October 23, 2015

# 1 Coffee cooling : Part B : automated data acquisition

*Experiment designed by Peter Crew, Navot Arad and Dr Alston J. Misquitta*

**IMPORTANT:** You should have made and calibrated your diode thermometer before coming to this section. Make sure you have done this and check to see that your thermometer is accurate before attempting to use it.

[The Arduino programme for this experiment is provided at the end of the assignment and is available online as for reference].

You should now have a working digital thermometer. In the next section (Part C) you will need to record cooling data for four different fluids. This takes time and is tedious, so here you will build an automatic data acquisition system for your thermometer. The idea is simple: rather than use a digital multimeter (DMM) to read out the voltage, and hence the temperature, we will use the Analogue to Digital converters (ADC) on the Arduino to read the potential difference across the 10kΩ potentiometer. In principle this is a simple step: all that needs to be done is to connect Arduino pins A0 and A1 (the 'A' indicates that these are analogue pins, that is, they are able to read an analogue signal like a voltage, and convert it to a digital signal) to the points where the DMM currently is (points I-1 and I-2 in fig. 1). The difference in the readings will then be proportional to the potential difference. This does not work well as the ADCs on the Arduino convert a voltage range of 0V to 5V to a value from 0 to 1023 (as the ADCs are 10bit converters and 10bits corresponds to a range of values from 0 to $2^{10} - 1 = 1023$). However, the voltage we expect across the 10kΩ potentiometer is only in the range 0mV to 100mV. This range would utilize only values 0 to 20 ($= 0.1 \times 1023/5$) on the ADC. This small range of values would give us very large uncertainties, something we need to avoid.

## 1.1 Op-Amps

One solution to this problem is to amplify the signal by a factor of 50 as that would convert the 0mV to 100mV range we expect across the 10kΩ potentiometer to 0V to 5V, which would fully utilise the range on the Arduino. We have constructed the amplifiers for you using two sets of operational amplifiers. There are two op-amps in each of the black chips in shown in fig. 2. We use operational amplifiers, or op-amps, as they can amplify signals with a minimal input current. The input resistance of an op-amp is typically a few MΩ, like the DMMs you have used in the Electrical experiment. Additionally, the amplification characteristic is usually fairly linear which is ideal for a good amplifier.

You will not be building the amplifier as you will not have the time to do so. Instead, we will supply you with a pre-built amplifier.

**Connections**:

- Connect the power (5V and ground) to the amplifier.
- Connect the two 'bridge' inputs (I-1 and I-2) to the points marked 'bridge'. You will need to experiment with the order of these connections. It will all work one way and not the other.
- Before connecting the Arduino, check to see that the amplifier works. To do this, connect a second DMM at the outputs A0 and Ground (this is the common ground of the circuit) of the amplifier.

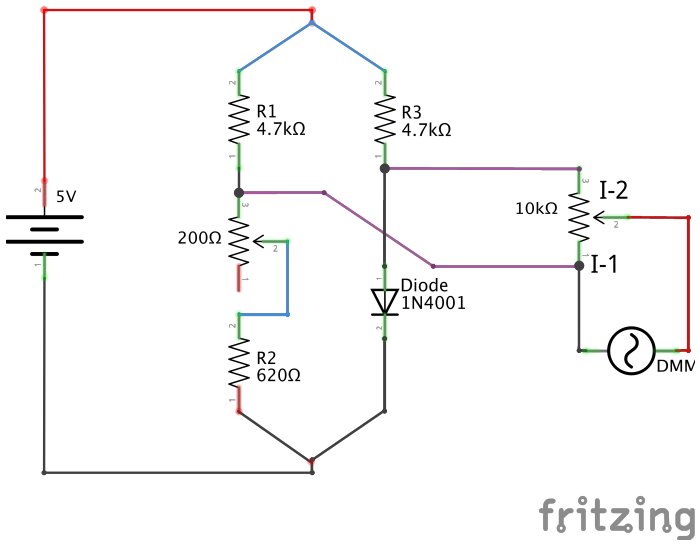


Figure 1: The Wheatstone bridge thermometer circuit with the points I-1 and I-2, across which the voltage needs to be read, indicated. The DMM shown reading the voltage across these points will be replaced by an Arduino Uno via an amplifier stage.

- Power it up. Check the voltage at the output. Is it nearly 50 times the input voltage? If yes, your circuit is functional. If not, check your connections.
- Adjust the potentiometer on the amplifier to make the gain almost exactly $A = 50$. Note the uncertainty on the gain.

