KIPS Mk. II

Group 16

KIPS Mk. II

- Designed as a portable personal computer with mobile and desktop use.
- Battery powered for use on the go
- Peripheral integration (mouse, keyboard etc.) in desktop mode
- Array of various sensors to provide various environmental readings
- AM/FM radio capability
- LED flashlight
- D-pad control for ease of use in mobile mode

Meet the Team

Scott Valentine
EE
Hardware

Jonathan Meyers
CpE
Software

Jessenia Argueta
CpE
Software

Harrison Mills
CpE
Software

Motivation and Background

Inspired by the pocket PC and other modern smart devices, our group wanted to create a portable PC device with higher functionality, versatility, and modifiability than today's commercially available devices.

The Knights Information Processing System Mark II (KIPS Mk. II) is a wearable, wrist-mounted computing device, designed as an extensive improvement and overhaul of the earlier KIPS Mk. I prototype.

Goals and Objectives

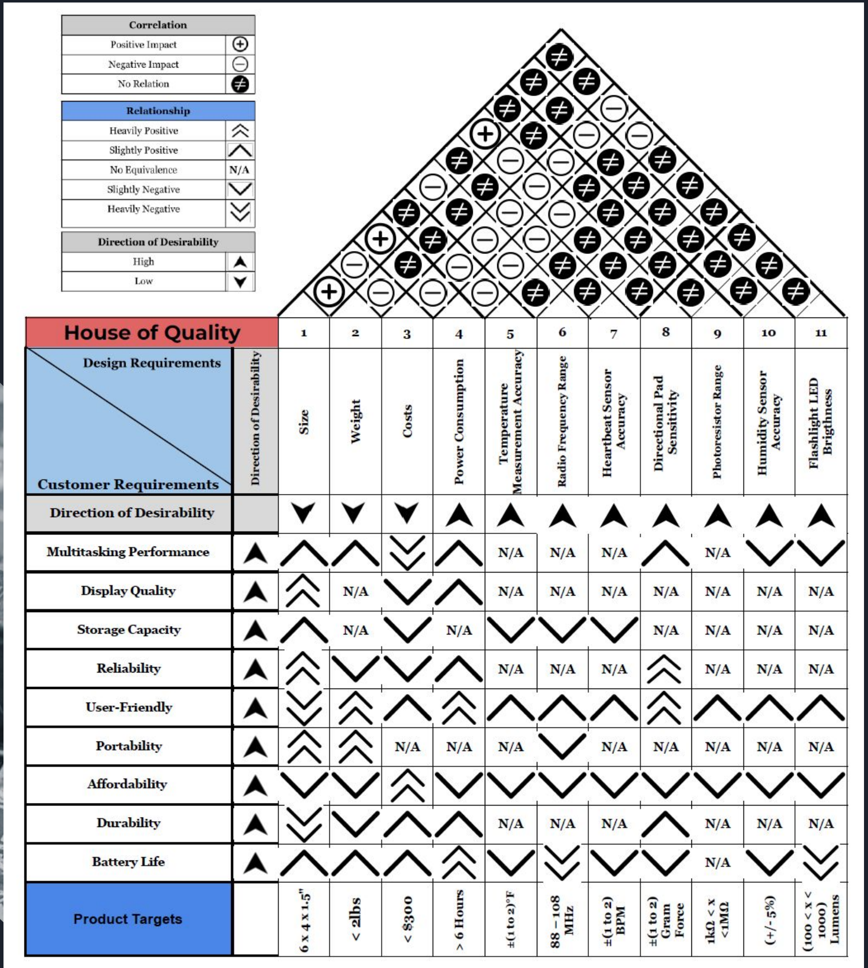
- Our goal with this project is to design an affordable and portable personal computer that can function as a wearable smart device as well as a dedicated PC, providing the user with immediate information about themselves and the environment.
- Due to the multifaceted nature of this project we have outlined the following goals, followed by objectives, categorized into hardware and software domains.

Table of Engineering Specifications

Component	Parameter	Specification
Desktop Battery Life	Time	> 2 Hours
Mobile Battery Life	Time	> 5 Hours
Dimensions	Size	~ 6 in. x 4 in. x 2 in.
Weight	Pounds	< 4 lbs.

Component	Parameter	Specification
Mobile to Desktop Conversion Time	Response time	< 30 seconds
Power Consumption	Wattage	< 10 Watts
Avg. System Operating Temp.	Temperature in fahrenheit	95°F - 120°F
Response time/ time delay	Response time	< 10 seconds

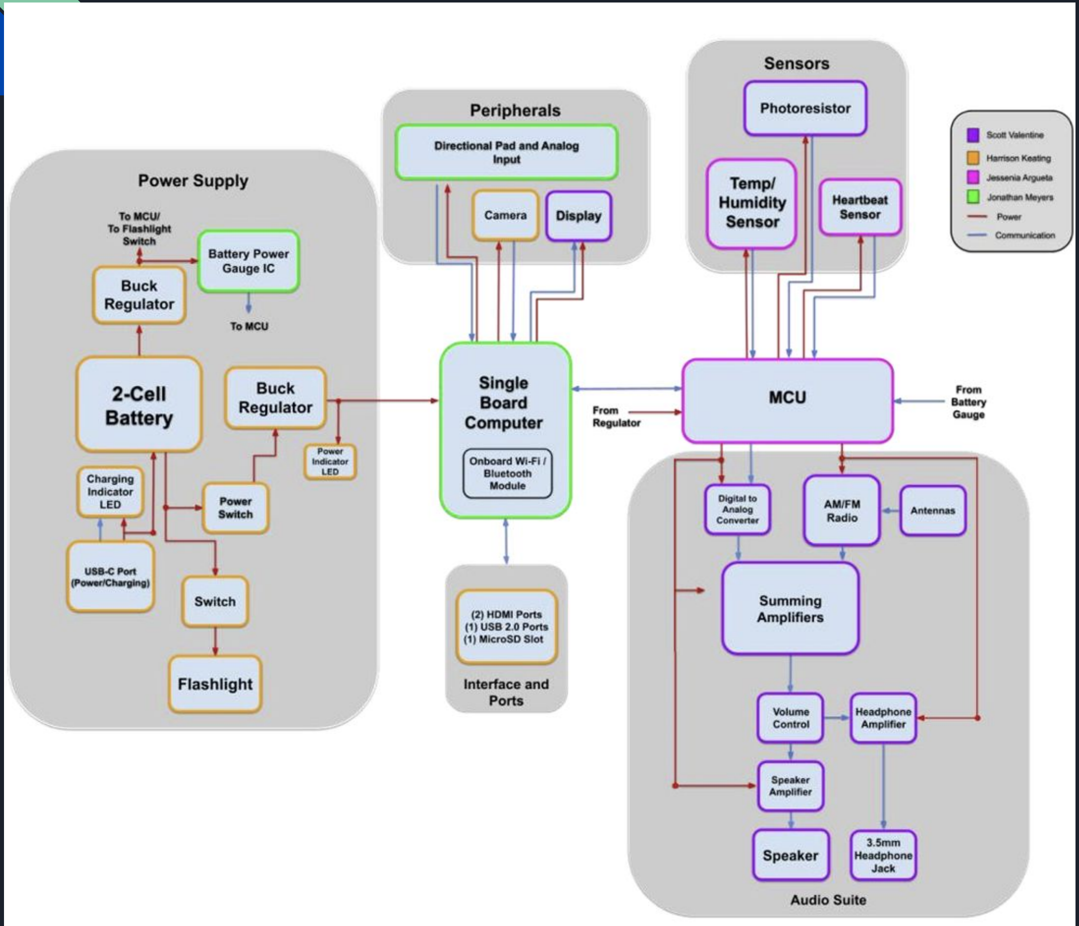
House of Quality



KIPS Mk. I

- Completed for 2024-2025 UCF IEEE Internal Projects Competition
- Rough draft of K.I.P.S.
- Features:
    - Touch Screen
    - I/O port
    - Basic battery power
    - Rough mobile UI (Rust & Python)

Hardware Design



- Our hardware design is composed of several different systems that work together to create KIPS MKII.
  - Power supply
  - Peripherals
  - SBC
  - MCU
  - Interface and Ports
  - Sensors
  - Audio Suite

Hardware Main Goals

- Design a wrist wearable device that serves as a marriage between a pocket PC and a smart-watch.
- Our prototype will be able to facilitate the real-time measurement of the Heart Rate
- Our prototype will be able to measure Temperature, Humidity, and Daylight in real time
- Enable the user to use battery power and wall power to operate the device.
- Allow the user to display the contents of our device.
- Enable user navigation of KIPS Mobile Mode via directional pad (D-Pad)
- Enable touchscreen capability
- Design a lower profile shell than MkI and wrist strap while ensuring even weight distribution.
- Incorporate an LED flashlight with the use of software control and a resistor to avoid burn out.
- AM/FM Radio

Hardware Objectives

- ❖ Develop a custom I/O board for SBC and MCU.
- ❖ Model and create a 3D printed shell to house the device.
- ❖ Battery Power
- ❖ Power Button
- ❖ Heart Rate Sensor
- ❖ Light Sensor
- ❖ Temperature and Humidity Sensor
- ❖ AM/FM Receiver
- ❖ Implement an HDMI port in the system

Hardware Selection

- SBC
- MCU
- Regulators

SBC, MCU, and FPGA Technology Comparison

	FPGA	MCU	SBC
OS Support	N/A	N/A	Yes
Power Draw	High	Low	Moderate/High
HDMI Support	Yes	N/A	Yes
USB Support	Yes	Limited	Yes
Boot Time	Near-instant	Fast	Slow
Price Point	High	Low	Moderate
Form Factor	Large	Very Small	Small

