|  |  |
| --- | --- |
| **Ex No : 08**  **Date:** | **IMPLEMENT LED BLINK AND LED PATTERN WITH ARDUINO** |

**Aim:**

To Implement LED Blink and LED Pattern With Arduino

**Procedure:**

**STEP 1:** Open Arduino Software.

**STEP 2:** Click Tools -> Board -> Arduino Nano.

**STEP 3:** In Tools Menu Select Port –> COM.

**STEP 4:** Click File-> New and Start Write the Code.

**STEP 5:** And Click tick Icon Button of the file Menu to compile.

**STEP 6:** Click Upload Icon After Verify.

**Program: LED Blink**

#include <LiquidCrystal.h>

LiquidCrystal lcd(A5,A4,A3,A2,A1,A0);

int i;

void setup() {

// set up the LCD's number of columns and rows:

lcd.begin(16, 2);

lcd.print("LED BLINK");

lcd.setCursor(0,1);

lcd.print("EXPERIMENT");

delay(3000);

}

void led\_blink()

{

lcd.clear();

lcd.print("LED Blinking");

pinMode(5,OUTPUT);

for(int i=0;i<10;i++)

{

digitalWrite(1,HIGH);

delay(50);

}

}

void loop() {

led\_blink();

delay(100);

}

**Program: LED Pattern**

#include <LiquidCrystal.h>

LiquidCrystal lcd(A5,A4,A3,A2,A1,A0);

int i;

void setup() {

lcd.begin(16, 2);

lcd.print("LED PATTERN ");

lcd.setCursor(0,1);

lcd.print("Experimenter");

delay(3000);

lcd.clear();

}

void ledPattern()

{

lcd.clear();

lcd.print("LED Pattern");

for (i =2;i<=7;i++)

{

pinMode(i, OUTPUT); // turn the LED on (HIGH is the voltage level)

digitalWrite(i, HIGH);

}

for(int j=0;j<5;j++)

{

for (i =2;i<=7;i++)

{

digitalWrite(i, LOW);

delay(100);

}

for (i =7;i>=2;i--)

{

digitalWrite(i, HIGH);

delay(100);

}

}

for (i =2;i<=7;i++)

digitalWrite(i, HIGH);

for (i =2;i<=7;i++)

pinMode(i, INPUT);

}

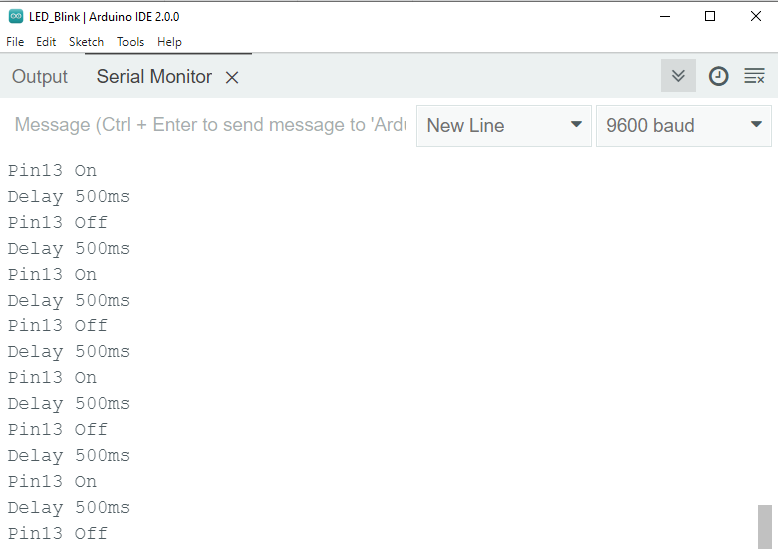
void loop() {

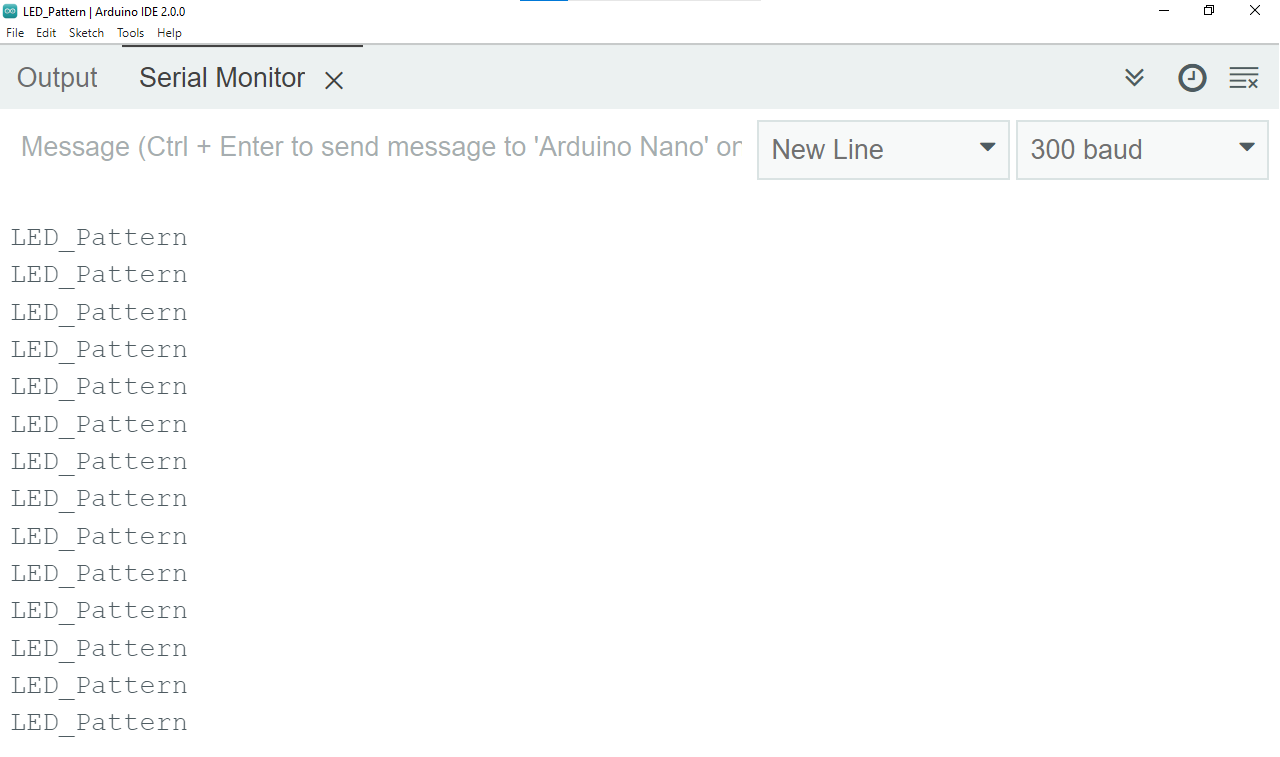
ledPattern();

