# Alexa Automated Pet Feeder

Cameron Nero, Jacob Paul, Carlos Lairet, and Liam Tsoi

Dept. of Electrical Engineering and Computer Science, University of Central Florida, Orlando, Florida, 32816-2450

***Abstract* — The goal of this project was to design a pet feeder that can be controlled via Alexa. The product being Alexa enabled means owners will not have to stress about being home on time. The microcontroller will be the brains of the pet feeder and the NodeMCU will be sending signals to the microcontroller so that it knows how to react to each command. The software makes it so that different scenario will result in a variable being set to a certain number. The pet feeder will know what each number means and depending on that number, the pet feeder will execute that command that the user asked. The design is user friendly since Alexa is not hard to use and it is also pet friendly. With the right adjustments, this pet feeder can have different uses like being a pill dispenser.**

## I. INTRODUCTION

The Alexa Automated Pet Feeder is a way for pet owners to easily feed their animal/pet when they are away for an extended period of time. The use of an Atmel microcontroller, a NodeMCU development board, ultrasonic sensor, weight sensor, and a NEMA 17 motor allow for this pet feeder to have many different features. The ultrasonic sensor is used to be able to tell how low the food level is, once the food level is low, the user will get an error message when they try to dispense the food. The weight sensor is there to make sure that the pet feeder does not accidentally over feed the pet. The user will also get an error if they try to dispense food when the food bowl is already filled. The Atmel microcontroller is one of easy implementation into our project. The use of this microcontroller on the Arduino UNO allows for easy testing and integration with the NodeMCU Wi-Fi board. Arguably the most important aspect of the project is the Alexa integration. The NodeMCU allows to connect to the internet and thus connect the Alexa skill system where we can create our own skills/commands to dispense food or give errors when something is wrong with the project. The Alexa Integration allows for a very simple use from the user and the ability to feed the pets from wherever they are at the time.

## II. HARDWARE OVERVIEW

The Alexa Automated Pet Feeder has several components that will all be introduced into more detail during this section of the paper. A hardware block diagram can be found in the image below.

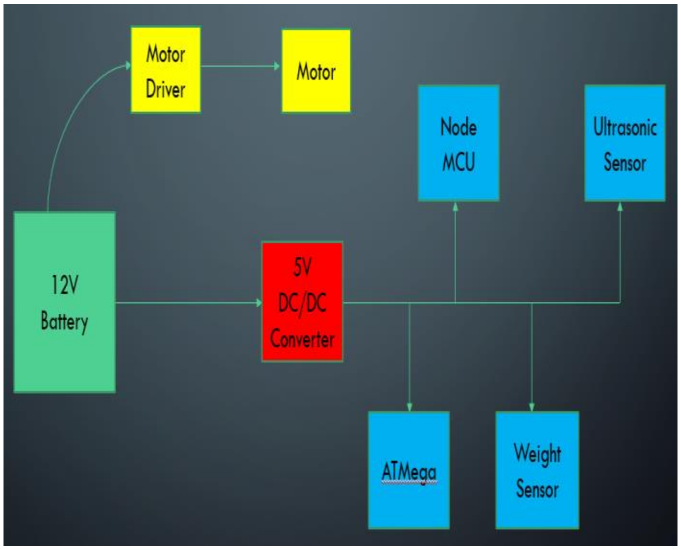


*Fig. 1. Hardware Block Diagram*

### *A. ATMega328P Microcontroller*

This microcontroller is made by Microchip Pico Power and is an 8-bit AVR RISC-based microcontroller that has 32KB of flash memory with read-while-write options. This MCU in particular could prove to be very helpful since it has 23 GPIO pins and 32 general purpose working registers. Since this MCU is easily found on the Arduino UNO, which is a development board, it would be rather easy for us to be able to test/debug this specific microcontroller. The specs of this microcontroller can be found in the table below.

### *B. NodeMCU*

The NodeMCU has Wi-Fi capabilities that will be used to connect to Amazon Web Services. This will make it so that it can connect to Alexa. The NodeMCU has low power consumption, which we need because the NodeMCU will always be on since it needs to check for request. There are dedicated pins for UART which is needed for our case since we are sending the weight value over UART communication. We also decided to use this device because it has 16 GPIO pins.

