# Tensegrity Medical Device – 67% Update

**Alicia Corona, Claire Mitchell, Norma Munoz**

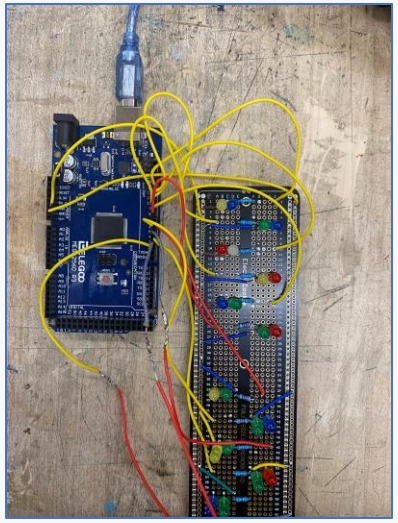
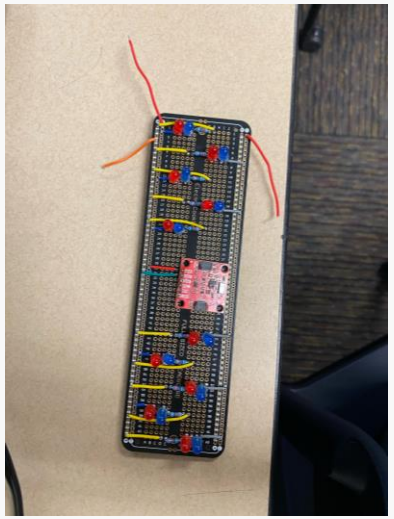
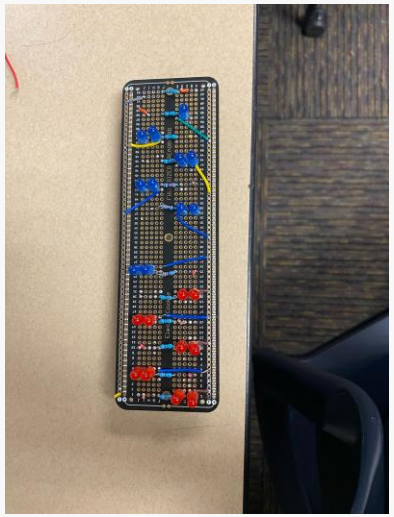
# Project Description

- Utilize photo biomodulation (PBM) technology
- Red LED lights, infrared sensors, & rechargeable battery
- Design a cutting-edge tool that monitors blood flow & oxygen circulation
- Offers a non-invasive solution for cardiovascular health monitoring

- Enhances cellular function, promotes tissue repair, and reduces inflammation
- Applicable to medical institutions, rehab centers, military, and sports teams
- Partnering with EE & CS Capstone to enhance teamwork skills
- Jesslynn Armstrong, President, Light Matter Solutions, LLC

# Overview

- Have a top encasing designed and printed
- Used PCBs we found online and soldered them in parallel along with our LEDs and resistors
- Code for lights, button timer, frequency, Heart rate monitoring, and temperature reading
- Using an UNO board currently to control the LEDs


# Code – Lights Modes

- Only going to be used if we use one of the digital pins
- For the basic set up we are only using the 5V output pin

# Code – Button Timer

- All we need to move forward with this design is to finalize placement
- We have the 10KOhm resistor as well as the wires to power and fix it into the board

