ECE 445

SENIOR DESIGN LABORATORY

Design Document

Bike Alert

Bike Lock with Real-Time Security Monitoring

Team 36

Diego Herrera (dherr4)

Kenneth Kim (kk67)

David Youmaran (dcy2)

TA: Aishee Mondal

Professor: Michael Oelze

March 6, 2025

Table of Contents

Introduction

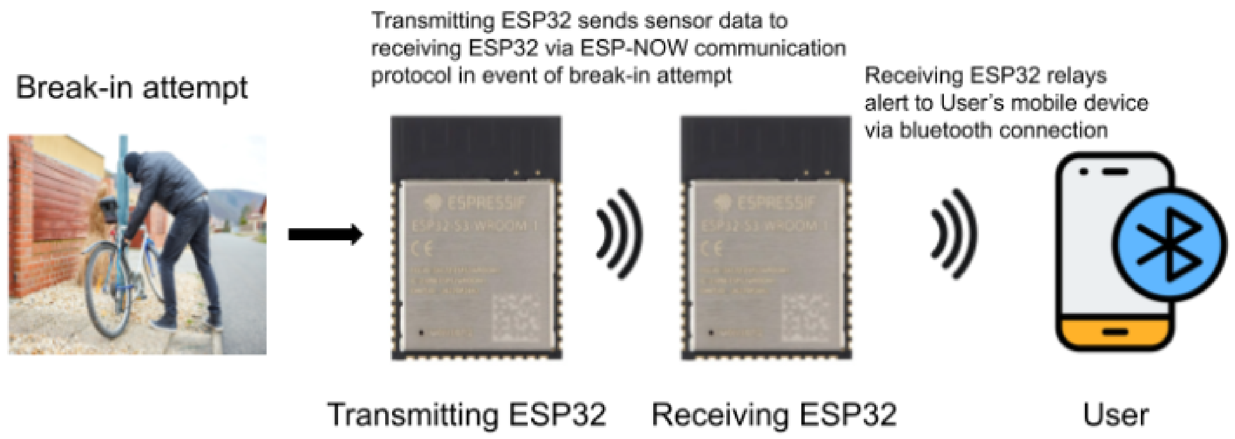
Problem

Bicycle theft is a significant concern in the Champaign-Urbana area, particularly on the University of Illinois campus. Between August 1 and November 13, 2023, 85 bike thefts were reported, up from 69 during the same period in the previous year. Since January 2022, a total of 255 thefts have been reported [1]. Traditional bike locks offer physical security but lack mechanisms to alert owners during theft attempts, leaving bicycles vulnerable to undetected tampering. This gap highlights the need for an enhanced security solution that not only deters theft but also provides real-time notifications to bike owners.

Solution

The Bike Alert system is an advanced security attachment designed to augment standard bike locks with real-time monitoring and alerts. It integrates multiple tamper-detection mechanisms, including Hall-effect sensors to monitor lock engagement and enclosure integrity, and a spring-based adjustable vibration sensor to detect physical tampering. An ESP32 microcontroller processes sensor data and communicates alerts via ESP-NOW and then to a mobile app, providing users with immediate notifications of potential theft attempts. Additionally, the system features an RFID-controlled lock as a secondary locking mechanism, enhancing security even if the primary lock is compromised. The device is battery-powered and rechargeable, ensuring long-lasting and reliable operation. By combining these features, the Bike Alert system offers a comprehensive solution to bicycle theft, addressing both prevention and timely owner awareness.

Visual Aid


Break-in attempt
Transmitting ESP32 sends sensor data to receiving ESP32 via ESP-NOW communication protocol in event of break-in attempt
Receiving ESP32 relays alert to User's mobile device via bluetooth connection
Transmitting ESP32
Receiving ESP32
User

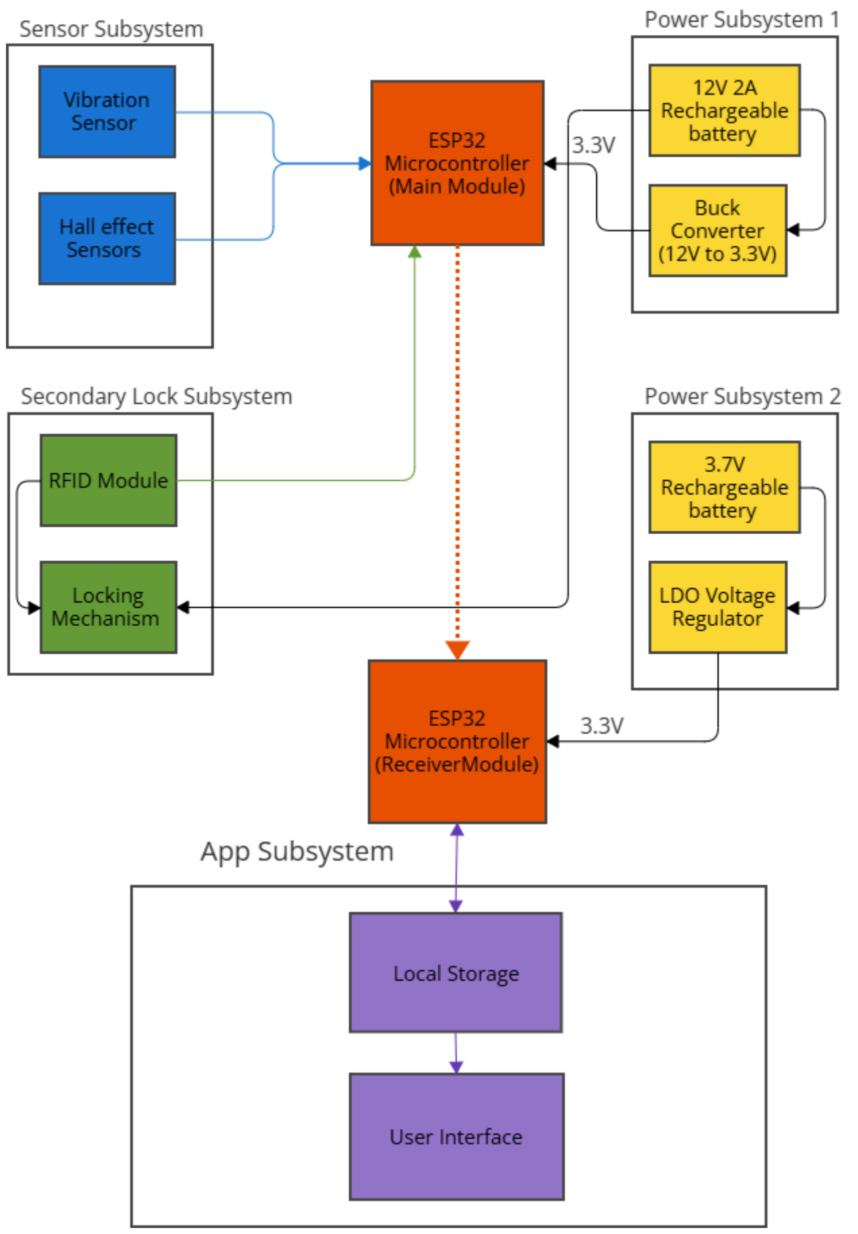
High-Level Requirements List

Our project has the following high-level requirements:

1. The system shall enable real-time communication between the bike lock and the user's mobile device via ESP-NOW, ensuring data transmission within 500 ms over a range of up to 200 meters between two ESP32 modules.
2. The system shall operate for at least 24 hours on a single charge using a rechargeable lithium-ion battery under typical usage conditions.
3. The system shall detect tampering via vibration and Hall-effect sensors, trigger an alert within 1 second, and prevent unauthorized access with an RFID-controlled lock.