### *C. Ultrasonic Sensor*

In our project there will be a sensor mounted on the inside of the housing unit to detect whether or not food level is low and that it needs to be refilled or not. The sensor would essentially look to see if anything is in front of it (in this case, food) and if it does not detect any food, we will have it alert the user. Obviously, we want the sensor to communicate with the microcontroller and if a flag is raised, it will then communicate with the Alexa software and either give an alert or send a text message saying that the "Food Level is Low". Now let us take at the HC-SR04 ultrasonic sensor which was the food reservoir sensor used for this project.

This HC-SR04 Ultrasonic ranging module can have a ranging distant from 2cm to 400cm. The basic principle of how this work is that it uses an IO trigger for 10us, and the module will send eight 40 kHz and detect whether or not a pulse signal is sent back. The working parameters for the HC-SR04. This sensor has a very low cost and is easily able to be integrated because of the available projects that have used this sensor.

### *D. Load Cell Weight Sensor*

The last sensor that is used in this project is the bar-style load cell. This sensor mounts on one end using the predrilled mounting holes and suspends the other end, where the load is placed on top of. Using a full bridge circuit, also known as a Wheatstone bridge, this sensor accurately displays minute changes in resistance caused by flexion in the material, in the form of voltage changes. The voltage is then amplified into a readable level. The advantage of this sensor is its low cost, accompanies with its ease of use as it can be purchased in a kit with an included amplifier and pre-built mounting plates. It also comes in the desired range for food measurement (0-5kg).

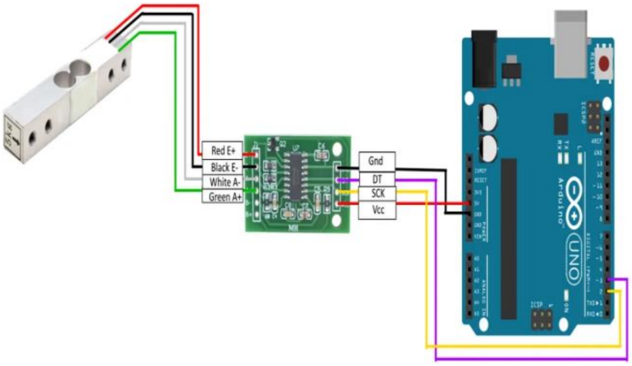


*Fig. 2. Load Cell Wiring Diagram*

### *E. Nema 17 Motor*

The Nema 17 series of bipolar stepper motors are well known in the 3D printing world for being powerful yet reliable. These stepper motors come in various sizes with differing output torques. We chose the 17HS2408 which is on the smaller, less powerful end of the spectrum compared to its counterparts. The 17HS2408 provides a holding torque of 12Ncm while consuming less than half of the current as the next size up, at 0.6A max.

Although the Nema 17 motor we chose draws a considerable amount of power compared to a standard DC motor, our group was willing to sacrifice some power consumption for the controllability and reliability a stepper motor provides. By using a stepper motor, the dispensing mechanism is easily controlled using a technique called micro stepping. The speed and rate at which the motor turns is tunable using microsecond delays between steps, which can be lengthened or shortened based on the desired flow rate.

### *F. A4988 Motor Driver*

When using a stepper motor, a stepper motor driver is necessary to interpret code from the main PCB and control the poles inside of the stepper motor. The A4988 motor driver is used in this project because it pairs well with the 17HS2408. The A4988 has an adjustable output current with a max of 2A and also is capable of operating two-phase on driving, which is what our stepper motor requires to get the most power. The A4988 has an input voltage range of 8V-35V and a logic control voltage of 5V. This allows us to connect the driver directly to the 12V battery for power and control it with 5V from the main PCB. To top off these features, the A4988 provides overcurrent protection and comes with an optional stick on heat sink. We are operating this driver well within its rated output; therefore, no overheating issues are to arise.

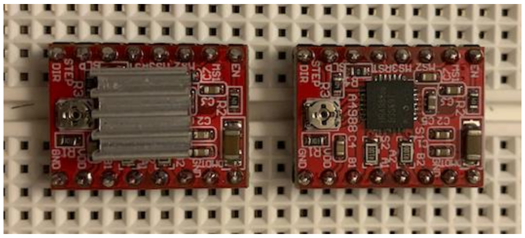


*Fig. 3. A4988 Driver with and without Heat Sink*

### *G. Battery*

The battery is a key component to the pet feeder because it allows it to be wireless and reliable. When choosing a battery, some power calculations needed to be done in order to select the proper sized battery to meet our specification of operating for one week, or 14 feeding cycles on one charge. The battery was originally sized including components for our stretch goals. The max amp draw at full run time was calculated to be approximately 4 amps. Using this max consumption value, the necessary Amp Hour calculations can be seen below.

