Lab Report 1: Digital Thermometer using an Arduino

BME 4503C-Medical Instrumentation
2/19/2018

Carlos Flores
Kevin Silva
Kevin Baza-Gonzalez
Jude Stervil
Deyi Tang

Table of Contents

Introduction

This laboratory allowed our group to explore the basics of Matlab interfacing with the Arduino Uno, along with basic analogue sensors. A basic circuit was constructed that included a temperature sensor, a piezoelectric speaker and the Uno, forming a digital thermometer. The purpose of the device is to read ambient temperature and display it on a computer. Whenever the temperature reaches above 25 °C, the piezoelectric speaker plays a tone, and the Matlab GUI displaying the temperature shows a "CRITICAL Temp" warning.

Background

A piezoelectric speaker is a small buzzer that generates a sound when applying an electric current. The speaker has the property of the piezoelectric effect which means that it generates a small physical force as a result of applying an electrical field to it. In the case of the speaker, it contains a thin film of piezoelectric material that shrinks or grows depending on the electric field applied to it. The electric field applied will cause the atoms in the material to squeeze, hit or bend the crystal structure which generates the movement of the film. This movement produces the sound emitted from the speaker (1). The left side of figure 1 shows a side view of a piezoelectric speaker, where the orange and red film constitute the material that grows and shrinks. Our specific piezoelectric speaker has an operating voltage of 3.0 – 5.0 Vp-p according to the data sheet (2).

Figure 1. Piezoelectric Speaker Diagram and Temperature Sensor

The temperature sensor that we used for the lab is the TMP36, which is a device that outputs a voltage proportional to the temperature sensed. The TMP36 has an operating voltage of 2.7V to 5.5V, with an output scale factor of 10mV/°C and an accuracy of about +/- 2°C. According to the data sheet, the pin-out for the three available pins are as follows: Vin, Vout and ground, respectively according to the right side of figure 1 (3).

Materials and Methods

The materials for this lab simply included an "Arduino Uno", a "TMP36" temperature senseor, a piezoelectric sensor, and a copy of Matlab 2017. Before moving on with any of the code, we first setup the circuit. We simply connected the TMP36 sensor to the 3.3V rail from the Arduino, ground, and the analogue pin "A0". Then, we connected the piezoelectric speaker to the digital pin "D6" and the other to ground. The setup for the circuit can easily be seen in figure 2.

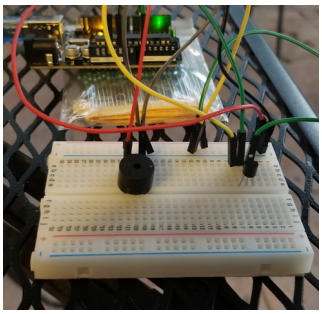

Figure 2. Circuit setup including the piezoelectric speaker, and the TMP36 sensor.

Next, our task was to create the Matlab GUI, so because we had purchased Matlab 2017b, we had access to the newer "App Designer" application, rather than the older "GUIDE" program (4). We simply used three different "Edit Field (Text)" common components from the app designer, as well as two buttons, one for starting the measurement, and another to end it. The "start" button was programmed to create an Arduino object, and it performs all of the temperature polling from the TMP36, which is then displayed on the top edit field. The edit field below the temperature simply displayed a state as "Normal Temp" or "CRITICAL Temp". When the temperature reaches above 25 °C, not only does the state change to "CRITICAL Temp", but the piezoelectric speaker also plays a light tone for one second. We chose this threshold because it was very easy to demonstrate the system working. Had we chosen a critical temperature such as 39 °C, it may have required a larger source of heat such as boiling water. Finally, the application stops recording either after ten seconds have elapsed, or the user presses the "stop" button. Note, that if the user allows the application to run for the entire ten seconds, it generates a plot featuring the temperature readings at intervals of 0.5 seconds. The GUI can be seen in figure 3a, and the code snippet for the "start" button can be seen in figure 3b.

