Instituto Politécnico Nacional
Escuela Superior de Computo

Learning unit: Instrumentation

Teacher: Ismael Cervantes de Anda

Final Project: Voltmeter

Team members:

- García Gonzaléz Aarón Antonio
- Pérez Sereno Ricardo Erick
- Ruiz Hernández Rodrigo Antonio

Delivery day: 12th June 2020

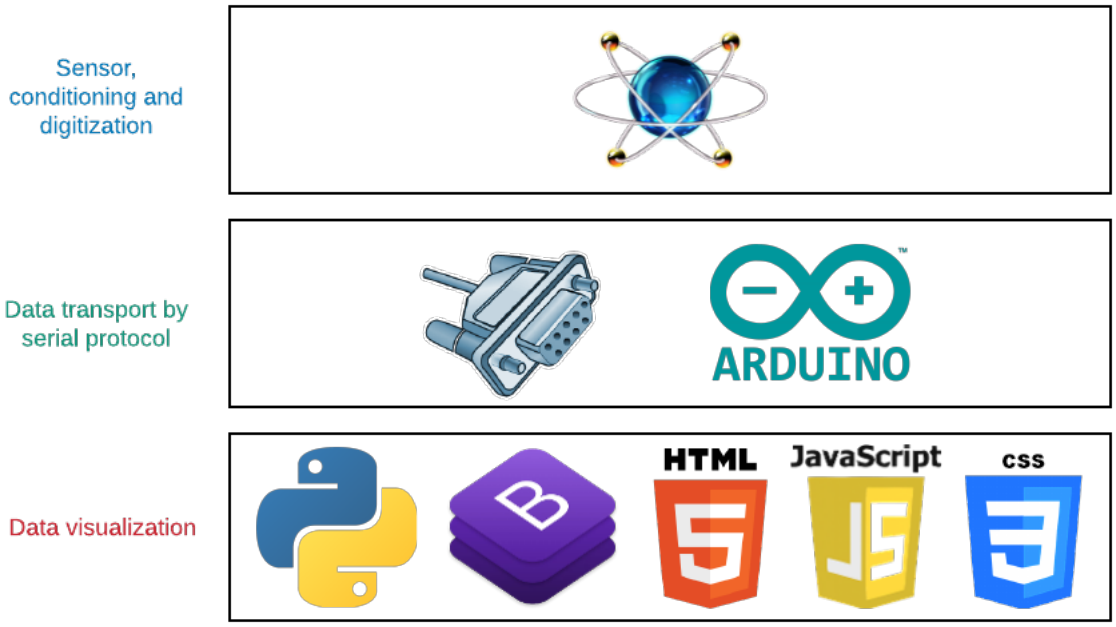
Index

Objective

The student must perform the sensor conditioning circuit depending on its nature, in the case of the present project, it is a sensor that delivers voltage, so it must only be conditioned to deliver the desired output voltage through a divider voltage and ensure its correct output with the high impedance that the voltage follower has, which will help to make the values without much noise, then you will have to digitize the voltage value delivered by the signal conditioning circuit, to finally be displayed graphically in a computer.

Software to use

For the virtual design of the circuit "Proteus" is used, for the serial connection "Virtual Serial Port Drive" and "Arduino" are used, for the transmission of data Python is used and for the graphic presentation HTML, CSS is used, JavaScript, Bootstrap, google charts and a web browser.



Introduction

Signal conditioning is a data acquisition process that is carried out using an instrument called a signal conditioner. This instrument converts one type of electrical or mechanical signal (input signal) to another (output signal). The goal is to amplify the signal and convert it to another format that is easy to read and compatible with data acquisition or machine control.

A signal conditioner helps to obtain accurate measurements, as an essential condition for the precision of data acquisition or machine control. These types of instruments can perform other additional functions.

On the other hand, the conversion of analog to digital signal is of vital importance, since computers work with this last type of data, that is why it is necessary to perform digitization in our circuit, for that we will use the ADC0804, also the algorithm that perfectly describes this process is that of successive approximations.

At the end, we will send said digitalized value to be processed with the help of a programming language to be able to visualize it in a more pleasant and efficient way.

Signal conversion

The main function of a signal conditioner is to collect a signal and transform it into a higher-level electrical signal. Signal conversion is often used in industrial applications that use a wide spectrum of sensors to make measurements. Due to the variety of sensors used, it may be necessary to convert the generated signals, so that they can be used by the instruments connected to the sensors. In principle, any signal from a sensor can be converted to any standard process signal.

Linearization

Certain signal conditioners could perform a linearization, if the signals provided by a sensor do not have a completely linear correspondence with the physical magnitude. To do this, they carry out a process of interpreting the signal using software. It is usual in the case of the time signals. This method is used to obtain greater accuracy, because not all sensors are fully linear. The parameters for linearization are evaluated during sensor calibration and are indicated in the sensor calibration protocol.

Amplification

The next step is signal amplification and the process of increasing the signal for processing or digitization. There are two ways to amplify a signal: increase the resolution of the input signal or increase the signal-to-noise ratio.
In signal conditioning different amplifiers are used for different purposes; These include instrumentation amplifiers, which are optimized to work with direct current signals, and which are characterized by high input impedance, high synchronous cadence suppression (CMRR) and high gain. Another example of a signal conditioner used in amplification is the isolation amplifier, which is designed to isolate high levels of direct current from a device, while allowing a small alternating or alternating current signal to pass through.

Filtered out

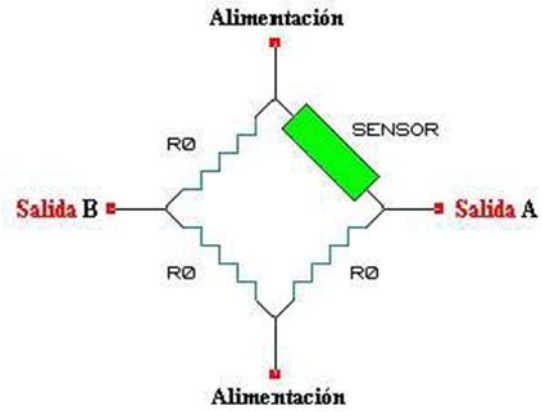
Another important function of the signal conditioner is filtering. It consists of filtering the frequency spectrum of the signal keeping only the valid data and blocking all noise. Filters can consist of passive and active components or a digital algorithm. A passive filter uses only capacitors, resistors and inductors with a maximum gain of one. An active filter uses passive components combined with active components, such as operational amplifiers and transistors. The most advanced signal conditioners use digital filters, because they are easy to adjust and do not require physical equipment. A digital filter is a mathematical filter that is used to manipulate a signal; for example, to block or miss a certain frequency range. They use logical components such as integrated circuits for specific applications (ASIC) or programmable door arrays (FPGA), or a sequential program with a signal processor.

Evaluation and intelligent functions

To provide additional benefits to the user and the process, modern signal conditioners have special functions for signal evaluation and preprocessing of measured data. Thus, they help to monitor and evaluate alarms and warnings quickly and directly, through a switched electrical output. Other additional intelligent functions, such as internal calculation channels, are responsible for performing mathematical operations, such as adding sensor signals, or technological operations, such as acting as a PID controller. These functions help the system react faster and reduce the workload of machine control.

Interfaces

Signal converters must transmit the sensor signals to the machine control, using standard interfaces and protocols. The interfaces can be analog or digital. Typical analog interfaces are voltage (+/- 10 V) or current (+/- 20 mA) signals, which are easy to handle but have the disadvantage that each signal requires independent wiring. Modern digital interfaces are designed as the Ethernet-based bus interfaces (Profinet, Ethercat, Ethernet / IP) and allow several components to relate to a single wire. In this way the wiring is simplified, and additional information can be transmitted; for example, diagnostic information of the components, which is very important to reduce downtimes and to accelerate maintenance.

