SMART STAIR GATE

By

Alexander Chin

Brandon Lau

Zeyad Irsheid

Final Report for ECE 445, Senior Design, Fall 2023

TA: Zicheng Ma

6 December 2023

Project No. 37

Abstract
Smart Stair Gate is a safety barrier in which prevents crawling babies from falling down a staircase. As the name suggests, Smart Stair Gate opens when a person or an animal approach, but remain closed when a crawling baby approaches. We propose using the distance and speed of an animate object to determine how project operates. This paper documents the design process, implementation, testing, and the construction of Smart Stair Gate.

Table of Contents

1 Introduction 1

1.1 Problem 1

1.2 Solution 1

1.2.1 Visual Aid 2

1.3 High-level Requirements 3

1.4 High Level Overview 4

1.4.1 Block Diagram 4

1.5 Subsystem Overview 5

1.5.1 Power Subsystem 5

1.5.2 Control Subsystem 5

1.5.3 Detection Subsystem 5

1.5.4 Prevention Subsystem 5

1.5.5 Application Subsystem 5

2 Design 6

2.1 Power Subsystem 6

2.1.1 Introduction 6

2.1.2 120VAC to 12VDC 6

2.1.3 12VDC to 5VDC 6

2.1.4 5VDC to 3.3VDC 7

2.1.5 Power Subsystem Verification 7

2.2 Control Subsystem 8

2.2.1 Control Subsystem Design 8

2.2.2 Control Subsystem Design (PCB) 8

2.2.3 Control Subsystem Verification 9

2.3 Detection Subsystem 9

2.3.1 Detection Subsystem Design 9

2.3.2 Detection Subsystem Verification 10

2.4 Prevention Subsystem 11

2.4.1 Prevention Subsystem Design (Mechanical) 11

2.4.2 Prevention Subsystem Design (Electrical) 11

2.4.3 Prevention Subsystem Verification 12

2.5 Application Subsystem ... 12
2.5.1 Application Subsystem Design ... 12
2.5.2 Application Subsystem Verification ... 12
4 Cost and Schedule ... 13
4.1 Cost Analysis ... 13
4.1.1 Parts ... 13
4.1.2 Labor ... 14
4.1.3 Total Cost ... 14
4.2 Schedule ... 14
5 Conclusion ... 15
5.1 Accomplishments ... 15
5.2 Challenges ... 15
5.3 Ethical considerations ... 15
5.4 Future work ... 16
6 References ... 17
Appendix A Requirement and Verification Table ... 18
A.1 Power Subsystem ... 18
A.2 Control Subsystem ... 19
A.3 Detection Subsystem ... 20
A.4 Prevention Subsystem ... 21
A.5 Application Subsystem ... 22
Appendix B Schedule ... 23
Appendix C Smart Stair Gate Logic ... 24
Appendix D Application Interface ... 25
Appendix E Smart Stair Gate Pictures ... 26
Appendix F Max Current Draw ... 26
Appendix G PCB Design ... 27
Appendix H Mechanical Drawing ... 28
Appendix I Circuits ... 28

1 Introduction

1.1 Problem

In today's society, parents have more things than ever to worry about especially parents of crawling children. Children are unpredictable at times especially when unsupervised, and the last thing parents want is their child to be doing something dangerous. One of the most dangerous activities a baby can do unsupervised is crawl down a flight of stairs. According to researchers, "Every six minutes, a child falls down stairs somewhere in the U.S., experts say. Over 90,000 kids under the age of 5 end up in emergency rooms because of stair-related falls every year" **[1]**. Although manual gates exist to solve this issue, parents may still forget to close the gate behind them leaving children susceptible to potentially life-threatening accidents.

1.2 Solution

Our solution provides peace of mind for parents of crawling children. We propose to create a smart gate enclosed at the top of a flight of stairs. The smart gate will be initially closed, and open if an adult or an animal approaches the gate.

This solution contains two main sub-systems: detecting when a baby approaches the gate and preventing the baby from crawling down a flight of stairs. Our detection system will be integrated into the gate containing a PIR sensor, a proximity sensor, and RF sensors to determine whether or not a crawling baby is approaching the gate. Our prevention subsystem will contain a motorized gate that will prohibit a crawling baby from crawling down a flight of stairs. We also plan to include a wireless system that enables parents to remotely monitor and control the gate's status, allowing them to open and close the gate.

1.2.1 Visual Aid

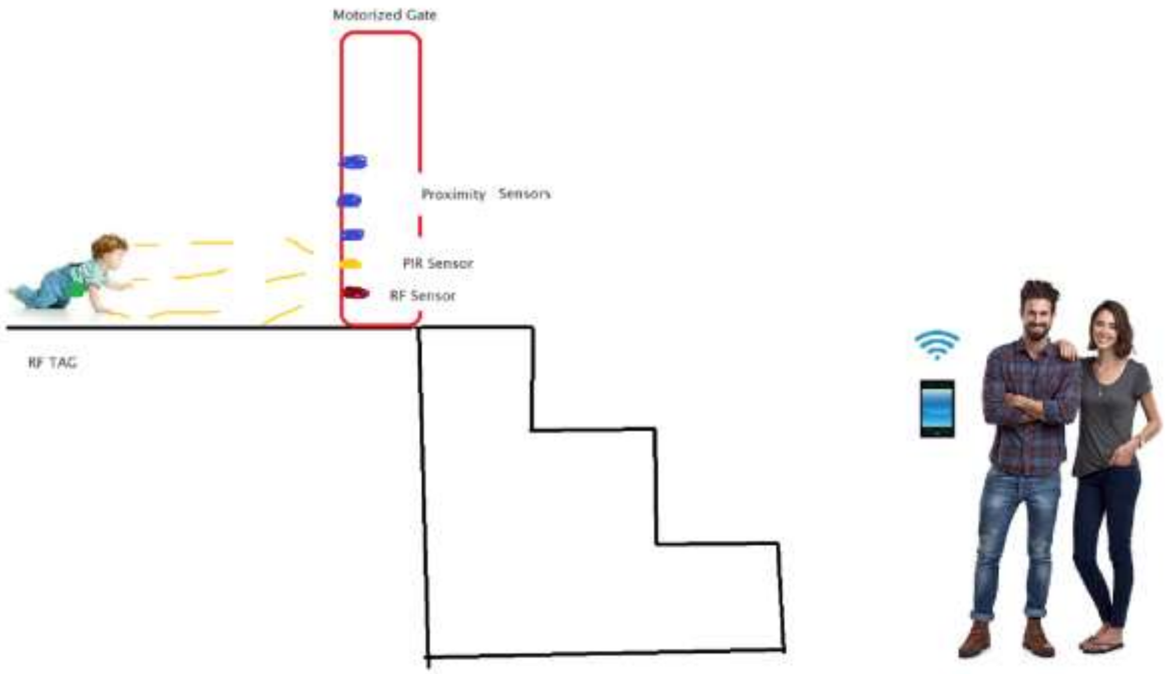

Figure 1: Simple Visual Aid of the Smart Stair Gate

Figure 2: Front-side with sensor placement of the 18-inch wide gate

Figure 3: Back-side with solenoid placement

1.3 High-level Requirements

- Gate detection occurs when an object moves within a 3 ±. 5 foot proximity of the gate and sends a signal to the microcontroller.

- Gate should open when a dog or an adult approaches the gate, but remain closed when a baby is unaccompanied by an adult.

