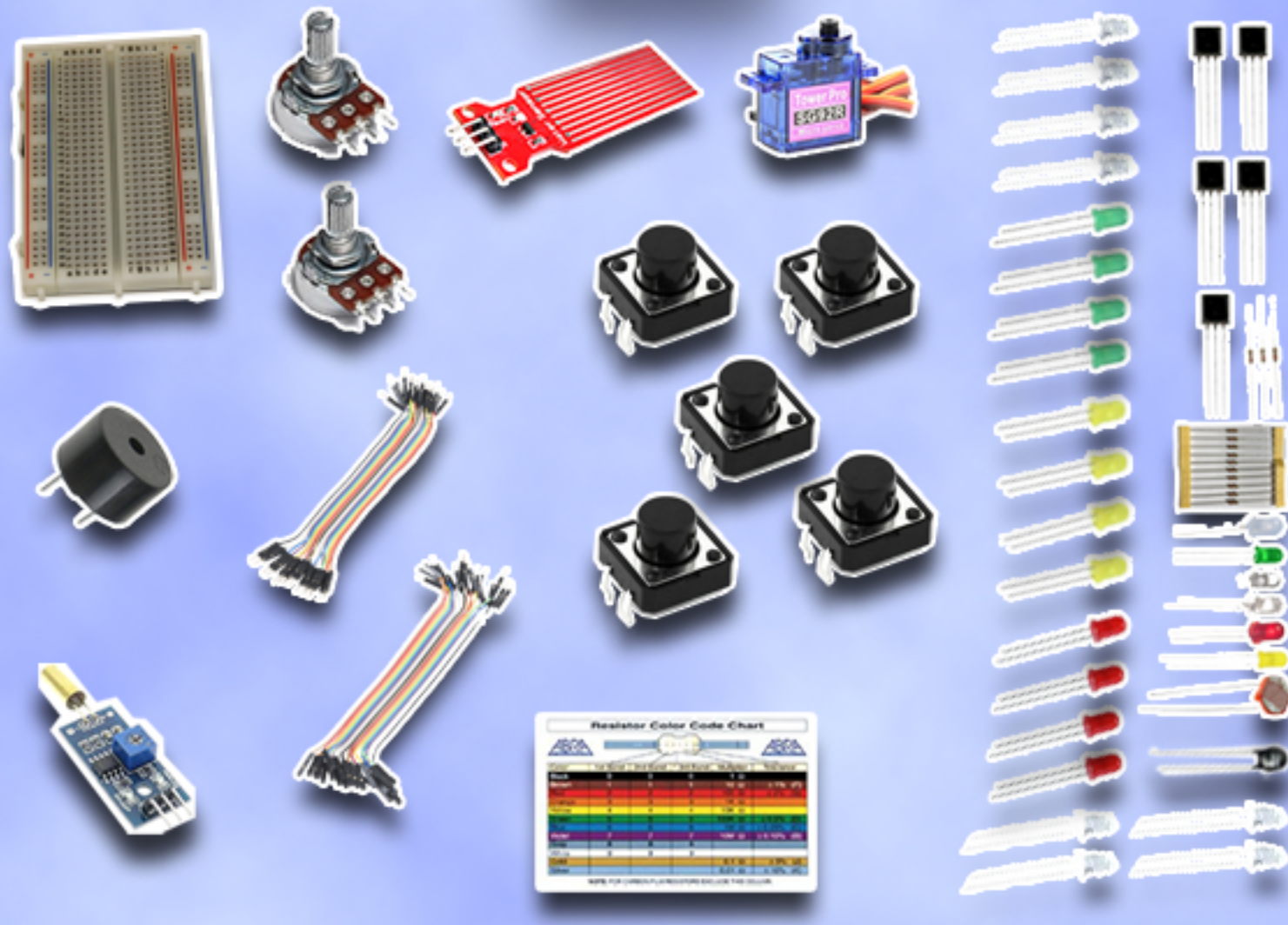
ARDUINO MINIMALIST KIT



ARD-MIN-KIT

ABRA

www.abra-electronics.com

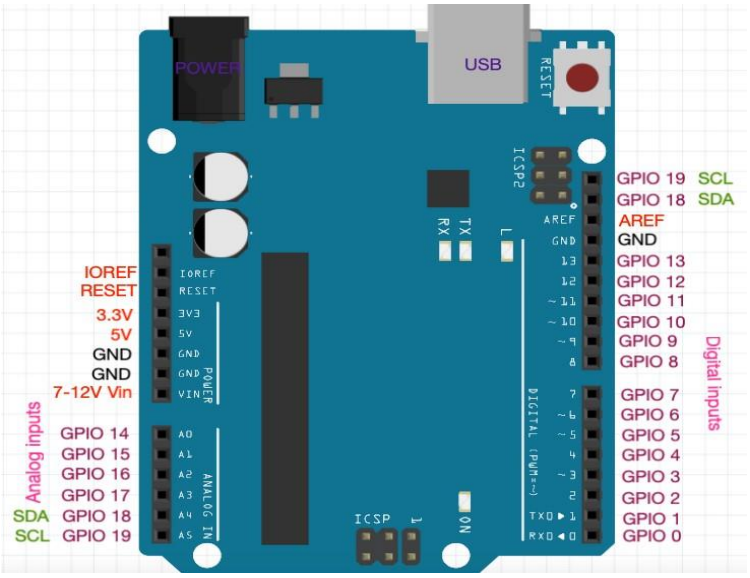
TEL: 1-800-361-5237

S.no	Name of the Component	Part number	Quantity
1	Active Buzzer	BUZ-120	1
2	Breadboard	ABRA-6	1
3	Tilt Sensor Module	SENS-39	1
4	10K Thermistor	334-103	1
5	Small Button Switch	PBS-315BK	5
6	Photoresistor	PHOTO-300	2
7	NPN Transistor	S8050	5
8	Servo Motor	SG90	1
9	Water Level Sensor Module	SENS-78	1
10	1N4007 Diode	1N4007	1
11	Resistors (220,330,1K,10K,100K,1M)		60
12	5mm LED (Red, Green, Yellow, Blue, White)		25 (5 each)
13	RGB LED	LED-5RGB-4-CC	1
14	Potentiometer	P10K-MIN-PC	2
15	Female to Male	JW-MF-20-6	1
16	Male to Male	JW-MM-20-6	1
17	Resistor card		1
18	Manual		1

Contents

1 Introduction to Arduino

Arduino Uno is a microcontroller board based on the ATmega328P.



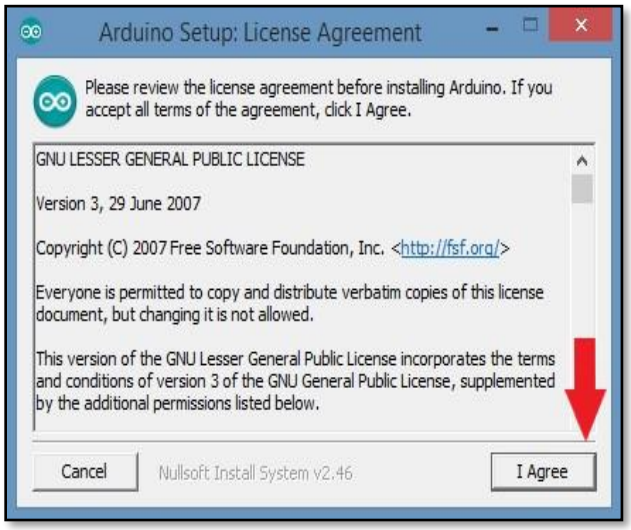
A0-A5: Analog data such as sensor photoresistor are read by these pins and Analog to digital converter (ADC) will convert the data into digital.

1.1 Installing Arduino

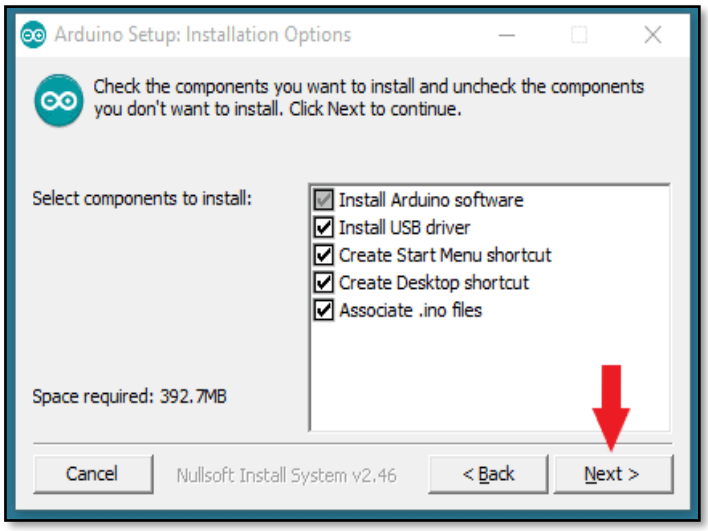
Step 1: Visit https://www.arduino.cc/. In software section select Downloads.

Step 2: Depending on the operating system in your computer select Arduino IDE accordingly. If windows, click on the installer instead of ZIP file as it is easy to install.