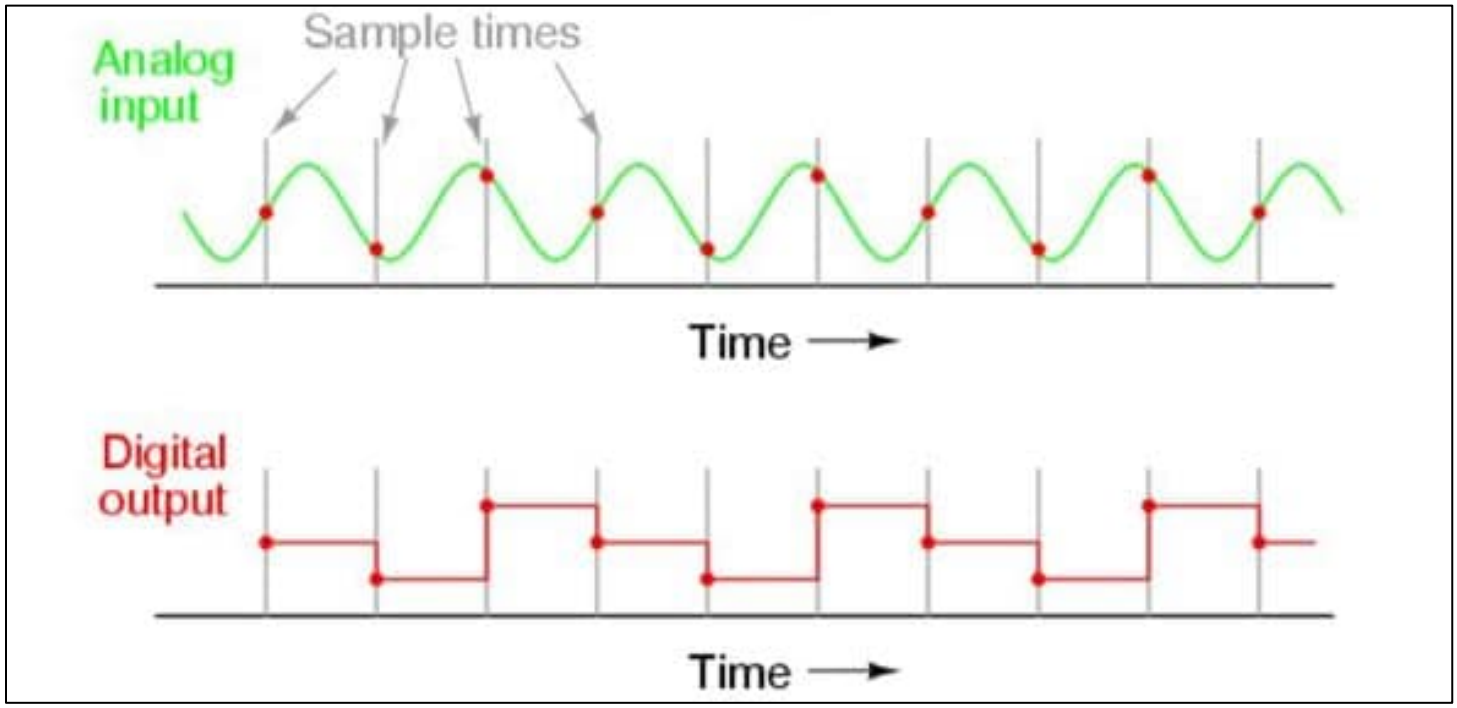


Digitalization

Analog-to-digital conversion consists of transcribing analog signals into digital signal, with the purpose of facilitating their processing (encoding, compression, etc.) and making the resulting (digital) signal more immune to noise and other interferences to which they are more sensitive analog signals.

Digitization or A / D conversion, basically, consists of periodically making measurements of the amplitude (voltage) of a signal; for example, the one that comes from a microphone if it is (hold) by a hold circuit, long enough to allow to evaluate its level (quantification). From a mathematical point of view, this process is not contemplated because it is a technical resource due to practical limitations, and it lacks a mathematical model.

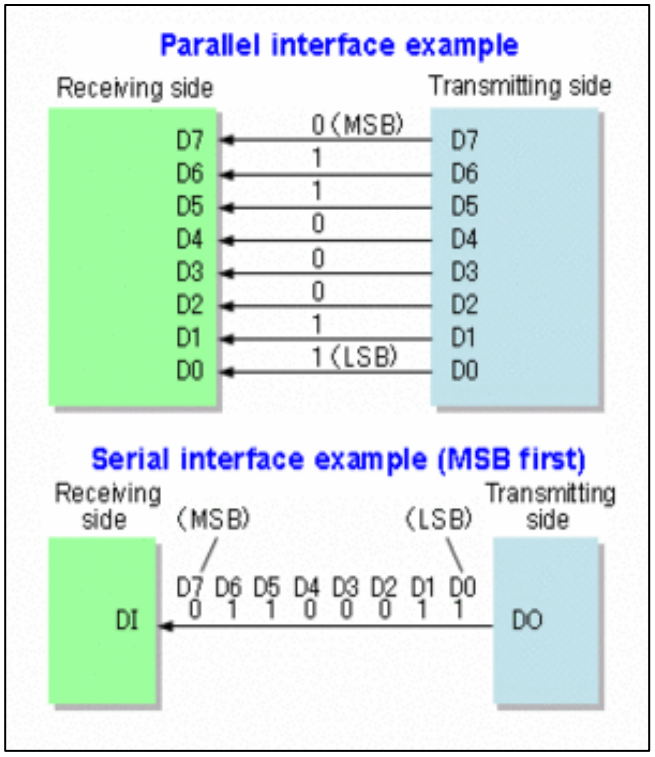
During "sampling" and "hold", the signal is still analog, since it can still take any value. However, from the "quantization", when the signal already takes finite values, the signal is already digital. All four processes take place in an analog-to-digital converter.



Serial Communication

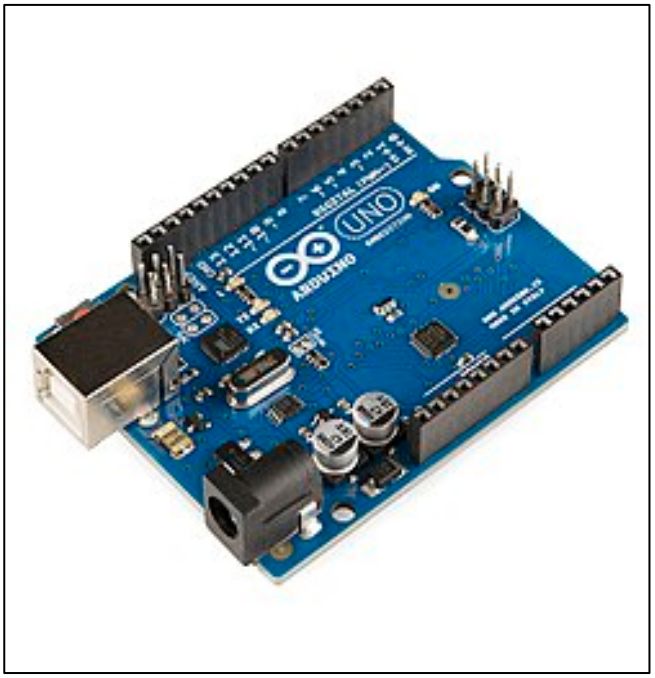
In telecommunication and data transmission, serial communication is the process of sending data one bit at a time, sequentially, over a communication channel or computer bus. This is in contrast to parallel communication, where several bits are sent as a whole, on a link with several parallel channels.

Serial communication is used for all long-haul communication and most computer networks, where the cost of cable and synchronization difficulties make parallel communication impractical. Serial computer buses are becoming more common even at shorter distances, as improved signal integrity and transmission speeds in newer serial technologies have begun to outweigh the parallel bus's advantage of simplicity (no need for serializer and deserializer, or SerDes) and to outstrip its disadvantages (clock skew, interconnect density). The migration from PCI to PCI Express is an example.



Microcontroller: Arduino Uno

The board has 14 digital pins, 6 analog pins programmable with the Arduino IDE (Integrated Development Environment) via a USB cable. It can be powered by the USB cable or by an external 9 volt battery, although it accepts voltages between 7 and 20 volts. It is the flagship of Arduino since it is the most popular board, the one that everyone uses to get started and the easiest to use. It is the starting point for many electronic programming enthusiasts.



Arduino is a free hardware electronic board that uses a reprogrammable microcontroller with a series of pins that allow establishing connections between the controller and the different sensors, that is, the "brain" of some circuit or machinery.

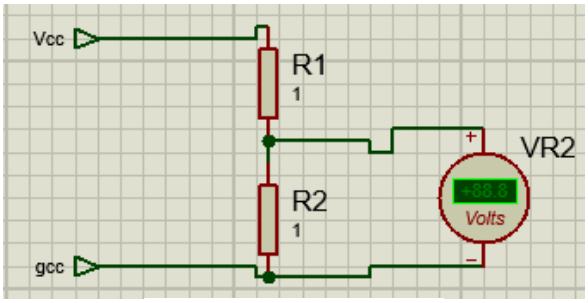
In a circuit it is usually used as a power source and a "bridge" between the different components to make them interact with each other.

In a robotic body it ceases to serve as a power source and becomes entirely the brain of the body with the help of external plates known as shields.

Development

1. Settings sensor (CAS)

As a first step, we perform the resistance measurements to obtain an output range of 0 to 5 volts, with an input range of 0 to 60 volts.



$$V_{R2} = \frac{R2\ x\ V_{ent}}{R2 + R1}$$

To calculate the resistances, start using the voltage divider with the maximum input value, that is 60 volts:

$$5V = \frac{R2\ x\ 60}{R2 + R1}$$

$$5V\ (R2 + R1) = R2\ x\ 60$$

$$5R2 + 5R1 = 60R2$$

$$5R1 = 55R2$$

$$R1 = 11R2$$

If R2 = 10KΩ

$$R1 = 110{,}000\ \Omega$$

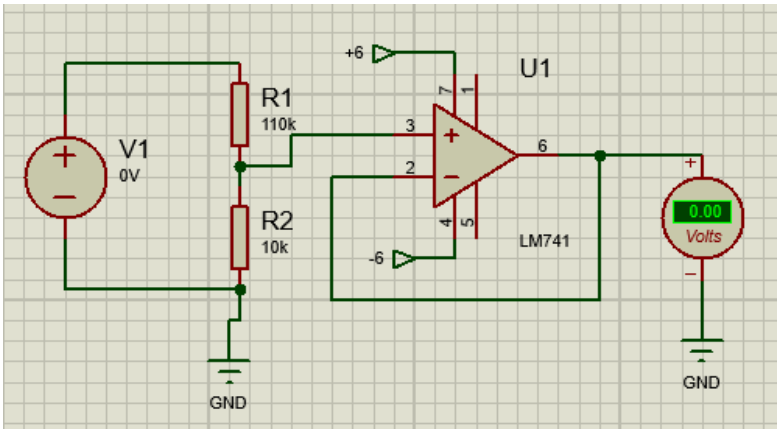
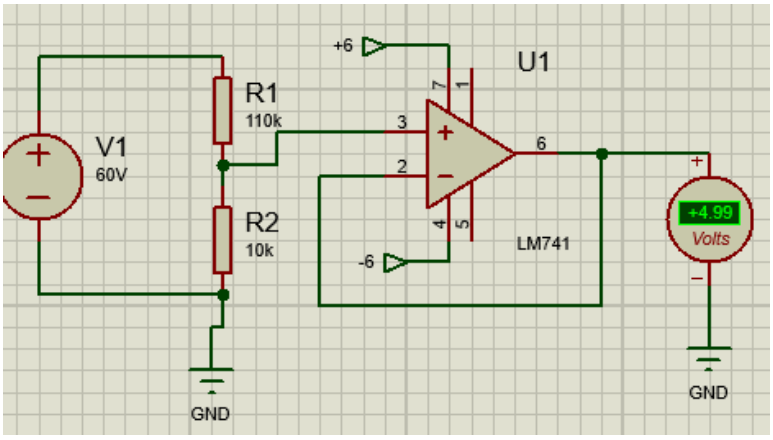
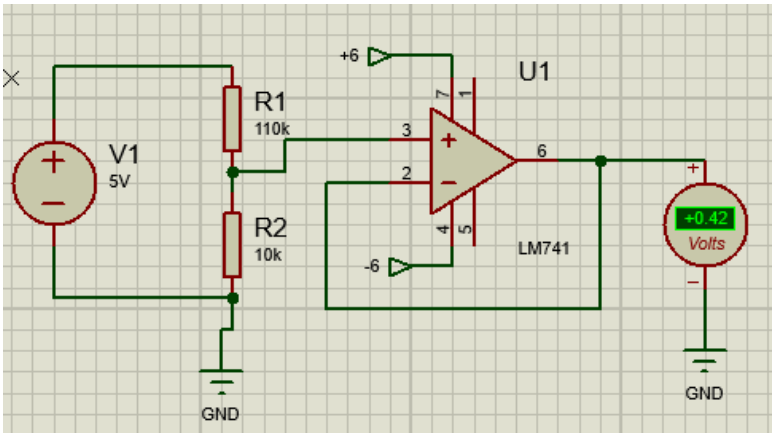
So, the current intensity is as:

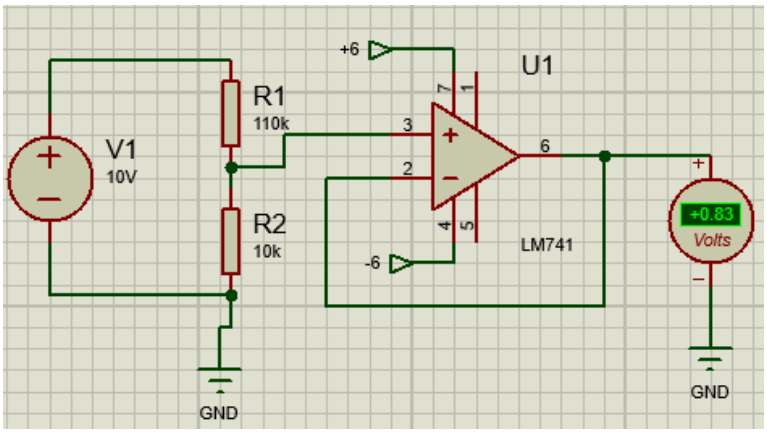
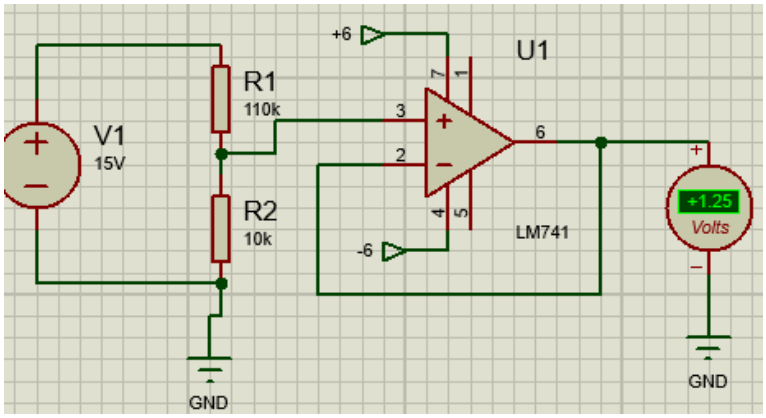
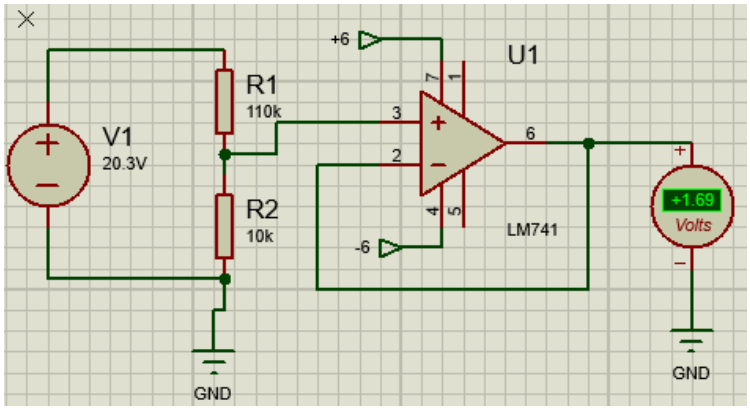
$$I = \frac{V}{R} = \frac{60\ V}{120{,}000\ \Omega} = 500\ \mu A$$

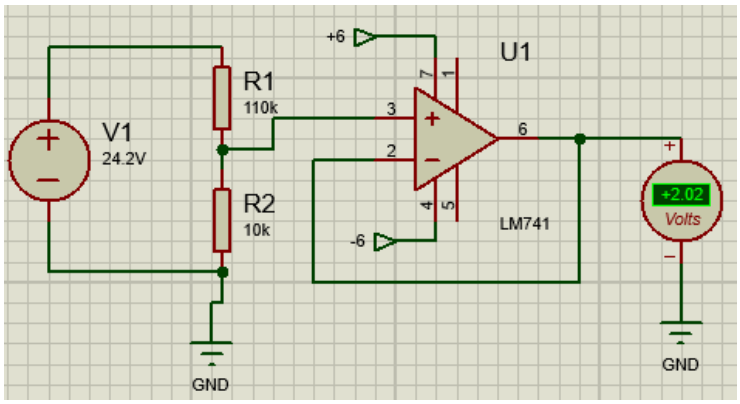
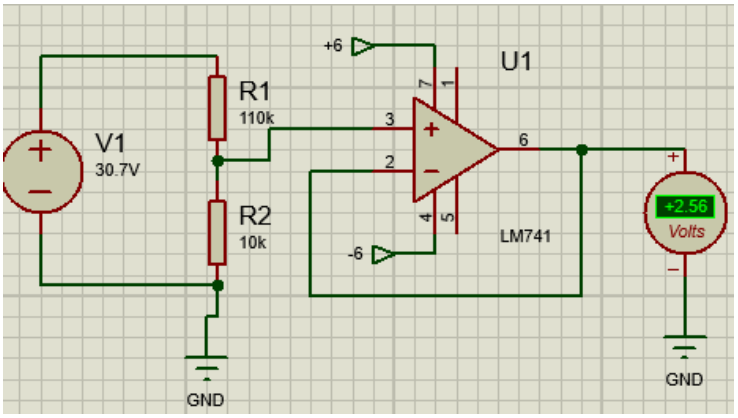
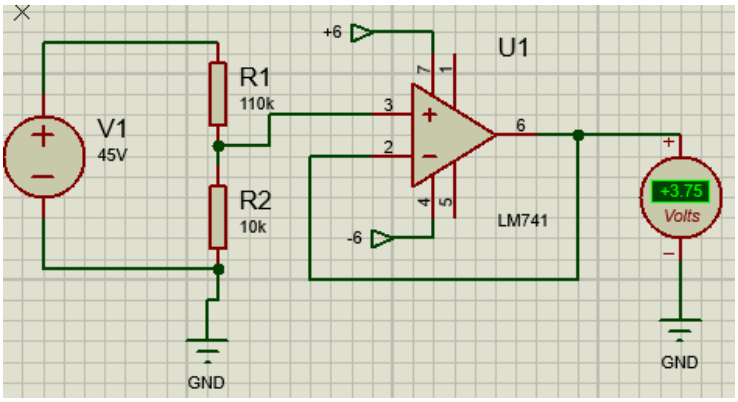
$$P_{max} = VI = 60v\ X\ 500\ \mu A = 30mW$$

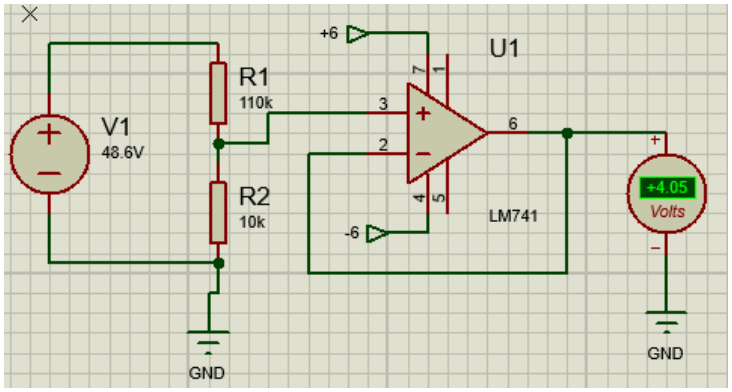
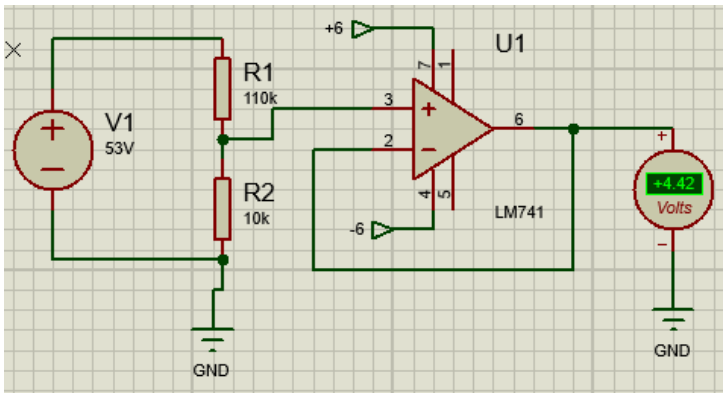
It has a current less than 10 mA so the resistance assignment is correct.

