BarPro Weightlifting Aid Device

By

Grzegorz Gruba

Kevin Mienta

Patrick Fejkiel

Final Report for ECE 445, Senior Design, Fall 2020

TA: Anthony Schroeder

09 December 2020

Project No. 28

Abstract

BarPro is a weightlifting aid device designed to help weightlifters keep their barbell level, count repetitions/sets and check for proper form. This device, which is housed in a metal case, is strapped onto a barbell and configured by the user with buttons to choose a repetition goal. There is an LCD screen on the device that indicates various workout information to the user. The device uses an accelerometer to measure the barbell’s angle of tilt, an ultrasonic sensor to measure the distance from the barbell during repetitions and a microcontroller to read digital inputs from these sensors and output feedback to the user with LEDs, a buzzer and an LCD screen. BarPro currently meets all its set requirements by reading a user’s barbell tilt angle within 3 degrees, counting reps within +/- 1 repetition, and counting repetitions and sets only if a full motion of workout repetition is completed.

Contents

1. **Introduction**

1.1 Objective 5

1.2 Background 5

1.3 High-Level Requirements 7

2. **Design**

2.1 Block Diagram 8

2.2 Subsystems 9

2.2.1 Power Module 10

2.2.2 Control Module 11

2.2.3 Sensing Module 11

2.2.4 User Interface 12

2.2.5 A/V Module 12

2.3 Requirements and Verifications 12

2.3.1 Verification Justifications 13

2.4 Tolerance Analysis 14

2.5 Software 15

2.6 Risk Analysis 16

2.7 COVID-19 Contingency Planning 16

3. **Cost and Schedule**

3.1 Cost Analysis 17

3.2 Schedule 17

4. **Conclusion**

4.1 Challenges 18

4.2 Accomplishments 18

4.3 Ethical Considerations 18

4.4 Future Work 19

References 20

Appendix A Requirements and Verifications Table 21

Appendix B Verification Justifications 23

Appendix C Key Software Functions 26

Appendix D Cost and Schedule 28

1 Introduction

1.1 Objective

A common issue among weightlifters, regardless of experience, is bad form when doing exercises. This occurs from a beginning lifter not understanding the correct motion or an experienced lifter wearing out when reaching the end of his or her set. Muscles begin to fatigue and when exercises are performed with two hands (bench press, squats, deadlifts, for example), the stronger hand compensates for the other and the bar can become unlevel; this leads to asymmetrical strain on the body. Mitigation of this issue is possible if the user is doing a workout on a Smith machine or is partnered with a spotter; but these machines have their drawbacks, causing different muscle activation than a free weight barbell [1]. Also, with the ongoing Covid-19 atmosphere and the difficulty of consistently aligning gym schedules between people, users more often now than ever find themselves at the gym alone.

A weightlifting device called BarPro, shown in Figure 1, is designed for people who do barbell exercises such as bench press and deadlift. Many weightlifters, especially beginners, have a problem with keeping the barbell level while doing their repetitions which can lead to serious injuries. Many weightlifters also do not complete full movements of their repetitions, especially at the end of their sets when muscle fatigue is forming. The BarPro solves these problems by checking if the barbell is level and notifying the lifter if it is not. It also allows the lifter to calibrate the minimum and maximum heights of their lifts and notifies them if they are not doing their full repetition. Finally, the device also keeps track of repetitions and sets so the lifter can focus more on their workout.

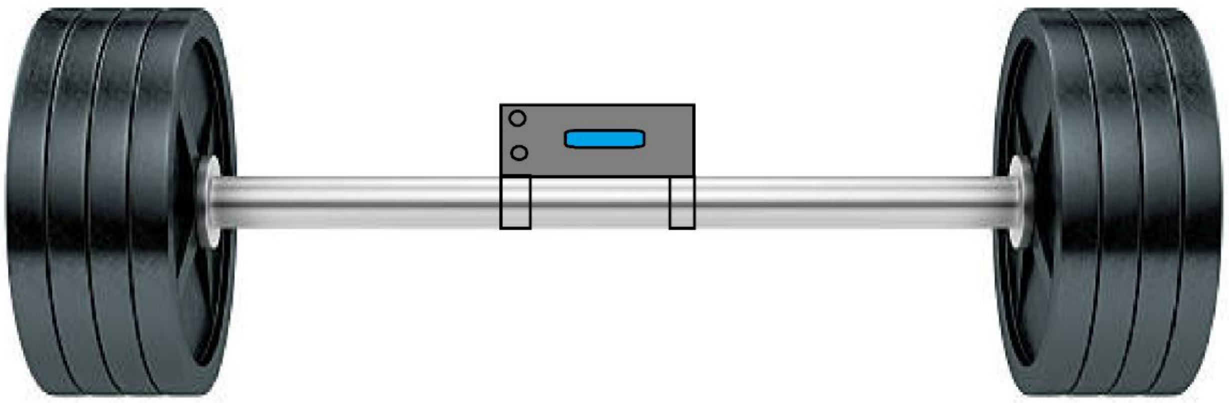


Figure 1: BarPro Physical Design Overview

1.2 Background

Uneven barbell positioning could result in serious injuries, especially when lifting heavy weights. This uneven positioning causes uneven weight distribution on muscles and joints such as shoulders [2] during the bench press exercise. To reduce the risk of these injuries, a weightlifter has to make sure he or she is keeping his or her barbell level at all times. Although a weightlifter can sometimes notice an uneven barbell by him- or herself, oftentimes weightlifters go through an entire motion with an uneven barbell unnoticed. A spotter or partner can definitely notice an uneven barbell, but many gym-goers work out alone, especially now during the COVID-19 pandemic where social distancing measures are enforced. The BarPro device has the ability to serve as a spotter or partner and notify the weightlifter when the barbell is uneven. It performs even better than a human being by using an accelerometer to know exactly when a barbell is not positioned evenly.

Exercises such as the bench press and squat require full motion to activate the desired muscle groups. When full motion is not completed by the weightlifter, maximum efficiency is not reached from the workout and some muscles may undergo insufficient usage. To eliminate the possibility of doing exercises with incomplete motion, weightlifters should practice proper form with little weights to create the muscle memory needed for a proper lift. The BarPro device can aid in this process by allowing the user to calibrate his or her full range of motion for the workout and providing a notification if the user is not completing a full repetition of motion by using an ultrasonic sensor.

