Introduction

In this activity, students will familiarize themselves with the basic operations of the Arduino ADC. They will use the ADC to read the voltage from the power supply and then determine the range and resolution of the ADC in voltage. Then they will connect one of the temperature sensors built earlier in the semester to the ADC. They will then use the ADC to read out the temperature sensor and determine its range and resolution.

Materials List

1. Arduino
2. Programming Laptop
3. Variable power supply (0-5V Range)
4. Multimeter
5. Temperature Sensor PCB (Diode Version recommended)

Basic ADC Readout

1. Our program for this activity will be relatively simple. We need a program that reads one of the ADC pins and prints to the serial monitor. The flowchart for such a program is shown in Figure 4.
2. First, write a program to read the voltage from Pin A0 and upload it to the Arduino. The code should look like what is shown in Figure 5.
3. Upload your code to the Arduino without hooking anything up to the analog pin.
4. Look at the output of the Serial monitor and take note of the ADC value. You should see it drift around somewhere in the ADC's range, and it will most likely not be at 0. Because the pin is floating, the input will be somewhat random. You should remember this behavior. Just because the ADC is not 0 does not mean it is correctly connected to an input signal.
5. Turn on your power supply and ensure it is turned down to ~0V output.

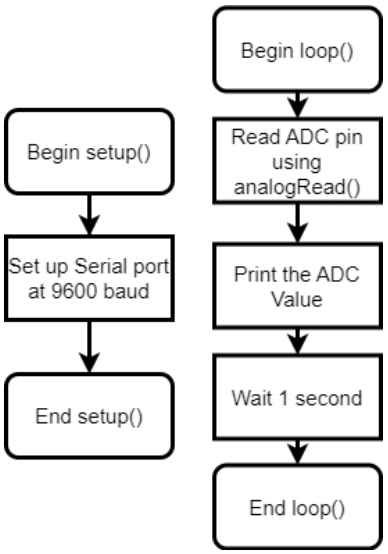

Figure 1: Flowchart for ADC readout.

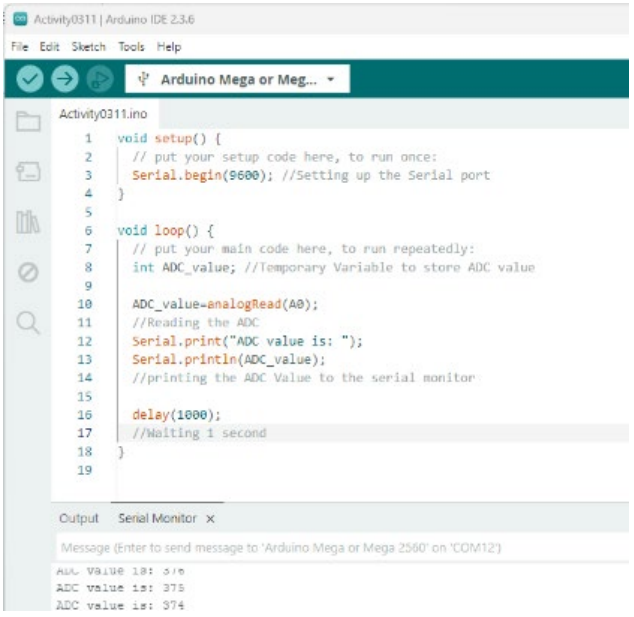

Figure 2: Sample code for ADC readout

NOTE: The analog pins have a maximum voltage of 5.5V. Applying a voltage greater than that can damage the Arduino.

IN PARTICULAR, VOLTAGES >6V CAN SHORT THROUGH THE ARDUINO AND SEND THE UNREGULATED VOLTAGE BACK THROUGH THE USB CABLE AND POTENTIALLY DAMAGE THE COMPUTER ATTACHED TO THE ARDUINO. TAKE CARE NOT TO APPLY EXCESSIVE VOLTAGE TO THE INPUT PINS.

