Linnéuniversitetet
Kalmar Växjö

Project with Embedded System

Hen House

Authors: Fardin Beg and Ghiath Al Omari
Course Responsible: Fredrik AHLGREN
Supervisor: Hemant GHAYVAT
Supervisor: Mehdi SAMAN AZARI
Semester: 24VT
Course code: 2DT304

Abstract

This project presents the development and implementation of the Hen House, an automated chicken coop designed to improve the quality of life for poultry and ease the workload for farmers. Utilizing FreeRTOS and Raspberry Pi Pico, the Hen House automates essential functions such as door operation according to the day-night cycle, temperature regulation with heat lamps, and predator alerts through motion sensors. The project was motivated by the challenges of poultry farming, including predator threats and the need for constant environmental monitoring. Through a series of design iterations and testing, the Hen House has demonstrated significant benefits in operational efficiency and poultry safety. This report details the project's conception, design challenges, methodology, and the achieved outcomes, showcasing a cost-effective solution to modernize poultry farming with limited resources.

Keywords:

- Poultry Farming
- Automated Chicken Coop
- FreeRTOS
- Raspberry Pi Pico
- Smart Agriculture
- Environmental Monitoring
- Predator Detection
- IoT in Farming

Contents

List of Figures

1 Introduction/Background **1**
1.1 Research aims/questions ……. 1
1.2 Method/Methodology ..... 1 - 2
1.3 Results/Findings …….. 2

2 Related Work …… 3

3 Hardware ……. 4 - 9

4 Software ……. 10 - 11

5 Results and Discussion ……. 12
5.1 Achieved Outcomes: 11- 16
5.2 Implementation: .....17 - 18

6 Conclusion 19

References 20

Appendix 1.1 A 21

Appendix 1.2 B 21

List of Figures

3.1 Rasberry Pi Pico. 4
5.1,1 The light sensor. 12
5.1,2 Heat Lamp. 13
5.1,3 Temp sensor inside. 13
5.1,4 Temp sensor outside. 14
5.1,5 IR sensor. 15
5.1,6 Buzzer and RGB LED. 15
5.1,7 OUTUT PUTTY. 16
5.2,1 DC motor. 17
5.2,2 Battery. 17

1 Introduction/Background

Imagine a chicken house that takes care of itself, making life easier for farmers. That's what we aimed to create with Hen House. It's not just any chicken coop; it's a smart one designed to tackle the everyday challenges of poultry farming head-on. Our goal was to make farmers' lives a bit less stressful by keeping chickens safe from predators and harsh weather, without needing constant human oversight.

In this report, we're taking you on a journey from the initial spark of an idea to the Hen House we've built. We'll walk you through why we started this project, how we designed the Hen House, the hurdles we overcame during testing, and wrap up with what we've learned and where we're headed next.

This whole adventure started with a simple story from a friend, which got us thinking about how we could make poultry farming less of a hassle. Of course, we dreamed big but had to navigate through real-world challenges like budget limits and tech hurdles. These didn't stop us; they just helped us zoom in on what was really doable and impactful.

1.1 Research aims/questions

When we started our project, we knew we'd face challenges, but we didn't anticipate just how many or how complex they would be. The biggest hurdle? Choosing the right environment to work in, which ended up being FreeRTOS. This choice changed everything for us, from the code we wrote to the hardware we used.

Getting FreeRTOS up and running was a project all on its own. It was super picky – if we moved or deleted the wrong file, we'd end up with errors that took forever to fix. This part of the project really tested our patience and problem-solving skills.

Then came the decision between using a Raspberry Pi Pico and an Arduino. Both are great in their own ways, but we needed to figure out which one was the best fit for our project. This meant not just testing them out, but also changing our hardware setup based on our choice. It was a lot of trial and error, but it was also pretty exciting to see how each decision we made affected our project.

1.2 Method/Methodology

At the start of our project, we had to choose the best software and hardware. We needed something that worked well and was reliable. After thinking it

over, we decided on FreeRTOS and the Raspberry Pi Pico.

FreeRTOS is a system made for small, low-power devices that makes everything from programming to managing easier. It was perfect for what we needed. The Raspberry Pi Pico, especially the W version, was great because it was affordable and had good speed, Wi-Fi, and Bluetooth. Even though Arduino boards have similar features, they were too expensive. So, the Pico was our best option.

We also carefully picked out different components for our project, like temperature sensors and door mechanisms. We tested a lot to find the best ones that fit our project well. This helped us make sure everything worked together smoothly.