Next, We compare measured results with calculated results, already with the connected voltage follower, in order to increase the impedance and prevent the entry of noise.

Vent [V]	Vsal measured [V]	Vsal calculated [V]	Vsal simulated [V]
0	1.50m	$Vsal = \frac{10k\Omega(0V)}{10k\Omega + 110k\Omega} = 0$	
60	5.06	$Vsal = \frac{10k\Omega(60V)}{10k\Omega + 110k\Omega} = 5$	
5	418.24m	$Vsal = \frac{10k\Omega(5V)}{10k\Omega + 110k\Omega} = 416.66m$	

10	846.7m	$Vsal = \frac{10k\Omega(10V)}{10k\Omega + 110k\Omega} = 833.33m$	
15	1.2632	$Vsal = \frac{10k\Omega(15V)}{10k\Omega + 110k\Omega} = 1.25$	
20.3	1.706	$Vsal = \frac{10k\Omega(20.3V)}{10k\Omega + 110k\Omega} = 1.691$	

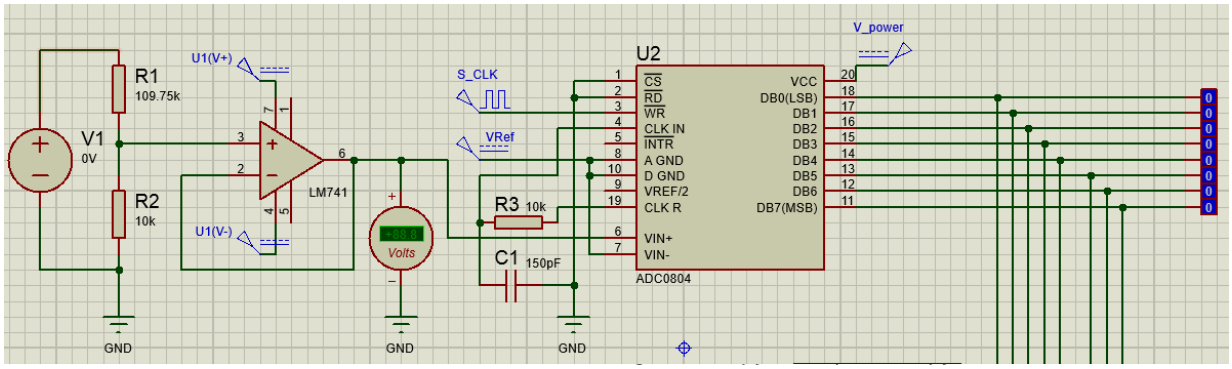
24.2	2.035	$Vsal = \frac{10k\Omega(24.2V)}{10k\Omega + 110k\Omega} = 2.016$	
30.7	2.589	$Vsal = \frac{10k\Omega(30.7V)}{10k\Omega + 110k\Omega} = 2.558$	
45	3.793	$Vsal = \frac{10k\Omega(45V)}{10k\Omega + 110k\Omega} = 3.75$	

48.6	4.098	$Vsal = \frac{10k\Omega(48.6V)}{10k\Omega + 110k\Omega} = 4.05$	
53	4.476	$Vsal = \frac{10k\Omega(53V)}{10k\Omega + 110k\Omega} = 4.416$	

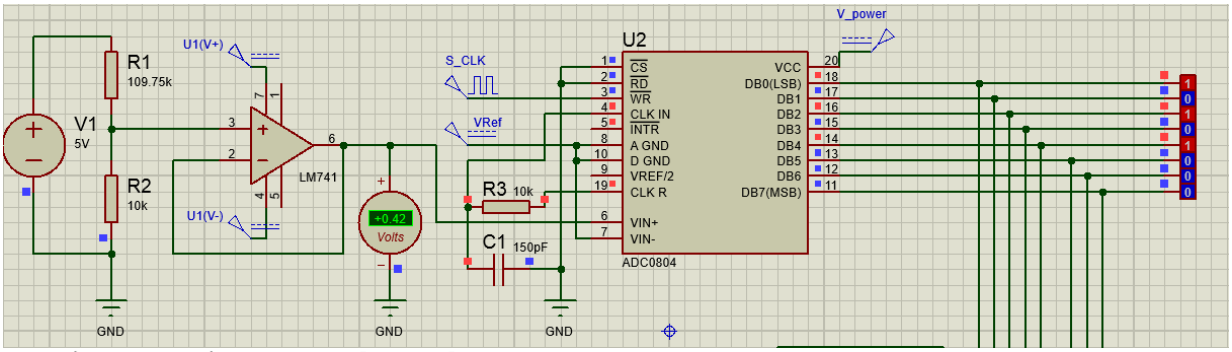
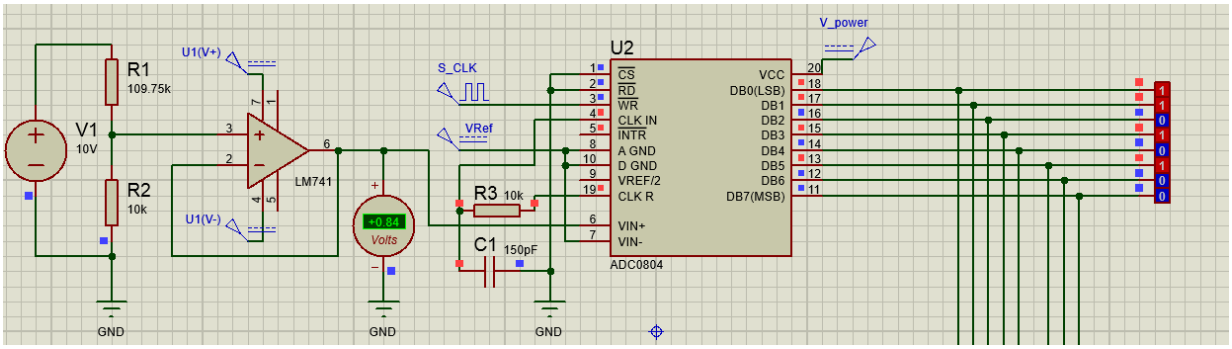
2. Digitization stage

But, how can we digitize to voltage value? This is a good question; we are going to use a successive approximation algorithm for the previous values:

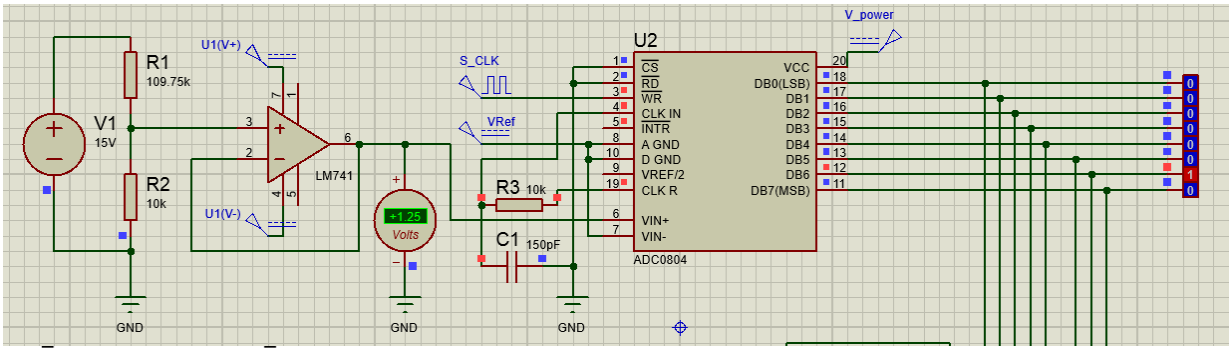
```
1. datos = [0,5,10,15,20.3,24.2,30.7,45,48.6,53,60]
2.
3. for dato in datos:
4.     inferior = 0
5.     superior = 60
6.     binario = ""
7.     for i in range(0,8):
8.         mitad = (inferior + superior)/2
9.         if dato >= mitad:
10.                print("Range(",inferior," - ",superior,") where middle equals to
   ", mitad, ",", dato, " >= ", mitad, "? SI, so B",7-i," = 1" )
11.                inferior = mitad
12.                binario = binario + "1"
13.            else:
14.                print("Range(",inferior," - ",superior,") where middle equals to
   ", mitad, ",", dato, " >= ", mitad, "? NO, so B",7-i," = 0" )
15.                superior = mitad
16.                binario = binario + "0"
17.
18.         print(dato,"_(10) = ",binario, "_(2)")
```

