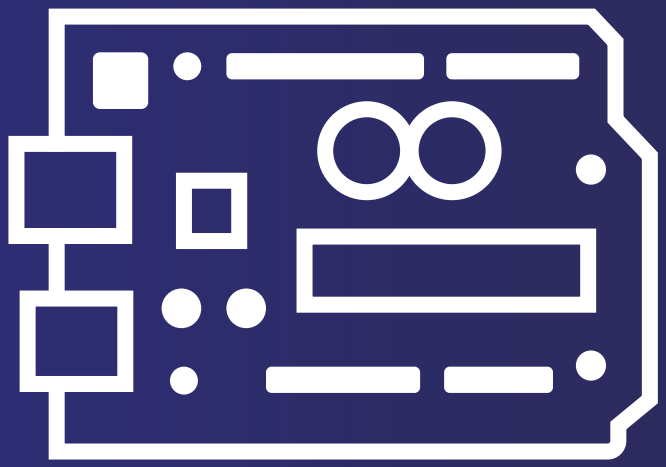
2021

Starter Kit

For Beginners



ORANGE KIT

Less hassle , more Fun !

01. Index

02. Index

Orange Beginner Kit

Hello Geek Thank you for purchasing our Orange Beginner Kit. We have specially designed this kit for those people who are interested in learning and Arduino.
After learning this kit, you will understand the following things:

- You will learn what is Arduino.
- You will learn about the active and passive components.
- You Will Understand the use of analog and digital read functions of the Arduino.

1. Arduino Introduction

In this section of this blog, we will talk about the Arduino Development Board.

Arduino is an open source project and was started with the intention of encouraging embedded systems students to learn about micro-controllers.

Before we begin, let me clear one very important doubt which every beginner faces in the initial stage of their learning phase and that is the difference between microcontroller and microprocessor.

Difference Between Microcontroller and Microprocessor

If you search this topic on the internet you will find a lot of information on this topic but as a beginner to all these things, this information will obviously confuse you.

But don't worry, I have explained that thing in the below section in such a way that it will clear all your doubts and after reading this you will not need to look back on this topic.

So, both microcontroller and microprocessor are things that are designed to control applications, perform logical and mathematical operations.

If both are designed to perform the same operation, what is the difference between these two things? This question may be on your mind.

The difference is, microcontrollers are designed to perform only predefined tasks whereas microprocessors are designed to perform tasks by considering the conditions that occur at the runtime of the process.

The other difference is that if you want to use a microprocessor then you have to interface the memory unit, EEPROM and everything that is essential to perform the tasks. Microcontrollers, on the other hand, don't need all those things because those things are built-in with them.

So, this was about the difference between microcontroller and microprocess.

In the next section we will learn more about Arduino.

What is Arduino?

As discussed earlier, Arduino is a microcontroller that can be used to build small-scale applications. Nowadays, we can see the use of Arduino extensively in the automation industry, there are also a lot of jobs available in the Indian market for Arduino Masters.

The reason for the massive popularity of Arduino is that Arduino is an open-source project and is designed by the community that constantly works on it.

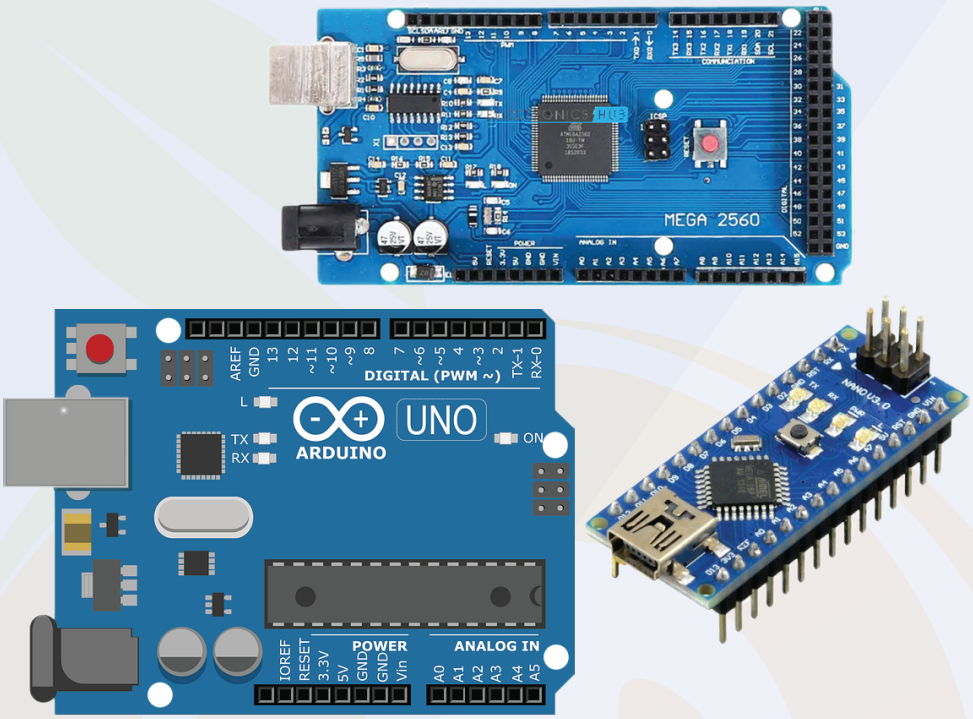
Since a large number of people contribute to this open-source project, Arduino users get quick solutions to their problems and their work never stops.

Varieties in board versions. This is another most important factor of Arduino boards.

The Arduino project was started in 2005. Aiming to provide a low-cost and easy way for novices and professionals.

1. Orange Arduino Uno Starter Kit
2. Orange Arduino Nano Starter Kit
3. Orange Arduino Mega Starter Kit

This was about the Arduino introduction in the next section of this blog we will learn to install the Arduino IDE on your system.



How to Install Arduino IDE On Windows?

Arduino IDE is a lightweight version software used to upload your written programs to Arduino boards.

This IDE gives us an easy-to-use UI that makes complex stuff so easy to understand.

In the following part, I have explained all these things step by step. Please take a look

To install Arduino IDE on your system, you need to download software package from internet.

How to Install Arduino IDE On Windows?

Arduino software from there or you can download it from their official website.

2. After the download is complete, extract that zip file. You can use WinRAR software to extract the file.

3. After extracting that zip file, you will see a folder. In that folder you will find an .exe file.

4. Right click on that file and select 'Run As Administer' option.

5. When you will click on that option, the system will start installing Arduino software.

6. Follow the instructions asked by the installer and do not make any changes to the settings.

7. Those options are for the user who installed Arduino on their system and they are updating the software or reinstalling the software on their system.

8. In your case, you are installing the software afresh, so you don't need to make those changes in your Arduino installation process.

9. While installing the software, the software installer will ask you to install the driver on your system. You have to allow the installer to install that driver.

10. Those drivers are of Arduino board and if you do not install them on your system then your system will not recognize Arduino board.

11. After this process is over, you will see the Arduino IDE icon on your system's applications list.

congratulation!! You have successfully installed Arduino IDE on your system.

There is one more driver that you need to install on your system. The name of that driver is CH340G driver and you will find the link to download that software on the product page

Please note - there is a misconception that boards that contain the ch340G driver are of poor quality but this is not true.

CH340G is just a driver used to transfer information from your computer to the controller board.

It is just a channel between the microcontroller and your computer. I have worked on those bords but never faced any problem.

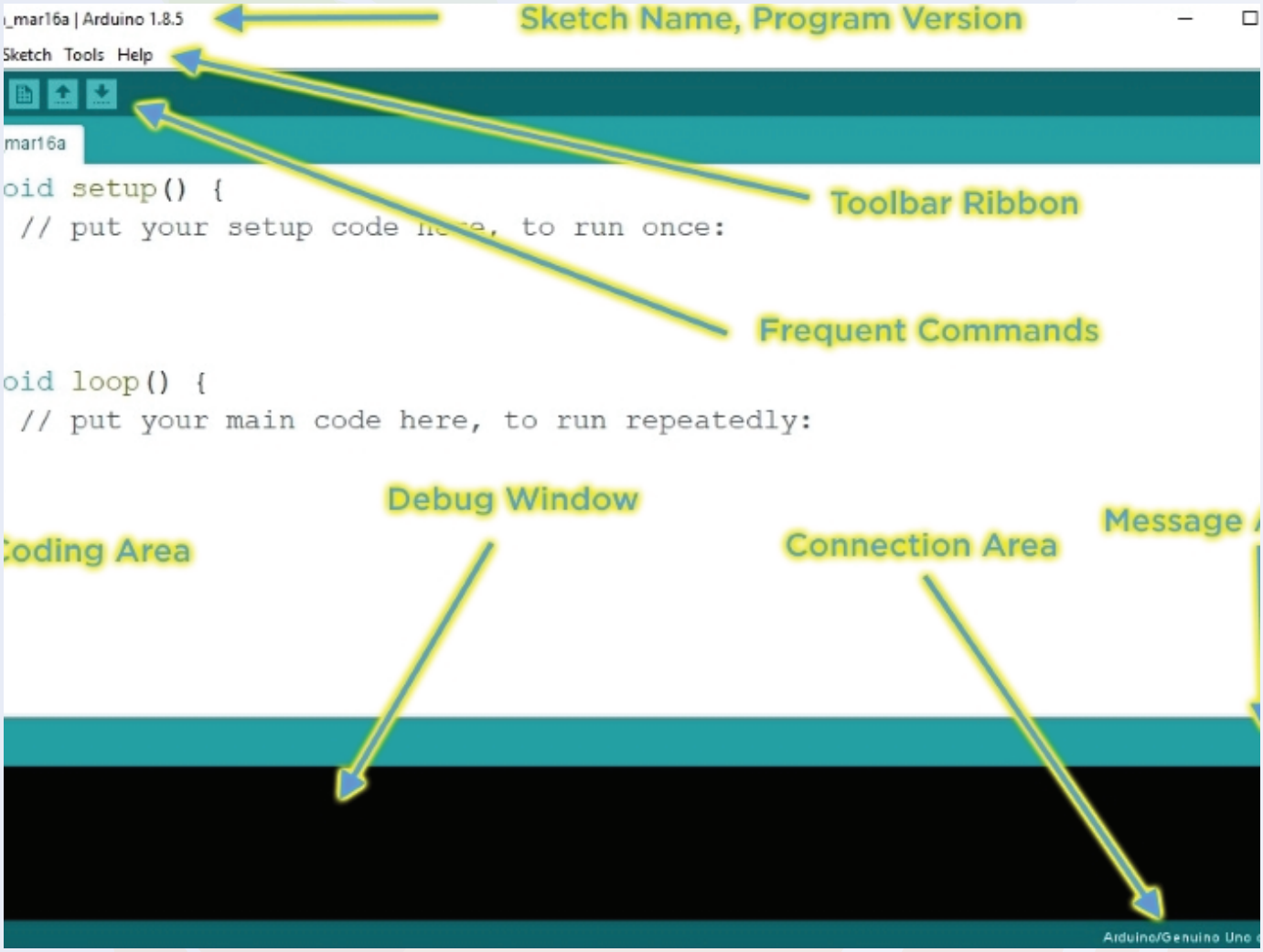
So stop worrying about this.

