DR. D. Y. PATIL INSTITUTE OF ENGINEERING, MANAGEMENT AND RESEARCH, AKURDI, PUNE-44

Department of Electronics & Telecommunication Engineering

2022-2023

LAB MANUAL

Subject – Modernized IOT Lab

(Lab Practice-II)

Subject code: 404187

Class – BE

Institute Vision:

To strive for excellence by providing quality technical education & facilitate research for the welfare of society.

Institute Mission:

1. To educate students with strong fundamentals by providing conducive environment.
2. To inculcate research with creativity &innovation.
3. To promote industry-institute collaboration & prepare students for life long learning in context of technologicalchange.
4. To strengthen leadership, team-work, professional & communication skills and ethicalstandards

Department Vision:

To impart quality education to produce competent E&TC Engineers

Department Mission:

1. To equip students with strong basics through excellent blend of theory and practicalknowledge
2. To inculcate creativity and innovation through curricular and co- curricularactivities
3. To give the knowledge about all possible areas of E&TC by interacting with professionalworld
4. To develop the students with communication skills and ethical standards to meet the professionalneeds

Program Educational Objectives (PEOs):

1. The graduate shall utilize the basic knowledge to address the Engineeringproblems
2. The graduate shall attain the qualities of professional leadership with ethical and moralstandards
3. The graduate shall develop their capabilities for lifelong learning throughout their professional career and highereducation
4. The graduate shall explore engineering capabilities through creativity andinnovation

Program Specific Outcomes (PSOs):

1. Apply principles of Electronics and communication, digital systems, signal processing, software programming in the field of Embedded, Telecommunication & Software services for real worldapplications
2. Comprehend the technological advancements, demonstrate the proficiency in the usage of engineering tools to analyze and design systems for variety ofapplications.

3Demonstrate professional ethics, apply communication skills for successful career and higher studies.

Programme Outcomes: Engineering Graduates will be able to:

1. **Engineering Knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals and E&TC engineering specialization to the solution of complex E&TC engineeringproblems

   PO1.a – Apply the knowledge of mathematics PO1.b – Apply the knowledge of science

   PO1.c – Apply the knowledge of engineering fundamentals
2. **Problem Analysis:** Identify and analyze complex engineering problems using first principles of mathematics, natural sciences, and E&TC engineeringscience

   PO2.a – Identify the engineering problem PO2.b – analyze the engineering problem

   PO3.c – reaching the conclusion for theproblem
3. **Design /development of Solution:** Design solutions for E&TC engineering problems and design system components for reallife

   PO3.a –Design solution for engineeringproblems

   PO3.b – Design system components for real life solution
4. **Conduct investigations of complex problems:** Use Engineering knowledge for analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

   PO4.a – analysis of data PO4.b – Interpretation of data

   PO4.c – synthesis of data for valid conclusion
5. **Modern tool Usage:** select and apply appropriate techniques, using IT tools to model E&TC engineering problems with an understanding of thelimitations.

   PO5.a – Select and apply appropriate technique

   PO5.b – knowledge of various IT tools

   PO5.c – Use IT tools to model E&TC engineering problems
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 E&TC engineeringpractice.

   PO6.a – ability to identify the problem

   PO6.b – assess the problem

   PO6.c – apply the engineering solution
7. **Environment and sustainability:** Understand the impact of the E&TC engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainabledevelopment.

   PO7.a – understand the impact of E&TC engineering solutions

   PO7.b – demonstrate the knowledge for sustainabledevelopment
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the E&TC engineeringpractice.

   PO8.a – have awareness of ethical principles

   PO8.b - be committed to professional ethics
9. **Individual and team work:** Function effectively as an individual , and as a member or leader in ateam

   PO9.a – ability to function effectively as anindividual

   PO9.b – ability to function as a leader in ateam
10. **Communication:** Communicate effectively ,comprehend and write effective reports and make effectivepresentations

    PO10.a – ability to communicate effectively

    PO10.b – ability to comprehend and write effective reports

    PO10.c – ability to make effective presentations
11. **Project management and finance:** Have knowledge and understanding of the E&TC engineering and management principles and apply these to one's own work, as a member and leader in a team, to manageprojects

    PO11.a – Ability to have the knowledge and understanding of Engineering and Management principles

    PO11.b – apply managerial skills effectively as a leader

    PO11.c – Apply the E&TC engineering skills as a team member
12. **Life-long learning :**Ability of self-education and understand the technologicalchanges

    PO12.a –Inculcate the habit of self-learning and understanding

    PO12.b – ability to adapt to technological changes

University course Code	SAR course code	COURSE OUTCOMES	BT Level
404184	LO 707.3	Understand and Design IOT system using IOT platforms	BT4
	LO 707.4	Make use of IOT concepts to build small Project	BT6

Subjectcode:404187 Subject Name: Lab Practice-II(ModernizedIOT) Class:

BE Course Outcomes: Studentswill

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
LO 707.3	2	3	2	2	2			2	2			
LO 707.4	2	3	2	3	2			2	1			

CO-PO mapping

CO-PSO mapping

CO	PSO1	PSO2	PSO3
LO 707.3	1	1	
LO 707.4	2	2	

Savitribai Phule Pune University, Pune
B.E. (Electronics & Telecommunication) 2019 Course
(With effect from Academic Year 2022-23)

Semester-VII

Course Code	Course Name	Teaching Scheme (Hours/Week)			Examination Scheme and Marks						Credit			
		Theory	Practical	Tutorial	In-Sem	End-Sem	TW	PR	OR	Total	TH	PR	TUT	Total
404181	Radiation & Microwave Theory	03	-	-	30	70	-	-	-	100	03	-	-	03
404182	VLSI Design and Technology	03	-	-	30	70	-	-	-	100	03	-	-	03
404183	Cloud Computing	03	-	-	30	70	-	-	-	100	03	-	-	03
404184	Elective - 3	03	-	-	30	70	-	-	-	100	03	-	-	03
404185	Elective - 4	03	-	-	30	70	-	-	-	100	03	-	-	03
404186	Lab Practice - 1 (RMT & Cloud Computing)	-	04	-	-	-	25	-	50	75	-	02	-	02
404187	Lab Practice - 2 (VLSI Design & Elective -3)	-	04	-	-	-	25	50	-	75	-	02	-	02
404188	Project Stage - 1	-	02	-	-	-	50	-	-	50	-	01	-	01
404189	Mandatory Audit Course 7	-	-	-	-	-	-	-	-	-	-	-	-	-
	Total	**15**	**10**	**-**	**150**	**350**	**100**	**50**	**50**	**700**	-	-	-	-
	Total Credits										**15**	**05**	**-**	**20**

Elective - 3	Elective - 4
1. Speech Processing	1. Data Mining
2. PLC SCADA & Automation	2. Electronic Product Development
3. JAVA Script	3. Deep Learning
4. Embedded & RTOS	4. Low Power CMOS
5. Modernized IoT	5. Smart Antennas

Guidelines for Student's Lab Journal :The student's Lab Journal can be experimental write-ups. It should include following as applicable: Assignment No, Title of Assignment, Date of Performance, Date of Submission, Aims & Objectives, Theory, Description of data used, Results, Conclusion

Guidelines for Lab /TW Assessment :The oral examination will be based on the work carried out by the student in the Lab course. Suitable rubrics can be used by the internal & external examiner for assessment.

List of Experiments

1. Study of Raspberry-Pi, Beagle board, Arduino, and different operating systemsfor Raspberry Pi/Beagle board/Arduino. Understanding the process of OS installationon Raspberry Pi/Beagleboard/Arduino

2. Open-source prototype platform- Raspberry-Pi/Beagle board/Arduino -Simpleprogram digital read/write using LED and Switch -Analog read/write using sensor andactuators.

3. Interfacing sensors and actuators withArduino/Raspberry-pi.

4. IoT based Stepper Motor/DC Motor Control with Arduino/RaspberryPi.

5. Introduction to MQTT/ CoAP and sending sensor data to cloud usingRaspberry- Pi/Beagleboard/Arduino.

6. Get the status of a bulb at a remote place (on the LAN) throughweb.

7. Interfacing Arduino to BluetoothModule

8. Communicate between Arduino and Raspberry PI using any wireless medium likeZigBee

9. IoT based small project implementation on the topics based on small problem statements of the fields like chat bot, smart home (Home Automation), social issues andenvironmental issues etc. This project can be built on any IoT simulation platform like Tinkercad, Cooja etc

Experiment No. - 1

Title: Study of Raspberry-Pi, Beagle board, Arduino, and different operating systems for RaspberryPi/Beagle board/Arduino. Understanding the process of OS installation on Raspberry-Pi/Beagle board/Arduino

Date of Performance:

Date of Submission:

Aim: Study Various IOT platforms and installation Process.

Objectives: Study of Raspberry-Pi, Beagle board, Arduino, and different operating systems for RaspberryPi/Beagle board/Arduino. Understanding the process of OS installation

Hardware Requirements: Arduino Board, Raspberry Pi Board

Software Requirements: Arduino IDE, Noobs OS

Theory: Arduino:

An Arduino board consists of an Atmel 8-bit AVR microcontroller with complementary components that facilitate programming and incorporation into other circuits. It's an open-source physical computing platform based on a simple microcontroller board, and a development environment for writing software for theboard.

Fig.1 Arduino UNO

Specification of Arduino Uno:

Microcontroller	:ATmega328P
OperatingVoltage	:5V
InputVoltage(recommended)	:7-12V
InputVoltage(limit)	:6-20V
DigitalI/OPins	:14 (of which 6 provide PWMoutput)
AnalogInputPins	6

DC Current perI/OPin	:20mA

DC Current for3.3VPin	:50mA
FlashMemory	:32 KB (ATmega328P) of which 0.5 KB usedbybootloader
SRAM	:2 KB(ATmega328P)
EEPROM	:1 KB(ATmega328P)
ClockSpeed	:16MHz
LED_BUILTIN	13

Raspberry Pi :

The Raspberry Pi is a series of small single-board computers developed in the United Kingdom by the Raspberry Pi Foundation to promote the teaching of basic computer science in schools and in developingcountries.

The original model became far more popular than anticipated, selling outside of its target market for uses such as robotics.