Voltage (V)	Digitization simulation
0	

```
Range( 0  -  60 ) where middle equals to  30.0 , 0  >=  30.0 ? NO, so B 7  = 0
Range( 0  -  30.0 ) where middle equals to  15.0 , 0  >=  15.0 ? NO, so B 6  = 0
Range( 0  -  15.0 ) where middle equals to  7.5 , 0  >=  7.5 ? NO, so B 5  = 0
Range( 0  -  7.5 ) where middle equals to  3.75 , 0  >=  3.75 ? NO, so B 4  = 0
Range( 0  -  3.75 ) where middle equals to  1.875 , 0  >=  1.875 ? NO, so B 3  = 0
Range( 0  -  1.875 ) where middle equals to  0.9375 , 0  >=  0.9375 ? NO, so B 2  = 0
Range( 0  -  0.9375 ) where middle equals to  0.46875 , 0  >=  0.46875 ? NO, so B 1  = 0
Range( 0  -  0.46875 ) where middle equals to  0.234375 , 0  >=  0.234375 ? NO, so B 0  = 0
0 _(10) =  00000000 _(2)
```

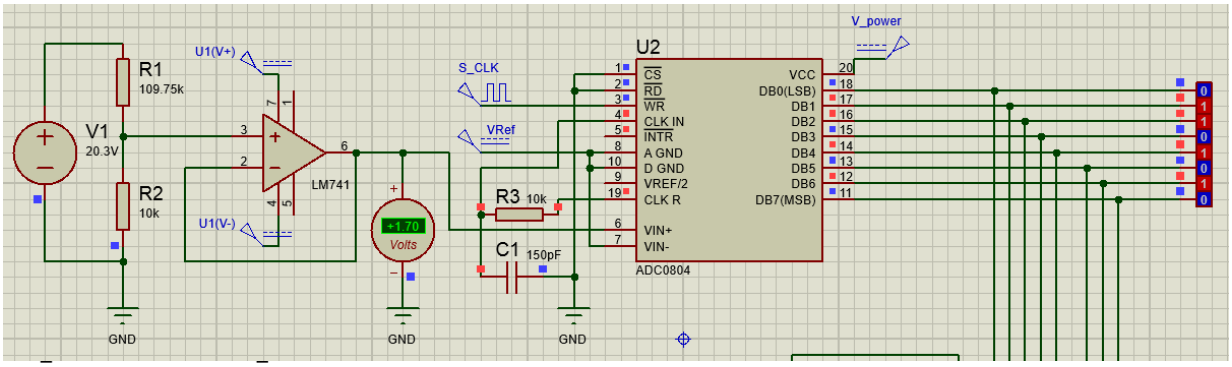
5	 Range(0 - 60) where middle equals to 30.0 , 5 >= 30.0 ? NO, so B 7 = 0 Range(0 - 30.0) where middle equals to 15.0 , 5 >= 15.0 ? NO, so B 6 = 0 Range(0 - 15.0) where middle equals to 7.5 , 5 >= 7.5 ? NO, so B 5 = 0 Range(0 - 7.5) where middle equals to 3.75 , 5 >= 3.75 ? SI, so B 4 = 1 Range(3.75 - 7.5) where middle equals to 5.625 , 5 >= 5.625 ? NO, so B 3 = 0 Range(3.75 - 5.625) where middle equals to 4.6875 , 5 >= 4.6875 ? SI, so B 2 = 1 Range(4.6875 - 5.625) where middle equals to 5.15625 , 5 >= 5.15625 ? NO, so B 1 = 0 Range(4.6875 - 5.15625) where middle equals to 4.921875 , 5 >= 4.921875 ? SI, so B 0 = 1 5 _(10) = 00010101 _(2)
10	 Range(0 - 60) where middle equals to 30.0 , 10 >= 30.0 ? NO, so B 7 = 0 Range(0 - 30.0) where middle equals to 15.0 , 10 >= 15.0 ? NO, so B 6 = 0 Range(0 - 15.0) where middle equals to 7.5 , 10 >= 7.5 ? SI, so B 5 = 1 Range(7.5 - 15.0) where middle equals to 11.25 , 10 >= 11.25 ? NO, so B 4 = 0 Range(7.5 - 11.25) where middle equals to 9.375 , 10 >= 9.375 ? SI, so B 3 = 1 Range(9.375 - 11.25) where middle equals to 10.3125 , 10 >= 10.3125 ? NO, so B 2 = 0 Range(9.375 - 10.3125) where middle equals to 9.84375 , 10 >= 9.84375 ? SI, so B 1 = 1 Range(9.84375 - 10.3125) where middle equals to 10.078125 , 10 >= 10.078125 ? NO, so B 0 = 0 10 _(10) = 00101010 _(2)

15



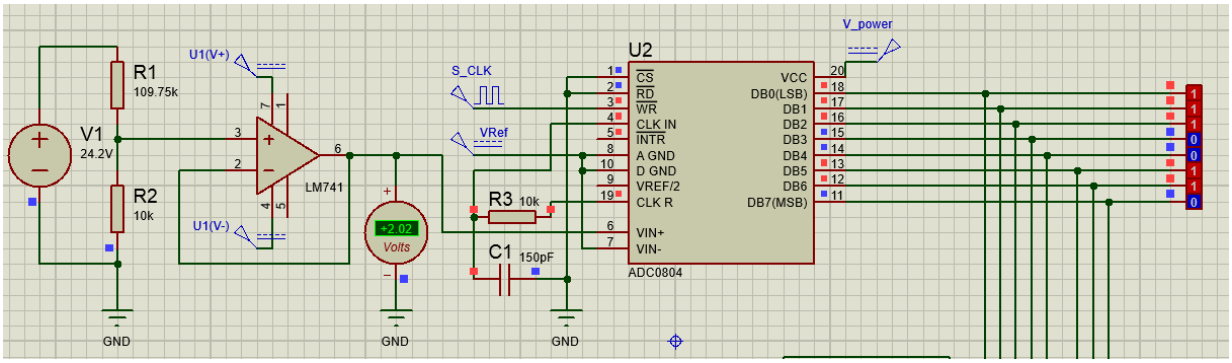

```
Range( 0  -  60 ) where middle equals to  30.0 , 15  >=  30.0 ? NO, so B 7  = 0
Range( 0  -  30.0 ) where middle equals to  15.0 , 15  >=  15.0 ? SI, so B 6  = 1
Range( 15.0  -  30.0 ) where middle equals to  22.5 , 15  >=  22.5 ? NO, so B 5  = 0
Range( 15.0  -  22.5 ) where middle equals to  18.75 , 15  >=  18.75 ? NO, so B 4  = 0
Range( 15.0  -  18.75 ) where middle equals to  16.875 , 15  >=  16.875 ? NO, so B 3  = 0
Range( 15.0  -  16.875 ) where middle equals to  15.9375 , 15  >=  15.9375 ? NO, so B 2  = 0
Range( 15.0  -  15.9375 ) where middle equals to  15.46875 , 15  >=  15.46875 ? NO, so B 1  = 0
Range( 15.0  -  15.46875 ) where middle equals to  15.234375 , 15  >=  15.234375 ? NO, so B 0  = 0
15 _(10) =  01000000 _(2)
```

20.3



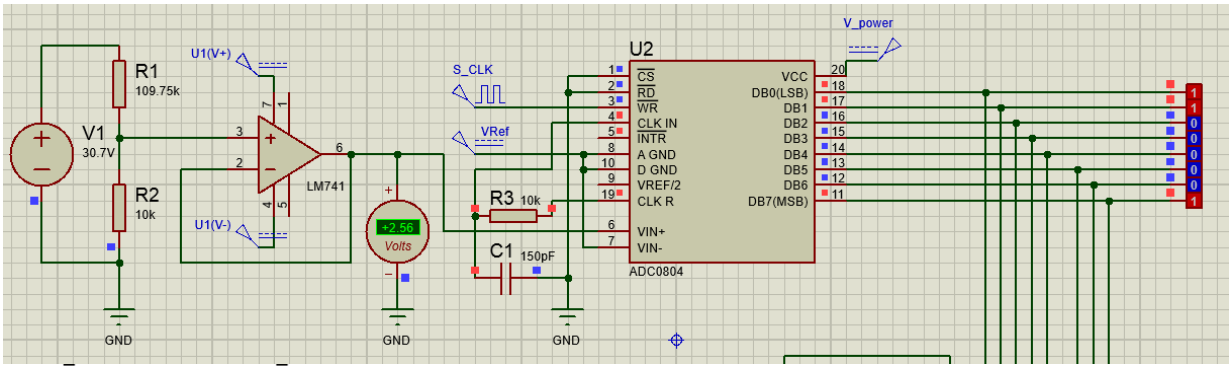

```
Range( 0  -  60 ) where middle equals to  30.0 , 20.3  >=  30.0 ? NO, so B 7  = 0
Range( 0  -  30.0 ) where middle equals to  15.0 , 20.3  >=  15.0 ? SI, so B 6  = 1
Range( 15.0  -  30.0 ) where middle equals to  22.5 , 20.3  >=  22.5 ? NO, so B 5  = 0
Range( 15.0  -  22.5 ) where middle equals to  18.75 , 20.3  >=  18.75 ? SI, so B 4  = 1
Range( 18.75  -  22.5 ) where middle equals to  20.625 , 20.3  >=  20.625 ? NO, so B 3  = 0
Range( 18.75  -  20.625 ) where middle equals to  19.6875 , 20.3  >=  19.6875 ? SI, so B 2  = 1
Range( 19.6875  -  20.625 ) where middle equals to  20.15625 , 20.3  >=  20.15625 ? SI, so B 1  = 1
Range( 20.15625  -  20.625 ) where middle equals to  20.390625 , 20.3  >=  20.390625 ? NO, so B 0  = 0
20.3 _(10) =  01010110 _(2)
```