- Collision prevention should be active while the gate is in motion.
    - After the object is gone, the gate will resume the former action

1.4 High Level Overview

1.4.1 Block Diagram

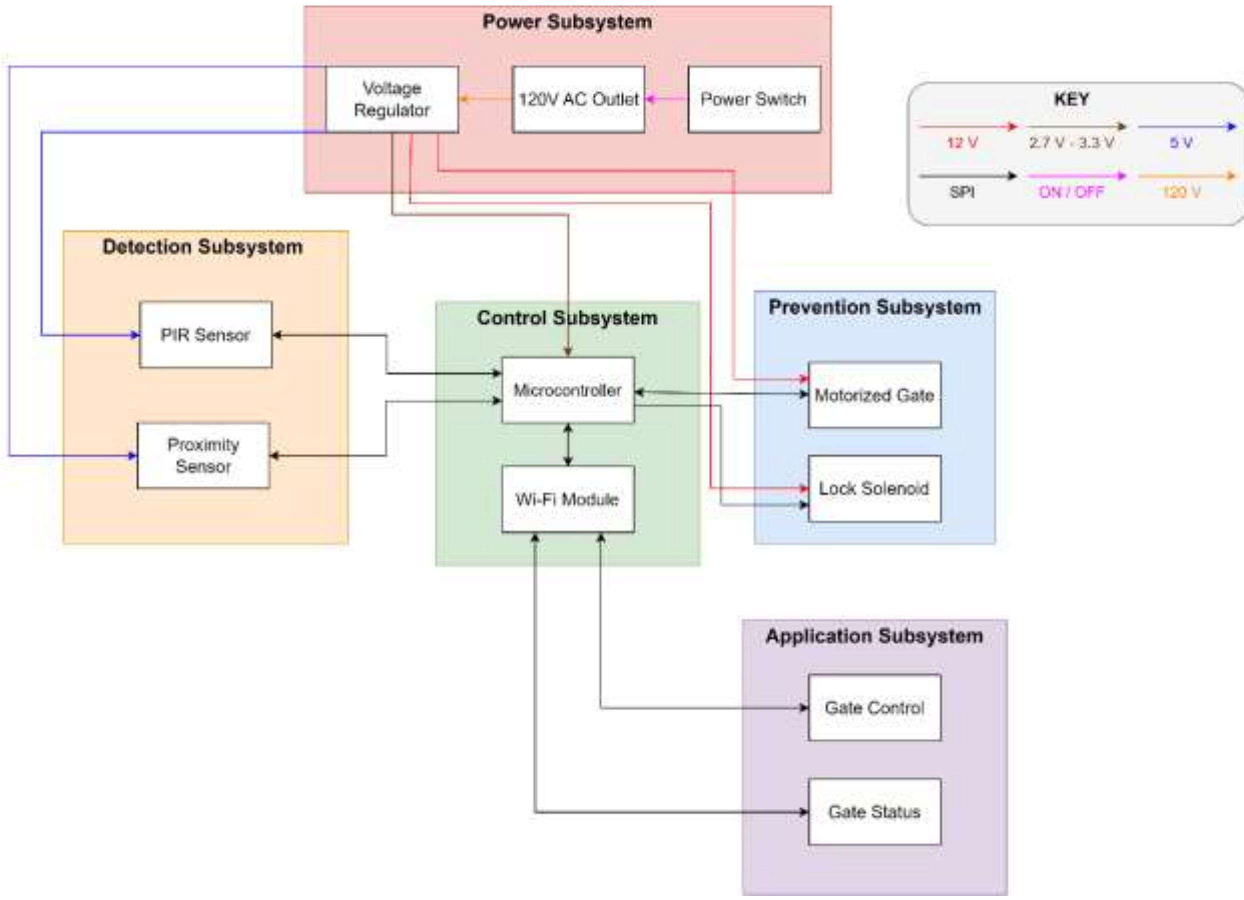

Figure 4: Block Diagram of Smart Stair Gate

1.5 Subsystem Overview

Our product calls for five subsystems: power, control, detection, prevention, application.

1.5.1 Power Subsystem

The power system will connect to the power line and step down by a DC supply. This signal with the use of filters, LDOs, and buck converters, gets the voltages to what we need to safely power the microcontroller and the motors. An LED will be used to indicate the device's functionality. This subsystem will provide functionality and power to the control, detection, and prevention subsystem.

1.5.2 Control Subsystem

The control system is where all the signals are sent to control the product. This subsystem consists of our main PCB equipped with a microcontroller that will send and receive electrical signals. This subsystem will provide functionality to the detection, prevention, and application system as it will be, as the name suggests, control the behavior of all the parts.

1.5.3 Detection Subsystem

The detection system will allow the gate to tell if it should remain open or closed. It consists of multiple sensors including PIR and proximity sensors. The PIR allows the gate to differentiate between objects and living things; proximity sensors for when the device should be on standby and will determine the distance of the object from the gate. The PIR sensor will be always on while all the other sensors will be on standby until PIR sensor is triggered. This subsystem will provide information directly to the control subsystem and determines when the gate opens/closes (prevention subsystem).

1.5.4 Prevention Subsystem

The prevention system is everything that controls the mechanical parts of the products. This subsystem consists of a DC brushed gear motor w/encoder, the gate itself, and a solenoid lock. This subsystem's behavior is dictated by the control subsystem, but both the detection and application subsystem indirectly handle when the gate opens/closes.

1.5.5 Application Subsystem

The application subsystem allows for mobile interaction. It consists of a Wi-Fi signal and a phone app. This phone app will be able to open and close the door from any location and will tell the user the state of the door. The application subsystem communicates with the control subsystem to determine and control the state of the gate (prevention subsystem). This subsystem will also override the detection subsystem to open/close gate.

2 Design

The following section will discuss the detail of the Smart Stair Gate. This section will include, but is not limited to the schematic design, assumptions, and considerations.

2.1 Power Subsystem

2.1.1 Introduction

The power subsystem takes in 120VAC, rectifies and steps the voltage down to 12VDC. From 12VDC, the system uses a buck converter to lower the voltage to 5V and finally a LDO to 3.3V. When developing a power system, it is important to understand the required voltages and load requirements to be able to choose the components and power electronics topologies. From the given sensors, microcontroller, motor, and solenoid lock, the required voltages were 12V, 5V, and 3.3V. Before deciding on components, a worst-case analysis was done to determine the max amount of current drawn from the product (Appendix F). Worst case analysis assumes the worst conditions that a component will handle. WCA is done by looking at the components datasheets and adding the max current. By doing this calculation, and selecting parts that are above the theoretical values, there is no worry about component failure

2.1.2 120VAC to 12VDC

According to the WCA, the maximum current is approximately 3.3A. Due to the relatively high amount of current draw and the unportable nature of a gate, line voltage instead of a battery was used. Batteries store energy based on Ah which considering the current draw would not last long. Originally, to step down the voltage, a transformer, full-bridge rectifier, and filter system was in development, however, this was scrapped due to the allowed use of an external power supply that converts 120VAC to 12VDC with a barrel jack connector to allow for easy connectivity. The external power supply chosen can support up to 5A continuously, well above the maximum current draw.

