K.I.P.S. Mk II

Knights Information Processing System Mk II

UCF | College of Engineering and Computer Science

UNIVERSITY OF CENTRAL FLORIDA

(Placeholder Image^^^)

Authors (Group 16):

Harrison Mills	Computer Engineering Comprehensive
Scott Valentine	Electrical Engineering RF/MW, Computer Science Minor
Jonathan Meyers	Computer Engineering Comprehensive
Jessenia Argueta	Computer Engineering Comprehensive

Mentors:

Dr. Chung Yong Chan

Sponsored By:

(TBD)

Reviewed By:

Dr. Sonali Das, Dr. Nazanin Rahnavard, Mark Maddox, (TBD)

Table of Contents:

List of Figures:

List of Tables:

1) Executive Summary: (TBA)

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 still having 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

System Specifications	
Desktop Battery Life	**> 2 Hours**
Mobile Battery Life	**> 5 Hours**
Dimensions	**~ 6 in. x 4 in. x 1.5 in.**
Weight	**< 2 lbs**
Mobile to Desktop Conversion Time	**< 20 seconds**
Desktop to Mobile Conversion Time	**< 20 seconds**
Cost Per Unit	**< $300**
Power Consumption (Mobile)	**< 7 Watts**
Power Consumption (Desktop)	**< 15 Watts**
Boot Time	**< 10 seconds**
Avg. System Operating Temp.	**95°F - 120°F**
Response time/ time delay	**10 seconds?**

Table 2.2: Component Specifications

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)**

Component Specifications:		
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

