

GNG2101
Design Project User and Product Manual

IR to RF Converter

Group E3.1

Linden Sheehy, 300287097

Skyler Thomas, 300303714

Luca Balan, 300161017

Nakul Parekh, 300129778

Milo Murillo, 300381208

Kayhan Krebs, 300294723

March 27th, 2024

University of Ottawa

Table of Contents

Table of Contents ii

List of Figures iv

List of Tables v

1 Introduction 1

2 Overview 2

 2.1 Cautions & Warnings 2

3 Getting started 3

 3.1 Configuration Considerations 3

 3.2 User Access Considerations 3

 3.3 Accessing/setting up the System 3

 3.4 System Organization & Navigation 7

 3.5 Exiting the System 9

4 Using the System 10

 4.1 Receive and Decode Command from IR Remote 10

5 Troubleshooting & Support 11

 5.1 Error Messages or Behaviors 11

 5.2 Maintenance 11

 5.3 Support 11

6 Product Documentation 12

 6.1 Electrical Components 12

6.1.1	BOM (Bill of Materials)	12
6.1.2	Equipment list	12
6.1.3	Instructions.....	12
6.2	Software and Code	13
6.2.1	BOM (Bill of Materials)	13
6.2.2	Equipment list	13
6.2.3	Instructions.....	13
6.3	3D Printed Case.....	13
6.3.1	BOM (Bill of Materials)	13
6.3.2	Equipment List.....	13
6.3.3	Instructions.....	14
6.4	Testing & Validation.....	14
6.4.1	Response Time.....	14
6.4.2	Connection Distance	14
6.4.3	Connection Angle	14
7	Conclusions and Recommendations for Future Work	16
8	Bibliography	17
	APPENDICES	18
9	APPENDIX I: Design Files	18
9.1	MakerRepo.....	18

List of Figures

FIGURE 1: HIGHLIGHTED INSTRUCTIONS FOR THE INSTALLING THE <code>DIYABLES_IRCONTROLLER</code> LIBRARY.....	4
FIGURE 2: POP-UP WINDOW FOR <code>DIYABLES_IRCONTROLLER</code> DEPENDENCY	4
FIGURE 3: POP-UP WINDOW WITH BOARDS AND PORT SELECTION	5
FIGURE 4: BUTTON USED TO OPEN THE SERIAL MONITOR WINDOW	6
FIGURE 5: IR SENSOR	7
FIGURE 6: WIRING ADAPTER	7
FIGURE 7: CONNECTIONS BETWEEN THE WIRING ADAPTER (LEFT) AND ESP32 (RIGHT)	8
FIGURE 8: LOCATION OF THE MICRO-USB PORT ON THE ESP32	9
FIGURE 9: RUNTIME BEHAVIOR OF THE SYSTEM.....	10

List of Tables

TABLE I. ACRONYMS	VI
TABLE II. GLOSSARY	VI
TABLE III. CONTACT INFORMATION.....	11
TABLE IV. BOM ELECTRICAL COMPONENTS	12
TABLE V. BOM SOFTWARE AND CODE	13
TABLE VI. BOM 3D PRINTED CASE	13
TABLE VII. REFERENCED DOCUMENTS.....	18

Table I. Acronyms

ACRONYM	DEFINITION
UPM	User and Product Manual, the piece you are currently reading.
IR	Infrared signals.
RF	Radiofrequency signals, such as Bluetooth and WiFi.
IDE	Integrated development software, used to access and interact with microcontrollers via a computer.

Table II. Glossary

TERM	ACRONYM	DEFINITION
Tobii Dynavox 1-13 EyeGaze	Eyegaze	The device that our client uses with their students to allow them to communicate and interact in school.
DIYbles_IRcontroller library	IR library	IR signal database, used to code and decode the signals used by our IR remote.
ESP32 Development Board	ESP32	Microcontroller with WiFi and Bluetooth connection.
Serial Monitor		A tab withing the Arduino IDE.
MakerRepo		A free-use website that allows the compilation and storing of project files online, for others to easily access.

