

AZ-Delivery

Benvenuto!

Grazie per aver acquistato il nostro *Lettore RFID RC522 con Chip RFID e Scheda RFID AZ-Delivery*. Nelle pagine seguenti, ti illustreremo come utilizzare e configurare questo pratico dispositivo.

Buon divertimento!



Az-Delivery

RFID significa identificazione a radiofrequenza. RFID utilizza campi elettromagnetici per trasferire dati su brevi distanze. I tag RFID sono utilizzati in molti settori, ad esempio un tag RFID attaccato a un'automobile durante la produzione può essere utilizzato per tracciare i suoi progressi attraverso la catena di montaggio, i prodotti farmaceutici con tag RFID possono essere monitorati attraverso i magazzini, impiantando microchip RFID nel bestiame e negli animali domestici consente un'identificazione positiva di animali, ecc

Un sistema RFID è composto da due parti: un tag e un lettore. Le etichette RFID sono integrati con un trasmettitore e un ricevitore. Il componente RFID sui tag ha due parti: un microchip che memorizza ed elabora le informazioni e un'antenna per ricevere e trasmettere un segnale. Il tag contiene il numero di serie specifico per un oggetto specifico e non è necessario che si trovi nel campo visivo diretto del lettore per essere rintracciato.

Per leggere le informazioni codificate su un tag, un trasmettitore-ricevitore radio chiamato lettore emette un segnale verso il tag usando un'antenna. Il tag risponde con le informazioni scritte nella sua memoria. Il lettore trasmetterà quindi i risultati letti a un microcontrollore.

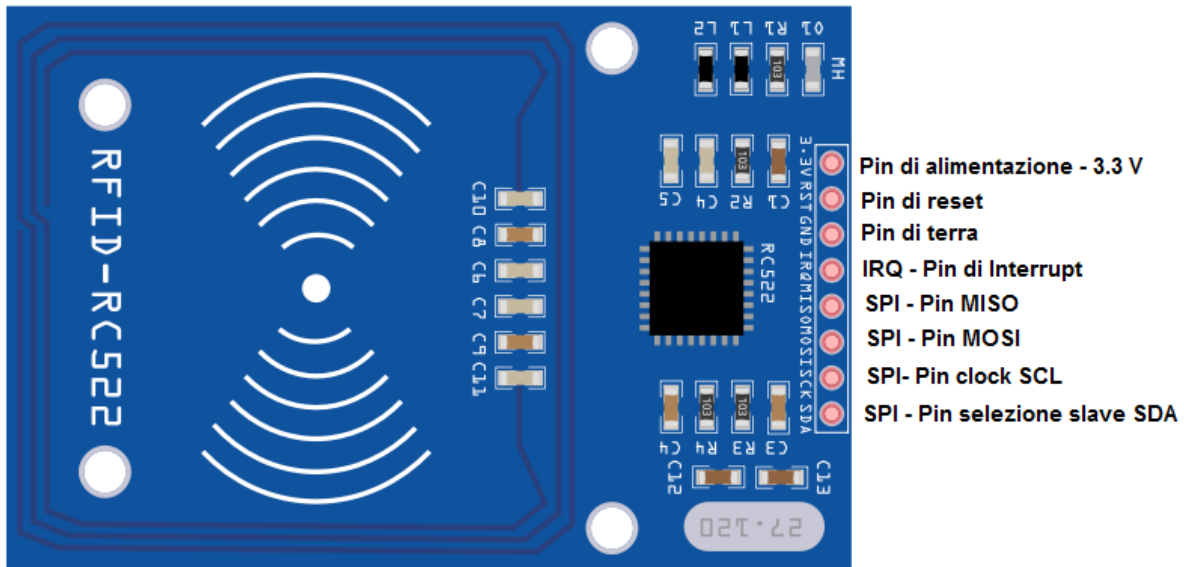


Specifiche:

- » Chip: MFRC522
- » Frequenza operativa: 13.56MHz
- » Tensione di alimentazione: 3.3V
- » Corrente: 13 - 26mA
- » Range di Lettura: Ca. 30mm
- » Comunicazione: Interfaccia SPI
- » Velocità Massima di Trasferimento Dati: 10Mbit/s
- » Dimensioni: 40 x 60mm [1.6 x 2.4in]

Il lettore è un dispositivo near-field, non una vera radio. Il modo in cui il lettore funziona non è inviando e ricevendo segnali radio, ma rilevando piccoli cambiamenti nell'impedenza del circuito dell'antenna sintonizzata causati dal tag RFID che modula il suo circuito sintonizzato. Se si sposta il tag RFID fuori dal campo vicino all'antenna non c'è alcun effetto da misurare. Il lettore non è in realtà un ricevitore, ma una versione amplificata delle fluttuazioni della tensione di uscita media nel circuito sintonizzato.

La piedinatura del lettore



La tensione di alimentazione è di 3,3 V! Non collegare 5 V a questo pin, altrimenti potresti distruggere il dispositivo!

Il lettore utilizza l'interfaccia SPI per comunicare con un microcontrollore.

