

RaspberryPi+Python (其他 x86/arm 架構 Linux 設備可參考)

環境：RaspberryPi 3 & 4 · Python 3.6+套件pyserial=3.4

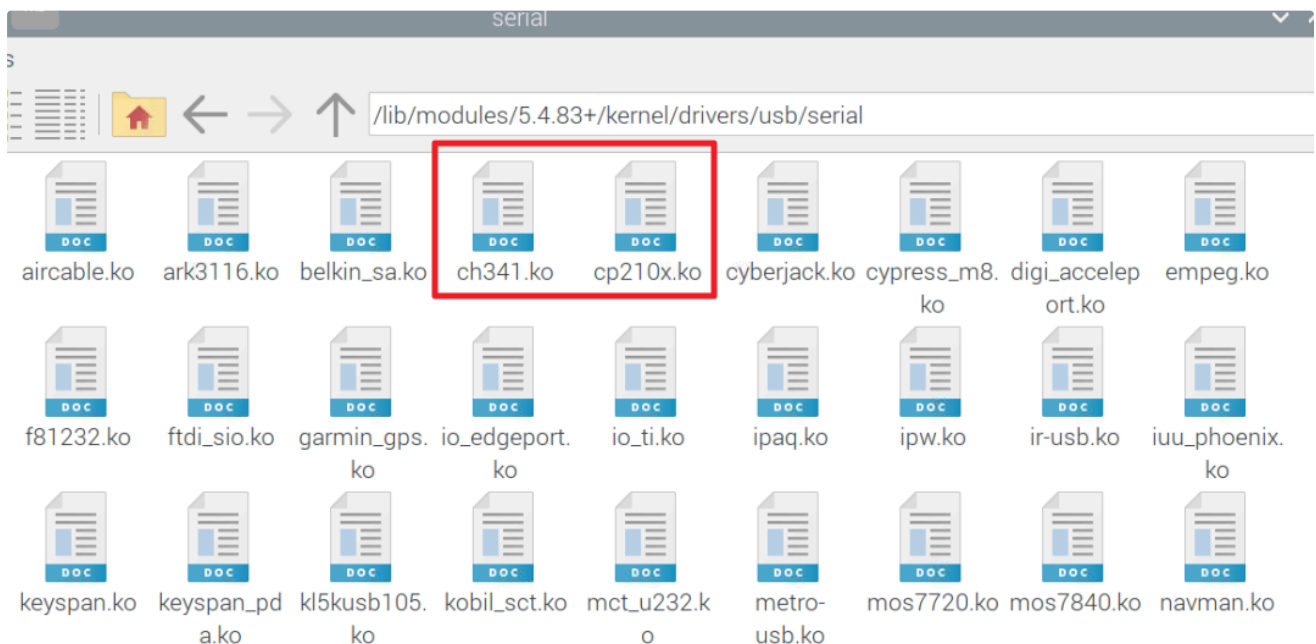
產品：以RS232(若要連接USB需自行搭配轉接器)、USB輸出的產品如 Hi221/Hi221Dongle/Hi221/Hi221/CH100/CH104/CH108

本教學僅供基本參考，更多的延伸或應用問題本司不提供支援

ANROTIMU使用的 Uart to USB 晶片，如 CP210x 與 CH340，近年 Linux kernel 都已內建驅動，

可以確認類似的路徑中 `/lib/modules/5.4.83+/kernel/drivers/usb/serial`，有沒有 `ch341.ko` & `cp210x.ko`，如下圖。