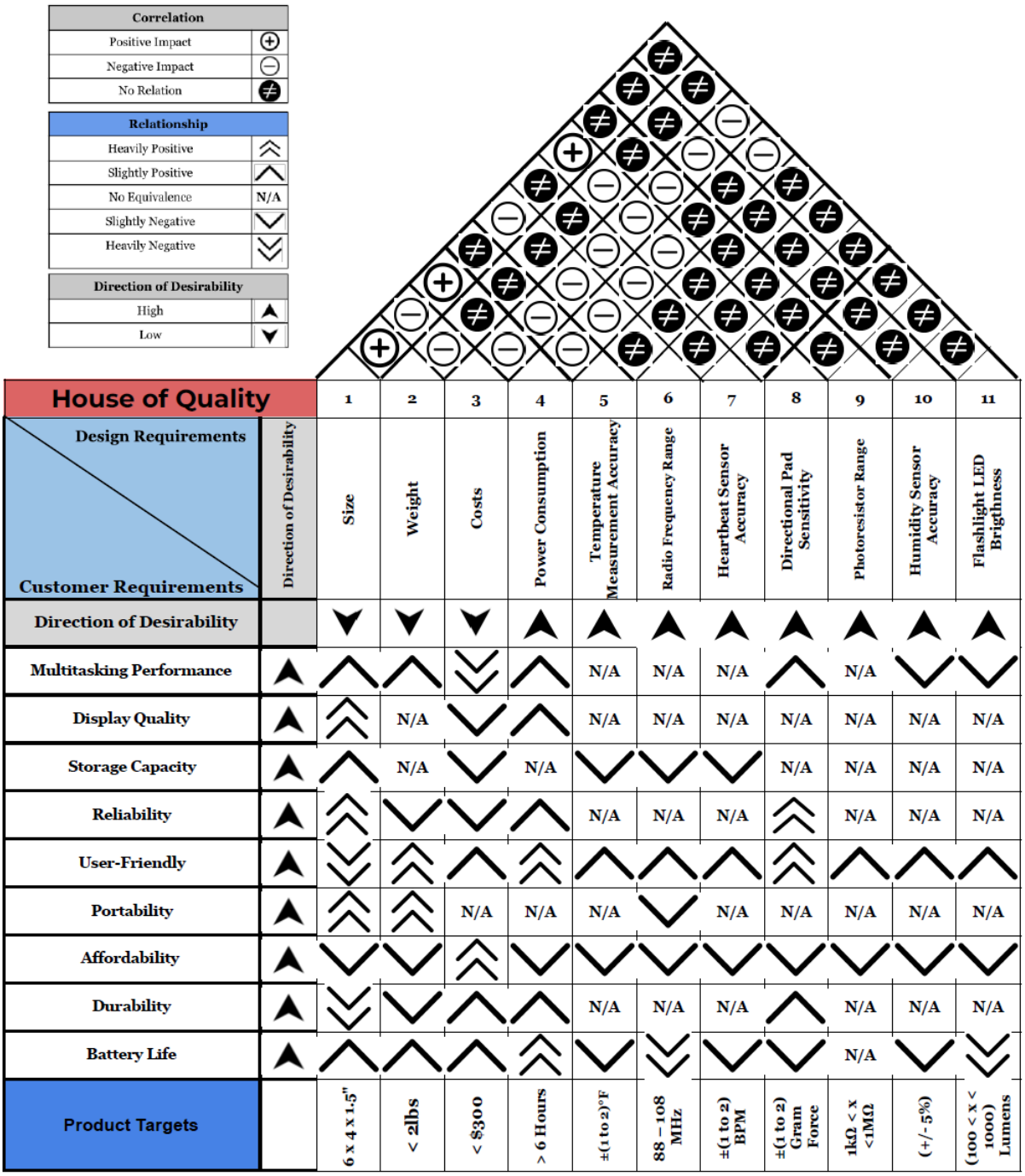


Figure 2.1: House of Quality

2.6.1) Hardware Block Diagram

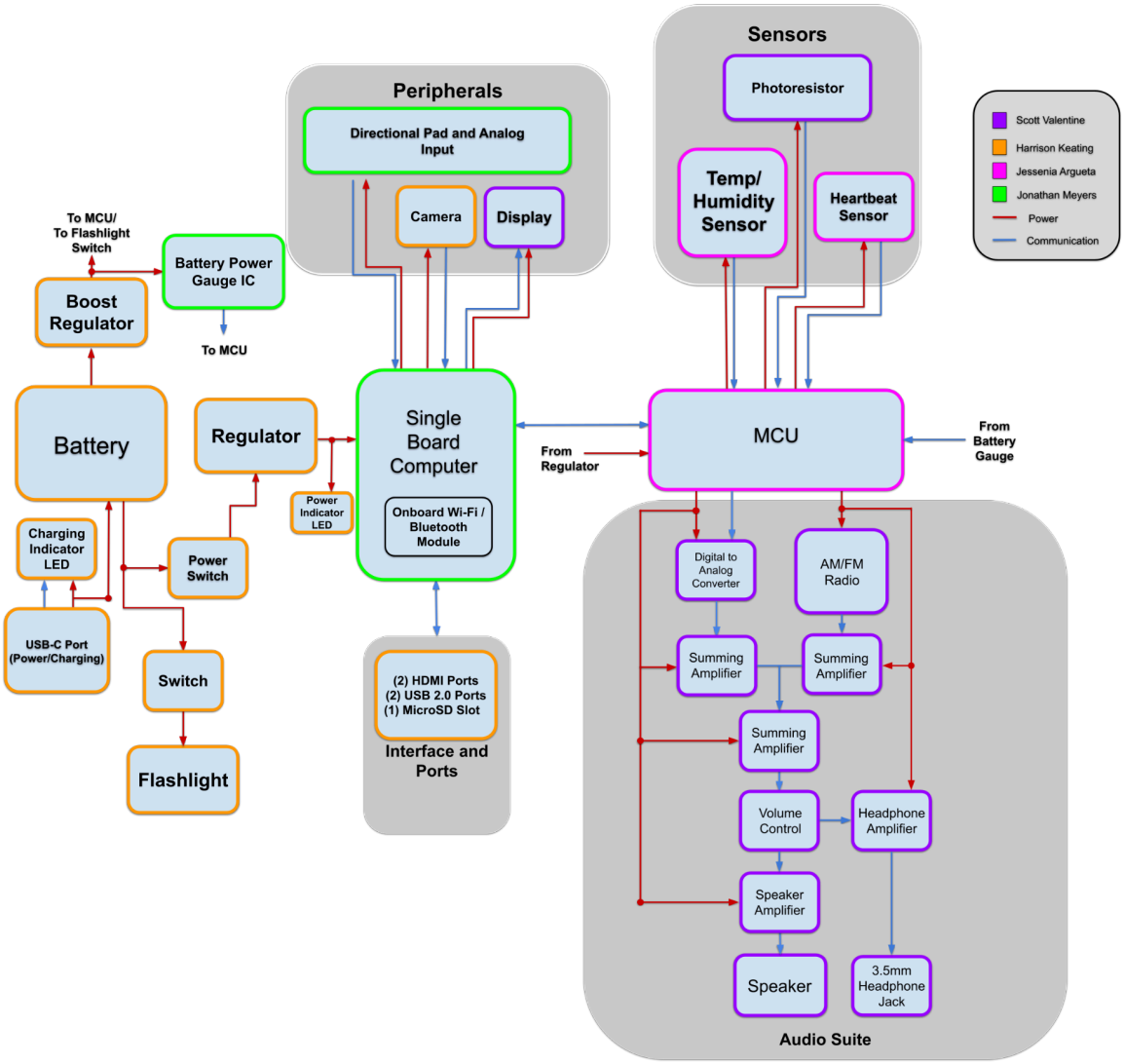


Figure 2.2: Hardware Block Diagram

2.6.2) Software Block Diagram

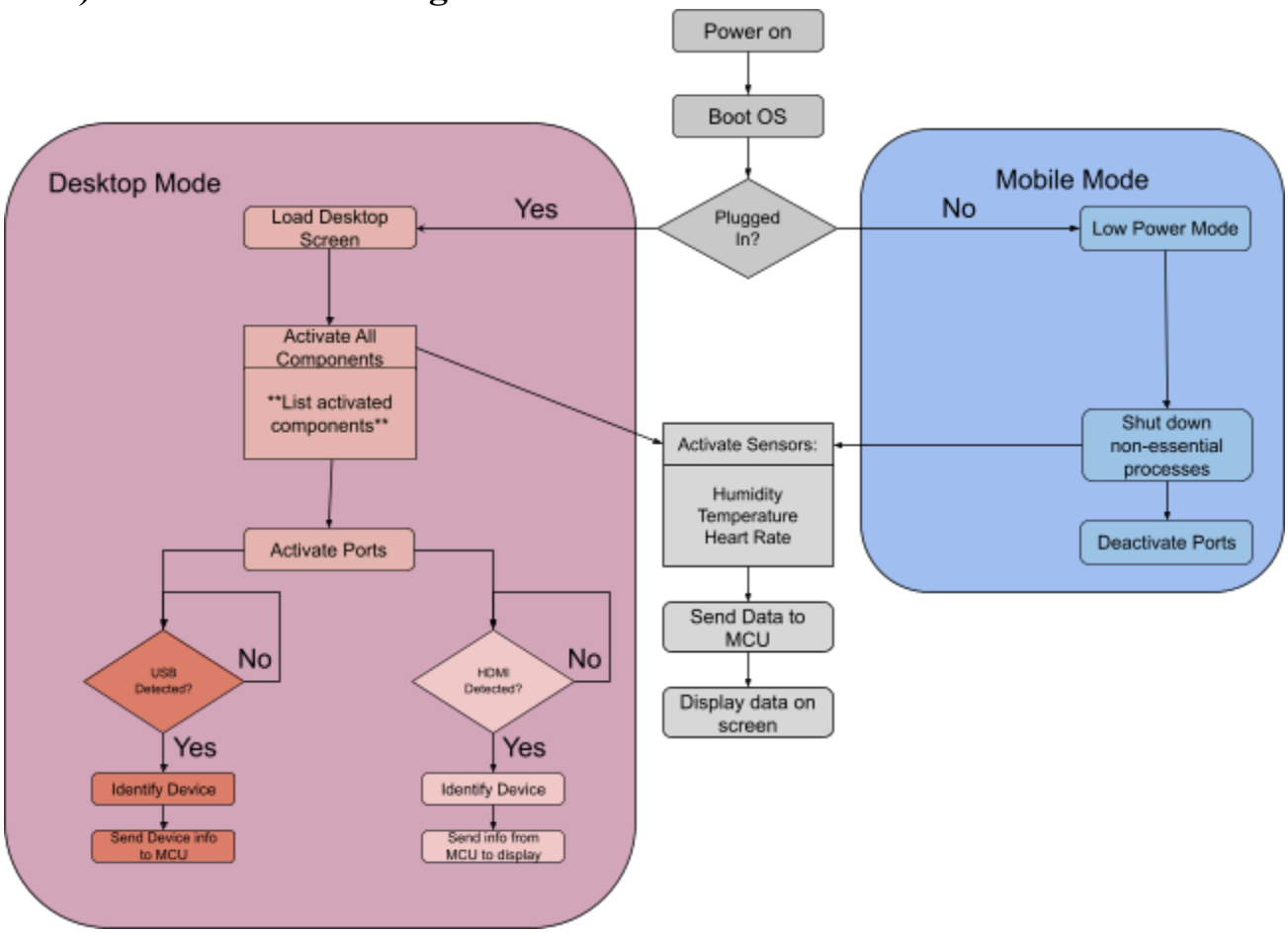


Figure 2.3: Software Block Diagram

(Belongs in Administrative Content After Proposal)

2.7) Financing and Budgeting

Table 2.3: Budget

Categories	Budget
PCBs/Electrical Parts	~$530
Materials	~$110
TOTAL	~ $640

2.8) SD1 and SD2 Milestones

Table 2.4: Milestones

Task	Timeline	Status
Senior Design 1 and Documentation		
Project Idea Document Brainstorm		
Project Selection		

3) Research

In Chapter 3, for each hardware and software part, discuss technology comparison & selection followed immediately by part/product comparison and selection for the selected technology

In general, start with hardware, followed by comm protocol and then software.

In hardware, PSU goes last. Present power distribution/requirement of the overall system using a 3-column table (component, supply voltage, etc.). Then, discuss comparison and selection of energy sources followed by technology/part comparison and selection on components such as converters/regulators/controllers, etc.

MCU, SBC, FPGA
***talk about upside and downsides of each, then compare them together in table format. After comparison make a selection; in this case we chose MCU and SBC, then JUSTIFY.**
Immediately after, then compare the different producers, i.e. MCUs, such as STM32, … then explain why to use the specific part in general (ESP32).

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, FGPAs 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 security 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.X: SBC Product Line 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		Yes

Boot Time		Fast	Slow
Price Point	High	Low	Moderate
Form Factor	Large	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. 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. 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.

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.X: 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, I2C	UART, SPI, I2C	UART, SPI, I2C
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.

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.

Table 3.X: 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.

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. 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.

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.X: Raspberry Pi Compute Module Model Comparison

	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
I2C Support	≤ 6 I2C	≤ 5 I2C	≤ 5 I2C
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 the KIPS device. It needed to be physically small [insert max size dimensions here], draw no more than [insert max MCU power draw value here], and have enough computing power overall 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.

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 [1]. 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.

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 [2]. 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.

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 [3]. 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.

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 [2] [3]. 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.

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 [4]. 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.

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 ~0.9μA - 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 [5]. 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.

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 0.16µA - 3.1mA. Additionally, it is very small, at around 7x7mm [6]. 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.

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.

Table 3.X: 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	240	133	64-128	160 (max)

(MHz)				
Power Consumption (min-max)	~2.5µA - 260mA	~50mA-150mA	~0.9µA - 5.1mA	~0.16µA - 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 (On DK/board)	USB FS + OTG FS
GPIO Pins	57 (max)	40 (max)	48 (max)	136 (max)
Comm. Protocol(s)	UART, SPI, I2C, I2S, CAN, PWM	UART, SPI, I2C, PWM, PIO	UART, SPI, I2C, PWM	UART, SPI, I2C, FDCAN
Wi-Fi	Yes (4)	Yes (4)	No	No
Price	$6-16	$4	$12	$6-12

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.

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. This makes ECG a less wearable-friendly method, but it is not impossible to implement.

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.

3.3.1.2) Heartbeat Sensor (Jess)

The first of the sensors to be covered is the heartbeat sensor. The idea behind using the 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 to be displayed 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.

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 (SpO_2).

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.

The final sensor that was selected is the Gravity Heart Rate Monitor, which uses the SON1303 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.

Ultimately, the sensor that was determined to work best with the KIPS device was the SON1303 sensor in the Gravity PPG Heart Rate Monitor. 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 operating range was below the specifications needed for KIPS to operate properly. The SON1303 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 SON1303 proved to be the most convenient to use without costing an unreasonable amount.

	MAX30102	The Original PulseSensor Kit	Gravity Heart Rate Monitor (SON1303)
Input Voltage (V)	1.8 - 3.3	3 - 5.5	3.3 - 5
Operating Current	0.7μA - 600μA	3mA - 4mA	< 10mA
Output Data Type	Heart rate + SpO_2	Heart rate	ECG waveform
Size (mm)	5.6 x 3.3 x 1.55	15.8	28 x 24
Measurement Method	PPG	PPG	PPG
Cost ($)	5.41	24.99	16.00

3.3.2.1) Light detection
mention photoresistor, photodiode, etc.

3.3.2.2) Photoresistor (Scott)

(Section about how photoresistor sensitivity and range is determined)

	VT935G	GL5537 LDR	GL5528 LDR
Resistance Range	200Ω < x < 1MΩ	1kΩ < x < 2MΩ	500Ω < x < 1MΩ
Lux Range	1 < x < 10k	1 < x < 10k	10 < x < 100k

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.

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 supply 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.

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.

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.

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), the resistive conductive polymer/salt, and the thermal/dew-point style. 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.

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.

