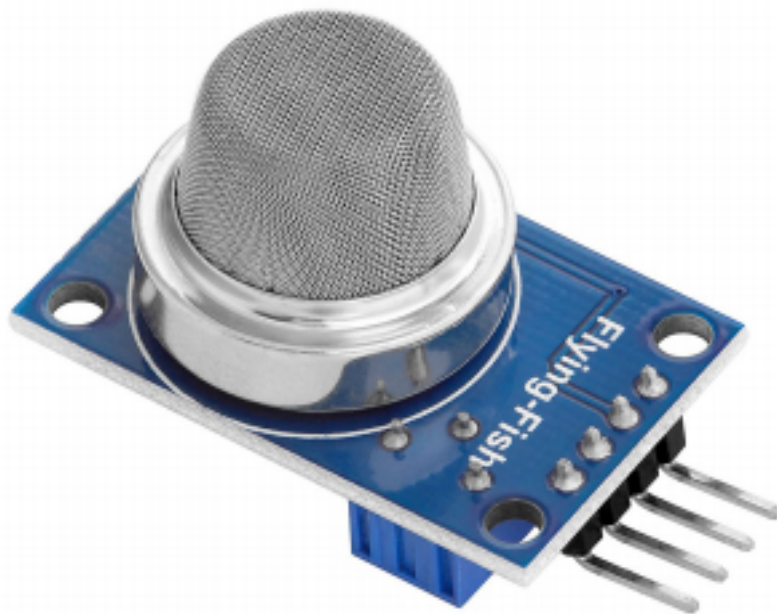




Benvenuto!

Grazie per aver acquistato il nostro *Modulo Sensore di Gas MQ-2* *AZ-Delivery* . Nelle pagine seguenti, ti illustreremo come utilizzare e configurare questo pratico dispositivo.

Buon divertimento!





Indice dei Contenuti

Introduzione	3
Specifiche	4
La piedinatura	5
Come configurare il Raspberry Pi e il Python	11
Collegamento del modulo con ATmega328p	12
Esempio di sketch	13
Collegare Nano V3.0 come ADC per Raspberry Pi	17
Collegamento del modulo con Raspberry Pi	22
Script Python per il modulo MQ-2	29



Introduzione

Il modulo sensore di gas MQ-2 è un dispositivo utilizzato per rilevare e misurare la concentrazione di gas nell'aria. Può rilevare gas come: GPL, propano, metano, idrogeno, alcool, fumo e monossido di carbonio. Anche se può rilevare questi gas, non è in grado di distinguere la differenza tra loro.

L'MQ-2 è un Semiconduttore in Ossido di Metallo (MOS), noto anche come chemiresistore. Il sensore contiene un materiale di rilevamento la cui resistenza cambia con diverse concentrazioni di gas. Questo cambiamento della resistenza è utilizzato per la rilevazione del gas. Il sensore ha anche un potenziometro incorporato, con il quale possiamo regolare la sua sensibilità.

Il sensore è racchiuso all'interno di due strati di maglia fine in acciaio inossidabile chiamata *Rete Antiesplorazione*. Come risultato, è in grado di rilevare gas infiammabili senza incidenti. Allo stesso modo, protegge il sensore e filtra le particelle in sospensione. In questo modo, solo i gas possono passare all'interno della camera di rilevamento.

Il modulo ha un chip comparatore LM393 a bordo che converte le letture in segnali digitali e analogici. C'è anche un potenziometro che viene utilizzato per calibrare la sensibilità di rilevamento.



Specifiche

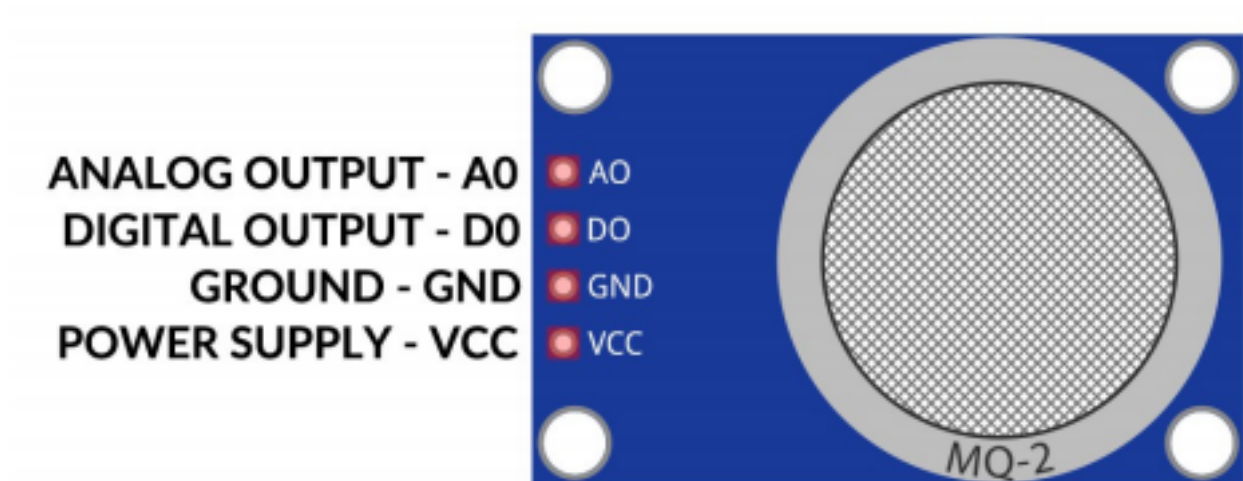
Tensione di esercizio:	5V
Tensione di esercizio:	150mA
Consumo energetico:	900mW
Resistenza al carico:	20k Ω
Resistenza del riscaldatore:	33 Ω +5%
Resistenza di rilevamento	10k Ω - 60k Ω
Tempo di preriscaldamento:	24h
Ambito di concentrazione:	200 – 10000ppm (parti per milione)
Uscita:	analogico, digitale
Dimensioni:	33x21x22mm (1.3x0.8x0.9in)

Per i migliori risultati di rilevamento, il sensore di gas deve essere preriscaldato. Il miglior tempo di preriscaldamento per il sensore è superiore a 24 ore. Per informazioni dettagliate sulle specifiche del sensore, fare riferimento alla scheda tecnica.

La sensibilità del modulo può essere regolata con un potenziometro di bordo. Spostando l'albero del potenziometro in senso orario si aumenta la sensibilità. Spostando l'albero del potenziometro in senso antiorario si riduce la sensibilità del modulo.

La piedinatura

Il modulo del sensore di gas ha quattro pin. La piedinatura è mostrata nell'immagine seguente:



NOTA: Il Raspberry Pi non ha un convertitore digitale-analogico e non può essere usato per leggere tensioni analogiche.