Design

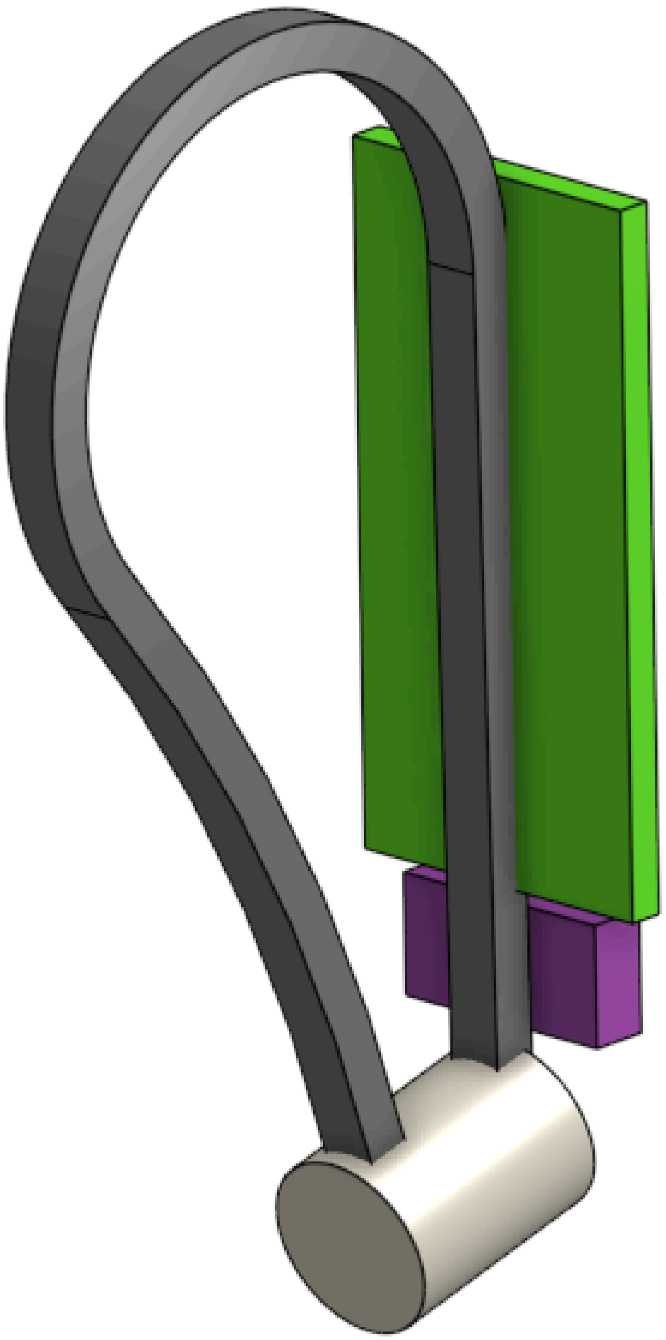
Block Diagram



LEGEND:
Green: SPI connection
Blue: GPIO Connection
Orange: ESPNOW data transfer
Purple: Bluetooth Connection

Subsystem Overview

Physical Design



3D Model of Lock Systems

The 3D Model above illustrates the physical design of our project’s mechanical components. The purple object is a 12V DC Geared Motor with Encoder. The green object is an electric box that houses the PCB, RFID module, and 12V battery. Figure 8 in Appendix A shows the actual chain lock we will be using along with an example motor and housing. The RFID module provides authentication inputs to the ESP32 microcontroller. The ESP32 controls the DC motor using a motor driver. This highlights our secondary locking mechanism, which is described in more detail in the Locking Subsystem. The object below the motor in Figure 8 is a detachable component of the chain lock. The meeting point between this component and the lock is where we will mount a magnet. One of our Hall-effect sensors will be mounted onto the detachable component so that in the event of lock disengagement, the sensor will sense a change in the magnetic field at a desired distance between the sensor and magnet that will cause the sensor to issue an alert. This is described in more detail in the Tampering & Lock Disengagement Detection Subsystem. We will also design and 3D print a rail system that will be used for tamper detection. This rail system is described in more detail in the Tampering & Lock Disengagement Detection Subsystem.

The Tampering & Lock Disengagement Detection Subsystem:

Monitors the lock for unauthorized access and physical tampering. It uses two Hall-effect sensors and one vibration sensor. One Hall-effect sensor detects lock disengagement and the other detects tampering. The lock disengagement Hall-effect sensor will be mounted on the detachable component of a chain lock, and a magnet will be mounted on the point where this component meets with the lock. When the detachable component is detached from the lock, the Hall-effect sensor will move away from the magnet. Once the distance between the sensor and the magnet reaches a desired amount(which we will measure and perform tests to ensure an appropriate value), the sensor's sensed magnetic flux density will surpass a predetermined limit and lock disengagement will be detected.

Our second Hall-effect sensor will be connected to a railing system(which we will design and 3D print), that will be placed in an electric box that houses other parts of our design such as the Main PCB, RFID Module, and 12V battery. The railing system consists of a rail that allows a magnet, which we will mount onto the rail, to move along this rail both backward and forward. Tampering attempts will cause the magnet to move along the rail and, similar to our lock-disengagement sensor, the distance the magnet moves will determine the Hall-effect sensor's sensed magnetic flux density. The sensor will trigger with enough magnet movement.

A vibration sensor, which will be included on our main module PCB, will register physical disturbances. The vibration sensor circuit shown in Figure 1 consists of a vibration sensor, a 10k ohm potentiometer LM393 comparator, 2 1k ohm resistors, 2 10k ohm resistors, 2 0.1uF capacitors, and a status LED. The vibration sensor is made up of a spring and rod; when the sensor experiences vibrations, the spring and rod make contact and the circuit will close [10]. The sensor will convert oscillations due to vibration into electrical signals to be used as input into the LM393 comparator [10]. The LM393 will have a threshold voltage set by the 10k ohm potentiometer; the LM393 outputs 1 if the input's amplitude exceeds this threshold(tampering detected) and 0 otherwise [10].

This subsystem directly communicates with the ESP32 to trigger real-time alerts. Specifically, the outputs of each Hall-effect sensor and the vibration sensor will be connected to ESP32 GPIO pins 4, 5, and 6 respectively as shown in Figure 7 within Appendix A. We will then use Arduino IDE to program the main ESP32 microcontroller to read sensor data and transmit sensor data to the receiver ESP32 microcontroller via ESP-NOW wireless communication protocol.

