ME 120
October 19, 2019

Analog Input on Arduino UNO
Gerald Recktenwald
gerry@pdx.edu

Analog Input from a Potentiometer

A potentiometer is an adjustable voltage divider that can be used as a method of user input to an Arduino program. The left side of Figure 1 is a photograph of a typical potentiometer with pins that fit into a breadboard. The right side of Figure 1 is a schematic representation of the potentiometer.

A potentiometer has three external electrical contacts. By turning the knob, the electrical resistance between the middle contact and the two external contacts is adjusted. The change in resistance allows the potentiometer to function as a variable voltage divider as depicted by the schematic in the right side of Figure 1. The resistance between terminals A and B is fixed. In the typical potentiometer circuit, a supply (or input) voltage is applied across terminals A and B. The output voltage is measured between terminals W and B or terminals W and A.

Figure 1: A typical rotary potentiometer (left). Schematic for a potentiometer (right). The middle pin is called the "wiper".

Figure 2 is a schematic of a potentiometer that can supply a variable voltage to an analog input pin of an Arduino. Corresponding to Figure 1, terminal A is connected to a 5V supply and terminal B is connected to ground. The wiper is connected to an analog input pin. Figure 3 is a physical representation of the wiring with the wiper of the potentiometer connected to analog input pin `A2` of an Arduino.

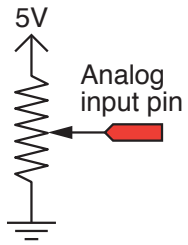

Figure 2: Potentiometer connected to an Arduino analog input pin.

The `potentiometer_input.ino` program in Listing 1 reads and prints the voltage between the wiper and ground for a potentiometer wired as in Figure 3. The the value in `potReading` will be an integer between 0 and 1023. To convert to voltage, multiply the reading by 5/1023.

Note that in the Arduino program, the conversion factor is written `5.0/1023.0` to force a floating point division. If the conversion factor was written `5/1023`, the compiler would perform an integer division resulting in zero, and the value of `potVoltage` would be 0.0, regardless of the value of `potReading`.

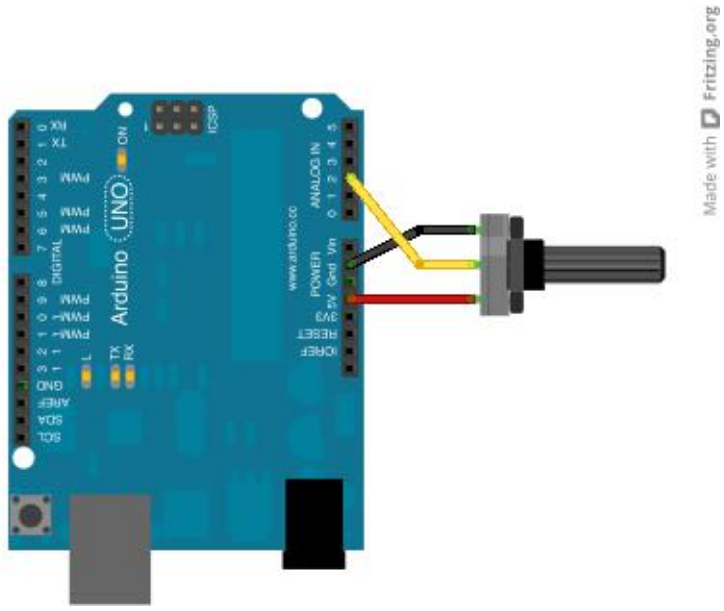

Figure 3: Arduino wiring of a potentiometer circuit for analog input on pin A2.

```
//  File:  potentiomter_input.ino
//
//  Read a potentiometer and print the raw analog input value
//  and the voltage

void setup()
{
  Serial.begin(9600);
}

void loop()
{
  int    potPin=A2, potReading;
  float  potVoltage;

  potVal = analogRead( potPin );
  potVoltage = potVal*5.0/1023.0;

  Serial.print( potVal );
  Serial.print("  ");
  Serial.println(potVoltage)
}
```

Listing 1: Arduino program to read and display voltage across a potentiometer with the wiper connected to analog input pin A2.

Scaling of Arduino Analog Input Readings

On a 10-bit scale there are 1024 numbers starting with 0 and ending with 1023.

Question: To convert a numerical value returned by the `analogRead` function to a voltage, should the scaling be 5V/1023 or 5V/1024?

Answer: The correct scaling is 5V/1023.

The Arduino Uno can read voltages on one of six analog input pins, which can be referenced by the built-in labels `A0`, `A1`, ..., `A5`[1]. The maximum input voltage is 5V, and the analog input readings are returned by `analogInput` as integer values with a 10-bit resolution. The maximum numerical value for a (unsigned) 10-bit number is 1023. Therefore, when the maximum input of 5 V is applied to one of the input pins, the maximum numerical value for an analog reading is 1023. This idea is explored further in the Appendix on page 5.