delay(1000);

}

**Output:**

****

****

**Result:**

Thus the above LED Blink and LED Pattern with Arduino program was executed and verified successfully

|  |  |
| --- | --- |
| **Ex No : 09**  **Date:** | **IMPLEMENT LED PATTERN WITH PUSH BUTTON CONTROL WITH ARDUINO** |

**Aim:**

Implement LED Pattern with Push Button Control with Arduino

**Procedure:**

**STEP 1:** Open Arduino Software.

**STEP 2:** Click Tools -> Board -> Arduino Nano.

**STEP 3:** In Tools Menu Select Port –> COM.

**STEP 4:** Click File-> New and Start Write the Code.

**STEP 5:** And Click tick Icon Button of the file Menu to compile.

**STEP 6:** Click Upload Icon After Verify.

**Program:**

#include <LiquidCrystal.h>

LiquidCrystal lcd(A5,A4,A3,A2,A1,A0);

int i;

void setup() {

lcd.begin(16, 2);

lcd.print("LED Pattern");

lcd.setCursor(0,1);

lcd.print("Experimenter");

delay(3000);

lcd.clear();

}

void pushbutton\_led()

{

lcd.clear();

lcd.print("Push Button");

lcd.setCursor(0,1);

lcd.print("Press Key6");

pinMode(7, INPUT); // turn the LED on (HIGH is the voltage level)

digitalWrite(7, HIGH);

for (i =2;i<=6;i++)

{

pinMode(i, OUTPUT); // turn the LED on (HIGH is the voltage level)

digitalWrite(i, HIGH);

}

while(digitalRead(7));

if(!digitalRead(7))

{

lcd.clear();

lcd.print("Key6 Pressed");

for (i =2;i<=6;i++)

{

digitalWrite(i, LOW);

delay(100);

}

for (i =6;i>=2;i--)

{

digitalWrite(i, HIGH);

delay(100);

}

for (i =2;i<=6;i++)

digitalWrite(i, HIGH);

