Analog sensors

This example shows how to read results from analog sensors with Raspberry Pi Pico Microcontroller.

In the example below we assume that the analog sensor is connected to the GP27 (ADC1) Pi pin

The script's task is to read the voltage on the GP27 pin and interpret it accordingly to the characteristics and type of the sensor. In the case of a rotary potentiometer, the resistance changes, which translates into voltage accordingly. The same applies to other analog sensors: a soil moisture sensor, a precipitation sensor or an LM35 thermometer. Micropython uses a 16-bit ADC, which allows you to reach a level from 0 to 65535 (2^16-1)

Very often it works on the principle of proportion, e.g. in the case of an analog thermometer LM 35 1% of 65535 corresponds to 1 degree Celsius or 1%, i.e. for each degree Celsius there are about 655 ADC levels. Similarly in the case of a rotary potentiometer with a maximum resistance of 10 kilo Ohms rotary potentiometer, for 1 ohm there are about 6.5 ADC levels.
Of course, in the case of other analog sensors, sometimes the reading is more complicated and depends on the function graph of the sensor characteristic.

Below is a listing of a Python script that, with minor modifications, can be used with almost all analog sensors.

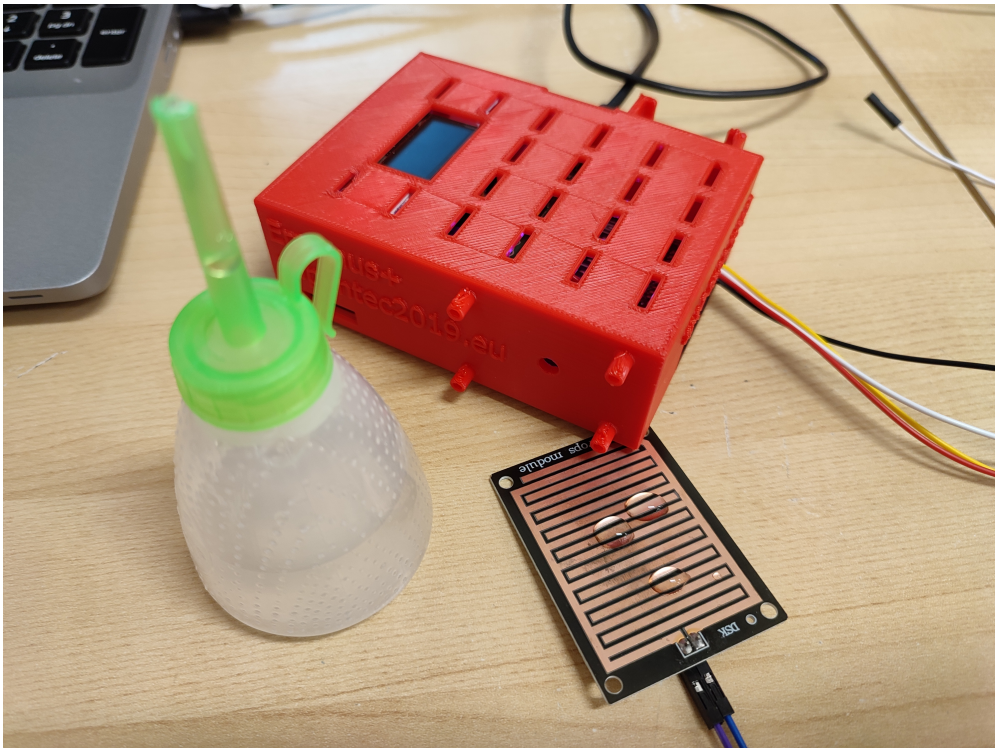
*Note. in reality Pico has a 12 bit ADC, which allows to reach one of 4096 levels, i.e. 0 corresponds to 0 Micropython ADC, and 4095 from Pico ADC corresponds to 65535 Micropython level (the actual reading from Pico ADC is multiplied by 16). Pico ADC is 4 times more accurate than Arduino Uno ADC (10 bit ADC)

Explanation: **ADC**-analog-to-digital-converter

```
1  from time import sleep
2  import machine
3  analog_value = machine.ADC(1)
4
5  while True:
6      sensor_raw = analog_value.read_u16()
7      print("raw:", sensor_raw)
8      sleep(2)
```

In the script below, the raw measurement results from the analog sensor are saved to the Pico board in a CSV file, which is a simple database format that can be used in many ways, for example for visualization and analysis

```
from machine import  ADC
from time import sleep
#sensor connected to pin GP27
#Raspberry Pi Pico
#save results in file readings.CSV inside Pico's memory
#This file format may be used with Excel
analog_value=ADC(1)
file=open("readings.CSV","a")
file.write("Measurement EU-Ballon Analog sensor \n")
file.write("No"+ "\t"+"result"+"\n")
file.flush()
no=1
while True:
    result=analog_value.read_u16()
    file.write(str(no)+ "\t"+str(result)+"\n")
    #print in consola
    print(str(nr)+ "\t"+str(result))
    no=no+1
    #save to the file without closing file:
    file.flush()
    #measurement every 5 seconds:
    sleep(5)
```



The picture below shows a reading using a rain sensor. The more water drops on the sensor, the lower the sensor resistance and the higher the voltage read on the ADC pin.