2.1.3 12VDC to 5VDC

Converting 12VDC to 5VDC calls for a DC-DC power supply. 12V to 5V is a 7V drop. LDOs dissipate power resistively means that it will convert the power into heat with the power equation of P = IV. 7V multiplied with the current draw from the 5V output will result in a high amount of temperature gain which can be harmful to the PCB. Thus, a buck converter which lowers the voltage using switching with inductors and capacitors was chosen for this design. Buck converters work by first converting the 12V signal to a square wave. According to the Fourier Series, the square wave is an infinite number of cosine or sine waves. This square wave is then fed into a low pass filter composed of an inductor and a capacitor essentially transforming the square wave back to a lower voltage DC signal. Inductors and capacitors are used for the low pass because a resistor would dissipate power lowering the efficiency. There are two options when creating a buck converter system: external component or an IC. Due to the difficulty of creating a stable buck converter with external components, an IC was chosen for the design: TPS563252DRLR. The TPS563252DRLR, like most buck converters, contains a feedback resistor system to allow for a more stable output. However, to add safety features to prevent brownout which would damage the microcontroller, there is an in-built UVLO system alongside a power good to ensure a constant output. The small size made it easier to create a compact and more professional looking PCB. It can also support the maximum current draw of the 5V rail. Components chosen were based on the datasheet with ceramic capacitors chosen due to their low amount of ESR and ESL and size compared to say an electrolytic. The inductor was selected with a current above the maximum according to the datasheet. If below it would cause the inductor to saturate, breaking the power supply. Some assumptions made when using a buck converter was that the noise produced at the input would not affect the motors and solenoid. This is a reasonable assumption because the noise and current ripple would be small compared to the input to the motor and solenoid.

2.1.4 5VDC to 3.3VDC

Since 5VDC to 3.3VDC is a relatively small voltage jump and the 3.3VDC rail does not require much current, a LDO is sufficient for the design. LDOs are significantly cheaper than buck converters and do not require many external components. LDOs work using the I-V curve of a mosfet and changing the resistance by changing the gate voltage. Due to the low amount of current drawn from the microcontroller, a 1A LM3940 5V to 3.3V IC was chosen for the design. LDOs are relatively similar to each other with the main difference being the package and maximum current. With LDOs producing heat, there was a need for a heat sink. The heat sink in this case is the many vias connected to ground and the heat pad.

2.1.5 Power Subsystem Verification

Testing this subsystem calls for the probing of the power lines. Due to the large amount of connectors and test points on the PCB, probing the PCB was very easy. Testing consisted of plugging in the barrel jack and either placing the probes at the connector or test points to see the value. In all cases, the voltages were what we were targeting. As to check for the heat produced in the LDO, we plug in the barrel jack for the product to run and see if the LDO overheats after an hour. Results were that it did not overheat.

2.2 Control Subsystem

2.2.1 Control Subsystem Design

The ESP32-S3-WROOM-1 microcontroller is the primary component for the Control Subsystem. This microcontroller receives analog inputs from the PIR sensor, proximity sensors, and motor encoder. Once certain conditions are met, the microcontroller will send digital high output signals to the lock solenoid and DC Gear Motor in the Prevention Subsystem to move the gate. For more information on how the Smart Stair Gate operates, a state diagram chart can be found in Figure 5 under Appendix C.

Another notable feature that is built-in the ESP32 microcontroller is its Wi-Fi capabilities. The microcontroller has been setup as a Wi-Fi Access Point (AP), so other mobile devices can communicate with the Smart Stair Gate. Once a mobile device is connected to the microcontroller, this mobile device can access a web application by using any common web browser. This web application is hosted directly on the microcontroller. For more information on the mobile application, see Application Subsystem Design.

Due to the nature of the proximity sensor, noise became a huge problem which caused a lot of undesired values. In order to resolve this issue, the team came up with the following solution to counteract these issues.

1) Initialize 2 arrays per measurement sensor: **previous measurement values** (holds 5 values) and **average speed values** (holds 10 values).
2) The proximity sensor measures the distance from the object to the gate.
3) Repeat Step 2 until previous measurement values is completely populated
4) Compute average speed values[j] = abs(measurement[i] – measurement[i+1]) where i < size of previous measurement array – 1 & j < size of average speed value array
5) Repeat Step 4 until average speed values array is completely populated
    a. For this algorithm, 50 measurement values are needed in order to calibrate the readings properly
6) Sort average speed values array & find median value

An important note to consider is the size of the arrays. By increasing the size of the array, the Smart Stair Gate will be less prone to accidents (i.e., accidentally opening the gate). However, collision detection is another important attribute that needed to be considered. It may take the Smart Stair Gate too long to realize that an object may be too close. The team deemed the size of initialized arrays as appropriate for this application.

2.2.2 Control Subsystem Design (PCB)

To allow for reusability and easy programmability, a Micro USB was used to allow the programming of the board. Micro USB from a computer outputs a 5V signal as well as a communication bus that is sent with a differential signal (D+ & D-). To turn on the microcontroller, a 3.3V needs to be supplied. The Micro USB is fed into the LDO to produce this 3.3V. In the case both a USB and barrel jack are plugged in a diode was placed between the USB and the LDO to prevent current being drawn into the computer. The computer's communication bus is converted into UART so that the microcontroller can read and be programmed. For the ESP-32 to get programmed both the GPIO 0 and RESET need to be pulled down. DTR and RTS are connected to reset with transistors to check if both pins are pulled down. If they are, the computer will know it can send data to flash the ESP-32.

2.2.3 Control Subsystem Verification

The control subsystem was verified by isolating and testing each component. Each component was then integrated with other components until all connections were built. According to Figure 4, there are seven connections that is connected to this subsystem.

First, the team verified that the ESP32 microcontroller can be powered on by a laptop through a micro USB cable. The datasheet says that the provided voltage from the cable should be sufficient to power the device. The microcontroller powers on as intended.

Second, the team verified that the microcontroller's Wi-Fi Access Point works as intended. We verified this by powering the microcontroller with the laptop, and using sample Arduino code to enable its Wi-Fi capabilities. In the Arduino sample code, the microcontroller will output a unique ID when a mobile device connects to it. Once the sample code is running, our team can connect to the microcontroller through Wi-Fi.

Next, the team verified that the PIR and Proximity sensors from the Detection Subsystem works. In order to verify each sensor works properly, a team member waves their hand at various measured distances.

Then the team verified that motor connections are properly connected. The team tests the motor by turning the gate in the clockwise direction for 500 milliseconds, and then turning the gate in the counter-clockwise direction for 500 milliseconds. The team also verified functionality of the motor-encoder by measuring the distance of the gate travelled. This was done by rotating the gate at various distances.

Finally, the team verified that the solenoid lock is working as intended. The team tests the solenoid lock by enabling and disabling the lock several times through the microcontroller.

2.3 Detection Subsystem

2.3.1 Detection Subsystem Design

The Detection Subsystem is responsible for sensing movement on both sides of the gate and notifying the microcontroller. It is also responsible for determining the distance between the gate and the object moving, as well as the object's speed. This is done through the use of Adafruit PIR motion sensors. The motion sensor is capable of monitoring the amount of infrared radiation in front of it. It splits its viewing into two halves and both halves are monitored at the same level of IR. When a living being passes through, the motion sensor detects the movement because of the change in the level of one half of the IR compared to the other.

Once the motion sensor detects movement, it sends a signal to the microcontroller which results in the HC-SR04 proximity sensor turning on. Now, the proximity sensor is constantly recording the distance between the gate and the object moving, as well as the object's speed. The proximity sensor works by sending out sound waves and monitoring how long it takes for them to bounce off an object and return back to the sensor. This allows the sensor to give a reading off the object's distance from the gate.

During the gate's operation, the sensor is receiving a new distance value every 0.1 seconds. We were able to use this change in time as well as the distance change between two consecutive values to determine speed through the formula $v = d/t$ where d represents distance and t represents time. If the values for speed recorded are higher than the threshold value discussed in the control subsystem section, the gate will assume that it has not detected a toddler and so it is safe to open. On the other hand, if the values for speed recorded are less than the threshold value, the gate will assume the movement is caused by a toddler and it will remain closed. The proximity sensors are also responsible for determining if it is safe for the gate to open based on the distance between the gate and the moving object. In other words, the gate will not start moving if it will collide with something in its path.