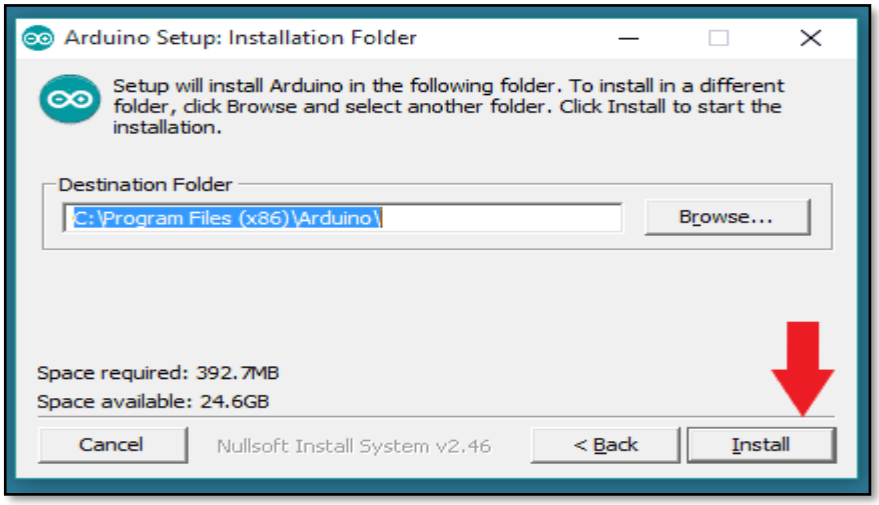
Step 3: Click **I agree**



Step 4: Click **Next**

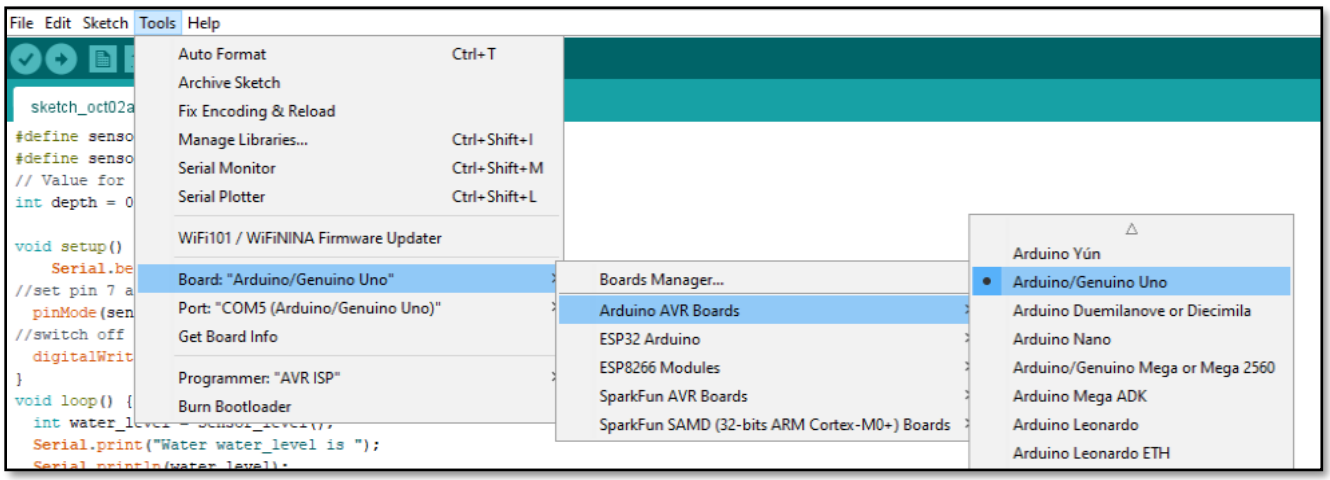


Step 5: Select the path where the Arduino IDE should be installed. By default it is C drive but you always have the option to browse and select your own path.

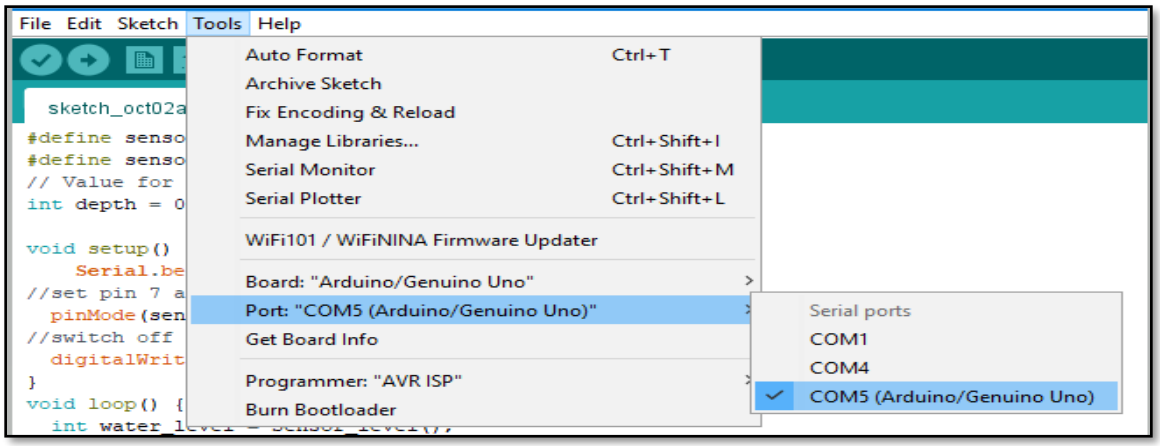


1.2 Getting started with IDE

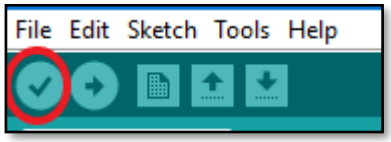
Open the IDE and connect your Arduino board to your computer. Before writing your code make sure you have selected the board. **Tools->Board->Arduino AVR boards-> Arduino/Genuino Uno**



Now select the port for the Arduino board. In this example its **Com 5 (Arduino/Genuino Uno).**



Now write your code and Compile and uploads the code to the board.

To compile your code click on the tick mark button shown in the 

figure

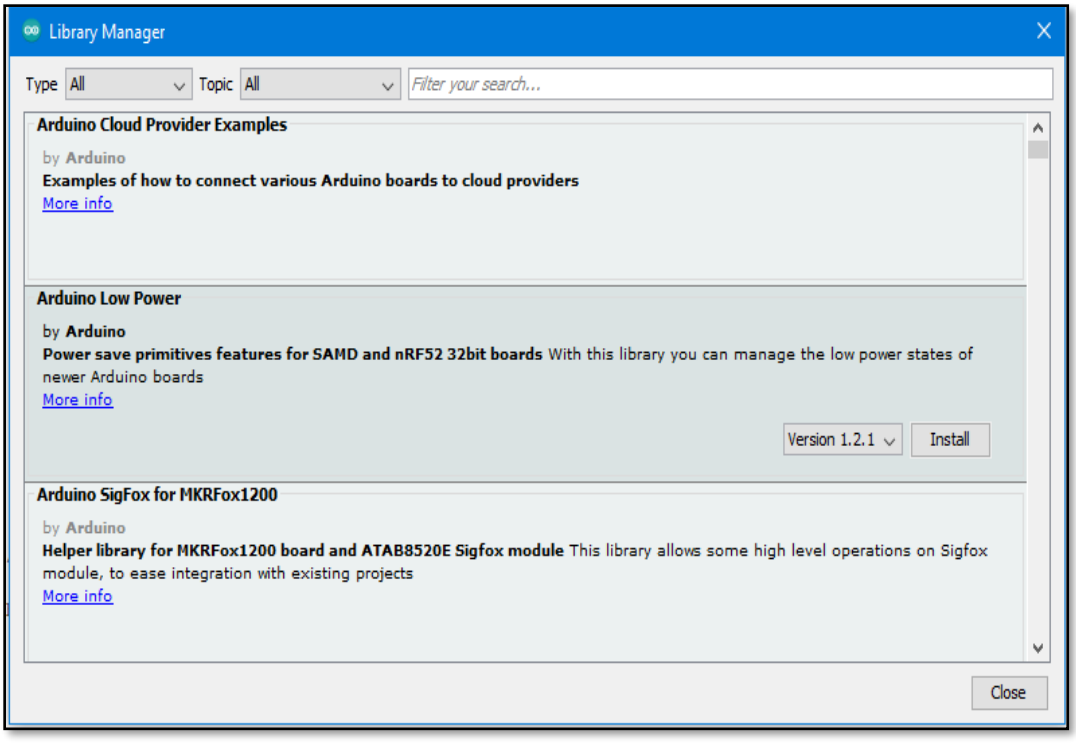
To upload the code select **Sketch-> upload**

1.3 Adding Libraries

The Arduino environment can be extended using libraries, just like most programming platforms. Libraries provide extra functionality for use in sketches, e.g. working with hardware or manipulating data. To use a library in a sketch, select it from Sketch > Import Library.