1 Introduction

This User and Product Manual (UPM) provides the information necessary for the continued development of our IR to RF signal transformer, made as part of our course GNG 2101.

It will begin with a description of the device and the role it seeks to fill, followed by detailed instructions on how to operate it, ending with the technical specifications and all information we believe to be necessary for the continued development of this device.

This device is still under development, so this document is primarily intended for those who wish to continue our work, primarily future students of GNG 2101. It should provide enough information for actual use of the device, although its present state does not offer much.

2 Overview

We were tasked with creating a small, portable, and durable device that receives IR and transmits RF to robots for students at John McInnis Centre.

Our client is Les Smith, who is an occupational therapist representing John McInnis Centre in Prince, George, British Columbia. He works with numerous young students that face challenges to their mobility and use eye-gaze devices to enable better participation in their classrooms. The devices they use have access only to IR transmission, leaving them unable to interact with devices that operate on RF frequencies, as many of their in-class devices do.

Our product is a proof of concept that demonstrates that we can receive IR and convert their information to RF successfully. It is by no means ready to use out of the box with the eyegazes used by Mr. Smith, as without access to the device or its IR library, we cannot fully tailor our device for compatibility with it.

Our product is composed of a microcontroller with an IR receiver connected to it, contained within a small plastic case. To function it must be connected to a computer for its power source, which is done through a USB-A to micro-USB-B type cord, inserted into the micro-USB port in the device's side. It can receive and decode IR from an IR remote from a maximum distance of 2.5 meters, and display these received messages in text form within an Arduino IDE.

2.1 Cautions & Warnings

Avoid:

- Exposure to water; our device is NOT water-resistant
- Exposure to high temperatures
- Prolonged exposure to heat sources
- Strong impacts

Our device is not made for high intensity environments and should not be expected to fare well in such roles. The device's IR receiver also has somewhat limited range, and so should be pointed in the direction of the transmitting device.

3 Getting started

3.1 Configuration Considerations

The physical device is composed of three main components:

- ESP32
- IR receiver build,
- Casing

The device itself must be connected to a computer via the micro-USB port located on the ESP32, and the circuitry can function independent of the casing. For access to the ESP32 code, an IDE is required.

3.2 User Access Considerations

Given that our prototype is not a finished product, it is primarily for product development and testing. Entry-level familiarity with C++ coding language is recommended, as is familiarity with circuitry.

3.3 Accessing/setting up the System

3.3.1 Install an IDE

You will require an IDE to access the ESP32 and its code. This manual will assume the use of Arduino IDE, which can be downloaded from the following link:

<https://www.arduino.cc/en/software>

3.3.2 Install Dependencies

For the IR remote, a third-party library is required and is accessed through Arduino IDE:

- In the ‘Library Manager’ tab, search for ‘DIYbles_IRcontroller’.



Figure 1: Highlighted instructions for the installing the DIYables_IRcontroller library

- Click 'Install', and the 'Install All' if prompted by a pop-up window.

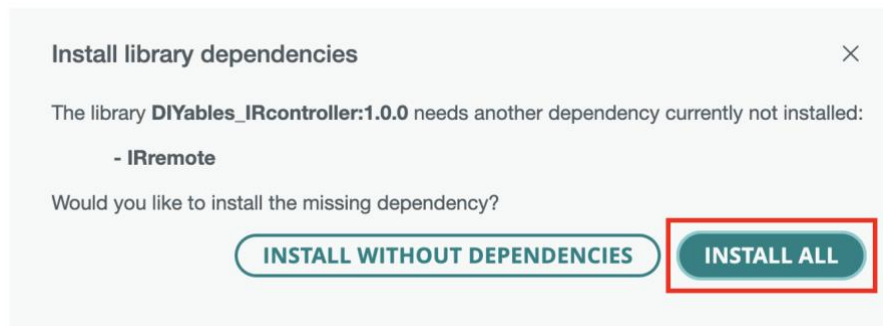


Figure 2: Pop-up window for DIYables_IRcontroller dependency

3.3.3 Configurations for Arduino IDE

To connect and configure your IDE to interact with your ESP32:

- Ensure the prototype is connected to your computer using a micro-USB to USB-A cable.
- Click on the drop-down menu and click on 'Select other board and port...?'.

