 16-bit reg pairs
- 16-bit shift of DE reg pair
- Unsigned 8 bit multiply
- And more: https://wiki.specnext.dev/Extended_Z80_instruction_set

- Writing Z80N code is easier (as far as 8-bit machine programming goes)!

- There are instructions helping with ULA screen layout
  - PIXELAD - HL = screen_addr(E, D)
  - SETAE - A = (0x80 >> E)
  - PIXELDN - HL = next_scanline(HL)

- Instructions adding 16-bit literals to HL i DE - greatly simplify walking up/down the scanlines of a linear screen
  - ADD HL, 32
  - ADD DE, 2
  - ADD DE, A

- Shifting DE without knowing the amount in advance
  - BSRA DE, B - DE = (DE >> B)

- Unsigned multiplication DE = D*E
  - MUL

- Test (non-destructive AND)
  - TEST $7

Ok cool, but how do I try a Next program?

And this is the largest pain point for the Next at the moment.

Emulators

- Only two emulators practically usable by an average human being. Pick your poison:
- CSpect
  - https://mdf200.itch.io/cspect
- ZEsarUX
  - https://github.com/chernandezba/zesarux/releases

- CSpect
  - Pros
    - The best support for Next features
    - Integrated with the DeZog debugger
    - Mac and Linux support (allegedly)
  - Cons
    - **Obfuscated and unsigned** closed source, so it gets routinely flagged by the antiviruses - **HUGE FLAW**
    - Very restrictive license that forbids redistribution **- HUGE FLAW**. I'd like to include a zip with the "turnkey" emulator setup here, but I cannot.
    - C#, so the performance is uneven
    - AY emulation issues
    - UX is lacking
- ZEsarUX
  - Swiss army knife that doesn't call the Next by its name
    - (look for TBBlue)
  - UX is more advanced, feels like a 1994 era
  - I did not use it in anger, don't really have an opinion

Emulation summary

- You need to make a system image
- “Proper” way is to put your files into that image and run from there
- Running “bare” NEX is technically only seen by the emulator authors as a quick test for standalone, self-contained files.

- Setup is a (minor) pain point.

- I did it once and keep copying between my systems (but cannot attach here!)

- I need to revisit it to support running files using the NextOS, but there’s always something more important than the “correct” setup, as it *mostly works* for my use cases

- Don’t sweat it - you won’t need it until you start writing a game or other software where you need to read/write to the disk.

- I just run CSpect and give it the NEX file on the command line.

- **An issue** (not a huge but notable): you cannot run .dot files that way. I work around that by saving DOT files also as NEX (with a different origin) for debugging, but there are some differences which I have ran into myself.

Time to stop scaring people and show how running and debugging Next software works in practice (hint: it’s not too bad).

There was a live presentation of running a ZX Next emulator and converting a program from ZX Spectrum to ZX Spectrum Next, as well as debugging it. Some screenshots from that are below.

Performance aspects

- Don’t get carried away, it’s still a 8-bit machine. 28Mhz sounds like a lot, but this is the same slow (4 cycles for a NOP) 8-bit instructions with 64KB address space
- I expected wonders when porting my raytracer from the classic Speccy, but even with 28Mhz and hardware MUL it was only 9x faster, instead of 40-something sec taking about 5 s.

- 28 MHz sounds like a lot **but it is just 8x faster** than the 1982 Speccy. Larger Video RAM sizes quickly reduce even this modest speed advantage
  - With a 6912 or 6144 byte screen you can indeed feel a nice speed increase
  - With a 12KB screen, especially split into parts, this feeling is diminished
  - With a 48 KB (256x192x8bpp) or 80 KB (320x256) screens, we’re back to that outmatched CPU feel.

- Next has **supporting video hardware**, but it is limited to classic 2D functionality:
  - Sprites
  - Tiles
  - Overlay / scanline effects

- For per-pixel work (3D, plasmas / roto zoomers) you should look into **128x96x4bpp mode** (LoRes / Radastan), which is easy to work with and takes just 6144 bytes.

- You can also **just use the classic 6912 byte screen**!

60 frames per second

- Next supports 60Hz. This is a more future-proof refresh rate, even in Europe.
- You can programmatically know if you're running 50 or 60Hz, so it is worth supporting both.
- Skipping each 6th music/logic update is an acceptable solution for a lot of cases.

- Be mindful that the Next can run **with a 60Hz refresh rate**.
    - IMHO we should get used to thinking it *will* be the default, if not now then in 2030
    - 50Hz support is waning, especially among the TVs
    - It'd be nice if our prods still worked well for the future demoscene museums

- The easiest way to support 60Hz is to skip every 6th main loop iteration / interrupt.
    - It can distort the music somewhat, but it is usually not noticeable especially if you don't know

- You can also set a 50Hz timer interrupt and update the music and logic in it, while keeping the drawing in the vblank interrupt
    - It sounds complicated but is not that hard in practice, however it is more work than skipping every 6th frame

- Last but not least, keep in mind that the number of CPU cycles per frame in 60Hz is **some 17% lower** (it drops from **560k** to about **467k** in 28Mhz)

Advanced functionality

- DMA
- CTC interrupts
- All the rest

- DMA is well documented in Next docs. There’s only one channel and you can treat it as a **faster LDIR**. I have not yet used DMA in my prods, only tested it in sample snippets.

- It can work in the background (burst mode) **but does not have an interrupt on completion** (you need to poll). This is mostly geared for digital audio playing and it slows the CPU down (things work like with the chip RAM on Amiga)

- CTC is a **timer** that can generate interrupts at a given frequency (a divisor of the system frequency). It is rather simple to program, but you’ll need to remember that HALT will no longer mean “wait for a new raster scan” -> you’ll have to invent a way to check that vblank interrupt was handled.

- I have not yet used scanline interrupts, sprites, tiles, copper etc. You can read about them in the docs where they seem well described.

- Emulators can have issue with these things, but CSpect seems to be past the teething period in this regard

Helpful links

- https://wiki.specnext.dev/
  - Wiki covering various technical topics
- https://github.com/tomaz/zx-next-dev-guide
  - A more official guide - much recommended!
- http://ped.7gods.org/Z80N_table_ClrHome.html
  - Table of all the Z80N instructions
- https://github.com/RCL/ZXSpectrumOpenSource/ oraz https://github.com/RCL/ZXSpectrumNextEtudes
  - Sources of my Next demos, and also some snippets
- There's more Next code examples from **taylorza** and many others - just google it
- https://discord.gg/UXVHCxuAWq - English language ZX Next Discord server

Next has larger expressive power than the classical Spectrum.

Nevertheless this is still a 8-bit, bare metal machine that you can completely control

I hope you will have fun programming it like I did!

Questions or comments

Discord: RCL1024

Mastodon:

rcl@mastodon.gamedev.place