Puzzle1

Objectius:

El principal objectiu d'aquest primer puzle és connectar i configurar la Raspberry Pi 3B+ al nostre ordinador, configurar un PN532 module NFC i escriure un document en Python3 per tal que llegeixi la informació del UID-user de la targeta i imprimeix per pantalla en hexadecimal, tal com es mostra a la següent imatge.

```
pi@raspberrypi:~/py532lib $ python3 puzzle1.py
E34BCB97
```

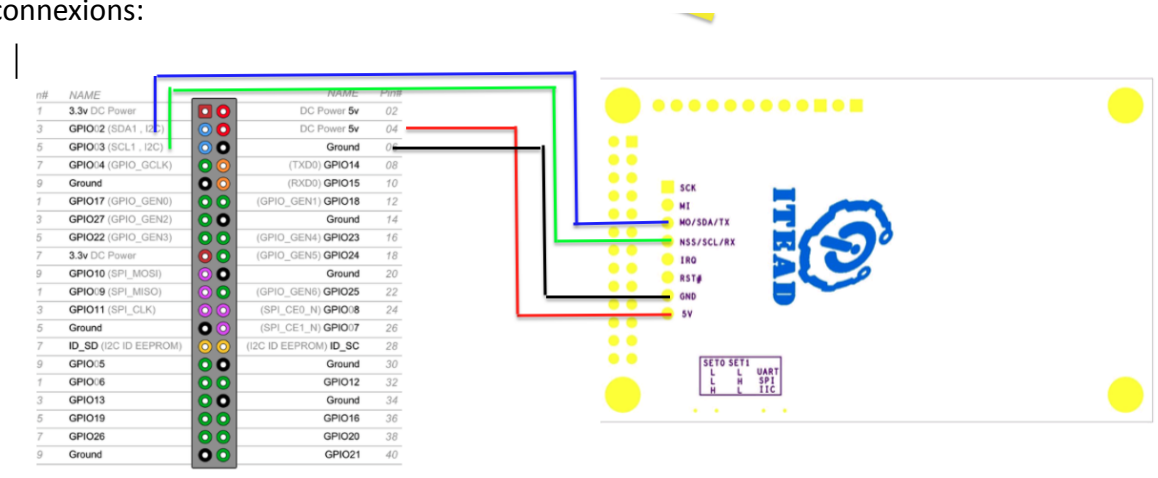
Configuració de la Raspberry:

La configuració de la raspberry la vaig fer a partir de la pàgina oficial de Raspberry pi https://www.raspberrypi.org/software/ allà em vaig instal·lar el Raspberry Pi Imager, una aplicació per instal·lar de manera ràpida la imatge a la targeta microSD. Un cop a dins de l'instal·lador, vaig triar la versió del sistema operatiu més complet, "Raspberrry Pi OS Full (32 bit)" a més a més vaig canviar la configuració avançada per tal de poder connectar-ho per SSH o pel meu dispositiu mòbil, d'altra banda, vaig canviar la contrasenya per poder accedir-hi, un cop fet tot això vaig planxar la imatge a la meva targeta microSD.
Tot seguit vaig procedir a connectar la Raspberry a l'ordinador a través d'ethernet, és a dir, per SSH, amb el següent codi:

```
anna@MacBook-Pro-de-Anna ~ % ssh pi@raspberrypi.local
pi@raspberrypi.local's password:
```

Configuració ITEAD PN325 NFC Module:

Un cop la raspberry està connectada a l'ordinador hem d'instal·lar la llibreria adient per tal de configurar l'ITEAD PN532 NFC Module i així poder aconseguir una bona detecció. Per aconseguir-ho he instal·lat la llibreria "libnfc-1.7.1" i he seguit les instruccions de la pròpia llibreria per fer el correcte insta lament a més a més he aplicat el següent codi per fer la detecció a través d'una connexió i2c.
"sudo raspi-config" i després a la pantalla que apareix Options->I2C->Enable
D'altra banda, vaig connectar l'Itead PN325 NFC Module amb la Raspberry amb les següents connexions:



Per comprovar que el NFC Module està correctament configurat fem el següent:

```
pi@raspberrypi:~ $ 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: -- -- -- -- -- -- -- --
pi@raspberrypi:~ $ nfc-list
nfc-list uses libnfc 1.7.1
NFC device: pn532_i2c:/dev/i2c-1 opened
1 ISO14443A passive target(s) found:
ISO/IEC 14443A (106 kbps) target:
    ATQA (SENS_RES): 00  04
       UID (NFCID1): e3  4b  cb  97
      SAK (SEL_RES): 08
```

Com es pot observar detecta correctament el nostre dispositiu.

Implementació del codi:

Un cop ja hem instal·lat tot ara hem d'escriure el codi en Python 3 per tal de fer un programa que compleixi el nostre principal objectiu. Per tal de dur el nostre programa, utilitzarem la llibreria "py532lib". La Instal·lació d'aquesta llibreria la faig a partir d'un repositori de github amb la següent adreça: https://github.com/HubCityLabs/py532lib.git i aplicar "git clone" a la terminal per tal de copiar tots els documents necessaris a la nostre raspberry.
Amb la llibreria instal·lada, hem escrit el següent codi:

```
from py532lib .i2c import*
from py532lib.frame import*
from py532lib.constants import*


pn532=Pn532_i2c()
pn532.SAMconfigure();

class RFID:
        card_data= pn532.read_mifare().get_data()
#passar de bytearray a hexadecimal
        card_info=int.from_bytes(card_data,byteorder='big', signed=True)
        card_hexa= hex(card_info)
        card=''
        for i in range(16,len(card_hexa)):
                card=card+str(card_hexa[i])
        print(card.upper())
```

Per començar he importat els tres documents necessaris de la llibreria. Tot seguit inicialitzo el sensor amb la funció "SAMconfigure()". Per tal d'obtenir el UID de la targeta utilitzo la funció "read_mifare()" i la funció de "get_data" per copiar tota la informació del Frame. Aquesta funció em retorna un bytearray amb tota la informació, però a nosaltres només ens interessa el UID-user que són els últims bits. Per aconseguir només el UID-user primer

converteixo el bytearray amb un integer amb la funció "int.from_byte(card_data,byteorder="big",signed=True)", un cop ho tinc en integer ho passo a hexadecimal amb la funció "hex()". Finalment, imprimeixo per pantalla només els bits necessaris per saber el UID-user i en majúscules.

Problemes:

El primer problema que vaig tenir va ser en el moment de instal·lar-me la primera llibreria per configurar el perifèric, ja que totes les pàgines web a les que em dirigia la pàgina web de la Wikipedia i altres per tal de descarregar-me la llibreria estaven fora de funcionament. Finalment, vaig trobar un GitHub que tenia la llibreria en .tar.bz2 però no podia descarregar-m'ho directament a la raspberry, per tant, el que vaig fer va ser descarregar-m'ho al meu ordinador descomprimir-ho i pujar-ho al meu GitHub i des de allà sí que m'ho podia baixar a la raspberry utilitzant "git clone".

El segon problema ha estat la funció de la llibreria que he utilitzat per tal de llegir el uid-user, ja que em retorna un bytearray amb un prefix de 16 números els quals son iguals per totes les targetes que detecti, ho he solucionat al codi.