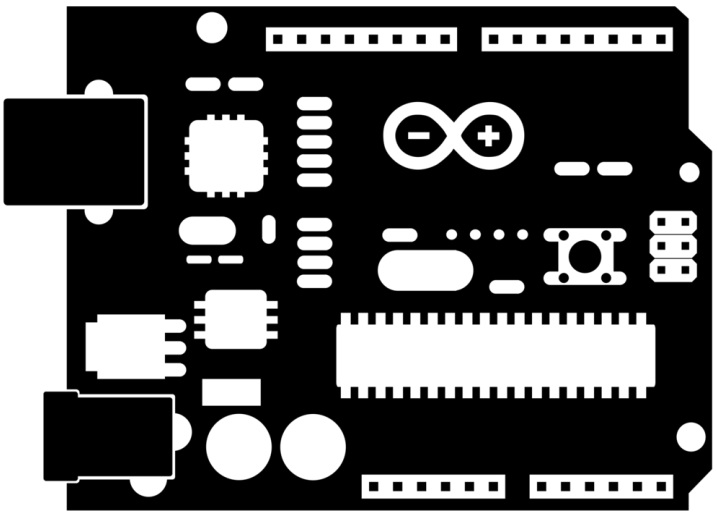
13 ARDUINO MINI PROJECTS



PRAJWAL M

B.E(EEE)

BIET

Preface

Welcome to the Arduino, and welcome to the exciting world of physical computing! Arduino is an open source project consisting of both hardware and software. It was originally created to give designers and artists a prototyping platform for interaction design courses. Today hobbyists and experts all over the world use it to create physical computing projects, and you can too.

The Arduino lets us get hands-on again with computers in a way we haven't been able to since the 1980s, when you could build your own computer. And Arduino makes it easier than ever to develop handcrafted electronics projects ranging from prototypes to sophisticated gadgets. Gone are the days when you had to learn lots of theory about electronics and arcane programming languages before you could even get an LED blinking. You can create your first Arduino project in a few minutes without needing advanced electrical engineering course work.

If you are interested in electronics—and especially in building your own toys, games, and gadgets—then this book is for you. Although the Arduino is a nice tool for designers and artists, only software developers are able to unleash its full power.

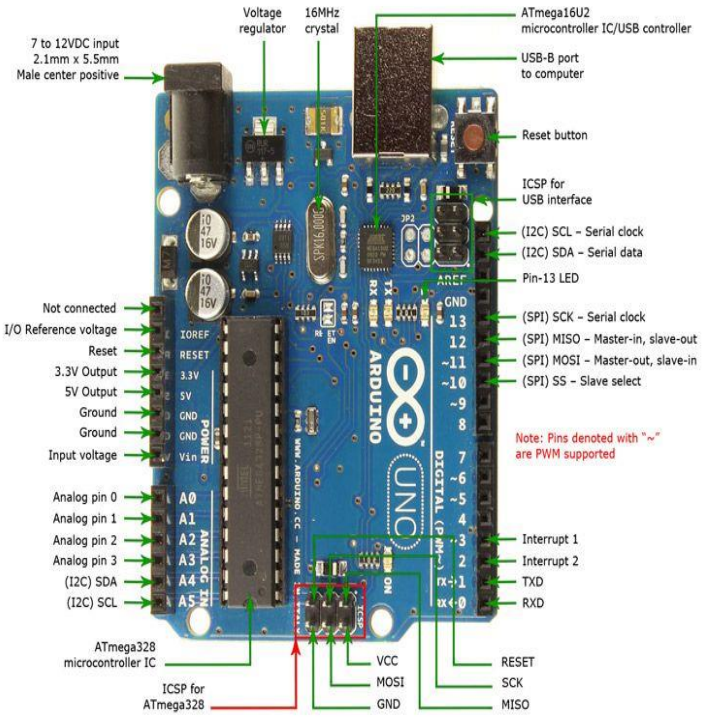
Basic Learning Tutorials

Arduino a general Introduction

Introduction:

Arduino is an Integrated Development Environment based upon Processing. It has made very easy several things namely these are embedded system, physical computing, robotics, automation and other electronics based things.

Pin Diagram:



Pin Description:

Pin Category	Pin Name	Details
Power	Vin, 3.3V, 5V, GND	Vin: Input voltage to Arduino when using an external power source. 5V: Regulated power supply used to power microcontroller and other components on the board. 3.3V: 3.3V supply generated by on-board voltage regulator. Maximum current draw is 50mA. GND: ground pins.
Reset	Reset	Resets the microcontroller.
Analog Pins	A0 – A5	Used to provide analog input in the range of 0-5V
Input/Output Pins	Digital Pins 0 - 13	Can be used as input or output pins.
Serial	0(Rx), 1(Tx)	Used to receive and transmit TTL serial data.
External Interrupts	2, 3	To trigger an interrupt.
PWM	3, 5, 6, 9, 11	Provides 8-bit PWM output.
SPI	10 (SS), 11 (MOSI), 12 (MISO) and 13 (SCK)	Used for SPI communication.
Inbuilt LED	13	To turn on the inbuilt LED.
TWI	A4 (SDA), A5 (SCA)	Used for TWI communication.
AREF	AREF	To provide reference voltage for input voltage.

Arduino Installation

Software Required:

The Arduino comes for the following operating systems. You may go for any of these.

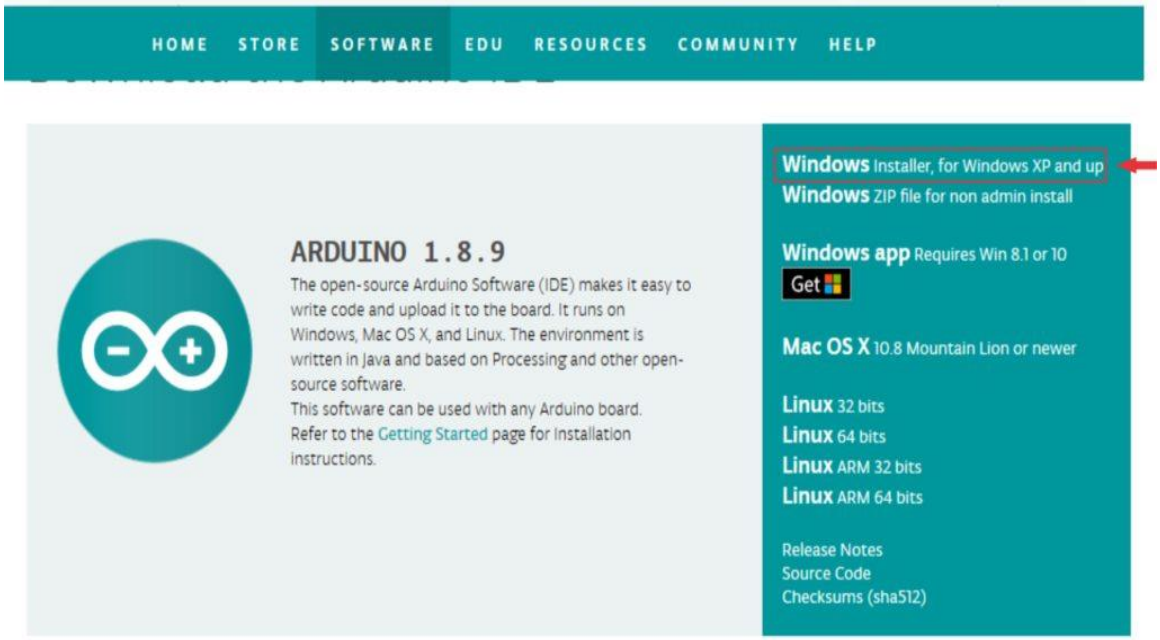
Windows

Linux

Mac OS

Download Arduino software from here

https://www.arduino.cc/en/Main/Software



Arduino Software Introduction:

Some general Understanding:

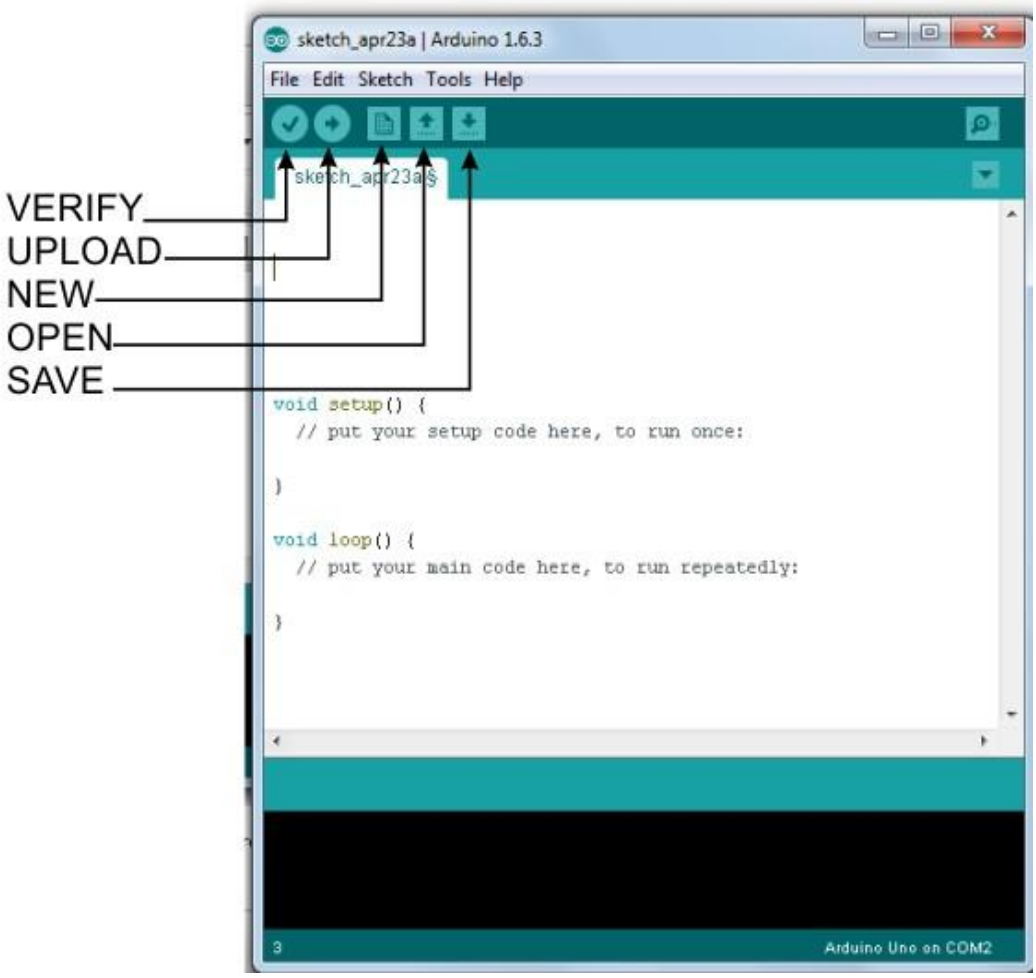
Code we write in Arduino is known as SKETCH.

Compilation of the code is known as VERIFY.

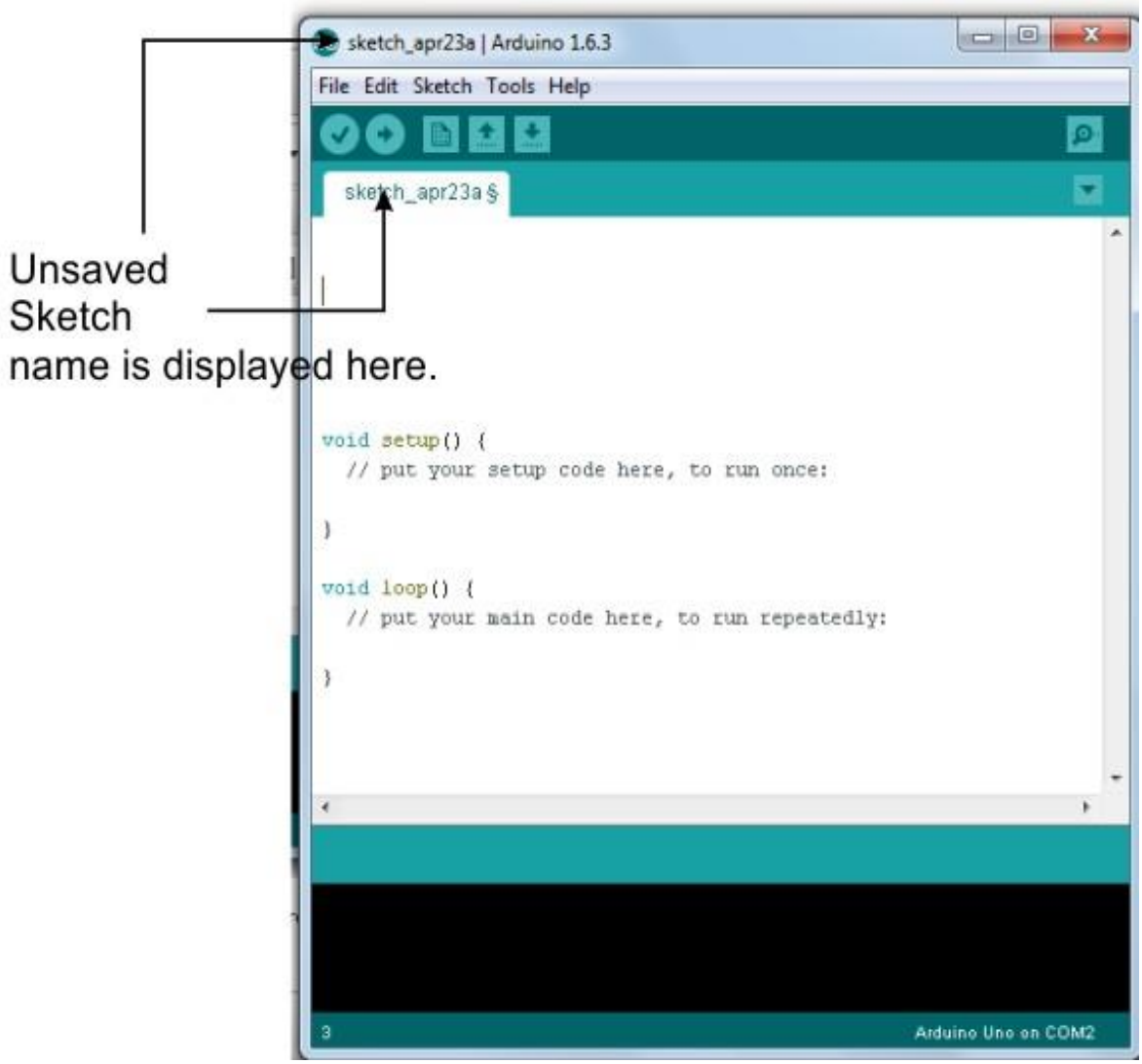
Transfer of the code from computer to Arduino board is known as UPLOAD.

If you directly hit UPLOAD button of Arduino software, the software will first VERIFY the code and then will transfer that code to Arduino Board.

