

K.I.P.S. Mk II

Knights Information Processing System Mk II



**College of Engineering
and Computer Science**

UNIVERSITY OF CENTRAL FLORIDA

Authors (Group 16):

Harrison Mills

Scott Valentine

Jonathan Meyers

Jessenia Argueta

Computer Engineering Comprehensive

Electrical Engineering RF/MW, Computer Science Minor

Computer Engineering Comprehensive

Computer Engineering Comprehensive

Mentor(s):

Dr. Chung Yong Chan

Sponsored By:

The K.I.P.S. Mk II Team

Reviewed By:

Dr. Sonali Das, Dr. Nazanin Rahnavard, Dr. Justin Phelps, Dr.
John Aedo, Mark Maddox

Table of Contents:

List of Figures:.....	7
List of Tables:.....	9
List of Equations:.....	12
1) Executive Summary: (TBA).....	12
2) Project Description:.....	13
2.1) Background and Motivation.....	13
2.1.1) Prior Project Experience.....	13
2.1.2) Anticipated Similarities.....	13
2.1.3) Existing Works.....	14
2.2) Goals and Objectives.....	15
2.2.1) Hardware Main Goals.....	15
2.2.2) Software Main Goals.....	15
2.2.3) Hardware Stretch Goals.....	16
2.2.4) Software Stretch Goals.....	16
2.2.5) Hardware Main Objectives:.....	17
2.2.6) Hardware Stretch Objectives:.....	18
2.2.7) Software Main Objectives:.....	18
2.2.8) Software Stretch Objectives:.....	19
2.3) Hardware Features.....	19
2.3.1) Software Features.....	20
2.4) System and Component Specifications.....	20
2.6) House of Quality Diagram.....	22
2.6.1) Hardware Block Diagram.....	23
2.6.2) Software Block Diagram.....	24
2.7) 3D Mockup.....	25
3) Research.....	25
3.0) Preface; SBC vs. MCU vs. FPGA.....	25
3.1) Single Board Computer (SBC).....	27
3.1.1) SBC Manufacturer Product Lines.....	27
3.1.2) Raspberry Pi SBC Series and Part Selection.....	29
3.2) Microcontroller Unit (MCU).....	32
3.2.1) MCU Manufacturers and Series.....	33

3.3) Sensors and Sensor Technologies.....	37
3.3.1.1) Heartbeat Detection Technologies.....	37
3.3.1.2) Heartbeat Sensor.....	38
3.3.2.1) Light Detection.....	40
3.3.2.2) Photoresistor.....	41
3.3.3.1) Temperature Detection Technologies.....	42
3.3.3.2) Humidity Detection Technologies.....	43
3.3.3.3) Temperature and Humidity Sensor.....	44
3.4) Peripherals.....	47
3.4.1.1) Physical Interface.....	47
3.4.1.2) Directional Pad (D-Pad).....	54
3.4.2) Joystick.....	56
3.4.3) Camera.....	58
3.4.4.1) Displays and Protocols.....	60
3.4.4.2) DSI LCD Screen.....	63
3.5) Audio Suite.....	64
3.5.1.1) SBC Digital Audio Signal Transmission.....	64
3.5.1.2) Analog Communications.....	68
3.5.2.1) AM/FM Radio.....	70
3.5.2.2) AM/FM Radio IC.....	72
3.5.3.1) Antennas.....	73
3.5.3.2) Whip Antenna/Loop Antenna.....	76
3.5.4.1) Signal Summation.....	78
3.5.4.2) Summing Amplifiers (Op-Amps).....	80
3.5.5.1) Volume Controls.....	81
3.5.5.2) Volume Potentiometers.....	83
3.5.6.1) Amps and Outputs.....	85
3.5.6.2) Headphone Amplifier.....	86
3.5.6.3) Speaker Amplifier.....	88
3.5.6.4) 3.5mm Headphone Jack.....	89
3.5.6.5) Speaker.....	90
3.6) Other.....	91
3.6.1) Flashlight Types.....	91

3.6.2) Flashlight.....	92
3.7.1) PSU.....	93
3.7.1.1) Battery composition.....	95
3.7.1.2) Battery.....	96
3.7.1.3) USB charge port (Power/Charging).....	98
3.7.1.4) USB-C Power Component.....	99
3.7.1.5) USB-C PD Controller.....	100
3.7.2.1) IC Functionalities.....	102
3.7.2.2) Charging IC.....	104
3.7.3) Battery Fuel Gauge IC.....	106
3.7.4.1) Wall power.....	107
3.7.4.2) Wall Power Safety Options.....	109
3.7.4.4) Power Source Switching.....	109
3.7.4.5) Switching MUX.....	111
3.7.4.6) MOSFET.....	113
3.8.5.1) Power Regulators.....	115
3.7.5.2) Switching Regulator Types.....	116
3.7.5.3) Buck Regulator.....	118
3.8) Communication Protocols.....	120
3.8.1) Internal Communication Protocols (part to part).....	120
3.8.1.1) I ² C Communication.....	120
3.8.1.2) Analog Communication.....	121
3.8.1.3) Other Communication Types.....	121
3.8.2) External Communication Protocols (part to peripheral).....	122
3.8.2.1) Video Communication.....	122
3.8.2.1.1) HDMI.....	123
3.8.2.2) USB 2.0 vs USB 3.0.....	124
3.9) Software.....	126
3.9.1) Operating System.....	126
3.9.2) Programming Languages.....	128
3.9.3) Version Control Software.....	132
3.9.4) Integrated Development Environment (IDE):.....	134
4) Standards and Design Constraints.....	138

4.1) Industrial standards.....	138
4.1.1) PCB Design Standards.....	138
4.1.2) I ² C Protocol Standards.....	141
4.1.3) UART Protocol Standards.....	142
4.1.4) Software Protocol Standards.....	143
4.2) Design Constraints.....	143
4.2.1) Time.....	143
4.2.2) Economic Constraints.....	144
4.2.3) Environmental Constraints.....	145
4.3) Other Constraints.....	146
4.3.1) Sustainability constraints.....	146
4.3.2) Ethical constraints.....	147
4.3.3) Health and Safety constraints.....	147
4.3.4) Manufacturability constraints.....	148
4.4) Software Constraints.....	149
5) ChatGPT.....	149
5.1) ChatGPT Prompt 1.....	150
5.2) ChatGPT Prompt 2.....	151
5.3) ChatGPT Prompt 3.....	152
5.4) ChatGPT Prompt 4.....	152
5.5) ChatGPT Prompt 5.....	153
5.6) ChatGPT Prompt 6.....	153
5.7) ChatGPT Prompt 7.....	154
6) System Hardware Design.....	155
6.1) Power Delivery System.....	155
6.2) SBC and MCU Connections.....	157
6.3) Sensors.....	158
6.3.1) Temperature and Humidity.....	158
6.3.2) Photoresistor.....	159
6.4) Ports/Peripherals.....	160
6.5) Flashlight.....	161
6.6) Audio Suite.....	163
7) System Software Design.....	165

7.1 Software Use Case Diagram.....	166
7.2 Software Activity Diagram.....	167
7.3.1 Heartbeat Polling Algorithm.....	168
7.3.2 Temperature and Humidity Polling Algorithm.....	169
7.3.3 Battery IC Polling Algorithm.....	170
7.4 Switching OS Algorithm.....	171
7.5 Audio Suite Algorithm.....	172
7.6 Display Layout.....	173
8) System Manufacturing.....	175
8.1 Motherboard PCB.....	175
8.2 High Speed PCB.....	176
8.3 Regulator PCB.....	177
8.4 Sensor PCBs.....	178
8.5 Control PCB.....	179
8.6) Flashlight PCB.....	180
9) Prototype and System Testing.....	181
9.1 Hardware.....	181
9.1.1) High Speed Board Testing.....	181
9.1.2) Regulator Board Testing.....	182
9.1.4) Camera Testing.....	185
9.2 Software.....	185
9.2.1) Sensor Board Testing.....	185
9.2.2) ESP32 to Pi CM4 Testing.....	188
10) Administrative Content.....	189
10.1) Financing and Budgeting.....	189
10.2) Bill of Materials.....	190
10.3) Distribution of Work Table.....	192
10.4) Milestones.....	193
11) Conclusion.....	193
APPENDIX A: REFERENCES.....	195
APPENDIX B: COPYRIGHT PERMISSION REQUESTS AND RESPONSES.....	196
APPENDIX C: AI/LLM CASE STUDIES.....	201
ChatGPT Prompt 1.....	201

ChatGPT Prompt 2.....	205
ChatGPT Prompt 3.....	210
ChatGPT Prompt 4.....	215
ChatGPT Prompt 5.....	216
ChatGPT Prompt 6.....	217
ChatGPT Prompt 7.....	218
ChatGPT Prompt 8.....	223
ChatGPT Prompt 9.....	225
ChatGPT Prompt 10.....	228
APPENDIX D: WORKING CODE:.....	232

List of Figures:

Figure 2.1: House of Quality.....	19
Figure 2.2: Hardware Block Diagram.....	20
Figure 2.3: Software Block Diagram.....	21
Figure 2.4: 3D Mockup of K.I.P.S. Mk II.....	22
Figure 3.1 Potentiometer Configuration [52].....	46
Figure 3.2: Magnetic Rotary Encoder Diagram [56].....	48
Figure 3.3: Optical Encoder Diagram [58].....	48
Figure 3.4: Laser Encoder Diagram [59].....	49
Figure 3.5: “Rule-of-Thumb Guidelines for PCB Signal Propagation Delay and Trace Length Matching”[72].....	63
Figure 3.6: “Figure 1 - A collision taking place on channel #2”[73].....	63
Figure 3.7: “Figure 2 - Missed packets in unsynchronized communication”[73].....	64
Figure 3.8: “PLL LCD Digital FM Stereo Radio Receiver Module 50Hz-18KHz Wireless Microphone Module DC 3-5V with LCD Display”[79].....	68
Figure 3.9: “Sony Corporation; TFM-6100W” by Eckhard Wegner[80].....	68
Figure 3.10: Dipole (Half Wavelength) Antenna Voltage Center by Andy Aka on Stack Exchange[86]... 71	
Figure 3.11: Whip (Quarter Wavelength) Antenna Voltage Center by Andy Aka on Stack Exchange [86]... 72	
Figure 3.12: “Jensen Transformers JT-123-FLPCH Line Output Transformer, PCB Mount”[95].....	76

Figure 3.13: “2 TI TLC2272AC 2 MHZ RAIL TO RAIL INPUT OUTPUT DUAL OP AMP IC NOS SOIC-8 USA”[96].....	76
Figure 3.14: “DIY stepped attenuator of the shunt type made with 0.5 watt metal-film resistors” by Dimitar Summerski from audio-at-home.com [101].....	80
Figure 3.15: “A typical single-turn potentiometer” [102].....	80
Table 3.38: Table of components that need power.....	91
Figure 3.16: “LTC4412 Datasheet” (reference datasheet for LTC4412).....	112
Figure 3.17: “Buck-Boost Converter" (reference recom-power).....	115
Figure 6.1: PD Board Schematic.....	154
Figure 6.2: A schematic of the daughter board for the regulators.....	155
Figure 6.3: A schematic of Main board which contains the ESP32.....	156
Figure 6.4: A schematic of the Temperature and Humidity sensor.....	157
Figure 6.5: A schematic of the Photoresistor.....	158
Figure 6.6: USB, HDMI, and FPC Ports.....	159
Figure 6.7: D-Pad and Thumbstick.....	160
Figure 6.8: Flashlight Schematic.....	161
Figure 6.9: Digital-to-Analog Converter and AM/FM IC.....	162
Figure 6.10: Summation Amplifiers.....	163
Figure 6.11: Switching MOSFET, Amps, Volume Knob, and Outputs.....	164
Figure 7.1 Software Use Case Diagram by Authors.....	165
Figure 7.2 Software Activity Diagram by Authors.....	166
Figure 7.3 Heartbeat Sensor Polling Algorithm by Authors.....	167
Figure 7.4 Temperature and Humidity Sensor Polling Algorithm by Authors.....	168
Figure 7.5 Battery IC Polling Algorithm by Authors.....	168
Figure 7.6 OS Switching Algorithm by Authors.....	169
Figure 7.7 Audio Suite Algorithm by Authors.....	170
Figure 7.8 Mobile Mode Display Layout.....	171
Figure 7.9 Preliminary Desktop Mode Display Layout Design.....	172
Figure 7.10 Working Desktop Mode Display Layout.....	173
Figure 8.1: 2D Motherboard layout.....	174
Figure 8.2: 2D High Speed PCB.....	175
Figure 8.3: 2D Regulator PCB.....	176
Figure 8.4: Temperature and Humidity sensor Board Layout.....	177
Figure 8.5: Photoresistor sensor Board Layout.....	177

Figure 8.6: Control Board Layout.....	178
Figure 8.7: 3D Model of Flashlight PCB Frontside.....	179
Figure 8.8: 3D Model of Flashlight PCB Backside.....	179
Figure 9.1: High Speed Board Testing.....	180
Figure 9.2: Regulator Board Testing.....	182
Figure 9.3: Regulator Board Testing 2.....	183
Figure 9.4: Camera Testing.....	183
Figure 9.5: Temperature and Humidity Sensor Testing.....	184
Figure 9.6: LDR Testing.....	184
Figure 9.7: Heartbeat Sensor Testing.....	185
Figure 9.8: All Sensors in One Print Testing.....	186
Figure 9.9: ESP32 to Pi CM4 Testing.....	187

List of Tables:

Table 3.1: SBC, MCU, and FPGA Comparison.....	23
Table 3.2: SBC Product Line Comparison.....	25
Table 3.3: Raspberry Pi Series Comparison.....	27
Table 3.4: Raspberry Pi Compute Module Models.....	29
Table 3.5: MCU Product Line Comparison.....	33
Table 3.6: Heartbeat Detection Technologies.....	35
Table 3.7: Heartbeat Sensors.....	37
Table 3.8: Light Detection Methods.....	38
Table 3.9: Photoresistor Models.....	39
Table 3.10: Temperature Detection Technologies.....	40
Table 3.11: Humidity Detection Technologies.....	41
Table 3.12: Temperature and Humidity Sensor Comparison.....	43
Table 3.13: Physical Interface Comparison.....	49
Table 3.14: D-Pad Model Comparison.....	52

Table 3.15: Joystick Model Comparison.....	55
Table 3.16: Camera Model Comparison.....	56
Table 3.17: Screen Types.....	58
Table 3.18: Screen Communication Protocols (Reference ChatGPT).....	59
Table 3.19: DSI LCD Screens (references above).....	60
Table 3.20: Signal Transmission Options.....	64
Table 3.21: Digital-to-Analog Converter Comparison.....	66
Table 3.22: AM/FM Radio Modules.....	69
Table 3.23: AM/FM Radio IC Comparison.....	70
Table 3.24: Antenna Type Comparison.....	73
Table 3.25: Whip Antenna Comparison.....	74
Table 3.26: Signal Summation Options.....	76
Table 3.27: Op-Amp Comparison.....	78
Table 3.28: Volume Control Options.....	81
Table 3.29: Volume Potentiometer Comparison.....	82
Table 3.30: Amplification Options.....	84
Table 3.31: Output Types.....	84
Table 3.32: Headphone Amplifier Comparison.....	85
Table 3.33: Audio Amplifier Comparison.....	86
Table 3.34: 3.5mm Headphone Jack Comparison.....	87
Table 3.35: Speaker Type Model.....	88
Table 3.36: Light emitter Model.....	89
Table 3.37: LED Model.....	90
Table 3.39: Battery Composition Type Model.....	93
Table 3.40: Battery Configuration Type Model.....	95
Table 3.41: Charge Port Type Model.....	96
Table 3.42: PD Controller VS NO PD Controller Comparison.....	97
Table 3.43: PD Controller Comparison.....	99
Table 3.44: Battery IC Configuration Comparison.....	101
Table 3.45: Charging IC Comparison.....	103
Table 3.46: Battery Power Gauge IC Comparison.....	104
Table 3.47: AC Adapter Comparison.....	106
Table 3.48: AC Safety Feature Comparison.....	106

Table 3.49: Power Switching type Model.....	108
Table 3.50: Power Switching device Model.....	110
Table 3.51: MOSFET Comparison.....	112
Table 3.52: Linear vs Switching Regulator Comparison.....	113
Table 3.53: Buck Regulator Comparison.....	117
Table 3.54: Output Types.....	121
Table 3.55: HDMI Type Comparison.....	122
Table 3.56: USB Type Comparison.....	123
Table 3.58: Onboard Operating System Comparison.....	126
Table 3.59: Programming Languages Comparison.....	129
Table 3.60: Version Control Software Comparison.....	132
Table 3.61: Embedded Tools and Frameworks Comparison for Visual Studio Code.....	134
Table 3.62: Integrated Development Environment (IDE) Comparison.....	135
Table 10.1: Budget.....	187
Table 10.2: Bill of Materials.....	188
Table 10.3: Distribution of Work Table.....	189
Table 10.4: Milestones.....	189

List of Equations:

$er(dB)=10\log_{10}(1- Z_{load}-Z_0 Z_{load}+Z_0 2)$ (reference Antenna Theory 4th ed) (1)	67
$Lf=\mu_0\mu_e N^2 A l f$ (insert reference paper) (2)	68

1) Executive Summary

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. Originally inspired by a fictional wrist-mounted device from a popular video game franchise, KIPS Mk. II adapts this concept into a fully realized, real-world platform that merges the capabilities of a compact personal computer, a smart device, and an environmental sensing system. Where the Mk. I served as a proof-of-concept demonstrating feasibility, the Mk. II represents a significant evolution of functionality, design and usability from the Mk. I prototype.

The KIPS Mk. II is designed as a portable personal computer with flexibility in orientation for mobile, on-the-go use and desktop use. It operates as both a mobile smartwatch-like device and a desktop-class computing platform, switching dynamically between the two depending on power source or user enabled switching. When powered by battery, the system defaults to a mobile mode where a low-power mobile operating system and GUI is launched, whereas when attached to wall-power, the system enters desktop mode where a full desktop Linux environment is launched and peripheral interfaces (such as HDMI and USB) are enabled, however the switch between modalities can be initiated by the user as well.

One of the core focuses and features of KIPS Mk. II is its robust sensor suite integration, which includes an onboard heart rate sensor, a combined temperature and humidity sensor and a daylight sensor for screen brightness with auto-dimming. These sensors interface directly with the ESP32-S3 microcontroller unit (MCU), which enables the coordination of sensor data acquisition and transition to the Raspberry Pi Compute Module 4, which serves as the single board computer (SBC) of the project. Additional hardware features include AM/FM radio capability, an LED flashlight, D-pad controls, and digital-to-analog audio output.

In this document we detail the KIPS Mk. II design process that led to its complete and final implementation. First, the background and situation of KIPS Mk. II is discussed within the context of previous works, including relation to KIPS Mk. I with projected similarities and improvements. The goals and objectives of the project are discussed next paired with the various features, specifications and proposed software and hardware breakdown diagrams. Afterward, we discuss the research behind the various required technologies, software, frameworks, and part selection that made production possible and encompassing every individual component and system. Next, we move to the standards and constraints that apply to the entirety of the project in addition to the various hardware and software components that make it up. We then discuss the ethical contributions and leveraged usage of Artificial Intelligence (ChatGPT specifically) in the planning and development of this project. The section following this discusses the hardware design of the various PCB schematics for the KIPS Mk. II, and the next section details overall system software architecture design and current implementation. Additionally, we include the system manufacturing information, which goes over PCB design choices and intended

characteristics during device operation. This is followed by the prototype system testing, which was conducted by first testing individual hardware components and software, then moving on to testing how they would work and communicate with each other via breadboards. The next section entails all of the administrative content of the project such as budgetary needs, work distribution, and milestones. Lastly is the conclusion which is followed by the appendices, containing various supplementary information and graphics that are too extensive to include in the main portion of the report.

2) Project Description:

2.1) Background and Motivation

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

2.1.1) Prior Project Experience

K.I.P.S. Mk I was a project started for the 2024-2025 UCF IEEE Internal Projects Competition. It exists as a physical "rough draft" of the actual vision of K.I.P.S., which led to us wanting Mk II to be our project for senior design. K.I.P.S. Mk I operates off of a standalone Raspberry Pi 4 8GB model running Raspbian OS. It was intended to have physical controls on top of the touch screen in the form of rotary encoders as scrollers; one horizontal, and one vertical. There was also a button that was meant to act as an "enter key" of sorts. Unfortunately, due to a last-minute issue with the code before our presentation, the controls had to be scrapped, but I have learned from this mistake and later in this paper will discuss a new control scheme. Other features of K.I.P.S. Mk I included an I/O port, basic battery power, a rough mobile UI programmed in rust and python, and an exhaust fan for cooling.

2.1.2) Anticipated Similarities

We intend on the system to still have I/O ports, some form of exhaust for cooling, I/O ports (although in more convenient spots), and touchscreen input. This project will most likely run on Raspbian OS but we are open to other Linux options.

2.1.3) Existing Works

There are several products that exist that we are taking features from to inspire KIPS MkII

KIPS MK1

As previously mentioned, KIPS MkI is an existing product that was built for IEEE's 2024-2025 internal projects competition. It featured a Raspberry Pi 4 which housed all of the ports and

GPIO pins, ran off of a 10000mAh 3.6v LiPo battery, and had scroller controls for ease of navigation in mobile mode. These sound great in theory, but in practice, all of this caused a lot of unnecessary bulkiness that led to the MkI prototype model being very top heavy. The GPIO pins sticking up made wiring very awkward, and all of the IO being in one place also caused unnecessary awkwardness with using peripherals. The scroll controls ended up not working well either, as incorporating them into the shell design did not work too well due to the round scrollers and some other issues with mounting of the rotary encoders. However, the chief idea worked. It was a wrist mounted portable PC capable of running a mobile and desktop mode, effectively bridging the gap between a pocket PC and a smart watch. We are going to implement and improve upon the mobile design by reducing the form factor of both the circuitry and the shell.

Pocket PC

Older model pocket PCs were a big inspiration for KIPS. Pocket PCs were a product that were the progenitor to modern smartphones. They gave users the ability to make phone calls, use the internet, use various office softwares, games, etcetera. They also had a proprietary mobile operating system made to be based off of existing operating systems for desktops. This gave them immense usability and familiarity with people picking them up for the first time. Basing the mobile OS on a desktop OS also allowed for developers to implement a desktop style file management system like we have today. While the features and OS being familiar to those who used them is good, all of these were brought down by a clunky and stylus based interaction. No keyboards or mice were able to be attached to them due to them being PDA styled, as well as hardware limitations of the time. The proprietary nature also made it incompatible with anything that was compatible for regular operating systems of the time, which means any program that ran on the pocket PC was also proprietary in nature, making development difficult. We would like to integrate a familiar environment for users to navigate, as well as the touchscreen design of pocket PCs while improving the idea by adding buttons for navigation while being worn. This was previously done on KIPS Mk I but did not translate well due to there being no stylus, and the shell design hindering access to screen edges.

Smart Devices

Another pair of existing products that inspired KIPS is the modern smartphone, and the tablet. These products both use customizable mobile operating systems to offer some, but not all, of the functionality of a laptop or desktop PC. They are compatible with all manner of media forms, as well as playback of said media. They also have the ability to play games, draft documents, access the internet, etcetera. They are also quite powerful. However, there are some drawbacks to these devices. They tend to be quite expensive, and in a lot of cases, more expensive than a desktop or laptop that could outperform the phone or tablet by a longshot. This leads into the second drawback, ecosystems. Mobile devices tend to lock users into an ecosystem controlled by the company who owns the OS that the manufacturer used for their devices, and the companies also

tend to charge a premium for their name, as well as the ecosystem you choose. Falling into an ecosystem also leads to some apps not being available on other systems, accessory compatibility issues due to protocols, lack of expandable storage nowadays, and data privacy issues as well. We wish to incorporate the customization, mobile form factor, and media compatibility of smart devices into KIPS Mk II for a unique mobile computing experience.

2.2) 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.

2.2.1) Hardware Main Goals

- Design a wrist wearable device that serves as a marriage between a pocket PC and a smart-watch.
- Give the user a light sensor
- 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. The type of port used for the charging will be USB-C.
- Allow the user to display the contents of our device.
- Enable user navigation of KIPS Mobile Mode via directional pad (D-Pad) inputs through the X and Y axes of the screen.
- 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

2.2.2) 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.

2.2.3) Hardware Stretch Goals

- Implementing a Bus Controller to include a USB 3.0 Port
- Include an LED Flashlight with a simple resistor-wire connection

2.2.4) Software Stretch Goals

- Implementing a few games
- Creating Distinct User Profiles for the Consumer

The overarching objective of this project is to essentially create a wearable, functional computer that can serve as a portable desktop in terms of computing power. Additionally, it will have certain features and capabilities not found on standard computers or mobile phones, such as an onboard heart monitor and temperature sensor. One of its most appealing features is to have the power of a desktop computer, with the charging abilities of a mobile device, allowing for easy and convenient access to the device at all times. The device will also be capable of taking HDMI input to move its display onto a larger screen if needed.

The main objective of the sensors is to be able to achieve consistently accurate data gathering to send to the MCU. This is due to the fact that other programs will be relying on this information to use for their own purposes. For example, the daylight sensor will be used by the device in order for it to know how to adjust the screen brightness during the daytime. Many of these readings will also be available for the user to see as well, in particular, the temperature, humidity, and heart monitor readings will be accessible to the user via the device's UI.

Physically, the objective of KIPS is to be versatile in how the user may choose to interact with it. For these reasons, it will have two main interaction methods, which are a touch screen and a D-pad input. A major goal in the physical design will also be to have it be much lighter and lower profile compared to the previous model, so the user will have a much easier time wearing the device.

The device will have various different software features in its programming, with applications meant to mimic a smartwatch. When it is powered on, the device will determine whether to enter desktop or mobile mode depending on if it is charging or not. The user will later have the option to switch between operating systems if they wish. Applications will communicate with the MCU, and the user can make their own additions to the applications, such as writing their own, if they wish. The sensor readings will be accessible regardless of operating system mode.

2.2.5) Hardware Main 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

For the I/O board, we hope to implement 1-2 HDMI ports for video output to monitors, 1-3 USB 2.0 ports to support peripherals, a directional pad and an OLED or LED touchscreen for user input in mobile mode. For audio output we hope to implement a Digital-to-Analog-Converter, an amplifier, and volume control that connect to a small speaker.

The 3-D printed shell will be designed to house the battery, the main board, and the sensors. It should have weight distribution to ensure ease of wear.

In the case of battery power, it is imperative that we accurately acquire the appropriate amount of voltage and amperage capacity needed for powering all of the components.

The next task is the simple inclusion of a push button or switch that will toggle the operation of the entire unit on/off. This button is known as the power button and should be an easy part to signal when the device is to be powered/sleep/off. In order to do so, we intend to use the Microcontroller to detect the debouncing of the button. And a step-down regulator to ensure the appropriate amount of voltage is supplied to the model.

Three sensors are to be included within our model, which will be a: Heartbeat Sensor, Temp/Humidity Sensor, and Daylight Sensor.

2.2.6) Hardware Stretch Objectives:

- ❖ USB 3.0 Port
- ❖ LED Flashlight
- ❖ Front Facing Camera

We are also looking into the possibility of incorporating a front facing camera as part of our model.

We are intending on implementing an analog AM/FM Receiver by using a digital input via an application. While having the receiver itself as more of an analog output for setting the correct frequency. More specifically, using a sort of summation amplifier to combine computer output and radio output.

Though it may not be necessarily an important goal to include a USB 3.0 port. Having an even faster more efficient USB port could be a nice addition to our unit. This requires a controller chip, as seen inside of the pi CM4 manual.

Another suggestion for us to include within our build is a low-powered LED flashlight that should be controlled through a switch.

2.2.7) Software Main Objectives:

- ❖ Develop and include various mobile-oriented applications
- ❖ Write software protocol for switching between mobile and desktop operating systems

Our device will support a variety of different applications in order to maximize the utility of the device. These applications will be presented using a UI that is similar to the style of modern mobile menu UIs. Each application will have its own unique icon and when it is tapped, it will activate the application. The core of the system settings will be under the settings application, where volume, screen brightness, Wi-Fi, storage, and device name. A top ribbon on the corner of the screen will contain statuses for the time, date, and battery percentage. Native to the device, there will be a heart monitoring application, image gallery, and file manager. Additionally, there will be API applications for weather, music, a calendar, and maps. When the device is idle for extended periods of time, there will also be a sleep screen mode to further conserve battery in mobile mode.

As the device boots up, it will determine whether to enter mobile or desktop mode based on its power source. We will implement a software protocol that detects whether the device is running on battery power or wall power. If operating on battery power only, it will default to the mobile OS, which is better oriented for low power consumption by disabling certain background processes, such as power-intensive peripheral polling and desktop-oriented features (specific processes to be finalized during development). On the other hand, if the device is plugged into wall power it will launch the desktop OS, enabling all features for full functionality. Users will also have the option to manually transition between modes via a dedicated software toggle within the UI, allowing flexibility regardless of power state.

2.2.8) Software Stretch Objectives:

- ❖ Basic games
- ❖ Profiles

We hope to include various lightweight games, which will be highly portable in order to add to the overall user experience.

Using a unique ID, we aim to create distinct user profiles such that individual users can personalize various aspects of the UI. These customizations would come in the form of theme selection, wallpaper customization and other preferences such as font size and icon layout/size.

2.3) Hardware Features

The hardware components of this device consists primarily of sensors, amplifiers, USB ports, and button layouts.

The single board computer was the main contributor for dictating what would be necessary for the specific components needing to be included within the build. It also doesn't go without mentioning that the hardware aspect of the assembly was mildly significant as it also affected the plan of how certain functionalities will be programmed. One main example being on how it affects the number of regulators required for the voltage to be stabilized and managed efficiently. Our project device needs a screen for displaying all of the applications used for certain tasks such as the potential radio implementation, heart rate sensing app, weather app, battery life level, etc. And this particular screen will need to be touchscreen in order to have the ability to be controlled by the user for input, alongside with a directional pad as a secondary. The speaker is imperative for playback on certain apps like games that are to be pre-installed, and the same can be said for why volume control is necessary. The power button is self-explanatory and will be needed for on/off operation control. And in speaking of power, finding the most optimal battery/batteries needed for the gadget will be a task in itself.

As for ports, the USB Type A ports are to be included for storage devices, syncing data, or regular charging of certain other devices. So far the current plan is using 2.0, but 3.0 ports is a stretch goal. The micro USB-C port is essentially for charging the main device itself. The Mini HDMI port will be an output that can be used for displaying the device as well as an alternative to using the already provided screen. This is due to the SBC chosen to have the capability of scaling up to 4K resolution.

Next are the sensors needed in order for a great majority of these functions to operate. The heartbeat sensor for detecting the pulse rate on the user's arm (possibly located somewhere

around the wrist strap), an ambient sensor for detecting humidity, and a daylight sensor for helping with the screen's auto brightness.

Last but not least is both the camera and the LED light. The camera will be for picture taking, all while the LED is to be a simple connection and to be used as a flashlight.

2.3.1) Software Features

Most of the software component of the device will be split into two sections: Applications, and stretch goals.

The settings application will consist of major functionalities that can be controlled and adjusted by the user. The first application will be the settings app, which would contain options for a dark mode, the volume, brightness, and sleep/power mode.

All other applications will be more auxiliary. Many of these will be API calls to display information. The list of planned files are a weather, music, calendar, map, and image gallery file. Only the image gallery will not rely on an API, all other files will. The music and image gallery files will not be connected to any form of wi-fi, meaning the user will be using data within them that was already downloaded prior to use.

Next are the stretch goals, these are features and applications that will be added if time permits. These include Doom, Minesweeper, theme/wallpaper customization, profiles/ID, and transferable user settings. These are mostly for aesthetics, games, or can make the user experience easier. Regardless of whether or not any of the stretch goals are achieved, they are purely for additional functionalities, and will not affect the core processes of the device.

Beyond the main three sections, there are other components the UI will have as well, such as a top ribbon with icon displays that will allow the user to know basic information about the state of the device at a glance, similar to a mobile phone when operating on the mobile OS.

2.4) System and Component Specifications

Table 2.1: System Specifications

Desktop Battery Life	> 2 Hours
Mobile Battery Life	> 5 Hours
Dimensions	~ 6 in. x 4 in. x 2 in.

Weight	< 4 lbs
Mobile to Desktop Conversion Time	< 20 seconds
Desktop to Mobile Conversion Time	< 20 seconds
Cost Per Unit	< \$300
Power Consumption (Mobile)	< 10 Watts
Power Consumption (Desktop)	< 20 Watts
Boot Time	< 10 seconds
Avg. System Operating Temp.	95°F - 120°F
Response time/ time delay	10 seconds?

Table 2.2: Component Specifications

Temperature Sensor	Detectable Range	(-20°F) - 115°F
	Accuracy	(+/- 2°F)
Humidity Sensor	Detectable Range	0% - 100%
	Accuracy	(+/- 5%)
Heartrate Sensor	Measurement Range	40 < x < 150 BPM
	Accuracy	(+/- 2 BPM)
Screen	Resolution	800 x 480 pixels
Photoresistor	Resistivity Range	1kΩ < x < 1MΩ
Battery	Capacity	> 8000mAh, 3.6V
Flashlight LED	Brightness	100 lumens < x < 1000 lumens

2.6) House of Quality Diagram

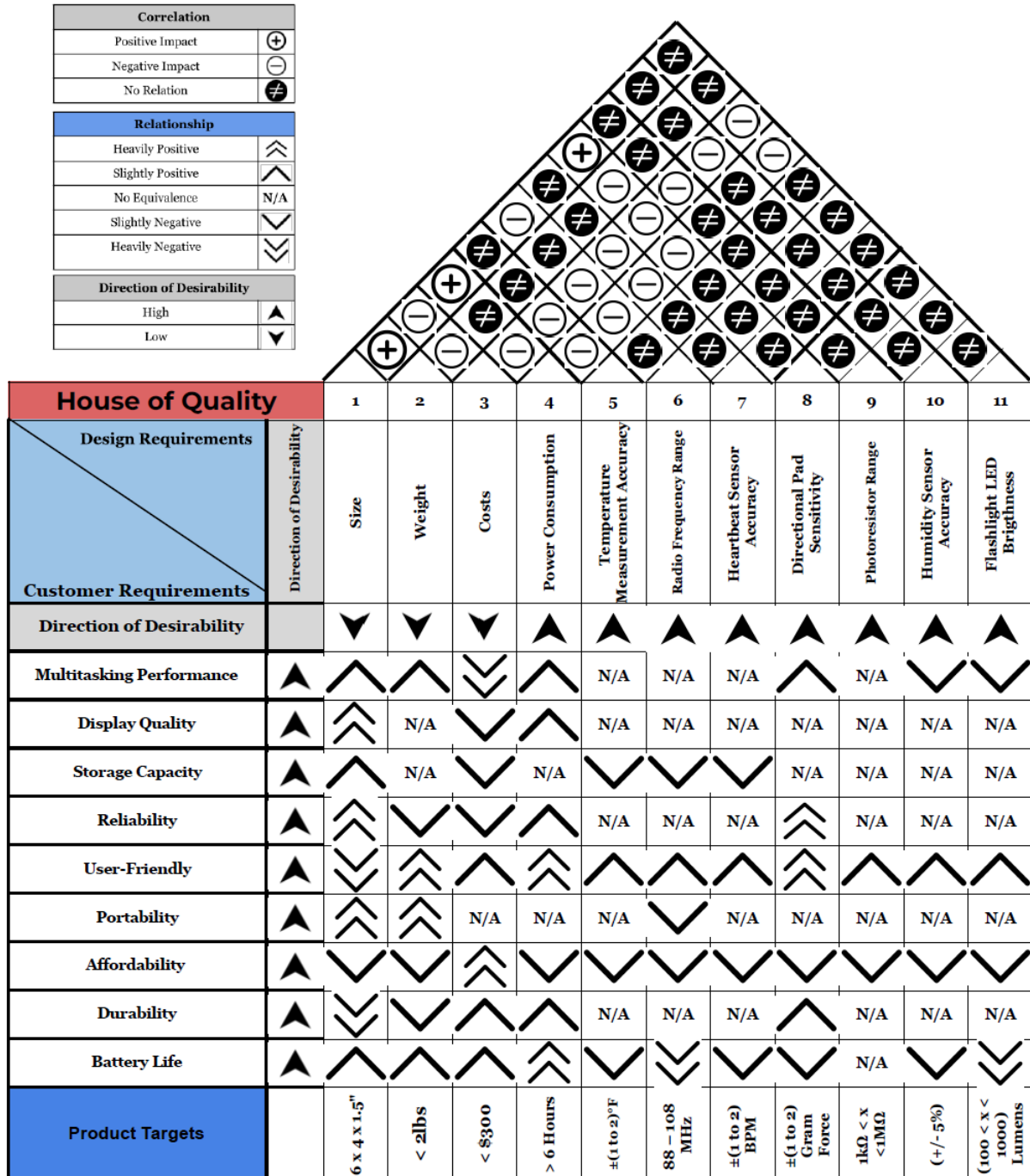


Figure 2.1: House of Quality

2.6.1) Hardware Block Diagram

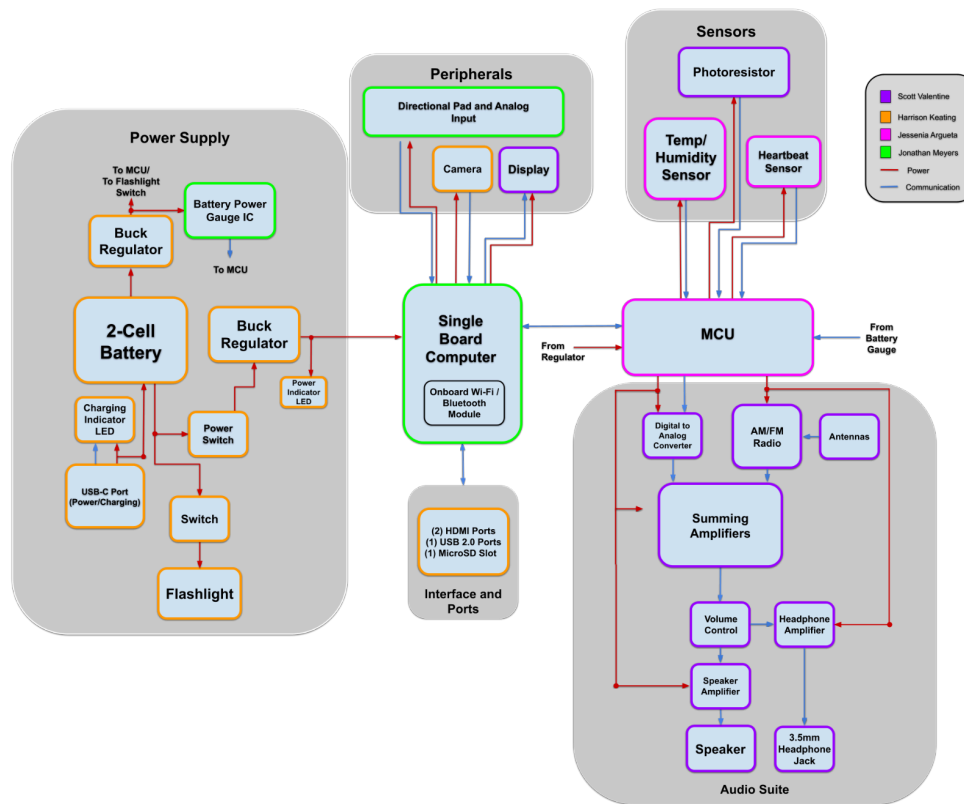


Figure 2.2: Hardware Block Diagram

2.6.2) Software Block Diagram

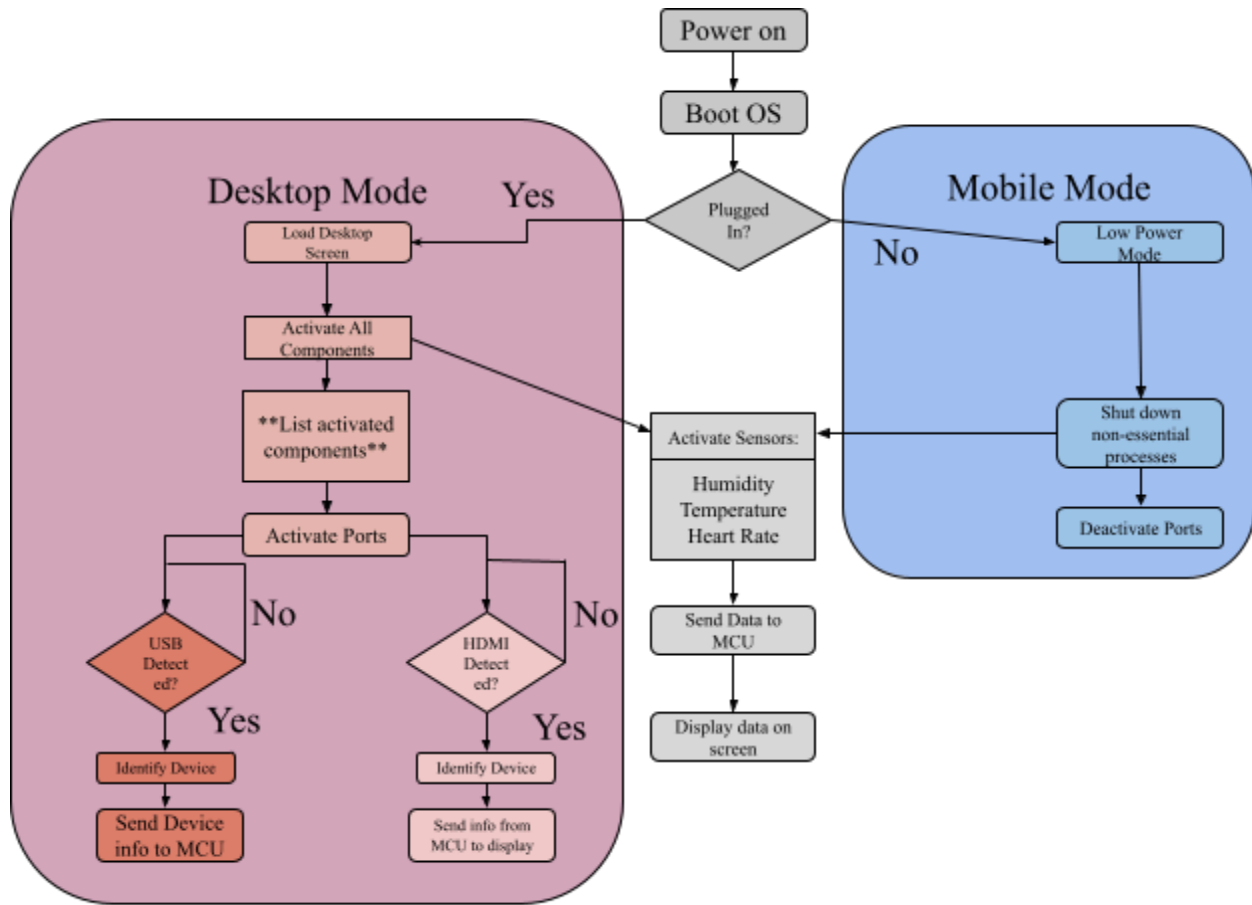


Figure 2.3: Software Block Diagram

2.7) 3D Mockup

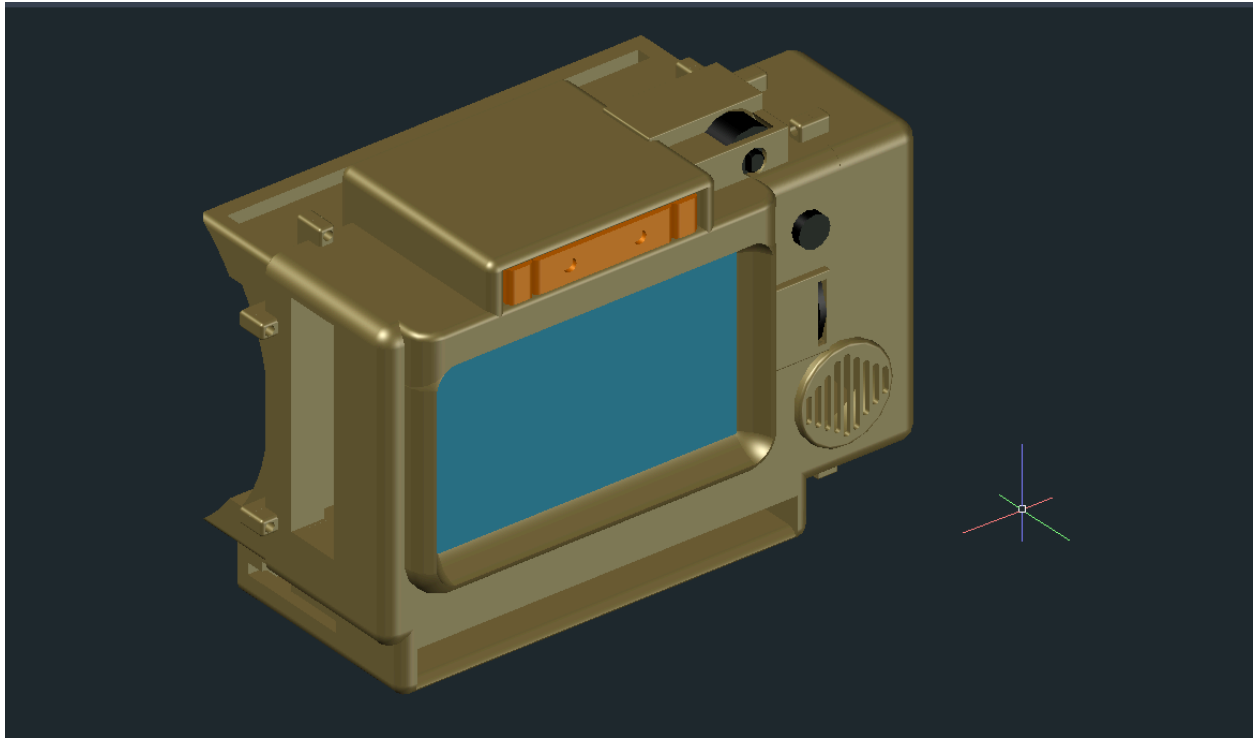


Figure 2.4: 3D Mockup of K.I.P.S. Mk II

3) Research

3.0) Preface; SBC vs. MCU vs. FPGA

As we continue to design our device, we consider three distinct types of embedded platforms in field-programmable gate arrays (FPGAs), single-board computers (SBCs) and microcontrollers (MCUs) to examine their individual strengths, weaknesses and also to determine how suitable each is for use in K.I.P.S. Mark II.

FPGAs are excellent at real-time parallel processing and their digital logic can easily be tailored for different uses. This could allow us to design a streamlined sensor input pipeline or control connected hardware with little-to-no latency. However, FPGAs lack built-in operating system support and require an additional processor just for general-purpose computing. FPGAs also draw a large amount of power and are typically listed at the highest price between the SBC, MCU and FPGA. Overall, we feel that an FPGA would be excessive for this project as we do not require the specialized data pipelines or incredibly low signal processing latency. The dimensions of most FPGA models are also much too large to be safely and securely housed within the desired dimensions of the KIPS Mk. II without modifying them.

MCUs are fairly simple, requiring little time to boot and are ideal for interfacing with sensors/peripheral connections in real-time. MCUs also consume a low amount of power in comparison to the other components which is practical for running the system on battery power or in a reduced power setting. Some of the shortcomings of MCUs however are the fact that they cannot handle HDMI output which is required to mirror/extend the display of the device to other connected monitors and screens. Another thing to note is that MCUs cannot run a multitasking OS meaning that multiple processes such as collecting sensor data and user interface with the touchscreen display would be unable to run concurrently and would greatly reduce the functionality of the device itself.

SBCs support HDMI and USB connection, both of which are central features of our project in order to interface with peripheral devices such as keyboards, mice, and additional monitors. An SBC can also run a full Linux OS distribution with multitasking, which will be essential in allowing multiple processes to run concurrently (such as the touchscreen and sensor data example above). Where the SBC falls short is its slower boot-time compared to the MCU and difficulty consistently communicating with sensors in real-time. Additionally, SBCs have a higher power draw than MCUs which is not ideal for working in low-power environments.

Table 3.1: SBC, MCU, and FPGA 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

Each of these options provide their own unique advantages, however none of these components on their own would be enough to fulfil all of the requirements that our project calls for. By combining an MCU and an SBC, we are able to satisfy these goals in their entirety. In this case the SBC would handle file storage, HDMI connection, as well as high-level logic and the MCU would manage real-time sensor interaction, GPIO and hardware-level interfaces. To solve the issue of the higher power draw of the SBC, we will reduce usage of the SBC's high power

functions and connected peripherals when in mobile or low power mode, utilizing mainly the MCU for its low power draw and in instances when low-power usage is not as important (such as in desktop mode, where wall power is connected) we will reactivate the entirety of the SBC for higher performance and added functionality.

3.1) Single Board Computer (SBC)

For our project we require a Single Board Computer (SBC) to serve as the central computation module for our device. The SBC will be used for a variety of different functions, including running the main operating system and managing high-level tasks such as the touchscreen display and various I/O operations. The SBC will also be responsible for communicating with the MCU in order to collect and process sensor data in addition to controlling the peripheral components of the device such as the USB peripherals and HDMI output. In this section we explore a variety of the most viable SBC product lines for this project to gain a better understanding of what each can contribute to our project. Our intention is to compile and discuss the relevant specifications for the array of SBCs available and eventually narrow down the selection to a single model of best fit.

3.1.1) SBC Manufacturer Product Lines

We begin by examining works adjacent to ours to view the SBCs others have selected. We find that some of the most widely used SBCs in similar projects are the Raspberry Pi series, Radxa's Rock Pi Series, and the BeagleBone series.

Raspberry Pi is a great candidate for this project as it offers a wide variety of configuration options and software resources. Of the three options, Raspberry Pi is arguably the most user-friendly and possesses the most extensive documentation. The Rock Pi line by Radxa is also a contender with higher-end processors available, and can handle more computationally expensive tasks. Lastly there is the BeagleBone line which is better suited for precise I/O control.

Because there are several different boards for each line specified, some research was done to narrow down the available product lines into a few baseline models. For Raspberry Pi the Raspberry Pi 5, CM5 and Zero 2 W were taken into consideration as the state of the art (SOTA) models for Raspberry Pi. These models fully encapsulate the capabilities of Raspberry Pi. The specifications of these models were fairly similar overall and are included in the table below. The Zero 2 W is the most lightweight and least capable board offered by Raspberry Pi of these three models and is reflected in the table by the lower-end specifications. Raspberry Pi's 5 and CM5 are both highly capable, making up much of the higher-end of the range of specs [1], [2], [3]. For the Rock Pi line the Rock Pi 4C+ and Rock Pi S0 were considered as they are some of the most recent SBCs and also show the range of functionality [4], [5]. Lastly, for the BeagleBone models, the BeagleBone Black and AI-64 were included in the comparison. The capabilities of the AI-64

are much higher with hardware specifically designed for computationally-expensive AI and ML application and computation. This is definitely the higher-end model of the BeagleBone line of products and is much more expensive in comparison, making up the majority of the upper price range for BeagleBone products. In comparison, the BeagleBone Black is much more practical for usage in real-time sensor applications and less AI-driven processes [6], [7], [8], [9].

Overall, the best fit for this project is the Raspberry Pi line of products for their extensive documentation, ecosystem, and variety of customization options in addition to their reliable and well-balanced processing power. Raspberry Pi is a great fit for this project due to its significant range of RAM options offered and is one of the only lines that comes with onboard flash memory, instead of just a MicroSD slot. Raspberry Pi is also the least expensive product line with similar performance in comparison to the other two options.

Table 3.2: SBC Product Line Comparison

	Raspberry Pi	Radxa (Rock Pi)	BeagleBone
Processor	Quad-core	Quad-core or Eight-core	Single-core, Dual-core or Quad-core
RAM	0 (lite), 512MB, 1 GB, 2, 4, 8, 16 GB	256MB, 512MB 1 GB, 2GB, 4GB, 8GB, 16GB, 32GB (range varies from model to model)	512MB, 4GB
Onboard (Flash) Storage	0 (lite), 16, 32, 64 GB	N/A	0, 4 GB, 16 GB
External Storage	MicroSD	MicroSD	MicroSD
Ethernet	Yes*	Yes*	Yes*
Wireless Connectivity	Yes* (optional)	Yes* (optional)	Yes* (optional)
USB	USB 2.0 & 3.0	USB 2.0 & 3.0	USB 2.0
HDMI Connection	HDMI or micro HDMI	micro HDMI	Micro HDMI
Communication Protocol	UART, SPI, I ² C	UART, SPI, I ² C	UART, SPI, I ² C

Input Voltage	5V	5V	5V
Input Current	0.7-1A, 3A	1A, 3A	1A, 3A
Size		86 mm x 56 mm	
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

* = only for some models

3.1.2) Raspberry Pi SBC Series and Part Selection

Following the conclusion that the Raspberry Pi line of SBCs is the best suited for our project, there are several models within this family to choose from. Raspberry Pi offers 3 main SBC series that are suitable for this project, the Flagship series, the Zero series, and the Compute Module series.

The Flagship series is the staple of the Raspberry Pi family and is used primarily for general-purpose computing. The Flagship series offers a balance of processing power, pre-soldered I/O ports, and a wide community of support. Its fixed layout of HDMI, USB, and other connectors make the design more rigid in configuration but allow for a much easier, turnkey approach compared to the more specialized Zero and CM series. The Compute Module (CM) series is more often used for embedded devices, custom I/O integration, and edge computing for AI inference. It is arguably the most customizable of the Raspberry Pi lines, allowing greater versatility in configuration due to the fact that it does not include pre-soldered I/O connectors. In lieu of these connectors, a custom carrier board is required to expose the interfaces and to allow communication with other peripherals and components. The Zero series is oriented towards conservation of space and power, sacrificing some levels of performance and functionality. The Zero series is ideal for minimal-cost projects and wearable technology with emphasis on small-scale size [10].

For this project we require a decent amount of RAM. Approximately 4 to 8 GB of RAM is a reasonable amount to run Linux or another operating system smoothly while also supporting the audio stack, and managing real-time communication between peripherals. Unfortunately, the Zero Series has a limited amount of RAM available, topping out at 512MB, ruling it out in this instance. Another specification to consider is form factor. The CM series tends to run smaller at either 55mm by 40mm or 67.6mm by 31mm when compared to the Flagship series which runs at 85mm by 56mm. While this differentiation is not substantial, it is significant enough to warrant

attention. The smaller the board is, the smaller the overall device can be, reducing potential bulk that could take away from the portability/wearability of the device. An alternative benefit to this conservation of space is a less crowded interior for the device, allowing adequate space for heat sinks and proper cooling elements. Lastly, comes the cost comparison. The CM series makes up the low and the high of the prices listed in the table below. As previously stated, Raspberry Pi offers several different variations for each of their boards. These variations account for the larger price range of the CM series, considering that not only are multiple RAM options available but so are 5 different onboard flash storage configurations in addition to the microSD slot to extend storage even further. Considering the specifications listed and the discussion above we select the Raspberry Pi Compute Module series [10].

Table 3.3: Raspberry Pi Series Comparison

	Flagship Series	Zero Series	Compute Module Series
(usable) GPIO Pins	28 pins	28 pins (unpopulated: meaning that the connections need to be soldered on	28 pins (via carrier board)
RAM	1GB, 2GB, 4GB, 8GB, 16GB	512MB	1GB, 2GB, 4GB, 8GB
Onboard (Flash) Storage	N/A	N/A	0GB (lite model), 8GB, 16GB, 32GB, 64GB
External Storage	microSD	microSD	microSD (via carrier board)
HDMI	2 micro HDMI	1 mini HDMI	1-2 HDMI (mini, micro or standard)
USB	2 USB 2.0, 2 USB 3.0	2 micro USB	1 USB 2.0 (and 2 USB 3.0 in CM5)
WiFi Functionality option	yes	yes	yes
Form Factor (mm)	85 x 56	65 x 30	55 x 40 or 67.6 x 31
Cost	\$35-\$120	\$15-\$18	\$25-\$135

Raspberry Pi also developed the Keyboard series of SBCs which integrates the technology of the corresponding Flagship series device into a compact keyboard case. We felt it safe to omit this series from comparison due to the functional parity of the Flagship series and the keyboard series, barring the case. The two models available in the Keyboard series have very limited customization in comparison with the Flagship, Zero, and Compute Module series and start at a much higher price point of \$70, with higher-end models priced at \$120. From a design standpoint though, neither the Raspberry Pi Keyboard 400 nor 500 models could be safely integrated into our schematic as the casing is much too large to reside within our build (286mm x 122mm x 23mm for both). Additionally, due to the proposed location of the SBC in our build the keyboard would be inaccessible, rendering the only distinguishing characteristic of these two models moot [11].

Part Selection:

Within the realm of the Raspberry Pi compute module series there is yet another set of options to choose from. The main three models we consider are the Pi Compute Module (CM) 4, 4S, and 5. At this level we see that CM4S has a longer but more slender shape at 67.6mm x 31mm in comparison to the CM4 and CM5 which are both 55mm x 40mm. Additionally, the CM4S only supports 1 HDMI port compared to the 2 offered by the CM4 and CM5 [12, p. 4], [13], [14, p. 5]. One of the more important specifications in this project is price point. These models each start at a base rate for the lite model with the minimum available RAM and no wireless option (if applicable); as the RAM and memory are increased and wireless capability is included, the price trends towards the ceiling of the price point. The Pi CM4S is the least expensive option overall, starting at \$25, however it does not include the option of wireless connectivity. The low end of the Pi CM4 is only \$5 more expensive than the CM4S at a base rate of \$30, and allows for a second HDMI connection in addition to offering the option of wireless connectivity. On the other end of the spectrum, the base model of the CM5 sees an even larger increase in price, with a base rate of \$45. One thing to note though is that the RAM and flash memory options for the CM5 start at 2GB and 16GB respectively instead of 1GB and 8GB options which accounts for a substantial portion of the price increase [2, p. 5], [15, p. 4], [16].

Another factor we must consider is availability. Unfortunately, because these boards are so popular, many vendors are sold out of several different models and configurations. While some models are available for resale, they are sold at a much higher rate due to lack of availability. An alternative would be to purchase a pre-used board, however, we do not feel comfortable about the integrity of the board itself after prolonged use and the ambiguity of the board's state/quality. Given that the CM5 released much more recently than the CM4S or CM4 models there is limited availability for these boards on the open market matching the configuration options we require.

With these specifications and limitations it can be concluded that the best SBC to use for this project is the Raspberry Pi Compute Module 4.

Table 3.4: Raspberry Pi Compute Module Models

	Pi Compute Module 4S (CM4S)	Pi Compute Module 4 (CM4)	Pi 5 Compute Module (CM5)
RAM (GB)	1, 2, 4, 8	1, 2, 4, 8	2, 4, 8, 16GB
Flash Memory (GB)	0 (lite), 8, 16, 32	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-A72 (ARM v8) 64-bit SoC @ 1.5GHz	quad core Cortex-A76 (ARMv8) 64-bit SoC @ 2.4GHz
Form Factor (mm)	67.6 x 31	55 x 40 x 4.7	55 x 40 x 4.7
USB	1 USB 2.0	1 USB 2.0	1 USB 2.0 2 USB 3.0
HDMI	1 HDMI 2.0	2 HDMI 2.0	2 HDMI 2.0
GPIO	≤ 46 pins	≤ 28 pins	≤ 30 pins
UART support	≤ 6 UART	≤ 5 UART	≤ 5 UART
I ² C Support	≤ 6 I ² C	≤ 5 I ² C	≤ 5 I ² C
SPI Support	≤ 6 SPI	≤ 5 SPI	≤ 5 SPI
Wireless Connectivity	N/A	Yes	Yes
Cost	\$25-\$75	\$30-\$85	\$45-\$135

3.2) Microcontroller Unit (MCU)

The KIPS device will be utilizing a microcontroller unit (MCU) to work with the SBC. The purpose of the MCU will be to handle analog data from the sensors and to allow the user to control the device's onboard AM/FM radio.

In this section we discuss alternative options for the MCU. This will be in the form of comparing a few different series among several different brands. Then, the reasoning behind why we choose a specific MCU from each series will be covered. The advantages, disadvantages, and technical details of each will be inspected in terms of their relevance to our physical and software goals for the KIPS device.

3.2.1) MCU Manufacturers and Series

The market for MCUs is extensive and vast, with MCUs being available for just about any purpose that can be thought of. With this in mind, several lines of MCUs were considered, each from a different manufacturer. In our search for a potential MCU, we had several major qualifications the MCU must meet in order to be considered as an option for KIPS Mk II. It needed to be physically small, draw a conservative amount of power, and have enough computing power to process sensor and audio data. With this in mind, the following manufacturers and series were closely matched our required specifications: the ESP32 series by Espressif Systems, Raspberry Pi series by Raspberry Pi Holdings, the nRF53 Series SoC by Nordic Semiconductors, and the STM32 series by STMicroelectronics.

To begin, Espressif System's ESP32 series is a well known, and well received line of MCUs, and is considered among one of the most popular MCUs on the market for many types of applications. For the purposes of its usefulness in the KIPS device, its CPU power, and energy efficiency were two of the major factors in its potential to be used. It has been well known from hobbyist to professional levels, the ESP32 series is arguably one of the most cost effective, feature-rich, and powerful MCUs on the market. Most products in the series feature dual-core processors, 520KB of 240 MHz RAM, and are designed with power conservation in mind. The ESP32's balance between very low power usage and robust computing capabilities make it a very strong contender to use out the gate [17] [18].

Raspberry Pi Holding's Raspberry Pi Pico series is another well known and universally acclaimed MCU line. Unlike the ESP32, using this MCU would have a further advantage of being the same brand as the SBC, which is a Pi CM4. This would make working with both systems, particularly for communication, easier. In terms of the demands of KIPS, the Pico series also features dual core processors and 240KB of 133MHz SRAM, which is not as fast as the ESP32, but still formidable. Additionally, they are made with low power consumption in mind, feature a programmable I/O (useful for offloading demanding tasks to the SBC and customization), and have extensive documentation. The Pico series and the Raspberry Pi series in general are known for their ease of use from the hobbyist level, to industry. All of these aspects make the Pico series a very appealing option to use with KIPS [19] [20].

Nordic Semiconductor's nRF53 series may not be as well known as the Pico or the ESP32, but the MCUs used in this line were still considered for KIPS. Like previous choices, it features a dual-core processor, and it has about 512MB of 128MHz RAM. The cores feature a more unique architecture, with one core managing application processing, and the other core managing wireless communications. This makes for an intriguing option for use as an MCU, as there is potential to better streamline the processor's efficiency to complete tasks. This can also feed into power consumption, where application processing could only be done as needed, while the

communications could be handled at all times instead of both running simultaneously. It is also made for low power consumption, and has a higher operating temperature of about 105 C. Although the nRF53 series is not as well known as the Pico or ESP32, the MCUs series' architecture makes for a unique approach to utilize in KIPS [21] [22] [23].

STMicroelectronic's STM32U5 series is the last series considered for KIPS. This particular series features more variety compared to previously discussed series, as each product contains different cores, and different products within the series are tailored to specific purposes. This makes for a better selection when deciding which specific product could be used for KIPS. Additionally, many products in this series feature very low power consumption, security features, and up to 3MB of SRAM. Interestingly, the chips in this series run on a single core, and some of the more advanced chips even feature built in graphics. The more advanced chips also have notably more SRAM, but all run at 160MHz. This series' extreme low power consumption and computing capabilities despite only having 1 processor make it an appealing option to use for KIPS. Given KIPS' mobile computing nature, the extremely low power draw is an especially suitable feature to better maximize the battery charge [24] [25].

Of the ESP32 series, we ultimately considered the ESP32-S3. The S3 features dual cores, a base clock speed of 240 MHz, 512KB of SRAM, built in Wi-Fi, 36 programmable GPIO pins, and about 150-250 mA of power consumption. The S3 has a very strong feature set, and powerful processing capabilities. Additionally, it has a large community behind it and has been extensively documented in a variety of scenarios due to its popularity. Because of this, it also makes hardware and software repairs much more accessible and/or easier to find a solution, as it is likely others have experienced similar issues in the past. Another contention point with the S3 is its price, as it is competitively priced at around \$6. At less than \$10 for the MCU, it makes the S3 chips extremely inexpensive and easy to replace if needed. The MCU has very strong computing capabilities overall while consistently maintaining a low power consumption, making it an overall more suited MCU for the device [26].

The next possible choice among the Raspberry Pi series was the Raspberry Pico W. The Pico W's main features include cores, a clock speed of up to 133 MHz, 264KB of SRAM, built in Wi-Fi, and 26 multipurpose programmable GPIO pins, and about 50-150mA of power consumption. Since the chosen SBC is already a Raspberry Pi CM4, this option has the additional benefit of working well with the CM4 since they would both be Raspberry Pis. This would make programming and searching information easier, as Raspberry Pi is known for having excellent documentation, and the set up between the SBC and MCU would be similar, as they would be from the same brand. The overall specs of the Pico W are not as powerful while also being physically larger. However, the compatibility with the SBC, and overall simplicity to configure make this MCU a strong contender. The price point is another advantage, as the Raspberry Pico W was made to be less costly and easier to replace. With the main concern being

the overhead needed in the case of the KIPS device, there would be less overhead compared to the ESP32-S3. Overall, its major strengths come from ease of use and compatibility with the system, at the cost of less overhead [27].

For Nordic Semiconductor's NRF53 series, the chip that was most compatible for the KIPS device was found to be the nRF5340. The main features of the MCU includes specialized dual cores (one is for application processing, the other is for wireless communications), a max clock speed of about 128MHz, 512KB of RAM, 48 GPIO pins, and about $\sim 0.9\mu\text{A}$ - 5.1mA of power. However, there is no Wi-Fi module. The strong appeal of the MCU is because of its very small size, low power consumption, and specialized cores. Being the physically smallest of the MCUs, it would help save space, allowing for a larger variety of potential other components to be potentially used in the space that is freed up. Additionally, it uses notably much less power compared to other MCU options, which would help extend the battery life of the device by a noticeable amount. The specialization of the cores is another interesting boon, as being prebuilt with the idea of focusing on application running and wireless communications would help ease the configuration aspect when programming the MCU. For example, coding for the applications within KIPS could be entirely finished and focus on running in the appropriate processor, and once this is finished, the process is repeated for the wireless communication processor. However, the lack of Wi-Fi would require a Wi-Fi module, which may defeat the purpose of the extra space freed up by the smaller physical size. Additionally, for these extra features, there is an amount of computing power that is exchanged, as compared to the ESP32-S3, there is at minimum about 120MHz slower speed in the processing power, which is roughly half the processing speed. In other words, in exchange for a smaller physical footprint and less power consumption, it does come at the cost of slower processing power and a lack of built in Wi-Fi [28].