Il pin IRQ di interrupt è sempre in stato *HIGH* e quando si verifica l'evento di interruzione, cambia il suo stato in *LOW* per un breve periodo di tempo. L'evento di interruzione si verifica quando viene rilevata un tag RFID nel campo vicino del lettore.

Az-Delivery

Come configurare Arduino IDE

Se non hai già installato Arduino IDE, ecco come farlo. Vai al link: <https://www.arduino.cc/en/Main/Software> e scarica il file di installazione per la tua piattaforma del sistema operativo.

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 containing a minus and a plus sign. To its right, the text reads: **ARDUINO 1.8.9**, followed by a description of the IDE as open-source software written in Java, based on Processing, and compatible with any Arduino board. It refers to the 'Getting Started' page for installation instructions. On the right side, there is a teal sidebar with links for different operating systems: Windows (installer and ZIP file), Windows app (with a 'Get' button), Mac OS X (10.8 Mountain Lion or newer), and Linux (32 bits, 64 bits, ARM 32 bits, and ARM 64 bits). At the bottom of the sidebar are links for Release Notes, Source Code, and Checksums (sha512).

Per Windows, fai doppio clic sul file ".exe" scaricato e segui le istruzioni nella finestra di installazione.

Az-Delivery

Per Linux, scarica il file con estensione `".tar.xz"`, che dovrai estrarre. Quando lo estrai, vai alla directory estratta e apri il terminale in quella directory. È necessario eseguire due script `".sh"`, il primo chiamato `"arduino-linux-setup.sh"` e in secondo chiamato `"install.sh"`.

Per eseguire il primo script nel terminale, eseguire il comando seguente:

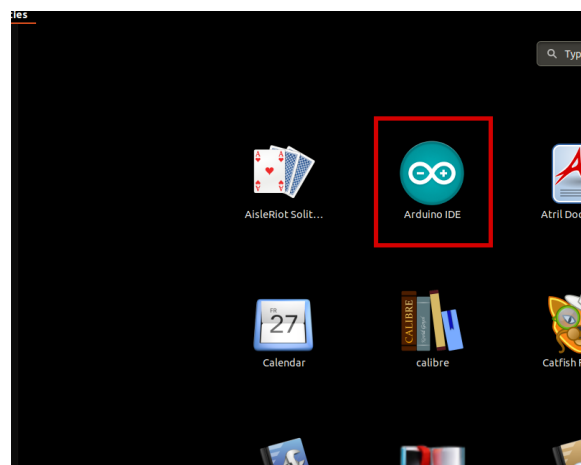
`sh arduino-linux-setup.sh user_name`

`user_name` - è il nome di super utente nel sistema operativo Linux. Successivamente, ti verrà richiesto di fornire la password per il super utente. Attendi qualche minuto affinché lo script completi tutto.

Dopo l'installazione del primo script, esegui il secondo script `"install.sh"`. Nel terminale, eseguire il comando seguente:

`sh install.sh`

Dopo l'installazione di questi script, vai su *All Apps* per trovare l'*Arduino IDE* installata.

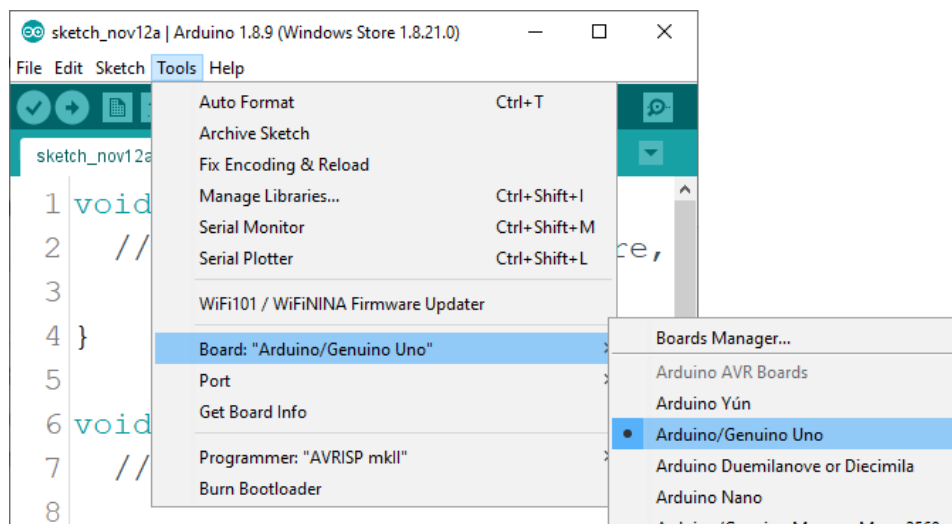


Az-Delivery

La prossima cosa è verificare se il tuo PC è in grado di rilevare la scheda Atmega328p. Apri l'*Arduino IDE* appena installato e vai a:

Strumenti > Scheda > {il nome della tua scheda qui}

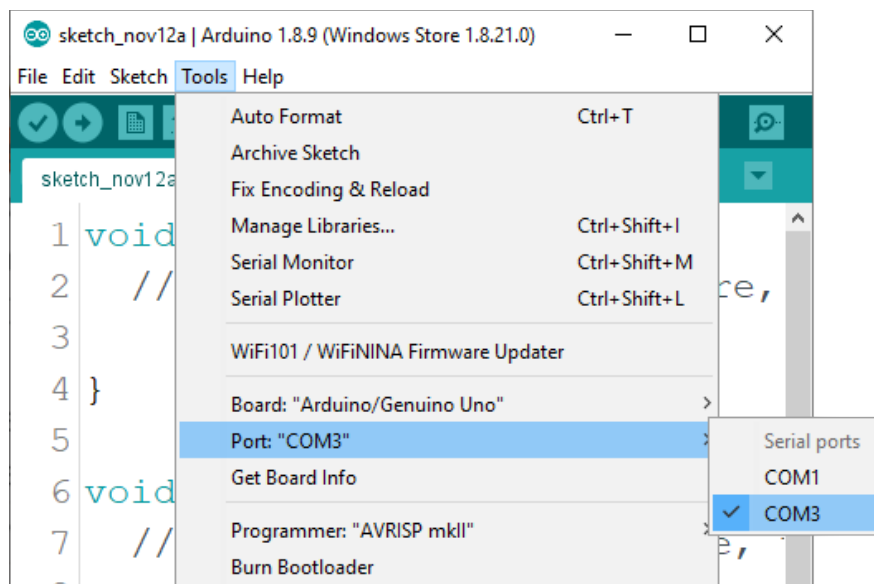
{il nome della tua scheda qui} dovrebbe essere *Arduino/Genuino Uno*, come puoi vedere nell'immagine qui sotto:



Az-Delivery

Successivamente è necessario selezionare la porta su cui è connessa la scheda Atmega328p. Vai su: *Strumenti > Porta > {il nome porta va qui}*

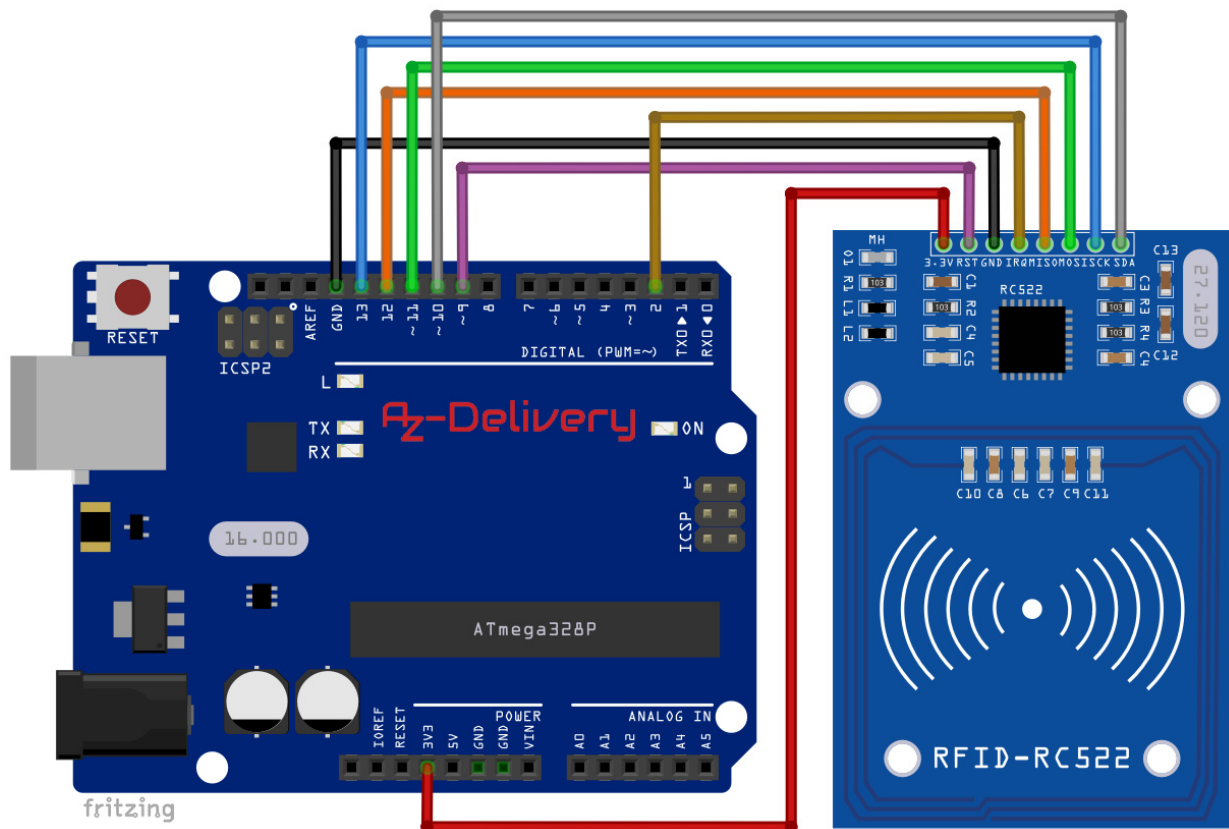
Se hai collegato la scheda Atmega328p sulla porta USB, dovrebbero esserci diversi nomi di porta. Poichéé stiamo usando *Arduino IDE* su *Windows*, i nomi delle porte sono come nell'immagine qui sotto:



Per gli utenti Linux, il nome della porta va su *"/dev/ttyUSBx"* ad esempio, dove *"x"* rappresenta un numero intero specifico compreso tra *0* e *9*, ad esempio.