## 1.2 Amplifier and Arduino

Once you are sure that the amplifier works (That is, the second DMM between the A0 output of the amplifier and ground shows a voltage that is 50 times the voltage across the bridge.), you can proceed to connect the Arduino Uno to the circuit. Essentially, the Arduino will take the place of the second DMM you have connected across points A0 and the Ground. With a suitable program, it will record the voltage at specified time intervals. A schematic view of the Arduino Uno is shown in fig. 3. The Arduino gets its power from the computer via the USB cable, so no additional power is needed. But it needs to be grounded to the common ground for it to be able to read voltages. This is why you needed to pass the ground to the Arduino.

**Connections:**

- Connect ouput A0 of the amplifier to pin A0 on the Arduino.
- Connect ground output of the amplifier to any of the Ground pins on the Arduino.
- Connect the Arduino Uno to the computer using the USB cable.
- You do not need to disconnect the second DMM. It can serve as a useful check on the readings of the Arduino.

Once this is done, upload the program (listed below, but please use the version on the SCM Wiki page) to the Arduino. The program will cause the Arduino to record temperatures in Celsius every 30sec. It works

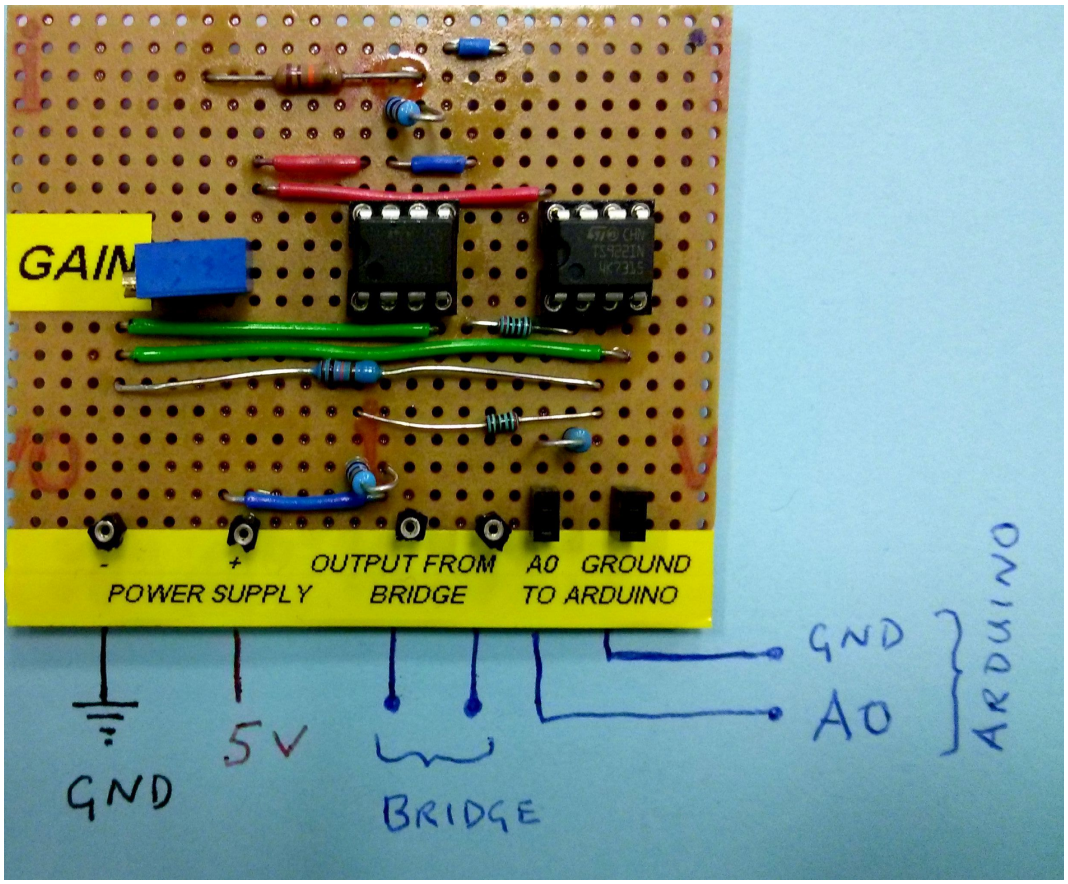




Figure 2: Image of the pre-built op-amp amplifier with input and output pins indicated. The variable potentiometer can be used to fine tune the amplification to be $A = 50$. The voltage difference across inputs I-1 and I-2 in fig. 1 is amplified and output is between the output at A0 and the common ground.