Over 5 million Raspberry Pis have been sold before February 2015, making it the best-selling British computer. By November 2016 they had sold 11 million units.

Fig.2 Raspberry Pi 3 B+

Features:

All models feature a Broadcom system on a chip (SoC), which includes an ARM compatible central processing unit (CPU) and an on-chip graphics processing unit (GPU, a VideoCoreIV).

CPU speed ranges from 700 MHz to 1.5 GHz for the Pi 4 and on board memory rangefrom 256 MB to 4 GBRAM.

Secure Digital (SD) cards are used to store the operating system and program memoryin either the SDHC or Micro SDHCsizes.

Most boards have between one and four USB slots, HDMI and composite videooutput, and a 3.5 mm phone jack foraudio.

Lower level output is provided by a number of GPIO pins which support commonprotocols likeI²C.

TheB-modelshavean8P8CEthernetportandthePi3andPiZeroWhaveon

boardWi-Fi 802.11n and Bluetooth.

Procedure to install the Operating System:

Install the SD Formatter software inComputer/Laptop

Insert the micro SD card into the card reader and connect itto Computer/Laptop

Select correct drive & format the SDcard

Operating Systems:

The Foundation provides Raspbian, a Debianbased Linux distribution for download, as well as third party Ubuntu, Windows 10 IOT Core, RISC OS, and specialisedmedia centerdistributions.

It promotes Python and Scratch as the main programming language, with support for many otherlanguages.

The default firmware is closed source, whilean unofficial open source is available.

Fig.3 Symbol of Operating

Peripheral

Fig.4 Peripherals required for assembling

Download the OS from official web site of raspberry pihttps://www.raspberrypi.org/downloads/noobs/

Download zipfile

Extract the zip file into SDcard

Insert the micro SD card into the slot on the underside of thePi

Plug in keyboard andmouse

Plug in monitor using the HDMIport

The Raspberry Pi doesn't have a power button. It boots up as soon as you plug in the power supply

If you've completed all the previous steps, plug in the power supply to boot the RaspberryPi

Fig.5 Setup of the Raspberry Pi

Conclusion:

Experiment No. – 2

Title: Open-source prototype platform- Raspberry-Pi/Beagle board/Arduino -Simple program digital read/write using LED and Switch -Analog read/write using sensor and actuators.

Date of Performance:

Date of Submission:

Aim: Understand the connection and configuration of GPIO and its use in programming. Write an application of the use of push switch, LEDs,Sensor.

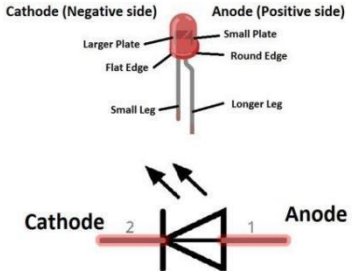
Objectives: Write Simple program digital read/write using LED and Switch -Analog read/write using sensor andactuators.

Hardware Requirements: Arduino Board, LED, Push Button, Raspberry Pi

Software Requirements: Arduino IDE

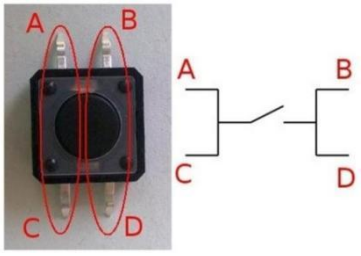
Theory:

LED:



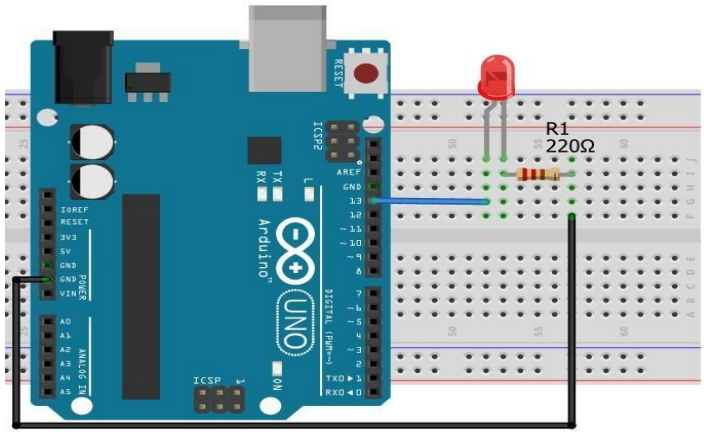
A light-emitting diode (LED) is a semiconductor light source that emits light when current flows through it. Electrons in the semiconductor recombine with electron holes, releasing energy in the form of photons. The color of the light (corresponding to the energy of the photons) is determined by the energy required for electrons to cross the band gap of the semiconductor. White light is obtained by using multiple semiconductors or a layer of light-emitting phosphor on the semiconductordevice

Push Button:

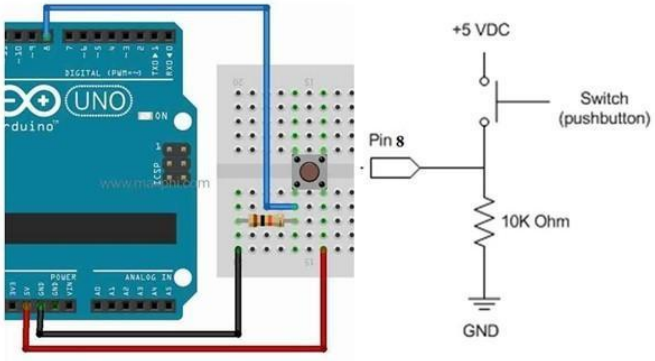


A push-button (also spelled pushbutton) or simply button is a simple switch mechanism to control some aspect of a machine or a process. Buttons are typically made out of hard

material, usually plastic or metal.[1] The surface is usually flat or shaped to accommodate the human finger or hand, so as to be easily depressed or pushed. Buttons are most often biased switches, although many un-biased buttons (due to their physical nature) still require a spring to return to their un-pushed state. Terms for the "pushing" of a button include pressing, depressing, mashing, slapping, hitting, andpunching.



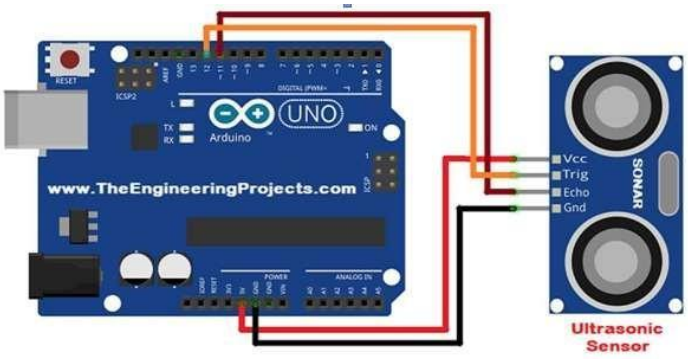
Interfacing of LED



Interfacing of Push Button

Ultrasonic Sensor:

An ultrasonic sensor is an electronic device that measures the distance of a target object by emitting ultrasonic sound waves, and converts the reflected sound into an electrical signal. Ultrasonic waves travel faster than the speed of audible .Ultrasonic sensors have two main components: the transmitter (which emits the sound using piezoelectric crystals) and the receiver (which encounters the sound after it has travelled to and from the target).In order to calculate the distance between the sensor and the object, the sensor measures the time it takes betweentheemissionofthesoundbythetransmittertoitscontactwiththereceiver.



Interfacing of Ultrasonic Sensor

Procedure:

Make the connection as per circuitdiagram

Open ArduinoIDE

Select the ArduinoBoard

Select the ArduinoPort

Write code in Editor Window and savefile

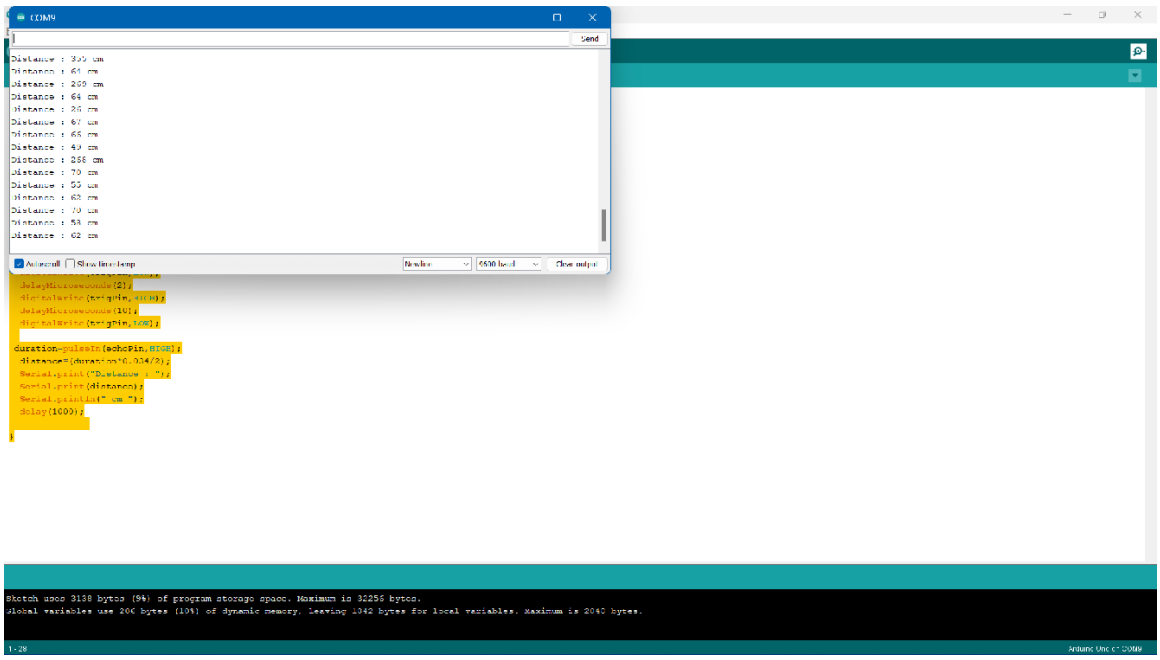
Compile the code

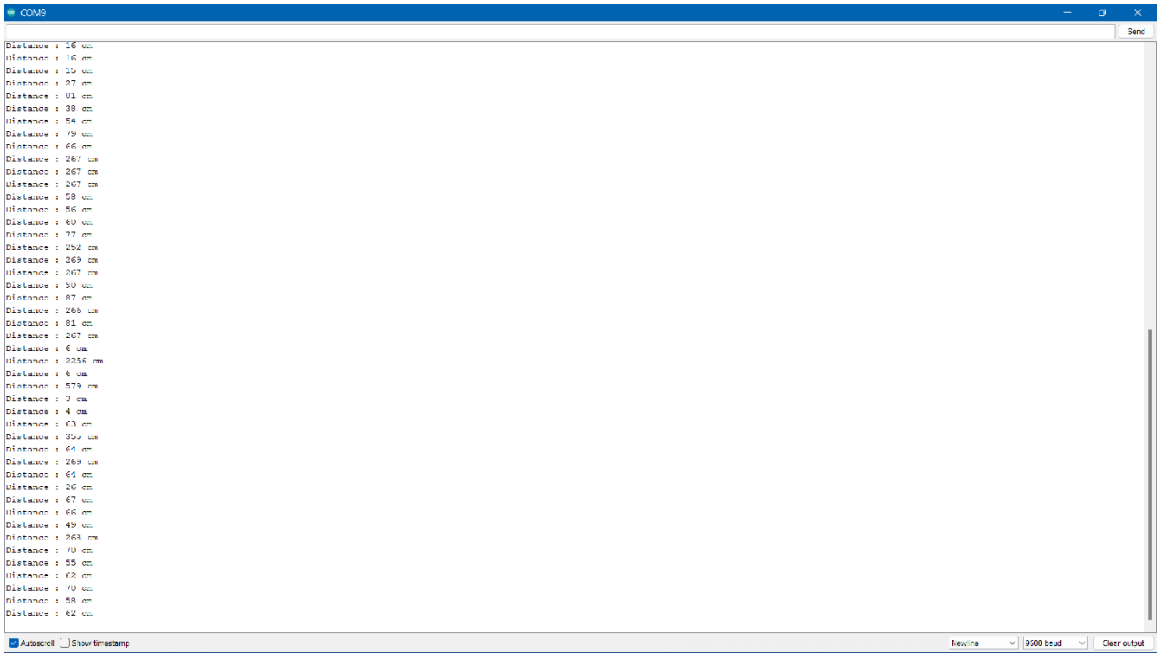
Upload thecode

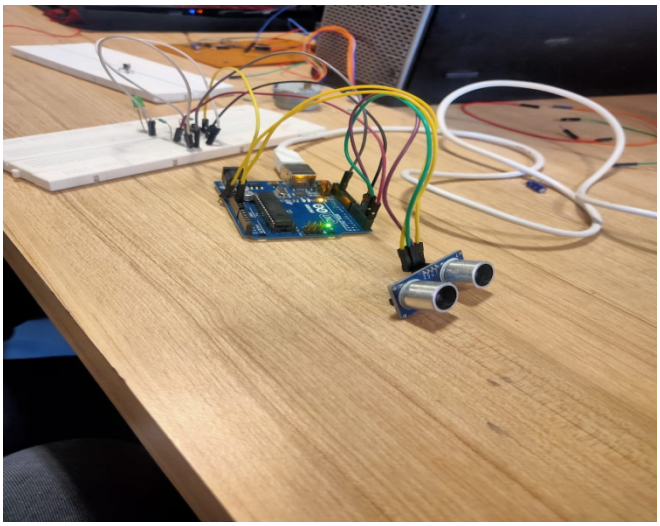
Code for Ultrasonic Sensor:

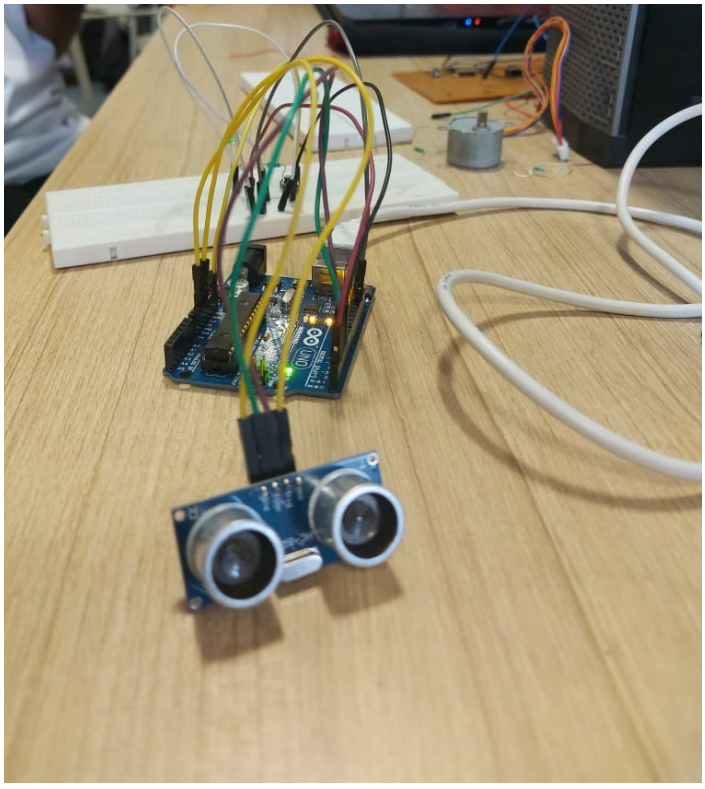
```
#define echoPin 2
#define trigPin 3


long duration;

int distance;

void setup(){

Serial.begin(9600);

pinMode(trigPin,OUTPUT);

pinMode(echoPin,INPUT);

}

void loop(){

digitalWrite(trigPin,LOW);

delayMicroseconds(2);

digitalWrite(trigPin,HIGH);

delayMicroseconds(10);

digitalWrite(trigPin,LOW);


duration=pulseIn(echoPin,HIGH);

distance=(duration*0.034/2);

Serial.print("Distance : ");

Serial.print(distance);

Serial.println(" cm ");

delay(1000);


}
```

Output-


COM9
Send
Distance : 365 cm
Distance : 64 cm
Distance : 269 cm
Distance : 64 cm
Distance : 26 cm
Distance : 67 cm
Distance : 66 cm
Distance : 49 cm
Distance : 258 cm
Distance : 70 cm
Distance : 55 cm
Distance : 62 cm
Distance : 70 cm
Distance : 58 cm
Distance : 62 cm
Autoscroll
Show timestamp
Newline
9600 baud
Clear output
Sketch uses 3138 bytes (9%) of program storage space. Maximum is 32256 bytes.
Global variables use 206 bytes (10%) of dynamic memory, leaving 1842 bytes for local variables. Maximum is 2048 bytes.

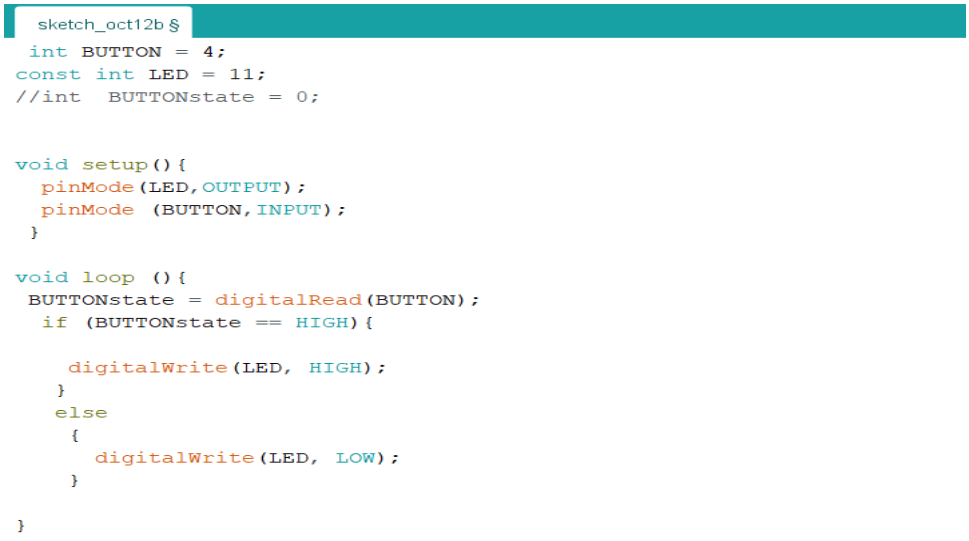

COM9
Send
Distance : 16 cm
Distance : 16 cm
Distance : 19 cm
Distance : 27 cm
Distance : 91 cm
Distance : 38 cm
Distance : 54 cm
Distance : 79 cm
Distance : 66 cm
Distance : 267 cm
Distance : 267 cm
Distance : 267 cm
Distance : 58 cm
Distance : 56 cm
Distance : 60 cm
Distance : 77 cm
Distance : 252 cm
Distance : 269 cm
Distance : 267 cm
Distance : 50 cm
Distance : 87 cm
Distance : 265 cm
Distance : 81 cm
Distance : 267 cm
Distance : 6 cm
Distance : 3256 cm
Distance : 6 cm
Distance : 579 cm
Distance : 9 cm
Distance : 4 cm
Distance : 63 cm
Distance : 365 cm
Distance : 64 cm
Distance : 269 cm
Distance : 64 cm
Distance : 26 cm
Distance : 67 cm
Distance : 66 cm
Distance : 49 cm
Distance : 263 cm
Distance : 70 cm
Distance : 55 cm
Distance : 62 cm
Distance : 70 cm
Distance : 58 cm
Distance : 62 cm
Autoscroll
Show timestamp
Newline
9600 baud
Clear output



