

Color and Blink Controlled RGB LED

Engr 102 intro to electronics

HALIMA, GUSTI.

Fall 2017

­**Description of the problem and reason for building the circuit**

This project will ­demonstrate how two different sensor inputs can control an output component. The following circuit uses a thermistor to control the colors of an RGB LED and uses a heart rate sensor to control the blink rate of the RGB LED. I chose to build this circuit as a simple first step project in learning how to create a medical device.

**Things to do with the circuit**

Placing a finger on the chip of the heart rate sensor will control the blink of the RGB LED with delay according to the person’s heartbeat. Touching the thermistor will either cause it to increase or decrease the temperature which will then change the color of the RGB LED. A brighter yellow indicates high temperature while a dimmed red indicates low temperature.

**Prior knowledge needed to create this circuit**

Other than basic knowledge for creating a circuit with a blinking LED and controlling the colors from a RGB LED, I needed to know which sensors are suitable to take a person’s heart rate and a sensor that is responsive for a small change in temperature.

**Parts Required:**

1. Sparkfun / Arduino Board
2. Breadboard
3. RGB LED
4. Thermistor (analog temperature sensor)
5. Heart rate sensor (Infrared LED and phototransistor)
6. 2 330Ω resistor

**Problem-Solving Approach**

To build the circuit, I initially had to determine what sensors are suitable. After knowing the parts that I needed, the rest had to do with creating and debugging the code.

**Solved Problems**

1. Circuit Lag: At first, the color change in the RGB LED seemed to lag. After checking the values in the serial print, it turns out that the sensor wasn’t too responsive to a small temperature change. This problem was solved by changing the initial Temp Sensor to a Thermistor.
2. Determining the color range of the RGB LED: The most time-consuming part of this project had been to figure out how I could consistently show the color change on the RGB LED without calibration. This was solved by balancing between values of the red and green LED. I gave an initial large value for red and a zero value for green. As the temperature raises, a certain amount of value is subtracted from the red value then the same value is added to the green value.

