ECE 445
SENIOR DESIGN LABORATORY

FINAL REPORT

Moving Alarm Clock

TEAM 31

Teja Nerella (nerella2)
Karthik Bagavathy (kb42)

TA
Gong, Rui

December 10, 2024

Abstract

The following report provides details into the design, implementation, and performance of a moving alarm clock system that offers a more effective way for the user to wake up by requiring physical activity to disable the alarm. The final system contains seven different subsystems, with some of the important ones including an alarm sound, user interface, and mechanical movement subsystems.

Some of the key functionalities of the product includes generating an attention grabbing alarm sound during the user set time, movement of the system (with obstacle detection and random turning) during the course of the alarm, and the alarm's quick deactivation. The overall system is powered by Double A (AA) batteries and the logic being controlled by the ESP32 microcontroller.

Several tests for each subsystem ensured that the system meets specific requirements in regards to sound level, movement speed, obstacle detection accuracy, and power supply.

Contents

1. Introduction

The following section contains a detailed look at the problem that has been addressed, as well as the solution that was used to address the problem. In addition, the visual aid (high-level physical design) was included to visualize the overall design of the product

1.1 Problem

Many individuals face issues with waking up in the morning, with a certain study indicating that nearly 70% of people regularly press the snooze button on their alarms [2]. Pressing the snooze button isn't merely a minor inconvenience as it can significantly impact daily routines and reduce productive time in the morning. Another research study has shown that waking up late leads to higher risks of developing mental health issues, such as depression and anxiety, as it leads to unhealthy sleep patterns and disruptions to the circadian rhythm [3]. The repeated ringing of alarm isn't just an individual problem as anyone who lives with them would also end up getting disturbed, leading to potential stress or conflicts with shared living environments.

1.2 Solution

Our proposed solution to the problem described above is a moving alarm clock. This alarm clock will start playing the alarm sound at the same time that it starts moving. The user will have to chase the alarm clock in order to disable it, helping them get exercise before turning off the alarm. Exercise raises core body temperature which helps wake people up in the morning, similar to a warm shower. It also gets the user away from their bed, decreasing the urge to sleep again. The alarm will have a sensor to detect obstacles and turn away from them. The device will randomly choose which direction to turn, making it harder for the user to catch the vehicle.

1.3 Visual Aid/Physical Design

The following physical design provides a clear idea of the layout of the components within the moving alarm clock.

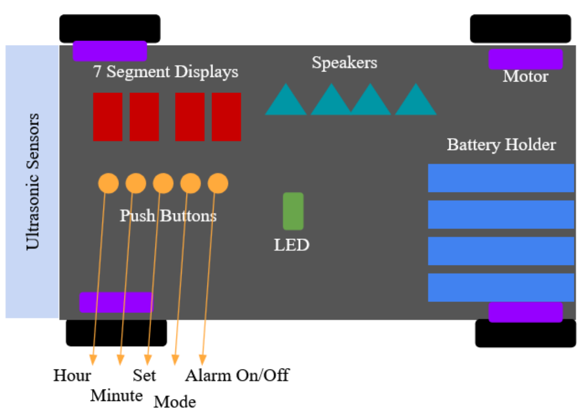

Figure 1: Physical Design for Moving Alarm Clock

2 Design

2.1 Block Diagram

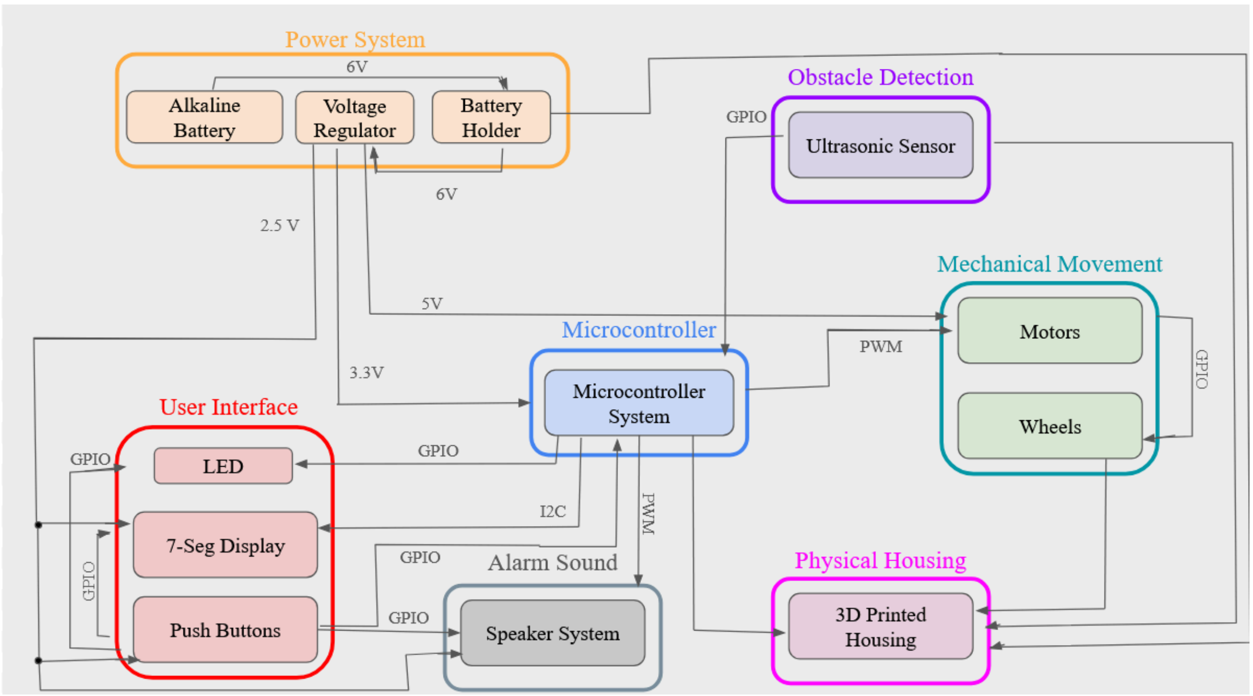

Figure 2: Block Diagram with subsystems and connections for Moving Alarm Clock

2.2 High-Level Requirements

The following are the high-level requirements that we had set to accomplish in the before starting the design and implementation the project:

1. The first requirement is the alarm should ring within 10 seconds of the clock strikes the set alarm time. This is important because it is what causes the user to wake up.
2. The next requirement for success is the robot should start to move while the alarm is sounding. The robot should also be able to avoid crashing into walls and other obstacles using sensors. The sensors should be able to detect walls from 30 cm away and should randomly turn left or right after detecting a wall.
3. The alarm sound system should also be able to be turned off within 10-60 seconds if the user manages to catch the robot. This gives the user an incentive to get up and catch it, waking them up in the process.

2.3 Subsystem Descriptions

The following section contains the description of all of the seven subsystems present in the system. Individual schematic diagrams for each subsystem are provided in Appendix D.

2.3.1 Power Subsystem

The device will be powered by EN91 AA Alkaline batteries to enable untethered movement for the alarm actions to be performed, essentially removing a space constraint because no power cords are needed. The batteries provide enough power to handle continuous motor operation, sensor activity, and the alarm sound for the necessary duration. A voltage regulator is included with the power system in order to provide different voltages for different subsystems (eg. microcontroller system needs 3.3V and the 7-segment display needs 2.5V [lower Vf])

2.3.2 User Interface Subsystem

The user interface system consists of a 4 digit 7-segment display, 5 push buttons, and a LED. The 7 segment display has 4 digits in order to show time that is accurate to the nearest minute. The 5 buttons are as follows: one for hour set, one for minute set, one to input confirmation, one for mode change, and one for alarm on/off (LED used to indicate that). The push buttons will send GPIO signals to the microcontroller system, with each button having a separate input in order to perform different functions. From the microcontroller, the output pins will lead into the 7-Segment display and the LED (corresponding to buttons 1,2, 3 and 5). Then the outputs from button 4 will be sent to the speaker for the alarm turn off mechanism.

