

INTERACTIVE SELF-STANDING TRAINING BAG



UNIVERSITY OF CENTRAL FLORIDA

COLLEGE OF ENGINEERING AND COMPUTER SCIENCE

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

Dr. Samuel Richie

EEL4915: Senior Design II

Final Report

Group 22

Hannah Clarke: Electrical Engineering

Joseph De La Pascua: Electrical Engineering

Nicole Karam Pannaci: Electrical Engineering

Natesha Ramdhani: Electrical Engineering

Table of Contents

Table of Contents	i
Table of Figures	x
Table of Tables	xiii
1.0) Executive Summary	1
2.0) Project Motivation	2
3.0) Project Goals and Objectives.....	3
3.1) General Specifications.....	3
3.2) Marketing Requirements	4
3.3) Technical Requirements	4
3.4) House of Quality	5
4.0) Realistic Design Constraints	6
4.1) Economic and Time Constraints.....	6
4.2) Environmental, Health, and Safety Constraints	7
4.3) Social, Ethical, and Political Constraints.....	8
4.4) Manufacturability and Sustainability Constraints	8
5.0) Standards	9
5.1) Power Supply Standards.....	9
5.2) Regulation Workout Bag Standards (weight, size, etc).....	9
5.3) PCB Standards.....	9
5.4) Communication Protocol Standards.....	10
6.0) Market Research	11
7.0) Technical Component Research and Standards Comparison	13
7.1) Sensors (Nicole).....	13
7.1.1) <i>Technical Standards</i>	13
7.1.2) <i>Candidates for Use in Device</i>	13
7.1.2.1) <i>Piezoresistance Sensors</i>	14
7.1.2.2) <i>Piezoelectric Sensors</i>	14

7.1.2.3) Strain Gauge	14
7.1.2.4) Inertial Measurement Unit (IMU) Sensors.....	14
7.1.2.5) Accelerometers	15
7.1.2.6) Binary Sensors.....	15
7.1.2.7) Textile Pressure Sensors	15
7.1.3) Design Matrix and Analysis	15
7.2) Indicators (Natesha).....	16
7.2.1) Technical Standards	16
7.2.2) Candidates for Use in Device	16
7.2.2.1) Addressable LEDs	16
7.2.2.2) LEDs	17
7.2.2.3) Electroluminescent Wire.....	17
7.2.3) Design Matrix and Analysis	18
7.3) Processor & Memory (Joseph)	19
7.3.1) Technical Standards	19
7.3.2) Considerations for Components.....	19
7.3.2.1) FPGA.....	19
7.3.2.2) DSP.....	20
7.3.2.3) Raspberry Pi.....	21
7.3.2.4) MCU	21
7.3.3) Design Matrix and Analysis	22
7.4) UI System (Hannah)	23
7.4.1) Technical Standards	23
7.4.2) Considerations and Candidates for Components.....	23
7.4.2.1) Displaying Results and Messages to User	23
7.4.2.2) Receiving User Input.....	24
7.4.2.3) Communication between the Base System and the User Interface	24
7.4.3) Design Matrix and Analysis	25
7.5) Power Supply (Joseph)	26
7.5.1) Technical Standards	26

7.5.2) Considerations for Components	26
7.5.2.1) Remote Power Supply and Conversion	27
7.5.2.2) Base System Power Supply and Conversion	27
7.5.3) Design Matrix and Analysis	27
7.6) Software (Hannah)	27
7.6.1) Technical Standards and Analysis	28
7.6.2) Considerations for Components	28
8.0) Device Design Concept	29
8.1) Functional Modes of Device's Side B.....	30
8.1.1) Accuracy Mode	31
8.1.2) Reaction Time Mode	31
8.1.3) Cardio Mode	31
8.2) Functional Mode of Device's Side A.....	32
8.2.1) Combination Generator Mode	33
8.3) General Performance	33
8.3.1) Display/Ambient Mode.....	33
9.0) Base System Hardware Design.....	34
9.1) Device Body Build and Design Plan	34
9.2) Sensor Design: Side B	36
9.2.1) Sensor Choice: Pressure Sensor.....	37
9.2.2) Side B Sensor Layout	38
9.2.3) Sensor Design: Side A	38
9.3) Indicator Design.....	39
9.3.1) Choosing an LED Strip	39
9.3.2) Cutting, Shaping, and Attaching the LED Strip.....	40
9.3.3) Issues Using the LED Strips	41
9.3.4) Choosing the LED String	41
9.3.5) Cutting, Shaping, and Attaching the LED String.....	42
9.4) Power Supply Design.....	44

9.5) Processor Design.....	46
9.5.1) TI ATmega2560.....	46
10.0) UI Hardware Design	47
10.1) Processor.....	48
10.2) Display.....	48
10.3) Buttons & Switches	48
10.4) Power Supply	48
10.5) Protective Case	48
11.0) Printed Circuit Boards.....	51
11.1) Base System Design.....	51
11.1.1) Pressure Sensors (1-8).....	52
11.1.2) Xbee Module (9).....	52
11.1.3) Power Supply and Conversion (10).....	52
11.1.4) MCU: ATmega2560 (9)	53
11.1.5) Addressable LED Strings (10-13, 15).....	53
11.2) UI Handheld System Design	54
11.2.1) MCU: ATmega2560 (1)	54
11.2.2) Xbee Module (2).....	55
11.2.3) Buttons (3-6).....	55
11.2.4) LCD I2C Module (7).....	56
11.2.5) Power Switch, Supply, and Conversion (8)	56
11.3) Schematics, Boards, and Bills of Materials	56
11.3.1) Base System.....	56
11.3.2) UI Handheld.....	61
11.4) Prototyping and Preliminary Testing	66
11.5) PCB Assembly Planning.....	67
11.5.1) Option 1: DIY Soldering Gun	67
11.5.2) Option 2: Purchasing a Soldering Gun	67
11.5.3) Option 3: Using a Local Company.....	68

11.5.4) Option 4: Ordering the PCB Online and Shipping to Orlando	68
11.6) PCB Final Assembly and Testing	68
12.0) UI and Base Software Design	70
12.1) Design Method	70
12.2) Design Environment / Tools.....	70
12.3) General Overview	70
12.4) UI System	71
12.4.1) Low Power Modes	71
12.5) LCD Display	71
12.5.1) Welcome Screen.....	71
12.5.2) User Input.....	72
12.5.2.1) MODE (MODE BTN).....	73
12.5.2.2) SELECT (SEL BTN).....	73
12.5.2.3) DIFFICULTY (DIFF BTN).....	73
12.5.2.4) CANCEL (CXL BTN)	74
12.5.3) UI Failsafe Timer 0	74
12.5.4) Results Transmission	74
12.6) Communication with Base	75
12.7) Base System.....	75
12.7.1) Target Areas (LEDs).....	75
12.7.1.1) Combination Generator Mode.....	77
12.7.1.2) Reaction Time Mode.....	78
12.7.1.3) Accuracy Mode.....	79
12.7.1.4) Cardio Mode	80
12.7.1.5) Ambient Mode.....	81
12.7.2) Results Computation	82
12.7.2.1) Combination Generator Mode.....	82
12.7.2.2) Reaction Time Mode.....	82
12.7.2.3) Accuracy Mode.....	83
12.7.2.4) Cardio Mode	83

13.0) Function Analysis and Development.....	84
13.1) LCD Configuration	84
13.1.1) <i>Clear Display</i>	85
13.1.2) <i>Return Home</i>	85
13.1.3) <i>Entry Mode</i>	85
13.1.4) <i>Display Control</i>	86
13.1.5) <i>Cursor / Display Shift</i>	86
13.1.6) <i>Function Set</i>	86
13.1.7) <i>Set DDRAM Address</i>	86
13.1.8) <i>Write Data to Display</i>	87
13.2) LCD Message Functions	89
13.2.1) <i>Pre-Mode Execution Messages</i>	89
13.2.1.1) <i>Power On and Initialization</i>	89
13.2.1.2) <i>Welcome Message</i>	90
13.2.1.3) <i>Mode Select</i>	90
13.2.1.4) <i>Difficulty Select</i>	91
13.2.1.5) <i>Start</i>	91
13.2.2) <i>Results Display Message Functions</i>	92
13.2.2.1) <i>Ambient Mode</i>	92
13.2.2.2) <i>Combination Generator Mode</i>	93
13.2.2.3) <i>Reaction Time Mode</i>	93
13.2.2.4) <i>Accuracy Mode</i>	94
13.2.2.5) <i>Cardio Mode</i>	94
13.3) LCD Communication	95
13.3.1) <i>I2C</i>	95
13.3.2) <i>GPIO Pin Connection</i>	95
13.4) Zigbee Configuration	96
13.4.1) <i>Coordinator & End Devices Configuration</i>	96
13.5) MCU and XBee Communication	97
13.5.1) <i>UI System</i>	97

13.5.2) Base System.....	97
13.6) Button Configuration	98
13.7) Button ISRs	98
13.7.1) Mode Select Button ISR	99
13.7.2) Difficulty Select Button ISR.....	99
13.7.3) Select Button ISR.....	100
13.7.4) Cancel Button ISR.....	100
13.8) Timer Configuration and ISR.....	101
13.8.1) Timer ISR	101
13.9) General Mode Functions 0-4	102
13.10) LED Use in Modes	102
13.10.1) LED Functions.....	102
13.11) Sensor Use in Modes	103
14.0) Results Calculation Functions	104
14.1) Combination Generator Mode Results	104
14.2) Reaction Time Mode Results	104
14.3) Cardio Mode Results	104
14.4) Accuracy Mode Results	104
14.5) Function Summaries.....	105
15.0) UI Integration Test Plan	108
15.1) LCD and Button Interaction	108
15.1.1) Welcome Message	108
15.1.2) Mode Select Message.....	108
15.1.3) Difficulty Select and Start Messages	109
15.2) Timer.....	109
15.3) Results.....	110
16.0) Complete Device Test Plan	111
16.1) Side A and B: Pressure Sensors.....	111
16.3) Final Integrated Mode Testing.....	112

17.0) User Manual	114
17.1) General Safety Considerations.....	114
17.2) Directions for Use	114
17.2.1) <i>Combination Generator Mode</i>	115
17.2.2) <i>Reaction Time Mode</i>	115
17.2.3) <i>Accuracy Mode</i>	115
17.2.4) <i>Cardio Mode</i>	115
17.2.5) <i>Ambient Mode</i>	115
18.0) Administrative Content	116
18.1) Initial Milestones and Timing for Senior Design 1 and 2	116
18.1.1) <i>Prototype Design & Build: January – March 2021</i>	116
18.1.2) <i>Prototype Testing: February 2021</i>	117
18.1.3) <i>Final Build & Test: March - Mid April 2021</i>	117
18.2) Group Member Familiarity with Project Elements	117
18.3) Project Challenges and Unfamiliarity	118
18.3.1) <i>Unfamiliarity with Project Elements</i>	119
18.4) Budget and Financing.....	120
19.0) Conclusion and Next Steps	122
References	1
Appendix.....	1
A) Pinout Diagram of ATmega2560.....	1
B) Connection for LCD.....	2
C) WS2811 LED Strip Driver.....	2
D) XBee Pin Diagram	3

Table of Figures

Please note that all figures within this document, unless cited otherwise in the photo caption, have been produced by the members of this group: Natesha Ramdhani, Hannah Clarke, Joseph De La Pascua, and Nicole Karam.

Number	Description	Page
1	Point to Point (Peer to Peer) vs Hub and Spoke networks	10
2	The main idea for the US Patent US20110172060A1	11
3	A variation of the US Patent US20110172060A1	11
4	Hanging Bag Structural Design	12
5	Standing Bag Structural Design	12
6	Association map of device active modes	29
7	“Side B” of the device, Annotated	30
8	“Side A” of the device, Annotated	32
9	Hardware connection block diagram; distribution of group roles	34
10	Century Martial Arts Wavemaster options for body base	35
11	“Side A” of the device	35
12	“Side B” of the device	35
13	Layers of Pressure Sensor	37
14	Layers with Conductive Thread Detail	37
15	Side B Target Sensor Internal	38
16	Physical Implementation of Figure 13 Sensors	39
19	WS2811 LED Strip with wire ribbon configuration	40
18	Diagram of LEDs on String	42
19	Side A Lighting Arrangement	43
20	Side A Lighting Arrangement, Physical	43
21	Side B Lighting Arrangement	43
22	Flow chart for Power Supply in Base System	45
23	Draft sketch of the UI system for the device	47
24	Technical Drawing of UI Remote Case Component	49

25	Physical Implementation of Remote Case, Side View	50
26	Physical Implementation of Remote Case, Front View	50
27	Side A with Component Reference Numbers	51
28	Side B with Component Reference Numbers	51
29	UI Handheld with Component Reference Numbers	54

Number	Description	Page
30	Base System Schematic Including Peripherals	58
31	Base System PCB Layout	59
32	Physical Base System PCB	60
33	UI Handheld System Schematic	63
34	UI Handheld System Board Layout	65
35	Software flow chart for UI and Initialization	72
36	Software flow chart for Base Active Mode Initialization	76
37	Software flow chart for the Combination Generator Mode on Side A	77
38	Software flow chart for the Reaction Time Mode on Side B	78
39	Software flow chart for the Accuracy Mode on Side B	79
40	Software flow chart for the Cardio Mode on Side B	80
41	Software flow chart for the Ambient Mode using both sides A and B	81
42	CGRAM Address Table	88
43	Welcome Screen Display Message Sample	90
44	Start Screen Display Message Sample	92
45	Combination Generator Results Display Message Sample	93
46	Reaction Time Mode Results Display Message Sample	93
47	Accuracy Mode Results Display Message Sample	94
48	Cardio Mode Results Display Message Sample	94
49	Sensor Reading of Five Quick Hits	111
50	Group 22 With Their Completed Device	123

Appendix Table of Figures

Letter	Description	Page
A	Pinout Diagram of ATmega2560	A – 1
B	I2C Connection for LCD	A – 2
C	WS2811 LED Strip Driver	A – 2
D	ZigBee Pin Diagram	A – 3

Table of Tables

Number	Description	Page
1	House of Quality Specification Matrix	5
2	Technical Standards for Sensors	13
3	Design Matrix for Sensors on Side B	15
4	Technical Standards for Indicators	16
5	Design Matrix for Indicators	18
6	Technical Standards for Processor and Memory	19
7	Design Matrix for Processor and Memory	22
8	Technical Standards for UI System	23
9	Design Matrix for UI System	25
10	Technical Standards for Power Supply	26
11	Technical Standards for Software	28
12	Indicator Specifications: LED Strip	41
13	LED Strip vs LED String	44
14	ATmega 2560 Specifications	46
15	Pressure Sensor Pin Connections	52
16	Xbee Pin Connections	52
17	Base System PCB Bill of Materials	61
18	UI Handheld System PCB Bill of Materials	65
19	Mode options and register values	73
20	Difficulty options and register values	74
21	Address indices corresponding to row and position of 20x4 LCD	87
22	CGRAM codes to display 'H' on one cell of the LCD screen	87
23	LCD Basic Function Table	88
24	LCD Message Function Summary Table	89
25	Mode Register Values with Corresponding Modes	91

Number	Description	Page
26	Difficulty Register Values with Corresponding Difficulties	91
27	LCD Command Function Summary	105
28	Display Function Summary	105
29	LCD Communication Function Summary	106
30	ZigBee Function Summary	106
31	Button ISR Summary	106
32	Timer Functions and ISR Summary	107
33	Mode Function Summary	107
34	Sensor and LED Function Summary	107
35	Results & LPM Function Summary	107
36	Expected Costs for this Project	121

1.0) Executive Summary

For our Senior Design project, we are creating an interactive boxing/martial arts stand up workout bag, where the athlete can choose different types of workout patterns, and receive results to view progress. A regular standup punching bag alone does not have the capability to improve a user's performance, since it does not provide feedback or results without a boxing/martial arts trainer. With an interactive design, the user would be able to improve performance in cardiovascular strength, accuracy, reaction time and speed. This design should be easy to use for first time athletes as well as experienced athletes because the user is able to select different mode levels that are most suitable for them, and increment their levels as needed to continue to receive better training.

In order to capture as many different aspects of training as possible, the bag will have a dual-sided design that accommodates various modes that the user can train with. "Side A" will have various zones with binary touch sensors that will detect whether a certain area was hit. These zones will be loosely based on different practical locations that a fighter would be making contact with in a boxing, kickboxing, or martial arts environment against a real opponent, so that the user can have the most immersive and accurate training possible. On "Side A", the device should be able to generate random sequences of techniques landing in various areas; this can be extended to using multi-color indicators to signal which side or technique type should be used to attack the opponent. "Side B" will contain a target zone with a cluster of multiple sensors arranged within this target so that more focused training can also take place and create a complete, rounded training session. These sensors will gather data on the training session and report it back to the user so that the user can quantify their performance in different types of workouts and track their progress. Some modes that can be implemented by "Side B" include a cardio mode where users try to make as many hits as possible within a time limit, a reaction mode where users aim to land their hits based on a stimulus by the indicator, and an accuracy mode where the user practices hitting a specific area as precisely as possible. For the design to be interactive, a user interface system is included: a handheld device with which the user is able to select their workout method, select the difficulty setting and is where they are able to see their performance results.

This report describes how the interactive martial arts bag will be designed, built and tested. In the beginning, the first sections will define our motivation behind the interactive martial arts bag and the goal and objectives our project must meet to be successful. The research and design considerations follow, giving details on the technical investigations done to meet our goals and objectives, as well as, compare different design approaches. In the next section, standards and constraints and their implications to our project design are discussed. The hardware and software design will then be overviewed, including the design of each subsystem and each component's role in the complete system. Next, our test plan is reviewed, detailing how each subsystem will be tested, as well as the system as a whole. Lastly, the administrative content is discussed, which includes the timeline of construction, group member familiarity and challenges, and the proposed budget of the design.

2.0) Project Motivation

Our motivation behind this project stems from the idea of creating an easy yet engaging way for a trainee of any level, to exercise, train, and track their progress in various perspectives of fight training without the need for a training partner in a safe and cost effective way. The Coronavirus pandemic has made this even more relevant since most training locations closed for an extended time, leaving many people without a method or physical partner to train at home with. Aside from the impact of coronavirus, there are many other factors that limit someone's ability to obtain access to a gym or personal trainer, such as social anxiety, lack of transportation, self consciousness or insecurities, lack of a babysitter, and more. With an interactive workout bag, you are able to train effectively within the comfort of your own home without sacrificing the quality of your exercise.

Another important aspect behind our group's motivation for this project is the important difference between using an interactive martial arts bag versus a standard martial arts bag. With a standard bag, a user with no martial arts experience may find it difficult to judge their performance, or even what type of training they wish to perform. An interactive workout design eliminates the confusion that beginners may face, especially when training alone. Our design is able to guide the user through the session, indicating where to hit and effectively measuring their performance for them. For users with martial arts experience, our design enables them to commit their undivided attention towards their training, rather than simultaneously tracking/measuring their own performance.

In addition, the current cost of interactive workout bags on the market are disproportionately high. Access to effective and engaging exercise should be affordable for the average American family. Our design will be at a lower cost to those on the market, which can do some, but not all, of the actions as our interactive design. Currently on the market, interactive standup punching bags usually sell for about \$25,000. Because our design can provide feedback/results after a session as well as provide interesting/fun workout patterns, the user can practice their skills in an engaging manner without paying the \$40 to \$70 average hourly cost of a personal trainer. Additionally, our design can be used at home which saves the user the monthly cost of a gym membership necessary for the correct equipment: a cost that ranges from \$10 to \$100 per month.

We believe that our design offers a unique solution to these problems, enabling people to exercise and train in new, practical, and useful ways without the need for an expensive trainer or class, and for a much less eye-watering price than the few similar products on the market.

3.0) Project Goals and Objectives

The interactive workout bag's goals and objectives are the first conditions that must be reviewed and satisfied to complete this project successfully and the order in which these objectives and goals should be done. This section will serve as a checklist for the implementations of the project and contain the general specifications since these will help us to recognize all goals and objectives of this project.

3.1) General Specifications

The general specifications help to guide us and our audience regarding what this project will accomplish, how it will behave, and how it will interact with external stimuli. This can be considered the "skeleton" of the project since every decision for this project was decided based on these general specifications, and what was needed to achieve a certain objective for the project. As shown below, each mode will have the order on which activity it should do first and how the user should interact with each mode. Some modes may share the same components, but the way that the mode behaves on the side differs.

This device should:

- Be able to implement downtime modes
 - Idle Mode
 - Display Mode
 - Off Mode
- Be able to implement exercise modes
 - Combination Generator (Side A)
 - Uses set zones on opponent
 - Binary sensors count correct hits and save data for user
 - Practice training against an "opponent" to improve the user's skill
 - Cardio Mode (Side B)
 - Uses sensor target to count hits
 - User goal: make as many hits as possible in a certain time period
 - more hits = higher score
 - Reaction Mode (Side B)
 - Uses sensor to detect time for user to hit target after illuminating indicator
 - User goal: hit target as soon as possible after it lights up
 - Accuracy Mode (Side B)
 - Uses sensor to detect how close to center mark target is hit
 - User goal: hit center as accurately as possible to increase score

3.2) Marketing Requirements

Marketing requirements detail characteristics that our team would like the device to embody, which can be used to appeal to the audience of users and possible users of this device. These requirements include behavior, interaction, and output of the equipment in these device. The plus sign(+) means that if this trait goes up, it will be beneficial for this project. The negative sign(-) means that if this trait goes down, it will be beneficial for this project to have.

- **Durability:** Device will be able to withstand impact from regular use. (+)
- **Cost:** Device must be cost efficient for a typical consumer; priced reasonably compared to other multimode home-gym fitness devices. (-)
- **Portability:** Device must be able to be moved if necessary without large risk of damage, and must be able to be used in a variety of different environments. (+)
- **Intuitive:** The user interface will be easy to understand for experts and enthusiasts alike. (+)
- **Versatile:** The device will have multiple modes of operation utilizing both sides and orientation of sensors for a well-rounded user experience. (+)
- **Device life:** The device should last as long as possible; at least 5 years for the user. (+)

3.3) Technical Requirements

Technical requirements are characteristics that we, as engineers of this device, are aiming for each component to have. These requirements will be met through our choices of components and our integration of the individual components within the device as a whole. As with the marketing requirements, the plus sign(+) means that if this trait goes up, it will be beneficial for this project. The negative sign(-) means that if this trait goes down, it will be beneficial for this project to have.

- **Durability:** Components will be able to stand up to impact at a rate of over 120 hits/minute. (+)
- **Power:** Components will be power efficient; no more than 1.1kW. (-)
- **Compatibility:** Components will integrate well with other components used in the device. (+)
- **Component Costs:** Total cost of type of components of good quality will be within our budget, but as low as possible. (-)
- **Implementation Time:** The time taken for the design and implementation of modes using the sensors should not be excessive. Our goal is 12 weeks or less to build and develop. (-)
- **Longevity:** Components will not die out quickly, providing approximately 5 of years of use. (+)

3.4) House of Quality

The house of quality helps us visualize the correlation between the Technical Requirements of this project with itself, and the Technical Requirements with the Marketing Requirements. In a perfect world, everything would work nicely without creating issues in another section, and you could implement anything without consequences. We, as engineers, have to always have to see what requirements that are in the sections clash with each other, and we also have to see how strong or weak they correlate with each other since we have to come up with a choice of balancing out these requirements to have a desirable outcome with this project.

Double Symbol = **Strong** Correlation | Single Symbol = **Weak** Correlation

	Implementation Time (-)						Technical Requirements vs Technical Requirements
	Component Cost (-)					↓	
	Compatibility (+)			↑	↓		
	Power (-)					↑↑	
	Durability (+)			↓		↑↑	
Technical Requirements (Green) vs Marketing Requirements (Blue)		Durability (+)	Power (-)	Compatibility (+)	Component Cost (-)	Implementation Time (-)	Longevity (+)
	Durability (+)	↑↑			↓		↑
	Cost (-)	↓		↓	↑↑		↓
	Portability (+)	↑			↓		↑
	Easy to Use (+)			↑		↓	
	Versatile (+)		↓	↓	↓	↓	↑
	Device Life (+)	↑↑			↑		↑↑

Table 1: House of Quality Specification Matrix

4.0) Realistic Design Constraints

Design constraints are involved in every engineering project. The following content provides certain constraints that will apply to the manufacturing and operation of the punching bag. These constraints include: Economic, Time, Environmental, Health, Safety, Social, Ethical, Political, and Manufacturability and Sustainability constraints. These constraints provide limits to engineering designs based on standard scarcity that exists in the material world. Some constraints, such as the health and safety, are less focused on scarcity and more focused on protecting both the consumer and the commons that the consumer is free to operate in. Each constraint will be considered both individually and with regard to the full scope of the project.

4.1) Economic and Time Constraints

The budget for our project is \$900. To cover the costs of our project, our group has decided to self-fund the budget. The primary costs of the project are the punching bag itself and the electronics that will be used. The punching bag itself is the largest direct cost for this project, to lower this cost a used punching bag will be acquired. A used bag is preferred as it will be heavily modified to accommodate the electronics used. Although our selection of MCU has lowered the costs associated with the bag, multiple sensors will be acquired and used throughout the bag, adding more cost. Although sensors are generally inexpensive, multiple sensors are needed to cover all sections of the bag required to bring full functionality to the bag. Some of the components will be bought twice to cover any component failure that could happen along development of the project. Economic constraints are necessary to consider as most items need to be purchased to finish and manufacture the product. This will be finalized until the end of the project.

Another constraint to consider for the running cost of this project is the cost of electricity. This constraint should be considered not in the manufacturing, but in the final product operating cost. This product will require standard US AC power input, and therefore will require standard energy costs for operation. Power supply design and operations will factor in to how much power is needed to run this machine and therefore how expensive the operating costs will be. This constraint will differ depending on which region and power supplier are used to power the machine. Standard Duke Energy operating costs in Florida is an average of \$0.12/kWh. This does not account for monthly fixed costs for power.

One large constraint for this project is the overall time limit of the project. The project will be done in April 2021. The research and testing of the project will be complete during the first semester of Senior Design, by the beginning of December 2020. After research and testing is complete, the manufacturing of the punching bag will begin in the second semester of Senior Design in Spring 2021. Product testing will be done during and after the manufacturing. Testing will be required to incorporate the design of both the code and the electronics in the punching bag. After extensive testing, the completed product will be presented to the judges panel at the end of the Spring 2021 semester. Time constraints are vital to consider as they dictate the schedule this project must follow.

4.2) Environmental, Health, and Safety Constraints

To protect the commons for the consumers and producers of different items, certain environmental, health, and safety constraints need to be considered. The main environmental constraint restricting the product is the cost and availability of electricity. Electricity in the surrounding area is supplied by Duke Energy, who owns a coal plant that provides energy and electricity to much of the Orlando/UCF area. The direct cost of the electricity, in terms of dollars, was given in the previous economic constraint section, but environmental considerations must be taken in the context of 21st century environmental degradation. Due to power being supplied by a coal power plant, running machinery in the Orlando area contributes to further carbon emissions in the atmosphere, exacerbating climate change.

To limit the effects of runaway climate change, measures should be taken in the design steps to use as little power as possible for the operation of the machine. This constraint is part of a collective effort to limit personal contributions to greenhouse emissions and should be evaluated in all future engineering projects and designs. This personal responsibility cannot replace certain standards set for power efficiency and regulations on limits of power and emissions usage.

Due to consumer use, health and safety constraints are required for the design of the machine. The most pressing issue is the electrical wiring of the machine. The sensors on the punching bag will need to be wired to the MCU to perform the calculations necessary to the project. These wires should be out of the way of the punching surfaces for the safety of the user and the components.

Another safety consideration is the overall weight of the machine. With the added components, the weight of the machine should not go over a threshold that would make it unsuitable to be carried without the water or sand stabilizer. This is important both for the usability and for the safety of the consumer. Due to the relatively low weight of the MCU, wires, and sensors when compared to the bag itself, this constraint should not be too limiting.

One final constraint is the heat of the circuitry and the MCU. The MCU should be housed in heat resistant plastic in order to shield it from the rest of the machine, which could be damaged when exposed to any heat from the MCU.

4.3) Social, Ethical, and Political Constraints

Surrounding the manufacturing of this project there are several social, ethical, and political constraints for our design. The social and ethical responsibility to lower power use and overall emissions has been previously discussed. Another ethical constraint applying to this project is the responsibility to report the correct calculations for each workout mode. This product will be used by athletes who must complete standard workout regimens many times a week. To aid their schedules, our machine needs to work every time it is powered on to be used and to report accurate information at the end of each section for the athlete to assess. For athletes to rely on our product, their trust should be gained by effective and habitual use. Sticking to this constraint will also fulfill political constraints including ethical manufacturing of the product; if the product does not report information correctly, customers have grounds to stop using the product and even grounds to sue depending on the accuracy of our product's ending information and assessment. Avoiding false advertising of our product in the future would work to ease this concern, especially around the assessment of the workout our product performs.

4.4) Manufacturability and Sustainability Constraints

Sustainability is a constraint with a lot of focus for this project. Not only should sustainability of power be taken into account for but also sustainability of building materials. Electronics especially do not fare well once thrown out, in fact many electronics need to be specifically recycled to be disposed of safely. Our component disposal will follow all electronic recycling guidelines outlined by the Environmental Protection Agency (EPA). Only the regulations from the EPA have to be followed as the State of Florida has no additional E-waste laws or regulations. An E-waste recycler with a third-party certification is recommended for use by the EPA and the Florida Department of Environmental Protection. Another constraint for manufacturing sustainability is the punching bag itself. To meet sustainability (and cost) constraints, a used bag will be acquired to use for the prototype. This assures that new material and products are not wasted in the construction of the machine. Buying used components is one major consideration we made to meet both the economic and sustainability constraints. Two constraints were effectively applied during our buying habits.

5.0) Standards

Although there are not many standards that apply to the interactive workout bag product, the few that do apply include power supply standards, standards for fitness equipment, communication standards and PCB standards. Each type of applicable standard is examined, as well as its correlation to the design.

5.1) Power Supply Standards

IEC 60906-2:2011

Also known as the IEC system of plugs and socket-outlets for household and similar purposes, Part 2: Plugs and socket-outlets 15 A 125 V a.c. and 20 A 125 V a.c., is a standard that applies any to the NEMA 5-15-P, also known as the “3-prong grounded plug” connection. This standard explains that this connection will provide protective earthing to equipment that is connected to the conductive parts of the socket. It also is electronically separated from the protective earthing circuit to provide electrical noise immunity.

This standard applies to this project since we will need this type of plug to be compatible with the standard american wall outlet. Our design will also need the three prong connector (positive, neutral and ground) for our circuit design. It is also critical that our system has no electrical noise from the power supply, which the IEC 60906-2:2011 standard defines.

5.2) Regulation Workout Bag Standards (weight, size, etc)

The American Society for Testing and Materials (ASTM) is an international organization that standardizes the safety and testing of materials. The standard ASTM F2276-10(2015), also known as the Standard Specification for Fitness Equipment, is a standard that all designers and manufacturers for a fitness equipment must follow. The standard lays out specific criteria which is applicable to the workout bag design. In the standard, it is defined that only individuals age 12 or older can use this equipment, that the product must include all the documentation on how to assemble/build the product, include a list of parts and instructions, and provide adequate warnings. This standard also defines which exercise equipment is required to be used in an indoor setting or environment.

This standard applies to our project design because the workout bag falls under the classification of fitness equipment. Therefore, our project design will need to include a parts list, instructions and applicable warnings that include an age restriction.

5.3) PCB Standards

IPC-221B

The workout bag system implements two printed circuit boards, one in the base system and one in the UI system (remote). Association Connecting Electronics Industries, previously known as the Institute of Printed Circuits (IPC) sets the standards for all types of printed circuit boards (PCBs). In conjunction with the IPC standards.

The standards that apply to our PCB designs from the IPC are in IPC-2220 series. The IPC-221B standard is the standard that defines the fundamental design requirements (also called the generic design requirements) for designing the printed circuit boards, component mounting, and interconnecting structures. Because our PCBs will include a design and mounted components, it is important to follow the guidelines set by these standards.

5.4) Communication Protocol Standards

IEEE standard 802.15.4

The UI system and the base system communicate low rate wireless personal area network (LR-WPAN) which is defined by the IEEE 802.15.4 standard. The IEEE 802.15.4 standard is designed to create a WPAN that is low power. The IEEE 802.15.4 standard defines the physical layer (the bottom layer of the protocol layers) and affects the MAC layer, a sub layer of the network layer.

Zigbee, which is used to control the communication between the UI system and the base, builds on the IEEE 802.15.4 standard. The physical layer controls the interface of transmission, which include the transceiver and the transmission frequencies. The MAC (medium access control) layer controls the MAC frames that are then sent using the physical layer. The IEEE 802.15.4 standard also has three types of nodes on the network, a coordinator node, router node and device node. These nodes are able to interact as peer to peer networks or in spoke-hub networks, pictured below.

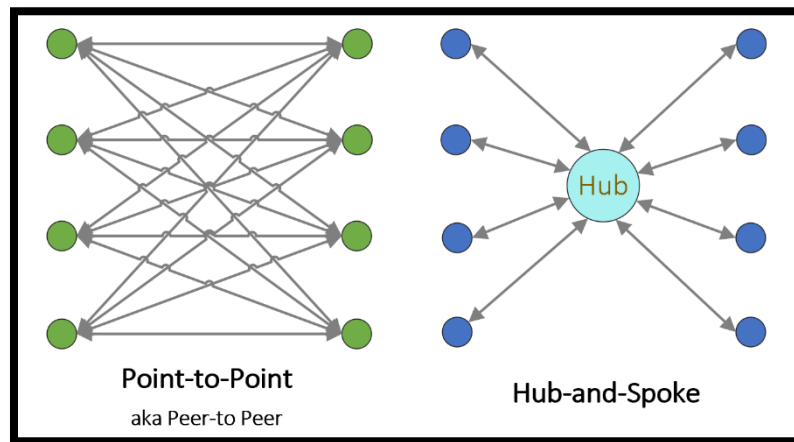


Figure 1: Point to Point (Peer to Peer) vs Hub and Spoke networks

The IEEE 802.15.4 standard also defines the RF parameters for the communication. In North America, the IEEE 802.15.4 standard uses 902 to 928 MHz frequency bands. There are a maximum of 10 supported channels with a bandwidth of 2MHz and a 0.6MHz guard band. The data rate is also defined at 40kbps. Although the data rate is low, this communication standard is perfect for our design, since only a few bytes need to be sent across the network, and the communication system needs to be low power.

6.0) Market Research

The idea of a dynamic interactive punching bag has existed for a while. The first time it was patented, a device like the one we are designing was back in 2010 [1] in which a patent was submitted for a martial arts bag with one or more sensors for measuring a workout performance. In which, they also show many pictures of many portrayals of this type of design in which our design is like a combination of the Figure 1 and figure 3A. In figure 1, it will have a coach that will show on the screen to help users to hit the bag. In the 3A case, it would impact areas around for the user to hit but it still does not mention how the results are recorded and shown to the user.

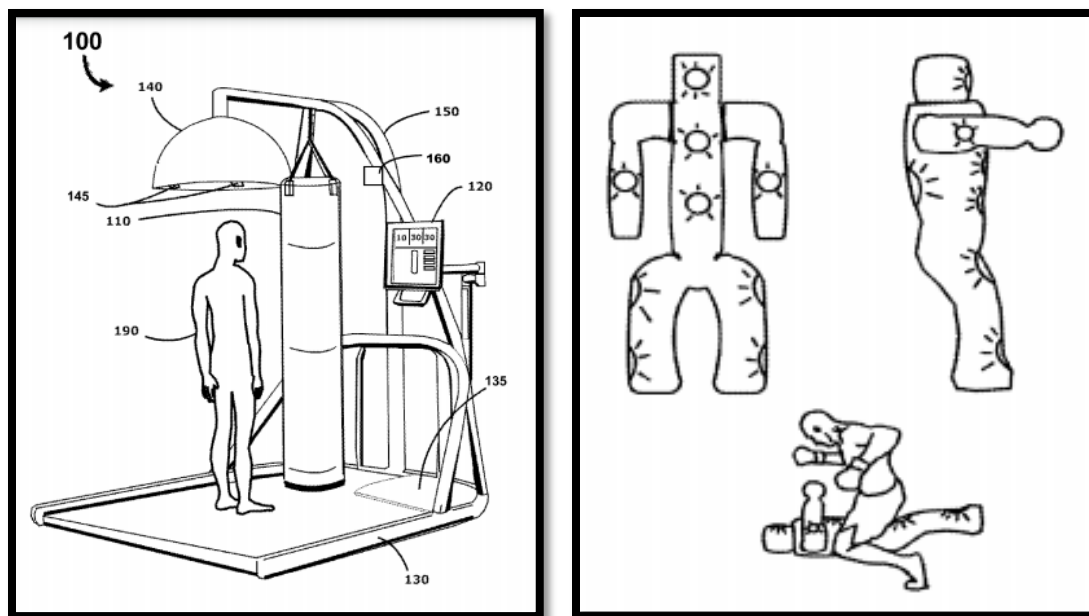


Figure 2 (Left): The main idea for the US Patent US20110172060A1
Figure 3 (Right): A variation of the US Patent US20110172060A1

We also found two research articles from the Czech Republic in which they used [7] and [8]. In which, both measure a direct punch applied to a martial arts bag using strain gauges inside of the martial arts bag. In regards to the type of strain gauge used, both [7] and [8] use a strain gauge type SRK-3/V, and in [8] also uses a strain gauge L6E-C3-300Kg in their measurements. It was found interesting since in [8] came to the conclusion that L6E-C3-300Kg strain gauge has a better precision, speed, and simpler to use than the strain gauge type SRK-3/V, but in [8] that was made with the same people used the latter instead to further intensify the research about measurement direct punch force.

PADIPATA™ created an interactive punching bag, with all its sensors in the middle of the punching bag and it only leaves the user to use punching instead of using the user's legs as well. It is quite like the Figure 2 in the [1] reference. This device comes in 3 sizes, but is only available in the form of a hanging bag as opposed to a self-standing bag with a base on the ground, the design which our group has chosen. It also has surround sound with an interactive screen inside of the band. This product was created in Changsha City, People's Republic of China and retails for a price of \$25,000 (USD); our intent is to create a device that is much lower in cost than that, so that it is accessible to a wider audience than the Padipata bag. The device our team will create should also employ different interactive modes to make it more efficient for the user.

