EGR 304 - Team 106

Final Report

Not Creepy Elf

Arizona State University
Shawn Jordan
Adriana Juarez, Sai Srinivas Tatwik Meesala,
Noah Blevins, Ian Mansfield

December 06, 2021

Table of Contents

Problem Definition

Problem Statement

The purpose of our project is to create an electronic decoration piece that will help entertain guests during the Halloween and/or Christmas holidays. Building the product will satisfy the user's need for a scary decoration on the Halloween holiday, while at the same time satisfying their need for a traditional Elf on The Shelf on the Christmas holiday. The Not Creepy Elf will bring guests fear, laughter, and excitement. It is perfect for the holidays as well as for normal usage throughout the year.

The Not Creepy Elf is an electronic holiday decoration that comes with two settings that could be used for both Halloween as well as Christmas. The product may come with some positive and negative effects. The device will encourage laughter and immersion during both holidays. It may also spook people during both holidays, which is the goal of the decoration during Halloween but not during Christmas.

See Appendix A. for the instructor-defined requirements that must be met by the product overall.

Performance Specifications

The performance specifications for the product are shown in Table 1. Performance specifications with actual values can be seen in Appendix B.

Table 1. Performance Specifications for Not Creepy Elf

#	Metric	Unit
1	Range of Detection	Meters
2	Weight	Kg
3	Range of Motion	Degrees
4	Variation of Voltage Input	Volts
5	Height	cm
6	Water resistance	IP
7	Proper Acceleration	Degrees
8	Brightness	Lumens/Watt

Design Concept

Visual Design Representation

A computer-generated sketch with the device's components is shown in Figure 1 (The power supply will be stored behind all other components for the user's ease of replacing batteries when necessary). An overall CAD representation of the Not Creepy Elf is shown in Figure 2.

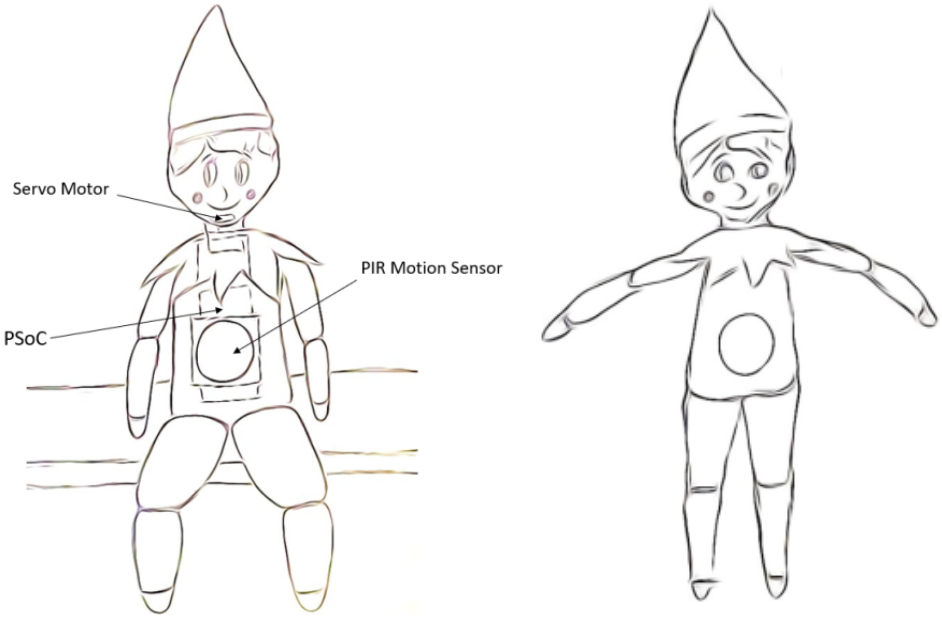


Figure 1. Sketch of The Not Creepy Elf Design

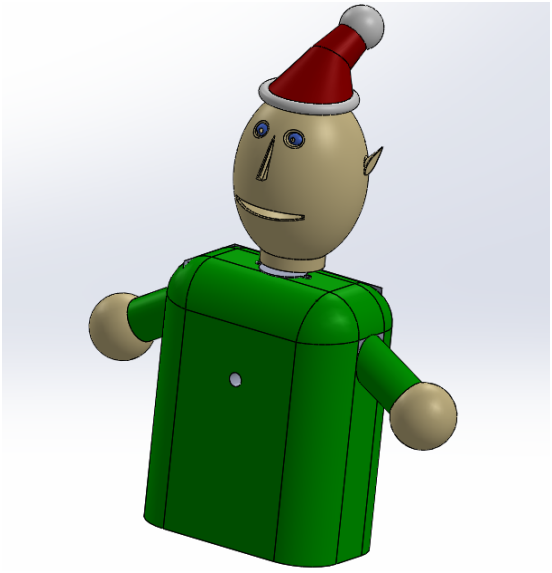


Figure 2. CAD model of Not Creepy Elf

Visual Story Representation

Below is a brief visual story-based representation of how the Elf will function during a holiday party (See Figure 3). The storyboard is annotated at the top of each section.

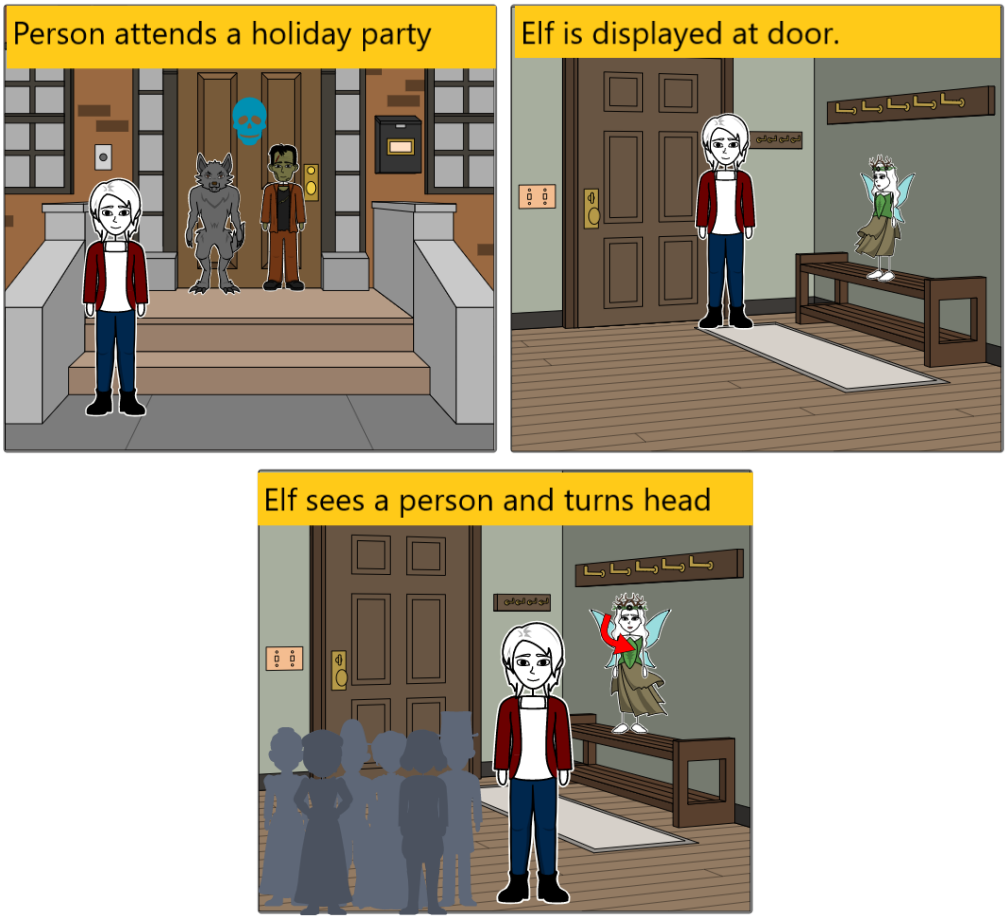


Figure 3. Story-Based Representation of The Not Creepy Elf Function

Electrical Block Diagram

The block diagram for the Not Creepy Elf is shown in Figure 4. This diagram represents all the major components and subsystems of the product, including how they will communicate with one another and where they are connected overall. For further detail on the selected major components and how the product's power will be budgeted, see Appendix C and Appendix D, respectively.

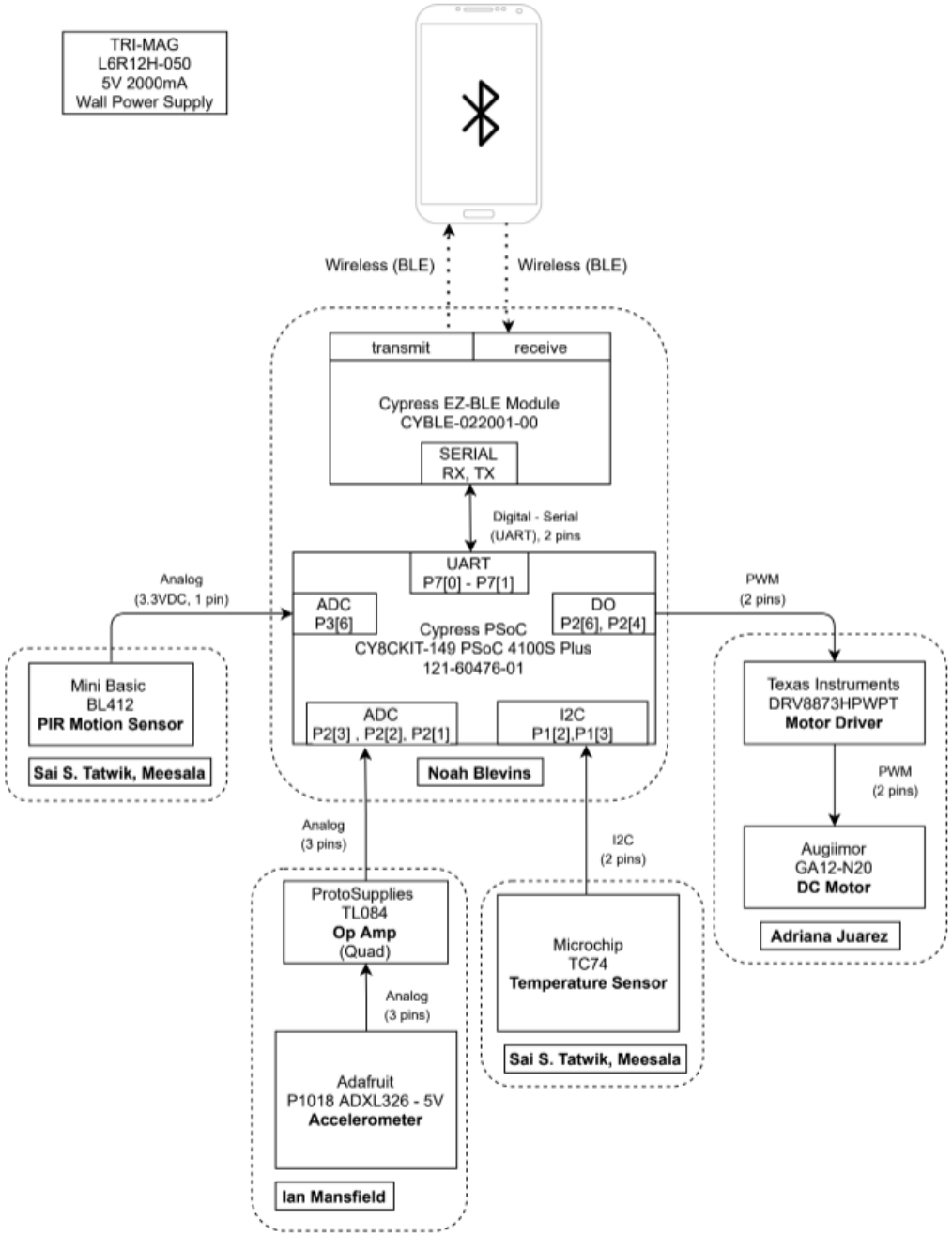


Figure 4. Not Creepy Elf Electrical Block Diagram

Progress, Lessons Learned, and Version 2.0 +

Project Photos

Figures 5 - 8 represent each team member's printed circuit board (PCB) design. Both the front (left image) and back (right image) views are displayed. As shown, several modifications outside of the PCB design were necessary for the motor driver and op-amp functionality of the system, including many fly wires and external components. See Appendix F for a more detailed outline of the PCB.