In the next part of this blog, we will learn about the UI of Arduino IDE and understand the usage of buttons available in the application.

2.0. Arduino IDE - User Interface

You have installed Arduino IDE on your system, now open it.

You will see a text editor as shown in the image below.



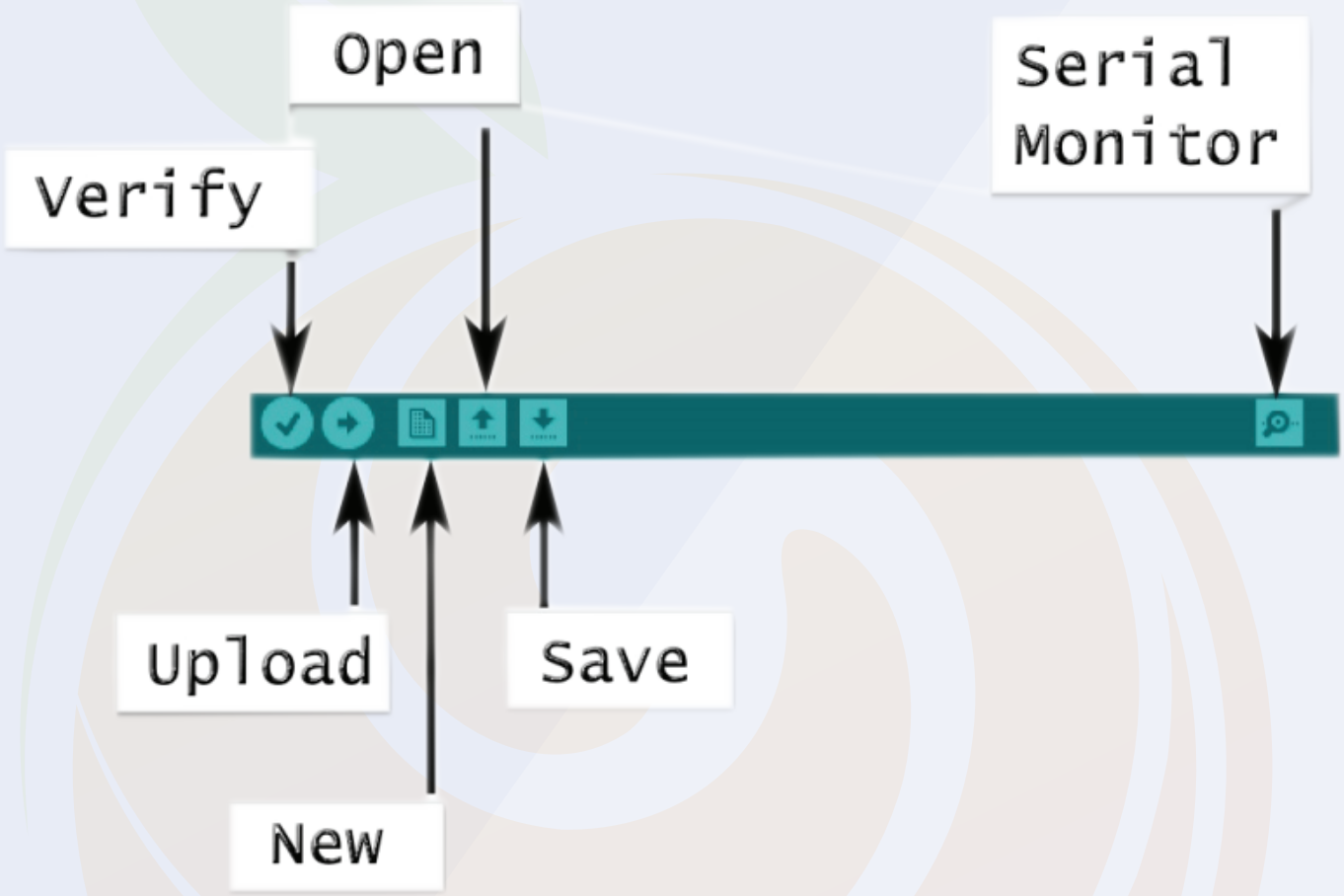
The Arduino IDE looks like this. In the above UI, you get all the functions that are useful for uploading code to your Arduino.

The IDE is divided into three sections. Text editor, output section and menu section.

You write your code in a text editor and you can upload your code to your Arduino with the help of the options available in the menu section.

2.1. Menu Option

Talking about the menu option, in that menu you get the following options.



The menu option looks like this. I have explained the working of those functions below please have a look.

2.1. Menu Opttion

Talking about the menu option, in that menu you get the following options.

1. File

- In this option you will get the option to save the file on the system and open the recent files if needed.

2. Edit

- Using this option, you can adjust the settings of your editor.

3. Sketch

- In this section you get the option to add libraries. To check what the library is all about, we request you to check the following section.

4. Tools

- This option is made for the selection of information related to the board. With the help of this option you can either add boards or install new boards in your IDE.

5.

- As the name suggests, you can use this section if you need any help regarding the software or program you have written.

5.Search

- This option is of serial monitor and with the help of this option you can open serial monitor.

If your code is using the serial print function, you will see the code's output

2.1. Menu Option

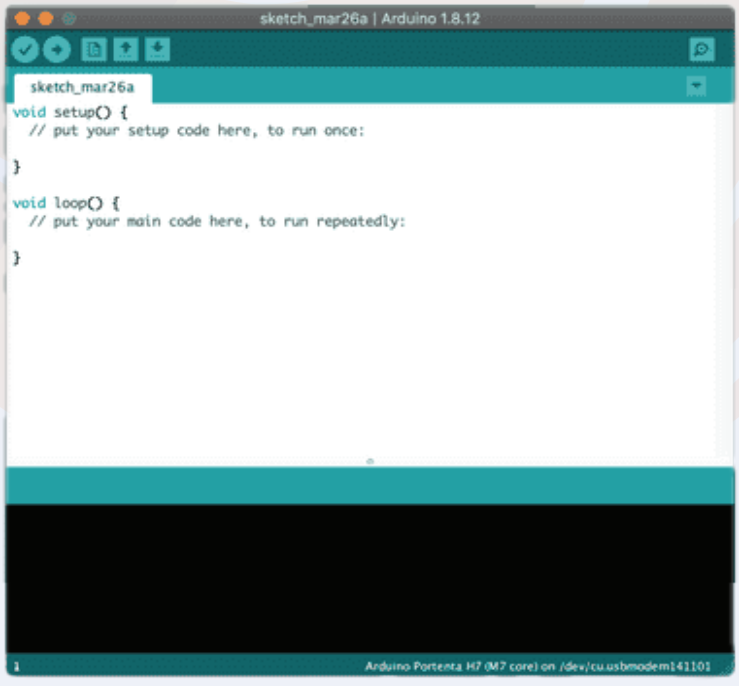
Output

In this option you will see the update of the uploading process of the code. If there is any problem with your code then IDE will generate some error and those errors we can see in this section.

In this option you will see the update of the uploading process of the code. If there is any problem with your code then IDE will generate some error and those errors we can see in this section.

Arduino Text Editor

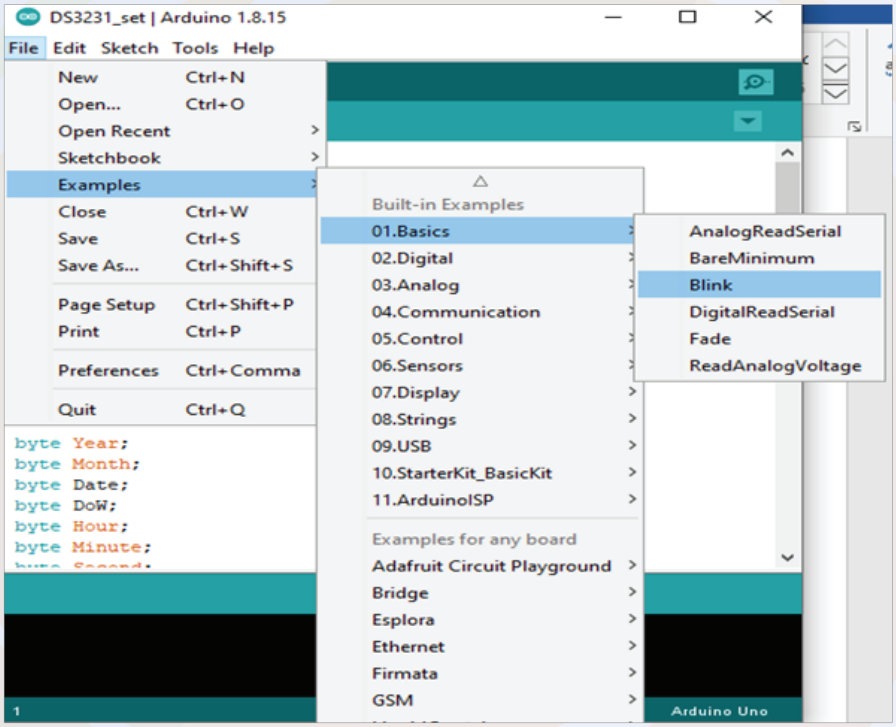
This section is designed for Arduino Code. In this section, we can write our Arduino.



2.2. How To Upload The Arduino Code To The Arduino Board?

In the above section we have seen different sections of Arduino IDE. In this section we will use those sections of the Arduino IDE to upload our first Arduino code to the Arduino board.

We will try to upload the blink code to the board. We will get the code from file section.
Please see the following image to understand the process.



When you will click on the blink option shown in the image, a new Arduino tab will open and in that tab, we will find the Arduino code, which we can use to toggle the 13-number pin.

So, now that we have the code, we are uploading this code to the Arduino and for uploading you can press the ctrl+U button on the keyboard to start the process of uploading the code.

So this is how we can upload the code to Arduino. In the next part of this blog, we will learn how to add libraries to that Arduino IDE?

3. What is an Arduino Library?

The Arduino library is a combination of code that has been written by an Arduino contributor.

The main purpose of designing libraries is to reduce code redundancy.

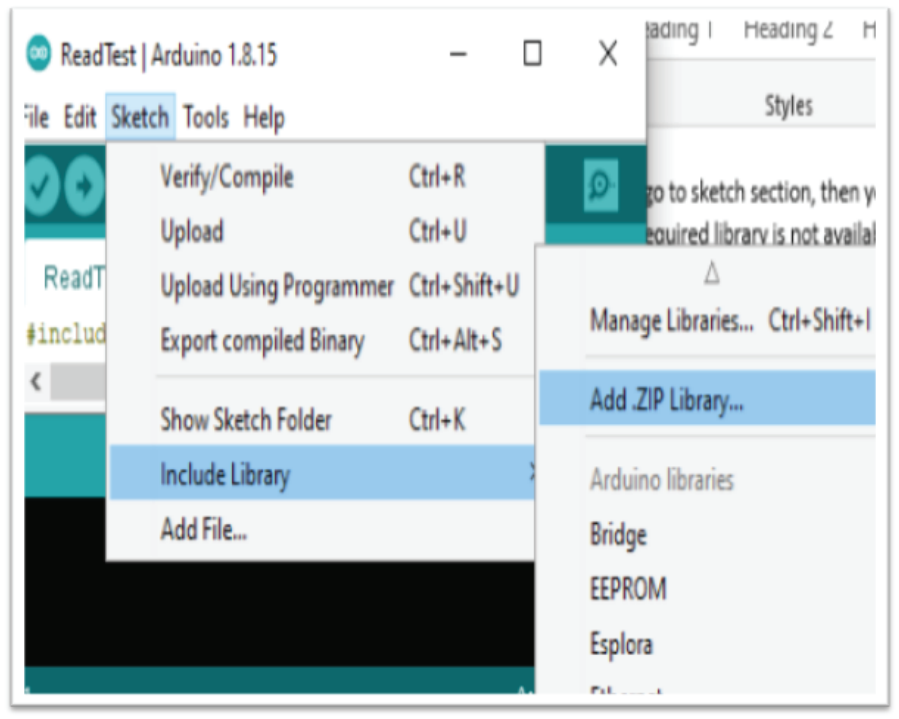
We use Arduino libraries, that means we use objects and methods of classes. So, even though we don't need to write much code, we can learn a lot by modifying existing code.

