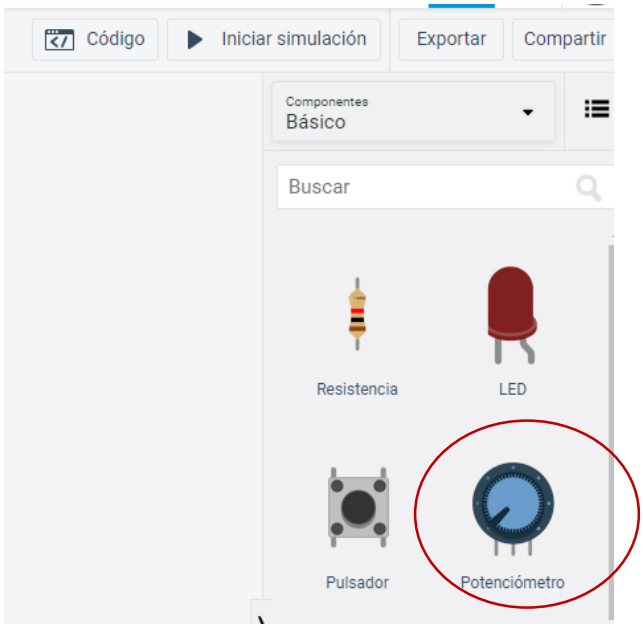
1. Analogic digital converter (ADC)

1.1. Introduction to the electronic component: Potentiometer

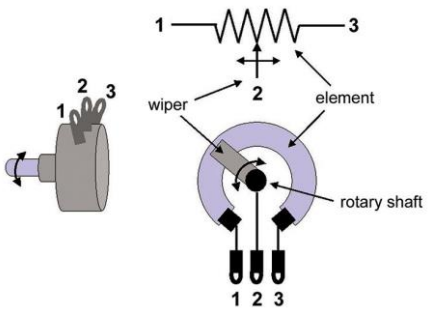
To carry out the following exercises we are going to use a new electronic component called "Potentiometer". Also a review of the topic "Voltage divider" is included as it is the simplest electronic topology to turn the resistance magnitude into a voltage. The voltage is the analog magnitude the microcontroller is capable to read thanks to the ADC peripheral.

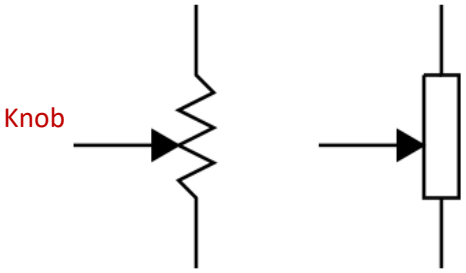
Tinkercad includes this component, you may find it at the "Basic components" menu



Potentiometer

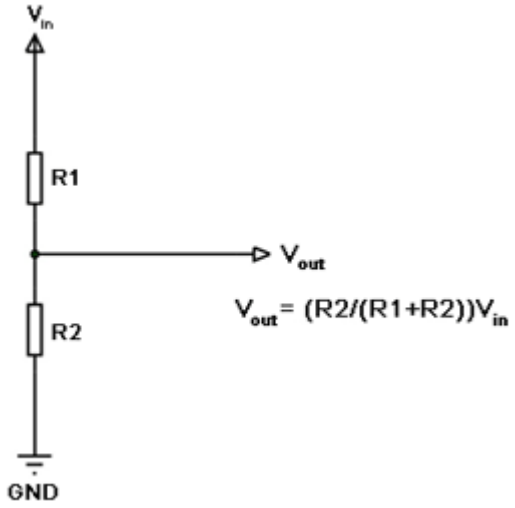
A potentiometer (trimpot) is a variable resistor. When powered with 5V, for instance, the middle pin outputs a voltage between 0V and 5V, depending on the position of the knob on the potentiometer. Internal to the *trimpot* is a single resistor and a wiper, which cuts the resistor in two and moves to adjust the ratio between both halves. There are usually three pins: two pins connect to each end of the resistor, while the third connects to the pot's wiper.



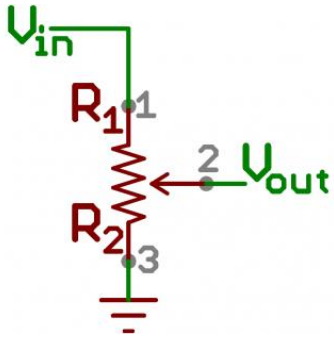


Voltage Divider

A voltage divider is a simple circuit that turns some voltage into a smaller voltage using two resistors. The following is a schematic of the voltage divider circuit.



A potentiometer is a variable resistor that can be used to create an adjustable voltage divider.



If the outside pins connect to a voltage source (one to ground, the other to Vin), the output (Vout) at the middle pin will mimic a voltage divider. Turn the trimpot all the way in one direction, and the voltage may be zero; turned to the other side, the output voltage approaches the input, 5V in this example. A wiper in the middle position means the output voltage will be half of the input.