**Amp hours**

$$Ah = Q = It$$

Using the equation above, we can estimate the size of battery, in Ah, needed. The calculations go as follows:

$$Q = 4A * 2mins * 14cycles$$

$$Q = (112\ A*mins)/60mins$$

$$Q = Ah = 1.867Ah$$

The value of 1.867Ah tells us that this is the minimum rated value needed to operate 14 feed cycles at approximately 2 minutes per cycle. Since this calculation does not consider idle time, where a small amount of power is drawn for long durations, we originally decided to use a 3Ah battery. Upon researching available batteries, it became apparent that the 5Ah battery option was more economic and had a negligible increase in size, along with eliminating any worry of underperforming on longevity. The final battery choice for this project is a Mighty Max 12V 5Ah model ML5-12. This battery is a Sealed Lead Acid (SLA)/Absorbent Glass Mat (AGM) style battery. Battery safety is always a factor that needs to be considered when using a wireless product. To ensure the safety of human and pet users, an SLA/AGM style battery is used. This means the battery will not spill any fluids out if tipped over and will not emit harmful gasses under proper operation and charging. The charging is handled by a 750mAh charger with overcharge protection. This battery charger falls within the 20% rule for battery charging as to not charge the battery too quickly and have overheating problems.

## III. SOFTWARE DETAIL

In this section, the software design of the pet feeder will be covered. This will include the design of the Alexa Integration as well as the ATMega328P code.

### *A. Alexa Integration*

The Alexa integration of the Pet Feeder is one of the most crucial components of the project. This is because it is the platform that the user will use to interact with the Pet feeder. To integrate Alexa, an Alexa Skill had to be developed for the Pet Feeder. This skill allows the Alexa Voice Service to interact with the Pet Feeder. Essentially, the user will tell Alexa a command, the Alexa voice service will call the Pet Feeder Skill and then the Skill logic will take care of interacting with the Pet Feeder Hardware. The pet feeder hardware is the NodeMCU, in this case. The link between the Alexa Skill and the NodeMCU is established by 2 files that are stored in the Amazon Cloud service. These files contain the status of the pet feeder. Every couple of seconds the NodeMCU uploads an updated status file to the Cloud and also reads the other files to check if the user has requested a feeding operation. This is accomplished with the Help of AWS Lambda Functions. These are Amazon Hosted Functions that run on the cloud every time they are called. The NodeMCU calls this function through an API with the most updated status of the Pet Feeder. This status includes the status of the bowl on the pet feeder, as well as the food reservoir status. The function takes this information and updates the online files with the latest status of the Pet Feeder. If the function sees that Alexa called to feed the pet, the function will let the NodeMCU know that it must dispense food. Any time that a user sends a feed request, the Alexa Skill will be called, and it will update the file on the cloud with a feeding command. This will let the NodeMCU know that it must dispense food next time it receives a packet from the lambda function. These files online, as mentioned before, contain the status of the pet feeder. If the user request for the status of the pet feeder (for example, is there food in the reservoir? Or is there food in the bowl?) the Alexa Skill will check this file online for the latest status (recall this file is updated every 5 seconds). Through this link, Alexa and the pet feeder interact and provide a user-friendly experience. Below is a flow diagram of how the Alexa integration portion of the pet feeder works:

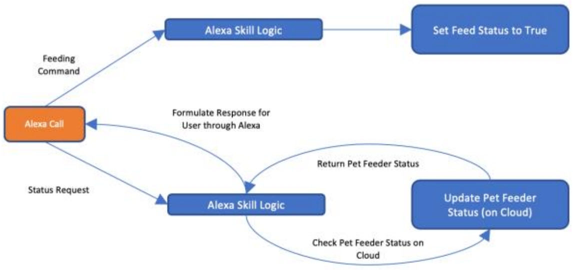


*Fig. 4. Alexa Integration Diagram*

The status files in the cloud contain status bits. There is a total of 3 bits. A feeding bit, reservoir bit, and a bowl bit. The feeding bit indicates that a feeding command was given by the user, the reservoir bit is the status of the reservoir (0 = reservoir low on food, 1 = reservoir contains food), and bowl bit is the status of the food bowl (0 = bowl is empty, 1 = bowl contains food). These bits are contained in a text file in the AWS Bucket Service online. Since all of this information is stored securely in the cloud, the Alexa Service can always access it. Through this methodology, the user can know the status of the pet feeder or send a feed command no matter where in the world they are. In the pet feeder's design, the NodeMCU is simply the broker for Alexa commands, the device in control of all the components is actually the ATMega328p. Below is a wiring diagram between the NodeMCU and the ATMega328p.