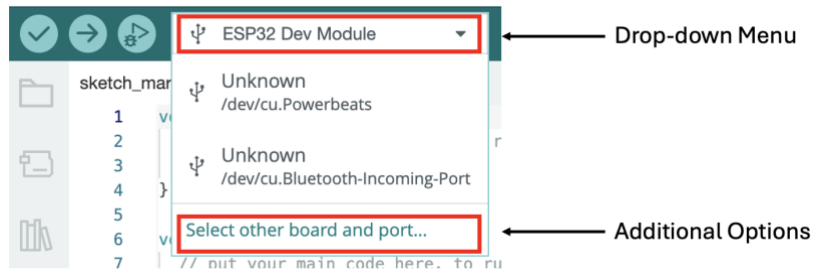


Figure 3.3.3.1: Drop-down menu where to find the additional options menu.

- Select 'ESP32 Dev Module' from the board's menu.
- Select the port that corresponds to the plugged-in prototype on your computer from the port's menu.

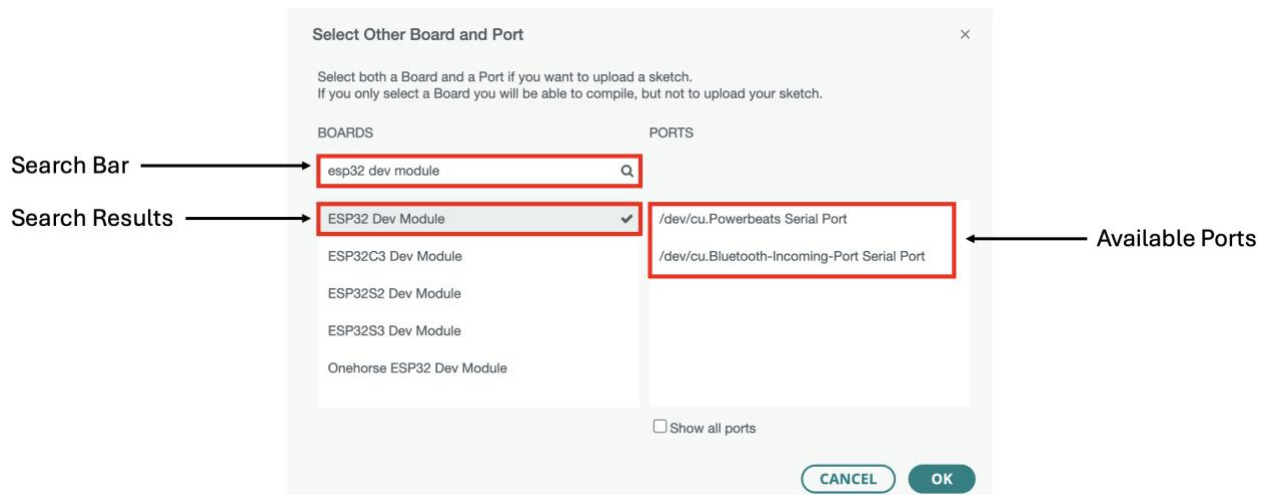


Figure 3: Pop-up window with boards and port selection

3.3.4 Upload and Run the Code

- Copy the code provided in our MakerRepo into the editor or open the provided prototype.ino file.
- Once you have the code, click on 'Verify'.

- Click on ‘Upload’.

