	BALLARI INSTITUTE OF TECHNOLOGY & MANAGEMENT		
Quality Manual (ISO 9001:2015)	Doc. No: **WI/L3**	Release No. **5.0** Date: **01/07/2017**	Section No: **R/WI DSPL/01**

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

Programming with Hardware Controllers Lab Manual

Course Code:21AEC38

Prepared By

Dr. Renuka Sagar
Associate Professor
Dept OF ECE

Signature of lab-incharge HOD Signature

Department of Electronics and Communication Engineering

List of Experiments

Course: Introduction to Hardware controllers

Part-A Basic Arduino Programming

S. NO	NAME OF THE EXPERIMENT	DATE	SIGN
1	Getting started with arduino :arduino platform, prototyping environment		
2	Arduino IDE: Arduino development Environment, setting up arduino board with electronic components and connections.		
3	Arduino First program: Creating sketches, using Libraries, using example codes, Debugging using serial Monitor.		
4	Arduino interfaces-Different sensors & Actuators		
Part-B Basic Raspberrypi Programming			
5	Getting started with Raspberrypi Basic functionality of the raspberrypi board and its processor, setting and configuring the board		
6	Introduction to Linux: overview of Linux and its terminal commands for operating Raspberrypi		
7	Programming the Raspberrypi: python-Introducing to python programming language & python programming Environment		
8	Exploring Electronics with Raspberrypi: sensors & Actuator interfacing.		
Part C: open Ended Experiments/Mini-project (only for CIE, not for SEE)			

Experiment-1

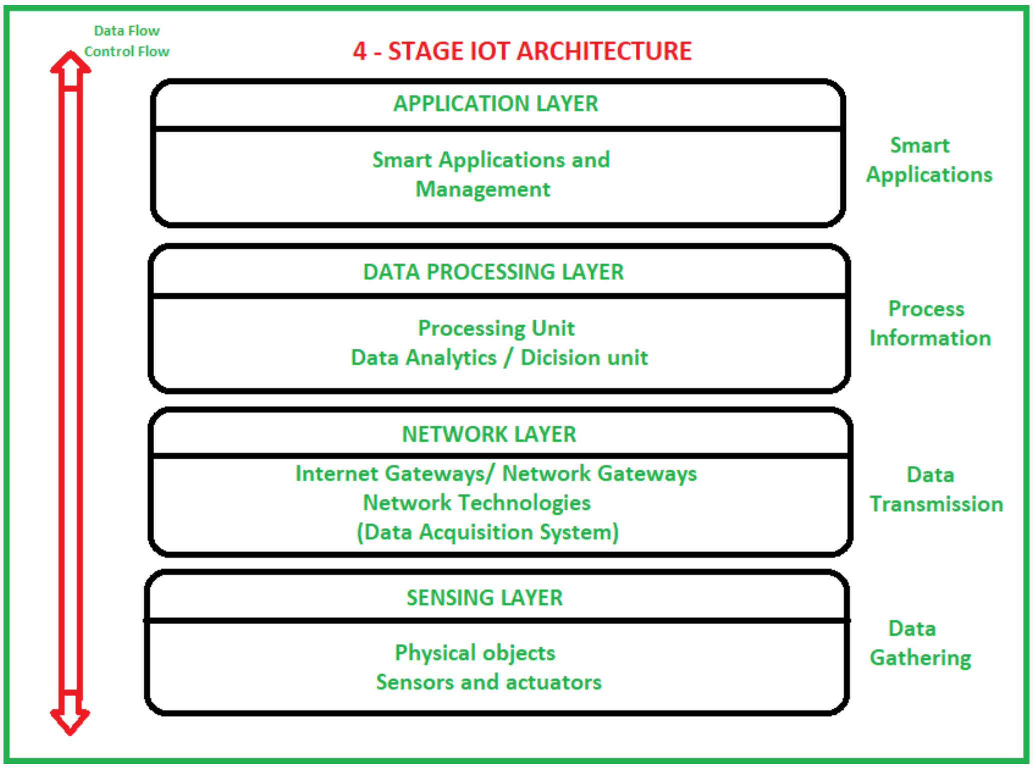
AIM: Getting started with Arduino: Arduino platform, prototyping environment

Introduction to IOT Architecture and its components.

Description:

Internet of Things (IOT) technology has a wide variety of applications and use of Internet of Things is growing so faster.

The architecture of IoT depends upon its functionality and implementation in different sectors. Fig shows basic fundamental architecture of IoT i.e., 4 Stage IoT architecture. Sensing Layer, Network Layer, Data processing Layer,and Application Layer.



These are explained as following below.

1. Sensing Layer –

Sensors, actuators, devices are present in this Sensing layer. These Sensors or Actuators accepts data(physical/environmental parameters), processes data and emits data over network.

2. Network Layer –

Internet/Network gateways, Data Acquisition System (DAS) are present in this layer. DAS performs data aggregation and conversion function (Collecting data and aggregating data then converting analog data of sensors to digital data etc). Advanced gateways which mainly opens up connection between Sensor networks and Internet also performs many basic gateway functionalities like malware protection, and filtering also some times decision making based on inputted data and data management services, etc.

3. Data processing Layer –

This is processing unit of IoT ecosystem. Here data is analyzed and pre-processed before sending it to data center from where data is accessed by software applications often termed as business applications where data is monitored and managed and further actions are also prepared. So here Edge IT or edge analytics comes into picture.

4. Application Layer –

This is last layer of 4 stages of IoT architecture. Data centers or cloud is management stage of data where data is managed and is used by end-user applications like agriculture, health care, aerospace, farming, defense, etc.

The best place to start is to define what an architecture is. Fundamentally, an architecture is a diagram or model that comprises two parts:

the key technology components that make it up and the relationship between those components.

Getting Started

Download Arduino IDE

The Arduino IDE (Integrated Development Environment) allows you to write programs and upload them to your Arduino board. Arduino also offers a web editor service called Arduino Create (https://www.arduino.cc)
Download the latest version of the IDE from arduino.cc/download.

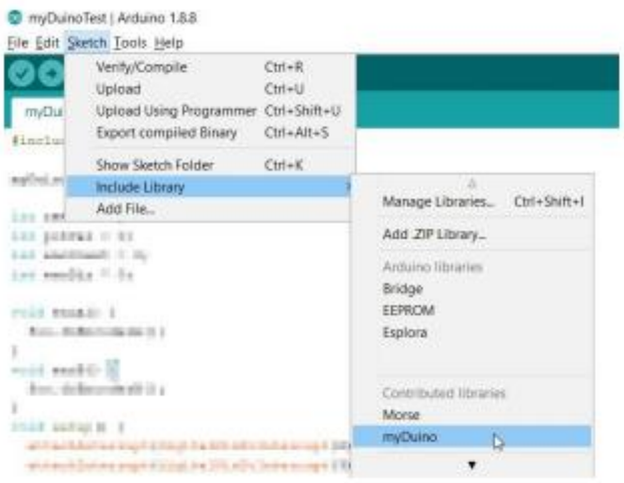
Windows Installation

1. Download the Windows Installer. Do not perform a non-admin install or download the Windows app.

2. After finishing the download, navigate to your download path and double-click on“arduino-1.x.x-rx-windows.exe” (where the x depends on the version you downloaded). If a security warning window shows up, click on “Run” or “Allow” and accept the License Agreement. Click on “next” to choose the folder to install the IDE and click on “Install”.