}

}

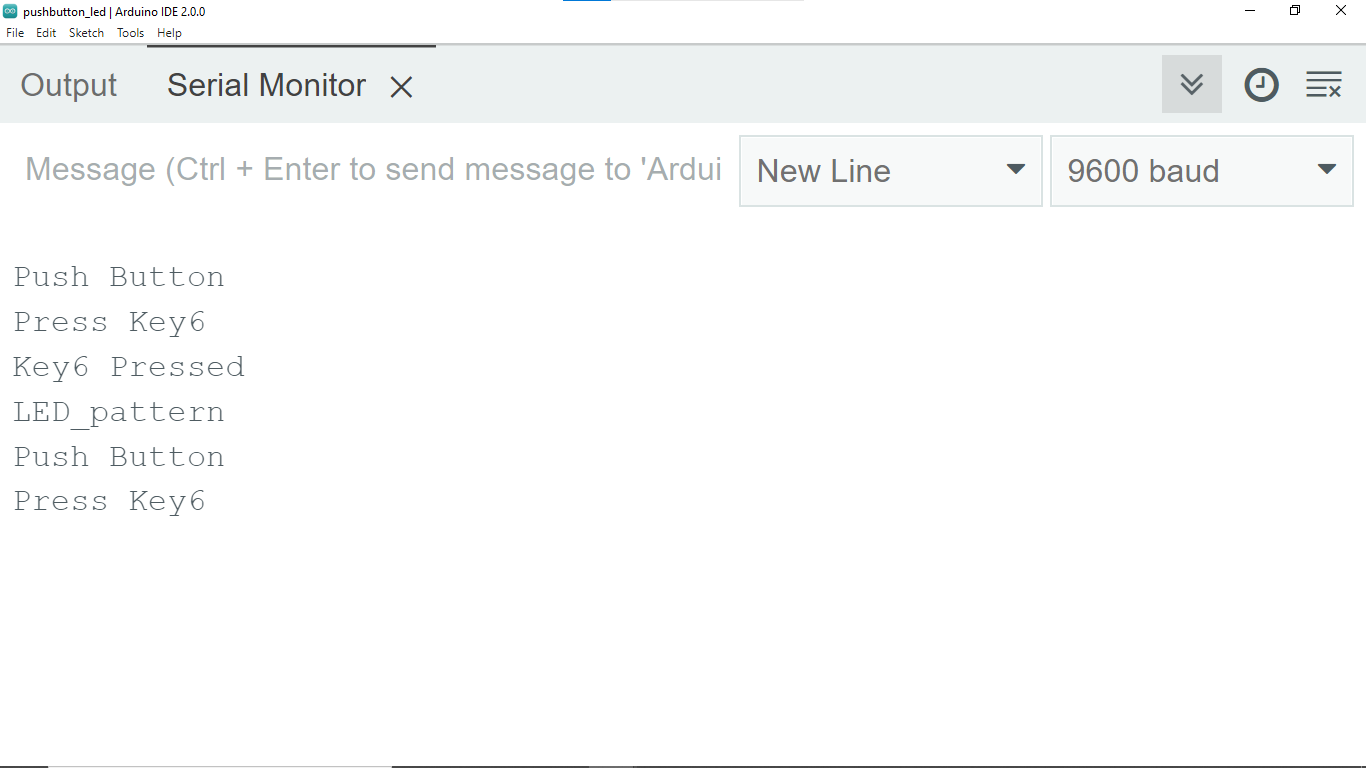
void loop() {

pushbutton\_led();

delay(1000);

}

**Output:**

****

**Result:**

Thus the above LED Pattern with Push Button Control with Arduino program was executed and verified successfully

|  |  |
| --- | --- |
| **Ex No : 10**  **Date:** | **DISPLAY “HELLO WORLD” IN LCD 16X2 DISPLAY WITH ARDUINO** |

**Aim:**

To Display “Hello World” in LCD 16X2 Display with Arduino

**Procedure:**

**STEP 1:** Open Arduino Software.

**STEP 2:** Click Tools -> Board -> Arduino Nano.

**STEP 3:** In Tools Menu Select Port –> COM.

**STEP 4:** Click File-> New and Start Write the Code.

**STEP 5:** And Click tick Icon Button of the file Menu to compile.

**STEP 6:** Click Upload Icon After Verify.

**Program:**

#include <LiquidCrystal.h>

LiquidCrystal lcd(A5,A4,A3,A2,A1,A0);

int i;

void setup() {

lcd.begin(16, 2);

lcd.print("TEXT DISPLAY");

lcd.setCursor(0,1);

lcd.print("Experimenter");

delay(3000);

lcd.clear();

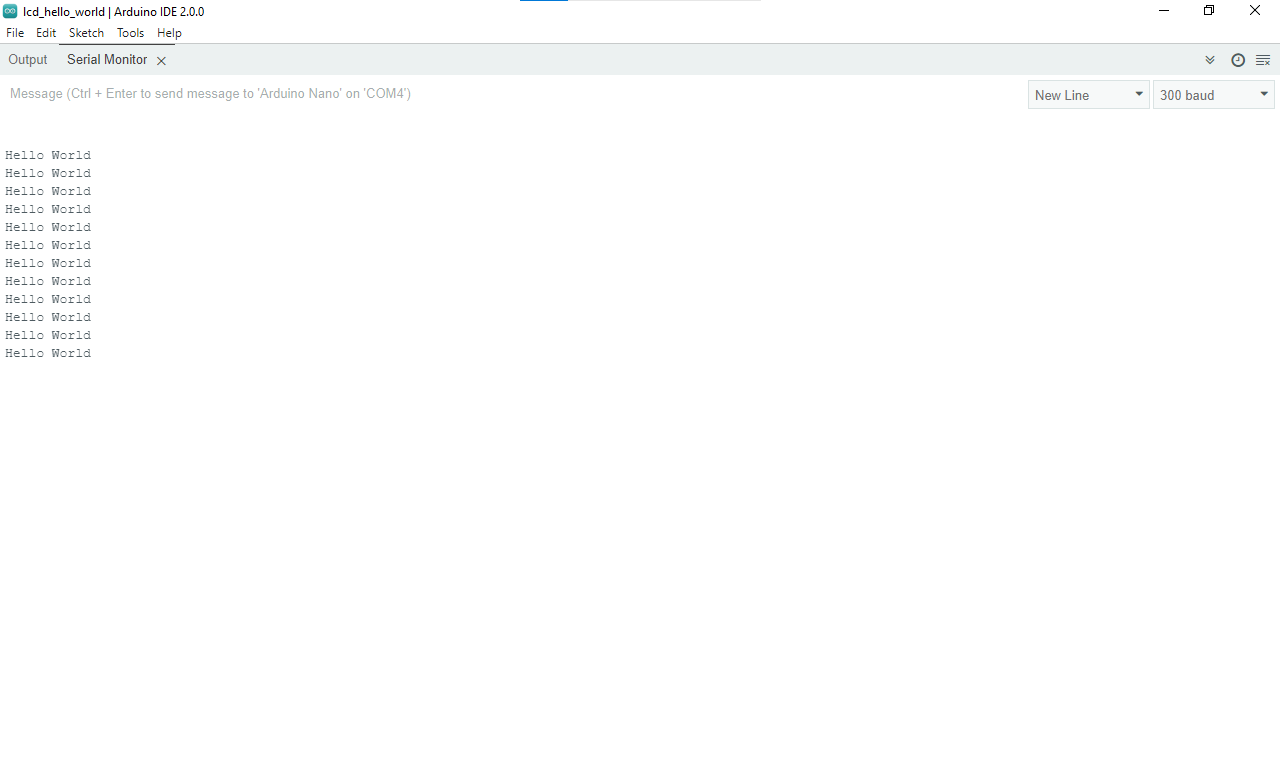
lcd.print("HELLO WORLD");

