NFC auf Raspberry Pi: PN532 Breakout Board

Ich entwickle zurzeit eine kleine Software, welche es ermöglicht, einen interaktiven, verschlüsselten Datenaustausch zwischen einem **NFC**-fähigen Android-Smartphone und einem computergesteuerten NFC-Lesegerät durchzuführen.

Um auch zu Hause an dem Projekt arbeiten zu können, habe ich mir ein **NFC-Breakout-Board für meinen Raspberry Pi** gekauft. In diesem Artikel will ich aufzeigen, wie man das Gerät **an die GPIO-Pins des Raspberry Pi anschließt** und wie man es konfiguriert.

Alles was man dafür braucht, ist ein Raspberry Pi, ein paar Jumperkabel und ein passendes NFC-Breakout-Board (Details siehe unten).

Zwischenbemerkung: Warum ein PN532-NFC-Breakout-Board?

Falls sich jemand fragt, warum ich mir nicht einfach einen **USB-NFC-Reader** gekauft habe: Für meinen Anwendungsfall ist es nötig, mit dem NFC-Lesegerät einen NFC-Tag zu emulieren. Dies funktioniert nur mit dem **NXP PN532 NFC-Controller** problemfrei (die neueren PN533-Chips können das nicht mehr).

Zwar gab es von *Identive (früher SCM Microsystems)* eine sehr gut funktionierende USB-Version des PN532-Controllers *(Arygon ADRB-USB)*, diese wird aber nicht mehr produziert und ist im Handel auch nicht mehr zu bekommen.

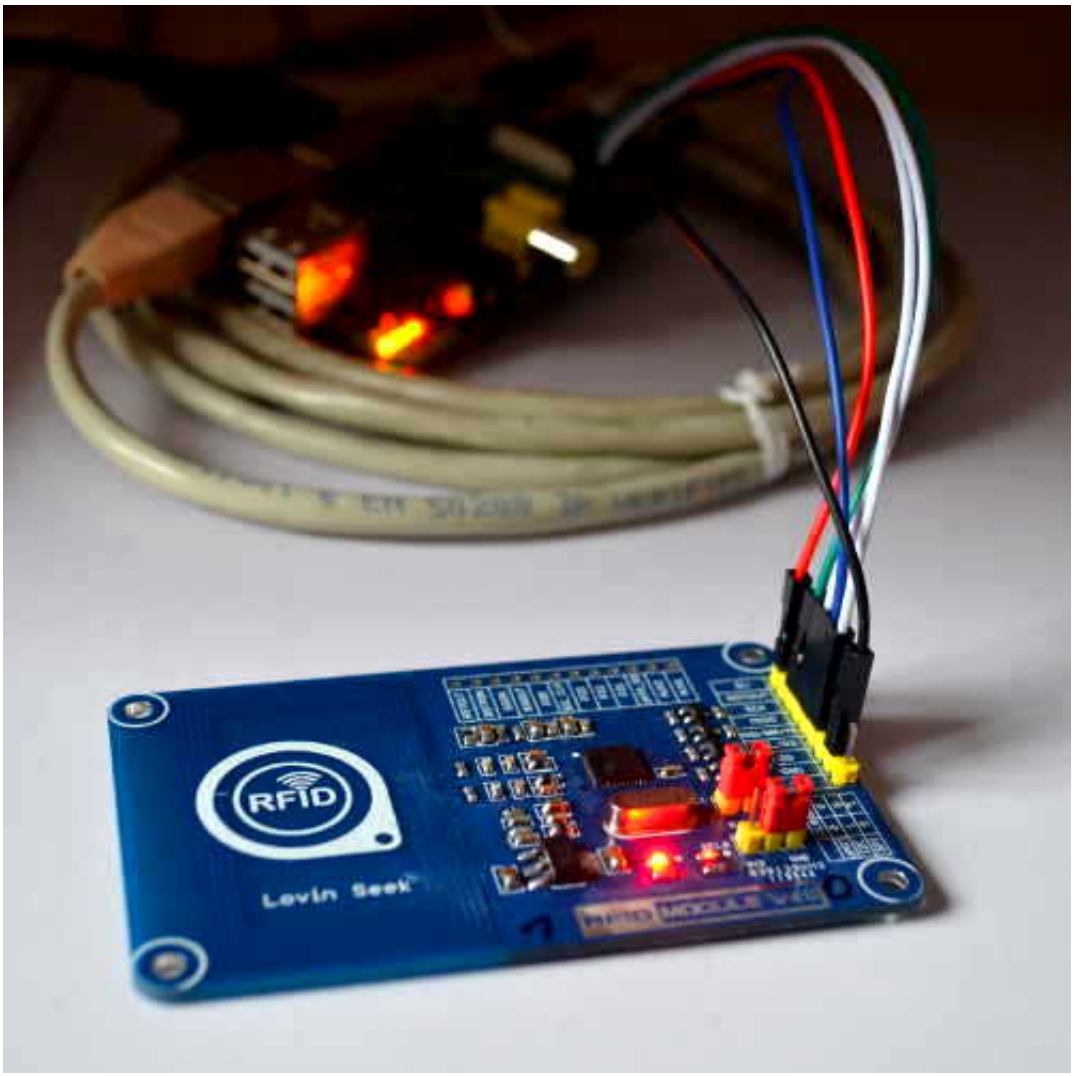
Die PN532-Breakout-Boards bekommt man hingegen sehr einfach. Es gibt verschiedene Ausführungen dieser Platinen. In der Regel steht in der

