Design Document

Willy the White Wabbit

Team A04: Allison Sawyer, Eugene Min, Rayan Dabbagh, Sophie Ayoung-Chee

Date: November 13th 2022

Table of Contents

Revision Record

Date	Author	Comments
Sept 14, 2022	Team	Document created (description, system design)
Sept 16, 2022	Dabbagh, Rayan	Reworked Software Subsystem
Sept 21, 2022	Team	Edit and review
Sep 25, 2022	Ayoung-Chee, Sophie	Added signal interface table, power budget
Sep 25, 2022	Sawyer, Allison	Added most current schedule
Sep 25, 2022	Min, Eugene	Added most recent Hardware Block Diagram
Sept 25, 2022	Sawyer, Allison	Review before PDR submission: formatting, grammar, table and figure captions, page breaks
Oct 12, 2022	Ayoung-Chee, Sophie	Added additional images, Updated Power subsection, Added Electrical Design section.
Oct 12, 2022	Dabbagh, Rayan	Added Software Simulation section, Created Simulation Results section
Oct 12, 2022	Min, Eugene	Added Mechanical Design section
Oct 13, 2022	Sawyer, Allison	Added Robot Design section
Oct 14, 2022	Sawyer, Allison	Added most recent schedule
Oct 14, 2022	Dabbagh, Rayan	Updated Software Simulation section
Nov 9, 2022	Dabbagh, Rayan	Added Integration section
Nov 11, 2022	Ayoung-Chee, Sophie	Added Conclusion section
Nov 11, 2022	Sawyer, Allison	Edited Integration Section, added Constraints section
Nov 12, 2022	Min, Eugene	Updated Mechanical Design Section with Results

Project Description

We have designed a White Rabbit robot based on the character of the same name from Alice in Wonderland. White Rabbit will play an audio clip from the animated movie, where he is exclaiming that he is “Very late!”. As he does this, his pocket watch will move away from his body, light up, and the hands on the clock will begin to spin. His red eyes will also glow as he panics about being late. When the audio is over, his eyes stop glowing and his pocket watch goes back to being ‘hidden’ and inactive.

Figure 1: Our second sketch (left) and final design (right) of the White Rabbit. Different from our preliminary design, the rabbit now rests on a vertical box rather than a horizontal box so that our components are not exposed.

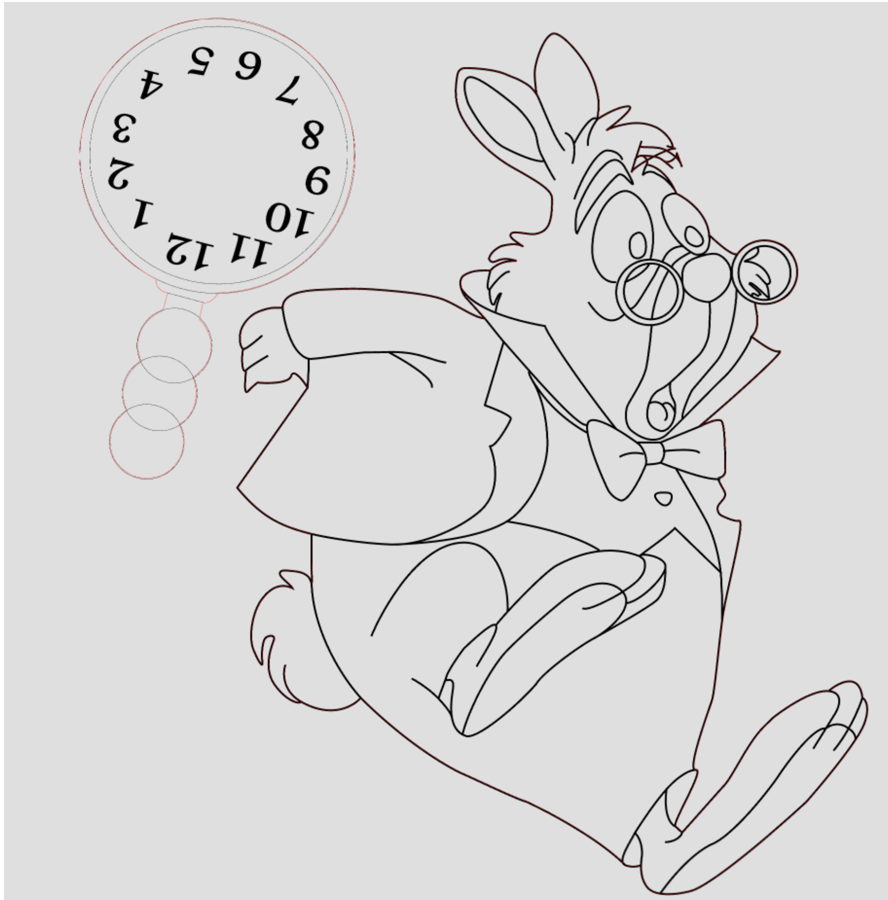

Figure 2: Vector design completed in Figma to be laser cut.

Figure 3: Preliminary laser cut of our design done in cardboard. A 0.25 scaled version was completed first.

System Design

The inputs of the system include 120V AC input power and user input. The outputs of this system include audio from a speaker, lights from LEDs, and movement produced by motors. The system consists of five sub-systems: power, control, audio, light, and mechanical. The user interface includes four pushbuttons for testing, and each pushbutton performs a distinct test for a particular subsystem. 'Move' tests mechanical, 'Light' tests light, 'Audio' tests audio. The fourth pushbutton 'Animate' performs all actions in parallel. Additionally, there is an on/off switch and a control knob to adjust the speed of the clock hands. The processor consists of a Raspberry Pi Zero 2. The power sub-system includes a 120V AC to 5V DC converter and a user input switch. The audio sub-system includes a 4W speaker. The light sub-system consists of two red LEDs. The motion sub-system includes one servo motor and 1 DC motor.

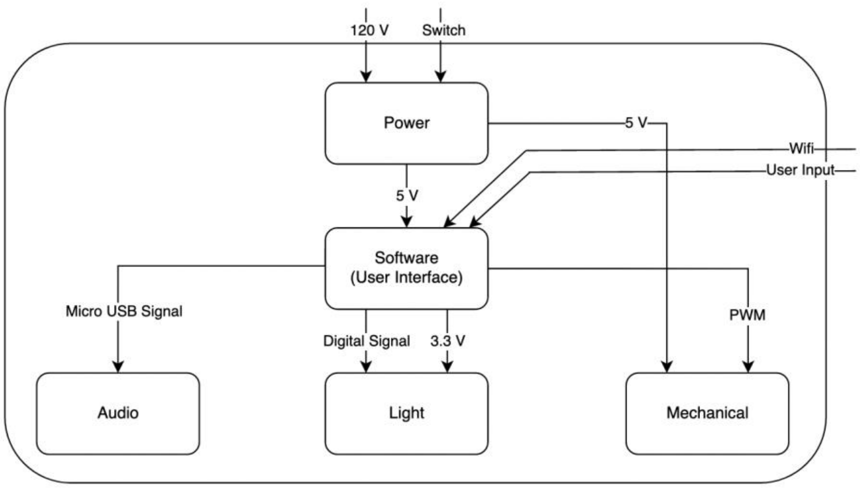

Figure 4: System Diagram.

