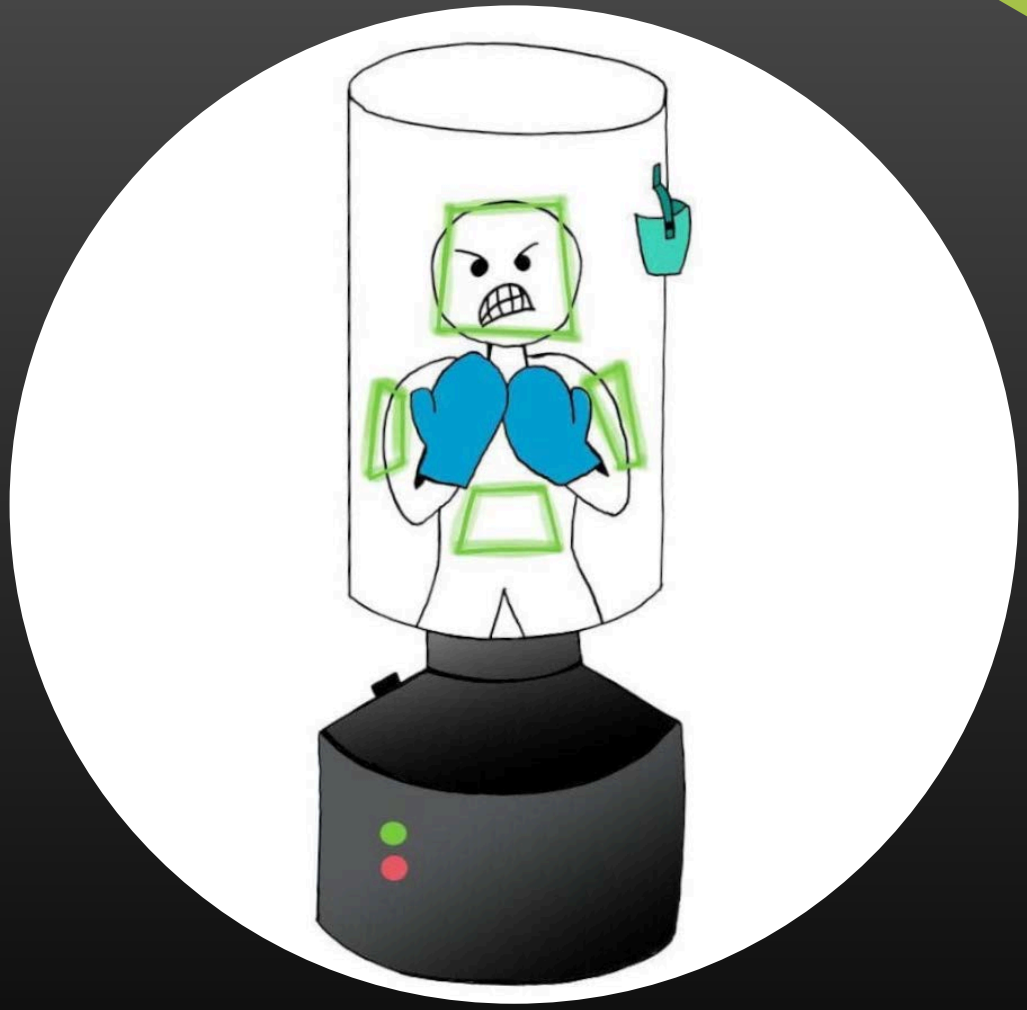


Interactive Self-Standing Training Bag

Critical Design Review

UCF CECS, Spring 2021

ECE Senior Design Group 22

Hannah Clarke
Joseph De La Pascua
Nicole Karam Pannaci
Natesha Ramdhani

Team Members

(listed clockwise from top left)

Nicole Karam Pannaci (EE):

Sensors and Crafting Lead

Hannah Clarke (EE):

Software and Communications Lead

Joseph De La Pascua (EE):

Power and PCB Lead

Natesha Ramdhani (EE):

Design Concept and Indicator Lead

Overview

- Motivation and Goals
- Specifications, Requirements, Standards, and Constraints
- Design Concept
- Selected Components
- Software Design
- Administrative Content
  - Current Progress
  - Next Steps

Project Motivation and Goals

Project Motivation

Train Wherever User Needs To

- Train without a partner
- Train without needing to physically be at a special location
- Especially relevant due to CoVID-19 Quarantining!

Create Workouts and Track User Progress

- Enable specific workout goals for user
- Flexibility within workout structure
- Focus on workout instead of tracking own performance

Cost of Existing Products

- No monthly cost needed to use on its own
  - ~~Gym~~
  - ~~Dojo~~
  - ~~Personal Trainer~~

Existing Products

PADIPATA

- Hanging bag
- Covered in sensor material
- Retails for $25,000+
- New product, not many reviews
  - Recently completed Kickstarter campaign
  - Not much knowledge about actual functionality as told by real users

FightClub

- Standing bag
- Must be used with connected boxing gloves or hand wraps
  - Only senses punches
- Package containing gloves retails for $1,219+
- Works with app
  - Separate paid membership needed

Goals and Objectives

Cost

Make device cost effective to produce and affordable for the consumer.

Interaction

Make device interactive for the user.

- Use indicators and sensors.
- Provide feedback to user.

Experience

Include multiple training modes to diversify the user's workout experience.

Specifications, Requirements, Standards, and Constraints

Overall Functional Specifications

Downtime Modes

Idle Mode

Ambient Mode

Off

Active Modes

Combination Generator (Side A)

Cardio Mode (Side B)

Reaction Mode (Side B)

Accuracy Mode (Side B)

Marketing Requirements
for device

DURABLE

AFFORDABLE

PORTABLE

INTUITIVE

VERSATILE

LONG-LASTING

Technical Requirements
for components

- **DURABLE** (CAN TAKE IMPACT OF 120 HITS/MIN)
- **EFFICIENT** (DEVICE POWER < 1.1KW)
- **COMPATIBLE** (WITH OTHER DEVICE COMPONENTS)
- **AFFORDABLE** (WITHIN $800 BUDGET)
- **SIMPLE TO IMPLEMENT** (WITHIN 12 WEEKS)
- **LONG-LASTING** (DEVICE LIFE ≥ 5 YEARS)

Constraints

Economic

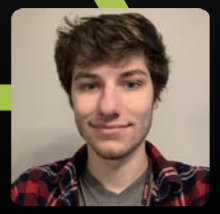
- Budget of $800
- Self-funded
- Running cost of electricity

Time

- Finished product by end of Spring 2021 semester
- Mid-Project deadlines

Environment and Safety

- Low-power operation design
- Keeping weight as low as possible
- Heat of electric equipment

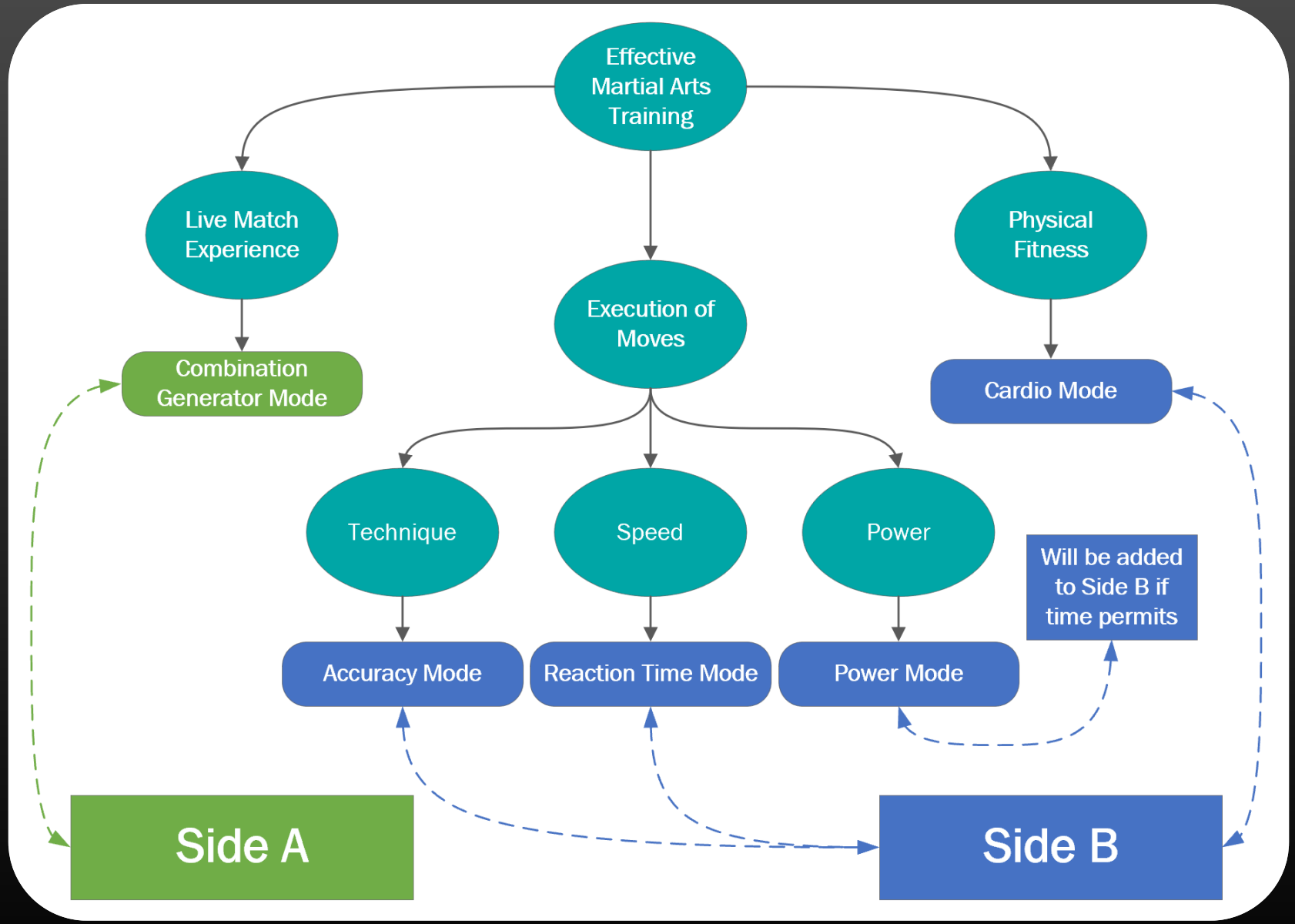


Related Standards

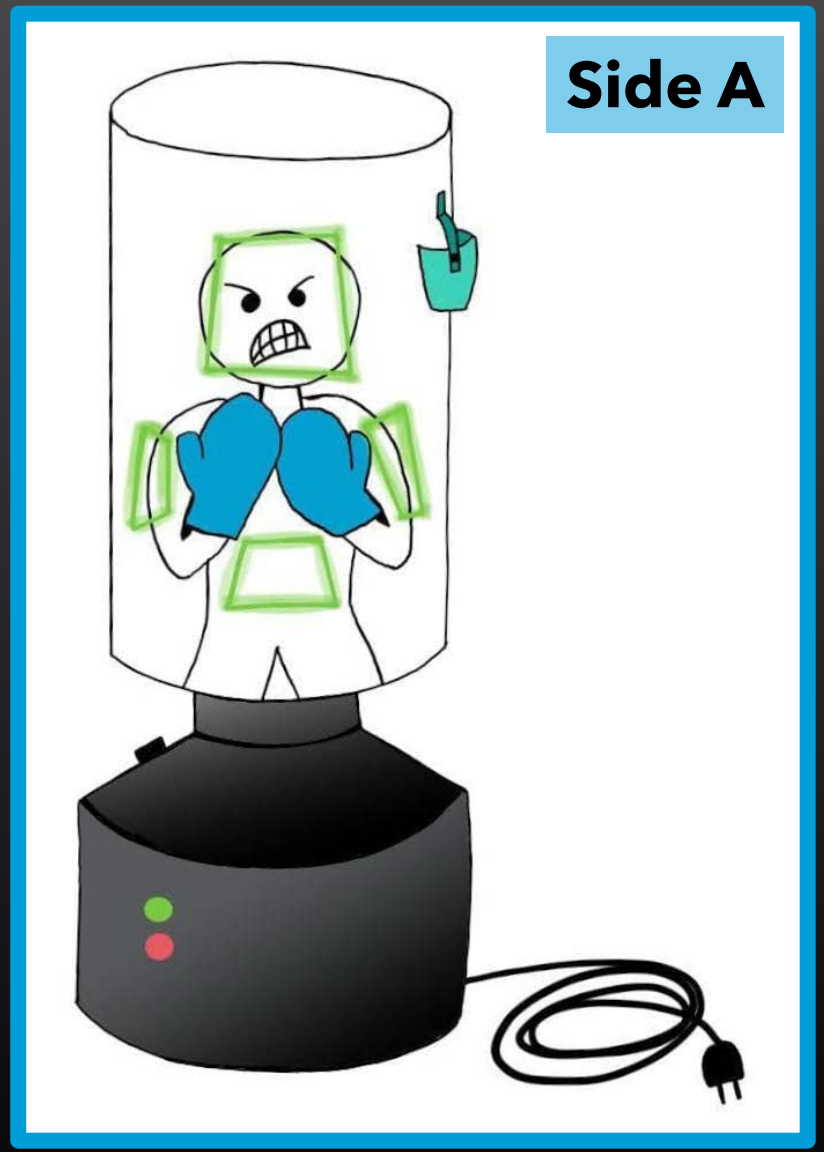
- **Power Supply**
  - IEC 60906-2:2011
  - 3-prong ground plug (NEMA 5-15R and 5-15P)
  - Provides grounding and electrical noise immunity
- **Regulation Workout Bag Standards**
  - ASTM F2276-10(2015)
  - Age Restriction, Documentation on assembly and build, adequate warning labels, indoor setting
- **PCB Standards**
  - IPC-221B: Generic design requirements for Printed Circuit Boards and Component Mounting
- **IEEE 802.15.4**
  - Physical Layer and MAC Sublayer
  - RF Parameters

