

Handheld Emulation Station

sdmay19-25

Jacob Nachman, Nic Losby, Nick Lang, Sean Hinchee, and Matthew Kirpes


Our Mission

Our goal is to create a dedicated Emulation Station with a focus on mobility, battery life, and user experience.

Why Do We Want To Do This?

Our passion for games drove us to do a project revolving around games that we think users of retro games would enjoy.

This project also allowed us to utilize the skills we learned throughout our academic career by designing an embedded system from scratch.



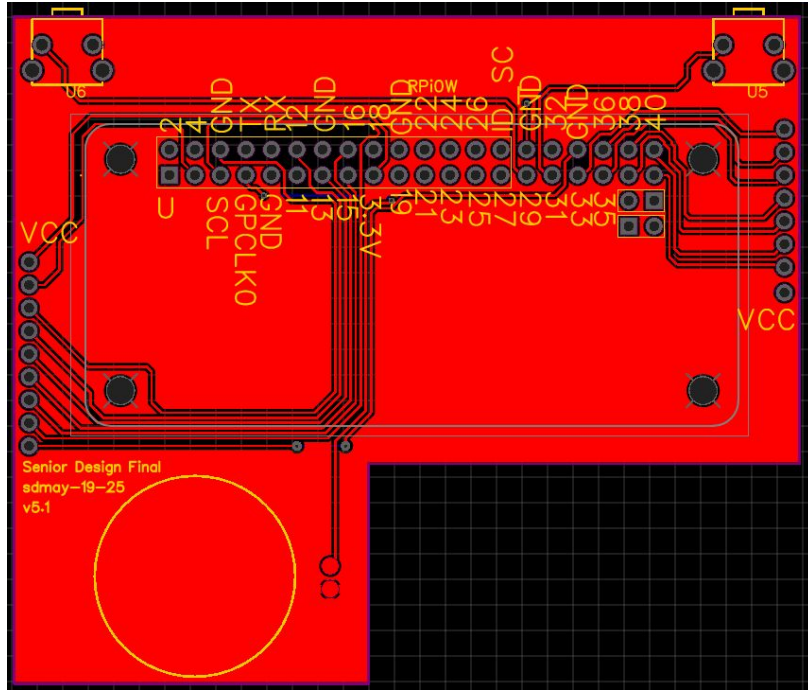
Our parts list

- 3.5 inch TFT LCD screen
- Raspberry Pi Zero W
- 3000mAh battery
- Buttons, joystick, and triggers
- Speaker (mono audio)
- Printed Circuit Boards
 - Main board
 - Two daughter boards

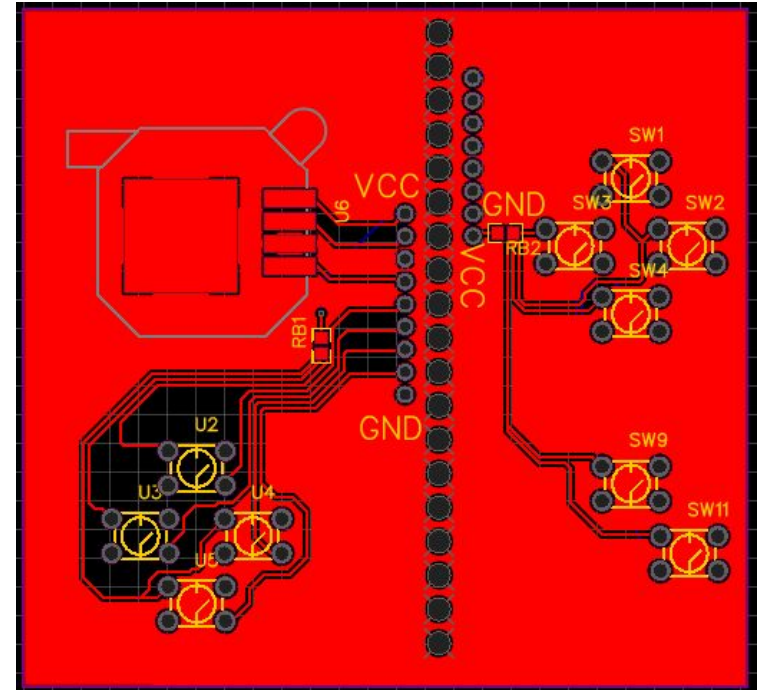


Our PCB Design

Main board:



Daughter boards:



PCB Design Challenges

- Keeping our boards under 100mm x 100 mm
 - Desired peripherals
 - Costs
- Balancing number of features
 - GPIO pin count limit on the Raspberry Pi Zero W
- Forgot to account for trace width
 - Ensure the trace can handle the required amount of power
- Power circuit has switched from custom to off the shelf
 - Pin out in datasheet was incorrect
 - Switched for timesake



PCB Status

- Original power circuit scrapped in favor of existent alternative
 - May be returned to in the future
- No sound currently, but necessary space is reserved
 - Planned for the future, purely software
- High quality board from OSHPark
- Layout is very functional
 - Ample space to solder button pins internally



Kernel Module Features

- Real-time input method
- Goal is to *feel* similar to using real hardware
 - Minimal input delay
- Fit into a minimal kernel configuration
 - Few dependencies
- Expose inputs from GPIO in the kernel to emulator
 - /dev filesystem



Kernel Module Challenges

- USB is well documented, but difficult to make real-time
 - Use GPIO instead
- Must accommodate GNU interfaces
 - udev, sysfs, etc.
 - Must be GPLv2+ licensed 😞
- Source must be written to strict standards
 - Memory safe
 - Thread safe
 - As compact as possible
- Must adhere to real-time requirements of input
 - Or as close as physically possible

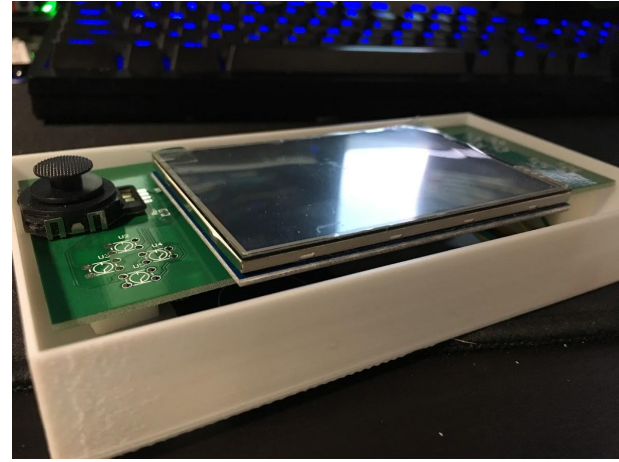
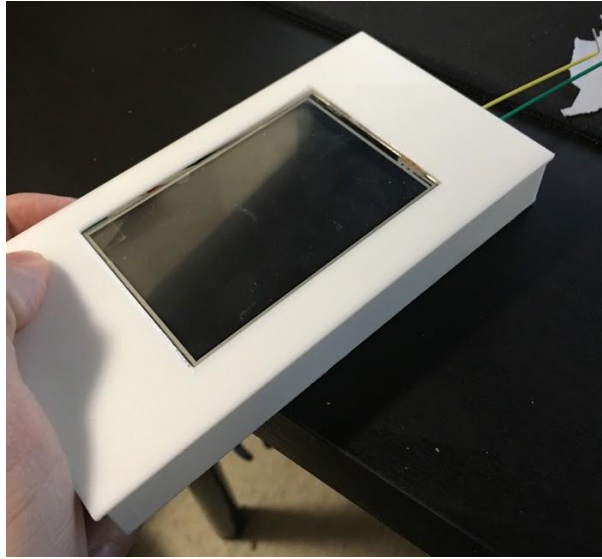
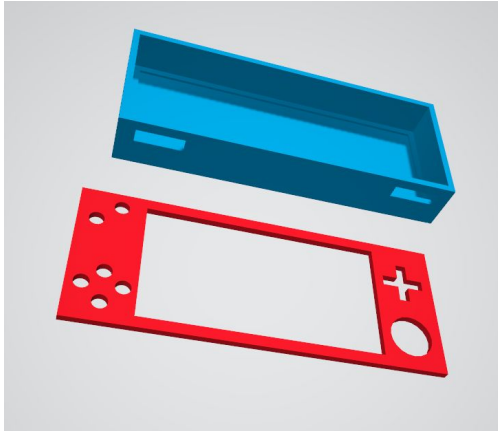


Kernel Module Status

- Scrapped in favor of a future alternative
 - Could not get consistent reads from pins 😞
 - Tested and working from Python gpio library
- Inconsistent documentation for using BCM2835 pins
 - Might not be able to use GPIO pin numbers on RPi pinout
 - Module source should be modifiable to align with other specs
- Module can create files and open/read/free GPIO pins!
- Module can create and support reads on /dev files!