Interface	Source	Destination	Description
120V AC	System Input	Power	120V AC input power from wall
User Input	User Input	Software	Digital input to the control system
Wi-Fi Input	Wi-Fi	Software	Commands from the director over Wi-Fi
Micro USB Signal	Software	Audio	Micro USB input of audio track
3.3V DC	Software	Light	Power to control LEDs
5V DC	Power	Mechanical	Power to control servo motors
		Software	5V DC used by the processor
		Audio	Power to control audio modules
PWM	Software	Mechanical	PWM signals to control movements
Digital Signal	Software	Light	Digital signal controls if lights are on

Table 1: Sub-system inputs and outputs.

Sub-System Designs

Our design incorporates five sub-systems: software, power, mechanical, audio, and lights. Audio will process the sound file and amplify it. Mechanical will execute the movement of the figure. Lights will turn on while audio plays. Power will convert 120V AC to 5V DC. Software will integrate all the sub-systems by converting all user input to signals to actionable items for the robot.

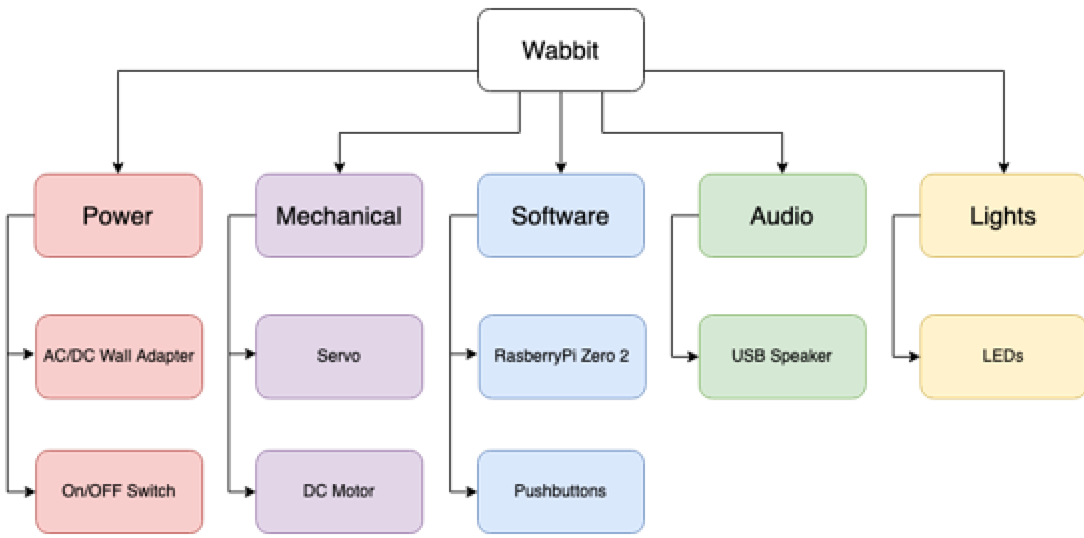

Figure 5: Hierarchical Diagram.

Software Sub-System

Task Lead: Rayan Dabbagh

Upon start up the system will go into an idle state until it either receives a message from the server, or a test button is pushed. The Software subsystem will take four distinct pushbuttons as inputs and trigger specific actions depending on which pushbutton was pressed. When “Move” is pressed, the system will rotate the pocket watch out, and the hands of the clock spin for 10 seconds. When "Light” is pressed, the system LEDs will flash for 10 seconds. When "Audio” is pressed, the system speaker will produce a sound (generated by an audio file, mp4). When "Animate” is pressed, the system will simultaneously produce all the movements, until the end of the audio file is reached. The system will decode the files received and make a queue of commands. Each of the commands will start a thread in which the corresponding subsystem will act.

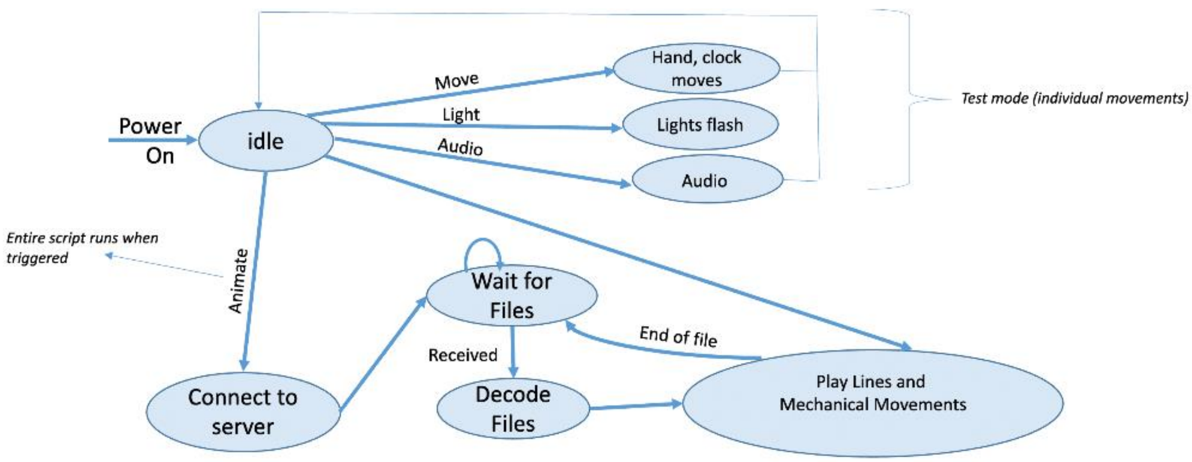

Figure 6: Software Subsystem Flow Diagram.

Power Sub-System

Task Lead: Sophie Ayoung-Chee

The power subsystem will take 120 Volts AC as an input from a wall outlet and convert it to 5 volts DC using an AC to DC wall adapter. The power goes through the wall adapter to a user input switch, through a fuse, then routes the 5V to the controller. Based on our power budget, we will be using a 5V wall adapter that has a 15W power capacity with a barrel jack output. The PCB design was completed for the power system. There is an LED in parallel with the circuit to determine if power is being delivered.

Subsystem	Component	Max Voltage (V)	Max Current (A)	Quantity	Worst case power drawn (W)
Software	Raspberry Pi Zero W	5	0.5	1	2.5
Audio	Speaker	5	0.8	1	4
Lights	LEDs	2	0.02	2	0.04
Mechanical	Servo	5	0.5	1	2.5
	Motor	5	0.5	1	2.5
Total					**11.54**

Table 2: Power Budget used to determine necessary power drawn.

