Automated Pet Feeder

Paola A. Buitrago, Malcolm A. Morgan, and Hector L. Rodriguez

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

***Abstract* — This paper provides a detailed description of the development process for the Automated Pet Feeder. The development process is divided into several stages, and several sections within each stage. Stages include: (1) Motivation for choosing the Automated Pet Feeder as a Senior Design project; (2) Research conducted for the desired functionality for the Automated Pet Feeder; (3) Development stages and the process within each stage of the development for Automated Pet Feeder.**

***Index Terms* — Automated, cat, dog, feeder, pet, smart.**

I. Introduction

According to Statista, the number of dogs living in households in the United States has reached a high of 89.7 million as of 2017 [1]. Considering that Census Bureau has estimated the United States population to be 325.8 million as of writing, this culminates in roughly 27.5% of the population owning a dog as a pet [2]. Undoubtedly, dogs are an important aspect of many households in the United States, though many individuals find themselves in the daily routine of feeding their dogs, and for many, complications arise when one's occupation separates one from his or her dog for long periods of time each day, or when one owns more than one dog, which requires a different feeding routine than the norm. Two of the three members in our group own at least one dog, with one member owning two dogs, and the other member planning on adopting a dog once shortly after graduation. Thus, several of the solutions to dog ownership that this project addresses are as a direct result of problems encountered by members of our group, and these solutions could certainly be employed in the lives of many other dog owners.

As its name implies, the objectives and goals of the Automated Pet Feeder include a collection of requirements that are meant to ease the burden of pet ownership. One objective is that the Automated Pet Feeder must continually monitor the weight of the food and water supply of both its dispensers, while also holding a minimum quantity of food and water capable of feeding a large dog for an extended period of time, particularly several days. The Automated Pet Feeder must also be able to differentiate between various dogs, and only dispense food to a dog specified by the owner, while simultaneously preventing unauthorized dogs, and other unauthorized pets from obtaining food from the food dispenser. Support for additional functionality of the Automated Pet Feeder will be accomplished using a dedicated IP address designated to a file server, and lastly, the Automated Pet Feeder must employ inexpensive materials and a relatively simple manufacturing process, which must allow it to be priced less than competitive products.

Functionality of the Automated Pet Feeder are derived from the several objectives listed above. The Automated Pet Feeder will routinely dispense water from its water reservoir whenever water is removed from the water bowl and dispense food from its food reservoir in portions of mass converted from weight that will have been specified by the pet's owner, and these portions must be able to be specified from an application on the pet owner's mobile device. Information regarding the pet's eating patterns will be sent to the pet owner's mobile device, with the inclusion of the ability to stream live footage of the apparatus's environment from a camera located on the apparatus. The ability to differentiate between an authorized pet and unauthorized pets will be accomplished using collar recognition, which is one of the primary features that sets apart the Automated Pet Feeder from its competition. The primary competitor to the Automated Pet Feeder is the Gosh EasyFeed, which is a similar project being funded and advertised on Kickstarter[3].

I. Embedded Hardware and software

Primary functionality of the Automated Pet Feeder is accomplished using a microcontroller. Considering that numerous types of microcontrollers are produced by many companies, choosing the most appropriate microcontroller for the desired functionality was a task in itself. One primary aspect of the selection process that narrowed the options of microcontrollers being considered was the knowledge that members in our group possessed of several microcontrollers already. These several microcontrollers included the Atmel ATmega2560, the Atmel ATmega328P, and the Texas Instruments MSP430. While any of these three microcontrollers could accomplish any single task required for the Automated Pet Feeder, the area of concern when choosing the most appropriate microcontroller was being able to integrate all functionality on a single microcontroller.

