Division of Data Science and Cyber Security

School of Engineering and Technology

Lab Manual

**23DC2007– IOT AND ITS APPLICATIONS LAB**

(0:0:1)

Department of Data Science and Cyber Security

School of Engineering and Technology

Lab Manual

**23DC2007– IOT AND ITS APPLICATIONS LAB**

(0:0:1)

|  |  |  |
| --- | --- | --- |
| Prepared by | Verified by | Approved by |
| Dr. Antony Taurshia |  | Dr. Grace Mary Kanaga |

|  |  |  |
| --- | --- | --- |
| INDEX | | |
| 1 | Vision & Mission Statements | 4 |
| 2 | PEO, PO& PSO- B.Tech. Artificial Intelligence and Data Science | 5 |
| 3 | PEO, PO & PSO - B.Tech. ComputerEngineering | 7 |
| 4 | Syllabus |  |
| 5 | List of Exercises |  |

**VISION**

To nurture and raise leaders, excelling in academia, industry, interdisciplinary research, innovation in the domains of Artificial Intelligence, Data Science and Cyber Security, with a passionate commitment to provide solutions to critical societal problems.

**MISSION**

* + To train students with strong foundational concepts, adept programming skills and efficient problem-solving abilities with exposure to emerging technologies, enabling them to tackle the digital security challenges.
  + To advance research in cutting-edge technologies, with a focus on delivering analytical solutions, especially in the domains of healthcare, water, energy and food
  + To build qualities of entrepreneurial leadership imbued with ethical values and a commitment to serve and uplift society.

### PEO, PO & PSO - B.Tech. Artificial Intelligence and Data Science

**Program Educational Objectives (PEOs)**

Graduates will

1. demonstrate the knowledge acquired in artificial intelligence and data science, to analyse andidentifytherequirements, formulateanddevelop innovative intelligent solutions.
2. Applyartificial intelligence anddatascience proficiency as professionals, academicians and entrepreneurs to solve human problems.
3. practise ethical and moral values and serve the humanity with social concern.

### Program Outcomes (POs)

1. **Engineering Knowledge:** Apply the knowledge of mathematics, natural sciences, engineering fundamentals specialization to the solution of complex engineering problems.
2. **Problem Analysis:** Identify, formulate, review research literature and analyze complex Engineering Problems reaching substantiated conclusions using first principles of mathematics, natural sciencesand engineering sciences.
3. **Design / Development of Solutions:** Design solutions for complex engineering problems and design system components or processes that meet specific needs with appropriate consideration for the public health and safety and the cultural, societal and environmentalconsiderations.
4. **Conduct Investigations of Complex Problems:** Use research- based knowledge and research methods including design of experiments, analysis and interpretation of data and synthesis of the information to provide valid conclusions.
5. **Modern Tool Usage:** Create, Select and Apply appropriate techniques, resources and modern engineering and IT tools including prediction and modellingtocomplexengineering activitieswithanunderstandingofthe limitations.
6. **The Engineer and Society:** Apply Reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and Sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts and demonstrate the knowledge of and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineeringpractice.
9. **Individual and Team Work:** Function effectively as an individual and as a member or leader in diverse teams and in multi-disciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large such as being able to comprehend and write effective reports and

design documentation make effective presentations and give and receive clear instructions.

1. **Project Management and Finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply this to one‟s own work, as a member and leader in a team, to manage projects and in multi-disciplinary environments.
2. **Life – Long Learning:** Recognize the need for and have the preparation and ability to engage in independent and lifelong leaning in the broadest context of technological change.

### Program Specific Outcome (PSOs)

Graduates will have ability to

1. understand, analyse, design and develop intelligent systems of varying complexity using statistical and computational principles in the areas of algorithms, artificial intelligence, datascience, robotics, multimediaand distributed systems
2. apply contemporary tools and techniques of artificial intelligence and data science in exhibiting skills for employability, entrepreneurship and research.

### B.Tech. Computer Engineering

**Program Educational Objectives (PEOs)**

Graduates will

1. demonstrate the knowledge attained, to analyse the technical requirements, and develop secure innovativesolutions.
2. provide solutions for real world problems through the assimilated expertise as technical professionals, academicians and entrepreneurs
3. contribute to the humanity by exercising the code of ethics and professional practice.

### Program Outcomes (POs)

1. **Engineering Knowledge:** Apply the knowledge of mathematics, natural sciences, engineering fundamentals specialization to the solution of complex engineering problems.
2. **Problem Analysis:** Identify, formulate, review research literature and analyze complex Engineering Problems reaching substantiated conclusions using first principles of mathematics, natural sciencesand engineering sciences.
3. **Design / Development of Solutions:** Design solutions for complex engineering problems and design system components or processes that meet specific needs with appropriate consideration for the public health and safety and the cultural, societal and environmentalconsiderations.
4. **Conduct Investigations of Complex Problems:** Use research- based knowledge and research methods including design of experiments, analysis and interpretation of data and synthesis of the information to provide valid conclusions.
5. **Modern Tool Usage:** Create, Select and Apply appropriate techniques, resources and modern engineering and IT tools including prediction and modellingtocomplexengineering activitieswithanunderstandingofthe limitations.
6. **The Engineer and Society:** Apply Reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and Sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts and demonstrate the knowledge of and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineeringpractice.
9. **Individual and Team Work:** Function effectively as an individual and as a member or leader in diverse teams and in multi-disciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large such as being able to comprehend and write effective reports and design documentation make effective presentations and give and receive clear instructions.
11. **Project Management and Finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply this to one‟s own work, as a member and leader in a team, to manage projects and in multi-disciplinary environments.
12. **Life – Long Learning:** Recognize the need for and have the preparation and ability to engage in independent and lifelong leaning in the broadest context of technological change.

### Program Specific Outcome (PSOs)

Graduates will have ability to

1. understand, analyse and develop products and software solutions using standard practices and strategies in the areas of algorithms, communication networks, real time systems, system administration, robotics, virtual reality, artificial intelligence, data science and cyber security.
2. employ modern programming languages and platforms in the field of computing to develop skills for employability, entrepreneurship, and research.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **23DC2007** | **IoT and its Applications Lab** | **L** | **T** | **P** | **C** |
| **0** | **0** | **2** | **1** |

**Course Objectives:**

Enable the student to

1. Describe the IoT devices, application areas and technologies involved.
2. Illustrate the IoT sensors and technological challenges faced by IoT devices.
3. Explore and learn IoT with the help of Raspberry Pi and Arduino.

**Course Outcomes:**

The student will be able to

1. Explain the building blocks of Internet of Things and characteristics

2. Apply the concepts of IOT.

3. Identify various sensors used for applications.

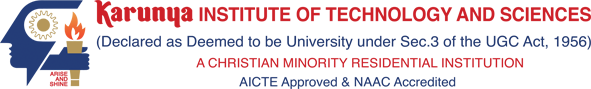
4. Describe the working principles of sensors, Arduino boards.

5. Apply IoT to various small-scale real-world applications.

6. Analyse the data received through sensors in IOT.

**List of Experiments:**

1. Interfacing LED with an Arduino
2. Controlling an LED with a Push Button using Arduino
3. Interfacing a Temperature Sensor with an Arduino
4. Interfacing an LCD and a Temperature Sensor with an Arduino
5. Interfacing an Ultrasonic Sensor with an Arduino to Measure Distance
6. Interfacing an Ultrasonic Sensor and an LCD with an Arduino
7. Interfacing a DC Motor with an Arduino
8. Direction Control of a DC Motor Using an Arduino
9. Speed and Direction Control of a DC Motor Using an Arduino
10. Familiarization with Raspberry Pi/ Node MCU to perform necessary software installation and interface LED/Buzzer with Raspberry Pi.



## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

**LAB MANUAL**

# 23DC2007- IoT and its Applications Lab

CREDIT 0:0:1

|  |  |  |
| --- | --- | --- |
|  | **Name** | **Signature** |
| Prepared by | Dr.V.Ebenezer |  |
| Approved by | Dr. Grace Mary Kanaga, HOD/ DSCS |  |

P a g e 2 | 44

**TABLE OF CONTENTS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.No.** | **Name of the Experiment** | **Page No.** | **Marks** | **Signature** |
| 1 | Interfacing LED with an Arduino |  |  |  |
| 2 | Controlling an LED with a Push Button using Arduino |  |  |  |
| 3 | Interfacing a Temperature Sensor with an Arduino |  |  |  |
| 4 | Interfacing an LCD and a Temperature Sensor with an Arduino |  |  |  |
| 5 | Interfacing an Ultrasonic Sensor with an Arduino to Measure Distance |  |  |  |
| 6 | Interfacing an Ultrasonic Sensor and an LCD with an Arduino |  |  |  |
| 7 | Interfacing a DC Motor with an Arduino |  |  |  |
| 8 | Direction Control of a DC Motor Using an Arduino |  |  |  |
| 9 | Speed and Direction Control of a DC Motor Using an Arduino |  |  |  |
| 10 | Familiarization with Raspberry Pi/ Node MCU to perform necessary software installation and interface DHT11 Sensor with Raspberry Pi. |  |  |  |

|  |  |
| --- | --- |
| **Exp. No. 1** | **INTERFACING LED WITH AN ARDUINO** |
| **Date:** |

### Aim:

To interface three LEDs with an Arduino and to program the Arduino to make the LEDs

turn on and off periodically.

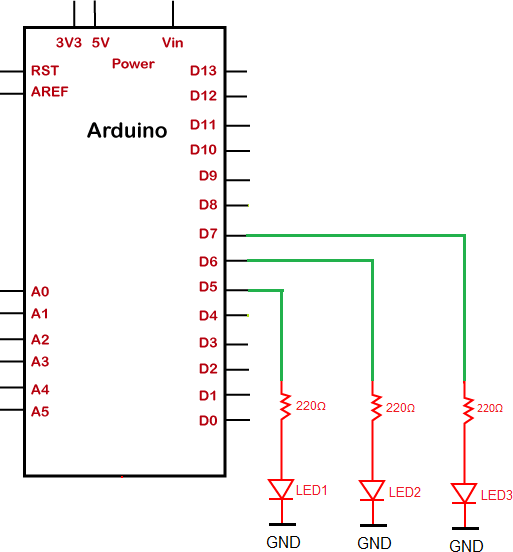
### Software required:

Tinkercad

### Procedure:

1. Open Tinkercad and click on “Create new Circuit”.
2. In the page that opens, choose the Arduino and the components in the circuit from the “Components” section and place them on the left side of the screen.
3. Wire the components as per the circuit diagram.
4. Click on “Code”, choose “Text” from the drop-down menu, and enter the code to program the Arduino.
5. Click on “Start Simulation” to simulate the circuit and verify the output.