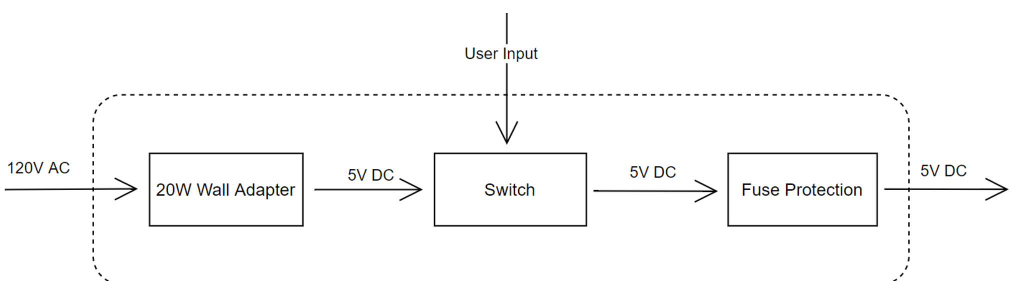

Figure 7: Power Subsystem Block Diagram.

Mechanical Sub-System

Task Lead: Eugene Min

The mechanical subsystem consists of two motors. First is a DC motor which is attached to the clock hands and will cause it to spin 360+ degrees continuously. This motor will be receiving a variable voltage from a potentiometer that the user can adjust to vary the RPM of the clock hands. The second motor is a servo which will move the whole clock. This motion is driven by a PWM signal and will swing the clock clockwise away from Willy's body 100 degrees and back counterclockwise to the clock's initial position.

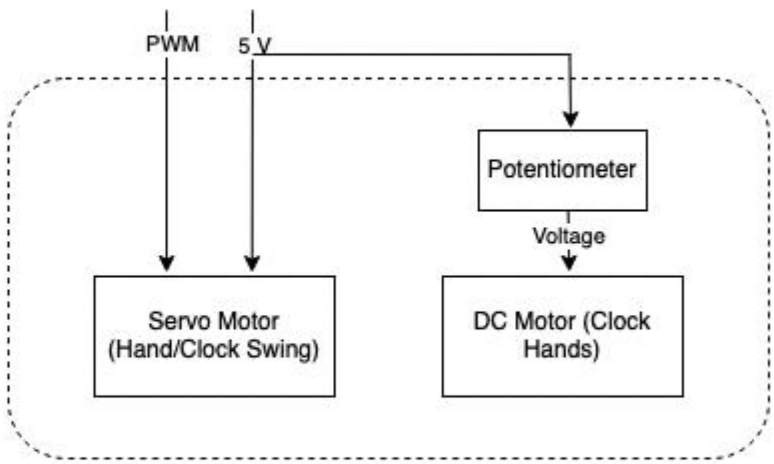

Figure 8: Mechanical Subsystem Block Diagram.

Audio Sub-System

Task Lead: Eugene Min

To ensure Willy has crisp and clear audio, the speaker is driven by the Micro USB port on the Raspberry Pi. This audio file is played using the "mpg321" command. As shown in Figure 9, the USB speaker is connected to a USB to Micro USB converter as the Raspberry Pi only takes in Micro USB and the speaker is located on the side of the box.

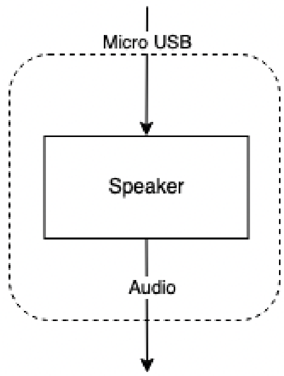

Figure 9: Audio Subsystem Block Diagram.

Light Sub-System

Task Lead: Allison Sawyer

The light subsystem consists of two red LEDs which serve as Willy's eyes. The LEDs are tied to the same digital signal from the Raspberry Pi's GPIO pins which control if the lights are on or off. The GPIO pins also serve a 3.3v power supply for the LEDs. This power supply voltage is stepped down slightly by a resistor before it reaches the LEDs.

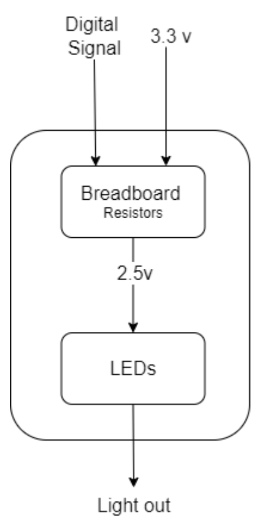

Figure 10: Light Subsystem Block Diagram.

Mechanical Design

The mechanical design primarily addresses the movement part of Willy and there are two different movements. The first is the clock swinging away and back to Willy which is controlled by a servo motor. As shown in the image below the clock, which is taped to the servo temporarily as a demo, rotates. We are able to rotate the servo 180 degrees clockwise and 180 degrees counterclockwise which meets the requirement of being able to rotate at least 100 degrees.

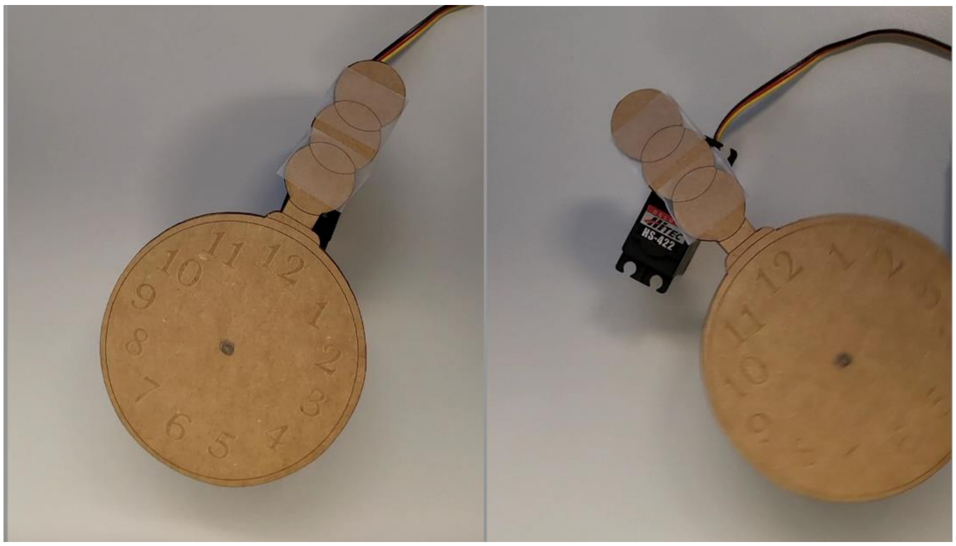

Figure 11: First draft of clock on cardboard to demonstrate clock motion.

