System Verification Final Documentation

Bike Team 3
by Farhiya Osman, Elizabeth Wade and Oluwaseun Samuel Popoola
1 General Overview:

The goal of this project was to create a system that creates an automatic bike light. This system controls LEDs based on inputs from a light sensor, hardwired buttons and an accelerometer. Based on different changes LED patterns are displayed.

2 Customer and Engineering Requirements:

The system should be robust.

• Engineering Requirement: The system should maintain all functionality with no interruptions after dropping 3 feet onto pavement.

The system should be safe.

• Engineering Requirement: The system must use MC4 (or similar weatherproof) connectors, have a disconnect switch, and not have any exposed conductors. Wires must be organized in split loom or other protective materials. All devices must be rated at least IP64 (https://en.wikipedia.org/wiki/IP_Code).

The system turn signals should be automatic.

• Engineering Requirement: The system turn signals should turn off within 15 feet of completing a turn.

The system brake lights should be automatic.

• Engineering Requirement: The system brake lights should linearly adjust to maximum brightness and flashing speed as the bicycle slows down until fully stopping.

The system should be visible.

• Engineering Requirement: All system lights should be visible by a driver with 20/20 vision from 40 feet away in complete darkness with a light output of at least 1000 lumens.

Extra Requirements:

• The system will have a rechargeable battery that will last for 2 hours.

• The system will automatically turn on a white light for safety when biking in the dark. It will turn off during the daylight hours.
3 Electrical Schematic:

Below is the attached KiCad schematic of our entire electrical system for our automatic bike light. We used a light sensor, accelerometer and hardwired buttons. These inputs change the LED patterns on our system.

![Image of Electrical Schematic]

Figure 3.1: Entire Electrical System Schematic

4 Code:

This is our arduino code we used to carry out all our projects functionality. Our system is only using arduino code.

```c
// Accelerometer libraries
#include <Adafruit_MPU6050.h>
#include <Adafruit_Sensor.h>
#include <Wire.h>

// LED libraries
#include <FastLED.h>
#define NUM_LEDS 50
#define BRIGHTNESS 20
#define NUM_LED_BLOCK 2
#define NUM_LED_PER_BLOCK 25
#define DATA_PIN 10
#define DATA_PIN2 7
```

// Accelerometer
Adafruit_MPU6050 mpu;
float acc_1;
float acc_2;
float acc_3;

// Button init values
int turnInd = 0;  // int to decide to what direction has been selected for the directional signal
unsigned long button_time = 0;  // for debouncing the interrupt
unsigned long last_button_time = 0;  // " " " 
const byte interruptPin1 = 2;  // pins assigned to carry out interrupt for left turn
const byte interruptPin2 = 3;  // pins assigned to carry out interrupt for right turn
volatile byte buttonState1 = 0;  // int to check state of button 1
volatile byte buttonState2 = 0;  // int to check state of button 2

// LED arrays for storing led output info
CRGB leds[NUM_LEDS];
CRGB leds2[NUM_LEDS];

void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);
  // LED setup
  FastLED.addLeds<NEOPIXEL, 7>(leds, 0, NUM_LED_PER_BLOCK);  // GRB ordering is assumed
  FastLED.addLeds<NEOPIXEL, 10>(leds2, 25, NUM_LED_PER_BLOCK);  // GRB ordering is assumed
  // light sensor setup
  pinMode(A1, INPUT);
  // Accelerometer Setup
  Serial.begin(115200);
  while (!Serial) {
    delay(10);  // will pause until serial console opens
  }
  Serial.println(" ");
  Serial.println("Adafruit MPU6050 test!");
  Serial.println("test2");
  // Try to initialize!
  if (!mpu.begin()) {
    Serial.println("Failed to find MPU6050 chip");  // if accelerometer chip is not found
    while (1) {
      delay(10);
    }
  }
  Serial.println("MPU6050 Found!");
  mpu.setAccelerometerRange(MPU6050_RANGE_8_G);  // sets the sensitivity of the accelerometer.
  Serial.println("Accelerometer range set to:");
  switch (mpu.getAccelerometerRange()) {  // shows other sensitivities available
    case MPU6050_RANGE_2_G:
      Serial.println("+-2G");
      break;
    case MPU6050_RANGE_4_G:
      Serial.println("+-4G");
      break;
    case MPU6050_RANGE_8_G:
      Serial.println("+-8G");
      break;
    case MPU6050_RANGE_16_G:
      Serial.println("+-16G");
      break;
  }
  Serial.println(" ");
  delay(100);
  // Button Interrupt Setup
pinMode(interruptPin1, INPUT_PULLUP);
attachInterrupt(digitalPinToInterrupt(interruptPin1), interrupt1, CHANGE);
//interrupts called for direction input
pinMode(interruptPin2, INPUT_PULLUP);
attachInterrupt(digitalPinToInterrupt(interruptPin2), interrupt2, CHANGE);
}