In the end, our project came together with all the right pieces. Every choice we made, big or small, was aimed at making our project work well and show how carefully we thought about our options

1.3 Results/Findings

Our project wasn't just about getting something to work. It was about learning – figuring out how to navigate through all these challenges and understanding the science behind our choices. We dove deep into why certain microcontrollers worked better with FreeRTOS, and what that meant for our project. It was a journey filled with obstacles, but also with a ton of valuable lessons. Simply we planned an automatic Hen house that keeps the birds safe and warm meanwhile helping farmers put their focus elsewhere and that is what we achieved at the end of this project. So we achieved what we planned.

2 Related Work

A company that is called Coop has launched an AI-powered smart home for chickens, making poultry farming easier and safer. Named Albert Eggstein, this AI system can count eggs and watch out for predators. The setup includes a chicken house with a wire fence, lights, and cameras, all controlled remotely with just Wi-Fi and a small outdoor area. This innovation makes caring for chickens more interactive and easier for both beginners and experienced farmers. [1]

Smart AG is another revolutionizing traditional farming with IoT technology, using an Arduino device with sensors for temperature and humidity, connected through a Soracom SIM card. This setup offers real-time environmental data inside chicken coops, emphasizing the importance of controlling the microclimate for poultry health and preventing losses from bad weather. Additionally, Smart AG is testing wearable IoT devices for tracking livestock and sensors to manage waste, presenting a comprehensive farming solution focused on animal welfare and environmental care. [2]

Naturally, it's clear that there are more sophisticated and advanced iterations of this project out there. Several companies have already perfected this concept, incorporating features like LCD screens, the ability to send signals to a specially designed app, and engineering compact, easily manageable boxes that are water-resistant and durable against all kinds of weather.

In comparison, our version stands as a more budget-friendly alternative. While it may not boast the complexity and advanced features of these commercial models, it represents a simplified approach that prioritizes accessibility and cost-effectiveness. Our project may lack some of the high-end functionalities, but it serves as a testament to what can be achieved with limited resources and a focus on the essentials.

3 Hardware

This project stands out because of how flexible it is. You can use many types of components for it, not just a Raspberry Pi Pico W; an Arduino can work too. After trying out different components, we picked the best ones for our goals.

Embedded Board Descriptions:

- RP2040 microcontroller chip designed by Raspberry Pi in the United Kingdom
- Dual-core Arm Cortex M0+ processor, flexible clock running up to 133 MHz
- 264kB of SRAM, and 2MB of on-board flash memory
- USB 1.1 with device and host support
- Low-power sleep and dormant modes
- Drag-and-drop programming using mass storage over USB
- 26 × multi-function GPIO pins
- 2 × SPI, 2 × I2C, 2 × UART, 3 × 12-bit ADC, 16 × controllable PWM channels
- Accurate clock and timer on-chip
- Temperature sensor
- Accelerated floating-point libraries on-chip
- 8 × Programmable I/O (PIO) state machines for custom peripheral support
- Datasheet [3]

