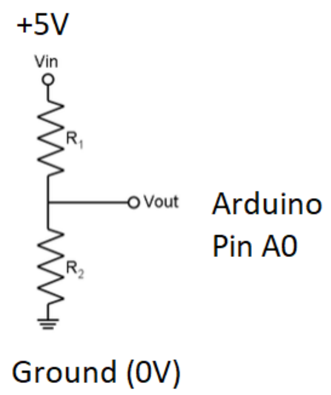
CSIS 10C   Lab 3 Electronics and Sensors

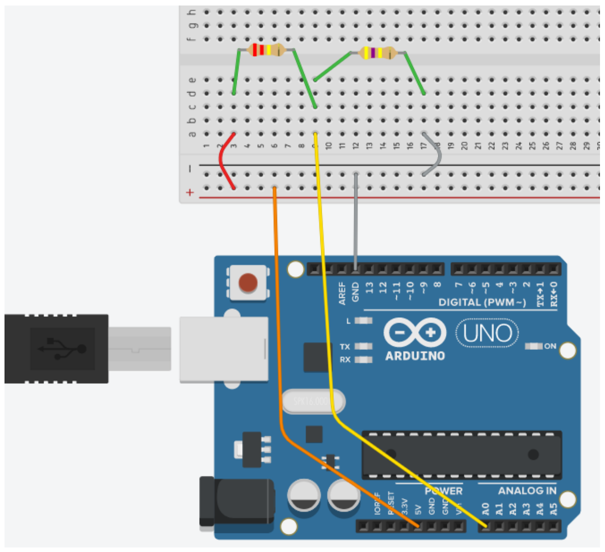
Part 1: Resistor Divider



Make a copy of this Tinkercad Circuit for Part 1. Use the folder 1_2_*AnalogRead* for this exercise.

Here at right is the schematic (circuit diagram) for the Arduino circuit you can see below. We have two resistors connected between +5 and ground. The midpoint connection is connected to Arduino pin A0, which can read analog input voltage values and convert them to a reading from 0 (= 0V) to 1023 (= 5V).

Build the resistor divider circuit above with your breadboard and Arduino as shown below. Pick two resistors at random, making sure they have different values (if you need extra resistors, ask the instructor). Make sure **R1 is the resistor that has one end connected to +5V**, and **R2 is the resistor with one end connected to Ground (0V)**. Then answer the following questions:



1. If R1 is the resistor with one end connected to +5V, Which resistor is R1 (left or right)? _________
2. What is the color code of R1? _____________
3. What is the resistance of R1? _____________
4. What is the color code of R2? _____________
5. What is the resistance of R2? _____________

(If you are using an MPC kit, the only possible resistor values are: 100, 220, 470, 1k, 10k)

6. Use the voltage divider formula to predict the voltage at the connection point between R1 and R2. Because of the yellow wire, this is also the voltage we will read at pin A0 on the Arduino.

   The formula is Vpin_A0 = 5V * R2 / (R1 + R2)

   Using your resistor values above, what is Vpin_A0? ____________________

7. The Arduino AnalogRead command senses the voltage at an analog input pin and converts it to a number between 0 and 1023, with 0 representing 0V, and 1023 representing 5V. We can predict the number the Arduino will read given the voltage we calculated in step 5, using the linear scaling formula:

   Numeric_Reading = Vpin (1024/5)

   Using this formula, predict what number the AnalogRead command will read at pin A0, given that this pin is wired to the connection point between R1 and R2.

   Predicted numeric reading = ___________________

8. Now run the program in the *Labs* > *3B* > *1_2_AnalogRead* > using *make flash*.

   What is the numeric reading you get? _________________

   If the number differs a large amount from your prediction, go back and check your calculation in step 5. It's possible you have R1 and R2 mixed up in your circuit or your formula. If the number differs by a small amount, why do you think that is?

9. You can also convert a numeric reading from the AnalogRead command back into a voltage using the formula:

   Vpin_A0 = Numeric_Reading (5/1024)

   Convert your reading from step 7 back into a voltage______________

   What is the voltage difference between your prediction and your measurement? ________

   What is the percent error between your prediction and your measurement? ___________

   You can determine this using the formula $\%\text{err} = \dfrac{\text{Measure} - \text{Predict}}{\text{Predict}}$