2.3.2 Detection Subsystem Verification

The first part of verification for the detection subsystem involved ensuring that each sensor was supplied with the correct voltage. This was done by using a voltage probe to measure the voltage input and ensure that for the motion sensors and for the proximity sensor, 5 volts was recorded.

The next step involved testing each sensor individually to ensure it operates as expected. To test the motion sensors, they were powered up and the output of the motion sensor was connected to an LED. If the motion sensor was working correctly, then every time we moved in front of it, the LED should turn on. Once this was verified, we added another form of verification with the microcontroller by connecting the output of the motion sensor to one of its pins. Code was written so that if the pin that the motion sensor output was connected to went high (meaning that the motion sensor detected movement), the program would print a message saying "Movement Detected". This form of verification was successful as well.

The proximity sensors were tested in a similar sense. They were powered up and connected to the microcontroller so that we could program it to print visual readings of the sensor's outputs. With this, we performed different actions in front of the proximity sensor and made sure that the output values printed were changing accordingly. For example, we put our hand in front of the sensor and move it forwards and backwards to ensure that the sensor reading decreased and increased accordingly. We also faced the sensor at a wall to make sure it was capable of consistently outputting the correct value for distance. This form of verification worked for the most part. There were some issues where the sensors would spike but this is due to environmental conditions and a lack of quality of the sensors used.

2.4 Prevention Subsystem

2.4.1 Prevention Subsystem Design (Mechanical)

The mechanical design of the gate is shown in Figure 2 and 3. Essentially, the motor would move the large cylindrical shaft that would open or close the gate. The reason for the large shaft is to be able to hide the PCB and wiring in the gate allowing for a professional design. Due to manufacturing constraints, the gate was significantly downsized and not similar to the original design with the shaft being extremely small. This design decision from the machine room forced the harnessing to be messy and look unprofessional. Motor was chosen based on the amount of torque it could output. Because there is a relatively large amount of mass that it needs to move, the chosen motor was a gear motor which allows the motor to have an increase in torque while giving up some angular velocity. This trade off does not warrant worry as the motor still functions within the given requirements. While the chosen motor has a large enough torque output, some calculations were done to ensure that the motor can output the amount of torque needed. The power equation is I*V and disregarding any losses that is equal to torque * angular velocity. With the given DC power supply, the maximum current allowed for the motor is around 2.2A with voltage being 12V. Multiplying the two gives us 26.4W of available power. Given that the required angular velocity is 0.105 radians per second, that gives us a torque of approximately 251.42 Nm. Next to ensure that the gate will open we equate the torque with the moment of inertia times the angular acceleration. If the angular acceleration is not near 0 then we do not have to worry about the motor's torque output. The moment of inertia about the axle will be approximately 1/2 m1*r2 + ⅓ *m2*L2 where m1 is the mass of the cylinder and m2 is the mass of the rectangle. Since the radius of the cylinder is mass and the length of the rectangle is low as well as the masses, the moment of inertia about the axis will be low allowing for a higher angular acceleration. Exact values did not need to be calculated, just the general relationships between the values. Appendix F is the mechanical drawing given to the machine shop. As shown in the CAD (Figure 2), the sensor will be placed on the rotating door parallel to the floor as it makes it easier for the firmware to be coded. Also, there is a solenoid lock placed behind the door to allow it to be firmly locked though the motor which has a high amount of torque from the gearbox would be sufficient to lock the door.

2.4.2 Prevention Subsystem Design (Electrical)

Both the motor and solenoid lock require 12V to move. Therefore, it is impossible to control the motor and lock directly with a microcontroller. Since the micro can only output 3.3V, a mosfet with a 3.3V threshold voltage can be used to control the lock. The chosen mosfet's IC curve must agree with current required by the solenoid lock. J4 in the solenoid lock schematic is the connector to the lock itself. The diode acts as a flyback diode as when the mosfet closes after charging the inductor (solenoid lock) the current will have nowhere to go without the diode. While one mosfet can control the motor, the motor also needs the ability to rotate the other direction. This can be done by changing the inputs or by using an H-Bridge circuit. H-Bridge can invert the signal, switching the direction of the inputs to the microcontroller using signals from the microcontroller. It is essentially a group of 4 mosfets. The DRV8870 was chosen due to the ease of use and simplicity.

2.4.3 Prevention Subsystem Verification

This module was very simple to test as it was the movement portion of the project. Simply apply a 12V to both the solenoid lock and motor and if it was able to move then this portion subsystem is successful. To test the angular velocity of the product, plug in the PCB and walk towards the gate. Once the gate begins to move start the timer until it closes. Divide the total angle with the time elapsed and that is the angular velocity.

2.5 Application Subsystem

2.5.1 Application Subsystem Design

Although the Application Subsystem is separate from the Control Subsystem, the ESP32-S3-WROOM-1 microcontroller (which is a part of the Control Subsystem) hosts the application. However, this section will focus on the features and components that relate to the application itself.

The Application Subsystem is responsible for notifying the user of the status of the gate and giving the user control over the Smart Stair Gate. First, a user connects a smart device to the ESP32 microcontroller through Wi-Fi. Once the user is connected to the ESP32 microcontroller, the user will then be able to access the web page that contains the gate functionalities (see Appendix D for the website interface).

While the web page is hosted on the ESP32 microcontroller, it is partitioned in a separate location, independent from the gate logic, so a WebSocket is required to connect the web page to the Smart Gate.

2.5.2 Application Subsystem Verification

This module was tested and verified in conjunction with the ESP32 microcontroller. Because the team already verified that the Wi-Fi module component works, the team mostly focused on the application to work as intended.

First, the team tested the “Status” button of the application by pressing it when the Smart Stair Gate is closed, the Smart Stair Gate is opening, the Smart Stair Gate is opened, and the Smart Stair Gate is closing.

Lastly, we tested the “Open Gate” and “Close Gate” in different manners. Refer to Table 6 in Appendix A for more information on the test cases.

4 Cost and Schedule

4.1 Cost Analysis

4.1.1 Parts

Part Description	Manufacturer	Part Number/ID	Supplier	#	Unit Cost ($)	Total Cost ($)
Gear Motor DC with Encoder	Hyuduo	Gear	Amazon	1	18.90	18.90
Full Bridge Rectifier	Comchip Technology	CDBHD260-G	Mouser Electronics	1	1.50	1.50
Wall to 12V Adapter	Alitove	N/A	Amazon	1	11.99	11.99
Microcontroller	Espressif Systems	ESP32-S3-DevKitC	ECE Shop	1	0.00	0.00
Motion Sensor	Adafruit	189	Digikey	2	9.95	19.90
Proximity Sensor	Sparkfun	HC-SR04	Sparkfun	2	7.95	15.90
H-Bridge Motor Controller	Texas Instruments	DRV8828	Mouser Electronics	1	3.810	3.810
Solenoid Lock	Adafruit	1512	Amazon	1	17.96	17.96
Buck Converter	TI	TPS563252DRLR	TI	1	0.34	0.34
LDO	TI	LM3940	Mouser	1	1.90	1.90
DC Power Jack	Wurth Elektronik	732-5930-ND	Digikey	1	1.02	1.02
10k Ω resistor	Yageo	RC0603FR	Digikey	2	0.18	0.36
47 μF Capacitor	Yageo	GRM188R60J476ME15JCT	Digikey	1	3.00	3.00
Diode	STMicroelectronics	1N5817	Digikey	1	0.42	0.42
Diode	Diotec Semiconductor	1N5401	Mouser	1	0.74	0.74
Micro-USB Connector	TE Connectivity	2040002-1	Mouser	1	2.20	2.20
BJT	Rectron	BC817	Mouser	2	0.14	0.28
Programmer	Jessinie	CH340C	Amazon	1	0.80	0.80
Tactile Switch	CUI Devices	TS04-66-95-BK-160-SMT	Mouser	2	0.18	0.18
0.1 μF Capacitor	Samsung Electro-Mechanics	CL10B104KB8NNNC	Digikey	2	0.10	0.20
10 μF Capacitor	Kyocera AVX	KGM15CR51C106KT	Mouser	2	0.16	0.16
1.0 μF Capacitor	Kyocera AVX	KGM15BR51E105KM	Mouser	4	0.44	1.76