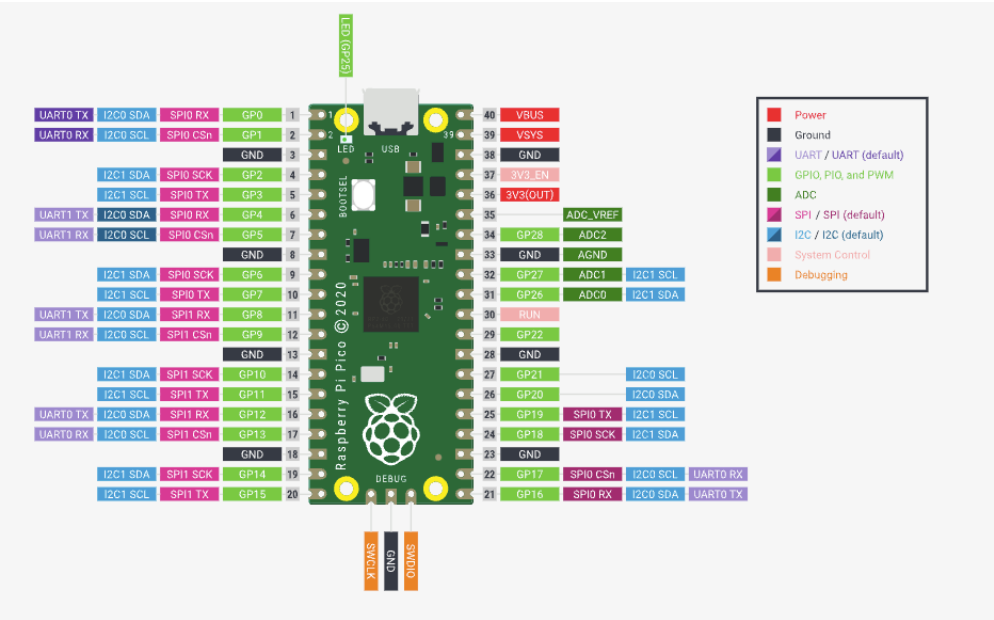

Figure 3.1: The Rasberry Pi Pinout.

I/O Capabilities:

- **GPIO Pins:** The Pico provides 26 multi-function GPIO pins, which are highly configurable and support capabilities like UART, I2C, SPI, and more directly. This flexibility allows for a wide range of peripherals and custom setups.

- **Analog Inputs:** There are three ADC (Analog to Digital Converter) channels available on the Raspberry Pi Pico, enabling the board to read analog signals, which is essential for a broad array of sensor integrations.

- **PWM Outputs:** The RP2040 supports PWM outputs across multiple GPIO pins, facilitating precise control over actuators and other devices.
  USB Connectivity: One of the standout features of the Raspberry Pi Pico is its support for USB 1.1 Host and Device, enabling direct connections to computers and other USB devices without needing additional serial converters.

Raspberry Pi Pico marks a big leap forward in computing programming, and the massive variety of possible projects that you can do. With its dual-core processor, enough memory, and thorough input/output capabilities, the Pico is ready for complicated tasks that demand simultaneous operations or taking care of large volumes of data. The Raspberry Pi Pico is a formidable and flexible platform, ideal for educational projects, for people with hobby for embedded system, or creating professional prototypes. Raspberry Pi Pico Datasheet [3].

Applications that operate in real-time need to process data and react within a precise, exact time window, a very important thing in areas such as robotics, industrial automation, automotive technology, and other embedded systems. Traditional operating systems may fall short in these events as they aim to enhance overall performance without offering assurances on response times. As a result, Real-Time Operating Systems (RTOS) and modifications to Linux for real-time operations (RT Linux) play a big role for devices like the Raspberry Pi, making sure they meet the exact requirements of real-time functionality.

Dedicated Real-Time Operating Systems (RTOS) are specifically made to support real-time events from their beginning. Unlike other operating

systems like Linux, they are streamlined to ensure quick response times and consistent scheduling of tasks. For Raspberry Pi devices, there exists a variety of RTOS choices that can be made for real-time applications' precise demands.

One such example is FreeRTOS, a widely known, efficient RTOS that has been adapted for various microcontroller architectures, including the ARM Cortex-M0/M3/M4. These are relevant to the Raspberry Pi Pico and potentially other Raspberry Pi models. FreeRTOS stands out for its simplicity, ease of comprehension, and its capability to be minimized to a small size, making it ideal for straightforward real-time operations.

Real-time operating systems (RTOS) are mainly used in simpler, time-sensitive projects where a big and complicated operating system would be too much. Typical use cases include real-time tracking of sensor data, robotic motor management, and other embedded applications that demand precise timing in control loops.

Input Devices:

- **Temp sensor (KY-013 analog sensor):**
  - Operating voltage 3,3 V - 5 V
  - Measuring range -55 °C to +125 °C
  - Measurement accuracy ± 0,5 °C
  - Known resistance 10 kΩ
  - Specific resistance of the NTC 3950 Ω

This module has a type of sensor called an NTC thermistor. It can measure temperatures from -55°C to +125°C. As the temperature increases, the resistance of the sensor decreases. By looking at how the resistance changes, we can figure out the temperature. KY-013 Datasheet [4]

- **Light Sensor (TEMT6000):**
  - Package type: surface mount.
  - Package form: 1206 18527
  - Dimensions (L x W x H in mm): 4 x 2 x 1.05

- AEC-Q101 qualified
- High photo sensitivity
- Adapted to human eye responsivity
- Angle of half sensitivity: □ = ± 60°
- Floor life: 168 h, MSL 3, acc. J-STD-020
- Lead (Pb)-free reflow soldering
- Compliant to RoHS Directive 2002/95/EC and in accordance to WEEE 2002/96/EC
- Specific resistance of the NTC 3950 Ω

The TEMT6000X01 ambient light sensor is a small, transparent phototransistor in a 1206 package designed for surface mounting. It functions similarly to the human eye, being sensitive to visible light, with its highest sensitivity at 570 nm. Datasheet TEMT6000 [6].

