# Raspberry Pi: RFID Tags auslesen

Diese Anleitung funktioniert mit einem RFID Reader von [Seeedstudio](http://www.seeedstudio.com/depot/electronic-brick-125khz-rfid-card-reader-p-702.html).  
In Deutschland einfach zu kaufen bei [Amazon](http://www.amazon.de/gp/product/B00CJCYV1G/ref=as_li_ss_tl?ie=UTF8&camp=1638&creative=19454&creativeASIN=B00CJCYV1G&linkCode=as2&tag=christhimbee-21).

Es handelt sich hierbei um ein fertiges RFID Modul, welches die Karten einliest und per UART oder “Wiegand” weitersendet. Ich verwende für die Kommunikation mit dem Pi die UART Methode.  
Dies lässt sich einfach per Jumper auf dem Modul einstellen.

## Los gehts:

Als erstes wird das Modul mit dem RPi wie folgt verdrahtet.

|  |  |
| --- | --- |
| **Pin RFID Modul** | **Pin Raspberry Pi** |
| GND | 6 (GND) |
| VCC | 2 (5V) |
| TX | 10 (RX) über Spannungsteiler!!! |
|  |  |

**Sehr wichtig:**  
**Der TX Ausgang des Moduls liefert einen 5V Pegel.**

**Dieser MUSS UNBEDINGT durch einen Spannungsteiler reduziert werden! Anderenfalls kann der Pi zerstört werden.**

Hierzu habe ich einen Spannungsteiler aus 2x 10kOhm Widerständen gebaut.  
Hierdurch wird die Spannung auf 2,5V halbiert.  
Dies genügt damit der Pi am Rx Eingang einen High Pegel erkennt.

Nun wird der UART Eingang freigeschaltet.  
Dies passiert in der *etc/inittab*.

Die letzte Zeile der Datei wird auskommentiert.  
Das sollte dann wie folgt aussehen:

#T0:23:respawn:/sbin/getty -L ttyAMA0 115200 vt100

Das nachfolgende Python-Script, liest die RFID-Karten ein und gibt die Seriennummer als Hex-Wert aus:

#!/usr/bin/python

# coding=utf-8

import serial

# Auslesen der RFID-Transponder

def read\_rfid():

ser = serial.Serial("/dev/ttyAMA0")

ser.baudrate = 9600

daten = ser.read(14)

ser.close()

daten = daten.replace("\x02", "" )

daten = daten.replace("\x03", "" )

return daten

id = read\_rfid()

print id

## Die Karten / Tags:

Es gibt verschiedene RIFD Tags und Karten. Jede Karte hat eine eigene ID.

Ein Profi bin ich darin auch nicht, was ich aber sagen kann, dass für das o.g. System die Karten mit den folgenden Standards funktioniert:

* 125 kHz
* EM 4100 oder EM4102

Bei den meisten Elektronikshops kostet eine Karte oder Tag ca. 1€ /Stk.  
Bei eBay bekommt man teilweise Pakete von 50-60 für umgerechnet ca 25cent /Stk.

Nun sind im wahrsten Sinne alle Türen geöffnet um jegliche Anwendung mit RFID-Tags zu bauen.

Zwei Beispiele finden sich hier:

* [RFID Card Game](http://www.elektronx.de/duelpi-rfid-game/)
* [RFID Türöffner und LCD Display](http://www.elektronx.de/rfid-turoffner-mit-lcd-display/)
* [Leseprobe aus: Raspberry Pi – Das umfassende Handbuch](http://www.elektronx.de/wp-content/uploads/2013/06/Leseprobe_RFID-Tueroeffner.pdf)

## Weitere denkbare Projekte:

* Türöffner per RIFD Tag
* Generelles Schalten von Relais nach einscannen der entsprechenden Karte.
* Personenbezogenes Licht
* Elektronischer Kalender, der durch Einlesen von personenbezogenen Karten jeweils die aktuellen Termine der Person zeigt.
* Etc, etc…

# Raspberry Pi RFID RC522 Tags auslesen (NFC)

<https://tutorials-raspberrypi.de/raspberry-pi-rfid-rc522-tueroeffner-nfc/>

RFID ist eine Technologie, wodurch Daten ohne Berührung übertragen werden, was in Chipkarten Anwendung findet. Mit einem Raspberry Pi RFID Modul (RC522) können Zugangskarten ausgelesen werden und somit z.B. Zugriff zu Türen oder Schlössern gegeben werden. Aktuelle Smartphones besitzen ähnliche

Wie man mit dem RC522 und dem Raspberry Pi RFID Tags ausließt und außerdem Chipkarten beschreibt, zeige ich in diesem Tutorial. Außerdem kann der vorgestellte Code für andere Projekte (Türöffner, Zugangskontrolle) verwendet werden.

