PaSoRiセットアップ資料(2024.12.22)

この資料は以下のサイトに適宜脚色をつけたものである

https://obenkyolab.com/?p=741#google_vignette

英語が読めるのであれば以下の資料の方が情報が最新である可能性が高い

https://nfcpy.readthedocs.io/en/latest/topics/get-started.html

【Python】PaSoRiによるICカードの読み取り

BY SEBONE · 公開済み 2019年8月13日 · 更新済み 2022年10月3日

2022/04/16追記：使用したlibusbのVersion記載

2022/10/03追記：Windows11環境での再検証

Table of Contents

開発環境
・Windows 64bit
・Python 3.12.8
・libsub v1.0.26
※最新v1.0.27をダウンロードするとこの資料の情報とフォルダの中身がちょこちょこ違います。

2022/10/3の最新の検証環境

- Windows11 64bit 21H2
- python 3.8.10
- PaSoRi(RC-S380)
- libusb v1.0.26

以下は過去の検証環境です

- Windows10 64bit
- python3.7(Anaconda3)
- PaSoRi(RC-S380)

pythonにはnfcpyというnfcデバイスの制御ができる、素晴らしいライブラリがあります。今回はこのnfcpyを使用します。

以下のnfcpyの説明サイトで詳しく書かれていますので、これを参考に進めたいと思います。
https://nfcpy.readthedocs.io/en/latest/topics/get-started.html

簡単に言うと、次のような順番で実施します。

1. Zadigというツールを使いWinUSBをインストールする。
2. libusbをインストールする。
3. pythonでnfcpyを使ってコーディングする。

※NFCポートソフトウェアがインストールされていると動作しないことを確認している

NFCポートソフトウェアリムーバーで削除する

https://www.sony.co.jp/Products/felica/consumer/support/download/NFCPortSoftwareRemover.html

WinUSBとはMicrosoftが提供するUSBドライバらしく、おそらく、PaSoRiなどのリーダのIO制御に必要なのかと思います。そして、ZadigはそのWinUSBを簡単にインストールできるGUIベースのツールのようです。
libusbはUSBデバイスへアクセスするためのCのライブラリで、おそらくnfcpyを使う前提として必要なのだと思います。