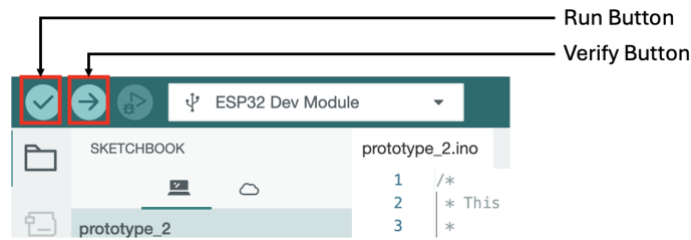


Figure 3.3.4: Verify and Run buttons.

- If the ‘Serial Monitor’ window is not open, click on the top right corner magnifying glass icon to open it.



Figure 4: Button used to open the serial monitor window

3.3.5 Getting Results

- Point the IR remote towards the device, ensuring the receiver is visible.
- Press on any of the 21 buttons on the remote.

Notice the relevant commands show up on the ‘Serial Monitor’ of Arduino IDE. For example, if the button ‘8’ is pressed on the remote, the string of text ‘8’ should appear on the ‘Serial Monitor’.

3.4 System Organization & Navigation

The circuitry of the prototype is composed of three functional components: the ESP32, the IR sensor and the wiring adapter. We will go over the connections between these components.

3.4.1 IR Sensor Connection to Wiring Adapter Connection

The IR Sensor is connected to the wiring adapter via three pins:

- VCC
- Ground
- Data Input

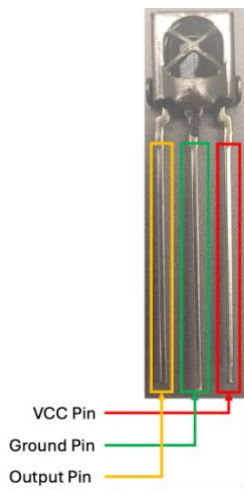


Figure 5: IR sensor

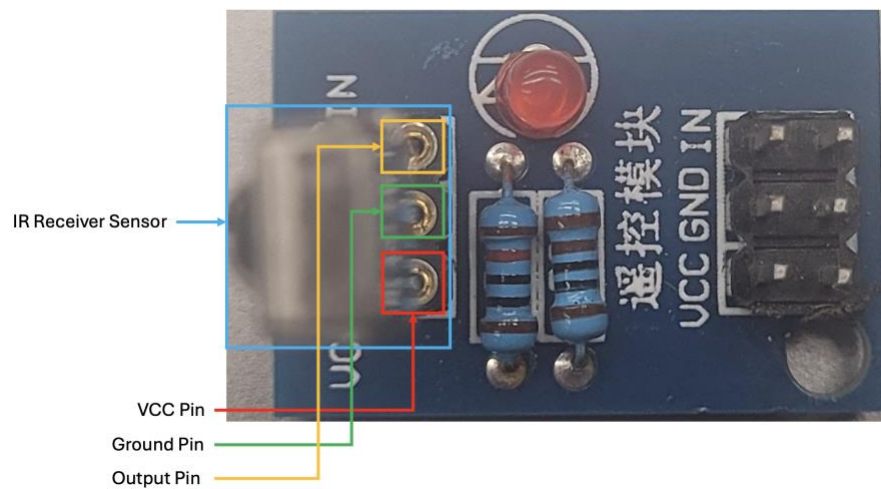


Figure 6: Wiring Adapter

3.4.2 Wiring Adapter to ESP32 Connection

The Wiring Adapter is connected to the ESP32 via three pins:

- VCC to 3V3
- Ground to ground
- Data input to data output

These connections are not soldered in our prototype, using female-to-female jumper wires instead. In a future prototype, it would be beneficial to have a custom PCB design to be able to solder connections, which provides increased reliability and longevity.