3. Connect the Arduino board to the computer using a proper USB cable. The board will automatically draw power from the USB connection of the computer and the green LED (labeled ON) will turn on.

4. In the “Device Manager”, under “Ports (COM & LPT)”, you should see a port similar to “Arduino Uno (COM4)”.


myDuinoTest | Arduino 1.8.8
File Edit Sketch Tools Help
Verify/Compile Ctrl+R
Upload Ctrl+U
Upload Using Programmer Ctrl+Shift+U
Export compiled Binary Ctrl+Alt+S
Show Sketch Folder Ctrl+K
Include Library
Add File...
Manage Libraries... Ctrl+Shift+I
Add .ZIP Library...
Arduino libraries
Bridge
EEPROM
Esplora
Contributed libraries
Morse
myDuino

Experiment-2

AIM : Setting up the Arduino board with electronic components and connections

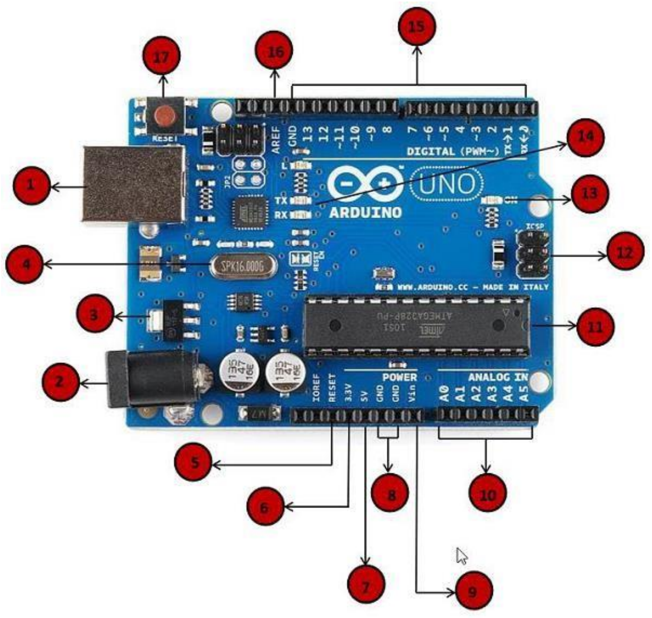
Description:

Arduino is a prototype platform (open-source) based on an easy-to-use hardware and software. It consists of a circuit board, which can be programed (referred to as a microcontroller) and a ready-made software called Arduino IDE (Integrated Development Environment), which is used to write and upload the computer code to the physical board.

The key features are −

- Arduino boards are able to read analog or digital input signals from different sensors and turn it into an output such as activating a motor, turning LED on/off, connect to the cloud and many other actions.
- You can control your board functions by sending a set of instructions to the microcontroller on the board via Arduino IDE (referred to as uploading software).
- Unlike most previous programmable circuit boards, Arduino does not need an extra piece of hardware (called a programmer) in order to load a new code onto the board. You can simply use a USB cable.
- Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program.
- Finally, Arduino provides a standard form factor that breaks the functions of the micro-controller into a more accessible package.

Daigram:



Board Description:

Arduino consists of following components listed below:

1. **Power USB:**

   Arduino board can be powered by using the USB cable from your computer.All you need to do is connect the USB.

2. **Power (Barrel Jack):**

   Arduino boards can be powered directly from the AC mains power supply by connecting it to the

Barrel Jack (2).

3. **Voltage Regulator:**
   The function of the voltage regulator is to control the voltage given to the Arduino board and stabilize the DC voltages used by the processor and other elements.
4. **Crystal Oscillator:**
   The crystal oscillator helps Arduino in dealing with time issues. How does Arduino calculate time? The answer is, by using the crystal oscillator. The number printed on top of the Arduino crystal is 16.000H9H. It tells us that the frequency is 16,000,000 Hertz or 16 MHz.
5. **Arduino Reset:**
   You can reset your Arduino board, i.e., start your program from the beginning. You can reset the UNO board in two ways. First, by using the reset button (17) on the board. Second, you can connect an external reset button to the Arduino pin labelled RESET
6. **3.3V** :Supply 3.3 output volt
7. **5V :**Supply 5 output volt
8. **GND:**
   There are several GND pins on the Arduino, any of which can be used to ground your circuit.
9. **Vin:**
   This pin also can be used to power the Arduino board from an external power source, like AC mains power supply.
10. **Analog pins:**
    The Arduino UNO board has six analog input pins A0 through A5. These pins can read the signal from an analog sensor like the humidity sensor or temperature sensor and convert it into a digital value that can be read by the microprocessor.
11. **Main microcontroller:**
    Each Arduino board has its own microcontroller . You can assume it as the brain of your board. The main IC (integrated circuit) on the Arduino is slightly different from board to board. The microcontrollers are usually of the ATMEL Company. You must know what IC your board has before loading up a new program from the Arduino IDE. This information is available on the top of the IC. For more details about the IC construction and functions, you can refer to the data sheet.
12. **Power LED indicator:**

    This LED should light up when you plug your Arduino into a power source to indicate that your board is powered up correctly. If this light does not turn on, then there is something wrong with the connection.
13. **TX and RX LEDs:**
    On your board, you will find two labels: TX (transmit) and RX (receive). They appear in two places on the Arduino UNO board. First, at the digital pins 0 and 1, to indicate the pins responsible for serial communication. Second, the TX and RX led . The TX led flashes with different speed while sending

the serial data. The speed of flashing depends on the baud rate used by the board. RX flashes during the receiving process.

14. **Digital I/O:**

    The Arduino UNO board has 14 digital I/O pins (of which 6 provide PWM (Pulse Width Modulation) output. These pins can be configured to work as input digital pins to read logic values (0 or 1) or as digital output pins to drive different modules like LEDs, relays, etc. The pins labeled “~” can be used to generate PWM.

15. **AREF:**

    AREF stands for Analog Reference. It is sometimes, used to set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pins.