We can learn oops concepts, functional programming and much more.

So, that was about introduction to Arduino library, if you have any doubt you can contact me at info@robu.in.

How to Add an Arduino Library to the Arduino IDE?

There are two ways of adding library, either you can go to sketch section, then you can add the required library by clicking on the add library option or if the required library is not available on Arduino then we get 'Add Zip Library' option in that section.



3.1. What is an Arduino Library?

I have faced this issue many times, I never got my required library in the Arduino IDE. I always used to download the zip files of the library from the internet and then used to add that to the Arduino with the help of the 'Add zip Library' Option.

I hope the above section has cleared all your doubts about what is Arduino library and how to add a new Arduino library to the Arduino?

Arduino is an open source platform supported by huge community that continuously contributes to help others. So do not worry because the community will be always there to help

Void setup() is used for initializing the pins. This function executes once only.

If you are using a sensor that needs calibration before running, then you can do all those calibration settings in this section.

In Arduino, We use pinMode built in function to initialize the pins of the Arduino.

This function takes two parameters, pin number and the mode of operation. (Either output or input).

For Example: pinMode(12, INPUT);
pinMode(12, OUTPUT);

Please check the above examples. In the above examples, the first line is defining 12 number pin as an input pin and the second line is defining the 12 number pin as an output pin.

Let me tell you a simple logic here, if you are defining any pin as an output pin that means you are putting some voltage on that GPIO pin or sending out some data.

3.2. Void Loop And Void Setup

Voidloop()

This is one more inbuilt function that is used in the Arduino IDE. This loop keeps continuously running until someone don't turn Off the Arduino.

We can write our code-cases in this loop and those cases will keep on running.

Analog Write And Analog Read

Analog read and analog write these are the two functions that deals with the analog values.

AnalogWrite function is used to write the analog values whereas analogRead function is used to read the analog values that we pushing on the GPIO pin of the Arduino.

Ex, analogWrite(1, pwm)
    analogRead();

The analogwrite function take two parameters, pin number and pwm value.

The pwm value could be in range of 0 to 255.

Whereas the analogread function only take one parameter and that is pin number.

Store = analogRead(A1);

In the above example the analogRead function is reading the the data which is coming on the pin number A1 and then storing the information in the store variable.

So, this was about the analogRead function. This function is used when we are required to work with the variable voltages.

In next section of this blog we will learn about the digitalRead and digitalWrite function.

3.2. Void Loop And Void Setup

Digital Read And Digital Write

DigitalRead and digitalWrite functions are used for reading and writing digital values.

Talking about the digitalWrite function, this function takes two parameters, pin number and HIGH/LOW String.

When we give HIGH parameter to the function, the function will put HIGH level signal on the GPIO pin of the Arduino and when we put LOW then the function will produce LOW level signal on the GPIO pin.

For Example,

```
digitalWrite (12, HIGH);
digitalWrite (12, LOW);
```

In the above line of code, the first line of code producing HIGH level signal on the 12 number pin and in the second line code the function is producing LOW level signal on the 12 number pin.

\
Talking about the digitalRead function, this function takes one parameter and that is pin number.

For example,

```
Store = digitalRead(12);
```

In the above function, the code is running the data of the pin number 12 and storing the received input in the ‘store’ variable.

4. IR Remote

Nowadays you see IR remote everywhere. The device you use to control your TV, fan, is nothing but an IR remote. Using IR Remote you can control any device. But do you know how this IR remote works?
The answer is very simple. The IR remote consists of two units. One is the transmitter and the other is the receiver.

Talking about IR transmitter, it transmits the IR signal to the receiver. It has some buttons. The output of those buttons is connected to the 555 timer IC and the output of that timer IC is connected to the IR transmitter.
The entire unit is powered via a coin cell battery..

4.1. IR Remote

Note - You must remove the plastic that is under the coin cell before using the remote. If you don't remove the plastic, the remote won't work.

This was about the IR transmitter, now we will talk about the second section of the unit and that is the IR receiver. It is nothing but a photoreceiver. When you press any button on the remote, the special button is connected to the timer IC and then the IC turns on the IR LED in a specific pattern.

That pattern is then received by the IR receiver where the IR receiver decodes the received signal and passes it to the control unit.

4.2. Working Principle Of IR Remote

Till this point, we discussed what is IR sensor and how it can be used but do you know the working principle of IR sensor?

IR sensor works on the principle of three laws. Those laws are as follows.

1. Planck's Radiation law
2. Stephen – Boltzmann law
3. Wien's Displacement law.

Planck’s Radiation law:

According to Planck's law of radiation, every object radiates some energy, even black bodies.

The black body absorbs the surrounding energy and sends that energy back to the air around it.

Stephen – Boltzmann law:

According to the Stefan–Boltzmann law, the energy released from a surface is proportional to the fourth power of its absolute temperature.

The first ovens were infrared! These were made of an infrared-emitting burner that heated the food with its molecules.

Wien’s Displacement law

This is the third law on which the working principle of the IR sensor depends. Wien's law describes the relationship between the temperature of a black body and the wavelength at which it emits light.

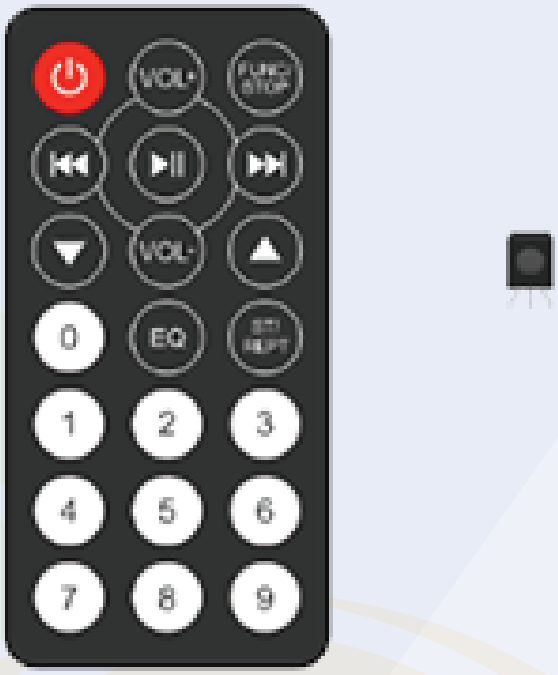
So, these above laws are used to detect the obstacle and find the distance of the obstacle.

4.3. IR Emitter

You can see a black LED on the front of the IR remote, which is the IR emitter.

IR emitter or transmitter is designed to generate IR signal. They are nothing but an IR LED.
The following image shows the image of IR LED.
The construction of IR LED is similar to that of normal LED.

4.4. IR Remote



Even though the construction of an IR LED looks similar to the construction of a normal LED, the light emitted by an IR LED is not visible to the human eye.

.

This is because the wavelength of the rays emitted by IR LEDs will be in the range of 700nm to 1mm which is far from the visible spectrum.

Humans can only see things that are in the wavelength range of 400nm to 750nm (visible spectrum).

The other difference between normal LEDs and IR LEDs is that the switching speed of those IR LEDs is much faster than that of normal LEDs. This is the only feature that makes IR LED stand out from the crowd.

4.4. IR Remote

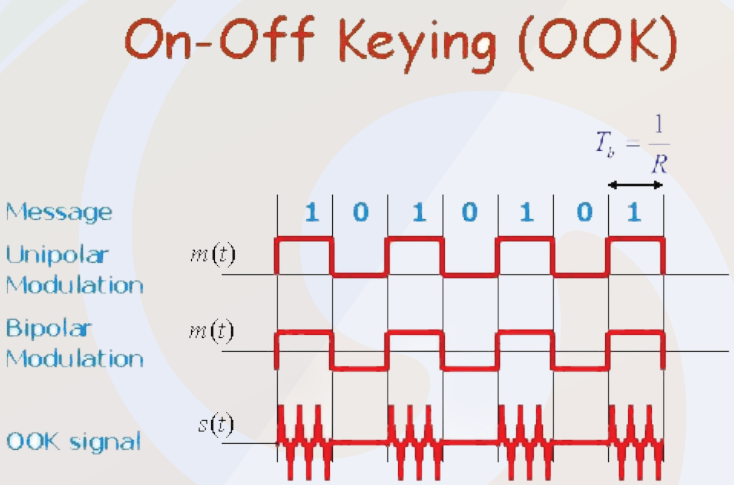
Based on this feature, the IR LEDs can be used for application such as data transfer, remote control, object detection etc.

The IR transmitter uses On-Off-Keying modulation to transmit the signals.

On-of-keying modulation is similar to amplitude shift keying.

In On-Off-Keying modulation, the carrier signals are turned on for the symbol one and turned off for the symbol one.

The output waveform of the OOK modulation is shown below.



T_b – bit period ; R – bit rate

$$m(t) = \begin{cases} 1 & \text{symbol 1} \\ 0 & \text{symbol 0.} \end{cases}$$

So, this was about the IR LED, in the next section of this blog, we will discuss about the IR Receiver.

4.5. IR Receiver

The IR receivers are nothing but optocouplers. These optocouplers are little bit different than the normal optocouplers.

They are designed to receiver only the IR rays. They do not respond to other Light.

When the IR light falls on the IR receiver, the photons are absorbed by the PN junction of the IR receiver and the voltage is generated based on the number of photons are absorbed.

This is how the IR receiver receives the signals that are transmitted by IR receiver.

The output of the IR receiver is then passed to the control unit, where the control unit takes the required action based on the inputs of the decision-making system.

At this stage, the signal gets decoded and transferred to the main control unit for the further communication.

So, this is was all about the working principle of the IR Sensor.

4.6. IR remote Interfacing With Arduino

In this section, we will understand how to interface IR receiver with Arduino. What you are getting with this module has three pins to connect to any microcontroller or any microprocessor. I have explained the function of those pins below.

GND - Here we have to connect the GND pin of the Arduino to this pin.