How to Install a Library

- *Using the Library Manager:* To install a new library into your Arduino IDE you can use the Library Manager. Open the IDE and click to the "Sketch" menu and then ***Include Library > Manage Libraries.*** The search for the required library you are looking for and Click install.



x

- *Importing a .zip Library:* Libraries are often distributed as a ZIP file or folder. The name of the folder is the name of the library. Inside the folder will be a .cpp file, a .h file and often a keywords.txt file, examples folder, and other files required by the library. Starting with version 1.0.5, you can install 3rd party libraries in the IDE. Do not unzip the downloaded library, leave it as is. In the Arduino IDE, navigate to **Sketch > Include *Library > Add .ZIP Library.*** You will be prompted to select the library you would like to add. Navigate to the .zip file's location and open it.

1.4 Code Basics

Certain basic functions that makes coding in Arduino easier are listed below.

Digital I/O for digital pins

- digitalRead() : Reads the value from a specified digital pin, either HIGH or LOW.
  Syntax: digitalRead(pin)
- digitalWrite() : Write a HIGH or a LOW value to a digital pin. If the pin has been configured as an OUTPUT with pinMode(), its voltage will be set to the corresponding value: 5V (or 3.3V on 3.3V boards) for HIGH, 0V (ground) for LOW. If the pin is configured as an `INPUT`, `digitalWrite()` will enable (`HIGH`) or disable (`LOW`) the internal pullup on the input pin.
  Syntax: digitalWrite (pin,value)

- pinMode() : Configures the specified pin to behave either as an input or an output.
  Syntax: pinMode(pin, mode).
  pin: the Arduino pin number to set the mode of. Mode is either INPUT or OUTPUT

```
int ledPin = 12;  // LED connected to digital pin 13
int inPin = 7;    // pushbutton connected to digital pin 7
int val = 0;      // variable to store the read value

void setup() {
  pinMode(ledPin, OUTPUT);  // sets the digital pin 12 as output
  pinMode(inPin, INPUT);    // sets the digital pin 7 as input
  pinMode(13, OUTPUT);    // sets the digital pin 13 as output

}
void loop() {
  val = digitalRead(inPin);   // read the input pin
  digitalWrite(ledPin, val);  // sets the LED to the button's value
  digitalWrite(13, HIGH); // sets the digital pin 13 on
  delay(1000);            // waits for a second
  digitalWrite(13, LOW);  // sets the digital pin 13 off
  delay(1000);            // waits for a second
}
```

Analog I/O for analog pins

- analogRead() : Reads the value from the specified analog pin
  Syntax: analogRead(pin)
  Example: digitalread(10); //reads the value either LOW or HIGH in pin number 10

- analogWrite() : Writes an analog value (PWM wave) to a pin. Can be used to light a LED at varying brightnesses or drive a motor at various speeds.

Syntax: analogWrite(pin, value)

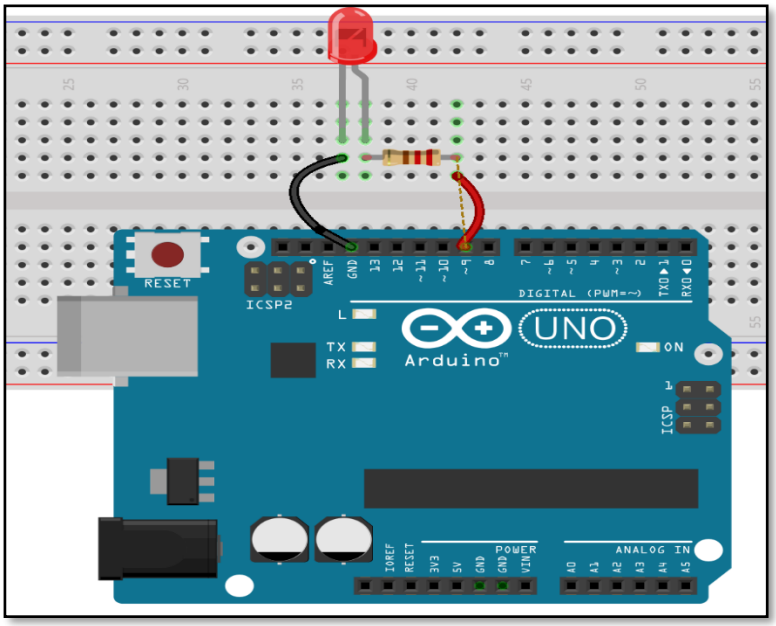
```
int ledPin = 9;       // LED connected to digital pin 9
int analogPin = 3;    // potentiometer connected to analog
pin 3
int val = 0;          // variable to store the read value

void setup() {
  pinMode(ledPin, OUTPUT);  // sets the pin as output
}

void loop() {
  val = analogRead(analogPin);  // read the input pin
  analogWrite(ledPin, val / 4); // analogRead values go
from 0 to 1023, analogWrite values from 0 to 255
}
```

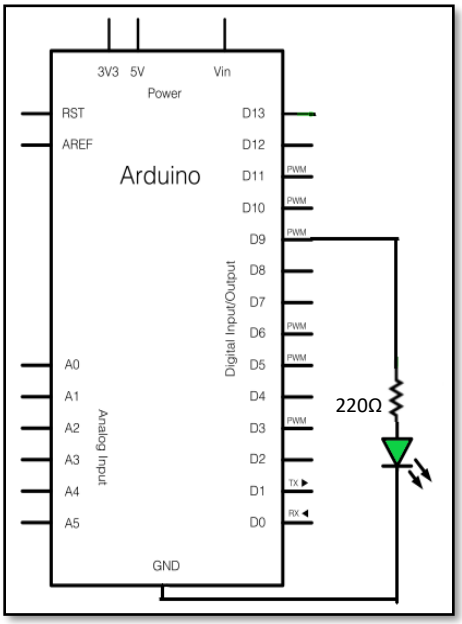
2 Projects

2.1 LED blink

Brief: Through this project we shall learn how to turn an LED on and off. If the pin has been configured as an OUTPUT with pinMode(), its voltage will be set to the corresponding value for digitalWrite: 5V for HIGH, 0V (ground) for LOW.

Components: LED, RESISTOR 220Ω

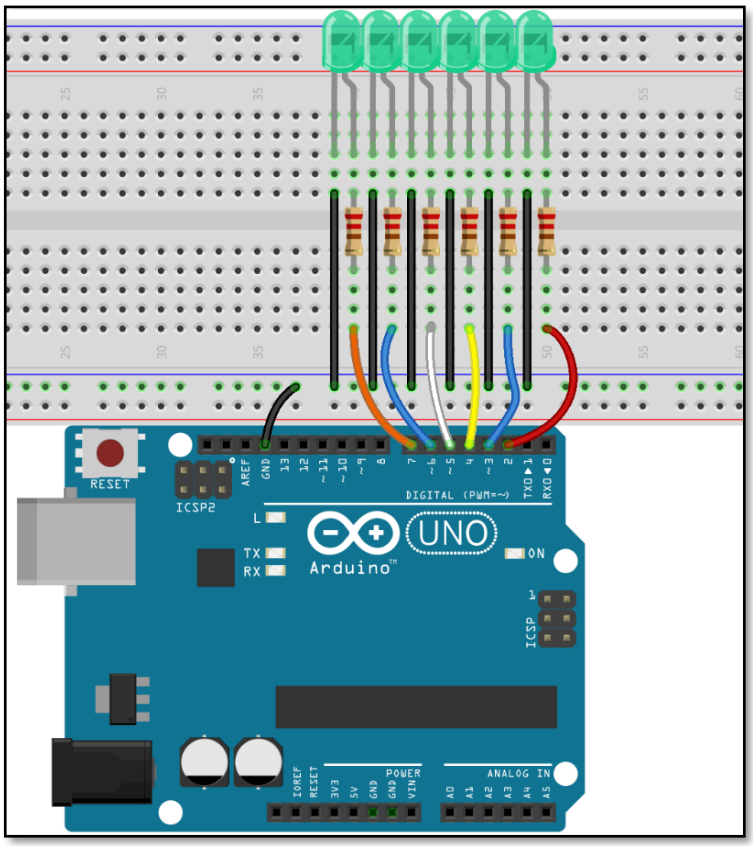


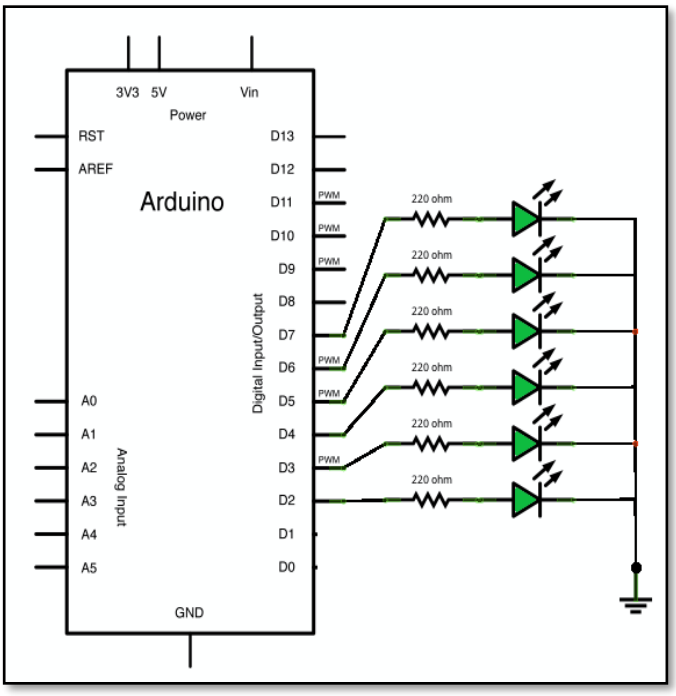


```
const int ledPin = 9;            //the Pin number of the on-board LED
void setup ()
{
pinMode (ledPin, OUTPUT);  //initialize the digital pin 9 as an output
}
void loop ()       //the loop routine runs repeatedly forever
{
digitalWrite(ledPin,HIGH);                  //turn the LED on
delay(500);                 //wait for half a second
digitalWrite(ledPin,LOW);                   //turn the LED off
delay(500);                  //wait for half a second
}
```