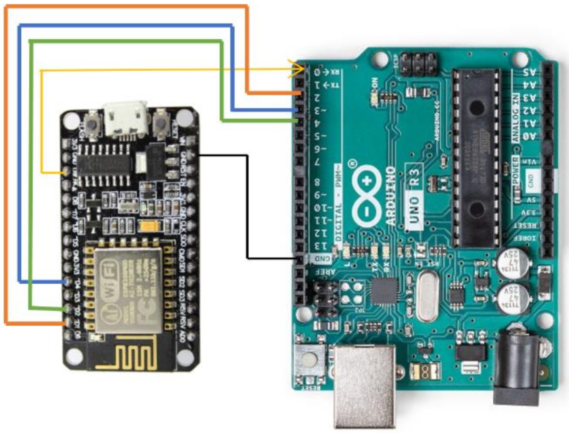


Fig. 5. Wiring Diagram between ATMEGA and NodeMCU

Using the setup above, the NodeMCU and the ATMega can share information about the status of the Pet feeder. As well as send feeding commands. There are a total of four wires used for communication between the ATMega and the NodeMCU (plus ground). Three of these wires are digital pins and one is the UART communication line. The green digital pin between the NodeMCU is used for Alexa requests, the Blue is used for reservoir status, and the orange is used for bowl status. The ATMega sets the value of the reservoir status and bowl status digital pins and the NodeMCU checks for a HIGH or a LOW value on these pins. Moreover, the Alexa pin (green) is set by the NodeMCU to high or low and read by the ATMega for Feeding commands. Finally, the TX pin of the NodeMCU is connected to the RX pin of the ATMega so that the NodeMCU can tell the ATMega how much food to dispense (depending on what the user sets through Alexa). To tell the ATMega of the requests, there is a digital pin between the NodeMCU and the ATMega. Next, the structure of the ATMega code will be discussed.

### B. ATMega Code

The microcontroller will be on constant standby mode until it received a high on a digital pin. Once the microcontroller receives a high then it will go through different cases and check if it satisfies all the requirements. There are different cases in the microcontroller that are triggered by the Alexa request. The first case is to dispense food and it gets the command from the nodeMCU via digital pins. There are two cases that need to be met before dispensing can occur. The first case is if the weight is below a certain threshold. The weight of the food is determined but the scale that is placed below the food bowl. Depending on if the food bowl already has food or not, the microcontroller will be sending a digital high to the nodeMCU which is indicate that the food has passed the threshold of food weight. The second case is if there is enough food in the reservoir and this is determined with the ultrasonic sensor measuring the distance of the food in the reservoir. Both cases have to be met so that the motor function, which means the food will dispense.

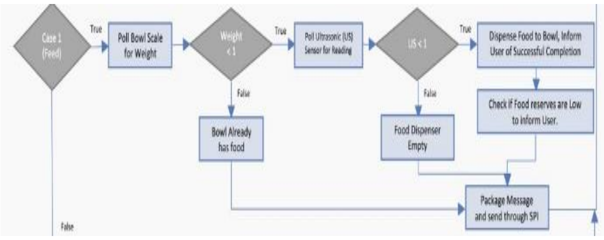


Fig. 6. Case 1 for feeding

Case 2 of the code is to check if the bowl already has food or not. This will just call the status of bowl function which will check if the weight of the bowl is past the threshold or not. Depending on if the threshold is met, the microcontroller will send a digital high or low. This will prompt Alexa to notify the user if the bowl is full or empty.

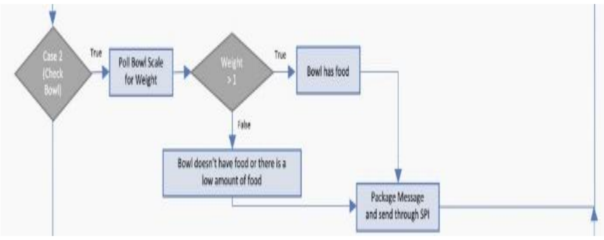


Fig. 7. Case 2 for bowl status

Case 3 of the code is to check the reservoir of the pet feeder. This will call the ultrasonic senor function and check if the distance if greater or less than 15 cm. If the sensor is reading something less than 15 cm then that will mean the reservoir has enough food and that the pet feeder can dispense food.