Come configurare l'Arduino IDE

Se l'Arduino IDE non è installato, seguire il [link](#) e scaricare il file di installazione del sistema operativo scelto.

Download the Arduino IDE



The screenshot shows the Arduino IDE download page. On the left, there is a teal circle with a white infinity symbol. To its right, the text reads: **ARDUINO 1.8.9**, followed by a description of the IDE as open-source software that runs on Windows, Mac OS X, and Linux. On the right side of the page, there are links for downloading the IDE for different operating systems: Windows (installer and ZIP file), Windows app, Mac OS X, and Linux (32 bits, 64 bits, ARM 32 bits, ARM 64 bits). There are also links for Release Notes, Source Code, and Checksums (sha512).

Per gli utenti *Windows*, fare doppio clic sul file .exe scaricato e seguire le istruzioni nella finestra di installazione.

Az-Delivery

Per gli utenti Linux, scaricare un file con estensione *.tar.xz*, che è necessario estrarre. Quando lo si estrae, andare nella directory estratta, e aprire il terminale in quella directory. È necessario eseguire due *script .sh*, il primo chiamato *arduino-linux-setup.sh* e il secondo chiamato *install.sh*.

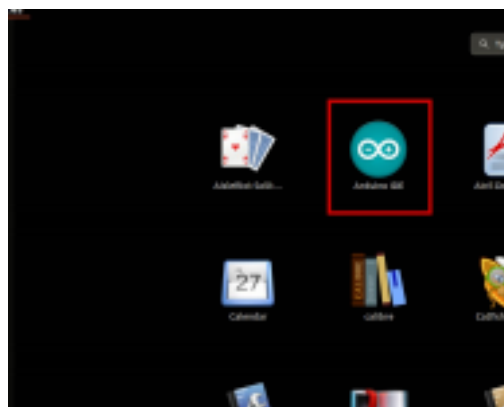
Per eseguire lo script, aprire il terminale nella directory in cui è stato salvato lo script ed eseguire il seguente comando:

sh arduino-linux-setup.sh user_name

user_name - è il nome di un superutente nel sistema operativo Linux.. All'avvio del comando deve essere inserita una password per il superutente. Aspettate qualche minuto che lo script completi tutto.

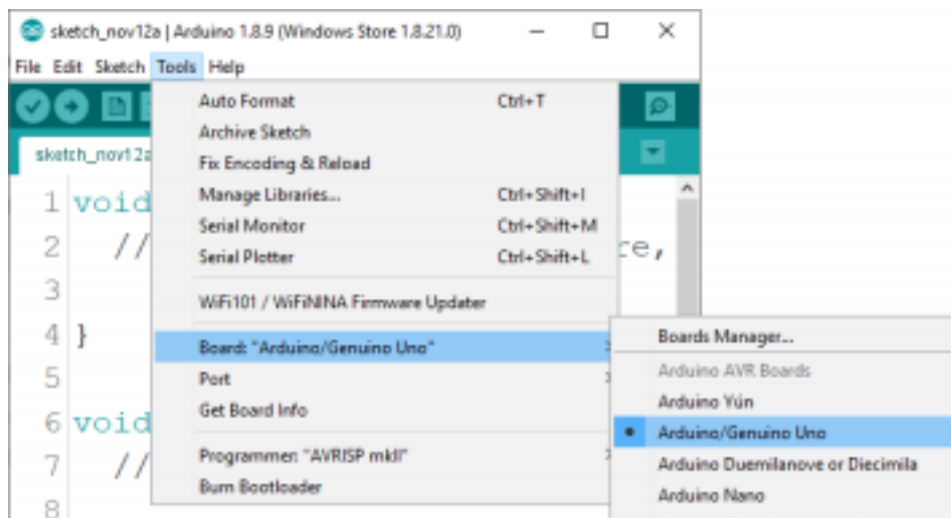
Il secondo script, chiamato *install.sh*, deve essere usato dopo l'installazione del primo script. Eseguire il seguente comando nel terminale (directory estratta): **sh install.sh**

Dopo l'installazione di questi script, andare su *Tutte le App*, dove troverai l'*Arduino IDE* installato.



Quasi tutti i sistemi operativi sono dotati di un editor di testo preinstallato (ad esempio *Windows* viene fornito con *Notepad*, *Linux Ubuntu* viene fornito con *Gedit*, *Linux Raspbian* viene fornito con *Leafpad*, ecc.). Tutti questi editor di testo sono perfettamente adatti allo scopo dell'eBook.

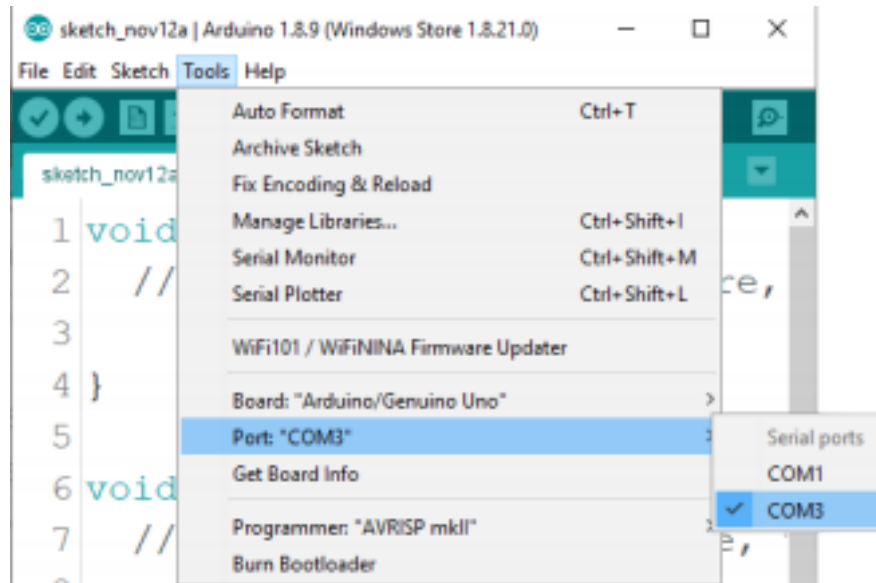
La prossima cosa da fare è controllare se il PC è in grado di rilevare la scheda microcontrollore. Aprite l'Arduino IDE appena installato e andate su: *Strumenti > Scheda > {your board name here}*
{your board name here} dovrebbe essere l'*Microcontrollore/Genuino Uno*, come si può vedere nella seguente immagine:



Deve essere selezionata la porta alla quale è collegata la scheda microcontrollore. Vai su: *Strumenti > Porta > {port name goes here}* e quando la scheda microcontrollore è collegata alla porta USB, il nome della porta è visibile nel menu a tendina dell'immagine precedente.

