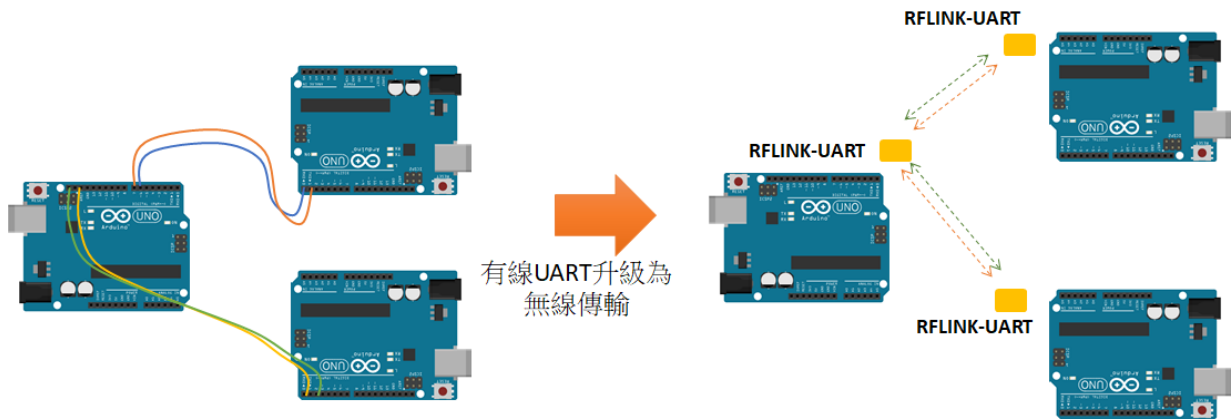


# RFLINK-UART

## 解放 UART

### 立即無痛升級無線傳輸



## 目錄

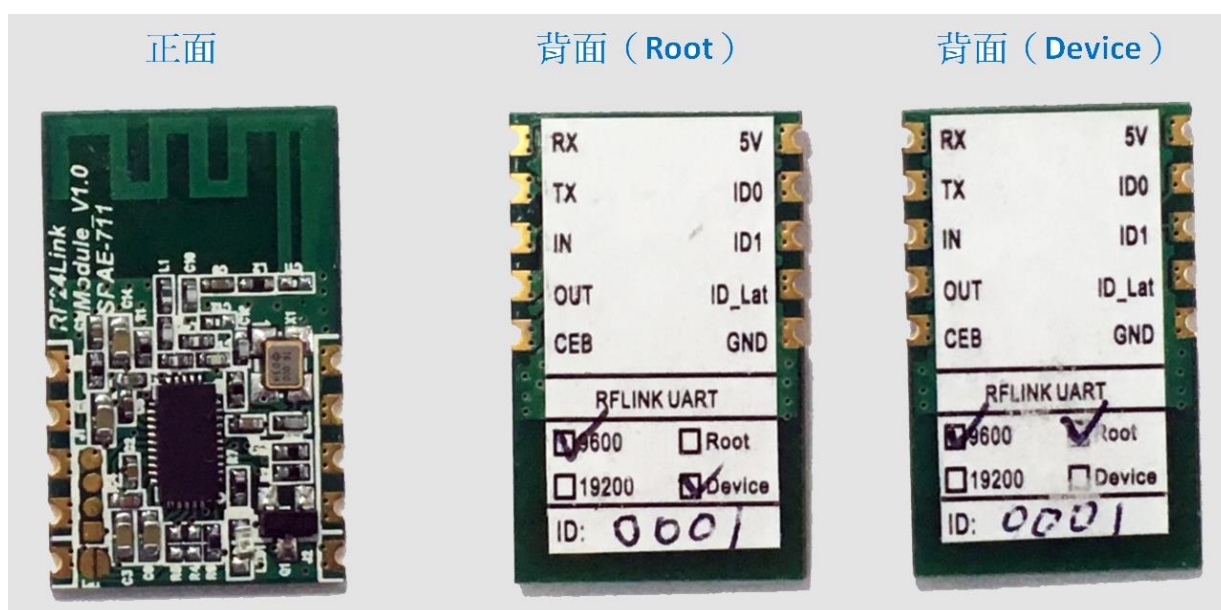
模組外觀與尺寸 .....	2
模組特性 .....	2
Pin 腳定義 .....	3
如何使用 .....	3
設定欲連接的 Device .....	3
與新連接端開始傳送／接收資訊 .....	5
與 Arduino 搭配使用 .....	5
Root 端傳送端程式範例： .....	6
RX 接收端程式範例： .....	7
執行 .....	8
與 Raspberry Pi 樹莓派搭配使用 .....	8
程式範例： .....	9
直接對接感測器 .....	11
使用 IO Ports .....	12

