Sub-Gigahertz Arduino Shield + Remote

ECE 445 Design Document

Team 16

Members: Alexander Beck, Christopher Bladwin, Faruk Toy

TA: Anthony Schroeder

Professor: Arne Fliflet

1. Introduction

1.1 Problem And Solution Overview

Electronic devices are becoming more and more common in our everyday lives. These electronic devices need to be controlled by the user through some medium. Although switches are the main source of control in today's world, they require physical contact. It is more convenient for users to be able to operate multiple devices remotely. For hobbyists who want to build their own project, there is currently a gap in the market for cheap, low power, and long range wireless communication. We, as a team, are motivated to provide solutions to this problem.

We recognized that Arduinos are easy-to-use tools that can be utilized by many people, including non-experts. In recent years, Arduino shields were introduced to the market to create new capabilities using Arduino devices and platforms. They can be used to solve the problem we are interested in. The goal of this project is to create a low-cost, low-power, long-range remote controller Arduino shield. This will allow people to include remote control capability in their projects or add remote control capabilities to their existing devices.

1.2 Visual Aid

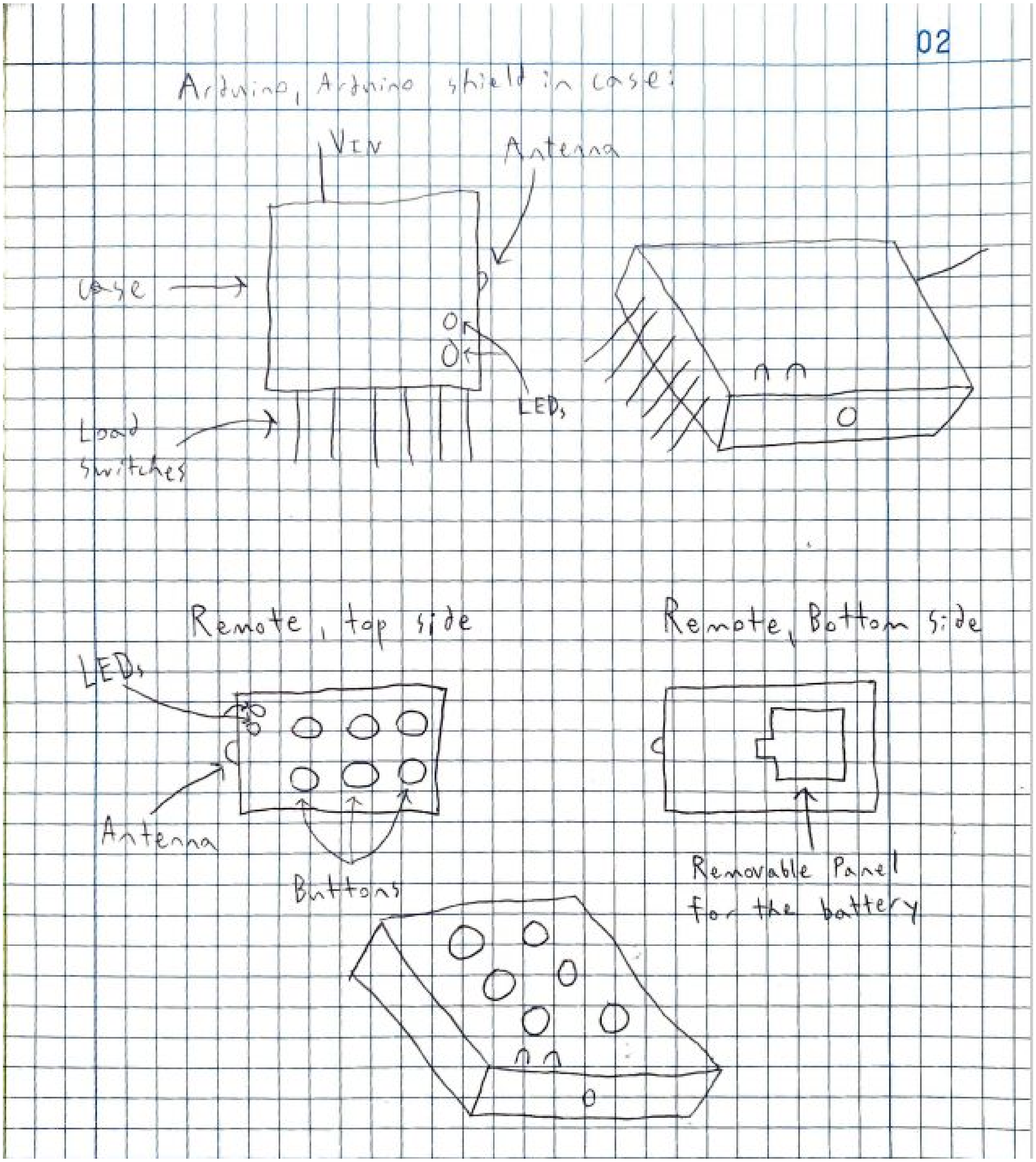


Figure 1: Some rough sketches of what the finished project will look like.

1.3 High Level Requirements

- The arduino shield must be able to convert a 7-17 DC input voltage into 5 volts for the Arduino.

- The remote control must be able to send and receive wireless signals with frequencies lower than 1 gigahertz to the arduino shield.
- The arduino must be able to send and receive wireless signals from the remote and execute a pre-programed action upon receiving a signal from the remote.

2. Design

2.1 Block Diagram

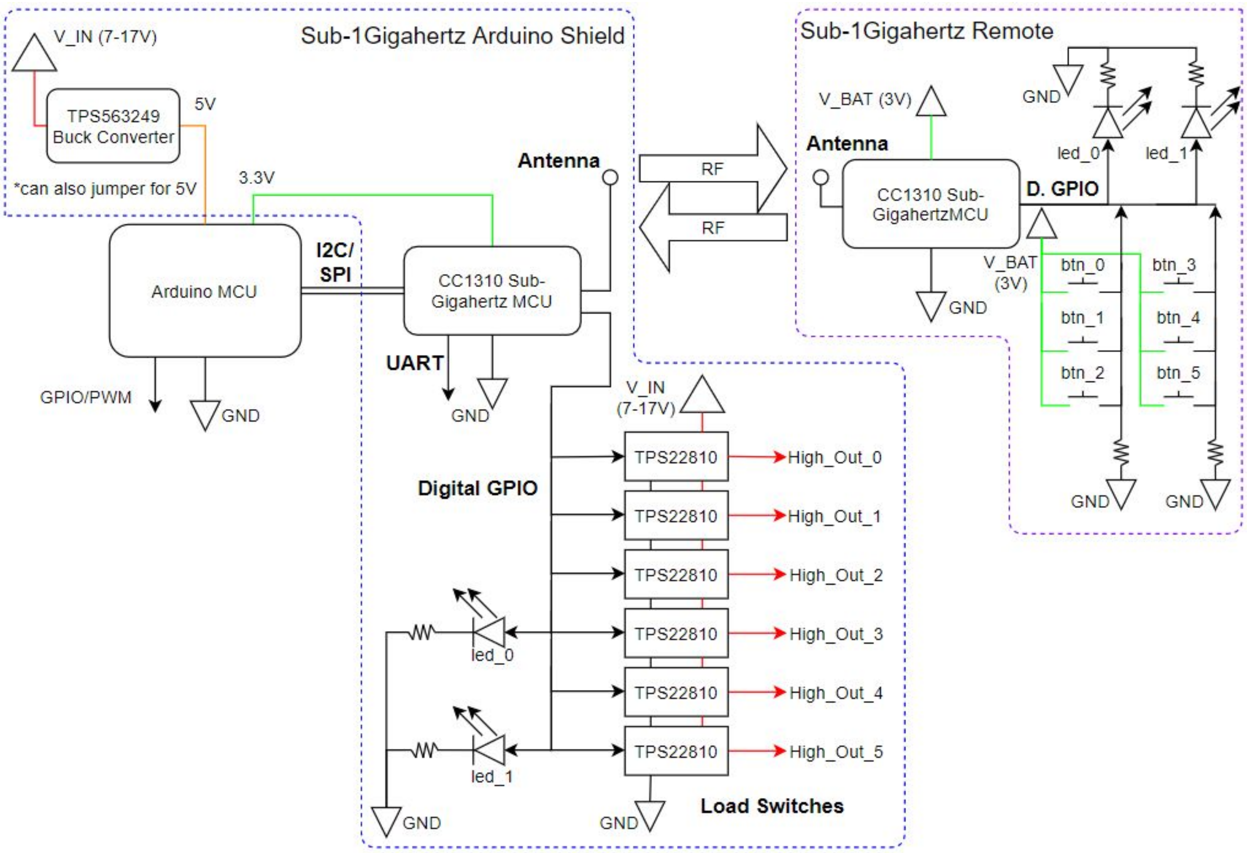
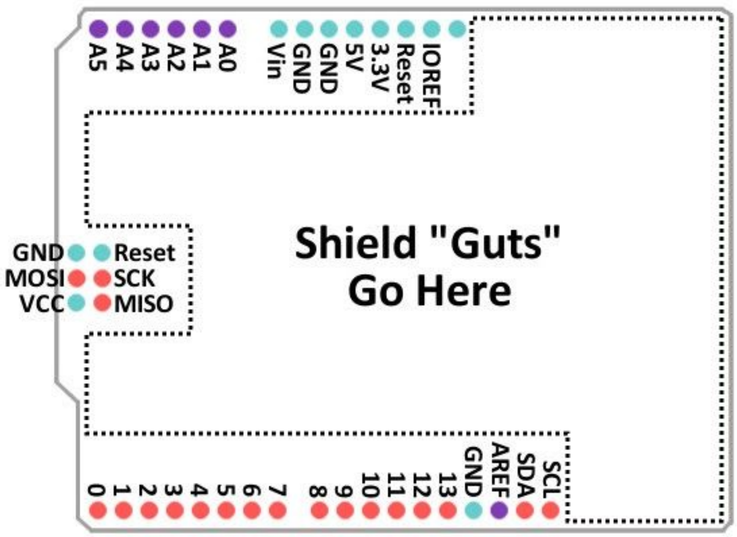