0.47 μF Capacitor	TDK Corporation	C1608X7R1C474K080AC	Digikey	1	0.11	0.11
2.2 μH Inductor	Vishay Dale	IHLP1616BZER2R2M01	Digikey	1	1.06	1.06
22 μF Capacitor	Samsung Electro-Mechanics	CL10A226MP8NUNE	Digikey	1	0.18	0.18
22k Ω resistor	Yageo	RC0603FR-1322KL	Digikey	1	0.10	0.10
2.2k Ω resistor	Yageo	RT0603BRD072K2L	Digikey	1	0.33	0.33
63.4 Ω resistor	Yageo	RC0603FR-0763R4L	Digikey	1	0.10	0.10
220k Ω resistor	Yageo	RC0603JR-07220KL	Digikey	1	0.10	0.10
30.1k Ω resistor	Yageo	RC0603FR-1330K1L	Digikey	1	0.10	0.10
0.2 Ω resistor	Yageo	RL0603FR-070R2L	Digikey	1	0.24	0.24
47k Ω resistor	Yageo	RC0603JR-1347KL	Digikey	1	0.10	0.10
10k Ω resistor	Vishay	RCA060310K0JNEA	Mouser	2	0.10	0.20

4.1.2 Labor

Labor calculations are below. The following values are assumed on a per student basis.

1. Hours of Work per Week = 10 hours
2. Hourly Salary = $40
3. Total Project Duration = 16 weeks

Total Labor Cost = (Student) * (Hourly Salary) * (Hours per Week) * (Total Weeks Worked)

Total Labor Cost = (3) * (40) * (10) * (16) = $19,200

4.1.3 Total Cost

Total Cost = Parts + Labor

Total Cost = $105.94 + $19,200 = $19,305.94

4.2 Schedule

The schedule can be found in Appendix B. The schedule had to be slightly modified from the original plan due to supply chain issues, and unexpected events.

5 Conclusion

5.1 Accomplishments

Overall, we completed our design and all high level requirements were met, although the requirement regarding collision detection would fail at times. We gained a great deal of useful exposure and experience as we went through all stages required in the process of creating a product. This included initial design, testing, debugging, redesigning, and then integrating all the modules together. In addition, our PCB worked as expected (excluding the microcontroller portion) and successfully outputted all required voltages at the different nodes and reduced current effectively to prevent the LDO from overheating.

In addition to the hardware component of the project, we accomplished a lot with the software component, specifically the phone application. The phone application was successful in all aspects and was capable of fully controlling the movement of the gate and updating the user with the gate's current status.

5.2 Challenges

Although the gate successfully operated as designed in the end, we had to overcome design challenges regarding our sensor choice throughout the semester. In our initial design, we planned to use and RFID sensor and tag in addition to the motion and proximity sensors. The RFID sensor was going to allow the gate to differentiate between an adult or pet, and a toddler. The RFID sensor would be placed on the gate, and the tag would be clipped onto the toddler's clothes. In doing so, the gate could detect when the tag is close to it meaning that there is a toddler nearby and it shouldn't open. Unfortunately, the RFID sensor that we chose did not have a range long enough to meet our requirements. We intended for the gate to be able to detect the toddler from a couple of meters away, however the RFID RC522 used in our initial design only had a range of around 10 cm when tested meaning the toddler would be able to get too close to the gate before detection occurred which is an obvious risk.

We were unable to find an RFID sensor that could meet our requirements of operation given the short time frame we had, so we thought of other ways we could end up differentiating between an adult, pet, and a toddler. After discussion with our TA, we decided on using the proximity sensor not only for collision detection, but also for speed detection to help differentiate different moving objects as explained previously.

5.3 Ethical considerations

There were various ethical considerations associated with our project. Firstly, since we were creating a product that was intended to help protect young children, the necessary precautions were taken to ensure compliance with the IEEE ethical code 7.8.1.1 stating "[to] hold paramount the safety, health, and welfare of the public" **[2]** as well as AMC code 1.2 in which we promise to "Avoid Harm" **[3]**. We adhered to these codes by ensuring the gate was only tested in a very safe setting where collision would not occur and monitored the speed of the gate so that it would move slow enough to stop upon detecting an object within its collision zone.

Furthermore, our project involved a technological aspect to it where the user of the gate could operate an app that sends and receives signals to/from the microcontroller. To abide by AMC code 1.6 stating we will "Respect privacy" **[3]**, we ensured that all user data would be stored locally on the microcontroller meaning only the user has access to their data. In addition, the connection between the microcontroller and the phone application is through a Wi-Fi network that only the user has access to.

5.4 Future work

We would like to extend our work on this project by adding various hardware and software aspects to improve usability and overall operation of the gate. Firstly, we would like to add more proximity sensors on the gate at different angles so that a wider view is covered. During our testing, we felt that there were some potential blind spots that would prevent the gate from operating correctly, specifically while the gate was in motion. Adding more proximity sensors would increase the gate's range of view and improve its operation. Secondly, we would like to scale our smart gate to a reasonable size (one which could be placed at the top of a flight of stairs and completely cover its entrance. The smart gate was created for this purpose so it's important that it gets scaled up and tested in that sort of environment.

Regarding the software component, we would like to add an additional feature of the phone application that would allow the user to choose and alter the speed at which the smart gate opens and closes, as well as how long it stays open after no movement is detected. We feel that this is a beneficial feature as different households may have different preferences concerning the operation of the gate. Lastly, we would like to include a detailed log of the smart gate activity, meaning that not only can the user control the gate, but it can also view previous actions performed by the application within a certain period of time.

6 References

[1] Iyamba, N. (2012, March 12). Stairs Among Leading Causes of Injury, Death for Kids. KSL.com. https://www.ksl.com/article/19560857/stairs-among-leading-causes-of-injury-death-forkids#:~:text=Every%20six%20minutes%2C%20a%20child,stair%2Drelated%20falls%20every%20year

[2] IEEE. “”IEEE Code of Ethics”.” (2016), https://www.ieee.org/about/corporate/governance/p7- 8.html

[3] AMC. “”AMC Code of Ethics and Professional Conduct”.” (2018), https://www.acm.org/code-of-ethics

[4] “Ultrasonic distance sensor - HC-SR04 (5V),” SEN-15569 - SparkFun Electronics, https://www.sparkfun.com/products/15569 (accessed Dec. 3, 2023).

[5] A. Industries, “Pir (motion) sensor,” adafruit industries blog RSS, https://www.adafruit.com/product/189 (accessed Dec. 3, 2023).