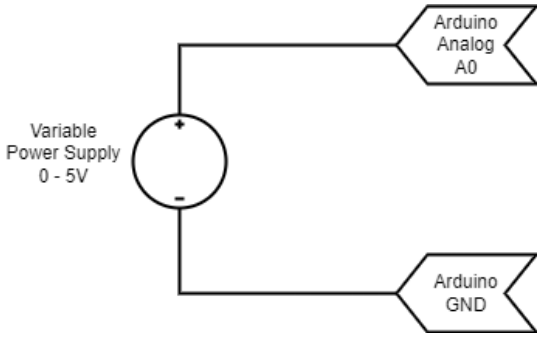

Figure 3: Schematic for ADC readout direct from power supply.

6. Connect the power supply to the Arduino's analog pin A0 and the ground pin of the Arduino, as shown in Figure 2. We recommend using solid jumper wires and alligator clips to do so.
7. Slowly raise the power supply voltage until the ADC is reading >0. Use your multimeter to measure the supply voltage. Record the voltage and the ADC value in your lab notebook.
8. Repeat this measure 4 more times over the full range of the ADC(0-1023) by raising the supply voltage. Try to space your measurements roughly every 250 ADC units apart.
9. NOTE: We do not want to include measurements at 0 and 1023 because unless we are, once we exceed those values, the ADC does not change. We have no way of knowing how far outside the range we are.
10. Now plot your data with ADC on the X axis and Voltage on the Y axis, and perform a linear fit to the data.
11. Find the equation of the line for your ADC data. This gives us a calibration from ADC to Voltage. We can use this equation to determine the range and resolution of our ADC in voltage.
12. To find the minimum value by plugging in the minimum ADC value, which is 0.
13. To find the maximum value by plugging in the minimum ADC value, which is 1023.

ADC	Voltage
10	0.07
241	1.18
570	2.77
752	3.65
986	4.78

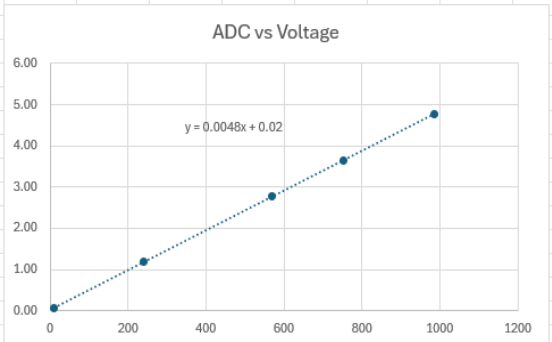

Figure 4: Sample ADC vs Voltage plot. For this set of data minimum voltage is 0.02V, the maximum voltage is 4.96V, and the resolution is 0.0048V.

14. Finally, to find the resolution, we just need to multiply the ADC resolution (which is just 1 ADC unit) times the slope.
15. Record the range and resolution for pin A0 in your lab notebook.

Table 1: Power Supply Voltage ADC readings for two different ADC Pins

Power Supply Voltage	ADC Value (A0)	ADC Value (A1)

16. Turn the power supply voltage back down to ~0V and move the positive lead from A0 to A1.
17. Repeat steps 7 through 15 to find the calibration, range, and resolution for pin A1.
18. Compare the performance of the two ADC pins; you might find that they are similar but not exactly the same. This is why we need to calibrate a sensor ADC system as a whole; if we move a sensor, we do not want to assume it will have the same performance.

Temperature Sensor ADC Readout

1. Now, let's connect the ADC to a real sensor. Connect the Temperature Sensor PCB you built earlier in the semester to the Arduino, as shown in Figure 6. Either the thermistor or the diode circuit can be used, but it is recommended to use the diode version.
2. You may need to add an output lead wire to the pin 6 column if you did not do so previously. We will be powering the PCB from the Arduino for this exercise. See Figure 5 if you are not sure which connection the schematic is referring to.
3. Change your code back to reading pin A0 and upload it to the Arduino.

