THE MAKING OF **PIMP MY SPECTRUM**

BY EVILPAUL OF ATE BIT

zine BEHIND THE SCENE

There are two questions about Pimp My Spectrum that I get asked a lot:

1. Is it just an emulator?
2. Could it run on a real Spectrum?

Hopefully I can give you the answers to both.

INITIAL CONCEPT

The initial concept for the demo came to me at Breakpoint 07. It seemed like a lot of people were talking about virtual machines for their 4k demos, i.e. inventing your own machine that can handle a small number of powerful, handpicked, instructions and then writing a virtual machine to interpret them. An alternative is, of course, to take an existing machine - in my case the ZX Spectrum - and virtualize it to create what is, essentially, an emulator. Only a small subset of the entire machine needed to be modelled to give it enough power.

PROOF OF CONCEPT

By the time I got home from the party I already had a basic, proof-of-concept demo written. Using SDL and running under OS X I had created the most basic of Z80 emulators. It only handled the most essential registers and flags and could only execute a few dozen of the most standard instructions. It also ignored clock cycles and ran as fast as it needed. Just like a real Spectrum it had 64k of RAM and the memory starting from 16384, was used to generate the screen display. This was done by converting the video memory once per frame and uploading it to an OpenGL texture. This was then displayed on a single quad.

I was then able to write code in a text file, compile it using Pasmo, load it into the project, and watch it run. I added a few basic hooks by hijacking the RST instruction so that whenever an RST was executed, some C code would be called, instead of following the normal Z80 behaviour of jumping to a specific memory address. I wrote a couple of small functions - get sin, multiply, lerp - using C, as well as some sprite and 3D routines. I could make them much more powerful, flexible and with much less effort than writing them in Z80, which would have been a real headache.

The graphics routines would read data from the emulated machine's RAM and write their results directly back into screen memory. The 3D routines handled the translation and lighting as well as the sorting, culling, clipping and rendering of the polys - much like a modern GPU would. They also had their own local memory for temporary storage of the current scene.

It worked, but it would be difficult to fit into 4k and make it a good enough demo to compete. So, I decided to leave the idea as "cute, but not worth the effort".

HIATUS

I have always loved the 64k scene. From Paper and Clone Meets Clone through FR-08 to Dead Ringer and Chaos Theory, it has always been my favourite category. However, with a handful of groups able to produce amazing demos seemingly at will and no one else coming close, it has been less than interesting of late. When I heard that the big boys of the 64k scene had decided to move on to other things I expected the competition at Assembly '07 to be a showcase of groups that had been waiting in the wings. Disappointingly, the category attracted minimal entrants and as a result the feeling all around was one of collective ennui.

It was then that I recalled my little 4k curiosity. What if I expanded it into 64k...would it be enough to kick some life back into my beloved competition? And so the pimping started...

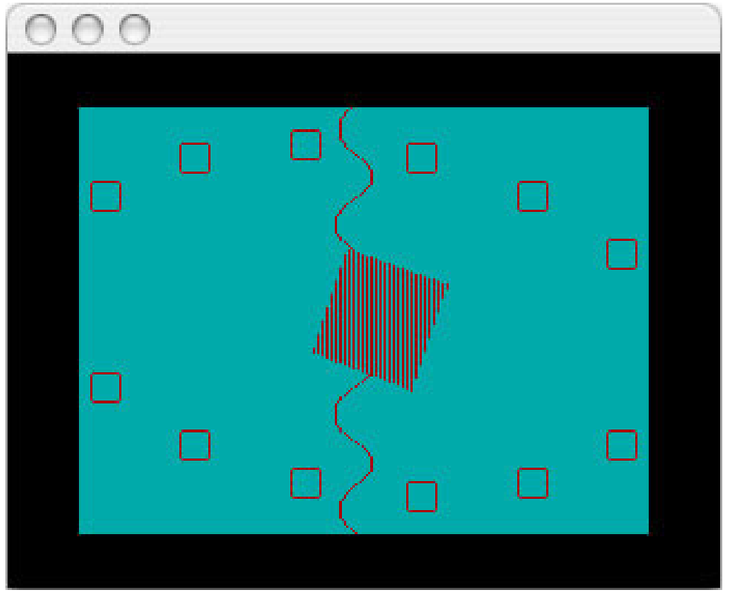
GETTING BACK TO THE PROJECT

I tinkered with the project over the next few months but didn't give it much focus until February. Firstly, I ported my work-in-progress over to the PC. Then I added in a cut down version of VTXPlay to handle the audio. This whole process was fairly painless and after KKrunchy had done its job, the first functional test code turned out to be nicely compact. I now had most of the major functionality in and working, yet still had plenty of space left for data and Z80 code.

> **WHAT IF I EXPANDED IT INTO 64K...WOULD IT BE ENOUGH TO KICK SOME LIFE BACK INTO MY BELOVED COMPETITION?**

Next I tweaked the Z80 core a little by adding a few non-standard instructions of my own. Alongside the familiar PUSH, I also added PUSHNN to directly stack a value and PUSHCNN to stack the contents of a memory address. Little changes like this made writing the effects less painful because I didn't have to juggle registers so much. I also started writing my Z80 in a more lazy way - when you don't have to worry about clock speeds things get much easier.

In all of my standard Spectrum demos I have used the same custom sprite and data packing tool. This demo was no exception. I also wrote a simple 3DS Max exporter to allow me to output mesh data in a format that my GPU could read. The packed data that was created by the tool was used as a file system. This was accessed via an RST instruction - the Z80 code would request a file by an ID and the C code would copy the data into the emulator's RAM.



A screenshot of a very early build of Pimp My Spectrum

I also had to start writing the actual demo. By this stage I already had a concept I was happy with and a few of the scenes, so most of the work now would be the implementation. Mofobaru had agreed to provide the tune (despite having to work with Vortex Tracker's slightly bizarre method of creating six channel tunes!) and my coder-art was being tidied up and replaced

by two artists at Denki, where I work. At short notice I managed, to my great pleasure and surprise, to get Diver/4D to help out with a full colour original picture - something that not even one of my Spectrum demos has ever had. The two other pictures in the demo were created with a convertor program - the difference in quality is huge.

My wife and I had a holiday planned for the end of March and I was looking forward to the break, even though I was a little stressed about how much work I had to do on the demo when I got home.

FINAL PUSH

Back from holiday and without a Windows laptop to take to the party, I now had four evenings to get the demo finished. Luckily, jet-lag seemed to have messed up my body clock enough that I didn't feel I needed a lot of sleep in those four days.

I NOW HAD FOUR EVENINGS TO GET THE DEMO FINISHED

I got the first complete version of the tune that Mofobaru had done and started hacking the music player to play six channels instead of three. I chased the artists for their work, getting most of it the day before I left. I failed to fix a bug in the 3D routines, so I had to push the objects a lot further away from the camera than I was planning to. This now meant I had to redesign a couple of the scenes. I added the transitions between the effects and added in some code to correct the aspect ratio, which worked fine on my home machine (luckily the same resolution as Breakpoint's massive projector screen) but annoyingly failed on other monitor sizes.

At 4am on the Friday morning I climbed into a taxi, tired from coding but with the demo finished...except I was still waiting for the final piece of artwork from Diver.

I finally got the picture on Saturday night and it was perfect. After a quick plea for help on the party IRC channel I was offered use a sleeping man's PC by Raceend of HCBA (thanks!) in time to make the final build.