2.2 LED Trailing effect

Brief: Through this project we shall learn how to turn on the LED one by one using Arduino. We here use for loop to activate the LEDs connected to pin 2, 3,4,5,6,7 one after the other with a delay 500 i.e. half a second.
Components: LED, 220Ω resistors



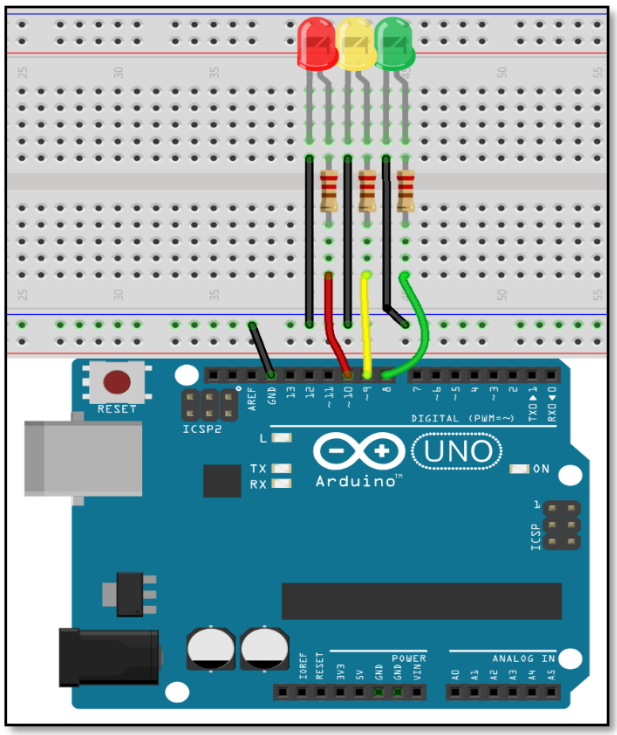


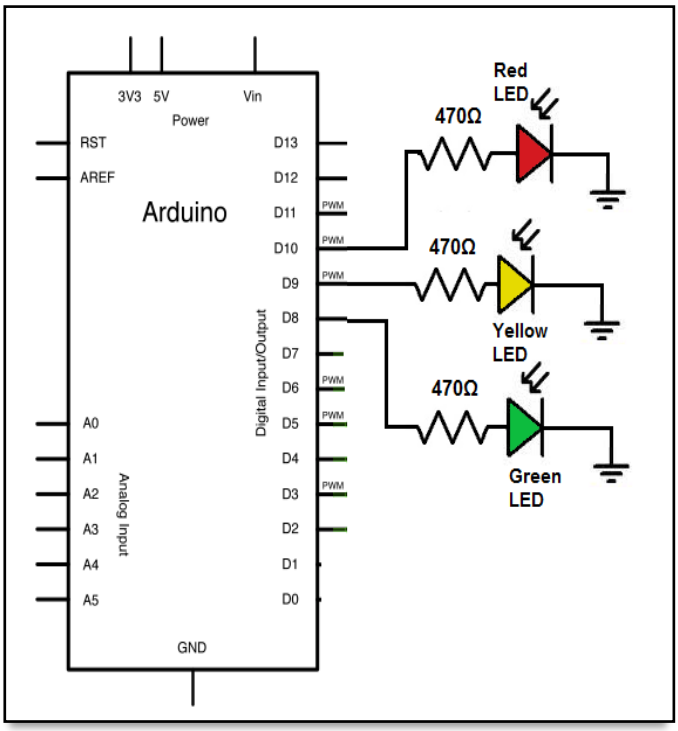
```
int timer=500;
void setup() {
  // initialize pin 2 to pin 7 as output pins
for (int number = 2; number < 8; number++) {
pinMode(number, OUTPUT);
  }
}
void loop() {
 // loop from the pin 2 to pin 7
for (int number = 2; number < 8; number++) {
digitalWrite(number, HIGH);   // turn on the pin
delay(timer);
digitalWrite(number, LOW); // turn off the pin
  }   // loop from pin 7 to pin 2
for (int number = 7; number >= 2; number--) {
digitalWrite(number, HIGH);    // turn on the pin
delay(timer);
digitalWrite(number, LOW);     // turn off the pin
  }
}
```

2.3 Traffic Light

Brief: Traffic light controller can be designed by giving definite amount of delay between switching of traffic light colors. In the code each light is turned on for one second while the other two colors are off and the cycle repeats with a delay if 1 second.

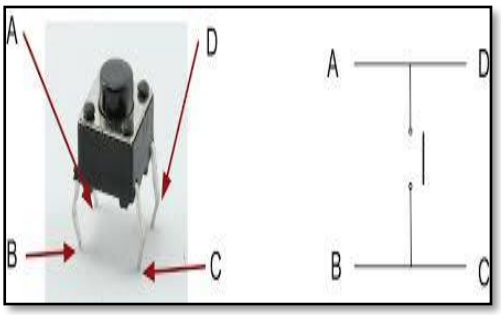
Components Required: LED (red, green, yellow), Resistor 220Ω





```
const int greenlight= 8; // assign greenlight to pin 8
const int yellowlight= 9; // assign greenlight to pin 9
const int redlight= 10; // assign greenlight to pin 10
void setup() {
pinMode (greenlight, OUTPUT);
pinMode (yellowlight, OUTPUT);
pinMode (redlight, OUTPUT);
}
void loop(){
digitalWrite (greenlight, HIGH); //turns greenlight on
digitalWrite (yellowlight, LOW); //turns yellowlight off
digitalWrite (redlight, LOW); //turns redlight off
delay(1000);
digitalWrite (greenlight, LOW); //turns greenlight  off
digitalWrite (yellowlight, HIGH); //turns yellowlight on
digitalWrite (redlight, LOW); //turns redlight off
delay(1000);
digitalWrite (greenlight, LOW);  //turns greenlight off
digitalWrite (yellowlight, LOW); //turns yellowlight off
digitalWrite (redlight, HIGH); //turns redlight on
delay(1000);
}
```

2.4 Controlling LED by Buttons

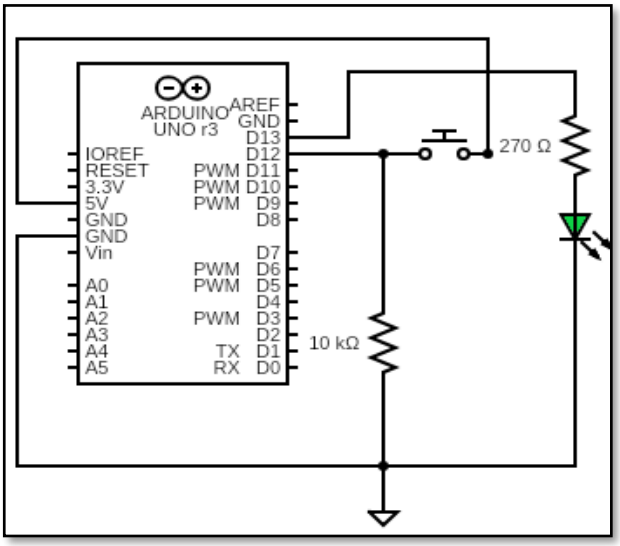


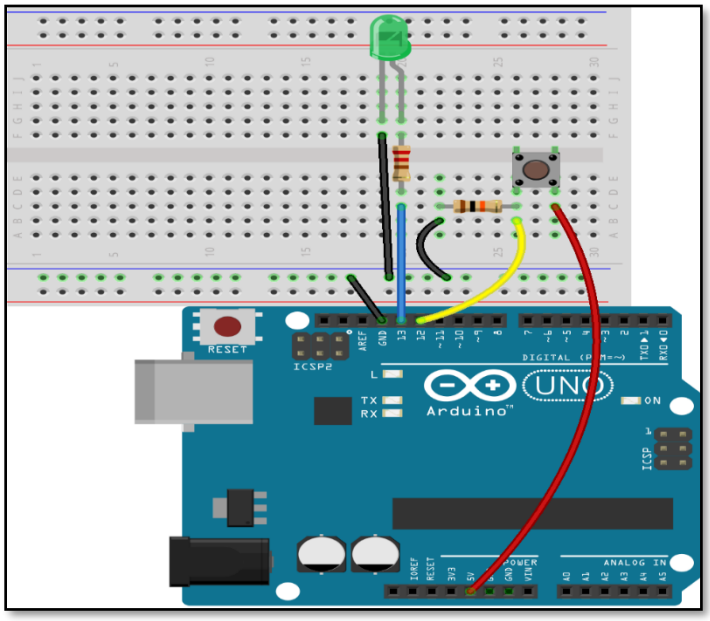
Brief: We use push button to ON and OFF LED. When button is pressed the circuit is closed which makes the LED to turn ON. In the code given below we read the value on the pin 12, if the button is pressed then the value at pin 12 will be HIGH then the LED is made to glow by setting the led pin to HIGH.