sketch_feb25b §

```
// Defined pins for LEDs
const int ledPins[] = {2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13}; // Array for LED pins
const int buttonPin = A0; // Button connected to analog pin A0 (or use any digital pin)

bool ledState = false; // Variable to track LED state
bool lastButtonState = LOW; // Variable to store previous button state
unsigned long lastPressTime = 0; // Stores the time when the button was last pressed
const unsigned long autoOffTime = 600000; // 10 minutes in milliseconds

void setup() {
  // Set LED pins as output
  for (int i = 0; i < 12; i++) {
    pinMode(ledPins[i], OUTPUT);
  }

  pinMode(buttonPin, INPUT); // Set button pin as input
}

void loop() {
  bool buttonState = digitalRead(buttonPin); // Read button state

  // Check for button press (state change from LOW to HIGH)
  if (buttonState == HIGH && lastButtonState == LOW) {
    ledState = !ledState; // Toggle LED state
    lastPressTime = millis(); // Update the last press time
    delay(200); // Debounce delay
  }

  lastButtonState = buttonState; // Update last button state

  // Check if 10 minutes have passed since the last button press
  if (ledState && (millis() - lastPressTime >= autoOffTime)) {
    ledState = false; // Turn off LEDs automatically
  }

  // Set all LEDs based on ledState
  for (int i = 0; i < 12; i++) {
    digitalWrite(ledPins[i], ledState ? HIGH : LOW);
  }
}
```

# Code - Frequency

- Plus 38 more lines of code
- The purpose is to demonstrate the different frequencies our client wants to include in the final

frequency_arduino

```
// Define pins 2-12 for LEDs
const int ledPins[] = {2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12};
const int numPins = 11;

float frequency = 1.0; // Default frequency in Hz
unsigned long intervalMicros = 500000; // Default: 1 Hz (500ms = 500000us)
unsigned long previousMicros = 0;
bool ledState[numPins] = {LOW}; // Array to store the state of each LED

void setup() {
  // Set up all pins 2-12 as OUTPUT
  for (int i = 0; i < numPins; i++) {
    pinMode(ledPins[i], OUTPUT);
  }
  Serial.begin(9600);
  Serial.println("Enter a number (1-7) or a custom frequency in Hz:");
}

void loop() {
  // Check for user input from Serial Monitor
  if (Serial.available() > 0) {
    float input = Serial.parseFloat();  // Read input as float
    if (input > 0) {
      int selection = int(input);  // Convert to int for switch case
      switch (selection) {
        case 1:
          frequency = 73;
          break;
        case 2:
          frequency = 147;
          break;
        case 3:
          frequency = 294;
          break;
```