3.3.3.3) Temperature and Humidity Sensor (Jess)

The temperature and humidity sensor will be placed on the exterior of the the device, as one component which measures two see

	AM2301B	SHT30A-DIS-B + SF2 Filter Cap	HTU21D(F) + Filter
Input Voltage (V)	2.2 - 5.5	2.4 - 5.5	1.5 - 3.6
Operating Current	250 nA	600µA - 1500µA	300µA - 500µA
Output Data Type	°C, %RH	°C, %RH	°C, %RH

Size (mm)	58.8 x 26.8 x 13.2 w/ enclosure	2.5 x 2.5 x 0.9	3 x 3 x 0.9 mm
Measurement Method (temperature)	NTC thermistor	Silicon bandgap	Silicon bandgap
Measurement Method (humidity)	Capacitive Polymer RH	Capacitive Polymer RH	Capacitive Polymer RH
Communication Protocol	Single-wire DHT	I2C	I2C
Cost ($)	7.95	3.21 + 1.26	10.95

3.4) Peripherals

3.4.1.1) Physical Interface

** Talk about general technologies, like buttons, d-pad, joystick, rotary encoder, etc.**
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.

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.

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.

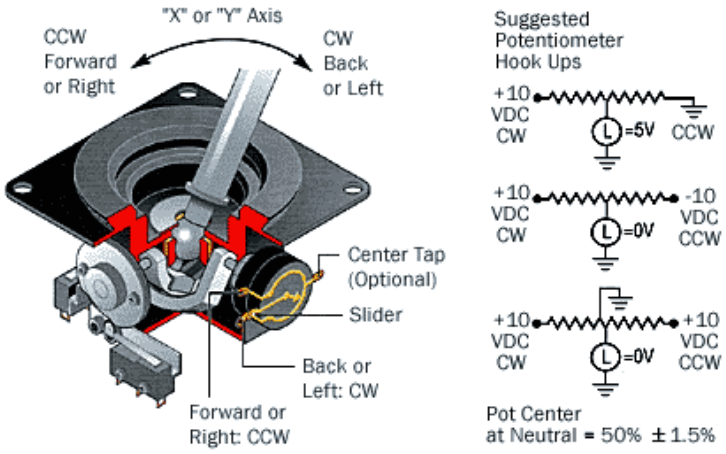
D-pads present several different benefits as opposed to other comparable technologies. D-pads have a very compact design with a low-profile which would not detract from the overall footprint of the device. Additionally, d-pads are very low cost and highly durable. D-pads are typically manufactured using 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.



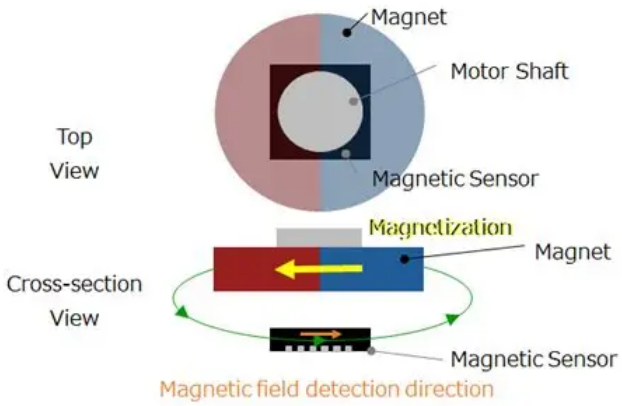
[Joystick] Potentiometer Configuration
Shows the internals of a joystick, including the potentiometer configuration and how that affects the equivalent electrical configuration.

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 . 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. 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.

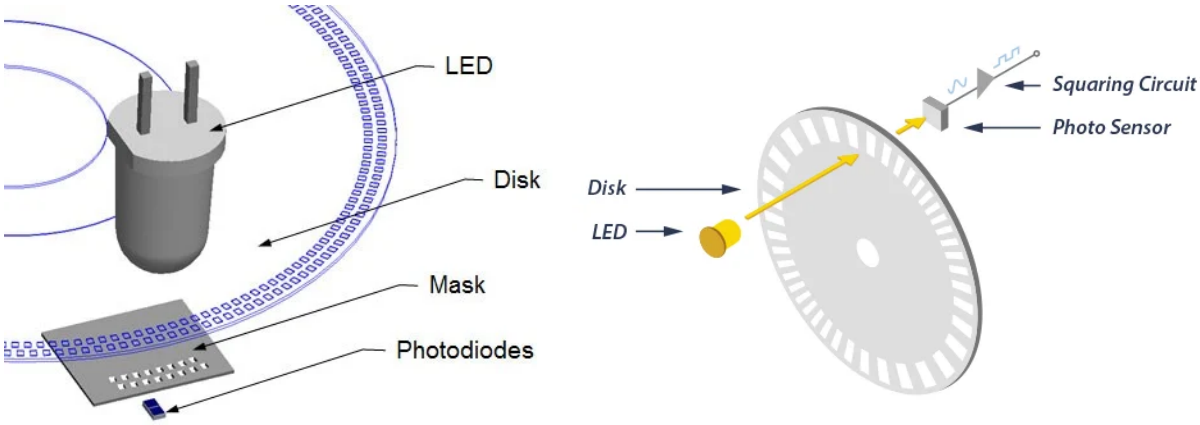
Rotary Encoder
Another option we considered was the rotary encoder. Rotary encoders are typically used for continuous output along a singular axis. A rotary encoder is a sensor that determines the angular position of a rotating shaft. There are several different types of rotary encoders, split based 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. 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.



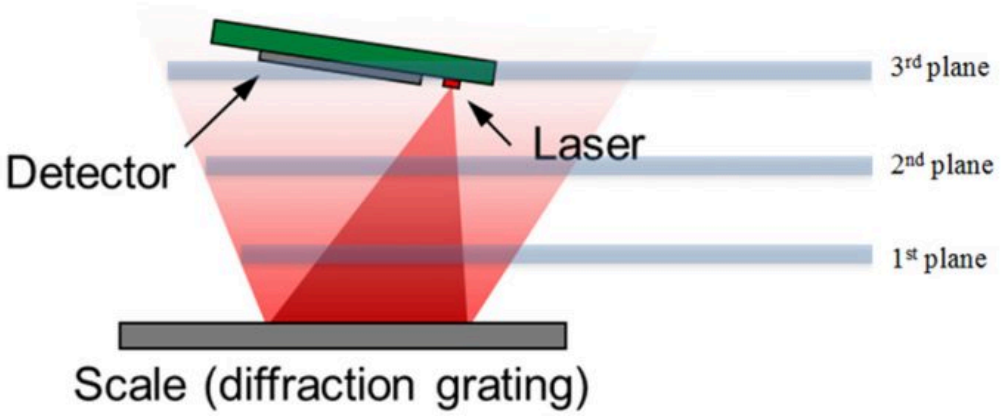
Magnetic rotary encoder diagram

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.



Optical Encoder

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.



Laser Encoder

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

	Buttons	D-pad	Joystick	Rotary Encoder

****insert table to compare technologies****
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℃ - 85℃. 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 (1.77N ± 0.59N) 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

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.

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℃ to 85℃. 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.

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.X: 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℃ - 85℃	-40℃ - 85℃	-25℃ - 85℃
Voltage Rating	12V	1V - 12V DC	12V
Current Rating	50mA	10μA - 50mA	100mA
Operating Force (Directional)	1.77N ± 0.59N	1.2N ± 0.69N	1.77N ± 0.34N
Operating Force (Center-Push)	3.14N (+0.78N / -0.59N)	2.35N ± 0.69N	1.77N ± 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

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 10kΩ 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℃ - 70℃.

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.

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.

Lastly, is the KY-023 joystick module. This part is also a two-axis analog design and uses two 10kΩ potentiometers, with one across each axis and is equipped with a built-in push-button. This part's temperature range (0℃ - 70℃) 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.

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.X: 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℃ - 70℃	0℃ - 70℃	Not specified	0℃ - 70℃
Voltage Rating	≤ 5V	≤ 10V	3.3V-5V	5V
Current Rating	≤ 50mA	~1mA	~10mA	≤ 50mA
Operating Force (Directional)	1.8N ± 0.5N	Not specified	Not specified	Not specified
Operating Force (Center-Push)	4.0N ± 0.8N	Not specified	Not specified	Not specified
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

3.4.3) Camera

The camera in our build was more like a plug-n-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[7]st reduced. [7]

Table 3.X: Camera Model Comparison

Camera Choice	IQL-IMX335/FF	HM01B0-MNA-00FT870 (Our Choice)
Cost	$50 (+$12.99 Shipping Fee)	$11.74
Resolution	2626 x 1960	324 x 324
Interface	MIPI	I2C
Power Consumption	DC 5V, 150mA (~0.75W or 750mW)	8-bit, QVGA 60FPS <4mW
Size / Dimension	0.866" L x 0.591" W (22.00mm x 15.00mm)	0.787" L x 0.492" W (20.00mm x 12.50mm)
Height - Seated (Max)	0.902" (22.90mm)	0.133" (3.39mm)

3.4.4.1) Screens and Protocols

Table 3.X: Screen Types

Table 3.X: Screen Communication Protocols

3.4.4.2) DSI LCD Screen (Scott)

3.5) Audio Suite

KIPS MkII 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 Handling (Scott)

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 (reference RPI CM4 datasheet). 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.