Experiment-3

AIM: Arduino First program: creating sketches, using libraries, using example codes, Debugging using serial Monitor.

write an arduino sketch to perform different mathematical computations ad display the result in serial monitor

```
void setup() {
  int a = 2;
  int b = 7;
  int result;
  float result_fl;
  Serial.begin(9600);

  Serial.print("Addition (a + b): ");
  result = a + b;
  Serial.println(result);

  Serial.print("Subtraction (10 - 2): ");
  result = 10 - 2;
  Serial.println(result);

  Serial.print("Multiplication (4 * 3): ");
  result = 4 * 3;
  Serial.println(result);

  Serial.print("Int Division (5 / 4): ");
  result = 5 / 4;
  Serial.println(result);

  Serial.print("Float Division (5.0 / 4.0): ");
  result_fl = 5.0 / 4.0;
  Serial.println(result_fl);

  Serial.print("Remainder (11 % 4): ");
  result = 11 % 4;
  Serial.println(result);
}
void loop() {
}
```

Experiment-4

AIM : Arduino Interfaces:- Introduction to various types of Sensors and Actuators.

Different Types of Sensors

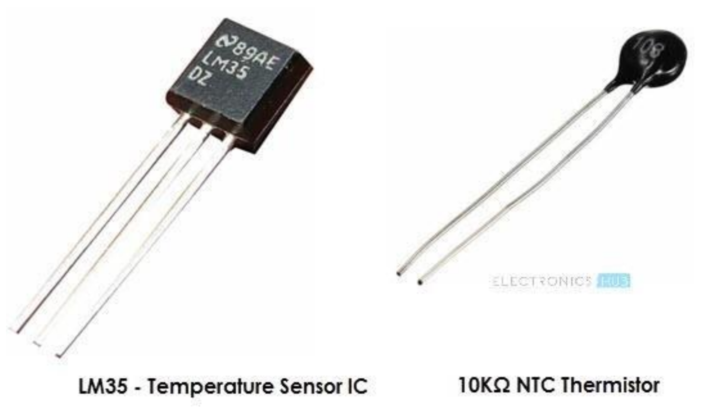
The following is a list of different types of sensors that are commonly used in various applications. All these sensors are used for measuring one of the physical properties like Temperature, Resistance, Capacitance, Conduction, Heat Transfer etc.

1. Temperature Sensor
2. Proximity Sensor
3. Accelerometer
4. IR Sensor (Infrared Sensor)
5. Pressure Sensor
6. Light Sensor
7. Ultrasonic Sensor
8. Smoke, Gas and Alcohol Sensor
9. Touch Sensor
10. Color Sensor
11. Humidity Sensor
12. Position Sensor
13. Magnetic Sensor (Hall Effect Sensor)
14. Microphone (Sound Sensor)
15. Tilt Sensor

Flow and Level Sensor

1. PIR Sensor
2. Touch Sensor
3. Strain and Weight Sensor

Temperature Sensor: One of the most common and most popular sensors is the Temperature Sensor. A Temperature Sensor, as the name suggests, senses the temperature i.e., it measures the changes in the temperature.

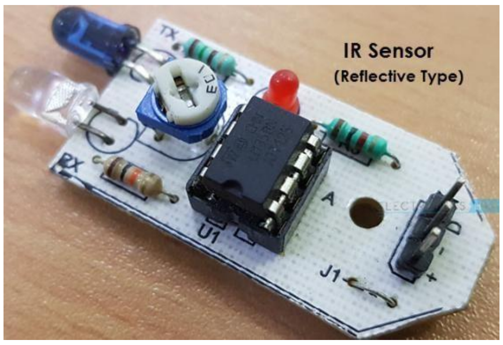


There are different types of Temperature Sensors like Temperature Sensor ICs (like LM35, DS18B20), Thermistors, Thermocouples, RTD (Resistive Temperature Devices), etc.Temperature Sensors can be analog or digital. In an Analog Temperature Sensor, the changes in the Temperature correspond to change in its physical property like resistance or voltage. LM35 is a classic Analog Temperature Sensor.Coming to the Digital Temperature Sensor, the output is a discrete digital value (usually, some numerical data after converting analog value to digital value).

Proximity Sensors: A Proximity Sensor is a non-contact type sensor that detects the presence of an object. Proximity Sensors can be implemented using different techniques like Optical (like Infrared or Laser), Sound (Ultrasonic), Magnetic (Hall Effect), Capacitive, etc.

Some of the applications of Proximity Sensors are Mobile Phones, Cars (Parking Sensors), industries (object alignment), Ground Proximity in Aircrafts, etc.

Infrared Sensor (IR Sensor):IR Sensors or Infrared Sensor are light based sensor that are used in various applications like Proximity and Object Detection. IR Sensors are used as proximity sensors in almost all mobile phones.



There are two types of Infrared or IR Sensors: Transmissive Type and Reflective Type. In Transmissive Type IR Sensor, the IR Transmitter (usually an IR LED) and the IR Detector (usually a Photo Diode) are positioned facing each other so that when an object passes between them, the sensor detects the object .The other type of IR Sensor is a Reflective Type IR Sensor. In this, the transmitter and the detector are positioned adjacent to each other facing the object. When an object comes in front of the sensor, the infrared light from the IR Transmitter is reflected from the object and is detected by the IR Receiver and thus the sensor detects the object. Different applications where IR Sensor is implemented are Mobile Phones, Robots, Industrial assembly, automobiles etc.

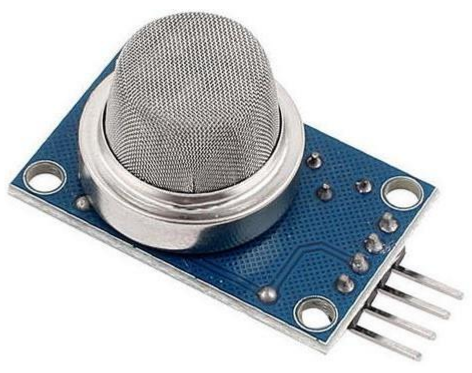
Ultrasonic Sensor: An Ultrasonic Sensor is a non-contact type device that can be used to measure distanceas well as velocity of an object. An Ultrasonic Sensor works based on the properties of the sound waves with frequency greater than that of the human audible range.

Using the time of flight of the sound wave, an Ultrasonic Sensor can measure the distance of the object (similar to SONAR). The Doppler Shift property of the sound wave is used to measure the velocity of an object.

Light Sensor: Sometimes also known as Photo Sensors, Light Sensors are one of the important sensors. A simple Light Sensor available today is the Light Dependent Resistor or LDR. The property of LDR is that its resistance is inversely proportional to the intensity of the ambient light i.e., when the intensity of light increases, its resistance decreases and vise-versa. By using LDR is a circuit, we can calibrate the changes in its resistance to measure the intensity of Light. There are two other Light Sensors (or Photo Sensors) which are often used in complex electronic system design. They are Photo Diode and Photo Transistor. All these are Analog Sensors.

There are also Digital Light Sensors like BH1750, TSL2561, etc., which can calculate intensity of light and provide a digital equivalent value.Check out this simple LIGHT DETECTOR USING LDR project.

