Introduction:

A room temperature monitor which is made using an Arduino board. The circuit is a combination of an Arduino UNO board,. It will display the temperature on the LCD screen in degree Centigrade. It can be powered from any standard Arduino AC mains adaptor (9VDC), or from a suitable battery. First of all note that this circuit can not measure the temperature of the whole city or province.It can only measure the temperature of the surroundings or it is worth noting that the sensitivity range is low for this perticuler project.After the increasing of the temperature a alarm and a fan will be automatically turned on.

Objectives:

* Checking the accuracy of temperature control and monitoring devices.
* Qualification of temperature-controlled storage areas.
* Temperature monitoring systems for transport operations.
* Adjusting the room temperature.
* Giving alarm about the excessive heat.
* Securing people by reducing the temperature.

Hardware and Software Requirements:

The following is the list of all hardware parts will need for this project

* The Arduino Uno Board
* LM35 te0mperature sensor
* 16×2 Hitachi compatible LCD
* Potentiometer
* Push button
* Buzzer
* L293D Display
* Bread Board
* Connecting Wires
* DC Motor.

On the software side, we will need the following:

* The Arduino IDE
* The Arduino LiquidCrystal library
* Fritzing or Proteus for circuit diagram

Configuring the hardware:

We will now build the hardware part of the project. The following is a schematic of the project:

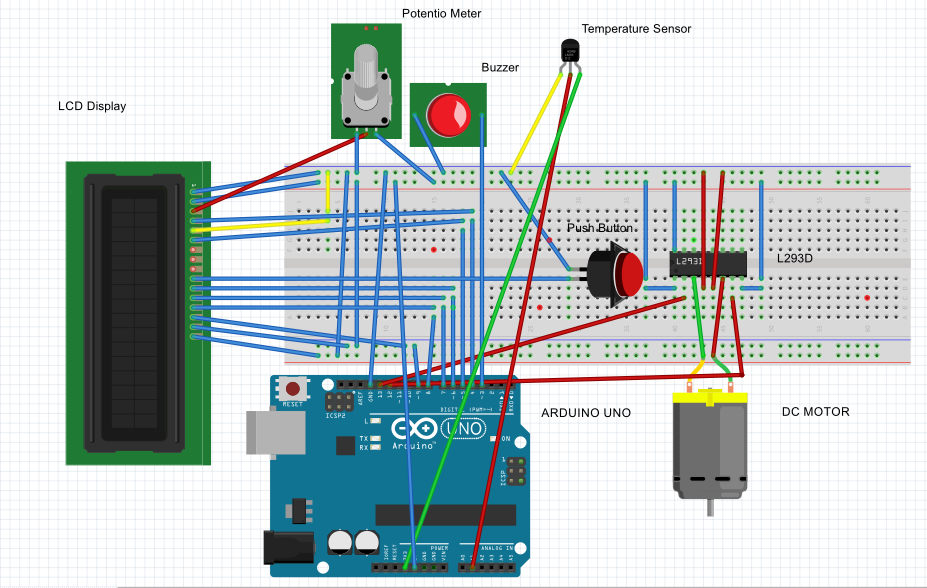


Fig-1:Connecting arduino uno, display module, temperature sensor,motor,L293D,buzzer

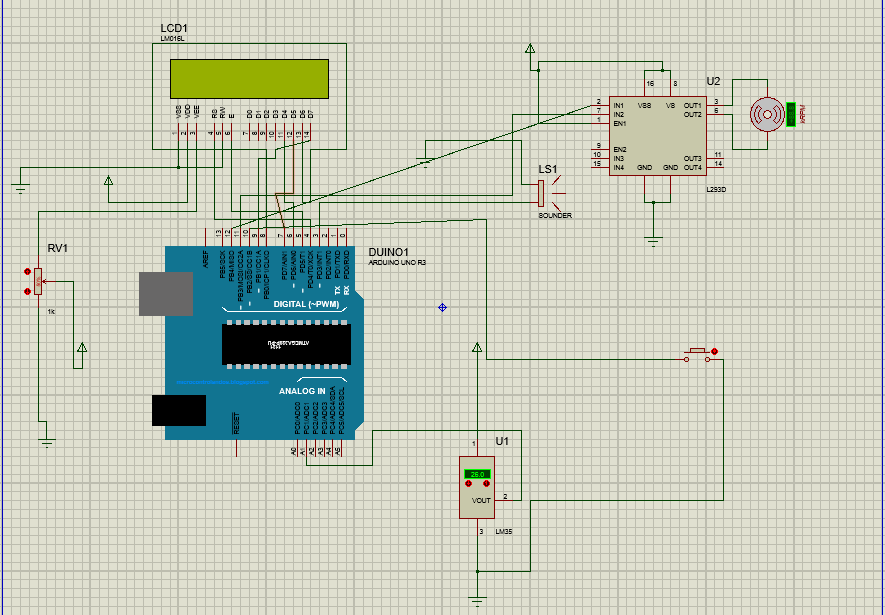


Fig-2:Connecting arduino uno, display module, temperature sensor, motor, L293D, buzzer