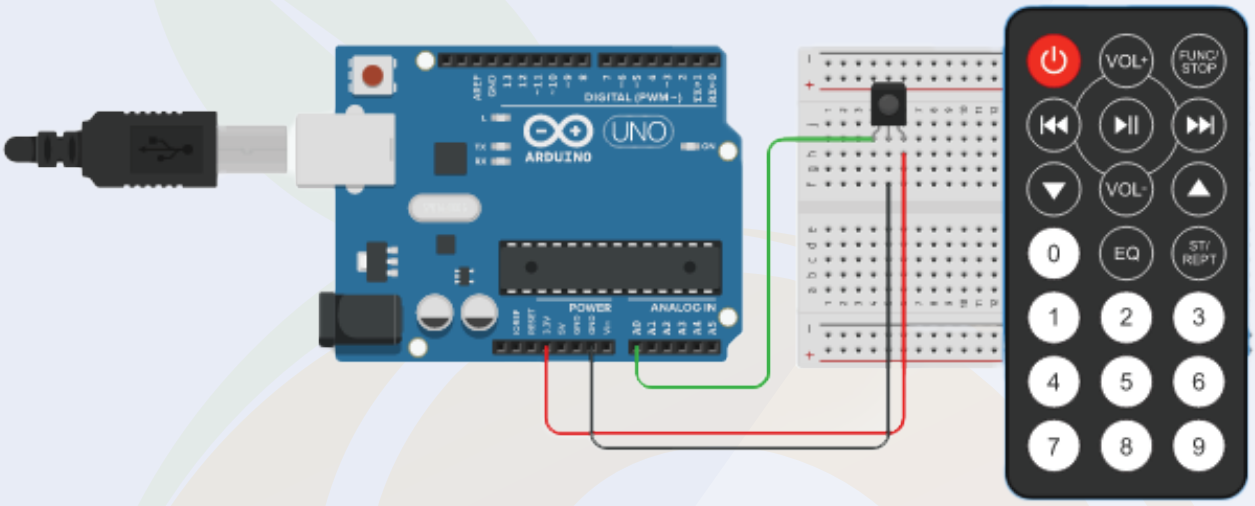
Vcc - Connect the 5V pin of the th8e Arduino to this pin.

Signal - You will find the module's output on that pin. You can connect this pin of the module to any GPIO pin of the Arduino.

4.8. IR remote Interfacing With Arduino

In my case, I have connected this pin to the 5-digit pin of the Arduino.

You can also refer to the following image to understand the interfacing properly.



4.7. Arduino Code For IR Remote

Ok, now you can use the following code to start working with the IR module.

The following code will turn on the IR module and print whatever the IR module receives on the serial monitor.

Once you start receiving the output of the IR sensor, you can use that receive code and assign that code to a specific function of code to trigger a particular function when that particular function is triggered. code is received.

I've tried to cover everything you need to know about the code you can use to test IR modules, but if you have any doubts you can ask me in the comments section.

4.8. IR remote Interfacing With Arduino

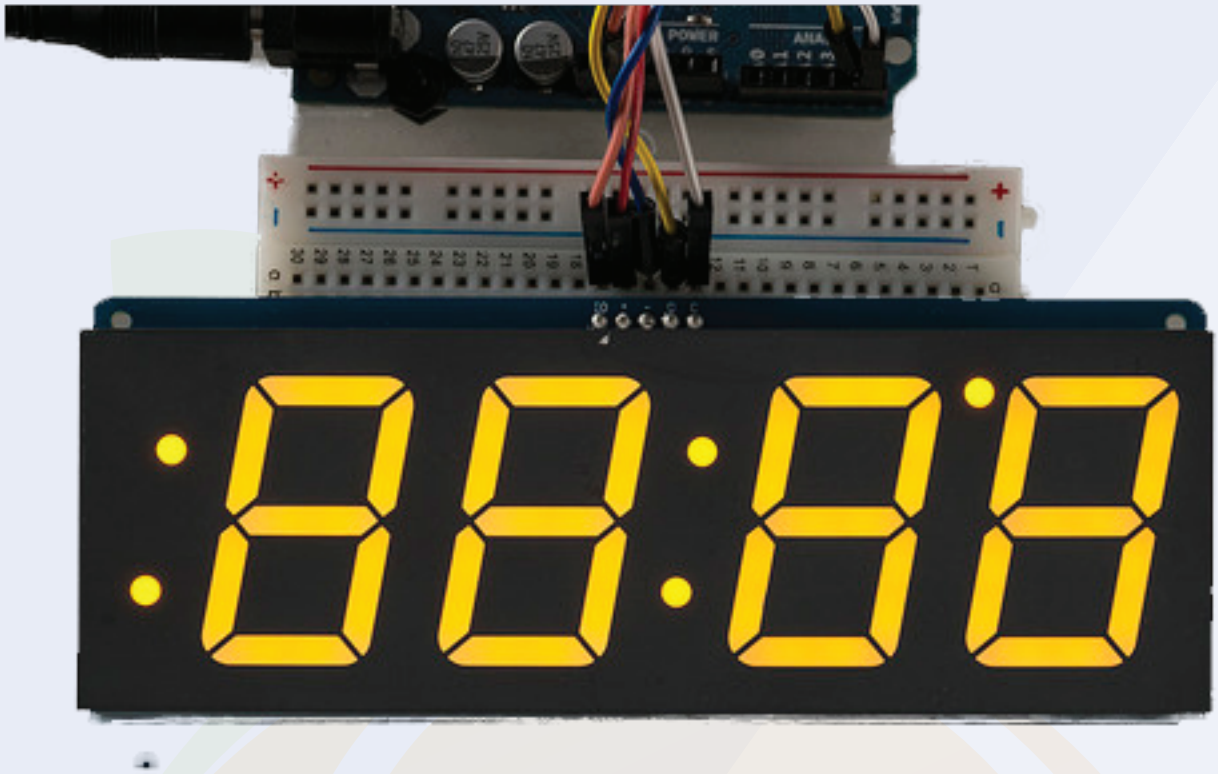
```
#include <IRremote.h>

const int RECV_PIN = 7;
IRrecv irrecv(RECV_PIN);
decode_results results;

void setup(){
  Serial.begin(9600);
  irrecv.enableIRIn();
  irrecv.blink13(true);
}

void loop(){
  if (irrecv.decode(&results)){
        Serial.println(results.value, HEX);
        irrecv.resume();
  }
}
```

5. Seven Segment Display

A 7-segment display is nothing but a pack of seven LEDs linked together where each LED is known as a segment. All of them can be controlled individually.

7-segment displays are available in a variety of colors (red, blue and green) and sizes (0.56 to 6.5 inches). Sometimes two to four 7-segment displays are packed together to make one larger display (see the following image).

5. Seven Segement Display

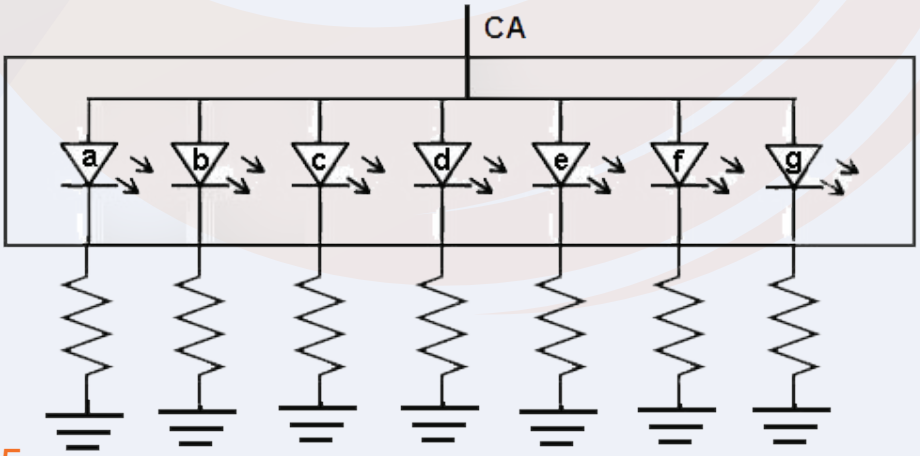


5.1. 7-Segment Display Types

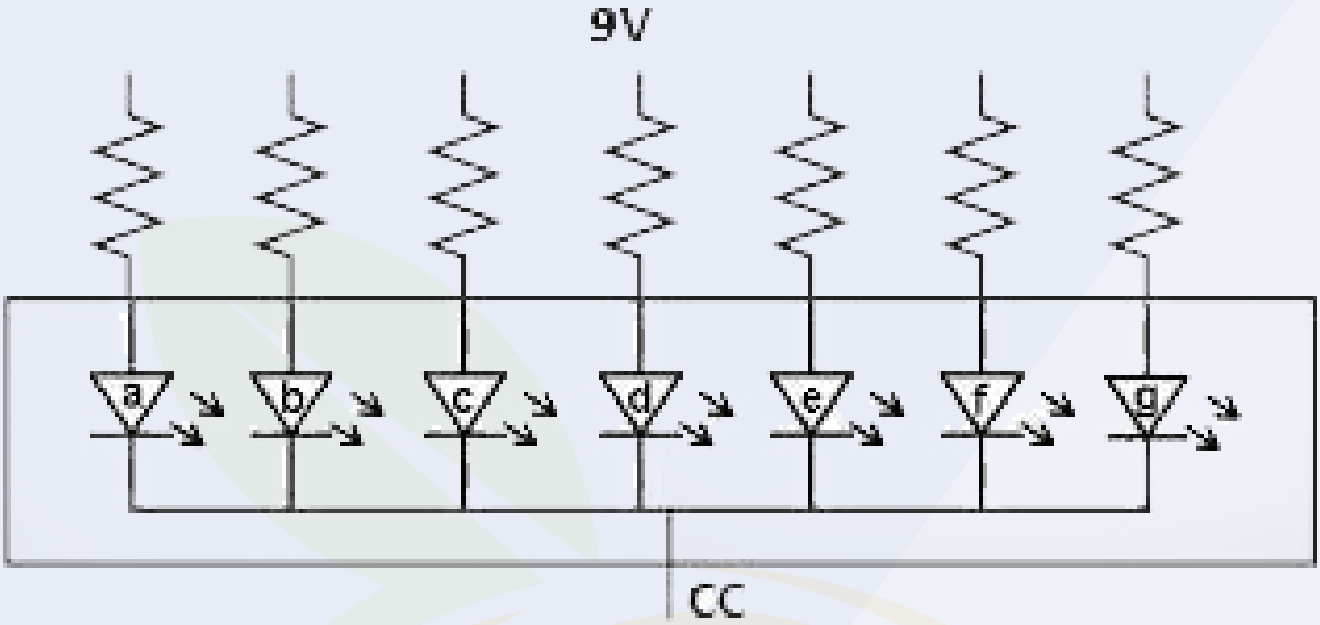
7-segment displays are divided into two types based on the LED anode and cathode connections.
We will briefly discuss these two types and their configurations.

A) Common Cathode (CC)

In this configuration, all the anode terminals are connected, and the cathode terminals are kept open.
To turn on the LED display, connect the anode to the 5V supply and the cathode to GND.



B) Common Anaode



In this configuration, all the anode terminals are connected, and the cathode terminals are kept open.
To turn on the LED display, connect the anode to the 5V supply and the cathode to GND.