# Code – Heart Rate Monitoring

- Plus 65 lines more of code

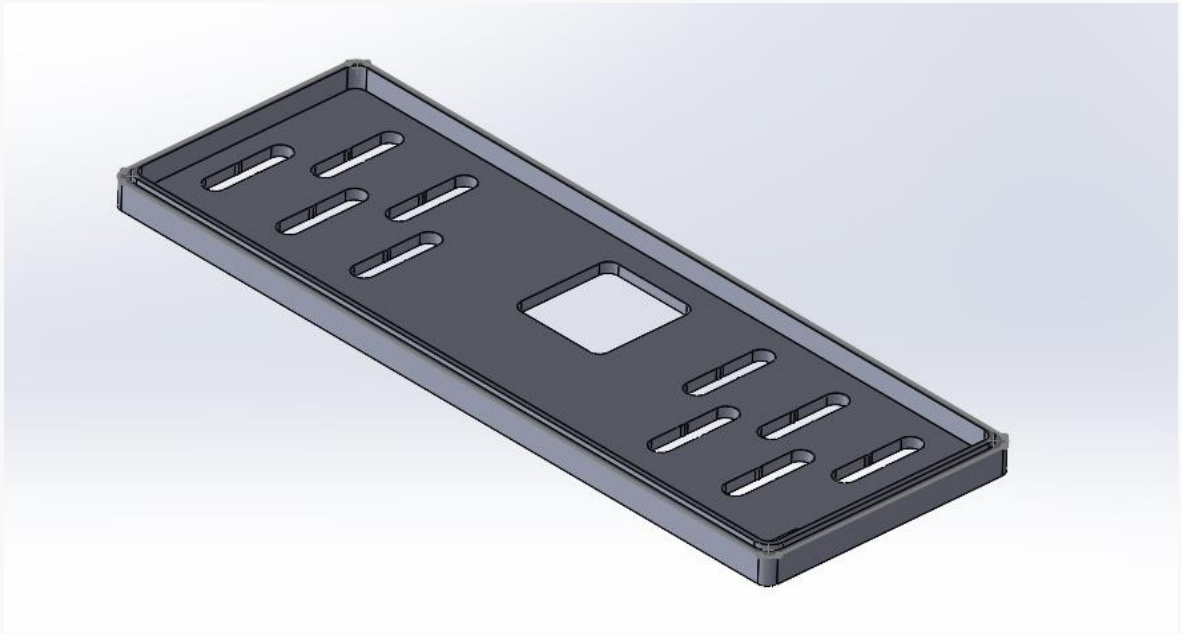
```
/*
  Optical Heart Rate Detection (PBA Algorithm) using the MAX30105 Breakout
  By: Nathan Seidle @ SparkFun Electronics
  Date: October 2nd, 2016
  https://github.com/sparkfun/MAX30105_Breakout

  This is a demo to show the reading of heart rate or beats per minute (BPM) using
  a Penpheral Beat Amplitude (PBA) algorithm.

  It is best to attach the sensor to your finger using a rubber band or other tightening
  device. Humans are generally bad at applying constant pressure to a thing. When you
  press your finger against the sensor it varies enough to cause the blood in your
  finger to flow differently which causes the sensor readings to go wonky.

  Hardware Connections (Breakoutboard to Arduino):
  -5V = 5V (3.3V is allowed)
  -GND = GND
  -SDA = A4 (or SDA)
  -SCL = A5 (or SCL)
  -INT = Not connected

  The MAX30105 Breakout can handle 5V or 3.3V I2C logic. We recommend powering the board with 5V
  but it will also run at 3.3V.
*/

#include <Wire.h>
#include "MAX30105.h"

#include "heartRate.h"

