©2016 Romanian Association of Nonconventional Technologies

DESIGN AND IMPLEMENTATION OF A THERMOMETER WITH ARDUINO AND THERMISTOR

Răzvan-Daniel ALBU
University of Oradea, Faculty of Electrical Engineering and Information Technology, Romania, E-mail: ralbu@uoradea.ro

ABSTRACT: In this paper I will present the implementation process of a thermometer using ARDUINO and a thermistor that has a reduced cost, and is able to serve in a variety of applications. The article presents the entire development process from both hardware and software perspectives. Thermistor resistance variation can increase with temperature, or inversely, while the temperature increases, the thermistor resistance decreases. The Steinhart–Hart equation is used to derive a precise temperature of the thermistor since it provides a closer approximation to actual temperature than simpler equations, and is useful over the entire working temperature range of the sensor. Compared with traditional thermometer, the one presented in this paper is more flexible, easy to use and integrate in a variety of application, and very cost effective.
KEYWORDS: thermometer, thermistor, ARDUINO,

1. INTRODUCTION

The thermistor is a semi-conductor device which has a nonlinear voltage-current characteristic V(I), because its resistance is related with temperature. Thermistor resistance variation can have a PTC (Positive Temperature Coefficient), which means the thermistor resistance increases with temperature, or a NTC (Negative Temperature Coefficient), once the temperature increases the thermistor resistance decreases [1,2].

In this paper, I made use of a measuring thermistor with NTC, having a resistance of 10 KΩ, at a temperature of 25º C.

It is known that a semi-conductor of type n, extrinsically and heavily doped, shows an electrical conductivity according to:

$$\sigma = e \cdot n_n \cdot \mu_n = \frac{1}{\rho} \quad (1)$$

where:

- σ is electrical conductivity,
- e is the electron charge,
- n_n is the concentration of electrons (it depends on temperature)
- μ_n is the electron mobility (it depends on temperature)
- ρ is the material resistivity.

According to (1) and using temperature variation of activation energy, we can describe resistance variation based on temperature, for an NTC thermistor, as follow:

$$R(T) = A \cdot \exp\left(\frac{B}{T} + \frac{C}{T^2} + \frac{D}{T^3} + \cdots\right) \quad (2)$$

For modelling the resistance dependence of temperature I used Steinhart - Hart equation.

$$\frac{1}{T} = A + B \cdot ln\, R + C \cdot (ln\, R)^3 \quad (3)$$

Where T is the absolute temperature, and A, B, C are constants available in catalogs.

To determine A, B and C constants, the thermistor resistance is measured at 3 different temperatures. The thermistor resistance at T1 temperature is R1, and similarly we have T2, R2, and T3, R3. And we solve the following equations system:

$$\begin{bmatrix} 1 & \ln(R_1) & ln^3(R_1) \\ 1 & \ln(R_2) & ln^3(R_2) \\ 1 & \ln(R_3) & ln^3(R_3) \end{bmatrix} \cdot \begin{bmatrix} A \\ B \\ C \end{bmatrix} = \begin{bmatrix} \frac{1}{T_1} \\ \frac{1}{T_2} \\ \frac{1}{T_3} \end{bmatrix} \quad (4)$$

The following notations are used:

$$X_1 = \ln(R_1),\ X_2 = \ln(R_2), \quad X_3 = \ln(R_3) \quad (5)$$

$$Y_1 = \frac{1}{T_1},\ Y_2 = \frac{1}{T_2},\ Y_3 = \frac{1}{T_3} \quad (6)$$

$$Z_2 = \frac{Y_2 - Y_1}{X_2 - X_1},\ Z_3 = \frac{Y_3 - Y_1}{X_3 - X_1} \quad (7)$$

Finally, we obtain A, B and C constants as follows:

$$C = \left(\frac{Z_3 - Z_2}{X_3 - X_2}\right) \cdot \left(\frac{1}{X_1 + X_2 + X_3}\right) \quad (8)$$

$$B = \frac{Y_2 - Y_1}{X_2 - X_1} - C \cdot (X_1^2 + X_1 \cdot X_2 + X_2^2) \quad (9)$$

$$A = Y_1 - (B + X_1^2 \cdot C) \cdot X_1 \quad (10)$$

This article is structured as follows: after a short introduction where mathematical background is briefly described, section two shows the wiring

diagram and the hardware implementation, continuing in section three with the software implementation, presenting the full source code, and ending with relevant conclusions and useful references.

2. SIMILAR RESEARCH WORKS

In [3] A comparative study on the recording of temperature by the clinical mercury thermometer and digital thermometer is presented. It was concluded that the difference in the temperature readings between the clinical mercury thermometer and the digital thermometer is of no significance, and consequently the digital thermometer is a better option since it is environmentally friendly.