- Although each single choice has its advantages, ultimately we decided combining SBC and MCU to work together would be best.
- This way the SBC handles high level logic and the MCU handles lower level logic

SBC Product Selection

	Raspberry Pi	Radxa (Rock Pi)	BeagleBone
Processor	Quad-core	Quad or Eight-core	Single, Dual or Quad-core
RAM	0 (lite), 512MB, 1 GB, 2, 4, 8, 16 GB	256MB, 512MB 1 GB, 2GB, 4GB, 8GB, 16GB, 32GB	512MB, 4GB
Onboard (Flash) Storage	0 (lite), 16, 32, 64 GB	N/A	0, 4 GB, 16 GB
USB	USB 2.0 & 3.0	USB 2.0 & 3.0	USB 2.0
Communication Protocol	UART, SPI, I^2C	UART, SPI, I^2C	UART, SPI, I^2C
Input Current	0.7-1A, 3A	1A, 3A	1A, 3A
OS Support	Raspberry Pi OS, Ubuntu, LibreELEC*, RetroPie*, DietPi*	Radxa OS, Ubuntu	Debian, Linux, and Android Support
Price/Price Range	$45-$135	$45-$200	$55-$228

- The Raspberry Pi benefits:
  - extensive documentation & ecosystem
  - variety of customization options
  - reliable and well-balanced processing power.
  - Onboard flash memory
  - Low cost

SBC Part Selection

	Pi Compute Module 4 (CM4)	Pi 5 Compute Module (CM5)
RAM (GB)	1, 2, 4, 8	2, 4, 8, 16GB
Flash Memory (GB)	0 (lite), 8, 16, 32	0 (lite), 16, 32, 64
Processor	quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz	quad core Cortex-A76 (ARMv8) 64-bit SoC @ 2.4GHz
Form Factor (mm)	55 x 40 x 4.7	55 x 40 x 4.7
USB	1 USB 2.0	1 USB 2.0 2 USB 3.0
HDMI	2 HDMI 2.0	2 HDMI 2.0
GPIO	≤ 28 pins	≤ 30 pins
Cost	\$30-\$85	\$45-\$135

- The factors that lead to us choosing the CM4 were:
  - Small size
  - Reasonable price
  - Ram and flash memory options
  - Availability vs CM5

MCU Product and Part Selection

	ESP32-S3	RP2040	nRF5340	STM32U585AI
Core Type	Dual-core Xtensa LX7	Dual-core ARM Cortex-M0+	Dual-core ARM Cortex-M33	Single ARM Cortex-M33
Clock Speed (MHz)	240	133	64-128	160 (max)
Power Consumptio n (min-max)	~2.5µA - 260mA	~50mA - 150mA	~0.9µA - 5.1mA	~0.16µA - 3.1mA
Size (mm)	18 x 25	51 x 21	4.4 x 4	7 x 7
USB Support	Native USB-OTG	Native USB	USB	USB FS + OTG FS

- The major reasons for this decision include the availability of info, the ability to receive assistance from others, the computing power the S3 offers at a reasonable power draw, its built-in Wi-Fi module, and its affordability

Temperature and Humidity Sensor Product Part and Selection

	AM2301B	SHT30A-DIS-B + SF2 Filter Cap	HTU21D(F) + Filter	AM2320
Input Voltage (V)	2.2 - 5.5	2.4 - 5.5	1.5 - 3.6	3.1- 5.5
Operating Current	250 nA	600µA - 1500µA	300µA - 500µA	950 µA
Output Data Type	°C, %RH	°C, %RH	°C, %RH	°C, %RH
Size (mm)	58.8 x 26.8 x 13.2 (with enclosure)	6.0 x 6.0 x 2.5 (with cap)	3 x 3 x 1.13 (with membrane)	15.00 x 12.10 x 4.50
Temperature Accuracy (°C)	±0.4	±0.3	±0.3	±0.5
Humidity Accuracy (% RH)	±2	±3	±2	±3
Cost ($)	7.95	3.21 + 1.26	3.89	3.95

- The AM2320 was selected best for:
- ease of integration
- list price
- required input voltage

Heartbeat Sensor Product and Part Selection

	MAX30102	The Original PulseSensor Kit	Gravity Heart Rate Monitor (SEN0203)
Input Voltage (V)	1.8 - 3.3	3 - 5.5	3.3 - 5
Typical Voltage (V)	1.8	Vdd/2 (output)	Unspecified
Operating Current	0.7µA - 600µA	3mA - 4mA	< 10mA
Output Data Type	Heart rate + SpO_2	Heart rate	ECG waveform
Size (mm)	5.6 x 3.3 x 1.55	15.8 x 15.8 x 3.6	28 x 24
Measurement Method	PPG	PPG	PPG
Cost ($)	5.41	24.99	16.00

- Although the output would be in an ECG waveform, there is also a digital option on the device itself.
- Additionally, it is within spec for all the electrical requirements of the device.
- The MAX30102 was a strong contender, but couldn't be chosen as its voltage operating range was below what KIPS needs to operate properly.

Camera Product and Part Selection

Camera Choice	IQL-IMX335/FF	Raspberry Pi Camera Module 3	IQL-OV7676/FF
Cost	\$50 (+\$12.99 Shipping Fee)	\$32.95	\$25
Resolution	2626 x 1960	11.9 megapixels, 4608 × 2592 pixels	640 x 480 (VGA)
FPS	60	Up to 120 FPS	Up to 30
Interface	MIPI	I²C	SCCB / I²C
Power Consumption	DC 5V, 150mA (~0.75W or 750mW)	Operating voltage: 3.3V Max:1.16W	~60mW
Size / Dimension	0.866" L x 0.591" W (22.00mm x 15.00mm)	25 × 24 × 11.5mm	0.236" L x 0.236" W (6.00mm x 6.00mm)
Height - Seated (Max)	0.902" (22.90mm)	11.5mm	0.142" (3.60mm)

- The Raspberry Pi Camera Module 3 struck the best balance between functionality, ease of integration, power consumption, and cost-efficiency.
- It fulfills all design constraints without exceeding budget limits.

Screen Product and Part Selection

	OSOYOO 3.5 Inch DSI	Waveshare 4.3 Inch DSI LCD	Waveshare 4-inch DSI LCD (C)
Size	3.5”	4.3”	4”
Shape	Rectangular	Rectangular	Circular
Resolution	800x480	800x480	720x720
Touchscreen	Capacitive	Capacitive	Capacitive
Price	$30 - $45	$40 - $50	~50$
Panel Type	IPS	IPS	IPS
Viewing Angle	170°	160°	160°
Refresh Rate	60Hz	60Hz	60Hz
Protocol	DSI	DSI	DSI

- We have decided to go with the OSOYOO 3.5 inch DSI display.
- We wanted something a bit smaller than KIPS Mk I to reduce weight, as well as a screen that benefits from the use of the control buttons on the side.
- Additionally, LCD screens are plentiful, reasonably priced, and the DSI protocol is directly compatible with the Raspberry Pi CM4 .

Radio Antennae Product and Part Selection

	Superbat Telescopic FM Antenna	Alibaba OEM Telescopic FM Antenna	Ebay Telescopic FM Antenna
Connector Type	F-Type Coax	F-Type Coax	F-Type Coax
Collapsed Length	13.5cm	9.5cm	Unknown
Extended Length	51cm	44cm	76cm
Impedance	75Ω	75Ω	75Ω

- For the sake of compact size, we chose the OEM antenna from Alibaba. We can get extra connectors that the Superbat brand includes if we deem it necessary, so the short length won out overall.

Power distribution

Component	Supply Voltage	Min Current	Max Current	Approximate wattage
SBC	5V	0.7A	3A	3.5-15W
MCU	3.0-3.6V	~2.5µA	260mA	7.5µ-0.936W
Heartbeat Sensor	3.3-5V	<10mA	10mA	33mW
Temperature and Humidity Sensor	3.1-5.5	10µA	950µA	31µ-5.225mW
Camera (SBC INCL.)	5V	100 mA	150mA	750mW
Screen (SBC INCL.)	3.3V	100mA	500mA	0.5W- 2.5W
Digital to Analog Converter	3.0V < x < 3.5V	0.5mA	13mA	1.5mW-45.5 mW
AM/FM Radio IC	2.0V < x < 5.5V	0.1mA	21.5mA	~30µW < x < 110mW
Headphone Amplifier	2.7V < x < 3.6V	0mA	10.5mA	0-37.8mW
Speaker Amplifier	2.2V < x < 5.5V	273mA	600mA	0.6006 W-3.3W
Speaker	2.83V-3.46V	0.354A	0.433A	1-1.5w
Total Wattage				Max 29.4043W

- Power is distributed to these components by way of two TPS566242 buck regulators giving 5.2V and 3.3V from 9V, which draw their power from a USB-C PD board feeding a 15V to 9V LM2678 buck regulator.

Regulator Product and Part Selection

Name	TPS566242	LTC3615
Brand	Texas Instruments	Analog Devices
Input Voltage Range	3V-16V	2.25V-5V
Reference voltage	Vin	Vin
Output voltage	0.6V-7V	0.6V-~5V
Max Current	6A	3A
Price	0.77	4.72
Efficiency	95%	94%
Switching Frequency	600 kHz fixed	200kHz to 4MHz (External resistor)

- Overall, the TPS566242 offers the necessary efficiency, protection, and simplicity for this project.
- Its wide input range, high current capability, and robust protection circuitry make it a dependable choice for converting the dual-cell battery voltage into a stable 5V rail for system operation.

