CHITKARA UNIVERSITY INSTITUTE OF ENGINEERING & TECHNOLOGY

COMPUTER SCIENCE ENGINEERING

Internet of Things

Course Code – CS201

Submitted By: **Submitted To:**

Name:
Roll No:
Class:
Session:

CHITKARA UNIVERSITY
NH-7, CHANDIGARH PATIALA NATIONAL HIGHWAY
RAJPURA, PUNJAB 140401.

INDEX

S. No.	Lab Experiment	Date	Signature
1	Write the code to blink an LED on Arduino MKR1000. Compile and verify the result on the serial monitor of Arduino IDE.		
2	Interfacing of Arduino MKR1000 with LED and switch. Write a program to control LED using Switch.		
3	Interfacing of Arduino MKR1000 with potentiometer and LED. Write a program to vary the intensity of LED using a potentiometer.		
4	(a) Interfacing of Ultrasonic sensor with Arduino MKR1000. Write a program to measure the distance from obstacle and display on the serial monitor. (b) Interface an IR sensor with Arduino MKR1000. Write a program to detect obstacle and display on the serial monitor.		
5	(a) Interfacing of Temperature sensor with Arduino MKR1000. Write a program to read the specific temperature of a room and display on the serial monitor. (b) Interfacing of LDR with Arduino MKR1000. Write a program to control the intensity of LED using LDR.		
6	(a) Interfacing of DC motor with Arduino MKR1000. Write a program to rotate the motor in clockwise and anticlockwise direction with using a delay of 2 sec. (b) Familiarize the concept of pulse width modulation. Write a program to control the speed of DC motor using PWM.		
7	(a) Interfacing of a display device, i.e., LCD x2 with Arduino MKR1000. Write a program to display “HELLO IOT” on LCD. (b) Write a program to scroll the message “Welcome to IoT Lab.” (c) Write a program to blink the message “Hello IoT.”		
8	Write down the steps to connect Arduino MKR1000 with IoT cloud. Write a program to blink an LED using Arduino IoT cloud.		
9	Write a program to visualize data of temperature sensor on Arduino IoT cloud.		
10	Design an interface to control the speed of DC motor using cloud services. Store and visualize the data of speed of DC motor after every one sec.		

Aim of the Experiment: Write the code to blink an LED on Arduino MKR1000. Compile and verify the result on the serial monitor of Arduino IDE.

Components Used: Arduino MKR1000, Resistor, LED, Bread-Board, Wires

Working:

- Connect the LED's cathode to the ground of MKR and anode to output pin through a resistor.
- Turn the LED on and off with the use of MKR1000. This works with the help of code which calls the digitalWrite() function twice, one with HIGH to turn the LED on and one with LOW to turn the LED off.
- Add two calls to delay() to slow things down otherwise the LED will turn ON and OFF too quickly which is not able to notice.
- The delay function works with milliseconds, so we pass it 500 to pause for a second. When the LED is ON "Led ON" is printed on the serial monitor and "Led OFF" when LED is OFF.

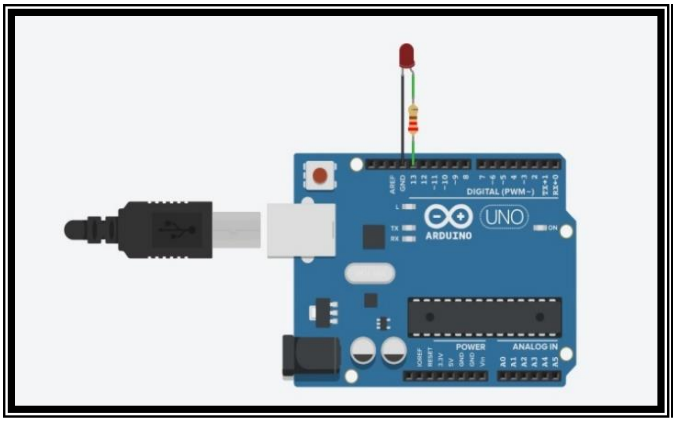


Fig 1.1 Circuit Diagram

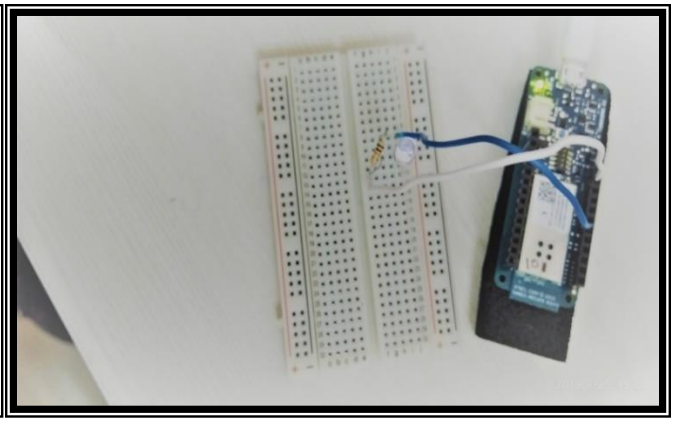
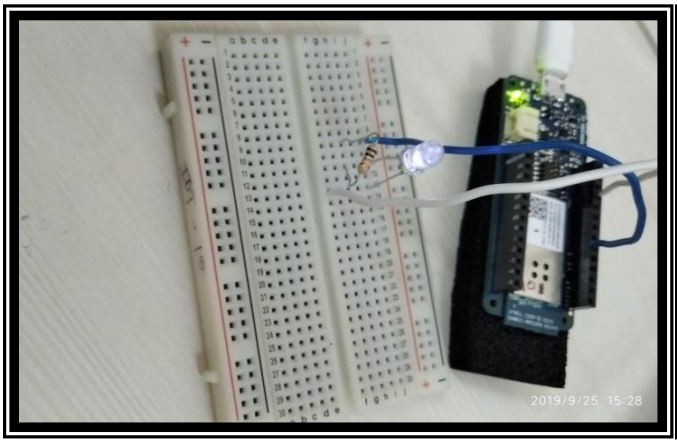


Fig 1.2 LED is OFF



. Fig 1.3 LED is ON

Fig. 1.4 Displayed result Serial Monitor

CODE:

```
void setup()
{
pinMode(11,OUTPUT);                    // define LEDpin as output
Serial.begin(9600);
}

void loop() {
digitalWrite(11,HIGH);                 // set pin state HIGH to turn LED on
Serial.println("Led ON");              // print Led ON on serial monitor
delay(500);                            // wait for half second = 500ms
digitalWrite(11,LOW);                  // set pin state LOW to turn LED off
Serial.println("Led OFF");             //print Led OFF on serial monitor
delay(500);                            // wait for half second = 500ms
}
```

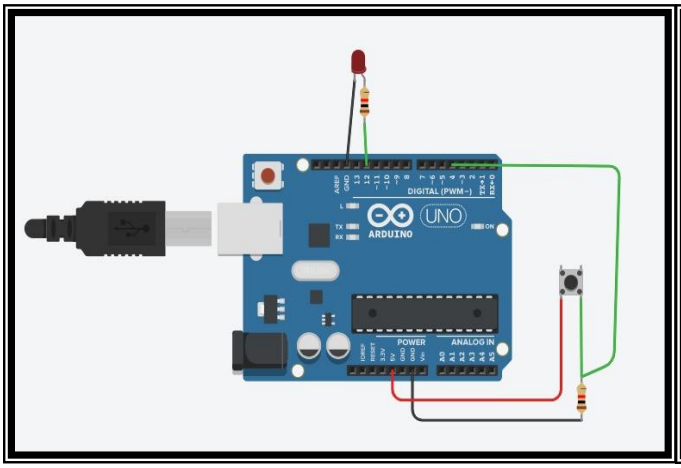
Aim of the Experiment: Interfacing of Arduino MKR1000 with LED and switch. Write a program to control LED using Switch

Components Used: LED,Switch,Breadboard,Arduino MKR1000,Resistor

Working:

Now here we will switch the LED ON or OFF using a switch.

