Phase #2 – Secure RF Rolling Codes

CSE321 – Real-time Embedded Operating Systems

Prepared by: Kenneth Pang, kpang
University at Buffalo
School of Engineering and Applied Sciences

November 10, 2025

Table of Contents

Hardware Components

Overview

The current hardware components are separated into two parts for our project. We have separated it into two different groups because there are a transmitter side and a receiver side. The transmitter side is mostly complete for hardware, but the receiver still needs some components to be implemented. The receiver is currently replaced with an LED configuration to show that the prototype is functional. Here are the components for the transmitter and receiver listed below.

On the transmitter side, our components include:

- Arduino UNO R3
- Blue LED
- 2 220Ω Resistors
- 433MHz Transmitter Module
- 4 Buttons with different colors (Red, Green, Blue, Black)
- Liquid Crystal Display
- Potentiometer

On the receiver side, our components currently include:

- Arduino UNO R3
- 4 LEDs with different colors (Red, Green, Blue, Yellow)
- 433MHz Receiver Module
- 4 220Ω Resistors

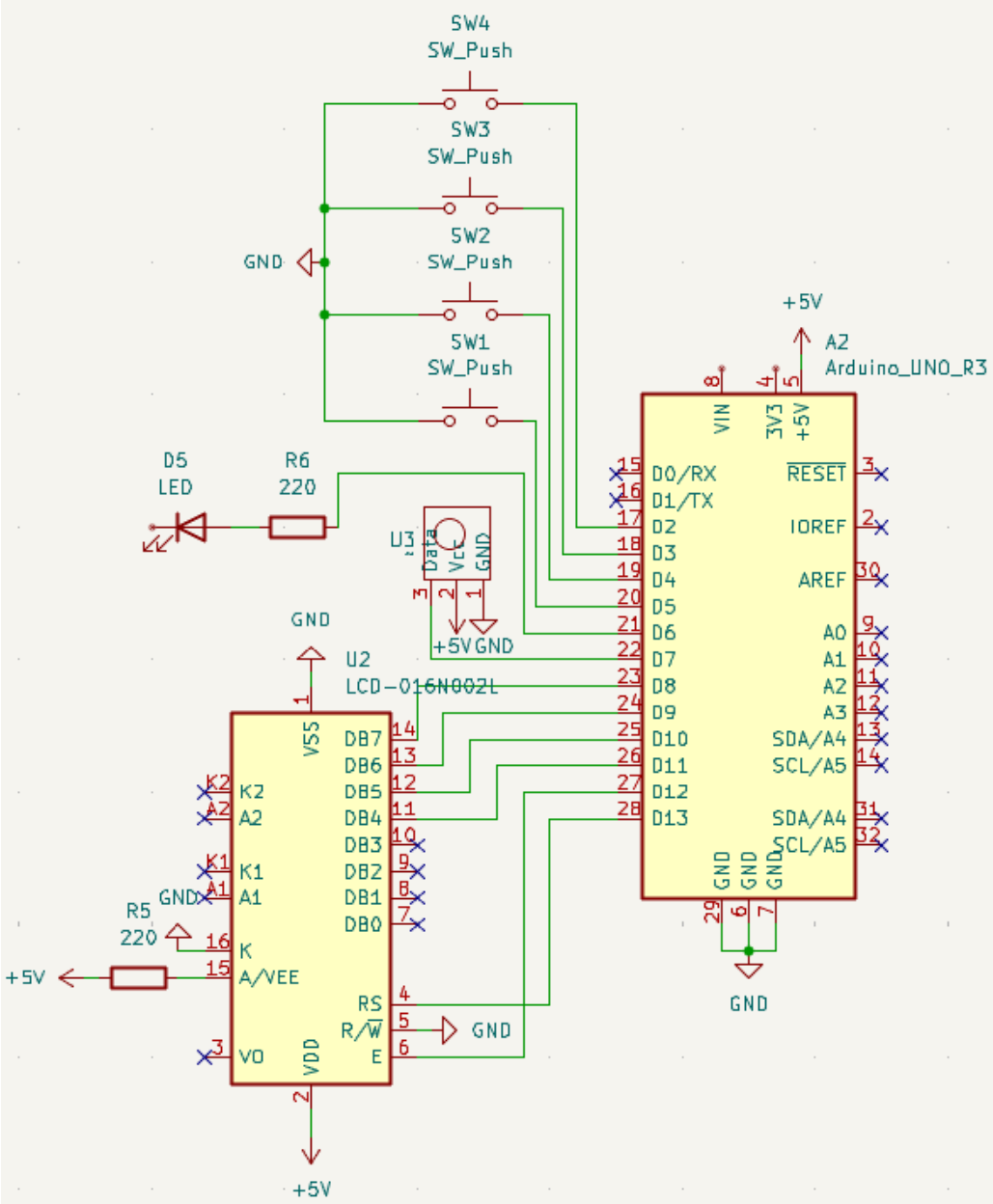
Functionality

The transmitter side has 4 buttons, one for each functionality, like a car key fob. Once pressed, the Arduino should pick up the debounced signal to be handled on the software side. The LED should light up briefly once a signal is sent, and the RF transmitter module handles the actual sending of signal. The LCD is just to display the current message and counter that is being sent, and the potentiometer was added to change the LCD lighting on the screen.

The receiver's side is very simple. The RF receiver module is supposed to collect RF signals sent by the transmitter that can be read by the Arduino. Once the Arduino picks up a signal, it should light up the corresponding LED based on the function that was selected by the user or transmitter side.

