ECE 1001/1002 Introduction to Robotics
Lab #6: Analog Input

Objectives

Introduction to the analog input, reading voltage

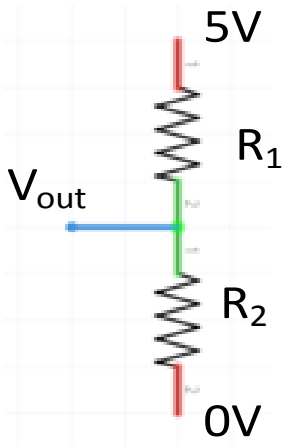
So far you have used digital inputs to receive information from the environment or yourself. These digital inputs can convey significant and important information, but there is not much subtly to the information; it's on/off, true/false, yes/no. We used the photoresistor to decide if it was light/dark in lab 2, a yes/no kind of input. The photoresistor can actually give a more nuanced answer of just how light or dark it is. In this lab, you will use a temperature sensor to give specific information on the environment; you will find the temperature it senses in degrees, not just is it hot/cold.

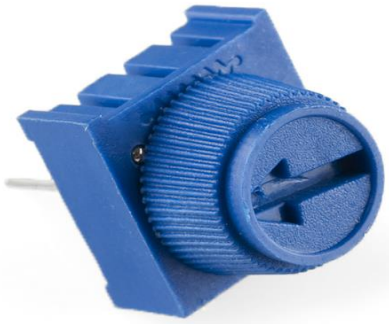
Measurements like this are an analog not digital measurement. Arduino has a sensitive analog input capability…a set of six Analog to Digital Converters (ADC). That means the Arduino can measure six separate analog inputs. It actually measures voltage, so you can say it has six sensitive voltmeters. How sensitive? Well, these ADCs are 10-bit, which means they can provide 10 bits of resolution. A 10-bit number can be between $2^0 = 1$ and $2^{10} = 1024$. The minimum input voltage is 0V, and the maximum input voltage is 5V, the Arduino measures voltage in 2^{10}=1024 steps. The ADC resolution is then:

$$ADC\ resolution = \frac{5V}{2^{10}} = \frac{5V}{1024} = 4.8828 \ldots mV\ resolution\ (about\ 5\ mV)$$

If you've used a voltmeter to measure voltage before this may not seem great, but remember the Arduino has six voltmeters in a device which can do many other things and only costs $20. And it is plenty good enough resolution do make sensitive measurements.

There are many types of sensors which can be read by the Arduino, one kind is the photoresistor which is a variable resistor. Using two resistors in series, a variable resistance will provide a variable voltage. The basic circuit looks like this,





The Resistors R_1 and/or R_2 could be any value (depending on your needs), and for a sensor, one of them would be a variable resistor such as the photoresistor. The output voltage is given by "the divider equation":

