AUTOMATIC BIKE LOCKING SYSTEM

By
Rohan Sreerama
Praneeth Mekapati
Cameron Obrecht

ECE 445
Senior Design Laboratory
Fall 2022
TA: Hanyin Shao

December 7, 2022
Team 19

Abstract

Bicycle theft is a prevalent issue that leads to millions of dollars worth of property loss every year. To date, the most common means of securing a bicycle is simply purchasing a strong lock. Although there have been numerous technological advances in lock design, there has been no effort to take the liability of securing one's bike off of the shoulder of the bike owner and onto some other entity. We propose an Automatic Bike Locking System (ABLS) with the core purpose of securing one's bike with effective security systems, thereby removing the liability of securing one's bike off of the bicycle owner. Affixed to the common bike rack, users who approach the ABLS will be able to set a passcode and pay a small price to lock their bike safely for a desired number of hours.

Contents

1 Introduction 4
1.1 Purpose & Function 4
1.2 Design 5
1.3 Performance Requirements 7
1.4 Block-Level Changes 8
2 Design 9
2.1 Design Procedure & Choices 9
2.1.1 Microcontroller 10
2.1.2 Display 10
2.1.3 Keypad 10
2.1.4 LEDs 10
2.1.5 Coin Slot 11
2.1.6 Accelerometer 11
2.1.7 Buzzer 11
2.1.8 Ultrasonic Sensor 11
2.1.9 12V NiMH Battery 11
2.1.10 Motor Driver 12
2.1.11 Motor 12
2.2 Challenges 12
2.2.1 Accelerometer 12
2.2.2 Printed Circuit Board 12
2.2.3 GPIO and I^2C 13
2.2.4 Timing Issues 13
2.2.5 Lack of Documentation and Support 13
2.2.6 Programming 14
2.3 Functions, Equations, & Data 14
2.4 Subsystem Requirement / Verification 17
3 Costs 18
3.1 Cost Analysis 18
3.2 Schedule 19
4 Ethics and Safety 19
5 Citations 21
Appendix A: Final PCB Design 22

1. Introduction

1.1 Purpose & Function

Bicycle theft is a major problem in the United States. It is estimated that 2 million bicycles are stolen annually, amounting to a value of $350,000,000.[1] A common practice among bicycle owners to deter theft is to buy one of many portable bicycle locks available on the market, and carry it around with them. This can be slightly cumbersome, and a sturdy metal U-lock costs at least $30 from Amazon, with quality Kryptonite® brand locks priced at more than twice that. Replacing a stolen low-end bicycle such as a Huffy costs at least around $150, and high-end bicycles and electric bicycles can reach well over $2,000.

As a solution to this problematic phenomenon and an alleviation of the costs associated with it, we have designed and built the Automatic Bike Locking System. This is an appliance that is supposed to be retrofitted by business or building owners onto an existing bicycle rack on their property, rather than integrated into some custom rack that comes with the appliance. It has a keypad, a display, a coin slot, and other elements for interaction with any bicycle owner who rides up to the building and wants to use the appliance to lock their bicycle. It is a rugged device, impossible to sever the lock with bolt cutters (as is a common vulnerability of personal locks) due to the thickness and position of the locking bar; and even a damaging blow to the keypad and user interface elements, or even a total loss of power, does not compromise the security of the device, because neither event will cause it to unlock and release the bicycle. When someone goes to use the appliance, a series of guiding messages are displayed on the alphanumeric screen as the user completes the steps for input and payment. The user specifies the desired duration of parking, and then is charged accordingly. The appliance actuates the locking motor after the last coin has been inserted. After the parking duration has passed, the appliance will keep the bicycle locked, rather than opening immediately (as an act of graciousness for less shrewd users), but will require additional overtime payment to release the bicycle. In the event that a thief attempts to tamper with the device, particularly by using an angle grinder in an attempt to destroy it, an ultrasonic proximity sensor will sense the person and an internal accelerometer will register the vibrations. The device will sound a loud buzzer to alert others nearby, similar to the alert noise produced by some rental scooters when tampered with or moved without payment.

1.2 Design

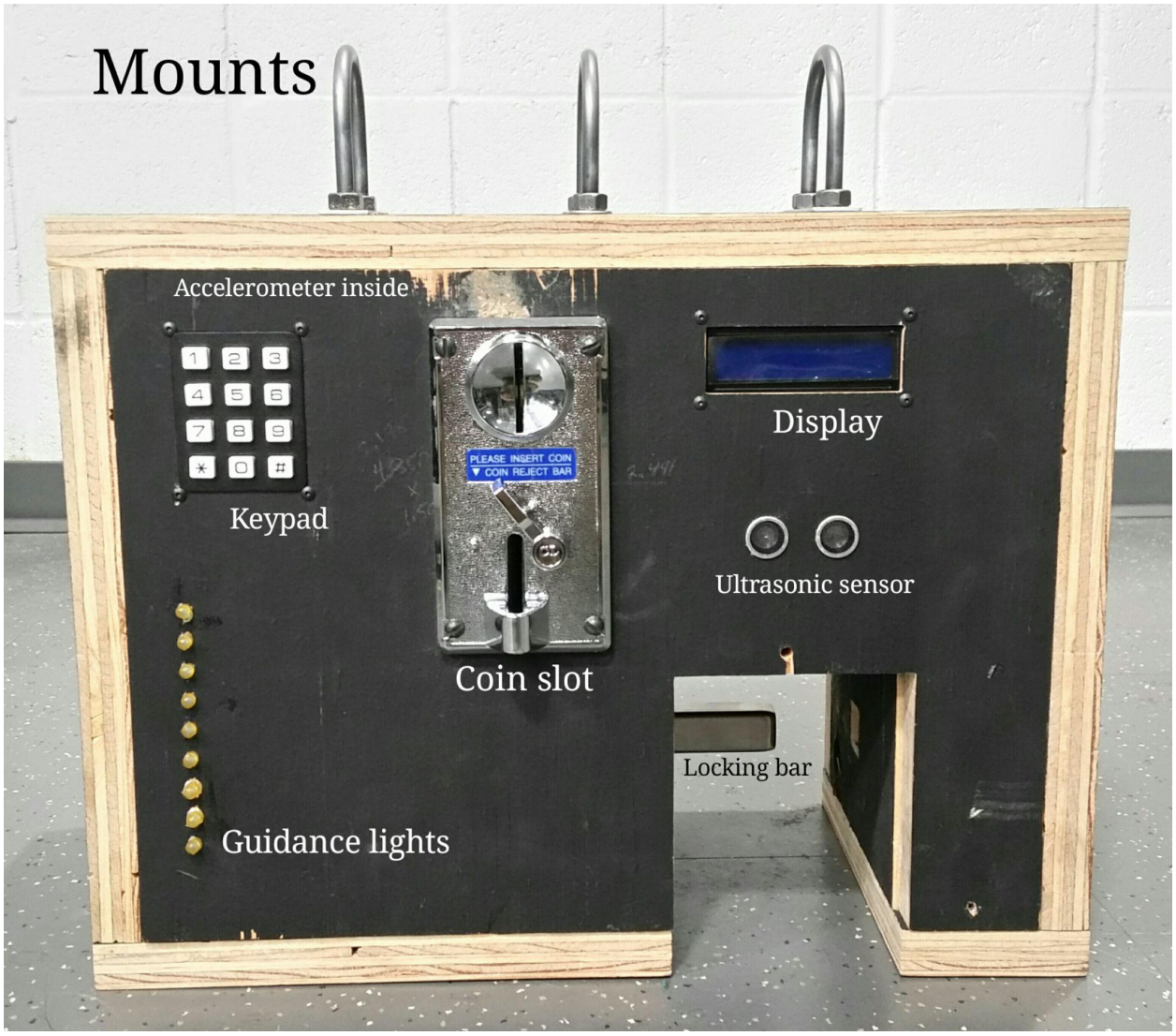


Figure 1: Photograph of the appliance, with interface elements labeled.