In STMicroelectronic's STM32 series, the STM32U585AI MCU was found to be the most compatible option for the KIPS device. It features a single core, 160MHz clock speed with 784KB of SRAM, up to 136 GPIO pins, and a power consumption of around $0.16\mu\text{A}$ - 3.1mA . Additionally, it is very small, at around $7\times 7\text{mm}$. This MCU was selected because of its low power consumption as it is the lowest of all the selected MCUs, while not sacrificing as much processing speed, and having the largest amount of SRAM of the MCUs and the most GPIO pins. This makes the MCU a competitive match, as it has a considerable amount of computing power while consuming less power and physical space compared to other options. However, these features come at a cost however, as unlike all other options, it contains a single core, and like the nRF5340, it lacks a Wi-Fi module. Even with its large RAM and higher clock speed, all processes would need to run on a single core, and within the KIPS device there will be several inputs at once depending on circumstances. At 160MHz, the core's ability to process all inputs within a reasonable time is uncertain. Additionally, like the nRF5340, the space freed by its physically small size may be negated by the need for a separate Wi-Fi module. Overall, the increased RAM, processing speed, GPIO pins, and extreme low power usage are all appealing

features that come at the cost of there only being one core to run all processes, and a lack of an integrated Wi-Fi module [24] [29].

Table 3.5: MCU Product Line Comparison

	ESP32-S3	Raspberry Pico W (RP2040)	Nordic 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 Consumption (min-max)	~2.5 μ A - 260mA	~50mA - 150mA	~0.9 μ A - 5.1mA	~0.16 μ A - 3.1mA
RAM (KB)	512 (SRAM)	264 (SRAM)	512	784 (SRAM)
Size (mm)	18 x 25	51 x 21	4.4 x 4 (WLCSP95)	7x7 (UFBGA169)
Flash (MB)	8 (max)	2	1 + external QSPI	2 (max)
USB Support	Native USB-OTG	Native USB	USB	USB FS + OTG FS
GPIO Pins	45	40 (max)	48 (max)	136 (max)
Comm. Protocol	UART, SPI, I ² C, I2S, CAN, PWM	UART, SPI, I ² C, PWM, PIO	UART, SPI, I ² C, PWM	UART, SPI, I ² C, FDCAN
Wi-Fi	Yes (4)	Yes (4)	No	No
Price ((\$), MCU only)	2.50-3.50	4	5-7	9-10

Among all the MCUs that were selected as possible choices for the KIPS device, ultimately it was decided that the ESP32-S3 would be sufficient to work with the Pi CM4 SBC. Additionally, there will be enough overhead in the S3's computing capabilities to account for any extra processing power that may be needed. The S3's dual cores and clock speed make it a power efficient option, capable of performing its tasks without consuming a problematic amount of

power. As with any MCU, the max power draw will likely never be reached, and it will operate at a more reasonable current draw. Additionally, any features that are not going to be used can be shut down upon start up. Unlike most of the other options, the availability of resources to find help for programming or placing an ESP32 in a system make it easy to find assistance from the internet before asking for help from another person. 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. While it may not be as straightforward to work with as a Raspberry Pi, the aforementioned pros make up for this difficulty.

Although we had initially chosen the ESP-32-S3, we decided to use an ESP-32-WROOM-S3 simply due to the fact it had some extra features, as well as the fact that it offered a drop in ready to go package containing all supporting circuitry needed for the ESP-32 S3 chip to properly function.

3.3) Sensors and Sensor Technologies

In order to deliver certain types of data or to fully implement a feature on the device, KIPS will have several integrated sensors within and attached outside the device to fulfill these purposes. The three main sensors will be a heartbeat sensor, a temperature sensor, and a humidity sensor. These will be used in corresponding applications, and they will only be active when the user chooses to view the data these sensors monitor. The heartbeat sensor will be active whenever the user chooses to enter the heartbeat tracker application to measure their heart rate. The temperature and humidity sensors will measure their respective data in the area surrounding the user, in real time.

3.3.1.1) Heartbeat Detection Technologies

There are several different types of common technologies found in heartbeat sensors that are used to measure heartbeat. These methods are photoplethysmography (PPG, also called optical), electrocardiography (ECG), and ballistocardiography (BCG). Each of these methods has its own pros and cons for accuracy and technologically. In the context of mobile technologies, PPG and ECG tend to be more favored.

The first overview will be of the PPG technology. All sensors that were considered for the KIPS device use PPG, and it is most commonly used in fitness trackers and smartwatches. The appeal to this set up is the simplicity and cheapness, as it only consists of LEDs, photodiodes, and an MCU. It essentially works by using the photodiodes to measure changes in light absorption when blood volume changes with each heartbeat. Additionally, this method has the added capability of measuring blood oxygen, if the measurement capability is included in a given sensor. However, the drawbacks include needing adequate skin contact for accuracy and power draw can vary based on how the LEDs are set up [Appendix C, 1b].

The second commonly used technology is the ECG. The ECG method is a favored one due to its accuracy, ability to detect abnormal heartbeats, and very low power consumption. The way it works is by placing electrodes on the skin to measure electrical signals created by the heart's muscle contractions, which gives a direct way to track the heart visually via its electrical activity. The downsides of this method however include the need for constant skin contact, and if it is placed anywhere that is not the chest area, muscle contractions can create noise and decrease the accuracy. Additionally, this contact is more direct, in the form of electrodes, pads, or adhesives. This makes ECG a less wearable-friendly method, but it is not impossible to implement [Appendix C, 1b].

The third heartbeat technology is the BCG. This is the less commonly used of the three technologies seen here, but it is still used in wearables and medical devices. It works by detecting vibrations or accelerations caused by blood ejections during each heartbeat. The BCG method is more wearable-friendly, since it does not need direct skin contact, and it is non-invasive. However, it is not as accurate as PPG and ECG, and although it is more wearable-friendly, it is meant for more niche research applications, not mainstream general wearables devices [Appendix C, 1b].

Table 3.6: Heartbeat Detection Technologies

	PPG	ECG	BCG
Power Draw	Varies (based on LEDs)	Low	Moderate
Skin Contact	Constant	Constant	Unnecessary
Accuracy	Moderate inaccuracy	Most accurate	Least accurate
Cost	Moderate	Expensive	Inexpensive

To conclude, the technology that was ultimately chosen was the PPG sensing technology. For the purposes of the KIPS device, it is simple, inexpensive, and easy to use. Many technologies that have a heartbeat detection feature use PPG technology, it is typically the default choice. The power draw can be controlled to an extent by manipulating the LEDs, and while accuracy is important, KIPS is not a medical device, therefore absolute accuracy and precision for heartbeat is not a requirement, and a certain level of inaccuracy is tolerable. Unlike ECG, the skin contact is not required to be quite as obstructive, since ECG requires a more medically inclined approach whereas PPG only requires the device to essentially be pressed against the skin. Overall, PPG resulted in being the best middle ground between cost, power draw, comfort for the user, and accuracy.

3.3.1.2) Heartbeat Sensor

The idea behind using the heartbeat sensor will be that the user can tap the heartbeat sensor application, and request the KIPS device to read their heartbeat. When this request is made, the sensor will be activated, the heartbeat measured, and the information will be sent to the MCU for processing, then the SBC will display the result to the user. In other words, the sensor will only be activated on command by the user. This way, it will save on power as it will not be running constantly, and it eliminates the need to constantly display the user's heartbeat, which may not be wanted or needed.

For this sensor, three potential sensors were selected based on several different characteristics. These characteristics were input voltage, operating current, the type of output it would send to the MCU, the physical size, software support, and the cost. With these in mind, the three most compatible sensors were determined and compared to each other: Analog Device's MAX30102, Word Famous Electronic's Original PulseSensor Kit, and DFRobot's Heart Rate Monitor (SEN0203).

The MAX30102 is a very useful device that has an extremely low overall voltage and current draw on the system, as well as being extremely small at 5.6mm x 3.3mm x 1.55mm, and it is low cost. For the purposes of the KIPS device, these characteristics are all favorable. The only exception being potentially the input voltage, which is not between 3.3V - 5.5V, and its ease of use is not the simplest. It would require a more traditional approach compared to the plug-and-play aspect of other sensors. However, once it is set up, it can not only monitor heart rate, but can additionally be made to monitor blood oxygen levels (SpO₂) [30].

The next sensor is not a sensor by itself, but rather found within a kit, called the Original PulseSensor Kit. This kit comes with a braided 610mm cable, a velcro finger strap, a heartbeat sensor, among several other items, some of which will not be used. The appeal of this device is its simplicity of use, as it is compatible with ESP32. This makes it essentially a plug-and-play type of set up, which would help significantly with the set up and integration of the sensor. Additionally, it runs within the desired input voltage range of 3.3V - 5.5V. Its size, at around 15.8mm, puts it much larger compared to the MAX30102, but still reasonable dimensions. The major drawback with this kit is the cost, and the inability to fully utilize the kit, as there are certain items that are inapplicable to KIPS [31] [32].

The final sensor that was selected is the Gravity Heart Rate Monitor, which uses the SON1303 processor with the SEN0203 sensor. This sensor also runs within the preferred input voltage, and uses a low amount of current draw, at less than 10mA. The physical size, however, is the largest at 28mm x 24mm. Like the sensor of the Original PulseSensor Kit, it can only take heart rate, however it is in the form of an ECG waveform, which will need to be manipulated in order to extract proper heartbeat data. Although the setup may not be as simple as the kit, it is compatible

with ESP32, making it a well rounded choice for KIPS. With the exception of the size which is on the larger end, it can perform all tasks required of it for the KIPS device [33] [34] [35].

Table 3.7: Heartbeat Sensors

	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	0.7 μ A - 600 μ A	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
Communication Protocol	I ² C	None, outputs a changing voltage proportional to the PPG signal	None, outputs a changing voltage proportional to the PPG signal
Cost (\$)	5.41	24.99	16.00

Ultimately, the sensor that was determined to work best with the KIPS device was the SEN0203 sensor in the Gravity PPG Heart Rate Monitor by DFRobot. Although the output would be in an ECG waveform, the data can be easily manipulated to be displayed in an easier to understand format for the user. Additionally, it is within spec for all the electrical requirements of the device. Unlike the PulseSensor, there will be no unused parts since it is not part of a kit. The MAX30102 was a strong contender, but ultimately could not be chosen as its voltage operating range was below the specifications needed for KIPS to operate properly. The SEN0203 can be placed against the wrist or the bottom of the forearm on the bottom of the device for the most skin contact, and wired to the MCU within the main body. Overall, the SEN0203 proved to be the most convenient to use without costing an unreasonable amount.

3.3.2.1) Light Detection

We decided that KIPS MkII should have some sort of light detection. This would enable some features, like an automatic screen brightness setting, or perhaps an automatic flashlight. To that

end, we looked at using photoresistors, photodiodes, and phototransistors for this task.

Photoresistors are passive light sensors; this means they require no active power input in order to function. They tend to be on the more inexpensive side, with parts ranging from cost in pennies to being close to a dollar. They tend to operate within the visible light spectrum [36].

Photodiodes are an interesting option due to them being able to function as both an active and passive sensor. This is because of their ability to function in a photovoltaic and photoconductive mode, those modes being passive and active respectively. When a photodiode is unbiased, it operates photovoltaically; meaning it draws no power but outputs small amounts of voltage when it gets illuminated by a light source [36]. When a photodiode is reverse biased, it operates Photoconductively; meaning it tends to have a faster response time, with the caveat of being more prone to noise [36]. Photodiodes also typically work within the visible light spectrum, and are low in cost [36].

Phototransistors are strictly active sensing components. They function similarly to photodiodes, but we must remember that they are BJTs; which means we can consider them to be like photodiodes with built in current amplification [36].

Table 3.8: Light Detection Methods

	Photoresistor	Photodiode	Phototransistor
Sensor Type	Passive	Passive/Active	Active
Power Requirement	None	Photovoltaic: None Photoconductive: Bias Supply	Bias Supply
Cost	Low	Low	Low
Spectral Response	400nm - 700nm (Visible Light)	350nm - 1100nm (Some IR + Visible Light)	350nm - 1100nm (Some IR + Visible Light)

We have decided to keep things simple and go with a photoresistor, as they are passive sensors that are cheap and effective for our uses.

3.3.2.2) Photoresistor

When looking at photoresistors, we found that there isn't much to discern them other than their resistance range, their rated power, temperature range, and their "seeable" range of lux; e.g.

where the sensor tops and bottoms out. To this end, we sampled products from Excelitas Technologies, and Senba Sensing Technology; those being the VT935G, GL5537LDR and GL5528LDR respectively.

Table 3.9: Photoresistor Models

	VT935G	GL5537 LDR	GL5528 LDR
Resistance Range	$200\Omega < x < 1M\Omega$	$1k\Omega < x < 2M\Omega$	$500\Omega < x < 1M\Omega$
Lux Range	$1 < x < 10k$	$1 < x < 10k$	$10 < x < 100k$
Operating Temp.	$-50^{\circ}C$ to $+75^{\circ}C$	$-30^{\circ}C$ to $+70^{\circ}C$	$-30^{\circ}C$ to $+70^{\circ}C$
Power Rating	80mW	100mW	100mW

Given the above parameters, we decided to go with the GL5528 due to its increased lux range and resistance range, allowing us more sensitivity when writing a program to interpret the output.

3.3.3.1) Temperature Detection Technologies

There are three main types of technologies used to detect and measure changes in temperature in mobile/wearable technologies. The three types are the negative temperature coefficient (NTC) thermistor, the silicon bandgap, and the resistance temperature detector (RTD). In the context of KIPS, the readings of the sensor do not need to be precision accurate, which is acceptable because body heat and heat from other parts of the device itself. Additionally, KIPS is not a weather device, so a certain amount of inaccuracy is tolerable.

The first technology is the NTC thermistor, which uses a resistor whose resistance will drop as temperature rises. The appeal of this technology is that it is small, inexpensive, and quick to record data. It can work for a variety of voltage supplies since it uses a resistor divider and an ADC. Additionally, it can work on very low power, which is ideal for mobile devices. However, its accuracy is questionable, as it is liable to heat itself if it is sampling using a higher current. It also reads surface temperatures, meaning parts of the mobile device, and body heat can influence the sensor's readings [37].

The second technology, and the most popular one of the technologies overviewed here, is the silicon bandgap. This technology works by asking the chip in the sensor for its temperature register value, then applying a formula to the value to convert it to degrees celsius. This means no analogue aspects to temperature reading. The convenience of this technology is that it is calibrated out of the box, which saves time when programming. It is also very low power, and comes in small packages, which for KIPS would make set up and physical implementation

easier. However, the disadvantage is that it measures its own die temperature. Therefore, the sensor must be thermally isolated from other heat sources if ambient temperature is desired. This can be a challenge on a mobile wearable device, given it is close to the body and space within the device itself is limited [37].

The third technology is the RTD. This technology uses a specialized resistor whose resistance will increase as the temperature rises. This resistor often utilizes a metal such as platinum which has a linear resistance, for stability purposes. RTD is excellent for its accuracy and stability because of these design choices, and it makes calibration easier for the programmer. However, to achieve these benefits, it requires a current source and ADC with precision. Additionally, it can be physically bulkier and more demanding in power [37].

Table 3.10: Temperature Detection Technologies

	NTC Thermistor	Silicon Bandgap	RTD
Power Draw	Very low power when duty-cycled.	Very low power, designed for battery devices.	Higher power draw
Physical Size	Very small	Very small	Bulkier
Accuracy	Lower absolute accuracy	Calibrated; accurate die temp. Accurate ambient if thermally isolated and vented	High accuracy + stability over time
Cost	Low	Low	Moderate-High

The technology chosen to work with KIPS was ultimately the silicon bandgap. It is the most commonly used technology in mobile devices, and was specifically created with battery devices in mind. This alone makes many of the characteristics of silicon bandgap technology very appealing to use for KIPS. Additionally, its low cost is also appealing, making replacement much more convenient. Overall, silicon bandgap technology is a compromise of quality, cost, power raw, and physical size.

3.3.3.2) Humidity Detection Technologies

The two most common technologies used for humidity detection in mobile wearable tech are capacitive polymer relative humidity (RH) and the resistive conductive polymer/salt. Similar to the temperature, the humidity readings of the humidity sensor do not need precision accuracy, so

a certain amount of inaccuracy is tolerable to account for exterior factors that can affect the sensor's readings.

The first technology, the capacitive polymer RH, is a hygroscopic polymer layer that changes its dielectric constant as RH changes. The sensor reads the capacitance of the hygroscopic polymer and converts this value into an %RH reading. This reading is what is displayed to the user. This technology is the most popular in wearable technology, as it is physically tiny, extremely low power (in the μA range), has a 1%-3% accuracy, and has a fast response time. Some of these sensors even include a micro heater to clear condensation. The drawbacks of this technology however is that its accuracy can drift with time and exposure to elements. Ideally, it is best used with a filter cap to protect the sensor from exposure [37].

The second technology is the resistive conductive polymer/salt. It works by having a special film in the sensor that can absorb water vapor. As the water content in the film rises, ions and charge carriers can move more easily, so the electrical resistance drops, the higher the RH. This change in resistance can be mapped to the change in humidity, which is ultimately displayed to the user. It has the benefits of being very low power, producing analog output, and being inexpensive to create. However it can be affected by hysteresis and drift, in addition to being slower and less stable compared to the capacitive polymer. It also prefers an AC current, as DC will cause polarization and electrolysis over long periods of use [37].

Table 3.11: Humidity Detection Technologies

	Capacitive Polymer Relative Humidity	Resistive Conductive Polymer/Salt
Power Draw	Extremely low (μA range)	Higher than capacitive polymer RH
Drift	Drifts with time/exposure	Hysteresis + drift, needs more protections
Accuracy	Decent, generally reliable for wearables	Less accurate, has more variation and drift
Cost	Low	Low

Of the two technologies, the more compatible technology for KIPS is the capacitive polymer RH. It has extremely low power use, and it is more accurate compared to the resistive conductive polymer/salt. Though it does drift with time and exposure, the effects of this are mitigated more since it is more accurate to begin with. Additionally, it does not require as much protection from this compared to the precautions that must be taken with the resistive conductive polymer/salt. These benefits are all for the same low cost as well. Therefore, the capacitive polymer RH would make for the more beneficial choice to use in KIPS.

3.3.3.3) Temperature and Humidity Sensor

The temperature and humidity sensor will be placed on the exterior of the device, as one component which measures the two weather conditions simultaneously. A protective cap will be needed to protect the device from the elements, hence the cap is included with the cost of the sensor itself.

With these specifications in mind, the first sensor considered for the KIPS device is the AM2301B by ASAIR. The sensor comes with both the sensor itself and the protective casing, meaning no separate costs for a separate cap. Additionally, it runs well within the voltage and current supply requirements, therefore no conflicts with powering the sensor. However, unlike the other sensors, it must be noted that due to the convenience of being all in one package the drawback is that it is significantly larger than the other sensors. This added bulk may cause difficulties in being able to place it in a more optimal position on the device to take measurements. It uses the thermistor method as well, which helps reduce price, takes low power to operate, and fast to record data, however it can be more liable to inaccuracies because of this [38] [39].

The next sensor is the SHT30A-DIS-B and the SF2 filter cap combination by Sensirion. The sensor and protective cap are sold separately for this sensor, so their total cost is summed together in the final chart below, while still showing the individual prices for each part. Unlike the AM2301B, it is much smaller. While it does consume more power than the AM2301B, this is not a significant increase. Therefore, this drawback is tolerable. Additionally, of the three sensors considered, even with the cap, the overall cost of the component is the cheapest. Overall, the appeal of the SHT30A-DIS-B is the much smaller size when compared to the AM2301B and its inexpensiveness. [40] [41] [42] [43].

Next is the HTU21D(F) by Measurement Specialties. This sensor is the smallest of the three sensors, which makes it the easiest option to physically place on KIPs. Unlike the SHT30A-DIS-B and AM2301B, it comes with a PTFE protective membrane, making it the smallest even with a protective cover on it. It also uses the second least amount of overall power of the three, which is an appealing option for power conservation. Additionally, it is the cheapest of the three selected sensors. However, the biggest caveats with the device are the input voltage, which runs lower than the preferred 3.3V for a typical input voltage, and the shipment time, which would take roughly a month in order to receive a new order [44] [45] [46].

The final sensor is the AM2320 by AOSONG Electronics. This sensor falls within the voltage range required to be used within the KIPS device with no issues. Although it is the most power hungry among the chosen sensors, the power draw is still within a tolerable range. Like the HTU21D(F) and the AM2301B, it comes with its own protective cap, which eliminates the cost

of having to purchase a separate protective cap. Particularly, the physical layout of the sensor is most compatible with the rest of the KIPS device layout. It utilizes pins, therefore making it easier to work with when connecting it to the rest of the device. Additionally, it is the second largest of the sensors selected, however when compared to the AM2301B, it is still significantly smaller [47] [48].

Table 3.12: Temperature and Humidity Sensor Comparison

	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
Typical Input Voltage	3.3	3.3	3.0	5.0
Operating Current	250 nA	600 μ A - 1500 μ A	300 μ A - 500 μ A	950 μ A
Output Data Type	$^{\circ}$ C, %RH	$^{\circ}$ C, %RH	$^{\circ}$ C, %RH	$^{\circ}$ 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
Measurement Method (temperature)	NTC thermistor	Silicon bandgap	Silicon bandgap	NTC thermistor
Temperature Accuracy ($^{\circ}$ C)	± 0.4	± 0.3	± 0.3	± 0.5
Measurement Method (humidity)	Capacitive Polymer RH	Capacitive Polymer RH	Capacitive Polymer RH	Capacitive Polymer RH
Humidity Accuracy (% RH)	± 2	± 3	± 2	± 3
Communication Protocol	I ² C	I ² C	I ² C	I ² C
Cost (\$)	7.95	3.21 + 1.26	3.89	3.95

The final selection for a temperature and humidity sensor was the AM2320. Compared to the other sensors, it is the second cheapest among them, and falls within the input voltage range that is required to run within KIPS. Since accuracy is not a major concern as KIPS is not a weather device, the level of inaccuracy seen for the AM2320 is tolerable despite having the least amount of accuracy of the sensors. Additionally, its physical layout, which uses pins, makes working with the AM2320 easier compared to the other sensors. Overall, despite the power draw being higher and it not being the smallest among the sensors, these aspects are still within tolerable limits. Additionally, the ease of physically integrating the sensor into the layout of the KIPS device is greatly valued, which is also why the AM2320 was selected to be used.

3.4) Peripherals

3.4.1.1) Physical Interface

In order to natively interact with K.I.P.S. without peripheral devices, different physical interfaces are required. A few different technologies we explored to achieve this functionality, including buttons, joysticks, directional pads (d-pads), and rotary encoders. Each of these input methods provide a unique way for the user to navigate menus or control functions.

Buttons

The first option we explored were simple push buttons that register on and off states. Buttons are one of the most basic input components available, producing a binary value which depends on its state. Buttons are simple switches that allow a user to open or close an electrical circuit through pressing or releasing it. When pressed, a connection is made between two endpoints which allows the circuit to complete. Buttons have a few key components, the actuator, contacts, and the spring or membrane. The actuator is the part that is pressed by the user to enact input. The contacts are the conductive elements that touch when the button is pressed to complete the circuit. The contacts can be oriented in one of two ways, normally open (where the default state of the button is open) and normally closed (where the default state is closed). The spring or membrane provides some resistance against the button being pressed and returns the button to its original position when it is no-longer pressed. Buttons are commonly used for selection-based inputs to a device. Buttons can serve multiple functions but traditionally, multiple are included in modern devices for simultaneous operation. Buttons are typically wired and programmed to specific functions such as selecting something or returning the screen to the previous state [49].

One of the advantages of buttons is that they are one of the simplest forms of physical interfaces available, meaning that they would be arguably the easiest to implement. Another benefit of buttons and their simplistic design is that they are extremely low cost. Buttons are also fairly durable, with anywhere from a few thousand to a few million actuations. Lastly, buttons can provide assurance to the user that the input has been registered through the physical movement of the component and sometimes also through a clicking sound. This could prove useful in

instances where input may be delayed for a split second and the user wants to ensure that their input is being accounted for.

One of the disadvantages of buttons is that they can only be used for one function at a time. For classic, four directional input, four different buttons would be required which could take up a significantly large amount of space on the surface of the device. Another disadvantage of buttons is their limited input complexity. Buttons only provide binary input; no information such as pressure or quad-directional input can be tracked which is a limiting factor as some more complex inputs may be required for smooth integration with the device's user interface. Additionally, buttons may suffer from a phenomenon called debouncing errors, where the button is pressed too rapidly and the separate presses are not registered as distinct inputs. This debouncing error can be mitigated with either hardware or software filtering.

Buttons could prove useful in this project for a select function or for other programmed uses in games and other applications.

Directional Pad

A directional pad (d-pad) is a cross-shaped button that sits atop a collection of switches or contact points to detect directional input. Each arm of the cross-shaped pad sits above a contact point or switch so that when a user presses a direction the pad slightly tilts and closes the circuit beneath that directional arm which sends a signal corresponding to that direction. D-pads were originally created as a compact method for compact video game consoles to translate movement of the thumb along the controller into readable input for the game. D-pads typically interface with connected components and devices digitally, meaning that only the state of whether or not the distinct directions of the d-pad are pressed is recorded. Unlike the analog interface of other comparable components, no information is passed about how far the directional inputs are pressed, only their state [50].

D-pads present several different benefits as opposed to other comparable technologies. D-pads have a very compact design with a low-profile which would not detract from the overall footprint of the device. Additionally, d-pads are very low cost and highly durable. D-pads are typically manufactured using mechanical switches for each direction (up, down, left, right) with a simple membrane on top which keeps the cost low. The d-pad can also have center-select functionality, allowing for fifth directional input or a selection element, providing greater flexibility and reducing the need for additional buttons or other hardware.

One thing to note is that due to the fact that the top of the d-pad is a singular, continuous button, unintended input may be registered. For instance, if the user were to press in the upward direction but they make contact with the d-pad in the top-right quadrant they may inadvertently trigger the right directional portion as well or instead. This proves troublesome in instances

where precise input is required, as a misstep like such could alter the directional outcome returned. Another thing to note is that d-pads cannot detect changes in speed, rather they just detect whether a directional input is being selected or not, meaning that input may not be as smooth or precise as desired.

Overall, this component would be a great addition to our project for usage in robust directional input, for navigating simpler menus and especially for control in different applications and games. D-pads provide intuitive four-way input in a compact package which makes them ideal for use in our build where surface space is limited and robust directional input is required.

Joystick

For this project we also looked into different types of joysticks. Joysticks are either an analog or digital stick used for multi-directional input. At its simplest, a joystick is a specialized electrical switch that translates the physical movement of the central shaft/stick into directional information that can be easily relayed to the connected circuit/device. Most standard joysticks today monitor the position of the stick along the x-axis and y-axis using potentiometers, where the potentiometer is essentially a resistor positioned along a curved track and a contact arm which slides along the track. Tilting the stick up, down, left, and right shifts the contact arm along the track, increasing or decreasing the resistance which acts upon the current flowing in the circuit. This electrical signal is then converted to an analog signal through the use of a very simple analog-to-digital converter, which is then able to send numerical data to the connected device in binary format. The device is then able to utilize this digital information to enact movement on the screen or cause other programmed responses such as menu navigation, activation of a certain function or control over external systems [51].

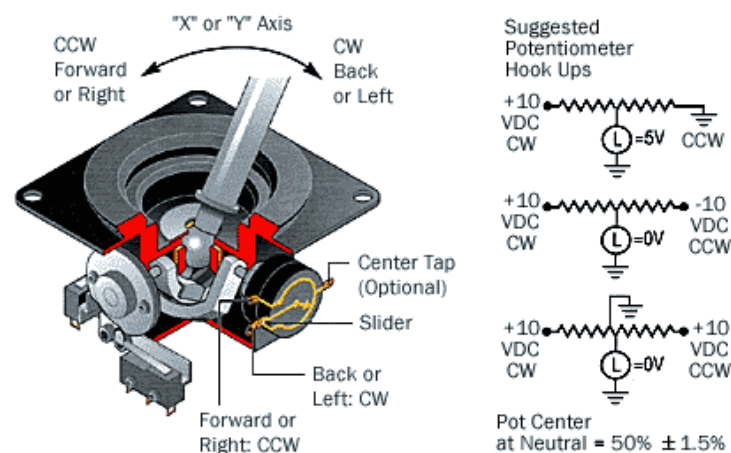


Figure 3.1 Potentiometer Configuration [52]

One of the advantages of using a joystick is that it is fairly intuitive for controls of directional movement. Many users are quite familiar with the use of joysticks in either gaming, robotics, or aviation capacities. Joysticks also produce analog output, compared to digital buttons, meaning they allow for more precise control over device operations, such as desktop or mobile app navigation. Another major advantage of the joystick is multi-dimensional input. Unlike buttons or other technologies like the rotary encoder, a joystick can read input from two axes at a time, sometimes even including center-select functionality making for a more compact and efficient footprint.

Some of the disadvantages of joysticks are the size, cost, and durability. Compared to other options, such as buttons and d-pads, joysticks tend to have a much larger form factor because of the usage of potentiometers and their housing to allow for a larger range of motion. Some aspects of this could be addressed by utilizing a stick that is shorter in height to reduce the amount it protrudes from the device, however, the width and depth of the component itself are still quite bulky for use in a wrist-mounted device. Because joysticks rely on the aforementioned internal potentiometer design, excessive use and wear can cause a phenomena called ‘stick drift’ where the joystick is unable to center itself properly and its default position sends a weak directional signal when it should not. This can be seen in several different hand-held electronics today, such as with the Nintendo Switch's controllers [53]. One last drawback of joysticks is their price. Some of the more advanced joystick designs are able to avoid stick drift and the majority of wear that renders some standard joysticks useless over time, however they are much more expensive and only typically utilized in industrial endeavors. While precision and durability are a great consideration of ours in this project, the difference in cost for these more advanced models is too much to warrant purchasing them and instead to mitigate the cost if joysticks are utilized, we will look into more cost-effective models.

A joystick would be perfect in this project for navigating the menu of the device as well as for smooth movements within applications that require more precise control over movement. A joystick would also benefit the device’s form factor by only requiring one component for bi-direction input, rather than multiple buttons or dials, which would take up a larger portion of the device’s surface.

Rotary Encoder

Another option we considered was the rotary encoder. Rotary encoders are typically used for continuous output along a singular axis. A rotary encoder is a sensor that determines the angular position of a rotating shaft. There are several different types of rotary encoders, split based upon the output signal produced and the sensor which is utilized. The two different output signal configurations of rotary encoders are incremental and absolute. Incremental decoders provide information relative to the motion of the shaft. Incremental decoders cannot provide information about the actual position of the encoder rather just when there is a change in position.

Conversely, absolute decoders provide the exact angle at which they are located at the moment they are powered on. Absolute decoders are much more precise but are also more complex and expensive than incremental encoders, which makes them much less practical for instances when price-point is a major concern. The different types of sensing technology utilized by decoders are magnetic, optical, and laser [54], [55]. Magnetic rotary encoders utilize magnetic fields in order to detect rotation. Essentially a magnet is attached to a rotating shaft and a magnetic sensor measures the magnetic field as the shaft rotates. The change in the magnetic field is converted into an electrical signal which represents the rotation or angle [56].

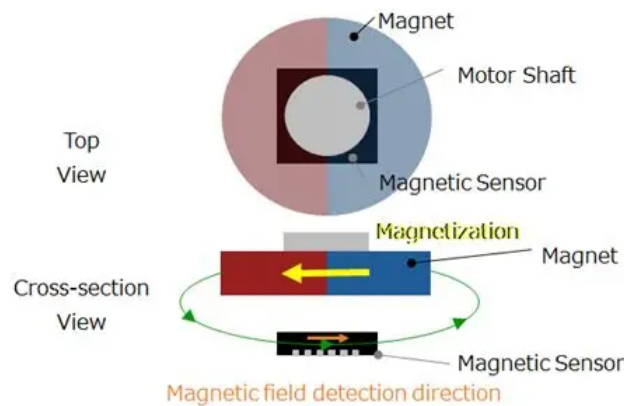


Figure 3.2: Magnetic Rotary Encoder Diagram [56]

Optical encoders use a patterned disk and shine light through it to detect rotation. Essentially a light source either reflects off the disk or shines through, where a light sensor (photodiode) detects the pulses of light as the disk rotates and converts those pulses into electrical signals [57].

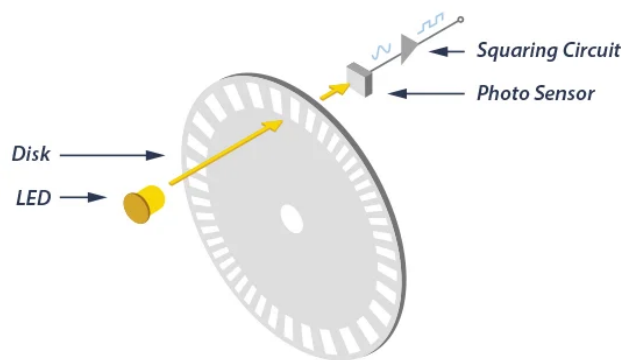


Figure 3.3: Optical Encoder Diagram [58]

Lastly, laser encoders detect precise rotations using a laser beam. In this configuration, the laser beam is pointed at a reflective pattern on the rotating disk from which it is reflected onto a

photodetector which determines the change in position based upon the pattern of how the laser is refracted [59].

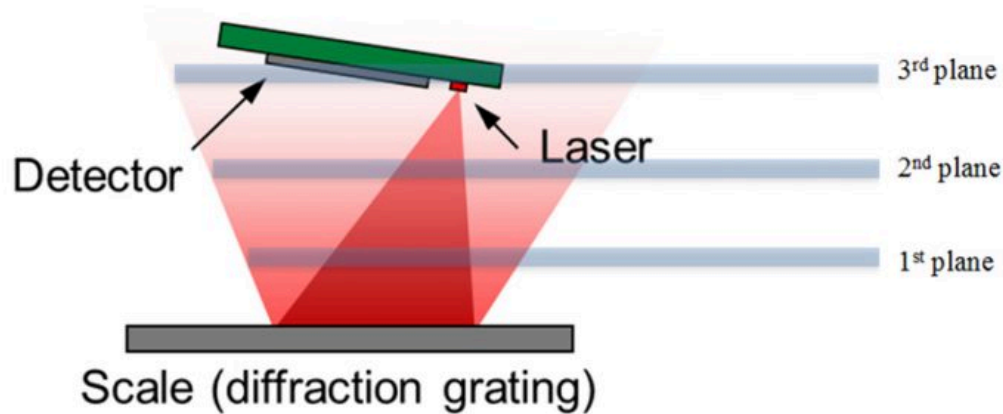


Figure 3.4: Laser Encoder Diagram [59]

Some advantages of rotary encoders are that they provide precise feedback, are durable, and have a variety of different configurations to fit different purposes. Unlike some of the other options presented in this section, rotary encoders can run in either an incremental or absolute configuration, meaning that it can either output the displacement information or the absolute location of the component with high accuracy for each. Additionally, because there is no direct contact between any moving parts like the potentiometer configuration of the joystick, rotary encoders are much more durable and are not nearly as susceptible to the wear of these connections as they do not require them. Rotary decoders also come in several different configurations with two forms of data transmission and three means of measuring the change in angle/position.

The usage of rotary encoders also comes with some drawbacks as well. The high-precision of some rotary encoder models, such as the laser configuration, come at a high cost which is. Rotary encoders are also much more complex than any other component in this comparison, and the proper implementation is a much steeper learning curve. Rotary encoders are very sensitive and their readings could be disturbed by dust, dirt, hair etc. which are present in every-day life. Arguably one of the largest short-comings of rotary encoders is that they only provide output along a single axis. Lastly, the installation of this part could prove quite difficult and to process the signal, additional hardware may be required, which is not ideal. Rotary encoders could be used for menu navigation or fine parameter tuning in this project. They would be a great fit for applications where highly-accurate single axis input is needed.

Comparison

Table 3.13: Physical Interface Comparison

	Buttons	D-pad	Joystick	Rotary Encoder
Input Type	Binary (on/off)	Digital, 4-way	Analog or digital multi-axis	Incremental or absolute (single-axis)
Precision	Low (binary only)	Moderate (4 direction but possible mispress)	High (smooth, multi-directional control)	Very high (precise angular measurement)
Complexity	Very low	Low	Moderate	High
Durability	High	High	Moderate	High
Cost	Very low	Low	Moderate to high	Moderate to very high
Form Factor	Small, flexible placement	Compact, low profile	Larger footprint, protrudes from device	Varies (can be compact but requires careful installation)
Feedback	Physical click	Tactile, sometimes center button	Smooth analog motion, possible center-click	Rotational feel

Between buttons, joysticks, d-pads, and rotary encoders there are a multitude of viable options for the physical interfaces with the device. Each device is optimized for a specific function for the device, such as buttons for simplicity, rotary encoders for precision, the d-pad for intuitive navigation, and the joystick for versatile control. Buttons are the simplest option and are fairly durable, extremely low cost, and are easy to implement, however they cannot represent any high-level input and instead can only indicate whether it is pressed or not. D-pads are widely utilized for directional input in handheld consoles and controllers. D-pads solve the issues of form factor with a compact, low-profile design and are also still relatively simple to implement. Unfortunately though, d-pads are unable to provide incremental directional input, and instead, like buttons, only provide discrete output of whether a direction is pressed or not, making it more suited for robust directional input. Dissimilarly, Joysticks can provide more dynamic input, with analog directional communication so that the connected components/devices can gain information of gradual changes in direction which is optimal for fine-tuned, precise navigation of

menus and other applications on the device. Lastly, we have rotary encoders which are very interesting technology which provides incredible precise output. The largest disadvantage of this is that it only applies to one-dimensional input and its usage would take up double the space in comparison to a dual-dimensional component like a joystick. Additionally, rotary encoders are quite expensive and could prove costly in this build. For these reasons we are able to rule out rotary encoders from use in this build. Compared to buttons, joysticks and d-pads provide a much more ergonomic form factor compared to continuous pressing of buttons, which can become uncomfortable after prolonged use. Buttons do not meet the complexity requirement for this project, as they only serve the single function of binary output, additionally, some joysticks and d-pads have center-select functionality, which could render a separate select button redundant. For these reasons we find it safe to rule out standard buttons for usage in this project. For this project we feel it best to utilize a d-pad in conjunction with a joystick, to allow for dynamic directional input and smooth navigation between menus with an ergonomic form factor that is comfortable enough for prolonged usage by the user.

3.4.1.2) Directional Pad (D-Pad)

In this project, the inclusion of a directional pad is used for more robust directional selection and navigation. This is most useful when navigating simple mobile-mode UIs and potentially some games if stretch goals and objectives are met. When researching different popular d-pads the following models were recommended: the JS1300AQ, ALPS SKRHADE010, and the SparkFun COM-26850.

The JS1300AQ is a low-profile four-way direction navigation switch with center select functionality. This part is fairly durable with each direction, including center-select, rated at an operating life of 200,000 actuations. The operating temperature range is more than sufficient to operate the device safely without part failure with a range of -40°C - 85°C . The voltage and current ratings for this component are both ideal for our device at 12v and 50mA respectively. The directional operating force of the JS1300AQ ($1.77\text{N} \pm 0.59\text{N}$) is comparable to the other options with the same base force required as the SparkFun model and within 0.5N of the base of the ALPS model. This part has the highest operating force required for the center-select [60], [61].

The ALPS SHKHADE010 is a digital four-direction navigation switch with center-push functionality. This part has a voltage rating of 1V to 12V which is well within the range that it will be operating in for use in the device. The footprint of this model is the smallest in this section at a size of 7.45mm by 7.5mm by 1.85mm. This part is very durable, with an operating life of 1,000,000 cycles for each direction and the push button, which is the highest of all the options offered in this section by a large margin. Such a long operating life also means that even with moderate to heavy use the part will still last quite a while. Another differentiating factor for

the ALPS SHKHADE010 compared to other options is that it is a surface-mount component. This difference is minor and the connected PCB design can easily be adjusted accordingly [62], [63].

Lastly, the SparkFun COM-26850 d-pad is also a 4-direction navigation switch with the center-push function. Although this is the smallest temperature range offered from these models, this part has an acceptable operating temperature range from -25°C to 85°C . This part has the largest footprint by a small margin, and is only less than 1mm larger than the next largest model in every dimension, making this differentiation practically moot. Overall the COM-26850 is the least expensive part at about \$0.95, however this does come at the detriment of durability. This part has the worst overall durability with an operating life of only 50,000 cycles [64].

Overall, each of these components satisfy the directional resolution and center-push functionality requirements of this build. These parts also satisfy the temperature requirements of this build as they each One of the largest differences among these components is the mount-type; both the JS1300AQ and SparkFun COM-26850 are through-hole parts whereas the ALPS SKRHADE010 is a surface mount part. This is not much of a concern as we can easily incorporate either mount style into our daughter-board schematic and can easily replace either configuration if the part is faulty or broken. In terms of part life, the clear front-runner is the ALPS model with an operating life of 1,000,000 cycles per each direction. This is five times as long as the JS1300AQ and 20 times as long as the SparkFun model. Due to its much shorter operating life in comparison to the other two models, the SparkFun COM-26850 can be ruled out for this project. The largest differences remaining between these two models are the form factor and cost. Of these models, the JS1300AQ model is approximately 2 to 3mm larger in both length and width than the SKRHADE010. Conversely though, the JS1300AQ is listed for \$0.98 lower than the ALPS model. Another thing to note is the difference in force required to trigger the center-push functionality. The ALPS SKRHADE010 has a lower threshold of force required to activate this center push at a base force of 2.35 Newtons, compared to the 3.14 Newtons required of the JS1300AQ. The more force required to activate the center-select, the more fatigue the user may feel after prolonged use, whereas if the threshold for required force is so low it could trigger inadvertently with minimal pressure. While the difference between these two values is minimal it still influences this characteristic on a small-scale, leading to more of a preferential concern rather than a functional one. Weighing these options, operating life and smaller form factor are desired over a slight difference in price, leading us to select the ALPS SKRHADE010 for the directional pad.

Table 3.14: D-Pad Model Comparison

	JS1300AQ	ALPS SKRHADE010	SparkFun / COM-26850
Mount Type	Through-hole	Surface-mount	Through-hole
Directional Resolution	4-direction	4-direction	4-direction
Center Push	Yes	Yes	Yes
Operating life	200,000 Actuations	1,000,000 cycles (for each direction)	50,000 cycles (min for each direction)
Operating Temperature	-40°C - 85°C	-40°C - 85°C	-25°C - 85°C
Voltage Rating	12V	1V - 12V DC	12V
Current Rating	50mA	10 μ A - 50mA	100mA
Operating Force (Directional)	1.77N $\pm$ 0.59N	1.2N $\pm$ 0.69N	1.77N $\pm$ 0.34N
Operating Force (Center-Push)	3.14N (+0.78N / -0.59N)	2.35N $\pm$ 0.69N	1.77N $\pm$ 0.34N
Dimensions	10.40mm x 9.80mm x 1.55mm	7.45mm x 7.5mm x 1.85mm	10.5mm x 10.5mm x 1.70mm
Cost	\$2.18	\$3.16	\$0.95

Although we had initially chosen the ALPS SKRHADE010 d-pad module, we actually saw fit to design one from scratch on the same board as the joystick using Adafruit 3101 8mm soft tactile push buttons. This gave us more control over button placement and made it easier to 3D model our buttons themselves.

3.4.2) Joystick

A joystick is crucial in the design of the device as it allows the user to accurately navigate the user interface more so than using the touchscreen alone. This part is practical when precise movements or selections are required and a mouse or other peripheral devices are not connected. This is especially useful in instances when the device is in mobile mode and there is a smaller

element on the screen that may be difficult to access utilizing the touchscreen alone (and considering peripheral devices are typically not connected in mobile mode). By using the joystick, the user is able to easily select the desired item instead of repeatedly tapping the screen and potentially missing the touch target. We viewed several different joysticks available on the market and found a few representative examples in the C&K THB001P, Parallax 27800 Joystick Module, Adafruit Mini 2-Axis Analog Thumbstick, and the KY-023 Joystick Module.

The C&K THB001P is a two-axis analog miniature thumbstick with a center-press switch. This model determines position using two 10k Ω potentiometers along both axes. This part is one of the smaller options available sitting at 17mm by 21mm by 19.45mm. This part is also fairly durable, with an operating life of 2,000,000 cycles for the joystick and 1,000,000 cycles for the center switch. Unfortunately, none of the other parts listed include an operating life specification, however this seems to be on the higher end of most comparable components. While it is not expected to reach the extremes of the temperature range, this is by far the best of the options provided at a range of -10°C - 70°C [65].

The Parallax 27800 joystick module is also a two-axis analog thumbstick. Unlike all of the other options presented, this part does not have center-select functionality which is not desired for this project. This is arguably the largest option available in this section at 41.67mm by 35.56mm by 27.94 mm which could alter the footprint of the build significantly if not properly accounted for. The Parallax 27800 is also the highest cost by a margin of at least \$4.99 from the next-most expensive option, which is a significant difference. One thing to note about this model is its low operating current, which is 10-50 times less than the other three options. Although this means the part can operate at higher voltages, such a low operating current is not ideal for this project [66].

The Adafruit Mini 2-Axis Analog Thumbstick is a two-axis analog thumbstick. This option does include the built-in push button which is desired for this project. The dimensions of this model are comparable to the THB001P, coming in slightly smaller at 17.5mm by 17.4mm by 12.0mm and representing the smallest model of those provided. This is tied with the KY-023 joystick module for the most inexpensive option at \$2.50. Unfortunately the temperature rating of this model is not provided as the datasheet is not as extensive as desired [67].

Lastly, is the KY-023 joystick module. This part is also a two-axis analog design and uses two 10k Ω potentiometers, with one across each axis and is equipped with a built-in push-button. This part's temperature range (0°C - 70°C) falls within a safe and reasonable range for our project. The KY-023 is also tied with the Adafruit mini for the least expensive component in this section, at \$2.50. This model tends towards the median of the form factors available of these parts at 34mm by 26mm by 32mm [68].

As previously mentioned, with the exception of the C&K THB001P, the operating life and both operating forces are not available in any of the product datasheets. As a whole, these parts all satisfy the temperature and voltage requirements of the device. The largest differences arise when we compare the presence of center-push functionality, where we can see that the Parallax 27800 joystick module is the only one without this function. As a result we can effectively remove this component from consideration. Of the remaining options, we can see a slight difference in the parts' current rating, with the Adafruit model rated for approximately 10mA in comparison to the 50mA rating of C&K's THB001P and the KY-023 joystick module. In the interest of better current rating, we can eliminate the Adafruit model from contention. This leaves the C&K THB001P and the KY-023 joystick module. Of the two, the C&K model has a much smaller form factor at a comparable price (only \$0.46 more). This, paired with the fact that THB001P has a much more extensive datasheet and its specified operating life potentially allows for years of heavy use, we select C&K's THB001P for this component.

Table 3.15: Joystick Model Comparison

	C&K THB001P	Parallax 27800 Joystick Module	Adafruit Mini 2-Axis Analog Thumbstick	KY-023 Joystick Module
Mount/ Connection Type	Through-hole	Through-hole	Through-hole	Through-hole
Directional Resolution	4-direction	4-direction	4-direction	4-direction
Center Push	Yes	No	Yes	Yes
Operating life (for each direction)	1,000,000 cycles	500,000 cycles	Not specified	Not specified
Operating Temperature	-10°C - 70°C	0°C - 70°C	Not specified	0°C - 70°C
Voltage Rating	$\leq 5V$	$\leq 10V$	3.3V-5V	5V
Current Rating	$\leq 50mA$	$\sim 1mA$	$\sim 10mA$	$\leq 50mA$
Operating Force (Directional)	$1.8N \pm 0.5N$	Not specified	Not specified	Not specified
Operating Force	$4.0N \pm 0.8N$	Not specified	Not specified	Not specified

(Center-Push)				
Dimensions	17mm x 21mm x 19.45mm	41.67mm x 35.56mm x 27.94 mm	17.5mm x 17.4mm x 12.0mm	34mm x 26mm x 32mm
Cost	\$2.96	\$7.95	\$2.50	\$2.50

Although we had initially chosen the C&K THB001P, we reverted our decision to the Adafruit Mini 2-Axis Analog Thumbstick since it offered a lower profile and seemed easier to use as a cursor due to it being more like a multidirectional slider as opposed to a tilting joystick you would find on most video game controllers today.

3.4.3) Camera

The camera in our build was more like a plug-and-play feature that was quite simplistic to introduce for our model. However, it's not to say that our camera was especially selected for its I²C interface (due to simpler connection complexity and lower data lines), size, and cost efficiency. When compared to most other cameras on the market, this particular option was one of the very few to ever fulfill all of our requests and stay within our budget range to keep the overall system cost reduced.

We also evaluated other potential camera modules for this project. The IQL-IMX335/FF offers a high resolution (2626×1960) and a MIPI interface, which provides more advanced imaging capabilities but comes at a higher cost (\$50 plus shipping) and larger size, making it less suitable for our compact design [69]. The Raspberry Pi Camera module. Our selected choice is a full color 4608×2592 pixels solution with an I²C interface, which is around \$32.95, meaning it is not as expensive as the IMX335. The resolution is 4608×2592 pixels, which is the greatest out of the three options also having up to 120 FPS which is the greatest of the three as well [70]. Lastly, the IQL-OV7676/FF was considered as another offering with VGA resolution (640×480) with an I²C-compatible SCCB interface. While it is compact, its lower resolution and limited frame rate (up to 30 FPS) made it less attractive compared to our primary choice for more detailed image capture requirements. It also costs 25 dollars, so this option, while less expensive than the other two options, provides the lowest quality, making it undesirable for this project [71].

Table 3.16: Camera Model Comparison

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	I ² C	SCCB / I ² C
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)

By comparing these options, it was clear that our selected camera, the Raspberry Pi Camera Module 3, struck the best balance between functionality, ease of integration, power consumption, and cost-efficiency, fulfilling all design constraints without exceeding budget limits. This will be used for recording video.

3.4.4.1) Displays and Protocols

KIPS MkII will have a display for the user to read when not plugged into an external display. This will enable interaction with KIPS and its features while being worn. The world of screens is quite a large one; so we narrowed it down to E-Ink, OLED, and LCD panel types. We must also consider the protocol used for the screens to communicate with the Pi CM4; of which we looked at DSI, DPI, and HDMI.

LCD Screens are one of the most widely available panel types on the planet. They tend to have refresh rates of anywhere from 60Hz to 120Hz, response times of about 5ms to 20ms, a longer lifespan than other options on the market, and occupy the lower price point of commercially available screens [Appendix C, 8b]. However, LCDs tend to have a higher power consumption, as it is based on the brightness of the backlight [Appendix C, 8b].

OLED screens are becoming more popular due to their vibrant colors, brighter whites, and darker blacks [Appendix C, 8b]. Due to how OLED functions, their power consumption varies as the image changes. This is because OLED tends to turn pixels all the way off when displaying colors like dark grays or blacks. OLED screens have refresh rates of 60Hz to 120Hz and have an incredibly fast response time of less than a millisecond [Appendix C, 8b]. However, their operational lifespan is significantly shorter than other options on the market due to pixel burn in; common on OLED screens that are left on for a long time. OLED screens are also not nearly as available in multiple sizes as LCDs and boast a much higher cost to boot [Appendix C, 8b].

E-Ink displays are commonly seen on e-readers, and more recently on special tablets made for low power consumption [Appendix C, 8b]. In terms of being used for a mobile device like KIPS Mk II, they almost only have downsides. E-Ink displays require ambient light in order to see due to having no backlight, they have refresh rates that usually top out around 10Hz, and typically have a response time of over one second [Appendix C, 8b]. However, their saving grace is their midrange price point and long lifespan, supported by the fact that E-Ink displays only consume power when refreshing; making them incredibly power efficient [Appendix C, 8b].

Let us now go over the protocols we researched. The DSI (Display Serial Interface) protocol is directly compatible with the Raspberry Pi CM4 [12]. It boasts a low latency, can display up to 1440p, tends to be low cost when finding compatible displays, and only consumes a moderate amount of power in comparison to other options [Appendix C, 9b]. However, it isn't as commonly available as other types. DSI communicates between the source and the display via flat flex cables [Appendix C, 9b].

DPI (Display Pixel Interface) is another type of display protocol. Though it has a similar name to DSI, it is completely different. DPI typically connects via custom cable or GPIO headers, and takes up a significant amount of power in the process. There are commercially available DPI displays for Raspberry Pi called HATs (Hardware Attached on Top) that can take up the entire GPIO header due to how DPI works; it can take up to 8 pins per color of RGB, giving a 24 bit output [Appendix C, 9b]. It also requires sync and clock signals from the source in order to function. Displays utilizing DPI have about the same availability as DSI; they are only really used for microcontroller based projects. They have low latency, can display up to 1080p, and are low cost options for DIY projects [Appendix C, 9b].

HDMI is a well known protocol used widely today in televisions, computer monitors, capture cards, etcetera. They aren't typically used for portable displays like we need due to them requiring a lot of pins for communication, consuming a lot of power, and there not being widely available displays in small form factors; especially since HDMI can support up to 4k60 on

displays [Appendix C, 9b]. Their cost can vary from relatively cheap to being very expensive depending on the product [Appendix C, 9b].

Table 3.17: Screen Types

	LCD	OLED	E-Ink
Brightness	High	High	Low to None, Needs Ambience
Refresh Rates	60Hz to 120Hz	60Hz to 120Hz	0.1Hz to 10Hz
Response Time	5ms - 20ms	<1ms	>1s
Power Consumption	Constant, Draw Dependent on Backlight	Constant, Draw Dependent on Image	Low; Refresh Only
Lifespan	Long	Varies; Image Burn in Possible	Very Long
Cost	Low - Medium	High	Medium
Availability	Widely Available	Limited	Limited

Table 3.18: Screen Communication Protocols (Reference ChatGPT)

	DSI	DPI	HDMI
Connector	Flat Flex Cable	GPIO Header or Custom Cable	HDMI/Micro-HDMI
Complexity	2-4 Data Lanes + 1 Clock	High; Up to 24 Data Lines	High; TMDS Differential Pairs
Latency	Low	Low	Very Low
Power Consumption	Moderate	Higher	Highest
Cost	Low to Medium	Low Cost	Wide Range
Availability	Moderate	Moderate	Low for Small Panels
Resolution	Up to 1440p	Up to 1080p	Up to 4K60

Given the parameters above, we have decided to seek out an LCD screen that will communicate using DSI protocol. This is because of the cost, availability, and support for LCD screens; not to mention that the DSI protocol is directly compatible with the Raspberry Pi CM4 that we are using to power KIPS Mk II [12].

3.4.4.2) DSI LCD Screen

Being that we want to use an LCD screen that communicates via the DSI protocol, we sampled two products from Waveshare, and one from OSOYOO on Amazon. The display from OSOYOO is a 3.5 inch capacitive touch LCD, one from Waveshare is a 4.3 inch capacitive touch LCD, and the final one from Waveshare is a round 4 inch capacitive touch LCD.

The OSOYOO brand 3.5 inch LCD has an 800x400 resolution on an IPS panel featuring capacitive touch. It has a 60Hz refresh rate with a 170° viewing angle, and it uses DSI protocol [72]. This option is the least expensive of the three coming in at around \$30 to \$45 dollars depending on sale or point of purchase.

The Waveshare 4.3 inch DSI LCD has an 800x480 resolution on an IPS panel featuring capacitive touch. It has a 60Hz refresh rate with a 160° viewing angle, and uses DSI protocol [73]. This option is the midrange in price, coming in to be anywhere from \$40 - \$50 depending on sale or point of purchase.

The round Waveshare 4-Inch DSI LCD is a circular display which features a 720x720 resolution on an IPS panel featuring capacitive touch. It has a 60Hz refresh rate with a 160° viewing angle, and uses DSI protocol [74]. This option is the most expensive of the three, costing around \$50.

Table 3.19: DSI LCD Screens (references above)

	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

Given the parameters above, we have decided to go with the 3.5 inch DSI display from OSOYOO. 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.

3.5) Audio Suite

KIPS Mk II will have an audio suite which converts I2S digital audio from the Raspberry Pi Compute Module 4, as well as combines that signal with that of a digitally controlled AM/FM radio, which will require its own antennas. This will enable the user to play audio from the SBC as well as listen to the radio if they so choose, even in the low power mode described in chapter 2.

3.5.1.1) SBC Digital Audio Signal Transmission

Digital to analog signal conversion will be instrumental in this project simply because the Raspberry Pi Compute Module 4 outputs its audio through GPIO in a digital protocol (I2S) that is unusable by regular speakers [12]. Without it, we would have no analog output of any kind from the SBC itself. To this end, we looked through three separate options for handling this; Digital-to-Analog Converter ICs (DACs), Bluetooth DAC ICs, and a USB audio interface. These technologies all have their own benefits and drawbacks.

Before we get into different kinds of circuits that can handle the transmission and conversion of the digital audio signal, we need to talk about the choice of going with an analog system or a digital communication protocol. Analog signal transmission is usually the most straightforward option, as it would require no interfacing or programming; hence why it isn't even a protocol, but rather a signal medium. This allows it to be directly compatible with things like headphone jacks, speakers, RCA output, etcetera. It also comes with the added benefit of having low latency; though unfortunately, it is inherently more subject to noise than digital protocols, and has no programming to handle errors. USB audio is a digital transmission protocol that utilizes specialized hardware (e.g. a USB controller) to send whatever data it is given to a host device that is compatible with the USB protocol. USB can transmit files, audio, and even video; though this means the device it sends all of this data to does the decoding itself. For example, a user could plug a USB speaker into KIPS Mk II to get sound, but all of the digital to analog conversion and whatever signal handling that comes with that is usually handled by the connected device. Due to this, USB can sometimes have more latency than an analog solution as a result of the more complicated circuitry required to transmit over the USB protocol [Appendix

C, 10b]. Bluetooth is the final protocol we looked at. It tends to be convenient for the user as it allows the device to connect to anything that communicates via Bluetooth (e.g. headphones, speakers, keyboards, mice, etcetera). However, Bluetooth also requires its own special circuitry in order to function, and comes with limited signal quality; as well as being much more open to noise than analog. Furthermore, Bluetooth is notorious for sometimes having inconsistent latency and signal drops, which we in the group have all experienced ourselves in using Bluetooth products. Just as a final point on Bluetooth, the Pi CM4 already has Bluetooth built in [12], so designing separate Bluetooth hardware would make no sense; especially if an off the shelf Bluetooth solution could be utilized through a USB port in place of dedicated Bluetooth hardware.

Digital-to-Analog Converters directly connect to the audio bus of the source and convert it to an analog signal, resulting in many benefits; low latency, high signal quality, less noise, and less voltage drop. In order to achieve this though, manufacturers recommend keeping the trace lengths short between the source and the DAC itself, (Figure 3.5 Cadence) due to longer traces being more prone to noise, as well as signal timings possibly being thrown off [75]. This brings a level of complexity to this option, as we must take care to ensure our design can function with minimal noise or interrupts. Some manufacturers make recommendations within the datasheets of their ICs as to how you should create a layout which properly supports the chip on a PCB. Another benefit of having a DAC is the smaller footprint it creates due to the compact nature and layout of the parts. All of the bigger parts (e.g. op amps, transistors, etcetera) have been compressed down inside of the IC itself, providing a smaller footprint and easier integration for better performance. The downside to having the benefit of this small footprint is that the DAC itself is not very customizable on a lower level, though for our needs this won't be a major issue. DACs contained in an IC package also have the benefit of being efficient with usage of power, which is important considering our project will be operable off not just wall power, but battery power as well, which means we have to be able to save on amps wherever we can. A final benefit of using a DAC IC, though obvious, is the analog output it provides. This allows direct integration with things like summation amplifiers, speakers, 3.5mm output, etcetera; though we will touch on those topics later on in this paper.

Signal Frequency	Clock Period (T)	Max Skew Tolerance (20% T)	Propagation Delay	Max Trace Length Mismatch
100 MHz	10 ns	2 ns	~150 ps/in (FR4)	~13 in
200 MHz	5 ns	1 ns	~150 ps/in	~6.7 in
500 MHz	2 ns	0.4 ns	~150 ps/in	~2.7 in
1 GHz	1 ns	0.2 ns	~150 ps/in	~1.3 in
2 GHz	0.5 ns	0.1 ns	~150 ps/in	~0.67 in
5 GHz	0.2 ns	0.04 ns	~150 ps/in	~0.27 in

Figure 3.5: “Rule-of-Thumb Guidelines for PCB Signal Propagation Delay and Trace Length Matching”[75]

Another option for dealing with the digital output of the Pi CM4 would be a Bluetooth DAC IC. Though they are similar with some of their benefits and downsides, a lot of the benefits of a traditional DAC are lost in this option. With a Bluetooth DAC, you no longer need to have the DAC itself closer to the source itself as noted by Figure 3.5 from Cadence PCB Solutions. This also makes it compatible with most sources that can connect via Bluetooth, making it ideal for Bluetooth speaker solutions; not necessarily something we had in mind for our project but was considered nonetheless. Having the DAC be wireless via Bluetooth also introduces the unavoidable problem of latency and inconsistent audio quality due to collisions (Figure 3.6: “Figure 1 - A collision taking place on channel #2”) and transmitter/receiver synchronization (Figure 3.7: “Figure 2 - Missed packets in unsynchronized communication”)[76]. The Bluetooth DAC option comes with the major downside of inherently having more added parts than a dedicated DAC IC. It needs to be able to drive not only the DAC, it also has to drive its own Bluetooth capability, which adds a larger footprint, as well as a higher power draw than a standalone DAC. A final downside to choosing this for our project comes with understanding that this requires more user interfacing than a dedicated DAC, and if there are issues with connectivity, that cannot simply be solved through just swapping a component as Bluetooth issues are usually solved through software and communication protocol.

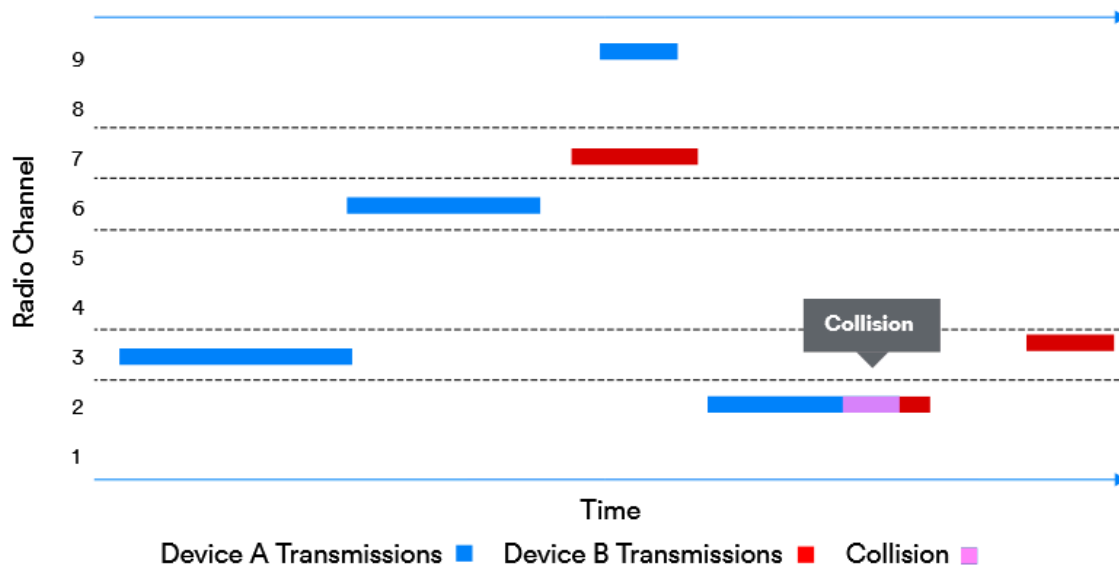


Figure 3.6: “Figure 1 - A collision taking place on channel #2”[76]

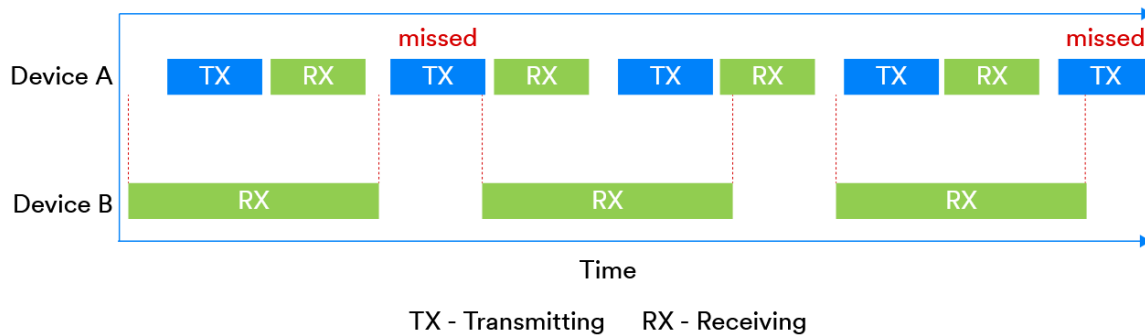


Figure 3.7: “Figure 2 - Missed packets in unsynchronized communication”[76]

The third and final technology we looked at was a USB audio interface. This option was quite interesting due to it being completely digital on the KIPS side of things, which means a lot of accessories (e.g. some speakers, headphones, USB to 3.5mm converters, etc.) would be essentially ‘plug and play’, and all of our analog audio needs would be handled by external devices. However, this would require us to change the entire audio system to digital, requiring a USB controller in place of a simple DAC. This would just add more power consumption on top of having whatever power draw comes from the connected accessories to the USB port, and would also create a situation where the user would have to carry extra accessories just to get audio output from KIPS. It is also worth noting that the customizability of a USB interface for audio would be just as minimal as the other two, considering all it allows for is a USB controller; which in and of itself is more complicated due to needing more complex timing and data routing

as opposed to a regular DAC IC.