```

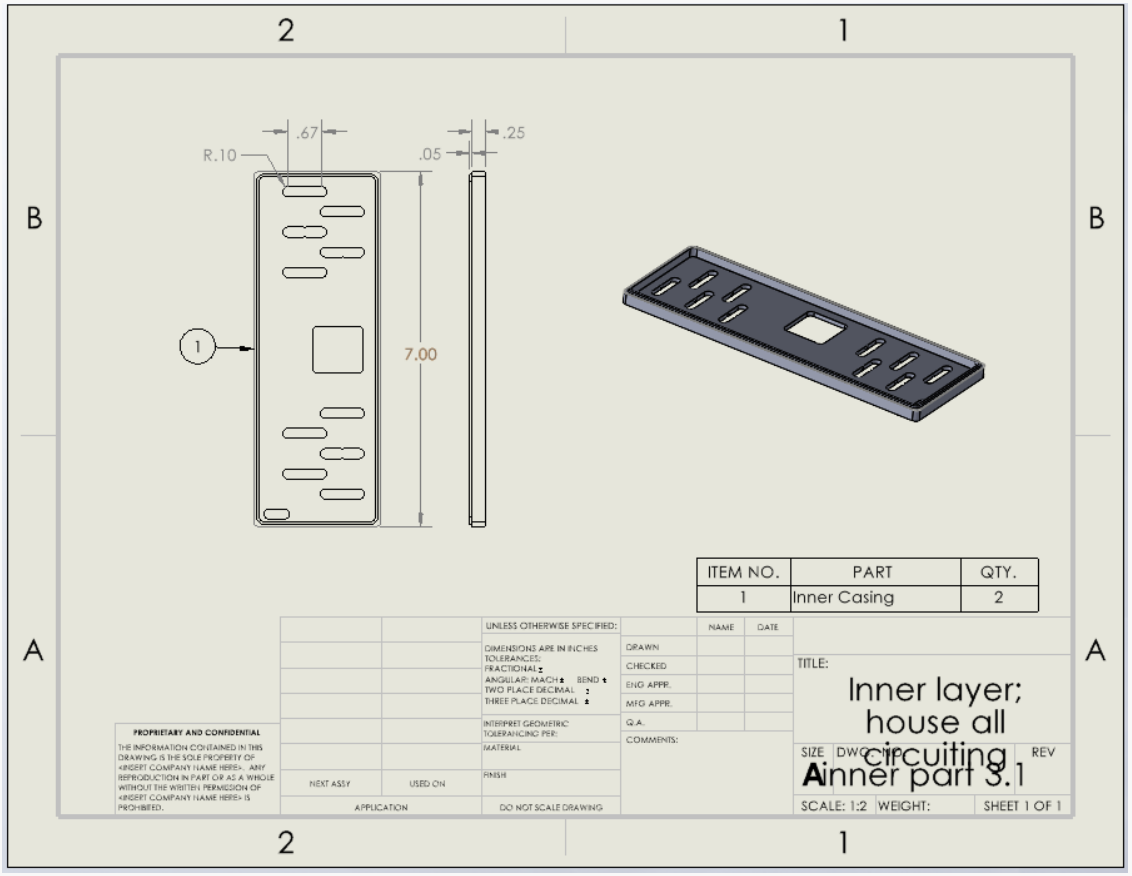
# Code – Temperature reading

- Plus 31 more lines of code

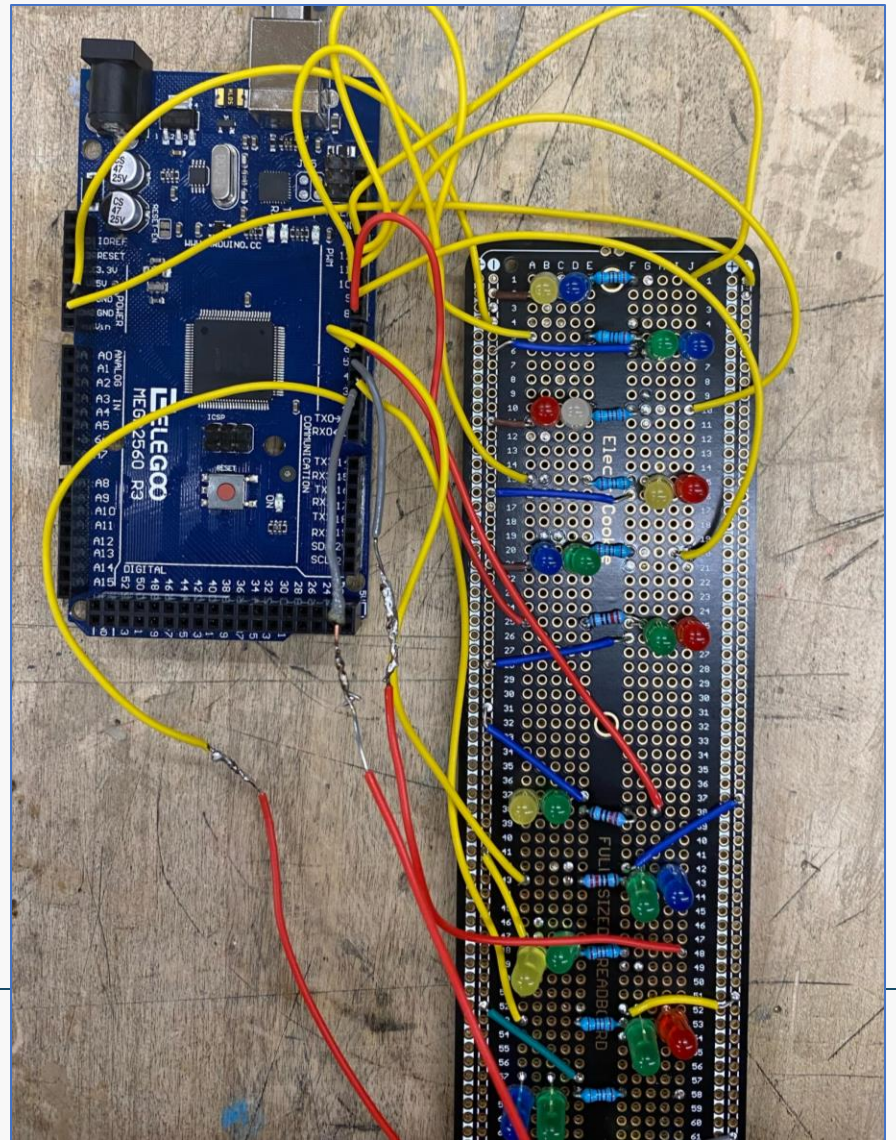
```
/*
  MAX3010 Breakout: Read the onboard temperature sensor
  By: Nathan Seidle @ SparkFun Electronics
  Date: October 20th, 2016
  https://github.com/sparkfun/MAX30105_Breakout

  This demo outputs the onboard temperature sensor. The temp sensor is accurate to +/-1 C but
  has an astonishing precision of 0.0625 C.

  Hardware Connections (Breakoutboard to Arduino):
  -5V = 5V (3.3V is allowed)
  -GND = GND
  -SDA = A4 (or SDA)
  -SCL = A5 (or SCL)
  -INT = Not connected

  The MAX30105 Breakout can handle 5V or 3.3V I2C logic. We recommend powering the board with 5V
  but it will also run at 3.3V.
*/

#include <Wire.h>

#include "MAX30105.h"  //Get it here: http://librarymanager/All#SparkFun_MAX30105
MAX30105 particleSensor;

void setup()
{
  Serial.begin(9600);
```