- The common leg of ON/OFF push button is connected to 5V supply and the other is connected to the LED through the resistor.
- It allows the current to flow through it only when we switch on the button, the LED will start glowing when it is pressed the first time.
- It continues the power supply till the switch is ON, the LED turned off when it is pressed the second time.



. Fig 2.1 Circuit Diagram

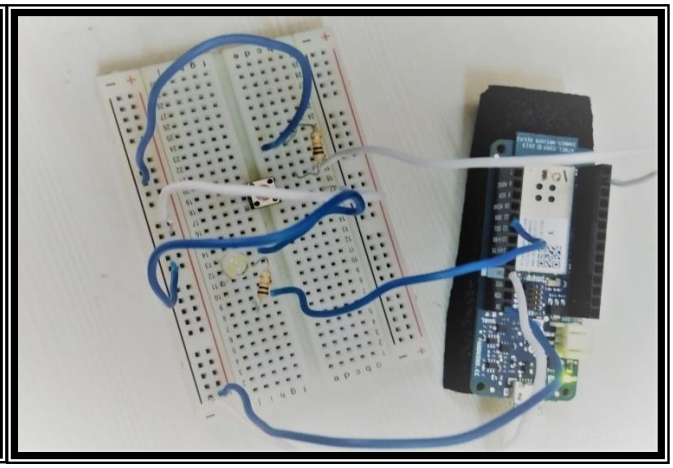


Fig 2.2 Button not pressed, LED is OFF

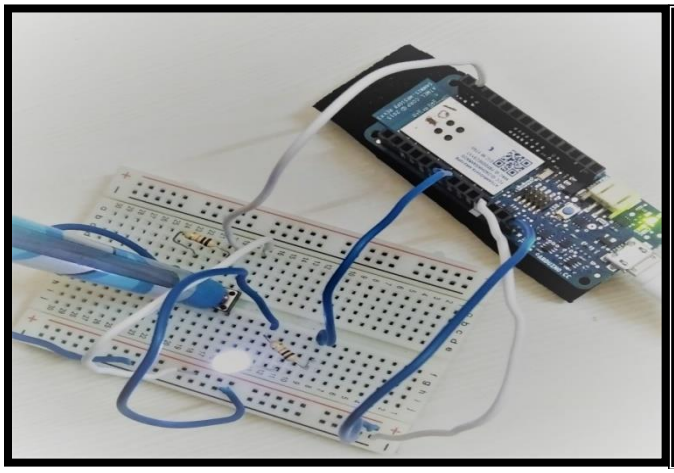


Fig 2.3 Button pressed, LED turns ON

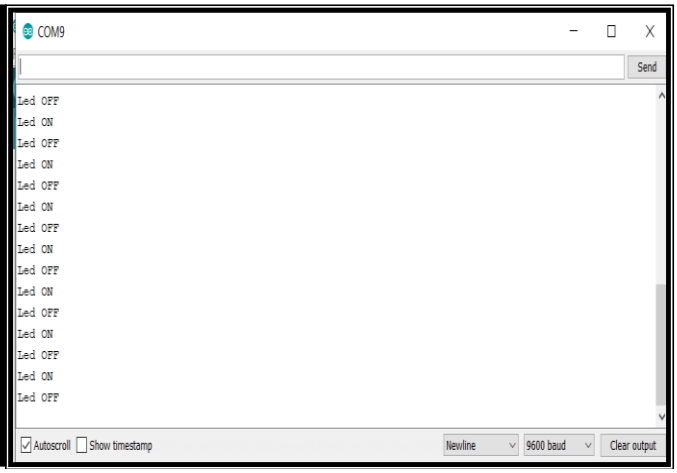


Fig. 2.4 Displayed result Serial Monitor

CODE:

```
void setup()
{
pinMode(11,OUTPUT);                    // pin 11 as Output Pin
pinMode(4,INPUT);                      // pin 4 as Input Pin
Serial.begin(9600);
}
void loop()
{
if(digitalRead(4)==HIGH)               //pin 4 set to HIGH
{
digitalWrite(11,HIGH);
Serial.println("Led ON");
}
Else
{
digitalWrite(11,LOW);                  //pin 11 set to LOW
Serial.println("Led OFF");
}
delay(10);
}
```

Aim of the Experiment: Interfacing of Arduino MKR1000 with potentiometer and LED. Write a program to vary the intensity of LED using a potentiometer.

Components Used: LED,Potentiometer,Breadboard,Arduino MKR1000,Resistor

Working:

In this experiment we will change the brightness of a LED using a Potentiometer.

- A potentiometer has 3 pins. The two terminals (on the corners) are connected to 5V input supply and the ground while the third terminal (middle one) is connected to an adjustable wiper.
- The potentiometer can vary the resistance from 0V to its maximum limit by rotating the wiper from left to right or vice-versa.