Table 3.20: Signal Transmission Options

	GPIO	USB	Bluetooth
Latency	Lowest Latency	Average Latency	Highest Latency
Customizability	Very Little Customization	Very Little Customization	Very Little Customization
Integration Complexity	Least Complex	Somewhat Complex	Most Complex
Connection Type	Integrated on PCB	Integrated on PCB	Integrated on PCB w/ Wireless Connection
Footprint	Small Footprint; DAC Required	Bulky Footprint; USB Controller Required	Bulky Footprint; Bluetooth Controller and DAC Required
Signal to Noise Ratio	Lowest	Moderate	Highest
Protocol	Analog (No Protocol)	USB (Digital Protocol)	Bluetooth (Digital Protocol)

Given the above factors and research, we have decided to go with a GPIO based DAC IC and build a custom audio system as opposed to having it handled externally by an internal USB controller. This will allow us to save power, and is directly compatible with the Raspberry Pi CM4's I2S digital output.

3.5.1.2) Analog Communications

We sampled several DAC integrated circuits, and boiled down the search to offerings from Cirrus Logic, Analog Devices, and Texas Instruments. Those being the CS4334, MAX98357A, and the PCM5102A. These are all relatively low power, affordable options for making a DAC meant for an embedded system.

The CS4334 is a relatively low power IC which occupies the higher end of the three in terms of power requirements. It requires an input voltage of 4.75V to 5.5V, an input current of 15mA to 19mA, it consumes from 75mW to 104mW of power, and has an internal reference voltage of 2.2V [77]. The CS4334 also communicates using the I2S we need; with a resolution of 24 bits[77]. Notably, this chip requires an extra clock input compared to the other two due to it

lacking an internal PLL [77], which means Phase Locked Loop. A PLL is an RF circuit which compares a reference signal to the input signal to generate a new set signal [78]. These help to reduce unwanted noise within a signal on a circuit before making it out of the DAC, and is a great feature for personal audio projects.

The MAX98357A is a relatively low power combination IC that contains both a DAC and an Audio Amp. It has the lower power requirements of the three at an input voltage of 2.5V to 5V, an input current of approximately 2.4mA, it consumes approximately 12mW of power, and uses input voltage as a reference voltage [79]. The MAX98357A also communicates using I2S protocol with a resolution range of 16 bits to 32 bits [79]. This DAC having a built-in audio amplifier before the output makes it popular for commercially available breakout boards used in many DIY projects [79]. Normally, this would be a good thing, as it would save PCB space and lower power consumption. However, for our needs, we would want a separate audio amplifier; as combining the output signal of the summed AM/FM circuit with the pre amplified DAC output could cause the signal to become poor quality due to the differences in levels, this is avoided by having both signals put through their own summation amplifiers to achieve a mono signal for the speaker, then an audio amp which boosts the signal to drive the speaker itself. Those will be covered next.

The PCM5102A is a relatively low power IC that settles around the middle in power consumption of the three chosen ICs. It takes an input range of 3.0V to 3.6V, an input current of 7mA to 13mA, and uses the input voltage as a reference [80]. The PCM5102A communicates using the I2S protocol and employs a resolution in the range of 16 bits to 32 bits [80]. A unique feature of this DAC is that it contains an internal charge pump, which eliminates the need for DC blocking capacitors on the output lines, and gives us the benefit of a single supply input which is caused by the charge pump creating an internal inverted source, allowing for a ground referenced output [80].

Table 3.21: Digital-to-Analog Converter Comparison

	CS4334	MAX98357A	PCM5102A
Resolution	24 bits	16 bits < x < 32 bits	16 bits < x < 32 bits
Reference Voltage (Vref)	2.2V	V _{in}	V _{in}
Input Voltage (V _{in})	4.75V < x < 5.5V	2.5V < x < 5.5V	3.0V < x < 3.6V
Output Voltage	3.25V < x < 3.75V	Not Listed	2.1 Vrms

Input Current	$15\text{mA} < x < 19\text{mA}$	$\sim 2.4\text{mA}$	$7\text{mA} < x < 13\text{mA}$
Power Consumption	$75\text{mW} < x < 104\text{mW}$	$\sim 12\text{mW}$	$21\text{mW} < x < 46\text{mW}$
Supported Interfaces	I2S	I2S	I2S

Given the above comparisons and table, we feel the best option for our project would be the PCM5102A due to its typical operating voltage of 3.3V (reference PCM5102A datasheet), the charge pump eliminating our need for DC elimination as well as the benefit of the single supply input. The others have the single inputs as well, but the CS4334 would require output capacitors due to a lack of a charge pump. The MAX98357A was held down in our decision making by it already having an amp built in, which will cause issues when trying to combine signals in a summation amp for the speaker to interpret.

3.5.2.1) AM/FM Radio

An AM/FM radio was originally a feature of KIPS Mk I, but did not end up making the final cut due to time, budget, and space requirements. We wanted Mk I to have a radio as a free to access entertainment option in both low power and regular modes. It also adds a level of utility to the device, as it allows it to serve as an emergency radio listening device during events such as hurricanes or tornadoes. There appear to be three common options nowadays for mobile radios; ICs designed to run AM/FM radios, discrete transistor radios, and hybrid designs that combine ICs with discrete transistor designs.

AM/FM ICs are perfect for portable radio designs. The first obvious benefit of using an IC is their compact size, and low power draw. Throughout this paper, we will find in most cases that using an IC as opposed to a discrete circuit will lead to the same drawbacks and benefits on the surface level. The compact layout and low power draw have the downside of little customizability as well as repair being harder due to smaller size and a lot of the bulkier components being compressed down within the IC, just like with the DAC; as well as an IC by nature being more proprietary than a bunch of transistors, capacitors, and diodes; they may go out of production in future, adding to the difficulty of repair. However, due to that compact nature, AM/FM ICs come with a major benefit in the way of tending not to need tuning of their components due to it being handled inside of the IC itself. The abstract of an article from Brad Brannon at Analog Devices Inc. explains how integrated digital receivers allow for software to program filters, thus simplifying design requirements of the receiver overall [81].

Discrete AM/FM receivers usually implement larger components in a portable package as opposed to a digitally controlled IC design that handles most of the components inside of the package. This comes with the obvious benefit of having more precise control over the design. However, this comes with the drawback of the receiver being less efficient compared to an IC

option, as well as it taking up much more space in terms of PCB footprint as seen in Figure 3.8 by icstation.com and Figure 3.9 by rasiomuseum.org. Note that some of the components in Figure 3.8 by icstation.com are elements we already considered adding (e.g. 3.5mm headphone jack, speaker out, etc.), and therefore should not be considered when looking at the space taken by the receiver alone as opposed to most, if not all, components in Figure 3.9 by rasiomuseum.org being critical for the operation of the portable radio. Another drawback of having a discrete design that one must do themselves is the need for precise RF tuning, which can be a hassle; especially in a portable design such as ours that combines many other elements. Though, with all of these parts, it makes it easier to repair due to components being generic (e.g. resistors, capacitors, transistors, etc.), meaning you can drop in any replacement component so long as it matches the specs of the original part as listed by the manufacturer.

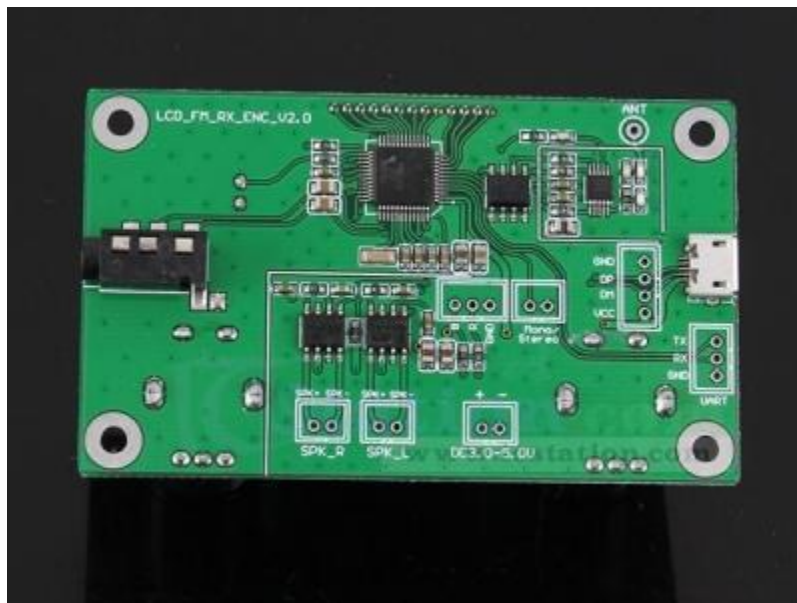


Figure 3.8: “PLL LCD Digital FM Stereo Radio Receiver Module 50Hz-18KHz Wireless Microphone Module DC 3-5V with LCD Display”[82]



Figure 3.9: “Sony Corporation; TFM-6100W” by Eckhard Wegner[83]

The third and final option would be a hybrid option, which is also common these days. This would rely on using IC based filters and other components like resonators, adding to the overall complexity of the design. However, this design would cut down on space when compared to a discrete non-IC based radio receiver, but not as much as a solely IC based one. Hybrid designs usually bring up the middle ground for efficiency, clarity, and power consumption as well; though they still require a lot of tuning just like their discrete counterparts.

Table 3.22: AM/FM Radio Modules

	Discrete	Hybrid	IC
Power Draw	Highest Draw	Moderate Draw	Lowest Draw
Power Efficiency	Least Efficient	Moderately Efficient	Very Efficient
Tuning	Requires Custom Tuning	May Require Custom Tuning	Minimal or None Required
Footprint	Largest Footprint	Moderate Footprint	Smallest Footprint
Repairability	Most Repairable	Moderately Repairable	Least Repairable

Given the above factors, we have decided to go with an IC as opposed to a discrete or hybrid design due to the AM/FM radio being a secondary feature of KIPS Mk II. An IC will do the trick perfectly fine, giving us adequate clarity and output for our needs. The added benefit of not needing to tune the IC very much, as well as the lowered footprint requirements pushed this option to the top due to the space and time that saves on design. Being able to control the radio via software (e.g. the Pi CM4 or ESP 32 output) was a large consideration as well.

3.5.2.2) AM/FM Radio IC

To select an AM/FM IC for KIPS Mk II, we sampled two products from Skyworks, and one from Sony; these being the SI4735, SI4703, and CXA1538 respectively.

The SI4735 from Skyworks is a portable radio IC that supports Amplitude Modulation (AM), Frequency Modulation (FM), Short Wave (SW), and Long Wave (LW) bands [84]. For our purposes, AM and FM will do just fine. This IC takes an input voltage of 2.0V to 5.5V, an input current of 0.1mA to 21.5mA, consumes anywhere from 30 μ W to 110mW in power, and outputs from 54mVrms to 67mVrms in AM mode; as well as 72mVrms to 90mVrms for FM mode [84].

The CXA1538 from Sony supports AM and FM bands of radio [85]. It takes an input voltage of 3V to 6V, an input current of 3.5mA to 15mA, consumes anywhere from 10.5mW to 90mW of power, and outputs from 35mVrms to 138mVrms in AM mode; and performs the same in FM mode [85].

The SI4703 is a radio IC that only supports the FM band [86]. It takes an input voltage of 2.7V to 5.5V, an input current of 0.3mA to 16.8mA, consumes anywhere from 810 μ W to 92mW of power, and outputs from 72mVrms to 90mVrms in its single FM mode [86].

Table 3.23: AM/FM Radio IC Comparison

	SI4735	CXA1538	SI4703
Supported Bands	AM/FM/SW/LW	FM/AM	FM
Input Voltage (Vin)	2.0V < x < 5.5V	3V < x < 6V	2.7V < x < 5.5V
Output Voltage (FM, Vrms)	72mV < x < 90mV	35mV < x < 138mV	72mV < x < 90mV
Output Voltage (AM, Vrms)	54mV < x < 67mV	35mV < x < 138mV	N/A

Input Current	$0.1\text{mA} < x < 21.5\text{mA}$	$3.5\text{mA} < x < 15\text{mA}$	$0.3\text{mA} < x < 16.8\text{mA}$
Power Consumption	$\sim 30\mu\text{W} < x < 110\text{mW}$	$10.5\text{mW} < x < 90\text{mW}$	$810\mu\text{W} < x < 92\text{mW}$

Given the provided specs, we have decided to choose the SI4735 AM/FM/SW/LW IC in spite of its higher power consumption range due to its features, form factor, and lower power consumption when considering battery level voltage input.

3.5.3.1) Antennas

The radio for KIPS Mk II of course will need two antennas. One for Amplitude Modulation (AM) band, and one for Frequency Modulation (FM) band. This is simply due to the fact that AM radio operates on frequencies of around 530 kHz to 1700 kHz, with each signal being differentiated in steps of 10kHz, and FM radio operates on a specific band of 88mHz to 108mHz broken into 100 channels each being 200kHz wide. It is possible to use one antenna for both AM and FM band radio, but we will get a clearer signal for both if we give each band a dedicated antenna. For this, we have looked at whip antennas, loop antennas, and dipole antennas. We considered many features such as orientation, polarization, impedance matching, and noise immunity; just to name a few.

Whip antennas are commonly seen on portable radios and vehicles. They come in a few configurations, but most are the telescopic kind you see on pocket radios or portable boom boxes; the kind you see on the tops of vehicles are fixed whip antennas. This is good because a telescopic whip antenna is perfect for a portable design such as ours. Whip antennas are omnidirectional with linear polarization, meaning their orientation isn't super important when it comes to receiving a signal; which is especially important when we consider that KIPS will be worn on the arm, meaning the direction of the antenna will change as the user moves. Another great feature of the whip antenna is that they generally do not need additional tuning in order to make them compatible with the FM band; nor is impedance matching 100% important due to them being a receive only component, so they can get away with using 75Ω coaxial connectors despite themselves having 36Ω impedances due to having a low reflection loss (appx -0.572 dB), which can be demonstrated with a simple mathematical calculation (1). However, since whip antennas don't have a positive and negative end to create a voltage centered at 0V, like a loop or dipole antenna (see Figures 3.X and 3.X for reference), they need a sufficiently sized grounding plane to function properly, so we must take care in PCB design to account for that [87]. This is due to whip antennas essentially being a half wave dipole that has been chopped in half, so instead of it having another end to deal with negative energy, the ground plane handles it itself [87]. For vehicles, this can be seen as the car itself being the ground plane for the whip antenna [88].

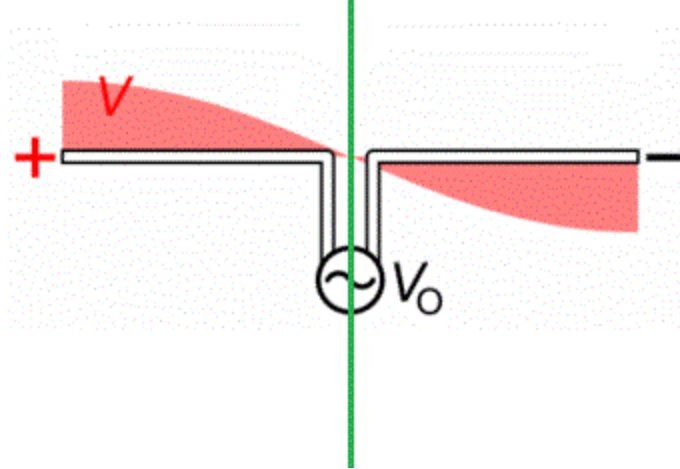


Figure 3.10: Dipole (Half Wavelength) Antenna Voltage Center by Andy Aka on Stack Exchange[89]

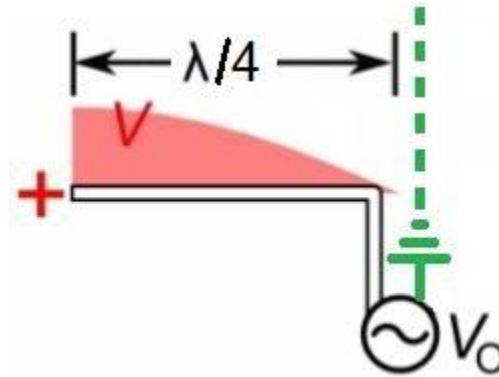


Figure 3.11: Whip (Quarter Wavelength) Antenna Voltage Center by Andy Aka on Stack Exchange [89]

$$e_r(dB) = 10\log_{10}\left(1 - \left|\frac{Z_{load} - Z_0}{Z_{load} + Z_0}\right|^2\right) \quad (1)[90]$$

$$e_r = 10\log_{10}\left(1 - \left|\frac{36 - 75}{36 + 75}\right|^2\right) = \sim -0.572dB$$

Loop antennas are commonly used in AM radios and shortwave (SW) applications for amateur radio enthusiasts. They tend to be much more compact than dipole antennas, which is important for our design. One feature of note is that loop antennas operate off of the magnetic field as opposed to the electric field like whip antennas and dipole antennas [91]. This allows them to have excellent reception indoors [91], within reason (e.g. no thick walls or isolation). The ability to operate clearly indoors comes from the fact that e-field based antennas might experience more

interference from electricity within a building due to conduits, outlets, etcetera. Another benefit of a loop antenna is that they have a narrow bandwidth influenced by their design. This, however, means they have lower sensitivity to signals [91], and their polarization depends on their orientation; either vertical or horizontal. A formula for a simple ferrite rod loop antenna can be found below (2).

$$L_f = \mu_0 \mu_e N^2 \frac{A}{l_f} (2) [92]$$

Dipole antennas are commonly used in radio applications, specifically FM and amateur bands. This is because they come with the added benefit of being easy to make in a DIY setting, being highly efficient, and they also have the ability to resonate across the FM band and beyond. They typically come in two packages, half wave and folded, each boasting around a 73Ω and 300Ω impedance respectively [93]. We must also consider that the folded variant will require a lot more impedance matching in order to be compatible with 75Ω coaxial connectors. Dipole antennas are also very large, especially the regular half wave variants, and have a vertical or linear polarization that is extremely dependent upon their orientation; making them nonideal for mobile device use.

Table 3.24: Antenna Type Comparison

	Whip	Dipole	Loop
Size	Compact (Half-Dipole)	Very Large	Compact
Impedance	$\sim 36\Omega$	$\sim 300\Omega$	Varies w/ Resonance
Matching	Unnecessary for Receiving	Matching to 75Ω Necessary	Not Needed
Polarization	Vertical (Linear)	Vertical/Horizontal (Linear)	H-Field
Orientation	Omni-Directional	Critical for Reception	Dependant Upon Signal Polarization
Efficiency	Less Efficient Than Dipole	High Efficiency	Average Efficiency
Tuning	Unnecessary for FM Band	Unnecessary for FM Band	Design Influences Frequency Band

Noise	Prone to E-Field Noise	Prone to E-Field Noise	Prone to H-Field Noise
-------	------------------------	------------------------	------------------------

Given the above factors, we have decided to go with both a whip antenna, as well as a loop antenna. This will allow us to have hardware fit for purpose on each band we wish to listen to. Loop antennas for our purposes are small and are not affected by electrical fields, making them ideal to be put inside of KIPS. Whip antennas are compact as well, making them able to be stowed away on the side of KIPS like a lot of pocket radios that are available for purchase today. Both of these antennas do not require any impedance matching in the grand scheme of things as shown by their reflection loss calculations, as well as the fact that they will not be used for transmission; only receiving.

3.5.3.2)Whip Antenna/Loop Antenna

Fortunately, telescopic whip antennas you see on all sorts of pocket and home radios are a dime a dozen online, and don't really have much to differentiate them on the basis of "features" aside from their length, their connector type, and their impedance; but for the sake of comparison we have selected three whip antennas we found online from Amazon, Ebay, and Alibaba. For the AM band, we have decided to design a loop antenna from scratch to add some challenge to our project, leaving this section solely to the selection of our whip antenna.

The whip antenna from Amazon was billed as a Superbat brand telescopic FM antenna for portable radios. It connects using f-type coax but comes with three other connector types; those being a male and female PAL connector, as well as a 3.5mm headphone jack connector, all of which give the antenna a line impedance of 75Ω [94]. The antenna is a measly 13.5cm when collapsed, but when extended is around 51cm [94].

The Alibaba OEM antenna (for lack of a better name) boasts the same basic features as the Superbat antenna, and has a 75Ω impedance, but does not come with the extra connectors from what we could tell [95]. This antenna comes in shorter though at 9.5cm collapsed, and 44cm extended [95].

The Ebay antenna I found also shows the same impedance and connection type as the other two, but does not explicitly state the collapsed length; only the extended length of 76cm and the 75Ω impedance of the connector [96].

Table 3.25: Whip Antenna Comparison

	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	75 Ω	75 Ω	75 Ω

Just 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.

3.5.4.1) Signal Summation

KIPS Mk II's audio suite will require some sort of signal summation. This is due to both the DAC and the AM/FM radio ICs we chose having stereo outputs, and the possibility of implementing both a single speaker (mono output) and/or a 3.5mm headphone jack (stereo output preferred) means we will need some kind of combination of the DAC and AM/FM signal to be output through a single source. To that end, we looked into using a passive signal summation system using a resistor network, op amp based summation, and transformer based summation. We considered many factors, including but not limited to power consumption, footprint, complexity, and signal integrity.

Passive resistor summation (voltage division) is an incredibly simple option that requires no external power. However, it involves zero amplification whatsoever, meaning there is absolutely no gain control over the signals coming in and going out. On top of that, there will be substantial signal loss due to the voltage division that occurs with a resistor network like this; also often resulting in a high output impedance.

Another option is to use op-amp based summation amplification. We learned about these over the course of electronics classes during our time at UCF. Op-amp based signal summation is great because it has the capability to scale the signals we give it however we want through customizable gain via configuring the op-amp using mathematical formulas to determine input and feedback resistors to produce the desired result. Op-amps also come with the added benefit of having lower output impedance, which is what we need in a system for summing output signals. The downside is that we have to consider our bandwidth in the design, which can

accidentally limit the range of signals that can be accepted through the amplifier. Op-amp based signal summation also requires power to function, which is a big downside in battery based systems like ours where every volt and amp matter.

A final option for signal summation would be transformer based summation. Although this option doesn't require active power input like op-amps, they are worse in about every other way. The components required for this are bulky compared to an op amp (Figures 3.12 and 3.13), and they suffer from limited bandwidth [97]; as well as this solution being very expensive compared to the other two options.

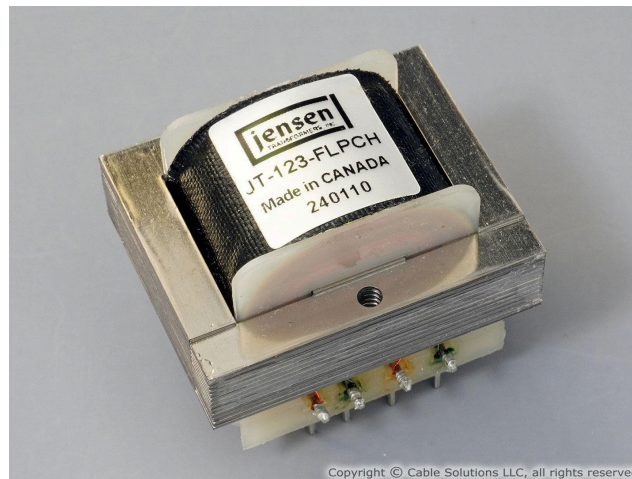


Figure 3.12: “Jensen Transformers JT-123-FLPCH Line Output Transformer, PCB Mount” by Cable Solutions, LLC[98]

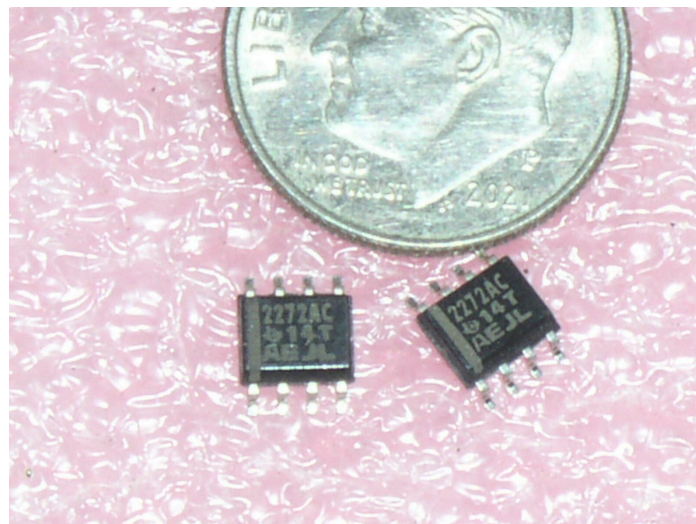


Figure 3.13: “2 TI TLC2272AC 2 MHZ RAIL TO RAIL INPUT OUTPUT DUAL OP AMP IC NOS SOIC-8 USA”[99]

Table 3.26: Signal Summation Options

	Passive Resistor Network	Op-Amps	Transformers
Cost	Low Cost	Low Cost	High Cost
Footprint	Minimal	Average	Large
Power Requirements	None	Power Required	None

Given what little there is to compare between the three options, we decided to go with op-amp based summation due to cost, familiarity, and overall customizability of the technology. Transformers are too large and expensive, and resistor networks have too much loss for our needs.

3.5.4.2) Summing Amplifiers (Op-Amps)

For our audio system, we will need summation amplifiers to handle the signals of both the AM/FM IC and the DAC. We want these to be combinable into single signals so that they will not be fighting over the output to the speaker and headphone jack.

For this section we sampled offerings from Analog Devices and Texas Instruments, those being the ADA4528-2-E and the ADTL082 from AD, and the TL082 from TI. These are all common dual input single op amps used for many purposes like summation amps, buffer amps, and boost amps, which make them excellent candidates for use within our project.

The ADA4528-2-EP is a low cost op amp advertised for use in things like electric scales, medical devices, and handheld testing devices; just to name a few [100]. It has a single supply range within which our 3.3V regulator output will fit very well, has an offset voltage in the range of microvolts as opposed to millivolts, and has a rail to rail output swing; meaning the peaks of the signal can be very close to the supply voltage without clipping [100]. The AD4528-2-EP also has high minimums for CMRR and PSRR at 135dB and 130dB respectively, which is wonderful for noise rejection; a huge benefit for having clean signals [100].

The ADTL082 is a low cost op amp alternative to the TL082 typically used for things like active filter design, power controls, buffering, and signal sampling [101]. It has a single supply range from 6V to 36V, outside of the 3.3V output we want to operate it off of; however, the dual supply range is $\pm 3V$ to $\pm 18V$, fitting our power requirement nicely with the caveat of needing an inverted 3.3V line to operate the power supply, as well as an output swing of $\pm 12V$ to $\pm 13.5V$ [101]. The ADTL082 also has a max offset current of 9mA, and a CMRR and PSRR lower than the ADA4528-2-EP, but higher than the TL082 of which it was based upon; coming in at a

minimum of 80dB for both.

The TL082 is billed as a low cost op amp used for applications that require lower offset voltages, input current biases, high slew rate, and input impedance [102]. It shares the same single and dual supply voltages as the ADTL082, which comes with the same caveat of needing a -3.3V rail for dual supply [102]. The TL082 also carries a higher offset current than the other two op amps we looked at, maxing out at 20mV while also having the lowest CMRR and PSRR of the three with a minimum of 70dB [102].

Table 3.27: Op-Amp Comparison

	ADA4528-2-EP	ADTL082	TL082
Input Voltage (V _{in})	Within V _{supp} Range	Within V _{supp} Range	Within V _{supp} Range
Single Supply Voltage (V _{supp})	$2.2V < x < 5.5V$	$6V < x < 36V$	$6V < x < 36V$
Dual Supply Voltage (V _{dsupp})	$\pm 1.1V < x < \pm 2.75V$	$\pm 3V < x < \pm 18V$	$\pm 3V < x < \pm 18V$
Offset Voltage (w/o drift)	$0\mu V < x < 4\mu V$	$0mV < x < 9mV$	$0mV < x < 20mV$
Input Current	$\pm 10mA$	$\pm 10mA$	$\pm 10mA$
Output Swing	Rail-to-Rail	$\pm 12V < x < \pm 13.5V$	$\pm 12V < x < \pm 13.5V$
CMRR	$\geq 135 \text{ dB}$	$\geq 80 \text{ dB}$	$\geq 70dB$
PSRR	$\geq 130 \text{ dB}$	$\geq 80 \text{ dB}$	$\geq 70dB$

Given the specs laid out above, we have decided to go with the Analog Devices ADA4528-2-EP. This is because of the single supply voltage allowing for us to avoid an inverted signal being needed, the benefit of an exceptionally high CMRR and PSRR, and a rail-to-rail output swing to boot; all allowing for clearer signal than if we used more common components like the TL082 or ADTL082.

We decided to go with the MCP6002 instead due to sensitivity issues with the ADA4528-2-EP. The MCP6002 is also cheaper, and is more suited for the needs of line level audio, since it can be

driven from 0V to 3.3V, they also have a low power requirement and are low noise for general use op amps.

3.5.5.1) Volume Controls

The audio for KIPS Mk II will need some kind of volume control before final amplification and output. For this, we considered analog potentiometers, IC based digital potentiometers, and stepped attenuators. Our primary criteria when considering these technologies were cost, complexity, taper type, and compatibility with analog signals.

Plain old potentiometers were the first option we looked at. They are simple components that are intuitive and easy to use. Potentiometers require absolutely zero firmware or setup aside from installation into the circuit path. These parts are also so versatile they can work with virtually any source; this is why they're used in applications like guitars, audio control boards, digital and analog audio amplifiers, headphones, etcetera. Unfortunately, potentiometers tend to run into scratchiness and signal drop out as they wear. They also have the tendency to emit noise, heard as a hum, when improperly grounded; author Scott Valentine has had personal experience with this in modifying electric guitars, which operate off of purely analog signal transmission. Potentiometers work on a logarithmic taper, which is ideal as human hearing too works on a logarithmic curve as opposed to a linear curve [103].

IC based potentiometers are another option we looked at. These offer a different method of control than the other options we looked into; digital control. Potentiometer ICs allow for us to use our MCU to have a buttonless control solution, which saves space on top of allowing us to have a part with a longer operational life cycle. However, the digital control adds a layer of complexity not present in non-IC based options; with a power input requirement being yet another downside to boot. Potentiometer ICs work on a logarithmic curve just like analog ones [103].

Stepped attenuators are vastly different from the other two. They are essentially stepped potentiometers that work off of a resistor bank that increases resistance in steps through adding more and more resistors in series. This makes them more expensive, as well as being substantially more bulky (Figures 3.14 and 3.15). The fact that they have a stepped taper as opposed to a logarithmic taper makes them less ideal for analog audio applications as well; though the massive resistor network in practice would help to reduce a lot of unwanted noise.



Figure 3.14: “DIY stepped attenuator of the shunt type made with 0.5 watt metal-film resistors” by Dimitar Sumerski from audio-at-home.com [104]

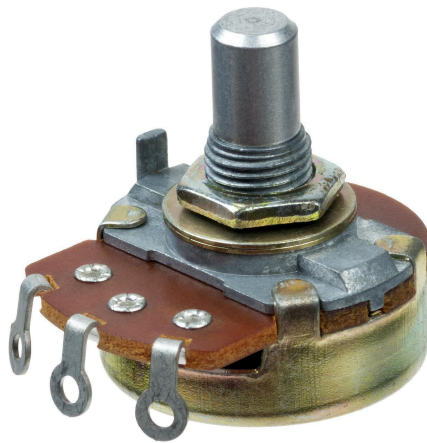


Figure 3.15: “A typical single-turn potentiometer” [105]

Table 3.28: Volume Control Options

	Potentiometers	Digital Potentiometer IC	Stepped Attenuator
Cost	Lowest Cost	Low to Medium Cost	Highest Cost
Complexity	Extremely Simple	Somewhat Complex	Most Complex
Taper	Logarithmic	Logarithmic	Stepped
Analog Compatibility	Compatible; Ideal	Compatible; Ideal	Compatible; Not Ideal

Given the above information, we have decided to go with analog potentiometers due to their wide availability, ease of install/replacement, and their logarithmic scale that works well with human hearing of audio.

3.5.5.2) Volume Potentiometers

The audio suite for KIPS MkII will require a volume control between the final amplifiers for both the headphones and the speaker. We feel an analog option would be best since it will allow the DAC, AM/FM IC, and summation amplifiers to operate at their intended levels the whole time, leaving signal attenuation to be done just before final amplification. There is a wide variety of analog volume controls out there, but we narrowed in on volume sliders, stereo potentiometers, and mono potentiometers; each with their own benefits and drawbacks. We chose a stereo potentiometer from Taiwan Alpha (model no. RV16A01F-41-15R1-A10K-30H4), a slide potentiometer from Dongguan Tianqian Electronics (model no. C2031N-1A2-B10K-GP), and CTS Electronic Components (model no. 450T328S103A1C1). We will refer to them by the respective company names for sake of simplicity due to the long model numbers.

The stereo potentiometer from Taiwan Alpha boasts an overall resistance of $10\text{k}\Omega$, with a tolerance of $\pm 20\%$, which means the actual value of the potentiometer can be anywhere from $8\text{k}\Omega$ to $12\text{k}\Omega$ [106], which is acceptable for our needs. It boasts a max power rating of 0.25W , and a mechanical life cycle of around 15k cycles [106], which is a full back and forth rotation of the potentiometer. Most importantly, it operates on a logarithmic taper and a stereo (L/R) signal [106]; this is great for analog signals being perceived by the human ear, especially since we are using a headphone jack. These signals are measured in decibels (dB), which in and of itself is a logarithmic ratio.

The slide potentiometer from Dongguan Tianqian Electronics is the most different of the three. It too boasts an overall resistance of $10\text{k}\Omega$ with a tolerance of $\pm 20\%$, but it begins to differ in its

life cycle, power rating, and taper [107]. It boasts a 0.1W power rating, a mechanical lifecycle of 10k-15k cycles, and most importantly a linear taper [107]. This is great for attenuating digital signals, not so much for analog signals. When using linear attenuation on an analog signal, the sound may begin to sound quieter but only on the lower end of the signal.

The CTS Electronics mono potentiometer is more closely related to the Taiwan Alpha stereo pot in the sense that it is a traditional rotational potentiometer. Most commonly used in audio equipment and guitars, this potentiometer has the tightest tolerance of the three options we sampled, with a resistance of 10k Ω at a $\pm 10\%$ tolerance, giving it a 9k Ω to 11k Ω range of resistance [108]. It too operates on a logarithmic taper just like the Taiwan Alpha potentiometer, but boasts double the power rating at 0.5W, with a tradeoff of having a lower rated mechanical lifecycle at 10k cycles [108].

Table 3.29: Volume Potentiometer Comparison

	RV16A01F-41-15R1-A10K-30H4	C2031N-1A2-B10K-GP	450T328S103A1C1
Resistance	10k Ω	10k Ω	10k Ω
Taper	Logarithmic	Linear	Logarithmic
Tolerance	$\pm 20\%$	$\pm 20\%$	$\pm 10\%$
Mechanical Life	15k Cycles	10k - 15k Cycles	10k Cycles
Power Rating	0.25W	0.1W	0.5W

Given the choice between the three, we have decided to go with the Taiwan Alpha potentiometer. This is a good middle ground of the three. Its loose tolerance and midrange power rating is bolstered by the higher mechanical lifecycle, as well as it giving us the option to work with stereo audio; which is beneficial for systems with headphones as an option.

3.5.6.1) Amps and Outputs

KIPS Mk II will need analog compatible audio outputs, as well as the proper amplifiers to drive them. To that end, we considered amplification options first. When considering them, we looked at cost, power draw, footprint, customizability, and complexity. As for output types, we were going to look at USB audio, speakers, and 3.5mm headphone jacks; but as we already saw earlier on, we decided against USB, so it will not be mentioned in this section going forward. When considering outputs, we thought about ideal applications, quality, and power consumption.

Amplification ICs offer a consistent audio quality in a compact and cost effective package.

However, with their limited size and footprint comes a limited power output as opposed to discrete or hybrid designs due to working off of a set power supply [109]. Their design is also fixed, so there is no chance of having extra output without something like a boost op-amp to add extra power.

Discrete amplifiers are the polar opposite of IC based amplifiers. They use bulky components and draw a lot of power in order to drive high quality, high powered outputs; making them popular with high end audio setups for things like recording, audiophile grade music listening, and stage use. This also means they take up a lot of space on a PCB with their highly customizable but complex layouts, which consequentially makes them very expensive.

Hybrid amplifiers take elements of both discrete and IC based amplifiers. They tend to be less power hungry than discrete designs, but more than IC based designs. Hybrid designs can sometimes be less expensive than discrete options depending on how you design your amplifier, but they will still boast a large footprint regardless.

For outputs, we first looked at a built-in speaker. This would allow a user to have audio output without the need of an external accessory; though this may be an inconvenience to others around them due to noise. Speakers tend to be pretty power hungry in the audio output world; with small ones drawing anywhere from 0.5W to 3W of power on average, pending their impedance and size. Speakers are also well suited for mobile systems like ours simply for their convenience as mentioned earlier; but it is worth noting that sound quality varies widely from speaker to speaker. We also looked at a 3.5mm headphone jack as an option. Headphone jacks tend to be much lower power than speakers. They also allow a user to have a private listening experience as opposed to their radio and computer audio being broadcast to those around them. 3.5mm headphone jacks also enable external accessory support; provided they are compatible with the headphone jack itself. Vehicles, headphones, speakers, and even some microphones use 3.5mm cables for connection.

Table 3.30: Amplification Options

	IC	Discrete	Hybrid
Cost	Low Cost	High Cost	Medium Cost
Power Draw	Moderate Power Draw	High Power Draw	Medium Power Draw
Footprint	Small Footprint	Large Footprint	Large Footprint
Customizability	None	High Customizability	Some Customizability
Complexity	Low Complexity	High Complexity	High Complexity

Table 3.31: Output Types

	Speaker	3.5mm Headphone Jack
Applications	Audio Listening/Sharing	Audio Listening/External Accessories
Quality	Varies	Varies
Power Consumption	High Consumption	Low Consumption

For outputs, we decided to go with both. We figured it would be a better experience overall to allow sound sharing and private listening. A user could also attach a higher power speaker if they so desired if the included speaker does not meet their needs.

To support these output mediums, we decided to go with IC based amplification to save space, as well as keep the final amplification step as simple as possible.

3.5.6.2) Headphone Amplifier

KIPS MkII will have a headphone jack, which requires its own amplifier separate from the audio amplifier for the speaker. This is due to differences in signal strength, and line impedance (e.g. consumer earbuds running off of 32Ω vs small speakers running off of 4Ω or 8Ω). For this we sampled two amplifier ICs from Texas Instruments, and one from Nisshinbo Micro Devices; these being the LM4910, LM4808, and the NJU72040.

The LM4910 from Texas Instruments is billed as being a 35mW headphone amplifier with an output at 32Ω ideal for use in mobile devices such as phones, PDAs, and MP3 players[110]. As

specified already, it has around 35mW of output on a 3.3V supply while only drawing around 6mA of current, delivering a highly effective and low power solution for a device just like ours [110]. The datasheet also specifies that it eliminates the need for coupling capacitors on the output, as well as pop and click prevention circuitry to suppress noise during activation [110].

The LM4808 from Texas Instruments is billed as being a dual audio amp that can deliver up to 70mW of power per channel at an of 32Ω with a 5V power supply [111]. This IC is a bit higher power than the LM4910, which is why it has been billed for use as a dedicated headphone amplifier in PCs and portable systems [111]. Texas Instruments claims this IC too has the pop and click suppression system in place on the LM4910, as well as needing minimal external components to operate [111].

The NJU72040 from Nisshinbo Micro Devices is billed as an audio headphone amplifier with pop suppression just like its Texas Instruments counterparts; as well as being suitable for all audio applications that contain a headphone interface [112]. It boasts an 80mW output at 32Ω to boot [112].

Table 3.32: Headphone Amplifier Comparison

	LM4910	LM4808	NJU72040
Output Power at 32Ω	30mW < x < 35mW	60mW < x < 90mW	55mW < x < 80mW
Input Voltage	2.2V < x < 5.5V	2.0V < x < 5.5V	2.7V < x < 3.6V
Input Current	0mA < x < 6mA	0mA < x < 3mA	10.5mA
Output Voltage	Not Listed	4.7V (~1.66Vrms)	2.2Vrms
Gain	Unity Gain Stable/Externally Configurable	Unity Gain Stable/Externally Configurable	5.4dB < x < 13.6dB (Selectable)
Power Dissipation	Not Listed	Not Listed	550mW
Total Harmonic Distortion	0.3%	0.05%	0.08% < x < 0.3%

Given the above factors, we have decided to go with the LM4910 due to its lower power draw and externally configurable gain. KIPS was never intended to drive larger loads like audiophile grade headphones, so we feel 35mW is more than adequate to get the job done.

3.5.6.3) Speaker Amplifier

When looking at speaker amplifiers, we sampled options from Diodes Incorporated and Texas Instruments. These being the PAM8301, PAM8302, and LM386 respectively. Each is designed to be low footprint medium power speaker amplifiers.

The PAM8301 is billed as being able to directly drive 8 Ω speakers with 1.5W at 5V [113]. Diodes Incorporated states that the PAM8301 was designed with 88% efficiency and minimal external parts; namely using filter-less designs that allow speakers to be directly driven by the IC as opposed to needing filtering [113].

The PAM8302 is billed as being able to drive 4 Ω speakers with 2.5W at 5V [114]. This IC shares many similarities with the PAM8301, hence the similar nomenclature. The PAM8302 boasts an 88% efficiency, as well as a minimally designed layout with the aforementioned ability to directly drive speakers as opposed to needing filters; just like the PAM8301 [114]. This IC is just a higher power version of the PAM8301.

The LM386 is billed as being able to accept voltage ranges of 4V to 12V or 5V to 18V for a wide range of applications including AM/FM radio amplification, television sound systems, power converters, and intercoms just to name a few [115]. This IC does not offer any of the niceties that the PAM series offers whatsoever; though it boasts a large gain range of 20 to 200 [115].

Table 3.33: Audio Amplifier Comparison

	PAM8301	PAM8302	LM386
Load Impedance	$4\Omega < x < 8\Omega$	$4\Omega < x < 8\Omega$	$4\Omega < x < 32\Omega$
Input Voltage	$2.5V < x < 5.5V$	$2.0V < x < 5.5V$	$4V < x < 12V$
Efficiency	~88%	~88%	~50%
Output at 3.3V at 8Ω	~0.8W	~0.8W	~0.8W
Output at 3.3V at 4Ω	~1.0W	~1.0W	~0.3W
Output at 5V at 8Ω	~1.5W	~2.0W	~0.5W
Output at 5V at 4Ω	~2.0W	~2.5W	~0.7W

Given the above comparison, we have decided to go with the LM386 for the sake of lower power consumption; which means we will have to choose a speaker according to the output of this amp.

3.5.6.4) 3.5mm Headphone Jack

When researching 3.5mm headphone jacks, we struggled to find any differences in features aside from how they are mounted to the PCB [116], [117], [118]. As such, this section will be short and sweet.

Table 3.34: 3.5mm Headphone Jack Comparison

	SJ-43504-SMT-TR	ASJ-125-3	SJ1-3525N
Mount Type	Surface Mount	Through Hole	Through Hole
Mounting Angle	Right Angle	Right Angle	Right Angle

Since these are all practically the same aside from mount type and shape, we decided to go with the SJ-43504-SMT-TR due to its small size and longer port; giving us more space to work with when making it fit inside a shell.

3.5.6.5) Speaker

For speakers we sampled products from Multicomp Pro, Soberton Inc., and DB Unlimited. These being the MP000291, SP-2027NW, and SW230704-1 respectively. These are all cheap, compact speakers that work perfectly for mobile applications, and should be compatible with the PAM8301 speaker amp we selected earlier.

The MP000291 is billed as a miniature rectangular speaker that is rated to accept 0.7W into its 8Ω package [119]. The manufacturer states it boasts a 90dB sound pressure level at 10cm, making it perfect for mobile devices and other compact applications [119].

The SP-2027NW is billed as a dynamic speaker unit with a 4Ω impedance that is rated to accept 1.0W of power [120]. The manufacturer states that this speaker boasts an 83dB sound pressure level at 50cm [120]. This product is the largest form factor of the three and seems to hold its volume at a longer distance as opposed to being close up.

The SW230704-1 is a round speaker meant to be used in mobile devices. It has a 4Ω impedance and is rated to accept 1.0W of power [121]. This speaker is round as opposed to square like the other two, hence it being measured in diameter as opposed to a base and height [121].

Table 3.35: Speaker Type Model

	MP000291	SP-2027NW	SW230704-1
Impedance	8Ω	4Ω	4Ω
Rated Power	0.7W	1.0W	1.0W
Max Power	1.0W	1.5W	1.5W
Max Output	89dB	83dB	90dB
Size	15mm x 11mm	20mm x 27mm	23mm Diameter

Given the above parameters, we have decided to go with the SP-2027NW due to it being rated to accept the rated output of the PAM8301 speaker amp, with plenty of room to handle power spikes that might come from the amp.

We ended up changing to the CQR0508 from CQRobot, an 8 ohm 3 watt speaker simply because it better suited the output of the LM386 speaker amp, and it also fit in our enclosure much better.

3.6) Other

For this project we also require a few other components that do not quite fit cleanly into other categories. In this section we explore the different flashlight technologies available and make a selection for the best fit for this project as well as the PSU and other assorted systems and components. These technologies, while miscellaneous in nature, still contribute a decent amount to the project overall and should not be overlooked.

3.6.1) Flashlight Types

For this project, a flashlight module will be added to the design. To ensure the most suitable technology is chosen, three options were evaluated: LEDs, incandescent bulbs, and laser diodes.

LEDs (Light-Emitting Diodes) are currently the dominant choice in modern flashlights due to their balance of efficiency, brightness, and durability. They achieve luminous efficiencies of 60–110 lumens per watt and offer an impressive lifespan of 10,000–50,000 hours. This makes them highly power-efficient and long-lasting compared to traditional options. LEDs are also versatile, capable of being optimized for wide flood beams or narrow long-throw beams. Their solid-state construction ensures durability, making them well-suited for portable devices that may encounter vibrations or impacts [122], [123].

Incandescent bulbs represent the oldest flashlight technology. They typically operate at 10–20 lumens per watt, which means producing significant brightness requires a high wattage and, therefore, more battery power. Their lifespan is also limited to only 1,000–2,500 hours, far shorter than LEDs. Incandescent bulbs rely on fragile filaments, which are vulnerable to mechanical shock, vibration, and impact. This lack of durability makes them less practical for portable or rugged devices [124].

Laser diodes, particularly in LEP (Laser-Excited Phosphor) flashlight designs, provide exceptional long-range beam throw. Unlike LEDs, which disperse light across a wide area, laser diodes produce a highly collimated beam capable of reaching distances over 1 km. While their luminous efficacy can exceed 150–170 lm/W in specialized designs, the actual lumen output is relatively modest—typically 100–600 lumens—since their strength lies in candela (beam intensity) rather than raw brightness. Laser diodes also require precise thermal management and optical alignment, making them more complex and fragile than LEDs. As a result, they are best suited for niche applications such as searchlights or signaling rather than general-purpose portable lighting [125].

Table 3.36: Types of Light Sources

Type	LED	Incandescent Bulbs	Laser Diodes
Luminous Efficiency	66-110 lm/W	10-20 lm/W	150-170 lm/W
Lifespan	10000-50000 Hours	1000-100000 hours	1000-25000 Hours

After comparing the three technologies, LEDs are the most suitable option for the KIPS MK II. They combine high luminous efficiency with long operational life, durability, and flexibility, without the fragility of filaments or the complexity of laser-based designs. LEDs provide consistent brightness while minimizing power consumption, making them the most reliable and practical choice for a mobile device.

3.6.2) Flashlight

The flashlight was another add-on to our design that didn't need as much attention as the other components. However, size once again was the main attribute that contributed to how we chose our flashlight for the module. For consumer appeal, we believed that it was also best that we chose an LED that of course was not only power efficient, but also visually appealing as well. And so, to fit in with today's technical standards, getting an LED that specifically provided a white cool temperature was a requirement department [122]. An LED will be the best option here as not only will its power consumption be minimal but also the luminosity will be high. What has to be considered is not only that but also other unique quantities for a light

Another option that was looked at was the CLM3C-WKW/MKW: PLCC2 1 IN 1 SMD LED. It is 2.7 x 2.0 mm, so it is very small and also has a luminosity of (1400-3550 mcd) for cool mode, and (1120-2800 mcd) for warm mode. Meaning it will be used in cooling mode for greatest cd. Bin code 29 (Vin min 3.2 Vin max 3.4). This means that it will take 3.3V which is what is going to be offered for this line. The issue with this one is that with a 120° viewing angle, and modest thermal capability, it's designed for wide, diffuse light and small currents. Meaning that it will need a lot of them to achieve an actual working flashlight [126].

The last potential option was the other type of Cree XLamp XP-L, the HD option was not chosen due to it being more designed for flood lighting. Meaning that its design is less designed for throw and more designed to spread light. This also means it will not be useful for this project. As we want a more controlled beam.

Table 3.37: LED Model

	XP-L HI	XP-L HD	CLM3C-WKW/MK W
Size	3.45 mm x 3.45 mm	3.45 mm x 3.45 mm	2.7 mm x 2 mm
Max drive current	3A	3A	20 mA
Luminous Intensity	Up to ~45000 cd	Up to ~27000 cd	1.4 -3.55 cd

The one chosen was the XP-L HI LEDs It is 3.45 x 3.45 mm. It is optimized to deliver maximum candela through secondary optics. Meaning that it will have maximum throw distance when it comes to range. For a flashlight project where throw, focused beam, and efficient conversion of LEDs candela through optics are primary goals, the Cree XLamp XP-L HI is the best practical choice: ~120% more candela vs HD, compact 3.45×3.45 mm package, max 3 A drive, $R_{\theta J-S} \approx 2.2$ °C/W, and readily available CRI/chromaticity bins to control tint — all supported by Cree’s datasheet and community testing [127].

We ended up not using any of the above LEDs at the end of our project, as the flashlight was a stretch goal. We instead used simple 10mm LEDs and a flashlight reflector in order to make a simple proof of concept of said goal. We also were not sure if we could swing enough output power to drive such high current LED clusters when it came time to implement said goal.

3.7.1) PSU

For this section, this is a general overview of every component that needs power. The main goal of this is to break down the power lines that will be used into two different sections. There will be a 3.3V line and a 5V line. These were decided upon for simplicity, therefore making each component that needs power fit into these two categories. The current that the devices will be set at, at maximum, will be 3A. The power for the system will come from a battery that will be a rechargeable battery and AC-DC converter. How these power lines will be achieved will be discussed in a further section.

The goal of this device is that it will be portable, not stationary. Meaning that for this project to be mobile, it needs to use a battery. Wall power as the only source will not allow this. And this device seeks to be used at all times, meaning a renewable source like solar is not the best, on top of then needing to install solar panels on the device, which is outside the scope of this project. The battery will be charged on wall power using a charging IC that is discussed below. Now for the battery, it then comes time to discuss the voltage coming from it. The voltage is intended to be around 7.2V, which will then need to be stepped down into the needed voltages of 3.3V and

5V. The reason this is done is for efficiency, as stepping down a voltage and then raising the current will allow for the voltage to be regulated.

Below is the table of components that need power. Notice all of the components run at either 3.3V or 5V. These will be attached to their appropriate lines. Based on the data gathered for each part, the approximate wattage is shown.

Table 3.38: Table of components that need power

Component	Supply Voltage	Min Current	Max Current	Approximate wattage
SBC	5V	0.7A	3A	3.5-15W
MCU	3.0-3.6V	~2.5 μ A	260mA	7.5 μ -0.936W
Heartbeat Sensor	3.3-5V	<10mA	10mA	33mW
Temperature and Humidity Sensor	3.1-5.5	10 μ A	950 μ A	31 μ -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	$\sim 30\mu W < x < 110mW$
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
---------------	--	--	--------------

3.7.1.1) Battery composition

For this project, the battery is one of—if not the most—important components to consider. The device must operate for extended periods while portable, which means it will rely on battery power for the majority of its operational time. Because portability is a core requirement, the system will not use an AC power adapter. Therefore, selecting the proper battery type is critical for both performance and usability.

The first battery type to consider is alkaline. These are simple, disposable batteries that can be purchased at nearly any store, typically costing between \$0.50 and \$1 per cell. They are available in common sizes such as AA and provide an output of around 1.5 V per cell. Alkaline batteries have an energy density of approximately 90–120 Wh/kg. However, they are non-rechargeable, meaning that once they are depleted, they must be replaced. If this power method were chosen, the device would require a user-friendly method for battery replacement during operation, which reduces convenience for a portable system [128].

The second battery type to evaluate is Lithium-Polymer (LiPo). These batteries come in a variety of form factors, such as soft pouch cells or rigid block-style packs. LiPo batteries typically output around 3.0 to 4.2V, their nominal voltage is 3.6–3.7 V per cell and have a much higher energy density of 150–250 Wh/kg compared to alkaline batteries. They are lightweight, making them ideal for mobile applications, and most importantly, they are rechargeable. This means the device can simply be plugged into a charging source when needed, eliminating the need to swap cells. The higher voltage per cell also reduces the number of cells required for the system to meet its power demands [128].

The third option is Nickel-Cadmium (NiCd). These batteries provide about 1.2 V per cell and have an energy density of only 40–60 Wh/kg, making them significantly less efficient in terms of energy storage per unit of weight. In addition, NiCd batteries are heavier compared to alkaline and LiPo cells—for instance, a typical AA NiCd weighs about 24 g, while an AA alkaline cell weighs about 20 g. NiCd batteries are rechargeable, but their lower energy density and higher weight make them less attractive for a portable system where efficiency and weight are crucial [129], [130].

Table 3.39: Battery Composition Type

	Lithium-polymer (LiPo)	Akaline	Nickel Cadmium
Recharable	Yes	No	Yes
Weight	Light	Medium	Heavy
Cost per cell	Higher upfront cost	Lower upfront cost	Medium
Cycle life	500-2,000+ cycles	1-time use	500–1,000 cycles
Energy density	150-250 Wh/kg	90-120 Wh/kg	40-60 Wh/kg,
Operating Temperature	-20°C to 60°C	-18°C to 55°C	-20°C to 60°C
Voltage	~3.6V	~1.5V	~1.2V
Typical uses	Drones, RC, mobile devices, high-performance gear	Remotes, clocks, low-drain toys, backup	Power tools, aviation, and industrial

After considering all three options, the Lithium-Polymer battery is the best choice for the KIPS MK II. It provides a higher per-cell voltage (around 3.6 V), the greatest energy density (150–250 Wh/kg), and is lightweight compared to the alternatives. Most importantly, LiPo batteries are rechargeable, allowing the system to be powered through a simple charging port rather than requiring frequent battery replacements. These advantages—higher energy efficiency, rechargeability, and portability—make LiPo the optimal power source for this project.

Due to lab safety concerns raised by Dr. Weeks and Dr. Chan, we deferred to using lithium ion batteries instead due to the commercial availability of preprotected cells, as well as available expertise on their use from Dr. Weeks as well.

3.7.1.2) Battery

On the matter of power comes the main component of the battery. This is very important as this is the main form of power for the device when it is not plugged in and therefore must sustain all of the components. It will need to sustain the components it is powering even at its worst and therefore must be accommodating to that. For this project to succeed at having the correct voltage for the battery to output, the correct configuration has to be chosen. The three options that are being looked at for this project are: one 3.6V cell, two 3.6V cells at 6000mAh, and two

3.6V cells at 8000mAh. The two-cell configurations will be in series and not parallel, and the composition of all cells will be LiPo as discussed previously.

The reason this project is not even considering two cells in parallel is a matter of safety. When two cells are in parallel, what can happen is the voltages will fight each other over what the dominating voltage is. This can cause an explosion if it were to occur, meaning that this cannot be an option even worth considering.

The first option to be considered is just one 3.6V cell. This option is simple as it is just one cell. This option is lightweight, simple, and compact. The disadvantage is that if you need a voltage higher than 3.6V, you will need a step-up converter (Boost). So, for options that need more voltage, this will increase design complexity relative to its simple architecture.

The next two options are based on the same type of architecture, just at different mAh values — those being two 3.6V cells in series. Since these designs are in series, the output voltage will be 7.2V. This increases efficiency when compared to one 3.6V cell due to not needing a boost regulator to increase the voltage, and also reduces current draw in the process. This also increases storage capacity due to having more cells. The downside is that these designs need to maintain the same voltage on the cells to achieve maximum voltage. Also, a second cell will increase size, weight, cost, and charging complexity. Not only that, but if the voltage needed is less than the 7.2V provided, this will then require a step-down regulator to work in the project. Additionally, a 2S protection board will be used in conjunction with the cells. This is for ensuring that the output voltages have overcharge protection, overdischarge protection, overcurrent and load short-circuit protection, and high-temperature protection. This extra component is required for using the LiPo composition that was decided above [131], [132].

Table 3.40: Battery Configuration

	One 3.6V	Two 3.6 V at 6000mAh (series)	Two 3.6 V at 8000mAh(series)
Voltage	3.6	7.2	7.2
Type	1s	2s	2s
mAh	Depends on the cell	6000 mAh	8000mAH

After consideration, the best implementation for the KIPS MK II is two 3.6V cells at 8000mAh. This is for several reasons. The series configuration allows for greater voltage output for the system to use. Not only this, but it is also the safest choice to avoid the cells fighting over which one is dominant for the voltages. 8000mAh was chosen for the greatest overhead when it comes

to storage, as the difference between 6000mAh and 8000mAh is capacity, not topology. To implement this style of battery, a step-down converter may be needed, but that will be discussed later. This design will also use the protection circuit for the LiPo batteries. The battery will be charged on wall power using the charging IC.

As stated above, we used lithium ion cells instead due to safety reasons, so we instead utilized two 3400mAh 18650 lithium ion battery cells from UltraLast. They came integrated with their own safety circuits for accident prevention.

3.7.1.3) USB charge port (Power/Charging)

For this section, what needs to be considered is the best type of power delivery port. For USB, not only does the standard of the connection port need to be considered, but also the type of physical port itself. This choice affects what types of different cables and devices can be plugged into KIPS Mk II. So, for this project, what has to be considered is what these different standards can offer and the advantages and disadvantages of them.

Our options for choosing the appropriate charging port were between a micro-USB connector and a USB-C. Though micro-USB connectors are much more common and easy to come by, we took into account that USB-C is not only becoming more widely used among many newer devices nowadays, but also utilizes more power for better power efficiency and faster charging capabilities. And though for us it does not necessarily matter, it's also important to note that USB-C is the fastest in data transfer as well [133]. As seen in the table below, the difference between the two can be seen plainly in how USB-C can have data speeds in the gigabyte range, while micro-USB is still measured only in the megabyte range. By choosing USB-C, this offers high-speed data transfers that could be needed depending on what is required. Also, power output between the two has to be taken into account. USB-C can output more than 100W, while micro-USB can offer only up to 18W. This means that USB-C completely outclasses micro-USB in this aspect. Another aspect is physical, as micro-USB is not reversible, meaning its connection has to be plugged in with a set direction. USB-C is reversible, meaning direction does not matter between upwards and downwards. Taking all of this into account means that USB-C is the clear choice to use between the two for this project [134].

Another type of charger to consider is the Lightning cable. This type of cable looks on the outside the same as USB-C, but that is where the similarities end. Lightning cable specs are closer in comparison to micro-USB; the key difference is that the Lightning cable has faster power transfer. This means the same points as to why micro-USB is not being used for this project apply too, meaning USB-C is the clear choice to use for charging. The major aspect of Lightning cables that has to be considered is that it is proprietary for Apple devices, meaning its range of usable devices is extremely limited when compared to both micro-USB and USB-C. Therefore, this is another trade-off that will have to be considered if Lightning is chosen as the connection. USB-C, meanwhile, although it is less common than micro-USB, is still accepted in

a wide range of devices. Therefore, USB-C is the preferred choice between Lightning and itself [135], [136].

Table 3.41: Charge Port Type Model

Form Factor	Micro USB	USB-C	Lightning
Power Output	Up to ~12W	Up to ~15W	Up to ~10W
Operating Voltage	5V	5V	5V
Devices applicable	Very Common	Less Common	Apple Proprietary
Reversible	No	Yes	Yes

After taking into account all three types of charge port, the one that will be chosen is USB-C. It has the fastest charging capabilities at up to 240W. The data transfer speeds are the greatest between the three types, and it has the capability to be reversible. This is why USB-C is chosen for KIPS MK II.

3.7.1.4) USB-C Power Component

When using USB-C for power there comes the need to discuss another component: a Power Delivery (PD) controller. This exists as USB-C can operate at higher voltages than other USB types but there needs to be communication between the two devices used so they can negotiate the voltage supplied/used. There is also the option of not having one but that comes with some limitations.

Using USB-C without Power Delivery (PD) is the simplest option, as it relies only on the default current advertisement through the CC pins. In this mode, a device can draw up to 5 V at 500 mA or 900 mA depending on the USB version, and up to 3 A if the source advertises higher capability. This means the maximum power available without PD is 15 W ($5\text{ V} \times 3\text{ A}$). The advantage of this approach is lower cost, simpler hardware, and no need for a dedicated PD controller, making it a good fit for low-power devices like small IoT boards, simple peripherals, or gadgets that only require 5 V operation [137].

By contrast, USB-C with PD unlocks the full potential of the standard, enabling negotiation for a wide range of voltages (5 V, 9 V, 12 V, 15 V, and up to 20 V) and currents up to 5 A, delivering as much as 100 W. A PD controller is required to manage communication and ensure safe negotiation, but the payoff is flexibility, efficiency, and the ability to power demanding devices such as laptops, monitors, and fast chargers. With PD, systems can scale power intelligently, reduce losses by using higher voltages, and support advanced features like role swapping and fast charging [137].

In short, no PD is simpler but limited, while PD adds complexity but vastly expands capability—making it essential for medium- to high-power designs where efficiency, versatility, and future-proofing matter.

Table 3.42: PD Controller VS NO PD Controller Comparison

	PD Controller	No PD Controller
Voltage	Fixed at 5 V	Multiple options: 5 V, 9 V, 12 V, 15 V, 20 V (up to 48 V with PD 3.1)
Max current	Up to 3 A (advertised via CC resistors)	Up to 5 A (with e-marked cable)
Communication	None (only CC pin resistor signaling for current capability)	Full PD protocol negotiation over CC pins
Hardware Complexity	Very simple (no PD controller needed)	Requires a PD controller (autonomous or MCU-based)

After considering both options the best one to implement is USB-C with a PD controller. This is as it will significantly increase the power capacities of the USB-C port up to 100W. This comes with having to implement a separate component, but this will be worth it for the benefit of power.

3.7.1.5) USB-C PD Controller

Since we are using USB-C for the power it needs a Power delivery (PD) controller. This is as USB-C does not only supply up to 5V but instead up to 20V. As such there needs to be a component which negotiates what is the best voltage to operate at so that both devices can deliver the highest power. It also Detects the orientation of the reversible USB-C connector (using CC1/CC2 pins) and determines if the device is a source (provides power), sink (consumes power), or dual-role (can swap between the two). The three considered are the STUSB4500, R9J02G012, and the TPS2575.

The STUSB4500 from STMicroelectronics is a sink-only autonomous USB Power Delivery controller, designed for applications where a device needs to negotiate and draw power without requiring firmware intervention. It is well-suited for embedded systems, IoT devices, and battery-powered designs because it can store up to three power profiles in non-volatile memory and handle PD negotiation independently. Its biggest advantage lies in its simplicity and ease of

integration, though it is limited to sink-only roles and is less flexible for dual-role or programmable designs [138].

On the other hand, the Renesas R9J02G012 provides a more versatile option. It supports both source and sink roles, and its programmability makes it better suited for more complex applications, such as docking stations, multi-role power systems, or consumer electronics that need flexible negotiation and role swaps. Renesas designed this controller to provide robust protection and compliance, but it generally requires more firmware development effort compared to the STUSB4500's autonomous operation [139].

The TPS2575 from Texas Instruments (often referred to as TPS25750 in documentation) is a highly integrated controller that supports both source and sink functionality, along with integrated policy management. It is targeted at applications needing advanced flexibility, such as USB-C chargers, laptops, and high-power consumer devices. Unlike the STUSB4500, which runs independently, the TPS2575 usually works alongside a host processor to implement full PD policy, but it offers more scalability and configurability. Its integration with TI's charging ICs and ecosystem also makes it appealing for designs where power management and battery charging are closely tied together [140].

Table 3.43: PD Controller Comparison

	STUSB4500	R9J02G012	TPS2575
PD spec / USB-C spec support	Supports USB PD up to 100 W (20 V, 5 A)	Supports USB PD rev 3.1 (standard power range, not EPR)	Compliant with USB PD 3.0, USB Type-C spec, supports both source & sink roles
Role of internal / external logic (policy engine)	Has built-in “auto-run” logic: no MCU needed for common sink negotiation	Requires / supports more flexible roles; firmware update support	Standalone solution: includes policy engine, so minimal external MCU logic required
Power path / switching elements	Provides integrated gate drivers (PMOS) for VBUS switching on the sink side	Does not emphasize integrated high-current FETs (more a controller)	Includes integrated switches / managed power paths (5 V 3 A, bi-directional load switch, protections)
Voltage / current ranges supported	PDOs from 5 V up to 20 V (max 5 A) in	As a PD 3.1 controller, supports	Power path supports up to 28 V, 7 A in

	sink mode	standards in specification (within SPR)	some configurations (bi-directional) in variant D, and 5 V/3 A source switch in other variants
Supply / voltage domain	VSYS / VDD ranges: VSYS = 3.0 to 5.5 V; VDD up to 22 V (dual supply)	VDD supply: 2.7 V to 5.5 V; NVCC: 1.8 V to 3.6 V	Can be powered from 3.3 V or VBUS; includes 3.3 V LDO for dead battery support
Key protection features	Short-to-VBUS protection on CC pins (up to ~22 V)	Emphasis on secure PD / authentication, but fewer built-in power-path protections listed	Reverse current protection, overvoltage protection, inrush control, robust power path protections integrated

After considering all the options the best to choose is the STUSB4500 for KIPS MK II. This is as it was the simplest to implement with it requiring no firmware to install or host intervention. This was the best choice for its wide range of voltage supported as well as how it controls where power is going. Since the KIP USB-C port will be used for charging the device and not using the device to charge others it needs to only be a sink. Therefore the STUSB4500 is the correct choice for this project,

3.7.2.1) IC Functionalities

For this project, we require two functionalities from the battery IC: charging and gauging remaining fuel. Two possible configurations exist for achieving this goal, these ICs can be packaged into one part or separated into two. In this section, we explore the two configurations available and decide which one is best suited for use in this project.