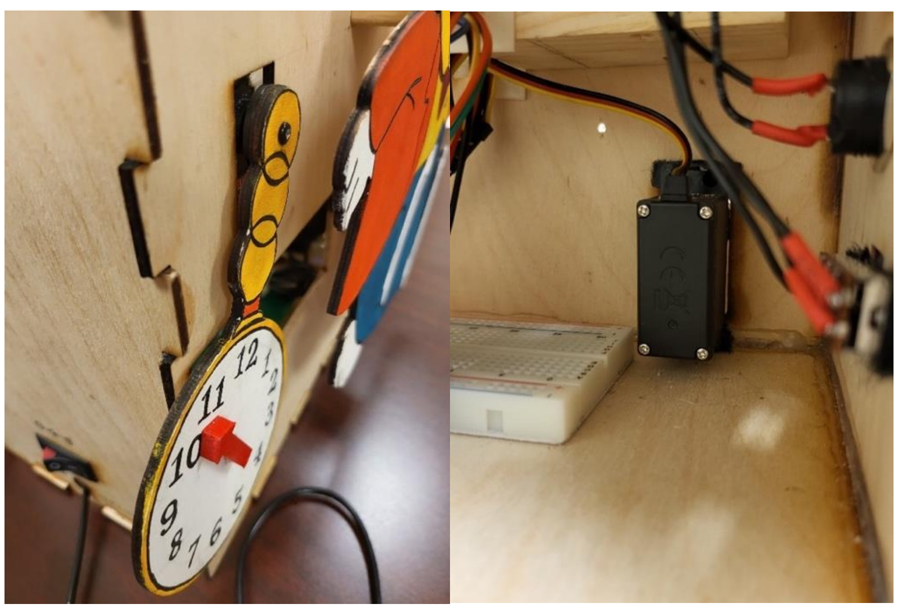

Figure 12: Clock integrated with servo motor and box in final design.

The second movement is the rotation of the clock hands. DC motors rotate at around 6000 RPM and in order to get it down to around 50 – 250 RPM, there needs to be a gearbox to slow the rotation down. In our case, we are using a DC motor that's attached to a 1:48 gearbox to get the desired RPM and this can further be controlled by a potentiometer, which meets our analog circuit requirement, to dial down the voltage. As the voltage increases, the RPM increases as shown in the table below.

Voltage	**Number of Seconds per 10 Rotations**	**Approximate RPM**
1 V	14	43
2 V	6.5	92
3 V	4.5	133
4 V	2.9	206
4.5 V	2.5	240

Table 3: Observed and Calculated RPM of DC Motor.

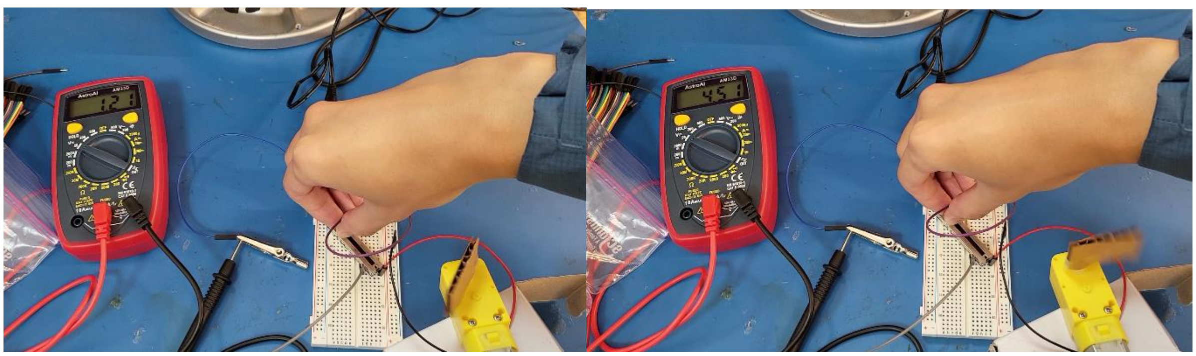

Figure 13: Voltage Reading (left) and Rotation of DC Motor (bottom right) Adjusted by Slider Potentiometer (center).

In the final implementation, the potentiometer could adjust the RPM of the clock hands anywhere between 75 – 135 RPM. The friction of the clock hands against the box as well as the Pi providing power to the servo, speakers, LEDs, and DC motor concurrently limited the maximum RPM of the clock hands.

Figure 14: The slider potentiometer attached to the side of the box (left) adjusts the speed of the DC motor (center) which rotates the clock hands (right).

Electrical Design

The PCB design was completed for the power system. The power system includes a fuse in series with a switch that controls the voltage. Header pin holes (JP2) are used in place of the switch, as the switch will later be soldered into the holes. The fuse part (F1) is a custom designed part to fit our selected fuse holder size, because the EAGLE libraries did not include a fuse symbol of our size. There is an LED in parallel with the circuit to determine if power is being delivered, connected to a 100 ohm resistor. The header pins for power and ground (JP1) control the power output of the system.

The pads for the fuse were too small when the PCB was produced, so we drilled holes next to the pads to insert the feet of the fuse into. We soldered into the drilled holes and made sure that it sat on top of the pad as well to keep the electrical connection. We also increased the size of the mounting holes by drilling into them.

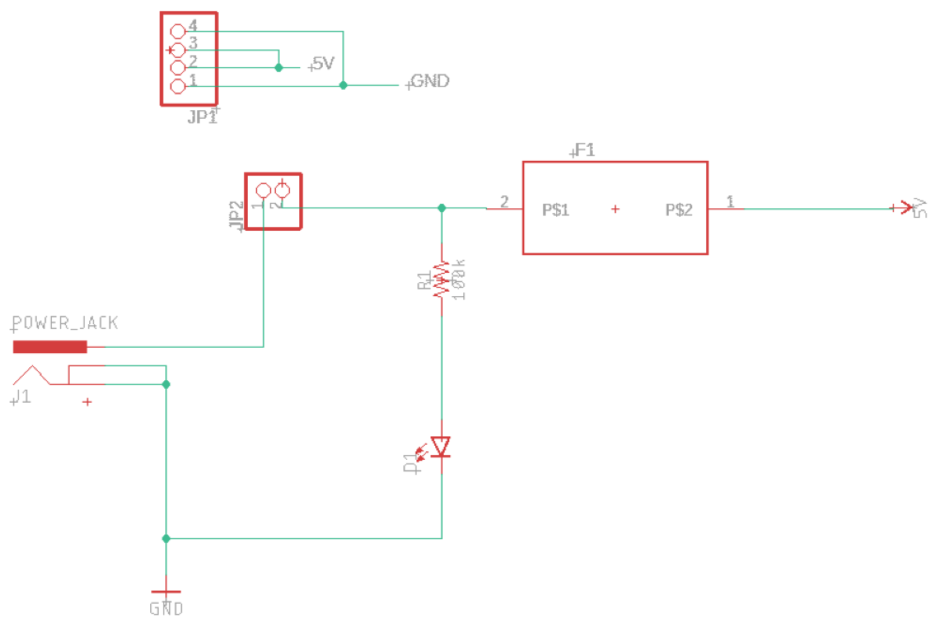

Figure 15: EAGLE schematic of the Power PCB design showing the power flow circuit as well as an indicator LED in parallel. The fuse part was custom designed in EAGLE to fit the size of our specific part.