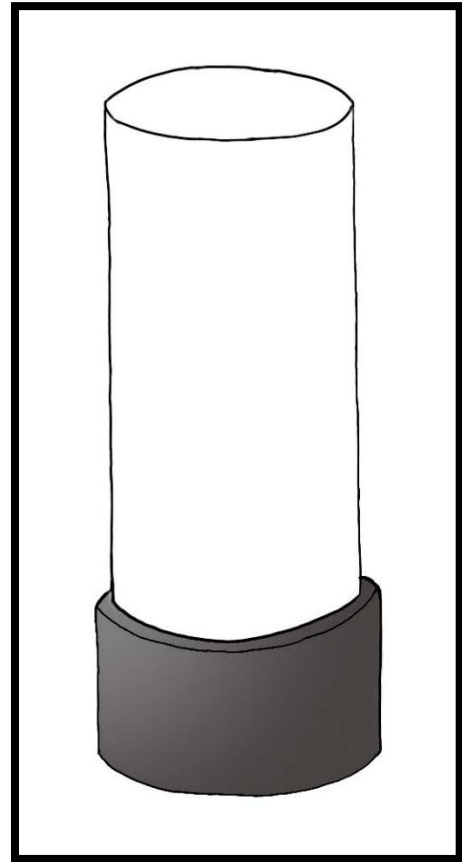
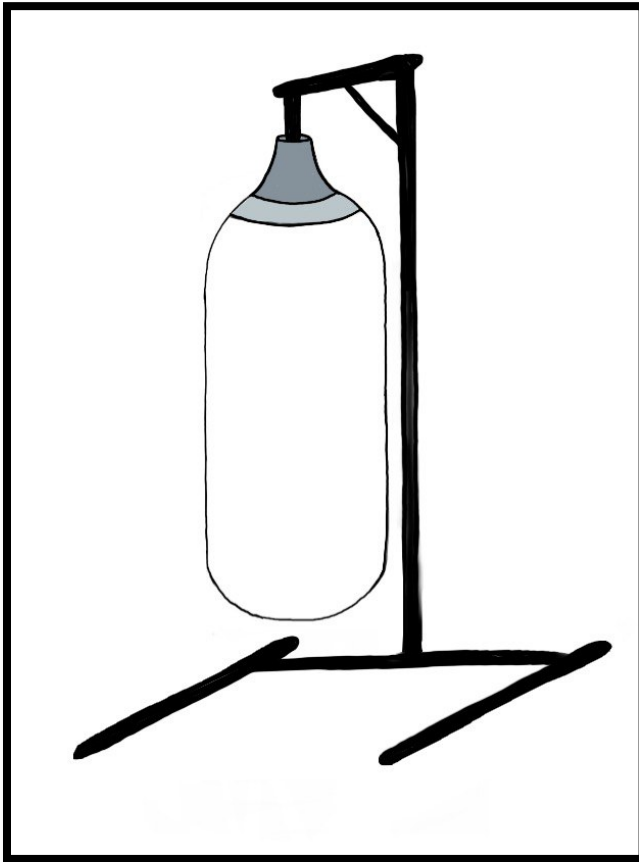


Figure 4 (Left): Hanging Bag Structural Design

Figure 5 (Right): Standing Bag Structural Design

7.0) Technical Component Research and Standards Comparison

In this section, we will detail our thorough research into each of the components that were a potential candidate for this project so that both our team and our audience of readers and users can learn more about all the potential components for this project.

We have also set up the standards we desire for the components to exhibit. We then ranked each of the components we have found in our research individually most exemplary of these standards to least exemplary. We chose the components that had the highest overall ranking among all the standards for this project since those components have the most desirable standards for this project.

7.1) Sensors (Nicole)

This section discusses the research and decision-making process to choose our sensors. These sensors must meet as many of our technical standards as possible, and present a cost-effective solution for our design.

7.1.1) Technical Standards

Sensors used in this device should:

1.1	Be durable enough to withstand the force of a punch/kick.
1.2a	Be able to detect distinct, rapid, successive hits in different locations (Side A).
1.2b	Be able to detect distinct, rapid, successive hits in the same location (Side B).
1.3	Be able to detect the location of a hit relative to a target point (Side B).
1.4	Consume a low amount of power.
1.5	Cover a zone with no more than four individual sensor units.

Table 2: Technical Standards for Sensors

7.1.2) Candidates for Use in Device

We will need sensors for our project, and these are the research done to find values of them. The original concept for the sensors in this device were inspired by the Dance Dance Revolution(DDR) home mats since these mats can support the weight of people stepping to it with extra force due to the excitement of the people when they play this game. These DDR mats use buttons to signal that an arrow was pressed, so no sensors are used. We could incorporate the following alongside/ instead of the buttons to enhance the how much force someone should apply to the self-standing kicking bag and/or the precision of their pressing to the area they have to punch. These components will be under a tarp to protect the electrical components.

7.1.2.1) Piezoresistance Sensors

The piezoresistance is the change of electrical resistance when a force is applied to it. These use a Wheatstone bridge to measure the change of resistance. It is very sensitive to changes, low cost, and resistant to many diverse changes. The drawback to this type of sensor is that it requires a lot of power for it to work.

7.1.2.2) Piezoelectric Sensors

Piezoelectrics use materials, such as quartz crystals or specially formulated ceramics, which generate a charge across the faces when pressure is applied. A charge amplifier converts this to an output voltage proportional to the pressure. A given force results in a corresponding charge across the sensing element. However, this charge will leak away over time meaning that the sensor cannot be used to measure static pressure. Low power and robustness. The drawback is that they are very complex, and they can only be used for dynamic pressure measurement.

7.1.2.3) Strain Gauge

A strain gauge is a mechanical way to measure the way the material changes when force is applied to an area. The possibility is using four long linear strain gauges to cover vertical, horizontal, and each diagonal sides, sharing the middle as the accuracy measurement of the hit, of an area to help the user to improve their precision and accuracy in each hit, as well when the user is hitting the area, this could tell the user to use more force to keep the same amount of force in each hit.

Strain Gauges are a possibility since these types of sensors are robust and their whole purpose is to feel the change of movement of an area. This sensor will be attached to the tarp and due to the force and change of strain this component will be able to sense it and send this change to the software to count it as a hit. If used, it would be ideal to have a 45 degree stack plus a horizontal in each area or a Membrane Rosette Strain Gauge since this one covers a more circular area than having four linear straight gauges.

7.1.2.4) Inertial Measurement Unit (IMU) Sensors

IMUs will allow us to know the direction and force of every hit on the target on side B of the design. An Inertial Measurement Unit (IMU) is a type of device that includes gyroscopes and accelerometers to measure angular rate, force, and force. An IMU works by collecting the raw data of what surrounds the device, and tells its user about the X,Y,Z surroundings of the device. Some uses of IMUs include robots, drones, video game remotes, aircrafts, antennas, GPS, sports applications, and virtual and augmented reality. IMUs are beneficial for our project since it will help the user tell the force and direction in each hit. The IMU will sense to move in consequence of the hit from the user and it will be recorded each time and then it will be processed by the MCU to make each hit clear and different from each other. The type of sensor that would be beneficial to our project would need to have the capabilities to sense sports movements and/or motion tracking and gesture location.

7.1.2.5) Accelerometers

In case that the IMUs on side B do not work as desired, accelerometers shall be used instead. Accelerometers measures the acceleration forces applied. In this case, it shall tell us the force that is applied to the hit area. Uses of accelerometers include cars, machines, buildings, process control systems, safety installations, and airbags. They will be satisfactory backup options for the devices if IMUs do not work; however, they only report force magnitude of the impact, not the direction. This information will be able to be manipulated to calculate desired results, regardless.

7.1.2.6) Binary Sensors

For Side A, we shall need five binary sensors. Binary sensor is a sensor that returns a 0 when it is not sensed and returns a 1 when it is sensed. Every switch and button can be considered a binary sensor since it gives a 1 when is pressed and 0 when it is not pressed. Some applications of binary sensors include buildings, cars, electronics, and robots. Binary sensors are the best option for this part of the device since it is basic but sufficient for the desired use and user input.

7.1.2.7) Textile Pressure Sensors

A pressure sensor works by measuring the physical force when it is applied to a certain area. Some piezoresistive material is used between two non-conducting, sturdy exterior materials. In the case of this sensor, they can cover both sides of the project since this applies to both, and it can cover the areas and difficulties very easily. The piezoresistive textile aspect can help absorb the shock produced by the impacts and it will not be easy to break. These types of sensors also recognize different types of forces applied to them and it can easily hold a bullseye design with at least three rings of difficulty for side B. For side A's sensors, this project can be designed, cut, and created very easily as well.

7.1.3) Design Matrix and Analysis

This design matrix summarizes the degree to which each sensor option presented in the previous section meets our technical standards.

Standard:	1.1	1.2a	1.2b	1.3	1.4	1.5	Total
Accelerometers	2	5	4	1	5	5	22
IMU	1	4	5	5	4	4	23
MIMU	3	3	3	4	1	1	15
Strain Gauge	5	1	2	3	2	2	15
Rosette Strain Gauge	4	2	1	2	3	3	15
Binary Sensors	6	6	1	1	6	6	26
Pressure Sensor	7	7	7	7	7	7	42

Table 3: Design Matrix for Sensors on Side B

We will be discussing textile pressure sensor and binary sensor since these had the highest ranking among all the sensors.

In **1.1**, the whole purpose of the textile pressure sensor and binary sensor is to be hit since it will signal the processor about the signal.

In **1.2**, these sensors can be made into separate sensors to make it work using the same materials for Side A but for the case of the binary sensor, it will need a foam surrounding the sensor that will bounce back faster. For side B, The pressure sensor material is the type that will return back to its original value while the binary one would not work for the side B unless there is a type of foam that will help it to bounce back

In **1.3**, the pressure sensor can be categorized into different regions to read how far a hit was from the target using a voltage signal that can be interpreted by the software. The binary sensor would make this task relatively more difficult since the binary sensor cannot be as easily divided into discrete areas to read from.

In **1.4**, both the binary and pressure sensors are able to function with low power, similar to the power needed to make a press button function correctly.

In **1.5**, each sensor type is able to cover an entire zone using only one sensor.

7.2) Indicators (Natesha)

This section discusses the research and decision-making process to choose our indicators. These must meet as many of our technical standards as possible, and present a cost-effective way to communicate information to users.

7.2.1) Technical Standards

The indicators used on this device should:

2.1	produce light with high enough intensity to be visible behind a translucent cover
2.2a	(each individual unit should) be durable enough to withstand repeated hits
2.2b	(connections between units should) be durable enough to withstand repeated hits
2.3	be able to easily shape as needed without excessive strain
2.4	sufficiently indicate zone in multiple colors with as few physical units as possible
2.5	have a lifetime comparable to that of the device itself.

Table 4: Technical Standards for Indicators

7.2.2) Candidates for Use in Device

There are three main options for the indicators used: addressable LED strip lights, individual LEDs, and electroluminescent (EL wire). More detail about each will be provided in the next sections, before being compared directly to each other in a decision matrix to pick the best candidate for indicators in this device.

7.2.2.1) Addressable LEDs

One indicator option is to use strips or strings of addressable LED lights. These have become commonplace in recent years due to their popularity for indoor and outdoor decorative applications. Each segment of the LED strip/string requires an input voltage for power and three connectors containing the needed value of each primary color (red,

green, blue) to create the desired color. The segment itself contains RGB LED light units and a controller to determine the values for each primary LED color. The strip can be cut in between segments and attached together by connecting the input of one segment to the output of another; each segment can work individually as long as it is a part of a complete closed circuit. Since each LED is individually addressable, this approach would allow for a large range of visual effects. Additionally, the popularity of these lights means we will have a large set of features to choose from, such as mounting color, input voltage, and differing color schemes.

Some of these features come at the cost of others; a strip with a waterproof sheath will be more durable, a feature we will likely need. However, these sheathes may make our such strips significantly more difficult to shape and cut to our design. Other features will challenge us to balance their worth with their cost; strips with higher pixel density can display higher resolution effects, but at increased costs. Strings of lights are easier to shape in abstract, non-straight line arrangements than their strip counterparts, but these strings will generally come with bulky bulbs that may not be easy to mount in a way that lowers the risk of the user accidentally hitting a bulb and potentially hurting themselves. Pricing is heavily dependent on the features offered as well as the length of the strip or string. To be confident in our choice for either of these options, a design plan is necessary.

7.2.2.2) LEDs

Individual LEDs are another option for indicators: using individual LED bulbs with some kind of material within which the LEDs are scattered will allow the light to diffuse through it so that the entire panel is illuminated. This would likely be the cheapest option since each individual component has the lowest cost and the group has the largest control of exactly how they would be used. Although LEDs themselves are fairly durable due to their plastic, rounded shell that is unlikely to break when faced with blunt force, their connections to each other may end up compromised if hit incorrectly. They can be controlled directly by a microcontroller and a DC power source, so they will be fairly straightforward to program for different effects and they require little power to function (meaning that they will not heat up easily). LEDs can burn out if left on for too long, though, making the life of the indicator unpredictable.

7.2.2.3) Electroluminescent Wire

Electroluminescent wire (EL Wire) is made of a stiff, phosphor coated core, wrapped by a corona wire acting as a capacitive plate around this core and an outer PVC covering protecting the wire. EL wire, as well as derivatives like EL tape and panels, are frequently used in clothing applications due to the fact that they produce no heat, consume less power than LEDs, and can retain their shape while still being flexible enough to curl around a finger. This makes them a good fit for our project, able to withstand accidental hits by the user better than an LED based approach might be.

Since EL wire is capacitive, it cannot be controlled with a typical PWM signal. Instead, they are powered by an AC inverter, with the frequency and voltage of the AC signal dictating the wire’s brightness. The wire’s brightness is therefore dependent on the load it places on the AC source; using more or longer wires will dim them.

Additionally, the wire dims gradually over time as the core’s phosphor degrades, with a typical lifespan of 1-2 years if outdoors or exposed to direct UV light, or about 3 years if in normal indoor conditions and maybe even longer in lower light conditions. Their lifespans are also dictated by the voltage being applied to the wire, so by regulating the voltage applied, we may be able to make the wire last longer. Ideally, making the EL wire segments modular would be the best option for both profit and allowing the user to keep the device functioning as long as they would like to use it (with the bonus benefit of allowing the user to buy whichever different colors they would like to use).

To use EL wire in our design, we will need to account for either a DC to AC inverter with sufficient voltage/frequency. Alternatively, we can also connect the wire directly to an AC source and eliminate a potentially noisy inverter, designing separate circuitry to control these AC components.

7.2.3) Design Matrix and Analysis

This design matrix summarizes the degree to which each indicator option presented in the previous section meets our technical standards. Since addressable LED strips and strings are so similar, this rating will be out of 3 rather than 4 with ties between multiple elements being allowed as well.

Standard:	2.1	2.2a	2.2b	2.3	2.4	2.5	Total
Addressable LED Strip	3	3	2	1	3	3	15
Addressable LED String	3	3	2	2	3	3	16
Electroluminescent Wire	2	1	3	3	2	2	13
Individual LEDs	1	2	1	2	1	1	8

Table 5: Design Matrix for Indicators

The best scoring option for our project's indicators are the addressable LED strings, but due to the caveat of heightened potential for user to injure themselves, we will use the second highest scoring option which had a nearly identical rating but removes the element of user safety considerations. These strips control many LEDs, which a controller is able to individually address to control brightness and color. They offer more than enough brightness (2.1) and lighting control (2.4) for our project, and are relatively more durable (2.2) and long-lasting (2.5) than our other options. One potential shortcoming of these strips may be their stiffness; the thickness of some LED strips can make them difficult to bend into tight shapes (2.3), but the benefits of using this indicator far outstrip this inconvenience.

We can easily rule out individual LEDs, our lowest scoring option. While this option can offer the same potential as LED strips, we would need to hand-solder and wire each individual indicator, making this choice significantly more fragile and difficult to shape (2.2, 2.3) than other indicators. Their brightness (2.1) and ability to indicate in different colors (2.4) is limited by this as well, since we would like to avoid using too many physical units, and their individual brightness cannot be pushed too high without risk of burnout (2.5).

Electroluminescent wires came close in score to addressable LED strips. These devices are made of a phosphor-covered core wire and two thin outer wires, and produce light (the color is chosen by its plastic sheath) when AC voltage is applied. Ultimately, they are limited in their brightness (2.1), especially over long periods of time, as they lose 50% of their brightness in 3000 hours (2.5). They are quite durable (2.2) and easy to shape to our needs (2.3), but we would need to pack multiple runs of EL wire in an area to use multiple colors (2.4), while LED strips are able to show multiple colors and effects with one unit.

7.3) Processor & Memory (Joseph)

This section discusses the research and decision-making process to create our processor and memory design for this project. This design must meet as many of our technical standards as possible, and offer a cost-effective solution to our processing needs.

7.3.1) Technical Standards

The processor used by the device should:

3.1	Have sufficient input lines for all sensors and communication necessary.
3.2	Have sufficient output lines for all indicators and communication necessary.
3.3	Communicate with the UI system to provide raw data.
3.4	Require internal ROM and RAM.

Table 6: Technical Standards for Processor and Memory

7.3.2) Considerations for Components

Four main contenders for our main processor were considered. The processor is required to take in inputs from the sensors and run calculations according to the data to assess the workout after it is completed. Our processor will also be the component that starts and controls all aspects of the workout modes. The main considerations taken to choose what type of processor was most appropriate for this design are: sufficient input lines, sufficient output lines, able to communicate with the sensors, and sufficient ROM and RAM. These four considerations were evaluated individually for each processor design and points were assigned to each selection considering how well it applied to that consideration. The product with the most points at the end of this analysis was chosen. Ultimately the processor chosen was the standard MCU.

7.3.2.1) FPGA

A field-programmable gate array (FPGA) is an integrated circuit that can be configured to have certain integrated circuitry configured by the designer after manufacturing. The main difference between an FPGA and a standard processor is the hardware of the FPGA can be configured as well as the software. To encode a FPGA, a hardware description language

(HDL), like Verilog, must be used. The HDL is used to change the internal circuitry of the FPGA by assigning different components to the circuitry of the board. Once configured, SW can be loaded onto the FPGA and it can be used as a flexible MCU to perform calculations.

When evaluating the design considerations with the FPGA, the FPGA can be standardized by its compatibility with our final project. The first consideration to be evaluated is the number of input lines for the processor. The FPGA has User I/O pins that can be programmed to be input or output pins. Given the flexible nature of the FPGA, the pins themselves are designed to operate multi directionally. Although the pins have this flexibility, there are not as many total I/O pins on the standard FPGAs, certainly not as much as other options considered. The next design consideration is the ability to connect to and communicate with the sensors. FPGAs are compatible with connections such as UART, I2C, and SPI, making it compatible with the sensors used in the machine. The last design consideration evaluated is the availability of sufficient RAM and ROM to store the data from the sensors. FPGA boards have RAM and ROM of sufficient size, at around 256MB of RAM. Although many of the considerations are met with the FPGA, the overall cost and lack of I/O make this a lower contender to be included in the final design.

7.3.2.2) DSP

A digital signal processor (DSP) is a specialized microcontroller chip that is optimized for the needs of digital signal processing (DSPg). These processors are especially and widely used in fields such as audio signal processing, telecommunications, and in consumer electronics such as mobile phones. The main use for these processors is to measure, filter, and compress analog and digital signals. It does this by first converting the analog signal to digital, then applying processing filters, then converting the output digital signal to analog signal. Although DSPg algorithms can be run on any microprocessor, the DSP offers advantages in areas such as data compression that make DSP favored to use for this operation when it is the main operation for a system.

The first consideration to be evaluated is the number of input and output lines. There has to be sufficient I/O to collect the data from all of the sensors simultaneously. DSP processors have sufficient GPIO pins and 3 full UART interfaces, four SPI connections, and three I2C interfaces. The DSP has sufficient I/O to incorporate all of the sensors into the microcontroller. Given that the sensors communicate using UART, I2C, and SPI, there is enough I/O to healthily support the data collection.

The last consideration is the amount of RAM and ROM that the processor is able to support. The DSP has up to 4GB of expansion, with an onboard amount of about 1024KB. This is enough expansion to store the data from the sensors. The DSP meets all the requirements for the project, however because it is specialized for signal processing, the DSP does all of these functions while being more expensive than other, non-specialized processors.

7.3.2.3) Raspberry Pi

The Raspberry Pi is the most all-in-one processor that was selected for consideration. The Raspberry Pi is essentially a small computer included with a chip, I/O, and a specialized power input. The Pi acts as a desktop computer on a chip, even being able to load an operating system on the chip. The Pi is usually used for consumer grade projects where an operating system can be beneficial to add more complexity and control to an electrical project. The raspberry pi is the second most expensive processor at \$50.

The PI has 40 standard GPIO pins used for I/O. There are also standard USB 2.0 and 3.0 ports for I/O. Even though there is an adequate amount of I/O. The key functionality of SPI, UART, and I2C is missing from this processor. This is enough to disqualify the raspberry pi from consideration, as the sensors communicate using the UART, SPI, or I2C protocols. The Raspberry Pi is not the component to be used.

7.3.2.4) MCU

A microcontroller is an integrated circuit that is designed to perform specific operations for a closed system. The standard components included with a microcontroller are the processor, memory, and general I/O. These are typically all included together on a single chip. Microcontrollers are primarily designed for embedded tasks that are automatically controlled. These tasks include anything from automobile engine control systems to implantable medical devices. Any time an automatic aspect of control is needed a microcontroller is generally an acceptable processor to use.

To evaluate the MCU, the first consideration of I/O lines needs to be addressed. There is a large amount of variability of I/O pins for different microcontrollers. The amount of GPIO pins that a standard MCU has is around 50 pins. This is a sufficient amount of I/O pins for the assigned tasks. The next consideration is the ability to communicate with the sensors. With 2 I2C ports, 4 UART ports, and 2 SPI interfaces, there is plenty of room to include all of the sensors for the machine. The final consideration is the amount and availability of RAM and ROM. With a ROM of 128KB and RAM of 8KB there should be sufficient memory to store the data from the sensors. Expansion of RAM and ROM is also available with the microcontroller. Due to the relatively low cost and the satisfaction of all of the considerations, the MCU will be chosen as the processor for the project. There were two different MCUs ultimately considered: And MSP430 platform MCU (MSP430F5528) and an AVR MCU from the ATmega platform (ATmega2560).

7.3.3) Design Matrix and Analysis

This design matrix summarizes the degree to which each processor option presented in the previous section meets our technical standards.

Standard:	3.1	3.2	3.3	3.4	Total
FPGA	4	4	3	1	12
DSP	5	2	1	2	10
MCU: AVR	2	5	4	5	16
MCU: MSP	3	3	5	4	15
Raspberry Pi	1	1	2	3	7

Table 7: Design Matrix for Processor and Memory

3.1 is the power analysis. The FPGA and DSP require low amounts of power. The MSP needs a supply voltage of 1.8V-3.6V. The AVR MCU needs a supply voltage range of 4.5V to 5.5V. And Finally the raspberry pi needs AC voltage from the wall. For power efficiency it is better for the voltage to be low, as this decreases the amount of power from the wall.

3.2 is the compatibility analysis. There are several considerations for the compatibility of this project. The main considerations are the number of I/O pins the processor can support, including communication lines, and the support for included libraries that helped to reduce the amount of work for the programming considerably. Because the AVR MCU was the only processor that had the capability to work with these libraries, it got the highest score on the compatibility section. Besides the support for AVR libraries the MCU has 86 GPIO pins with multiple different forms of communications in those pins with interrupts on all the GPIO pins. The FPGA has very fast switching times and is capable of performing all of the actions needed for the project. The MSP does not have proper AVR support so the libraries cannot be used and it does not have the switching capabilities of the FPGA. Finally the DSP and Raspberry Pi do not have the proper GPIO pins outputs necessary for the project, they are also not built for switching purposes which is the main technology behind our project.

3.3 is the cost analysis. The processor with the highest cost will receive the lowest score. The DSP processor cost anywhere from \$150-\$300 giving it the highest cost. The FPGA cost anywhere from \$15-\$120. The raspberry pi is \$50. The AVR MCU is \$15. The MSP is \$9.

3.4 is the implementation time analysis. The implementation time includes both the time it takes to assemble the components into a usable PCB prototype and the time it takes to load the project code onto the PCB successfully. Due to the AVR libraries and the surface mount style of the package, the AVR MCU gets the highest score in this section. The MSP is rated at a four as it is composed in C which is a familiar language and it has the same package as the AVR so it is able to be assembled onto a PCB. The FPGA is coded in VHDL which is mostly unfamiliar and it also is a surface mounted device. The DSP is a surface mount and uses C programming.

Overall the AVR MCU was selected to be our MCU as it has a sufficient number of I/O ports, adequate memory, and it is compatible with AVR libraries which simplify the code.

7.4) UI System (Hannah)

This section discusses the research and decision-making process in designing a User Interface (UI) for users to interact with our product. This interface must meet as many of our technical standards as possible, and offer a cost-effective, user-friendly way to control the device and see users' results.

7.4.1) Technical Standards

The device's user interface should:

4.1	Provide completed results to the user including number of total targets, number of total successful hits, total time per mode run, and average time per hit.
4.2	Receive user inputs such as turn on/off, select mode, and select level of difficulty.
4.3	Be capable of communication between itself and the base.

Table 8: Technical Standards for UI System

7.4.2) Considerations and Candidates for Components

Our UI System design must accomplish three goals in order to meet our needs; it must display results and messages to a user, receive user input, and communicate with the base system of our device to transmit these results and input. Unlike the other Technical Component Research subsections, there are no pre-made solutions for us to choose between for this component. Instead, this section will discuss the research and design process in creating a multi-part device to meet all of these goals.

7.4.2.1) Displaying Results and Messages to User

One of the main concepts that makes the workout bag an interactive design is the ability to show results of the workout session to the user in a simple and effective way. To do this, several types of ways to display results were examined and compared.

A simple option to provide the user with results would be to incorporate a USB port that the user can download the results from. This option would require the user to acquire a flash drive and a machine to view the results. Because we want our design to be self contained, meaning the user shouldn't need any other devices to use all the functions of the workout bag, this option is less favorable as the only method to provide the user with results. If there is extra time, our team can quickly implement this method as an additional function.

Another approach to provide results to the user, would be to incorporate a screen that displays results. There are several different types of screens that are comparable. The two types considered were a LED monitor and an LCD display. The LED monitor supports inputs of VGA, DVD or HDMI. This would require a processor that is fast enough along with a framer chip to support HDMI output. The LED monitor would be viable, if the processor in

the handheld remote is a beaglebone (or similar), however, the cost of the LED monitors is far above our budget for displays, roughly \$200 to \$300. The LCD display (2004) is a compatible display with the microcontroller (by using I2C or SPI) and is within our budget. The LCD displays also come with a display driver, therefore no additional driver chips are necessary. Although the display can only support 4 rows with 20 columns, it is sufficient for the data that is being displayed.

7.4.2.2) Receiving User Input

In order for the workout bag to be an interactive design, there must be a way for the UI system to receive user inputs. User inputs include selecting the workout mode for the session, selecting a difficulty level, and a way to begin, cancel, or exit the workout session. To reduce the complexity of the design, the types of user input options considered are limited to switches, buttons and LED lights, instead of considering any touchscreen inputs.

The best option to receive the user input would be a push button for each type of user input, instead of toggle switches. If toggle switches were used, each type of mode and difficulty setting, resulting in almost twice as many toggle switches as push buttons. In order to reduce the amount of push buttons, the options will need to be cycled through each time the button is pushed.

The user input also requires a way to indicate to the user which selections they have made, since the push buttons have to be pressed to cycle through the options. Two methods to complete this would be to have an LED light that is enabled by each option or have the option displayed on the LCD screen. Having an LED enabled for each option would require several extra components that use up several of the GPIO pins, therefore displaying the option on the LCD screen is the best choice.

7.4.2.3) Communication between the Base System and the User Interface

The UI system of the workout bag requires a way to communicate with the base system of the design. One way for the base to communicate to the UI system would be a wired connection, however, due to the amount of physical movement around/near the base system, there is a high possibility of the user tripping over the wires and causing injury to the user or damage to the device. Several different approaches were examined and compared.

There are several types of wired connections that are comparable. The two compared for this project were SPI and UART. UART stands for universal asynchronous reception and transmission. The advantages of using UART are the simplicity of implementation and operation, the lack of need for a clock, and the use of error detection (parity bit). However, UART's frame sizes are limited to 9 bits and transmissions speeds are relatively low. Another disadvantage is that each chip's baud rate must be within a certain percentage of each other to avoid data loss.

SPI stands for serial peripheral interface. The advantages to using SPI is that the implementation and operation are also very simple. SPI is also able to send data continuously due to not requiring a start and stop bit for data transmission. SPI also has a

faster data transmission than UART. However, there are downsides to using SPI as well. Such as the lack of flow control, receiving acknowledgments, and the use of more pins limiting the number of devices the chip can support.

There are also several types of wireless connections that are comparable as well. Three types of wireless communication types that were compared are WiFi, Bluetooth and Zigbee. The advantages to using WiFi allow each MCU to connect to the home's router, which allows the UI system to communicate with the base without line of sight. However, WiFi modules consume more power than the other options, which is critical when the UI system is battery operated and they are usually more expensive. The advantage to using Bluetooth is its relatively low power and inexpensive, unlike WiFi. However, the range of the communication is limited to relatively close range. The advantage to using Zigbee is that there is no need for a router and can operate in larger distances than Bluetooth. Zigbee also consumes a low amount of power, which is suitable for battery operated devices. The disadvantage to using Zigbee is the communication requires line of sight to the other device, and has a relatively low data rate.

The best fit for our design would be using Zigbee for the communication between the UI system and the base system. The UI system is battery powered, therefore, Zigbee's low power consumption has a high impact on the design decision. Although the data transmission rate is relatively low, our design only needs to transmit a couple bytes of data. Because this will be our teams first exposure to using Zigbee, an alternate choice of using SPI for a wired connection, as our team is familiar with this interface.

7.4.3) Design Matrix and Analysis

This design matrix summarizes the degree to which each of the display, input, and communication options presented in the previous section meets our technical standards. The aforementioned categories will be analyzed independently in the table, according to the standard that they correspond to.

Category/Standard	Options	Ranking		
4.1: Display Results	LCD Screen (LCD2004)	3		
	USB Port	2		
	LED Monitor	1		
4.2: Receive Input	Buttons with LCD		3	
	Buttons with LED		2	
	Toggle Switch		1	
4.3: Communication Base ↔ UI	Zigbee			5
	WiFi			4
	Bluetooth			3
	SPI			2
	UART			1

Table 9: Design Matrix for UI System

Standard **4.1** (provide results to user)'s highest rank was given to the LCD screen, as it is compatible with many processors, and can immediately display results to the user. However, this is not a finalized parameter of the design, as the display will be the last component that is configured as its dependent on the processor selection and type of communication used.

Standard **4.2** (receive user inputs) was won by the push buttons w/ LCD Screen to show selection option. Push buttons are relatively easy to implement and to program a debounce method. As push buttons and LCD display units are relatively inexpensive, it is an easy and cheap way to be able to cycle through modes, while showing which sections have been made. Since push buttons can be easily programmed to cycle through modes with repetitive pushes, few buttons are necessary for proper execution.

Standard **4.3** (communication between base and UI system) is best matched in our design by using Zigbee for the communication between the UI system and the base system. The UI system is battery powered, therefore, Zigbee's low power consumption has a high impact on the design decision. Although the data transmission rate is relatively low, our design only needs to transmit a couple bytes of data. Because this will be our teams first exposure to using Zigbee, an alternate choice of using SPI for a wired connection, as our team is familiar with this interface.

7.5) Power Supply (Joseph)

This section discusses the research and decision-making process in designing a power supply for this project. This design must meet all of our technical standards, while providing a cost-effective solution to our product's power needs.

7.5.1) Technical Standards

The device's power supply should:

5.1	Connect to a standard US AC 110-120V input.
5.2	Use an AC-DC voltage converter (such as a 20V 60Hz AC to 12V DC converter) to provide power to the system's DC components.
5.3	Provide AC voltage of necessary magnitude to any components that require an AC input.

Table 10: Technical Standards for Power Supply

7.5.2) Considerations for Components

The components that require power are the MCU, the sensors, and the LED strips. These can be powered in combination with a standard AC/DC power supply along with the output power pin with the MCU. The AC/DC power supply will need to take in a 120V/60Hz signal and convert it to a 12V DC signal. This signal will be used to power the LED strips, which require 12V of DC voltage. To power the MCU, an input voltage of 1.8 to 3.6V DC is required. The 12V output of the AC/DC converter will be lowered to this level with a DC/DC 12V-3.3V converter. After the conversion, the voltage will be the sufficient level to power the MCU. The power to the sensors will come from the MCU output power.

7.5.2.1) Remote Power Supply and Conversion

The remote has one power supply and two conversions. The power is going to a PCB including the MCU, converters, and I/O ports. Outside of the PCB, the remote contains an LCD screen which requires 5V DC supply voltage. The MCU as well needs 5V of supply. The Xbee chip in the remote needs 3.3V supply voltage. To power these components a 6V 4xAA battery pack will be used as the power source. The original design had the input line route through onboard converters, but in the final product the converters failed to output the required amount of power necessary to power the system, so instead a regulated DC/DC breadboard power supply was used to power the MCU, I/O, and the LCD. The battery pack was included in the remote to make it mobile.

7.5.2.2) Base System Power Supply and Conversion

There are two power sources for the bag. A 100W LED strip power supply, which also had its own AC/DC conversion from the wall. This power source was only used to power the LED lights on the bag. They require large amounts of current relative to the other parts of the project so for the prototype the decision was made to keep the LED power source separate from the sensor and MCU power source. To power the MCU and sensors, the original design was very similar to the design of the remote. Two conversions were necessary: 5V and 3.3V. Again the same problem occurred during testing and a DC/DC breadboard power supply was used to power the on board MCU and the sensors in the bag. The power supply was powered off a battery pack located on top of the bag.

7.5.3) Design Matrix and Analysis

The power supply selection is highly dependent on which processor is ultimately chosen. If the raspberry pi is chosen, the included power supply is what will be used. The Multi output wall adapter has a wide selection of different output voltages and can be used with the FPGA and MCU. The 3.3V wall adapter will be used if the DSP or MCU is chosen to be the processor. The total is the same for all three power supplies as they all have very similar implementations. They all are wall outlet adapters that will plug directly into the processor unit. The Multi output PS has a micro USB outlet. Due to several of the processors use of USB-A input, no matter which PS is chosen (apart from the raspberry pi PS) a micro USB to USB-A adapter will be needed.

7.6) Software (Hannah)

This section discusses the research and decision-making process in preparing our software design for this project. This design must meet all of the technical standards for our software needs, tying the hardware components equipped in the device in with the software package our users will interact with.

7.6.1) Technical Standards and Analysis

The software implemented by this device should:

6.1	Work with both the base system and the UI system.
6.2	Receive user inputs, communications, and computational mathematics.
6.3	Include low power modes while the system waits for input.
6.4	Utilize multiple timers to control the indicators and the sequence.
6.5	Detect when a sensor registers a hit and continue the sequence.

Table 11: Technical Standards for Software

7.6.2) Considerations for Components

The code for the entire project would be split up into two groups, the base and the UI system. For the UI system, the code will need to incorporate user inputs, communication, and simple math. It will also need to be able output the results to a downloadable file (USB connection) or to a LED screen. The UI code will require reading inputs from the buttons, and sending the proper data to the base processor. Next, the code will require to wait in a low power mode until it has received data back from the base, and process the data to provide results such as, number targets, number of hits, hit ratio, selected mode and total time. Lastly, the UI code will send the processed data to either be downloaded by the USB port, or printed to an LED screen.

The base system's code must incorporate ways to handle sensor input, communications to the UI system, and timers. The base code will use a low power mode until data is received from the UI system, holding the mode of the run followed by the start signal. The code then processes which LED lights go on, and use a timer to keep the LED lit for a certain period of time. Next the code will need to read the sensors that correspond to the segment of where the LED target is. Next, the code will need to determine, from the sensor's data, if the target was hit during the allotted time, and change a register to hold the number total hits, as well as total of lit up targets. Lastly, the code will need to send the data to the UI system receiver, and return to low power mode.

As shown by the set of software standards, the programming language selection is highly dependent on which processor is ultimately chosen. The language in the FPGA is Verilog. The IDE for the MCU can compile embedded C. DSP programming uses C/C++. The language used for a Raspberry Pi is Java/Python/C/C++/C#.

8.0) Device Design Concept

This design is largely based on the experience that Natesha has had throughout years of martial arts training and teaching to students of all ages and experience levels. In order to build the necessary skills to be a well-rounded fighter, a martial artist must be able to execute their moves well, and be able to apply it in practice during live sparring matches. This concept is reflected in the dual-side design of the bag: Side A aids application in practice while Side B is geared toward refinement of the move being used.