Finally, keeping track of repetitions and sets is an important factor of every weightlifting session. Different repetitions and sets are executed by the weightlifter depending on what his or her work out goal is. Some people want to build muscle strength and size so they stay in the lower range of repetitions, while others want to build muscle endurance and train with higher repetitions. It is actually recommended to train with less weight and more repetitions if reducing the risk of injuries is desired [3]. Although mentally keeping track of repetitions/sets seems like a fairly easy process, lifting heavy weights for a long duration of time does lead to fatigue that may cause a weightlifter to forget what set he or she is on or how many reps he or she has completed. BarPro has the ability to keep track of these repetitions and sets so the user can focus solely on lifting the weight and keeping the barbell level.

1.3 High-Level Requirements

- Accurately count repetitions of motion during a workout (+/- 1 rep): This is done using an ultrasonic sensor. An intuitive user interface with buttons and LED display allows the user to see the repetitions/sets and reset them using the buttons. These reps/sets are displayed on an LCD display.

- Accurately read the user's barbell tilt angle (~30°) established during initial testing: This is done using an accelerometer. The two LEDs display which side is unlevel and a buzzer sounds with a frequency corresponding to the level of the bar at varying increments.

- Measure the height of motion (+/- 2cm): This is done during a workout using an ultrasonic sensor. If the user is not performing full repetitions, then their reps/sets are not counted on the LCD display.

2 Design

2.1 Block Diagram

The BarPro requires five main component areas to fulfill desired operation as shown in Figure 2. These include the power module, sensing module, control module, user interface and LEDs. The power module supplies the 9V to run the components of the device. The sensing module contains the accelerometer and ultrasonic sensors needed for device operation. The control module sends and receives data to control various device components. Lastly, the user interface contains the buttons and LEDs for the user to read and also provide input to the device.