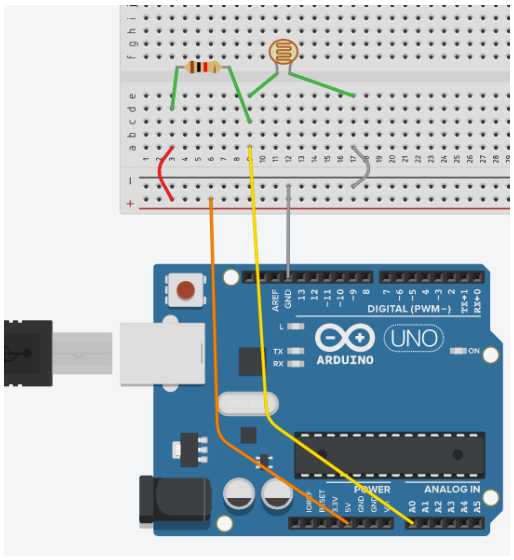
Part 2: Light Sensor

Make a copy of this Tinkercad circuit for Part 2. Use the folder 1_2_*AnalogRead* for this exercise.

We are now going to experiment with using a dynamic resistance in a resistor divider. Look for a photocell in your kit. This is a two-wired component that looks something like the **object at right**. **Photocell -->**

A photocell is a resistor that changes its resistance based on how much light is illuminating it. When it receives a lot of light, the resistance gets small. When the light is dim, the resistance gets large.

10. In your circuit, replace the R1 resistor with a 1KΩ or similar resistor, and replace R2 with the photocell. Your circuit should look like the one at right.



11. Run *Labs* > *3B* > *1_2_AnalogRead* > using *make flash* again and record the maximum and minimum numeric readings you get on the display when the photocell reads different light levels.

    Minimum_______________

    Maximum_______________

12. Challenge - Based on your measurements, and the voltage divider formula (in step 5), can you determine the range of resistance values that your photocell is changing between? This is an algebraic problem, where you solve for R2 given R1 = 1000 and Vin is the voltage inferred from the numeric readings in step 10.

    Max R2 =

    Min R2 =

    Hint: use this formula:

    $$Vpin\_A0 = Numeric\_Reading\ (5/1024)$$

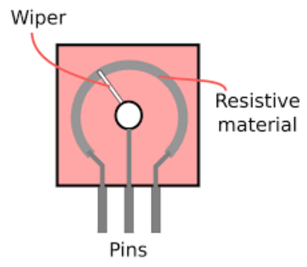
    Given Vpin_A0 above and R1 = 1000,

    solve this equation for R2:

    $$Vpin\_A0 = 5V * R2 / (R1 + R2)$$

There are many sensors of physical properties (temperature, pressure, humidity) that work on this principle – some environmental change causes them to vary their resistance value. Because of this trait, they can be inserted into a voltage divider circuit so that we can sense the physical property they respond to with a microcontroller.

Part 3: Potentiometer (Variable Resistor Divider)



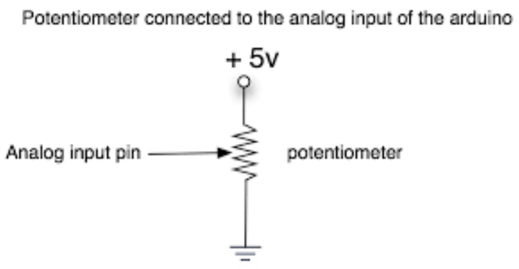
Make a copy of this [Tinkercad circuit for Parts 3 and 4](). Use the folder *3_Potentiometer* for this exercise.

Resistor dividers are one of the most common circuits you will work with. There is even a circuit component that has a resistor divider built into it. It's called a potentiometer, and is shown at right.

If you've ever used an electronic device with any kind of knob, slider or dial the control was most likely a potentiometer. In fact, many game controller joysticks use built in potentiometers to detect the position of the stick along one axis.

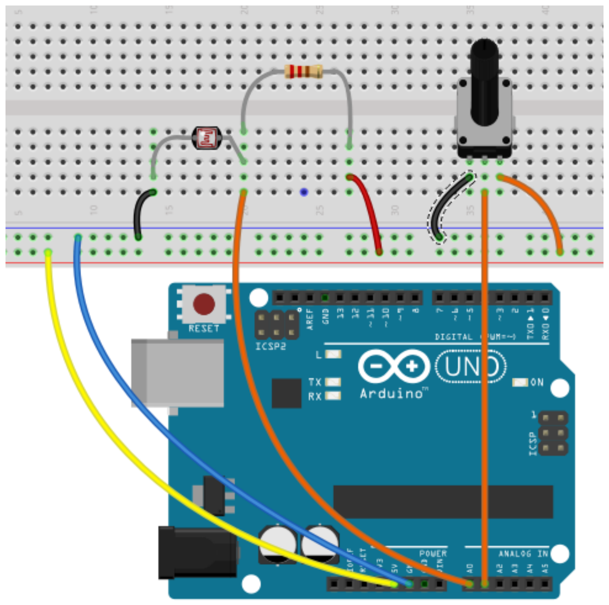
A potentiometer has 3 wires on it – the outer two connect between +5 and ground, and the inner pin connects to the input pin of the Arduino. By rotating the knob, we can control how much of the total resistance goes into R2 and how much goes into R1, allowing us to change the voltage at the center pin just by rotating the knob.