Buttons of Arduino software:

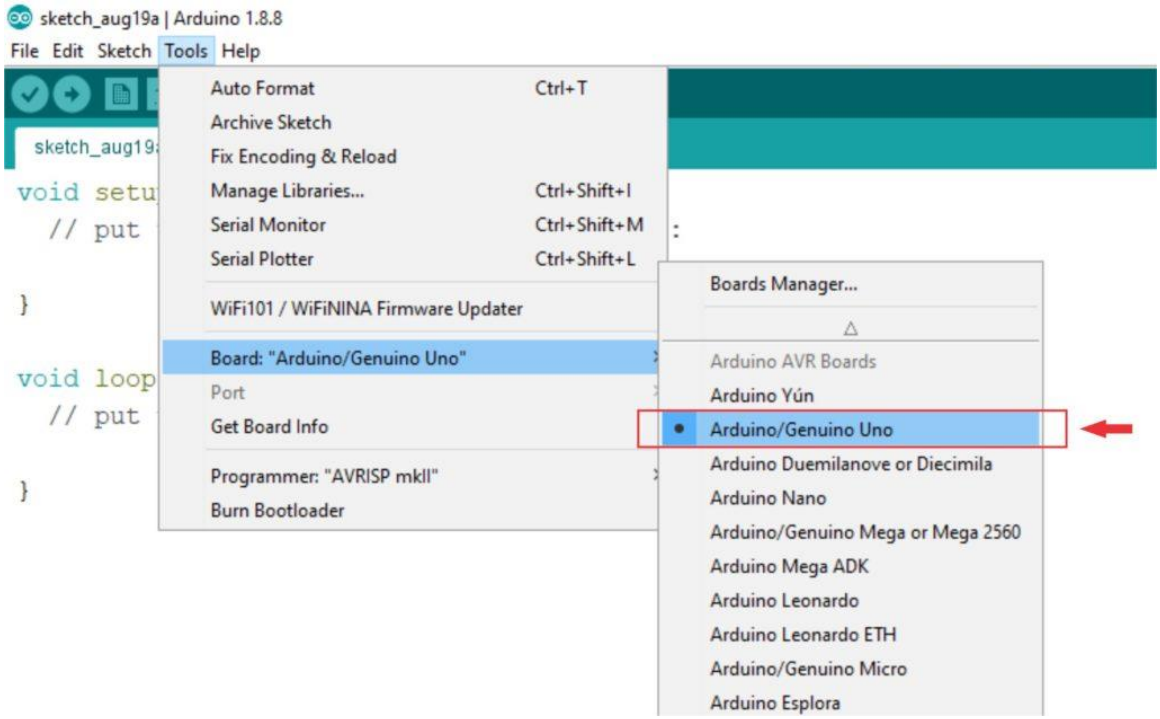


Untill you save your project, it displays its name as sketch_date. Default folder to save Arduino code(sketch) is My Documents/Arduino or similar location for Mac OS and Linux. Arduino saves each code in a folder extension of code is .ino. By default it creates a folder and in that folder create a file with the extension we have seen. It is to be noted that name of the folder and containing file should be same (If you want to rename the sketch then change name of both



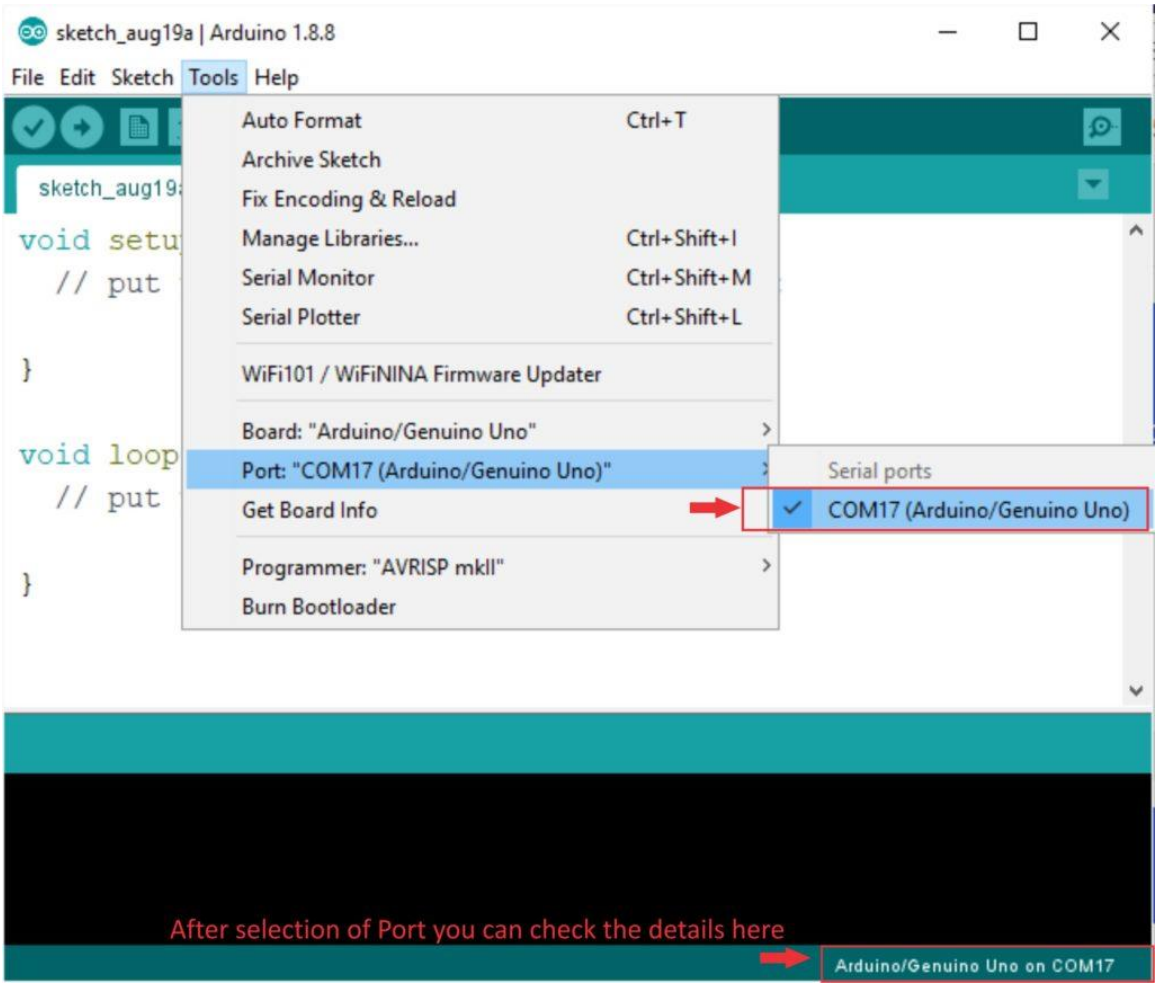
Selection of Board: Select the Arduino Board you are having from the available list.

For Arduino UNO – Arduino/Genuino UNO



Connecting Arduino:

Connect the Arduino Board to your computer using USB Cable once connected you will see a COM Port in your Arduino IDE as shown below.



Now you are ready to write you code. Once you have written your code save it and press UPLOAD button.

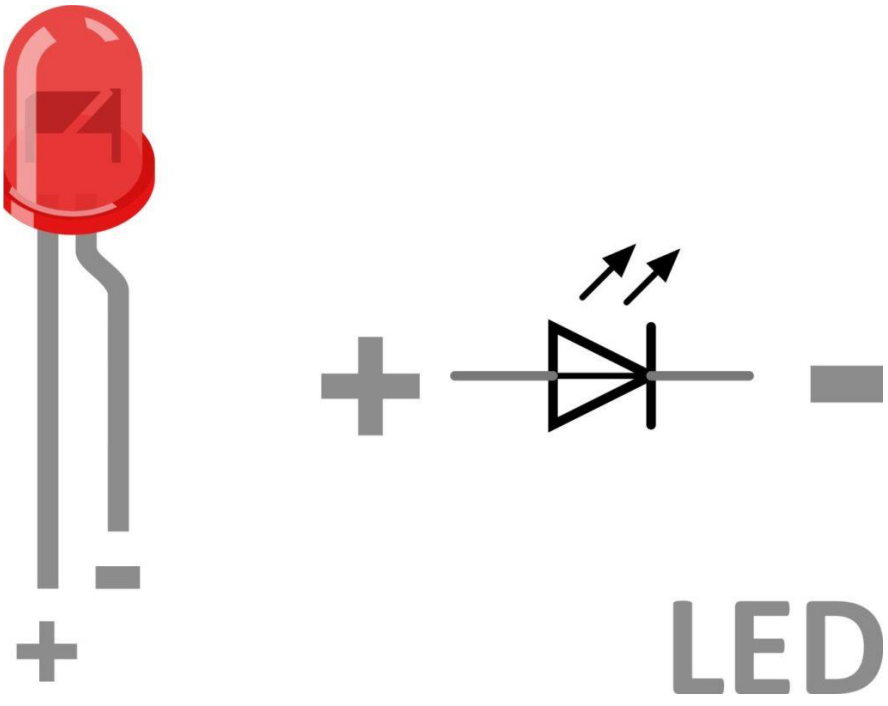
Experiment-1
Arduino Digital Output – LED Blinking

This tutorial explains how to take digital output from Arduino. One of the basic tutorials for Arduino. The output is taken on a LED that blinks for an interval of 1 second.

1.Introduction:

1.1 LED Pins:

LED has two pin interface. Both of these pins are to be given input supply to the LED. Long legs is for positive supply the smaller one is for negative supply. following image depicts it clearly.



Digital means either 0/1, in other words HIGH/LOW or ON/OFF. So in the form of digital output we shall get either +5V or 0V on digital pin of Arduino. How does it happen that is illustrated ahead.

2. Required Hardware:

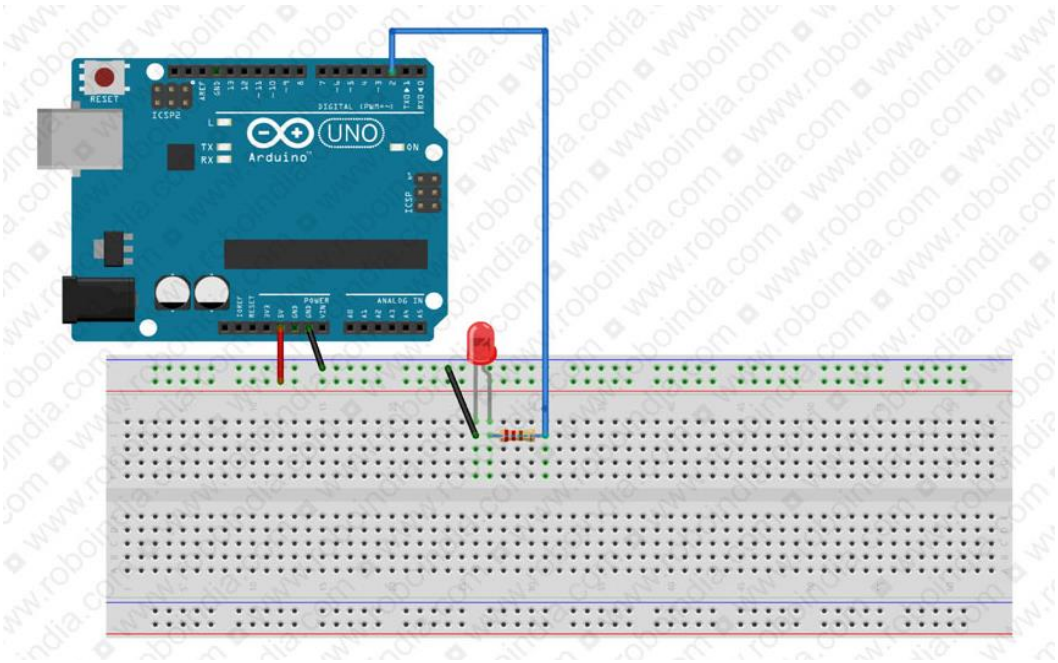
Following Hardware will be required to perform this LED blinking sketch.

S.No.	Item	Quantity
1.	**Arduino UNO Board**	1
2.	**Bread Board**	1
3.	**Male to male Jumpers**	4
4.	**Indicatior LED**	1
5.	**100 OHM Resistance**	1

3. Building Circuit:

Make following circuit with the help of above mentioned components.

You may go with original Arduino UNO Board



4. Programming:

Download the code:

https://drive.google.com/open?id=1YkX-UfTelPbI8dCvyESlU6zSJpjQPwgN

```
// Digital output is taken on a LED that remains ON for one
second and
// OFF for another.

// Defining Pin 2 as LED.

const int    LED = 2;   // from the circuit we can see that we have connected LED on Pin 2

void setup() {

    pinMode(LED, OUTPUT); // Defining LED pin as OUTPUT
Pin.

}

void loop() {

  digitalWrite(LED, HIGH); // LED gets turned ON
(1/HIGH/+5V)

 delay(1000);              // Waiting for one second.

  digitalWrite(LED, LOW);  // LED gets OFF (0/LOW/0V/GND)

  delay(1000);         // here and above Delay is in mili
second (1000 = 1 second)

}
```

Output:

After successful uploading of the code,the execution of code makes the LED on for one second and off for another one second.This happens in an infinite loop.Thus the LED keeps blinking.

Experiment-2

Arduino Analog Output – LED fade in and fade out

This tutorial explains how to take analog output from Arduino. One of the basic tutorials for Arduino. The output is taken on a LED that fades in and fades out.

1. Introduction:

A step by step illustrated very basic tutorial for Arduino. Here we are taking analog output on a LED. This LED gets fade in and then fade out.

Arduino gives analog output in range of 0 to 255. Technically the output is digital but in the form of PWM, but it seems to be analog.

Arduino Boards have 6 PWM(Analog Pins) these are PIN No. 3,5,6,9,10,11.

2. Required Hardware:

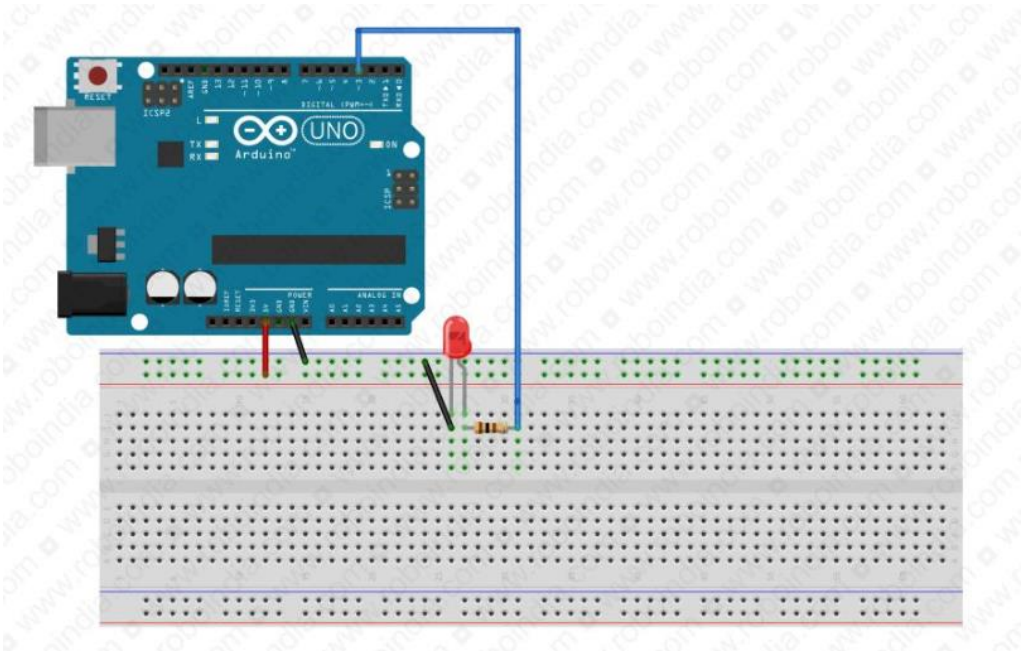
Following Hardware will be required to perform this LED fade in and fade out circuit.

S.No.	Item	Quantity
1.	**Arduino Board**	1
2.	**Bread Board**	1
3.	**Male to male Jumpers**	4
4.	**Indicator LED**	1
5.	**100 OHM Resistance**	1

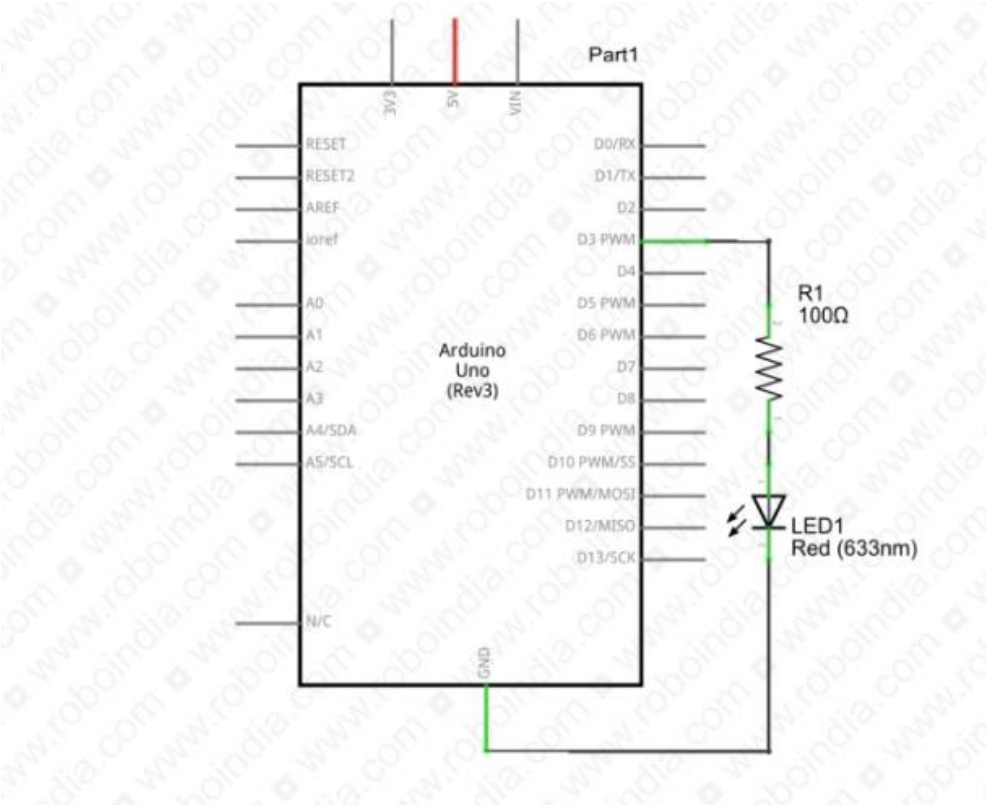
3. Building Circuit:

Make following circuit with the help of above mentioned components.