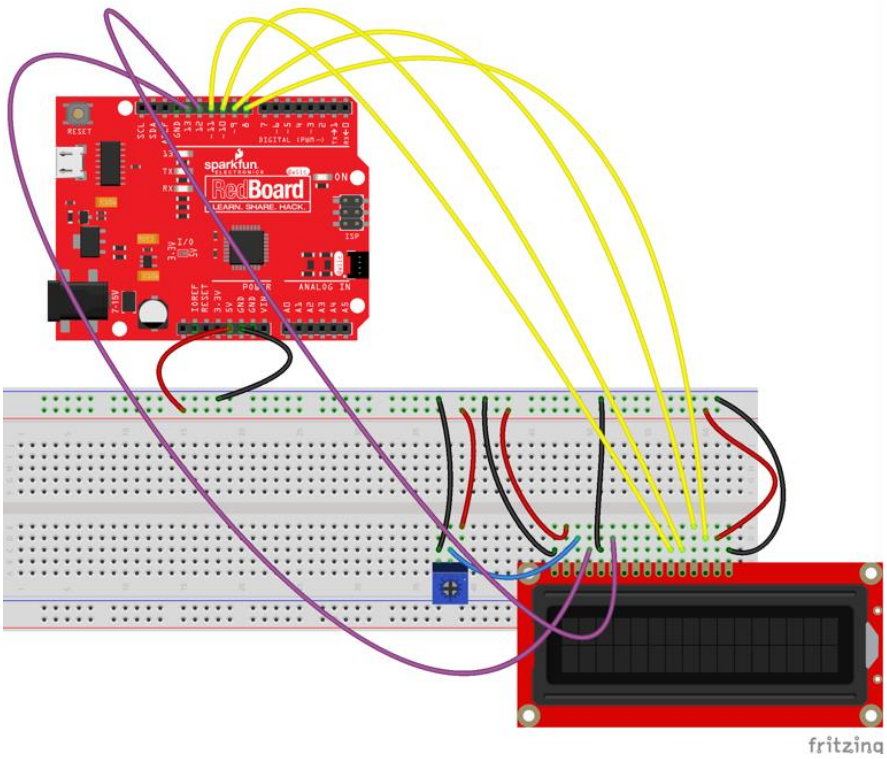
$$V_{out} = \frac{R_2}{R_2 + R_1} \cdot 5V$$

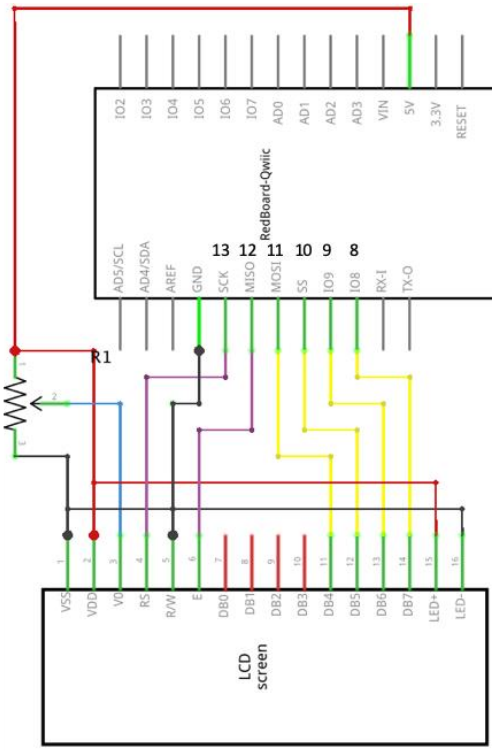
There is much more to learn about dividers, but this is enough at this point—the output voltage will vary between the minimum voltage 0V and maximum 5V depending on the ratio of resistances R_1 and R_2.

Requirements and Signoffs

Voltage Display Circuit

Build the Arduino display circuit as in Lab-4, the wiring diagrams and table are repeated here:





Arduino pin	Wire color suggestion	Display pin	display pin description
GND	black	1	GND
5V	red	2	5V
Connect to Pot Middle pin	blue	3	Contrast adj
13	purple	4	Register select
GND	black	5	R/W select
12	purple	6	Enable
11	yellow	11	Data d4
10	yellow	12	Data d5
9	yellow	13	Data d6
8	yellow	14	Data d7
5V	red	15	Backlight 5V
GND	black	16	Backlight GND

Voltage Display Software

Open the Lab6_analog_voltage_display.ino program. This program is similar to lab 4, except there are some lines for displaying analog voltages, for instance the upper row of the display shows the raw ADC value (between 0 and 1023)

