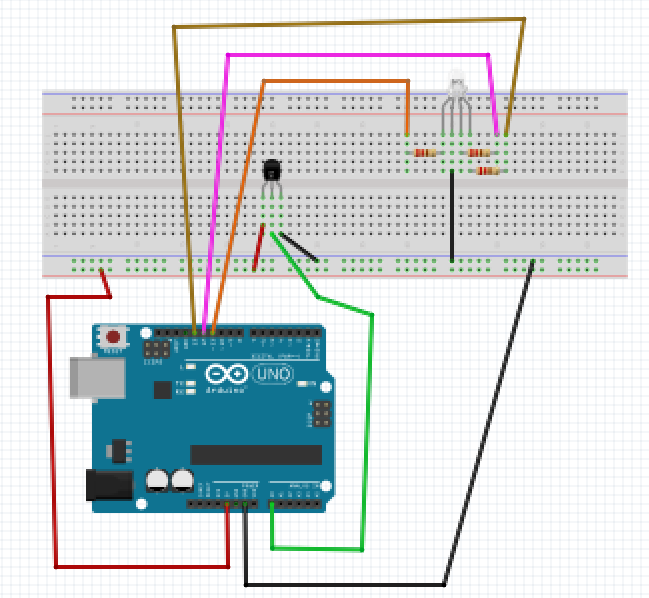
**ARDUINO LED PROJECT**

**RGB LED with Temperature sensor**



For this project we will need:

* Arduino board.
* RGB led
* Breadboard.
* 7 Male to Male wires.
* 3 470 Ohm resistor
* Temperature sensor

Circuit Design:

1. First make sure that the Arduino is powered off (no USB cable plugged to power).
2. Check the temperature sensor, identify the 3 legs on the pin. Identify the flat part of the resin. Observe orientation. Facing the flat part is the orientation used in subsequent steps.
3. Plug the center pin of the temperature sensor to a breadboard horizontal line.. Using a green male jumper wire, connect to the Arduino via pin A0.
4. Plug the left pin of the sensor to the breadboard on a separate horizontal line. Using a red jumper wire connect to the Arduino via pin 5V.
5. Plug the right pin of the sensor to the breadboard on a separate horizontal line. Using a black jumper wire connect to the common ground of the breadboard.
6. Check the LED, you will see that one of the leg is longer than the other 3.
7. Plug the longer leg of the LED to a hole on the breadboard. Connect that leg to a common ground of the breadboard (vertical blue line), using a black cable if possible (convention for GND).
8. Plug the 1st short leg of the LED to a different hole, on a different and independent line of the breadboard.
9. Add a 470 Ohm resistor between this longer leg and pin 9(has ~ ) of the Arduino, using an additional colored wire (no red, no black) for convenience.
10. Plug the 3rd short leg of the LED to a different hole, on a different and independent line of the breadboard.
11. Add a 470 Ohm resistor between this longer leg and pin 10(has ~ ) of the Arduino, using an additional colored wire (no red, no black) for convenience.
12. Plug the 4th short leg of the LED to a different hole, on a different and independent line of the breadboard.
13. Add a 220/470 Ohm resistor between this longer leg and pin 11(has ~ ) of the Arduino, using an additional colored wire (no red, no black) for convenience.
14. Plug the common ground of the breadboard to the ground pin of Arduino.

|  |
| --- |
| int redPin=9;  int greenPin=10;  int bluePin=11;  int tempPin=A0;  float temp;  void setup(){  pinMode(redPin,OUTPUT);  pinMode(greenPin,OUTPUT);  pinMode(bluePin,OUTPUT);  pinMode(tempPin, INPUT);  Serial.begin(9600);  }  void loop(){  temp=analogRead(tempPin);  temp=(5.0\*temp\*1000.0)/(1024\*10);    if(temp>=24.0 && temp <=25.5){  setColor(255,0,0);  Serial.println(temp);  delay(1000);  }  else if(temp>=25.6 && temp <=26.5){  setColor(0,255,0);  Serial.println(temp);  delay(1000);  }  else if(temp>=26.6 && temp <=27.5){  setColor(0,0,255);  Serial.println(temp);  delay(1000);  }  }  void setColor(int red, int green, int blue){  analogWrite(redPin ,blue);  analogWrite(greenPin ,green);  analogWrite(bluePin ,blue);  }  } |

int redPin=9;

int greenPin=10;

int bluePin=11;

We instruct the Arduino to use these values as the pins attached to the external LED.The pins are attached to pin 1,3,4.

int tempPin=A0;

float temp;

This assigns the pin attached to the middle pin of the temperature sensor. This is an analog pin.

It accepts values that are continuous .

pinMode (redPin,OUTPUT);

pinMode (greenPin, OUTPUT);

pinMode (bluePin, OUTPUT);

This sets the LED pins functionality to output. Orientation of the LED helps to determine color changes.

pinMode(tempPin, INPUT);

After the execution of this line, the analog pin A0 will be set as input, and this will enable us to get information from it.

temp=analogRead(tempPin);

The pin we selected allows us to receive analog signals.

temp=(5\*temp\*1000.0)/(1024\*10);

This converts the 10 bit number received from sensor to a voltage between 0 and 5v.

This value is then multiplied by 1000 to convert the value to miliVolts

We then divide the resulting value by 10 because each degree rise results in a 10mV increase.

if(temp>=24.0 && temp <=25.5

else if(temp>=25.6 && temp <=26.5)

else if(temp>=26.6 && temp <=27.5)

These statements set conditions to be checked.

They take the value temp and compare to the ranges set. This can be modified as required.

setColor(0,0,255);

The pins attached to 1,3,4 of the led are unique. They allow us to output analog signals.

The values passed are in the range of 0-255.This will vary the brightness of the specific color.

These particular sets of instructions are likely to be repeated during execution.

Instead of rewriting them we can create a function that will execute this set of instructions.

Serial.println(temp);

This reads the value stored in tempValue and prints to the serial monitor.

This value is checked every 500 ms/5s.

This value represents the temperature at the given moment.’

delay(500);

This determines after how long the temp measurements will be taken. The value passed is measured in milliseconds.

void setColor(int red, int green, int blue){

This creates a function.

The function receives 3 values red, green ,blue.

void shows that this function does not return any value when executed.

Same with the setup(),loop() functions they do not return any values when executed.

analogWrite(redPin,blue);

analogWrite(greenPin,green);

analogWrite(bluePin,blue);

This assigns the pins of the LED to a particular brightness on corresponding pins.

This combination of LED brightness allows us to modify the color displayed.

This pin has PWM allowing it to write both digital and analog signals.

This sends the brightness value to the attached LED pin.

void setup (){}

This initializes the arduino and assigns functionality to its pins.

This also provides required resources for monitoring.

void loop(){}

After executing the void setup() function, we enter the void loop() and this function is executed continuously and repeatedly, until your Arduino is powered off.