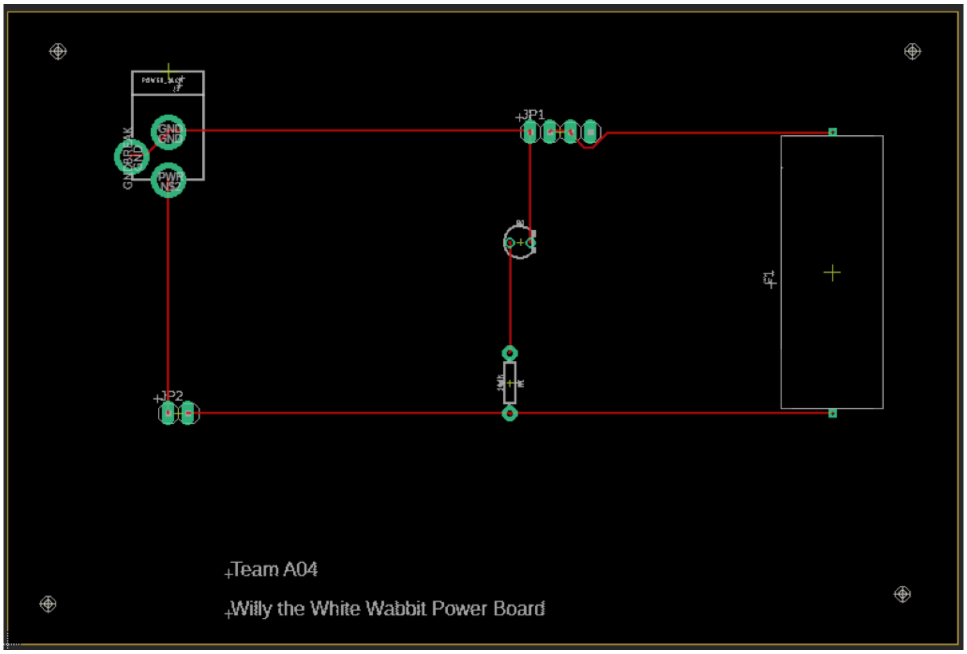

Figure 16: EAGLE PCB board design that houses the power system including the fuse, switch, and power input.

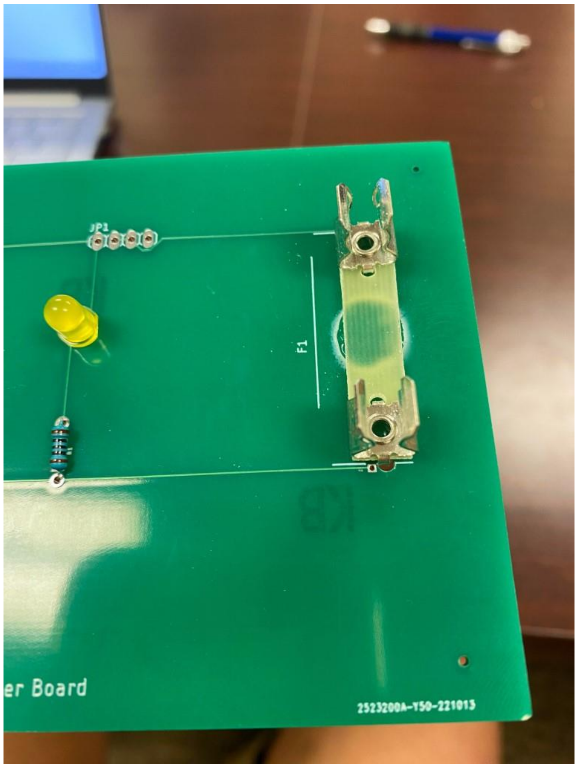

Figure 17: The custom fuse part that is attached to the PCB is inserted into holes that we drilled and then soldered to the pads.

Software Simulation

The connection and file-sending capabilities between the team computer and the Raspberry Pi were tested. Following a successful connection, data (i.e., simulation files) could be transferred from the team computer to the raspberry pi through a command-line utility that allows the user to securely copy files and directories between two locations. The files were successfully saved in the appropriate folder in the Raspberry Pi.

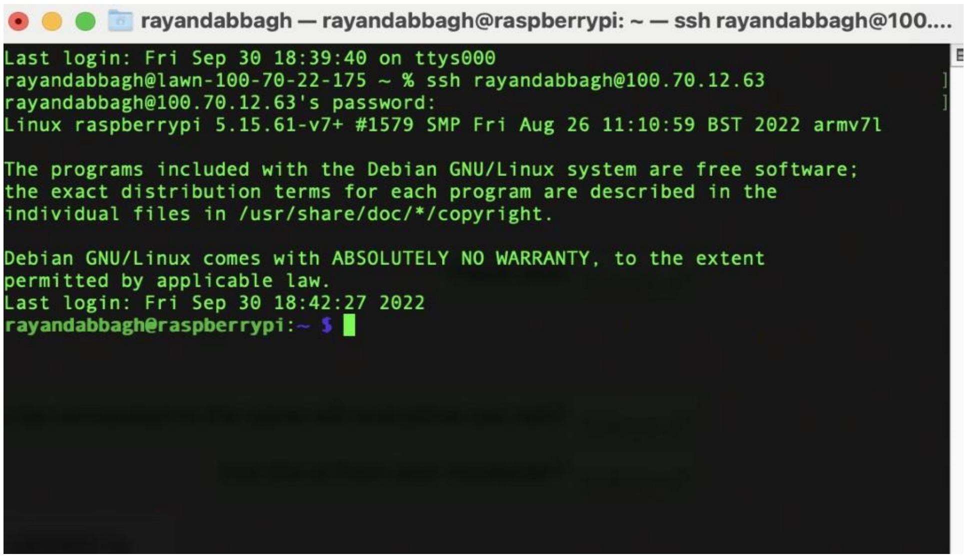

Figure 18: Connection of Team Computer and Raspberry Pi.

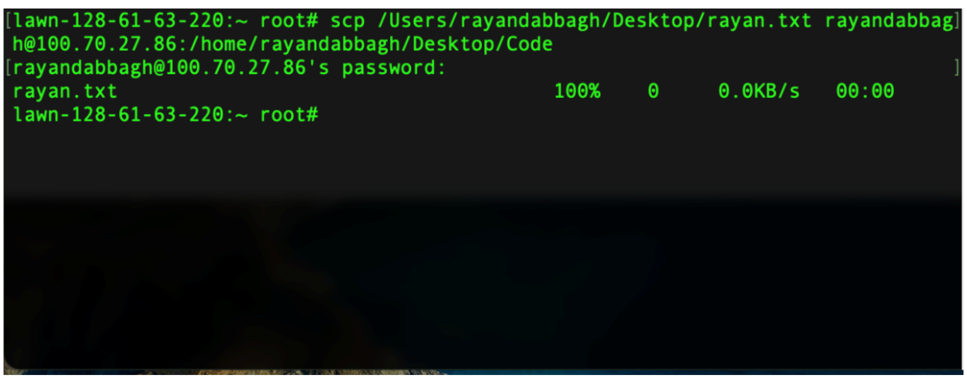

Figure 19: SCP (Secure copy protocol) command demonstration.

Simulation Results

Through the Raspberry Pi, we sent a digital signal to blink two LEDs using GPIO (general purpose input output) pin 27 and pin 22. The program being used on the Pi sets LED 1 on, pauses for one second, then turns off LED 1 and turns on LED 2, pauses for one second again, and then loops back to LED 1. This runs until terminated. The voltage sent from the Pi was measured at 3.3 volts, which is the expected voltage input to the red LEDs. We used 330 ohm resistors to reduce the input voltage on both LEDs, which reduced the voltage and made the LEDs not as bright as desired for the eyes of Willy the White Rabbit and will need to be adjusted.