Figure 2: A block diagram of the arduino shield and the remote. The arduino shield will communicate with each other using RF signals.

The sub-gigahertz Arduino shield consists of many individual parts and will be connected to the Arduino. The shield includes a TPS563249 buck converter to ensure that the arduino will have a 5 volt input voltage. It has a CC1310 Sub-gigahertz MCU to send and receive wireless signals with frequencies below 1-gigahertz. The shield includes six switches that can be turned on and off wirelessly using the controller. It's these switches that people will use to wirelessly control their projects. The remote control consists of a CC1310 Sub-gigahertz MCU, six buttons, and a battery. Pressing a button on the remote will cause the CC1310 Sub-gigahertz MCU to send a signal to the Arduino shield, which will then turn on one of the switches.

2.2 Physical Design

The shield's physical design is determined by the footprint of the arduino. The shield will fit on top and have male to female headers which allow the user to connect through and use all the pins. The shield may use one of the digital IO pins of the Arduino as device select for SPI.



[https://learn.sparkfun.com/tutorials/arduino-shields-v2/all]

Figure 3: Pin Layout/Footprint of Arduino Shield

The remote will have a square pattern with LEDs and Buttons. We will ensure that all buttons are easily reached and will try to use a single side PCB.

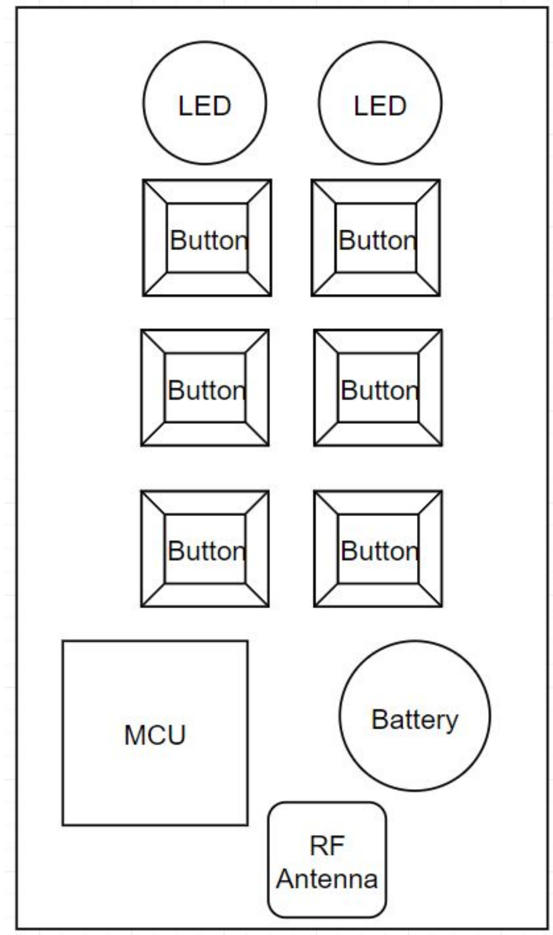


Figure 4: Footprint of Remote

2.3 Subsystem Descriptions

2.3.1 Buck Converter

Requirements	Verification
1. Provide 5 V ±5% from a 7-17V DC source in the range of 0-600 mA	1A. Measure the output voltage using a multimeter, ensuring the output voltage stays within 5% of 5V with a current source load sweeping from 0 to 600 mA.
2. The system should remain below the maximum operating temperature of all	2A. During verification for Requirement 1 an IR thermometer or K-type

the devices at room temperature (20 °C - 30 °C).	thermocouple will be used to check all devices and ensure they remain within their thermal operating temperature.
3. Efficiency of the DC-DC conversion from 7-17 V to 5 V must be 80% or higher at 600 mA.	3A. We will measure the current and voltage at both the input and output when the output current is at 600 mA.We will use this to calculate the total power and efficiency of the buck/PMIC.

2.3.2 Sub-1 Gigahertz Arduino Shield

Requirements	Verification
1. CC1310 controller will be powered from the 3.3 V output of Arduino	1A. Measure 3.3 V output of Arduino with a multimeter to ensure that it will meet the input range of the CC1310 with input and load transients.
2. The shield will be able to send and receive instructions and data with the Arduino over the I2C and/or SPI bus protocols.	2A. Given a command from the Arduino over I2C and/or SPI, we ensure that the CC1310 is able to trigger a response**. 2B. The shield will also be able to send data to the Arduino either by having the Arduino poll a register on the CC1310 or by changing the voltage of a digital pin on the Arduino.