The first configuration we explore is the combined configuration. In this instance, both functionalities required would be derived from one IC. For a combined configuration, where both the charging IC and battery gauge IC are integrated into a single component, there is a decrease in design footprint since only one component is needed. This can provide several benefits in terms of power management. Because they are the same device, there is little to no risk of transition errors between the charging and gauging functions. An example of this approach is the MAX77818, which combines both a switching regulator and a power gauge in

one device. This integration brings clear advantages in compactness and simplicity, but it also introduces a major drawback: reduced flexibility. Since both functions are tied to one component, you cannot easily replace one component if an issue arises with the other, instead the whole component would need to be removed and replaced in the case of a failure in either function. In addition, the design specifications for one aspect would directly constrain the other, which can lead to compatibility issues and limit the possible range of circuit combinations. This combined approach is best suited for situations where space savings and component reduction are a higher priority than flexibility [141].

The second configuration possible is the separate configuration. This is where there is one IC dedicated to charging the battery and another which is dedicated to gauging the power remaining in the battery. This option provides greater flexibility in design specifications, allowing the charging IC and battery gauge IC to meet different requirements while still working together as a cohesive unit. This also increases scalability and replacement as it means that new parts can be added as needed if one breaks or needs an upgrade. The downside of this approach is that it requires more careful consideration of the power path, ensuring that what enters one component and passes to the other remains within the constraints of both. Additionally, this configuration increases the bill of materials (BOM) since two different components must be purchased. For example, the BQ24040 is a standalone charging IC that still offers several advantages. It features a built-in power path, with a focus on accuracy in current and voltage regulation, feedback loops, charge status display, and charge termination. Some of these features such as accuracy in current/voltage regulation and charge status could be more restricted if an integrated IC were used instead [142].

Both the combined and separate IC configurations present unique benefits and challenges. One of the most glaring concerns with the combined approach is the reduced customization, as the different individual values are geared towards both functions running together and more general applications. Conversely, the separate approach allows for much more specific values to be attained by selecting individual parts that better fit the specifications of the project. The main downsides for the separate approach however is that it could cost more (2 ICs to purchase instead of one) and it would take up more space overall.

Table 3.44: Battery IC Configuration Comparison

Type	Combined	Separate
BOM	Fewer parts, smaller PCB area; simpler layout.	More ICs, larger PCB footprint; more passives
Cost estimate	One price	2 different prices

Design complexity	Easier integration (charger and gauge already coordinated).	More flexible but requires careful coordination of charger and gauge.
Flexibility	Fixed feature set; limited charger/gauge combinations.	Choose any charger and any gauge separately; customize for unusual batteries or power paths.
Power Path	Many integrated PMICs handle input/battery/system seamlessly.	May need extra circuitry or power-path controller.
Scalability	Harder to change charger or gauge independently.	Swap charger or gauge as needed for new products.
Safety	Typically covers charger safety and some gauge monitoring.	Can mix robust standalone charger + advanced protection gauge for higher reliability.

Based on what is needed for this project, the best option is the separate ICs for the switching and the power gauge.

3.7.2.2) Charging IC

In addition to selecting the appropriate battery configuration, the charging circuit must also be designed. The role of this circuit is to safely recharge the battery when the device is connected to wall power. During this process, the charger must not only replenish the battery but also ensure that the system can continue to operate if it is in use while charging. A charging IC is a specialized integrated circuit designed to safely and efficiently charge rechargeable batteries, typically lithium-ion (Li-ion) or lithium-polymer (LiPo) cells. Its role is more than simply applying power to a battery: it manages the constant current (CC) and constant voltage (CV) charging profile, handles precharge for deeply discharged cells, enforces termination conditions, and often includes safety features such as overvoltage, thermal protection, and automatic recharge. Some advanced devices also integrate cell balancing and power-path management, which are especially important when multiple cells are connected in series.

When comparing different ICs for an application involving higher-capacity cells (such as an 8000 mAh 2S pack), several design considerations become critical: maximum charge current, support for multi-cell packs, package type, and complexity of external components.

MAX1551: This IC is a simple linear charger intended for single-cell applications only, with charge currents limited to ~100 mA (USB input) or ~280 mA (DC input). While it is compact and easy to use, it is unsuitable for high-capacity batteries or multi-cell configurations [143].

LT1769: A step-up in capability, the LT1769 is a switch-mode charger controller that can deliver higher charge currents (up to ~2 A depending on configuration). However, it is also limited to single-cell Li-ion charging, making it a poor fit for a 2S pack without additional balancing and protection circuitry [144].

LTC4006: Unlike the above, the LTC4006 directly supports 1- to 4-cell Li-ion packs, including 2-cell (2S) configurations. It can deliver up to 4 A of charge current, which makes it far more suitable for an 8000 mAh pack, especially if the desired charge rate is in the 0.25C–0.5C range (2–4 A). As a switching charger controller, it is efficient and flexible, though it requires careful design of external MOSFETs, inductors, and sense resistors. One limitation is that it does not provide built-in cell balancing, so an external 2S protection or balancing circuit must be added to prevent cell drift over time [145].

Table 3.45: Charging IC Comparison

	MAX1551	LT1769	LTC4006
Brand	Analog Devices	Analog Devices	Analog Devices
Input Voltage Range	4.25-7V	4.5 V – 28 V	4.5 V – 28 V
Output voltage	~4.2 V	~4.2 V	~4.1 X # of cells
Max charge Current	~280 mA	Up to ~2 A	Up to ~4 A
Cell Count	1	1	1-4
Architecture	Linear	Switching	Switching

In summary, while simpler ICs like the MAX1551 or LT1769 work for small, single-cell devices, the LTC4006 is a more robust solution for higher-capacity, multi-cell packs. Its ability to charge 2S Li-ion batteries with several amps of current, combined with efficiency from its switch-mode design, makes it the most appropriate choice among the compared options—provided that external protection and balancing are implemented.

Note that the LTC4006 does not gauge power, so to achieve that functionality, a separate power gauge IC will be required. The reason this part was chosen instead of a two-in-one solution—where both the charging IC and battery gauge are integrated into the same

component—was due to compatibility. Combining both functions into a single IC would heavily limit the available options for the charging system.

The LTC4006 was selected as the best choice for this application based on the reasons listed above. This means there will be a separate section for the charging IC, as it will be treated as a distinct component.

During the final design phase, we made the decision to swap to an LTC4002 li-ion battery charger due to it being a smaller package, more affordable, and has smaller BOM count for driver circuitry. It was also easier to implement overall with the exact same desired output.

3.7.3) Battery Fuel Gauge IC

In order to determine the remaining battery life of the battery, a battery power gauge IC is required. Currently our battery configuration is two batteries in series. With this configuration comes a slightly different Battery Power Gauge IC required; this configuration requires that this part be of a higher operating voltage range.

The Analog Devices DS2781G+ is an IC that measures current, voltage, temperature, and estimates the available capacity for lithium-ion batteries. Some of the most common applications of this component are in digital video cameras, handheld PC data terminals and commercial two-way radios. One of the largest drawbacks about this component is the fact that it does not have I²C communication functionality, instead it uses one-wire serial communication. While this is still a valid way of communicating with the MCU and the SBC, it is not nearly as efficient as some other communication protocols and would require a little more work to implement. The DS2781G+ has a wider voltage range at 2.5 to 10V and a relatively low current rating of about 70 microamps. Additionally, this component can support up to 2 cells. This makes the DS2781G+ a good candidate to use for the two batteries in series configuration. This component has the highest price of all of those listed in this comparison, at about \$9.46 [146].

The Analog Devices DS2786BG+ is an IC that estimates a lithium-ion battery's available capacity based upon the cell voltage in the open-state following a relaxation period. Its most common applications are digital audio (MP3) players and digital cameras. This component is able to take advantage of I²C communication protocol, which makes reading data from the component much easier in comparison to a single-wire serial format. This component has the smallest voltage range at 2.5 to 4.5V which is not large enough to accommodate a larger voltage battery or two batteries in series. Additionally, this configuration only supports a single cell, meaning only battery cell could be connected at a time, furthermore discouraging the usage of this part for a multi-cell battery configuration. Overall, if the configuration were to be reverted back to a single-cell design, then the Analog Devices DS2786BG+ may be a legitimate contender for the best fit for both its lower cost and voltage range which more closely

encapsulates the operating voltage that the battery will output. One benefit of this component is that it is a relatively affordable price, coming in at \$5.29, which is the middle price point for components in this comparison [147].

The Texas Instruments BQ34Z100-G1 is an impedance track fuel gauge for a wide variety of batteries including lithium-ion. A few common applications for this component are power tools, mobile radios, and some smaller electric vehicles. The BQ34Z100-G1 has the largest overall voltage range at 3 to 60V. This component also works independently of the battery series-cell configuration. Both of these specifications make this component a great candidate for use in this project. This component is also the most versatile in terms of communication protocol, allowing for I²C, HDQ, an alert output pin, and even four directly wired LEDs. Another benefit of the BQ34Z100-G1 is its low price. This is the least expensive part provided in this section, at \$3.80 which is \$1.49 less than the next, least expensive component, the DS2786BG+ [148].

Table 3.46: Battery Power Gauge IC Comparison

	Analog Devices DS2781G+	Analog Devices DS2786BG+	Texas Instruments BQ34Z100-G1
Interface	1-wire serial communication	I ² C	I ² C / HDQ
Quiescent Current	70μA	50μA	145μA
Voltage Range	2.5-10V	2.5-4.5V	3-65V
Max Cells Supported	1-2 cells	1 cell	Multi-cell
Accuracy	±1-2%	±1%	±1%
Price	\$9.46	\$5.29	\$3.80

For this use-case, with the two batteries in series, the best option is the TI BQ34Z100-G1, with its wide voltage range, multi-cell support, various communication protocols and lowest-cost of all of those provided. Even if the configuration were to revert back to a single-cell design, this component is still arguably the best available in this comparison.

At the advice of Dr. Weeks, we decided not to go with the BQ34Z100-G1 due to past projects having major complications with the BQ series of parts. We instead went with an Analog Devices MAX17049G+ fuel gauge since the other two options were not favorite choices anyways. It also came with the added benefit of having only three parts needed to drive the circuit.

3.7.4.1) Wall power

The KIPS MK II will not only run off the battery for power but will also operate on wall (AC) power. In this case, power will come from the wall and therefore require an AC/DC adapter.

Three different types of AC power adapters were considered for this project:

Linear Regulated Adapter

This adapter is simple in design but has a major drawback: poor efficiency. A large portion of the energy drawn from the wall is wasted as heat. In addition, linear regulated adapters tend to be bulkier than other options, which limits portability [149].

USB Power Adapter

This option is ideal for devices that require a dedicated 5V input. Thanks to standardization and safety features, USB adapters make it easy to swap cables and ensure consistent performance.

The downside is that USB adapters are limited to fixed outputs (most commonly 5V), making them unsuitable for systems that require higher voltages, such as 9V or above [133], [137].

Universal Adapter

A universal adapter can adjust its output based on the connected cable, allowing it to support a variety of standards and devices. However, this flexibility introduces lower efficiency compared to dedicated adapters. There is also a safety concern, as selecting the wrong voltage output may damage the system [150].

Table 3.47: AC Adapter Comparison

Type	Linear Regulated Adapter	USB Power Adapter	Universal / Adjustable Adapter
Advantages	- Simple design - Very low output ripple - Reliable in low-power applications	- Standardized (5V DC output) - Safe for low-power devices - Supports fast-charging protocols (PD, QC)	- Selectable output voltages - Multiple tips for different devices
Disadvantages	- Bulky and heavy (large transformer) - Inefficient (wastes energy as heat) - Limited input voltage range	- Limited output voltage/current - Mostly only for 5V or 9V devices - Not suitable for high-power	- Less efficient than dedicated adapters - User must ensure correct voltage selection to avoid damage

		applications	
--	--	--------------	--

After reviewing these options, the best choice for this project is the USB power adapter. It aligns perfectly with the system requirements of a 5V input, provides higher efficiency than the alternatives, and incorporates strong safety features without the risk of mismatched voltages. Since this is an external device, the system will only need a proper USB-C port to receive power from the adapter.

3.7.4.2) Wall Power Safety Options

For AC power to work safely, several protection measures must be in place to ensure the safety of both the user and the device. These measures are summarized in Table 3.48.

AC adapters are designed with multiple built-in safety features because they directly convert high-voltage AC (120V/240V) from the wall into low-voltage DC. As a result, they must comply with strict international safety standards. These protections help prevent hazards such as electric shock, overheating, fire, or damage to sensitive electronics [151].

Table 3.48: AC Safety Feature Comparison

Safety feature	Isolation (Galvanic Isolation)	Fusing / Overcurrent Protection	Surge Protection (MOVs)	EMI/RFI Filtering	Regulation and Voltage Control
Purpose	Prevents direct electrical connection between the AC input and the DC output, safeguarding the user.	Protects the charger from damage due to excessive current; typically includes internal fuses.	Absorbs voltage spikes from lightning or power line disturbances to prevent damage.	Reduces electromagnetic and radio-frequency interference, ensuring compliance with standards.	Maintains stable DC output voltage, ensuring safe operation of connected devices.

3.7.4.4) Power Source Switching

Now that both of the voltage sources have been decided it comes time to decide which technology will be used to switch them. This is needed as when the AC power is used then the KIPS will run off of that and charge the battery. When it is not plugged into the wall then the battery will be the main source of power and the device will run off of that. To achieve this

design a power switch will be used to connect the lines of the AC and battery power so they will go into the same power path.

Diodes are the simplest solution, often used in "diode-OR" circuits. They automatically select the higher voltage source and prevent reverse current flow, requiring no control logic. While inexpensive and reliable, traditional diodes suffer from voltage drops (0.3–0.7 V) that reduce efficiency. MOSFET-based “ideal diodes” largely eliminate this drawback, offering high efficiency with minimal added complexity. Diodes are best suited for low-power applications where simplicity and reliability outweigh programmability [152].

Relays provide a more direct switching method by physically connecting the load to one source at a time. Mechanical relays are capable of handling high currents and voltages but are limited by slower switching speeds (milliseconds) and finite mechanical lifetimes. Solid-state relays address some of these limitations, offering faster switching and higher reliability, though they may introduce additional design complexity. Relays are often chosen for systems where robust isolation and high current handling are more important than speed or programmability [153].

Switching MUXes offer the most flexibility, allowing power sources to be selected electronically via logic signals or microcontroller control. They can switch extremely fast (microseconds or faster), introduce minimal voltage drop, and enable software-based power management strategies. This makes them attractive for advanced systems requiring fine-grained, programmable control over multiple power sources. However, they typically handle lower currents than relays and require careful design to integrate protection features such as overcurrent or thermal shutdown [154].

Table 3.49: Power Switching type Model

Type	Diode-OR / Ideal Diode Circuits	Switching Multiplexers (MUXes)	Relay or Solid-State Switches
Control method	Automatic, based on voltage levels	Controlled via logic signals (e.g., MCU)	Manual or automatic (mechanical or solid-state)
Efficiency	Moderate; traditional diodes have higher voltage drop; MOSFET-based ideal diodes offer lower drop	Very high, with minimal resistance when properly configured	Low for mechanical relays due to contact resistance; high for solid-state switches

Switching Speed	Instantaneous for ideal diodes; slower for traditional diodes	Very fast, in the nanosecond to microsecond range	Slow for mechanical relays; fast for solid-state switches
Cost	Low for traditional diodes; moderate for MOSFET-based ideal diodes	Moderate; depends on the complexity and specifications of the MUX	Low for mechanical relays; moderate for solid-state switches
Reliability	Very high for ideal diodes; traditional diodes have long lifespan	High; solid-state components have long lifespan when properly rated	High for mechanical relays; solid-state switches have long lifespan but may be affected by thermal stress
Voltage range	Limited by diode or MOSFET specifications	Limited by MUX specifications; typically used for low to moderate currents	Wide; mechanical relays can handle high currents; solid-state switches are limited by component ratings

In summary, diodes excel in simplicity and reliability, relays provide high-current robustness with straightforward control, and switching MUXes enable flexible, programmable power management ideal for modern embedded systems. The choice among them depends on whether the application prioritizes simplicity, current capacity, or flexibility. The best choice for this project is the Switching Multiplexers. This is due to how they can switch between different sources the fastest and are not handled by the user. The lower current should not be an issue so long as it can supply up to 3A.

3.7.4.5) Switching MUX

For this project the next component relies on both the battery input and the AC-DC line. When the KIPS is plugged into a wall it needs to switch to using the AC-DC line and stop using battery power. Meaning that it needs a way to switch between the two lines. This is where the switching MUX comes into play to allow this change between the two sources. When selecting a power multiplexer or ideal diode controller for switching between AC-DC adapter power and a battery pack, three strong candidates are the TSP2121, LTC4412, and LTC4417. Each device offers different strengths depending on system complexity, current demands, and the number of input sources.

The TSP2121 is a compact, integrated dual-channel power multiplexer that requires minimal external components. With its internal MOSFETs capable of handling up to 3 A per channel, it is well suited for portable electronics that need automatic or logic-controlled switching between a battery and wall adapter. Its ease of use and small footprint make it ideal for straightforward two-source applications where board space and simplicity are priorities [155].

In contrast, the LTC4412 offers more flexibility by acting as an ideal diode controller for an external P-channel MOSFET. This allows designers to scale the current capacity by selecting the appropriate MOSFET, making it useful for applications that require higher current handling than integrated solutions can provide. It also features very low quiescent current, which is beneficial for battery-powered systems where standby efficiency is critical. However, it does add complexity since the designer must select and size the external MOSFET correctly [156].

The LTC4417 is a more advanced solution, supporting up to three power sources with built-in prioritization logic. Unlike the TSP2121 and LTC4412, which manage two inputs, the LTC4417 is particularly useful for systems that need to coordinate among multiple supply options such as a wall adapter, USB power, and a battery. It requires external MOSFETs, similar to the LTC4412, but provides far greater flexibility for complex systems like servers, telecom equipment, or industrial devices where redundancy and source prioritization are essential [157].

Table 3.50: Power Switching device Model

Name	TSP2121	LTC4412	LTC4417
Brand	Texas instruments	Analog	Analog
Type	Dual-input power multiplexer (TI)	Ideal diode controller (Linear Tech / ADI)	Triple-supply prioritized power-path controller
Input Voltage range	2.7V – 18V	3V-28V for AC line 2.5V for Battery line	2.5V – 36V
Max Load current	Up to 3A (per channel, integrated FETs)	Depends on external MOSFET (scalable, 100mA–several A)	Depends on external MOSFET
Number of inputs	2	1	3
Power Path Control	Automatic switchover or logic controlled	Drives external MOSFET as ideal diode OR'ing	Automatically prioritizes sources (e.g., wall > USB >

			battery)
Quiescent Current	~35 μ A	~11 μ A	~24 μ A
Reverse current blocking	Yes integrated MOSFET	Yes external MOSFET	Yes external MOSFET

In summary, the TSP2121 excels in simple two-source designs with moderate current needs, the LTC4412 provides scalability for higher currents with external MOSFETs, and the LTC4417 is best suited for multi-source, priority-driven systems. The right choice depends on whether the project values simplicity, scalability, or advanced source management. This project will use the LTC4412 for its low QC, wide voltage range, and reverse current blocking.

3.7.4.6) MOSFET

Since the LTC4412 was chosen as the switching device for the battery and AC-DC lines it needs an external MOSFET to work. Based on how the diagram is set up (figure 3.X) it needs a P-channel MOSFET. This can be seen how the arrow in the middle is going out. This is an external component that will go into the LTC4412 to ensure efficient conduction, minimal voltage drop, and thermal safety at the intended load current. Three suitable options—SIL2305B-TP, NTJS3151PT1G, and AO3413—offer different trade-offs in voltage rating, current handling, and package size.

The SIL2305B-TP provides the highest continuous current margin of the three, rated at 5.4 A with a maximum drain-source voltage of 20 V. Its $R_{DS(on)}$ of approximately 60 m Ω at 4.5 V gate drive ensures a low voltage drop across the MOSFET even at 3 A load, resulting in minimal power dissipation. Its SOT-23-6L package is compact, though care must be taken with PCB thermal design to maintain junction temperature under heavy load. Overall, it is the most robust choice for sustained 3 A operation, balancing low $R_{DS(on)}$ and current capability [158].

The NTJS3151PT1G offers slightly lower on-resistance (~45 m Ω at -4.5 V) compared to the SIL2305B-TP, which translates to marginally lower conduction losses. However, its continuous current rating is only about 2.7A, making it borderline for continuous 3A operation. This MOSFET is best suited for applications where the load current is intermittent or where PCB thermal management can ensure adequate cooling. Its very small SMD footprint is advantageous in space-constrained designs [159].

The AO3413, rated at 3A continuous with a maximum V_{DS} of 20 V, is fully capable of handling the target load, though its $R_{DS(on)}$ is slightly higher than the SIL2305B-TP, leading to somewhat more voltage drop and heat generation at 3 A. Its SOT-23 package allows compact layouts, making it suitable when board space is limited. While not as robust in conduction losses

as the SIL2305B-TP, it provides a good balance of size, availability, and sufficient performance for moderate current switching applications [160].

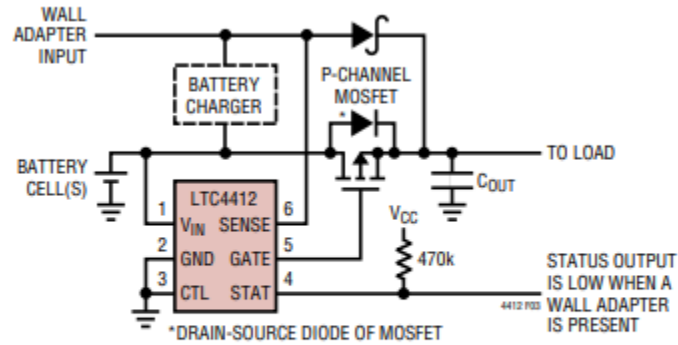


Figure 3. Automatic Switchover of Load Between a Battery and a Wall Adapter in Comparator Mode

Figure 3.16: “LTC4412 Datasheet” (reference datasheet for LTC4412)

Table 3.51: MOSFET Comparison

Name	AO3413	NTZS3151PT1G	SIL2305B-TP
Brand	Omega and Alpha semiconductor	Onsemi	MCC
V _{DS}	20V	12V	20V
ID (typ / steady @ 25°C)	~3 A (continuous, TA conditions per datasheet).	Steady: ~2.7 A (TA=25°C); short-time/pulse to ~3.3	5.4 A (Ta / 25°C spec).
R _{DS(on)} (test V _{GS})	R _{DS(on)} < 80 mΩ @ V _{GS} = ~4.5 V (also specified values for lower V _{GS}).	~45–60 mΩ @ V _{GS} = ~4.5 V (45 mΩ typical @ -4.5 V; 67 mΩ @ -2.5 V	~60 mΩ @ V _{GS} = 4.5 V (60 mΩ @ 2.7 A, 4.5 V listed).
V _{GS(th)} (type)	V _{GS(th)} ≈ -0.4 ... -1.0 V (typ ~ -0.65 V)	~0.4 to 1.2 V (V _{GS(th)})	~0.5 V (typ / device table)
Q _g (type/max)	Q _g ≈ 8.5 – 11 nC	Q _g ≈ 8.6 nC (total at	~15 nC @ 4.5 V

	(datasheet)	–4.5V listed).	(max listed).
--	-------------	----------------	---------------

In summary, the SIL2305B-TP is the best choice for robust, low-loss operation at 3 A with the LTC4412 making this the best choice for this project, while the AO3413 is a solid compromise for compact designs. The NTJS3151PT1G is suitable primarily for short-duration or thermally managed loads, offering the lowest $R_{DS(on)}$ but limited continuous current margin. This makes the SIL2305B-TP the best choice for this project between the three.

We actually found that this choice was an error on our part. The SIL2305B-TP cannot handle the drive voltage put out by the LTC4412 ideal diode switcher since our power board was using 9V and not our regulated voltages of 3.3V and 5.2V. We instead used SIR4409DP-T1-RE3 P channel MOSFETs.

3.8.5.1) Power Regulators

For this project to consider a regulator, there needs to be a discussion between the two types of regulators: Linear vs Switching.

A linear regulator, as its name suggests, uses linear components to regulate the voltage. All linear regulators require an input voltage at least some minimum amount higher than the desired output voltage. That minimum amount is called the dropout voltage. A low-dropout or LDO regulator is a DC linear regulator that can regulate the output voltage even when the supply voltage is very close to the output voltage. Linear regulators are a great choice for powering very low-powered devices or applications where the difference between the input voltage and output voltage is small. They are a simple and inexpensive solution, but linear regulators are normally inefficient because the difference between the input voltage and the regulated output voltage is continually dissipated as heat. Of note, due to this design style, the only supported topology is a buck regulator [161].

The second type of regulator is the switching regulator. These regulators were created due to the need for a wider input voltage range, high efficiency, and low quiescent current (IQ). Switching regulators rapidly switch a series element on and off. They can operate with both synchronous and non-synchronous switches (FETs). These devices store the input energy temporarily and then release that energy to the output at a different voltage level. The switch's duty cycle sets the amount of charge transferred to the load. Switching regulators are efficient because the series element is either fully conducting or fully switched off, so it dissipates almost no power. Unlike linear regulators, switching regulators are able to generate output voltages that are higher than the input voltage or of opposite polarity. Their versatility allows configuration for buck, boost, buck-boost, flyback, and inverting in both isolated and non-isolated applications [161].

Table 3.52: Linear vs Switching Regulator Comparison

Type	Linear	Switching
Designs	LDO	Buck, Boost, Buck-Boost
Efficiency	Normally low to medium-high for a low difference between V_{in} - V_{out}	High
Heat Dissipation	High due to excess being outputted as heat	Low due to inductors
Size	Larger at high voltages	Smaller
V_{in} range	Low due to V_{out} needing to be close to V_{in}	Wide
Cost	Low	Higher due to extra components and IC complexity

After considering both options, the best regulator for use in this project is the switching regulator. This is for the following reasons:

High efficiency – minimizes power lost from the regulator, ensuring all components receive the required power.

Wide input voltage range – ideal for battery-powered designs, since battery voltage changes over time during discharge.

Greater topology flexibility – allows multiple converter types if the project requires them.

Therefore, the best style of regulator for the KIPS MK II is the switching regulator.

3.7.5.2) Switching Regulator Types

For the KIPS MK II one of the things that has to be considered is what type of regulators are needed for this project. These converters will be running off the battery so they will be DC-DC. Each has its own purpose and input and output voltages. So the correct option for this project will be based on what the voltage needs are for the KIPS MK II.

The first type is the Buck converter. This type of converter takes a voltage and steps it down to a lower voltage. This is used when the output voltage needs to be less than the input voltage. The benefit of this is also that this raises the current as the ratio must be maintained. This topology has a high efficiency of 92+ and has low ripple [162].

The second type is a boost converter. This type is where the output voltage is higher than the input voltage. It achieves this by storing energy in an inductor and releasing it at a higher voltage. So it is a step-up converter. This type of converter lowers current however as it raises the voltage [162].

The third type of converter is the Buck-Boost regulator. This type of converter is as its name suggests a combination of a Buck and Boost regulator. Meaning that the converter can either step up or down the output voltage relative to the input voltage based on the conditions needed. The downside of this topology is that it has lower efficiency and also the polarity on some models is opposite to the input (see figure 3.x) . Also to implement this type of converter not only needs more complex components to go alongside it. Best used when the input voltage varies dramatically and there needs to be a steady output voltage [162].

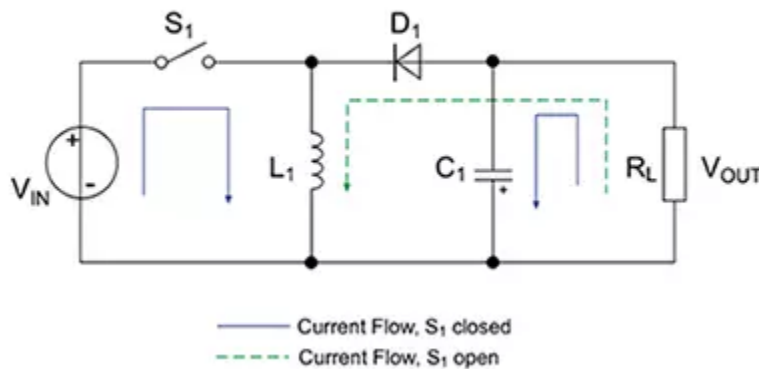


Figure 3.17: “Buck-Boost Converter” (reference recom-power)

Table 3.X: Converter type Comparison

Type	Buck	Boost	Buck-Boost
Function	Reduces voltages	Increase voltage	Either increase or decrease voltage
Output	$V_{out} < V_{in}$	$V_{out} > V_{in}$	$V_{out} < V_{in}$ or $> V_{in}$
Polarity	Same as input	Same as input	Inverting (basic) or

			Non-inverting (advanced)
Efficiency	High 92+%	High 90+%	Moderate 80+%

We decided to go with the Buck Regulator, as both the battery and the AC-DC line will output over 5V. This means that to use those power lines, they will need to be stepped down. This method also has high efficiency, meaning that it will not waste a large amount of power. Making this the best choice for KIPS Mk II.

3.7.5.3) Buck Regulator

When the battery voltage comes out, it is not only at a higher voltage than is needed, but also the current is too low. The setup for this project is two batteries in series, which increase the voltage to 8.4V. As well as the AC-DC line will take in 9V. The way to rectify this is to use a Buck Regulator as discussed above. It will step down the voltage to step up the current on the line. A buck regulator needs less current to run, and this will help decrease the voltage for the system to use. This also helps with the regulation of voltages, too, as the buck regulator can offer stability to the system.

Choosing which buck regulator will be used in this project is based on the requirements above—the TPS566242 from Texas Instruments. The TPS566242 is a single-channel synchronous step-down converter designed for compact, high-efficiency power systems. It can deliver up to 6 A of continuous output current with excellent transient response and low ripple. Operating over an input voltage range of 3 V to 16 V, it is well suited to handle the 8.4 V provided by two lithium-ion batteries in series, making it ideal for battery-driven systems where the supply voltage fluctuates with the charge state and also 9V from wall power. The TPS566242 integrates both high- and low-side MOSFETs and operates at a fixed 600 kHz switching frequency, simplifying design and ensuring predictable performance with minimal external components. Its peak current-mode control provides stable operation across a wide load range, while maintaining high efficiency even at lighter loads. In this design, the regulator is used to step down the 8.4 V battery voltage to 5 V, providing power for the Raspberry Pi and other 5 V subsystems. In addition to its strong electrical performance, the TPS566242 includes integrated soft-start, power-good indication, and comprehensive protection features such as overvoltage, undervoltage, overcurrent, and thermal shutdown. These ensure robust and reliable operation even under varying load and temperature conditions. The device's compact 3 mm × 3 mm VQFN package further reduces PCB footprint and simplifies thermal management, an important factor in compact embedded designs [163].

There are other options in this project that were considered for the buck regulator, those being the LTC3615 and the LM5642. The LTC3615 was another choice that was considered as the

buck regulator for the batteries. This device is also from Analog Devices and has an input range between 2.25 and 5V. It is a dual 3A synchronous p-down regulator so it also has two different output channels that can be used. The main issue with this choice comes in the input voltage range [164]. The battery configuration that has been picked will supply an input voltage of 8.4V. This means that the LTC3615 will not be able to take all of the voltage the battery is supplying for the device, meaning this choice will not work for the usage we need it to. Another three aspects to consider are efficiency, the switching frequency, and the output voltage. The efficiency of the LTC3633 is rated at 95% and the LTC3615 is at 94%. This means the LTC3633 will have greater power conservation when it comes to output, meaning that the LTC3633 is the better choice in this aspect as well. When it comes to the switching frequency between the two, although the LTC3615 has a broader range, the application will be over 500kHz so this is not an applicable difference for this project. Lastly, the output voltage on the LTC3615 is only up to around 5V due to the input voltage being low. This means that it cannot go higher than 5V in any application. Meanwhile, the LTC3633 has a greater output voltage range due to the greater input voltage range. This means that it is more applicable in more designs due to being able to handle low and moderate loads, which our device needs.

The other option up for consideration was the LM5642. This device is from Texas Instruments and has two synchronous buck regulator controllers. This is an important distinction from the rest of the components shown. The LTC3633 has MOSFETs built in, while the LM5642 does not. Meaning that for this component to work, there will have to be external MOSFETs built onto the board. This is unnecessary and therefore is not suitable for this project. Another key aspect is how the output voltage range exceeds what is required and, therefore, can interfere with normal operations if left unchecked. Not only that, but it has vastly less efficiency at 90% than the LTC3633, which will increase power consumption unnecessarily. Also, the switching frequency is set at either 200 kHz or 375 kHz, which can cause issues if that needs to ever be changed [165].

Table 3.53: Buck Regulator Comparison

Name	TPS566242	LTC3615	LM5642
Brand	Texas Instruments	Analog Devices	Texas Instruments
Input Voltage Range	3V-16V	2.25V-5V	4.5-36V
Reference voltage	V _{in}	V _{in}	V _{in}
Output voltage	0.6V-7V	0.6V-~5V	2.9-38V
Max Current	6A	3A	3A

Price	0.77	4.72	5.63
Efficiency	95%	94%	90%
Switching Frequency	600kHz fixed	200kHz to 4MHz (External resistor)	200 kHz (Fixed) for LM5642 375 kHz (Fixed) for LM5642x

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 and 3.3V rail for system operation.

3.8) Communication Protocols

Now that all the components have been decided, it must be determined how each of these components will communicate with others. In this section, we explore the communication protocols of both internal and external connections. For the internal communication protocols, we look at each of the components that communicate with another component inside the device. For the external communication protocols, we examine how the device communicates with peripheral devices such as video output and USB.

3.8.1) Internal Communication Protocols (part to part)

Ensuring internal components can communicate with each other, they will be primarily operating on I²C communication protocol. An exception to this is the heart rate sensor, it will output a voltage equivalent to the PPG signal that is received, which will be processed by the MCU to be displayed to the user. The reason behind this is because it makes overall communication easier by having the protocol be the exact same. The reason the heart monitor is different is because we were unable to find a sensor that meets the requirements while also using I²C and was not part of a development board or other device.

The SBC is a little more versatile in the sense that it can communicate with other components using UART, SPI, and I²C. As previously mentioned, most of the parts present within the project communicate through I²C protocol, however the functionality of the SBC should not be overlooked.

3.8.1.1) I²C Communication

The Raspberry Pi Camera Module 3 utilizes the I²C (Inter-Integrated Circuit) interface primarily for configuration and control of the sensor. Through I²C, a host processor—such as a Raspberry Pi or microcontroller—can adjust key camera settings, including exposure, gain, frame rate, and

operational modes. While the sensor transmits image or video data over a separate parallel interface (1-bit, 4-bit, or 8-bit), the I²C bus remains essential for initiating captures, switching modes, and fine-tuning the sensor parameters in real-time. This separation allows for low-complexity wiring, minimizes data lines required for control, and ensures that the module can operate efficiently in low-power applications without burdening the main processor with continuous configuration tasks. The Camera Module 3's I²C interface is fully compatible with standard I²C voltage levels (typically 3.3 V), making it straightforward to integrate into most development boards while maintaining robust control and flexibility over the camera's operation [70], [166].

3.8.1.2) Analog Communication

The heart sensor specifically works by shining an LED's light into the skin. Depending on the level of absorption, a photodiode or phototransistor will see these changes in light and emit a tiny current in response. These tiny currents will track these light changes. If more light is absorbed, the output current will be smaller, therefore V_{out} will be smaller. If less light is absorbed, the output current will be bigger, and therefore a larger V_{out} result. The current from the photodiode/phototransistor will be converted into a voltage by a transimpedance amplifier. This is then passed through a low pass filter to remove high frequency noise, and a high pass filter to remove low frequency noise and drift. The result is a voltage line graph with a smooth rise and fall, whose peaks will align with the heartbeat itself. This line graph is then converted to a digital format by subtracting a moving average. Next, BPM is detected by finding the peaks along the line, and computing the time between beats, dividing 60 by the time between beats, and this computed result will be displayed as the user's BPM.

Another component that communicates using analog data is the joystick. As explained in the joystick section, output from the joystick is dependent on the position of two potentiometers at the base of the device. Based upon the position of the potentiometers, the connected component (the MCU in this case) receives a continuous stream of analog information depicting the position of each potentiometer.

3.8.1.3) Other Communication Types

For the internal display, the SBC will be connected to the screen through DSI. DSI is designed specifically for high-bandwidth video which is required to display high-resolution output to the screen. UART and I²C are far too slow in this case and routing the video data through them would cause significant latency and display delays. DSI allows for high-speed differential data lanes to transmit pixel data from the SBC to the display, ensuring smooth frame rates and highly responsive graphics on the display. By utilizing DSI, the SBC can reliably handle the full bandwidth required of the display while also reducing delays, allowing the user an immersive experience.

In order to pass data from the sensors, joystick, d-pad, etc. from the MCU to the SBC we need to implement a UART connection. While I²C and SPI are both available for this connection it is purely secondary, meaning that they cannot replicate the entirety of the functionality of UART on their own. This also means they would only be used in the case that UART is unavailable for the required communication. UART is universally supported between the SBC and the MCU as they both have hardware dedicated to UART communication. UART is also needed for OS support; Linux and its related operating systems treat UART like a serial port and are able to read information from the MCU directly, like a file stream. The alternate communication protocols offered such as I²C and SPI are either much slower or require much more hardware in order to create a lasting connection. In this case, UART is the simplest way for the MCU to communicate information to the SBC both from an operating system perspective and in general.

3.8.2) External Communication Protocols (part to peripheral)

Communication with external sources and devices is crucial for K.I.P.S.'s functionality, allowing it to receive input from peripheral devices such as mice and keyboards as well as extending video output to connected monitors and screens.

3.8.2.1) Video Communication

For this project, we intend to have a video output port. This will allow KIPS MK II to have the ability to send out its video signal elsewhere. The only challenge with this is considering what standard of video output signal to use for the project. What has to be considered is not only the quality of the images provided, but also other features that may or may not be included in all of the figures.

The first standard to consider is HDMI. This type of connector offers bandwidth of 18 Gbps up to around 96 Gbps depending on what standard of HDMI 2.0 is being looked at. HDMI has 19 pins. It is the default interface for televisions, projectors, game consoles, and home theater systems. HDMI 2.1 supports up to 8K at 60Hz, 4K at 120Hz, and full 16-bit “Deep Color” with HDR formats like HDR10 and Dolby Vision. It also integrates features such as Audio Return Channel (ARC/eARC) for simplifying home theater setups and Consumer Electronics Control (CEC) for device coordination. Unlike USB-C and DisplayPort, HDMI does not support data transfer or significant power delivery — it is a dedicated audio/video interface. Its ubiquity in entertainment devices makes it the most consumer-friendly option, but it lacks the flexibility of USB-C or the extreme performance of DisplayPort [167], [168].

The second standard is DisplayPort. It is engineered for maximum display performance, with DisplayPort 2.1 supporting up to 16K resolution at 60Hz, refresh rates of 240Hz at 4K, and advanced features like Adaptive Sync for smooth gaming performance. DisplayPort also supports Multi-Stream Transport (MST), which allows multiple monitors to be daisy-chained from a single port — this is something the other two options do not have. It delivers the same

deep color support (up to 48-bit, HDR10) as HDMI and USB-C, and with Display Stream Compression (DSC), it can transmit extremely high-resolution signals without visible loss. Unlike USB-C, DisplayPort does not carry power, and unlike HDMI, it is less common in consumer devices, focusing instead on professional and high-end PC use cases. DisplayPort also has 20 pins and a locking mechanism, which is something unique to it as well [167], [169], [170].

The third standard that was considered was DisplayPort over USB-C. What is interesting about this option is that it can deliver power to accessories that are connected to the port. Through DisplayPort Alt Mode, it can transmit high-resolution video (up to 8K and beyond), support refresh rates up to 120Hz at 4K, and deliver full 48-bit color depth. Unlike HDMI and DisplayPort, it also integrates high-speed data transfer (up to 40Gbps). The connector is small at around half the size of the HDMI or DisplayPort connector. This option is best when power and data connection are needed [171].

Table 3.54: Output Types

	HDMI	USB-C	DisplayPort
Max Resolution / Bandwidth	HDMI 2.1: up to 48 Gbps (8K @ 60 Hz (HDMI 2.1))	Up to 8K at 60Hz via DP Alt Mode;	32.4Gbps DisplayPort(1.4)
Data transfer	N/A	~40Gbps	N/A
Power delivery	N/A	Up to ~240W	N/A
Connector design	Larger, not reversible	Small, reversible, universal	Larger, not reversible
Color Depth	Up to 48-bit (16 bits per channel) with HDMI 2.1	Up to 48-bit (16 bits per channel) with DP Alt Mode /	Up to 48-bit (16 bits per channel)

Taking all three options into account, the best option for what this project needs is HDMI. Not only does it offer high performance at up to 96 Gbps with HDMI 2.2, but it also has HDR support should that be a feature that is implemented. When thinking about all of the possible connections the KIPS MK II will have, the output signal will be mainly to outside display devices and not much more, so the most standard connection for display devices is the best option for this task. Therefore, power nor a locking mechanism is needed. This is a dedicated display option that was chosen due to all of these factors but also because HDMI is directly compatible with the SBC chosen, making HDMI the best choice for the KIPS MK II.

3.8.2.1.1) HDMI

Upon doing research on video input components, we decided that the best approach was without a doubt to use an HDMI (High-Definition Multimedia Interface) port. More specifically, we chose a mini HDMI option, as regular HDMI ports would require more space, meaning that it would be a disadvantage to our goal in ensuring our model remains at a respectable size. Choosing HDMI also was for the sake of quality, as our SBC (Single Board Computer) has the capabilities of acquiring Quad-High Definition resolution. And making certain that any consumer has the ability to obtain such quality, using connectors such as a VGA or RCA was never a preference [167], [172].

Now when it comes to HDMI the main concern is due to size. Based on the table below all HDMI types all have the same number of pins that being 19 what this means is that all of the types of HDMI can transmit the same amount of data at the same speed. Therefore no matter what size the port is, the amount of data transferred and the quality of the signal remain the same. What it then comes down to when it comes to HDMI is what connector size will be used for this project. The standard HDMI size is 13.9mm x 4.45mm, mini is 10.42mm x 2.42mm and micro is 6.4× 2.8 mm. What it then comes down to is which one will be the most common for what applications this will be used for. The main usage of the HDMI port will be to broadcast information from the device onto another screen. This means that it is the source for all of the data going outward [172].

Table 3.55: HDMI Type Comparison

HDMI	Standard	Mini	Micro
Connector size	13.9mm x 4.45mm	10.42mm x 2.42mm.	6.4× 2.8 mm
Pin Count	19	19	19
Use Case	The majority of audio-visual equipment, like TVs, computer monitors, game consoles, etc	small portable devices like tablets, camcorders, and DSLR cameras.	smaller portable devices like smartphones.

To that end, the best choice to use will be to use the standard HDMI size. Not only is this the most commonly accepted type of connection for HDMI but for the use case that the HDMI port on the device is used . that being projection, it is the most fitting for our purposes. Even though it is a portable device the main aspect of this device is to make it like a portable computer.

3.8.2.2) USB 2.0 vs USB 3.0

When designing this project one of the aspects that have to be considered is what standard of USB should the KIPS MK II support. USB is the standard for connecting various outside components of what is built on the device to that end. We require USB 2.0 connectivity at a minimum in order to connect peripheral devices such as keyboards, mice, and USB drives for peripheral interaction with the device and for additional storage. Alongside USB 2.0 connectivity, we also require HDMI support in order to extend the device's display to external sources or screens. So then comes the issue of what standard has to be considered between USB 3.0 and USB 2.0. USB functionality allows the user to be able to do more such as: add more data storage, have another low power charging option, and potentially connect to additional peripherals like a keyboard.

USB 2.0 was introduced back in 2000 while USB 3.0 was introduced in 2008 [173]. What has to be considered is which one of these two standards should be chosen in this project. The difference between choosing one over the other is a series of trade offs and compromises that have to be considered. The main difference between the two standards is data speed and power output. USB 2.0 outputs around 480 Mbps while USB 3.0 outputs over 10 times that at 5 Gbps. They both output the same power output voltage the main difference is current with USB 3.0 having 900mA which is twice the speed of USB 2.0's 450mA. Overall after considering these trade offs the one that is to be chosen for this project is USB 2.0. USB 2.0 is used on more devices and is more commonly accepted between the two standards but this is not the only reason why USB 2.0 is preferred in this project. Another factor to consider is cross compatibility. What happens when a USB 2.0 device and a USB 3.0 device connect. Thankfully USB 3.0 has the ability of backwards compatibility. What that means is that when the devices are connected the two devices will default to a standard when they are connected so they can work together. This standard is USB 2.0, meaning if you only use USB 2.0 devices then you will not notice the difference between USB 2.0 and USB 3.0 [173]. This is more than enough for all of the potential needs for the project peripherals. Another aspect is cable length with USB 2.0 having more length at 5 meters. Meaning USB 2.0 is preferred for longer cables.

We specifically chose utilizing USB 2.0 over USB 3.0 due to compatibility reasons and signal integrity. Signal integrity especially played a major role in as to why USB 2.0 was chosen for our design, as this would affect the complexity of the PCB design. USB 2.0 is simpler to implement at only 5 pins.

Table 3.56: USB Type Comparison

	USB 3.0	USB 2.0
Signal Integrity	~5 Gbps	~480 Mbps
Recommended Cable Length	3 Meters (~9 Feet)	5 Meters (~15 Feet)
Compatibility	Less Range of Devices	Much Broader Range of Devices
Power Output	5V @ Up to 900mA	5V @ Up to 450mA
Size	USB A connection	USB A connection

Meaning to implement on the PCB it is best to choose USB 2.0 over 3.0. So after taking into account the tradeoffs between USB 3.0 and 2.0, USB 2.0 is the best choice to use for KIPS Mk II.

3.9) Software

For this project, a proper development environment is required to produce the finalized software and operating system for the device to function. A number of different things are required to create this proper development environment, including the operating system, programming languages for both operating system-level programming and embedded-level programming, as well as the IDE and version control software. In this section, we explore the different aspects of the development environment and which available options are the best fit for this project.

3.9.1) Operating System

Raspberry Pi OS

Raspberry Pi OS is the natively supported operating system (OS) of the Raspberry Pi line of products. Raspberry Pi OS originated from the community-maintained, open source OS Raspbian, which was tailored specifically for the Raspberry Pi's ARM processors. Now, Raspbian has been adopted officially by Raspberry Pi and absorbed into Raspberry Pi OS as the 32-bit distribution. Raspberry Pi OS comes in 32-bit and 64-bit distributions based on the ARM architecture of the board. At its core, Raspberry Pi OS is Debian-based which means it inherits the Debian package management system (APT) and numerous different software repositories. Being the default operating system of the Raspberry Pi line of products, Raspberry Pi OS is optimized for Raspberry Pi hardware, complete with preconfigured drivers for Wi-Fi, Bluetooth, GPIO and the camera as well. Despite the extensive features offered by Raspberry Pi OS, it is still fairly lightweight, which is crucial for maximizing available memory. Another defining characteristic of the Raspberry Pi OS is that it is Free use and Open Source Software (FOSS)

barring a small portion of proprietary firmware, drivers, and bootloaders which would not need to be modified for this project anyway. This means that for this project we could modify and/or rebuild practically the whole operating system to better fit the needs and constraints of the device [174].

DietPi

Another operating system available is DietPi. DietPi is an extremely lightweight Debian-based operating system. DietPi is optimized for minimal resource utilization, meaning that it strips away a lot of the bloat associated with other operating systems such as Raspberry Pi OS or Ubuntu in order to greatly reduce the amount of RAM and disk space required. Additionally, DietPi can be further customized in order to include only specific software distributions, maintaining a minimal and efficient system which is oriented exclusively towards the needs of the device. This operating system is well suited for running on Raspberry Pi as well as other ARM SBCs and is preconfigured for low-resource environments. DietPi runs with minimal overhead, prioritizing performance. One benefit of DietPi is its automated software installer which provides a menu-driven UI to quickly install and configure various different popular software. Unfortunately, with this auto-installer, some more advanced or niche libraries for certain sensors may not be available. Additionally, DietPi is geared towards single-purpose devices as opposed to multimodal devices like K.I.P.S. which requires desktop and mobile modes in addition to a functioning GUI [175].

Ubuntu Core

The last operating system we consider is Ubuntu, which is another Debian-based distribution. Although there are three main flavors of Ubuntu (desktop, server, and core) the best fit for this project is Ubuntu core for its lightweight RAM and CPU usage as well as the inclusion of a graphical user interface (GUI) in comparison to the other flavors which have much heavier RAM and CPU usage in addition to the fact that neither of them include a GUI which is essential for a device such as this. Ubuntu core is oriented towards usage in embedded devices in the sense that it is lightweight and secure which aligns well with the goals/features of the device. This flavor of Ubuntu utilizes snap packages instead of apt-based installs which provides easier updates and better security than apt-based package management in comparison. Ubuntu as a whole releases scheduled software updates regularly, every six months with additional updates in between as needed for regular bug-fixes and security updates. This ensures that the most recent software is made available in a timely fashion and makes it easy to anticipate OS updates. One thing to consider about Ubuntu core is that it utilizes more resources than comparable, lightweight distributions [176].

Table 3.58: Onboard Operating System Comparison

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

Selection

Comparatively, each of these operating systems present great strengths but each also have various weaknesses. Of the three options, Ubuntu core uses more resources and has a smaller package ecosystem in comparison. Ubuntu core also may be slightly less flexible or faster for real-time tasks compared to Raspberry Pi OS or DietPi. For this reason Ubuntu core can be ruled out. When viewing Raspberry Pi OS and DietPi together, there are two glaring differences. DietPi is highly minimalist, only including exactly what is required for the device to run and not much else, whereas Raspberry Pi OS includes other necessary packages but also other software and drivers that are commonly used in conjunction with Raspberry Pi products. In this use case the additional space utilized by Raspberry Pi OS is excusable considering that the majority of it will be utilized in this project. The other glaring discrepancy between the two operating systems is functionality. DietPi is only really suited to serve a single-purpose device such as a network service or media server, whereas Raspberry Pi OS is geared towards multimodal devices. Considering that K.I.P.S. with its dual modality (desktop and mobile) alone are enough to classify it as a multimodal device, Raspberry Pi OS is the better fit. Overall, the best option for this project is Raspberry Pi OS due to its great support for Raspberry Pi devices and its flexibility in supporting multimodal functionality.

3.9.2) Programming Languages

Python

Python is a very high-level, object-oriented programming language that is easily readable and is great for introductory programming and advanced programming projects as well. Python is generally used for AI and machine learning, web development and automation. The Python language has an extensive number of libraries available for practically any use case. Python's massive ecosystem of libraries would be very useful for use in this project as there are existing libraries for sensor communication through UART, SPI, and I²C communication as well as for GPIO control and camera integration. Additionally, these libraries also make for easier communication with the ESP32. Python's modularity also makes it quite simple to integrate new hardware components and features as the project evolves. Python's interpreted nature means that testing portions of code is much faster as compilation is not needed and instead code testing can be done much faster. One thing to note though is that Python runs much slower than compiled languages in applications, real-time readings (like reading from sensors) and in CPU-intensive tasks which may cause issues with response time for our device. Additionally, Python does not allow nearly as much control over memory as other programming languages which could prove problematic as efficiently managed memory is paramount in this project [177], [178].

C++

C++ is a procedural, object-oriented language that is best known for its memory management. C++ is a high-level language that allows for an immense amount of control over memory through the usage of pointers as well as manual allocation and deallocation of memory. C++ is commonly used in system software, games and other high-performance applications. Because C++ is a compiled language testing code may take longer, however the runtime in the context of applications for the device would be much faster in comparison to python and other interpreted languages. This is also very important in real-time applications such as polling sensors and communication with peripheral devices. This language also supports modular design, which is highly useful in the differentiation of desktop and mobile mode features such as sensor handling. The manual memory management offered by C++ is a great fit for this project as it allows for meticulous control over RAM usage, helping ensure efficient performance on a wearable, resource-constrained device. C++ also has several libraries for graphics, hardware communication, and real-time scheduling which makes handling sensors, GPIO, and other hardware interactions much easier. One very important thing to note is that due to the lower-level, resource-constrained nature of the ESP32 of the languages included in this comparison, C or C++ are required in order to control the hardware directly and achieve decent performance. Unfortunately, C++ has a steeper learning curve and takes longer to develop within because of its complexity and lower-level syntax, meaning that more time may be required to program the required software for the device [179].

Rust

Rust is a fairly recent programming language best known for its safety and performance. Rust is utilized in software with a focus on security. Like C++ and C, Rust compiles to native machine code. Like C++, Rust also includes memory management techniques, however the implementation of these techniques is where C++ and Rust differ. Rust has no garbage collector and does not require manual memory allocation, instead it handles memory management automatically and is enforced by the language rules at compile time. Rust has a fairly quick execution time which is comparable to C++ and is well suited for high-performance applications. Rust is able to prevent some bugs common to other programming languages such as race conditions, memory leaks, and floating pointer exceptions at compile time. This would benefit the device by making it more stable and would reduce the risk of it crashing. As previously stated, Rust is a relatively new programming language, meaning that it has a much smaller ecosystem compared to older programming languages like C++ and Python where support is much more extensive. Rust is also very difficult to learn, with arguably the steepest learning curve of all the languages included in this comparison. Rust was actually used in the previous iteration of K.I.P.S. and although it was utilized effectively and ran well, there was an egregiously steep learning curve [180], [181].

Arduino

The Arduino programming language is a simplified derivative of C/C++. The Arduino programming language provides high-level functions for interacting with microcontrollers in an effort to make programming hardware more accessible. Arduino is fairly easy to grasp and is fully compatible with C/C++. The abstraction associated with the high-level functions used to make hardware communication more straightforward comes with the drawback that it makes the hardware details like registers and memory addresses inaccessible. Like C/C++, dynamic memory management is also available which could prove very helpful in efficient and sparing use of memory. While Arduino is based on C/C++, it does not support the full breadth of features and standard libraries available in C/C++. Arduino's programming language is primarily suited for their own products and connected devices, however, it can also be used in external devices as well. Arduino also has its own IDE, which restricts code development to the Arduino environment. This limits access to some of the debugging tools and advanced features which are made available in other programming environments. Also unlike C/C++, Arduino's native programming language is not suited for use in complex or operating system level applications. Arduino cannot run conventional software that requires operating system features. This eliminates Arduino from contention in the OS-level programming language selection [182], [183].

Table 3.59: Programming Languages Comparison

	Python	C++	Rust	Arduino
Compatibility with ESP32	Limited (MicroPython)	Yes	Yes	Possible to
Memory Management	Automatic (garbage collection and limited control)	Manual (pointers, allocation and delocation)	Ownership model (safe, enforced at compile time)	Dynamic Memory supported but abstracted
Execution Speed	Slow	Fast (compiled)	Fast	Fast
Real-time capability	Poor	Strong	Strong	Moderate
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
Testing/Debugging Speed	Fast (interpreted)	Slower (requires compilation)	Slower (requires compilation, strict rules)	Restricted (Arduino IDE only, fewer tools)

OS-level Programming Language Selection

Overall, each of these programming languages present many qualities that would make them solid contenders for the overall programming language to be used in this project. Python is a great, high-level programming language that would make development and debugging a very simple process at the cost of execution when the device is in use. In contrast, C++ and Rust are a steeper learning curve, but offer memory management and faster execution time in context. For this project, we prioritize the execution time of the device rather than reduced time debugging, detracting some of the draw towards Python but not eliminating it entirely. As mentioned before, the previous iteration of K.I.P.S. used Rust for the underlying firmware and core functionality of the project. While the device ran well, the learning curve was much too steep and as a result, programming the device took much longer than expected. Considering that this project is to be completed in a finite timeframe and learning Rust well enough to utilize it effectively in this

project would take a significant amount of time, we find it best to eliminate Rust from contention. With this in mind, the best choice of programming language for this project is C++ for its efficient memory management, moderate learning curve and strong real-time capability.

Although C++ was initially selected as the best overall programming language for OS-level tasks, the team later pivoted to Python due to project time limitations and existing team familiarity. Since Python had already been used throughout development for testing, front-end hardware communication, and interface prototyping, continuing with it allowed us to make faster progress without spending additional time restructuring the software in C++. During testing, the expected performance drawbacks of Python were negligible for the specific real-time requirements of K.I.P.S.. Sensor readings, user interface updates, and control responses operated with minimal delay on the front end. As a result, Python became the more practical choice for the final implementation, balancing acceptable runtime performance with faster development, easier debugging, and greater confidence in completing the project within the available timeframe.

Embedded Programming Language Selection

One major thing to consider is the fact that the ESP32 is much more constrained in terms of resources so it has a limited selection of programming languages that can be used. The most commonly used languages for the ESP32 are C/C++, Arduino (which is C/C++ based), micropython, Rust and javascript. Although javascript was not mentioned in the comparison earlier, it can easily be ruled out due to the fact that it only supports real-time applications in a limited capacity, it does not allow for explicit memory management, has very limited libraries and is highly inefficient in terms of performance. MicroPython is a lightweight derivation of python that is small enough to be programmed to the ESP32, however, this is not the Python included in this comparison. MicroPython is much more lightweight than standard python and is much less memory-efficient than C and C++, making it not ideal for any real-time tasks. Arduino is a good option for programming to the ESP32, however, it causes restrictions with the programming environment and limits access to low-level hardware features compared to using native C/C++. Rust is a good option for its high performance on the ESP32 and high memory efficiency as well. As explained in the previous section though, Rust is the most difficult to learn of those mentioned in this comparison. Rust also has a much smaller embedded ecosystem than C/C++ and MicroPython which have several libraries and resources to specifically support the ESP32 and ESP32 peripherals. In the interest of development time and available resources, we can eliminate Rust as an option for the overall embedded programming language. C/C++ is a great option based upon its complex memory management capability as well as its real-time capabilities in order to communicate with connected peripherals and sensors. C/C++ is also a great choice in terms of performance, with a relatively quick execution time due to its compiled nature. MicroPython falls behind in this regard as an interpreted language. For testing, MicroPython may be faster, however for regular use, C/C++ are a much better option in terms of

overall performance. Considering all these factors, we select C/C++ for the embedded programming language as well due to its efficient and extensive memory management, comprehensive libraries for supporting the ESP32 and the high performance.

3.9.3) Version Control Software

In a project with multiple team members working on the same code at once, implementing reliable version control software is paramount. Ensuring that edits can be made simultaneously while also preserving past changes when needed is necessary to allow forward progress without detracting from the work already done. Version control software also would allow for changes to be reversed in the case that they are found to introduce bugs or cause the code to behave unexpectedly. We explore a few different popular options for usage in this project below.

GitHub

GitHub is a free cloud-based version-control platform for storing, sharing, and collaborating with others to develop code. In terms of usage, GitHub is by far the most popular, seeing usage by over 150 million plus developers in just 2025 alone [184]. Within the context of this project as well, we are all the most familiar with GitHub for version control due to extensive use in previous courses and projects. GitHub utilizes Git which is a version control system that is able to track changes in files in an intelligent manner. GitHub repositories can either be private or public depending on the use case, allowing privacy for more restricted access applications while also allowing for fully open source access to other applications as well [185]. GitHub would be a great addition to this project primarily because of our group's familiarity with it, and for its ease of use and powerful collaboration features. Additionally, GitHub's extensive documentation makes it much easier to find helpful resources and troubleshoot issues through the development process.

GitLab

GitLab is another Git-based platform that is free to install. Unlike the other options presented in this comparison, GitLab is actually open-source. This distinction means that there is more control available over the environment. There are also two different hosting configurations for Gitlab which are GitLab's native cloud hosting or self-hosting on a personal server or cloud virtual machine [186]. This flexibility is great for enterprise-level deployments, workflow management and for personal projects as well. For this project, cloud hosting would be preferred for ease of use and minimal maintenance. Compared to GitHub, GitLab has a much smaller user base and fewer publicly available repositories. This could introduce issues in the future with finding some example projects or reusable code libraries, however considering the familiarity of the team with Git workflows, this limitation becomes more manageable.

Bitbucket

Another alternative option for version control software is Atlassian's Bitbucket. Bitbucket is another Git based code hosting tool which is designed for team collaboration [187]. Bitbucket has built-in compatibility with Jira and Trello, simplifying the project management process. This integration would make Gantt charts much easier to track for the software portion of this project. This also includes the integration of Trello which is another software used to quickly and efficiently organize tasks. The integration of both of these project management technologies builds a great case in the task-management aspect for Bitbucket. Bitbucket also presents two different hosting options, either through Atlassian's cloud-storage or in their data for different use cases. In comparison to the other two options though, Bitbucket experiences the lowest use rate. The free tier of bitbucket only supports 5 users. This user requirement is sustainable for this project as the total number of group members does not exceed 5, however if the scope of this project was to be scaled up later down the line, this limitation would not be ideal. Bitbucket is also rarely used for open-source software because of its smaller community and limited ecosystem limitations.

Table 3.60: Version Control Software Comparison

	GitHub	GitLab	Bitbucket
Core System	Git-based	Git-based	Git-based
Hosting Options	Cloud or self-hosting via GitHub enterprise server	Cloud or full self-hosting on a personal server/VM	Cloud or self-hosted (server/data center)
User Base	Largest	Midline	Smallest
Project Management Tools	Basic Project boards, integrations with outside tools	Built-in issue tracking and agile planning features	Strong integration with Jira and Trello (project management softwares)
Open-Source Availability	Proprietary (with some exposed APIs)	Fully open-source	Proprietary
User Limit (Free tier)	Unlimited	5 for top-level private groups, unlimited otherwise	5

Each of these three options present various individual benefits that make them all good candidates for use in this project. In the end, we decided upon GitHub as our version control

software platform for its widespread use, team familiarity, no-cost, and privacy options. The main differentiating factor in this decision was team familiarity, as the learning curve for some of these platforms can be quite high, however with prior experience, navigating them can be done with ease.

3.9.4) Integrated Development Environment (IDE):

In order to effectively write and develop code for this project, our developers require an integrated development environment (IDE). This allows developers to write and test code in a unified and efficient environment. Picking the right IDE is incredibly important as it has a direct impact on the compatibility, productivity and ease of debugging for this project. Popular environments like Visual Studio Code, Arduino IDE, and Notepad++ are explored in this comparison.

Arduino

Arduino IDE is an IDE created by Arduino as a user-friendly way to communicate with their hardware in an all-in-one approach. Arduino IDE is fairly easy to use and has some minimal built-in code autocompletion assistance [188]. This is a great editor for beginners for its wide support across various hardware platforms and its simplicity. Arduino IDE simplifies the process of importing and managing external libraries through its Library Manager, which allows users to search, install, and update libraries easily. Arduino IDE also now offers the synching of sketches (projects) in their cloud, which is useful for coordinating development across multiple devices. This could also prove useful if a file is lost locally, it can then be recovered from Arduino's cloud. The ESP32 is not natively supported by Arduino IDE initially, however this can be resolved by installing the ESP32 board definitions via the Arduino IDE's Boards Manager. Through this process, the ESP32 can then be accessed and programmed via Arduino IDE. One of the drawbacks of Arduino IDE is that it does not have built-in Git integration, meaning that commits, pushes and pulls cannot be made directly from the Arduino IDE, in this case it would need to be handled externally through either the command line or another interface. Arduino IDE has limited debugging capability, providing only a basic serial monitor for hardware output and requires additional, external tools for breakpoints or step-through debugging. Arduino IDE is a great choice for quick C++ MCU prototyping and smaller embedded projects but may not be ideal for larger-scale performance-critical projects [183], [189].

Notepad++

Notepad++ is a free, extremely lightweight text editor based in Microsoft Windows. Notepad++ supports 90 plus different languages and is well known for its small footprint and fast performance. Notepad++ offers some code completion assistance in certain contexts which could prove useful in this project for writing and organizing source code efficiently without added overhead [190]. Notepad++ unfortunately has no debugging features and very limited plugins. Notepad++ does not offer any native support for the ESP32 and lacks the toolchains or

frameworks required for embedded development. The reason we even consider Notepad++ despite its shortcomings in practically every aspect is because it is a good baseline IDE to compare against, with all of the basic features required to write a functioning program. Notepad++ was a great starting point for each member of the group as we first started programming and although it has a limited capacity it serves as a solid start for simple, small-scale projects.

Visual Studio Code

Visual Studio Code (VS Code) is a lightweight, yet extensible IDE released by Microsoft. VS Code supports several different programming languages including Python, C++, Rust, etc. and many more through different imports and extensions [191]. One edge VS Code has over its competition is that it has built-in support for git-based version control systems. VS Code also has a fairly extensive debugging framework and natively supports its own AI coding assistant to help fix syntax issues, generate complex lines of code and functions. Of the IDEs mentioned in this section, VS Code arguably has the most extensive coding assistant, being able to understand context across different files and generate working, applicable lines of code [192]. VS Code at its base is fairly light, but supports a multitude of extensions and libraries to extend its use into different domains allowing for all the functionality that the user desires without the bloat associated with unnecessary imports. VS Code is also the most familiar IDE among the members of this group, seeing use in previous courses and projects which minimizes the learning curve required in order to use it effectively. VS Code is not able to natively support the ESP32, however, there are multiple extensions that can be used in conjunction with VS Code to interact with the ESP32. In order to communicate with the ESP32, either PlatformIO or ESP-IDF are required because of toolchain and build system requirements.

PlatformIO is an open-source extension in VS Code for embedded development. PlatformIO enables building, compiling, and uploading firmware to microcontrollers like the ESP32 [193]. PlatformIO also allows the user to manage dependencies and libraries automatically. In addition, PlatformIO easily integrates with Git-based version control software. PlatformIO is incredibly simple to set up in VS Code and supports multiple microcontroller frameworks, including ESP-IDF and Arduino. Some of the limitations of PlatformIO is that there is less direct control over toolchains compared to using ESP-IDF alone and that it may lag behind the ESP-IDF in support for newer features as it is not officially supported by the ESP32 line.