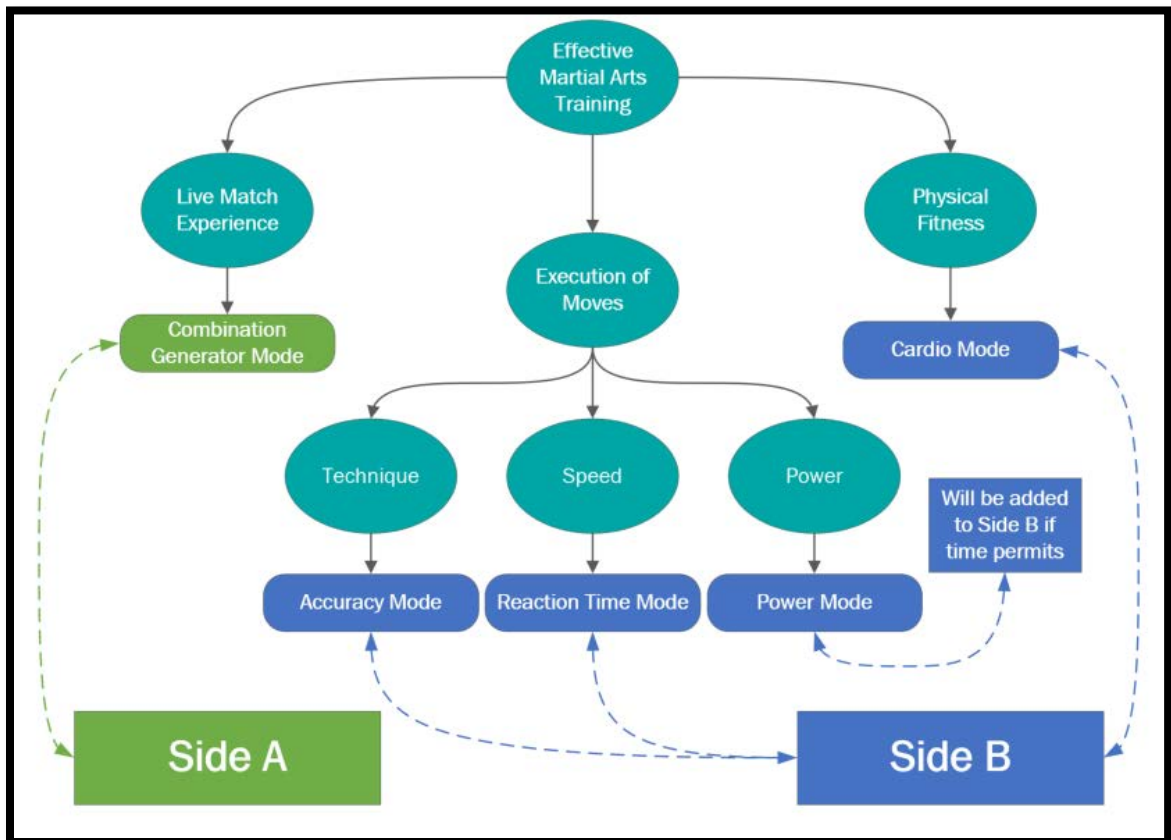


Figure 6: Association map of device active modes

In order to refine a move, such as a punch, a martial artist must first refine the technique (including how the move is executed and where the move is supposed to land on an opponent), then work on speed or timing. Power mode will be added to the device only if there is time to work on it without compromising the other determined aspects of the project. The hardware is available without adding extra budget, but it will be somewhat of a time drain, taking time that could be used to further improve existing features. This method builds on a move from the base up, so that more essential parts of execution in technique are not lost when aspects such as power and speed are added.

8.1) Functional Modes of Device's Side B

Side B was designed with these principles in mind. In the center of this side of the bag, there is a target that will be fitted with sensors and indicators to implement training modes inspired by the process of refining a move. Figure 9 shows the original concept of the target on Side B. The physical target's shape will be largely dependent on the orientation and size of the sensors, and the ability to shape the indicators around the target to sufficiently prompt the user to attack. Addressable LED strips are less conducive to round shapes, so a shape with relatively straight lines, such as a parallelogram, square, or rectangle, would likely be the most suitable for these indicators. Since this zone will include modes where the aim is to hit as close to the center of a target as possible, the best option will likely be a square, where all 4 sides are at an equal radius from the center, even if all points on each side are equidistant from the center.

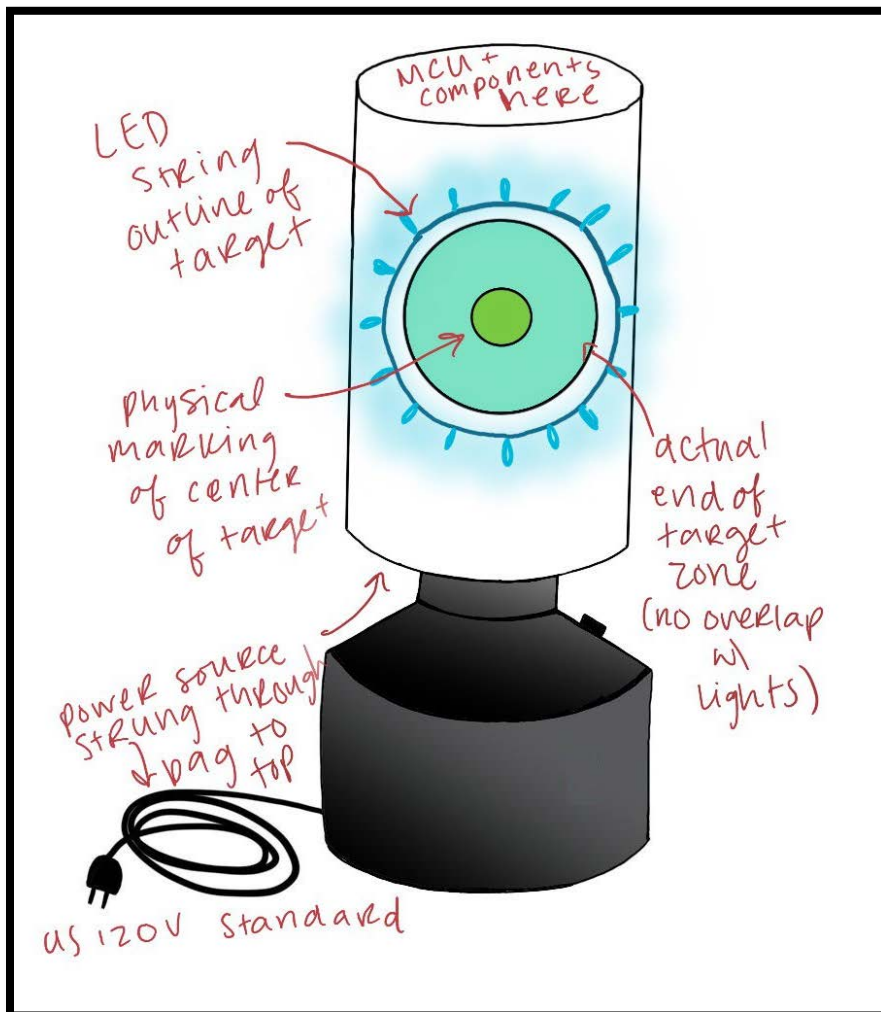


Figure 7: "Side B" of the device, Annotated

8.1.1) Accuracy Mode

One mode will test the accuracy of user actions, so that they can improve the technique needed to land their moves where they intend on an opponent. Dividing the side B sensor into four areas from innermost to outermost, the device will be able to detect the distance of a hit from the center of the clearly marked target on the bag. Using the output registers from the high performance mode, a numerical score will be calculated for each impact. After a specific amount of time, the session ends and the numerical scores will be averaged and displayed to the user so they can record this value and use it to track their progress as they train.

8.1.2) Reaction Time Mode

Another mode will test the user's reaction time using the sensors and indicators to allow the user to refine the speed of an attack on their opponent while reacting to a live stimulus as they would be in a real sparring match. The target's indicators will light as a timer is set to test the user's reaction time executing a technique. Once the target has sensed a hit, the timer will stop and register the reaction time of the user's technique. In one session, the user will have several reaction times stored that will be averaged and displayed to them. The user can save this value and may use it to track their progress as they continue their training over time.

8.1.3) Cardio Mode

As stated previously, general physical fitness is vital to the user's performance as a martial artist and athlete. With this in mind, a third mode on Side B was designed to augment the user's cardio-style warmup or workout. When using cardio mode, the user will be tasked to rapidly punch the bag for a given time period, with the goal of increasing their heart rate. The time will be determined by the user's difficulty selection: higher difficulty will provide a longer time duration. Their performance will not be measured in terms of accuracy or reaction time as in the other workouts, but instead by measuring how many hits were detected in the given time. Results will be reported to the user in the form of total hits counted and hit rate per second. The user can save their results, if desired, and may use it to track their progress.

8.2) Functional Mode of Device's Side A

While Side B focuses on refinement of the user's move execution, it lacks in getting the user in the mindset of using their skills directly against an opponent. Side A was designed to supplement Side B's refinement with the ability to target these techniques at practical locations as they align with the physical figure of an opponent. Key locations that a martial artist aims for on an opponent are highlighted by the indicators, and fitted with sensors that will tell whether or not the zone was hit during an allotted time. As with Side B, the shape of the zones will be dependent on the equipment needed for their proper function. The only requirement of the sensors on this side will be to detect when a hit has been successfully landed; therefore only a binary touch function of a sensor will be required, as long as the sensor is able to detect contact at any location within the zone. Indicators in Side A have the same constraints on the design of the zones as they did on Side B: zones made of straight lines, such as parallelograms, rectangles, and squares will be optimal.

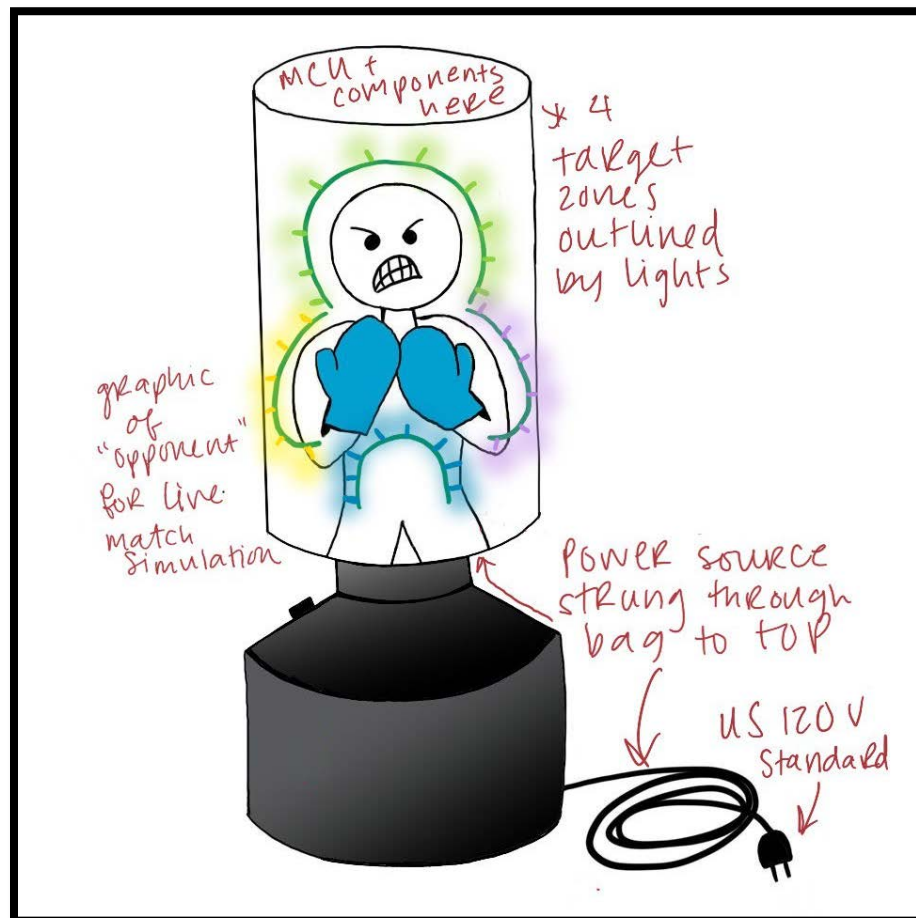


Figure 8: "Side A" of the device, Annotated

8.2.1) Combination Generator Mode

This setup on Side A will be used for a “combinations” mode which will prompt the user to hit them while the indicator is on so as to simulate target locations that open for contact on an opponent. Each successful hit will gain the user one point towards their total. After the zone is hit or timer runs out, the indicator will turn off and the next zone will soon be illuminated. Although it will operate similarly to the reaction time mode on Side B, more zones for contact provide an opportunity to expand on the skills used and further allow for growth on the skills practiced on a stationary target. The zones will light up in a random pattern, simulating inconsistently opening and closing target options while in a live match with an opponent. After a specific amount of time, the session ends and the user’s score will be recorded. The user will have the option to save the score so that they can track their progress over time.

8.3) General Performance

As mentioned in the descriptions, all modes will gather data to “score” the user on their performance quantitatively. The user will have the option to save this data and add it to the collection of their performance data. All data calculations will be performed on the base unit, and the UI system will display the mode specific results on the screen. In the event that the USB port is implemented, the user will have the option of downloading their session results to the flash drive. Otherwise, the results are cleared, and the user can select a new mode.

In addition to the active modes, the device should also employ passive modes for the time in between the active modes. One such mode will be used when the user is starting up the device and selecting a mode to use. The UI system will enter a low power mode, while the user is actively training with the base, and will exit the low power mode as once the base transmits the results of the session to the UI.

Alternatively, the base system will implement its low power modes while the user is using the remote to select the mode and difficulty. Once the base receives the mode and difficulty setting, it will exit the low power mode. The base returns to the low power mode once the results have been sent back to the UI.

8.3.1) Display/Ambient Mode

The option to add a display mode is also open, and would add a fun, homely touch to this device intended for home workouts. Within the typical home, a large standalone punching bag may be considered an eyesore in all places except the garage or home gym area. To make this device more attractive to consumers, we can program the lights on the device to create patterns with the LED lights that are fun for the user to look at or have running while the bag is not in use. This option is not necessary, but it may help the marketability of the product to consumers who may be on the fence regarding whether or not such an item belongs in their place of living.

9.0) Base System Hardware Design

This device will have two components that communicate with each other to make the user experience interactive and immersive. There will be a main device/Base System and a User Interface device. The base system will be based on the standing bag, fitted with indicators and sensors that will gather data on the user's performance. The UI device will be smaller and handheld with a space for safekeeping while the bag is in use. This will be where the user can give input (such as mode of exercise) and receive outputs (such as their exercise statistics).

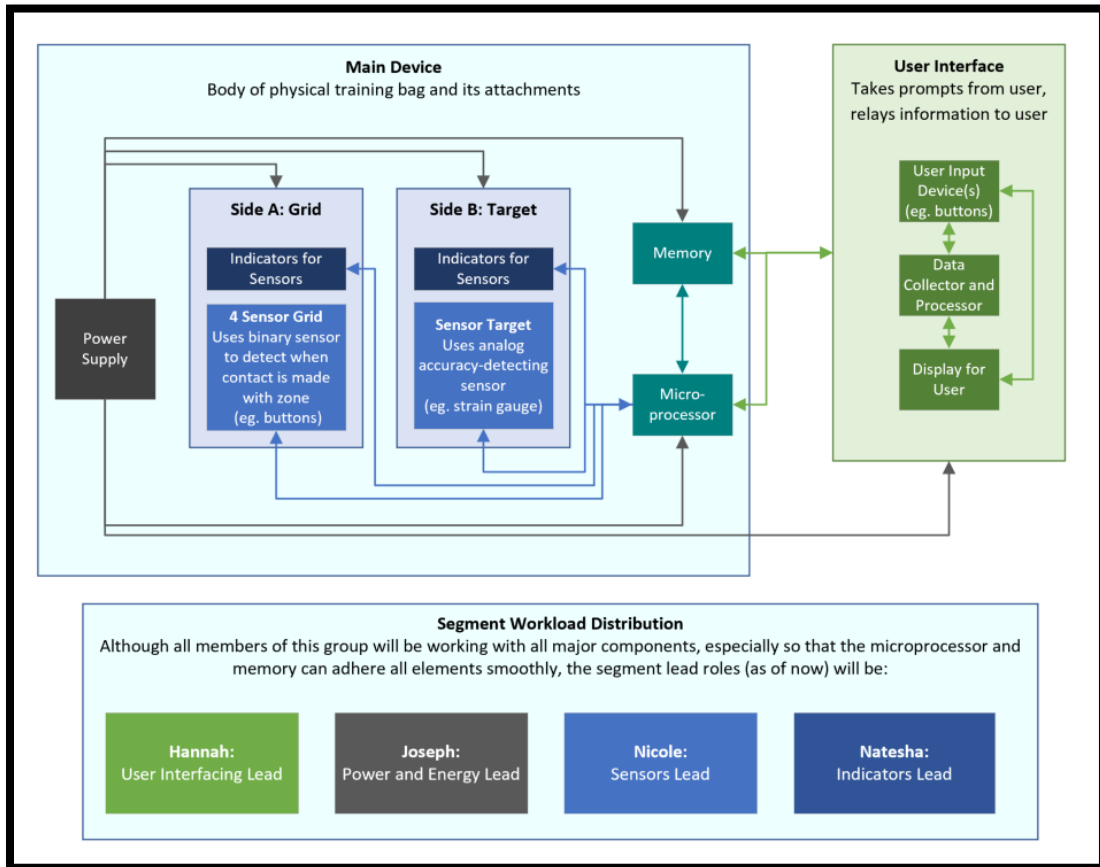


Figure 9: Hardware connection block diagram; distribution of group roles

9.1) Device Body Build and Design Plan

The body of this device will be a used Wavemaster branded bag by Century Martial Arts. These bags come in various sizes, but the most likely candidates for use in this project are the far left and far right positioned bags in the image below.



Figure 10: Century Martial Arts Wavemaster options for body base [32]

On this bag, there will be LED indicators designating the zones to hit as well as physical markings to guide the user. Side A's grid of sensors will be laid out to simulate hitting those areas on a physical opponent. Side B has a target showing the center that the user will be aiming for when using modes on this side. For each of these zones, it is required of the markers used to endure large amounts of physical contact without much deterioration of the design. These markings can be seen on the bag designs below.

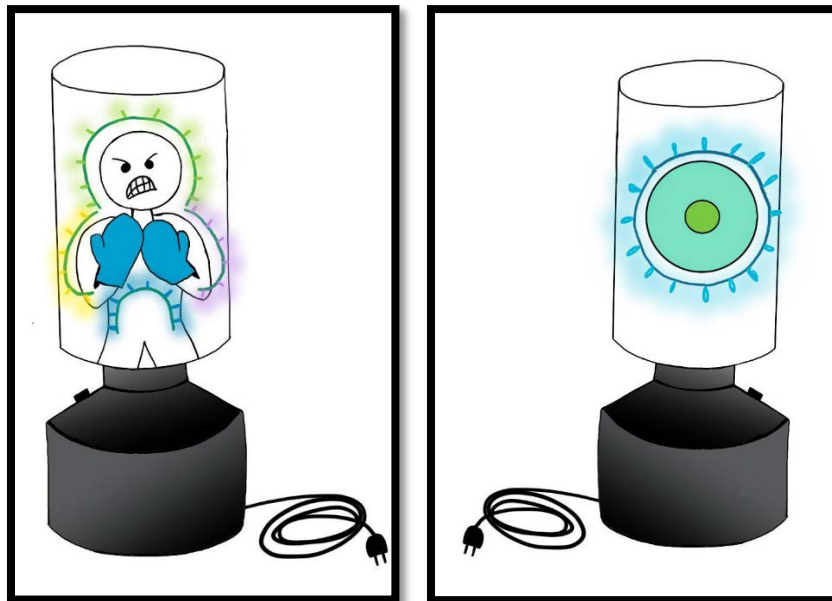


Figure 11 (Left): "Side A" of the device
Figure 12 (Right): "Side B" of the device

To create this, some mediums we could use include permanent marker, paint, tape, and vinyl decals. Permanent marker is cost effective, easy to use, and takes little dry time. Similarly, paint is inexpensive, has an intense pigment even over a black background, and simple to implement. However, extended periods of physical contact will cause the markings to erode and possibly stain the hands of the user. Tape (particularly electrical tape) can cover large zones and make many different shapes. However, the edges may start to raise after lots of contact, and bumps in the tape where the pattern curves may be uncomfortable to hit. Vinyl decal stickers may stand up to prolonged contact the best out of these options, but do not easily cover large areas. This would be ideal to use for small detail, especially in areas that will be frequently touched.

For the side A, the size of the hit areas will be determined on the side of the martial art that we get, so that the hit areas can be sized properly. For side B, the area should be broad enough to have the IMU move and record the direction and the force it moved. Since the bag we will be using for the project will be a used one since we will be modifying the martial arts from the inside as well, we will need to mention the modifications that shall be done on the inside as well. There shall be cabling for power connecting everything and there shall be a button to have an on/off for the whole system. There also should be a sturdy cushion between the point of contact of the user and the sensor to avoid damages done to the sensor and damages done to the user. LED cutouts will surround each target and the light will indicate the user which target to hit, for the program to count and to do the data science for the user interaction data.

For protection of the handheld unit while the bag is in use, a secure pocket will be securely attached to the bag.

9.2) Sensor Design: Side B

The sensor design for Side B must be a type of sensor that has the ability to tell us more about each hit that receives from the user. The sensor that was chosen for this design from the Technical Components Research and Standards Comparison section is the handmade pressure sensor. This type of sensor provides the information we need without the inclusion of sensor fusion across a grid as the smaller sensors would require.

9.2.1) Sensor Choice: Pressure Sensor

The Velostat, conductive fabric, and conductive wire were purchased from Adafruit. Any traditional fabric materials necessary for this project were purchased from JOANN'S. Three layers make up these sensors: a layer of conductive stitching, the Velostat material in the middle, and another layer of conductive stitching perpendicular to the first. The conductive layers are used as the voltage input/output.

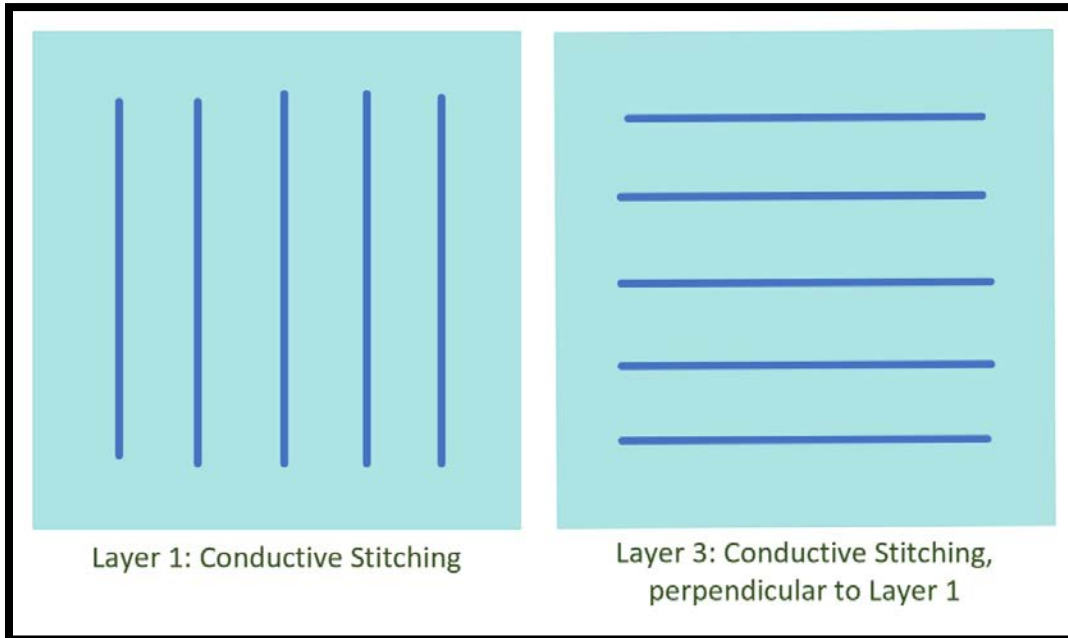
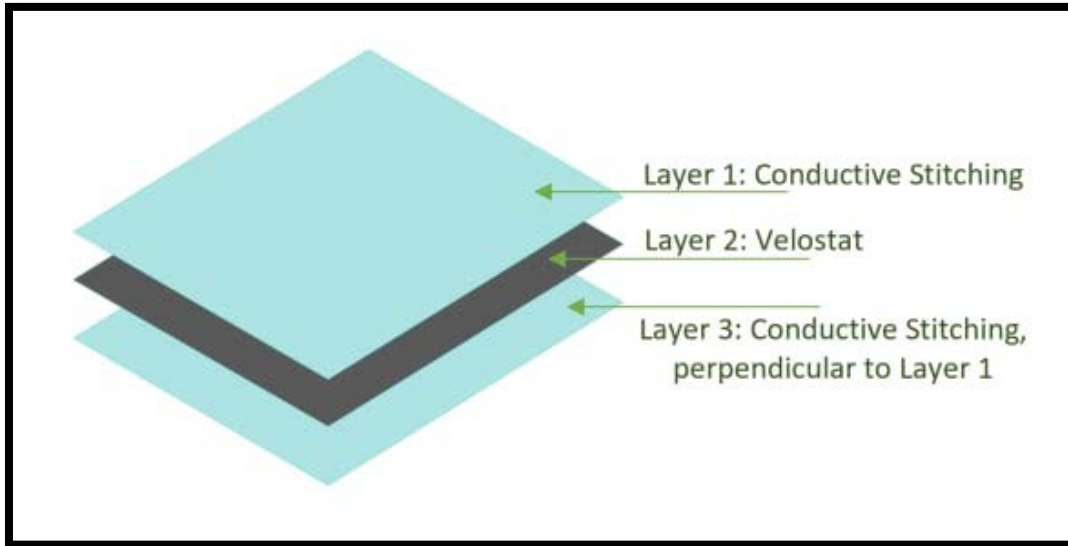


Figure 13 (Left): Layers of Pressure Sensor
Figure 14 (Right): Layers with Conductive Thread Detail

9.2.2) Side B Sensor Layout

The sensor layout for side B is unique, since we need to differentiate between the different areas of a single large area. In order to conserve conductive thread while preserving the conductive properties, several squares of conductive fabric were used as substitutes, serving as voltage sources for the sensor.

The output side of the sensor was divided into four concentric circles. The innermost circle has a diameter of 1 inch. All the outer circles are drawn to provide 1 inch of separation between each circle. Each of these circles is independently connected to the MCU, providing a clear indication of where the user applies pressure relative to the center of the circle. Plastic tape was used to keep the Velostat in place between each side of the sensor.



Figure 15: Side B Sensor Target Internals: (left) Voltage area for the pressure sensor.(right) Side B area divided between four areas for demarking the accurate areas.

9.2.3) Sensor Design: Side A

For side A, each sensor was made using stencils based on the design of the bag, in order to create sensors of the correct shape and size. Three parts of the sensor are needed: one part for the voltage input, the Velostat material in the middle, and lastly another layer of foam and conductive thread to serve as an output of electricity. This sensor was constructed a total of 5 times, one for each sensor on the final product (4), and one additional sensor used to assess and test this design before moving forward. On side A, the part on one side of the Velostat has vertical lines, and the other part has horizontal lines. This allows the sensor to best capture all interactions made to this sensor. Aside from being horizontal or vertical, the structure of each part is similar enough that they can be interchangeably used as the voltage source and the sensing surface/voltage output. Plastic tape was used to keep the Velostat in place between the two sides of the sensor.



Figure 16 : Physical Implementation of Figure 13 Sensors

9.3) Indicator Design

The indicator that was decided upon after technical research is the addressable LED strip.

9.3.1) Choosing an LED Strip

When deciding on an LED light strip for the indicators in this device, the most important requirements are addressable LED cells, RGB LED capability, and the ability to cut these strips to any length and connect these strips in any arrangement. The strip should be durable, so descriptors such as waterproof and weatherproof were positives as well.

The original most viable option we found for use in this device was the BTF-LIGHTING 5 m (16.4 ft) RGB WS2811 Addressable LED Strip reel from Amazon. This product provides sufficient length for our needs. Its LEDs are organized in groups of 3 (2" long segments), with each group able to be fully individually controlled. The strip can be cut into lengths that are multiples of 2 inches as needed, and resoldered together to make smaller sections of LED lighting, as our project requires. It is also IP67 rated waterproof, so it comes in a sturdy protective sheath.

The price for this LED strip is \$25.00 (about \$1.56/Ft) which is an ideal cost compared to many other strips we have seen, some of which have exceeded \$90 total. This strip requires a 12 V input power supply and a ground connection. Information is sent to each of the addressable LED cell clusters on the data line of the strip. This driver is compatible with our microcontroller.

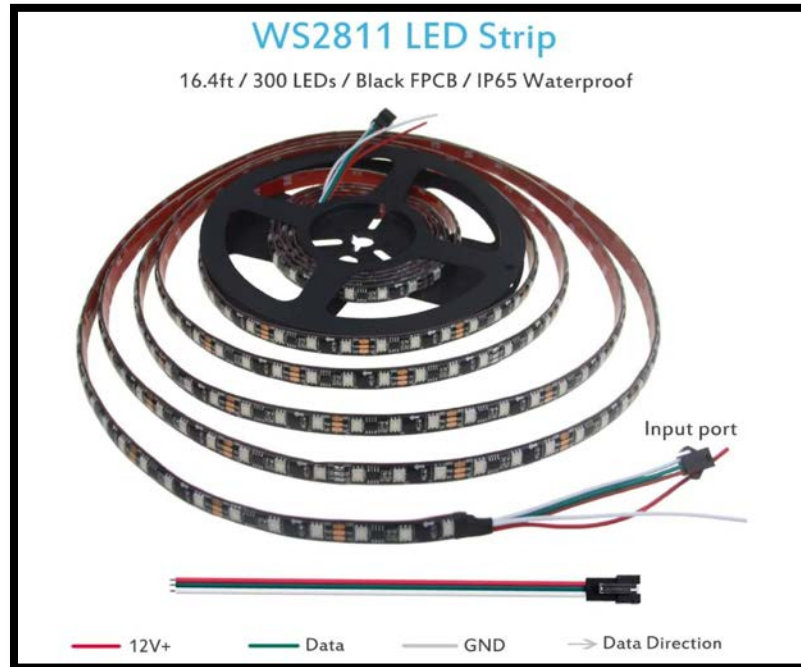


Figure 17: WS2811 LED Strip with wire ribbon configuration [2]

9.3.2) Cutting, Shaping, and Attaching the LED Strip

The LED strip chosen came wrapped in a silicone sheath, with no adhesive on it as a preliminary security method prior to permanently attaching the sheaths to the bag. Our plan to attach the LED strips to the bag was to combine stitching the lights on with extra enforcement from an adhesive such as glue, epoxy, or tape.

Once lengths of sensor borders were measured with respect to the sensor area and bag cover design, the total length of LED strip needed was 14 ft (168 in), leaving us with only 2 ft of excess. Details can be seen in [table x](#). We cut along the segmentation lines so that there were solder pads on the power, ground, and data lines available on both the data in and data out sides of each strip.

	Side A				Side B	Total
	Head	Left Arm	Right Arm	Body	(Target)	
Length (in)	40	26	26	28	48	168
# LEDs	60	39	39	42	72	252

Table 12: Indicator Specifications: LED Strip

9.3.3) Issues Using the LED Strips

We originally chose to solder the LED strips using jumper wires (approximately 24 gauge) for all three terminals. When trying to program and power the LED strips with these wires, the data would get distorted: color shades were inaccurate, solid lights would flash and change color, and certain lights within the strip would not light up. This was ultimately because the LED strip was not being powered sufficiently. Each LED needed 20 mA to run at our input voltage of 12 V, but the wires used were too thin to allow such a volume of current flowing through.

Once this problem was determined, we planned to resolve it by changing the 12 V and ground line wires to a thicker gauge wire; the data line wire was able to stay as it only sends information through signals of small magnitude. This process involved removing the necessary soldered wires from their pads, cutting and stripping short wire segments to connect power lines and ground lines, and resoldering the thicker wire to the pads. However, the solder pads were so fragile once the original wires were removed that they would tear while soldering the larger wire to the small solder pad, or while trying to bend the stiff wires into shape after soldering. We only had 2 ft of remaining LED strip at this point, so we could not proceed until getting more indicator supply.

9.3.4) Choosing the LED String

Since more LEDs had to be purchased after the group's struggle with the strips, we took the opportunity to reevaluate our original design. Bending the thick wires needed for power and ground lines on the LED strips proved to be a difficult task, and the LED strip soldering pads were too fragile to be soldered to more than once so we would have any room for modifications to our design plans. We had come across LED strings later in our search for the LED strips used, but originally ruled them out due to their stiff plastic shell to protect their interior components being deemed a risk for hurting the user's hand or foot if hit. Taking inspiration from the silicone sheath on the strip, we planned to find a soft yet firm, clear silicone-type material to encase the lights once attached to the bag. Additionally, when searching for an individually addressable LED string to use in this project, we learned that the WS2811 driver, used in the original LED strips chosen, was available in LED string form. This would allow us to use our previously developed code on the new LEDs, saving time when implementing the replacement indicator.

The LED string chosen was a WESIRI WS2811 Diffused Digital RGB LED Pixel Light. The 12 V DC option cost \$18.99 for a 3.5m/11.5 ft IP-68 waterproof string with 50 LEDs total. We retained all benefits of the driver while gaining the ability to shape the string however we

would like to as long as the cord remaining allowed soldering extra wire for connection to power source and PCB data pins.

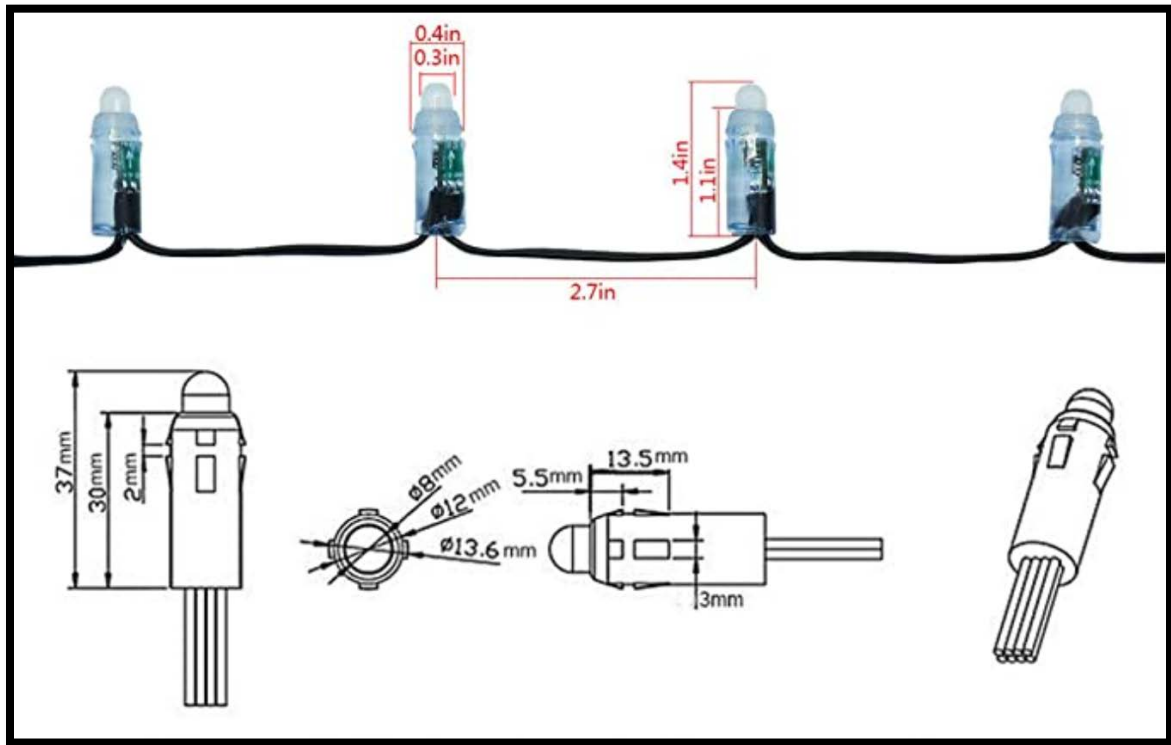


Figure 18: Diagram of LEDs on String

9.3.5) Cutting, Shaping, and Attaching the LED String

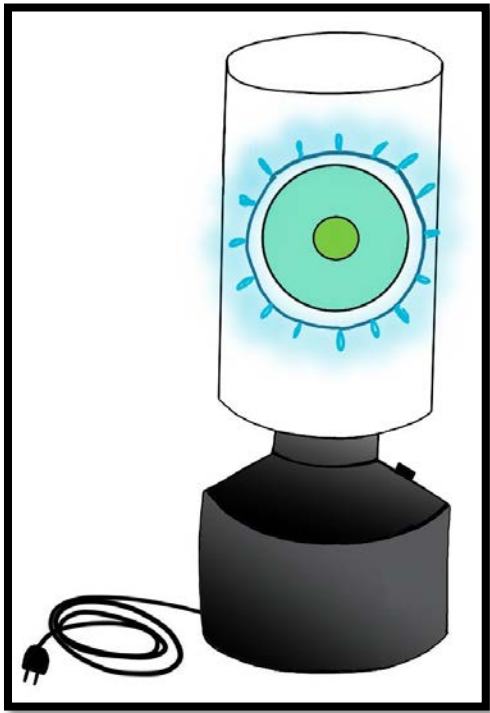
The LEDs on the string chosen were a 37 mm/1.46 in tall, 13.6 mm/0.535 in diameter bulb encased in a silicone-based shell for its waterproofing abilities. Sourcing a soft tubing with an interior diameter that can fit the lights but can still be shaped in a rounded form was rather difficult; when the string was received and we could physically feel the bulbs, we determined that the bulb casing was soft enough that that no sheath was necessary as long as the bulbs were not directly within the striking zone. To fit this requirement while allowing enough space for all striking targets and light bulbs on the cover, the design was modified to only include LEDs on select sides of the target, as seen in figures 19 and 20, or around the target in a circle instead of a quadrilateral, as seen in figure 21.



Figure 19 (left): Side A Lighting Arrangement

Figure 20 (right): Side A Lighting Arrangement, Physical

Figure 21: Side B Lighting Arrangement



The LED arrangement modification greatly reduced the length of LEDs needed to implement our design plan. Details comparing this design to the original design can be seen in table 13. We cut between the LEDs so that there was enough length on the power, ground, and data lines available on both the data in and data out sides of each LED. We used 16 gauge wires for the power and ground and used the previous approximately 24 gauge signal wires for the data wires. We used hot glue to attach the bulbs to the bag covering since hot glue was secure unless we purposely tried to remove them; epoxy may have been a more permanent solution, but the dry time and permanence of the epoxy made this option inconvenient for our needs. The wires traveled on the inside of the bag's cover to connect the data lines to the MCU and the power and ground lines to the power source, both of which are located on the top of the bag. Special consideration had to be taken to verify that the data lines were off of the sensor's outer surface so that the signals would not experience interference.

		Side A				Side B (Target)	Total
		Head	Left Arm	Right Arm	Body		
<i>Strip</i>	<i>Length (in)</i>	40	26	26	28	48	168
	<i># LEDs</i>	60	39	39	42	72	252
String	Length (in)	~ 28	~16.5	~16.5	~22	~44	~127
	# LEDs	10	6	6	8	16	46

Table 13: Indicator Specifications: LED Strip vs LED String

9.4) Power Supply Design

The power supply will originate from the AC outlet and will be converted to a 12V DC signal with an AC/DC converter. This 12V DC signal will be used to power the LEDs directly. This is necessary as the LEDs require a 12V signal to power on. The sensors will be powered using the breadboard power supply using an onboard header which is connected to the Vin line on the PCB. The MCU will be powered using this Vin line from the breadboard power supply. The Xbee is also powered from the breadboard power supply's 3.3V line.