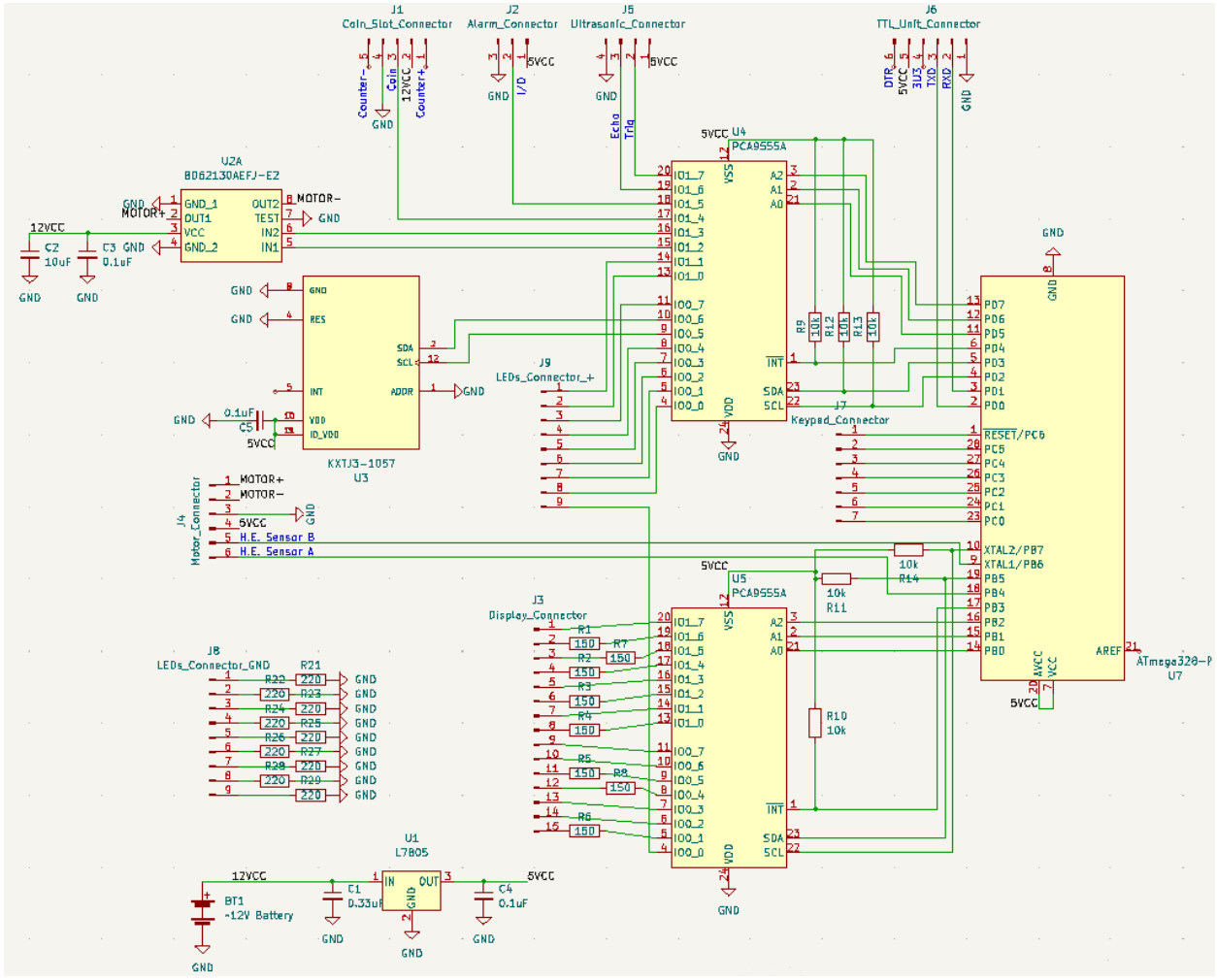


Figure 2: Original circuit schematic.

The design of our appliance consists of four subsystems, not completely distinct from each other in the design and in the way components interact, but which are distinct in their functions. These are the power, control, locking, and alarm subsystems.

- The aptly named power subsystem is responsible for supplying power to the appliance. The source of energy is a 12-volt nickel-metal-hydride battery, which powers the coin slot and the locking motor, which are both rated for 12 volts. This battery also connects to a 5-volt regulator, which powers the rest of the components. No regulator is necessary for the 12-volt components, since they can operate within a small range around 12 volts. There is no intelligent control of this subsystem, as the battery simply delivers power.
- The control subsystem is responsible for governing the behavior of the appliance. It entails the pair of ATmegas that were used in the final design, their I/O-expanding devices, and the user input and output elements, including the coin slot.

- The locking subsystem is simply the motor and its driver chip, which takes logic input from the main ATmega to spin the motor. The motor encoder and its output, had we used this available feature, would have comprised a part of this subsystem as well.
- The alarm subsystem is responsible for outputting a high pitched noise to the buzzer when the accelerometer detects movements breaching a certain threshold from the X, Y, or Z direction and when the ultrasonic sensor detects a nearby presence.

1.3 Performance Requirements

High Level Requirements:

A. Be secure
    a. The target system must be one that prioritizes safety and security of the bicycle at all times, ensuring it cannot be compromised. Intrusions need to be detected via the accelerometer and ultrasonic sensor. When a thief attempts to steal a bicycle by grinding or rocking the module frame of the ABLS, the accelerometer and ultrasonic sensor needs to identify irregular behavior in the measurement readings, and trigger the buzzer.
B. Accept payment honestly
    a. The target system must accurately process the input for the desired number of hours from the keypad device and use this value only to compute the cost. Once the user has entered the exact requested payment, the device should proceed to service the user appropriately.
C. Process inputs and display output correctly.
    a. All I/O transactions must be processed as the user would expect. For example, the desired number of hours, passcode, remaining balance should all be taken as input correctly from the I/O devices.
    b. All of this data should also be logged out to the user via the LCD display as the user goes through the interactions.