The Espressif Internet of Things (IoT) Development Framework (ESP-IDF) is the official development framework for the ESP32 line of microcontrollers. The ESP-IDF offers low-level access to all ESP32 hardware including GPIO, Bluetooth, timers, interrupts, WiFi and peripherals [194]. ESP-IDF is open source and is freely available on github. It is also fairly stable and optimized for applications in which performance is paramount. The ESP-IDF uses primarily C/C++ in order to provide low-level control of the hardware and real-time tasks in addition to

high-performance. ESP-IDF is also available as an extension on VS Code and can be used via command-line tools or other editors. Some disadvantages of ESP-IDF are that it is a steeper learning curve than PlatformIO or Arduino as a whole and there is more manual setup required.

Table 3.61: Embedded Tools and Frameworks Comparison for Visual Studio Code

	PlatformIO	ESP-IDF
Control over ESP32	Native level for ESP-IDF framework	Full control possible
Real-time functionality	Depends on the frame work (with ESP-IDF, yes)	yes
Debugging	Integrated in VS Code; easy to use	Supported in VS Code but requires setup
Version Control Integration	Easy git integration via VS Code	Git integration available but a more manual approach
Programming Languages	C/C++	C/C++

The best Visual Studio code configuration for this project would be to utilize both PlatformIO and ESP-IDF in conjunction to smoothly communicate with the ESP32 and streamline development. This approach would pair together the benefits of each approach such as the low-level control of ESP-IDF and ease of use of PlatformIO and could potentially solve some of the shortcomings of both approaches as well.

Table 3.62: Integrated Development Environment (IDE) Comparison

	Arduino IDE	VS Code (with PlatformIO or ESP-IDF)	Notepad++
Ease of use	Beginner-friendly	Moderate; some setup required	Very easy for editing
ESP32 Support	Yes (via board manager)	Yes (via PlatformIO and/or ESP-IDF)	No native support; requires external build and flash tools
Code Assistance	Basic Autocomplete	Advanced debugging and code navigation	Syntax highlighting only

Debugging	Limited	Full debugging; breakpoints, watch variables, etc.	None
Build/Flash support	Built-in for Arduino framework	Supports Arduino framework and ESP-IDF natively	Needs external command line and other tools
Git Integration	Minimal; requires external tools or plugins	Very good; best listed	None natively, relies on external tools

Overall, the best option for this project is VS Code in conjunction with PlatformIo and ESP-IDF for its familiarity with the team, compatibility with the ESP32, extensive Git integration, and intuitive debugging framework. In addition, the inclusion of the AI assistant to provide insight in times where the programming team may have made a slight syntax error or there is a compilation error could prove quite helpful. While the other options are well suited for compatibility with this project, they simply are unable to match the usability of VS Code.

Although Visual Studio Code was initially identified as the strongest overall IDE option due to its debugging tools, extension support, Git integration and team familiarity, the final development workflow used a combination of Visual Studio Code, Geany, and Arduino IDE. For the Python portions of K.I.P.S., Visual Studio Code was used primarily when programming off-device on external development hardware, where storage space and system resources were less limited. However, when programming directly on K.I.P.S., the team used GEany because it was already included with the Raspberry Pi software environment and provided a lightweight editor without requiring additional installation, extensions, or storage overhead. This was important because K.I.P.S. had limited available storage space, so avoiding unnecessary software bloat was a practical design consideration.

For the ESP32-specific programming, the team used Arduino IDE. Although Arduino IDE was not selected as the strongest full-scale IDE overall, it was the most accessible and practical option for programming the ESP32 from the available external hardware. It allowed the team to upload firmware, monitor serial output, and quickly test ESP32 behavior while developing the sensor and communication logic. This made it useful for programming the ESP32 to poll sensors and read/write data between the microcontroller and the Raspberry Pi. Therefore, the final IDE choice was not a single environment, but a combined workflow of Geany for lightweight on-device Python programming, Visual Studio Code for off-device Python development, and Arduino IDE for ESP32 firmware development. This combination best matched the actual

development constraints of the project while still supporting productivity, accessibility and hardware compatibility.

4) Standards and Design Constraints

4.1) Industrial standards

4.1.1) PCB Design Standards

PCB design standards define the criteria required for a PCB to be manufacturable, and these standards are determined by individual manufacturers. For this project, JLC PCB is the designated manufacturer, so their specifications will be followed. Beyond manufacturability, additional design standards are applied to ensure the PCB functions correctly, such as using appropriate trace widths for high-current applications. JLC PCB's standards include requirements for clearances, maximum and minimum thicknesses and diameters, and acceptable file formats.

For the selected manufacturer, the outer layers of the 2-layer boards are limited to 1 or 2 oz copper. Hole sizes range from 0.3 mm to 6.3 mm, with vias requiring a minimum 0.3 mm diameter and 0.5 mm pads. Minimum clearances specify 0.5 mm between holes and pads for different nets. Vias must be at least 0.254 mm away from other vias and tracks, regardless of net. Pads need to be 0.127 mm from pads of other nets and 0.2 mm from tracks. The minimum trace width is 0.127 mm, with tracks spaced at least 0.127 mm apart. Finally, the minimum distance from any track to the edge of the board is 0.3 mm [195].

IPC-2221, formally known as the “Generic Standard on Printed Board Design”, is a widely recognized standard published by the Institute for Printed Circuits (IPC). It provides comprehensive guidelines for designing printed circuit boards (PCBs) and other forms of printed wiring, covering electrical, mechanical, thermal, and manufacturability considerations. IPC-2221 serves as a foundational document for PCB designers and engineers, offering standardized rules to ensure reliable, manufacturable, and cost-effective designs across a wide range of applications [195].

1. Scope and Purpose

IPC-2221 establishes generic design requirements applicable to various types of PCBs, including single-sided, double-sided, and multilayer boards. Its primary objectives are; Ensuring electrical performance, including signal integrity, current-carrying capacity, and impedance control. Maintaining mechanical reliability, covering board dimensions, hole sizes, and spacing tolerances. Promoting thermal management, such as via sizes for heat dissipation and pad design

for soldering reliability. Supporting manufacturability and testability, guiding designers to produce boards that can be reliably fabricated, assembled, and tested. The standard is intentionally broad and serves as a baseline, allowing specific applications or higher-performance requirements to be addressed by supplementary standards, such as IPC-2222 for rigid boards, IPC-2223 for flexible circuits, or IPC-2226 for high-speed designs [195].

2. Key Design Guidelines

IPC-2221 provides detailed design rules for several key areas of PCB development. It defines conductor width and spacing requirements based on current-carrying capacity, temperature rise, and copper thickness, and specifies minimum trace spacing to prevent electrical shorts and dielectric breakdown, including for high-voltage applications. The standard also outlines via and hole design guidelines, recommending via sizes for electrical and thermal performance, and establishing annular ring dimensions and tolerances to ensure reliable plating and solderability. For component placement and pad design, IPC-2221 provides guidance on pad sizes and shapes for both through-hole and surface-mount components to optimize solder joint reliability, as well as recommendations for component spacing to improve thermal dissipation, assembly ease, and minimize interference. It also suggests appropriate board materials and thicknesses based on application requirements, addressing factors such as thermal expansion, dielectric strength, and mechanical rigidity to ensure structural support and durability during manufacturing. Finally, the standard covers thermal and mechanical considerations by recommending practices for effective heat dissipation through thermal vias, copper pours, and component layout strategies, while addressing mechanical stress and long-term reliability, particularly in multilayer and flexible circuit designs [195].

3. Importance and Applications

IPC-2221 is widely adopted across multiple industries, including consumer electronics, automotive systems, aerospace, and industrial equipment, serving as a universal reference for PCB designers and manufacturers. Its comprehensive guidelines standardize key design parameters such as conductor width, spacing, via dimensions, and material selection, which helps ensure consistency and reliability across products developed by different teams or suppliers. By adhering to IPC-2221, engineers can significantly reduce design errors, minimize the risk of electrical or thermal failures, and streamline the manufacturing process by aligning with common fabrication and assembly capabilities. This not only lowers production costs but also shortens development cycles by minimizing the need for redesigns and prototype iterations. Furthermore, IPC-2221 supports the creation of high-reliability electronic systems used in safety-critical applications such as medical devices, aerospace avionics, and military hardware, where even minor failures can lead to severe consequences. Its emphasis on mechanical integrity, thermal management, and long-term performance makes it an essential foundation for achieving dependable and durable PCB designs across a wide range of modern technologies [195].

4. Relationship with Other IPC Standards

While IPC-2221 provides a general framework for printed circuit board (PCB) design, other IPC standards offer more specialized rules for specific applications. For example, IPC-2222 focuses on rigid printed boards, IPC-2223 addresses flexible circuits, IPC-2226 covers high-speed and high-frequency designs, and IPC-7351 defines surface mount design and land pattern requirements. Designers commonly use IPC-2221 in conjunction with these related standards to achieve robust, manufacturable, and high-performance PCB layouts. As illustrated in Figure 4.X, IPC-2221 serves as the baseline standard, while IPC-2222 through IPC-2226 expand upon it with detailed guidelines tailored to different types of PCB designs [195].

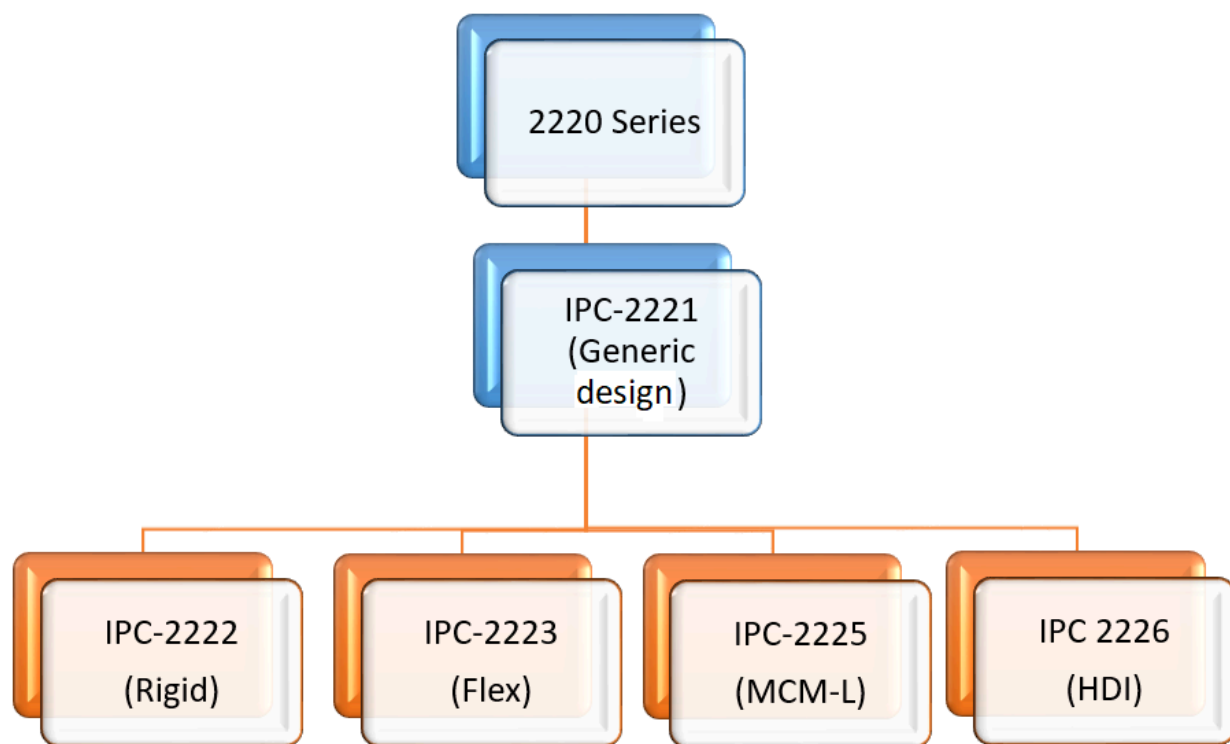


Figure 4.X: “Hierarchy of IPC Design Specifications” (protoexpress [196])

Summary

IPC-2221 is the cornerstone of PCB design standards, offering a comprehensive set of guidelines for electrical, mechanical, thermal, and manufacturability aspects of printed circuit boards. Its generic framework makes it applicable across diverse industries and technologies, promoting reliable, cost-effective, and high-quality PCB designs. By following IPC-2221, designers can ensure that their boards meet industry-standard performance, assembly, and reliability requirements, while providing a strong foundation for more specialized standards tailored to advanced applications.

4.1.2) I²C Protocol Standards

The Inter-Integrated Circuit (I²C) bus is a standardized serial communication protocol developed by Philips Semiconductors (now NXP) in the early 1980s. It was created to provide a simple, efficient, and low-cost method for communication between microcontrollers and various peripheral devices on the same circuit board. I²C has since become a fundamental part of modern electronics and embedded systems because it allows multiple components—such as sensors, displays, and memory chips—to communicate using only two shared lines. These lines are known as the Serial Data Line (SDA) and the Serial Clock Line (SCL). The SDA line is responsible for carrying data between devices, while the SCL line carries the clock signal that synchronizes communication. Both lines are open-drain, meaning they require external pull-up resistors to function correctly. This design allows multiple devices to share the bus without risking short circuits when one tries to drive the line low while another attempts to drive it high [197], [198], [199].

I²C 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. However, only one master can control the bus at a time. The master device is responsible for initiating and terminating data transfers, generating clock pulses, and addressing the appropriate slave device. Each communication transaction begins with a START condition during which the master pulls the SDA line low while the SCL line remains high, signaling all devices on the bus that communication is about to begin. This is followed by the transmission of a 7-bit or 10-bit address of the target slave device and a read/write (R/W) bit that specifies whether the master intends to read data from or write data to the slave. The addressed slave acknowledges by pulling the SDA line low for an ACK (acknowledge) signal. After this, data is transferred in 8-bit bytes, each followed by an ACK or NACK (no acknowledge) bit, until the master sends a STOP condition releasing the bus for other devices to use [197], [198], [199].

The I²C standard defines multiple speed modes to accommodate different applications and performance requirements. The most common are Standard Mode (up to 100 kbit/s) and Fast Mode (up to 400 kbit/s), suitable for sensors, displays, and low-speed peripherals. For higher-speed applications, Fast Mode Plus (1 Mbit/s), High-Speed Mode (3.4 Mbit/s), and Ultra-Fast Mode (5 Mbit/s) are also available, though these are less commonly used in typical embedded designs. Despite its versatility, I²C is best suited for short-distance communication—typically less than one meter—because bus performance can degrade with increased line capacitance and longer cables [197], [198], [199].

One of the key advantages of I²C is its simplicity and scalability. It requires only two lines regardless of the number of connected devices, making it extremely efficient for systems with limited pin availability, such as microcontrollers and compact embedded boards. The addressing system supports both 7-bit and 10-bit addresses, allowing for a large number of devices to

coexist on the same bus. Furthermore, its multi-master capability adds flexibility to complex systems, enabling multiple controllers to share peripherals without additional hardware. However, these benefits come with certain trade-offs. Compared to faster protocols like SPI (Serial Peripheral Interface) or UART (Universal Asynchronous Receiver/Transmitter), I²C has lower data rates and more complex electrical requirements, particularly due to the open-drain design and the need for carefully chosen pull-up resistors to ensure proper signal timing and reliability [197], [198], [199].

I²C's combination of simplicity, low hardware cost, and wide adoption has made it a cornerstone of embedded system communication. It is extensively used in connecting microcontrollers to peripheral components such as temperature and pressure sensors, real-time clocks (RTCs), EEPROMs, power management ICs (PMICs), and display modules. In consumer electronics, I²C can be found in smartphones, laptops, and televisions, while in industrial and automotive systems, it is used for diagnostic communication and sensor networks. The protocol's specifications and electrical parameters are maintained and periodically updated by NXP Semiconductors, ensuring compatibility across devices from different manufacturers. The latest official documentation, I²C-bus specification and user manual (UM10204), defines timing requirements, bus arbitration rules, addressing formats, and communication sequences [197], [198], [199].

In summary, the I²C standard represents one of the most practical and enduring communication methods in electronics. Its two-wire simplicity, robust device addressing, and wide compatibility make it ideal for interconnecting multiple integrated circuits within the same system. Although it is not the fastest serial interface available, I²C's design optimizes reliability, hardware efficiency, and ease of implementation—qualities that have solidified its role as an essential standard in embedded and digital circuit design.

4.1.3) UART Protocol Standards

UART (Universal Asynchronous Receiver/Transmitter) is an asynchronous serial communication protocol that transfers data between two devices without using a shared clock signal. Instead, both the transmitter and receiver rely on agreed-upon communication settings that define how data is packaged and interpreted [200]. A UART data frame begins with a start bit, which signals the beginning of transmission by pulling the line low. This is followed by a selectable number of data bits—typically 8 bits—sent least significant bit first. An optional parity bit may be included to provide simple error detection, and the frame ends with one or two stop bits, which return the line to a high state and allow the receiver time to prepare for the next frame [200].

The baud rate, which represents the number of bits transmitted per second, must be identical on both devices for communication to work correctly. Common standardized baud rates include 9600, 19200, 38400, 57600, and 115200 bps, with higher rates used for faster data applications.

UART does not define physical voltage levels by itself, so different electrical standards are used depending on the system [200]. Microcontrollers typically use TTL-level UART operating at 3.3 V or 5V, while RS-232 uses higher and inverted voltages, and RS-485 uses differential signaling for long-distance communication [200].

Error handling is built into the protocol and includes parity errors, framing errors when stop bits are misread, and overrun errors if incoming data is not read from the buffer in time [200]. Flow control can be added through hardware signals such as RTS and CTS or through software-based methods like XON and XOFF to prevent data loss when one device is slower than the other. Overall, UART protocol standards define the frame structure, timing, electrical characteristics, and error-handling mechanisms that make asynchronous serial communication reliable and widely compatible across embedded systems.

4.1.4) Software Protocol Standards

The primary programming language standard for KIPS will be the C++ programming language. C++ is governed by the International Organization for Standardization (ISO) 14882 standard. This standard is responsible for ensuring uniform syntax and semantics across different instances of the C++ programming language. Additionally, this standard also accounts for standard library behavior of C++ [201].

This standard ensures that code that is written for one C++ compiler in compliance with this protocol will work universally on other compliant compilers. This is critical for reproducibility and scalability. For example, if KIPS were to be reproduced on a larger scale with subtle tweaks, we can ensure that by following this standard that the KIPS source-code could be run on any compliant compiler.

4.2) Design Constraints

4.2.1) Time

The most important constraint to keep in mind for this project is time. We only have two semesters to complete the entire project from scratch, meaning that from initial research to the final product, the timeline is limited by a hard deadline. The project is ultimately constrained by what can be realistically accomplished within this timeframe. To ensure the deadline is met, several considerations and decisions must be carefully planned.

Since the project deadline is fixed by the academic calendar, it is absolutely bound by this timeframe. There will be no extensions under any circumstances, including major disruptions such as hurricanes or injuries. This means that all work, including testing, must fit within the

allotted time. The team has alleviated this constraint somewhat by starting research in Spring 2025 and working on the project continually throughout the year.

A general challenge with this time constraint is that the group must complete significant testing, fixing, and designing within the short two-semester window. Setbacks and errors are inevitable in senior design, so the margin for error must be as low as possible. The time constraint becomes critical because these potential errors directly threaten the success of the project.

Another time-sensitive issue is part sourcing particularly the PCB. The PCB is an absolutely critical component, and multiple units are required. Shipping times are highly time-sensitive, especially as manufacturers may be overloaded with orders from other groups. This makes early ordering essential, but revisions and additional PCBs may still introduce delays that threaten the hard deadline.

The main concern with the time constraint is its cumulative effect. If setbacks stack up, the project could experience significant delays potentially resulting in reduced features or even project failure. Careful planning and execution are required to prevent this.

To mitigate these risks, the team must work collectively, establishing rules and mentalities that promote time management, accountability, responsibility, and communication. Detailed planning ensures that setbacks can be accounted for, minimizing their impact on the schedule. By keeping the time constraint in mind and actively working around it, the project can be completed on time with all intended features.

4.2.2) Economic Constraints

For this project, one of the major limiting factors—aside from time—is economic constraint. Since the project is not externally sponsored, all funding must come from our group of four team members pooling resources. As a result, every component we procure must be carefully justified and fall within the agreed-upon budget of approximately \$2,000.

A significant portion of the budget is allocated to PCB fabrication, which represents one of the largest expenses. This is due to the need to design, manufacture, and ship four separate PCBs, each of which is critical to the overall system. Shipping, in particular, contributes substantially to the cost. To address this, we have set aside \$700 specifically for PCBs. These boards form the foundation of the entire project, integrating power systems, the motherboard, and other essential components. Without them, the project cannot function. Ensuring their quality and on-time delivery is therefore a top priority, and justifies the large portion of the budget dedicated to this area.

Another major cost is the Raspberry Pi Compute Module 4, priced at around \$90 each including tax. We require two units, bringing the total allocation to roughly \$200. This investment is necessary because these modules serve as the main processing units, handling data acquisition and system control. Their performance and reliability are critical, making them a justified and essential expense.

Finally, unforeseen costs present an additional economic challenge. Factors such as shipping price fluctuations, component availability, or unexpected hardware failures can increase expenses beyond planned estimates. For example, if a critical part becomes unavailable or needs replacement, the budget could easily be exceeded. To mitigate these risks, strategic purchasing and robust design decisions are essential.

By taking these economic constraints into account early in the design process, we can prioritize key components such as PCBs and MCUs, conduct thorough research to select cost-effective and reliable parts, and maintain a well-structured budget that includes contingency for unforeseen expenses. This approach allows the project to proceed smoothly without compromising functionality or quality due to monetary limitations.

4.2.3) Environmental Constraints

One of the major constraints when it comes to the KIPS MK II is environmental. This is due to how the device is designed with the intention of having some degree of mobility. Designing an arm-mounted computer requires careful attention to thermal and environmental conditions. These devices may be exposed to extreme temperatures, ranging from cold outdoor conditions to hot indoor environments, which can stress both electronic components and the enclosure. Thermal cycling caused by moving between these extremes can lead to solder joint fatigue, warping of plastics, or component failure. Effective heat dissipation is crucial, especially in compact devices where airflow is limited. Using metal chassis, thermal vias, or low-power components can help maintain safe operating temperatures, while ensuring the device remains comfortable to wear.

Another major consideration is moisture, humidity, and particulate protection. Since the device is in close contact with the user, sweat and oils can corrode contacts and internal circuitry over time. Rain, high humidity, or dusty outdoor environments add additional risk. Employing IP-rated enclosures, gaskets, and conformal coatings can prevent moisture ingress, while filter vents can help manage air pressure without allowing dust to enter. Dust, dirt, and other particulates, particularly in industrial or outdoor settings, can accumulate and cause mechanical or electrical failures if not properly mitigated.

Mechanical stresses are also significant for wearable devices. The constant motion of the arm introduces vibration and shock, which can damage delicate electronics or cause connectors to loosen. To address this, PCBs can be mechanically supported with standoffs or silicone pads, and

ruggedized connectors can be used to prevent disconnections. Standards such as MIL-STD-810G are often referenced to ensure devices can withstand repeated impact, vibration, and rough handling. In addition, ergonomic design plays a role, as the weight, balance, and surface temperature of the device directly affect comfort and usability during prolonged wear.

Chemical and electromagnetic factors must also be considered. Exposure to sweat, cleaning agents, or other chemicals can degrade seals, coatings, and external materials. Choosing corrosion-resistant materials, such as anodized aluminum or silicone gaskets, helps maintain longevity. Close proximity to the body and other electronics may introduce electromagnetic interference, potentially affecting wireless communications or sensitive sensors. Appropriate shielding, grounding, and compliance with EMI/EMC standards are important to maintain reliable operation.

Finally, power and energy considerations constrain the design of arm-mounted computers. Limited space restricts battery size, requiring careful selection of energy-dense cells and efficient power management. Batteries must also remain safe under thermal and mechanical stress to prevent overheating or failure. Depending on the application, additional factors such as altitude and pressure changes may also be relevant, necessitating pressure-equalizing features or venting solutions. Taken together, these environmental constraints highlight the complexity of designing wearable computing devices that are durable, reliable, and comfortable for real-world use.

4.3) Other Constraints

4.3.1) Sustainability constraints

Sustainability is another key constraint for the KIPS MK II, driven by considerations such as energy consumption, material usage, and environmental impact. Although this version of the project primarily focuses on functionality, sustainability remains an important factor in both the design and long-term viability of the system.

The KIPS MK II operates using a combination of AC wall power and battery power, depending on the mode of operation. Both power sources contribute to overall energy consumption. The battery is designed for approximately 500 charge cycles, but this lifespan depends on the balance between wall-powered operation and battery discharge. Components such as the Raspberry Pi CM4, display, and other peripherals contribute significantly to power draw. While this prototype does not emphasize power efficiency, future iterations will need to prioritize lower power consumption and improved thermal management.

Another major sustainability consideration is electronic waste (e-waste). Components like the Raspberry Pi CM4 and ESP32 have limited recyclability due to their complex manufacturing and integrated designs. Once they fail or are replaced, they contribute to electronic waste. To help

mitigate this, the project design incorporates modularity, allowing individual components to be replaced or upgraded without discarding the entire system.

PCB manufacturing also poses sustainability challenges. Even though the current project uses a small number of boards, large-scale production would require careful attention to material sourcing and waste reduction. Many PCBs use non-biodegradable and potentially hazardous materials, so future designs should consider lead-free solder, RoHS-compliant materials, and efficient board layouts to minimize waste.

By accounting for energy efficiency, component lifespan, and material sustainability, this project acknowledges its environmental impact and identifies clear paths for improvement. While these factors are secondary at the prototype stage, they will become central to future development, guiding efforts toward a more sustainable, efficient, and environmentally responsible design.

4.3.2) Ethical constraints

Ethical constraints come from choosing the correct parts from the correct manufacturers. This means choosing from manufacturers that do not engage with unethical labor practices, such as child labor or forced labor. It is essential that our project does not indirectly support companies that exploit workers in developing countries. While this may sometimes present a trade-off between cost and ethics, our group decided that maintaining ethical integrity takes priority.

To uphold this standard, we sourced components only from reputable and verified manufacturers with strong records of fair labor practices and responsible supply chains. Additional research was conducted to confirm that the suppliers we used operate transparently and comply with international labor and sourcing regulations. By doing so, we aimed to ensure that our project is not only technically sound but also socially responsible and ethically aligned with fair business practices.

4.3.3) Health and Safety constraints

Health and safety is another critical constraint to consider in the design and development of the KIPS MK II. The primary goal is to ensure that the device poses no risk to users, developers, or bystanders throughout its testing, operation, and demonstration phases. Since this device is intended to be worn by humans, special care must be taken to prevent hazards such as burns, electric shock, or irritation. To comply with ethical standards, an Institutional Review Board (IRB) form must be submitted and approved before involving any human participants in testing.

One of the most significant safety concerns involves the battery system. Improper handling, charging, or discharging of batteries can lead to overheating, leakage, or even explosions. Strict safety procedures must therefore be followed during assembly, testing, and operation to prevent

fire hazards. This includes selecting appropriate protection circuits, ensuring proper enclosure ventilation, and avoiding physical damage to battery cells.

Material selection is also an essential factor for user safety. All materials used in the device's construction—especially those that make direct contact with skin—must be non-toxic, heat-resistant, and comfortable for extended wear. Considering these health and safety measures early in the design process not only ensures compliance with safety standards but also enhances the reliability and usability of the final product.

4.3.4) Manufacturability constraints

Another constraint to consider for this project is manufacturability. The primary method of construction and production depends on the tools and resources available at the university. As an undergraduate team, we are limited to what the university's facilities and senior design lab can support. Therefore, all designs and components must be compatible with the available fabrication tools and equipment. Keeping the design as simple as possible helps alleviate this constraint and ensures that the project can realistically be built and tested within the university environment.

For this project to be successful, certain components must be premanufactured. Our design work focuses on how these components integrate with each other rather than designing every subsystem from scratch. For example, the Raspberry Pi Compute Module 4 and its carrier board are prebuilt devices that serve as the foundation of the system. It would not be practical for students to design and fabricate a single-board computer (SBC) from the ground up. Instead, our design effort is concentrated on interfacing the Pi with other components such as the screen, ESP32, and DAC. Since these modules are purchased online, their availability and cost can fluctuate, meaning alternative components may need to be considered if supply issues arise.

Another important factor in manufacturability is the operating environment. A mobile device must be capable of withstanding external conditions such as heat, humidity, and potential physical impact. The version produced for senior design will serve primarily as a proof of concept, not a mass-market product. However, a future production-ready version would need to account for these factors in both component selection and enclosure design.

To address these challenges, the system will be designed with modularity in mind. This approach ensures that a failure in one module will not cause the entire system to fail, improving maintainability and reliability. Proper component selection will also ensure that the system functions effectively in the intended environment. Additionally, careful PCB layout design will be essential to guarantee that all circuits operate correctly and that any necessary revisions can be made using the university's available tools. Understanding and planning for these constraints now will be invaluable for both the current project and any future iterations of the design.

4.4) Software Constraints

The types of software the KIPS device will run also place constraints onto its capabilities. From the operating system to the KIPS mobile application, each type of software used to implement the device's software features all have advantages and disadvantages. The major limitations of the software mostly stem from its mobile mode, where resources and power consumption become the primary concern.

In mobile mode, the device will be running solely off the battery. Therefore, it is imperative to make a power efficient, lightweight software ecosystem that allows for the device to use up as little energy as possible to complete essential tasks in mobile mode. In order to mitigate power use and utilize it as efficiently as possible, our chosen programming language, C++, was chosen. C++'s general purpose use, and lower abstraction to allow for memory control, means it gives the programmer more control over its operations. This means the program can be more tailored to be power efficient, as it will have more fine control over high level processes.

Another aspect of the software, the operating system, was chosen to be the Raspbian OS that came with the Raspberry Pi Compute Module 4. Due to the nature of the Raspberry Pi being a smaller, more mobile computer, using the operating system already installed on it will mean the operating system's architecture will have these aspects in mind. For the programmers, it saves time and cost to use a pre-installed operating system instead of building an entirely new one, or using a different one.

5) ChatGPT

AI Models like ChatGPT can be beneficial for doing research and writing papers. It allows the user to gather information on a question in an instant, concisely delivering quick explanations of what was found. This also enables the user to ask further questions and let GPT do the work of finding sources that correlate to what the user wants to know. However, this information should never be taken at face value. Misquoting, incorrect sources, incorrect mathematics, etcetera are all issues with AI models that are present today. Users can get around such things by directly asking for the sources GPT used for its answer, doing math themselves, or just not using GPT at all. It is important to use it as only a tool to enhance your ability to quickly gather sources, and it is equally important to vet the information you find.

Since its release, GPT has become much more advanced and refined with each iteration of the LLM. Now with GPT-5, it can assess if the user prompt requires a deeper level of thinking, and is promoted as essentially being able to converse with a PhD holder on subject matters being asked. As we began research for our project prior to senior design 1, we were all under the belief that GPT can be used as essentially an advanced search engine. That by prompting it, we can create a search query containing specific details that a normal search engine would not be able to comprehend. We would be having GPT conduct the actual search for us, saving us a lot of time

and search effort, so that we can instead verify its choices. Later on, we began to ask GPT for more help on concepts and technologies related to the project, design questions, or guidance with programming aspects

To this end, our members did use GPT for many of our search queries, with varying results. In some cases, we were able to narrow down the search to a few potential items and select an appropriate part after verifying its accuracy towards our requirements. Other times, GPT was unable to provide suggestions that matched our requirements for a part, sometimes even claiming the part matched our requirements despite this not being the case upon further research. For programming, while it proved to be a good starting point for tasks such as setting up the compute modules, after a certain point it would begin to repeat itself with information that was unhelpful to us.

Below are several case study examples of queries we gave while researching.

5.1) ChatGPT Prompt 1

For conceptual questions, I found the structure of ChatGPT's replies to be simple, understandable, and useful to what KIPS needed. In this prompt, which can be found in Appendix C, parts 1a and 1b. The technologies being assessed were ones that the heartbeat sensors we were researching did indeed utilize. Its suggestions for specific models of sensors were a part of its response because I had prompted it for suggestions on specific models prior to asking about different sensor technologies. For the purposes of KIPS, as it is not a designated medical device, accuracy was not top priority so long as it did not deviate beyond an established margin of error. Despite this, GPT's attention to noting the accuracy of certain methods was useful and appreciated. GPT could not have known the required compromise between expense, ease of use, and accuracy that was required of the technology methodology. Its input on PPG technology for the heartbeat sensor was ultimately one of the reasons why we chose the heartbeat sensor used in KIPS. It significantly reduced the amount of time required to research what the technologies are and their applications to the KIPS project down to a single prompt. For the purposes of getting a quick run down of heartbeat technologies without needing to do a deep dive as if I am a medical student, I find GPT gave me exactly the information I was looking for in order to make a decision on the sensor to use for KIPS.

As with any LLM, it is best to verify the response it gives with your own quick search. To this end, using a search engine to find websites that align with what the LLM says is important to do. For the purposes of more surface level conceptual research, this verification can arguably not be required to be as thorough as it would be for example with other device parts. In order to learn the concepts needed to properly develop and align the technologies needed to create a working product, another option is to do deeper research of the technologies the LLM brings up. For the

purposes of my research for the KIPS project, overall, I found that further inquiry of Chat GPT was sufficient as well, if I needed more information.

5.2) ChatGPT Prompt 2

The prompt above, found in Appendix C, parts 2a and 2b, was the second prompt I gave GPT after my initial, more lengthy prompt about the context of KIPS, and what we needed the MCU to do. This is the beginning of how I ultimately selected the ESP32-S3 MCU to use with the Pi CM4. When it comes to specific part selection, I found Chat GPT significantly reduced research time, since it would perform the search for me. My role for these searches would be to verify its results, and find information on its selected parts to ensure accuracy and the truthfulness of the part's capability. This in itself can be argued as being time consuming, but it ultimately removes about half of the effort required since potential parts have been listed. It also listed the features of the S3 that were most applicable and helpful to our project, which helped narrow down which aspects of the S3 I needed to verify. In other words, by using this second prompt, Chat GPT narrowed down its part selection for me, allowing me to more deeply research the S3 to ensure its suggestion would be our best choice for KIPs. In further prompts later on in the conversation, GPT was able to list strong reasoning behind why the S3 is the most well-suited choice for our needs among its suggestions of MCUs. Upon investigating the S3's datasheets, the specs Chat GPT stated were featured in the S3, were indeed what was stated on its datasheet. Overall, Chat GPT's part searching capabilities were sufficient enough to provide a good base upon which could be verified and built upon for other aspects of the KIPS project.

In this specific use case, I had Chat GPT searching for compatible MCUs based upon other hardware components and our desired end goal for the device to be capable of performing certain actions. Unlike researching more conceptual topics such as types of technologies, this was a part selection inquiry. Therefore, to this end, verifying the suggestions GPT made was extremely important. Not just for the S3, but for all other MCUs it suggested, even if we ultimately did not select them. I found upon inspecting their respective data sheets, the specifications of the other MCUs did fulfill the requirements of KIPS, in different ways. In the end, of the suggestions Chat GPT made and after verifying the accuracy of the specifications, it came down to weighing preferences in features of specific MCUs that were valued more for the KIPS project. In other words, from then on, Chat GPT had essentially given me a choice of 4 different MCUs, each from different manufacturers, to choose from. Using Chat GPT for this purpose was extremely helpful and did produce useful results, however this did not come without extensive searching of the products it suggested. This is even more imperative for a crucial component such as the MCU.

5.3) ChatGPT Prompt 3

The prompt and outcome for this specific scenario can be referenced in Appendix C, parts 3a and 3b.

In this case ChatGPT was asked to give a design of a system with several components. That being a USB-C charging circuit , a pd controller for it, charging IC and Gauge IC, and a switching MUX. First it gave a High level diagram of how those parts connect together. ChatGPT then provided some parts that it considered suitable for the task. It also gave some tips for how to design the system including to minimize BOM. Lastly it gave references to the datasheets of all the parts it listed. Some details to keep note of is that ChatGPT can misunderstand or even get facts wrong about the datasheets. Also that it may not understand compatible parts and incorrectly gives recommendations based on that. So that is something to keep in mind when considering the responses it gives. So it is important to read the datasheets of the listed parts and to do further research on information it gives.

5.4) ChatGPT Prompt 4

The prompt and outcome for this specific scenario can be referenced in Appendix C, parts 4a and 4b.

In this instance ChatGPT was tasked with creating a list of the required hardware and software for flashing a Raspberry Pi CM4 board. ChatGPT essentially synthesized information consistent with the Raspberry Pi part data-sheet, users manual and general reference materials in order to compile a list of required elements for proper setup. This seemingly instantaneous compilation and fusion of source materials makes referencing this information much faster and convenient in real time. One thing to note, however, is that ChatGPT may not actually access or reference the specified documents directly, even if explicitly specified. Nevertheless, if any of the underlying sources used during training were mislabeled or contained inaccuracies, the resulting response could also be affected by this bias. To ensure the validity of the information provided by ChatGPT, our team cross referenced this information with the Raspberry Pi CM4 datasheet and other setup-tutorials available online.

5.5) ChatGPT Prompt 5

The prompt and outcome for this specific scenario can be referenced in Appendix C, parts 5a and 5b.

In this example, ChatGPT is utilized to provide a few of the more commonly used Single-Board Computer (SBC) Compute Modules. Early on in the design of the project we understood the

need for an SBC but were unsure of the capabilities and popularity of the different models available. In this case, ChatGPT was prompted to provide some of the best SBC compute modules available for general purpose applications. ChatGPT provided a few different models including the Raspberry Pi CM5 and the NVIDIA Jetson Nano where it also directly compared the two. This helped the team gain a better understanding of the different SBCs that are currently available on the open market.

This prompt was intentionally left vague in the regard of application in an effort for ChatGPT to provide more general, overarching information about different SBCs such that team members could further research these models and get a better understanding of SBCs themselves. Once more information was known of the overall specifications of KIPS, more discrete and specialized prompts could be provided to further narrow down the search to components that may be more suited for the device.

5.6) ChatGPT Prompt 6

In this example, found in Appendix C parts 6a and 6b, some minor flaws can be found in the reasoning of ChatGPT and a need for specificity. In this use-case ChatGPT is essentially used as an advanced search engine. In Figures 5.X and 5.X, ChatGPT is prompted to find reputable sources, specifically from scientific journals, that explain how directional pads (d-pads) work. ChatGPT does a satisfactory job at explaining how these components work in both prompts, utilizing context switching to alter the description/explanation to fit the context in which it is being prompted. We can see the slight difference in phrasing between Figure 5.X and Figure 5.X, where the more brief explanation is offered first, followed by the more technical terminology of the second response, demonstrating that it was able to draw some semblance of context from each prompt and the subtle differences between the two.

One glaring issue though is the ‘credibility’ aspect of this prompt. In both instances, ChatGPT provides sources that are generally deemed non-reliable. In Figure 5.X ChatGPT responds with links to a wikipedia entry, a blog and an article by the digital game museum. Wikipedia entries have been characterized as non-reliable sources as the platform is open source and can be modified by anyone. While there is an extensive community of reviewers, they may not make changes right away leaving some malicious edits to go unnoticed and information cannot always be verified. Typically a blog is also classified as non-reputable scientific source as the majority of content is opinion based and can be authored by anyone, with experience in the field not being required. In Figure 5.X, after explicitly specifying a desire for sources with scientific ties sources related to the component itself, ChatGPT still failed to provide links that could realistically be utilized in this report. The first source is also from Wikipedia which has previously been ruled out. Additionally, the response includes a reddit discussion in addition to another discussion on an internet forum (NeoGAF). Reddit is a discussion platform utilized for a variety of different

topics and themes. While some more technical topics may be discussed over reddit, the credibility of contributors cannot be verified. This is also true for the other website listed.

5.7) ChatGPT Prompt 7

The prompt and outcome for this specific scenario can be referenced in Appendix C, parts 7a and 7b.

In this example, ChatGPT was asked to find summation amplifiers that operate with a 3.3 V input. This allows the circuit to be powered using the existing voltage rails. ChatGPT began by analyzing the request, providing feedback on the choice of input voltage and highlighting key design considerations. It then suggested several components from various manufacturers, including Texas Instruments and Analog Devices. Afterward, it offered design tips and recommendations for implementing those parts.

What is important to note about this prompt is that it specified only a single requirement—the 3.3 V input—without providing details such as the number of inputs or the maximum power rating. Because of this, ChatGPT based its results solely on the given information. An example of this can be seen with the OPA820, which supports both single and dual supply configurations. Its single-supply range is 5 V to 12 V, while the dual-supply range is ± 2.5 V to ± 6 V. Therefore, to meet the 3.3 V requirement, the dual-supply configuration must be used.

This limitation is not a fault of ChatGPT but rather a result of the prompt's lack of specificity. The key takeaway from this exercise is the importance of understanding what information is provided—and ensuring that component selections are verified by consulting the datasheet for precise specifications. ChatGPT is a valuable tool for guidance, but it should complement, not replace, thorough engineering research.

6) System Hardware Design

6.1) Power Delivery System

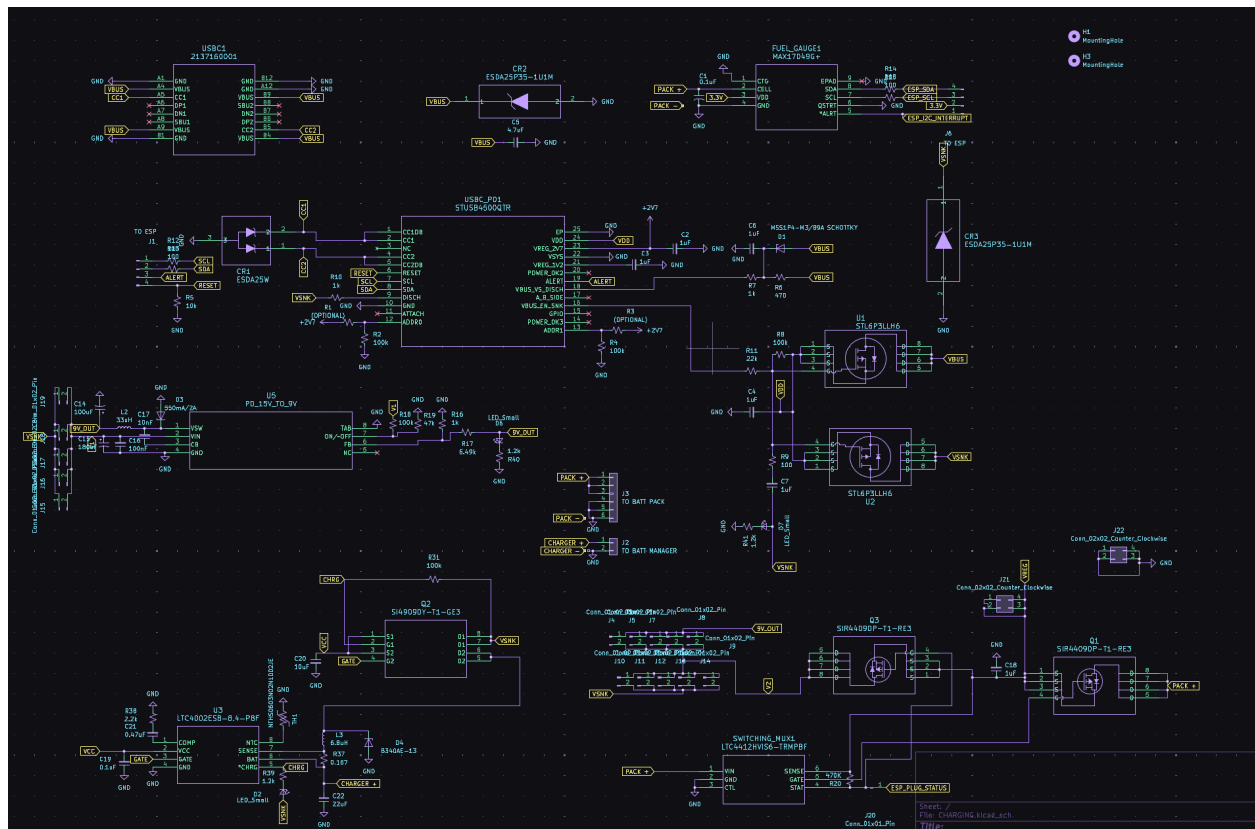


Figure 6.1: PD Board Schematic

This is the schematic for the Power delivery system. This is made of several components all tied together. There is the STUSB4500 which is the PD controller that ensures that the wall power line is 9V when it can. This connects to the USB-C port. This connects to the LTC4412 switching mux which changes between the 9V line and the battery which has a nominal 7.2V Max 8.4V. There will be some drop due to diode drops but that is the range. And both the charging and fuel gauge IC.

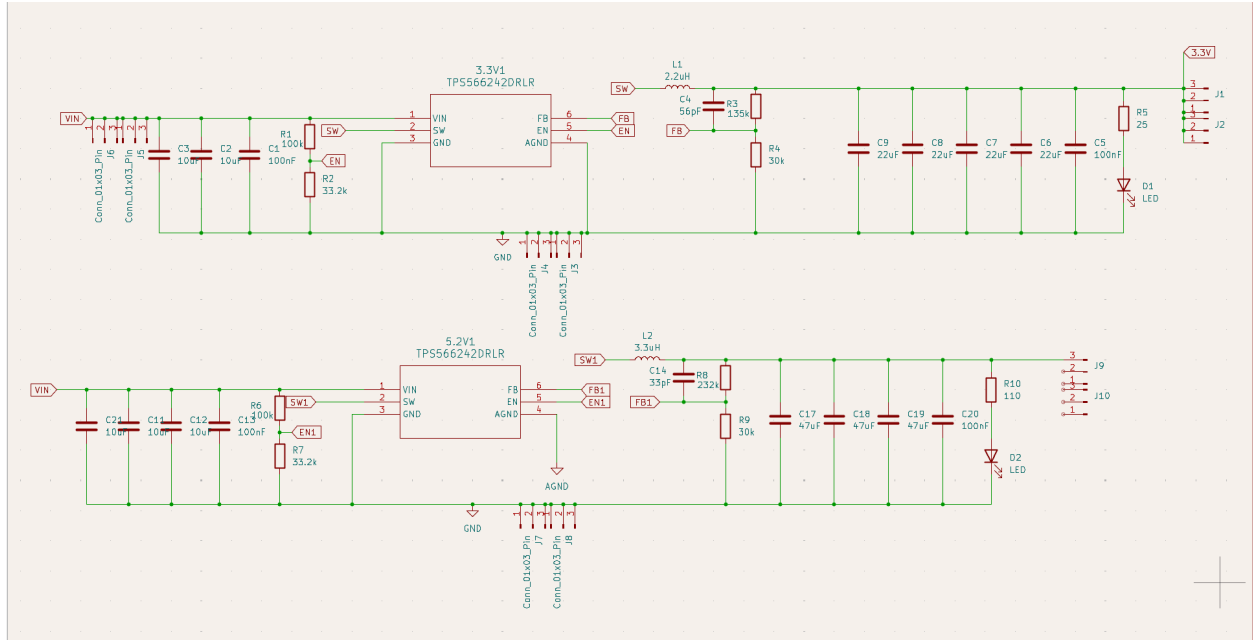


Figure 6.2: A schematic of the daughter board for the regulators

This is the schematic for the Buck Regulator daughter board. There are two TPS566242DRLR ICS one which outputs a 3.3V line and another which outputs 5.2V . The 3.3V line will go into some sensor and the MCU while the 5.2V line will power the SBC and the USB controller. The use of one regulator type was chosen as not only does it fulfill the requirements but also reduces the shipping costs. The input for the regulator will come in from the switching MUX. There is an LED to indicate activity on the board, this is for troubleshooting.

6.2) SBC and MCU Connections

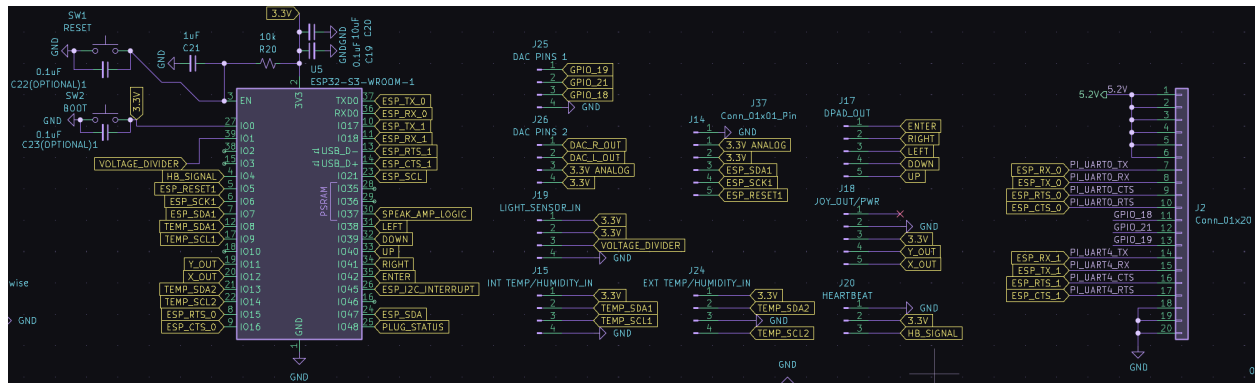


Figure 6.3: A schematic of Main board which contains the ESP32

This board contains the ESP32-S3-WROOM module, which serves as the main controller for all connected sensors. It is powered by the 3.3 V input rail and includes two ground pins tied to a common ground reference. The board features jumper headers that interface with the sensors, providing power, ground, and data connections for each. In addition to the sensor interfaces, the board also includes connections for a digital-to-analog converter (DAC). These interfaces ensure that all sensor readings and data collection are routed through this board. Due to its role as the central hub for sensor communication, it has the highest number of jumper connections, accommodating all necessary data lines.

6.3) Sensors

6.3.1) Temperature and Humidity

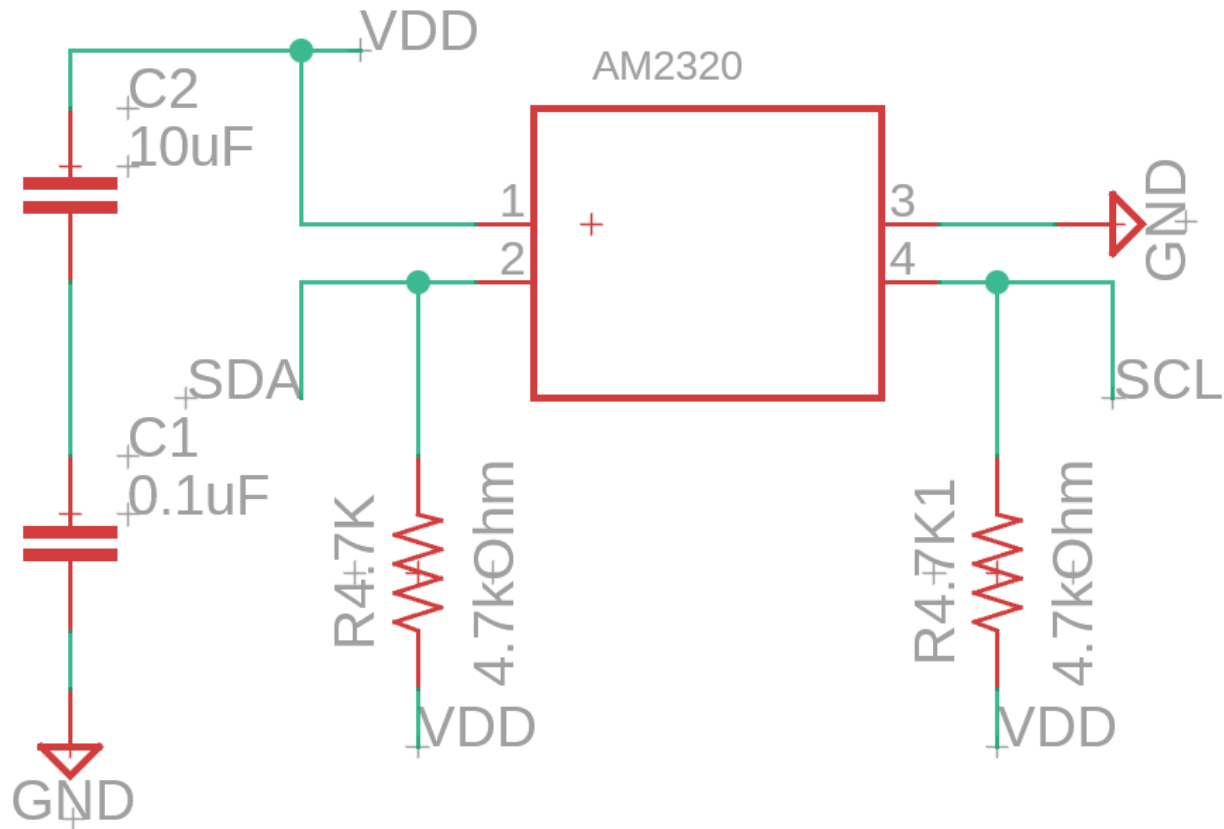


Figure 6.4: A schematic of the Temperature and Humidity sensor

The AM2320 temperature and humidity sensor operates using a straightforward pin configuration. Pin 1 is the supply pin (VDD), which is connected to a decoupling capacitor to ground in order to stabilize the local power supply and reduce noise, while also powering the internal circuitry of the sensor. Pin 2 and Pin 4 serve as the I²C communication lines (SDA and SCL). These lines must be pulled up to VDD with resistors to ensure proper logic-level operation and are connected to the I²C bus of the microcontroller for data transfer. Finally, Pin 3 is the ground (GND), which provides the return path and completes the power circuit. This arrangement ensures stable power delivery, reliable communication over I²C, and proper grounding for the sensor's operation.

6.3.2) Photoresistor

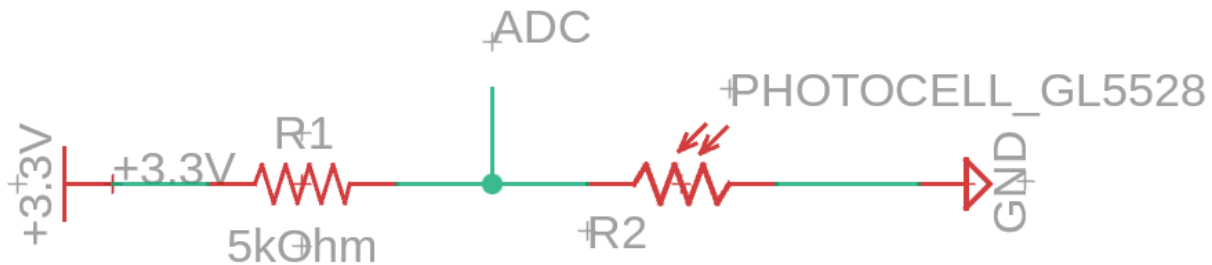


Figure 6.5: A schematic of the Photoresistor

The schematic for the GL5528 photoresistor is arranged so that it functions as part of a voltage divider circuit. In this configuration, changes in light intensity translate into changes in voltage, which can then be measured as lux readings. The GL5528 itself varies its resistance depending on the amount of light it receives — higher light levels reduce resistance, while lower light levels increase it. A fixed resistor is placed in parallel with the GL5528 and connected to the 3.3V power rail, forming the other half of the divider. The junction between the photoresistor and the fixed resistor is then connected to the ADC port of the Raspberry Pi. This allows the system to measure the varying voltage and interpret it as corresponding light intensity values.

6.4) Ports/Peripherals

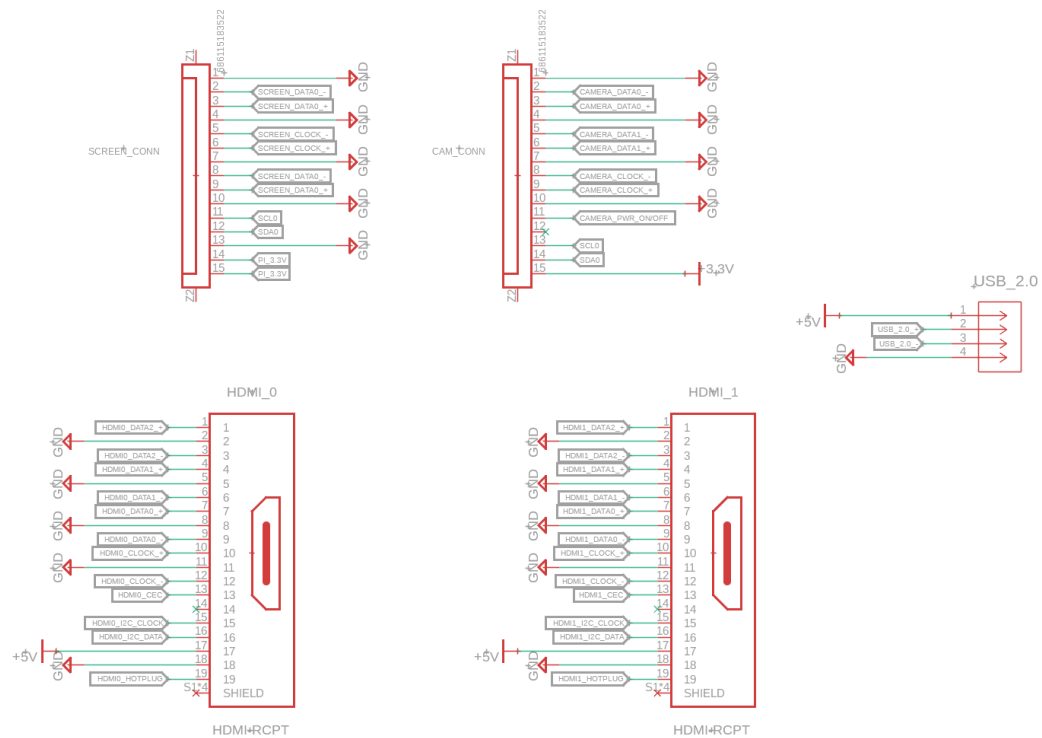


Figure 6.6: USB, HDMI, and FPC Ports

Above are the connections to our ports. We have 1x USB 2.0, 2x HDMI, and 2x 15 pin FPC connectors for the screen and camera respectively. All driver circuitry for the USB 2.0 port and HDMI ports are handled internally by the Pi CM4, all circuitry for the screen connector is handled by a driver board behind the screen, and all circuitry for the camera is handled on the camera's driver board. These have predetermined trace sizes and lengths

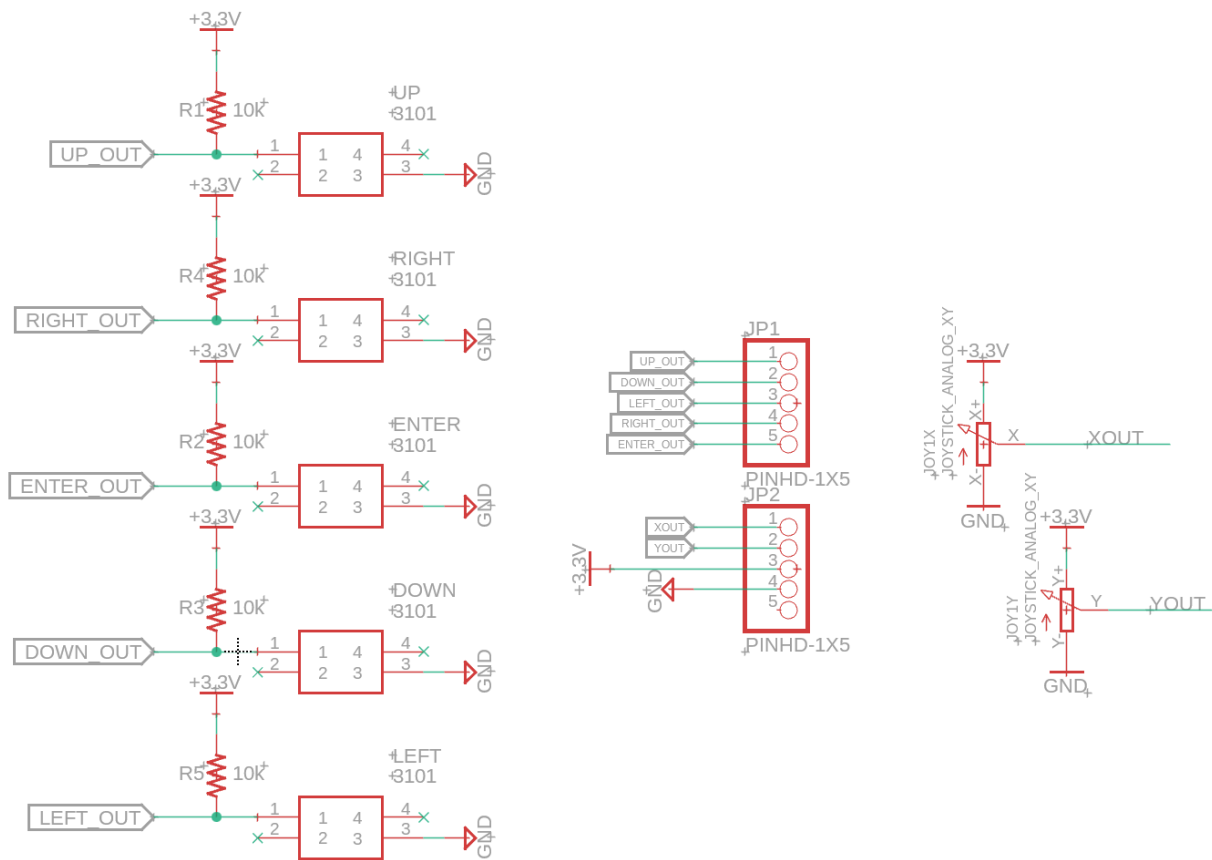


Figure 6.7: D-Pad and Thumbstick

This is the daughterboard for the D-Pad and the thumbstick. These are used for controlling the cursor of the KIPS and for scrolling. The board is powered by the 3.3 V rail for all of its inputs. Each direction is implemented as a separate component that comes together to provide the full range of directional inputs. Each input has a 10 k Ω resistor for current limiting. There are also X and Y inputs for the joystick, which will connect to jumpers leading to the main board. These signals will be programmed and processed on that board.

6.5) Flashlight

This is the daughter board for the flashlight module. It consists of three main components: a resistor, an LED, and a switch. This configuration allows the flashlight to operate independently of the main computer, using power from the 3.3 V rail. The circuit essentially forms a loop from the voltage input to ground through the LED, with the switch controlling the connection to ground. When the switch is closed, the circuit completes, allowing current to flow and the flashlight to turn on.

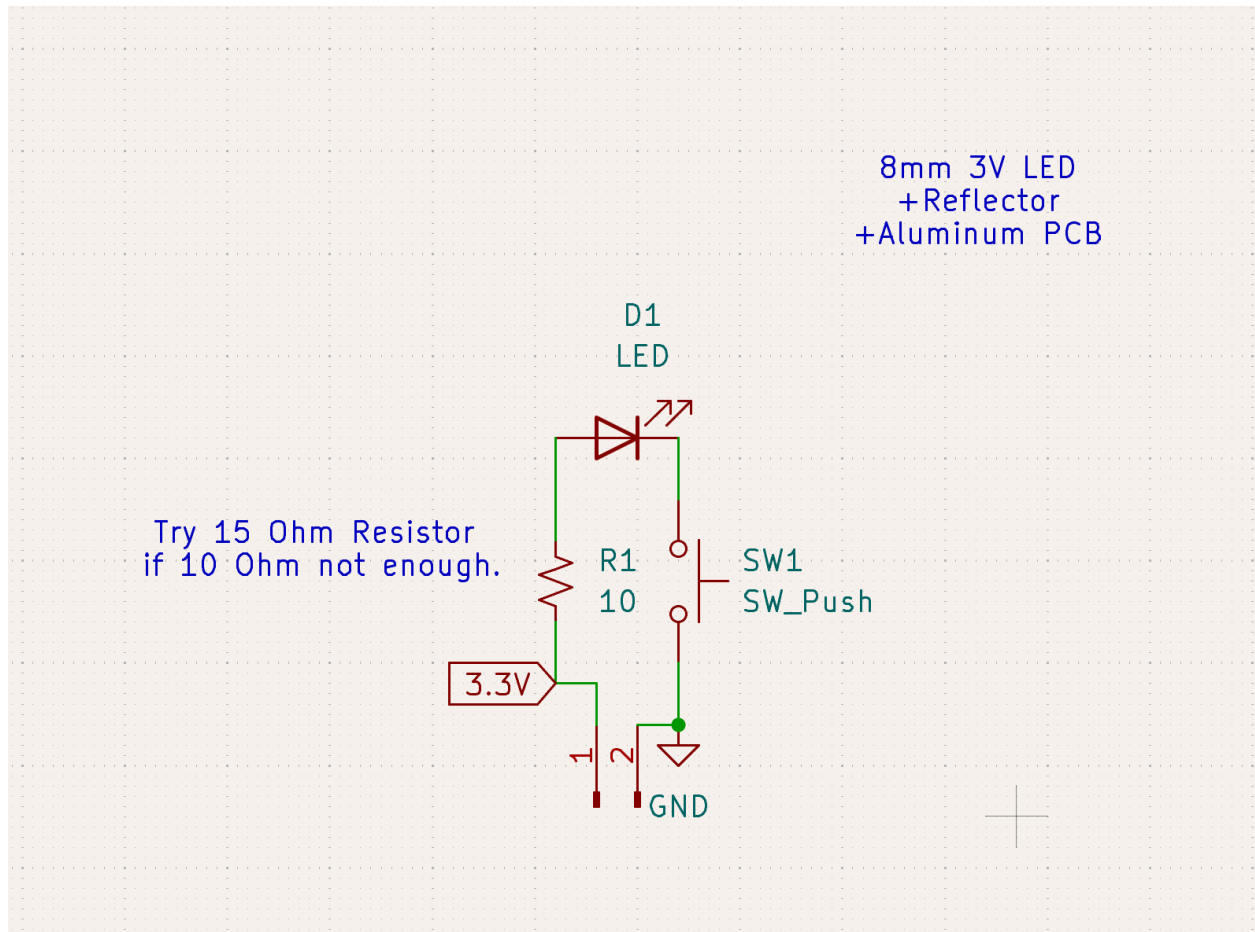


Figure 6.8: Flashlight Schematic

6.6) Audio Suite

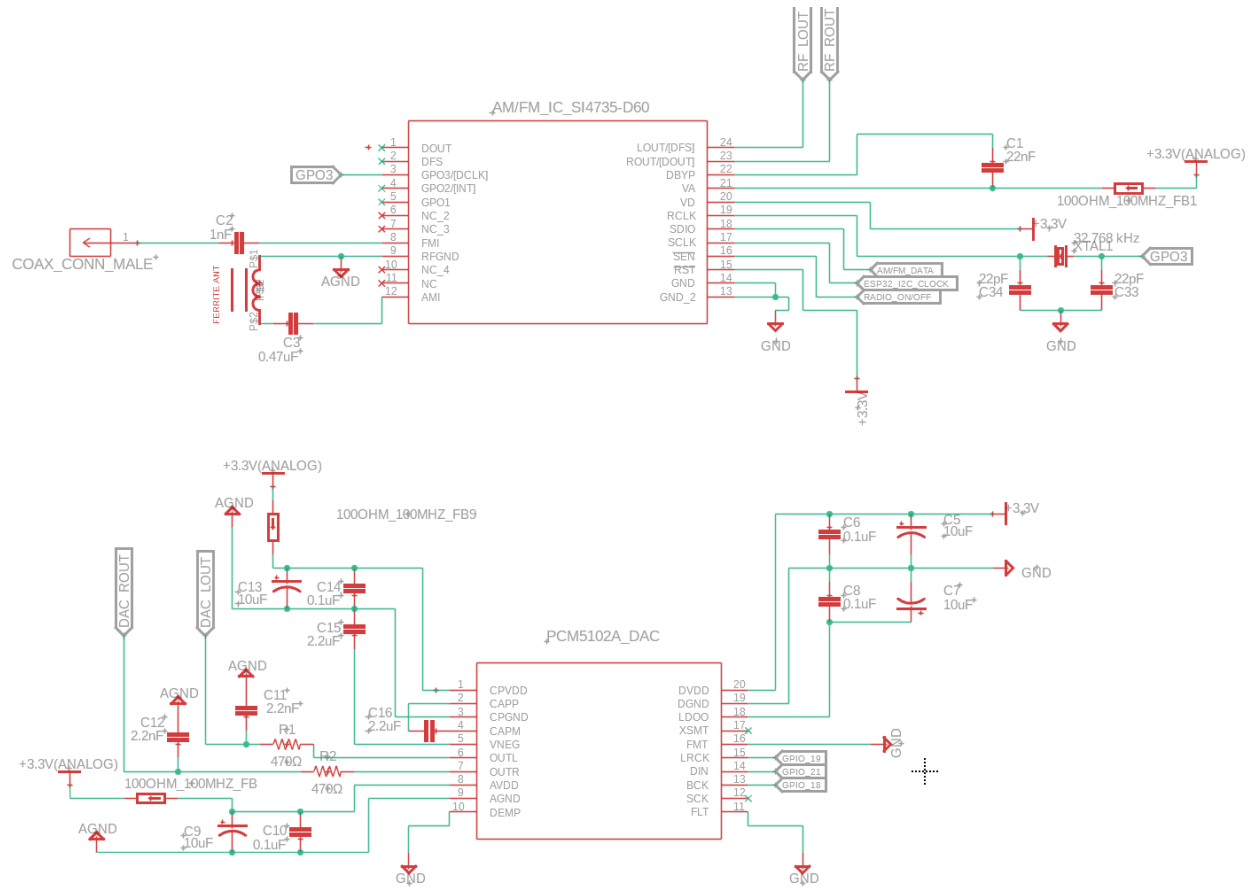


Figure 6.9: Digital-to-Analog Converter and AM/FM IC

This board contains the Digital-to-Analog Converter (DAC) and the AM/FM integrated circuit. Both components operate at 3.3 V but have separate power lines. Since these are analog components, they are connected to the analog ground (AGND). The DAC uses both polarized and ceramic capacitors for filtering and stability. It interfaces with the MCU through GPIO pins. The AM/FM radio IC is also connected to an external antenna. Both the AM/FM IC and the DAC output signals that feed into the summation amplifier.