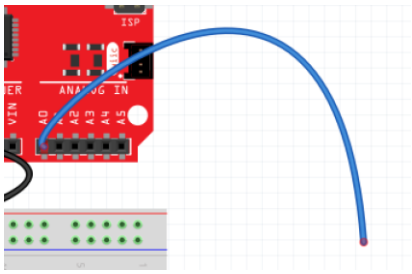
```
lcd.setCursor(0, 0);
lcd.print("Raw Value:");
lcd.setCursor(11, 0);
lcd.print( analogRead(A0) );
```

the bottom row of the display shows ADC output converted to mV, we just multiply the ADC value (between 0 and 1023) by 4.8828mV, the resolution of a single step in the ADC:

```
lcd.setCursor(0, 1);
lcd.print("mV:");
lcd.setCursor(4, 1);
lcd.print( analogRead(A0) * 4.8828); //   0 ≤ A0 ≤ 1023,   0 ≤ A0 ∗ 4.8828 ≤ 5,000
```

The lower line will display voltage in mV, so 5V will display as 5000. Finally, the program delays for 100ms just so the display won't bounce around too much; you could change that value as you please.

Attach one end of a jumper wire to the A0 pin and leave the other free, like this:



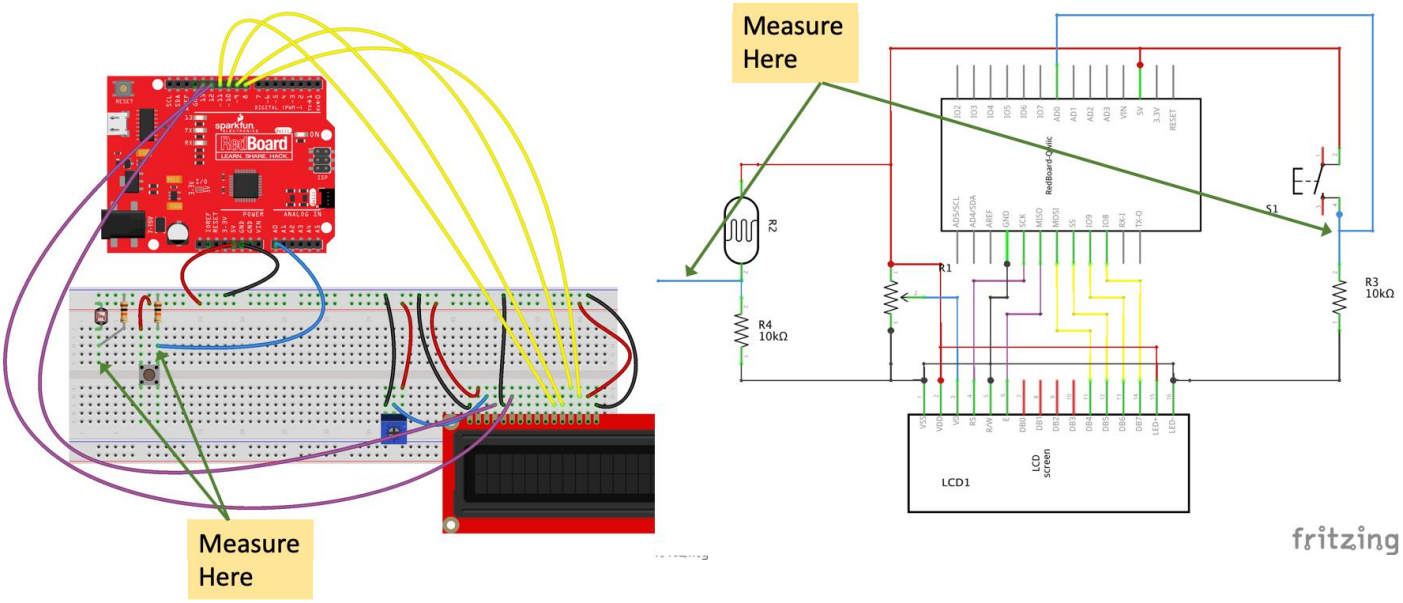
Upload the program to the Arduino, and the display will probably bounce around showing what might look like random noise. Without the wire connected to anything, it will be an antenna receiving various radio interference, which will in fact be somewhat random. Now plug the loose end of the wire into the GND rail on your breadboard, it should read about 0mV. Then move it to the 5V rail, it should read near 5000mV. Plug it into the Arduino pin labeled 3.3V, it should read about 3300mV. You've got a voltmeter!

