Cyber Physical Systems

Wintersemester 2020/21

Hochschule München University of Applied Sciences

Lektion 05: NFC Registrierung

von: Max Holzmann

NFC wird in der Rally Anwendung dazu verwendet um die RallyESPs bei m Rally MAster zu Registrieren. Beim Registrierungvorgang wird dem Rally ESP ein eineindeutiger Topic übergeben.

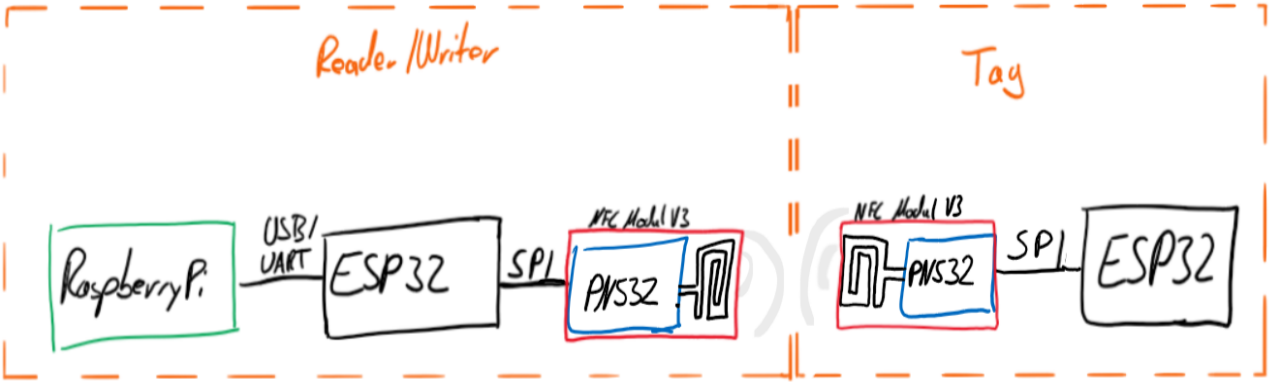
Ziele

- Emulieren eines NFC Tags mit dem ESP32 und einem NFC Board mit PN532 Chip
- ESP32 mit NFC Board mit PN532, um den Inhalt des emulierten NFC Tags zu lesen und zu beschreiben
- Anbinden des Lese/Schreibe-ESPs über USB an den Raspberry und weitegeben der Daten in Node Red

Hinweis: Zum Zeitpunkt der Implementierung (Januar 2021) konnte keine Bibliothek für den ESP gefunden werden, die die NFC Tag Emulation vollkommen implementiert hat.
Die noch am besten passende Bibliothek ist die Adafruit-PN532 Bibliothek, welche drei rudimentäre Methoden zur Tag Emualtion beinhaltet. Jedoch noch komplett ohne NDEF oder spezielle Tag Type 1-4 Methoden. Es wird lediglich ein Beispiel für Payment Karten nach dem APDU Standard bereitgestellt ohne Dokumentation und Details.
Da die komplette Entwicklung einer Tag-Emualtion-Bibliothek zu viel Aufwand ist, wurde in der Lektion eine einfache HEX Daten Tagemulation auf Basis des Payment Beispiels verwendet. Diese Funktion wiederum unterstützt keine Raspberry Pi Bibliothek für den PN532, daher muss auch hier ein ESP zwischengeschalten werden. In der Weiterentwicklung dieses Projekts müssen die Hex Werte durch NDEF Records ersetzt werden, um näher an die NFC Tag Emulation zu gelangen. Beziehungsweise werden in naher Zukunft wahrscheinlich Bibliotheken für den ESP veröffentlicht.
Bei der Recherche wurde auch die Bibliothek von SeedStudio getestet, welche gemeinsam mit der NDEF Bibliothek zu verwenden ist. Leider ist diese zur Zeit sehr fehleranfällig und konnte nicht stabil zum Laufen gebracht werden. (link SeedStudio Git repo)

Voraussetzungen und erforderliches Equipment

- Die Tutorial Lektion 01: Grundlegende Einstellungen und Installationen muss erledigt sein link
- Hardware
    - 1x Raspberry Pi 3 B
    - 2x ESP DevKitC v4
    - 2x NFC Board with PN532 shopping link



Lösungsschritte

In dieser Lektion wird die C/C++ Bibliothek von Adafruit verwendet um ein NFC board mit PN532 über SPI anzusteuern. link to git repo

1. Verbinden des PN532 Chips auf dem NFC Breakout Board via SPi mit dem ESP32
    1. Wähle SPI mit dem Dipswitch am Board
    2. Anschlussplan: (sowohl VSPI und HSPI am ESP sind möglich)