In [4] is presented the implementation and design of a digital alarm clock with thermometer. They made use of a LM 35 temperature sensor and ARDUINO (Atmel ATmega328) microcontroller.

In [5] they introduced a novel design method of a digital IR thermometer, which uses Arduino unoR3 as the main control device, and MLX90614 as the Infrared (IR) thermometer sensor. They claim their implementation is cost-effective, accurate, portable, and easy to operate.

In [6] is presented a low cost and flexible home control and environmental monitoring system using an Android based Smart phone. They used a micro web server and Bluetooth communication as an interoperable application layer for communicating between the remote user and the home devices.

In [7] a complete description of the thermometer is made, showing its theory, why to use it, the difficulties of measuring, and two solutions, each one optimized for a specific aspect, first for simplicity, and the second one for accuracy. The first solution concentrates on reducing the complexity of the measurement by means of a hardware-based analogue reference junction compensation technique. The second solution provides the highest accuracy measurement and also enables the use of various thermocouple types.

3. HARDWARE IMPLEMENTATION

ARDUINO is a hardware and software project, and user community that designs and manufactures computer open-source hardware, open-source software, and microcontroller-based kits for building digital devices, and interactive objects, that can sense and control physical devices. The open-source integrated development environment, ARDUINO, is available for the most known operating systems (Windows, Mac OS X, Linux) [8,9].

To implement the thermometer, I utilized the Arduino Uno development board equipped with ATmega 328 single-chip 8 bits' microcontroller, that has a RISC architecture. The microcontroller can operate at frequencies up to 20 MHz and it can do 1 MIPS per MHz. Its internal memory consists of 32 KB programmable flash memory, 1 KB user memory of type EEPROM and 2 KB data memory of type SRAM.

The circuit diagram is shown in figure 1. As we can notice, the results are displayed on an alphanumeric 16X4 LCD.

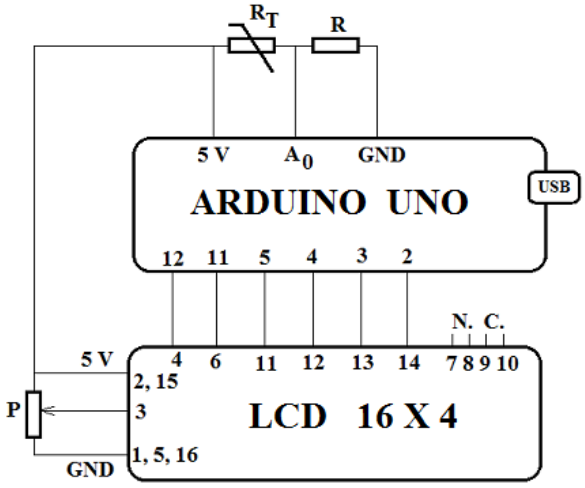

Figure 1. The circuit diagram of the thermometer

Analysing the data sheet of ATmega328 microcontroller we can see that is has 6 channels for analogic signal acquisition, that are capable of converting it to a digital signal with a resolution of 10 bits. The 6 pins of Arduino Uno development board are noted: A0, A1, A2, A3, A4, A5, A6. Since they accept input voltage between 0 and 5 Volts, the measuring resolution of input voltage is:

$$\frac{5\,V}{2^{10}} = \frac{5\,V}{1024} = 4{,}88\,mV \qquad (11)$$

The information about temperature value, which modifies the thermistor resistance, can be found in input voltage formula that is applied at A0 pin.