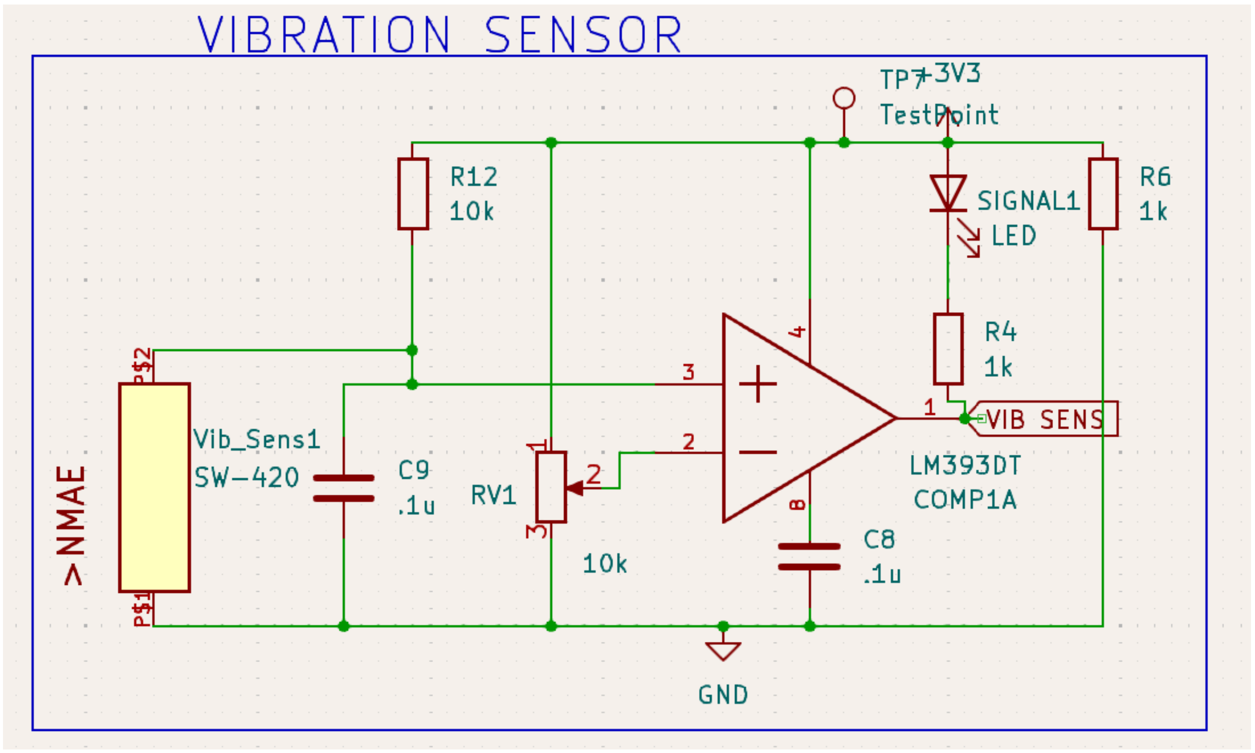


Figure 1: Vibration Sensor Circuit

Requirement	**Verification**
The vibration sensor must distinguish between normal environmental vibrations and tampering.	1. Use varying degrees of force to shake the sensor and determine what amount of force causes the sensor to trigger. 2. Adjust potentiometer value on vibration sensor circuit by turning knob to determine comparator threshold voltage needed to distinguish between normal vibrations and tampering.
The ESP32 must process sensor data and transmit an alert via ESP-NOW within seconds of detection.	1. Use Arduino IDE to program the Main ESP32 to read sensor data. 2.Transmit sensor data to Receiving ESP32 via ESP-NOW wireless communication protocol 3. Measure processing delay by logging sensor input timestamps and comparing them to the timestamps when an alert is transmitted via ESP-NOW.

Hall-effect sensors must detect changes in the magnetic field within desired distance.	1. Use a measuring tape to measure the distance the lock disengagement sensor moves away from the magnet mounted on the lock. 2. Repeat step 1 for tampering detection Hall-effect sensor; in this case, we will use the measuring tape to determine that the sensor detects changes to the magnetic field at desired distance the magnet moves on the rail.

Power Subsystem

The power subsystem is responsible for providing stable and efficient voltage levels to all components of the Bike Alert system. The system consists of two distinct PCBs: the Main PCB, which powers the ESP32, motorized locking mechanism, RFID module, and sensors, and the Receiver PCB, a separate, low-power board that solely receives alerts via ESP-NOW.

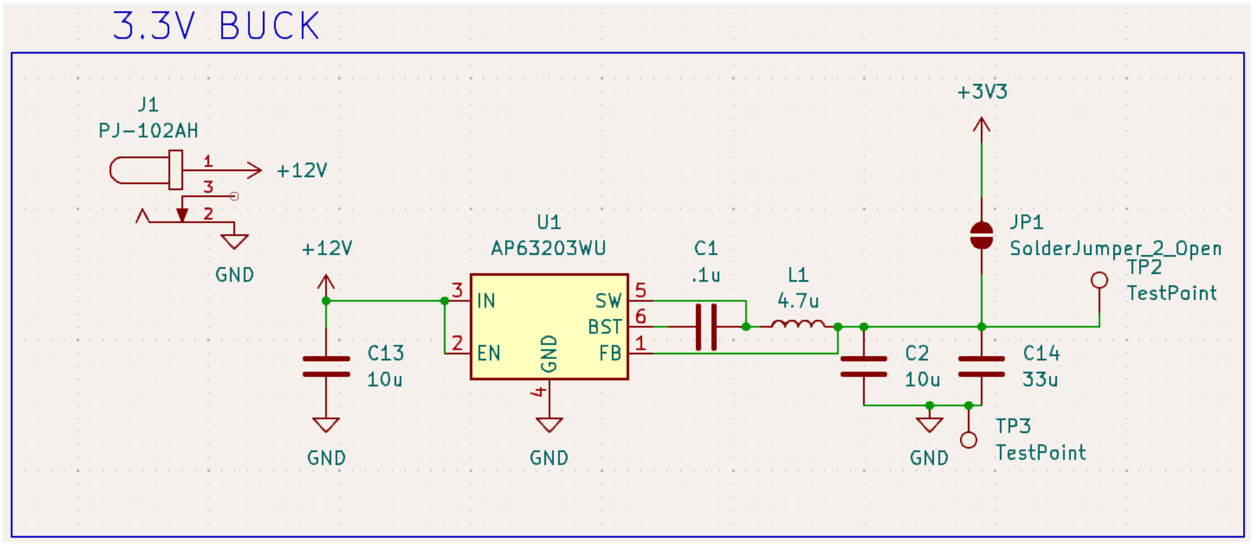


Figure 2: Power Subsystem

In the Main PCB Power Configuration, a rechargeable 12V lithium-ion battery serves as the primary power source. A switching regulator (buck converter) is used to step down the 12V to 3.3V, efficiently supplying power to the ESP32 and sensors while avoiding the inefficiencies of an LDO (Figure 2). The motorized locking mechanism operates directly from the 12V battery, ensuring sufficient power for activation. Additionally, the RFID module and sensors, which require a 3.3V input, are powered through the buck converter to maintain a stable voltage supply.

For the Receiver PCB Power Configuration, a separate 3.7V 3000mAh replaceable lithium-ion battery is used. Since this board consumes minimal power, an LDO regulator steps down the 3.7V to 3.3V, providing a steady power supply for the ESP32 while maintaining energy efficiency.