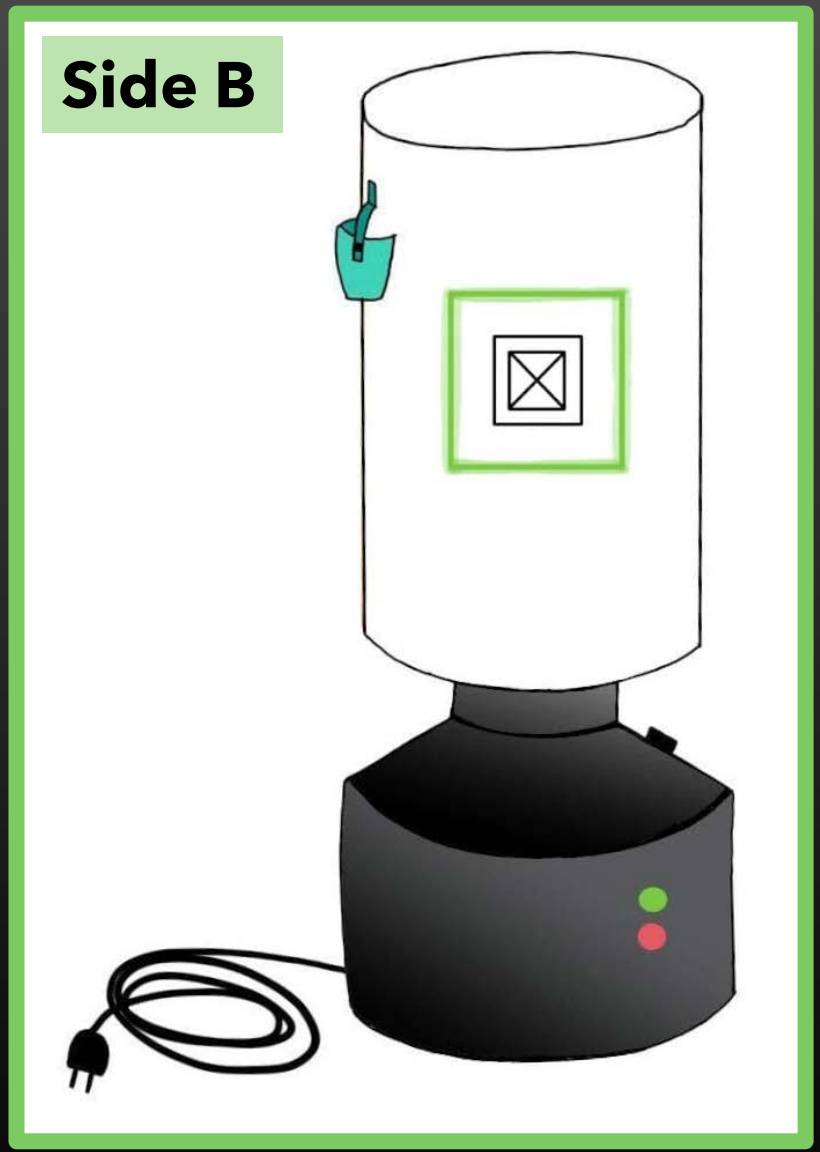
Design Concept

Overall Design



Base Design Concept Sketches



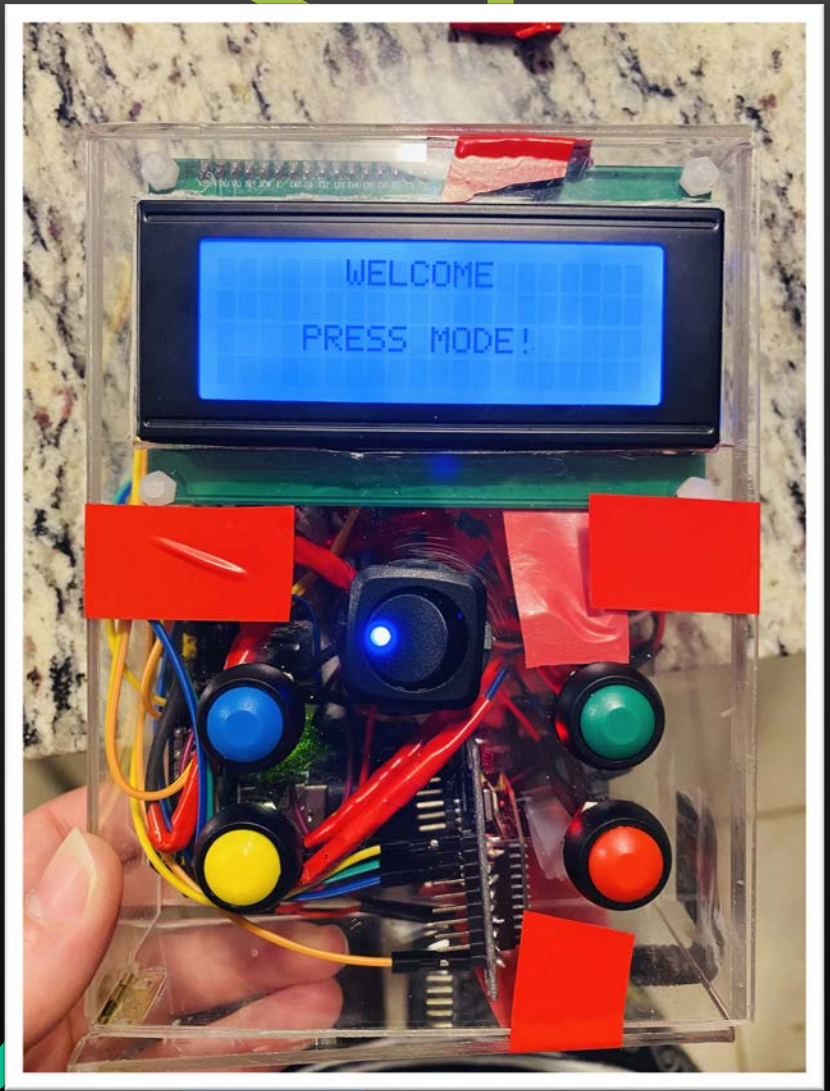


Physical Bag Design

Side B

Side A

Remote (UI) Design Concept Sketch



Physical Remote (UI) Design

Device Modes

	Cardio Mode	Reaction Time Mode	Accuracy Mode	Combination Generator Mode	Ambient Mode
Side Used	Side B	Side B	Side B	Side A	Both
User's Goal	Hit target as many times as possible within specified time period	Hit target ASAP once indicator goes off	Hit target as close to center marking as possible	Hit designated target of grid within specified time period	Enjoy the LED display ☺
Difficulty Options	Change duration of session	Change consistency of hit prompt timing	Change threshold of "accurate" hit	Change duration and consistency of mode timing	N/A
Stats Reported to User	Total hits Session time Avg. time per hit Hits per second	# of hits landed Shortest time Longest time Avg. time per hit	M/L accurate hit L/S hit time Avg. accuracy Avg. time per hit	Total hits Total possible hits Hit success ratio	N/A

Components and Hardware Implementation

Sensor Specifications

Sensors must be able to:

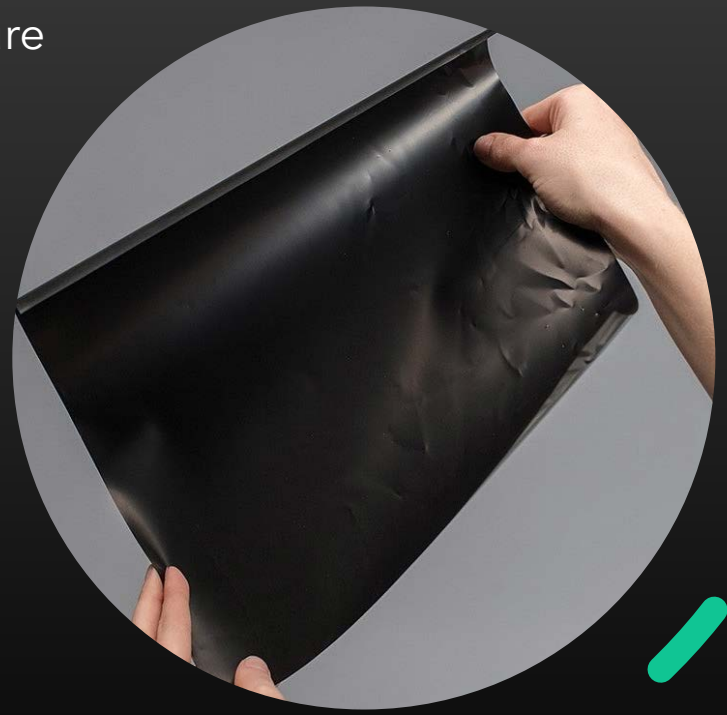
1. Withstand the force of many punches and kicks.
2. Detect rapid hits in multiple locations.
3. Count sequential hits in the same area.
4. Detect location of hits relative to designated target.
5. Consume minimal amount of power.
6. Cover entire sensing area with no more than four units.

Textile Pressure Sensor Materials

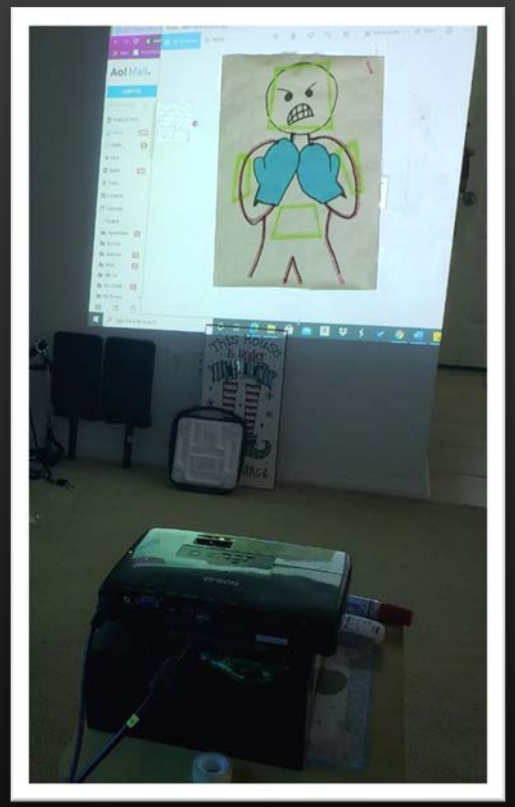
- Velostat: Pressure-Sensitive Conductive Sheet (Adafruit | 4.95$)
  - Sheet Dimensions: 11" x 11" (28cm x 28cm) before shaping
  - Surface Resistivity : < 31,000 ohms/sq.cm, changes with pressure
  - Temperature Limits : -45°C to 65°C
- Shapeable Foam Sheet (Joann's | 12.99$)
- Conductive Thread (Stainless Steel)

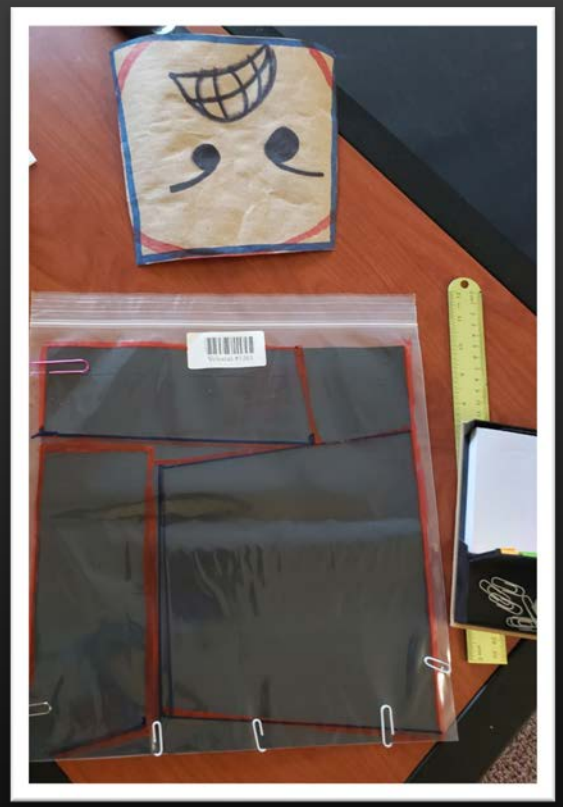
(Adafruit | 6.95$)