- **Motion Sensor (IR transmitter):**
  - Ambient light protected IR receiver
  - pin easy interface connectors
  - Indicator LED & Power LED
  - Distance 2cm to 30cm
  - Can differentiate between dark and light colours
  - Active Low on object detection
  - 3.3 to 5V operation

Proximity sensors are utilized to identify objects and obstacles positioned in front of the sensor.
The sensor emits infrared light continuously, and

when an object approaches, it detects it by observing the reflected light from the object. These sensors find applications in various fields such as robotics for avoiding obstacles, automatic doors, parking assistance devices, security alarm systems, and contactless tachometers for measuring the RPM of rotating objects such as fan blades.
Datasheet IR Sensor.[7].

Output Devices:

- **DC Motor:**
  - Direct current motor
  - Voltage 18VDC
  - Current 1.6A
  - Speed 13000rpm
  - Power 20W

High-speed DC motor (max. 12500 RPM under load).
Operates on 2.5-6 V (DC) power supply.
Rotation direction is determined by polarity reversal.
Maximum current consumption: 1.34 A. Torque: 46.2 g/cm.
Solder connections. Motor dimensions: Ø24x45 mm (including shaft).
Shaft dimensions: Ø2x12 mm.
Datasheet DC Motor. [5]

- **LED RGB:**
  - Low Thermal Resistance
  - No UV rays
  - Forward Current for Red, Blue, and Green color: 20mA
- **LED:**
  - 1.8-2.2VDC forward drop
  - Max current: 20mA
  - Suggested using current: 16-18mA

- **Buzzer:**
  - Black in colour
  - With internal drive circuit
  - Sealed structure
  - Wave solderable and washable

Datasheet Buzzer. [8]

Optimizing power consumption for the Hen House project involves leveraging the Raspberry Pi Pico's sleep modes, selectively activating peripherals only when needed, and using interrupt-driven programming for sensors to minimize energy use.

Dynamic power management within FreeRTOS can adjust the microcontroller's operation based on task requirements, significantly reducing power draw.

Efficient communication protocols for optional remote monitoring, feedback loops for actuators like the DC motor, and the potential integration of solar panels offer further energy savings.

Implementing a smart battery management system for battery-powered setups can optimize energy costs and extend battery life. These measures collectively enhance the project's energy efficiency, ensuring it operates sustainably and effectively in automated poultry care.

4 Software

Programming language: for this project we chose to use C programming language as it fast with compiling code and it is the most embedded systems programming is done with either C or C++. It is also very suitable to integrate with FreeRTOS and our Micro controller raspberry pi Pico.

Comparing the programming language that we have used which is C with if we have done the same using Assembly language, Firstly, C language provides more readable and maintainable compared to Assembly language. Which enables us to easily understand the logics and make modifications.

Assembly language requires deep understanding of hardware architecture and can be hard to read and maintain.

Assembly language needs to manually manage memory, registers and hardware peripherals, because of that debugging in assembly language more challenging and time-consuming.

So, taking what we mentioned above despite Assembly language can be faster for such project where real time is needed it makes the work more time consuming and more complex.

In our case to introduce real time to our project we went with FreeRTOS as our real time operating system. As we are working with FreeRTOS we have used scheduling system to make sure that every component takes its turn without having to interfere with each other. FreeRTOS allows to define task priorities and manage task execution, ensuring that critical tasks are given precedence over less critical ones. In case of unexpected large delay potentially some components might not respond promptly to changes in the environment, leading to suboptimal performance or incorrect decisions.

Taking security in consideration there are several security factors we can look at. Starting with cyber security: our device is less likely to get any cyber attacks because our device operates locally, e.g. no internet connection or so what. Also, there is physical security: our device is now very safe to physical tampering. There is also data security: because we don't have any use credential or private data so there is no concern about losing data or unauthorized access. Lastly, we will talk about theft: in our device we have installed a motion sensor with an alarm system to detect

any unusual motion near to the Hen House border. In this way we prevent any theft activities.