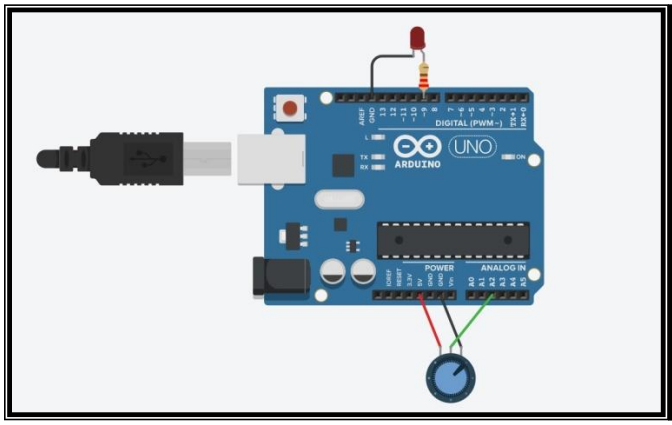


Fig 3.1 Circuit Diagram

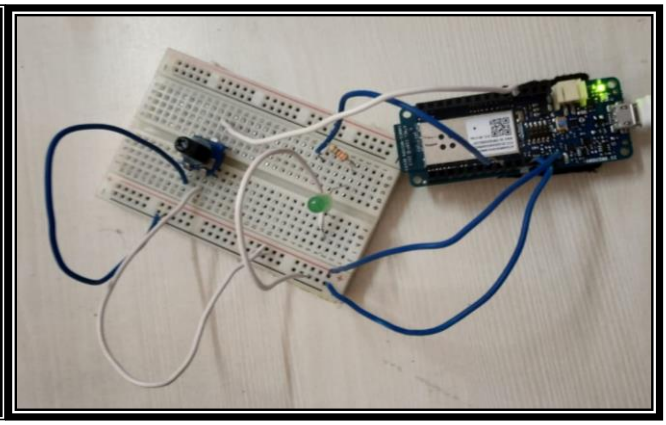


Fig 3.2 LED is OFF at lowest value of Potentiometer

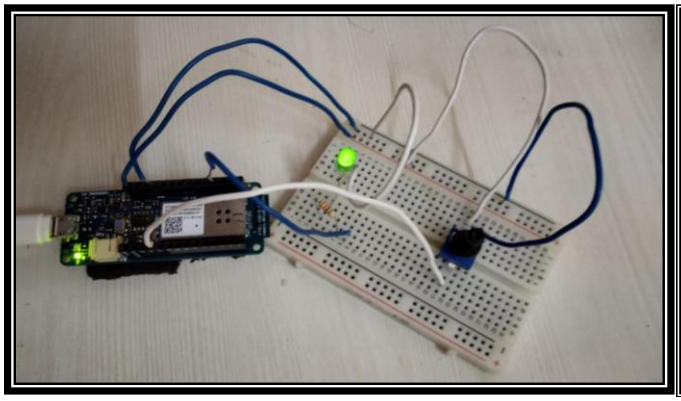


Fig 3.3 As the value increases LED grows brighter

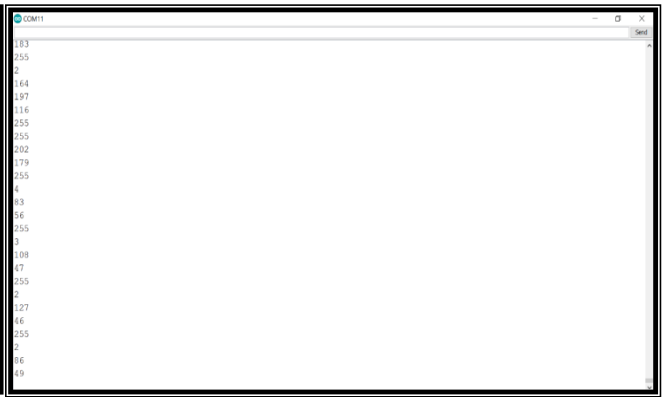


Fig 3.4 Display of output on the Serial Monitor

CODE:

```
void setup()
{
pinMode(A1,OUTPUT);                  // analog pin A1 defined as output
Serial.begin(9600);
}

void loop()
{
int pot=analogRead(A2);              // variable pot is taken to store values from potentiometer
int bright=pot/4;
delay(100);
analogWrite(9,bright);               //analog pin 9 set to value received from potentiometer
Serial.println(bright);
}
```

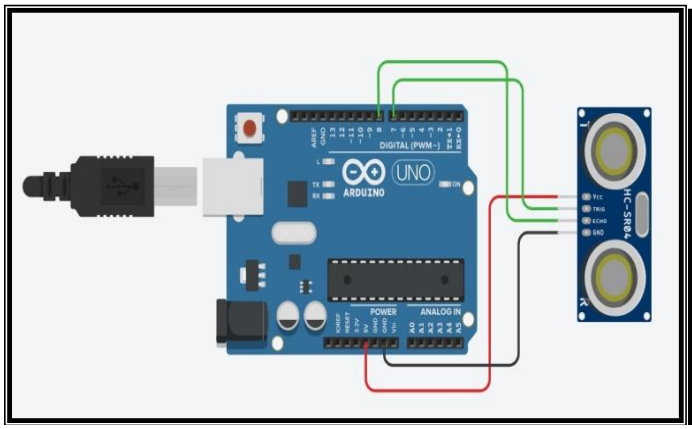
Aim of the Experiment: Interfacing of Ultrasonic sensor with Arduino MKR1000. Write a program to measure the distance from obstacle and display on the serial monitor.

Components Used: Ultrasonic Sensor, Breadboard, Arduino MKR1000

Working :

Here we will measure the distance of an obstacle using a ultrasonic sensor and will display the values in cm and inches on the serial monitor.

- The ultrasonic sensor works on the principle of SONAR and RADAR system which is used to determine the distance to an object.
- An ultrasonic sensor generates the high-frequency sound (ultrasound) waves.
- When this ultrasound hits the object, it reflects as echo which is sensed by the receiver as shown in below figure. By measuring the time required for the echo to reach to the receiver, we can calculate the distance.



. Fig 4.1 Circuit Diagram

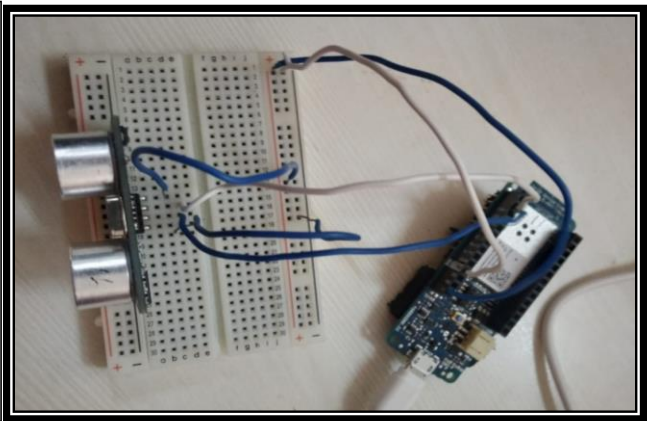


Fig 4.2 Top View of the circuit

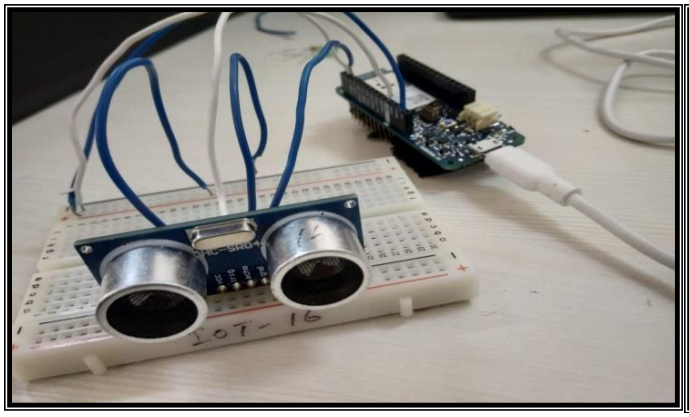


Fig 4.3 Front View of the circuit

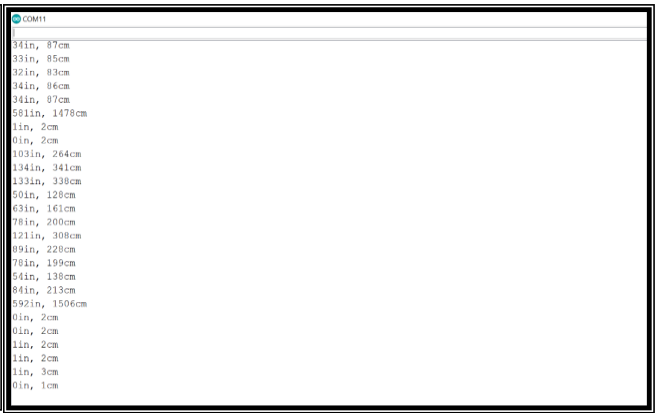


Fig 4.4 Readings of the sensor printed on Serial Monitor

CODE:

```
int trigPin = 7;                    // trigger pin connected on 7
int echoPin = 8;                    // echopin connected on 8
long duration, cm, inches;
void setup()
{
Serial.begin (9600);                // serial monitor added
pinMode(trigPin, OUTPUT);           // trigpin defined as output
pinMode(echoPin, INPUT);            // echopin defined as input
}
void loop()
{                                   //toggling trigPin and echoPin      ON and
                                    //OFF
digitalWrite(trigPin, LOW);
delayMicroseconds(5);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
pinMode(echoPin, INPUT);
duration = pulseIn(echoPin, HIGH);

cm = (duration/2) / 29.1;           //pre defined formula for cm
inches = (duration/2) / 74;         // pre defined formula for inches

Serial.print(inches);
Serial.print("in, ");
Serial.print(cm);
Serial.print("cm");
Serial.println();

delay(70);
}
```

Aim of the Experiment: Interface an IR sensor with Arduino MKR1000. Write a program to detect obstacle and display on the serial monitor.