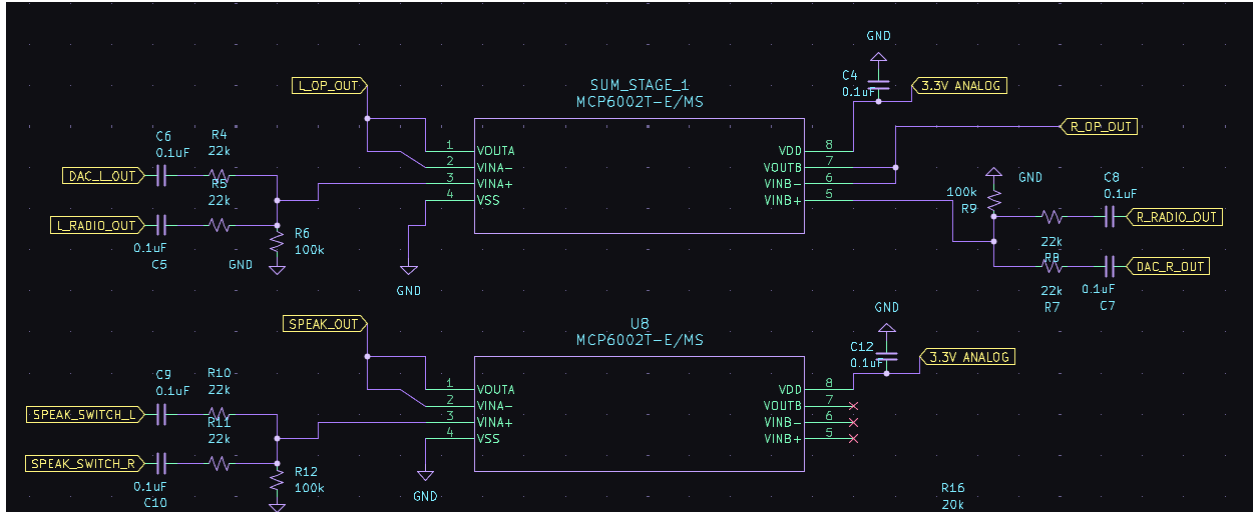


Figure 6.10: Summation Amplifiers

Volume Control and Final Amp ICs w/ Outputs (speaker connected via cable connector)

This section contains the summation amplifiers. There are two of them—one for the speaker and one for the headphone jack. The amplifiers combine the signals from the radio and the Digital-to-Analog Converter (DAC). Since this is an entirely analog stage, it uses analog ground (AGND) instead of the digital ground (GND). The headphone jack only requires a single output channel, so the InB and OutB pins are left unconnected (NC). The speaker, on the other hand, requires both left and right outputs to operate properly.

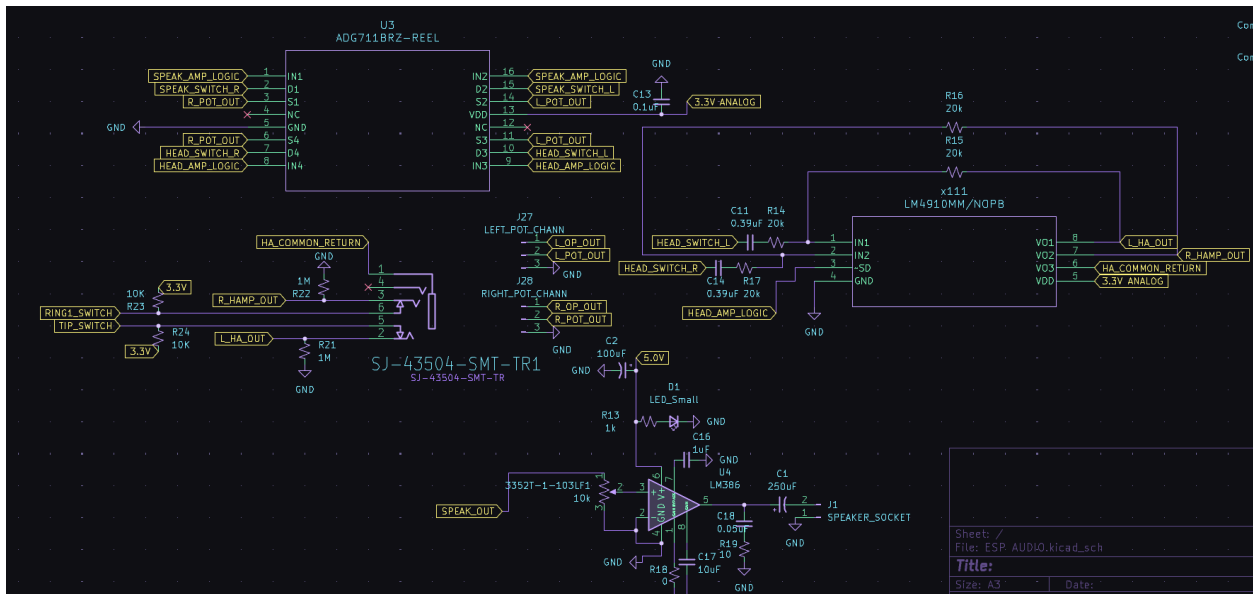


Figure 6.11: Switching IC, Amps, Volume Knob, and Outputs

This section covers the remainder of the audio suite beyond the summing amplifiers, DAC, and antenna. It includes the volume controls for the speaker, allowing adjustment of the output signal

amplitude. The 3.5 mm headphone jack is also located here as an audio output. Both the speaker and the headphone jack have their own dedicated amplifiers. A switching IC is used to select between different output signals, such as the left and right channels for the speaker and the headphone jack. This circuit operates from the 3.3 V rail and, since it is entirely analog, all components are connected to the analog ground (AGND).

7) System Software Design

The software component of this project can be split into two major sections. The first is the mobile software, which is what the device will be using to run a majority of the time as the user wears it. The primary goal of the mobile mode is to allow the user the ability to complete important tasks on the KIPS device, while consuming the least amount of power possible to preserve battery life. The second section is the desktop mode, which can be switched to when the user removes the device and plugs it into a power source. This mode will allow for less critical functions to be available to the user, since power is no longer a concern. Additionally, the user will be able to access the internet, and perform operations that would only be feasible on a desktop computer. Desktop mode is ideal for when the user may have a moment to sit down and search the internet for any information that is needed, or run any applications they may have installed that requires more power than what the mobile mode can afford to safely provide.

7.1 Software Use Case Diagram

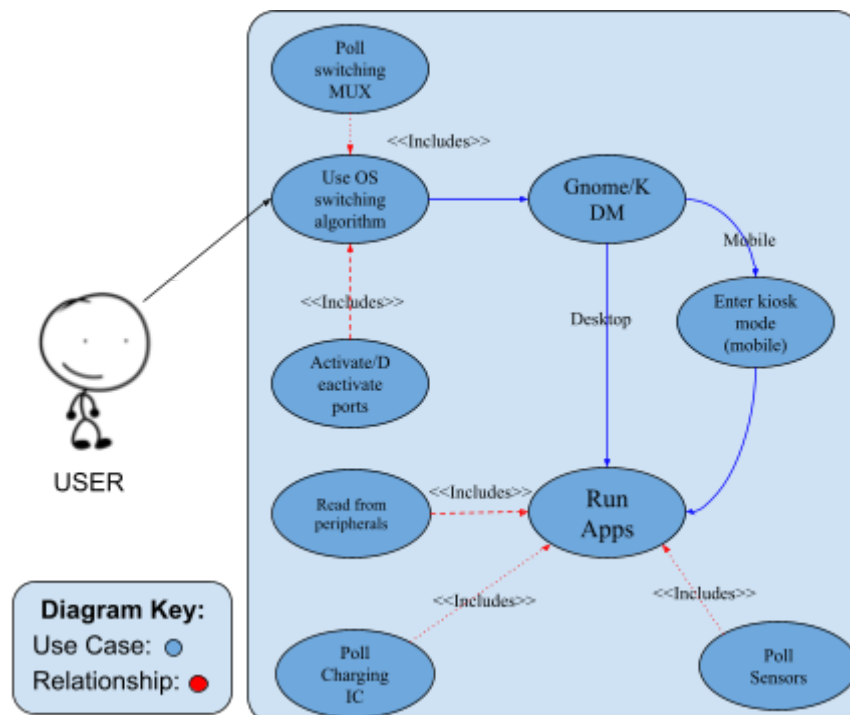


Figure 7.1 Software Use Case Diagram by Authors

The user will have two options with the overall software depending on the power status of the system and the user's preference. Red dotted arrows include actions taken when the event within the blue circle occurs. Solid blue arrows indicate the next step taken in the process, in traditional flowchart style. When the OS switching algorithm is initiated, the poll switching MUX is used to get a read on the current power level of the device at that time. Depending on which OS it is switching to, the ports on the device will be activated or deactivated accordingly. Next, if mobile mode is entered, it will enter an OS (either Gnome or KDM) that will serve as a base that is used to launch the KIPS application. Then, it will begin to run different applications once the main KIPS application is running. When running in desktop mode, this additional OS step is skipped over since the desktop mode runs on a full on traditional OS like a dedicated PC would.

7.2 Software Activity Diagram

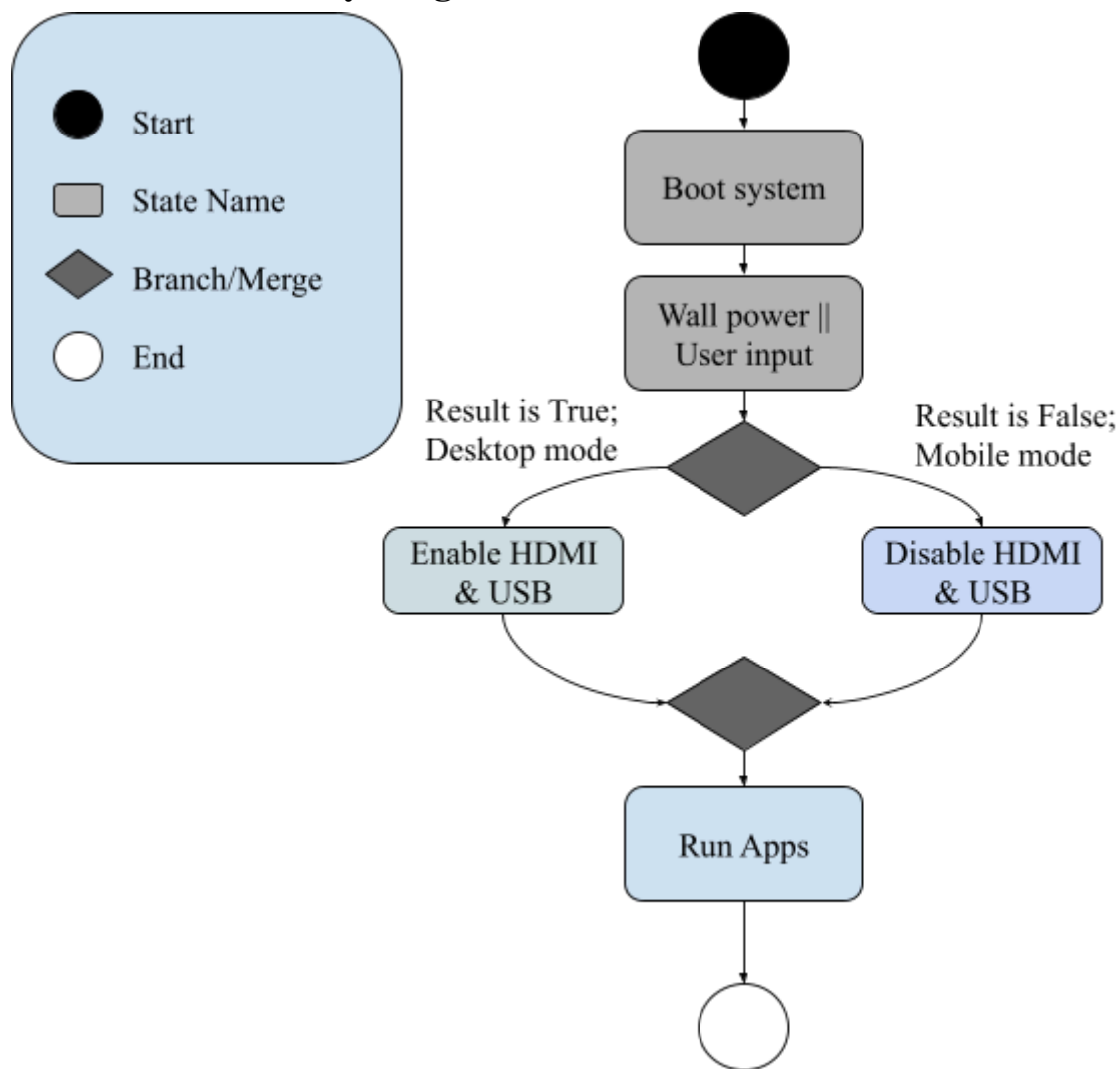


Figure 7.2 Software Activity Diagram by Authors

7.3 Algorithms

Various underlying algorithms are utilized to accomplish all tasks necessary for the smooth and safe operation of KIPS in both mobile and desktop modes. Below we detail a few of such algorithms, including the operating system switching algorithm and a few algorithms to read data from the various sensors included in the sensor suite.

7.3.1 Heartbeat Polling Algorithm

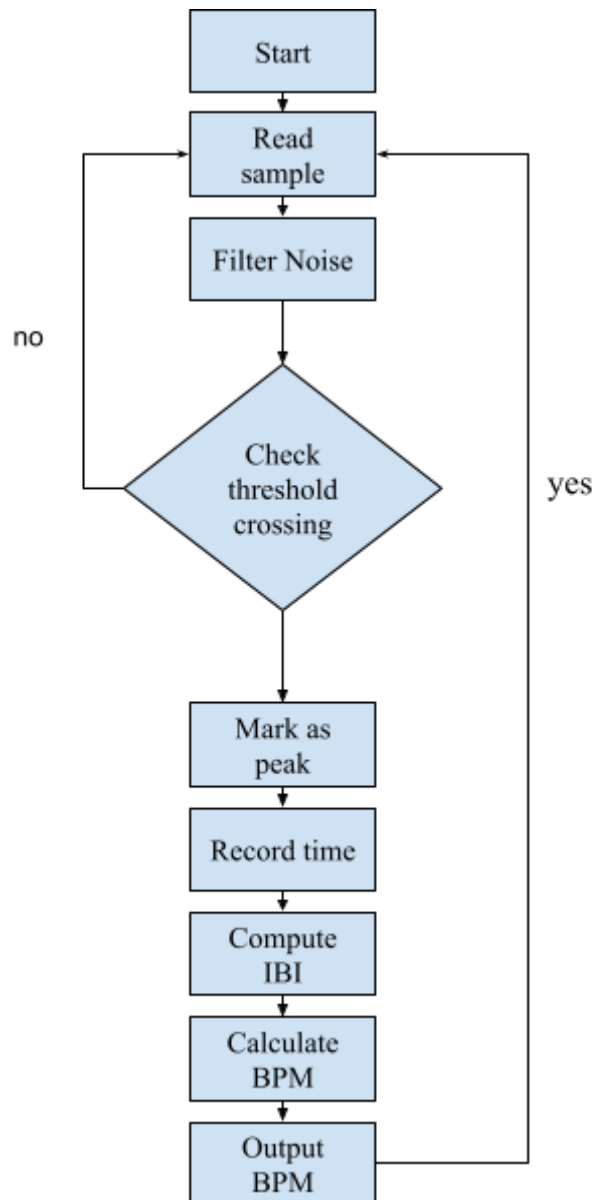


Figure 7.3 Heartbeat Sensor Polling Algorithm by Authors

7.3.2 Temperature and Humidity Polling Algorithm

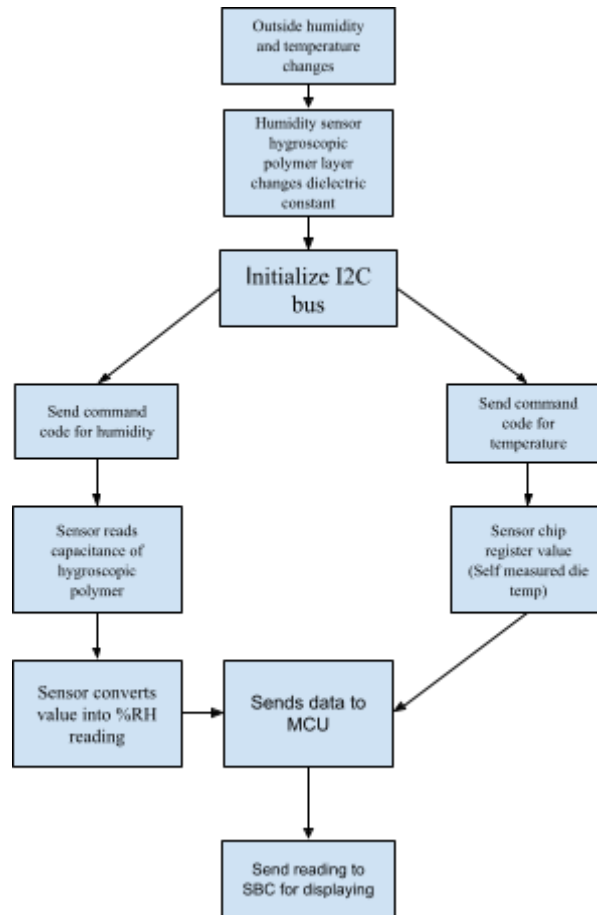


Figure 7.4 Temperature and Humidity Sensor Polling Algorithm by Authors

Although the sensor itself is one component, it contains two separate mechanisms for reading temperature and humidity respectively. For this reason, the diagram splits into two main paths once the I²C bus is initialized. When active, both processes occur simultaneously within the sensor itself.

7.3.3 Battery IC Polling Algorithm

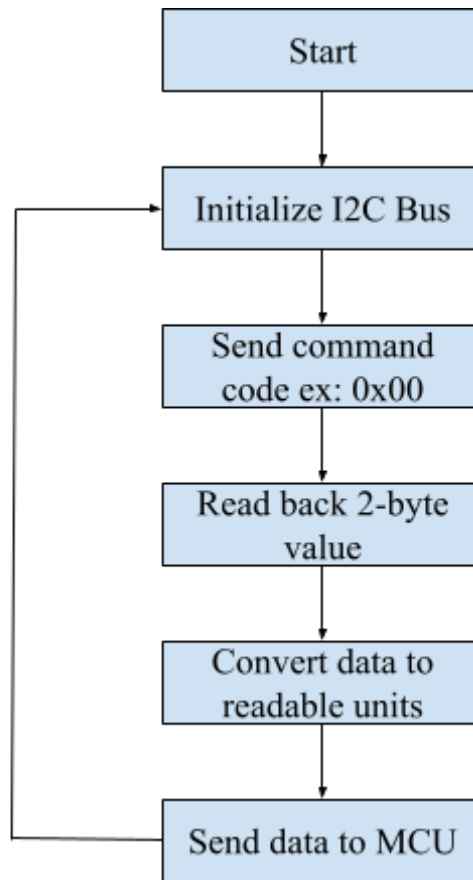


Figure 7.5 Battery IC Polling Algorithm by Authors

7.4 Switching OS Algorithm

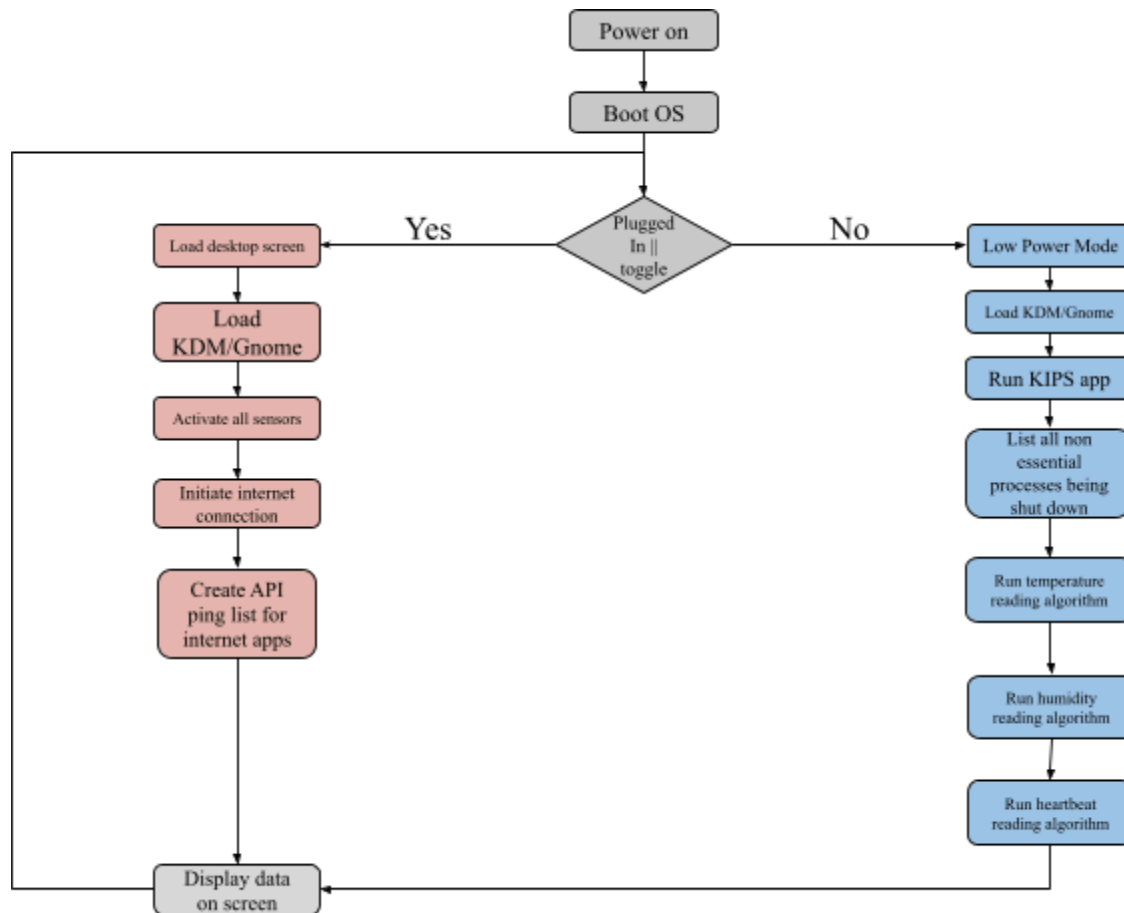


Figure 7.6 OS Switching Algorithm by Authors

The Switching OS Algorithm describes in detail how KIPS will transition between different operating system modes. First, the device is powered on and the OS is booted. The switching IC is then polled in addition to user input to determine if the user should be routed to mobile mode or desktop mode. The result of both of these readings is effectively OR'ed where if either of the inputs is true, the result is true; returns false otherwise. A true result directs the device into desktop mode, whereas a false result leads to mobile mode. In mobile mode, low power mode is activated, the window management software is loaded. The ports are then deactivated while the sensors are activated to read temperature, humidity, power level, and heartbeat information. This information is relayed to the screen for display purposes. The alternative path takes the user through desktop mode where the window management software is launched and necessary sensors are activated. The internet connection is then established and the various necessary APIs are accessed. This data is then relayed to the screen. At the end of each cycle, the charging MUX

is checked repeatedly to observe changes in charging status. If a change is detected, the sequence is repeated with the operating system running.

Due to the nature of how the scheduler assigns tasks to the cores within the Raspbian OS, we were unable to implement the OS switching algorithm as we intended. One of the features within the algorithm was to shut down “all non-essential processes”, under which one such process would be shutting down 2 of the CM4’s 4 cores. During development, we discovered shutting down cores during operation time would cause errors to occur within the CM4’s operating system itself, as we were shutting down cores performing essential processes. However, as we cannot control which tasks the scheduler will delegate to a processor, we can never know which processors, if any, would be performing solely non-essential functions. For this reason, we resolved to work around this issue by instead having the system shut down and reboot while only using 2 cores when switching the OS mode.

7.5 Audio Suite Algorithm

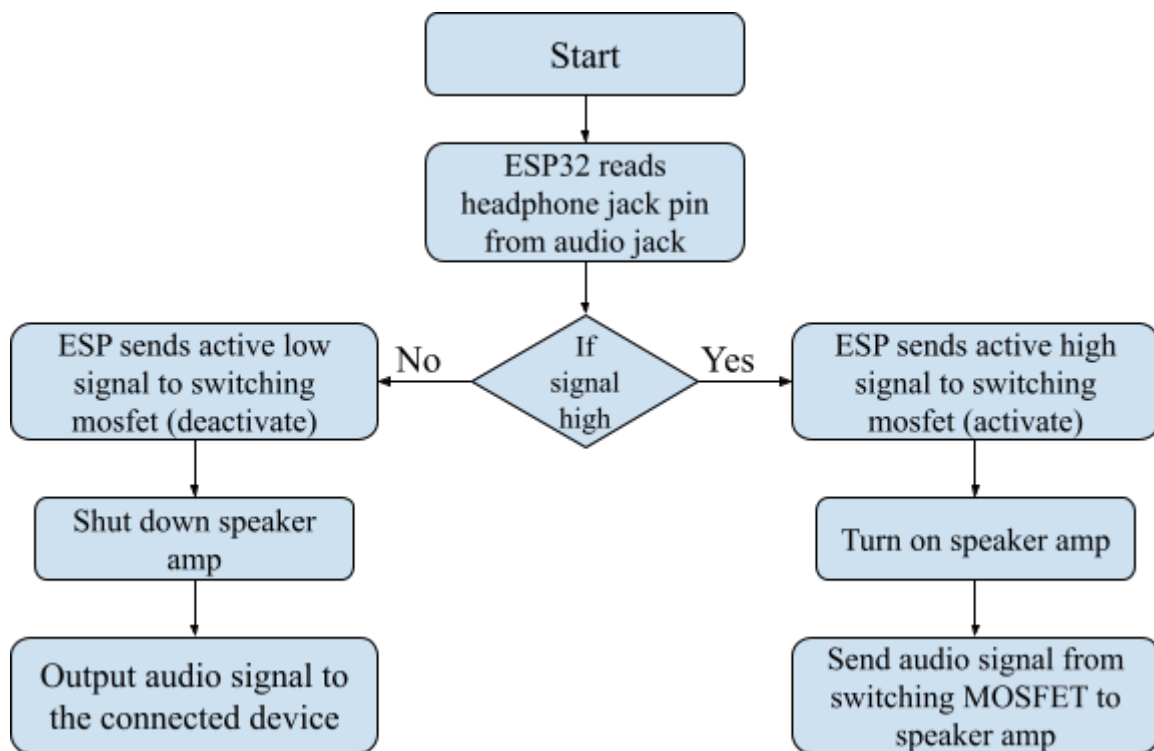


Figure 7.7 Audio Suite Algorithm by Authors

Within KIPS there is an internal speaker. KIPS also contains an audio jack to extend audio to outside sources. In order to reduce feedback, we disable KIPS’s internal speaker so that it does not conflict with the signal output by the external device. In order to perform this process, the

above algorithm is required. It begins by reading the headphone jack pin from the audio jack. From there, depending on whether or not an external audio source is connected the ESP sends an active high or low signal to the switching mosfet respectively. Following the active high signal branch, the speaker amp is turned on and audio from KIPS is played locally on the internal speaker. Following the other branch, we see that the ESP sends an active low signal to the switching mosfet, where the speaker amp is then deactivated and audio from kips is routed to the connected device.

7.6 Display Layout

In order to get a better idea of the visual input of the screen, we designed a rough proof of concept drawing for KIPS in both mobile and desktop modes. It is worth noting that there are a few subtle differences between the mobile and desktop displays.

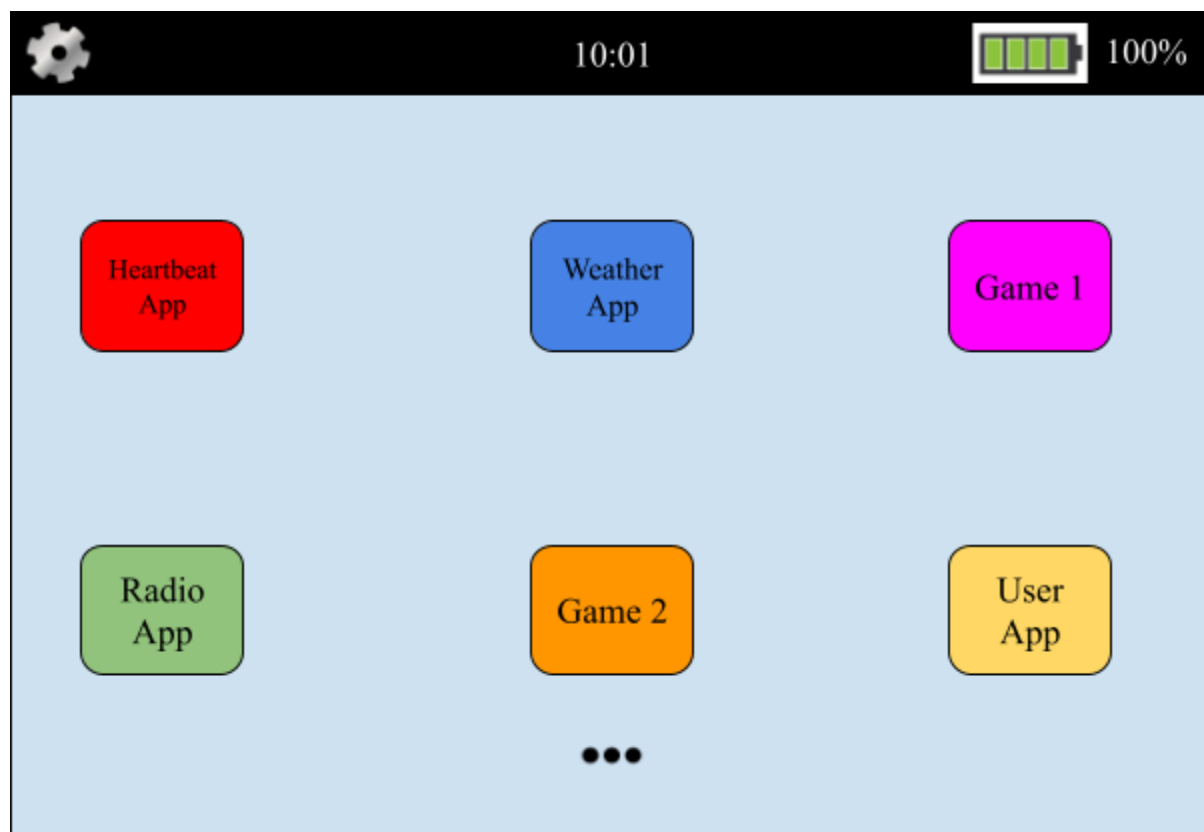


Figure 7.8 Mobile Mode Display Layout

In the mock mobile display, we can see larger icons on the screen with multiple pages that the user can navigate between. We chose this configuration to make the display more touch-friendly, visible on the KIPS screen, and so that it would better resemble application icons on mobile devices. Another notable feature is the bar at the top of the screen. This bar is considered our

information bar and displays the time, battery, and a gear icon which will direct the user to settings when selected.

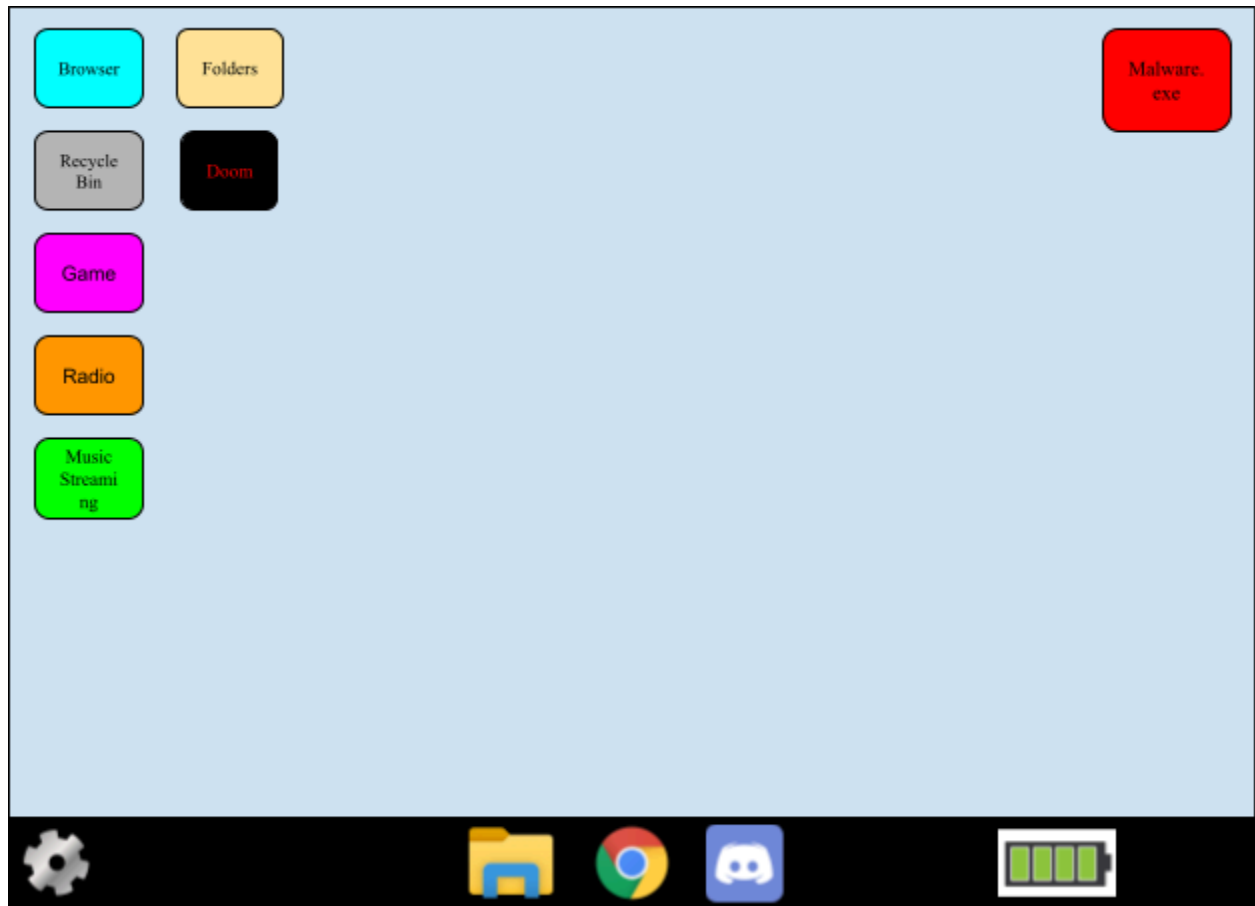


Figure 7.9 Preliminary Desktop Mode Display Layout Design



Figure 7.10 Working Desktop Mode Display Layout

The mock desktop display more accurately mimics a standard desktop computer. As seen in figure 7.X, there is a navigation bar at the bottom of the screen which contains, battery percentage, time, settings, pinned applications, and applications that are currently open. In contrast to the mobile UI, the desktop application icons are much smaller, to accommodate more shortcuts, files, and apps. While this could reduce user visibility, this is moot as KIPS will likely be connected to an external monitor or display via HDMI, which expands this image to a more readable format/display.

8) System Manufacturing

8.1 Motherboard PCB

The PCBs for this project were designed using a combination of Autodesk Fusion and KiCad. This particular PCB serves as the motherboard, which houses the ESP32-S3 WROOM module. It provides the primary connections for most of the daughterboard sensors, including the photoresistor and the temperature and humidity sensor. In addition to sensor interfaces, the motherboard also includes connections for the Digital-to-Analog Converter (DAC) and the control board. Because of this, the motherboard is primarily responsible for handling system logic and input processing. As a result, a large portion of the layout consists of JST-XH connectors that route signals to the various peripheral boards. A large ground plane is

incorporated into the design to improve thermal performance and maintain signal integrity across the board.

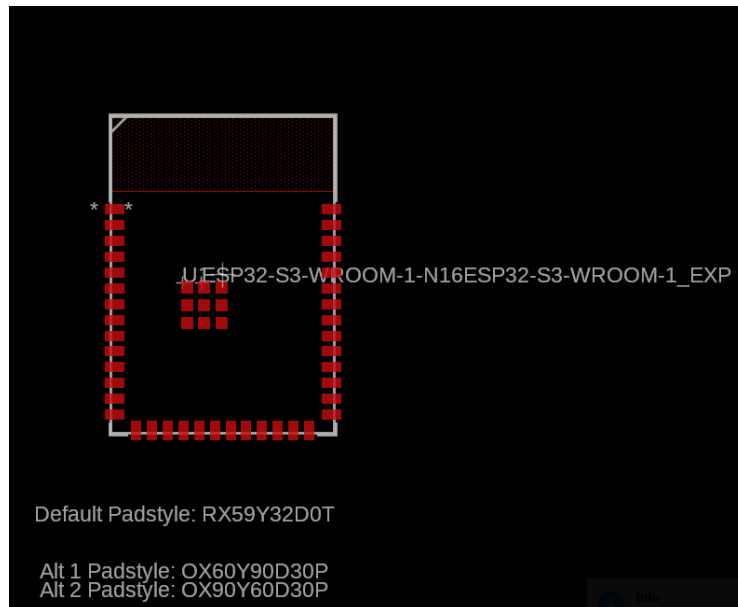


Figure 8.1: 2D Motherboard layout

8.2 High Speed PCB

This board contains two HDMI ports and four USB ports and was one of the most critical designs to get right. It is a four-layer PCB, meaning that any errors on the internal layers cannot be corrected after fabrication. The HDMI interfaces require differential pair routing for the high-speed data lines, demanding extremely tight tolerances—on the order of microns—to maintain signal integrity. Because these lines carry high-frequency data, they must be laid out with great precision and care. Additionally, grounded vias are distributed throughout the board to ensure adequate thermal dissipation and to maintain a solid ground plane for noise reduction and stability.

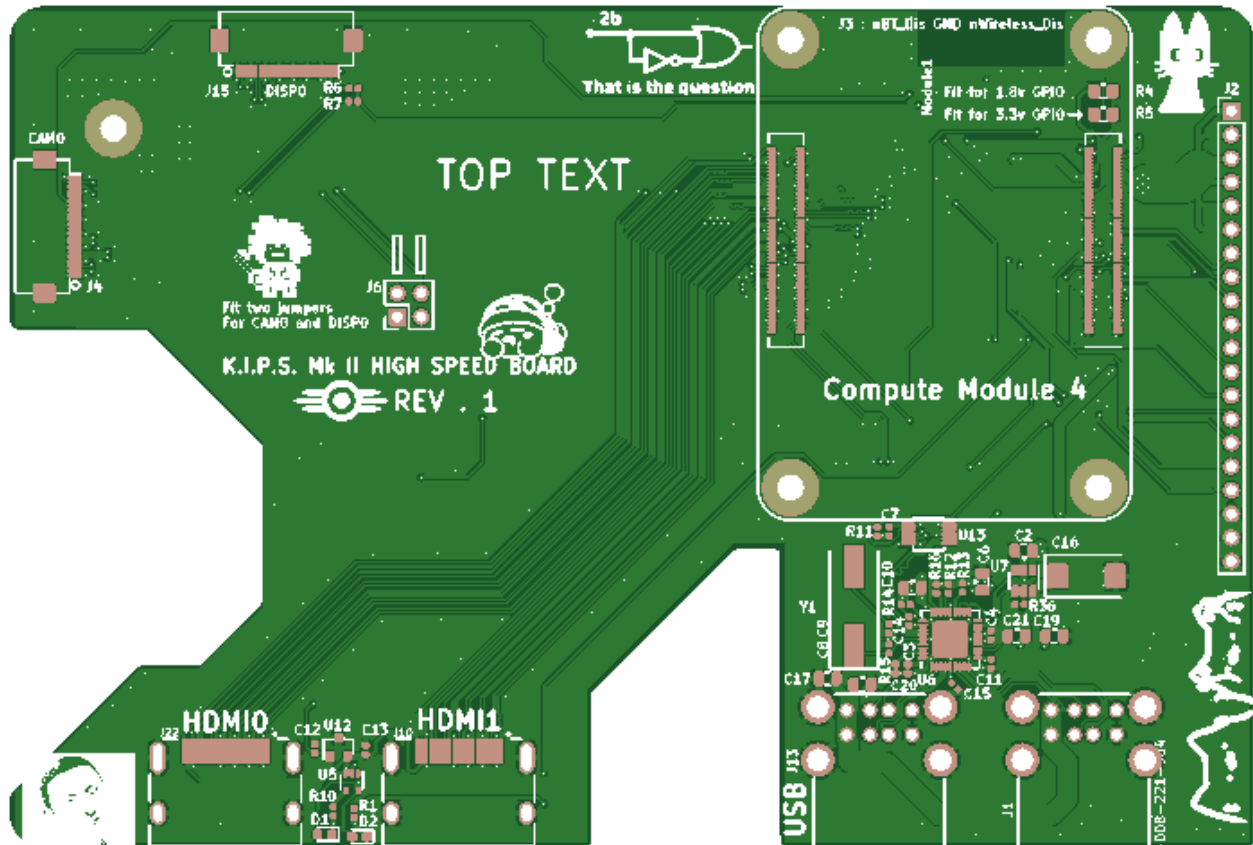


Figure 8.2: 2D High Speed PCB

8.3 Regulator PCB

This is the board for the voltage regulators. One regulator provides 3.3 V at 3 A, while the other supplies 5.2 V at 6 A. This regulator board is critical to the success of the project, as these precise voltage and current levels are required to ensure all components operate at their optimal performance.

The TPS566242 buck regulator is used for both outputs. It is selected because it can deliver the required 6 A at 5.2 V. The 5.2 V output was chosen to compensate for voltage drops across the PCB traces and connectors supplying the Raspberry Pi CM4. Since the CM4 triggers a low-power warning at 4.7 V and ceases operation at 4.6 V, maintaining a nominal 5.2 V ensures stable operation with sufficient overhead. The 6 A current rating also accounts for the load from peripherals such as the USB hub, which contributes significantly to the total current demand.

The 3.3 V, 3 A rail is used to power analog components such as the summing amplifiers. To meet these power requirements and maintain efficiency, the board layout was carefully optimized. Because a 6 A trace would require an extremely wide conductor at 1 oz copper, copper pours were used to handle the high current paths instead. Numerous ground vias were also implemented to reduce impedance and improve thermal performance.

The board uses 2.2 μH and 3.3 μH inductors to handle the required current while maintaining stable operation. In addition, decoupling capacitors are placed as close as physically possible to the ICs to minimize ripple and improve transient response. The design, shown in Figure 8.x, incorporates all of these considerations to achieve maximum efficiency and reliability.

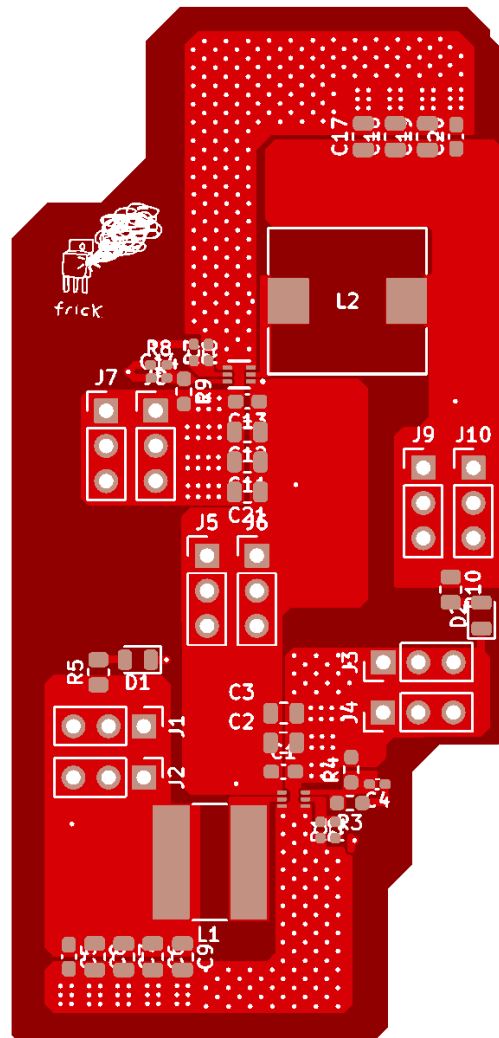


Figure 8.3: 2D Regulator PCB

8.4 Sensor PCBs

These are the two sensors that require daughterboards: the temperature and humidity sensor, and the photoresistor. Both use JST-XH connectors to interface with the main motherboard and include mounting holes for secure attachment.

The temperature and humidity sensor, shown in Figure 8.x, communicates using the I²C protocol through its SCL and SDA lines. The photoresistor is used as a voltage divider that connects to an

ADC input on the Raspberry Pi CM4 to measure light intensity. Both sensors are powered from the 3.3 V rail through their respective JST connectors.

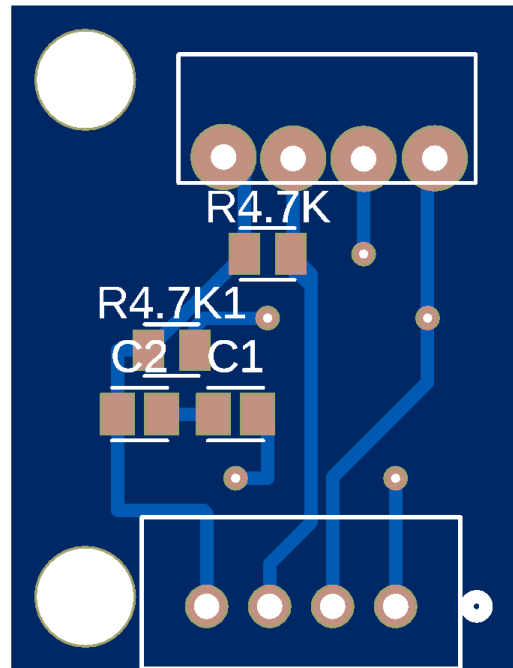


Figure 8.4: Temperature and Humidity sensor Board Layout

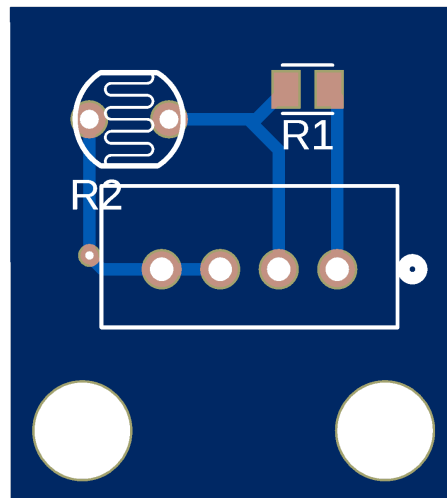


Figure 8.5: Photoresistor sensor Board Layout

8.5 Control PCB

This is the PCB that contains both the D-Pad and also the thumb stick. They both use the JST XH connectors. The D pad has a connector for all of the directions and enter. They each have those 10k resistors which can be seen above, The thumbstick is the large circle seen in figure 8.x. Both the thumbstick and the D pad both need the 3.3V which is what the JST connector is for. GND is the plane below which is what the Vias are for.

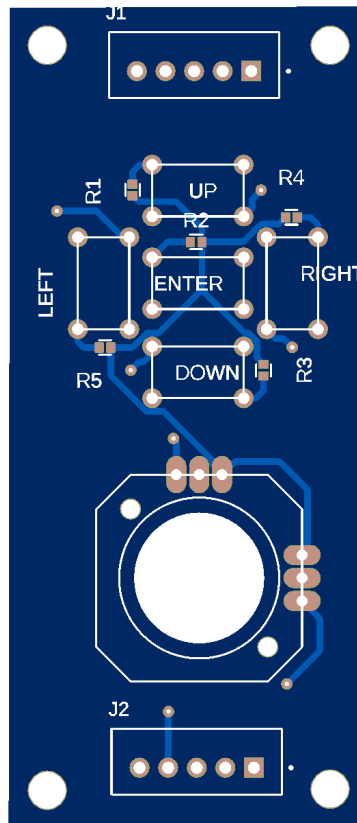


Figure 8.6: Control Board Layout

8.6) Flashlight PCB

This PCB implements the flashlight feature, consisting of an LED controlled by a switch for manual operation. It is powered by the 3.3 V rail, supplied through a JST-XH connector. As shown in Figure 8.X, the connector is positioned on the opposite side of the LED to allow the light to project outward. The PCB has been designed with a wider layout to improve heat dissipation during operation. The current series resistor is set to 10 Ω , though it can be adjusted to 15 Ω if necessary to optimize brightness and power consumption.

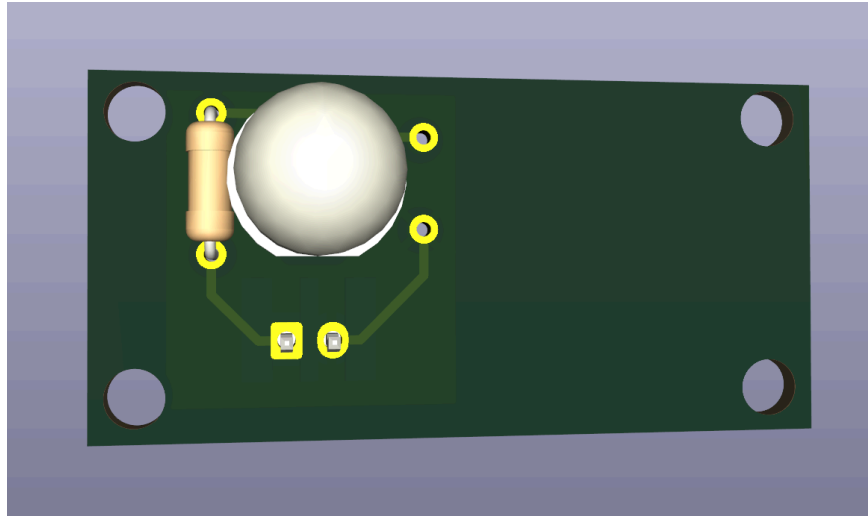


Figure 8.7: 3D Model of Flashlight PCB Frontside

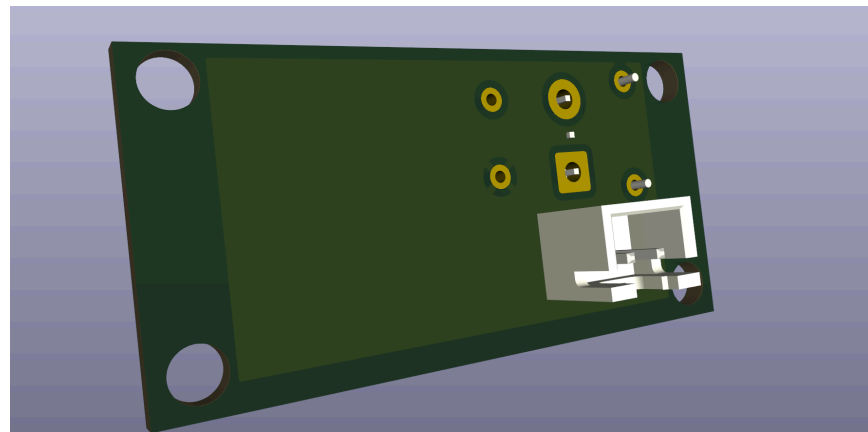


Figure 8.8: 3D Model of Flashlight PCB Backside

9) Prototype and System Testing

9.1 Hardware

9.1.1) High Speed Board Testing

This high-speed board is designed to interface with the Raspberry Pi Compute Module 4 (CM4). To operate correctly, it first requires a stable power source, provided by the power supply shown here. The supply outputs 5.2 V to compensate for voltage drops across the board. The board connects to the Pi through the bottom pin headers and, for testing purposes, was assembled on a breadboard using Dupont cables for all necessary connections. There is also a monitor and mouse for testing peripherals.

After completing the wiring, the board was connected to an HDMI output device—in this case, a monitor—to verify video signal integrity and confirm proper data connections. A lack of signal would have indicated a major configuration issue, while a distorted or intermittent signal would suggest only minor wiring problems. However, the test produced a stable video output with no low-power warnings from the Pi, drawing 0.2 A, confirming both correct board configuration and accurate voltage calculation. Next, the USB functionality was tested using a mouse, as shown below. The mouse operated as expected, with current draw increasing to 0.4 A during use. This verified that the board is functioning normally and that all intended features are working correctly.

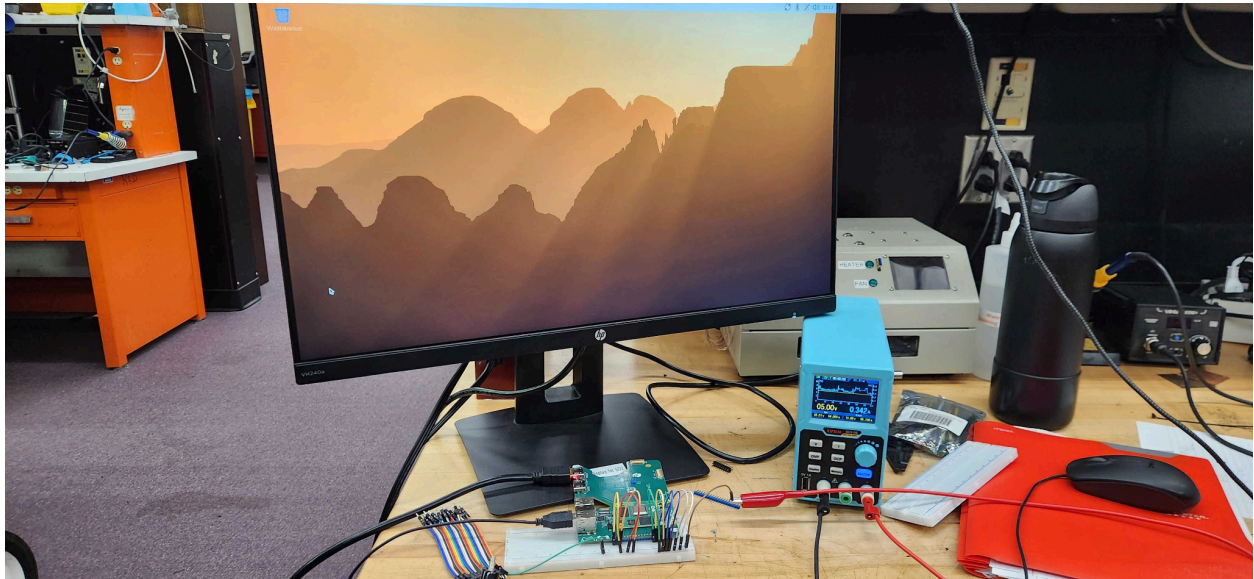


Figure 9.1: High Speed Board Testing

9.1.2) Regulator Board Testing

This is the regulator board, which contains two TPS566242 buck regulators. One is configured to output 3.3 V at 3 A, and the other provides 5.2 V at 6 A. The board was initially tested by connecting it to a power supply set to 7.6 V at 4 A, simulating both the battery input and the AC–DC rail conditions. A simple visual check was used to confirm regulator functionality via onboard LEDs. The series resistors for these LEDs were calculated for the expected output currents—200 Ω for the 5.2 V rail and 86 Ω for the 3.3 V rail. Since the LEDs illuminated brightly without failing, this indicated that the regulators were producing the expected voltages. The 5.2 V rail functioned correctly on the first test, but the 3.3 V rail did not, due to a missing PCB trace that should have connected two capacitors between the EN and VIN pins. Once this trace was corrected, the 3.3 V regulator operated as intended. This was verified using a multimeter, as shown in Figure 9.x. Next, the regulators were tested under load. While the board generally performed correctly, thermal issues caused the 5.2 V regulator to shut down at around 5.5 A. This will be addressed by adding a heatsink to improve thermal performance.

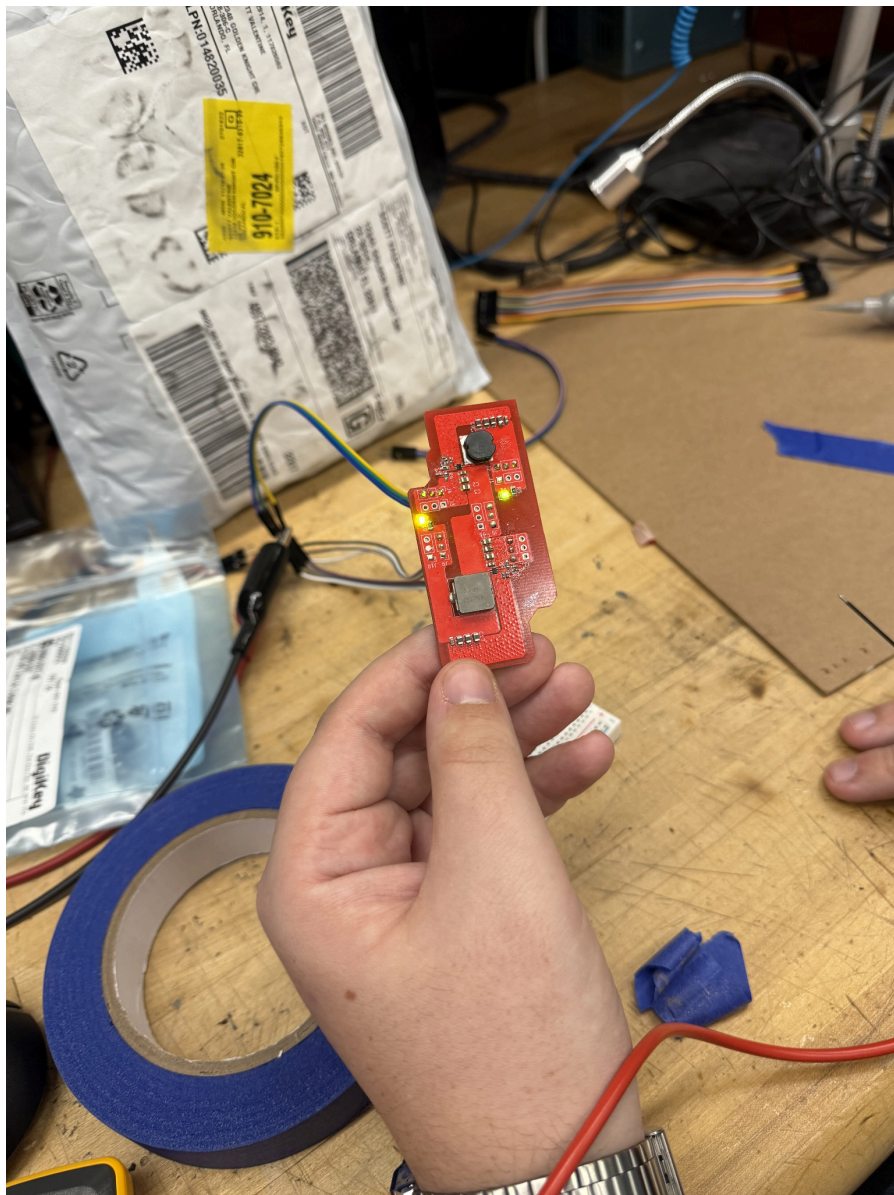


Figure 9.2: Regulator Board Testing

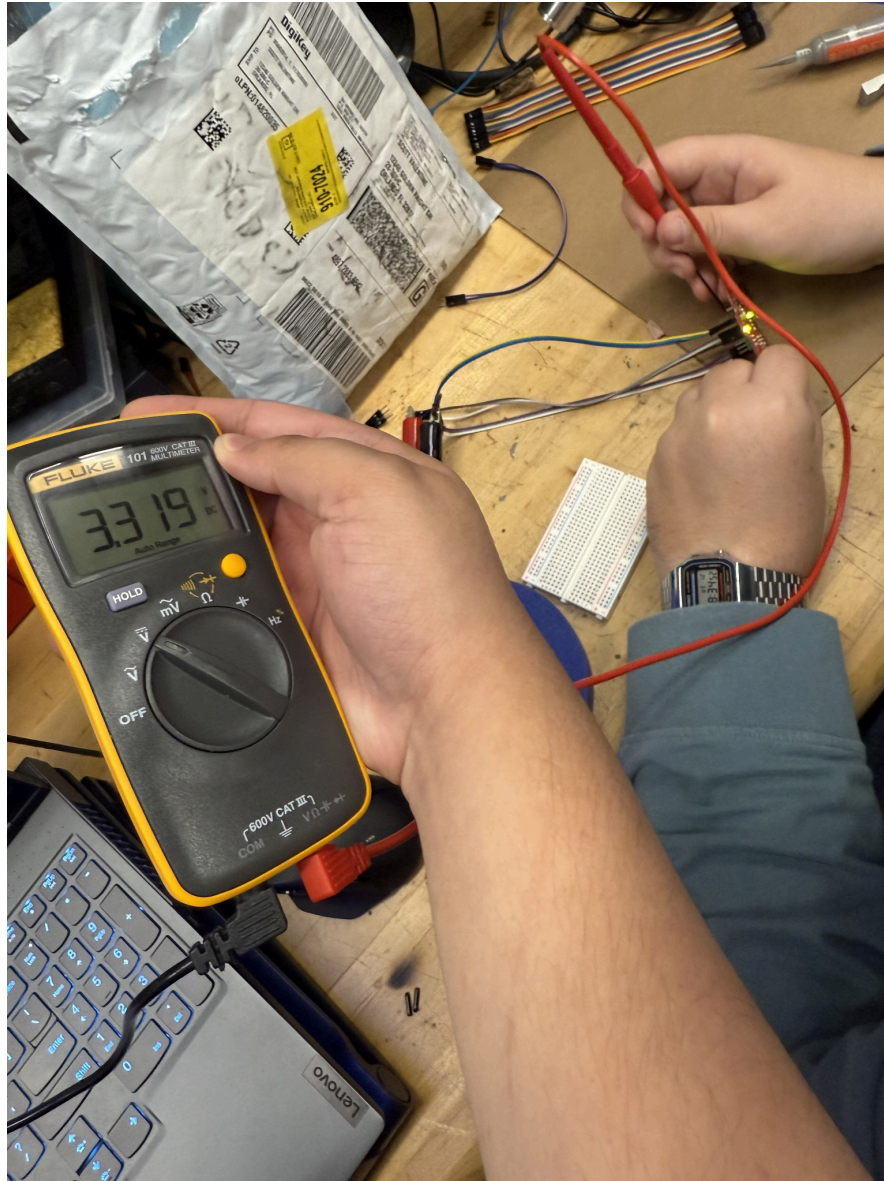


Figure 9.3: Regulator Board Testing 2

9.1.4) Camera Testing



Figure 9.4: Camera Testing

9.2 Software

Testing the software consisted of several different stages. In the first stage, we began by creating a simple guided user interface (GUI) to be used in mobile mode. To formulate the GUI itself we used Glade in conjunction with a Python runner script. For testing we created a rough draft of the mobile mode GUI in Python for readability purposes and subtle tweaks. We plan to convert this to C++ later once we are able to refine the interface, however for this stage of development we are using Python.