Az-Delivery

Se si utilizza l'Arduino IDE su Windows, i nomi delle porte sono i seguenti:



Per gli utenti Linux, il nome della porta è `/dev/ttyUSBx` per esempio, dove *x* rappresenta un numero intero compreso tra 0 e 9.



Come configurare il Raspberry Pi e il Python

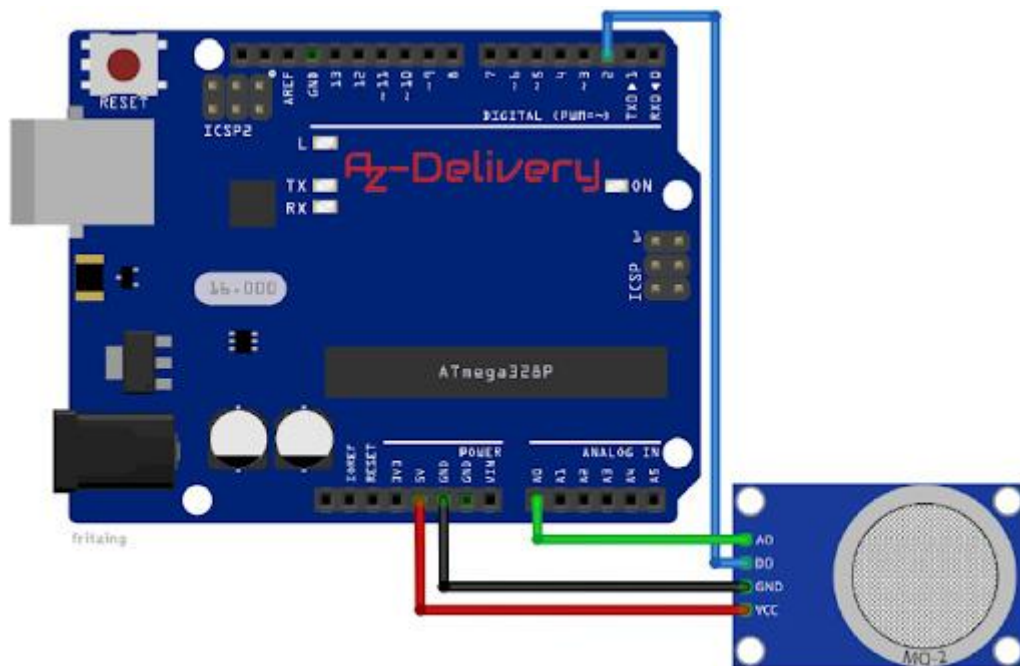
Per il Raspberry Pi, prima deve essere installato il sistema operativo, tutto deve essere impostato in modo da poter essere utilizzato in modalità Headless. La modalità Headless consente la connessione remota al Raspberry Pi, senza la necessità di uno schermo di un PC, mouse o tastiera. Le uniche cose di cui avete bisogno per questa modalità sono il Raspberry Pi, l'alimentazione e la connessione internet. Tutto questo è spiegato in dettaglio nell'eBook gratuito:

[Raspberry Pi Quick Startup Guide](#)

Il sistema operativo Raspbian viene fornito con il Python preinstallato.

Collegamento del modulo con ATmega328p

Collegare il modulo con ATmega328p come indicato nella seguente immagine:



Pin modulo	Pin ATmega 328p	Colore filo
VCC	5V	Filo rosso
GND	GND	Filo nero
D0	D2	Filo blu
A0	A0	Filo verde



Esempio di sketch

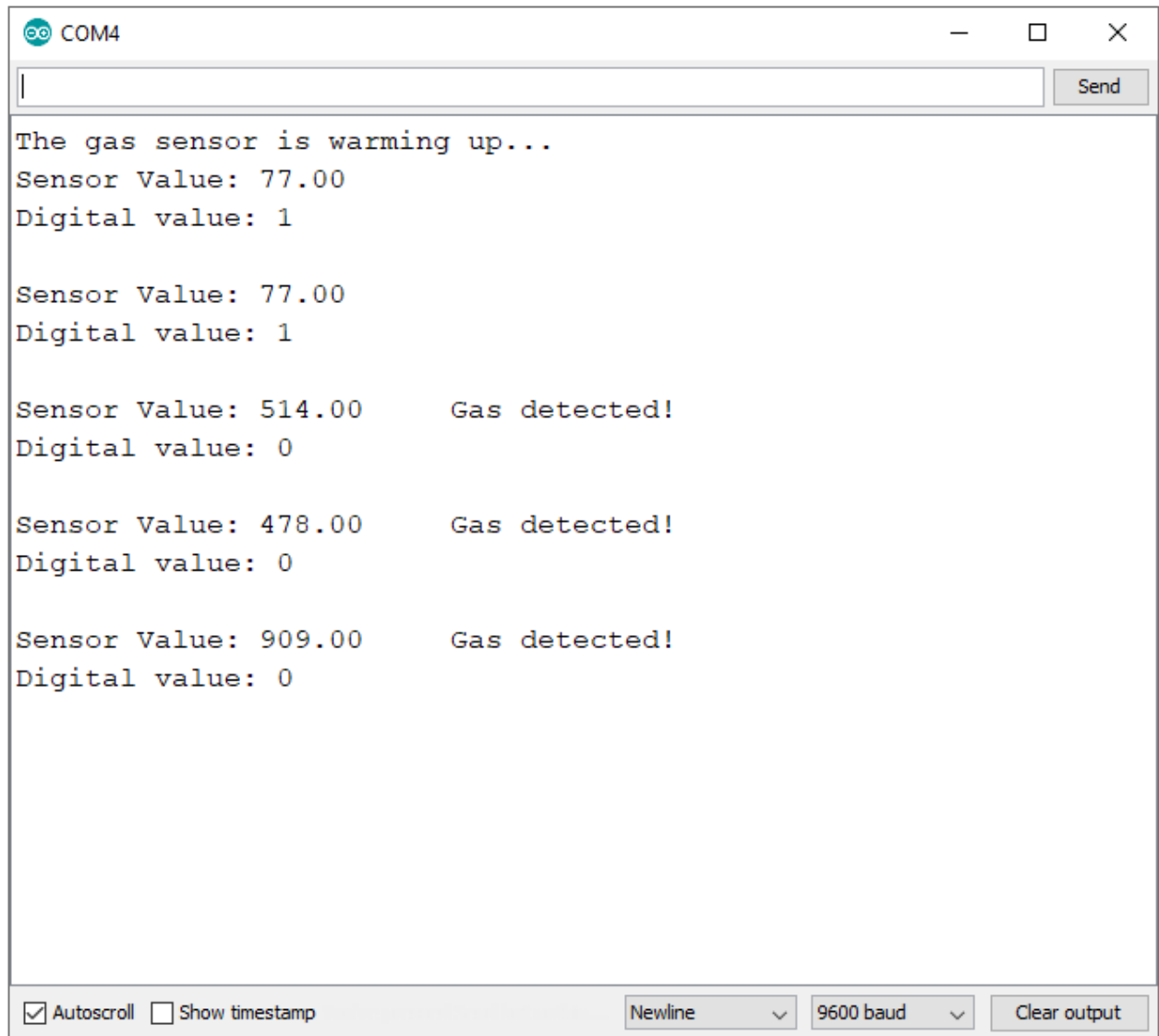
```
#define DIGITAL_PIN 2
#define ANALOG_PIN 0
uint16_t gasVal;
boolean isgas = false;
String gas;
void setup() {
  Serial.begin(9600);
  Serial.println("The sensor is warming up...");
  delay(30000);
  pinMode(DIGITAL_PIN, INPUT);
}
void loop() {

  gasVal = analogRead(ANALOG_PIN);
  isgas = digitalRead(DIGITAL_PIN);

  if (isgas) {
    gas = "No";
  }
  else {
    gas = "Yes";
  }
  gasVal = map(gasVal, 0, 1023, 0, 100);
  Serial.print("Gas detected: ");
  Serial.println(gas);
  Serial.print("Gas percentage: ");
  Serial.print(gasVal);
  Serial.print("%\n");
  delay(2000);
}
```

