BUN LAB
Broadband Ubiquitous Networking Lab

通訊網路實驗

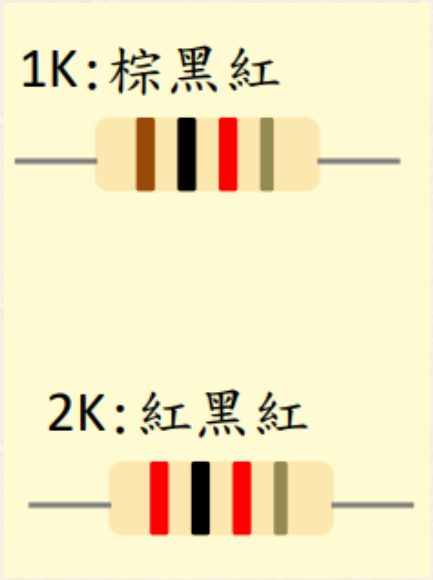
IoT應用

Raspberry PI

Dept. of Electrical and Computer Engineering (ECE) **National**
Yang Ming Chiao Tung University

實驗器材

1.Raspberry Pi
2.Micro SD card
3.電源線
4.USB 轉 TTL 序列傳輸線
5.溫溼度 sensor(DHT11)
6.超音波 sensor(HR-SRO4)
7.小麵包板
8.杜邦線(公對母)*10
9.LED燈
10.電阻(1K*2,2K*1)



評分標準 & 結報注意事項

- 出席　30%
- Demo　30%
- 結報　40%
  - 實驗目的
  - 實驗過程 (Code + 說明)
  - 問題及解法
  - 心得

抄襲0分哦☹

PDF檔案,期限為1週 (下次上課前一天晚上12點)

學號_姓名_Labx.pdf

BUN LAB
Broadband Ubiquitous Networking Lab

課程大綱

- 1. 認識Raspberry Pi, Python複習
- 2. GPIO介紹與感測器應用
  - LED + 溫溼度感測器 + 超音波感測器

BUN LAB
Broadband Ubiquitous Networking Lab

Demo項目

- Q1：用LED產生SOS的摩斯密碼
- Q2：設計一個溫度警示燈
- Q3：設計一個距離警示燈
- Q4：結合溫度偵測和距離偵測

Introduction

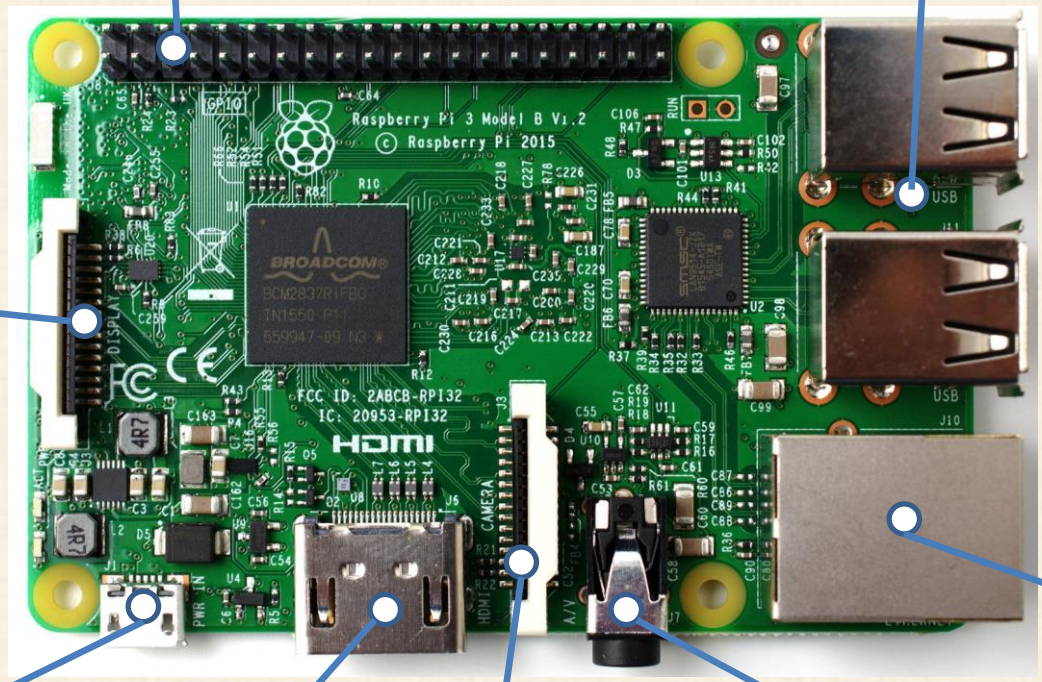
- 信用卡大小般的電腦

http://www.flickr.com/photos/fotero/7697063016/

Raspberry Pi 可以用在?