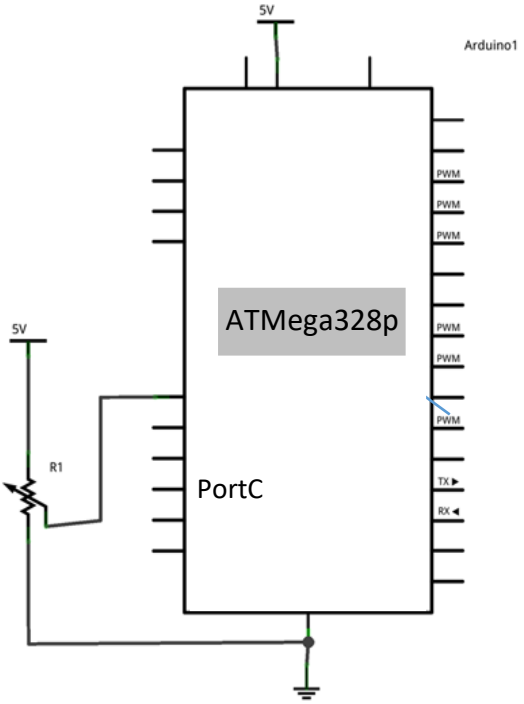
Source: https://learn.sparkfun.com/tutorials/sparkfun-inventors-kit-experiment-guide---v40/circuit-1b-potentiometer

To find out more click on https://cq.cx/interface.pl#8

Exercise 1

Let's write a code to read the voltage in the cursor of a potentiometer. The potentiometer's knob must be connected to PORTC, pin ADC1. Remember PORTC is connected to the ADC peripheral.

Also, connect four LED to the port D.



a) *The code must accomplish the following requiremenst:*

- *LED0 is switch on when the voltage is in the range [0-255].*
- *LED1 is switch on when the voltage is in the range [256-512]*
- *LED2 is switch on when the voltage is in the range [513-768]*
- *LED3 is switch on when the voltage is in the range [769-1023].*

TIPS:

*- It is recommended to create a function (Static inline) to **Initialize the ADC**:*

0.- Delete a previous configuration

1.- Select Vref by means a function argument. Use macro defines for the choices.

AVCC --> (REFS1-0 = "01") AREF --> (REFS1-0 = "00")

2.- Select the alignment format by means a function argument. Use macro defines for the choices.

ALIGN_LEFT --> (ADLAR=1) ALIGN_RIGHT --> (ADLAR=0)

3.- Set pre-scaler --> Prescaler 16MHZ/128 = 125 KHz --→ PRESCALER_128

4.- Enable ADC (ADEN = '1')

5.- Single Mode

*- Create a second function (file "functionsADC.c" for **setting and reading the channel** (by polling).*

- *Use at least one argument to pass the channel for reading.*
- *It is recommended to discard the first readout because the ADC is "calibrating".*

- *Do not forget to clear the flag previously to a new ADC conversion (reading) and clear a previous channel.*

b) *Add a new "main.c" file for printing the value on the screen by means of the USART (polling) instead of switch on the LEDs.*

TIPS:

- You need to use the "utoa()" function belonging to the library "stdlib.h".

(https://www.nongnu.org/avr-libc/user-manual/group__avr__stdlib.html)

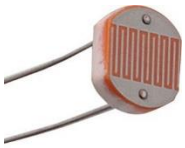
1.2. Introduction to the electronic component: LDR

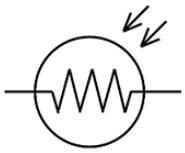
Photoresistor or LDR, acronym for Light Dependent Resistor, is a passive electronic component which has a resistance that varies depending of the light intensity.

The resistance is very high in darkness, almost high as 1MΩ but when there is light that falls on the LDR, the resistance is falling down to a few KΩ (10-20kΩ @ 10 lux, 2-4kOmega; @ 100 lux) depending on the model.

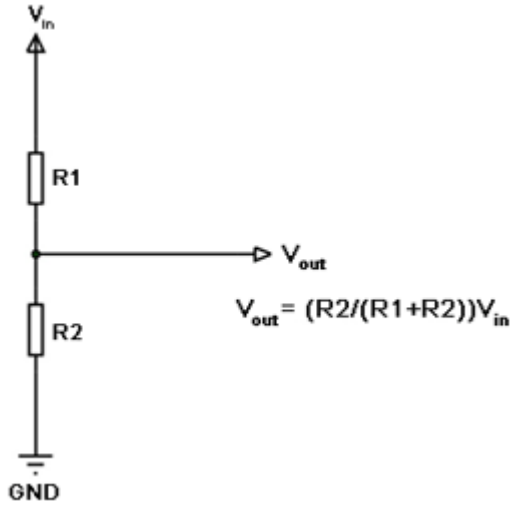
When light is incident on a LDR it usually takes about 8 to 12 ms for the change in resistance to take place, while it takes one or more seconds for the resistance to rise back again to its initial value after removal of light. This phenomenon is called resistance recovery rate.

LDRs are very useful in many electronic circuits, especially in alarms, street lights and more. It is used to turn ON or OFF a device according to the ambient light.



