POE - Project 0/.5 Blinky Light and Infared Sensor

Claire Diehl and Emily Wang

September 2013

Contents

1 Introduction

The purpose of this lab was to use a microcontoller and a simple circuit to create a bike light with various light modes. We used a circuit comprised of five leds and a button to switch between the four light modes. In the following class (lab 0.5), we added an infared sensor into the bike light device as an additional mode.

2 Circuit

For this lab, we followed the provided circuit diagram. We built the circuit with 5 green leds (each with its own 690Ω resistor), a button, and then later added the infared sensor (Infrared Proximity Sensor Long Range - Sharp GP2Y0A02YK0F) based on the information provided in its datasheet. The circuit diagram for our project can be seen in figure 1.

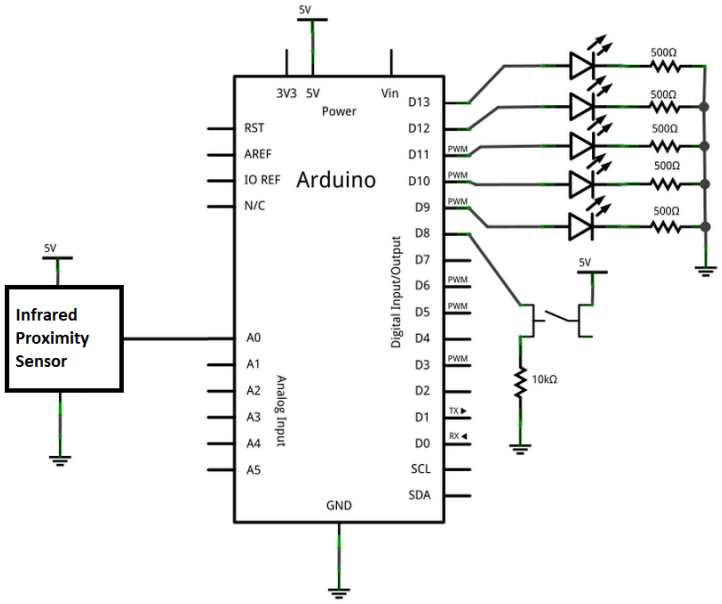

Figure 1: Circuit Diagram

3 Code

We took a modular approach to progamming for the sake of efficient debugging.

3.1 General Structure

At the top of the script, we declared the variables to be used throughout the program. Variable names were used as needed for the sake of readability (i.e. writing ledPin0 rather than the number 9 for the name of a pin). We used "old" and "new" variables to record the status of the button and time durations (old refers to the status at the end of the previous iteration of the loop, while new refers to the status of the current loop - therefore the old variable inherits the new variable's value at the end of each loop to maintain the purposes).

Next, the setup() function initalized the LED, button, and sensor pins on the Arduino and took the first readings for oldState and oldTime. setup() only runs once when the Arduino is turned on or after new code is uploaded.

Afterwards, the loop() function is run continuously. Firstly, the Arduino checks whether the button has been pressed or not and changes the mode only if the button was pressed. Secondly, the Arduino checks whether one second has passed with millis() and the oldTime/newTime/passedTime variables - and acts

appropriately according to the current mode (see code below for more details). For example, if one second has passed and the bike light is on mode 1 (flashing), then the Arduino will change the on/off status of the LEDs. Every time the Arduino changes the on/off status for the flashing or bouncing modes, the oldTime variable is reassigned to the newTime value (to "recalibrate" the one-second-passing mechanism). At the very end of every loop() iteration, oldState is reassigned to the newState value in preparation for the next loop (following the definition that oldState is the status of the button at the very end of the previous loop).

3.2 Button press

At the beginning of each loop, new readings for button state (newState) and time values (newTime) are calculated. Immediately afterwards, the program checks if the button was recently pressed. A button press can be represented by the change in state from low to high. This can be detected by whether oldState is LOW and newState is HIGH in an if statement and ensures that the mode is only changed once per button press (see the time vs voltage graph shown below, i.e. Figure 2).
Every time the button is pressed, the modecounter variable is incremented by 1 and then a modulo operator is used to determine the appropriate mode (an integer from 0 to 4).