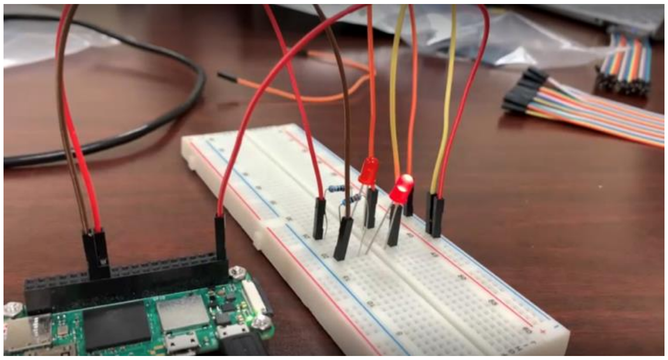

Figure 20: Set up of Pi based LED simulation.

Another simulation includes using pushbuttons to turn on LEDs. The pushbuttons act as an intermediary, allowing the signal from the Raspberry Pi to propagate to the LEDs when the pushbutton is pressed. We introduced two new variables, BS1 and BS2, in order to indicate the state of the LEDs (A “BS1” = False means the LED1 is off, a “BS1” = True means the LED is on). This concept allows us to determine whether we should turn the LED on or off when the button is pushed. In this test, we demonstrate that both LEDs can be controlled independently, and both can be turned on or off at the same time.

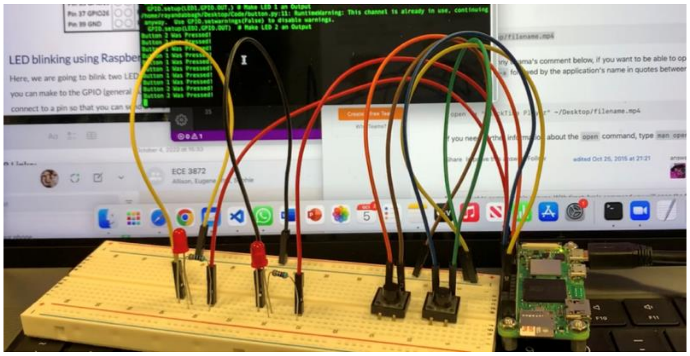

Figure 21: Set up of the Pi based pushbutton simulation.

The last simulation included the production of a PWM signal from a pin to control the servo motor. There is only one user accessible PWM pin on all Raspberry Pi models and it’s GPIO18. Using this pin as well as the PI’s 5 V and GND, we have been able to rotate the servo 180 degrees counterclockwise and then back clockwise which meets the requirement for Willy’s clock (See Fig. 12).

Robot Design

The design we created for Willy created some complications when it came to deciding how to encapsulate and protect our internal hardware. Our first design gave him a base to stand on that would serve as our box for components, however we realized that the components such as the motor for the arm, and the LED eyes would have to be exposed.

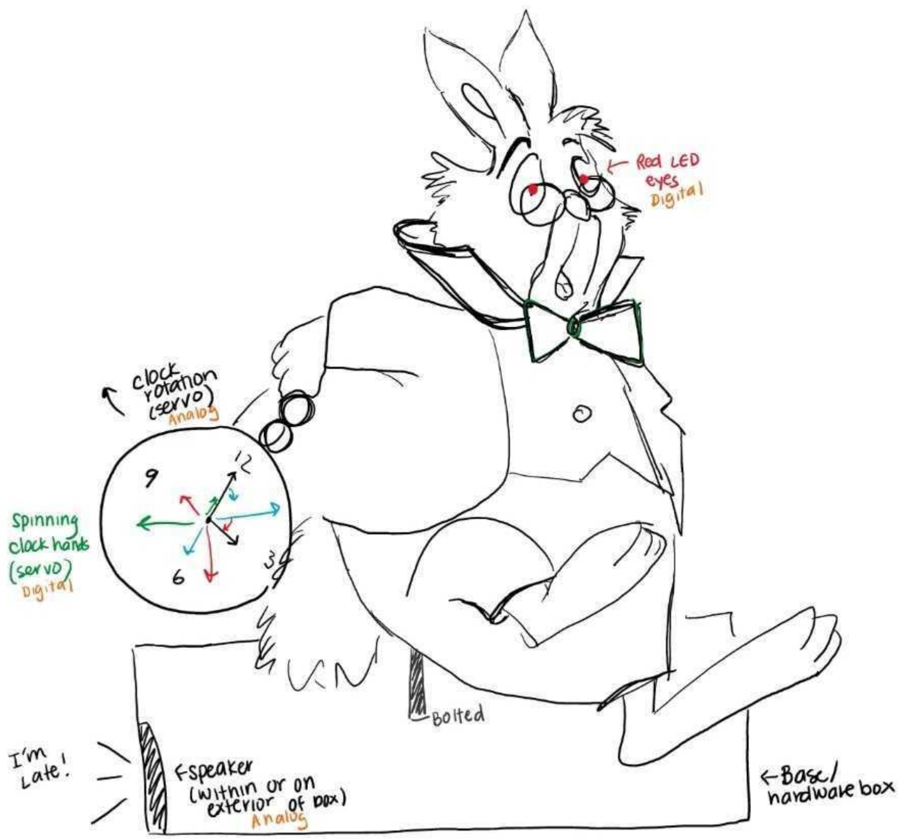

Figure 22: Our preliminary design of the White Rabbit. The design shows him being bolted on top of the box, which was changed because our hardware interacts with the back of the rabbit.

We immediately switched to a box that would stand up behind him so that no components would be exposed, and the weight of the robot could be better distributed. We also decided that there needs to be a shelf in the middle of the box to support the PCB and any connections made to it. This allows the mechanical components' weight to be better supported inside, and to prevent stress on those components.