2.3.3 Mechanical Movement Subsystem

A motorized base with omnidirectional wheels is present in the vehicle, allowing for free movement in each direction, enabling smooth, dynamic movement, enhancing the effectiveness of the alarm by making deactivation challenging. The motor will obtain instructions from code present in the microcontroller and will set the speed accordingly. Once the alarm time has been reached, the signal will be transferred to the microcontroller which will then provide instructions to the motor for movement.

2.3.4 Obstacle Detection Subsystem

Utilizing ultrasonic sensors to detect obstacles nearby and enable smoother and safer movement around obstacles. Real-time data will be provided to the microcontroller, enabling the device to adjust its movement path when obstacles are detected, preventing collisions or getting stuck. Detecting obstacles and moving around them form a necessary part of the overall system's function, thereby making it an important system.

2.3.5 Alarm Sound Subsystem

There is a speaker controlled by the PCB and microcontroller to produce the alarm sound at a given user-specified time. The microcontroller will trigger the speaker based on the programmed alarm schedule, ensuring that the activation occurs on time. The sound system will be designed to produce a loud, attention-grabbing alarm, making it difficult to ignore. The sound of the alarm will continue until the user deactivates it.

2.3.6 Physical Housing Subsystem

There will be a 3d printed durable housing that will make sure that the internal electronics are shielded from damage during movement and impact. The microcontroller, sensors, and motors are essential to the overall system and therefore needs to be protected. The housing will be designed in such a way in order to absorb maximum impact and still keep the product lightweight and usable.

2.3.7 Microcontroller Subsystem

The microcontroller system will utilize an ESP32 microcontroller which will handle all core operations, including motor control, alarm activation, sensor input, and user interface interactions. It will also ensure smooth interaction between the LED / user interface and the functionality of the device, ensuring seamless integration while optimizing performance.

2.4 Design Details

Some of the design considerations in regards to certain important components of the product have been discussed below.

2.4.1 Alarm Sound Subsystem

There was a specific design choice made to use 4 different speakers of the model we were using instead of just one. The average alarm clock is around 70-80 dBA. In order to be as loud as a regular alarm clock we would like our alarm clock to be at least 75 dBA. The speakers that we use have a sound level of 72 dBA with a margin of error of 3 dBA.

Since dBA is logarithmic we can't obtain the sound level by simply adding the dBA for each speaker together. The formula for a combination of sound sources is:

$$L_{tot}=10\bullet log_{10}(10^{L1/10} + 10^{L2/10} + \ldots + 10^{Ln/10})$$

where Ltot is the total sound level of all of the sources combined and Ln is the sound level of the nth sound source. The table below shows how, using this formula, the expected sound level increases as the number of speakers increases.

Table 1: Expected Sound level values

	1 Speaker	2 Speakers	3 Speakers	4 Speakers	5 Speakers
Expected sound level (dB)	72	75.01	76.77	78.02	78.99

The table shows that just adding 2 speakers is enough to reach the desired sound level of 75 dBA. However we can't take these numbers as the actual value. Since there is a margin of error of 3 dBA we can't be sure that 2 speakers will actually produce enough sound. If we take the margin of error into account and refer to the minimum sound levels, 4 speakers will guarantee that the sound level will be at least 75dB.

2.4.2 Obstacle Detection Subsystem

There are several reasons as to which ultrasonic sensors are used over Light Detection and Ranging (LiDAR) or other alternatives. One of the reasons is cost-effectiveness, where a typical ultrasonic sensor would cost $2-$10 while a typical LiDAR sensor would cost $80-$200 for basic models, making the cost ratio around 8 (80/10) in the best case scenario. In addition, let's consider the following formula:

$$Distance \ = \ (Speed\ of\ Sound \ * \ Time\ Delay)/2$$

If we set this equation to be representing the time delay instead, we get the value of 5.83ms ((2*1)/343), which is much lower than the tolerance level set for the sensor communication with the microcontroller. The final reason to consider, in regards to design choices, is the fact that ultrasonic sensors perform well in low light and varying environmental conditions. They are robust against dust and debris in comparison to LiDAR which can struggle in dusty, foggy, or brightly lit conditions as these impact laser performance.

2.4.3 Power Subsystem

The presence of the voltage regulator is to ensure that different subsystems end up receiving different voltages from the power system. This is to make sure that all of these systems end up receiving more voltage than their Voltage forward (Vf) so that they can function as intended, but also not too much voltage to the point where it causes component degradation. The following table shows the Vf values for all of the components and it can be seen that they are all met through utilization of the voltage regulator.

Table 2: Vf per System

Component	Voltage Forward (V)
ESP32 Microcontroller	3.3
LED	2.0 - 3.3
Ultrasonic Sensor	5.0
Motor	5.0
7-Segment Display	2.4 - 2.5
Push Buttons	0 (no forward voltage, acts as a switch)
Sound System	2.5

3. Design Verification

The following section contains all of the subsystem requirements and the verification results for each of those requirements. All of the subsystems were tested extensively to ensure that they met the design goals. Provided below are the requirements, testing methodologies, and data.

3.1 Power Subsystem

3.1.1 Requirements

The following are the three requirements for the power subsystem:

1. The power subsystem needs to provide at least 3.3V to the ESP32 microcontroller, LED, and ultrasonic sensor. This has been achieved as shown in Table 1
2. To ensure continuous operation, the power subsystem needs to provide at least 5V +/- 0.5 V to the motor. This has been achieved as shown in Table 1
3. The power subsystem needs to provide at least 2.5V to the display, push buttons, and sound system. This has been achieved as shown in Table 1

3.1.2 Verifications

Table 3: Voltage per System

System	Voltage (V)
Microcontroller	3.30
LED	3.32
Sensor	3.41
Motor	6.00
Speaker	3.31
Display	3.32

This table showcases the amount of voltage that each system receives. It is clear throughout this table that each system receives a sufficient amount of voltage.

3.2 User Interface Subsystem

3.2.1 Requirements

The following are the four requirements for the user interface subsystem:

1. The 7-segment display must maintain a brightness of at least 125 +/ 10 cd/m^2 (obtained from typical screen brightness) to ensure visibility. This has been achieved as shown in Figure 3
2. All of the push buttons must be able to read input with a maximum response time of 100 +/ 20 ms to ensure smooth functionality. This has been achieved as shown in Table 2
3. In a 1 hour period, the clock in the system should be accurate to the nearest +/- 1 minute. This has been achieved as shown in Figure 4 and Figure 5
4. The alarm status LED must be visible from a 2 meter radius and change status within 200 +/- 20 ms of the push button press This has been achieved as shown in Table 2

3.2.2 Verifications

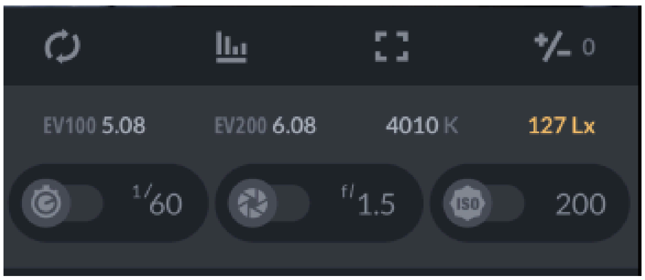

Figure 3: Lumen Measurement of Display

This figure shows how the brightness of the display was measured using a phone application. This application measured the brightness as 127 Lx, greater than the required 125 Lx.

Figure 4: Setting Time of Clock

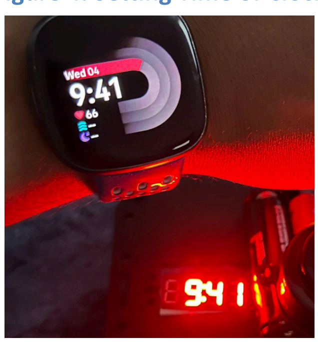

Figure 5: Ending Time of Clock After 3 Hours of Running

These 2 figures show how the displacement of time was measured. We ran the clock for 3 hours and found that it didn't deviate from the real time by more than a minute.