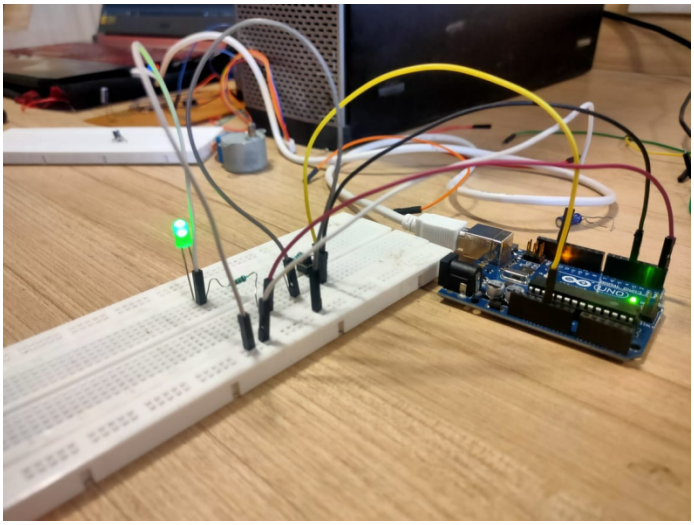


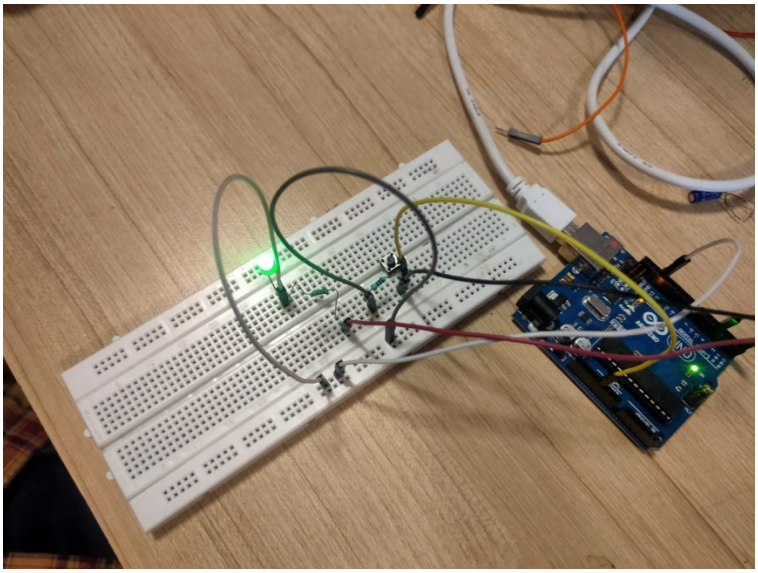
Code for LED blink:

```
int BUTTON = 4;
const int LED = 11;
//int  BUTTONstate = 0;


void setup(){
  pinMode(LED,OUTPUT);
  pinMode (BUTTON,INPUT);
 }

void loop (){
 BUTTONstate = digitalRead(BUTTON);
  if (BUTTONstate == HIGH){

    digitalWrite(LED, HIGH);
   }
   else
    {
      digitalWrite(LED, LOW);
    }

}
```



Output-





Conclusion:

Experiment No. – 3

Title: Interfacing sensors and actuators with Arduino/Raspberry-pi.

Date of Performance:

Date of Submission:

Aim: Interfacing of IR sensors and Servo Motor with Arduino.

Objectives: In this try to move the servo Motor if an object is detected by IR S nsor.

Hardware Requirements: Arduino Board, IR Sensor, Servo Motor, Connecting Wires

Software Requirements: Arduino IDE

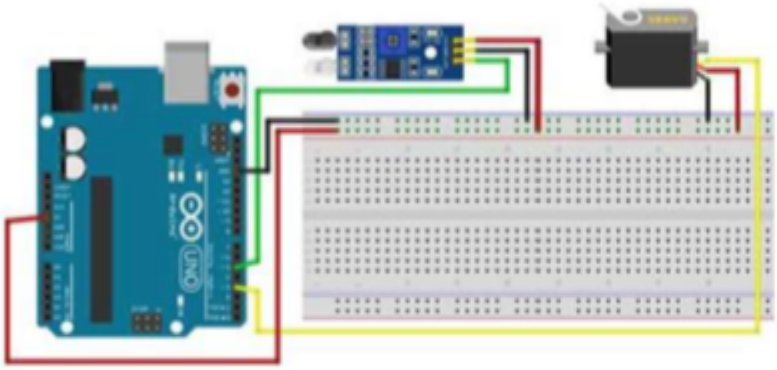
Theory:

IR Sensor:

IR sensor is an electronic device that emits the light in order to sense some object of the surroundings. An IR sensor can measure the heat of an object as well as detects the motion. Usually, in the infrared spectrum, all the objects radiate some form of thermal radiation. These types of radiations are invisible to our eyes, but infrared sensor can detect these radiations.

Servo Motor:

A servomotor (or servo motor) is a rotary actuator or linear actuator that allows for precise control of angular or linear position, velocity and acceleration. It consists of a suitable motor coupled to a sensor for position feedback. It also requires a relatively sophisticated controller, often a dedicated module designed specifically for use with servomotors.



Interfacing IR sensor and Servo Motor

Procedure:

Make the connection as per circuitdiagram

Open Arduino IDE

Select the ArduinoBoard

Select the ArduinoPort

Write code in Editor Window and savefile

Compile thecode

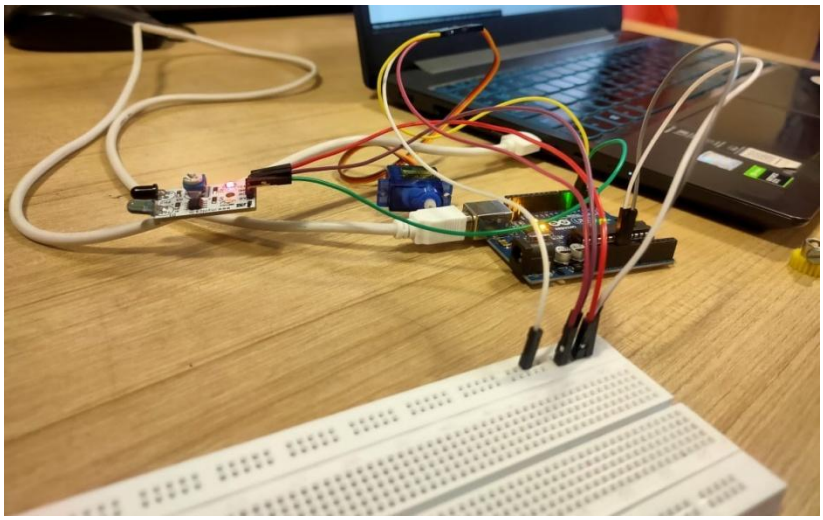
Upload thecode

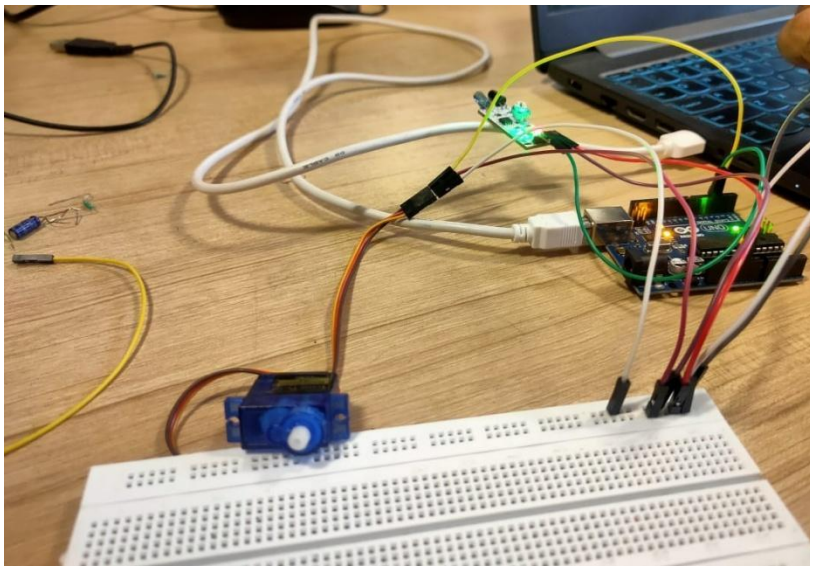
Code-

```
 //Servo motor
#include <Servo.h>
#include <Servo.h>
Servo tap_servo;
int sensor_pin = 4;
int tap_servo_pin =3;
int val;
void setup(){
  pinMode(sensor_pin,INPUT);
  tap_servo.attach(tap_servo_pin);
  Serial.begin(9600);
}
void loop(){
  val = digitalRead(sensor_pin);
  Serial.println(val);
  if (val==0)
  {tap_servo.write(0);
  }
  if (val==1)
```

```
  {tap_servo.write(180);
    }
}
```

Output-





Conclusion:

Experiment No. – 4

Title: IoT based Stepper Motor/DC Motor Control with Arduino/Raspberry Pi.

Date of Performance:

Date of Submission:

Aim: To interface Stepper Motor to Raspberry Pi Board and Rotate it in clockwise direction continuously.

Objectives: 1. To study Raspberry pi board and python language

2. To interface Stepper Motor to Raspberry Pi Board an Rotate it in clockwise direction continuously

Hardware Requirements: Raspberry pi board, Stepper Motor, Connecting Wires

Software Requirements: Raspbian OS

Theory:

Stepper Motor:

- Stepper motor is a brushless DC motor that divides the full rotation angle of 360° into number of equal steps.
- The motor is rotated by applying certain sequence of control signals. The speed of rotation can be changed by changing the rate at which the control signals are applied.
- Raspberry Pi's GPIOs can be used to control stepper motor rotation. We can generate sequence of control signals on the GPIO pins of Raspberry Pi.

Interfacing of Stepper Motor

Rotate a Stepper Motor in clockwise and counter-clockwise directions alternately.

🎬 we are using six wire unipolar stepper motor. Only four wires are required to control this stepper motor. The two center tap wires of the stepper motor are connected to 5V supply.

🎬 ULN2003 driver is used to drive unipolar stepper motor.