}

void loop() {

}

**Output:**

****

**Result:**

Thus the above Display “Hello World” in LCD 16X2 with Arduino program was executed and verified successfully

|  |  |
| --- | --- |
| **Ex No : 11**  **Date:** | **IMPLEMENT THE SERVO MOTOR CONTROL WITH ARDUINO** |

**Aim:**

To Implement the Servo Motor Control with Arduino

**Procedure:**

**STEP 1:** Open Arduino Software.

**STEP 2:** Click Tools -> Board -> Arduino Nano.

**STEP 3:** In Tools Menu Select Port –> COM.

**STEP 4:** Click File-> New and Start Write the Code.

**STEP 5:** And Click tick Icon Button of the file Menu to compile.

**STEP 6:** Click Upload Icon After Verify.

**Program:**

#include <LiquidCrystal.h>

#include <Servo.h>

Servo myservo;

LiquidCrystal lcd(A5,A4,A3,A2,A1,A0);

int i;

void setup() {

lcd.begin(16, 2);

lcd.print("ROTATE SERVO");

lcd.setCursor(0,1);

lcd.print("Experimenter");

delay(3000);

lcd.clear();

lcd.print("START ROTATING");

delay(1000);

lcd.clear();

}

void servoSweep()

{

lcd.clear();

lcd.print("MOTER DEGREE");

myservo.attach(11);

int pos;

for(int i=0; i<2;i++)

{

for (pos = 0; pos <= 90; pos += 1) { // goes from 0 degrees to 180 degrees

// in steps of 1 degree

myservo.write(pos); // tell servo to go to position in variable 'pos'

delay(15); // waits 15ms for the servo to reach the position

lcd.setCursor(0,1);

lcd.print(" ");

lcd.setCursor(0,1);

lcd.print(pos);

}

for (pos = 130; pos >= 0; pos -= 1) { // goes from 180 degrees to 0 degrees

myservo.write(pos); // tell servo to go to position in variable 'pos'

delay(15); // waits 15ms for the servo to reach the position

lcd.setCursor(0,1);

lcd.print(" ");

lcd.setCursor(0,1);

lcd.print(pos);

}

}

}

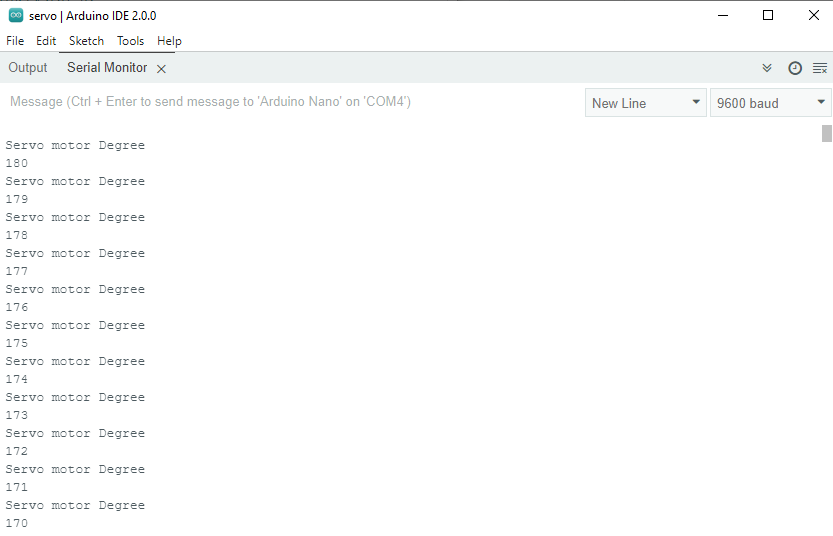
void loop() {

servoSweep();

delay(1000);

}

**Output:**

****

**Result:**

Thus the above Implement the Servo Motor Control with Arduino program was executed and verified successfully

|  |  |
| --- | --- |
| **Ex No : 12**  **Date:** | **IMPLEMENT AND MONITOR THE LM35 TEMPERATURE SENSOR AND ULTRASONIC DISTANCE MEASUREMENT WITH ARDUINO** |

**Aim:**

To Implement and Monitor the LM35 Temperature Sensor and Ultrasonic Distance Measurement With Arduino

**Procedure:**

**STEP 1:** Open Arduino Software.

**STEP 2:** Click Tools -> Board -> Arduino Nano.

**STEP 3:** In Tools Menu Select Port –> COM.

**STEP 4:** Click File-> New and Start Write the Code.

**STEP 5:** And Click tick Icon Button of the file Menu to compile.

**STEP 6:** Click Upload Icon After Verify.

**Program for LM35:**

#include <LiquidCrystal.h>

LiquidCrystal lcd(A5,A4,A3,A2,A1,A0);

int i;

void setup() {

lcd.begin(16, 2);

lcd.print("Find Temperature");

lcd.setCursor(0,1);

lcd.print("Experimenter");

delay(3000);

lcd.clear();