RFLINK-UART 無線 Uart 傳輸模組是一款簡單易用的模組，它能將有線的 UART 立即無痛升級為無線 UAR 傳輸，此外還提供一組 I/O pin，讓您不需要額外 coding 及硬體及就有可相互遙控的 IO 開關來使用。

## 模組外觀與尺寸

RFLINK-UART 模組包含 Root 端（左側）一片，以及最多四片的 Device 端（如下圖右側，編號 1~4），兩者外觀雖相同，但可由背面的標籤來辨識 Root 或 Device 是否勾選來辨識。

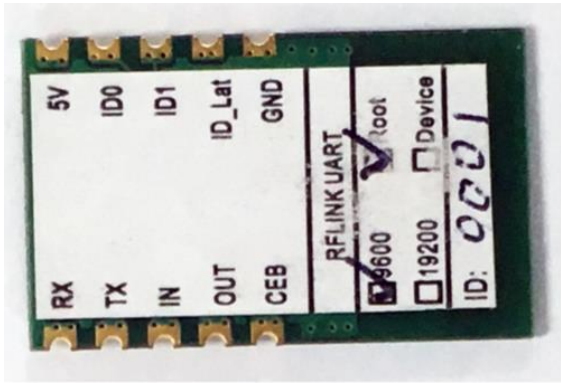
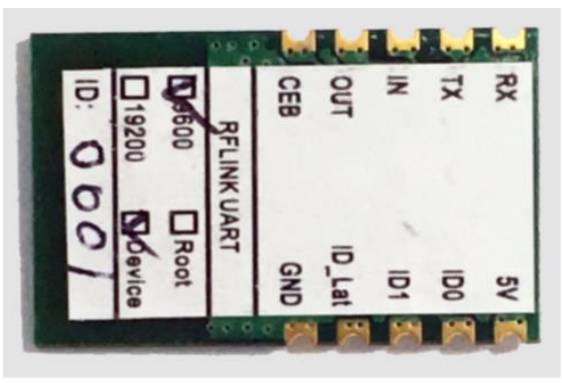
如下圖，該組 RFLINK-UART 模組的 Group ID 為 0001，Baud rate 9600。



## 模組特性

1. 操作電壓：3.3~5.5V
2. RF 頻率：2400MHz~2480MHz。
3. 耗電量：傳送約 24mA@+5dBm，接收約 23mA。
4. 發射功率：+5dBm
5. 傳輸速率：250Kbps
6. 傳輸距離：空曠處約 80~100m
7. Baud rate：9,600bps 或 19,200bps
8. 尺寸：25 mm x 15 mm x 2 mm (LxWxH)
9. 可支援 1 對 1 或 1 對多（最多四個）的傳輸。

## Pin 腳定義

Root	Device
 <p>The image shows the Root module with a white label. The label has two columns of pins. The top row contains 5V, ID0, ID1, ID_Lat, and GND. The bottom row contains RX, TX, IN, OUT, and CEB. In the center, there is a section for 'RFLINK UART' with checkboxes for 'Root' (checked) and 'Device'. Below this are checkboxes for '6600' and '19200'. At the bottom, there is a handwritten 'ID: 0001'.</p>	 <p>The image shows the Device module with a white label. The label has two columns of pins. The top row contains RX, TX, IN, OUT, and CEB. The bottom row contains GND, ID_Lat, ID1, ID0, and 5V. In the center, there is a section for 'RFLINK UART' with checkboxes for 'Root' and 'Device' (checked). Below this are checkboxes for '6600' and '19200'. At the bottom, there is a handwritten 'ID: 0001'.</p>
<p><b>GND</b> → Ground  <b>+5V</b> → 5V 電壓輸入  <b>TX</b> → 對應到開發板 Uart 的 RX  <b>RX</b> → 對應到開發板 Uart 的 TX  <b>CEB</b> → CEB pin 腳需接地(GND)模組才會通電運作，可作為省電控制功能使用。  <b>OUT</b> → IO Port 的輸出 pin (On/Off 輸出)  <b>IN</b> → IO Port 的輸入 pin (On/Off 接收)  <b>ID1, ID0</b> → 透過此兩 pin 腳的 HIGH/LOW 組合來選擇與那一片 device 連接。  <b>ID_Lat</b> → 正式切換 pin。當 Root 透過 ID0, ID1 設定好欲連接的 device ID 之後，需針對此 pin 腳輸入 LOW 才會正式切換至指定的 device ID。</p>	<p><b>GND</b> → Ground  <b>+5V</b> → 5V 電壓輸入  <b>TX</b> → 對應到開發板 Uart 的 RX  <b>RX</b> → 對應到開發板 Uart 的 TX  <b>CEB</b> → CEB pin 腳需接地(GND)模組才會通電運作，可作為省電控制功能使用。  <b>OUT</b> → IO Port 的輸出 pin (On/Off 輸出)  <b>IN</b> → IO Port 的輸入 pin (On/Off 接收)  <b>ID1, ID0</b> → 透過此兩 pin 腳的 HIGH/LOW 組合，Device 可設定為不同的裝置編號。  <b>ID_Lat</b> → 此 Pin 腳在 Device 並無作用。</p>