🎬 We will interface Stepper Motor with Raspberry Pi using Python language. In this program,

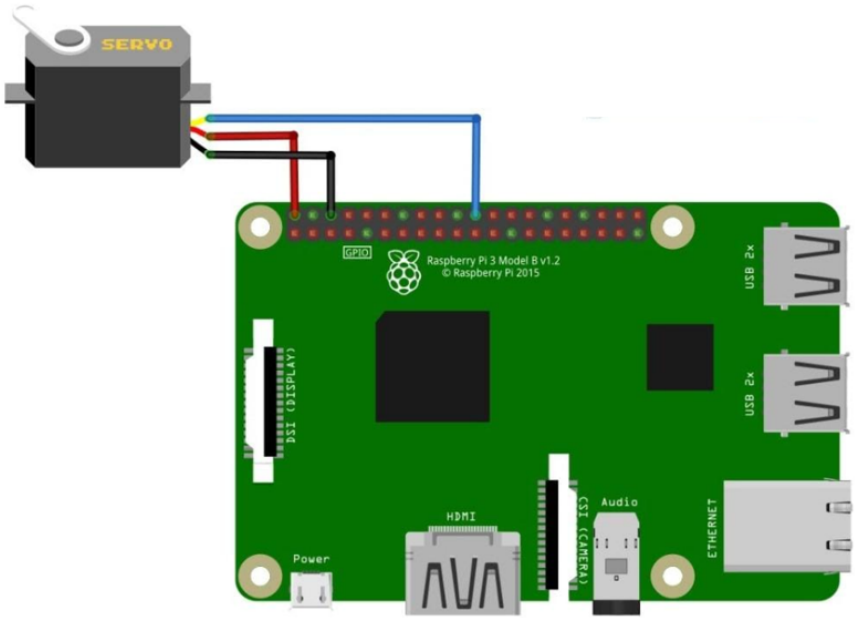
we have to use keyboard key as input for selecting motor rotation direction (i.e. clockwise or anti-clockwise).

Servo Motor:

A servo motor is a type of motor that can rotate with great precision. Normally this type of motor consists of a control circuit that provides feedback on the current position of the motor shaft, this feedback allows the servo motors to rotate with great precision. If you want to rotate an object at some specific angles or distance, then you use a servo motor. It is just made up of a simple motor which runs through a servo mechanism. If motor is powered by a DC power supply then it is called DC servo motor, and if it is AC-powered motor then it is called AC servo motor. For this tutorial, we will be discussing only about the DC servo motor working. Apart from these major classifications, there are many other types of servo motors based on the type of gear arrangement and operating characteristics. A servo motor usually comes with a gear arrangement that allows us to get a very high torque servo motor in small and lightweight packages. Due to these features, they are being used in many applications like toy car, RC helicopters and planes, Robotics, etc.

Servo motors are rated in kg/cm (kilogram per centimeter) most hobby servo motors are rated at 3kg/cm or 6kg/cm or 12kg/cm. This kg/cm tells you how much weight your servo motor can lift at a particular distance. For example: A 6kg/cm Servo motor should be able to lift 6kg if the load is suspended 1cm away from the motors shaft, the greater the distance the lesser the weight carrying capacity. The position of a servo motor is decided by electrical pulse and its circuitry is placed beside the motor.

Interfacing with Servo Motor



Interfacing a Servo Motor with Raspberry Pi is an interesting topic as Servo Motors are the main components of a Robot and with the help of Raspberry Pi

A Servo Motor is a simple device that consists of a DC Motor, Gears and a Feed Back based Position Control System. The main advantage of a Servo Motor is its ability to hold the angular position of its shaft.

There are several varieties of Servo Motors you can choose from. Two of the most common Servo Motors are Tower Pro SG90 and Tower Pro MG90S.

Procedure:

1. Do Connections as per Diagram

2. Write python code and save.

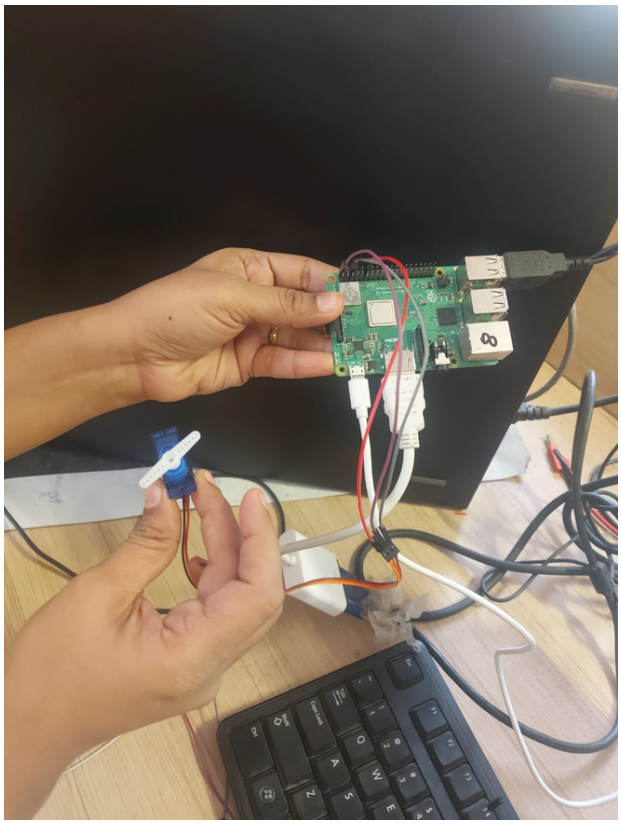
3. Run the code and observe the output**.**

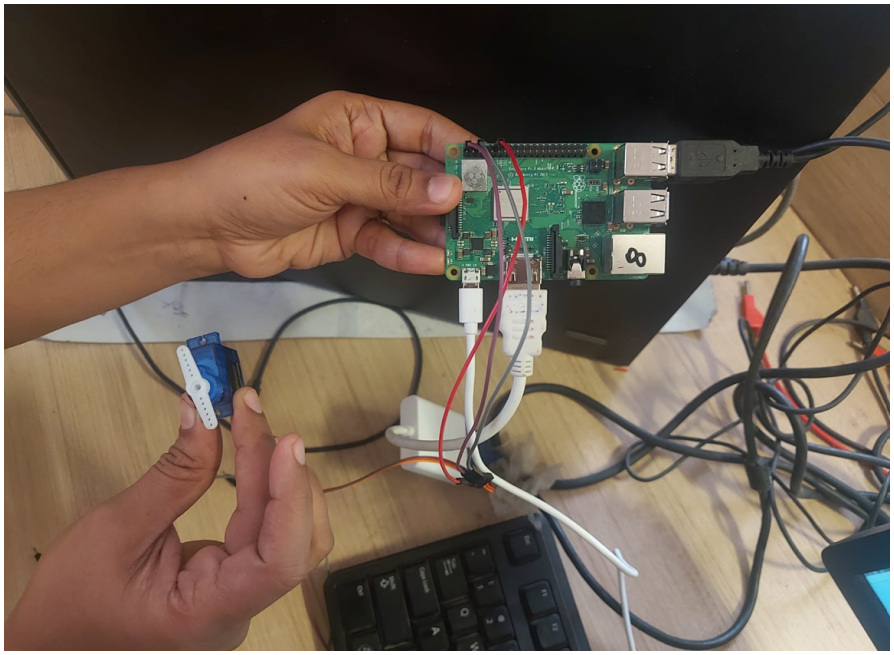
Program:

```
import RPi.GPIO as GPIO
import time

servoPIN = 17
GPIO.setmode(GPIO.BCM)
GPIO.setup(servoPIN, GPIO.OUT)

p = GPIO.PWM(servoPIN, 50) # GPIO 17 for PWM with 50Hz
p.start(2.5) # Initialization
try:
  while True:
    p.ChangeDutyCycle(5)
    time.sleep(0.5)
    p.ChangeDutyCycle(7.5)
    time.sleep(0.5)
    p.ChangeDutyCycle(10)
```

```
    time.sleep(0.5)
    p.ChangeDutyCycle(12.5)
    time.sleep(0.5)
    p.ChangeDutyCycle(10)
    time.sleep(0.5)
    p.ChangeDutyCycle(7.5)
    time.sleep(0.5)
    p.ChangeDutyCycle(5)
    time.sleep(0.5)
    p.ChangeDutyCycle(2.5)
    time.sleep(0.5)
except KeyboardInterrupt:
  p.stop()
  GPIO.cleanup()
```

Output:





Conclusion:

Experiment No. – 5

Title: Introduction to MQTT/ CoAP and sending sensor data to cloud using Raspberry-Pi/Beagle board/Arduino.

Aim: Interfacing of DHT11 (Humidity, Temperature) sensor with Raspberry Pi board and store the information on thingspeak cloud.

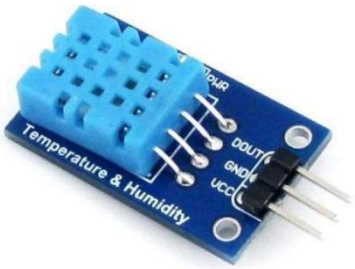
Objectives:
1. Interfacing of DHT11 with Espnode
2. Create a channel on the cloud(Thingspeak)
3. Upload Sensor data on Thingspeak

Hardware Requirements: Espnodei, DHT11 Sensor, connecting Wires

Software Requirements: Arduino IDE

Theory:

DHT11 Sensor:



The DHT11 is a basic, ultra low-cost digital temperature and humidity sensor. It uses a capacitive humidity sensor and a thermistorto measure the surroundingair,and spitsouta

digital signal on the data pin (no analog input pins needed). It's fairly simple to use, but requires careful timing to grab data. The only real downside of this sensor is you can only get new data from it once every 2 seconds, so when using library, sensor readings can be up to 2 seconds old.

