# The Smart Bird Feeder

Paul Amoruso, John Hauff,

Nikki Marrow, Matthew Wilkinson

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

***Abstract*** **— This document details the design and functionality of the Smart Bird Feeder. The purpose of this device is to address the common issues regarding time input in relation to the hobby of bird feeding, and fill the space in the market with a design that addresses these concerns. The Smart Bird Feeder is designed around automatization to minimize effort required by a user while still maintaining maximum results. By designing the engineering requirements with this in mind, a base is set by which to judge the success of the project. The Smart Bird Feeder incorporates machine learning to identify and detect specific bird species, an accompanying application for notification systems, and defense mechanisms for backyard pests. The paper details what components were selected and how they impacted the overall design, as well as relevant designs and schematics for these components. Thus, by meeting or exceeding the design requirements, the goal was accomplished through the Smart Bird Feeder's creation.**

***Index Terms*** **— Machine Learning, Object Detection, PCB, Ultrasonic, Microcontroller, Pulse-width modulation, Serial Communication, Mobile Application, Autonomous.**

## I. Introduction

The design of the Smart Bird Feeder began with recognizing the common pitfalls regarding a group member's favorite hobby: bird feeding. A hobbyist could spend an indefinite amount of time waiting and watching their feeder while nothing happens, especially if they are waiting for a specific bird species to show up. This is time the hobbyist could instead spend performing other tasks. Add in the risk of backyard pests scaring away or inhibiting birds from approaching, and it's clear why the time sink for the hobby is almost unapproachable. This project's motivation is to maximize the potential of the bird watching and feeding hobby and minimize the required time commitment from the user, while still offering an enjoyable hobbyist experience.

This line of thinking led to the creation of the Smart Bird Feeder. It has been equipped with hardware that is used to train and deploy a machine learning model that can detect various species of birds and alert the user when a bird is spotted. The ability to utilize machine learning to keep track of not only when a bird arrives to feed, but what kind of species is currently feeding is essential to the automatic nature of the device. To pass on this information to the hobbyist effectively, the feeder is paired with a mobile application which includes the ability to autosave bird memories the user might have missed, categorize birds, send notifications for power and feed levels, and defense mechanisms for backyard pests.

There is no product that exists on the marketplace with a combination of all of these features, and thus the group seeks to fill this empty space by making a specialized bird feeder like no other. The Smart Bird Feeder has achieved the goals that this project was established to meet, with a focus set on maximizing potential of the hobby while requiring a minimum amount of effort.

## II. Engineering Requirements

With the overarching design goal in mind, the group set out to meet a number of engineering requirements that would emphasize the completion of that goal. The full list of engineering requirements can be found in the full paper for this project, but the three major demonstrable requirements are listed as follows.

Table 1 - Major engineering requirements

| Engineering Requirement | Specification |
|---|---|
| Accuracy of bird species and squirrel detection must exceed a certain confidence level to capture images. | ≥ 90% |
| Hatch door closing time should close within a certain amount of time. | ≤ 3 seconds |
| Users will be notified of a new bird memory within a certain time period. | ≤ 10 seconds |

The group has met each one of these requirements with The Smart Bird Feeder design. The demo requirements are set to focus on the accuracy of species detection, defense mechanism capability, and notification timeline. Further requirements specification can be found in the full paper for this project.

## III. System Concept

Detailing the concept for how The Smart Bird Feeder operates is important. A system flowchart helps to illustrate the overall predicted actions of the system.

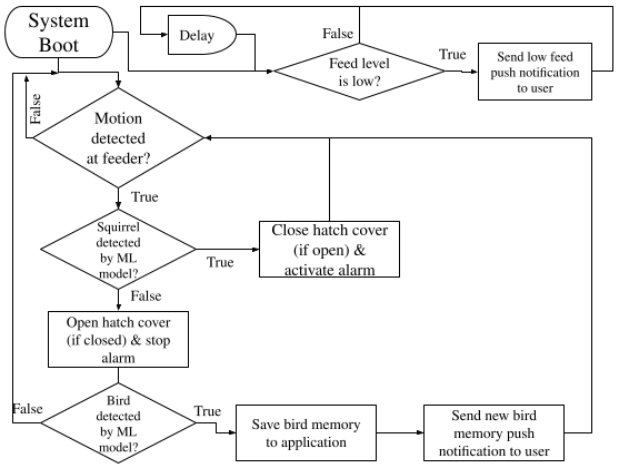


Fig 1. System flowchart

After system startup, a PIR sensor senses when *objects are present at the feeder*, and this causes the camera capture and machine learning model object detection cycle for detecting birds and squirrels to start.