3. The shield will trigger RF/GPIO signals from commands received over I2C and/or SPI bus protocols. The shield will also buffer commands sent by the	3A. We will measure the GPIO output signal with a multimeter to ensure that the output reaches logic high* and returns to logic low* whenever specified by the

remote and/or other shields.	command/message.

2.3.3 Sub-1 Gigahertz Remote

Requirements	Verification
1. The remote will be powered by a coin style battery without the need for a DC-DC converter.	1A. We will measure that a 3V coin style battery (2032 or equivalent) is able to meet the input voltage range of the CC1310 when drawing the maximum rated current of the CC1310 and the active current of all the devices connected to the GPIO. The voltage of the battery will be measured using a multimeter/oscilloscope and the load can be simulated using a current source with a calculated current.
2. The remote will be able to trigger a GPIO output given a command over RF.	2A. Given a command over RF from either a shield or CC1310 evaluation board, or another remote we will check that the CC1310 is able to trigger a response**.
3. The remote will send a command over RF when a button is pressed.	3A. By assigning an RF command/message to a button. When that button is pressed the command/message should trigger a response** from another CC1310 either on a shield, remote, or EVM.

* Logic high refers to the range between VDD and VDD/2. Logic low refers to the range between VCC and VDD/2. VDD refers to the max rated current of that IO value and VCC refers to the effective ground/neutral/zero voltage of the subsystem.

** A triggered response from the CC1310 may be any internal/external state change which is caused by an external stimulus. A triggered response may be but is not limited to an internal register value change, a change in GPIO output, an RF command/message output, a print statement over UART, etc.. External stimulus may be but is not limited to an RF command sent by another Sub-Gigahertz chip either by shield, remote, or EVM; a change in GPIO input; a command/message sent over I2C and/or SPI; etc..

2.4 Tolerance Analysis

The most challenging requirement is going to be sending and receiving RF signals to and from the microcontrollers. This is due to the many components, variables, and variances that are present within our PCB, components, and antenna. We will be using Texas Instruments' various reference designs as the basis for our design and we can consult Simon Josephsen who is the applications manager for the Sub-1 Gigahertz line of products at Texas Instruments. We plan on adding pads for a BNC connector in case our antenna design isn't working. This way we can diagnose any issues using a spectrum analyzer and isolate the issue with the design.

