



Design Document

Le Bleu Caterpillar

Team #9A: Ivan A. Lopez Martinez, Camryn Quam, Jennifer Wolfe, Zhong Zhang

Date: November 13, 2022

Table of Contents

Project Description	4
System Design	5
Interface Table	5
Hierarchical Design	6
Team Composition	6
Sub-System Designs	6
User Interface Sub-System	6
Audio Sub-System	9
Motion Sub-System	10
Lighting Sub-System	12
Power Sub-System	13
Software/Control Sub-System	15
Constraints, Alternatives, and Trade-Offs	15
Electronic Design	16
Electrical Simulation	18
Audio	18
Lights	19
Power	20
Mechanical Design	21
Mechanical Simulation	22
Software Design	24
Software Simulation	25
Schedule	29
Integration	30
Final Build	31
Conclusion	35

Revision Record

<u>Date</u>	<u>Author</u>	<u>Comments</u>
September 15, 2022	Team 9	Document Created (Project description and hierarchical design; schedule)
September 21, 2022	Ivan	Audio and Power subsystems
September 21, 2022	Camryn	UI and motion sub-systems
September 22, 2022	Zhong	Software subsystem/simulation
September 23, 2022	Team 9	Document Updated (Subsystems, Subsystems design)
September 24, 2022	Jennifer, Ivan	Updated block diagram, Interface table; lighting/UI, updated audio sub-system diagram, added constraints/alternatives/trade-off section
September 25, 2022	Camryn	Updated UI and motion sub-system sections
September 25, 2022	Team 9	Document completed and turned in (integration, conclusion, constraints)
October 13, 2022	Team 9	Added simulations for subsystems, updated schedule
October 14, 2022	Team 9	Added schematics for subsystems
November 10, 2022	Team 9	Added more information on Integration
November 13, 2022	Team 9	Updated final assembly and testing process, conclusion

Project Description

Le Bleu Caterpillar is the creature who gives Alice advice in Wonderland while smoking from a hookah on top of a mushroom. Our representation of Le Bleu includes a moving arm connected to the hookah mouthpiece, LED eyes that turn red when he smokes, audio, and buttons for user interaction. The user can independently activate the lights, audio, and motion, as well as activate all three simultaneously for demonstration/test purposes. Additionally, the robot can receive and play audio wirelessly from a Director computer.

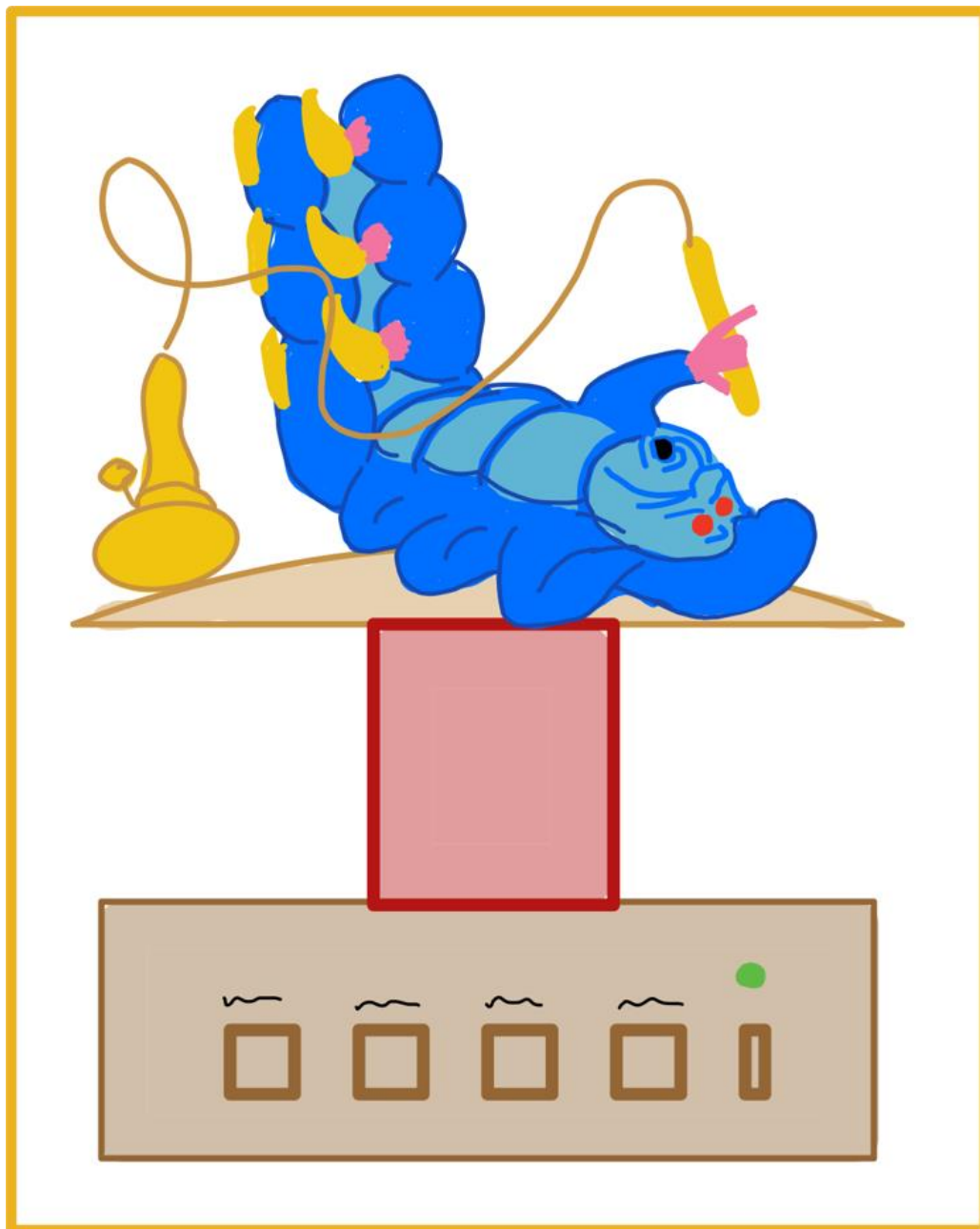


Figure 1. Initial Sketch of Le Bleu Caterpillar smoking on top of his mushroom

System Design

The system uses 120V AC power which is converted to 5V DC for use in subsystems. Buttons, a switch, and a volume knob function as inputs from the user. Additionally, the system receives audio from the Director computer's signal. Outputs of the system includes a speaker, movement of the caterpillar's arm produced by a motor, and lights that turn Le Bleu Caterpillar's eyes from white to red. The system consists of six subsystems: user interface, audio, motion, lighting, power, and software/control. The audio subsystem includes an analog circuit with a 10K ohm potentiometer to adjust the volume. Figure 2 shows an overall system block diagram, Figure 3 shows an interface table for the subsystems, and Figure 4 breaks down the hierarchical design.

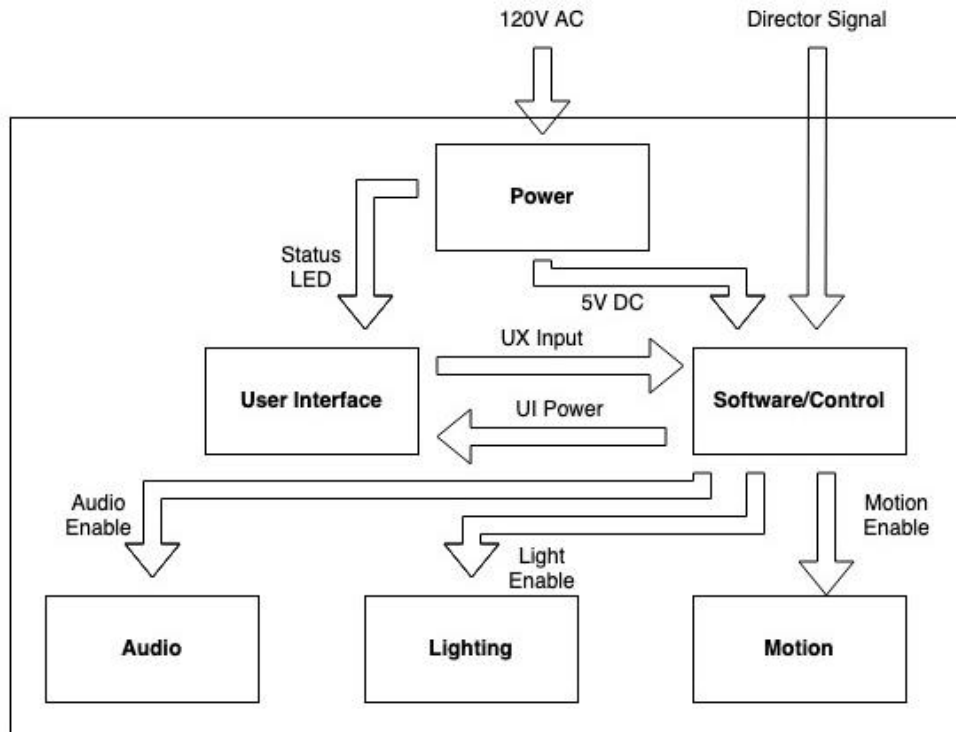


Figure 2. Overview of System Block Diagram for Le Bleu Caterpillar

Interface Table

Interface	Source	Destination	Description
120V AC	System input	Power	120 V AC input power
Director Signal		Software/Control	Signal from Director
5V DC	Power	Software/Control	5V DC powers Pi Zero W
Status LED		User Interface	LED output
UX input	User Interface	Software/Control	Signal from User to Software
Light Enable	Software/Control	Lighting	Signal to control and power lighting; 3.3 V and GPIO signal
Audio Enable		Audio	PWM signal to control and power audio; 5V from Pi (Pin)
Motion Enable		Motion	PWM signal to control and power motion; 5V from buck converter
UI Power		User Interface	Power User Interface; 3.3 V

Figure 3. Sub-systems inputs, outputs, and interfacing

Hierarchical Design

The hierarchical design simply shows the system breakdown into subsystems and components and does not indicate subsystem connections. For this, view the block diagram and interface table in Figures 2 and 3.

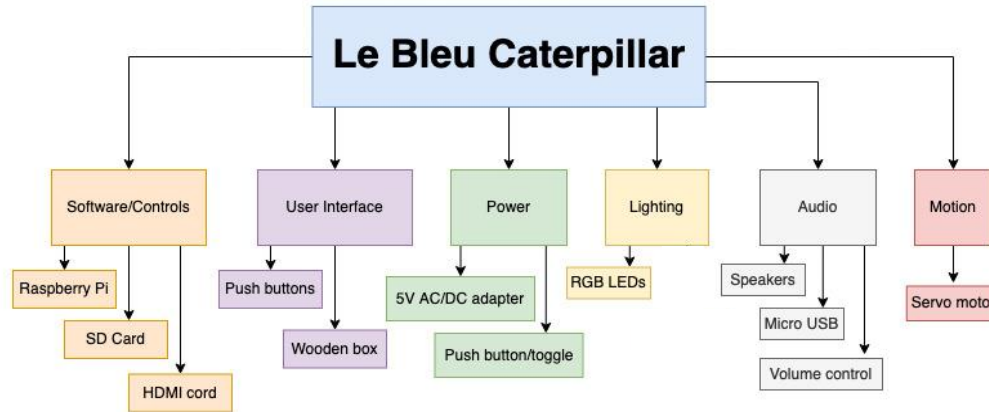


Figure 4. Hierarchical Design of Le Bleu Caterpillar

Team Composition

Team Lead + Lighting Subsystem	Jennifer Wolfe
Electrical Lead (Audio/Power Subsystems)	Ivan A. Lopez Martinez
Mechanical Lead (Motion/User Interface Subsystems)	Camryn Quam
Software Lead (Software/Controls Subsystem)	Zhong Zhang

Sub-System Designs

User Interface Sub-System

The user interface consists of four input pushbuttons, one input switch, a volume control knob, and one output LED. The interface takes in 3.3 V of power, which is controlled by an on/off switch. Once powered, users have four options of operation: lights, audio, or motion individually; or a test mode. The power switch includes an LED to visually inform the user that the system is powered on. A volume knob is included to allow the user to control the sound level. The user interface is accessible from the front panel of the box. See below for the subsystem block diagram and interface table.