Software Selection

- Operating system
- Primary programming language (Python)
- Primary communication protocols (UART and I2C)

Software Main Goals

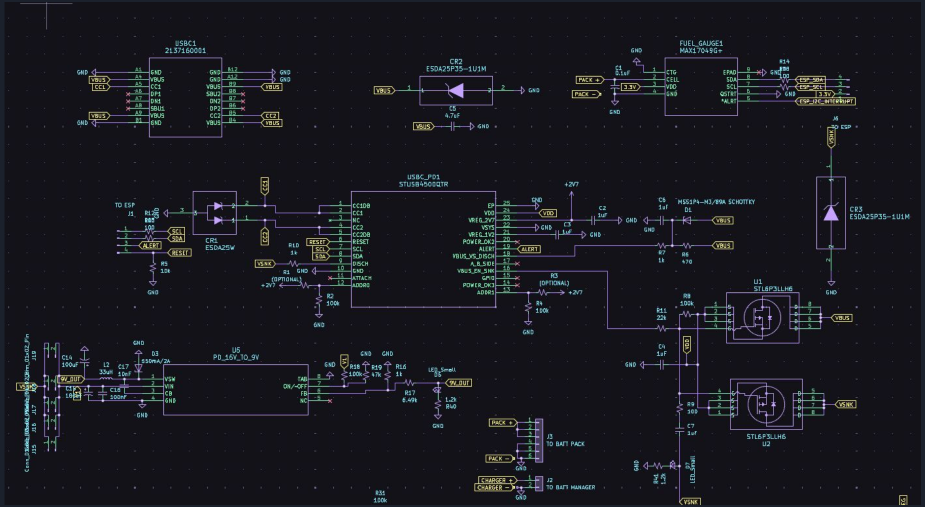
- Relay accurate temperature and humidity information back to the MCU.
- Enable user vital signs to be taken and displayed through KIPS through reading of a heart rate sensor
- Allow for auto brightness setting through daylight sensor readings
- Enable mobile use through a suite of applications designed to mimic a smartwatch.
- Enable desktop use through selection of an OS and write programs to communicate with the MCU, which enables the user to make their own modifications as well.
- Integrate sensor suite with both the mobile and desktop environments.
- Enable the user to switch from mobile mode to desktop mode and vice-versa.
- Design different power modes associated with the desktop and mobile OS.

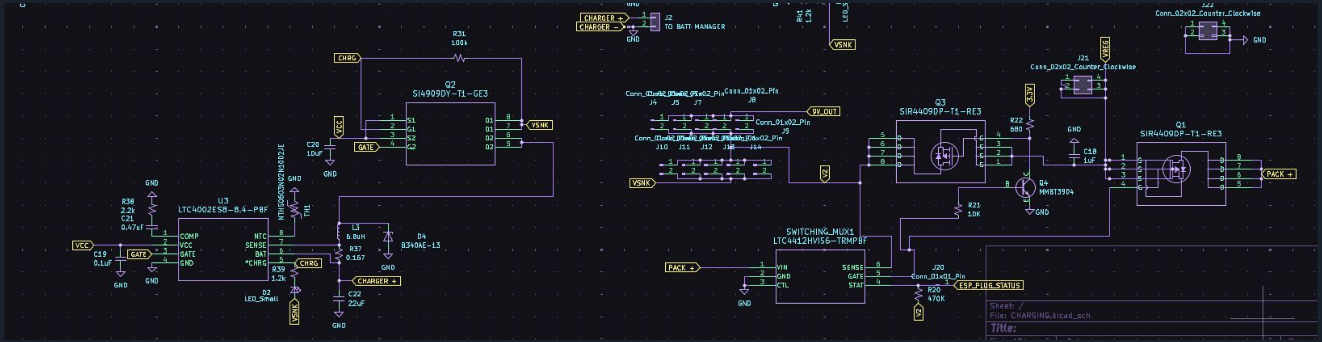
Software Objectives

❖ Develop and include various mobile-oriented applications

❖ Write software protocol for switching between mobile and desktop operating systems

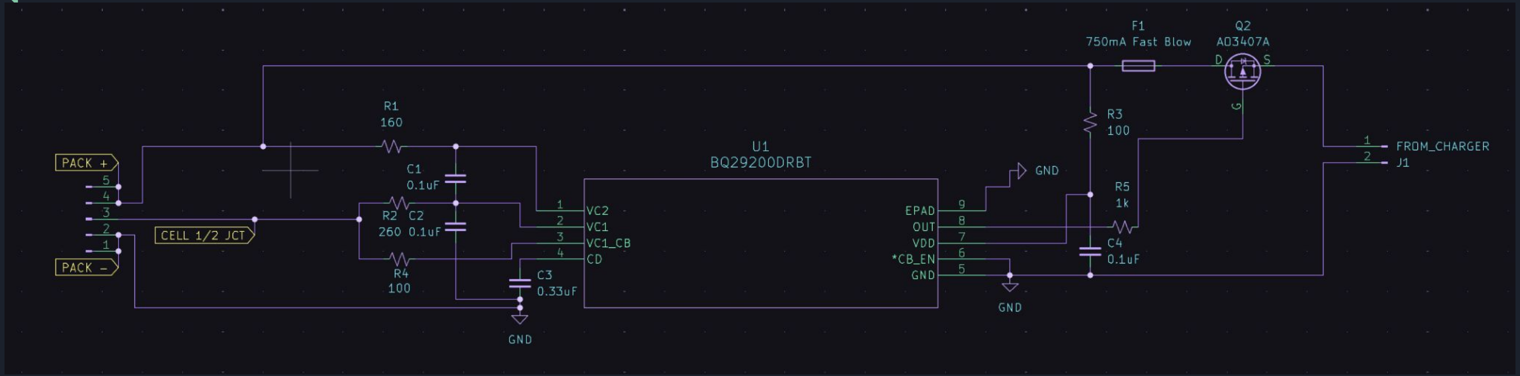
Power Supply Schematics - PSU





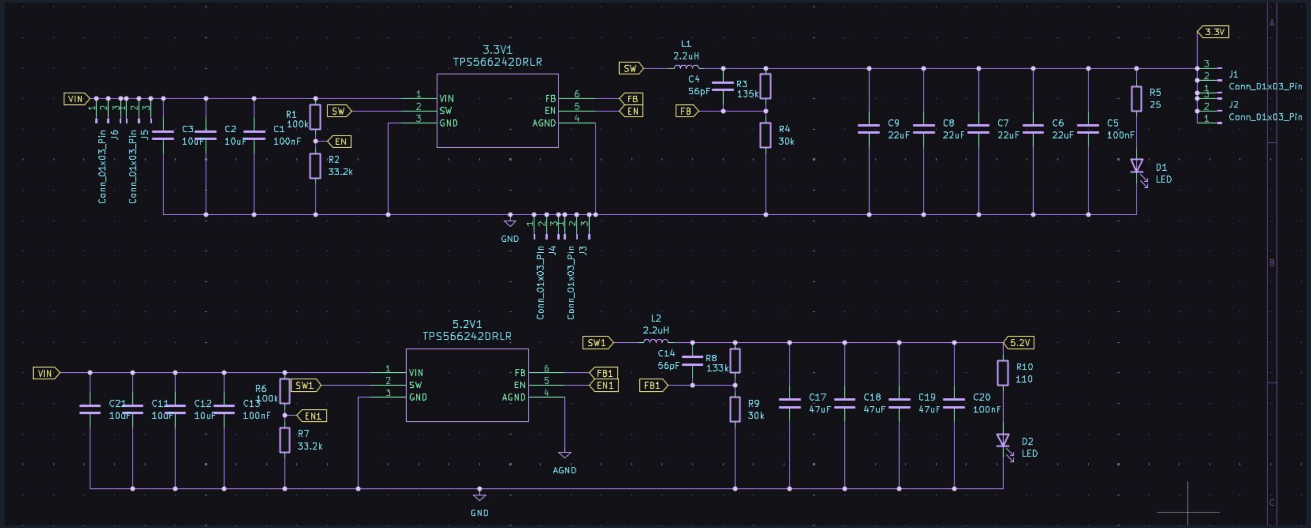
- The power supply boards incorporate several features such as USB-C power/2S li-ion battery charging, a battery fuel gauge, and an ideal diode power switcher to power the system.

Power Supply Schematics - Battery Management System



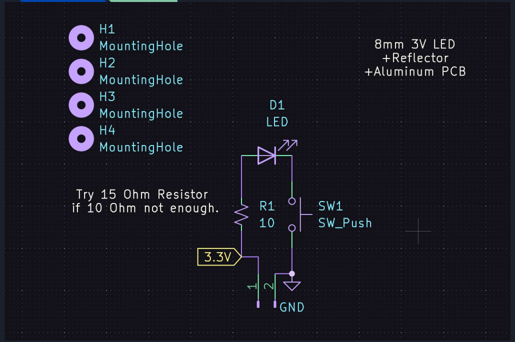
- The 2S Li-Ion battery manager's purpose is to ensure the 2 Li-Ion cells have overvoltage and overcurrent protection.

