Indian Institute of Technology Bombay
Department of Electrical Engineering
Academic Session: 2021-22, Semester: II

Course Project Report

Course: EE 344: Electronic Design Lab

Project Group No: MON-JJ-9-1

Students:

Sl	Roll No.	Name	Email
1	19D070070	Mayur Ware	19d070070@iitb.ac.in
2	19D070034	Rushikesh Metkar	19d070034@iitb.ac.in
3	19D070057	Siddhant Shingade	19d070057@iitb.ac.in

Project Mentors: Prof. Joseph John

Project Title: Contactless Thermometer

Date of Submission: May 3, 2022

Abstract:

Temperature of an object gives an idea about its degree of hotness or coldness with respect to some standard value. Coming in direct contact with very hot/cold objects can be dangerous. Also, in various applications requiring temperature measurements, the surfaces/objects are unavailable for direct contact. Such applications require the use of a contactless thermometer. An Infrared lens is used to focus the incoming radiation from an object/surface onto an Infrared sensor which forms an important part of the contactless thermometer. Various subcircuits designed are then used to reduce the noise in the signal obtained and amplify it to extract useful information.
An LCD interfaced with an arduino is used to display the temperature after performing an analog to digital conversion. LM35 is used for calibration of the temperature and for checking the correctness of the temperature measuring system. The results obtained show that the temperature obtained contactlessly by our thermometer is correct and in accordance with LM35 measurements. The thermometer developed can sense temperatures contactlessly over a wide range of 15^oC to 80^oC. The project stands completed with the thermometer ready for use in day-to-day life.

Contents

1 Introduction

Temperature determination is an important part of many applications and situations in our day to day life. Measuring temperature without contact with the object/surface helps us determine the temperature of very hot/cold surfaces and objects which are otherwise unreachable to us. In this project we aim to build a thermometer capable of measuring temperature of objects/surfaces over a wide range without coming in direct contact with them.
We would be using an IR lens mounted on an infrared temperature sensor for focusing the radiation from the object. We would then apply signal processing techniques to amplify the signal and reduce the noise. Finally, we would interface Arduino to our circuit and use a LCD to display the final temperature. For testing, we compare our temperature with the temperature from LM35.

1.1 Block Diagram

Fig. 1 shows the brief block diagram describing our Contactless Thermometer. In the start, object whose temperature is to be measured, is brought near the system. The IR waves coming from the object are concentrated on the ZTP-135SR IR temperature sensor using the IR Lens which adds directivity and sensitivity to the inputs. This sensor has two independent outputs, thermistor and thermopile. thermistor measures temperature using its resistance dependency and thermopile measures temperature using its voltage dependency.
Thermopile output is passed through a 2 pole sallen key low-pass filter and then amplified using an OpAmp. Whereas, thermistor output is biased using a voltage divider. Both the outputs are then given to the Arduino. As both the sensor outputs are independent, we can analyze them separately using Arduino and choose the suitable method to us. After several tests at different temperatures, we decided to go with thermopile method as its outputs were stable. This is explained in the later sections. Then, the thermopile output data was converted into digital values and displayed using LCD.

1.2 Design Approach

An IR thermometer consists of four basic stages: optics, sensing, signal conditioning, and digital output. Fig. 2 shows our system block diagram with all the subsystems included. The optics stage involves a lens for long-range Infrared Temperature sensing. We have used this lens to increase the sensing range of our IR Temperature Sensor. For temperature compensation, we employ a thermopile sensor with a negative temperature coefficient (NTC) thermistor in the sensing stage.
The thermopile sensor absorbs IR radiation and converts it to a small voltage (typically in the order of μVs). It's relation with the temperature is given as

$$V = AT^4 \tag{1}$$

[where A is some constant]
The noise in this signal is removed by using a Sallen key low-pass filter and then amplified using a non-inverting amplifier. The NTC thermistor is a temperature-dependent resistor that decreases in resistivity as temperature increases. In order to turn that resistance into a processable signal, we will use a basic voltage divider. The resistance-temperature relation is given as follows

$$R = R_o(e^{\alpha(\frac{1}{T} - \frac{1}{25})}) \tag{2}$$

[where R_o is the resistance at room temperature]
The thermopile and thermistor signals are sent as input to the microcontroller (Arduino) to compare the respective outputs as mentioned earlier. We choose to use thermopile over thermistor as it gave better results. This is discussed in detail in the test results section. As we can get the temperature with the help of LM35, we will calibrate the output voltage of our circuit to a corresponding temperature value. These values, along with the thermistor voltage, are converted to a final compensated temperature. That temperature value is then output to a 16×2 LED display.