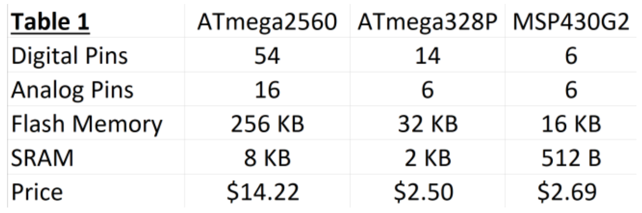


Table 1	ATmega2560	ATmega328P	MSP430G2
Digital Pins	54	14	6
Analog Pins	16	6	6
Flash Memory	256 KB	32 KB	16 KB
SRAM	8 KB	2 KB	512 B
Price	$14.22	$2.50	$2.69

A. MSP340

As seen in Table 1, the variation in the amount of general purpose input and output (GPIO) pins provided by each microcontroller differs greatly between the three. It was quickly understood by our group that although all members had prior classroom experience programming the MSP430G2, its limited number of 12 GPIO pins rendered it unsuitable for the Automated Pet Feeder as the number of peripherals required to implement all functionality would require more GPIO pins than the MSP430G2 can provide.

B. ATmega328P

The second microcontroller under consideration was the Atmel ATmega328P. While none of our group members had prior experience programming this microcontroller beyond the equivalent of *Hello, World!*, the extensive opensource hardware and software documentation provided through Arduino caused this microcontroller to be an attractive option. One issue of concern regarding the ATmega328P was also its limited number of 20 GPIO pins. At the time when the decision to choose a specific microcontroller was made, external motor drivers were intended to be used to control the two direct current motors being utilized to dispense food and water, thus requiring a greater number of GPIO pins than reflected in the final design. Thus, it was incorrectly understood at the time that if the ATmega328P were to be our microcontroller of choice, two would have to be utilized to implement all required functionality of the Automated Pet Feeder.

C. ATmega2560

The third microcontroller under consideration was the ATmega2560. The ATmega2560 shares many similarities with the former ATmega328P, most notably being manufactured by the same company, and having extensive software libraries and hardware documentation provided by Arduino. While the excessive amount of digital and analog GPIO pins would increase flexibility in the design process, the increase in price and added complexity in the printed circuit board design rendered the ATmega2560 not ideal for this application as well.

D. ATmega328P – Final Choice

The ATmega328P was chosen as the microcontroller of choice to implement all embedded functionality required for the Automated Pet Feeder, as its moderate amount of GPIO pins, low cost, and extensive documentation proved ideal for the overall design of the apparatus. The schematic for the apparatus, however, was initially designed using two ATmega328P microcontrollers, as it was initially understood that an external L9110 motor driver utilizing an h-bridge was necessary to modulate the operation of the two direct current motors. However, as the schematic was finalized, and the rising concern increased regarding synchronization between both ATmega328P microcontrollers, an alternate option of removing the L9110 motor driver altogether was chosen to be the more feasible design. This design alteration would remove the L9110 motor driver, which required 6 pins. Thus, using a single pin for controlling the on and off states of each direct current motor in conjunction with support from two relays mounted to the printed circuit board, allowed for a reduction in 4 pins from the overall design.

This change in design allowed all peripherals required to implement functionality to be routed to and controlled by a single ATmega328P microcontroller.

Several key aspects of the ATmega328P that enabled it to be integrated seamlessly with other components of the Automated Pet Feeder include its operating voltage range, operating frequency range, and operating temperature range. Per the parametrics table provided by Atmel, the ATmega328P can operate at voltages between 1.8V and 5.5V, and can also operate at frequencies of up to 20MHz. The temperature for a suitable environment for the ATmega328P has a broad range between -45 °C and 80 °C, thus allowing operation of the Automated Pet Feeder to not be limited to indoors nor specific seasons of the year. Lastly, while Atmel does not provide a C++ compiler for its microcontrollers, numerous libraries exist that can be compiled using the Arduino IDE. This allows for faster software development in comparison to utilizing AVR assembly.