The *squirrel is detected* cycle is triggered by the ML model detecting a squirrel nearby the bird feeder. This model was retrained well enough to meet the engineering requirement of a 90% or higher confidence level when detecting bird species and squirrels. The Smart Bird Feeder reacts by entering its defensive state/cycle. This means that the feed cover hatch is closed by a rotating motor, and an alarm to deter the invading squirrels is sounded. It should be noted that the closing of the hatch occurs fast enough to meet the engineering requirement established for its closing speed. The system's defense state will persist until squirrels are no longer detected by the ML model for some time. However, the command used to close the hatch and sound the alarm is not repeated while the hatch is closed, and is only given if the hatch is presently open (i.e., to switch states from non-defensive to defensive).

If a squirrel is not detected for some time, this means that the system will not go into its defense cycle, but will check if a *bird is detected*. If this is true, the system will open the feed hatch cover (if it is presently closed), which in turn causes the alarm to deactivate. The Smart Bird Feeder then communicates wirelessly with a backend API to autonomously save an image of the detected bird to a user's mobile application, and communicates with a push notification service to send a push notification to a user's phone that a new bird memory has been created. It is notable that this notification will be received by the user in 10 seconds or less, which satisfies one of this project's engineering requirements. If a bird is not found by the ML model, the system cycle simply loops back to where it began at startup.

The remaining cycle in The Smart Bird Feeder's system concept is a check for low bird feed levels using an ultrasonic range finder sensor. When *bird feed levels are low*, the system communicates with a push notification service to send a push notification to a user's phone notifying the user to refill the bird feed in The Smart Bird Feeder. It is not necessary to repeat this cycle constantly in an uninterrupted order, and so a delay is used before triggering the cycle to check feed levels again.

## IV. PCB Design & Overall Schematic

The PCB is designed to link and power all of the components for the Smart Bird Feeder. Figure 2 on the next page has a detailed schematic that breaks down how components are wired together. The DC power port, voltage regulation circuit, pin headers, sensors, speaker, and MCU circuit are shown to be connected through the PCB pins. A more in-depth breakdown of the functionality of these components is found throughout this section.

### *A. DC Power Jack Port*

The DC power Jack is a port used to connect our 12V rechargeable battery to the circuit. This jack has 3 pins, called the insertion detection pin, the tip, and the sleeve. The tip will have the positive 12V applied to it from the battery, while the sleeve acts as the 0V, or ground. When no plug is inserted, the insertion detection pin is shorted to the sleeve pin. When a plug is inserted, these two pins are no longer connected. In our design, we have shorted these two pins via traces on the PCB.

### *B. Voltage Regulation*

The components in our system need either 5V or 3.3V for operation. The only component that requires 3.3V is the microcontroller, while the Jetson Nano Developer Kit, ultrasonic sensor, speaker, micro servo motor, and PIR sensor all require 5V for operation. Since most components are utilizing 5V, we have two regulators supplying 5V and one regulator supplying 3.3V. These regulators, the LM1084, convert 12V from the battery input to the respective 5V and 3.3V. There are two regulators for supplying 5V, and one for the 3.3V. One of the two 5V regulators supplies current to the Jetson Nano, while the other supplies current to the rest of the peripherals. This was the chosen configuration since the Jetson Nano Developer Kit consumes the most power out of all components.

