82. Leggere tag RFID

Con Arduino è possibile leggere tag RFID e smartcard utilizzando degli appositi lettori. Un tag RFID può essere usato come chiave elettronica o dispositivo di identificazione perché al suo interno ha codificato un ID univoco.

In dettaglio

Un interessantissimo tipo di dispositivi che potete utilizzare con Arduino sono i lettori di tag RFID (Radio-Frequency IDentification - identificazione a radio frequenza). Un lettore di questo tipo utilizza un'antenna per identificare dei tag (etichette elettroniche o transponder). Il lettore invia un impulso radio e il transponder, attivato dall'energia dell'impulso, si attiva e risponde con un suo codice identificativo. Nel tag è presente un piccolo chip che si occupa della ritrasmissione. Il lettore è in grado anche di modificare lo stato del chip presente nel tag. I chip possono essere inseriti in etichette vere e proprie perché sono minuscoli. Sono molto utilizzati come sistemi antitaccheggio o all'interno di carte e biglietti "magnetici" o smartcard. L'evoluzione di questa tecnologia si chiama NFC (Near Field Communication) e prevede anche una comunicazione bidirezionale.

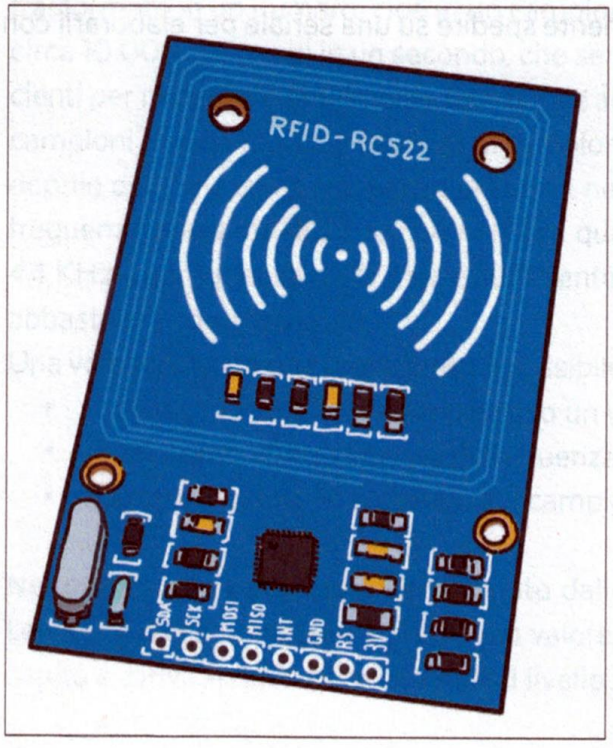

Figura 4.52 – Lettore RFID su breakout board per Arduino.

In commercio esistono svariati dispositivi, come l'Innovation ID-12, con antenna integrata, che costa qualche decina di euro, oppure l'RC522 di Mifare dal prezzo molto più accessibile.
I vari dispositivi possono avere diverse modalità di funzionamento e i dati sono forniti da una linea seriale. Il modulo ID-12 può essere collegato a un LED o a un cicalino per dare conferma della lettura e comunica con una linea seriale. Il lettore RC522 di Mifare richiede una libreria specifica e comunica utilizzando il protocollo SPI.

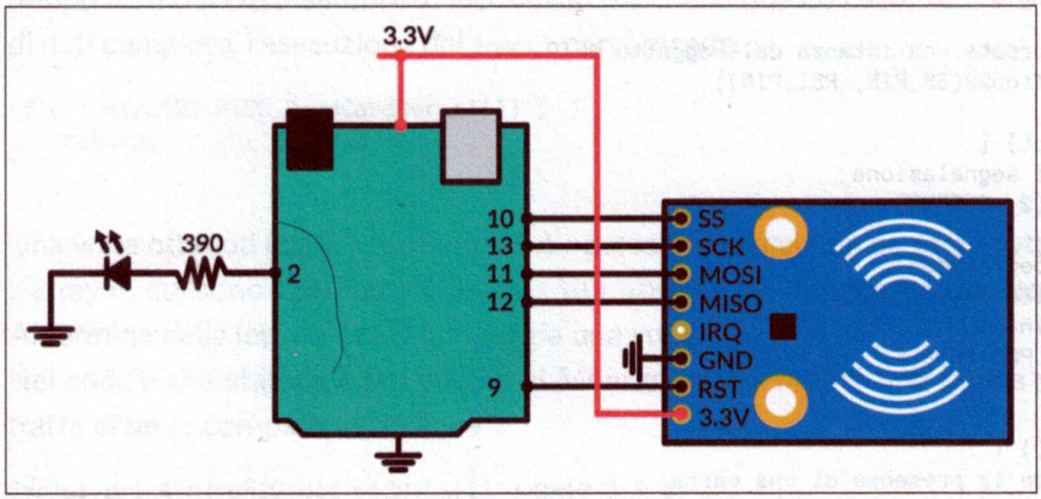

Figura 4.53 - Schema elettrico di collegamento del modulo RC522 ad Arduino.