Collegamento del lettore con Atmega328p

Collegare il lettore con Atmega328p come mostrato nello schema di collegamento seguente:



Pin modulo > Pin Mc

3.3V > **3.3V !**

RST > D9

GND > GND

IRQ > D2

MISO > D12

MOSI > D11

SCK > D13

SDA (SS) > D10

Filo rosso

Filo viola

Filo nero

Filo ocra

Filo arancio

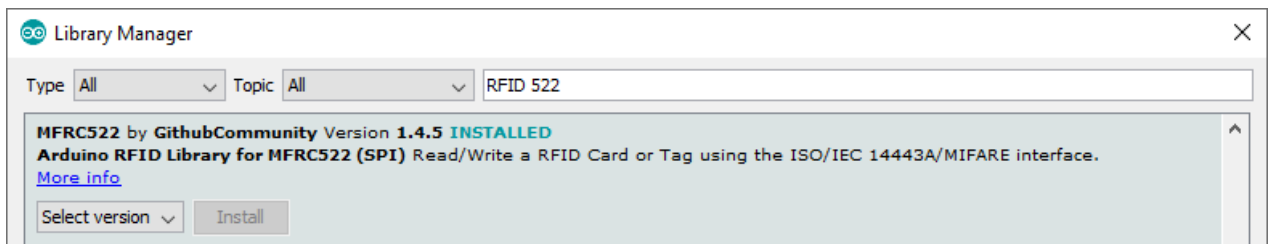
Filo verde

Filo blu

Filo grigio

La libreria per Arduino IDE

Per utilizzare il modulo con Atmega328p, si consiglia di scaricare e installare la libreria per esso. Apri Arduino IDE e vai su: *Strumenti > Gestione Librerie*. Si aprirà una nuova finestra, digita “*RFID 522*” nella casella di ricerca, e installa la libreria chiamata “*MFRC522*” realizzata da “*GithubCommunity*”, come mostrato nell'immagine qui sotto:



Con la libreria vengono forniti molti sketch di esempio. Useremo uno sketch di esempio, chiamato “*MinimalInterrupt*”. Per aprirlo, vai su:

File > Esempi > MFRC522 > MinimalInterrupt



Esempio di Sketch:

```
#include <SPI.h>
#include <MFRC522.h>
#define RST_PIN 9 // Configurable, see typical pin layout above
#define SS_PIN 10 // Configurable, see typical pin layout above
#define IRQ_PIN 2 // Configurable, depends on hardware
MFRC522 mfrc522(SS_PIN, RST_PIN); // Create MFRC522 instance
MFRC522::MIFARE_Key key;
volatile bool bNewInt = false;
byte regVal = 0x7F;

void setup() {
  Serial.begin(9600); // Initialize serial communications with the PC
  SPI.begin(); // Init SPI bus
  mfrc522.PCD_Init(); // Init MFRC522 card

  // Read and printout the MFRC522 version (valid values 0x91 & 0x92)
  Serial.print(F("Ver: 0x"));
  byte readReg = mfrc522.PCD_ReadRegister(mfrc522.VersionReg);
  Serial.println(readReg, HEX);

  pinMode(IRQ_PIN, INPUT_PULLUP); // Setup the IRQ pin
  // Allow the irq to be propagated to the IRQ pin
  regVal = 0xA0; // Rx IRQ
  mfrc522.PCD_WriteRegister(mfrc522.ComIEnReg, regVal);

  bNewInt = false; // Interrupt flag
  // Activate the interrupt
  attachInterrupt(digitalPinToInterrupt(IRQ_PIN), readCard, FALLING);

  do { // Clear a spurious interrupt at start
    ;
  } while (!bNewInt);
  bNewInt = false;
  Serial.println(F("End setup"));
}
```