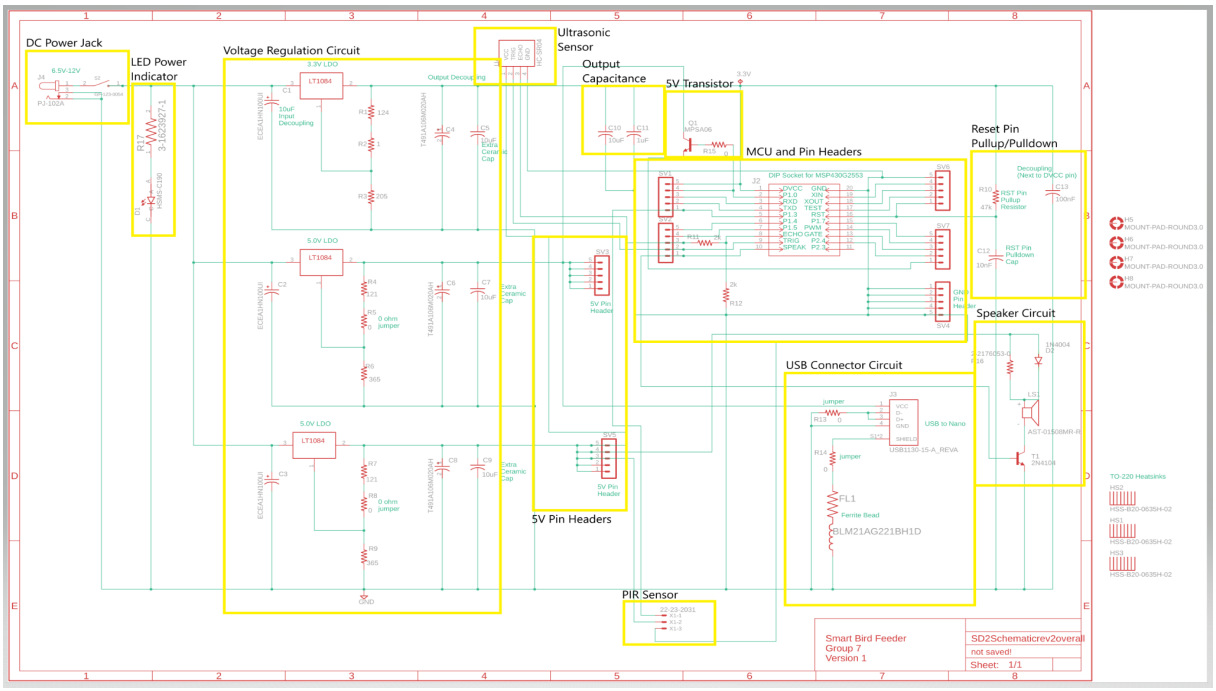


Fig 2. Overall Schematic

### *C. Output Capacitance & Reset Pin Pullup/Pulldown*

According to the MSP430G2553 documentation [13], the reset pin requires a 47k ohm pullup resistor and a 10nF pulldown capacitor. This is in the PCB design, in addition to a 10uF and 1uF capacitors to help with decoupling.

### *D. USB Power Output Connector*

The USB connector provides 5V from the output of the voltage regulation circuit to the power input of the Jetson Nano Developer Kit. A ferrite bead was added between the shield of the USB connector and ground in order to prevent noise interference.

### *E. Fielect Piezo Alarm*

The piezoelectric alarm is an additive component to the defense mechanism to ensure that squirrels choose not to linger after the hatch door prevents access to the bird feed. The less time that backyard pest spends near the bird feeder, the greater the usefulness of the device. Squirrels and other small rodents are easily spooked by sudden loud sounds. The requirements for the alarm started with alarm safety requirements, ensuring that the alarm was not louder than 90dB. Since the regulators on the PCB circuit output 5 Volts, it was necessary to find an alarm rated at 5 Volts. The chosen piezo alarm is rated at 80dB at 5 V (5mA), making it great for this particular application.

The alarm functions by converting electrical energy into mechanical energy. The applied voltage will induce a stress inside the metal diaphragm to undergo tension or compression. The design connects the speaker to the MSP430 via pins on the PCB, and will control the speaker using PWM (Pulse Width Modulation) to reverberate the metal diaphragm. The speaker is set to go off when a squirrel is detected by the Jetson Nano, which will send a command via UART to the MCU, which will then provide a PWM signal to the alarm causing it to blare until a squirrel is no longer present.