24.2



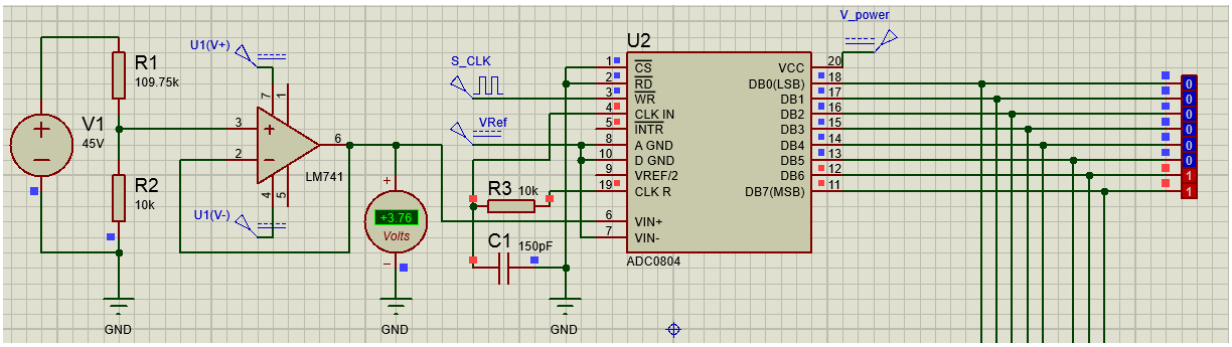

```
Range( 0  -  60 ) where middle equals to  30.0 , 24.2  >=  30.0 ? NO, so B 7  = 0
Range( 0  -  30.0 ) where middle equals to  15.0 , 24.2  >=  15.0 ? SI, so B 6  = 1
Range( 15.0  -  30.0 ) where middle equals to  22.5 , 24.2  >=  22.5 ? SI, so B 5  = 1
Range( 22.5  -  30.0 ) where middle equals to  26.25 , 24.2  >=  26.25 ? NO, so B 4  = 0
Range( 22.5  -  26.25 ) where middle equals to  24.375 , 24.2  >=  24.375 ? NO, so B 3  = 0
Range( 22.5  -  24.375 ) where middle equals to  23.4375 , 24.2  >=  23.4375 ? SI, so B 2  = 1
Range( 23.4375  -  24.375 ) where middle equals to  23.90625 , 24.2  >=  23.90625 ? SI, so B 1  = 1
Range( 23.90625  -  24.375 ) where middle equals to  24.140625 , 24.2  >=  24.140625 ? SI, so B 0  = 1
24.2 _(10) =  01100111 _(2)
```

30.7



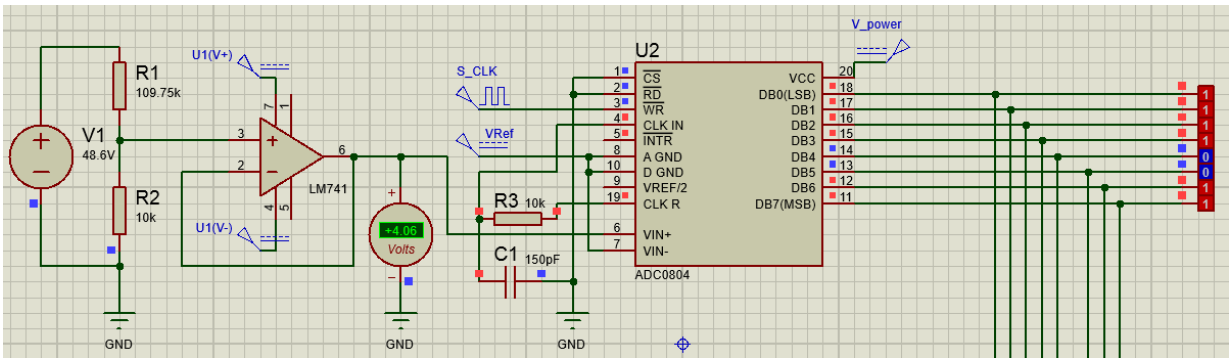

```
Range( 0  -  60 ) where middle equals to  30.0 , 30.7  >=  30.0 ? SI, so B 7  = 1
Range( 30.0  -  60 ) where middle equals to  45.0 , 30.7  >=  45.0 ? NO, so B 6  = 0
Range( 30.0  -  45.0 ) where middle equals to  37.5 , 30.7  >=  37.5 ? NO, so B 5  = 0
Range( 30.0  -  37.5 ) where middle equals to  33.75 , 30.7  >=  33.75 ? NO, so B 4  = 0
Range( 30.0  -  33.75 ) where middle equals to  31.875 , 30.7  >=  31.875 ? NO, so B 3  = 0
Range( 30.0  -  31.875 ) where middle equals to  30.9375 , 30.7  >=  30.9375 ? NO, so B 2  = 0
Range( 30.0  -  30.9375 ) where middle equals to  30.46875 , 30.7  >=  30.46875 ? SI, so B 1  = 1
Range( 30.46875  -  30.9375 ) where middle equals to  30.703125 , 30.7  >=  30.703125 ? NO, so B 0  = 0
30.7 _(10) =  10000010 _(2)
```

45



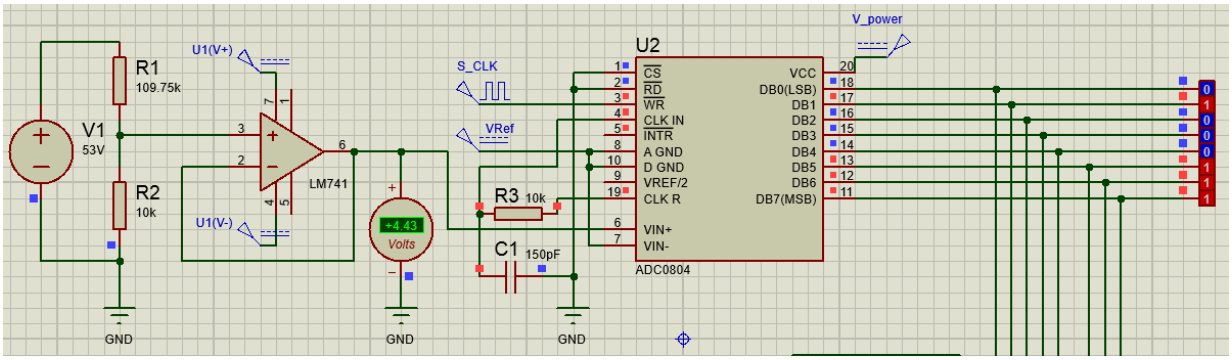

```
Range( 0  -  60 ) where middle equals to  30.0 , 45  >=  30.0 ? SI, so B 7  = 1
Range( 30.0  -  60 ) where middle equals to  45.0 , 45  >=  45.0 ? SI, so B 6  = 1
Range( 45.0  -  60 ) where middle equals to  52.5 , 45  >=  52.5 ? NO, so B 5  = 0
Range( 45.0  -  52.5 ) where middle equals to  48.75 , 45  >=  48.75 ? NO, so B 4  = 0
Range( 45.0  -  48.75 ) where middle equals to  46.875 , 45  >=  46.875 ? NO, so B 3  = 0
Range( 45.0  -  46.875 ) where middle equals to  45.9375 , 45  >=  45.9375 ? NO, so B 2  = 0
Range( 45.0  -  45.9375 ) where middle equals to  45.46875 , 45  >=  45.46875 ? NO, so B 1  = 0
Range( 45.0  -  45.46875 ) where middle equals to  45.234375 , 45  >=  45.234375 ? NO, so B 0  = 0
45 _(10) =  11000000 _(2)
```

48.6



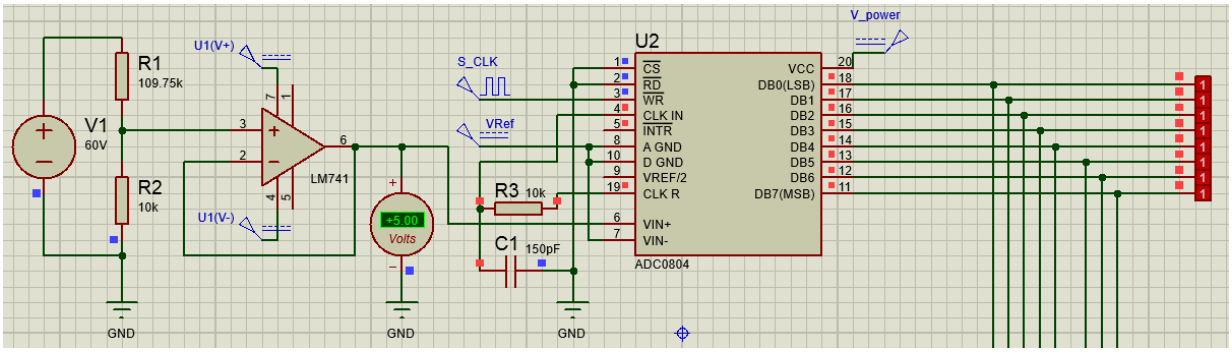

```
Range( 0  -  60 ) where middle equals to  30.0 , 48.6  >=  30.0 ? SI, so B 7  = 1
Range( 30.0  -  60 ) where middle equals to  45.0 , 48.6  >=  45.0 ? SI, so B 6  = 1
Range( 45.0  -  60 ) where middle equals to  52.5 , 48.6  >=  52.5 ? NO, so B 5  = 0
Range( 45.0  -  52.5 ) where middle equals to  48.75 , 48.6  >=  48.75 ? NO, so B 4  = 0
Range( 45.0  -  48.75 ) where middle equals to  46.875 , 48.6  >=  46.875 ? SI, so B 3  = 1
Range( 46.875  -  48.75 ) where middle equals to  47.8125 , 48.6  >=  47.8125 ? SI, so B 2  = 1
Range( 47.8125  -  48.75 ) where middle equals to  48.28125 , 48.6  >=  48.28125 ? SI, so B 1  = 1
Range( 48.28125  -  48.75 ) where middle equals to  48.515625 , 48.6  >=  48.515625 ? SI, so B 0  = 1
48.6 _(10) =  11001111 _(2)
```