Table 4: Time to Execute Each Task Given Input

Input	Time to execute (ms)
Alarm Status Button	0.00000
Hours Button	0.00000
Minutes Button	0.00000
Set Alarm Button	0.00000
Sound Turn Off	0.00000
Turn Signal	10.00000

This table shows how long each action took in ms since the input was received. All actions are executed in 10 ms or less, fulfilling the requirements

3.3 Mechanical Movement Subsystem

3.3.1 Requirements

The following are the three requirements for the mechanical movement subsystem:

1. The vehicle must be able to move at a speed of 7 +/ 2 mph (3.13 +/- 0.83 m/s) on a flat smooth surface. This has been achieved as shown in Table 4
2. The turning radius of the system must be at least 10 cm to ensure 90 degree turns. This has been achieved as shown in Table 3
3. Once the obstacle signal has been provided from the microcontroller, the wheels must turn within 500 +/ 50 ms of the signal. This has been achieved as shown in Table 2

3.3.2 Verifications

Table 5: Turn Radius of Moving Alarm Clock

Turn Radius Trial	Values (cm)
1	13.6
2	11.2
3	12.3
4	13.1
5	11.2

The turn radius of the device was measured across 5 trials. All 5 trials measured a turn radius over the required 10 cm.

Table 6: Speed of Moving Alarm Clock

Movement Speed Trial	Values (mph)
1	7.9
2	7.2
3	7.0
4	7.7
5	7.4

This table shows how the speed of the device was measured across 5 trials. All trials show a speed above the required 7 mph.

3.4 Obstacle Detection Subsystem

3.4.1 Requirements

The following are the two requirements for the obstacle detection subsystem:

1. The obstacle detection system shall accurately detect obstacles within 1 +/- 0.5 meters. This has been achieved as shown in Figure 6
2. The sensor must have a field of view of 120 +/- 10 degrees to ensure large coverage for most indoor environments This has been achieved as shown in Table 5

3.4.2 Verifications

Table 7: Detection of Objects by Ultrasonic Sensor at Angles

Angle	LED
100 degrees	ON
120 degrees	ON
140 degrees	OFF

This table shows how the sensor detects objects from multiple angles. It shows how objects within the required 120 degree range are able to be detected.

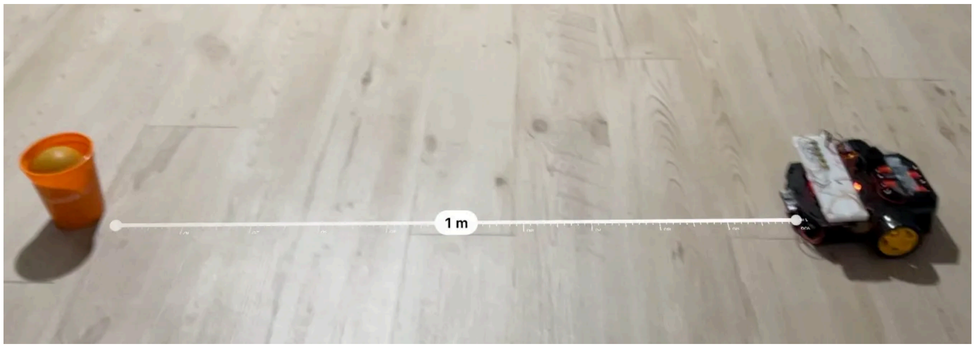

Figure 6: Moving Alarm Clock Detecting Object from 1 m Away.

This figure shows how the device can detect obstacles from 1 m away. The LED is lit up indicating that the ultrasonic sensor is detecting an object

3.5 Alarm Sound Subsystem

3.5.1 Requirements

The following are the three requirements for the alarm sound subsystem:

1. The speaker must produce a sound with a volume of at least 75 dB +/- 5 dB at a distance of 1 meters from the device. This has been achieved as shown in Figure 7
2. The alarm shall stop within 200 +/- 20 ms of the user pressing the button to deactivate it. This has been achieved as shown in Table 2
3. The alarm sound must have a frequency between 2000 and 4000 Hz to match an attention grabbing alarm sound. This has been achieved as shown in Figure 8

3.5.2 Verifications

Figure 7: Decibel Level of Moving Alarm Clock Speaker

This figure shows that the decibel value, measured through a phone application, is within the bounds of over 75 dB

Figure 8: Frequency Level of Moving Alarm Clock Speaker

This figure shows that the frequency value, measured through a phone application, is within the bounds of between 2000-4000

3.6 Physical Housing Subsystem

3.6.1 Requirements

The following are the three requirements for the physical housing subsystem:

1. The physical housing system must be able to protect internal components with no more than 2cm of surface deformation after being dropped from a height of 10m. After performing drop tests this has been verified and no visible damage has occurred to the housing
2. After the internal components are mounted, the 3D printed housing must weigh no more than 3000 +/- 10g. This has been achieved as shown in Table 6
3. After exposure to around 20mL of water splashes in a span of 30 seconds, there should be no visible moisture in the internal components. After performing this test there was no moisture within the housing.

3.6.2 Verifications

Table 8: Weight of Moving Alarm Clock

Weight (g)	1451

This table shows that the weight of the device is less than 3000 g, showing that it is within the bounds

3.7 Microcontroller Subsystem

3.7.1 Requirements

The following are the two requirements for the microcontroller subsystem:

1. The ESP32 Microcontroller must execute major operations (time set, alarm activation, motor movement) within +/- 10 ms for each task (time specified in each subsystem). This has been achieved as shown in Table 2
2. After performing core tasks for 5 minutes, the temperature of the microcontroller's temperature must not exceed 70 +/- 5C. This has been achieved as shown in Table 7

3.7.2 Verifications

Table 9: Temperature of Microcontroller Over Time

Time (min)	Temperature (C)
10	40.3
20	40.5
30	40.6

This table shows that the temperature of the device is less than 70 C as time goes on, showing that it is within the required range.

4. Cost and Schedule

4.1 Parts

Table 10: Parts Costs

Part	Manufacturer	Retail Cost ($)	Bulk Purchase Cost ($)	Actual Cost ($)
ESP32 Microcontroller	Espressif Systems	3.48	3.48	3.48
Speaker	PUI Audio	4.01	1.82	16.04
Ultrasonic Sensor	OSEPP Electronics LTD	6.07	4.05	6.07
Motors	Digilent	7.22	4.18	28.88
Wheels	TinyCircuits	3.95	3.95	7.90
7 Segment Display	Lumex Opto/Components Inc.	5.67	3.36	5.67
Push Button	E-Switch	0.51	0.18	2.55
Batteries	Energizer	0.65	0.65	2.60
Battery Holder	DFRobot	3.39	2.5	3.39
Voltage Regulator	Texas Instruments	2.51	1.35	2.51
Total Cost:				79.09

4.2 Labor

After conducting research on the average salary of Computer Engineers in the state of Illinois on ziprecruiter.com, we have determined the figure to be around 55 dollars per hour ($55/hr). Given that this project is in a 10 week duration and there is a rough expectation of working 20 hours a week by each of the 2 teammates in order to accomplish the tasks set in the project, the overall labor cost comes out to be 55 ($/hour) * 2 * 20 (hours/week) *10 (weeks) = $22,000 over the entire project.

4.3 Total Cost

The total cost can be determined by adding the labor cost and the cost for the parts, which in this case would be $22,000 + $79.09= $22,079.09.

4.4 Schedule

The project schedule, from start to finish, is included below with updates based on any changes the team had to face.

Table 11: Schedule Table

Week of 9/30	Complete Design Document and Circuit Schematic
Week of 10/7	Work on PCB Design
Week of 10/14	Work on PCB Design
Week of 10/21	Order PCB, Order Parts
Week of 10/28	Work on coding
Week of 11/4	Work on coding
Week of 11/11	Work on coding, Prototype Assembly
Week of 11/18	Prototype Assembly, Solder PCB, Mock Demo
Week of 11/25	Fall Break
Week of 12/2	Prototype Assembly, Final Demo
Week of 12/9	Final Presentation

5. Conclusion

5.1 Accomplishments

Overall, the product has accomplished all 3 of its high-level requirements. The alarm system sounds within 10 seconds of the alarm set time, the obstacle detection system can detect obstacles within 30 cm with random turning logic complete and working, and the sound system turns off within 10-60 seconds of the user pressing the button. In addition, all of the subsystem requirements were satisfied within a reasonable range. The final prototype is an affordable and innovative solution, integrating critical features such as obstacle detection and user interaction, ensuring it fulfills its purpose as an engaging and efficient alarm system.