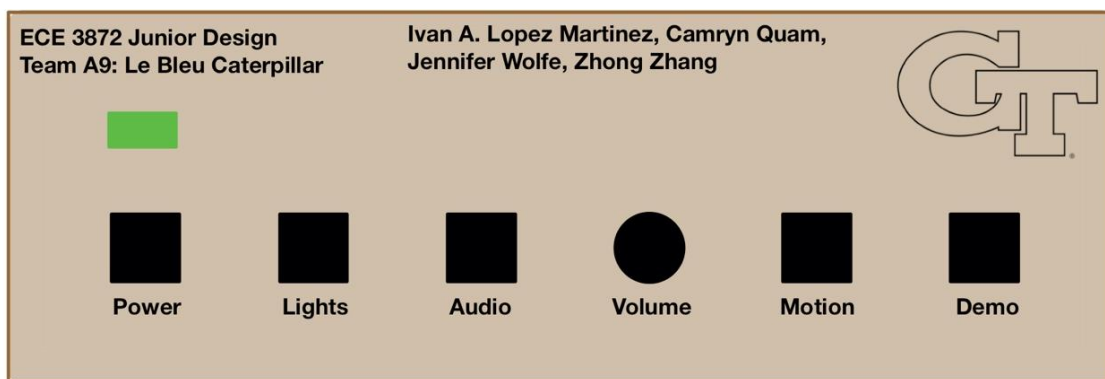


Figure 5. Initial layout of User Interface panel

User Interface Sub-system

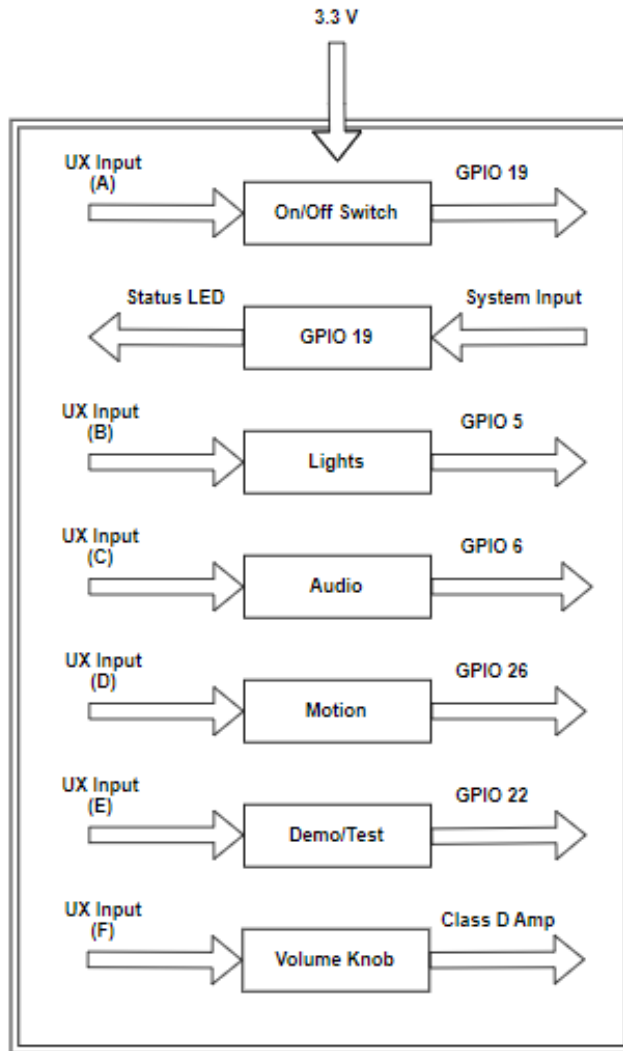


Figure 6. Block diagram of the User Interface Sub-system

Interface	Source	Destination	Description
3.3 V	Software/Control	User Interface	User interface is powered by 3.3 V of power from software/control
UX Input (A)	User Interface	On/Off Switch	User input command for power switch
GPIO 19	On/Off Switch	Software/Control	LED light enabled when system ON; LED light disabled when system OFF
System Input	Software/Control	GPIO 19	Value read from GPIO 19 will indicate whether system is in on or off state with LED
UX Input (B)	User Interface	Lights Button	User input command for lights
GPIO 5	Lights Button	Software/Control	Lights display will be performed
UX Input (C)	User Interface	Audio Button	User input command for audio

GPIO 6	Audio Button	Software/Control	Audio will be played through speaker
UX Input (D)	User Interface	Motion Button	User input command for motion
GPIO 26	Motion Button	Software/Control	Motion actions will be performed
UX Input (E)	User Interface	Demo/Test Button	User input command for demo/test
GPIO 22	Demo/Test Button	Software/Control	Demo/test actions will be performed
UX Input (F)	User Interface	Volume Knob	User input command for volume control
Class D Amp	Volume Knob	GPIO 13	Volume levels will vary depending on the value read from the potentiometer

Figure 7. User interface sub-system interface table

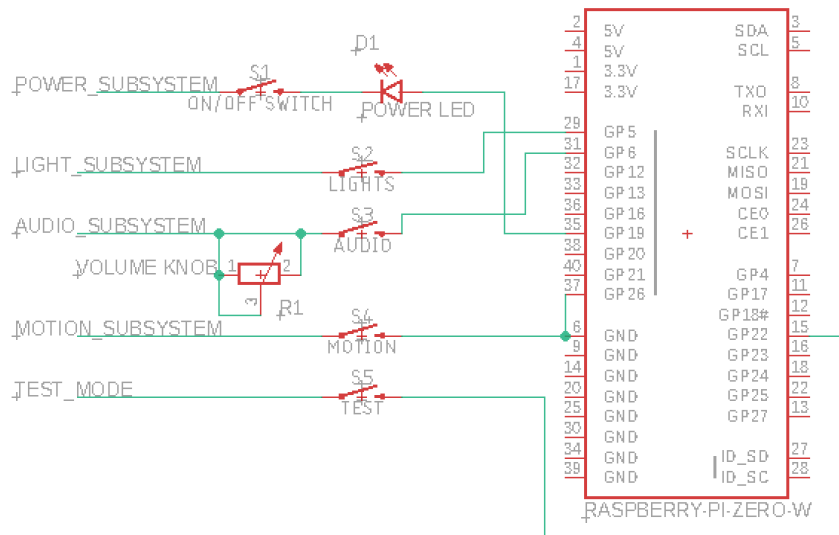


Figure 8. Schematic of user interface connections



Figure 9. Finished user interface panel

Audio Sub-System

The audio sub-system is the analog circuit of the robot. This system is comprised of a 10kΩ potentiometer, the BOB 11094 sound amp chip, controller inputs from the Pi, and the speaker. The potentiometer and speaker are wired to the sound chip, which in turn receives the signal from the controller. The signal coming from the Pi is a .wav audio file containing lines of the Caterpillar character of Alice in Wonderland. The potentiometer acts as volume control as it modulates the signal coming from the Pi by adjusting the resistance through the wiper. The wiper from the potentiometer is controlled by the user. The sound chip is powered by the 5V coming from the Pi.

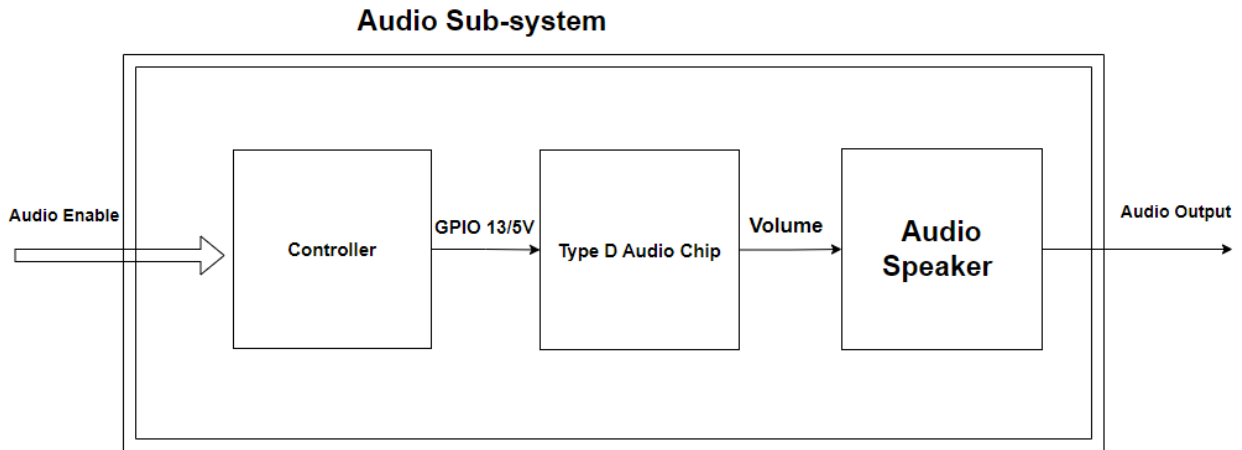


Figure 10. Audio sub-system with interfaces between controller and audio chip

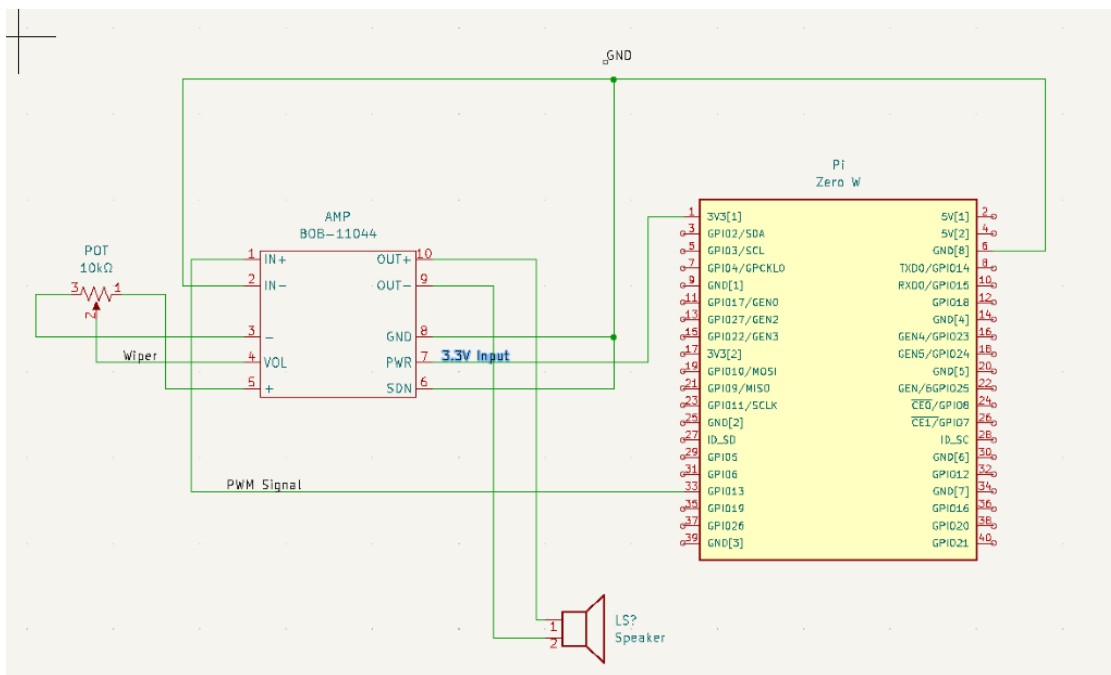


Figure 11. Audio subsystem Schematic

Interface	Source	Destination	Description
Audio Enable	UX Input (C)	Software/Control	User presses audio button, enables audio system
GPIO 13/ 5V	Software/Control	Type D Audio Chip	Controller powers audio chip and sends audio signal to sound chip
Volume	Type D Audio Chip	Audio Speaker	Audio chip attenuates volume of audio signal through POT input
Audio Output	Audio Speaker	System Output	Audio file plays through speaker

Figure 12. Audio sub-system interface table



Figure 13. Final audio system for integration using USB speaker

Motion Sub-System

The motion sub-system utilizes a servo motor to produce the movement of the caterpillar's arm. The servo takes in 5 V of power and is controlled through a PWM signal sent from the Raspberry Pi Zero. The servo is mounted between two segments of the caterpillar's arm and sweeps 90° to emulate smoking.

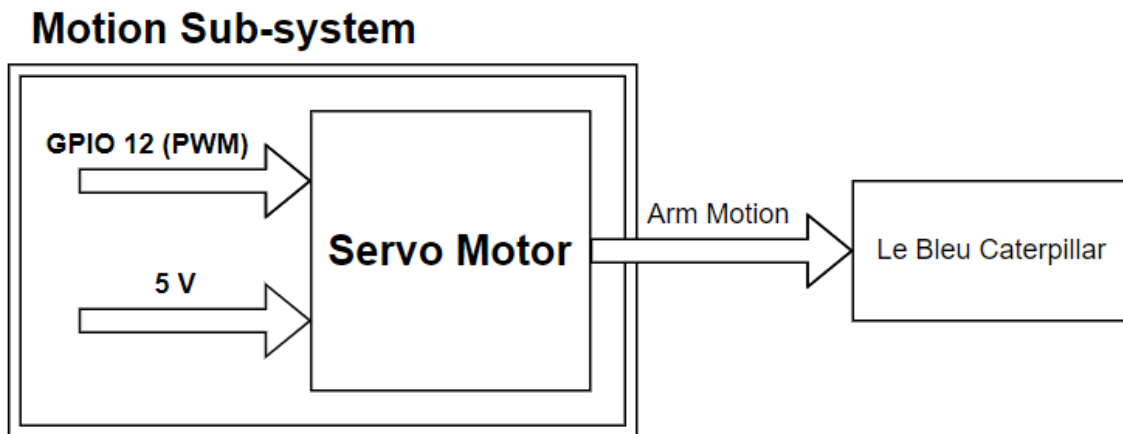


Figure 14. Block diagram of the motion sub-system

Interface	Source	Destination	Description
GPIO 12	Software/Control	Servo Motor	Servo motor position will come from software/control output
5 V	Software/Control	Servo Motor	Motor is powered with 5 V from an external power supply
Arm Motion	Servo Motor	Le Bleu Caterpillar	Motor will be mounted on Le Bleu to produce up and down arm movement