Az-Delivery

```
void loop() {
  if (bNewInt) { // New read interrupt
    Serial.print(F("Interrupt. "));
    mfrc522.PICC_ReadCardSerial(); // Read the tag data
    // Show some details of the PICC
    Serial.print(F("Card UID:"));
    dump_byte_array(mfrc522.uid.uidByte, mfrc522.uid.size);
    Serial.println();
    clearInt(mfrc522);
    mfrc522.PICC_HaltA();
    bNewInt = false;
  }
  activateRec(mfrc522);
  delay(100);
}

// MFRC522 interrupt serving routine
void readCard() {
  bNewInt = true;
}

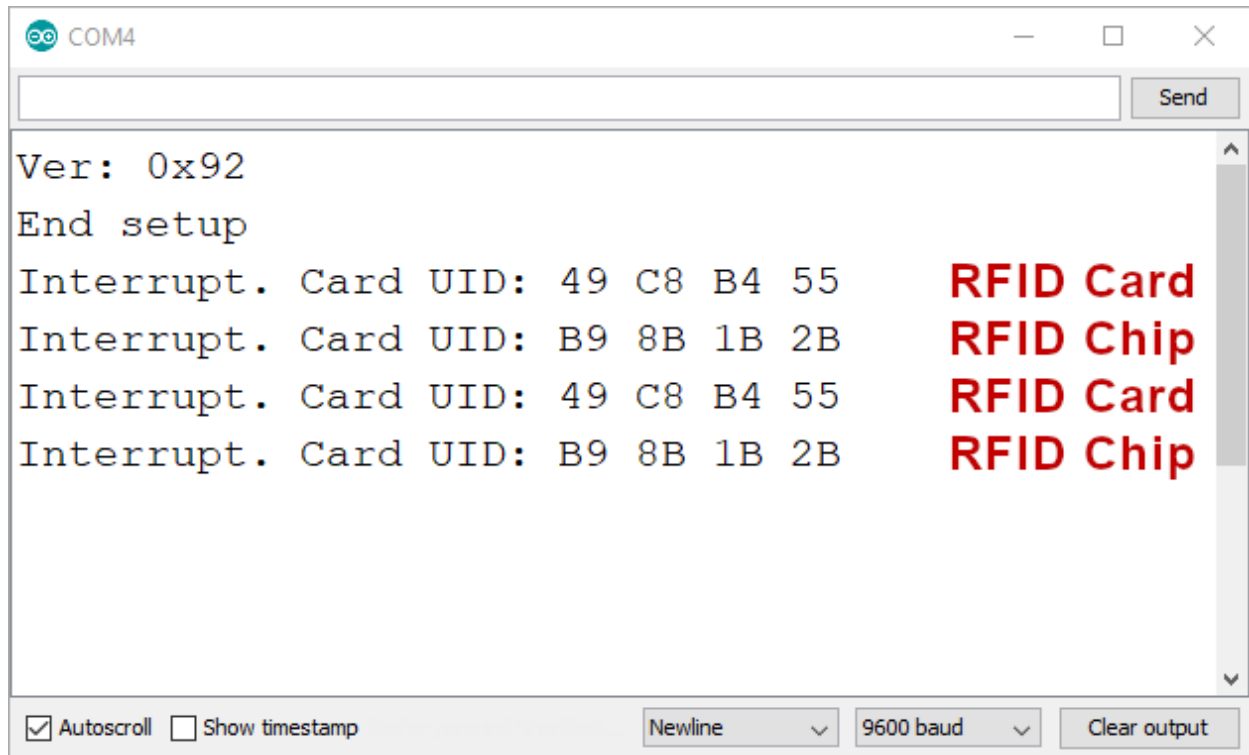
// Helper routine to dump a byte array as hex values to Serial
void dump_byte_array(byte *buffer, byte bufferSize) {
  for (byte i = 0; i < bufferSize; i++) {
    Serial.print(buffer[i] < 0x10 ? " 0" : " ");
    Serial.print(buffer[i], HEX);
  }
}

// The function sending to the MFRC522 the needed commands to activate the reception
void activateRec(MFRC522 mfrc522) {
  mfrc522.PCD_WriteRegister(mfrc522.FIFODataReg, mfrc522.PICC_CMD_REQA);
  mfrc522.PCD_WriteRegister(mfrc522.CommandReg, mfrc522.PCD_Transceive);
  mfrc522.PCD_WriteRegister(mfrc522.BitFramingReg, 0x87);
}

// The function to clear the pending interrupt bits after ISR
void clearInt(MFRC522 mfrc522) {
  mfrc522.PCD_WriteRegister(mfrc522.ComIrqReg, 0x7F);
}
```

Az-Delivery

Quando si carica lo sketch su Atmega328p, aprire il Monitor Seriale, (*Strumenti > Monitor Seriale*). L'output dovrebbe assomigliare all'output sull'immagine seguente:



Quando si avvicina la scheda o il chip RFID al lettore, si verifica un'interruzione e il numero UID della scheda/chip viene inviato al Monitor Seriale.

All'inizio dello sketch vengono importate due librerie, *SPI* per l'interfaccia SPI e *MFRC522* per il lettore RFID. Quindi vengono create tre macro, una per il numero di pin di reset, la seconda per il numero di pin di selezione slave e l'ultima per il numero di pin di interruzione.

Az-Delivery

Dopo l'oggetto "*mfrc522*", che rappresenta il lettore nel software, viene creata la variabile *key*. La variabile *key* viene utilizzata per memorizzare il numero UID della scheda specifica.

Viene poi creata la variabile *bNewInt*. Quando si verifica l'interrupt, lo stato della variabile *bNewInt* cambia in *true*, che indica che si è verificato l'interrupt. Quindi nel codice controlliamo lo stato della variabile *bNewInt* e stampiamo l'output corrispondente sul Monitor Seriale (maggiori dettagli più avanti nel testo).

Creiamo poi la variabile *regVal*, che viene utilizzato per impostare i registri del chip nel lettore RFID.

Nella funzione *setup()*, per prima cosa impostiamo l'Interfaccia Seriale con baud rate di *9600*. Successivamente, configuriamo l'interfaccia SPI e inizializziamo l'oggetto *mfrc522*.