The images are organized as follows:

- Adriana Juarez's Final PCB (See Figure 5)
- Noah Blevins' Final PCB (See Figure 6)
- Sai Srinivas Tatwik Meesala's Final PCB (See Figure 7)
- Ian Mansfield's Final PCB (See Figure 8)

See Appendix E. for a bill of materials containing all parts shown in each final design.

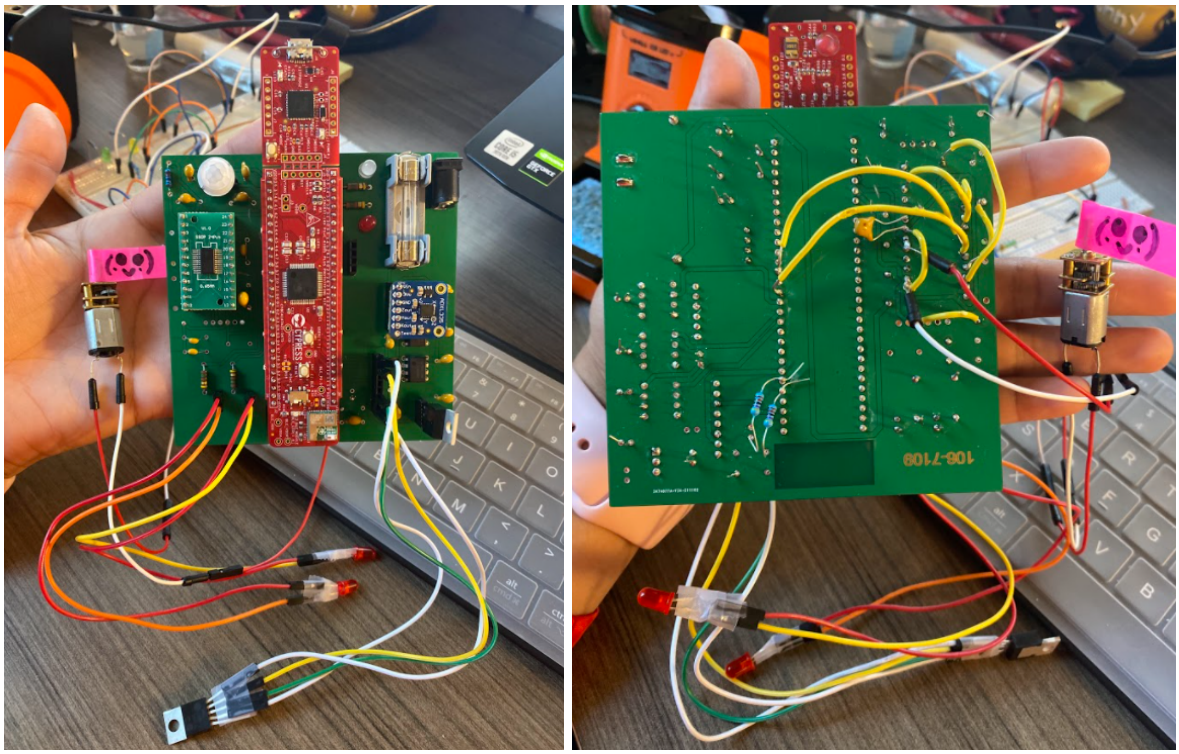


Figure 5. Front and Back View of Adriana Juarez's Final PCB (106-7109)

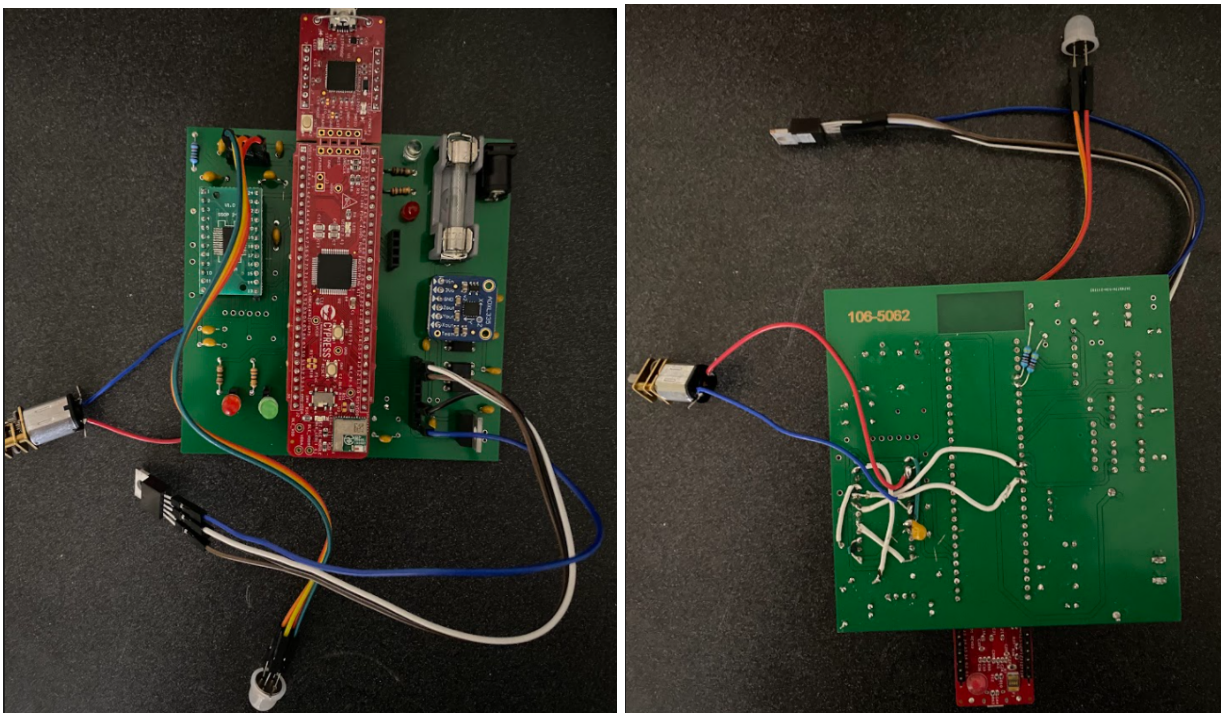


Figure 6. Front and Back View of Noah Blevins' Final PCB

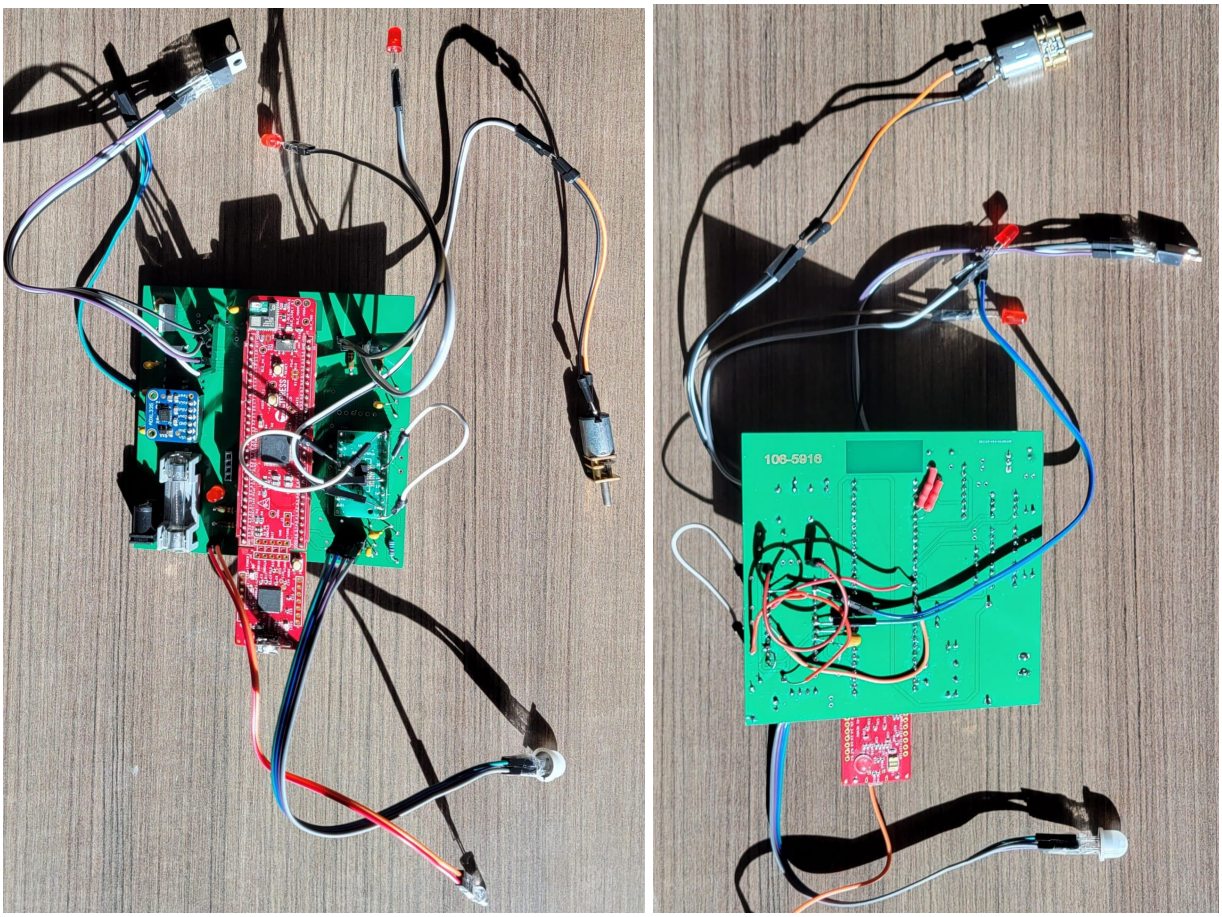


Figure 7. Front and Back View of Sai Srinivas Tatwik Meesala's Final PCB (106-5916)

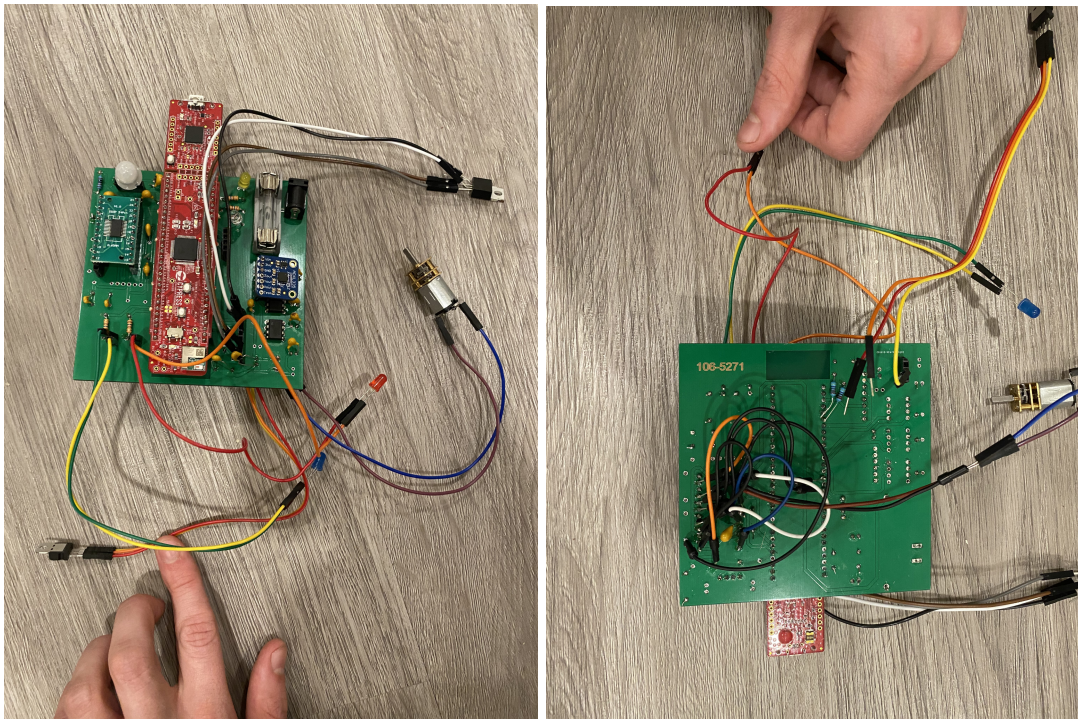


Figure 8. Front and Back View of Ian Mansfield's Final PCB (106-5271)

Hardware Design

Several modifications were made to the hardware design due to time constraints and component compatibility issues.

