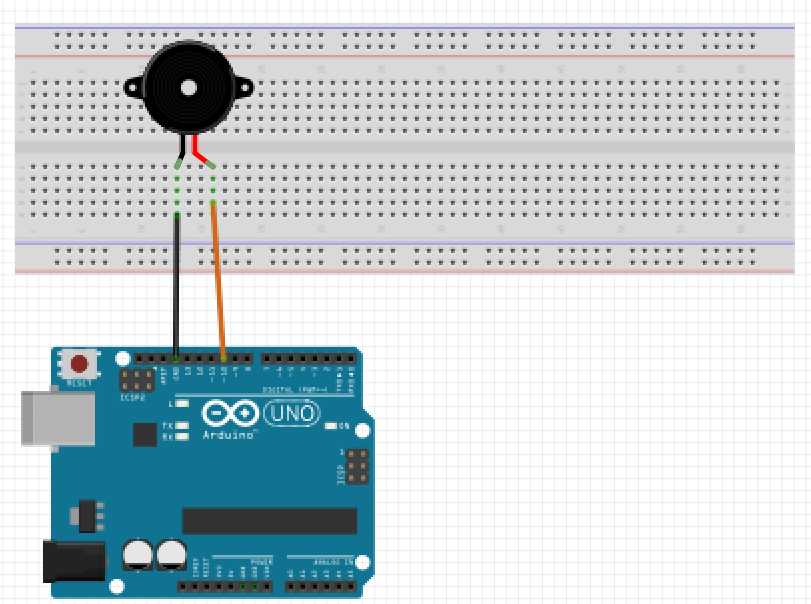
**ARDUINO LED PROJECT**

**Buzzer**

For this project we will need:

* Arduino board.
* Buzzer.
* Breadboard.
* 470 Ohm resistor.
* 3 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 12 of the arduino.
4. Plug the shorter leg of the Buzzer(cathode) to the breadboard. Separate horizontal line. Plug to the ground pin of the arduino using a black male jumper wire.

.

|  |
| --- |
| const int buzzerPin=10; // declare ARDUINO pin connected to Buzzer anode  const int pitchValue = 0;  void setup(){  pinMode(buzzerPin, OUTPUT);  }  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 pitchValue = 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 value of pitch since we will not be using an external source to control it

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 previously initialized, and permanently stored in memory.Since we are not relying on an external input to modify this property this works.

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.