Successivamente emettiamo la versione del lettore MFRC522 sul Monitor Seriale. I numeri validi sono *0x91* and *0x92*.

Quindi impostiamo la modalità pin del pin di interrupt su *INPUT* con resistenza *PULL UP* interna. Successivamente, alleghiamo la routine di interrupt con la seguente riga del codice:

```
attachInterrupt(digitalPinToInterrupt(IRQ_PIN), readCard, FALLING)
```

Dove il primo argomento, quella è una funzione integrata:

digitalPinToInterrupt()

Az-Delivery

che configura l'*IRQ_PIN* (pin digitale I/O 2), per ascoltare l'evento di interrupt. Il secondo argomento è la funzione che verrà eseguita quando si verifica un interrupt. E il terzo argomento è il limite del segnale digitale, che rappresenta l'evento di interrupt stesso.

Ai fini di questo esempio stiamo usando il limite *FALLING* del segnale digitale, che arriva su *IRQ_PIN*, come evento di interrupt. Il valore di questo argomento può anche essere *RISING* o *BOTH*, ma non lo stiamo usando nel nostro esempio.

Alla fine della funzione *setup()* puliamo le letture degli interrupt sbagliate con le seguenti righe di codice:

```
do { ; } while (!bNewInt);  
bNewInt = false;
```

Nella funzione *loop()* c'è una dichiarazione *if*, dove controlliamo lo stato della variabile *bNewInt*. Se lo stato *bNewInt* è *true*, allora si è verificato l'interrupt, e stampiamo l'output nel Monitor Seriale. L'output contiene il messaggio "*Interrupt. Card UID: **number***", dove *number* è il numero UID della scheda RFID che ha causato l'interrupt. Alla fine della dichiarazione *if*, configuriamo la variabile *bNewInt* su *false*.

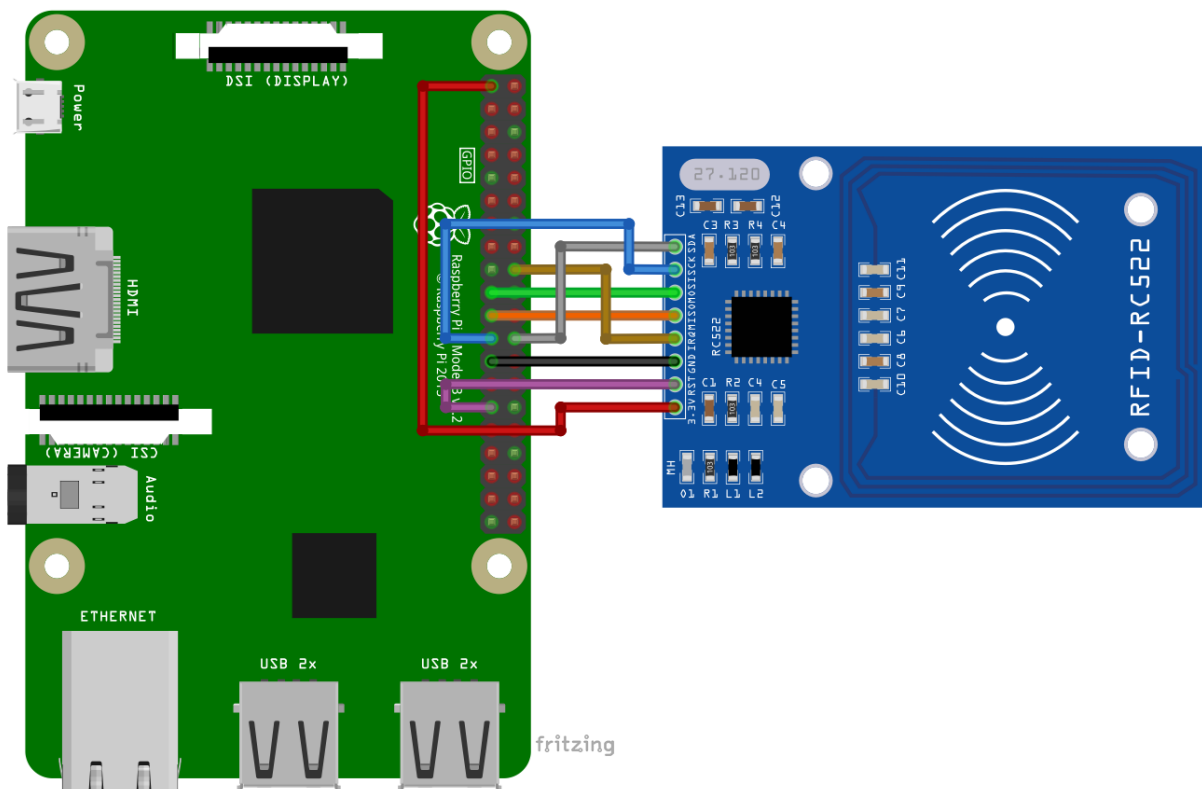
Alla fine della funzione *loop()*, configuriamo i registri del chip del lettore nuovamente, e aspettiamo per 100 millisecondi.

Dopo la funzione `loop()`, creiamo la funzione `readCard()`, che viene eseguita quando si verifica l'interrupt. Nella funzione `readCard()` configuriamo solo lo stato della variabile `bNewInt` in `true`.