You may go with original Arduino UNO Board



Schematic circuit:



4. Programming:

Download the code here:

https://drive.google.com/open?id=1UDCzM6jmwATJJo_dZejBq5FQaH9kMlmC

Once we are done with circuit part, here is our programme to this circuit.

```
/*Analog output is taken through PWM.*/

int LED_ao = 3; // The LED attached to Pin 3 for analog
output.

void setup()  {

pinMode(LED_ao, OUTPUT); // Declaring Pin as output.

}

void loop()  {

 // Range of PWM is 0 to 255. So we are running FOR LOOP
for 1 to 255.

  for (int brightness=1; brightness<=255;
brightness++)  // For loop for Fade In effect.

    {

      analogWrite(LED_ao, brightness);  // LED will glow
for value of 1 to 255.

      delay(20);                        // Small delay to
see fade effect.

    }
```

```
  for (int brightness=255; brightness>0; brightness--) //
Same FOR LOOP for Fade out effect.

    {

      analogWrite(LED_ao, brightness);

      delay(20);

    }









}
```

Output:

The execution makes fed in and fed out effects on the LED.To vary speed of fed in and fed out effect try changing delay.

Experiment-3

Arduino Analog input and output on LED & Serial.

This tutorial explains how to take analog input to Arduino. One of the basic tutorials for Arduino. This input is shown through LED and Serial monitor.

1. Introduction:

Analog input form a potentiometer. And this input is shown on LED as PWM and analog values on Serial monitor.

Arduino gives analog output in range of 0 to 255. Technically the output is digital A step by step illustrated basic tutorial for Arduino. Here we are taking but in the form of PWM, but it seems to be analog.

Arduino Boards have 6 PWM(Analog Pins) these are PIN No. 3,5,6,9,10,11.

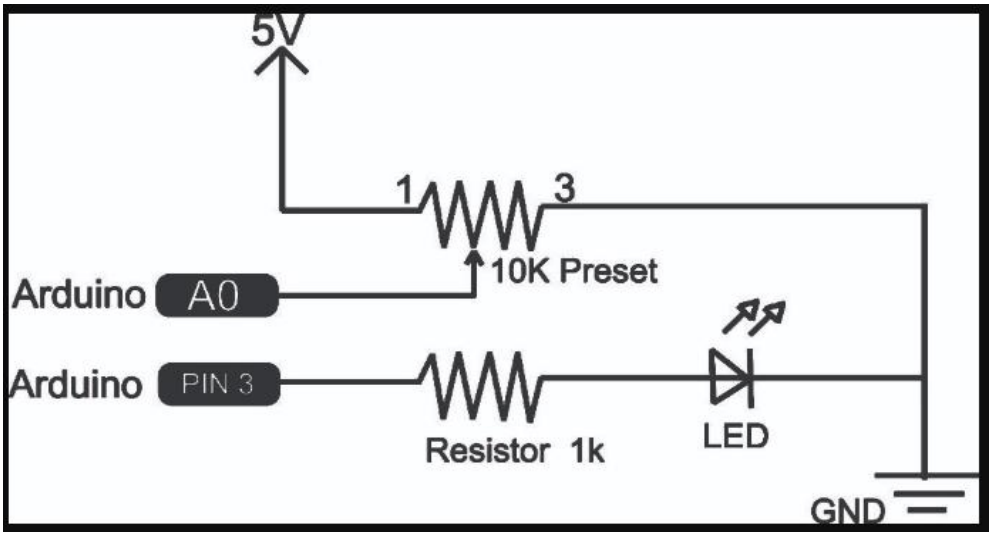
Arduino Boards have 6 Analog Input pins these are PIN A0, A1, A2, A3, A4 and A5.

2. Hardware required:

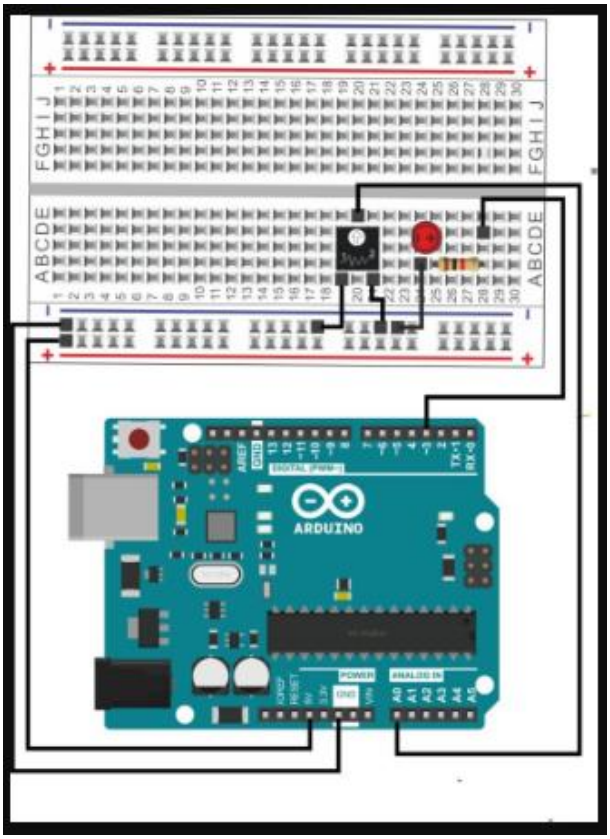
S.No.	Item	Quantity
1	Arduino UNO	1
2	Breadboard	1
3	LED	1
4	Resistor 1k	1
5	Jumper Male to male	7
6	Preset 10k	1

2. Building Circuit:

Schematic Diagram:



Circuit layout:



3. Programming:

Download the code:

https://drive.google.com/open?id=1BhwX0QzzBdxvh2xHIzfuvaAMFw5q_X3_

Once we are done with circuit part, here is our programme to this circuit.

A few points to understand:

1. Arduino has got 10 bit ADC so input we are taking from potentiometer will give us 10 bit data i.e. 0 to 1023.

2. Arduino's PWM (output for LED) has got a range of 0 to 255.

3. Thus in order to take analog input to PWM we have divided the input value by 4. Please go through the coding you will be clear about these points.

4. This programme runs for ever, every time it takes reads value from input, if it is near about the same to last value it will not do anything, if we dont apply this algo of comparing last value to new value we will receive every value on serial monitor and it will be a difficult thing to even read the serial monitor.

```
/*

Tutorial for Analog input and Output.

Input through Potentiomete and output through PWM on LED
and serial monitor.

 */



int analog_ip = A0;   // select the input pin for the
potentiometer
```

```
int LED = 3;           // select the pin for the LED

int inputVal = 0;     // variable to store the value coming
from potentiometer

int old_ip_val = 0;   // varialbe to store historic value
of analog input.




void setup() {

  pinMode(LED, OUTPUT);  // Defining LED Pin as output

  Serial.begin(9600);   // Setup Serial
Communication.

  Serial.print("ROBO INDIA\nroboindia.com\nTutorial on
Analog input and output.\n");

}




void loop(){

    inputVal = analogRead(analog_ip);  // Reading and
storing analog input value.

    // following if condition is to check last value with
new value if new value = +/-5

    // then we are not interested to do anything. For other
values we want to it to to -
```

```
    if (inputVal <old_ip_val-5 || inputVal >old_ip_val+5)

    {

        Serial.print("Input Value    : ");

        Serial.print(inputVal);                 //
Printing Analog input value (0-1023)

        Serial.print("\nLED Brightness : ");

        Serial.print(inputVal/4);               //
Printing PWM values for LED (0-255)

        Serial.print("\n");

        analogWrite(LED, inputVal/4);           //
Sending PWM value to LED.

        delay(500);                             //
Waiting for a while.

        old_ip_val = inputVal;                 // Storing
historic value of analog input.

  }

}
```

Output:

The analog input is printed on the serial monitor.LED brightness is adjusted as per the analog input value.Try rotating preset and see the effect on the LED and serial monitor.

Project-4

Arduino – Digital Input & Output | Pushbutton & LED

This tutorial explains basic concepts of arduino. Digital input and digital output. Digital input is taken through push button and that is detected by Arduino. This input is processes by Arduino and it send digital command to attached LED.

When the button is pressed LED glows.

1. Introduction:

A step by step illustrated basic tutorial for Arduino. In this tutorial we are taking digital input from a push button switch. That input is read by Arduino board and decision is taken accordingly. When we press the button LED glows. Thus this tutorial is for both digital input and digital output.

1.1 Digital Input:

Digital input means when we are supplying ***HIGH/1/+5V or LOW/0/GND*** to the Arduino board. On the contrary digital output means when we are taking ***HIGH/1/+5V or LOW/0/GND*** from the Arduino.

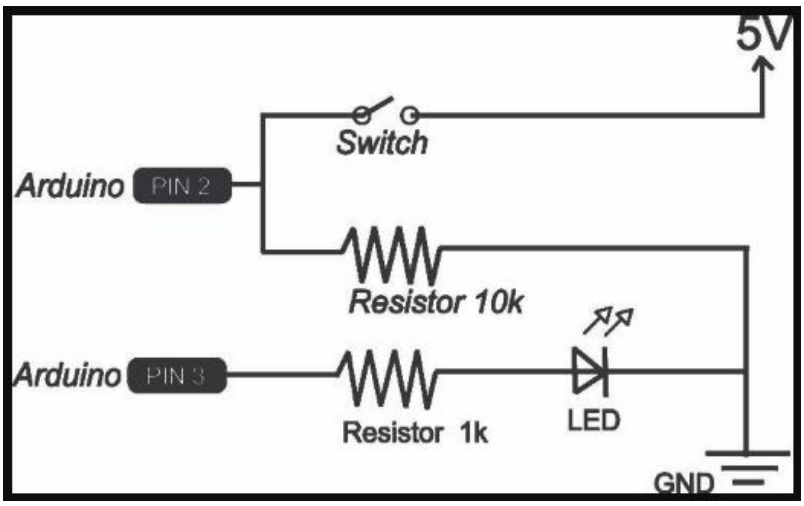
In this example Pin No. 2 of Arduino is connected to +5V through a switch and the same pin is also connected to GND via 10K resistance. This resistance keeps this pin at GND and when we press button it gets connected to +5V. So we can understand that if switch is released the Input Pin (Pin No.2) is *GND/LOW/0* and when switch is presses it is *+5V/HIGH/1.*

2. Required Hardware:

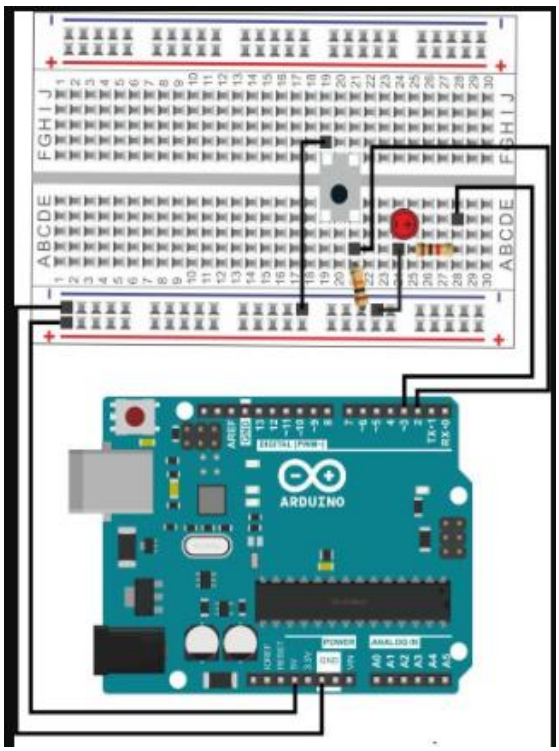
S.No.	Item	Quantity
1	Arduino UNO	1
2	Breadboard	1
3	LED	1
4	Resistor 1k	1
5	Resistor 10k	1
6	Jumper Male to male	6
7	Push Button Switch	1

3. Building Circuit:

Schematic Diagram:



Circuit Layout:



4. Programming:

Download the code:
https://drive.google.com/open?id=1d-O5hqmtrISIzmF6er4vio1UF-KNDo-j

This Once we are done with circuit part, here is our programme to this circuit. All of code has a simple function to perform, if Button is pressed glow the LED.

```
/*

Tutorial on Digital input through button.

*/



const int BUTTON = 2;      // Pushbutton Input to Pin No.2

const int LED =  3;        // Output LED to Pin No. 3




int BUTTONState = 0;       // To store input status



void setup() {

  pinMode(LED, OUTPUT);     // Define LED pin as output.

  pinMode(BUTTON, INPUT);   // Define BUTTON pin as
Input.
```

```
}



void loop(){

  BUTTONState = digitalRead(BUTTON); // Reading input from
Button Pin.



  if (BUTTONState == HIGH) // Checking if Input from button
is HIGH (1/+5V)

   {

     digitalWrite(LED, HIGH);  // If input is High make LED
ON (HIGH/1/+5V)

   }

  else

  {

     digitalWrite(LED, LOW);  // For every other condition
make LED OFF (0/GND/LOW)

  }

}
```

Output:

After successful uploading of the code,try pressing and releasing the button,the LED glows if the button is pressed and gets off when the button is released.

Project-5

Arduino-rgb-led

This tutorial explains all you need about RGB LED and Arduino. You can set any color to RGB LED and can transform from one color to another with different speed. This Tutorial includes Header file for RGB control and that makes it very easy to use.

1. Introduction:

A step by step illustrated basic tutorial for Arduino. This tutorial explains how to control RGB LED on Arduino platform. We have included two examples here.

1.1 RGB LED:

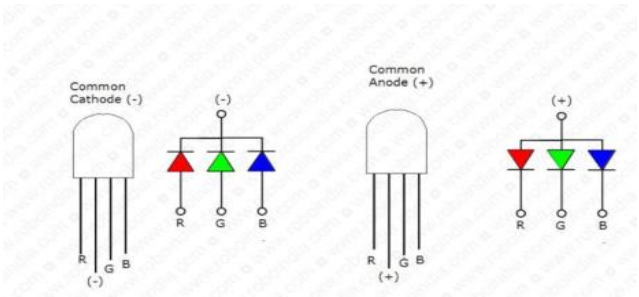
When three different LEDs i.e. Red, Green and blue are merged to one single LED that LED is known as RGB LED. So RGB led comprises 3 LEDs in itself. It could be two types

1. Common Cathode – Cathode(-) is common for all three Red, green and Blue LEDs.

2. Common Anode – Anode(+) is common for all three Red, green and Blue LEDs.

In this tutorial we are dealing with common Anode type LED.

Pin diagram of RGB LED:



1.2 RGB LED Arduino:

We have included a header file to make it easy to control RGB LED on Arduino. It is – RGBled.h.

Funtions of RGB LED Headers.

1. Declartion of RGB object.

RGB *Object_Name;*

You may choose any name for this object, later on in the programme it will be called by the name you declare here.

2. *Object_Name*.**init(9, 10, 11);**

This function initialize RGB Object. You need to tell that RGB pins are connected to which Pins of Arduino. Here 9, 10 and 11 are connected to Red, green and blue respectively.

3. *Object_Name*.**fadeToColor(RED, GREEN, 10);**

This function transforms to Green from Red at speed of 10(speed is decreased with increase in this number).

4. *Object_Name*.**setColor(RED);**

This function sets RGB LED to defined color.

