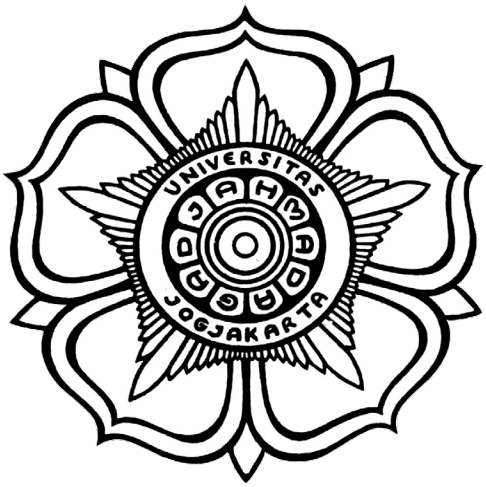
LAPORAN TUGAS KELOMPOK

PRAKTIK TEKNIK KENDALI DAN MESIN LISTRIK

"*PR1 ESP32 ADC to Python Charting***"**



Disusun Oleh:

Kelompok 3/ARM 2

Lukman Aulia Rahman (19/441201/SV/16553)

Aditya Bayu Maulana (19/447106/SV/16825)

DEPARTEMEN TEKNIK MESIN

SEKOLAH VOKASI

UNIVERSITAS GADJAH MADA

YOGYAKARTA

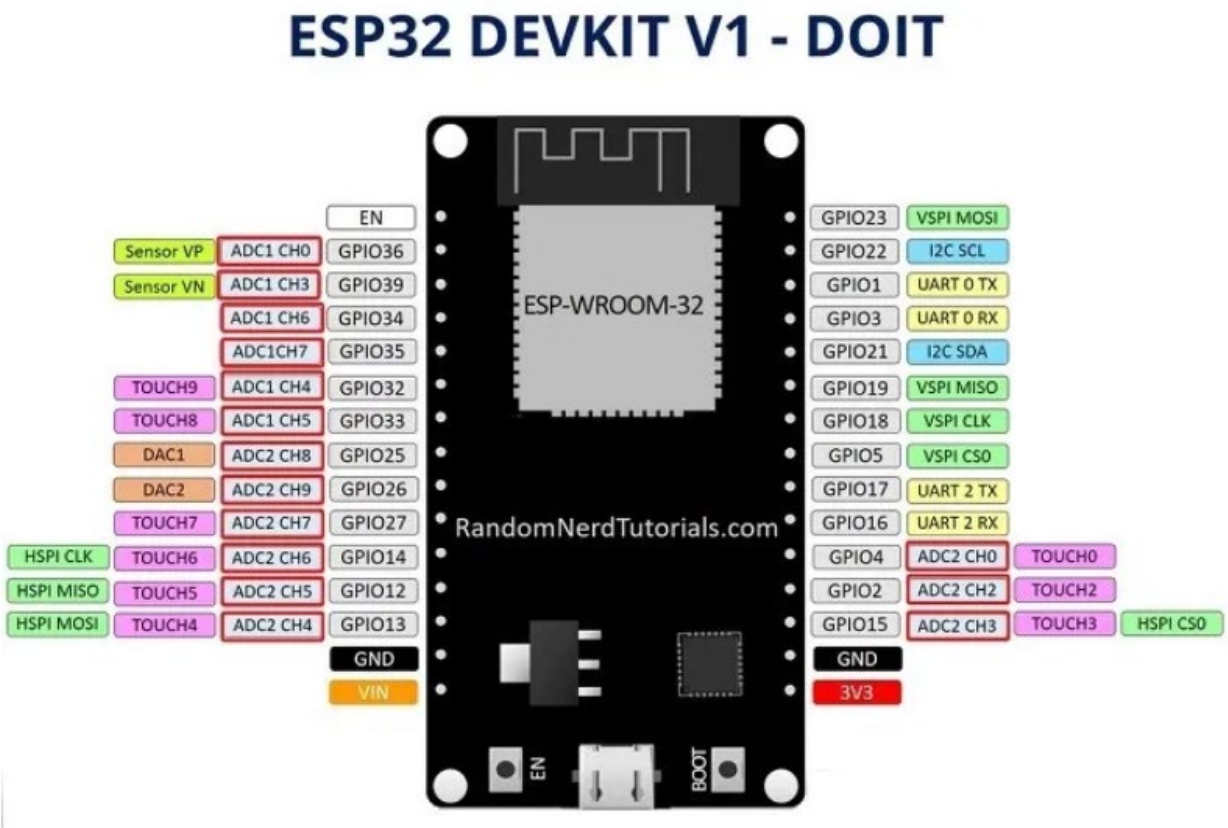
2022

BAB I
PENDAHULUAN

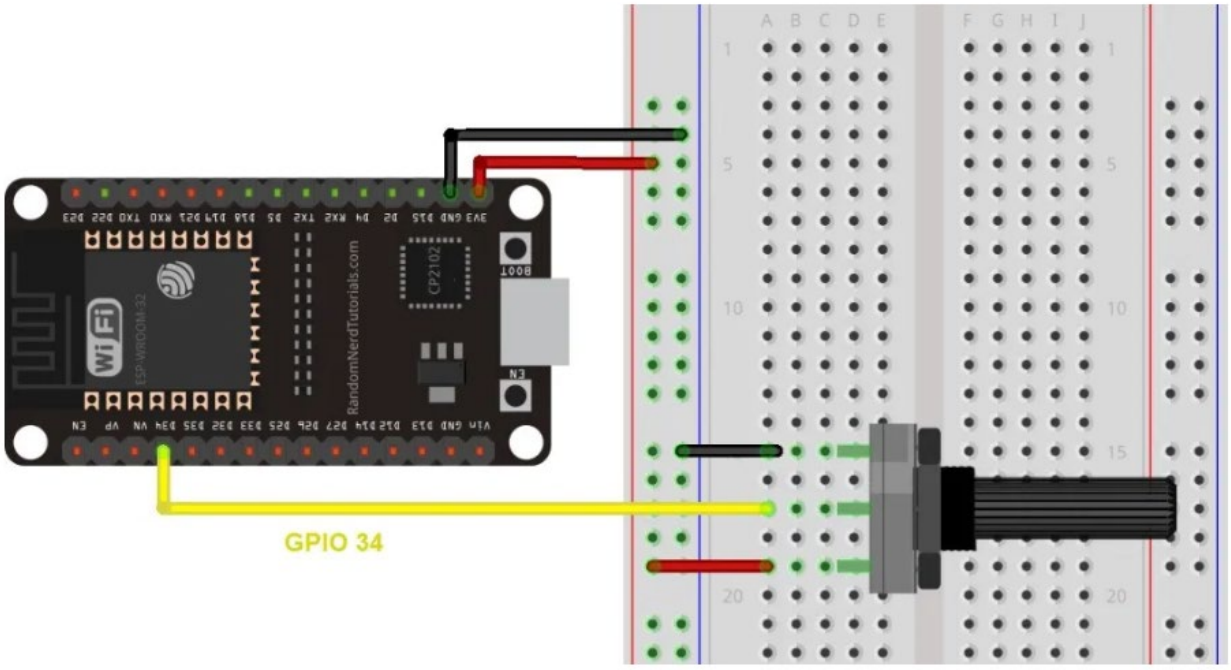
A. Latar Belakang

1. ESP32 ADC

Analog to Digital Converter (ADC) adalah rangkaian yang mengubah nilai tegangan kontinu (analog) menjadi nilai biner (digital) yang dapat dimengerti oleh perangkat digital sehingga dapat digunakan untuk