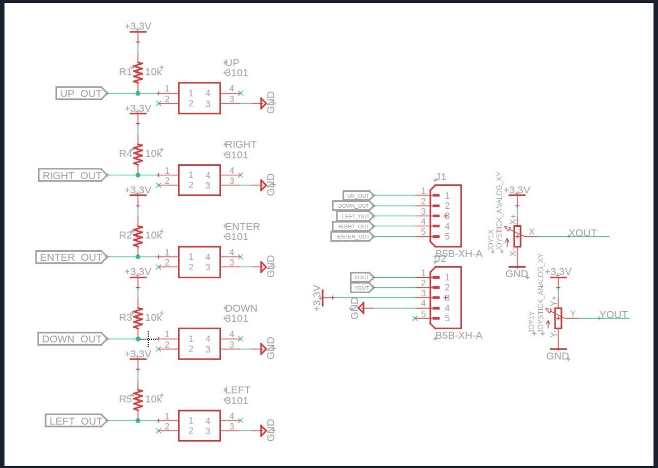
Power Supply Schematics - Regulator



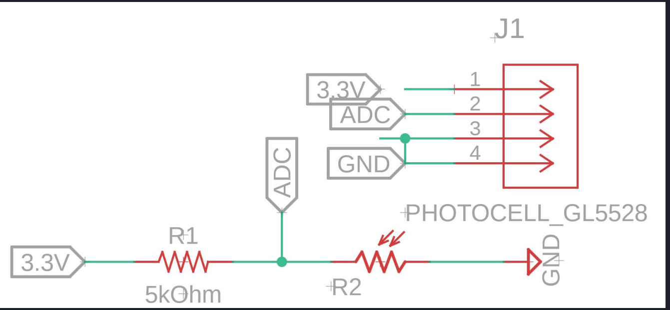
- These are 2 of the same regulator (TPS56624) configured with one at 5.2V and the other at 3.3V stepped down from 9V.

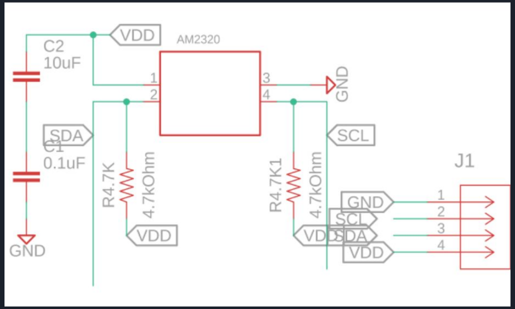
Sensors/Controls/Flashlight Schematics



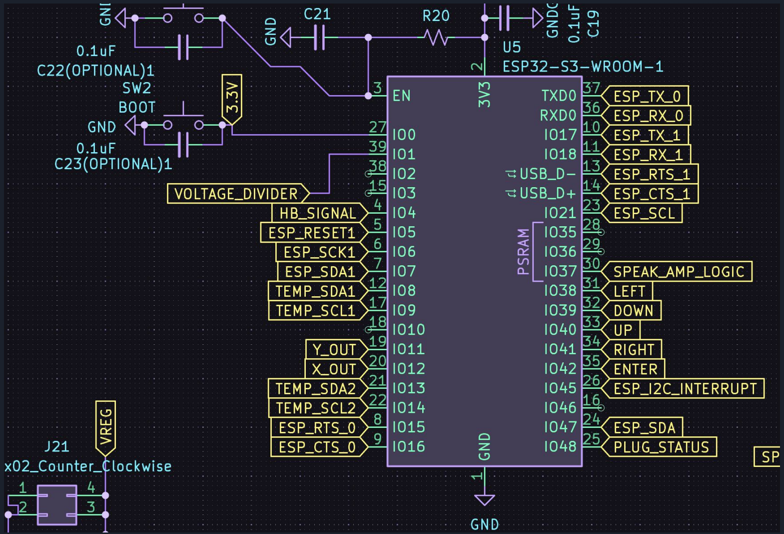


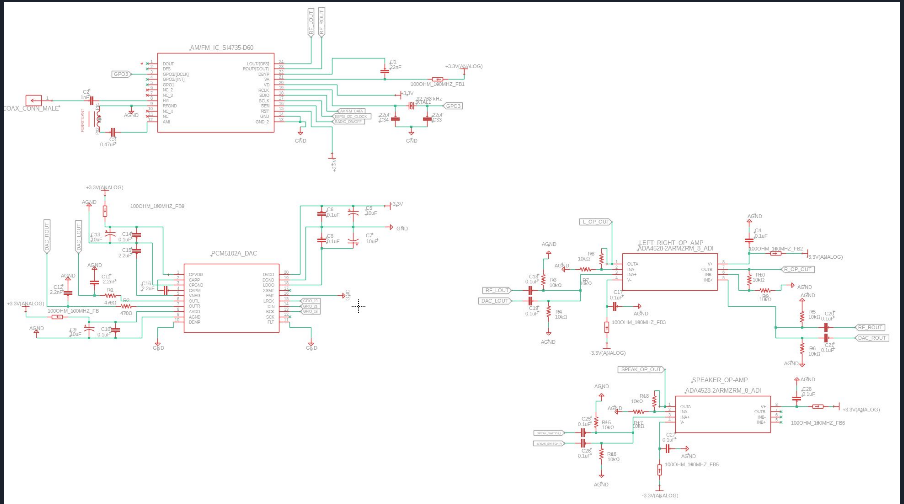
- Here, the following two sensors are pictured:
  - Photoresistor
  - Temperature and humidity
- Flashlight and the Control Board.





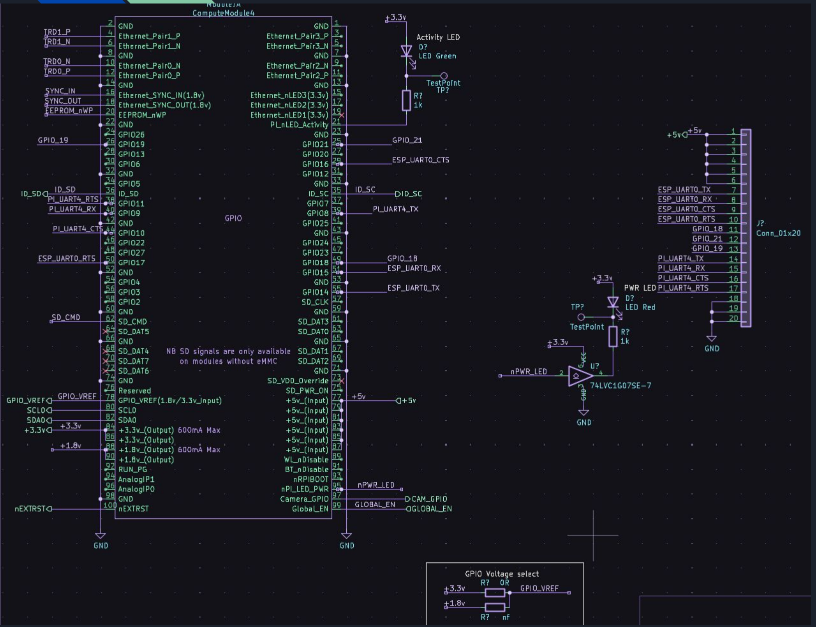
ESP/Audio Board Schematic





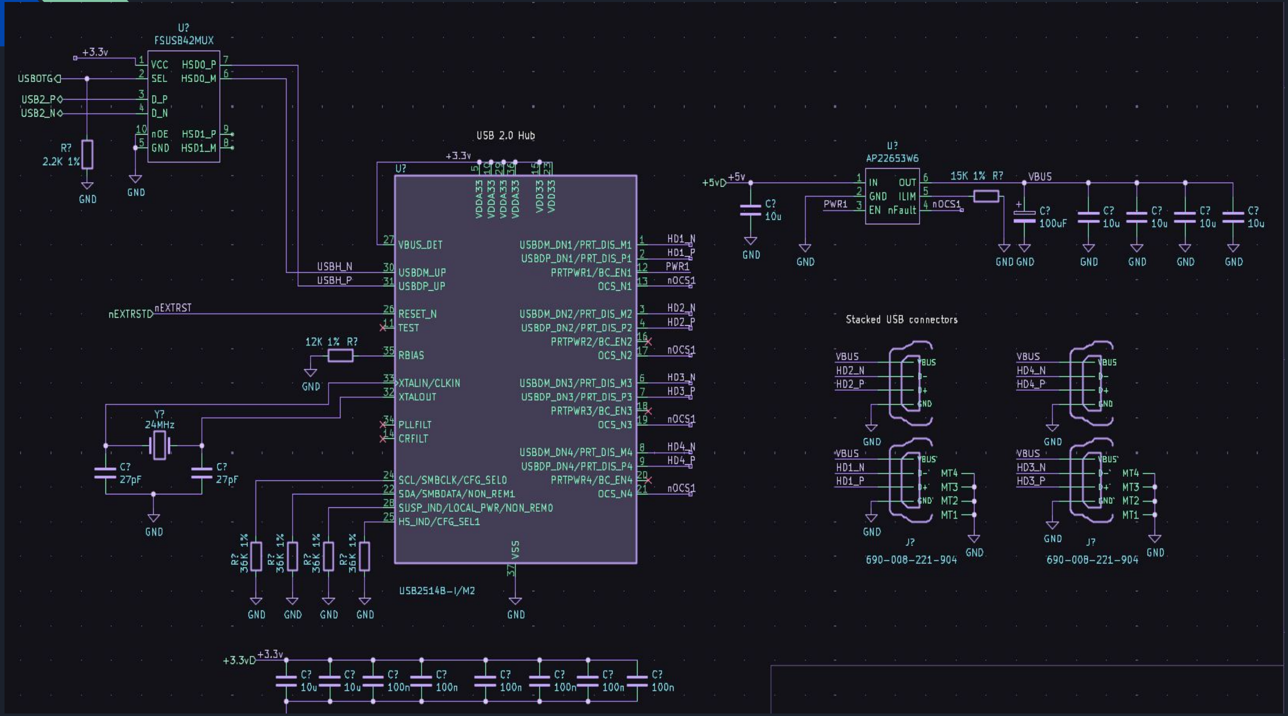
- These are the schematics for the ESP32-S3-WROOM-1U MCU for sensor and control management, as well as the schematics for the audio system.