1.2.1 Defined colors in header:

1. *RED*
2. *ORANGE*
3. *YELLOW*
4. *GREEN*
5. *BLUE*
6. *INDIGO*

7. *VIOLET*
8. *CYAN*
9. *MAGENTA*
10. *WHITE*
11. *BLACK*
12. *PINK*

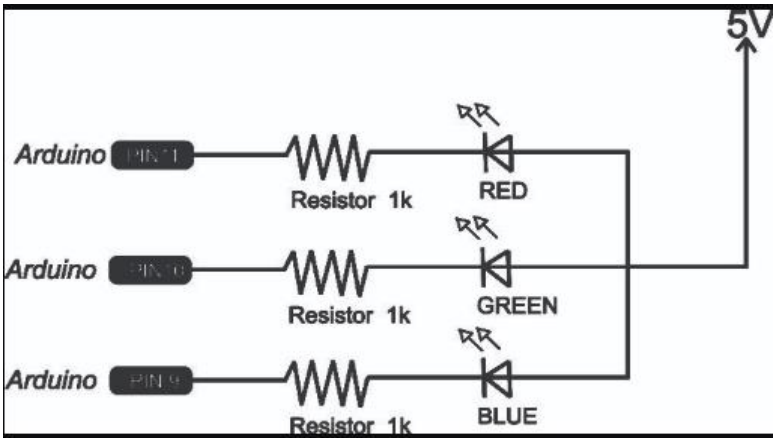
You can call these colors by name in the programme.

2. Required Hardware:

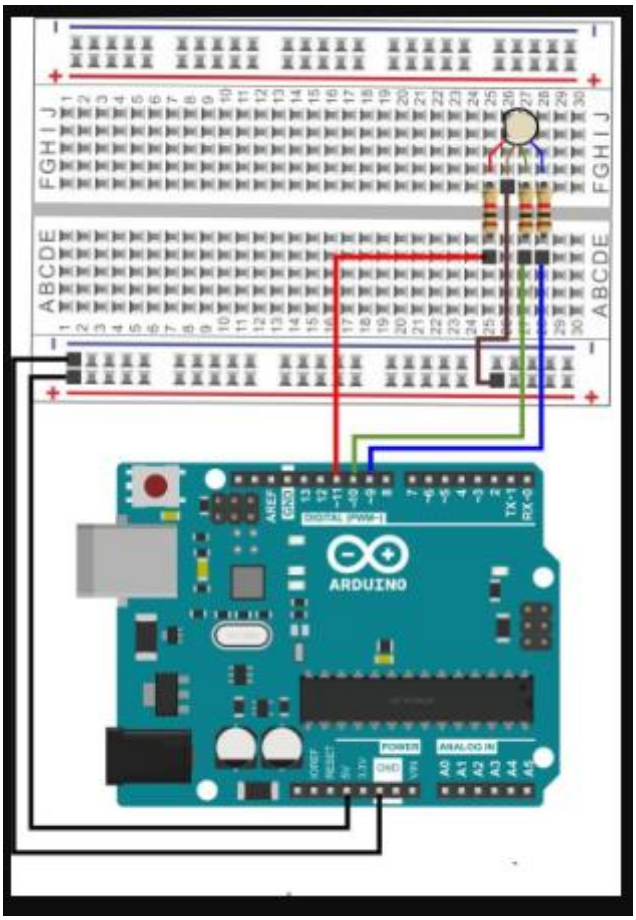
S.No.	Item	Quantity
1	Arduino UNO	1
2	Breadboard	1
3	RGB LED	1
4	Resistor 1k	3
5	Jumper Male to male	6

3. Building Circuit:

Schematic circuit:



Layout of circuit:



4. Programming:

Once we are done with circuit part, here is our programme to this circuit.

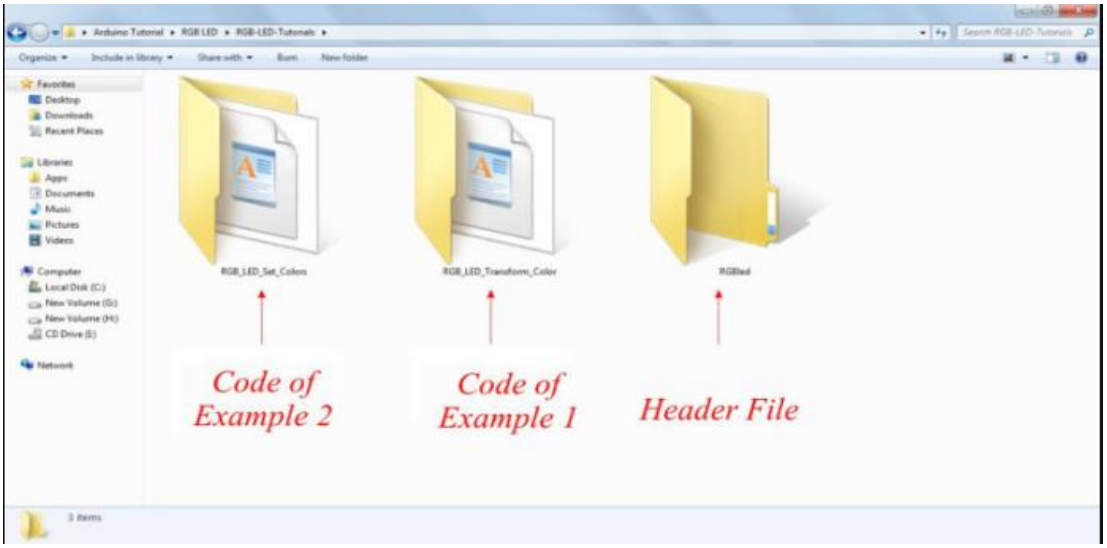
You may download codes (Arduino Sketch) and RGBled Library from here.

https://drive.google.com/open?id=1nGhsQORK5UIlhQi6rjx-kue_CKimq4Uo

4.1 Installation of RGBled Header file-

Extract the downloaded file from above link.

The extracted folder contains three folders in it as mentioned in the below snapshot.



first two folders are codes of examples. The third folder contains Header file. This folder is to be pasted in Library folder of Arduino, it is generally found of the following location-

mydocuments/Arduino/Libraries/

or

similar location for Mac and Linux users.

Once you have pasted this folder to the above mentioned location restart your Arduino Software and you are ready to start with programming.

4.2 Example – 1:

This code has got three function to perform. RGB LED changes its color to three different combination.

```
//Tutorial on RGB LED

// Header file courtesy- www.github.com/jscrane/RGB

#include <RGBled.h> // Header file

RGB myLED; // defining RGB LED object



void setup()

{

  myLED.init(9, 10, 11); // Pin for (Red, Green, Blue)

}



void loop()

{

  myLED.fadeToColor(RED, GREEN, 10); // Transform to green
from Red. 10 is the speed
```

```
  myLED.fadeToColor(GREEN, BLUE, 10);//Transform to Blue
from Red. 10 is speed.


  myLED.fadeToColor(BLUE, RED, 10);  // Tranform to Red
from Blue. 10 is speed.

}



// You may try with another set of colors and speed.
```

4.3. Example -2:

In this example RGB LED is set to different colors, it observes a delay of 1 second between each change.

```
//Tutorial on RGB myLED

// Header file courtesy- www.github.com/jscrane/RGB



#include <RGBled.h> // Header file

RGB myLED; // defining RGB myLED object



void setup()
```

```
{

  myLED.init(9, 10, 11); // Pin for (Red, Green, Blue)

}

void loop()

{

  myLED.setColor(RED); // set.Color function sets the
color.

  delay(1000);          // Halt for one second.

  myLED.setColor(ORANGE);

  delay(1000);

  myLED.setColor(YELLOW);

  delay(1000);

  myLED.setColor(GREEN);

  delay(1000);

  myLED.setColor(BLUE);

  delay(1000);

  myLED.setColor(INDIGO);
```

```
  delay(1000);

  myLED.setColor(VIOLET);

  delay(1000);

  myLED.setColor(MAGENTA);

  delay(1000);

  myLED.setColor(WHITE);

  delay(1000);

  myLED.setColor(PINK);

  delay(1000);

}
```

Output:

After successful uploading of the code,the RGB LED generates two different colors in a time interval of 1 second.Try different set of RGB to get different colors.

Project-6
Arduino Servo Control

This tutorial is about servo control on Arduino. Arduino has got a library for servo control. This tutorial explains how to control servo by using in-built library of Arduino. It has got two examples of servo control to give a better understanding of servo control.

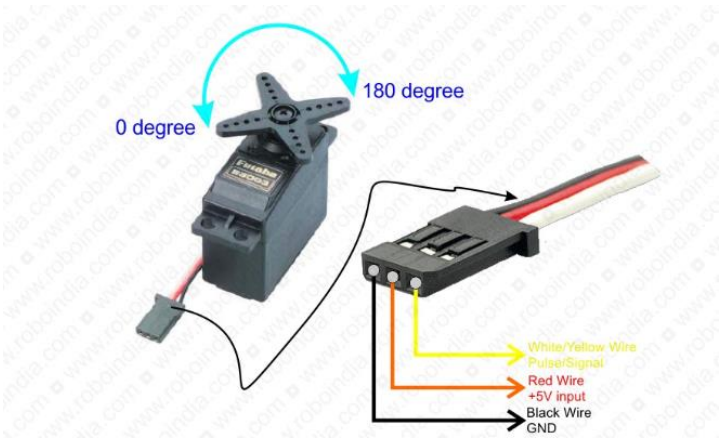
1. Introduction:

A step by step illustrated basic tutorial for Arduino. This tutorial explains Servo motor control through Arduino. We have included two exmaples in this tutorial.

1.1 Servo Motor:

The meaning of servo is feedback. So a servo is an actuator that takes feedback itself and moves precisely. Example of preciseness and feedback can be understood by a daily life example, suppose you pull up/down glass of your car window, You push up/down the power window button and keep on watching the actual position of glass as it reaches to the desired position you releases the button. So this is feedback, we are taking feedback here, if it would be an feedback based system we would have to tell open window by 10%-20%.

Lets come back to servo motor. Servo motor rotate from 0 degree to 180degree. We send the command to servo as it reaches to the commanded value it stops there. The below diagram exhibits how it rotates and its wire interface.



1.2 Servo and Arduino:

Arduino has got a library to control servo. It is – Servo.h.

This library can control above shown servo motors. This library supports up to 12 servos on most Arduino boards and 48 servos on Arduino Mega. It disables analogWrite() for Pin 9 & Pin 10 except Arduino Mega.

2. Required Hardware:

Following Hardware will be required to perform this LED fade in and fade out circuit.

S.No.	Item	Quantity
1.	**Arduino Board**	1
2.	**Bread Board**	1
3.	**Male to male Jumpers**	8
4.	**Micro Servo**	1
5	**10K Potentiometer**	1

3. Building Circuit – 1

This is first example of servo control. Servo alone is controlled through Arduino. In this examples servo takes 5 motions-

1. 0 degree to 45 degree

2. 46 degree to 90 degree

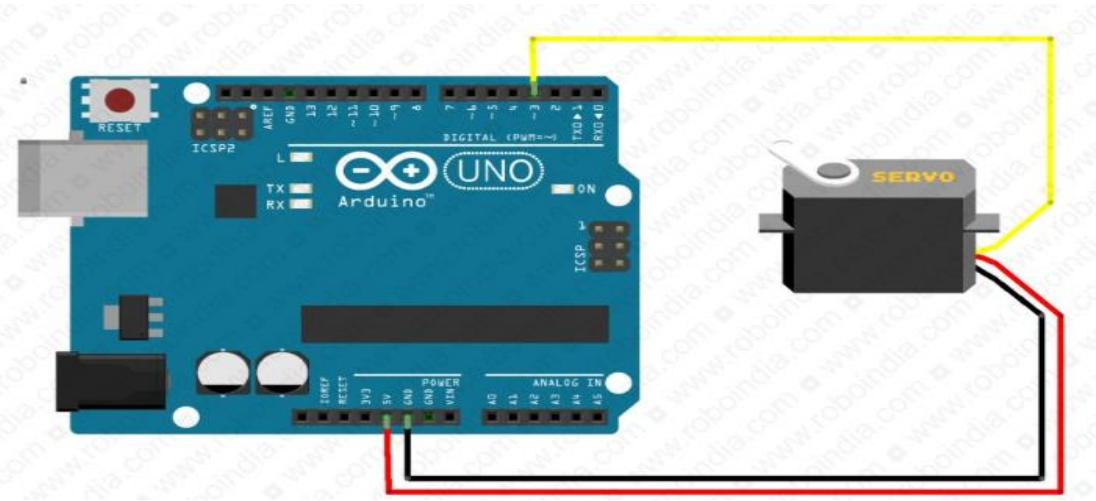
3. 91 degree to 135 degree

4. 136 degree to 180 degree

5. 180 degree to 0 degree (Back to zero)

After completing every movement it stops there for one second.

You may go with original Arduino UNO Board



4. Programming – 1:

Download the code:

https://drive.google.com/open?id=1z20pDhD31z-jVqb7mczl2mYHtZ2CaPVg

Once we are done with circuit part, here is our programme to this circuit.

```
// Servo Control Tutorial#1



// Arduino Servo Library can add up to 12 servo on most of
Arduino boards.

#include <Servo.h>  // Includes servo library.



Servo servo_1;       // Creating Servo object.



int servo_pos = 0;    // Storing servo position (0 degree
to 180 degree)



void setup()

{

  servo_1.attach(3);  // Attaching servo to Pin No.3

}
```

```
void loop()

{



  for(servo_pos = 0; servo_pos <= 45; servo_pos++) // loop
to go to 45 degree from 0 degree.

   {                                    // increment of 1
degree in each step

    servo_1.write(servo_pos);           // commanding
servo to reach at Servo_pos.

    delay(15);                          // waiting a bit
for the servo to reach commanded position.

   }

  delay(1000);                          //Delay of 1
second to observe stoppage of servo.

  for(servo_pos = 46; servo_pos <= 90; servo_pos++)  //
loop to go to 90 degree from 46 degree.

   {

    servo_1.write(servo_pos);
```

```
    delay(15);

   }

  delay(1000);

  for(servo_pos = 91; servo_pos <= 135; servo_pos++)  //
loop to go to 135 degree from 91 degree.

   {

    servo_1.write(servo_pos);

    delay(15);

   }

  delay(1000);

  for(servo_pos = 136; servo_pos <= 180; servo_pos++) //
loop to go to 180 degree from 136 degree.

   {

    servo_1.write(servo_pos);

    delay(15);

   }

  delay(1000);
```

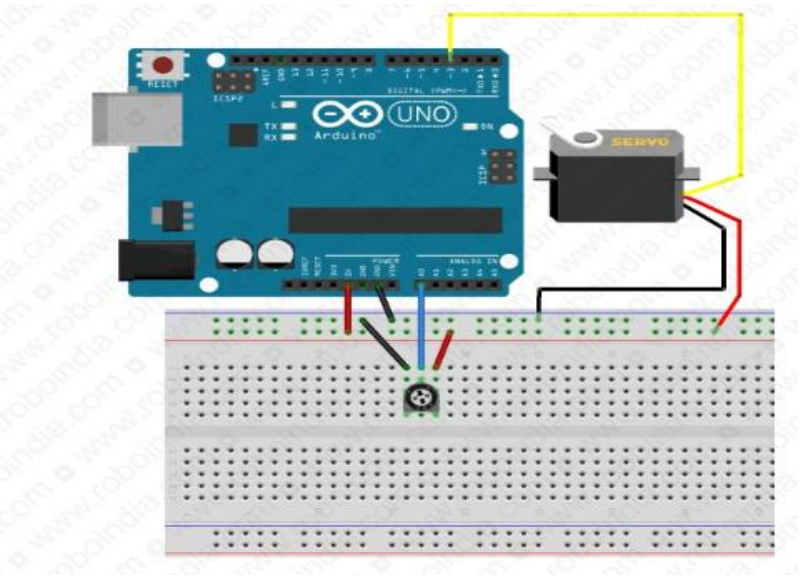
```
  for(servo_pos = 180; servo_pos>=1; servo_pos--)     //
Loop to go back to 0 degree.


   {


    servo_1.write(servo_pos);


    delay(15);


   }


  delay(1000);


}
```

5. Circuit -2:

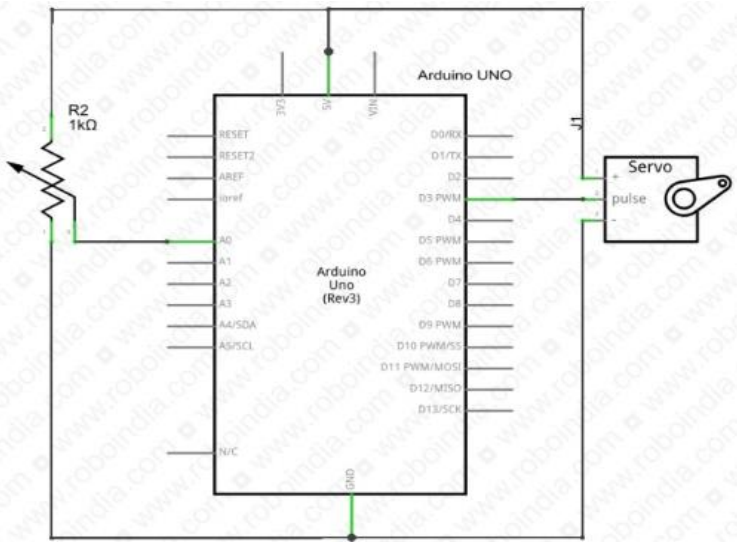
This is second example of Servo Control. In this example We are controlling servo through Analog input. Analog input is taken from a potentiometer. And command of motion is given to the attached servo as per the input we are getting from potentiometer.