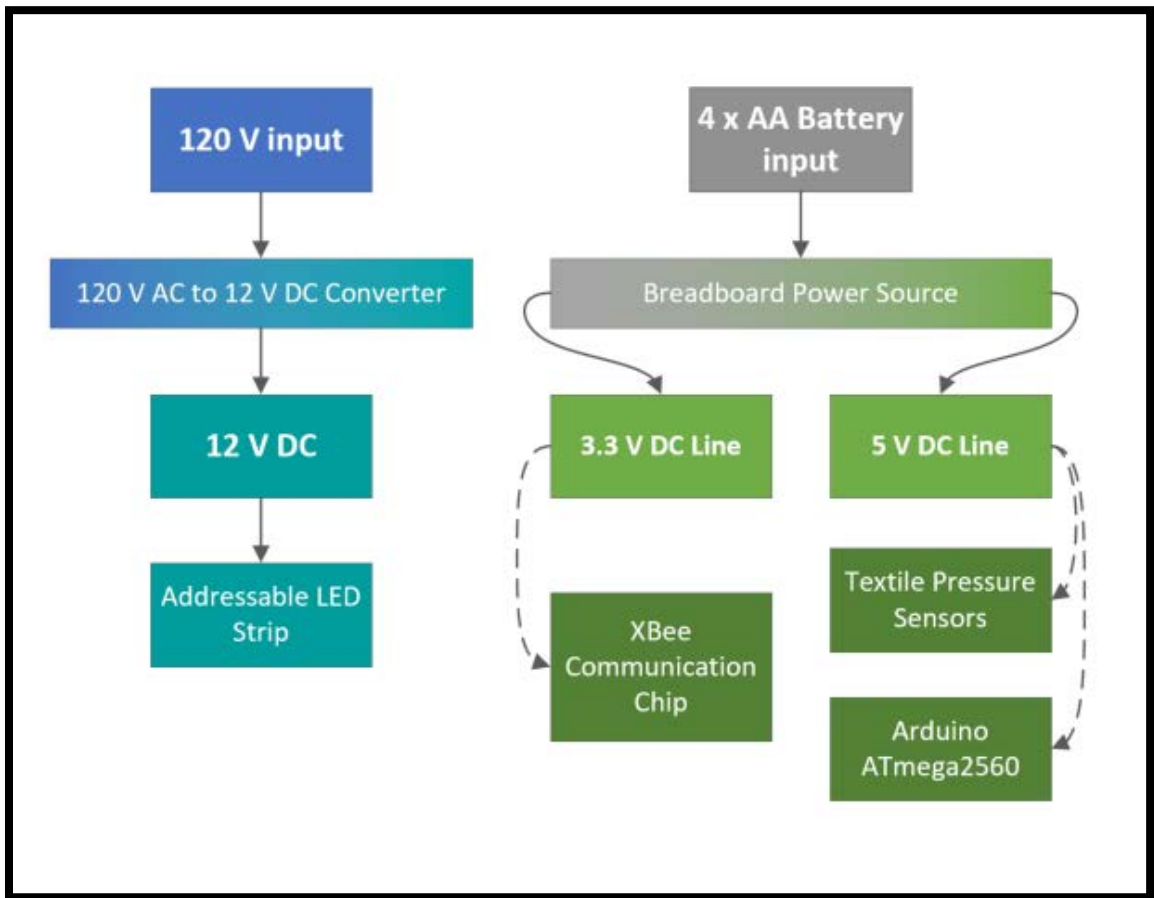


Figure 22: Flow chart for Power Supply in Base System

9.5) Processor Design

The processor ultimately chosen was the ATmega2560. This was because it has compatibility with the AVR libraries used in the code.

9.5.1) TI ATmega2560

This processor was chosen because it meets all of the technical requirements for storage, speed, ease of use, and access to I/O ports that were outlined in the research section. The relevant specifications for this component are listed in the table below. The pinout diagram for this microcontroller is also pictured in the appendix of this document.

RAM	256 KB
Clock Speed	16 MHz
Input Voltage	4.5V – 5.5V
GPIO pins	87
Interrupts	On every GPIO
SPI Interfaces	5
UART Interfaces	4
I2C Interfaces	1
Timers	6

Table 14: ATmega2560 Specifications

The processor will be used to collect data from the sensors and to use that data to calculate results according to the data and power mode. The sensors will communicate with the processor using UART, SPI, and I2C. Most of these interfaces will be used to connect to all the sensors on the punching bag. The main mode that the processor is used for in this project is switching.

10.0) UI Hardware Design

The UI system that will be used to control the punching bag will be controlled by a handheld, wireless remote. Since the UI component of this overall device is essentially its own individual device with independent hardware requirements from the base, the details of its hardware design will be discussed in this section. The remote will have all of the component sections detailed in the Base hardware section. Each part should be evaluated and picked to best meet the requirement specifications. A sketch of the first draft of the UI Remote can be seen below.

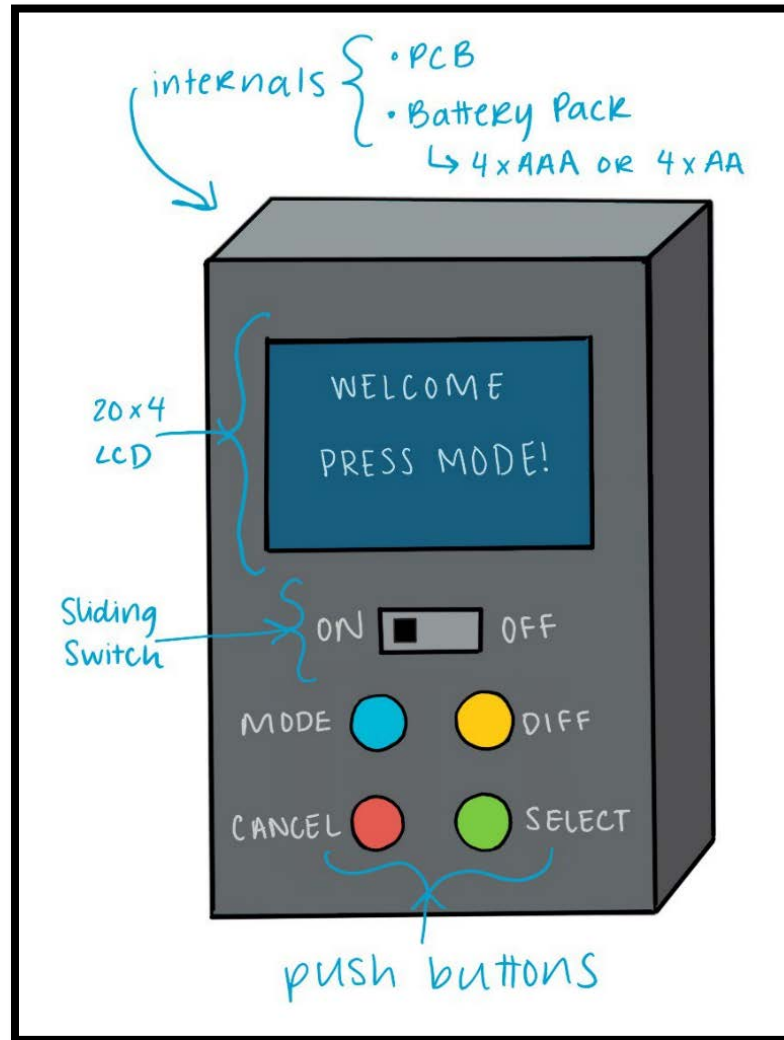


Figure 23: Draft sketch of the UI system for the device

10.1) Processor

To control the remote, a processor is required to perform the main calculations and operations based off the inputs for the UI hardware system. The processor needs standard GPIO pins to connect I/O to the MCU. The processor should also be capable of I2C communication. The I2C communication is required to talk to the LCD display on the remote. With these specifications a processor can be chosen to put in the remote. Due to the low expense and the meeting of the technical specifications, the same processor (ATmega2560) can be used for the machine and the wireless remote.

10.2) Display

The LCD display is a 20 column by 4 row screen. The MCU chip communicates with the display controller (driver) to configure the input and output pins. The LCD driver uses I2C and requires two GPIO ports.

10.3) Buttons & Switches

Push buttons will be used to control different functions for the remote. The push buttons require GPIO pins to connect to the MCU. One pin is needed for each push button, for a total of four GPIO pins necessary for the four buttons. To read a high value from the push buttons, the push buttons should be pushed in. To set this up in the correct way, the resistors for these configurations should be set to pull down mode.

Another input method needed for the remote is a sliding switch. This switch will act as the power on/off for the remote. To connect the switch to the MCU one GPIO pin is required. The resistor can be set to pull down or pull up as each would have the same functionality on a switch.

10.4) Power Supply

The processor requires an input power from 1.8V to 3.6V. The LCD display requires 5V for full functionality and the Zigbee chip is 1.8V to 3.8V. This means that a 5V DC signal is the lowest possible voltage required to power on all of the electronics inside the remote. To generate this amount of voltage using wireless methods four AA batteries will be included in the remote. The four batteries together will supply 6V, which will be enough voltage to power all the essential electronics inside of the remote.

10.5) Protective Case

The housing of the remote is required to hold all of the components inside the remote. The only components that will be accessible from the outside are the switch, buttons, and the battery pack on the back of the housing. To design and manufacture a shell that will fit these requirements a 3D printer was the original option that we would have used. The main advantages of using a 3D printer to fabricate the housing is the greater design flexibility by starting from scratch. This flexibility will allow the design to conform to any constraints that could come up during the manufacturing and designing phases. 3D printing will also greatly decrease the manufacturing time due to general ease of use. Holes will need to be cut out and sanded in order to put the final touches on the remote. The sanding is advised as the 3D print could leave over leftover scaffolding that could interfere

with the fittings and the design. The largest downside of 3D printing is attaining the resources necessary. The cost of materials can be steep if there is a need to re-print due to any errors. There is also the issue of finding the facility to print the case during a time when space is limited in UCF creator spaces. This particular downside, however, can be circumnavigated if we begin prototyping with sufficient time to find an alternative method of case construction.

To keep the majority of the benefits listed while also considering our material cost and time, we decided to go with the route of cutting the case out of Plexiglas. Plexiglas is strong, inexpensive, and allows the user to see the internal components of the remote while they are using it, which is an aesthetically pleasing feature. We modeled the case on inventor with and without vital internal parts to verify that all components would fit, before cutting the shell and holes for all external components including the LCD and the buttons.

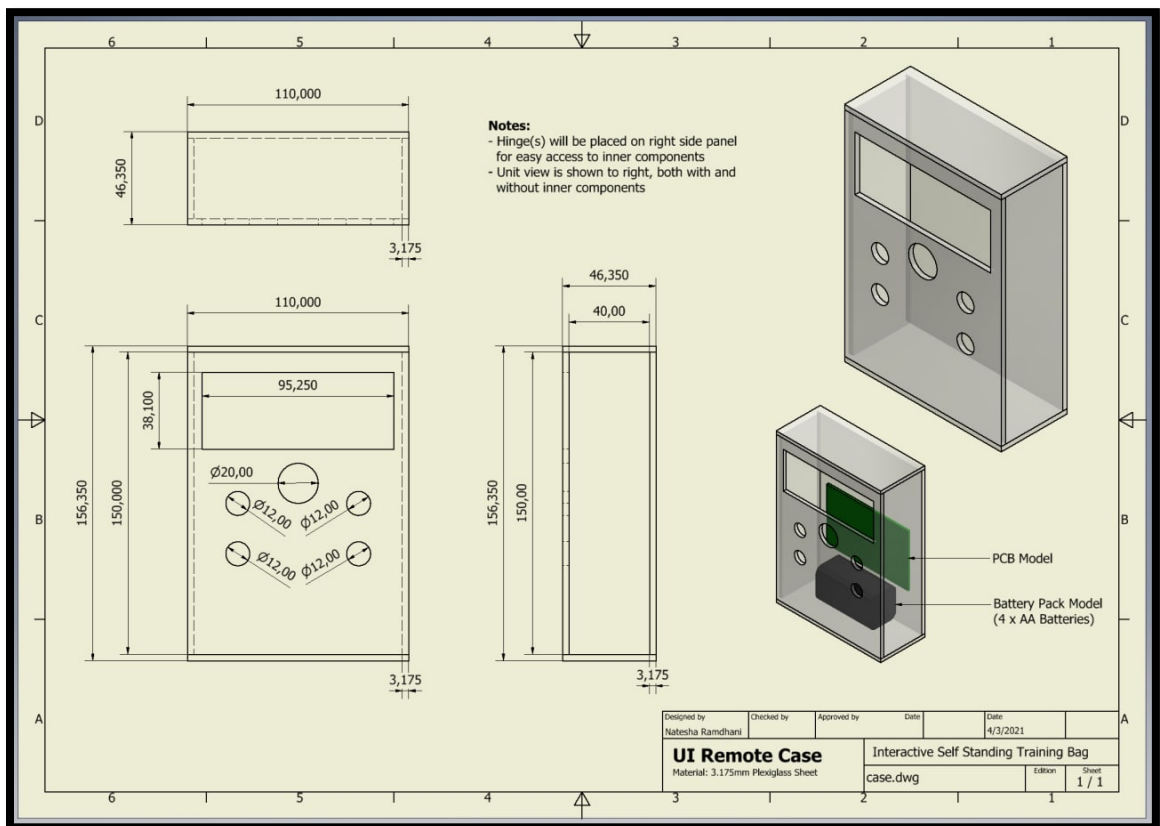


Figure 24: Technical Drawing of UI Remote Case Components

However, due to our buck converter not working as intended and the addition of an external power source, the breadboard power source, we were unable to fit the battery pack in the case as intended. With this information, we decided to attach it on the outside of the case for easy access when replacing batteries, and to house our internal components as securely as possible. The final remote design can be seen in the following images.

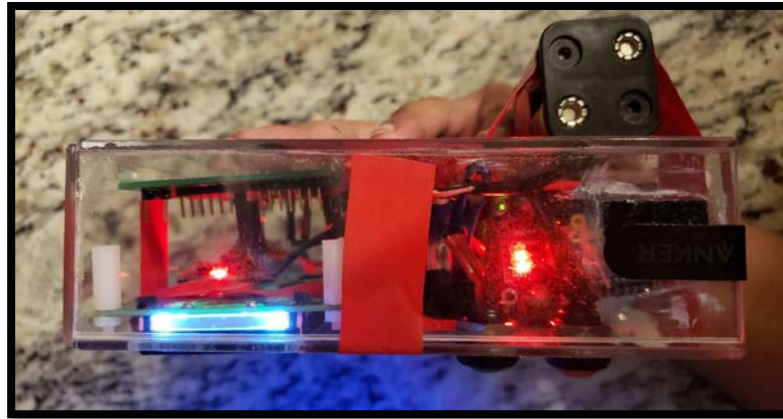


Figure 25 (top): Physical Implementation of Remote Case, Side View

Figure 26 (bottom): Physical Implementation of Remote Case, Front View

11.0) Printed Circuit Boards

Since there are two main components of this device that each require their own printed circuit boards (PCB). These PCBs will each hold one microcontroller, as well as sufficient connection pins for all power, sensor, and indicator inputs and outputs. The base system will house one of the PCBs, as this is the location on the bag least likely to get damaged by a hit. The UI system handheld component will house the second PCB.

11.1) Base System Design

A summary of the connections in the base system device are seen below. They will be detailed throughout this section. There are 16 points of interests between sides A and B: (1-8) Binary Sensor 1-8, (9) Xbee Module, (10) Power Supply and Conversion, (11) MCU, (12-15) LED strips 1A-4A, and (16) LED strip B.

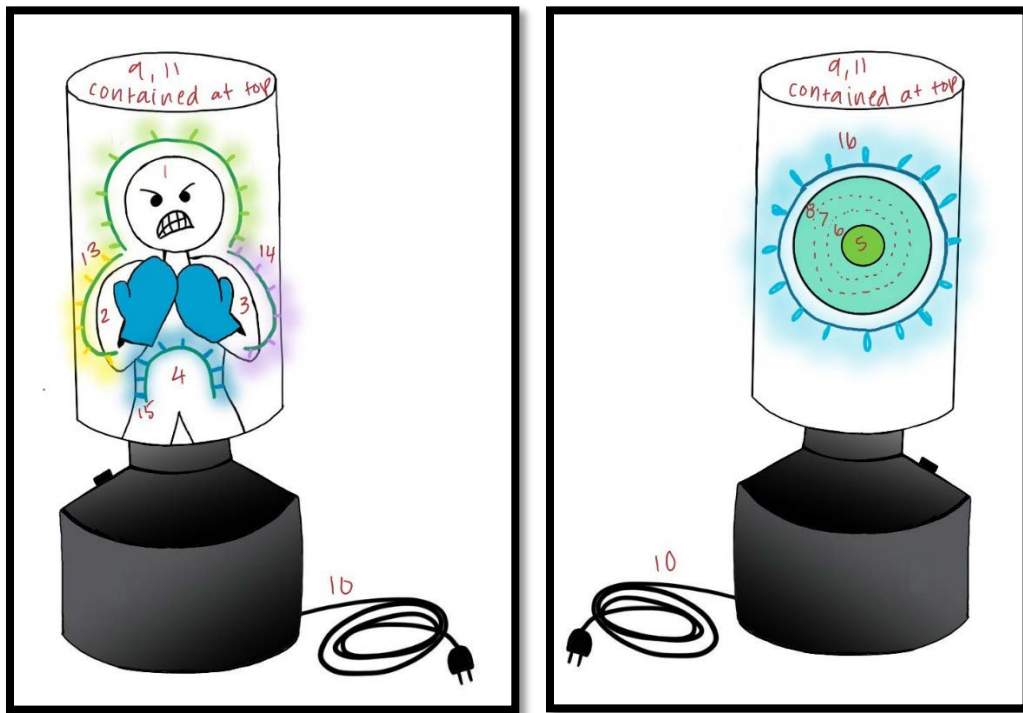


Figure 27 (Left): Side A with Component Reference Numbers

Figure 28 (Right): Side B with Component Reference Numbers

11.1.1) Pressure Sensors (1-8)

Eight pressure sensors will be used to track the hits to the five targets on the punching bag. These sensors will be used for both sides A and B. These sensors require access to one GPIO pin each w/port interrupts. The port interrupts are necessary for optimal control.

Sensor Number	Pin Name	Pin Number
1	PK7	82
2	PK6	83
3	PK5	84
4	PK4	85
5	PK3	86
6	PK2	87
7	PK1	88
8	PK0	89

Table 15: Pressure Sensor Pin Connections

11.1.2) Xbee Module (9)

The Xbee Module is a device that creates a personal area network that allows multiple different devices with these modules to communicate with each other. The module is well suited for applications with low-power and low-bandwidth needs. There will be both a Zigbee coordinator (ZC) and a Zigbee end device (ZED). The Xbee transmits its signal over the 2.4GHz frequency band. The Xbee module in our device is the ZED, to allow for the control of the system with the remote. The Xbee chip needs an input voltage 1.8-3.8V. The Xbee needs an RX and TX connection for UART communication with the MCU.

Connection	Pin Name	Pin Number
RX	PE0	2
TX	PE1	3

Table 16: Pressure Sensor Pin Connections

11.1.3) Power Supply and Conversion (10)

There are three main components to consider in the power analysis of the device: (1) The 100W 60Hz AC/ DC LED power supply, (2) a 6V/3.3V DC/DC converter, and (3) a 6V/5V DC/DC converter. The first stage in the power supply chain is the main power from a wall outlet. A wall outlet converter will be used to transfer the energy from the AC wall outlet into a 12V DC signal. This signal will be used to power the LED strip.

A DC signal will be used to power all of the electronics in the device as no functions of the device require a time dependent signal for analysis or power. After conversion, the 6V DC signal will be split into two separate signals. One of the signals will be run through a 6V/5V DC/DC converter. After conversion, the 3.3V DC signal will be used to power the Xbee chip. The second 6V DC signal will go through a 6V/5V DC/DC converter. After conversion, the 5V DC signal will be used to power the MCU and the sensors on the base.

11.1.4) MCU: ATmega2560 (9)

The MCU to be used in both the device and the UI Handheld is the ATmega2560 AVR microcontroller. The input power for this MCU is 4.5-5.5V. The input voltage for the MCU in the remote will be 5V DC. To power the MCU, the high line from a DC/DC buck converter (5V) will be applied to all of the Vcc pins on the MCU. This has sufficient communication lines and GPIO ports to accommodate all of the modules that will be added to the remote control.

11.1.5) Addressable LED Strings (10-13, 15)

Four LED strings will be mounted next to the sensors on side A. These LED strings will indicate the target for several modes of operation for side A. The LEDs need to be powered and will have to be connected to the MCU to communicate with the rest of the device. To power the LED strips, a 12V DC signal is needed. This signal will come from the 120V/12V converter covered in the power section above. The ground for all of the LED strips will be connected to the MCU's ground.

11.2) UI Handheld System Design

There are several points of interest and function on the remote used to power on and control the device. There are a total of 9 points of interest labeled on the diagram. They include: (1) MCU, (2) Zigbee Module, (3-6) Buttons 1-4, (7) LCD Module, (8) ON/OFF switch, and (9) Power Supply. The following sections review the design considerations for the electrical and mechanical design of the UI Handheld.



Figure 29: UI Handheld with Component Reference Numbers

11.2.1) MCU: ATmega2560 (1)

The MCU to be used in both the device and the UI Handheld is the ATmega2560 AVR microcontroller. The input power for this MCU is 4.5-5.5V. The input voltage for the MCU in the remote will be 5V DC. To power the MCU, the high line from a DC/DC buck converter (5V) will be applied to all of the Vcc pins on the MCU. This has sufficient communication lines and GPIO ports to accommodate all modules that will be added to the remote control.

11.2.2) Xbee Module (2)

The Xbee Module is a device that creates a personal area network that allows multiple different devices with these modules to communicate with each other. The module is well suited for applications with low-power and low-bandwidth needs. There will be both a Zigbee coordinator (ZC) and a Zigbee end device (ZED). The Xbee transmits its signal over the 2.4GHz frequency band. The Xbee module in the UI Handheld is the ZC, to allow for the control of the system with the remote. The Xbee module needs an input voltage 1.8-3.8V, so the same rail powering the MCU will be used to power the module.

To connect the power, connect the high 3.3V rail to pins 8, 11, 14, 16, 20, 22, and 24 on the module. To communicate with the MCU, the Xbee Module needs to be connected via several pins. The module can communicate using UART and SPI. The SPI connection will be used for the remote MCU. To connect the SPI comm, connect the A1 SPI sCLK pin on the MCU (MCU pin 59) to pin 28 on the Xbee. To connect the SPI MOSI data, connect the A1 MOSI pin on the MCU (MCU pin 60) to pin 2 on the Xbee with a one way connection. To connect the SPI MISO data, connect pin 1 on the Xbee with the A1 MOSI data pin on the MCU (MCU pin 61) with a one way connection. The last pin needed for SPI connection is the chip select, connect pin 62 on the MCU to pin 3 on the Xbee module. Finally the reset and VREG pins are to be connected to the MCU. Connect the reset pin on the Xbee (Xbee pin 25) to pin 57 on the MCU. Connect the VREG pin on the Xbee (Xbee pin 26) to pin 42 on the MCU. Once fully connected, the Xbee will talk with both the second Xbee module and the MCU on the remote with SPI. The module also has optional GPIO connections to use UART or to free up data from SPI. These won't be used in this device.

11.2.3) Buttons (3-6)

4 buttons will be used by the user to select different modes for different workouts on the remote. These buttons will need to be set with a pull down resistor so it will read a high input, triggering an interrupt, when the buttons are pressed. The buttons have relatively simple connections to the MCU. Each button only needs one GPIO port w/interrupt to operate effectively. Button 1 will be connected to P1.1 (MCU pin 35); Button 2 will be connected to P1.2 (MCU pin 36); Button 3 will be connected to P1.3 (MCU pin 37); Button 4 will be connected to P1.4 (MCU pin 38). Simple interrupts are the only function of these buttons so these are the only connections needed.

11.2.4) LCD I2C Module (7)

The 20x4 LCD display comes connected to an external I2C module. This module greatly simplifies the connections needed to take full control of the LCD display. It reduces the number of inputs from 4 to 2 through I2C. The I2C has 4 pins that need to be connected. The input power required is 5V DC, this will be applied to the VCC pin on the I2C module. The ground pin on the I2C module will be connected to the ground of the battery pack. To communicate with the MCU through I2C a serial data and clock signal needs to be connected to both the LCD and MCU. 2 wires and 2 sets of pins are needed. Connect the SDA pin on the I2C module to P2.1 on the MCU (MCU pin 18). Connect the SCL pin on the I2C module to P2.2 on the MCU (MCU pin 19). Once these connections are made I2C communication will be available.

11.2.5) Power Switch, Supply, and Conversion (8)

To power the UI Handheld a portable battery pack will be connected to the back of the housing. It should be accessible in order to replace dead batteries. Four AAA (or AA) will be needed to generate a 6V DC input signal. The high signal will be connected to the switch in order to establish an On/Off switch. After coming through the switch the high will be split into two separate signals. One will power the LCD display and the other will power the MCU. To power the LCD one of the high lines from the switch will be connected to a voltage divider resistor circuit to convert the 6V signal into a 5V signal. The 5V signal will then become the input power for the LCD display. To power the MCU, the other high line from the switch should be run through a 6V/3.3V buck boost DC/DC converter. This converted 3.3V signal will be used to power the MCU by connecting it to pins 11, 25, 64, and 89 on the MCU.

11.3) Schematics, Boards, and Bills of Materials

This section will contain the schematics, board layouts, and bill of materials for the two PCBs discussed in the previous two sections.

11.3.1) Base System

To connect these components to the MCU, a pin header has to be installed on the PCB, and is shown in the schematic just above the MCU. Several components need distinct voltages for their input power. The power comes from a wall outlet adapter and gets converted through an AC/DC converter into a 12VDC signal. This power goes to the LED drivers. To accommodate all the components, two DC/DC adapters are needed for step down conversion from 6V to 3.3V and 5V.

The 3.3V DC signal is used to power the Zigbee. The 5V will be used to power the MCU, sensors, and the LCD display in the remote. The components that need a 12V DC to power are the LED strips. The LED drivers are not shown in the schematic since they are outside the PCB located on the LED strips themselves in the design and thus will have to be connected to the MCU through the pin headers.

The main logic center for the bag is the ATmega2560 MCU. This chip was chosen for the bag for its compatible AVR libraries. All the components, barring the DC/DC converters, have logical connections to the MCU in the schematic. The first logical connection concerns the LED drivers. These will be connected to the MCU with on board pin headers. The ZigBee chip is also connected to the communication ports of the MCU. To communicate with the MCU the Xbee module will use an open RX and TX port. 4 connections are needed for the SPI connection: CLK, MOSI, MISO, and CS. Another logical connection to the MCU in the schematic is the four sensors on side A of the bag. These are located to the left of the MCU in the schematic. These sensors can be modeled in the schematic as a push button with a pull down resistor. For proper functionality, these sensors should be connected to a port that can use timer interrupts.

There are several components that show up in the schematic that are not included in the PCB design. The components that will be off the PCB are the LED strips and drivers, the buttons and the homemade sensors. These components will be placed in key positions around the bag in order to gather the data that will be output at the end of the session. The wires will be run through the bag and will be connected to the PCB using the outboard pin header.

A similar procedure will be used for power and ground. The homemade binary sensors will be placed behind each of the targets on side A and will be wired to the GPIO pins through the bag and eventually the pin header. A similar procedure will be used for the mode start and early end buttons on the front of the device. The last components in the schematic that will be located off of the PCB are the LED strips. The drivers shown in the schematic will be located on the actual LED strips in the final product. With the LEDs positioned around the targets, the wires to the drivers will be routed from the driver, into the bag, and eventually into the pin header on the PCB according to the schematic.

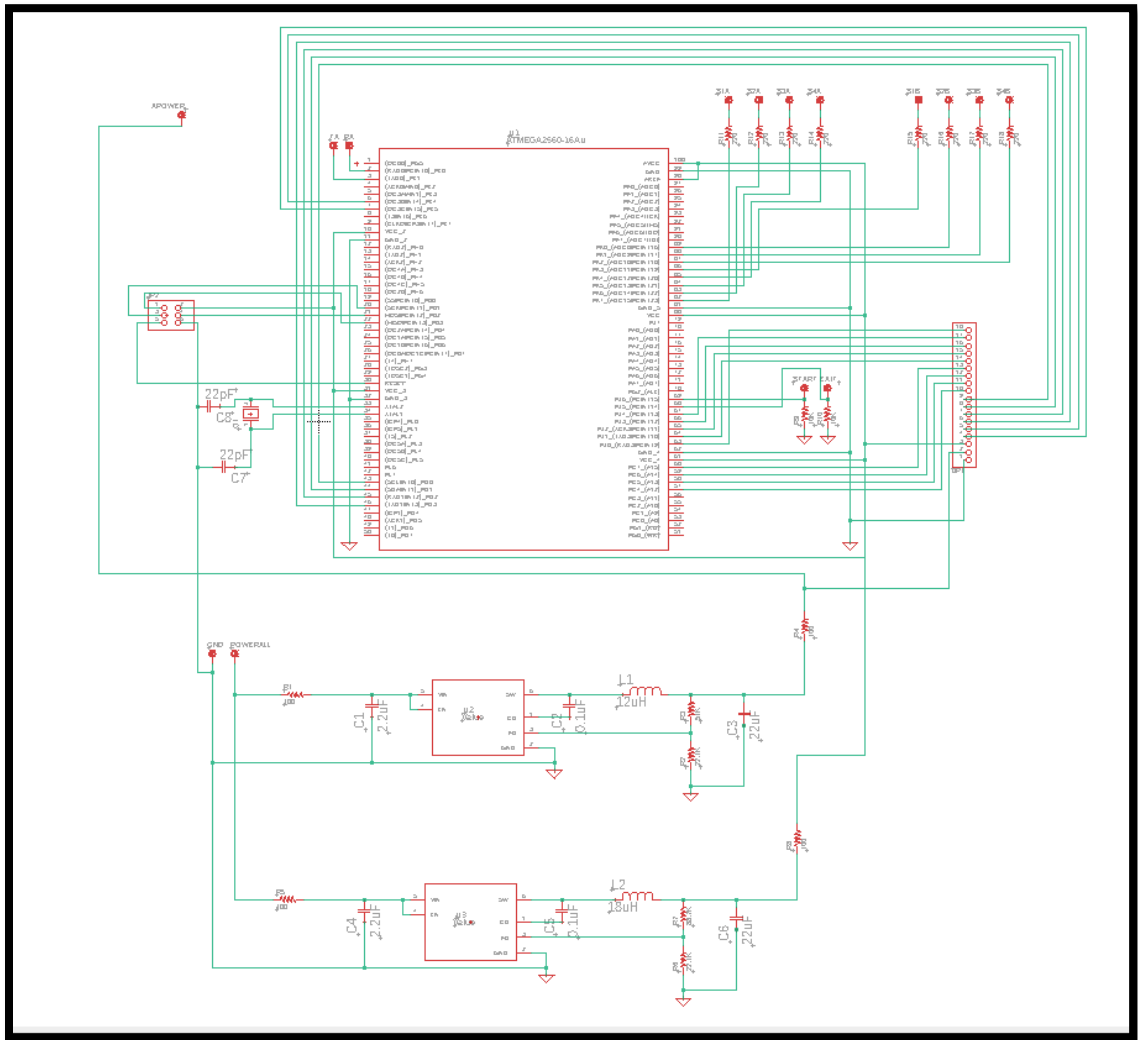


Figure 30: Base System Schematic

The last component on the PCB for the bag are the pin headers. The connections to the pin headers represent the connections to all of the other devices located on the bag. The position of the pin header is not extremely important, and was just put where extra space was left after placing the components. Surface mount resistors and capacitors are shown in application for the ICs. The .brd layout of the bag can be seen below, followed by the physical implementation of board as well as the bill of materials for the base system.

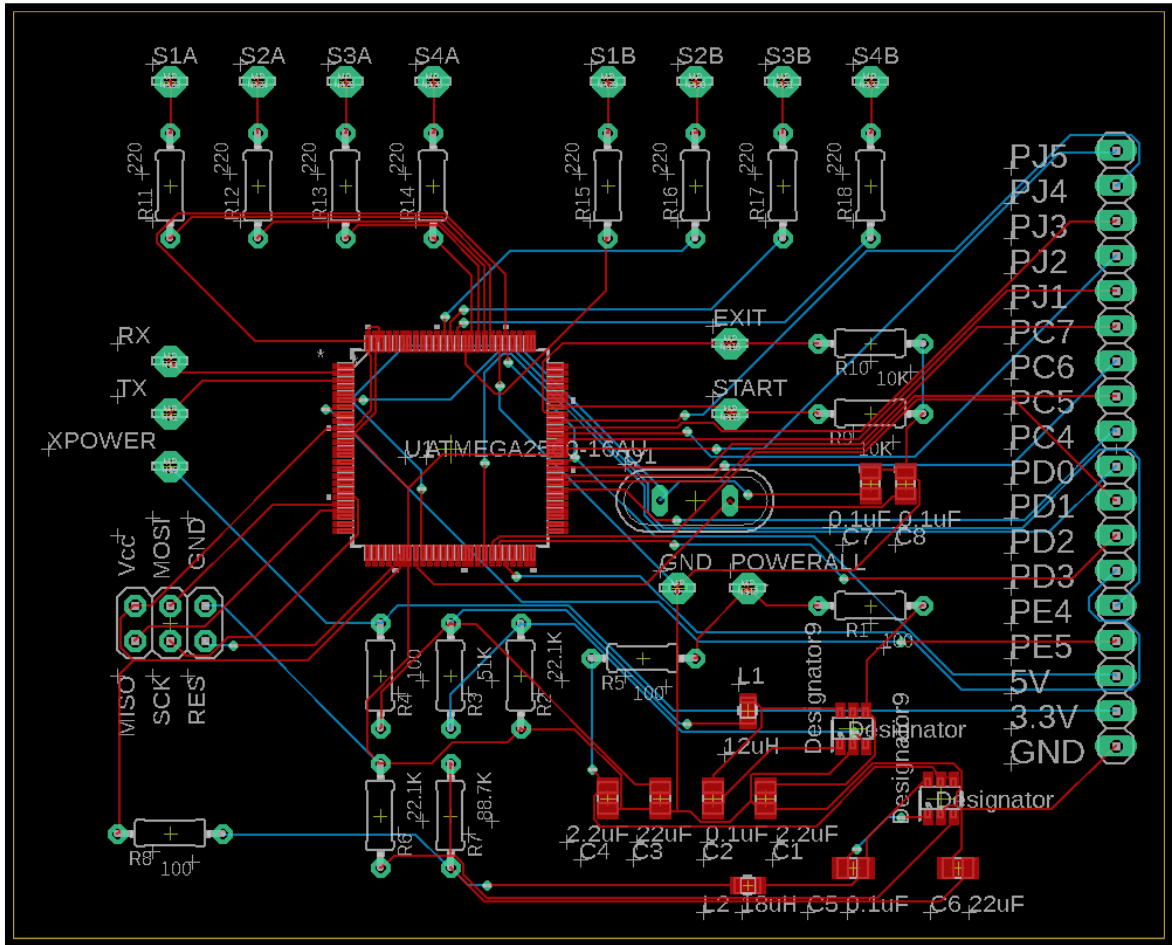


Figure 31: Base System PCB Layout

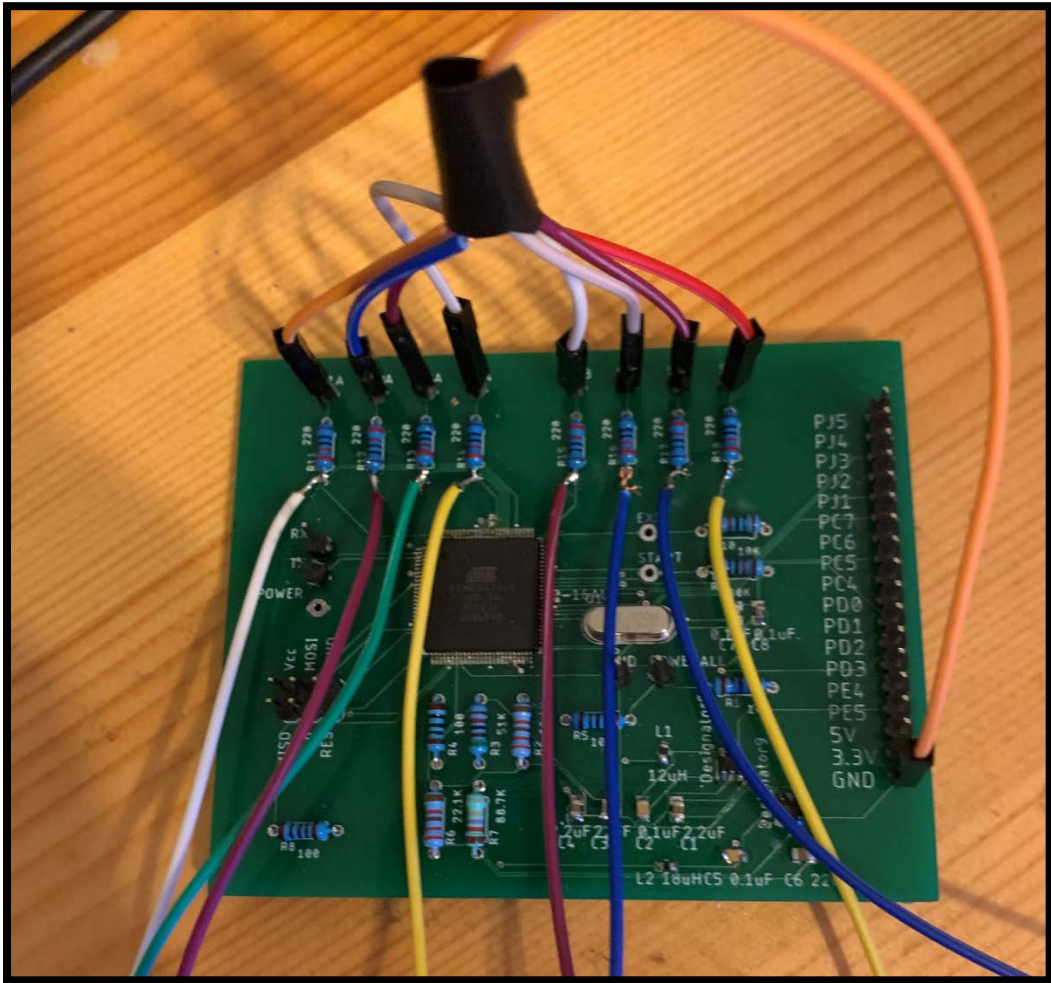


Figure 32: Physical Base System PCB

Bill of Materials: Base System PCB				
Quantity	Value/Type of Item	Item	Price (\$)	Manufacturer
2	2.2uF	Capacitor (SMD)	0.24	Samsung Electro-Mechanics
2	22uF	Capacitor (SMD)	0.82	Samsung Electro-Mechanics
2	100pF	Capacitor (SMD)	0.68	AVX
2		Capacitor (SMD)	1.28	Knowles Syfer
2	TPS560430	Voltage Converter	2.90	Texas Instruments
1	2x18	Pin Header	0.98	Adam Tech
1	2X18	Pin Header	0.12	Metz Connect
1	22uH	Inductor (SMD)	0.16	Taiyo Yuden
1	18uH	Inductor (SMD)	0.12	Taiyo Yuden
1	240K	Resistor (SMD)	0.42	Mouser
1	300K	Resistor (SMD)	0.004	Panasonic
1	100K	Resistor (SMD)	0.004	Vishay
1	56K	Resistor (SMD)	0.032	Panasonic
1	16 MHz	Crystal	0.48	IQD Frequency
1	ATMEGA2560-16AU	ATMEGA Processor	13.68	Microchip Technology

Table 17: Base System PCB Bill of Materials

11.3.2) UI Handheld

The schematic for the remote has a total of 10 main components. The logic comes from the ATmega2560 processor. The processor controls several components in the remote. The Xbee chip, LCD screen, four buttons, a switch, and several DC/DC converters will be connected to the MCU. The remote will be powered by a 6V battery which will be run through two DC/DC converters to step down the voltage power to a level accepted by the components. The components that need power are the ZigBee, the MCU, the LCD, and the buttons. Two DC/DC converters are needed as the power routed to the LCD display needs to be 5V, while 3.3V are needed to power the other components mentioned. The input for both converters will be the 6V from the battery pack. Once the switch is flipped, the power runs through both converters, turning on the MCU and the LCD. The buttons are connected to a logic high and connected to the MCU pins with pull down resistors, to keep the correct functionality for the buttons.