komputasi digital. Dengan kata lain Analog to Digital Converter adalah perangkat yang menjadi perantara untuk mengubah sinyal analog menjadi sinyal Digital agar dimengerti oleh mikrokontroler dan mikroprosesor. Tidak semua pin ESP32 dapat dihubungkan menjadi Analog to Digital Converter (ADC).

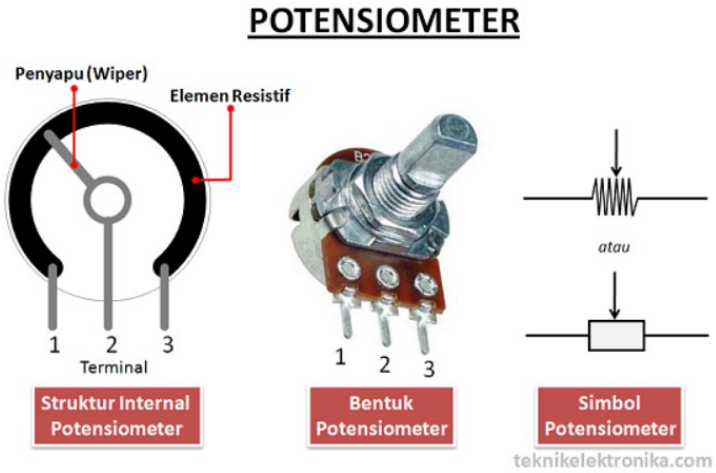


Tegangan yang diukur kemudian ditetapkan ke nilai antara 0 dan 4095, di mana 0 V sesuai dengan 0, dan 3,3 V sesuai dengan 4095. Setiap tegangan antara 0 V dan 3,3 V akan diberikan nilai yang sesuai di antaranya. ESP32 ADC dirangkai dengan komponen-komponen lain. Dengan skema potensiometer dihubungkan ke ESP32 GPIO 34, VCC dan GND menggunakan kabel.