Make the following circuit

You may go with original Arduino Board:



Schematic circuit:



6. Programming – 2:

Download the code:

https://drive.google.com/open?id=1SbSrlg1Fk1G3z9Kni4aeSNAkeYRHBEb3

Here is programming for example – 2 (above mentioned circuit).

Points to under stand.

1. Analog input is taken through Potentiometer on Pin A0. It will give us value from 0 to 1023.

2. The servo library need value 0 to 180 to give command to servo.

3. So we will convert our input values (0-1023) to the command value for servo (0-180).

4. This converted value is sent to the servo through Pin 3.

The following programme explains all of the command in comment sections.

```
// Servo Control Tutorial#2

// Arduino Servo Library can add up to 12 servo on most of
Arduino boards.

#include <Servo.h>  // Includes servo library.



Servo servo_1;       // Creating Servo object.



int input_pin = A0;  // Analog Input Pin (Potentiometer)
```

```
int input_val;        // to store analog Input value
(0-1023)


int servo_angle =0;  // Servo angle value (0-180 degree)




void setup()


{


  servo_1.attach(3);  // Atteching Arduino's Pin 3 to
servo.


}


void loop()


{


  input_val = analogRead(input_pin);              // To
read analog input value (0-1023)


  servo_angle = map(input_val, 0, 1023, 0, 179); //
Converting input value (0-1023) to servo angle (0-180)


  servo_1.write(servo_angle);                     //
Commanding servo to reach servo angle


  delay(15);                                      //
waiting for servo to reach commanded angle.


}
```

Output:

As the code executes,servo comes to 45 degree from whatever angle it was.Then there is a delay of one second.After that it goes to 90 degree stays there for one seconds and comes back to 45 degree.This operation is performed in a continous loop.Try this code for different angles.

Project-7

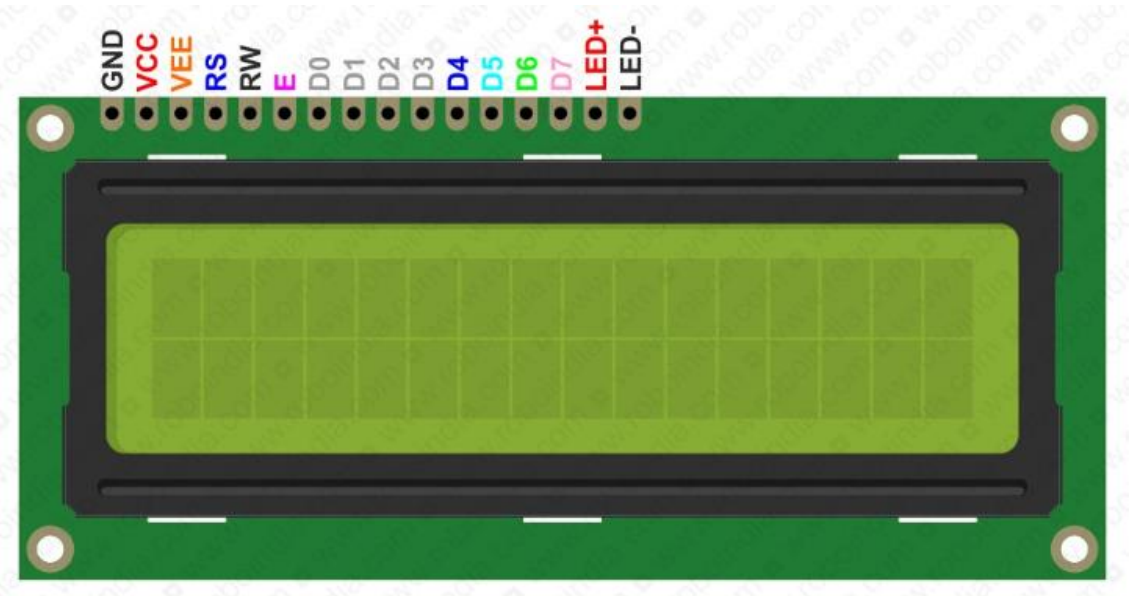
Arduino – 16X2 LCD

This tutorial is to explain how to use a 16×2 character LCD on Arduino board. Tutorial is explained through circuits and sketches. This tutorial is based upon LiquidCrystal Library of Arduino.

1. Introduction:

This tutorial is for 16×2 character LCD (Hitachi HD44780 driver based.) This tutorial uses original LiquidCrystal Library.

1.2 Pinout of 16×2 LCD:



2. Required Hardware

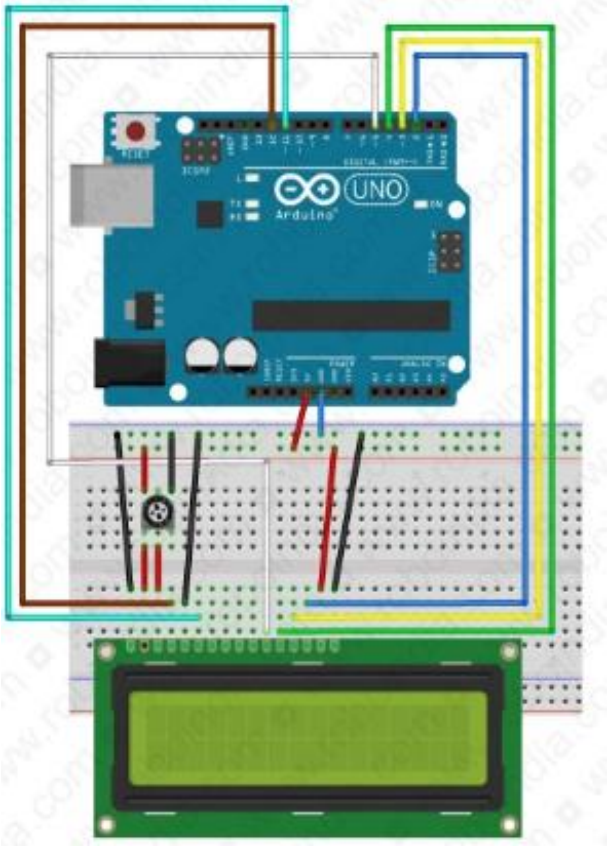
out circuit. Following Hardware will be required to perform this LED fade in and fade

S.No.	Item	Quantity
1.	**Arduino Board**	1
2.	**Bread Board**	1
3.	**Male to male Jumpers**	20
4.	**16×2 LCD**	1
5	**1K Potentiometer**	1
6.	**LCD Board – (LCD Base)***It is optional. If you have this you will not require breadboard and potentiometer.*	1

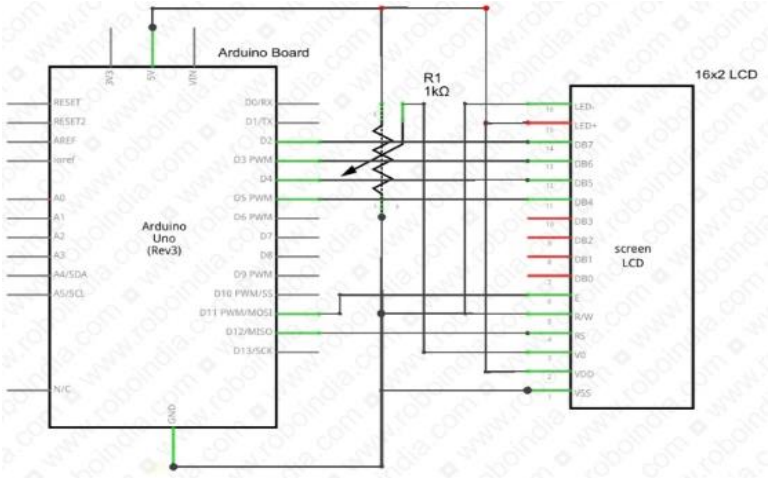
3. Building Circuit:

To run these practical examples make following circuit with the help of above mentioned components.

You may go with original Arduino UNO Board



Schematic circuit:



4. Programming:

Download the code:

https://drive.google.com/open?id=1Qx6hUP71oZrGxUYqNWh_RHL6dpPMk2vO

Once we are done with circuit part, here is our programme to this circuit.

```
/*

16X2 LCD

*/



#include <LiquidCrystal.h> //includes LCD library.




LiquidCrystal MyLCD(12, 11, 5, 4, 3, 2); // Defininig LCD.

int counterVal = 0;  // Variable for counter.



void setup() {

   MyLCD.begin(16, 2); // Initializing 16X2 LCD.

   MyLCD.home(); // Home location : 0,0
```

```
   MyLCD.print("Robo India"); // Print on LCD.

   MyLCD.setCursor(0, 1);

   MyLCD.print("16x2LCD Tutorial");

   delay(2000);



}



void loop() {

  MyLCD.clear();  // Clearing LCD

  MyLCD.home();   // Cursor set to 0,0

  MyLCD.print("Counter");

  MyLCD.setCursor(0,1);     // Setting cursor 2nd row
first column

  MyLCD.print(counterVal);  // Printing counter value.

  counterVal++;             // Increment in counter
value.

  delay(1000);              // delay of 1 second.

}
```

Output:

Arduinjo prints “Hello World” in the first line and “16x2LCD” in the second line.

Project-8

Arduino – LM 35 Temperature Sensor

This tutorial explain to use LM 35 Temperature sensor on Arduino Platform. This tutorial has got two example both of the example reads temperature, one displays it on LCD where as other one displays at serial monitor.

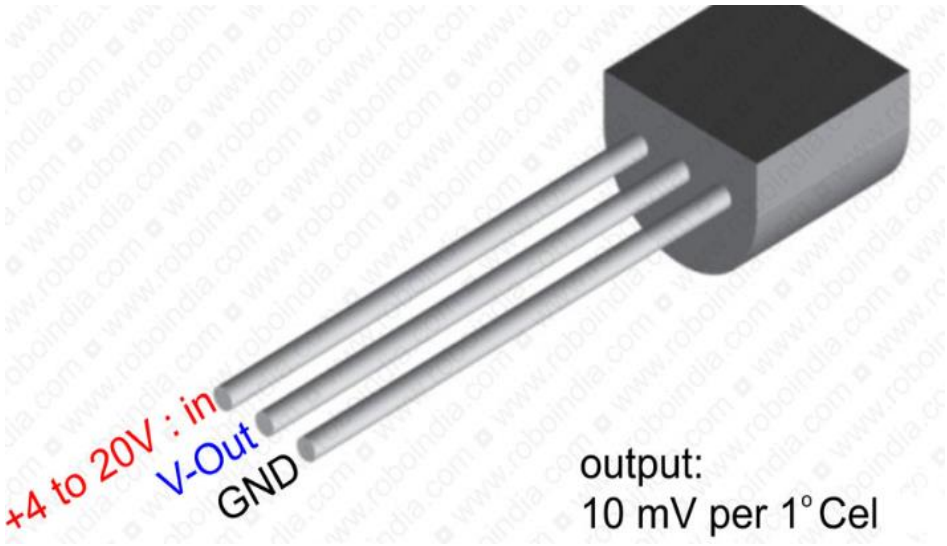
1. Introduction:

This tutorial deals with LM 35 Temperature sensor. It has two examples both read temperature through LM 35 Temperature sensor, one display the temperature on LCD where as other examples displays at serial monitor.

1.2 LM 35 Temperature sensor:

It is a 3 pin IC, Operated on 4 to 20 volt. It gives output in voltage according to the temperature. 10 mili volt per degree Celsius is the output format of this sensor.

1.3 Pinout of LM 35 Temperature Sensor:



2. Required Hardware:

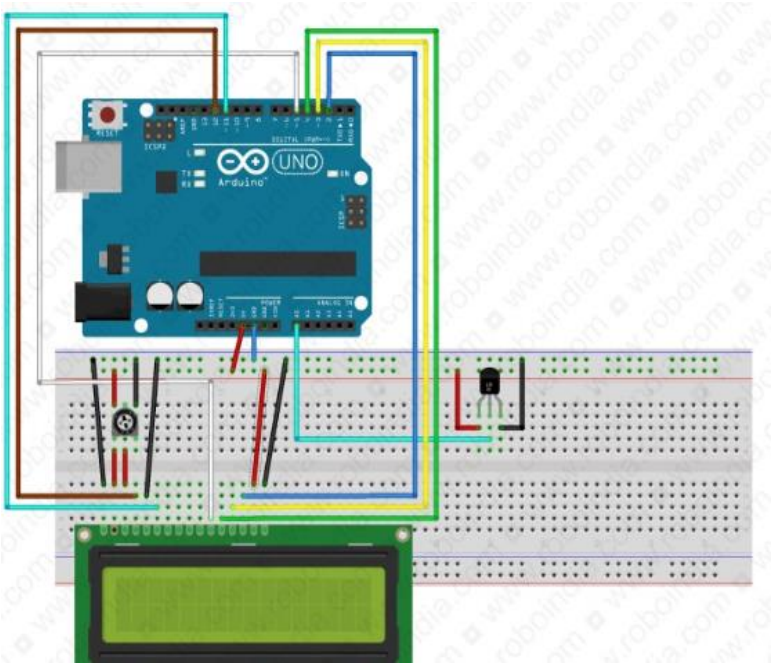
Following Hardware will be required to perform this LED fade in and fade out circuit.

S.No.	Item	Quantity
1.	**Arduino Board**	1
2.	**Bread Board**	1
3.	**Male to male Jumpers**	24
4.	**16×2 LCD**	1
5	**1K Potentiometer**	1
6.	**LM-35 Temperature Sensor**	1

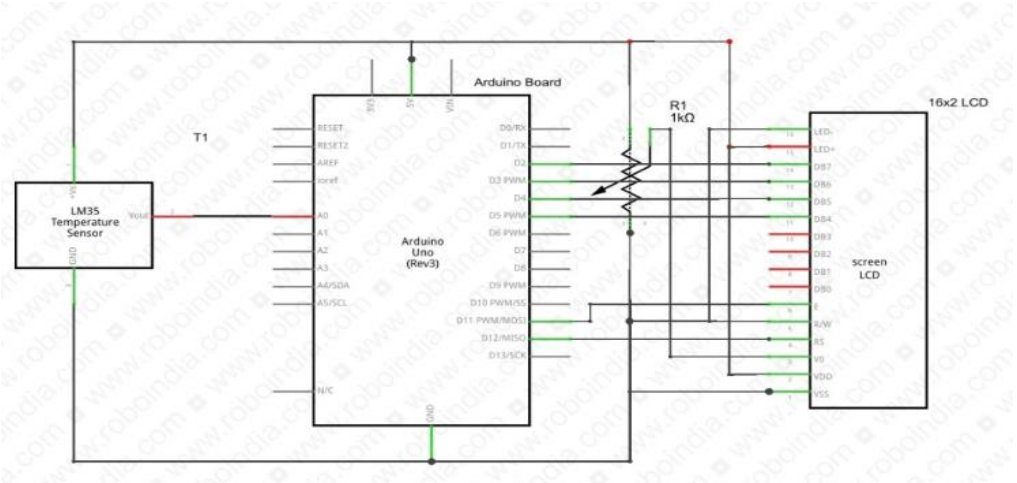
3. Building Circuit – 1 (Displays Temperature on LCD):

To run these practical examples make following circuit with the help of above mentioned components.