1.4 Block Level Changes

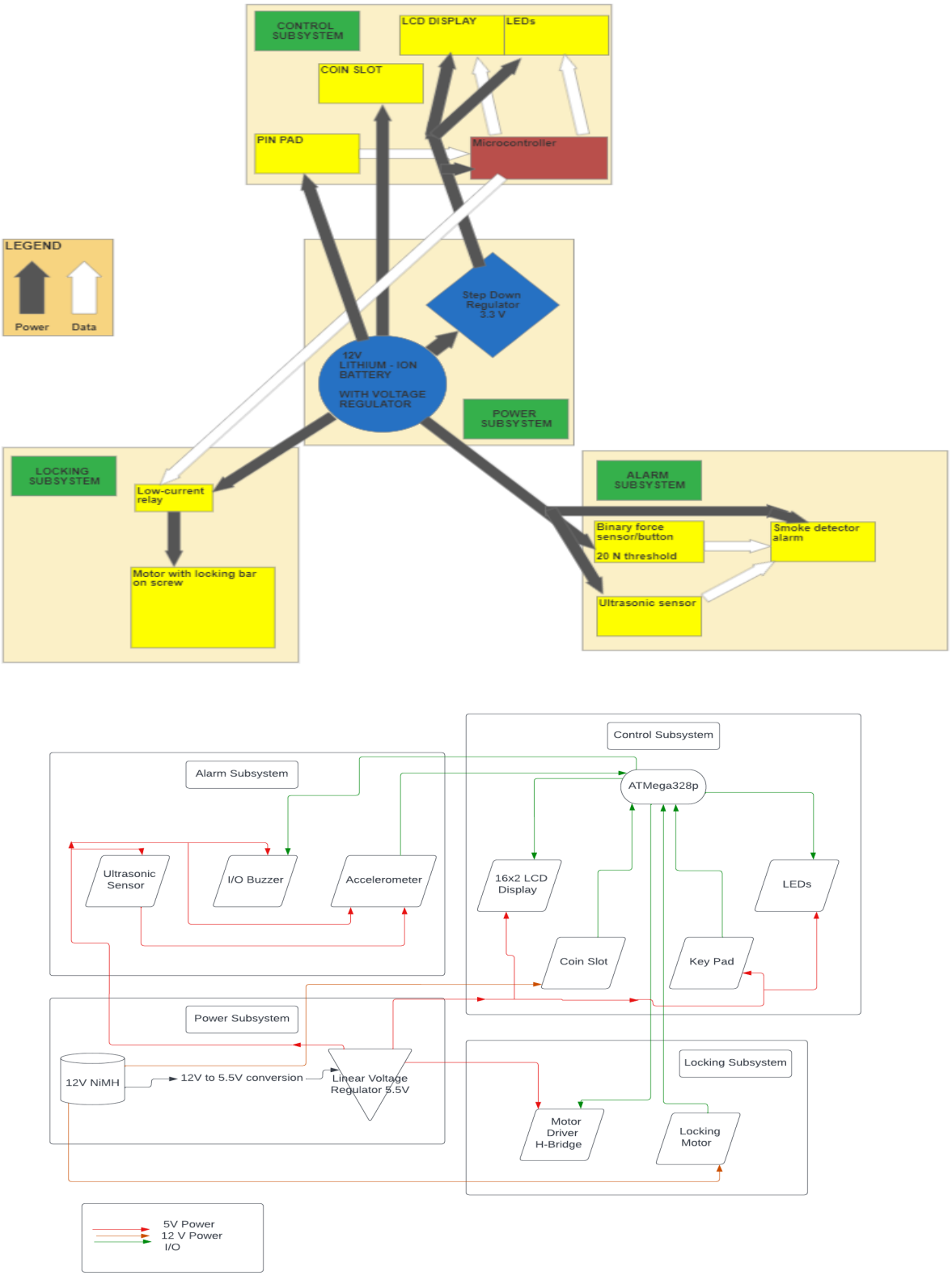


Figure 3: Original Block Diagram To Final Block Diagram

There are several changes that we made to the design, reflected in the Final Block Diagram.

We have replaced the batteries in the power subsystem with the 12V NiMH battery and the linear voltage regulator outputting 5V. We have replaced the 12V Lithium Ion battery with that as the NiMH had higher availability. The linear voltage regulator was able to properly convert the 12V from the NiMH battery to 5V and the 5V was needed to power the ATMega328P chip and the remaining components as the 3.3V regulator did not have enough voltage for most of the components.

We have replaced the 3x4 keypad with the 4x4 keypad as there are more buttons and we can utilize them for scenarios such as deleting numbers and entering the numbers. This would prove to be useful as the user would be able to backtrack if they have entered a wrong digit from their initial four digit pin and then press enter when they think the passcode is correct. We have utilized a 16x2 LCD Display in order to create better communication between the user and the appliance as it would print messages like “type the passcode”, “enter the number of hours”, and “enabling lock”. This also is able to lock the numbers for the passcode using asterisks. We replaced the relay with a H-Bridge motor driver as this will be able to control the speed and the direction as this is important in opening and closing the lock for the bike.

In the Alarm Subsystem, we have replaced the force sensor with the accelerometer as this would be able to easily detect any type of motion from the appliance in any part when a certain threshold is breached whereas the force sensor will only detect a force of 20N which is very high and this could create greater damage to appliance where the security won’t be able to function properly. These changes bettered the security system and provided an easier interaction between the user and the appliance in the process of locking the bike. In place of the microcontroller, we have utilized the standalone ATMega328p chip which was able to send arduino I/O signals to the electronic components by being powered by 5V. This was easy in connecting all of the components and integrating a condensed and interactive system.

2 Design

2.1 Design Procedure, Details, & Choices

The first thing we did in designing the circuit was to select components. Our criterion for a good component to use, whichever component it may have been, was that it was the most relevant search result of its kind on the DigiKey website. This is how we originally decided to use the PCA9555 GPIO expander, for example. The next step was to glean as much information as seemed necessary from the datasheets for each component, particularly the maximum operating voltages and proper routing of connections, and other things that most directly meant the distinction between proper functionality and failure of the component. After all of our internal questions and doubts about aspects of each component were resolved by enough reading and research, we dove into drawing up the schematic using KiCAD 6.0, and then making the PCB from it. With each change we were to make became evidently necessary, we changed the PCB design, and sometimes the schematic with it, although by this point the two were separate and it was not by any means worth the time to re-generate the PCB from an altered schematic with every revision. The changes we made after completing the first iteration of the PCB were mostly due to the evincement of flaws as we realized their presence, with the change of display being an

exception. The very end of the process saw us using unconventional and low-quality methods of attempting to rectify the issues that came about, such as jumper wires and rigs, because of a lack of time to do it properly.