## 如何使用

凡是支援 UART 通訊介面的各類開發板及 MCU 皆可直接使用本模組，不需要安裝額外的 driver 或 API 程式。

### 設定欲連接的 Device

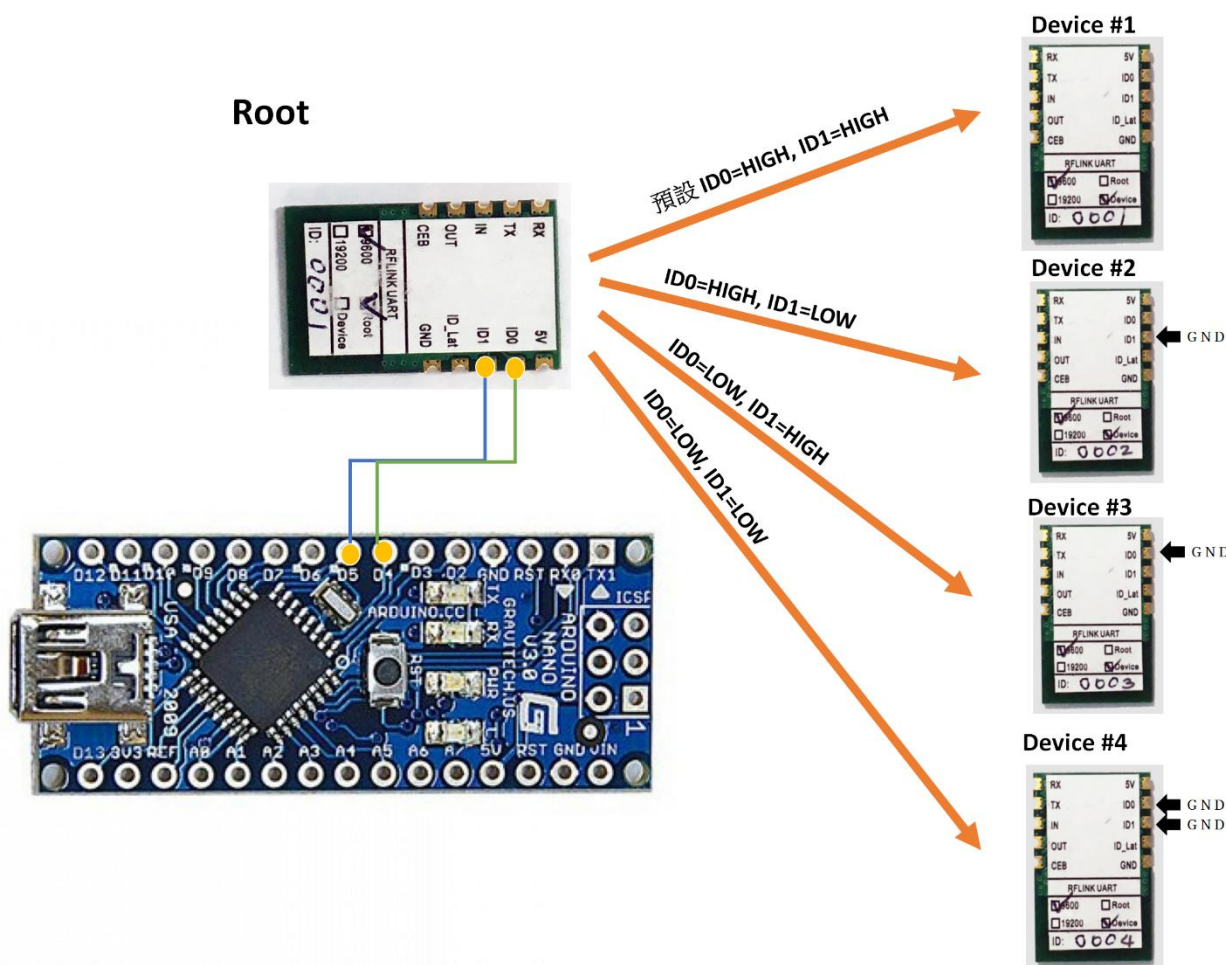
傳統有線 TTL 為 1 對 1 方式傳輸，本 RFLINK-UART 無線 Uart 傳輸模組額外支援 1 對多模式，預設 Root 端 (#0) 通電後與 Device (#1) 連接，如果您有其它編號的 Device

