BIRD BOX DESIGN DOCUMENT

February 22, 2019

Kevin Chen, María Nacenta Fernández, Michael Zhang
ECE 445 - Spring 2019
TA: Christopher Horn
Team 39

Contents

1 Introduction

1.1 Objective

Observation of animal behaviors and responses to certain audio stimuli can become the backbone for how our technologies are shaped or how our secret codes are constructed. Researchers at the University of Illinois are working to uncover patterns from these behaviors among birds through the means of conditioning. However, researchers are faced with a dire problem - there is no existing system that would perfectly cater towards their research needs and is cost-efficient. Thus, the need to build a system suitable for their research necessities becomes increasingly apparent.

The solution would be a system comprised of a a hardware and software interface. The software interface will accept parameters fed through a Graphical User Interface (GUI) to construct a unique file (with a certain number of trials) set by the researcher. The hardware would then respond to the data provided by the research in the GUI to play audio sounds for the bird to respond to. The bird would provide responses though color-differentiated buttons and trigger certain outcomes from the system. The bird would solely interact with the hardware side of the system. The system will be designed to reward the bird with food upon favorable action and punish the bird by turning the lights off upon unfavorable action. At the end of the research period, an excel sheet wold be generated documenting the results of each unique response that the bird provided.

1.2 Background

Modern technology has evolved at an incredible rate and digital signal processing is no exception to this rapid growth. With this growth in technology, it is important to also observe natural aspects regarding the field to draw more inspiration for advancements in signal processing. Thus, the analysis of bird behavior and responses to certain audio stimuli becomes a valuable observation for furthering knowledge in this field of study.

To highlight the problem, there will be varying tiers of impacts to provide emphasis on the scale in which the project contribution can help with understanding this field. Creating a product to suit the needs of researchers will unlock further contribution towards various insights within the field. The trials the system would help conduct further enables understanding of bird communication, which can be applied to save certain endangered specifies upon identifying a certain cry - from a bird, or, in a broader sense, this can contribute towards how language is perceived among birds - how the communicate amongst each other and how certain sounds are assigned meanings [1].

1.3 Experiment Procedure and Various Terminology

The experiment is composed of a variable amount of trials specified by the researcher. Each trial can be classified according to if can be called a sham. A legitimate trial will eventually play an audio track that differentiates from the background sound while a sham trial will never play this differentiating cue and continue to play the background sound until the next trial is started.

1.4 High Level Requirements

- The overall device must dispense food to the tested subject within a latency period of 2 seconds of successful trial completion - A trial is counted as successful if the trial is not designated as a sham and the subject correctly presses the Trial Attempt Button after a specified audio cue.

- The overall device must shut of cage lighting in the event that the tested subject failed a trial. Trial failure is defined strictly as the subject pressing the Trial Attempt Button during a trial specified as a sham.

- The device must not punish inaction from the subjects end, and, in the event of extended inaction, must alert the researchers that the trials have failed to initiate before shutting itself off (leaving a light on to not stress the bird).

- The software must be capable of accepting .wav file inputs for audio signals and produce an excel sheet briefing the results from the subject-system interaction as an output.

- The project must be cost-efficient and within a $500 budget as existing devices exist for a far more extravagant price.

2 Hardware Specifications

The hardware component can be broken into four major modules as shown in Figure 1. Additionally, all hardware, with exception of the computer speakers, will be inside of the testing environment as shown in Figure 2. It is assumed that a host-computer will always be within proximity of the testing environment to support our block diagram. We have informed our sponsor of this development and have reached an agreement with regards to this.

The connections in Figure 1 do not necessarily denote data buses for data connections or the data and power outputs managed by a singular input. For the sake of simplicity and compactness, the block diagram merely denotes a connection between one node to another with the connection intent denoted. Specifics of each connection are purposefully left unspecified and are detailed in greater length in each individual section via schematic diagram.