The following modifications were made in Team 106's Hardware Design:

- Changed motor driver pins on the microcontroller (see Figure 9).
- Changed the RFID Reader to a temperature sensor (see Figure 10).
- Changed from a stepper motor to a DC motor (see Figure 11).
- Changed the LED resistor values to work with a 3.3V power rail (see Figure 12).
- Changed to a voltage follower circuit for the operational amplifiers (see Figure 13).

See Appendix J. for the full, overall image of the schematic.

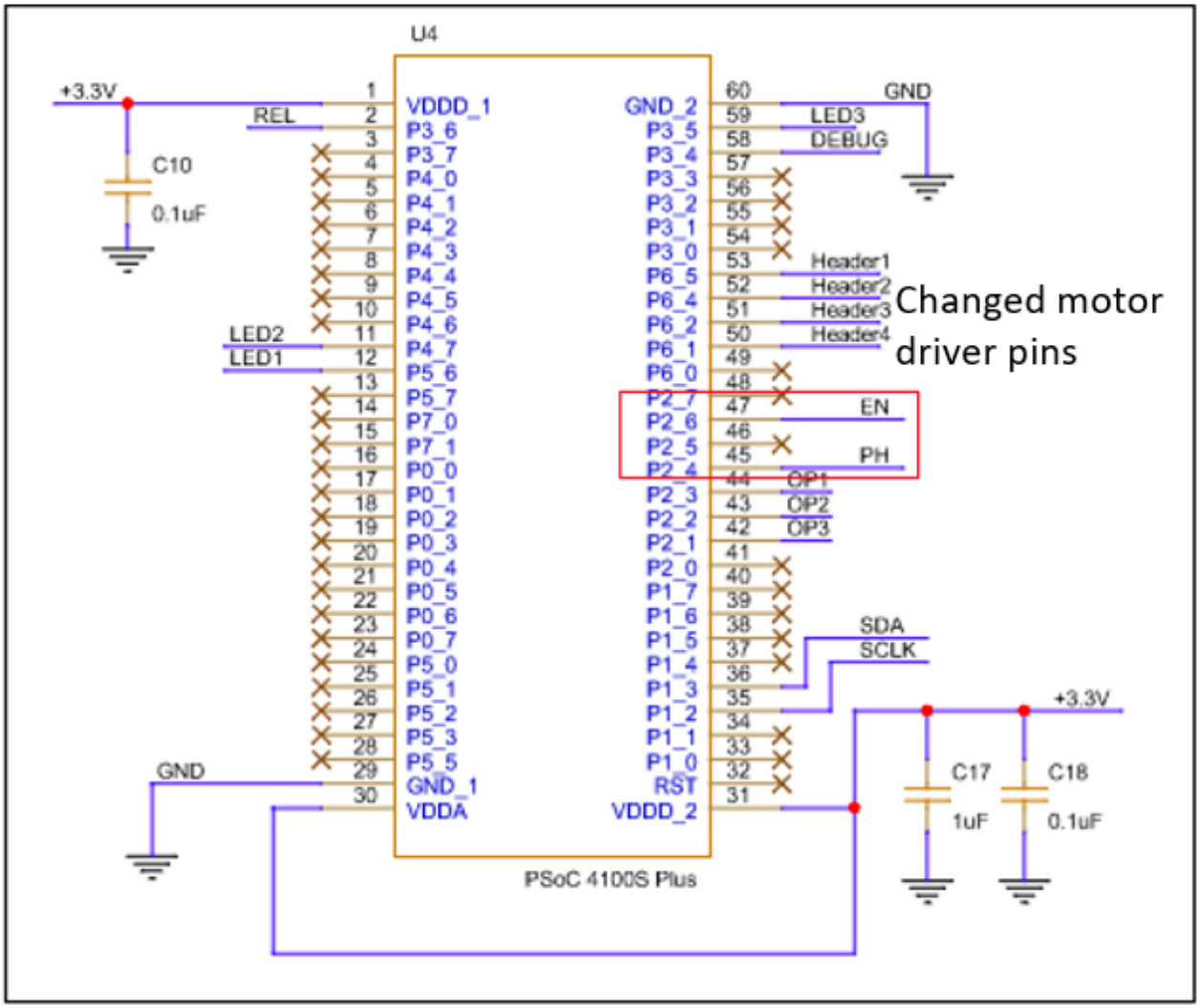


Figure 9. Microcontroller (PSoC 4100S Plus) Updated Schematic

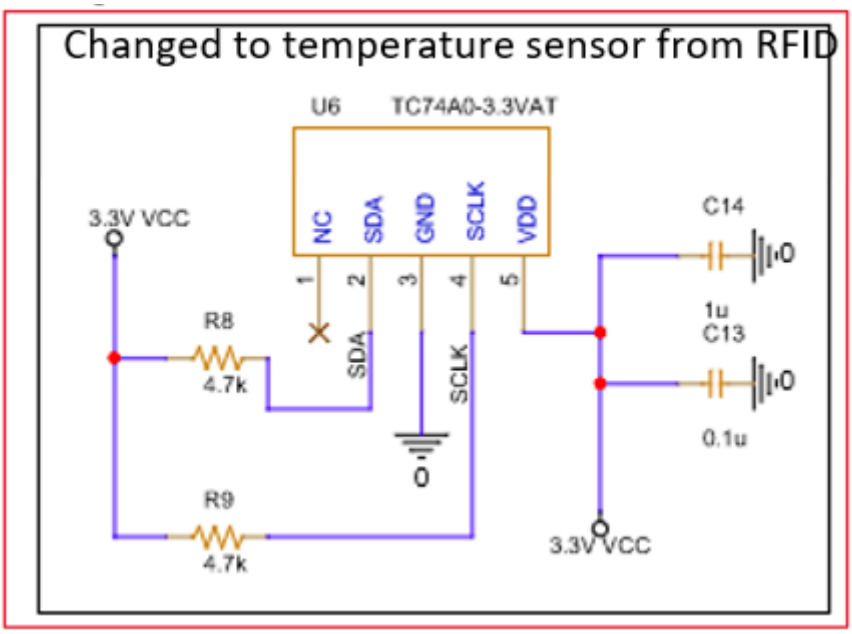


Figure 10. Serial Sensor Updated Schematic

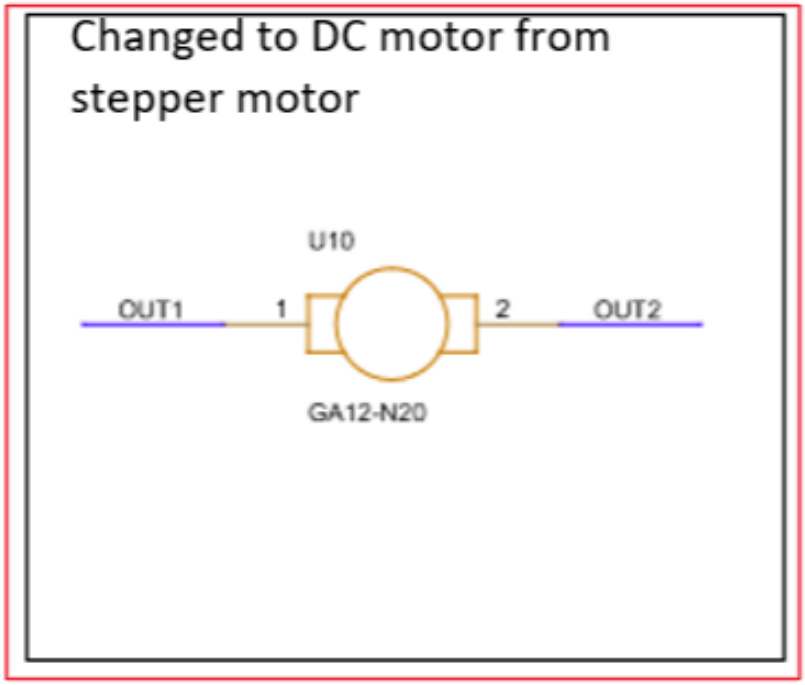


Figure 11. Motor Updated Schematic

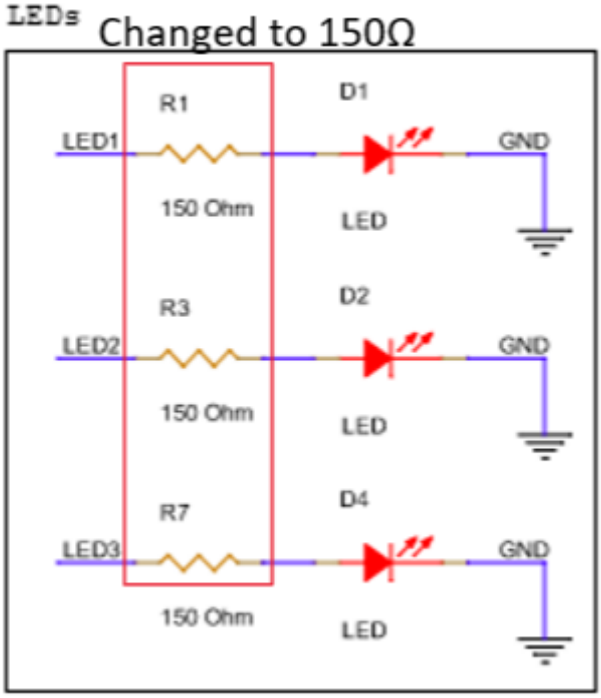


Figure 12. LED Resistor Updated Values

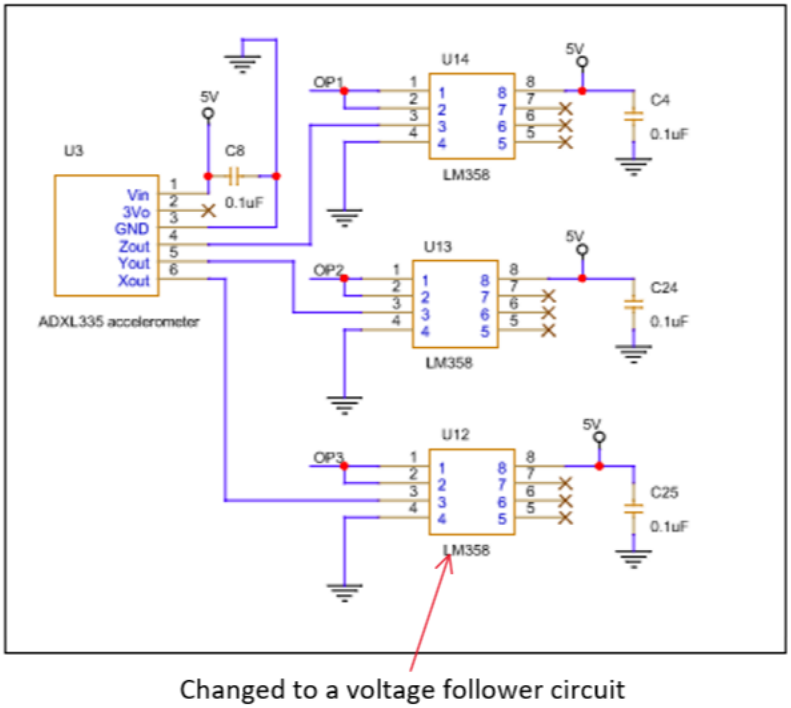


Figure 13. Accelerometer Updated Schematic

1. In the first design, the motor had a stepper motor that allowed for a motion to be measured in degrees. The motor was changed from a stepper motor to a DC motor. The motor driver is unable to support a four coil stepper motor, as there are two sets of output pins that are connected.
2. The pins that connect the PSOC to the motor driver were changed to ensure that the motor works correctly; the pins are GPIO pins. Pins for the motor driver were changed to 2.4 and 2.6.
3. The original design included a RFID reader. RFID was switched to a temperature sensor due to time constraints. The RFID reader was built for Arduino and was extremely difficult to port the libraries needed to make it functional.
4. The op-amp analog circuit was changed from an amplifier circuit to a voltage follower circuit because the amplifier circuit was redundant and did not work. The voltage follower circuit was the simplest replacement for the amplifier circuit.
5. LED resistor values were changed from 220Ω to 150Ω to ensure that the LEDs were able to get sufficient current.

Version 2.0: Not Creepy Elf Hardware Design

Some improvements that could be made in future iterations of a design are reinserting the RFID reader into the final project, since the team has a better understanding of how to convert libraries of code it would be possible in the future to use the RFID reader that was originally planned. This would allow for the mode to be changed with the RFID reader as well as Bluetooth. Another change would be to change the motor driver to one that has four output pins to allow the use of a stepper motor, which has greater accuracy in motion, as it is based into datasheet the steps the motor takes rather than the time in which the motor has power, which is the case with DC motors. Change the shape and form factor of the PCB to better allow for the full PCB to fit inside of a housing that is roughly the same size as a normal Elf on the Shelf. Change the PCB design to reduce flywiring and clean up the PCB making it easier to read, and identify where errors are occurring. All of the changes allow for a better design overall, creating a final project that would be closer to our original design concept.

Software Design

As with the Hardware Design of the system, several modifications were made to the software design due to time constraints and component compatibility issues. See Appendix H. and Appendix I. for the overall project code and Appendix G for the PSoC Creator 4.4 Top Design used.

1. Initially, the accelerometer was chosen to drive the motor and eye socket LEDs in the design when triggered. Due to time constraints, the programming needed to be simpler to at least show the component was able to function. Because of this, the accelerometer was used to only dim or brighten an LED depending on how much it is moved on the z-axis. The motor was not implemented in this portion of the code.
2. Initially, an RFID Reader was part of the overall design. The RFID component was replaced with a temperature sensor, which drastically changed the programming that was initially made. Though they were both using serial communication, the RFID Reader had a much more complex design.
3. Initially, the BLE module was going to be used to communicate to the microcontroller. Unfortunately, the team was not able to achieve this before the live demonstration. Because of this, the code for the BLE was simpler and did not exactly do what was intended. There was no input coming from the PSoC, but the team was able to output to an onboard LED.
4. Initially, a stepper motor was used in the product's design, which is designed with four different outputs. Because this type of motor was incompatible with the selected motor driver, a DC motor was purchased, which is designed with only two outputs. The top design as well as the code needed to be modified to work with the DC motor instead of the stepper motor.
5. Initially, the debug LED performed a blinking pattern when triggered. To reduce the number of delays within the code, the debug led was programmed to stay lit when triggered.

Version 2.0: Not Creepy Elf Software Design

If given the opportunity to create a second version of the code, the following things would be changed/implemented:

Not all of the desired features were implemented for the Elf, such as bidirectional Bluetooth control and the RFID reader. The temperature sensor would send a message via Bluetooth when the Elf got too cold. Currently, the Elf can only do unidirectional Bluetooth communication with a smartphone. The phone sets the mode of the Elf from Halloween to Christmas mode. Originally, an RFID reader controlled this and in a second version, both the phone and the RFID reader would control the setting.

Several changes can be made to improve ease of use and debugging. The map function would be declared in a separate C file. As is, it is currently declared in the "main.c" file, muddling readability. The pins to control the motor driver are currently in an inconvenient location. Jumper wires go from one side of the board to the other to form

the connection. The pins would be changed to a more convenient location. Not all of the names for the variables used are intuitive. For example, one of the variables is named test, an entirely undescriptive name. This would be an easy fix in a second version. To add to that, additional explanatory comments would enhance readability. There are also many unnecessary comments from prototype versions of the code that would be deleted to increase readability. As of now, there is minimal PuTTY functionality. PuTTY is a very useful tool when it comes to debugging. Adding in the ability to read various values, and the current mode (Halloween or Christmas) would make diagnosing problems easier.

As a miscellaneous change, the Elf's reaction to sensor input would be changed. Currently, when the Elf detects motion, the motor is activated and LEDs are turned on with varying lengths depending on the mode. A second version of the code would make the Halloween reactions more frightening. The flashes would be more dynamic, and the motor would turn the head more quickly.

Lessons Learned