Beschreibung, dass sie für die **Arduino**-Plattform gedacht sind. Mit dem **Raspberry Pi** kann man sie aber genauso benutzen.

(Hinweis für ganz Interessierte: Man kann sich auch ein Breakout-Board kaufen und es an eine USB-UART-Bridge hängen, dann hat man einen selbst gebauten USB-NFC-Reader.)

PN532 Breakout Board: Was ich gekauft habe …

Ich habe mich für diese Ausführung entschieden, weil hier schon alle Pins auf dem Board angebracht sind. Bei Amazon und eBay findet man auch Platinen, bei denen man erst ein bisschen Löten muss, bevor man loslegen kann (die benötigten Stecker und Pins werden oftmals nicht mitgeschickt).



Zum Lieferumfang gehört auch ein kleiner *(Mifare Classic)* NFC-Tag mit 1 Kilobyte Speicherkapazität (Schlüsselanhänger-Ausführung).

PN532-Breakout-Board Modi: SPI, UART und I²C

Das RFID/NFC-Shield unterstützt gleich drei Kommunikationsarten: SPI, UART und I²C. Je nachdem, für welche Kommunikationsart man sich entscheidet, verändert sich auch die Verkabelung zwischen dem Breakout-Board und dem Raspberry Pi sowie die Software-Konfiguration *(LibNFC)*.

Ich habe alle drei Varianten getestet und werde sie im Folgenden auch alle erklären. Zuvor möchte ich aber noch (als Entscheidungshilfe) ein paar Worte zu den Geschwindigkeitsunterschieden zwischen UART, I²C und SPI, und auch zu den bei mir aufgetretenen Problemen los werden.

UART (manchem vielleicht eher als serielle Schnittstelle *RS232* bekannt):

- Recht langsame Verbindung (115,2 kbps).
- Für die Verbindung mit dem Raspberry Pi werden vier Jumperkabel benötigt.
- Wird auch im oben genannten Arygon ADRB-USB-NFC-Reader genutzt (über eine CP2102 USB-UART-Brücke). Man müsste also davon ausgehen, dass es gut funktioniert. Bei meinem Breakout-Board ist dies leider nicht zu 100% der Fall. Ich kann via UART zwar problemlos NFC-Tags lesen und schreiben, beim Emulieren eines Tags gibt es aber Timing-Probleme, die zur Fehlfunktion führen.

  Ich vermute, dass dieser Fehler bei einer anderen Ausführung des Breakout-Boards nicht auftritt. Wahrscheinlich hat meins nur eine kleine Macke, ich habe dazu jedenfalls nichts im Netz gefunden.

  (Noch mal für die ganz Interessierten: Das gleiche Problem habe ich auch, wenn ich das Board über die oben verlinkte UART-Bridge an den USB-Port meines Laptops hänge.)