}

void temperature()

{

int val;

float mv;

float cel;

float farh;

for(int i=0;i<20;i++)

{

lcd.clear();

lcd.print("Temperature");

val = analogRead(A6);

mv = ( val/1024.0)\*5000;

cel = mv/10;

farh = (cel\*9)/5 + 32;

lcd.setCursor(0,1);

lcd.print(cel);

lcd.print("\*C");

delay(1000);

}

}

void loop() {

temperature();

delay(1000);

}

**Program for Ultrasonic (SR04):**

#include <LiquidCrystal.h>

LiquidCrystal lcd(A5,A4,A3,A2,A1,A0);

int i;

void setup() {

lcd.begin(16, 2);

lcd.print("Finding Distance");

lcd.setCursor(0,1);

lcd.print("Experimenter");

delay(3000);

lcd.clear();

}

void sr04()

{

pinMode(13, OUTPUT);

pinMode(12, INPUT);

for(int i=0;i<100;i++)

{

lcd.clear();

lcd.print("UltraSonic");

lcd.setCursor(0,1);

digitalWrite(13, LOW);

delayMicroseconds(2);

digitalWrite(13, HIGH);

delayMicroseconds(10);

digitalWrite(13, LOW);

long duration = pulseIn(12, HIGH);

int distance = duration \* 0.034 / 2; // Speed of sound wave divided by 2 (go and back)

lcd.print("Distance:");

lcd.print(distance);

lcd.print("cm");

delay(100);