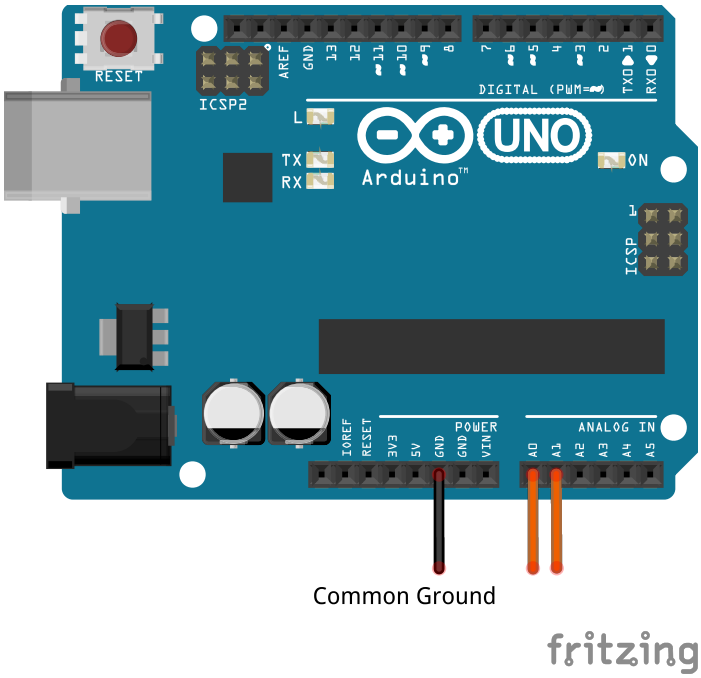


Figure 3: Bread-board view of the Arduino. You will use only the Ground and pin A0. We will not use pin A1 in this experiment.

continuously and so the reset button on the Arduino needs to be pressed to re-start it. Data will be displayed on the serial monitor. Ask a demonstrator for help if you are unsure of how to start the serial monitor. Before relying on the data recorded by the Arduino make sure it is recording the correct temperature (i.e., check the output against the reading on the alcohol thermometer). Once you are satisfied it is, proceed to part C of this experiment.

The Arduino will display readings on the Serial Monitor (on the computer screen). You may cut-and-paste these readings into a file for your records. Make sure you include these readings in your report!

### 1.2.1 Reflection

What are the sources of error in this setup? What do you expect the uncertainty on the temperature readings to be? Is the uncertainty limited by the thermometer? Or the calibration step? Or the amplifier? Or the Arduino? What will the random and systematic errors be? Is the amplification of the amplifier really $A = 50$? How would you test this? If it is not 50, what effect would the error in the amplification have on your results?

Consider these questions (and others you may have) **while you proceed with Part C** of this experiment.

## 1.3 Other Information

Larger versions of the circuits used can be found on the SCM Wiki page where the Arduino program needed to take the readings may also be found. A listing of the program is also provided in sec. 1.3.2.

### 1.3.1 The Dual Op-Amp pin diagram

The pin diagram for the Dual Op-Amp (TSS922IN or equivalent) used in the amplifier is shown in fig. 4.

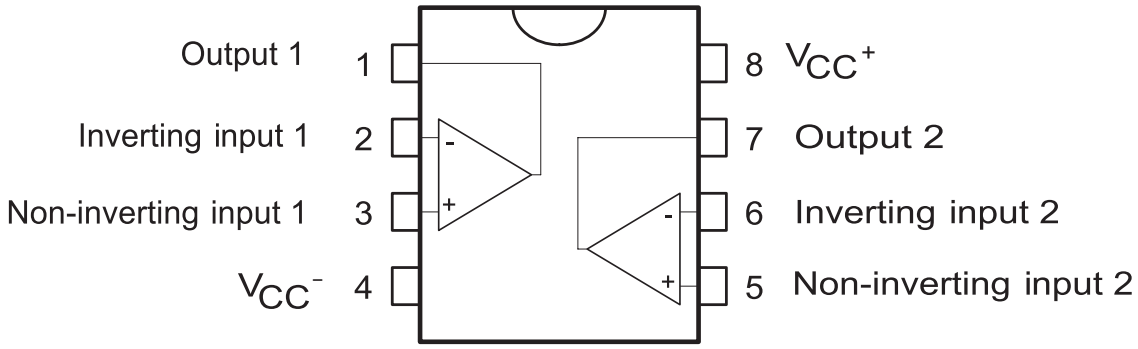


Figure 4: Pin diagram for the TS922IN dual op-amp.

### 1.3.2 The Arduino program

This will read out the temperature to the serial monitor in degrees Celsius. Readings are printed out every 30sec. To smooth-out random fluctuations, readings are averaged for 1 sec.