To ensure continuous operation and monitor battery levels effectively, the system implements a voltage divider network for each power source, allowing the ESP32 to read and calculate battery levels through its ADC (Analog-to-Digital Converter) pins.

For the Main PCB, which is powered by a 12V lithium-ion battery, a voltage divider (Figure 3) consisting of 100kΩ and 27.4kΩ resistors steps down the voltage to approximately 2.58V, which is within the ESP32's ADC input range as well a some protection diodes. The ESP32 will use this reading to estimate the battery percentage and display it accordingly.

For the Receiver PCB, powered by a 3.7V 3000mAh lithium-ion battery, another voltage divider using two 100kΩ resistors reduces the voltage to approximately 1.85V for safe ADC measurement. The ESP32 on the receiver board will also calculate and display battery levels to ensure users are aware of remaining charge.

Custom firmware will be developed to periodically read these ADC values, convert them into actual battery percentages, and display them on the mobile application ensuring timely battery replacement or recharging as needed.

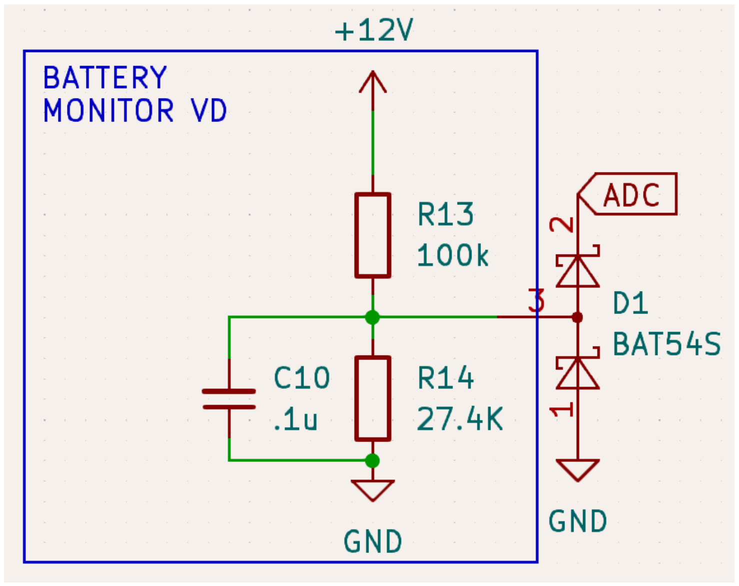


Figure 3: Battery Monitor

Requirements and Verification for Power Subsystem:

Requirement	Verification
The 12V to 3.3V buck converter must output a stable 3.3V ± 0.1V at up to 500mA.	Use a multimeter to verify the voltage remains within range under different load conditions.
The 3.7V to 3.3V LDO must supply a steady 3.3V output to the receiver ESP32.	Measure output voltage with a multimeter under normal operating conditions.
The 12V battery must provide at least 500mA to power the motorized locking mechanism.	Measure current draw while actuating the motor to confirm sufficient supply.

The receiver PCB must function for at least 24 hours on a single battery charge.	Perform continuous operation tests to see consumption over time.

Locking Subsystem

The locking subsystem consists of an encoder motor controlled by the ESP32, which turns a specific amount to engage or disengage the locking mechanism. The RFID module provides secure user authentication before the motor activates.

The locking mechanism operation is managed by a 12V DC encoder motor, which ensures precise movement. The motor is driven by a TB6612FNG motor driver, which controls the motor's movement and direction. The ESP32 sends PWM signals via GPIO pins to the motor driver to regulate speed and direction. The RFID module (RC522) communicates with the ESP32 over SPI, and upon scanning a registered RFID tag, the ESP32 sends a command to unlock the motorized mechanism.

For wiring, the RFID module connects to the ESP32 as follows: SS to GPIO10, MOSI to GPIO11, SCK to GPIO12, MISO to GPIO13, and RST to GPIO14. The motor driver (TB6612FNG) interfaces with the ESP32 through AIN1 (GPIO38), AIN2 (GPIO39), and PWM_A (GPIO40) to control the motor's direction and speed. The encoder motor itself provides position feedback to the ESP32 via ENC_A (GPIO41) and ENC_B (GPIO42) to ensure precise movement and prevent over-rotation. All these connections can be seen in the appendix figure 7.

The communication protocols used in this subsystem include SPI for the RFID module, PWM and GPIO control for the motor driver, and quadrature encoding for position tracking. The ESP32 processes encoder signals to confirm the motor's position and avoid excessive rotation.

Requirements and Verification for Locking Subsystem:

Requirement	Verification
The RFID module must authenticate a valid tag within 500 ms.	1. Test response time by scanning multiple tags and measuring delay with a high-resolution timer. 2. Conduct security testing with unauthorized tags to ensure proper rejection. 3. Log successful and failed authentication attempts to verify system reliability.
The motor must rotate a precise amount to engage and disengage the lock reliably.	1. Monitor encoder feedback and compare expected vs. actual motor positions. 2. Use an oscilloscope to verify PWM signal integrity and stability. 3. Conduct repeated locking/unlocking cycles and measure any deviation in motor position using high-speed video analysis or an encoder data logger.

The motor driver must operate within a 12V ± 5% range.	1. Measure voltage across the motor driver input under varying load conditions to confirm it remains within 11.4V to 12.6V. 2. Use a current probe to monitor power draw and verify it remains within expected parameters. 3. Perform mechanical stress tests by simulating lock resistance and ensuring the motor consistently completes its rotation.
The ESP32 must properly interpret encoder signals to prevent over-rotation.	Monitor encoder feedback in real-time and confirm expected pulse counts for each locking/unlocking cycle.

By implementing these power and locking system improvements, the Bike Alert system ensures efficient power distribution, precise locking control, and reliable authentication mechanisms.

Microcontroller Subsystem

The ESP32 microcontroller serves as the central hub of the Bike Alert system, coordinating communication between all subsystems. It processes real-time sensor data, controls the locking mechanism, and transmits alerts and authentication data via Bluetooth and ESP-NOW. The microcontroller is responsible for managing inputs from the Hall-effect sensors, vibration sensor, and RFID module while controlling the motorized lock through the motor driver. Additionally, it handles wireless communication to ensure seamless data transmission between the main and receiver PCBs, as well as to the mobile application.

The ESP32 board includes a programming circuit (Figure 4) that facilitates updates and debugging. This circuit consists of a boot and reset button, which are standard for programming the ESP32. The programming interface includes a USB-UART header with pins that connect to the boot/reset buttons and a MOSFET circuit that automates the programming process. Additionally, an optional USB micro port is available for direct programming, offering an alternative method for flashing firmware. This setup ensures that the ESP32 can be easily programmed and debugged during development and testing, allowing seamless updates to the firmware and system functionality [9].