Case Design



Case Design Challenges

- Learning the feature set of the tool
- Communication and changing of specifications (Size, Buttons, Ports, etc)
- How to mount internals (Battery, Raspberry Pi)
 - Screen and RPi must be very low-profile
- Accurately rendering various ports (USB, HDMI)
- Making the case look aesthetically pleasing
 - Ergonomics



Case Design Status

- Skeleton of the case is the perfect size
- Modular design to make updates easy and fast
- Perfect cutouts pre printing still needs to be done
 - Cut outs post printing with soldering iron




Software Emulator Features

- Implement the GameBoy and GameBoy Color
- Utilize kernel module directly for low-latency inputs
- Target SDL2 for graphics
- Load rom images from disk
 - Cartridges
- Implement dumping/loading save states of memory
- CPU cycles through, pulls and executes opcodes



Software Emulator Challenges

- The GameBoy Z80 processor is not actually a Z80
 - Hybrid 8-bit and 16-bit mode
 - Differing registers
 - Differing instructions
 - All publicly available documentation is reverse engineered
 - Contradict each other and may be incorrect
 - Undocumented behavior
 - 16-bit register loads
 - Refer to existing - accurate - emulators
 - Exceptions to implementation rules
 - Special registers
 - Special memory-write hooks
- 

Software Emulator Status

- Cartridge parsing and loading
- Memory management
 - Dump/restore to/from save states
- CPU instruction set via opcodes complete
- SDL2 graphics in resolution of GameBoy with full RGB
- Test suite working
 - Continuous development on GitLab master
- GPU is **not** complete
 - Work in progress
- Integrated with kernel module



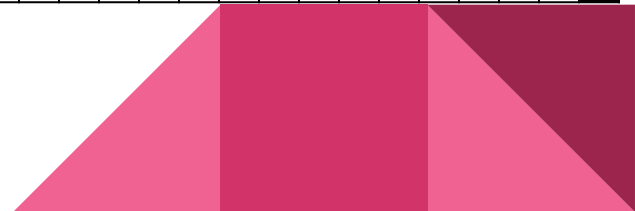
User Experience Development

- We have found a very pretty UI that supports many existing emulators and can be extended for our own emulator written by a previous ISU student
 - EmulationStation
- cronjob + rsync + encryption + Google Drive = instant backups of save files
 - Could use Amazon's EC2



Our Schedule

Proposed																Actual																
Tasks	Semester 2 (week)															Tasks	Semester 2 (week)															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
Develop Emulator	█																Develop Emulator	█														
CPU						█	█	█								CPU														█	█	█
GPU	█	█	█	█		█	█	█								GPU	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█	
MMU	█	█	█	█		█	█	█	█							MMU	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█	
Kernel Module	█	█				█	█	█	█							Kernel Module	█	█														
PCB	█	█	█	█		█	█	█	█							PCB	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█	
Assemble 1st Prototype						█	█	█	█							Assemble 1st Prototype																
Initial Testing						█	█	█	█							Initial Testing																
Final Product Completed											█	█				Final Product Completed																
Final Testing													█	█	█	Final Testing																
Final Presentation																Final Presentation																




Project Successes

- A fantastic looking product in the case
- 7.5+ hours of battery life
- A working PCB configuration
- A decent amount of the emulator implemented
 - GPU is the major missing part
- Tons of lessons learned!
 - Linux kernel development
 - Golang development
 - Implementing a custom CPU
 - Proprietary and poorly defined
 - Custom power circuit on a PCB
 - System integration with off the shelf parts



Project Pitfalls


- PCB and Parts
 - Bill of Material (BOM) was not double checked before ordering thus we were missing resistors
 - Power circuit ended up being non-trivial
 - Optimal case design is difficult
 - Space concerns
 - Licensing of Software
 - Software we wanted to use is licensed with a very restrictive license (GPLv3)
 - Want MIT/Apache license
 - Inconsistent or incorrect documentation
 - Applies to the kernel module, software emulator, and ICs used on PCB
- 

Future Work

- PCB
 - Custom power circuit
 - Change to IPS LCD
 - Higher than 30fps
- Emulator
 - Complete the GPU
 - Implement sound
- Kernel Module
 - Cohesive rewrite



Legal

- EmulationStation is licensed under the MIT license
 - GUI solution for multiplexing emulators
 - Permissive
 - We can license our solutions under MIT
 - Except the kernel module, must be GPLv2+
 - Writing and distributing **emulators** is legal in the US
 - Sony Computer Entertainment v. Connectix Corporation, 203 F.3d 596 (2000)
 - **Dumping** roms is legal in the US
 - Lewis Galoob Toys, Inc. v. Nintendo of America, Inc. (Ninth Circuit Court of Appeals, 1992)
 - We do not distribute any **rom** images
 - This is illegal
- 



Questions?