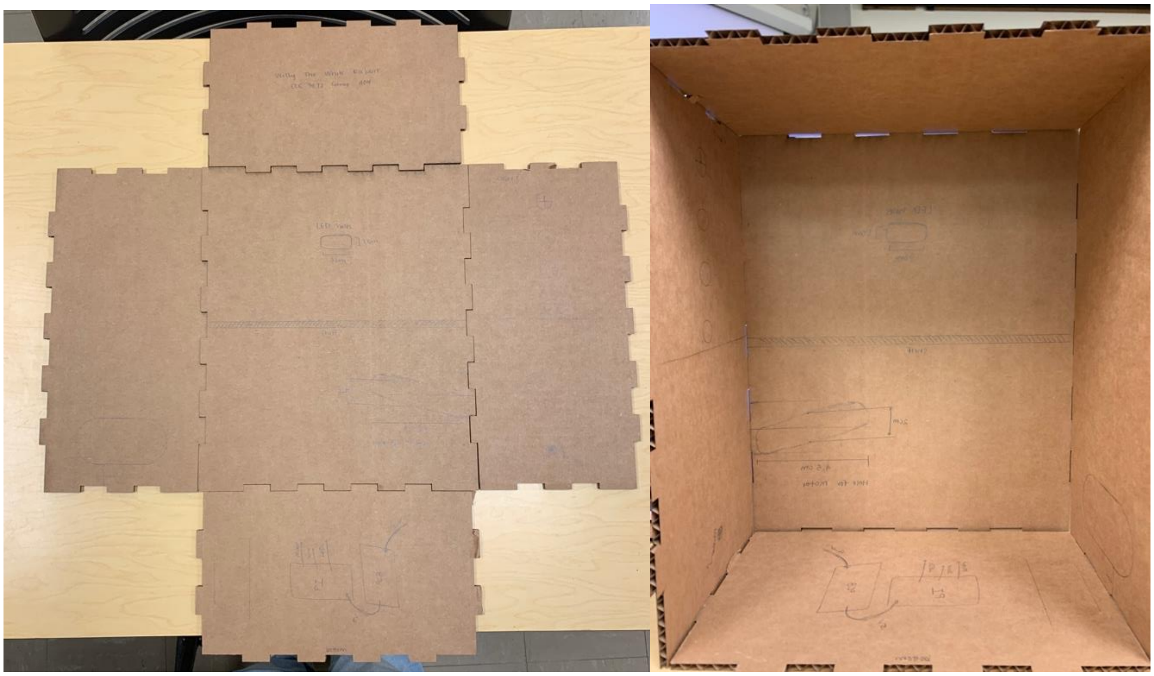

Figure 23: Our first box prototype on cardboard, simple box with finger joints created based on the approximate size we wanted Willy to be. This is where we sketched out where components would rest, and where holes for external components should be placed.

Figure 24: Second box prototype on cardboard, after creating holes and allowances for components. (Of note: the back panel is upside down in this photo.)

Figure 25: First box prototype on wood, many of these panels are used in the final project. Notable: DC motor to move behind the clock has been enlarged significantly.

Figure 26: First prototype of Willy and his clock on wood.

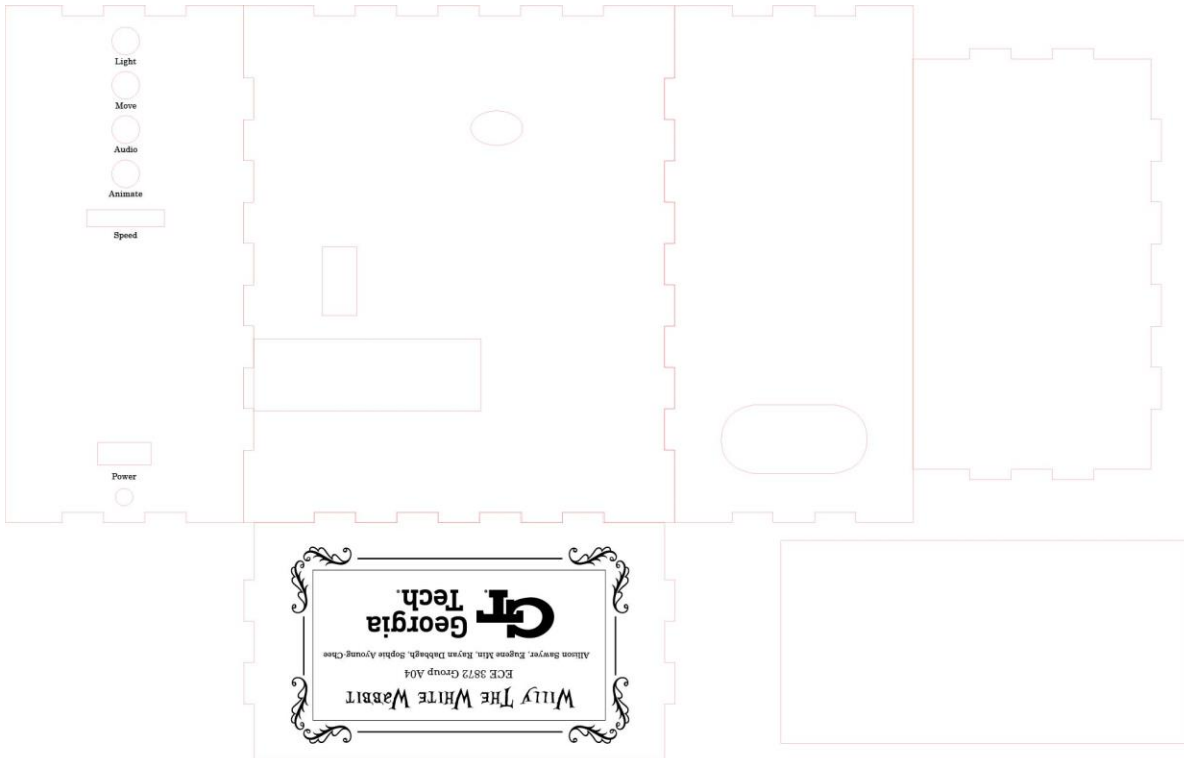

Figure 27: Final Adobe Illustrator drawing of the box. Featuring our project label and holes for externally facing components. The blank rectangle (bottom left) is a shelf that will be glued in.

Figure 28: Side view and interior view of our final box. Notable changes: there are wooden squares glued in that hold magnets that secure the door closed, and the DC motor is not using the space originally designed for it (long rectangular hole under shelf), and instead on a shelf in the upper right-hand corner.

Integration

Software High-Risk Parts:

- The microprocessor is the most high-risk part of our design because it interfaces with each sub-system and is used to control the entire device. To reduce and prevent failure, we performed multiple integration steps. Our first simulation used a breadboard connected to a power supply to test that the LEDs are functional. We then integrated pushbuttons to ensure our software could properly interface with pushbuttons. Then, in order to test Willy's many functions simultaneously, we integrated our microprocessor, a RaspberryPi Zero 2, pushbuttons, speaker, and motors on the breadboard. Our analog components, such as the servo and DC motor, were added as the next test. The last test integrated the PCB, as we connected the power supply directly to the pi, as well as power the analog circuit.
- Over-drawing power and overheating is our second large high-risk problem. After we connected the pi to a power supply instead of a computer, it began to overhear quickly. Within a matter of five minutes, it becomes very hot to the touch. Additionally, performing multiple tests with the pushbuttons, or turning on the DC motor with the potentiometer, begins to draw too much power, causing overheating quickly and creates performance errors.

Hardware High-Risk Parts:

- The motors also pose a substantial risk. They create the bulk of the visual interest of the robot, which makes them highly important to our design. If the motors fail, Willy’s watch will not rotate out, and the clock hand will not spin.
- Five PCBs will be available for testing and revision of our design. We want to minimize failure while soldering our pieces together because we only have this limited number of tests available. The drill holes in the PCB were too small for fusing which created more risk in destroying our limited supply of PCBs, we had to manually drill out the holes to be bigger, which runs the risk of splitting or breaking the board.