### Circuit diagram:



**Program:**

void setup()

{

pinMode(7, OUTPUT); pinMode(6, OUTPUT); pinMode(5, OUTPUT);

}

void loop()

{

digitalWrite(7, HIGH);

delay(1000); // Wait for 1000 millisecond(s) digitalWrite(6, HIGH);

delay(1000); // Wait for 1000 millisecond(s) digitalWrite(5, HIGH);

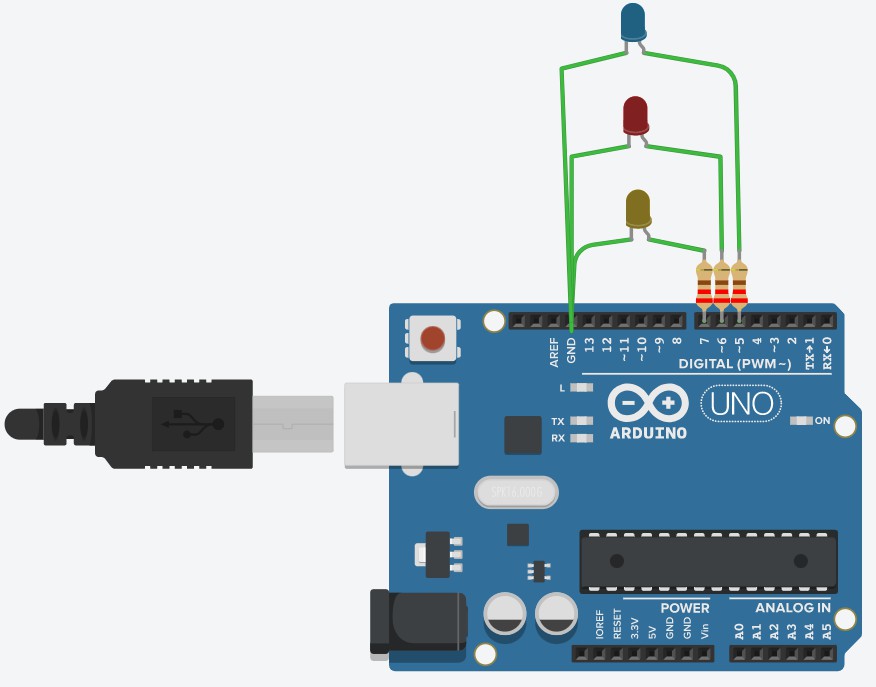
delay(1000); // Wait for 1000 millisecond(s) digitalWrite(7, LOW);

digitalWrite(6, LOW); digitalWrite(5, LOW);

delay(1000); // Wait for 1000 millisecond(s)

}

### Screenshot:



**Result:**

LEDs were interface with an Arduino and the Arduino was programmed to make the LEDs turn on and off at regular intervals.

|  |  |
| --- | --- |
| **Exp. No. 2** | **CONTROLLING AN LED WITH A PUSH BUTTON USING ARDUINO** |
| **Date:** |

### Aim:

To interface an LED and a push button with an Arduino and to program the Arduino to

make the LED turn on whenever the button is pressed.

### Software required:

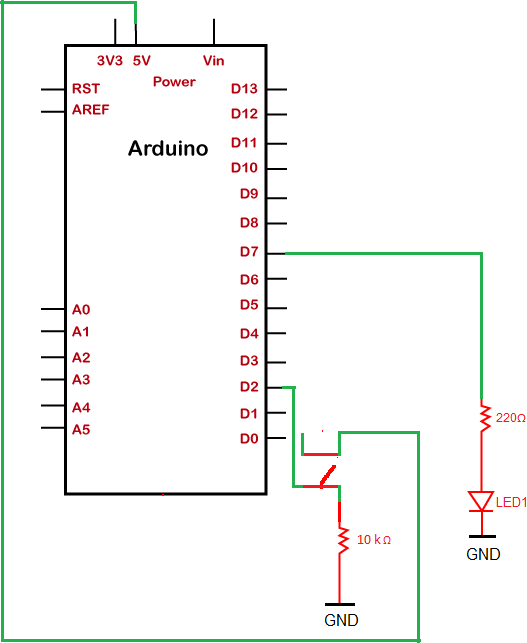
Tinkercad

### Procedure:

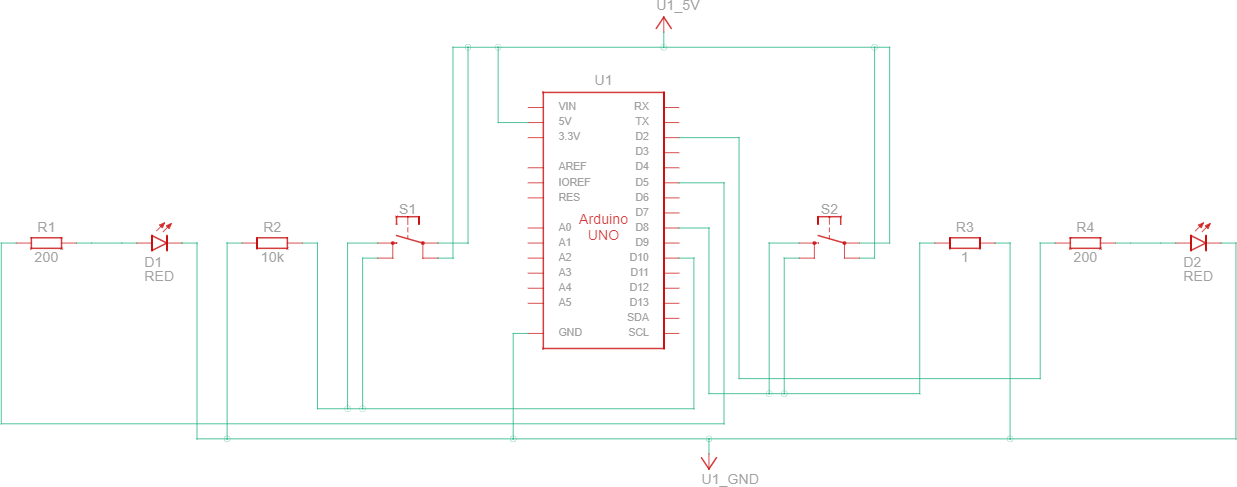
1. Open Tinkercad and click on “Create new Circuit”.
2. In the page that opens, choose the Arduino and the components in the circuit from the “Components” section and place them on the left side of the screen.
3. Wire the components as per the circuit diagram.
4. Click on “Code”, choose “Text” from the drop-down menu, and enter the code to program the Arduino.
5. Click on “Start Simulation” to simulate the circuit and verify the output.

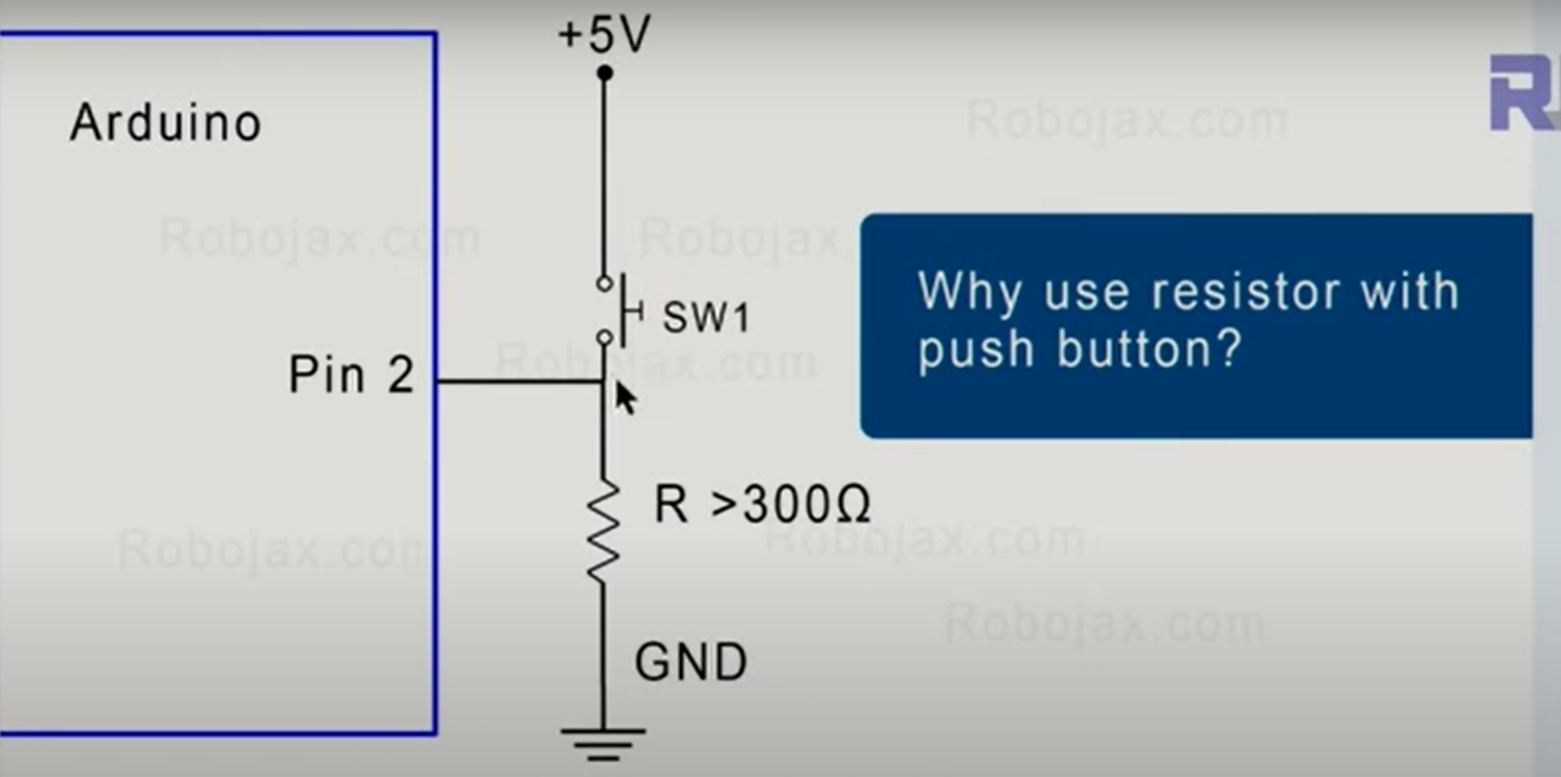
**Circuit diagram:**

1. **LED and 1 push button**



1. **LEDs and 2 push buttons**





**Program:**

1. **LED and 1 push button**

void setup()

{

pinMode (7, OUTPUT);

pinMode (2, INPUT);

}

void loop()

{

int bstate = digitalRead(2); digitalWrite(7, LOW);

if (bstate == HIGH)

{

digitalWrite(7, HIGH);

}

else

{

digitalWrite(7, LOW);

}

}

### LEDs and 2 push buttons

void setup()

{

pinMode(5, OUTPUT); pinMode(2, OUTPUT); pinMode(10, INPUT);

pinMode(8, INPUT);

}

void loop()

{

int bstate1 = digitalRead(10); int bstate2 = digitalRead(8); if(bstate1 == HIGH)

{

digitalWrite(5, HIGH);

}

else

{

digitalWrite(5, LOW);

}

if(bstate2 == HIGH)

{

digitalWrite(2, HIGH);

}

else

{

digitalWrite(2, LOW);

