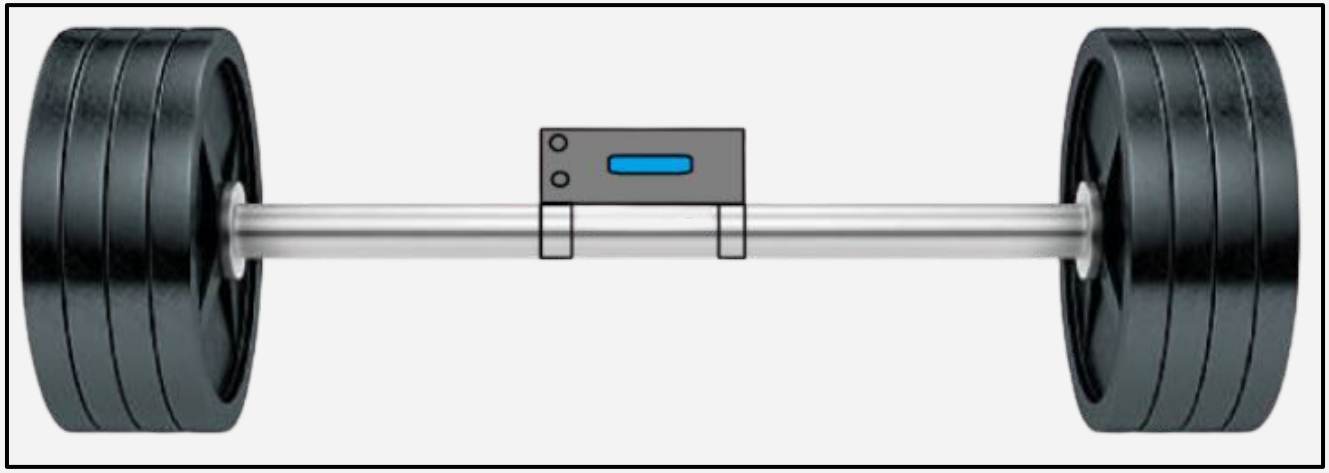
BarPro: A Weightlifting Aid Device

Group 28
Patrick Fejkiel, Kevin Mienta, and Greg Gruba
ECE 445 Senior Design
December 3, 2020

Introduction



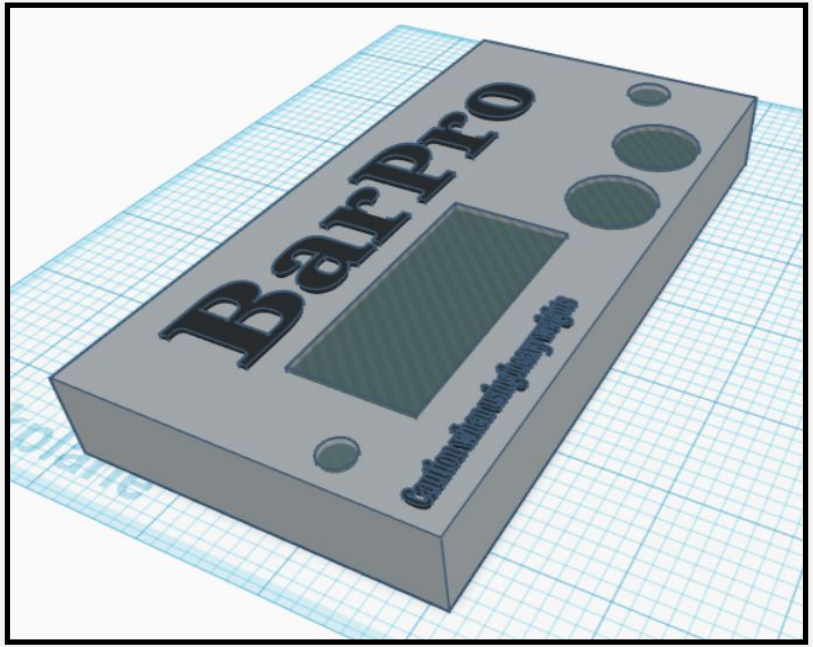
A weightlifting aid device to help lifters keep their barbell level, count reps/sets and check for proper form.