Components Required: Button switch, LED, Resistor 220Ω, 10KΩ, breadboard.

Push button works on the following principle

- Button not pressed = disconnected circuit
- Button pressed = connected circuit

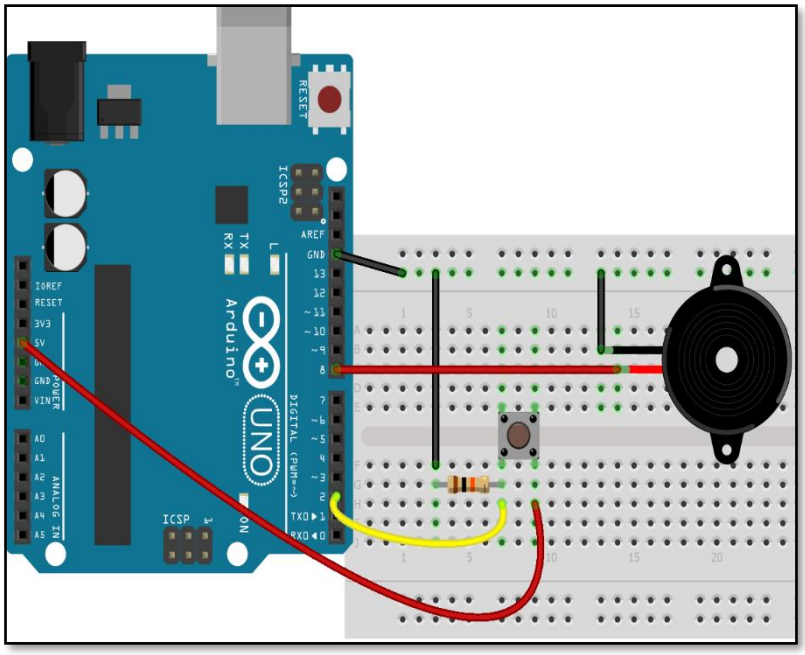


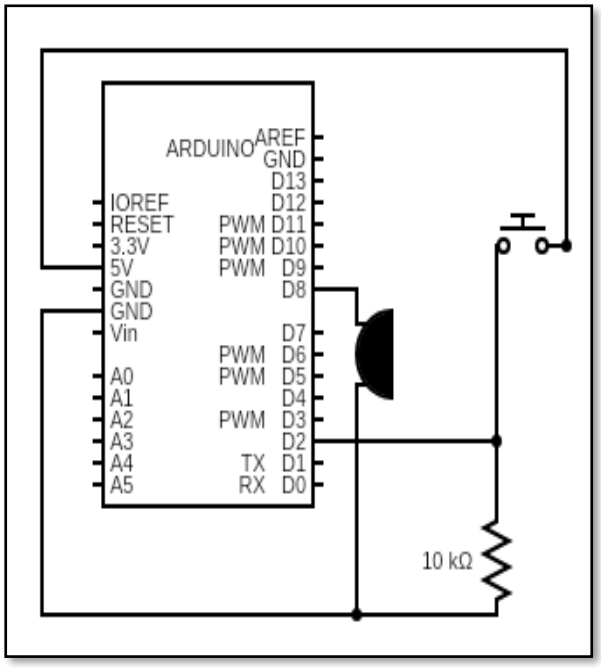


```
//LED is switched on and off using push button
// Written by ABRA ELECTRONICS
const int buttonPin = 12; //the button connected to pin 12
const int ledPin = 13; //the led connected to pin13
int buttonState = 0; // variable for reading the pushbutton status
void setup()
{
pinMode(buttonPin, INPUT); //initialize thebuttonPin as input
pinMode(ledPin, OUTPUT); //initialize the led pin as output
}
void loop ()
{   //read the state of the button value
buttonState = digitalRead (buttonPin);
 if (buttonState == HIGH)       // check if button is high (pressed)
          {
digitalWrite (ledPin, HIGH); //turn the led on
          }
else
          {
digitalWrite (ledPin, LOW); //turn the led off
          }
}
```

2.5 Doorbell using active buzzer

Brief: An **active buzzer** will generate a tone using an internal oscillator, so all that is needed is a DC voltage which we now provide through one of the digital pins on Arduino. Components: BUZ-120, Resistor 220Ω, Push button

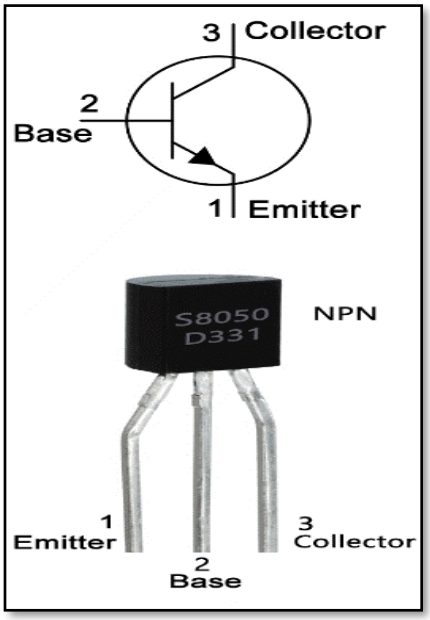


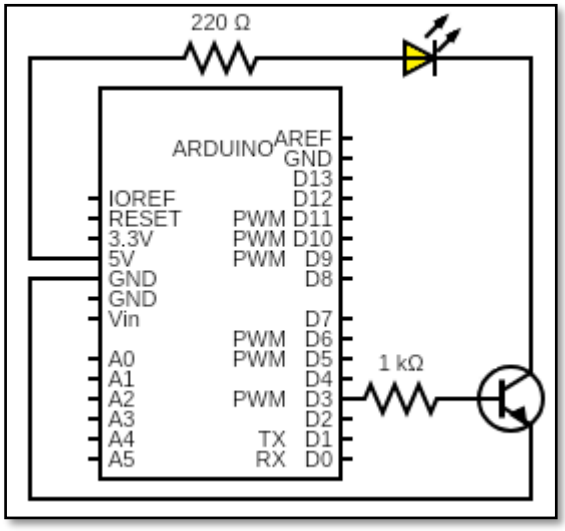


```
//Doorbell
//Turns buzzer on and off using the button.
// Written by ABRA ELECTRONICS
const int buttonPin = 2; //the button connect to pin2
const int buzzerPin = 8;  //the led connect to pin8
int buttonState = 0;  // variable for reading the pushbutton     status
void setup() {
pinMode(buttonPin, INPUT); //initialize the button pin as input
pinMode(buzzerPin, OUTPUT); //initialize the buzzer pin as output
     }
void loop() {
       //read the state of the button value
 buttonState = digitalRead(buttonPin);
            if (buttonState == HIGH ) { //and check if the button is pressed if it    is, the state will be HIGH
for (int i = 0; i < 25; i++) {
digitalWrite(buzzerPin, HIGH); //Activate the buzzer
delay(500);       //wait for half second
digitalWrite(buzzerPin, LOW); //Deactivate the buzzer
delay(500);       //wait for half second
        }
      }
    }
```

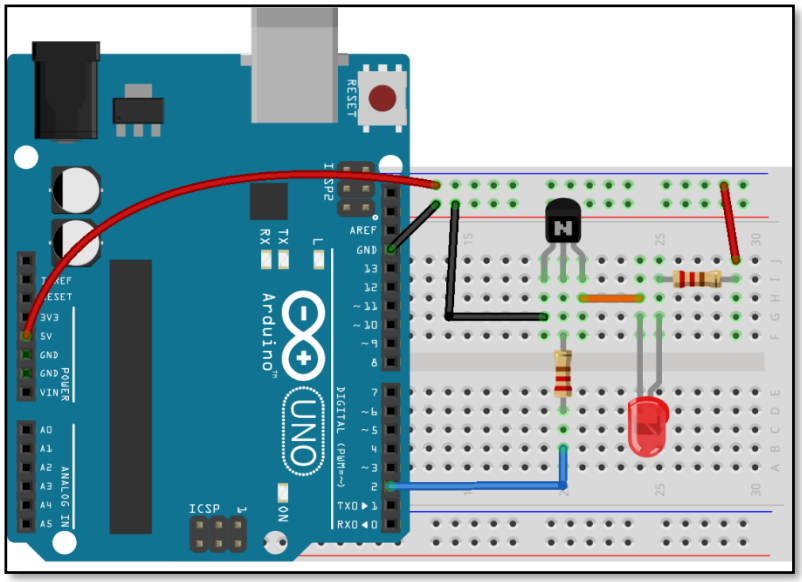
2.6 Transistor as switch

The most common uses for transistors in an electronic circuit is as a switch. Transistor conducts current across the collector-emitter path only when a voltage is applied to the base. When no base voltage is present, the switch is off. When base voltage is present, the switch is on. Basically Base-Emitter junction acts as a diode which conducts electricity only if bias voltage is applied.