Figure 15. Motion sub-system interface table

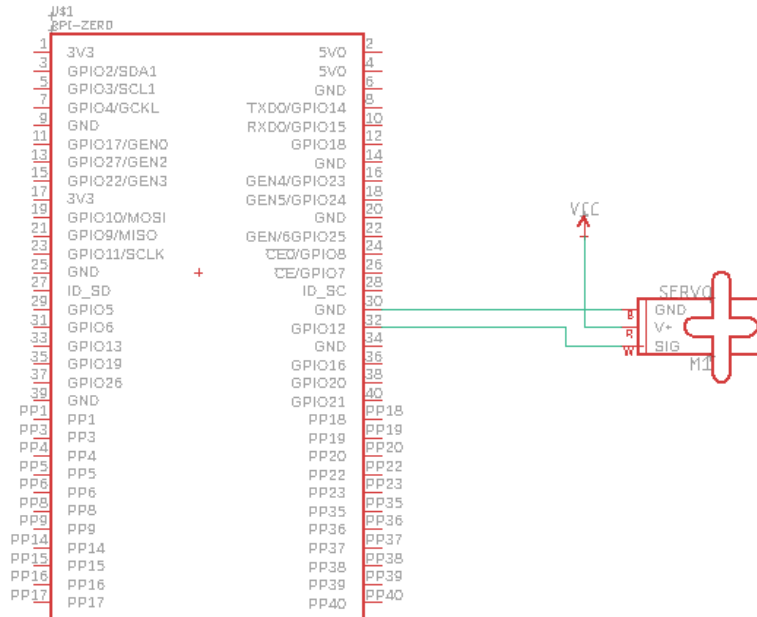


Figure 16. Schematic of motor connected to Pi

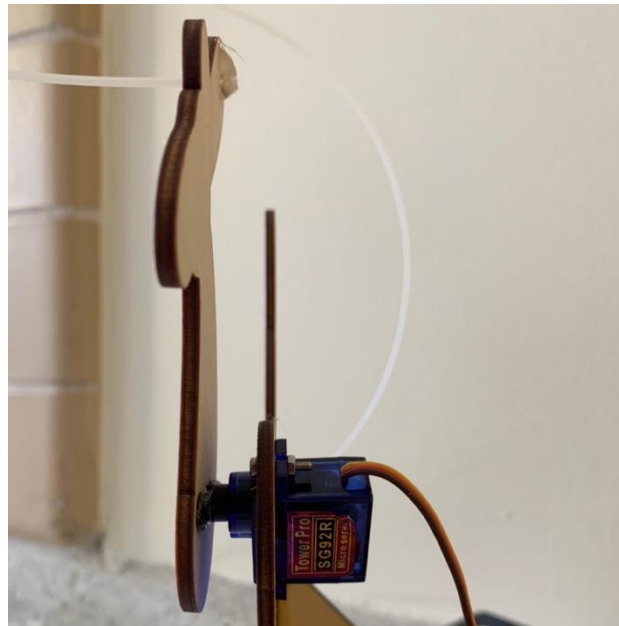


Figure 17. Servo motor attached to arm

Lighting Sub-System

The lighting sub-system receives a light enable signal from the Raspberry Pi in the Software/Control sub-system. This includes controls for the lighting and power. The Raspberry Pi can output 3.3 V or 5 V DC; however, the RGB red lead requires 1.8 to 2.2 V and 20mA, the green requires 3.0-3.4 V and the blue requires 2.9-3.3 V. After calculations with Ohm's law and the 3.3V pin, 110-ohm resistors are used to manage the voltage through the red RGB lead and to connect the green and blue leads with wires. To simplify integration, the red leads are connected to 3.3 V and the green and blue leads are connected together to GPIO 27. When a control signal is sent from the Raspberry Pi, green and blue turn on together with the red light and create a white light. When another control signal is sent again, the green and blue lights turn back off and leave the eyes red.

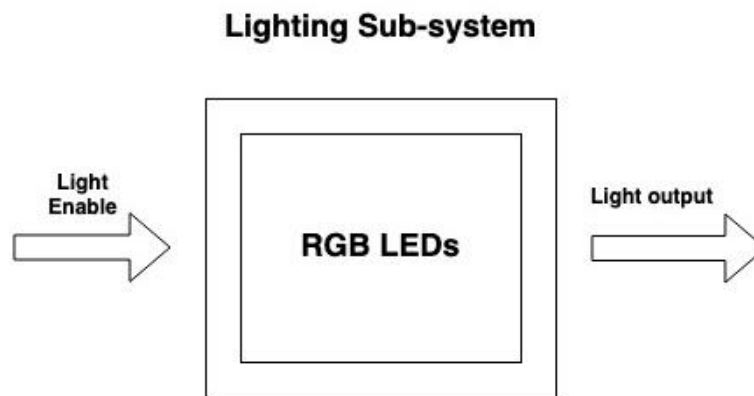


Figure 18. Block diagram of the lighting sub-system

Interface	Source	Destination	Description
Light Enable	Software/Control	RGB LEDs	Pi Zero W powers LEDs; Sends control signal
Light Output	RGB LEDs	System output	Lights change from White to Red

Figure 19. Lighting sub-system interface table

Since the lighting color change is connected to the motion of Le Bleu's arm moving towards his mouth, the controls for the lighting are connected to the motion controls in the software. Based on results from the electrical hardware simulations, a GPIO signal is used to toggle the green and blue lights to make Le Bleu's eyes change from white to red.

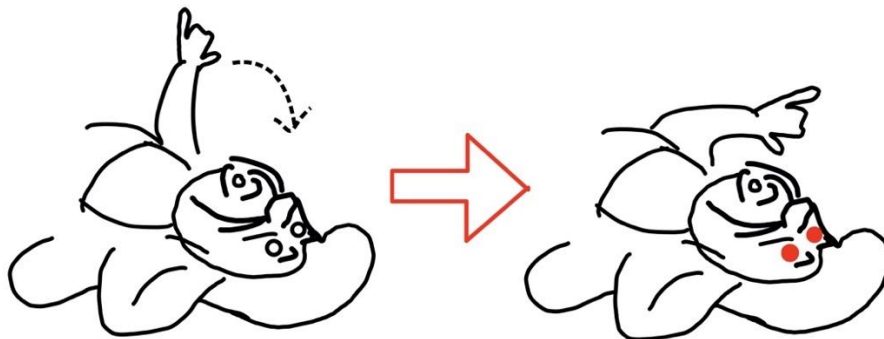


Figure 20. Concept showing the change of Le Bleu's eyes from white to red as his arm moves.

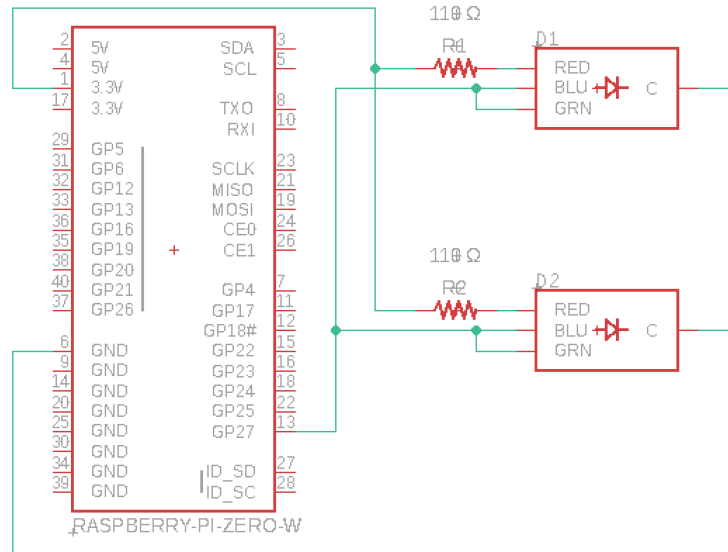


Figure 21. Schematic of RGBs connected to Pi

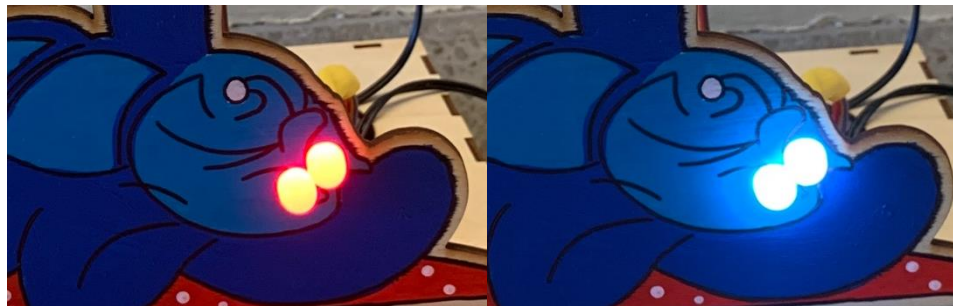


Figure 22. Red and white eyes on Le Bleu

Power Sub-System

The power sub-system receives 120 V AC from a wall outlet and directs it to a 5 V AC-to-DC converter. Initially, the 5 V output was then directed to an ON/OFF switch which redirects the voltage to the Pi controller when it receives input from the user. After testing was done, it was determined that the Pi does not output enough current to properly power the motor. A buck converter regulates the input voltage while stepping up the output current to the load. This function of the buck converter is useful given that once integration of all systems takes place, the current drawn might overwhelm the controller's capacity to power the systems.

In the current design, power comes from the buck converter and connects to the ON/OFF switch and the servo motor. When the switch is high, it will signal the controller to initiate the states of the overall system. The controller will interact with user inputs which sets a state; once a state has been determined, the controller then directs voltages to the audio and lighting systems, powering them and initiating communication between their respective interfaces.

One safety option was to connect a fuse between the output of the buck converter and the motor for risk prevention, although it was not necessary since the AC/DC converter already contains protective mechanisms built in.

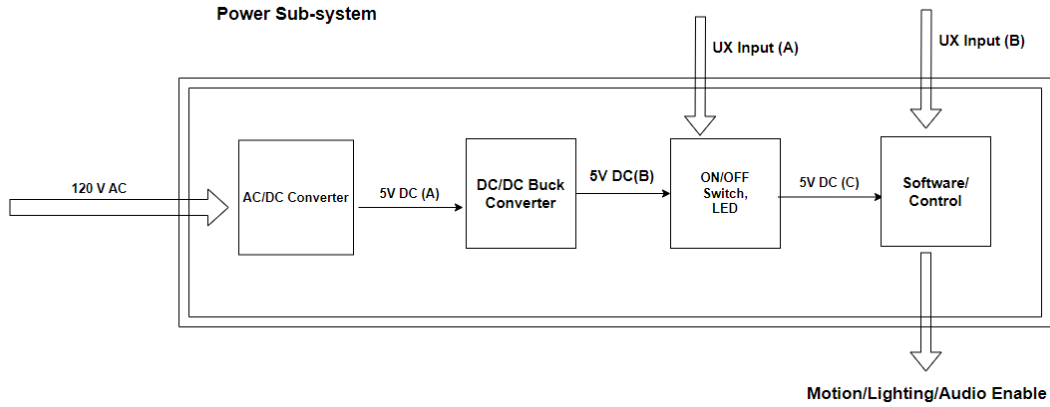


Figure 23. Power sub-system converts AC to DC, steps current up with buck converter, and powers the software controller

Interface	Source	Destination	Description
120V AC	System input	AC/DC Converter	120 V AC input power
5V DC (A)	AC/DC Converter	DC/DC Buck Converter	Buck converter regulates voltage and steps up current
5V DC (B)	DC/DC Buck Converter	ON/OFF Switch, LED	Buck regulates voltage to power ON/OFF switch and servo motor
UX input (A)	User Interface	ON/OFF Switch, LED	User activates system, LED indicates system is on
5V DC (C)	ON/OFF Switch, LED	Software/Control	5V DC powers Pi Zero W
UX Input (B)	User Interface	Software/Control	User input command
Motion/Lighting/Audio Enable	Software/Control	Audio/Motion/Lighting	Pi Zero W enables performance of sub-systems

Figure 24. Power sub-system interface table

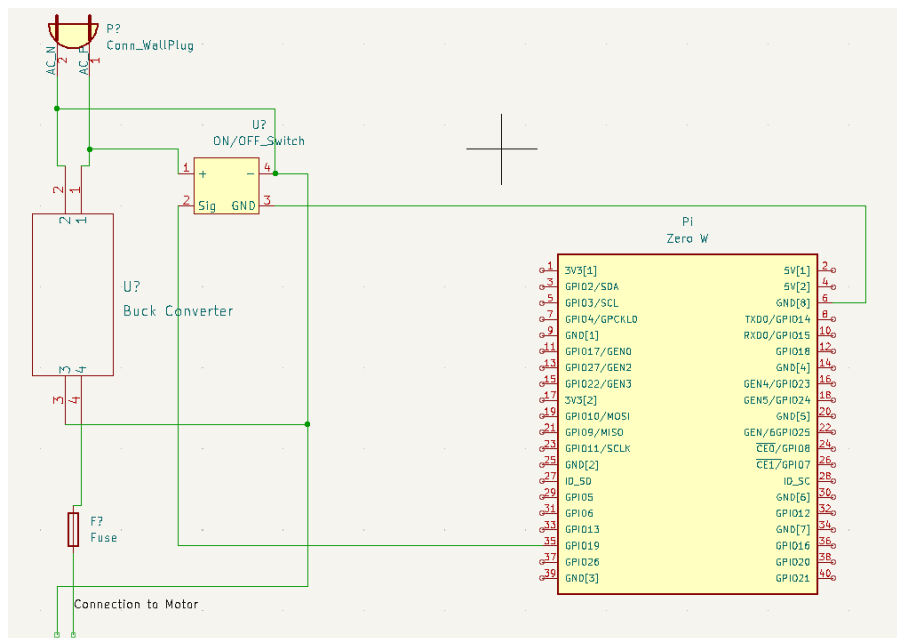


Figure 25. Power sub-system schematic including buck converter

Software/Control Sub-System

The software/control subsystem receives a 5V DC power supply from the power subsystem. This powers on the Raspberry Pi. User inputs are also received by the Raspberry Pi, and the Raspberry decides how to output the Light Enable, Audio Enable, and the Motion Enable based on user inputs. A script is given from the director, and the Raspberry Pi enables the robot to read the line from the script. Further descriptions are given in the Software Design section.