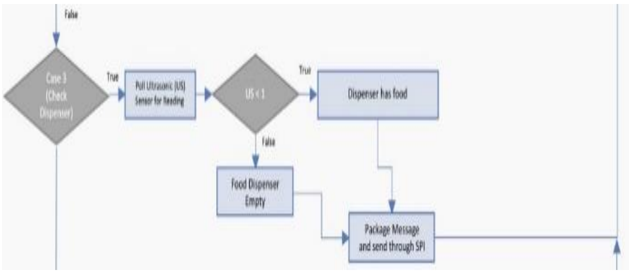


Fig. 8. Case 3 for reservoir status

# IV. HARDWARE DESIGN

This part of the report will talk about the hardware design and will go into more detail about our two separate PCBs and discuss the decisions in how they were designed.

### A. *Switching Voltage Regulator*

One of the most important features of a lot of senior design projects is the ability to deliver power to all of the components of the projects, whether that be voltage or current. Another important aspect to keep in mind is keeping the parts as efficient as possible. For our project in particular, we need to have a battery that can supply enough voltage and current to the microcontroller. The Atmega328 microcontroller that we are using takes an input voltage of 5V. We plan on using a 12V battery to power everything, but we need to make sure that we step down the voltage to 5V to be able to power the microcontroller therefore we are going to need a DC/DC converter in order to do that.

Using the WEBENCH tool that is designed by Texas Instruments, we are able to find a good converter that will be able to take 12Vin and supply a voltage of 5V with enough current to be able to support all of other supporting features connected to the microcontroller. Below are the design parameters that were inputted to WEBENCH for where it will provide multiple circuits that can perform these parameters:

| Input Voltage | Output Voltage | Max Load Current |
|---|---|---|
| 12V | 5V | 3A |

*Table 1: WEBENCH Input Requirements*

After inputting these parameters into WEBENCH we are able to select from a variety of different DC/DC converter circuits. Due to economic constraints of our project, we need to pick a design that is relatively inexpensive and that does not have a lot of components since the more components the circuit has, the more expensive the converter will be. The design we choose, and the schematic can be found in the image below. This design uses an LM2576HV Step-Down Regulator.

### B. *Volage Regulator Schematic*

Designed from WEBENCH, we are able to put in our design requirements for a DC/DC convertor. The design requirements that are required for a 12V input voltage with a 5V output voltage are input minimum voltage, maximum input voltage, output voltage along with maximum output current. The schematic that was built using EasyEDA can be found in the image below. The switching regulator that is used for this design is a LM2576HVSX, the design also requires two capacitors, a 100uF and a 1000uF along with one diode. We can then take this schematic and convert that to a PCB and then trace the parts on the PCB and then finally add a copper ground plane on the top and bottom layers on the board. The reason to include a ground layer is to have the shortest possible path to ground for all parts that require a ground connection in the design.

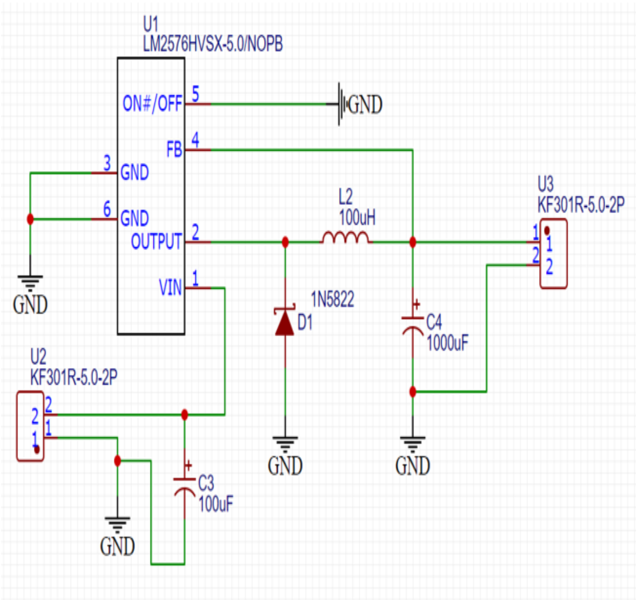


*Fig. 9. DC/DC Schematic*

### C. *Voltage Regulator PCB Layout*

The top layer of the board can see below with a copper ground plane also included. Also, below, we can see the initial traces with a ground plane so we can see how a via was implemented. We needed to use a via since we could not trace underneath an existing trace. Because we did not want a very long trace, the use of a via allowed us to trace underneath the board and then finally trace back on the top layer.