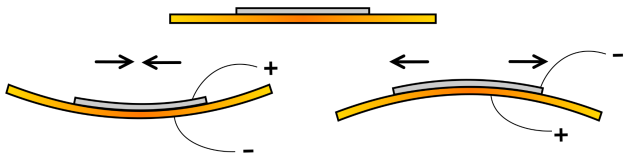


Fig 3. The Piezoelectric Effect [12]

### *F. HC-SR04 Ultrasonic Sensor*

The ultrasonic sensor is used to monitor the feed levels within the bird feeding chamber. The team started with the proposed functionalities to define what the required specs would be for the sensor. Given the constrained space inside the chamber, a sensor that could accurately and consistently measure distance readings in spaces as small as 4 - 9 inches away from the sensor location was important. The group also wanted to ensure that the current draw for the device was low given the unit was powered by a battery system.

The HC-SR04 Ultrasonic Sensor exceeded expectations. With a minimum detection range of 2 cm, the enclosed

space concern was promptly addressed. The device also has an accuracy of 3 mm which is precise enough for the application. The current draw of the device is 15 mA.

The device is set up to communicate with the MSP430 using PWM (Pulse width modulation). The sensor works by sending out a signal from its transmitter at an 8 cycle burst of ultrasound waves at 40 kHz and then receiving its reflected signal back to its receiver. This signal is then sent from the Echo pin of the sensor to the MCU. The pins from the sensor to the MCU are Vcc, Trig, Echo, and GND. The Trig pin is responsible for receiving the start signal from the MSP430. In order to save power, the sensor is only active on startup and when pulsed by the MSP430. The MSP430 is commanded by the Jetson Nano to send a pulse to the Ultrasonic device periodically to monitor bird feed levels.

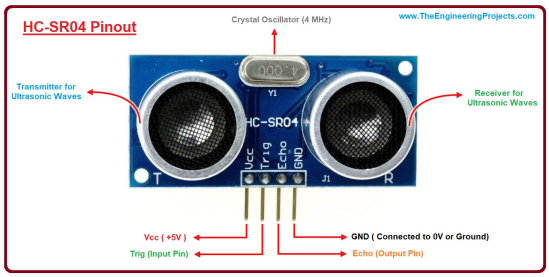


Fig 4. HC-SR04 Sensor [6]

*G. PIR Sensor*

The PIR sensor is a bonus feature which was established as a stretch goal to make the Smart Bird Feeder more power efficient by only having it run the intensive machine learning object detection cycle only when the PIR sensor detects something at the perch.

In the design, a PIR sensor is placed on the bird feeder facing the perch. The sensor uses a pair of pyroelectric sensors to detect heat energy in the surrounding environment. These two sensors sit next to each other, and when the signal differential between them fluctuates, the sensor will engage. The MSP430 becomes aware of this by using its ADC module, which can read analog voltage. The ADC pin on the MSP430 is connected to the PIR sensor's signal pin. This signal pin outputs a logic high to the MSP430 if an object is detected. Once the MSP430 is aware that a bird or squirrel has arrived at the perch, it sends a UART message to the development kit, notifying the kit that it should start its object detection program.

*H. MSP430G2553 Microcontroller*

While most of the processing power for the Smart Bird Feeder is handled by the Jetson Nano, the MSP430 is the hub of any and all interaction between devices and peripherals for the project. The MSP430 software is responsible for gathering input data from the Nano, PIR sensor, and ultrasonic sensor, and outputting signals to the alarm and motor.

The focused requirements when choosing a MCU were available pins and low current consumption. The processor speed was not of concern, as there are a minimal amount of computations the MCU needs to perform in the design. The MSP430 is an ultra-low-power microcontroller with 16-bit RISC CPU and 16-bit registers. Consuming only 203uA at 3.3 V, it's a strong choice regarding power consumption. The 24 available pins were more than enough for the design. The group's familiarity with this line of processors also made the design and testing phase go by more smoothly.

The software written in C for the MSP430 is responsible for interpreting commands received from the Nano, and then executing the action corresponding to the command. These actions include opening/closing the hatch using the servo motor, taking readings of the bird feed level with the ultrasonic sensor, and playing tones on the piezoelectric speaker. The only action that does not require a command from the Nano relates to the PIR sensor. The MSP430 measures the voltage level emitting from the PIR sensor, and only when this voltage goes high, corresponding to an animal at the feeder's perch, will the MSP430 then send an alert to the Jetson Nano. The Jetson Nano will then resume object detection.

*I. SG90 Micro Servo Motor*

The servo motor is a defense mechanism component used to rotate the hatch door which blocks access to the feed under certain conditions. This component directly relates to one of the demoable requirements. Given the hatch door needs to close in 3 seconds or less, the speed of the motor needs to suit that requirement.