}

}

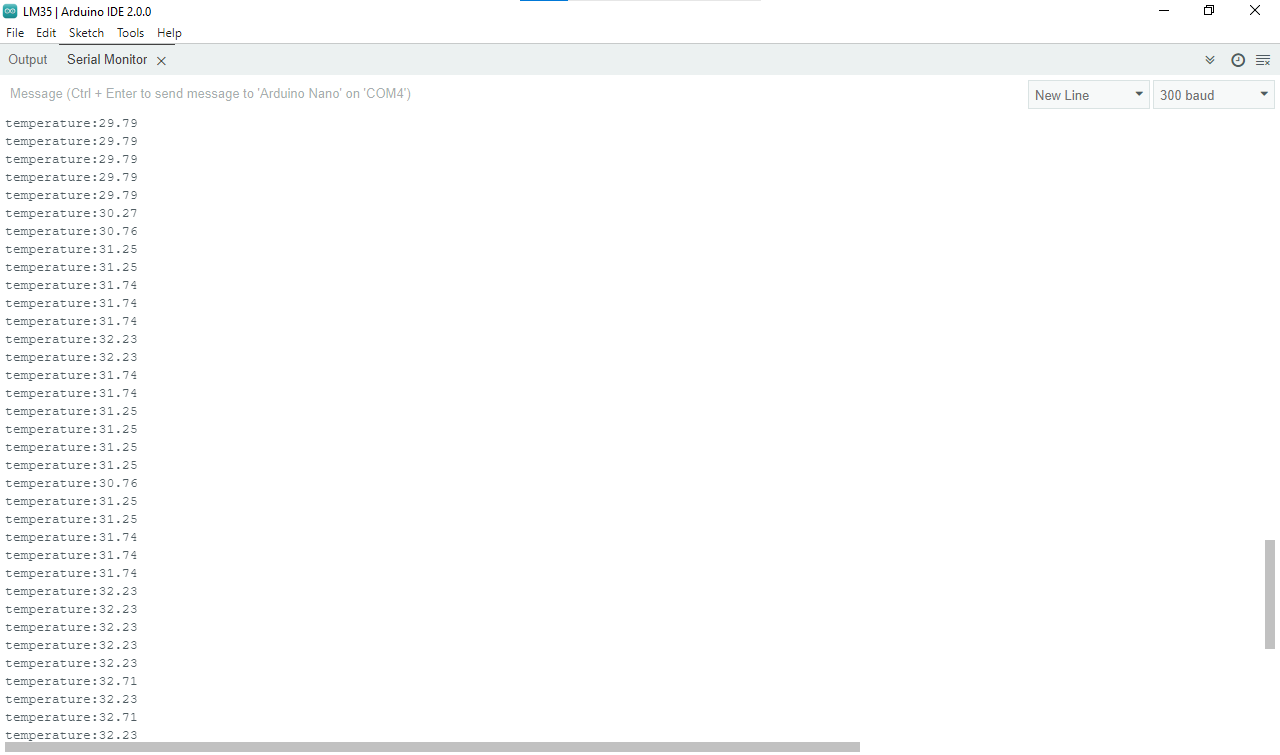
void loop() {

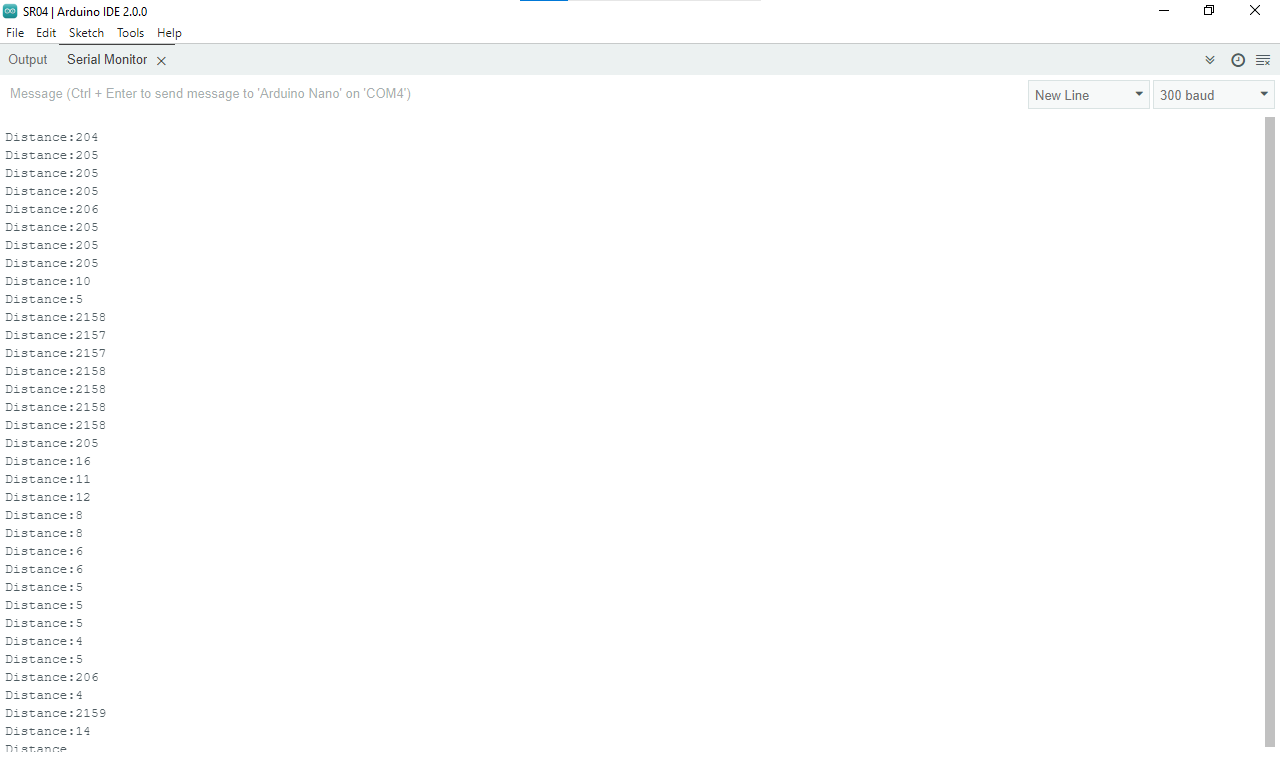
sr04();

delay(1000);

}

**Output:**

****

****

**Result:**

Thus the above Implement and Monitor the LM35 Temperature Sensor and Ultrasonic Distance Measurement with Arduino Program was executed and verified successfully

|  |  |
| --- | --- |
| **Ex No : 13**  **Date:** | **IMPLEMENT THE IR SENSOR ANALOG INPUT WITH ARDUINO** |

**Aim:**

To Implement the IR Sensor Analog Input with Arduino

**Procedure:**

**STEP 1:** Open Arduino Software.

**STEP 2:** Click Tools -> Board -> Arduino Nano.

**STEP 3:** In Tools Menu Select Port –> COM.

**STEP 4:** Click File-> New and Start Write the Code.

**STEP 5:** And Click tick Icon Button of the file Menu to compile.

**STEP 6:** Click Upload Icon After Verify.

**Program:**

#include <LiquidCrystal.h>

LiquidCrystal lcd(A5,A4,A3,A2,A1,A0);

int i;

void setup() {

lcd.begin(16, 2);

lcd.print("IR INPUT");

lcd.setCursor(0,1);

lcd.print("Experimenter");

delay(3000);

lcd.clear();

}

void ir\_sensor()

{

pinMode(10,INPUT);

pinMode(2,OUTPUT);

digitalWrite(2,HIGH);

for(int i=0;i<100;i++)

{

if(!digitalRead(10))

{

lcd.clear();

lcd.print("IR Triggered");

digitalWrite(2,LOW);

}

else

{

lcd.clear();

lcd.print("IR Not Triggered");

digitalWrite(2,HIGH);

}

delay(100);

