Prototype Lab

ME 2900

Instructor: Dr. Sandra Metzler

Date of Submission: 10/17/16

Mark Luke

I. Introduction

When living in a house with ten roommates, locking the door when leaving and coming into the house can be an issue. To solve this, a prototype sensor was built to detect whether a door is locked or not. The prototype detects the distance the lock is from itself, displays a written message and a green or red light depending on the lock, and also counts amount of times the door is used. Both electrical and physical components had to be used with the prototype, as well as one new actuator and one new sensor. The prototype should be pre-factory quality when constructed. Code for the prototype needs to have at least two user defined functions with variables being passed to and from them. New libraries and several flow structures are also required.

II. Idea Development Process

To come up with a solution to ensure my housemates would lock the door, I brainstormed two different prototypes. While similar, they differed in the way they detected the door was opened. The first option used an accelerometer to tell whether the door was open or not, while the second used a distance sensor. Once open, a message and sound would appear for both prototypes, alerting the person to lock the door. Both would use similar programming, the only thing changing would be the logic depending on the sensor chosen. In the end, a hybrid of these two prototypes with added features was chosen.

III. Discussion

The final prototype features an accelerometer, IR distance sensor, servo motor, RBG LED, and a 7 segment clock display. The accelerometer waits until it detects an acceleration over 0.15g's, which simulates a door opening. Then, the distance sensor starts taking readings of the lock. If the lock is locked (turned vertically), the sensor will read a value above 600, and will prompt the RBG LED to turn green and the servo motor to display the LOCKED message. If the sensor reads a value below 600, the LED will turn red, and the servo motor will flip 180

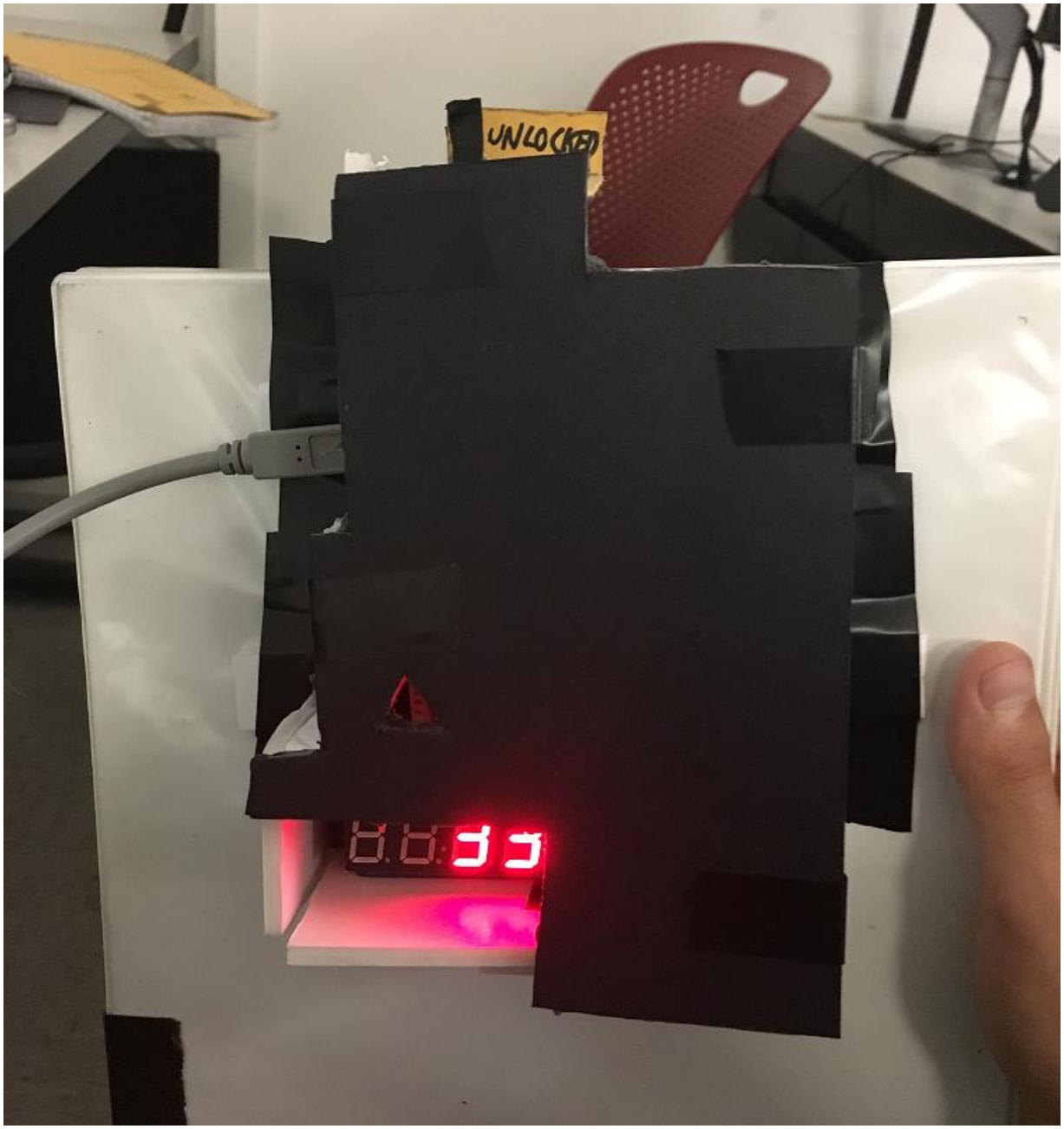
degrees and display an UNLOCKED message. Once this segment of the code is done running, the clock display will display how many times the door has been opened.