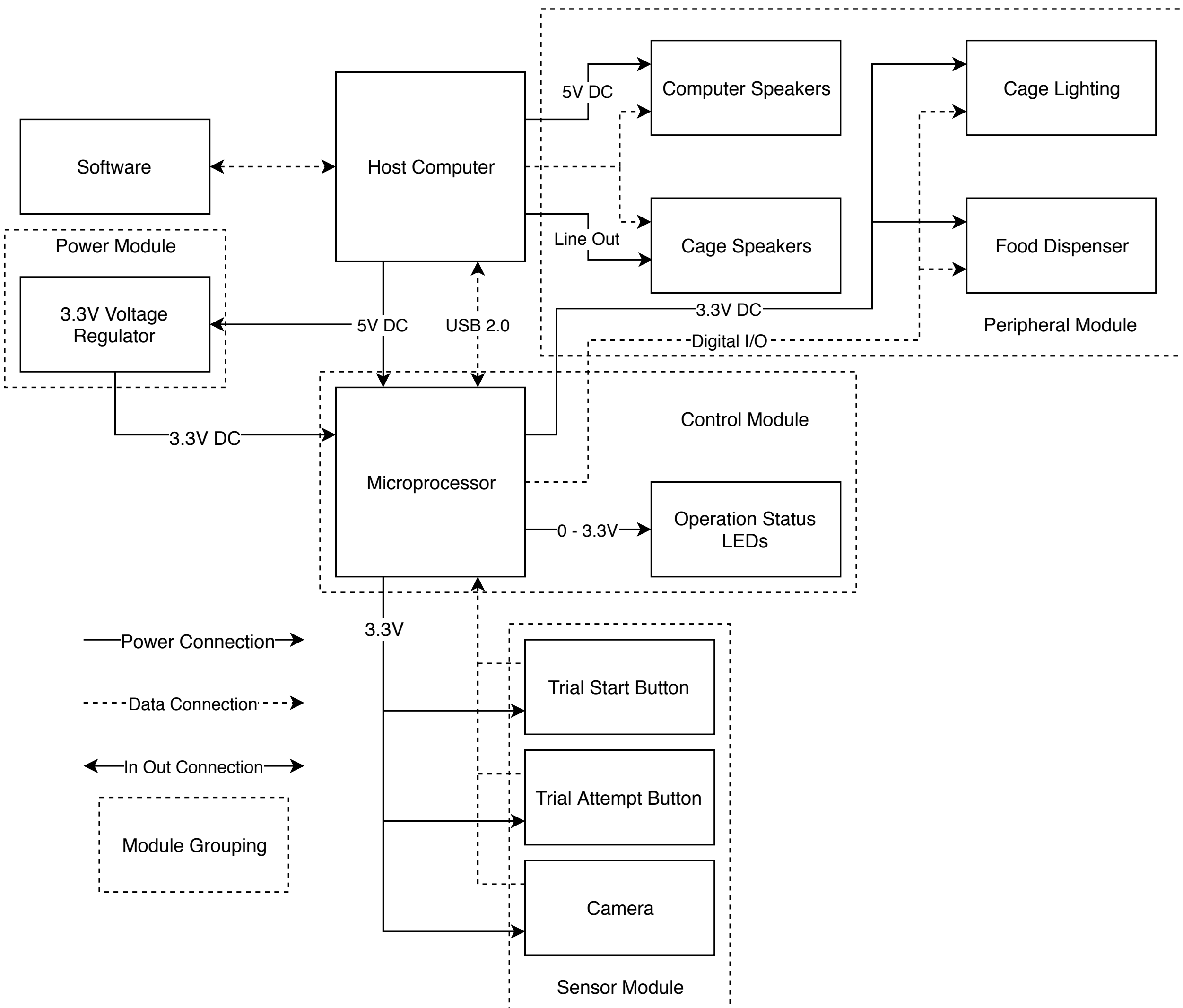

Figure 1: Block Diagram of Hardware Modules and Component Nodes

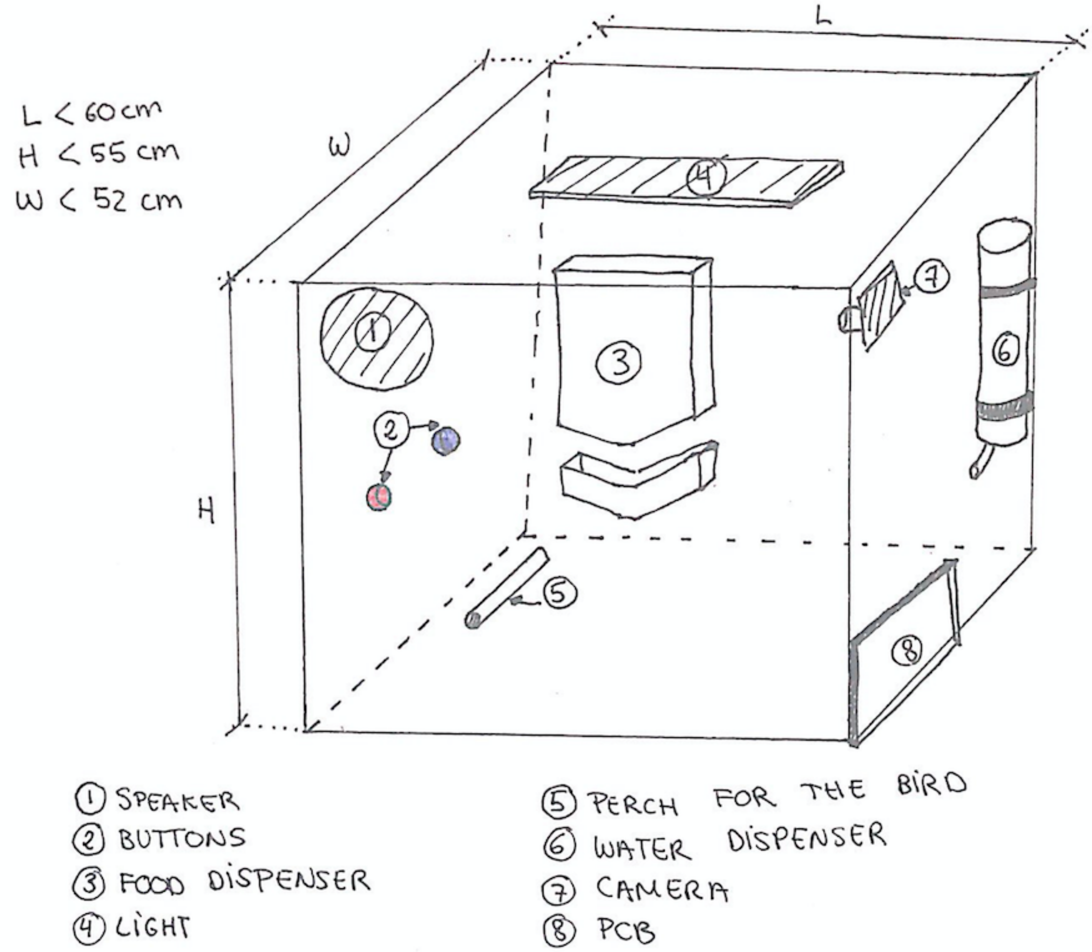

Figure 2: Physical Dimensions of Testing Environment and Placement of Nodes Within

2.1 Power Module

The power must convert standard USB 5V into 3.3V through a voltage regulator while still allowing a 5V channel to connect to the control module to facilitate data transfer between the software and control module. To execute that operation, the external power is supplied with USB 2.0 (we will assume that the device is always connected to this power source during operation). With this design consideration in mind, we eliminate the need for a battery or power storage component.

Our power consumption is averaged at around 2.325W with an average of 250mA at 3.3V and 300mA at 5.0V. The 3.3V consumption is justified because it is consistent with a majority of our hardware, encompassing the peripheral, sensor, and control modules. The 5V consumption is chosen to supply power to facilitate the data transfer between the software and control module. These values adhere to the maximum load able to be drawn from a USB 2.0 port - 500mA at 5V or 2.5W.

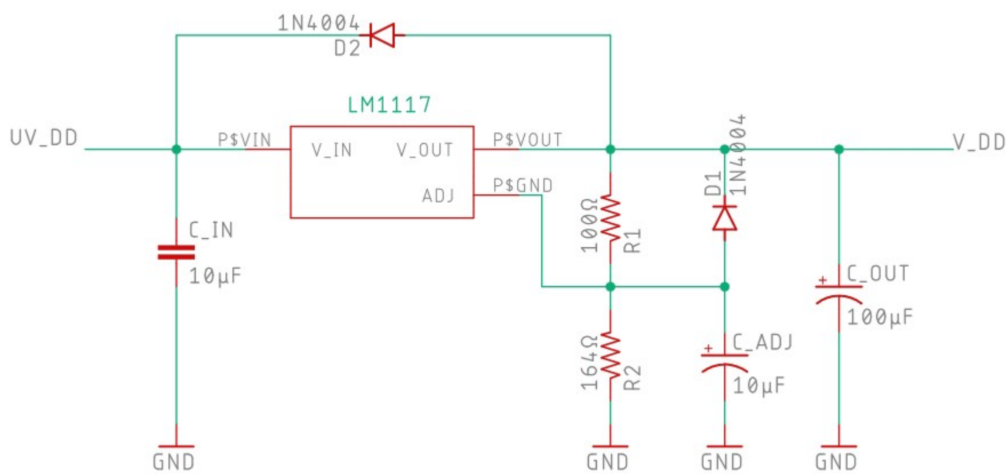

Figure 3: Circuit Schematic of the Power Module

Figure 3 outlines the external components needed to properly regulate this module to work in accordance with the rest. C_{in} and C_{out} are set according to the datasheet requirements for the LM1117 chip that we choose to utilize. C_{ADJ} helps improved ripple rejection when set to its state capacitance. D_1 helps prevent breakdown of our device if V_{DD} is shorted due to capacitative charge while D_2 does the same in case if UV_{DD} does the same.