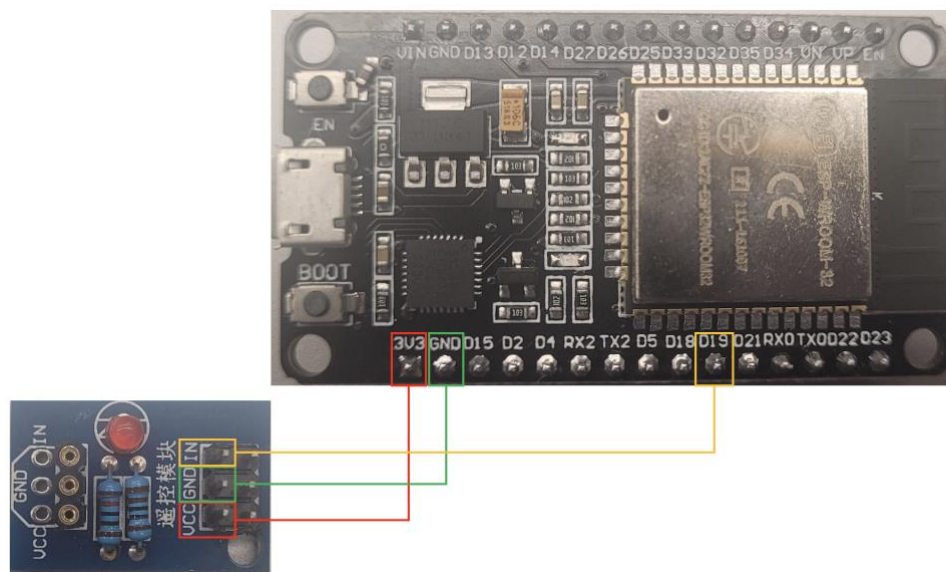


Figure 7: Connections between the Wiring Adapter (left) and ESP32 (right)

3.4.3 Computer to ESP32 Connection

The ESP32 is simply connected to the computer via the micro-USB port on the ESP32 and a normal USB port on the computer.

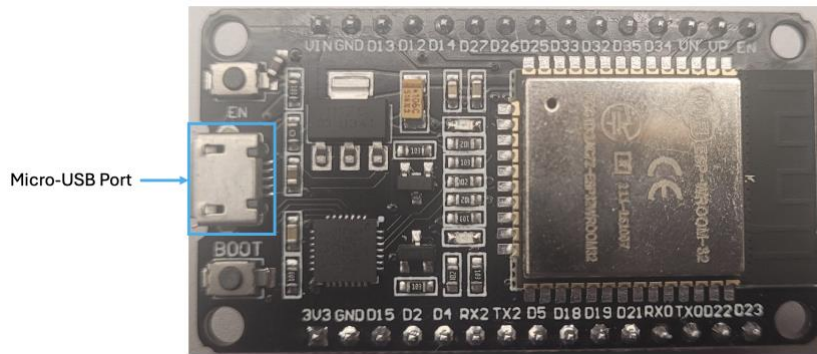


Figure 8: Location of the micro-USB port on the ESP32

3.5 Exiting the System

There is no power button for our device. Simply unplug the prototype from the computer and stow away safely. For Arduino IDE, save your work files and then close the program.

4 Using the System

The following sub-sections provide detailed, step-by-step instructions on how to use the various functions of our IR to RF transformer.

4.1 Receive and Decode Command from IR Remote

The prototype serves not as a completed product, but as proof of concept. We are looking to demonstrate that our prototype can receive, decode and print the IR messages from the IR remote. Below is a diagram that summarizes the runtime behavior of the system.