2.1.1 Control Subsystem - The Microcontroller

There are a vast array of microcontrollers available on the market that help achieve the core functionalities our product requires. Among the options we considered were the ESP-32, ATTiny85, and even moving to a minicomputer like the Raspberry Pi. The ESP-32, while being robust and highly capable, had far more functionalities than we needed. For instance, we did not need Wi-Fi or Bluetooth connectivity, so we abandoned this option in the spirit of being cost-efficient. The Raspberry Pi was an early option we considered; however, this would take the footprint of an entire computer and cost far more as it has expensive components like RAM. We only considered using this so that we could program the device in a high level language, like Python, which would be easier to debug and utilize. The ATTiny85 was also a great option for us as it meant we could utilize the Arduino as a debugger circuit and have our code loaded onto a bootloader chip. However, this had a low amount of pins for I/O, which our design really needed.

As a result, we opted for the ATMEGA328p which is a microcontroller that comes pre-fitted on standard Arduino UNO boards. This microcontroller could be easily programmed via the Arduino language, which is similar to C/C++. Moreover, there is copious support online for interfacing with the devices we used via Arduino. Finally, it has a small footprint and could be boot-loaded to run code on startup, which is a very useful feature to our design.

2.1.2 Control Subsystem - The Display

Initially, we had selected a 7-segment numeric display that we intended for displaying numeric information like the number of hours and passcode. While this was quite simple to integrate into our system, it required reasonable effort to write software that displays numbers as we needed it to. Because this display could only show numbers, we also had to add LED indicators that we planned to use as guidance indicators for when an action is complete (ie. password is set, hours are confirmed, locking is complete, etc.). We later realized that an LCD display is far superior as Arduino had robust libraries that we could leverage to display all kinds of information. By using this, we were able to report numeric and guidance information all on the LCD itself.

2.1.3 Control Subsystem - The Keypad

The Keypad we had initially chosen was one with good tactile response and numeric functionality that would work well with our 7-segment display. However, we wanted to ensure our final design was robust and customizable, so we went with a different keypad model that offered alphanumeric input. From a technical standpoint, this also registered values more quickly and accurately.

2.1.4 Control Subsystem - The LEDs

LEDs were chosen as a means of guiding the user through the interactions necessary to use the ABLS. We planned to place stickers next to each LED that identified stages in the user flow such as “enter coins,” “enter passcode,” “locking complete,” etc. LEDs would glow as the user proceeded through the interactions. (This was eventually rendered unnecessary as we used the LCD.)

2.1.5 Control Subsystem - Coin Slot

Our device required a means of payment and we desired something that was easily configurable and customizable to our needs. Because coin slots have a lot of precedent and require no Internet, we decided to use it. We could have opted for a credit card reader or some other form of electronic payment; however, we foresaw potential challenges in debugging a Wi-Fi driven system. Given the already complex design posed by the number of I/O devices we have, we decide the coin slot would be a perfect choice.

2.1.6 Alarm Subsystem - Accelerometer

One of the primary value propositions of the ABLS is its ability to protect and safeguard a user's bicycle. The simplest way to compromise our lock would be to simply grind the bar that goes across the bike. We initially proposed a solution where we would utilize force probes to detect if a user was applying a force to the bar that is greater than a set threshold value (ie. 20 N). After some revisions, we determined this would not be very effective because the force sensor would only detect applied force on the lock. A thief could easily realize this and instead opt to steal the entire device, thereby bypassing the force sensor. Additionally, the force sensor would only detect an irregular force value after damage has been done to the locking mechanism. We could catch this act much earlier and more effectively with the use of an accelerometer. An accelerometer would be installed on the inner part of the box, so if any irregular movements in the x, y, or z direction are detected, the device has a clear cause to sound the alarm. To be clear, the alarm will ring only when the accelerometer's readings are above our pre-set threshold and also when the distance to the device is too close as determined by the ultrasonic sensor's threshold value. Moreover, any vibrations generated by grinding on the locking bar would be easily registered by the accelerometer and flagged as inappropriate.

2.1.7 Alarm Subsystem - I/O Buzzer

The ABLS design requires a means of sounding an alarm when an intrusion is detected. We initially thought of using a smoke detector's alarm component as it is known to be loud and obtrusive. However, we later came across the digital buzzer device, an easily programmable Arduino-interfaceable device.

2.1.8 Alarm System - The Ultrasonic Sensor

Simply put, we required a means of confirming an intrusion is detected once the accelerometer flags inappropriate handling of the device. The ultrasonic sensor is a great device to detect distance for this purpose, and functions quite simply so we moved to use it.

2.1.9 Power Subsystem - 12 V NiMH Battery

We chose a 12-volt battery because 12 V was the highest voltage needed anywhere in the circuit. We were aware that lithium-ion technology is contemporarily much more common and preferred across many industries and applications than the older nickel-metal-hydride variety of rechargeable batteries, but nickel-metal-hydride is still an appropriate power source in some hobbyist remote-controlled cars, which have motors similar in size to ours and are designed to run continuously for a half hour or more on one charge. We knew our motor would only run extremely intermittently, and the

battery we chose also has the advantage of being a single packaged unit, rather than several independent cells. We also already had a NiMH charger available to use, whereas lithium-ion would have required the purchase of a charger.

2.1.10 Locking Subsystem - Motor Driver

The first motor driver we chose was based on its search relevance on the DigiKey website. We later changed to a different model, however, because of its greater availability (such as on Amazon, qualifying it for quicker delivery) and because of its breadboard-compatible footprint. It had the exact same functionality as our first motor driver, so changing to the different model allowed more versatility.

2.1.11 Locking Subystem - Locking Motor

We required a powerful motor capable of driving the lock in and out at a reasonably fast speed. To that end, we opted for the YC2010-22 12-V geared motor which operated smoothly and efficiently, with minimal setup difficulty.

2.2 Challenges

Throughout the development of this product, we experienced numerous setbacks and shortcomings, which will be detailed below:

2.2.1 Accelerometer

Part acquisition, in the beginning, was a rudimentary process of identifying what functionality we needed and then proceeding to order a part that achieved this functionality. As we later found out, the accelerometer we had purchased (KXTJ3-1057) had a voltage incompatibility with our design. The voltage was far too high, leading to a bad output from the device. This situation was remedied by purchasing the ADXL335, a different accelerometer that proved to be far more effective in outputting the appropriate x, y, and z accelerometer values.

An additional setback with the accelerometer we did not quite realize initially was the fact that it is an analog device, which meant we could only programmatically interface with it via the analogWrite() and analogRead() functions in Arduino. This was not ideal because in our PCB Design, we initially planned to route the accelerometer via the GPIO expander to conserve pins on the ATMEGA328p microcontroller. The GPIO we utilized is only meant to interface with digital devices. As such, we remedied this situation by directly connecting the accelerometer to the ATMEGA328p.