ESP32 (HSPI)	ESP32(VSPI)	NFC Board
5V	5V	VCC
GND	GND	GND
GPIO 12	GPIO 19	MISO
GPIO 13	GPIO 23	MOSI
GPIO 14	GPIO 18	SCK
GPIO 15	GPIO 5	SS

2. Wenn der ESP mit einer Tacktrate von 240kHz betrieben wird, dann müssen in der Adafruit Bibliothek Änderungen vorgenommen werden, um den SPI Clock anzupassen. Dieser Bug ist aktuell in Version 1.2.2 noch nicht in der Bibliothek behoben, aber im Issue 80 des Git Repos wird die Fehlerbehebung beschrieben Link zum issue auf git.Link zur geänderten Lib Die angepasste Bibliothek kann auch diesem Tutorial entnommen werden (Link zum Ordner).

    Änderungen in der Adafruit_PN532.cpp ...

    1. Am Anfang der Datei ein Definitionsabschnitt

```
#ifdef ESP32
#define SPI_FREQUENCY 100000
#else
#define SPI_FREQUENCY 1000000
#endif
```

2. In der Methode

```
Adafruit_PN532::Adafruit_PN532(uint8_t ss) {
  spi_dev =
      new Adafruit_SPIDevice(ss, SPI_FREQUENCY, SPI_BITORDER_LSBFIRST,
SPI_MODE0);
}
```

3. In der Methode

```
Adafruit_PN532::Adafruit_PN532(uint8_t clk, uint8_t miso, uint8_t mosi,
uint8_t ss) {
  spi_dev = new Adafruit_SPIDevice(ss, clk, miso, mosi, SPI_FREQUENCY,
SPI_BITORDER_LSBFIRST, SPI_MODE0);
}
```

3. Kontrollieren der Verkabelung und der Bibliothek durch Übersetzen und Ausführen des readMifare Beispiels der Bibliothek example/readMifare.ino .

4. Das Programm zur NFC TAG Emulation im Detail
   Wie bereits zu Beginn erwähnt, wird nur ein einfacher Tag mit einzelnen HEX Werten emuliert ohne die Verwendung des NDEF Formates. Der gesamte Code zum Programm kann im Ordner 00_Code/NFC_Rally_Tag/NFC_Rally_Tag.ino eingesehen werden.

   Hier die wichtigsten Codeausschnitte mit Kommentaren:

```
...
//Buffer for the recived write Content
uint8_t apdubuffer[255] = {}, apdulen;

//the content buffer to tell the Tag reader, that it is an ESP for this
application
uint8_t ppse[] = {0x8E, 0x52, 0x41, 0x4c, 0x4c, 0x59, 0x5f, 0x45, 0x53,
0x50}; //RALLY_ESP in ascii

void setup(void) {
  ...

  // Set the max number of retry attempts to read from a card
  // This prevents us from waiting forever for a card, which is
  // the default behaviour of the PN532.
  nfc.setPassiveActivationRetries(0xFF);

  // configure board for mifare
  nfc.SAMConfig();
  ...
}
```

```
void loop(void) {
  //send the comand to the PN532 to run in emulate mode
  nfc.AsTarget();

  // set the data for the read process
  nfc.setDataTarget(ppse, sizeof(ppse));   // tell the reader that it is as
Rally ESP
  // get the data from the write process
  nfc.getDataTarget(apdubuffer, &apdulen); // get the topic back

  //if data sent to the tag print it on the serial monitor
  if (apdulen>0){
    for (uint8_t i = 0; i < apdulen; i++){
      Serial.print(" 0x"); Serial.print(apdubuffer[i], HEX);
    }
    ...
  }
}
```

5. Das Programm zum Lesen und Schreiben des NFC TAGs im Detail
   Der gesamte Code zum Programm kann im Ordner
   00_Code/NFC_Rally_Tag/Test_NFC_Reader_Writer.ino eingesehen werden.