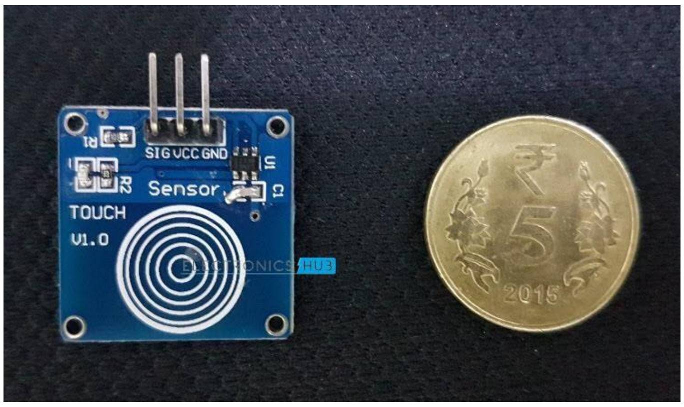
Smoke and Gas Sensors: One of the very useful sensors in safety related applications are Smoke and Gas Sensors. Almost all offices and industries are equipped with several smoke detectors, which detect any smoke (due to fire) and sound an alarm. Gas Sensors are more common in laboratories, large scale kitchens and industries. They can detect different gases like LPG, Propane, Butane, Methane (CH4), etc.



Now-a-days, smoke sensors (which often can detect smoke as well gas) are also installed in most homes as a safety measure. The “MQ” series of sensors are a bunch of cheap sensors for detecting CO, CO2, CH4, Alcohol, Propane, Butane, LPG etc. You can use these sensors to build your own Smoke Sensor Application.

Alcohol Sensor: As the name suggests, an Alcohol Sensor detects alcohol. Usually, alcohol sensors are used in breathalyzer devices, which determine whether a person is drunk or not. Law enforcement personnel uses breathalyzers to catch drunk-and-drive culprits.

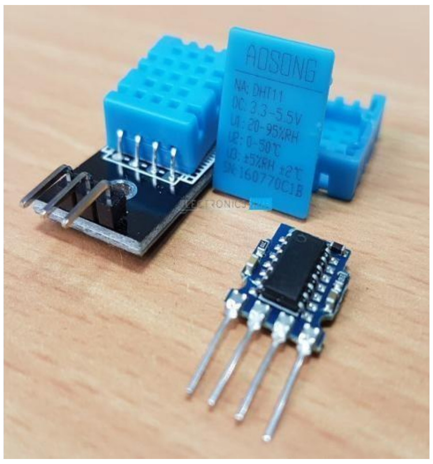
Touch Sensor: We do not give much importance to touch sensors but they became an integral part of our life. Whether you know or not, all touch screen devices (Mobile Phones, Tablets, Laptops, etc.) have touch sensors in them. Another common application of touch sensor is trackpads in our laptops.



Touch Sensors, as the name suggests, detect touch of a finger or a stylus. Often touch sensors are classified into Resistive and Capacitive type. Almost all modern touch sensors are of Capacitive Types as they are more accurate and have better signal to noise ratio.

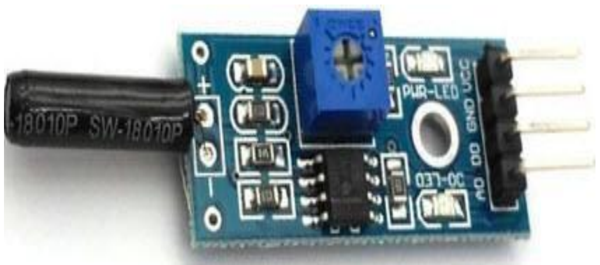
Color Sensor: A Color Sensor is an useful device in building color sensing applications in the field of image processing, color identification, industrial object tracking etc. The TCS3200 is a simple Color Sensor, which can detect any color and output a square wave proportional to the wavelength of the detected color.

Humidity Sensor: If you see Weather Monitoring Systems, they often provide temperature as well as humidity data. So, measuring humidity is an important task in many applications and Humidity Sensors help us in achieving this. Often all humidity sensors measure relative humidity (a ratio of water content in air to maximum potential of air to hold water). Since relative humidity is dependent on temperature of air, almost all Humidity Sensors can also measure Temperature.



Humidity Sensors are classified into Capacitive Type, Resistive Type and Thermal Conductive Type. DHT11 and DHT22 are two of the frequently used Humidity Sensors in DIY Community (the former is a resistive type while the latter is capacitive type).

Tilt Sensor: Often used to detect inclination or orientation, Tilt Sensors are one of the simplest and inexpensive sensors out there. Previously, tilt sensors are made up of Mercury (and hence they are sometimes called as Mercury Switches) but most modern tilt sensors contain a rollerball.



PROGRAM 1 LED LIGHT

AIM: Interface LED with Arduino and write a program to turn on LED. LED should be turn on for 1 second after every 2 Second.

Components:

The components used are:

1. Arduino Uno
2. Breadboard
3. LED
4. Resistor

DESCRIPTION:

Arduino Uno R3:

Arduino UNO is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

Pins used:

Digital pin 13: The pin13 is connected to the Anode terminal of the Led through a Resistor on Breadboard. GND: The Ground is connected to the Cathode of Led.

Breadboard:

A Breadboard is simply a board for prototyping or building circuits on. It allows you to place components and connections on the board to make circuits without soldering. The holes in the breadboard take care of your connections by physically holding onto parts or wires where you put them and electrically connecting them inside the board.

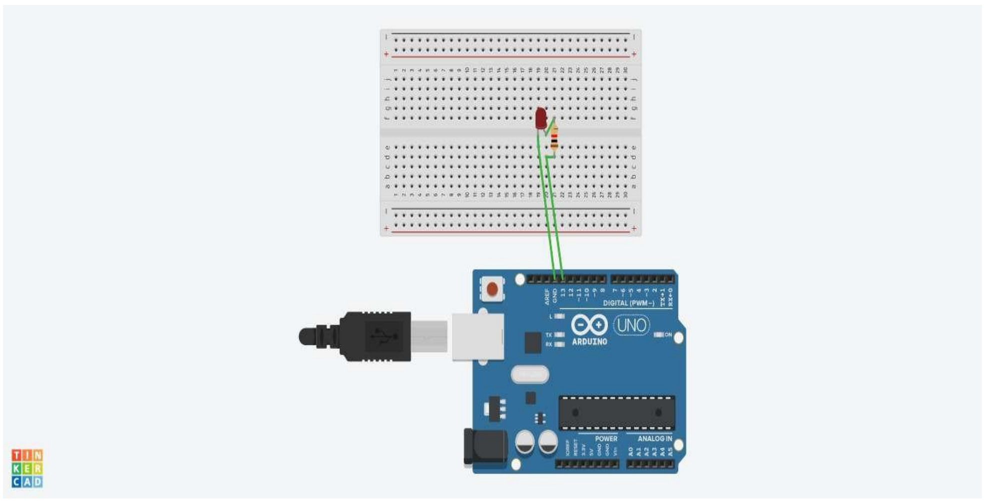
LED:

Led has two terminals Anode and Cathode .The shorter leg toward the flat edge of Led is the Negative terminal of the Led.