High Speed Board Schematics



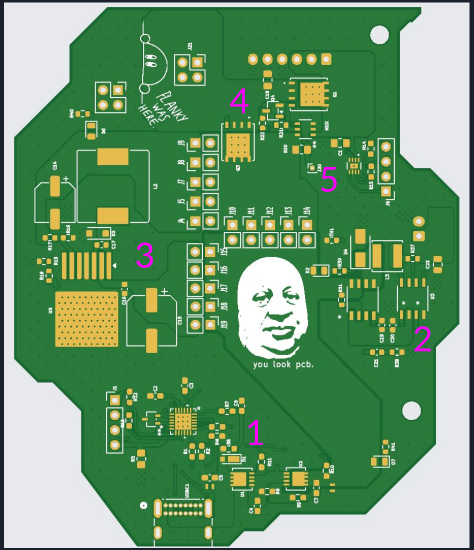
The high speed board features two FPC connectors for CSI and DSI interfacing, two HDMI ports, and the row of pins that connect to the rest of KIPS Mk II.

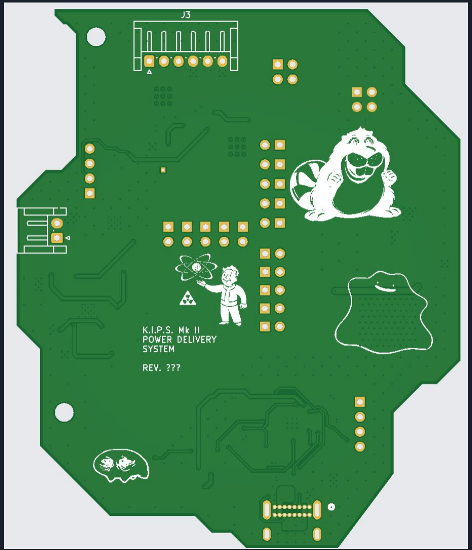
High Speed Board Schematics Cont'd



This is the USB 2.0 hub for the high speed board, featuring 2x USB 2.0 ports.

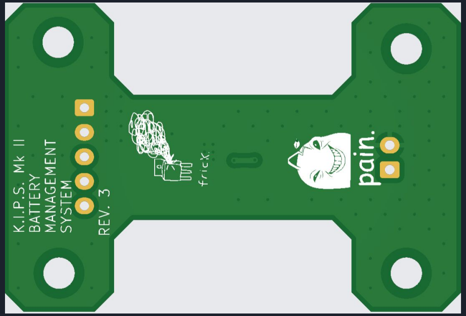
Power Supply Board

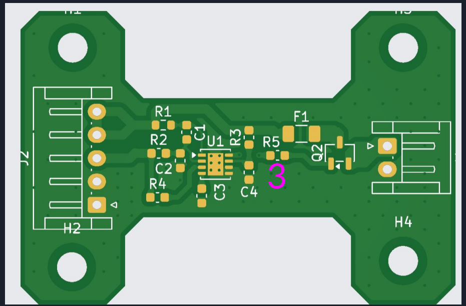




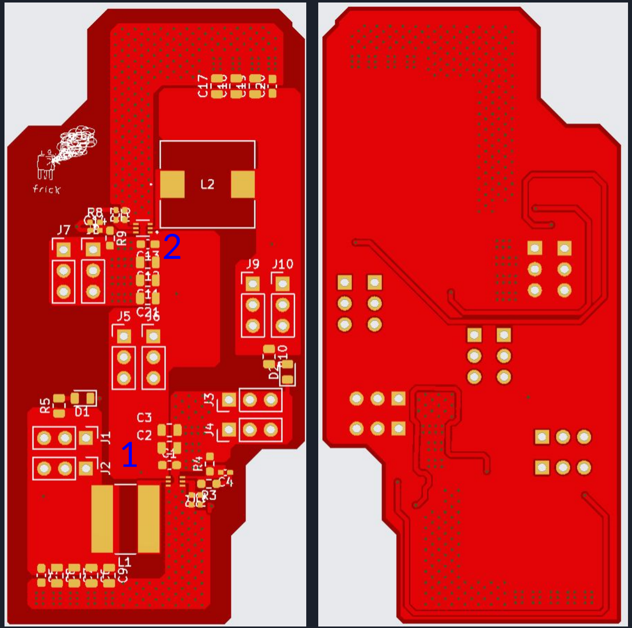
- Power Supply
  - USB-C PD Controller (1)
  - 2S Li-ion Battery Charger (2)
  - 15V to 9V Regulator (3)
  - Ideal Diode Power Switcher (4)
  - Li-ion Fuel Gauge (5)

Regulators and Battery Management





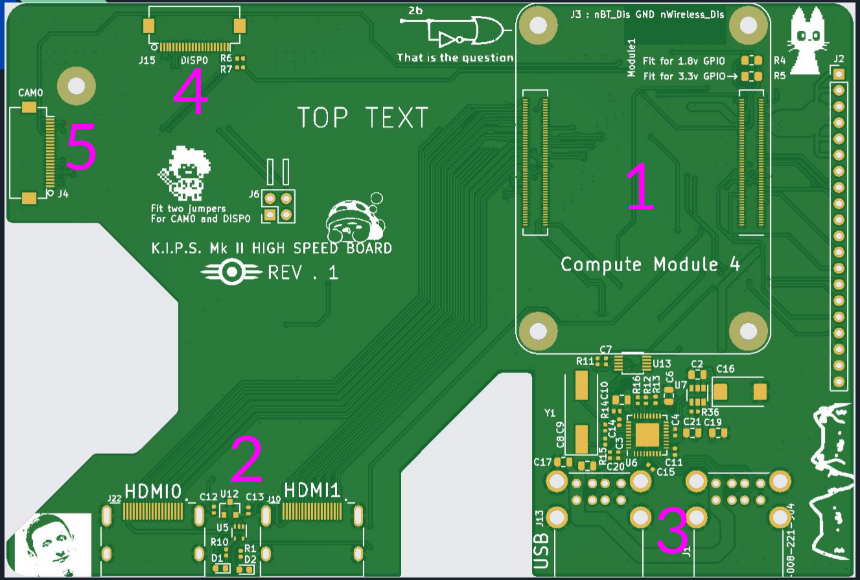
Battery Management (BMS) Board

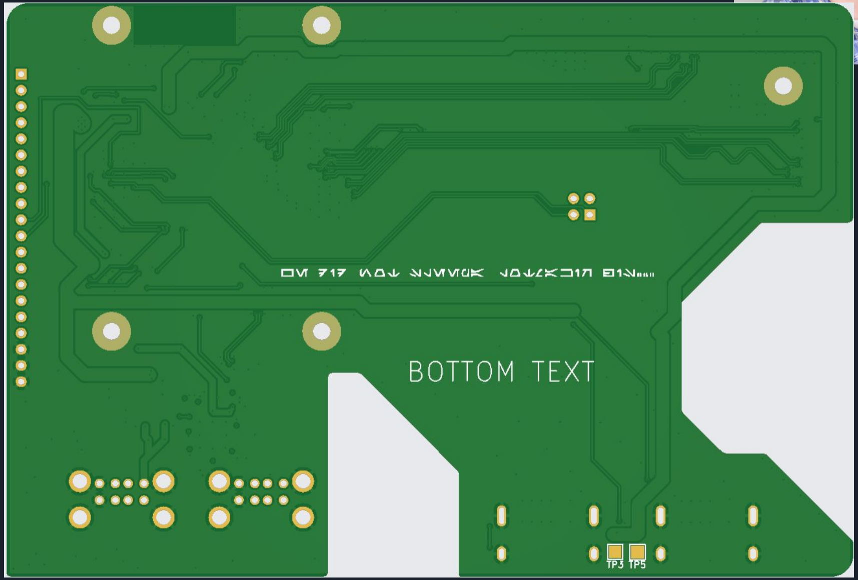


5.2 and 3.3 Regulator

- 3.3V Regulator (1)
- 5.2V Regulator (2)
- 2S Li-Ion Battery Management System (3)