ということで、まずはWinUSBとlibusbを準備した上で、pythonのnfcpyを使ってコーディングをしていきたいと思います。

Step1 : WinUSBのインストール

以下のZadigのサイトからZadigの最新版をダウンロードします。

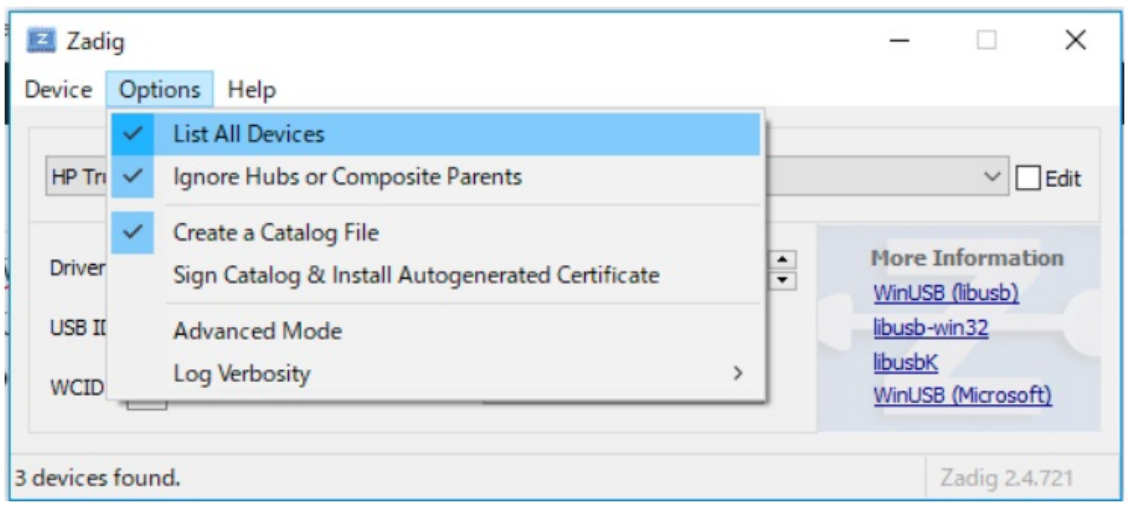
2022/10/3追記

以前はzadig-2.4を使用しましたが、最新の検証環境ではzadig-2.7を使用しました。

今回はzadig-2.4をダウンロードしました。

zadig-2.4.exeをダウンロードして任意の場所に保存した後、右クリック から 管理者として実行を選択します。

すると、GUIが開くので、以下のようにOptions > List All Devicesにチェックを入れます。

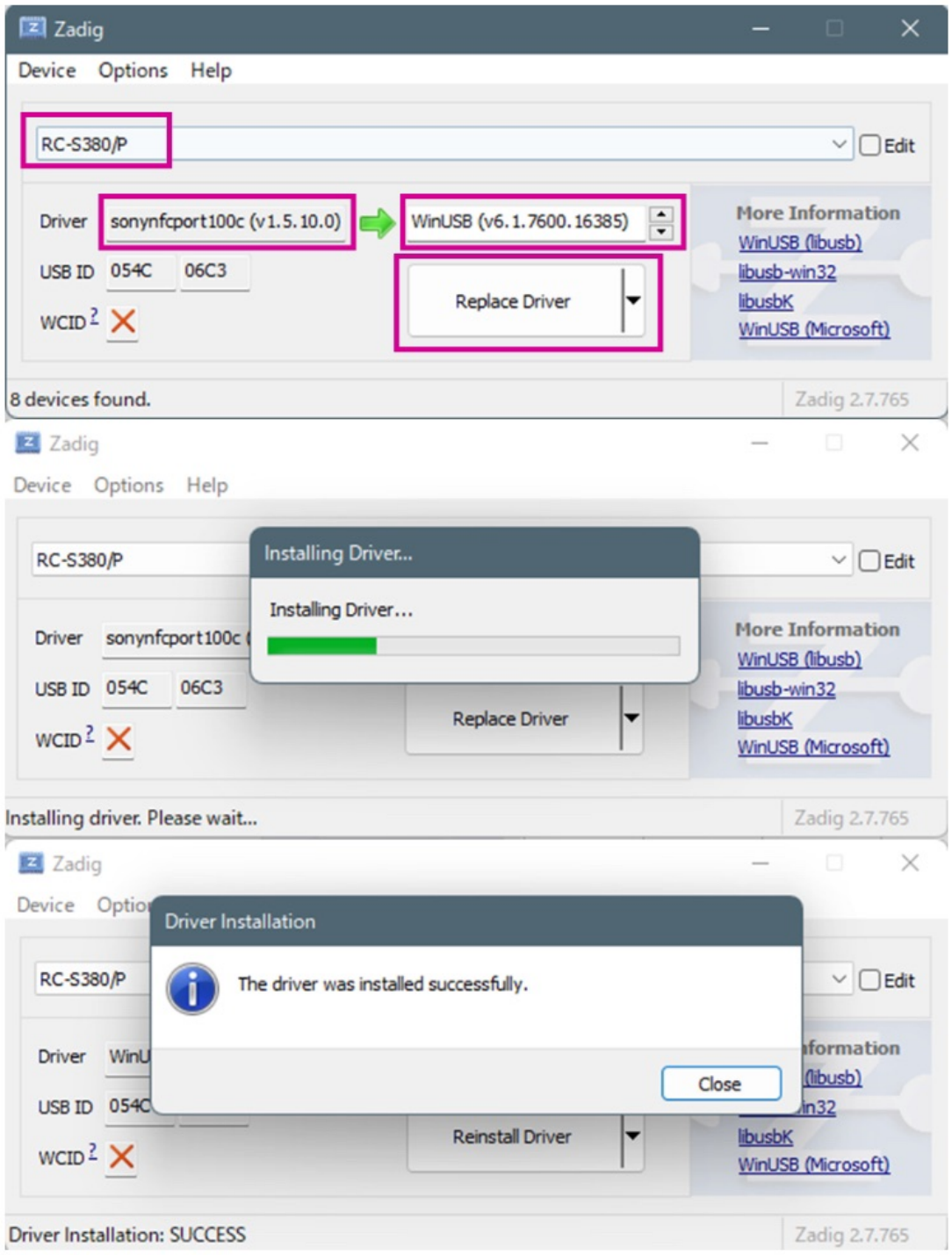


次に、**PCのUSB端子にPaSoRi(RC-S380)を接続し**、下図のようにプルダウンメニューからNFC Port/PaSoRi 100 USBを選択します。PaSoriを接続した状態にしないと選択しとして出てこない場合があります。