Components Used: Arduino MKR1000, IR sensor, Resistor, LED, Bread-Board, Wires

Working:

IR sensor here will switch the LED ON/OFF based on the obstacle in front of it.

- The ground of the IR sensor is connected to the ground of MKR and VCC to the 5V input pin.
- The middle pin of the IR sensor is connected to pin 4 of MKR as input.
- An LED is connected to the MRK through a resistor to pin 11 for output.
- When and object is placed in IR's range the IR sends a single to the MKR and the MRK turns the LED on and turns off when the object is moved away.
- When the object is in the range, "Object Found" is printed on the serial module and "No Object" when object is moved away.

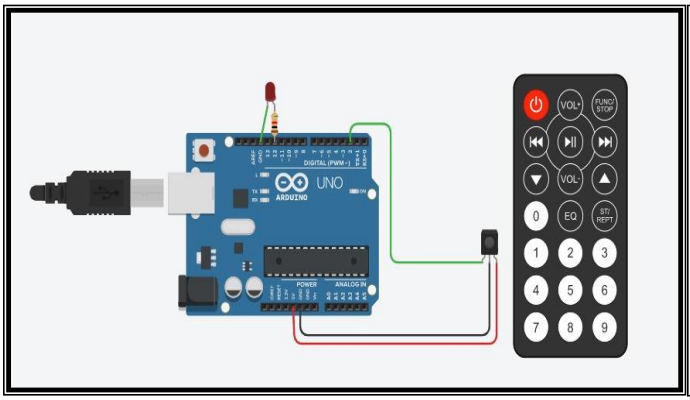


Fig 5.1 Circuit Diagram

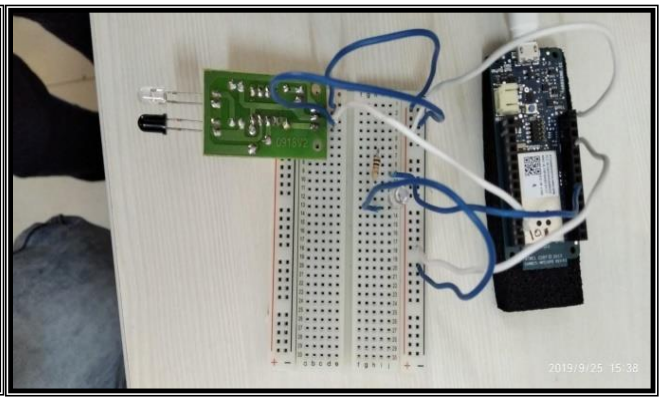


Fig 5.2 No Obstacle in IR's range, LED is OFF

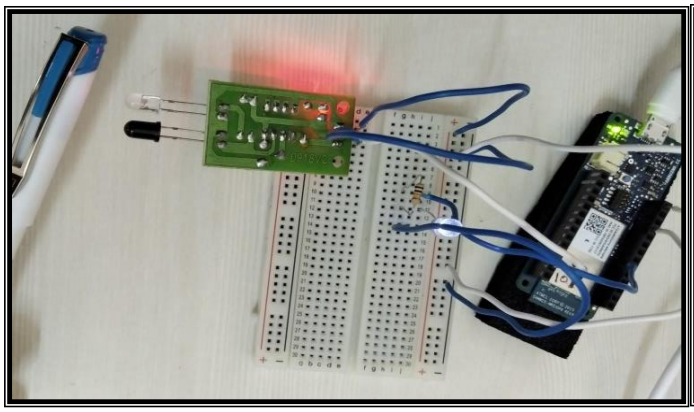


Fig 5.3 Obstacle placed in IR's range LED turns ON

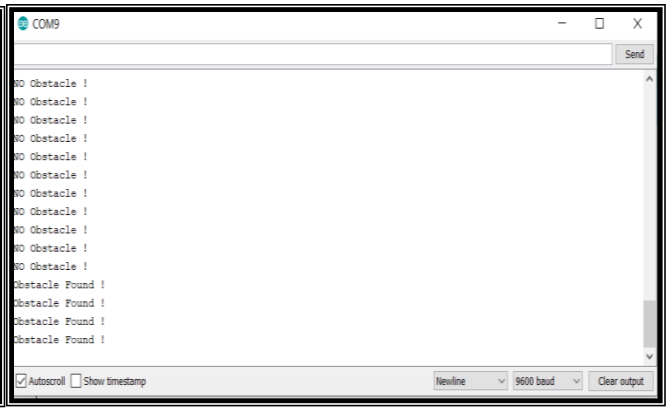


Fig 5.4 Display of output on the Serial Monitor

CODE:

```
void setup()
{
pinMode(11,OUTPUT);                    // set pin 11 as output
pinMode(4,INPUT);                      // set pin 4 as input
Serial.begin(9600);
}

void loop()
{
if(digitalRead(4)==HIGH)               //Check the sensor output
{
digitalWrite(11,HIGH);                 // set the LED on
Serial.println("Obstacle Found !");    //print Obstacle Found
delay(100);
}
else
{
digitalWrite(11,LOW);                  // set the LED off
Serial.println("NO Obstacle !");       //print No Obstacle
delay(100);
}
}
```

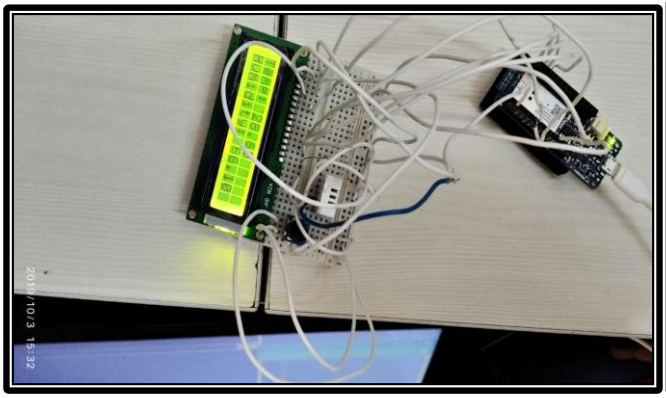
Aim of the Experiment: Interfacing of Temperature sensor with Arduino MKR1000. Write a program to read the specific temperature of a room and display on the serial monitor.

Components Used: Arduino MKR1000, Potentiometer, LCD, DHT11 Sensor , Bread-Board, Resistor, Wires

Working:

DHT11 sensor is a electronic sensor that is used to sense the humidity and the temperature of the room in which it is present.