Digital-to-Analog Converters have many benefits in a portable project like ours. DACs directly connect to the audio bus of the source, which results in several 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.X Cadence) due to longer traces being more prone to noise, as well as signal timings possibly being thrown off (reference Cadence article here). 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.X: "Rule-of-Thumb Guidelines for PCB Signal Propagation Delay and Trace Length Matching" (reference Cadence article)

Another option for dealing with the digital output of the DAC 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.X 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.X: "Figure 1 - A collision taking place on channel #2") and transmitter/reciever synchronization (Figure 3.X: "Figure 2 - Missed packets in unsynchronised communication") (reference Bluetooth reliability manual). 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.

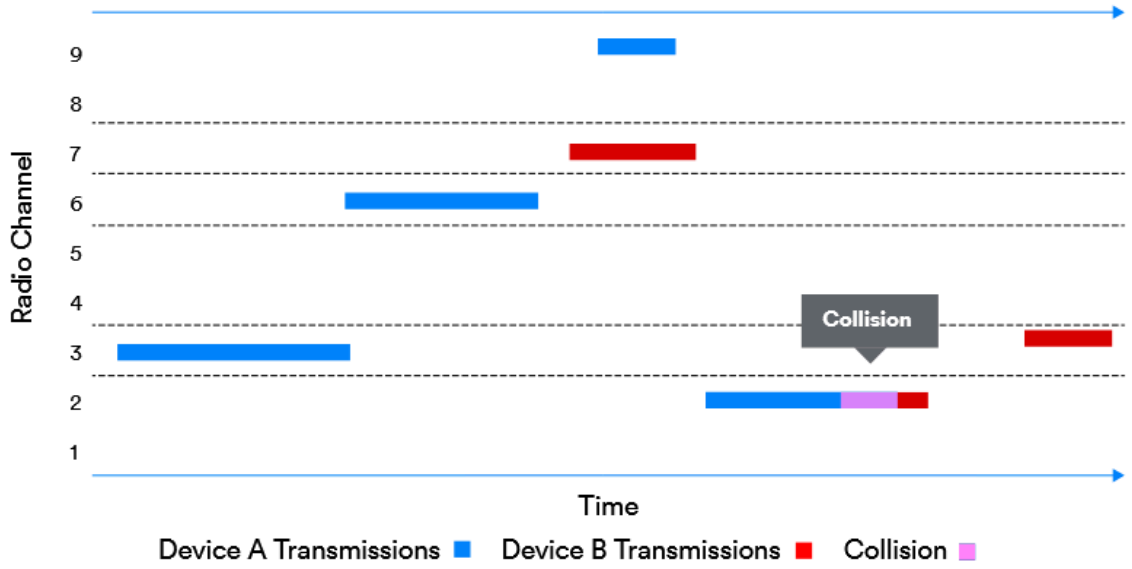


Figure 3.X: "Figure 1 - A collision taking place on channel #2" (reference bluetooth manual)

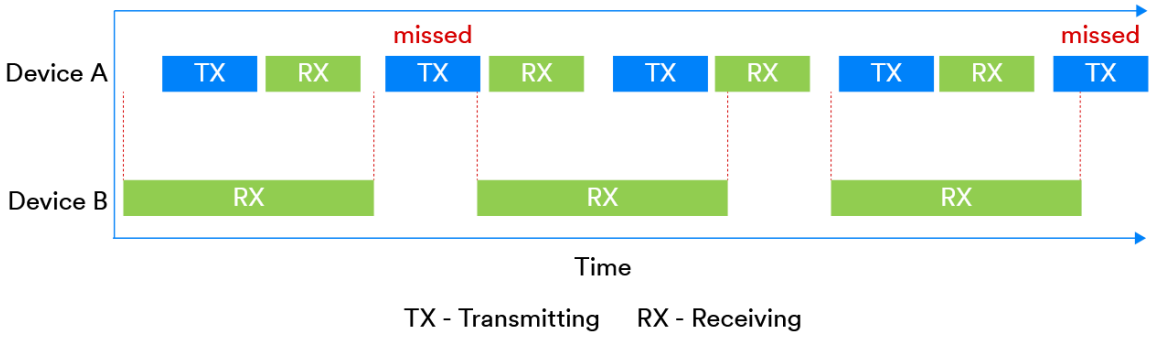


Figure 3.X: "Figure 2 - Missed packets in unsynchronised communication" (reference bluetooth manual)

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.

(WRITE AT BEGINNING ABOUT TRANSMISSION PROTOCOLS THEN COME DOWN HERE AND MAKE A TABLE)

Table 3.X: Signal Transmission Options

	DAC IC	USB Interface	Bluetooth DAC IC
Latency	Lowest Latency	Average Latency	Highest Latency
Customizability	Very Little Customization	Very Little Customization	Very Little Customization
Complexity	Least Complex	Somewhat Complex	Most Complex
Connection Type	Integrated on PCB	Integrated on PCB	Integrated on PCB w/ Wireless Connection
Footprint	Small Footprint	Bulky Footprint	Bulky Footprint
Efficiency	Most Efficient	Somewhat Efficient	Least Efficient
Protocol	Analog (No Protocol)	USB (Digital Protocol)	Bluetooth (Digital Protocol)

Given the above factors and research, we have decided to go with a 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) Digital to Analog Converter (DAC)

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(reference cs4334). The CS4334 also communicates using the I2S we need; with a resolution of 24 bits (reference cs4334). Notably, this chip requires an extra clock input compared to the other two due to it lacking an internal PLL (reference cs4334), 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 (reference analog devices article). 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 (reference MAX98357A). The MAX98357A also communicates using I2S protocol with a resolution range of 16 bits to 32 bits (reference MAX98357A). This DAC having a built-in audio amplifier before the output makes it popular for commercially available breakout boards used in many DIY projects (reference MAX98357A). 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 (reference ChatGPT). 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 (reference PCM5102A datasheet). The PCM5102A communicates using the I2S protocol and employs a resolution in the range of 16 bits to 32 bits (reference PCM5102A datasheet). 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 (reference PCM5102A datasheet).

Table 3.X: 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	Vin	Vin
Input Voltage (Vin)	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	15mA < x < 19mA	~2.4mA	7mA < x < 13mA
Power Consumption	75mW < x < 104mW	~12mW	21mW < x < 46mW
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 (Scott)

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 (reference ChatGPT).

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 (reference analog devices article).

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.X by icstation.com and Figure 3.X by rasiomuseum.org. Note that some of the components in Figure 3.X 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.X 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.

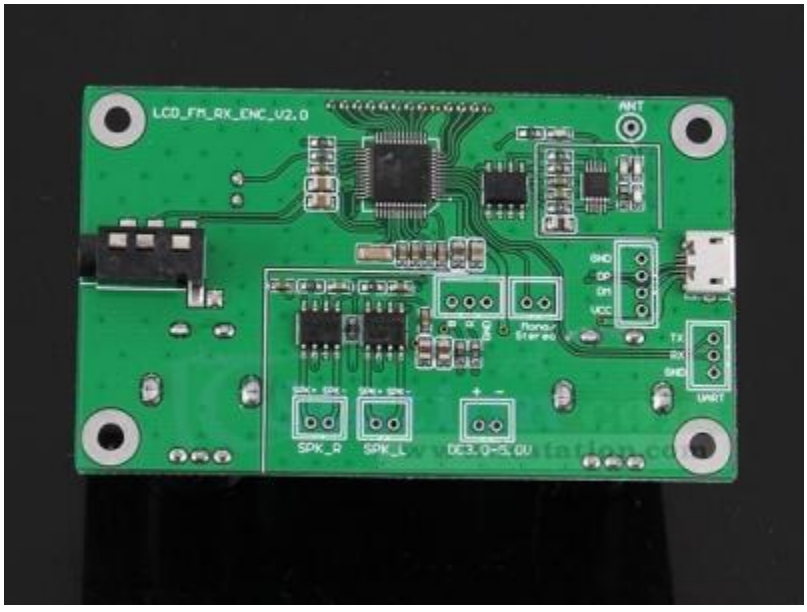


Figure 3.X: "PLL LCD Digital FM Stereo Radio Receiver Module 50Hz-18KHz Wireless Microphone Module DC 3-5V with LCD Display" (reference icstation)

Figure 3.X: "Sony Corporation; TFM-6100W" by radiomuseum.org (reference)

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.X: AM/FM Radio Options

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 (Scott)

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 (reference SI4735 datasheet). 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 30µW to 110mW in power, and outputs from 54mVrms to 67mVrms in AM mode; as well as 72mVrms to 90mVrms for FM mode (reference SI4735 datasheet).

The CXA1538 from Sony supports AM and FM bands of radio (reference CXA1538 datasheet). 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 (reference CXA1538 datasheet).