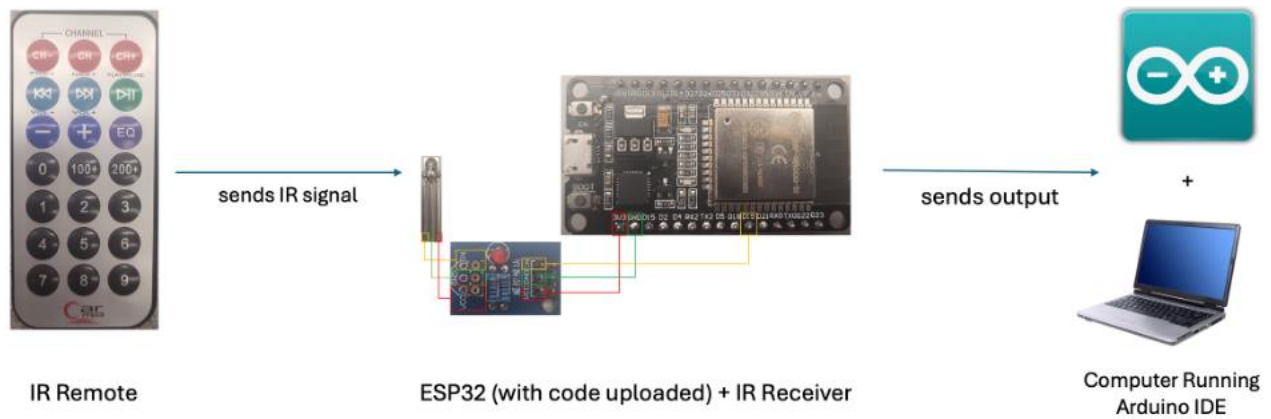


Figure 9: Runtime Behavior of the System

The microcontroller serves as the device's core, as it decodes incoming IR and transmits the translated messages directly through its built-in RF emitter.

5 Troubleshooting & Support

5.1 Error Messages or Behaviors

The main errors stem from issues related to the use of Arduino IDE. General system errors may include:

- Empty Serial Monitor - After running the code and giving IR input, serial monitor remains empty:
 - Wait 30 seconds and try clicking buttons on the IR remote again.
 - Close and reopen the serial monitor.
 - Re-run the code.
 - Disconnect and reconnect the prototype.
 - Close and reopen Arduino IDE application entirely.
- Casing Fit: Casing cover refuses to reattach:
 -

5.2 Maintenance

If the ESP32 is ever removed from its case for whatever purpose, please store it in a static-shielding bag and use a protective foam for its pins. In addition, the 3D printed case that holds the components is not airtight. For this reason, there could be dust residue built up from time to time. Please take the time to remove the components from the case from time to time and clear out the dust from the case if there is any.

5.3 Support

If there are any problems or questions, you can contact one of the team members. See the list of names and contact information below.

Table III. Contact Information

Student name	Student email
Linden Sheehy	lshee103@uottawa.ca
Skyler Thomas	sthom237@uottawa.ca
Luca Balan	lbala022@uottawa.ca
Nakul Parekh	npare083@uottawa.ca
Milo Murillo	cmuri013@uottawa.ca
Kayhan Krebs	kkreb028@uottawa.ca

6 Product Documentation

6.1 Electrical Components

6.1.1 BOM (Bill of Materials)

Table IV. BOM Electrical Components

Part	Price	Link
ESP32	\$12.00+	Espressif.com
Micro-USB cord	\$10.00+	Amazon.com
FtF Jumper Wires	\$6.99+	Amazon.com
IR Sensor + Wiring Adapter	\$13.99	Amazon.com

6.1.2 Equipment list

- ESP32
- Micro-USB cord
- 3 Female-to-Female jumper wires
- Wiring adapter
- IR sensor

6.1.3 Instructions

- Using the female-to-female jumper wires, make the following three connections:
 - Connect the 3V3 pin on the ESP32 to the VCC pin on the wiring adapter.
 - Connect the GND pin on the ESP32 to the GNG pin on the wiring adapter.
 - Connect the D15 pin on the ESP32 to the IN pin on the wiring adapter.
- Plug in the micro-USB cord into the micro-USB port of the ESP32. Once attached to the ESP, connect the cord to your desired power source, at which point a red light on the ESP will indicate a working power supply.

