**Πείραμα με control ποτενσιόμετρο 5 ΚΩ**

1. **Περιγραφή:**

Στο πείραμα που ακόλουθει χρησιμοποιώ ένα ποτενσιόμερο 5 ΚΩ περιστρεφόμενο όπου στην έξοδο του παίρνω μεταβαλλόμενη τάση .

1. **Υλικά:**

|  |  |  |
| --- | --- | --- |
|  |  |  |
| **1\* Raspberry pi** | **1\* GPIO Extension board** | **1\* 40 pin colorful jumper wires** |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| **1\* Breadboard** | **Jumper Wires** | **One (1) control with 5kΩ potentiometer** | **1\*analog to digital signal converter and vice versa** |

1. **Γνώση υλικού:**

Το περιστροφικό ποτενσιόμετρο είναι μια ρυθμιζόμενη ηλεκτρική αντίσταση που μπορεί να μετακινηθεί με περιστροφική κίνηση. Αυτό επιτρέπει την εφαρμογή διαδικασιών ελέγχου σε διάφορα ηλεκτρικά ή ηλεκτρονικά συστήματα, για παράδειγμα για τον έλεγχο της έντασης ήχου σε συσκευές πολυμέσων.

Ένα περιστροφικό ποτενσιόμετρο αποτελείται από τρεις επαφές, έναν φορέα και ένα κυκλικό υλικό αντίστασης. Οι δύο εξωτερικές επαφές είναι σταθερές και συνδέονται και οι δύο με την αντίσταση. Η τρίτη επαφή είναι κινητή και είναι γνωστή ως ρυθμιστής. Στις συμβατικές κατασκευές, αυτή μπορεί να περιστραφεί έως και 300 μοίρες μέσω ενός περιστροφικού άξονα. Με τη μετατόπιση του ρυθμιστή, η ενσωματωμένη αντίσταση γίνεται είτε μεγαλύτερη είτε μικρότερη. Με αυτόν τον τρόπο, για παράδειγμα, μπορούν να ελεγχθούν μεταβλητά οι τάσεις σε ένα κύκλωμα ή να δημιουργηθούν εξωτερικές αντιστάσεις.

**Πειραματική συσκευή:**

* Πίνακας ελέγχου Raspberry Pi
* Μονάδα ποτενσιόμετρου
* Γραμμές σύνδεσης
* Μονάδα μετατροπής ψηφιακού σε ανλογικό σήμα

**Πειραματική σύνδεση:**

|  |  |
| --- | --- |
| The 5 ΚΩ potentiometer module | Raspberry Pi |
| Vcc | 5V |
| Gd | GND |
| OUT | Ain0(ADC converter module) |

|  |  |
| --- | --- |
| **SCL** | **SCL** Raspberry Pi pin |
| **SDA** | **SDA** Raspberry Pi pin |
| **VCC** | **5V** Raspberry Pi pin |
| **GND** | **GND** Raspberry Pi pin |

1. **Πειραματικό συμπέρασμα:**

Για να συνδέσουμε ένα περιστροφικό ποτενσιόμετρο 5kΩ και να διαβάσουμε την αναλογική τιμή του χρησιμοποιώντας ένα Raspberry Pi χωρίς εξωτερικό ADC(analog to digital converter).

Αυτός ο κώδικας διαβάζει την αναλογική τιμή από το περιστροφικό ποτενσιόμετρο που είναι συνδεδεμένο στον καθορισμένο ακροδέκτη GPIO και εκτυπώνει την τιμή. Μπορείτε να χρησιμοποιήσετε αυτή την τιμή για να ελέγξετε διάφορες λειτουργίες στο Raspberry Pi.

**Βεβαιωθείτε ότι έχετε εγκαταστήσει τη βιβλιοθήκη RPi.GPIO στο Raspberry Pi σας, αν δεν είναι ήδη εγκατεστημένη. Μπορείτε να το κάνετε αυτό με την ακόλουθη εντολή στη γραμμή εντολών(command line):**

**pip install RPi.GPIO**

Σημαντική επισήμανση για αναλογικές τιμές εισόδου στο Raspberry Pi:

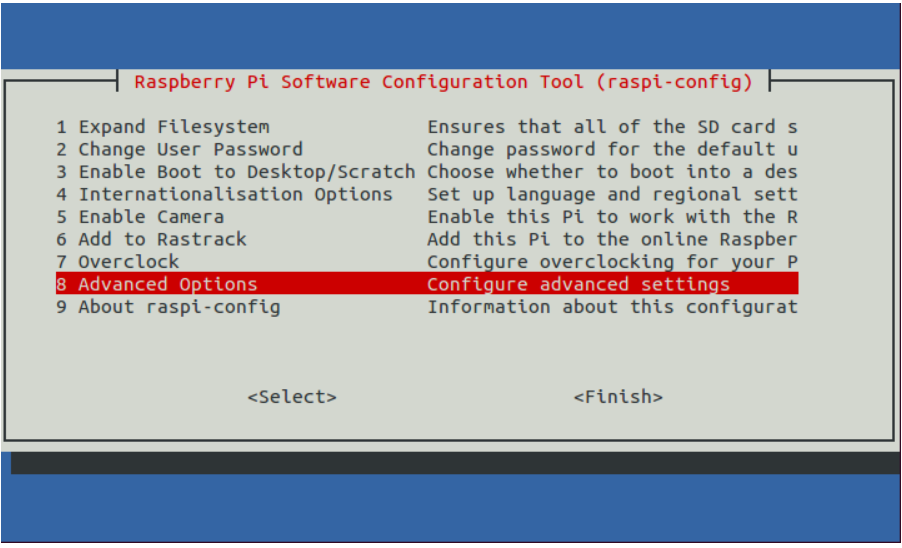
**Ενεργοποίηση Ι2C θύρας:**

**Βήμα 1**:

Ενεργοποιήστε τη θύρα I2C του Raspberry Pi σας (Αν έχετε κάνει την ενεργοποίηση, παραλείψτε αυτό. Αλλά αν δεν είστε σίγουροι αν την έχετε ενεργοποιήσει ή όχι, συνεχίστε):

sudo raspi-config

**Advanced options**

****

**A screenshot of a computer

Description automatically generated**

**Βήμα 2:**

**Ελέγξτε ότι οι μονάδες i2c είναι φορτωμένες και ενεργές:**

lsmod **|** grep i2c

**Τότε θα εμφανιστεί ο ακόλουθος κώδικας (ο αριθμός μπορεί να είναι διαφορετικός)**

i2c\_dev                     6276    0

i2c\_bcm2708                 4121    0

**Βήμα 3:**

**Εγκαταστήστε τα i2c-tools**

sudo apt-get install i2c-tools

**Βήμα 4:**

**Ελέγξτε τη διεύθυνση της συσκευής I2C:**

**A screenshot of a computer

Description automatically generated**

**Αν είναι συνδεδεμένη μια συσκευή I2C, τα αποτελέσματα θα είναι παρόμοια με τα παραπάνω - αφού η διεύθυνση της συσκευής είναι 0x48, εκτυπώνεται η τιμή 48.**

**Βήμα 5:**

sudo apt-get install python-smbus

Ανεβάστε τον κώδικα στην αναπτυξιακή πλακέτα και εκτελέστε τον.

1. **Κώδικας Python:**

#Terminal prints the analog value read by adjustable potentiometer. The LEd brightness will vary with the the rotary of potentiometer

import smbus

import time

address = 0x48 #default address of PCF8591

bus=smbus.SMBus(1) #Create an instance of smbus

cmd=0x40 #command

# A0 = 0x40 ##A0 ----> port address

# A1 = 0x41

# A2 = 0x42

# A3 = 0x43

def analogRead(chn): #read ADC value,chn:0,1,2,3

    value = bus.read\_byte\_data(address,cmd+chn)

    return value

def analogWrite(value):#write DAC value

    bus.write\_byte\_data(address,cmd,value)

def loop():

   while True:

        value = analogRead(0) #read the ADC value ofchannel 0

        analogWrite(value) #write the DAC value

        voltage = value / 255.0 \* 3.3 #calculate the voltage value

        print ('ADC Value : %d, Voltage : %.2f'%(value,voltage))

        time.sleep(0.01)

def destroy():

    bus.close()

if \_\_name\_\_ == '\_\_main\_\_':

    print ('Program is starting ... ')

    try:

        loop()

    except KeyboardInterrupt:

        destroy()

#Smbus is based on iic communication. We treat it as iic communication library.

#     bus.read\_byte\_data(address,cmd+chn)      ---->>>>> Read the corresponding modules with iic address，

# address is the address of pcf8591 module，cmd+chn correspond to the address

# of analog port pcf8591: A0 = 0x40，A1 = 0x41，A2 = 0x42，A3 = 0x43

#    bus.write\_byte\_data(address,cmd,value)    ------->>>>>    D/A analog value outputs, address is address of pcf8591 module，

#      cmd outputs the address of pins，value: output value

#  Smbus library file ------>>>>    https://pypi.org/project/smbus2/0.1.2/