[6] “Micro bit lesson - using the Ultrasonic module,” Micro bit Lesson - Using the Ultrasonic Module " osoyoo.com, https://osoyoo.com/2018/09/18/micro-bit-lesson-using-the-

Appendix A Requirement and Verification Table

A.1 Power Subsystem

Number	Requirements	Verifications
1	Supply 12 Volts after voltage transformer.	Measure positive 12 Volts with voltmeter at the output node of the voltage transformer, with respect to ground.
2	Convert 12 Volts to 5 Volts with a buck converter.	Measure positive 5 Volts with voltmeter at the output node of the buck converter, with respect to ground.
3	Convert 5 Volts to 3.3 Volts with a low-dropout regulator.	Measure positive 3.3 Volts with voltmeter at the output node of the buck converter, with respect to ground.
4	UVLO halts circuit operation in response to power outage/decrease in power.	Measure 0 Volts with voltmeter across the rest of the circuit if UVLO is activated.
5	Power Good sends signals to microcontroller to control LED	Microcontroller turns on LED if power is plugged in

Table 1: Requirements and Verifications of Power Subsystem

A.2 Control Subsystem

Number	Requirements	Verification
1	Supply current between 180 mA to 240mA to Microcontroller.	Measure between 180 to 240 mA with an ammeter at ESP32 node.
2	Solenoid lock notifies the microcontroller when it is locked and the gate is in zero state (closed) position.	Check to see if the microcontroller receives data recognizing that the gate is closed.
3	LDO output within 2.7 Volts to 3.6 Volts.	Measure positive 3.3 Volts (rms) with voltmeter at the output node of the buck converter, with respect to ground.
4	Successful communication between ESP32 and sensors using SPI.	Verify the measurements from a distance of 3 feet away from the gate.

Table 2: Requirements and Verifications for Control Subsystem

A.3 Detection Subsystem

Number	Requirements	Verification
1	Supply 5 Volts to the HC-SRO4 breakout board.	Measure positive 5 Volts with voltmeter at the HC-SR04 node, with respect to ground.
2	Supply 3.3 Volts to the RC522.	Measure positive 3.3 Volts with voltmeter at the RC522, with respect to ground.
3	HC-SR04 proximity sensor detects motion 3 feet from the gate.	Move object within 3 feet distance and check sensor's output on terminal.
4	RC522 RFID sensor detects RF tags 3 feet from the gate.	Place a RFID tagged toddler approaching the gate from >3 feet away, and attempt to trigger the sensor when toddler is within required distance (at 3 feet)

Table 3: Requirements and Verifications for Detection Subsystem

A.4 Prevention Subsystem

Number	Requirements	Verification
1	Supply 12 Volts to the solenoid lock.	Measure positive 12 Volts with voltmeter at the solenoid lock node, compared to ground.
2	Gate position is capable of changing between 0-90 degrees.	Check gate position in closed and open state to ensure that the degrees are 0 and 90 respectively.
3	Gate opens and closes slowly, removing risk of hitting an object before it is detected.	Monitor and record gate movement speed to ensure it fits our requirements. Gate should rotate 0.105 radians per second

Table 4: Requirements and Verifications for Prevention Subsystem

A.5 Application Subsystem

Number	Requirements	Verification
1	The application should notify the user of the current status of the gate (whether it is open or closed).	Repeatedly open and close the gate and check ensuring whether the application constantly displays the correct state.
2	The user should be able to manually open/close the gate by pressing a button on the application.	Monitor whether the gate behaves appropriately when the user presses the button to open/close it.

Table 5: Requirements and Verifications for Application Subsystem

Number	Test Case	Verification Procedures
1	Application should notify the user the status of the gate	The user presses the "Status" button once on the mobile website.
2	"Open Gate" overrides sensor's behavior and opens the Smart Gate.	The user presses the "Open Gate" button once on the mobile website.
3	"Close Gate" overrides sensor's behavior and closes the Smart Gate.	The user presses the "Close Gate" button once on the mobile website.
4	Pressing "Open Gate" will open the gate and ignores sensor behaviors and other override behaviors (i.e., pressing "Close Gate")	The user will press "Open Gate" once and then repeatedly press "Open Gate" and "Close Gate" on the mobile website.
5	Pressing "Close Gate" will close the gate and ignores sensor behaviors and other override behaviors (i.e., pressing "Close Gate")	The user will press "Close Gate" once and then repeatedly press "Open Gate" and "Close Gate" on the mobile website.

Table 6: Test Cases for Application Subsystem

Appendix B Schedule

Week Starting	Alexander Chin	Brandon Lau	Zeyad Irsheid
(9/25)	• Design Document • PCB Design	• Design Document	• Design Document • PCB Design
(10/2)	• Prepare Design Review • Order Power Subsystem & Control Subsystem • PCB Design	• Prepare Design Review • Programming microcontroller • Verify part orders	• Prepare Design Review • Order Detection & Prevention subsystem parts
(10/9)	• First Round PCB Order • Programming Microcontroller • Build and test Power and Control System	• First Round PCB Order • Programming Microcontroller • Finalize with machine shop on mechanical design and make adjustments • Assist with building and testing subsystems	• First Round PCB Order • Programming Microcontroller • Build and test Detection subsystem
(10/16)	• Second Round PCB Order • Review PCB Design • Begin integrating subsystems	• Second Round PCB Order • Review PCB Design • Program mobile app	• Second Round PCB Order • Review PCB Design • Program mobile app
(10/23)	• Continue integrating subsystems	• Continue integrating subsystems	• Continue integrating subsystems
(10/30)	• Continue integrating subsystems	• Continue integrating subsystems	• Continue integrating subsystems
(11/6)	• Verify/Debug product and make necessary adjustments	• Verify/Debug mobile app and make necessary adjustments	• Verify/Debug mobile app and make necessary adjustments
(11/13)	• Mock Demo Preparation	• Mock Demo Preparation	• Mock Demo Preparation
(11/20)	• Fall Break	• Fall Break	• Fall Break

(11/27)	• Final Demo	• Final Demo	• Final Demo
(12/4)	• Final Presentation	• Final Presentation	• Final Presentation