}

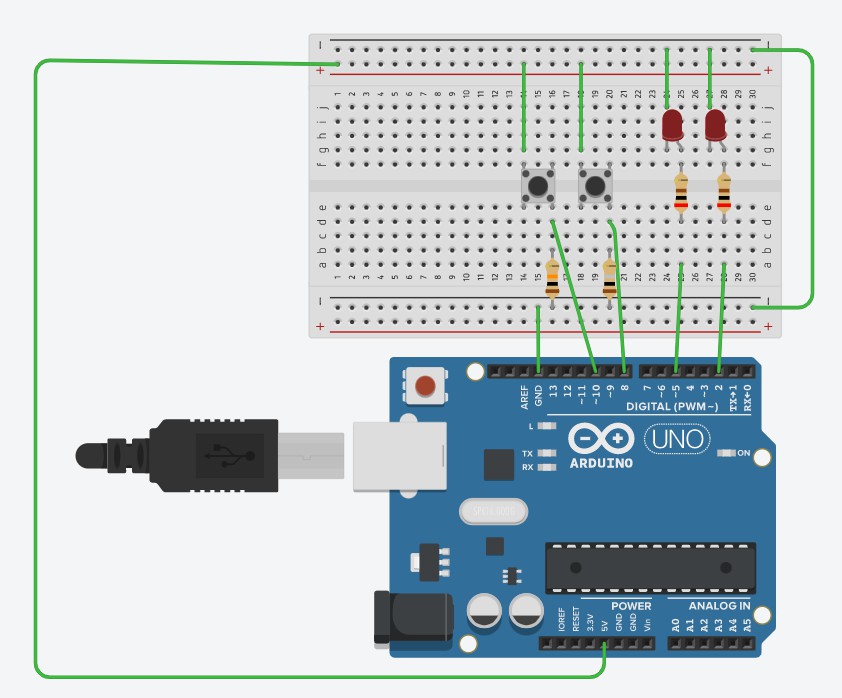
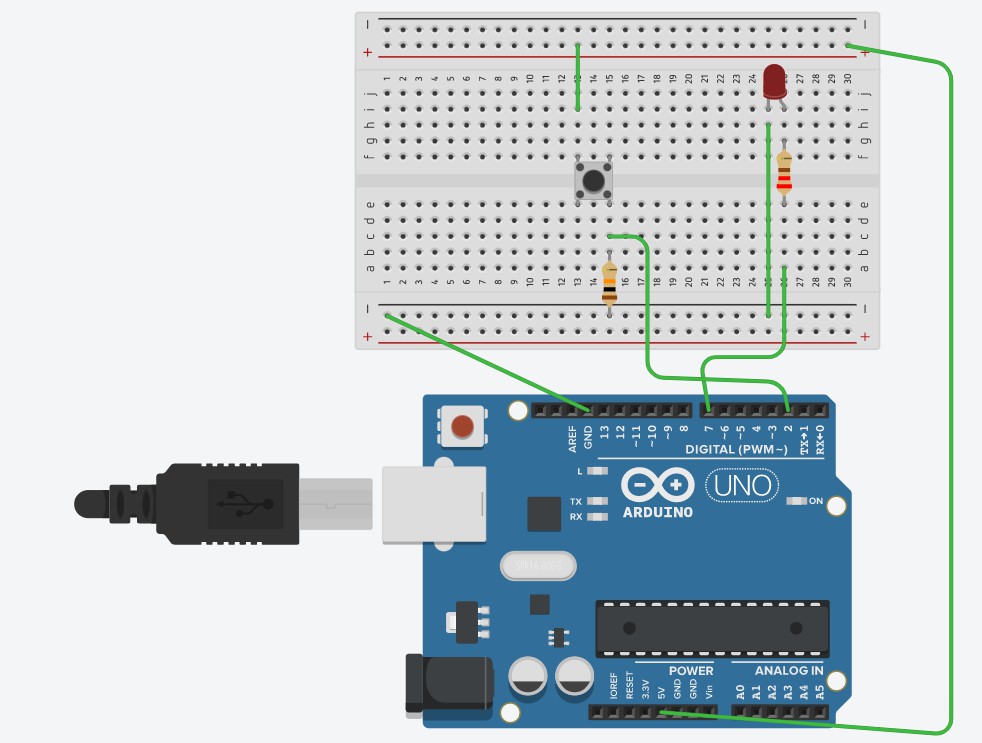
}

Extra Programs:

1. Glowing three different LED&#39;s using three Push Button

2. Generating Traffic Light Scenario using push button

### Screenshot:



**Result:**

An LED and a push button were interfaced with an Arduino and the push button was used to turn on the LED by programming the Arduino.

|  |  |
| --- | --- |
| **Exp. No. 3** | **INTERFACING A TEMPERATURE SENSOR WITH AN**  **ARDUINO** |
| **Date:** |

### Aim:

To interface a temperature sensor with an Arduino and to print the detected temperature

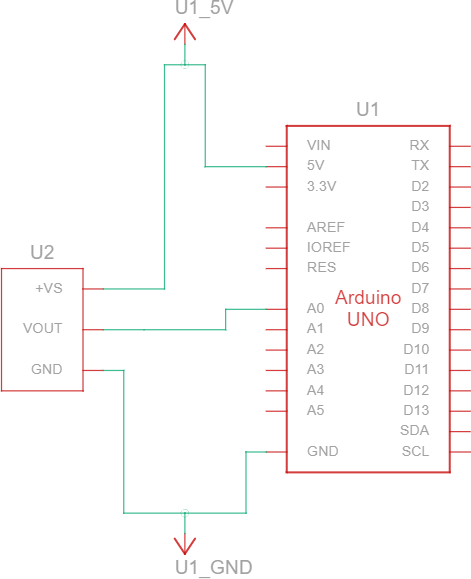
### Software required:

Tinkercad

### Procedure:

* 1. Open Tinkercad and click on “Create new Circuit”.
  2. In the page that opens, choose the Arduino and the components in the circuit from the “Components” section and place them on the left side of the screen.
  3. Wire the components as per the circuit diagram.
  4. Click on “Code”, choose “Text” from the drop-down menu, and enter the code to program the Arduino.
  5. Click on “Start Simulation” to simulate the circuit and verify the output.

### Circuit diagram:



**Program:**

int sensorInput; double temp; void setup()

{

Serial.begin(9600);

}

void loop()

{

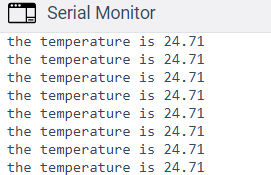
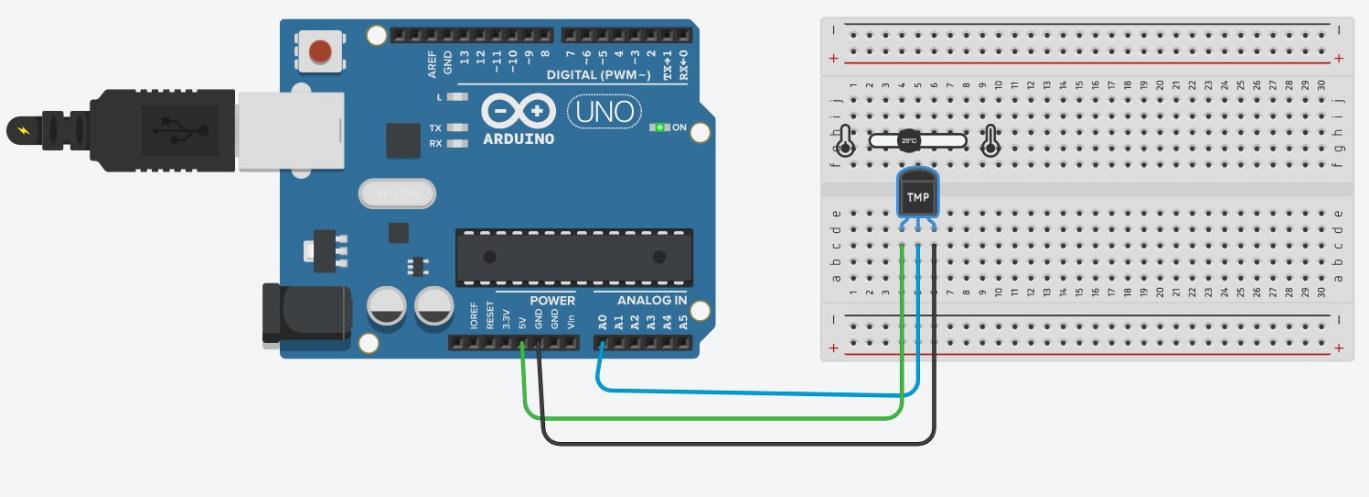
sensorInput = analogRead(A0); temp = (double)sensorInput/1024; temp = temp \* 5;

temp = temp - 0.5; temp = temp \* 100;

Serial.print("the temperature is "); Serial.println(temp);

}

### Screenshot:



**Result:**

A temperature sensor has been interfaced with an Arduino and the detected temperature has been printed.

|  |  |
| --- | --- |
| **Exp. No. 4** | **INTERFACING AN LCD AND A TEMPERATURE**  **SENSOR WITH AN ARDUINO** |
| **Date:** |

### Aim:

To interface an LCD and a temperature sensor with an Arduino and to display the

detected temperature on the LCD.

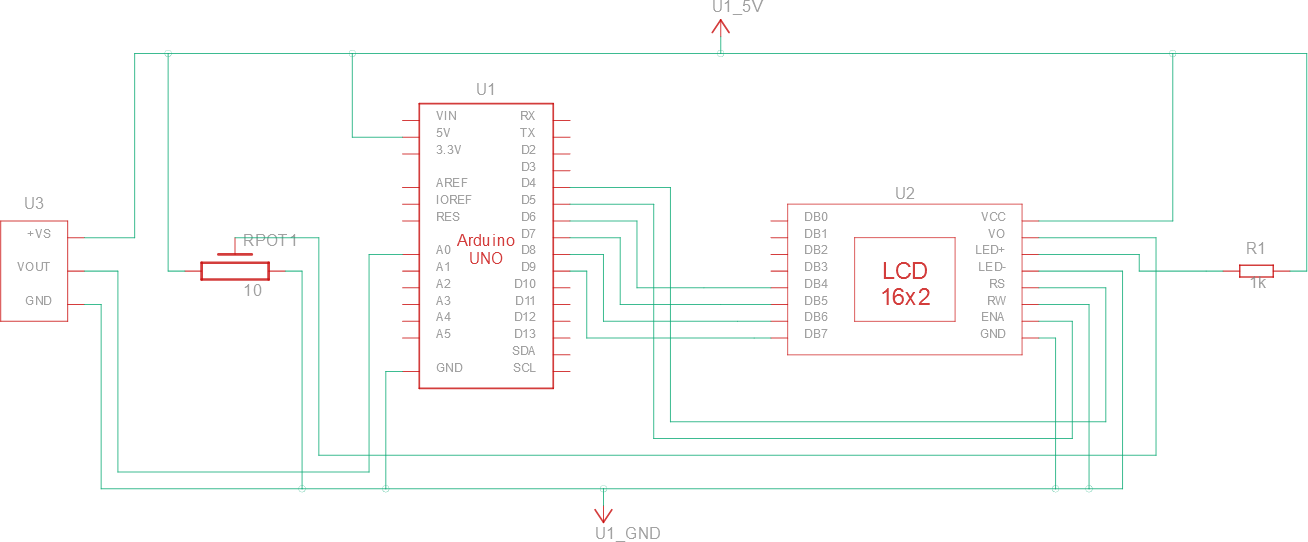
### Software required:

Tinkercad

### Procedure:

1. Open Tinkercad and click on “Create new Circuit”.
2. In the page that opens, choose the Arduino and the components in the circuit from the “Components” section and place them on the left side of the screen.
3. Wire the components as per the circuit diagram.
4. Click on “Code”, choose “Text” from the drop-down menu, and enter the code to program the Arduino.
5. Click on “Start Simulation” to simulate the circuit and verify the output.