Specifications

- Low cost
- 3 to 5V power and I/O
- 2.5mA max current use during conversion (while requesting data)
- Goodfor20-80%humidityreadingswith5%accuracy
- Good for 0-50°C temperature readings ±2°Caccuracy
- Bodysize15.5mmx12mmx5.5mm
- 4 pins with 0.1" spacing

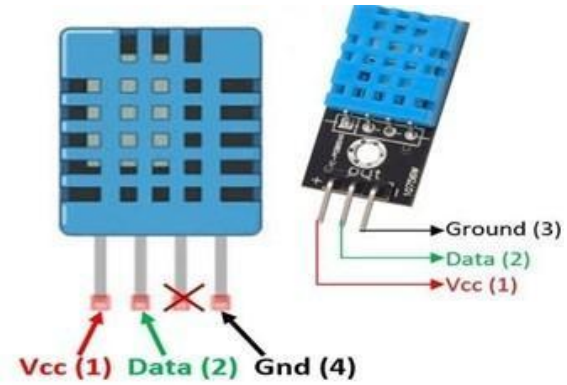
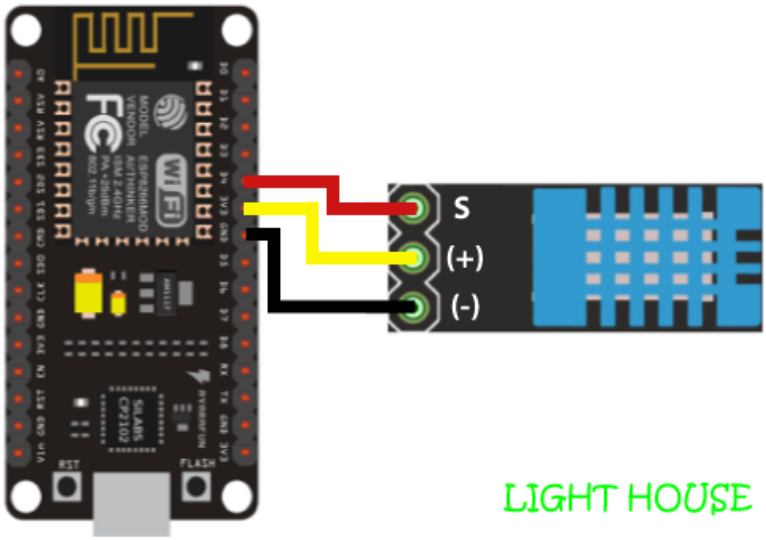


Fig: 2 Pin description of DHT11

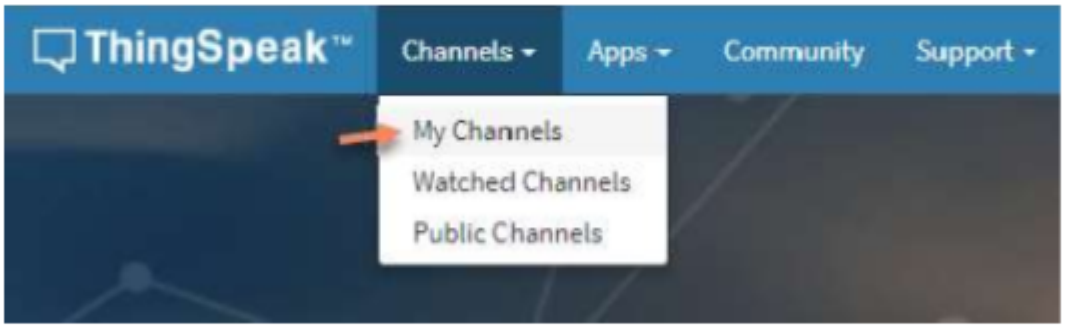
No:	Pin Name	Description
For Sensor		
1	Vcc	Power supply 3.5V to 5.5V
2	Data	Outputs both Temperature and Humidity through serial Data
3	NC	No Connection and hence not used
4	Ground	Connected to the ground of the circuit
For module		
1	Vcc	Power supply 3.5V to 5.5V
2	Data	Outputs both Temperature and Humidity through serial Data
3	Ground	Connected to the ground of the circuit



Interfacing of DHT11

Create a Channel

1. SignIntoThingSpeakusingyourMathWorks®Account credentials,or create a new account.
2. Click **Channels** >**MyChannels**.



3. On the Channels page, click **NewChannel**.
4. Check the boxes next to Fields 1–3. Enter these channel settingvalues:

- **Name**: Dew PointMeasurement
- **Field 1**: Temperature(F)
- **Field 2**: Humidity
- **Field 3**: DewPoint

5. Click **Save Channel** at the bottom of the settings. You now see these tabs:

- **Private View**: This tab displays information about your channel that only you can see.
- **Public View**: If you choose to make your channel publicly available, use this tab to display selected fields and channel visualizations.
- **Channel Settings**: This tab shows all the channel options you set at creation. You can edit, clear, or delete the channel from this tab.
- **Sharing**: This tab shows channel sharing options. You can set a channel as private, shared with everyone (public), or shared with specific users.
- **API Keys**: This tab displays your channel API keys. Use the keys to read from and write to your channel.
- **Data Import/Export**: This tab enables you to import and export channel data.

6. Your channel is available for future use by clicking **Channels** >**MyChannels**.

Procedure:

- Create a channel on the cloud (Thingspeak)
- Make the connection as per circuit diagram
- Open Python IDE
- Write code in Editor window and save file
- Compile the code
- Open thingspeak channel
- Observe the result.

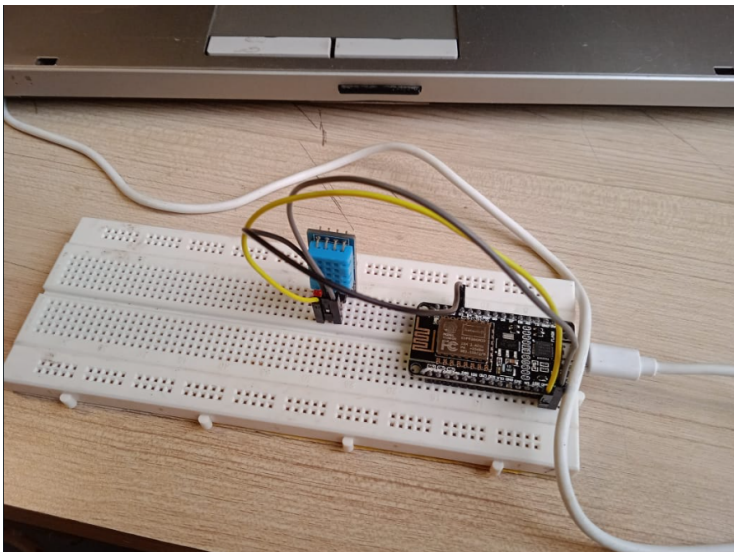
Arduino Code

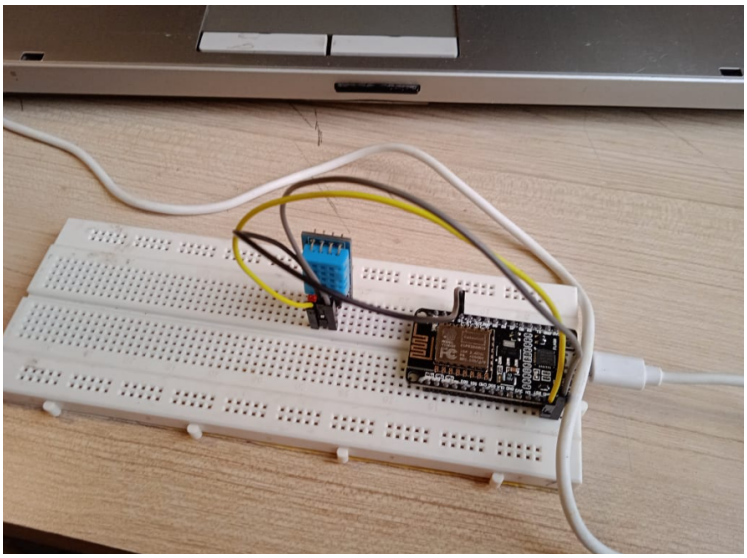
```
#include <DHT.h>  // Including library for dht


#include <ESP8266WiFi.h>


String apiKey = "T35KP6JY8R6LD2V8";     //  Enter your Write API key from ThingSpeak


const char *ssid =  "Center";     // replace with your wifi ssid and wpa2 key
const char *pass =  "20202020";
const char* server = "api.thingspeak.com";


#define DHTPIN 0          //pin where the dht11 is connected


DHT dht(DHTPIN, DHT11);


WiFiClient client;
```

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

      Serial.println("Connecting to ");
      Serial.println(ssid);



      WiFi.begin(ssid, pass);

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

}

void loop()
{

     float h = dht.readHumidity();
     float t = dht.readTemperature();

           if (isnan(h) || isnan(t))
              {
                 Serial.println("Failed to read from DHT sensor!");
                  return;
              }
```

```
            if (client.connect(server,80))   //   "184.106.153.149" or api.thingspeak.com
          {

              String postStr = apiKey;
              postStr +="&field1=";
              postStr += String(t);
              postStr +="&field2=";
              postStr += String(h);
              postStr += "\r\n\r\n";

              client.print("POST /update HTTP/1.1\n");
              client.print("Host: api.thingspeak.com\n");
              client.print("Connection: close\n");
              client.print("X-THINGSPEAKAPIKEY: "+apiKey+"\n");
              client.print("Content-Type: application/x-www-form-urlencoded\n");
              client.print("Content-Length: ");
              client.print(postStr.length());
              client.print("\n\n");
              client.print(postStr);

              Serial.print("Temperature: ");
              Serial.print(t);
              Serial.print(" degrees Celcius, Humidity: ");
              Serial.print(h);
              Serial.println("%. Send to Thingspeak.");
           }
    client.stop();

    Serial.println("Waiting...");