NFC ist eine zugehörige Technologie, dessen Unterschiede [hier](http://blog.atlasrfidstore.com/rfid-vs-nfc) nachgelesen werden können (Englisch). Sowohl RFID als auch NFC senden auf einer Frequenz von 13.56 MHz, weshalb die Module miteinander kompatibel sind.

## Verwendetes Zubehör

Die folgenden (Bau-)Teile habe ich für dieses Tutorial verwendet:

* [Raspberry Pi 3](http://www.amazon.de/gp/product/B01CEFWQFA?ie=UTF8&linkCode=as2&camp=1634&creative=6738&tag=754-21&creativeASIN=B01CEFWQFA) (funktioniert auch mit allen Vorgängern)
* [Mifare RC522 RFID Modul](http://www.amazon.de/gp/product/B00QFDRPZY?ie=UTF8&linkCode=as2&camp=1634&creative=6738&tag=754-21&creativeASIN=B00QFDRPZY) (inkl. KeyCard)
* Female – Female [Jumper Kabel](https://rover.ebay.com/rover/1/707-53477-19255-0/1?toolid=20001&campid=5338064322&customid=link&mpre=http%3A%2F%2Fwww.ebay.de%2Fsch%2Fi.html%3F_from%3DR40%26_sacat%3D0%26_nkw%3Dfemale%2Bfemale-jumper%2Bkabel%26rt%3Dnc%26LH_PrefLoc%3D1)
* Lötutensilien

Es gibt auch [RFID USB Lesegeräte](http://www.amazon.de/mn/search/?_encoding=UTF8&linkCode=ur2&camp=1634&creative=19450&tag=754-21&field-keywords=RFID%20USB%20Leseger%C3%A4t), allerdings habe ich diese nicht ausprobiert.

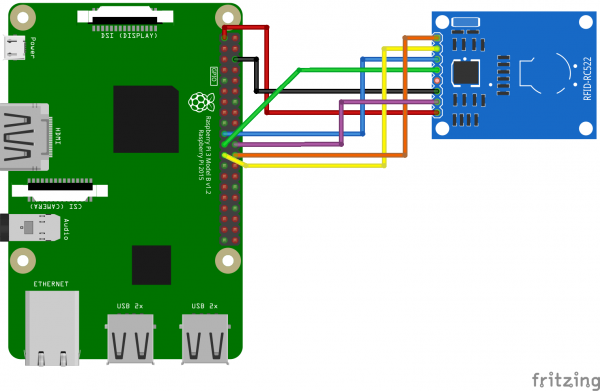
Falls man das Kartenlesegerät als Eingangskontrolle, etc. verwenden will, macht es Sinn jedem Nutzer eine Karte zu geben. Man kann diese Chipkarten in kleineren und größeren Mengen auch für kleines Geld [zusätzlich erwerben](http://www.amazon.de/mn/search/?_encoding=UTF8&linkCode=ur2&camp=1634&creative=19450&tag=754-21&field-keywords=RFID%20Karten) und dann mit dem RC522 jede Karte individuell beschreiben (Anleitung dazu weiter unten).

## Aufbau

Bei meinem Modul war die Pin Leiste noch nicht angelötet, daher musste ich sie erst löten, bevor ich die Pins verbinden konnte.  
Die Verkabelung ist dabei wie folgt:

| **RF522 Modul** | **Raspberry Pi** |
| --- | --- |
| SDA | Pin 24 / GPIO8 (CE0) |
| SCK | Pin 23 / GPIO11 (SCKL) |
| MOSI | Pin 19 / GPIO10 (MOSI) |
| MISO | Pin 21 / GPIO9 (MISO) |
| IRQ | — |
| GND | Pin6 (GND) |
| RST | Pin22 / GPIO25 |
| 3.3V | Pin 1 (3V3) |

Schematisch sieht das Ganze dann so aus:

[](https://tutorials-raspberrypi.de/wp-content/uploads/2016/04/Raspberry-Pi-RFID-RC522-NFC_Steckplatine.png)

Raspberry Pi RC522 Verbindung

## SPI aktivieren und Software installieren

Um das RFID RC522 Shield verwenden zu können brauchen wir den SPI Bus. Damit der Kernel beim Starten geladen wird, bearbeiten wir die config Datei:

sudo nano /boot/config.txt

Folgender Inhalt wird an das Ende der Datei hinzugefügt:

device\_tree\_param=spi=on

dtoverlay=spi-bcm2708

Gespeichert und beendet wird mit STRG+O, STRG+X. Danach aktivieren wir SPI noch:

sudo raspi-config

Unter „Advanced Options“ > „SPI“ aktivieren und den Neustart bestätigen (alternativ sudo reboot now).