**Schematic & PCB**

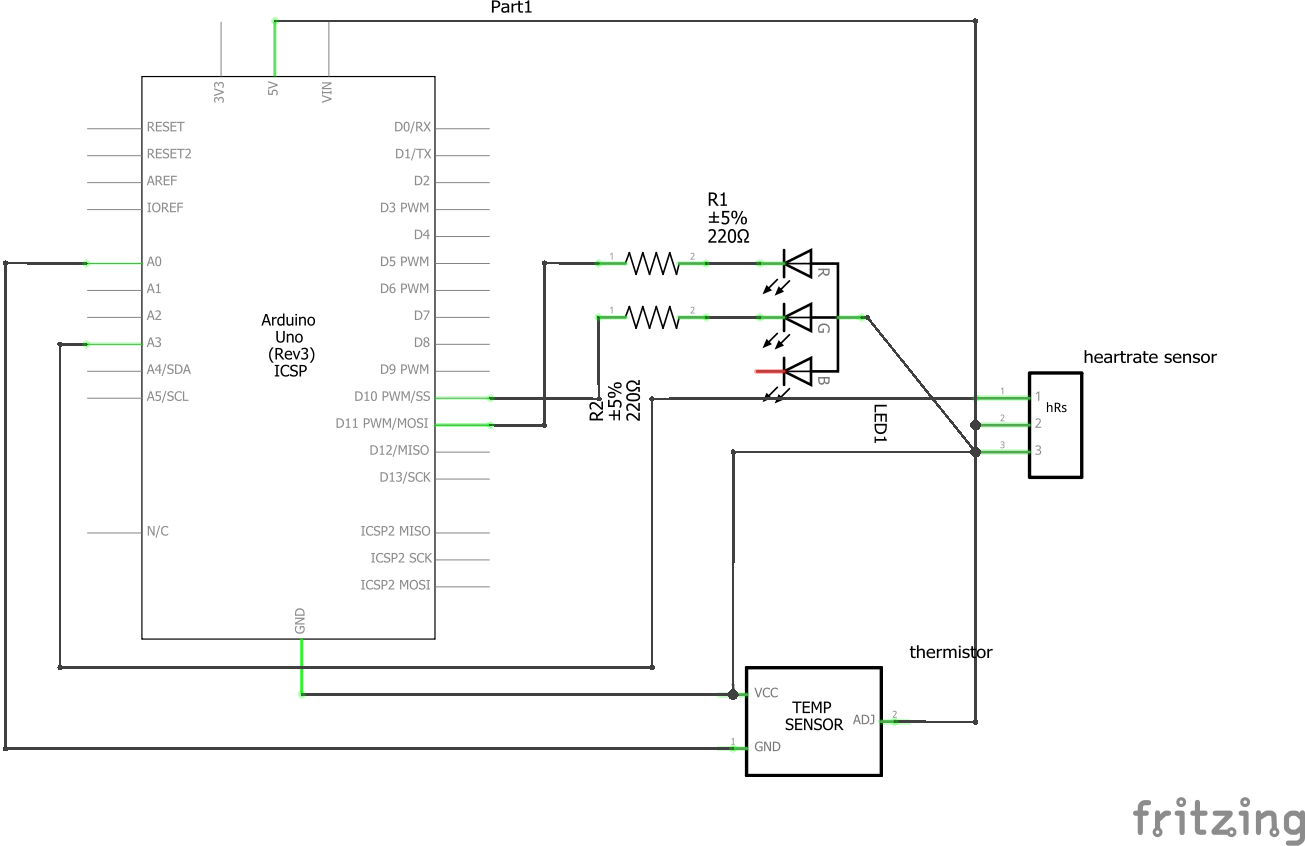


Figure . Schematic of circuit

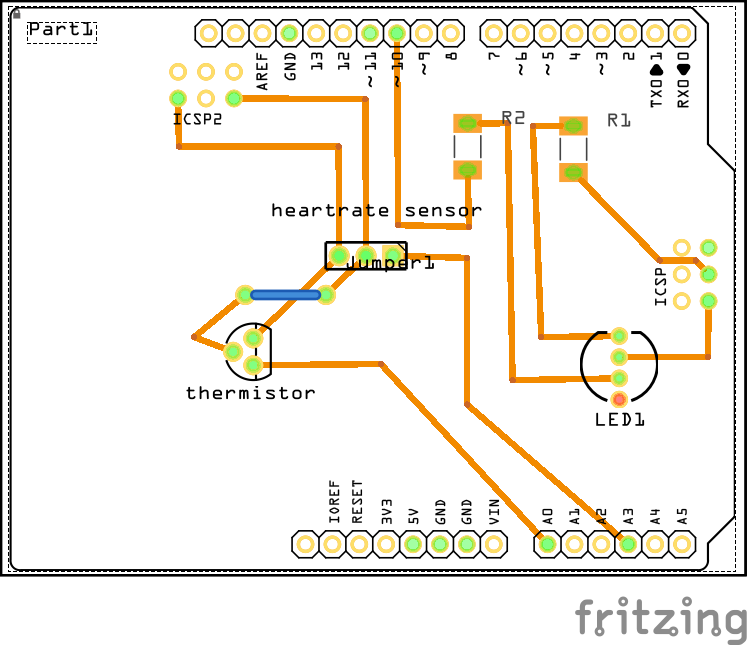


Figure . circuit pcb

**Sample Outputs**

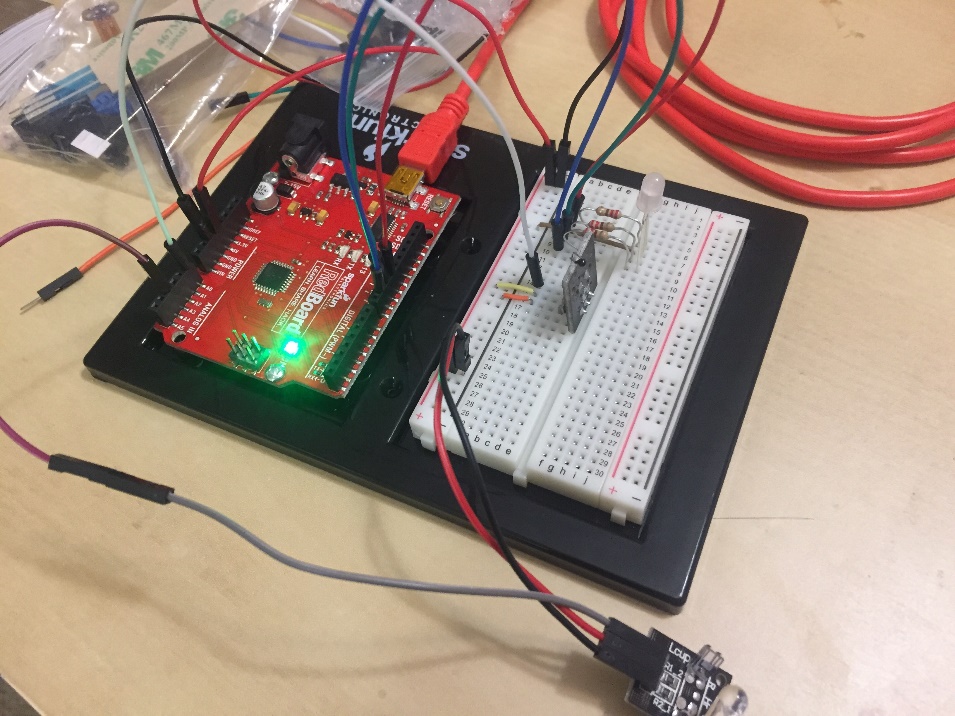


Figure . circuit turned off/circuit LED is off

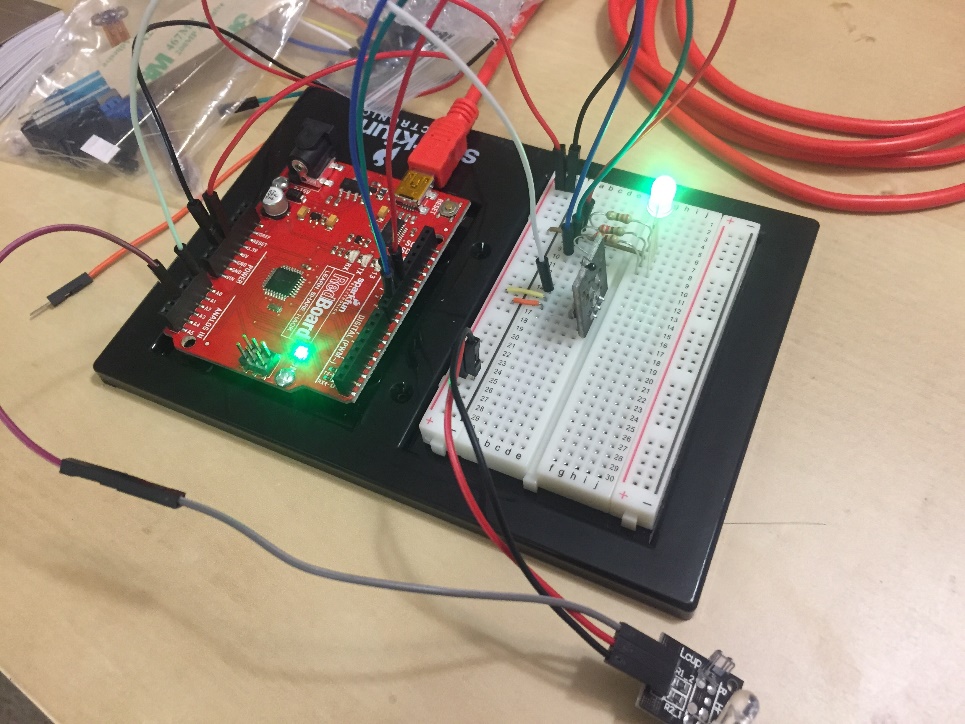


Figure . LED is on, color is green based on temperature

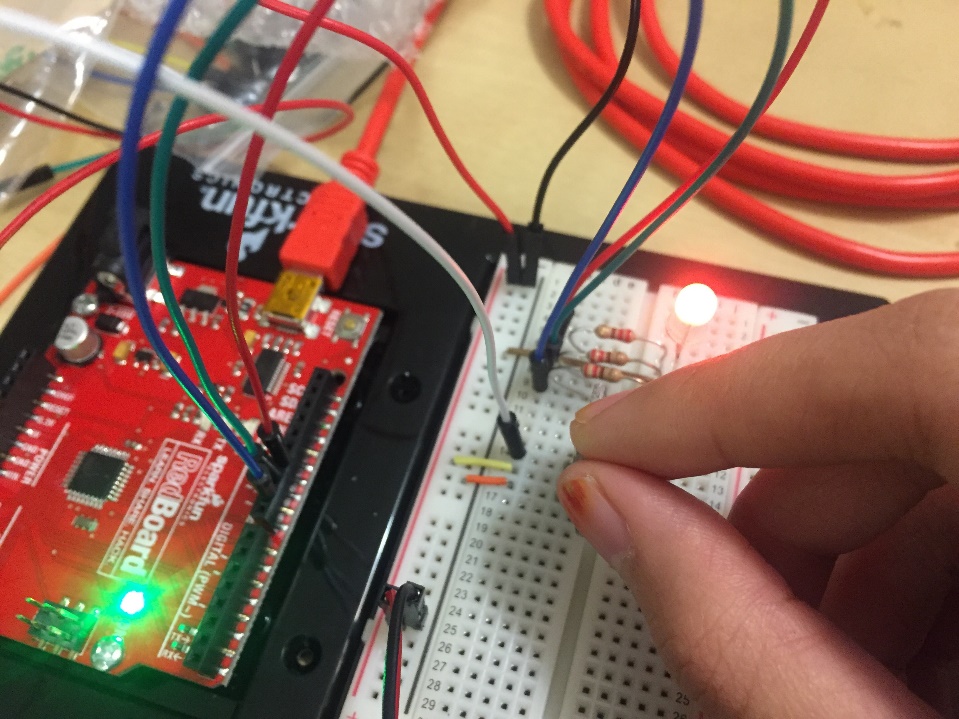


Figure . LED is on, color turned red based on an increase in temperature

**Code:**

1. /\*
2. \* Name: Gusti Scarlett Halima
3. \* Date: 23rd October 2017
4. \* class: Engr 102
5. \* Software version: Arduino 1.8.2
6. \* What it does: The circuit has an RGB LED that can change color based on the temperature
7. \* from a thermistor and have a blink rate based on the heartrate sensor.
8. \*/
10. #include <Math.h>
12. //Setting constants for pins
13. **const** **int** temperaturePin = A0;
14. **const** **int** heartRatePin = A3;
16. /\*
17. \* The RGB LED only uses the red and green LED
18. \* therefore we don't need to connect the blue LED