(#2~#4)，可在 Root 端針對 ID0, ID1 pin 送出不同的 HIGH/LOW 組合來選擇欲連接的 device 編號，Root 端的 ID0, ID1 編號選擇組合請參考下方表格。

	Device 1 (#1)	Device 2 (#2)	Device 3 (#3)	Device 4 (#4)
ID0 pin	HIGH	HIGH	LOW	LOW
ID1 pin	HIGH	LOW	HIGH	LOW

ID0, ID1 pin 預設為 HIGH，若接到 GND 則為 LOW  
 注意：Device 端需先依上表設定為需要的 Device ID，Root 端也是依同樣的方式來連接到指定 ID。

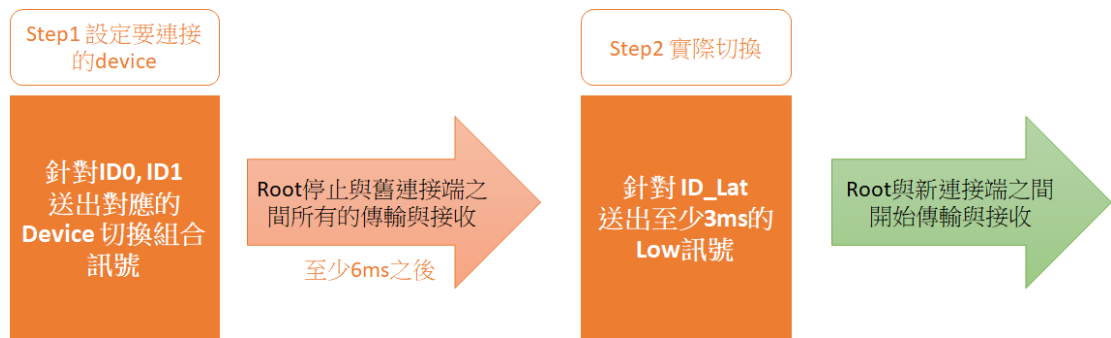
您可以參考上方的對應表格，先將各個 device 模組端的 ID0, ID1 pin 腳接到 GND 設定為不同的 device ID，接著再從 Root 模組端將 GND 線路接到 ID0 或 ID1，便可讓 Root 端與指定 Device 相連接。Root 端也可以透過開發板的 pin 腳位，透過程式送出 LOW/HIGH 訊號來動態指定要連接的 Device。例如下圖，Arduino Nano 透過 D4 與 D5 pin 來選擇要連接的 Device。



在針對 ID0, ID1 腳位送出對應的 High/Low 訊號之後，此時 Root 端將中斷與舊連接端的傳輸（亦即停止與舊連接端的傳送及接收），並等待來自 ID\_Lat 腳位的 Low 訊號以便進行切換到新連接端。

## 與新連接端開始傳送 / 接收資訊

在您透過 ID0, ID1 送出欲連接的 device 對象後，此時 Root 端與舊連接端之間會停止所有的傳送與接收動作，待您針對 ID\_Lat 腳位送出一個至少 3ms 的 Low 訊號，才會與新連接端開始傳送及接收資訊。