In order to read the correct acceleration, the accelerometer had to be calibrated vertically. This forces the Z axis acceleration to be 0g when the prototype is set up on the door. A tiny structure is actually held by springs in place, and when it is deflected, the accelerometer calculates the acceleration from this deflection. The distance sensor shoots out an infrared light, which bounces back into it, and calculates the distance from the intensity of the reflected light.

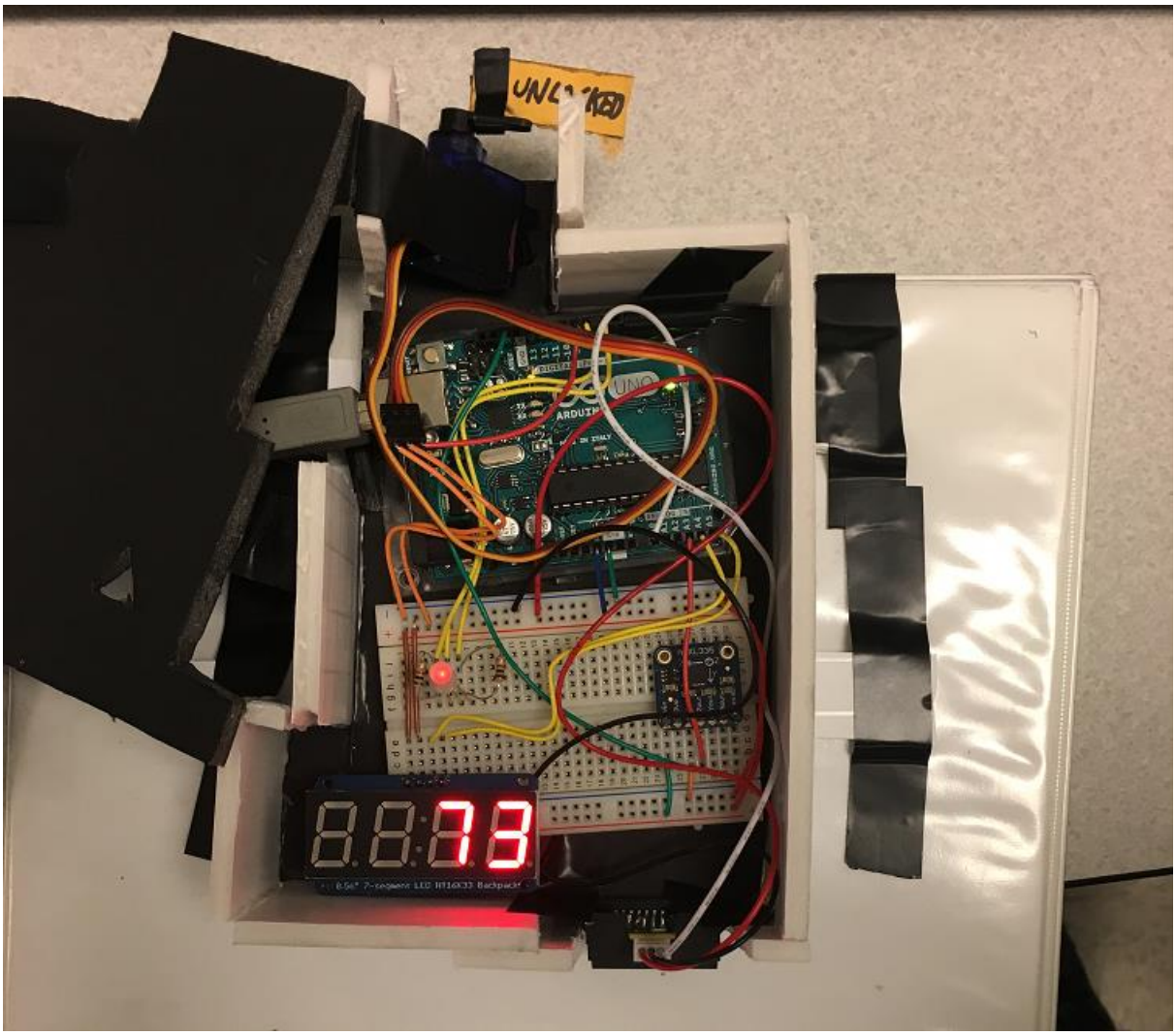
IIV. Conclusion

The prototype created successfully detects when a door is open, and whether it is locked or unlocked. Parts of the servo that work well include the servo motor stating whether the door is unlocked or locked, and the clock display counter. The code for the counter is particularly clever once counting past 9. One difficulty when creating the prototype was when trying to detect the locks position. An optosensor was first going to be used, but it was only reading values when the lock was extremely close to it. This was not ideal, and the switch to the IR distance sensor was made. The distance sensor works much better, and is pretty accurate when sensing the lock. It still does have trouble when detecting a thin object, and can be observed sensing the floor instead of the lock in the video. With more time, a better physical shell would be made around the prototype, as well as attachments to bolt the device onto a door. The distance sensor would be lined up with the lock and screwed in place. A clock would also be added, and the code would be changed to have the sensor only turn on during certain times of the day.

Appendix A: Photographs of Prototype



Picture 1: Picture of the prototype with the outside shell on.