void loop() {
  /* Get new sensor events with the readings */
  // Light sensor
  int analogValue = analogRead(A1); // input pin for light sensor
  // Turn the LED on, then pause
  if (analogValue < 400) {
    // checks if the light is below 100
    Serial.println(" - Dark");
    Serial.print("Analog reading 2: ");
    Serial.println(analogValue); // if below, print dark
  }
  if (analogValue < 400) {
    // set LEDs to white
  } else if(analogValue > 400) {
    // checks if value is above 100
    // Now turn the LED off
    clearLEDs();
  }
  // Accelerometer loop check
  sensors_event_t a, g, temp;// gets variables for gyro and temp sensor, but only
  mpu.getEvent(&a, &g, &temp);// gets the data from accelerometer and writes to variable a.
  // Print the values
  // Serial.print(a.acceleration.y);
  mpu.getEvent(&a, &g, &temp);
  if(a.acceleration.y <= -1.0) { // checks if acceleration is negative
    Serial.println("Turn on one row of LEDs");
    Serial.println("Acceleration Y second event");
    Serial.println(a.acceleration.y);
    clearLEDs();
    oneLightCall(); // displays one row of lights on each LED
    buttonCheck(); // checks for button interrupt, signalling turn
  }
  mpu.getEvent(&a, &g, &temp);
  mpu.getEvent(&a, &g, &temp);
  mpu.getEvent(&a, &g, &temp);
  mpu.getEvent(&a, &g, &temp);
  mpu.getEvent(&a, &g, &temp);
  acc_1 = a.acceleration.y;
  delay(25);
  delay(25);
  delay(25);
  delay(25);
  delay(25);
  acc_2 = a.acceleration.y;
  delay(25);
  delay(25);
  delay(25);
  delay(25);
  delay(25);
  acc_3 = a.acceleration.y;
}
if(acc_1 <= -0.5 || acc_2 <= -0.5 || acc_3 <= -0.5){ // checks if acceleration is still negative
    Serial.print("Acceleration Y Third event");
    Serial.print(a.acceleration.y);
    threeLightCall(); // displays three rows of lights on each LED
    buttonCheck(); // checks for button interrupt, signaling turn
    threeLightCall();
    buttonCheck();
    threeLightCall();
    buttonCheck();
} } buttonCheck();
mpu.getEvent(&a, &g, &temp);
acc_1 = a.acceleration.y;
delay(50);
mpu.getEvent(&a, &g, &temp);
acc_2 = a.acceleration.y;
delay(50);
mpu.getEvent(&a, &g, &temp);
acc_3 = a.acceleration.y;
if(acc_1 <= 0 || acc_2 <= 0 || acc_3 <= 0){ // final check if still slowing down
    fiveLightCall(); // displays all rows of each LED
    buttonCheck(); // checks for button interrupt, signaling turn
    fiveLightCall();
    buttonCheck();
    fiveLightCall();
    buttonCheck();
    fiveLightCall();
    buttonCheck();
    fiveLightCall();
    buttonCheck();
} }

/*Function: Interrupt1
 * Input: N/A
 * Returns: This function interrupts when the left turn button is pushed!
 */
void interrupt1(){
    button_time = millis();
    //check to see if increment() was called in the last 250 milliseconds
    if (button_time - last_button_time > 250){
        Serial.println("left turn");
        turnInd = 1;
        last_button_time = button_time;
    }
}