Various application components were necessary to use in conjunction with the ICs and buttons. One possible disadvantage with powering this system this way is the extra space that two DC/DC converters use up on the PCB. This could be alleviated by using a bigger PCB and remote if financially possible.

There are several logical connections that were routed and shown in the schematic. The components that have a logical connection to the MCU are the LCD, the Xbee, the Buttons. The buttons have the simplest logical connection. They are connected to the MCU using a pull-down resistor, meaning the buttons will register a logic high when the button is pushed. These are connected to ports 1.1-1.4 on the MCU. These ports were used because they are each capable of accessing a timer interrupt that when triggered can perform some function through the MCU.

The next logical connection in the remote is the LCD display. One important note about the LCD display, in the schematic a LCD (20x2) display is used. This is not the actual correct LCD that we will use. Our LCD has a size of 20x4. The 20x2 LCD was used as a placeholder for the schematic because the correct LCD's template and product information could not be found to add it to an eagle library. In order to maintain the schematic, the LCD placeholder has the same connections as the LCD we have, the only difference is the size of the screen. The LCD is not connected directly to the MCU, it has an I2C IC connected to it that simplifies the data transfer process. The I2C IC connects to the same pinout data ports that would be used for direct connect. There are four connections to be made with the I2C module, necessary for power, ground, SDA, and SDL. One SDA bus and one SDL bus are connected to ports 2.0 and 2.1 respectively. These are one of three available I2C ports on the MCU.

The last component with logical connections to the MCU is the ZigBee chip. The Xbee chip will be communicating with the MCU using an SPI connection. To set up an SPI connection, 4 connections will have to be made between the MCU and the ZigBee. A MOSI, MISO, SCLK and a CS. These connections will come from the ports 8.1-8.4. These ports are used because they are one of two SPI ports on the MCU. Accommodations will be made for the Xbee antenna in the remote.

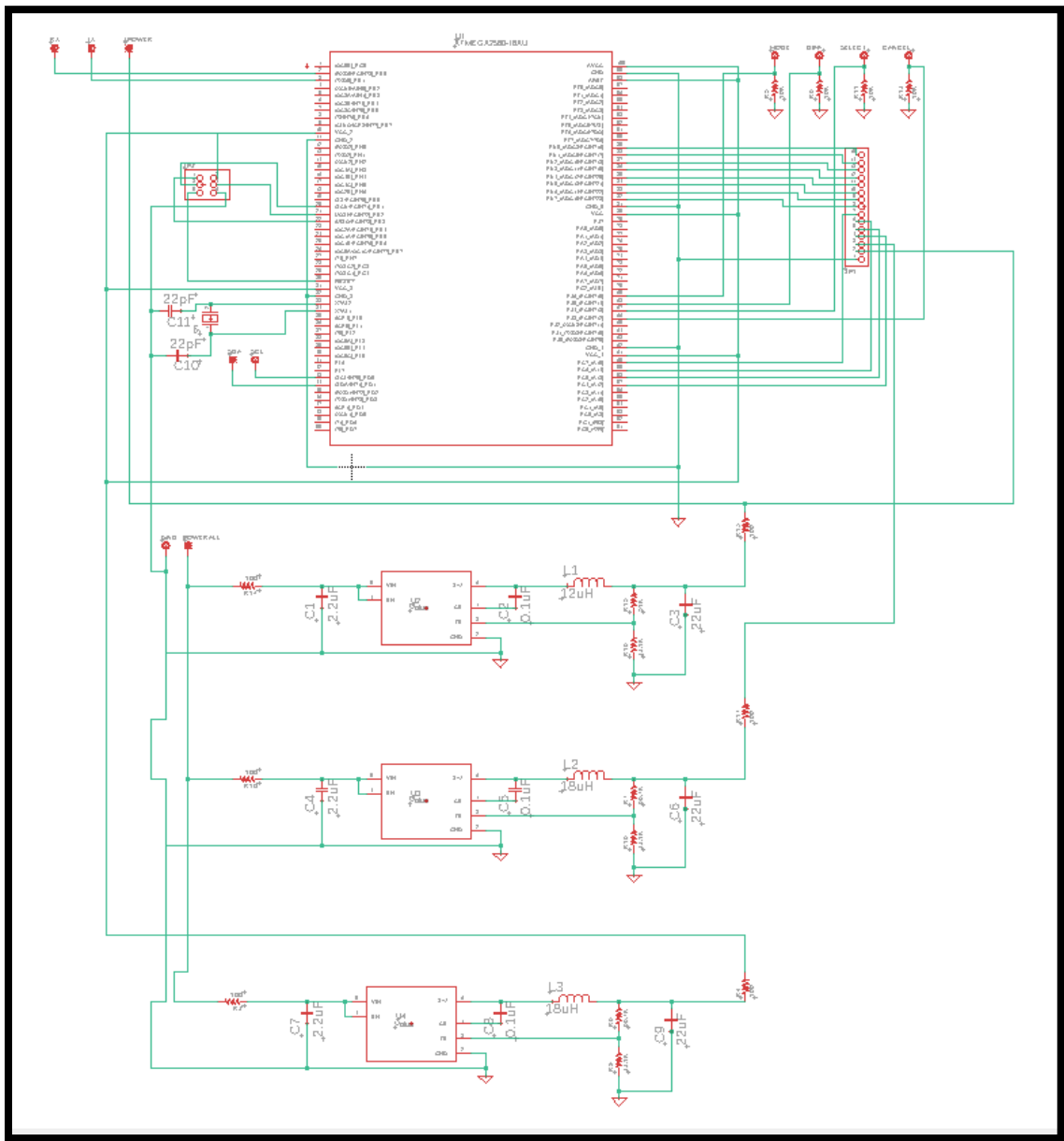


Figure 33: UI Handheld System Schematic

Because the remote is an all-in-one device, all of the components for the remote are located on the PCB of the remote. The central component in the PCB is the MCU, which is located in an area where all of the pins are able to be used with plenty of space. To the right of the capacitor the ZigBee chip is placed, allowing for close connections with the Y-capacitors that reduce data transfer noise for RF communication.

The LCD driver and screen are located at the bottom of the PCB. On the design the screen is located off of the board, this is uncertain and was done to accommodate the space constraint from the student version of EAGLE that was used to design the PCB. One option to allow for more flexibility is to replace the LCD screen on the PCB with a pin header that is then routed to the LCD. This would allow for far more freedom of movement for the LCD screen in the remote.

The driver is located just under the screen so the connections are efficient and stable. The four GPIO buttons are located along the left side of the PCB, with one button in the middle of the PCB. These buttons were placed in this configuration as a result of the limited space from the design (student version of EAGLE), final changes to the placement of these buttons are expected before the PCB is manufactured.

The last components included in the PCB design for the remote are the power conversion and generation components. A voltage source from a battery pack is attached to the PCB on the bottom right of the design. From this pack, the high voltage line is wired from to the switch that is placed just above the battery pack. The line then goes from the switch to both DC/DC converters positioned to the right of the pack. These converters take in the 6V from the battery pack and step the voltage down to 5V and 3.3V that can power the components on the PCB. One possible disadvantage of this design is the location of the battery pack. Currently it is located on the PCB which is inside the case of the remote. In order to change the batteries when they die, a slot in the case should be made above the battery pack. This slot will be essential for changing the batteries when dead. The .brd file and bill of materials for this PCB are shown below.

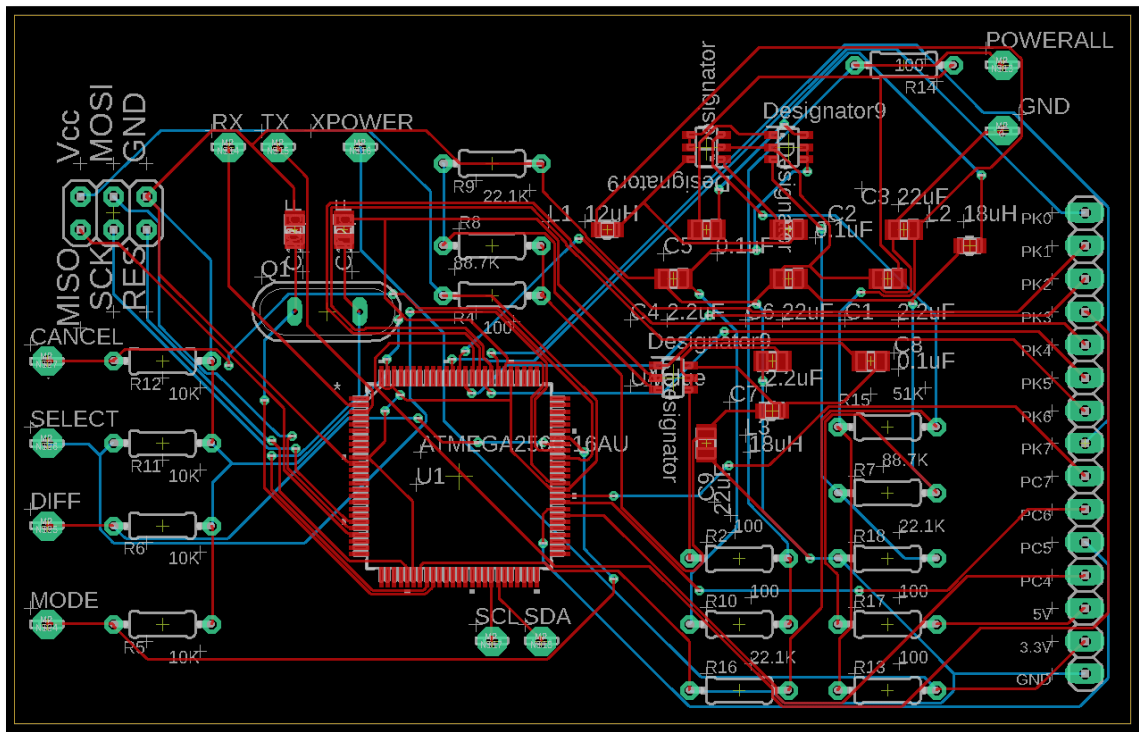


Figure 28: UI Handheld System Board Layout

Bill of Materials: UI Handheld System PCB				
Quantity	Value/Type	Item	Unit Price	Manufacturer
1	MCU	ATmega2560 chip	13.68	Microchip Technology
3	5 V	Buck Converter	1.45	Texas Instruments
3		SMD Capacitor	0.64	Knowles Syfer
3	2.2 uF	SMD Capacitor	0.12	Samsung Electro-Mechanics
3	22 uF	SMD Capacitor	0.41	Samsung Electro-Mechanics
1	15 pin	Pin header	0.81	Amphenol ICC (FCI)
1	dual row	Pin header	0.12	METZ CONNECT USA Inc.
2	22pF	SMD Capacitor		0.34
1	16 MHz	Oscillator	0.48	IQD Frequency Products
1	12 uH	SMD Inductor	0.16	Taiyo Yuden
2	18 uH	SMD Inductor	0.12	Abracon LLC

Table 18: UI Handheld System PCB Bill of Materials

11.4) Prototyping and Preliminary Testing

Before any PCBs are assembled, it is extremely important to make sure all of the circuits work properly in the prototyping phase. Prototyping circuits involves using a breadboard to ensure that both the ICs and the passive circuit components are arranged in a way that ensures smooth functioning. Prototyping should begin by testing the ICs to make sure that each chip works properly. This is essential to learn which chips are DOA, so they can be discarded. After the chips are verified, the next step is to verify that the circuits in the schematic perform the proper functions when tested on the breadboard. Final design for the PCB will be outlined during this prototyping phase as there is a large chance some circuits will have to be altered from the schematics.

One important check is for all of the I2C busses. Both the 5x5 grid of IMUs and the LCD drivers require I2C connection. The busses in the schematic will be tested and confirmed independently on the breadboard to make sure it is designed correctly. Another important component to test before manufacturing is the Xbee chip system. There are three parameters to test for the ZigBee pair. One is that the Xbee leader works properly; two is that the ZigBee follower works properly; three is that the pair can establish and maintain a connection over RF transmission. Only when each of these parameters is confirmed can the Xbee chips be mounted to the PCB. The homemade sensors and buttons should be tested to ensure proper functioning before mounting. Pull down resistors should be double check to make sure they are wired properly.

Power components should be tested to ensure the specified power is being delivered. Specifically the battery pack and converters should be checked for proper conversion and generation. This can be done just by measuring the output voltage of the converters on the breadboard using a multimeter. This same method should be used to measure the output voltage from the battery pack. The output voltage should be 6V. The converters should only be mounted once successful conversion is recorded in testing.

Components on both the remote and the bag will be tested before use. The main parameters to test for the remote are (1) the MCU, (2) LCD screen and I2C module, and (3) push buttons. The MCU will be tested in a similar fashion to all the other ICs mentioned previously. Each connection in the schematic will be observed to make sure proper functionality is kept for all used ports and connections. The LCD screen should be tested for functionality both with and without the I2C module. They each need to be tested independently to verify that both work and are not DOA, if they are not tested independently this won't be possible. Once both are confirmed to function properly, the paired functionality must be tested. Once the LCD is able to be programmed freely, it is ready to be connected to the PCB.

The last main parameter for the remote testing is to test the push down buttons. Each push button needs to record a logic high when pushed down, this functionality will be tested with the pull down resistors on the breadboard. Once this function is confirmed, the buttons can be mounted on the remote PCB.

11.5) PCB Assembly Planning

Due to the circumstances brought by the global pandemic we are currently living through, we do not have the typical facilities and environment containing soldering tools normally provided to us on campus. Our advisor, Dr. Samuel Richie, advised us to look into different methods for soldering to the PCB with real or makeshift soldering guns, and suggested sources to order our PCB to be made locally and online. Since this project requires us to use PCBs instead of breadboards, we will need to have a clear idea about how we will be creating the PCB for all the electrical components for this project. This section details possibilities for creating and soldering the PCB for this project.

11.5.1) Option 1: DIY Soldering Gun

One option would be creating a soldering gun and soldering the PCB ourselves for the project. This would be the cheapest way for us to solder if we get the board alone printed, since there are different ways to create your own soldering gun if no one between us were to have a soldering gun. One example is doing the soldering gun from a toaster, and another example would be creating this from a mechanical pencil. It is very easy to find these tutorials on YouTube and Google. However, this option will likely not be followed since most people in the group have enough soldering guns to share for the person who does not have the soldering gun.

11.5.2) Option 2: Purchasing a Soldering Gun

Another option would be buying the soldering gun and soldering the PCB ourselves for the project. This would help us to avoid the struggles and the stress of creating the soldering gun ourselves, and there are a variety of guns we could buy even from the local Walmart. The good thing is that the majority in this group already have soldering guns, soldering iron, and a multimeter, so we do not have to buy anything. This would help with the overall budget, time, and avoiding getting sick of coronavirus from trying to do this on campus. For the case of the sensors, we would be soldering the sensor ourselves, and we will be following the pinout of the sensor to properly solder it to a PCB board since the sensor does not come with pins. If we are a little short of time, we will be soldering the PCB ourselves. If there is plenty of time and money, we will ask any local companies if they could solder for us the PCB, or we will order the PCB online depending how much time we have. This is the option we will follow.

11.5.3) Option 3: Using a Local Company

A third option would be asking locally any companies to complete the assembly of the PCB for us. This option would be great since we would not have to wait a long time for them to do the soldering, but it will be considerably expensive compared to the other options. One option for a local company to complete PCB printing, assembly is EMS Technologies, as this company has a distance of 30 minutes from the university.

11.5.4) Option 4: Ordering the PCB Online and Shipping to Orlando

A fourth option would be ordering online, and they would be doing the soldering and shipping the PCB to us. We would choose between 4pcb.com and jlcpcb.com to make this happen. If we have more than 2 months before the submission and presentation and the PCB design is done and tested in a breadboard, we will follow this step since it will be the cheapest option between option three and four.

11.6) PCB Final Assembly and Testing

We ultimately decided to source our PCBs from JLCPCB due to their good reputation among our peers, as well as their speed and cost fitting our time and money budgets. This board comes unassembled, so assembly was considered separately from board purchasing. The method of assembly picked for the PCBs was self-assembly. This presented an opportunity for learning new skills as none of the group have soldered surface mount parts to a PCB before. Because the PCBs were self-assembled. Rigorous testing was needed to make sure all of the connections had proper function.

After the PCB is assembled, including all surface mount components and through hole components added. The first step to test the PCB is to make sure the MCU connections are working properly. This is important as any shorts between the MCU pins could cause serious problems for the MCU or even lead to a damaged MCU. To test the MCU pin connection, only a digital multimeter is required. Touch one lead to the top of the SMD MCU pin and follow the trace until the first exposed metal is shown, usually a resistor. Touch the other lead to this metal and observe the reading. The connection will be good if the reading shows a low (<1 ohm) resistance as this is the resistance of the trace. This procedure should be repeated until all of the pins on the MCU are checked and working properly. This procedure can also be used to check the converter IC connections. This test is an important step to do first because it takes the least amount of time to finish inspecting.

After all the connections are checked for the ICs on the PCB, the next step is to use the oscilloscope to measure the output voltage and current from the DC/DC converters. 5V and 3.3V output pins are added to the pin headers to make this very straightforward. It is important to check the voltage and the current as the proper power will only be transmitted if both of these parameters are correct. This is the section where our PCBs ran into the most trouble during testing. Due to a redesign in the middle of the second semester, the converters were found too late to be not supplying the power required. Unfortunately since this was found late in the semester there was not enough time to reorder a part that would meet the specifications better. This is why we ultimately powered our sensors and MCUs from the breadboard power supplies.

The last test to do is to make sure the MCU performs logic events correctly to each GPIO pin. To do this an oscilloscope is required to measure the waveform of the voltage at the pin. Program the MCU to send a repeating pulse and observe the output from the oscilloscope. Once the oscilloscope showed the correct output we could then put a LED on the pin to test if the pin is receiving power. Once all of these tests are done for each GPIO pin then we can head to the final phase of testing. The final phase of testing is to incorporate the full development code into the MCU. The sensor thresholds will have to be tuned differently to what they were during the initial testing of the sensors.

12.0) UI and Base Software Design

The software for the interactive workout bag is implemented using agile software development. This method emphasizes the use of adaptive design and development. The technologies integrated in our designs thus far are not set in stone, per say. During development, the technologies, and design choices may change, due to our team taking advantage of adapting our designs and development as new information is uncovered while the product is being built.

With this method of design, it allows us to implement specific components, test their performance, and make adjustments to each component along the way, it also allows our design to be flexible, and to take other design restrictions into account.

12.1) Design Method

The software for the interactive workout bag is implemented using agile software development. This method emphasizes the use of adaptive design and development. The technologies integrated in our designs thus far are not set in stone, per say. During development, the technologies, and design choices may change, due to our team taking advantage of adapting our designs and development as new information is uncovered while the product is being built.

With this method of design, it allows us to implement specific components, test their performance, and make adjustments along the way, it also allows our design to be flexible, and to take other design restrictions into account.

12.2) Design Environment / Tools

The software design environment depends greatly on the processor that is used. Because our design implements the use of Atmel microcontrollers, the Arduino IDE will be used as the IDE to write, push, and debug the program onto the microcontrollers. This choice is beneficial for the team because everyone has experience with programming microcontrollers to perform various tasks, which include the low power modes, interruptions, display drivers, and more. Arduino uses embedded C as the program language and supports AVR architecture, which all of our team members have experience in as well. Arduino also already comes with the header files and interrupt vectors for the Atmel chips, and multiple useful libraries used for complex math and communications, which makes it easier to use as well.

12.3) General Overview

The software for the workout bag will be split into two different areas: UI system and Base system. The UI system is a separate device from the base and handles the interface between the user and the system. The base system is the electronic components tasked with reading sensors, providing light sequence patterns, and communicating with the UI system (also called the remote).

In general, the user turns on the device with the UI system. In the UI system, the user selects the mode and difficulty on the remote. The UI system then transmits the users' selections to the base. Once the base receives the mode and difficulty, the workout bag's targets blink to indicate to the user that they can press the start button. The base system then executes the desired mode and difficulty by enabling various timers, sensors and target area LEDs. The base system will then complete the sequence, and compute the results related to the mode, and send them back to the UI system for the user to see. The UI system then returns to the beginning, allowing the user to select another mode and difficulty or to shut down.

12.4) UI System

The first action to happen, is when the user turns on the system by using the on/off switch. The system then turns on the screen, and displays its welcome message, and prompts the user for their selection. There are two branches based on the user selection, system off or mode. If the OFF button is pushed, the system calls an interrupt to terminate the program and shut down. If the Mode button is pushed, the user is able to select a mode. Once the Select button is pressed, the user then can choose a difficulty setting, based on the mode. The user then presses select a second time. The mode type and difficulty type are sent to the base system MCU and the UI system enters a low power mode, while the rest of the sequence happens at the base. In order for the user to be in place when the session begins, the workout bag's target areas will blink to indicate to the user that they can begin the session by pressing the start button on the base.

12.4.1) Low Power Modes

The UI system will not be used the majority of the time, since the user's main interaction is with the workout bag. Since the UI system is battery powered, the low power modes on the MCU will be used while the user is completing a session. Once the UI system transmits the necessary data to the base system, the UI system will enter a low power mode to conserve power.

As the UI system receives data, the MCU is interrupted, and begins processing the display sequence for the data that is given based on the mode selected.

12.5) LCD Display

12.5.1) Welcome Screen

The LCD display will have several different types of messages printed to the screen. The first screen displayed is the Welcome Screen. On the first row, the display shows "<name of product>". On the third row, the display shows "Press Mode". Because the buttons do not have individual LEDs, the display will show the mode and difficulty selection on the screen, as the user presses the buttons to cycle through the options.

12.5.2) User Input

The user inputs their workout selection by using the buttons under the display. Each button is labeled with their function. The functions for the buttons are Mode, Difficulty, Select and Cancel. The buttons used for the user inputs are susceptible to the bouncing effect, where the user may press the button only once, but the MCU receives several phantom pushes. To correct this, a de-bouncing method is used, that incorporates a small delay that disables the button to avoid the processor receiving multiple button pushes.

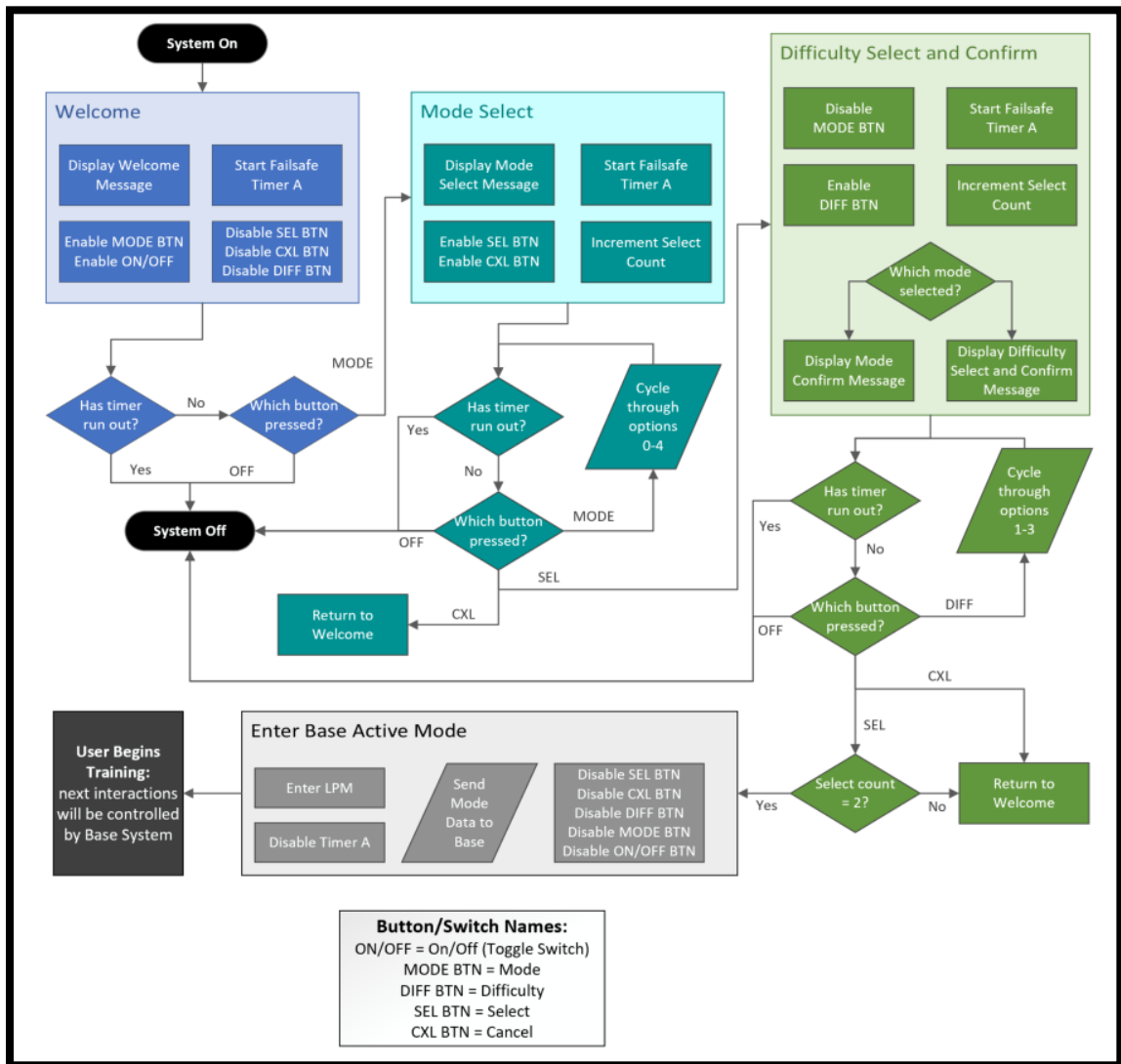


Figure 35: Software flow chart for UI and Initialization

12.5.2.1) MODE (MODE BTN)

The user is to press the MODE button to cycle through the available modes (5). On the display in the first row, the first six columns display “MODE: “. Columns 7 through 20 display the mode options, where each press of the mode button, changes the mode option that is displayed. Internally, the first time the mode button is pressed, the mode register is set to 0. Each time the button is pressed, it increments the register, until it holds number 4 (last mode), and then returns to 0 and keeps going. To select a mode and continue, the user presses the select button. The mode option and the register values are shown in Table 17, below.

Reg Val	Mode Option
0	Combination Generator Mode
1	Reaction Time Mode
2	Accuracy Mode
3	Cardio Mode
4	Ambient Mode

Table 19: Mode options and register values

12.5.2.2) SELECT (SEL BTN)

When the user has the mode they wish to use on the screen, the select button is used to store this mode, and move on to the next function. When the select button is pushed 1 time, the mode button is disabled and the difficulty button is enabled based on the mode type. Then on the display, the second row reads “DIFF: “. The second time the select button is pressed, the mode and difficulty buttons are disabled and the UI system sends the mode and difficulty setting to the base.

12.5.2.3) DIFFICULTY (DIFF BTN)

Based on the mode selection, different options of difficulty will be available. The options in total are: Easy, Medium and Hard. Modes 0,1,2,and 3 are able to have different difficulty settings (Easy, Med, Hard), whereas mode 4 does not have a difficulty setting. Only in modes 0, 1, 2 and 3 can cycle through the modes, meaning the difficulty button is enabled for those modes only.

When mode 0 is selected, the difficulty setting is adjusting the amount of time the allotted time to hit a target area, where the allotted time gets shorter as the difficulty setting is increased. When mode 1 is selected, the delay between targets is randomized with higher variances. When mode 2 is selected, the difficulty setting is adjusting the threshold distance for accurate hits. As the difficulty setting increases, the threshold distance reduces, effectively making it harder to complete accurate hits. When mode 3 is selected, the difficulty setting is controlling how long the session lasts. As the difficulty setting increases, the length of the session increases.

When mode is set to 4, the difficulty setting is not applicable. In mode 4, the workout bag is in the ambient mode and is not used for training or measuring the user’s ability.

Reg Val	Difficulty Option
0	Easy
1	Medium
2	Hard
3	X

Table 20: Difficulty options and register values

12.5.2.4) CANCEL (CXL BTN)

The cancel button is used to start over the selections. When pressed, it resets the mode, difficulty, and select registers, and returns to the welcome screen. This can be pressed at any time until the user presses the select button for the second time. To cancel a workout after the select button has been pressed twice, the user will need to press the exit early button on the base.

The cancel button is also used to return to the welcome screen after results have been displayed to the user. When pressed after the results are displayed, the function remains the same, which returns to the welcome screen to make selections.

12.5.3) UI Failsafe Timer 0

The UI system is highly dependent on the user performing an action (making selections and pressing buttons), which leaves the UI system to be potentially stuck in an infinite loop of waiting for input response. In the event that the user walks away from the UI system, the remote requires a way to shut off to avoid remaining on and wasting battery. To do this, a timer is used that resets every time an active button is pressed. If the timer is able to reach its end, an ISR is called to shut down the UI system.

12.5.4) Results Transmission

After the base system has transmitted the user's results to the UI system, the UI system is interrupted to exit low power mode. The display is turned on, and the result categories and their values are displayed to the user. The results are displayed until the user presses cancel or until the failsafe timer goes off. The display then returns to the welcome screen, allowing the user to make another selection, or shuts off, depending on if the user presses mode, uses the OFF button or the failsafe timer goes off.

Because the LCD display has 4 lines that can hold up to 20 characters, the results category names and values must be able to be displayed within those limits. This means if the category abbreviations are within 5 characters and the results values are 3 characters (0-999), then a maximum of 6 results can be displayed (two columns of 3 rows). This is because the first row of the display is needed to display the users' selections, in the manner of: MODE / DIFFICULTY.

12.6) Communication with Base

The base and the UI system will need to communicate in order to send the user inputs to the base and send the results back to the user. To do this, the MCU in the base and the UI system will require a transceiver. The transceiver is connected to the MCU with SPI (4 wire) connections, and uses the IEEE standard 802.15.4 for creating the wireless personal area network (WPAN). To integrate this with the MCU chips, a CC2XXX transceiver chip can be used to support the frame handling, buffering, and transmission. The chip contains the zstack software developers kit to configure the communications between the two devices on the WPAN. This works with IEEE Standard 802.15.4 as shown in Figure 18.

Our team has not had the opportunity to work with ZigBee to create wireless communications between two devices. Therefore, the program's design is not solely dependent on the use of wireless communication. In the event our team is unable to successfully transmit the required data between the two MCU's, a wired connection is able to be used instead (SPI, I2C, UART). The remote's pouch is on the side of the bag, reducing the probability of the user tripping on any exposed wires.

12.7) Base System

When the base system receives power, the MCU initializes its devices (communication, LEDs and Sensors) and then enters low power mode. Once the base system receives the user's mode selection and difficulty selection from the UI system, the base system will enable its LEDs to cycle through a color sequence. This is to indicate to the user that the base system is ready for the user to press the Start Button. Once the user presses Start, the base MCU will then load the premade sequences based on the mode selected. The base system cannot enter low power modes here, as the processor is actively turning on/off target indicators and capturing sensor data from the target areas that are light up. Once the main sequence timer ends, the processor can begin to compile the data into the results which are provided to the user, and then sends the results to the UI system and returns to low power mode.

12.7.1) Target Areas (LEDs)

The workout bag has five different modes: Combination Generator mode, Reaction Time mode, Accuracy mode, Cardio mode and Ambient mode. Each mode will have unique target areas that the base MCU will need to enable to indicate to the user where the hits should take place. On Side A, the four target areas will incorporate binary sensors that detect whether a certain zone on the bag was hit. On Side B, four target areas will also incorporate the binary sensors but will be configured as concentric rings to not only determine if a hit is registered, but also determine the distance of the hit from a center location marked on the bag.

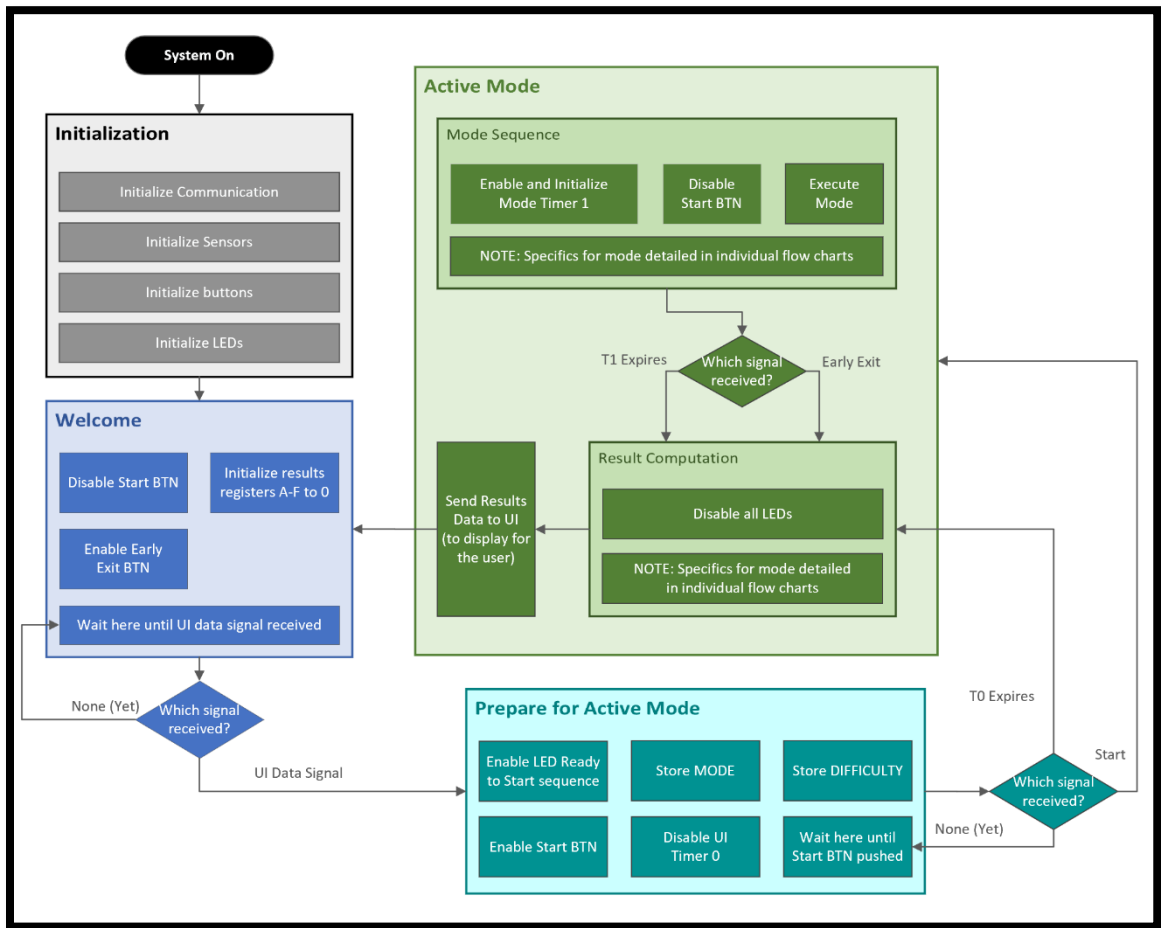


Figure 36: Software flow chart for Base Active Mode Initialization

12.7.1.1) Combination Generator Mode

The Combination Generator mode is on side A of the workout bag. The addressable LED strips outline four target areas that light up to indicate where the user should hit. Each target area is identified with a numerical value, 0 through 3. In this mode, the MCU will begin its first timer, which limits the session to a specific number of seconds.

Next, the MCU will generate a random number between 0 and 3, and turn on the corresponding target area. Simultaneously, the MCU will start its second timer that controls how long the target area is on. When the second timer ends, an interrupt is called, and the program returns to the random number generator, and the second timer is started. The length of the second timer is variable, as the user is able to select different difficulty settings, where the harder settings result in shorter secondary timers, meaning the user has less time to hit the target area.