- It consists of a capacitive humidity sensing element and a thermistor for sensing temperature.
- The temperature range of DHT11 is from 0 to 50 degree Celsius with a 2-degree accuracy.
- Humidity range of this sensor is from 20 to 80% with 5% accuracy.
- DHT11 is small in size with operating voltage from 3 to 5 volts. The maximum current used while measuring is 2.5mA.



. Fig 6.1 Less Humidity and Room

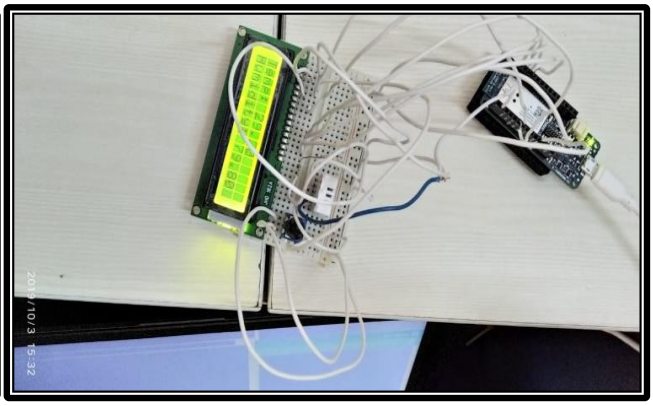


Fig 6.2 High Humidity on blowing air Temperature

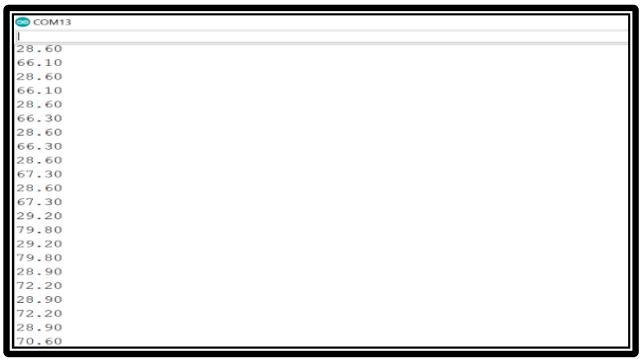


Fig. 6.3 Displayed result Serial Monitor

CODE:

```
#include <DHT.h>                          // DHT libraries included
#include <DHT_U.h>
#include <LiquidCrystal.h>                // LCD library included

#define DHTPIN 7
#define DHTTYPE DHT22
LiquidCrystal lcd(12,11,5,4,3,2);
DHT dht(DHTPIN, DHTTYPE);
float temp, humidity;                     // temperature and humidity defined
void setup()
{
  Serial.begin(9600);                     // serial monitor defined
  dht.begin();
  lcd.begin(16,2);                        // lcd screen size declared
}
void loop()
{
  temp = dht.readTemperature();
  humidity = dht.readHumidity();
  lcd.setCursor(0,0);
  lcd.print("Temp:");
  lcd.setCursor(6,0);
  lcd.print(temp);                        // value of temperature from sensor
                                          //displayed on lcd
  lcd.setCursor(0,1);
  lcd.print("Humidity:");
  lcd.setCursor(10,1);
  lcd.print(humidity);                    // value of humidity from sensor displayed
                                          //on lcd

  Serial.println(temp);                   // values shown on serial monitor
  Serial.println(humidity);
  delay(1000);
}
```

Aim of the Experiment: Interfacing of LDR with Arduino MKR1000. Write a program to control the intensity of LED using LDR.

Components Used: Arduino MKR1000, LDR, LED, Bread-Board, Resistor, Wires

Working:

LDR is a two pin device that increases the resistance with increase in intensity of light.

- The LDR can vary the resistance from 0V to its maximum limit depending upon the intensity of light.
- It is connected always to a analog pin as the value received from it varies continuously.

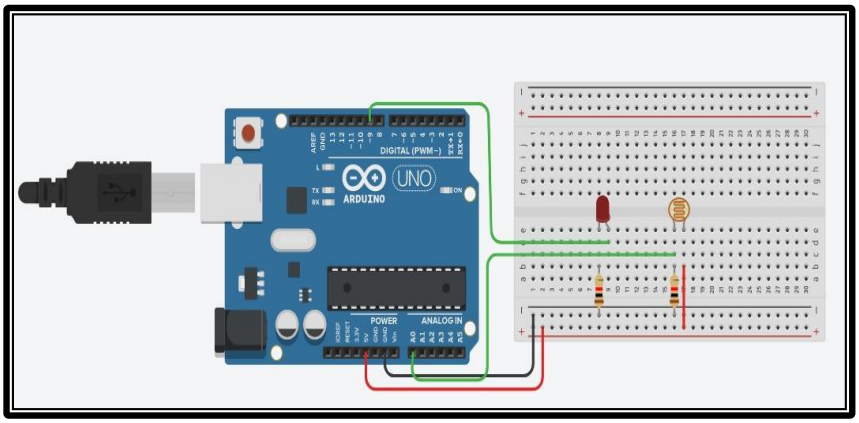


Fig 7.1 Circuit Diagram of LDR

CODE:

```
void setup()
{
pinMode(9, OUTPUT);                    // pin 9 set as output
pinMode(A0,INPUT);                     // pin A0 set as input
Serial.begin(9600);
}
void loop()
{
int y= analogRead(A0);                 // value from LDR stored in variable 'y'
digitalWrite(9,y/4);
Serial.println(y/4);
}
```

Aim of the Experiment: Interfacing of DC motor with Arduino MKR1000. Write a program to rotate the motor in clockwise and anticlockwise direction with using a delay of 2 sec.

Components Used: Arduino MKR1000, DC Motor, L293D IC, Bread-Board, Wires

Working:

L293D is a motor controlling integrated circuit that is used for rotating the motor in clockwise and anti-clockwise direction without reversing the voltage.

- ❖ It works on the concept of H-bridge. H-bridge is a circuit which allows the voltage to be flown in either direction.
- ❖ In a single L293D chip there are two h-Bridge circuit inside the IC which can rotate two dc motor independently.
- ❖ There are two Enable pins on l293d. Pin 1 and pin 9, for being able to drive the motor, the pin 1 and 9 need to be high.
- ❖ If anyone of the either pin1 or pin9 goes low then the motor in the corresponding section will suspend working. It's like a switch.