2.2.2 Printed Circuit Board

The PCB was meticulously designed and redesigned (re-ordered 4 times) to ensure the traces were routed correctly in order to interface with our devices as expected. Unfortunately, due to mismatched footprints and traces that we did not originally know were incorrect, our PCB design did not work as expected. Some of its shortcomings were discovered too late in the process to wait for delivery of a revised PCB order, so we decided to solder jumper wires and bypass incorrect traces with some creative wiring, as our best attempt to make it functional given the time and resources that we had. The

two most significant instances of the need to bypass PCB traces were due to unfamiliarity with the I^2C output pins on the ATmega328P, and with the analog nature of the ADXL335 accelerometer we used. We had assumed that any two I/O pins on the ATmega could be programmed to drive an I^2C bus, and we did not see anything in the datasheet to suggest otherwise. But upon discovering an online tutorial on the subject, and after further investigation, we learned that it is specifically pins 27 and 28 on the chip that are meant for the SDA and SCL I^2C outputs. To rectify this, we meticulously cut and stripped 28 wires and soldered them to an ATmega socket, crossing some of them to reroute pins on the socket to different pads on the PCB, so that whatever originally was routed to the SDA and SCL pins would simply use the I/O pins that the I^2C originally used. With the accelerometer, we found that only the analog pins on the ATmega could be used, but this was also too late in the process to order another PCB. So we soldered some jumper wires to another free-hanging ATmega socket, with the the other ends of the wires soldered to the proper pads on the PCB. There were a couple of other less significant rerouting demands, but these were only connections from the coin slot and ultrasonic sensor to some unused I/O pads on the main ATmega, to bypass the GPIO expanders to which they were originally connected.

2.2.3 GPIO and I^2C

The ABLS utilizes a number of peripheral I/O devices to process input and even more devices to ensure the security of the bicycle, as mentioned before. In order to control all of these with one microcontroller, we realized early that more pins would be needed, which is why we opted for the GPIO expander. As we quickly found out, the model we purchased - the PCA9555 expander - did not have a lot of documentation of software support online. As a result, we could not source an entirely functional code library that enables interfacing with this expander.

We moved to use two different GPIO expanders - the PCA8574 for the LCD and the PCA8575 for the motor driver and ultrasonic sensor. These expanders had far more support from the Arduino and open-source community, allowing us to find adequate code libraries to assist us in using them.

2.2.4 Timing Issues

When we wired the entire system to interface with only one ATMEGA328p microcontroller, we began to notice timing issues. From a macro perspective, it appeared that data was being relayed potentially abnormally as output devices like the buzzer were acting in a delayed manner. For example, when the ABLS device was shaken so as to trigger the buzzer, it would take 30s for the system to acknowledge this interaction and then initiate the buzzer. Of course, we could have sought to mitigate this by composing more robust software; however, at this point we had proceeded to utilize a breadboard for the demonstration. As a result, we decided to utilize two ATMEGA328p chips which would allow us to separate the security system from the control systems, which dramatically eliminated the timing issues.

2.2.5 Lack of Documentation and Support

For some areas of our design, we had purchased devices that did not necessarily have thorough documentation. For instance, the coin slot had next to no documentation, which led to us spending a few days figuring out the optimal voltage and configuration necessary to enable its use. This was the same situation with the keypad device.

2.2.6 Programming

The majority of this project was hardware-based, as many of the challenges were also hardware-driven. We had unwittingly assumed that once we successfully assembled the product, the code would function as expected. However, Arduino code is low-level code that interfaces directly with our hardware; as a result, there were clear timing issues that persisted and we needed to spend time debugging these problems. Moreover, we had to experiment with different device models when we could not find supported software libraries and then learn to work with the new libraries, which also was a major time sink.

The important subsystems of this project are the alarm subsystem and the control subsystem. Determining the distance of a person from the ultrasonic sensor and the position and pressure of the appliance are the key to sensing any intrusions. In the control subsystem the coin acceptor was responsible for helping to activate the motor lock. In order to understand these components, we also had to understand the calculations related to each of the components.

2.3 Formulas, Equations, & Data

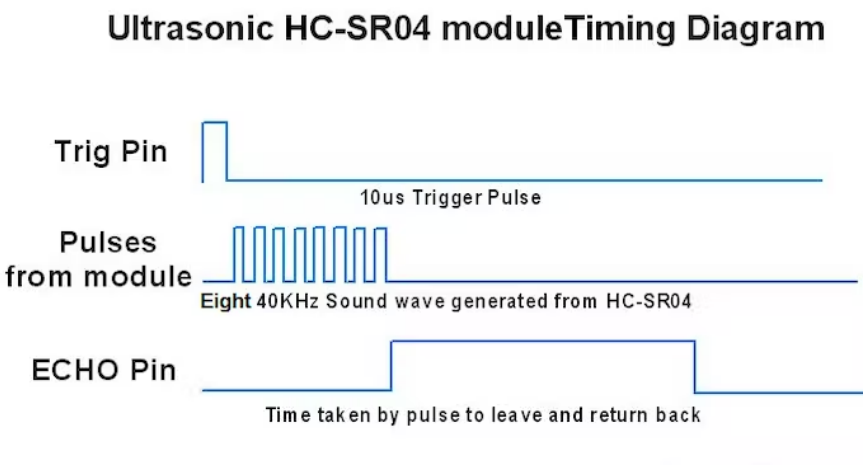


Figure 4: Ultrasonic Timing Diagram

Figure 4 from the Ultrasonic Sensor HC-SR04 with The Working Principle of Ultrasonic Sensors by Reese exemplifies the activation of the Ultrasonic sensor. The trigger pin is set to high for the time of 10 microseconds. This will in turn send an electronic pulse consisting of eight cycles. The electronic pulse will travel at the speed of sound and the echo pin will receive this pulse. The pulse input will be taken from the echo pin which will output the overall time in microseconds. The calculated time will be integral for retrieving the distance of an object or a person near the ultrasonic sensor. The equation, *Distance = Time * Speed / 2* where Speed is the value of 340 m/s, calculates the distance in centimeters. Here, the soundwave is bouncing back and forth between the sensor and the entity. We have set a threshold value of 150 cm in order to increase the range of finding the person. If the threshold is breached, the buzzer won't sound right away as there is a need to make sure the appliance is in the right position. The ultrasonic sensor is responsible for detecting any movements or changes in position of the appliance with the use of the equation *Change in Position = Final Position - Initial Position*. This is applied to all three of the sensor dimensions X, Y, and Z. With this information, we have set the change condition to be no greater