I²C:

- Etwas schnellere Verbindung (400 kbps).

- Für die Verbindung mit dem Raspberry Pi werden vier Jumperkabel benötigt.
- Funktioniert mit meinem PN532-Board eher mäßig, weil die I²C-Verbindung instabil ist. Laut *# i2cdetect* springt die ID des Boards auf dem I²C-Bus immer zwischen 24 und 25 hin und her. Somit ist keine längere NFC-Verbindung möglich, wie ich sie für meinen Anwendungsfall bräuchte. Dieses Problem scheint es mit vielen NFC-Breakout-Boards zu geben. Im Netz findet man jedenfalls einige englischsprachige Foreneinträge dazu.

SPI:

- Sehr schnelle Verbindung (>4 mbps, Full Duplex).
- Für die Verbindung mit dem Raspberry Pi werden sechs Jumperkabel benötigt.
- Läuft perfekt und sehr schnell, sogar schneller als der bereits erwähnte Arygon ADRB-USB-Reader, den ich außer Haus nutze. (Wenn man bedenkt, dass die maximale Geschwindigkeit einer NFC-Funkverbindung bei 106 bis 424 kbps liegt, ist dies auch nicht verwunderlich, denn UART schafft nur 115,2 kbps.)

Damit sollte eigentlich auch klar sein, in welche Richtung meine Empfehlung geht. SPI siegt auf ganzer Linie, braucht aber auch zwei Datenleitungen mehr. Via UART läuft das Breakout-Board ebenfalls nahezu stabil, von I²C würde ich eher die Finger lassen.

Es folgen nun die Beschreibungen zur **Einrichtung des RFID/NFC-Boards** für die genannten Betriebsmodi. Softwareseitig beziehe ich mich bei diesem **Tutorial** auf LibNFC, die wohl am weitesten ausgereifte Open-Source-Library für NFC und RFID. (Wer lieber Python mag, kann sich mit nfcpy vertraut machen - ich selbst habe das nur kurz angetestet.)

Beim Betriebssystem gehe ich von einer aktuellen **Raspbian**-Version aus. Hierzu sei noch erwähnt, dass ich mein Breakout-Board erst mit UART

und I²C zum Laufen bekam, nachdem ich mein Raspbian-Linux neu aufgesetzt habe. Meine alte Version (vor circa einem Jahr installiert) habe ich zwar immer geupdatet, aber aus irgendeinem Grund wollte sich das System nicht mit der Hardware verstehen. Wer also aus nicht nachvollziehbaren Gründen gar nichts zum Laufen bekommt, sollte sich mal eine zweite SD-Karte nehmen und einen Versuch mit einem neu installierten Linux wagen.

LibNFC installieren

So, jetzt geht's aber wirklich los mit der Installationsanleitung. :)

Zuerst muss LibNFC installiert werden. Mit den folgenden Kommandos wird das Paket heruntergeladen und entpackt:

 # wget https://libnfc.googlecode.com/files/libnfc-1.7.0.tar.bz2
 # tar -jxvf libnfc-1.7.0.tar.bz2
 # cd libnfc-1.7.0

Danach geht es ans Konfigurieren und Kompilieren. Mit dem folgenden Befehl wird LibNFC samt der PN532-Treiber für UART, SPI und I2C konfiguriert. Man kann die *--with-drivers*-Option auch weglassen, dann werden alle verfügbaren Treiber bereitgestellt. In diesem Fall sollte man LibUSB installiert haben *(# apt-get install libusb-dev)*, da dies für die USB-NFC-Treiber benötigt wird.

Wer sich sicher ist, dass er nur einen bestimmten Treiber braucht, kann die anderen natürlich aus der Zeile entfernen.