$$V_{A0} = \frac{R}{R+R_T} \cdot 5V \qquad (12)$$

4. SOFTWARE IMPLEMENTATION

In this section is presented the Arduino C++ program that captures the temperature and displays it on the LCD. This program makes use of math.h library for mathematical operations and, for solving the Steinhart-Hart equation, and the LiquidCrystal.h library for displaying data on the LCD.

```
#include <math.h>

#include <LiquidCrystal.h>

LiquidCrystal lcd (12, NULL, 11, 5, 4, 3, 2);
```

```
// The 16x4 LCD is connected to the ARDUINO //board at corresponding pins
double TermistorF (int RawADC) {
 double Temp;
 // Voltage value corresponding to measured //temperature is stored using 8 bytes, as a double //variable
 Temp = log(10000.0*((1024.0/RawADC-1)));
 Temp = 1 / (0.001129148 + (0.000234125 + (0.0000000876741 * Temp * Temp)) * Temp);
 // Making use of Steinhart-Hart equation
 Temp = Temp - 273.15;
 Temp = (Temp * 9.0)/ 5.0 + 32.0;
 // Using the temperature in Fahrenheit degrees
 return Temp;
}
double TermistorC (int RawADC) {
 double Temp;
 Temp = log(10000.0*((1024.0/RawADC-1)));
 Temp = 1 / (0.001129148 + (0.000234125 + (0.0000000876741 * Temp * Temp)) * Temp);
 // Making use of Steinhart-Hart equation
 Temp = Temp - 273.15;
 // Making use of temperature in Celsius degrees
 return Temp;
  }
void setup () {
 lcd.begin(16, 4);
// Using the LCD with 16 columns and 4 rows
}
void loop () {
  int valC, valF;
  double tempC, tempF;
  valF=analogRead (0);
  tempF=TermistorF(valF);
  valC=analogRead (0);
  tempC=TermistorC(valC);
  lcd.print (“Temperature ");
  // printing first line of text on the LCD
  lcd.setCursor(0,1);
  lcd.print (" measured is: ");
  // printing second line of text on the LCD
  lcd.setCursor(0,2);
  lcd.print(tempC);
  lcd.print (" CELSIUS &");
  // printing temperature in Celsius degrees
  lcd.setCursor(0,3);
  lcd.print(tempF);
  lcd.print (" FAHRENHEIT");
  // printing temperature in Fahrenheit degrees
  delay (1000);
  lcd.clear();
}

#include <math.h>
double TermistorF (int RawADC) {
 double Temp;
 Temp = log(10000.0*((1024.0/RawADC-1)));
 Temp = 1 / (0.001129148 + (0.000234125 + (0.0000000876741 * Temp * Temp)) * Temp);
 Temp = Temp - 273.15;
 Temp = (Temp * 9.0)/ 5.0 + 32.0;
// Using the temperature in Fahrenheit degrees
 return Temp;
}
double TermistorC (int RawADC) {
 double Temp;
 Temp = log(10000.0*((1024.0/RawADC-1)));
 Temp = 1 / (0.001129148 + (0.000234125 + (0.0000000876741 * Temp * Temp)) * Temp);
 Temp = Temp - 273.15;
// Compute temperature in Celsius degrees
 return Temp;
}
void setup () {
 Serial.begin(9600);
}
void loop () {
  int valF, valC;
  double tempF, tempC;
```

```
  valF=analogRead (0);
  tempF=TermistorF(valF);
  valC=analogRead (0);
  tempC=TermistorC(valC);
  //printing on the LCD the temperature in both
//Fahrenheit and Celsius degrees
  Serial.print("Temperature = ");
  Serial.print(tempF);
  Serial.print(" FAHREINHEIT; ");
  Serial.print(tempC);
  Serial.println(" CELSIUS");
  delay (1000);
}
```

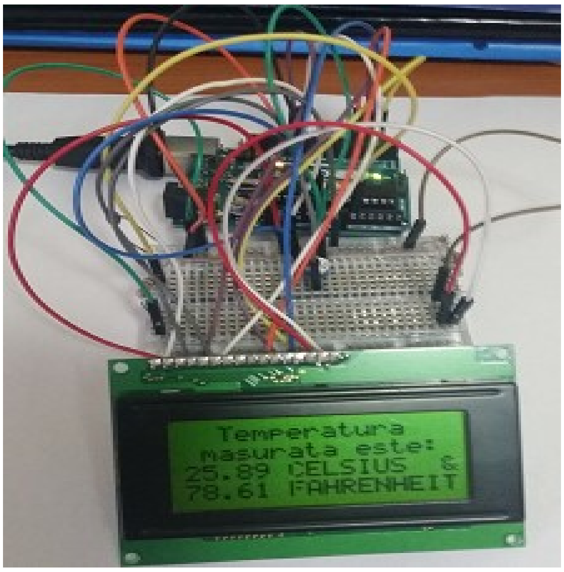

Figure 2. The experimental setup

In figure 2 is presented the experimental setup.