2.1.1 5V Power Supply

This node is significant enough to warrant its own piece in the specifications for this module. The bulk of our hardware components is powered by this node and the transfer of information via USB 2.0 (from the software and control modules) is also facilitated by this node. As these are two of the most vital components in our design, we must make sure that both of these qualities are functional.

Table 1: Requirements and Verification Processes of the USB 2.0 Power Supply

Requirements	Verification Process
- Outputs a maximum of 500mA at 5 ± 0.25V at all times when connected to an external device	- Attach equivalent load to when device is at peak power consumption to USB port Measure current using an ammeter is series - Measure open circuit voltage

2.1.2 3.3V Voltage Regulator

The low dropout regulator supplies 3.3V for the corresponding components in which it is required from an input voltage of 5.0 ± 0.25V. The classic LM1117 must be able to handle an input voltage at the theoretical low for USB 2.0 (4.75V) as well as the maximum (5.25V) at peak current draw (500mA) [2].

Table 2: Requirements and Verification Processes of the 3.3V Voltage Regulator

Requirements	Verification Process
- Provides a $3.3 \pm 5\%$ output from a 5.0 $\pm$ 5% input source	- Attach the voltage regulator to an equivalent load - Connect the output terminal of the voltage regulator to an oscilloscope and measure the voltage fluctuations at maximum draw
- Operates with a current draw between 0 - 300mA	- Using an oscilloscope as a resistive sweep, measure the current with a varying resistance until it reaches the lowest that the device does naturally
- Maintains a temperature below 80C during trialing	- Use an IR thermometer to ensure device temperature is below upper bound after a standard testing session (About an Hour).

2.2 Control Module

The control module handles communication between the onboard devices as well as connected devices and software. It is powered by the power module and will disable itself and the device if voltage and current are significantly out of operation bounds or if device temperature is not within safe levels for the chip. This module consumes approximately 50mA ± 5% between its blocks.

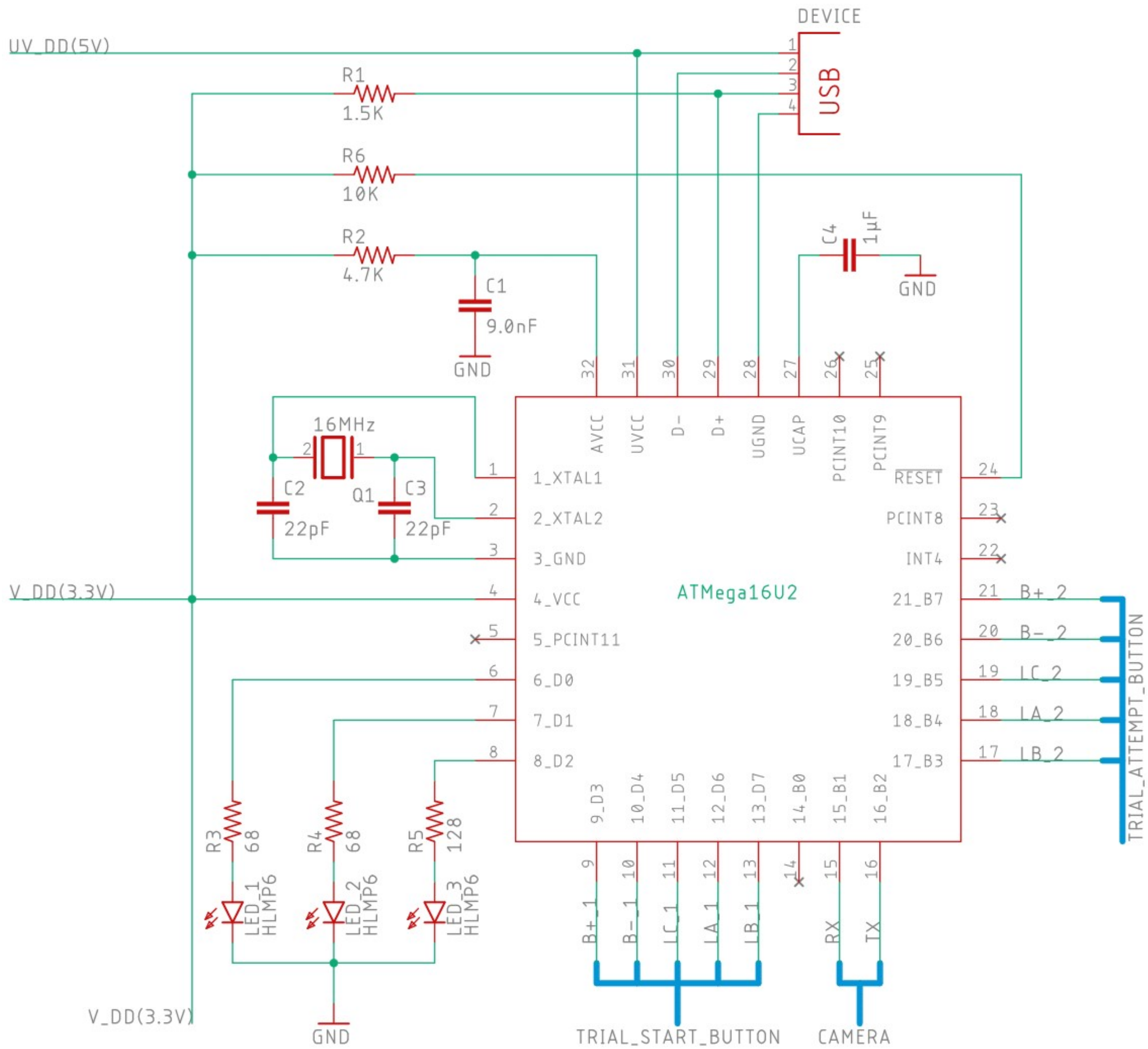

Figure 4: Circuit Schematic of the Control Module

Figure 4 is a circuit schematic of the internal wiring of this module. The microcontroller drives a majority of the components, three operating LEDs are driven through digital pins depending on the current operations and conditions. A crystal filter removes the internal clock's remaining signal from the chips ground terminal. Our chip also fully supports high-speed USB communication and as such, we also have a pull-up resistor for the D+ port.

2.2.1 Microcontroller

The microcontroller, ATmega16U2 handles processing commands from both input sensors from the sensor module, as well as commands from software to active components in the peripheral module [4]. The communication with the sensor module is done with UART and the communication with the peripheral module is done with SPI. This microcontroller was chosen for its data retention (20yr 85C) as well as its ability to communicate with a USB 2.0 interface. The chip contains two SPI interfaces for required peripherals and a singular UART interface. Additionally, the microprocessor monitors onboard voltage and current and will disable functionality if out of operational bounds. 16kB of on chip flash allows for programming without an external NAND flash chip.