Az-Delivery

Caricare lo sketch su ATmega328p e aprire il Monitor Seriale (*Strumenti > Monitor Seriale*). Il risultato dovrebbe assomigliare all'immagine seguente:



The screenshot shows the Arduino IDE Serial Monitor window for COM4. The window has a title bar with standard Windows controls (minimize, maximize, close) and a 'Send' button. The main area displays the following text:

```
The gas sensor is warming up...  
Sensor Value: 77.00  
Digital value: 1  
  
Sensor Value: 77.00  
Digital value: 1  
  
Sensor Value: 514.00      Gas detected!  
Digital value: 0  
  
Sensor Value: 478.00      Gas detected!  
Digital value: 0  
  
Sensor Value: 909.00      Gas detected!  
Digital value: 0
```

At the bottom, there are checkboxes for 'Autoscroll' (checked) and 'Show timestamp' (unchecked). To the right are dropdown menus for 'Newline' and '9600 baud', and a 'Clear output' button.



Lo sketch inizia definendo e creando due macro chiamate *DIGITAL_PIN*, *ANALOG_PIN*.

Il *DIGITAL_PIN* rappresenta il pin digitale di ATmega328p che viene utilizzato per collegare il pin di uscita digitale del sensore.

ANALOG_PIN rappresenta il pin di ingresso analogico di ATmega328p che viene utilizzato per collegare il pin di uscita analogica del sensore.

I dati del modulo possono essere letti in due modi. Una è la lettura del pin di uscita analogica del modulo, l'altra è la lettura del pin di uscita digitale del modulo. Per leggere il pin di uscita analogica del modulo, la variabile chiamata *gasVal* è utilizzata per memorizzare il valore di ritorno dalla funzione *analogRead()*. Il valore di ritorno è un numero intero nell'intervallo da 0 a 1023. Per convertirla in percentuale, si utilizza la funzione *map()*. Questa è una funzione integrata dell'Arduino IDE. Ha cinque argomenti e restituisce un valore intero.

Az-Delivery

Ad esempio:

```
gasVal = map(input, in_min, in_max, out_min, out_max)
```

Il primo argomento è il valore di *input*, che è nell'intervallo da *in_min* a *in_max*. Il valore di ritorno è un numero intero nell'intervallo da *out_min* a *out_max*. Questa funzione mappa un numero nell'intervallo di input, ad un altro numero che si trova in un intervallo diverso.

Per leggere il pin di uscita digitale del modulo, la variabile *isGas* è utilizzata per memorizzare il valore di ritorno della funzione `digitalRead()`.

Alla fine della funzione *loop()*, i dati vengono visualizzati nel Monitor Seriale.

Tra due misurazioni c'è una pausa di 2 secondi:

```
delay(2000);
```



Collegare Nano V3.0 come ADC per Raspberry Pi

Poiché il Raspberry Pi non ha un Convertitore Analogico-Digitale (ADC), il compito è di rendere il Raspberry Pi capace di leggere tensioni analogiche. Per questo scopo si può usare ATmega328p o Nano V3.0. Per farlo, Nano V3.0 deve essere collegato al sistema operativo Raspbian. Nano V3.0 può leggere tensioni analogiche, e può utilizzare l'interfaccia seriale tramite la porta USB per inviare dati al Raspberry Pi.

Per prima cosa, l'IDE di Arduino deve essere installato su Raspbian. In secondo luogo, il firmware per il microcontrollore deve essere caricato su Nano V3.0 e la libreria Python deve essere scaricata.

Per fare questo, avviare Raspbian, aprire il terminale ed eseguire il seguente comando per aggiornare Raspbian:

`sudo apt-get update && sudo apt-get upgrade -y`

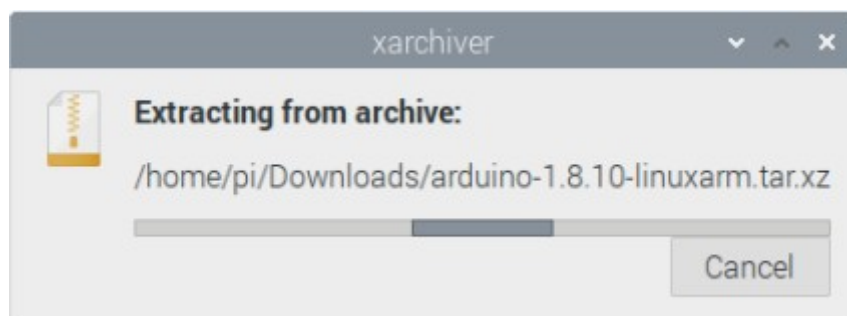
Az-Delivery

Per scaricare e installare l'IDE di Arduino, vai al [sito Arduino](#): e scaricare il file *tar.xz* di Arduino IDE per **Linux ARM 32 bit** come mostrato nella seguente immagine:

Download the Arduino IDE



Poi, estrarre il file *tar.xz*. Aprire file explorer nella directory in cui è stato scaricato il file *tar.xz*, fare clic con il tasto destro su di esso ed eseguire l'opzione Estrai Qui. Attendere qualche minuto che il processo di estrazione sia completato.