Button and Light Sensor Circuits

Add a button in series with a resistor (10k) between 5V and ground, and a similar circuit using the photoresistor, as shown below. The blue wire attached to pin A0 on the Arduino can be connected between the photoresistor/10k Resistor or the switch/10k resistor, as noted "measure here"

Measure Light Sensor and Switch Voltages

Plug the Arduino in (reference Light sensor Lab-2 for more sensor information) and look at the output.



First see what happens as you push the button with the A0 measure wire is attached between the switch and resistor (as shown above).

What values does the voltage change between?

Does this make sense?

Would this work as a digital input?

Then move the A0 measure wire to the measurement location for the photosensor and resistor. Shine bright light on it, make it dark, make it various levels of light dark.

How does the voltage change?

Does it make sense?

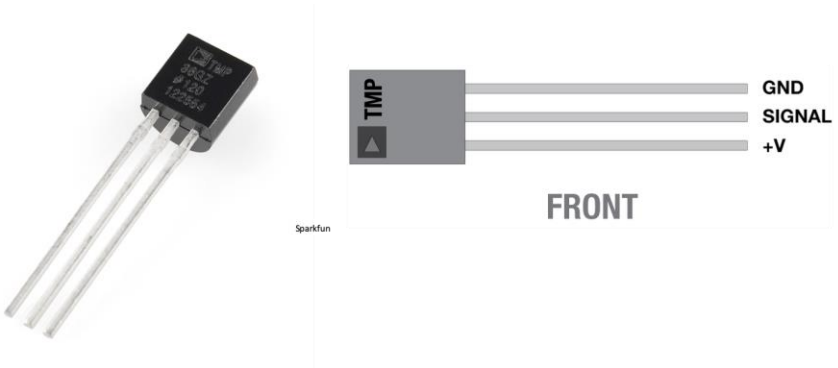
Does there seem to be a correlation between the light and voltage?

What voltage represents really bright light?

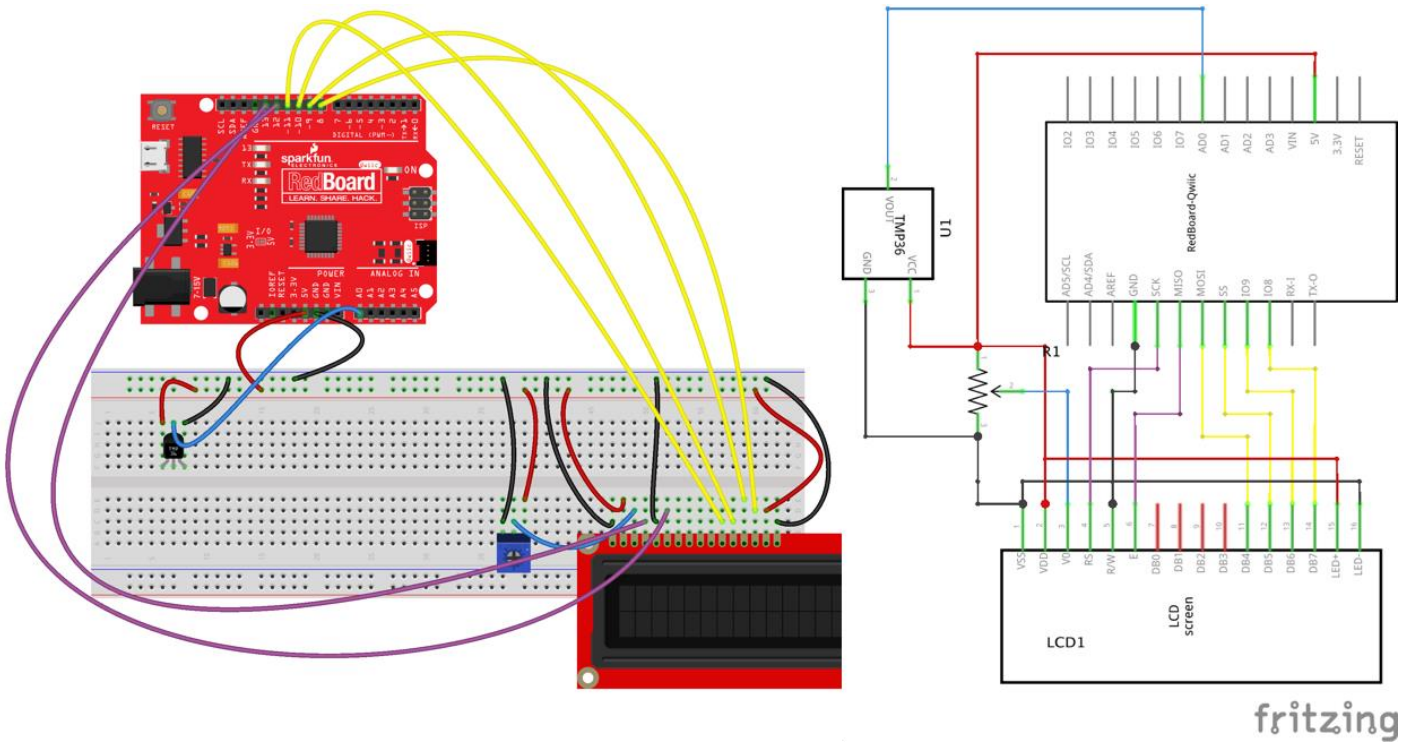
What voltage represents a dark situation?