You may go with original Arduino UNO Board



Schematic circuit:



4. Programming – 1 (To display temperature on LCD):

Download the code:

https://drive.google.com/open?id=1gOKdnoNM2DavjJg3HN3ZeZcsQ0m9TGOi

Once we are done with circuit part, here is our programme to this circuit.

```
/*

Temerature on 16X2 LCD

LM 35 Temp. IC is used.

*/



#include <LiquidCrystal.h> //includes LCD library.







LiquidCrystal MyLCD(12, 11, 5, 4, 3, 2); // Defining LCD.

int temp_pin = A0;  // LM 35 is connected to A0 Pin.

float temp;         // Variable to store temperature
value.



void setup() {

   MyLCD.begin(16, 2); // Initializing 16X2 LCD.
```

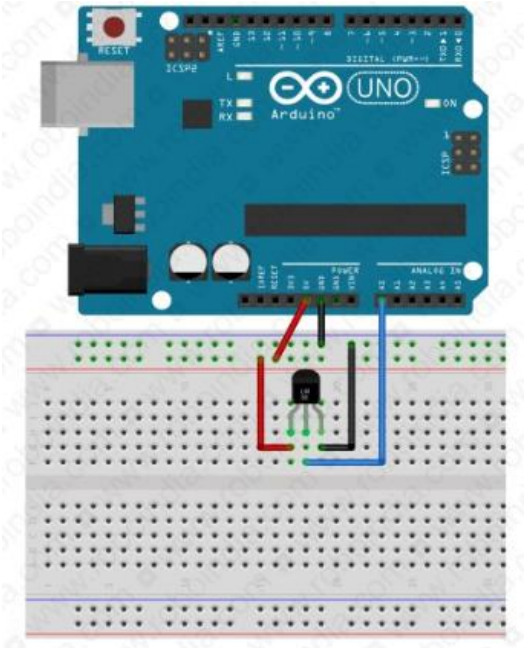
```
   MyLCD.home(); // Home location : 0,0

   MyLCD.print("Robo India"); // Print on LCD.

   MyLCD.setCursor(0, 1);

   MyLCD.print("LM 35 Tutorial");

   delay(4000);

}



void loop() {



  MyLCD.clear();     // Clearing LCD

  MyLCD.home();      // Cursor set to 0,0

  temp = ReadTemp(); // Reading Temperature.

  MyLCD.print("Temperature is-");

  MyLCD.setCursor(0,1);

  MyLCD.print(temp); // Printing Temperature of LCD.

  MyLCD.print(" Deg. C.");
```

```
  delay(1000);        // delay of 1 second.

}



// Function to read temperature.

float ReadTemp(){

int ip_adc_val = 0;

float temp =0;

temp = (5.0 * analogRead(temp_pin) * 100.0) / 1024;

return temp;

}
```

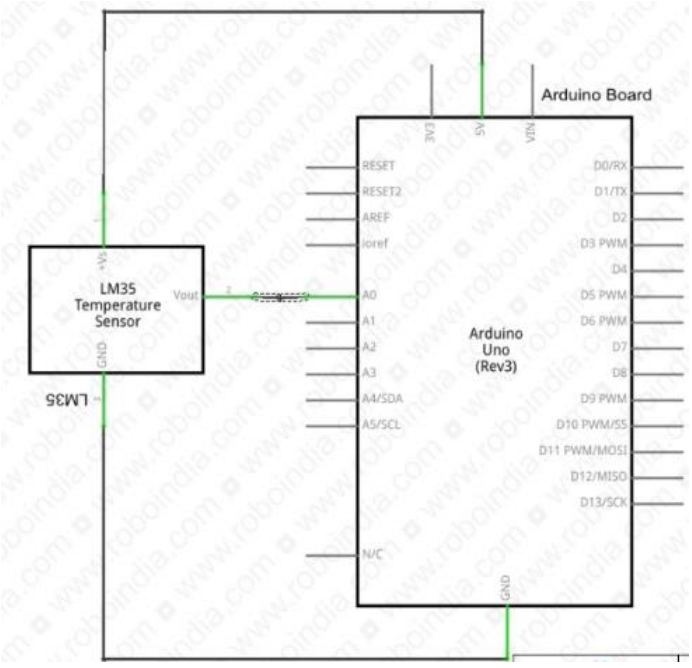
5. Circuit -2 (Displays temperature on Serial Monitor):

You need to make following circuit to run this programme.

You may go with original Arduino Board



Schematic circuit:



6. Programming – 2 (Displays temperature on Serial Monitor):

Download the code:

https://drive.google.com/open?id=11rwf57Y-sXSmP2fOsUX8T3s4B0FCabkX

This is the programme for the above mentioned circuit -2. This sketch is written to read temperature from LM 35 Temperature sensor IC and to display the read temperature at serial monitor. After uploading this programme to the Arduino Board, you will need to start Serial Monitor to see the result. We have included Screen shot of output at serial monitor.

```
/*

Temperature on serial monitor

LM 35 Temp. IC is used.

*/



int temp_pin = A0;  // LM 35 is connected to A0 Pin.

float temp;         // Variable to store temperature
value.



void setup() {

   Serial.begin(9600); // Serial comm. starting.

   Serial.println("Robo India");

   Serial.println("LM 35 Tutorial");
```

```
   delay(2000);

}



void loop() {



  temp = ReadTemp(); // Reading Temperature.

  Serial.print("Temperature is-");

  Serial.print(temp); // Printing Temperature on serial
port

  Serial.print(" degree celsius\n");

  delay(1000);       // delay of 1 second.

}



// Function to read temperature.

float ReadTemp(){

int ip_adc_val = 0;

float temp =0;
```

```
temp = (5.0 * analogRead(temp_pin) * 100.0) / 1024;

return temp;

}
```

Output:

Output of this code is shown on the serial monitor.Temperature is displayed every second on the serial monitor.

Project-9

Arduino – LDR (Light-Dependent Resistor)

This tutorial explains concept and how to use LDR (Light Dependent Resistor) with Arduino. An example is included in which a LED is controlled on the basis of LDR.

1. Introduction:

A step by step illustrated tutorial to use LDR on Arduino.

This is made on Original Arduino

1.1 LDR:

A LDR (Light Dependent Resistor) or a photo resistor is a photo conductive sensor. It is a variable resistor and changes its resistance in a proportion to the light exposed to it. It's resistance decreases with the intensity of light.

2. Required Hardware:

Following Hardware will be required to perform this LDR circuit.

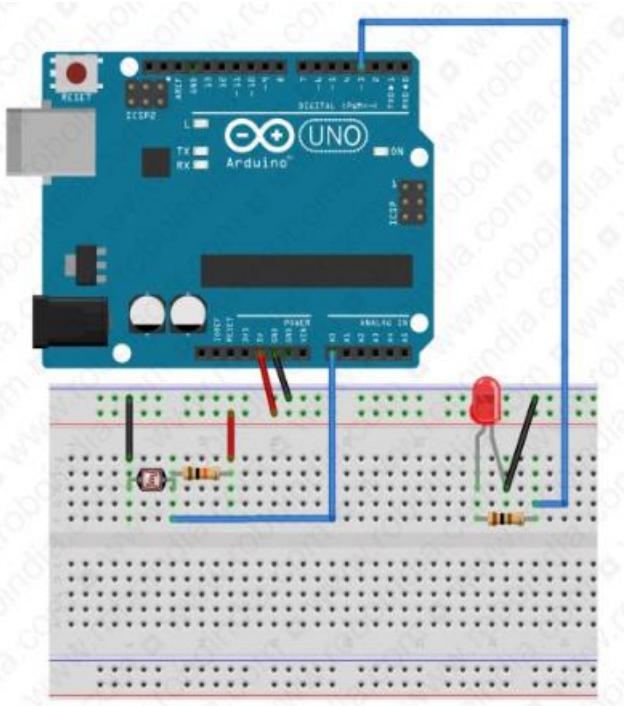
S.No.	Item	Quantity
1.	**Arduino Board**	1
2.	**Bread Board**	1
3.	**Male to male Jumpers**	7
4.	**Indicator LED**	1
5.	**100 OHM Resistance&10 K Resistance**	11
6.	**LDR**	1

3. Building Circuit:

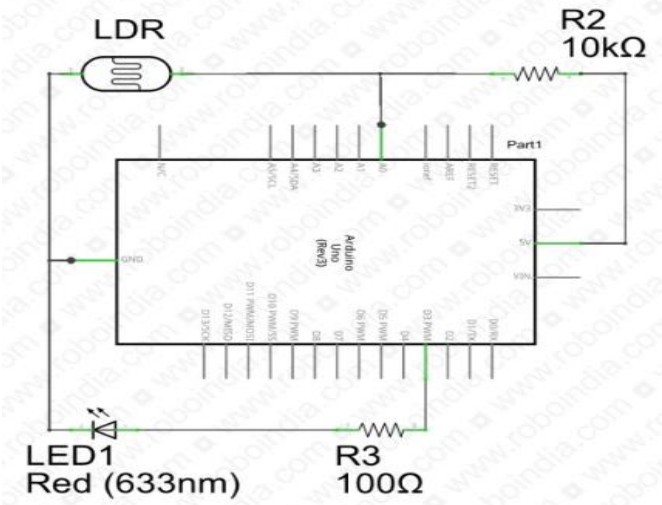
Make following circuit with the help of above mentioned components. Some key points to understand about the circuit-

LDR is connected to a 10 Resistance in series. +5 Voltage is applied to this arrangement. As the light intensity changes LDR value changes thus the voltage drop on LDR will change and we are going to measure that voltage change.

You may go with original Arduino UNO Board



Schematic circuit:



4. Programming:

Download the code:

https://drive.google.com/open?id=1YkKKpdO3Zv1_99Ml--mBKMSPTpBUHGbv

Once we are done with circuit part, here is our programme to this circuit. A few point to consider for this sketch.

1. It reads LDR value and prints them on Serial monitor. Once you upload this programme to your Arduino board open serial monitor and observe how values are changing with the change of Light intensity.

2. The attached LED glows in analog mode according to the LDR Values.

3. There is a condition of threshold; The attached LED remains OFF for all the values below Threshold limit. You can set your own threshold limit. In this programme we have given 800 as threshold. You can set t threshold to any value between 0 and 1023.

```
int LDR = A0;              // LDR input at A0 pin.

int LED = 3;              // LED is connected to PWM Pin 3.

int LDRReading = 0;       // to store input value of LDR

int lEDBrightness = 0;    // to store the value of LED
Brightness

int threshold_val = 800; // Check your threshold and modify
it.



void setup(){
```

```
  Serial.begin(9600);     // initializing serail
communication.

  pinMode(LED, OUTPUT);   // Defining LED pin as output.

}



void loop(){

  LDRReading = analogRead(LDR);    // Reading LDR Input.

  Serial.println(LDRReading);      // Printing LDR input
value.



  if (LDRReading >threshold_val){                    //
Condition to make LED ON.

  lEDBrightness = map(LDRReading, 0, 1023, 0, 255); //
Converting LDR to LED Brightness.

  analogWrite(LED, lEDBrightness); // Writing Brightness
to LED.

  }

  else{

  analogWrite(LED, 0);             // If LDR is below
threshold make LED OFF.
```

```
  }




  delay(300);                          // delay to make
output readable on serial monitor.


}
```

Output:

Values of LDR depend upon the light exposed on it.These values are displayed on the serial monitor.

Project-10

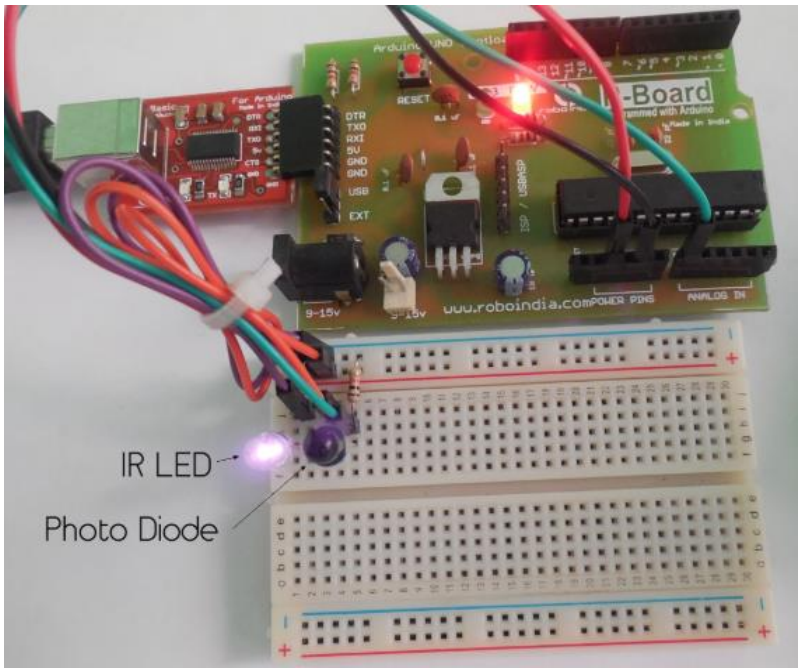
Arduino – IR Proximity and color detection sensor

This tutorial of Robo India explain the working concept of Infrared(IR) sensor for color detection and proximity. The constructed circuit is performed on Arduino platform.

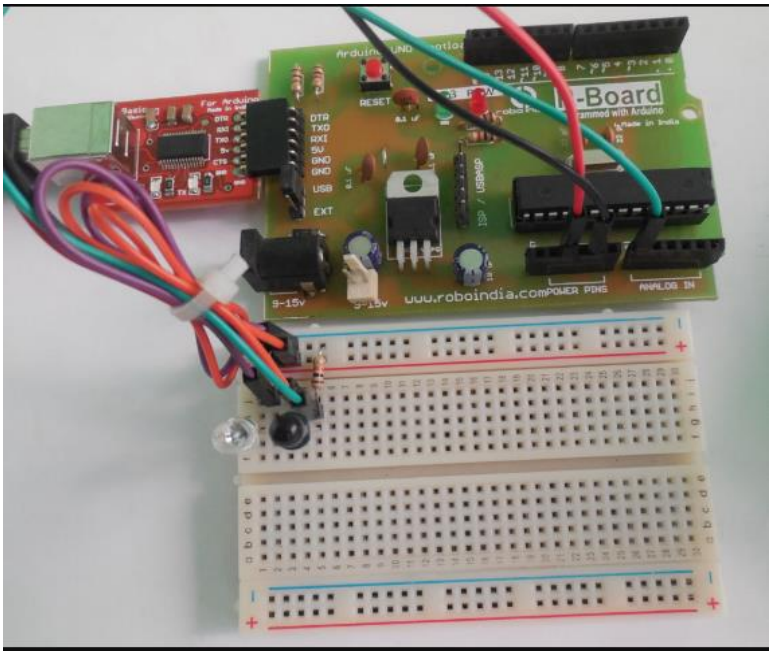
1. Introduction:

A step by step illustrated tutorial to use IR Sensor on Arduino. This tutorial explains the concept and construction of IR Sensor for color detection and proximity measurement. This sensor comprises following two main parts (IR-LED and Photo Diode).

1.1 IR LED (Infrared Transmitter): It is as same as other LED we generally see, but it emits light of Infrared range 700 nanometers (nm) to 1 mm. This light is not visible through naked eyes but can be seen by camera (that is why these are also used in night vision camera). Following image shows how IR is seen on camera. In this image IR LED is ON.



In the following image IR LED is OFF(This is how it is seen by naked eyes). So camera is a good option to test whether IR LED is working or not.

