Wearable & Ubiquitous Computing

CSCE 4114

Sprint 2019

Mon, Wed, Fri (12:55 PM – 1:45 PM)

Assignment 1

Visual Light Meter

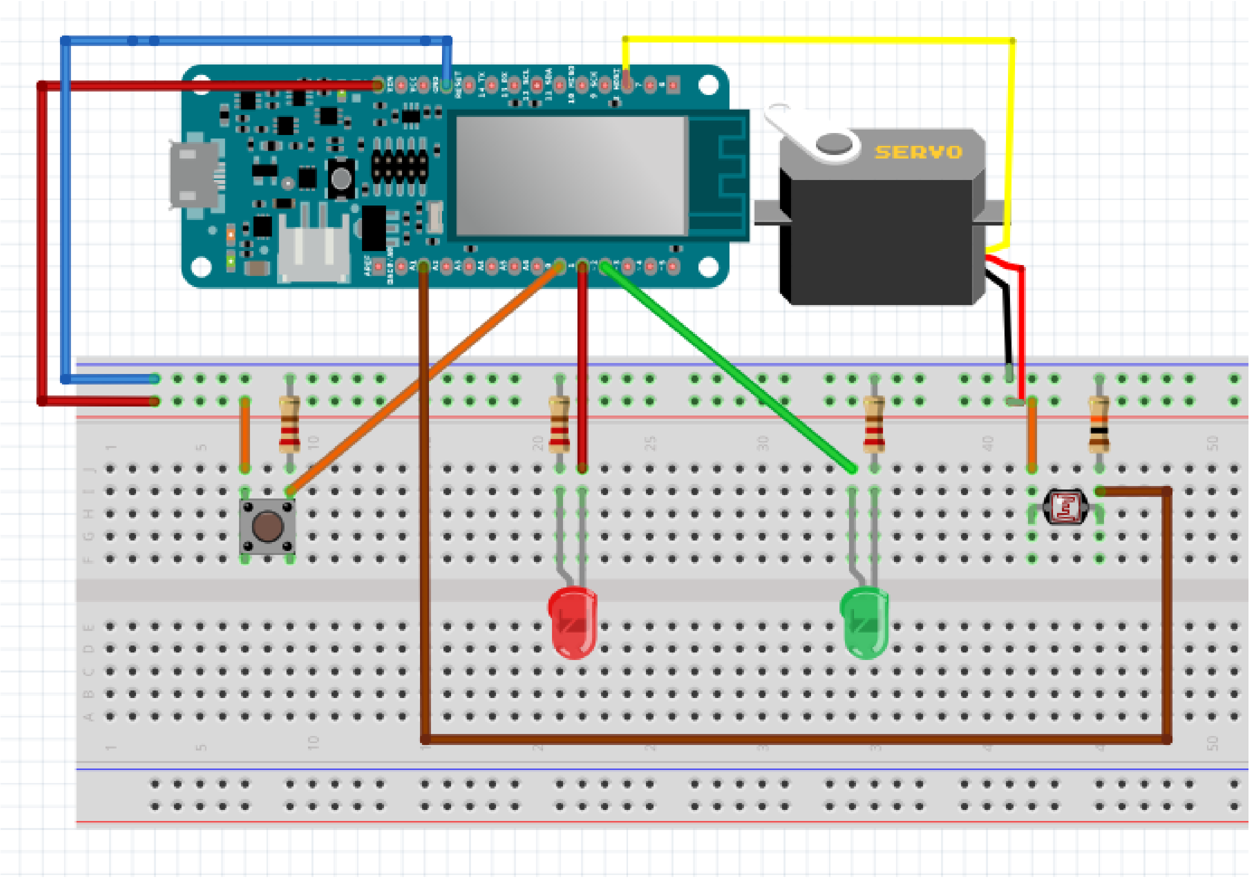
02-26-2019

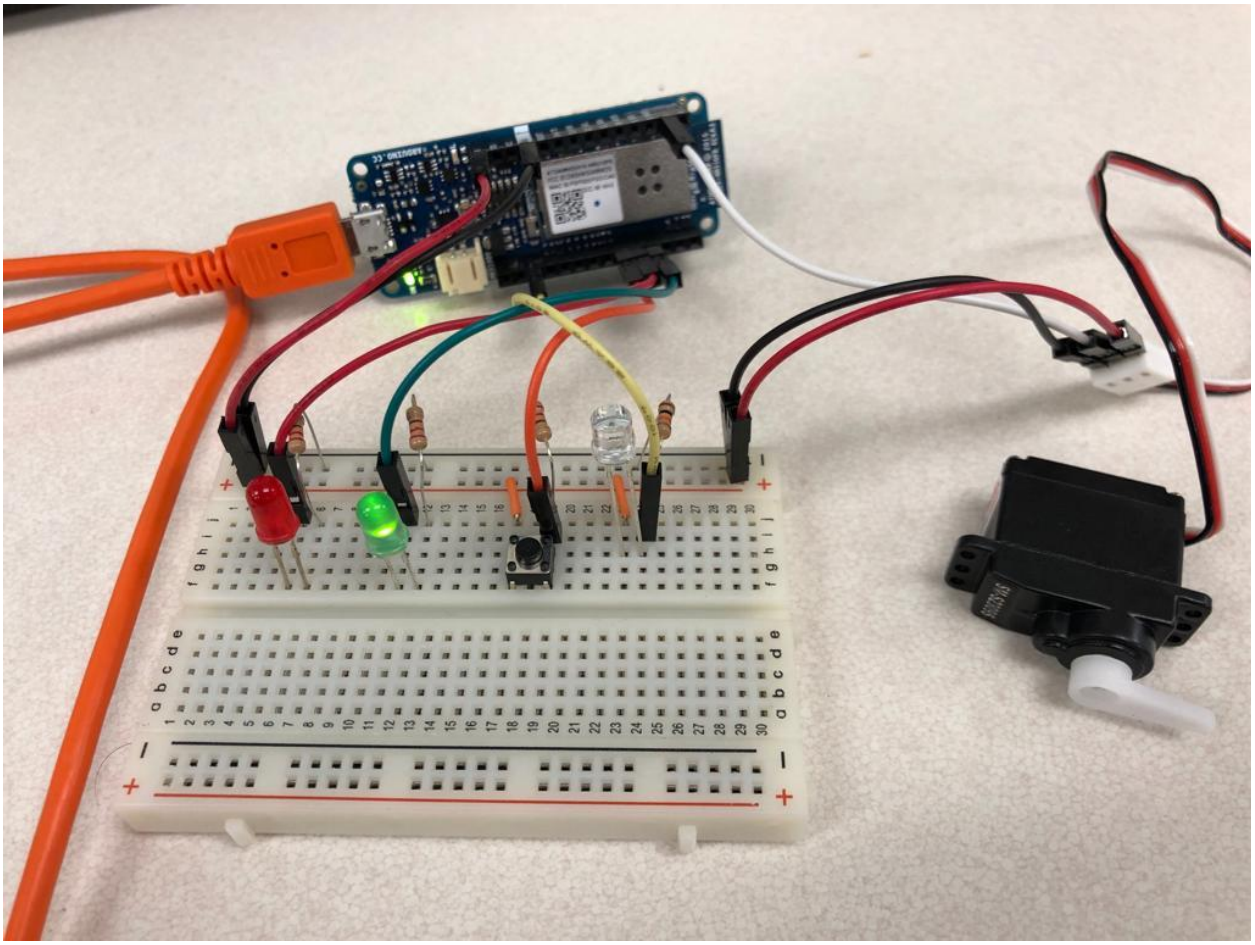
Suhail Basalama

010786839

sebasala@email.uark.edu

Diagram




ARDUINO.CC
a b c d e
f g h i j
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
+
−

Report

In this project, I created a system that measures brightness and provides the user with two outputs based on two modes:

1- First mode: representing the degree of brightness in terms of a gauge pointer, where the pointer starts from the left most, meaning zero brightness, to the right most, meaning full brightness.
2- Second mode (sensitive mode): Being exposed to a certain level of brightness, the system will start a countdown of 30 seconds using the gauge pointer before alarming the user with a red light.