Background

- Uneven barbell positioning could result in serious injuries
- Uneven positioning causes uneven weight distribution on muscles and joints such as shoulders during the bench press exercise
- Exercises such as the bench press and deadlift require full motion to activate the desired muscle groups

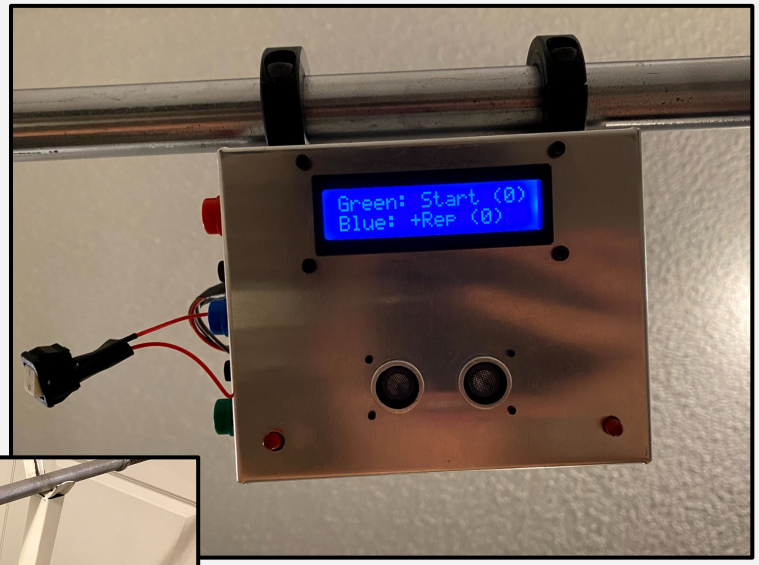
Features

- Count repetitions of motion and sets during a workout and display to LCD screen
- Notify user if barbell is not level with buzzer and LEDs
- Assist the user in performing full motion of exercise during a workout



Original Concept

How to Use the BarPro



Buttons

- Red: Reset reps/sets/calibration
- Blue: Add reps to the rep goal
- Green: Start the workout
- Switch: Power

How to Use the BarPro

Steps

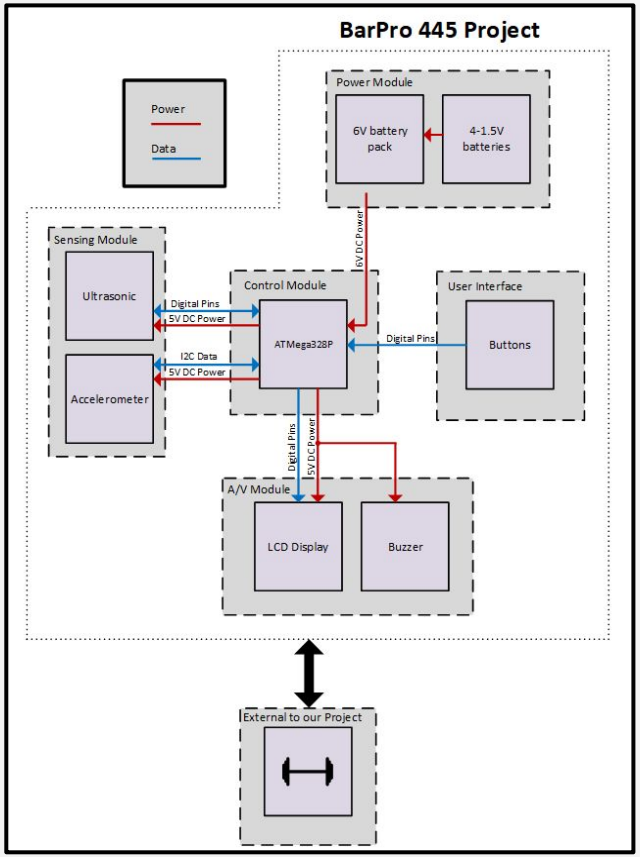
1. Get into workout position
2. Tap blue button repeatedly to set a rep goal
3. Press green button to start workout
4. Repeat for each set

Optional

- Reset device manually with red button tap
- Rep goal does not need to be set

Subsystems

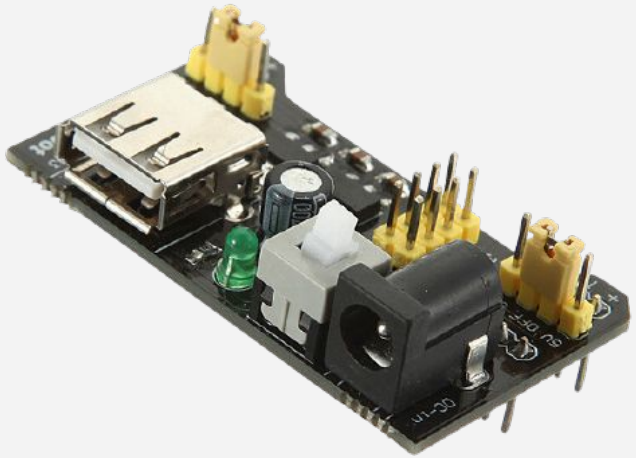
- Power Module
- Control Module
- Sensing Module
- User Interface Module
- Audio/Video Module