Many lessons were learned regarding how to prevent and recover from mistakes.

On the hardware side of things, datasheets are a significant portion of a project and should be read through very thoroughly. It is important to test components together beforehand to verify that they work. Breadboarding is important, and it makes it much easier to fix a problem before PCBs are designed and manufactured. Once PCBs were manufactured using a program new to all team members (Cadence), team members had to learn and employ improvisation to fix several problems. Example solutions include the liberal use of jumper wires on final PCBs and changing major components to fit designs. The motor driver for the project was a surface mount, and team members had to learn alternative soldering methods to solder it onto the breakout board.

On the software side of things, team members learned how to use header files during EGR 304 class. Additionally, team members learned how to port code from one microcontroller/language to another (Arduino and C++ to PSoC and C). The team learned how to use and apply Bluetooth functionality to complete the final project.

Overall, a very important lesson learned was predicting and budgeting time for assignments so that they could be completed appropriately. However, sometimes, even when the proper time was budgeted, things didn't go as planned, and features had to be cut. Time management is key to a successful project.

In summary:

1) Datasheets are significant to understanding the functionality of components.
2) Testing things beforehand is essential to fixing mistakes early.
3) Learning how to improvise after mistakes can still ensure a working design.
4) Header files can be used to better organize code.
5) Code can be ported from one microcontroller to another.
6) PCB design can better organize projects and should be done when breadboarding a design is successful.
7) There are several alternative soldering methods for surface-mounted components.
8) Time management is important for project completion and success.
9) Learned when to cut losses and move on.
10) Bluetooth can be used in several simple and complex applications.

Recommendations for Future Students

Having experienced and encountered several issues and inconveniences during the project, there are many things the team would have done differently.

1. Read and understand all project requirements beforehand. Ask for clarification for anything you don't understand as soon as possible.
2. Don't get adventurous with the project ideas. Choose something that you can work within the span of time given.
3. Breadboard your design before anything else to verify it works. It's much easier to fix a breadboarded design than a PCB board. It will eliminate several problems you may have later on in the project.
4. If you need help from TA's, go to office hours. They will help and support you as much as they can. Try attending before everyone needs checkoffs. You may not get help otherwise.
5. Do not wait to start things last minute. The assignments will pile up along with the project. Be mindful of your time.
6. Try to start working on the final code at least a week before it's due. This will help in debugging.

Appendix

The appendices will consist of instructor-defined requirements with actual values, performance specifications with actual values, the major component selection rationale, the power budget, bill of materials, a PCB layout, the top design, and the code used to complete the final product.

Appendix A. Instructor Defined Requirements

#	Metric	Unit	Marginal Value	Ideal Value	Actual Values
0	Embedded (HW + SW) subsystem designed and used in the final device	#	1 by each team member	∞	1
1	Prototype budget ● Does not include development kits, components or PCBs from PRLTA, or components acquired for free	$	$50 / team member	< $50 / team member	$65
2	Programming language	programmed in C or C++			C
3	Power supply (not a subsystem) ● Regulated AC adapter, or unregulated AC adapter, battery, or solar panel with the regulator circuit	#	1	1	1
Microcontroller Subsystem (1 team member responsible for both #4 and #5)					
4	Cypress PSoC® microcontroller ● Must use the CY8CKIT-149 ● This counts as a subsystem	#	1	1	1
5	Bluetooth Low Energy (BLE) communications with a phone, computer, or other device ● BLE is built into the CY8CKIT-149	#	2 (1 input, 1 output)	2 (1 input, 1 output)	2
Sensor + Actuator Subsystem(s) (1 team member per sensor/actuator)					
6	Serial sensor(s) read by a microcontroller ● Must communicate with the PSoC® via a serial protocol (e.g., I^2C, SPI, UART) ● Must use custom software to read, filter (if necessary), and store data	#	1	1	1
7	Analog sensor(s) read by a microcontroller ● Must use custom-designed non-trivial amplification and/or filtering ● Values read must be used to control output(s) on the microcontroller	#	≥ 1	2	2
8	Actuator(s) controlled by a microcontroller	#	1	1	1

	● Must have a multi-pin digital or serial interface ● Must utilize complex custom software ● Control must be based on sensor input				
Additional Requirements (Graded during Live Demonstration, Individual grades based on team PCB)					
9	Functioning custom printed circuit board created in Cadence ● No commercial PCBs	#	1per team member	1	1
10	Functioning individually during final demonstration	%	50% functioning	100% functioning	75%
11	Bidirectional BLE communication during final demonstration	%	Unidirectional common.	Bidirectional common.	Unidirectional

Appendix B. Performance Specifications and Rationale

#	Metric	Unit	Marginal Value	Ideal Value	Actual Value
1	Range of Detection	Meters	> 1	2	5
2	Weight	Kg	> 0.75	> 0.5	0.4
3	Range of Motion	Degrees	< 160	150	120
4	Variation of Voltage Input	Volts	± 0.2	± 0.1	± 0.1
5	Height	Cm	25	25	25
6	Water resistance	IPX	IPX2	IPX4	IPXX
7	Proper Acceleration	Degrees	±20	±20	±20
8	Brightness	Lumens/Watt	80	60	75

Rationale & Plan

Specification 1: The Not Creepy Elf will track a person(s) in a room. The product will need to be able to detect where the person(s) is standing from both a short distance and a far distance. The product's abilities will be limited to how far or close the person is. A PIR motion sensor will be used in the range between 1 and 2 meters.

Specification 2: The Not Creepy Elf will be moved around and placed in different areas around the user's home. Because of this, it is necessary to specify the weight of the product to the customer. The product must be light enough to carry and from a position and heavy enough for it to sit on its own without much support. Team 106 anticipates the product's weight to be between 500 and 750 grams.

Specification 3: The Not Creepy Elf will have the ability to turn its head a certain amount. The degrees of freedom the device has to move its head to track the motion of a person is necessary to specify to the customer. This limitation will help the customer determine the best area to place the product in their home. A servo motor will be used to allow the product to function properly. The degrees of freedom will vary depending on the product's setting.