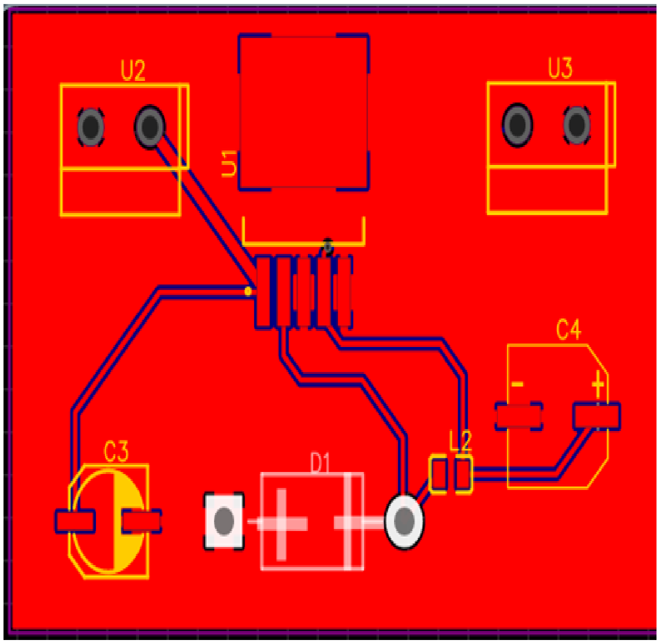


Fig. 10. DC/DC PCB Layout

### D. Main PCB Design

The next couple of sections will go into more detail about our main PCB of this project. This PCB is critical for the success of this project and must be designed correctly for it to function correctly when everything is integrated onto the PCB.

### E. Main PCB Schematic

This schematic was designed using EasyEDA. The schematic design has a terminal block for the microcontroller to accept 5V from the DC/DC printed circuit board that was just previously discussed. It also includes pin headers for in circuit serial programming. This is an important aspect of our PCB because we do not want to have to keep mating/de-mating our microcontroller from our PCB because that runs the risk of damaging our chip.

We have also added three status LEDs on our PCB to have the ability to give the user feedback on errors/power/etc. We also have a reset button, so the user can manually reset the code incase something is not working properly. Lastly, there is multiple pin headers for all of our sensors. The motor driver, ultrasonic sensor and HX711 load cell amplifier all require 4x1 pin headers. Two of the pins are left for two digital GPIO pins that are on the microcontroller and the other two are for +5V and GND.

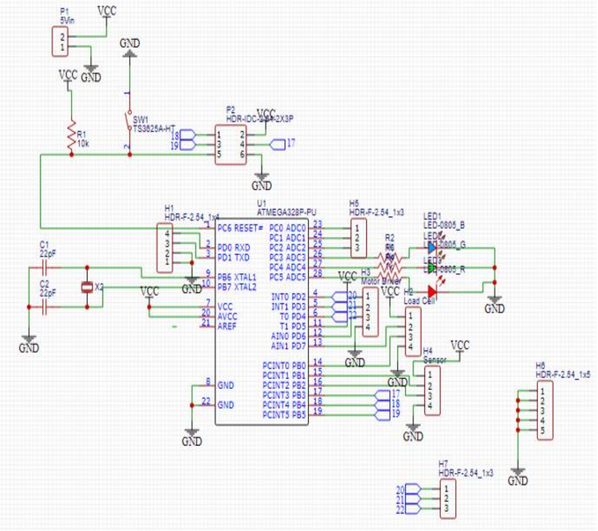


Fig. 11. Main PCB Schematic

### F. Main PCB Layout

As seen in the image below, we have out main PCB layout which is a 2-layer, 1 ounce copper design to accommodate the larger traces required. Overall, this PCB has a total bill of materials of 17 components. Along with all of that, the board is of relatively small size, with dimensions of 75mm x 55mm.