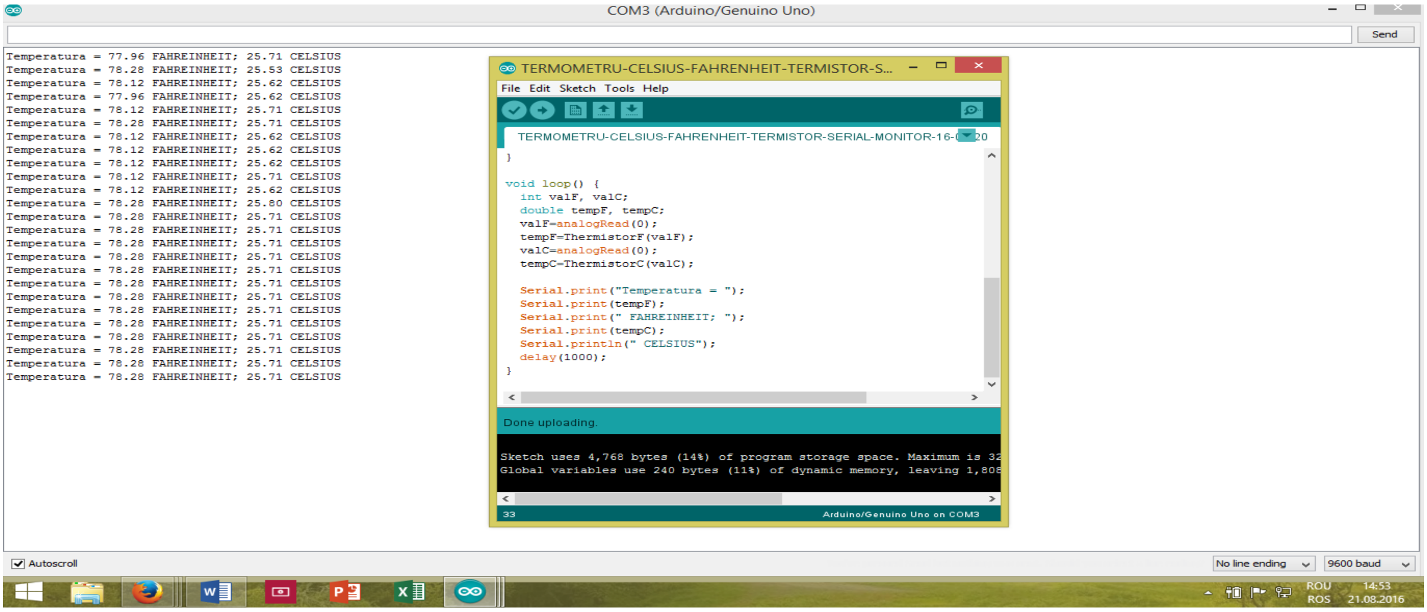

Figure 3. The measured temperature displayed on console

5. CONCLUSIONS

In this paper I have presented the implementation of thermometer with ARDUINO and thermistor. The main advantages of this thermometer are

- It is ecological. Despite the mercury thermometer it is nontoxic.
- It is very easy to read, since the reading of a number with decimals is easier than reading from a graduated scale.
- Since thermistor is very cheap, this thermometer costs less than other thermometers implemented with microcontrollers and temperature sensors (LM-35, TMP-102).
- Temperature is displayed in Celsius and Fahrenheit degrees; (Kelvin degrees can also be implemented) despite electronic thermometers with analogic circuits.
- It is very flexible; it can be easily reprogrammed to control a buzzer, or the lighting of a LED, at a certain temperature. The microcontroller can also communicate with other electronic devices using I^2C (Inter-Integrated Circuit), or SPI (Serial Peripheral Interface Bus).
- Can be built in both mobile and stationary devices, and also it can be connected to an Ethernet network.

As future working plans, I want to extend this project and integrate this thermometer into more complex applications, like an intelligent house system.

6. REFERENCES

1. NTC thermistors for temperature measurement TDK, https://en.tdk.eu/inf/50/db/ntcsmd_11/SMD_NTCs_NiBarrier_Standard_0805.pdf
2. Oltean I.D, Componente electronice passive, Brasov: Lux Libris, 2000,
3. Rinzin Dolkar, Surinder Kapoor, Neena Vir Singh, Vikas Suri, A comparative study on the recording of temperature by the clinical mercury thermometer and digital thermometer, Nursing and Midwifery Research Journal, Vol-9, No.1, January 2013.
4. Bose Mathew Jose, Ammu Catherine Treesa, Anand M., Arjun S., Aswathy Ramachandran, Design and Implementation of Digital Alarm Clock with Thermometer, International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering, Vol.3, Issue 2, February 2014.
5. Mahammad Dada Vempali, K. Tanveer Alam, D. Punyasheshudu, B. Ramamurthy, Design and Development of Digital Proximity Type IR Thermometer Based on Arduino unoR3, International Journal of Innovative Research in Science, Engineering and Technology, Vol. 5, Issue 2, February 2016.
6. Nathan David, Abafor Chima, Aronu Ugochukwu, Edoga Obinna, Design of a Home Automation System Using Arduino, International Journal of Scientific & Engineering Research, Volume 6, Issue 6, June-2015 795 ISSN 2229-5518.
7. Matthew Duff, Joseph Towey, Two Ways to Measure Temperature Using Thermocouples Feature Simplicity, Accuracy, and Flexibility, Analog Dialogue 44-10, October (2010), http://www.analog.com/library/analogDialogue/archives/44-10/thermocouple.pdf
8. ARDUINO: https://www.arduino.cc/en/Main/Software http://www.arduino.org
9. HD44780U (LCD-II), HITACHI, https://www.sparkfun.com/datasheets/LCD/HD44780.pdf