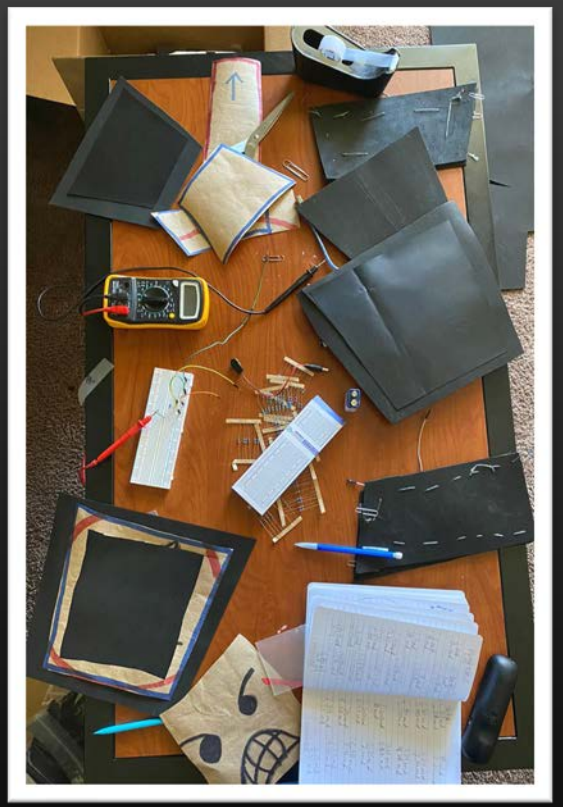
- Conductive Fabric (Adafruit | 4.95$)
- Standard Needle and Thread
- Paper Clips
- Soldering Materials
- Stencils for Sensor Shapes



Textile Pressure Sensor Construction







Gathering and Using Sensor Data

- Resting vs Being Pressed
  - 5 quick hits
- Measuring 220Ω resistor in series with sensor
- Threshold is generally about 0.65 V

Indicator: Addressable LED Strip

Requirements for Indicators:

- Multicolor Programmable Units
- Cut and Connect in Various Shapes
- Durable and Water Resistant
- Cost-Effective

WS2811 Pixel Lights:

- 50 LEDs, 3.5m / unit
- IP68 Waterproof Rated
  - suitable for outside use
- **Seller:** WESIRI (via Amazon)
- **Price:** $18.99 / unit

Prominent LED Testing Considerations:

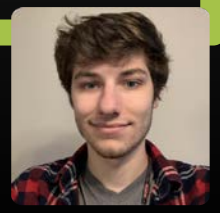
- **Power LEDs sufficiently** so they will be stable and respond accurately
- Avoid interference with LED data signal wire

Indicator: Testing

- Power Supply (Voltage Testing)
- Current Testing
- Data Lines
- Simple Code
  - FastLED Library functions
  - RGB Testing

Power Generation

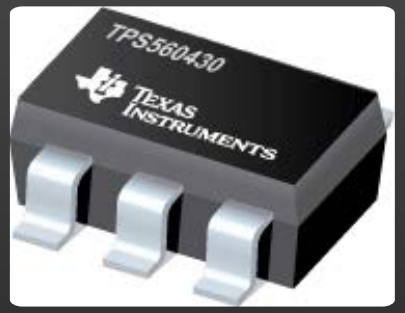
- **Base System (Amazon| 32.88$)**
  - 120 V 60 Hz AC / 12 V DC LED driver transformer.
  - 12V DC, 8.3 A Output Signal
- **UI Handheld**
  - 6V battery holder from LampVPath. (Amazon| 7.99$)
  - Holds 4 AA batteries
  - Generates a 6V DC signal.

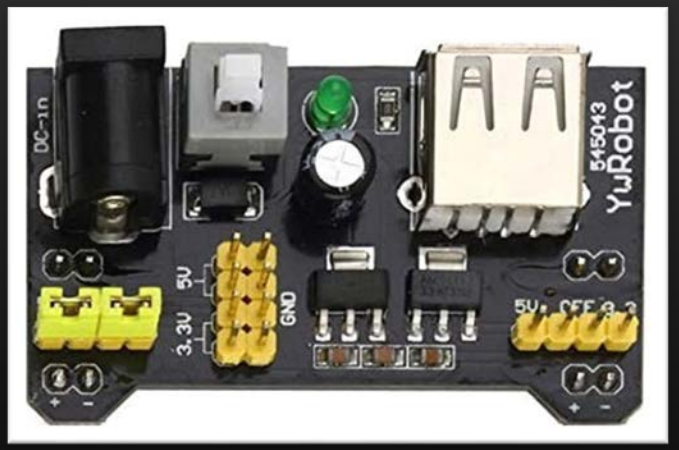


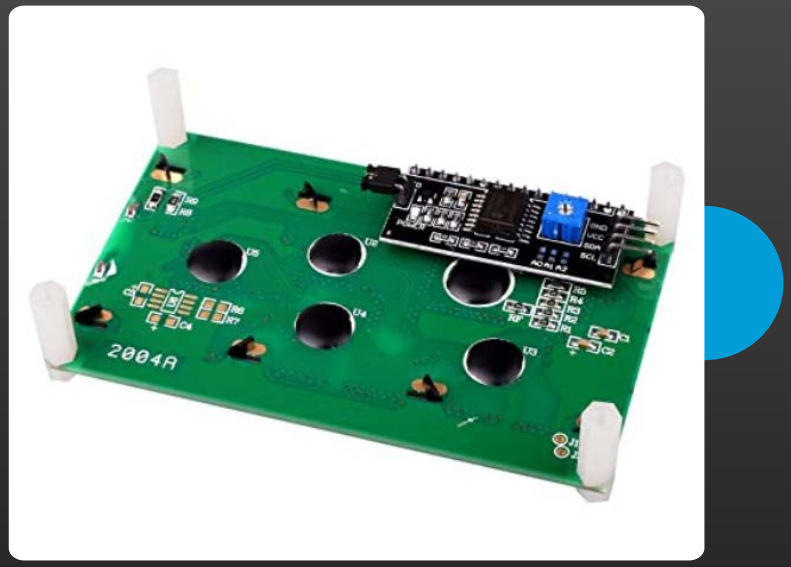
Power Conversion

- **AC/DC**
  - Included with the base LED power supply

- **DC/DC: Breadboard Power Supply**
  - Provides regulated 5V and 3.3V for all the PCB components
  - Originally designed the PCB with onboard buck DC/DC converters (TPS560430)
    - Learned too late that these provided insufficient current to the MCU and other components
    - Bucks were replaced with the breadboard power supply connected directly to the MCU in order to use the MCU design.







UI Components

1) LCD

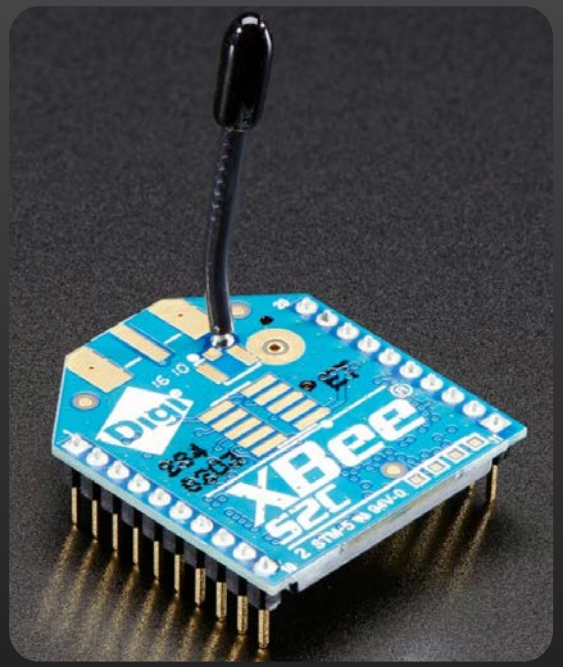
- HD44780: 20x04 display
- PCF8574T: 8-bit I/O expander for I2C bus
- 2 wire I2C serial communication
- 1 hot and 1 ground
- Operates at 5V
- Small enough to fit on remote
- Large enough to fit user results on one screen
- Price :12.15$ (Amazon)

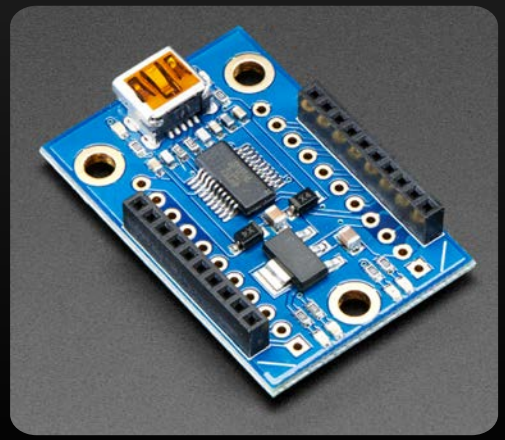
UI Components

2) XBee

- XBee S2 – chip or wire antenna
- XBee USB adapters: required for configuration
- Price: 26.95$ each Antenna (Amazon) , and 7.99$ each adapter (Digikey)

- **Pros**
  - Low power consumption (2mW while transmitting)
  - Plenty of range (indoor at 40m)
  - Sufficient data rate to keep low power (40kpbs)
- **Cons**
  - Requires manual configuration
  - Not compatible with previous generations of XBee chips





UI Components

3) User Input

- **Push Buttons:** push to close circuit, release to open circuit, waterproof, large enough for finger, small enough to fit 4 on the remote
- **On/Off slide switch:** provides stability to remain in the on position while user is interacting with base (remote will experience movement)

4) Power Supply

- **4xAA batteries:** provides 6V with 9600mAh, enough to withstand larger current drawn from LCD backlight; provides enough voltage for LCD, processor chip, and XBee antenna

MCU Selection

Standards for MCU Selection

- Have sufficient input lines for all sensors and communication necessary.
- Have sufficient output lines for all indicators and communication necessary.
- Communicate with the UI system to provide raw data.
- Require internal ROM and RAM.
- If possible, use MCU for both base and UI

Main Contenders

- FPGA
- TI MCU
- **AVR MCU**
- DSP MCU
- (Raspberry Pi)

MCU Selection Top Two

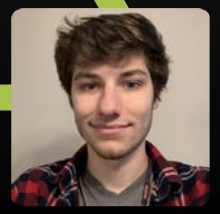
TI MCU (MSP430F6459)

- **RAM:** 66KB
- **Pins:** 74
- **I2C:** 3
- **UART:** 3
- **SPI:** 6

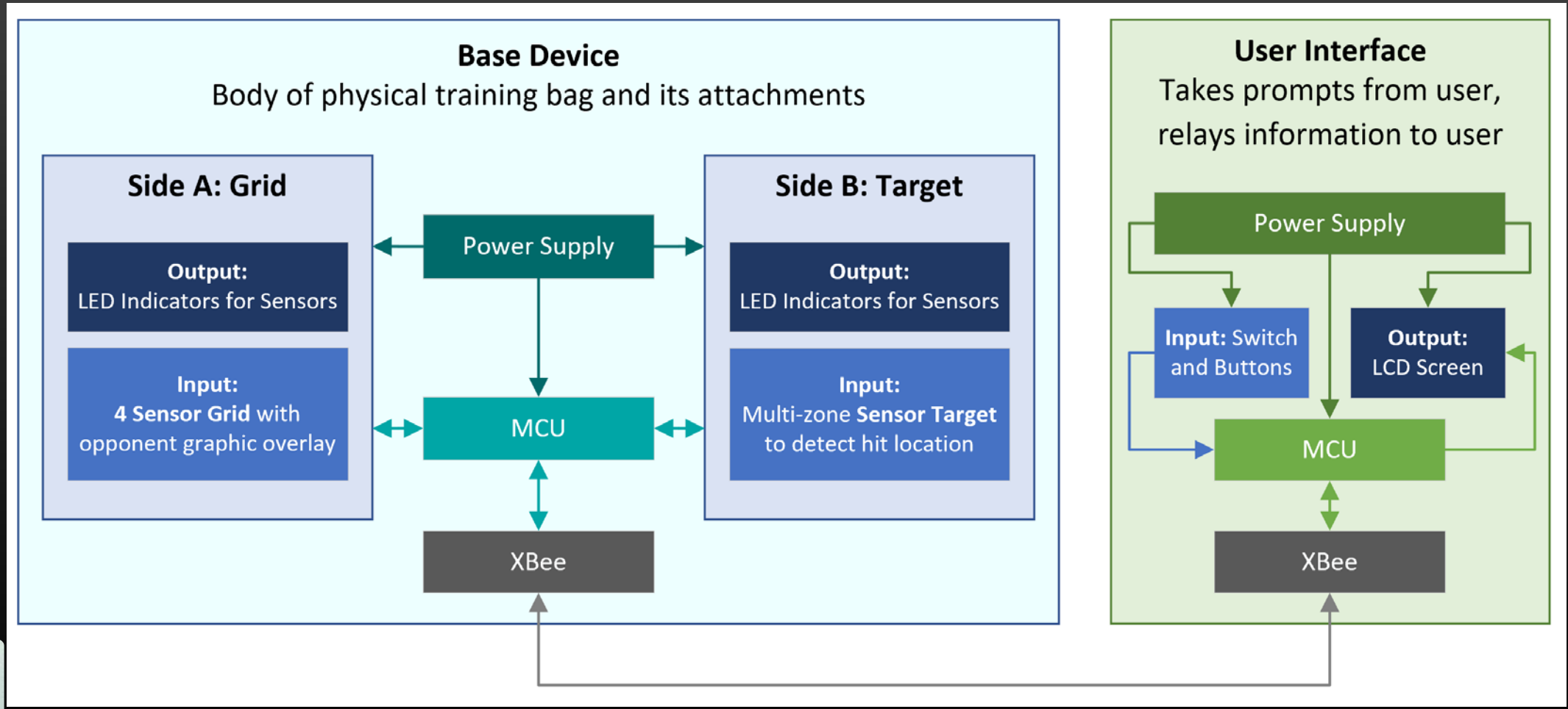
AVR MCU (ATmega2560):