/*Function: Interrupt2
 * Input: N/A
 * Returns: This function interrupts when the right turn button is pushed!
 */
void interrupt2(){
    button_time = millis();
    //check to see if increment() was called in the last 250 milliseconds
    if (button_time - last_button_time > 250){
        Serial.println("right turn");
        turnInd = 2;
        last_button_time = button_time;
    }
}
/* Function: Right Turn
 * Input: N/A
 * Returns: After being called, it clears the outer 2 rows of the 5x5 arrangement for the left pair
 * and lights the other 3 to red. It then flashes the outer 2 LEDs on the right pair yellow.
 * The reduced amount of LEDs is to save power.
 */
void rightTurn(){
    Serial.println("Made to right turn");
    clearLEDs();
    for(int i = 0; i < 25; i++){
        leds[i] = CRGB::Black;
    }
    FastLED.show();
    for(int i = 25; i < 50; i++){
        leds2[i] = CRGB::Black;
    }
    FastLED.show();
    for(int i = 5; i < 20; i++){
        leds[i] = CRGB::Red;
    }
    FastLED.show();
    int j = 0;
    while(j < 6){
        for(int i = 25; i < 35; i++){
            leds2[i] = CRGB::Yellow;
        }
        FastLED.show();
        delay(500); // Delay for 500 milliseconds
        for(int i = 25; i < 35; i++){
            leds2[i] = CRGB::Black;
        }
        FastLED.show();
        delay(500); // Delay for 500 milliseconds
        j++;
    }
    clearLEDs();
    turnInd = 0;
}

/* Function: Left Turn
 * Input: N/A
 * Returns: After being called, it clears the outer 2 rows of the 5x5 arrangement for the right pair
 * and lights the other 3 to red. It then flashes the outer 2 LEDs on the left pair yellow.
 * The reduced amount of LEDs is to save power.
 */
void leftTurn(){
    Serial.println("Made to left turn");
    clearLEDs();
    for(int i = 0; i < 25; i++){
        leds[i] = CRGB::Black;
    }
    FastLED.show();
    for(int i = 25; i < 50; i++){
        leds2[i] = CRGB::Black;
    }
    FastLED.show();
    for(int i = 5; i < 20; i++){
        leds[i] = CRGB::Red;
    }
    FastLED.show();
    int j = 0;
    while(j < 6){
        for(int i = 25; i < 35; i++){
            leds2[i] = CRGB::Yellow;
        }
        FastLED.show();
        delay(500); // Delay for 500 milliseconds
        for(int i = 25; i < 35; i++){
            leds2[i] = CRGB::Black;
        }
        FastLED.show();
        delay(500); // Delay for 500 milliseconds
        j++;
    }
    clearLEDs();
    turnInd = 0;
}
```cpp
// FastLED.show();
int j = 0;
while(j < 6){
    for(int i = 0; i < 10; i++){
        leds[i] = CRGB::Yellow;
    }
    FastLED.show();
delay(500);
    for(int i = 0; i < 10; i++){
        leds[i] = CRGB::Black;
    }
    FastLED.show();
delay(500);
    j++;
}
clearLEDs();
turnInd = 0;
}

/*Function: button Check
 * Input: N/A
 * Returns: checks to see what button has been pressed and carries out the command for the button press
 */
void buttonCheck(){
    if (turnInd == 1){
        leftTurn();
    }
    if(turnInd == 2){
        rightTurn();
    }
}

/*Function: displays no color on LEDs. Turns them off
 * Input: N/A
 * Returns: LED turned off
 */
void clearLEDs(){
    for(int i = 0; i < 25; i++){
        leds[i] = CRGB::Black;
    }
    FastLED.show();
    for(int i = 25; i < 50; i++){
        leds2[i] = CRGB::Black;
    }
    FastLED.show();
}

/*Function: First Brake Signal
 * Input: Accelerometer
 * Returns: If the accelerometer reports that the bike has started slowing down
 * then the middle row of each LED will turn on and blink slowly.
 */
void oneLightCall(){
    for(int i = 10; i < 15; i++){
        leds[i] = CRGB::Red; // set LEDs to yellow
    }
    FastLED.show(); // turn off LEDs
}
```
FastLED.show(); // turn off LEDs
delay(150);
clearLEDs();
delay(125);