2. Potensiometer

Potensiometer adalah salah satu jenis Resistor yang nilai resistansinya dapat diatur sesuai dengan kebutuhan rangkaian elektronika ataupun kebutuhan pemakainya. Potensiometer memiliki struktur internal dengan bentuk dan simbolnya.



Bagian-bagian penting dalam Komponen Potensiometer adalah :

1. Penyapu atau disebut juga dengan Wiper
2. Element Resistif
3. Terminal

Cara kerja Potensiometer yang terdiri dari sebuah elemen resistif yang membentuk jalur (track) dengan terminal di kedua ujungnya. Sedangkan terminal lainnya (biasanya berada di

tengah) adalah Penyapu (Wiper) yang dipergunakan untuk menentukan pergerakan pada jalur elemen resistif (Resistive). Pergerakan Penyapu (Wiper) pada Jalur Elemen Resistif inilah yang mengatur naik-turunnya Nilai Resistansi sebuah Potensiometer.

3. Kabel USB ESP32

Kabel USB ini digunakan untuk mengupload code pada Arduino IDE dari laptop kedalam ESP32.

4. Kabel Jumper

Kabel Jumper adalah kabel elektrik yang memiliki pin konektor di setiap ujungnya dan memungkinkanmu untuk menghubungkan dua komponen yang melibatkan Arduino tanpa memerlukan solder. Kegunaan kabel jumper ini adalah sebagai konduktor listrik untuk menyambungkan rangkaian listrik. Biasanya kabel jumper digunakan pada breadboard atau alat prototyping lainnya agar lebih mudah untuk mengutak-atik rangkaian. Konektor yang ada pada ujung kabel terdiri atas dua jenis yaitu konektor jantan (male connector) dan konektor betina (female connector). Prinsip kerja kabel jumper yaitu menghantarkan arus listrik dari satu komponen ke komponen lainnya yang dihubungkan.

B. Deskripsi Kasus

Pada kasus ini disediakan ESP32, dua buah potensiometer, dan kabel jumper. Dua buah potensiometer dihubungkan ke ESP32 dengan tujuan Analog to Digital Converter (ADC). Selanjutnya dilakukan proses coding pada ESP32 ADC sesuai dengan potensiometer yang terhubung. Setelah itu hasil yang didapatkan dilanjutkan ke dalam Python Charting. Untuk ditampilkan ke dalam bentuk grafik.

C. Tujuan

Hal ini bertujuan untuk menjalankan ESP32 sebagai Analog to Digital Converter (ADC) yang terhubung dengan dua potensiometer dimana digambarkan sebagai sensor real time. Setelah itu, data-data yang didapatkan dilanjutkan menjadi Python Chart.

BAB II
CODE ARDUINO DAN PYTHON

A. Kode Pada Arduino

```
#include <Arduino.h>

float pot1;
float pot2;

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600); //turn on serial monitor
  pinMode(34, INPUT);
  pinMode(35, INPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
  pot1 = analogRead(34);
  pot2 = analogRead(35);


  Serial.print(float(pot1));
  Serial.print(",");
  Serial.println(float(pot2));
  delay(250); //Pause between readings.
}
```

```
#include <Arduino.h>
float pot1;
float                                                    pot2;
```

Codingan ESP32: masing2 potensio diwakili variabel pot1 dan pot2.

```
void setup() {
 // put your setup code here, to run once:
 Serial.begin(9600); //turn on serial monitor
 pinMode(34, INPUT);
 pinMode(35, INPUT);
}
```

Dalam void setup() (program yg dijalankan sekali) terdapat perintah untuk memulai komunikasi, menetapkan pin yg digunakan sebagai input.

```
void loop() {
 // put your main code here, to run repeatedly:
 pot1 = analogRead(34);
 pot2 = analogRead(35);
```