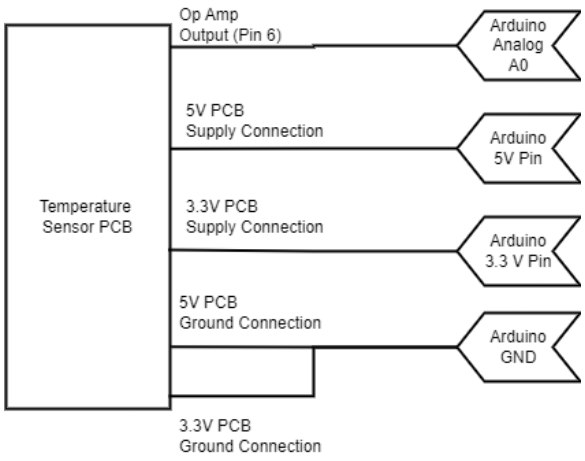

Figure 5: Schematic for connecting the temperature sensor PCB to the Arduino for ADC readout. For simplicity, the PCB has been represented as a single block. The full schematic can be found in A02.07 Figure 1 if needed.

4. You should now be reading the output of the temperature circuit.
5. You should have previously set the amplifier on the circuit to output ~0V at 50 °C and ~5V at 0C so the ADC value should be somewhere in the middle of the 0-1023 range. While watching the ADC output, briefly heat up the diode sensor with your hand, a heat gun, or a

