

Z80 VM for iOS Games

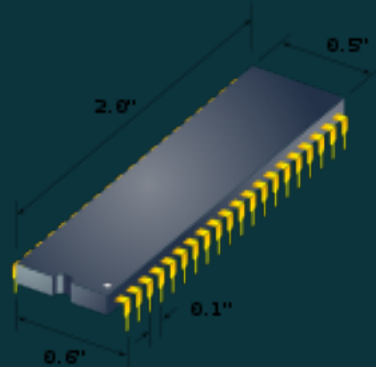
Scott Lawrence
yorgle@gmail.com

Presented at
BarCamp Rochester
October 23, 2010

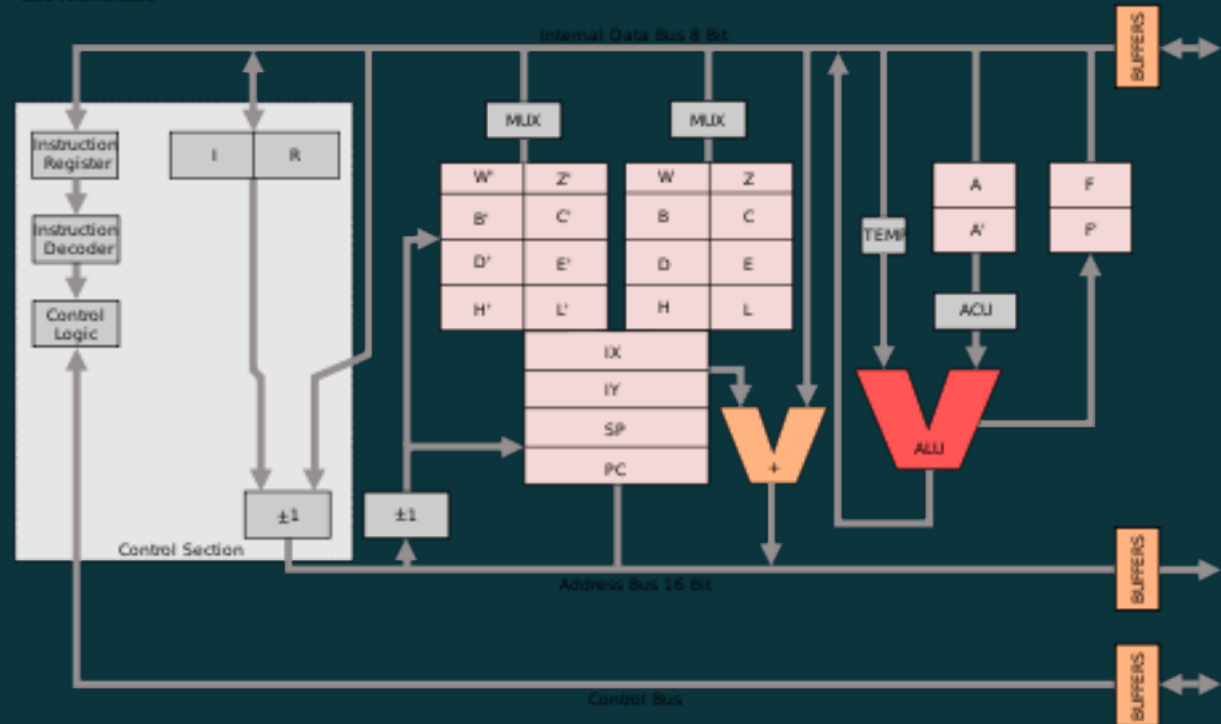
Z80 Basics

- 1976 by Zilog - 8-Bit processor (similar to i8080)
- Mfg by Zilog, NEC, Sharp, Toshiba, etc.
- 16 and 8 bit registers, 64k addressable, Port IO
- Used in MANY games in the 80s:
 - Pac-Man, Pengo, Crush Roller, Zaxxon, Xevious, Ataxx, Galaga, Scramble, Time Pilot, ...
 - GB*(Sharp), GameGear, Genesis

27	HL	A0	30
		A1	31
19	MREQ	A2	32
20	I/OREQ	A3	33
21	MR	A4	34
22	RD	A5	35
		A6	36
28	REPSH	A7	37
		A8	38
18	HALT	A9	39
		A10	40
24	WAIT	A11	1
		A12	2
16	INT	A13	4
17	NMI	A14	5
		A15	6
26	RESET		
		D0	14
25	BUSRD	D1	15
23	BUSAK	D2	12
		D3	8
6	CLK	D4	7
		D5	9
11	Vcc	D6	10
29	GND	D7	13



Z80 Architecture



Why not just an emulator?