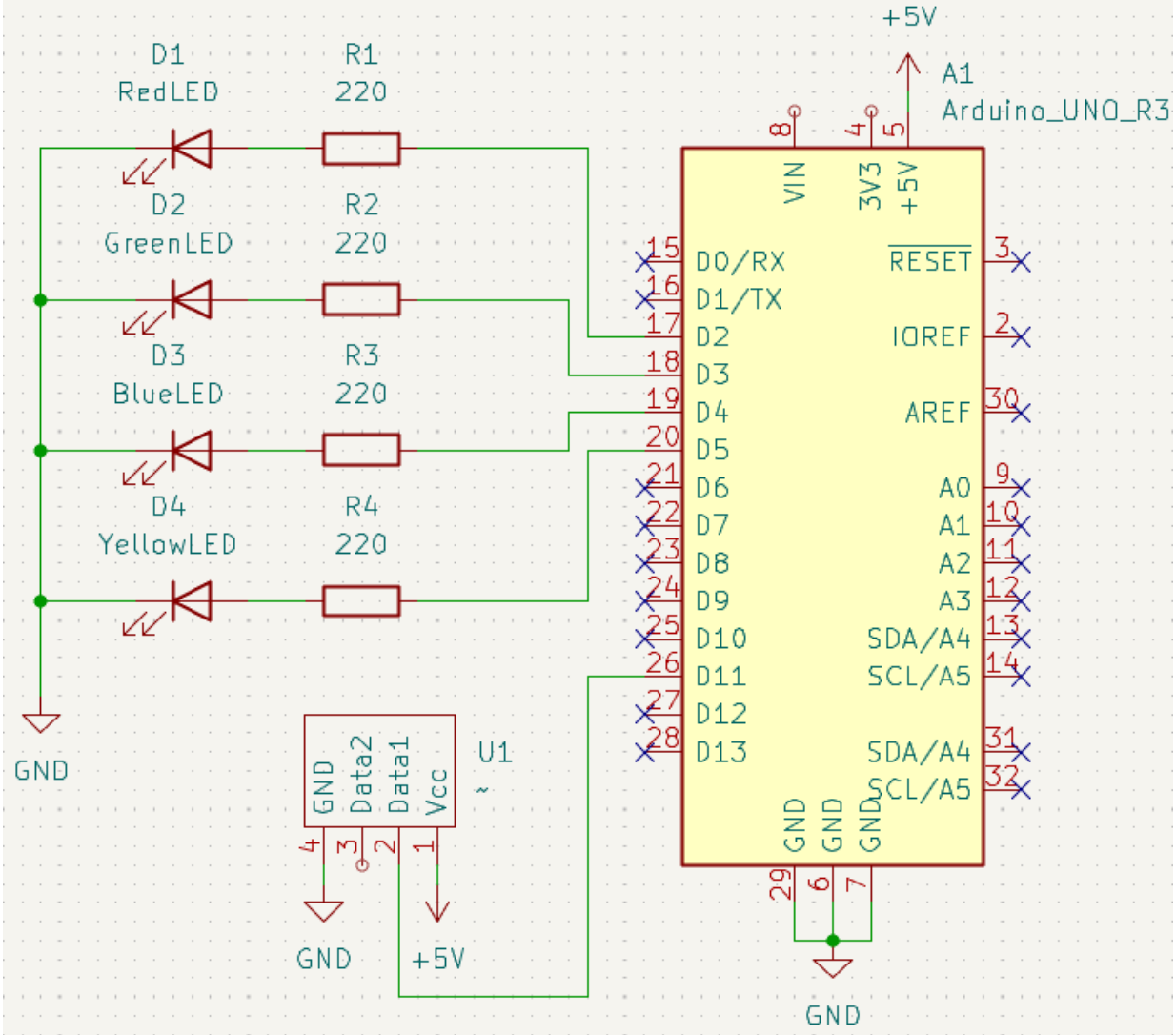
Prototypes

Transmitter Schematic



These components were chosen because the user should be able to interface with the system using buttons so they can do the functionality that they wanted (like unlocking or locking in a car key fob). The transmitter module was chosen because it is a standard to use RF communication in the car key fobs usually. The LCD is provided in the project because this is a simulation and allows us (the students and everyone) to see the message and counter when the signal is ready to be sent. The LED is a way to tell the user that the signal was transmitted, which is also like some car key fobs where they have a brief blink of an LED.

Receiver Schematic



These components were chosen for the current phase of the project to test the receiver and transmitter modules. This allows an easy way to understand and test the hardware of the receiver's side. The Arduino here allows us to also read the data that is picked up on the receiver with simple Serial prints. The LEDs provide a way to change to a new functionality later when implemented in the final project and a way for testing at the moment for the FSM.

Software Components

Project Overview

The code base is currently being managed in GitHub, I push my program every time I make a bit of decent progress on the project. Currently, the code base is split into two different parts, the transmitter and the receiver.

Project Design

The receiver is currently in one main loop only; it is fairly simple at this stage of the project due to some grounding issues (also discussed in later sections). The main loop basically just receives the signal, decrypts the signal, and turns on LEDs based on the functionality.

The transmitter is also split into different handlers where that is similar to how it was documented in phase 1:

1. Input Buttons Handling

   The input handling is just a function that keeps looping the buttons checking if one was pressed. I followed a similar format to our Washing Machine Project to handle the button presses. I wanted to dedicate a task using FreeRTOS but ran into some issues that is discussed later.

2. Encryption Handling (this is still working progress)

   The encryption handling is the function that can be called to generate an AES encrypted hash value that stores our current counter and our function that corresponds to button pressed. However, this is still kind of in progress since I ran into some issues with the library.

3. Transmission of Signal

   The transmission of signal handler is the function that checks if there is a ready encrypted signal to send. This periodically checks the queue if there is a ready signal (I called it a packet).

4. Main Loop

   My main loop was empty because I was trying to use FreeRTOS tasks, but I ran into issues and converted my task functions into the handlers stated above and called it in my main loop to handle all functionality above.

Test Scripts & Plan

Current Hardware Verification

On the transmitter side, there aren't a lot of verification scripts that can be run on the hardware. However, I did set up my program to verify the LCD, buttons, and LED. To test this hardware, run the main program for the transmitter with Serial communication on.