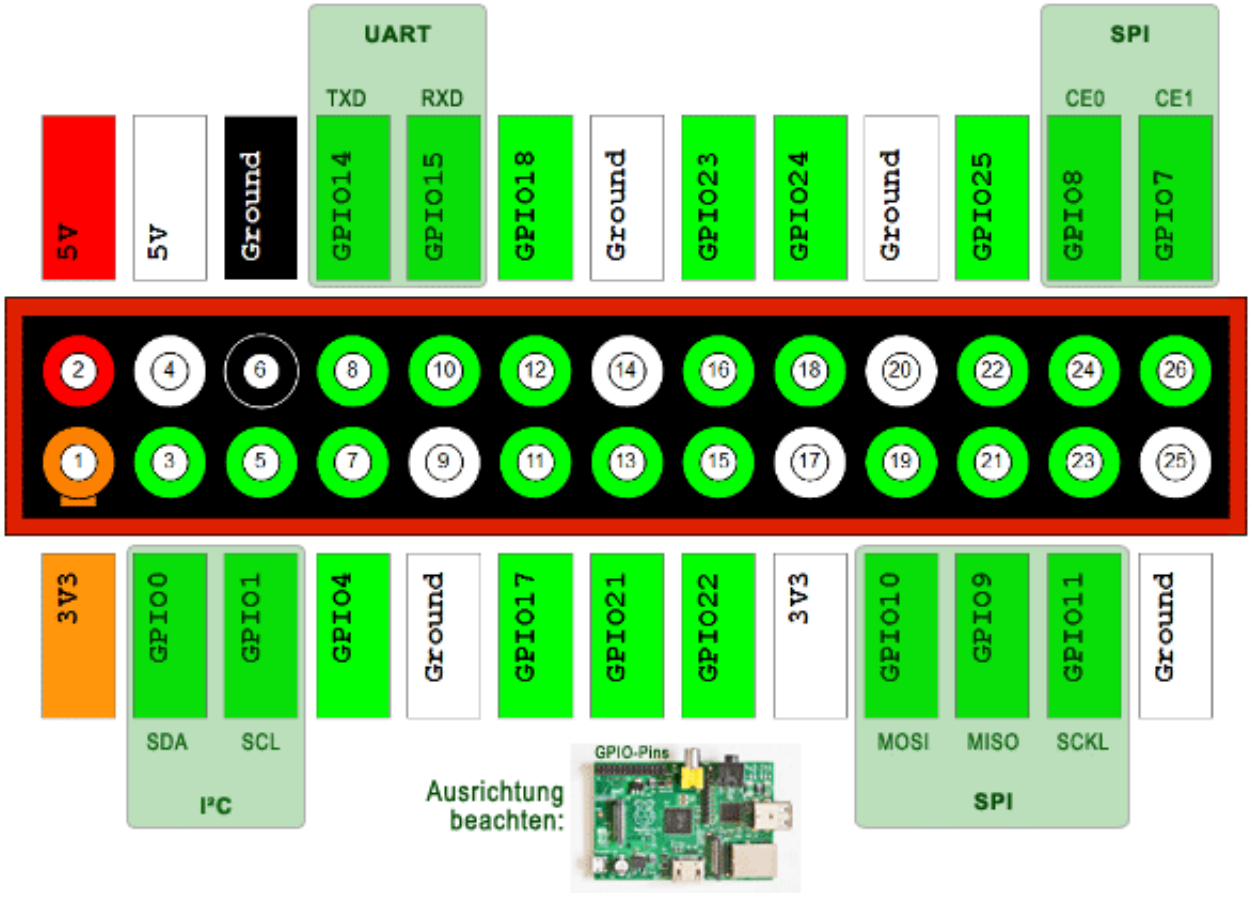
./configure --prefix=/usr --sysconfdir=/etc --with-drivers=pn532_uart,pn532_spi,pn532_i2c

Der Befehl *# make install all* startet die Installation. Der Kompiliervorgang kann auf dem langsamen ARM-Prozessor des Raspberry Pi einige Minuten

dauern.

Die GPIO-Pins des Raspberry Pi: UART, I²C und SPI

Damit man das Breakout-Board an den Raspberry Pi anschließen kann, muss man natürlich erst einmal wissen, welche Pins die richtigen sind. Für alle drei Kommunikationsarten sind GPIO-Pins vorgesehen:



PN532 Breakout-Board Anschlüsse

Am PN532-Breakout-Board findet man Pins mit den Beschriftungen "3,3V", "5V", "SCK", "MISO", "MOSI/SDA/TX", "SSEL/SCL/RX", "RSTOUT_IN", "IRQ" und "GND". Außerdem sind auf dem Board **zwei Jumper** (bei manchen Ausführungen auch kleine Schiebeschalter) angebracht, welche zur Auswahl der Kommunikationsart dienen. Ich beschreibe im Folgenden, wie die Jumper auf meinem Board gesetzt werden müssen. Soweit ich es in Bildern gesehen habe, ist dies für alle PN532-Boards in gleicher Weise gelöst. Dennoch empfehle ich jedem, sich nach dem Aufdruck auf seiner Platine zu richten, da es nicht auszuschließen ist, dass sich die Belegung

unterscheidet.