有的話比較能確定能夠讀取設備，但開發版種類多且未必整合完善，不保證沒有 Bug。



經測試並正常運行的設備：

- RaspberryPi 3 & 4
- NVIDIA XAVIER

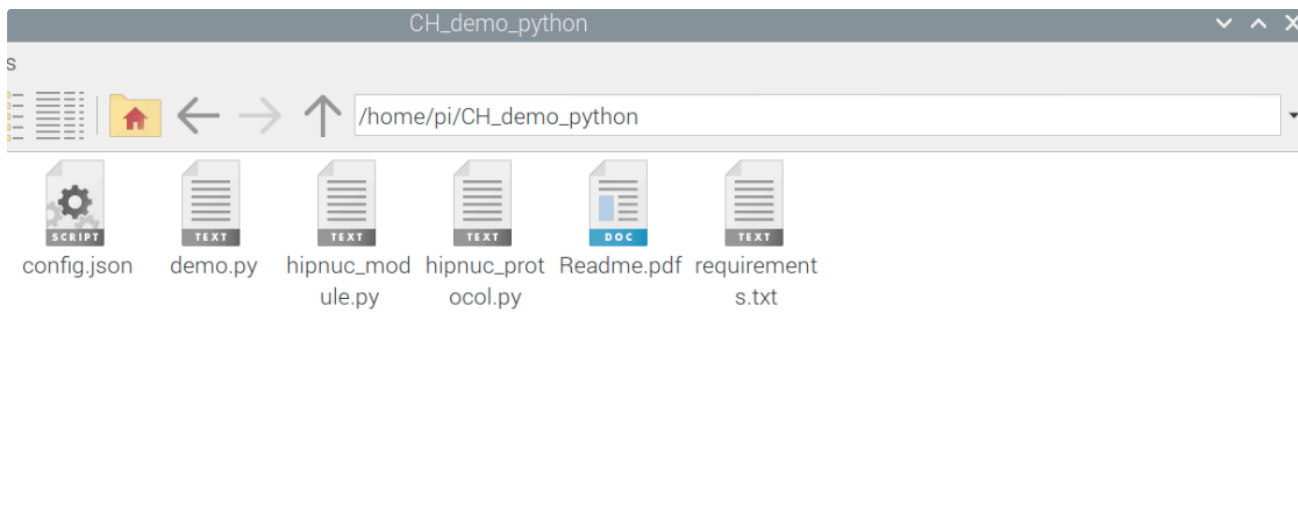
Step 1. 下載

瀏覽器下載並解壓：[anrot_demo_python.zip](#)

或：

```
1 cd ~
2 wget https://github.com/avmm9898/anrotimu_doc/raw/master/03_Examples/Python/anrot_demo_python.zip
3 unzip anrot_demo_python.zip
```

檔案如下：



Step 2. 連接與設定

將產品 USB 連接上 RPI，連接成功的話，在 `/dev` 下會多出 `tttyUSB0`（如果有連接其他 USB 設備，可能會是其他編號如 `tttyUSB3`）