5.2 Uncertainties

There were a few uncertainties and challenges that were encountered during the course of the project. One of the main challenges was that the PCB couldn't be used in the final implementation of the product. The motor driver that was used in the schematic design was for a motor that had been discontinued, making it not possible to integrate with the alternative motor that we were using. However, all of the code logic works as intended in the development board, so when the PCB design is fixed, the overall product should function as intended. Another uncertainty that was faced was the fact that the physical housing was larger than intended, making sharp turns more difficult. While the requirement was still fulfilled, we intend to develop a smaller housing so that it can still encapsulate all of the components while being small enough to make the necessary turns easily. Another uncertainty that was faced was the motor speed wasn't fast enough given the 5V input. To account for this issue, we utilized a motor driver component instead of MOSFETs in the development board, after which the motor speed was high enough to fulfill the requirements.

5.3 Ethical considerations

There are a few ethical and safety considerations for this product that have been accounted for through design and functionality choices. All of these considerations have been derived in accordance with the IEEE (Institute of Electrical and Electronics Engineers) code of ethics.

One of the concerns is that the product could wake up people other than the intended user which may violate the IEEE Code of Ethics (I.1). This issue is not in control of our design as it is related to how the user uses the product. Therefore, we would warn any user to be mindful of others around them as they use this.

Another ethical and safety concern is the user may get hurt while trying to disable the alarm. This could violate the IEEE Code of Ethics (II.9). In order to prevent these issues, we have ensured that the space between the wheel and the frame is large enough to safely place a finger. In addition, the product weighs less than 4 lbs so that the user does not get injured from having a part of their body run over.

A possible safety issue that can arise from this product is the presence of batteries, which on very rare occasions could catch fire or start leaking. The likelihood of Alkaline batteries catching fire is noticeably lower than that of Lithium ion batteries, which partially served as the motivation for choosing Alkaline batteries for this system. In addition, the physical housing has been designed to ensure that there is minimal contact between the user and the batteries unless necessary.

5.4 Future work

There are quite a few considerations that can be made for future work regarding this product. First, the PCB design issues should be fixed and the PCB should be properly integrated with the product along with a smaller housing system. After that, another important integration that can be added to this system is adding a task that the user needs to complete before turning off the alarm. Given more time and resources, this should definitely be a reasonable goal to accomplish, and adding a task for the user to complete (math equation, shaking the product, etc.) would cause them to be more awake as not only are they physically engaged but also mentally engaged.

6. References

[1] “5. Addition of Sound Intensities.” Personalpages.manchester.ac.uk. [Online]. Available: personalpages.manchester.ac.uk/staff/richard.baker/BasicAcoustics/5_addition_of_sound_intensities.html
(Accessed: 4 Oct. 2024).
[2] D. Blum, “Snoozing your alarm might not be that bad for sleep, study suggests,” The New York Times, 18 Oct. 2023. [Online]. Available:
www.nytimes.com/2023/10/18/well/live/sleep-snooze-alarm-wake-up.html
(Accessed: 4 Oct. 2024).
[3] K. Deoras, “4 benefits to waking early and how to make the transition,” Summahealth.org, 2023. [Online]. Available:
www.summahealth.org/flourish/entries/2023/09/4-benefits-to-waking-early-and-how-to-make-the-transition.
[4] DIODES, “LOW DROPOUT LINEAR REGULATOR.” [Online]. Available:
https://www.diodes.com/assets/Datasheets/AZ1117C.pdf
(Accessed: 16 Apr. 2024).
[5] EPA, “Frequent questions about landfill gas.” [Online]. Available:
https://www.epa.gov/lmop/frequent-questions-about-landfill-gas
(Accessed: 30 Apr. 2024).
[6] “ESP32 vs Arduino speed comparison.” Makerguides.com. [Online]. Available:
https://www.makerguides.com/esp32-vs-arduino-speed-comparison/
(Accessed: 4 Oct. 2024).
[7] “How fast can RC cars go? | Fast RC cars.” Horizonhobby.com, 2014. [Online]. Available:
www.horizonhobby.com/blog-article-562065-how-fast-can-fast-rc-cars-go.html
(Accessed: 4 Oct. 2024).
[8] “How loud is 70 decibels | What is a 70 DB equivalent.” Decibelpro.app, 14 Sept. 2021. [Online]. Available: decibelpro.app/blog/how-loud-is-70-db/
(Accessed: 4 Oct. 2024).
[9] IEEE, “IEEE Code of Ethics.” [Online]. Available:
https://www.ieee.org/about/corporate/governance/p7-8.html
(Accessed: 16 Feb. 2024).
[10] I. D. of Health, “Food safety,” 2016. [Online]. Available:
https://dph.illinois.gov/topics-services/food-safety.html
(Accessed: 7 Feb. 2024).
[11] jamesfoster0, “Most effective hearing frequency - what you need to know.” Solid State Security, 14 May 2021. [Online]. Available:
www.solidstatesecurity.co.uk/post/most-effective-hearing-frequency-what-you-need-to-know.

[12] Johns Hopkins Medicine, “Exercising for better sleep,” 2019. [Online]. Available: www.hopkinsmedicine.org/health/wellness-and-prevention/exercising-for-better-sleep.
[13] “LDQ-N516RI | DigiKey Electronics.” DigiKey Electronics, 2015. [Online]. Available: www.digikey.com/en/products/detail/lumex-opto-components-inc/LDQ-N516RI/252641 (Accessed: 4 Oct. 2024).
[14] “Lithium vs alkaline.” Renewablesadvice.com. [Online]. Available: https://renewablesadvice.com/battery/lithium-vs-alkaline/ (Accessed: 4 Oct. 2024).
[15] National Institute on Deafness and Other Communication Disorders, “Noise-induced hearing loss,” NIDCD, 14 Dec. 2018. [Online]. Available: www.nidcd.nih.gov/health/noise-induced-hearing-loss.
[16] Olimex, “TECHNICAL DATA MQ-135 GAS SENSOR.” [Online]. Available: https://www.olimex.com/Products/Components/Sensors/Gas/SNS-MQ135/resources/SNS-MQ135.pdf (Accessed: 16 Apr. 2024).
[17] OSHA, “OSH Act: Section 5(a)(1).” [Online]. Available: https://www.osha.gov/laws-regs/standardinterpretations/2013-06-04-0 (Accessed: 16 Feb. 2024).
[18] OSHA, “OSHA 1910.141 - Sanitation.” [Online]. Available: https://www.osha.gov/laws-regs/regulations/standardnumber/1910/1910.141 (Accessed: 16 Feb. 2024).
[19] “Salary: Computer engineering in Illinois (July, 2024).” (n.d.-a). [Online]. Available: https://www.ziprecruiter.com/Salaries/Computer-Engineering-Salary--in-Illinois (Accessed: 4 Oct. 2024).
[20] Sparkfun, “Digital-output relative humidity temperature sensor/module.” [Online]. Available: https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf (Accessed: 3 Feb. 2024).
[21] Sparkfun, “Ultrasonic Ranging Module HC - SR04.” [Online]. Available: https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf (Accessed: 3 Feb. 2024).
[22] “Which is better, lidar or ultrasonic?” Indmall.in. [Online]. Available: https://www.indmall.in/which-is-better-lidar-or-ultrasonic/ (Accessed: 4 Oct. 2024).

Appendix A Requirement and Verification Table

The following Appendix section includes all of the requirements and verification tables that were used to test the functionality of the different subsystems of the product. Each subsystem has a set of requirements and clear methods to verify the effectiveness of each of the requirements.