Specification 4: The Not Creepy Elf will more than likely be turned on for long periods of time. The device will be powered from a 5V wall power supply. A variation between 4.8V and 5.2V is expected during the usage of the product.

Specification 5: Because the Elf on the Shelf is a very commonly used holiday decoration, it is reasonable to build the device close to the size of the traditional Elf on the Shelf. Not Creepy Elf will have its unique features apart from the traditional design, but it will look very similar. The dimensions of the product will be approximately 25 centimeters tall.

Specification 6: There may be instances where the customer would prefer to use the decoration in an outdoor environment. Because of this, the Not Creepy Elf, during normal operation, may come into contact with water. Identifying the amount of moisture that the product can withstand is crucial to its functionality in an outdoor environment. The failure of the product due to rain exposure is to be determined later in the course of the project.

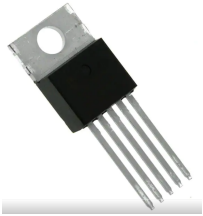
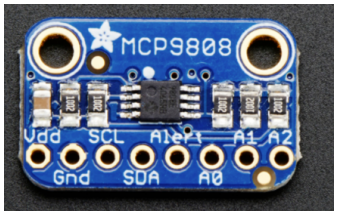
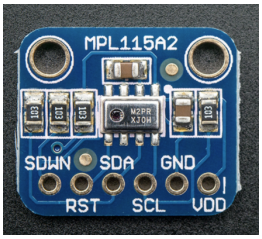
Specification 7: In the case where the Not Creepy Elf is lifted from its position, it will sense a change in pressure, allowing it to perform additional features. This pressure will be dependent on the weight of the Elf.

Specification 8: To further the effect of the Not Creepy Elf on the Halloween holiday, the Not Creepy Elf will need to have LED lights in each eye socket. The LEDs will turn on and glow when motion is detected and tracked. Two red LEDs will most likely be used, lighting at approximately 60 lumens.

Appendix C. Major Component Selection Rationale

Several potential off-the-shelf electrical and mechanical components that could be used in the project were researched for the device's main subsystems: Temperature Sensor, Accelerometer, and Stepper Motor. See the tables below for each solution.

Temperature Sensor Subsystem

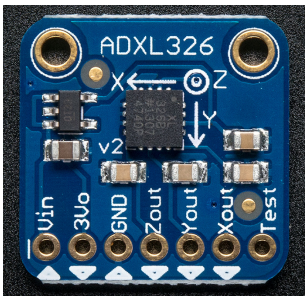
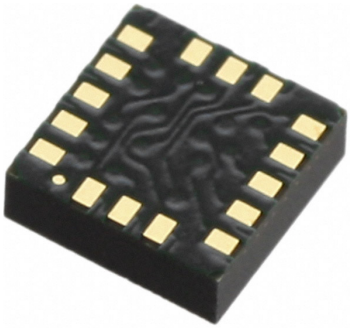
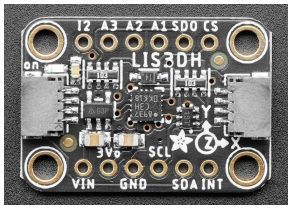
Solution	Pros	Cons
 Microchip TC74A5-3.3VAT $1.57 Link to Product	• Inexpensive • Runs with 2.7V to 5.5 • Operating temp range: -40 to +125 °C • Compatible with I2C interface for communication.	• Not very accurate when compared to competitors (Accuracy +/-2°C) • Does not read the temperature readily • Cheaply made product when compared to the competition
 Microchip MCP9808 $4.95 Link to Product	• Operating Temperature range: -40 to +125 °C • Highly accurate (+/-0.25°C) • Comes with header pin connections • Runs with 2.7V to 5.5 • Compatible with I2C interface for communication.	• Very Expensive • Relatively big when compared to other products
 MPL115A2 - I2C Barometric Pressure/Temperature Sensor $7.95 Link to Product	• Compatible with I2C interface for communication. • MPL115A2 can both pressure and temperature sensor • 1.5 hPa / 50 m altitude resolution • Comes with a detailed datasheet • Runs with 2.7V to 5.5 • Readily available	• Relatively more expensive than its competitors • Relatively big when compared to other products • Comes with a highly inaccurate and very basic temperature sensor

	libraries for coding	

Selection: The chosen Temperature sensor is **Microchip TC74A5-3.3VAT.**

Rationale: Microchip TC74A5-3.3VAT comes at a very affordable price, and the module comes with a relatively small footprint suitable for Not Creepy Elf. It is compatible with PSoC. It comes with very detailed specifications and libraries for the product are not hard to create.

Accelerometer Subsystem

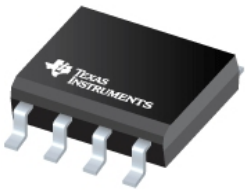
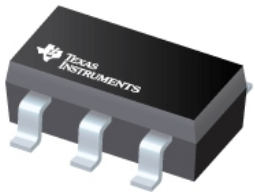
Solution	Pros	Cons
 Adafruit ADXL326 Triple AxisAxis Accelerometer $17.95 Link to Product	• Small form factor(19mmx19mm) • 3.3V or 5V intake • Already on a PCB (comes as a breakout board)	• Expensive • Soldering header pins required
 LIS3DSHTR $2.92 Link to Product	• Inexpensive • High Sensitivity • 3.6V intake • Adjustable Bandwidth • Selectable Scale • Temperature Sensor • Very small form(3mmx3mm) • Low power consumption	• Surface mount component • Not widely used
 Adafruit LIS3DH Triple	• 3.3V intake • Internal shift lever • Popular accelerometer • SPI or I2C communication	• Expensive • Soldering header pins required

Axis Accelerometer $10.64 Link to Product		

Selection: The chosen accelerometer is the Adafruit **ADXL326** Triple Axis Accelerometer.

Rationale: The Adafruit ADXL326 is ready to handle 5V and has the highest precision of measurement of the other potential solutions. While it is the most expensive solution, the difference in quality is worth the price. The outputs are analog and ratiometric, meaning that the 0g measurement is set at 1.65V, allowing for high precision in all directions of measurement.

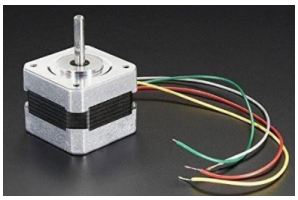
Operational Amplifier

Solution	Pros	Cons
 LM358 $0.42 Link to Product	• Inexpensive • Commonly used general-purpose OP Amp • Stable in DIY circuits • Easily obtainable • Number of channels: 2 • Supply: 3V - 32V • Bipolar	• Low slew rate and limited bandwidth • Bad for audio circuits
 OP07 N/a Link to Product	• Low Noise • No External Components Required • Supply: 5V - 44V • Bipolar	• Out of stock • Cost of component not given
 OPA210 $2.33 Link to Product	• Precision Op amp • Supply: 4.5V - 36V • Bipolar	• More expensive than its competitors

Selection: The chosen operational amplifier is **LM358**.

Rationale: While the LM358 is an older product and has a lower slew rate, it is reliable, easily obtainable, and fits our specifications. Team 106 does not believe the low slew rate will prove detrimental to our project. Team 106 is using the through-hole version for easy soldering. Additionally, because the LM358 is older and widely used, there are many examples of its use in cases where troubleshooting is necessary.

DC Motor Subsystem

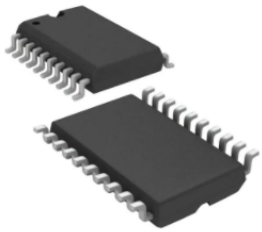
Solution	Pros	Cons
 GA12-N20 (Augiimor DC 6V 60RPM Speed Reduction Motor N20) $8.50 Link to Product	• 6VDC • Inexpensive • DC Motor • 60 RPM • Lightweight • In stock • Low torque (enough to move head)	• Does not come with a compatible motor driver
 MIKROE-1530 (STEPPER MOTOR PM GEARED UNI 5V) $8.00 Link to Product	• 5VDC • Inexpensive • Unipolar (doesn't need reverse current, multiple leads per phase) • 4096 steps per revolution • Lightweight • Step angle 0.088 • In stock	• Unipolar (less torque, less efficient) • 4 weeks lead time • Does not come with a compatible motor driver
 324 (NEMA-17) $14.00 Link to Product	• Bipolar (more torque, more efficient) • 200 steps per revolution • Step angle 1.8 Inexpensive • Lightweight • In stock	• 12V rated voltage • expensive • Bipolar (single winding per phase, needs reverse current) • Requires two full H-bridges (not provided) • Does not come with a

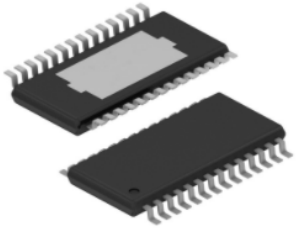
		compatible motor driver
 MS5N-1885-R (12V DC Motor) $3.95 Link to Product	• Inexpensive • Lightweight • In stock • Datasheetinto datasheet available • Small motor	• 12VDC • 9820 RPM • Too powerful for the application • Does not come with a compatible motor driver

Selection: The chosen stepper motor is the **GA12-N20** manufactured by Augiimor.

Rationale: The GA12-N20 DC motor is compatible with PSoC, running at 5VDC. The motor is inexpensive at $8.50. Of the four solutions, it is the only one compatible with the chosen motor driver. This motor has enough torque to move the Elf's head but is also not strong enough to get past a manual stopper if necessary.

Motor Driver

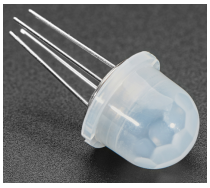
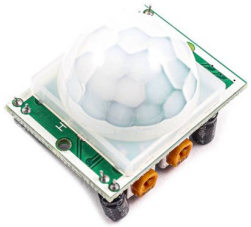
Solution	Pros	Cons
 DRV8804DW $3.78 Link to Product	• Works with unipolar stepper motors • Inexpensive • Small size • Lightweight • SPI Interface • 129 in stock • Max current: 800mA	• Surface mount • 6 weeks lead time • Supply: 8.2V - 60V Load: 8.2V - 60V
 DRV8873HPWPT $5.75 Link to Product	• Works with DC motors • Inexpensive • Small size • Lightweight • Supply: 0V - 5.5V • Load: 4.5V - 38V • Hardware Interface • Max current: 10A	• More expensive than its competitors • 35 weeks lead time • Surface mount

 DRV8811PWP $4.73 Link to Product	• Inexpensive • Small size • Lightweight • Supply: 3V - 5.5V • Load: 8V - 38V • 250 in stock • Max current: 1900mA	• Only works with bipolar stepper motors • 35 weeks lead time • Logic Interface • Surface mount

Selection: The chosen stepper motor drive is the **DRV8873HPWPT** manufactured by Texas Instruments.

Rationale: The DRV8873HPWPT is compatible with the chosen motor. The motor uses direct current and the motor drive works with DC motors. The motor pulls about 30 - 40 mA of current and the motor driver can handle up to 10A of current, which is more than enough! The DRV8873HPWPT is inexpensive at $5.75, can run at 5VDC, and is lightweight and compact. Additionally, the component is currently in stock.

PIR Motion Sensor Subsystem

Solution	Pros	Cons
 Mini Basic PIR Sensor - BL412 $1.95 Link to Product	• Small size • Inexpensive • Up to 8m range • 120-degree spread	• Needs external components to function correctly • Runs on 3.3V power • Low Range
 HC-SR501 PIR Motion	• Up to a 7-meter range • Integrated circuit • Works with 5V - 20V • 110-degree spread	• Bulky due to integrated circuit • Expensive

Detection Sensor $8.99 Link to Product		
 PIR Motion Sensor (JST $9.95 Link to Product	• Easy wiring • Greater range • Works with 5V - 12V	• Expensive • Degree spread not provided

Selection: The chosen PIR motion sensor is the **Mini Basic PIR Sensor** manufactured by Adafruit Industries LLC.