/*Function: Second Brake Signal
* Input: Accelerometer
* Returns: If the accelerometer reports that the bike is still slowing down
* after it has already completed the first brake signal calls, then the three
* middle rows of each LED will turn on and blink a bit faster.
*/
void threeLightCall(){
    Serial.println("Turn on three rows of LEDs");
    for(int i = 5; i < 20; i++){
        leds[i] = CRGB::Red; // set LEDs to yellow
    }
    FastLED.show(); // turn off LEDs
    for(int i = 30; i < 45; i++){
        leds2[i] = CRGB::Red; // set LEDs to yellow
    }
    FastLED.show(); // turn off LEDs
delay(100);
clearLEDs();
delay(75);
}

/*Function: Final Brake Signal
* Input: Accelerometer
* Returns: If the accelerometer reports that the bike is still slowing down
* after it has already completed the first brake signal calls, and the second
* then all five rows of each LED will turn on and blink the fastest.
*/
void fiveLightCall(){
    Serial.println("Turn on all LED's");
    for(int i = 0; i < 25; i++){
        leds[i] = CRGB::Red; // set LEDs to yellow
    }
    FastLED.show(); // turn off LEDs
    for(int i = 25; i < 50; i++){
        leds2[i] = CRGB::Red; // set LEDs to yellow
    }
    FastLED.show(); // turn off LEDs
delay(75);
clearLEDs();
delay(50);
}

5 Mechanical Drawings:

Below are the dimensions of the light sensor, accelerometer, LEDs, Battery, step-up and our enclosure
Figure 5.1: Accelerometer chip dimensions from data sheet

Figure 5.2: Accelerometer board dimensions

Figure 5.3: Light Dependent Resistor
Figure 5.4: Arduino Nano datasheet dimensions

Figure 5.5: Data Sheet WS2812 LEDs
Total Dimensions: 3.46 x 2.32 x 0.47 inches

Figure 5.6: Button dimensions - units are mm
Figure 5.7: Power dimensions - units are mm

Figure 5.8: Step-up dimensions

- DC-DC boost converter module, operating frequency 150kHz, typical conversion efficiency of 85%.
- Pin 2.54MM pitch.
- Input voltage: 0.9-5V, output voltage: 5V, maximum output current: 480 mA.
- Dimensions: 11mm x 10.5mm x 7.5mm (ultra-small module, 1mm=0.0393 inch).
- Weight: about 1g
<table>
<thead>
<tr>
<th>Dimensions</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Length</td>
<td>28 mm (~1.103&quot;)</td>
</tr>
<tr>
<td>Diameter</td>
<td>17 mm (~0.669&quot;)</td>
</tr>
<tr>
<td>Weight</td>
<td>1.6 g (0.057 oz)</td>
</tr>
</tbody>
</table>

Figure 5.9: Battery Charger

Figure 5.10: Enclosure - all measurements in mm
6 Block Diagram:

This is our final block diagram that encompasses the functionality of our whole system.

![Block Diagram](image1)

7 Top Level Diagram

![Top Level Diagram](image2)
## 8 Interface Specifications:

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
<th>Expected Values</th>
</tr>
</thead>
</table>
| **ls_CPU** | The light sensor will measure light outside and report that to the Computing block. Expected value: An 1.95 V indicates "dark" and above 1.95V indicates "light". | Vmeasured = 3.88 V  
Vmax = 5 V DC  
Pmax = 100 mW  
Imeasured = 0.248mA |
| **b_cpu** | One push of the button (active low) leads to a change in LED. If the left button or right button is pushed it will be reported to the Computing block. Expected value: 0 indicates button pushed while 1 indicates no button has been pushed. | Vmeasured = 3.87 V  
Vmax = 5 V  
Vmin = 3.3 V  
Imeasured = 3.76mA |
| **ac_cpu** | If the bike speed is decreasing it will be reported to the computing block. Expected value: A speed decrease of at least one mph. It has an acceleration sensitivity range from -2g to +16g | Vmeasured = 3.87 V  
Vmax = 5 V  
Vmin = 3.3 V  
Imax = 5.7 mA |
| **light_con** | If a decrease in speed, a button is clicked or the light sensor outputs that its dark outside an specific LED pattern will be displayed. If a button is clicked a one LED will blink while the other remains solid, if the rider decreases in speed both LEDs will increase in brightness and if the light sensor indicates darkness outside both LEDs will turn on. Expected Value: Light > 1.95 V and Dark < 1.95 V for light sensor. | Vsig measured = 0.09 V  
Vmax = 5.0 V  
P = 0.7W  
Scan Frequency: 400 Hz/s  
Duty Cycle Range: 48-52(%)  
Imeasured = 0.003mA |
cb_pow  | LiPo battery supplies power to our controlling block. It supplies 4.95 V to the arduino nano which controls all interfaces. | Vnom = 5 V  
Vmin = 4.7 V  
Vmeasured = 4.96V  
Imin = 12.7 mA  
Imax = 16.7 mA