E. Adafruit MicroSD card breakout board+ [ADA254]

As the functionality of the Automated Pet Feeder includes the ability to record and track the pet's eating patterns, this data may be collected and stored indefinitely until a request is made from the Raspberry Pi for this data to be transmitted to the Raspberry Pi via the serial communication interface of the ATmega328P. Considering that the static random access memory of the ATmega328P is merely two kilobytes, storing an ever-increasing buffer of variable size would prove catastrophic in the event that the data in the buffer was not requested, thus causing the ATmega328P to run out of static random access memory and the program flashed onto it to crash as a result.

This memory shortage obstacle was circumvented through the implementation of a micro secured digital card breakout board manufactured by Adafruit. The specific model implemented is the ADA254, and when used in conjunction with a micro secured digital card, permanent storage in the magnitude of gigabytes is achieved. This allowed the Automated Pet Feeder to track the pet's eating pattern for an indefinite period of time and provide all of this data to the user once a request for this data is made.
Operation of the ADA254 is achieved using libraries found in SdFat-1.0.5, which is an opensource collection of classes and examples licensed by Massachusetts Institute of Technology for use within the Arduino IDE. One challenge encountered while developing the embedded software using SdFat-1.0.5 was the inability to change the selection of pins that the ATmega328P utilized to communicate with the ADA254. This is in striking contrast to every other peripheral connected to the ATmega328P, as the pins that the user would like to use for a specific peripheral are typically specified in the constructor of an object upon declaration. Thus, all other peripherals connected to the ATmega328P had to be integrated around the placement of the ADA254.

F. HX711 ADC Converter and Digital Load Cell Weight Sensor

Obtaining information regarding the pet's eating pattern, and dispensing water in appropriate quantities is achieved using two HX711 analog to digital converter produced by Avia Semiconductor's in conjunction with two digital load cell weight sensors. Two digital pins from the ATmega328P are required for operation of each HX711, and these pins can be specified in the constructor when declaring an HX711 object from the HX711 libraries. This ability to choose which digital pins are utilized added flexibility to the design of the schematic.
While two HX711's are utilized to achieve complete functionality, the way in which each one is used differs. One HX711 is configured to produce data concerning the weight of the pet's bowl, which correlates to a portion of food eaten by the pet if a final value is subtracted from an initial value. These values concerning the pet's eating pattern are then stored using the ADA254 until requested from the user. The second HX711 is utilized to continually monitor the level of water in the water bowl of the Automated Pet Feeder, thus allowing the ATmega328P to determine when insufficient water is present and dispense additional water.

G. RDM6300

One of the specification requirements of the Automated Pet Feeder is prohibiting unauthorized pets from accessing the food and water of the apparatus. This is achieved through implementing an RDM6300 RFID reader to read the radio frequencies produced by the tags attached to the pets as they approach the system. Fortunately, the RDM6300 requires merely two digital pins for transmissions of data and recognizes 125KHz RFID tags that adhere to the EM4100 protocol. While this translates to a relatively inexpensive implementation, as 125KHz RFID readers and 125KHz RFID tags are readily available, the range at which these tags are verified is limited to approximately 4 centimeters or less.

H. HC-SR04 Ultrasonic Sensor Distance Module

While the RDM6300 is responsible for verifying that the authorized pet is near the feeder, its inconsistency in continually capturing data from an RFID tags rendered it ineffective for determining if the pet is still present at the apparatus. Thus, an HC-SR04 ultrasonic distance sensor was implemented to resolve this issue. The HC-SR04 requires two digital pins from the ATmega328P, and is configured to detect pets at up to 200 centimeters away, though for our implementation, a minimum distance at which the pet must be near the feeder is defined such that the feeder will restrict access to unauthorized pets once the authorized pet is not within the minimum distance.