Ultimately, as we continued development on this project, we determined the best use of Python was for the CM4 within the GUI. This was split between using Python for higher level programming within the GUI, and as a method of integrating Glade with the GUI Python program functions. Meanwhile C++ remained as the language of choice for lower level, backend programming. This was what was used for all peripheral sensor and radio backend programming.

9.2.1) Sensor Board Testing

This section covers the testing of the temperature and humidity sensor. The sensor board, shown in Section 6, includes 4.7 kΩ pull-up resistors for the I²C lines. Initial testing was inconclusive because the board was assembled with incorrect resistor values. After correcting this issue, the board was ready for functional testing. The sensor was programmed and tested using the available Adafruit libraries. While this approach is sufficient for initial validation, the final project will use a more refined and optimized codebase. As shown in Figure 9.x, the I²C communication was successful, and the sensor returned accurate and readable data. This confirms that the sensor board is fully operational and ready for integration into the final system.

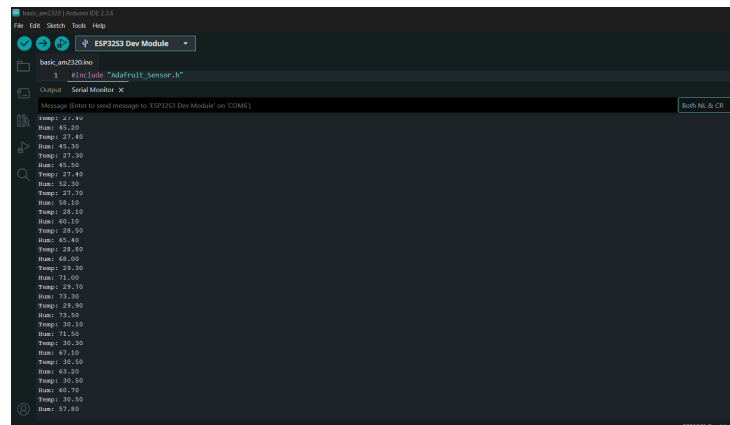


Figure 9.5: Temperature and Humidity Sensor Testing

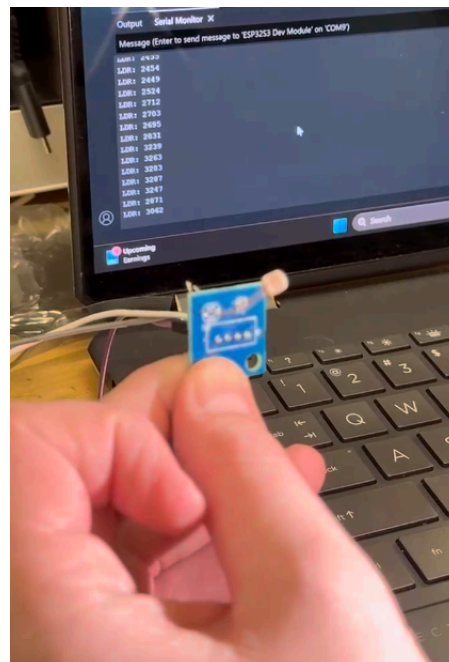


Figure 9.6: LDR Testing

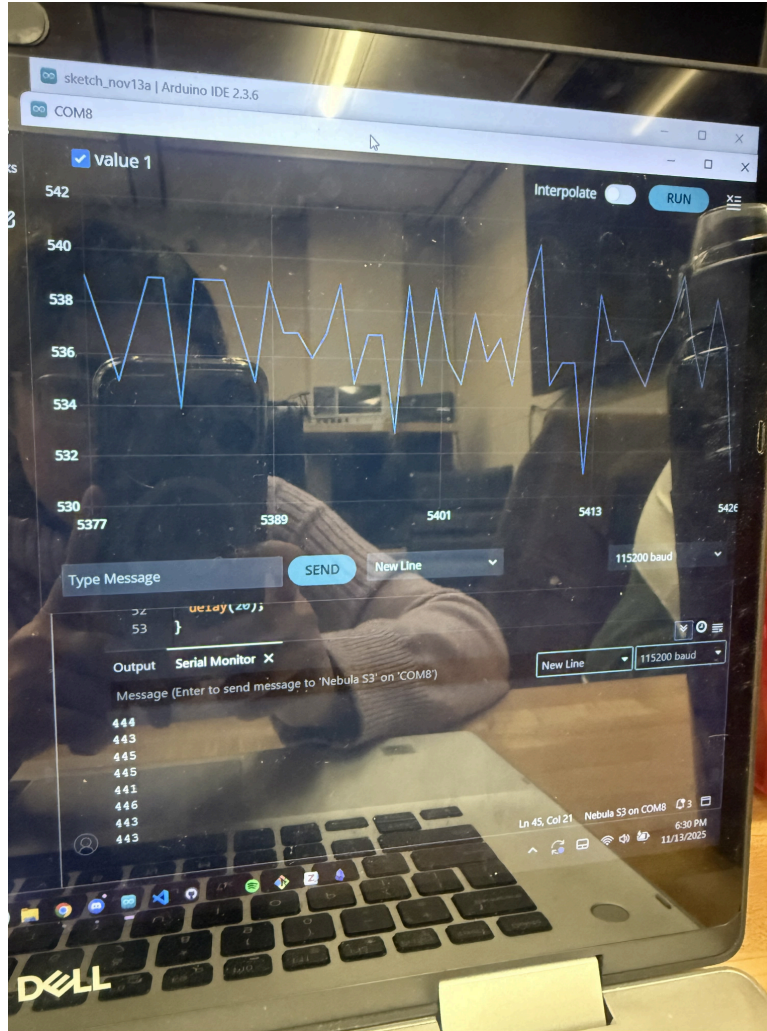


Figure 9.7: Heartbeat Sensor Testing

Combining all previous sensor tests into a single program was necessary to ensure that every sensor not only functions correctly on its own but also operates reliably alongside the others as a unified system. This was accomplished by integrating the previously validated code for each individual sensor. A baud rate of 115200 was selected to provide a stable and efficient communication speed for all sensors. Using this combined codebase, we verified that all sensors were able to output data simultaneously over I²C, as shown in Figure 9.x. Each sensor was then individually tested within the combined setup: a flashlight was used to validate the LDR readings, the heartbeat sensor was tested on a finger, and the temperature and humidity sensor readings were also confirmed. Once all outputs were verified to be accurate and stable, this demonstrated that the full sensor suite can run concurrently without conflict. This configuration is therefore suitable for implementation on the final board.

```

Temp: 24.90
Hum: 52.70
1060 1055 5 0
LDR Value: 373
Temp: 24.90
Hum: 52.50
1059 1055 4 0
LDR Value: 61
Temp: 25.00
Hum: 54.90
1055 1055 0 0
LDR Value: 11
Temp: 25.30
Hum: 59.60
1056 1055 1 0
LDR Value: 24
Temp: 25.60
Hum: 60.10
1059 1055 4 0
LDR Value: 13
Temp: 25.80
Hum: 60.50
1054 1054 0 0

```

Figure 9.8: All Sensors in One Print Testing

9.2.2) ESP32 to Pi CM4 Testing

Up to this point, all sensor testing had been performed through the Arduino IDE using the ESP32. However, the final system will not output data this way; instead, all sensor data must be received and processed by the Raspberry Pi CM4. To achieve this, the sensor data transmitted by the ESP32 must be delivered to the Pi through UART. For testing, the ESP32's UART pins were connected to the CM4 using Dupont wires. Initially no communication was detected between the ESP32 and the Pi. Through debugging, it was determined that the issue stemmed from the Pi's Bluetooth service, which occupies UART0 by default. Once Bluetooth was disabled, the Pi successfully received data from the ESP32. Because disabling Bluetooth is not a viable long-term solution, an alternative configuration was found that allows both Bluetooth and UART0 to operate simultaneously. As shown in Figure 9.x, the communication link was successfully established, with the ESP32 transmitting data that appears correctly within the Pi environment.



Figure 9.9: ESP32 to Pi CM4 Testing

10) Administrative Content

10.1) Financing and Budgeting

Table 10.1: Budget

Categories	Budget
PCBs/Electrical Parts	~\$1000
Materials	~\$300
TOTAL	~ \$1300

Due to having to reorder boards and buy additional parts, the cost estimate ended up being closer to \$2,000 dollars total.

10.2) Bill of Materials

Table 10.2: Bill of Materials

ITEM	AMOUNT	COST PER UNIT	ESTIMATED TAX	ESTIMATED SHIPPING	TOTAL
Raspberry Pi Compute Module 4, Wireless, 8GB RAM, 16GB Flash	2	\$80.00	\$10.40	\$5.32	\$175.72
Mini Base Board (B) Designed for Compute Module 4	2	\$27.95	\$4.33	\$5.32	\$65.55
ESP-32 S3 Development Boards (3 Pack)	1	\$17.99	\$1.17	\$0.00	\$19.16
OSOYOO 3.5 Inch Capacitive Touch Screen	1	\$42.99	\$2.79	\$0.00	\$45.78
Heartbeat Sensor	1	\$16.00	\$1.05	\$0.64	\$17.69
Mini D-Pad	1	\$2.18	\$0.15	\$0.64	\$2.97
PCM5102 DAC	2	\$3.97	\$0.53	\$0.64	\$9.11
Pi Camera Module 3	1	\$25.00	\$1.64	\$0.64	\$27.28
SI4735 AM/FM Module	2	\$5.30	\$0.70	\$0.64	\$11.94
LM4610 Headphone Amp	3	\$1.80	\$0.36	\$0.64	\$6.40
PAM8301 Speaker Amp	4	\$0.42	\$0.12	\$0.64	\$2.44
3.5mm Headphone Jack	1	\$1.00	\$0.08	\$0.64	\$1.72
Speaker	1	\$3.77	\$0.26	\$0.64	\$4.67

BQ34Z100-G1 Battery Fuel Gauge	2	\$4.64	\$0.61	\$0.64	\$10.53
Ferrite Rod (Loop Antenna)	2	\$2.65	\$0.35	\$0.64	\$6.29
Temp/Humidity Sensor	1	\$3.95	\$0.26	\$1.80	\$6.01
Mini Joystick	1	\$2.50	\$0.16	\$1.80	\$4.46
Dual Potentiometer	1	\$2.75	\$0.18	\$1.80	\$4.73
Assorted Breakout Boards (Multiple in one order)	1	\$26.25	\$1.71	\$5.39	\$33.35
JLCPCB High Speed Board	1	\$269.19	\$137.61	\$37.42	\$444.22
JLCPCB Daughter Boards/Regulator	1	\$21.20	\$19.42	\$38.95	\$79.57
Digikey Regulator Parts/Other Parts	1	\$38.29	\$4.19	\$6.99	\$49.47
PEBA Crimping Tool and Dupont Connector Kit	1	\$30.99	\$2.01	\$0.00	\$33.00
ELEGOO 120pcs Multicolored Dupont Wire	1	\$6.98	\$0.45	\$0.00	\$7.43
COMDEX USB A to USB C Data Transfer Cables	1	\$8.69	\$0.61	\$0.00	\$9.30
Pastall 6 Pack Raspberry Pi Camera Cable 15 Pin Ribbon Cable	1	\$6.99	\$0.49	\$0.00	\$7.48

Raspberry Pi Zero Camera Cable Ribbon 11.8 inch Extension Cable	1	\$4.99	\$0.35	\$0.00	\$5.34
Solder Wick Braid	1	\$6.95	\$0.45	\$0.00	\$7.40
4 Pack 2mm Extended Needle Test Probes	1	\$12.99	\$0.91	\$0.00	\$13.90
2 Rolls 10mm X 33m 108ft Heat Tape	1	\$7.99	\$0.56	\$0.00	\$8.55

10.3) Distribution of Work Table

Table 10.3: Distribution of Work Table

Scott Valentine - Electrical Engineer	PCB design, Project lead.
Harrison Mills - Computer Engineer	PCB design, software design.
Jonathan Meyers - Computer Engineer	UI design, sensor programming, software design.
Jessenia Argueta - Computer Engineer	Sensor programming, UX design, software design.

10.4) Milestones

Table 10.4: Milestones

Milestone Number	Milestone Description	Start Date	Anticipated End Date	Dependencies & Requirements
1	Finalize Project Idea	5/10	5/10	Approval from Dr. Chan
2	Finalize general Design Choices	5/10	5/12	Approval from Dr. Chan
3	D&C Finalization	5/10	5/14	Approval from Dr. Chan
4	Begin research	5/14	8/20	Approval from Dr. Chan

11) Conclusion

The development of the KIPS Mk. II represents a major overhaul in transforming the initial idea of a wearable wrist-mounted computing device into a robust and versatile real-life device. Expanding upon the baseline laid by the Mk. I prototype, Mk. II integrates a sophisticated combination of mobile and desktop computing modalities including an enhanced sensor suite and a carefully structured hardware architecture built around the ESP32-S3 and Raspberry Pi Compute Module 4. Through extensive research, multiple iterations of design modifications, and disciplined implementation, the completed system demonstrates the viability of a hybrid wearable–desktop device that adapts to different operating environments and user needs on command.

As outlined in the executive summary, this project showcases a wide array of technical accomplishments from advanced power-mode switching to comprehensive sensor integration and robust peripheral support, clearly establishing the Mk. II as a significant improvement over the original system. The documentation of standards, design constraints, schematics, software and hardware architecture, manufacturing processes, and testing results highlights the interdisciplinary nature of the work and the careful ethical and collaborative considerations that guided the development process.

This project can best be divided into two complete partitions in the software and hardware sections respectively. In order to accommodate for the more physically constrained nature of the project, various PCBs needed to be designed to deliver proper performance while still being able to fit within the KIPS Mk. II case. To this end, many daughter boards were made for components

such as the regulators, sensors, flashlight, etc in order to create a more modular design. This allows for ease of design and easy replacement if a component were to fail. The most essential and difficult to design board among these was the high speed PCB, as it needs to maintain the integrity of high frequency data transfers while remaining a reasonable temperature within the case. Despite the significant thermal and spatial constraints, the final hardware design balances compactness with performance, enabling reliable communication between subsystems, stable sensor operation, and consistent power regulation under varying loads.

Within this project, we required the use of various different softwares and development environments which were met with a few standards and limitations. By leveraging the use of C++ as our primary programming language, we were able to achieve efficient execution times and real-time response across both desktop and mobile mode for sensor polling and fast UI rendering. Additionally the usage of different software frameworks and libraries (Linux-based development tools) and version-control platforms allowed for the team to manage revisions more effectively and help to coordinate GUI and backend development. Also, by enforcing rigorous coding standards and modular design principles, we ensured that the system would remain scalable and maintainable for future software expansion or modification. Overall, the software structure was shaped by the need for a reliable and maintainable interface across both modalities in addition to the defined software goals.

This project emphasizes the need for extensive testing and evaluation to ensure a final working product. Throughout the course of this project, various testing sessions were held to ensure that all systems functioned and communicated properly. Through these testing sessions, we found that certain listed specifications and documentation were slightly off or outdated, leading us to rely on other, multiple external sources to verify the details outlined in the source documentation. We first tested individual components during prototype system testing, which was done by breadboarding the PCBs while running basic essential functions, then moving on to connecting the components together to ensure they worked in tandem, for the hardware. The software was then tested by creating our own small test programs or using premade programs from the manufacturer to test different component functionalities separately, then in conjunction with the SBC and MCU.

While KIPS Mk. II successfully improves upon KIPS Mk. I to a substantial degree, further improvements could still be realized in future implementations. The power consumption and thermal range of operation are critical factors for a personal-use device, especially in cases where the device makes direct skin-contact for prolonged periods of time. Further reductions in the thermal load, power density and overall energy consumption would benefit user safety, comfort, and also the operating life of the device itself. Additionally, improvements in component placement, airflow, and more efficient heat dissipation methods could be utilized to address component cooling concerns and prevent the device from overheating. One last consideration for

future implementations/improvements is the integration of USB 3.0 for more flexibility and utility from peripheral devices in addition to the inclusion of an SD card for expanded storage. If a KIPS Mk. III were to be designed, the integration of these features would be heavily considered.

The KIPS device is intended to be an alternative and a middle-ground to standard mobile and desktop technologies. In using the device, the user will be able to have access to essential features found in both technologies, all while on the go and being wearable, eliminating the need to carry the device on them in some form. Additionally, the device's extensive customization options to add applications according to user needs increases its versatility and convenience. In designing and constructing the KIPS Mk. II, it is not only meant to be a significant improvement upon its proof of concept (KIPS Mk. I), but also allow its users a level of functionality not commonly found in mobile and desktop devices. The KIPS Mk. II features address the issue of accessibility and common troubles of having to switch between a mobile and desktop device in order to access operations exclusive to those devices. This report documents the technical growth and collaborative effort that enabled the team to achieve this goal.

APPENDIX A: REFERENCES

- [1] Raspberry Pi, "Raspberry Pi 5 Product Brief." Dec. 01, 2025. Accessed: Dec. 06, 2025. [Online]. Available: <https://pip-assets.raspberrypi.com/categories/892-raspberry-pi-5/documents/RP-008348-DS-3-raspberry-pi-5-product-brief.pdf?disposition=inline>
- [2] Raspberry Pi, "Raspberry Pi Compute Module 5 Product Brief." Raspberry Pi, May 2025. Accessed: Aug. 11, 2025. [Online]. Available: <https://datasheets.raspberrypi.com/cm5/cm5-product-brief.pdf>
- [3] Raspberry Pi, "Raspberry Pi Zero 2W Product Brief." Apr. 01, 2024. Accessed: Dec. 06, 2025. [Online]. Available: <https://datasheets.raspberrypi.com/rpizero2/raspberry-pi-zero-2-w-product-brief.pdf>
- [4] Radxa, "ROCK Pi 4C Plus Product Brief." Mar. 04, 2022. Accessed: Dec. 06, 2025. [Online]. Available: https://rokland.com/support/manuals/Rock_4C_Plus-Datasheetv1.pdf
- [5] Radxa, "Radxa ROCK Pi S0." Sept. 29, 2024. Accessed: Dec. 06, 2025. [Online]. Available: https://dl.radxa.com/rockpis0/radxa_rock_pi_s0_product_brief.pdf
- [6] Beagleboard, "PocketBeagle 2," beagleboard.org. [Online]. Available: <https://www.beagleboard.org/boards/pocketbeagle-2>
- [7] Beagleboard, "BeagleBone Black," beagleboard.org. [Online]. Available: <https://www.beagleboard.org/boards/beaglebone-black>
- [8] BeagleBoard.org Foundation, "PocketBeagle 2 Datasheet." May 14, 2025. Accessed: Aug. 20, 2025. [Online]. Available: <https://docs.beagleboard.org/pocketbeagle-2.pdf>
- [9] BeagleBoard.org Foundation, "BeagleBone Black Datasheet." May 14, 2025. Accessed: Aug. 20, 2025. [Online]. Available: <https://docs.beagleboard.org/beaglebone-black.pdf>
- [10] Raspberry Pi, "Raspberry Pi Computer Hardware." Accessed: Dec. 05, 2025. [Online].

Available: <https://www.raspberrypi.com/documentation/computers/raspberry-pi.html>

[11] Raspberry Pi, “Keyboard Computers.” Accessed: Dec. 06, 2025. [Online]. Available: <https://www.raspberrypi.com/documentation/computers/keyboard-computers.html>

[12] Raspberry Pi, “Raspberry Pi Compute Module 4 Datasheet.” Raspberry Pi, Nov. 24, 2023. Accessed: Aug. 11, 2024. [Online]. Available: <https://datasheets.raspberrypi.com/cm4/cm4-datasheet.pdf>

[13] Raspberry Pi, “Raspberry Pi Compute Module 4S Datasheet.” Raspberry Pi, Apr. 29, 2024. Accessed: Aug. 11, 2024. [Online]. Available: <https://datasheets.raspberrypi.com/cm4s/cm4s-datasheet.pdf>

[14] Raspberry Pi, “Raspberry Pi Compute Module 5 Datasheet.” Raspberry Pi, Nov. 27, 2024. Accessed: Aug. 11, 2024. [Online]. Available: <https://datasheets.raspberrypi.com/cm5/cm5-datasheet.pdf>

[15] Raspberry Pi, “Raspberry Pi Compute Module 4 Product Brief.” Raspberry Pi, May 2025. Accessed: Aug. 11, 2025. [Online]. Available: <https://datasheets.raspberrypi.com/cm4/cm4-product-brief.pdf>

[16] Raspberry Pi, “Raspberry Pi Compute Module 4S Product Brief.” Raspberry Pi, Apr. 2024. Accessed: Aug. 11, 2025. [Online]. Available: <https://datasheets.raspberrypi.com/cm4s/cm4s-product-brief.pdf>

[17] “ESP32: The Ultimate Guide for Beginners and Enthusiasts.” [Online]. Available: <https://www.electronicsforu.com/technology-trends/esp32>

[18] “Exploring the ESP32 versions. Differences and similarities,” ESPBoards. [Online]. Available: <https://www.espboards.dev/blog/esp32-soc-options/#:~:text=AI%2DPowered,S2%20is%20the%20better%20choice.>

[19] “Raspberry Pi product series explained,” Raspberry Pi. [Online]. Available: <https://www.raspberrypi.com/news/raspberry-pi-product-series-explained/>

[20] “RP2040,” Raspberry Pi. [Online]. Available: <https://www.raspberrypi.com/documentation/microcontrollers/silicon.html#rp2040>

[21] “Nordic Semiconductor nRF5340 Multicore System on Chip (SoC),” Mouser Electronics. [Online]. Available: <https://www.mouser.com/new/nordic-semiconductor/nordic-nrf5340-soc/>

[22] “nRF51, nRF52, and nRF53: A Guide to Nordic Chips,” Hardfault.

[23] “nRF53 Series SoC & Kits.” [Online]. Available: <https://www.crowdsupply.com/nordic/nrf53-series-soc-and-kits>

[24] “STM32U5 Series,” STMicroelectronics. [Online]. Available: <https://www.st.com/en/microcontrollers-microprocessors/stm32u5-series.html>

[25] “STMicroelectronics — STM32U5 Ultra-Low-Power Microcontrollers (MCUs).” [Online]. Available: <https://www.futureelectronics.com/np/stmicroelectronics-stm32u5-microcontrollers>

[26] “ESP32S3WROOM1N16R8 Datasheet.”

[27] “pico-w-datasheet.”

[28] “nRF5340 SoC PB.”

- [29] “STM32U585xx Family Data Sheet.”
- [30] “MAX30102--High-Sensitivity Pulse Oximeter and Heart-Rate Sensor for Wearable Health”.
- [31] “The Original PulseSensor Kit,” PulseSensor. [Online]. Available: <https://pulsesensor.com/products/pulse-sensor-amped>
- [32] “PulseSensor Datasheet 2024.11”.
- [33] “Gravity: PPG Heart Rate Monitor Sensor for Arduino (Analog/Digital),” DFRobot. [Online]. Available: <https://www.dfrobot.com/product-1540.html?srltid=AfmBOopMBRqWj2WSNYNKnXq4TrVMxf12lawFPpo5b7v7m969jZngna5C>
- [34] “HEARTBEAT_SEN-11574_SparkFun.”
- [35] “SON1303 Data Sheet.”
- [36] A. Pini, “The Basics of Photodiodes and Phototransistors and How to Apply Them.” [Online]. Available: <https://www.digikey.com/en/articles/the-basics-of-photodiodes-and-phototransistors-and-how-to-apply-them>
- [37] “KIPS MKII Sensors GPT Chat.” [Online]. Available: <https://chatgpt.com/share/6933ca04-1328-800b-b536-237f0dd56eb7>
- [38] “AM2301B - Wired Enclosed AHT20 - Temperature and Humidity Sensor,” adafruit. [Online]. Available: <https://www.adafruit.com/product/5181>
- [39] “AM2301B Temp + Humidity Data Sheet.”
- [40] “HT_Datasheet_SHT3xA_DIS.”
- [41] “Sensirion_Humidity_Sensors_SF2_Filter_Cap_Datasheet.”
- [42] “SHT30A-DIS-B.” [Online]. Available: <https://sensirion.com/products/catalog/SHT30A-DIS-B>
- [43] “SF2 filter cap.” [Online]. Available: <https://sensirion.com/products/catalog/SF2-filter-cap>
- [44] Mouser Electronics, “HPP845E131R5 (HTU21D(F)),” HPP845E131R5 (HTU21D(F)). [Online]. Available: <https://www.mouser.com/ProductDetail/Measurement-Specialties/HPP845E131R5?qs=sGAepiMZZMt8%252Bp86Qvn10ut036V2xCFQRZ0aUwPQGkoSsuVU2WXKYA%3D%3D>
- [45] “HTU21D Digital High Accuracy RH/T Sensor,” TE Connectivity. [Online]. Available: <https://www.te.com/en/product-CAT-HSC0004.html>
- [46] “HTU21D(F) Data Sheet.”
- [47] “AM2320 Data Sheet”.
- [48] “AM2320 Digital Temperature and Humidity Sensor,” adafruit. [Online]. Available: <https://www.adafruit.com/product/3721>
- [49] Eaton, “A Complete Guide to Push Button Switches.” Accessed: Dec. 05, 2025. [Online]. Available: <https://uk.rs-online.com/web/content/discovery/ideas-and-advice/push-button-switches-guide>
- [50] DevX Editorial Staff, “Directional Pad.” Oct. 17, 2023. Accessed: Dec. 06, 2025.

[Online]. Available: <https://www.devx.com/terms/directional-pad/>

[51] Tom Harris, "How Joysticks Work," How Joysticks Work. Accessed: Dec. 05, 2025. [Online]. Available: <https://electronics.howstuffworks.com/joystick.htm>

[52] State Electronics, "Industrial Joystick Controls," Potentiometers.com. [Online]. Available: <https://www.potentiometers.com/joydesign.cfm>

[53] Nintendo Switch Support, "Joy-Con Control Sticks Are Not Responding or Respond Incorrectly (responsiveness syndrome or so-called 'drifting')," Joy-Con Control Sticks Are Not Responding or Respond Incorrectly (responsiveness syndrome or so-called "drifting"). Accessed: Dec. 05, 2025. [Online]. Available: <https://www.nintendo.com/en-gb/Support/Repair/Joy-Con-Control-Sticks-Are-Not-Responding-or-Respond-Incorrectly-responsiveness-syndrome-or-so-called-drifting--1867803.html>

[54] Joe Kimbrell, "Fundamentals of Industrial Encoder Sensing Technologies, Motion Detection Theory and Methods, and Signal Output Styles." Dec. 05, 2025. Accessed: Dec. 05, 2025. [Online]. Available: <https://cdn.automationdirect.com/static/press/encoder-white-paper.pdf>

[55] How to Mechatronics, "How Rotary Encoder Works," How Rotary Encoder Works. Accessed: Dec. 05, 2025. [Online]. Available: <https://howtomechatronics.com/tutorials/arduino/rotary-encoder-works-use-arduino/>

[56] AsahiKASEI Microdevices, "Principle and advantages of magnetic encoder," Principle and advantages of magnetic encoder. Accessed: Dec. 05, 2025. [Online]. Available: <https://www.akm.com/us/en/products/rotation-angle-sensor/tutorial/magnetic-encoder/>

[57] DYNAPAR, "Optical Encoder: Understanding the Basics." Accessed: Dec. 05, 2025. [Online]. Available: <https://www.dynapar.com/knowledge/encoder-basics/encoder-technology/rotary-optical-encoders/>

[58] Encoder Products Company, "What is a rotary encoder?," What is a rotary encoder? Accessed: Dec. 05, 2025. [Online]. Available: <https://www.encoder.com/rotary-encoders>

[59] Novanta, "What is an optical encoder?," What is an optical encoder? [Online]. Available: <https://www.celeramotion.com/optical-sensors/optical-encoders/>

[60] E-Switch, "JS1300AQ Datasheet." Dec. 20, 2006. Accessed: Aug. 27, 2025. [Online]. Available: <https://configured-product-images.s3.amazonaws.com/2D/specs/JS1300AQ.pdf>

[61] E-Switch, "Series JS1300 Switches Datasheet." Accessed: Aug. 27, 2025. [Online]. Available: <https://configured-product-images.s3.amazonaws.com/Datasheets/JS1300.pdf>

[62] ALPSALPINE, "SKRHADE010 Datasheet," SKRHADE010 4-direction Type with Center-push Function (Surface Mount) SKRH Series. Accessed: Aug. 27, 2025. [Online]. Available: <https://tech.alpsalpine.com/e/products/detail/SKRHADE010/#ancFig1>

[63] ALPSALPINE, "4-direction Type with Center-push Function (Surface Mount) SKRH Series," SKRHADE010 Product Information. Accessed: Aug. 24, 2025. [Online]. Available: <https://tech.alpsalpine.com/e/products/detail/SKRHADE010/#ancFig1>

[64] HRO Electronics, "SparkFun COM-26850 Datasheet." [Online]. Available: <https://cdn.sparkfun.com/assets/5/4/a/e/b/Korean-Hroparts-Elec-K1-1210UN.pdf>

[65] C&K, "THB Series Miniature Thumbstick." Littelfuse, May 15, 2024. Accessed: Aug.

- 25, 2025. [Online]. Available: <https://www.ckswitches.com/media/2873/thb.pdf>
- [66] Parallax, “Parallax 2-Axis Joystick Potentiometer Datasheet.” 2018. Accessed: Aug. 27, 2025. [Online]. Available: <https://www.parallax.com/package/2-axis-joystick-potentiometer-datasheet/>
- [67] Adafruit, “Adafruit Mini 2-Axis Analog Thumbstick.” [Online]. Available: <https://www.adafruit.com/product/2765>
- [68] “KY-023 Dual-axis Joystick Module.” Accessed: Aug. 28, 2025. [Online]. Available: <https://www.mantech.co.za/datasheets/products/KY-023-220909A.pdf?srsId=AfmBOorFGhmr5q7nKZPdND9DhNdn24XFvfdOgyuaUpW3ol1uw6wot1uN>
- [69] “IQL-IMX335/FF,” DigiKey Electronics. Accessed: Sept. 17, 2025. [Online]. Available: <https://www.digikey.com/en/products/detail/image-quality-labs-inc/IQL-IMX335-FF/23568697>
- [70] “camera-module-3-product-brief.pdf.” Accessed: Dec. 06, 2025. [Online]. Available: <https://datasheets.raspberrypi.com/camera/camera-module-3-product-brief.pdf>
- [71] “OV7670_DS (1.4).fm.” Accessed: Dec. 06, 2025. [Online]. Available: https://web.mit.edu/6.111/www/f2016/tools/OV7670_2006.pdf
- [72] OSOYOO, “OSOYOO 3.5” DSI Datasheet.” [Online]. Available: https://osoyoo.com/picture/3.5inch_DSI/OSOYOO_3.5dsi/3.5DSI_datasheet.pdf
- [73] Waveshare, “4.3inch DSI LCD.” [Online]. Available: https://www.waveshare.com/wiki/4.3inch_DSI_LCD
- [74] Waveshare, “4inch DSI LCD (C).” [Online]. Available: [https://www.waveshare.com/wiki/4inch_DSI_LCD_\(C\)](https://www.waveshare.com/wiki/4inch_DSI_LCD_(C))
- [75] Cadence PCB Solutions, “Length Matching for High Speed Signals in PCB Design.” [Online]. Available: https://resources.pcb.cadence.com/blog/2019-guide-on-pcb-trace-length-matching-vs-frequency?utm_source=chatgpt.com
- [76] M. Woolley, “Understanding Reliability in Bluetooth Technology”.
- [77] CIRRUS LOGIC, “CS4334/5/8/9.” CIRRUS LOGIC, Jan. 2003.
- [78] I. Collins, “Phase-Locked Loop (PLL) Fundamentals,” Analog Devices, 2018.
- [79] Analog Devices, “MAX98357A/MAX98357B - Tiny, Low-Cost, PCM Class D Amplifier with Class AB Performance,” July 2019.
- [80] TEXAS INSTRUMENTS, “PCM510xA.” May 2015.
- [81] B. Brannon, “Designing a Super-Heterodyne Multi-Channel Digital Receiver.” Accessed: Sept. 08, 2025. [Online]. Available: https://www.analog.com/media/en/technical_documentation/technical-articles/371982266wideband.pdf
- [82] ICStation, “PLL LCD Digital FM Stereo Radio Receiver Module 50Hz-18KHz Wireless Microphone Module DC 3-5V with LCD Display,” PLL LCD Digital FM Stereo Radio Receiver Module 50Hz-18KHz Wireless Microphone Module DC 3-5V with LCD Display. Accessed: Sept. 08, 2025. [Online]. Available: <https://www.icstation.com/digital-stereo-radio-receiver-module-50hz-18khz-wireless-microphone-module-with-display-p-10545.html>
- [83] E. Wegner, “TFM-6100W.” [Online]. Available:

https://www.radiomuseum.org/r/sony_tfm_6100_w.html

[84] SKYWORKS, “Si4730-31-34-35-D60.” SKYWORKS, Nov. 10, 2021. [Online]. Available: <https://www.skyworksinc.com/-/media/SkyWorks/SL/documents/public/data-sheets/Si4730-31-34-35-D60.pdf>

[85] F. Fe, “CXA1538M/N/S”.

[86] SKYWORKS, “Si4702-03-C19.” SKYWORKS, Nov. 01, 2021. [Online]. Available: <https://www.skyworksinc.com/-/media/SkyWorks/SL/documents/public/data-sheets/Si4702-03-C19.pdf>

[87] NXP, “Quarter-Wave Monopole Antennas.” [Online]. Available: https://docs.nxp.com/bundle/AN2731/page/topics/quarter-wave_monopole_antennas.html

[88] G. L. Hall, Ed., “The ARRL antenna compendium. 1 / ed. Gerald L. Hall,” in Radio amateur’s library, no. 58. , Newington, Conn: American Radio Relay League, 1985.

[89] Multiple Authors, “Why does a 1/4 wavelength have a ground plane, and 1/2 wavelength needs none?” [Online]. Available: <https://electronics.stackexchange.com/questions/480508/why-does-1-4-wavelength-have-a-ground-plane-and-1-2-wavelength-needs-none>

[90] “Antenna Theory Analysis & Design 4th Ed.”

[91] American Radio Relay League, “Test Procedures to Measure BPL Interference.” [Online]. Available: <https://www.arrl.org/test-procedures-to-measure-bpl-interference>

[92] P. Koskimaa, “Ferrite Rod Antenna in a Nanosatellite Medium and High Frequency Radio”.

[93] B. Schweber, “The Reasons for 50 Ohm and 75 Ohm Transmission Lines.” [Online]. Available: <https://www.digikey.in/en/blog/the-reasons-for-50-ohm-and-75-ohm-transmission-lines>

[94] “Amazon - Superbat FM Telescopic Antenna Kit 75 Unbal F Type Connector DAB Radio Replacement Antenna for TV AM FM Radio Stereo Receiver Bose Wave etc.” [Online]. Available: <https://www.amazon.com/Superbat-Telescopic-Connector-Replacement-Receiver/dp/B07S3SR2RP>

[95] “Alibaba - Factory price mini F connector brass rod telescopic antenna for radio.” [Online]. Available: https://www.alibaba.com/product-detail/Factory-price-mini-F-connector-brass_60550080845.html

[96] “Ebay - 75 Ohm W/TV / 3.5 Adapter F Type Telescopic Aerial Antenna For BoseWave Radio FM.” [Online]. Available: https://www.ebay.com/itm/317023557831?_trksid=p2332490.c101196.m2219&itmprp=cksum%3A317023557831b69038de00cf4af2b26de2018dfd6f75%7Cenc%3AAQAKAAABQKf%252BZDynwIBzPFOUXzsYfl6V00AKQ%252B%252B7P%252BQrUe%252FaNcA%252F6g5%252BKJDCW80yl3sDyiOtDc%252Bwp382udDvnYxF7ZDSjYqJm5GyOUkm75MegRvdMcZOp66%252FyDudCANRIWAoDDDX9VSJkBrHcrwnz%252BCm8i0Koa1xRQWdP6N2ZBT3YOMG3ykMbY0jRThUnQYKNbNbPMB%252F1%252FMIVajx3a1WSQp7bjSoYr5rZoIpRDYYkNMNzIf3pN%252FQwaaliZ%252BDEaYk%252BAI5N8RjO7oWLybUxYCNbyRG8ksWRtGbFeWZR82N%252BdsYAsgs0crzS%252Bv2%252B78n1IOeTqV4NXnZpNc%252F3XnFsUg%252Biv

2pFlioBvwsRpN5Ci8F3gJoGz2F3kKpTE6CDt6kS6Ld66iAejxDJ6YI4oVSQB724vEZV4TaHG
GbVEkC5KEJ%252BpyYPCaoR32v%7Campid%3APL_CLK%7Cclp%3A2332490&itmmeta=
01K444THWCH4FZM1PRET3RKYG0

[97] “Microphone Input Transformer Selector Guide.” [Online]. Available:

https://www.jensen-transformers.com/transformers/mic-input/?utm_source=chatgpt.com

[98] “Jensen Transformers JT-123-FLPCH Line Output Transformer.” [Online]. Available:

https://www.cs1.net/products/jensen_transformers/JT-123-FLPCH.htm

[99] “2 TI TLC2272AC 2 MHZ RAIL TO RAIL INPUT OUTPUT DUAL OP AMP IC NOS
SOIC-8 USA.” [Online]. Available: <https://www.ebay.com/itm/385615723767>

[100] “ADA4528-2-EP.”

[101] “adt1082_084.”

[102] “tl082-n.”

[103] “Logging the Linear Digital Potentiometer.” [Online]. Available:

https://www.analog.com/en/resources/technical-articles/logging-the-linear-digital-potentiometer.html/?utm_source=chatgpt.com

[104] “DIY Stepped Attenuator.” [Online]. Available: <https://audio-at-home.com/diy-stepped-attenuator>

[105] “Potentiometer.” [Online]. Available: en.wikipedia.org/wiki/Potentiometer

[106] “Taiwan-Alpha-16mm-pots-datasheet.”

[107] “C-31-.”

[108] “Potentiometer-450-datasheet.”

[109] Analog Devices, “MT-041 Tutorial.” Analog Devices, Oct. 2008. [Online]. Available:

<https://www.analog.com/media/en/training-seminars/tutorials/MT-041.pdf>

[110] TEXAS INSTRUMENTS, “lm4910.” TEXAS INSTRUMENTS, Mar. 2007. [Online]. Available:

https://www.ti.com/lit/ds/symlink/lm4910.pdf?ts=1765058403785&ref_url=https%253A%252F%252Fwww.google.com%252F

[111] TEXAS INSTRUMENTS, “lm4808.” TEXAS INSTRUMENTS, May 2013. [Online]. Available:

https://www.ti.com/lit/ds/symlink/lm4808.pdf?ts=1765058415537&ref_url=https%253A%252F%252Fwww.google.com%252F

[112] JAPAN RADIO CO., “NJU72040_E.” JAPAN RADIO CO. [Online]. Available:

https://www.mouser.com/datasheet/2/294/NJU72040_E-1604853.pdf

[113] DIODES INCORPORATED, “PAM8301.” DIODES INCORPORATED, Sept. 2020.

[Online]. Available: <https://www.diodes.com/datasheet/download/PAM8301.pdf>

[114] DIODES INCORPORATED, “PAM8302A.” DIODES INCORPORATED, Apr. 2013.

[Online]. Available: <https://cdn-shop.adafruit.com/datasheets/PAM8302A.pdf>

[115] TEXAS INSTRUMENTS, “lm386.” TEXAS INSTRUMENTS, Aug. 2023. [Online].

Available: <https://www.ti.com/lit/ds/symlink/lm386.pdf>

[116] SAME SKY, “sj-43504-smt-tr.” SAME SKY, Sept. 12, 2024. [Online]. Available:

- <https://www.sameskydevices.com/product/resource/sj-43504-smt-tr.pdf>
- [117] SAME SKY, “sj1-352xn.” SAME SKY, Sept. 12, 2024. [Online]. Available: <https://www.sameskydevices.com/product/resource/sj1-352xn.pdf>
- [118] ADAM TECH, “asj-125-3-data-sheet.” ADAM TECH, June 10, 2025. [Online]. Available: https://app.adam-tech.com/products/download/data_sheet/203685/asj-125-3-data-sheet.pdf
- [119] Multicomp Pro, “2819666.” Multicomp Pro, June 19, 2017. [Online]. Available: <https://www.farnell.com/datasheets/2819666.pdf>
- [120] SOBERTON INC., “SP-2027NW-completed-on-16-Sept-2018.” SOBERTON INC., Sept. 16, 2018. [Online]. Available: <https://www.soberton.com/wp-content/uploads/2018/10/SP-2027NW-completed-on-16-Sept-2018.pdf>
- [121] DB UNLIMITED, “SW230704-1.” DB UNLIMITED, Sept. 10, 2017. [Online]. Available: https://dbunlimitedco.com/images/product_images/2D-Drawings/SW230704-1.pdf
- [122] “Luminous efficiency of LED luminaires,” Lena Lighting. Accessed: Dec. 05, 2025. [Online]. Available: <https://lenalighting.com/company/knowledge-base/1705-luminous-efficiency-of-led-luminaires>
- [123] “LED Lighting,” Energy.gov. Accessed: Dec. 05, 2025. [Online]. Available: <https://www.energy.gov/energysaver/led-lighting>
- [124] “United States lighting energy policy,” Wikipedia. Nov. 20, 2025. Accessed: Dec. 05, 2025. [Online]. Available: https://en.wikipedia.org/w/index.php?title=United_States_lighting_energy_policy&oldid=1323240160
- [125] “LEP vs LED: Advantages Comparison.” Accessed: Dec. 05, 2025. [Online]. Available: <https://www.jg-ledlight.com/Revolutionizing-Illumination-The-Superiority-of-LEP-Over-LED-in-Lighting-Technology-id67935797.html>
- [126] “PDF.” Accessed: Sept. 09, 2025. [Online]. Available: <https://downloads.creeled.com/files/ds/x/XLamp-XPL.pdf>
- [127] “PDF.” Accessed: Dec. 05, 2025. [Online]. Available: <https://downloads.creeled.com/files/ds/h/HB-CLM3C-MKW-WKW.pdf>
- [128] adminw, “Rechargeable Lithium Batteries vs Alkaline Batteries.” Accessed: Sept. 03, 2025. [Online]. Available: <https://www.redwaypower.com/rechargeable-lithium-batteries-vs-alkaline-comprehensive-guide/>
- [129] admin3, “Choosing the Right Battery: A Comparison of Lithium-ion and Nickel cadmium - Global Batteries.” Accessed: Sept. 03, 2025. [Online]. Available: <https://www.globalbatteries.com/choosing-the-right-battery-a-comparison-of-lithium-ion-and-nickel-cadmium/>
- [130] “BU-107: Comparison Table of Secondary Batteries,” Battery University. Accessed: Dec. 05, 2025. [Online]. Available: <http://www.batteryuniversity.com/article/bu-107-comparison-table-of-secondary-batteries/>
- 192
- [131] “BU-302: Series and Parallel Battery Configurations,” Battery University. Accessed:

Dec. 05, 2025. [Online]. Available: <http://www.batteryuniversity.com/article/bu-302-series-and-parallel-battery-configurations/>

[132] “Batteries in Series vs Parallel: Which is Better? - PKENERGY.” Accessed: Dec. 05, 2025. [Online]. Available: <https://pkenergypower.com/batteries-in-series-vs-parallel-which-is-better/>

[133] “USB Power Adapters Guide | Types and Uses Explained.” Accessed: Sept. 16, 2025. [Online]. Available: https://www.cablesandkits.com/learning-center/usb-power-adapters?srsId=AfmBOoq8qnjBdgOjVT1-lb90dqPFH1SqWuLl09J6ApTRICRR23J_d5Q-

[134] L. Roy, “Micro USB vs USB-C: What’s the Difference?,” Help Desk Geek - Tech Tips from Trusted Tech Experts. Accessed: Dec. 05, 2025. [Online]. Available: <https://helpdeskgeek.com/micro-usb-vs-usb-c-whats-the-difference/>

[135] “lightning port vs usb c: Key Differences Explained.” Accessed: Dec. 05, 2025. [Online]. Available: <https://ibinary.net/tech/lightning-port-vs-usb-c-key-differences-explained/>

[136] A. Jalan, “USB-C vs. Lightning Port: Which Is Best?,” MUO. Accessed: Dec. 05, 2025. [Online]. Available: <https://www.makeuseof.com/usb-c-vs-lightning-port-which-is-best/>

[137] “USB Power Delivery: The Technology 1 - Convenience and Safety | Renesas.” Accessed: Dec. 05, 2025. [Online]. Available: <https://www.renesas.com/en/support/engineer-school/usb-power-delivery-02>

[138] “STUSB4500 | Product - STMicroelectronics.” Accessed: Dec. 05, 2025. [Online]. Available: <https://www.st.com/en/interfaces-and-transceivers/stusb4500.html>

[139] “R9J02G012 - Custom USB PD Controller with USB Type-C Authentication | Renesas.” Accessed: Dec. 05, 2025. [Online]. Available: <https://www.renesas.com/en/products/r9j02g012>

[140] “TPS25750 USB Type-C and USB PD Controller with Integrated Power Switches datasheet (Rev. A).” Accessed: Dec. 05, 2025. [Online]. Available: https://www.ti.com/lit/ds/symlink/tps25750.pdf?ts=1764917451111&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FTPS25750

[141] “MAX77818 - Dual Input, Power Path, 3A Switch Mode Charger with Fuel Gauge”.

[142] “BQ2404x 1A, Single-Input, Single Cell Li-Ion and Li-Pol Battery Charger datasheet (Rev. H).” Accessed: Sept. 16, 2025. [Online]. Available: https://www.ti.com/lit/ds/symlink/bq24040.pdf?ts=1758035042074&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FBQ24040

[143] “MAX1551 DS.” Accessed: Aug. 24, 2025. [Online]. Available: <https://www.analog.com/media/en/technical-documentation/data-sheets/MAX1551MAX1555.pdf>

[144] “LT1769 - Constant-Current/Constant-Voltage 2A Battery Charger with Input Current Limiting.” Accessed: Dec. 05, 2025. [Online]. Available: <https://www.analog.com/media/en/technical-documentation/data-sheets/1769fa.pdf>

[145] “LTC4006 - 4A, High Efficiency, Standalone Li-Ion Battery Charger.” Accessed: Dec. 05, 2025. [Online]. Available: <https://www.analog.com/media/en/technical-documentation/data-sheets/4006fa.pdf>

- [146] Analog Devices, “DS2781 Fuel Gauge IC Datasheet.” Accessed: Dec. 06, 2025. [Online]. Available: <https://www.analog.com/media/en/technical-documentation/data-sheets/ds2781.pdf>
- [147] Analog Devices, “Stand-Alone OCV-Based Fuel Gauge.” Accessed: Dec. 06, 2025. [Online]. Available: <https://www.analog.com/media/en/technical-documentation/data-sheets/DS2786B.pdf>
- [148] Texas Instruments, “BQ34Z100-G1 Wide Range Fuel Gauge with Impedance Track™ Technology.” Apr. 01, 2021. Accessed: Dec. 06, 2025. [Online]. Available: https://www.ti.com/lit/ds/symlink/bq34z100g1.pdf?ts=1764941020889&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FBQ34Z100-G1
- [149] “Advantages and Disadvantages of Linear Power Supplies | SEACOMP.” Accessed: Sept. 16, 2025. [Online]. Available: <https://www.seacomp.com/resources/advantages-and-disadvantages-linear-power-supplies>
- [150] “Everything You Need to Know About Universal Power Cords.” Accessed: Sept. 16, 2025. [Online]. Available: <https://www.sfcable.com/blog/universal-power-cords>
- [151] “What is a PSU: Power Supply Units explained.” Accessed: Dec. 05, 2025. [Online]. Available: <https://www.corsair.com/us/en/explorer/diy-builder/power-supply-units/what-is-a-psu-power-supply-units-explained/>
- [152] “Robust Power Source with Ideal Diodes | Analog Devices.” Accessed: Dec. 05, 2025. [Online]. Available: https://www.analog.com/en/resources/analog-dialogue/articles/robust-power-source-with-ideal-diode.html?utm_source=chatgpt.com
- [153] “Analog Devices : Choosing the Correct Switch, Multiplexer, or Protection Product for Your Application.” Accessed: Dec. 05, 2025. [Online]. Available: https://www.analog.com/media/en/news-marketing-collateral/product-selection-guide/choosing_switches_or_muxes.pdf
- [154] “Basics of Power MUX (Rev. A).” Accessed: Dec. 05, 2025. [Online]. Available: https://www.ti.com/lit/an/slvae51a/slvae51a.pdf?utm_source=chatgpt.com&ts=1764993342137&ref_url=https%253A%252F%252F
- [155] “TPS212x 2.8-V to 22-V Priority Power MUX with Seamless Switchover datasheet (Rev. F).” Accessed: Dec. 05, 2025. [Online]. Available: <https://www.ti.com/lit/ds/symlink/tps2121.pdf>
- [156] “LTC4412 (Rev. C).” Accessed: Dec. 05, 2025. [Online]. Available: <https://www.analog.com/media/en/technical-documentation/data-sheets/ltc4412.pdf>
- [157] “LTC4417 (Rev. G).” Accessed: Dec. 05, 2025. [Online]. Available: <https://www.analog.com/media/en/technical-documentation/data-sheets/ltc4417.pdf>
- [158] “SIL2305B(SOT23-6L).pdf.” Accessed: Dec. 05, 2025. [Online]. Available: [https://www.mccsemi.com/pdf/Products/SIL2305B\(SOT23-6L\).pdf](https://www.mccsemi.com/pdf/Products/SIL2305B(SOT23-6L).pdf)
- [159] “MOSFET — Power, Single, P-Channel, Trench, ESD Protected, SC-88 12 V, 3.3 A”.
- [160] “AO3413.pdf.” Accessed: Dec. 05, 2025. [Online]. Available:

<https://www.aosmd.com/sites/default/files/res/datasheets/AO3413.pdf>

[161] “Linear vs. Switching Regulators: Unlocking the Best Power Solution for Your Design | Renesas.” Accessed: Sept. 08, 2025. [Online]. Available: <https://www.renesas.com/en/products/power-management/linear-vs-switching-regulators>

[162] “Buck, Boost, and Buck-Boost Converters Explained | RECOM.” Accessed: Sept. 08, 2025. [Online]. Available: <https://recom-power.com/en/rec-n-an-introduction-to-buck,-boost,-and-buck-boost-converters-131.html>

[163] “TPS56624x 3-V to 16-V Input Voltage, 6-A Synchronous Buck Converter in a SOT-563 Package datasheet (Rev. A).” Accessed: Dec. 05, 2025. [Online]. Available: <https://www.ti.com/lit/ds/symlink/tps566242.pdf?ts=1764952234728>

[164] “LTC3615/LTC3615-1 - Dual 4MHz, 3A Synchronous Step-Down DC/DC Converter.” Accessed: Aug. 26, 2025. [Online]. Available: https://www.analog.com/media/en/technical_documentation/data-sheets/3615fb.pdf

[165] “LM5642/42X High V, Dual Sync Buck Converter w/Oscil Sync datasheet (Rev. K).” Accessed: Aug. 26, 2025. [Online]. Available: <https://www.ti.com/lit/ds/symlink/lm5642x.pdf?ts=1756186276301>

[166] “12MP IMX708 Autofocus Wide Angle Camera Module 3 NoIR for Raspberry Pi, HDR Mode and PDAF supported.” Accessed: Dec. 06, 2025. [Online]. Available: <https://www.arducam.com/12mp-imx708-autofocus-wide-angle-camera-module-3-noir-for-raspberry-pi-hdr-mode-and-pdaf-supported.html>

[167] “HDMI vs DisplayPort: Which One Should You Use?,” RTINGS.com. Accessed: Sept. 07, 2025. [Online]. Available: <https://www.rtings.com/monitor/learn/hdmi-vs-displayport>

[168] “USB-C vs HDMI: Battle of Modern Connectivity Options 2025 - Anker US.” Accessed: Dec. 06, 2025. [Online]. Available: <https://www.anker.com/blogs/hubs-and-docks/usb-c-vs-hdmi>

[169] R. Shafer, “What Is DisplayPort 2.1? - Simple Explanation,” Display Ninja. Accessed: Dec. 06, 2025. [Online]. Available: <https://www.displayninja.com/what-is-displayport-2-1/>

[170] “VESA Releases Updated DisplayPort™ Alt Mode Spec to Bring DisplayPort 2.0 Performance to USB4™ and New USB Type-C® Devices,” DisplayPort. Accessed: Dec. 06, 2025. [Online]. Available: <https://www.displayport.org/pr/vesa-releases-updated-displayport-alt-mode-spec-to-bring-displayport-2-0-performance-to-usb4-and-new-usb-type-c-devices/>

[171] “DisplayPort over USB-C,” DisplayPort. Accessed: Sept. 08, 2025. [Online]. Available: <https://www.displayport.org/displayport-over-usb-c/>

[172] A. Negi, “5 HDMI Types & 7 Standards Explained(Speed Compared).” Accessed: Aug. 25, 2025. [Online]. Available: <https://www.etechnophiles.com/hdmi-types-standards/>

[173] “USB 2.0 vs 3.0: What’s the Difference and Which to Use?,” Anker. Accessed: Sept. 02, 2025. [Online]. Available: <https://www.anker.com/ca/blogs/cables/usb-2-0-vs-3-0>

[174] Raspberry Pi, “Raspberry Pi OS.” Accessed: Dec. 06, 2025. [Online]. Available: <https://www.raspberrypi.com/documentation/computers/os.html>

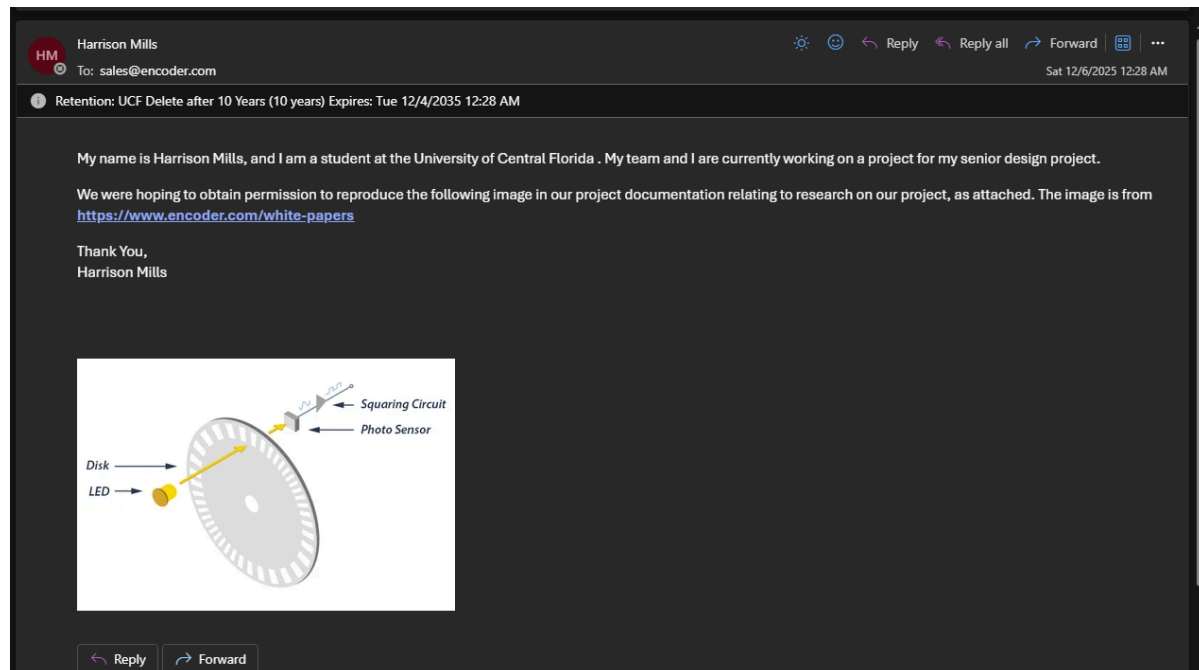
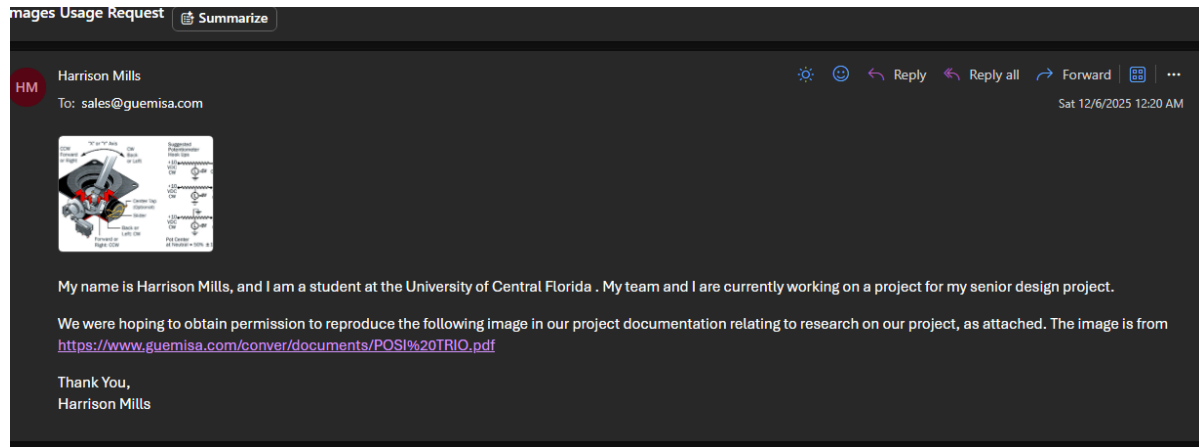
[175] DietPi, “DietPi.com Docs.” Accessed: Dec. 06, 2025. [Online]. Available: <https://dietpi.com/docs/>

- [176] Ubuntu, “Ubuntu Core documentation.” July 27, 2025. Accessed: Dec. 06, 2025. [Online]. Available: <https://documentation.ubuntu.com/core/>
- [177] Python, “An Informal Introduction to Python.” Dec. 06, 2025. Accessed: Dec. 06, 2025. [Online]. Available: <https://docs.python.org/3/tutorial/introduction.html>
- [178] Python, “Python 3.14.2 documentation.” Dec. 06, 2025. Accessed: Dec. 06, 2025. [Online]. Available: <https://docs.python.org/3/>
- [179] Standard C++ Foundation, “Big Picture Issues.” Accessed: Dec. 06, 2025. [Online]. Available: <https://isocpp.org/wiki/faq/big-picture>
- [180] Rust, “Rust Documentation.” Accessed: Dec. 06, 2025. [Online]. Available: <https://doc.rust-lang.org/stable/>
- [181] Rust, “The Rust Programming Language.” Accessed: Dec. 06, 2025. [Online]. Available: <https://doc.rust-lang.org/stable/book/ch00-00-introduction.html>
- [182] Arduino, “What is Arduino?” Jan. 25, 2022. Accessed: Dec. 06, 2025. [Online]. Available: <https://docs.arduino.cc/learn/starting-guide/whats-arduino/>
- [183] Arduino, “Arduino Documentation.” Accessed: Dec. 06, 2025. [Online]. Available: <https://docs.arduino.cc/>
- [184] Priya Bhalla, “GitHub Statistics By Users, Security And Facts (2025),” GitHub Statistics By Users, Security And Facts (2025). Accessed: Dec. 05, 2025. [Online]. Available: <https://electroi.com/stats/github-statistics/>
- [185] GitHub, “GitHub Docs.” Accessed: Dec. 06, 2025. [Online]. Available: <https://docs.github.com/en>
- [186] GitLab, “GitLab Docs.” Accessed: Dec. 06, 2025. [Online]. Available: <https://docs.gitlab.com/>
- [187] Bitbucket, “Bitbucket Documentation.” Accessed: Dec. 05, 2025. [Online]. Available: <https://bitbucket.org/product/guides/getting-started/overview#key-terms-to-know>
- [188] Arduino, “ArduinoDocs: The Autocomplete Feature.” Feb. 14, 2024. Accessed: Dec. 05, 2025. [Online]. Available: <https://docs.arduino.cc/software/ide-v2/tutorials/ide-v2-autocomplete-feature/>
- [189] Arduino, “Using the Arduino Software (IDE).” Jan. 16, 2024. Accessed: Dec. 06, 2025. [Online]. Available: <https://docs.arduino.cc/learn/starting-guide/the-arduino-software-ide/>
- [190] Notepad++ Developers, “What is Notepad++.” Accessed: Dec. 05, 2025. [Online]. Available: <https://notepad-plus-plus.org/>
- [191] Visual Studio Code, “Visual Studio Code documentation.” Oct. 01, 2025. Accessed: Dec. 05, 2025. [Online]. Available: <https://code.visualstudio.com/Docs>
- [192] Visual Studio Code, “GitHub Copilot in VS Code.” Oct. 01, 2025. [Online]. Available: <https://code.visualstudio.com/docs/copilot/overview>
- [193] PlatformIO, “PlatformIO Home.” Feb. 13, 2025. Accessed: Dec. 06, 2025. [Online]. Available: <https://docs.platformio.org/en/stable/home/index.html>
- [194] Espressif, “ESP-IoT Development Framework.” Accessed: Dec. 05, 2025. [Online]. Available: <https://www.espressif.com/en/products/sdks/esp-idf>

- [195] P. Ghosh, “IPC-2221 Standards in PCB Design,” Sierra Circuits. Accessed: Oct. 14, 2025. [Online]. Available: <https://www.protoexpress.com/blog/ipc-2221-circuit-board-design/>
- [196] P. Ghosh, “IPC-2221 Standards in PCB Design,” Sierra Circuits. Accessed: Dec. 05, 2025. [Online]. Available: <https://www.protoexpress.com/blog/ipc-2221-circuit-board-design/>
- [197] “I2C Info – I2C Bus, Interface and Protocol | I2C Bus Specification.” Accessed: Dec. 05, 2025. [Online]. Available: <https://i2c.info/i2c-bus-specification>
- [198] “I2C Primer: What is I2C? (Part 1) | Analog Devices.” Accessed: Dec. 05, 2025. [Online]. Available: <https://www.analog.com/en/resources/technical-articles/i2c-primer-what-is-i2c-part-1.html>
- [199] “I2C-bus specification and user manual,” vol. 2021, 2021.
- [200] Geeks for Geeks Editorial Staff, “Universal Asynchronous Receiver Transmitter (UART) Protocol.” Dec. 29, 2022. Accessed: Dec. 06, 2025. [Online]. Available: <https://www.geeksforgeeks.org/computer-networks/universal-asynchronous-receiver-transmitter-uart-protocol/>
- [201] “The Standard.” [Online]. Available: <https://isocpp.org/std/the-standard>

APPENDIX B: COPYRIGHT PERMISSION REQUESTS AND RESPONSES

REQUESTS:



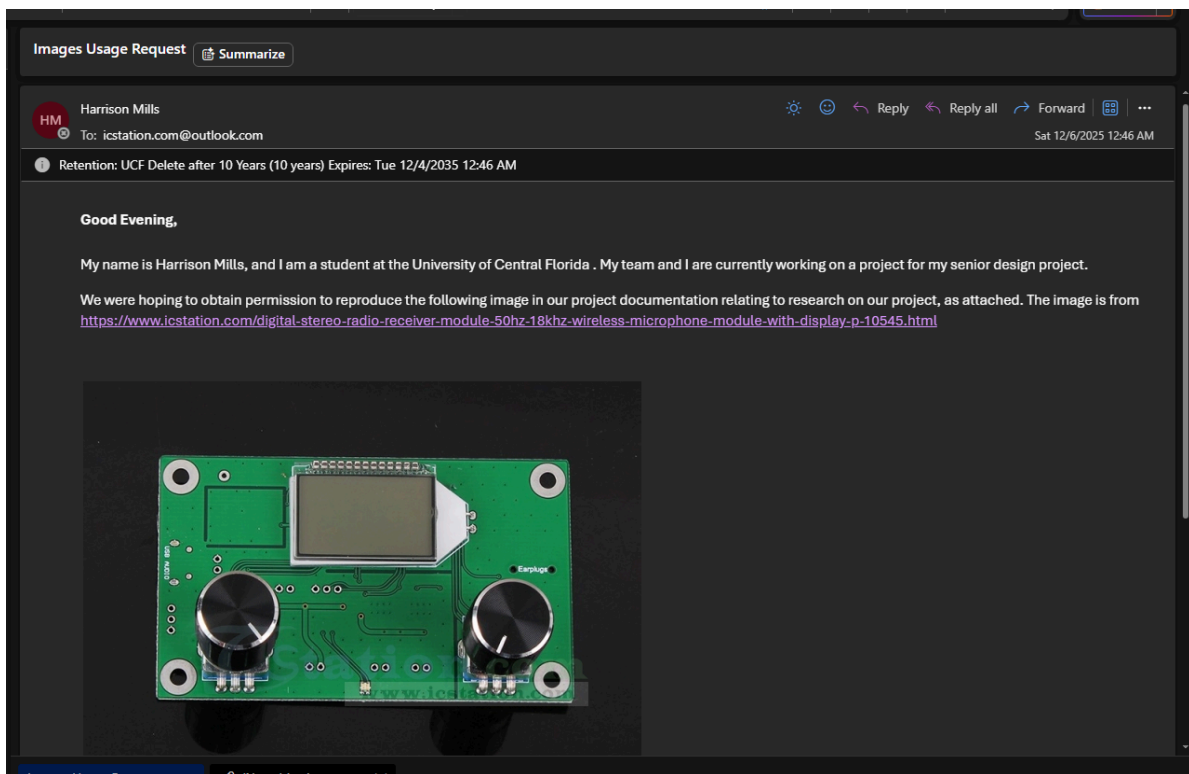
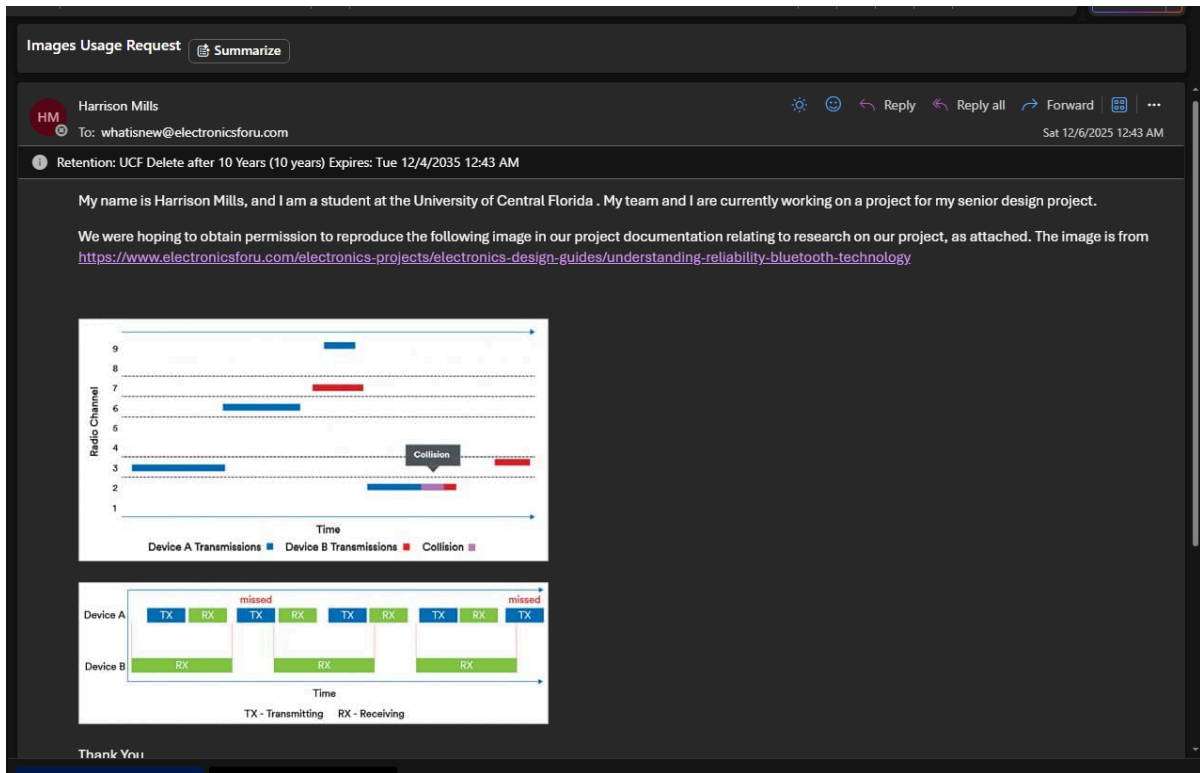


Image usage request

 Summarize

HM

Harrison Mills

To: support@cs1.net

   Reply  Reply all  Forward  

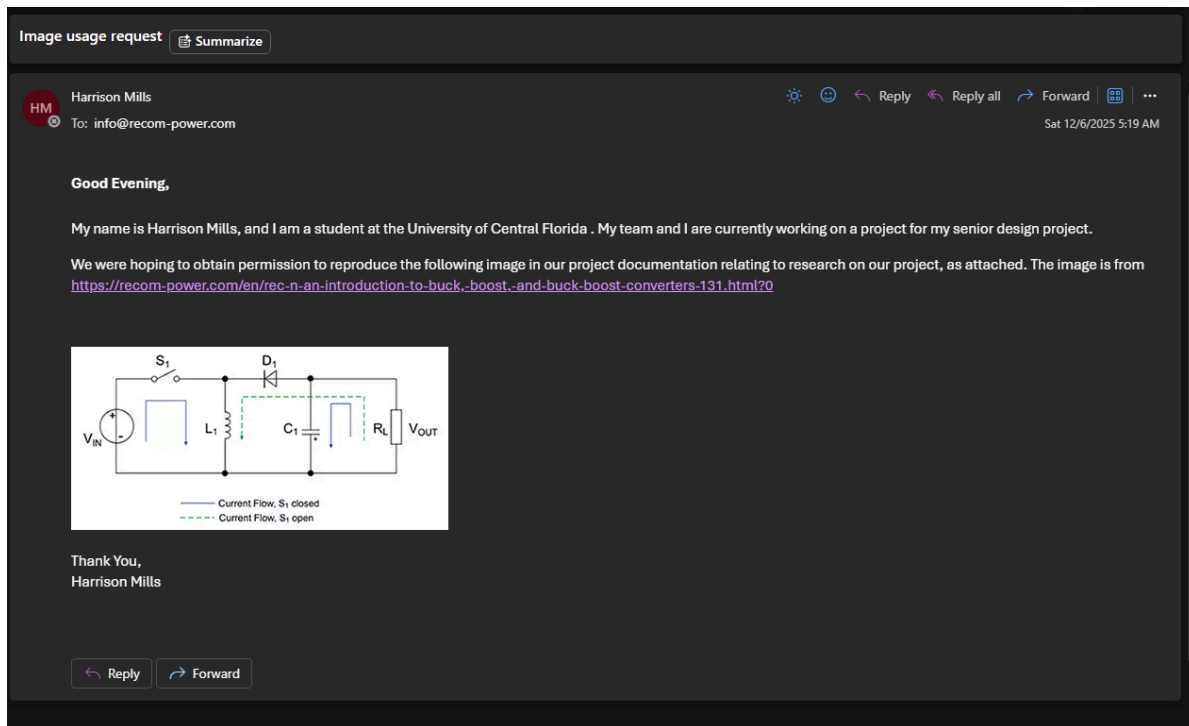
Sat 12/6/2025 5:10 AM

Good Evening,

My name is Harrison Mills, and I am a student at the University of Central Florida . My team and I are currently working on a project for my senior design project. We were hoping to obtain permission to reproduce the following image in our project documentation relating to research on our project, as attached. The image is from https://www.cs1.net/products/jensen_transformers/JT-123-FLPCH.htm

Thank You,
Harrison Mills





AsahiKASEI

MICRODEVICES

[Products](#)
[Applications](#)
[Support](#)
[About AKM](#)

[Inquiries](#)
[Purchase a product or sample](#)
[Technical questions](#)
[Other inquiries](#)
[myAKM](#)

Email Address

ha934819@uct.edu

First Name

HARRISON

Last Name

MILLS

Company

UCF

Organization

UCF

Type of business

Other

Official position

Other

Phone

14074175103

Country/Location

United States

State

FL Florida

Product Category

Select product category

Product Name

Application

Select application

Inquiry Type

Select Inquiry Type

Questions

at the University of Central Florida, my team and I are currently working on a project for my senior design project.