2.5 COVID-19 Contingency Plan

Even if in-person classes and activities are suspended due to COVID-19, we can still program and test communications from the arduino to the MCU. We can still do this thanks to Texas Instruments for providing us an evaluation board. Since they also provided an evaluation board for the TPS22810, we can still test our design with the load switches. This means we will have a demo which proves that the Arduino controls the CC1310 and the CC1310 controls the TPS22810. We can also finish the Arduino API and work on the security protocol.

3. Cost and Schedule

3.1 Cost Analysis

Part Name	Cost Per Part (In $)	# of Parts Needed	Total Cost (In $)
TPS563240 Buck Converter	0.25	1	0.25

Arduino	22.00	1	22.00
CC1310 Sub-Gigahertz MCU	1.78	2	3.56
LED	0.36	4	1.44
TPS22810 Switch	0.141	6	0.846
Button	0.11	6	0.66
Battery	2.99	1	2.99
Resistor Kit (0 ohm - 1M ohm)	5.99	1	5.99

* Does not have all final passives for Buck/Antenna
Labor costs:
$20 per hour * 2.5 hours * 3 people = $150

Total cost of project: $187.74

3.2 Schedule

Week	**Alexander Beck**	**Christopher Baldwin**	**Faruk Toy**
10/5	*Finish schematic for the buck	* Figure out I2C/SPI for Arduino	* Design Remote PCB with Eagle
10/12 (First round of PCB orders)	*Get Bill of Materials for LEDs, Buttons, Headers, Inductors, Capacitors, and Resistors.	* Figure out I2C/SPI for CC1310 * Order Bill of Material Components	* Design Shield PCB with Eagle * Order PCBs
10/19	*Design Case for Arduino + Shield *Order case for Arduino + Shield	* Figure out RF Communication for CC1310	* Design Case for Remote * Order case for remote

10/26		* Figure out RF Communication for CC1310	*Help writing Arduino code
11/2	*Assemble Case for Remote & Arduino + Shield	* Solder Boards	* Solder Boards
11/9	* Test and Verify Shield + Remote	* Test and Verify Shield + Remote	* Test and Verify Shield + Remote
11/16	Everybody prepared for the demonstration		
11/23	*Thanksgiving Break*		
11/30	Everybody prepared for the presentation		
12/7	Everybody writes their part in final paper		

4. Ethics and Safety

The arduino can be ruined if it is exposed to water. This is a concern, since our device is meant to be used in outdoor projects.To prevent unwanted exposure to water and dirt, we will design a case for the arduino and the arduino shield. This case will prevent water and dirt from getting into the arduino and the arduino shield.

The remote control will use a battery as its power source. The battery can be a potential safety issue, since the battery can corrode and it does not have any short circuit protection. To protect the user from battery corrosion, the battery will be kept inside the remote's case. The case should only be opened when the battery needs to be replaced. The case will greatly reduce the risk of battery corrosion exposure, since the user will not touch the battery very often. To help prevent short circuits, we will isolate traces and implement an eFuse.

Since the arduino shield accepts wireless signals, an unauthorized person may try to replicate those signals to access the arduino. To prevent this, we will encrypt the messages that the arduino accepts. This will prevent an unauthorised person from simply listening to an exchange and decoding the message structure.

5. Citations

[1] Texas Instruments, *CC1310 SimpleLink™ Ultra-Low-Power Sub-1 GHz Wireless MCU*, 2018. [Online]. Available: https://www.ti.com/lit/ds/swrs181d/swrs181d.pdf?ts=1600391985951
[2] Texas Instruments, *TPS563249 17-V, 3-A, Constant 1.4-MHz Synchronous Step-Down Voltage Regulator*, 2018. [Online]. Available: https://www.ti.com/lit/ds/symlink/tps563249.pdf?ts=1600392228838
[3] Texas Instruments, *TPS22810, 2.7-18-V, 79-mΩ On-Resistance Load Switch With Thermal Protection*, 2018. [Online]. Available: https://www.ti.com/lit/ds/symlink/tps22810.pdf?ts=1600392338461
[4] Octopart, *Arduino Uno*, n.d.. [Online]. Available: https://datasheet.octopart.com/A000066-Arduino-datasheet-38879526.pdf
[5] IEEE, IEEE Code of Ethics, June 2020. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html