Block Diagram

Power Module

- A 3.3V/5V power supply powered by a 9V battery provides power to the various components in the device



Control Module

- An ATMega328P chip with digital and analog I/O pins to communicate with various components such as through I2C protocol with the accelerometer

User Interface Module

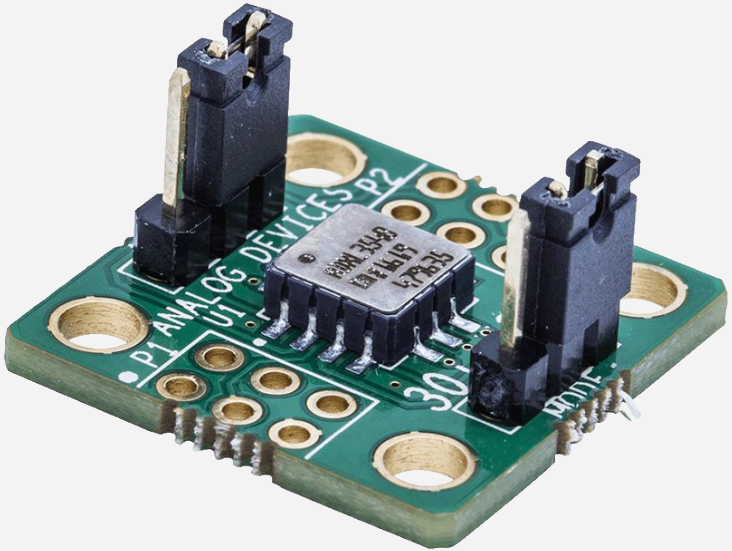
- Intuitive user interface with buttons and LEDs allowing the user to control the device and set reps/sets

Audio/Video Module

- Minimal delay between user input/response from project allowing for a seamless experience

Sensing Module: Accelerometer

- ADXL355Z chosen because of ease of integration with Arduino IDE using I2C protocol and its precise outputs



Sensing Module: Accelerometer (R&V)

Requirement

Accurately read the user's barbell tilt angle (+/- 3 degrees)

Verification

A user will tilt the barbell to the left and right side to see if the unlevel barbell tilt angle notification is given by LEDs and buzzer after 3 degrees.

Sensing Module: Accelerometer (Testing)

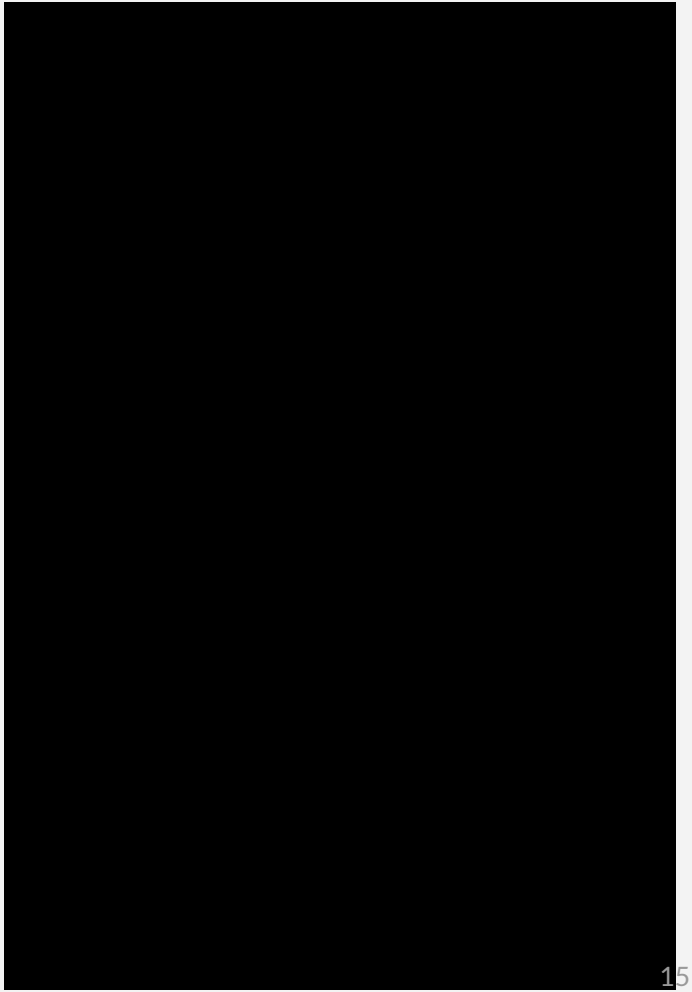
- Average difference between actual and calculated angles is 1.65°