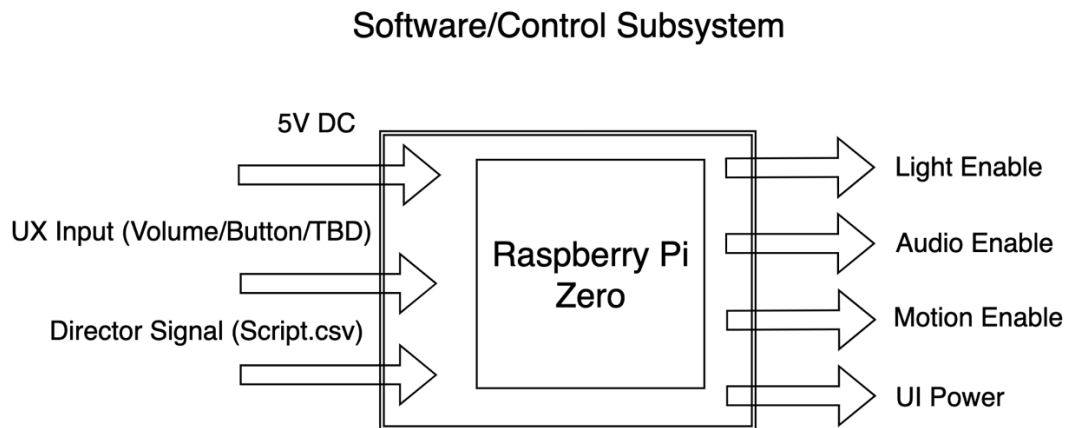


Figure 26. Control Subsystem

Constraints, Alternatives, and Trade-Offs

Controller

The Pi Zero W controller has the ability to output 3.3 V and 5 V DC which limits the selection of components needed to meet the needs of the client.

Speaker

Raspberry Pi contains a mini-USB port, whereas the original speaker has a regular USB connection. This can be remedied by using a mini-USB adapter to use with the speaker. If the mini-USB adapter is not available, then the team will design a circuit that connects the speaker to the Pi controller using its regular USB connector. Alternatively, if it is not possible to use the USB speaker, then other types of speakers with polarized cable connection can be used in a circuit that uses the 5V output of the Pi controller to power said speaker.

For volume control, an analog circuit incorporating a 10 kΩ POT, an audio amplifier, and the Pi Zero W controller routed to a PCB is used. This setup can be constrained by the capabilities of the Pi controller and the difficulty in merging both the amplifier circuit with the controller. The speaker will be powered and controlled through the mini-USB port, and the intention is to link both the speaker and the potentiometer output through a GPIO signal. Given the situation where such a setup will not work, one alternative could be to use GUI screen from an MBED kit to connect to the Pi controller in order to directly control the volume of the system through it.

After testing the volume control function using the USB speaker, it was determined that communication between the POT and the controller cannot be established this way. This revelation inspired redesigning the audio system into the current one. The POT is connected to a Class D sound chip which takes the audio signal from the controller to be attenuated by the POT's input. As the signal gets attenuated, the chip will output the signal through the speaker connected to it. The tradeoff of this setup is that the speaker is lower quality than the USB speaker. The quality of the sound is not as optimized as it could be using the USB speaker. However, the USB speaker setup did not function with the POT as intended, meaning the user would have less control of the system. With the current setup, the user can listen to and adjust the volume of the sound in the system. Other positive aspects of this new audio system are the simplicity of its design and implementation, as well as its accessibility for modifications if needed.

Power

The Pi Zero W controller is powered through a mini-USB port, which limits the options of types of AC/DC converters compatible. Additionally, the limit voltage that the Pi Zero W allows in order to work efficiently is around 5.25V DC.

The power drawn by most systems is satisfied by the power pins of the controller, but the motor draws more current than the Pi can distribute. One alternative to resolve this is to connect the motor directly to the power source of the system. For this, a Buck converter was connected by the input of the AC/DC converter, and then to the motor. This design works properly. In the case that the system is not sufficient to power all the components during the global integration of the system, then a step-up converter can be connected to the input of the AC/DC adapter in conjunction to the Buck converter and have the output of this step-up converter be directed towards the other sub-systems.

Electronic Design

There are some space constraints in the customer requirements for the product. Given this space limitation, the probability of parasitic elements affecting the product's performance is increased significantly. These parasitic elements can affect audio quality as well as motor movement. Incorporating a PCB, with resistors and capacitors that set sub-systems to operate under their target impedance is an alternative that can ameliorate the negative effects of the parasitic elements.

Motor

The movement of the Servo SG90 motor is approximately 180°, which limits the positions of Le Blue Caterpillar. Additionally, the motor weighs close to 0.32 oz, and the motor produces a torque of 2.5 kg/cm (34.7 oz/in). This information helps narrow down the type and quantity of material needed to build the arm of the caterpillar. Concerns regarding the weight of material the motor could support have resulted in the decision for the moving arm piece to be cut out of cardboard, while the rest of the robot will be cut out of wood.

Electronic Design

The electronic design of the system, Le Blue Caterpillar, is comprised of five sub-systems: power, software/control, lighting, audio, and motion. The interactions of these sub-systems are prompted by user inputs, which are a switch and pushbuttons that initiate the functions of: converting 120 V AC to 5 V DC, turning system on, lighting LED's, output audio, and moving the arm of robot.

The controller, a Raspberry Pi Zero W, receives 5 V DC from the power converter. Once the controller is active, it proceeds to send enabled signals and voltages to the lighting, audio, and motion sub-systems. The motion sub-system powers the servo motor on the caterpillar's arm. The lighting sub-system changes the color of the RGB LEDs in the eyes from white to red. The audio sub-system plays sounds through a USB volume-controlled speaker. To route power and control these subsystems a PCB has been developed for the design.

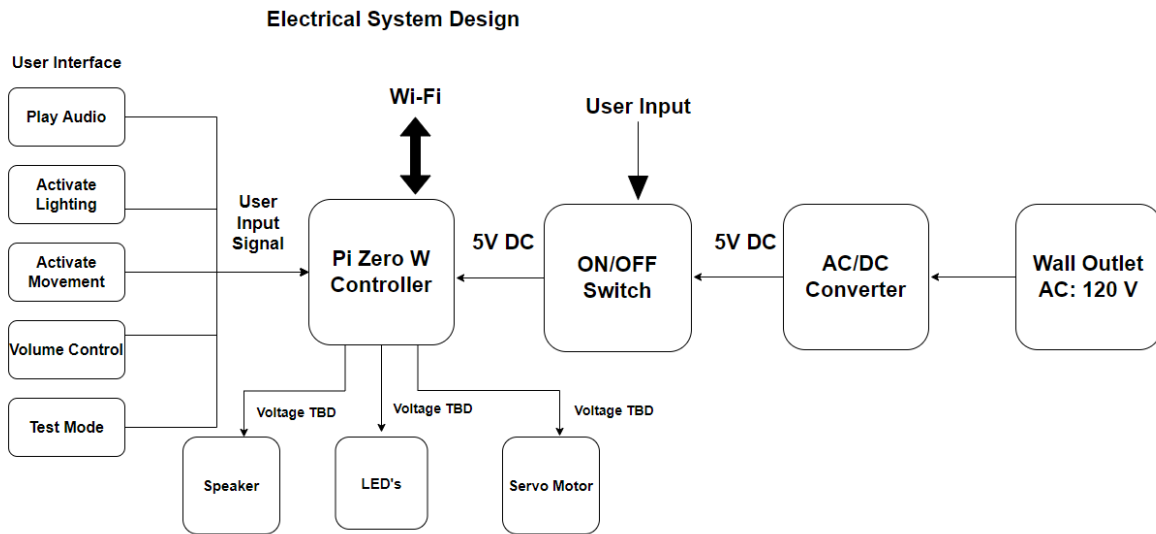


Figure 27. Electronic design highlighting interaction of components between sub-systems

Following the simulation of individual electrical subsystems and with these the development of their individual schematics, an overall PCB has been designed to combine these subsystem controls into a single circuit board. To view the individual schematics, refer to each subsystem above. The overall PCB design is shown below and in the process of being laid out and manufactured for the final product.

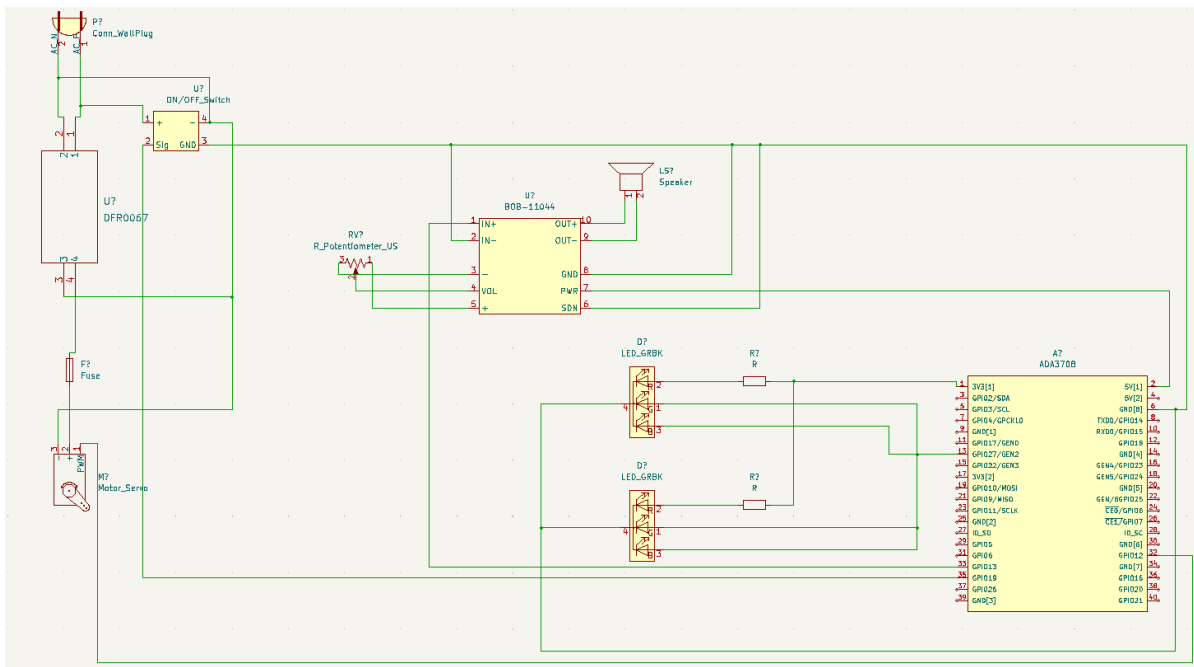


Figure 28. Full schematic for the PCB containing audio, power, lighting, and motor subsystems.

Electrical Simulation

Overall, electrical subsystem designs have evolved from the preliminary design through the simulation process. Components have been specified, chosen, and tested individually with breadboards and initial integration with the Raspberry Pi and Software subsystem has begun.

Audio

The design of the audio sub-system was reworked in order to incorporate the volume control function of the system. Initially, the audio output was delivered by the USB speaker. This setup facilitated connectivity between the Pi and the speaker, and it allowed for higher audio quality. However, after connecting the Pi with the speaker, it was revealed that incorporating audio control with this design was not optimal. To add volume control capability, an interface between the controller and the POT was necessary. The class D audio chip BOB-11044 provides inputs and outputs which allows to easily connect the Pi, the POT and the speaker. One tradeoff of using this sound chip is that the USB speaker couldn't be used anymore, which is the reason the speaker was replaced by a wired-connected speaker.