The SG90 Micro Servo Motor max speed is rated at 0.1 sec/60° which is ample speed to achieve the 90° rotational turning distance. It also has a low current consumption of 0.2 A.

The SG90 servo uses PWM to communicate with the MCU. Supplying the motor with a certain pulse will result in the desired rotational movement. In this application, the motor supplied with a 1.5 ms pulse will rotate approximately 90° in either direction depending on the period of the PWM signal. The MCU determines the necessary action based on readings from the Jetson Nano using UART communication.

## V. Object Detection Technology Concept and Design

Proper care and consideration regarding the choice of technologies to use for object detection via machine

learning was imperative to the success of the project. Without adequate components to perform these tasks, none of the software developed for The Smart Bird Feeder would function properly.

### *A. NVIDIA Jetson Nano Developer Kit*

The NVIDIA Jetson Nano Developer Kit 4GB model was selected to run the software for object detection and mobile application interaction for The Smart Bird Feeder. The Jetson Nano includes a 128-core NVIDIA Maxwell™ GPU and a Quad-core ARM® A57 CPU, which are perfectly suited to perform the aforementioned tasks. When selecting a product to perform these tasks, the Jetson Nano Developer Kit was compared with similar Edge devices built to perform machine learning inference, such as the Arduino Nano 33 BLE Sense, the Coral Dev Board, and a Raspberry Pi combined with an Intel Neural Compute Stick 2. However, for the price of $99 at MSRP, the Jetson Nano offered the best capability in terms of RAM (4GB), CPU and GPU performance, and community support, when placed next to its competitors that were priced similarly or higher.

### *B. Raspberry Pi Camera Module V2*

The camera used for The Smart Bird Feeder to capture images of birds is the Raspberry Pi Camera Module V2. This camera has a Sony IMX219 8-megapixel sensor, and provides clear high-quality image and video capture for The Smart Bird Feeder. It connects to and interfaces with the Jetson Nano via a MIPI CSI-2 connector.

### *C. Pytorch SSD-MobileNet & Jetson Nano Software*

The software for object detection of birds and squirrels was written in Python and deployed on the Jetson Nano Developer Kit. The NVIDIA TensorRT SDK built on CUDA that comes packaged within the JetPack SDK [7] was utilized for this project, since it packages together many of the software components required for performing object detection, and some team members already have experience with CUDA. Figure 5 shows the MobileNet-SSD network architecture that was retrained to meet the requirements for object detection of 15 of the most common bird species in Florida backyards [5] and squirrels. The ML model also detects Green-cheeked Parakeets, since a team member owns a bird of this species, and so it is convenient to run tests and show demos with this species of bird.

For retraining the MobileNet-SSD model, the Pytorch-SSD programs, documents, and tutorials provided by NVIDIA in the *jetson-inference* [3] project were especially helpful. The *jetson-inference* project included an open-use training Python script that was utilized to retrain the MobileNet-SSD model.

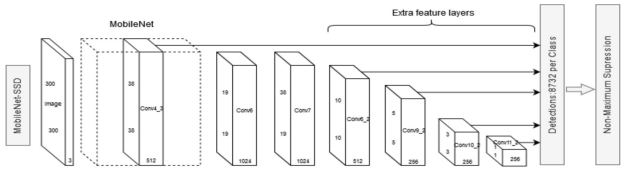


Fig 5. MobileNet-SSD architecture [2]

The software created to be run on the Jetson Nano for The Smart Bird Feeder was written exclusively using Python. The program that is primarily responsible for the Jetson Nano's actions with regard to the bird feeder system (aptly named *smart_feeder_run.py*) uses various Python APIs such as PySerial, PyTime, OpenCV, and NumPy, to operate and handle the various responsibilities of the Jetson Nano in the overall system of The Smart Bird Feeder. These APIs are used to establish serial communication between the Jetson Nano and the MSP430G2553 on the custom PCB, track time intervals for triggering certain events, and to convert captured images to a proper format for storing. The live camera feed I/O and this project's implementation of the retrained ML model are located in this program as well.