II. Electrical Hardware

A. Dry Food motor

High Torque and low speed were the deciding factor when choosing which motor to use for dispensing the dry food. High torque is needed for to ensure the motor could push passed pebble that may get stuck between the dispensing paddle and the reservoir wall. For those reasons we chose the Nextrox Mini 12V DC motor. Its limited documentation emphasized on it high torque and low noise capabilities. Since the speed of the motor would be dependent on the amount of voltage supplied, we decided further testing would be needed. After testing various input voltages, we noted that speed of the motor using 5V would be too slow for the application at hand. The group decided that 12V provided the proper speed our system would need. While testing we noted with no load the total current draw was around 50mA, medium tension 100mA and fully locking the motor would draw approximately 350mA. We also needed to ensure that not only did the motor need to have high torque but also be fairly small in size. The Nextrox DC motor is approx. 68mm in length and 37mm in diameter.

B. Reservoir Water pump

The water pump that was chosen for our project was the Zjchao Peristaltic Liquid Pump. Since a pump would be used to transport the water from the reservoir, we decided that a peristaltic motor would be best appropriate for our application. The deciding factors were sanitation, speed and accuracy. Sanitation was a big factor since this would be the water that the pets will be drinking from on a daily basis. Peristaltic motors are ideal for this because the water would never come in contact with anything other than the tubing therefore minimizing the chance for contamination.

The speed of the motor was also of great importance. The speed at which to water dispensed needed to be slow enough that no water would splash when dispensed into the bowl to avoid unwanted spills. At 12V the motor would output up to 100mL/min which would meet the speed specification. Lastly the peristaltic motor is known for its accuracy. Like stated above, this motor could be precisely driven to calculate down to the last drop. Although we would not be using this feature to its full capabilities we will be using it to ensure proper dispensing measurements.

C. Door Servo Motor

Our pet feeder will use a door that will open and close to prevent unwanted pets from accessing food that is not meant for them. Consideration of how the door would open and close needed to be decided first. We wanted to limit the possibility of the door closing on an pet and also limit the ability of a pet being able to damage the door. If the door were to open on a hinge and open up and down, this would allow the pet to easily damage the door if the pet was curious enough. We figured the door would need to be inside the enclosure when open. Therefore, we designed a 180-rotating semicircular shaped door that would rotate open and close. When closed the door would close along a grooved track that would not allow the pet to pry open. When open the door would be completely inside the enclosure and the pet would not have any ability to damage the door. With these specifications, a servomotor would be ideal. The servo motor chosen rotates between 0 and 180 degrees and its operating voltage is between 4 and 6 volts. This is optimal since we already designed a 5v rail for our system and the door needs to precisely rotate at 0 and 180 degrees. Some issues encountered with the servo motor chosen is that its initial position upon start up is 96 degrees. This was fixed by initializing, in the code, the door to 0 degrees (closed position) to ensure proper alignment.

D. Voltage regulators

In order to provide the correct voltages to the all the operating majors components, a custom Power PCB was designed to power the systems needs form the Signal PCB. By designing a custom Power PCB, we had more versatility on the amount of voltage rails needed on one board making it easier to route power connections from the Signal PCB at different locations.

The Automated Pet Feeder will operate off either 5V or 12V rails. Since we are utilizing a 12V 3A AC to DC converter we needed the use of a voltage regulator to create a 5V rail for the rest of the system. We initially used a design taught at UCF in Electronics 2 that utilizes a LM7805. The issue with this was the efficiency and its power dissipation into heat. This heat issue would require huge heat sinks and would still make a very inefficient system. We wanted a voltage regulator that would be able to operate a much higher rate of efficiency and less heat dissipation.

This would be accomplished with the use of a switching regulator. We narrowed our search down to the LM2596 from Texas Instruments and MIC4576 by Micrel. Both were very similar in terms of specifications and features. The main notable differences were the operating frequency. The MIC4576 operating frequency is 200kHz while the LM2596 is 150Khz. According to the datasheets, the higher switching frequency of the MIC4576 may allow up to a 2:1 reduction in output filter inductor size. The ideal regulator for the feeder would have been the MIC4576 if it were not for its lack of availability. For this reason, we decided we would utilize the LM2596. After testing our current configuration in the lab, we were able to achieve an output of precisely 5.01V at the output of the voltage regulator circuit.