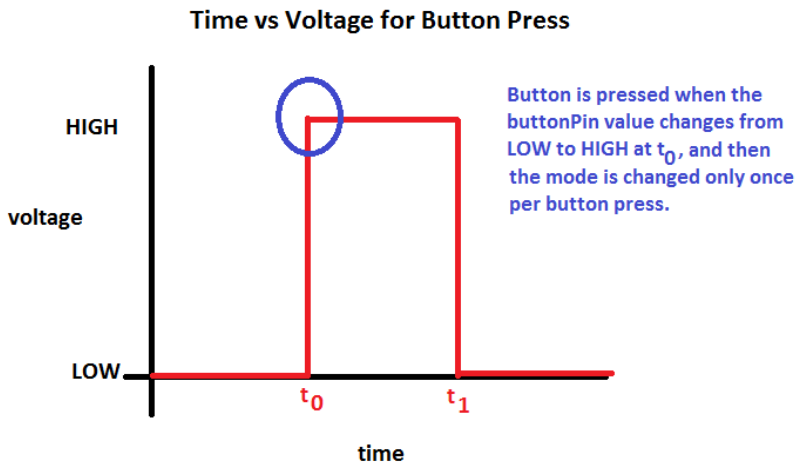

Figure 2: Visualizing what happens when the button is pressed (closes/open a switch and thus changes the buttonPin's reading).

3.3 Lab 0 - Modes

For lab 0, we chose to have four different modes

1. All on
2. All flashing
3. Bouncing LEDs
4. All off

To switch between the modes we created a variables mode and modecounter. When the button registers as being pressed, modecounter is incremented by one. As there are only four modes, the modecounter is divided by 5 with the remainder being set equal to the desired mode (this is easily accomplished with a modulo operator).

Mode zero is the configuration when all of the LEDs are turned on. This simply means that all of the ledPin variables are set to HIGH.

Mode one is flashing, a mode where the LEDs alternate between all on and all off. The boolean variable is used to determine the current "command" for the Arduino - the LEDs are all on for one second, and then change to all off for one second, and the pattern repeats as long as the bike light is still in this mode.

Mode two is bouncing. In this mode the LEDs alternate between three on and two on in the same way that mode one alternates (with a boolean variable ledConfig instead of ledOn).

Mode three is all off. In comparison to mode zero this means that all ledPins are set to LOW.

Mode four involves the infrared sensor, which is explained in detail in the next section.

3.4 Lab .5 - Sensor

The original goal for the sensor mode was to change which light was on for every 15cm of distance away from the sensor. However, the sensor appears to "peak" at approximately 15cm rather than express a decrease in voltage as distance increases. In other words, the sensor voltage increases quickly as you move from 0cm to 15cm distance, but the sensor voltage decreases as you move from 15cm and increase the distance.

However, this behavior can still be used to produce an interesting effect with the LEDs - we specified different measurement ranges (measurements taken with analogRead) and assigned a particular LED to light up for each measurement range. The "infrared sensor" mode can be accessed by button presses (known as mode four).