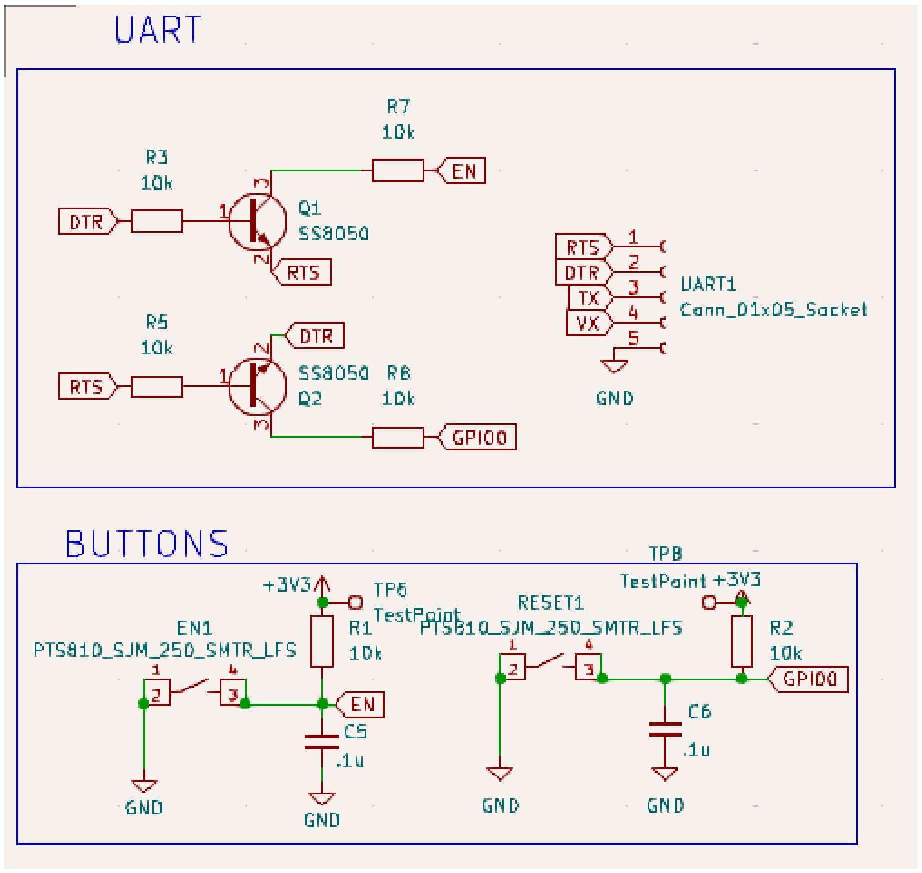


Figure 4: UART and Button Circuit Diagram

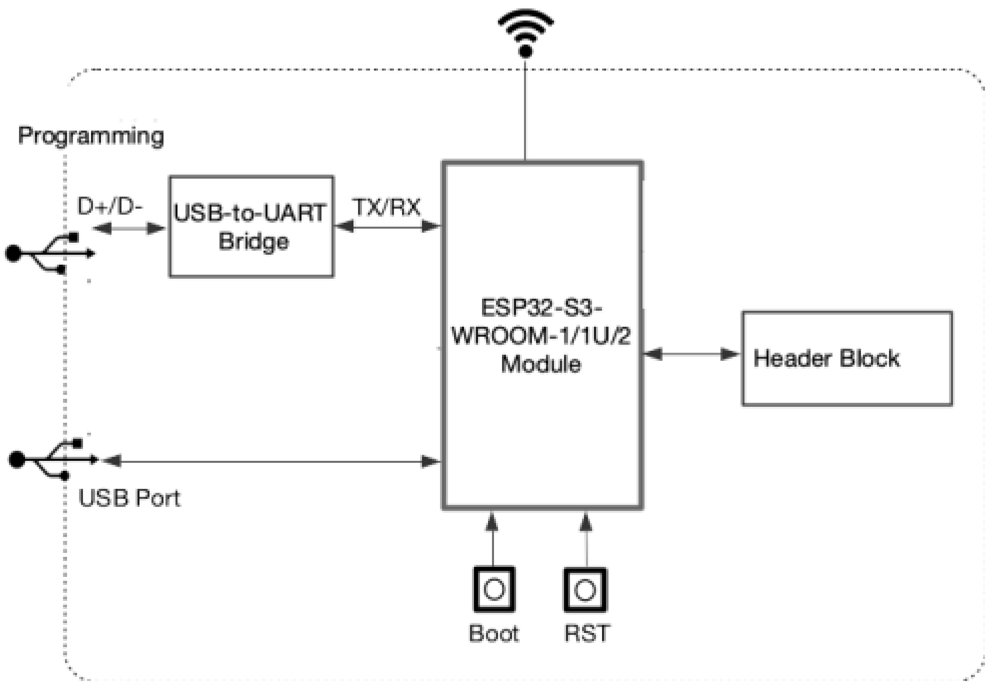


Figure 5: ESP-32 Programs Block Diagram [9]

Requirements and Verification for Microcontroller:

Component	Requirement	Verification
ESP32 Microcontroller	Must process sensor inputs within 1 second and transmit alerts via ESP-NOW within 2 seconds of tampering detection. Must operate at 3.3V with a current draw of up to 120mA in active mode and an additional	1. Measure processing delay by logging sensor input timestamps and comparing them to the timestamps when an alert is transmitted. 2. Use a multimeter to measure voltage stability on the 3.3V rail under various loads. 3. Monitor current draw using an oscilloscope or multimeter in

	160mA during wireless transmission.	different operating states (idle, processing, transmitting).
Sensor Data Processing & Alert Transmission	Must accurately process data from the Hall-effect sensors, vibration sensor, and RFID module and transmit alerts within 2 seconds via ESP-NOW.	1. Use Arduino IDE to program the Main ESP32 to read sensor data. 2.Transmit sensor data to Receiving ESP32 via ESP-NOW wireless communication protocol 3. Measure processing delay by logging sensor input timestamps and comparing them to the timestamps when an alert is transmitted via ESP-NOW.
Motorized Lock Control	Must generate PWM signals to precisely control the motorized lock and prevent over-rotation.	1. Monitor encoder feedback and compare expected vs. actual motor positions. 2. Use an oscilloscope to verify PWM signal integrity and stability. 3. Conduct repeated locking/unlocking cycles and measure any deviation in motor position.

Bluetooth Communication	Must support Bluetooth Low Energy (BLE) communication with a mobile app, ensuring real-time data exchange.	1. Monitor data exchange between the ESP-32 and mobile app, ensure that the ESP is sending data using serial monitoring to print confirmation messages. 2. Ensure the app has received said data, by checking local storage in real time. 3. Send arbitrary data to the ESP-32 and check functions return type, should be True if data was sent successfully. 4. Ensure ESP-32 receives data from the app by using serial monitoring and checking if data printed matches the data sent.
ESP-NOW Communication	Must support communication between the server ESP-32 and the receiver ESP-32.	1. Broadcast data from the server ESP-32 to the receiver's MAC address and print out sent confirmations 2. Print the data from the receiver and ensure it aligns with the data sent from the server.

Connection to Other Units

- Tampering & Lock Disengagement Detection Subsystem: Reads data from Hall-effect sensors and vibration sensors to detect unauthorized access.
- Locking Subsystem: Controls motorized locking mechanism based on authentication signals from the RFID module.
- Application Subsystem: Communicates via BLE with the mobile app to send alerts and receive user commands.
- Power Subsystem: Receives regulated 3.3V from the buck converter, ensuring stable operation.