The detailed datasheet provided by TI assisted greatly in the overall regulated circuit. A great feature of the LM2596 was the fact that its overall regulated circuit only utilizes 4 external components. The external components needed are an input capacitor, inductor, diode and an output capacitor. The specifications for choosing the required input capacitor was the RMS current rating rather its voltage or capacitance rating. It is recommended that the input capacitor have a ripple current rating of approximately 50% of the DC load current. The input Capacitor must also be at least 1.25 times greater than the max input voltage. With these specifications and the help of fig 23 found in the LM2596 data sheet, a 680uF capacitor with a 35VDC rating was chosen. The output capacitors main criterial was RMS ripple current rating, voltage rating and capacitance value. Using the provided its tables provided a 330uF rated at 35VDC was chosen. The Schottky diode used allows for a return path for the inductor current when the system turns off. Once again TI provided a table to simplify the choosing of the diode value. The diode chosen for this application was a 5A 20V schottky diode. Like the catch diode, TI doesn't provide much details on how to calculate the inductor value needed and instead provide us with a table to calculate the desired inductor value. The inductor chosen based of the provided tables was a 33uH rated at 100kHz.

When testing this configuration on a bread board in a lab environment, we had some troubles being able to achieve the desired 5V output. After further investigation, the datasheet provided specifies that the traces of the input capacitor, schottky diode, output capacitor, ground pin and on/off pin be kept short within a few mm and utilize a ground plane for best results. After optimizing my bread board circuit to minimize the circuit traces, I was able to achieve the desired 5V output.

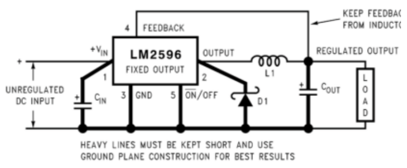


Figure 1: TI recommended LM2596 regulated circuit

E. Battery Backup System

One the most important features of the Automated Pet Feeder is that the system needs to be able to provide access to its food and water even when the main power supply is disconnected. This is important when considering power outages. A back up power supply would need to be implemented in order to ensure the proper nutrition is being provided no matter if the main power supply is connected or not. We decided that a 24-48 hour back up power supply would be sufficient enough time for the pet owner to be able to address the matter at hand.

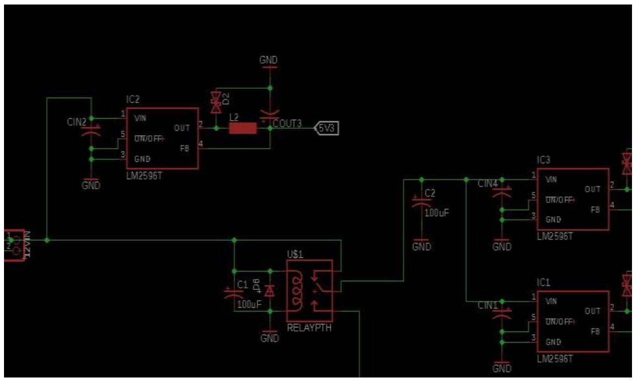


Figure 2: TI recommended LM2596 regulated circuit

When looking into what kind of batteries to utilize for the backup battery system many different options were available. Batteries such as nickel cadmium, nickel-Metal Hydride, Lead Acid, and Lithium Ion were researched. When designing the backup battery system, we wanted to keep in mind the total size needed to implement, safety precautions, and cost.

With the growing market for lithium-ion batteries and the dropping cost along with the desired cell voltage, made it an easy choice. With use of a BMS system along with 3 batteries in series configuration and proper connections, we were able to ensure that the 3 lithium ion batteries were being charged evenly. Not only are we able to achieve even charging but we are also able to eliminate the chance of overcharging the batteries along with protection from excessive discharging. This is done with the on-board protection circuit.