### Circuit diagram:



**Program:**

#include <LiquidCrystal.h>

LiquidCrystal lcd(4,5,6,7,8,9); // rs, en, d4, d5, d6, d7 void setup()

{

Serial.begin(9600); pinMode(A0,INPUT); lcd.begin(16,2); lcd.print("Temperature");

}

void loop()

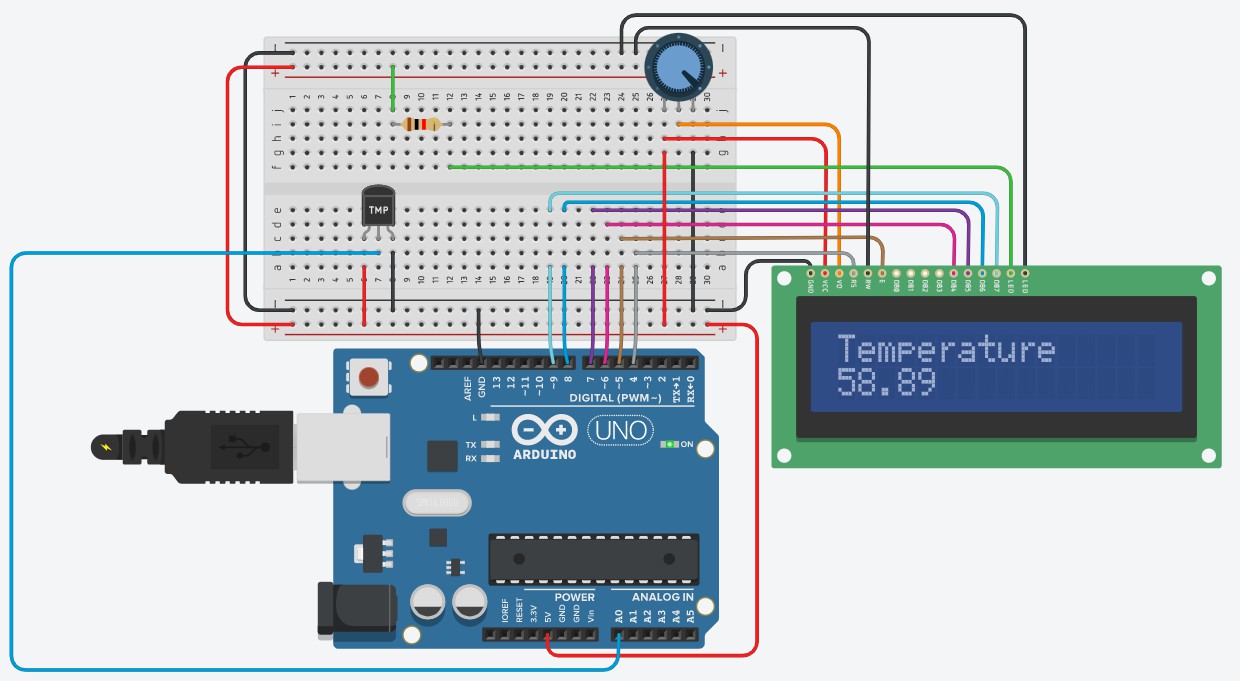
{

float temp = ((analogRead(A0)\*(5.0/1024))-0.5)/0.01; lcd.setCursor(0,1);

lcd.print(temp); Serial.println(temp); delay(1000);

}

### Screenshot:



**Result:**

An LCD and a temperature sensor have been interfaced with an Arduino and the detected temperature has been displayed on the LCD.

|  |  |
| --- | --- |
| **Exp. No. 5** | **INTERFACING AN ULTRASONIC SENSOR WITH AN**  **ARDUINO TO MEASURE DISTANCE** |
| **Date:** |

### Aim:

To interface an ultrasonic sensor with an Arduino to measure the distance of an object

from the sensor and to turn on LEDs depending on the distance of the object.

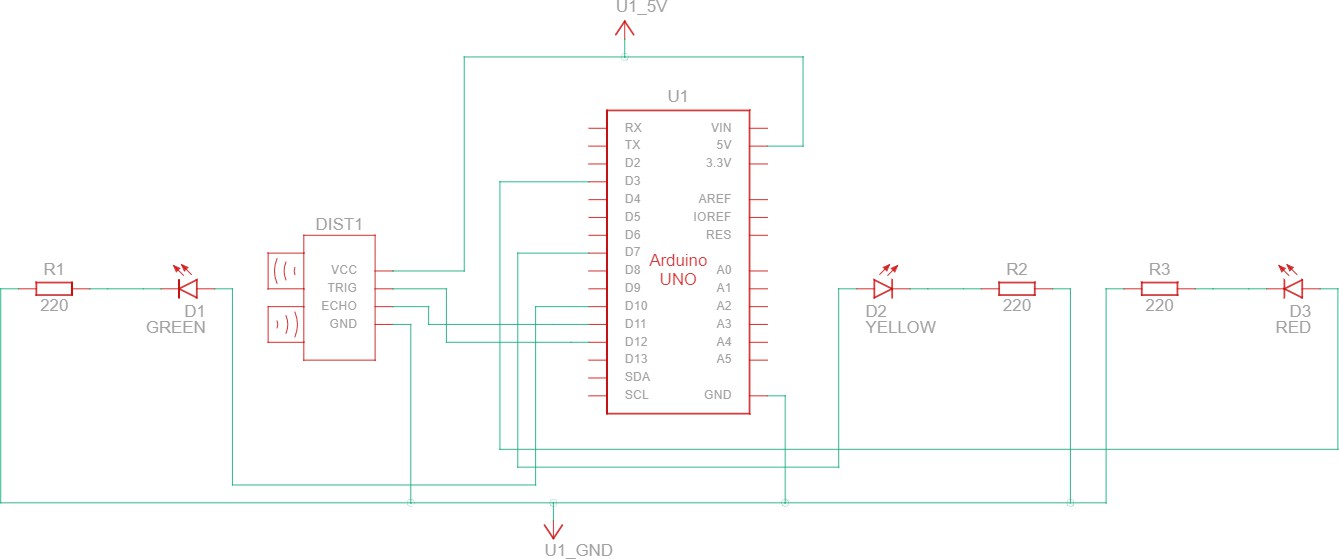
### Software required:

Tinkercad

### Procedure:

1. Open Tinkercad and click on “Create new Circuit”.
2. In the page that opens, choose the Arduino and the components in the circuit from the “Components” section and place them on the left side of the screen.
3. Wire the components as per the circuit diagram.
4. Click on “Code”, choose “Text” from the drop-down menu, and enter the code to program the Arduino.
5. Click on “Start Simulation” to simulate the circuit and verify the output.

### Circuit diagram:



**Program:**

long duration; int distance;

void setup()

{

pinMode(10, OUTPUT); pinMode(7, OUTPUT); pinMode(3, OUTPUT); pinMode(12, OUTPUT); pinMode(11, INPUT); Serial.begin(9600);

}

void loop()

{

digitalWrite(12, LOW); delayMicroseconds(2); digitalWrite(12, HIGH); delayMicroseconds(10); digitalWrite(12, LOW);

duration = pulseIn(11, HIGH); distance = duration\*0.034/2;

Serial.print("Distance: "); Serial.println(distance);

if (distance > 200)

{

digitalWrite(10, LOW); digitalWrite(7, LOW); digitalWrite(3, LOW);

}

else if (distance <= 199 && distance > 175)

{

digitalWrite(10, HIGH); digitalWrite(7, LOW); digitalWrite(3, LOW);

}

else if (distance <= 174 && distance > 150)

{

digitalWrite(10, LOW); digitalWrite(7, HIGH); digitalWrite(3, LOW);

}

else

{

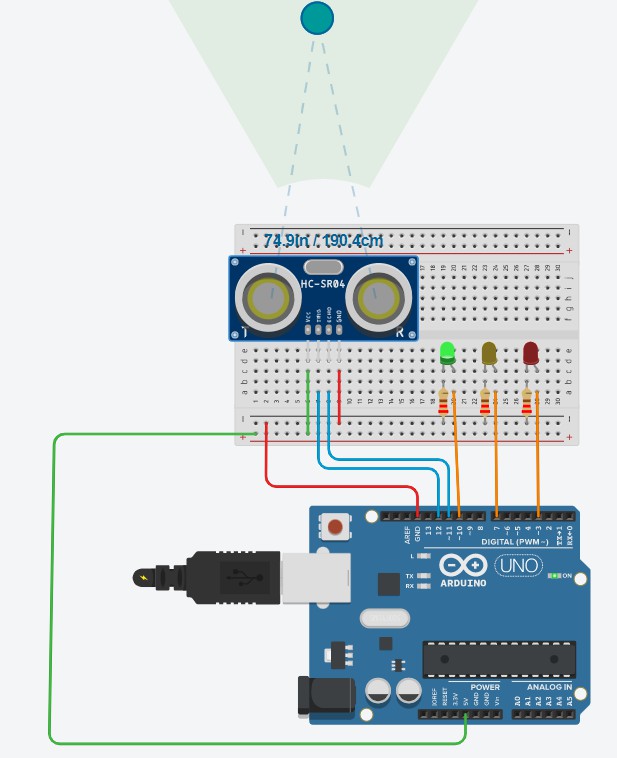
digitalWrite(10, LOW); digitalWrite(7, LOW); digitalWrite(3, HIGH);

}

delay(100);

}

### Screenshot:



**Result:**

An ultrasonic sensor has been interfaced with an Arduino and the distance of an object from the sensor has been measured. LEDs have also been interfaced with the Arduino and made to turn on based on the distance measured.

|  |  |
| --- | --- |
| **Exp. No. 6** | **INTERFACING AN ULTRASONIC SENSOR AND AN**  **LCD WITH AN ARDUINO** |
| **Date:** |

### Aim:

To interface an ultrasonic sensor with an Arduino to measure the distance of an object

from the sensor and display the measured distance on an LCD.

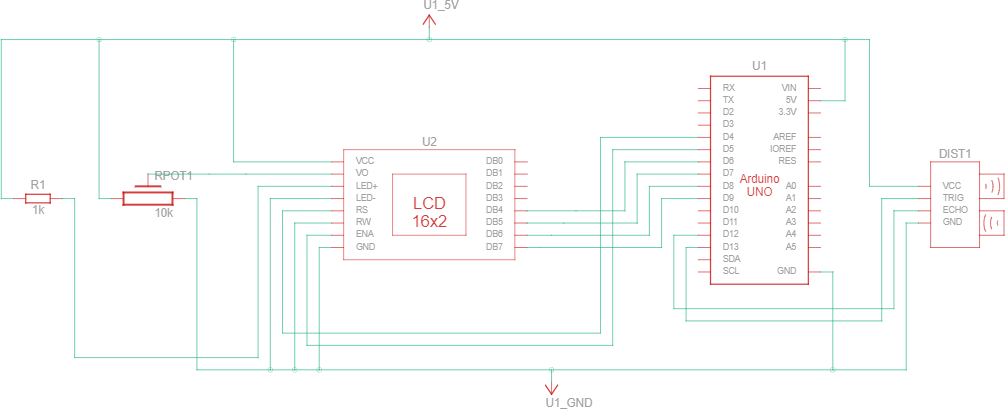
### Software required:

Tinkercad

### Procedure:

1. Open Tinkercad and click on “Create new Circuit”.
2. In the page that opens, choose the Arduino and the components in the circuit from the “Components” section and place them on the left side of the screen.
3. Wire the components as per the circuit diagram.
4. Click on “Code”, choose “Text” from the drop-down menu, and enter the code to program the Arduino.
5. Click on “Start Simulation” to simulate the circuit and verify the output.