// thingspeak needs minimum 15 sec delay between updates
delay(100);
```

}



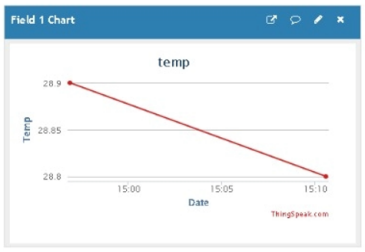


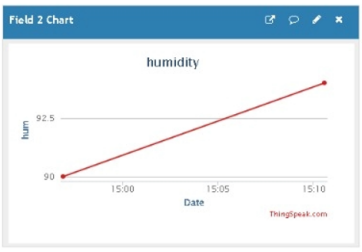
Add Visualizations | Add Widgets | Export recent data | MATLAB Analysis | MATLAB Visualization

Channel 4 of 4

Channel Stats

Created: about 14 hours ago
Last entry: 3 minutes ago
Entries: 2





Conclusion:

Experiment No. – 6

Title: Get the status of a bulb at a remote place (on the LAN) through web.
Date of Performance:
Date of Submission:
Aim: We'll build and demonstrate a simple remote controlling a device from a web browser.
Objectives: Controlling device through web.
Hardware Requirements:

- LED
- ESP8266
- Breadboard

Software Requirements:

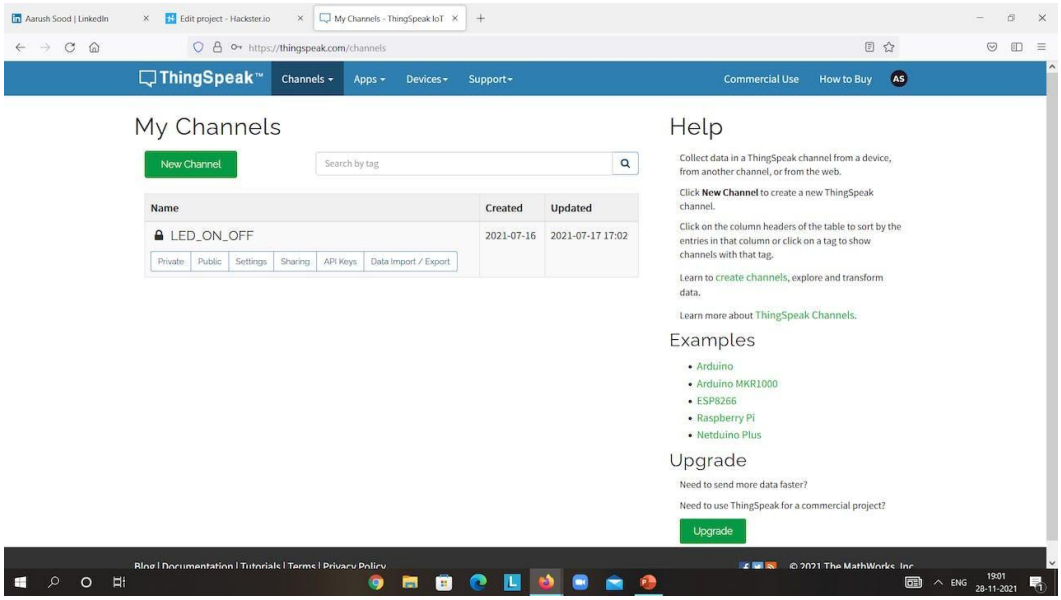
- Arduino IDE
- ThingSpeak API

Theory:

Light-emitting diode (LED) is a semiconductor device that emits light when an electric current is passed through it. Light is produced when the particles that carry the current (known as electrons and holes) combine together within the semiconductor material.
Linking with ThingSpeak Server and creating your own App

Open the website and sign in.

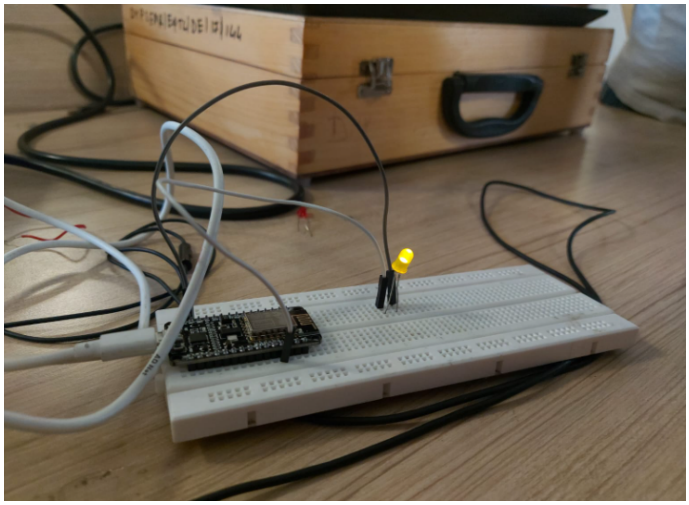
Under the channels bar click on New Channels and give it any name



Procedure:

1. Open the website and sign in.
2. Under the channels bar click on New Channels and give it any name
3. you want
4. Then click the API tab and then click on generate new write API key
5. and copy the (selected) link as shown in the image.
6. Now paste that link in new tab and change links last value to 1 to turn on the led and 0 to turn and check the status on thingspeak .

Circuit Diagram:



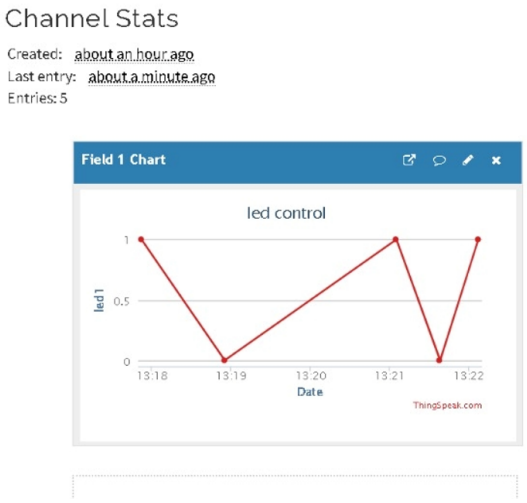
Program:#include "ThingSpeak.h"
#include <ESP8266WiFi.h>

//Replace your wifi credentials here
const char* ssid = "Center-2";
const char* password = "Admin@1234";

```
//change your channel number here
unsigned long channel =1590278;//Replace with your own ThingSpeak Account Channle ID


unsigned int led1 = 1;


WiFiClient  client;


void setup() {
  Serial.begin(115200);
  delay(100);

  pinMode(0, OUTPUT);

  digitalWrite(0, 0);



  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");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
  Serial.print("Netmask: ");
  Serial.println(WiFi.subnetMask());
  Serial.print("Gateway: ");
  Serial.println(WiFi.gatewayIP());
  ThingSpeak.begin(client);

}

void loop() {
```

```
  //get the last data of the fields
  int led_1 = ThingSpeak.readFloatField(channel, led1);

  if(led_1 == 1){
    digitalWrite(0, 1);
    Serial.println("D1 is On..!");
  }
  else if(led_1 == 0){
    digitalWrite(0, 0);
    Serial.println("D1 is Off..!");
  }



  Serial.println(led_1);

  delay(500);

}
```

Output:



Conclusion:

_____

_____

_____

Experiment No. –7

Title: Interfacing Arduino to Bluetooth Module

Date of Performance:
Date of Submission:

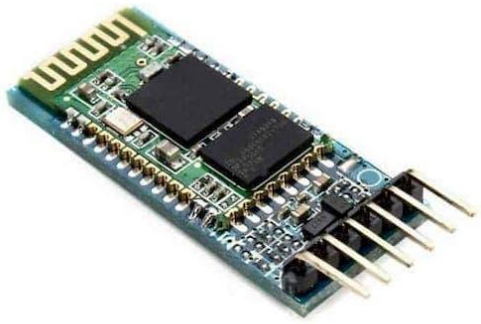
Aim: Interface a Bluetooth module with HC-05 with Arduino Uno.

Objectives: Interface a Bluetooth module with HC-05 with Arduino Uno.

Hardware Requirements:
- Bluetooth Module HC-05
- Arduino UNO

Software Requirements:
- Arduino IDE

Theory:
Bluetooth is a one of the great example for wireless connectivity. It is used in many fields. Bluetooth consumes very small amount of energy. Do you know about Smartphone controlled robot or car. Commonly one of these two wireless technology is used in Smartphone controlled robot. One is WIFI and other is Bluetooth. And another commonly used wireless technology for controlling Robot car is RF. Which is the same remote and receiver used in drones. Here we are going to interface a Bluetooth Module (HC-05) with Arduino Uno.

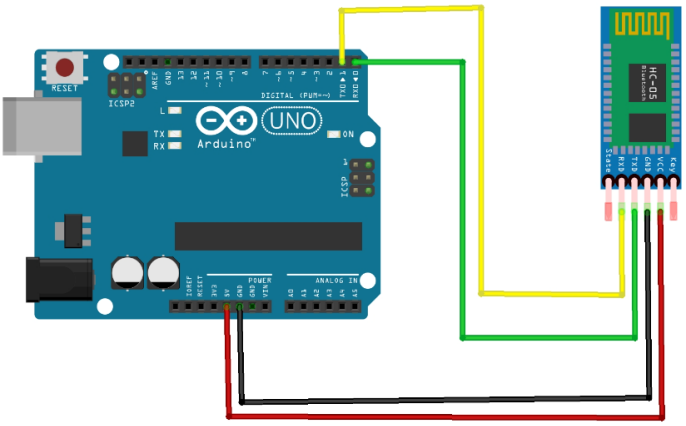


HC-05 is a Bluetooth module which can communicate in two way. Which means, It is full-duplex. We can use it with most micro controllers. Because it operates Serial Port Protocol (SSP). The module communicate with the help of USART (Universal Synchronous/Asynchronous Receiver/Transmitter) at the baud rate of 9600. It also support other baud rate. So we can interface this module with any microcontroller which supports USART.
As HC-05 Bluetooth module has 3.3 V level for RX/TX and microcontroller can detect 3.3 V level, so, there is no need to shift TX voltage level of HC-05 module. But we need to shift the

transmit voltage level from microcontroller to RX of HC-05 module.

Technical Specifications

- Operating Voltage: 4V to 6V (Typically +5V)
- Operating Current: 30mA
- Range: <100m
- Works with Serial communication (USART) and TTL compatible
- Can be easily interfaced with Laptop or Mobile phones with Bluetooth



Procedure:

1. Create a sketch for Arduino Uno to Interface the HC-05.

2. Code the setup part. HC-05 use the serial communication. So begin the serial communication by using the function "Serial.begin()".

3. Code the loop part. Use a while loop and the function "Serial.available()".

4. Make the Connections Arduino Uno HC-05.

Program:

```
char inputByte;
void setup(){
 Serial.begin(9600);
 pinMode(13,OUTPUT);