5.2. Seven Segment Display Interfacing With Arduino

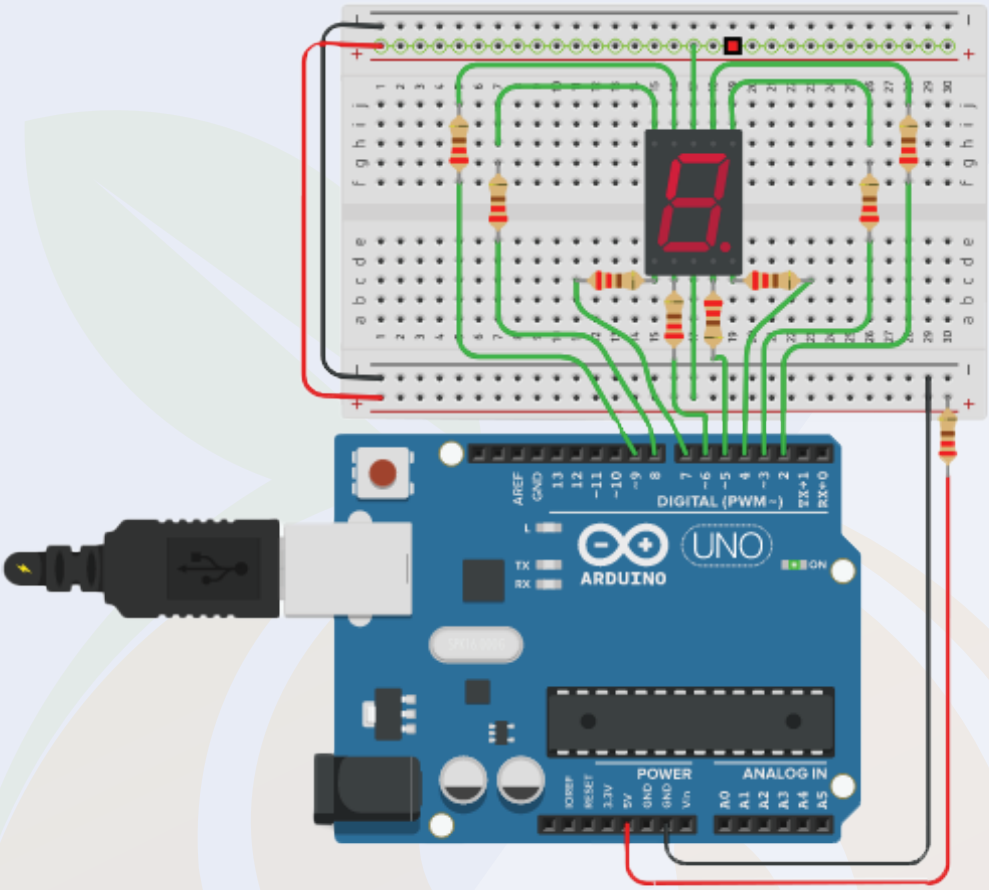
Each LED in the 7-segment display is separately connected to a GPIO pin on the Arduino board.

For interfacing purpose let us consider a common anode (CA) 7-segment display.

Since the anode is the common terminal here, let us connect it to the 5V supply on the Arduino. The remaining pins will be connected to the GPIO pins on the Arduino.

We will be using separate wiring (see connection diagram) for each LED segment and power up the display in an ornate manner.

5.2. Seven Segment Display Interfacing With Arduino



5.3. Arduino Code For Seven Segment Display

The code for a seven-segment display is very simple and you don't need to install any libraries to work with a seven-segment display.

You can use the following code to work with the seven-segment display.

One more thing you can refer to the following truth table to understand the logic of Arduino code.

If you have any doubt you can ask me in the comment section.

5.3. Arduino Code For Seven Segment Display

```
void setup()
{
  // define pin modes

 pinMode(2,OUTPUT);
 pinMode(3,OUTPUT);
 pinMode(4,OUTPUT);
 pinMode(5,OUTPUT);
 pinMode(6,OUTPUT);
 pinMode(7,OUTPUT);
 pinMode(8,OUTPUT);

}

void loop()
{
  // loop to turn leds od seven seg ON

  for(int i=2;i<9;i++)
  {
    digitalWrite(i,HIGH);
    delay(600);
  }

  // loop to turn leds od seven seg OFF
  for(int i=2;i<9;i++)
  {
    digitalWrite(i,LOW);
    delay(600);
  }


  delay(1000);

}
```

6.0. Potentiometer

You must have seen the variable knob on the audio system. We use that knob to adjust the level of sound.

In electronic terms, that knob is called a potentiometer and is nothing but a variable resistor. The output of the potentiometer varies according to the change in resistance.

It consists of three pins. I have mentioned the functionality of those pins below.

GND - Here you can connect the GND pin of the power source

Vcc - The positive pin of the power supply will be connected to this pin.

Signal - On this pin you will get the output of the potentiometer.

This was about the introduction of potentiometers. In the next part of this blog, we will learn how to interface a potentiometer with Arduino.

6.1. Working Principle Of the Potentiometer

As discussed earlier, potentiometer has three parts, one movable and two fixed. The movable part is known as wiper. That wiper slides on the wire winded resister.

When we rotate the pot, the position of the wiper changes and so the resistance of the potentiometer also changes.

So, this is how the potentiometer works.

There are different types of the potentiometers are available in the market. I have shared the names of those potentiometers below.

6.2. Different Types of the potentiometer

- **Logarithmic Potentiometer**
- **Linear Potentiometer**
- **Rheostat**

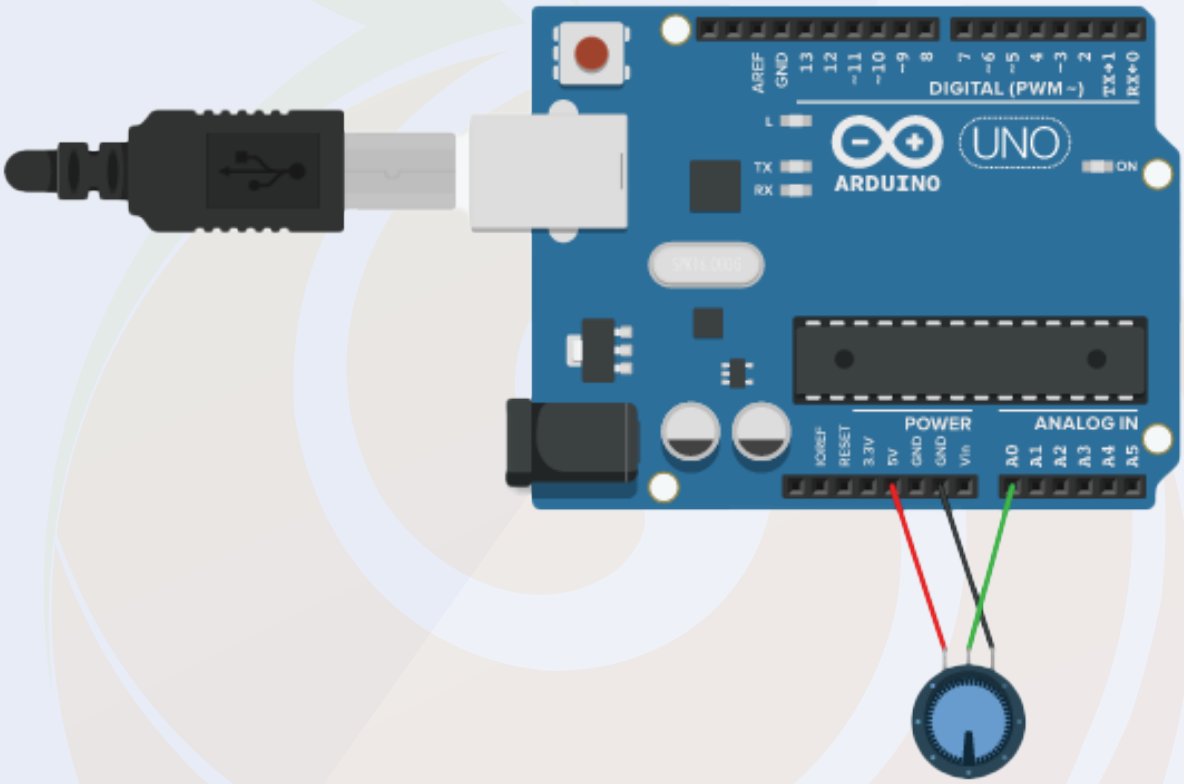
6.3. Interfacing Potentiometer with Arduino

Interfacing of the potentiometer with the Arduino is so easy. Here you have to connect the GND pin of the Arduino to the GND pin of the potentiometer and the VCC pin of the Arduino to the Vcc pin of the potentiometer.

Talking about the output pin of the potentiometer, you can connect to any GPIO pin of the Arduino.

In my case, I have connected the output pin of the potentiometer to the 8 number pin of the Arduino.

Please refer to the following image to understand the connection diagram.



6.4. Arduino Code For Potentiometer

Since the interfacing diagram of the potentiometer is easy, the Arduino code for this is also easy.

You can use the code below to work with a potentiometer.
You will see the output of the code on the serial monitor.

6.4. Arduino Code For Potentiometer

```
/*
by: di2tnugraha
https://www.youtube.com/kenziechannel
 */
int led = 11;
int pinPot = A0;
int potVal = 0;

void setup() {
  Serial.begin(9600);
  pinMode(led, OUTPUT);

}

void loop() {
   potVal = analogRead(pinPot);
   potVal = map(potVal, 0, 1023, 0, 255);
   analogWrite(led, potVal);
   Serial.println(potVal);
   delay(200);
}
```

So, in this way, you learned how to interface potentiometer to Arduino. In the next part of this blog you will learn about flame sensor.

7. IR Flame Sensor

Different flame sensors are available in the market. What you are getting with this kit is the IR flame sensor.

IR flame sensors are used in fire systems. They detect the fire and send a signal to the controller unit, then the controller unit takes the necessary actions to avoid damage from the fire. But do you know how the IR flame sensor detects fire.