Rationale: The Mini Basic PIR Sensor is perfect for Team 106's needs. Its small design will be easy to hide and not ruin the aesthetic of the product as a whole. Additionally, a full-size PIR sensor would dig too deeply into the project budget to be comfortably viable. With some fine-tuning and potentially some troubleshooting, the mini Basic circumvents those issues.

Appendix D. Power Budget

The final PCB has two voltage rails: 5V and 3.3V. The temperature sensor, PIR sensor, PSoC microcontroller, accelerometer, and operational amplifiers take power from the 3.3V regulator. The DC motor, motor driver, and 3.3V regulator take power from a 5V wall power supply. The total current used is 1520mA. For cautionary measures, a 2A fuse is used.

Power Budget

Team Number:	106
Project Name:	Not Creepy Elf
Team Member Names:	Sai Srinivas Tatwik Meesala, Noah Blevins, Adriana Juarez, and Ian Mansfield
Version:	1

A. List ALL major components (active devices, integrated circuits, etc.) except for power sources, voltage regulators, resistors, capacitors, or passive elements

All Major Components	Component Name	Part Number	Supply Voltage Range	#	Absolute Maximum Current (mA) [1]	Total Current (mA)	Unit
Temperature Sensor	Microchip TC74A0-3.3VAT	TC74A0-3.3VAT	3.3V	1	26	26	mA
Stepper motor	Unipolar Stepper Motor Permanent Magnet Gear Motor 5VDC	MIKROE-1530	+5V	1	260	260	mA
Motor drive	DRV8873HPWPT Unipolar Motor Driver NMOS Hardware 24-HTSSOP	DRV8873HPWPT	0V - 5.5V	1	10	10	mA
BLE Module	PSoC™ 4 BLE Module	CYBLE-022001-00	1.9V - 5.5V	1	200	200	mA
Accelerometer	Adafruit ADXL326	ADXL335	5V	1	300	300	mA
Opamp	Texas Instruments LM358	LM358	3V - 32V	3	100	300	mA
Microcontroller	PSoC™ 4100S Plus	CY8CKIT-149	+1.8 - 5V	1	350	350	mA
PIR Motion Sensor	Mini Basic PIR Sensor	BL412	2.7V - 3.3V	1	100	100	mA

B. Assign each major component above to ONE power rail below. Try to minimize the number of different power rails in the design. Add additional power rails or change the power rail voltages if needed.

+5V Power Rail	Component Name	Part Number	Supply Voltage Range	#	Absolute Maximum Current (mA)	Total Current (mA)	Unit
	Stepper motor	MIKROE-1530	5V	1	260	260	mA
	Motor Drive	DRV8873HPWPT	0V - 5.5V	1	10	10	mA
	+3.3V low-dropout regulator	LD1117AV33	+2V - 26V	1	1250	1250	mA
						0	mA
						0	mA
					Subtotal	1520	mA
					Safety Margin	25%	
					Total Current Required on +5V Rail	1900	mA
c2. Regulator or Source Choice	+5V Regulator	L6R12H-050	90 - 264VAC	1	2000	2000	mA
					Total Remaining Current Available on +5V Rail	100	mA

+3.3V Power Rail	Component Name	Part Number	Supply Voltage Range	#	Absolute Maximum Current (mA)	Total Current (mA)	Unit
	Temperature sensor	TC74A0-3.3VAT	+3.3V to -3.2V	1	26	26	mA
	Mini Basic PIR Sensor	BL411	2.7V - 3.3V	1	100	100	mA
	PSoC™ 4 BLE Module	CYBLE-022001-00	1.9V - 5.5V	1	200	200	mA
	PSoC™ 4100S Plus	CY8CKIT-147	+1.8 - 5V	1	2.2	2.2	mA
	Accelerometer	ADXL335	3.3V	1	350	350	mA
	Opamp	LM358	3V - 32V	3	100	300	mA
					Subtotal	978.2	mA
					Safety Margin	25%	
					Total Current Required on +3.3V Rail	1222.75	mA
c4. Regulator or Source Choice	+3.3V low-dropout regulator	MIC2940A-3.3WT	+2V - 26V	1	1250	1250	mA
					Total Remaining Current Available on 3.3V Rail	27.25	mA

C. For each power rail above, select a specific voltage regulator using the same process as for major component selection. Confirm that the Total Remaining Current Available on each rail above is not negative.

D. Select a specific external power source (wall supply or battery) for your system, and confirm that it can supply all of the regulators for all of the power rails simultaneously. If you need multiple power sources, list each separately below and indicate which regulators will be connected to each supply. Confirm that the Total Remaining Current Available on each power source below is not negative.

External Power Source 1	Component Name	Part Number	Supply Voltage Range	Output Voltage	Absolute Maximum Current (mA)	Total Current (mA)	Unit
Power Source 1 Selection	Plug-in Wall Supply	L6R12H-050	90 - 264VAC	5	2000	10000	mA
Power Rails Connected to External Power Source 1						0	mA
	+3.3V low-dropout regulator	MIC2940A-3.3WT	+2V - 26V	3.3	1000	3300	mA
						0	mA
					Total Remaining Current Available on External Power Source 1	6700	mA

E. Calculate Battery Life (if applicable). For each battery, also check the worst-case lifetime of the battery by indicating the capacity in mAh.

	Component Name	Part Number	Supply Voltage Range	Capacity (mAh)	Required By Regulators	
(Not using a battery)	Battery	(full part number)	+12V	500	#REF!	
				Battery Life	#REF!	hours

Notes

External Supply Voltage should be determined by the dropout voltage for highest-voltage regulator (e.g., +14V for a +12V regulator).

If you have multiple units in your design (e.g., a base unit and remote unit) then you need a separate power budget for each unit

[1] For inductive loads (e.g., motors, solenoids) this is often called "stall current" on the data sheet

Appendix E. Bill of Materials

Not Creepy Elf Revised: Saturday, 12/4/2021 19:14:51

Bill Of Materials **Revision: 2.0**

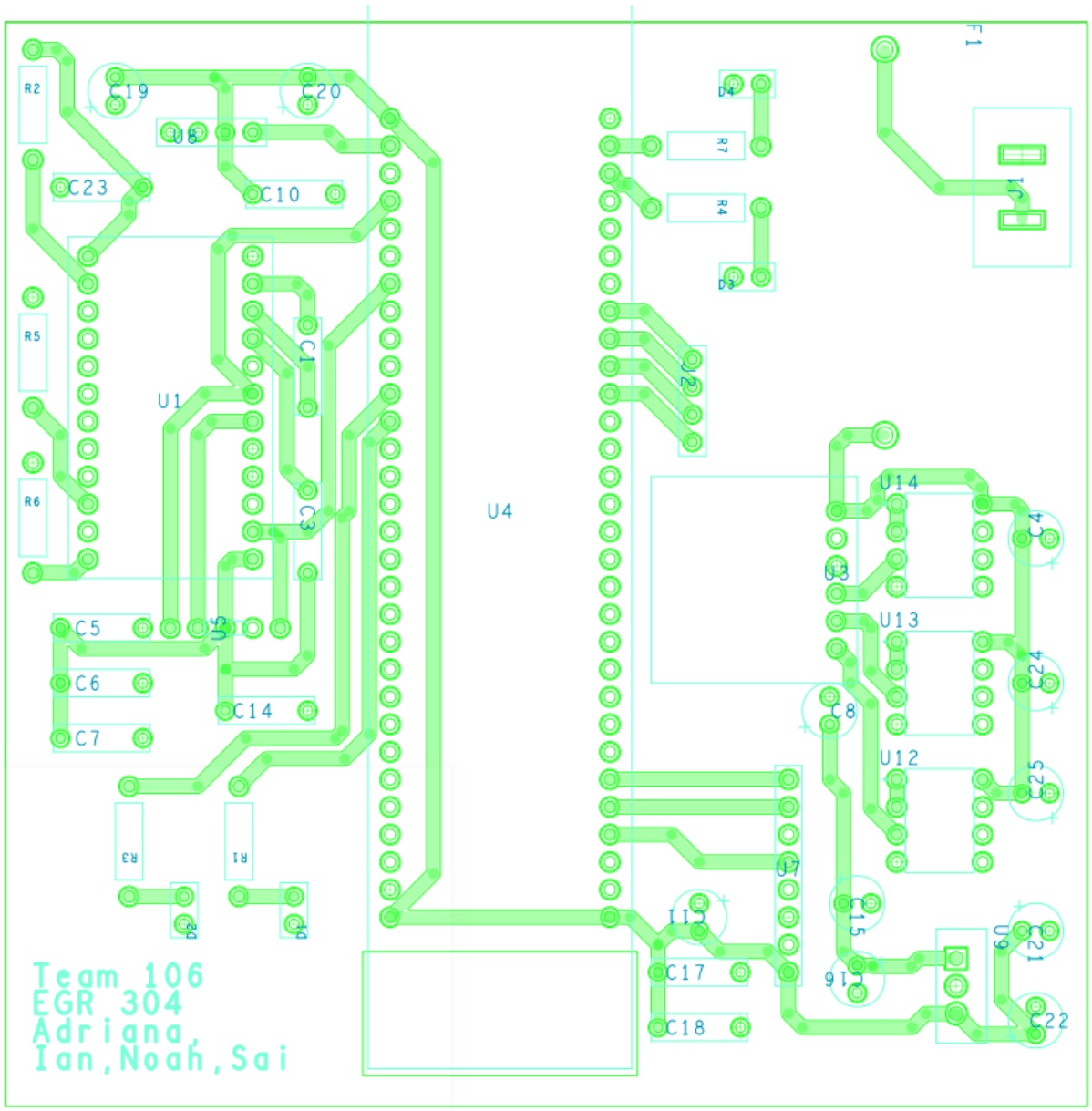
Team 106

Name Adriana J., Noah B., Sai Srinivas Tatwik M., Ian M.