This would be performed by the use of a 12VDC relay. If you observe the circuit shown in Figure 1 you will see the implemented circuit design. When the main power supply is connected, the relay will activate and will supply the necessary voltage to the system along with the voltage needed to charge the backup batteries. In any instance that the main power supply was to be disconnect such as in a power outage, the relay will be turned off and now the backup battery supply will take over. The backup battery system will now provide all needed power to the Automated Pet Feeder.

Concern for the system losing power in between the switching from main power supply to the backup battery system came to mind when designing the circuit. For that reason, we would need to use a large enough capacitor to ensure enough voltage would be available during the transition state of the relay. Another concern was back feed from the backup battery supply to the main therefore a protection diode was placed in line to ensure proper current flow.

III. High Level Application Design and Interface

The diagram shown on the following page demonstrates the overall mobile application layout. User experience is enhanced using a database and a mobile application which will interact with the system's signal PCB via the Raspberry Pi's on-board Wi-Fi module. By having the Wi-Fi module, the user can communicate with the pet feeder wirelessly and push changes when needed. The MySQL database is modeled after unique configuration file which act as triggers on the signal PCB. It is the main communication grounds for how the system interacts with the mobile application sending data back and forth and is essentially for tracking much the pet has consumed and at what times.

A. Mobile Application

When making the decision on which development platform we used for the mobile application the team went with Android rather than Apple iOS. Apple's development platform needed to be done on a Mac. While one of the team members did have a Mac, it only had 4 GB's of RAM which was not enough processing power for smooth and stress-free development. Also, the Apple Store has a steep fee of $99/year to put an application in their store which is a cost the team felt like it was not necessary. The Google Play Store has more reasonable one-time fee of $25 to place applications in their store.

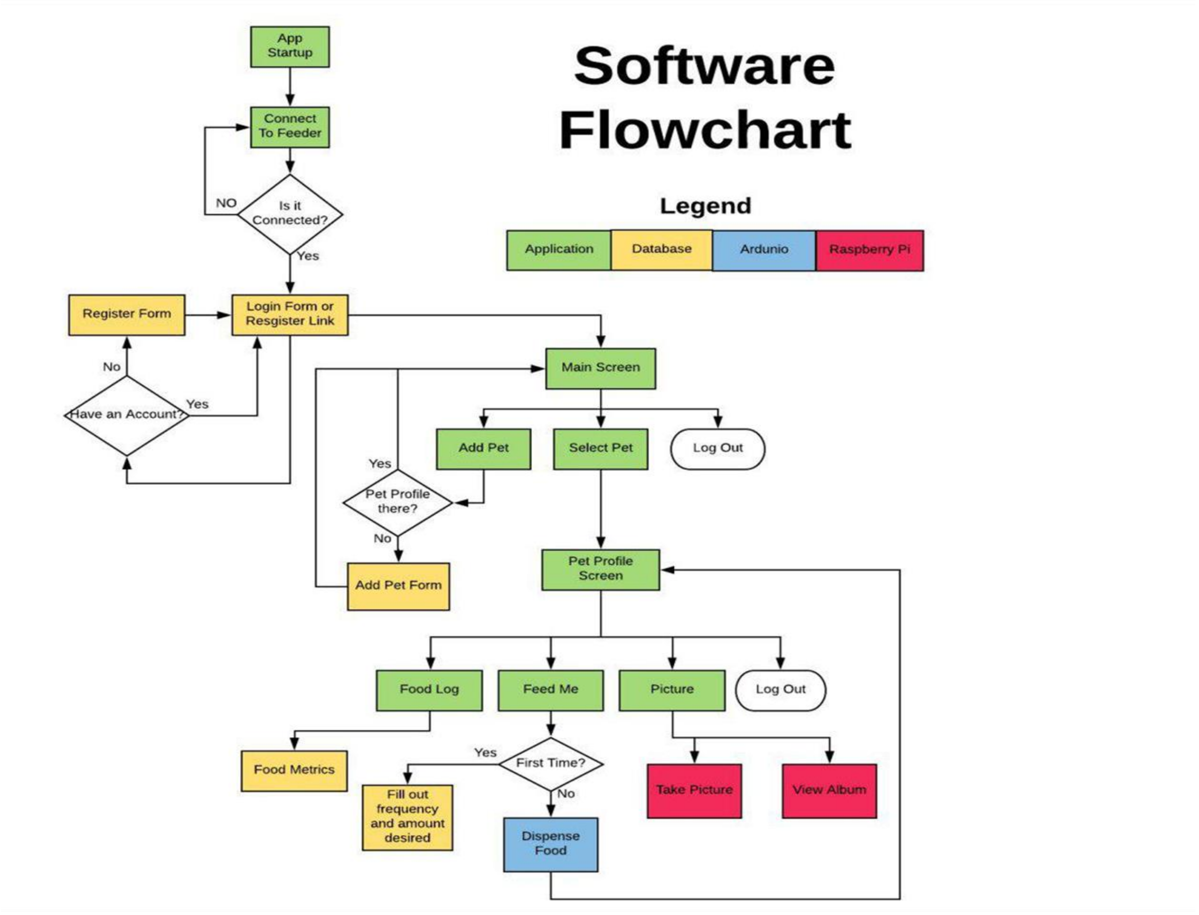
With choosing an Android Development platform, the team used Android Studio using Java as the main programming language. Java made it possible to have modular functionality throughout the application. Before any application development can occur web server plus a database needed to be set up.