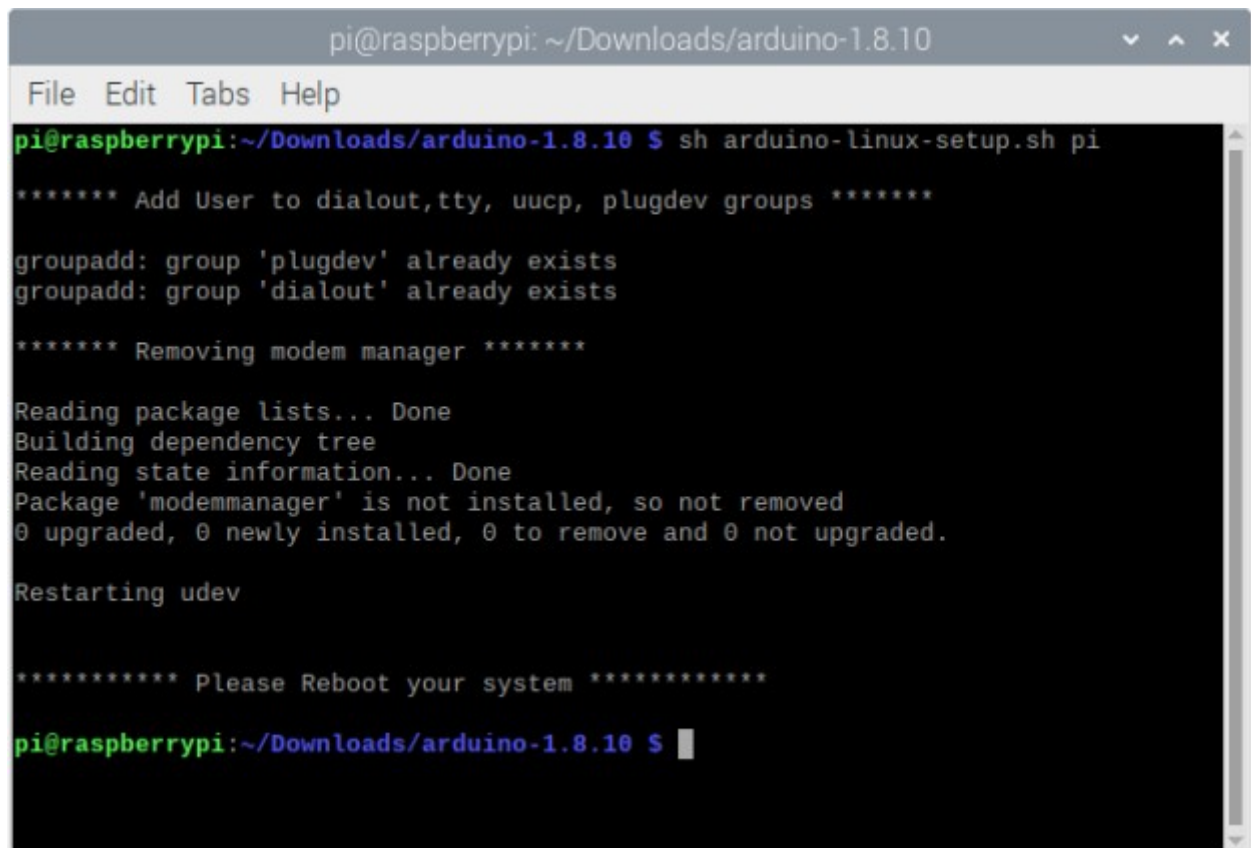


Az-Delivery

Aprire il terminale nella directory in cui sono stati estratti i file di installazione ed eseguire il seguente comando:

sh arduino-linux-setup.sh pi

dove pi è il nome del superutente in Raspbian.



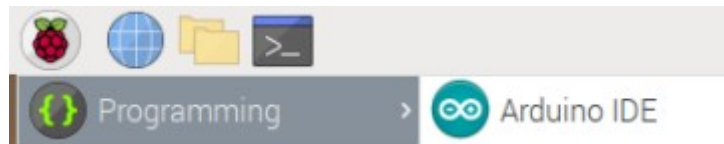
```
pi@raspberrypi: ~/Downloads/arduino-1.8.10
File Edit Tabs Help
pi@raspberrypi:~/Downloads/arduino-1.8.10 $ sh arduino-linux-setup.sh pi
***** Add User to dialout, tty, uucp, plugdev groups *****
groupadd: group 'plugdev' already exists
groupadd: group 'dialout' already exists
***** Removing modem manager *****
Reading package lists... Done
Building dependency tree
Reading state information... Done
Package 'modemmanager' is not installed, so not removed
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
Restarting udev

***** Please Reboot your system *****
pi@raspberrypi:~/Downloads/arduino-1.8.10 $
```

Dopo di che, per installare l'IDE Arduino, eseguite il seguente comando: **sudo sh install.sh**

Az-Delivery

L'Arduino IDE è ora installato. Per eseguire Arduino IDE, aprire l'app:
Menu Applicazioni > Programmazione > Arduino IDE



Prima dei prossimi passi, è necessario installare le applicazioni *pip3* e *git*; aprire il terminale ed eseguire il seguente comando.

sudo apt install python3-pip git -y

La libreria per Python si chiama *nanpy*. Per installarla, aprirete il terminale ed eseguite il seguente comando: **pip3 install nanpy**

```
pi@raspberrypi: ~/Scripts
File Edit Tabs Help
pi@raspberrypi:~/Scripts $ pip3 install nanpy
Looking in indexes: https://pypi.org/simple, https://www.piwheels.org/simple
Collecting nanpy
  Downloading https://www.piwheels.org/simple/nanpy/nanpy-0.9.6-py3-none-any.whl
  (47kB)
    100% |#####| 51kB 209kB/s
Requirement already satisfied: pyserial in /usr/lib/python3/dist-packages (from
nanpy) (3.4)
Installing collected packages: nanpy
Successfully installed nanpy-0.9.6
pi@raspberrypi:~/Scripts $
```

Az-Delivery

Dopo l'installazione della libreria nanpy, scaricate un firmware microcontrollore eseguendo il seguente comando:

```
git clone https://github.com/nanpy/nanpy-firmware.git
```

Cambiare la directory in nanpy-firmware eseguendo il seguente comando: **cd nanpy-firmware**