2022/10/3再検証

zadig-2.7を使用した場合、以下のようにRC-S380/Pを選ぶと、Driverとしてsonynfcport100c(v1.5.10.0) → WinUSB(v6.1.7600.16385)の組み合わせを選ぶことができました。

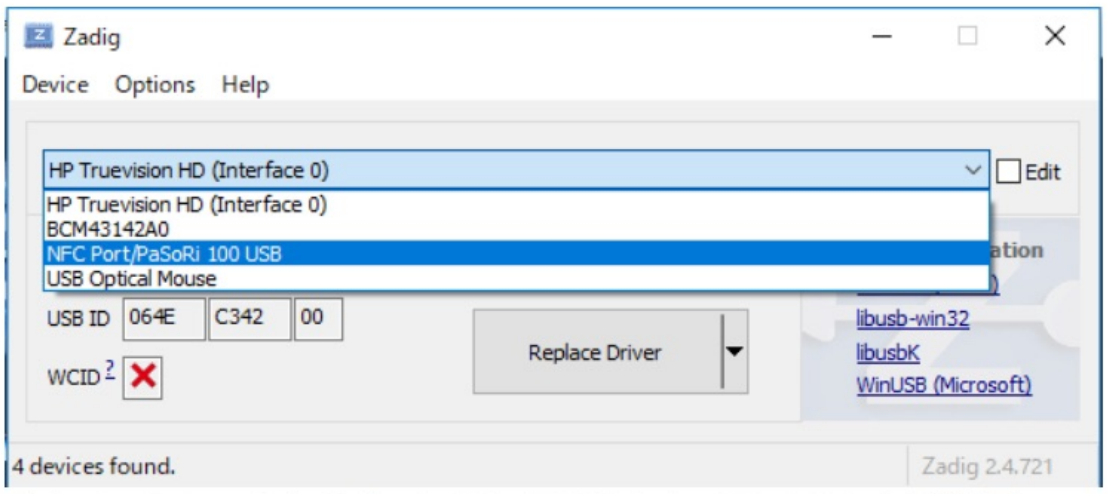
この組み合わせでReplace Driver （あるいはInstall Driver）を選択すると、インストールが始まり、完了できました。