Temperature Sensor Circuit

Change the circuit so that temperature can be measured, as below. The temperature sensor looks like this:



This sensor is connected from 5V (+V) on the diagram, to GND. The middle wire is connected to Arduino pin A0 for voltage measurement. There are transistors and various circuits inside of this temperature sensor which cause it to output a voltage between 0V and 5V related to temperature (it's actually a "band-gap" sensor, based on fundamental quantum mechanical properties of Silicon, but that's for another class!) Be very careful to connect it properly, looking at the flat side, the right lead is GND and the left lead is 5V. If you connect it wrong it will get VERY hot, possibly damaging the sensor, the Arduino, your fingers, and maybe your pride.



Temperature measurement Software

Open the program Lab6_temp_display.ino and look at it. Now three variables are declared:

```
float voltage = 0.0;
float TemperatureInC = 0.0;
float TemperatureInF = 0.0;
```

These are floating point variables, which means they have decimals, they can be negative or positive. Right in the declaration a value is given to each, in this case the three variables are all set to 0.0. It's usually good

practice to assign a default value of some kind to all variables when you declare them. Otherwise, there could be a strange random number stored in the memory location allocated to each variable.

In the main loop, the first thing to happen is the TMP365 voltage is read by pin A0:

That voltage is then converted to degrees C. The TMP36 reads 750 mV at 25C, so:

$$Temp\ C = \frac{\left(TMP36_{voltage} - 500\right)}{10}$$

And then the temperature is converted to degrees F. From there the voltage and temperature in Celsius is displayed for 0.5 seconds, then the voltage and temperature in Fahrenheit is displayed for 0.5 seconds, then the temp is measured again and displayed in the loop.
The code is:

```
voltage = analogRead(A0) * 4.8828125;
TemperatureInC = (voltage - 500)/10;
TemperatureInF = TemperatureInC * (9.0 / 5.0) + 32.0;
```

Try putting your finger on the TMP36 sensor (touch the plastic, but don't touch the metal leads).

How does the temperature respond?

How often does the temperature update?

Release your finger, how long does it take to cool back to room temperature?

Does the temperature seem correct?

Your instructor will either have you print off a sign-off sheet to sign or have a roster to check your name off. To receive credit for this lab, you *must* get the sign-off.