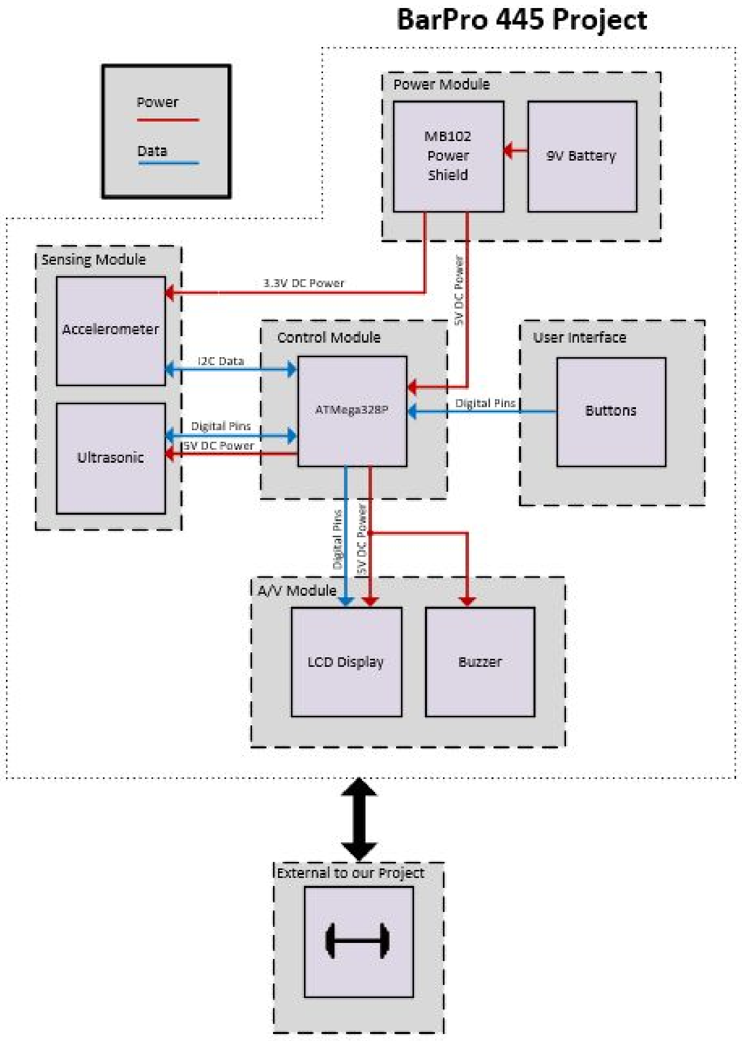


Figure 2: BarPro Device Block Diagram

2.2 Subsystems

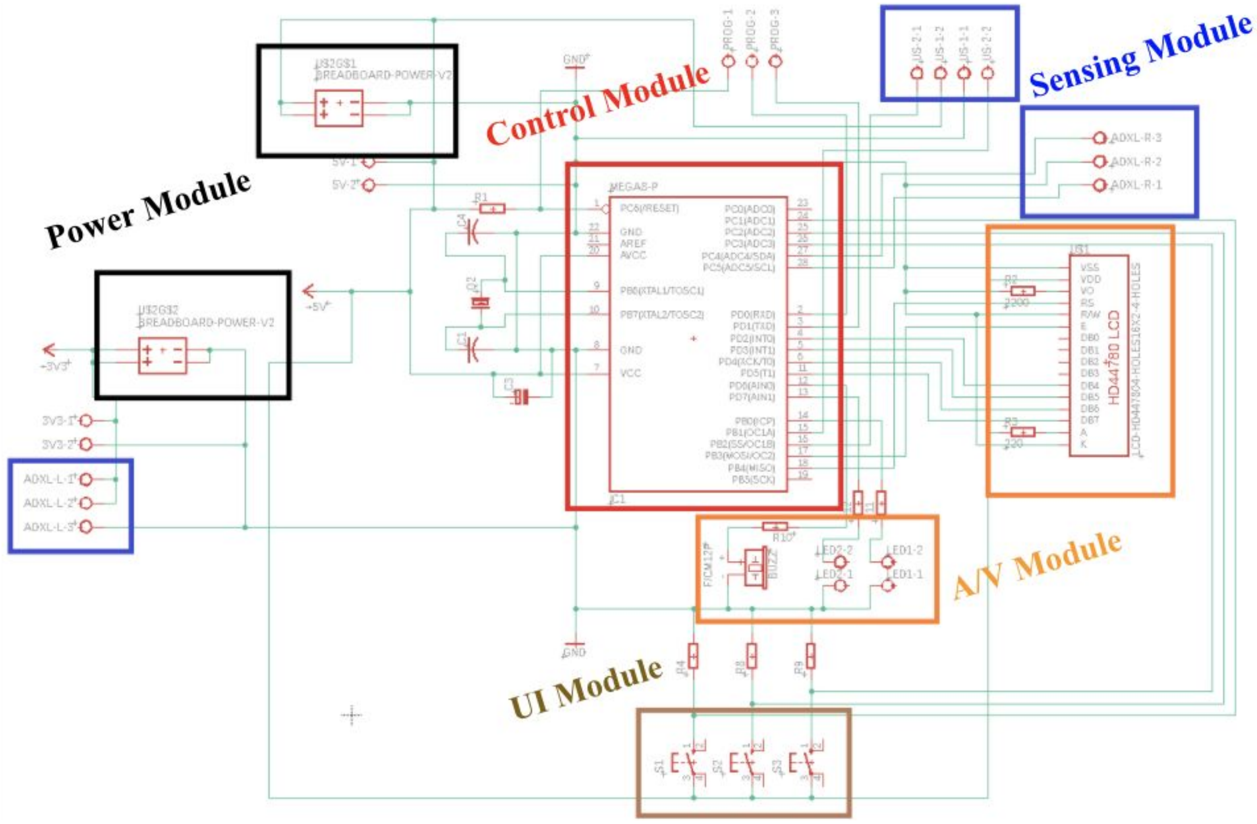


Figure 3: Circuit Schematic of BarPro Device

The circuit schematic in Figure 3 conveys our entire BarPro project. We have our ATMega328P chip at the core. Connected to it are our ultrasonic sensors, an accelerometer, and LCD display. Outputted are a buzzer and LEDs. The rest of the circuitry is required to have proper ATMega328P functionality.

Figure 4 details an image of our physical design, a box roughly 3in by 6in that holds our PCB. A rectangle is carved out of the center for the LCD display that can be seen turned on. There are holes on the bottom corners housing the LEDs. The larger holes in the center are for the ultrasonic sensor. The three buttons to the left of the device are for the user interface along with the black and white power button that is extended out for demonstration purposes. The black clamps use screws to attach the entire device to a barbell.

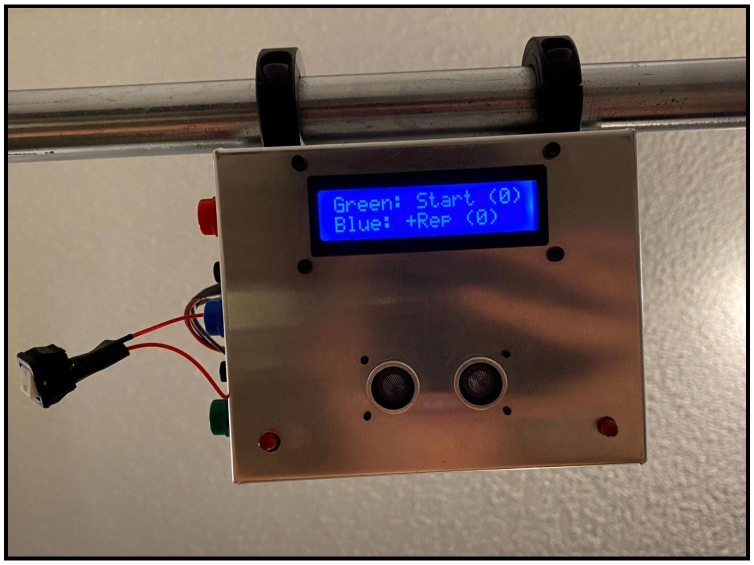


Figure 4: BarPro Physical Device Housing

There were a few challenges that guided design choices that we made throughout the semester. The first significant challenge we ran into was switching from an Arduino Uno as the center of our control module to a microcontroller. This was due to a change in high-level scope of our project. The next small change we implemented was updating our power module from a 6V AA battery pack to a single 9V battery and a DC/DC converter on our PCB. This change made space more efficient within our device and allowed for the nominal 5V DC input into our microcontroller. Another point worth mentioning was our decision to adhere to our $100 budget as closely as possible. A design issue related to our sensors occurred when we were balancing the cost of our sensors and the complexity of running the software to compensate for their low tolerances. Since our project did not have to be extremely precise, we opted for purchasing cost-friendly sensors and programming averaging functions to interpret them. These points will be elaborated below and in our conclusion.

2.2.1 Power Module

This module provides the power for the rest of our system. Since we are not working with high voltages or magnetism, this module is simplified to strictly low voltage DC power. The Arduino needs a range of 5-12V to operate, and we had a 9V battery clip in one of our supply kits, so that was the reason for using this type of battery. This type of battery was also chosen for minimizing overall space in the physical housing case. For example, a 6V battery clip using 4 1.5V AA batteries takes up significantly more surface area than 1 9V battery. A 9V battery is connected to a 3.3V/5V power supply module that distributes the 3.3V and 5V to the necessary components on our device. There is no data transfer between the power module and the rest of the system, so only two wires leave this module.

2.2.2 Control Module

The control module is the entire brain of our system consisting of an ATMega328P chip. This microcontroller was chosen because it is through-hole for easy soldering onto our PCB. It was also chosen for cost efficiency since the amount of pins were maximized on this microcontroller, and if we used a different microcontroller such as an ATMega2560, it would be more expensive and not all pins would be used. This module receives 5V DC power from the power module and distributes it to the rest of the modules that require it. Regarding data, the digital IO pins are used for the majority of our system. Every other module is connected through digital pins, with the exception of the accelerometer that communicates with the control module through additional I2C protocol. It has a strict input from user interface, strict output to our LEDs, and input/output to our sensing module.