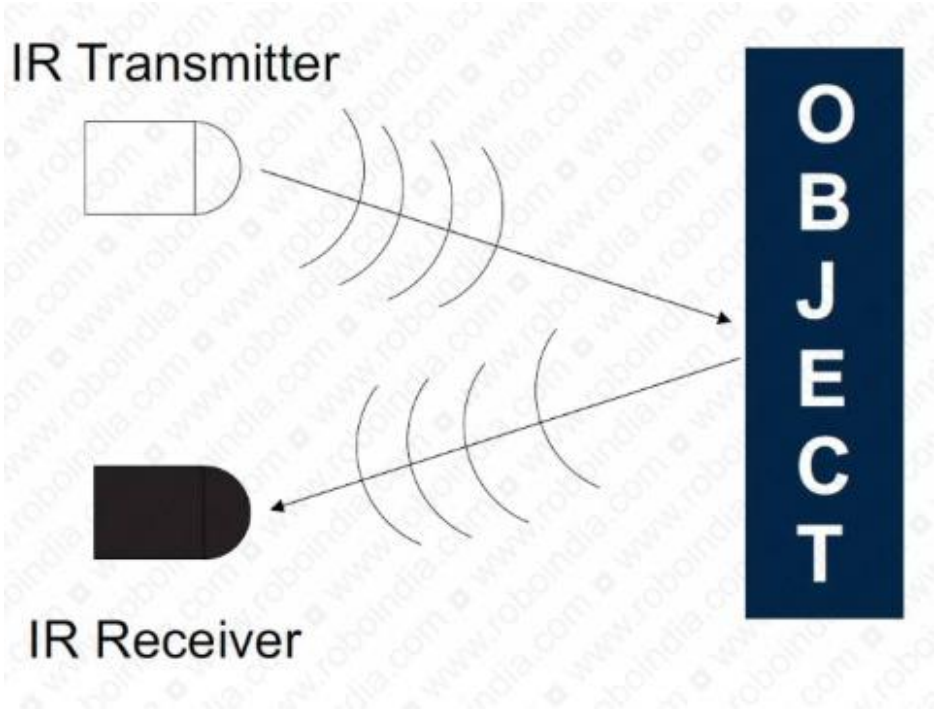


1.2 Photo Diode: It gives response in term of change in resistance when light falls on it. That change we measure in terms of voltage.

1.3 IR Sensor as a proximity sensor:

IR LED (Transmitter) emits IR light, that light get reflected at the object, the reflected light is received by IR receiver (Photo Diode). Amount of reflection and reception as the distance varies. This difference causes to change in input voltage through IR input. We observe these values and use it out projects. This is our Proximity sensor. Following diagram explains this concept.

Application of this sensor can be an obstacle avoiding robot.

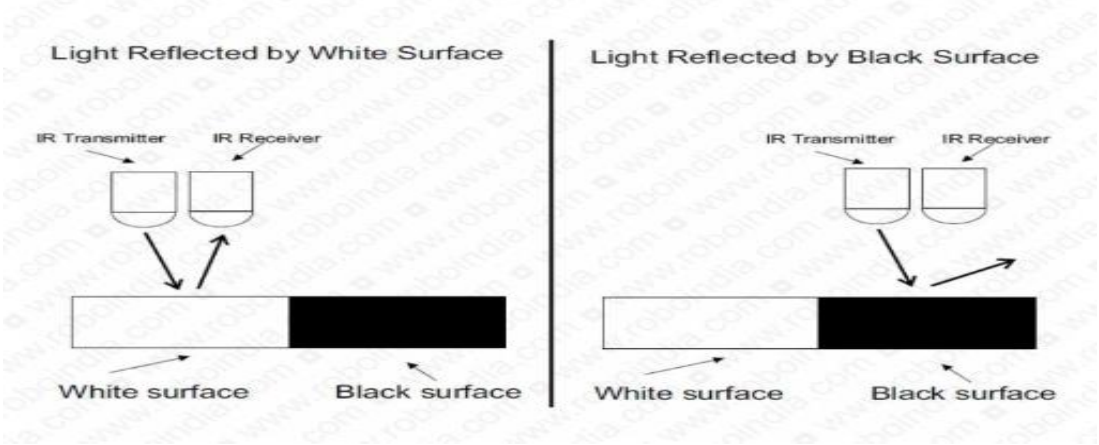


1.4 IR sensor as color detector:

Lets see above mentioned concept from a different angle. The amount of reflection light will also depend upon the color of surface from which it is being reflected. Black is said to be perfect absorber and white is to be said perfect reflector. The reflection will be different for different colors. Thus make it a color detector.

Application of this sensor can be a line follower, a micro mouse or a grid solver.

Following diagram will help you to understand the concept of IR sensor as a color detector.



2. Required Hardware:

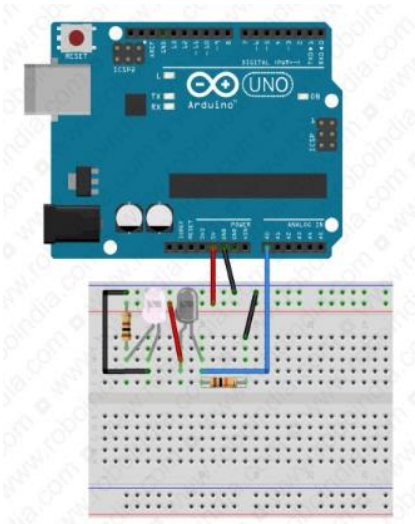
Following Hardware will be required to perform this sketch of shift register.

S.No.	Item	Quantity
1.	**Arduino Board**	1
2.	**Bread Board**	1
3.	**Male to male Jumpers**	5
4.	**100 OHM Resistance 10K OHM Resistance**	11
5.	**IR LED Photo Diode Pair**	1

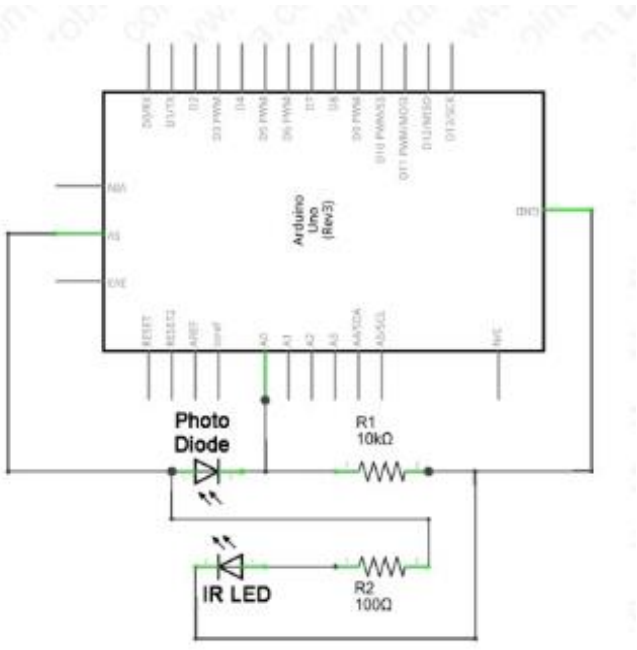
3. Building Circuit:

Make following circuit with the help of above mentioned components.

You may go with original Arduino UNO Board



Schematic circuit:



4. Programming:

Download the code: https://drive.google.com/open?id=1ocHQpb1fk9ciYETmflK3-3cml1Pv6pZy

Once we are done with circuit part, here is our programme to this circuit.

Programming is simple Analog input programming. Input is coming from Photo Diode to Pin A0. Upload the programme to Arduino board and observe change in value at serial port. These values are useful while making an application using this IR sensor.

```
/*

Tutorial of Analog IR Sensor.

 */



int analog_ip = A0;    // select the input pin Photo Diode.

int inputVal = 0;      // to store photo diode input




void setup() {

  Serial.begin(9600);    // Setup Serial Communication.

  Serial.print("ROBO INDIA\nroboindia.com\nTutorial on
Analog IR Sensor.\n");

}
```

```
void loop(){

    inputVal = analogRead(analog_ip); // Reading and
storing analog input value.




        Serial.print("Input Value:");

        Serial.print(inputVal);       // Printing Analog
input value of Photo Diode.

        Serial.print("\n");           // moving to new
line

        delay(500);                   // Waiting for a
while.









}
```

Output:

Place an object in front of the IR LED & Photo diode.The analog input values from photo diode are displayed on the serial monitor.Try using different colored surfaces and varying distance of object from the IR LED & Photo diode.

Project-11
Transistor and Relay with Arduino

Button switch. This tutorial explains following concept, Transistor Switching, Working of Relay, Switching with relay on Arduino, Motor control on Arduino using Relay.

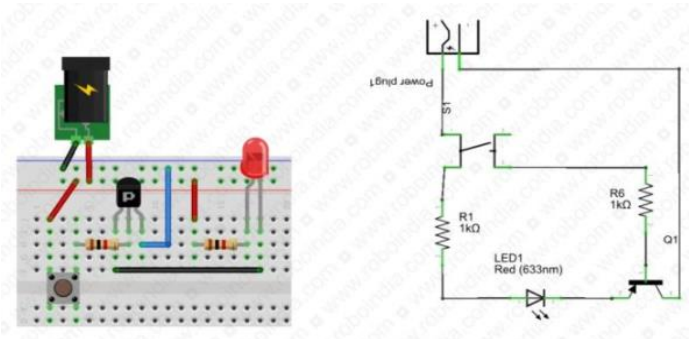
1. Introduction:

This tutorial explains following things-

1. Switching with NPN transistor.
2. Working and concept of Relay
3. Switching by relay.
4. Relay and Arduino
5. Motor control by Relay on Arduino

1.2 Switching by NPN Transistor:

make the following circuit. When you give High signal on Input it connects the LED to GND. Here we are taking HIGH input by push



1.3 46ND006-P:

This is the relay we are going to use. This relay has got two poles, means there are two switches but both are trigged by a single Coil thus they

operate simultaneously. Following Diagram will give you better understanding about the relay we are using.

2. Required Hardware:

Following Hardware will be required to perform the example of this tutorials.

S.No.	Item	Quantity
1.	**Arduino Board**	1
2.	**Bread Board**	1
3.	**Male to male Jumpers**	20
4.	**1 K Resistance**	6
5	**5 Volt – 2 Pole relay.46ND006P**	1
6.	**NPN Transistor**	1
7.	**Indicator LEDIn three different colors.**	5
8.	**Hobby DC Motor**	1

3. Building Circuit – 1 (Understanding of Relay):

This circuit is to make you understand the concept of relay. Our relay has got two poles, thus we have connected 4 LED to this. Two at Normally Connected pins and two are normally open pins. One status LED is also there. The Status LED tells that Relay is on of Off. If Status LED is on it means that relay is on (Common is connected to NC.) and vice versa.

Red LED- Statuts LED of Relay

Yellow LED – Connected to NC terminal of Relay

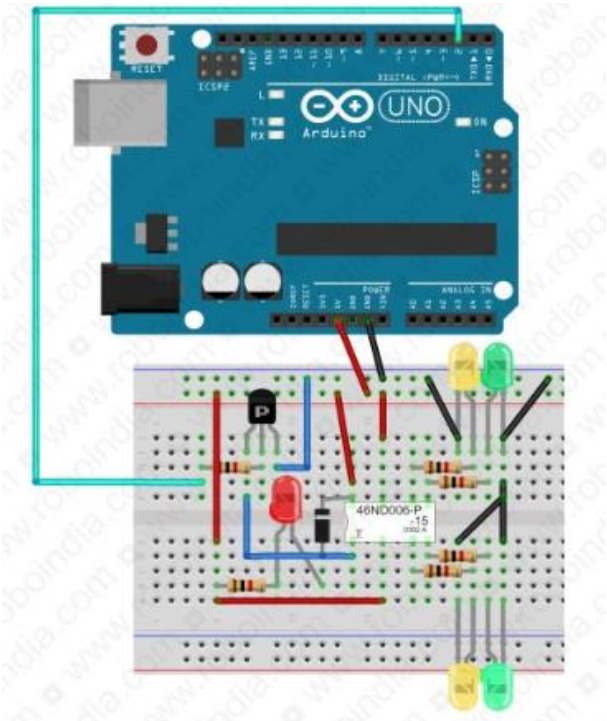
Green LED – Connected to NO terminal of Relay

Conclusion we can draw –

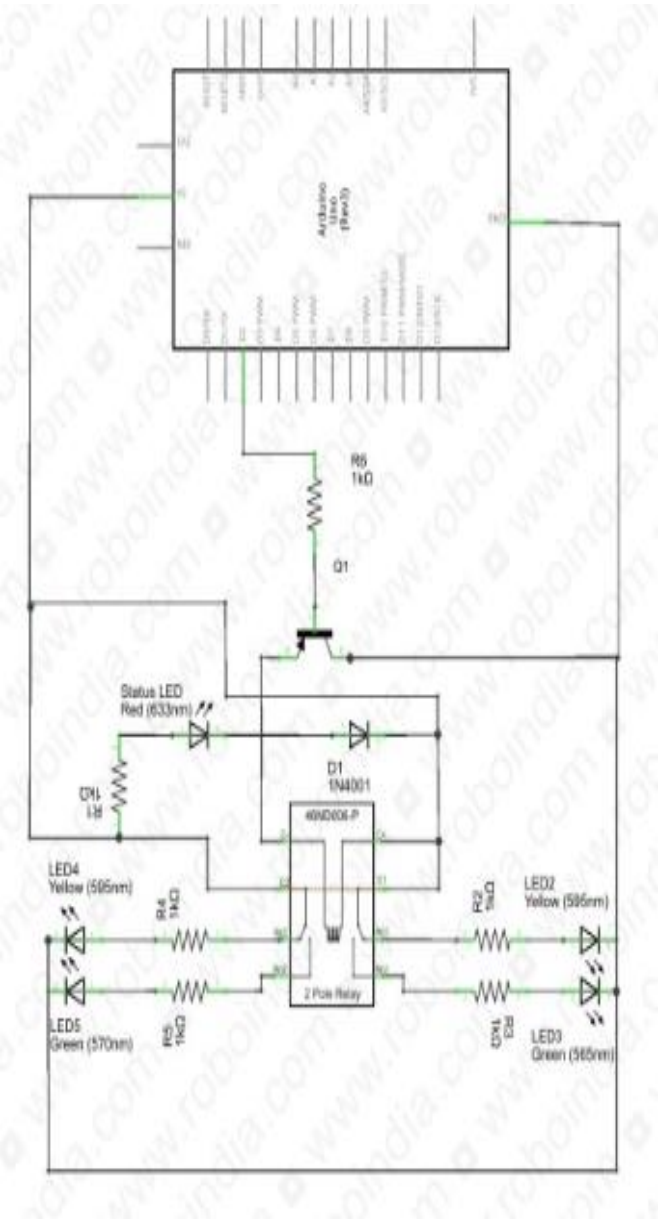
If red is off – means relay is off – means – Common terminals of Relay are connected to NC – Thus Yellow LEDs are ON.

If red is ON – means relay is ON – means – Common terminals of Relay are connected to NO – Thus Green LEDs are ON.

You may go with original Arduino UNO Board



Schematic circuit:



4. Programming:

Download the code:

https://drive.google.com/open?id=1cJXwjuzQ4sbwLak7T1pwsM76W7TdpoES

We don't need any special programming to operate Relay of Transistor, Simple Digital output programming is required. So the programme we have added here is as same as in our other tutorial of ***Digital Output – LED Blinking.*** The same codding is used throughout the tutorial.

```
// Digital output is taken on a LED that remains ON for one
second and


// OFF for another.




// Defining Pin 2 as LED.


const int LED = 2;  // from the circuit we can see that we
have connected LED on Pin 2









void setup() {


    pinMode(LED, OUTPUT); // Defining LED pin as OUTPUT
Pin.


}
```

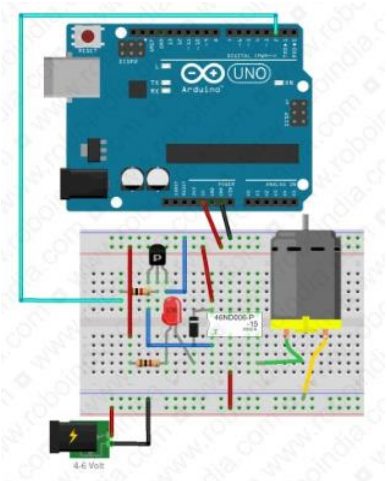
```
// Below mentioned code runs for ever(infinite loop)

void loop() {

  digitalWrite(LED, HIGH); // LED gets turned ON
(1/HIGH/+5V)

  delay(1000);              // Waiting for one second.

  digitalWrite(LED, LOW);  // LED gets OFF (0/LOW/0V/GND)

  delay(1000);         // here and above Delay is in mili
second (1000 = 1 second)

}
```

5. Circuit -2 (Motor control on Arduino using Relay):

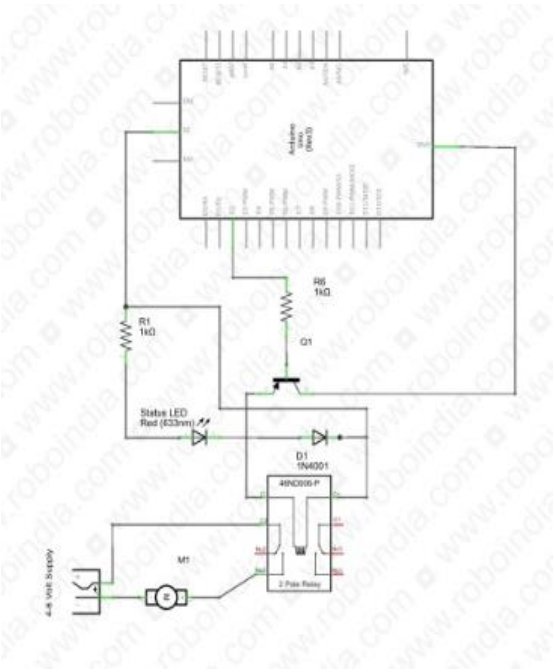
As we have mentioned earlier that the codding is same for the entire tutorial. Here we are controlling one DC motor on Arduino board. Motor consumes more power than the other circuit thus you will need separate Power supply to run a motor. It will not run on Laptop/PC's USB.