- The LCD when plugged into battery should display "LCD Started." This shows that all initialization code is done and should be working fine if that shows up.
- Press a button and verify that the Serial prints on a function number that corresponds to the button.
- Watch the LED as the button is pressed because it should light up briefly showing the signal was sent.

On the receiving end, there aren't any verification scripts that can be run yet. This is because there isn't a proper implementation for each of the functionalities. Currently, the receiver is set up to test the transmitter module and verify if that is working. To test the transmitter module, both programs for the transmitter and receiver with Serial communication on.

- The receiver module should be able to pick up signals sent by the transmitter. This shows that both the receiver and transmitter modules are working. If it doesn't show up, then it is likely a grounding issue or a module is broken.

Future Hardware Verification

I am planning on testing the hardware individually with test scripts on the transmitter and receiver side. However, I don't have a proper idea on how to test the receiver and transmitter because there is no good way to test each component by itself.

Current Software Verification

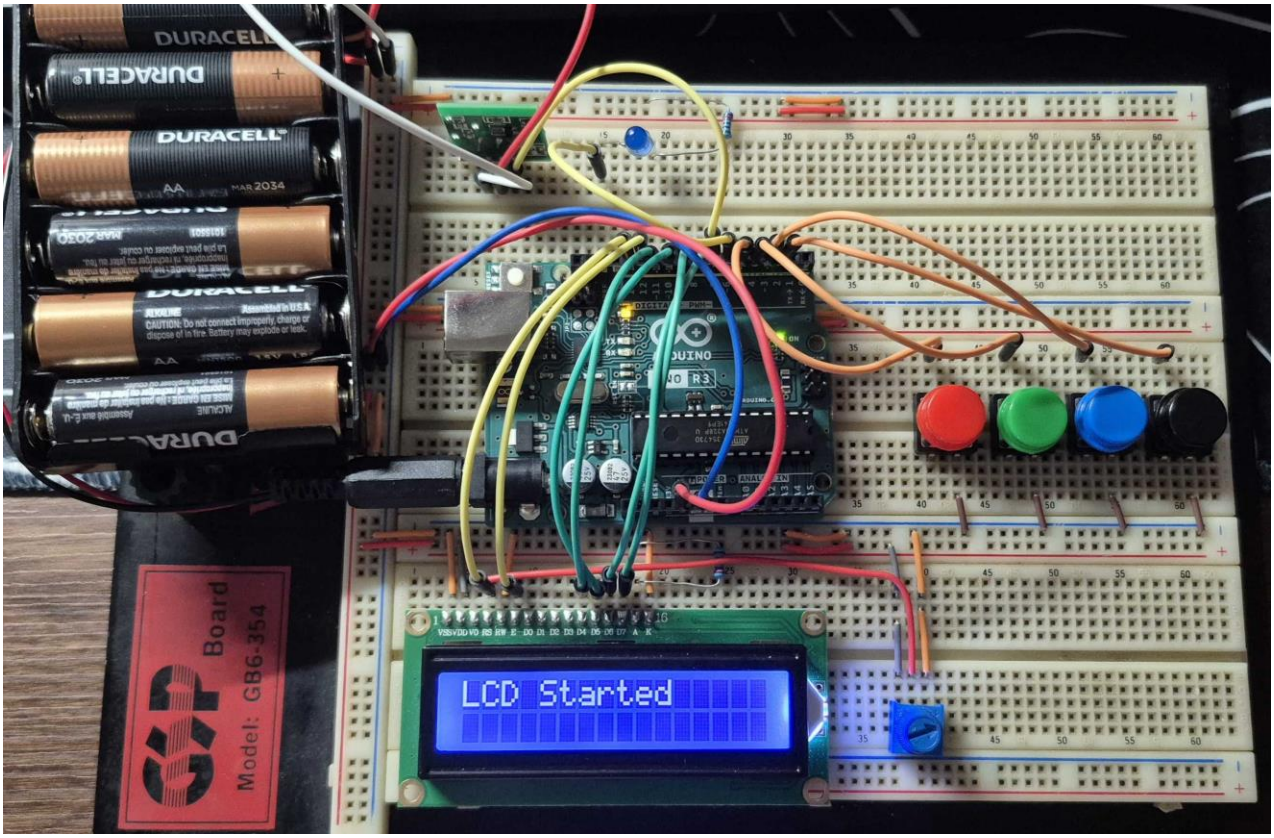
The only software verification for functions that I really need is for the encryption and decryption algorithms. At the moment, I have a simple program setup on Wokwi that fully tests the functions that uses the AESLib library. However, I ran into issues including this library in my main program which will be discussed later.

