(Super)CHIP 8 Secrets

Edit New page

Petr Kratina edited this page on Jan 2, 2014 · 1 revision

During emulator development I have encountered few "WTF?" moments that are not documented enough elsewhere so I will share my solutions with you:

Compatibility mode (SAVE / RESTORE)

In available documentation for SAVE/RESTORE opcodes is written that I register should be increased by amount of saved data (this differs many times in many documents). My observations are that by default this behavior should be disabled by default. There is however chance that some programs might rely on this behavior so here is special non-official instruction that will set interpreter to "compatibility" mode. Opcode for this instruction is 0x00FA and is documented (as far as I know) only in Peter Miller's reference manual: http://chip8.sourceforge.net/chip8-1.1.pdf

Super CHIP's display memory dilemma

There are two approaches how to handle rendering of two available resolutions:

1. Render modes separately (2048 pixels in basic, 8192 pixels in extended mode)
2. Render 8192 pixel every time, when in basic mode, pixel coordinates are doubled when drawing and pixel is 2x2 pixels in size. I have used first approach since it makes more sense from my point of view, but that reveals some

Pages 2

Find a page...

Home

(Super)CHIP 8 Secrets

+ Add a custom sidebar

Clone this wiki locally

https://github.com/AfBu/haxe-ch

incompatibility issues, especially when using emulator test programs - for example HAP's emulator test should display equal sign (=) in the bottom left corner and then scroll screen one pixel down. By HAP's description, bottom line of equal sign should now be half pixel high. But we are in 64x32 mode and by logic, there is nothing like half-pixel. So I have sticked to first solution but I am still searching for evidence which approach is correct.

Speed of emulation

When you are writing emulator, you have original CPU speed as a reference in most cases, but since CHIP-8 is interpreted language, speed varies based on device program was designed for. By my observations best universal speed for CHIP-8 programs is 500Hz and for SuperCHIP it's 1000hz, but you have to give user ability to change it so their experience is as good as possible. Also don't forget that delay and sound timers should always tick down at 60Hz, no matter how fast emulator is running.

Understanding of "Store BCD" instruction

I and I believe many other beginners have had quite hard time understanding that this instruction do and how it really do it. So here is my lame explanation:

- You have value 0x94 in register V0, that is 148 in decimal representation.
- You call Store BCD instruction.
- Memory will change to this: M[I] = 1, M[I+1] = 4, M[I+2] = 8 My lame solution of how to achive this is:

1. Divide number by 100 and floor it. It will give you fist digit.
2. Divide number by 10 and floor it. (you will get 14 in our case) and then modulo by 10. It will

result in 4 as second digit.
3. Modulo number by 100, which will give you 48 and then modulo by 10 to get final digit of 8.
4. now write those numbers to memory starting at location defined by I.

Final tip about emulation speed

My first attempt was to create thread with loop inside. Stepping one instruction by cycle and then sleeping few milliseconds. However this approach show itself at not-very-clever since sleep instruction does not always gives you accurate enough result and based of target platform you can easily reach it's precision limit. So it's much better to run our loop at much lower speed (100Hz in my case), calculating time between two loop cycles, then based on target frequency calculate number of operations that should be performed and perform them at once. This will give you much more precise control over emulation speed.