CHIP-8 Emulation on CY8CKIT-059 PSoC Kit

6.115 Final Project Proposal Draft

Keshav Gupta (keshav21@mit.edu)

April 8, 2019

1 Background and Introduction

CHIP-8 is a simple interpreted programming language. It was first developed for the COSMAC VIP computer to make programming games for it easier. It supports a 64×32 monochrome display, 4×4 keypad input and a monotone buzzer for sound output. Despite it's limited hardware resources, it proved to be very easy to write games in, resulting in interesting implementations of Connect-4, Blink, Pong and many more classics.

For my 6.115 Final Project I aim to build a CHIP-8 emulator based around the Cypress PSoC provided in class. The interfaces to/from the system will be a VGA output for video, a generic buzzer for sound, and the official 6.115 keypad for input.

The inspiration for this project comes from the final project idea about Atari 2600 emulation on the class web page, which prompted me to read about console emulators. With some research I determined that CHIP-8 was a good starting point for emulation since it hits the right simplicity-to-usefulness point.

2 Hardware Description

The CY8CKIT-059 PSoC Kit provided in class will be used as the central processor. The display module is controlled by an SH1106 driver, which will interface with the central PSoC over SPI. For the input, the official 6.115 keypad driven by a 74C922 will be used.

No external memory will be needed, since only 4KB is used by CHIP-8 which will reside on the PSoC RAM. Sound generation and in-game timing require a 60Hz interrupt, which will be provided by an on-board timer.

Three additional output pins will be needed for the VGA interface, two for `hsync`, `vsync` and one for `red`, `green` and `blue` (note that the display on CHIP-8 is monochrome, so 1-bit color suffices).

Please refer to Figure 1 for a diagrammatic representation of system hardware.

3 Software Description

During initialization, the software copies initial ROM contents from flash. Since game ROMs are typically much smaller than 4 KB, multiple of those can be stored in the program flash and the user can chose which one to load.

Once initialized, the software would run a fetch-decode-execute-write back loop. The instruction word (16 bits wide) is fetched from memory, decoded into components, executed and the result is written back into the memory. This process is simplified thanks to fixed length instructions (total of 36) on the platform.

I will use Thomas P. Green's [1] CHIP8 reference for instructions.

Please refer to Figure 2 for a diagrammatic representation of system software.

4 Project scope

My goals for the project are divided into three levels:

- Safety: Minimum goals required for my project to be considered functional. I consider these to be delivering a working version of my project on a breadboard with video output on at least a 1.3" OLED.
- Match: A reasonable expectation from the project. I consider these to be delivering a pleasant user experience and putting out video via VGA.
- Reach: Upon completion of the Match goals, I will push towards designing a portable, permanent version of my project on a PCB.

5 Special components needed

Major components that I will need:

Item	Status
Cypress PSoC Kit	Available
6.115 Keypad	Available
IC - 74C922	Available
Buzzer	Available
IC - LM358	Available
VGA Connector (Female)	Ordered on Amazon [2]
Assortment of headers, IC sockets, resistors, ...	Available
1.3" OLED (as a backup)	Available

6 Timetable

Week	Agenda
April 15 - April 21	Write and debug the emulator
April 22 - April 28	Write and debug the emulator (contd.)
April 29 - May 5	Work on VGA
May 6 - May 12	Design, mill and solder PCB
May 13 - May 16	Finish Lab Report

7 References

1. devernay.free.fr/hacks/chip8/C8TECH10.HTM
2. www.amazon.com/Twinkle-Bay-Terminal-Breakout-Connector/dp/B07F9QFMKN

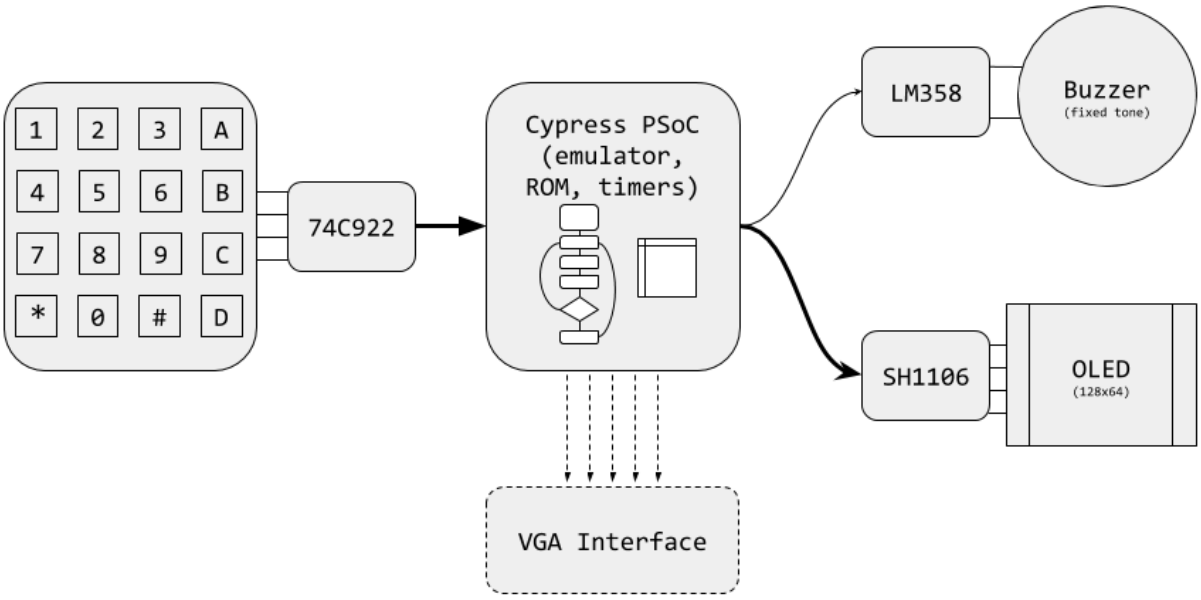

Figure 1: System Hardware Diagram

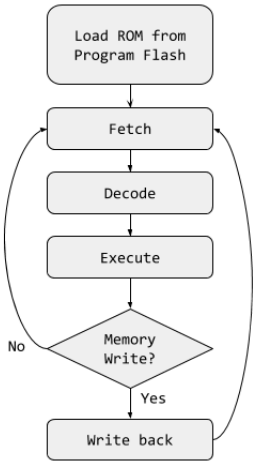

Figure 2: System Software Diagram