Isabel Serrato, Brent Usui, Jackie Zeng
9/13/21

Mini Project 1: Build Your Own Bike Light

Introduction

For this project, we created a “bike light” that can change blinking patterns when a switch is pressed. Our design includes three LEDs, and can run the following patterns:

1. All LEDs turn on
2. LEDs turning on sequentially then turning off in the same order
3. LEDs turning on and off in a bouncing fashion
4. LEDs turning on and off randomly
5. All LEDs turn off

After we implemented Part 1, our minimum viable product, we then added in an infrared distance sensor as an analog input. The distance sensor varied the delay between our lights’ blinking, causing them to switch on and off at a higher frequency when an object was moved closer to the sensor and at a slower frequency when the object was moved away. This can be used as a warning signal if a car moves too closely to the bike.

Materials

To create our blinking bike lights, we needed the following items:

- Arduino Uno R3 (x1)
- Sharp IR sensor (x1)
- Sharp IR connector (x1)
- Push button (x1)
- 93Ω resistor (x2)
- 1 kΩ resistor (x1)
- 10 kΩ resistor (x1)
- Jumper wires (x5)
- Breadboard

Arduino IDE was used to program the Arduino and the source code for the project can be seen in the Appendix.

Method

Figure 1 shows the schematic for our bike light. The Sharp IR Sensor was added for part 2 and the rest was completed for part 1.

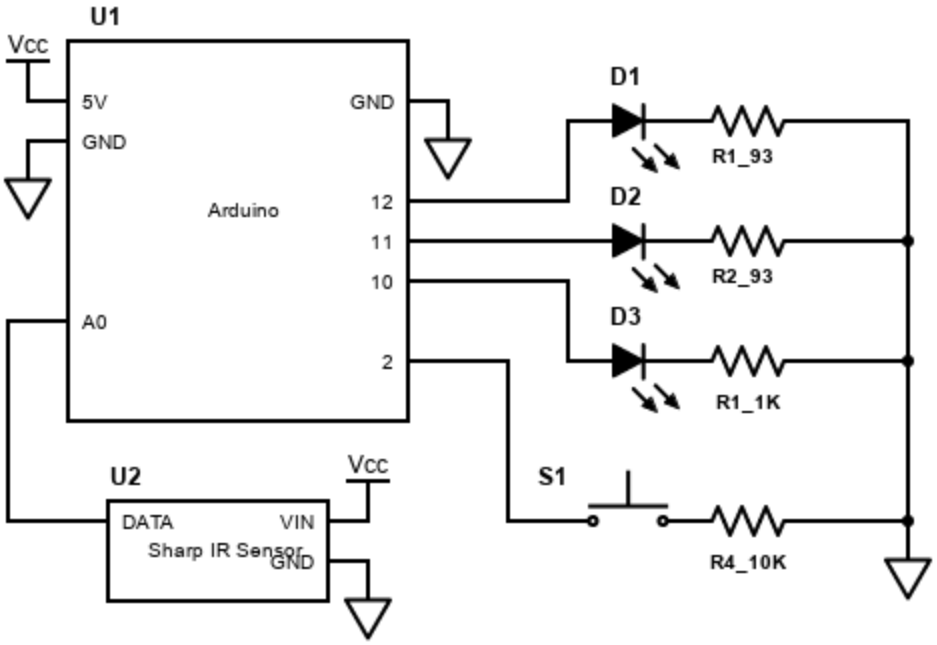

Figure 1: Schematic for mini-project 1

We arrived at our resistor values through both calculations and manual adjustments. At the time of completing the mini-project, we didn't have access to the datasheets of all the LEDs so our calculations are solely based on the red LED datasheet. The forward voltage is 2.3 V and the max forward current is 30 mA. Our LEDs are powered through the digital pins, which is 5V.

$$\frac{V_{source} - V_{LED}}{I} = \frac{5V - 2.3V}{30\ mA} = 90\Omega$$

While the resistor value calculated was 90 Ω, we chose to use a 1k Ω resistor for the D3 LED. This was for aesthetic value since the blue LED was much brighter than the other two LEDs. By increasing the resistance value for the D3 LED, we reduced the amount of current through the component, consequently reducing the brightness of the LED. We selected a 10kΩ resistor to use in series with the switch as given by the circuit diagram included with the mini-project description.

Our code is located in the Appendix. We utilized switches and interrupts in order to change the mode of our bike light. To add in the IR sensor, we also needed to include an analog pin to read the voltage value from the sensor. To vary the delay between blinks based on the Sharp IR sensor readings, we read the voltage value and subtracted it from 614, a constant that represents a voltage value of 3V according to the documentation given by Arduino.

$$\frac{1024*3V}{5V} = 614.4$$

The maximum read value is 5V, and the analog read is a 10-bit converter, so analog read values are returned as integers from 0-1023, which map to 0-5V readings. According to the datasheet of the Sharp IR sensor, the maximum output value is roughly 2.75V. In order to ensure that there will always be a slight, visible delay we approximated this to 3V, or a value of 614.