This program is listed here for reference only. Please use the version from the SCM Wiki page!

```
/*
OpAmp Thermometer
written by: Navot Arad, Queen Mary University of London
*/

int InputPin = 0;           // A0 is the amplified signal pin

float temp = 0;             // variable to store data from pin A0
float AmpVoltage = 0;       // Use float to convert data from A0 to voltage
float G = 50.0;             // Amplification factor
float scaleG = 0;
float conversion = 0;       // Conversion rate to degrees
float average = 0;

void setup() // Only runs once after board is reset
{
  Serial.begin(9600); //Rate at which data is sent to serial monitor
  Serial.println("Time    Temperature"); // Sends everything inside " " as a string to the
      serial monitor
}

void loop() // Runs continuously on repeat
{
  // average temperature over first second
  for (int i = 0; i < 100; i++) {
    temp = analogRead(InputPin);   // Read data from the input pin (A0) on a 10 bit scale
    scaleG = G + ((temp - 1023) * 1.5 * 0.00147);
    conversion = (1023 * scaleG)/50; // conversion factor
    AmpVoltage = temp *(100.0/conversion); // Converts signal to temperature
    average = average + AmpVoltage;
    delay(10);
  }

  Serial.print(millis()/1000); // Sends value stored on AmpVoltage
                               // to serial monitor and starts new line
  Serial.print("    ");
  average = average/100;
  Serial.println(average,1);
  average = 0;
  delay(29000); // 29 second delay before loop runs again
}
```

# 2 Coffee cooling : Part C

**HINT:** You should have built your calibrated thermometer and got the automated data acquisition with the Arduino working before coming to this section. Additionally, please read the paper the Rees and Viney before you start this part of the experiment.

Rees and Viney found that black and white coffee cooled at different rates, and sought to explain this. We are not asking you to attempt an explanation, but to repeat some of Rees and Viney's measurements and comment on whether your results agree with theirs and if not, what differences you find. You should be able to make the measurements and draw the graphs in one laboratory period — you may also have time to make some of the additional checks mentioned by Rees and Viney.

We suggest that, after checking that your thermometer still records the ice and boiling points correctly, you measure the cooling curves of:

- 200 ml of plain water, brought to the boil and poured into the china mug.
- Black coffee made by adding 200 ml of water to a level teaspoon of granules in the mug.
- White coffee made by adding 20 ml of cold milk to a freshly-made mug of black coffee (200 ml again), stirring, and pouring out 20 ml to leave 200 ml. Do not re-boil the coffee.
- Black coffee made by adding 20 ml of cold water to a freshly-made mug of black coffee (200 ml again), stirring, and pouring out 20 ml to leave 200 ml. Do not re-boil the coffee.

- Can you say why the fourth procedure might be informative?
- Use the digital thermometer to record the ambient room temperature.

## 2.1 Writing your report

Your report should describe what you did and the conclusions you draw, but it should not be as long as Rees and Viney's — about half the length is sufficient. It **must** be written using a word processor, a handwritten version is **not acceptable.**

The report must include:

1. A title.
2. Author's name and affiliation.
3. An abstract, which is a brief summary of a few lines including results but not too detailed.
4. A short introduction — what are you describing, and why did you do it?
5. A brief summary of the theory (you can refer to other publications for this, e.g. 'It is shown by Rees and Viney (1) that ...'). It is unacceptable to copy verbatim from the paper. You should present your results in your own words. Do not include theoretical details that are not relevant to your experiment, especially if you do not fully understand them. When you refer to the paper by Rees and Viney you should cite it appropriately.
6. A brief description of your digital thermometer.
7. A brief description of how you used the Arduino to automate the data collection.
8. A brief description of what you did and measured.

9. A summary of the results, together with calculated quantities. Show raw data as graphs (much more informative than tables), while derived quantities (such as time constants) can conveniently be tabulated. Use log rather than linear plots where appropriate.
10. A discussion of the significance of the results, including any uncertainties due to measurement precision (errors) and whether or not differences found are meaningful.
11. A short conclusion.
12. A list of references with name(s) of author(s), journal name and volume number or book title and publisher, page number(s), and date.

Marks will be deducted for poor spelling (use a spell checker), lack of clarity (proof read and be scientifically objective), and poor presentation. We will go over these points and answer questions about the reports in the lectures.

Remember to proof read and spell check your report. Look at it with print preview to check that it looks o.k. before you print it. Remember to save your work frequently, and keep a separate (backup) copy on your own USB stick.

If you need help with word processing ask a demonstrator or laboratory technician. Print your final copy well before the hand-in deadline, since the printers may be very busy (or down!) just before the deadline!