Appendices

```
1  // set constants
2  const int buttonPin = 8;
3  const int ledPin0 = 9;
4  const int ledPin1 = 10;
5  const int ledPin2 = 11;
6  const int ledPin3 = 12;
7  const int ledPin4 = 13;
8
9  //set variables (will change)
10 boolean oldState;
11 boolean newState;
12 boolean ledOn = 0;
13 boolean ledConfig = 0;
14 int modecounter = 0;
15 int mode = 0;
16 int sensorVoltage;
17 long oldTime;
18 long newTime;
19 long passedTime;
20
21 //begin
22 void setup() {
23   //allows serial monitor print
24   Serial.begin(9600);
25
26   //initialize the LED pins
27   pinMode(ledPin0, OUTPUT);
28   pinMode(ledPin1, OUTPUT);
29   pinMode(ledPin2, OUTPUT);
30   pinMode(ledPin3, OUTPUT);
31   pinMode(ledPin4, OUTPUT);
32
33   //initialize the button pin
34   pinMode(oldState, INPUT);
35   pinMode(newState, INPUT);
36
37   //initialize the sensor pin
38   pinMode(A0, INPUT);
39
40   //first time oldState is being read and first time
         oldTime is being recorded
41   oldState = digitalRead(buttonPin);
42   oldTime = millis();
```

```
43 }
44
45 //looping
46 void loop() {
47   //taking new readings at the beginning of each loop
48   newState = digitalRead(buttonPin);
49   newTime = millis();
50   passedTime = newTime - oldTime;
51
52   //was the button pressed?!
53   if (oldState == 0 && newState == 1){
54     modecounter = modecounter + 1;
55     mode = modecounter % 5;
56   }
57
58   if (passedTime >= 1000)
59   {
60     //reset oldTime after 1 sec has passed
61     oldTime = newTime;
62
63     //act according to mode
64     if (mode == 0){
65     //All On
66       Serial.println("All On");
67       digitalWrite(ledPin0, HIGH);
68       digitalWrite(ledPin1, HIGH);
69       digitalWrite(ledPin2, HIGH);
70       digitalWrite(ledPin3, HIGH);
71       digitalWrite(ledPin4, HIGH);
72     }
73
74     else if (mode == 1){
75     //Flashing
76       Serial.println("Flashing");
77
78       if (ledOn == 0){ // change to ledOn = 1
79         digitalWrite(ledPin0, HIGH);
80         digitalWrite(ledPin1, HIGH);
81         digitalWrite(ledPin2, HIGH);
82         digitalWrite(ledPin3, HIGH);
83         digitalWrite(ledPin4, HIGH);
84         ledOn = 1;
85       }
86       else if (ledOn == 1){ // change to ledOn = 0
87         digitalWrite(ledPin0, LOW);
88         digitalWrite(ledPin1, LOW);
```

```
89          digitalWrite(ledPin2, LOW);
90          digitalWrite(ledPin3, LOW);
91          digitalWrite(ledPin4, LOW);
92          ledOn = 0;
93        }
94      }
95
96      else if (mode == 2){
97      //Bouncing
98        Serial.println("Bouncing");
99
100       if (ledConfig == 0){ // change to ledConfig = 1
101         digitalWrite(ledPin0, HIGH);
102         digitalWrite(ledPin1, LOW);
103         digitalWrite(ledPin2, HIGH);
104         digitalWrite(ledPin3, LOW);
105         digitalWrite(ledPin4, HIGH);
106         ledConfig = 1;
107       }
108       else if (ledConfig == 1){ // change to ledConfig =
              0
109         digitalWrite(ledPin0, LOW);
110         digitalWrite(ledPin1, HIGH);
111         digitalWrite(ledPin2, LOW);
112         digitalWrite(ledPin3, HIGH);
113         digitalWrite(ledPin4, LOW);
114         ledConfig = 0;
115       }
116     }
117
118     else if (mode == 3){
119     //All Off
120       Serial.println("All off");
121       digitalWrite(ledPin0, LOW);
122       digitalWrite(ledPin1, LOW);
123       digitalWrite(ledPin2, LOW);
124       digitalWrite(ledPin3, LOW);
125       digitalWrite(ledPin4, LOW);
126     }
127
128     else if (mode == 4){
129       sensorVoltage = analogRead(A0);
130       Serial.println("Infrared time!");
131       Serial.println("sensorVoltage:");
132       Serial.println(sensorVoltage);
133       Serial.println("");
```

```
134
135         if (sensorVoltage <= 560 && sensorVoltage >= 400) {
136           Serial.println(sensorVoltage);
137           digitalWrite(ledPin0, HIGH);
138           digitalWrite(ledPin1, LOW);
139           digitalWrite(ledPin2, LOW);
140           digitalWrite(ledPin3, LOW);
141           digitalWrite(ledPin4, LOW);
142         }
143
144         else if (sensorVoltage <= 399 && sensorVoltage >=
              280) {
145           Serial.println(sensorVoltage);
146           digitalWrite(ledPin0, LOW);
147           digitalWrite(ledPin1, HIGH);
148           digitalWrite(ledPin2, LOW);
149           digitalWrite(ledPin3, LOW);
150           digitalWrite(ledPin4, LOW);
151         }
152
153         else if (sensorVoltage <= 279 && sensorVoltage >=
              220) {
154           Serial.println(sensorVoltage);
155           digitalWrite(ledPin0, LOW);
156           digitalWrite(ledPin1, LOW);
157           digitalWrite(ledPin2, HIGH);
158           digitalWrite(ledPin3, LOW);
159           digitalWrite(ledPin4, LOW);
160         }
161
162         else if (sensorVoltage <= 219 && sensorVoltage >=
              160) {
163           Serial.println(sensorVoltage);
164           digitalWrite(ledPin0, LOW);
165           digitalWrite(ledPin1, LOW);
166           digitalWrite(ledPin2, LOW);
167           digitalWrite(ledPin3, HIGH);
168           digitalWrite(ledPin4, LOW);
169         }
170
171         else if (sensorVoltage <= 159 && sensorVoltage >=
              140) {
172           Serial.println(sensorVoltage);
173           digitalWrite(ledPin0, LOW);
174           digitalWrite(ledPin1, LOW);
175           digitalWrite(ledPin2, LOW);
```

```
176           digitalWrite(ledPin3, LOW);
177           digitalWrite(ledPin4, HIGH);
178         }
179
180         else {
181           digitalWrite(ledPin0, LOW);
182           digitalWrite(ledPin1, LOW);
183           digitalWrite(ledPin2, LOW);
184           digitalWrite(ledPin3, LOW);
185           digitalWrite(ledPin4, LOW);
186         }
187       }
188     }
189     // at the end of this loop() iteration,
190     // oldState obtains the value of newState to maintain
          the
191     // definition that "oldState is the state of the button
192     // at the very end of the previous loop"
193     oldState = newState;
194 }
```