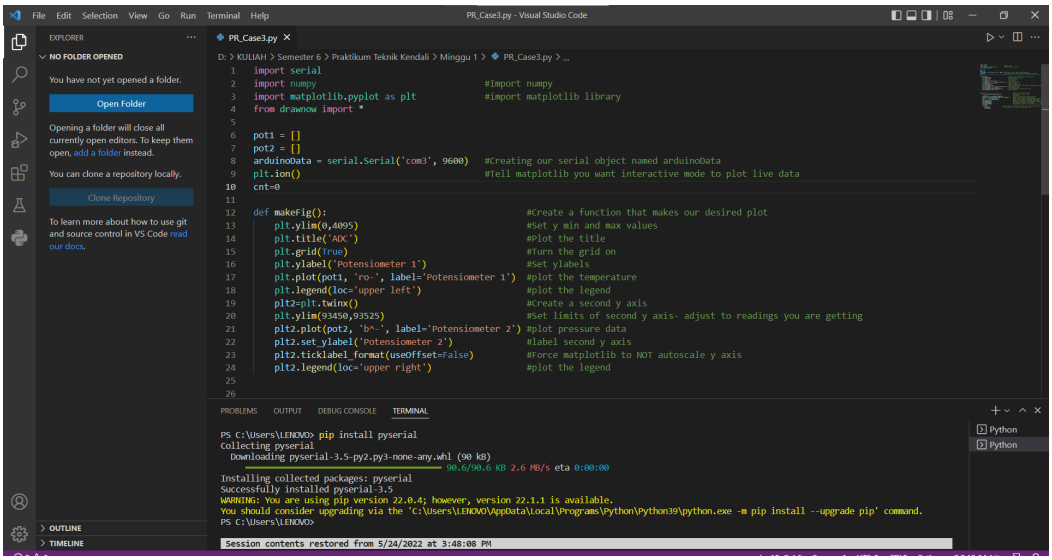
Lalu dalam void loop() (program yang dijalankan berulang) terdapat perintah untuk membaca nilai potensio menggunakan ADC yang diwakili oleh perintah analogRead.

```
Serial.print(float(pot1));
  Serial.print(",");
  Serial.println(float(pot2));
  delay(250); //Pause between readings.
}
```

Lalu terdapat serial.print untuk mengirim data ke usb.

B. Kode Pada Python (Visual Code)

Gambar Kode




```
import serial
import numpy                                    #Import numpy
import matplotlib.pyplot as plt                 #import matplotlib
library
from drawnow import *
```

Untuk membuat grafik, perlu 4 library, serial, numpy, matplotlib, drawnow.

- Serial = Mengambil data dari usb
- Numpy = Untuk perhitungan matematis
- Matplotlib = Membuat grafik
- Drawnow = Mengupdate grafik secara berkala

```
    pot1 = []
    pot2 = []
```

```
      arduinoData = serial.Serial('com3', 9600)    #Creating our
serial object named arduinoData
      plt.ion()                                    #Tell matplotlib
you want interactive mode to plot live data
      cnt=0
```

```
11
12  def makeFig():                                 #Create a function that makes our desired plot
13      plt.ylim(0,4095)                           #Set y min and max values
14      plt.title('ADC')                           #Plot the title
15      plt.grid(True)                             #Turn the grid on
16      plt.ylabel('Potensiometer 1')              #Set ylabels
17      plt.plot(pot1, 'ro-', label='Potensiometer 1')  #plot the temperature
18      plt.legend(loc='upper left')               #plot the legend
19      plt2=plt.twinx()                           #Create a second y axis
20      plt.ylim(93450,93525)                      #Set limits of second y axis- adjust to readings you are getting
21      plt2.plot(pot2, 'b^-', label='Potensiometer 2') #plot pressure data
22      plt2.set_ylabel('Potensiometer 2')         #label second y axis
23      plt2.ticklabel_format(useOffset=False)     #Force matplotlib to NOT autoscale y axis
24      plt2.legend(loc='upper right')             #plot the legend
25
```

"def makeFig", merupakan fungsi untuk membuat tampilan grafik pada windows.