2 Project Design

2.1 User Requirements

1) Direct the measuring end of the thermometer towards the object.
2) The object must be within a range of 10-15cm to get accurate results.

3) Don't try to use the thermometer beyond 120^{o}C. This may harm the thermometer.
4) Don't remove the sensor if a stable temperature is not seen. It may take some time to measure the temperature.
5) Keep the sensor away from direct sunlight, fire and water.

2.2 Technical Specifications

Battery :
- 2 General purpose HIW 6F22 9V batteries
- Easy to assemble
- Removable

LCD :
- 16x2 LCD Module
- Single power supply (5V)
- Refresh frequency of 270kHz

Operational Amplifier :
- OP07CP OpAmp
- 75μV input bias voltage at room temperature with rate of 1.3μV/^{o}C
- 3.8nA input bias current

Dimensions and other specifications :
- 10cm × 8cm
- Lightweight

2.3 Subsystems

The subsystems involved in our system are IR temperature sensor, signal conditioning, Arduino interfacing and compact PCB. All the subsystems are discussed in detail in the following subsections.

2.3.1 Infrared (IR) Temperature Sensor

Choosing the sensor was one of the most important decisions in our project. The important specifications we considered while deciding on the sensor were :
a) Low dark current
b) High output range which can avoid the issues caused by the input bias voltage of OpApmp
c) Suitable temperature range and dimensions
d) Low signal to noise ratio (SNR)
For our project, we are using a ZTP-135SR thermopile IR sensor as shown in Fig. 3. The thermopile and thermistor can be modeled as a voltage source and a resistor respectively. Both change their values when the temperature rises as discussed in section 1.2.

2.3.2 Signal Conditioning Circuit

To remove the noise in the thermistor and the thermopile circuits, we use a Sallen key 2 pole low-pass filter shown in Fig. 4. The cut-off frequency of the low-pass filter is calculated as follows

$$f_{cutoff} = \frac{1}{2\pi\sqrt{R_1 R_2 C_1 C_2}} = \frac{1}{2\pi(18.2k)(1\mu F)} \approx 10Hz \tag{3}$$

It has a 1.6× gain because of non-inverting stage as shown in the equation below

$$G_{lpf} = \left(1 + \frac{R_2}{R_1}\right) = \left(1 + \frac{12k}{20k}\right) = 1.6 \tag{4}$$

This output signal at this stage is in the range of few hundreds of microvolts (μVs) at room temperature. The thermopile signal processing has an additional amplification stage with a 1000× gain (Fig. 5) using a non-inverting amplifier. After this stage, the output signal gets into the range of hundreds of millivolts (mVs) at room temperature.
Thus, the total gain of the low-pass filter and the amplifier is around 1600×. At room temperature, the output voltage of the thermopile circuit is 80mV.
The thermistor changes its resistance with temperature. As a result, a voltage divider is included in the thermistor model to convert the resistance to a voltage as shown in Fig. 6. The thermistor output is typically in the order of μVs, but for comparing it with thermopile we need to scale this output.

2.3.3 Arduino Interfacing

The thermistor and thermopile voltages are given as inputs to the Arduino as shown in Fig. 7 in order to produce a compensated output temperature. We plotted the graphs of thermopile vs temperature and thermistor vs temperature for our IR sensor. The test results are discussed in section 3. We choose to use the thermopile depending on the results. We found the best fitting curve between the data points. This gives us the output Voltage-Temperature relation. The relation is discussed in section 3.
From the output Voltage-Temperature relation, we convert the analog value read by the Arduino to a precise temperature value. Now, this temperature value is displayed using a 16×2 LCD display interfaced with Arduino.

2.3.4 Optics

We used an IR lens as shown in Fig. 8 for long-range infrared temperature sensing. We have used this lens to increase the sensing range of our IR Temperature Sensor. At 2cm the lens can focus a circle of radius of 1cm on the sensor. It is a small size lens so that we can easily implement it in our project and make our design compact.

2.3.5 Compact PCB

We built a PCB (Fig. 9) by designing it in Eagle software. It was fabricated in PCB Lab, Department of Electrical Engineering, IIT Bombay by using chemical etching process. The dimensions of PCB are 5cm×8cm. We soldered it in WEL. We observed a significant improvement in results by using PCB instead of a breadboard.