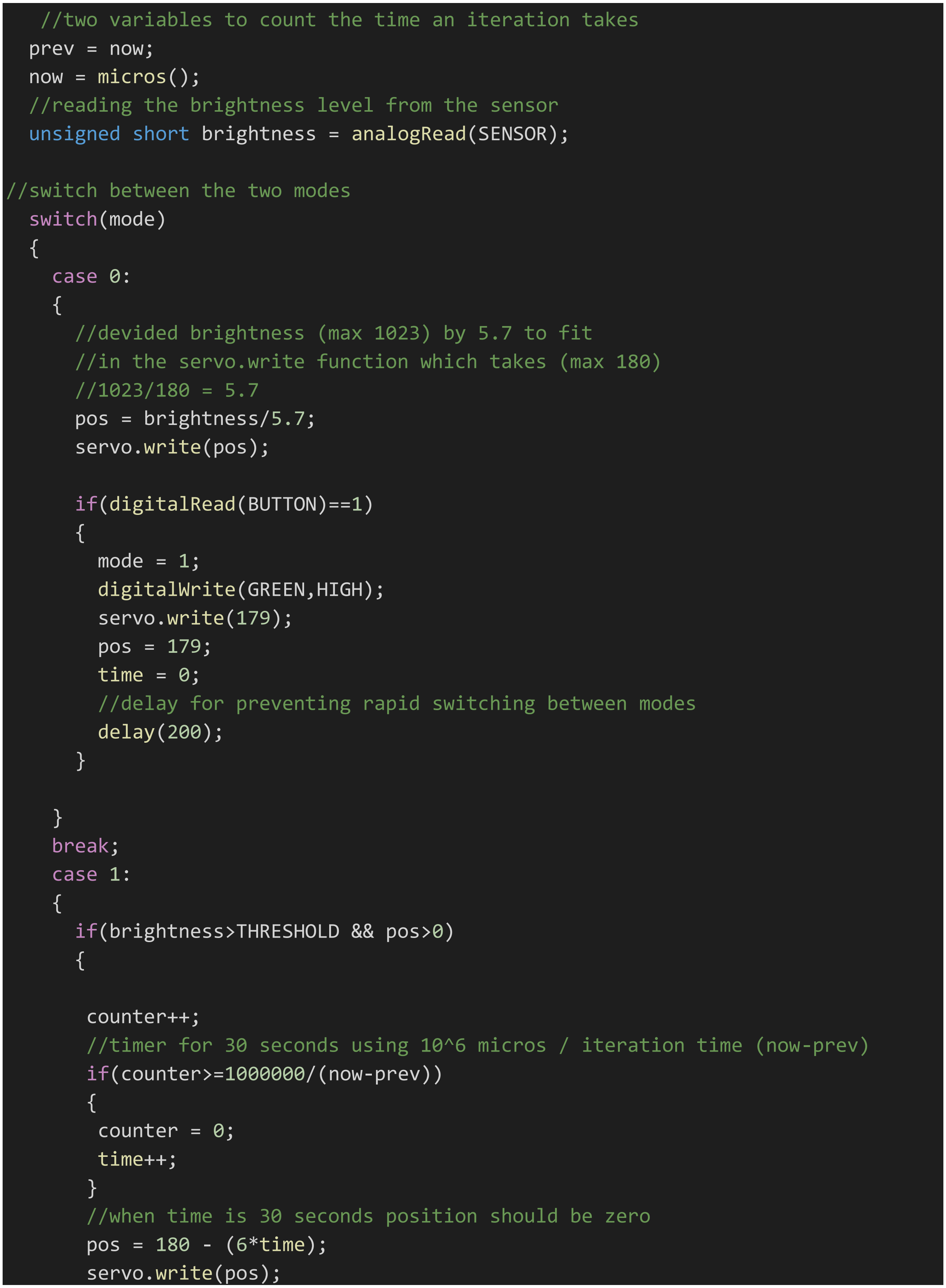
Hardware setup is as shown in the diagram above. The code is relatively easy and self-explanatory. To avoid rapid transition between the two modes upon pressing the button, I added a delay of 200 ms after each press which simply resolves this issue. There was no need to implement state machines for debounced button as we did in Embedded Systems because counting multiple clicks within 200 ms will not affect the operation of the system. For the Servo, I used the built-in class called Servo with its functions. To count for 30 seconds, I used the built-in function micros to figure out the time a single iteration of the loop takes and used this result to count for 30 seconds. The threshold for the second mode is set as 800 but can be modified based on the sensitivity level needed.

Results

I faced many challenges, but I overcome them through unit testing and functions. The system now passed all the 10 cases, and it produces the expected outputs.

Source Code

```
#include <Servo.h>

//pin and constant definitions
#define BUTTON 0
#define RED 1
#define GREEN 2
#define SENSOR A1
#define SERVO 7
#define THRESHOLD 800

//global variables and objects
Servo servo;
unsigned char mode = 0;
unsigned int counter;
unsigned char time;
int now;
int prev;
unsigned char pos;

void setup() {

  //setting up different components
  pinMode(BUTTON,INPUT);
  pinMode(RED,OUTPUT);
  pinMode(GREEN,OUTPUT);
  pinMode(SENSOR,INPUT);
  servo.attach(SERVO);

  //initial blinking upon reset
  for(int i=0; i<3; i++)
  {
    digitalWrite(GREEN,HIGH);
    digitalWrite(RED,HIGH);
    delay(100);
    digitalWrite(GREEN,LOW);
    digitalWrite(RED,LOW);
    delay(100);
  }
}

void loop() {
```



```
   //two variables to count the time an iteration takes
  prev = now;
  now = micros();
  //reading the brightness level from the sensor
  unsigned short brightness = analogRead(SENSOR);

//switch between the two modes
  switch(mode)
  {
    case 0:
    {
      //devided brightness (max 1023) by 5.7 to fit
      //in the servo.write function which takes (max 180)
      //1023/180 = 5.7
      pos = brightness/5.7;
      servo.write(pos);

      if(digitalRead(BUTTON)==1)
      {
        mode = 1;
        digitalWrite(GREEN,HIGH);
        servo.write(179);
        pos = 179;
        time = 0;
        //delay for preventing rapid switching between modes
        delay(200);
      }

    }
    break;
    case 1:
    {
      if(brightness>THRESHOLD && pos>0)
      {

       counter++;
       //timer for 30 seconds using 10^6 micros / iteration time (now-prev)
       if(counter>=1000000/(now-prev))
       {
        counter = 0;
        time++;
       }
       //when time is 30 seconds position should be zero
       pos = 180 - (6*time);
       servo.write(pos);
      }

      else if(pos==0)
      {
        digitalWrite(RED,HIGH);
        digitalWrite(GREEN,LOW);
      }

      if(digitalRead(BUTTON)==1)
      {
        mode = 0;
        digitalWrite(RED,LOW);
        digitalWrite(GREEN,LOW);
        //delay for preventing rapid switching between modes
        delay(200);
      }
    }
  }

}
```