PN532-Breakout-Board via SPI an den Raspberry Pi anschließen

Um **das Breakout-Board via SPI mit dem Raspberry Pi zu verbinden**, stellt man den Jumper 0 auf "Off" (Jumper sitzt rechts, linker Pin bleibt frei, bei manchen Boards ist die Stellung mit "GND" gelabelt). Jumper 1 muss hingegen auf "On" stehen (manchmal auch mit "3V3" gekennzeichnet).

Für die Verkabelung benötigt man sechs Jumperkabel:

- RPi GPIO Pin 2 (5V) ↔ PN532 5V
- RPi GPIO Pin 6 (GND) ↔ PN532 GND
- RPi GPIO Pin 8 (SPI/CE0) ↔ PN532 SSEL/SCL/RX
- RPi GPIO Pin 10 (SPI/MOSI) ↔ PN532 MOSI
- RPi GPIO Pin 9 (SPI/MISO) ↔ PN532 MISO
- RPi GPIO Pin 11 (SPI/SCKL) ↔ PN532 SCK

Software-Konfiguration: PN532 via SPI

Damit **SPI** auf dem Raspberry Pi verfügbar ist, muss man das zugehörige Kernel-Modul freischalten. Dieses ist in Raspbian standardmäßig über die Blacklist-Datei */etc/modprobe.d/raspi-blacklist.conf* gesperrt. Die Zeile mit dem Modul *spi-bcm2708* sollte man aus der Datei entfernen oder auskommentieren (ein #-Zeichen davorschreiben). Nach der Änderung muss der Raspberry Pi neu gebootet werden.

Für die Konfiguration von LibNFC folgen unten noch Hinweise.

PN532-Breakout-Board via UART an den Raspberry Pi anschließen

Das **Hardware-Setup für UART** ist ähnlich simpel, wie das für SPI. Beide Jumper werden auf "Off" (="GND") gestellt. Mit vier Jumperkabeln

verbindet man die beiden Platinen:

- RPi GPIO Pin 2 (5V) ↔ PN532 5V
- RPi GPIO Pin 6 (GND) ↔ PN532 GND
- RPi GPIO Pin 14 (TXD) ↔ PN532 SSEL/SCL/RX
- RPi GPIO Pin 15 (RXD) ↔ PN532 MOSI/SDA/TX

Software-Konfiguration: PN532 via UART

Die Verkabelung wäre also schon mal erledigt. Damit man aber eine **UART**-Datenverbindung mit dem NFC-Reader herstellen kann, muss man die betreffenden Pins noch softwareseitig freilegen. Denn unter Raspbian sind diese in der Regel schon mit einer Linux-Konsole belegt.

Um diese abzuschalten (und die UART-Pins somit verfügbar zu machen) muss man Folgendes tun:

- In der Datei */boot/cmdline.txt* entfernt man alle Einträge und Parameter, in denen "ttyAMA0" vorkommt.
  Beispiel:
  dwc_otg.lpm_enable=0 console=ttyAMA0,115200 kgdboc=ttyAMA0,115200 console=tty1 $
  wird zu
  dwc_otg.lpm_enable=0 console=tty1 $
- In der Datei */etc/inittab* kommentiert man die folgende Zeile aus (indem man ein #-Zeichen an den Zeilenanfang schreibt):
  # T0:23:respawn:/sbin/getty -L ttyAMA0 115200 vt100
- Der Raspberry Pi muss neu gestartet werden.

Weiter unten kommen noch Hinweise zur Konfiguration von LibNFC.