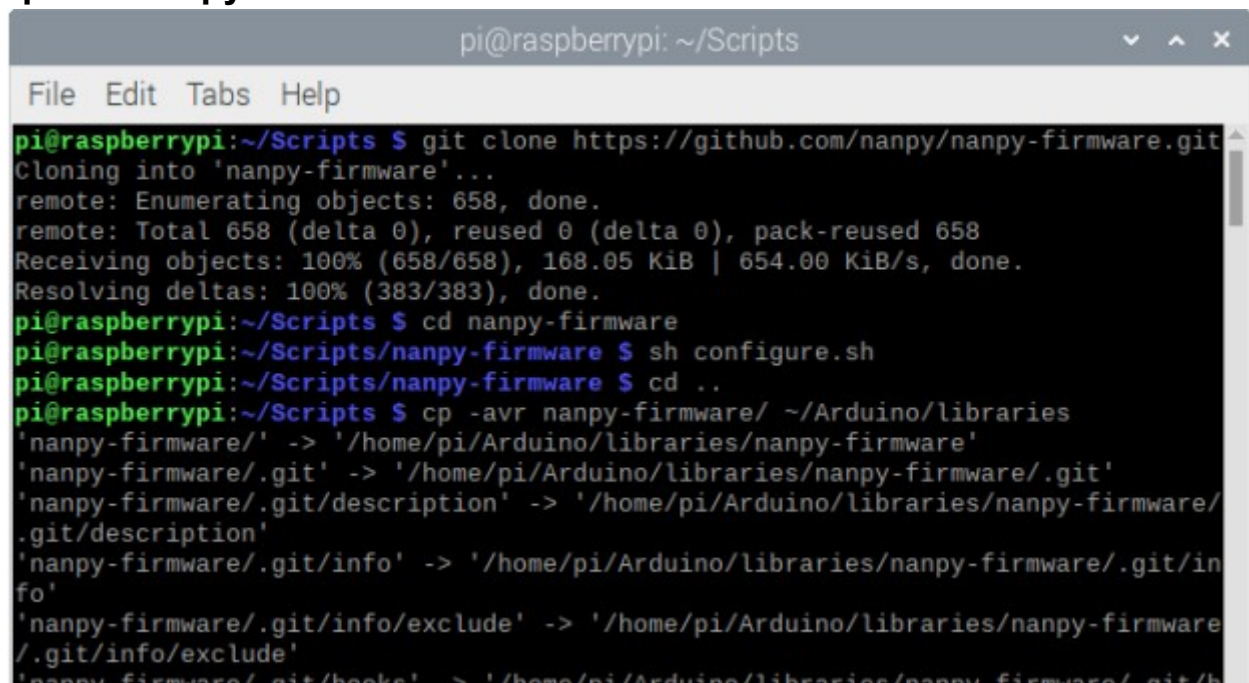
Ed eseguite il seguente comando:

```
sh configure.sh
```

Successivamente, copiare la directory nanpy-firmware in: *Arduino/libraries* directory.

Per farlo, eseguite il seguente comando:

```
cp -avr nanpy-firmware/ ~/Arduino/libraries
```

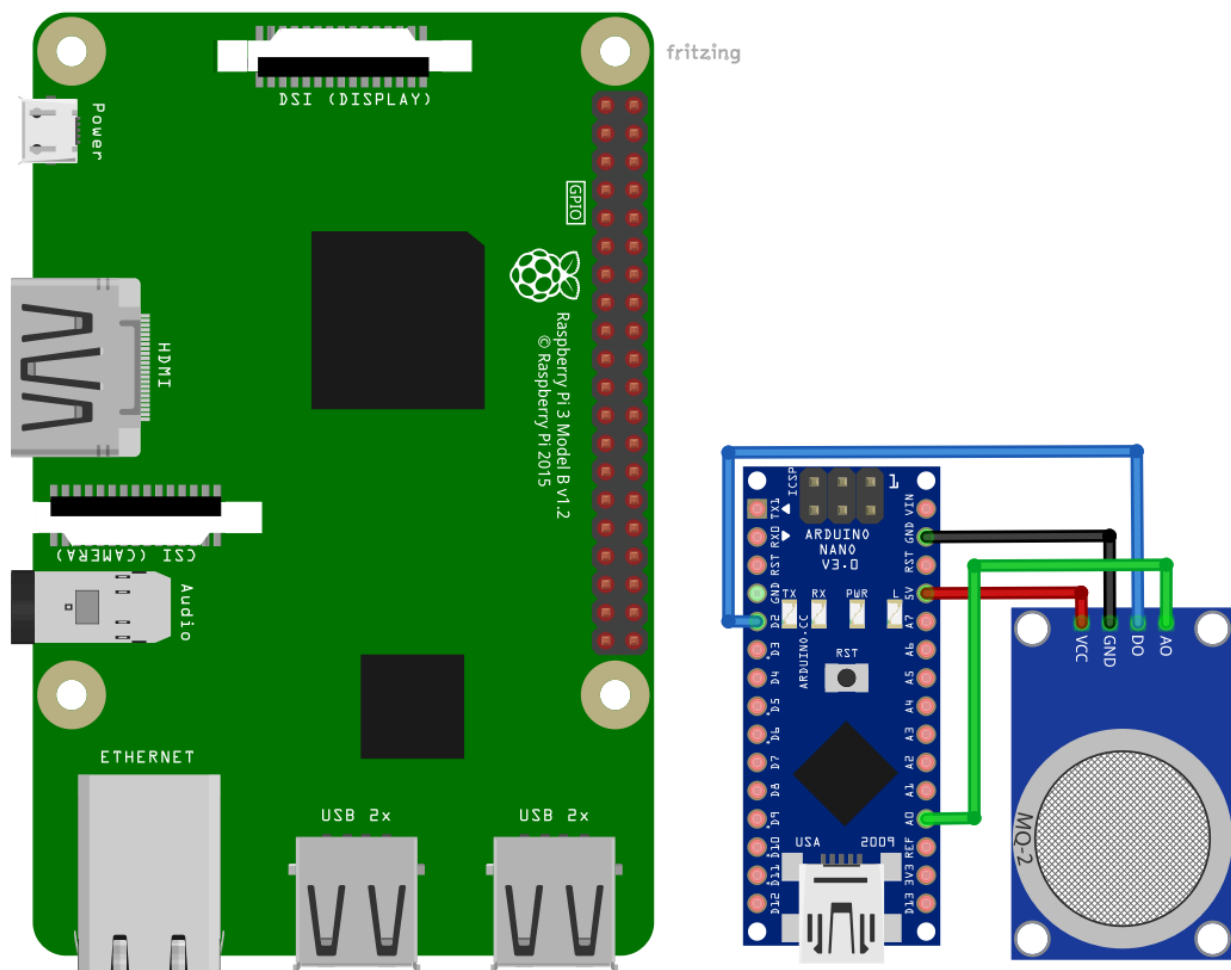


```
pi@raspberrypi: ~/Scripts
File Edit Tabs Help
pi@raspberrypi:~/Scripts $ git clone https://github.com/nanpy/nanpy-firmware.git
Cloning into 'nanpy-firmware'...
remote: Enumerating objects: 658, done.
remote: Total 658 (delta 0), reused 0 (delta 0), pack-reused 658
Receiving objects: 100% (658/658), 168.05 KiB | 654.00 KiB/s, done.
Resolving deltas: 100% (383/383), done.
pi@raspberrypi:~/Scripts $ cd nanpy-firmware
pi@raspberrypi:~/Scripts/nanpy-firmware $ sh configure.sh
pi@raspberrypi:~/Scripts/nanpy-firmware $ cd ..
pi@raspberrypi:~/Scripts $ cp -avr nanpy-firmware/ ~/Arduino/libraries
'nanpy-firmware/' -> '/home/pi/Arduino/libraries/nanpy-firmware'
'nanpy-firmware/.git' -> '/home/pi/Arduino/libraries/nanpy-firmware/.git'
'nanpy-firmware/.git/description' -> '/home/pi/Arduino/libraries/nanpy-firmware/.git/description'
'nanpy-firmware/.git/info' -> '/home/pi/Arduino/libraries/nanpy-firmware/.git/info'
'nanpy-firmware/.git/info/exclude' -> '/home/pi/Arduino/libraries/nanpy-firmware/.git/info/exclude'
'nanpy-firmware/.git/hooks' -> '/home/pi/Arduino/libraries/nanpy-firmware/.git/hooks'
```

Il firmware *nanpy* è ora installato e pronto per essere utilizzato.