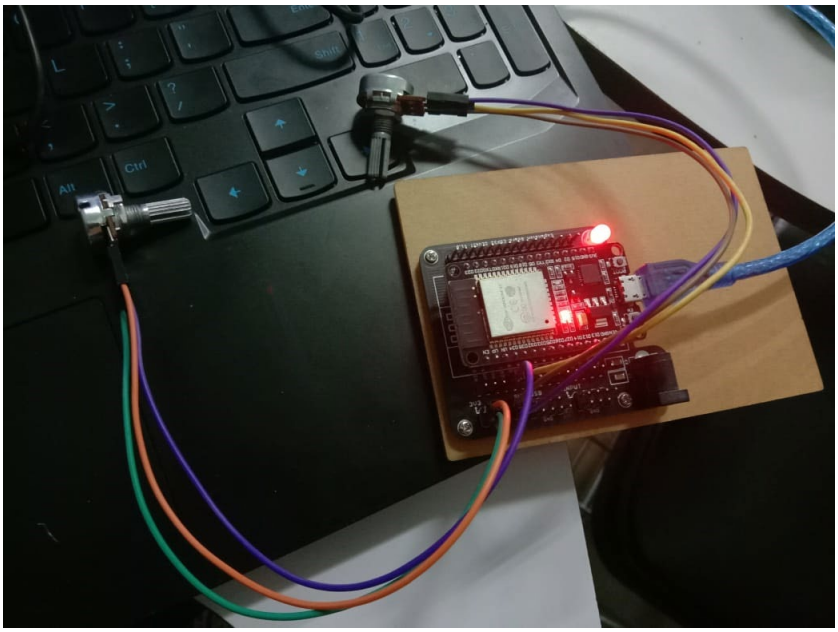
```
26
27  while True:                                    #While loop that loops forever
28      while (arduinoData.inWaiting()==0):        #Wait here until there is data
29          pass                                   #do nothing
30      arduinoString = arduinoData.readline()     #read the line of text from the serial port
31      dataArray = arduinoString.decode('utf-8').split(',')  #Split it into an array called dataArray
32      pot_1 = float( dataArray[0])               #Convert first element to floating number and put in temp
33      pot_2 =    float( dataArray[1])            #Convert second element to floating number and put in P
34      pot1.append(pot_1)                         #Build our tempF array by appending temp readings
35      pot2.append(pot_2)                         #Building our pressure array by appending P readings
36      drawnow(makeFig)                           #Call drawnow to update our live graph
37      plt.pause(.000001)                         #Pause Briefly. Important to keep drawnow from crashing
38      cnt=cnt+1
39      if(cnt>50):                     #If you have 50 or more points, delete the first one from the array
40          pot1.pop(0)                 #This allows us to just see the last 50 data points
41          pot2.pop(0)
```

Lalu dibawahnya terdapat "While True" dimana tujuannya agar program berjalan terus-menerus melakukan loop.

Di dalam loop tersebut, ada perintah untuk mengambil, mengolah dan menyusun data ke dalam array sehingga data2 yang sudah dikumpulkan dapat ditampilkan dalam grafik.