Table 7: Project Schedule

Appendix C Smart Stair Gate Logic

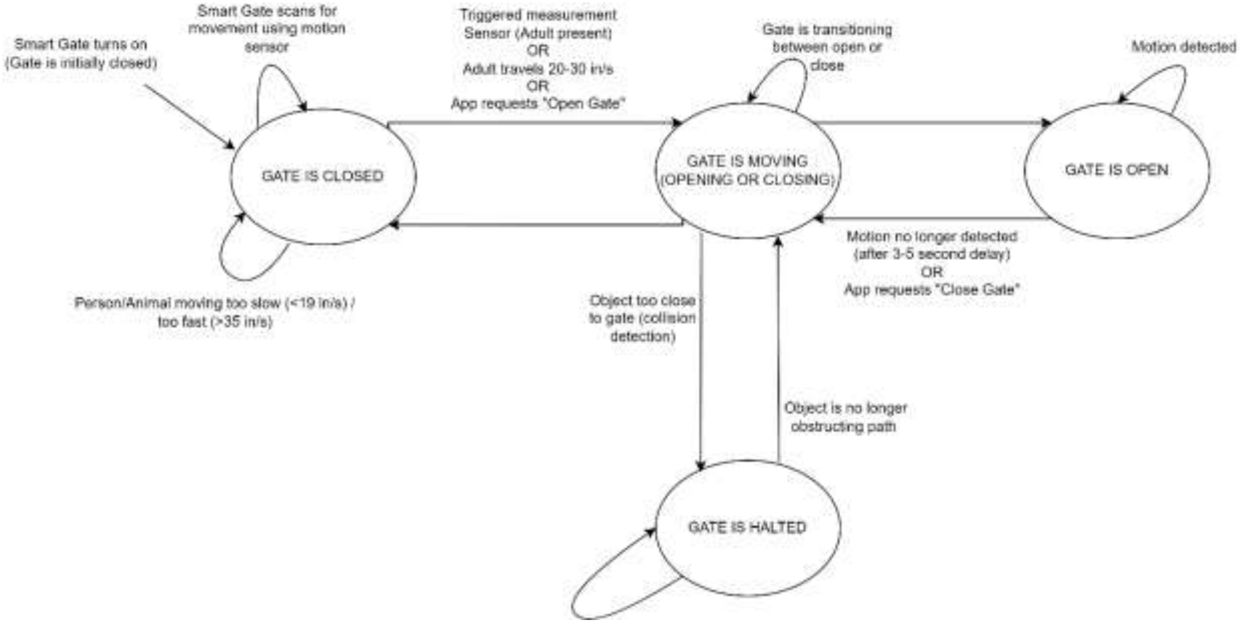

Figure 5: Smart Stair Gate Logic

Appendix D Application Interface

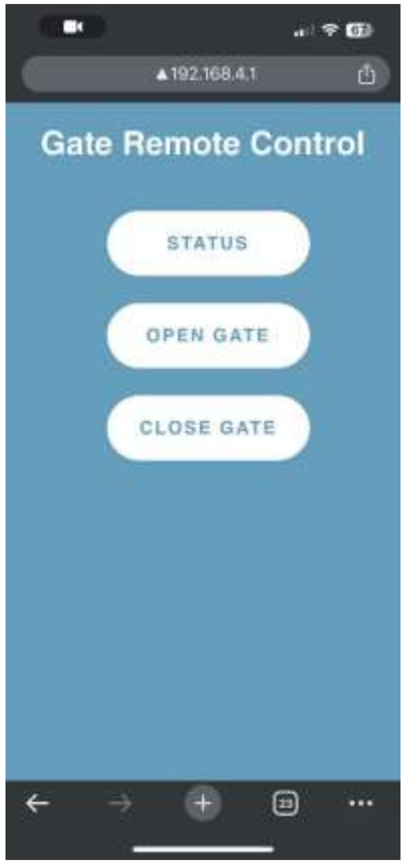

Figure 6: Smart Gate Website Interface

Appendix E Smart Stair Gate Pictures

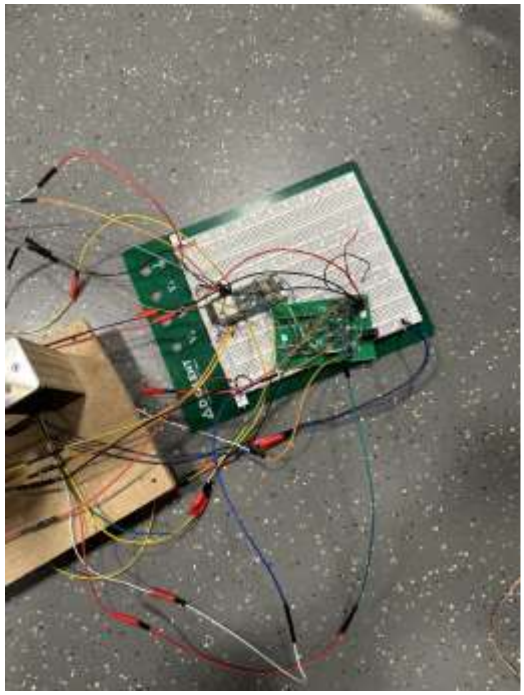

Figure 7: Control Subsystem requires a development ESP-32 board.

Appendix F Max Current Draw

Component	Typical Current	Max Current
Motor	1.15A	2.2A
Solenoid Lock		650mA
ESP 32	107.9mA (Moden	340mA
Motion Sensor		123 uA @ 5V
Proximity Senso	15mA	15mA