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

Table 3.X: 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	0.1mA < x < 21.5mA	3.5mA < x < 15mA	0.3mA < x < 16.8mA

Power Consumption	~30µW < x < 110mW	10.5mW < x < 90mW	810µW < x < 92mW

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 (Scott)

The radio for KIPS MkII 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 omni directional 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, which can be demonstrated with a few mathematical calculations as seen in (NAME FIGURES AND FORMULAS HERE). 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 (reference NXP article). 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 (reference NXP article). For vehicles, this can be seen as the car itself being the ground plane for the whip antenna (reference ARRL Antenna Compendium Vol. 1).

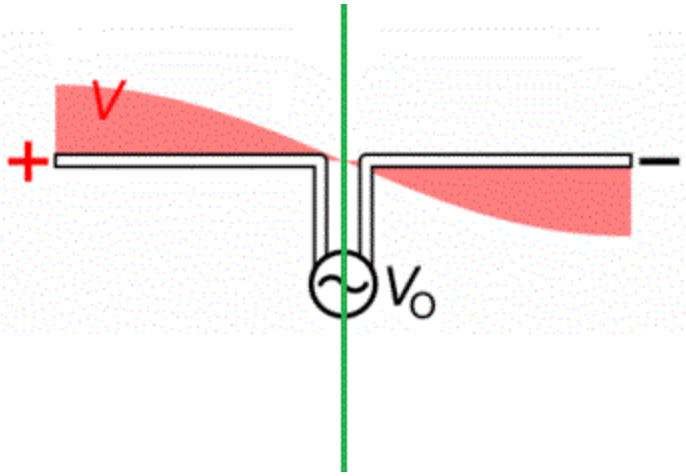


Figure 3.X: Dipole (Half Wavelength) Antenna Voltage Center by Andy aka on stackexchange (REFERENCE)

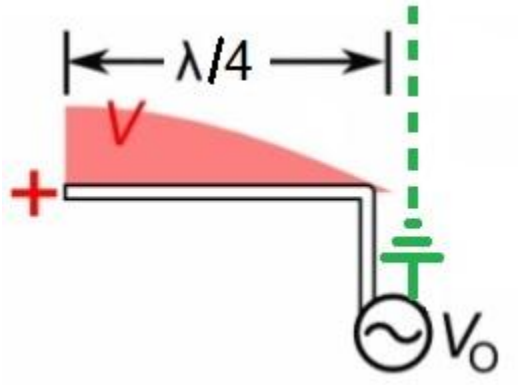


Figure 3.X: Whip (Quarter Wavelength) Antenna Voltage Center by Andy aka on stackexchange (REFERENCE)

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 (reference ARRL). This allows them to have excellent reception indoors (reference ARRL), 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 (reference ARRL), and their polarization depends on their orientation; either vertical or horizontal. A formula for a simple ferrite rod loop antenna can be found below (INSERT FORMULA AND REFERENCE IT HERE). (REFERENCE REFLECTION LOSS FOR LOOP HERE)

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 (reference digikey here). 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)

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 Antennas(Scott)

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 (reference amazon listing link). The antenna is a measly 13.5cm when collapsed, but when extended is around 51cm (reference amazon listing link).

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 (reference alibaba listing). This antenna comes in shorter though at 9.5cm collapsed, and 44cm extended (reference alibaba listing).

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 (reference ebay listing).

Table 3.X: 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	75Ω	75Ω	75Ω

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 (Scott)

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 is an incredibly simple option.

Table 3.X: Signal Summation Options

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 (reference AD4528-2-EP datasheet). 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 (reference AD4528-2-EP datasheet). 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 (reference AD4528-2-EP datasheet).

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 (reference ADTL082 datasheet). 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 ±3V to ±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 ±12V to ±13.5V (reference ADTL082 datasheet). 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 (reference TL082 datasheet). 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 (reference TL082 datasheet). 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 (reference TL082 datasheet).

Table 3.X: Op-Amp Comparison

	ADA4528-2-EP	ADTL082	TL082
Input Voltage (Vin)	Within Vsupp Range	Within Vsupp Range	Within Vsupp Range
Single Supply Voltage (Vsupp)	2.2V < x < 5.5V	6V < x < 36V	6V < x < 36V
Dual Supply Voltage (Vdsupp)	±1.1V < x < ±2.75V	±3V < x < ±18V	±3V < x < ±18V
Offset Voltage (w/o drift)	0µV < x < 4µV	0mV < x < 9mV	0mV < x < 20mV
Input Current	±10mA	±10mA	±10mA
Output Swing	Rail-to-Rail	±12V < x < ±13.5V	±12V < x < ±13.5V
CMRR	≥135 dB	≥80 dB	≥70dB
PSRR	≥130 dB	≥80 dB	≥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.

3.5.5.1) Volume Controls (Scott)

Table 3.X: Volume Control Options

3.5.5.2) Volume Potentiometers (Scott)

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 10kΩ, with a tolerance of ±20%, which means the actual value of the potentiometer can be anywhere from 8kΩ to 12kΩ (reference taiwan alpha datasheet), which is acceptable for our needs. It boasts a max power rating of 0.25W, and a mechanical life cycle of around 15k cycles (reference taiwan alpha datasheet), 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 (reference taiwan alpha datasheet); 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 (reference ChatGPT).

The slide potentiometer from Dongguan Tianqian Electronics is the most different of the three. It too boasts an overall resistance of 10kΩ with a tolerance of ±20%, but it begins to differ in its life cycle, power rating, and taper (reference Dongguan Tianqian Electronics datasheet). It boasts a 0.1W power rating, a mechanical lifecycle of 10k-15k cycles, and most importantly a linear taper (reference Dongguan Tianqian Electronics datasheet). 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 (reference ChatGPT).

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 10kΩ at a ±10% tolerance, giving it a 9kΩ to 11kΩ range of

resistance (reference CTS Electronics datasheet). 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 (reference CTS Electronics datasheet).

Table 3.X: Volume Potentiometer Comparison

	RV16A01F-41-15R1-A10K-30H4	C2031N-1A2-B10K-GP	450T328S103A1C1
Resistance	10kΩ	10kΩ	10kΩ
Taper	Logarithmic	Linear	Logarithmic
Tolerance	±20%	±20%	±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 (Scott)

Table 3.X: Amplification Options

Table 3.X: Output Types

3.5.6.2) Headphone Amplifier (Scott)

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.

Table 3.X: 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%

3.5.6.3) Audio Amplifier (Scott)

Table 3.X: Audio Amplifier Comparison

	PAM8301	PAM8302	LM386
Load Impedance	4Ω < x < 8Ω	4Ω < x < 8Ω	4Ω < x < 32Ω
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			
Output at 5V at 8Ω			
Output at 5V at 4Ω			

3.5.6.4) 3.5mm Headphone Jack (Scott)

Table 3.X: 3.5mm Headphone Jack Comparison

	SJ-43504-SMT-TR	ASJ-125-3	SJ1-352XN
Mount Type	Surface Mount	Through Hole	Through Hole
Mounting Angle	Right Angle	Right Angle	Right Angle

3.5.6.5) Speaker (Scott)

Table 3.X: Speaker Type Comparison

3.6) Signals and Ports

For this project, we are intending to have a video output port. This will allow the 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.

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.

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.

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.

*Figure 3.X: "*Versions and Bandwidth" (reference rtings)

Type	HDMI			DisplayPort		
Version (Alternative Names)	2.0 (2.0a, 2.0b)	2.1 (2.1 a, 2.1 b)	2.2	DP 1.2 (1.2a)	DP 1.4 (1.4a)	DP 2.1 (2.0, 2.1a)
Release Year	2013	2017	2025	2010	2016	2019
Max Bandwidth	18Gbps	48Gbps	96Gbps	21.6Gbps	32.4Gbps	80Gbps
Compression	No	Yes	Yes	No	Yes	Yes
HDR	Yes	Yes	Yes	Yes	Yes	Yes

Table 3.X: Output Type Comparison

	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)

3.6.1) HDMI (Harrison)

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[8]preference. [8] compare formfactors

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[9] 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 then 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 [9] , 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 be to broadcast information from the device onto another screen. This means that it is the source for all of the data going outward. To this 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.

Table 3.X: 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.

Reference [9]

3.6.2) USB 2.0 vs USB 3.0 (Harrison)

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 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 2.0 was introduced back in 2000 while USB 3.0 was introduced in 2008 [10]. 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 2.0s 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 whent 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. 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.

USB 2.0 was another feature that is accessible when using the SBC we've selected for this project. This component will be another feature for a potential customer to possess. As this 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. 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. Meaning to implement on the PCB it is best to choose USB 2.0 over 3.0. . [10] So after taking into account the tradeoffs between USB 3.0 and 2.0, USB 2.0 is the best choice to use for the KIPS MK II project.