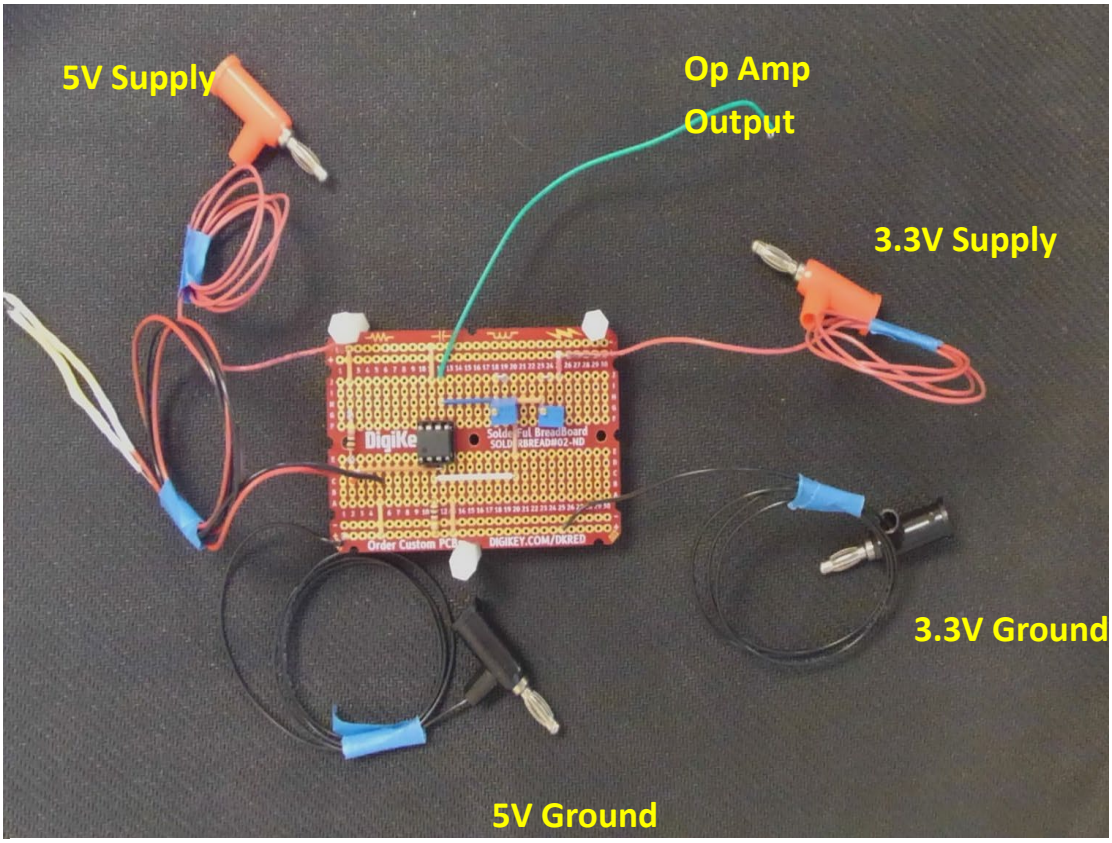

Figure 6: Assembled temperature sensor PCB with the external connections labelled. The easiest way to connect to the Arduino would be to use alligator clips, clipping one end on the connector and clipping the other on jumper wires installed in the Arduino headers,

water bath. You should see the ADC value drop (since the output voltage should decrease with an increase in temperature). If you do not see a response, check your connections and use your multimeter to determine if your circuit is still working properly.

6. Once you are sure your circuit is working properly, perform a temperature calibration by taking ADC measurements at 4 different temperatures at least 5 degrees apart. Be sure to record both the temperature and the ADC value in your lab notebook.
7. Similar to the voltage measurements, plot your data with ADC on the X axis and Temperature on the Y axis, and find the calibration equation by performing a linear fit on the data.
8. Now, calculate the Temperature Range and Temperature resolution using the same method you did for voltage. Notice that, due to the negative slope, the maximum temperature should occur at 0 ADC, and the minimum will occur at 1023. Additionally, for resolution, we simply take the absolute value; we are concerned with the magnitude of the change, not its sign.

9. Now make a small change in the gain or offset of your temperature circuit. For example, decrease the gain slightly by reducing R_{gain} by 500 Ohms. Repeat the calibration process for the new gain and offset.

Table 2: ADC Values for the Temperature Sensor PCB, the first two columns

Temperature	ADC Value (Initial)	Temperature	ADC Value (After Adjustment)

10. Compare the range and resolution of the new gain and offset to the original. What happened with the size of the range(Max – Min) compared to the resolution?
11. Hopefully, you can see how you could adjust the amplifier to reach a desired range and resolution in order.
12. Now, as the final step, modify your ADC readout program so that it prints out calibrated temperature instead of ADC units. **HINT: You probably want a function that accepts the ADC value as an argument, applies the calibration equation, and then returns the result of the calibration equation.**

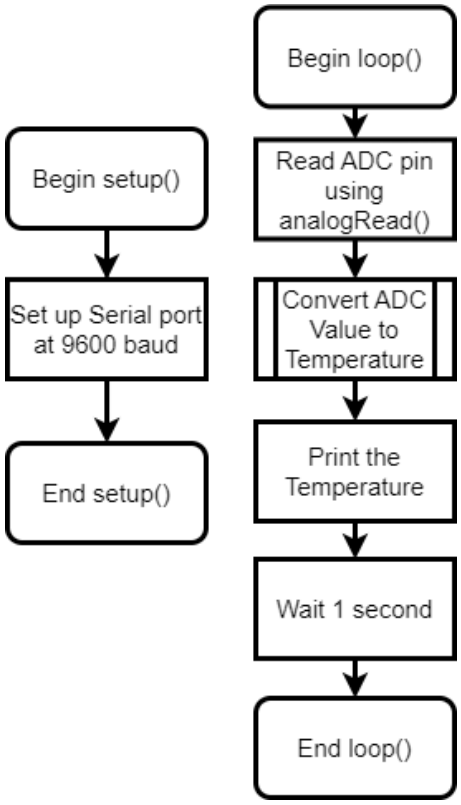

Figure 7: Flowchart for Temperature readout. The conversion function should take the ADC and apply the calibration you determined for the