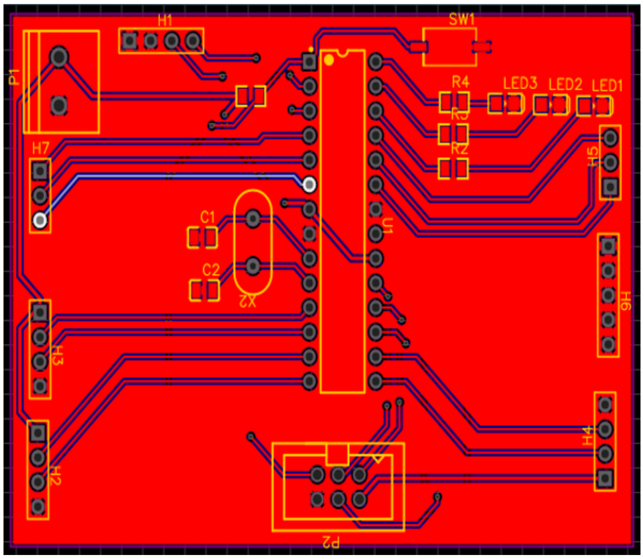
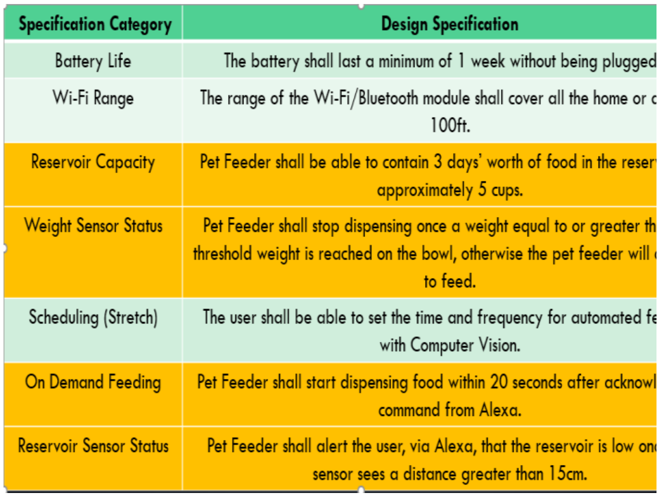


Fig. 12. Main PCB Layout

## V. TESTING RESULTS

The specifications that were tested for our project can be found highlighted in the table below. We will also go over the results to ensure that all of the engineering specifications were passed.



| Specification Category | Design Specification |
|---|---|
| Battery Life | The battery shall last a minimum of 1 week without being plugged |
| Wi-Fi Range | The range of the Wi-Fi/Bluetooth module shall cover all the home or 100ft. |
| Reservoir Capacity | Pet Feeder shall be able to contain 3 days' worth of food in the reser approximately 5 cups. |
| Weight Sensor Status | Pet Feeder shall stop dispensing once a weight equal to or greater th threshold weight is reached on the bowl, otherwise the pet feeder will to feed. |
| Scheduling (Stretch) | The user shall be able to set the time and frequency for automated fe with Computer Vision. |
| On Demand Feeding | Pet Feeder shall start dispensing food within 20 seconds after acknowl command from Alexa. |
| Reservoir Sensor Status | Pet Feeder shall alert the user, via Alexa, that the reservoir is low on sensor sees a distance greater than 15cm. |

*Table 2: Specification Table*

### *A. Reservoir Capacity*

The reservoir capacity specification states that the pet feeder shall be able to contain 3 days' worth of food in the reservoir, or approximately 5 cups. The image from our testing is below, we have placed 5 cups of beans (a substitute from pet food) into the reservoir and we are only halfway full. Meaning our pet feeder will be able to contain approximately 10 cups of food, which is way more than our specification requires.

*Fig. 13. Reservoir Capacity Testing Results*

### *B. Weight Sensor Status*

For the weight sensor status, the specification says the pet feeder shall stop dispensing once a weight greater to or equal that the threshold weight is reached on the bowl, otherwise will continue to feed.

This specification was also passed as during testing when a weight was placed on the load cell that was greater than the predetermined maximum weight, the motor would stop spinning meaning that no more food would be dispensed. This is exactly what we want because we do not want to be overfeeding the pet every time, we attempt to feed them.

### *C. On Demand Feeding*

Our On Demand Feeding Specification states that the pet feeder shall start dispensing food within 20 seconds after acknowledging the command from Alexa. The screenshot below is from testing and as seen in the photo, it took approximately 12.9 seconds for the motor to begin spinning. Thus, this specification has passed.

