IOWA STATE UNIVERSITY
Electrical and Computer Engineering

Senior Design Poster Presentation Fair
4/22/2019

Senior Design sdmay19-25

Members: Nic Losby, Sean Hinchee, Jacob Nachman, Nick Lang, Matthew Kirpes, and Faculty Advisor Dr. Julie Rursch

Handheld Emulation Station

Introduction/Summary:

The Handheld Emulation Station is a student designed and proposed project. We enjoy playing retro games and wanted to create a better product than those currently available on the market. Our goal was to create an Emulation Station that focused on mobility, battery life, and user experience. Our emulator is based on a Raspberry Pi Zero with a 3.5 inch LCD screen and a 3000mAh battery. This project allows us to utilize the skills we learned throughout our academic career by designing an embedded system from scratch.

Architecture Design

- PCB
  - Buttons
    - Used to operate and perform functions on the emulator
  - USB 2.0
    - Used for peripherals and supports external storage
- 3000 mAh Battery
  - Powers the system for at minimum of five hours
- Raspberry Pi Zero
  - Computer system that runs the software
  - Input given from PCB
- Kernel Module
  - Abstraction of gpio pin input state to /dev file system
  - Closer to real-time input than providing buttons over USB
- Emulator Software
  - CPU cycling supported with 510 opcodes (255 regular, 255 CB)
  - Cartridge Loading supported
  - Reading and Writing to Memory
  - Screen Rendering functionality
- 3.5 inch LCD Display
  - Renders the data from the GPU onto the screen

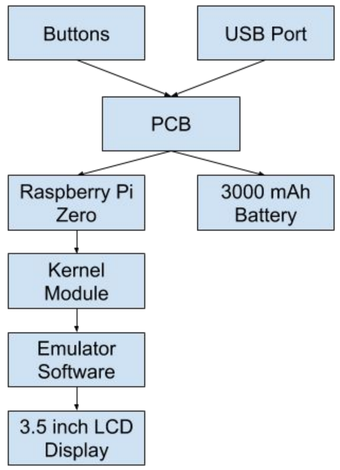

Figure 1
High Level System Architecture Design

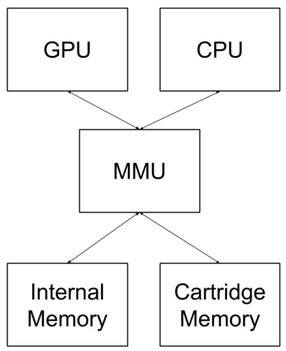

Figure 2
Emulator System Architecture Diagram

Implementation and Design:

GPIO Pin Number	Dev File	Peripheral
03	/dev/rt0	Y-axis Joystick
05	/dev/rt4	D on D-Pad
07	Reserved	Speaker
12	/dev/rt1	X-axis Joystick
13	/dev/rt2	U on D-Pad
15	/dev/rt3	R on D-Pad
16	/dev/rt5	L on D-Pad
29	/dev/rt12	L trigger
31	/dev/rt13	R trigger
33	/dev/rt6	X
34	/dev/rt7	A
35	/dev/rt9	Y
36	/dev/rt8	B
37	/dev/rt11	Start
40	/dev/rt10	Select

Figure 3
Kernel Module System Layout

Emulator

Our emulator was implemented in the GO programming language. It has a fully functioning CPU with 510 implemented opcodes, working memory that can be read and written from, functional screen rendering, partially functional GPU and the ability to make save states.

PCB

The PCB started with a layout of a Raspberry Pi Zero W and aligning it by the GPIO pins and mounting holes. The corner where the battery would have attached has been carved out. The extra space for a speaker has been reserved for future work.