3 Test Results

Firstly, we tested the effect of 2 pole sallen key low pass filter on the thermopile output. We analyzed the output with LPF and without LPF. The results are shown in Fig. 10 and Fig. 11 respectively. Without LPF, noise can be clearly seen in high quantity and the mean output is fluctuating. Whereas, with LPF, noise level is decreased significantly, mean output is stable.
After this, we tested the effect of 9V battery on the circuit. Before using battery, we used 9V direct power supply from WEL Lab. We observed that there were some fluctuations in the supply. Also, peak voltage was not very stable. But, when we used batteries instead of this, results were improved because of a stable voltage supply.
After making the circuit, we observed the dependency of sensor output with temperature. For this, we used solder gun and hot water made using electric kettle available at WEL Lab. During these tests, we avoided the direct contact with sensor body as it might have damaged the sensor. The experimental setups are shown in Fig. 12 and Fig. 13.
We took readings of both the thermistor and thermopile output voltage of our sensor at different temperatures as shown in Fig. 14. We calibrated the data by mounting a LM35 near our sensor. This is the plot of the output voltage versus temperature. It is very important to notice that output of thermistor (blue curve) is non-uniform and fluctuating as compared to the output of thermopile (red curve). This data helped us to decide that thermopile method is more suitable for us.
From the 31 data points of the thermopile output as shown in Fig. 14, we calculate the best fit quadratic polynomial using online resources which is as follows

$$Temperature(T) = 0.042 \times (V_{Out})^2 + 13.376 \times (V_{Out}) - 390.82 \tag{5}$$

The best fit quadratic graph of the data is shown in Fig. 15. This is the relation between the thermopile output voltage and temperature.
The output temperature of our circuit and the temperature given by LM35 is displayed on the LCD interfaced with Arduino (Fig. 16).

4 Bill of Materials

Component	Price
Infrared Lens	Rs. 267
ZTP-135SR IR Sensor	Rs. 312
2 × OP07CP OpAmp	Rs. 40 (Issued from WEL)
2 × 9V battery with connector	Rs. 74 (Issued from WEL)
Arduino UNO Board	Rs. 700 (Issued from WEL)
LCD Display (16 × 2)	Rs. 130 (Issued from WEL)

5 Conclusions

The current performance of the circuit is limited by the IR sensing range of the sensor and the IR lens as well. Wide-range sensing sensor and a better IR lens would result in more stable and accurate results. The 50Hz noise due to the electric components in the surroundings and other environmental effects can be resolved by making a more compact design and enclosing it within a box.
The range of temperature values that we can sense is currently in the range of 15^oC to 80^oC due to the environmental effects and optics used. Future work can aim at improving this dynamic range.

Acknowledgements

We would like to extend our sincere thanks to Prof. Joseph John Sir, Prof. P.C. Pandey Sir, Prof. Kushal Tuckley Sir and Prof. Siddharth Tallur Sir for their invaluable and crucial guidance and supervision without which the successful completion of this project would have been impossible. We would also like to thank the whole WEL Lab team for their invaluable help in the completion of this project. Also, we would like to thank the PCB Lab team for successfully preparing our PCB.

Appendices

Code used by us for Arduino interfacing is shown below :

```
//include the library code:  include <LiquidCrystal.h>
const int sensor=A5; // Assigning analog pin A5 to variable 'sensor'
const int s2 = A0;
float tempc; //variable to store temperature in degree Celsius
float tempf; //variable to store temperature in Fahreinheit
float temp1;
float vout; //temporary variable to hold sensor reading
float vckt_out;
float v1;
// initialize the library by associating any needed LCD interface pin
// with the arduino pin number it is connected to
//const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(9,11,5,4,3,2);
int Contrast=20;
void setup()
pinMode(sensor,INPUT); // Configuring sensor pin as input
pinMode(s2,INPUT);
Serial.begin(9600);

void loop()
vout=analogRead(sensor); //Reading the value from sensor
Serial.print("LM35 out = ");
Serial.print(vout);
Serial.print(" ");
vout=(vout*500)/(1023);
tempc = vout; // Storing value in Degree Celsius
```

```
Serial.print("in DegreeC=");
Serial.print("");
Serial.print(tempc);
Serial.print(" ");
v1=analogRead(s2);
v1 = (v1*150)/1023;
Serial.print("ZTP135 out = ");
Serial.print(v1);
Serial.print(" ");
vckt_out = 0.042*v1^2 + 13.376*v1 - 390.82
temperature = vckt_out;
Serial.print("in DegreeC=");
Serial.print("");
Serial.print(tempc);
Serial.print(" ");
// set up the LCD's number of columns and rows:
lcd.begin(16, 2);
// Print a message to the LCD.
lcd.print("Output :", temperature);
analogWrite(6,Contrast);
// set the cursor to column 0, line 1
// (note:  line 1 is the second row, since counting begins with 0):
lcd.setCursor(0, 1);
// print the number of seconds since reset:
lcd.print("LM35 :", tempc);
Serial.println();
delay(1000); //Delay of 1 second for ease of viewing
```