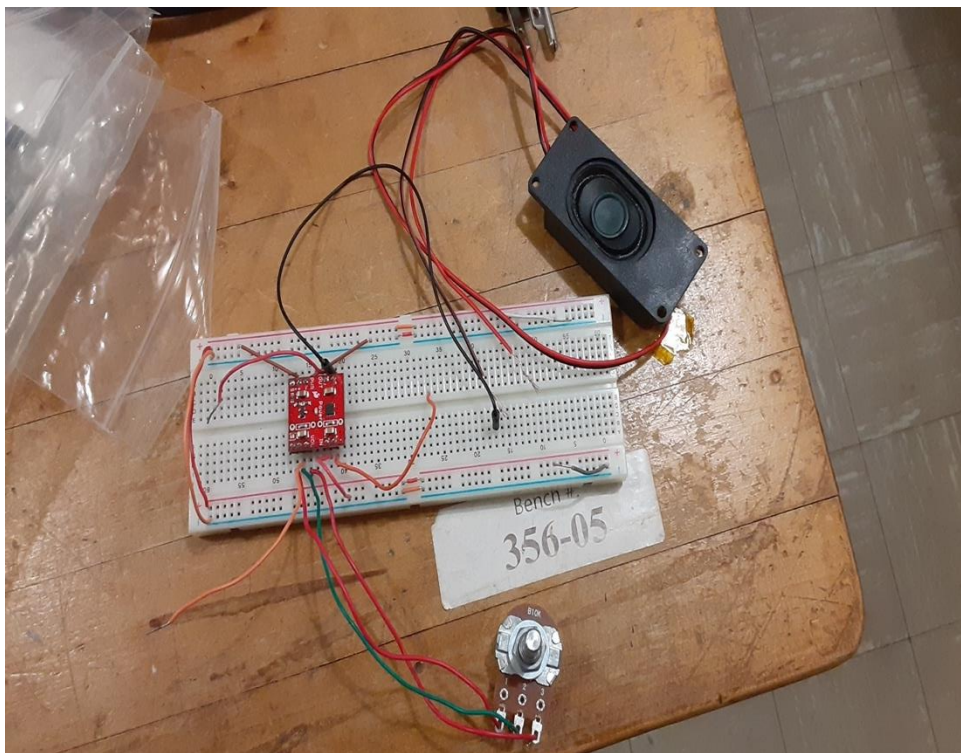


Figure 29. Audio sub-system setup with volume control

The BOB-11044 chip contains pins to connect the POT, input pins for audio signal, and power pins to connect the speaker. The first test done with this setup was to determine that it can play sound. To execute this, a sine wave produced by a function generator was fed to the IN+ pin of the chip. Once the other pins were properly connected, the system was powered up. The test was successful as the speaker produced a continuous tone. Following that result, the POT was then connected to the chip, and the second test began. With the same sine wave being fed to the chip, the volume of the speaker was successfully attenuated by adjusting the knob of the potentiometer. The sound quality of the system deteriorated slightly while still staying well within accepted levels.

Lights

The lighting subsystem design is now finalized in terms of components, necessary resistors, signals from the Raspberry Pi, and appropriate power voltage. An Mbed microcontroller was used in these simulations. Before simulating, a voltage of 3.3 V or 5 V DC was considered. The first simulation test used 3.3V to determine if the RGB LEDs were common cathode or common anode. After wiring the LEDs with 3.3V to the leads and ground to the common cathode, the lights turned on. See the figure below for the set up to test that the LEDs were common cathode vs common anode.

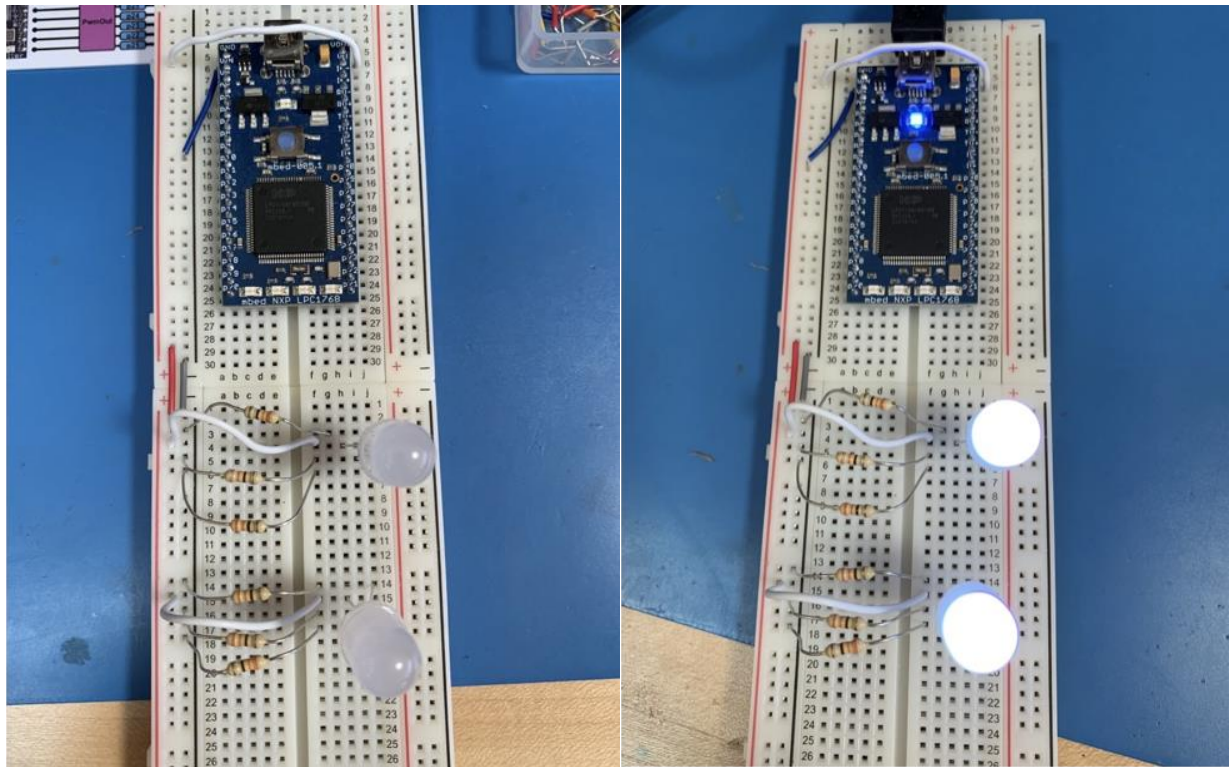


Figure 30. Left shows common anode set up with lights off. Right shows common cathode set up with lights on.

This first test used resistor values of 180 ohms for red and 330 ohms for blue and green to produce white light. Visual inspection showed these resistor values were too high as the lights were not as bright as intended. Consultation of the data sheet showed that the red lead requires 20 mA and a voltage between 1.8 to 2.2 V DC. Using Ohm's Law, it was determined that a resistor of 110 ohms is required for the red lead. The blue and green leads require a maximum of 3.3 and 3.4 volts, so it is not necessary to use any resistors for these leads.

To simplify the design for integration with the software/control subsystem, the red LED will always be connected to 3.3 V and therefore, always be on. The blue and green LEDs will be connected together and controlled by GPIO 27 from the Raspberry Pi. This allows for a GPIO signal to be sent and turn on the blue and green lights, along with the red that is always on, and produce a white light for Le Bleu's eyes. Earlier simulations used the Mbed to confirm changing the color of the lights with a GPIO signal and were successful. Later simulations used the Raspberry Pi to confirm that a Python script and GPIO signal would produce the same result. See the figures below for screenshots from a video simulation of the lights changing from red to white with code from the Raspberry Pi.

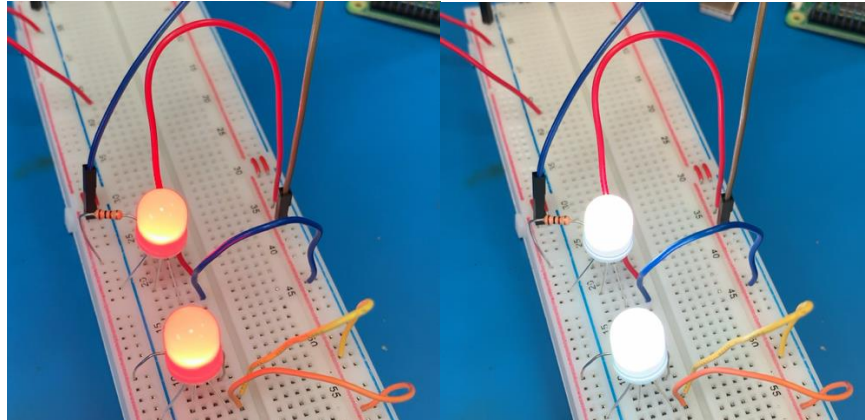


Figure 31. Left shows red lights powered from the Pi. Right shows white lights controlled by GPIO signal from Pi.

Power

During testing and simulations, the power sub-system design was modified to accommodate the new power consumption from the redesigned systems. The first version of the power system design didn't consider current drain from the sub-systems since it was determined that the Pi will handle this operation. As more systems were incorporated into the design, the complexity of the power system increased. Through testing, it was revealed that the servo motor draws more current than the controller can supply. Based on this, the option of directly connecting the motor to the power source was explored. To address the current issue of the motor, a DC/DC Buck converter was added to the power system. This converter steps-up the current while maintaining the same voltage output coming from the AC/DC converter.

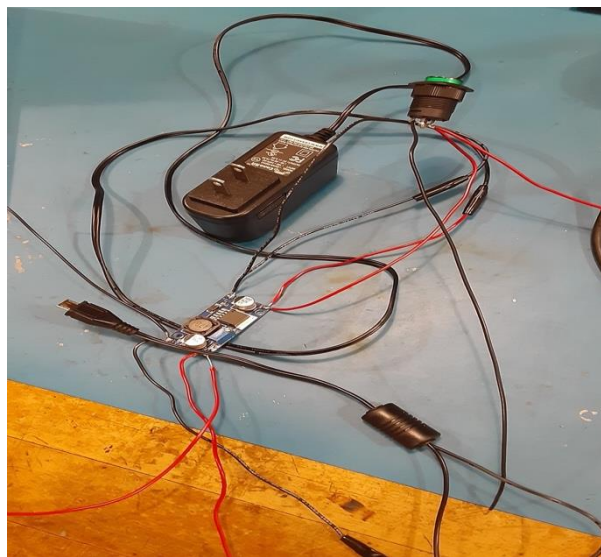


Figure 32. Power system with ON/OFF switch, buck converter, and additional wiring

For the test of the new setup, the stripped power wires of the AC/DC adapter were soldered to the input of the buck converter. From this input, extra wires were soldered to connect the pins of the ON/OFF switch to the controller, allowing for the status LED to be on when system is turned on. The wires that have the mini-USB connector that powers the Pi were soldered on the output of the buck converter. Pushing the button powered the Pi, making the first part of the test successful.

The second part of the test involved connecting the motor directly to the power system and verifying that the current drawn by the motor is addressed by the buck converter. The buck converter has a voltage adjuster that allows for the output voltage to stay in the range of 5V. The power of the motor was connected to the output of the buck converter, and the signal pin of the motor was connected to the controller. The results of this test were positive, as all components of these systems were working properly without any drop of voltage, nor any current issues occurred.

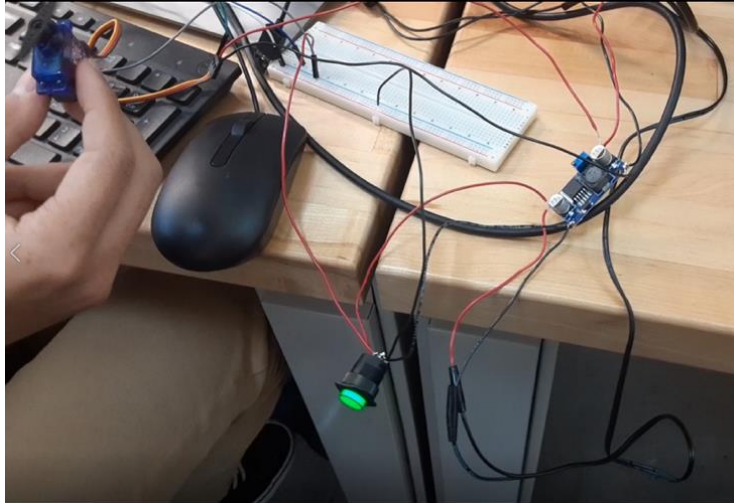


Figure 33. Power system and motor system integration

Mechanical Design

Le Bleu Caterpillar's mechanical design includes a servo motor to rotate the arm as the Caterpillar smokes his hookah. Le Bleu displays apparent growth and shrinking by rotating his arm 90° up and down. The mechanical designs are initially hand drawn and after simulations and testing, more finalized designs were created using Adobe Illustrator.



Figure 34. Final design of Le Bleu Caterpillar sitting on the top of his mushroom

Le Bleu Caterpillar's aesthetic design is inspired by Disney's version in the cartoon Alice in Wonderland movie from 1951. Therefore, the design is more two dimensional. The team laser cut and painted Le Bleu Caterpillar's body, arm, and mushroom from wood. The body and mushroom are a single piece for ease of assembly. The servo is mounted behind Le Bleu's body to control his arm movement. Initial designs included a separate base for the mushroom, but the team refined the design to remove the stem and simply have Le Bleu Caterpillar rest upon the cap of the mushroom on top of the box.

The box that houses the electronics and that Le Bleu sits on was laser cut from wood and contains a front panel for the user interface and a back acrylic panel to view the electronics. The box design follows a finger joint design. To facilitate debugging and assembly, the acrylic back panel was mounted with a hinge. Supports for the mushroom and Le Bleu were designed and laser cut from wood. Testing was done with initial cardboard cutouts of Le Bleu to determine the necessary thickness of the body/mushroom base and arm.