}

void loop(){
while(Serial.available()>0){
inputByte= Serial.read();
```

```
  Serial.println(inputByte);
if(inputByte=='Z'){
  digitalWrite(13,HIGH);
}
elseif(inputByte=='z'){
  digitalWrite(13,LOW);
}
}
}
```

Conclusion:

_____

_____

_____

_____

Experiment No. 8

Title: Project (Automatic Room Temperature Controller)

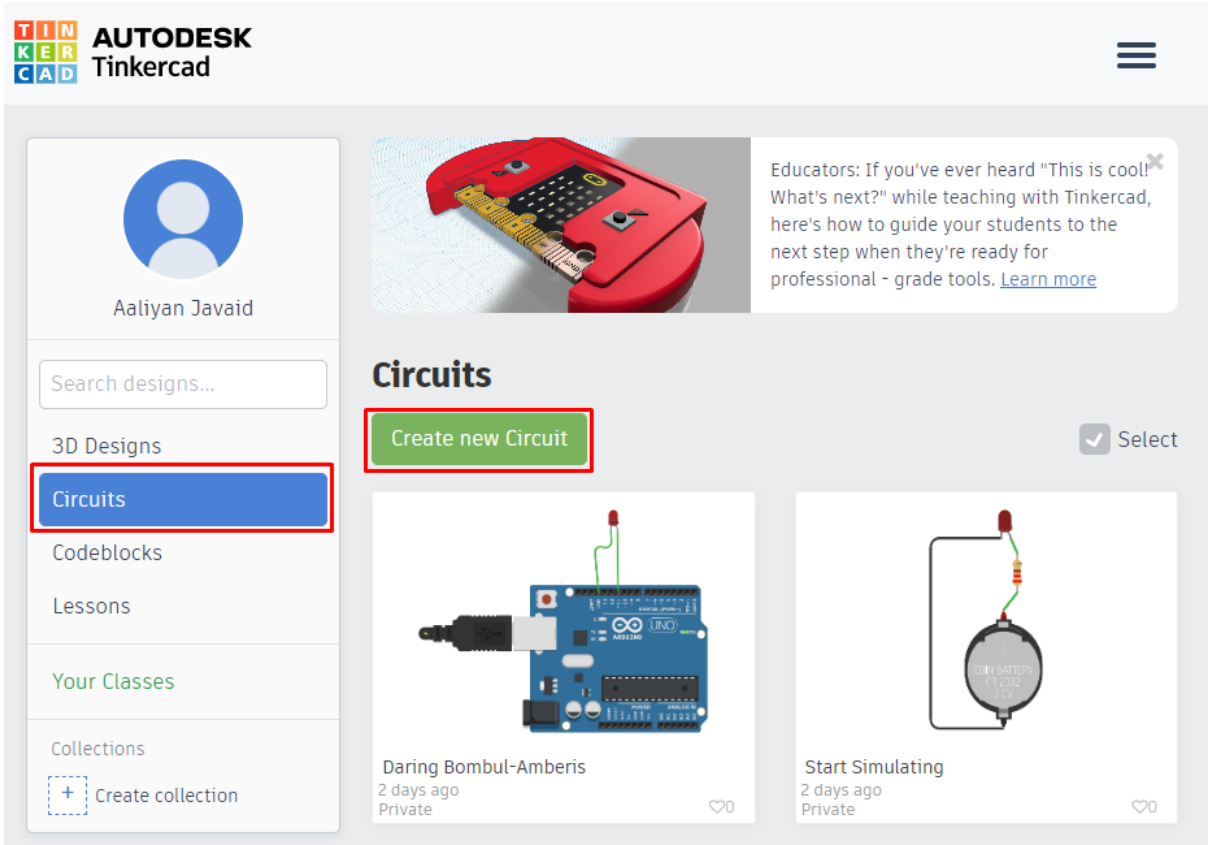
Date of Performance:
Date of Submission:

Aim: To create a project on Tinkercad
Theory:

What is Tinkercad?

Tinkercad is an online collection of software tools from Autodesk that enable complete beginners to create 3D models. This CAD software is based on constructive solid geometry (CSG), which allows users to create complex models by combining simpler objects together. As a result, this 3D modeling software is user-friendly and currently enjoyed by many, particularly teachers, kids, hobbyists, and designers. Best of all, it's free and you only need an internet connection to answer it. The software allows users to create models that are compatible with 3D printing, a great option for beginners to the technology.

Tinkercad is a good alternative to other 3D modeling software such as SketchUp or Fusion360—another solution from Autodesk—if you do not need the more advanced features of these solutions.



Even though Tinkercad is perfect for beginners, it does not mean that those who are more experienced with 3D modeling will not also enjoy this software. Given that it is based on CSG to create solid models, you can always make your model more complex by adding more shapes. In more concrete terms, all you have to do is select one of the available shapes, add or remove material and voila you're done! For example, you could start with a cylinder before adding triangles, circles, cones, etc. The shape can then be moved and rotated, allowing users to see it from all angle.

Additionally, the software allows you to add electronic circuits to 3D models in order to create objects with light and movement. The end result can even be simulated on the software to check how the components will respond in real life. Another feature of Tinkercad is it's ability to transform a 3D design into buildable brick models, similar to creating legos. Finally, for those that love Minecraft, you will be well served, as you will be able to make creations compatible with the application.

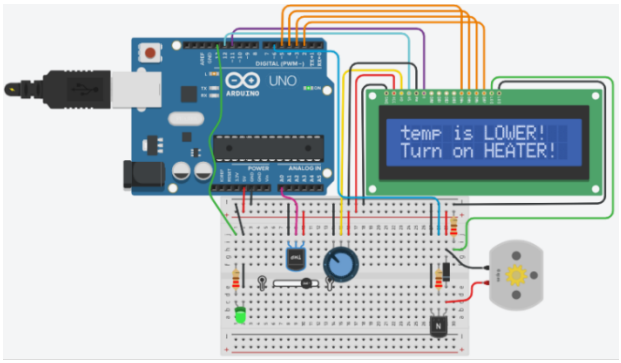
Sample Project: Atomatic Room Temperature Controller

Weather changes become hard to adapt. That is, during Winter we face difficulties tolerating the freezing cold, and that is why people often prefer wearing coats during the season. On the other hand, the weather becomes too warm in summer. Thus, having understood the switching operation of transistors, unidirectional current flow in diodes, the principle of operation of motors, the resistance from resistors combined with the transformation capability of the transducer, the temperature sensor in this case.
1. If the temperature exceeds the maximum of the aforementioned "desired" range, then the LCD displays that the temperature is higher and informs the FAN to turn on. Then the FAN starts its rotation/vibration, and after a while the temperature gets lowered falling in the range, then the LCD commands the FAN to turn off.
2. Whenever the sensor's temperature reading goes down below the possible minimum temperature in the range, the LCD notifies that the temperature is LOWER and tells the heater to be turned on, and after the temperature is in the range, it displays that temperature is OK and orders the heater to be switched off.
3. The last condition is whenever the temperature is within the desired range, the LCD tells that the temperature is normal. Thus, it asks to turn off everything. This condition can, for instance, be observed if you run the code and the slider's position remains unchanged. That tells one that the default slider position is within the desired room temperature range (i.e. the default is 24.78 degree C).
Likewise, system iterates forever unless it gets terminated by the user due to the behavior of the loop ( ) function.

Software Requirement:

- Arduino IDE
- Tinkercard

Circuit Diagram:



Program:

```
 // Declare/assign Arduino IO-pins

  const int temp_trans_pin = A0, Heater_pin = 13, FAN_pin = 6;
 /*FAN_pin: here I used DC motor in stead of FAN because
  I couldn't find the symbol for it. Similarly,for the
   Heater (Heater_pin), I used LED.*/

// Set the range of the desired temperature

   float MinTemp = 20, MaxTemp = 25;/*Room temperature is [20,25] degree C */

// Include the LCD library code

  #include <LiquidCrystal.h>

 // Initialize the library with the numbers of the interface pins

  LiquidCrystal LCD(12, 11, 5, 4, 3, 2);

  void setup() {

  // System initialization

    LCD.begin(16, 2);
    pinMode(Heater_pin, OUTPUT);//LED in our case
    pinMode(FAN_pin, OUTPUT);

  // Display the desired range of temperature

    LCD.print("Room temp(C):");
    LCD.setCursor(2,1);
    LCD.print(MinTemp); LCD.print("-");LCD.print(MaxTemp);

    delay(2000);
 }
```

```
void loop() {

  float Eqv_volt, SensorTemp;

// Read voltage and convert to temperature (Celsius)

  Eqv_volt = analogRead(temp_trans_pin) * 5.0 / 1023;
  SensorTemp = 100.0 * Eqv_volt-50.0;

 // Display the sensor reading

   LCD.clear();
   LCD.print("Sensor reading:");
   LCD.setCursor(2,1);
   LCD.print(SensorTemp); LCD.print(" C");

   delay(2000);

/*Compare the sensor reading with the range of
 acceptable temperatures*/

  if(SensorTemp > MaxTemp){
     LCD.clear();
     LCD.print("temp is HIGHER!");//higher than the max

     /*Turn on FAN (dc motor)! to regulate the temp.
      Increase FAN speed at a slow rate*/

     LCD.setCursor(0, 1);LCD.print("Turn on FAN!");
     for( int i = 0; i <= 255; i++ ) {
       analogWrite(FAN_pin, i);
      }
      delay(2000);

      LCD.clear();
      LCD.print("Now temp is OK!");
      LCD.setCursor(0, 1);
      LCD.print("Turn off FAN!");

 // Turn off FAN slowly
      for( int i = 255; i >= 0; i-- ) {
       analogWrite(FAN_pin, i);
      }
       delay(2000);
      }
 else if(SensorTemp < MinTemp){
     LCD.clear();
     LCD.print("temp is LOWER!");//Less than the mini
     LCD.setCursor(0, 1);
     LCD.print("Turn on HEATER!");
```

```
    //Turn the heater ON, LED in our case

     digitalWrite(Heater_pin, HIGH);

     delay(3000);

     LCD.clear();
     LCD.print("Now temp is OK!");
     LCD.setCursor(0, 1);
     LCD.print("Turn off HEATER!");

     delay(1000);

     digitalWrite(Heater_pin, LOW);
     LCD.clear();
     }
  else if(SensorTemp > MinTemp && SensorTemp < MaxTemp){/*Now temperature is perfect.
       That is,it is in the desired range. Hence no need of changes!!*/
     LCD.clear();
     LCD.print("Temp is NORMAL!");LCD.setCursor(2,1);
     LCD.print("Turn off all!");

     delay(1000);
     LCD.clear();
  }
  else {
     LCD.clear();
     LCD.print("Something went");
     LCD.setCursor(2,1); LCD.print("WRONG in the ckt");
     delay(1000);
     LCD.clear();
    }
    delay(1000);
  }
```

Conclusion:

_____
_____
_____
_____