Resistor:

LEDs need resistors to help limit the current that passes through them so they do not get damaged. Every LED has a current rating that should not be exceeded and resistors have the ability to limit the current to below the maximum allowable current allowed for the LED.

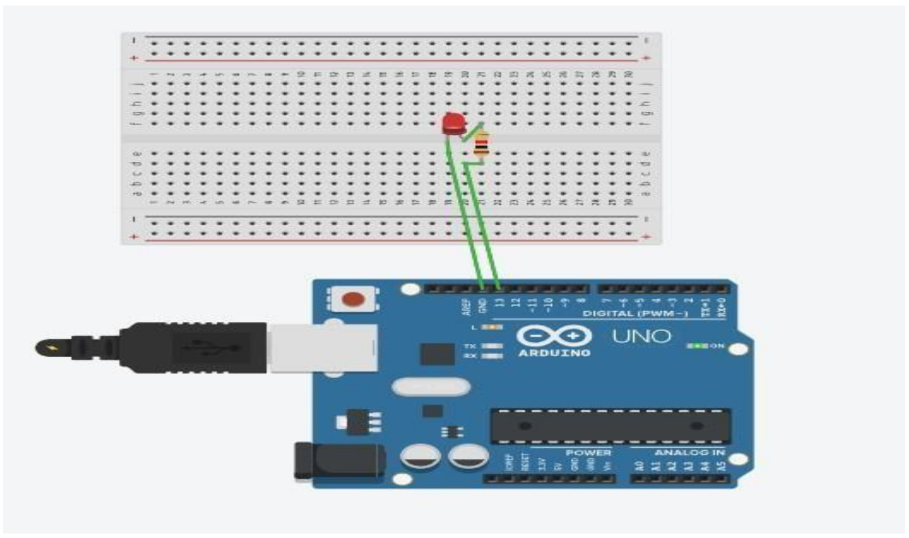
CIRCUIT VIEW



CODE:

```
void setup()
{
 pinMode(13, OUTPUT);
}
void loop()
{
 digitalWrite(13, HIGH);
 delay(1000); // Wait for 1000
 digitalWrite(13, LOW);// millisecond(s)
 delay(2000); // Wait for 200 millisecond(s)
}
```

OUTPUT:



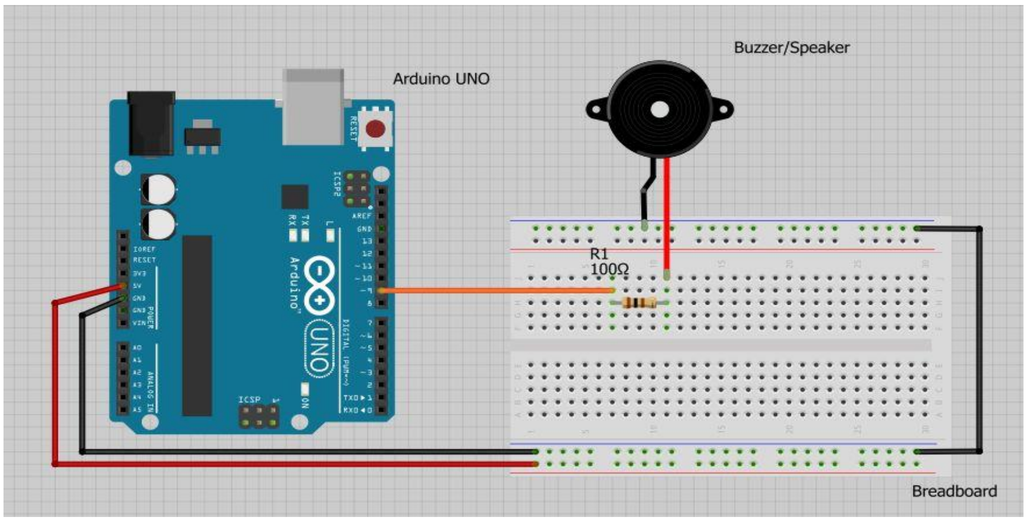
PROGRAM 2 BUZZER

AIM: Interface Buzzer with Arduino and write a program to turn on Buzzer. Buzzer should be turn on for 1 second after every 2 Second.

Components Required :

1. Arduino Uno R3
2. Breadboard
3. Buzzer
4. Resistor

CIRCUIT VIEW



CODE:

```
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

      }
```

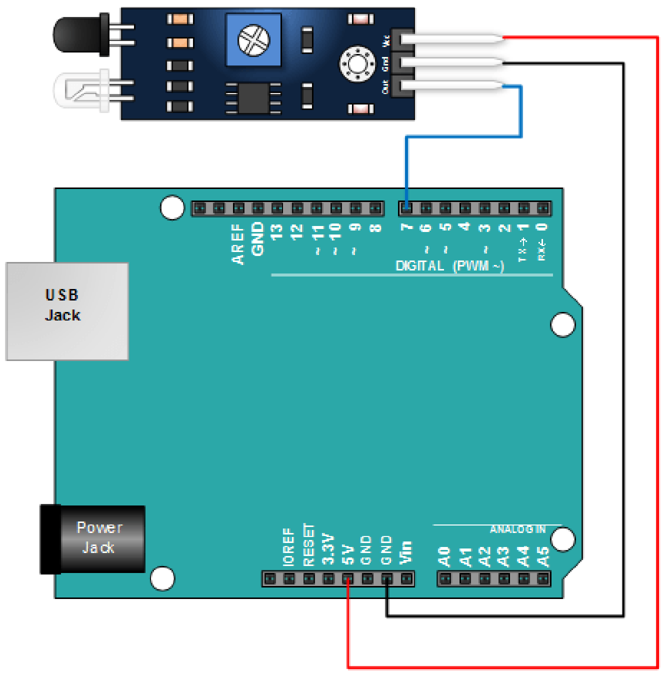
PROGRAM 3 IR SENSOR

AIM: write an Arduino sketch to Interface IR Sensor module with Arduino .