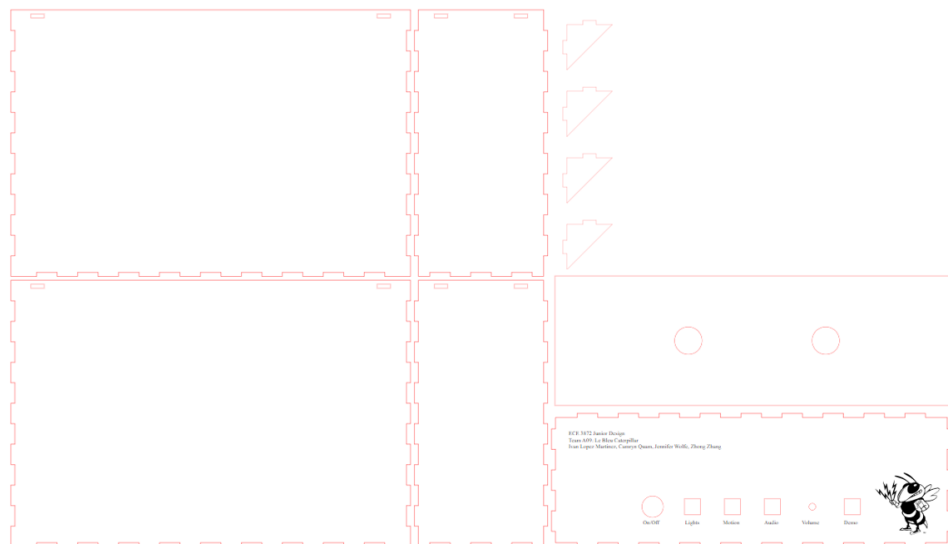


Figure 35. Final design of the box, including UI panel and finger joints. The back panel will be acrylic and mounted with a hinge.

Mechanical Simulation

Mechanical simulations for Le Bleu's mechanical design included simulating the motor's motion and fabricating a test model for his figure design out of cardboard. After the motor was simulated successfully on its own, it was later integrated with the figure model to test the arm motion.

Motor simulations were run using an Mbed microcontroller. The first simulation was a basic sweep of the motor's possible positions to ensure that the motor was functional and could provide appropriate angles of motion. Following simulations focused on moving the arm in both directions and adjusting the speed at which it would move. The simulations proved successful.

For the mechanical model, the figure was designed in Adobe Illustrator and laser cut in The Hive. For the first cut, cardboard was used. The cardboard model turned out well and showed what adjustments need to be made for the final product. The main adjustments included adding holes for the LED eyes and the motor. In addition, supports were needed to attach the caterpillar to the box.

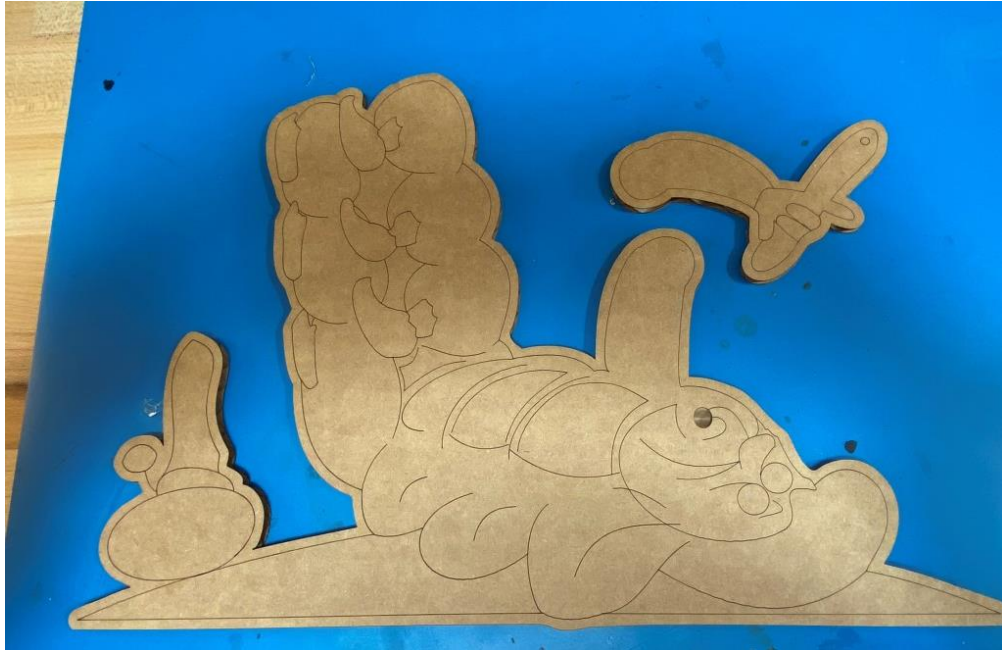


Figure 36. Initial laser cut of figure out of cardboard.

When mounting the motor to the cardboard model, a hole needed to be manually cut out in order to mount it successfully. The main objective of the integrated simulation was to observe whether the arm movement was satisfactory and whether the figure's design would need to be modified. The movement of the figure's arm worked as expected.

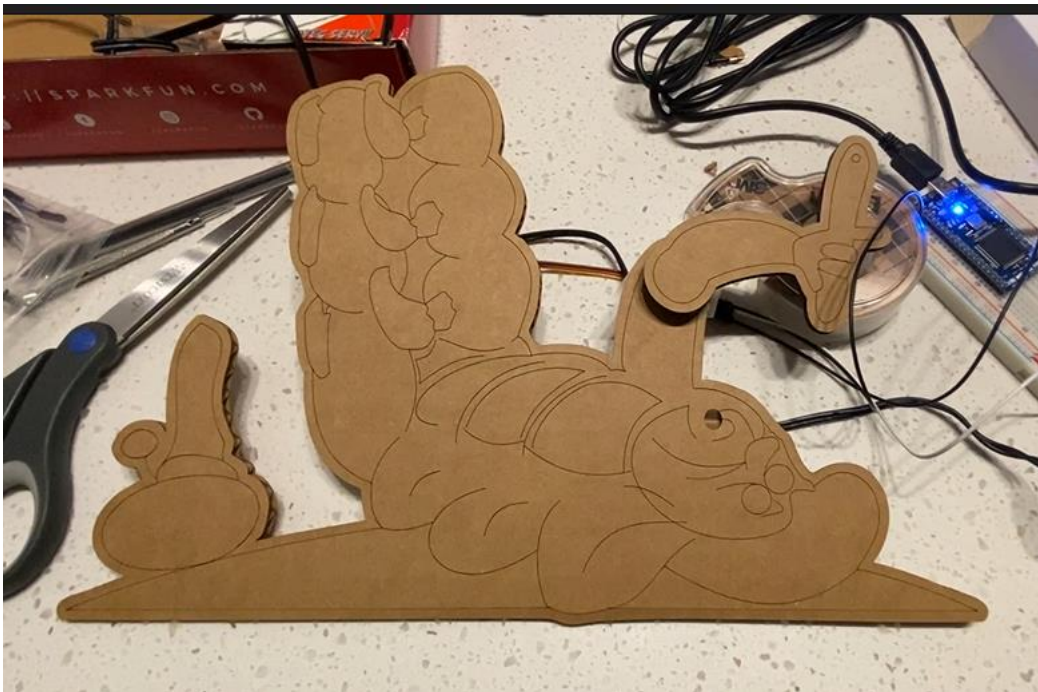


Figure 37. Integrated simulation testing the arm motion produced by the motor.

Software Design

There are five stages in our software design: Idle stage where the microprocessor is waiting to be powered on, Initialization stage where the system will start detecting the status of the test button, Listening stage where the microprocessor will start connecting to the server, Decode stage where the scripts received will be decoded for playing, and finally Action stage where the system will do the assigned actions. Relations and transitions between each state are illustrated below by Figure 34.

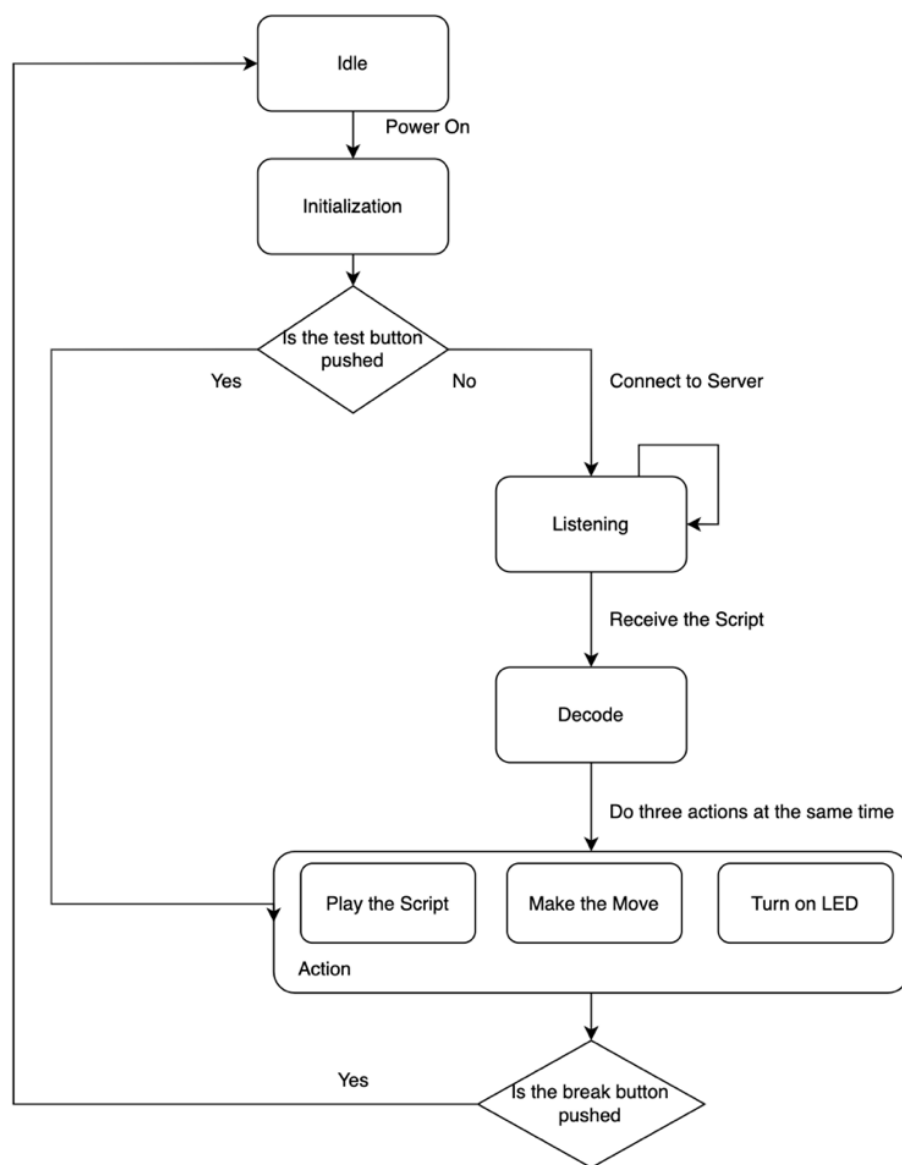


Figure 38. Software state flow diagram

The Software Architecture Diagram illustrates the structure of the software design. We have four different classes. The control/main class receives the script file from the server and creates corresponding objects for other classes based on user inputs. The light class, motion class, and audio class include all the necessary functions used to drive other subsystems.