Some code we found on internet, but it was in python, so we had to take the idea of the implementation and translate it to C and used it to our project. We have not used any external libraries; all our libraries are integrated with FreeRTOS and Pico. Other codes we have used their data sheets to know how they work, and we have used our experiences from our previous course.

The key features of FreeRTOS include:

- Task management (Multitasking and Priorities)
- Memory Management
- Inter-Task Communication and Synchronization
- Software Timers
- Low Power Consumption
- Safety and Reliability

5 Results and Discussion

In this chapter, we dive into what we accomplished with our Hen House project and what we learned along the way. Our goal was to make a smart chicken coop to help poultry farmers by using automated systems and monitoring. We'll first talk about the actual results we got, like how well the system worked on a farm. After that, we'll discuss what these results mean, looking at them in the bigger picture of using technology in farming.

5.1 Achieved Outcomes:

From the beginning of this project, we set out with a clear goal in mind: to deliver an automated Hen House that enhances the safety and comfort of chickens while easing the workload for farmers. We're proud to say we've achieved that. Our innovative Hen House ensures that chickens are kept safe and warm, allowing farmers to shift their focus to other important tasks around the farm without constant worry about their poultry.

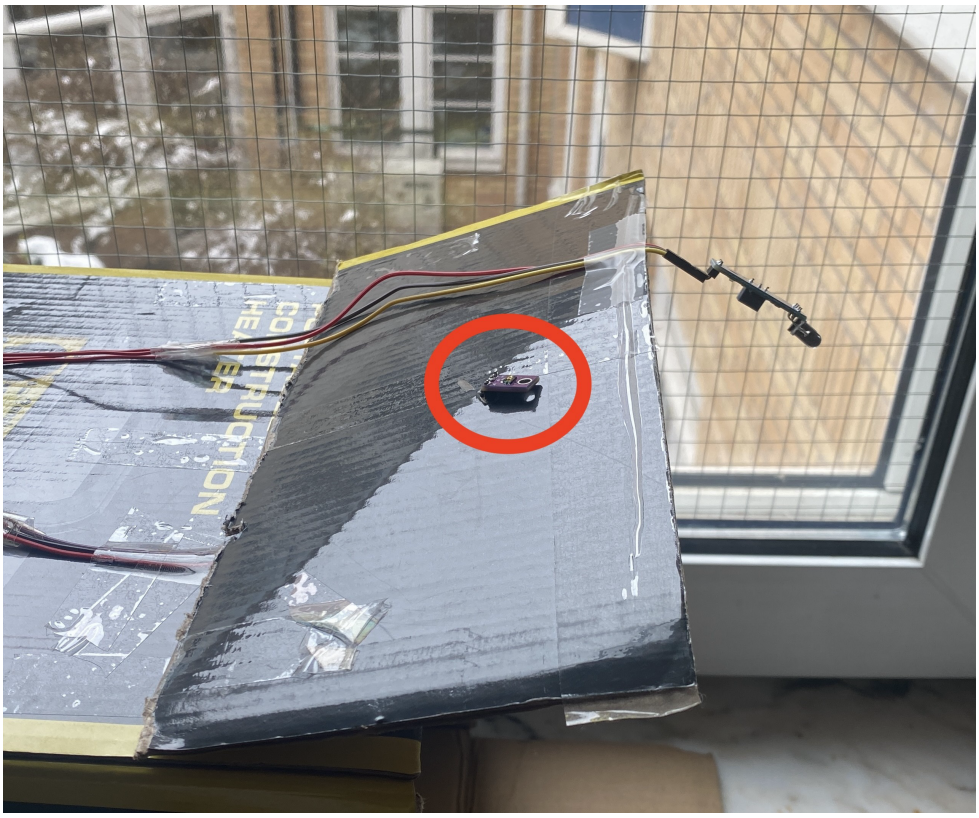

Figure 5.1,1: The light sensor.

The light sensor as shown in Figure 5.1,1 on the top our door, our system smartly integrates with the natural cycle of day and night. The Hen House door automatically opens with the morning light, welcoming the new day, and closes as darkness falls, ensuring all chickens are safely tucked inside for the night. This feature alone significantly reduces the risk of predator attacks during vulnerable hours.

Figure 5.1,2: Heat Lamp.

Figure 5.1,3: Temp senor inside.