- **RAM:** 64KB
- **Pins:** 100
- **I2C:** 1
- **UART:** 4
- **SPI:** 5
- **Timers:** 6
- **Input voltage:** 1.8V to 5.5V

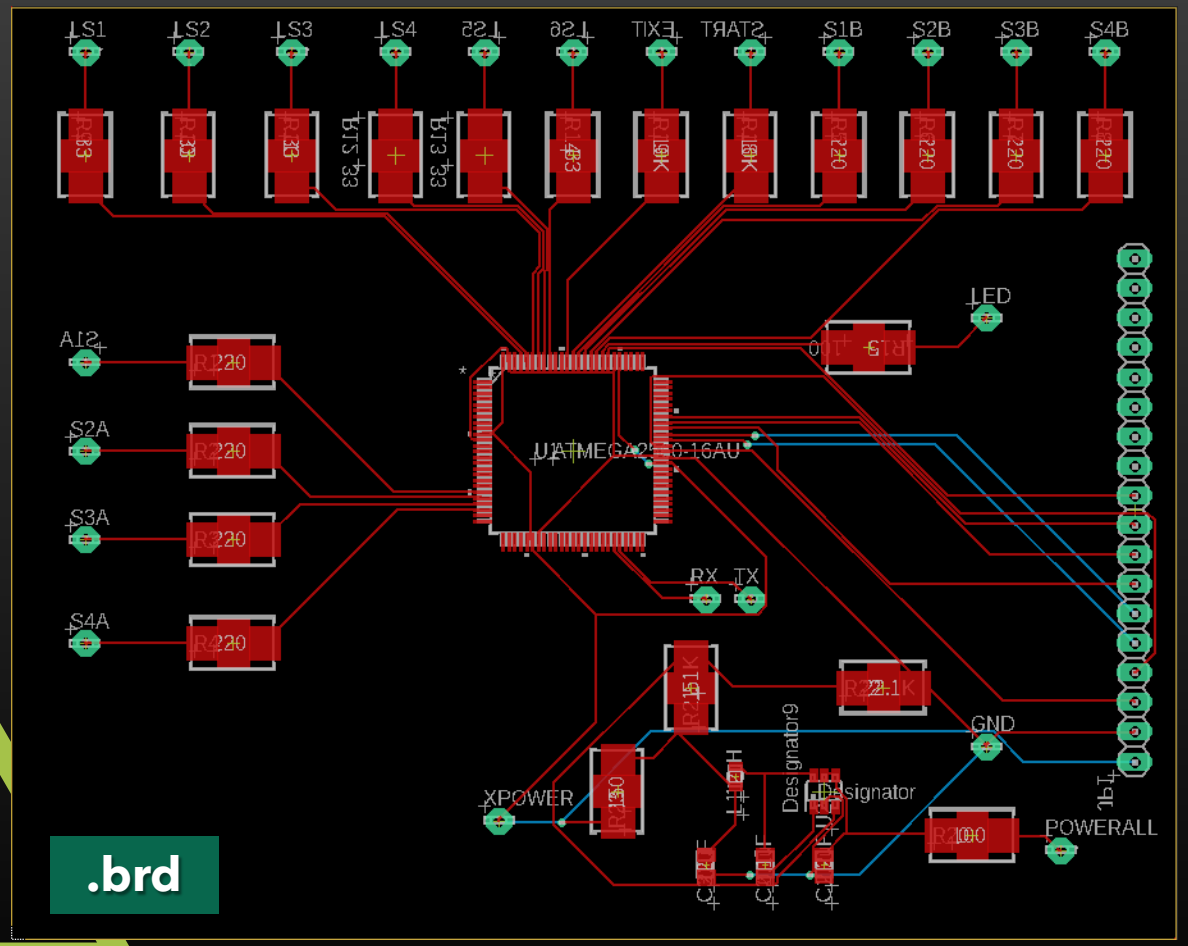
Supports function libraries that facilitate the software coding!