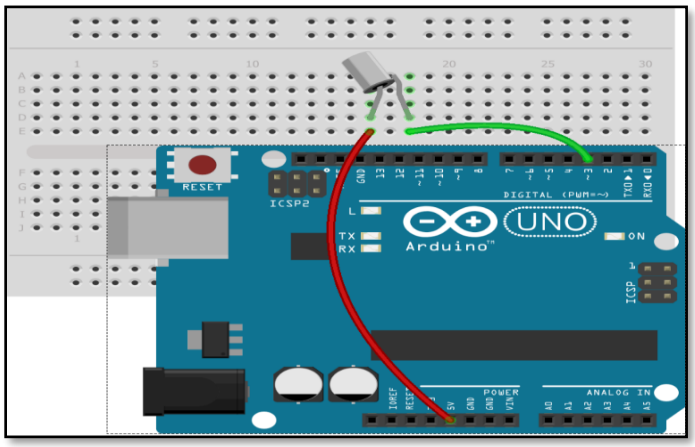
```
const int transistor_base = 3;

void setup()
{
  pinMode (transistor_base, OUTPUT);
}
void loop()
{
  digitalWrite (transistor_base, HIGH);// on the
switch
  delay(2000);
  digitalWrite (transistor_base, LOW);//off the
switch
  delay(2000);
}
```

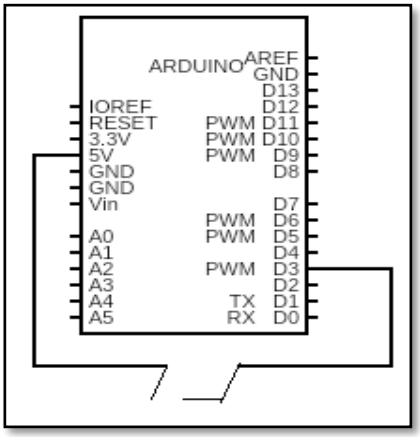


2.7 Tilt Switch (SENS-39)

Brief: Tilt switch is a switch which opens and closes an electrical circuit when it is tilted at certain angles. After connecting the circuit if the breadboard is tilted to certain angle the LED will glow. They are small, inexpensive, low-power, and easy-to-use.

Components: SENS-39



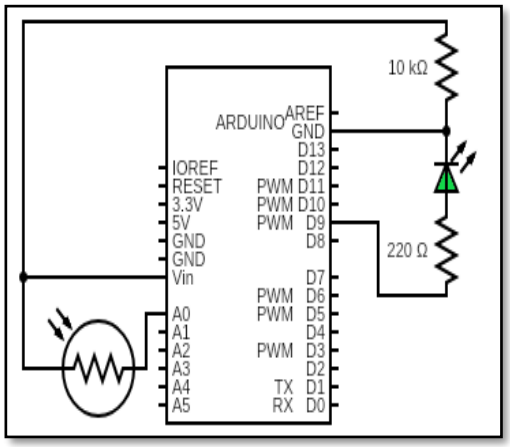


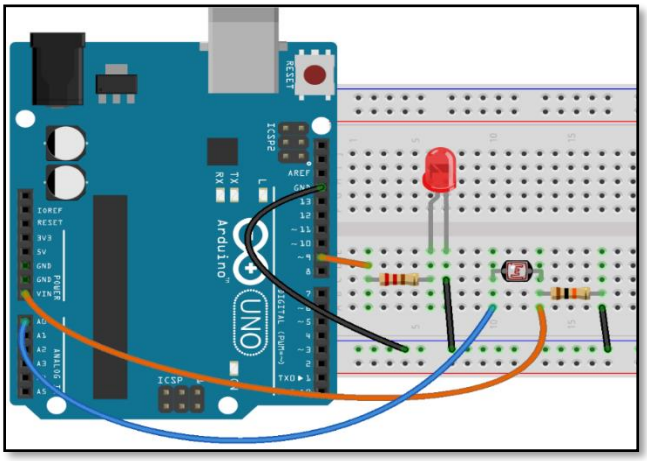

```
// Tilt switch
// Written by ABRA ELECTRONICS
int led_pin = 13; // Built in LED connected to pin
13
int tilt_switch = 3; // tilt switch connected to pin 3
int value;

void setup()
{
        pinMode(led_pin, OUTPUT);
        pinMode(tilt_switch, INPUT);
}
void loop()
{
value = digitalRead(tilt_switch); //Read the tilt
value
        if(value == HIGH)
        {
                digitalWrite(led_pin, HIGH);
        }
        else
        {
                digitalWrite(led_pin, LOW);
        }
}
```

2.8 Photoresistor (Photo-300)

Brief: A photoresistor or photocell is a light-controlled variable resistor. It works on the principle of photoconductivity. The resistance of a photoresistor decreases with increasing incident light intensity. When resistance decreases current flow through it.

```
//Use a photoresistor to turn on an LED in the dark
const int photocell = A0; // Photoresistor at Arduino analog pin A0
const int ledPin=9;      // Led pin at Arduino pin 9
int digitalvalue; // to store digitalvalue from photoresistor (0-1023)
void setup(){
 pinMode(ledPin, OUTPUT);  // Set ledPin - 9 pin as an output
 pinMode(photocell, INPUT);
}
void loop(){
  digitalvalue = analogRead(photocell);

  if (digitalvalue > 30){
    digitalWrite(ledPin, LOW);  //Turn led off if there is enough light in room
  }
  else{
    digitalWrite(ledPin, HIGH); //Turn led on if it is dark
  }
}
```

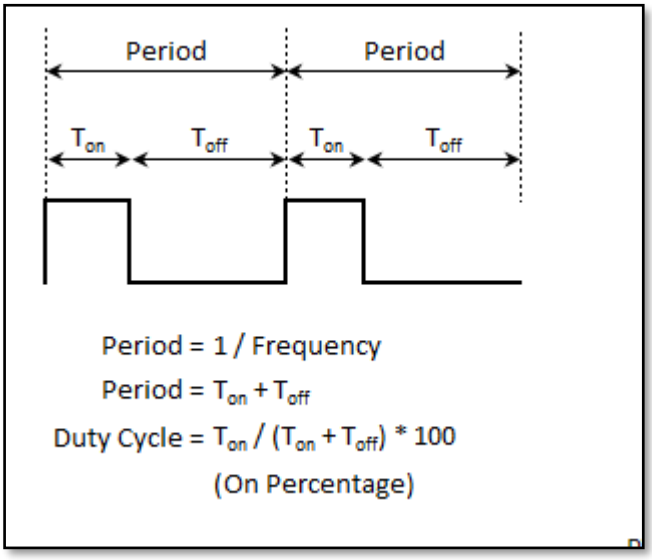


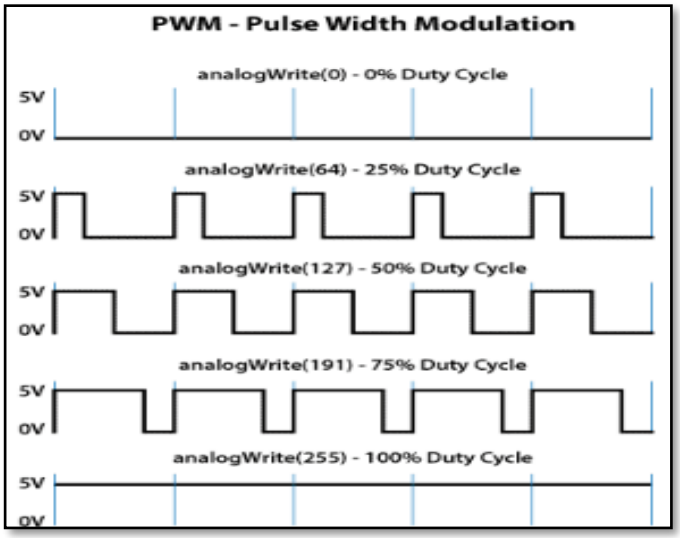


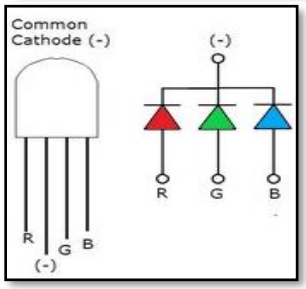
2.9 RGB LED

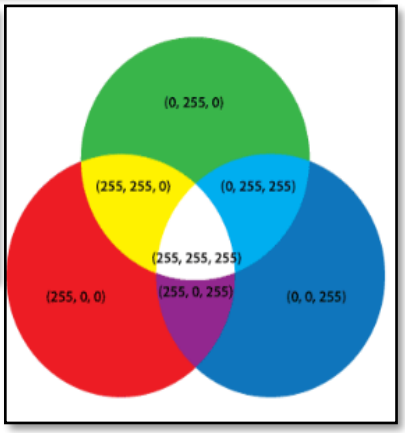
Brief: In this example we will learn how to vary the color of RGB led using Arduino. Varying the PWM values causes the change in color. Pulse Width Modulation (or PWM) is a technique for controlling power. We also use it here to control the brightness of each of the LEDs. Using PWM the amount of power delivered to a device can be controlled. Duty Cycle and Frequency concepts are used in PWM to control brightness of RGB LED. Duty cycle indicates the duration for which the pulse is HIGH over it period. In lay man terms this duty cycle is a value in percent of ON status compared to OFF status. From the figure below we can see the formula to calculate the duty cycle. It is measured in percentage and it indicates the voltage between OFF and ON levels (usually 0V and 5V).