可自行插拔，用以下指令確認多出來的是哪個設備：

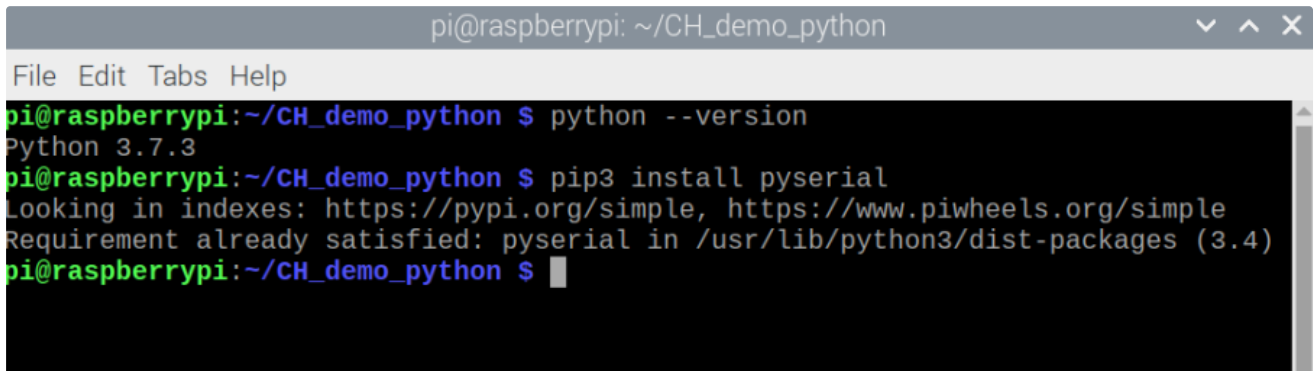
```
1 | ls /dev
```

回到下載的範例，任意文字編輯器打開 `config.json` 進行設定，設定 `port` 名稱、與設備匹配的 `baudrate`，如下圖：

（詳細設定請參考 Step 3）

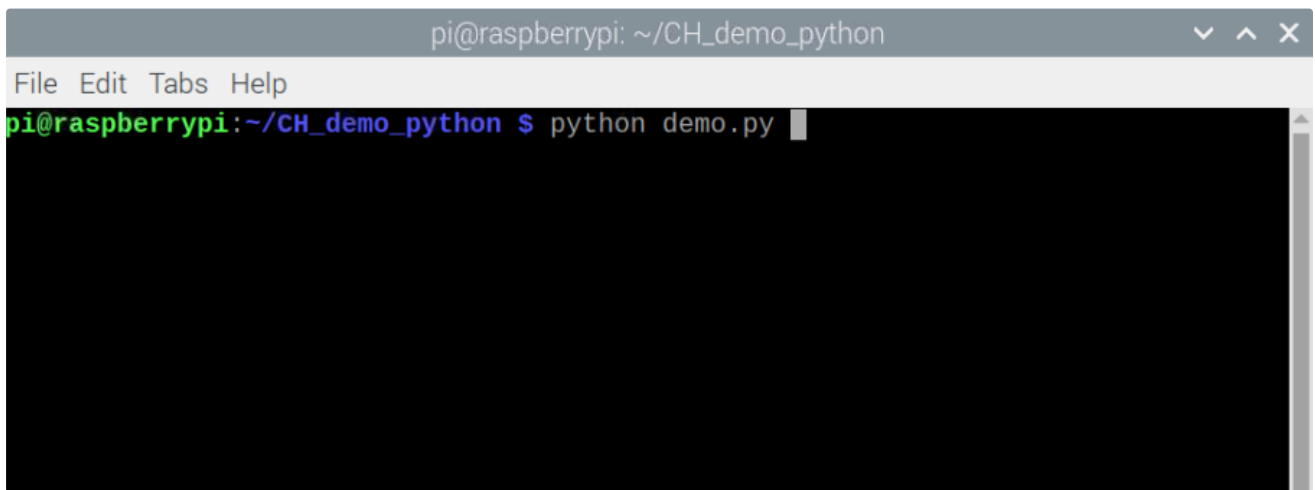
```
1  {
2      "port": "COM3",
3      "baudrate": 115200,
4      "report_datatype": {
5          "imusol": true,
6          "gwsol": true,
7          "id": false,
8          "acc": false,
9          "gyr": false,
10         "mag": false,
11         "euler": false,
12         "quat": false,
13         "imusol_raw": false,
14         "gwsol_raw": false
15     }
16 }
```

檢查 python 版本 · 安裝 pyserial :

A terminal window titled 'pi@raspberrypi: ~/CH_demo_python' with a menu bar (File, Edit, Tabs, Help). The terminal shows the following commands and output:

```
pi@raspberrypi:~/CH_demo_python $ python --version
Python 3.7.3
pi@raspberrypi:~/CH_demo_python $ pip3 install pyserial
Looking in indexes: https://pypi.org/simple, https://www.piwheels.org/simple
Requirement already satisfied: pyserial in /usr/lib/python3/dist-packages (3.4)
pi@raspberrypi:~/CH_demo_python $
```

然後於 terminal 執行 demo.py :

A terminal window titled 'pi@raspberrypi: ~/CH_demo_python' with a menu bar (File, Edit, Tabs, Help). The terminal shows the command to run a script:

```
pi@raspberrypi:~/CH_demo_python $ python demo.py
```

開始接收資料 :


```

23         #id, timestamp, acc, gyr, quat, mag, euler
24         print(data['euler'])
25     except:
26         print("Print error.")
27         m_IMU_serial.close()
28         break

```

API 與 config.json 說明

`class anrot_module(*path_configjson=None*)`

超核模組類，用於接收、處理超核模組資訊。

參數: `path_configjson (str)` - json 配置文件的路徑。

- `get_module_data(*timeout=None*)`
獲取已接收到的模組數據。
timeout - 可選參數。若為None(默認值)，將會阻塞直至有有效值；若timeout為正數，將會嘗試等待有效數據並阻塞
timeout 秒，若阻塞時間到，但仍未有有效數據，將會拋出Empty異常。
返回 : data
類型為 dict。字典的 key 為數據類型，value 為所有節點該數據的 list。**
例如 : 返回數據中，第1個模組的加速度數據為 data["acc"][0]；返回數據中，第16個模組的四元數資訊為 data["quat"][15]。
- `get_module_data_size()`
獲取已接收到的模組數據的數量。注意: 返回長度大於0, 不保證 `get_module_data` 時不會被阻塞。
參數: 無
返回: size - 返回模組數據，類型為字典
返回類型: int
- `close()`
關閉指定的模組。
參數: 無
返回: 無

json 文件配置說明

在初始化 `anrot_module` 時，需要傳入 json 配置文件的路徑

```

1 //config.json
2
3 {
4     "port": "COM8",
5     "baudrate": 115200,
6     "report_datatype": {
7         "imusol": true,
8         "gwsol": true,
9         "id": true,
10        "acc": true,
11        "gyr": true,
12        "mag": true,
13        "euler": true,
14        "quat": true,
15        "imusol_raw": false,
16        "gwsol_raw": false
17    }
18 }