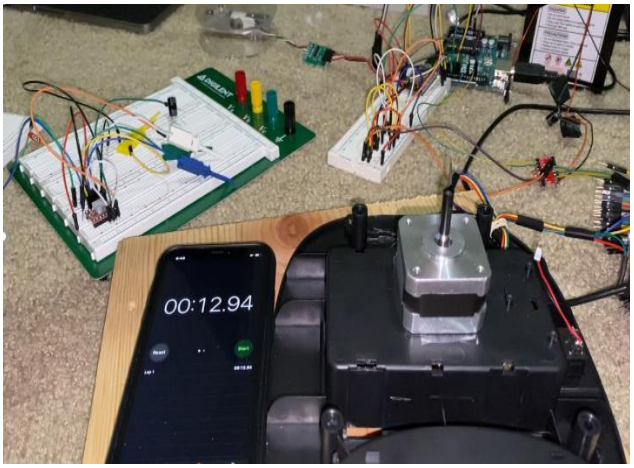


*Fig. 14. Main PCB Layout*

### *D. Reservoir Sensor Status*

The last specification that was tested is the reservoir sensor status. The specification states that the pet feeder shall alert the user, via Alexa, that the reservoir is low once the sensor sees a distance greater than 15cm.

To test this specification, we started by simulated the reservoir as full (the ultrasonic sensor would see a distance less than 15cm) then asked the pet feeder to feed the pet. The motor would start to run and once we emptied the pet feeder, it would stop because the pet feeder was out of food.

A second test we did was to start with the pet feeder empty and then ask Alexa to feed the pet, this would give an error via Alexa stating, "food reservoir is low, please refill and try again." This is exactly what the specification is stating, thus this and all of the specifications for this project is successful.

## VI. CONCLUSION

To discuss some concluding remarks, we would like to talk about what has been accomplished during our time working on the Alexa Enabled Pet Feeder. As a group, we have completed extensive research on individual components as well as the system as a whole. It is imperative that the correct components were selected to be used in this project to avoid component related issues down the line. While completing research for the most ideal components, we were required to adhere to a strict budget due to our collectively limited resources. Throughout the semester, we have been able to remain within our budget. The budget did get a little close due to the fact that we needed to re-purchase a PCB because we did not have in circuit serial programming on the board itself.

As for overall system function, we have purchased a prebuilt pet feeder and stripped it of its already performing electronics and then placed our own electronics inside. Throughout this semester we have learned how to work together and overcome challenges to complete our Alexa Automated Pet Feeder. as we are going to have to do in the future as engineers.

## VII. THE ENGINEERS

**Cameron Nero**
**Electrical Engineering**

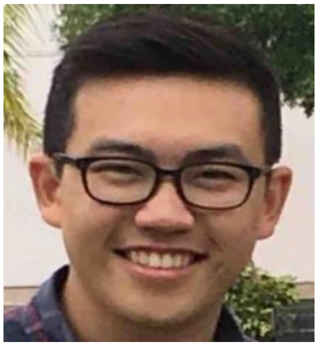
Cameron Nero is an undergraduate student at the University of Central Florida, majoring in Electrical Engineering. Cameron has been a CWEP for the past two years while doing two actual internships with Lockheed Martin the past two summers. Cameron will be a full-time employee as an Electrical Engineer Associate following graduation this summer. Cameron's main role in this project was the DC/DC design along with the main PCB design.

**Jacob Paul**
**Electrical Engineering**

Jacob Paul is an undergraduate student at the University of Central Florida, majoring in Electrical Engineering. He has been an intern at Curis System for the past year doing database management. Jacob will be a full-time employee for Black Cat Integrated Power as applications engineer following graduation. Jacob’s focus on the project was power supply, stepper motor programming and hardware adaptation. He has taken circuit analysis and software engineering classes but feels most comfortable with the hands on, hardware side of the project.

**Carlos Lairet**
**Computer Engineering**

Carlos Lairet is an undergraduate student at the University of Central Florida, majoring in Computer Engineering. He has been taking classes at the university for four years under the Electrical and Computer Engineering Department. Carlos’s focus on the project is Alexa integration of the Pet Feeder. Moreover, Carlos was also involved in developing the code for the ATMega MCU.



**Liam Tsoi**
**Computer Engineering**

Liam is an undergraduate at the University of Central Florida, majoring in Computer Engineering. Liam is currently working at TRC as a power delivery intern. He is going to pursue a master's degree in the near future but plans on taking a break in the meantime. Liam’s main role in the project has been developing code for the ATMega MCU.