Conclusion

Following the schematic from Figure 1, we built the circuit shown in Figure 2:

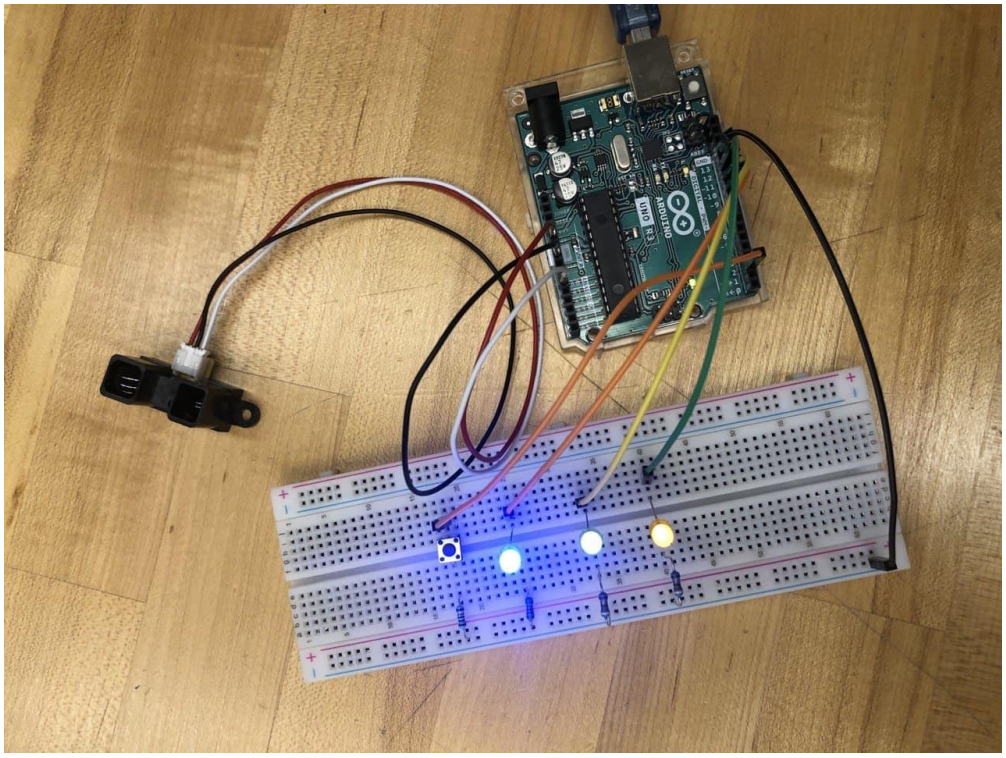

Figure 2: Built Circuit with Arduino

We completed the project and included all of our desired features successfully. Our bike light can switch through the 5 modes through a simple button press and adjusts the frequency of the flashing depending on readings from the Sharp IR sensor.

Overall, it was good to get back into circuit building and rapid prototyping with Arduinos. As we wrote the code, we got to familiarize ourselves with concepts such as software-hardware interactions, interrupts, switches, analog reading, digital writes, and the Arduino IDE interface.

During the project, we faced a complication of the arduino switching through multiple modes with a single button press. This was due to the Arduino reading a single button click multiple times. To combat the problem, we introduced a required 1000ms time delay between each button press. We implemented this time delay by saving a timestamp of the last button press and subtracting it from the current time. With the time delay, we fixed the Arduino and it stopped skipping over modes.

Appendix

Part 1

```
/*
  Project 1 Code

  Part 1 of Mini-Project 1: Build Your Own Bike Light
  Turns three LEDs on/off in different patterns depending on the mode, which can be
altered using the switch

*/

// define each of the pins we are using for the LEDs
int  LED1 = 12;
int LED2 = 11;
int LED3 = 10;
int mode = 0;

unsigned long myTime;

void setup()
{
  // initialize digital pins as outputs.
  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  pinMode(LED3, OUTPUT);

  // initialize digital pin as an input that is held high
  pinMode(2, INPUT_PULLUP);

  // initialize the start time
  myTime = millis();

  Serial.begin(9600);

  // enables interrupts to allow us to increment the mode whenever switch is
pressed
  attachInterrupt(digitalPinToInterrupt(2), update_mode, RISING);
}

void loop() {
  // LED behavior based on mode
  switch(mode) {
    case 0: // all on
```

```
      digitalWrite(LED1, HIGH);
      digitalWrite(LED2, HIGH);
      digitalWrite(LED3, HIGH);
      break;
    case 1: // all off
      digitalWrite(LED1, LOW);
      digitalWrite(LED2, LOW);
      digitalWrite(LED3, LOW);
      break;
    case 2: // sequential on
      digitalWrite(LED1, HIGH);
      delay(100);
      digitalWrite(LED2, HIGH);
      delay(100);
      digitalWrite(LED3, HIGH);
      delay(100);
      digitalWrite(LED1, LOW);
      delay(100);
      digitalWrite(LED2, LOW);
      delay(100);
      digitalWrite(LED3, LOW);
      delay(100);
      break;
    case 3: // bounce
      digitalWrite(LED1, HIGH);
      delay(100);
      digitalWrite(LED1, LOW);
      delay(100);
      digitalWrite(LED2, HIGH);
      delay(100);
      digitalWrite(LED2, LOW);
      delay(100);
      digitalWrite(LED3, HIGH);
      delay(100);
      digitalWrite(LED3, LOW);
      delay(100);
      digitalWrite(LED3, HIGH);
      delay(100);
      digitalWrite(LED3, LOW);
      delay(100);
      digitalWrite(LED2, HIGH);
      delay(100);
      digitalWrite(LED2, LOW);
      delay(100);
      digitalWrite(LED1, HIGH);
      delay(100);
      digitalWrite(LED1, LOW);
```

```
      delay(100);
      break;
    case 4: //random on
      int randint = floor(random(10,13));
      digitalWrite(randint, HIGH);
      delay(100);
      digitalWrite(randint, LOW);
      delay(100);
      break;
  }
}

void update_mode() {
  // Increment mode based on switch
  unsigned long diff = millis()- myTime;
  if (diff > 1000) {
    mode+=1;
    mode = mode % 5;
    Serial.println(mode);
    myTime = millis();
  }
}
```