By implementing these measures, the microcontroller subsystem ensures real-time processing, secure communication, and efficient control of all Bike Alert system components.

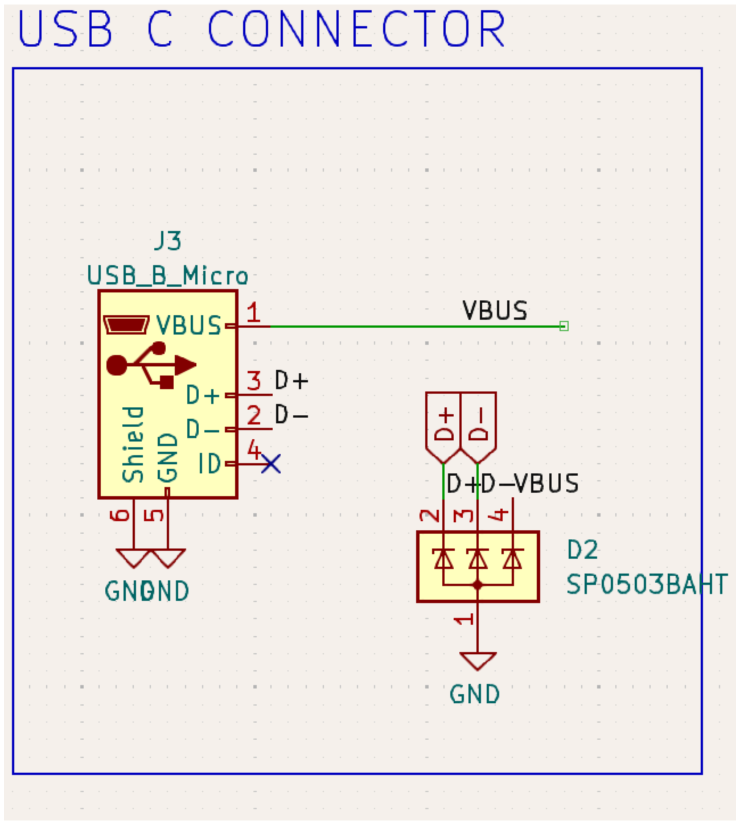


Figure 6: USB-C Connector Wiring

The Application Subsystem

Consists of the receiver ESP-32, which processes alerts sent from the main lock ESP-32 and relays them to the user. This subsystem ensures that notifications are delivered promptly to keep the owner informed of potential theft attempts.

The ESP-32 will act as a BLE peripheral, broadcasting sensor data for our mobile application to read and write. The sender ESP-32 will have to be configured into a BLE server, able to transmit data quickly towards BLE clients (our receiver). Setup code for both ESP-32's will be written using Arduino IDE and the provided BLE library and code will be flashed onto each microcontroller and verified via unit tests. Data between the two ESP-32 will be transmitted via the ESP-NOW wireless communication protocol where a server can broadcast sensor data to a hard configured MAC address of the receiver ESP. We will configure a data payload sending our data from our sensors and current battery usage. Additionally our mobile application will need access to the ESP-32's via Apple's CoreBluetooth library where a bluetooth device manager will need to be configured to retrieve sensor data and display it to the users in a timely fashion. With the current capabilities expected for the mobile application, a large database such as Firebase or AWS cloud may be impractical. We have instead chosen to use a lightweight offline local storage method using Apple's CoreData library. A data model will need to be defined to store all the relevant sensor data, which we can define using the Entity class in CoreData. Once the ESP-32 sends data via BLE to our mobile application, we can use our Entity to save the needed data locally and use it for displaying alarms and battery monitoring.

Requirements	**Verification**
One ESP-32 must act as a sender, using ESP-NOW Peers to send data to the receiver	Use a serial monitor, where the sender ESP-32 prints out confirmation of sending messages, while the receiver prints out the received data/message.
Receiver ESP-32 must send data to the mobile application via bluetooth	Debug using Apple's CoreBluetooth framework, initialize a CBCentralManager and develop unit tests to ensure that each different packet is sent and received to

	the mobile application.
Mobile App must be able to communicate with the ESP-32 and translate real time data into notifications	Similar to the above, we can use the CoreBluetooth framework to write unit tests to send data to the ESP-32 and monitor the packets sent to the microcontroller and see if they match with the ones we sent.

Tolerance Analysis

An aspect of the design that draws concerns is the ESP-NOW connections between the main ESP-32 connected to our bike lock subsystem, and our receiver ESP-32 that will be uploading the data to our mobile application. The main drawbacks of this have to do with long range connection between the two devices. We can test this by sending out periodic broadcast messages from our sender ESP-32 and checking how many packets are dropped and how many are received by our receiver ESP-32 and how many are uploaded successfully to the cloud. We can use this to calculate a suitable range on the bike lock.

A critical aspect of the Bike Alert design is ensuring that the power subsystem can reliably supply sufficient energy to all components while maintaining efficiency and longevity. The system must support a 12V 0.15A motor, an ESP32 microcontroller, an

RFID module, Hall-effect sensors, and a vibration sensor, all running off a 12V lithium battery. The key concern is whether the battery capacity is adequate for daily operation, given typical usage conditions.

Power Consumption Breakdown

1. Motor (12V, 0.15A) [7].

- Operates for a maximum of 3 seconds per use, with an estimated 6 uses per day.
- Energy per use: 12V×0.15A×3s=5.4J
- Total daily energy: 5.4J×6=32.4J/3600=0.009Wh

2. ESP32 Microcontroller (3.3V, ~240mA average draw) [8].

- Always active.
- Power consumption: 3.3V×0.24A=0.792W
- Total daily energy: 0.792W×24h=19.01Wh

3. RFID Module (3.3V, 26mA) [6].

- Always active.
- Power consumption: 3.3V×0.026A=0.0858W
- Total daily energy: 0.0858W×24h=2.06Wh

4. Vibration Sensor (3.3V, 15mA) [5].

- Always active.
- Power consumption: 3.3V×0.015A=0.0495W

- Total daily energy: 0.0495W×24h=1.19Wh

5. Hall-Effect Sensors, Resistors etc.

- Power draw is negligible. Estimated contribution is minimal (~0.5Wh).

Battery Feasibility Analysis

12V 2.6Ah Battery

- Total capacity: 12V×2.6Ah=31.2Wh
- Total system energy requirement: 19.01Wh+2.06Wh+1.19Wh+0.5Wh+.009Wh=22.78Wh
- The 2.4Ah battery can sustain the system for over 24 hours, making it a good choice

By selecting the appropriate battery the Bike Alert system ensures reliable operation.

Ethics and Safety

The Bike Alert system must adhere to ethical and safety guidelines to ensure responsible development and usage. The project aligns with the IEEE Code of Ethics [2] by promoting user security and preventing theft without infringing on privacy. Ethical considerations include:

- Privacy Protection: The system does not store or transmit personally identifiable information, ensuring compliance with ethical data handling practices.

- Safety in Operation: The lock’s motor/solenoid mechanism must be designed to prevent accidental injury. Electrical components must be insulated to prevent short circuits or fire hazards.
- Regulatory Compliance: The system should comply with FCC regulations for wireless communication and battery safety standards such as UN 38.3 for lithium-ion batteries [3].
- Misuse Prevention: The system must prevent unauthorized access and ensure that only the rightful owner can disable the security features.