### Circuit diagram:



**Program:**

#include <LiquidCrystal.h>

LiquidCrystal lcd(4,5,6,7,8,9); // rs, en, d4, d5, d6, d7

long duration; int distance;

void setup()

{

lcd.begin(16,2);

lcd.print("Distance:"); pinMode(13, OUTPUT); pinMode(12, INPUT); Serial.begin(9600);

}

void loop()

{

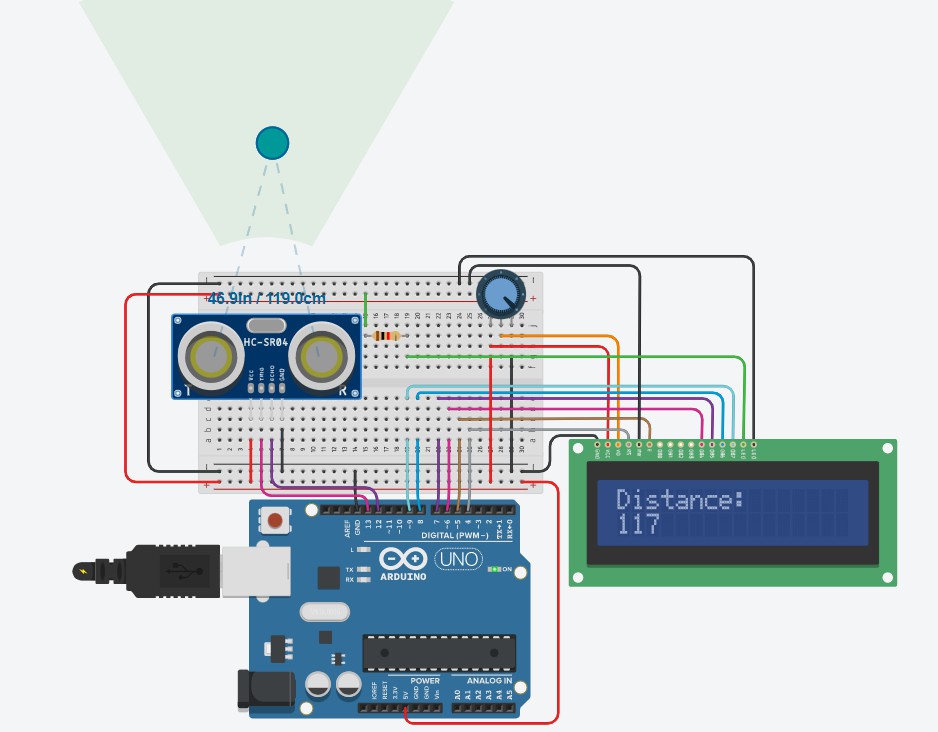
digitalWrite(13, LOW); delayMicroseconds(2); digitalWrite(13, HIGH); delayMicroseconds(10); digitalWrite(13, LOW);

duration = pulseIn(12, HIGH); distance = duration\*0.034/2;

lcd.setCursor(0,1); lcd.print(distance); delay(100);

}

### Screenshot:



**Result:**

An ultrasonic sensor has been interfaced with an Arduino and the distance of an object from the sensor has been measured and displayed on an LCD.

|  |  |
| --- | --- |
| **Exp. No. 7** | **INTERFACING A DC MOTOR WITH AN ARDUINO** |
| **Date:** |

### Aim:

To interface a DC motor with an Arduino and to turn it on/off using a push button.

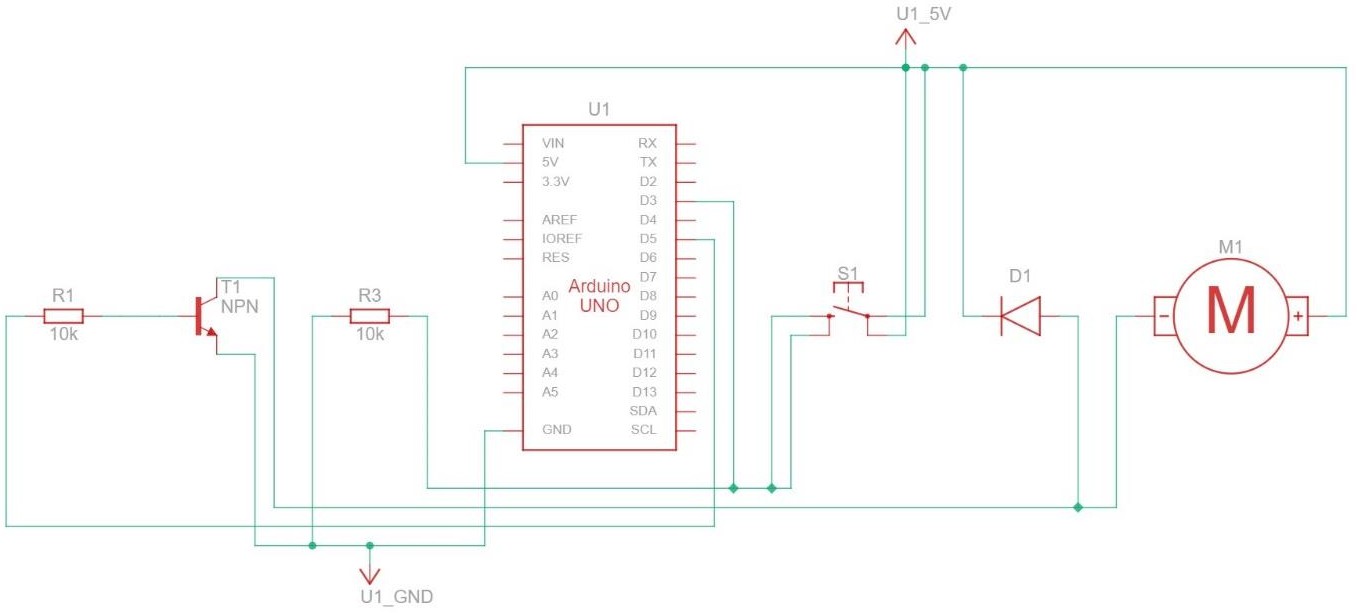
### Software required:

Tinkercad

### Procedure:

1. Open Tinkercad and click on “Create new Circuit”.
2. In the page that opens, choose the Arduino and the components in the circuit from the “Components” section and place them on the left side of the screen.
3. Wire the components as per the circuit diagram.
4. Click on “Code”, choose “Text” from the drop-down menu, and enter the code to program the Arduino.
5. Click on “Start Simulation” to simulate the circuit and verify the output.

### Circuit diagram:



**Program:** int button; void setup()

{

pinMode(5, OUTPUT);

pinMode(3, INPUT);

}

void loop()

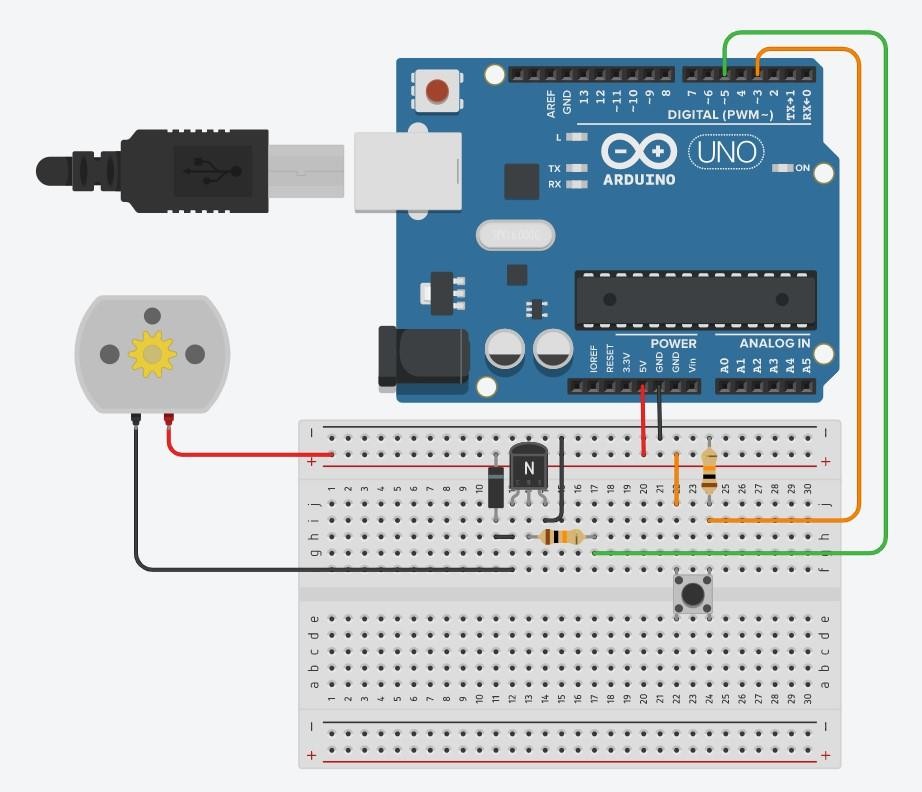
{

button = digitalRead(3); if(button == HIGH) digitalWrite(5, HIGH); else

digitalWrite(5, LOW);

}

### Screenshot:



**Result:**

A DC motor has been interfaced with an Arduino and a push button has been used to turn it on and off.

|  |  |
| --- | --- |
| **Exp. No. 8** | **DIRECTION CONTROL OF A DC MOTOR**  **USING AN ARDUINO** |
| **Date:** |

### Aim:

To interface a DC motor with an Arduino and to control its direction of rotation using a

push button.

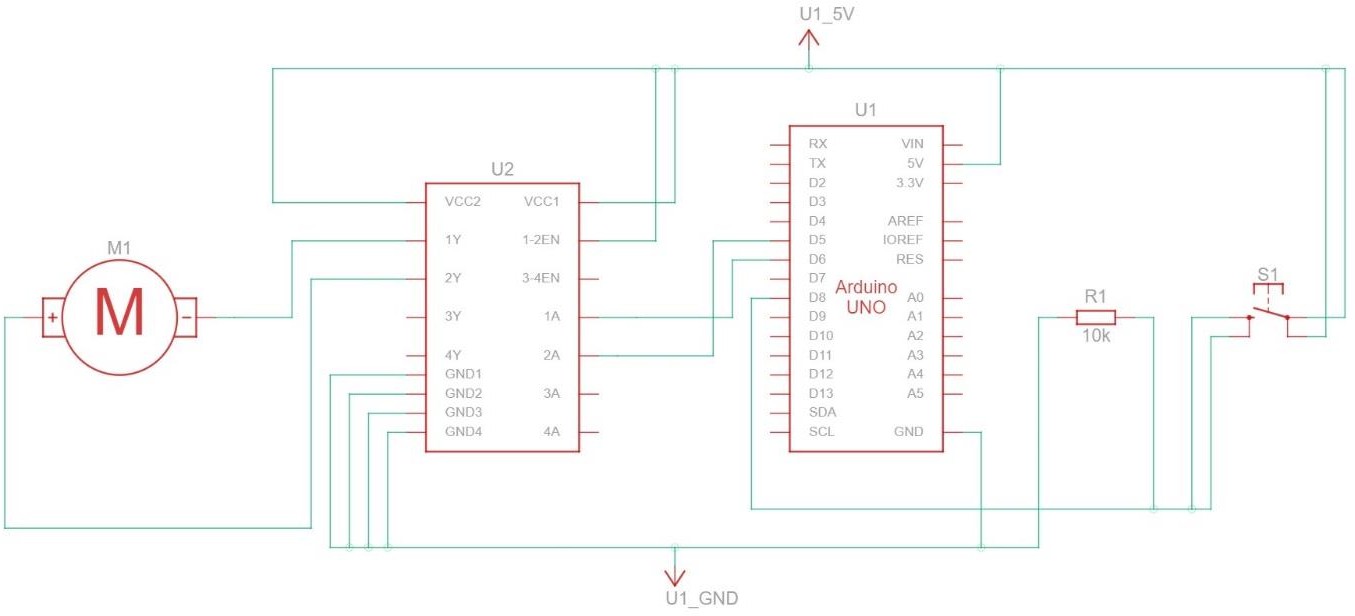
### Software required:

Tinkercad

### Procedure:

1. Open Tinkercad and click on “Create new Circuit”.
2. In the page that opens, choose the Arduino and the components in the circuit from the “Components” section and place them on the left side of the screen.
3. Wire the components as per the circuit diagram.
4. Click on “Code”, choose “Text” from the drop-down menu, and enter the code to program the Arduino.
5. Click on “Start Simulation” to simulate the circuit and verify the output.