Use the following link to see the visual description of our working prototype :
https://drive.google.com/file/d/1DU2fUrfvwK998bJtFqbiQJEidhrkvHjm/view?usp=sharing

References

[1] Quadratic regression calculator
Available at : https://keisan.casio.com/exec/system/14059932254941
Last accessed on 03/05/2022

[2] Electronic components datasheet
Available at : https://pdf1.alldatasheet.com/datasheet-pdf/view/27256/TI/OP07CP.html
Last accessed on 03/05/2022

Figures

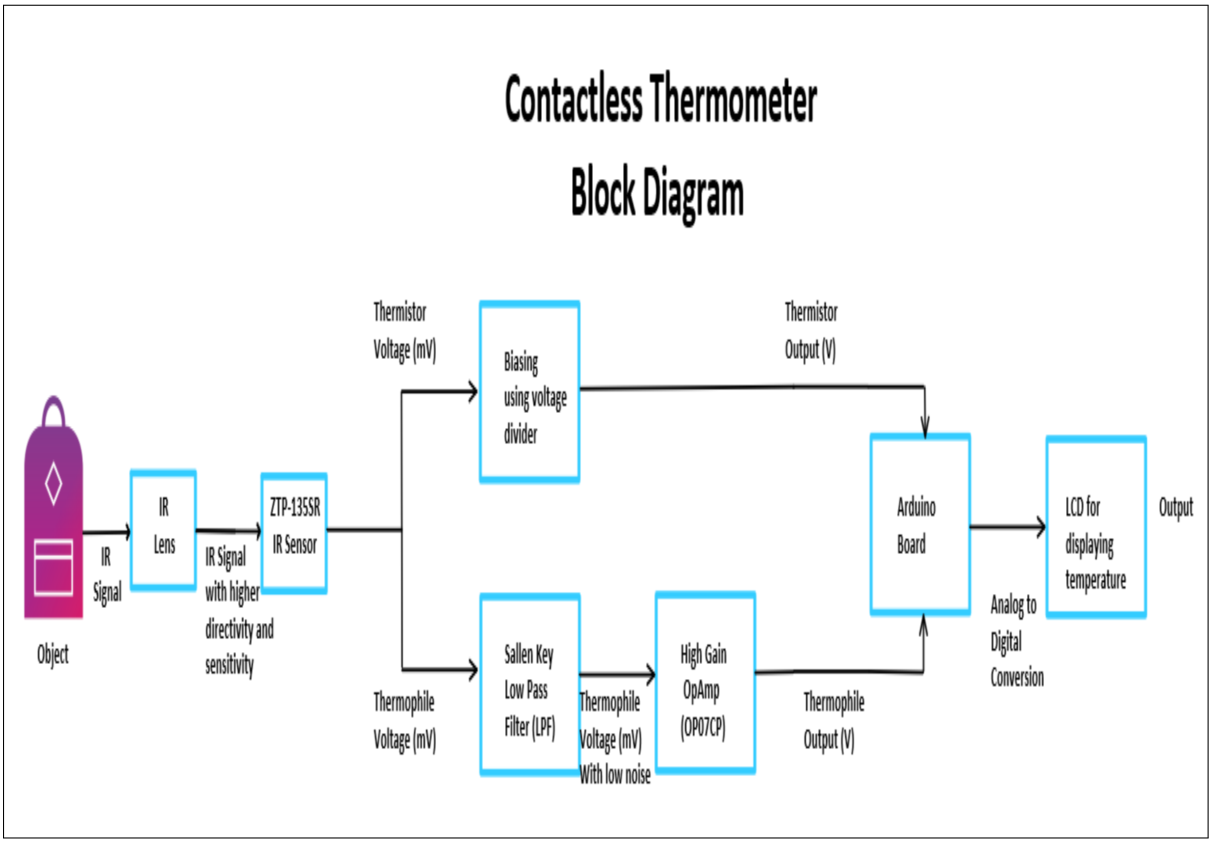

Fig. 1 : Block Diagram

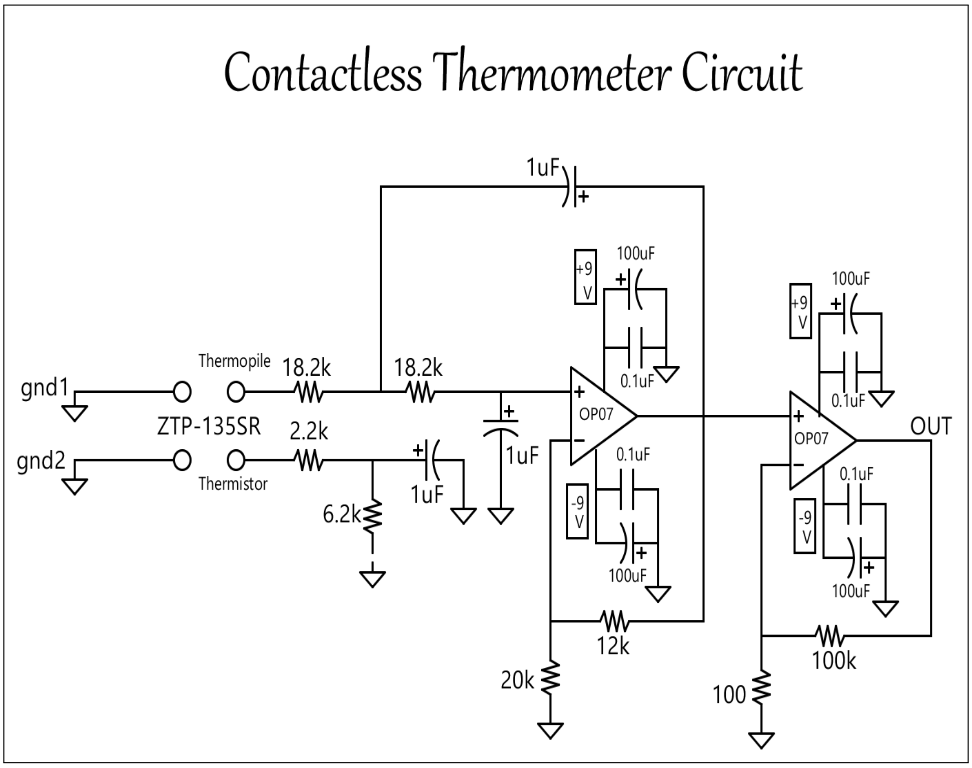

Fig. 2 : Contactless Thermometer Circuit