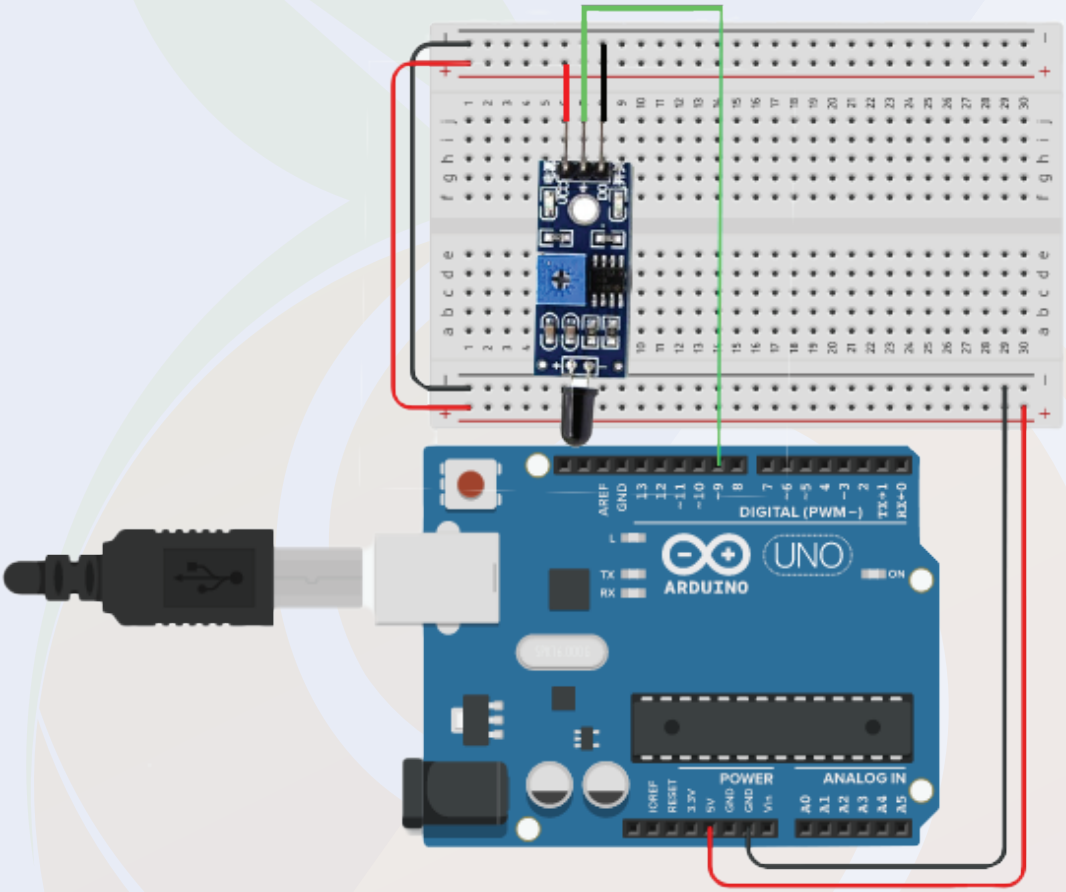
The answer to the question is, IR sensors are designed to capture gases that appear in the infrared spectral band.

7.1. Interfacing Flame Sensor With Arduino

As we know that IR Flame Sensor has three pins, interfacing of IR Flame Sensor with Arduino is so easy.

You just need to connect the signal pin of the flame sensor to the analog pin of the Arduino and the other power pin of the sensor to the power pin of the Arduino.

Please refer to the image below to understand the wiring diagram.



7.2. Arduino Code For Flame Sensor

So, as you connected the IR Flame Sensor to the Arduino, we will now move to the Arduino Code section of the Flame Sensor.

The code is very simple and requires no explanation. In this code, we have used a variable and the function of that variable is to collect the output of the IR flame sensor.

7.2. Arduino Code For Flame Sensor

```
// lowest and highest sensor readings:
const int sensorMin = 0;     // sensor minimum
const int sensorMax = 1024;  // sensor maximum

void setup() {
  // initialize serial communication @ 9600 baud:
  Serial.begin(9600);
}
void loop() {
  // read the sensor on analog 9:
          int sensorReading = analogRead(9);
  // map the sensor range (four options):
  // ex: 'long int map(long int, long int, long int, long int, long int)'
          int range = map(sensorReading, sensorMin, sensorMax, 0, 3);

  // range value:
  switch (range) {
  case 0:    // A fire closer than 1.5 feet away.
    Serial.println("** Close Fire **");
    break;
  case 1:    // A fire between 1-3 feet away.
    Serial.println("** Distant Fire **");
    break;
  case 2:    // No fire detected.
    Serial.println("No Fire");
    break;
  }
  delay(1);  // delay between reads
}
```

8. Buzzer

Buzzers are used in many instruments. It is used as an indicator in many systems such as home appliances, alarm systems, electronic bells.

Talking about the working principle of the buzzer, the buzzer converts electrical energy into sound energy.

When voltage is applied to the buzzer, the piezo crystal expands and contracts inside the plastic casing. This causes the plate to vibrate near the crystal and the sound you hear is of that vibration.

Changing the frequency to the buzzer changes the speed of the vibration and as a result you hear a variety of sounds.

So, it was about the buzzer. There are two main types of buzzers. Active buzzer and passive buzzer. In the next section of this blog, we will understand the difference between these two types.

8.1. Buzzer Classification Based on the construction of the Buzzer –

Buzzers are classified in to two categories based on the type of the construction.

1. Piezo Electric Buzzer
2. Electromagnetic Buzzer

Piezo Electric Buzzer

In piezoelectric electric buzzer, when we apply external force on the piezo electric buzzer, the piezo electric material will deform and some amount of current will be produced according to the applied external force.

In the same way, when we apply charge to the piezo electric buzzer, the piezo electric material will deform.

If we give continuous PWM cycles to the piezo electric buzzer, the buzzer will produce a sound that is interpretable to the ears.

8.1. Buzzer Classification Based on the construction of the Buzzer –

Electromagnetic Buzzer

As the name implies, these buzzers make use of the electromagnetic effect, the diaphragm vibrates when we power the electromagnetic effect.

And finally, the vibration turns into human audible sound.

So, this is how electromagnetic buzzer works.

8.2. Difference Between Active and Passive Buzzer

As I mentioned earlier, there are many types of buzzers. Active buzzer and passive buzzer.

Talking about the active buzzer, it has an inbuilt oscillating source. Active buzzer starts ringing as soon as you turn it on but in case of passive buzzer they do not have inbuilt oscillating source.

If you want a passive buzzer to produce a sound signal, you must give a different frequency to the buzzer.

8.3. Buzzer Interfacing with Arduino

No matter which buzzer you are using, it only has two pins. You can connect the vcc pin of the buzzer directly to the 9 GPIO pin of the Arduino and the GND pin of the buzzer to the GND pin of the Arduino.

In my case, I have used an active buzzer. But you can use passive buzzer, but remember, if you want to make different sound from passive buzzer here, you have to give different frequency to VCC pin of buzzer.

Please check the interfacing diagram given below to understand the interfacing diagram.

As you have wired the buzzer. Now, we will move on to the Arduino code for the buzzer.

8.4. Arduino Code For Buzzer

The code for the buzzer is very simple. To turn the buzzer on or off you just need to use the digitalWrite function and apply a high or low voltage.
You can use the following code to turn on the buzzer.

```
const int buzzer = 9; //buzzer to arduino pin 9


void setup(){

  pinMode(buzzer, OUTPUT); // Set buzzer - pin 9 as an output

}

void loop(){

  tone(buzzer, 1000); // Send 1KHz sound signal...
  delay(1000);        // ...for 1 sec
  noTone(buzzer);     // Stop sound...
  delay(1000);        // ...for 1sec

}
```

09. Vibration Sensor

Vibration sensors are used in many devices such as mechanical machines. It is used to measure the frequency of vibrations generated by machines and those measured frequencies are then converted into voltage signals.

The vibration sensor we get with this kit has an onboard gain amplifier. That amplifier is used to adjust the gain of the module. If the signals generated by the vibration sensor are not accurate you can adjust the trimmer next to the gain amplifier.

9.1. Working Principle of the Vibration Sensor

Vibration sensors are used in many electronics applications. There are different types of vibration sensors available in the market.
To get optimum performance of the machine, we need to continuously monitor the parameters of the machine such as vibration, speed and temperature.

The vibration sensor has a strain gauge. The resistance of that strain gauge changes according to the change in velocity of vibration.

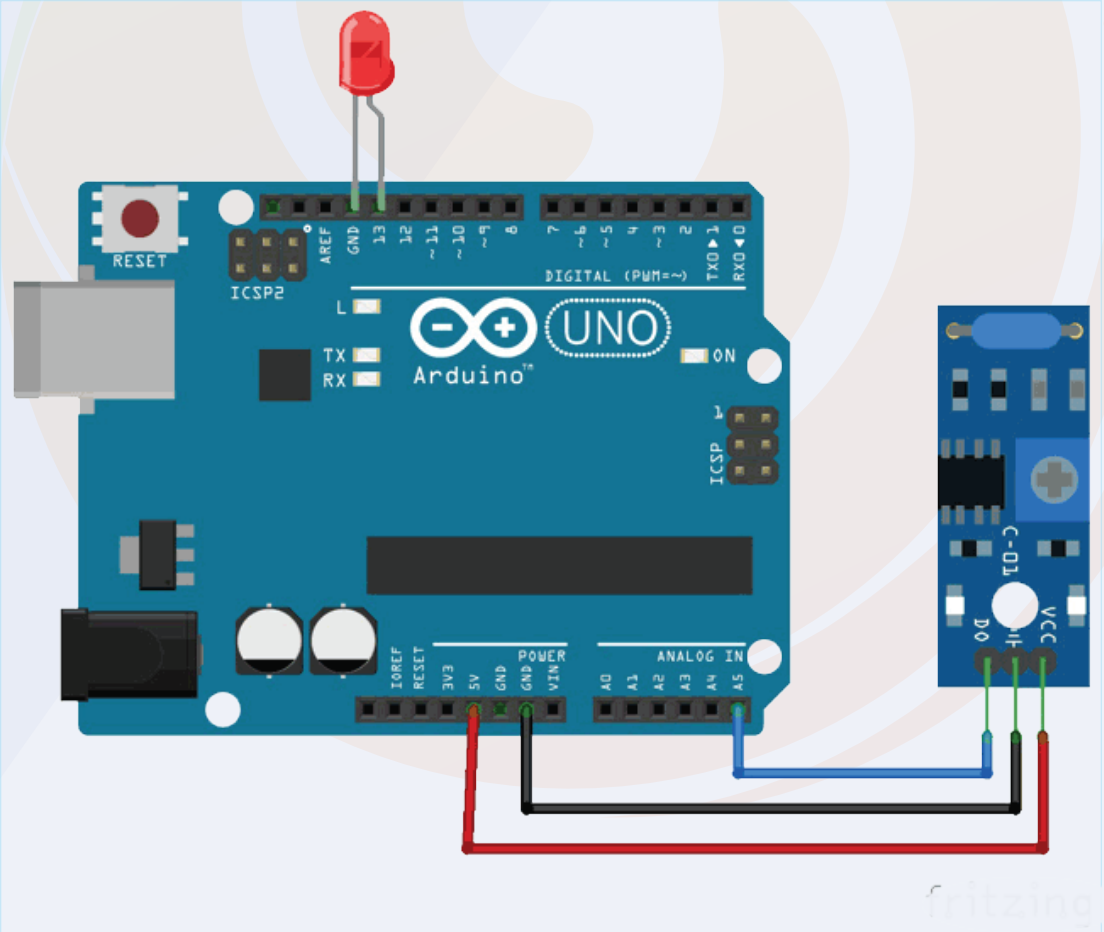
A change in resistance will also change the output of the sensor.
So, the vibration sensor works like this.

9.2. Interfacing Vibration Sensor With Arduino

Interfacing the vibration sensor with Arduino is super easy. It has three pins, two for powering the module and one for output.

We need to connect the power pin of the vibration sensor to the Arduino and the signal pin to any GPIO pin of the Arduino.

You can refer to the following image to understand the wiring diagram.



9.3. Arduino Code For Vibration Sensor

I have shared the code for the vibration sensor below. In this code we are using analog read function and a global variable to read the output of the sensor.