### Circuit diagram:



**Program:**

int pb, pb\_old = 0, dir = 0; void setup()

{

pinMode(5, OUTPUT); pinMode(6, OUTPUT);

pinMode(8, INPUT);

}

void loop()

{

pb = digitalRead(8);

if (pb == HIGH and pb\_old == LOW) dir = 1 - dir;

if(dir == 1)

{

digitalWrite(5, HIGH); digitalWrite(6, LOW);

}

else

{

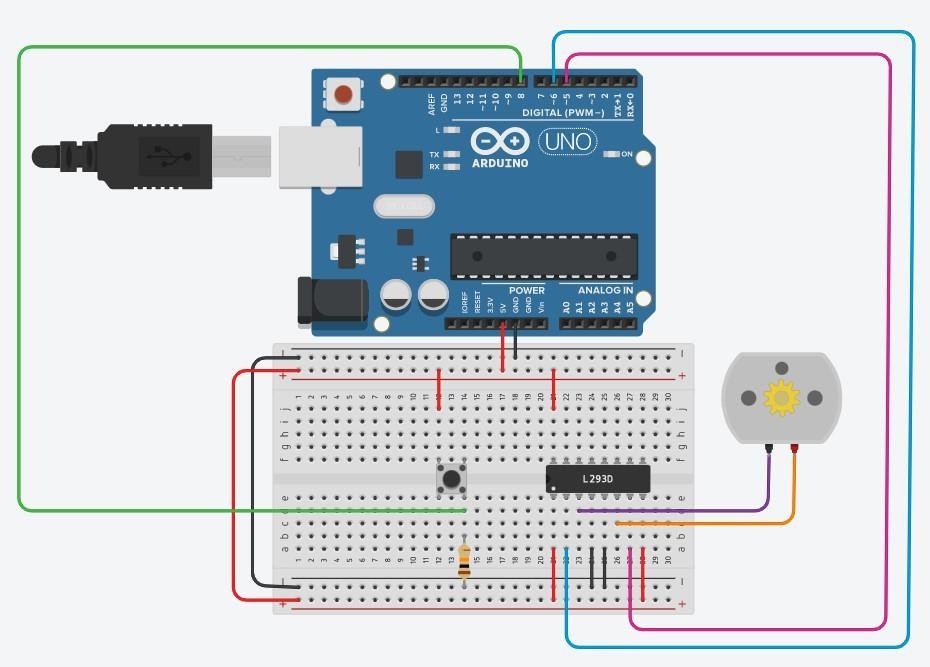
digitalWrite(6, HIGH); digitalWrite(5, LOW);

}

pb\_old = pb;

}

### Screenshot:



**Result:**

A DC motor has been interfaced with an Arduino and a push button has been used to change the direction of spin.

|  |  |
| --- | --- |
| **Exp. No. 8** | **SPEED AND DIRECTION CONTROL OF A DC MOTOR USING AN ARDUINO** |
| **Date:** |

### Aim:

To interface a DC motor with an Arduino and to control its speed and direction of

rotation using a potentiometer and a push button respectively.

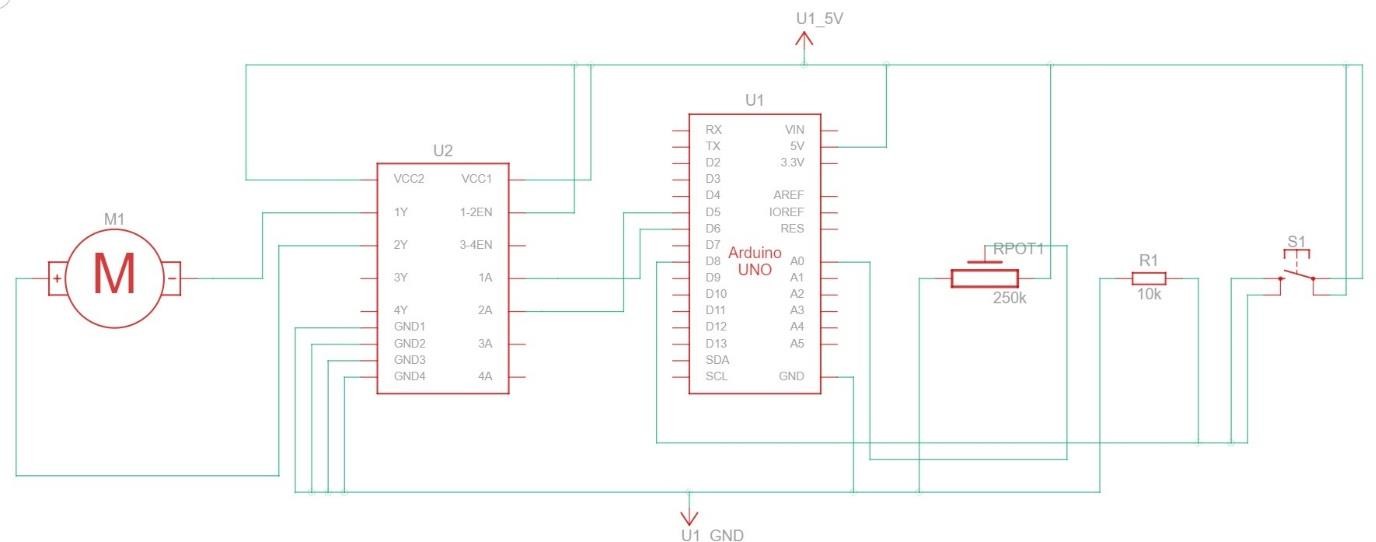
### Software required:

Tinkercad

### Procedure:

1. Open Tinkercad and click on “Create new Circuit”.
2. In the page that opens, choose the Arduino and the components in the circuit from the “Components” section and place them on the left side of the screen.
3. Wire the components as per the circuit diagram.
4. Click on “Code”, choose “Text” from the drop-down menu, and enter the code to program the Arduino.
5. Click on “Start Simulation” to simulate the circuit and verify the output.

### Circuit diagram:



**Program:**

int p, pwm, pb, pb\_old = 0, dir = 0; void setup()

{

pinMode(5, OUTPUT); pinMode(6, OUTPUT); pinMode(8, INPUT); pinMode(A0, INPUT);

}

void loop()

{

p = analogRead(A0);

pwm = map(p, 0, 1023, 0, 255);

pb = digitalRead(8);

if(pb == HIGH && pb\_old == LOW) dir = 1-dir;

if(dir == 1)

{

analogWrite(5, pwm); digitalWrite(6, LOW);

}

else

{

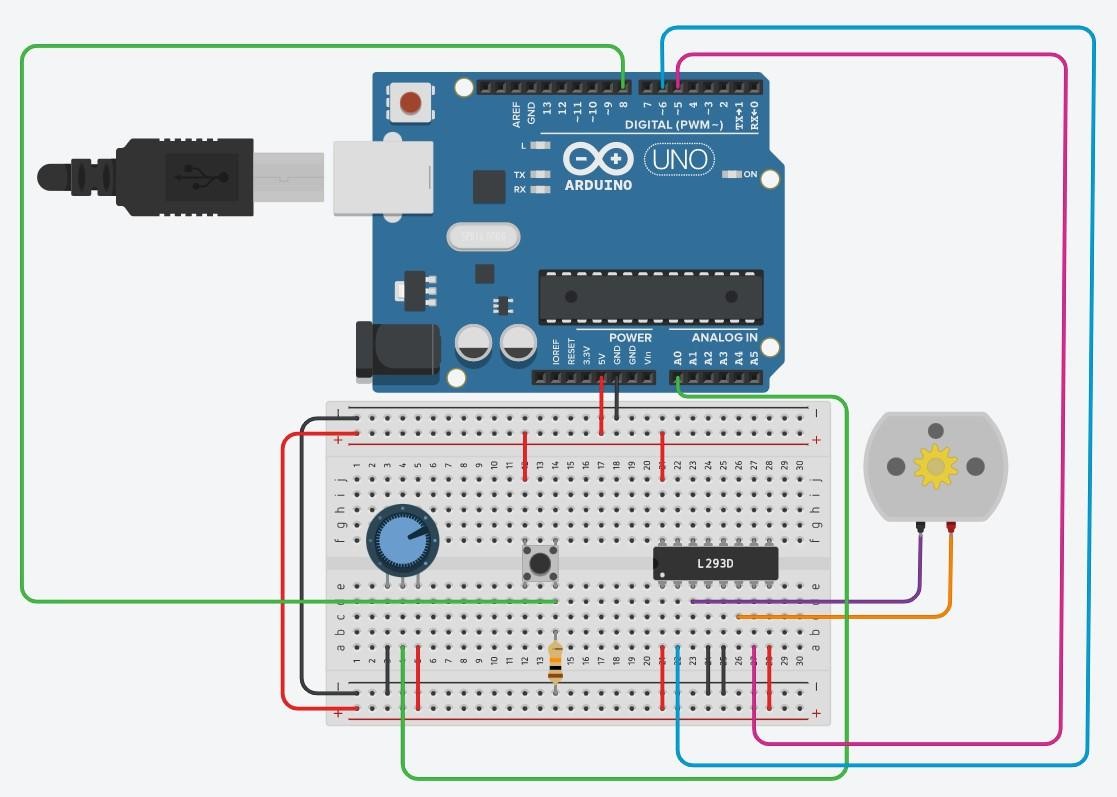
analogWrite(6, pwm); digitalWrite(5, LOW);

}

pb\_old = pb;

}

### Screenshot:



**Result:**

A DC motor has been interfaced with an Arduino, a push button has been used to change the spin direction, and a potentiometer has been used for speed control.

# Ex. No. 10

**Familiarization with Raspberry Pi/ Node MCU - To interface DHT11 ( basic temperature – humidity) sensor with RaspberryPi**

## AIM

Familiarization with Arduino/Raspberry Pi/ Node MCUand perform necessary software installation.

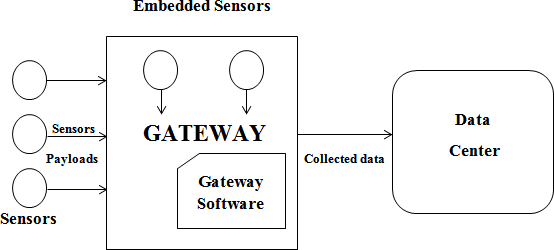
## THEORY

**INTERNET OF THINGS**

The Internet of Things (IoT) is generally connecting things to the Internetand using that connection to provide some kind of using remote monitoring orcontrol of things. It is basically a Machine-to-Machine communication throughwhich the electrical appliances are communicated. However, the devices arestill primarily things on the Internet that require more human interaction andmonitoring through apps and interfaces. The three basic platform of IoT arethe things, gateway and cloud. The thing in IoT can be a person or an animal orany other non-living object that has a sensor.