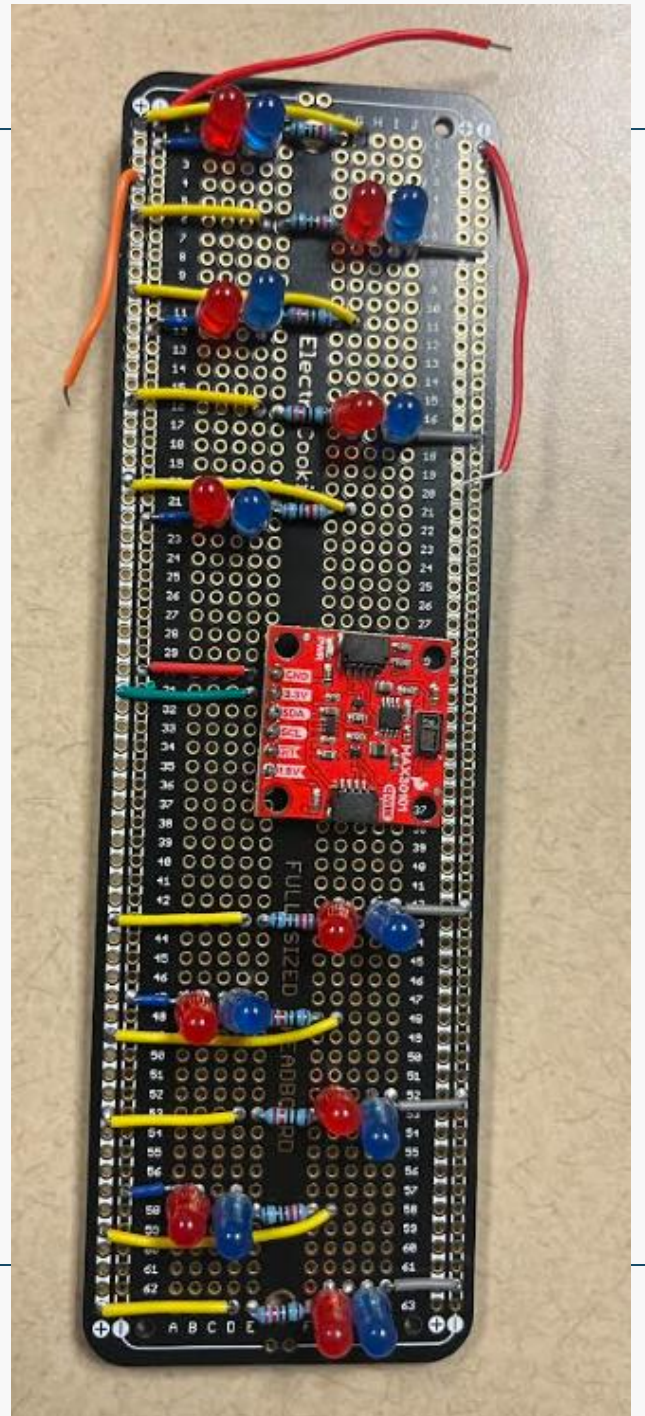
# Case Design – CAD





# EE – Before and After





# Purchasing Plan

| # | Part Name | Quantity Needed | Purchase Quantity | Quantity Arrived | Price | Total Unit Price | Ordered (Y/N) | Notes |
|---|---|---|---|---|---|---|---|---|
| 1 | Multiplexer Switch ICs | 10 | 10 | 10 | $1.39 | $9.15 | Y | |
| 2 | Resistors (1500 pcs) | 1 | 1 | 1 | $7.99 | $7.99 | Y | |
| 3 | SMD Capacitors (720 pcs) | 1 | 1 | 1 | $7.99 | $7.99 | Y | |
| 4 | Red LEDs | 16 | 16 | 16 | $0.44 | $7.04 | Y | |
| 5 | Infrared LED | 32 | 32 | 32 | $6.67 | $66.70 | Y | |
| 6 | PPG Sensor | 1 | 1 | 1 | $15.90 | $15.90 | Y | |
| 7 | HUZZAH32 - ESP32 Feather Board | 2 | 2 | 2 | $24.50 | $49.00 | Y | |
| 8 | Custom PCB | 1 | 1 | 0 | $64.79 | $64.79 | N | PCBs have been paid for and sent to manufacturer |
| 9 | Lithium Ion Polymer Battery | 1 | 1 | 0 | $9.99 | $9.99 | N | |
| 10 | Flexible Lithium Polymer Battery | 1 | 1 | 0 | Need quote | N/A | N | |
| 11 | TPU 95A HF | 1 | 1 | 1 | $41.99 | $41.99 | Y | |
| | | Percent Purchased | 96.44 | % | Total Spent | $280.54 | Total on Hand | 92.64705882 % |
| | | Budget Spent | 56.108 | % | Total Budget | $500.00 | | |