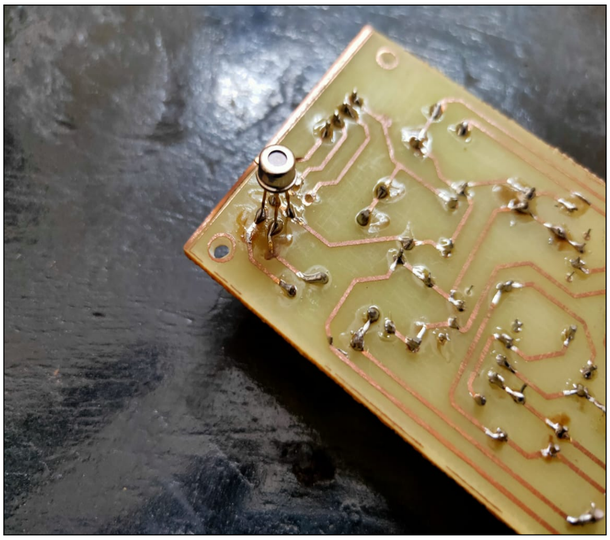

Fig. 3 : ZTP-135SR IR Sensor

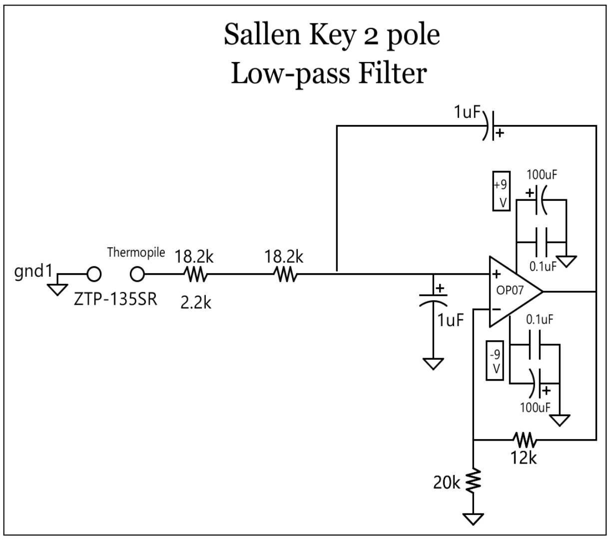

Fig. 4 : 2 Pole Sallen Key Low-pass Filter

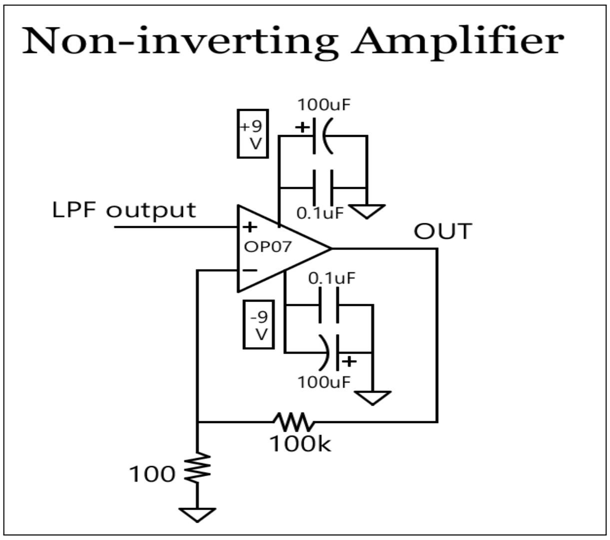

Fig. 5 : Non-inverting Amplifier

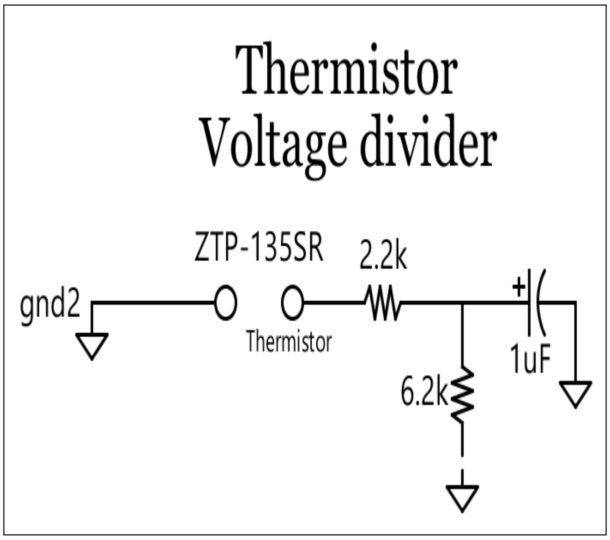

Fig. 6 : Thermistor Voltage Divider