Functional/ Non-Functional Requirements:

Functional

- Faithfully emulate multiple retro systems
- Load and save games
  - Cloud backup support

Non-Functional

- Portable
- Low latency
- Battery life > 5hrs

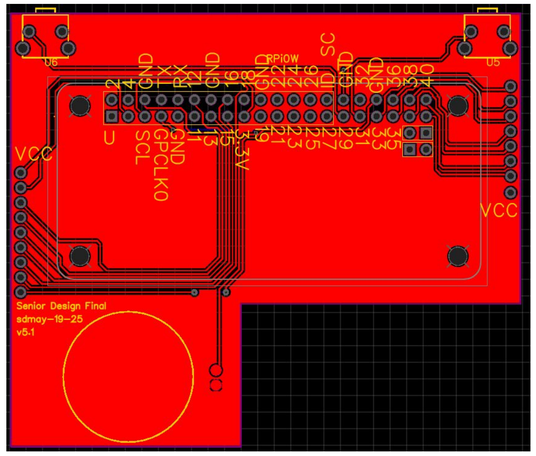

Figure 4
Main board/Motherboard

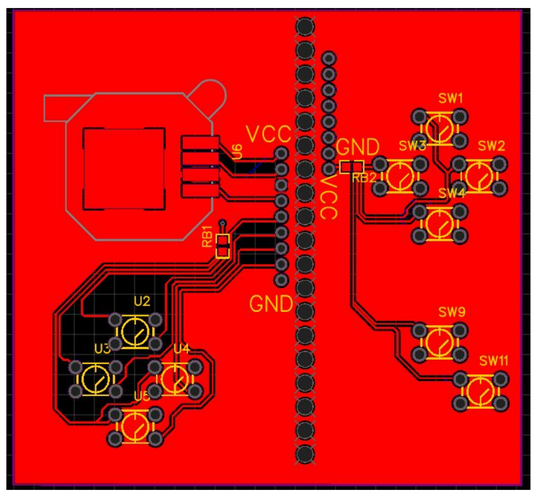

Figure 5
Daughter input boards

Testing and Evaluation:

Hardware and Software Testing

- The PCB was tested via a trial and error testing
- We used a multimeter and a bench power supply in our testing. The multimeter was used for testing the battery voltage, current draw, as well as continuity testing. The bench power supply was used to simulate a battery for testing the circuit

Functional Testing

- The emulator was tested using unit tests available in the Go programming language toolkit program, test. Each CPU opcode type has its own test to validate its accuracy
- Testing using the test package within Go can create unit tests for internals and externals at once as the tests run within the package being tested

Standards/Best Practices:

Emulator

- CPU instruction set. This is an important standard to be aware of to make correctly implementing the CPU possible
- ROM formats for the various systems our emulator platform may support

Battery

We will have to adhere to the international standards for lithium batteries.

- One such standard is the UL 1642
- We also must follow a standard created by the United Nations on safe transportation of dangerous goods, including lithium batteries, see ST/SG/AC.10/27/Add.2
- We also must follow BS EN 60086-4:2000 and EC 60086-4:2000 since lithium batteries will be our primary power source

Emulator Roadblocks:

- Implementation of a dual purpose 8 bit and 16 bit processor
- Varied documentation
  - Multiple documents
  - Inconsistencies
- GPU implementation
  - Screen layering

Hardware Roadblocks:

- Manufacturer changed pinout without updating datasheet
- Shipping time/Needing to wait an additional two days to retest

Module Roadblocks:

- Interactive /dev filesystems can yield complex source
- Difficult to test or build without working on an RPi directly

Future improvements:

Emulator

- Comprehensive emulation for the Game Boy system
- Comprehensive emulation for the Game Boy Color system
- Graphical dialog to select rom files

Kernel module

- Further tune real-time input support
- More intuitive naming scheme for /dev/rt* files

PCB

- Implement independent power circuit
  - Currently abandoned due to time constraints
- Thickness can be trimmed down an extra 3-5 mm with four more hours
- Add docking functionality as well as design a dedicated docking station
- Audio Output (Speaker)

Special thanks to Brian Wagner for the motivation to propose this project as well as Jack Potter for use of his 3D printer