   Hier die wichtigsten Codeausschnitte mit Kommentaren:

```
//uniq id counter for the registrated Tags
uint8_t id = 0;
//the Prefix in front of the id for uniq target topics
String topicPrefix;
//the expectet conntent that will be recived from an vaid Tag
uint8_t target_ppse[] = {0x8E, 0x52, 0x41, 0x4c, 0x4c, 0x59, 0x5f, 0x45,
0x53, 0x50}; // RALLY_ESP in ascii

void setup()
{
  ...
}

void loop() {
  ...

  // set shield to inListPassiveTarget
  success = nfc.inListPassiveTarget();
  if(success) {
    //a target tag is registrated

    //generate the next qunic topic
    String topic = topicPrefix + "_" + id;

    //convert in asci array
    uint8_t apdu[topicLen];
```

```
    for(int i = 0; i<topicLen;i++)
    {
      apdu[i]= topic.charAt(i);
      ...
    }
    ...

    //do the Data exchange with the target
    success = nfc.inDataExchange(apdu, apduLength, response,
&responseLength);
    if (success)
    {
      //data revived
      ...

      if(success)
      {
        //data vaid
        ...
        //id increment for next Target
        id++;
      }
      ...
    }
  }
}
```

6. Test der NFC Anwendung indem beide Programme ausgeführt werden.(Tag und Read_Write)
   1. Starten beider Programme mit seriellem Monitor. Entweder wird der serielle Monitor der Arduino IDE verwendet oder als alternative die serielle Console von Putty.
   2. Bei der NFC Lese- und Schreibeanwendung muss zuerst über die serielle Eingabe ein Prefix für die Topics übergeben werden. Beispielsweise "car"
   3. Wenn nun die beiden NFC Boards voreinander gehalten werden, sollten die Daten ausgetauscht werden und man sollte die folgenden Ergebnisse am Monitor dargestellt bekommen.

   Ausgabe des seriellen Monitors des emulierenden Programms:

```
Application: Rally ESP
Found chip PN532
Firmware ver. 1.6
Init finished
0x63 0x61 0x72 0x5F 0x30 #The Recived Toppic vrom the Reader in HEX
get the topic: car_0 from the Rally Master
```

Ausgabe des seriellen Monitors des Lese-und Schreibeprogramms:

```
Application: to wait for an ESP tag and give it an uniq toppic in this Rally
APP
Found chip PN532
Firmware ver. 1.6
Wait for topic prefix input..
recived the Toppic Prefix:
car #the input via the serial monitor for the prefix
Init finished
Waiting for Rally ESP
...
Tag number: 1
currentToppic:car_0
topic len: 5
63 61 72 5F 30 the sendet HEX value to the tag
get response 52 41 4C 4C 59 5F 45 53 50  RALLY_ESP
ESP device sucessvuly detected. And Topic name transimittet
created Topic car_0
...
```

7. Der Code zum Emulieren des Tages ist nicht nur in diesem Testprogramm verwendet worden, sondern auch im Code des gesamten RallyESPs. Das Programm findet man im fogenden Ordner 0100_Rally_ESP_Code.

Weitere Informationen

NFC Board mit PN532 am Raspberry pi verwenden

Aufgrund der Diskrepanz zwischen Anwendung und verfügbaren Bibliotheken, kommt diese Art der Verbindung nicht zur Anwendung. Es kann jedoch für andere Anwedungen interessant sein, deshalb wird hier kurz darauf eingegangen, wie man das NFC board am Raspberry betreiben kann.

1. Aktivieren der I2C Schnittstelle in der Raspberry Konfiguration
2. Anschließen des PN532 Chips I2c zum Pi

Raspbberry Pi	NFC
5V(Pin 4)	VCC
GND(Pin 14)	GND
GPIO2(pin 3)	SDA
GPIO3(pin 5)	SCL

3. I2C am DIP Schalter auswählen
4. Scannen der ISC Geräte am ICC Buss durch folgenden Terminal aufrufen

```
sudo i2cdetect -y 1
```

In meinem Aufbau erhalte ich die Adresse 0x24 für den PN532 Chip, welche nun im weiteren Verlauf verwendet wird.

Verwenden von libNFC

Installieren der Tools -libnfc5 -libnfc-bin -libnfc-examples

```
sudo apt install libnfc5 libnfc-bin libnfc-examples
```

Konfigurieren der Tools über die Datei etc/nfc/libnfc.conf

```
sudo nano /etc/nfc/libnfc.conf
# uncoment and change the last two lines
device.name = "PN532 I2C"
device.connstring = "pn532_i2c:/dev/i2c-1"
```

Testen der Konfiguration durch den folgenden Aufruf

```
nfc-scan-device -v
```

Wenn alles funktioniert, erhält man die Ausgabe

```
...
1 NFC device(s) found :...
...
```

Verwenden mit Python

link zum git repo

```
pip3 install adafruit-circuitpython-pn532
```

Testen der Bibliothek durch Ausführen des mitgelieferten Beispiels examples/pn532_simpletest.py .

Quellen

[1] nfc-forum.org

[2] werner.rothschopf.net

[3] www.pcwelt.de

[4] blog.stigok.com