Picture 2: Picture of the prototype showing the electrical components

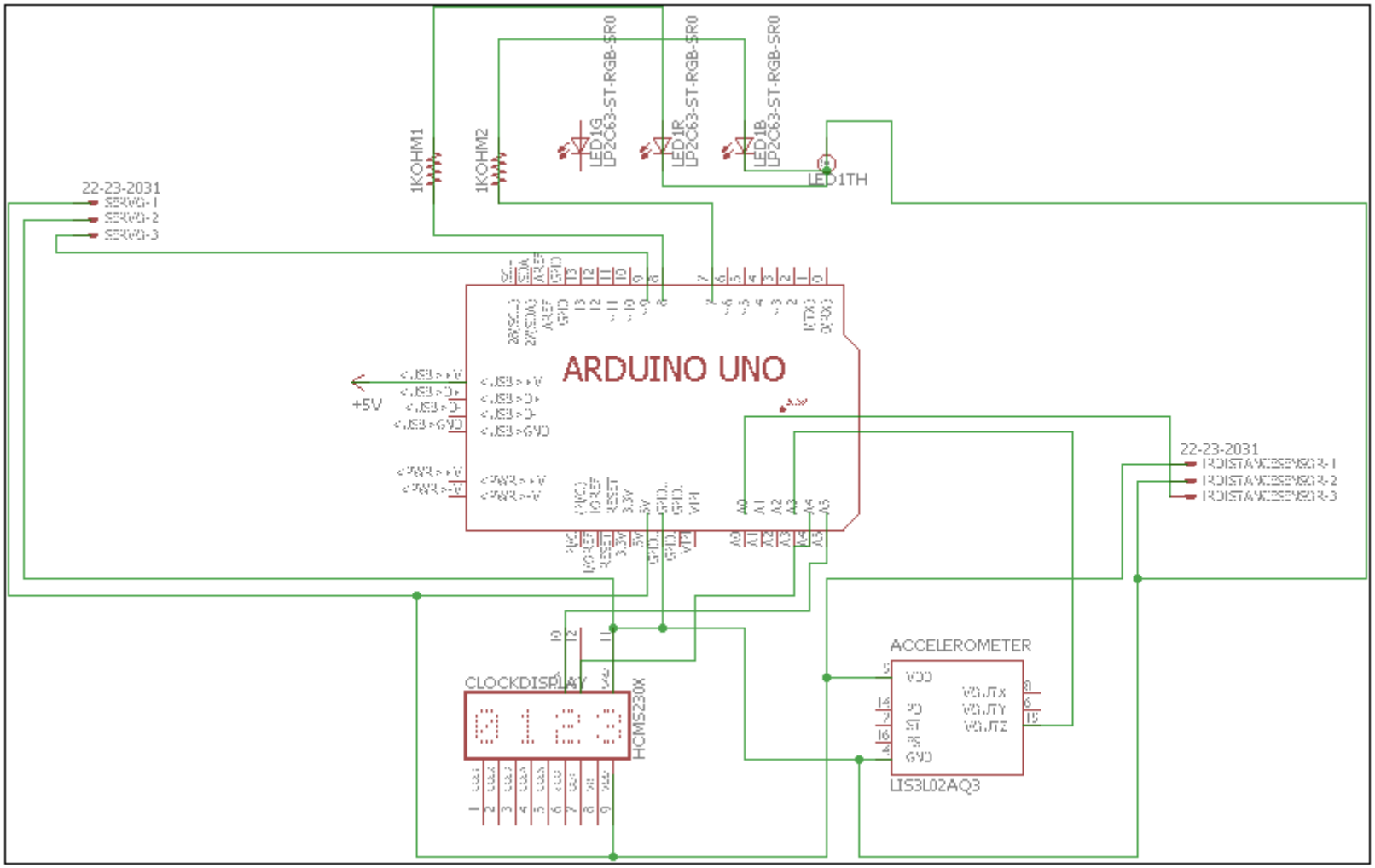
Appendix B: Arduino Sketch

```
//Backpack display libraries
#include <Adafruit_LEDBackpack.h>
#include <gfxfont.h>
#include <Adafruit_GFX.h>
#include <Wire.h>
Adafruit_7segment matrix = Adafruit_7segment();

//Library for servo
#include <Servo.h>

//Declaring sensors and variables for them
const int distsens = A0;
int distVolt;

Servo myservo;    //Declaring the servo as myservo
int pos = 0;    //Starting the servo at position 0

//Constants for accelerometer sensor
const int zInput = A3;

//Raw ranges of accelerometer
int zRawMin = 421;
int zRawMax = 629;

//Take multiple readings to reduce the sensor noise
const int sampleSize = 10;

//RGB LED variables
const int ledRED = 8;
const int ledGREEN = 7;

//Counter variable
int i = 0;

void setup()
{

  Serial.begin(9600);   //Begin communicating with the arduino
  analogReference(EXTERNAL);  //For the accelerometer
  pinMode(distsens, INPUT);   //Declare the distance sensor as an input
  myservo.attach(9);    //Servo attached to pin 9
  myservo.write(0);     //Sets initial servo position
  matrix.begin(0x70);         //Begin matrix inputs
  matrix.writeDigitNum(4, 0);   //Start counter at 0
  matrix.writeDisplay();

}

void loop()
{

    //Don't need x and y values
    int zRaw = ReadAxis(zInput);

    // Convert raw values to 'milli-Gs"
    long zScaled = map(zRaw, zRawMin, zRawMax, -1000, 1000);

    // re-scale to fractional Gs
    float zAccel = zScaled / 1000.0;

    //Check to make sure still getting right values
    Serial.print(zAccel);
    Serial.println("G");

    if(abs(zAccel) > 0.15)
    {

      doorOpen();

    }

    delay(50);
}
```

```
// Read the samplesize variable and return the average
int ReadAxis(int axisPin)
{
  long reading = 0;

  analogRead(axisPin);  //Reads pin for accel
  delay(1);

  for (int i = 0; i < sampleSize; i++)  //For loop to take 10 samples
  {

    reading += analogRead(axisPin);

  }