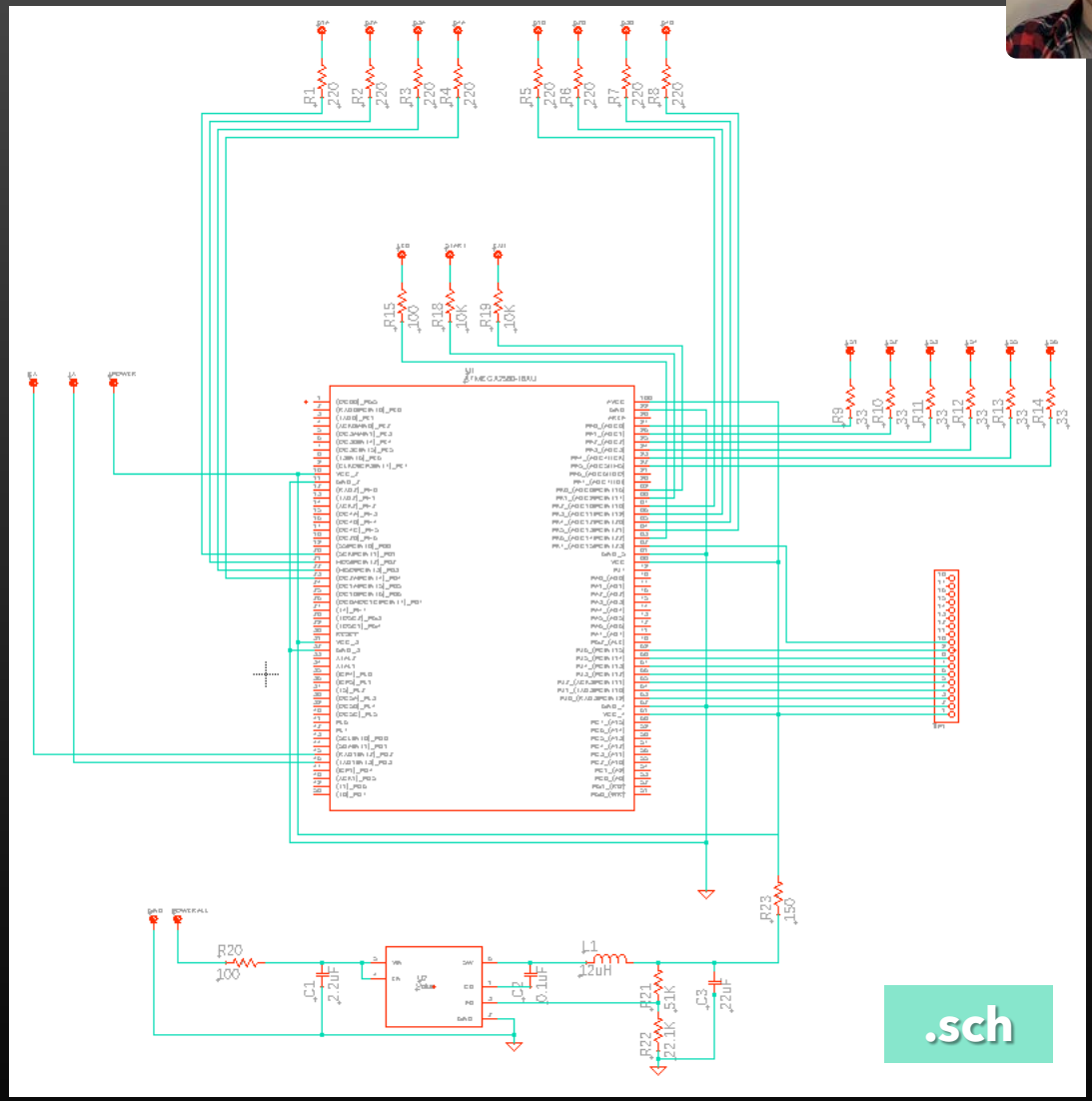


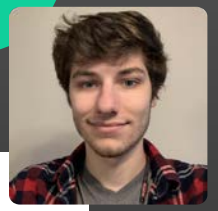
Hardware Connection Block Diagram



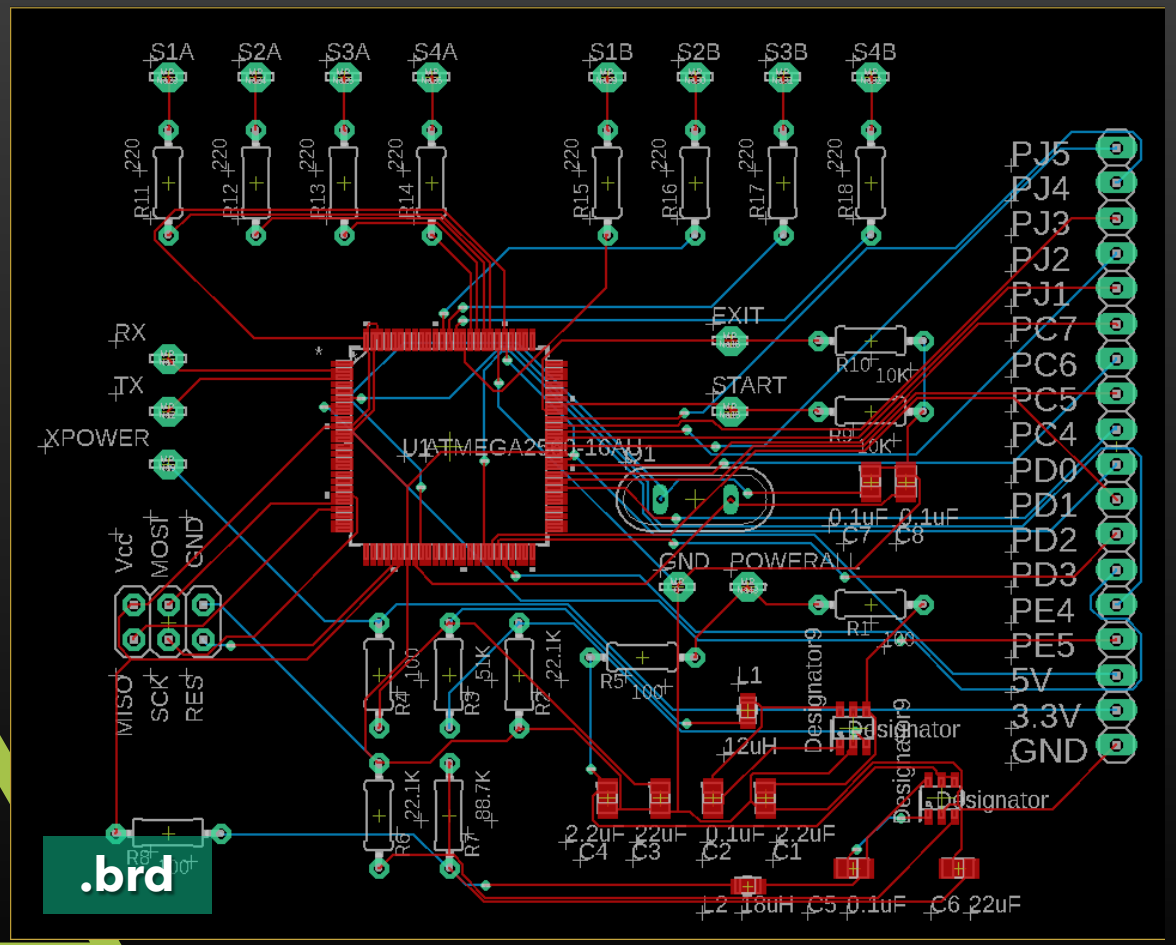
Base PCB Version 1



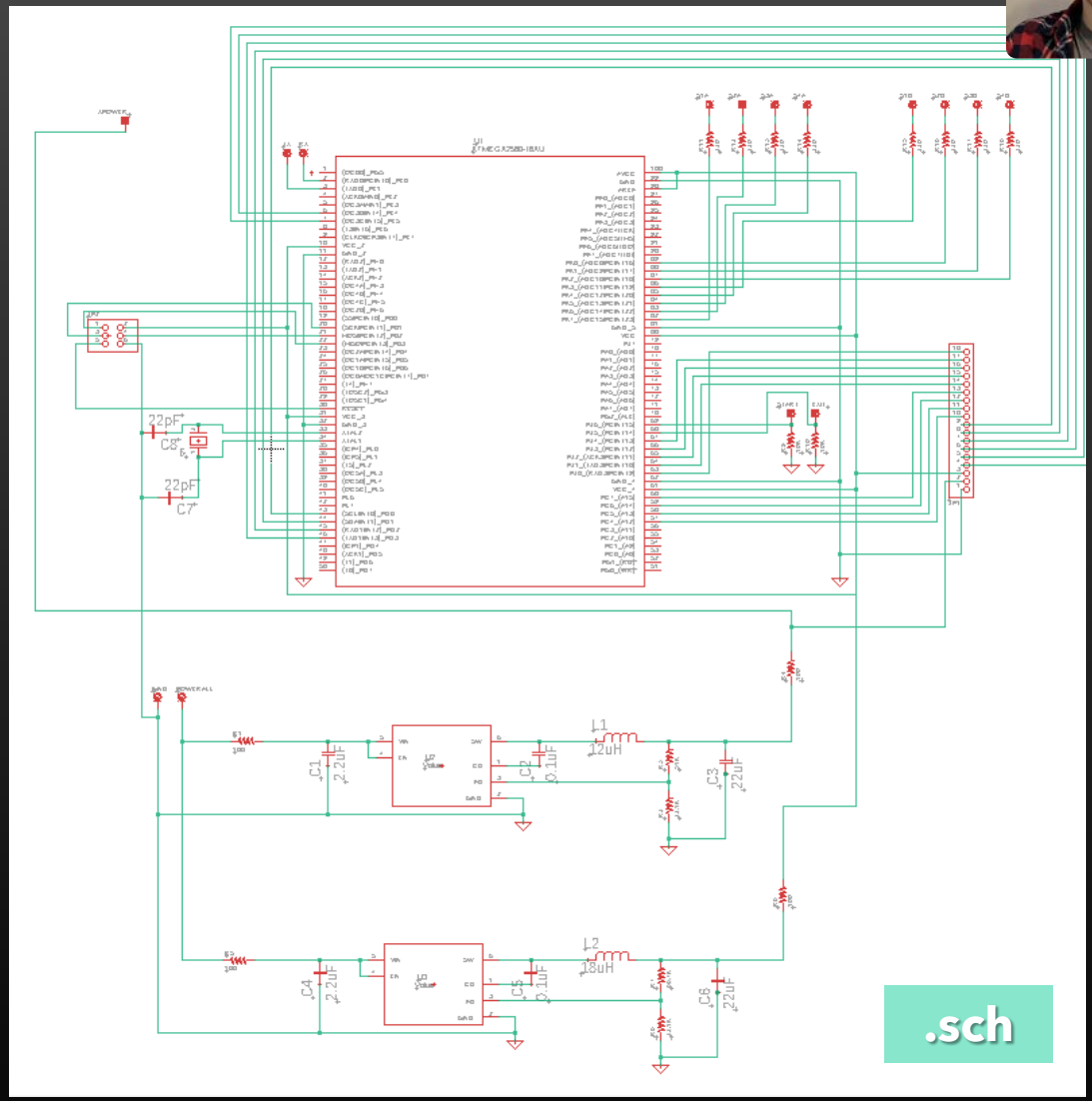




Base PCB Version 2

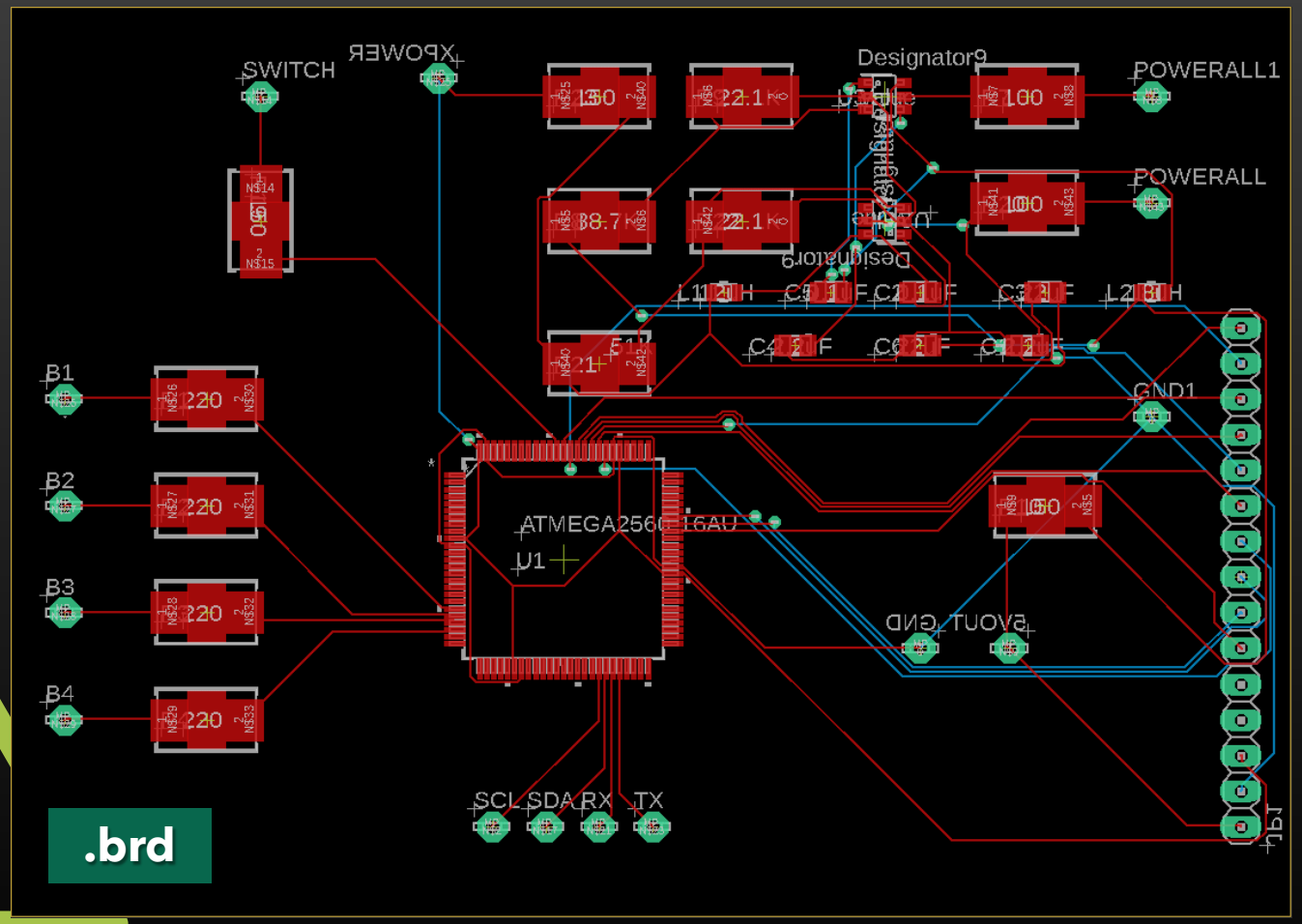


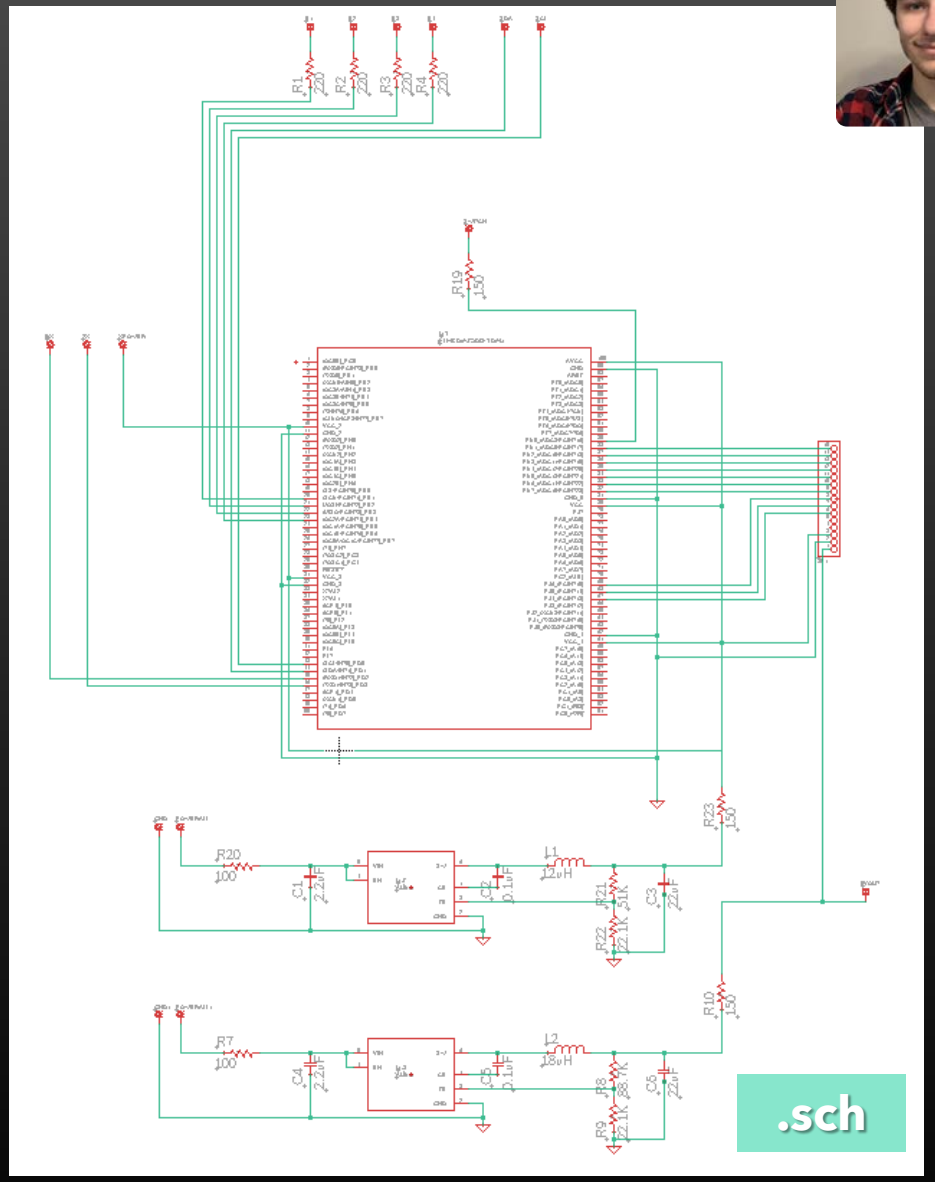
.brd

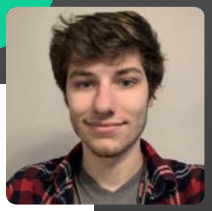


.sch

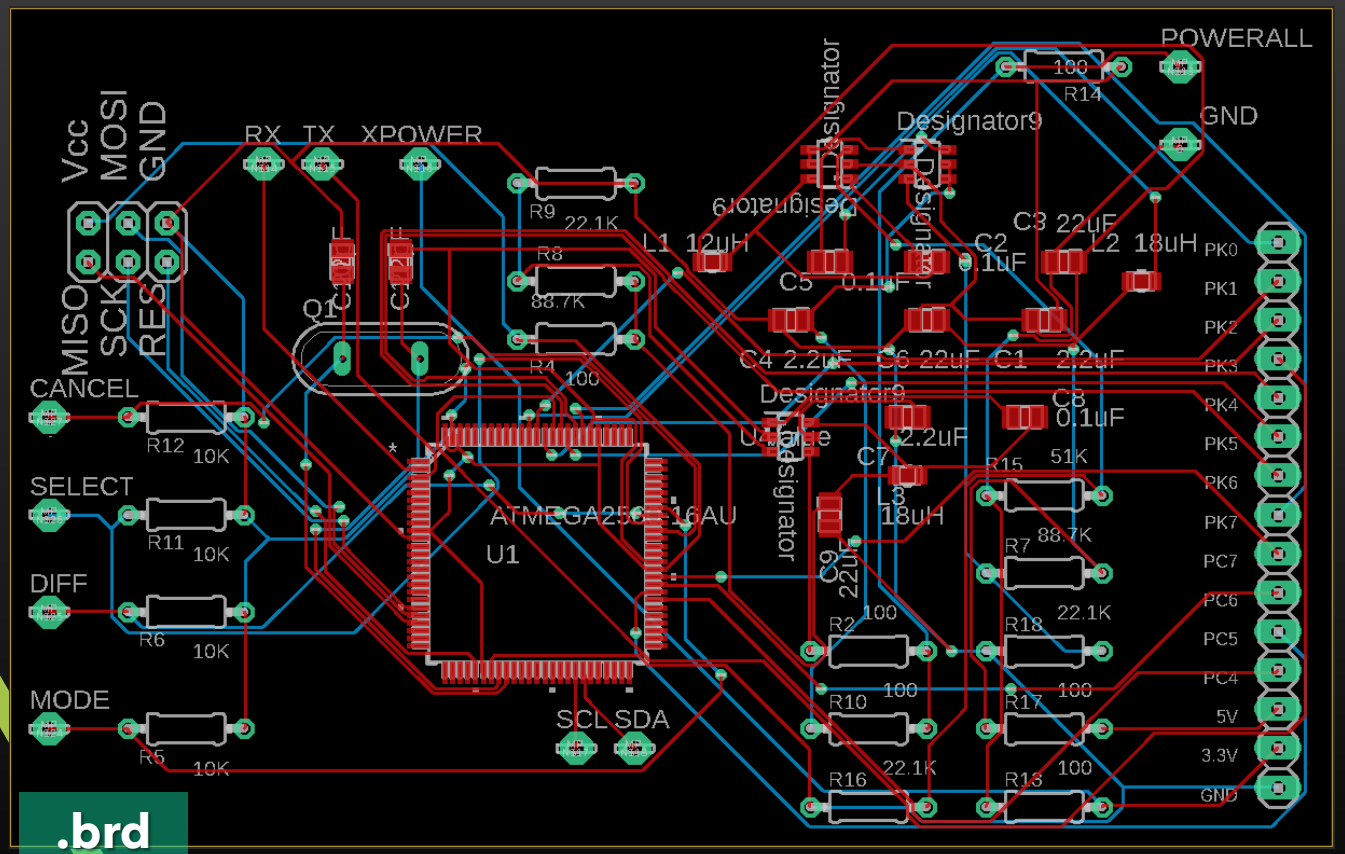
UI Remote PCB Version 1



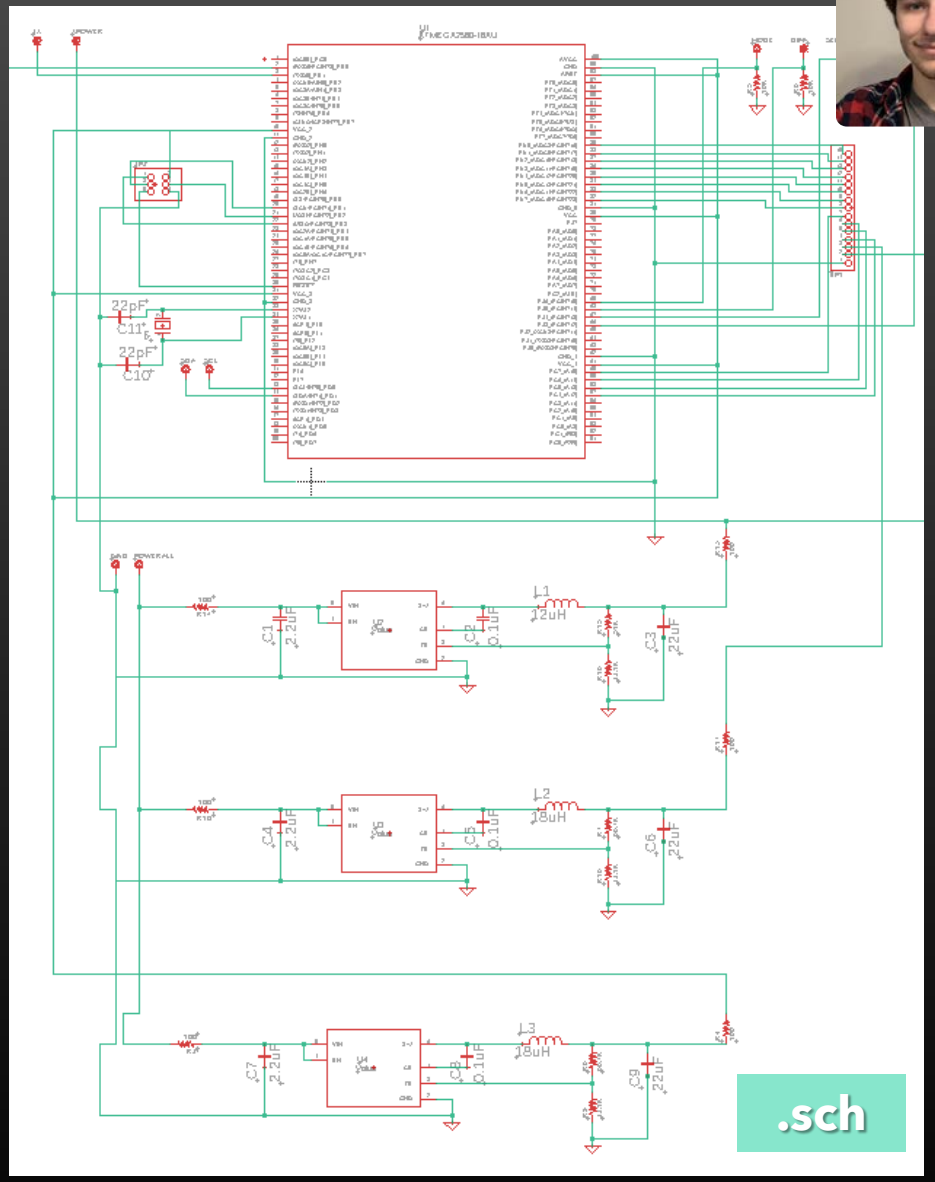




UI Remote PCB Version 2

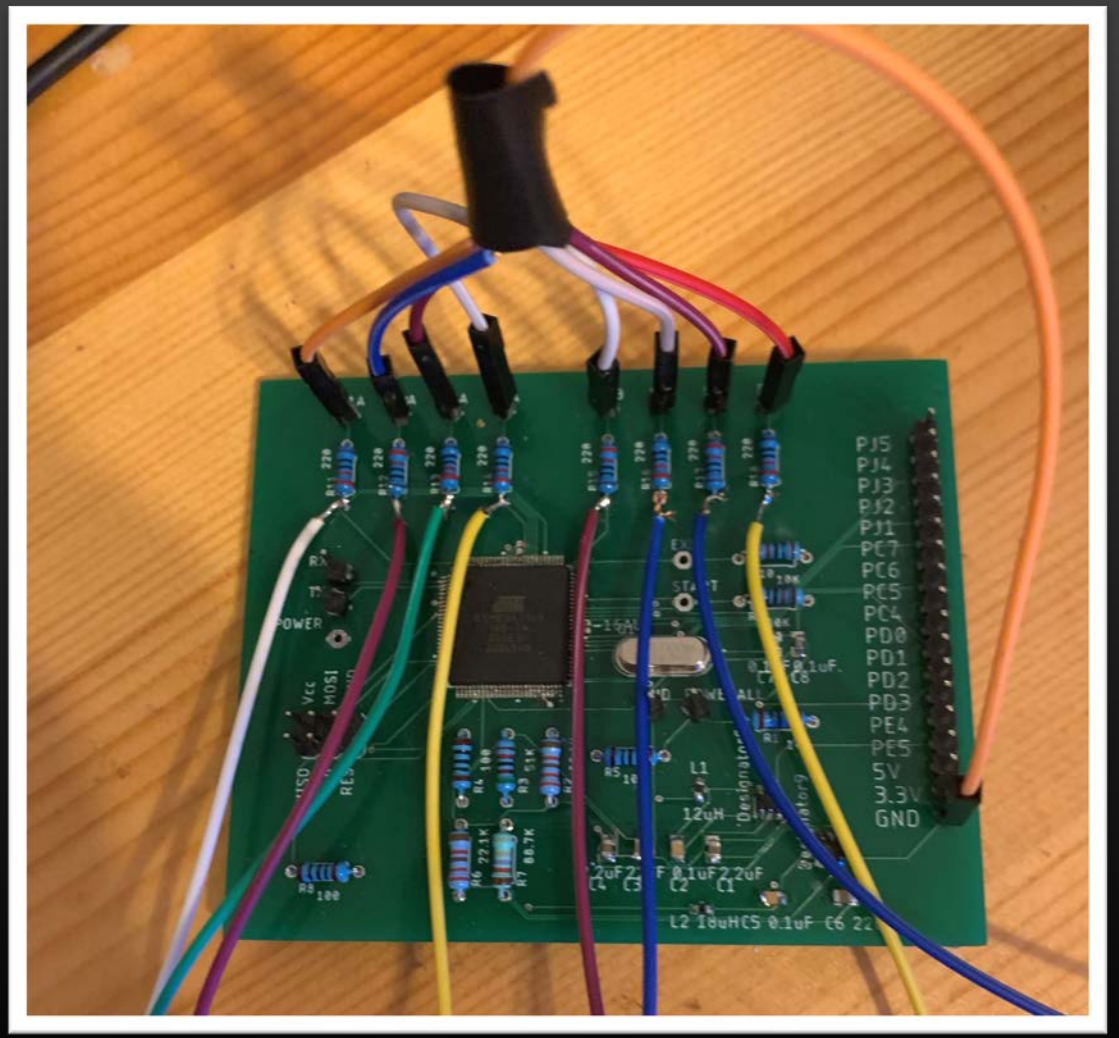


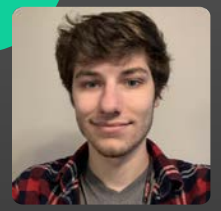
.brd



.sch

Bag PCB Rework





PCB Changes

Bag PCB

- Changed surface mount resistors to through hole
- Added an ISP header to the board
- Added labels to all pins
- Added 16 MHz oscillator and associated filter capacitors
- Added one more buck converter (ultimately unused)

UI PCB

- Changed surface mount resistors to through hole
- Added an ISP header to the board
- Added labels to all pins
- Added 16 MHz oscillator and associated filter capacitors
- Added one more buck converter (ultimately unused)

Software Design

Interconnected web of interrupt service routines to handle actions.

Software Design Approach

- **User Interface System**
  - Heavily dependent on user input through buttons
  - Relies on button ISR's to handle actions:
    - Enable/Disable specific button interrupts
    - Reset Timer values
    - Display instructions to User
    - Display Mode and Difficulty selection changes
    - Display correct screen when Cancel button is pressed
    - Trigger UART TX interrupt
  - Relies on UART RX ISR to display results at the end of the session
  - Relies on Timer ISR to prevent getting "stuck"

Software Design Approach

- **Base System**
  - Heavily dependent on pressure sensor ISRs and timer ISRs to handle actions:
    - Keeping score
    - Continue through the session, even without user interaction
    - Enable appropriate indicators for the mode of the session
    - End the session
  - Relies on UART TX ISR to send data back to the UI system

Software Testing

- External Devices