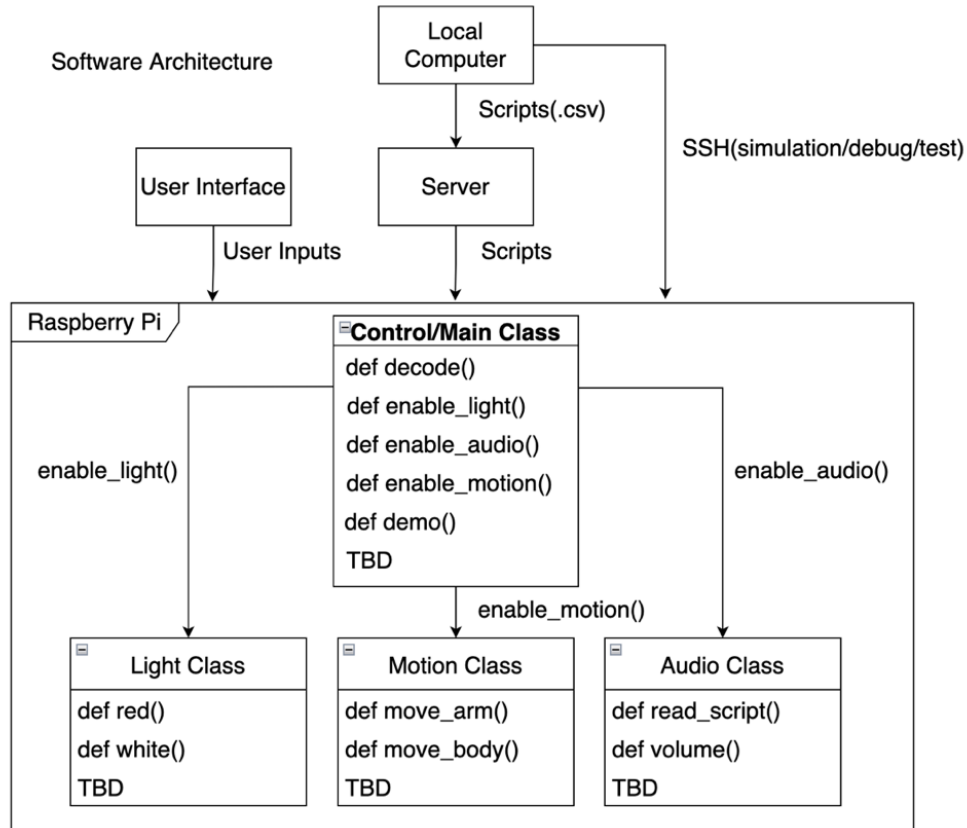


Figure 39. Software architecture diagram

Software Simulation

1. Simulation of the code in Linux system:

This is to check if the given legacy code works properly. The following screenshots from the terminal show that the code works well under a Linux system both for director and the robot. The director can listen for robots and robots can be registered to the director.

```

root@ubun-VirtualBox: /home/ubun/Desktop/3872/ECE-3872/Projects/Wonderland# sudo
python3 director.py test.csv
Creating Registration and Key Process
Starting Registration and Key Process
#####
Director listening on ('127.0.0.1', 65432) for registration
#####

Accepted connection from ('127.0.0.1', 34212)
Received request {'name': 'Robot1', 'message': 'Register', 'listenPort': 65433}
from ('127.0.0.1', 65433)
Closing connection to 127.0.0.1

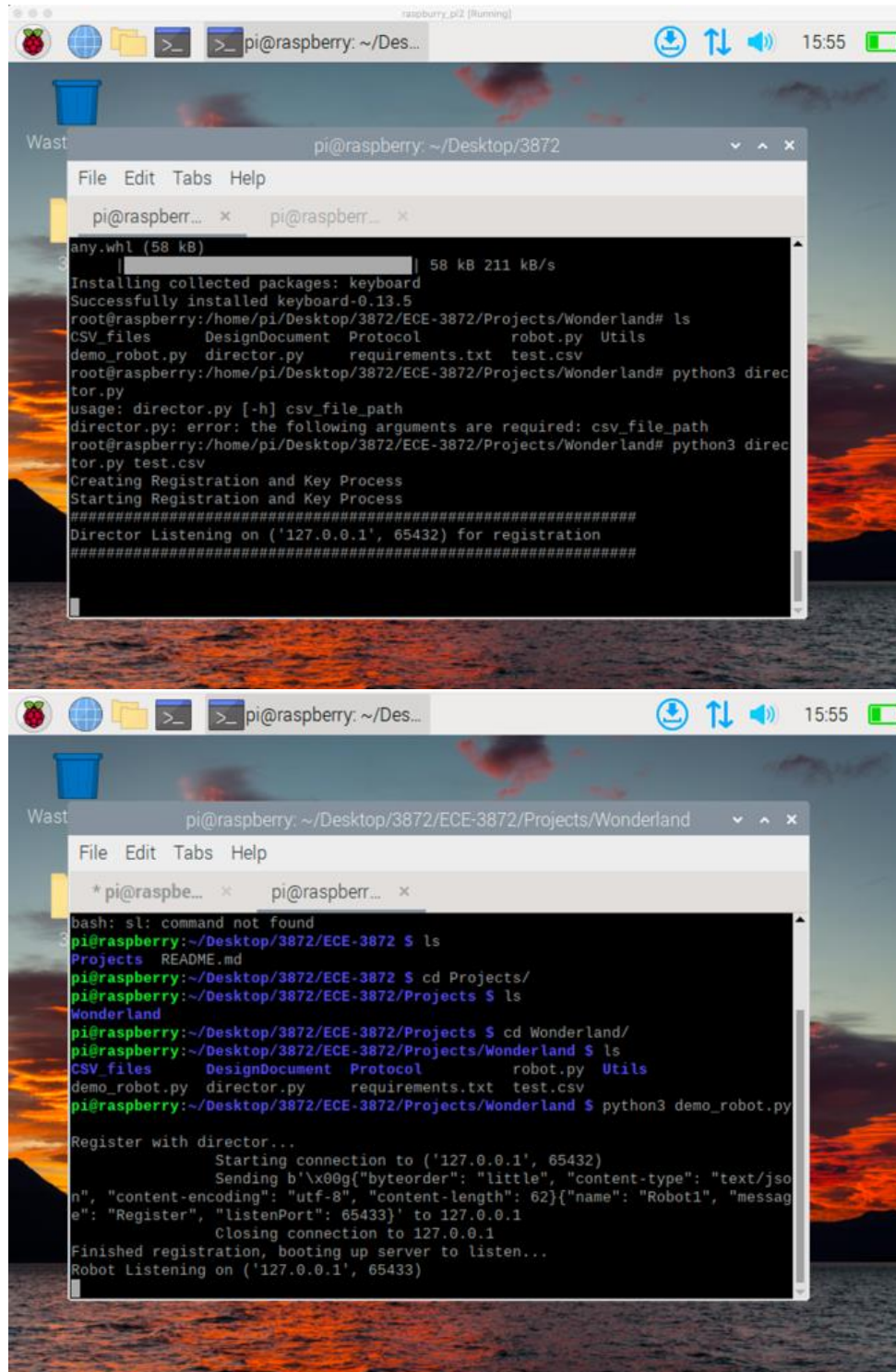
#####
Added Robot1 from ('127.0.0.1', 65433)
There are now 1 robots registered
Robot Name: Robot1 IPv4: 127.0.0.1
Press Q key after all robots have registered
#####

root@ubun-VirtualBox: /home/ubun/Desktop/3872/ECE-3872/Projects/Wonderland# pyth
on3 demo_robot.py test.csv
Register with director...
Starting connection to ('127.0.0.1', 65432)
Sending b'\x00g{"byteorder": "little", "content-type": "text/js
on", "content-encoding": "utf-8", "content-length": 62}{"name": "Robot1", "mess
age": "Register", "listenPort": 65433}' to 127.0.0.1
Closing connection to 127.0.0.1
Finished registration, booting up server to listen...
Robot listening on ('127.0.0.1', 65433)
  
```

Figure 40. Linux system code simulation

2. Simulations of the code in Raspberry Pi Zero Desktop:

This is to check if the given legacy code works properly inside the Raspberry Pi's operating system. The following screenshots from the pi's terminal show that the code works well both for director and the robot. The director can listen for robots and robots can be registered to the director.



```
pi@raspberr... x  pi@raspberr... x
any.whl (58 kB)
Installing collected packages: keyboard
Successfully installed keyboard-0.13.5
root@raspberry:/home/pi/Desktop/3872/ECE-3872/Projects/Wonderland# ls
CSV_files  DesignDocument  Protocol      robot.py  Utils
demo_robot.py  director.py  requirements.txt  test.csv
root@raspberry:/home/pi/Desktop/3872/ECE-3872/Projects/Wonderland# python3 direc
tor.py
usage: director.py [-h] csv_file_path
director.py: error: the following arguments are required: csv_file_path
root@raspberry:/home/pi/Desktop/3872/ECE-3872/Projects/Wonderland# python3 direc
tor.py test.csv
Creating Registration and Key Process
Starting Registration and Key Process
#####
Director Listening on ('127.0.0.1', 65432) for registration
#####

pi@raspberr... x  pi@raspberr... x
bash: sl: command not found
pi@raspberry:~/Desktop/3872/ECE-3872 $ ls
Projects  README.md
pi@raspberry:~/Desktop/3872/ECE-3872 $ cd Projects/
pi@raspberry:~/Desktop/3872/ECE-3872/Projects $ ls
Wonderland
pi@raspberry:~/Desktop/3872/ECE-3872/Projects $ cd Wonderland/
pi@raspberry:~/Desktop/3872/ECE-3872/Projects/Wonderland $ ls
CSV_files  DesignDocument  Protocol      robot.py  Utils
demo_robot.py  director.py  requirements.txt  test.csv
pi@raspberry:~/Desktop/3872/ECE-3872/Projects/Wonderland $ python3 demo_robot.py
Register with director...
Starting connection to ('127.0.0.1', 65432)
Sending b'\x00g{"byteorder": "little", "content-type": "text/jso
n", "content-encoding": "utf-8", "content-length": 62}{"name": "Robot1", "messag
e": "Register", "listenPort": 65433}' to 127.0.0.1
Closing connection to 127.0.0.1
Finished registration, booting up server to listen...
Robot Listening on ('127.0.0.1', 65433)
```

Figure 41. Raspberry Pi Zero code simulation

3. Simulations for light control code

The lights are controlled by the Raspberry Pi. Below is a simulation script in Python showing the controlling code sample. The simulation code aims to turn the light on and off with a one second gap. Once the python simulation script is executed and the pins from 27 and 14 are connected as the input and the ground, the lights successfully turn on and off as expected.

```
import RPi.GPIO as io
import time
io.setmode(io.BCM)
switch_no = 27
switch_ground = 14

io.setup(switch_no, io.OUT)
io.setup(switch_ground, io.OUT)
while 1:
    io.output(switch_no, True)
    time.sleep(1)
    io.output(switch_no, False )
    time.sleep(1)
    io.output(switch_ground, True)
```

Figure 42. Raspberry Pi Zero code simulation for light control

4. Simulations for motor control code

The motor is controlled by the Raspberry Pi. Below is a simulation script in Python showing the controlling code sample. The simulation code aims to rotate the servo to minimum angle, middle angle, and maximum angle with a 0.5 second gap. Once the python simulation script is executed and pin 25 is connected as the input, the servo successfully rotates as expected.

```
from gpiozero import Servo
from time import sleep
servo = Servo(25)
val = -1

while True:

    servo.min()
    sleep(0.5)
    servo.mid()
    sleep(0.5)
    servo.max()
    sleep(0.5)
```

Figure 43. Raspberry Pi Zero code simulation for motor control

5. Simulations for speaker control code

The speaker is controlled by the Raspberry Pi. Below is a simulation script in Python showing the controlling code sample. The simulation code aims to play a song using pre-defined frequencies. Once the python simulation script is executed and pin 22 is connected as the input, the speaker successfully plays the song as expected.

```
import RPi.GPIO as GPIO
import time

BuzzerPin = 22

GPIO.setmode(GPIO.BCM)
GPIO.setup(BuzzerPin, GPIO.OUT)
GPIO.setwarnings(False)

global Buzz
Buzz = GPIO.PWM(BuzzerPin, 440)
Buzz.start(50)

B0=31
C1=33
CS1=35
D1=37
DS1=39
E1=41
F1=44
FS1=46
G1=49
GS1=52
A1=55
AS1=58
B1=62
C2=65
CS2=69
D2=73
DS2=78
E2=82
F2=87
FS2=93
G2=98
GS2=104
A2=110
AS2=117
B2=123
C3=131
CS3=139
D3=147
DS3=156
E3=165
F3=175
FS3=185
G3=196
GS3=208
A3=220
AS3=233
B3=247
C4=262
CS4=277
DS4=278

song = [
    G4,
    E4, F4, G4, G4, G4,
    A4, B4, C5, C5, C5,
    E4, F4, G4, G4, G4,
    A4, G4, F4, F4,
    E4, G4, C4, E4,
    D4, F4, B3,
    C4
]

beat = [
    8,
    8, 8, 4, 4, 4,
    8, 8, 4, 4, 4,
    8, 8, 4, 4, 4,
    8, 8, 4, 2,
    4, 4, 4, 4,
    4, 2, 4,
    1
]

while True:
    for i in range(1, len(song)):
        Buzz.ChangeFrequency(song[i])
        time.sleep(beat[i]*0.13)
    ~
    ~
```

Figure 44. Raspberry Pi Zero code simulation for speaker control

Schedule

Task	Week Number											
	1	2	3	4	5	6	7	8	9	10	11	12
Brainstorm	Jennifer											
Design Top Level Block Diagram		Camryn										
Design Software Block Diagram		Zhong										
Schedule		Ivan										
Decide on overall design/Sketch		Jennifer										
Establish requirements		Jennifer										
Document/Presentation for Proposal		Jennifer										
Proposal												
Software/Controls Design			Zhong	Zhong								
Software - update design				Zhong								
Software Simulation - set up PI			Zhong									
Lighting Design - initial schematic				Jennifer								
Audio Design - initial schematic				Ivan								
Power Design - initial				Ivan								
Motion Design - initial				Camryn								
User Interface Design - initial				Camryn								
Gather components				Ivan								
Test software Development				Zhong								
PDR Test Plan					Camryn							
Updated hierarchical design					Jennifer							
Document/Presentation for PDR					Jennifer							
PDR												
Software Build/Simulations					Zhong	Zhong	Zhong					
Software Build					Zhong							
Software - test other system integration						Zhong						
Software Simulations							Zhong					
Electrical Design					Ivan	Ivan	Ivan	Ivan				
Power breadboard						Ivan						
Power simulations							Ivan					
Software/Power integration							Ivan					
Power PCB/update schematic							Ivan					
Audio design						Ivan						
Audio breadboard							Ivan					
Audio PCB/update schematic								Ivan				
Find audio file								Jennifer				
Light breadboard						Jennifer						
Software/Light Integration							Jennifer					
Light PCB/update schematic								Jennifer				
Integrate all PCB designs								Ivan				
Mechanical Design					Camryn	Camryn	Camryn	Camryn				
Motor breadboard					Camryn							
Motor simulations						Camryn						
Motor integration with software								Camryn				
Caterpillar figure design						Camryn						
Caterpillar figure test manufacture							Camryn					
Mechanical box design							Camryn					
User Interface breadboard							Camryn					
User Interface simulations								Camryn				
Test plan								Camryn				
Bill of Materials								Jennifer				
Simulation Videos								Jennifer				
Document for CDR								Jennifer				
CDR												
Software Integration/Test								Zhong	Zhong	Zhong	Zhong	
User Interface Integration/Test								Zhong				
Audio Integration/Test									Zhong			
Motion Integration/Test										Zhong		
Overall Software Tests										Zhong		
Final Debugging											Zhong	
Mechanical Build/Integration/Test									Camryn	Camryn	Camryn	
Caterpillar figure final manufacture									Camryn			
Caterpillar figure-paint									Camryn			
Mechanical box build									Camryn			
Motion integration with figure										Camryn		
User Interface build/mechanical integration										Camryn		
Final testing											Camryn	
Electrical Build/Integration/Test									Ivan	Ivan	Ivan	
PCB fabrication									Ivan			
Build - Lights										Jennifer		
Build - Audio										Ivan		
Build - Power										Ivan		
Test - Power											Ivan	
Test - Lights											Jennifer	
Test - Audio											Ivan	
Electrical Integration with mechanical												Ivan
Final System Integration												Camryn
Final System Test												Zhong
Final Documentation/Presentation												Jennifer
Final Inspection and Demonstration												

Figure 45. Final Schedule for the development of Le Bleu Caterpillar including task leads

Integration

For the system to successfully perform requirements, each sub-system needed to be able to work together. A series of simulations and tests were performed for each individual sub-system, as well as between sub-systems.