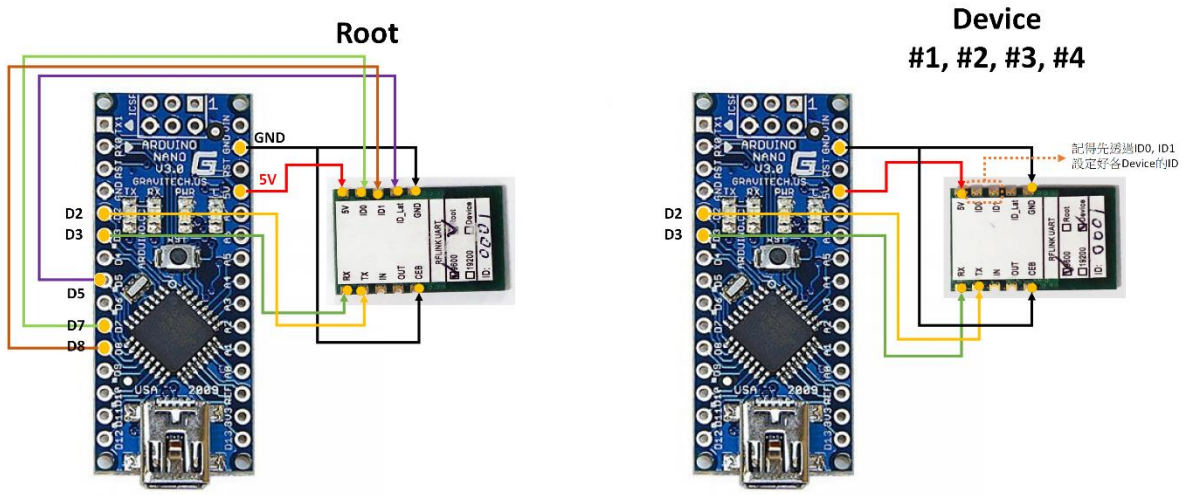
以下分別以 Android、樹莓派兩種開發板以及感測器直接對接的三種方式為例來說明：

## 與 Arduino 搭配使用

除了直接使用 Arduino 的 hardware TX/RX ports 之外，本模組也支援 software serial，因此能以軟體模擬 Uart 方式來使用以避免佔用實體的 UART 介面。

下圖範例為透過 software serial，分別連接 D2 及 D3 到 RFLINK-UART 模組 Root 端的 TX 及 RX，D7, D8 為設定連接 device 的腳位，D5 則作為確定切換腳位。透過 Arduino 的指令 digitalWrite 針對 D7, D8 以及 D5 腳位輸出 LOW 或 HIGH，我們就能達到動態連接到不同 device 的功能。





Arduino	D2	D3	D5	D7	D8	5V	GND
RFLINK- UART	RX	TX	ID_Lat (Root)	ID0 (Root)	ID1 (Root)	5V	GND CEB

**Root 端傳送端程式範例：**

```

#include <SoftwareSerial.h>
SoftwareSerial mySerial(2, 3); // RX, TX

#define pinLat 5
#define pinID0 7
#define pinID1 8

void setup() {
  pinMode(pinLat, OUTPUT);
  pinMode(pinID0, OUTPUT);
  pinMode(pinID1, OUTPUT);

  Serial.begin(9600);
  mySerial.begin(9600);
}

void loop() {
  #ID0=LOW, ID=HIGH, 表示要連接 Device #3
  digitalWrite(pinID0, LOW);
  digitalWrite(pinID1, HIGH);
  #等待 3ms 秒

```

```

delay(3)
#確定連接
digitalWrite(pinLat, LOW);
#開始傳送
mySerial.print("0123456789");
Serial.println("0123456789");
delay(1000);

#ID0=LOW, ID=LOW,表示要連接 Device # 1
digitalWrite(pinID0, LOW);
digitalWrite(pinID1, LOW);
#等待 3ms 秒
delay(3)
#確定連接
digitalWrite(pinLat, LOW);
#開始傳送
mySerial.print("abcdefghij");
Serial.println("abcdefghij ");
delay(1000);

}