- Individual code is prepared to test and
- Integrate each external device.
  - LCD
  - XBee
  - LEDs
  - Sensors

- Libraries for devices
  - #include <Wire.h>
  - #include <LiquidCrystal_I2C.h>
  - #include <SoftwareSerial.h>
  - #include <FastLED.h>

```
while(count2<1){
  if(XBee.available()){
  val[count] = (char)XBee.read();
  if(val[count] == 47){
    count2++;
  }
  count++;
```

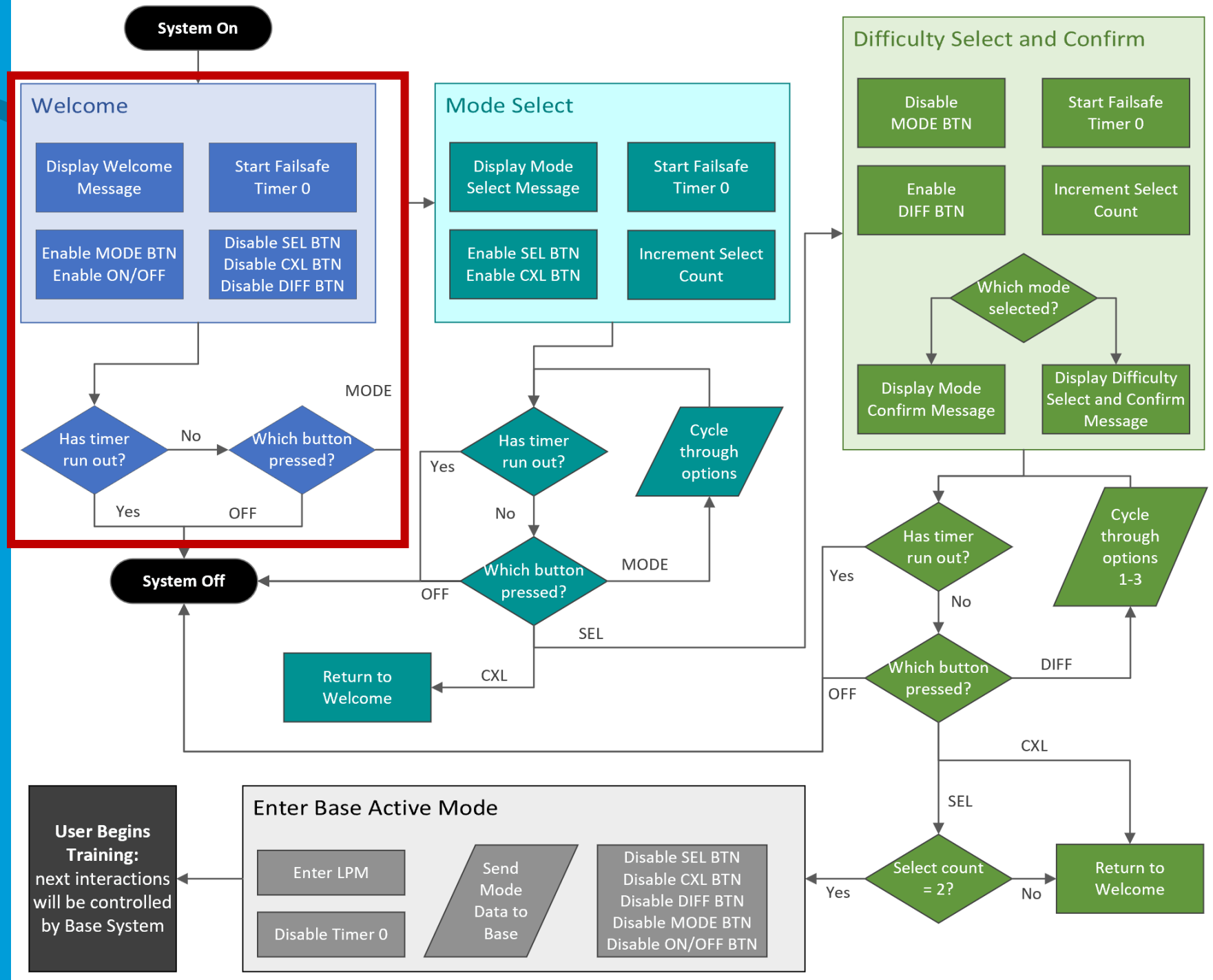
```
sensorVoltage1 = analogRead(A11);
voltage1 = sensorVoltage1 * (5 / 1024.0);

while (voltage1 > thresh_1_1 ) {

  for (int i = 0; i < NUM_LEDS_sideb ; i++) {
    leds_sideb[i] = CRGB(0, 0, 0);
  }
  FastLED.show();
  delay(15);
  sensorVoltage1 = analogRead(A11);
  voltage1 = sensorVoltage1 * (5 / 1024.0);

  while ( (voltage1 < thresh_1_2) && (voltage1 > 0) ) {

    delay(15);
    sensorVoltage1 = analogRead(A11);
    voltage1 = sensorVoltage1 * (5 / 1024.0);

    if ( (voltage1 < thresh_1_3) && (voltage1 > 0 ) ) {

      result1++;
```

```
sensorVoltage1 = analogRead(A11);
voltage1 = sensorVoltage1 * (5 / 1024.0);

while (voltage1 > thresh_1_1 ) {

  for (int i = 0; i < NUM_LEDS_sideb ; i++) {
    leds_sideb[i] = CRGB(0, 0, 0);
  }
  FastLED.show();
  delay(15);
  sensorVoltage1 = analogRead(A11);
  voltage1 = sensorVoltage1 * (5 / 1024.0);

  while ( (voltage1 < thresh_1_2) && (voltage1 > 0) ) {

    delay(15);
    sensorVoltage1 = analogRead(A11);
    voltage1 = sensorVoltage1 * (5 / 1024.0);

    if ( (voltage1 < thresh_1_3) && (voltage1 > 0 ) ) {

      result1++;
```

```
for (int j = 0; j < 5; j++) {
  for (int i = 0; i < NUM_LEDS_top; i++) {
    leds_top[i] = CRGB(red_arr[j], green_arr[j], blue_arr[j]);
  }
  for (int i = 0; i < NUM_LEDS_btm; i++) {
    leds_btm[i] = CRGB(red_arr[j], green_arr[j], blue_arr[j]);
  }
  for (int i = 0; i < NUM_LEDS_right; i++) {
    leds_right[i] = CRGB(red_arr[j], green_arr[j], blue_arr[j]);
  }
  for (int i = 0; i < NUM_LEDS_left; i++) {
    leds_left[i] = CRGB(red_arr[j], green_arr[j], blue_arr[j]);
  }
  for (int i = 0; i < NUM_LEDS_sideb; i++) {
    leds_sideb[i] = CRGB(red_arr[j], green_arr[j], blue_arr[j]);
  }

  FastLED.show();
  delay(500);
}
```