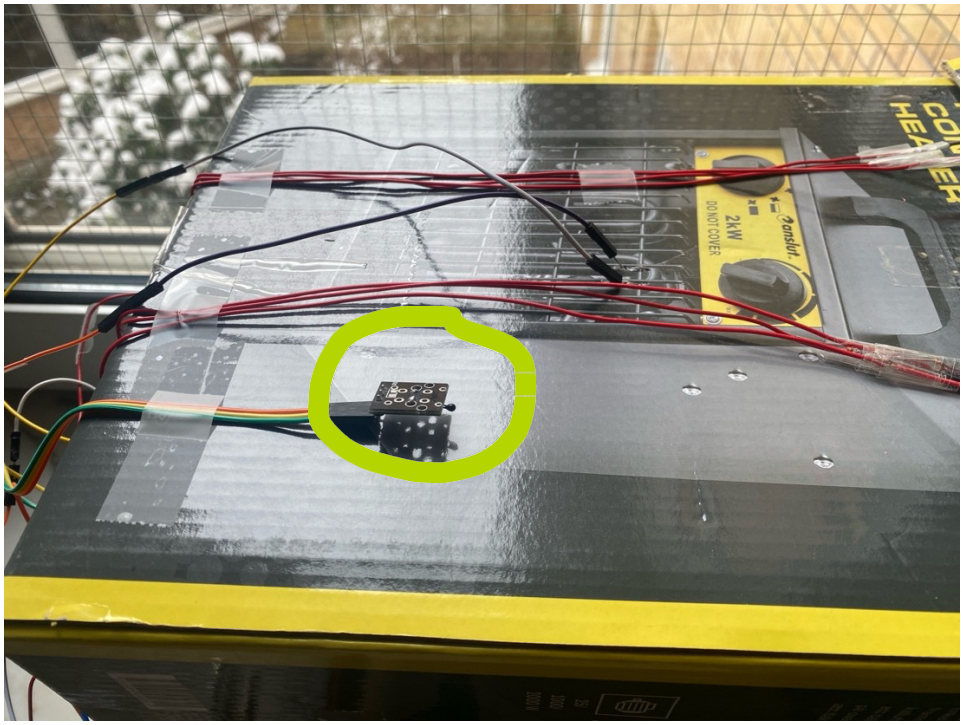

Figure 5.1,4: Temp senor outside.

But we didn't stop there. Recognizing the importance of maintaining an optimal environment, we incorporated heat lamp as shown in figure 5.1,2 and equipped with sensors to monitor both the internal shown in figure 5.1,3 and external temperature sensor as shown in figure 5.1,4. The heat lamp turns onto provide warmth when the temperature drops inside the coop and turn off to avoid overheating during warmer conditions. This intelligent temperature management ensures the chickens remain comfortable regardless of the weather outside.

Figure 5.1,5: IR senor outside.

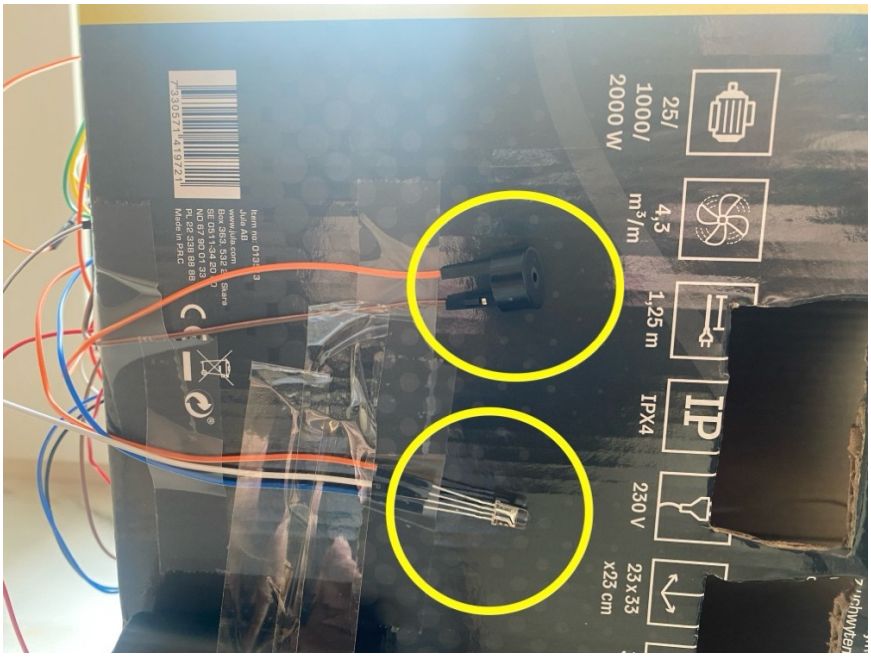

Figure 5.1,6: Buzzer and RGB LED.