Date December 4,2021 19:14:51

Part Name/Description	Item	Quantity	Schematic Reference Designators	Part	Manufacturer	Manufacturer Part #	Supplier	Supplier Part #	Unit Prototype Cost	Total Prototype Cost	Unit Production Cost	Total Production Cost	Date Ordered	# Received	Surplus
Capacitors	1	3	C1	1uF	WiMas	-	Amazon						09/21/2021	30	17
Capacitors			C6	1uF	WiMas	-	Amazon						09/21/2021	-	-
Capacitors			C14	1uF	WiMas	-	Amazon						09/21/2021	-	-
Capacitors	2	1	C2	47nF	WiMas	-	Amazon						09/21/2021	30	-
Capacitors	3	8	C3	0.1uF	WiMas	-	Amazon						09/21/2021	30	-
Capacitors			C4	0.1uF	WiMas	-	Amazon						09/21/2021	-	-
Capacitors			C13	0.1uF	WiMas	-	Amazon						09/21/2021	-	-
Capacitors			C18	0.1uF	WiMas	-	Amazon						09/21/2021	-	-
Capacitors			C19	0.1uF	WiMas	-	Amazon						09/21/2021	-	-
Capacitors			C20	0.1uF	WiMas	-	Amazon						09/21/2021	-	-
Capacitors			C21	0.1uF	WiMas	-	Amazon						09/21/2021	-	-
Capacitors			C22	0.1uF	WiMas	-	Amazon						09/21/2021	-	-
Capacitors	4	1	C5	BULK	WiMas	-	Amazon						09/21/2021	-	-
Capacitors	5	1	C7	0.1 uF	WiMas	-	Amazon						09/21/2021	-	-
Capacitors	6	4	C8	0.1uF	WiMas	-	Amazon						09/21/2021	-	-
Capacitors			C9	0.1uF	WiMas	-	Amazon						09/21/2021	-	-
Capacitors			C11	0.1uF	WiMas	-	Amazon						09/21/2021	-	-
Capacitors			C16	0.1uF	WiMas	-	Amazon						09/21/2021	-	-
Capacitors	7	3	C10	1uF	WiMas	-	Amazon						09/21/2021	-	-
Capacitors			C12	1uF	WiMas	-	Amazon						09/21/2021	-	-
Capacitors			C15	1uF	WiMas	-	Amazon						09/21/2021	-	-
Capacitors	8	1	C17	10uF	WiMas	-	Amazon						09/21/2021	30	-
LED (Red)	9	4	D1	LED	-	-	-						09/21/2021	-	-
LED (Red)			D2	LED	-	-	-						09/21/2021	-	-
LED (Red)			D3	LED	-	-	-						09/21/2021	-	-
LED (Red)			D4	LED	-	-	-						09/21/2021	-	-
Female Barrel Jack	10	1	J1	CON2	Tri-Mag	L6R12H-050	DigiKey		$7.60	$7.60	$5.32	$5.32	09/21/2021	-	-
Resistor	11	4	R1	220 Ohm	-	-	-						09/21/2021	-	-
Resistor			R6	220 Ohm	-	-	-						09/21/2021	-	-
Resistor			R7	220 Ohm	-	-	-						09/21/2021	-	-
Resistor			R8	220 Ohm	-	-	-						09/21/2021	-	-
Resistor	12	1	R2	SENSE-2	-	-	-						09/21/2021	-	-
Resistor	13	1	R3	SENSE-1	-	-	-						09/21/2021	-	-
Resistor	14	1	R4	10 kOhm	-	-	-						09/21/2021	-	-
Resistor	15	3	R5	100 kOhm	-	-	-						09/21/2021	-	-
Resistor			R9	100 kOhm	-	-	-						09/21/2021	-	-
Resistor			R11	100 kOhm	-	-	-						09/21/2021	-	-
Resistor	16	3	R10	200 kOhm	-	-	-						09/21/2021	-	-
Resistor			R12	200 kOhm	-	-	-						09/21/2021	-	-
Resistor			R13	200 kOhm	-	-	-						09/21/2021	-	-
Microcontroller	17	1	U1	PSoC 4100S Plus	Cypress	PSoC 4100S Plus	Cypress		$19.95	$19.95	$19.95	$19.95	09/21/2021	-	-
H-Bridge Motor Driver	18	1	U2	DRV8873H	Texas Instruments	DRV8873HPWPT	DigiKey		$5.07	$5.07	$3.28	$3.28	09/21/2021	5	1
5V Stepper Motor	19	1	U3	28BYJ-48	MikroElektronika	MIKROE-1530	DigiKey		$8	$8	$8	$8	09/21/2021	5	1
PIR Sensor	20	1	U4	BL412	Senba Sensing	BL412	Adafruit		$1.95	$1.95	$1.56	$1.56	09/21/2021	8	4
Temperature Sensor	21	1	U5	TC74A0-3.3VAT	Microchip Technology	TC74A0-3.3VAT	Mouser		$1.67	$1.67	$1.67	$1.67	09/21/2021	4	1
Voltage Regulator	22	1	U6	BD3570YFP-ME2	ROHM Semiconductor	BD3570YFP-ME2	DigiKey		$2.68	$2.68	$1.94	$1.94	09/21/2021	4	-
Dual-Operational Amplifiers	23	2	U7	LM358	Texas Instruments	LM358	DigiKey		$0.40	$0.80	$0.40	$0.80	09/21/2021	15	3
Dual-Operational Amplifiers			U9	LM358	Texas Instruments	LM358	DigiKey						09/21/2021		-
Accelerometer	24	1	U8	ADXL335 Accelerometer	Adafruit	ADXL335	Adafruit		$14.95	$14.95	$11.96	$11.96	09/21/2021	5	-
								TOTAL		62.668		54.478			

Appendix F. Printed Circuit Board Layout

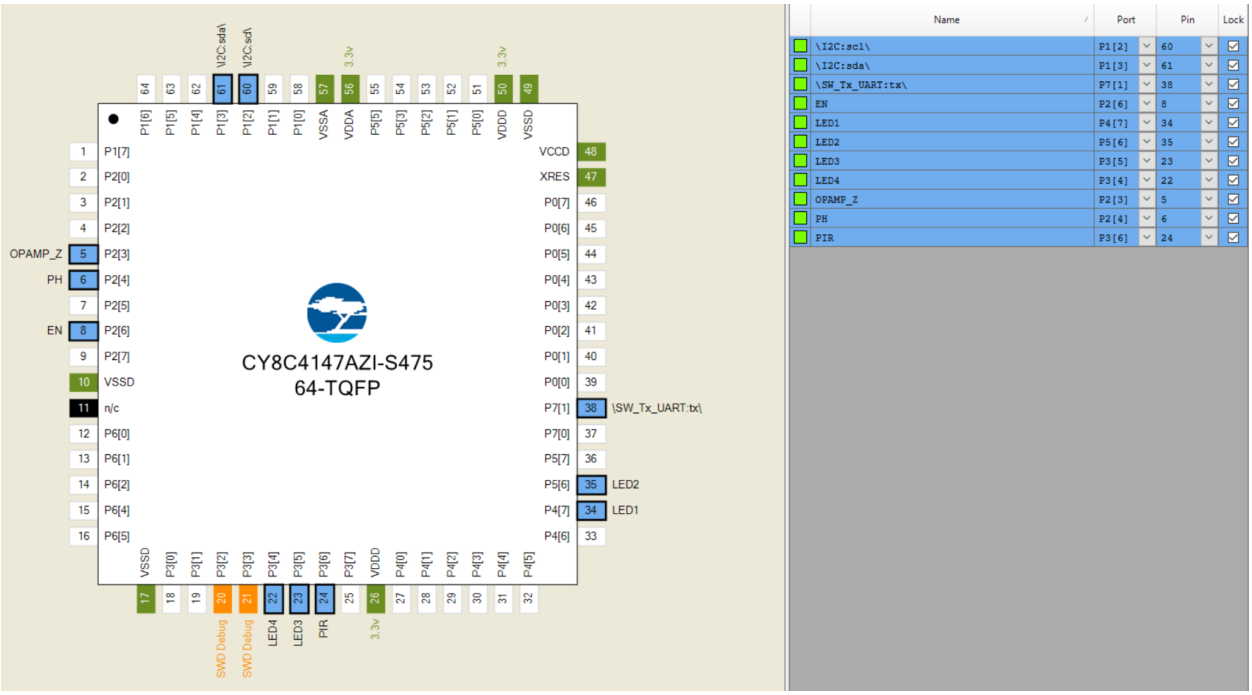


ART FILM - TOP

ART FILM - BOTTOM

Appendix G. Top Design

The PSoC® Creator Top Design with all of the necessary components for our project is shown below as well as the pins and ports that will be used.



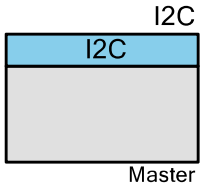
PIR Motion Sensor

The PIR motion sensor will sense heat energy when a person walks by and will trigger the DC motor.

PIR 3[6]

Temperature Sensor

The RFID reader will be used to change the holiday mode on the Elf.



Eye Socket LEDs

Two LEDs will be placed in the Elf's eye sockets and will turn on when the motor and motion sensor are triggered.

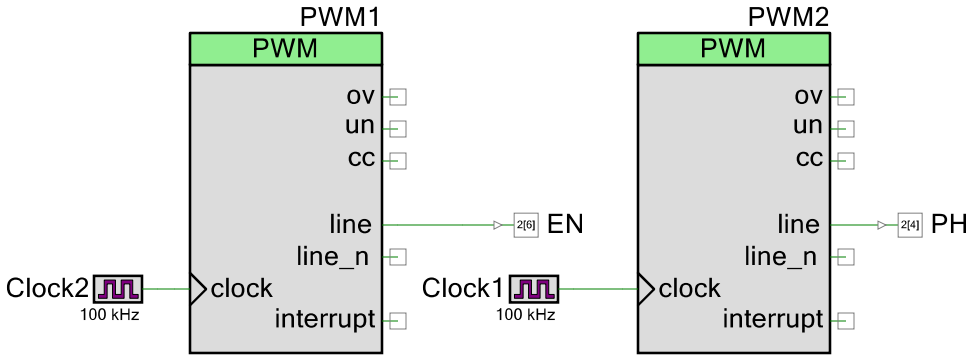
4[7] LED1
5[6] LED2
3[5] LED3

Debugging LED (LED 4)

This LED will be used for debugging.

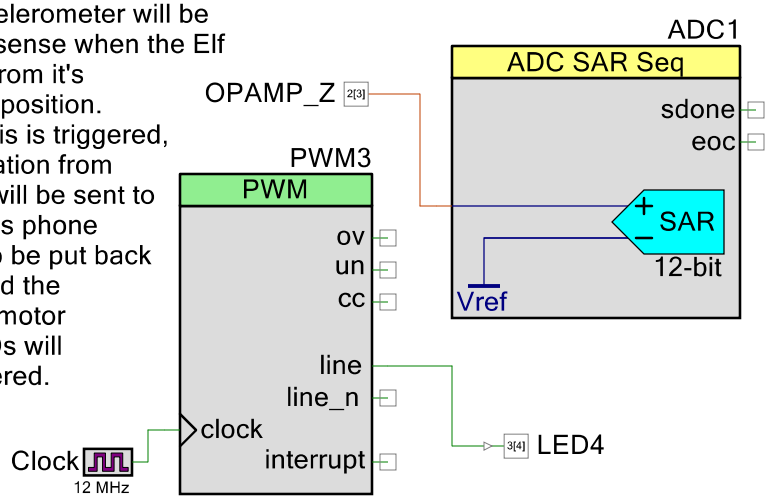
DC Motor

The DC motor will move the Elf's head when the motion sensor is triggered.



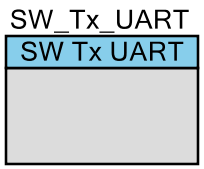
Accelerometer

The accelerometer will be used to sense when the Elf is lifted from it's original position. When this is triggered, a notification from the Elfnwill be sent to the user's phone urging to be put back down and the stepper motor and LEDs will be triggered.



BLE Module

The BLE module integrated on the PSoC microcontroller will be used for Bluetooth communication between the users cellphone and the Elf.