Danach kann mittels dmesg | grep spi überprüft werden, ob das Modul geladen wurde. Es sollte dann so einen Output ergeben:

pi@raspberrypi:~ $ dmesg | grep spi

[   10.784368] bcm2708\_spi 20204000.spi: master is unqueued, this is deprecated

[   10.813403] bcm2708\_spi 20204000.spi: SPI Controller at 0x20204000 (irq 80)

Nun müssen noch die Pakete installiert werden, sodass wir auf den SPI Bus zugreifen können und eine entsprechende Bibliothek aus [GitHub](https://github.com/mxgxw/MFRC522-python.git) laden können.

sudo apt-get install git python-dev --yes

Zuerst installieren wir also das Python SPI Modul

git clone https://github.com/lthiery/SPI-Py.git

cd SPI-Py

sudo python setup.py install

cd ..

und danach die Raspberry Pi RFID RC522 Bibliothek:

git clone https://github.com/mxgxw/MFRC522-python.git && cd MFRC522-python

## Raspberry Pi RFID Reader/Writer testen

Zusätzlich zum RC522 Modul werden meist eine weiße Karte sowie ein NFC fähiger Schlüsselanhänger geliefert. Diese Teile können als Authentifizierung benutzt werden, da sie beschreibbar und lesbar sind. Ebenfalls könnte ein NFC fähiges (Android/iOS) Smartphone verwendet werden (was die meisten neueren Handys ja sind).

Um einen ersten Test der Karte/Schlüsselanhängers durchzuführen, lassen wir das Skript laufen:

sudo python Read.py

Sobald die Chipkarte daran gehalten wird und erkannt wird, sieht man eine Ausgabe wie diese:

pi@raspberrypi:~/MFRC522-python $ sudo python Read.py

Welcome to the MFRC522 data read example

Press Ctrl-C to stop.

Card detected

Card read UID: 69,245,238,117

Size: 8

Sector 8 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

Um nun die gespeicherten Daten (Zahlen) auf dem Chip zu ändern, bearbeiten wir die „Write.py“ Datei (sudo nano Write.py). Dazu bearbeitest du den Code ab Zeile 55 folgendermaßen (die 16 Zahlen von data kannst du dabei frei zwischen 0 und 255 wählen. Ich habe dabei ein Wort mit [ASCII Zeichen dargestellt](http://asciivalue.com/index.php)):

## Python

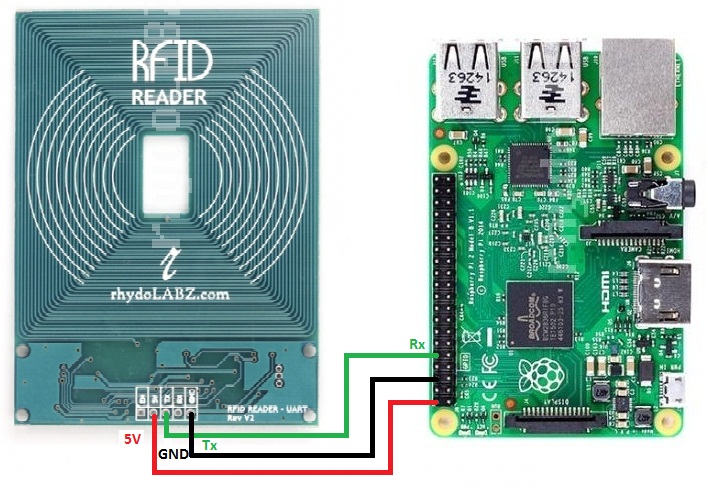
|  |  |
| --- | --- |
| 55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91 | # Variable for the data to write              data = [114, 97, 115, 112, 98, 101, 114, 114, 121, 45, 116, 117, 116, 111, 114, 0]                # Fill the data with 0xFF              #for x in range(0,16):              #    data.append(0xFF)                print "Sector 8 looked like this:"              # Read block 8              MIFAREReader.MFRC522\_Read(8)              print "\n"                #print "Sector 8 will now be filled with 0xFF:"              # Write the data              MIFAREReader.MFRC522\_Write(8, data)              #print "\n"                print "It now looks like this:"              # Check to see if it was written              MIFAREReader.MFRC522\_Read(8)              print "\n"                """              data = []              # Fill the data with 0x00              for x in range(0,16):                  data.append(0x00)                print "Now we fill it with 0x00:"              MIFAREReader.MFRC522\_Write(8, data)              print "\n"                print "It is now empty:"              # Check to see if it was written              MIFAREReader.MFRC522\_Read(8)              print "\n"              """ |