## ARCHITECTURE

The basic IoT architecture comprises of applications such as webpage or mobile devices, data processing platform, edge and things or sources. Edge is used to maintain a robust connection between the cloud and devices. In the cloud, data processing such as analysis and storage of data takes places in the IoT platform. The application layer takes care of the broadcasting of data andanalyze them. Building on top of the Internet of Things, the web of things isarchitecture for the application layer of the Internet of Things looking at theconvergence of data from IoT devices into Web applications to createinnovative use-cases.



## NETWORK ARCHITECTURE

The hardware has an important component, the sensors which are basically a transducer that used to get the output in digitized form from the given electrical or optical input. The gateway is the way through which the sensor data enters into the cloud environment. Device manager is responsible for the reallocation of sensor data. The data management unit takes care of collection and storage and analysis of sensor data. It also creates the data stream and expose to the HTTP link APIs.

The API is nothing but the application program which is responsible for creating the applications that can access data from OS. The IoT cloud is another important design that should be considered in the IoT design. The Internet of Things requires scalability to handle the surge of devices. It is the virtual area where all the sensor data and information are stored. The computing resources are shared instead of having a local server or personal devices to handle an application. It uses cloud computing technology.

## APPLICATIONS OF IOT

Smart City Technology can be applied in Energy Production and Energy Efficiency (Smart Energy and Smart Lighting)

* It can be applied in Smart water and waste management (smart utility)
* On-line information about consumption for future analysis
* Smart Parking helps in reduce traffic and manpower
* Smart water technology controls the leakage of water

## NETWORK PROTOCOLS

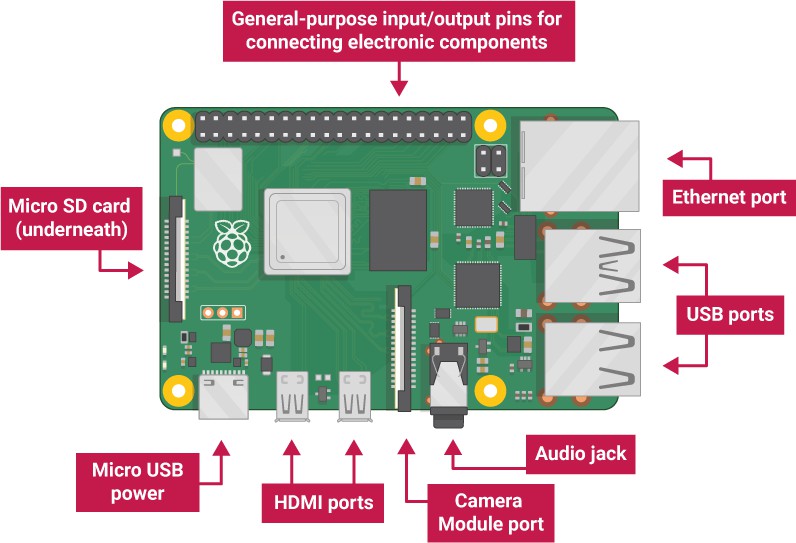
* Bluetooth
* Wi-Fi
* Zigbee
* LoRa

**INTRODUCTION TO Raspberry Pi / Arduino / Node MCU**

To develop an IoT application or product, many components are required. Raspberry Pi, Arduino and Node MCU are some of the basic building blocks. In this session we will discuss about these basic building blocks.

# Raspberry Pi

The Raspberry Pi is a small computer that can do lots of things. User can plug it into a monitor and attach a keyboard and mouse.

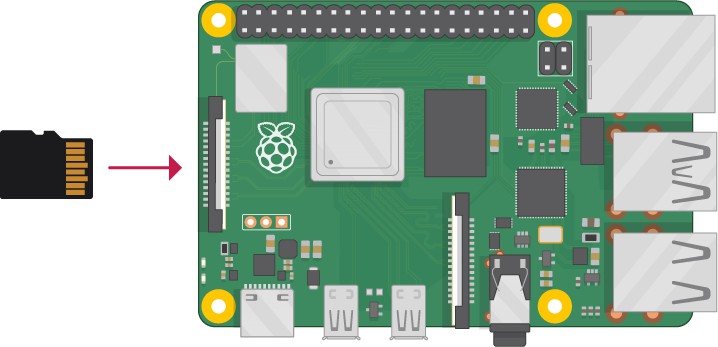


* USB ports — these are used to connect a mouse and keyboard. You can also connect other components, such as a USB drive.
* SD card slot — you can slot the SD card in here. This is where the operating system software and your files are stored.
* Ethernet port — this is used to connect Raspberry Pi to a network with a cable. Raspberry Pi can also connect to a network via wireless LAN.
* Audio jack — you can connect headphones or speakers here.
* HDMI port — this is where you connect the monitor (or projector) that you are using to display the output from the Raspberry Pi. If your monitor has speakers, you can also use them to hear sound.
* Micro USB power connector — this is where you connect a power supply. You should always do this last, after you have connected all your other components.
* GPIO ports — these allow you to connect electronic components such as LEDs and buttons to Raspberry Pi.

# Connect your Raspberry Pi

Let’s connect up your Raspberry Pi and get it running.

* Check the slot on the underside of your Raspberry Pi to see whether an SD card is inside. If no SD card is there, then insert an SD card with Raspbian installed (via NOOBS).

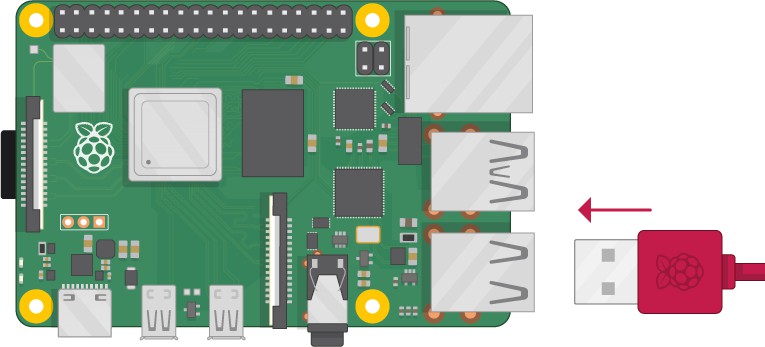


Note: Many microSD cards come inside a larger adapter — you can slide the smaller card out using the lip at the bottom.

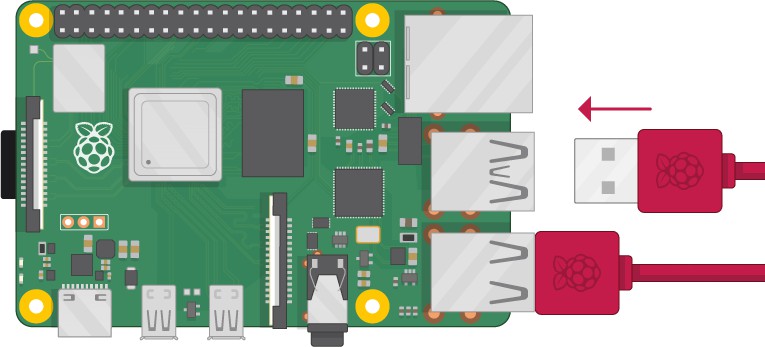


Installing Raspbian with NOOBS

* Find the USB connector end of your mouse’s cable, and connect the mouse to a USB port on your Raspberry Pi (it doesn’t matter which port you use).



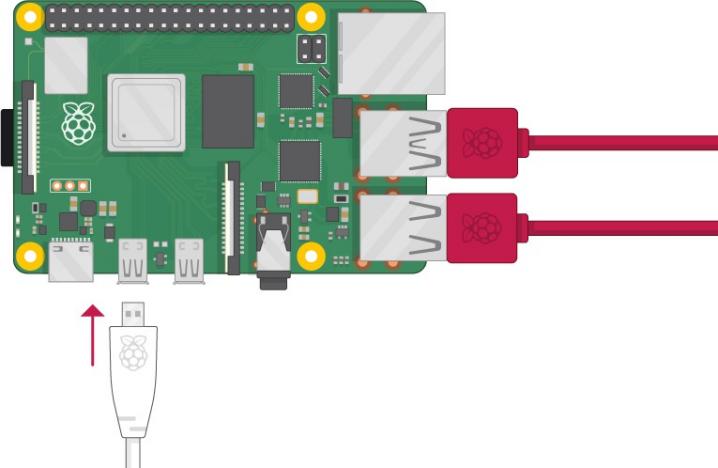
* Connect the keyboard in the same way.



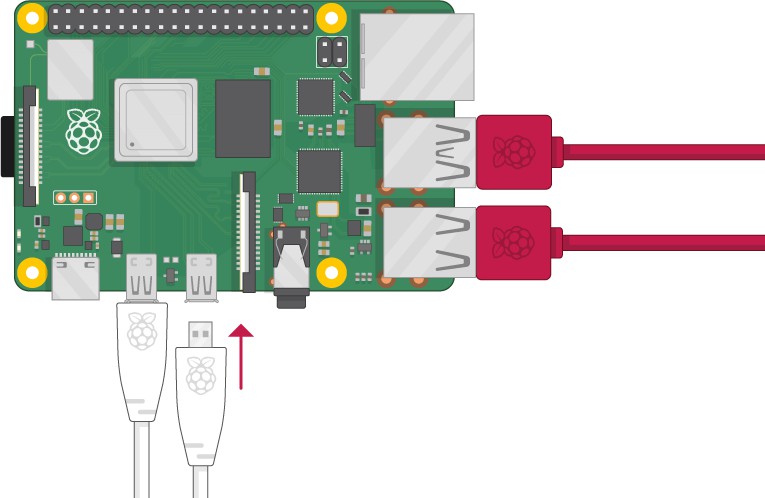
* Make sure your screen is plugged into a wall socket and switched on.
* Look at the HDMI port(s) on your Raspberry Pi — notice that they have a flat side on top.
* Use a cable to connect the screen to the Raspberry Pi’s HDMI port — use an adapter if necessary.

Raspberry Pi 4

Connect your screen to the first of Raspberry Pi 4’s HDMI ports, labelled HDMI0.

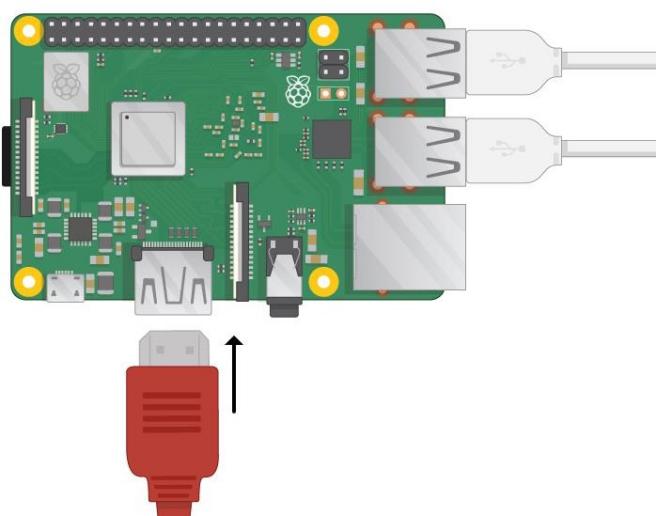


You could connect an optional second screen in the same way.