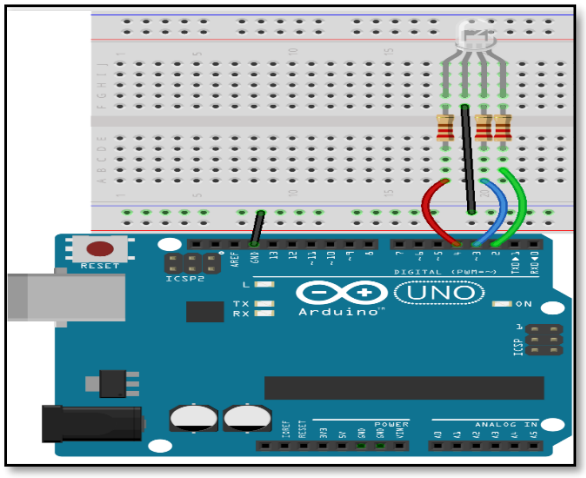
Components: Button switch (4), Red LED, Yellow LED, Green LED, Resistors 220Ω (3), 10KΩ (4),Breadboard



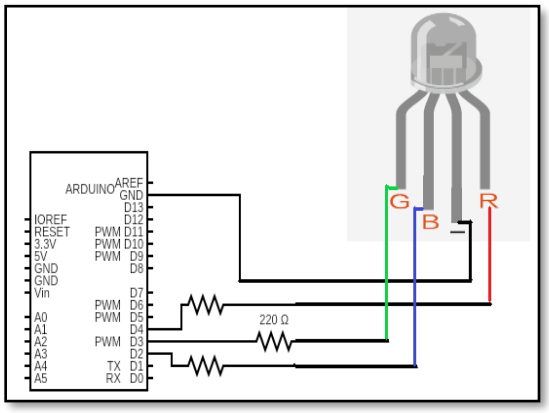









```
//RGB Led
// Written by ABRA ELECTRONICS
int blue_pin = 2;
int green_pin = 3;
int red_pin= 4;
void setup() {
  pinMode(red_pin, OUTPUT);
  pinMode(green_pin, OUTPUT);
  pinMode(blue_pin, OUTPUT);
}
void loop() {
  RGB_CODE(255, 0, 0); // Red colour
  delay(500); // delay by half a second
  RGB_CODE(0, 255, 0); // Green colour
  delay(500);
  RGB_CODE(0, 0, 255); // Blue colour
  delay(500);
RGB_CODE(255, 255, 0); // Yellow colour
  delay(500);
  RGB_CODE(255, 255, 255); // White colour
  delay(500);
RGB_CODE(255, 255, 125); // Raspberry colour
  delay(500);
  RGB_CODE(0, 255, 255); // Cyan colour
  delay(500);
  RGB_CODE(255, 0, 255); // Magenta colour
  delay(500);
}
void RGB_CODE(int red_code, int green_code,
int blue_code)
{ // function to read the pin values of RGB
  analogWrite(red_pin, red_code);
  analogWrite(green_pin, green_code);
  analogWrite(blue_pin, blue_code);
}
```



2.10 Thermistor (334-103)

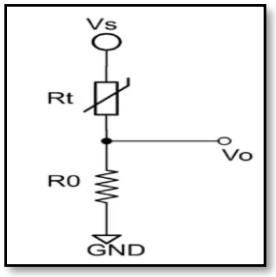
Brief: A thermistor is a type of resistor whose resistance is dependent on temperature. Thermistors are of two opposite types:

- With **NTC** thermistors, resistance ***decreases*** as temperature rises. An NTC is commonly used as a temperature sensor.
- With **PTC** thermistors, resistance ***increases*** as temperature rises.

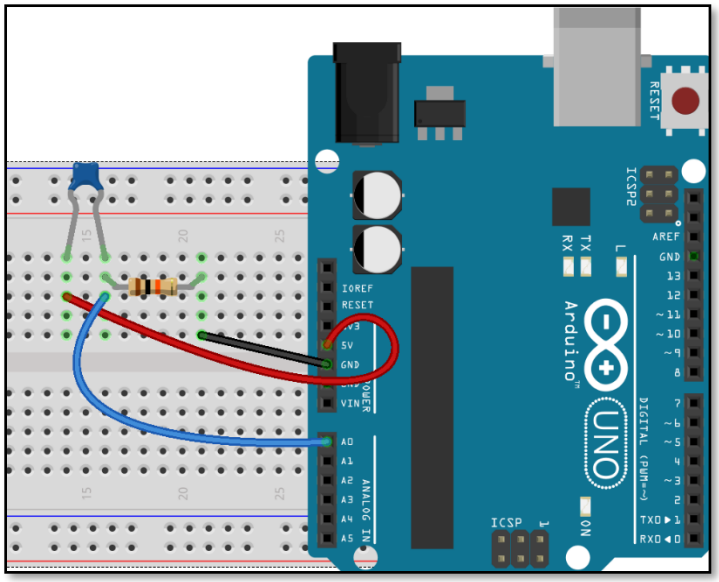
In this experiment we create a voltage divider between thermistor and 10kΩ resistor and perform the calculation.

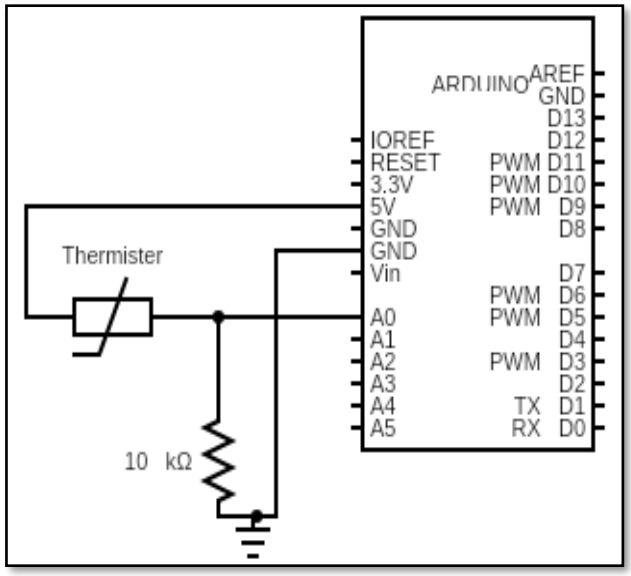
- **Vo = Vs * (R0 / (Rt + R0**
- **Rt = R0 * ((Vs / Vo) - 1)**
- **1/T = A + B*ln(R) + C*(ln(R))^3**

Components: 10K thermistor, 10kΩ resistor.




```
#include <math.h>
int thermistor = A0;
int Vo;
float R1 = 10000; / / resistor value (R1)
float logR2, R2, T, Tc, Tf;
float A = 1.009249522e-03, B = 2.378405444e-04, C =
2.019202697e-07; / / Value of constants

void setup() {
Serial.begin(9600);// set baud rate for serial
communication
}
// calculations
void loop() {

  Vo = analogRead(thermistor);
  R2 = R1 * (1023.0 / (float)Vo - 1.0);
  logR2 = log(R2);
  T = (1.0 / (A + B*logR2 + C*logR2*logR2*logR2));
  Tc = T - 273.15; //Kelvin to celcius conversion
  Tf = (Tc * 9.0)/ 5.0 + 32.0; //celcius to Fahrenheit

  Serial.print("Temperature measured: ");
  Serial.print(Tf);
  Serial.print(" Fahrenheit; ");
  Serial.print(Tc);
  Serial.println(" Celcius");
  delay(500);
}
```



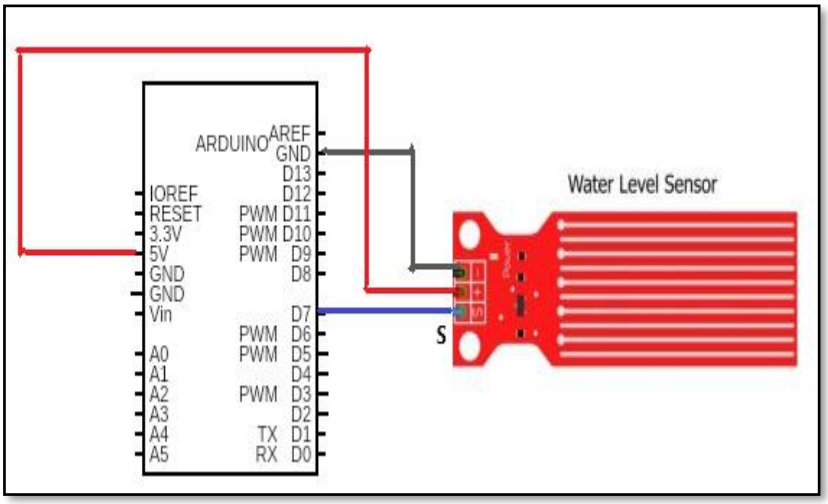


2.11 Water level sensor (SENS-78)

Brief: Water level sensor work based on variable resistance of the series of exposed parallel conductors (just like a potentiometer) whose resistance change according to the water level. The resistance is inversely proportional to the height of the water.

