Chip8 Emulator für ZX Spectrum 48k

Die frühen Videospiele wurden in einer virtuellen Maschine programmiert. Sie liefen auf einem Prozessor mit 4 KB Ram, wovon die ersten 512 Bytes das Betriebssystem beinhalten. In diesem Bereich befand sich der Interpreter und einige Tools, die man aufrufen kann (z.B. Bildschirm löschen).

Die Grafikauflösung betrug 64x32 Pixel in schwarz/weiß, dies entspricht der Grafikauflösung des ZX81 wenn man mit dem Block-Zeichensatz zeichnet. Im Prinzip könnte man den Emulator auch auf dem ZX81 zum Laufen bringen.

Die Tastatur war eine Hex-Tastatur mit Tasten 0..9 und A..F. In den Spielen kann man mehrere Tasten gleichzeitig drücken, zum Beispiel Laufen und gleichzeitig Springen oder Schießen. Dies ist auch in diesem Emulator möglich.

Man programmierte auf dem Papier, gab dann Code ein konnte auf Kassette speichern. Die Geräte kosteten ca. 99$ wenn man sie selbst zusammengelötet hat und waren daher ungefähr in der Preisklasse des ZX80 aber einige Jahre vorher in den USA erhältlich.

Bekannt wurde das System wieder, nachdem es einen Chip8-Emulator für den HP48 Taschenrechner gab. Hierfür wurde Super-Chip8 erfunden, die Auflösung beträgt 128x64 Pixel, immerhin fast die Hälfte der Grafikauflösung unsers ZX Spectrums. Später kam die Octo-Erweiterung hinzu mit bis zu 4 Farben, was sich leider nicht auf dem ZX Spectrum umsetzen lässt die letzte Entwicklung ist Mega Chip8 mit 256x192 Bildpunkten in 256 Farben. Dies ist nicht nur mehr als unser Speccy anzeigen kann, es benötigt auch 48 KB nur für den Bildschirm.

Der Emulator emuliert Chip8 und Super Chip8 mit einigen Octo-Erweiterungen.

Es gibt einige recht kurzweilige Spiele und Emulatoren für diverse Computersysteme. Mit Rush Hour kann man sich viele Stunden beschäftigen und die komplizierten Rätzel lösen, mit Cave Explorer kann man eine große Landschaft erkundigen, außerdem gibt es Mine Sweeper, Tetris und andere.

Um Chip8 herum gibt es eine aktive Szene, d.h. es werden laufend neue Spiele entwickelt und neue Emulatoren. Jedes Jahr im Oktober findet der Octo-Jam statt, Programmierer stellen dort ihre neuesten Spiele vor.

Es ist relativ einfach Spiele in Chip8 zu programmieren und gleichzeitig zu lernen, wie man in Assembler programmiert, da es nur eine kleine Anzahl von Befehlen gibt und einen sehr begrenzten Funktionsumfang.

Auf der Webseite <http://johnearnest.github.io/Octo/> kann man den Compiler und den Emulator ausprobieren. Spiele, egal ob im Quellcode oder kompiliert kann man einfach mit der Maus auf den Editor schieben.

Chip8 ist auch eine Spielwiese für jeden, der einmal einen virtuellen Computer samt CPU, Display, Keyboard programmieren möchte. Der Emulator für den Spectrum besteht aus ca. 6000-7000 Zeilen Z80 Assembler, der größte Teil davon Handoptimiert. Der Code verwendet keine Routinen aus dem ROM, lediglich der Zeichensatz stammt aus dem Rom.

Der größte Teil der Optimierung bezieht sich auf die Darstellung, wenn Chip8 einen Sprite zeichnet muss dies sehr schnell passieren. Je nach Auflösung werden pro Chip8 Pixel 1x1, 2x2 oder 4x4 Pixels auf dem ZX Screen gefüllt. Sprites bestehen aus mindestens einem Byte (8x1 Pixel) und maximal aus 16x16 Pixeln.

Der Emulator zeichnet entweder jede Änderung sofort, was positiv ist, wenn man debuggt oder den gesamten Screen im vertikalen Interrupt.

Die Chip8 Opcodes werden direkt interpretiert, dadurch ist der Emulator sogar so schnell, dass man ihn für einige Spiele verlangsamen muss, sonst werden sie unspielbar. Trotzdem möchte ich mehr Speed und mehr Lernen.

Mein Chip8 Disassembler kann die Ausgabe direkt in Z80-Assembler umwandeln, man benötigt noch eine kleine Laufzeitumgebung und kann dann das Spiel direkt Maschinencode umsetzen

Spiele:

Es gibt hunderte Spiele, die man sich herunterladen kann. Siehe Liste im Anhang.

Um die Speile ohne einen Assembler in das Programm zu bekommen, gibt es das Java-Programm chip8tapegen. Es gibt 3 Orte für Spiele:

1. Das into. Hier startet normalerweise das Demo welches am Start gezeigt wird, man kann es aber mit einem Spiel überschreiben. Die Datei startgame.bat erstellt ein neues Tape mit dem ersten Parameter und startet es.
2. Der Speicher. Zwischen dem Ende es Emulators und dem Ende des Speichers ($FFFF) können Spiele geladen werden.
3. Auf Band. Mit dem Befehl laden aus dem Hauptmenü wird der Emulator verlassen und das nächste Spiel von der Kassette geladen.