Actual Angle (degrees)	Calculated Angle (degrees)
0	.53
2	2.8
4	5.29
6	6.74
8	9.85
10	11.7
12	13.66
14	15.73
16	17.75
18	20.02
20	22.18
22	24.02
24	25.9
26	27.3
28	28.95
30	31.3

Actual vs. Calculated Angles

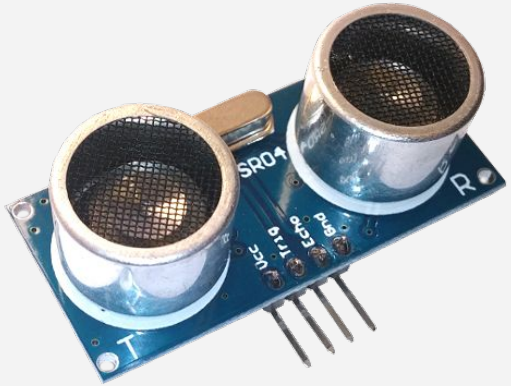
Sensing Module: Accelerometer (Demo)

- If BarPro is tilted to 3° or more beeping occurs
- Beeping frequency increases with tilt angle
- Real-time feedback throughout workout



Sensing Module: Ultrasonic Sensor

- HC-SR04 ultrasonic sensor used to measure the height of the workout in order to count reps and keep track of form



Sensing Module: Ultrasonic Sensor (R&V)

Requirement

Accurately measure the height of motion during a workout and count a rep if height is within 5cm of min or max height depending on workout

Verification

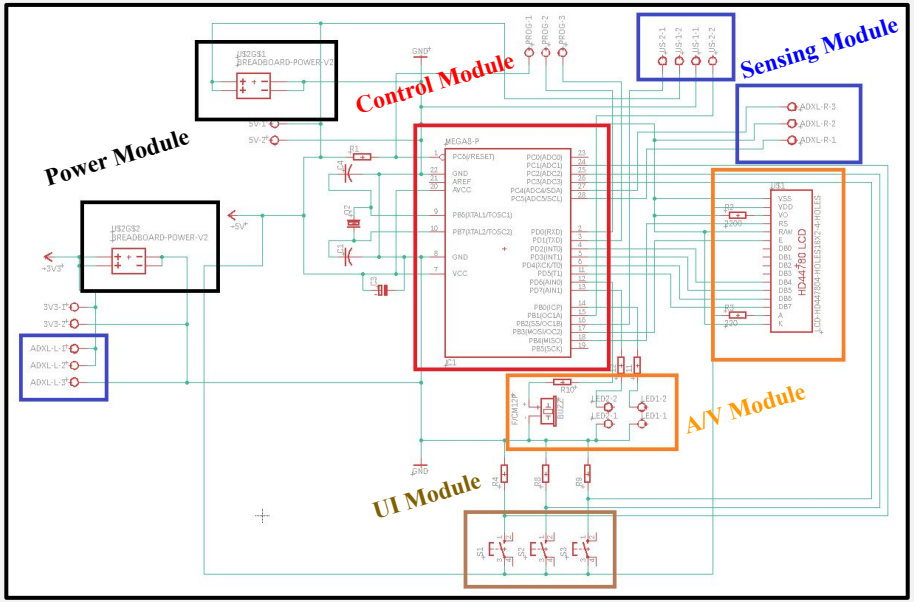
User does a full range motion exercise to set the min. and max. heights, and then does half of a repetition to see if no rep is actually counted

Sensing Module: Ultrasonic Sensor (Testing)