Figure 8: Max Current Draw

Appendix G PCB Design

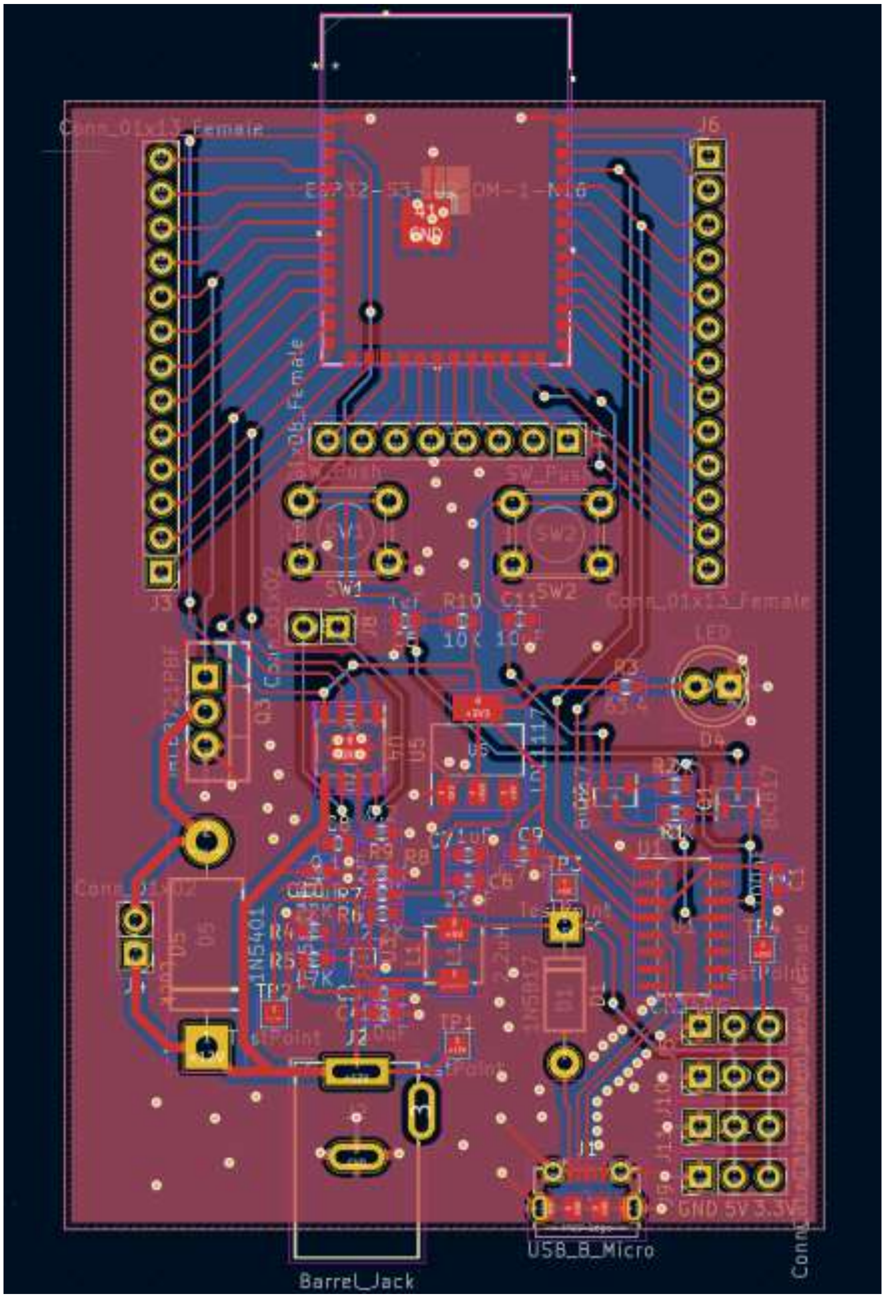

Figure 9: Smart Stair Gate PCB Design

Appendix H Mechanical Drawing

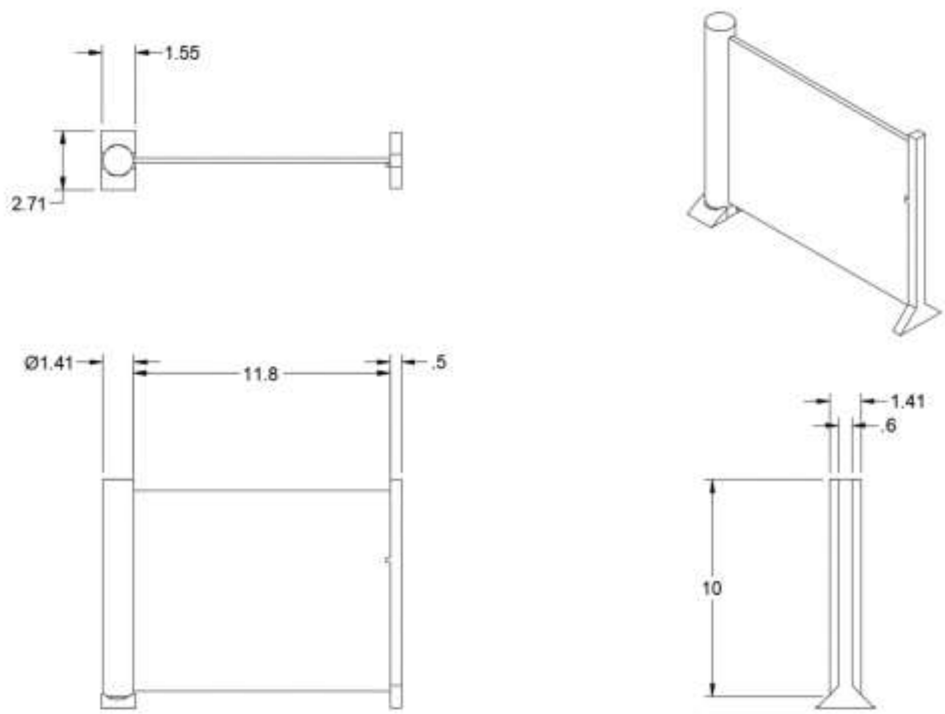

Figure 10: Mechanical Drawing

Appendix I Circuits

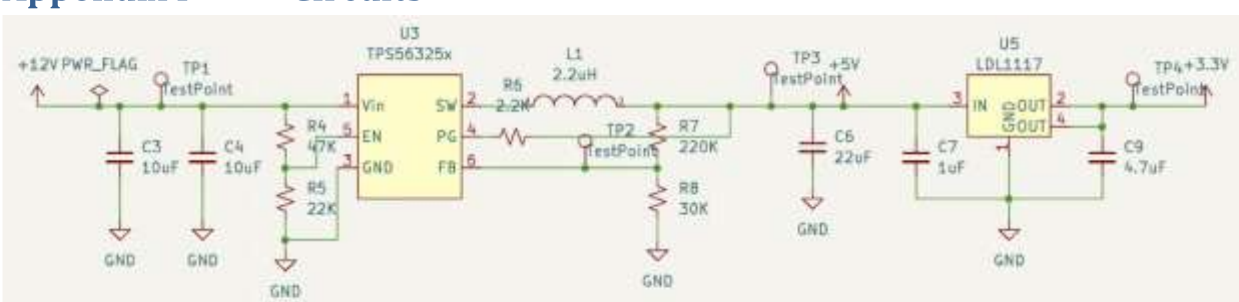

Figure 11: Power Supply Schematic

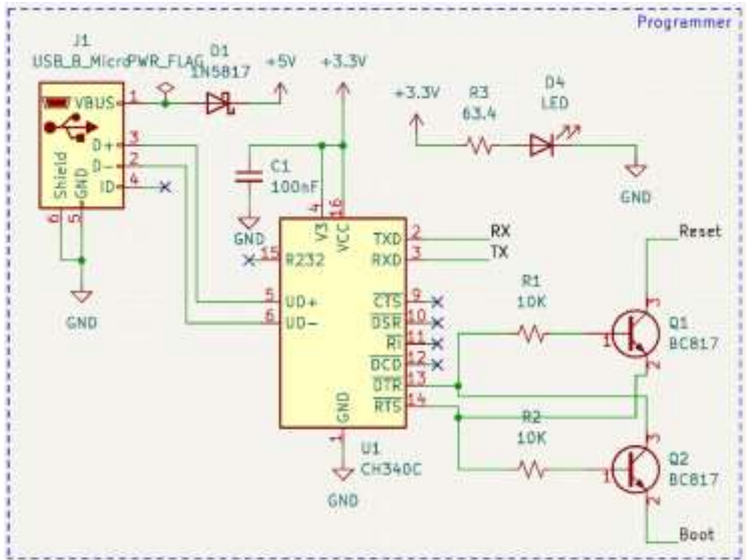

Figure 12: PCB Programmer

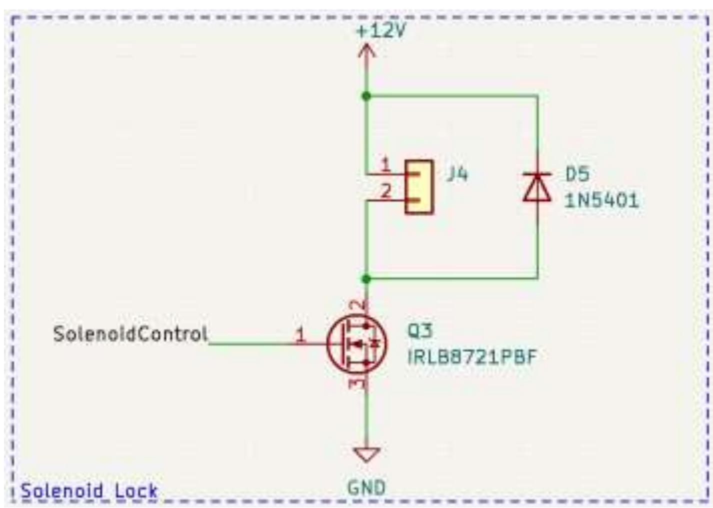

Figure 13: Solenoid Lock Schematic

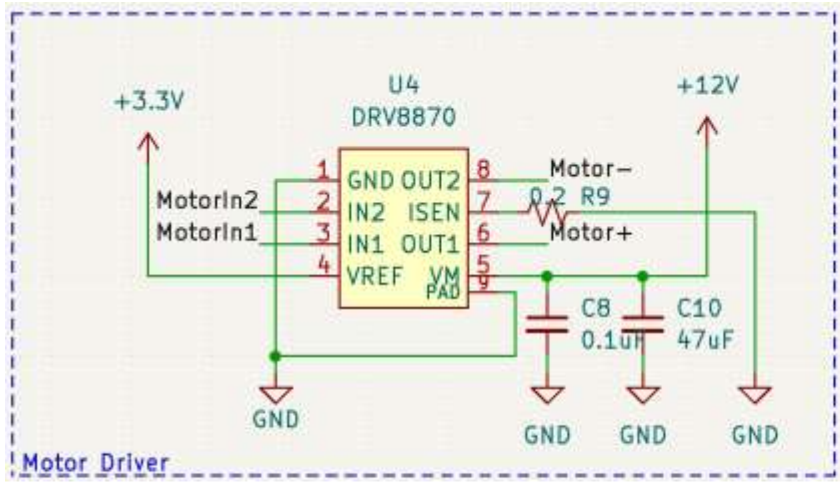

Figure 14: Motor Driver Schematic