Mechanical High-Risk Parts:

- The wood of the box and the acrylic used for the door is 1/8th of an inch thick which made it difficult to drill holes and add screws without any splitting or breaking.
- The hand of the clock creates friction with the box due to the lack of space between the part and the wood. This wears down the box over time, thus shortening the life span of the robot.

Repository Management:

- We have a GitHub repository of our python codebase with the code for the different pushbutton functionalities. The GitHub also includes our software code.
  - https://github.gatech.edu/rdabbagh3/rabbit_team04_repo.git
- We also share files in our central Microsoft Teams, organized by project phase.

Constraints, Alternatives, and Trade-offs

As we progressed through the project, we had to make several trade-offs based on time, efficiency, design constraints, and reliability. Here we identify the large design choices we made, and analyze the effect it had on the project, as well as determine if there was an alternative solution.

1. Changing the position of the DC motor and clock hand:
   Our original plan was to have the clock hand in the watch spinning via a DC motor. This DC motor is connected to the power supply directly and fulfilled our requirement of having an analog circuit. However, after we attached all the parts together, we noticed that the DC motor was moving too much and limiting the swing of the pocket watch because its heavy. We attempted to secure it via zip tie to the shelf above it, so it could swing like a pendulum but have support for its weight. Additionally, we could not successfully attach a zip tie, and there wasn't room on the shelf to drill for the zip tie to loop through. Our end solution was to move the DC motor up behind Willy, we added a shelf for the DC motor to rest on and secured the clock hand to the motor through the box. This created a visual trade off, as the clock hand was no longer where we wanted, and added two unsightly screws to the front of the box. This was ultimately our best choice because we needed to salvage our analog circuit.

2. Using a breadboard:
   Our design constraints indicated that we should avoid breadboards and opt for only electrical connections on a PCB because its more secure, professional, etc. However, we did not know exactly what connections we needed on the PCB by the deadline to have one fabricated. We could have opted to solder the connections to a different board, but due to time restrictions and changing designs, we did not have time for this step.

3. Restrictions of the Pi regarding power and performance:
   Due to the total power being consumed by the system when everything is active at the same time, if the sliding potentiometer is adjusted to increase the clock hand speed, it frequently causes other functions to activate even though no button was pressed. This is because the Pi is overcompensating for the power being drawn. Additionally, the Pi quickly gets hot, which also causes malfunctions and delays. A big proponent of this is the speaker; we wanted to use a micro-USB speaker because it has much better audio than its alternatives, but it had to be connected to the Pi rather than directly to the power supply. A long-term solution might require a different microprocessor.

Conclusion

Over the course of 11 weeks, our team was able to successfully create and design Willy the White Wabbit. Our project was divided into smaller subsystems that allowed for us to efficiently delegate tasks and fulfill our responsibilities. There were issues within each of our subsystems that we were able to account for in an organized manner because of how we divided the work. Mechanically, we had issues with turning our ideas into a reality as we learned the constraints of the parts we wanted to use. Our two-motor design (one DC motor, one servo) was still implemented, but the placement of the motors had to change due to weight constraints on the parts. The PCB for the power subsystem design became difficult because of the custom fuse part that we had to create, which did not come out as expected when the PCB was printed. We were able to salvage the PCB using a drill and some solder work. The software had many issues that required debugging, in which we were unsure how the code would work with our specific hardware. We were able to resolve these issues through testing and software simulations.

If we were to do this project again, there are some things that we would have done differently. Our initial design included a wooden clock piece attached to a servo that then had a DC motor attached to the clock. This was something we realized was unfeasible only a week before the demonstration. If we had created the schedule and test plan to be more simulation- and integration-based, especially in the beginning of the semester, then this and other issues could have been avoided. There were also some issues with the USB speaker working while the potentiometer was on, possibly due to power constraints. This is also something that could have been caught had we integrated simulation and integration testing earlier in the schedule.

In the above subsections, subsystem designs, constraints and considerations, and the distribution of work is outlined. Overall, our team was able to effectively distribute work with our schedule, which we followed very closely at the beginning of the project. This helped us a lot with the amount of work that we had near the end of the project where we were able to focus mostly on the smaller additions.

Acknowledgements

Alice in wonderland font courtesy of Marco T.L., free for non-commercial use.
www.fontspace.com/marcotl

Decorative borders courtesy of Takeshi Ishikawa, free for non-commercial use.
https://www.vecteezy.com/vector-art/182064-a-set-of-assorted-vintage-frames

Audio clipped from the Disney animated film Alice in Wonderland (1951).

Schedule

The schedule for completion of the project is shown in Table 3 where the task lead is indicated on each task.

Task	1	2	3	4	5	6	7	8	9	10	11
Proposal											
Brainstorm	X										
Master Schedule Planning		Allison									
Project Sketch		Sophie									
Software State Flow Diagram		Rayan									
Hardware Block Diagram		Eugene									
Proposal Slides		Allison									
PDR											
Ideate customer needs into design		Allison									
Design Power System		Sophie	Sophie								
Power - Compare parts			Sophie	Sophie							
Power - Part Selection					Sophie						
Design Software System		Rayan	Rayan								
Software - Part Selection			Rayan	Rayan							
Software - Set up					Rayan						
Design Audio Subsystem		Eugene	Eugene								
Audio - Compare Parts			Eugene	Eugene	Eugene						
Audio - Part Selection					Eugene						
Design Light Subsystem		Allison	Allison								
Lights - Compare Parts			Allison	Allison							
Light - Part Selection					Allison						
Design Mechanical Subsystem		Eugene	Eugene								
Mechanical - Compare Parts			Eugene	Eugene							
Mechanical - Part Selection					Eugene						
Design/Plan Box				Sophie	Sophie	Sophie					
PDR Deliverables					Allison						

Task	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
CDR											
Build Power Subsystem						Sophie					
Test Power Subsystem							Sophie				
Build Software Subsystem					Rayan						
Test Software Subsystem						Rayan					
Build Audio Subsystem						Eugene					
Test Audio Subsystem							Eugene				
Build Light Subsystem						Allison					
Test Light Subsystem							Allison				
Build Mechanical Subsystem							Eugene				
Test Mechanical Subsystem								Eugene			
Integrate Power & Software Subsystems									Sophie		
Integrate Power & Audio, Lights, Mechanical									Sophie		
Integrate Software & Audio, Lights, Mechanical								Rayan			
Test Software Tests for Audio, Lights, Mechanical									Rayan		
Connect Software to Pushbuttons for Tests									Eugene		
Test Push Buttons Perform Tests									Eugene		
Laser-cut Rabbit prototype								Sophie			
Laser-cut Box Prototype							Allison				
Laser-cut Box Prototype 2								Allison			
Laser-cut Final Box Wood										Allison	
Laser cut Rabbit & Clock on wood, paint if needed										Sophie	
Assemble Box										Allison	
Test Robot once inside box										Eugene	
System Test (Analysis, Inspection, Dem., Test)										Allison	
Final Inspection and Demonstration											
Final Reports & Submission											Allison
Final System Test (Test Report)											Allison
Final Demo Video											Sophie

Table 4: Schedule to complete Willy the White Rabbit.