Instead of choosing a paid service for hosting a site, we decided to turn the Raspberry Pi into an affordable webserver with a 32GB's of memory that uses 1GB of processing power. First a LAMP (Linux, Apache, MySQL and PHP) server had to be set up. PHP is the main language that will be used to push and pull specific data from the database which will be encoded into JSON format. When the JSON object is returned from the API, the code in the android application will interpret the JSON object into meaningful information that the user will be able to view in the application such as food history for the registered pet on the application.

B. User Interface

The image above displays an overall layout of the mobile application. When the user opens the application, they will be prompted to login. In most cases the user will be using the system for the first time, so they have an option to register. Once they login, the user can register their pet or view their pet's profile. Every system comes with a unique RFID tag that the pet will wear, and the user will have to input that tag when they add their pet profile.

Once the pet has been registered, the user can go to their pet's profile and choose from three main functions: "Feed", "Pictures" and "Food History". The first function would



be used if the owner wants to give their pet an extra meal without changing the original settings. The communication would start from the mobile app when the button is pressed and send "1" to the database to the "O" field. The "O" represents an override for a manual feeding. The Raspberry Pi will have a script running that is checking every 5 seconds if there are changes in the database. Since new data is pushed and pulled from the configuration file it is easy to check changes which makes it dynamic from both the mobile app and Raspberry Pi. When a new change is detected, the Raspberry Pi will attempt to send the new data until it receives a header signal from the Signal PCB. The code on the Signal PCB is primarily ran sequentially, indicating when the PCB code reaches the part of the loop where it sends a header signal, then and only then it is ready to receive the new data from the Raspberry Pi. The data will be transmitted serially byte by byte and will trigger the appropriate peripherals, in this case the DC motor that will dispense the food. The food will dispense until it hits the desired amount that the user had configured through the application.

The Signal PCB scale peripheral will return a string containing information of the amount dispensed back to the Raspberry Pi. Once the Raspberry Pi receives that scale amount, the python script will send to the database that value and send a "0" back to the override field to reset it for the next manual feeding.

The next main feature is the "Picture" button. The user has the option to view a live stream of their pet at any time of the day. This feature posed a several difficulties in how to set up a live feed through the android application. The major component that was used was the Keyestudio Camera in conjunction with the Raspberry Pi. The Raspberry Pi was chosen for better image processing through its CSI-2 bus compared to the ATMEGA328P-PU SPI Serial Interface.