# Manufacturing Plan

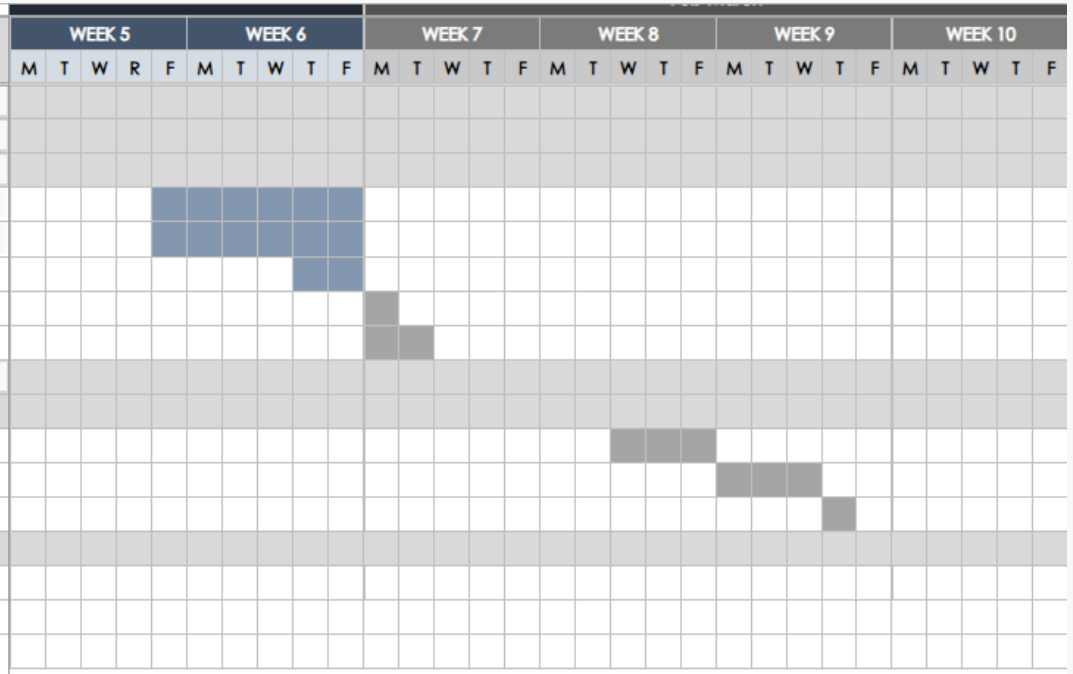
- ME portion – 64%
- Entire capstone – 72%

| Part | Time [hours[ | Manufacturing Method | Quantity | Progress % |
|---|---|---|---|---|
| Custom PCB | 2 | Ordered- Third party provider | 19 | 100% (Design) 80% (Manufactured) |
| Wiring Components | 3 | Soldering Kit | 38 | 85% |
| LED Components | 3 | Manufactured w/ PCB | 95 | 95% |
| Casing - inner | ~ 4 | 3D Printed | 2 | 50% |
| Casing - outer | ~ 4 | 3D Printed | 2 | 50% |
| Adhesive | - | Ordered- Third party provider | 1 | 0% |

- some design revisions are in order especially for casing component.
- Implementing an in-circuit system configured in series instead of parallel
- enabling Bluetooth integration with the designated app for the device
  - Collaborated with Computer Science
- Finalize material selection for Adhesive

# Gantt Chart

| TASK ID | TASK TITLE | TASK OWNER | START DATE | DUE DATE | DURATION IN DAYS | PCT OF TASK COMPLETE |
|---|---|---|---|---|---|---|
| 1 | Kickoff Meetings | | | | | 100% |
| 2 | Hardware Status Update (33%) | | | | | 100% |
| 3 | Hardware Status Update (67%) | ME | | | | 100% |
| 3.1 | Order remaining materials | All | 02/14/25 | 02/21/25 | 7 | 100% |
| 3.2 | Begin wiring circuit | ME & EE | 02/14/25 | 02/17/25 | 3 | 100% |
| 3.3 | 3-D Print Other half of casing | Norma | 02/13/25 | 02/22/25 | 9 | 80% |
| 3.4 | Finish wiring | ME & EE | 02/18/25 | 02/23/25 | 5 | 70% |
| 3.5 | On/off | ME & EE | 02/18/25 | 02/23/25 | 6 | 0% |
| 4 | Website Check 2 | ME | 02/19/25 | 02/21/25 | 3 | 100% |
| 5 | UGRADs | | | | | 0% |
| 5.1 | Draft of Abstract | | 02/25/25 | 02/28/25 | 3 | 20% |
| 5.2 | Final Edit of Abstract | | 03/03/25 | 03/05/25 | 3 | 0% |
| 5.3 | Register & turn in Abstract | | 03/06/25 | 03/06/25 | 0 | 0% |
| 6 | Product Testing | | | | | 0% |
| 6.1 | Draft Testing Plan | All | | | | 0% |
| 6.2 | Finalize Testing Plan | All | | | | 0% |
| | | | | | | 0% |



# Thank You, Questions?