than the absolute value of 10 cm. If this threshold is breached along with the ultrasonic sensor's threshold, then the buzzer will be outputting frequent high pitched sounds until there is no more movement detected.

```
17:51:36.229 -> x=0
17:51:36.229 -> y=13
17:51:36.259 -> z=-6
17:51:36.259 ->  $
17:51:36.259 -> Distance: 9
17:51:38.501 -> x=10
17:51:38.501 -> y=13
17:51:38.501 -> z=2
17:51:38.534 ->  $
17:51:38.534 -> Distance: 9
17:51:43.212 -> x=1
17:51:43.212 -> y=-11
17:51:43.212 -> z=4
17:51:43.212 ->  $
17:51:43.212 -> Distance: 9
17:51:46.478 -> x=-2
17:51:46.478 -> y=-11
17:51:46.511 -> z=5
17:51:46.511 ->  $
17:51:46.511 -> Distance: 10
17:51:53.302 -> x=0
17:51:53.302 -> y=12
17:51:53.335 -> z=-6
17:51:53.335 ->  $
17:51:53.335 -> Distance: 10
17:51:56.533 -> x=-3
17:51:56.533 -> y=2
17:51:56.566 -> z=-11
```

```
17:52:27.451 -> y=-3
17:52:27.451 -> z=2
17:52:27.451 ->  $
17:52:27.451 -> Distance: 10
17:52:34.705 -> x=30
17:52:34.705 -> y=67
17:52:34.738 -> z=2
17:52:34.738 ->  $
17:52:34.738 -> Distance: 72
17:52:42.352 -> x=1
17:52:42.352 -> y=47
17:52:42.352 -> z=-12
17:52:42.352 ->  $
17:52:42.352 -> Distance: 11
17:52:44.365 -> x=12
17:52:44.365 -> y=35
17:52:44.365 -> z=-11
17:52:44.397 ->  $
17:52:44.397 -> Distance: 10
17:52:47.332 -> x=-3
17:52:47.364 -> y=-12
17:52:47.364 -> z=5
17:52:47.364 ->  $
17:52:47.364 -> Distance: 45
17:52:49.571 -> x=0
17:52:49.571 -> y=32
17:52:49.571 -> z=-9
17:52:49.604 ->  $
17:52:49.604 -> Distance: 83
```

Figure 5: Alarm Subsystem Data

Figure 5 displays the numerical data of the combination of the ultrasonic sensor and the accelerometer. This is the data that is collected whenever the thresholds are both breached. This was calculated with the use of creating an AND function where the ultrasonic sensor detects someone within the 150 cm range and there is movement of 10 cm or greater for any of the three dimensions. For example, there is a distance for x at 0 cm, y at 12 cm, and z at -6 cm. This will sound the buzzer because one of the dimensions which is the y is above the 10 cm threshold.

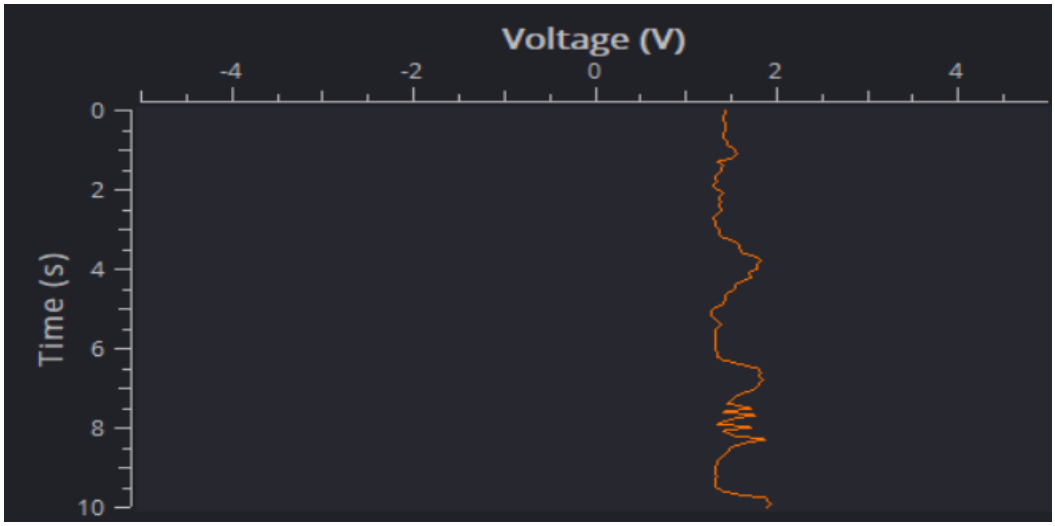


Figure 6: Voltage Output from Accelerometer

The graph from Figure 6 shows voltages either increasing or decreasing based on the movement of the accelerometer. This depends on the speed of the accelerometer. If it changes position quickly, then there would be greater voltage changes from the initial voltage and this would in turn be able to quickly alert the microcontroller chip of the movement of the appliance.

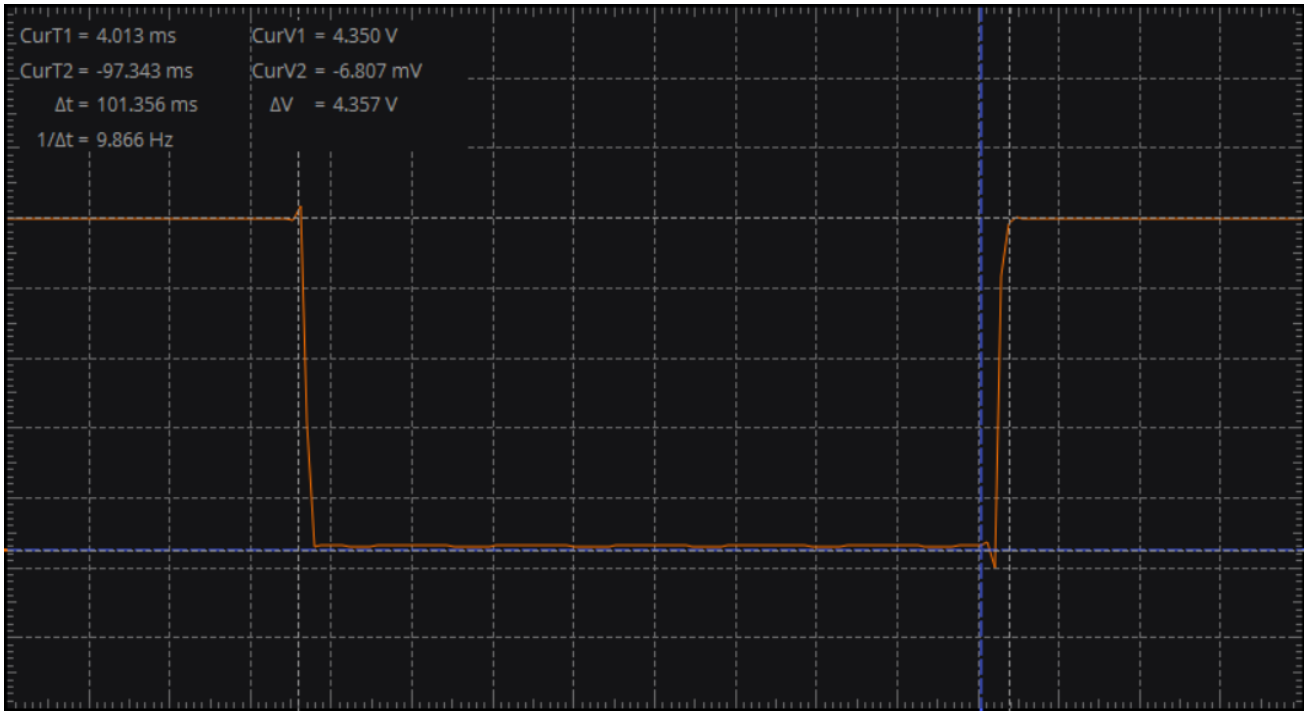


Figure 7: Coin Slot Graph

Figure 7 is able to detect a coin slot interrupt by recording the voltage drop. The voltage drop of 4 V then sends a signal to pin 2 of the ATmega328P chip. this would later output this fall to the LCD in the form of balance that would increase each time a fall is detected. After a certain balance value is reached that is proportional to the inputted number of hours, this would send an output signal to the motor to start the locking process with the use of the HBridge motor driver chip.