Future Software Verification

I am planning on creating better test benches for each of my functions that I created so far, but it terms of software, the only function that is key is encryption and decryption. The other verifications would probably be hardware and software verification because they would need user input to test both the software function and the hardware integrity.

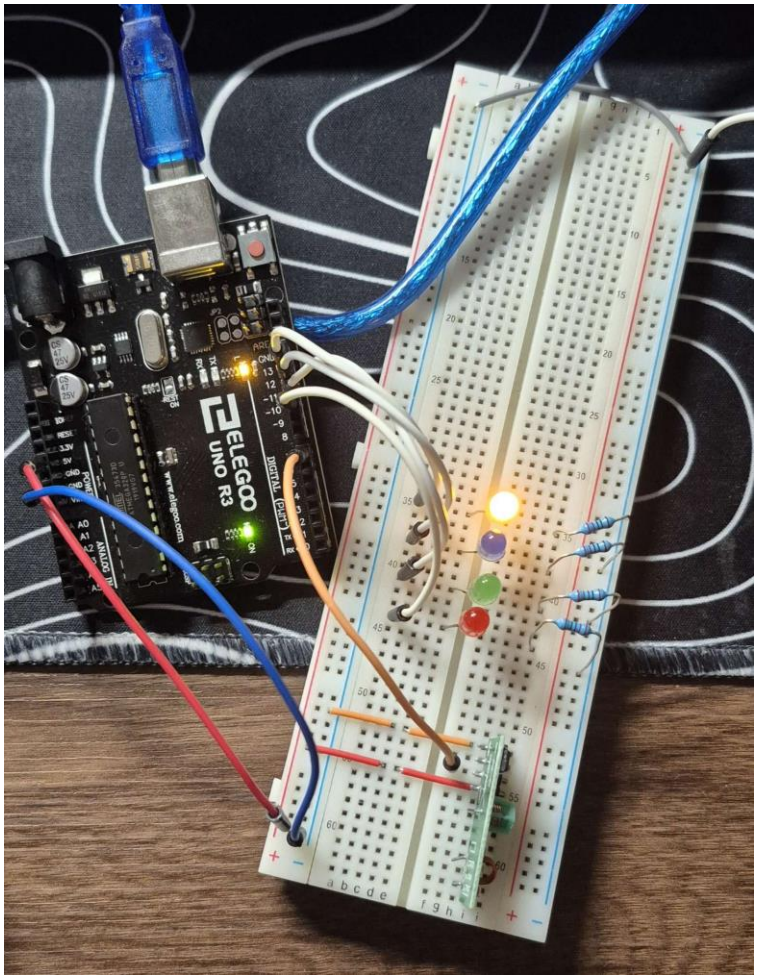
Photos of Project

Transmitter (Key Fob)



On the top of the photo, we have the transmitter module and LED that should be facing outwards and directed towards the receiver when trying to send a signal. The Arduino is in the middle for easier wiring of all the components. The buttons are on the side where it is clearer (less wires), so the user can press them to transmit the function that they want the receiver to do. The LCD is at the bottom of the photo, and the brightness of the back lights can be adjusted with the potentiometer on the side of it. This LCD when starting shows the starting message as shown in the picture. When a button is pressed, it shows a confirmation of the message sent and some values at the bottom representing the counter and the function.