2.2.3 Sensing Module

Our sensing module is composed of two sensors: the ADXL355Z accelerometer and HC-SR04 ultrasonic sensor. This type of accelerometer was used because of prior project member research experience incorporating this sensor with software. The HC-SR04 ultrasonic sensor was used because of how readily available it is and because several of these sensors were also available to us from previous course lab kits. Both of these sensors were also easily integratable with Arduino's IDE platform. The sensing module is where the reps a user is doing are counted, and the movement/level is tracked and sent back to the control module for interpretation. The sensors in the sensing module are wired in parallel to the control module receiving 5V DC power. In terms of data transfer, the sensors receive a data request from the control module and send back data, one sensor after the other. Both sensors communicate through digital IO pins, with additional I2C protocol for the accelerometer.

2.2.4 User Interface

This module consists of buttons allowing the user to interact with the control module. It has a calibration button for beginning a workout, a button for counting reps, and a reset button. The tactile push momentary buttons that were chosen were used specifically for their cost-friendliness and easy integration into existing hardware.

2.2.5 A/V Module

This module primarily allows the user to see the level of the bar in real-time and hear the buzzer when the barbell is unlevel. It uses two red LEDs on either side to illuminate the level of the bar to the user. The LEDs blink a red light at different frequencies depending on the tilt of the barbell. The buzzer is activated when the barbell is unlevel to notify the user and also sounds at different frequencies depending on the tilt of the barbell. Specifically, the HD44780 LCD was chosen because it was already readily available from previous course lab kits, and it was easily integratable into existing hardware using Arduino's IDE platform.

2.3 Requirements and Verifications

A requirements and verification Table 1 is located in Appendix A. The main six requirements of our device consist of control, sensing (accelerometer), sensing (ultrasonic), user interface, power and A/V modules. Verifications included in Table 1 provide a clear description of each verification procedure. This table also contains the verification status for each requirement. The BarPro device passed or exceeded all of the verifications and they were justified using a quantitative analysis included in the next section and in Appendix B.

2.3.1 Verification Justifications

The control module verification was justified by performing various sets of repetitions during device testing. Every set was performed with a different amount of repetitions chosen, from one to ten, and the number of actual reps completed was compared with the reps shown on the LCD screen. The reps actually completed and reps shown on the BarPro device were equal throughout the test, and this provided justification for the control module requirement being verified. Results of this justification procedure are included in Table 2 in Appendix B.

The sensing module's requirement verification for the accelerometer sensor was justified using an iPhone's level sensing feature to check if the device has accurate angle of tilt readings within 3 degrees. Throughout the test, angles of tilt were increased from 0 to 70 degrees, and the actual angles were compared with the angles calculated from the accelerometer's acceleration values. This module's requirement verification for the ultrasonic sensor was justified using a ruler to compare the actual distance of the BarPro from an object and the distance measured by the ultrasonic sensor. Both of these tests were also a part of the tolerance analysis for BarPro and described in further detail in the following section. The justification tests for the accelerometer and ultrasonic sensor showed that the actual angle of tilt vs. calculated angle of tilt was always within 3 degrees and the actual distance of BarPro from an object vs. measured distance was at a maximum percentage error of 1.98%. This fulfills the requirement of having BarPro operate within 5 cm of minimum or maximum height because during a workout such as the bench press, maximum height is around 50 cm, so 1.98% of 50 cm is .99 cm and significantly less than 5 cm. Results of these tests are included in Tables 3 and 4 in Appendix B

The user interface requirement verification was justified by performing a test for the usability of each button on the BarPro device. Each button was pressed, and a verification was written down if the button worked or not. These buttons included the 3 functionality buttons of reset, rep count and start workout along with the power button. The barbell was also tilted to one side and the other to see if the buzzer and LED's properly functioned. All of the buttons along with the buzzer and LED performed their desired functionalities. Results for these components' functionality are included in Table 5 in Appendix B.

The power module's verification justification was performed using a multimeter to measure voltages on the 3.3 V and 5 V power supply connections to see if they were within the desired range specified in Table 1. The voltage measurements were all within range and included in Table 6 in Appendix B.

Finally, the A/V modules verification justification was performed by using a timer to check if the time between a user input such as an unlevel barbell position and notification from the device in the form of LED/LCD display and buzzer is less than 1 second. As can be seen in Table 7 in

Appendix B, all the durations between a user's input and device's output were less than 1 second so the verification was successful.

2.4 Tolerance Analysis

Quantitative analysis was required on the tolerance of the accelerometer. The accelerometer reads acceleration values corresponding the level of tilt of the barbell. These acceleration values are converted into degrees of tilt using Eq. 1 And Eq. 2, in order, along with Figure 5.

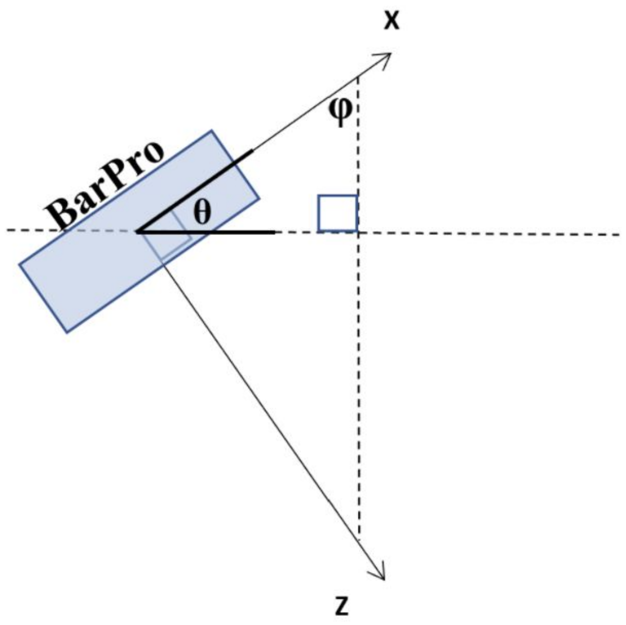


Figure 5. Angle of Tilt

$$\textit{angle of tilt}\ (\textit{degrees}) = 90 - \arctan(z_{acceleration}/x_{acceleration}) * (180/pi) \qquad \text{(Eq. 1)}$$

$$\textit{if angle of tilt} > 90,\ \textit{then angle of tile} = \textit{angle of tilt} - 180 \qquad \text{(Eq. 2)}$$

A tradeoff was needed to be made between desired maximum tilt angle and acceleration values depending on how closely a relationship could be calculated between degree of tilt and m/s^2 of acceleration. Table 3 in Appendix B shows the magnitude of difference between actual angle measured using an angle tilt measuring device and angle calculated by the acceleration values from the accelerometer.