Component Required:

NAME	QUANTITY	COMPONENT
1	1	Arduino Uno R3
2	1	IR Sensor
3	Few	Jumper wires

CIRCUIT VIEW



CODE:

```
int LED = 13; // Use the onboard Uno LED

int isObstaclePin = 7;  // This is our input pin

int isObstacle = HIGH;  // HIGH MEANS NO OBSTACLE

void setup() {

  pinMode(LED, OUTPUT);

  pinMode(isObstaclePin, INPUT);

  Serial.begin(9600);

  }

void loop() {

  isObstacle = digitalRead(isObstaclePin);

  if (isObstacle == LOW)

  {

    Serial.println("OBSTACLE!!, OBSTACLE!!");

    digitalWrite(LED, HIGH);

  }

  else

  {

    Serial.println("clear");

    digitalWrite(LED, LOW);

  }

  delay(200);

}
```

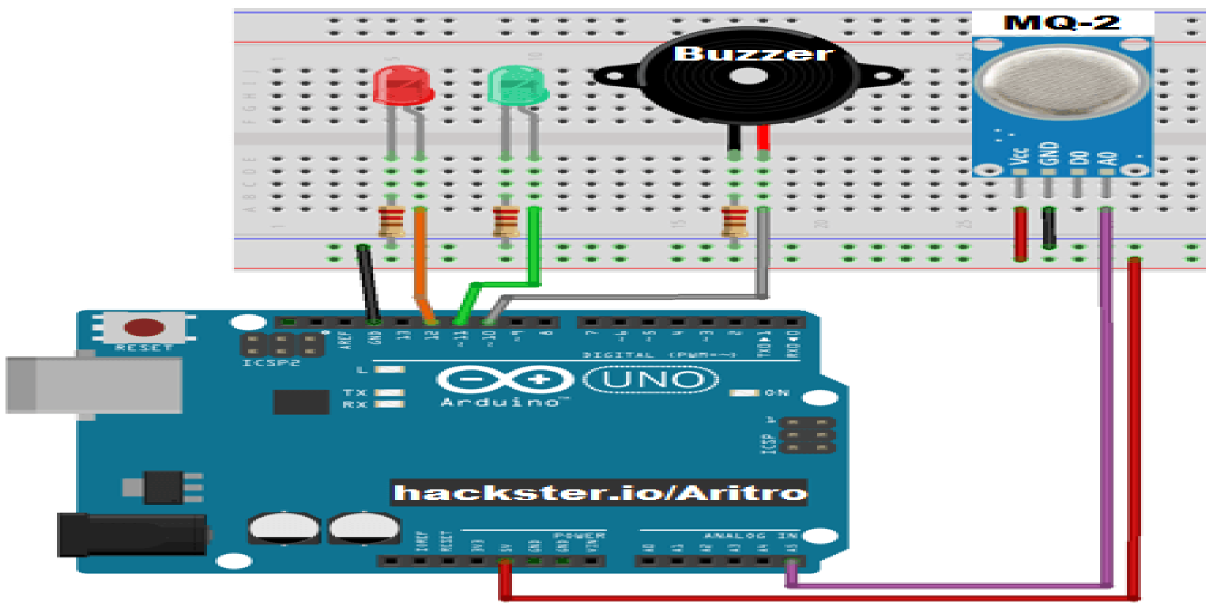
PROGRAM 4

AIM: write an arduino sketch to Interface MQ-2 Sensor with Arduino/Raspberry Pi and write a program to print its readings**.**

Component Required:

NAME	QUANTITY	COMPONENT
U1	1	Arduino Uno R3
U2	1	MQ-2 Sensor
U3	2	LED
U4	1	Buzzer

CIRCUIT VIEW



CODE:

```
int redLed = 12;

int greenLed = 11;

int buzzer = 10;

int smokeA0 = A5;

// Your threshold value

int sensorThres = 400;

void setup() {

  pinMode(redLed, OUTPUT);

  pinMode(greenLed, OUTPUT);

  pinMode(buzzer, OUTPUT);

  pinMode(smokeA0, INPUT);

  Serial.begin(9600);

}

void loop() {

  int analogSensor = analogRead(smokeA0);

  Serial.print("Pin A0: ");

  Serial.println(analogSensor);

  // Checks if it has reached the threshold value

  if (analogSensor > sensorThres)

  {

    digitalWrite(redLed, HIGH);

    digitalWrite(greenLed, LOW);

     tone(buzzer, 1000, 200);

  }
else

  {

    digitalWrite(redLed, LOW);

    digitalWrite(greenLed, HIGH);

    noTone(buzzer);

  }

  delay(100);

}
```

PROGRAM 5 ULTRASONIC SENSOR

AIM: write an arduino sketch to Interface Ultrasonic sensor with Arduino/Raspberry Pi and write a programto print its readings.

Component View:

NAME	QUANTITY	COMPONENT
U1	1	Arduino Uno R3
U2	1	Ultrasonic Sensor
R1	1	220 ohm Resistor

CIRCUIT VIEW

CODE:

```
 const int pingPin =7;  //Trigger Pin of Ultrasonic sensor

const int echo Pin=6;  // Echo Pin of Ultrasonic sensor

void setup( )  {

Serial . begin(9600)

}

void loop( )    {

long duration,inches,cm;

pinMode(pingPin,OUTPUT);

digitalWrite(pingPin,LOW);

delayMicroseconds(2000);

digitalWrite(pingPin,HIGH);

delayMicroseconds(1000);

digitalWrite(pingPin,LOW);

pinMode(echoPin,INPUT);

duration =pulseln(echoPin,HIGH);

inches=microsecondsToInches(duration);

cm=microsecondsToCentimeters(duration);

Serial.print(inches);

Serial.print(“in”);

Serial.print(cm);

Serial.print(“cm”);

Serial.println( );

delay(100);

}
```

```
long microsecondsToInches(long microseconds)

{

return microseconds / 74 / 2;

}

long microsecondsToCentimeters(long microseconds)

{

return microseconds / 29 / 2;

}
```

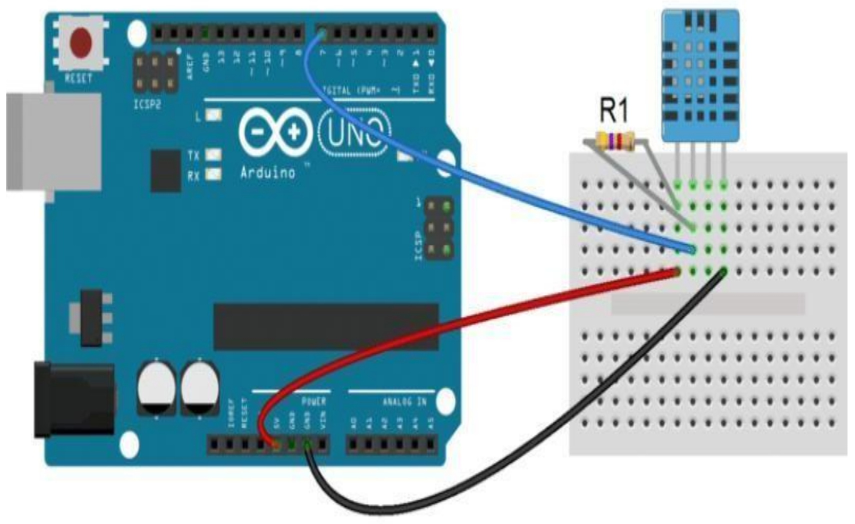
PROGRAM 6 DHT11 SENSOR

AIM: write an arduino sketch to Interface DHT11 sensor with Arduino and print its values.