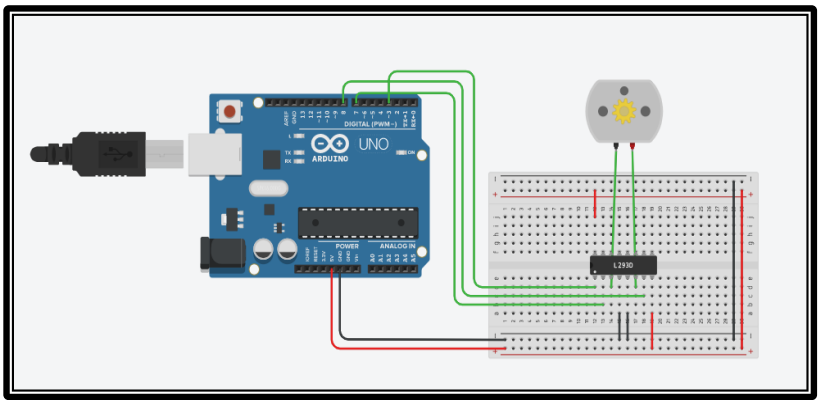


Fig 8.1 Circuit Diagram

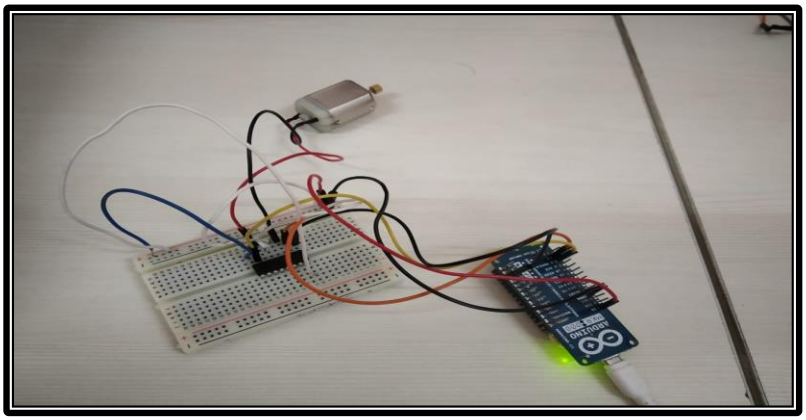


Fig 8.2 Motor Connections

CODE:

```
int in1 = 8;                          // l293d pins defined
int in2 = 7;
int enA = 3;
void setup()
{                                     // set all pins of led to output
  pinMode(enA, OUTPUT);
  pinMode(in1, OUTPUT);
  pinMode(in2, OUTPUT);
}
void loop()
{ analogWrite(enA, 255);
  digitalWrite(in1, HIGH);            // set in1 to HIGH
  digitalWrite(in2, LOW);             // set in2 to LOW
 delay(2000);
  digitalWrite(in1, LOW);             // now reverse the configuration
  digitalWrite(in2, HIGH);
 delay(2000);

}
```

Aim of the Experiment: Interfacing of a display device, i.e., LCD x2 with Arduino MKR1000. Write a program to display “HELLO IOT” on LCD.

Components Used: Arduino MKR1000, Resistor, LCD, Bread-Board, Wires

Working:

In this experiment we will use a Liquid Crystal Display(LCD) to display the message on the message on the screen.

- The size of the led screen used is 16*2 i.e. can display 16 columns of letters and 2 rows.
- For varying the contrast of the screen a potentiometer is attached to the LCD screen.
- There are 16 pins in the Lcd out of which pin 1,15 are grounded and 2,16 are connected to the 5V input.

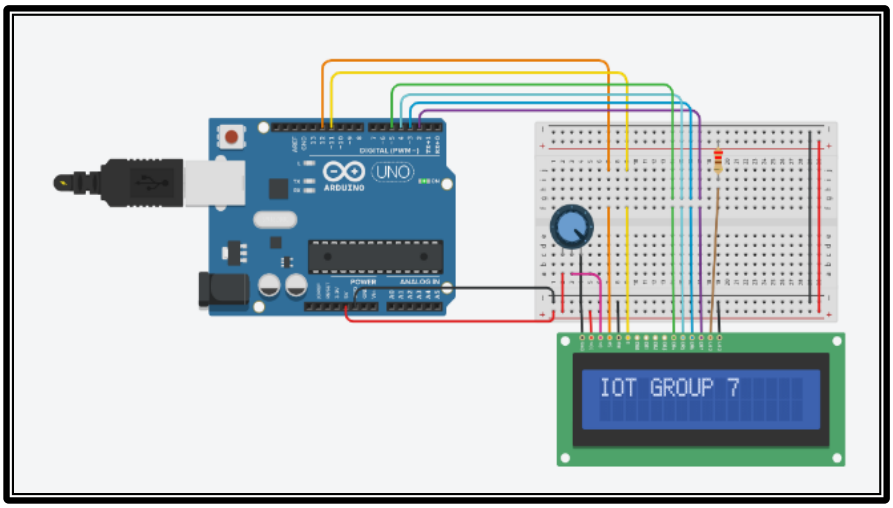


Fig 9.1 Circuit Diagram

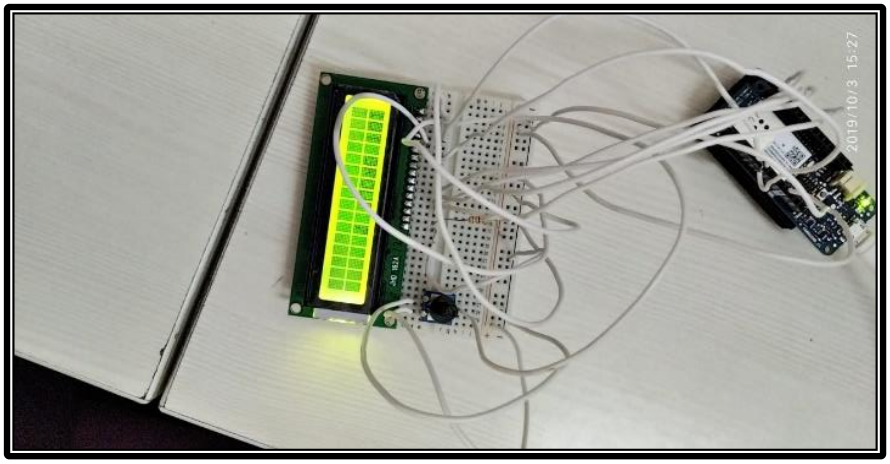


Fig 9.2 IOT GROUP 7 displayed on LCD

CODE:

```
#include <LiquidCrystal.h>
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3,     // define all the LED pins
d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

void setup()
{
lcd.begin(16, 2);                                       // set up the LCD's number of columns and rows:
lcd.print("IOT GROUP 7");                                // Print a message to the LCD.
}
void loop()
{
lcd.noDisplay();                                        // Turn off the display:
delay(500);
lcd.display();                                          // Turn on the display:
delay(5000);
}
```

Aim of the Experiment: Write down the steps to connect Arduino MKR1000 with IoT cloud. Write a program to blink an LED using Arduino IoT cloud.

Components Used: Arduino MKR1000, Resistor, LED, Bread-Board, Wires

Working:

In this experiment we will switch the LED ON/OFF using the cloud services.

An online switch is created on the cloud that will control the led.

- Connect the LED's cathode to the ground of MKR and anode to output pin through a resistor.
- Turn the LED on and off with the use of MKR1000 on the cloud.
- When we toggle the ON/OFF button on the cloud the led performs the function accordingly.