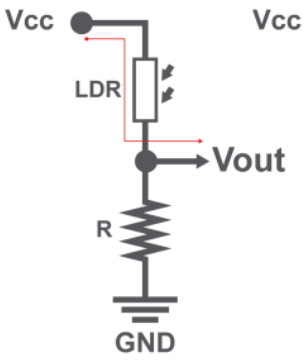


LDR is connected to a microcontroller by means of a circuit such as a voltage divider.

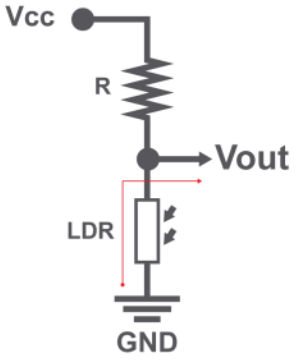


The midpoint of branch is taken to measurement. Suppose the resistance R2 changes, the Vout changes with it linearly.

What we are going to do here is to replace one of the resistors by a LDR. There are two choices:



Light → Vout HIGH



Light → Vout LOW

The sensor is one of the resistances in the Voltage divider. It can be at the top (R1) or at the bottom (R2), the choice is determined by when you want a large value for the output Voltage Vo:

- Put the sensor at the top (R1) if you want a large Vo when the sensor has a small resistance.
- Put the sensor at the bottom (R2) if you want a large Vo when the sensor has a large resistance.

The value of the resistor R will determine the range of the output Voltage Vo. For best results you need a large 'swing' (range) for Vo and this is achieved if the resistor is much larger than the sensor's minimum resistance Rmin, but much smaller than the sensor's maximum resistance Rmax.

You can use a multimeter to help you find the minimum and maximum values of the sensor's resistance (Rmin and Rmax). There is no need to be precise, approximate values will do. Then calculate the R value as follow:

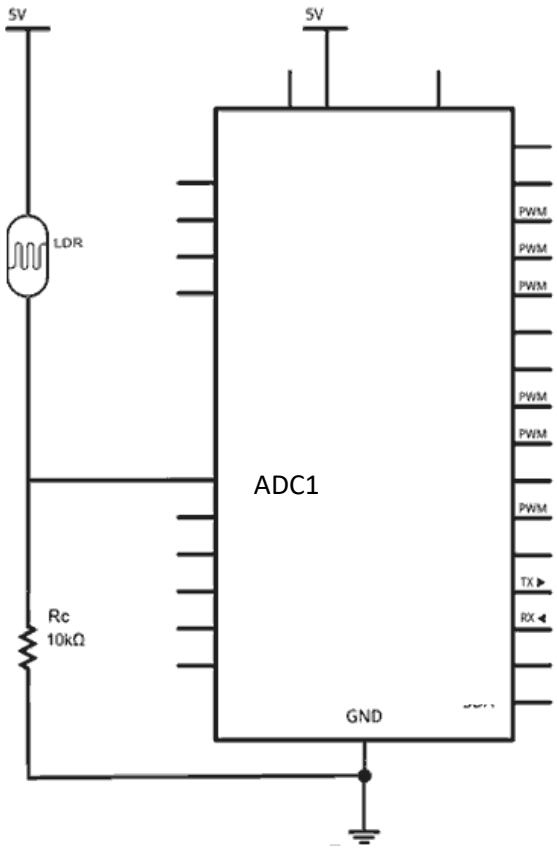
For example:

An LDR: Rmin = 100ohms, Rmax = 1MOhms, so R = square root of (100 × 1M) = 10K.

Exercise 2

"Twilight switch"

Connect a LDR and a 10k resistor as a voltage divider by using a breadboard. Connect the middle point to an input of the ATMEGA328, namely to an ADC input (PORTC). Also connect a LED with a current limiting resistor, 220 ohms, to a pin in PORTB.



Write a code to switch on the LED when the LDR detects the ambient light has fallen below a set point. Configure the ADC to work by means of an interrupt.

TIPS:

- *Add a static inline functions for enable and disable the interrupt.*
- *Write a "PARAMETERS.h" file including:*
  - *a SETPOINT parameter to modify easily this value.*
  - *a SAMPLE_TIME to set how often a sample is taken.*
- *Reuse the "MACROS.h" header file.*

Notes:

If you are using real components, not a simulator, you may watch the LED does not work as expected. Maybe it does not light on. The reason is how much ambient light is the LDR receiving. In a real world you should measure the resistance across the LDR either when it is exposed to the light in our room or covered by your hand, for example. Or better to measure the voltage across the LDR to know what is the maximum and minimum voltages in dark or light. Based on these values you can select the set point to change the LED state.

Exercise 3

Assembly a circuit to read a potentiometer and a LDR in two different channels of the ADC (polling or interrupts, you decide). Write a code to show a menu by means of the serial monitor (polling). The menu choices should be:

1.- Measurement of the LDR

2.- Measurement of the potentiometer

Tips:

- *In order to measure a more precise value, create a function to take several samples of the ADC. The function must have one **argument** to set the* **channel** *and the* **number of samples** *(1,2,4,8 or 16 samples) and return the average value of the samples taken.*
    - *Use a switch-case into the function to assign the shift right amount to calculate the average (division by power 2).*