Besides the *smart_feeder_run.py* program, the Jetson Nano also occasionally runs two other Python scripts. One script is responsible for locally saving birds memories captured by the feeder's camera and sending the saved image to a cloud NoSQL database (MongoDB) to be viewable from a user's Smart Bird Feeder mobile application. The other Python script interacts with a server hosted by the Expo push notification service to notify a user's phone of important events, such as low feed container levels and new bird memories being created.

## VI. Birdhouse Design

Design began with the purchase of a basic bird feeder house, in which all of the components were emptied out to be replaced with the new design. In order to provide an optimum viewing angle for the camera, the feed trough was placed outside of the bird feeder. To do this, an additional perch was added to extend past the feeder bottom. A bird feed container and a feed fill spout were then both added to the birdhouse. These two components utilize gravity to source the feed from the fill spout opening, into the feed container, down the container's inclined ramp, and finally into the trough on the front of the birdhouse. The feed cover hatch was placed outside on the front of the birdhouse to block access to the feed, so the micro servo motor was mounted to the exterior of the

feeder and connected to the hatch. A perch for the birds to stand on and feed from was also added in front of the feeding trough. Finally, this design calls for all electrical components besides the servo motor to be stored on the interior of the birdhouse to protect from interference from entities such as animals and weather. The ultrasonic range finder sensor was mounted above the bird feed container to easily measure feed levels.

Fig 6. The Smart Bird Feeder assembled birdhouse

## VII. Application Design

This section of the paper details the design of the accompanying application to the Smart Bird Feeder. This application is essential to achieving the design goal, as it is the interface by which the user interacts with all of the automatic features included in the package. The primary functionality of the app is to notify users of the species of birds visiting the feeder, and to save a memory associated with each visit. The application also acts as a notification system to alert the user that the bird feed container is low and needs to be refilled. These two key components ensure that the user isn't required to watch the feeder during downtime, and in doing so furthers the design goal.

### *A. Frontend Application*

The frontend was built with simplicity and functionality in mind. A user will first be asked to login or create an account. The app is designed with a pleasant color scheme and utilizes common UI symbols to deliver important information. Notifications will prompt the user when the feed is low and a bird visits the feeder.

The Smart Bird Feeder's mobile application was developed using the framework of Expo and React Native together. Expo provided helpful components and APIs to make development faster and simpler than if a bare React Native workflow was used. The frontend design includes a user login screen and signup screen, along with a welcome screen to greet the user, along with a *Bird Memories* screen which allows the user to view an organized collection of all of their autonomously captured bird memories.

### *B. Backend Application*

The backend application for The Smart Bird Feeder was written and deployed with JavaScript and the Express framework for the NodeJS runtime environment. This backend application handles all of the routes required to provide proper interaction between The Smart Bird Feeder, a MongoDB cloud database for storing documents, and a user's mobile application. These routes include user login/signup, as well as storing and retrieving bird memories from the cloud database where they are to be saved. This backend application uses a library called Mongoose to build a model for a new user or new bird memory, and a middleware called Multer is used to store the modeled information in noSQL documents on the MongoDB cloud database.

## VIII. Communication

When discussing the communication aspects of the Smart Bird Feeder, it's important to first explain that there are two communications protocols in use. The first communication protocol addressed is the serial communication between the Jetson Nano and the PCB, while the second is to connect to the internet so that the Jetson Nano can communicate with users of the Smart Bird Feeder. The communication between the Jetson Nano and the microcontroller is vital to the performance of this device, as it is important to have information be sent back and forth to operate the peripherals and collect data. It is also important that the user is able to be virtually anywhere and still be up to date with the status of the Smart Bird Feeder, thus wireless communication technology to send notifications, and an updated memory gallery on the mobile application were implemented. Therefore, the following two subsections dive deeper in the justification and data needed to be transferred by the UART serial communication and the REST API wireless communication.

### *A. UART Serial Communication*

UART stands for Universal Asynchronous Reception and Transmission. When deciding what protocol for serial communication was best, focus was placed on simplicity and developer preference. First, it was necessary to identify what was being transferred between the Jetson Nano and the MSP430 microcontroller. There are only a few things that must be transmitted between the two devices. The microcontroller must reply with the status of

the feed level when asked by the Jetson Nano, close/open the hatch when instructed, sound the alarm, and tell the Jetson Nano when the PIR sensor has been tripped so that the object detection cycle may run.

