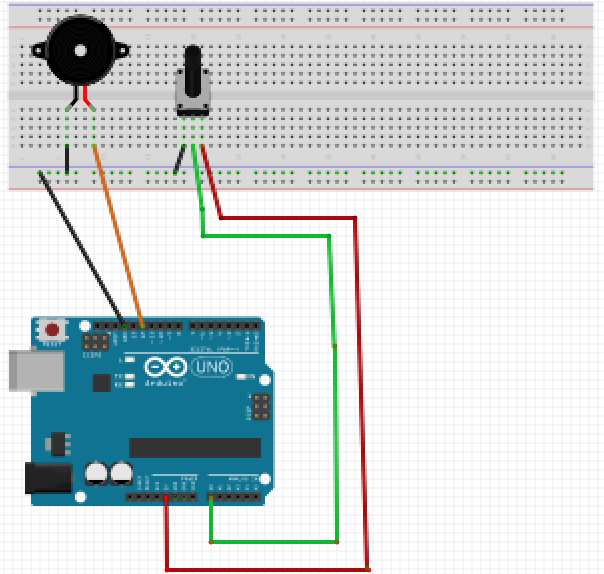
**ARDUINO LED PROJECT**

**Potentiometer to control buzzer tone.**

For this project we will need:

* Arduino board.
* Buzzer.
* Potentiometer
* Breadboard.
* 470 Ohm resistor.
* 6 Male to Male wires.

Circuit Design:

1. First make sure that the Arduino is powered off (no USB cable plugged to power).
2. Check the Buzzer, you will see that one of the leg is shorter than the other one.
3. Plug the longer leg of the Buzzer(anode) to a horizontal line on the breadboard. Connect using an orange wire from this line to pin 10 of the arduino.
4. Plug the shorter leg of the Buzzer(cathode) to the breadboard. Separate horizontal line. Plug to the common ground(vertical blue(-) line) pin of the breadboard using a black male jumper wire.
5. Identify the 3 pins on the rotary potentiometer.
6. Plug the center pin of the rotary potentiometer to the breadboard. Take a green male jumper wire to pin A0 of the Arduino.
7. Plug the left pin of the rotary potentiometer to the breadboard. Take a black male jumper wire from the line connected to the left pin, to the ground common ground of the breadboard (marked by a blue (-) line
8. Plug the right pin of the rotary potentiometer to the breadboard. Take a red male jumper wire from the line connected from the line connected to the Right pin, to the 5V pin of the Arduino.
9. Plug the common ground of the breadboard to the GND pin of the arduino using a black male jumper wire.

.

|  |
| --- |
| const int buzzerPin=10; // declare ARDUINO pin connected to Buzzer anode  const int potPin = A0; // pin attached to potentiometer  void setup(){  pinMode(buzzerPin, OUTPUT);  pinMode(potPin,INPUT);  }  void loop(){  int frequencyValue = analogRead(pitchValue);  int frequency = map(frequencyValue,0,1023,100,5000);  int duration= 250;  tone(buzzerPin,frequency,duration);  delay(1000);  } |

const int buzzerPin = 10;

const int potPin = 0;

First, we create a reference for the analog pin we intend to use and define a variable for that pin number. This will enable us to reference the pin by that variable name buzzerPin instead of the hard-coded number. In subsequent designs, if you need to use a different analog pin (for example pin 11), then you just need to change the number here and it will update it everywhere in your design program.

We then declare the pin which will provide the external input used to modify the tone. This will be our potentiometer reference pin.

const shows that the value assigned doesn’t change during program execution

int shows the value is a number. Values declared without the keyword const can be modified later in the program execution.

pinMode(buzzerPin, OUTPUT);

After the execution of this line, the digital pin 10 will be set as output, and this will enable us to send information to it and control the LED.

int frequencyValue = analogRead(pitchValue);

This receives a value from the potentiometer. The value is in the range of 0-1023 since the position of the rotating dial can give 1023 levels of resistance.

int frequency=map(frequencyValue,0,1023,100,5000);

This converts the value of frequencyValue from a range of 0-1023 to 100-5000Mhz.This range is used to give the pitch of the tone to be played. This refers to the number of times it can be played per unit time.

tone(buzzerPin,frequency,duration);

This is a predefined function that takes in 3 values.

The first value represents the device we wish to use to play the sound.

The second value represents the frequency of the tone.

The third value represents the duration to play the tone.

delay(1000);

This determines how long the tone will not be playing .Allows user to differentiate the tone interval.After this duration the program will begin executing again. The value passed is measured in milliseconds.

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 you Arduino is powered off.