Moreover, our advanced motion sensor shown in Figure 5.1,5 plays a crucial role in the safety of the Hen House. It's always on the lookout for potential threats outside. Upon detecting any unusual movement, it instantly alerts farmers with a combination of sound and light signals shown in figure 5.1,6. This immediate notification system allows for quick action to be taken,

further securing the well-being of the poultry.

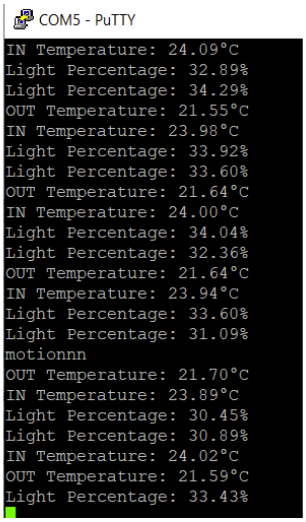

Figure 5.1,7: Output.

In Figure 5.1,7 can we see the output for our devices like the temperature for inside, the temperature for outside the Hen House, the light percentage and even a prompt for the motion sensor which activates by motion.

In summary, our Hen House is more than just a shelter; it's a comprehensive system designed to automate crucial aspects of poultry care, thereby enhancing the overall efficiency of farm management. By integrating these smart features, we've not only fulfilled our promise but also set a new standard for poultry care.

5.2 Implementation:

The development of a multifunctional system begins with initializing the microcontroller's digital pins, a critical step where a function assigns each pin as either an input for receiving signals or an output for sending them, depending on the component it connects to. Following this, temperature data is gathered through a function utilizing an Analog-to-Digital Converter (ADC) pin, which translates the analog signals from the temperature sensor into digital values the microcontroller can interpret. A similar approach is employed for the light sensor, connected through another ADC pin to monitor environmental light levels by converting light intensity into digital data.

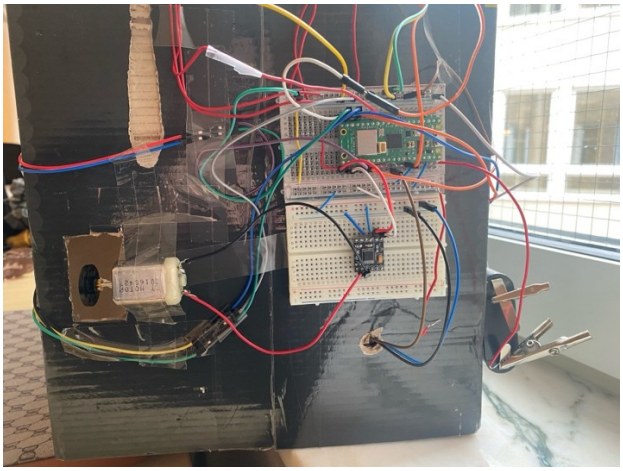

Figure 5.2,1: DC motor.

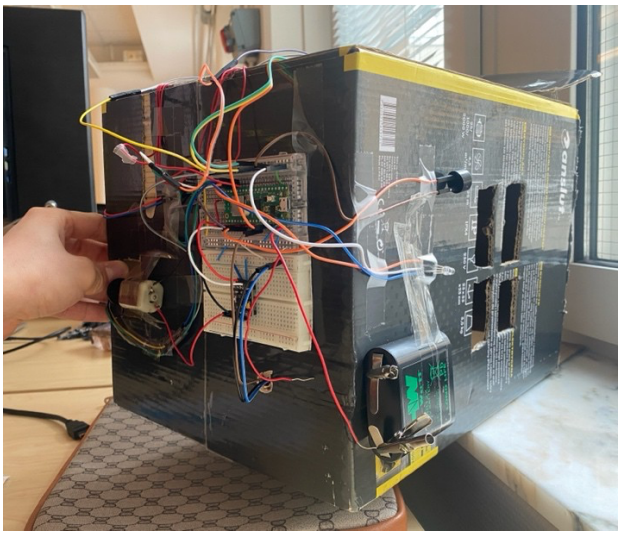

Figure 5.2,2.