The circuit you will need is here

You may go with orginal Arduino Board



Schematic circuit:



output:

The motor rotates for 1 second and then stops,remains stopped for 1 second.This loop continues.

Project-12

Arduino – Shift Register (Serial In Parallel Out)

This tutorial of Robo India explains how to extend output pins of Arduino Board. It is major constrain of Arduion. But with the help of this tutorial you can increase output pins of Arduino Board.

1. Introduction:

A step by step illustrated tutorial to extend output pins of Arduino. This tutorial uses 74HC595N shift Register for Serial in and Parallel out.

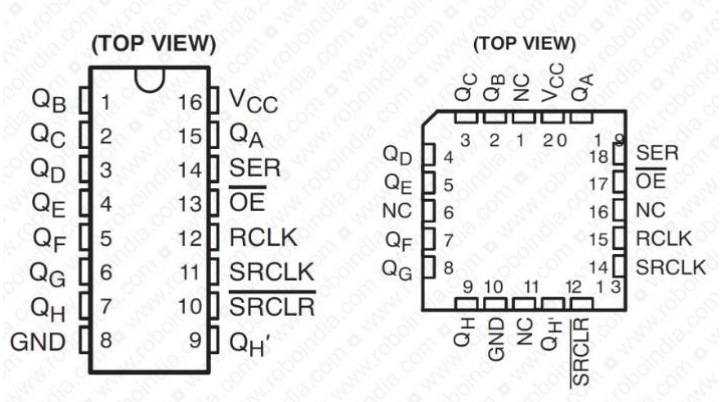
This register IC takes 3 pin input and gives output at 8 Pins. Thus this extends 3 pins to 8 pins.

1.1 SN74HC595N Shift Register:

This is a popular shift register we are using in this tutorial. It requires a three pin interface from Arduino

1. Data pin: Data is sent in serial mode.
2. Clock Pin: A clock runs on this pin
3. Latch Pin: This pin is used to toggle so that shift register shows 8 bit data on output.

Following is the pinout diagram of SN74HC595N shift Register



2. Required Hardware:

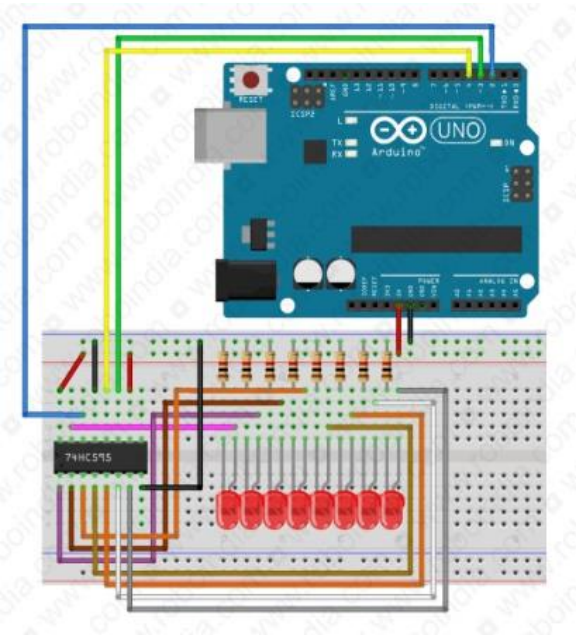
Following Hardware will be required to perform this sketch of shift register.

S.No.	Item	Quantity
1.	**Arduino Board**	1
2.	**Bread Board**	1
3.	**Male to male Jumpers**	24
4.	**Indicator LED**	8
5.	**100 OHM Resistance**	8
6.	**SN74HC595N Shift Register**	1

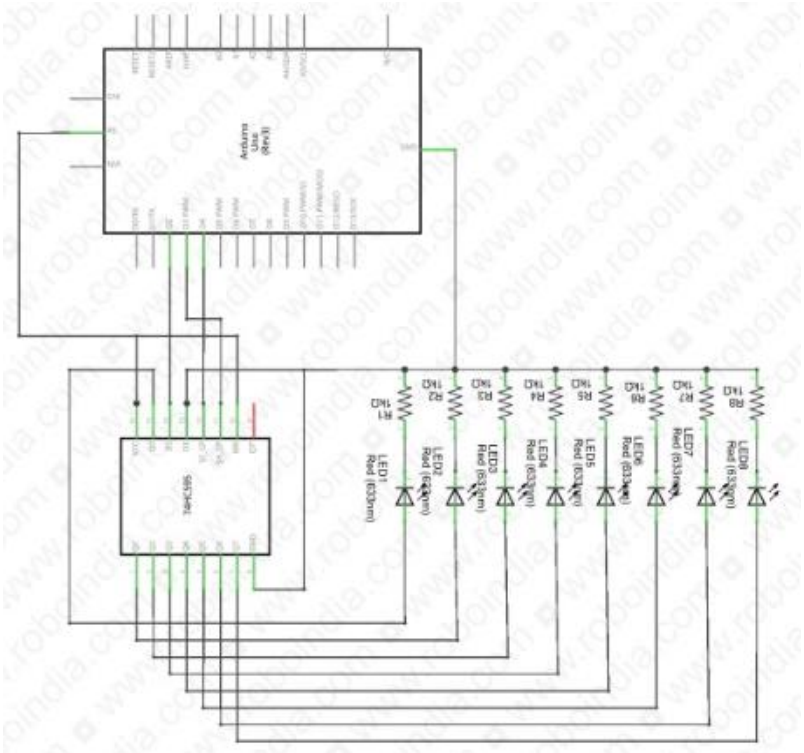
3. Building Circuit:

Make following circuit with the help of above mentioned components.

You may go with original Arduino UNO Board



Schematic circuit:



4. Programming:

Download the code:

https://drive.google.com/open?id=1mKhZ0hbapfulB1ItnlBPXvqEo3yIYbUU

Once we are done with circuit part, here is our programme to this circuit. A few function to consider for this sketch.

4.1 shiftWrite(Pin, State): This function is same as ***digitalWrite*** function. It makes Pin HIGH/LOW. Usage is as same as the digitalWrite function.

4.2 increament() : This function is designed for LED array on shift register, LED starts glowing from LED 0 to LED 7 and as all gets ON these starts getting OFF from LED 7 to LED 0.

4.2OneByOne(): This function is similar to above mention ***increment()*** function but the difference is that in this function only one LED glows at a time. So in this function LED starts glowing from LED 0 to LED 7 and in the reverse order too.

4.3. AllHigh(): This function makes all output pins HIGH.

4.4 AllLow(): This function makes all output pins LOW.

4.5 SOS(): This function repeats AllHigh() and AllLow() function 10 times with an interval of 100ms between two steps.

```
int DataPin = 2;  // Data Pin is connected to Pin No. 2

int ClockPin = 3; // Data Pin is connected to Pin No. 3

int LatchPin = 4; // Data Pin is connected to Pin No. 4



byte Data = 0;  // 8 Bit Data to be sent through DataPin
```

```
void setup()
{
  pinMode(DataPin, OUTPUT);   // All 3 pins are output
  pinMode(ClockPin, OUTPUT);
  pinMode(LatchPin, OUTPUT);
}



void loop()
{


  increment(); // LEDs Increament start from 0 - 7
  delay(2000);
  SOS();         // All LEDs ON and OFF 10 times
  OneByOne();   // LEDs Glow one by one from 0 to 7
```

```
  delay(2000);









}



// Function defined below-



void shiftWrite(int Pin, boolean State) // Function is
similar to digitalWrite

{                                       // State-0/1 |
Pin - Pin No.

  bitWrite(Data,Pin,State);             // Making
Pin(Bit) 0 or 1

  shiftOut(DataPin, ClockPin, MSBFIRST, Data); // Data out
at DataPin

  digitalWrite(LatchPin, HIGH);                 //
Latching Data

  digitalWrite(LatchPin, LOW);

}
```

```
void increament()   //LEDs increment start from 0 - 7

{

  int PinNo = 0;

  int Delay = 100;



  for(PinNo = 0; PinNo < 8; PinNo++)

  {

    shiftWrite(PinNo, HIGH);

    delay(Delay);

  }

  for(PinNo = 7; PinNo >= 0; PinNo--)

  {

    shiftWrite(PinNo, LOW);

    delay(Delay);

  }

}
```

```
void OneByOne()  // LEDs Glow one by one from 0 to 7
{
  int PinNo = 0;
  int Delay = 100;

  for(PinNo = 0; PinNo < 8; PinNo++)
  {
    shiftWrite(PinNo, HIGH);
    delay(Delay);
    shiftWrite(PinNo, LOW);
  }
  for(PinNo = 7; PinNo >=0 ; PinNo--)
  {
    shiftWrite(PinNo, HIGH);
    delay(Delay);
```

```
    shiftWrite(PinNo, LOW);

  }



}



void AllHigh()   // sets all High

{

  int PinNo = 0;

  for(PinNo = 0; PinNo < 8; PinNo++)

  {

   shiftWrite(PinNo, HIGH);

  }

}



void AllLow()   // Sets all low

{
```

```
  int PinNo = 0;

  for(PinNo = 0; PinNo < 8; PinNo++)

  {

   shiftWrite(PinNo, LOW);

  }

}



void SOS(){                    // All LEDs ON and OFF 10
times

  for (int x=0; x<10; x++){

  AllHigh();

  delay(100);

  AllLow();

  delay(100);

  }

}
```

Output:

All 8 LEDs blink together at an interval of 1 second.In this code the entire output is written at once.Pin 5 of shift register blinks at an interval of 1 second.Try to make some pattern on LEDs using multiple *shiftWrite()* commands.

Project-13

Arduino – Peizo Buzzer

This tutorial of Robo India is on Peizo Buzzer. Here two examples are included one plays beep sound and other one plays a song.

1. Introduction:

This tutorial of Robo India expalins how to use a Peizo Buzzer on Arduino. When voltages are supplied to its terminal it generates beep sound. This buzzer is used to give beep sound to various embedded systems.

2. Required Hardware:

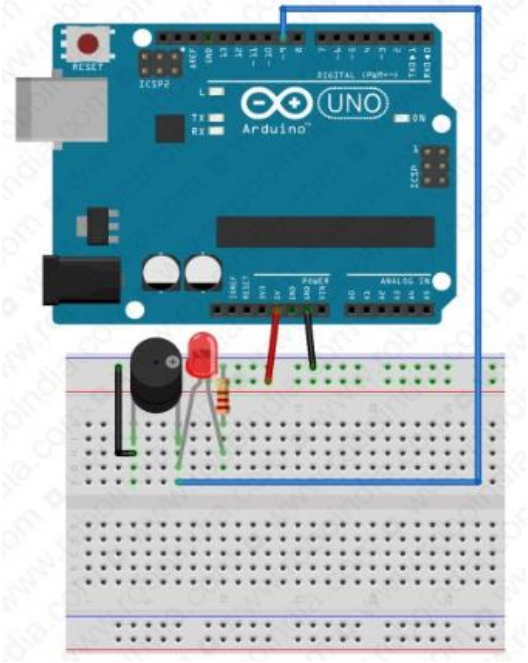
Following Hardware will be required to perform this sketch of shift register.

S.No.	Item	Quantity
1.	**R-Board with FTDI or Arduino Board**	1
2.	**Bread Board**	1
3.	**Male to male Jumpers**	5
4.	**100 OHM Resistance**	1
5.	**5 Volt Buzzer**	1

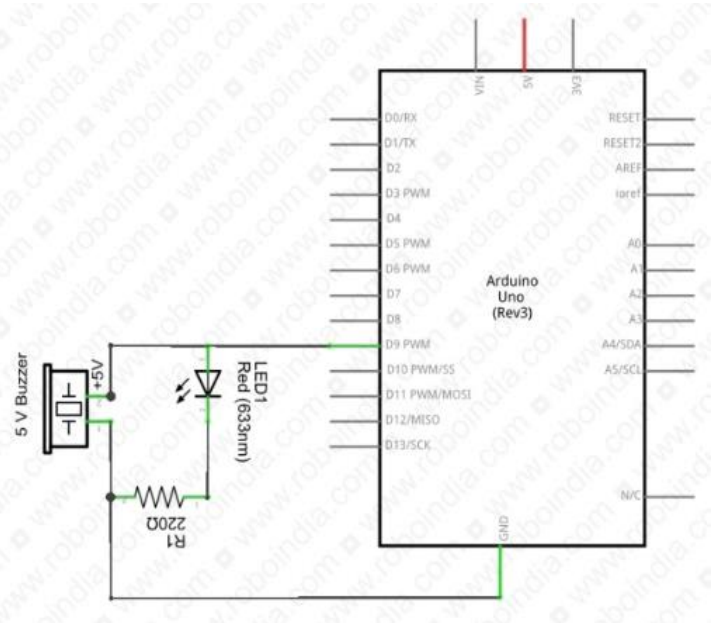
3. Building Circuit:

Make following circuit with the help of above mentioned components. We have included two examples here both of them use the following circuit.

You may go with original Arduino UNO Board



Schematic circuit:



4. Programming 1 – Playing a beep sound.

Download the code:

https://drive.google.com/open?id=10XgDqx5GLOFmzrf-h5DlpiDe7yXsqWWA

Once we are done with circuit part, here is our programme to this circuit.

```
/*

Beep sound generation Tutorial

*/




const int Buzzer = 9;



void setup()

{

  pinMode(Buzzer, OUTPUT);

}



void loop()

{
```

```
  digitalWrite(Buzzer, HIGH);

  delay(400);

  digitalWrite(Buzzer, LOW);

  delay(2000);

}
```

5. Programming 2 – Playing a song:

Download the code:

https://drive.google.com/open?id=1bI5yaHd9a9Muqo2TM-oxscbNqjkjnEub

Cicruit is same as above mentioned. Copy this code and upload to your Arduino board.

```
/*

Tone generation Tutorial

 note       frequency

  c     262 Hz

  d     294 Hz

  e     330 Hz

  f     349 Hz
```

```
  g     392 Hz

  a     440 Hz

  b     494 Hz

  C     523 Hz

*/



const int Buzzer = 9;

const int songLength = 18;



char notes[] = "cdfda ag cdfdg gf "; // a space represents
a rest

int beats[] = {1,1,1,1,1,1,4,4,2,1,1,1,1,1,1,4,4,2};

int tempo = 150;  // Speed of tempo



void setup()

{

  pinMode(Buzzer, OUTPUT);
```

```
}



void loop()

{

  int i, duration;



  for (i = 0; i < songLength; i++) // step through the song
arrays

  {

    duration = beats[i] * tempo;  // length of note/rest
in ms



    if (notes[i] == ' ')          // is this a rest?

    {

      delay(duration);            // then pause for a
moment

    }

    else                          // otherwise, play the
note
```

```
    {

      tone(Buzzer, frequency(notes[i]), duration);

      delay(duration);             // wait for tone to
finish

    }

    delay(tempo/10);               // brief pause between
notes

  }









  while(true){}     // Remove this line if you want to play
this song for ever.

}









int frequency(char note)

{

  int i;
```

```
  const int numNotes = 8;       // number of notes we're
storing

  char names[] = { 'c', 'd', 'e', 'f', 'g', 'a', 'b', 'C'
};

  int frequencies[] = {262, 294, 330, 349, 392, 440, 494,
523};




  for (i = 0; i < numNotes; i++)

  {

    if (names[i] == note)

    {

      return(frequencies[i]);

    }

  }

  return(0);

}
```

Output:

The buzzer generates beep sound in a loop.Try delay in microseconds to generate melody.Volume is controlled through *analogWrite*.