First, simulations were done for the software sub-system, followed by tests for the rest of the sub-systems. After each sub-system passed tests individually, tests between sub-systems were performed to ensure that they work together correctly. This step-by-step approach helped prevent major obstacles from taking place and allow for smaller obstacles to be solved along the way within a reasonable timeframe.

The first simulations that took place were for the software sub-system with the Raspberry Pi Zero. Tests for lights, audio, and motion were performed using simplified test files to ensure that they are functional. Once functionality had been confirmed, tests to meet system requirements were performed.

Software High-Risk Parts

The programming for the Raspberry Pi Zero was done in Python, which our team was not highly skilled in. Similarly, our team did not have experience with any Raspberry Pi microcontrollers. Background and prior experience with other microcontrollers and programming languages allowed the team to adapt to using the Raspberry Pi for the system.

Electrical High-Risk Parts

In terms of power for the system, the motor presented risk as motors tend to require a large amount of current. It was determined that the motor would be powered from an external power source to avoid drawing too much current from the Pi.

Power and the motor systems were successfully integrated by connecting the motor to the power source using a buck converter to correct the issue of current supply. A fuse connected between the motor power and the output of the buck converter was considered to help reduce the risk of current surges.

Risk involving sound system and volume control was addressed by including a sound chip that takes the input from the potentiometer and attenuates the sound signal from the Pi.

Lighting system issues were addressed by testing different resistor values and by connecting the 3.3V from the Pi to the red LED and having it always be turned on. The green and blue LED are connected and share the same GPIO to minimize signals from the Pi. Their function is to change the lights from red to white.

Mechanical High-Risk Parts

The structure of the system must be stable enough to support the components. Specifically, the team needed to consider how the motor will be mounted for arm movement. The material on the moving part needed to be light enough for the motor to support, while the section that holds the motor needed to be sturdy enough so as not to produce any shaky movements when the motor is rotating. This risk was evaluated and mitigated through the testing of both cardboard and wood. Wood proved to be light enough to provide smooth movement, thus this is the material used in Le Bleu's design.

Configuration Controls:

The user interface for the system consists of four pushbuttons, a switch, a potentiometer, and an LED for user experience. These were integrated into the system through the microprocessor and PCB.

Repository Management:

For software repository management, GitHub contains all versions of the code. For deliverables, sketches, ideas, schematics, etc., the files are stored and shared in Microsoft Teams and the team notebook.

Final Build

Electrical:

The electrical system integration into the final product caused a few issues due to PCB production, but the team was able to adapt and successfully overcome these challenges.

Lights:

Two RGB LEDs were incorporated into the figure by soldering the leads to wires and connecting these wires to the PCB. The PCB handles all connections to the Raspberry Pi for the GPIO signal, power, and ground. Heat shrink was added to contain the wires and insulate all solder joints. Additionally, the LEDs serve as a visual indicator for when the Raspberry Pi is powered up and executing the stored code. The Red leads are connected to power, but on power up, the Pi sends a signal to change the color to white. Thus, white LED lights indicate successful connection to the Pi. The lighting subsystem of the project worked as expected and met the requirements established by the client.

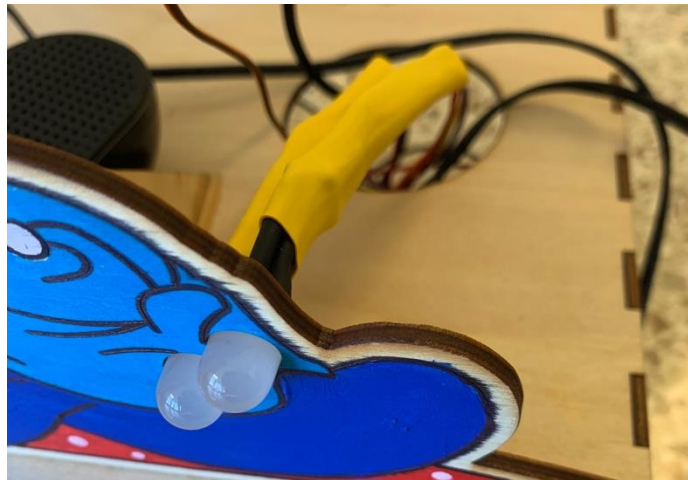


Figure 46. Assembled LEDs

Audio:

At the time of integrating the system to the PCB, the audio system ceased to work. The speaker wasn't outputting any audible sound and the potentiometer did not control the volume. After some testing, the team narrowed down the causes of this failure into two possible reasons: a suspected bridge between the positive and negative terminals of the speaker caused by close traces on PCB and low resistance in

the speaker artificially bridging the connections. The team reverted to the original audio subsystem design of a USB speaker; however, this resulted in losing volume control but gaining sound again. The team left the original analog circuit including the potentiometer and amplifier chip in the machine because of the time and effort spent in developing this circuit. Once the product was fully assembled, the audio system was successful in producing sound through the USB speaker that met the requirements established by the client.

Power:

Further testing revealed that the 3.3 V and 5V outputs of the Pi were not enough to power all subsystems of the robot since the servo motor required more current. The team modified the power system to include a buck converter whose 5V output was the source of power for the motor. The wires from the adapter are connected to the input of the buck converter. Additionally, the signal for the LED of the ON/OFF switch is connected to this input. The power and ground of the motor are connected to the output of the buck converter. Testing for this configuration was successful as the buck converter was able to supply the motor with the current needed to operate.

Integrating the power system to the PCB was mostly successful, except for the LED of the ON/OFF switch. This LED was working initially, but ultimately stopped working as more testing was done throughout the integration. One possible reason the LED stopped working might be that the power traces of the PCB were not wide enough to properly power the switch. Aside from the LED of the switch, the power system worked as intended and met the requirements established by the client.

PCB:

Schematics and layout for the PCB were done using the software KICAD. Upon testing, various complications were discovered. Some of the holes drilled into the board were not the right size to connect the components or place through-hole parts. This issue required the team to use wires to attach components that could not fit through the holes. Additionally, some of the traces of the board were very close to others. This made soldering difficult and may have resulted in some connectivity issues involving the power and audio system. Further, some traces had 90° angles, which goes against the convention of routing traces with 45° turns, which helps with signal transmission. This error was due to the autoroute function through KICAD. Another aspect of the traces that could be improved in the future is to increase the thickness of the power lines of the board. In the future, the team could use breakout boards to also minimize the number of wires or design the interface in such a way that the PCB could include the buttons and fit inside the box. Aside from the flaws mentioned, the overall performance of PCB met the team's expectations and the requirements of the client.

Mechanical:

For the final mechanical build, our box and figure were laser cut out of wood and a clear back panel for the box was cut out of acrylic. The wooden figure was hand painted. 3-D printer filament was used for the hookah string. The motor was mounted with screws between the two arm segments, and the acrylic

panel was attached to the box with a hinge to provide ease of access to electronics. Buttons for the user interface were secured into their designated holes using hot glue and the printed circuit board and buck converter were mounted on the inside of the box using screws and stand-offs. The figure of the Caterpillar was secured to the box with wood glue, and triangular wood pieces were used as supports. All features meet the requirements established by the client.



Figure 47. Final mechanical build of Le Bleu Caterpillar

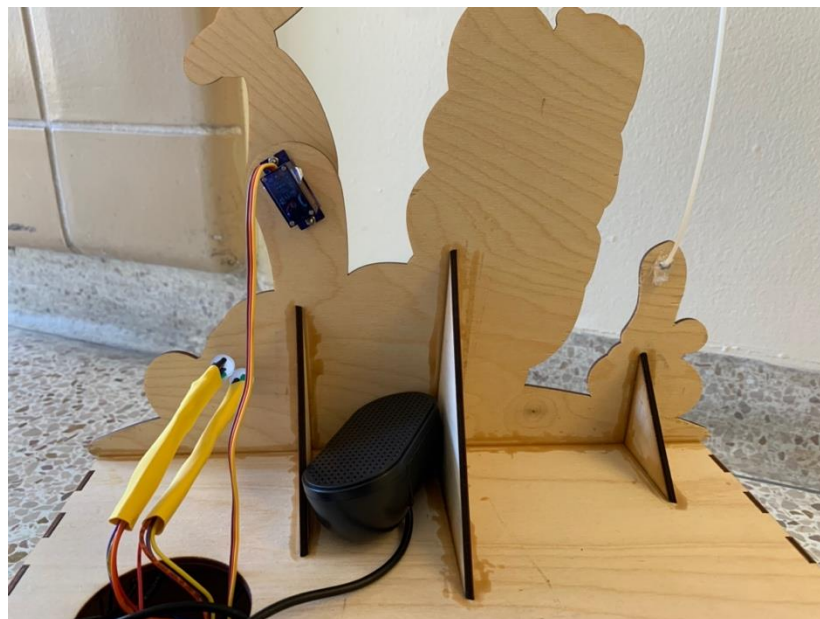


Figure 48. Triangle mounts to hold up and mount the figure to the box

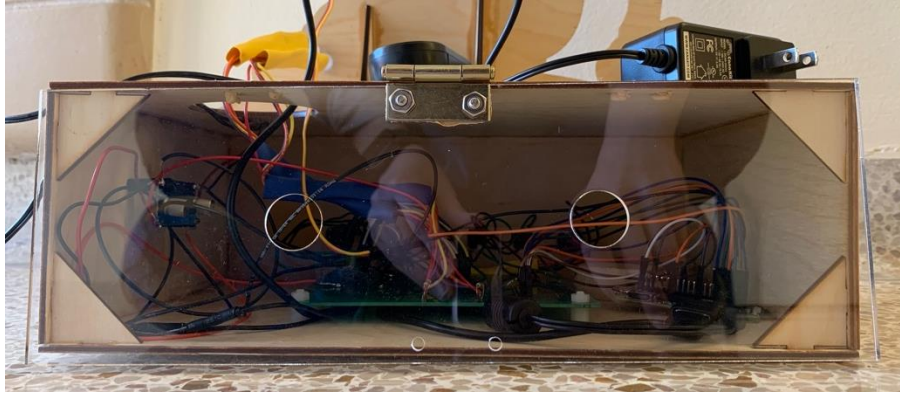


Figure 49. Acrylic back panel to show electronics

Software:

In order to have the speaker, light, and motor code run at the same time, a multiprocessing library from Python is used. Once the demo button is pushed, three functions will be executed at the same time.

```
73
74 while 1:
75
76     if GPIO.input(11) == GPIO.LOW:
77         print("demo has been pushed, everything will start")
78         p1 = Process(target=light)
79         p2 = Process(target=motor)
80         p3 = Process(target=speaker)
81         p1.start()
82         p2.start()
83         p3.start()
84         p1.join()
85         p2.join()
86         p3.join()
87
```

Figure 50. Multiprocessing code for the demo button

The system has the ability to connect remotely to a Director computer over a Wi-Fi connection. The team tested this ability on a local network. After a connection is established, the Director can send commands to the Raspberry Pi to control functionality remotely.

Final Integration:

The final integration of our system involved connecting all the subsystems together through the printed circuit board and securing everything to the mechanical structure. Components were connected to the PCB through soldered wire connections, and heat shrink was used to provide insulation and safety. The PCB and buck converter were mounted inside of the box using standoffs and screws, and the User Interface pushbuttons were secured into their designated holes with hot glue.

Overall, integration of the subsystems went smoothly. Once all the components were connected, final software adjustments were made to provide the desired performance from the robot.

Due to Wi-Fi issues, the demo in class was completed by using the pushbuttons to interact with features, rather than the Director.

Conclusion

The design for Le Bleu Caterpillar began with formulating requirements based on the customer's need for a robot thespian from Alice in Wonderland. By breaking the system into sub-systems and assigning task leads, Team A09 successfully created Le Bleu Caterpillar over 12 weeks. Each sub-system was assigned to a specific team member to oversee the design, testing, and integration. This allowed for parallel development and the ability to remain on target with our team's schedule throughout the semester. As the project entered the phase after the preliminary design review and simulations, the team met more often to collaborate and discuss successes and problems. Ultimately, this division of labor and team collaboration led to the successful demonstration of Le Bleu Caterpillar.