Power Subsystem

Table 12: Power System Requirements and Verifications

Requirement	Verification	Verification status (Y or N)
The power subsystem needs to provide at least 3.3V to the ESP32 microcontroller, LED, and ultrasonic sensor	● Power on the system by connecting the 4 batteries and ensuring the voltage regulator is connected to the power subsystem and all other components. ● Use a multimeter and measure the input of the power pin in the microcontroller, LED, and sensor components and connect the other end to ground ● Record the values and verify that it's at least 3.3V	Y
To ensure continuous operation, the power subsystem needs to provide at least 5V +/- 0.5 V to the motor	● Power on the system by connecting the 4 batteries and ensuring the voltage regulator is connected to the power subsystem and all other components. ● Use a multimeter and measure the input of the power pin in the motor component and connect the other end to ground ● Record the values and verify that it's at least 5V	Y
The power subsystem needs to provide at least 2.5V to the display, push buttons, and sound system	● Power on the system by connecting the 4 batteries and ensuring the voltage regulator is connected to the power subsystem and all other components.	Y

	● Use a multimeter and measure the input of the power pin in the display, push button, and alarm components and connect the other end to ground ● Record the values and verify that it's at least 2.5V	

User Interface Subsystem

Table 13: User Interface System Requirements and Verifications

Requirement	Verification	Verification status (Y or N)
The 7-segment display must maintain a brightness of at least 125 +/ 10 lux (lux), which is obtained from typical screen brightness, to ensure visibility	● Power on the system by connecting batteries and ensuring the voltage regulator is connected to all components. ● Set a random time and ensure that the seven segment display displays the time ● Use a light meter to measure the brightness of the seven segment display and ensure that it is over 125 lx	Y
All of the push buttons must be able to read input with a maximum response time of 100 +/ 20 ms to ensure smooth functionality	● Power on the system by connecting batteries and ensuring the voltage regulator is connected to all components. ● Connect the microcontroller to all necessary components and include code to measure the time of response from the button press ● Press the time setting push buttons and refer to the print statements to collect data on response time	Y
In a 1 hour period, the clock in the system should be accurate to the nearest +/- 1 minute.	● Power all systems and ensure that the time on the seven segment display is set accurately	Y

	● Record the time set along with the time on a standard watch, including a picture ● Then, record the time after 1 hour on both the display and the watch and ensure that it is within range	
The alarm status LED must be visible from a 2 meter radius and change status within 200 +/- 20 ms of the push button press	● Power all systems, turn on the LED and stand 2 meters away and record whether it is visible ● Connect the microcontroller to all components and include code to log response time from the button press and LED change ● Press the mode setting push buttons and refer to the print statements to collect data on response time	Y

Mechanical Movement Subsystem

Table 14: Mechanical Movement System Requirements and Verifications

Requirement	Verification	Verification status (Y or N)
The vehicle must be able to move at a speed of 7 +/ 2 mph (3.13 +/- 0.83 m/s) on a flat smooth surface	● Power all systems ● Measure 10 meters (m) on the ground and set endpoints ● Once the motor system is triggered, start the stopwatch and stop recording when the vehicle has reached the endpoint ● Divide 10m by time to get the speed and repeat this for 5 trials	Y
The turning radius of the system must be at least 10 cm to ensure 90 degree turns	● Power all systems ● Place an obstacle and determine the location of turn for the vehicle ● Measure the turning path by tracing the wheel's arc and determine the radius of the turn. ● Verify that the measured radius is at least 10 cm for a complete 90-degree turn.	Y

Once the obstacle signal has been provided from the microcontroller, the wheels must turn within 500 +/ 50 ms of the signal	- Power all systems and set an obstacle for the vehicle - Connect the microcontroller to all components and include code to log response time from the sensor signal to motor turn command - Set the vehicle in front of the obstacle and refer to the print statements to collect data on response time	Y

Obstacle Detection Subsystem

Table 15: Obstacle Detection System Requirements and Verifications

Requirement	Verification	Verification status (Y or N)
The obstacle detection system shall accurately detect obstacles within 1 +/- 0.5 meters	- Power all systems - Set obstacles at 0.5m, 1m, and 1.5m in different locations - Place the vehicle in those locations and use the LED as a debug indicator to record whether the obstacle has been detected	Y
The sensor must have a field of view of 120 +/- 10 degrees to ensure large coverage for most indoor environments	- Power all systems - Set obstacles at 100, 120, and 140 degrees in different locations - Place the vehicle in those locations and use the LED as a debug indicator to record whether the obstacle has been detected	Y

Alarm Sound Subsystem

Table 16: Alarm Sound System Requirements and Verifications

Requirement	Verification	Verification status (Y or N)

The speaker must produce a sound with a volume of at least 75 dB +/- 5 decibels (dB) at a distance of 1 meters from the device	● Power on the system and ensure that the voltage regulator is connected to all components. ● Activate the alarm speaker system and place a decibel meter 1m away from the system ● Use the meter to measure the sound level of the alarm and ensure that it is around 75 dB	Y
The alarm shall stop within 200 +/- 20 ms of the user pressing the button to deactivate it	● Power all systems and set an obstacle for the vehicle ● Connect the microcontroller to all components and include code to log response time from the push button press to sound system deactivation ● Press the push button and refer to the print statements to collect data on response time	Y
The alarm sound must have a frequency between 2000 and 4000 Hz to match an attention grabbing alarm sound.	● Power on the system and ensure that the voltage regulator is connected to all components. ● Activate the alarm speaker system and place a spectrum analyzer 1m away from the system ● Use the device to measure the sound frequency and ensure that it is between 2000 and 4000 Hz	Y

Physical Housing Subsystem

Table 17: Physical Housing System Requirements and Verifications

Requirement	Verification	Verification status (Y or N)
The physical housing system must be able to protect internal components with no more than 2 centimeters (cm) of surface deformation after being dropped from a height of 1m	● Use a secondary 3d printed housing and drop it from the 1m height ● Measure any visible damages using a ruler and ensure that it is 2 cm or less ● Repeat the trial to ensure accuracy	Y

After the internal components are mounted, the 3D printed housing must weigh no more than 3000 +/- 10 grams (g)	- Ensure that all of the components are mounted within the housing - Place the vehicle in a weighing scale and record the value in g	Y
The wheels of the vehicle should be able to navigate at least 1m of surface with water	- Power all systems and pour a thin layer of water in a 1m path - Let the vehicle traverse the path and record whether it has successfully crossed	Y

Microcontroller Subsystem

Table 18: Microcontroller System Requirements and Verifications

Requirement	Verification	Verification status (Y or N)
The ESP32 Microcontroller must execute major operations (time set, alarm activation, motor movement) within +/- 10 ms for each task (time specified in each subsystem)	- Power on the system and ensure that the voltage regulator is connected to all components. - Verify the response time for components specified in earlier tests and ensure that all of them fall within range	Y
After performing core tasks for 5 minutes, the temperature of the microcontroller temperature must not exceed 70 +/- 5C	- Power all systems and connect the microcontroller - Let the system perform the time setting, alarm sound, and motor tasks for 5 minutes - Use a calibrated infrared thermometer and record the temperature	Y

Appendix B Circuit Schematic and PCB Design

The following sections include the full circuit schematic and the PCB design with the routing.