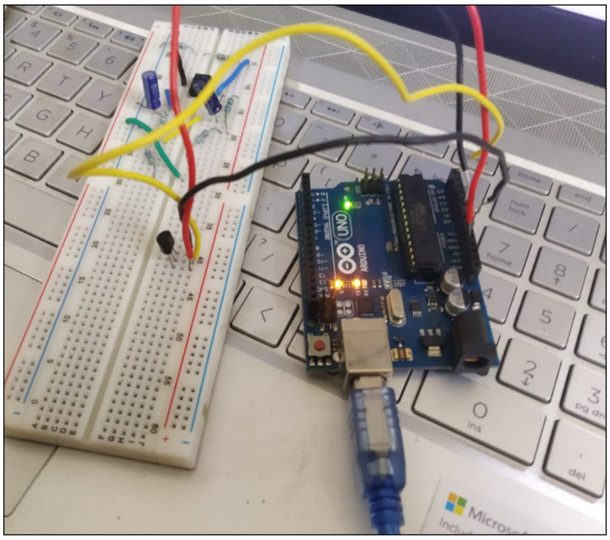

Fig. 7 : Arduino Interfacing

Fig. 8 : IR Lens

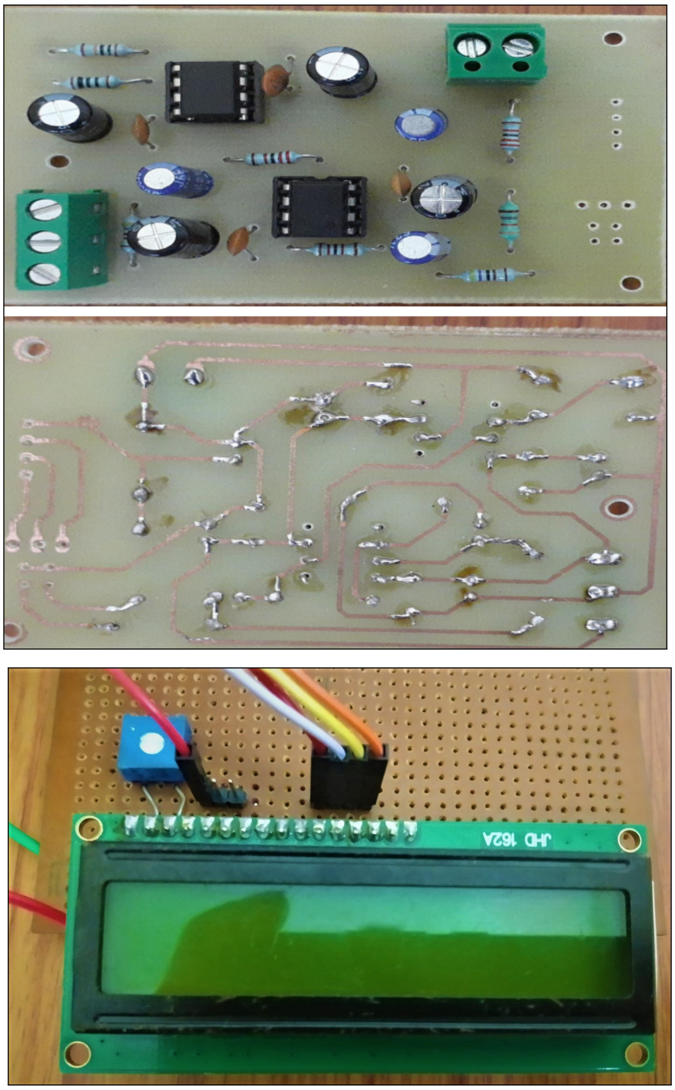

Fig. 9 : PCB

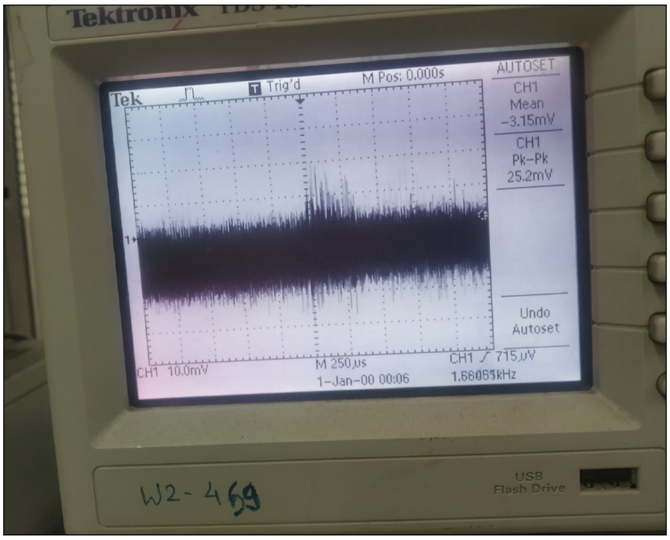

Fig. 10 : Sensor output without LPF

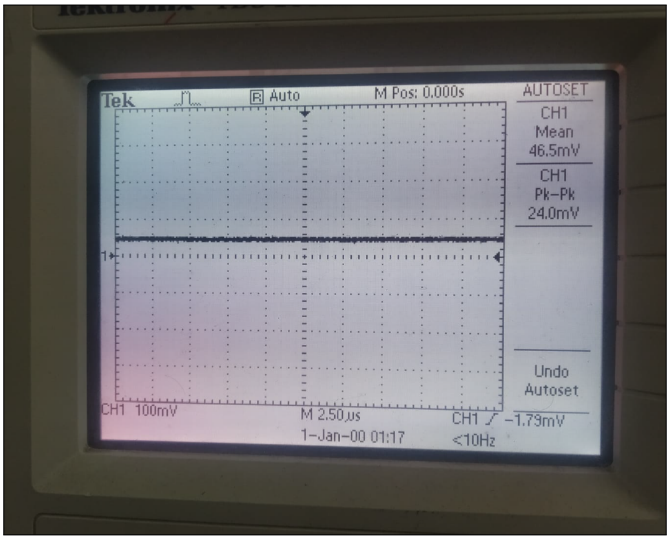

Fig. 11 : Sensor output with LPF

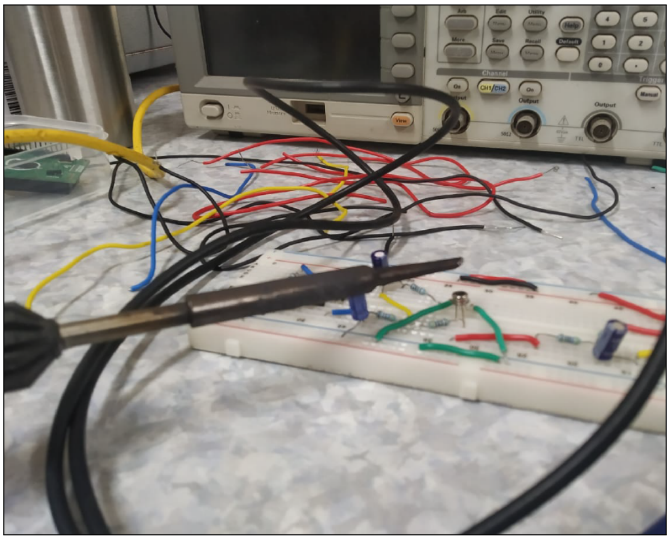