Successivamente, creiamo tre funzioni che vengono utilizzate per impostare i registri del chip del lettore e per comunicare con il lettore.

Collegamento del lettore con Raspberry Pi

Collega il lettore a Raspberry Pi come mostrato nello schema di collegamento seguente:



Pin modulo > Pin Raspberry Pi

SDA (SS) > GPIO8 [pin 24]

SCK > GPIO11 [pin 23]

Filo grigio

Filo blu

Az-Delivery

MOSI	>	GPIO10	[pin 19]	Filo verde
MISO	>	GPIO9	[pin 21]	Filo arancio
IRQ	>	GPIO24	[pin 18]	Filo ocra
GND	>	GND	[pin 25]	Filo nero
RST	>	GPIO25	[pin 22]	Filo viola
3.3V	>	3V3	[pin 1]	Filo rosso



La libreria per Python

Per utilizzare il lettore con Raspberry Pi, si consiglia di scaricare una libreria per Python. La libreria che useremo si chiama “*pi-rc522*”. Per installare questa libreria, aprire un terminale ed eseguire i seguenti comandi, uno per uno:

```
sudo apt-get update
```

```
sudo apt-get upgrade
```

```
sudo pip3 install pi-rc522
```

Ora siamo pronti per iniziare a creare lo script Python.

Az-Delivery

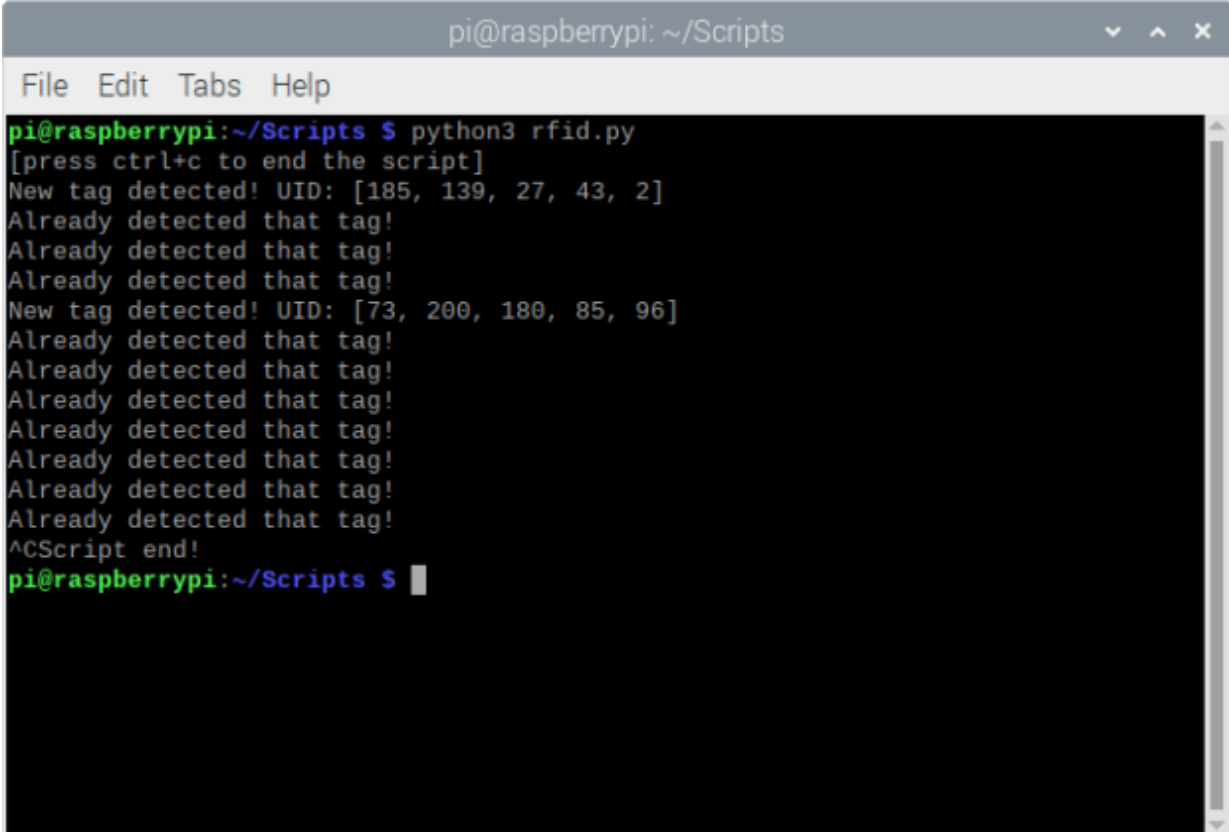
Script Python :

```
from pirc522 import RFID
from time import sleep
reader = RFID()
keys = list()
key_read = True
print('[press ctrl+c to end the script]')
try:
    while True:
        reader.wait_for_tag()
        error, tag_type = reader.request()
        if not error:
            error, uid = reader.anticoll()
            if not error:
                if len(keys) > 0:
                    for key in keys:
                        if key == uid:
                            key_read = False
                            break
                        else:
                            key_read = True
                else:
                    key_read = True
            if key_read:
                keys.append(uid)
                print('New tag detected! UID: {}'.format(uid))
                reader.stop_crypto() # always call this when done working
                key_read = False
            else:
                print('Already detected that tag!')
        sleep(0.1)
# Scavenging work after the end of the program
except KeyboardInterrupt:
    print('Script end!')
finally:
    reader.cleanup() # Calls GPIO cleanup
```