}

}

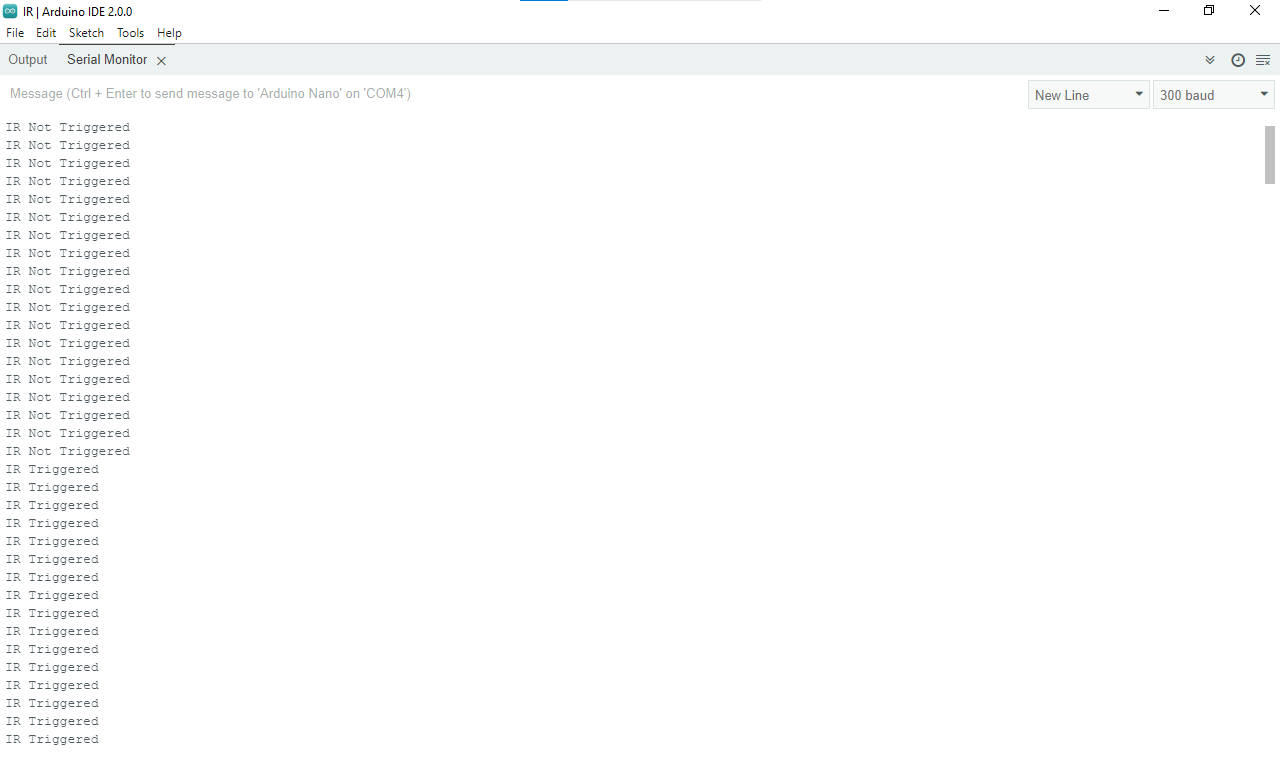
void loop() {

ir\_sensor();

delay(1000);

}

**Output:**

****

**Result:**

Thus the above Implement the IR Sensor Analog Input with Arduino program was executed and verified successfully

|  |  |
| --- | --- |
| **Ex No : 14**  **Date:** | **USING THINKSPEAK CLOUD READING TEMPERATURE SENSOR MONITORING WITH NODEMCU** |

**Aim:**

To Using ThinkSpeak Cloud Reading Temperature Sensor Monitoring with NodeMCU

**Procedure:**

**STEP 1:** Open Arduino Software.

**STEP 2:** Click Tools -> Board -> NodeMcu.

**STEP 3:** In Tools Menu Select Port –> COM.

**STEP 4:** Click File-> New and Start Write the Code.

**STEP 5:** Login Your Think speak Cloud and check API key and Channel Number.

**STEP 6:** Open secrets.h Header file.

**STEP 7:** Edit SSID &Password of the WIFI and add Your API key and Channel Number of Think speak cloud

**STEP 8:** And Click tick Icon Button of the file Menu to compile.

**STEP 9:** Click Upload Icon After Verify.

**STEP 10:** Now, Check you think speak cloud temperature will sense as Graph

**Program:**

**Secrets.h**

#define SECRET\_SSID "MySSID" // replace MySSID with your WiFi network name

#define SECRET\_PASS "MyPassword" // replace MyPassword with your WiFi password

#define SECRET\_CH\_ID 0000000 // replace 0000000 with your channel number

#define SECRET\_WRITE\_APIKEY "XYZ" // replace XYZ with your channel write API Key

**NodeMcu.ino**

#include <ESP8266WiFi.h>

#include "secrets.h"

#include "ThingSpeak.h" // always include thingspeak header file after other header files and custom macros

#include <SoftwareSerial.h>

SoftwareSerial swSer(D6, D7);

char ssid[] = SECRET\_SSID; // your network SSID (name)

char pass[] = SECRET\_PASS; // your network password

int keyIndex = 0; // your network key Index number (needed only for WEP)

WiFiClient client;

unsigned long myChannelNumber = SECRET\_CH\_ID;

const char \* myWriteAPIKey = SECRET\_WRITE\_APIKEY;

// Initialize our values

float number1 = 0;

float number2 = 0;

float number3 = 0;

float number4 = 0;

float number5 = 0;

float number6 = 0;

float number7 = 0;

float number8 = 0;

String myStatus = "";

String response;

int ESPwait(String stopstr, int timeout\_secs)

{

bool found = false;

char c;

long timer\_init;

long timer;

response="";

timer\_init = millis();

while (!found) {

timer = millis();

if (((timer - timer\_init) / 1000) > timeout\_secs) { // Timeout?

Serial.println("!Timeout!");

return 0; // timeout

}

if (swSer.available()) {

c = swSer.read();

//Serial.print(c);

response += c;

if (response.endsWith(stopstr)) {

found = true;

delay(10);

swSer.flush();

Serial.flush();

Serial.println();

}

} // end Serial1\_available()

} // end while (!found)

return 1;

}

int ESPwait1(String stopstr, int timeout\_secs)

{

bool found = false;

char c;

long timer\_init;

long timer;

response="";

timer\_init = millis();

while (!found) {

timer = millis();

if (((timer - timer\_init) / 1000) > timeout\_secs) { // Timeout?