```

#### **RX 接收端程式範例：**

```

#include <SoftwareSerial.h>
SoftwareSerial mySerial(2, 3); // RX, TX

void setup() {
  Serial.begin(9600);
  mySerial.begin(9600);
}

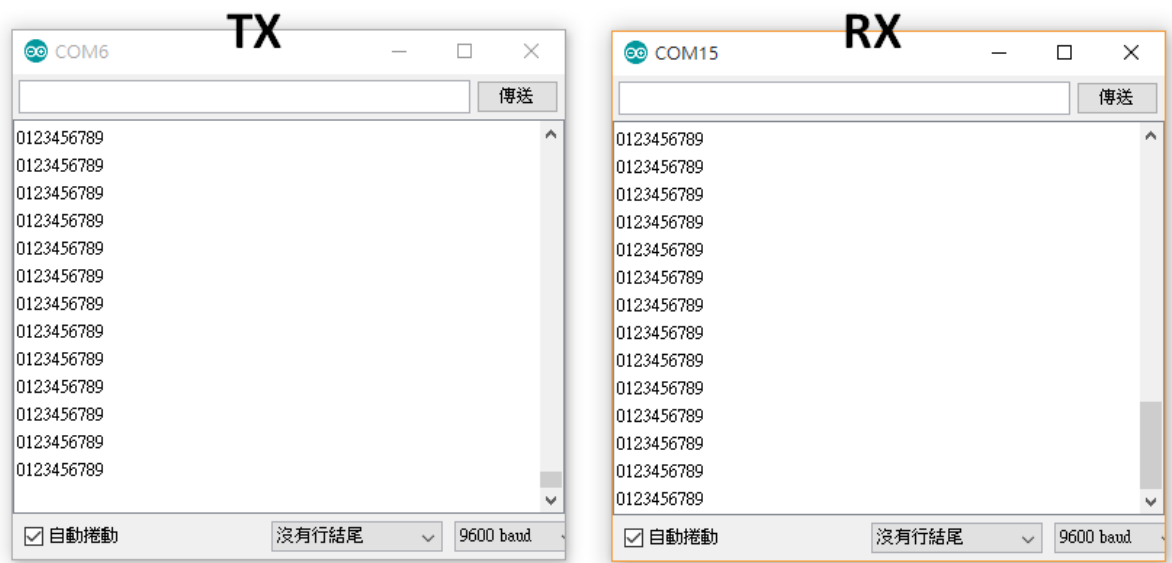
void loop() { // run over and over

  if (mySerial.available()) {
    Serial.println("");
    while (mySerial.available()) {
      Serial.print(char(mySerial.read()));
    }
  }

  delay(1000);
}

```

執行

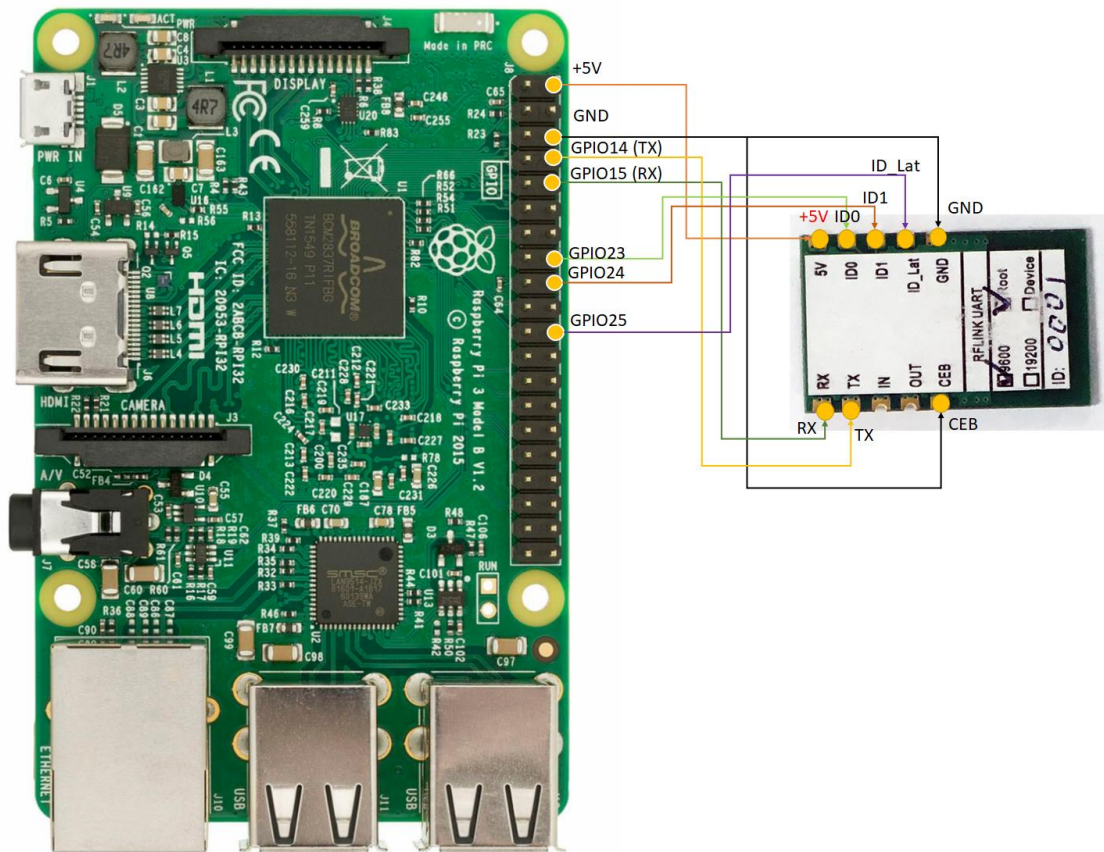


## 與 Raspberry Pi 樹莓派搭配使用

在樹莓派上使用本模組也是相當容易的！將本 RFLINK-UART 模組各腳位如同上述 Arduino 的範例，接到樹莓派各相對應的腳位即可，您就可以直接對 RX/TX 腳位進行讀取寫入並指定要連接的 device，使用方式就如同傳統的 UART。