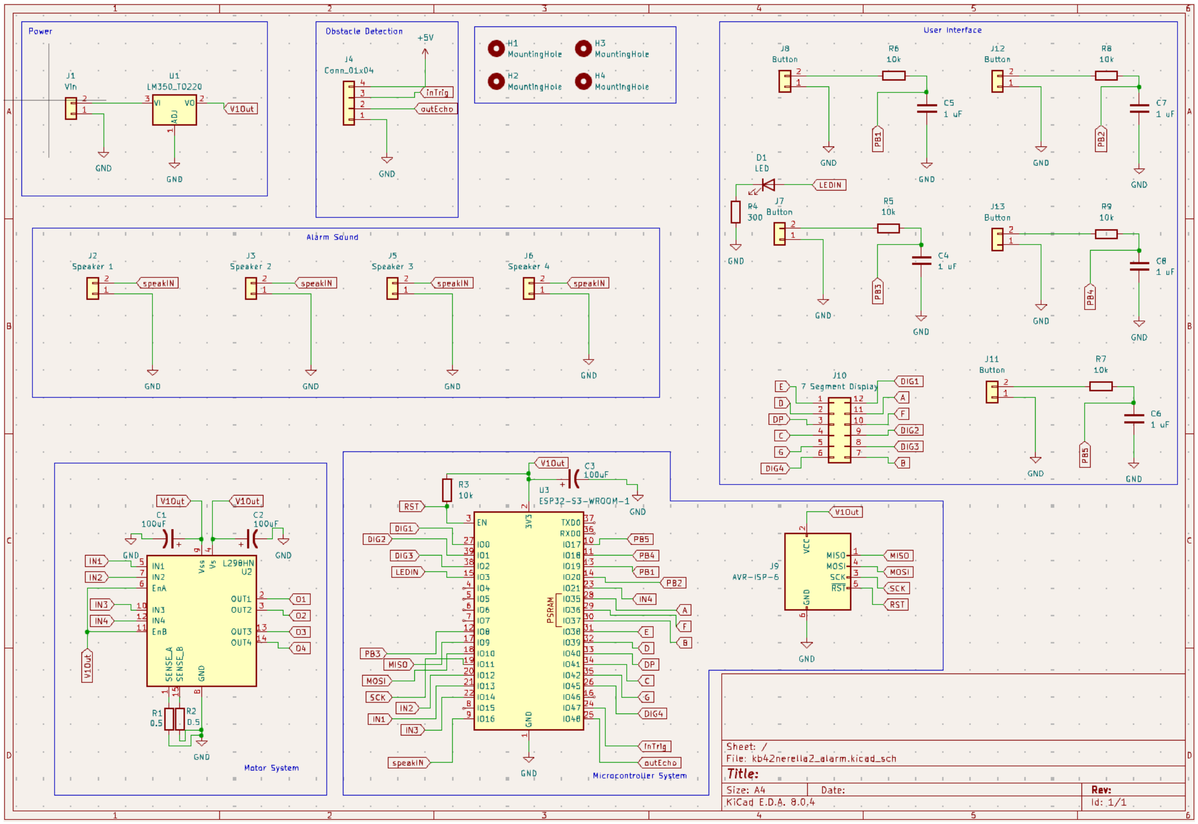

Figure 9: Schematic of the PCB Design

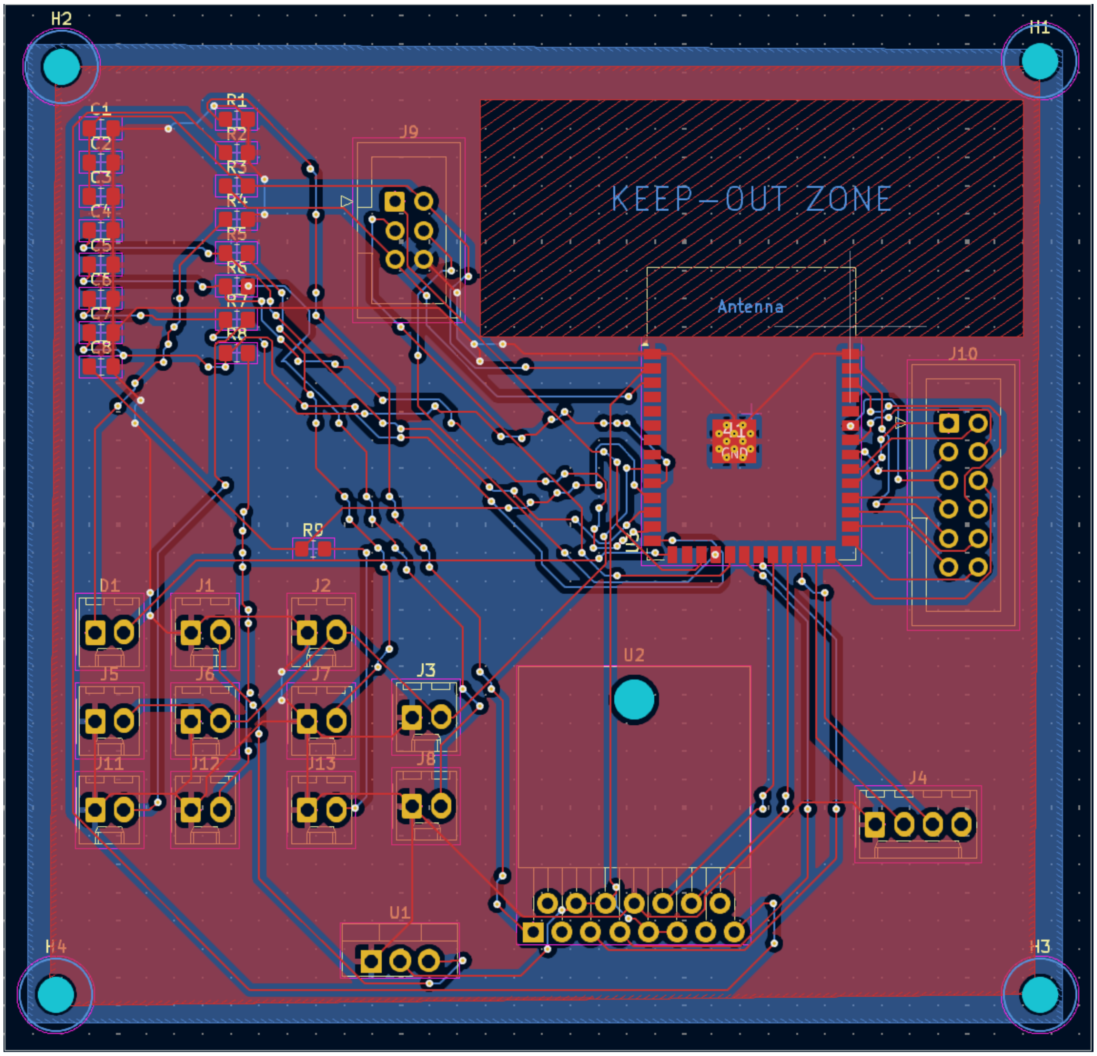

Figure 10: Final Design of the PCB

Appendix C Microcontroller Code

```
#include <TM1637Display.h>



#include <Wire.h>
#include <RTClib.h>



#define dio_pin 13
#define clk_pin 12
u_int32_t seconds;
u_int32_t minutes;
u_int32_t hours;
u_int32_t alarmMinutes;
u_int32_t alarmHours;
bool timekeeping;
bool measuring;
bool changeAlarm;
bool changeTime;
bool alarmSet;
bool beep;
bool turn;
bool sound;
bool alarmAM;
bool realAM;
bool distWait;
unsigned long lastMillis;
unsigned long b1Millis;
unsigned long b2Millis;
unsigned long b3Millis;
unsigned long b4Millis;
unsigned long b5Millis;
unsigned long testMillis;
unsigned long motorMillis;
unsigned long curMillis;
unsigned long distMillis;
unsigned long distTime;
unsigned long turnMillis;
int displayTime;
TM1637Display display(clk_pin, dio_pin);

void setup() {
 Serial.begin(9600);
 b1Millis=millis();
 b2Millis=millis();
 b3Millis=millis();
 b4Millis=millis();
 b5Millis=millis();
 motorMillis=millis();
 distMillis=millis();
```

```
 distTime=millis();
 turnMillis=millis();
 testMillis=millis();
 timekeeping=true;
 sound=false;
 alarmAM=true;
 realAM=true;
 hours=1;
 minutes=0;
 alarmHours=1;
 alarmMinutes=1;
 seconds=50;
 alarmSet=true;
 distWait=false;
 measuring=false;
 turn=false;
 lastMillis=millis();
 pinMode(2,OUTPUT);  //Speaker
 pinMode(35,INPUT);  //Off Button
 pinMode(32,INPUT);  //Hours Button
 pinMode(21,INPUT);  //Minutes Button
 pinMode(14,INPUT);  //Alarm set button
 pinMode(17,OUTPUT); //LED
 pinMode(34,INPUT);  //Change Alarm
 pinMode(18,OUTPUT); //Ultrasonic Echo
 pinMode(19,INPUT);  //Ultrasonic Trigger
 pinMode(13,OUTPUT); //Display pins
 pinMode(12,OUTPUT); //Display pins
 pinMode(26,OUTPUT);
 pinMode(25,OUTPUT);
 pinMode(27,OUTPUT);
 pinMode(33,OUTPUT);

 beep=false;
 digitalWrite(17,HIGH);
 display.setBrightness(7);
 displayTime=hours*100+minutes;
 display.showNumberDecEx(displayTime,0b01000000);
 }
void rightTurn(){
 digitalWrite(33,HIGH);
 digitalWrite(27,LOW);
 digitalWrite(26,LOW);
 digitalWrite(25,LOW);
}
void leftTurn(){
 digitalWrite(33,LOW);
 digitalWrite(27,LOW);
 digitalWrite(26,LOW);
 digitalWrite(25,HIGH);
}
void straight(){
```

```
 digitalWrite(33,HIGH);
 digitalWrite(27,LOW);
 digitalWrite(26,LOW);
 digitalWrite(25,HIGH);
}
void randomTurn(){
 turn=true;
 turnMillis=millis();
 if(random(0,10)>5){
   rightTurn();
 }
 else{
   leftTurn();
 }
}
void loop() {
 if(timekeeping){
   if(millis()-lastMillis>=1000){
     lastMillis=millis();
     if(!changeAlarm){
       Serial.printf("%d:%02d:%02d\n",hours,minutes,seconds);
     }
     seconds++;

     if(seconds>=60){
       minutes++;
       if(minutes>=60){
         hours++;
         if(hours>12){
           hours=1;
           realAM=!realAM;
         }
         minutes=0;
       }
       seconds=0;
       if(alarmSet && hours==alarmHours && minutes==alarmMinutes && alarmAM==realAM &&
seconds==0){
         beep=true;
         straight();
         motorMillis=millis();
       }
       displayTime=hours*100+minutes;
       display.showNumberDecEx(displayTime,0b01000000);

     }
   }
 }
 else if(changeTime){
   if(digitalRead(35)){

     seconds=0;
     Serial.printf("Time changed");
```

```
    changeTime=false;
    timekeeping=true;
  }
  if(digitalRead(21)){
    if(millis()-b4Millis>300){
      b4Millis=millis();
      minutes++;
      if(minutes>=60){
        minutes=0;
      }
      Serial.printf("New Time: %d:%02d\n", hours,minutes);
      displayTime=hours*100+minutes;
      display.showNumberDecEx(displayTime,0b01000000);
    }
  }
  if(digitalRead(32)){
    if(millis()-b5Millis>300){
      b5Millis=millis();
      hours++;
      if(hours>12){
        hours=1;