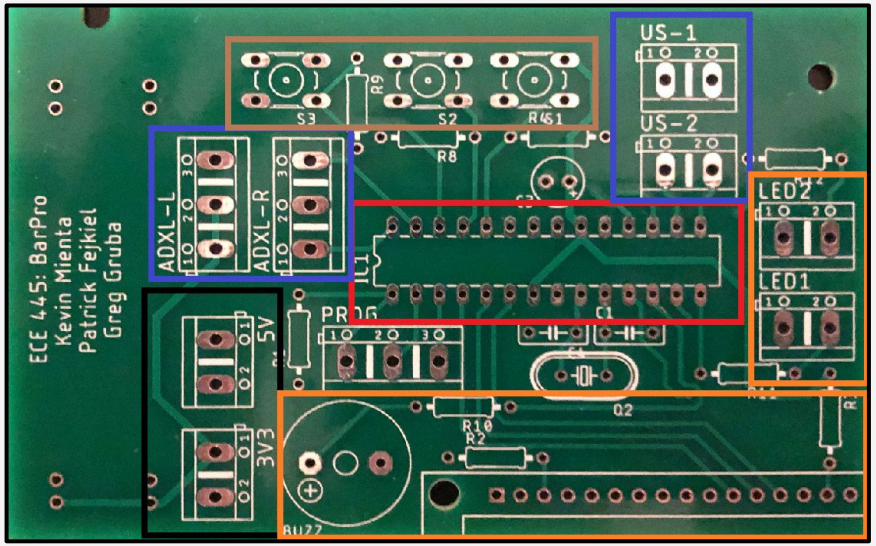
Actual (cm)	Measured (cm)	Percentage Error (%)
5	5.11	2.2
6	6.28	4.667
7	7.21	3
8	8.17	2.125
9	9.21	2.333
10	10.3	3
11	10.98	.182
12	12.11	.917
13	13.13	1
14	13.95	.357
		Avg. P. Error = 1.98

Actual vs. Measured Values

Circuit Schematic and Printed Circuit Board



Schematic



Assembled PCB

Software: getDistance()

```
double getDistance() { //returns averaged distance
  digitalWrite(trigPin, LOW);
  delay(10);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH, 10000);
  distance = (duration * .0343) / 2;

  if (distance < 70 && distance > 2) {
    distance_avg = 0;
    for (int i = 0; i < 9; i++) {
      dist_arr[i] = dist_arr[i + 1];
      distance_avg += dist_arr[i];
    }
    dist_arr[9] = distance;
    distance_avg += dist_arr[9];
  }
  return distance_avg / 10;
}
```

Software: getAngle()

```
double getAngle() {
  double SCALEFACTOR = 0.0000039;
  double values[3]; //array for holding raw, x-z data
  byte drdy; //data ready
  Wire.beginTransmission(ADXL);
  Wire.write(0x04);
  Wire.endTransmission();
  Wire.requestFrom(ADXL, 1);
  drdy = Wire.read();
  drdy = (drdy & (0x01));
  //get axis data from ADXL
  if (drdy == 0x01) {
    Wire.beginTransmission(ADXL);
    Wire.write(0x08);
    Wire.endTransmission();
    Wire.requestFrom(ADXL, 9, true);
    byte x1, x2, x3;
    for (int i = 0; i < 3; ++i) {
      x3 = Wire.read();
      x2 = Wire.read();
      x1 = Wire.read();
      unsigned long tempV = 0;
      unsigned long value = 0;
      value = x3;
      value <<= 12;
      tempV = x2;
      tempV <<= 4;
      value |= tempV;
      tempV = x1;
      tempV >>= 4;
      value |= tempV;

      if (x3 & 0x80) {
        value = value | 0xFFF00000;
      }

      values[i] = ((SCALEFACTOR * (long)value));
    }
```

```
x = 90 - (atan(values[2] / values[0]) * (180 / PI));
if (x > 90) x -= 180;

time_ = millis();
if (time_ - previous_time1 >= 200) {
  angle_tot += x;
  x_avg = angle_tot / count;
  count += 1;
  previous_time1 = time_;
}

time_ = millis();
if (time_ - previous_time2 >= 1000) {
  angle_tot = 0;
  count = 1;
  previous_time2 = time_;
}

return x_avg;
```