wall_pow  | Battery pack is charged using a wall connector. It uses a usb-c connection to plug into the wall. This plug uses 5V and 2.0A. | Vnom = 5.2 V  
Vmeasured = 4.97 V  
Imeasured= 2.4 A  
Pmax = 12W

led_pow  | Battery pack supplies power to our LEDs directly. This is connected before the step up, so the LEDs are supplied 3.68 V. | Vnom = 3.7 V  
Vmin = 3.5 V  
Vmeasured = 3.68 V  
Imeasured = 800mA

ls_in  | Light input to the light sensor | N/A

user_in  | User input via button press | N/A

accel_in  | Acceleration input | Acceleration_Max: 78.4 m/s²

led_out  | Light output from the LED | N/A

9 PCB Layers:

This is a diagram of the PCB we decided to use for our project. This image showcases the different connections and components we decided to use on the PCB.

![Figure 9.1: PCB layer;](image-url)
# Bill of Materials:

<table>
<thead>
<tr>
<th>Ref.</th>
<th>Block</th>
<th>Part</th>
<th>Part Number</th>
<th>Seller</th>
<th>Purchase Link</th>
<th>Datasheet Link</th>
<th>Qty</th>
<th>Unit Cost</th>
<th>Total Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>LS1</td>
<td>Light Sensor</td>
<td>LDR Resistor</td>
<td>GM5539</td>
<td>Amazon</td>
<td>Link Here</td>
<td>Link Here</td>
<td>1</td>
<td>$0.22</td>
<td>$0.22</td>
</tr>
<tr>
<td>R1</td>
<td>Light Sensor</td>
<td>5K Resistor</td>
<td>DigiKey</td>
<td>Amazon</td>
<td>Link Here</td>
<td>Link Here</td>
<td>1</td>
<td>$0.10</td>
<td>$0.10</td>
</tr>
<tr>
<td>LED1</td>
<td>LED</td>
<td>5V 20 LED Matrix - RGB</td>
<td>WS2812</td>
<td>Amazon</td>
<td>Link Here</td>
<td>Link Here</td>
<td>2</td>
<td>$3.66</td>
<td>$7.32</td>
</tr>
<tr>
<td>R2</td>
<td>Light Sensor</td>
<td>10K Resistor</td>
<td>DigiKey</td>
<td>Amazon</td>
<td>Link Here</td>
<td>Link Here</td>
<td>2</td>
<td>$0.10</td>
<td>$0.20</td>
</tr>
<tr>
<td>B1/B2</td>
<td>Button</td>
<td>Tactile Switches (Push Buttons)</td>
<td>CKN9092-ND</td>
<td>Amazon</td>
<td>Link Here</td>
<td>Link Here</td>
<td>2</td>
<td>$1.00</td>
<td>$2.00</td>
</tr>
<tr>
<td>R4/R5</td>
<td>Button</td>
<td>1K Resistor</td>
<td>DigiKey</td>
<td>Amazon</td>
<td>Link Here</td>
<td>Link Here</td>
<td>2</td>
<td>$0.10</td>
<td>$0.20</td>
</tr>
<tr>
<td>N/A</td>
<td>Button</td>
<td>Spiral Cable Wrap</td>
<td>B0918</td>
<td>Amazon</td>
<td>Link Here</td>
<td>N/A</td>
<td>1</td>
<td>$0.99</td>
<td>$0.99</td>
</tr>
<tr>
<td>ACC1</td>
<td>Accelerometer</td>