下圖為 Root 端樹莓派與 RFLINK-UART 模組的接法，Device 端的接法基本上也相同，但 ID\_Lat pin 腳則不需要接，ID0 及 ID1 則視需求設定為不同的 ID 編號。





程式範例：

傳送端反覆的傳送到資訊到 device #3 及 device #1

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import time
import serial
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)

GPIO.setup(23, GPIO.OUT)
GPIO.setup(24, GPIO.OUT)
GPIO.setup(25, GPIO.OUT)

ser = serial.Serial(
    port='/dev/ttyS0',
    baudrate = 9600,
    parity=serial.PARITY_NONE,
    stopbits=serial.STOPBITS_ONE,
```

```

        bytesize=serial.EIGHTBITS,
        timeout=1
    )
    counter=0

    while 1:
        #指定要連到 device #3
        GPIO.output(23, GPIO.LOW)
        GPIO.output(24, GPIO.HIGH)
        #等待 3ms
        time.sleep(0.03)
        #確定切換到 device #3
        GPIO.output(25, GPIO.LOW)

        #讀取 UART 資料
        x=ser.readline()
        print x
        #寫入資料到 UART ， device #3 將收到此訊息
        ser.write('Write counter: %d \n'%(counter))

        #指定要連到 device #1
        GPIO.output(23, GPIO.HIGH)
        GPIO.output(24, GPIO.HIGH)
        #等待 3ms
        time.sleep(0.03)
        #確定切換到 device #1
        GPIO.output(25, GPIO.LOW)

        #讀取 UART 資料
        x=ser.readline()
        print x
        #寫入資料到 UART ， device #1 將收到此訊息
        ser.write('Write counter: %d \n'%(counter))