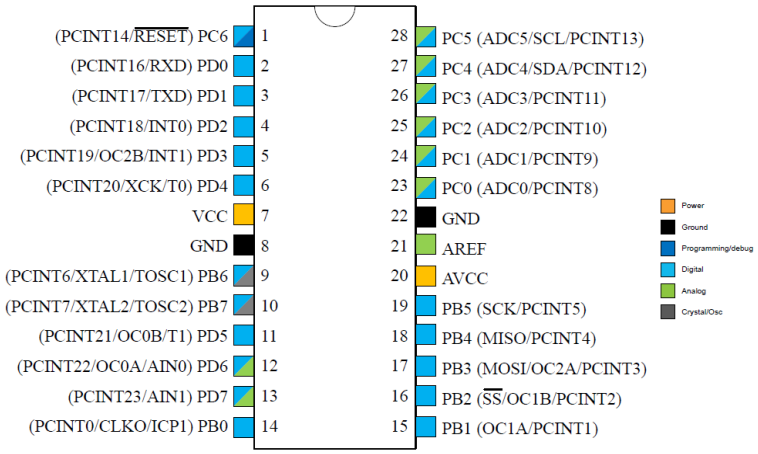


Figure 8: ATMega328P Pinout

The pinout of the ATMega shows us the pins that are utilized as digital and analog inputs/outputs and pins that are required for uploading the updated arduino code which are Pin 0 for Reset, Pin 1 for RX, and Pin 2 for TX. Along with that, it is important that both sides of the chip are powered by 5V for the chip to send the I/O signals for both the digital and analog pins. This consumes 26 uA with the use of 5V.

2.4 Requirements & Verifications

Requirements	Verification
Lock Subsystem: 1. The motor should move the bar to the right after the correct amount of money is inserted 2. The motor should move the bar back to its original position after being locked an inputted period of time in hours or the user is able to enter the right passcode	1. The LCD 16x2 Display should output “Engaging Lock” right after the correct amount of money is deposited in quarters 2. If successful, the H-Bridge motor driver sends the signal to move the bar towards the right direction for the locking.
Alarm Subsystem: 1. Sounds a high pitched noise whenever any unexpected changes in the position of the appliance occurs and there is a presence sensed near the appliance 2. Stays quiet if there is no presence nearby or there is no unexpected movement of the appliance	1. In order to verify the Ultrasonic sensor is alerted, we blink a colored LED indicating a nearby presence within the distance of 150 cm or less 2. The Accelerometer is alerted whenever there are changes within the X, Y, or Z direction breaching the threshold change of 10 cm. This will also blink a colored LED. 3. When both of these components are alerted at the same time, the buzzer will sound a high-pitched noise.
Control Subsystem:	

1. LCD displays outputs based on the User Inputs for passcode and number of hours 2. Atmega chip sends a signal to the motor to lock after “engaging lock” is displayed 3. Chip records coins accepted to the LCD display	1. Updates balance in serial monitor by adding 25 cents after inserting a quarter 2. Motor starts locking after the balance is proportionate to the number of inputted hours and the motor turns to the right 3. Motor unlocks after the inputted time has passed in microseconds and the process begins again asking for a passcode

3 Costs & Schedule

3.1 Costs

Name	Manufacturer	Cost	Link
Arduino Uno ATMega328P	Arduino	$17.89	link
I/O Buzzer	Ximimark	$6.59	link
4x4 KeyPad	Adafruit	$5.00	link
YC2010-22 DC Motor	HZMagnet	$18.79	link
Ultrasonic Sensor	Arduino	$6.99	link
LCD Display 2x16	Adafruit	$10.99	link
Accelerometer	Adafruit	$14.89	link
Coin Slot	N/A	$20.85	link
12V Battery	Arduino	$10.21	link
Linear Regulator	Arduino	$11.79	link
Motor H-Bridge	Adafruit	$15.50	link
GPIO Expander	Adafruit	$9.00	link

Components Total Cost: $148.49
Labor Cost: $45.00 * 2.5 hrs / day * 60 days = $6,750
$7875 * 3 = $20,250
Total Costs overall: $20,399

3.2 Schedule

Week	Praneeth	Rohan	Cameron
1	Start on component analysis	Start on researching code for the components	Start on pinouts for components
2	Create the component schematic	Research PCB footprints	Research PCB footprints
3	Create the PCB based on the schematic	Research online for components	Consult with the machine shop of our overall design
4	Finalize components	Finalize components	Finalize components
5	Consult with Professors and Teaching Assistants of our components' functionalities	Consult with Professors and Teaching Assistants of our components' functionalities	Consult with Professors and Teaching Assistants of our components' functionalities
6	testing coin slot	testing Keypad	testing Motor Driver
7	Testing coin slot	Testing LCD	Testing DC Motor
8	Testing Atmega328p	Testing Atmega328p	Testing Atmega328p
9	Combine components on PCB	Combine components on PCB	Combine components on PCB
10	Test PCB	Test PCB	Test PCB
11	FInalize components on breadboard and test	Write code for the components	Start wiring the components with resistors and capacitors

4 Conclusions

Overall, our entire design functioned successfully, enabling a user to easily lock their bicycle and be ensured of its safety. Our control, locking, and power subsystems achieved their core functionality of ensuring a user can input a number of hours, a passcode, enter payment, and have the motor engage the locking mechanism. Furthermore, our security system can properly detect an intrusion and trigger a buzzer to alarm when this occurs.

Uncertainties

Although our security system is fully functional; at times, there can be a delay in its response when an intrusion is detected. To mitigate this, we would likely work on improving our software to deal with timing issues and consider using hardware components with better response time and reliability.

Future Work

Our team fully intends to create a new iteration of this project where we utilize better components and design. In the near future, we plan to make the following upgrades:

- Utilize an ESP-32 microcontroller so we can have Wi-Fi connectivity.
- Install a credit card reader that would have to interface with the Internet to process transactions.
- Create a new device with a much smaller footprint.
- Install a touchscreen instead of the LCD. This would offer much smoother user experiences.

Broad Impact

On a broader scale, we believe that our project has the potential to mitigate millions of dollars worth of property loss and damage to bicycles around the world. By taking on the liability of protecting people's bicycles, we will be providing an essential service that can safeguard bicycles at a reasonable price.