Receiver (Car/Mechanisms)



On the left, we have a basic ELEGOO Arduino Uno R3 that does most of the handling of the finite state machine and reading data from the receiver. On the bottom right, we have the RF receiver module that is supposed to collect the signal that is transmitted from the transmitter side. The LEDs just display what button was pressed; this functionality will be implemented later but will just serve as simple demo for phase 2.

Progress & Observations

Progress Overview

Currently, the transmitter portion of the project is mostly done. There are a few problems with the transmitter that doesn't seem to be working sometimes. The hardware still needs to be debugged and fixed. Also, there seems to be a grounding issue where both the transmitter and receiver modules need to be on the same ground for them to pick up the signal.

The receiver portion is the project that needs more focus. There is still a lot of functionality that needs to be implemented here. I am not perfectly sure what I want to do yet, like opening and closing doors, showing unlock and locked, etc. However, I don't know how realistic the door will be like with 3D printing or just made of cardboard. I also haven't thought of a fun way to show unlock and locked states except for using LEDs.

Setbacks & Solutions

Currently there has been a lot of setbacks in the project that I will list here

1. The Arduino on the transmitter side cannot support FreeRTOS with so little stack space.

   My solution is to maybe buy a better Arduino (like the Mega) or play with the values a little bit to see if there is a possible value on the stack that works, so I can use FreeRTOS in my implementation.

2. The Arduino on the transmitter side cannot support AESLib or Crypto Libraries for encryption and decryption because of limited memory space.

   Similar to before, either buy a better Arduino or I could either find a new library to work with to implement my encryption, or I would have to create a simple encryption and decryption algorithm to work with in this project.

3. The transmitter side and receiver side seem to need to be on the same ground to pick up signals.

   I found a solution where I would need to hook up an external 5V source to the transmitter specifically for the transmitter to send something where the receiver could pick up. However, once I hooked up the external power source, the transmitter seemed to stop working right after a few signals being sent.

4. The receiver side seems to be all fine, except I don’t have a direct plan to implement the functionality yet.

   I am still looking for fun ways to demonstrate my project with the remote key fob basically, I have found a few that I will implement, but the rest might be just simple LEDs at the moment.

It is a bit unfortunate that I ran into these problems so late in the process of making the project because I implemented my tests and functions before adding them to my overall project. When adding the functions to my final program submission, I ran into a lot of these library support and hardware issues. I will include my test programs for these, but I was not able to fully include them in the main program yet.

My strategies to overcome and find errors in the first place were a lot of research online, often finding people with similar issues on the topic, such as the RF transmitter and receiver issue. I searched up error codes in the library showing that the two libraries were using the same timers which caused even more headaches for me since it meant that I basically couldn't use the library. Finally, I just did extensive testing to make sure that the problem was specifically addressed to a single component of my project and looked for solutions around the component.

Conclusion

I would say that I am on a good start for the project, but working alone definitely made me face a lot of challenges. I have a lot of working individual parts but putting them together into my main program has made me face some challenges that set me back from where I would have wanted to be at in the current standing of my project. I would say that at the beginning of phase two, I wanted to be at least 70% done with the project, and just implementing features on the receiver's side. However, it feels that I am closer to the 50% completion side, with still needing to fix my encryption and decryption issue stated above.

Documentation

Here are some works cited and documentation that I found helpful towards my project.

- https://www.freertos.org/
- https://www.airspayce.com/mikem/arduino/RadioHead/index.html
- https://forum.arduino.cc/t/433mhz-using-radiohead-constructor-argument-to-change-rx-tx-pins/1077731
- https://docs.arduino.cc/libraries/crypto/
- https://docs.arduino.cc/libraries/aeslib/
- https://docs.arduino.cc/libraries/queue/