Table 3 data reflects an average difference in angle of 1.65 degrees. This average difference in angle of 1.65 degrees was then multiplied by 2 in order to roughly give the accepted range in barbell tilt angle of 3 degrees mentioned in requirement #2 in Table 1 located in Appendix B. A multiple of 2 was chosen because the maximum angle difference value seen was 2.8 degrees so 3 degrees would be a reasonable maximum value range.

The ultrasonic sensor was another module that required quantitative analysis on the tolerance. A ruler was used to measure actual distance values between the sensor and an object, and this actual distance was compared with the distance measured by the ultrasonic sensor. An average percentage error of 1.98% was found as can be seen in Table 4 in Appendix B.

2.5 Software

Software integration was a key aspect in successful operation of the BarPro device. The main component of BarPro's software is a centralized state machine consisting of the following states: main menu, checking maximum height, checking minimum height and workout. In order to incorporate the desired functionality of various sensors into the code, several important functions had to be implemented. Out of all the functions included in BarPro's software, 4 main functions are key in device operation: getDistance(), getAngle(), buzzerLED() and LCDdisplay(). Screenshots of these functions are included in Appendix C along with descriptions in the following text.

The first key function in BarPro's software is the getDistance() function that is essentially the main piece of software/hardware integration with the ultrasonic sensor. Distance values from the ultrasonic sensor are added into an array of 10 values and an average distance value is constantly computed from this array. Distance values beyond reasonable limits are not placed in the array in order to avoid skewed average distance values not representative of actual distances during the workout.

The next key function is the getAngle() function that allows the accelerometer sensor to be integrated into the BarPro system using the I2C protocol. The function also computes tilt angles using Eq.'s 1 and 2 and computes a running average of the tilt angle. This average is updated every 200 milliseconds and reset if a new angle measurement is not recorded within 1 second.

Finally, the last 2 key functions in software are the buzzerLED() and LCDdisplay() functions. They are essential in displaying device output to the user. The buzzerLED() function computes a wait time depending on the angle of tilt of the barbell. A greater angle of tilt corresponds to a smaller wait time, and this wait time is used as a frequency setpoint for the buzzer sounding on/off and the LEDs blinking. The LCDdisplay() function allows for data to be displayed on the LCD screen such as the current mode, reps completed and sets completed. The 5 main display modes are main menu, setting maximum height, setting minimum height, workout and reps completed.

2.6 Risk Analysis

The greatest threat to a successful final project was a malfunctioning sensing module. The entire system relied on the accuracy of the sensors within it to send back accurate information to the control module for correct parsing and data interpretation. The user interface is very difficult to not use correctly because it is primarily passive buttons. Faulty LEDs are easily replaceable and programming them comes very easy. The control module accepts a wide range of voltages (5-12V), so the power module providing the wrong power is not a big issue. Microcontrollers are very reliable, but the control module can malfunction if the code implemented on ours is not optimized or written correctly. This is an issue, but regardless if our sensors do not have a high tolerance, our code (whether written well or not) will not work anyway. This is how we decided that our sensor module is the riskiest module to combine into our system.

2.7 COVID-19 Contingency Planning

If the University transferred to all online classes due to COVID-19, the overall project would not change at all. The team members all have physical access to each other while following COVID-19 guidelines, and testing items such as a barbell and bench are available without going to a gym. If component shipments were cancelled or delayed due to COVID-19, similar and readily available components were used to make the BarPro device function to meet high-level requirements.

3 Cost and Schedule

3.1 Cost Analysis

Assuming an average hourly pay of \$35/hour, the total labor cost is calculated to be \$39,375 in Eq. 3. The estimated ECE shop cost will be \$140 as calculated in Eq. 4. Total price for the BarPro project is \$39,573.49 using Table 8's total part cost located in Appendix D and Eq. 5.

$$Labor\ cost = 3\ students * \$35/hr * 150 hours * 2.5 = \$39,375 \qquad \text{(Eq. 3)}$$

$$ECE\ shop\ cost = \$35/hr * 4\ hours = \$140 \qquad \text{(Eq. 4)}$$

$$Total\ BarPro\ cost = \$39,375 + \$140 + \$58.49 = \$39,573.49 \qquad \text{(Eq. 5)}$$

3.2 Schedule

Overall, the project schedule stayed consistent with what was initially planned. Table 9 in Appendix D contains a general layout of the schedule that was followed by the BarPro team in the months of October and November 2020. The initial weeks of October consisted of finalizing a parts list order and starting to work on software integration. The ending weeks of October consisted of designing the physical aspects of the device such as the housing case and component layout. The beginning weeks of November consisted of the physical build of the device such as soldering all the components. Finally, the middle weeks of November consisted of finalizing the successful operation of BarPro for the final demonstration.

4 Conclusion

4.1 Challenges

The biggest challenge that was faced during the project was a change in high-level scope during the design phase. Instead of using an Arduino UNO as the base of the control module, it was decided that an ATmega328P microcontroller would be used as the main component in the module. This design change was made to not only increase complexity of the hardware portion of our project, but also decrease physical space on the PCB as an ATmega328P is only a fraction of the size of a whole Arduino UNO. This design change required a complete redesign of our circuit schematic, involving the addition of additional components for the necessary circuitry to substitute an Arduino UNO for the ATmega328P.

Another challenge involved sensor issues during the testing phase of our project. The sensors would not always read accurately, and random spikes in measurements would cause our device to not function properly. This was fixed using modifications in our code such as a queue for the distance measurements and running averages for the sensor data to always have an accurate data reading that the device could use to output feedback to the user.

4.2 Accomplishments

The greatest accomplishment of BarPro is that all high-level requirements were met, along with individual requirements for the modules. The device has full functionality, is housed in a physical case and can be attached to a barbell to be used in an actual workout. The PCB is optimized to take up little space resulting in an overall package that is user-friendly.