Table 3.X: 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

3.6.3) Micro SD (Harrison)

Implementing an SD slot of any sort was a decision that took quite a bit of contemplation, as we realized that it not only required an adequate amount of power, but pins as well. Our choice in specifically using a MicroSD as an option was due to once again our strict regulation in maintaining a more compact approach. And though both regular and micro use about the same amount of power (~3.3V Operating Voltage), the MicroSD us[12]ne pin less [12]b[11]d modeling. [11]. Even though it does not have Write-Protected Switch on the Micro SD, the trade off is having less pins to fit within our guidelines. The switch would only take up physical space on the board that is not needed for this project and the benefits of using MicroSD out weight the disadvantages. On the topic of physical space it can be seen bellow that the size of the MicroSD cards is almost 10x in diameter smaller than the full size SD card. By choosing MicroSD all that space can be saved up and be used in another section. For all these reasons the MicroSD is the card of choice for memory on this project between SD and MicroSD.

Table 3.X: SD Card Slot Type Comparison

Form Factor	SD	MicroSD
# of Pins	9 Pins	8 Pins
Operating Voltage	3.3V	3.3V
Write-Protected S[13]h	Yes	No
Size[12]	32 x 24 x 2.1 mm	11 x 15 x 1 mm

3.7) Other

3.7.1) USB charge port (Power/Charging) (Harrison)

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 the KIPs. 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 [13]. 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 up to 240W, 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.

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.

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 the KIPS MK II.

Table 3.X: Charge Port Type Comparison

Form Factor	Micro USB	USB-C	Lightning
Power Output	Up to ~18W	Up to ~240W	Up to ~20W
Operating Voltage	5V	5V	5V
Data Transfer Speed	~480Mbps	~40Gbps	~480Mbps
Devices applicable	Very common	Less common	Apple proprietary
Reversible	No	Yes	Yes

3.7.2) Flashlight Types (Harrison)

For this project we are adding a flashlight into the design. This means that in order to add this flashlight, we have to consider the various types of flashlights out there. The types that are up for consideration are LEDs, Incandescent bulbs , and Laser diodes.

LEDs are currently the most used option when it comes to making flashlights due to having a great balance of efficiency, brightness, and durablity. They have a luminous efficiency of 60-110 lumens per watt. They also have a lifespan rate around 10000-50000 hours. This makes it a great option when it comes to conserving power. LEDs also can be used for a variety of tasks such as a flood light or long beam if this option is something to be considered. This option is to be considered favorably for these reasons.

An incandescent bulb is another option being considered. This option is the oldest one being considered. This is shown in how the operate at 10-20 lumens per watt. This means that it would take a large wattage to make a substanale amount of light. They also only average 1000-2500 hours. Finally a blub has an issue that the other options do not have which is they have a physical filament that has to be protected from vibrations or impacts. This means that it is less durable then the other options considered as well as it may be impractical for a device that is mobile.

The last option being considered is Laser diodes.Laser diodes, particularly when used in LEP (Laser-Excited Phosphor) flashlight designs, provide exceptional long-range performance but remain a niche option. Unlike LEDs, which spread light across a wider beam, laser diodes emit a narrow, highly collimated beam capable of reaching distances well over 1 km. This makes them ideal for searchlights, signaling, and specialty applications where beam throw is more important than total brightness. While their luminous efficacy can approach or exceed 150–170 lm/W in some designs, the practical lumen output of LEP flashlights usually ranges between 100–600 lumens, far less than what modern LEDs can provide. Their strength lies in candela (beam intensity) rather than raw lumen output. However, laser diodes are more fragile than LEDs, requiring precise thermal management and optical alignment.

Based on all of the data gained above, the best option for this project are LEDs. Not only do they have great lm/W at 60-110. They also have a great life span meaning they will last long and not need to be replaced on the KIPS MK II. They also have the benefit of not relaying on filaments or managing thermals to maintain the light they provide. Meaning they are the most durable and that matters in a device that will be mobile. Meaning that for this project LEDs will be used.

3.7.3) Flashlight (Harrison)

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[15] requirement [15][14] department. [14]. 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

The one chosen was the XP-L HI LEDs It is 3.45 x 3.45 mm.

3.8) PSU (Harrison)

Include a table to show the overall power requirement of the entire system (component, supply voltage, current)
discuss which type of power you want to use for each system; wall, battery, renewable, etc. and why
After you decide on sources then you talk about power converters; if you choose batteries you will need certain regulators, etc.

Needs its own regulator, have a charging IC using. zSwitching power to wall, Get a power brick Voltage charging IC 3A for each

For this section this is a general overview of every component that needs power.

Table 3.X: Table of components that need power

	3.3 V	5V	
Component			
Supply Voltage			
Current			
Applcations for circuit			

3.8.1.1) Battery composition(Harrison)

For this project the battery is one of if not the most important component to consider. This is due to how the battery is what the system will run off during the majority of its operational time. This is due to wanting the device be portable and not have it be stationed to a wall of the time. This is why the device will not be using an AC power adapter. So then it comes down to consider what type of batteries will be most suitable for this project.

The first battery type to consider is alkaline. These are simple batteries that can be bought at the store and they cost around 50 cents to one dollar and come in various shapes and sizes such as AA. They deliver around 1.5V output. The energy density of them is around 90-120Wh/kg. One feature is that alkaline batteries do not have is the ability to be rechargeable. Meaning that once

they are used up they can not be reused. If this power method is used as the battery then there will need to be a way to change out the batteries during use.

The second battery type up for consideration is the Lithium-Polyemer (LiPo) batteries. The batteries come in various styles such as a pouch style or a block like case. These are more expensive in comparison to the Alkaline batteries. LiPo batteries can output around 3.7 V. Their energy density is at 150-250Wh/kg. They are light weight and rechargeable. Meaning that if this style of battery is chosen then the battery can be recharge with an outside source.

The third type of battery composition is Nickle Cadmium. These batteries can out put around 1.2V. Their energy density is at 40-60 Wh/kg, This coosindes with these batteries being quite heavy a AA style Nickle cadmium battery is 24g compared to the 20g of alkaline. This makes it quite a heavy option and that bust considered as this project is mobile. Nickel cadmium is also rechargeable.

After considering all of the different options, the best one for the battery will be the Lithium-Polyemer batteries. This is due to its greater power output at around 3.6V. Not only that but is has the greatest energy density of the options 90-120Wh/kg. This means that it will have the greatest energy capacity relative to its weight, this is important to minimize due to this project being mobile. On that topic the last reason that this type of battery was chosen was due to it being rechargeable. Meaning that the batteries in the device can be recharge through a port rather than having to change out the entire battery cells. All these factors make it the best choice for the KIPS MK II

Figure 3.X: "Lithium vs Alkaline Battery Comparison Chart"(refrence redway power)

Feature	Rechargeable Lithium Batteries	Alkaline Batteries
Normal Voltage	3.6-3.7 (Li-ion) or 1.5V (Li-Primary)	1.5V
Rechargeability	Yes	No
Cycle life	500-2,000+ cycles	Disposable (single use)
Weight	~30% lighter than alkaline	Heavier
Cost per cell??	Higher upfront cost	Lower upfront cost
Operating Temperature	-20℃ to 60℃	-18℃ to 55℃

Safety Features	Battery Management Systems (BMS)	Limited

Table 3.X: Battery Composition Type Comparison

	Lithium-polymer (LiPo)	Akaline	Nickel Cadmium
Recharable	Yes	No	Yes
Weight	Light	Medium	Heavy
Energy density	150-250Wh/kg	90-120Wh/kg	40-60 Wh/kg,
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, industrial

3.8.1.2) Battery (Harrison)

On the matter of power comes the main component of the battery. This is very important as this is what the main form of power will be for the device when it is not plugged in and therefore must sustain all of the components. It will be able 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. What this means is the correct configuration has to be chosen for this project. The three options taht are being looked at for this project is 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 perviously.

The reason for 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 this 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 then it 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 those being two 3.6V in series. Since these designs are in series then the output voltage will be at 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 as well due to having more cells. The downside is that these styles of designs need to maintain the same voltage on the cells to achieve max voltage . Also that a second cell will increase size, weight,cost and charging complexity. Not only that but consider if the voltage needed is less than the 7.2 V required, this will then need a step down regulator to work in the project. Not only that but 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-circuiting protection, and High-temperature protection. This extra component is for using the LiPo composition that was decided above.

After consideration the best implementation for the KIPS MK II is two 3.6V at 8000mAh. This is for several reasons. The in series configuration allows for greater voltage output for the system to use. Not only this but it is the greatest for safety as to avoid the cells fighting over which one is dominant for the voltages. 8000 mAh was chosen for the greatest overhead when it comes to storage as the difference between 6000 mAh and 8000 mAh is capacity not topology. To implement this style of battery a step down converter may be needed but that will be discussed later. Not only that but also using the protection circuit for the LiPo batteries.