We were hoping to obtain permission to reproduce the following image in our project

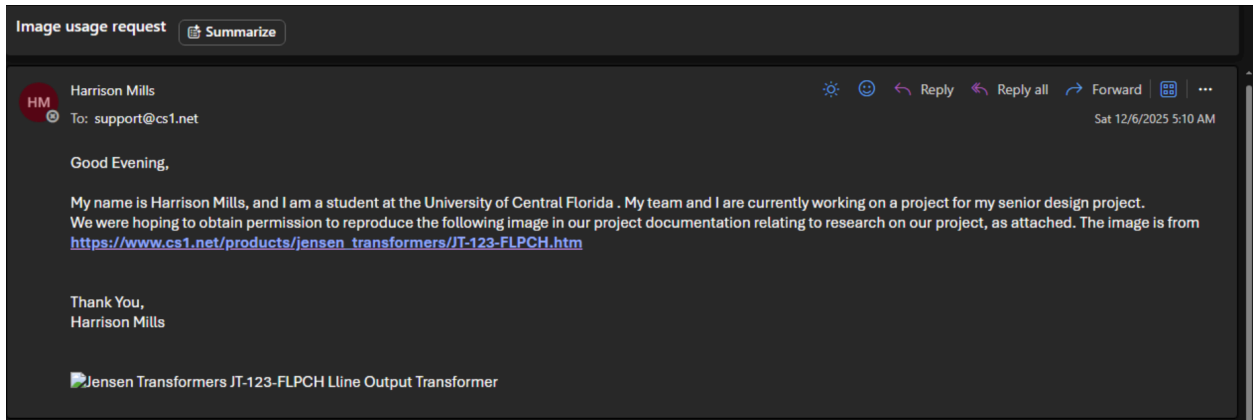
I agree to AKM's collection of the information entered above

☒

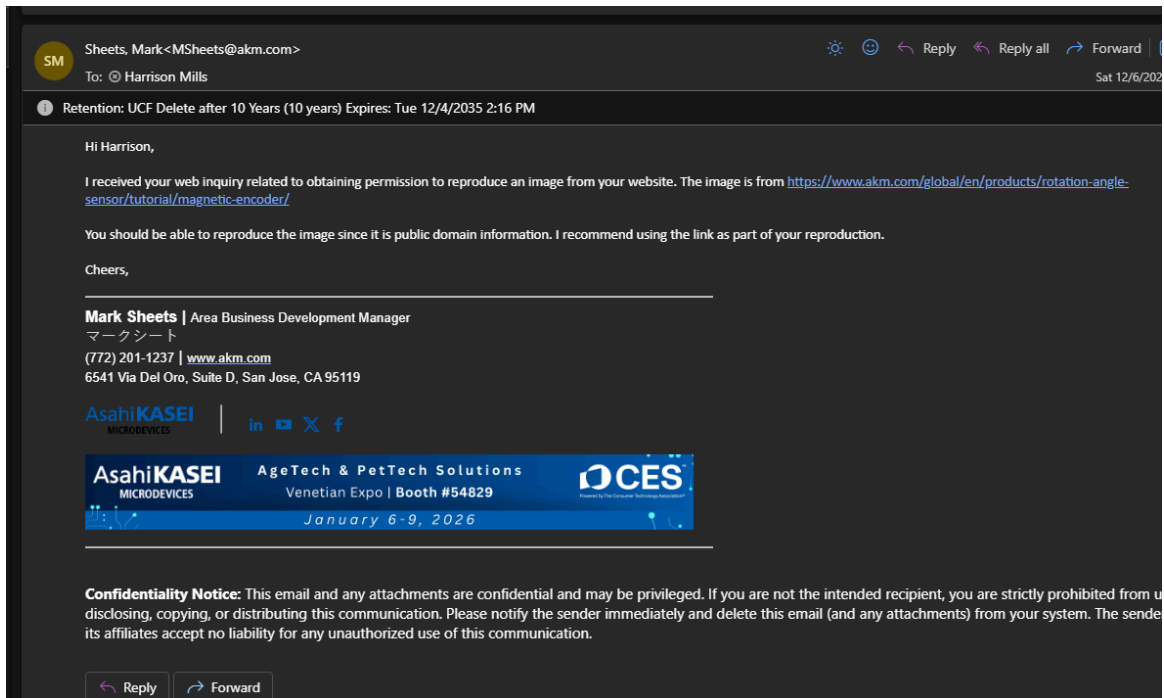
Sign up for myAKM?

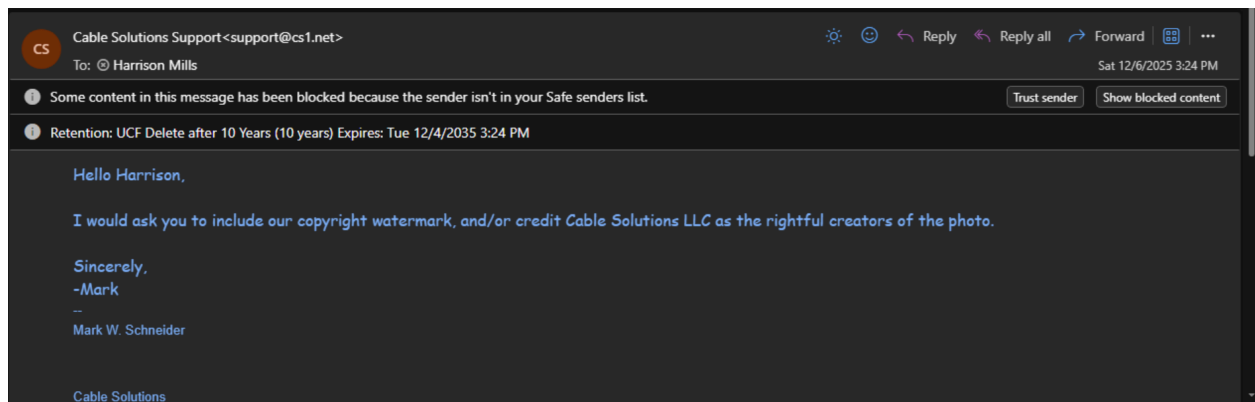
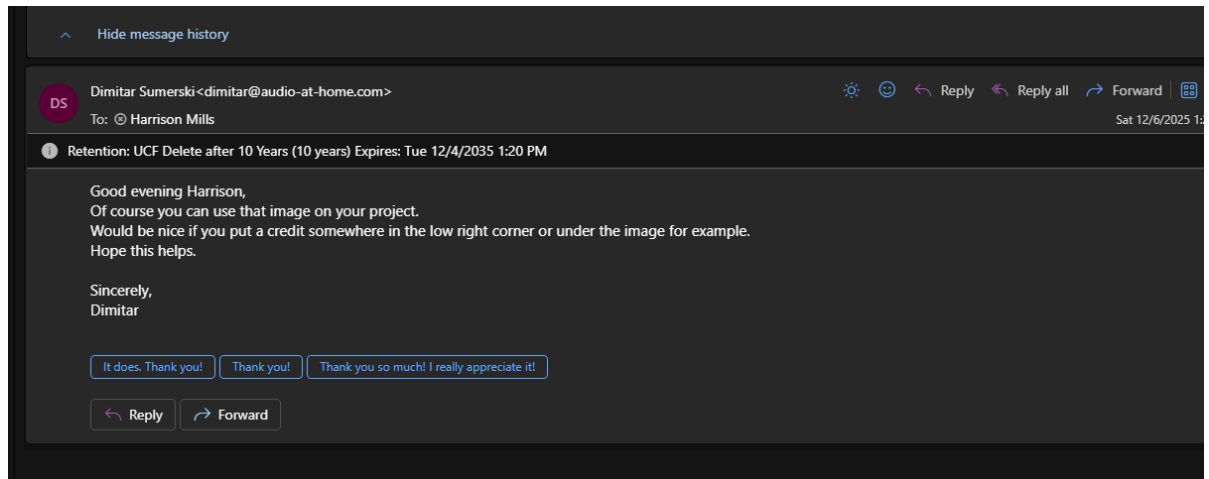
☐ yes
☒ no

Submit



RESPONSES:





APPENDIX C: AI/LLM CASE STUDIES

ChatGPT Prompt 1

[1a] Prompt:

“What are the most common sensor technologies used to measure heartbeat? Can you describe to me the pros, cons, and how these technologies work?”

[1b] Response:

What are the most common technologies used to measure heartbeat?
Can you describe to me the pros, cons, and how these technologies work?

Great question 😊 — heartbeat (or heart rate) can be measured using several different sensing technologies, each with distinct trade-offs. Here are the **most common ones**, how they work, plus pros & cons:

1. Photoplethysmography (PPG, Optical)

How it works:

- An LED (often red or IR) shines light into skin.
- A photodiode measures changes in light absorption as blood volume pulses with each heartbeat.
- Common in fitness trackers, smartwatches, fingertip pulse oximeters.

Pros:

- Small, cheap, easy to integrate (just LEDs + photodiode + microcontroller).

- Works with simple contact (finger, wrist, earlobe).
- Can also estimate **blood oxygen saturation (SpO₂)** with multiple wavelengths.

Cons:

- Sensitive to **motion artifacts** (movement causes noise).
- Needs good skin contact and ambient light shielding.
- Power draw depends on LED intensity (can be non-trivial for wearables).

Examples: MAX30102, PulseSensor, SON1303 modules.

2. Electrocardiography (ECG / EKG)

How it works:

- Electrodes placed on skin measure the tiny electrical signals generated by heart muscle contractions.
- Gives a direct trace of heart's electrical activity (P, QRS, T waves).

Pros:

- **Most accurate method** for heart rate and rhythm.
- Can detect arrhythmias, irregular beats, and provide clinical data.
- Very low power once electrodes are connected.

Cons:

- Requires multiple electrodes + skin contact (adhesive pads, chest strap).
- Less user-friendly for casual/wearable devices.
- Noise from poor contact or muscle activity.

Examples: AD8232 module, chest strap HR monitors.

3. Ballistocardiography (BCG / Seismocardiography)

How it works:

- Detects small mechanical vibrations or accelerations caused by blood ejection with each heartbeat.
- Uses accelerometers, gyroscopes, or piezo sensors to pick up subtle body movements.

Pros:

- Can be measured without direct skin contact (through a chair, bed, etc.).
- Non-invasive and unobtrusive.

Cons:

- Lower accuracy than ECG/PPG.
- Strongly affected by body motion and external vibrations.
- Mostly research / niche applications, not mainstream wearables.

Examples: Experimental smart beds, some research wearables.

4. Phonocardiography (PCG / Acoustic)

How it works:

- Uses microphones or piezo sensors to detect **heart sounds** ("lub-dub").
- Analyzes timing of valve sounds to infer heart rate.

Pros:

- Can give additional information about valve function, murmurs.
- Non-electrical, doesn't need electrodes.

Cons:

- Very sensitive to ambient noise and placement.
- Rare in consumer devices, mostly medical/research.

Examples: Digital stethoscopes, hospital diagnostic tools.

5. Pressure-Based (Arterial Pulse, Piezoelectric, Tonometers)

How it works:

- A piezoelectric film or pressure sensor detects the pulse wave at an artery (wrist, finger, etc.).
- Each beat creates a pressure spike, measurable mechanically.

Pros:

- Low power, simple electronics.
- Can be embedded in straps or cushions.

Cons:

- Needs precise placement over an artery.
- Signal quality varies a lot with movement/pressure.
- Mostly supplemental, not as robust as ECG/PPG.

Quick Summary

Method	How It Works	Pros	Cons	Common Use
PPG (optical)	Light absorption changes	Small, cheap, SpO ₂ possible	Motion-sensitive, power draw	Fitness bands, oximeters
ECG	Electrical heart activity	Clinical-grade, arrhythmia detection	Needs electrodes, less convenient	Chest straps, hospital
BCG	Body micro-vibrations	Non-contact, unobtrusive	Very noise-sensitive	Research, smart furniture
PCG	Heart sounds (microphone)	Detects murmurs, valve issues	Placement + noise sensitive	Stethoscopes
Pressure/Piezo	Arterial pulse pressure	Simple, low power	Placement critical, less accurate	Basic HR straps, cushions

For your mobile device project:

- PPG (MAX30102 / PulseSensor) is the most **practical + compact** option.
- ECG (AD8232) would give higher accuracy but requires **electrode pads**, which may not be ideal for a small portable form factor.
- **Outdoor/robustness** also favors PPG modules with some shielding, since they can be embedded and just need skin contact.

Do you want me to go a step further and map **which heartbeat tech makes the most sense for your ESP32-S3 + CM4 device**, given you want **low-power, mobile, outdoor-friendly** sensors?

   ...

ChatGPT Prompt 2

[2a] Prompt:

“The MCU would be working with the SBC, a Pi 4, to feed it information from peripherals such as the sensors to display to the user.”

[2b] Response:

The MCU would be working with the SBC, a Pi 4, to feed it information from peripherals such as the sensors to display to the user.

Thanks for the clarification — that setup makes perfect sense for your capstone. Here's how I'd architect and recommend your system given that:

System Overview

You're using:

- An **MCU** to handle **real-time sensor input and mobile-mode operations**
- A **Raspberry Pi 4** as your **main computing unit** for desktop-mode tasks like GUI, music, and maps

The MCU acts like a **smart sensor hub**, offloading real-time and power-sensitive tasks from the Pi.

✓ Role Breakdown

🎯 MCU Responsibilities

- Runs in both mobile and desktop mode
- Interfaces directly with:
 - Heart rate sensor
 - Humidity sensor
 - Temperature sensor
- Operates in low-power mode when unplugged
- On USB plug-in (desktop mode):
 - Feeds formatted sensor data to Pi 4
 - Can act as a USB CDC (serial over USB) or UART device

💻 Pi 4 Responsibilities

- Boots full Linux (e.g., Raspberry Pi OS, Ubuntu)
- Detects when docked
- Launches GUI apps:
 - Dashboard showing sensor values
 - Maps (possibly via offline map tiles)
 - Music player
- Requests sensor data from MCU via serial or USB
- Displays output via HDMI or local screen

🔧 Communication Between MCU & Pi 4

🔧 Recommended Interfaces:

Interface	Notes
USB CDC (Serial over USB)	Most plug-and-play; Pi sees MCU as <code>/dev/ttyUSB0</code>
UART (GPIO serial)	Direct pins, but need level shifting if voltages differ
I2C (less preferred)	Pi as master, MCU as slave (not ideal for async updates)

Use a simple packet protocol over serial — e.g., JSON, CSV, or a custom binary protocol with header and checksum.

🔧 Recommended MCU (Updated)

🔧 ESP32-S3

- Dual-core, low-power modes
- Built-in USB OTG (for CDC mode!)
- Wi-Fi/BLE (if needed)
- Tons of library support (Arduino, ESP-IDF, PlatformIO)
- I2C/SPI/ADC for all your sensor needs

It pairs very well with the Pi 4, especially using USB serial or UART.

Programming Languages

For MCU (ESP32-S3):

- ESP-IDF (C) – Native, robust
- Arduino (C++) – Easier, quicker
- PlatformIO – IDE-agnostic, supports both above
- Optional: add FreeRTOS if you want concurrent tasks (e.g., heart rate sampling vs USB I/O)

For Pi 4:

- Python – Ideal for:
 - Serial communication via `pyserial`
 - GUI via `tkinter`, `PyQt`, `Kivy`, etc.
- C++/Qt – If you want a more polished GUI
- Shell scripting or `systemd` – For mode detection and startup control



Power & Mode Switching Strategy

Detect Docking:

- Use GPIO pin or USB power detect on MCU
- Or detect VBUS (5V in) directly with a divider into ADC or digital input

When Docked (Desktop Mode):

- Wake Pi 4 via GPIO or just power line
- MCU:
 - Switches to high-power mode
 - Increases sensor polling rate
 - Enables full data streaming

When Undocked (Mobile Mode):

- Pi 4 powers down
- MCU enters low-power loop
- Sensors run in low-power mode
- Minimal display (if any) for info

Example Packet Flow (Serial via USB)

css

 Copy code

[MCU] --> [Pi 4]

> {"temp":24.5,"humidity":65,"hr":74}

Or a simpler CSV:

markdown

 Copy code

> TEMP:24.5,HUM:65,HR:74

Final Advice

- Go with ESP32-S3, use **USB CDC serial** to communicate with Pi
- Use **Python** on Pi for sensor parsing, dashboard UI, and app launching
- Design clear **mode-switch logic** on the MCU side
- Add robust **error handling** and **protocol framing** in serial communication



Would you like example code stubs for the ESP32 side and Python side serial communication to get started?

ChatGPT Prompt 3

[3a] Prompt:

“Build me a me a system with a usb-c charging , a pd controller for it, charging IC and Gauge IC, and a switching MUX tell me things I need to consider”

[3b] Response:

Thought for 18s >

Nice — below is a compact, practical system design you can drop into a schematic/PCB for a single-cell Li-ion product that accepts USB-C Power Delivery. I'll give you a suggested parts list (PD controller, charger IC, fuel-gauge, power-path / switching MUX), a block-diagram concept, and a prioritized checklist of electrical, mechanical and firmware considerations with short rationales and links to datasheets/app notes you can use for implementation and verification.

High-level block diagram (single-cell Li-ion device)

USB-C receptacle (VBUS, CC1/CC2, D+/D-, SBU, Shield) → USB-C PD controller (handles CC, PD negotiation)

→ VBUS power path / FET / ideal-diode MUX → switch-mode battery charger (power-path) → Battery

- Charger provides regulated charge current, V_{sys} /regulation for system while charging.
- Fuel-gauge (I^2C /SMBus) monitors battery and reports SOC to MCU.
- MCU communicates with PD controller (I^2C /INT) if PD controller is programmable; MCU can change requested PDOs or handle vendor-defined behaviour.
- Optional low-level power MUX / ideal-diode controller (for reverse current protection or multiple supply OR-ing) to manage source selection or protect VBUS when the system supplies power.

Recommended parts to consider (good starting points)

USB-C / PD controller

- ST STUSB4500 — standalone PD sink controller with auto-run, dead-battery support and integrated VBUS FET drivers (can handle up to 20 V / 5 A PDOs when combined with appropriate FETs). Good for a sink that doesn't need a full MCU for PD negotiation. [STMicroelectronics](#) +1

Charger / power-path IC (single-cell, high integration)

- TI BQ25895 — switch-mode single-cell Li-ion charger with system power-path, programmable input-current limit (for USB2.0/3.0/adaptor), supports high charging currents (up to ~5 A variants) and thermal regulation. Good if you want integrated buck charger + power path. [Texas Instruments](#) +1

Fuel gauge / battery gauge

- TI BQ34Z100-G1 — wide-range Impedance-Track™ fuel gauge (very capable for production devices, supports many chemistries, SMBus/ I^2C). Good for accurate SOC and rich features. [Texas Instruments](#) +1

Fuel gauge / battery gauge

- **TI BQ34Z100-G1** — wide-range Impedance-Track™ fuel gauge (very capable for production devices, supports many chemistries, SMBus/I²C). Good for accurate SOC and rich features. [Texas Instruments +1](#)
- Alternative low-power gauge: **MAX17055** — ModelGauge M5 EZ, very low quiescent current and easy/quick to bring up for 1-cell devices. [Analog Devices +1](#)

Power MUX / ideal-diode / power-path switch

- **LTC4412** (ideal-diode controller) — implements near-ideal diode function using an external P-MOSFET for low loss OR-ing and reverse-current protection. Useful when you need to prevent battery feeding VBUS or to select between adapter and battery with minimal drop. [Analog Devices +1](#)

Key design considerations (what you must think about)

1. Role & PD negotiation strategy

- Decide whether port is sink only, source, or dual-role. Sink-only is simplest (use STUSB4500 or similar). DRP/Source requires more complex PD stacks and possibly MCU involvement. STUSB4500 supports auto-run sink modes so you can avoid MCU for many use cases. [STMicroelectron... +1](#)

2. Power path & reverse-current protection

- If PD controller/charger does not include a low-loss integrated power path, you must provide external FETs and an ideal-diode controller or MOSFET arrangement to prevent backfeeding VBUS and to allow the system to be powered while battery charges. See TI's app notes on power-path for examples. [Texas Instruments +1](#)

3. Input current limiting & PDO selection

- Use charger + PD negotiation to limit input current to acceptable USB levels or negotiated PDO current. Charger IC should support programmable input-current limit (bq25895 does). If you request high-voltage PDOs (9/12/15/20V), ensure charger and downstream components tolerate that voltage or that the PD controller + charger support step-down. [Texas Instruments +1](#)

4. Dead-battery / VBUS startup

- If the product must boot from a completely depleted battery, pick a PD controller with dead-battery support or design a separate low-voltage path to allow the PD controller to power up and negotiate. STUSB4500 documents dead battery mode. [STMicroelectron... +1](#)

5. Thermal & PCB layout

- High charging currents require careful copper width, thermal vias under power ICs, and placement of inductors and current sense components. Switch-mode chargers require tight layouts: minimize switch node loop area and place input/output capacitors close to the IC pins. (See charger datasheet layout recommendations.) [Texas Instruments](#)

6. Battery safety & cell monitoring

- Add NTC temperature sense, battery voltage sense, and proper charge termination. Follow the charger and gauge datasheet recommended resistor/thermistor networks. Use protection ICs (battery protector FETs or standalone protection IC) if the battery pack is removable or exposed to abuse.

7. Current sensing for the gauge

- Fuel gauges either use internal current estimation or require a sense resistor. Choose R_{sense} value to get adequate gauge accuracy vs. power loss tradeoff. Impedance-Track (BQ34Z100) provides good accuracy but needs correct R_{sense} and learning cycles. [Texas Instruments](#) +1

8. ESD and surge protection on the USB connector

- Add TVS diodes on VBUS and data lines and use good connector shielding and ground stitching. The USB-C connector ground/shield must be robust for mechanical durability.

9. USB-IF compliance & certification

- If you will market the product, confirm PD, USB-C, and BC1.2 / USB charging requirements and consider pre-compliance testing. Improper CC resistor or PD behavior can damage hosts or chargers.

10. Software / firmware integration

- Decide which device talks to the PD controller and gauge. Many PD controllers expose I²C and interrupts so the system MCU can change PDO requests or respond to VBUS events. Also build error handling: overtemp, adapter removal, role swap, and safe mode when negotiation fails.

11. Switching MUX meaning

- Clarify which MUX you want:
 - **Power MUX:** ideal diode / FET + LTC4412 style controller to select supply or prevent backfeed (discussed above).
 - **USB data / orientation MUX:** if you need to route USB2.0/USB3.1/Alt Mode lines depending on orientation, you'll need a USB-C orientation MUX (e.g., Fairchild/ON analog MUX chips or dedicated Type-C switch ICs). If you want to implement DisplayPort Alt Mode, you need a more complex switch and SuperSpeed muxing. (Search for "USB-C SuperSpeed mux" when you get to data routing.) [freescale.org](#) +1

Example minimal BOM (single-cell, sink device, moderate current)

- USB-C receptacle (5 A rated) + connector footprint, shield stitching, TVS on VBUS.
- STUSB4500 (PD/CC controller) — handles CC negotiation and PDOs. [STMicroelectron...](#)
- TI BQ25895 (switch-mode charger + power path) — battery charging and system power path. [Texas Instruments](#)
- BQ34Z100-G1 (fuel gauge) or MAX17055 as low-power alternative. [Texas Instruments](#) +1
- LTC4412 + appropriate P-MOSFET for ideal diode / OR-ing (if your charger or PD controller lacks integrated low-loss power path). [Analog Devices](#)
- Rsense, NTC, ESD TVS parts, input bulk and decoupling caps, power inductor per charger datasheet.
- Optional: MOSFET USB data/orientation MUX if you support SuperSpeed / Alt Mode.

Practical tips & step plan

1. **Decide target USB PD profile:** e.g., want only 5 V@3 A, or also 9/12/15/20 V? This determines PD negotiation and charger front end. (If only 5 V is needed, simpler CC only resistor-based designs may suffice; for >5 V you need PD controller.) [Hackaday](#)
2. **Prototype with eval boards:** get STUSB4500 EVB (or dev board) and BQ25895EVM plus BQ34Z100 evaluation kit or MAX17055 EV kit to test negotiation, charging behaviour, and gauge readings. [Mouser Electron...](#) +1
3. **Layout & thermal review:** follow datasheet reference layout for the charger; simulate thermal for worst-case charge current. [Texas Instruments](#)
4. **Implement safety features:** watchdogs for charger faults, FET short detection, and proper MOSFET sizing for current. [Texas Instruments](#)
5. **Test scenarios:** dead battery attach, plug/unplug at high load, PD negotiation failures, and overtemperature throttling.

Useful datasheets & app notes (start here)

- STUSB4500 datasheet / product page (PD controller). [STMicroelectron...](#) +1
- TI BQ25895 charger datasheet. [Texas Instruments](#) +1
- TI BQ34Z100-G1 gauge datasheet & product page. [Texas Instruments](#) +1
- MAX17055 gauge (low power alternative). [Analog Devices](#) +1
- LTC4412 ideal-diode controller datasheet (power MUX). [Analog Devices](#)
- TI app note on USB-C/PD power-path design (p1 ↓ ,cal considerations). [Texas Instruments](#)

Final quick checklist (before you start schematic)

- ☐ Single vs dual role? (sink-only is easiest)
 - ☐ Max charge current and thermal envelope defined
 - ☐ PD PDO list defined (voltages & max current)
 - ☐ Charger with power-path selected (or implement external ideal-diode)
 - ☐ Fuel gauge type chosen (Impedance-Track vs ModelGauge)
 - ☐ ESD / TVS & connector mechanical robustness
 - ☐ PCB layout plan for high current and switching node loops
 - ☐ Test/eval boards ordered for the chosen ICs
-

If you want, I can immediately:

- produce a one-page schematic-level reference (component footprints + net names) using the parts above, or
- create a more detailed BOM with Digi-Key/Mouser search links and recommended component values for a target (e.g., 3 A charging, single-cell, sink-only PD) — tell me which one you prefer and I'll generate it now.



ChatGPT Prompt 4

[4a] Prompt:

“Can you make a list of all the software and hardware required to setup my raspberry pi CM4 to ensure that I have everything installed and have all the necessary hardware before I begin?”

[4b] Response:

Can you make a list of all the software and hardware required to setup my raspberry pi CM4 to ensure that I have everything installed and have all the necessary hardware before I begin?

Got it — let's make this super clear, step-by-step, with everything you need so there's no confusion. No fluff.

🔧 What you need

1. Hardware

- Raspberry Pi CM4 (yours has 16 GB eMMC)
- CM4 IO Base B board
- USB cable (micro-USB or USB-C depending on your board, for the USB SLAVE / OTG port)
- Power supply for the IO Base B (barrel jack or USB-C)
- Optional: keyboard, mouse, HDMI display (for after the OS is installed)

2. Software (on your Windows PC)

- Raspberry Pi Imager → [Download here](#)
- rpiboot → [Direct link for Windows](#)

3. Settings on the board

- BOOT switch / jumper (labeled OFF / ON)
 - ON → USB boot mode (for flashing eMMC)
 - OFF → Normal boot from eMMC

⚡ Step-by-step flashing sequence

1. Power off the IO Base B completely.
2. Set the **BOOT switch to ON** (USB boot mode).
3. Connect USB cable from the PC to the **USB SLAVE / OTG port** on the IO Base B.
4. Power on the IO Base B (barrel jack or USB-C).
5. Run **rpiboot** on your PC.
 - The CM4's eMMC should now appear as a USB drive.
6. Open **Raspberry Pi Imager** → choose your Linux OS → select the CM4 eMMC as the target → **Write**.
7. After flashing is done:
 - Power off the IO Base B.
 - Move the **BOOT switch to OFF** for normal boot.
 - Power on again — CM4 now boots Linux from eMMC.

✅ Notes / Tips

- Do **not** flip the switch while powered on — it only works at power-on.
- If rpiboot doesn't detect the CM4:
 1. Check that USB is connected to the correct OTG port.
 2. Make sure the BOOT switch is ON.
 3. Make sure the board is powered.
- You can reflash or reconfigure anytime by repeating this process.

If you want, I can also make a **one-line cheat sheet** that you can keep in front of you while doing this — basically a “do exactly this and it will work” list.

📄 📌 🔄 ⬆️ ⬇️ ⋮

ChatGPT Prompt 5

[5a] Prompt:

“What is the best sbc compute module product line available for general purpose applications?”

[5b] Response:

what is the best sbc compute module product line available for general purpose applications?

If you're asking for the **best single-board-computer (SBC) compute module** product line for general-purpose embedded applications (rather than specialized AI/robotics usage), here's a well-rounded comparison:

🔥 Top Picks: General-Purpose Compute Modules (as of summer 2025)

Raspberry Pi Compute Module 5 (CM5)

- Latest in the Raspberry Pi CM lineup, launched late 2024 with availability expanding in 2025. Features a 2.4GHz quad-core Cortex-A76 CPU, VideoCore VII GPU supporting OpenGL ES3.1/Vulkan 1.3, PCIe 2.0, USB 3.0, Gigabit Ethernet (IEEE 1588), MIPI CSI/DSI, plus rich I/O—all in a compact form with mechanical compatibility with CM4 carriers [XDA Developers](#) [Raspberry Pi](#).
- Available in options from 2GB to (coming soon) 16GB LPDDR4X memory, and 16–64GB eMMC, with or without Wi-Fi/Bluetooth [Electronic Speccy](#) [CM5 Software](#).
- Priced competitively (~\$45–\$95 as of late 2024/early 2025), with wider availability and robust support in the Pi ecosystem [The Register](#).
- Excellent developers' support and long-term availability.

NVIDIA Jetson Orin Nano / Nano Super

- Designed around AI acceleration (RPU/tensor cores), but viable for general tasks that benefit from GPU and AI support.
- Jetson Orin Nano Super** (late 2024 entry): 67 TOPS performance, 8GB RAM, supports ROS, vision, ML workloads; kit includes carrier board and accessories (~\$249) [theverge.com](#).
- Jetson Orin Nano (standard)**: ~34 TOPS, 4-8GB RAM; lower power profiles for less demanding use cases [en.wikipedia.org](#) [theverge.com](#).
- Higher cost and more complexity—stronger if you plan on doing ML or GPU-accelerated processing, but less mainstream for typical embedded compute.

✅ Best for "General-Purpose" Applications:

Raspberry Pi Compute Module 5 stands out for:

- Balanced performance.
- Absolutely solid software stack (Raspberry Pi OS, wide driver support).
- Excellent value at low price points.
- Broad availability, ecosystem support.
- User-friendliness for developers and integrators.

💡 If your project needs GPU-accelerated AI or heavy vision, the Jetson line (Orin Nano Super/NX) is compelling—but not necessary for most general-purpose embedded systems.

Product Options (for prototyping or purchase):

Raspberry Pi Compute Module 5 (CM5)
\$84.95 • [ameiDroid](#)

NVIDIA Jetson Orin Nano Super Developer Kit
\$549.00 • [RobotShop USA](#) + 4 others

- Raspberry Pi Compute Module 5 (CM5)**: Ideal entry-level compute module in compact form, excellent value, widely supported.
- NVIDIA Jetson Orin Nano Super Developer Kit**: For applications needing AI/vision acceleration; includes carrier board and extras.

📊 Quick Comparison

Criteria	Raspberry Pi CM5	NVIDIA Jetson Orin Nano Super
Processor / GPU	Quad-core Cortex-A76 + VideoCore VII	Ampere GPU + tensor cores, ~67 TOPS
RAM / Storage options	2–16GB DRAM (eMMC, 16–64GB)	Typically 8GB RAM, eMMC optional
Connectivity / I/O	PCIe 2.0, USB 3.0, GigE, MIPI, GPIO	USB, PCIe, GigE across carrier board
Price (board only)	\$45–95 (late 2024 pricing)	Developer kit ~\$249
Ecosystem & support	Massive Pi community & docs	Good for AI developers
Best fit	General purpose, embedded, industrial	Edge AI, robotics, vision projects

🔍 Final Recommendation

For general-purpose embedded or industrial design, the **Raspberry Pi Compute Module 5** is currently the top choice. It balances performance, ecosystem, price, and flexibility, making it highly practical for a wide range of uses.

If your application demands significant neural-network processing, real-time vision, or AI acceleration, the **Jetson Orin Nano Super** may be worthwhile—but it comes at higher cost and complexity.

Let me know if you want carrier-board options, IO board advice, or comparisons with earlier CM4 or CM3 modules!

[🔗 Sources](#)

220

ChatGPT Prompt 6

[6a] Prompt:

“Do you have an article or scientific journal that explains how d-pads work?”

[6b] Response:

do you have an article or scientific journal that explains how d-pads work?

Certainly! Here's a concise overview of how **D-pads** (Directional Pads) function, along with credible sources for further reading.

🔧 How D-pads Work

A **D-pad** is a flat, cross-shaped input device commonly found on video game controllers and various electronic devices. It translates directional input (up, down, left, right, and sometimes diagonals) into electrical signals that the device can interpret.

Mechanism:

- **Structure:** Typically consists of a plastic cross resting on a central pivot point.
- **Switches:** Beneath each arm of the cross are individual switches or contacts.
- **Operation:** Pressing a direction causes the cross to tilt, activating the corresponding switch beneath it. In some designs, pressing two directions simultaneously can activate multiple switches, allowing for diagonal inputs.

Benefits:

- **Compact Design:** Offers directional control without requiring much space.
- **Precision:** Provides accurate input for 2D navigation and retro gaming.
- **Durability:** Fewer moving parts compared to analog sticks, leading to longer lifespan.

Limitations:

- **Limited Range:** Typically offers only digital (on/off) input, lacking the nuanced control of analog sticks.
- **Comfort:** Prolonged use can be less comfortable than analog sticks, especially for complex movements.

📖 Further Reading

For a more in-depth understanding of D-pads, consider the following sources:

1. **Wikipedia - D-pad:** Provides a comprehensive history and technical details of D-pads.
 - <https://en.wikipedia.org/wiki/D-pad>
2. **Noah Bannister's Development Blog:** Discusses the mechanics and design considerations of D-pads.
 - <https://noahbannister.blog/2020/01/26/examining-the-d-pad/>
3. **Digital Game Museum:** Explores the evolution of console controllers, including the D-pad era.
 - <https://www.digitalgamemuseum.org/consolecontrollerrevolution/d-pad/>

If you need diagrams, schematics, or further technical details for academic purposes, feel free to ask!

could you provide sources that are more related to electrical components rather than video game consoles. I need this for a scientific report

Certainly! Here's a detailed explanation of how **D-pads** (Directional Pads) function, focusing on their electrical components and mechanisms:

🔌 Understanding D-pad Mechanisms

A **D-pad** is a compact, cross-shaped input device commonly found on video game controllers and various electronic devices. It translates directional input (up, down, left, right, and sometimes diagonals) into electrical signals that the device can interpret.

Mechanical Design

- **Structure:** Typically consists of a plastic cross resting on a central pivot point.
- **Switches:** Beneath each arm of the cross are individual switches or contacts.
- **Operation:** Pressing a direction causes the cross to tilt, activating the corresponding switch beneath it. In some designs, pressing two directions simultaneously can activate multiple switches, allowing for diagonal inputs.

Electrical Components

- **Membrane Switches:** Many D-pads utilize membrane switch technology, where conductive traces on flexible layers form the electrical contacts. Pressing the D-pad completes the circuit, sending a signal to the device.
- **Carbon-Contact Switches:** Some controllers use carbon-coated contacts for the metal switches. These can wear out over time, leading to issues like unresponsive or "ghost" inputs Read More.

📖 Further Reading

For more in-depth information on D-pad mechanisms and designs, consider the following sources:

1. **Wikipedia - D-pad:** Provides a comprehensive history and technical details of D-pads.
 - <https://en.wikipedia.org/wiki/D-pad>
2. **Reddit Discussion on Membrane Switches:** Discusses the design and reliability of membrane switches in gamepads.
 - https://www.reddit.com/r/AskElectronics/comments/pbdoeh/why_on_gamepads_and_other_such_membrane_switch/
3. **NeoGAF Forum - D-pad Design Discussion:** A thread discussing various D-pad designs and their effectiveness.
 - <https://www.neogaf.com/threads/d-pads-thread-discuss-debate-design-image-heavy.805638/>

If you require diagrams, schematics, or further technical details for academic purposes, feel free to ask!

🔍 Sources

ChatGPT Prompt 7

[7a] Prompt:

“Give me some summation amplifiers that have an input range that includes 3.3V”

[7b] Response:

give me some summation amplifiers that have an input range that includes 3.3V

Great — if you’re looking for op-amps suitable for *summing (summer) amplifier* configurations and that can handle a 3.3 V logic/analog input range (i.e., the input common-mode should include ~0–3.3 V), here are some good choices **plus** a few notes on how to pick them for your summation circuit.

✓ Key selection criteria

Since you want the inputs (and presumably the output) to work well around 0 V up to ~3.3 V, check:

- Input common-mode range includes 0 V (or slightly below) and goes up to near the positive rail.
- Output swing is sufficient (i.e., rail-to-rail or close so you can drive the summing node to required levels).
- Supply voltage: You may power the op-amp from, say, 0–5 V or 0–3.3 V if it supports it.
- Summing amplifier configuration: you’ll arrange the resistors appropriately (inverting or non-inverting summer) and ensure the op-amp’s spec fits your amplitude swing and speed.

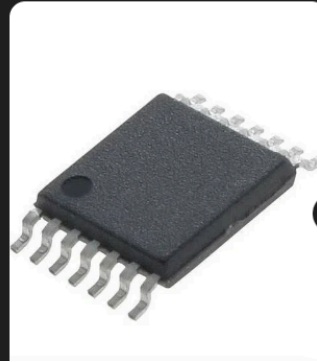
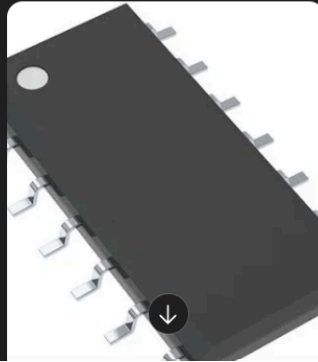
As a reminder: many older op-amps don’t allow the input to go down to the negative rail (0 V) when single-supply, so you’ll want one with rail-to-rail input or “it ↓ range includes ground”. For example, see this explanation:

"In the op-amp's datasheet ... the acceptable range of input voltages (often called 'common-mode input voltage range') ... if the negative supply is 0 V then the input must stay above some positive threshold above ground." [Electrical Engineer...](#)

Also, there are modern op-amps characterized for single supply 3.3 V or 5 V systems. [LCamtuif](#)

Recommended op-amps suitable for 3.3 V input range

Here are some specific product picks. Note: these are **just op-amps** (not full pre-wired summing amplifier modules), so you'll use them in your own summation circuit (with appropriate resistors) to build your summing amplifier.



TI LMV824 (quad rail-to-rail, low-voltage)

\$2.05 • Mouser Electronics

onsemi LMV824 (another sourcing of same)

\$1.04 • DigiKey

STMicro LMV824IY (low-voltage quad)

\$1.49 • Mouser Electronics

Here's a bit more detail on each:

- [TI LMV824 \(quad rail-to-rail, low-voltage\)](#): Operates from ~2.5 V to 5.5 V supply, input common-mode includes ground, rail-to-rail output. [Texas Instruments](#) +1
→ Good bet for 3.3 V system.
- [onsemi LMV824 \(alternate sourcing quad\)](#): Same family; you can pick the manufacturer/package you prefer. [onsemi](#)
- [STMicro LMV824IY \(low-voltage quad\)](#): Another sourcing of the same basic device. Low cost, quad package.
- [STMicro LMV824IY \(alternate listing\)](#): Same device again listed on another merchant — you can compare.
- [Analog Devices ADA4622-2 \(precision rail-to-rail\)](#): Higher performance: precision, rail-to-rail input includes negative supply, output rail-to-rail. Good if you need low offset, etc. [Analog Devices](#) +1
- [TI OPA820IDBVT \(5 V rated op-amp\)](#): Works at 5 V supply; you'd likely use it in a 5 V system (or maybe 3.3 V if qualified) but check input range.
- [OPA855IDSGR \(2.625 V min supply, high-speed\)](#): Good for low-voltage supply (2.625 V) which is even lower than 3.3 V — might be overkill if you don't need the speed, but shows the possibilities.
- [LM258P \(legacy, single supply 3 V+ but check input range\)](#): Very low cost; but note: many legacy ones don't have input range down to ground or up to rail, so you'd need to confirm if it meets your input

- LM258P (legacy, single supply 3 V+ but check input range): Very low cost; but note: many legacy ones don't have input range down to ground or up to rail, so you'd need to confirm if it meets your input swing (0–3.3 V) safely.

✂ Application to a Summation Amplifier

Since your goal is a summing amplifier (you likely want to add signals, maybe offset them, etc.), here are a few tips:

- Use an **inverting summing amplifier** configuration if you want to sum multiple inputs with fixed weights. The op-amp's inverting input ties to resistor network from your various sources; non-inverting input tied to ground (or some reference). Output = $-(R_f/R_1 * V_1 + R_f/R_2 * V_2 + \dots)$
- If you need a **non-inverting summing amplifier**, you might have to do a different arrangement (perhaps sum via resistors into a buffer then scale, or use a dedicated summer module).
- Make sure the op-amp **common-mode input range** is appropriate. If you are summing signals from 0–3.3 V, you want the op-amp inputs to be able to sit at 0 V and maybe up to 3.3 V (or slightly above) without distortion/clipping.
- Make sure the **output** can swing to the required output range. If your output also needs to cover from 0 V up to some number (maybe 3.3 V or more), then a rail-to-rail output or near-rail output is helpful.
- Pay attention to supply rails: If the op-amp is powered from 0 V & 3.3 V (i.e., single-supply 3.3 V), make sure the datasheet supports that (or at least works at that supply). If you power at 5 V, you then ensure the input and output ranges still work for your levels.

- Consider offset or biasing if you need the output to sum and produce a DC offset (for example, summing 3.3 V plus something else). In your case if summing 3.3 V + some other signal, you might design resistors accordingly and maybe add offset if needed.

🔧 My suggestion

Given your context (you've been working on embedded/microcontroller systems, likely 3.3 V logic, and need summation of analog signals), I'd recommend:

- Use the LMV824 family (e.g., TI variant) since it is clearly specified for low-voltage single supply, input range includes ground.
- Choose a variant in a package you prefer (quad, dual, single) depending on how many channels you need.
- Design your summing amplifier around it: determine your input signals, their weights (resistors), and ensure the output range meets your system (e.g., 0–3.3 V or maybe 0–5 V).
- If you need very high precision (low offset, low drift), or if you're summing many signals or doing fine instrumentation work, consider the ADA4622 family.
- Double-check the datasheet for *input common-mode range*, *output swing*, *supply voltage min/max*, *gain-bandwidth*, *slew rate* (especially if your summation has dynamic components).
- Since you're summing maybe microcontroller level signals or analog sensors, noise and offset may matter — pick accordingly.

If you like, I can **pull together a curated list of 5–10 op-amps (with full part numbers, datasheet links, cost, package options)** that are all well-suited for summing amplifiers in a 3.3 V-input-range system, and we can compare specs side-by-side (input range, output swing, supply voltage, bandwidth, cost). Would you like me to proceed with that?



ChatGPT Prompt 8

[8a] Prompt:

“Can you tell me the benefits, applications, and basic working info of LCD, OLED, and E-Ink displays?”

[8b] Response:

Basic Info on LCD, OLED, and E-Ink

LCD (Liquid Crystal Display)

How it works:

Uses a **backlight** (usually LED) that shines through liquid crystals. The crystals twist to block or allow light, forming the image.

Key traits:

- **Brightness:** High because the backlight is always on.
- **Refresh Rates:** Commonly 60–120Hz, gaming displays go higher.
- **Response Time:** 5–20 ms (slower than OLED).
- **Power Use:** Moderate; always drawing power since backlight never turns off.
- **Pros:** Cheap, bright, durable, no burn-in.
- **Cons:** Blacks aren't true black, viewing angles vary, slower pixel transitions.

Best for: Budget screens, laptops, monitors, outdoor visibility.

OLED (Organic Light-Emitting Diode)

How it works:

Each pixel lights itself individually — **no backlight**. This gives perfect blacks and extremely fast pixel switching.

Key traits:

- **Brightness:** High, but varies by model and can be limited at full-screen white.
- **Refresh Rates:** Usually 60–120Hz.
- **Response Time:** <1 ms — extremely fast.
- **Power Use:** Depends on image; dark images use very little, bright ones use more.
- **Pros:** Perfect blacks, high contrast, vibrant colors, best response time.
- **Cons:** Expensive, can suffer **burn-in**, sometimes lower full-screen brightness.

Best for: High-end phones, TVs, premium laptops, VR headsets.

E-Ink (Electronic Ink)

How it works:

Uses microcapsules with charged particles that move under an electric field. Once the pixels move into place, **they stay there without power** until the page changes.

Key traits:

- **Brightness:** Very low by itself — needs ambient light (like paper).
- **Refresh Rates:** Extremely low (0.1–10 Hz), not meant for motion.
- **Response Time:** >1 second for full page refresh.
- **Power Use:** Almost zero except when changing the image.
- **Pros:** Insanely low power, readable in sunlight, paper-like.
- **Cons:** Very slow, no color or limited color versions, low contrast in dim light.

Best for: E-readers (Kindle), signage, ultra-low-power displays.

ChatGPT Prompt 9

[9a] Prompt:

“Can you tell me the benefits, applications, and basic working info of DSI, DPI, and HDMI communication protocols?”

[9b] Response:

Basic Info on DSI, DPI, and HDMI

DSI (Display Serial Interface)

What it is:

A high-speed serial display interface defined by MIPI. Used heavily in smartphones, tablets, and compact embedded systems.

How it works:

- Uses 2–4 differential data lanes + 1 clock lane
- Transfers pixel data as high-speed packets
- Designed for low power + minimal wiring

Key traits:

- **Connector:** Thin flat-flex cables (FFC/FPC)
- **Complexity:** Moderate; requires proper initialization sequences
- **Latency:** Low
- **Power:** Moderate
- **Resolution:** Commonly up to 1440p on embedded devices
- **Pros:** High bandwidth with few wires, great for small high-res panels
- **Cons:** Requires careful driver setup; panels aren't plug-and-play

Common uses: Phones, portable game systems, Raspberry Pi DSI displays.

DPI (Display Parallel Interface)

What it is:

A **parallel RGB interface** often used by microcontrollers and early Raspberry Pi-compatible panels. It sends each pixel's color channels over multiple wires at once.

How it works:

- Up to **24 separate signals** (8 bits each for R, G, B)
- Plus HSYNC, VSYNC, DE, and pixel clock
- Essentially raw pixel timing

Key traits:

- **Connector:** GPIO header or ribbon through custom PCB
- **Complexity:** High due to many signals that must be perfectly timed
- **Latency:** Low
- **Power:** Higher because you're toggling many GPIO lines at MHz speeds
- **Resolution:** Practical limit ~720p–1080p (Pi maxes ~1080p in theory)
- **Pros:** Very low latency, simple protocol rules
- **Cons:** Uses many pins; susceptible to signal integrity issues

Common uses: Microcontroller displays, dev boards, older LCD modules.

HDMI (High-Definition Multimedia Interface)

What it is:

A consumer-friendly digital video/audio interface using **TMDs differential pairs**. Basically plug-and-play for TVs, monitors, and high-resolution displays.

How it works:

- Uses high-speed serial differential pairs
- Carries video, audio, EDID, and control signals
- Display tells the source what formats it supports

Key traits:

- **Connector:** HDMI or micro-HDMI
- **Complexity:** High at the electrical level, but user-friendly
- **Latency:** Very low
- **Power:** Highest of the three
- **Resolution:** Up to 4K60 (and much higher on newer versions)
- **Pros:** Standardized, widely supported, plug-and-play
- **Cons:** Requires HDMI PHY hardware; not used for very small panels

Common uses: TVs, monitors, Raspberry Pi output, cameras, game consoles.

ChatGPT Prompt 10

[10a] Prompt:

“Why can USB have more latency when compared to other audio transmission protocols?”

[10b] Response:

Here is a clean, direct explanation of **why USB audio typically introduces more latency**, broken down into the specific “pieces” you can use in a paper or technical write-up.

✓ Why USB Audio Has More Latency (Compared to I²S or Analog)

1. USB Is a Packet-Based Protocol

USB does **not** stream audio as a continuous signal.

Instead, it sends audio frames in **periodic packets** (typically every 1 ms for Full Speed, 125 µs for High Speed).

Because of this:

- Audio data must be **buffered** before transmission.
- The receiving device must **buffer again** to reassemble packets.

Result: Added buffering = added latency.

2. USB Requires Host Scheduling

USB is **host-controlled**, meaning:

- The device cannot send audio whenever it wants.
- The host (PC, SBC, etc.) polls the device on a schedule.

Even in isochronous mode:

- The host reserves bandwidth and requests data each interval.
- Scheduling adds predictable but unavoidable delay.

3. USB Audio Uses Larger Buffers to Avoid Dropouts

To prevent underruns or overruns, USB audio stacks often use:

- 2–10 ms buffers (Windows)
- 1–5 ms buffers (Linux/PulseAudio/PipeWire)
- 10+ ms on SBCs with weaker CPUs

Because USB can burst or stall due to USB bus activity, larger buffers stabilize audio—but increase latency.

4. USB Drivers and OS Layers Add Software Latency

USB audio passes through:

1. USB controller
2. Kernel USB driver
3. USB Audio Class driver
4. ALSA or CoreAudio or WASAPI layer
5. Mixer/Resampler (if enabled)
6. Audio rendering engine
7. DAC interface

Each layer adds small but cumulative delays.

Compare this to I²S:

- Direct digital stream
- No packetization
- No host polling
- No USB stack

Thus I²S latency is typically **tens of microseconds**, while USB is **milliseconds**.

5. USB Needs Clock Recovery (Asynchronous Mode)

Modern USB DACs often use **asynchronous USB audio**, where:

- The DAC runs on its own clock
- It must report its buffer fill state
- The host adjusts packet rates dynamically

This closed-loop control introduces:

- Timing adjustments
- Buffer smoothing delays

Typical Latency Numbers

These are widely accepted engineering ranges:

Interface	Typical Latency
Analog output	<1 ms
I ² S DAC	~0.02–0.3 ms
Bluetooth SBC codec	150–250 ms
Bluetooth aptX LL	30–40 ms
USB Audio Class 1/2	2–10 ms (common), can be 20+ ms on SBCs

USB is higher than I²S, lower than Bluetooth.

If you want sources, here are trusted statements you can quote:

USB Implementers Forum (USB-IF)

While they don't give a single number, they explain packetized interval-based transfers:

"Isochronous transfers occur within a reserved time during each frame, providing guaranteed latency and bandwidth."

Meaning: latency *exists* and is tied to the frame timing.

Texas Instruments – PCM29xx USB Audio DACs

TI documents explicitly state buffering-induced latency:

"USB audio data is transmitted in packetized frames, requiring buffering in both the host and the device."

XMOS (well-known USB audio silicon provider)

"Host scheduling and packet buffering introduce latency which must be managed within the USB audio pipeline."

Linux ALSA Documentation

Describes USB audio latency:

"USB audio inherently has more latency due to its periodic packet transfers and required buffering."

If you want, I can provide the PDF locations or exact document sections.

APPENDIX D: WORKING CODE:

****Results for this section can be seen in Chapter 9: Prototype and System Testing****

GUI Backend:

```
def _make_app_tile(self, app):
    box = Gtk.Box(orientation=Gtk.Orientation.VERTICAL, spacing=4)
    icon_widget = None

    icon_file = app.get("icon_path")

    if icon_file and os.path.isfile(icon_file):
        icon_widget = Gtk.Image.new_from_file(app["icon_path"])
    else:
        icon_name = app.get("name") or app.get("icon") or "application-x-executable"
        icon_widget = Gtk.Image.new_from_icon_name(icon_name, Gtk.IconSize.DIALOG)

    lbl = Gtk.Label(label=app.get("name", "App"))
    lbl.set_max_width_chars(10)
    lbl.set_ellipsize(3) # PANGO_ELLIPSIZE_END=3

    box.pack_start(icon_widget, True, True, 0)
    box.pack_start(lbl, False, False, 0)

    eb = Gtk.EventBox(); eb.add(box)
    eb.connect("button-press-event", lambda _: self.launch_app(app))
    return eb

def _make_spacer_tile(self):
    box = Gtk.Box(orientation=Gtk.Orientation.VERTICAL); box.set_sensitive(False)
    return box

def launch_app(self, app):
    print("Launch:", app["name"])
    # TODO hook into your launch mechanism

# ----- C. Dots -----
def _rebuild_dots(self, num_pages):
    for child in list(self.pager_box.get_children()):
        self.pager_box.remove(child)

    for i in range(num_pages):
        dot = Gtk.Label(label="•" if i == 0 else "◻")
        dot.set_name("pager-dot")
        dot.set_margin_left(3)
        dot.set_margin_right(3)

        dot.add_events(Gdk.EventMask.BUTTON_PRESS_MASK)
        dot.connect("button-press-event", self._on_dot_clicked, i)

        self.pager_box.pack_start(dot, False, False, 3)
    self.pager_box.show_all()

def _on_dot_clicked(self, widget, event, index):
    children = self.stack.get_children()
    if not children:
        return
    index = max(0, min(index, len(children)-1))
    self.stack.set_visible_child(children[index])

def _on_page_changed(self, *):
    idx = self._current_index()
    for i, child in enumerate(self.pager_box.get_children()):
        if isinstance(child, Gtk.Label):
            child.set_text("•" if i == idx else "◻")

if __name__ == "__main__":
    Launcher().run([])
```

```
def goto_next_page(self):
    children = self.stack.get_children()
    if not children:
        return
    i = self._current_index()
    if i < len(children) - 1:
        self.stack.set_visible_child(children[i+1])

def goto_prev_page(self):
    pages = self.stack.get_children()
    if not pages: return
    i = self._current_index()
    self.stack.set_visible_child(pages[max(i-1, 0)])

def _current_index(self):
    child = self.stack.get_visible_child()
    if child is None:
        return 0
    name = self.stack.get_visible_child_name()
    if name and name.startswith("page-"):
        try:
            return int(name.split("-", 1)[1])
        except ValueError:
            pass
    children = self.stack.get_children()
    try:
        return children.index(child)
    except ValueError:
        return 0

# ----- B. Dynamic pages -----
def _rebuild_pages(self):
    # clear existing pages
    for child in list(self.stack.get_children()):
        self.stack.remove(child)

    chunks = [self.apps[i:i+PAGE_SIZE] for i in range(0, len(self.apps), PAGE_SIZE)] or [[]]
    for idx, chunk in enumerate(chunks):
        page = self._make_page(chunk)
        self.stack.add_named(page, f"page-{idx}")

    self.stack.set_transition_type(Gtk.StackTransitionType.SLIDE_LEFT_RIGHT)
    self.stack.set_transition_duration(250)
    self.stack.set_visible_child_name("page-0")

    self._rebuild_dots(len(chunks))
    self.stack.show_all()

def _make_page(self, app_chunk):
    flow = Gtk.FlowBox()
    flow.set_selection_mode(Gtk.SelectionMode.NONE)
    flow.set_max_children_per_line(COLS)
    flow.set_min_children_per_line(COLS)
    flow.set_row_spacing(8)
    flow.set_column_spacing(8)
    flow.set_valign(Gtk.Align.FILL)
    flow.set_halign(Gtk.Align.FILL)

    for app in app_chunk:
        flow.add(self._make_app_tile(app))

    for _ in range(PAGE_SIZE - len(app_chunk)):
        flow.add(self._make_spacer_tile())

    return flow
```


ESP32 Sensor Communication Backend:

```
1  /*
2  * @file heartrateAnalogMode.h
3  * @brief This is written for the heart rate sensor the company library. Mainly used for real
4  * @n time measurement of blood oxygen saturation, based on measured values calculate heart rate values.
5  * @copyright Copyright (c) 2010 DFRobot Co.Ltd (http://www.dfrobot.com)
6  * @license The MIT license (MIT)
7  * @author [linfeng](Musk.ling@dfrobot.com)
8  * @maintainer [qsjhy](yihuan.huang@dfrobot.com)
9  * @version V1.0
10 * @date 2022-04-26
11 * @url https://github.com/DFRobot/DFRobot_Heartrate
12 */
13
14 #include "DFRobot_Heartrate.h"
15 #define HEARTRATE_PIN 4
16 #define SAMPLING 10
17
18 //Beat detection parameters
19 #define THRESHOLD 20
20 #define BASE_ALPHA 5 //Baseline smoothing (% per sample)
21
22 //Acceptable beat interval range (ms)
23 #define MIN_INTV 350 //~171 BPM upper bound
24 #define MAX_INTV 2000 //~30 BPM lower bound
25
26 int lastSample, lastBeat, baseline, bpm = 0;
27 bool above = false;
28
29 #include <Arduino.h>
30
31 // Define the pin where the LDR is connected (e.g., GPIO 34)
32 const int ldrPin = 5;
33
34 #include "Adafruit_Sensor.h"
35 #include "Adafruit_AM2320.h"
36
37 Adafruit_AM2320 am2320 = Adafruit_AM2320();
38
39 void Setup() {
40     Serial.begin(115200);
41     delay(1000);
42
43     analogReadResolution(12); //0-4095 on the ESP32
44     analogSetPinAttenuation(HEARTRATE_PIN, ADC_11db);
45
46     //Take samples to initialize a baseline
47     long sum = 0;
48     const int N = 200;
49
50     for (int i = 0; i < N; i++) {
51         sum += analogRead(HEARTRATE_PIN);
52         delay(5);
53     }
54
55     baseline = sum / N;
56     // Set the pin mode (though analogRead works without explicit pinMode for ADC pins)
57     pinMode(ldrPin, INPUT);
58     am2320.begin();
59 }
60
61 void loop() {
62     unsigned long now = millis();
63
64     if(now - lastSample < SAMPLING) {
65         return; //Wait until the next sample time
66     }
67
68     lastSample = now;
69
70     int raw = analogRead(HEARTRATE_PIN);
71
72     //Exponential moving average for baseline
73     baseline = (BASE_ALPHA * raw + (100 - BASE_ALPHA) * baseline) / 100;
74
75     int diff = raw - baseline; //Will be positive when above baseline
76
77     //Peak detection
78     //If it crosses ABOVE the threshold, its a "beat"
79
80     if(above && diff > THRESHOLD) {
81         above = true;
82
83         if(lastBeat != 0) {
84             unsigned long interval = now - lastBeat;
85
86             if(interval > MIN_INTV && interval < MAX_INTV) {
87                 bpm = 60000 / interval;
88             }
89             lastBeat = now;
90         }
91
92         if(above && diff < 0) {
93             //If it falls back below the baseline -> end of beat window
94             above = false;
95         }
96
97         //Serial plotter output
98         Serial.print(raw);
99         Serial.print(" ");
100         Serial.print(baseline);
101         Serial.print(" ");
102         Serial.print(diff);
103         Serial.print(" ");
104         Serial.println(bpm);
105
106         delay(200);
107
108         // Read the analog value from the LDR pin (value between 0 and 4095)
109         int ldrValue = analogRead(ldrPin);
110
111         // Print the value to the Serial Monitor
112         Serial.print("LDR Value: ");
113         Serial.println(ldrValue);
114
115         // Add a small delay before the next reading
116         delay(500); // Wait for 0.5 seconds
117
118         Serial.print("Temp: "); Serial.println(am2320.readTemperature());
119         Serial.print("Hum: "); Serial.println(am2320.readHumidity());
120
121         delay(2000);
122     }
123 }
```

EXTRA CREDITS & THANK YOUS:

For the greatest amount of advice and patience with our group, we would like to thank Dr. Weeks, Dr. Chan, and all of our reviewers for their support and help throughout this project.

KIPS Mk II Mobile Iconography:

Madison Garrett (ArtStation: www.artstation.com/mlgarrett177)

KIPS Mk II Housing Printing:

Gabriel Crowe at Folded Prints LLC (Instagram: [@folded_prints](https://www.instagram.com/folded_prints))

Additional Insight and Support:

Dr. Arthur Weeks, Dr. Chung Yong Chan, Dr. John Aedo, Mark Maddox, and Dr. Chris Iannello.

Lastly, we would like to thank our friends and above all our families for their unwavering support during this herculean task of a project. We could not have done it without them whatsoever.