        time.sleep(1)

```

接收端：此範例為單純的接收

```

#!/usr/bin/env python
import time
import serial

ser = serial.Serial(
    port='/dev/ttyS0',
    baudrate = 9600,

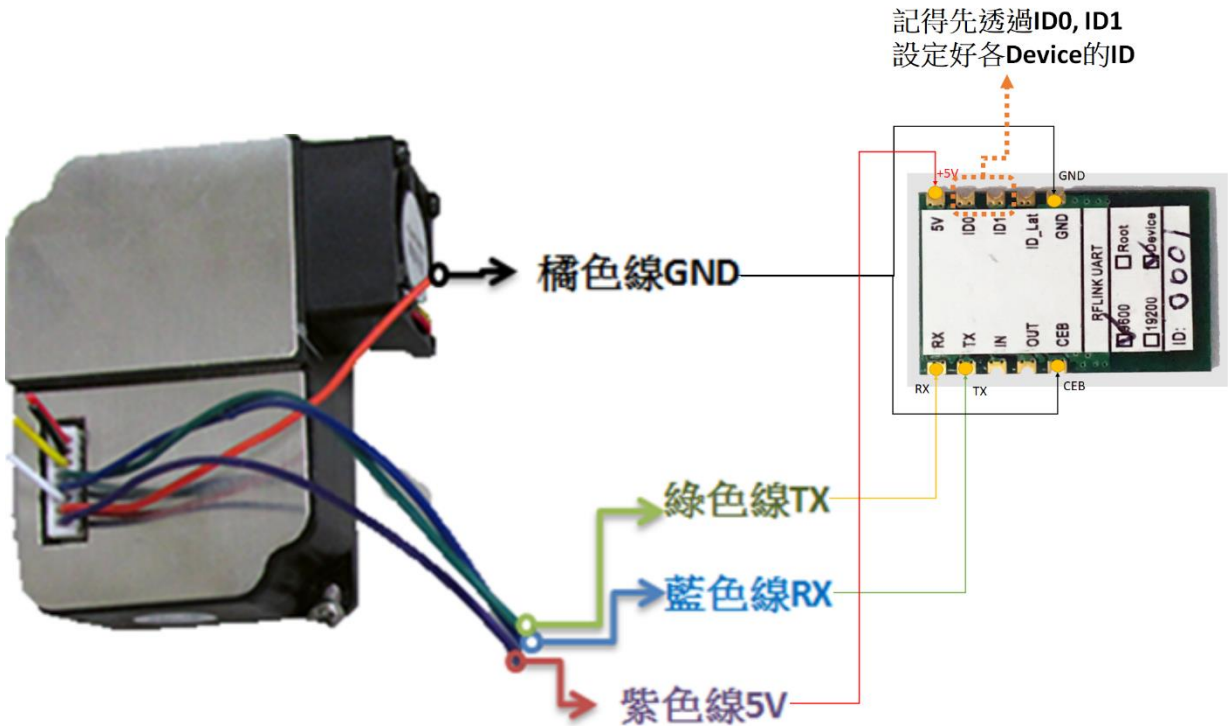
```

```
parity=serial.PARITY_NONE,
stopbits=serial.STOPBITS_ONE,
bytesize=serial.EIGHTBITS,
timeout=1
)
counter=0

while 1:
    #讀取 UART 資料
    x=ser.readline()
    print x
    #寫入資料到 UART
    ser.write('Write counter: %d \n'%(counter))
```

### 直接對接感測器

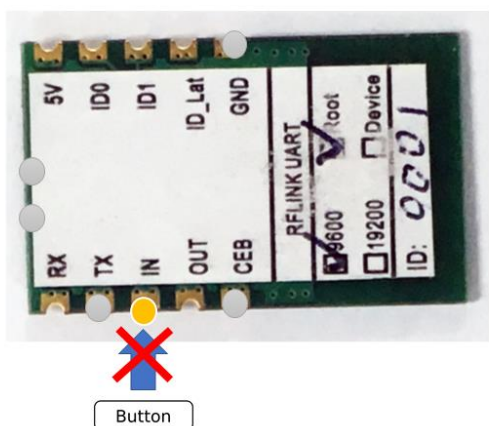
如果您的感測器支援 UART 介面而且 Baud rate 支援 9,600 或 19,200，那麼您可以直接將它接到 RFLINK-UART 模組的 device 端，就能迅速無痛升級成無線功能的感測器了。下方以 G3 PM2.5 感測器為例，參考如下接法，



接著再請您準備一張開發板（Arduino 或樹莓派皆可）接上 RFLINK-UART 模組的 Root 端，就可以像一般的 UART 方式來讀取到 G3 所傳來的 PM2.5 數據，恭喜你，該 G3 已升級為具有無線傳輸功能的 PM2.5 感測模組。

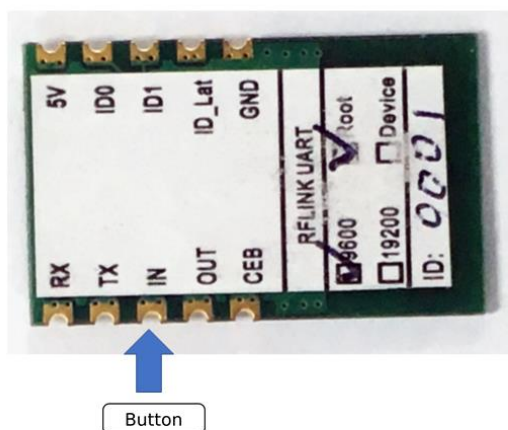
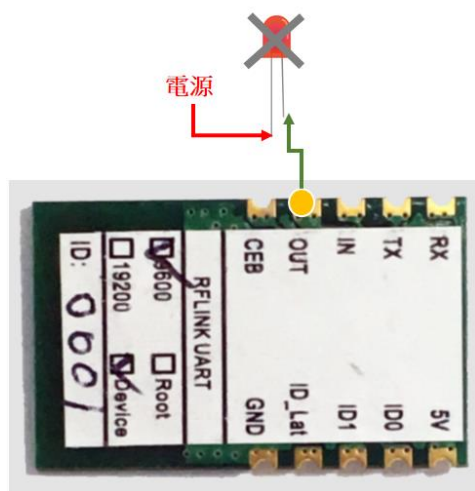
## 使用 IO Ports

RFLINK-UART 模組提供一組 IO port，它讓您可以透過無線來傳送 on/off 指令，且這組 IO Port 並不受限於模組的傳送或接收端，兩端皆可相互控制。您只要在任何一端的 IN port 改變電壓，就會同步改變另一端 Out port 的輸出電壓。請參考如下的使用範例，說明如何使用 IO Port 來遙控開關 led 燈泡。



當一端(Root/Device)的IN腳位未接上低電位 GND時，則為高電壓HIGH。

則另一端(Root/Device)對應的OUT亦為高電壓HIGH狀態 → LED 熄滅



當一端(Root/Device)的IN腳位接上低電位Ground時

則另一端(Root/Device)對應的OUT亦為GND接地狀態 → LED 點亮