BAB III
TESTING RANGKAIAN & SIMULASI

A. Rangkaian Modul

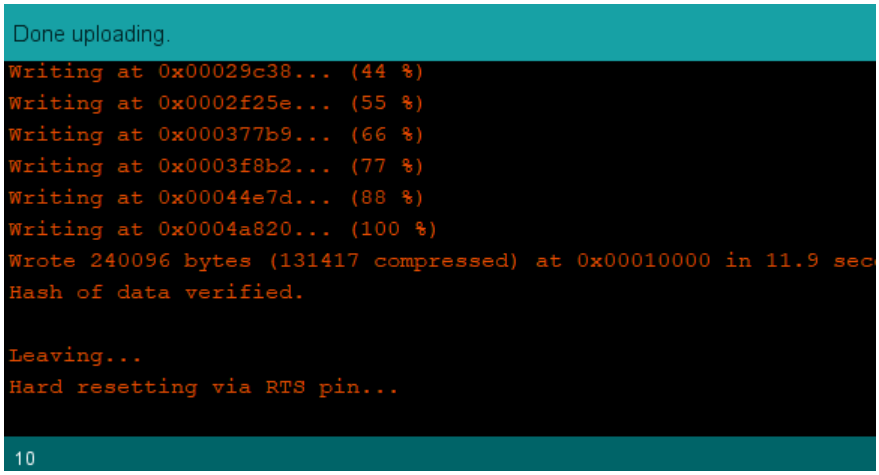


B. Testing Simulasi

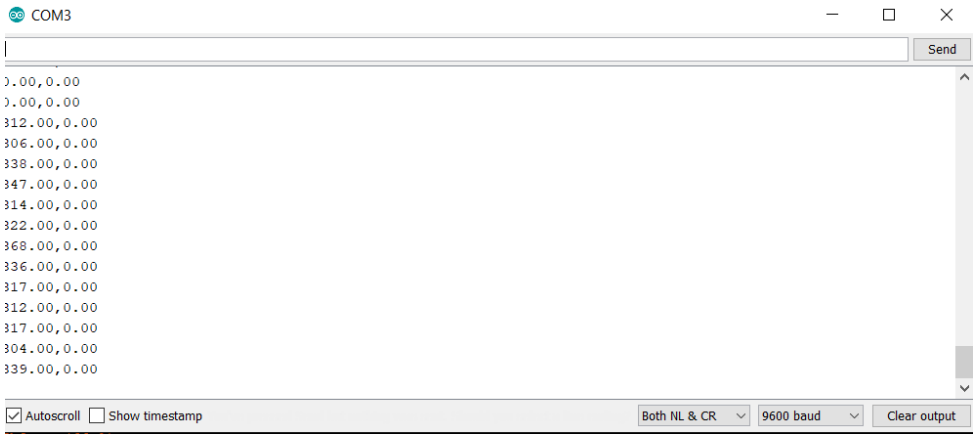
Jika program dan peralatan yang dibutuhkan sudah ada maka selanjutnya adalah tes rangkaian

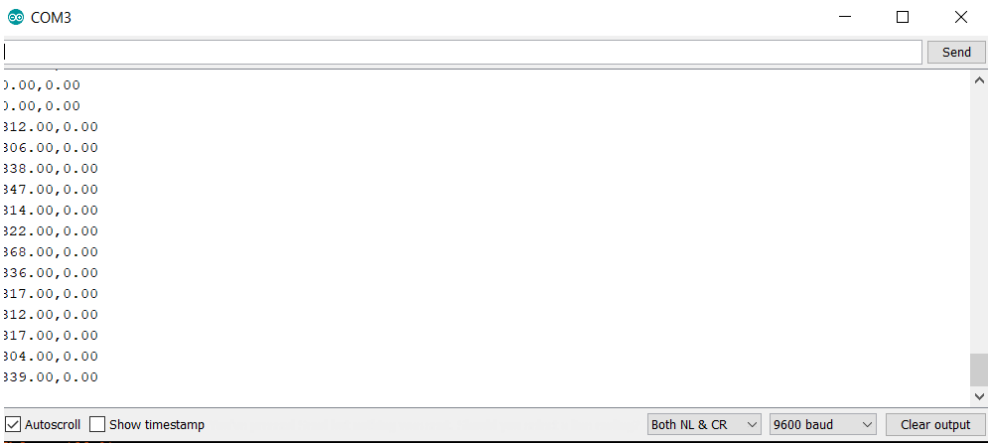
- Pertama masuk ke code arduino dan klik upload pada menu

- Tunggu beberapa saat sampai muncul “ Done Uploading”

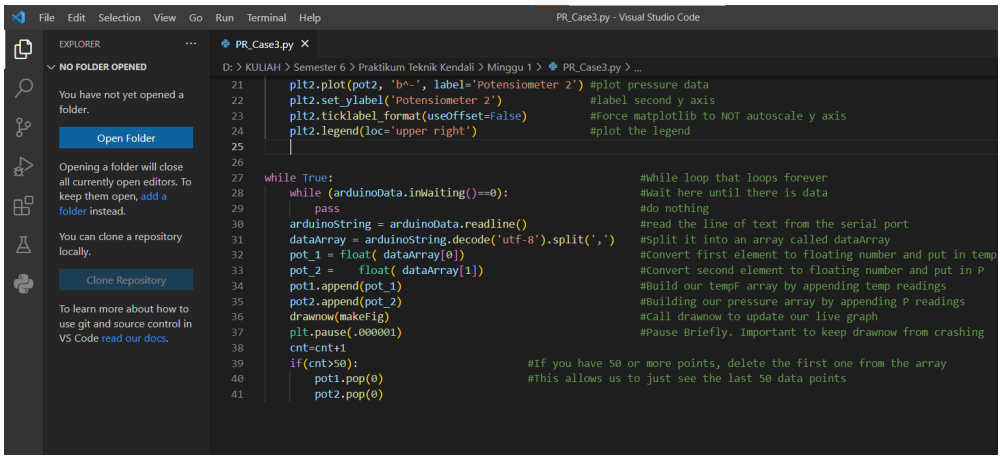


- Periksa pada serial monitor untuk mengecek apakah potensio 1 dan potensio 2 sudah masuk arduino ide apa belum. Cek dengan memutar potensio satu satu.





- Jika sudah aman pada Arduino Ide. Lalu pindah ke Code Python di Visual Studio. Selanjutnya klik run di pojok kanan.



- Jika sudah selesai runing program maka muncul tampilan display dari grafik yang yang ingin ditampilkan. Lalu periksa masing-masing potensiometer agar dapat memperlihatkan grafik yang berbeda.