        realAM=!realAM;
      }
      Serial.printf("New Time: %d:%02d\n", hours,minutes);
      displayTime=hours*100+minutes;
      display.showNumberDecEx(displayTime,0b01000000);
    }
  }
}
if(changeAlarm){
  if(digitalRead(21)){
    if(millis()-b4Millis>300){
      b4Millis=millis();
      alarmMinutes++;
      if(alarmMinutes>=60){
        alarmMinutes=0;
      }
      Serial.printf("Alarm Time: %d:%02d\n", alarmHours,alarmMinutes);
      displayTime=hours*100+minutes;
      display.showNumberDecEx(alarmHours*100+alarmMinutes,0b01000000);
      Serial.printf("Time to change Minutes: %f ms \n",millis()-b4Millis);
    }
    if(digitalRead(35)){
      Serial.printf("Alarm changed");
      changeTime=false;
      changeAlarm=false;
      timekeeping=true;
      displayTime=hours*100+minutes;
      display.showNumberDecEx(displayTime,0b01000000);
    }
  }
  if(digitalRead(32)){
```

```
    if(millis()-b5Millis>300){
      b5Millis=millis();
      alarmHours++;
      if(alarmHours>12){
        alarmHours=1;
        alarmAM=!alarmAM;
      }
      Serial.printf("Alarm Time: %d:%02d\n", alarmHours,alarmMinutes);
      display.showNumberDecEx(alarmHours*100+alarmMinutes,0b01000000);
      Serial.printf("Time to change hours: %f ms \n",millis()-b5Millis);

    }
  }
}
if(beep){
  if(!sound){
    sound=true;
    // tone(2,1000);
    digitalWrite(2,HIGH);
  }
  //make motor go
  if(!distWait || millis()-distTime>3000){ //measure distance
    distWait=true;
    measuring=false;
    distTime=millis();
    digitalWrite(18,HIGH);
    delayMicroseconds(10);
    digitalWrite(18,LOW);
  }
  if(!measuring && digitalRead(19)){
    distMillis=micros();
    measuring=true;
  }
  if(turn && (millis()-turnMillis)>1200){
    turn=false;
    straight();
    motorMillis=millis();
  }
  if((millis()-motorMillis>3000 || !digitalRead(19) && measuring)&&!turn){ // or ultrasonic
sensor detects something
    testMillis=millis();
    if((0.0344*(micros()-distMillis)/2)<30.0){
      Serial.printf("obstacle detected");
    }
    if((!digitalRead(19) && (0.0344*(micros()-distMillis)/2)<10.0) ||
millis()-motorMillis>3000)
    {
      motorMillis=millis();
      randomTurn();
    }
    Serial.printf("Time to start turning: %f ms \n",millis()-testMillis);
    distWait=false;
```

```
      measuring=false;
    }
    if(digitalRead(35)){
      b1Millis=millis();
      beep=false;
      // noTone(2);
      digitalWrite(2,LOW);
      digitalWrite(26,LOW);
      digitalWrite(25,LOW);
      digitalWrite(33,LOW);
      digitalWrite(27,LOW);
      sound=false;
      Serial.printf("Sound Turn Off: %f ms \n", millis()-b1Millis);
    }

  }

  if(digitalRead(14)){
    if(millis()-b2Millis>300){
      b2Millis=millis();
      alarmSet=!alarmSet;
      digitalWrite(17,alarmSet);
      Serial.printf("Alarm Status Button: %f ms \n", millis()-b2Millis);
    }
  }
  if(digitalRead(34)){
    if(millis()-b3Millis>300){
      b3Millis=millis();

      changeAlarm=!changeAlarm;
      if(changeAlarm){
        display.showNumberDecEx(alarmHours*100+alarmMinutes,0b01000000);
      }
      else{
        display.showNumberDecEx(hours*100+minutes,0b01000000);
      }
      changeTime=false;
      timekeeping=true;
      Serial.printf("Change alarm button: %f ms \n",millis()-b3Millis);
    }
  }
  if(digitalRead(21) && digitalRead(32)){
    if(millis()-b4Millis>300 && millis()-b5Millis>300){
      b4Millis=millis();
      b5Millis=millis();
      changeAlarm=false;
      changeTime=!changeTime;
      timekeeping=!timekeeping;
    }
  }

}
```

Appendix D Subsystem Schematics

The following appendix section includes schematic diagrams of all of the subsystems in the overall system. Each figure description contains the name of the subsystem the schematic belongs to.