Once the connections are made insert the sensor into water and you can notice the level of water on serial monitor.

Components: SENS-78, Jumper wires.

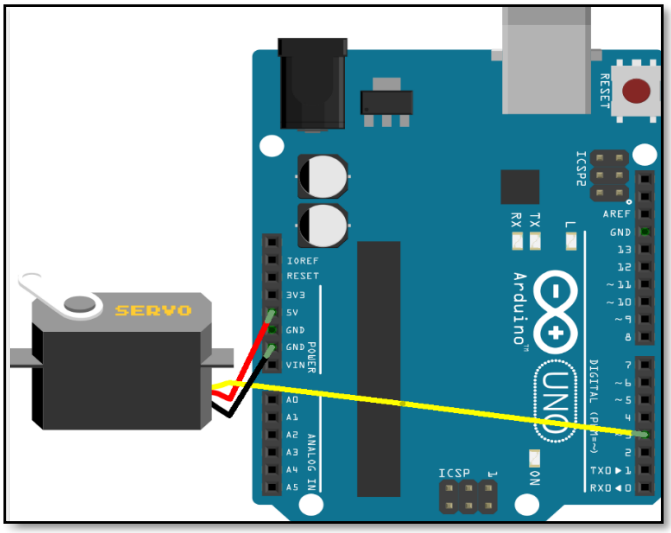


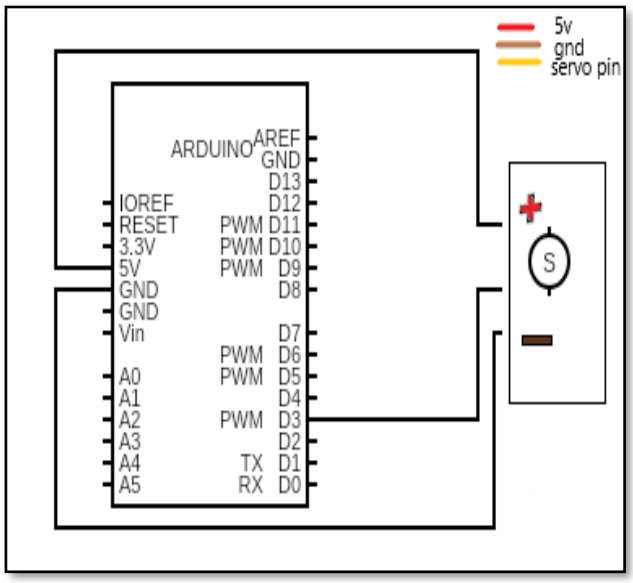

```
#define sensor_switch 7

#define sensor_input A0
// Value for storing water level
int depth = 0;

void setup() {
   Serial.begin(9600);
//set pin 7 as the switch
  pinMode(sensor_switch, OUTPUT);
//switch off sensor
  digitalWrite(sensor_switch, LOW);
}
void loop() {
  int water_level = Sensor_level();
  Serial.print("Water water_level is ");
  Serial.println(water_level);
  delay(100);
}
int Sensor_level() {
  digitalWrite(sensor_switch, HIGH);  // Turn the sensor ON
  delay(10);
  depth = analogRead(sensor_input);   // Read the analog value form sensor
  digitalWrite(sensor_switch, LOW);   // Turn the sensor OFF
  return depth;           // send depth of water water_level
}
```

2.12 Servo Motor (SG90)

Brief: A **servo motor** is an electrical device which can rotate an object with high precision. If you want to rotate an object at a specific angle, we use servo motor. Servo motor is controlled by PWM (Pulse with Modulation). In this project we can connect small servo motors directly to an Arduino to control the shaft position very precisely. Arduino sends PWM signal to the servo which then rotates by an angle depending upon the pulse width.

Components: Servo motor



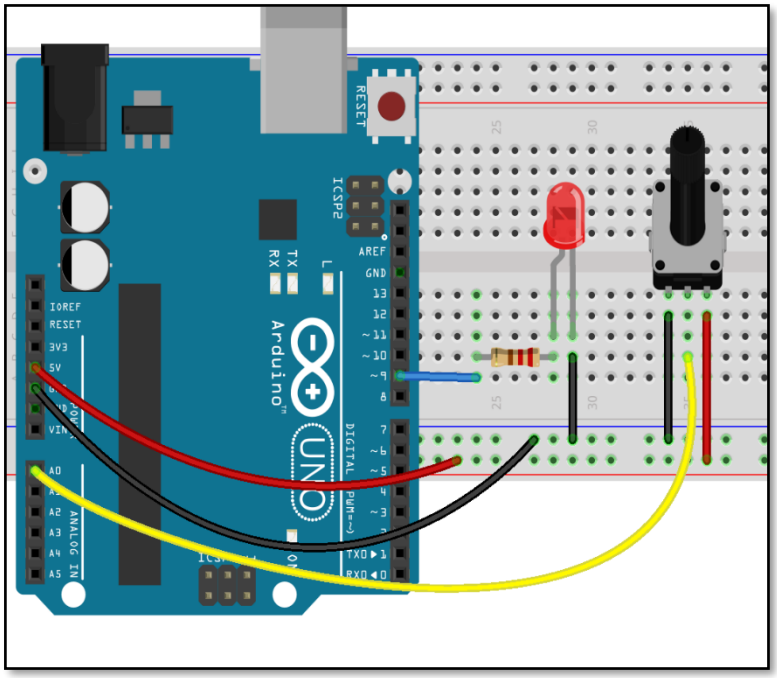


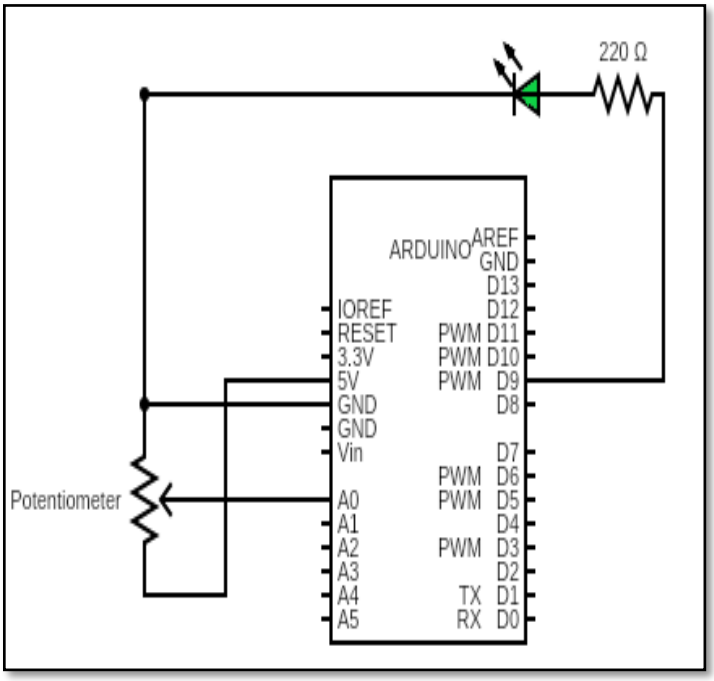

```
#include <Servo.h>  // Include the Servo library
int ser_pin = 4; //  Servo pin connected to pin 4
Servo ser_obj;  // instantiate servo object
int i=0;
void setup() {
  ser_obj.attach(ser_pin);
}
void loop(){
   while(ser_pin=HIGH){
  for (i = 0; i <= 360; i++) { //anti clockwise
   ser_obj.write(i);
   delay(10);
  }
  for ( i = 360; i>= 0; i--) { //clockwise
   ser_obj.write(i);
   delay(10);
  }
   }
}
```

2.13 Controlling an LED by potentiometer

Brief: Brightness of the LED is controlled using potentiometer by changing the value of the resistance.

Components: Breadboard, 220Ω resistor, LED.






```
//          Controlling an LED by potentiometer
int Analoginput=A0;  //Analog pin A0 receives input controlled by potentiometer.
int out1=9;   //Pin 9 is output to which LED is connected.
void setup()
{
pinMode(out1, OUTPUT); //Pin 9 is output
}
void loop()
{
int recievedinput=analogRead(Analoginput); //Reading the voltage out by potentiometer
int ledbrightness=recievedinput/4;      //Dividing by 4 to bring it in range of 0 – 255
analogWrite(out1, ledbrightness);
}
```