4.3 Ethical Considerations

The product that emerges from completing this project has the possibility of providing a fantastic aid to the general public that enhances workouts and body fitness goals. This also comes at the cost of possible misuse causing safety issues. We do not recognize any ethical issues with the BarPro product. It is simply a device to aid beginners on their workout journey by stopping bad habits with form. It does not store any personal data that could be misused such as user profile information. To avoid safety issues regarding the BarPro product, only very light weight is used for testing purposes. Using light weights will minimize the possibility of injury during testing. When the BarPro is used at the gym, individuals need to make smart decisions about the weight they use in workouts. The BarPro is only present to ensure proper motion and data tracking, it will not stop a careless individual from pushing his or her limits too far. This product will have warnings regarding the use of heavy weights designed to steer individuals in the right direction and promote safety measures if ever released to the public. These precautions are in line with rule #9 in the IEEE Code of Ethics regarding the requirement to never injure individuals [4]. Our product also uses 9V batteries from a reputable manufacturer such as Energizer so they meet US Consumer Product Safety Commission regulations, but as with any battery, there is always a possibility for an explosion or fire. The batteries used are alkaline instead of lithium to decrease this possibility, and alkaline batteries are also easier to dispose of by the user.

4.4 Future Work

BarPro is a device that has true potential to be an aid to individuals in the fitness industry. Additional functionality can be implemented in the future based on weightlifters' desires, and further testing and modifications can hopefully guarantee foolproof operation of this device to a customer base. Currently, the most important thing to consider in future work aspirations for BarPro is making the device more affordable by using less expensive components. The accelerometer alone is around $35 so finding cheaper alternatives can lead to a more attractive price for the device. The layout of components on the PCB can be optimized even further to create a smaller physical housing case in order to decrease weight and increase device appeal through aesthetics. Finally research can be conducted on investment opportunities to sponsor the device development along with filing for patents.

References

[1] Schwanbeck, S., Chilibeck, P. and Binsted, G., 2020. “A Comparison Of Free Weight Squat To Smith Machine Squat Using Electromyography,” *A Comparison Of Free Weight Squat To Smith Machine Squat Using Electromyography,* Dec. 2009. Accessed Sept. 30, 2020. [Online]. Available: 10.1519/JSC.0b013e3181b1b181

[2] D. Hatfield, *Shoulder Injuries,* ISSA, 2020. Accessed on: Sept. 18, 2020. [Online]. Available: https://www.issaonline.com/blog/index.cfm/2011/2/11/shoulder-injuries

[3] Loyola University Health System, *Military Surgeons Report 'Alarming Frequency' Of Bench Press Injuries*, ScienceDaily, March 2018. Accessed on: Sept. 18, 2020. [Online]. Available: https://www.sciencedaily.com/releases/2018/03/180322124948.htm

[4] IEEE, *7.8 IEEE Code Of Ethics,* IEEE Policies, Section 7 - Professional Activities (Part A - IEEE Policies) 7.8. Accessed on: Sept. 18, 2020. [Online] Available: https://www.ieee.org/about/corporate/governance/p7-8.html

Appendix A Requirements and Verifications Table

Table 1: Requirements and Verifications

Requirements	Verification	Verification Status (Y or N)
1) Control Module: Accurately count repetitions of motion during a workout (+/- 1 rep)	1) A user will do full repetitions of motion during various workouts and compare actual rep counts to rep count shown on the LCD screen to check if it is within the required range.	Y
2) Sensing Module: Accurately read the user's barbell tilt angle (+/- 3 degrees)	2) An experienced user will purposely tilt the barbell with no weight to the left and right side to see if the unlevel barbell tilt angle notification is given by LEDs and buzzer after 3 degrees. This angle measurement will be verified by either a smartphone using its level of tilt angle detecting feature or Arduino IDE's serial display to see if it is within required range.	Y
3) Sensing Module: 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	3) An experienced user will do a full range motion exercise to set the min. and max. heights, and then the user will do half of a repetition to see if no rep is actually counted since a full rep was not completed and the	Y

workout	actual height measured during the repetition was less than the min. and max. range of motion. The current height measured will also be displayed on Arduino IDE's serial display and compared with the min and max heights to test if a rep is counted if current height is within 5cm of min and max height.	
4) User Interface: Intuitive user interface with buttons and LEDs	4) A weightlifter will do a workout with the device, pressing each button to test functionality such as total reset and set increment / rep reset and checking if LED/buzzer signal is given for uneven barbell. This includes testing the on/off power button to check if the device turns on and off.	Y
5) Power Module: 9V battery will power the Power Supply Module that will provide a voltage in the range 2.8-3.8V and 4.5-5.5V, respectively, for the 3.3V/5V power supply connections on the board	5) A multimeter will be used to measure the voltage on the board on both 3.3V and 5V power supply connections on the board to make sure it is within requirement range.	Y
6) A/V module: Minimal delay between user	6) A timer will be used to check if the time between a user input	Y

input/response from project	such as an unlevel barbell position and notification from the device in the form of LED/LCD display and buzzer is less than 1 second.	

Appendix B Verification Justifications

Table 2: Control Module Justification (Actual Reps vs. Reps Shown)

Reps Physically Completed	Reps Shown on BaPro
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	10

Table 3: Sensing Module Accelerometer Justification (Actual Angle vs. Calculated Angle)

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
40	41.3
50	52.7
60	62.4
70	72.8

Table 4: Sensing Module Ultrasonic Justification (Actual Distance vs. Measured Distance)

Actual Distance (cm)	Measured Distance (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

Table 5: User Interface Module Justification (Component Functionality)

Component	Performs Function? (Y or N)
reset button	Y
rep count button	Y
start workout button	Y
on/off button	Y
buzzer	Y
LEDs	Y