By addressing these ethical and safety considerations, the Bike Alert system ensures a secure, responsible, and effective solution to bicycle theft.

Cost Analysis

Labor (Each Team Member): ($37/hour) * 2.5 * 75 hours = **$6,937.50**

Total Labor: $6,937.50 * 3 = **$20,812.50**

Bill of Materials

Description	**Manufacturer**	**QTY**	**Extended Price**	**Link**
DC Gear Motor 12V Low Speed 10RPM Encoder Metal Gearmotor	Bemonoc	1	$15.88	Link
10PC Male Header Pins,40 pin Header Strip	MCIGICM	1	$4.99	Link
12V Rechargeable Lithium-ion Battery Pack with Charger	Rapthor	1	$23.89	Link
2PC RFID Reader Writer RC522 Sensor Module	WWZMDiB	1	$6.99	Link
5PC SW-420 Vibration Sensor Module	EC Buying	1	$4.99	Link
CONN PWR JACK 2X5.5MM SOLDER	Same Sky	1	$0.76	Link
CONN HEADER VERT 2POS 2.5MM	JST Sales America Inc.	1	$0.10	Link
SLOW VIBRATION SENSOR SWITCH (HA	Adafruit Industries LLC	1	$0.95	Link

IC REG BUCK 3.3V 2A TSOT23-6	Diodes Incorporated	1	$1.38	Link
CAP TANT 10UF 20% 10V 1206	KEMET	1	$0.35	Link
SWITCH TACTILE SPST-NO 0.05A 16V	C&K	2	$0.98	Link
KBT 3.7V 3000mAh Rechargeable Li-ion Battery with JST 2.54 2Pin Plug, Charging Cable for Voice Power Amplifier, Speaker	KBT	1	$12.99	Link
3.7V USB Charging Cable XH 2.54mm 2Pin Plug to USB Connector Lithium Battery Charger for Aircraft Helicopter Toys 3.7v Battery Charger Cord	GMBYLBY	1	$4.16	Link
Inductor 4.7UH 4.5A 26 MOHM SMD	Bourns Inc.	1	$0.44	Link
Grand Total - Parts and Labor			$20,891.35	

SMD parts ordered from Electronic Services Shop (No cost):

Part	**QTY**
Diode - SP0503BAHTG	3
Voltage Regulator - AZ1117CD-3.3TRG1	1
Transistor - SS8050-G	5
Comparator - LM393DR2	1

Micro Controller - ESP32-S3-WROOM	2
Connector - Micro USB-B Connector	2
Diode - LTST-C150CKT	2
Capacitor - 10 uF (0805)	9
Capacitor - 0.1 uF (0805)	15
Resistor - 10k Ohm (0805)	17
Diode - BAT54SLT1G	3

Schedule

Week	**Tasks**	**Person**
Week of March 3rd	1. Design Documents are due Thursday. 2. Finish ordering parts used for testing 2. Continue working on circuitry for breadboard demo	1,2,3 - Everyone
Week of March 10th	1. Breadboard Demo Wednesday @ 2 pm	1 - Everyone

Week of March 17th	SPRING BREAK	Nobody
Week of March 24th	1. Unit test Test/Revise PCB 2. Finish MVP for app, get it ready for testing with microcontroller	1 - Diego, David 2 - Kenny
Week of March 31st	1. Unit test Test/Revise PCB 2. Test connection between the two ESP-32s	1 - Diego, David 2 - Kenny
Week of April 7th	1. Unit test Test/Revise PCB 2. Finalize testing and add visuals for app	1 - Diego, David 2 - Kenny
Week of April 14th	1. Get Final PCB working 2. Finalize app, do some range tests	1 - Diego, David 2 - Kenny
Week of April 21st	Practice our mock demos, finalize presentations	Everyone
Week of April 28th	Final demo with TA and continue working on final papers and documentation	Everyone
Week of May 5th	Final Presentations	Everyone

Works Cited

[1] CU-CitizenAccess, “Bike thefts surge on University of Illinois campus, but online reporting system aids police investigations,” Jan. 2024. [Online]. Available: https://cu-citizenaccess.org/2024/01/bike-thefts-surge-on-university-of-illinois-campus-but-online-reporting-system-aids-police-investigations/. [Accessed: Feb. 9, 2025].

[2] IEEE, “IEEE Code of Ethics.” [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: Feb. 10, 2025].

[3] United Nations, “UN Manual of Tests and Criteria, Section 38.3.” [Online]. Available: https://unece.org/trans/dangerous-goods/un-manual-tests-and-criteria. [Accessed: Feb. 10, 2025]

[4] Honeywell, *Bipolar Digital Hall-Effect Sensor IC Installation Instructions SS41*, 2020. [Online]. Available: https://prod-edam.honeywell.com/.../sps-siot-bipolar-digital-hall-effect-sensor-ic-installation-instructions-ss41-pk87881-6-en-ciid-50342.pdf. [Accessed: Feb. 10, 2025]

[5] Components101, "SW-420 Vibration Sensor Module," *Components101*, 2022. [Online]. Available: https://components101.com/sensors/sw-420-vibration-sensor-module. [Accessed: Feb. 10, 2025]

[6] RFID Technology, *RFID Module 4411 Datasheet*, 2021. [Online]. Available: https://mm.digikey.com/.../4411_CN0090%20other%20related%20document%20%281%29.pdf. [Accessed: Feb. 10, 2025]

[7] Tronsun Motor, *DC Brush Motor with Encoder Datasheet*, 2019. [Online]. Available: https://www.tronsunmotor.com/.../e78fcf93ed604a64c69852b5db49a03f.pdf. [Accessed: Feb. 28, 2025]

[8] Espressif Systems, *ESP32-S3-WROOM-1/WROOM-1U Datasheet*, 2021. [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp32-s3-wroom-1_wroom-1u_datasheet_en.pdf. [Accessed: Feb. 28, 2025]

[9] Espressif Systems, *ESP32-S3-DevKitC-1 User Guide*, 2024. [Online]. Available: https://docs.espressif.com/projects/esp-dev-kits/en/latest/esp32s3/esp32-s3-devkitc-1/user_guide.html#hardware-reference. [Accessed: 05-Mar-2025].

[10] SunFounder, *Vibration Sensor Module (SW-420)*, 2025. [Online]. Available: https://docs.sunfounder.com/projects/ultimate-sensor-kit/en/latest/components_basic/04-component_vibration.html. [Accessed: 05-Mar-2025].