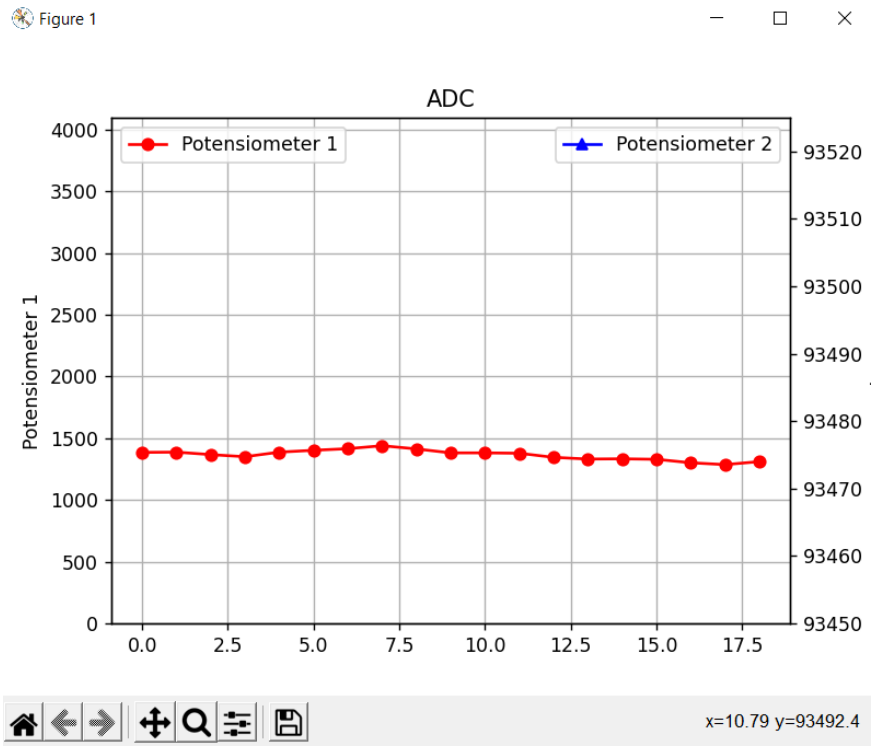


Figure 1
ADC
Potensiometer 1
Potensiometer 2
Potensiometer 1
4000
3500
3000
2500
2000
1500
1000
500
0
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
93520
93510
93500
93490
93480
93470
93460
93450
x=10.79 y=93492.4

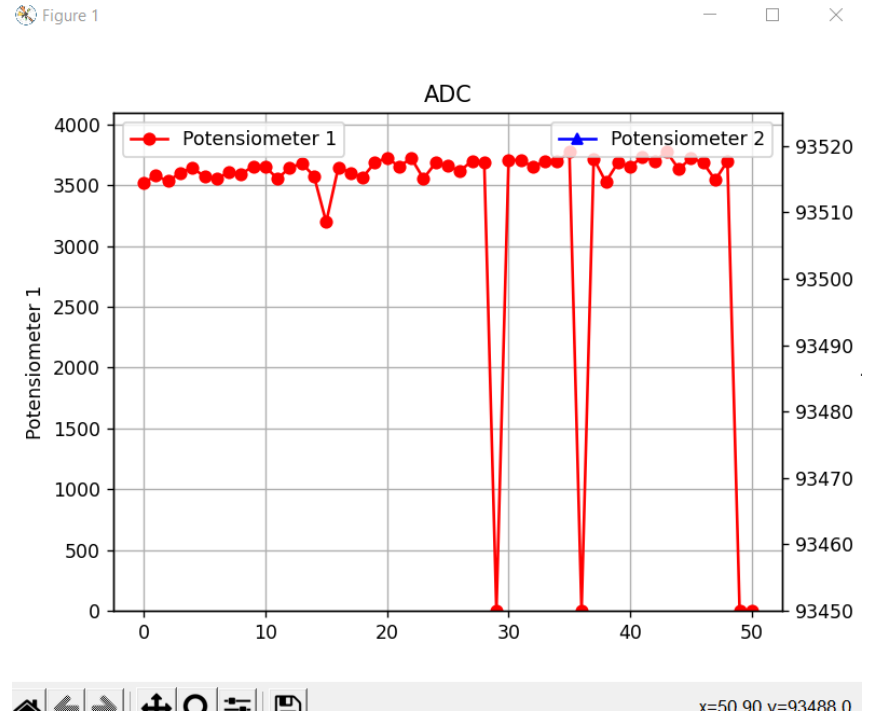


Figure 1
ADC
Potensiometer 1
Potensiometer 2
Potensiometer 1
4000
3500
3000
2500
2000
1500
1000
500
0
0
10
20
30
40
50
93520
93510
93500
93490
93480
93470
93460
93450