Collegamento del modulo con Raspberry Pi

Collegare il modulo con Nano V3.0 come mostrato nella seguente immagine:



Pin modulo	Pin Nano V3.0	Colore filo
VCC	5V	Filo rosso
GND	GND	Filo nero
D0	D2	Filo blu
A0	A0	Filo verde



Poi, collegare il Nano V3.0 via cavo USB al Raspberry Pi e aprire l'IDE di Arduino nel sistema operativo Raspbian. Controllare se Arduino IDE può rilevare la porta USB su cui è collegato il Nano V3.0: *Strumenti > Porta > dev/ttyUSB0*

Poi andate su: *Strumenti > Scheda > {board name}*
e selezionate la scheda ATmega328p.

Poi, per aprire lo sketch per il firmware *nanpy*, andate su: *File > Esempi > nanpy-firmware > Nanpy*

Caricare lo sketch su Nano V3.0. Per verificare se tutto funziona correttamente, è necessario creare il semplice script *Blink*, dove il LED a bordo di ATmega328p lampeggia.

Creare lo script *Blink.py* e aprirlo nell'editor di testo predefinito.

Az-Delivery

Nello script *Blink.py* scrivete le seguenti righe di codice:

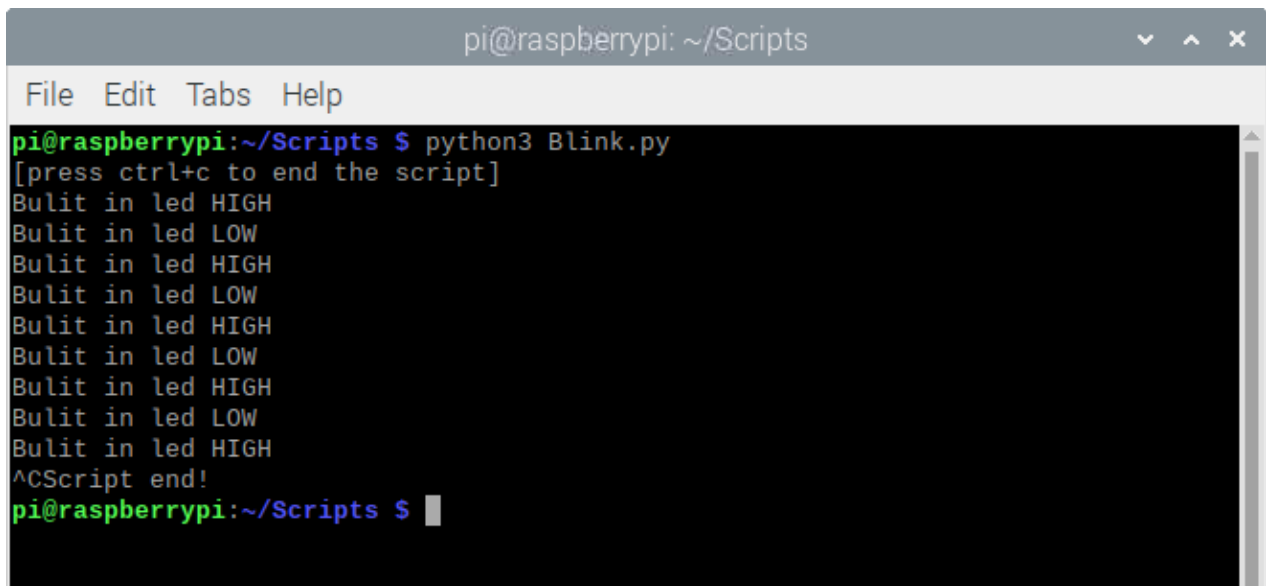
```
from nanpy import (ArduinoApi, SerialManager) from time
import sleep
ledPin = 13
try:
    connection1 = SerialManager()
    a = ArduinoApi(connection=connection1) except:
    print('Failed to connect to the Arduino') print('[Press CTRL
+ C to end the script!']') a.pinMode(ledPin, a.OUTPUT) #
Setup Arduino try:
    while True:
        a.digitalWrite(ledPin, a.HIGH)
        print('Bulit in led HIGH')
        sleep(1)
        a.digitalWrite(ledPin, a.LOW)
        print('Bulit in led LOW')
        sleep(1)

except KeyboardInterrupt:
    print('\nScript end!')
    a.digitalWrite(ledPin, a.LOW)
```

Az-Delivery

Salvare lo script con il nome *Blink.py*. Per eseguire lo script, aprire il terminale nella directory in cui è stato salvato lo script ed eseguire il seguente comando: *python3 Blink.py*

Il risultato dovrebbe assomigliare all'immagine seguente:



```
pi@raspberrypi: ~/Scripts
File Edit Tabs Help
pi@raspberrypi:~/Scripts $ python3 Blink.py
[press ctrl+c to end the script]
Bulit in led HIGH
Bulit in led LOW
Bulit in led HIGH
Bulit in led LOW
Bulit in led HIGH
Bulit in led LOW
Bulit in led HIGH
Bulit in led LOW
Bulit in led HIGH
Bulit in led LOW
Bulit in led HIGH
^CScript end!
pi@raspberrypi:~/Scripts $
```

Per fermare lo script premere *CTRL + C* sulla tastiera.

Il LED collegato al pin digitale 13 di Nano V3.0 dovrebbe iniziare a lampeggiare ogni secondo.

Az-Delivery

Lo script inizia con l'importazione di due librerie, le funzioni della libreria *nanpy* e *time*.

Quindi, viene creata la variabile chiamata *ledPin* e inizializzata con il numero *13*.

Il numero *13* rappresenta il numero del pin digitale su cui è collegato il LED (LED a bordo di Nano V3.0).

Dopo di che, il blocco di codice *try-except* viene utilizzato per provare a connettersi a Nano V3.0. Se la connessione non va a buon fine, il messaggio: *Failed to connect to the microcontrollore* viene visualizzato nel terminale.