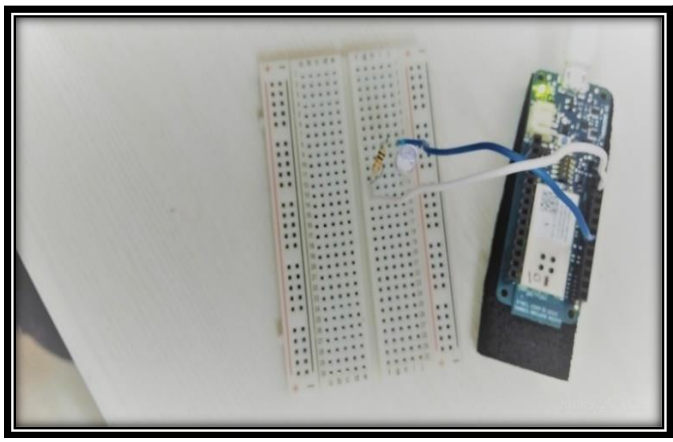


Fig 10.1 Circuit Diagram

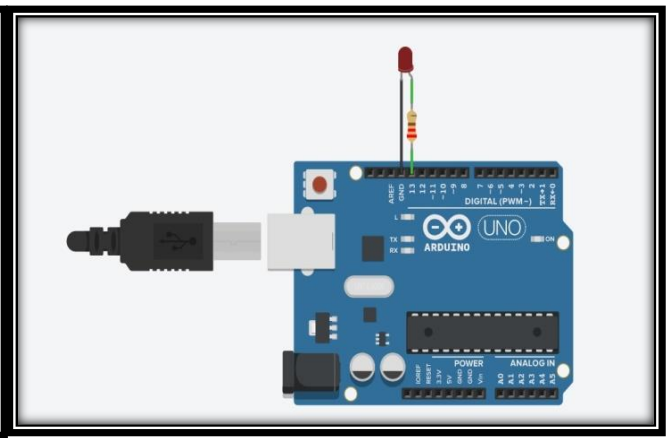
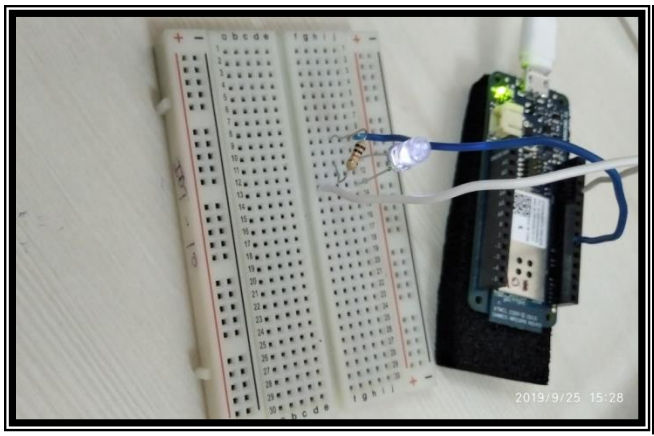


Fig 10.2 LED is OFF



. Fig 10.3 LED is ON

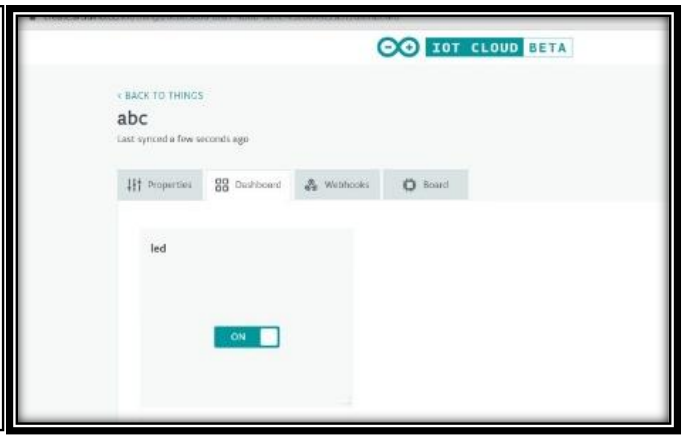


Fig. 10.4 Displayed result Serial Monitor

CODE:

```
#include "thingProperties.h"
void setup() {
Serial.begin(9600);                                    // Initialize serial and wait for port to open:
pinMode(10,OUTPUT);
delay(1500);
initProperties();                                      // Defined in thingProperties.h
ArduinoCloud.begin(ArduinoIoTPreferredConnection);     // Connect to Arduino IoT Cloud
setDebugMessageLevel(2);
ArduinoCloud.printDebugInfo();
}
void loop() {
ArduinoCloud.update();
}
void onLedChange() {
digitalWrite(10,led);
}
```

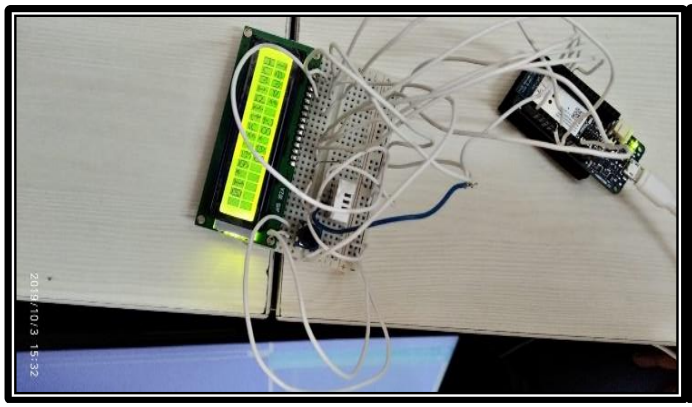
Aim of the Experiment: Write a program to visualize data of temperature sensor on Arduino IoT cloud.

Components Used: Arduino MKR1000, Potentiometer, LCD, DHT11 Sensor , Bread-Board, Resistor, Wires

Working:

DHT11 sensor is a electronic sensor that is used to sense the humidity and the temperature of the room in which it is present.

- ❖ It consists of a capacitive humidity sensing element and a thermistor for sensing temperature.
- ❖ The temperature range of DHT11 is from 0 to 50 degree Celsius with a 2-degree accuracy.
- ❖ Humidity range of this sensor is from 20 to 80% with 5% accuracy.
- ❖ DHT11 is small in size with operating voltage from 3 to 5 volts. The maximum current used while measuring is 2.5mA.