Vediamo un esempio con il modulo RC522 della Mifare. Prima di tutto è necessario effettuare il cablaggio del lettore ad Arduino. Prestate molta attenzione alla tensione di alimentazione perché questo lettore lavora a 3,3 V e non a 5! Sulla scheda ogni piedino è contraddistinto da un nome. Collegate i pin come indicato di seguito (l'esempio è pensato per Arduino Uno):

- RST al pin 9 di Arduino;
- SDA al pin 10;
- MOSI al pin 11;
- MISO al pin 12;
- SCK al pin 13 e quindi le alimentazioni.

Per l'utilizzo del modulo RC522 serve una libreria e ne esistono di vario tipo. Potete scaricare e installare manualmente una libreria RFID all'indirizzo: https://github.com/miguelbalboa/rfid, oppure utilizzare la libreria "MFRC522" consigliata dai produttori di alcune di queste schede. Per il nostro esempio abbiamo utilizzato la "MFRC522", ottenibile dal Library Manager. La libreria "MFRC522" contiene numerosi esempi, anche

se non proprio semplicissimi, per utilizzare il lettore di tag, non solo per identificare gli ID, ma anche per accedere ai dati interni delle smartcard.
Per leggere semplicemente gli ID dei tag sono sufficienti poche righe di codice. Lo sketch riportato di seguito accende un LED, collegato al pin 2, a ogni lettura:

```
#include <SPI.h>
#include <MFRC522.h>

// Vengono definiti PIN del RFID reader
#define SS_PIN 10  // 53 per Arduino Mega
#define RST_PIN 9

// Viene creata una istanza dell'oggetto RFID
MFRC522 mfrc522(SS_PIN, RST_PIN);

void setup() {
  //led di segnalazione
  pinMode(2, OUTPUT);

  Serial.begin(9600);
  /* Abilita SPI*/
  SPI.begin();
  mfrc522.PCD_Init();
}

void loop() {
  //verifica la presenza di una carta
  if ( !mfrc522.PICC_IsNewCardPresent()){
    return;
  }
  // legge l'ID della carta
  if ( ! mfrc522.PICC_ReadCardSerial()) {
    return;
  }
  //stampa l'UID
  Serial.print("UID tag :");

  for (byte i = 0; i < mfrc522.uid.size; i++) {
    Serial.print(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");
    Serial.print(mfrc522.uid.uidByte[i], HEX);
  }
  //accende e spegne il LED
  digitalWrite(2, HIGH);
  delay(300);
  digitalWrite(2, LOW);

  Serial.println();
  delay(1000);
}
```

La lettura dell'ID (o UID) è effettuata nel `loop()`, con alcuni passaggi particolari. Il lettore si mette in attesa di un tag o una smartcard e se non rileva nulla, esce dal `loop()` con

una "return". Questa tecnica evita che il codice successivo sia eseguito. Fino a che non avvicinerete una carta valida, l'esecuzione del `loop()` si bloccherà alle prime due righe:

```
//verifica la presenza di una carta
  if ( !mfrc522.PICC_IsNewCardPresent()){
    return;
  }
```

Lo stesso approccio è utilizzato per leggere i dati della carta con `PICC_ReadCardSerial()`. La lettura può essere critica per vari motivi: la carta potrebbe allontanarsi prima del tempo e quindi i dati sarebbero incompleti. Anche in questo caso, se non si ha un set di dati completo, l'esecuzione del `loop()` non procede:

```
if ( ! mfrc522.PICC_ReadCardSerial()) {
    return;
}
```

Una volta ottenuti i dati, per stamparli è necessario leggerli da un array byte per byte. L'array in cui sono contenuti è `mfrc522.uid.uidByte[]` di lunghezza `mfrc522.uid.size`. Al termine della lettura il LED lampeggia una volta.
Nel codice che stampa i dati sul Serial Monitor avrete notato una strana sintassi: si tratta di un `if` compatto o "in linea".

```
Serial.print(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");
```

Questa riga verifica se il byte ha un valore compreso tra 0 e 9 oppure maggiore o uguale a 10. Nel primo caso stampa uno "0". Questa azione serve solo per formattare il testo scritto e aggiungere la cifra "0" a tutti i numeri tra 0 e 9. L'"if in linea" è solitamente racchiuso tra parentesi e ha tre elementi separati da un "?" e un ":":

- la condizione di test;
- il codice da eseguire se il test è vero;
- il codice da eseguire se il test è falso.

La sintassi è:

```
( test ? se-vero-fai-questo : se-falso-fai-questo)
```

ed equivale a:

```
if (test) {
  // se-vero-fai-questo
} else {
  // se-falso-fai-questo
}
```