Table 3: Requirements and Verification Processes of the Microcontroller

Requirements	Verification Process
- Shutoff at nonstandard voltages and currents	- Use a voltage generator and set voltage out of bounds ($>$ 5.25V) and ensure nono power is supplied by the microcontroller to other components - Use a current generator and set current out of bounds ($>$ 500mA) and ensure no power is supplied by the microcontroller
- Can accurately transfer 32 bits of data via USB within 10ms	- Connect the microcontroller with a host computer and open a terminal on said host - Send a 32 bit character from computer - Using the computer, Request a 32 bit character from the microcontroller Check the data's accuracy as well as the time stamps

2.2.2 Operation Status LEDs

Status LEDs powered by the microcontroller will display whether or not the circuit is operational during trialing and if the microcontroller is sending data via USB [8]. Consists of three seperate LEDs for the following events: circuit is operational, microprocessor is sending data, a trial is currently being run [9].

Table 4: Requirements and Verification Processes of the Operation Status LEDs

Requirements	Verification Process
- All LEDs must be visible from 3m away	- Drive each LED circuit with an equivalent 3.3V source - Stand a distance of 3m form the LED and ensure it is visible

2.3 Sensor Module

The sensor module receives impulses from the trialed specimen via physical buttons. These buttons must be differentiable to the specimen in addition to easily-pressable. The impulse created by the buttons is detected via UART and sent to the control module. A camera is mounted to the host computer directly via USB 2.0 in order for video footage of the trial to be sent to the researcher to determine whether or not the trial is proceeding as intended or if blocked by some factor.

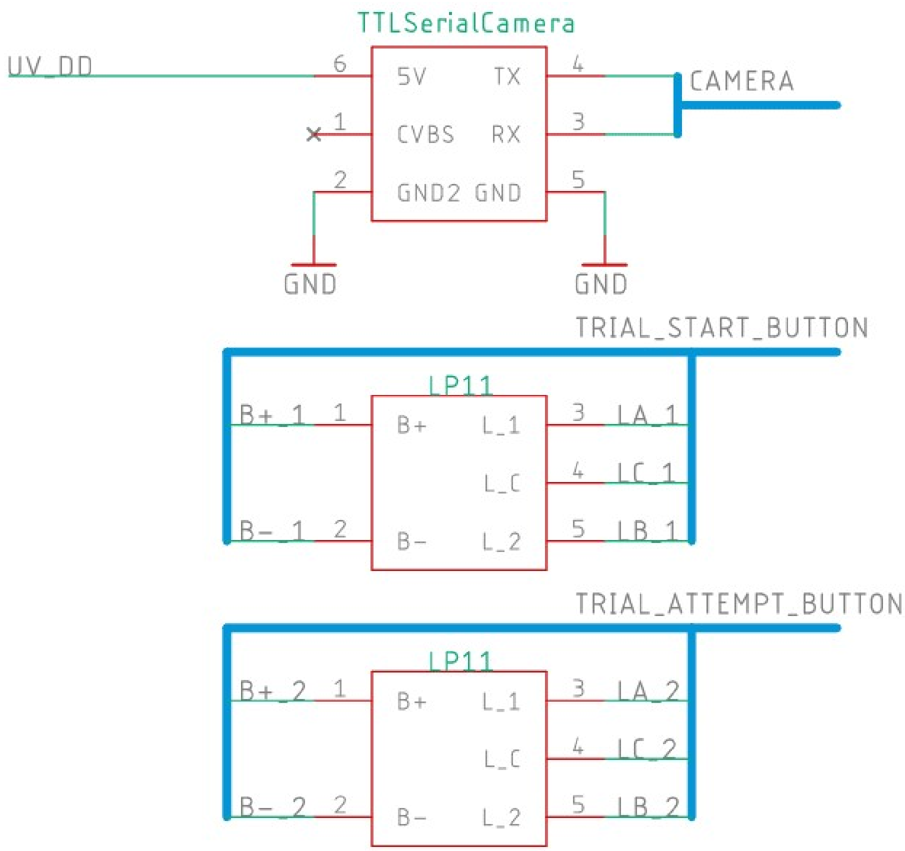

Figure 5: Circuit Schematic of the Sensor Module

2.3.1 Trial Start Button

Sends an impulse to the control module that the test subject wants to begin a trial. This button has no effect outside of this single function. If this button is not pressed for 10 minutes or more after a trial has concluded, then the software will timeout and alert the researcher [6].

Table 5: Requirements and Verification Processes of the Trial Start Button

Requirements	Verification Process
- Distinguishable from the Trial Attempt Button	- Place in separate environment with the Trial Attempt Button - Reward subject if they can distinguish between the two buttons
- Sized appropriately such that the test subject can identify and press	- Place button alone with subject in separate environment and reward if button is successfully pressed

2.3.2 Trial Attempt Button

Sends an impulse during the software's testing window that the test subject has made a response to the current trial. The impulse determines whether to reward or punish the test subject depending on whether the trial was labeled as a sham on the software's end. This button has no function outside of a specified testing window [7].

Table 6: Requirements and Verification Processes of the Trial Attempt Button

Requirements	Verification Process
- Distinguishable from the Trial Start Button	- Place in separate environment with the Trial Attempt Start - Reward subject if they can distinguish between the two buttons
- Sized appropriately such that the test subject can identify and press	- Place button alone with subject in separate environment and reward if button is successfully pressed

2.3.3 Cage Camera

Simple visible spectrum camera placed inside the testing area and interfaced with USB 2.0 with the host computer. While this sensor does not have to have the highest resolution and frame rate, it must be extremely high-fidelity and not interpolate much, especially if the resolution is on the lower end. This node must be consistent amongst all else [3].

Table 7: Requirements and Verification Processes of the Cage Camera

Requirements	Verification Process
- Must have a maximum delay of 1s or less between real world event and display time	- Monitor a timer filmed by camera and system and compare to actual timing display
- Must be innocuous as to not distract the subject from testing procedure	- Place bird in environment with both buttons (offering no reward) and watching to see subject interest in buttons over camera

2.4 Peripheral Module

This module contains all outputs to the real world. Obtains input signals from SPI interface from the control module as well as 3.5mm and USB 2.0 interface from the host computer. This module is mainly responsible for reward and punishment of the specimen depending on trial results as well as alerting the researcher if the specimen has failed to start he trial within a specified lockout time via audio cue.

2.4.1 Food Dispenser

The food dispenser is made by mounting a 40mm fan over a cut piece of plastic. By sending a current far below the normal operation regime [5]. We can control end amount of seed we deliver to the specimen with a carefully constructed control system.

Table 8: Requirements and Verification Processes of the Food Dispenser

Requirements	Verification Process
- Delivers expected food mass set within 5% error upon trial success	- Simulate a successful trial, weight the output of the dispenser
- Test subject must not be able to tamper with the dispenser and obtain food at any time	- Reward subject in separate environment and note through camera is subject is able to tamper with the dispenser to obtain additional reward.
- Delivers absolutely no reward under any other outcome other than a success	- Simulate a failure and both inaction clauses and note if any food is dispensed

2.4.2 Cage Speakers

The cage speakers must be high-fidelity and accurately replay the corresponding pitch and timbre of the given audio accurately and swiftly. The expected audio is given in 3s bursts with the average trialling session equating to a total of an hours worth of time. The speaker must be able to accurately reproduce the given audio with zero breakup or distortion. A spectrogram of the speakers must ensure that a transition from low to high frequency signals is met with zero breakup or distortion as well. Since these is the crux of the entire experiment, this device is given the least tolerance for variation and error. This device will be mounted inside of the cage and connected to the host computer via a grounded 3.5mm jack [10]. We choose to use a grounded 3.5mm jack as a USB 2.0 interface is poorly grounded and contains background noise detrimental to the experiment.