Component View:

NAME	QUANTITY	COMPONENT
U1	1	Arduino Uno R3
U2	1	DHT11 Sensor
R1	1	220 ohm Resistor

CIRCUIT VIEW



CODE:

```
#include <dht.h> dht DHT;

#define DHT11_PIN 7
void setup(){

        Serial.begin(9600);

}
void loop(){

        int chk = DHT.read11(DHT11_PIN);

        Serial.print("Humidity = ");

        Serial.println(DHT.humidity);

        Serial.print("Temperature = ");

        Serial.println(DHT.temperature);

        delay(1000);

}
```

OUTPUT:-

```
Humidity = 15.12% Temperature = 40.17 C 104.30 F

Humidity = 15.12% Temperature = 40.17 C 104.30 F

Humidity = 15.15% Temperature = 40.56 C 105.01 F

Humidity = 15.12% Temperature = 40.17 C 104.30 F

Humidity = 15.12% Temperature = 40.17 C 104.30 F

Humidity = 15.12% Temperature = 40.17 C 104.30 F
```

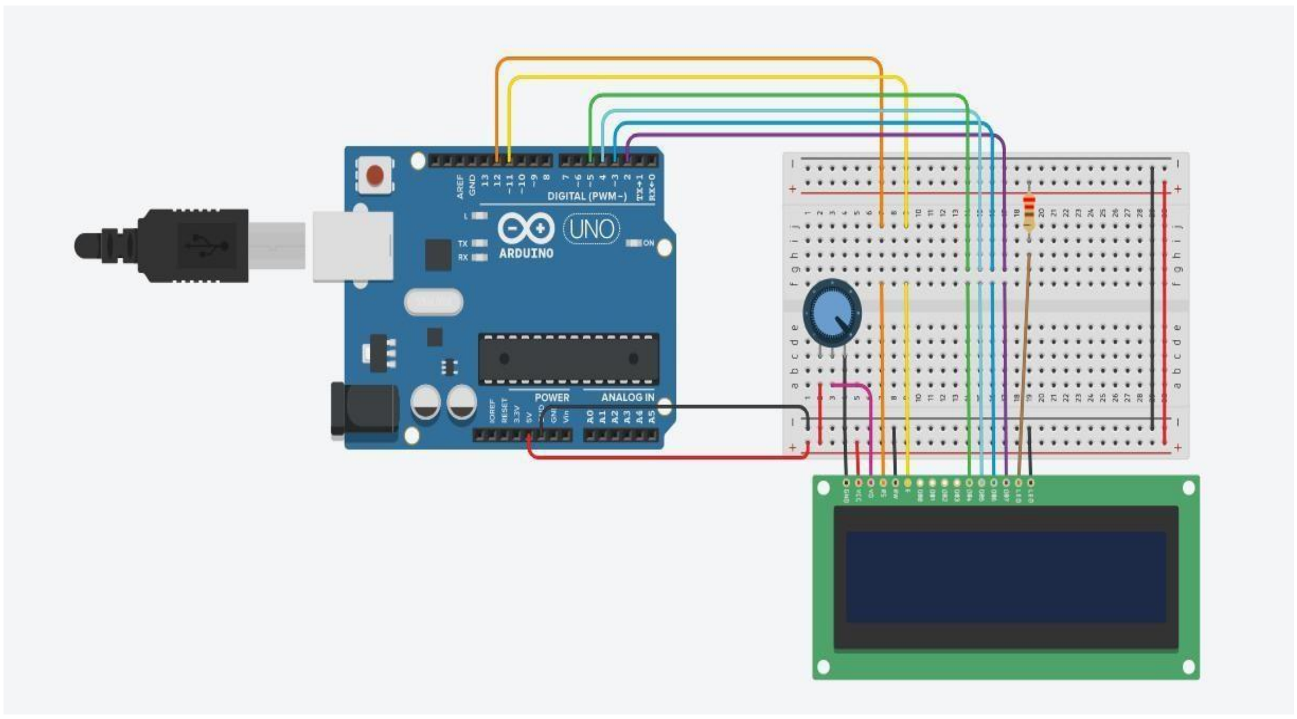
PROGRAM 7 LCD DISPLAY

AIM : Interface 16*2 LCD with Arduino/ Raspberry Pi and write a program toprint your name and simulation window second time duration on LCD.

Component view:

Name	Quantity	Component
U1	1	Arduino Uno R3
U2	1	LCD 16 x 2
Rpot1	1	250 kΩ Potentiometer
R1	1	220 Ω Resistor

CIRCUIT VIEW

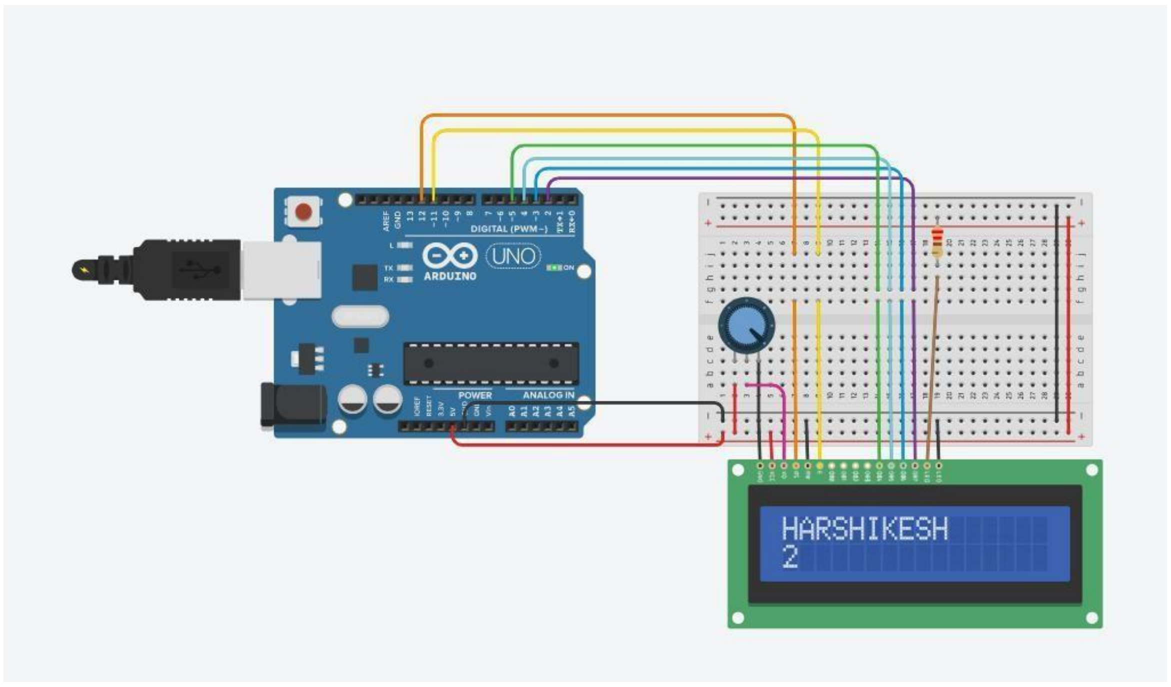


CODE :

```
//include the library code:
#include <LiquidCrystal.h>


//initialize the library with the numbers of the interfacepins
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {

//set up the LCD's number of columns and rows:
lcd.begin(10, 2);
//Print a message to the LCD.
lcd.print("BITM,DEPT OF
ECE");
}


void loop() {

// set the cursor to column 0, line 1

//(note: line 1 is the second row, since counting begins with 0):
lcd.setCursor(0, 1);
// print the number of seconds since reset:

lcd.print(millis() / 1000);


}
```

Simulation :

When you turn the potentiometer , the LCD 16*2 display the “HARSHIKESH”.