In the below code we are using serial print function to print the output of sensor. You can open Serial Monitor and see the output there.

```
const int ledPin=A5;
void setup() {
   Serial.begin(9600);
   pinMode(ledPin,OUTPUT);
}

void loop() {
   int sensorState = digitalRead(2);
   Serial.println(sensorState);
   delay(100);
   if(sensorState == HIGH)
   {
      digitalWrite(ledPin,HIGH);
   }
   else
   {
      digitalWrite(ledPin,LOW);
   }
}
```

10. 74HC595N 8 bit shift Register

Have you ever felt that your Arduino has fewer pins than necessary? When I work with big projects I face these type of issues more often. In those cases, we get only two options, either we use another Arduino and write part of the code for no reason or use a shift register.

Shift register not only saves our time but it also saves that bad connection which we make while working on a big project.

In this kit we are using 74HC595N shift register which has three input pins and eight output pins.

10.1 Working of 74HC595N 8-bit shift Register

Shift registers are used in many applications such as calculators, serial to parallel data converters. They store two-bit binary data before the data is processed with any other data.

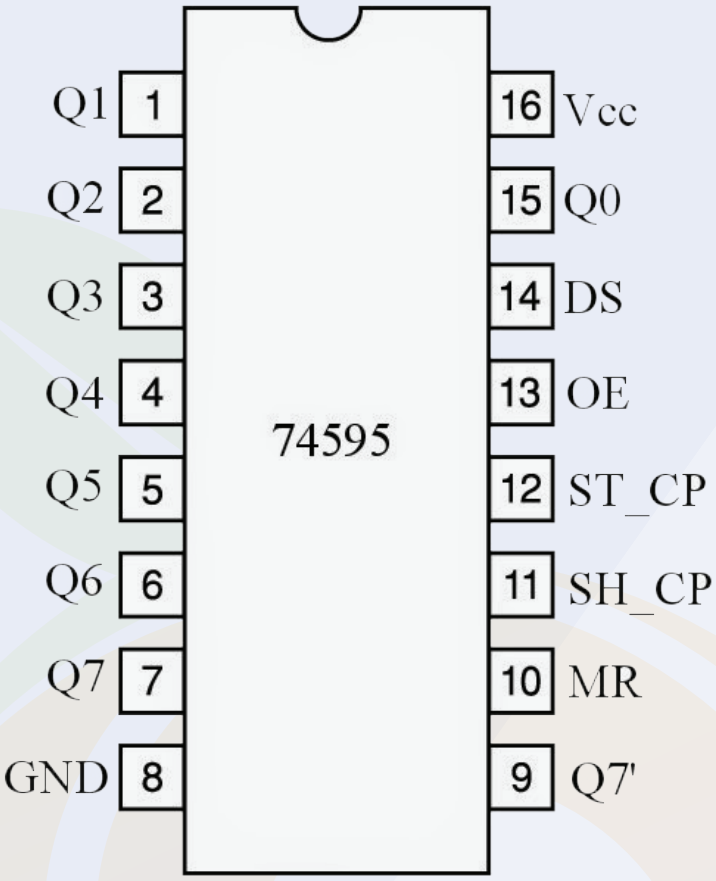
So, this was the introduction to the shift-register but do you know how the shift register works? In the next section, we will learn about all these things.

The shift register uses D-type data latches, these D-type data latches are connected in a cascade format. The output of one D-type data latch is connected to the input of another data latch.

The data we would pass to the shift register would be passed sequentially to each of those D-type data latches.

This sequence will be either right to left or left to write.
This is how the data is processed by the shift register.
Now, we will understand the interfacing of 74HC595N with Arduino.

10.2. Interfacing 74HC595N With Arduino



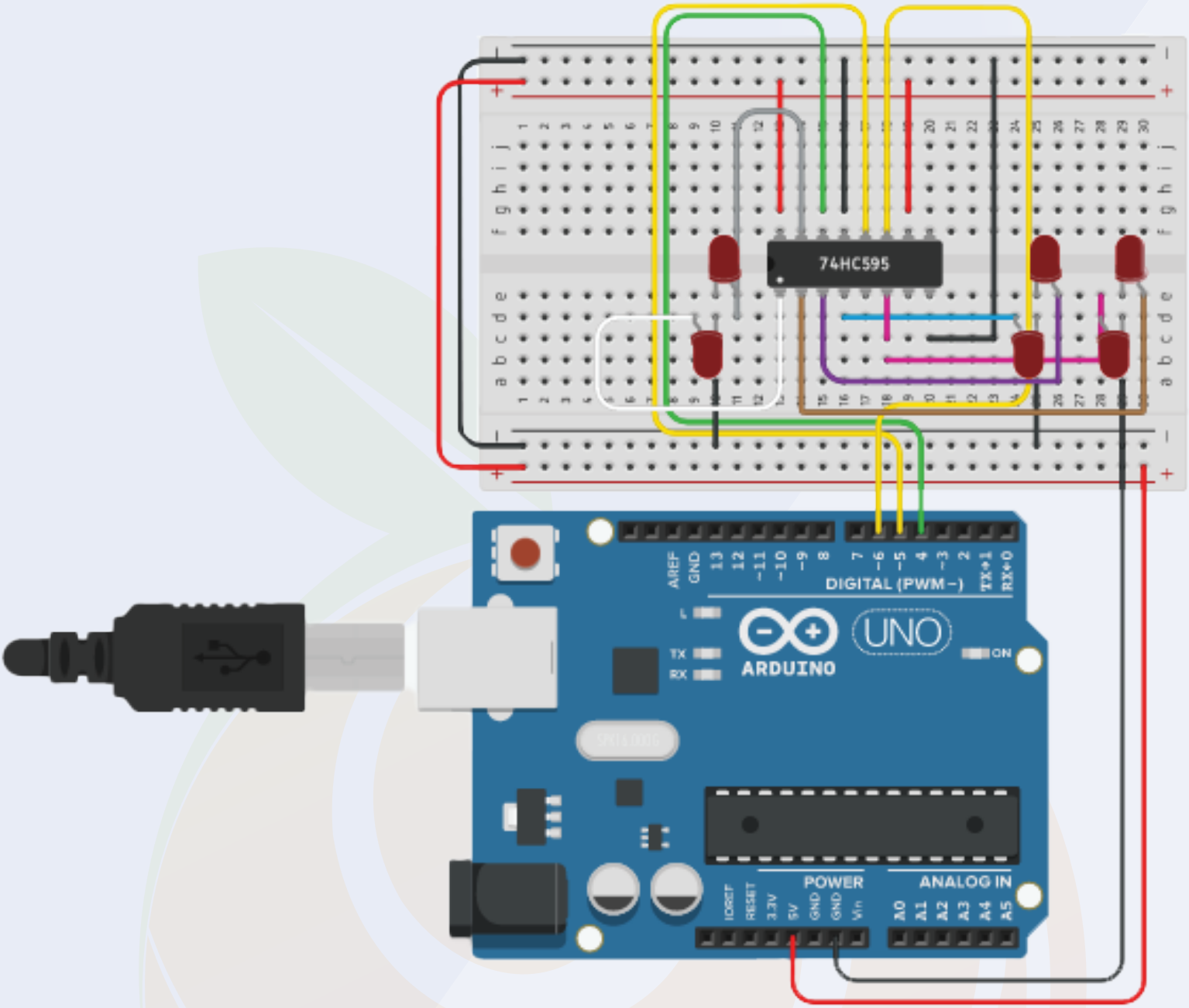
So, before we talk about interfacing the multiplexer with Arduino, we will understand the pinout of the IC.

This IC has 16 pins, out of which we will use only 11 pins. Pin numbers are 1,2,3,4,5,6,7, and 15 are output pins and 14, 12 and 11 are data pins.

Now we have to enable the IC to work with this IC. And to enable IC we have to connect GND pin of Arduino to IC's 8, 10 and 13.

The reason I connect the GND pin to those pins is because those pins are active low pins so we need to ground the Arduino to enable those pins.

This is about the introduction part of IC. In the next part of this blog we will understand how Arduino is coded?



10.3 Arduino Code For 74HC595N

I have designed the following Arduino code for 74C595N. This code is generating the output on the output pin of the 74C595N.

10.3. Arduino Code For 74HC595N

```
int latchPin = 5;          // Latch pin of 74HC595 is connected to Digital pin 5
int clockPin = 6;          // Clock pin of 74HC595 is connected to Digital pin 6
int dataPin = 4;// Data pin of 74HC595 is connected to Digital pin 4

byte leds = 0;             // Variable to hold the pattern of which LEDs are currently turned on or off

/*
 * setup() - this function runs once when you turn your Arduino on
 * We initialize the serial connection with the computer
 */
void setup()
{
  // Set all the pins of 74HC595 as OUTPUT
  pinMode(latchPin, OUTPUT);
  pinMode(dataPin, OUTPUT);
  pinMode(clockPin, OUTPUT);
}

/*
 * loop() - this function runs over and over again
 */
void loop()
{
  leds = 0;         // Initially turns all the LEDs off, by giving the variable 'leds' the value 0
  updateShiftRegister();
  delay(500);
  for (int i = 0; i < 8; i++) // Turn all the LEDs ON one by one.
  {
    bitSet(leds, i);                    // Set the bit that controls that LED in the variable 'leds'
    updateShiftRegister();
    delay(500);
  }
}

/*
 * updateShiftRegister() - This function sets the latchPin to low, then calls the Arduino function 'shiftOut' to shift
out contents of variable 'leds' in the shift register before putting the 'latchPin' high again.
 */
void updateShiftRegister()
{
   digitalWrite(latchPin, LOW);
   shiftOut(dataPin, clockPin, LSBFIRST, leds);
   digitalWrite(latchPin, HIGH);
}
```

So, it was about the components that we are getting in this kit. In the next section we'll be working on some projects using the components of this kit.

Arduino Project

P1. Fire Detection Using Arduino

You must have seen fire alarm systems in many places. Such systems are designed to avoid casualties in an explosion.

To complete this project, we are going to use the following components.