# RhydoLabz RFID Reader (125Khz) – Serial TTL Interfaced with Raspberry Pi2

[Mahesh](http://www.rhydolabz.com/wiki/?author=9) August 8, 2015 [Raspberry Pi](http://www.rhydolabz.com/wiki/?cat=39), [RFID Reader & Tag](http://www.rhydolabz.com/wiki/?cat=8)

The demand in security made the engineers to develop more intelligent electronic equipments which could be used in the day to day life process. RFID(Radio Frequency Identification) is a high end secured device which can be applied in any area where security is inevitable. RFID is the wireless use of [electromagnetic fields](https://en.wikipedia.org/wiki/Electromagnetic_field) to transfer data, for the purposes of automatically identifying and tracking tags attached to objects. The tags contain electronically stored information which is transferred to the reader in wireless manner. The rhydoLABZ RFID reader reads EM4100 family transponder tags that are brought in proximity to the reader and output the unique tag identification number through serial port @9600 bps. The reader output is 12 byte including one start, stop byte and 10 unique data byte. The normal detection range is 10-15CM for Card Type TAGs. The RFID Reader (as well as the RFID tags sold by rhydoLABZ) uses the EM4102 protocol. Any other tags that also use the EM4102 protocol can be used with the rhydoLABZ RFID reader.

Based on the unique Id of each card, it can be discriminated after receiving the suitable data serially. To develop a highly advanced security system RFID module is interfaced with Raspberry Pi2. As the Raspberry pi  can perform face recognition, both the process can be clubbed ie the authentication must be only given by obtaining the satisfied outputs from RFID and image processing.

[](http://www.rhydolabz.com/wiki/wp-content/uploads/1109.jpg)

## **Modules Needed**

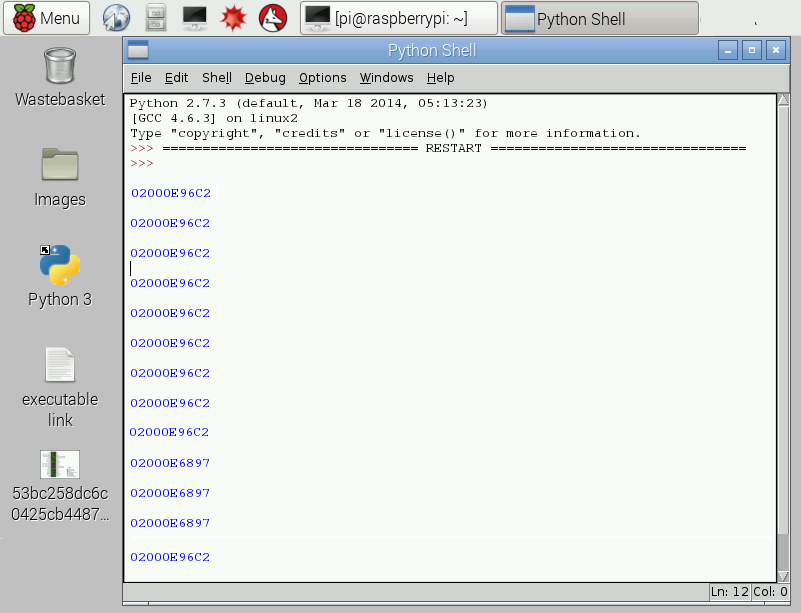
1. Raspberry Pi 2
2. RFID Reader (125Khz) – Serial TTL
3. RFID Clamshell Card
4. Connecting Wires

Connect the 5V pin(pin No 2) and ground(pin No 6) of the Raspberry Pi to the 5V and ground pin of the RFID module. As the data transmission is taking place in the RFID connect the Tx pin of the RFID to the Rx pin(pin No 10) of the Raspberry Pi. The power led will become on after the connections. Using Pyrhon script read the serial data and display the output on the Python shell.

A sample code is provided for reading the RFID using Python.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22 | import serial  import os, time    #Enable Serial Communication  port = serial.Serial("/dev/ttyAMA0", baudrate=9600, timeout=0.01)    # Find a character in a string  def find(str, ch):      for i, ltr in enumerate(str):          if ltr == ch:              yield i    fd=''  while True:      # Read the port      rcv = port.read(10)      if len(rcv) &gt; 1:          fd=fd+rcv          ps=fd.find('\r')          if ps &gt;= 0:              print fd[0:ps]              fd='' |

After running this code place the tag near to the reader and get the data on the Python shell.

[](http://www.rhydolabz.com/wiki/wp-content/uploads/2015-08-08-104406_1318x736_scrot.png)

After checking the corresponding ID each cards could be discriminated and then applied for any secured purposes.