The user shouldn't have to wait for the secondary timer to end if they have successfully hit the target area. Therefore, if the user hits the target area before the secondary timer ends, the MCU calls an interrupt to return to the random number generator to continue with the session. Lastly, once the first timer ends another interruption is called, and it ends the session by turning off all LEDs and sensors.

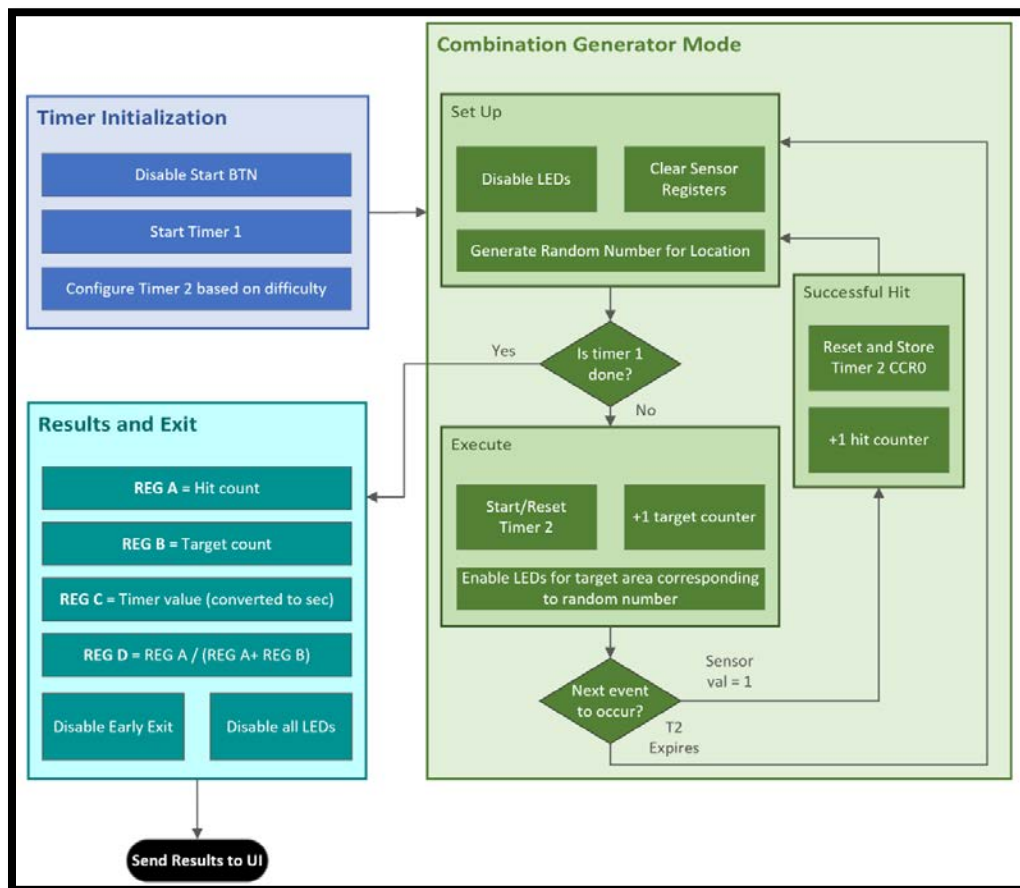


Figure 37: Software flow chart for the Combination Generator Mode on Side A

12.7.1.2) Reaction Time Mode

The Reaction Time mode is on side B of the workout bag. For the reaction mode, the MCU starts its first timer, which controls how long the session lasts. This time is set by the developers, and not adjustable. Next, a second timer begins as the target area is enabled. Once the sensor detects a hit, the second timer ends and the target area is turned off. The MCU waits in a delay loop for a short period of time and the process repeats. This delay is random and the range is adjustable, with the difficulty setting. The time from when the target area is enabled (second timer starts) and when the second timer ends is the amount of time the user reacts. Each hit's reaction time is stored, and averaged for the final result, until the first timer ends.

Since this mode is calculating reaction time, the program is dependent on the user's hit to move forward with the sequence. However, the entire sequence is based on the first timer. Therefore, when the first timer expires, the program is able to move on to the results computation, without getting stuck in an infinite loop.

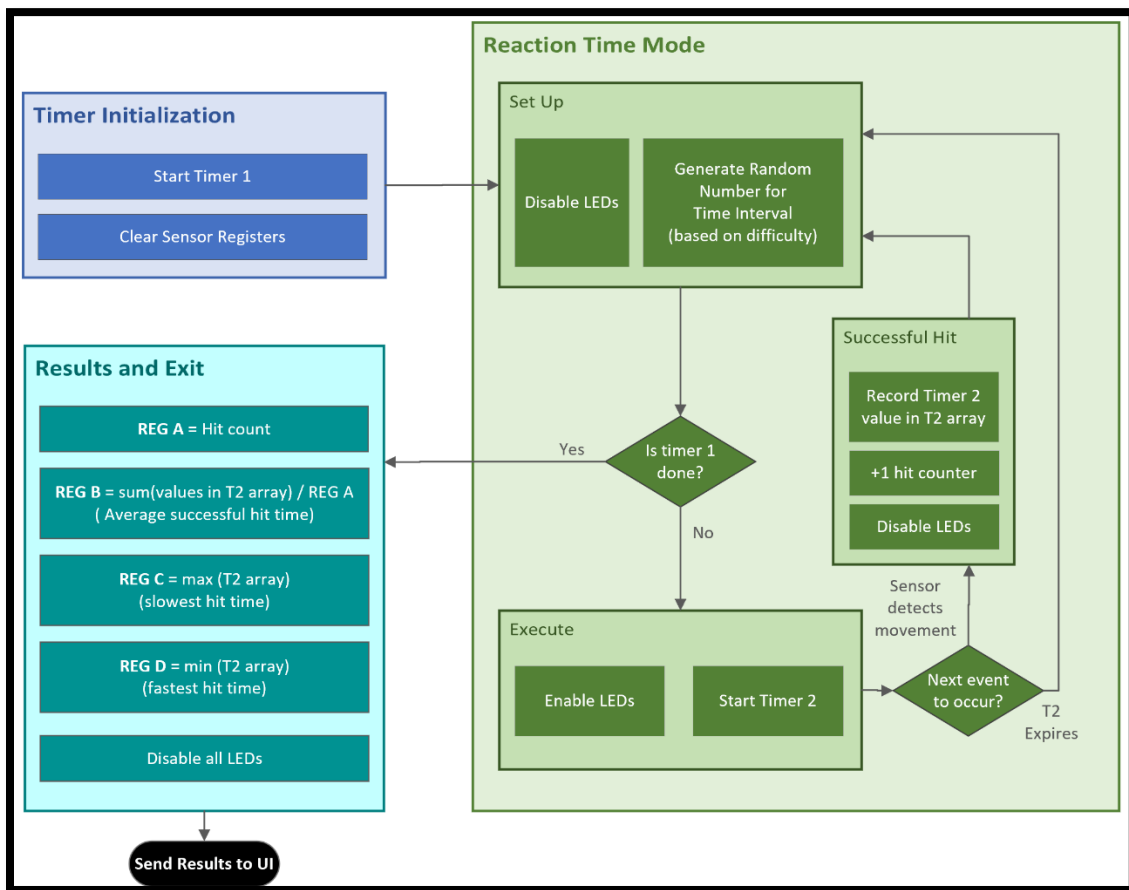


Figure 38: Software flow chart for the Reaction Time Mode on Side B

12.7.1.3) Accuracy Mode

The Accuracy mode is on side B of the workout bag. Similar to the reaction time mode, the accuracy mode has one target area that lights up. The MCU starts its first timer, which controls the length of the session. Next, the second timer starts when the target area is lit. Different to the reaction time mode, the sensors are used to locate the hit's distance from the center of the target area, which is marked on the outside of the workout bag. Once a hit is detected, the process repeats itself in the same way as the reaction time mode. The difficulty setting changes the threshold distance for an accurate hit, instead of the delay, as in the Reaction Time mode. When the first timer reaches its limit, an ISR is called to exit the session and display the results.

The MCU will have predetermined thresholds of distance to measure the level of accuracy of each hit. The user can choose the difficulty setting, whereas the difficulty settings get harder, the threshold distance becomes smaller. Hits with a distance greater than the threshold are considered a "hit", while hits with a distance smaller than the threshold are considered an "accurate hit".

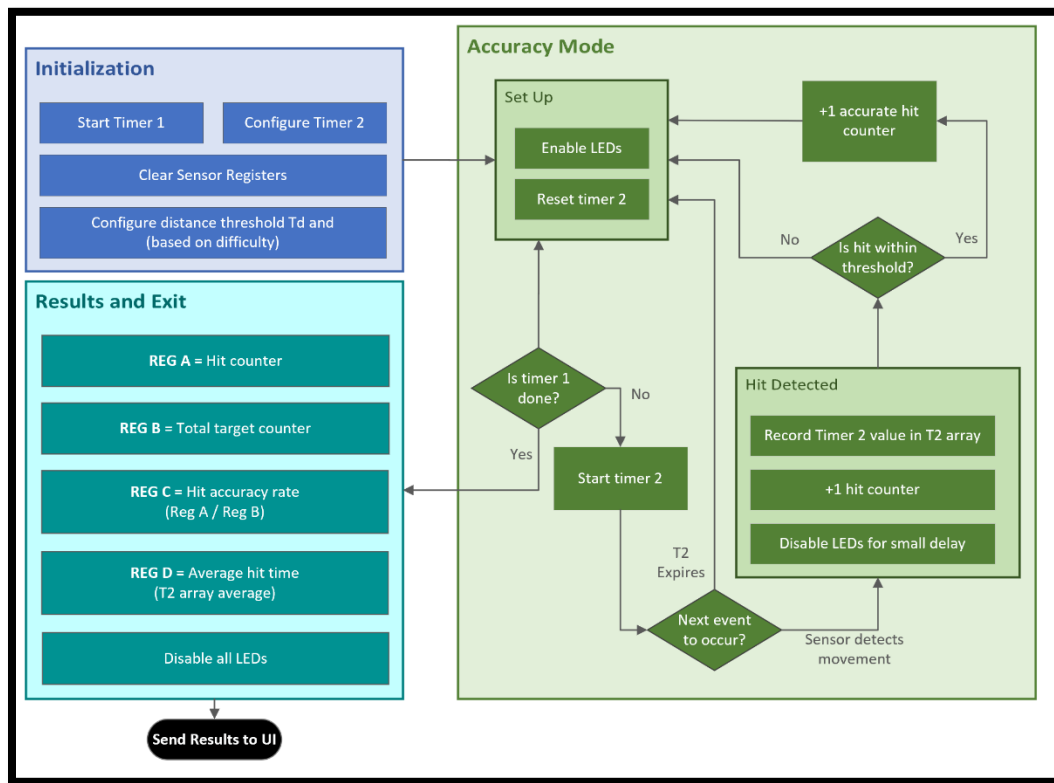


Figure 39: Software flow chart for the Accuracy Mode on Side B

12.7.1.4) Cardio Mode

The cardio mode is also on side B of the workout bag. In the cardio mode, one target area is used and remains lit for the entire session. The goal is to have as many hits as possible during the allotted time. The MCU will start its timer and the target area will be lit. Because the cardio mode is only counting the number of hits during the session, only one timer is used. Each time the user hits the target area successfully, a register holding the number of hits increments by one. Once the timer ends, the session is over, and the program moves on to calculating and displaying results.

The cardio mode is not dependent on receiving the user's hits to move forward, therefore, if the user decides to not hit the workout bag during the time frame, the score will be affected at the end. In the cardio mode, the length of the session is variable with the difficulty setting. Therefore, to make the workout more challenging, the length of the session increases for each difficulty setting.

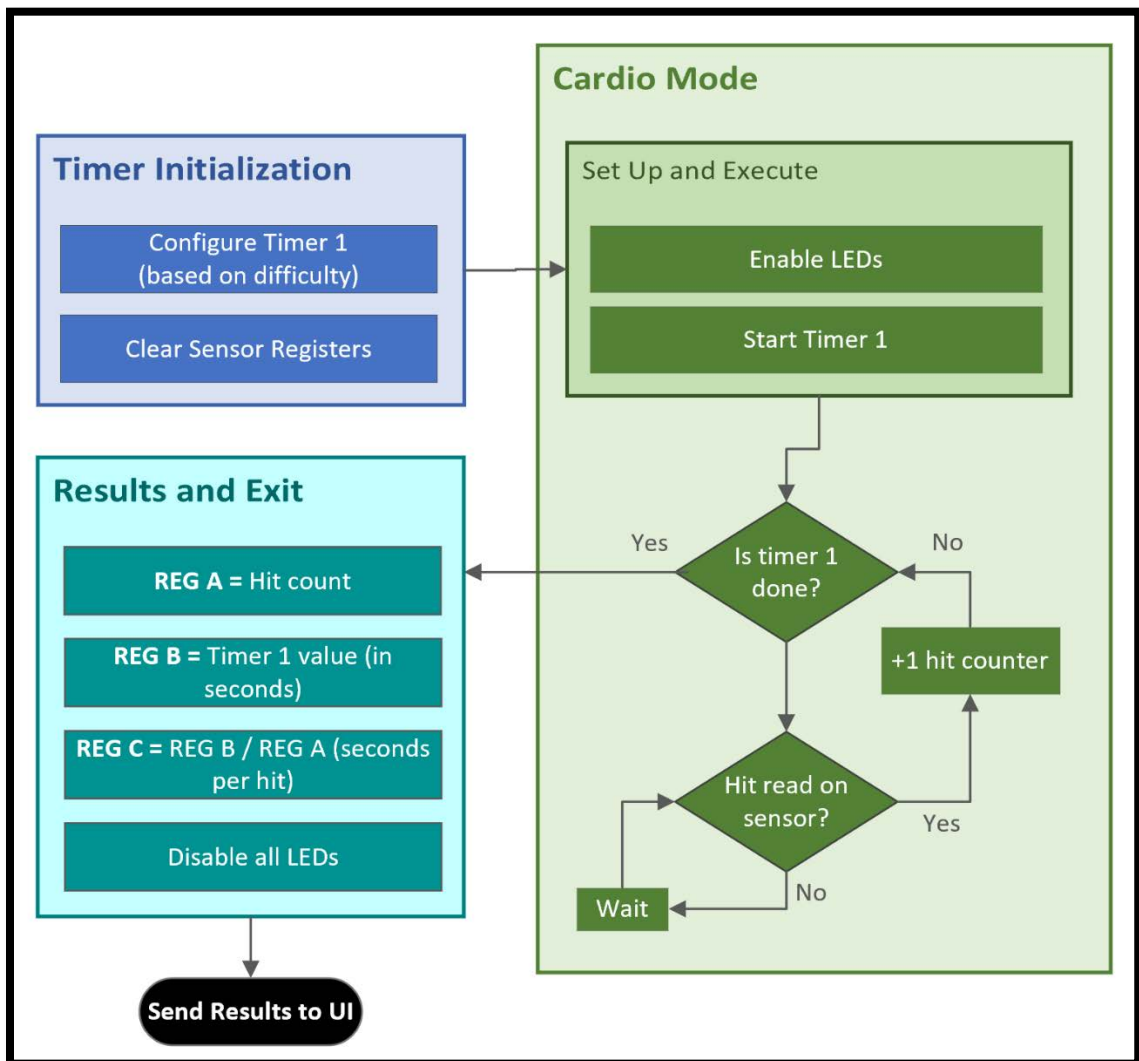


Figure 40: Software flow chart for the Cardio Mode on Side B

12.7.1.5) Ambient Mode

The last mode that the workout bag includes is the Ambient mode. This mode uses side A and B. In the ambient mode, all the LEDs are enabled, and cycled through the display colors. Because this mode does not require any user interaction, the mode will require an automatic way to shut off after a given amount of time. This prevents the workout from being in an infinite loop with all the LEDs on. This requires the use of one timer, that repeats itself for a given number of times, to allow the Ambient mode to remain on for a decent length of time before shutting off.

The ambient mode can also be disabled by using the exit early button to return to the welcome screen/mode selection.

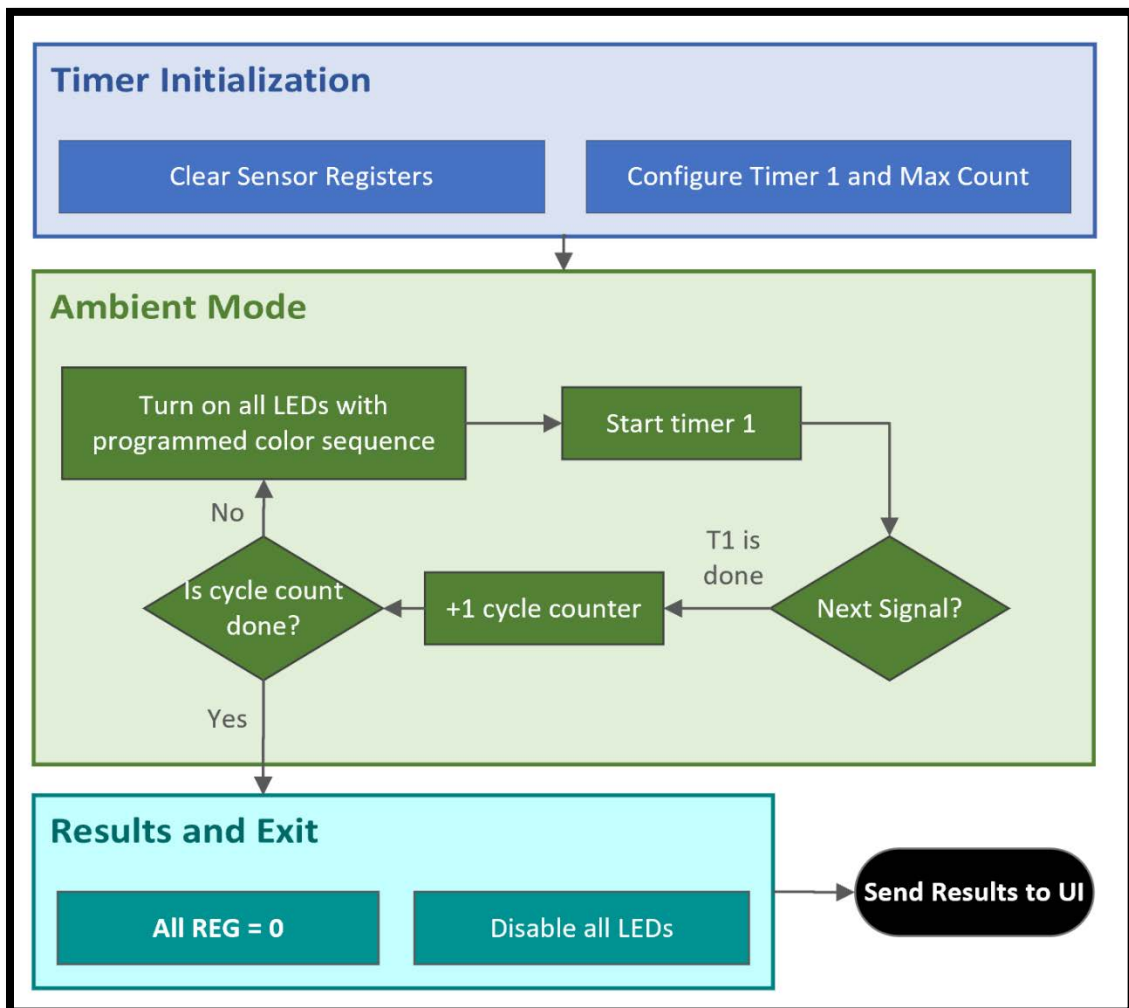


Figure 41: Software flow chart for the Ambient Mode using both sides A and B

12.7.2) Results Computation

The workout bag will have several different methods of computing results based on the mode of the session. In general, the MCU will collect the relevant data according to the mode in easily accessible memory. Once a session is complete, the MCU will perform basic arithmetic to find the desired output values. These are the values that the base system transmits to the UI system.

12.7.2.1) *Combination Generator Mode*

In this mode, the user is tasked with hitting different target areas in a specific amount of time. The overall goal of this mode is to measure the number of successful hits within the given time allotment. Therefore, the results sent to the UI system include the total time of the session, the number of successful target hits, the number of total target areas, and the average time for a hit.

The total time of the session comes directly from the first timer. Because the first timer can be interrupted with the exit button, the program uses the value of the timers compare register to calculate the time by knowing the clock type's frequency. This is done by dividing the total number of cycles (this also needs to be calculated, as the total number of cycles can be higher than the maximum value a register can hold) by the clock's frequency. For example, if the total number of cycles is 32,768 (0 to 32,767) and the clock frequency is 32kHz, then the elapsed time is 1 second.

The total number of successful hits is calculated in real time. As a sensor detects a hit, the MCU increments the register that holds this value. Similarly, the total number of targets is also calculated in real time. Because this mode is timer based, the total number of targets depends on how fast the user is able to hit them. Therefore, this value is also calculated in real time, by incrementing another register every time the target area is updated, meaning when the secondary timer's interrupt is called.

Lastly, the average time for a hit is calculated in two stages. First, each time a hit is registered, the secondary timer value is added into a specific register. This creates a sum of the total amount of time of every hit. In the second stage, the total time of every hit is divided by the number of successful hits.

12.7.2.2) *Reaction Time Mode*

In this mode, the user is training their reaction speed, therefore the results that are most useful are the average time for a hit, the number of hits, the longest time for a hit, and the shortest time for hit. These are the four results that will be sent to the UI system.

Similar to the combination generator, the average time for a hit is calculated the same exact way. In the event the user discontinues the session without pressing the exit early button, the time it takes for the system to cycle through will not be counted, since the sensors did not detect a hit, allowing the results to not be affected. The total number of hits are also calculated the same way as the combination generator's successful hits.

The longest time for a hit and shortest time for a hit are found by storing each hit's time in an array. At the end of the session, the maximum and minimum value of the array are found, and stored in separate registers. The array holding all the times is then cleared to avoid keeping unnecessary data. If the sensors do not detect a hit, meaning the user has allowed the secondary timer to expire and the exit ISR is called, the time is discarded so that it will not be stored into the array, allowing the results to be unaffected.

12.7.2.3) Accuracy Mode

In the accuracy mode, the user is training their ability to produce accurate hits to a given target. The results that are most useful are: most accurate hit (best) distance, most accurate hit (best) reaction time, least accurate hit (worst) distance, least accurate hit (worst) reaction time, accuracy rate, and hit rate. These results are sent to the UI system.

The best and worst hit distances are found by storing each hit distance into a distance array. At the end of the session, the maximum and minimum value of the array are found, and stored into separate registers. The best and worst hit times are found the same way. Similar to the reaction mode, when the second timer's ISR is called, it means the bag did not receive a hit, and the time is not stored. This keeps the index of both distance and time arrays to be consistent, where the index relates the time of the hit to the distance of the hit from the center.

The accuracy rate is found by using the distance array, where each index stores that hits distance. Each hits' distance is then compared to the threshold value, which is set depending on the difficulty setting. If the hits' distance is less than the threshold distance, the hit is considered accurate and a register that holds the number of accurate hits is incremented by one. At the end of the array, the total number of accurate hits is divided by the total number of hits (length of distance array) in order to give the accuracy rate as a decimal ratio).

The hit rate is found by dividing the total time of the session by the total number of hits. This allows the user to identify their progress of their accuracy rate with the speed at which they are training.

12.7.2.4) Cardio Mode

In the cardio mode, the user is performing as many hits as possible in a given amount of time. The most useful results for this mode are total number of hits, total time, average hit time and hit rate.

The total number of hits is found in real time, by incrementing a register each time a sensor detects a hit in the target area. The total time is directly from the timer. Because the timer can be interrupted with the exit button, the program uses the value of the timers compare register to calculate the time by knowing the clock type's frequency. This is done the exact same way as in the combination mode generator.

The average time per hit is calculated by dividing the total time value by the total number of hits. This allows the user to see the average time it takes them to complete a hit. Lastly, the hit rate is found by dividing the total number of hits by the total time value, or inverting the average time per hit value. This allows the user to see the average amount of hits per second they are able to complete.

13.0) Function Analysis and Development

The software in the UI and base system will depend on the creation and implementation of several functions and interrupt service routines that interact with each other. From a high-level viewpoint, the program will remain in a loop, waiting for instructions. The instructions are provided from the user input, which can either be a push of a button or when a sensor detects a user's hit. Within the functions that respond to the user input, several other functions will be called indirectly, depending on specific status registers which indicate which state/mode the program is in. For example, within the button interrupt service routines, multiple functions need to be indirectly called to display information on the display, and to communicate with the base. In addition, several more functions are required to initialize both the base and UI system before receiving any user inputs, displaying any information or any communication, as well as functions to handle moving the system into a low power mode which also depends on the state of the program. The next several sections review the information related to the design and development of the needed functions.

13.1) LCD Configuration

The type of LCD used will be the HD44780 that has 4 display rows. Each line is able to display up to 20 characters. Before data can be displayed to the screen, the LCD chip must be configured. The LCD is able to be configured using the register select pin (Rs). When Rs is low (logic 0), the LCD chip interprets the data lines (D0 - D7) as control commands, whereas when Rs is high (logic 1), the LCD chip interprets the data lines as display data. In order to simplify our design, a function will be called that sets Rs to 0, and then transmits a series of control commands to initialize the LCD before any display data is transmitted. Unlike a 1602 display, the display addressing must be configured before each display data is sent. This function will be a return type of void and the argument will be an unsigned integer. By changing the argument integer, the function will transmit a series of preset commands to initialize the LCD in the correct way based on what is desired. For example, to display the welcome screen message, the cursor location and display data address must be configured before any character generator addresses are sent. By using the data sheet of the HD44780 display, the desired control commands can be created. The next several sections summarize the types of commands that will be needed.

13.1.1) Clear Display

One of the most used types of commands will be to clear the display and to set the display data address (DDRAM) to 0 in the address counter, which is the first cell of the first line. Since clearing the display is a control command, the register select (Rs) will be set to 0. When the register select is 0, the read/write register (R/W) is set to low (logic 0). The only bit set to logic 1 is D0. This command essentially sends a space character (code 20H) to the character generator RAM (CGRAM) address/DDRAM address, and then sets the DDRAM address counter (AC) to 0.

The clear display command will be used each time the LCD display is initialized, either when it first powers on or when returning to the welcome screen. The return to the welcome screen function is called when the cancel button is pressed. The clear display command is also sent after waiting a specific amount of time after the select button is pushed for the second time. This allows the user to return the remote to the holder, and press the Start button on the base, before clearing. The clear command will also be sent when the failsafe timer ISR is called, to clear the display.

13.1.2) Return Home

In order to return the cursor to the home position (DDRAM AC set to 0), the return home command can be used. This command leaves the contents of the CGRAM/DDRAM unchanged, meaning the display will not clear. This command also returns the display shift to the original position. To execute this command, the register select is still set to logic 0, and D1 is set to logic 1. D0 is not read for this command, so its value can be logic 0 or 1.

13.1.3) Entry Mode

The entry mode is used to configure the address counter increment/decrement as well as the setting the display shift. I/D (increment/decrement) is set to 1 to increment AC as the DDRAM is written to. Therefore, once a character code is written to DDRAM, the DDRAM AC is incremented so that the next symbol is in the next address over. It is important to note that at the end of a row, the next address does not go to the following row. Instead, it continues on the same line. Line 1 is split across rows 1 and 3. Similarly, S (display shift) will shift the entire display when the DDRAM is written to in the opposite direction that is set by I/D. The effect of the display shift makes it look as if the cursor is standing still. For our purposes, the display shift will always be 0 and the increment will always be to the right (logic 1), meaning the display shift is disabled and text will go from left to right.

13.1.4) Display Control

Display control is used to turn the display on or off, turn the cursor on or off and turn the cursor blink on or off, where on is logic 1 and off is logic 0. For our purposes, the cursor and cursor blinking will always be set to 0 (off), as the UI system is only used to display the user selections and results. The display will be set to on during initialization and set to off when the failsafe timers ISR is called. Turning the display off does not change the contents in the CGRAM/DDRAM. If the power is removed from the LCD, the next time the LCD is turned on, the chip automatically is configured to its original settings. The display control command will need to be sent to configure the settings desired for our projects use.

13.1.5) Cursor / Display Shift

The cursor or display shift allows the display to shift the cursor position (left or right) or the display (left or right). By setting S/C to 1, the display is enabled to shift and setting S/C to 0 enables the cursor to shift. The left or right shifting is controlled by setting R/L. R/L set to 1 shifts right, and R/L set to 0 shifts left. Since our program does not require a display shift for any messages, the S/C bit will always be set to 0.

13.1.6) Function Set

The function set controls several functions of the device. First is the data length (DL), which is how the display chip knows how to read the data lines. For our purposes, we will always use 8-bit data (1 byte). To do this, DL is set to 1. If set to 0, the display chip reads data 4 bits at a time. Next, N controls the number of display lines, where if set to 0 it is 1 line and set to 1 it is 2 lines. Although the user sees four rows on the display, there are really two lines (rows 1 and 3 are considered 1 line). One line can hold up to 40 characters (40 addresses), which would be using rows 1 and 3 for line 1 and rows 2 and 4 for line 2. The table below shows the DDRAM addressing of each row and column, where the addresses are in hexadecimal. Lastly, F is controlling the font size of the characters. Because we will be implementing two lines, F must be 5x8 dot count, which is logic 0.

13.1.7) Set DDRAM Address

To display characters in specific position/row on the display, the DDRAM address can be specified, by setting the ADD bits to the correct hex address listed in the table below. For our purpose, we will always be using a 2-line display, where 1 line is 40 addresses long. Rows 1 and 3 are line 1 and rows 2 and 4 are line 2. This function will be used repeatedly, as several of the screen functions have data displayed in specific areas. For example, the welcome screen is centered across row 1. Therefore, to begin displaying text, the starting address would need to be set using this command.

The UI device displays several strings of characters on the four rows of the display. The set DDRAM address command is used repeatedly to set the cursor to a specific position and row to begin displaying data. This command only needs to be used when the incremented DDRAM address is not the desired position for the next characters sent.

Position	1	2	3	4	5	6	7	8	9	10
Row 1	00	01	02	03	04	05	06	07	08	09
Row 2	40	41	42	43	44	45	46	47	48	49
Row 3	14	15	16	17	18	19	1A	1B	1C	1D
Row 4	54	55	56	57	58	59	5A	5B	5C	5D

Table 21a: Address indices corresponding to row and position of the 20x4 LCD

Position	11	12	13	14	15	16	17	17	19	20
Row 1	0A	0B	0C	0D	0E	0F	10	11	12	13
Row 2	4A	4B	4C	4D	4E	4F	50	51	52	53
Row 3	1E	1F	20	21	22	23	24	25	26	27
Row 4	5E	5F	60	61	62	63	64	65	66	67

Table 21b: Address indices corresponding to row and position of the 20x4 LCD

13.1.8) Write Data to Display

To write data to the display, Rs is set to 1. Because of this, D0 – D7 are read into the CGRAM/DDRAM. Therefore, to send a character to the screen, D0 – D7 are configured to the specific character generator codes. For example, to send the character “H”, the following bits are configured, which are shown in the table below. When this happens, assuming the DDRAM address is set at 0x00, the display will print the letter “H” at Row 1 Position 1, and then increment the DDRAM address to 0x01 (assuming in entry mode I/D is set to 1). Therefore, the next time that DDRAM is written to, it displays the character the next position in row 1, which is at address 0x01.

Rs	R/W	D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	1	0	0	1	0	0	0

Table 22: CGRAM codes to display ‘H’ on one cell of the LCD screen

The list of CGRAM addresses and their corresponding character are found in the HD44780 datasheet and is also shown below. Each character’s data byte (D0-D7) will need to be defined in the UI system’s code. D7-D4 are the upper bits, and D3-D0 are the lower bits, therefore the hexadecimal code for the character “H” is 0x48. This command is used repeatedly throughout the LCD message functions, as it is used to print one character to the display at a time. Therefore, to create a string in a row, the Write Data to Display command is used for each character.

Lower 4 Bits	Upper 4 Bits	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
xxxx0000	CG RAM (1)	█	0	@	P	`	P	E	&		°	À	Á	Â	Ã	Ä	Å
xxxx0001	(2)	4	!	1	A	Q	a	q	Ä	Ï	±	À	Ñ	á	ñ	ä	å
xxxx0010	(3)	“	”	2	B	R	b	r	W	Γ	φ	²	À	Ó	ã	ö	
xxxx0011	(4)	”	#	3	C	S	s	s	3	π	ε	³	À	Ó	ã	ö	
xxxx0100	(5)	±	\$	4	D	T	d	t	H	Σ	×	¼	À	Ó	ã	ö	
xxxx0101	(6)	±	%	5	E	U	e	u	Ï	σ	¥	¼	À	Ó	ã	ö	
xxxx0110	(7)	█	&	6	F	V	f	v	J	J	ı	ı	ı	ı	ı	ı	ı
xxxx0111	(8)	ı	'	7	G	W	w	w	ı	ı	ı	ı	ı	ı	ı	ı	ı
xxxx1000	(1)	↑	<	8	H	X	h	x	Y	#	ƒ	ω	É	Ê	Ë	€	€
xxxx1001	(2)	↓	>	9	I	Y	i	y	U	Θ	¹	É	Ò	é	ù		
xxxx1010	(3)	→	*	:	J	Z	j	z	4	Ω	∞	∞	É	Ó	é	ú	
xxxx1011	(4)	←	+	:	K	L	k	l	ω	∞	∞	É	Ó	é	ú		
xxxx1100	(5)	≤	,	<	L	\	ı	ı	ı	ı	ı	ı	ı	ı	ı	ı	ı
xxxx1101	(6)	≥	-	=	M	ı	m	>	b	#	Œ	ı	ı	ı	ı	ı	ı
xxxx1110	(7)	▲	.	>	N	^	n	~	W	ε	ı	ı	ı	ı	ı	ı	ı
xxxx1111	(8)	▼	/	?	O	_	o	ó	ó	ı	ı	ı	ı	ı	ı	ı	ı

Figure 42: CGRAM address table

The complete function table discussed in this section can be seen below.

Rs	R/W	D7	D6	D5	D4	D3	D2	D1	D0	Function
0	0	0	0	0	0	0	0	0	1	Clear Display
0	0	0	0	0	0	0	0	1	—	Return Home
0	0	0	0	0	0	0	1	I/D	S	Entry Mode
0	0	0	0	0	0	1	D	C	B	Display Control
0	0	0	0	0	1	S/C	R/L	—	—	C/D Shift
0	0	0	0	1	DL	N	F	—	—	Function Set
0	0	1	ADD	ADD	ADD	ADD	ADD	ADD	ADD	Set DDRAM ad
1	0	Upper bits				Lower bits				Write to CG/DD RAM

Table 23: LCD Basic function table

13.2) LCD Message Functions

There are several premade messages that the MCU will send to the LCD display when a specific event occurs. The list of messages are: Welcome Screen, Mode Select, Difficulty Select, Start, and various results messages, depending on the mode. The message functions are called by the various interrupt service routines from either button interrupt flags, timer interrupt flags or receive data interrupt flags.

The table below summarizes the functions needed for the LCD display. This list is not a finalized list, as more functions may be required to make the system perform better. Each category will be discussed in more detail throughout this section.

Function	Commands
Initialize LCD	Clear, Function Set, Entry Mode, Display Control
Welcome	Clear, Set DDRAM, Write to DDRAM
Mode Select	Clear, Set DDRAM, Write to DDRAM
Difficulty Select	Clear, Set DDRAM, Write to DDRAM
Start Screen	Clear, Set DDRAM, Write to DDRAM
Results	Clear, Set DDRAM, Write to DDRAM

Table 24: LCD Message function summary table

13.2.1) Pre-Mode Execution Messages

Before any modes are selected, the LCD must display messages to the user to indicate what type of input is requested. Depending at what stage the UI system is at, different messages and instructions are presented to the user. These functions are critical, as the user would have no way of knowing which buttons to press to navigate through the UI system. In general, each of these messages are displayed at either initialized or when a button is pushed. Because they are dependent on user input, the fail safe timer is used to return back to the welcome screen, or power down.

13.2.1.1) Power On and Initialization

When the device is powered on, the LCD must be initialized before the screen can display any data or text. One of the first functions called in the main will be to initialize the LCD, by sending the following commands. Once this step is complete, the LCD will continue to increment the DDRAM address as data is written to the DDRAM. This step also configures the display to have 2 lines (4 rows), with a font of 5x8 dots (required for 2-line display).

1. Clear Display
2. Function Set:
 - a. DL = 1 (8-bit interface)
 - b. N = 1 (2-Line Display)
 - c. F = 0 (5x8 Dot Font)
3. Display Control
 - a. Display ON
 - b. Cursor OFF
 - c. Blinking OFF
4. Entry Mode
 - a. I/D = 1 (Increment)
 - b. S = 0 (No Display Shift)

13.2.1.2) Welcome Message

The welcome message function is called after the LCD initialization function. The welcome function will consist of several commands and data writes. The welcome message consists of "WELCOME" centered in the first row and "PRESS MODE!" in the third row, also centered. To do this, the next LCD function to be called will first set the DDRAM address to 0x06, which is where the first letter (W) will print. For the next six characters, the DDRAM address will increment for each write. Next, the DDRAM address is updated to 0x18, where "P" will be printed. For the next 10 characters, the DDRAM address will be incremented for each write. An example is shown below.

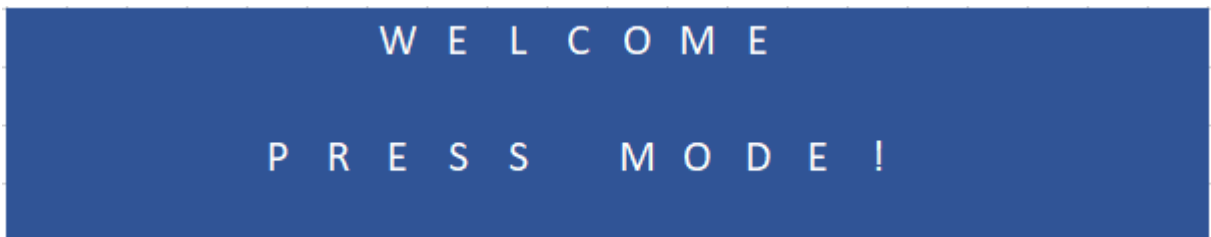


Figure 43: Welcome screen display message sample

13.2.1.3) Mode Select

Once the mode button is pushed, and the correct ISR completes, the next LCD message function to be sent is the mode select. In the mode select, the first command sent is to clear the display, and the DDRAM is set to 0x00 automatically by using the clear display command. Next, each letter of "MODE:" is displayed in row 1. The first mode displayed is the combination generator mode, which is abbreviated to "COMBO GEN", where the first character is a space at address 0x06. As the Mode button is pushed to cycle through the options, the DDRAM address is set to 0x06 and prints each type of mode corresponding to the mode select register that is updated with the button's ISR. Each mode selection option is shown below, where the space is counted as a character that is sent.