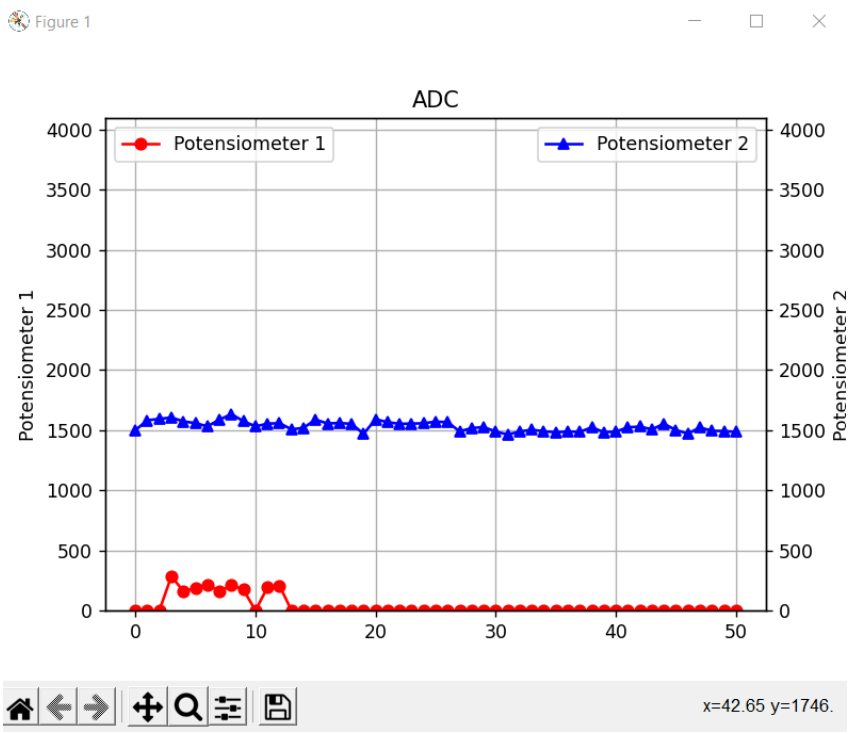
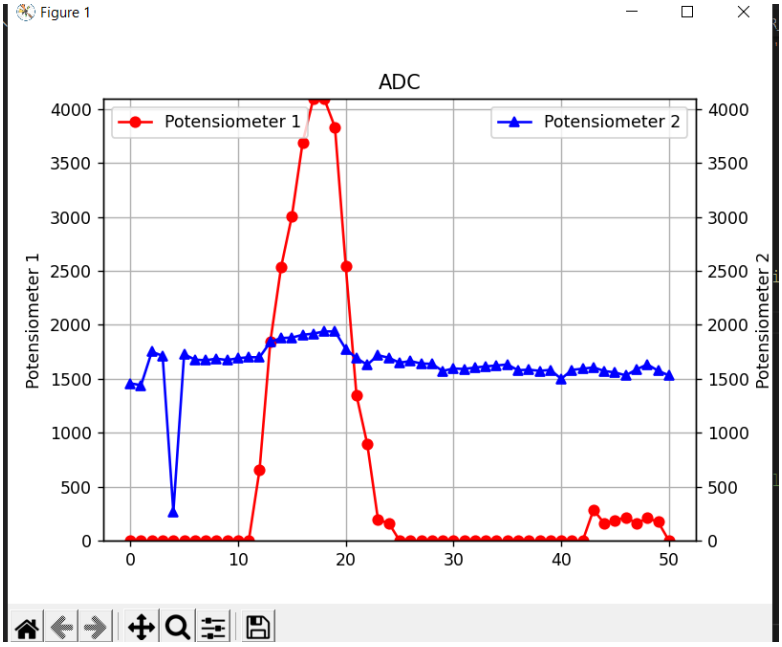


Figure 1
ADC
Potensiometer 1
Potensiometer 2
Potensiometer 1
Potensiometer 2
4000
3500
3000
2500
2000
1500
1000
500
0
0
10
20
30
40
50
x=42.65 y=1746.

BAB IV
KESIMPULAN



Pada rangkaian case 3 (*ESP32 ADC to Python Charting)* ini dapat disimpulkan bahwa setiap inputan potensio pada pot 1 dan pot 2 menghasilkan output yang berbeda dan grafik berjalan secara realtime. Hal ini dapat dilihat pada gambar. Bahwa pot 1 bergerak dengan warna merah dan pot 2 bergerak dengan warna biru.