Az-Delivery

Salva lo script con il nome *"rfid.py"* nella directory predefinita per gli script. Per eseguire lo script, aprire il terminale nella directory in cui è stato salvato lo script ed eseguire il comando seguente:
python3 rfid.py

L'output dovrebbe assomigliare all'output sull'immagine seguente:



```
pi@raspberrypi: ~/Scripts
File Edit Tabs Help
pi@raspberrypi:~/Scripts $ python3 rfid.py
[press ctrl+c to end the script]
New tag detected! UID: [185, 139, 27, 43, 2]
Already detected that tag!
Already detected that tag!
Already detected that tag!
New tag detected! UID: [73, 200, 180, 85, 96]
Already detected that tag!
Already detected that tag!
Already detected that tag!
Already detected that tag!
Already detected that tag!
Already detected that tag!
^CScript end!
pi@raspberrypi:~/Scripts $
```

Per terminare lo script premere *"CTRL + C"*.

Az-Delivery

All'inizio dello script importiamo due librerie, una chiamata *pir522* che abbiamo appena installato e l'altra è per l'ora.

Quindi creiamo l'oggetto lettore, usando la seguente riga del codice:

```
reader = RFID()
```

Puoi collegare il pin *SDA* del lettore a *CE1* (GPIO7 [pin 26]) al posto di *CE0* (GPIO8 [pin 24]). Quando lo fai devi creare un oggetto lettore con la seguente riga del codice:

```
reader = RFID(bus=0, device=1)
```

Inoltre, puoi collegare il pin *RST* a qualsiasi altro pin GPIO libero. Se lo fai devi creare un oggetto lettore con la seguente riga del codice:

```
reader = RFID(pin_rst=number)
```

Dove *number* è il numero di pin della scheda Raspberry Pi (non numero GPIO, ma numero pin).

Inoltre, puoi collegare il pin *IRQ* a qualsiasi altro pin GPIO libero. Se lo fai devi creare un oggetto lettore con la seguente riga del codice: `reader = RFID(pin_irq=number)`

Dove *number* isè il numero pin della scheda Raspberry Pi (non il numero GPIO, ma il numero pin).

Dopo di che creiamo in elenco *keys* dove memorizziamo le nuove chiavi UID. Inoltre creiamo una variabile booleana chiamata *key_read* utilizzata per verificare se la chiave è già nell'elenco delle *keys*.

Az-Delivery

Infine creiamo il blocco *try-except-finally*. Lo facciamo per rilevare l'interrupt da tastiera. L'interrupt da tastiera si verifica quando si preme *CTRL + C* sulla tastiera. Il blocco *finally* part of *try-except-finally* viene usato per pulire i pin GPIO di qualsiasi interfaccia usata o di modalità pin deginire dopo l'interrupt da tastiera.

Nella parte *try* del blocco *try-except-finally* creiamo un loop infinito (*while True:*). In questo loop prima aspettiamo che venga rilevato il tag. Quando viene rilevato un tag, si verificherà un interrupt su *IRQ_PIN*. Quando si verifica l'interruzione, possiamo leggere i dati dei tag.

La funzione *reader.request()* restituisce una tupla di due elementi. Il primo elemento è un valore booleano che rappresenta l'errore (*True* c'è un errore, *False* non c'è un errore), e il secondo elemento rappresenta il tipo di tag (che non usiamo nello script).

Quindi controlliamo se c'è un errore. Se non ci sono errori, si procede alla lettura dell'UID del tag.

La funzione *reader.anticoll()* restituisce una tupla di due elementi. Il primo elemento è un valore booleano che rappresenta l'errore (*True* c'è un errore, *False* non c'è un errore), e il secondo elemento è il numero UID del tag.

Az-Delivery

Quindi, controlliamo ancora se c'è un errore durante la lettura dell'UID. Se non ci sono errori, si procede a verificare se il numero UID è già presente nell'elenco delle *keys*. Se il numero UID non è nell'elenco delle *keys*, aggiungiamo un nuovo numero e stampiamo il messaggio “*New tag detected! UID: **number***”, dove *number* è il numero UID del tag. Se il numero UID del tag è nell'elenco delle chiavi, stampiamo il messaggio “*Already detected that tag!*”.

Quando abbiamo finito con la lettura dei dati inviati dal lettore, dobbiamo eseguire la seguente funzione:

```
reader.stop_crypto()
```

Alla fine del blocco loop infinito chiamiamo *sleep(0.1)*, oppure un intervallo di tempo di attesa di *100* millisecondi, un intervallo di tempo di attesa tra due loop del blocco di loop infinito.

Ce l'hai fatta, ora puoi usare il tuo modulo per i tuoi progetti.



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 dei microelettronica e 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>