53




```
Range( 0  -  60 ) where middle equals to  30.0 , 53  >=  30.0 ? SI, so B 7  = 1
Range( 30.0  -  60 ) where middle equals to  45.0 , 53  >=  45.0 ? SI, so B 6  = 1
Range( 45.0  -  60 ) where middle equals to  52.5 , 53  >=  52.5 ? SI, so B 5  = 1
Range( 52.5  -  60 ) where middle equals to  56.25 , 53  >=  56.25 ? NO, so B 4  = 0
Range( 52.5  -  56.25 ) where middle equals to  54.375 , 53  >=  54.375 ? NO, so B 3  = 0
Range( 52.5  -  54.375 ) where middle equals to  53.4375 , 53  >=  53.4375 ? NO, so B 2  = 0
Range( 52.5  -  53.4375 ) where middle equals to  52.96875 , 53  >=  52.96875 ? SI, so B 1  = 1
Range( 52.96875  -  53.4375 ) where middle equals to  53.203125 , 53  >=  53.203125 ? NO, so B 0  = 0
53 _(10) =  11100010 _(2)
```

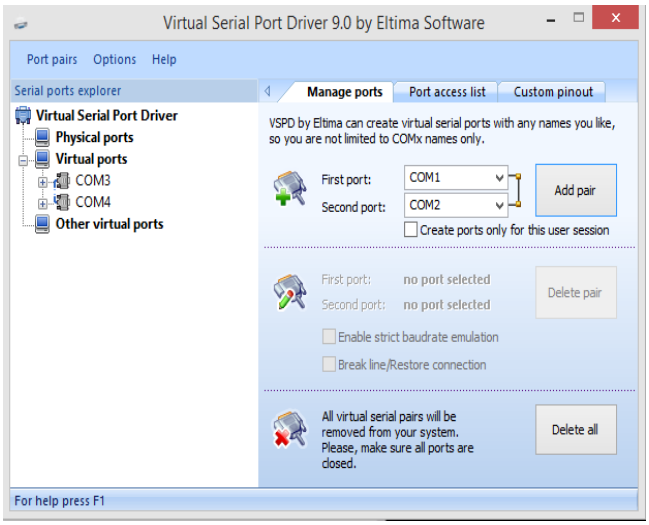
60

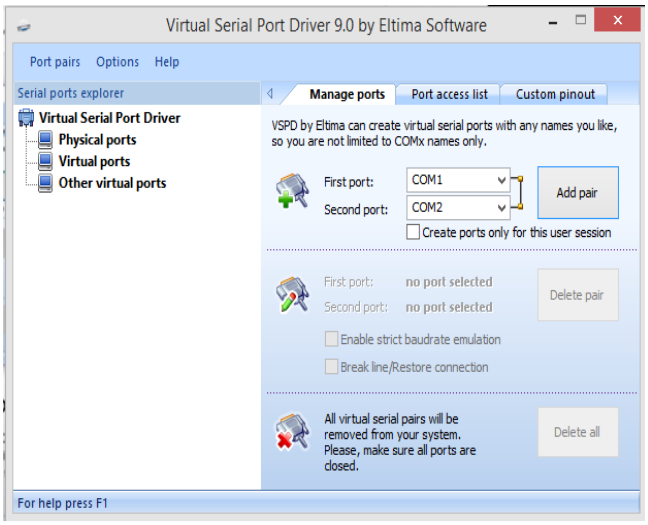



```
Range( 0  -  60 ) where middle equals to  30.0 , 60  >=  30.0 ? SI, so B 7  = 1
Range( 30.0  -  60 ) where middle equals to  45.0 , 60  >=  45.0 ? SI, so B 6  = 1
Range( 45.0  -  60 ) where middle equals to  52.5 , 60  >=  52.5 ? SI, so B 5  = 1
Range( 52.5  -  60 ) where middle equals to  56.25 , 60  >=  56.25 ? SI, so B 4  = 1
Range( 56.25  -  60 ) where middle equals to  58.125 , 60  >=  58.125 ? SI, so B 3  = 1
Range( 58.125  -  60 ) where middle equals to  59.0625 , 60  >=  59.0625 ? SI, so B 2  = 1
Range( 59.0625  -  60 ) where middle equals to  59.53125 , 60  >=  59.53125 ? SI, so B 1  = 1
Range( 59.53125  -  60 ) where middle equals to  59.765625 , 60  >=  59.765625 ? SI, so B 0  = 1
60 _(10) =  11111111 _(2)
```

3. Serial connection and catching data on the computer

Well, since everything is simulated, we had to use the program “Virtual Serial Port Driver” for simulate real device ports, then there is an image with the settings:





The virtual connection is done, now we have to send the digital values that we got with the ADC to a microcontroller (in this case we are going to use “Arduino”) and this receives in its digital inputs the ADC outputs, then Arduino sends the decimal value through the serial connection, in the computer we are going to catch the value through the specific port by a python program:

Arduino source code

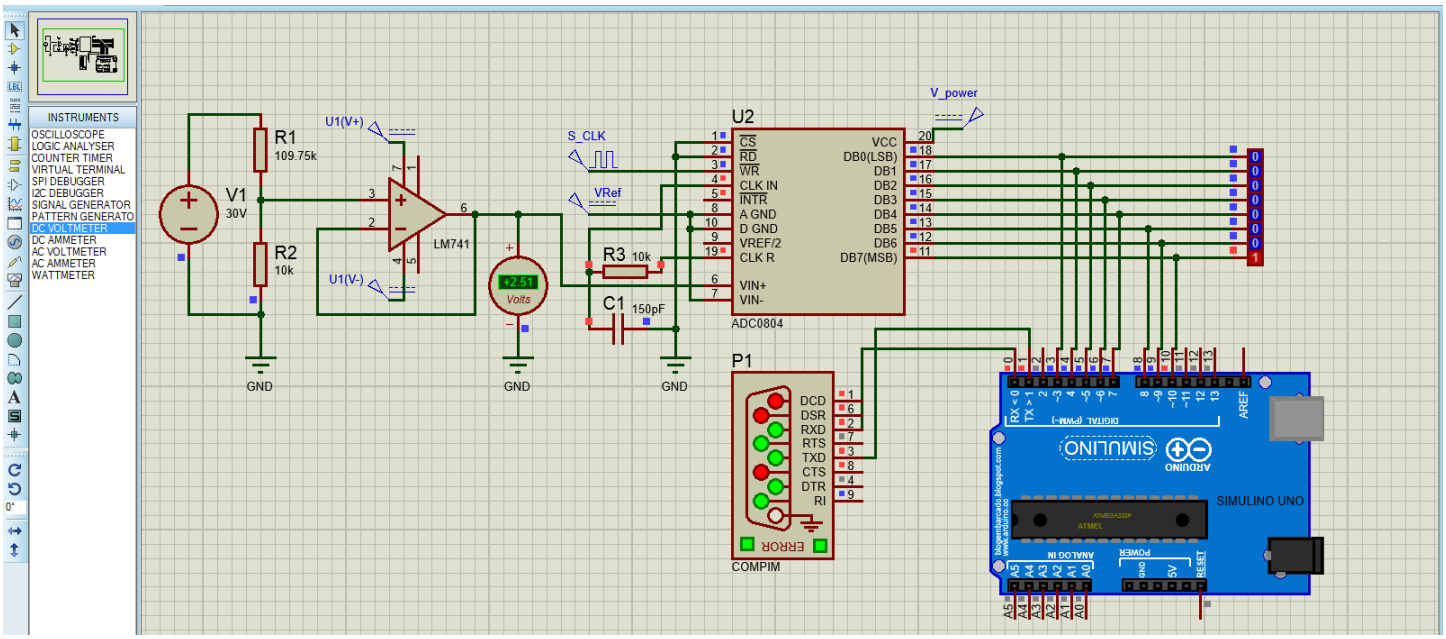
```
void setup() {
  Serial.begin(9600);
}

void loop() {
  boolean dig0 = digitalRead(3);
  boolean dig1 = digitalRead(4);
  boolean dig2 = digitalRead(5);
  boolean dig3 = digitalRead(6);
  boolean dig4 = digitalRead(7);
  boolean dig5 = digitalRead(8);
  boolean dig6 = digitalRead(9);
  boolean dig7 = digitalRead(10);

  int dec = dig0*pow(2,0) + dig1*pow(2,1) + dig2*pow(2,2) + dig3*pow(2,3)+ dig4*pow(2,4) +
dig5*pow(2,5) + dig6*pow(2,6) + dig7*pow(2,7);

  Serial.println(dec);
  delay(300);

}
```