. Fig 11.1 Less Humidity and Room

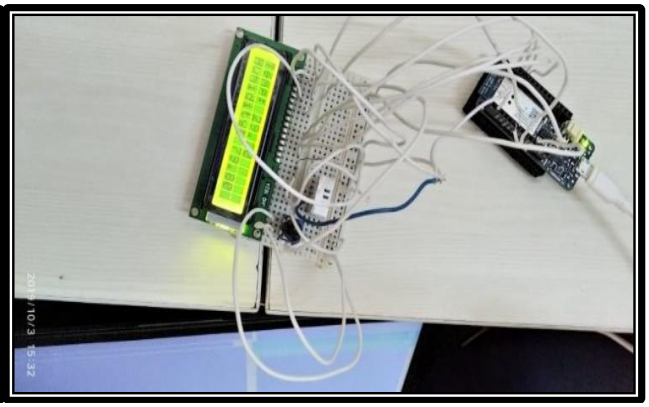


Fig 11.2 High Humidity on blowing air Temperature

Fig 11.3 Displaying values of humidity and temperature on the cloud

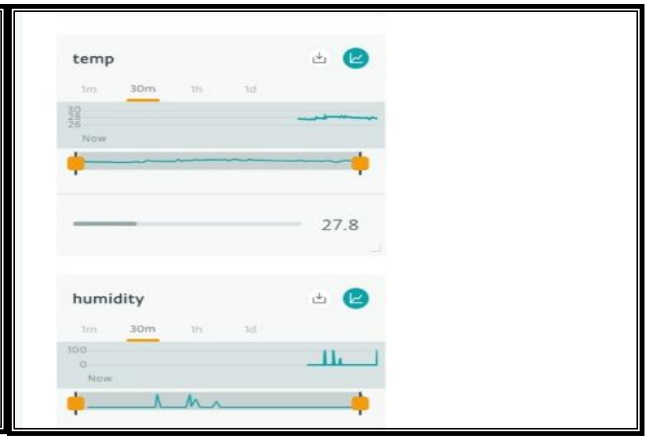


Fig 11.4 Graphs of Humidity and temperature on the cloud

CODE:

```
#include "arduino_secrets.h"
#include <DHT.h>                                        // DHT sensor library included
#include <DHT_U.h>
#include "thingProperties.h"
#define DHTPIN 7
#define DHTTYPE DHT22
DHT dht(DHTPIN, DHTTYPE);
float humidity;
float temp;
void setup()
{
Serial.begin(9600);                                     // Initialize serial and wait for port to open
dht.begin();
delay(1500);
initProperties();                                       // Defined in thingProperties.h
ArduinoCloud.begin(ArduinoIoTPreferredConnection);      // Connect to Arduino IoT Cloud
setDebugMessageLevel(2);
ArduinoCloud.printDebugInfo();
}
void loop()
{
ArduinoCloud.update();
temp = dht.readTemperature();                           //  temperature value from sensor updated on
                                                        // cloud
humidity = dht.readHumidity();                          //  humidity value from sensor updated on
                                                        //cloud
}
```

Aim of the Experiment: Design an interface to control the speed of DC motor using cloud services. Store and visualize the data of speed of DC motor after every one sec.

Components Used: Arduino MKR1000, DC Motor, L293D IC, Bread-Board, Wires

Working:

L293D is a motor controlling integrated circuit that is used for rotating the motor in clockwise and anti-clockwise direction without reversing the voltage.

- ❖ It works on the concept of H-bridge. H-bridge is a circuit which allows the voltage to be flown in either direction.
- ❖ In a single L293D chip there are two h-Bridge circuit inside the IC which can rotate two dc motor independently.
- ❖ There are two Enable pins on l293d. Pin 1 and pin 9, for being able to drive the motor, the pin 1 and 9 need to be high.
- ❖ If anyone of the either pin1 or pin9 goes low then the motor in the corresponding section will suspend working. It's like a switch

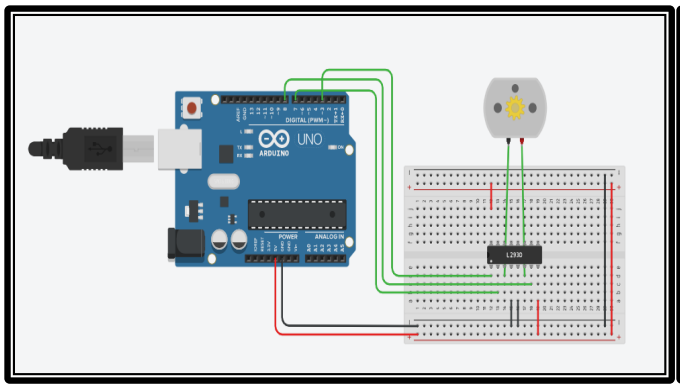


Fig 12.1 Circuit Diagram

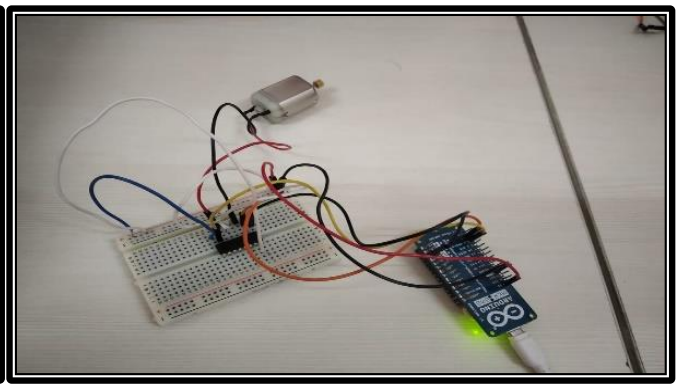


Fig 12.2 Actual Circuit

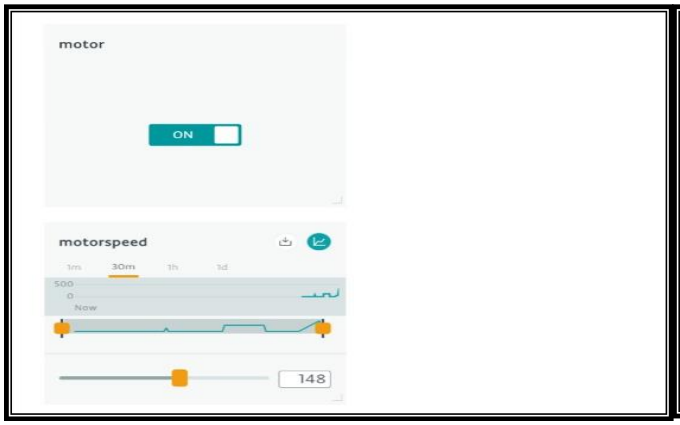


Fig 12.3 Switch to control motor on the Cloud and graph

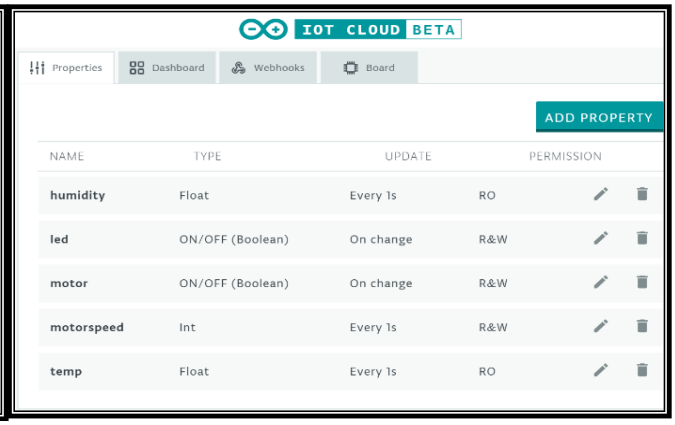


Fig 12.4 Properties of motor added on the cloud

CODE:

```
#include "arduino_secrets.h"
int motorspeed;
#include "thingProperties.h"                          // Defined thingProperties.h

#define en 2 //motor enable pin
#define ledpin 8 //led pin
void setup()
{
Serial.begin(9600);                                   // Initialize serial and wait for port to open
pinMode(en,OUTPUT);                                   // all IC pins set to Output
pinMode(3,OUTPUT);
pinMode(4,OUTPUT);
digitalWrite(3,0);
digitalWrite(4,0);
initProperties();                                     // Defined in thingProperties.h
ArduinoCloud.begin(ArduinoIoTPreferredConnection);    // Connect to Arduino IoT Cloud
setDebugMessageLevel(2);
ArduinoCloud.printDebugInfo();
}
void loop() {
ArduinoCloud.update();
}
void onMotorspeedChange() {
analogWrite(en,motorspeed);                           //motor speed is set here
}
void onMotorChange() {
digitalWrite(3,motor);                                // rotation of motor is set here
digitalWrite(4,!motor);
}

void onMotordirChange() {
}
```