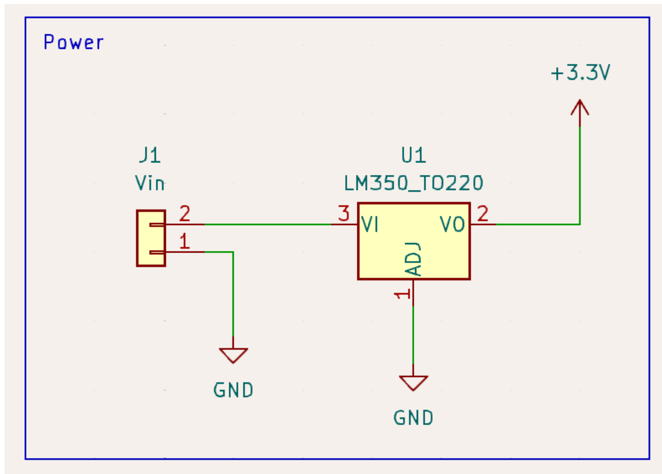

Figure 11: Schematic of the Power System

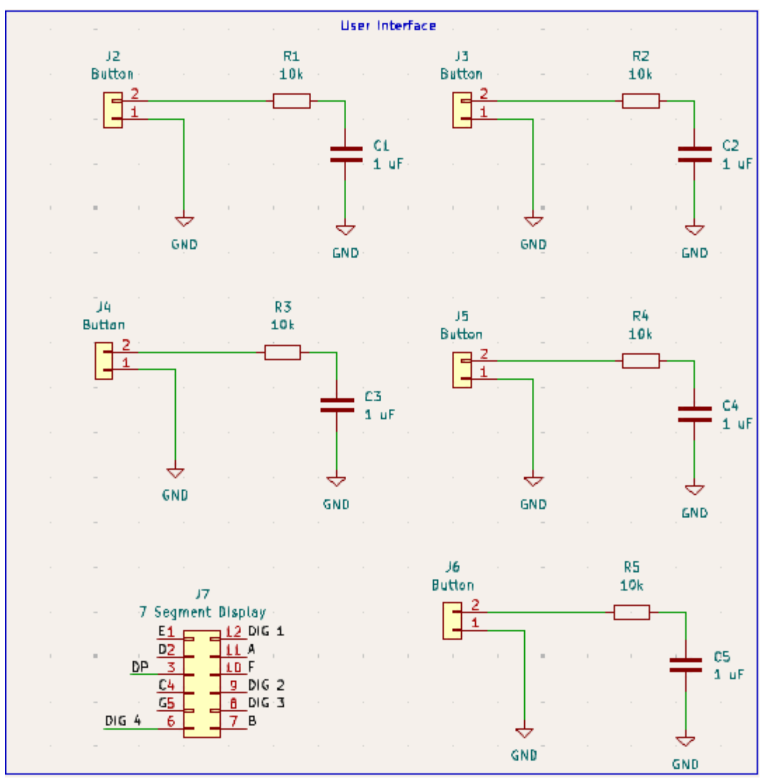

Figure 12: Schematic of the UI System

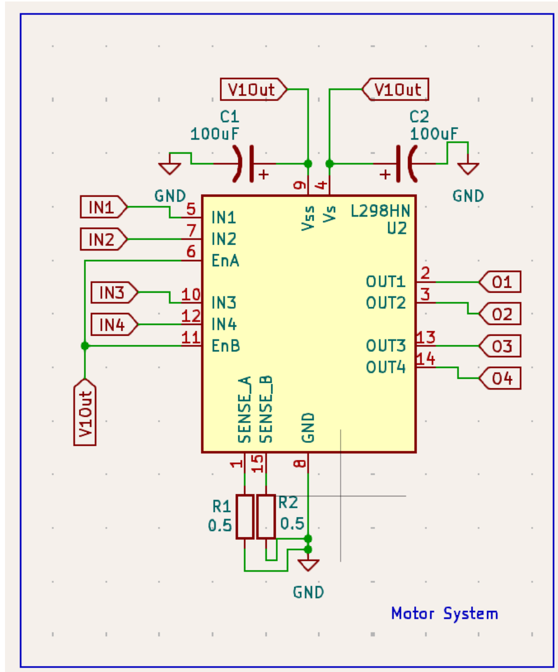

Figure 13: Schematic of the Motor System

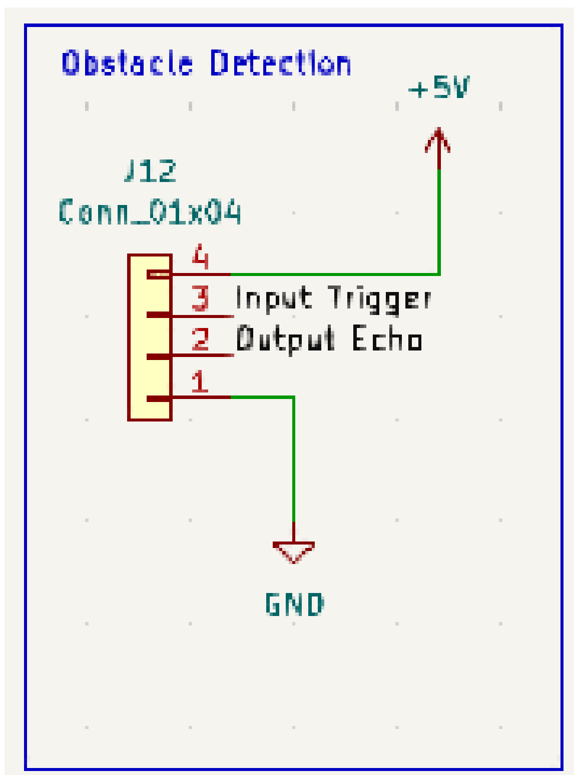

Figure 14: Schematic of the Obstacle Detection System

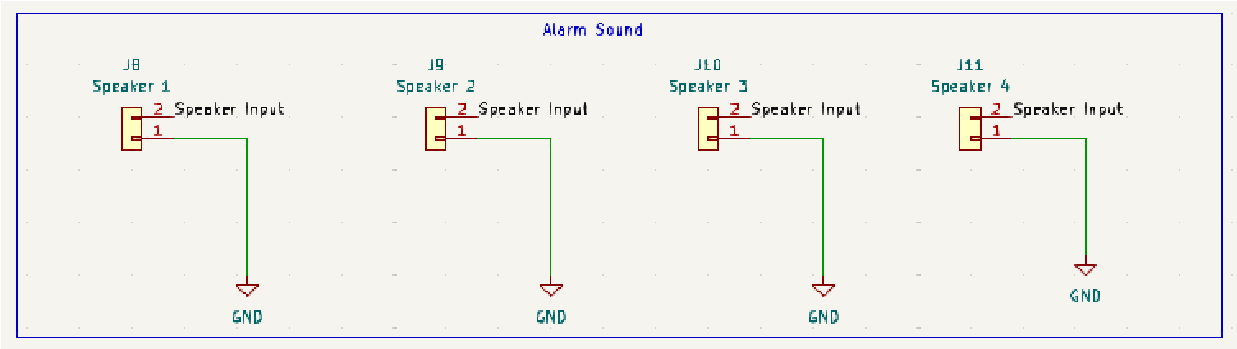

Figure 15: Schematic of the Alarm System

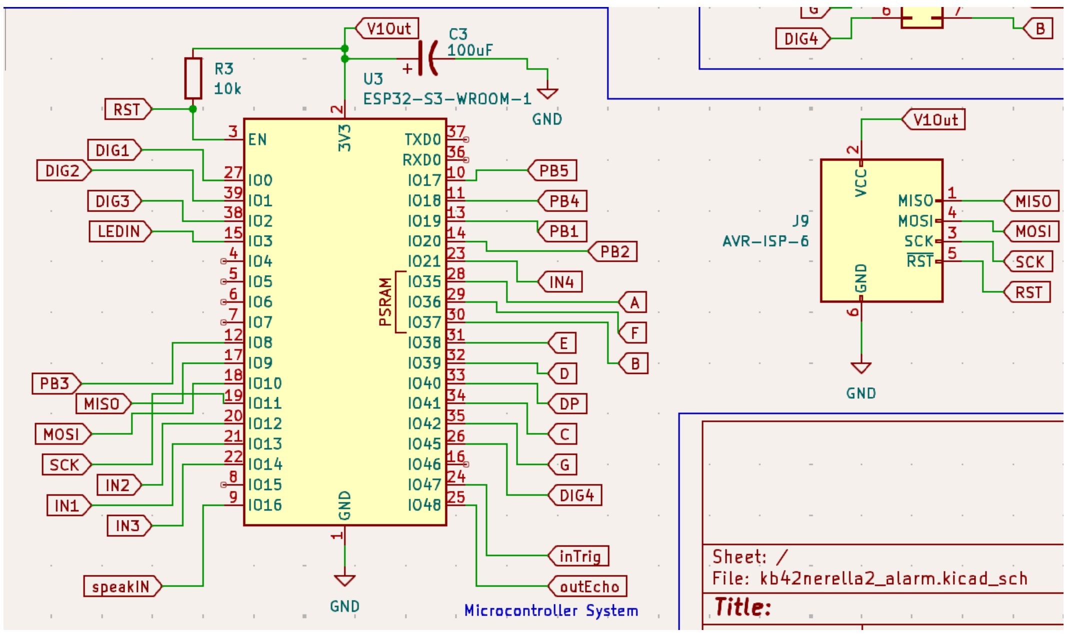

Figure 16: Schematic of the Microcontroller System