  return reading/sampleSize;    //Returns average of 10 samples

}

//Code to run when door opens
void doorOpen()
{

delay(4000);   //Wait 4 seconds for door to close

int time1 = millis();
int time2, onesDigit, tensDigit;

  distVolt = analogRead(distsens);  //Reads value of distance sensor, need to figure out values when about 3-4 cm apart
  Serial.println(distVolt);   //Check to see value

     if (distVolt < 600)   //Door is unlocked when sensor is less than 600
     {

       Serial.println("Door is unlocked");   //Checks for testing
       delay(500);
       digitalWrite(ledRED, HIGH);

       do
       {

         myservo.write(180);   //Change the orientation of the servo to display a message

         distVolt = analogRead(distsens);
         Serial.println(distVolt);   //Check to see value
         delay(500);

         time2 = millis();
         if(time2 - time1 > 10000)
         {
           break;    //breaks the loop if the door is not locked within 10 seconds
         }

       } while (distVolt < 600);     //checks until the door is locked

     }

  distVolt = analogRead(distsens);
  Serial.println(distVolt);
  delay(500);

  if (distVolt > 600)     //Door is locked
  {

    Serial.println("Door is locked");   //Checks for testing
    digitalWrite(ledRED, LOW);
    digitalWrite(ledGREEN, HIGH);
    myservo.write(0);   //Moves servo to say door is locked
    delay(1000);

  }
```

```
i++;    //Add one to counter


if(i < 100 && i >= 10)    //Sets clock display to # of times door opens
{

  onesDigit = i % 10;
  tensDigit = (i - onesDigit) / 10;

  matrix.writeDigitNum(3, tensDigit);
  matrix.writeDigitNum(4, onesDigit);
  matrix.writeDisplay();

}
else if(i < 10)
{
  matrix.writeDigitNum(4, i);
  matrix.writeDisplay();
}

    digitalWrite(ledRED, LOW);
    digitalWrite(ledGREEN, LOW);

}
```

Appendix C: Arduino Schematic


1KOHM1
1KOHM2
LED1G
LP2C63-ST-RGB-SR0
LED1R
LP2C63-ST-RGB-SR0
LED1B
LP2C63-ST-RGB-SR0
LED1TH
22-23-2031
ARDUINO UNO
+5V
22-23-2031
CLOCKDISPLAY
HCMS230X
ACCELEROMETER
LIS3L02AQ3