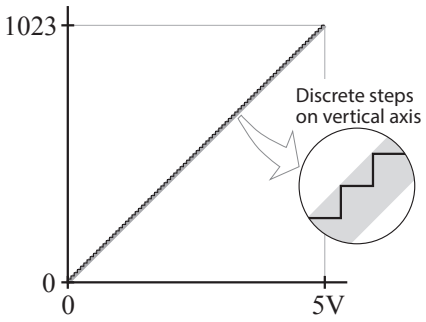

Figure 4: Scale 5V into 10-bit values.

Analog to Digital Conversion

The relationship between a 5V input range and the 10-bit values returned by `analogInput` can be visualized with the plot in Figure 4. The nominal relationship is a line from $(0, 0)$ to $(5, 1023)$. However, as depicted by the inset in Figure 4, the input value is a continuous voltage and the output values are limited to one of 1024 discrete values.

The `analogRead` function is a handy interface to the *analog-to-digital* or ADC conversion process that happens in the Arduino microcontroller. The general topic of ADC is deep and fascinating, but we will only briefly discuss *resolution* as the most salient characteristic.

The resolution of and ADC process is the smallest voltage interval that can be distinguished on the output scale of the converter. For a maximum output range of 5V, the 10-bit output scale for `analogRead` on an Arduino UNO discrete (integer) scale has a resolution of

$$\frac{5\,\mathrm{V}}{1024\,\mathrm{values}} = 4.8 \times 10^{-3}\,\mathrm{V}$$

per unit value of output from `analogRead`. In general, the resolution, δV, of an n-bit ADC is

$$\delta V = \frac{V_{\max}}{2^n}$$

where $V_{\max}$ is the maximum voltage accepted by the electronics in the ADC. Table 1 provides a sample of ADC resolutions.

Table 1: ADC resolution as a function of n, the number of bits in the conversion.

n	δV (mV)
10	4.9
12	1.2
14	0.31
16	0.076
18	0.019

[1] On an Arduino UNO the input pins can also be referenced by the integers 0, 1, ... 5. On other microcontrollers, the analog input pins may have different numerical values. The built-in labels `A0`, `A1`, etc. provide a convenient way of writing code that will work on a variety of microcontrollers.

Exercises

Conceptual Exercises

1. In many example Arduino programs, analog readings are made with statements like `x = analogRead(A0)`, `x = analogRead(A1)`, `x = analogRead(A2)`, ... What are `A0`, `A1`, `A2`, etc.? Should users define `A0`, `A1`, `A2`, etc. as global variables?

2. In the example program in Listing 1, why are `potPin`, `potReading` and `potVoltage` not defined as global variables?

3. The maximum safe voltage for an input signal on any analog input pin is 5V. How does the design of the circuit in Figure 3 guarantee that the 5V limit will not be exceeded?

   Note: Many new microcontrollers are limited to 3.3V.

4. Some microcontrollers have 12-bit analog input resolution. What is the appropriate scaling between the analog reading and voltage for a 12-bit analog input?

Coding Exercises

Write Arduino sketches to perform the following actions. These problems build upon each other so chunks of the code can be reused. Save your solution to each problem as a separate sketch. Figure 5 has schematics of the circuits used in the programming exercises.

5. Read a photoresistor and print the raw (10-bit) reading to the Serial Monitor.

6. Read a photoresistor and turn on an LED when it is "dark".

7. Read a potentiometer and turn on an LED when the voltage is less than 1.0V and greater than 4.0V.

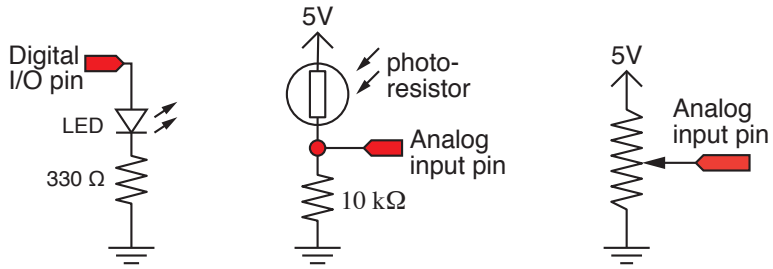

Figure 5: Circuits used for demonstration of `analogInput`.

Appendix

On the Appropriate Scaling from 10-bits to Volts

Consider a 4-bit analog to digital converter. An unsigned 4-bit number can have the values 0, 1, 2, ... 15 as depicted in Figure 6. Although there are 16 numbers that label points on the line, there are only 15 equal-length segments. Therefore, the scale factor for a 4-bit analog-to-digital converter is 5V/15 because the 5V range is divided into 15 intervals.

By analogy, a number line starting at 0 and ending at 1023 has 1024 values that define 1023 intervals. Therefore, the correct scaling for a 10-bit analog to digital converter having a maximum input range of 5V is 5V/1023.

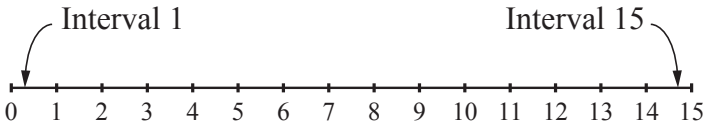

Figure 6: A 4-bit number line.