Table 9: Requirements and Verification Processes of the Cage Speakers

Requirements	Verification Process
- Accurately reproduce hi-fidelity audio with less than 10% difference between given and played spectrograms when noise is filtered	- Crate spectrgram of audio sent through 3.5mm line out - Record playback audio from speakers and compare to previous spectrogram and ensure that it is accurate enough
- Must have a standby noise of less than 10dB	- Set speakers to desired volume used in trials - Measure the standby playback of the speakers. Ensure standby is less than 10dB after filtering noise and accounting for recording errors
- Speakers must have a relatively uniform amplification in the frequency band of 100Hz to 20kHz	- Using a function generator, sweep the input signal from 100Hz to 20kHz and record the output waveform from the speaker - Compare the resulting Fourier Transforms of the recorded signals and ensure that the spectral peaks all have relatively the same height after accounting for recording microphone bias.

2.4.3 Cage Lights

The cage light is a simple LED strip with all individual electrical components connected in parallel. We choose a LED strip with each of the individual LEDs covered with a yellow phosphor in order to create a soft white light instead of a standard blue white created with a white phosphor. The total current draw of this device is estimated at an average of 200mA 5% at 3.3V when operational. Power is supplied by the control module via microcontroller as we wish to either toggle this device based on trial results and subject performance.

Table 10: Requirements and Verification Processes of the Cage Lights

Requirements	Verification Process
- Brightness of the light must not affect the subjects ability to correctly perform the test	- Progressively increase resistance of series resistor stopping when subject first displays signs of discomfort or limit imposed by sponsor as to not harm subject

2.4.4 Computer Speakers

Standard computer speakers that use a USB input. This module does not have any research impact and as such has little to no requirements

Table 11: Requirements and Verification Processes of the Computer Speakers

Requirements	Verification Process
- Audible from a 10m distance	- Play a preset audio file and ensure that it is audible from a distance of 10m

2.5 Tolerance Analysis

While the ideal speaker would replicate all harmonics and fundamentals with equal intensity, defects in production or structure exist alongside of the acoustical properties of the testing environment prohibit such an event from necessarily happening. Below in Figure 6 contains median audio grams for 48 species of birds based on classification. Though they possess a hearing range from 100Hz to 20kHz, most birds are the most sensitive to sounds within the frequency range of 1kHz to 5kHz with their most sensitive hearing at 2kHz to 3kHz, allowing a sound as soft as -15dB to be heard.

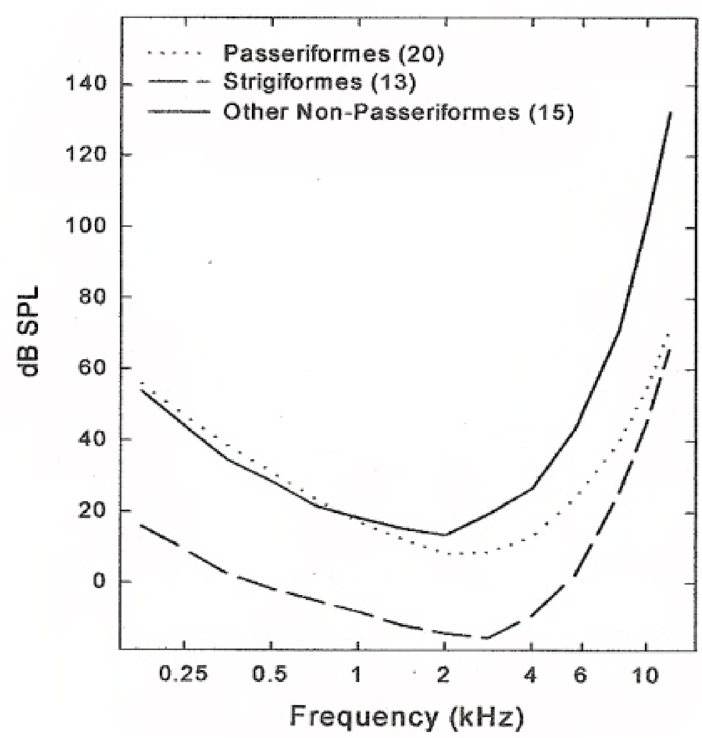

Figure 6: Median Audiogram of 48 Avian Species Sorted by Classification

The testing period's audio cues are primarily bird calls from various quiet to moderately loud birds. From our sponsor, a bird call is typically normalized 48db for their trials, though they did not deliver a relative range of loudness. If we assume that all frequencies that require an audio pressure of 60dB or greater can be ignored, then we can reduce the harmonic range that needs to be tested greatly in Passeriformes (Perching Birds), our main subject interest.

The frequency range for these perching birds could be effectively reduced to 0.1kHz to 10kHz from 0.1kHz to 20kHz, effectively halving the frequency range that needs to be sampled. To reduce the potential amount of noise, a low pass filter could be used in order to potentially remove unfaithful upper frequency harmonics that are more difficult to replicate.

In order to maximize the tolerance of our devices, we decide to set the cutoff frequency of the filter to 9kHz. We choose to use a second-order low pass filter in order to preserve phase information for signal reconstruction, however, due to this, we have a significant overshoot at the cutoff-frequency. Yet, if we look at the avian audiogram, a 9kHz sound would need around 60dB of pressure in order to reach the threshold on hearing. The overshoot is not instantaneous and would also decrease the sound pressure needed for frequencies between the range of 6kHz to 9kHz as well and, thus, we are not creating a dual peak should we sum these two curves but instead smoothing the upper frequency range of avian hearing.

To meet these requirements, we choose to create a filter with a damping factor of $\zeta = 0.10$ and a natural frequency of $f_n = 9$kHz. Below in Figure 7. We understand that components are not produced with exact specification, but meet a certain tolerance. With this, due to the inverse relationship between the cutoff frequency and resistance or capacitance, a 10% tolerance equates to roughly an 11% shift in our cutoff frequency