Writing the Arduino sketch:

We will now write the Arduino sketch so that the Arduino board can connect with the temperature sensor, display module and sense temperature from surroundings. Here is the complete sketch for this part:

#include <LiquidCrystal.h>

#define SENSOR\_PIN A1

#define BUZZER\_PIN 3

#define MAX\_TEMP 27.0

#define MIN\_TEMP 25.0

int pin11 = 11;

int pin12 = 12;

LiquidCrystal lcd(4, 5, 6, 7, 8, 9); // Creating an LCD object

**// declaring global variable which are essential for operation time**

unsigned long start = 0, finish = 0, elapsed = 0, operationTime = 0;

float tempMax = 0;

int flag=0;

**/\* flipping of a number is needed to calculate time correctly.**

**\* it prevents start & finish to override & only calculates**

**\* elapsed time when both start & finish were calculated.**

**\*/**

int tmpmin=0;

int tmpmax=0;

unsigned short flip = 0;

void setup() {

Serial.begin(9600);

lcd.begin(16, 2**); // Initializing LCD object by specifying 16x2 display**

lcd.clear(); **// Clearing the display**

pinMode(SENSOR\_PIN, INPUT); **// sensor pin as input pin**

pinMode(BUZZER\_PIN, OUTPUT); **// buzzer pin as output pin**

pinMode(10,INPUT\_PULLUP);

pinMode(pin11, OUTPUT);

pinMode(pin12, OUTPUT);

}

void loop() {

//Serial.println(start);

float tempCurrent = readTemp(); **// read current temp**

if (tempCurrent > MAX\_TEMP) { **// when current tempearture greater than Max\_TEMP**

// buzz the buzzer

if (flip == 0) { **// when flip isn't flipped**

start = millis()/1000; **// start time**

flag=0;

Serial.println(start);

//Serial.println(flag);

}

tone(BUZZER\_PIN, 1300, 1400);

digitalWrite(pin11,LOW);

digitalWrite(pin12,HIGH);

flip = 1; **// flip the number**

}

else if (tempCurrent < MIN\_TEMP) { **// when current temperature less than MIN\_TEMP**

**// buzz the buzzer**

if(flag==0){

// Serial.println(flag);

finish=millis()/1000;

flag=1;

}

if (flip == 1) { **// when flip is flipped**

finish = millis()/1000; **// finish time**

elapsed = finish - start; **// time needed**

operationTime += elapsed; **// add to operation time**

setTimeElements(); **// set the time elements to default**

}

tone(BUZZER\_PIN, 2500, 500);

}

else if(tempCurrent < MAX\_TEMP && tempCurrent > MIN\_TEMP)

{

finish = millis()/1000;

digitalWrite(pin11,LOW);

digitalWrite(pin12,LOW);

}

if(digitalRead(10)==LOW)

{

//finish = millis()/1000; **// finish time**

elapsed = millis()/1000- start; **// time needed**

operationTime += elapsed;

Serial.println("Time difference :");

Serial.println(elapsed);

Serial.println("Total operation Time :");

Serial.println(operationTime);

digitalWrite(pin11,LOW);

digitalWrite(pin12,LOW);

}

if(digitalRead(10)==HIGH)

{

//Serial.println("push button off");

}

if (tempCurrent > tempMax) {

tempMax = tempCurrent;

}

lcd.setCursor(0, 0);

lcd.print("Max\_Temp:");

lcd.print(tempMax);

lcd.setCursor(0,1);

lcd.print("CT:");

lcd.print(tempCurrent);

lcd.print(" T:");

lcd.print(operationTime);

**//serial print section**

//Serial.setCursor(0, 0);

Serial.print("Max\_Temp:");

Serial.print(tempMax);

//Serial.setCursor(0,1);

Serial.print(" Current Temperature : ");

Serial.print(tempCurrent);

Serial.print(" Total Operation Time : ");

Serial.println(operationTime);

delay(1000);

}

**// Creating a function to read temp**

float readTemp() {

int reading = analogRead(SENSOR\_PIN);

float tempC = (5.0 \* reading \* 100.0) / 1024;

return tempC;

}

**// Creating a function to set the time elements**

void setTimeElements() {

flip = 0;

start = 0;

//finish = 0;

elapsed = 0;

}

Results and Discussion:

This project emphasizes on two fact the time it takes to increase the temperature to certain max value and the time it takes to cool the temperature back to a certain low value.Another noticeable fact is there is a advavtage of controlling the fan when it is ON so if the fan is turned off on a certain time ,the sum of the default and manual turned off time .