- Pac-Man screen layout (and descendants) is messy
- Sound emulation is intensive and inaccurate
- CPU timing might be inaccurate
- Dealing with tileset + color mapping in GL
- Screen resolution means weird scaling (1.5x)

So...

let's make a "virtual machine" that suits our needs better

The Virtual Machine

Virtual Machine - "Emulator" without baggage

Making a fictitious platform affords us some flexibility

Emulation of a specific platform implies certain behaviors, expectations; sets yourself up for failure to achieve "perfect emulation"

"The perfect is the enemy of the good." - Voltaire

4 Components

- CPU - Many available, mostly GPL or MAME license
 - Z80pack -- BSD license
 - <http://www.unix4fun.org/z80pack/>
 - Uses globals, so 1 CPU only for now
- Screen
 - BL2D tile/sprite library (MIT License) (OpenGL ES 1.1)
 - <http://github.com/BleuLlama/BL2D>
- Audio
 - Simple layer on top of iOS frameworks for SFX+Music
- Input
 - Use iOS-specific inputs (touch, tilt, not joystick or button)

Comparison of architectures

(Ms.) Pac-Man

0000-3FFF 16k ROM (pac)
4000-43FF tilemap bg
4400-47FF tilemap color
4C00-4FEF ~3k User RAM
4FF0-4FFF Sprite RAM
5000-503F IO
5040-40c0 Audio
8000-9FFF 8k ROM*
A000-BFFF 8k ROM**

* Ms.Pac only, patched

** Ponpoko only

LlamaCore VM

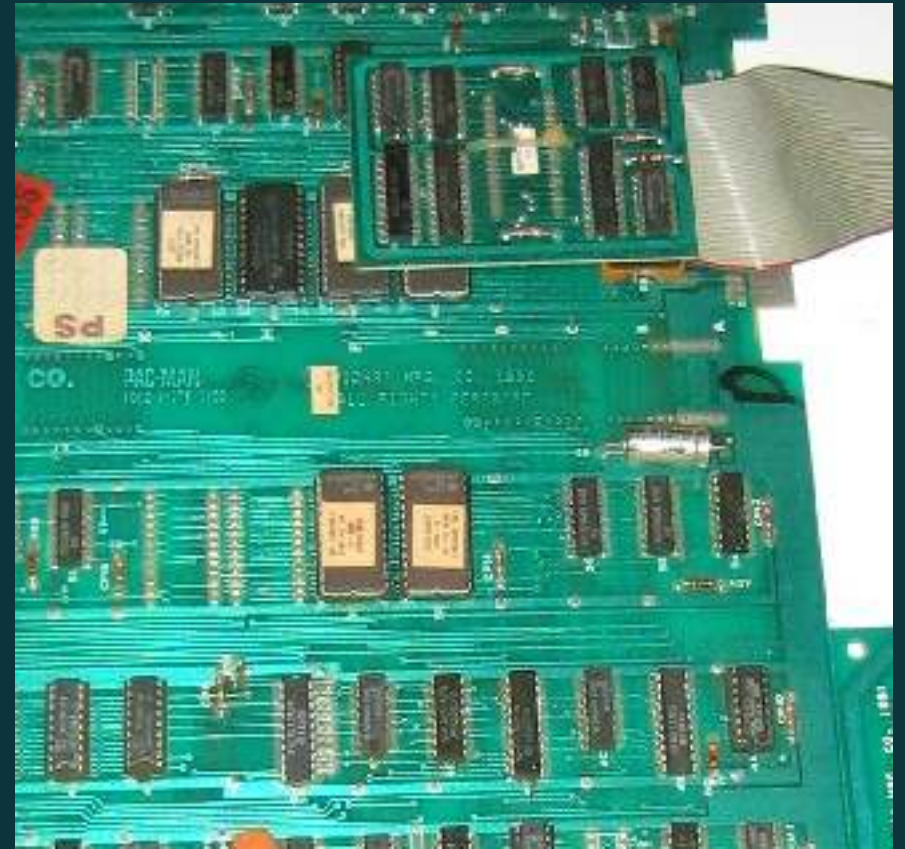
0000-BFFF 48k ROM
C000-CFFF tilemap bg
D000-DFFF Sprite RAM
E000-EFFF 4k User RAM
F000-FFFF System

Example: Loading graphics tilemaps

Ms. Pac-Man



ROMs



Example: Loading graphics tilemaps

OpenGL

```
- (void)loadPng:(NSString *)fn
{
    glEnable(GL_TEXTURE_2D);
    glEnable(GL_BLEND);
    glBlendFunc(GL_ONE, GL_SRC_COLOR);
    // make texture name
    GLuint texture[1];
    glGenTextures(1, &texture[0]);
    glBindTexture(GL_TEXTURE_2D, texture[0]);
    #ifdef kPixellyScaleups // for pixelly scale-ups
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST);
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_NEAREST);
    #else
    // configure (for smooth scaleups)
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
    #endif
    // load image
    NSString *path = [[NSBundle mainBundle] pathForResource:fn ofType:@"png"];
    NSData *texData = [[NSData alloc] initWithContentsOfFile:path];
    UIImage *image = [[UIImage alloc] initWithData:texData];
```

... 2 more screens of this...

Limitations of a VM

- 48k of ROM space - (bank switchable?)
- 4k of RAM
- Write in Z80 ASM, (or bloated Small C)
- No IDE
- Slow CPU
- 2D only
- Single Screen

Who cares about the limitations?

- It's fun to write
- It's fun to have these limitations
- The games will have a retro look and feel
- The "single screen" game aesthetic
- Interface is much simplified for 2D games

The irony

Using the invented VM to simplify -- had to create an abstraction layer to simplify GL for the VM.

Demos

some demos...

Thanks!

Scott Lawrence

yorgle@gmail.com

<http://UmlautLlama.com>

<http://github.com/BleuLlama>