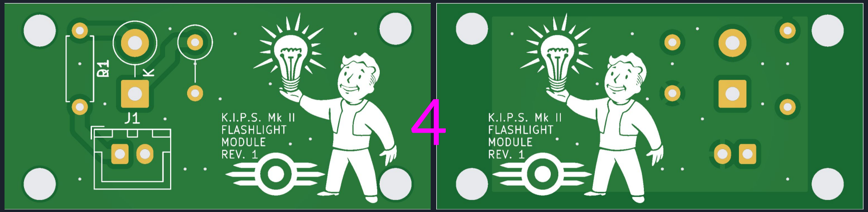
High Speed Board

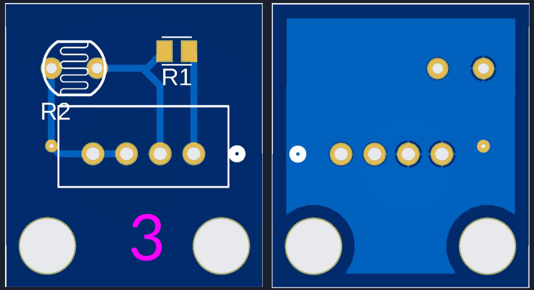


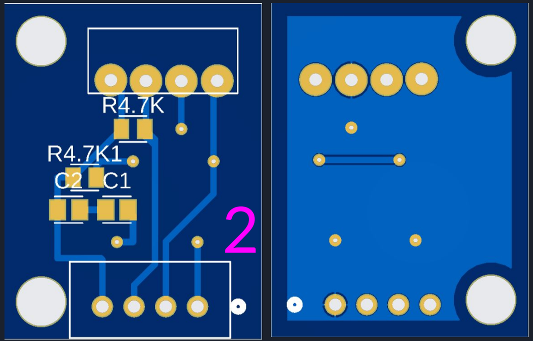


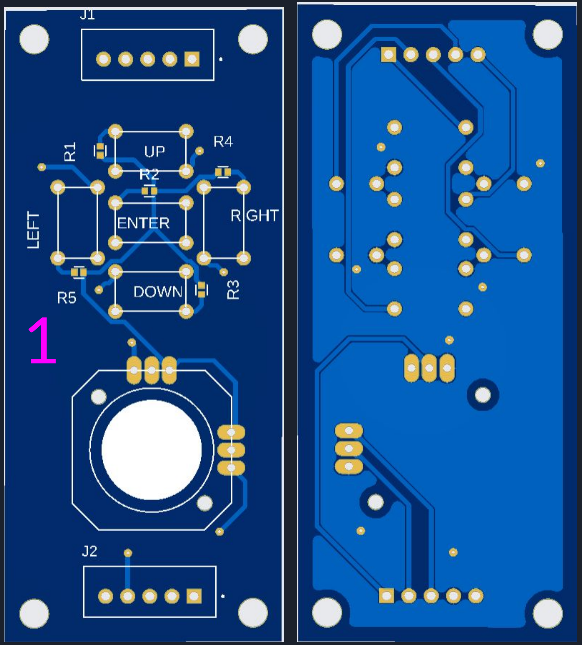
- High Speed Board
  - Raspberry Pi CM4 Socket (1)
  - 2x HDMI Ports (2)
  - 4x USB 2.0 Ports (3)
  - 1x DSI FPC (4)
  - 1x CSI FPC (5)

Peripherals/Controls









- Peripherals/Controls
  - D-Pad/Joystick (1)
  - Sensor Boards (Temp/Humidity (2), Photoresistor (3))
  - Flashlight (4)

ESP/Audio Board

For our project we used a microcontroller to manage most of the peripherals so the Pi CM4 will be the software controller. On the ESP/Audio Board we have:

- ESP32-S3-WROOM-1U Microcontroller
- PCM5102A Digital-to-Analog Converter
- SI4735 AM/FM Radio IC
- Speaker, Headphone Jack, and A-Type Volume Potentiometer

Communication Protocol Selection

I^{2}C follows a synchronous, multi-master, multi-slave communication model, meaning that more than one master (controller) can exist on the same bus, and each slave device (peripheral) has a unique address.

I^{2}C works by having a host processor in our case a (Raspberry Pi CM4) that can tell slave devices to transmit data and also to respond to inputs

I^{2}C is the best choice for this project as all of our peripheral devices use it . And also for its ease of use as it requires only 1 data lane.

Operating System Selection

	Raspberry Pi OS	Ubuntu Core	DietPi
Base	Debian-Based	Debian-Based	Debian-Based
Resource Usage	Moderate, lightweight	Low-medium, higher than other options	Extremely Low, minimal overhead
Architecture	ARM (32-bit & 64-bit)	ARM SBCs / Embedded	ARM SBCs
Software Management	APT, Debian repositories	Snap packages	Menu-drive auto-installer
Update Cycle	Community / official updates available	Regular 6-month updates plus security patches	Manual/ automated installer
Ease of Customization	High, open-source	Moderate, snap packages easier to update but less flexible	Very high, minimalist

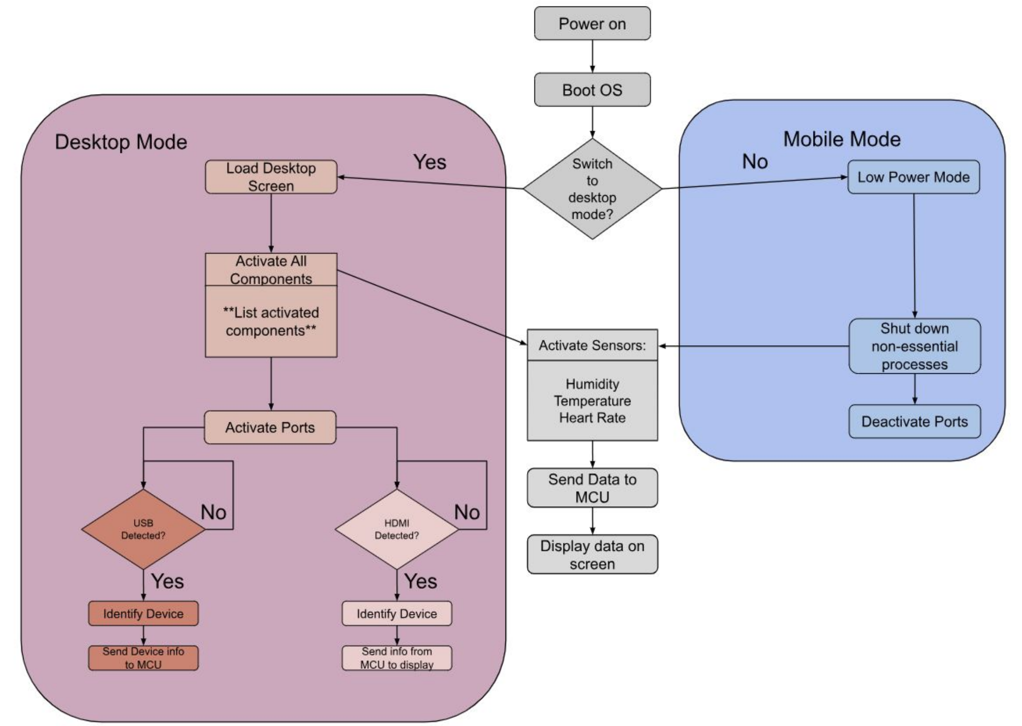
- Ubuntu Core uses too many resources for its small ecosystem
- DietPi was very minimalist and single-purpose oriented
- Raspberry Pi OS allowed for enough flexibility to add extra features without being barebones or bloatware.

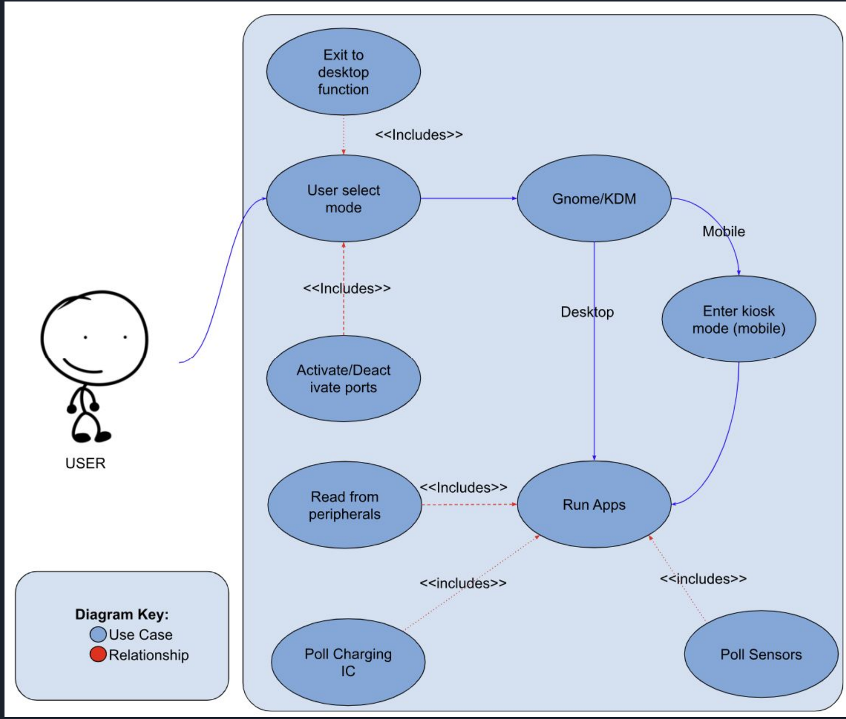
Programming Language Selection

	Python	C++	Rust	Arduino
Compatibility with ESP32	Limited (MicroPython)	Yes	Yes	Possible to
Execution Speed	Slow	Fast (compiled)	Fast	Fast
Ecosystem/ libraries	Very large (sensors, GPIO, etc)	Large (hardware, graphics, real-time, etc)	Fairly small	Limited to mainly Arduino boards
Ease of Learning	Very easy	Moderate/Hard	Hardest by far	Easy

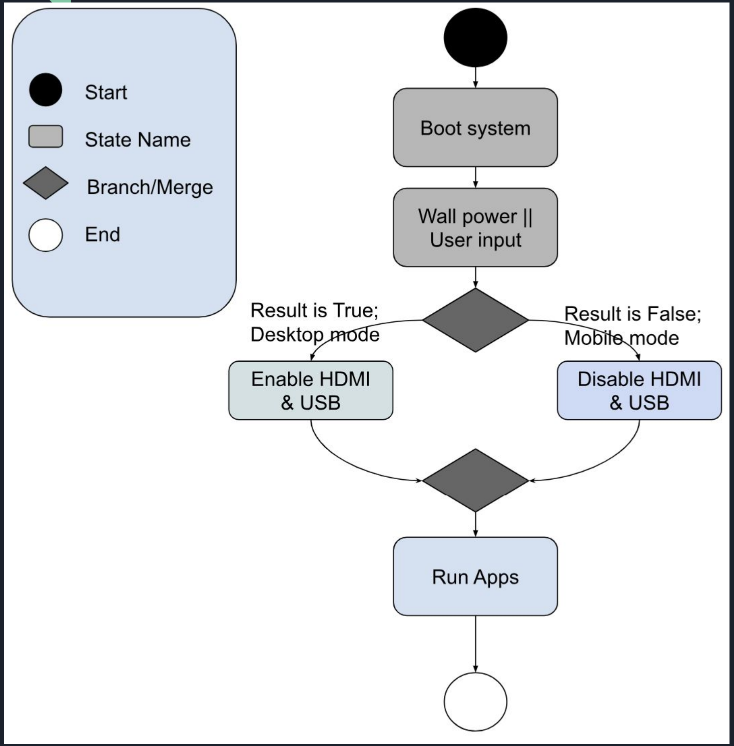
- Originally we had planned to use as much C++ as we could.
- But as development continued, we found the best selection would be using Python for high level features, and C++ for low level features.