The Jetson Nano and MSP430G2553 only transmit and receive messages containing a single Byte-sized character. The Jetson Nano transmits a single Byte message to the MSP430 to check for the following reasons:

- Ask if bird feed needs to be refilled
- Tell the MSP430 to open/close the feed hatch and sound the alarm

The MSP430G2553 receives one of these messages, which interrupts it from its low power mode, then it will either pulse the servo motor, or pulse the ultrasonic rangefinder sensor and interpret the readings to determine whether or not the bird feed is low. The MSP430 transmits Byte messages to the Jetson Nano for the following reasons:

- Tell the Jetson Nano that feed is or is not low
- Tell the Jetson Nano to run object detection cycle

### *B. Wireless Communication*

The Jetson Nano performs all wireless communication for The Smart Bird Feeder, as it contains the WI-FI module. Such wireless communication is executed solely to either store a newly created bird image from The Smart Bird Feeder device onto a MongoDB noSQL cloud database, or to communicate with an Expo push notification backend server to push a notification to a user. The programs running on the Jetson Nano use REST API calls to backend apps to direct the communication.

When it comes to the user side of things and making the Smart Bird Feeder keep users up to date with the status of the bird feeder, it is important that a wireless communication protocol is able to have a range acceptable for being outside the house and still be able to update the database and send notifications. The reasoning behind choosing wireless vs wired communication is mainly due to the preference that aimed for the bird feeder to be as user friendly and easy to install as any other competitive similar products. Therefore, to reduce the complication that comes with running wires outside, it was decided to keep the Smart Bird Feeder wireless and utilize WI-FI to transmit data.

In consideration and eventual justification for using WI-FI for the wireless protocol, it was important for the group to look at the range to determine the usefulness of implementation. After looking at the range the next consideration is ease of implementation. Therefore, once looking at the IEEE 802.11 standard for WI-FI, and understanding that the 2.4 GHz band can reach up to around 150 feet indoors and around 300 feet outdoors, which is ideal for this implementation, the only thing next is how to integrate it within the bird feeder. Luckily, the Intel Dual Band Wireless-AC 8265 chip was installed to the Jetson Nano to add WIFI and Bluetooth, with many documentations describing WI-FI to connect to the internet and send data to the cloud and mobile application.

## IX. Power

Power is provided to the Smart Bird Feeder using a TalentCell rechargeable 12V 6000mAh battery pack. The PCB is outfitted with a barrel jack port which connects directly to the battery pack to provide the regulators with 12 Volt to be converted to 5 or 3.3 Volts. As mentioned above, these regulators provide power to the system.

Table 2 - System hardware power requirements

| | **Voltage Requirement** | **Current Requirement** |
|---|---|---|
| **Jetson Nano** | ~5 V | 0.4 - 1.1 A |
| **MCU** | 1.8 - 3.6 V | Operating: 230 uA<br>Standby: 0.5 uA |
| **Servo Motor** | 4.8 - 6 V | 500 mA |
| **Speaker** | 5 V | 4 - 6 mA |
| **Fan** | 12 V | 0.137 A |
| **Ultrasonic Sensor** | 5 V | 15 mA |
| **PIR Sensor** | 5 V | 65 mA |
| **Total** | 3.6 - 12 V | 1.2 - 1.8 A |

Power consumption has been a major focal point for this project, as the NVIDIA Jetson Nano draws upwards of 1.1 Amps while running its smart bird feeder programs. Since the device is required to be placed outdoors, a battery based power source was a requirement for functionality. The Smart Bird Feeder consumes a maximum of 1.8 A on average. However, in an effort to lower power consumption, a sensor was added to only activate the machine learning module when movement is detected. This feature lowers the power consumption of the device to around 0.5 A on average, which correlates to 60% savings in battery life. The battery pack is therefore able to run the system in a range of 3 hours 20 min to 6 hours 40 mins, which satisfies an engineering requirement specified in the full paper for this project.

## X. Conclusion

The purpose of the Smart Bird Feeder is to address the common concerns surrounding the hobby of bird feeding by creating a device that would minimize required time from the user while still showing optimal results. By establishing engineering requirements that when met would achieve this goal, the group defined a solid metric by which to gauge the success of the project. This paper details the methods and design choices by which this functionality was achieved. Thus, by creating a device that meets or exceeds these requirements, the original design goals were met through the creation of the Smart Bird Feeder.