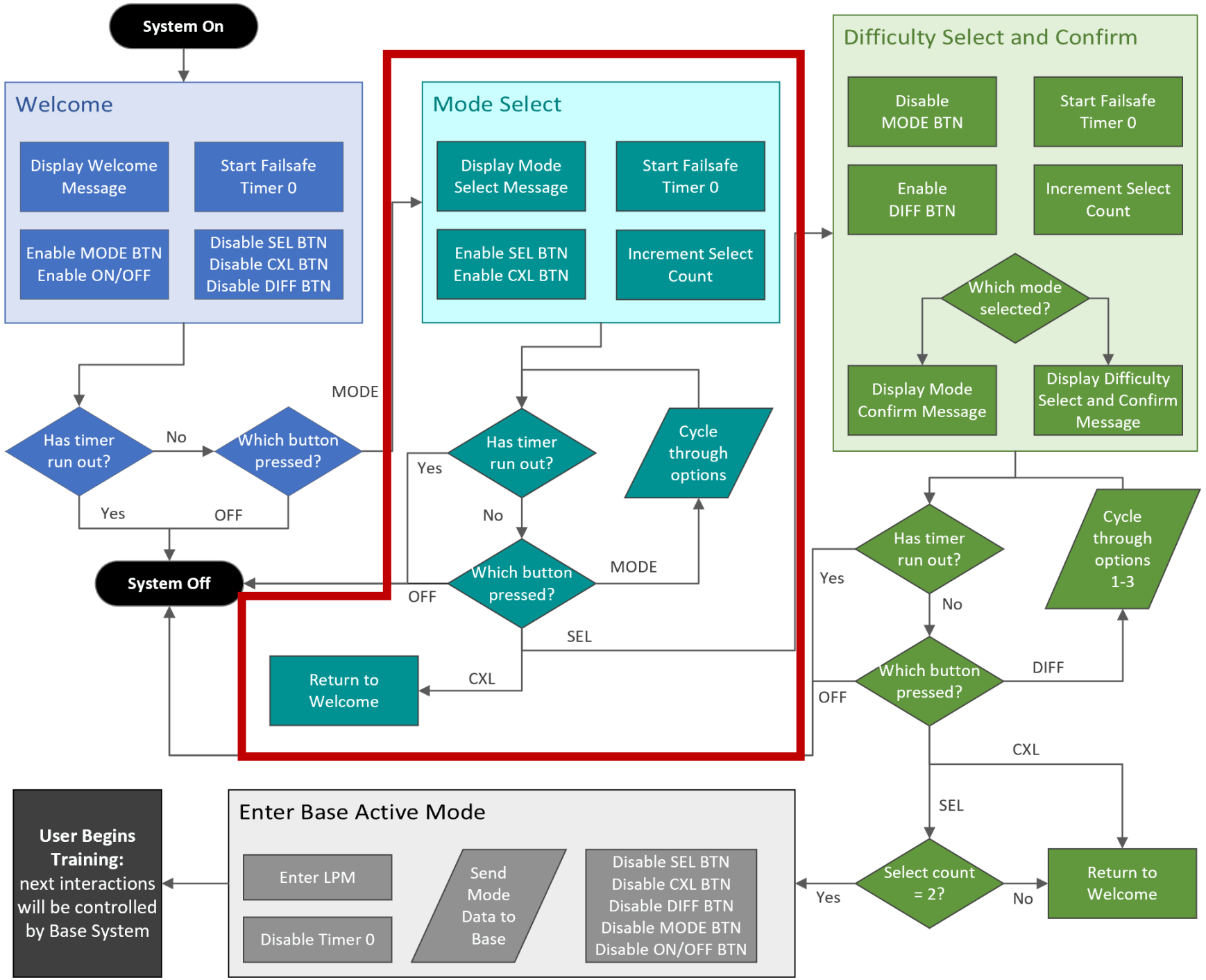
UI System Software Flowchart

Button/Switch Names:
ON/OFF = On/Off (Toggle Switch)
MODE BTN = Mode
DIFF BTN = Difficulty
SEL BTN = Select
CXL BTN = Cancel



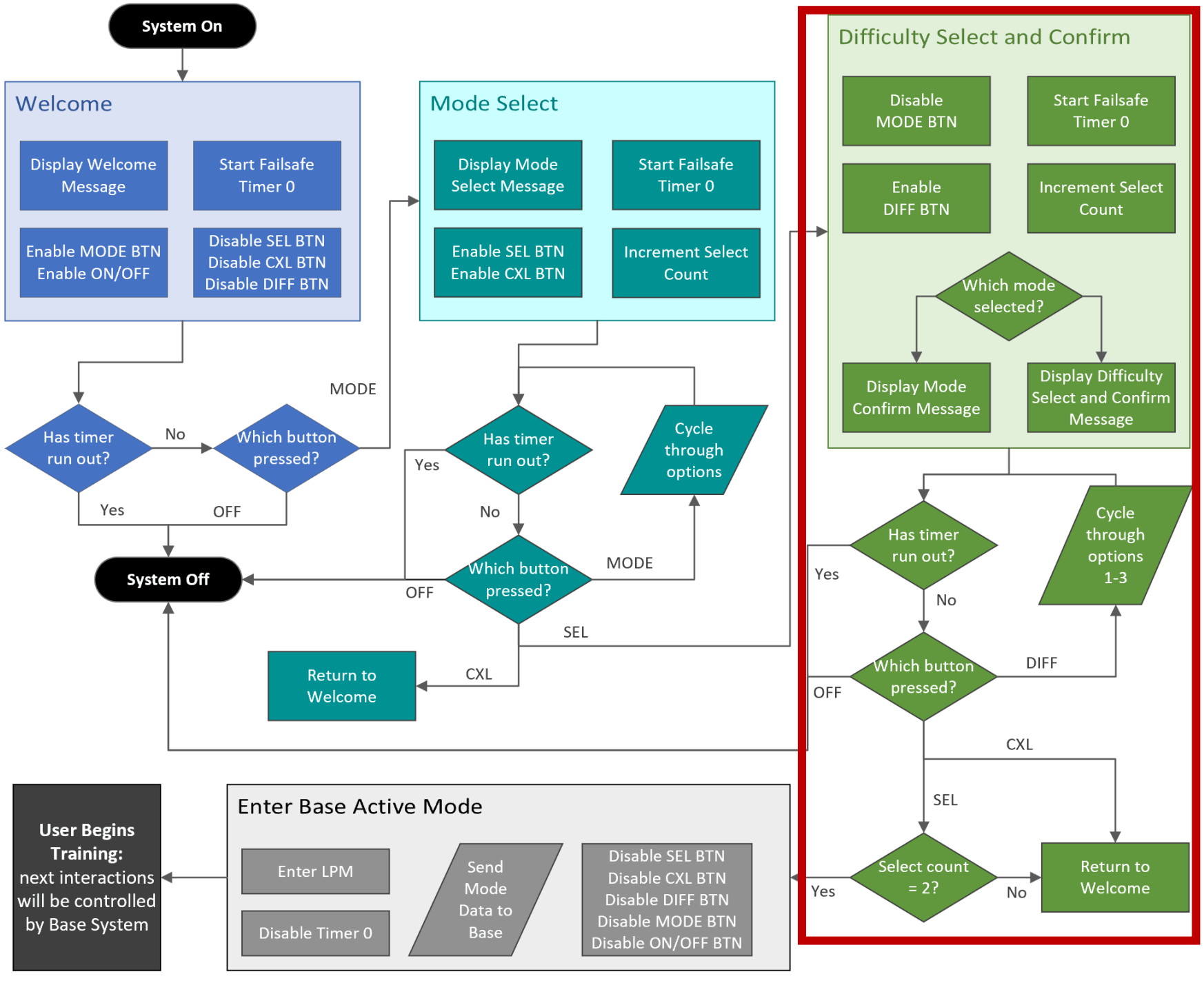
UI System Software Flowchart

Button/Switch Names:
ON/OFF = On/Off (Toggle Switch)
MODE BTN = Mode
DIFF BTN = Difficulty
SEL BTN = Select
CXL BTN = Cancel



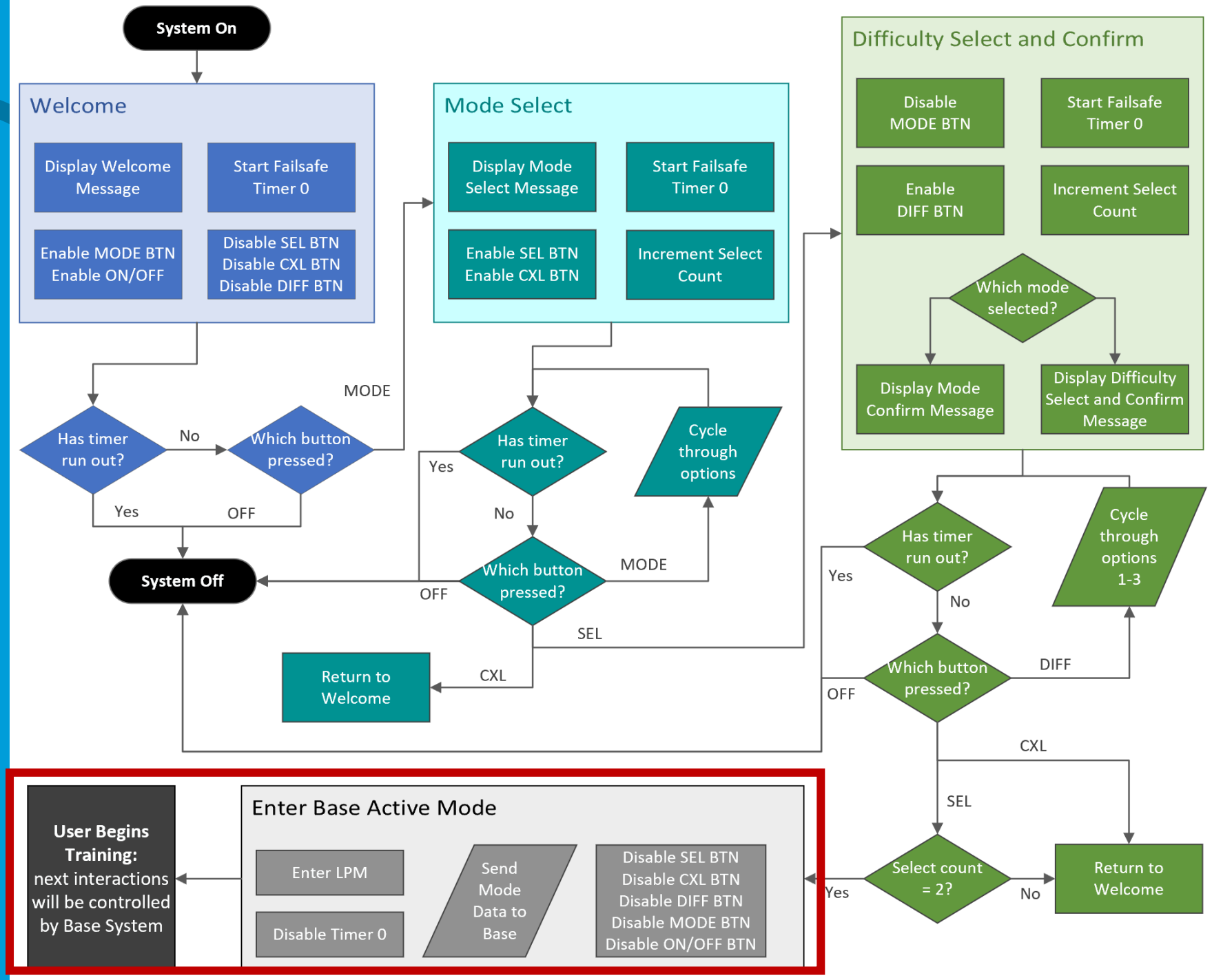
UI System Software Flowchart

Button/Switch Names:
ON/OFF = On/Off (Toggle Switch)
MODE BTN = Mode
DIFF BTN = Difficulty
SEL BTN = Select
CXL BTN = Cancel