Figure 3.X: "Li-ion protection IC"(refrence Ablic)

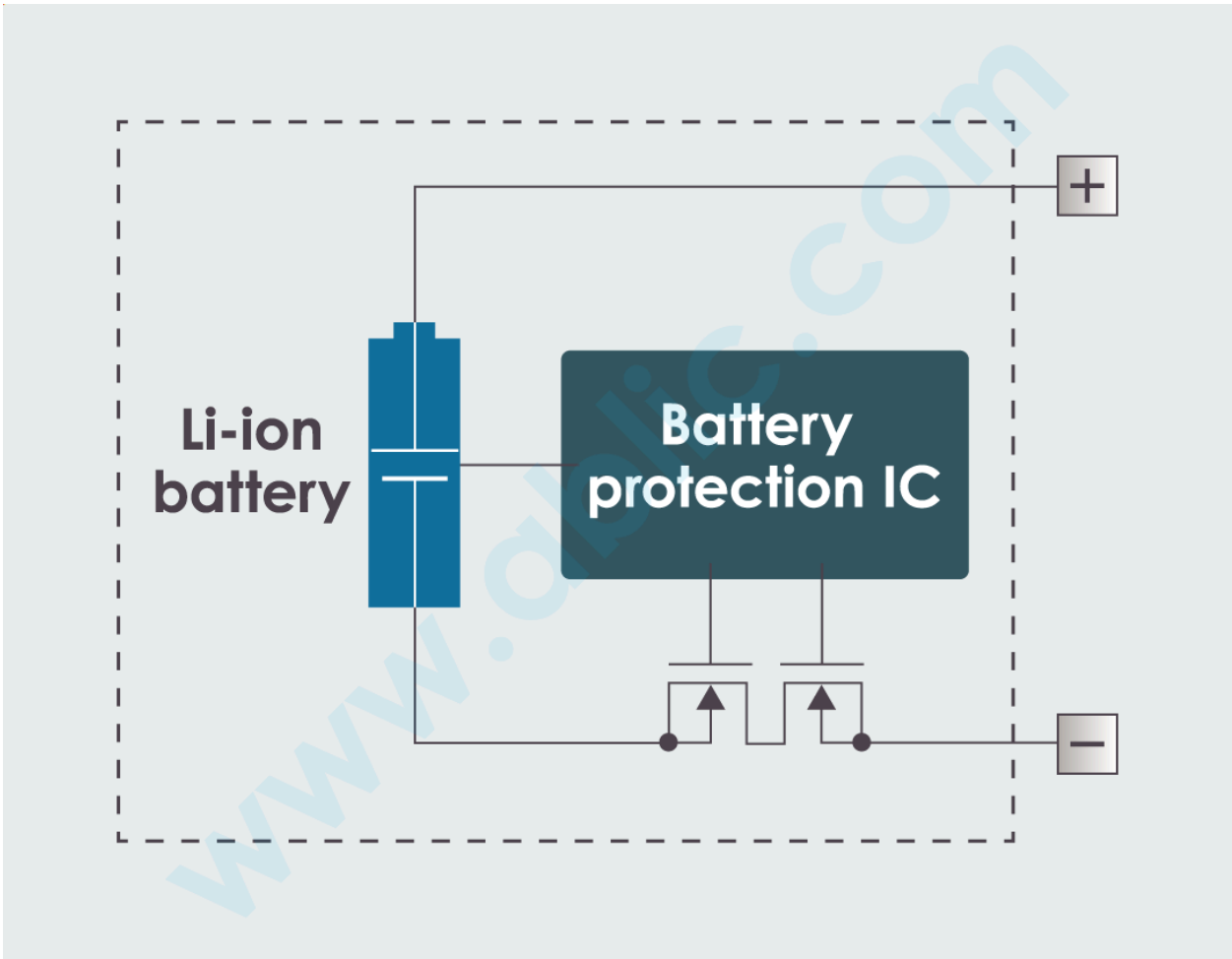


Table 3.X: Battery Configutation Type Comparison

	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 cell	6000 mAh	8000mAH

3.8.3) Battery Power Gauge IC (Jonathan)

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.

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 mutli-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.

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+.

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.

Table 3.X: Battery Power Gauge IC Comparison

	Analog Devices DS2781G+	Analog Devices DS2786BG+	Texas Instruments BQ34Z100-G1
Interface	1-wire serial communication	I2C	I2C / 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

3.8.4.1) Linear vs Switching Regulator (Harrison)

For this project to consider a regulator, there needs to be a discussion between the two type of regulators. Those being Linear vs Switching. A linear regulator is 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 which 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 cheap solution, but linear regulators are normally inefficient because the difference between the input voltage and regulated output voltage is continually dissipated as heat. The thing of note is due to this style of design the only topology supported with this is Buck Regulator.

The second type of regulator is switching regulators. These type of regulators were created due to the need for wider Vin 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 switched off so it dissipates almost no power. Switching regulators are able to generate output voltages that are higher than the input voltage or of opposite polarity, unlike linear regulators. The versatility of these converters allow configuration for buck, boost, buck-boost, flyback, inverting in isolated and non-isolated applications.

After considering both options for the regulators the best one for use is the switching regulators. This is for the following reasons. High efficiency, this is so the is minimal power being lost from the regulator to ensure all components that need power get that required amount. Not only that but switching regulators have a high input voltage range, this is best for using a battery for the voltage source that will discharge over time. The battery output over time will change and therefore need a greater range for input voltage then the stricter requirements of Linear regulators. This also allows for a greater number of topologies that the project might require. Therefore the best style of regulators used for the KIPS MK II is the switching regulator.

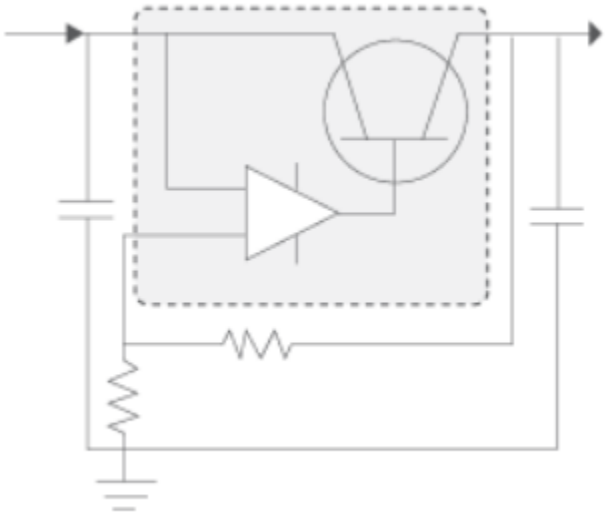


*Figure 3.X: "*Linear regulator" (reference renesas)

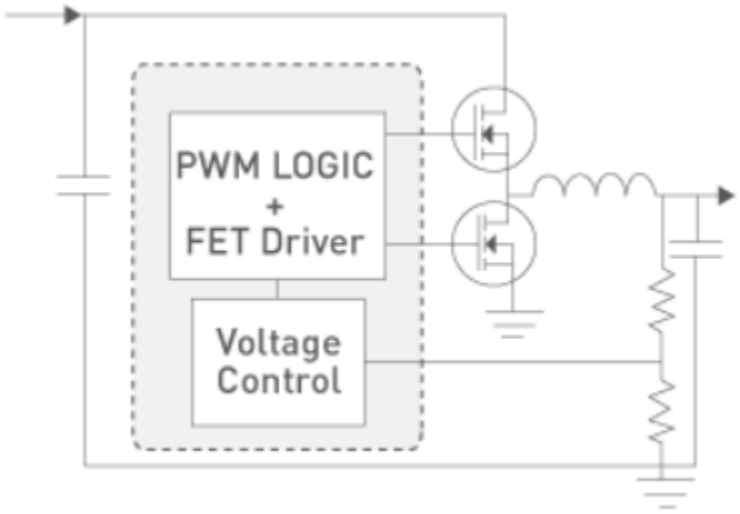
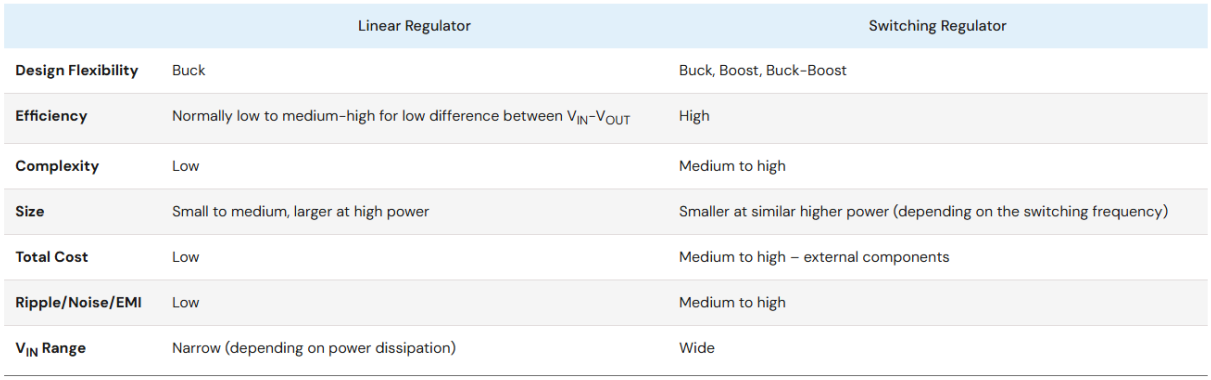


*Figure 3.X: "*Switching regulator" (reference renesas)

Table 3.X: Linear vs switching regulator comparison

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



	Linear Regulator	Switching Regulator
Design Flexibility	Buck	Buck, Boost, Buck-Boost
Efficiency	Normally low to medium-high for low difference between V_{IN}-V_{OUT}	High
Complexity	Low	Medium to high
Size	Small to medium, larger at high power	Smaller at similar higher power (depending on the switching frequency)
Total Cost	Low	Medium to high – external components
Ripple/Noise/EMI	Low	Medium to high
V_{IN} Range	Narrow (depending on power dissipation)	Wide

*Figure 3.X: "*Linear vs switching regulator" (reference renesas)

3.8.4.2) Buck vs Boost vs Buck Boost (Harrison)

For the KIPS MK II one of the things that have 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 have their 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 also is that this raises the current as the ratio must be maintained. This topology has high efficiency 92+ and has low ripple

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.