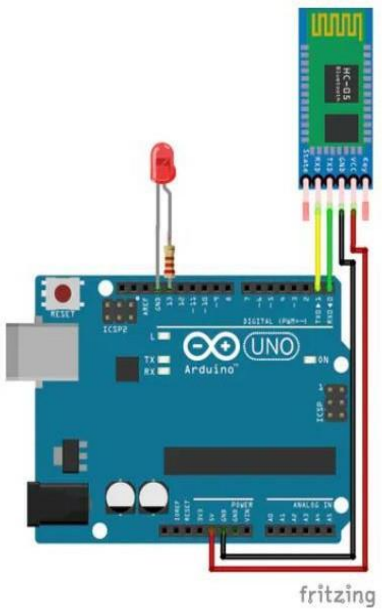
PROGRAM 8

AIM :To interface Bluetooth with Arduino and write a program to send sensor data to smartphone using Bluetooth.

<u>Component View</u> :

NAME	QUANTITY	COMPONENT
U1	1	Arduino Uno R3
U2	1	HC-05 Bluetooth Module
R1	1	221 OHM Resistor
D1	1	Red LED
U3	1	Android Device

<u>CIRCUIT VIEW</u>



CODE :

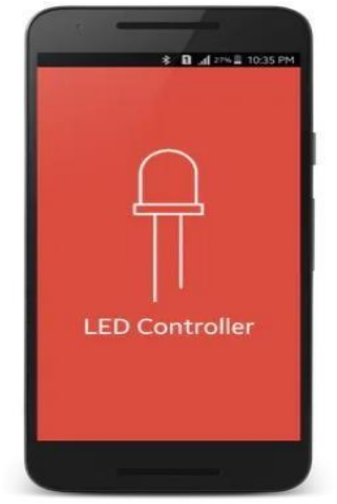
```
char Incoming_value = 0;

void setup() {

Serial.begin(9600);

pinMode(13, OUTPUT);

}

void loop(){

if(Serial.available() > 0) {

Incoming_value = Serial.read();

   Serial.print(Incoming_value);

   Serial.print("\n");

   if(Incoming_value == '1')

   digitalWrite(13, HIGH);

   else if(Incoming_value == '0')

   digitalWrite(13, LOW);

   }

}
```

Simulation :

How to connect your andriod device with bluebooth module

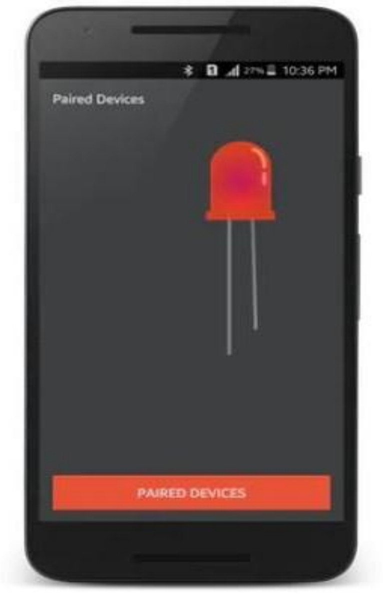
✓ Pair your device with HC 05/06 Bluetooth module1) Turn ON HC 05/06 Bluetooth module2) Scan for available device3) Pair to HC 05/06 by entering default password 1234 OR 0000

- ✓ D application on your android device
- ✓ Open the Application

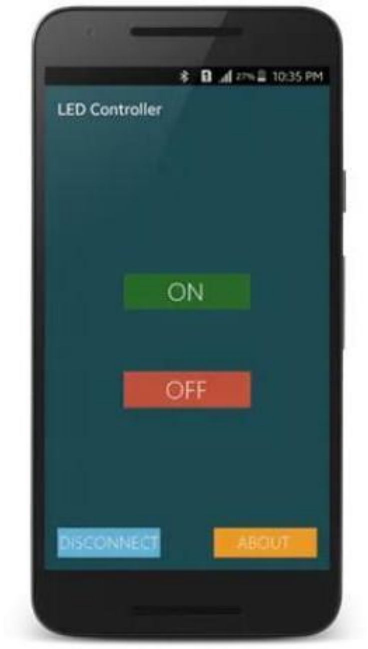


splash screen

- ✓ Press paired devices
- ✓ Select your Bluetooth module from the List (HC 05)



- ✓ After connecting successfully
- ✓ Press ON button to turn ON LED and OFF button to turn OFF the LED

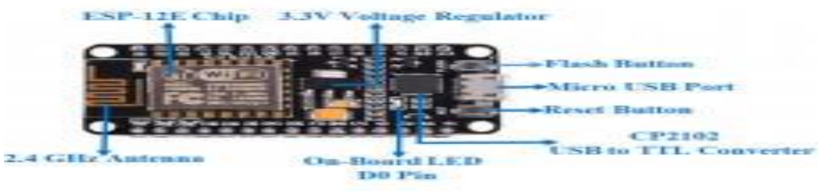


PROGRAM 10

AIM : Programming NodeMCU ESP8266 with Arduino IDE

Description:

NodeMCU is an open-source Lua based firmware and **development board** specially targeted for IoT based Applications. It includes firmware that runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module.



```
#include <ESP8266WiFi.h>

const char* ssid = "Magesh";
const char* password = "jayakumar";

int ledPin = 13; // GPIO13
WiFiServer server(80);

void setup() {
  Serial.begin(115200);
  delay(10);

  pinMode(ledPin, OUTPUT);
  digitalWrite(ledPin, LOW);

  // Connect to WiFi network
  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");

  // Start the server
  server.begin();
  Serial.println("Server started");

  // Print the IP address
  Serial.print("Use this URL to connect: ");
  Serial.print("http://");
  Serial.print(WiFi.localIP());
  Serial.println("/");

}

void loop() {
  // Check if a client has connected
  WiFiClient client = server.available();
  if (!client) {
    return;
  }
```