The schematic symbol at right shows even more clearly how a potentiometer implements a resistor divider.



In this part, we are going to add a potentiometer to our circuit and read its position with the Arduino.

13. First, add the potentiometer to the circuit. Your breadboard should look something like the one at right.



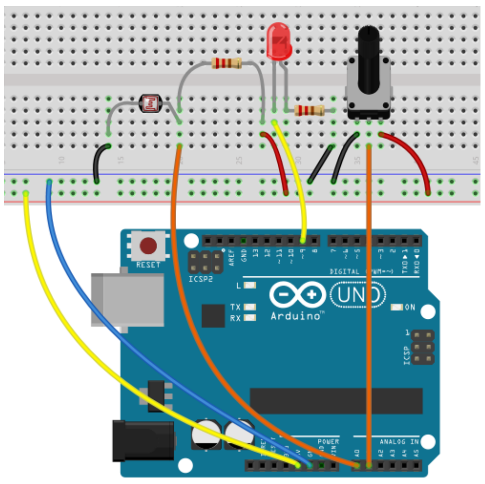
14. Using the code from 1_2_AnalogRead, modify the program to ALSO read the voltage at pin A1 and show it on the screen as well. Some code from a successful program is already in the file, use it to help guide you as to what to add.

    You should see two readings, one for the photocell circuit, and the other for the potentiometer circuit.

    Notice how they respond to different light levels and rotations of the potentiometer dial.

15. Add additional code to determine if the value is greater than the maximum value and or less than minimum values found for both the photocell and the potentiometer. Modify your printf statements, so the max and min values are printed along with the current value. These values will be important for the next exercise.

Part 4: LED Brightness Control



Now that we have two ways of reading information into our Arduino – one based on light levels, the other based on the position of the potentiometer dial, we can use these to control other components in our project. Since we've been working with LEDs already, lets start there (other things we will incorporate control of later will be speakers and motors). We can use these two inputs to control the brightness and the delay of a flashing LED.

In this exercise, the only existing code is the structure of the C program. Use the previous files along with new code to solve the exercise.

16. Wire up the LED following the diagram at right. Note that the input to the LED is coming from pin 9 on the Arduino. You can use other pins, but if you want to control the brightness, you have to choose a pin that has a ~ by it. These pins support Pulse Width Modulation using AnalogWrite that we explored in Lab 1B.

17. The analogWrite command takes a pin number and a value between 0 and 255 to determine what level of output to establish. It does this by transitioning the pin voltage between +5 and 0 V so rapidly it seems to average out to a value between the two extremes, based on the proportion of time it spends at each value.

    However, the analogRead command will produce a value from 0 to 1023. There is a mismatch in the number ranges between analogRead and analogWrite. So, in order to convert the two sensor input values into something you can send to analogWrite, we will need to do a linear transformation (and convert to an int) to get them into the correct range.

    In many situations, you would need to use a *map* function which maps or converts one set of number to another set of numbers. In this case, we have an easier situation. Our analogRead command produces a value from 0 to 1023 or 2^{10} and our digitalWrite requires a number from 0 to 255 or 2^8. To convert a number from 2^{10} to 2^8, we need to divide by 4, which is the same thing as shifting the number to the right, 2 places (2^{-2})! The code would look like this:

    ```
    int outputValue = sensorValue >> 2;
    ```

    (If this is confusing think of shifting decimal numbers, a shift of one digit to the left or right, will multiply by 10 or divide by 10, respectively. When we do this with binary numbers, the shift will either multiply (left shift) by 2 or divide by 2 (right shift).)

From the previous program, you probably saw a maximum value for the potentiometer to be 1023 and the minimum value to be 0. This makes sense as we see a full resistance swing using a potentiometer from maximum resistance to 0 ohms.

18. Now modify the code in 4_LEDControl to:
    1. Using the formula above and analogWrite, add statements to send to pin 9 a value based on the position of the potentiometer dial. You should be able to change the brightness of the LED by rotating the dial of the potentiometer.
    2. Add code to control the time delay for on or off of the LED based on the light level sensor reading from the photocell. You should notice the delay time of the LED changing based on the light level picked up by the sensor.

19. Now you have a program which will blink the LED at a rate, based on the photocell reading and will dim or brighten based on the potentiometer reading!

    When you are finished, save your code and upload it to the Lab 3 assignment page on Canvas.