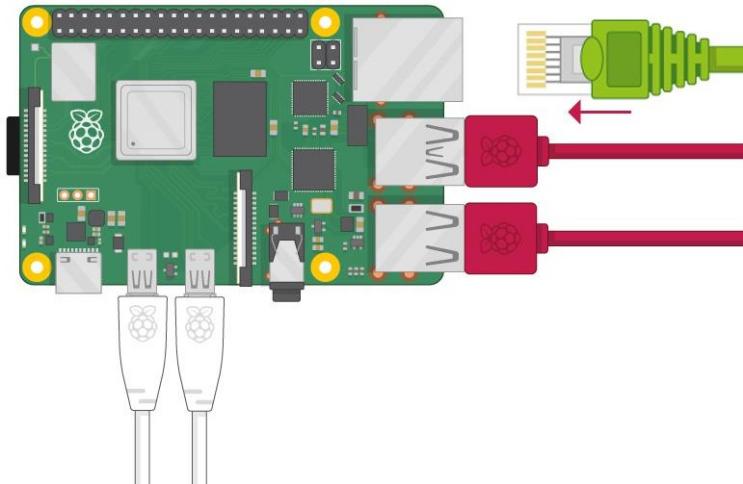
Raspberry Pi 1, 2, 3

Connect your screen to the single HDMI port.

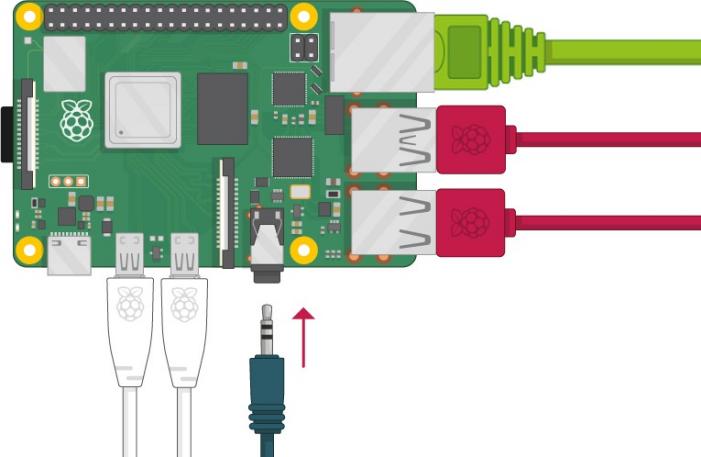


Note: nothing will display on the screen, because the Raspberry Pi is not running yet.

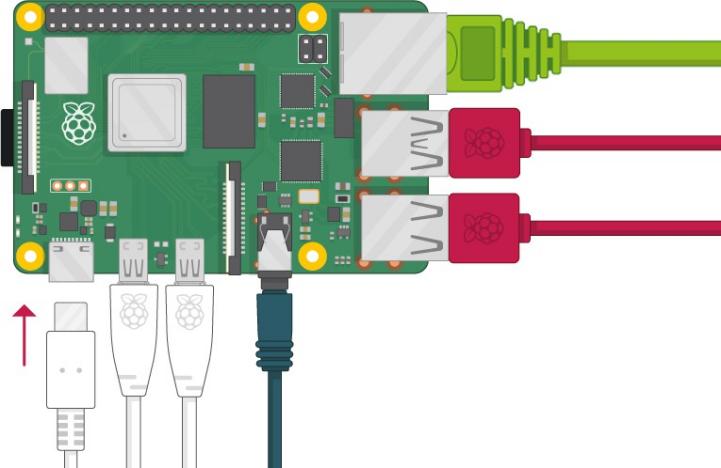
* If you want to connect the Pi to the internet via Ethernet, use an Ethernet cable to connect the Ethernet port on the Raspberry Pi to an Ethernet socket on the wall or on your internet router. You don’t need to do this if you want to use wireless connectivity, or if you don’t want to connect to the internet.



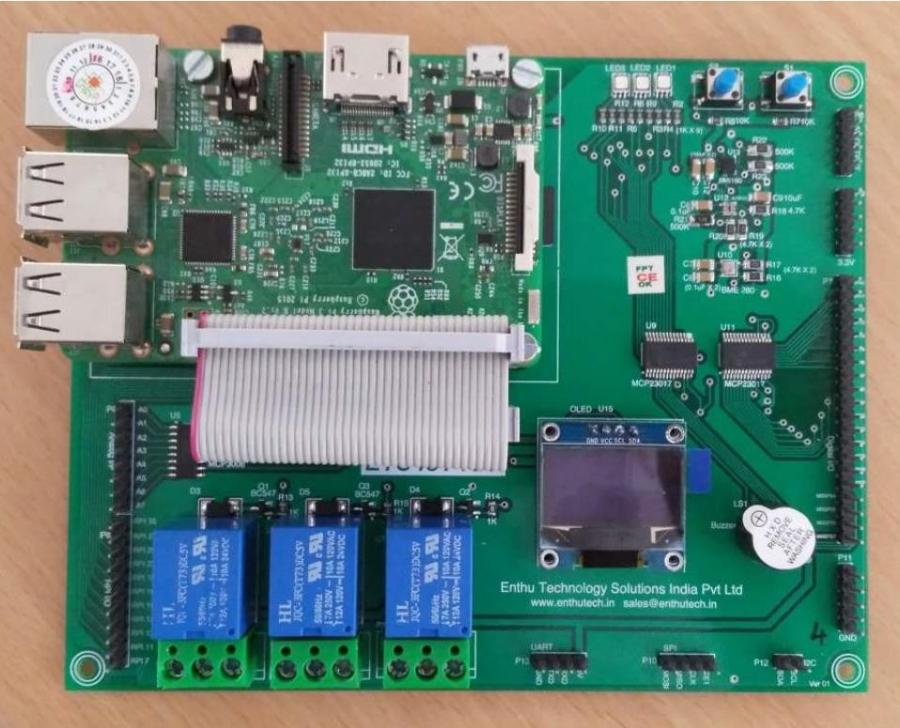
* If your screen has speakers, your Raspberry Pi can play sound through these. Or you could connect headphones or speakers to the audio port.



* Plug the power supply into a socket and then connect it to your Raspberry Pi’s USB power port.



You should see a red light on your Raspberry Pi and raspberries on the monitor. Your Raspberry Pi then boots up into a graphical desktop.



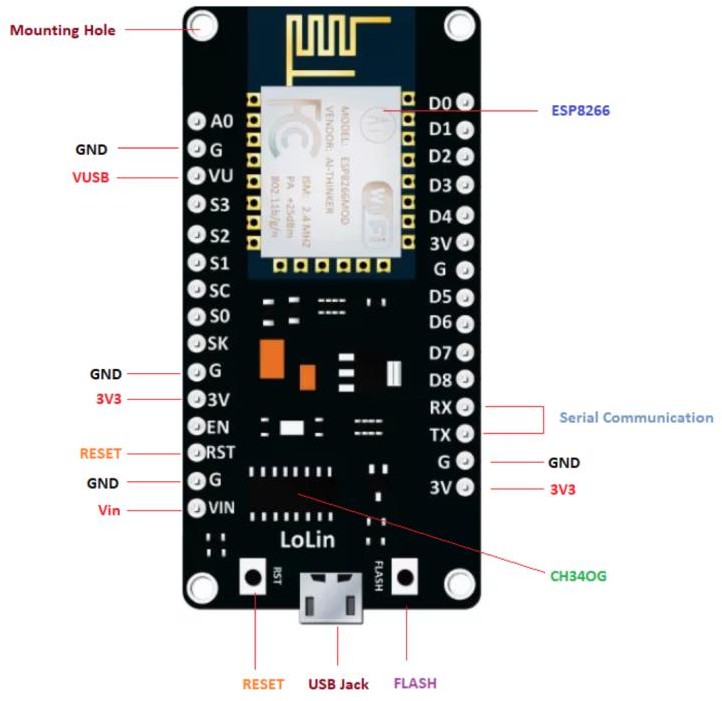
Top view of ETS IoT KIT

**Software:**

## NODE MCU

NodeMCU is an open source [IoT](https://en.wikipedia.org/wiki/Internet_of_Things) platform.It includes [firmware](https://en.wikipedia.org/wiki/Firmware) which runs on the [ESP8266](https://en.wikipedia.org/wiki/ESP8266) [Wi-](https://en.wikipedia.org/wiki/Wi-Fi) [Fi](https://en.wikipedia.org/wiki/Wi-Fi) [SoC](https://en.wikipedia.org/wiki/System_on_a_chip) from Espressif Systems, and hardware which is based on the ESP-12 module. The term "NodeMCU" by default refers to the firmware rather than the development kits. The firmware

uses the [Lua](https://en.wikipedia.org/wiki/Lua_(programming_language)) scripting language. It is based on the eLuaproject, and built on the Espressif Non- OS SDK for ESP8266. It uses many open source projects, such as lua-cjsonand [SPIFFS.](https://en.wikipedia.org/w/index.php?title=SPIFFS&action=edit&redlink=1)



Features of NODE MCU:

* Open-source
* Arduino-like hardware
* Status LED
* MicroUSB port
* Reset/Flash buttons
* Interactive and Programmable
* Low cost
* ESP8266 with inbuilt wifi
* USB to UART converter
* GPIO pins

# To interface DHT11 ( basic temperature – humidity) sensor with RaspberryPi

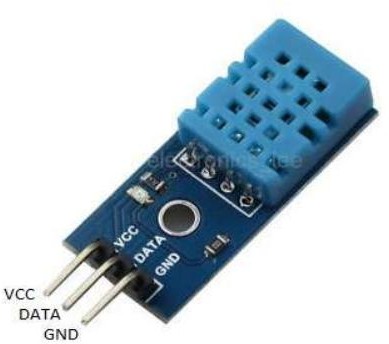
## AIM:

To connect a DHT11(Digital) sensor and print the Temperature and Humidity

## DESCRIPTION:

DHT11 SENSOR:

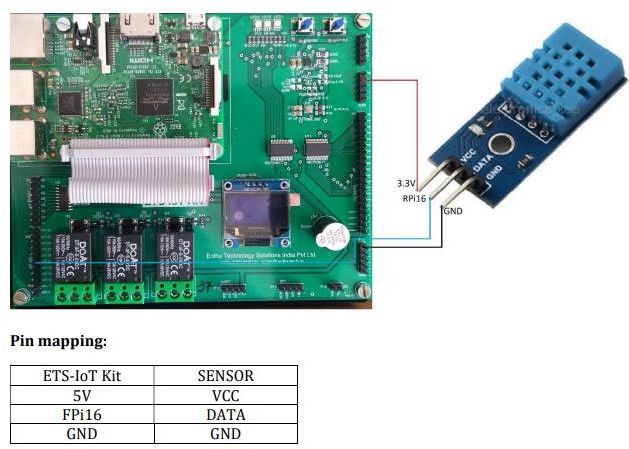
DHT11 digital temperature and humidity sensor is a composite Sensor contains a calibrated digital signal output of the temperature and humidity.



Applications of DHT11 Sensor:

* HVAC
* testing and inspection equipment
* consumer goods
* automotive
* automatic control
* data loggers
* weather stations
* home appliances

## CONNECTION DIAGRAM OF DHT11:



**SAMPLE CODE:**

# Application to connect a DHT11(Digital) sensor and print the Temperature and Humidity. import Adafruit\_DHT

sensor = Adafruit\_DHT.DHT11 pin = 23

humidity, temperature = Adafruit\_DHT.read\_retry(sensor, pin) while True:

if humidity is not None and temperature is not None:

print('Temp={0:0.1f}\*C Humidity={1:0.1f}%'.format(temperature, humidity)) else:

print('Failed to get reading. Try again!')

## RESULT:

DHT interface to Raspberry Pi has been completed. Also executed the program to print the current temperature and humidity

**SUBJECT FACULTY HOD SIGNATURE**