Software: buzzerLED()

```
void buzzerLED(double angle) {
  double offset = -1.5;
  double angle_offset = angle + offset;
  int wait = 0;
  if (angle_offset > 3) {
    wait = 500 / (int)angle_offset;
    digitalWrite(LED_left, LOW);
    digitalWrite(LED_right, HIGH);
    tone(buzzer, 400);
    delay(wait);
    digitalWrite(LED_right, LOW);
    noTone(buzzer);
    delay(50);
  }
```

```
  else if (angle_offset < -3) {
    wait = 500 / (int)angle_offset * (-1);
    digitalWrite(LED_right, LOW);
    digitalWrite(LED_left, HIGH);
    tone(buzzer, 400);
    delay(wait);
    digitalWrite(LED_left, LOW);
    noTone(buzzer);
    delay(50);
  }
  else {
    noTone(buzzer);
    digitalWrite(LED_left, LOW);
    digitalWrite(LED_right, LOW);
  }

}
```

Software: LCDdisplay()

```
void LCDdisplay() {
  if (mode == 0) { //0=Standby Mode
    lcd.clear();
    lcd.print("Green: Start (");
    lcd.print(sets);
    lcd.print(")");
    lcd.setCursor(0, 1);
    lcd.print("Blue: +Rep (");
    lcd.print(rep_goal);
    lcd.print(")");
  }

  else if (mode == 1) { //1=Maximum Height Mode
    lcd.clear();
    lcd.print("Checking Maximum Height");
  }

  else if (mode == 2) { //2=Minimium Height Mode
    lcd.clear();
    lcd.print("Checking Minimum Height");
  }
```

```
else if (mode == 3) { //3=Workout Mode
  lcd.clear();
  lcd.print("Workout Mode:");
  lcd.setCursor(0, 1);
  lcd.print("Reps:");
  lcd.print(reps);
  lcd.print(" Sets:");
  lcd.print(sets);
}

else if (mode == 4) { //4=End Screen Mode
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("    Rep Goal");
  lcd.setCursor(0, 1);
  lcd.print("    Complete");
  tone(buzzer, 523);
  delay(200);
  noTone(buzzer);
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("    Rep Goal");
  lcd.setCursor(0, 1);
  lcd.print("   <Complete>");
  tone(buzzer, 1319);
```

Cost Analysis

$$Labor\ cost = 3\ students * \$35/hr * 150 hours * 2.5 = \$39,375$$

$$ECE\ shop\ cost = \$35/hr * 4\ hours = \$140$$

$$Total\ BarPro\ cost = \$39,375 + \$140 + \$58.49 = \$39,573.49$$

Part	Price [$]	Quantity	Part #	Manufacturer
Atmega328P	2.08	1	32538KB	Microchip
LCD Display (HD44780	7.99	1		HiLetgo
ADXL 355Z	35	1	584-EVAL-ADXL355Z	Analog Devices
Ultrasonic Distance Sensor HCSR04	3.95	1	15569	Sparkfun
Buzzer	8.52	15 (1 required)	G306	GFORTUN
16MHz Crystal (Clock)	0.95	1	00536	Sparkfun
Total	58.49			

Ethics and Safety

- Does not store any personal data
- Only very light weight used for testing purposes
- Precautions are in line with rule #9 in the IEEE Code
- 9V batteries are alkaline instead of lithium

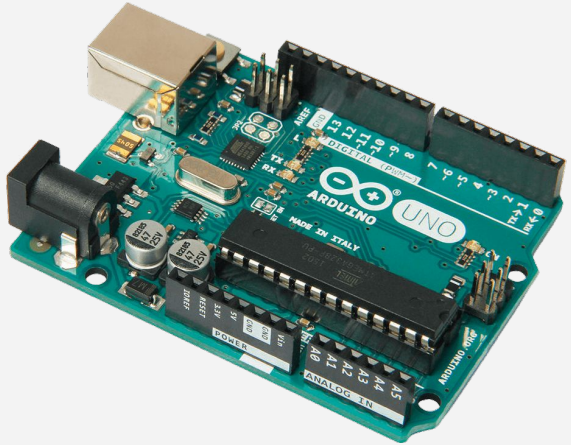
Successes and Challenges

Challenges

- Change in high-level scope
- Time-management
- Sensor issues

Successes

- All high-level requirements were met
- Optimized PCB



Recommendations for Further Work

- Find alternative cost-friendly components such as a different accelerometer
- Revise layout of components on PCB to minimize its size
- Create a smaller case in order to decrease weight and increase device appeal through aesthetics
- Research investment opportunities and filing for patents

Credits

- Anthony Schroeder
- ECE Machine Shop
- Professor Arne Fliflet
- Professor Jing Jiang
- Professor Rakesh Kumar
- Professor Michael Oelze
- Professor Jonathan Schuh
- Professor Casey Smith
- Professor Gary Swenson
- Course TAs

Thank You

Questions?