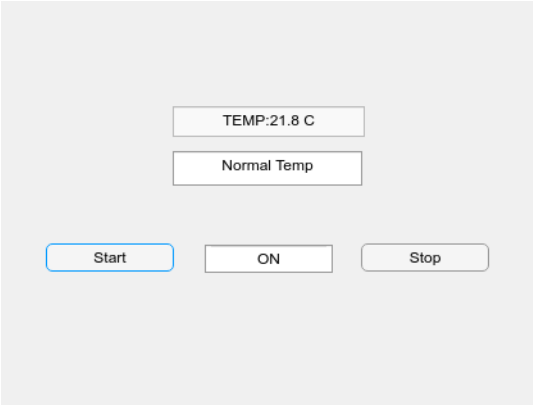

Figure 2a. Matlab GUI

```
temp = 1.0; % Holds the temperature reading.
iterations = 0; % Count iterations to know when to stop.
tempArray = [];

% Show that the thermometer is now on.
app.progStatus.Value = 'ON';

while(strcmp(app.progStatus.Value, 'ON') && iterations < 21)

    % Read output voltage from temp sensor, and convert to temp in C.
    rawVoltage = readVoltage(a, tempPin);
    temp = (rawVoltage - 0.5) * 100;

    % Buzz the speaker if the temperature is above 25C, and show
    % the state on the GUI.
    app.tempText.Value = strcat(strcat('TEMP:  ', num2str(temp, 3)),
    if(temp >= 25.0)
        playTone(a, speakerPin, 1500, 1);
        app.tempState.Value = {'CRITICAL TEMP'};
    else
        app.tempState.Value = {'Normal Temp'};
    end

    tempArray = [tempArray, temp];
    pause(0.5);
    iterations = iterations + 1; % Increment iterations count.
end
```

Figure 2b. Matlab GUI code snippet

Results

Measuring the ambient temperature with the sensor proved to be relatively accurate, but as the temperature sensor data sheet said, there tended to be a +/- 2 °C discrepancy with some of the readings. As stated before, our application graphs the results of ten seconds worth of temperature readings, with 0.5 second intervals. To test the accuracy of the sensor, we tested the

TMP36 under two conditions: ambient temperature only, and the temperature when held by a human hand. Because of the +/- 2 °C accuracy of the sensor, we noticed that there was a lot of fluctuations in the data readings, so we had to smooth out the graph so it doesn't end up looking jagged. The results can be seen in figures 3a and 3b. Although graph in figure 3a shows a lot of fluctuations, it only occurs between 20 °C and 22 °C. The graph in figure 3b clearly shows an upward trend for the temperature reading, which makes sense as it is heating up from the human hand.

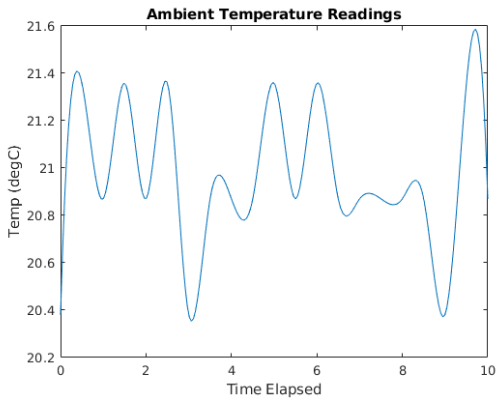

Figure 3a. Ambient Temperature Readings

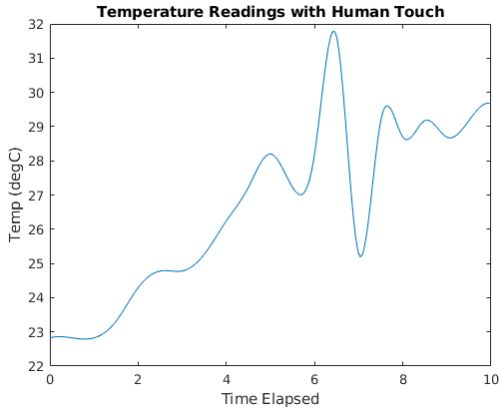

Figure 3b. Human Touch Temp Readings.

Conclusion

In this lab, we created a basic digital thermometer simply using an Arduino Uno, TMP36 temperature sensor, and a piezoelectric speaker. Although the real-time values displayed on the GUI as well as the piezoelectric speaker buzz gave a good idea of the temperature of the room, there was still a large amount of variation. This means that in order for the temperature sensor to be used in the field, it will have to go through a filter that will average the values so that outliers hardly affect the actual outcome. If for example, a machine will dispense water after a certain threshold, it should not immediately dispense it every time it reaches the threshold, otherwise it might alternate on/off because of the variation defined in the TMP36 data sheet.

Discussion

In this lab, our group was exposed to the basics of Matlab as well as Arduino programming. We were able to apply the experience from Wiki 1 and 2 directly into this laboratory assignment. Wiki 1 taught us the basics of interfacing with the Arduino using Matlab, whereas Wiki 2 taught us the basics of interfacing with analogue sensors. This lab required us to use the experience gained from both of those Wikis, as well as needing to learn how to create a basic Matlab GUI. No one in our group had any experience with GUIs in general, so it was interesting to see how all those components come together to form a light GUI.

Demonstration Video Link: https://www.youtube.com/watch?v=mKnY6khY12o

References

1. https://www.allaboutcircuits.com/technical-articles/how-piezoelectric-speakers-work/
2. https://cdn.sparkfun.com/datasheets/Components/General/cem-1203-42-.pdf

3. https://cdn.sparkfun.com/datasheets/Sensors/Temp/TMP35_36_37.pdf
4. https://www.mathworks.com/videos/app-designer-117921.html