Software Flowchart and Use Case Diagram



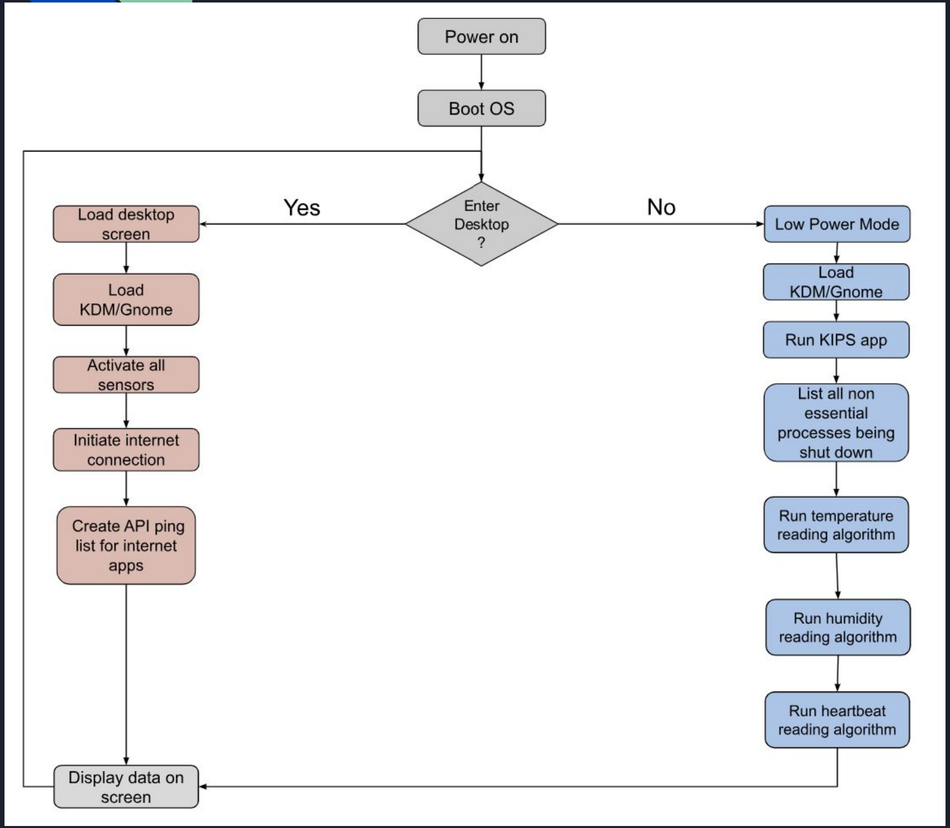


Software Activity Diagram



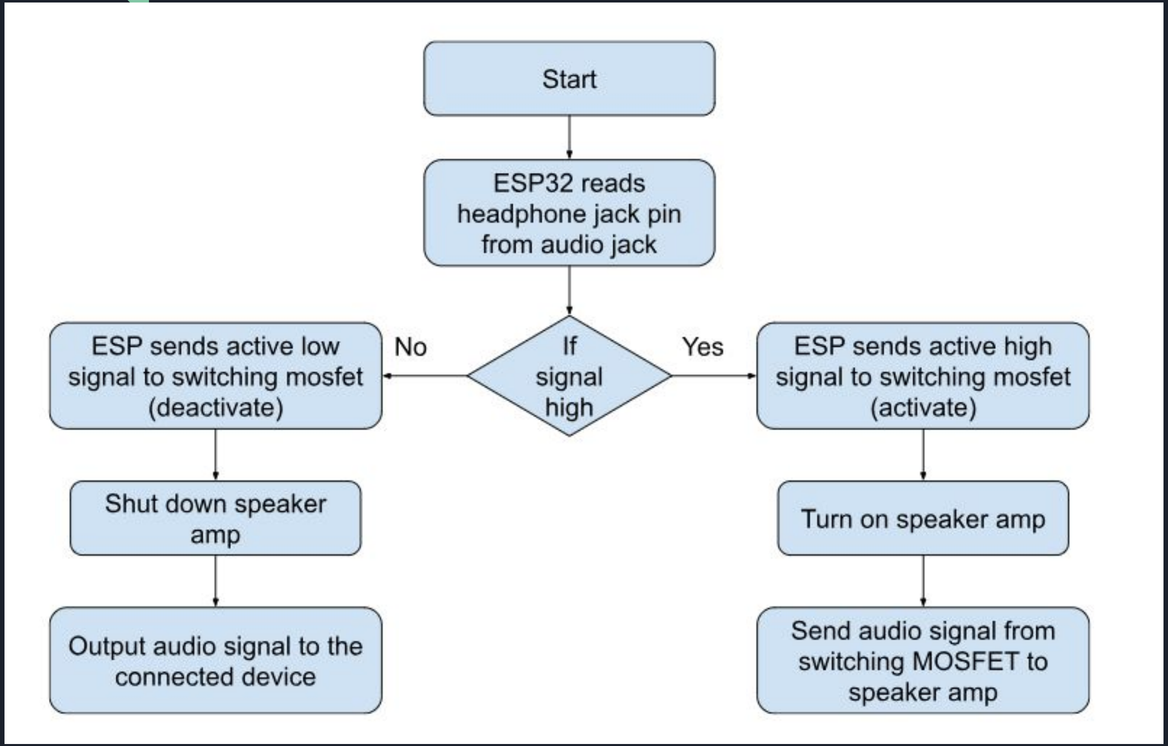
- The system works by deciding what power and operating mode to enter upon start up
- This is based upon being charging or not charging.
  - Charging means it will enter desktop mode. Or, the user can manually have the device enter this mode
  - No charging means it will enter mobile mode.
- Once it has decided, the applications in either mode will be loaded in as well.

OS Switching Algorithm



- The mode it enters is based upon the state the charging port is detected to be in when it first boots, or if it is toggled by the user.
- If it is plugged in, all systems will be activated including ports, sensors, cores, etc.
- If it is not, two cores are shut down, ports are inactive, and all nonessential processes are disabled.
- Then the KIPS application is launched to enter mobile mode.

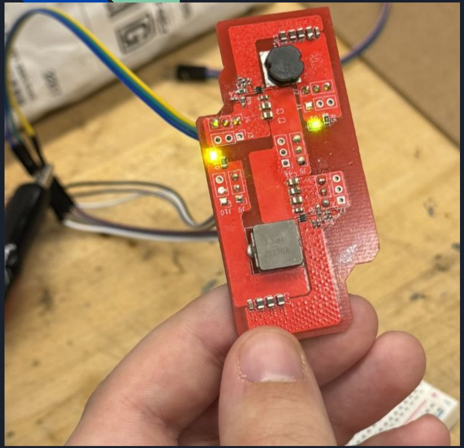
Audio Suite Algorithm

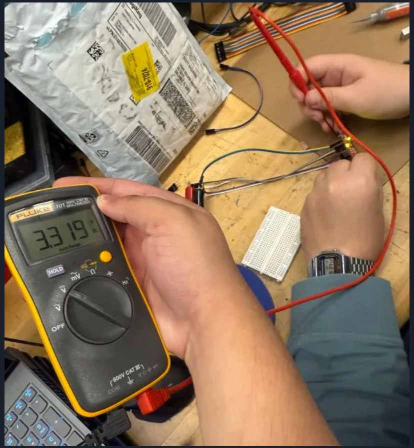


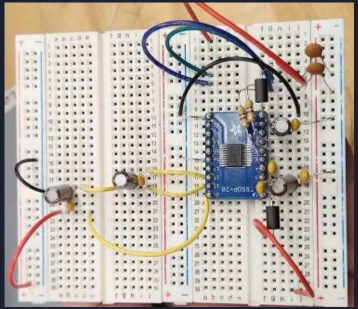
- The audio suite utilizes the 3.5mm audio jack pin.
- If it has a high signal, the ESP sends a low signal to shut down the speaker.
- Instead, it will send the audio signal to the audio jack.
- If the signal is low, the ESP sends a high signal to turn on the speaker and KIPS will instead send the audio signal to the speaker to play audio.