Mode Select Register Value	Mode Chosen (SIDE)	Mode Choice Display
0	Combination Generator (A)	MODE : COMBO GEN
1	Reaction Time (B)	MODE : REACT TIME
2	Cardio (B)	MODE : CARDIO
3	Accuracy (B)	MODE : ACCURACY
4	Ambient (BOTH)	MODE : AMBIENT

Table 25: Mode register values with corresponding modes

13.2.1.4) Difficulty Select

After the select button is pushed once, the DDRAM address is updated to 0x40, which is the address of the first position of row 2. The letters "DIFF : " are printed in order, since the DDRAM address increments after every write. When the mode select register is 0-3, the first difficulty selection displayed is "EASY", where the first character (space) is at address 0x46. As the difficulty select button is pushed to cycle through the options, the DDRAM address is set to 0x46 and prints the three types of difficulty that correspond to the difficulty select register which is updated with the button's ISR. When the mode select register is 4, the LCD will fill the rest of the 2nd row with spaces, starting at 0x40, as the difficulty select button will be disabled. Each option for row 2 is shown below.

If Mode Select Register Value = 0-3		
Difficulty Select Register Value	Difficulty Chosen	Difficulty Choice Display
1	Easy	DIFF : EASY
2	Medium	DIFF : MED
3	Hard	DIFF : HARD
If Mode Select Register Value = 4		
N/A; no difficulty choice for ambient mode		

Table 26: Difficulty register values with corresponding difficulties

13.2.1.5) Start

Once the select button has been pushed twice within the process of choosing mode and difficulty, the UI system is sending the mode and difficulty selections to the base. The UI system waits for all data to be transmitted before displaying the next screen. For the next screen, the DDRAM address is updated to 0x58 and displays the words "PRESS START" in the center of the row 4. After a timer's ISR is called, the clear display command is sent, and the display control command is used to turn the display off so the UI systems MCU can enter a low power mode as it waits for results to be sent back.

Since the clear display command is not used during the mode and difficulty selection screens, the user's chosen options remain on rows 1 and 2. An example of the display is shown below:

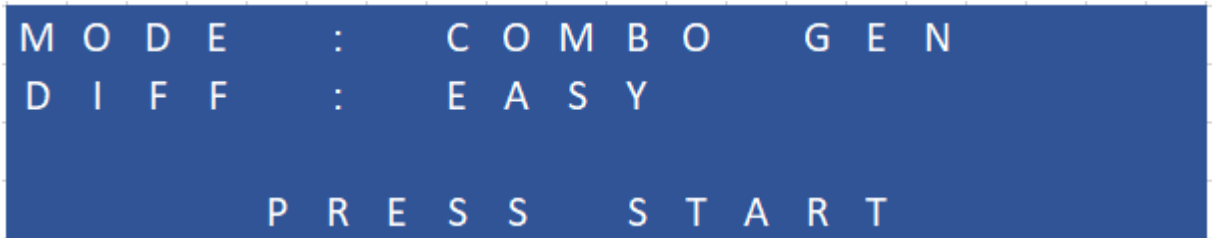


Figure 44: Start screen display message sample

13.2.2) Results Display Message Functions

As the MCU wakes up when it receives the results from the base, the LCD initialize function is called first. Next, based on the mode select and difficulty select registers, specific result messages are displayed using row 1. Because the DDRAM address is already at 0x00, the next function sends words "MODE:XX DIFF:YYY/Y", where XX indicates the which mode is used (combination generator = CG; Reaction Time = RT; Accuracy = AC; Cardio = CD) and YYY/Y indicates which difficulty is used (EASY, MED, HARD) when mode select register is equal to 0-3. When the mode select register is set to 4 (ambient mode), the clear display and welcome message are called immediately, since there are no results. The next rows will hold the results, where the message displayed depends on the mode selected.

Each modes' results are examined below. In general, each mode's results use a combination of setting the DDRAM address and writing to the DDRAM. In addition, the results numerical value must be parsed into single digits, left to right. This allows each digit to be defined as the character generator code that is sent when using the write to DDRAM command.

13.2.2.1) Ambient Mode

In the event that the user presses the cancel button or the failsafe timer's ISR is called, the clear display command is sent, and then the welcome screen message function is called again to repeat the process. Note, that if the mode was Ambient, the clear display command and welcome message function are called immediately, as there are no results to display.

13.2.2.2) Combination Generator Mode

The results that the UI system receives when in the combination generator mode are the number of hits, number of targets and the hit ratio (hits/targets). The DDRAM address is set to 0x40 (position 1 of row 2). Next, the MCU sends data to write to the screen "HITS:##", where ## is the character that represents the numerical value of the number of hits. The DDRAM address is then set to 0x14 (position 1 of row 3) and sends the data "TARGETS:##", where # is the character that represents the numerical value of the number of total targets. Next, the DDRAM address is set to 0x54 (position 1 of row 4) and prints "HIT RATIO:X.XX", where X.XX is a character representation of a float. Each digit of the integer / float value will need to be extracted to send the correct character code. An example of the results screen is shown below.



Figure 45: Combination Generator results display message sample

13.2.2.3) Reaction Time Mode

In the reaction time mode, the results received are the number of hits, the average time per hit (in seconds), the longest hit time (in seconds) and the shortest hit time (in seconds). The DDRAM address is set to 0x40, and the data sent is "HITS:## AVG TIME:X.X". Next the DDRAM address is updated to 0x14. Note that this must happen, as row 3 position 1's address is not next once the entire row 2 is used. The data sent is now "LONGEST TIME: X.X". The DDRAM is then set to 0x54, and the data sent is "SHORTEST TIME: X.X". Each numerical value must be parsed to extract each digit which corresponds to the character code. An example is shown below.

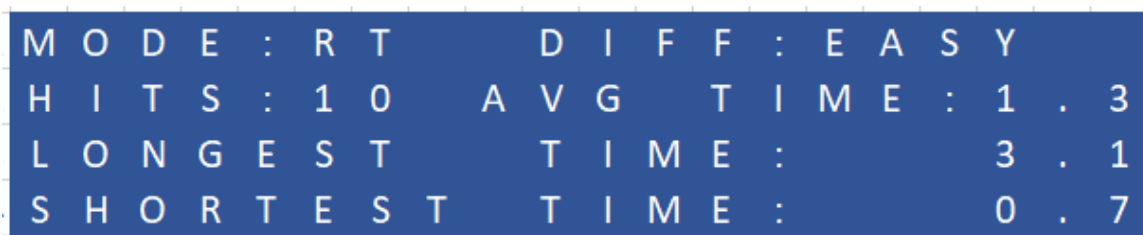


Figure 46: Reaction Time Mode results display message sample

13.2.2.4) Accuracy Mode

In the accuracy mode, the received results include most/least accurate distance, most/least accurate time, accuracy rate, and hits per second. Because there are six total results and a limited number of characters on the display, the categories are abbreviated, where MAD/MAT is the most accurate distance and time, respectively. LAD/LAT is the least accurate distance and time. ACR is the accuracy rate, and HPS is the hits per second. First the DDRAM address is set to 0x40 and the following data is displayed: "MAD:X.XX MAT:X.XX". Next, the DDRAM address is set to 0x14 and the data sent to be displayed is "LAD:X.XX LAT:X.XX". Next, the DDRAM address is set to 0x54, and the data sent to be displayed is "ACR:X.XX HPS:X.XX". Each place holder (X) is the character representation of each digit of the desired result. An example is shown below.

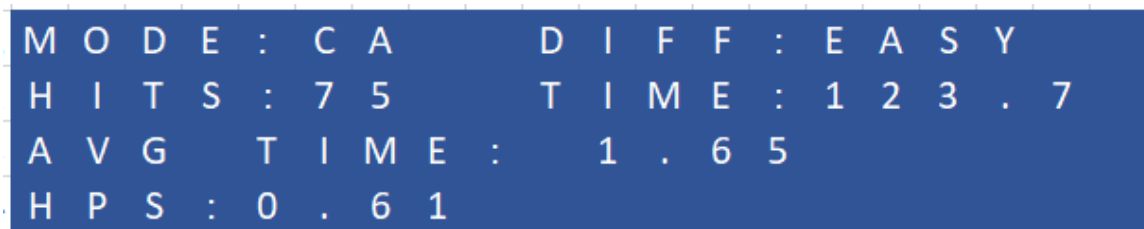


```
M O D E : A C          D I F F : E A S Y
H I T S : X X          T A R G : X X
A C C .   R A T E : X . X X
A V G   T I M E : X . X X
```

Figure 47: Accuracy Mode results display message sample

13.2.2.5) Cardio Mode

In the cardio mode, the received results include number of hits, total time, average time per hit and hits per second. First the second row's data is sent by first setting DDRAM to 0x40 and then using the write command and sending the following sequence "HITS:XXX TIME: XXX.X". Because the values can be from 1 digit to 3 digits, each character code will depend on the length of the data. Next, the DDRAM is updated to 0x14, and the write command sends "AVG TIME:XX.XX". Lastly, the DDRAM is updated again to 0x54 and the data sent is "HPS:X.XX".



```
M O D E : C A          D I F F : E A S Y
H I T S : 7 5          T I M E : 1 2 3 . 7
A V G   T I M E :     1 . 6 5
H P S : 0 . 6 1
```

Figure 48: Cardio Mode results display message sample

13.3) LCD Communication

The desired method for the MCU to communicate with the LCD display is to implement the I2C protocol. By using I2C, the MCU only requires the use of two pins, the serial data line and the serial clock line (SDA and SCL). To begin, a function is called to initialize the I2C pins. First, the two pins must be diverted to use the I2C functionality by configuring the pin select lines. Next, the two control registers for I2C must be configured to allow the MCU to act as the master, set the baud rate, select the clock frequency, select the clock divider, set the slave device address, enable the transmit buffer interrupts, and more.

13.3.1) I2C

Within the eUSCI_Bx Control Word Register 0, specific bits are enabled to change the functionality of the eUSCI. First the control register must have its least significant bit equal to 1 (UCSWRST). This allows the other bits of the control register to be modified. Once done, the UCSWRST must be set back to 0.

Next, the following masks are used to enable certain bits of the word control register. The mask UCMST added with UCMODE_2 added with UCSSEL_2 added with UCTR sets the MCU as the master device that transmits with I2C functionality and selects its clock as the submaster clock. When the master device is ready to send the start signal, the word control register is masked with the start signal (UCTXSTT). When the master is ready to stop sending data, the control register is masked with the stop signal (UCTXSTP).

Next, another function is created to transmit data from the master to the slave. The master sends the start sequence, UCTXSTT and sends the slave address with the least significant bit set to 0. This tells the display that the master is writing to the data lines. Upon acknowledged, the master begins to transmit the desired data: either control commands or data write commands. Before the MCU places more data into the transmit buffer, it must wait for the transmit buffer interrupt flag to go down. This means that the transmit buffer is emptied after the data in the buffer has been sent out. Once the master is finished sending data, it transmits a stop signal, UCTXSTP, and the function returns. This transmit function must be called for every sequence of data that is requested to be displayed.

13.3.2) GPIO Pin Connection

In the event that the I2C protocol is unable to be successfully be used to display data on the LCD, 6 additional GPIO pins can be connected directly to the LCDs pins to control Rs, R/W and D0 – D7. However, by controlling these registers directly, a delay must be incorporated for the LCD chip to read these registers. The HD44780 datasheet lists the minimum required delay for each type of command. Each command requires at least 37s except for the return home command, which requires 1.52ms. A simple delay loop function can be called each time a delay is needed.

The GPIO pins used to control Rs, R/W and D0 – D7 must be configured as output pins, by masking them with their corresponding bit ($P1DIR |= BIT0$). Next, to set a bit to high, the P1OUT control register is masked with the bit ($P1OUT |= BIT0$). To set the bit to low, the control register is masked with the inverse of the bit ($P1OUT \&= \sim BIT0$).

13.4) Zigbee Configuration

The base system and UI system communicate using the IEEE 802.15.4 protocol. Both MCUs (in the base and the UI system) will be connected to a Zigbee device (XBee chip). The two XBee chips communicate over a wireless personal area network (WPAN). The WPAN must be configured in each XBee before they can be used in the system.

Within the network, there are three roles of an XBee chip: a router, an end device, and a coordinator. Each network must have one and only one coordinator. The coordinator XBee can communicate with all end devices within the network (acting as a router) as well as establishing and maintaining the network (allowing specific devices to join). The end device XBee can only communicate with a router (or coordinator, as the coordinator has the functions of a router). Because our design only incorporates two devices that communicate, it does not matter which device is the coordinator.

An XBee device is technically composed of two devices, the antenna and the explorer. The explorer is connected to the MCU using serial data lines (UART), and then the explorer is placed on top of the antenna (aligning the pins). When the MCU sends data through the antenna, it uses the transmission (TXD) pin which is connected to the IN pin on the explorer. The receive data, the MCU reads the received pin (RXD), which is connected to the OUT pin on the explorer. Lastly, the explorer is powered using the 3.3V pin from the MCU. The ground pin is connected to one of the ground pins of the MCU.

In order to configure the two XBee devices, a program called XCTU. XCTU is a free program that allows developers to interact with general digital RF modules (like Xbee) in a simple, easy to use, graphical interface. XCTU also helps test the communication between devices within the network. The XBee explorer comes with a micro USB port, to let the developer connect the XBee directly to a computer to configure the WPAN.

13.4.1) Coordinator & End Devices Configuration

To set up the configuration of each XBee, the explorer must be attached to the antenna. On the computer, open the XCTU program and plug the explorer into the USB port. In XCTU, search the USB port by clicking "Discover Devices" and selecting which port to search. Once the device shows up, select "Add Device". Then open the configuration settings to configure the role of the XBee. First select a Channel ID and PAN ID. These ID's must be the same for both XBee devices. Next, select a destination high and a destination low. This allows the coordinator to know which device (if more than one end device) that the data should be routed to. 64-bit, 16-bit and strings are acceptable as inputs.

To enable the first device as a coordinator, follow the instructions above until the last step is reached. In this step, enable the device as a coordinator. Once finished, the settings are written to the XBee. It is best practice to physically label which XBee is the coordinator.

To enable the second device as an end device, plug the second XBee device into the USB port and follow the same instructions above, making sure that the channel ID and PAN ID are matching. Instead of enabling this device as the coordinator, the device is enabled as an end device. The settings are then written to the XBee chip.

To test the communication between the devices, plug both XBee's into two different USB ports. Once both devices are added, the console can be opened. When selected on each device, data can be typed into the console. To make sure they are communicating, both devices should echo the same data in both the consoles.

13.5) MCU and XBee Communication

Because the MCU and the XBee explorer use UART to communicate, the MCU will need to divert two pins that support the eUSCI communication. Therefore, a function for diverting those pins and configuring the two word control registers will need to be called before any communication can happen between the explorer and the MCU. In addition, the receive and transmit interrupt vectors will also need to be configured, which allows the MCU to transmit/receive the data stored in the transmit/receive buffers.

13.5.1) UI System

Because the XBee explorer is directly connected to the TXD and RXD pins on the MCU, the transmit and receive interrupts will be called when their respective flags are raised. Therefore, there is not a direct function called when data is ready to be sent from the UI system to the base system. Instead, once the select button has been pushed twice, the correct button ISR is called which stores the mode select and difficulty select into the TXD buffers. This action calls the UART ISR that transmits the mode and difficulty selection to the XBee.

Similarly, when data is being received from the XBee chip, meaning the base has sent results to the UI, the MCU exits low power mode from the ISR. The receive buffer is emptied into a register on the MCU, and the flag is lowered. The results are then stored one by one in separate registers, as data is received from the XBee. These are the registers that LCD write function uses to send to the DDRAM data registers for the LCD.

13.5.2) Base System

For the base system, the process is nearly identical. When the UI system has sent data through the XBee network, the receive interrupt is called when the RXD interrupt flag is raised. The data is moved over to an internal register until all data is received. Based on the data received, the base system performs the specific mode/difficulty settings and computes results. Once the results are ready, the data is moved over, one by one, into the TXD buffer, which sends the data to the XBee, which sends the information UI system.

It is important to note that the several status registers must be polled before transmitting data. First, within the UART function, the MCU must check and wait until the transmit lines are available, meaning there is not a current interrupt pending/happening. Next, the MCU must also check and make sure the XBee is available and not currently receiving or transmitting data. If both of those conditions are met, then the system may send data to the buffers.

13.6) Button Configuration

The buttons on the UI device and at the bottom of the base are used as the user inputs. When each button is pressed, an ISR is called to handle the specific action that must be completed. The button ISRs are using the majority of the created functions to perform the necessary actions, such as updating a specific register, writing data to the LCD, or initiating serial communication to XBee. Depending on which pins are used as button inputs, the corresponding port vector will be used to handle the ISR. Several pins share a port vector, therefore each time the ISR is called, the program must check which pin had raised an interrupt flag.

As stated before, the pins assigned to the buttons will need to be configured as inputs, by masking the pin with the specific bit. For example, to have P1.0 be used as an input, the direction must be set to logic 0 ($P1DIR \&= \sim \text{BIT0}$). This will have to be done for each pin that is acting as an input to the system (in the UI and the base). Next, the pins will have to also be configured for how they are used as inputs, for example, as a pull down or pull up. For our case, each button is a pull up, meaning when the button is pushed, the circuit is pulled up to high, closing the circuit. The resistor must also be enabled for this pin to work. Next, the interrupt must be enabled, so that when the flag is raised, the port's ISR is called.

Lastly, the buttons must incorporate a debouncing method. When the button is pushed, the MCU may interpret several pushes, due to the instability at the edges of the push. Therefore, the buttons interrupt must be disabled for a short duration, so that it is not called during these phantom pushes. The duration should be made to be around 20ms. To implement this, a basic for loop can be used to wait out the duration.

Instead of the program polling to check if the button has been pressed each clock cycle (by checking the interrupt flag), the program will use interrupt service routines (ISRs) to perform specific actions. This allows the MCU to sit idle while the user is not pressing any buttons and keeps the pipeline free of instructions. Another benefit is that the code size can be reduced slightly, as each button would require a function to poll the interrupt flags.

13.7) Button ISRs

The buttons on the UI handheld device and on the base system of the bag will each require individual interrupt service routines, or ISRs, for their unique functions. The ISR for each button is detailed in the following sections.

13.7.1) Mode Select Button ISR

When the UI system first turns on, the first button that is enabled in the main is the Mode Select Button. When this button is pushed, the mode select register is incremented, until it reaches the last mode, and then loops back to the beginning value. The mode select register must be set at mode 4 within the main; when the mode button is pushed and the ISR is called, the next increment will force the mode select register to go to 0, so that the first mode is displayed (combination generator). This is also the register that is sent to XBee (and to the base) to determine which mode sequence is selected.

Within the ISR, the failsafe timer is reset, and the clear display function is called, to clear the welcome message. This also sets the cursor to position 1 or the first row. Then based on the mode select register, the write to DDRAM function is called a specific amount of times to print the first mode. The ISR exits and returns to the main and waits for another user input. Every time the mode select button is pressed, this ISR is called and these actions are repeated.

A counter will be used to determine if it is the first time the mode button is pushed, so that the clear display function is not called every time. This will also determine the DDRAM address that is sent to move the cursor to the correct location to print only the mode to the screen. Also note, that the failsafe timer is reset for every press, not just the first press.

Within the mode select ISR, the select button is enabled, so that the user is able to select a mode and move forward with the next selection (difficulty setting). The cancel button is also enabled so that the user can return to the welcome screen at any time.

13.7.2) Difficulty Select Button ISR

After the user has selected a mode, they move on to the difficulty selection (meaning the difficulty select button is enabled from the select button ISR). The program will call the difficulty select button's ISR each time the user presses the difficulty button. The ISR is similar to the mode button ISR, as each time it is pressed, the difficulty select register is incremented (and looped) based on the value of the mode select register. The failsafe timer is also reset for every press. The ISR will also call the set DDRAM address and write to DDRAM functions to print the specific difficulty message, depending on the mode selected and the value of the difficulty select register. Then, every time the difficulty button is pushed, the ISR repeats itself, and performs the correct function based on the new updated values.

Because this ISR is never calling the clear display function, a counter does not need to be used. However, similar to the mode button ISR, the beginning value of the difficulty select register will also need to be at the last option, so when the button is pushed, the first difficulty selection that appears is "Easy".

13.7.3) Select Button ISR

The select button is used for two different applications. When the mode button is enabled and pressed for the first time, its ISR enables the select button interrupt. The select button performs different tasks based on the number of times it has been pressed. Therefore, a counter is used to determine which tasks must be done. This is basically keeping track of what stage the user is at when using the UI system.

Once the user has made their mode selection and presses the select button for the first time, the counter is increased by 1. Next, the mode select button enable interrupt is turned off, so that when the user accidentally presses the Mode Button, the mode button ISR is not called. Next, the difficulty select button interrupt is enabled based on the mode select register. This allows the user to use the difficulty select button to cycle through the options, if applicable. The failsafe timer is also reset each time the difficulty button is pressed. However, if the mode is in the ambient mode, meaning the difficulty button is disabled, the timer will not reset. Next, the set DDRAM address function is called to move the cursor to position 1 of row 2, and prints the category title, only if the mode select register is set to 0 through 3. This is where the difficulty category is printed, by calling the write to DDRAM function. This allows the difficulty button ISR to only need to set the DDRAM address at the position where the difficulty type will be printed.

Once the user has pressed the select button twice, meaning they have made their difficulty setting selection, the ISR is called again and increments the counter. When this counter is set to 2, the interrupts for the mode, difficulty and cancel buttons are disabled, and the failsafe timer interrupt is also disabled. The ISR then moves the required data (mode select register value and difficulty select register value) into the UART transmit buffers, one at a time. Before the buffer is written to, the ISR checks to see if XBee is available and the transmit buffer flags are down. Then the ISR calls the set DDRAM address and writes to DDRAM to print the Start message on the fourth row.

After a suitable amount of time, the clear display function is called. Lastly, the ISR exits and enters a suitable low power mode that waits until the receive buffer interrupt flag is raised which calls the receive buffer interrupt. The mode and difficulty select registers are not reset to their initial values, as they are needed when the results are displayed.

13.7.4) Cancel Button ISR

As stated before, the cancel button is used to return to the welcome screen. This allows the user to “start over” with the mode and difficulty selections. The cancel button interrupt is enabled when the mode button is pressed for the first time and is disabled when the select button is pushed for the second time. Because the cancel button ISR performs the same action each time it is pressed (not mode dependent or status dependent), no counters will be necessary.

The first action performed is checking if the mode button interrupt is enabled. If not, it is enabled. Next the difficulty button and select button interrupts are disabled. Next, the mode and difficulty select registers are reset to the initialized value, as well as the mode and select button counters. Next, the clear display function is called, and the welcome message is called. This clears the display and prints the welcome message for the user, which instructs them to press the mode button. Lastly, the mode and difficulty select registers are set to their initial values, as well as the mode and select button counters.

The cancel button interrupt is enabled again once the MCU has received data from XBee. The failsafe timer is also started, and the interrupt for the timer is also enabled. This is because the user will require a way to return to the welcome screen once they have reviewed their results.

13.8) Timer Configuration and ISR

The UI and base system implement timers. In the UI system, a failsafe timer is used and in the base system a failsafe timer and various mode function timers are used. Because our design is dependent on receiving inputs from the user to proceed in the next step in the program, failsafe timers can be implemented to either move on to the next sequence (in the base system) or turn the system off (in the UI system).

When choosing which pin is diverted to be used as the failsafe timers, the ISR priority list must be referenced. In the event that multiple ISRs are called at the same time (unlikely, but possible), the MCU has a list of which ISRs have higher priority. The failsafe timers should have a lower priority than the button ISRs. This is because the design should move forward if the button is pressed as the timer goes off. This is partly due to the fact that when the buttons are pressed, the failsafe timers are renewed, meaning the interrupt is temporarily disabled and the interrupt flags are lowered. Then the timer's interrupt is re-enabled, and the timer is started. The other part is due to the fact that the design is waiting on the user to press a button. Therefore, when the user does press it at the same time as the timer ISR is called, it would not make logical sense to shut down (sleep).

13.8.1) Timer ISR

For the UI system, the failsafe timer's ISR is called if the timer is able to reach its end. Because each button ISR renews the timer, the timer ISR is only called if the user chooses to no longer select choices to begin a workout session. Therefore, when the failsafe timers ISR is called, the first function called is the clear display function. Then the display control command is called to turn the display off. Next, all button interrupts are disabled. The only way to turn the system on again, is to use the on/off sliding switch. First slide the switch to the off position to disconnect the power. Then slide the switch to the on position to power the device back on. This allows the MCU to begin at the top of the main, allowing all systems to re-initialize its systems.

For the base system, the failsafe timer's ISR is used to prevent the program from waiting for the user to press the start button. In the event that the failsafe timer's ISR is called, all button interrupts are disabled, the results functions are executed and the results registers are moved into the transmit buffers, to send XBee (and the UI) the requested data. Upon exiting the ISR, the MCU returns to the low power mode and waits for the receive buffer interrupt to be called. This allows the program to pick up at the UI system to print results (if any). If the user still has not interacted with the device, the UI system's failsafe timer will activate and send the UI system into a sleep mode.

13.9) General Mode Functions 0-4

When the base has detected that the user has pressed the start button, the mode function (dependent on the mode select register) is called. The mode functions each begin by starting the first timer, which controls the total length of the session. Depending on the mode, the first timer can be fixed, or variable (dependent on the difficulty select register). Next, the sensor and LED configuration functions are called. the second timer is configured and started. The rest of the mode function is controlled by interacting several ISR's (timer and sensor) to move through the session. The session ends when the first timer's ISR is called, which turns the LEDs and sensors off and continues to the results sections.

13.10) LED Use in Modes

The LEDs used to outline the target areas are controlled by the WS2811 driver. The driver is connected to the MCU over SPI to control the LED group, the LED brightness and the LED color. The color is controlled by sending 24 bits of RGB values (255,255,255). The brightness is controlled with 8 bits (255) and the LEDs are grouped into 3 (3 LEDs per group). During the mode function, these are the values that will be controlled to either turn on a target area or turn off a target area.

13.10.1) LED Functions

The LEDs are controlled using the FastLED library, which configures and sends data to the LEDs. To turn the LEDs on, the brightness will be turned up, and the color will be set with a RGB value. To turn them off, the brightness will be turned all the way down, and the color will be set to black (0,0,0).

13.11) Sensor Use in Modes

The sensors will be used within the mode functions, to retrieve data as the user interacts with the workout bag. The combination generator mode function reads from the binary sensors located at each of the target areas on Side A. The accuracy, reaction time and cardio modes functions reads from the ring of binary sensors on Side B. Because the ambient mode only uses the LEDs, the ambient mode does not read from any sensors.

The sensors are connected to the analog ports of the MCU. Within a while loop inside the mode function, the program reads from the ports. The analog ports return a bit value between 0 and 1023 from the ADC, which then is multiplied by the reference voltage of 5V and divided by the bit resolution, 1024. This provides an estimate of the voltage reading across the sensor. As the user hits a sensor, the voltage increases. Once the voltage increases enough to pass a threshold, the system reads the sensors again to see when the user's hand has released from the sensor. Once the voltage comes back down, the hit counter is incremented.

14.0) Results Calculation Functions

Once the mode function is completed and the program returns to the loop, the next function called is the results function, based on the mode select register. The results function reads the global stored data that was collected throughout the mode function. Each modes' results function is explained below. After each result is computed and sent, the registers are then cleared.

Note that Ambient mode does not have action to be taken, so no results need to be calculated upon mode completion.

14.1) Combination Generator Mode Results

During the combination generator mode function, the hit counter and target counter registers are accessed and calculated. The hit counter and target counter values are copied into the byte 1 and byte 2 registers. Byte 3 register value is found by dividing the hit counter value by the target counter value. When the program is ready to send the results data back to the UI system, byte 1, 2 and 3 registers are loaded into the transmit buffer, one at a time. It is important to note that the base system must send them in the same order that the UI expects to receive them.

14.2) Reaction Time Mode Results

During the reaction time mode function, the hit counter is calculated and an array is filled with each hit's timer value, where each index indicates the hit number and the value at that index is that hit's time. The hit counter value is copied into byte 1 register. Byte 2 register's value is found by finding the maximum value within the timer array, and byte 3 register value is found by finding the minimum value within the array. Lastly, byte 4 register is found by summing the time array, and dividing by hit counter. Then, byte 1-4 registers are moved into the transmit buffer, to send the data back to the UI system.

14.3) Cardio Mode Results

The cardio mode function of the hit counter is calculated during the cardio mode function, and the timer value is stored at the end of the mode. In byte 1, the hit counter value is stored. In byte 2, the one timer's value is stored. In byte 3, the time is divided by the hit counter, to give the average hit time (seconds per hit). In byte 4, the hit counter is divided by the time to give the hits per second value. All byte 1-4 are then stored into the transmit buffer, and then cleared.

14.4) Accuracy Mode Results

In the accuracy mode function, the hit counter is calculated and the time array is filled, where each index stores each hit's time. The target counter is also calculated during the accuracy mode function. In byte 1, the hit counter is stored, and in byte 2, the number of targets is stored. In byte 3, the hit counter is divided by the targets counter to give the accuracy rate and then in byte 4 the average of the hit time array is stored to give the average time per hit.

14.5) Function Summaries

The table below is a summary of all the functions that will be developed in order for the project to work correctly. Because the project has not been built yet, this list is not definite and is subjected to removing or adding functions to ensure the system works correctly.

The table's first and second columns hold identifiers of the functions used in this project. The function number found in column 1 is used to identify the functions within the table, while a descriptive name of the function or the ISR is located in column 2. The third column in the tables lists which functions are called inside that larger function; some functions will call several other functions in order for the correct actions to take place, while others will not call any other functions.

In general, the LCD commands, Display, LCD Communication, Zigbee, Button, Timer, Modes, Sensors & LEDs, and Results are listed within these tables.

LCD Commands		
Function #	Function Name/ ISR Name	Function (#) Called Within
1	Clear Display	15
2	Entry Mode	15
3	Display Function	15
4	Function Set	15
5	Set DDRAM	15
6	Write DDRAM	15

Table 27: LCD Command Function Summary

Display Functions		
Function #	Function Name/ ISR Name	Function (#) Called Within
7	Initialize LCD	1,2,3,4,15
8	Welcome Message	1,5,6,15
9	Mode Select	1,5,6,15
10	Difficulty Select	1,5,6,15
11	Start Screen	1,5,6,15
12	Results	1,5,6,15
13	Parsing Digits	N/A

Table 28: Display Function Summary

LCD Communication		
Function #	Function Name/ ISR Name	Function (#) Called Within
14	Initialize I2C	N/A
15	Send Data	N/A
16	Transmit ISR	N/A

Table 29: LCD Command Function Summary

ZigBee Functions		
Function #	Function Name/ ISR Name	Function (#) Called Within
17	UI Initialize UART	N/A
18	UI Send	N/A
19	UI Receive	N/A
20	UI Transmit ISR	N/A
21	UI Receive ISR	N/A
22	Base Initialize UART	N/A
23	Base Send	N/A
24	Base Receive	N/A
25	Base Transmit ISR	N/A

Table 30: ZigBee Function Summary

Button ISR		
Function #	Function Name/ ISR Name	Function (#) Called Within
27	Mode Button ISR	9, 34, 35
28	Diff. Button ISR	10, 34, 35
29	Select Button ISR	11, 18, 34, 35, 53
30	Cancel Button ISR	8, 34, 35
31	Start Button ISR	35, 40,41,42,43,44

Table 31: Button ISR Summary

Timer		
Function #	Function Name/ ISR Name	Function (#) Called Within
32	Initialize Timer	N/A
33	Start Timer	N/A
34	Reset Timer	N/A
35	Capture Timer	N/A
36	Fail Safe ISR	N/A
37	Timer 1 ISR	N/A
38	Timer 2 ISR	N/A

Table 32: Timer Functions and ISR Summary

Mode Functions		
Function #	Function Name/ ISR Name	Function (#) Called Within
39	Combo Gen Mode	38, 39, 47, 49, 50
40	Reaction Time Mode	38, 39, 46, 49, 50
41	Accuracy Mode	38, 39, 46, 49, 50
42	Cardio Mode	38, 39, 46, 49, 50
43	Ambient Mode	38, 39, 46, 49, 50

Table 33: Mode Function Summary

Sensors and LEDs		
Function #	Function Name/ ISR Name	Function (#) Called Within
45	IMU Comm. Initialize	N/A
46	IMU Receive ISR	N/A
47	Binary Sensor ISR	N/A
48	LED Comm. Initialize	N/A
49	LED On	N/A
50	LED Off	N/A

Table 34: Sensor and LED Function Summary

Results and LPM		
Function #	Function Name/ ISR Name	Function (#) Called Within
51	Compute	52
52	Send	23, 53

Table 35: Results Function Summary

15.0) UI Integration Test Plan

To ensure the UI device works accordingly, each component's code must be tested individually and then as a whole system. This is to ensure that each component's functionality operates as expected. These can't be tested all together, as a full UI system test might not trigger a bug that would be found by testing each functionality individually. Similarly, testing the UI system as a whole may introduce additional bugs that would have not been found by testing each component separately. In general, each component will be testing the expected action and result (actions and results that are correct) to ensure the components code works. In addition, each component will also test unexpected actions and their results (actions that are incorrect and results that handle the mistake) to ensure the code cannot be broken with a mistake (applies when the user inputting something that the code does not expect).

15.1) LCD and Button Interaction

To test the LCD and button interaction, a series of tests must be done for each LCD message (Welcome, Mode Select, Difficulty Select, and Start). Each button's (Select, Mode, Difficulty, and Cancel) corresponds to a specific LCD message that is displayed therefore, it makes logical sense to test these features together. First, the expected action and result will be tested for each message.

15.1.1) Welcome Message

The welcome message can be displayed in the event that one of two events occurs, when the system turns on and when the cancel button is pushed. The first event can be tested by simply turning the system on. When the display is displaying the correct message "WELCOME" centered on the first row, and "PRESS MODE!" centered on the third row. For the second event, the cancel button will be pressed in each stage (during mode selection, during difficulty selection and during the results display), to ensure the LCD display returns to the welcome screen. Once the select button is pushed twice (system is entering LPM), the system should not return to the welcome message if the cancel button is pushed. Each of these actions will be tested to ensure the system reacts appropriately.

15.1.2) Mode Select Message

The mode select message is displayed after the welcome message, while the user is selecting the desired mode. In this stage, the mode, select and cancel button are enabled, while the difficulty button is disabled. First, the mode button will be tested. The tester will need to navigate to the stage of selecting a mode, and pressing the mode button to cycle through the available mode select messages (5). The system should update the display with the correct mode (Combination Generator Mode → Reaction Time Mode → Accuracy Mode → Cardio Mode → Ambient Mode → Combination Generator and so on).

Next, the difficulty button will be tested. While the tester is still on a mode select message, the tester will press the difficulty button. The system should not react, as the button should be disabled. Lastly, the tester will test the select button, by pressing the select button while still on each of the mode select messages, the display should update to move to the difficulty select messages.

15.1.3) Difficulty Select and Start Messages

The difficulty mode select message is displayed after the mode select message, while the user is selecting a difficulty mode. In this stage, the difficulty, select and cancel buttons are enabled, while the mode button is disabled. The difficulty select options are mode dependent, therefore each mode will have to be tested for the difficulty select test. For each modes that have a difficulty setting, the tester will need to navigate to the stage of selecting the difficulty setting and pressing the difficulty button to cycle through the options (Easy → Medium → Hard → Easy and so on).

Next, the mode button will be tested, and the system should not react, since it is disabled. Lastly, the select button is tested, while pressing the select button on each of the difficulty select messages. The display should update to the start message. Next, the system will be tested for the ambient mode, which does not have a difficulty selection, by navigating to the stage of selecting the difficulty setting. The tester will press the difficulty button, the system should not react, since it is disabled. The process is repeated with the mode select button, which should have the same outcome. Lastly, the select button is tested, while pressing the select button on each of the difficulty select messages, for each mode type. The display should update to the start message.

15.2) Timer

In the UI system, there is one timer used which is the failsafe timer. To test the failsafe timer, the tester will navigate to each stage, the welcome message, each mode (5), and each difficulty setting for each mode (4 settings for 4 of the modes, 1 for one of the modes). At each stage, the tester will wait until the time elapsed is 5 minutes (300 seconds), using a stopwatch. The UI system turns the display off and enters a LPM, at the 5 minute mark. Next, the tester will repeat the process, and will wait various intervals within the 5 minute limit, and then press a button that is enabled during that stage. The tester will restart the stop watch and wait the 5 minutes to make sure the UI system is restarting the failsafe timer with each button press. Lastly, the tester will repeat the process again, and wait various intervals within the 5 minute mark (with the stopwatch going). The tester will press a disabled button at that stage and verify that the timer does not reset and the system shuts down at the 5 minute mark.

15.3) Results

The UI system is split into two basic components, before a mode is completed and after a mode is completed. To test the UI system after a mode is completed, a session must be done, for each mode and for each difficulty setting per mode. For this test, two testers are required, tester one to complete the session and tester two to record data.

At the beginning, tester two records the mode and difficulty setting of the test. Tester one completes the session, while tester two uses a stopwatch to record hit times, and count the number of hits, and the other required data for the session. At the end of the session, the testers compute the data scores that are expected to be displayed on the UI system, to compare what the system puts out versus what the testers calculated.

Although the timer values may not be 100% accurate, as the tester will have a delay of starting the timer for each hit, it will be enough to be able to see if the code is stopping and starting timers correctly. This test also verifies that the code is displaying the correct text based on the mode that is selected. Lastly, while on the results message, the all buttons will be tested to ensure the correct action is happening.