Se la connessione ha successo, l'oggetto di comunicazione chiamato "a" viene creato e inizializzato. L'oggetto "a" rappresenta la scheda Nano V3.0. Qualsiasi funzione utilizzata nell'Arduino IDE può essere utilizzata con l'oggetto "a", come si può vedere nel codice.

Con la seguente riga di codice, si configura la modalità pin del pin digitale *13*:

```
a.pinMode(ledPin, a.OUTPUT)
```

Poi, nel blocco del loop infinito (*while True:*) viene usata la funzione *digitalWrite()* per configurare lo stato del pin digitale *13* (stato HIGH o LOW). Con la funzione *digitalWrite()* il LED collegato al pin *13* può essere **ACCESO** o **SPENTO**.



Nel blocco loop indefinito, il LED viene prima *ACCESO* per un secondo, e poi *SPENTO* per un secondo. Questo si chiama *blinking the LED* . L'intervallo di tempo di un singolo lampeggio può essere modificato nella seguente riga di codice: *sleep(1)*

Dove il numero 1 rappresenta il numero di secondi per la durata dell'intervallo di tempo.

Per fermare il loop infinito premere CTRL + C sulla tastiera. Questo è chiamato interrupt da tastiera, che è impostato nel blocco *except (except KeyboardInterrupt)*. Nel blocco expect il LED a bordo viene SPENTO.



Script Python per il modulo MQ-2

```
from nanpy import (ArduinoApi, SerialManager)
import time
```

```
try:
    connection_1 = SerialManager()
    a = ArduinoApi(connection=connection_1)
except:
    print('Failed to connect to the Arduino')
DIGITAL_PIN = 2
ANALOG_PIN = 0
time.sleep(2)
print('Sensor is warming up...')
print('[Press CTRL+C to end the script]')
time.sleep(5) # Sensor warming up...
```

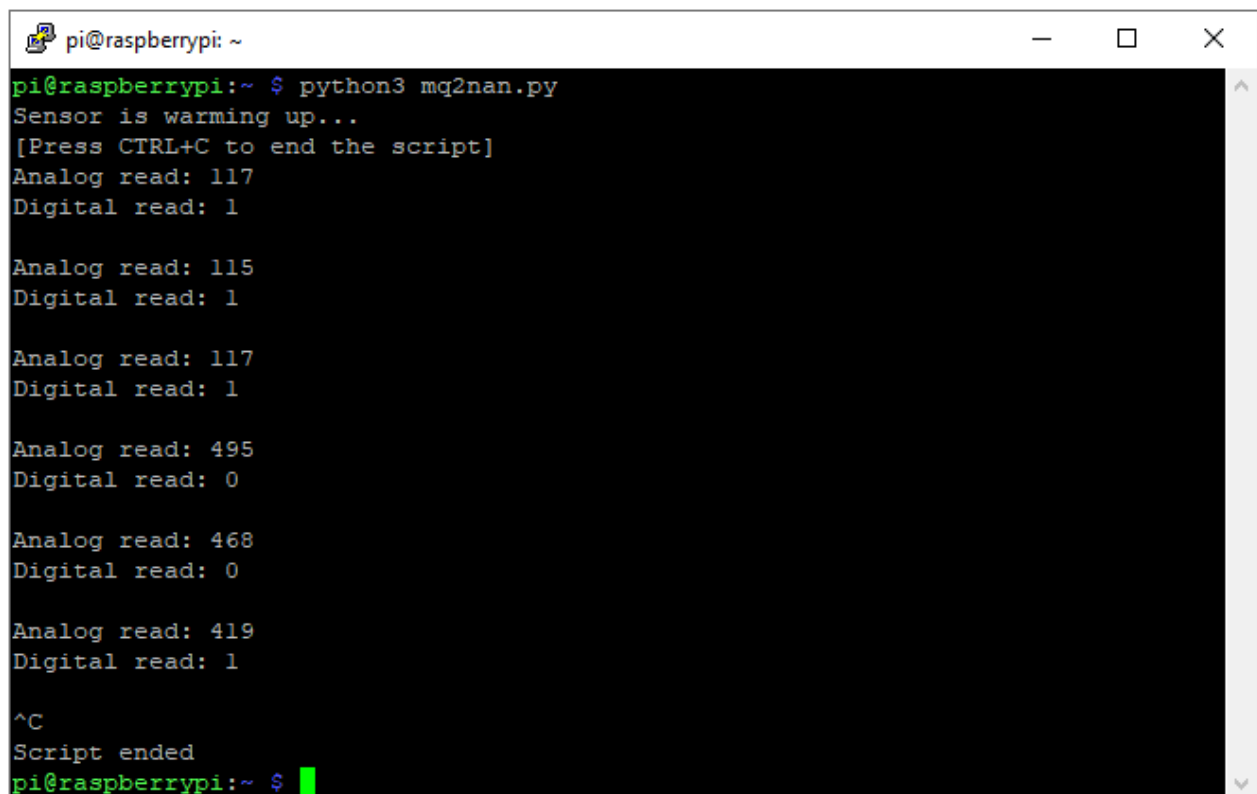
```
try:
    while True:
        analogReading = a.analogRead(ANALOG_PIN)
        digitalReading = a.digitalRead(DIGITAL_PIN)
        print("Analog read: {}\nDigital read: {}".format(analogReading, digitalReading))
        time.sleep(2)
except KeyboardInterrupt:
    print("\nScript end!")
```

Az-Delivery

Salvare lo script con il nome *mq2nan.py*. Per eseguire lo script, aprire il terminale nella directory in cui è stato salvato lo script ed eseguire il seguente comando:

python3 mq2nan.py

Il risultato dovrebbe assomigliare all'immagine seguente:



```
pi@raspberrypi: ~  
pi@raspberrypi:~ $ python3 mq2nan.py  
Sensor is warming up...  
[Press CTRL+C to end the script]  
Analog read: 117  
Digital read: 1  
  
Analog read: 115  
Digital read: 1  
  
Analog read: 117  
Digital read: 1  
  
Analog read: 495  
Digital read: 0  
  
Analog read: 468  
Digital read: 0  
  
Analog read: 419  
Digital read: 1  
  
^C  
Script ended  
pi@raspberrypi:~ $
```

Per fermare lo script premere *CTRL + C* sulla tastiera.



E ora è tempo di imparare e di creare dei Progetti da solo. Lo puoi fare con l'aiuto di molti script di esempio e altri tutorial, che puoi trovare in internet.

Se stai cercando la microelettronica e gli accessori di alta qualità, AZ-Delivery Vertriebs GmbH è l'azienda giusta dove potrai trovarli. Ti forniremo numerosi esempi di applicazioni, guide di installazione complete, e-book, librerie e l'assistenza dei nostri esperti tecnici.

<https://az-delivery.de>

Buon divertimento!

Impressum

<https://az-delivery.de/pages/about-us>