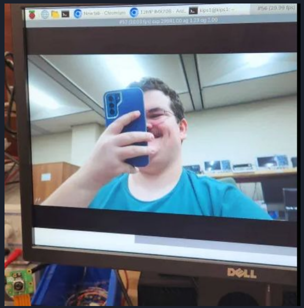
Main App Screen

Hardware Testing





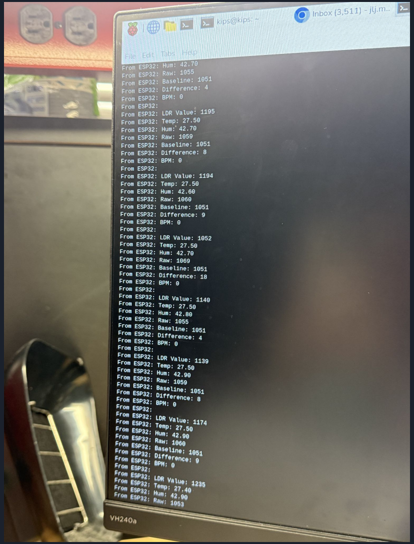


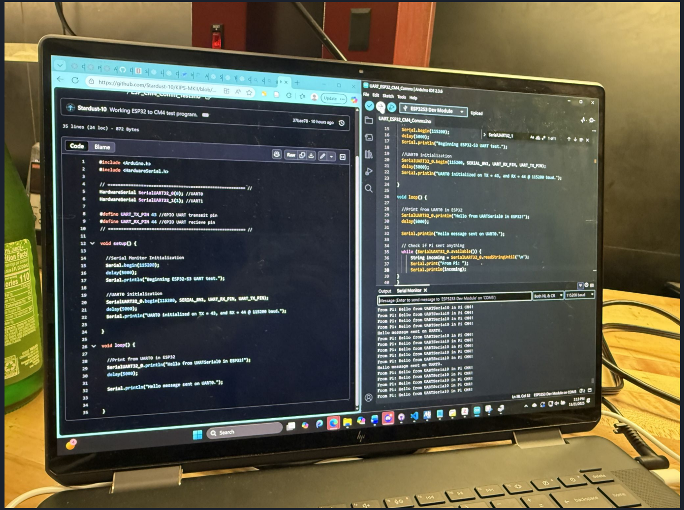


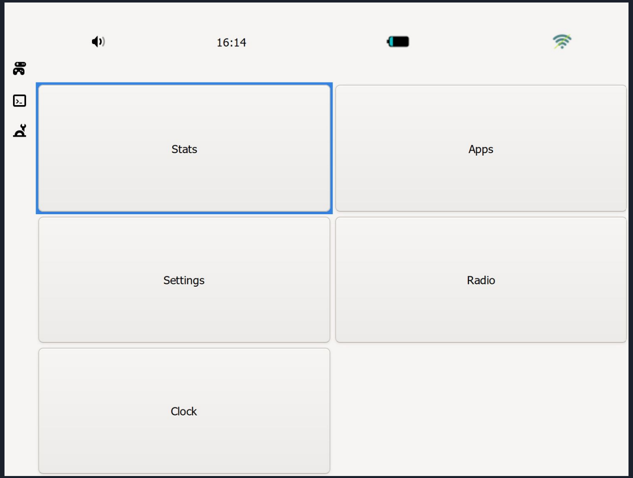
Hardware Testing Cont’d

Software Testing

The two main components that needed to be tested for software were the esp integrated software and raspberry pi front-end software.

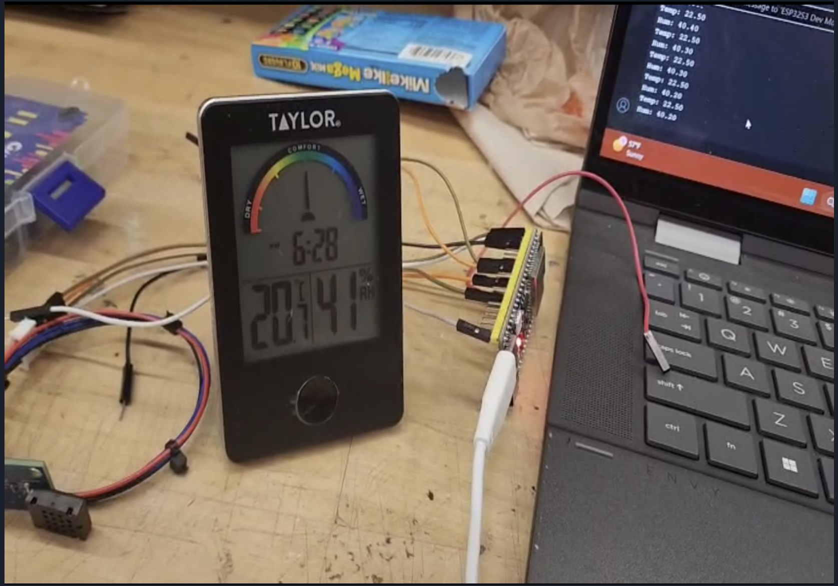






Software/Hardware Performance Evaluation

- We compared our sensors' measurements against tools and devices known to be accurate
- Sensor readings from the sensor suite are visible in the terminal in the top-right of the included image.
- The corresponding readings are shown on the device in the image.



Hardware Challenges Faced

Issue	Solution
3.3V Regulator Malfunction	Trace Repair
Regulator Voltage Drop	Designed for 5.2V Instead of 5.0V
HDMI Trace Complexity	Heavily Modified a Pre-Existing Design

Software Challenges Faced

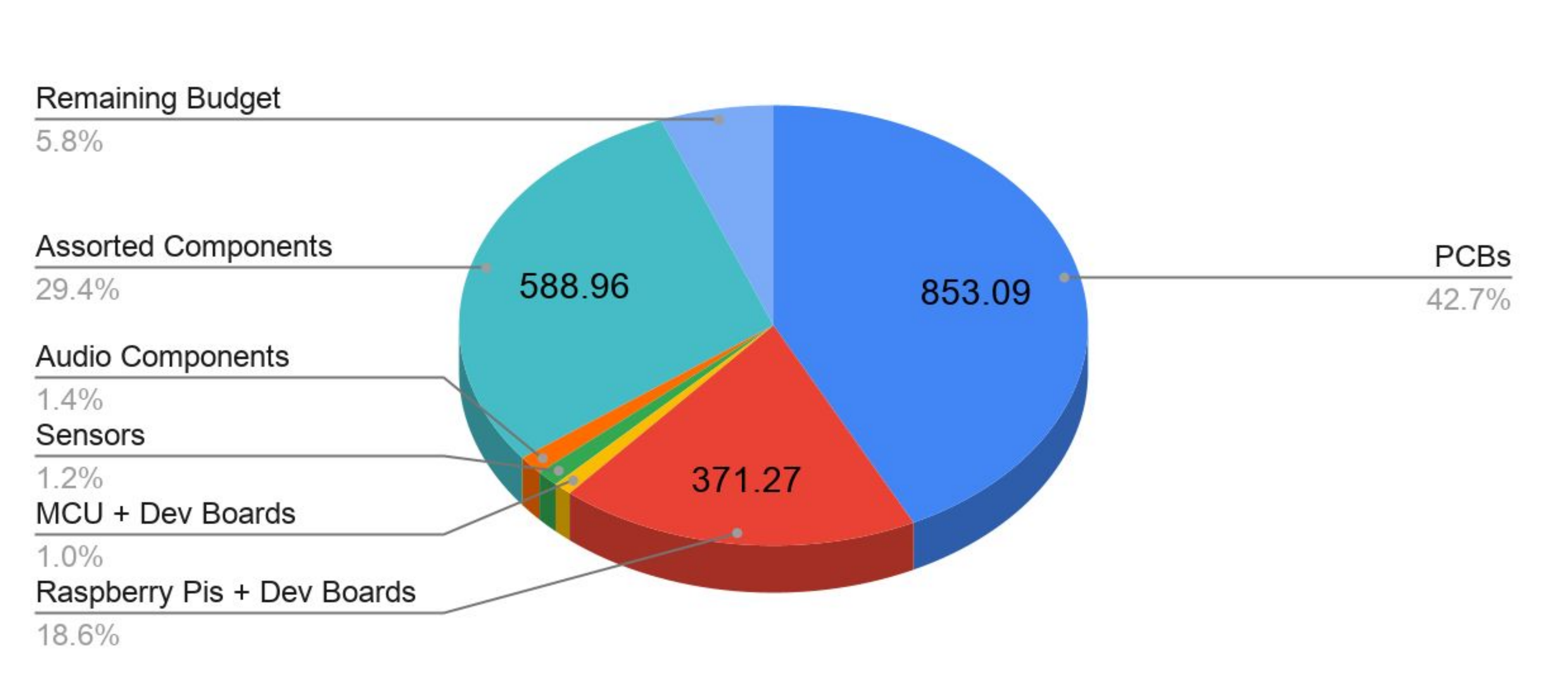
Issue	Solution
Faulty connection between ESP32 and Raspberry Pi	Rerouted bluetooth connectivity lanes to unassigned pins
Quirks in initial CSS styling	Established differentiation of object_ids and object_names; also began creating CSS groups
Icon and image import issues	Transitioned to local file paths and created custom icons to account for inconsistencies in default icons for different operating systems
Heartbeat sensor reading errors	Reread documentation and switched sensor from digital to analog output (still some lag)

What Didn't Work

- One of the biggest software challenges was the switch between mobile and desktop mode without rebooting the system.
- The regulator and the lack of preliminary protection on the regulator

Budget

Total Budget: $2,000



Distribution of Work

<table>
<tr><th></th><th>Sensors</th><th>PCBs</th><th>High Level Software</th><th>User Interface</th></tr>
<tr><td>Primary</td><td>Harrison</td><td rowspan="2">Scott</td><td>Jonathan & Harrison</td><td rowspan="2">Jonathan</td></tr>
<tr><td>Secondary</td><td>Jessenia</td><td>Jessenia</td></tr>
</table>

Thank you!