Python source code

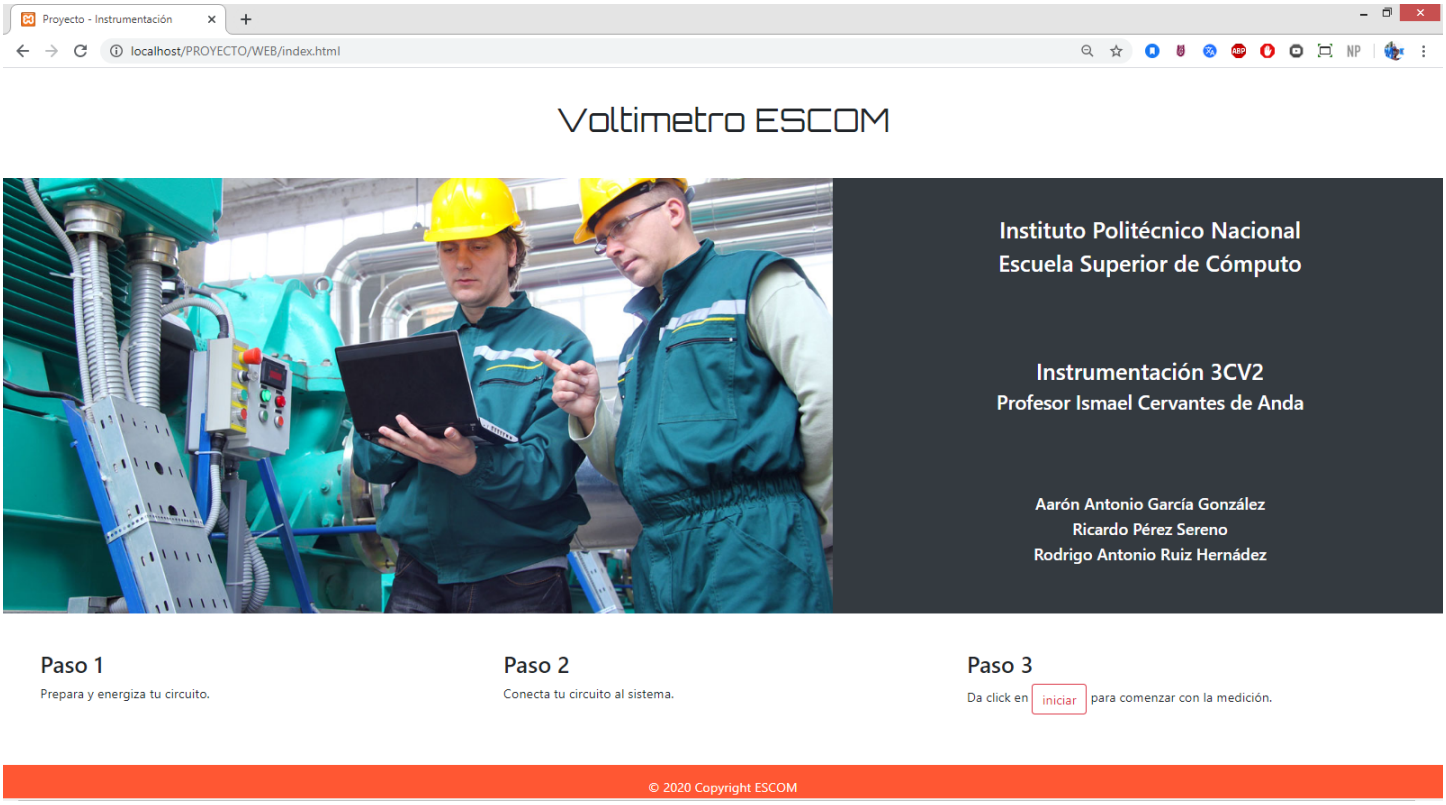
```
import serial
import time

import json
import os

arduino = serial.Serial('COM4',9600)

data = {}
DIR = "C:\\xampp\htdocs\PROYECTO\WEB"
FILE_NAME = "index.json"

data['historico'] = [0]

anterior = -1

while 1:
    dato_lectura = int(arduino.readline())
    voltaje = 60/255*dato_lectura
    data['voltaje'] = voltaje

    if data['historico'][-1] != voltaje:
        data['historico'].append(voltaje)

    with open(os.path.join(DIR, FILE_NAME), 'w') as file:
        json.dump(data,file)

    if anterior != voltaje:
        anterior = voltaje
        print(voltaje," volts")

arduino.close()
```

4. Data visualization

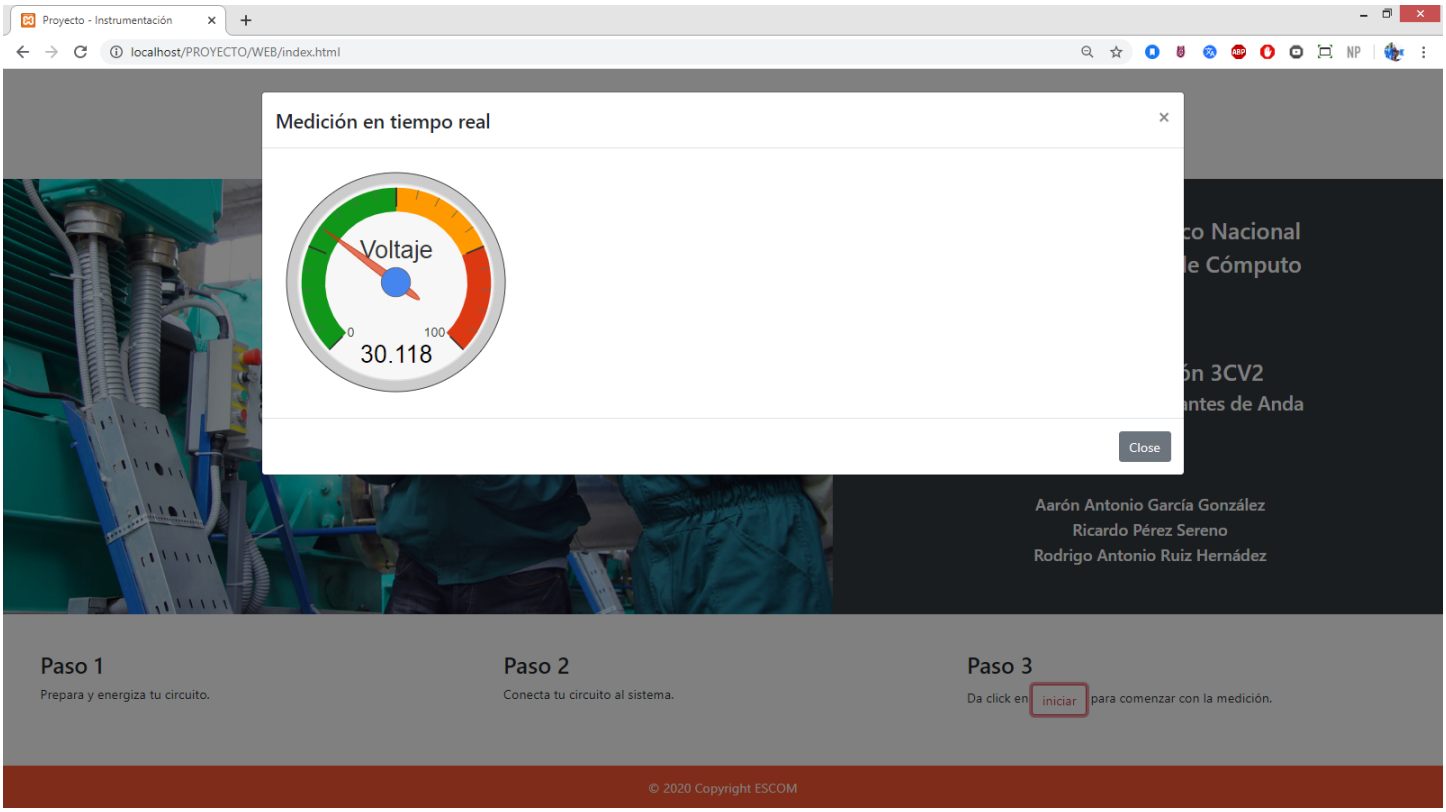
But how is the system at the end? The connection between CAD, ADC and Arduino is the next:



In the data visualization we can show a web page like this:



How the website works? Its very simple, first our python program is running all the time and each time that the system measures a different value (compared to the previous) write this value in a json file, this json will be read by a JavaScript program and put the read value in the gauge value to graph.
For example, in the next picture we can show the final result when we are measuring 30 volts, this result is not exact, but its depends of the bit number that we used in the digitization stage (8 bits), so, the middle of 255 binary combinations is 127.5, and when the voltage value is the middle, only the most significant bit will be on, this value is 128, 127.5 is not equal to 128, so that is the reason why our value in the web page is not exactly the same that we put in proteus program.



Like another fact, in the json file that we mentioned also we save the historical facts according to there is a change (compared to the previous):


```
{"historico": [0, 40.0, 0.0, 30.11764705882353], "voltaje": 30.11764705882353}
```

Conclusions

We can say that the initial part of the voltmeter was fulfilled, we only had a little difficulty with the amplifier part, but for the most part the first stages of this project were simple, at least the initial stage, which we could verify with the tests. laboratory, obviously that go hand in hand with the calculations and simulations that were made, tried to make it as accurate as possible and the difference shown is minimal.
Then we move on to the digitization stage which, because of the 8-bit precision, the results will not be entirely accurate.
It was good to have at least for us in the team an interaction of the analog electronics with the programming and with the computers, that in other previous courses we had not done, it generates a great interest and curiosity for the microcontrollers and the many applications and systems that we could carry out for later work, for example terminal work.

Bibliography

- Anónimo. (2019). ¿Qué es un acondicionador de señal? 02/03/2020, de HBK company Sitio web: https://www.hbm.com/es/7339/que-es-un-acondicionador-de-senal-funciones/?fbclid=IwAR0OiSOJ-nyE2xqIfL7o07ncY6bz1ncDt3pfRz2Pve36kdjl3Jm8K_yJNFM

- Using Node.JS, how do I read a JSON file into (server) memory? (2012, 4 abril). Recuperado 8 de junio de 2020, de https://stackoverflow.com/questions/10011011/using-node-js-how-do-i-read-a-json-file-into-server-memory