Safety

The ABLS device has clear measures enabled to ensure the safety of our product when in use. The IEEE Code of Ethics states that we must prioritize safety and the welfare of the public. The only moving part in our product is the motor, which causes the screw-driven locking bar to move across a cavity that locks the bike in place. Because it can be loud and dangerous, we display a number of key guiding messages to the user on the LCD to ensure their safety. When the system is ready to lock the bike, it prints to the LCD the following messages: "Stand back," "Engaging Lock," "Locking" and finally "Locking Complete." There are reasonable time delays between these messages so that any user can appropriately react and step back while the motor moves the locking mechanism. Also, no voltages used in the appliance are large enough to cause harm.

Acknowledgements

This was a very detail-oriented project that required a lot of research and meticulous planning. Our TA, Hanyin Shao, was integral to helping us plan everything and stay on top of schedule. Moreover, she offered relevant advice that helped us design and implement the ABLS, so we'd like to thank her for her efforts. We'd also like to thank Prof. Fang who gave us honest feedback about our project and helped drive innovation. Finally, we'd like to thank the ECE 445 course staff for providing a hub of creativity where we have access to a world-class lab and equipment to design our products.

5. Works Cited

[1] “The Ultimate Guide to Bicycle Theft Statistics & Insurance.” Simple Bike Insurance, Biker, 2 Oct. 2021, https://simplebikeinsurance.com/guide-bicycle-theft-insurance/.

[2] “EcoTank Et-2803 Wireless Color All-in-One Cartridge-Free Supertank Printer with Scan and Copy.” *C11CJ66203 | EcoTank ET-2803 Wireless Color All-in-One Cartridge-Free Supertank Printer with Scan and Copy | Inkjet | Printers | For Work | Epson US*, Epson America, Inc., 2022, https://epson.com/For-Work/Printers/Inkjet/EcoTank-ET-2803-Wireless-Color-All-in-One-Cartridge-Free-Supertank-Printer-with-Scan-and-Copy/p/C11CJ66203.

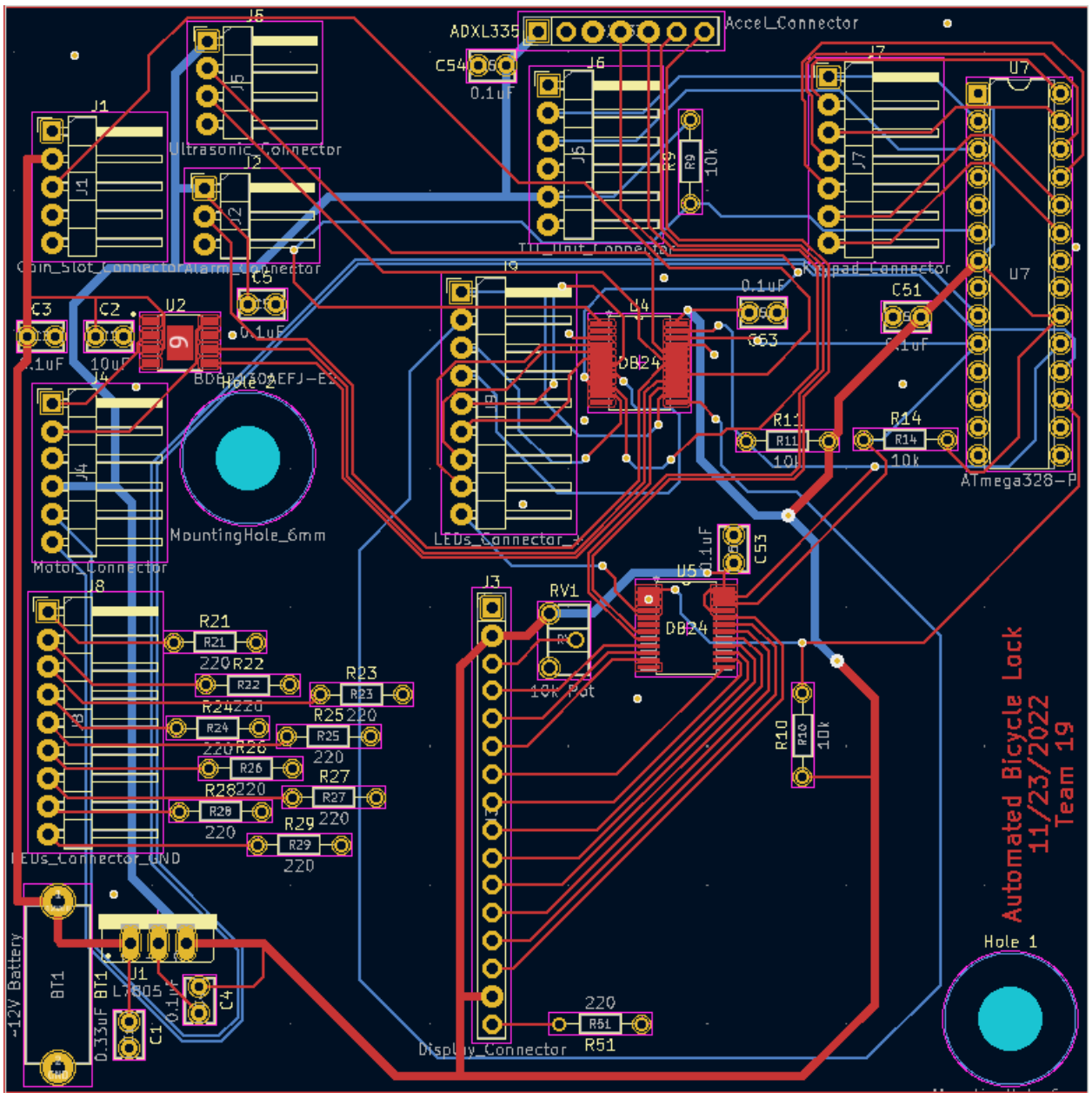
[3] “ATmega328/p.” Atmel Corporation, Jun. 2016, https://datasheet.octopart.com/ATMEGA328P-MU-Microchip-datasheet-65729177.pdf

[4] “PCF8575 I2C 16-Bit Digital Input Output Expander.” Create.arduino.cc, Project Hub, 7 Aug. 2019, https://create.arduino.cc/projecthub/xreef/pcf8575-i2c-16-bit-digital-input-output-expander-48a7c6

[5] “The Working Principle, Applications, and Limitations of Ultrasonic Sensors”, Microcontrollertips.com, Reese, 2019, https://www.microcontrollertips.com/principle-applications-limitations-ultrasonic-sensors-faq/

[6] “IEEE Code of Ethics”, IEEE, 2022, https://www.ieee.org/about/corporate/governance/p7-8.html

Appendix A: Final PCB Design



Ground planes have been removed for better clarity. The circuit schematic was changed a few times to reflect changes in the PCB, but eventually we found it sufficient to keep only a mental record of changes, aside from the PCB file itself which was also a good means of accounting for changes.