## Acknowledgement

The authors wish to acknowledge the assistance and support of the instructors of the Senior Design course: Dr. Samuel Richie and Dr. Lei Wei. The authors would also like to offer thanks to the University of Central Florida faculty who were kind enough to review this project.


## References


[1] About IPC. IPC International, Inc. (2020, November 19). https://www.ipc.org/about-ipc.

[2] Dusty-Nv. (2021, May 21). Jetson-inference/pytorch-ssd.md at master · Dusty-NV/Jetson-inference. GitHub. Retrieved November 11, 2021, from https://github.com/dusty-nv/jetson-inference/blob/master/docs/pytorch-ssd.md.

[3] Dusty-Nv. (n.d.). dusty-nv/jetson-inference: Hello AI World guide to deploying deep-learning inference networks and deep vision primitives with TensorRT and NVIDIA Jetson. GitHub. https://github.com/dusty-nv/jetson-inference.

[4] EBird - discover a new world of birding... (n.d.). Retrieved November 11, 2021, from http://ebird.org/home.

[5] Gillson, G. (2019, July 17). 26+ common backyard birds of Florida (Photos, ID). 26+ Common backyard birds of Florida (Photos, ID). https://www.whatbirdsareinmybackyard.com/2019/07/most-common-backyard-birds-of-florida.html.

[6] HC-SR04 Pin Details. (n.d.). [Illustration]. https://microcontrollerslab.com/hc-sr04-ultrasonic-sensor-interfacing-with-arduino-distance-measurement-example/

[7] "IEEE Standard for Information Technology--Telecommunications and Information Exchange between Systems - Local and Metropolitan Area Networks--Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," in IEEE Std 802.11-2020 (Revision of IEEE Std 802.11-2016) , vol., no., pp.1-4379, 26 Feb. 2021, doi: 10.1109/IEEESTD.2021.9363693.

[8] Jetpack 4.5 Archive. NVIDIA Developer. (2021, January 22). https://developer.nvidia.com/jetpack-sdk-45-archive.

[9] JetPack SDK. NVIDIA Developer. (2021, May 24). https://developer.nvidia.com/embedded/jetpack.

[10] NVIDIA. (2021, July 21). Jetson download Center. NVIDIA Developer. https://developer.nvidia.com/embedded/downloads.

[11] NVIDIA Corporation, "DATA SHEET NVIDIA Jetson Nano System-on-Module" | DA-09366-001_v1.0 datasheet, Feb. 2020.

[12] PiezoBendingPrinciple.svg. (2011, February 17). [Illustration]. https://commons.wikimedia.org/wiki/File:PiezoBendingPrinciple.svg

[13] Texas Instruments. (2011, April). SLAS735J MIXED SIGNAL MICROCONTROLLER. Dallas. https://www.ti.com/lit/ds/symlink/msp430g2553.pdf?ts=1636685136752&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FMSP430G2553

[14] Wah C., Branson S., Welinder P., Perona P., Belongie S. "The Caltech-UCSD Birds-200-2011 Dataset." Computation & Neural Systems Technical Report, CNS-TR-2011-001.


## Biography

Paul Amoruso is a Computer Engineering student at the University of Central Florida planning to graduate this semester with his bachelor's degree. After graduating, he will pursue his Master's degree in Computer Engineering starting Spring of 2022 at UCF. Other interests include Computer Architecture topics and gardening in the greenhouse he built.

John Hauff is a Senior baccalaureate student in Computer Engineering at the University of Central Florida. His interests include software development and embedded systems, and mentoring others in these areas. After graduating, John hopes to begin his career as a computer engineer or software engineer in the Orlando area.

Nikki Marrow is an Electrical Engineering student at the University of Central Florida. She is currently working as a firmware engineer at Lockheed Martin in Orlando, previously working on circuit design and testing. After graduating, she plans to work at Lockheed full-time. Other interests of hers include research in surface science and molecular dynamics.

Matthew Wilkinson is a graduating Computer Engineering student from the University of Central Florida. Following graduation he will be working for Disney's Design and Engineering team as a Controls Engineer. He hopes to continue pursuing a career path in controls with a focus in entertainment.