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.

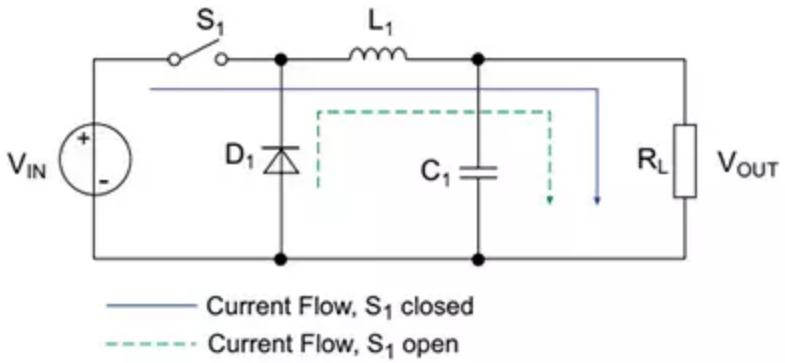


Figure 3.X: ""Buck converter" (reference recom-power)

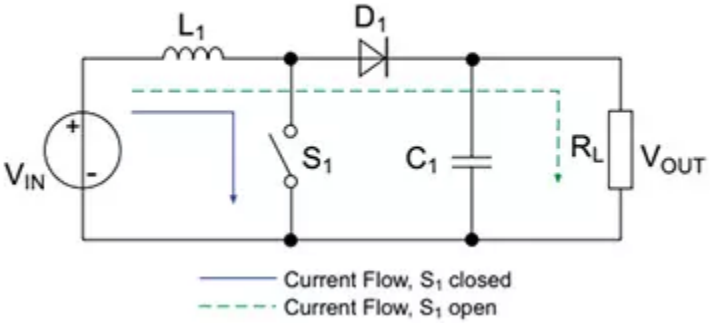


*Figure 3.X: "*Boost converter" (reference recom-power)

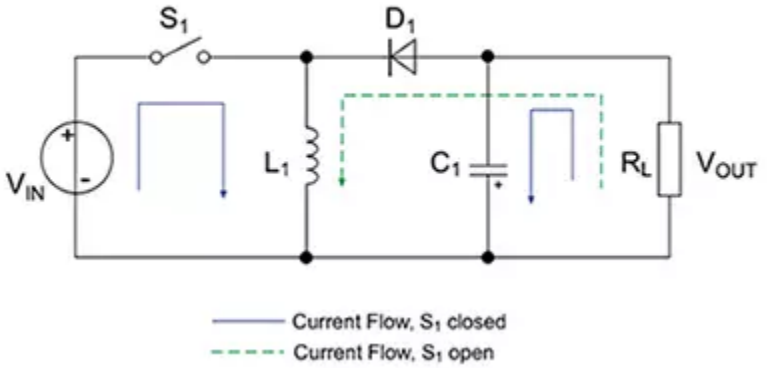


*Figure 3.X: "*Buck-Boost converter" (reference recom-power)

Table 3.X: Converter type Comparison

Type	Buck	Boost	Buck-Boost
Function	Reduces voltages	Increase voltage	Either increase or decrease voltage
Output	Vout < Vin	Vout > Vin	Vout < Vin or > Vin
Polarity	Same as input	Same as input	Inverting (basic) or Non-inverting (advanced)
Efficiency	High 92+%	High 90+%	Moderate 80+%

3.8.4.3) Buck Regulator (Harrison)

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. 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 LTC3633 from Analog Devices. The LTC3633 dual- channel monolithic synchronous buck regulator [15] which has the feature to offer two different channels of

different voltages on the output, 180 degrees out of sync. What this allows is two different power lines to come from the same source. This offers the same advantages of using a normal buck regulator, which include voltage regulation and efficiency, while also reducing input ripple current. One channel will run at 5V for the Raspberry Pi, and the 3.3V line will go into the MCU and the audio lines. This will allow the device to use the two batteries in series to the fullest advantage by combining the two input voltages and stepping them down to the needed voltages while also providing two lines of output voltages to use for different purposes. The LTC3633 takes the \~8.4 volts from the series batteries and steps them down into the more usable 5V and 3.3V. Not only this, but the LTC3633 has an input range between 3.6 and 15V, so the battery does not have to maintain just one constant voltage, which would not be suitable for the purpose we need from it.

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[16]p-down regulator [15]so it also has two different output channels that can be used. The main issue with this choice comes in the input voltage range. 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 is two synchronous buck r[17]ator controllers [16]This is an important distinction from the rest of the components shown. The LTC3633 has MOSFETs built in, while the LM5642 does not. Meaning, 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.

Comparing the LTC3633 between the LTC3615 and the LM5642, the choice becomes clear for the LTC3633 to be the buck regulator of choice for this project. It has the highest efficiency for this project. At 95%, this will allow the device to run at the lowest wasted energy and allow for almost all of the voltage from the batteries to go to either the 5V or 3.3V line.

Brand	LTC3633	LTC3615	LM5642
Brand	Analog Devices	Analog Devices	Texas Instruments
Input Voltage Range	3.6V-15V	2.25V-5V	4.5-36V
Reference voltage	0.6V(ref)	Vin	Vin
Output voltage	0.6V-12.5V	0.6V-~5V	2.9-38V
Max Current	3A	3A	3A
Price	5.07	4.72	5.63
Efficiency	95%	94%	90%
Switching Frequency	500kHz to 4MHz (External resistor)	200kHz to 4MHz (External resistor)	200 kHz (Fixed) for LM5642 375 kHz (Fixed) for LM5642x

3.9) Communication Protocols

Ex i2c, uart, etc?

3.10) Development Environment: Languages and Repositories

** discuss the need for programming languages and platforms to host them; repos

3.10.1) 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.

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 autoinstaller, 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.

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.

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.

3.10.2) 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.

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 in because of its complexity and lower-level syntax, meaning that more time may be required to program the required software for the device.

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.

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.

Table 3.X: SD Card Slot Type 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

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.

3.10.3) Repositories

[Preface]

[**GitHub**]*
GitHub is a free Git-based version-control software

[**GitLab**]
[**Bitbucket**]
[**Gitea**]

3.10.4) Integrated Development Environment (IDE):

- Type
- ESP32 support?
- Ease of use
- Code assistance
- Debugging

- Extensibility
- Git integration?
- Build/Flash support

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. 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.

[Notepad++]

Notepad++ is a free, extremely lightweight text editor based in Microsoft Windows. Notepad++ has no debugging features and very limited plugins.

[Visual Studio Code]

Visual Studio Code (VS Code) is a full-feature IDE released by Microsoft. VS Code supports several different programming languages including Python, C++, etc. and can support even more through different imports. One edge VS Code has over its competition is that it has built-in support for git-based version control _____s. VS Code also has a fairly extensive debugging _____ and natively supports its own AI coding assistant to help fix syntax issues, generate complex lines of code and ____.

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 _____.

> PlatformIO is an open-source extension in VS Code for embedded development. PlatformIO enables building, compiling, and uploading firmware to microcontrollers like the ESP32. 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 setup 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. 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.m 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.

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 _____. This approach would pair together the benefits of each such as the _____ of ESP-IDF and _____ of PlatformIO and could potentially solve some of the shortcomings of both approaches as well.

	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++

VS Code is highly versatile.

- Type
- ESP32 support?
- Ease of use
- Code assistance
- Debugging
- Extensibility

- Git integration?
- Build/Flash support

	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 abd ESP-IDF natively	Needs external command line and other tools

because it is compatible with the ESP32
Notepad++ because why tf not lol

3.10.5) Other Software Packages

literally have no clue if we even need this section or what to put in it…

***Communication Protocol Chapter

3.10) Software

Naturally, software comes next

-embedded programming

- talk about languages (C++, python)
- platform; IDE
- other software packages (for web or mobile ui)

- MUST BE EMBEDDED PROGRAMMING (minimum)

4) Standards and Design Constraints

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.

Prompt 1:

APPENDIX A: REFERENCES

Ask for permission for citing sources.

- If the author responds and says yes, use the source
- If the author refuses, do not use the source.
- If the author does not respond, do not use.
- Use source at your own risk, if you choose to use it if no response is given.

APPENDIX B: COPYRIGHT PERMISSION