Serial.println("!Timeout!");

return 0; // timeout

}

if (Serial.available()) {

c = Serial.read();

Serial.print(c);

response += c;

if (response.endsWith(stopstr)) {

found = true;

delay(10);

Serial.flush();

Serial.println();

}

} // end Serial1\_available()

} // end while (!found)

return 1;

}

void setup() {

Serial.begin(9600); // Initialize serial

swSer.begin(9600);

pinMode(LED\_BUILTIN,OUTPUT);

while (!Serial) {

; // wait for serial port to connect. Needed for Leonardo native USB port only

}

WiFi.mode(WIFI\_STA);

ThingSpeak.begin(client); // Initialize ThingSpeak

}

void loop() {

char c;

// Connect or reconnect to WiFi

if(WiFi.status() != WL\_CONNECTED){

Serial.print("Attempting to connect to SSID: ");

Serial.println(SECRET\_SSID);

while(WiFi.status() != WL\_CONNECTED){

WiFi.begin(ssid, pass); // Connect to WPA/WPA2 network. Change this line if using open or WEP network

Serial.print(".");

delay(5000);

}

Serial.println("\nConnected.");

for(int i=0;i<5;i++)

{

digitalWrite(LED\_BUILTIN,LOW);

delay(100);

digitalWrite(LED\_BUILTIN,HIGH);

delay(100);

}

}

if(swSer.available())

{

c=swSer.read();

swSer.println(c);

if(c=='\*')

{

if(ESPwait("#",3))

{

char \* strtokIndx;

response.remove(response.length()-1);

strtokIndx = strtok(const\_cast<char\*>(response.c\_str()),","); // get the first part - the string

number1 = atof(strtokIndx);

Serial.println(response);

response="";

ThingSpeak.setField(1, number1);

int x = ThingSpeak.writeField(myChannelNumber, 1, number1, myWriteAPIKey);

if(x == 200)

{

Serial.println("Channel update successful.");

digitalWrite(LED\_BUILTIN,LOW);

}

else{

Serial.println("Problem updating channel. HTTP error code " + String(x));

digitalWrite(LED\_BUILTIN,HIGH);

}

Serial.println("Sent");

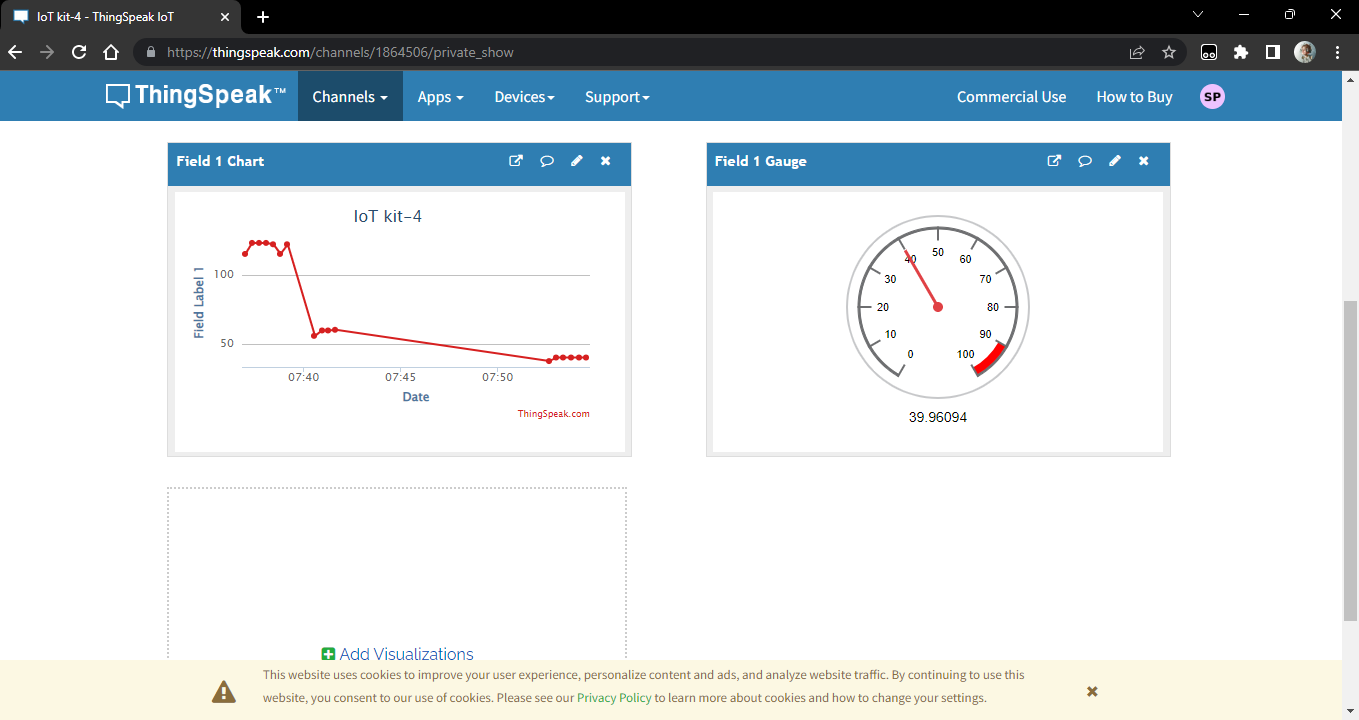
}

}

}

}

**Output:**

****

**Result:**

Thus the above ThinkSpeak Cloud Reading Temperature Sensor Monitoring with NodeMCU was executed successfully