6.2 Software and Code

6.2.1 BOM (Bill of Materials)

Table V. BOM Software and Code

Part	Price	Link
Arduino IDE	\$0.00	Arduino Software
“DIYables_IRcontroller” Library	\$0.00	DIYables Github
Our Code	Blood, sweat, and tears	IR to RF Converter

6.2.2 Equipment list

- Arduino IDE
- “DIYables_IRcontroller” Library
- Our Code

6.2.3 Instructions

The instructions for this part can be found in section 3.3 - Accessing/setting up the System.

6.3 3D Printed Case

6.3.1 BOM (Bill of Materials)

Table VI. BOM 3D Printed Case

Part	Price	Link
PLA plastic	\$24.99+	MatterHackers.com
CAD Software	\$0.00+	Free CAD Software

6.3.2 Equipment List

- 3D Printer

- PLA Plastic
- Computer Assisted Design (CAD) Software

6.3.3 Instructions

- Download the blueprint file for the design of the case from MakerRepo.
- Print the case

6.4 Testing & Validation

We have tested for the different target specifications that we set out. More specifically, we are testing for response time, connection distance, and connection angle.

6.4.1 Response Time

The device can receive, decode and print to the screen the command received from the IR remote in real time. Meaning, from the moment the user points the remote at the device and presses a button to the moment the string that corresponds to that button pops up on the screen, there is no noticeable delay observable to the human eye.

6.4.2 Connection Distance

The device can receive the commands from the IR remote from up to 2.5 meters approximately. This is compatible with a real-life scenario where our device is actually tailored to be used with an eye gaze device. In fact, the device would be plugged into the eye gaze device so the distance between the eye gaze device emitting the IR signals and our IR-to-RF converter device would be less than 2.5 meters.

6.4.3 Connection Angle

The device works better the closer the incoming signal is to being perpendicular to the receiver. The prototype still works in the case where the IR receiver is at an angle of less than 90 degrees, but we notice that as we surpass 90 degrees, the device started to encounter interference and struggled to receive all signals. In a real-life scenario, we would have to study the eye gaze

device used and find an ideal placement position for our device, to assure that we are the closest to an ideal angle between the IR emitter and our device.

7 Conclusions and Recommendations for Future Work

More proper documentation of our processes and progress would likely be the greatest change we could have made, as our development started out quite unorganized. A greater focus on this from the start would serve any future groups tackling this issue well.

The greatest area of difficulty of development came with the use of the ESP32's RF capabilities, which we were never able to fully unlock. A better understanding of how the system works and more time to properly integrate its use into your code would have served us greatly.

Another area of issue was the IDE, and its interaction with our micro-controller. It proved to be very temperamental, so testing the use of different IDEs aside from Arduino's might save time that would otherwise be used dealing with issues arising from that program.

Two hardware choices that were scrapped, were the use of different power sources and the use of multiple IR receivers.

The receivers have a somewhat limited cone of effect, so to address this we considered using multiple in an array, to cover a greater range around the device. Ultimately, we chose not to for size and cost considerations, but a larger device could more reasonably accommodate such a design choice.

For power sources, we considered the use of internal batteries and of different cord connections, but settled on micro-USB B as the ESP32 already contains the needed port. Were a microcontroller without such a port to be used, options such as removable coin or rechargeable lithium-ion batteries, or different cords such as lightning USB or even 3.5 mm audio jack.

The group adjacent to ours chose a software solution to our problem as opposed to our hardware one, choosing to use increased specificity to our situation to streamline their work, at the cost of potential applications of their solution. A hybridization of the two methods is a potential avenue of future development that could hold great potential.

We wish the best of luck to anyone who might chose to follow in our steps and take on the requirement we were tasked to help fill.

8 Bibliography

APPENDICES

9 APPENDIX I: Design Files

9.1 MakerRepo

[IR to RF Converter](#)

Table VII. Referenced Documents

Document Name	Document Location and/or URL	Issuance Date

[1]