Appendix A

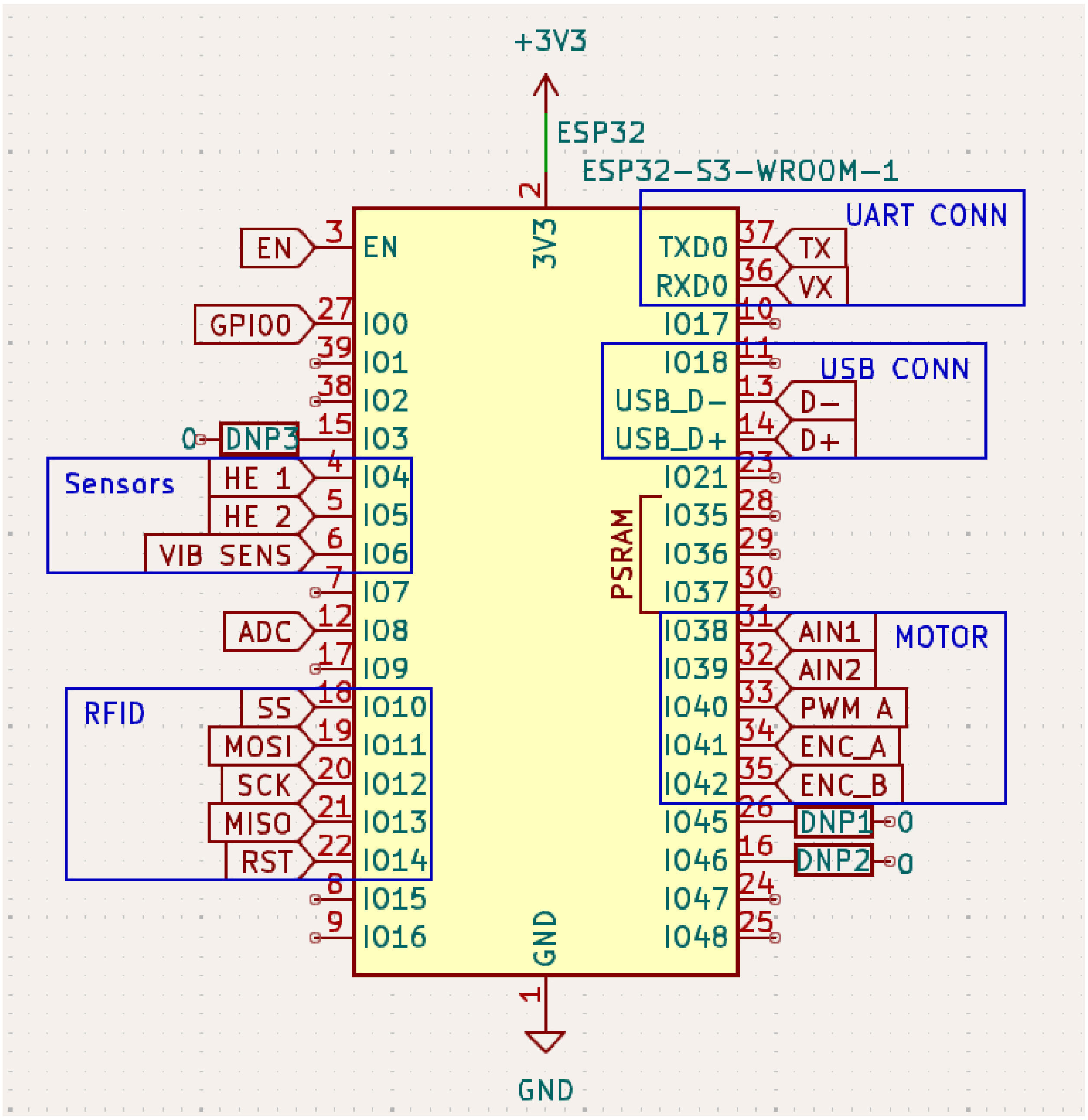


Figure 7: GPIO Connections to the ESP32

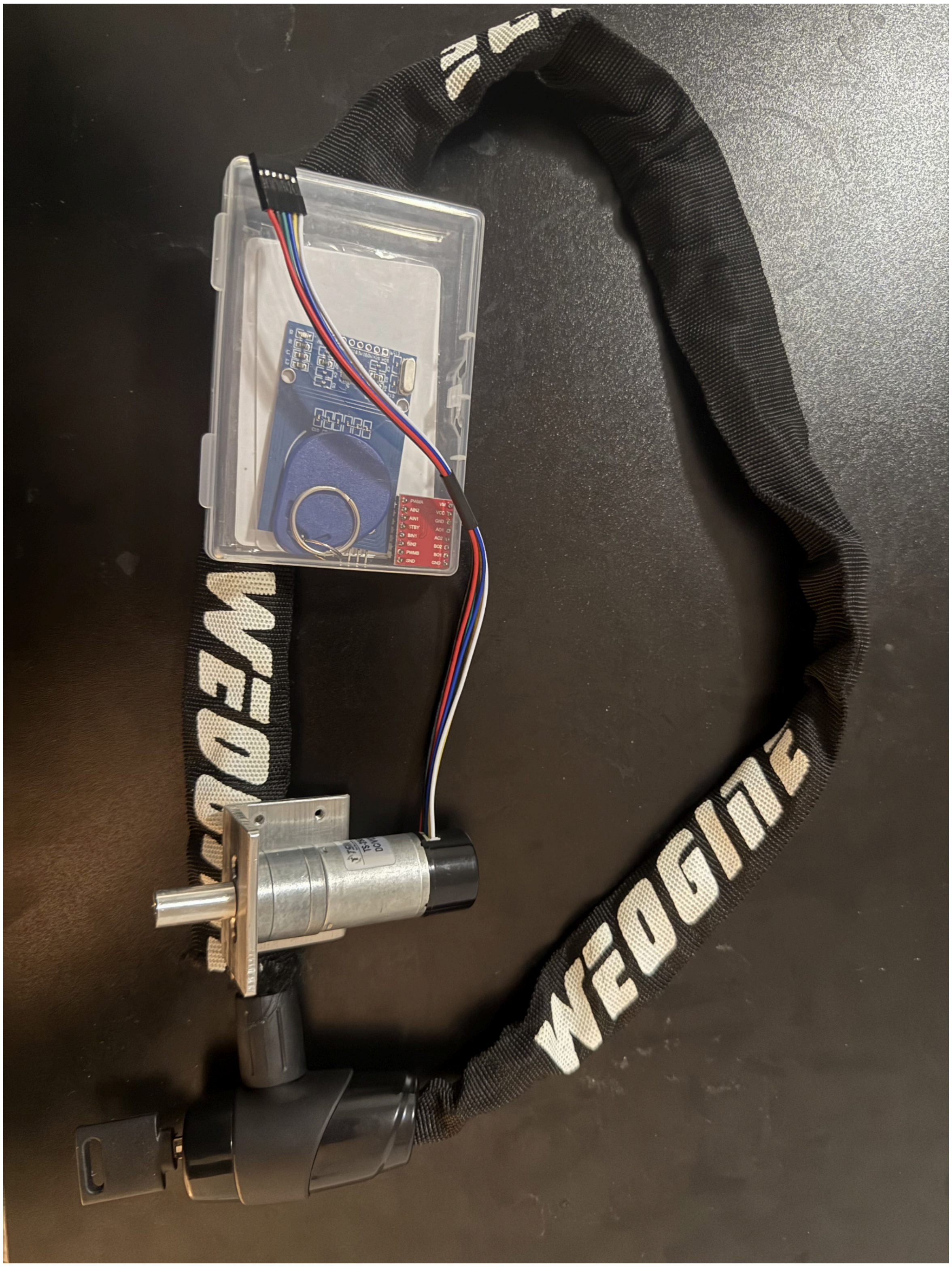


Figure 8: Real Lock Reference