The first software that was initially used was "Motion". It is a camera monitoring software that is used on the Raspberry Pi. The setup was relatively easy, but the issue was the large network latency between the live feed to a webpage. The delay at times was anywhere from 1 – 3 minutes which was unacceptable for our purposes. After configuring the output parameters, the issue for latency was still not resolved.

The solution to the latency issue was solved using a VPN tunneling service called Dataplicity. The next step was to setup the Raspberry Pi camera as a USB device rather than using the CSI bus. There was more reliable documentation on camera streaming via USB. Through this VPN tunnel, MJPEG-streamer software was compiled and configured using port 80 for HTTP service access. The initial live feed had a much lower latency with a 2-10 second delay which was in the acceptable range of delay.

Developing the live stream for the mobile application was presented a slight challenge due to the fact the video output was MJPEG and the .XML layout had a widget to only display .mp4 format. After researching different custom classes for a MJPEG viewer, configuring the live feed through the mobile application was possible.

The third main feature for the application is viewing the pet's "Food History." This feature involves a more constant communication between the embedded and software side. The user will be able to dynamically view over time how much their pet has consumed. On the mobile application, a POST request will be sent to the webserver database for retrieving the field for amount eaten. On the embedded side, every time food gets dispensed, that amount is recorded from the scale into a custom SDcard Object that is constantly recording the weight. The Raspberry Pi has a python script running pulling the string byte by byte from the SD Card reader into a variable. Once the variable contains the complete string, the data point is sent to the database which is ready for display on the mobile application.

A bonus feature that the application will have to notify the user when it is time to reorder the pet's food and when the pet's food dispenses through push notifications. These push notifications will be set up with the assistance of the Google Cloud Messaging. It will be implemented on the webserver end using a Python script pulling the appropriate information from the database. This feature would enhance user experience in terms of convenience.

C. Wireless Access Point

The system needs to be able to connect via Wi-Fi from any internet connection. It was necessary to set up the Raspberry Pi as an access point but also be able to connect to the internet. The onboard network card was configured to behave as an access point. Using that access point, now the embedded system can communicate with the database and update the fields that correspond to the configuration file.

D. Database

The database is the communication point of the whole system on both the software and embedded side. There are several tables that refer to user and pet specific information but, the table that will primarily be used is the configuration file table. The Raspberry Pi will constantly decipher and parse out specific information from the Signal PCB's SD Carder reader and constantly update when new information is detected keeping the information dynamic. The database is also a convenient way for the mobile application not to directly interact with the system, so the backend can constantly change without interfering with system functionality.

IV. Conclusion

Altogether, the automated pet feeder combines electrical hardware, relatively low-level embedded software, and high-level software to produce a pet feeding apparatus that addresses many hardships of pet ownership.

V. Group Members

Paola A. Buitrago
Paola is currently a senior at the University of Central Florida and expected to graduate in May 3rd, 2018 with a Bachelor of Science in computer engineering. She plans to attend graduate school after working in the industry for at least a year, though the University wherein she plans to attend is currently unknown.

Malcolm A. Morgan
Malcolm is currently a senior at the University of Central Florida and is expected to graduate with a Bachelor of Science in computer engineering. He plans to attend the University of Florida for a Master's in computer engineering within a year after graduation.

Hector L. Rogriguez
Hector is an Electrical Engineer student and will be graduating spring 2018. He is planning on continuing his career as an Electrical Engineer at the National Aeronautics and space Administration when he graduates.

References

[1] [1] Number of dogs in the U.S. Retrieved September 21, 2017, from
https://www.statista.com/statistics/198100/dogs-in-the-united-states-since-2000/

[2] U.S. and World Population Clock Tell us what you think. (n.d.). Retrieved September 22,
2017, from https://www.census.gov/popclock/

[3] EasyFeed Automatic Pet Feeder w/ Webcam and Amazon Delivery. (n.d.). Retrieved September 22, 2017, from https://www.kickstarter.com/projects/1214906246/gosh-easyfeed-the-100-hassle-free-automatic-pet-