List of The Components

1. Arduino
2. Jumper Cable
3. Flame Sensor
4. Active Buzzer
5. Bread baord

Interfacing Diagram For Fire Alarm System

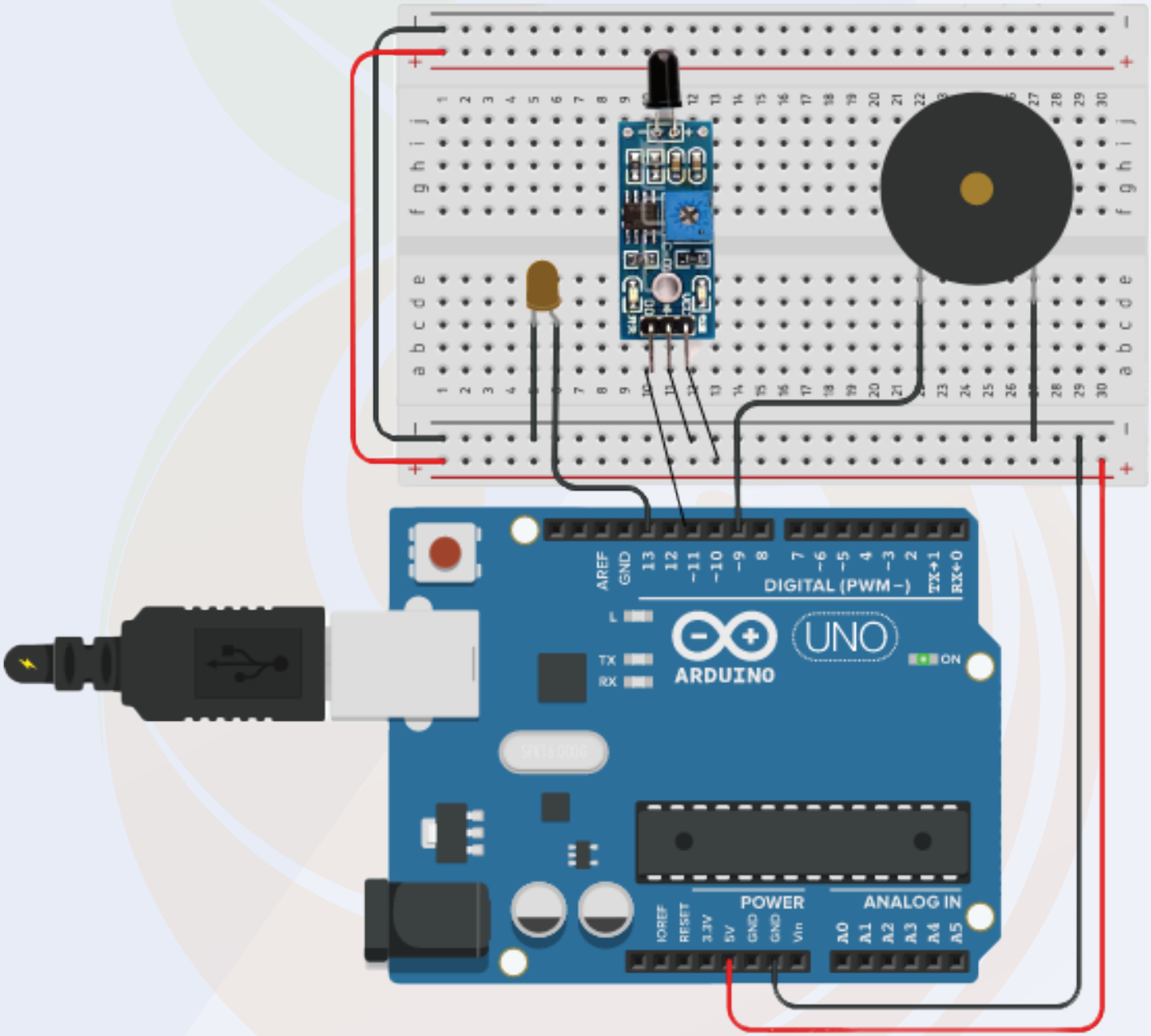
In this project, we are using flame sensor to detect fire and for alarming purposes, we are using an active buzzer.

First, we will connect the flame sensor's signal pin to pin number seven and the flame sensor's power pin to the Arduino's power pin.

Now, as I told you earlier, we are using the active buzzer as an indicator. So we have to connect the VCC pin of the flame sensor to the 5v pin of the Arduino.

Please check the following interfacing diagram to understand the connection.

Interfacing Diagram For Fire Alarm System



P1. Arduino Code For Fire Alarm System

The following code is designed for a fire alarm system. In this code, we are reading the output of the flame sensor and comparing the output of the flame sensor with the threshold value.

We are turning on the buzzer when the output value of the flame is exceeding the set value.

So it was about the fire alarm system. Now you can use the following code.

According to me you may not have any problem but if you face any problem then you can contact us.

```
#define Fire_sensor 11
#define Buzzer 9
#define Light 13
#define ldelay 500
#define bdelay 500
void setup() {
   Serial.begin(9600);
   pinMode(Fire_sensor, INPUT);
   pinMode(Buzzer, OUTPUT);
   pinMode(Light, OUTPUT);
}
void loop() {
   if (int a = digitalRead(Fire_sensor) == LOW) {
      alert();
   } else {
      digitalWrite(Buzzer, LOW);
      digitalWrite(Light, LOW);
   }
}
void alert() {
   digitalWrite(Light, HIGH);
   digitalWrite(Buzzer, HIGH);
   delay(ldelay);
   digitalWrite(Light, LOW);
   digitalWrite(Buzzer, LOW);
   delay(ldelay);
   digitalWrite(Light, HIGH);
   digitalWrite(Buzzer, HIGH);
   delay(ldelay);
   digitalWrite(Light, LOW);
   digitalWrite(Buzzer, LOW);
   delay(ldelay);
   digitalWrite(Light, HIGH);
   digitalWrite(Buzzer, HIGH);
   delay(ldelay);
   digitalWrite(Light, LOW);
   digitalWrite(Buzzer, LOW);
   delay(ldelay);
}
```

P2. Vibration Detection Using Arduino

As we have discussed in the section on Vibration Sensors, Vibration Sensors are used to detect the frequency of the object to which they are attached.

We have already talked about the basics, so you check the essential information in the above section of the blog.

List of the components

1. Bread Board
2. Arduino
3. Vibration Sensor

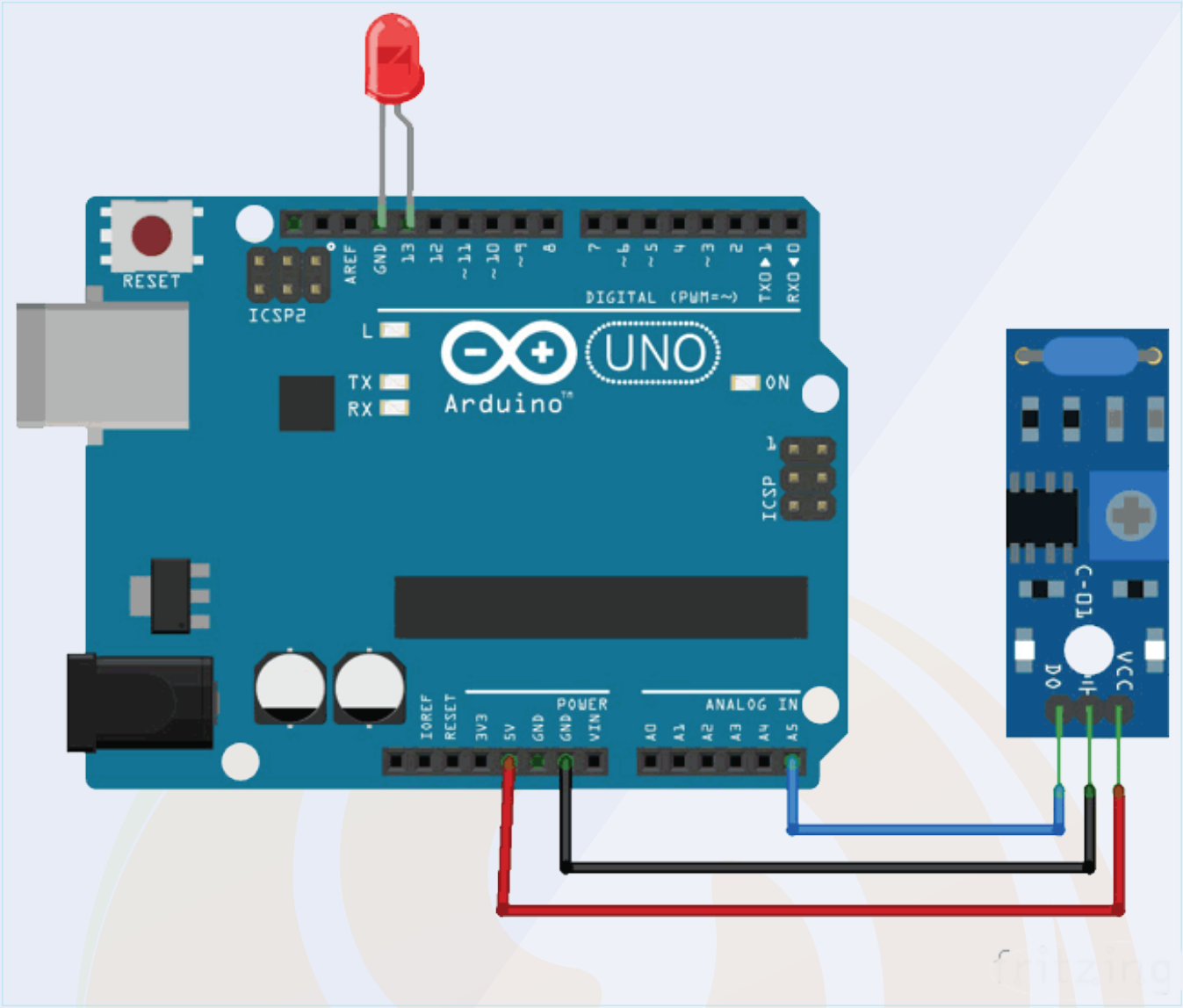
Interfacing Diagram For Vibration Sensor –

The connection diagram is similar to the diagram shown in the above sections.

We have installed the vibration sensor on the breadboard and its signal pin is connected to the number 7 pin of Arduino.

You can refer to the following image to understand the connection diagram.

Interfacing Diagram For Vibration Sensor –



Arduino Code For Vibration Sensor

We are using the same logic that we did in our previous section. In the code below we have used a variable to store the output of the sensor.

And once the output of the sensor crosses the certain threshold the code will turn on the LED

You can use the following code and see the result by vibrating the sensor.
If you have any doubt regarding this section then please let me know in the comment section.

Arduino Code For Vibration Sensor

```
int vib_pin=7;
int led_pin=13;
void setup() {
  pinMode(vib_pin,INPUT);
  pinMode(led_pin,OUTPUT);
}

void loop() {
  int val;
  val=digitalRead(vib_pin);
  if(val==1)
  {
    digitalWrite(led_pin,HIGH);
    delay(1000);
    digitalWrite(led_pin,LOW);
    delay(1000);
   }
   else
   digitalWrite(led_pin,LOW);
}
```

Thank You...!

A beginner at electronics systems comes across this common problem “How should I step into learning embedded systems? Which components should I buy? This is why Robu has designed the kits which will assist you to understand the embedded system from zero to hero level. This will solve the dilemma while selecting the proper products for your need.

Moreover, free e booklets and Video tutorials that are being shared with these kits have everything in it that's being taught within the paid lectures. That means you'll master those technical skills without paying an enormous amount to training institutes. So prepare to be the master of your dreams and start gaining the knowledge which is effective to you!

Happy Learning !!!

Buzzers are used in many instruments. It is used as an indicator in many systems such as home appliances, alarm systems, electronic bells.

Talking about the working principle of the buzzer, the buzzer converts electrical energy into sound energy.

When voltage is applied to the buzzer, the piezo crystal expands and contracts inside the plastic casing. This causes the plate to vibrate near the crystal and the sound you hear is of that vibration.

Changing the frequency to the buzzer changes the speed of the vibration and as a result you hear a variety of sounds.

So, it was about the buzzer. There are two main types of buzzers. Active buzzer and passive buzzer. In the next section of this blog, we will understand the difference between these two types.