```

配置如下：

1. 序列埠端口 "port" :

序列埠端口，類型為字符串。在Windows下為 "COM*"，例如 "COM11"；Linux下一般為 "/dev/tty*"，例如 "/dev/ttyUSB0"。請根據設備的實際情況進行配置。

2. 波特率(鮑率/Baud/baudrate) "baudrate" :

序列埠波特率請根據模組實際參數進行設置，為 115200/460800/921600，3種可能，115200 為單顆 IMU 適用，某些 Hi221 或 Hi221Dongle 客戶可能會設定為 460800 或 921600。可透過官方軟體 CHCenter 進行設定。

3. 數據類型 "report_datatype" :

匯報數據種類。模組將會回傳多種數據資訊，true 表示開啟解碼，false 表示禁止解碼。

- imusol : 0x91 單節點全資訊封包，Hi221/Hi221/Hi221/CH100/CH110/CH104/CH108 產品預設傳輸模式。
- gwsol : 0x62 多節點接收機全資訊封包，Hi221Dongle 預設傳輸模式。
- id : 0x90，單節點ID
- acc : 0xA0，單節點加速度
- gyr : 0xB0，單節點角速度
- mag : 0xC0，單節點磁力計
- euler : 0xD0，單節點歐拉角
- quat : 0xD1，單節點四元數
- imusol_raw : 0x93 單節點封包，僅保留加速度與角速度。
- gwsol_raw : 0x63 多節點接收機封包，僅保留加速度與角速度，可設定最高 4 節點 400Hz。

模組傳輸內容示範

Hi221/Hi226/Hi229/CH100/CH110/CH104/CH108 預設傳輸模式 (0x91協議)

產品回傳內容為：

```
1 temp_dic = {
2     "id":id_temp_list,
3     "timestamp":timestamp_temp_list,
4     "acc":acc_temp_list,
5     "gyr":gyr_temp_list,
6     "mag":mag_temp_list,
7     "euler":int_eul_temp_list,
8     "quat":quat_temp_list
9 }
10 return temp_dic
```

一幀 data 內含以下資料：`id`，`timestamp`，`acc`，`gyr`，`mag`，`euler`，`quat`

- **id**: 節點 ID
- **timestamp** : 自開機起傳送的幀數
- **acc** : 加速度，順序為 X Y Z 軸，單位=1G (1G = 1x重力加速度)
- **gyr** : 角速度，順序為 X Y Z 軸，單位=deg/s
- **mag** : 磁場強度，順序為 X Y Z 軸，單位=μT (10⁻⁶ T)
- **euler** : 歐拉角，順序為 橫滾(X-Roll)/俯仰(Y-Pitch)/航向(Z-Yaw)，單位=degree
- **quat** : 四元數，順序為 W X Y Z 軸

因此，如果要取得 **acc**：

```
1 print(data['acc'])
```

Hi221Dongle 無線接收器預設傳輸模式（0x62協議）

產品回傳內容為：

```
1 temp_dic = {
2     "GWD":gwid,
3     "CNT":cnt,
4     "id":id_temp_list,
5     "timestamp":timestamp_temp_list,
6     "acc":acc_temp_list,
7     "gyr":gyr_temp_list,
8     "mag":mag_temp_list,
9     "euler":eul_temp_list,
10    "quat":quat_temp_list
11 }
12 return temp_dic
```

一幀 data 內含以下資料：**GWD**, **CNT**, **id**, **timestamp**, **acc**, **gyr**, **mag**, **euler**, **quat**

- **GWD**：無線接收機 ID（GWID）
- **CNT**：在線的無線節點（Hi221）數量
- **id**：節點 ID
- **timestamp**：自開機起傳送的幀數
- **acc**：加速度，順序為 X Y Z 軸，單位=1G（1G = 1x重力加速度）
- **gyr**：角速度，順序為 X Y Z 軸，單位=°/s
- **mag**：磁場強度，順序為 X Y Z 軸，單位=μT（10⁻⁶ T）
- **euler**：歐拉角，順序為 橫滾(X-Roll)/俯仰(Y-Pitch)/航向(Z-Yaw)，單位=degree
- **quat**：四元數，順序為 W X Y Z 軸

因此，如果要取得 **euler**：

```
1 print(data['euler'])
```

若發現數據傳輸失敗或迴圈中斷：

請檢查並重新設定：

- 數據協議是否匹配（0x91 或 0xA0 等等）
- Baud 是否與模組匹配
- port 名稱是否與模組匹配