For the DC motor's operation shown in Figure 5.2,1 and 5.2,2 parameters are established to dictate its activation, using stored light values for comparison to avoid unwarranted operation. Functions are then created to control the motor's direction, allowing for precise adjustments based on sensor inputs. To enhance the safety of the chickens an infrared (IR) sensor is integrated to detect motion, activating an alarm system comprised of a buzzer and LEDs for visual and auditory alerts upon detecting unauthorized entry. The culmination of this process is the seamless integration and communication between all components—ensuring a cohesive operation of the sensors, motor, and safety features, all orchestrated to respond dynamically to environmental changes, temperature variations, light conditions, and potential security threats. Full demonstration our project: Here

6 Conclusion

In this chapter, we're going to talk about the bumps we hit along the way, the things that held us back, and what we hope to do next.

First off, we ran into a bit of a roadblock with our project because we couldn't get our Raspberry Pi to connect to WiFi using FreeRTOS with C programming language. We thought about a workaround using two Pico boards - one to do the main tasks and another, running on MicroPython, to handle the WiFi connection and send data to users. But this just showed how we were stuck without a direct way to connect online, which was a big limitation for us.

We also faced quite a few challenges that slowed us down. Just getting our work environment set up took us a few days longer than we expected. This compatibility issue made us switch to an Arduino board, hoping it would make things easier. But that brought its own problems, like when we tried using a DHT11 temperature sensor and couldn't get accurate readings. We even had to change our plans for the motion sensor and the door mechanism because the first options we tried didn't work out - the motion sensor was too sensitive, and we burned out a few servo motors trying to get the door to open and close right. In the end, we used a DC motor with a special controller to make the door work.

Another thing we wish we could do better is the door mechanism. We think a part made with a 3D printer would make it work perfectly, but we didn't have one to use.

Looking forward, we've got our eyes on fixing these issues to make our project even better. We want to find an easier way to connect to the internet, make sure our hardware and software get along, and figure out a smooth door mechanism. If we could get our hands on a 3D printer, that might just solve our door problem. These challenges have taught us a lot, not just aboutthe project, but also about being flexible and keeping at it, even when thingsdon't go as planned.

We were a team of two and took a different path than most. Instead of splitting the project up and checking in at the end, we chose to work closely together. We met every day at school, which really helped us work through things quickly and efficiently. Some days, we tackled the same issues together until we solved them. Other times, we split up different problems to work on individually, but we never divided the project itself. This way of working together was a big reason why everything went so well for us.

References

[1]. Hu, Charlotte. 2023. Finally, a smart home for chickens. *Popular Science*. https://www.popsci.com/technology/smart-home-for-chickens- coop/ [Accessed: April 4, 2024].

[2]. Soracom. n.d. Smart AG: Using IoT to Build a Better Chicken Coop. *Soracom*. https://www.soracom.io/blog/smart-ag-through-iot-chicken- coops/ [Accessed: April 4, 2024].

[3]. Raspberry Pi Foundation. (2024). Datasheet for Raspberry Pi RP2040.[Datasheet].https://datasheets.raspberrypi.com/rp2040/rp2040-datasheet.pdf [Accessed: April 4, 2024].

[4]. GeeksValley. (2021). Data for KY-013.[Datasheet]. https://geeksvalley.com/wp-content/uploads/2021/05/KY-013.pdf [Accessed: April 4, 2024].

[5]. Tom's Hardware. (n.d.). How to Control DC Motors With Raspberry Pi Pico.[Datasheet]. https://www.tomshardware.com/how-to/dc-motors-raspberry-pi-pico [Accessed: April 4, 2024]

[6]. SparkFun Electronics. (n.d.). TEMT6000 Ambient Light Sensor Datasheet. [Datasheet]. https://www.sparkfun.com/datasheets/Sensors/Imaging/TEMT6000.pdf [Accessed: April 4, 2024].

[7]. Sunrom Electronics. (n.d.). Infrared Sensor Board. [Datasheet]. https://www.sunrom.com/p/infrared-sensor-board [Accessed: April 4, 2024].

[8]. Farnell. (n.d.). L293D - Quadruple Half-H Drivers. [Datasheet]. https://www.farnell.com/datasheets/2171929.pdf [Accessed: April 4, 2024].

Appendix 1.1 A

Source Code: GitHub repo.

https://github.com/Senchi01/Hen_house

Appendix 1.2 B

Demo: Google Drive.

https://drive.google.com/file/d/16_7qhX5JPwQ1vOdor1uuvIBzF-z771k-/view?usp=sharing