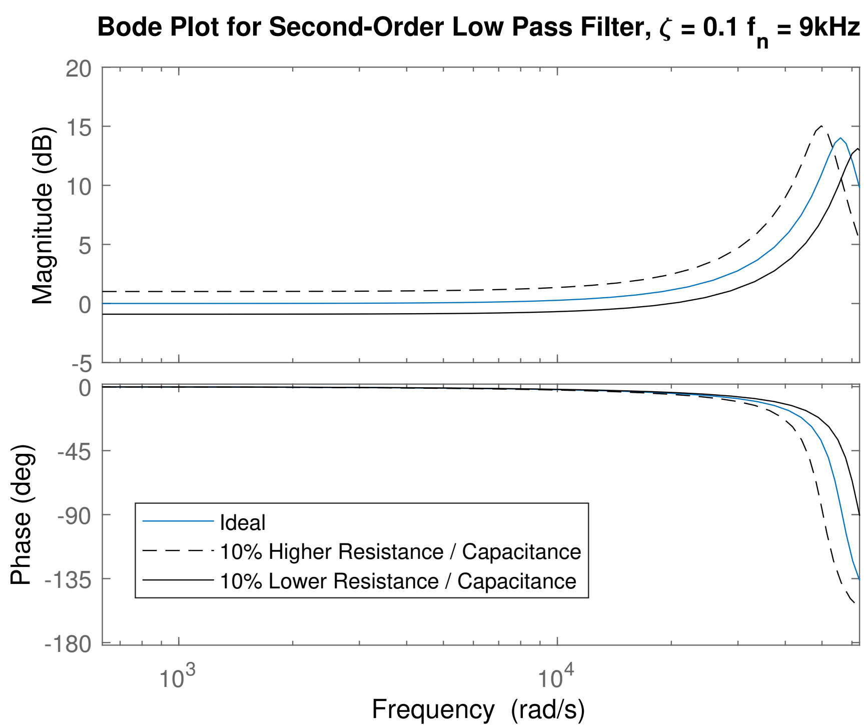

Figure 7: Bode Plot for Second-Order Low Pass Filter

A 10% tolerance translates to a shift in the cutoff frequency which will ultimately change the peak of the amplitude response and the median of the phase response. The phase component shift is acceptable. We shift the phase of relevant upper frequency components by -45 degrees if we use the maximum resistance / capacitance. If the resistance / capacitance is 10% lower than ideal, then the phase shift becomes more negligible at the cost of increased upper frequency noise as the harmonics are not as quickly suppressed.

We are comfortable stating that a 10% tolerance on the components for this portion of our device is within acceptable operating range.

3 Software Specifications

The software component of our device will be programmed to run on a Windows 10, 64 bit, platform using Python 3.5.4 as our main coding language as specified by our sponsor. The minimum timing resolution specified by the researcher for user input is said to be 100ms. Thus, we choose to run the core program at a rate of one cycle per 10ms. Higher resolution timing is not easily achievable as the timing thread in Window's software is not set up to support such a resolution.

3.1 User Interface

The application, when initiated, will first load a Graphical User Interface that must be easily accessible and intuitive for a user without computing or programming experience. From this interface, the researcher must be able to perform the following:

- Upload background and discriminatory audio files in 44kHz .wav format
- Edit trial specifications from a previous session loaded by a .ini settings file
- View generated trial statistics including sham percentage and latency timings
- Change output CSV / XML file as well as output parameters
- Generated trials to be pickled and saved for later

These are the requirements to form the minimum viable product. Other details can be added by request of the researcher assuming time and viability permit.

3.2 Trial Program

The trialing program is the crux of the software component. Thus, the design objective is that this portion of the software runs flawlessly and only faults or bugs in the most extreme circumstances or never.

A flowchart of the trialing portion of the software can be found in Figure 8. Rectangles represent sequential actions that move towards the next without any condition. Diamonds represent conditional branches based on a Boolean answer. Circles represented pauses until a certain action is performed by the user or program specified.

Trials are generated pseudo-randomly. The researcher will give a minimum number of shams such that if a set of trials does not meet the quota, we must regenerate the trials. In addition to this, the researcher will input a desired "Sham Weight" that represents the probability of a sham over the entire trial set. A minimum and maximum latency must also be given to the program in order to effectively generate trials. The latency is present to discourage the bird from repetitively pressing the trial start and attempt buttons while occasionally receiving a reward without properly performing the test.

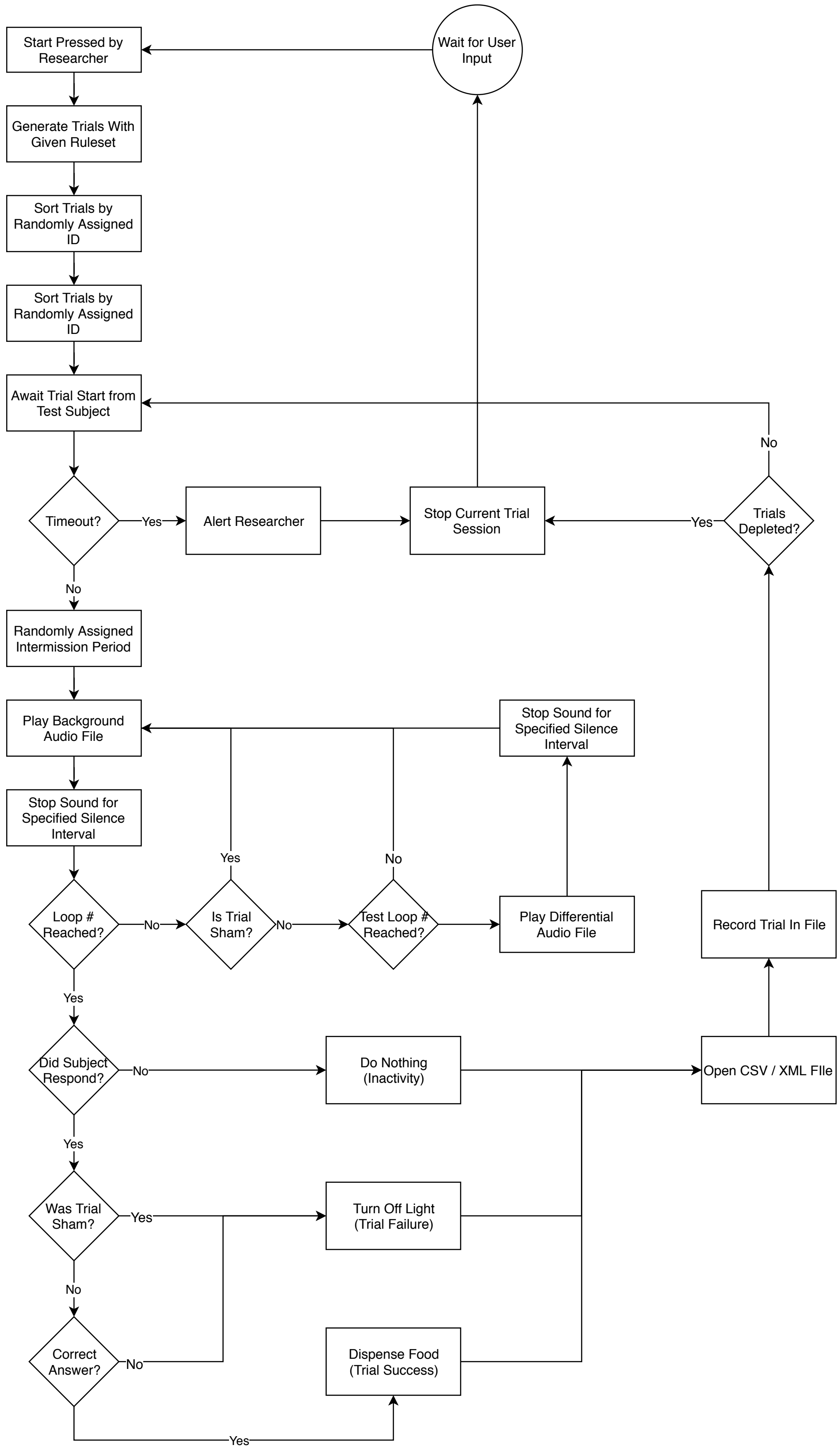

Figure 8: Flowchart of Trialing Program

3.3 Data Storage

Due to sponsor requirements, we will need to store both profiles for an individual test subject - past performance and associated metrics - as well as sets of trials to used for future research. For both of these data storage requirements, a relevant class will be created and pickled into a dump file. We will load these files later if the researcher so chooses to use these new trials

User settings such as sham weight and minimum shams would be stored elsewhere in a settings file along with other adjustable parameters.