Serial Terminal Configuration:

Baud Rate:	57600
Data Bits:	8
Parity:	None
Stop Bits:	1
Flow Control:	None

Top Design

Team 106: Not Creepy Elf
Adriana Juarez, Sai Tatwik Meesala, Ian Mansfield, Noah Blevins

Appendix H. Main C/C++ Code

```
/*
//============================
//TEAM 106 FINAL PROJECT CODE
//============================
*/

#include <project.h>
#include <stdio.h> // for sprintf function
#include <tc74_i2c.h> // header file for tc74 library

#define ON 1
#define OFF 0

uint16 map(uint16 x, uint16 in_min, uint16 in_max, uint16 out_min, uint16 out_max)
  {
    return (x - in_min) * (out_max - out_min) / (in_max - in_min) + out_min;
  }

int i = 0;


int main(void)
{
  uint8 var = 0; // for motion sensor
  uint8 CurrentTemp = 0; // for temp sensor

  uint16 ADC = 0; // for accelerometer
  uint16 test = 0; // for accelerometer

  //char outputBuffer[16] = {0}; // used for Putty

  CyGlobalIntEnable; /* Enable global interrupts. */
  I2C_Start();
  //SW_Tx_UART_Start(); // used for Putty

  PWM1_Start();
  PWM2_Start();
  PWM3_Start();

  ADC1_Start();
  ADC1_StartConvert();
  ADC1_IsEndConversion(ADC1_WAIT_FOR_RESULT);

  // initialize motor off
```

```
PWM1_WriteCompare(1000);
PWM2_WriteCompare(1000);

for(;;)
{
    var = PIR_Read();
    // Putty - PIR Motion Sensor
    /*sprintf(outputBuffer, "%d", var );
    SW_Tx_UART_PutString(outputBuffer);
    SW_Tx_UART_PutCRLF();
    CyDelay(100);*/

    CurrentTemp = TC74_ReadTemp(TC74_RTR);
    // Putty - Temperature Sensor
    /*SW_Tx_UART_PutCRLF();
    sprintf( outputBuffer, "Current Temperature = %d", CurrentTemp );
    SW_Tx_UART_PutString(outputBuffer);
    SW_Tx_UART_PutCRLF();
    CyDelay(100);*/

    // For Accelerometer
    ADC = ADC1_GetResult16(0);
    test = map(ADC,576,860,0,2049);


    // Temperature Sensor, PIR Motion Sensor, Accelerometer
    if(CurrentTemp <= 24 && var == 1 && test >= 0300)
    {
        // Temperature Sensor
        LED3_Write(ON);
        CyDelay(200);
        LED3_Write(OFF);
        CyDelay(200);

        // PIR Motion Sensor
        LED1_Write(ON);
        LED2_Write(ON);

        PWM1_WriteCompare(0); // motor on
        CyDelay(1500);

        PWM1_WriteCompare(1000); // motor off
        CyDelay(1000);

        PWM2_WriteCompare(0); // motor on
        CyDelay(1500);
```

```
    PWM2_WriteCompare(1000); // motor off
    CyDelay(1000);

    LED1_Write(OFF);
    LED2_Write(OFF);

    // Accelerometer
    PWM3_WriteCompare(test);
}

// Temperature Sensor
else if(CurrentTemp <= 24)
{
    // blink led 3 continuously until false
    LED1_Write(OFF);
    LED2_Write(OFF);
    LED4_Write(OFF);

    LED3_Write(ON);
    CyDelay(200);
    LED3_Write(OFF);
    CyDelay(200);
}

// PIR motion Sensor
else if (var == 1)
{
    // turn led 1 & 2 on and rotate motor shaft bidirectionally
    LED1_Write(ON);
    LED2_Write(ON);

    PWM1_WriteCompare(0); // motor on
    CyDelay(1500);

    PWM1_WriteCompare(1000); // motor off
    CyDelay(1000);

    PWM2_WriteCompare(0); // motor on
    CyDelay(1500);

    PWM2_WriteCompare(1000); // motor off
    CyDelay(1000);

    LED1_Write(OFF);
    LED2_Write(OFF);
```

```
            var = PIR_Read();
        }

        // Accelerometer
        else if(test >= 0300)
        {
            PWM3_WriteCompare(test);
        }

        // Turn everything off
        else
        {
            LED1_Write(OFF);
            LED2_Write(OFF);
            LED3_Write(OFF);
            LED4_Write(OFF);

            PWM1_WriteCompare(1000);
            PWM2_WriteCompare(1000);
            PWM3_WriteCompare(0);
        }

    }
}

/* [] END OF FILE */
```

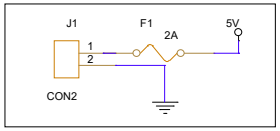
Appendix I. BLE C/C++ Code

```
/*
//========================
//TEAM 106 BLE PROJECT CODE
//========================
*/

#include <project.h>

void NUS_SendStr(char *str)
{

   CYBLE_GATTS_HANDLE_VALUE_NTF_T notifyHandle;
   notifyHandle.attrHandle = CYBLE_NUS_NUS_TX_CHAR_HANDLE;
   notifyHandle.value.len = strlen(str);
   notifyHandle.value.val = (uint8*)str;

   CyBle_GattsNotification(cyBle_connHandle, &notifyHandle);
   CyBle_ProcessEvents();
}

void BLECallBack(uint32 eventCode, void *eventParam)
{
   switch(eventCode)
   {
      case CYBLE_EVT_GAP_DEVICE_DISCONNECTED:
      case CYBLE_EVT_STACK_ON:
         {
            //LED1_Write(1u);
            CyBle_GappStartAdvertisement(CYBLE_ADVERTISING_FAST);
            break;
         }

      case CYBLE_EVT_GAP_DEVICE_CONNECTED:
         {
            //LED1_Write(0u);
            CyBle_GappStopAdvertisement();
            break;
         }

      case CYBLE_EVT_GATTS_WRITE_REQ:
      case CYBLE_EVT_GATTS_WRITE_CMD_REQ:
         {
            //
            CYBLE_GATTS_WRITE_REQ_PARAM_T *eventData;
```

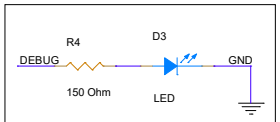
```
            eventData = (CYBLE_GATTS_WRITE_REQ_PARAM_T *) eventParam;

            uint16 length = eventData->handleValPair.value.len;
            char str[20] = "";
            memcpy(str, eventData->handleValPair.value.val, length);

            if(!strcmp(str, "Halloween") || !strcmp(str, "halloween"))
            {
                NUS_SendStr("Halloween mode on.");
                LED1_Write(0);
            }
            else if(!strcmp(str, "Christmas") || !strcmp(str, "christmas"))
            {
                NUS_SendStr("Christmas mode on.");
                LED1_Write(1);
            }

            //NUS_SendStr(str);

            /*LED1_Write(1);
            CyDelay(100);
            LED1_Write(0);*/

            //response
            if(eventCode == CYBLE_EVT_GATTS_WRITE_REQ)
                CyBle_GattsWriteRsp(cyBle_connHandle);
        }
    }
}

int main (void)
{
    /* Place the initialization/startup code here (e.g. MyInst_Start()) */

    CyGlobalIntEnable; /* Uncomment this line to enable global interrupts. */

    //I2C_Start();
    CyBle_Start(BLECallBack);

    for(;;)
    {
        CyBle_ProcessEvents();
    }
}

/* [] END OF FILE */
```

Appendix J. Hardware Design

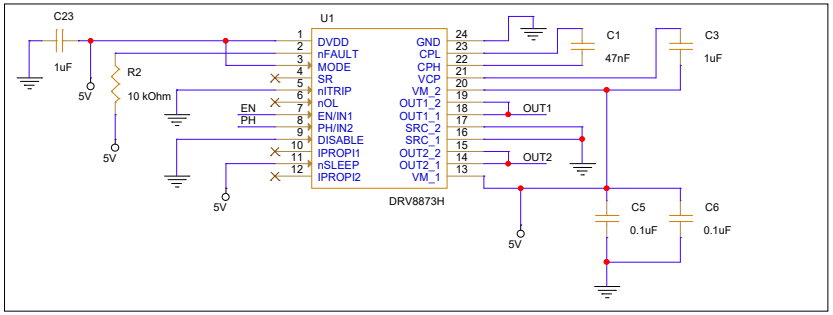
POWER SYSTEM (L6R12H-050)



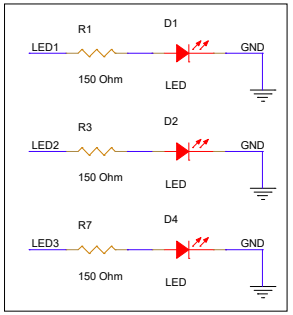
DEBUGGING LED



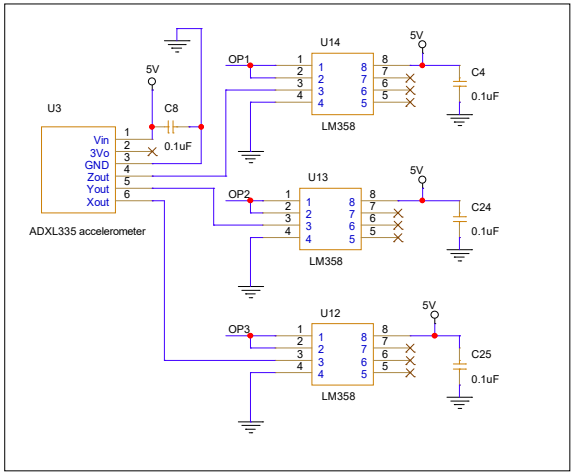
MOTOR DRIVER (DRV8873H)



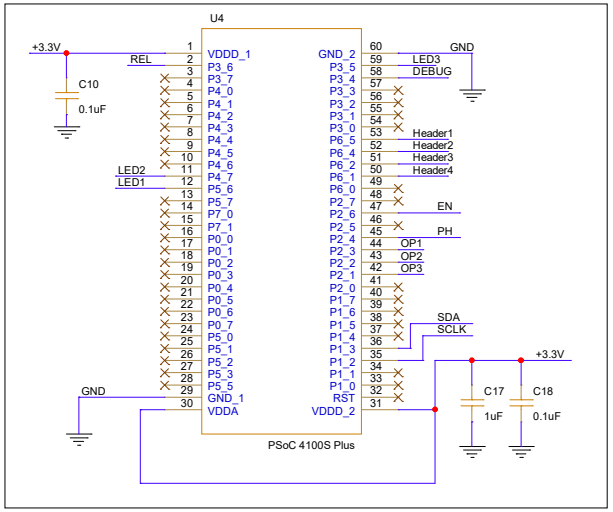
LEDs



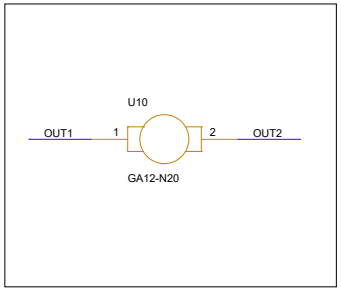
Accelerometer



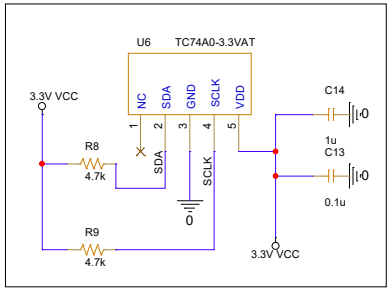
MICROCONTROLLER (PSoC 4100S Plus)



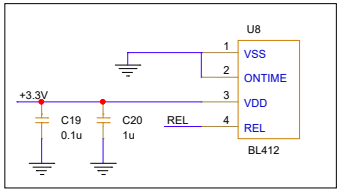
DC MOTOR (GA12-N20, 6V)



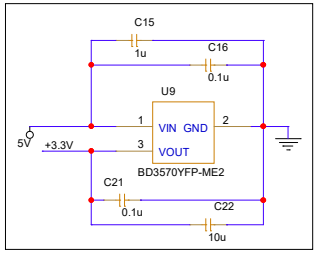
Temperature Sensor



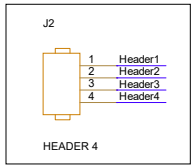
PIR Motion Sensor



+3.3V Regulator



Header Pins



NOTE: EACH RESISTOR IS 1/4 WATTAGE UNLESS OTHERWISE SPECIFIED

Hardware Design

Title: Not Creepy Elf		
Size: Custom	Document Number: Team 106 - Adriana J., Noah B., Sai Tatwik M., Ian M.	Rev: 1.0
Date: Tuesday, November 23, 2021	Sheet 1 of 1	