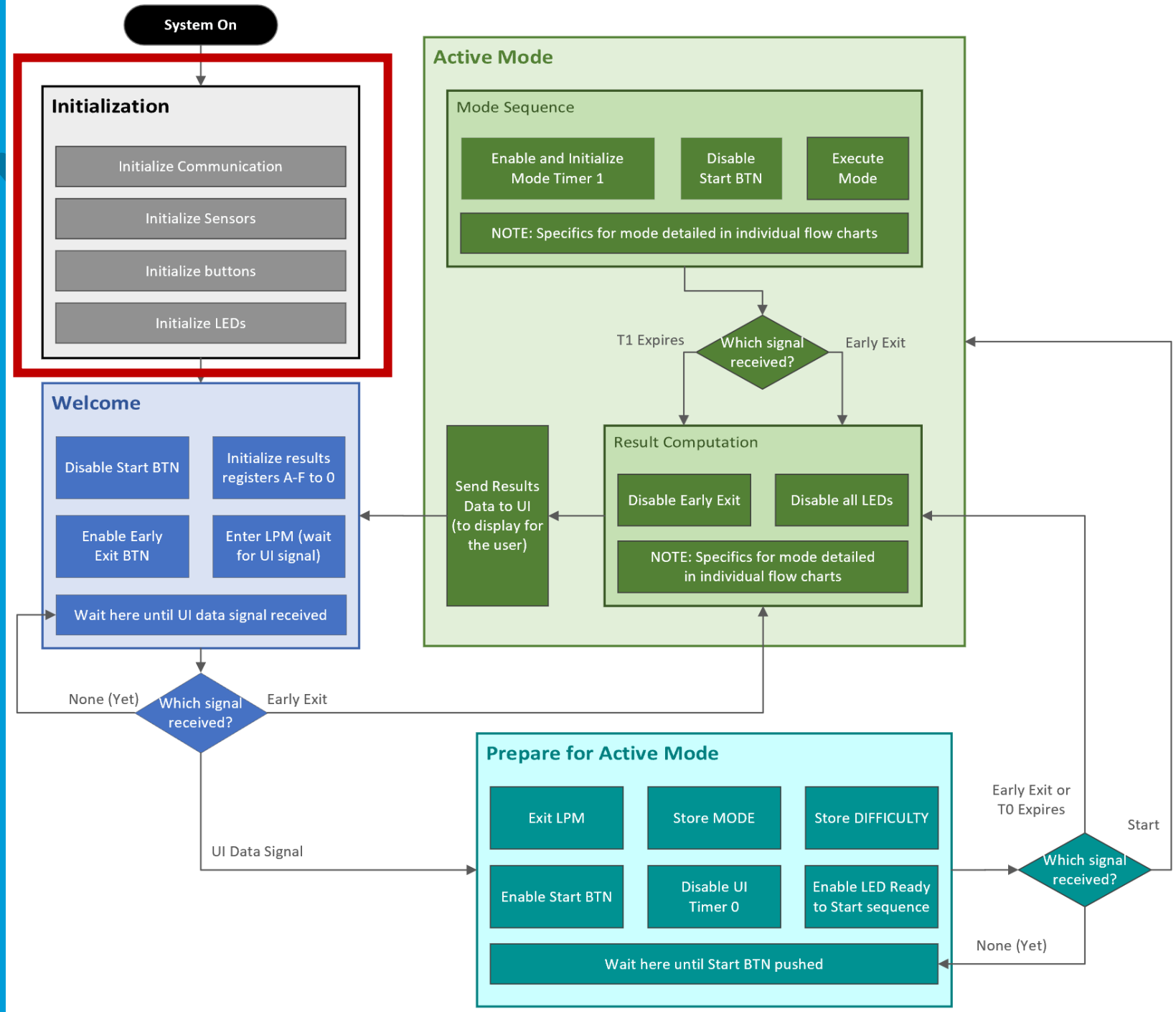


UI System Software Flowchart

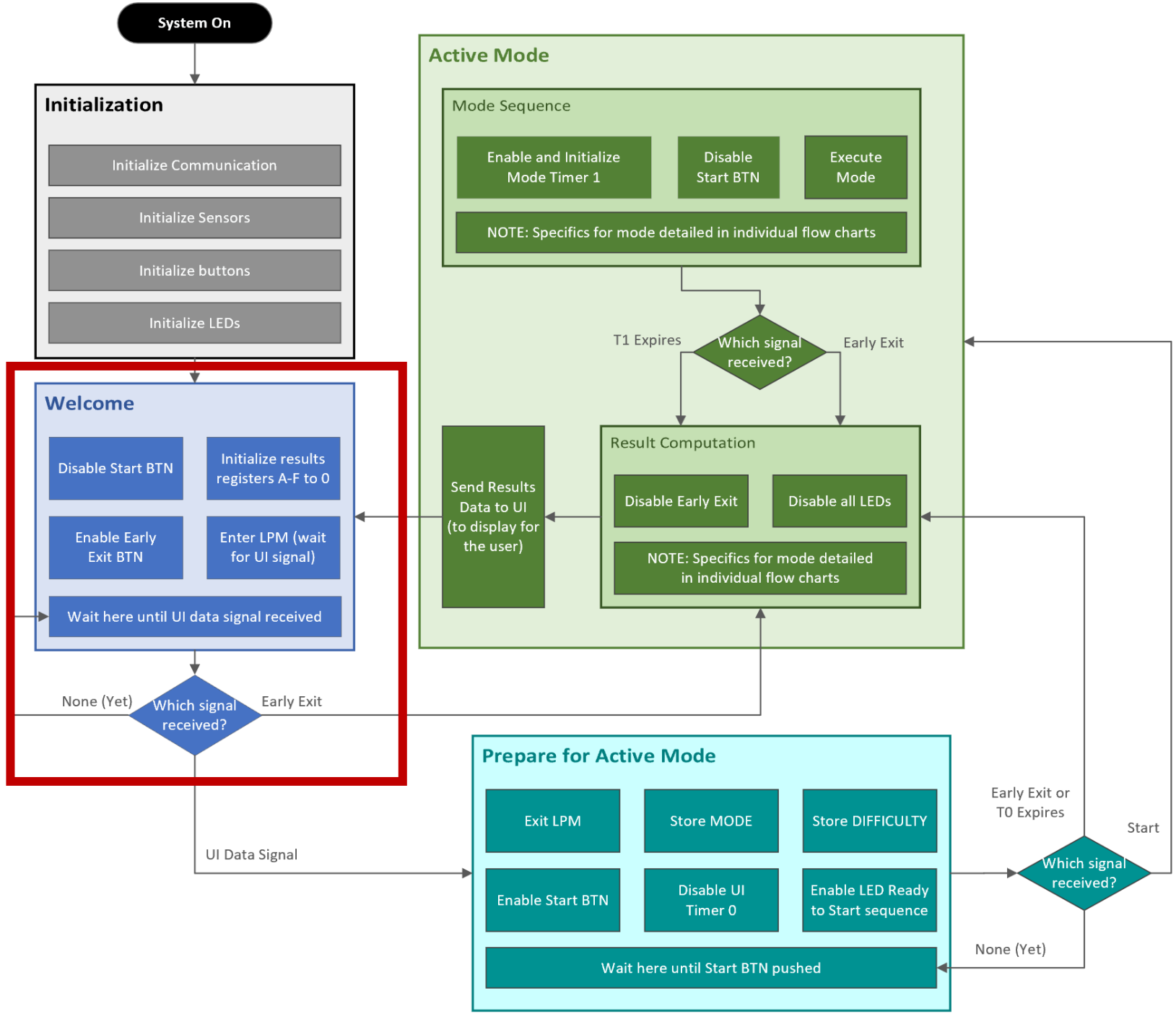
Button/Switch Names:
ON/OFF = On/Off (Toggle Switch)
MODE BTN = Mode
DIFF BTN = Difficulty
SEL BTN = Select
CXL BTN = Cancel



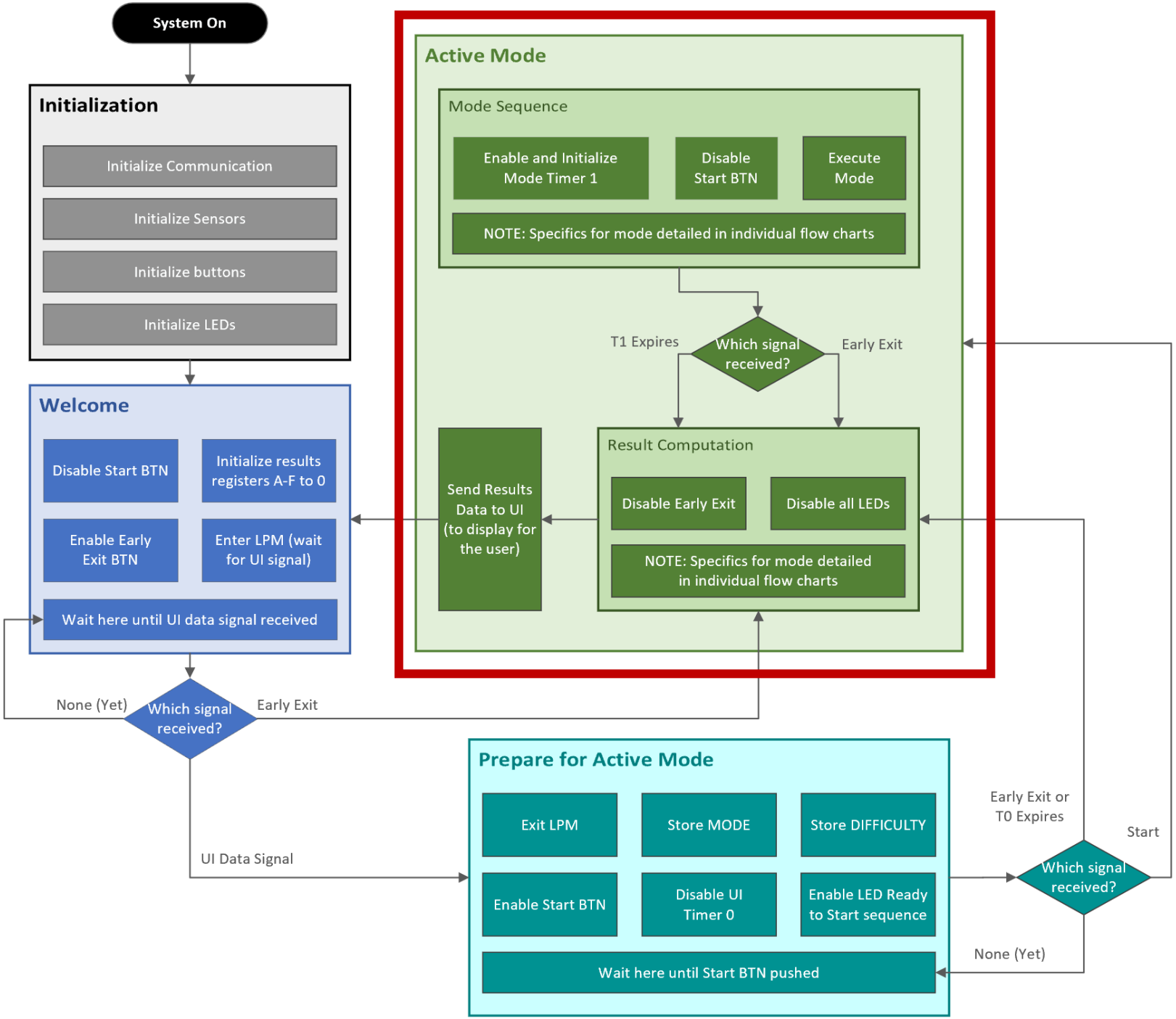
Base System Software Flowchart



Base System Software Flowchart



Base System Software Flowchart



Base System Software Flowchart

System On

Initialization
- Initialize Communication
- Initialize Sensors
- Initialize buttons
- Initialize LEDs

Welcome
- Disable Start BTN
- Initialize results registers A-F to 0
- Enable Early Exit BTN
- Enter LPM (wait for UI signal)
- Wait here until UI data signal received

Which signal received?
- None (Yet)
- Early Exit
- UI Data Signal

Active Mode

Mode Sequence
- Enable and Initialize Mode Timer 1
- Disable Start BTN
- Execute Mode
- NOTE: Specifics for mode detailed in individual flow charts

Which signal received?
- T1 Expires
- Early Exit

Result Computation
- Disable Early Exit
- Disable all LEDs
- NOTE: Specifics for mode detailed in individual flow charts

Send Results Data to UI (to display for the user)

Prepare for Active Mode
- Exit LPM
- Store MODE
- Store DIFFICULTY
- Enable Start BTN
- Disable UI Timer 0
- Enable LED Ready to Start sequence
- Wait here until Start BTN pushed

Which signal received?
- Early Exit or T0 Expires
- Start
- None (Yet)

Administrative Content

Budget and Financing

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

Component	Price (USD)	Quantity	Total (USD)
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

Work Distribution

GREEN (1): Lead

BLUE (2): Primary Assistant(s)

Task	Hannah	Joseph	Nicole	Natesha
Design	2		2	1
Sensors			1	2
UI	1			
Power		1		
Indicators				1
Software	1			2
MCU		1		
PCB	2	1		
Sewing and Crafting			1	2
Video Editing		1	2	

Questions?

Group 22 appreciates your time!