3.4 Trial Tolerance Adjustment

In order to combat potential redundant trial generation in the case of a relatively high sham requirement with a relatively low sham weight, we will guarantee generation of the minimum number of sham trials modifying the sham weight of the remaining trials so that the resulting distribution is relatively normal. In Figure 9 we notice the two histogram outlines.

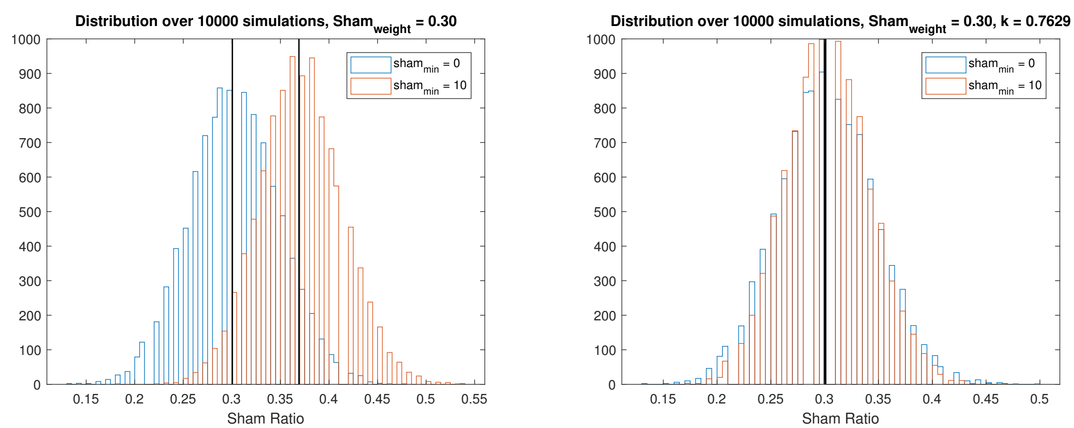

Figure 9: Distribution of trial sham rations for $n_{trials} = 100$, $sham_{weight} = 0.3$ and $sham_{min} = 10$ without error correction (left) and with error correction (right)

Without correction, though normal, the means of the generated curves do not line up. We can reduce the probability of the sham generation post guarantee sham with the following formula:

$$p_{new} = p_{old} - Sham_{min} * \frac{k}{100}$$

Since our minimum shams and number of trials are held constant by the researcher's specifications, we can generate a lookup table to adjust the probability of shams with different values of k. We would linearly interpolate between two values if a sham weight falls between two lookup values.

Using a pseudo-random process, we constructed a script that simulated trial generation with various sham weights and sham minimums. We used an estimation with a total of 20 attempts at finding the best fit for a particular correction factor given the variables, before performing the same test another 50 times and then plotting the median of the resulting data set. Below, in Figure 10 are five individual curves for various combinations of variables.

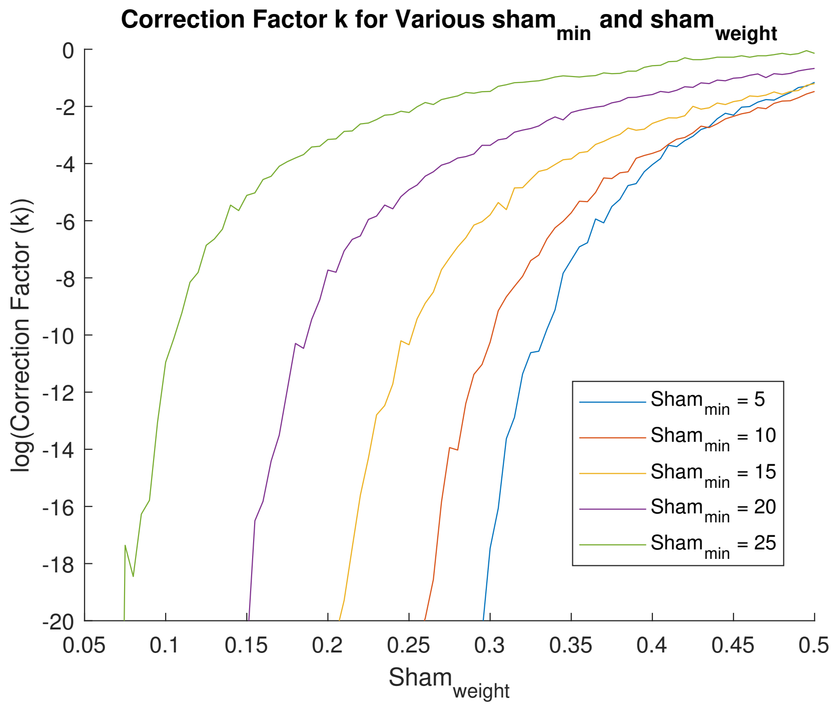

Figure 10: Median Values of log(k) with Various Variable Combinations

As seen, the log(k) can be linearized smoothly between two close points. Since we created this graph with a resolution of $\Delta S_{weight} = 0.01$, we are confident in our ability to interpolate intermediate values given our data set. In all of this, we still assume the researcher will not enter contradictory values that imply a negative probability in order to correct. Thus, in the software, we limit the ratio of the minimum number of shams with respect to the entire trial set to always be below the sham weight of said trial set.

4 Cost and Schedule

4.1 Costs Analysis and Breakdown

The total cost of the project would be $13,925.91 which is the sum of the Labor costs and Parts costs.