16.0) Complete Device Test Plan

All components need to be evaluated individually as detailed in the previous sections. After this is complete, the components will be tested in groups of components, before finally being tested as a whole system. This section will detail the grouped component and the system testing plans.

16.1) Side A and B: Pressure Sensors

The test plan for both sides is similar, since our choice of sensor is the same across the bag. Side A only requires testing of the combination mode, using 4 pressure sensors, and Side B requires testing of all other modes. As all sensors were created by us, we began by testing their basic functionality to confirm they fulfill the desired behaviors as pressure sensors for each expected input states.

The electrical parameters of the textile sensors were tested by connecting a 5-ohm resistor, one of the sensors, and a voltage source in series, and then testing this system with an oscilloscope and multimeter, to observe the voltage and current of a sensor while being hit. After this testing, we decided to use a 220 ohm resistor to protect the MCU from excess current on each sensor pin, while still allowing the range of possible sensor values to be large.



Figure 49: Sensor Reading of Five Quick Hits

To test the sensors on the actual device, we attached them to the bag using sewing pins, and pressed them each repeatedly and at random times. The voltage across each sensor was recorded during this process, helping us to determine the proper threshold values to register a hit. The MCU was coded based on the results of the procedure. The MCU used simple LEDs on a breadboard as stand-in indicators during the sensor testing process, so that we could easily move and adjust the sensors without worrying about large strips of LEDs in the way.

Since there are only two states (on or off) that the sensor will read based on our determined thresholds, it is sufficient to test each of the sensors with solely human input. Each of the members of this group exhibit different strengths and levels of experience, so there will be enough variety to thoroughly verify that the sensor components on side A are correctly functioning between the four of us. If more varied input were deemed necessary, it would be more efficient to use human subjects for testing as this will more accurately simulate the input range provided when being used as intended, in addition to saving the time that would have been used to set up and test using a specific setup as detailed in the section for side B.

16.3) Final Integrated Mode Testing

Once testing is complete for individual component clusters, our team will move from the previous testing methods to pure human interaction with the fully functional modes. Given the circumstances of the pandemic, we will begin by personally testing the system ourselves. This trial will include that each person of this group originally will attempt once each mode, so a minimum of 48 individual trial tests will be done for this project, four people attempting four modes in each three difficulty settings.

We will be saving everyone's feedback at the end of each session to compare how each different system influences people with different levels of endurance and strength, and we will be also recording each of these early sessions to compare the equipment behavior in each session and with each person. We will be able to additionally confirm results of "hit" versus "not hit" visually, and be able to confirm reaction time results by comparing the time recorded by the sensors to a team member measuring reaction time manually, with a stopwatch. After, we will be doing further testing within this group to also include how well each member interacted with the equipment by comparing each workout session, and we are going to use this feedback to calibrate the difficulty setting thresholds to something possible yet challenging as the level of difficulty states it should be.

Once the team members have finished the initial round of system testing, we will increase the tester sample group as safely as possible with the restrictions placed by CoVID-19. Precautions taken will include requiring every person coming to test the equipment to wear a mask, maintain social distance from other test subjects and team members, and to sign into a tester log, where they self-certify that have not been infected, nor have they are have not been in contact with a person who has been infected. We, as team members running the test, are also able to maintain the tester contact tracing log, provide amenities for safety such as hand sanitizer and gloves, and will disinfect the surfaces that have been touched by human subjects at the end of each section.

Aside from the coronavirus considerations listed above, the team must also conduct human testing trials ethically. No human test subjects are at risk for significant harm or injury while testing this device. We will only use consenting test subjects who volunteer to help with gathering feedback so that no one will be forced to complete a test session if they, at any time, decide that they would like to exit the test. All test subjects will be briefed at the beginning and debriefed at the end of the test session. We will guarantee all subjects will maintain anonymity by only collecting general demographic content (eg. age, gender, experience level, self-reported physical fitness level) and feedback on their experience using the device.

Test subject groups will vary with the time allotted to complete this project as well as the state of coronavirus in the world at the time. Ideally, we will bring the device to the dojo that Natesha works at (after obtaining written permission), and allow martial artists of all experience levels test out the device. If this is not possible due to time or growing levels of CoVID-19 infection in the area, we will use smaller and more controlled samples of close friends and family who are within the quarantine “bubble” of the team members. This sample may be less vast, but contains people from a variety of backgrounds, including at least 3 experienced martial artists. Either sample will be sufficient for gathering feedback on the user experience and the functionality of the device as a whole.

17.0) User Manual

This section will include information that is vital to the User Manual that will be provided with the device, including user considerations for safety and directions for use.

17.1) General Safety Considerations

This equipment can be used by individuals of 12 of age or older (ASTM F2276-10(2015)). This device should be used with enough space for the user to move around the equipment, to interact freely with it, and to avoid hurting others around them. If the user has medical conditions, they should ask their doctor before performing any type of interactions on the equipment. This device can be interacted by the user if they are wearing gloves or by their own bare hands. If the user becomes disoriented while interacting with this equipment, they must stop immediately and ask for help.

This device is required to be used in an indoor setting or environment (ASTM F2276-10(2015)). The user must interact with the screen to make any selections on the desirable mode and difficulty the user wants to interact with.

Please read and follow the directions for proper use of the device.

17.2) Directions for Use

1. Turn on the device's UI Remote. The Welcome screen will display. Press the mode button to continue.
2. Use the mode button to cycle through the mode options until you find the desired mode. Press select to continue.
3. Use the difficulty button to cycle through the difficulty options until you find the desired option. Changing difficulty may affect the length or speed of the mode, depending on the mode selected. Note that if Ambient mode is selected, there will be no difficulty options to select from. Press select to continue.
4. Place the UI Remote in the safe and secure place, such as the pouch included on the side of the device. It will not be needed until completion of the mode.
5. Position yourself on the correct side of the bag for your chosen mode. Combination Generator Mode will be on side A while the Accuracy, Reaction Time, and Cardio Modes will be on side B.
6. Press the start button on the side of the base of the device to indicate that you are ready to start the mode. After pressing Start, there will be a short delay for you to return to the correct position to begin the session. More information about each mode can be found below.
7. Otherwise, complete the mode to receive your results.
8. Once the mode has been completed, retrieve the UI Remote to view your results.
9. Press cancel to return to the welcome screen and choose a new mode, or power off device UI Remote.

17.2.1) Combination Generator Mode

This mode takes place on side A of the bag. Once the mode has started, press the areas that light up before the timer for the area is up. Your ultimate goal is to hit as many correct areas as possible before while they are lit. The sensors in each area will count how many times a correct hit was landed on the bag. After each hit is detected, the target area is turned off and the next random target area is activated. Once the session is complete, the handheld device will provide you with the number of correct hits landed, number of total targets, and your hit ratio of correct hits to total targets.

17.2.2) Reaction Time Mode

This mode requires use of side B of the bag. Once the mode has started, the target's indicators will light up as a signal to hit the target. Your ultimate goal is to have the shortest possible reaction time by hitting the target as quickly as possible once the indicator has lit up. The sensors and timers will determine your reaction time for each hit. Once the timer is complete, the handheld device will provide you with the total number of hits you landed, your average reaction time, your shortest reaction time of the session, and your longest reaction time of the session.

17.2.3) Accuracy Mode

This mode requires use of side B of the bag. Once the mode has started, the target area will light up as a signal to hit the target. Your ultimate goal is to hit the target as close to the center as possible for as many times as you can before time runs out. Once the timer is complete, the handheld device will provide you with the distance from the center (MAD) and time taken (MAT) to land your most accurate hit, the distance from the center (LAD) and time taken (LAT) to land your least accurate hit, your average accuracy over the session (ACR), and your hit rate (HPS) in hits per second.

17.2.4) Cardio Mode

This mode requires use of side B of the bag. Once the mode has started, the target area's indicators will light up as a signal for you to begin hitting the target. Your ultimate goal is to increase your heart rate by hitting the target as many times as you can before time runs out. Once the timer is complete, the handheld device will provide you with your total number of hits, the overall time of your session, your average time per hit, and your hit rate (HPS) in hits per second.

17.2.5) Ambient Mode

This mode is not an exercise tool, but instead will illuminate the LEDs equipped in a multicolor pattern for the duration of the internal timer. Once the timer is complete, the bag's lights will turn off and the UI handheld will return you to the Welcome screen. If you wish to turn off the lights before the end of the timer, simply reset the system with the remote.

18.0) Administrative Content

This section contains content regarding managing our group and the members within the group. We will lay out all the planning that is required to ensure a final product is complete, the familiarities and unfamiliarity to our group and as individuals, and the expected budget for this project. These details are important since we are looking at how this project will be distributed between the members of the group, and how they complement each other to make this project a reality. The budget is also included in this section to illustrate the reader the exact amount that this project is expected to cost.

18.1) Initial Milestones and Timing for Senior Design 1 and 2

Milestones serve as a checklist of the overall progress of a project, and by putting a deadline alongside the milestone, it helps us to track progress with the project. Like in industry, we as engineers, need to keep up with the milestones and timeline in every project we have. If this is not followed, then every project would have very significant delays that would delay the completion of the project. In general, a prototype will first be built and tested. Next, those components will be moved over to the final product, and tested again to ensure all functions are working properly. If the schedule is followed correctly, the final product will be completed and ready before needed, to allow us time to prepare our presentations.

18.1.1) Prototype Design & Build: January – March 2021

Prototypes for many of this device's components were completed in January and used to test many parts of the design prior to making a more final device prototype. The first full design prototype, completed in March, included a prototype base system and a UI system which simulated the connectivity of the final product, as well as prototype software to handle this communication. The textile sensors were created and would continue to be used into the final version of the product, with minor adjustments. The RGB indicators were modeled with simple LEDs on a breadboard in this stage, so that the proper indicating behavior could be tested without needing to worry about mounting long strips of wire around components which were not finalized and may have needed to be moved. After completing this prototype in March, we had time to debug issues that arose as well as move all components as needed for a final build later on.

The first revision of our printed circuit boards was ordered in January to ensure they arrive and parts are mounted for the final build. This allows us to have extra time in case design mistakes have happened, and require new boards to be printed and new components to be mounted. This will also allow us to test the circuit board to ensure everything is working correctly for when we move on to the final product build.

18.1.2) Prototype Testing: February 2021

Once the prototype is built and operational and the testing apparatus is built and configured, the first round of the testing sequence will begin. Each test plan will be executed and any bugs will be noted and corrected. Depending on the number of bugs found, testing will be repeated to ensure they've been corrected and are no longer persistent. Because this timeline is variable, the absolute end date for testing, debugging and retesting the prototype is until the end of February. This allows us one month to square away any issues present in the prototype, and to ensure the code is working properly.

During this month, if the PCB has had issues, we are able to use this time to remount spare components, and retest the board to make sure it is ready for the final product build. In addition, the final product body of the wave master will be modified to ensure a smooth transition of all the components needed. This allows us extra time in case serious modifications are required to fit all the subsystems into the body of the wavemaster in an aesthetic way. The UI system case will also be sent in for production to be 3D printed during this time frame.

18.1.3) Final Build & Test: March - Mid April 2021

Once the prototype is bug-free and working properly, and the PCB is working properly, the prototype will be disassembled and components will be moved over onto the final product. Because the final product body was modified ahead of time, the process of moving all components over will be a smoother transition, than rushing during this short time frame. In addition, the UI system case will be ready to assemble and have a final build as well.

Once all systems are built, the next series of testing will be executed. We will begin testing within the group. As before, any bugs that are found will be noted, corrected and retested, until no issues remain. As the preliminary round of testing concludes, we will increase test sample size according to the allowed practices outlined in local coronavirus guidelines once this time arrives.

18.2) Group Member Familiarity with Project Elements

Hannah is familiar with software design since she works in a company which specializes in designing software architecture, and enjoys coding in her free time, in C, C++ and Python. In addition, she has first-hand experience with implementing embedded programming for LCD displays, buttons, interrupt service routines, serial communication and timers. Three main courses, Embedded Systems, Junior Design and Engineering Analysis, provided her with the experience needed to become familiar with these components to integrate them into a complex system. Although Hannah hadn't directly worked with Zigbee, she did have experience with designing communication systems from taking the Computer Communications class, which deals with designing the layers of the network, which Zigbee builds on. Joseph is familiar with processors and microcontrollers.

Along with the standard circuits courses, Joseph has taken an in-depth course on power systems fundamentals. Methods of conversion and system power were covered in the course, including conversions and transformation. This background assisted in the design of the schematic and PCB for the circuitry inside the workout bag and the remote. The main courses that are useful for the design are Junior Design, Linear Circuits 1 and 2, Electronics 2, and Power Fundamentals. These courses cover moderate circuit design to intermediate circuit design, especially useful for the mostly linear circuits that will be used for the project. Another course, embedded systems, was good background knowledge for building circuits with microcontrollers. Joseph's experience from this course assisted in choosing and implementing a microcontroller design for the remote and the punching bag.

Nicole took an in-depth sensors class, so she is familiar with various types of sensors that we could use for this project. Additionally, within her sensors class, she had to design and build a sensor using passive elements, as she did for this design. She is comfortable using the skills that she learned in the class for this project. The same approach of the passive sensor requirement from that class was used for the binary sensor due to the size that the targets on side A are planned to be. Nicole also enjoys sewing as a pastime, so she was able to help with mending the bag once it was opened to hold components such as sensors and LEDs.

From previous classes Natesha has taken, she has experience in addressable LED design and programming for sensors and LEDs. She also has experience creating PCB designs for test boards from her time helping a professor in their lab. Additionally, she has designed and created 3D-printed models or otherwise constructed models in the past, which will help with construction of the device and any cases that must be created for the UI handheld. In addition to knowledge of typical electrical engineering curriculum, Natesha has many years of experience with and regularly uses a self-standing kicking bag since she is a fourth-degree black belt in karate and works as a sensei in her dojo.

With the group's combination of knowledge of electrical engineering principles and real-world experience with all aspects of this project, we were able to successfully create and develop this device.

18.3) Project Challenges and Unfamiliarity

Most challenges that our group will be facing are a result of the CoVID-19 pandemic that we are currently living through. Due to social distancing, we will need to work on our sections separately; as of now we will not be able to get together unless it is to test the device. As of now we are doing most of our meetings and communication via Discord, but once sensor/indicator testing begins, we will likely face difficulty in being sure everyone has what they need to individually test components.

In general, being able to optimize the parts used to cater to all specifications as well as being low cost, accessible, and realistic for the scope of this design prototype will be a challenge. The pandemic has also slowed electrical component production around the world, meaning that we will have to be sure about our designs before ordering many items and begin attaining and assembling components as soon as possible in attempts to avoid roadblocks.

One non-pandemic related obstacle is the size and scale of this project. A self-standing training bag is a large and heavy piece of boxing equipment, so this component should be in a location accessible to all members with sufficient space to work on device assembly and construction.

18.3.1) Unfamiliarity with Project Elements

Throughout this project, we as a team have gained large amounts of knowledge regarding implementation of elements that we have not worked with before. The most significant challenges we have overcome in terms of group unfamiliarity with project elements are wireless communication with ZigBee and utilizing specialty sensors such as IMUs on a project of this scale.

Our team had varying amounts of experience with sensor hardware, such as gyroscopes and was knowledgeable about the sensors such as gyroscopes, accelerators, and switch/button type sensors. However, none of the team members had heard of the IMU sensor prior to the suggestion of our advisor, Dr. Richie, to consider IMUs as a candidate for the sensors used on side B. Since then, we have gained experience with the theoretical side of using IMUs, including connections required and using the data gathered. We will gain practical experience using IMUs when we physically implement them next semester.

A large aspect of unfamiliarity for this project rests in the hardware components. Although all members of the group have experience designing and prototyping PCBs (with breadboards), no member of the group has experience in putting in an order for a PCB or in mounting parts to a PCB. There are several different methods that will be considered for this task. There are multiple fabrication shops around the Orlando area that will be able to create PCBs for each component of the project. Several PCBs are necessary to have an element of redundancy for mounting. Mounting the components to the PCB does have a couple different methods to consider. One option is to mount the components at a professional shop. This option will result in the highest quality finished product, but hands on experience is not gained and additional cost will be added to the final budget. Another option would be to solder the parts ourselves onto the PCB, possibly with the help of an oven. Using this oven and solder method does cost less than the previous option at no additional cost and critical experience is gained, however this option has a high possibility of failure. It seems reasonable that the second option could be tried with the extra PCBs then the possibility of professional mounting can be saved as a backup if all of the extra PCBs are unsuccessfully mounted. This will be a large aspect of growth once this process begins in the second semester.

The biggest portion of the software design that our group is unfamiliar with is Zigbee. Zigbee is the generalized name for the z-stack, XBee chip and XBee antenna that is used to create a WPAN for the UI system and base system to communicate. Although our group has never implemented this type of technology in a real world scenario, there is a considerable amount of resources to implement Zigbee in our project to make the microcontroller units communicate with each other. But because our group has not used it before, UART communication will be used as a backup plan for the MCUs to communicate, as we are familiar with that type of serial communication. Lastly, we have not had any experience with the code needed to retrieve data from IMU sensors. Although they use SPI as the communication protocol, this will be the first experience on configuring and reading from IMU sensors in an embedded environment.

18.4) Budget and Financing

Our group will self-fund this project. We are budgeting to contribute a maximum of \$900 USD, divided equally among our teammates. As of now, no equipment aside from standard circuit components (eg. wires, resistors) has been acquired. All parts of this project must be attained at some later point. Table 25 below depicts the estimated costs of each component along with the number of components required.

The individual component that would cost the most money is a self-standing kicking bag since it is a big and technical instrument aimed at people who are experienced in this area. However, finding a used bag will lower the cost of this item, and will not make a difference in the quality of the bag as it will need to be changed to incorporate our components regardless of its condition.

Although sensors are generally inexpensive, we would need enough components to detect contact in the large target area so that the device can test accuracy in addition multiple sensors to detect contact in a binary “hit or no hit” manner in all areas of the sensor grid.

Additionally, the inclusion of two separate PCBs in our design requires factoring the production of each type of PCB. This means that we will have to include enough PCB budgeting to fund multiple versions of the board in case of errors or damages in the testing phase of the build.

An important consideration in our budget will be shipping costs. Shipping costs are not initially present when viewing prices, but add up if ordering from many different sources at different times. To combat this issue when ordering as many parts as this project requires, our team will aim to have neatly planned shopping lists organized by store so that we can combine orders as well as possible. This will allow one payment of shipping to extend to more items, which is vital as the items themselves, especially basic circuit components, are so low in price. Another strategy will be to buy in bulk if the part is one that we plan to use many of within the build of the device, such as the IMUs and PCBs. However, components that do not require much volume will be bought independently so our team avoids surplus inventory that will go unused.

Component	Price (USD)	Quantity	Total (USD)
Conductive Thread	\$ 5.99	2	\$ 11.98
Velostat/Linqstat	\$ 4.95	5	\$ 24.75
Conductive Fabric	\$ 4.95	2	\$ 9.90
Foam Sheet	\$ 12.99	2	\$ 25.98
LCD Screen	\$ 12.25	1	\$ 12.25
LED Strips	\$ 25.99	2	\$ 51.98
LED Pixel Lights	\$ 18.99	2	\$ 37.98
Arduino Uno	\$ 23.00	2	\$ 23.00
Training Bag	\$ 120.00	1	\$ 120.00
Xbee Antenna	\$ 26.00	2	\$ 52.00
Xbee Explorers	\$ 12.00	2	\$ 50.00
Buttons	\$ 10.99	15/pack	\$ 10.99
Breadboard Power Supply	\$ 7.99	2/pack	\$ 7.99
120V AC to 12 V DC Converter	\$ 14.00	1	\$ 14.00
eBotot Mini DC/DC Converter	\$ 9.00	6	\$ 9.00
PCB Production		4 (2 ea.)	\$ 150.00
AA Battery Holder	\$ 7.99	1(6each)	\$ 7.99
Styrofoam	\$ 8.99	2	\$ 17.98
Miscellaneous	\$ 97.78	1	\$ 97.78
Total Cost ≈ \$736		Expected Cost ≈ \$800 to \$900	

Table 36: Expected costs for this project

19.0) Conclusion and Next Steps

While we encountered many usual struggles like finding the ideal components, the debugging of the code, and soldering the materials, we also faced the obstacle of COVID-19. Weekly meetings were held virtually if possible. All team members did COVID self checks before meeting in person, and personal protective equipment was used when necessary. The Interactive Self-Standing Training Bag project requirements were listed and taken into account in the conceptual stage of this project.

The design for this project was carefully measured since we wanted the handling of data between components detailed, spread out across the equipment, and easy to process in the end to analyze each interaction from the user to give them their feedback at the end of their session. Considerations were made to make this equipment safe and that it followed guidelines to also make sure the user is also safe to use our device. All types of testing will be done by the creators to properly assess that this product is behaving the way is desired and to bring to the user an optimal way to exercise. It was very important to incorporate the group's concept ideas, interest, and technical skills into this final project since we wanted to make this project a fun one but also impactful, meaningful, and innovative. One of the accomplishments of this project is that we can improve a person's wellbeing in a more detailed and fun way without having an extra person to assist them.

Another accomplishment would be using Zigbee in our system. No one in our group has had any experience with integrating Zigbee as the communication system before, however, our group does have experience with designing the network protocol layers, which Zigbee builds upon. Zigbee is configured with a program (XCTU) that uses a graphic interface to set up the wireless personal area network that the two XBee chips communicate in. This accomplishment would make our product more appealing and safer, due to the lack of external wires that the user could trip on.

The implementation of the Handmade Pressure Sensor was a suggestion of our teammate, Natesha, and it was a very interesting application. She found out that this is how light up shoes are being done with materials bought from Adafruit. Due to the Handmade Pressure sensor ability to accurately track the area of the hits by dividing the area of impact in sections, this will be a unique way to measure the hits and their section of the interaction of the user. The size of this sensor can vary, yet it is a strong, accurate, and durable component for all the sides and modes for this project.

The recommendations that we have if other people are trying to recreate this project in the future are the following: the MCU, power supply for the components, and cabling to and from the devices for measurement should be in a place that will not move such as the base of the bag; the control should be made a little more bigger to have the battery in the inside of the control or having an app that interacts with the bag instead; and, PCBs should have been the first thing ordered to have tested on them and fix any potential issues sooner than later.

The biggest goal and hope that we were set to achieve is that getting into fitness should not be expensive and to expand the limited variety in the current market. Fitness equipment should allow the user to be more interactive with the product and should make their interaction with the equipment worthwhile, especially now that everyone is in their homes and unable to go to gyms due to COVID-19. Everyone should be able to have fun while working out in the commodity of their homes at a reasonable price.

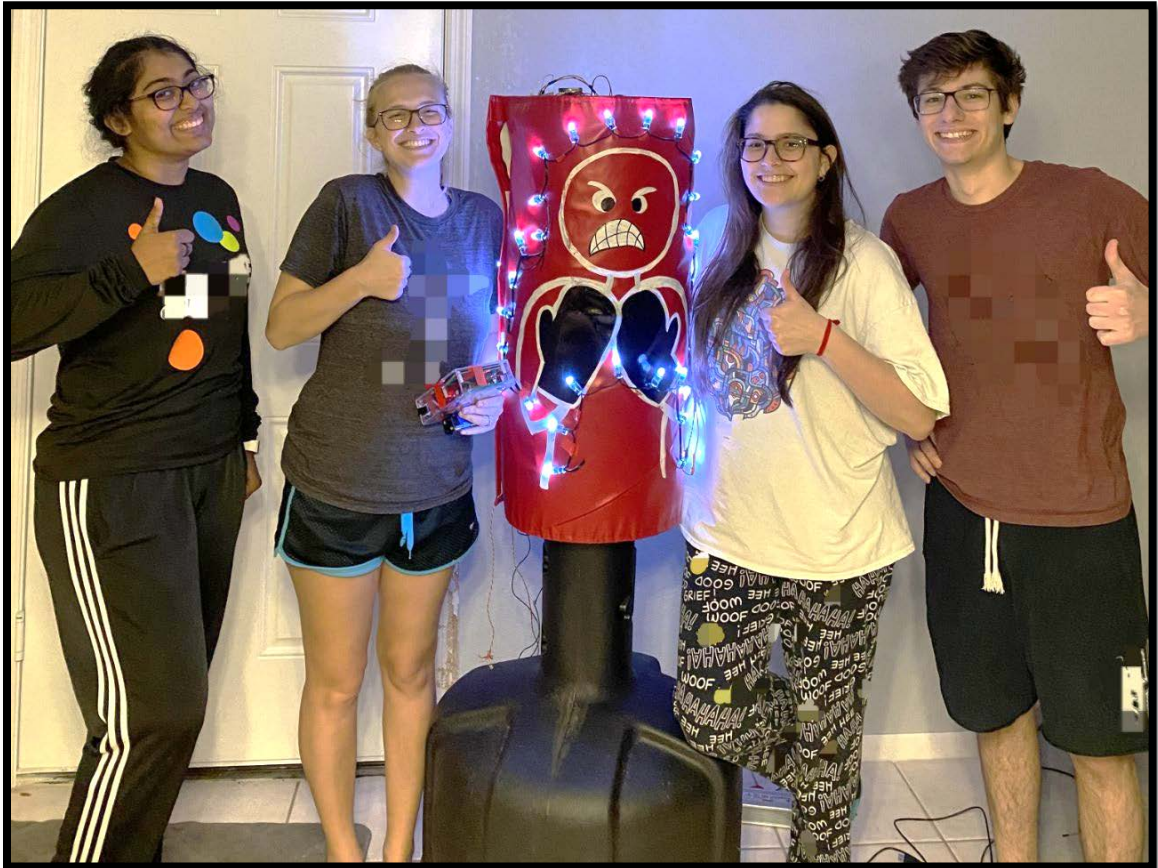


Figure 50: Group 22 with their completed device

References

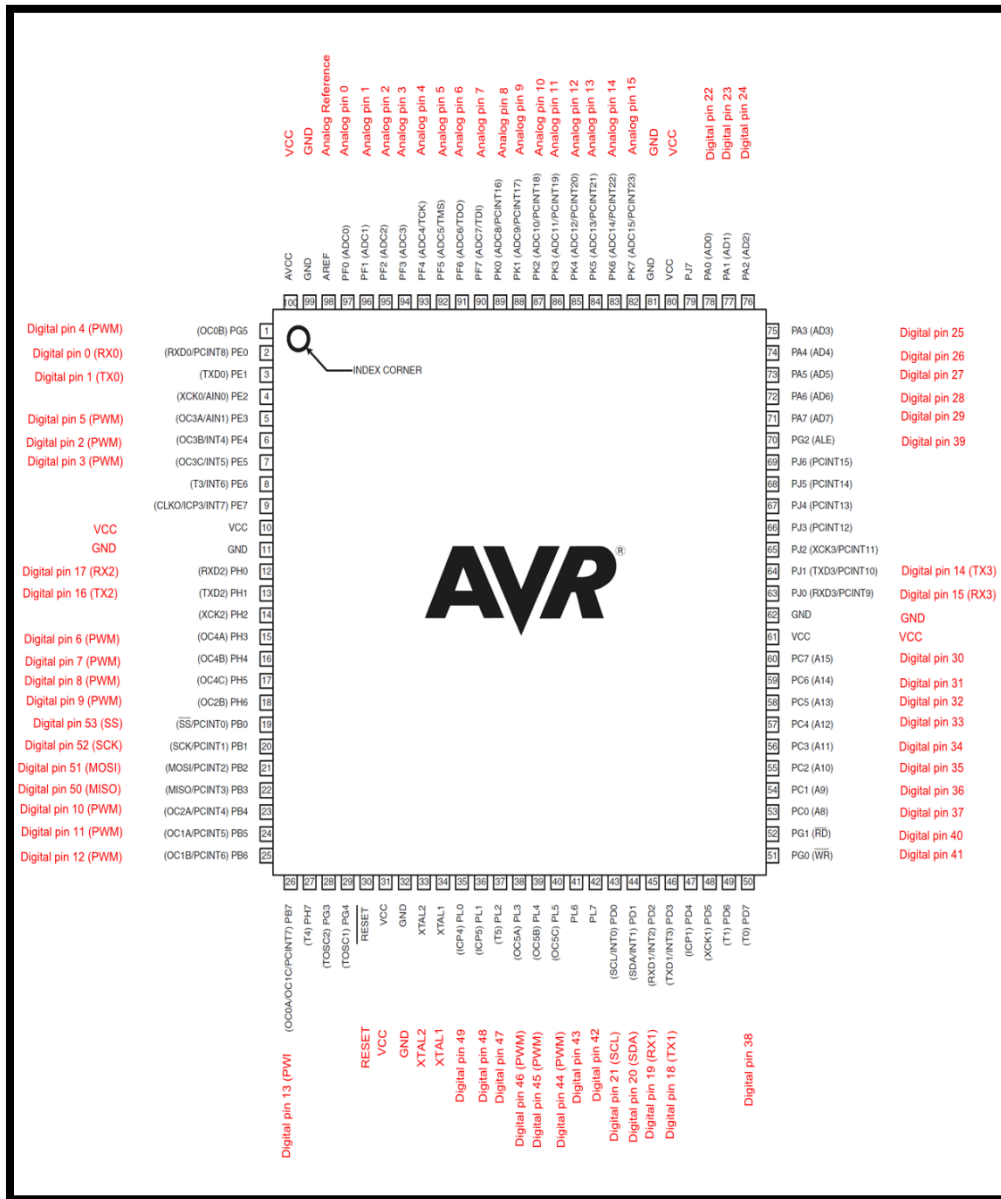
- [1] A. D. Morales, M. C. Morales and L. C. Eldridge, Jr., "Interactive systems and methods for reactive martial arts fitness training". US Patent US20110172060A1, 18 11 2010.
- [2] Atmel " Atmel ATmega 640/V-1280/V-1281/V-2560/2561" Microchip [Online]. Available www.microchip.com
- [3] ALITOVE Store, "ALITOVE RGB Addressable LED Strip WS2811 12V LED Strip Lights," Amazon, [Online]. Available: amazon.com.
- [4] American Heart Association, "Warm Up, Cool Down," American Heart Association, 1 09 2014. [Online]. Available: <https://www.heart.org>.
- [5] Association Connecting Electronics Industries, "IPC 2221B," Techstreet, 01 11 2012. [Online]. Available: <https://www.techstreet.com>.
- [6] ASTM International, "ASTM F2276 - 10(2015)," ASMT International, 2015. [Online]. Available: <https://www.astm.org>.
- [7] D. Hunt, "MSP430F6459LP example code: LCD Counting Example Using a 4-bit Interface," davesmotleyprojects, 2019. [Online]. Available: <http://www.davesmotleyprojects.com>.
- [8] D. Lapkova and M. Adamek, "Using Strain Gauge for Measuring of Direct Punch Force," *XXI IMEKO World Congress "Measurement in Research and Industry"*, 2015.
- [9] D. Lapkova, L. Kralik and M. Adamek, "Possibilities Of Force Measuring In Professional Defense," *XXI IMEKO World Congress "Measurement in Research and Industry"*, 2015.
- [10] Digi International Inc, "XBee 1mW Trace Antenna - Series 1 (802.15.4)," Sparkfun, [Online]. Available: <https://www.sparkfun.com/products/retired/11215>.
- [11] Duke Energy Indiana, "Average Duke Energy Electric Bill," Duke Energy, 08 05 2020. [Online]. Available: <https://www.citact.org>.
- [12] Environment and Protection Agency, "Regulations, Initiatives and Research on Electronics Stewardship," Environment and Protection Agency, [Online]. Available: epa.gov.
- [13] F. D. o. E. Protection, "Electronic Waste," Florida Department of Environmental Protection, [Online]. Available: <https://floridadep.gov/waste/permitting-compliance-assistance/content/electronics-waste>.
- [14] Hitachi, "HD44780U(LCD-II)," Hitachi, [Online]. Available: <http://site.gravitech.us/MicroResearch/Others/LCD-20x4B/HD44780.pdf>.
- [15] International Electrotechnical Commission, "IEC 60906-2:2011," Webstore, 12 05 2011. [Online]. Available: <https://webstore.iec.ch>.

- [16] InvenSense, "MPU-9250 Product Specification," InvenSense, 20 June 2016. [Online]. Available: <https://invensense.tdk.com/wp-content/uploads/2015/02/PS-MPU-9250A-01-v1.1.pdf>.
- [17] InvenSense, "MPU-9250 Register Map and Descriptions," InvenSense, 7 January 2015. [Online]. Available: <https://invensense.tdk.com/download-pdf/mpu-9250-register-map/>.
- [18] J. Lightfoot, "Get Started with XBee – A Beginner’s Tutorial," Atomic Object, 18 July 2016. [Online]. Available: <https://spin.atomicobject.com/2016/07/18/xbee-tutorial/>.
- [19] J. T. Adams, "An Introduction to IEEE STD 802.15.4," IEEE, 24 07 2006. [Online]. Available: <https://ieeexplore.ieee.org>.
- [20] [J.-P. Rodrigue, "Point-to-Point versus Hub-and-Spoke Networks," Hofstra University, 2020. \[Online\]. Available: <https://transportgeography.org>.](#)
- [21] Lady Ada, "Overview |EL Wire," Adafruit, 29 07 2012. [Online]. Available: <https://learn.adafruit.com>.
- [22] Longrunner, "for ArduinoIDE, Longrunner 20x4 LCD Display Module IIC/I2C/TWI Serial 2004 with Screen Panel Expansion Board White on Blue, 4 pin Jump Cables Wire Included," Amazon, [Online]. Available: <https://www.amazon.com>.
- [23] [M. Alves, "The IEEE 802.15.4/ZigBee protocol stack architecture," ResearchGate, \[Online\]. Available: <https://www.researchgate.net>.](#)
- [24] Raspberry Pi Foundation, "Raspberry Pi 4 Tech Specs," Raspberry Pi Foundation, [Online]. Available: <https://www.raspberrypi.org>.
- [25] "SparkFun XBee Explorer USB," Sparkfun, [Online]. Available: <https://www.sparkfun.com/products/11812>.
- [26] ST Microelectronics NV, "Data Sheet iNEMO inertial module: always-on 3D accelerometer and 3D gyroscope," [Online]. Available: <https://www.st.com/resource/en/datasheet/lsm6dsr.pdf>.
- [27] Systronix, "Systronix 20x4 LCD Brief Technical Data," Systronix, 31 July 2000. [Online]. Available: http://www.systronix.com/access/Systronix_20x4_lcd_brief_data.pdf.
- [28] T. K. Hareendran, "EL Wire Experiments & A Minor Hack," Electro Schemantics, 02 05 2019. [Online]. Available: <https://www.electroschematics.com>.
- [29] Texas Instruments Incorporated, "66AK2G1x Multicore DSP+Arm KeyStone II System-on-Chip (SoC) (Rev. F)," Texas Instruments Incorporated, [Online]. Available: <https://www.ti.com>.
- [30] Texas Instruments Incorporated, "MSP430F6459," Texas Instruments Incorporated, 11 9 2020. [Online]. Available: <https://www.ti.com>.

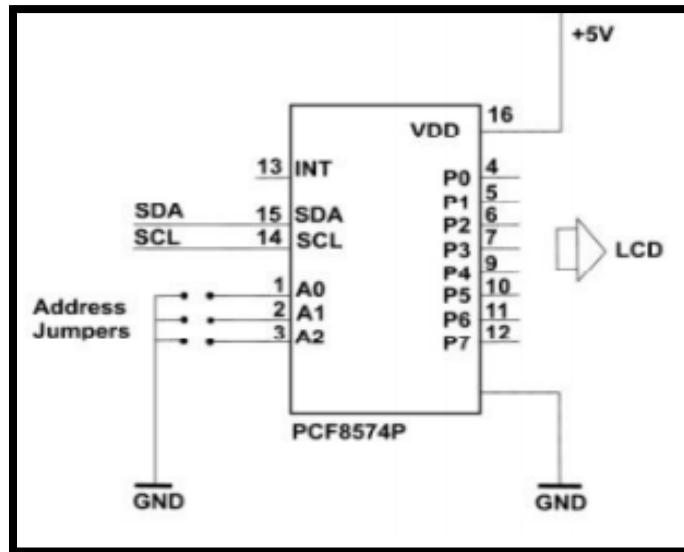
- [31] Texas Instruments Incorporated, "ZigBee RF Transceiver Datasheet," Texas Instruments Incorporated, 12 2007. [Online]. Available: ti.com.
- [32] The Hook Up, "The COMPLETE guide to selecting individually addressable LED strips," The Hook Up, 5 08 2019. [Online]. Available: <http://www.thesmarthomehookup.com>.
- [33] "Wavemaster," Century, [Online]. Available: <https://www.centurymartialarts.com>.
- [34] yida, "UART vs I2C vs SPI – Communication Protocols and Uses," Seed Studio, 25 09 2019. [Online]. Available: <https://www.seeedstudio.com/blog/2019/09/25/uart-vs-i2c-vs-spi-communication-protocols-and-uses/>.
- [35] yida, "UART vs I2C vs SPI – Communication Protocols and Uses," Seed Studio, 25 09 2019. [Online]. Available: <https://www.seeedstudio.com/blog/2019/09/25/uart-vs-i2c-vs-spi-communication-protocols-and-uses/>.

Appendix

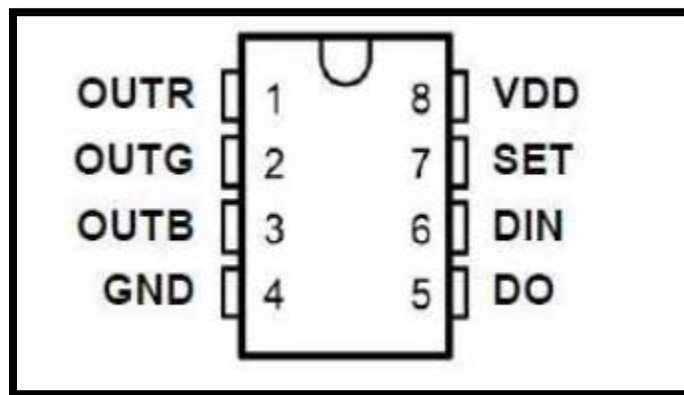
A) Pinout Diagram of ATmega2560



B) Connection for LCD



C) WS2811 LED Strip Driver



D) XBee Pin Diagram