19. \*/
20. **const** **int** RED\_PIN = 11;
21. **const** **int** GREEN\_PIN = 10;
23. **void** setup() {
24. Serial.begin(9600);
26. pinMode(RED\_PIN, OUTPUT);
27. pinMode(GREEN\_PIN, OUTPUT);
29. }
31. **void** loop() {
33. /\*
34. \* Variable 'getValue' will read the raw value from the analog pin.
35. \* To see the change in temperature, the red and green LED values will
36. \* balance out each other by taking the tenths place from 'getValue'.
37. \* Red LED will have an initial value of 300 while the Green LED is 0.
38. \* While the tenths place will subtract the Red LED value, it will add the
39. \* Green LED value.
40. \*/
41. **int** getValue = analogRead(temperaturePin);
42. **int** subtract = getValue % 100;
43. **int** red = 300 - subtract;
44. **int** green = subtract;
46. /\*
47. \* variable 'heartRate' will receive its raw value from the heart rate sensor
48. \*/
49. **int** heartRate = analogRead(heartRatePin);
51. /\*
52. \* The following code will read the values in the serial print
53. \*/
54. Serial.print("Raw temperature sensor value: ");
55. Serial.println(getValue);
56. Serial.print("Red LED value: ");
57. Serial.println(red);
58. Serial.print("Green LED value: ");
59. Serial.println(green);
60. Serial.print("Raw heart rate sensor value: ");
61. Serial.println(heartRate);
63. /\*
64. \* The values of red, green, and heartRate are passed in the
65. \* following method to display the color and blink rate
66. \*/
67. colorAndBlink(red, green, heartRate);
68. }
70. /\*
71. \* Method colorAndBlink displays a single loop of the LEDs being turned
72. \* on then off with the heartRate value as the delay
73. \*/
74. **void** colorAndBlink(**int** r, **int** g, **int** hR) {
75. setColor(r, g);
76. delay(hR);
77. setColor(0, 0);
78. delay(hR);
79. }
81. /\*
82. \* Method setColor sets the color of the LED
83. \*/
84. **void** setColor(**int** r, **int** g) {
85. analogWrite(RED\_PIN, r);
86. analogWrite(GREEN\_PIN, g);
87. }

Video: <https://youtu.be/m7LPpep2z9Q>

Syntax Highlighter used:<http://www.planetb.ca/syntax-highlight-word>

Schematic created using Fritzing