4.1.1 Cost of Labor

We assume that the labor costs of the design would be $45 USD per partner and 10 hours a week. We will consider 63% of the semesters weeks into this calculation. Thus, labor costs for each partner in this project would be encompassed with the following calculation:

$$\$45/\text{Hour} \cdot 3 \text{ partners} \cdot 10 \text{ hours/week} \cdot 0.63 \cdot 16 \text{ weeks} = \$13,608$$

4.1.2 Cost of Parts and Materials

Below is a table of the cost of our components

Table 12: Requirements and Verification Processes of the Computer Speakers

Part	Cost	Cumulative Total
Female USB Input	$5.79	$5.79
LM1117 Voltage Regulator	$1.10	$6.89
ATMEGA16U2A2 Microprocessor	$2.52	$9.41
Blue LP11 Series Switch	$12.26	$21.67
Red LP11 Series Switch	$10.97	$32.64
TTL Serial Camera 1528-1401-ND	$39.95	$72.59
OrionFan OD4020 Series	$4.74	$77.33
3.5mm Jack Breakout	$3.95	$81.28
Vifa Compact HiFi Bluetooth Speaker	$229.00	$310.28
Passive Components	$7.63	$317.91

4.2 Schedule

The schedule is designed with a two-week debugging buffer period, which is also why nearing the end of the schedule various roles are the exact same. These roles will be assigned accordingly based or the problems that require debugging attention. The schedule prioritizes a design process, thus slating three trials to interact with the bird to account for any unforeseen issues. As the designers of the project, our paramount goal is to create a system that works for the researchers that demand the product

Table 13: Design Schedule

Week #	Group Objectives	Individual Responsibilities
Week 1 Feb 18-24	- Finish design document - Finish circuit schematics - Place initial part orders	- Kevin - Design Document, Part Orders - Michael - Design Document, Software - Maria - Design Document, Eagle
Week 2 Feb 25 - Mar 3	- Power test the parts - Check peripheral modules - Lay out general structure of software	- Kevin - Assemble parts on breadboard to test, Sensor verification - Michael - Trial Program Software assignments - Maria - Voltage tests for power module
Week 3 Mar 4 - 10	- Prepare for hardware run through with bird	- Kevin - Software GUI, Prepare for round 1 of system feedback - Michael - Data storage assignments - Maria - Finalize first round of PCB Eagle design
Week 4 Mar 11 - 17	- Submit first PCB design - First integrated run through with bird (Hardware)	- Kevin - Review PCB initial submission, GUI handoff to Mike - Michael - Hardware bird trial verification tests - Maria - Software inspection for hardware integration
Week 5 Mar 18 - 24	- Iterate upon hardware design - Prepare for hardware / software integration	- Kevin - Make adjustments based on hardware issues from bird test - Michael - Link GUI with trial program to prepare for hardware integration - Maria - PCB verification tests
Week 6 Mar 25 - 31	- Integrate hardware / software components - Perform second bird test with the system	- Kevin - Link hardware with software - Michael - Link hardware with software - Maria - Link hardware with software
Week 7 Apr 1 - 7	- Finalize PCB design for submission - Debug any additional problem with second bird test	- Kevin - Debug integration - Michael - Debug integration - Maria - Finalize PCB design
Week 8 Apr 8 - 14	- Final integration with software / hardware - Final bird test - Prepare for final demos - Leeway week to debug any final issues	- Kevin - Anything the project requires - Michael - Anything the project requires - Maria - Anything the project requires
Weeks 9, 10 and 11 are all demo / presentation weeks		

5 Ethics and Safety

Following the ACM code of ethics section 2.3: “Know and respect existing rules pertaining to professional work”, we need to learn existing guidelines and construct our project around and according to them [12]. Before any work with animals, it is mandatory to submit IACUC protocols and they adhere to nationwide rules for animal care and research. Our project fulfils all these requirements and has been approved already [11].

The birds are rewarded with specific amount of food for successful trials, and, we shut off the cage light in an ethical way to not harm the subject either physically or mentally. The light shut off is transient and only serves as an indicator that the subject has done something wrong, and as such, does not frighten or stress the subject. The timeout function ensures that the subject does not necessarily have to undergo testing if it does wish to do so in accordance with ACM code of ethics section 1.2: “Avoid harm”, we are treating the subjects in such a manner that we avoid any potential harm that may befall them otherwise [12].

In the design of our project, we will follow the guidelines set by the IEEE code of ethics section 6: To maintain and improve our technical competence and to undertake technological tasks for others only if qualified by training or experience, or after full disclosure of pertinent limitations [13]. So we will follow every guideline that the doctorate responsible of this project tell us to be sure that we give the right amount of food to the bird and how much time the bird can be with the light off. Following this code, we need to make sure that we have an user friendly interface that everyone without programming knowledge can understand. We need to make sure that our project works seamlessly and without error because the integrity of our system will also reflect the reputation of those who use it, following the IEEE code of ethics section 9.

Our design must account for certain ethical risks. All the wires must be outside the cage and hidden such that the bird will not be subject to electrical damage. If any wires are required within the cage, they must be concealed by opaque material to guarantee risk aversion. All hardware that the bird interacts with must be attached in a manner in which is irremovable by the bird. A proper speaker frequency range is paramount to ethical hardware design with the bird. Given that bird songs typically range from 1,000Hz to 8,000Hz, our design will adhere to that range to avoid any stress on the bird [15].

References

[1] R. O. Davis *Computer Methods and Programs in Biomedicine. Digital signal processing in studies of animal acoustical communication, including human speech.* Volume 23, Issue 3, December 1986, Pages 171-196

[2] Texas Instruments, *LM1117 800 mA Low-Dropout Linear Regulator,* LM1117 datasheet, February 2000. [Revised January 2016].

[3] Adafruit Learning Industries, *TTL Serial Camera,* TTL Serial Camera with NTSC Video Output tutorial. October, 12, 2018.

[4] Microchip Technologies, *8 bit AVR Microcontroller with 16K bytes of ISP Flash and USB Controller,* ATMega16U2 Datasheet, 2010.

[5] Orion Fans, *OD4020 Series,* OD4020 Datasheet, Publishing Date Unknown.

[6] eswitch, *LP11 Series Pushbutton Switch,* LP11EE1NCSRGB Datasheet. September 5, 2018.

[7] eswitch, *LP11 Series Pushbutton Switch,* LP11EE1NASRGB Datasheet. September 5, 2018.

[8] Broadcom, *4mm Oval Precision Optical Performance LED,* HMLPLG71XD0DD Datasheet. March 30, 2012.

[9] VCC *4303F Series Solid State LED T1 (3mm),* 4303F5 Datasheet. Publishing Date Unknown.

[10] Sparkfun, *4UCON Technology Inc,* Audio 3.5mm Datasheet. August 2004. [Revised in May 2005].

[11] *Illinois Institutional Animal Care and Use Committee (IACUC). Regulatory Compliance & Safety*, 2016. Available at: http://research.illinois.edu/regulatory-compliance-safety/research-integrity-and-ethics

[12] *ACM.org ACM Code of Ethics and Professional Conduct*, 1992. [Online]. Available at: https://www.acm.org/code-of-ethics

[13] IEEE.org *IEEE Code of Ethics*, 2016. [Online]. Available at https://www.ieee.org/about/corporate/governance/p7-8.html. (Accessed February 4, 2019).

[14] Ritchison, G. (n.d.). Retrieved February 20, 2019, from http://people.eku.edu/ritchisong/birdbrain2.html

[15] *All About Birds*, Do Bird Songs Have Frequencies Higher Than Humans Can Hear? https://www.allaboutbirds.org/do-bird-songs-have-frequencies-higher-than-humans-can-hear/