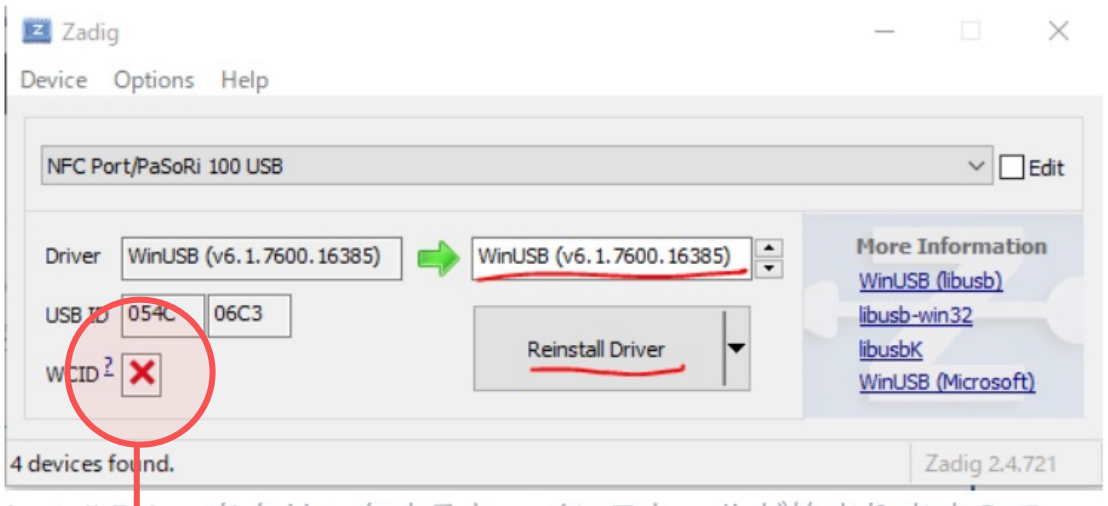
この通り進めて良い

10分ほどかかる

以下は旧環境の検証結果



そして、Driverのところで、WinUSB(v6.1.7600.16385)を選択し、Install Driverをボタンを押します。下図は一旦インストールした際に撮ったスクショのため、Replace Driverになってますが、初回はInstall Driverになっているはずです。



Install Driverをクリックすると、インストールが始まりますので、完了するまで待ちます。以上でWinUSBのインストールは完了です。

バツで問題ない

Step2 : libusbのインストール

以下のサイトからlibusbをダウンロードします。ページの Downloads > Lates Windows Binariesを選択すると、ダウンロードが開始され、7zの拡張子の圧縮ファイルが落ちてきます。

https://libusb.info/

ダウンロードしたファイルを任意の場所へ保存したら、7zipを使って解凍します。

圧縮先に7-zipが見当たらない時は
その他のオプションを確認>7zip
で選択できる

2022/10/3追記

再検証時はlibusb v1.0.26を使用しました。

今回使用したのはlibusb 1.0.22です。

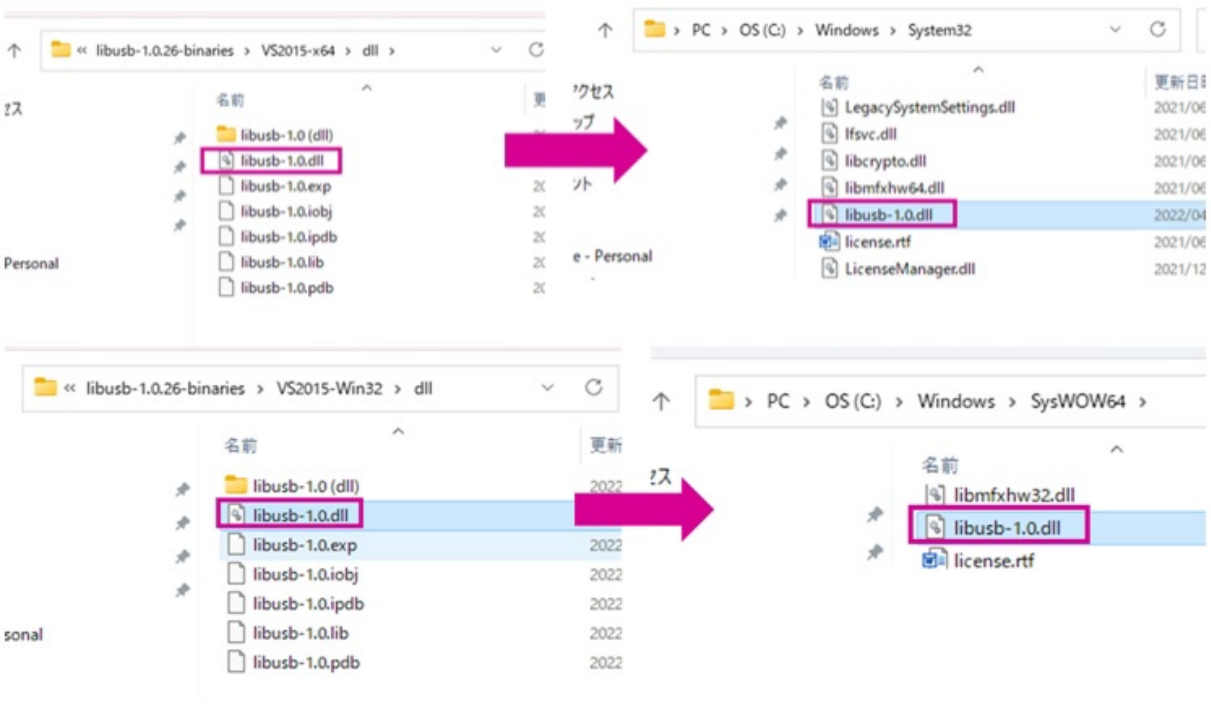
解凍して生成されたフォルダの中から以下の２つのファイルを指定の場所にコピーします。この作業を忘れるとPaSoRiは動きませんので、注意してください。