PN532-Breakout-Board via I²C an den Raspberry Pi

anschließen

Für eine **I²C**-Verbindung zwischen dem Raspberry Pi und dem PN532-Breakout-Board, setzt man den Jumper 0 auf "On" (="3V3") und Jumper 1 auf "Off" (="GND"). Die Verkabelung erfolgt so:

- RPi GPIO Pin 2 (5V) ↔ PN532 5V
- RPi GPIO Pin 6 (GND) ↔ PN532 GND
- RPi GPIO Pin 0 (SDA) ↔ PN532 MOSI/SDA/TX
- RPi GPIO Pin 1 (SCL) ↔ PN532 SSEL/SCL/RX

Software-Konfiguration: PN532 via I²C

Auch für **I²C** wird ein gesperrtes Kernel-Modul benötigt. Dieses heißt *i2c-bcm2708* und ist ebenfalls in der Datei */etc/modprobe.d/raspi-blacklist.conf* geblacklisted. Der Eintrag muss auskommentiert (ein #-Zeichen davorschreiben) oder entfernt werden.

Des Weiteren sollte man das Modul *i2c-dev* explizit zur Datei */etc/modules* hinzufügen (einfach "i2c-dev" in eine neue Zeile schreiben).

Außerdem empfiehlt es sich, die i2c-Tools zu installieren *(# apt-get install i2c-tools)*. Damit verfügt man auch über das Programm *i2cdetect*, mit welchem man überprüfen kann, ob der NFC/RFID-Reader gefunden wurde. Hierzu gibt man als root-Nutzer *# i2cdetect -y 1* ein (bei älteren Rasperry Pi Revisionen muss der Befehl *# i2cdetect -y 0* lauten).

Der Output sieht dann ungefähr so aus:

```
sudo i2cdetect -y 1

     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00: -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- 24 -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

In diesem Fall meldet sich das Board mit der ID 24 auf dem I²C-Bus zu Wort. Wie oben schon erwähnt, springt der Wert bei mir des Öfteren zwischen 24 und 25 hin und her, weswegen ich I²C nicht benutze.

Es folgt der letzte Schritt der Konfiguration: die LibNFC-Einrichtung.

LibNFC Konfiguration

LibNFC benötigt noch zwei Konfigurationsdateien, damit es den NFC/RFID-Reader auch findet. Für diese legt man im */etc/*-Verzeichnis die Ordner *nfc* und *nfc/devices.d* an.

Die Datei */etc/nfc/libnfc.conf* enthält nur eine Zeile.

allow_intrusive_scan=true

Wichtiger ist die Datei */etc/nfc/devices.d/device.conf*, welche für **SPI** den folgenden Inhalt aufweist:

name = "Breakout Board"
connstring = pn532_spi:/dev/spidev0.0:500000

Für **UART** lautet der *Connstring* wie folgt:

name = "Breakout Board"
connstring = pn532_uart:/dev/ttyAMA0

Bei **I²C** ist der *Connstring*, wie oben schon angedeutet, von der Geräte-Version abhängig. Er lautet

name = "Breakout Board"
connstring = pn532_i2c:/dev/i2c-0

oder

name = "Breakout Board"
connstring = pn532_i2c:/dev/i2c-1

Damit wäre die Konfiguration beendet und das Breakout-Board sollte sich mit LibNFC verwenden lassen. Zum Testen kann man einfach mal den Befehl *# nfc-list* eingeben. Für SPI sieht der Output dann so aus:

 # nfc-list
nfc-list uses libnfc 1.7.0
NFC device: pn532_spi:/dev/spidev0.0 opened

(Wenn ein Tag auf dem Reader liegt, steht noch ein bisschen mehr da.)

Mehr zur LibNFC-Konfiguration findet man hier. Bei Problemen kann man vor jeden LibNFC-Befehl *"LIBNFC_LOG_LEVEL=3"* schreiben, dann bekommt man ausführlichere Debug-Informationen angezeigt.

Damit wäre ich auch am Ende dieses Artikels angelangt. Ich hoffe das Tutorial hilft jemandem bei der Konfiguration seines NFC/RFID-Readers weiter.

Mehr Artikel zum Thema NFC, gibt es hier.
Bei Fragen, einfach kommentieren. :)