Fig. 12 : Testing using solder

Fig. 13 : Testing using hot water

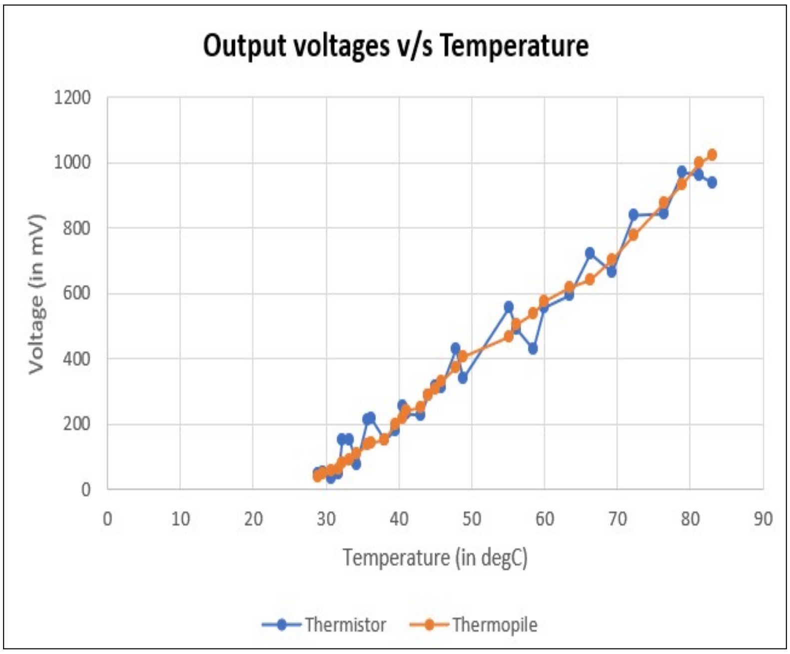

Fig. 14 : ZTP-135SR Output Voltage vs Temperature

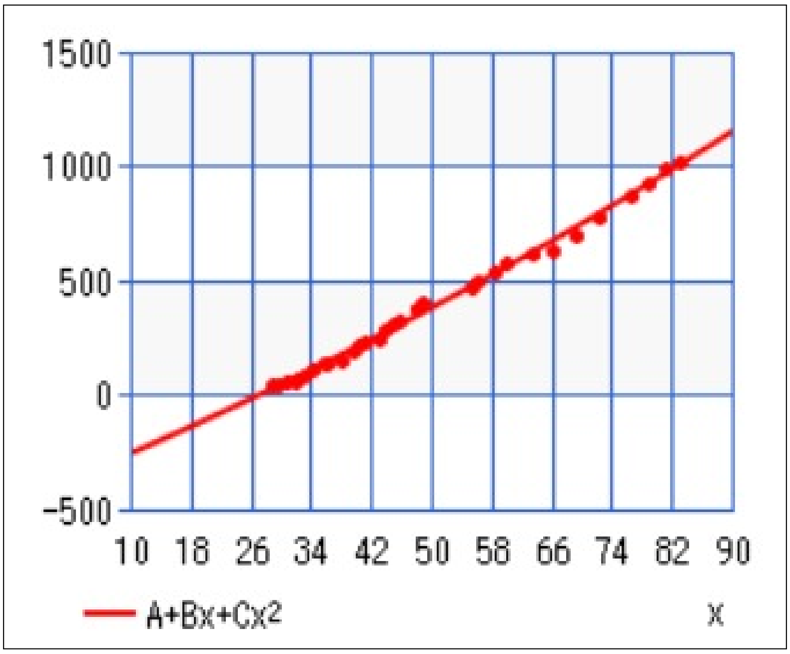

Fig. 15 : Fit Quadratic Graph

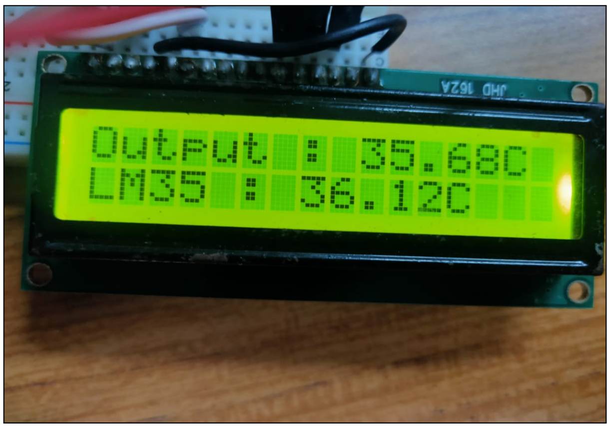

Fig. 16 : LCD Interfacing