存在しない

2022/10/3再検証

1. ./VS2015-x64\dll\libusb-1.0.dllを C:\Windows\System32 へコピーする。
2. ./VS2015-x64\dll\libusb-1.0.dll を C:\Windows\SysWOW64 へコピーする。

32



X61,X32ではなく
MS64,MS32 フォルダを使う

- For 32-bit Windows:
  - Copy `MS32\dll\libusb-1.0.dll` to `C:\Windows\System32`.
- For 64-bit Windows:
  - Copy `MS64\dll\libusb-1.0.dll` to `C:\Windows\System32`.
  - Copy `MS32\dll\libusb-1.0.dll` to `C:\Windows\SysWOW64`.

https://nfcpy.readthedocs.io/en/latest/topics/get-started.html

---

旧環境検証時

1. MS64dll\libusb-1.0.dllを C:\Windows\System32 へコピーする。
2. MS32\dll\libusb-1.0.dll を C:\Windows\SysWOW64 へコピーする。

Step3 : Pythonによる実装

2022/10/3追記

再検証した環境はAnacondaではなく純粋な Pythonの環境にて実施しました。以下で作成した環境になります。

※ CygwinやVS codeのターミナルでは動作しないことを確認済み

旧環境の検証時

まず、Anaconda promptを立ち上げてnfcpyをインストールします。

次に、PaSoRiの上にICカードを載せます。

そして、以下のコードを実行します。

Windows PowerShell で実行する

```
1  import nfc
2
3  #接続定義
4  clf = nfc.ContactlessFrontend('usb')
5  print(clf)
6
7  #タグの取得
8  tag = clf.connect(rdwr={'on-connect': lambda tag: Fa
9
10 #結果表示
11 print(tag)
12 print(dir(tag))
```

```
SONY RC-S380/P on usb:002:001
Type3Tag 'FeliCa Standard (RC-S???)' ID=xxxxxx
PMM=xxxxxx SYS=xxxxxx
['IC_CODE_MAP', 'NDEF', 'TYPE', 'class', 'delattr',
'dict', 'dir', 'doc', 'eq', 'format', 'ge', 'getattribute', 'gt',
'hash', 'init', 'init_subclass', 'le', 'lt', 'module', 'ne',
'new', 'reduce', 'reduce_ex', 'repr', 'setattr', 'sizeof',
'str', 'subclasshook', 'weakref', '_authenticated', '_clf',
'_format', '_is_present', '_ndef', '_nfcid', '_product',
'_target', 'authenticate', 'clf', 'dump', 'dump_service',
'format', 'identifier', 'idm', 'is_authenticated', 'is_present',
'ndef', 'pmm', 'polling', 'product', 'protect',
'read_from_ndef_service', 'read_without_encryption',
'request_response', 'request_service',
'request_system_code', 'search_service_code',
'send_cmd_recv_rsp', 'sys', 'target', 'type',
'write_to_ndef_service', 'write_without_encryption']
```

ICに用意されているタグが簡単にわかりました。
あとは、tag.dumpすると、実際のデータを見ることができます。
(途中割愛しています)

```
1  tag.dump()
```

```
'System 0003 (Suica)',
'Area 00
' Cyclic Service 36: write with key & read w/o key…,
' 0000: 16 01 00 …
|……..D…….|',
```

データを見ることはできましたが、データの意味まで理解するためには、そもそもどんな形でデータが入っているかの予備知識があったり、暗号化されていないことが前提となりそうです。

結果

python nfcpyを使うと、初心者でもデータの中身まで簡単に表示できました。
ただし、中身の意味を知ろうとした場合はICカードがどう運用されているのかの予備知識も必要そう。平文ベタ打ちなら何かしらはわかるかも。

以上