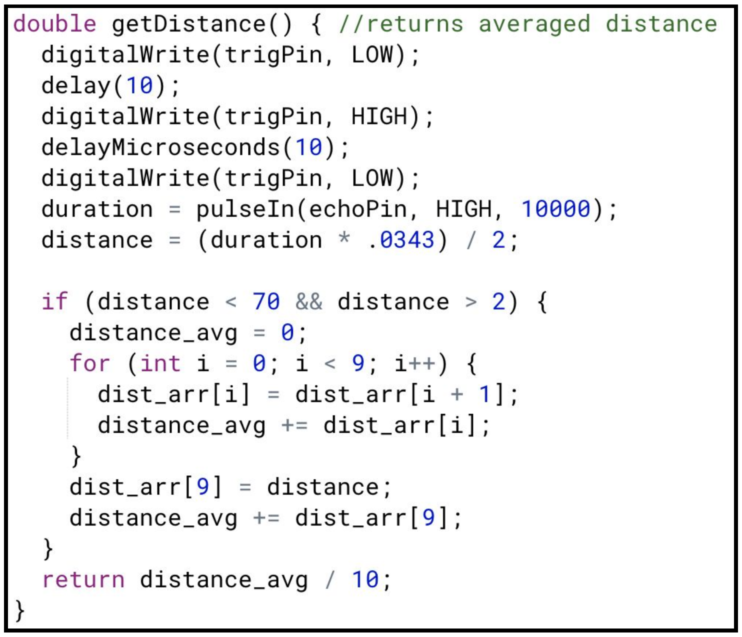
Table 6: Power Module Justification (Voltage Measurements)

Power Supply Connection (V)	Measured Voltage (V)
3.3	3.3
5	5

Table 7: A/V Module Justification (User Input/Device Output Time)

User Input	Device Output	Time (s)
button press	LCD screen notification	.1
barbell tilt	Buzzer sound	.4
barbell tilt	LED blink	.4
rep completion	LCD screen notification	.5

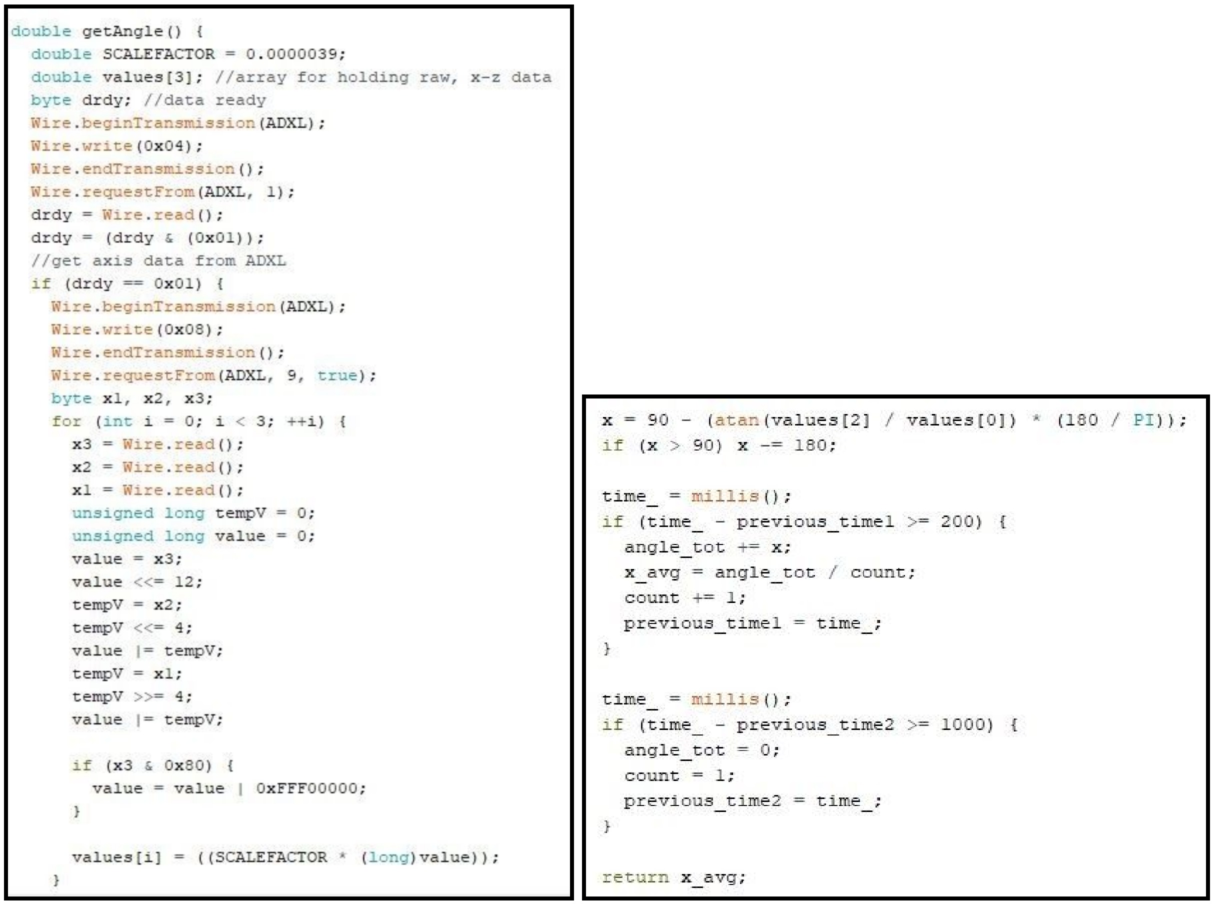
Appendix C Key Software Functions



```
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;
}
```

Figure 6. getDistance() Function



```
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));
    }
```