Part 2

```
/*
  Project 1 Code

  Part 2 of Mini-Project 1: Build Your Own Bike Light
  Uses an infrared distance sensor to modulate the delay between our lights turning
on/off.

*/

// define each of the pins we are using for the LEDs
int  LED1 = 12;
int LED2 = 11;
int LED3 = 10;
int mode = 0;

unsigned long myTime;

// define pin for IR sensor
int sharpIR = A0;
```

```
// initialize starting values for our sensor and delay
int sensorValue = 0;
int dVal = 500;

void setup()
{
  // initialize digital pins as outputs.
  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  pinMode(LED3, OUTPUT);
  // initialize digital pin as an input that is held high
  pinMode(2, INPUT_PULLUP);

  // initialize the start time
  myTime = millis();

  Serial.begin(9600);

  // enables interrupts to allow us to increment the mode whenever switch is
pressed
  attachInterrupt(digitalPinToInterrupt(2), update_mode, RISING);
}

void loop()
{
  sensorValue = analogRead(sharpIR);
  Serial.println(sensorValue);
  // LED behavior based on mode.
  // The value of 614 tracks to a 3V reading on the analog pin, or (1024*3V)/5V, a
value slightly larger than the maximum on the datasheet (~2.75V)
  dVal = 614 - sensorValue;


  switch(mode) {
    case 0: // all on
      digitalWrite(LED1, HIGH);
      digitalWrite(LED2, HIGH);
      digitalWrite(LED3, HIGH);
      delay(dVal);
      digitalWrite(LED1, LOW);
      digitalWrite(LED2, LOW);
      digitalWrite(LED3, LOW);
      delay(dVal);
      break;
    case 1: // sequential on
      digitalWrite(LED1, HIGH);
      delay(dVal);
```

```
      digitalWrite(LED2, HIGH);
      delay(dVal);
      digitalWrite(LED3, HIGH);
      delay(dVal);
      digitalWrite(LED1, LOW);
      delay(dVal);
      digitalWrite(LED2, LOW);
      delay(dVal);
      digitalWrite(LED3, LOW);
      delay(dVal);
      break;
    case 2: // bounce
      digitalWrite(LED1, HIGH);
      delay(dVal);
      digitalWrite(LED1, LOW);
      delay(dVal);
      digitalWrite(LED2, HIGH);
      delay(dVal);
      digitalWrite(LED2, LOW);
      delay(dVal);
      digitalWrite(LED3, HIGH);
      delay(dVal);
      digitalWrite(LED3, LOW);
      delay(dVal);
      digitalWrite(LED3, HIGH);
      delay(dVal);
      digitalWrite(LED3, LOW);
      delay(dVal);
      digitalWrite(LED2, HIGH);
      delay(dVal);
      digitalWrite(LED2, LOW);
      delay(dVal);
      digitalWrite(LED1, HIGH);
      delay(dVal);
      digitalWrite(LED1, LOW);
      delay(dVal);
      break;
    case 3: // random on
      int randint = floor(random(10,13));
      digitalWrite(randint, HIGH);
      delay(dVal);
      digitalWrite(randint, LOW);
      delay(dVal);
      break;
    case 4: // all off
      digitalWrite(LED1, LOW);
      digitalWrite(LED2, LOW);
```

```
      digitalWrite(LED3, LOW);
      break;
  }
}

void update_mode() {
  // Increment mode based on switch. diff ensures the switch does not happen if the
sensor reads a change withing one second of the last change
  unsigned long diff = millis()- myTime;
  if (diff > 1000) {
    mode+=1;
    mode = mode % 5;
    Serial.println(mode);
    myTime = millis();
  }
}
```