<td>3 Axis Accelerometer Gyro Scope Module 6</td>
<td>GY-521 MPU-6050</td>
<td>Amazon</td>
<td>Link Here</td>
<td>Link Here</td>
<td>1</td>
<td>$1.99</td>
<td>$1.99</td>
</tr>
<tr>
<td>B1</td>
<td>Power</td>
<td>Lithium Battery Charger</td>
<td>TP4056</td>
<td>Amazon</td>
<td>Link Here</td>
<td>Link Here</td>
<td>1</td>
<td>$0.80</td>
<td>$0.80</td>
</tr>
<tr>
<td>SU10C</td>
<td>Power</td>
<td>5V DC-DC Step Up Power Module</td>
<td>20484253</td>
<td>Amazon</td>
<td>Link Here</td>
<td>Link Here</td>
<td>1</td>
<td>$1.46</td>
<td>$1.46</td>
</tr>
<tr>
<td>SW1A</td>
<td>Power</td>
<td>Switch</td>
<td>Q0M20GA</td>
<td>Fakro</td>
<td>Link Here</td>
<td>Link Here</td>
<td>1</td>
<td>$0.33</td>
<td>$0.33</td>
</tr>
<tr>
<td>LPOR1</td>
<td>Power</td>
<td>Lipo Battery Rechargeable</td>
<td>S03573</td>
<td>Amazon</td>
<td>Link Here</td>
<td>Link Here</td>
<td>1</td>
<td>$10.99</td>
<td>$10.99</td>
</tr>
<tr>
<td>A1</td>
<td>Computing</td>
<td>Arduino Nano</td>
<td>CH340G</td>
<td>Amazon</td>
<td>Link Here</td>
<td>Link Here</td>
<td>1</td>
<td>$6.55</td>
<td>$6.55</td>
</tr>
<tr>
<td>N/A</td>
<td>All blocks</td>
<td>Wire</td>
<td>AWG 26</td>
<td>Amazon</td>
<td>Link Here</td>
<td>Link Here</td>
<td>10</td>
<td>$1.00</td>
<td>$10.00</td>
</tr>
<tr>
<td>N/A</td>
<td>All blocks</td>
<td>Protoboard</td>
<td>P1401 783757</td>
<td>Fakro</td>
<td>Link Here</td>
<td>N/A</td>
<td>2</td>
<td>$2.00</td>
<td>$4.00</td>
</tr>
<tr>
<td>N/A</td>
<td>Enclosure</td>
<td>PLA for 3D print</td>
<td>250007156</td>
<td>Amazon</td>
<td>Link Here</td>
<td>N/A</td>
<td>1</td>
<td>$10.00</td>
<td>$10.00</td>
</tr>
<tr>
<td>N/A</td>
<td>Enclosure</td>
<td>M3 Hex Cap Screw</td>
<td>M3 12mm</td>
<td>Amazon</td>
<td>Link Here</td>
<td>N/A</td>
<td>6</td>
<td>$0.04</td>
<td>$0.24</td>
</tr>
<tr>
<td>N/A</td>
<td>Enclosure</td>
<td>M3 Nut</td>
<td>M3 Nut</td>
<td>Amazon</td>
<td>Link Here</td>
<td>N/A</td>
<td>6</td>
<td>$0.04</td>
<td>$0.24</td>
</tr>
<tr>
<td>N/A</td>
<td>PCB</td>
<td>PCB</td>
<td>N/A</td>
<td>Cолн Park</td>
<td>N/A</td>
<td>N/A</td>
<td>2</td>
<td>$2.10</td>
<td>$4.20</td>
</tr>
</tbody>
</table>
11 Team Time Report:

<table>
<thead>
<tr>
<th>Team Members</th>
<th>Time spent in hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oluwaseun Samuel Popoola</td>
<td>55</td>
</tr>
<tr>
<td>Farihya Osman</td>
<td>70</td>
</tr>
<tr>
<td>Elizabeth Wade</td>
<td>75</td>
</tr>
</tbody>
</table>