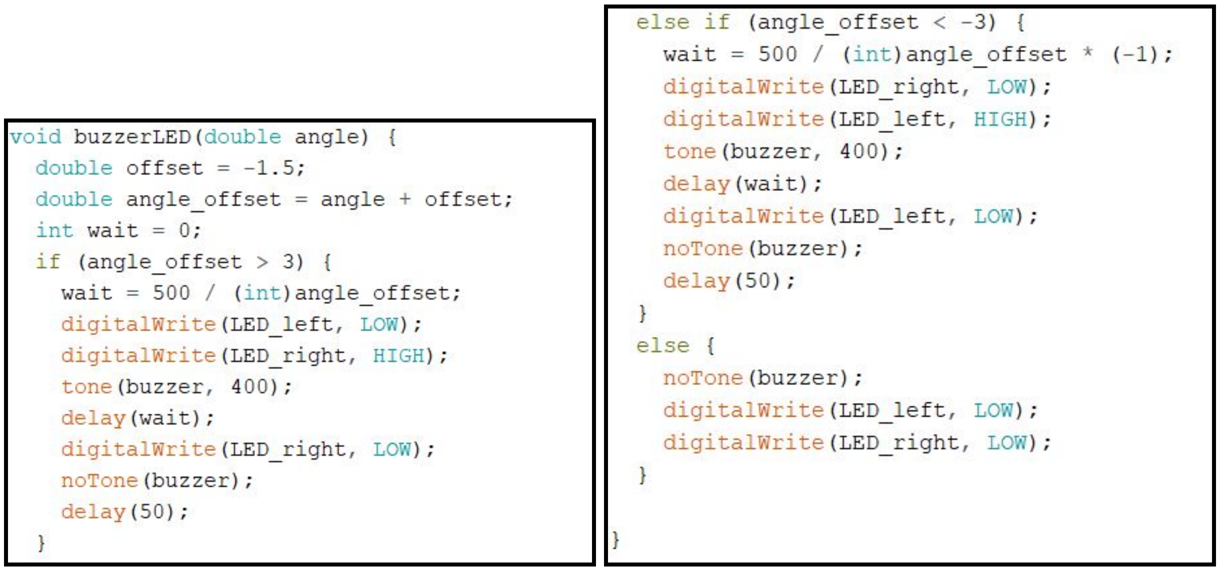
```
x = 90 - (atan(values[2] / values[0]) * (180 / PI));
if (x > 90) x -= 180;

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;
```

Figure 7. getAngle() Function

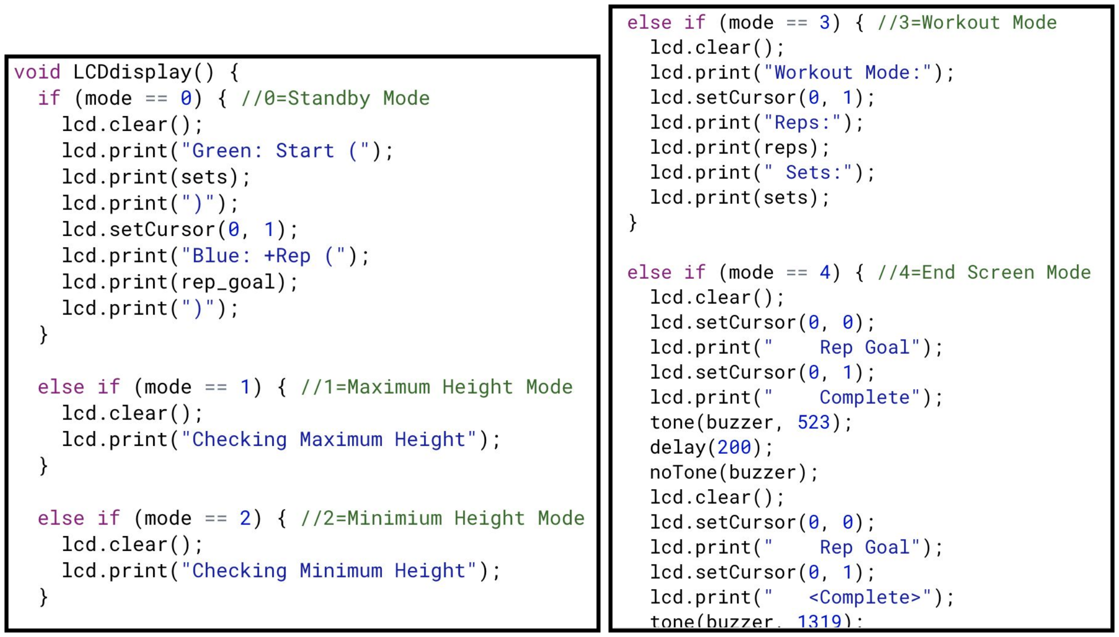


```
void buzzerLED(double angle) {
  double offset = -1.5;
  double angle_offset = angle + offset;
  int wait = 0;
  if (angle_offset > 3) {
    wait = 500 / (int)angle_offset;
    digitalWrite(LED_left, LOW);
    digitalWrite(LED_right, HIGH);
    tone(buzzer, 400);
    delay(wait);
    digitalWrite(LED_right,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

}
```

Figure 8. buzzerLED() Function



```
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);
```

Figure 9. LCDdisplay() Function

Appendix D Cost and Schedule

Table 8: Parts Cost

Part	**Price [$]**	**Quantity**	**Part #**	**Manufacturer**
Atmega328P	2.08	1	32538KB	Microchip
LCD Display (HD44780)	7.99	1		HiLetgo
ADXL 355Z	35	1	584-EVAL-AD XL355Z	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			

Table 9: Group Schedule

Week	Patrick Fejkiel	Kevin Mienta	Greg Gruba
10/5/20	Coding and Finalize Parts List: Create general layout of code and make sure all necessary parts are ordered	Coding and Schematic: Create general layout of functions needed in software and start designing schematic	Schematic and Finalize Parts List: Start designing schematic using components in current parts list
10/12/20	Coding and Schematic: Start integration of software functions with rest of code and finalize schematic	Coding and PCB: Work on finalizing PCB along with testing functions with different components	Coding and PCB: Work on optimizing PCB for a minimal package and software/hardware integration
10/19/20	Physical Design and Build: Start assembling components onto breadboards	Physical Design and Build: Start designing component layout in physical case	Physical Design and Build: Start designing physical housing case
10/26/20	Physical Design and Build: Solder components onto PCB	Physical Design and Build: Solder components onto PCB	Physical Design and Build: Solder components onto PCB
11/02/20	Testing and Improvements/Verification: Verify all components meet verification justifications	Testing and Improvements/Verification: Verify all components meet verification justifications	Testing and Improvement/Verification: Verify all components meet verification justifications
11/09/20	Testing: Test software portion designated for ultrasonic sensor distance values	Testing: Test software portion designated for accelerometer measurements	Testing: Test software portion designed for user interface with device components
11/16/20	Testing: Finalize working device before Demo	Testing: Finalize working device before Demo	Testing: Finalize working device before Demo