- **桌機 (Raspberry Pi OS)**
  - FTP, Web, NAS, AP, 自動化控制…等
- **多媒體影音作業系統**
  - OSMC (Open-Source Media Center)
  - OpenELEC (Open Embedded Linux Entertainment Center)
- **遊戲機 (RetroPie、PiPlay)**
- **網站滲透測試 (Kali Linux)**
- **Android系統 (RaspAnd)**
- **超級電腦**
  - 將一堆PI組裝在一起

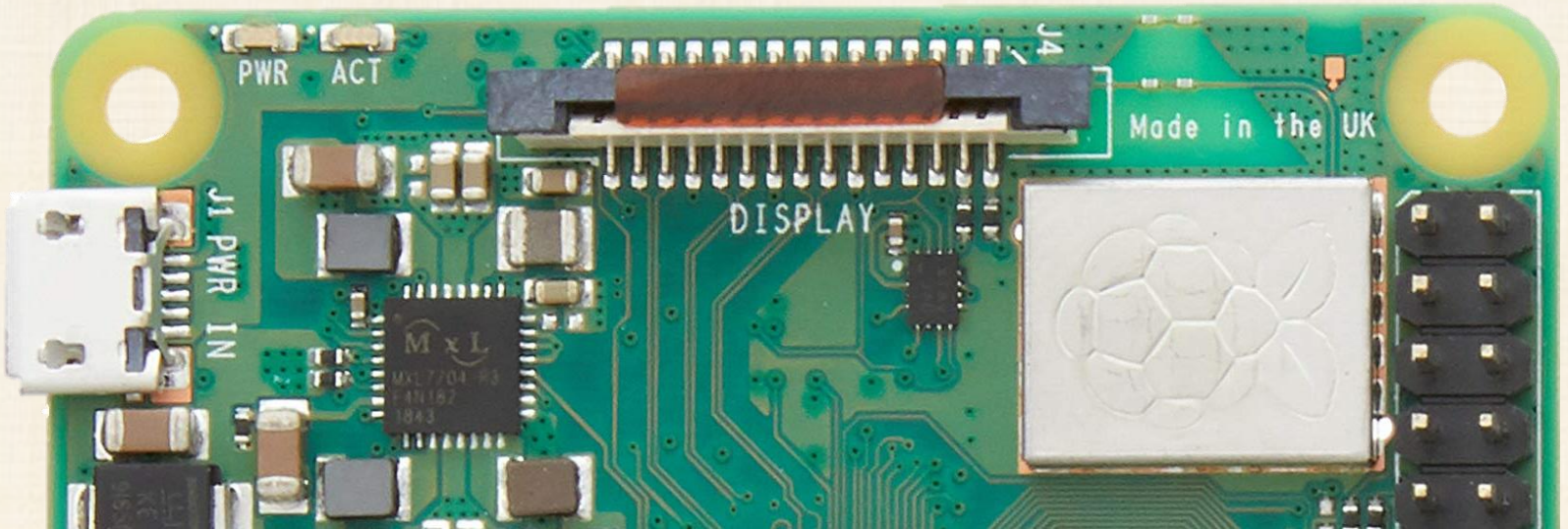
Raspberry PI 硬體週邊



BUN LAB
Broadband Ubiquitous Networking Lab

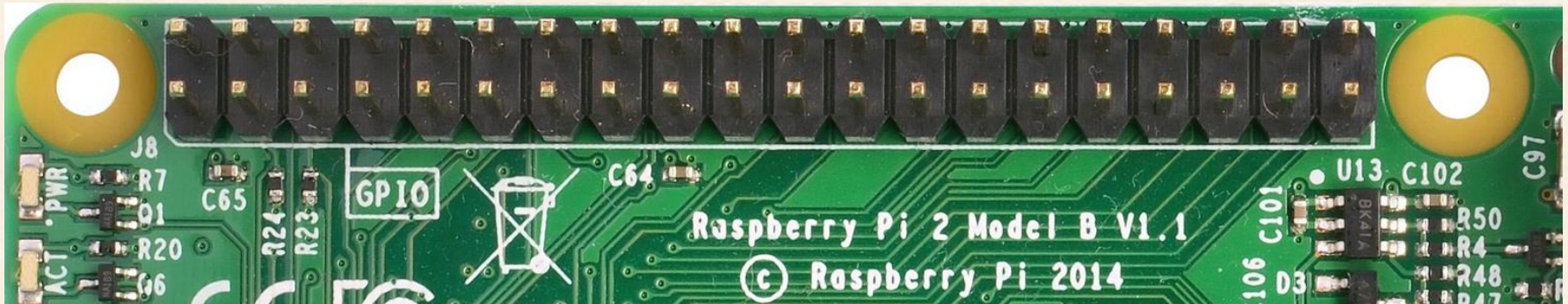
PI LED指示燈說明

LED	顏色	功能	代表
ACT	綠	Card status	閃爍：SD卡正在活動
PWR	紅	power	恆亮：有電源輸入



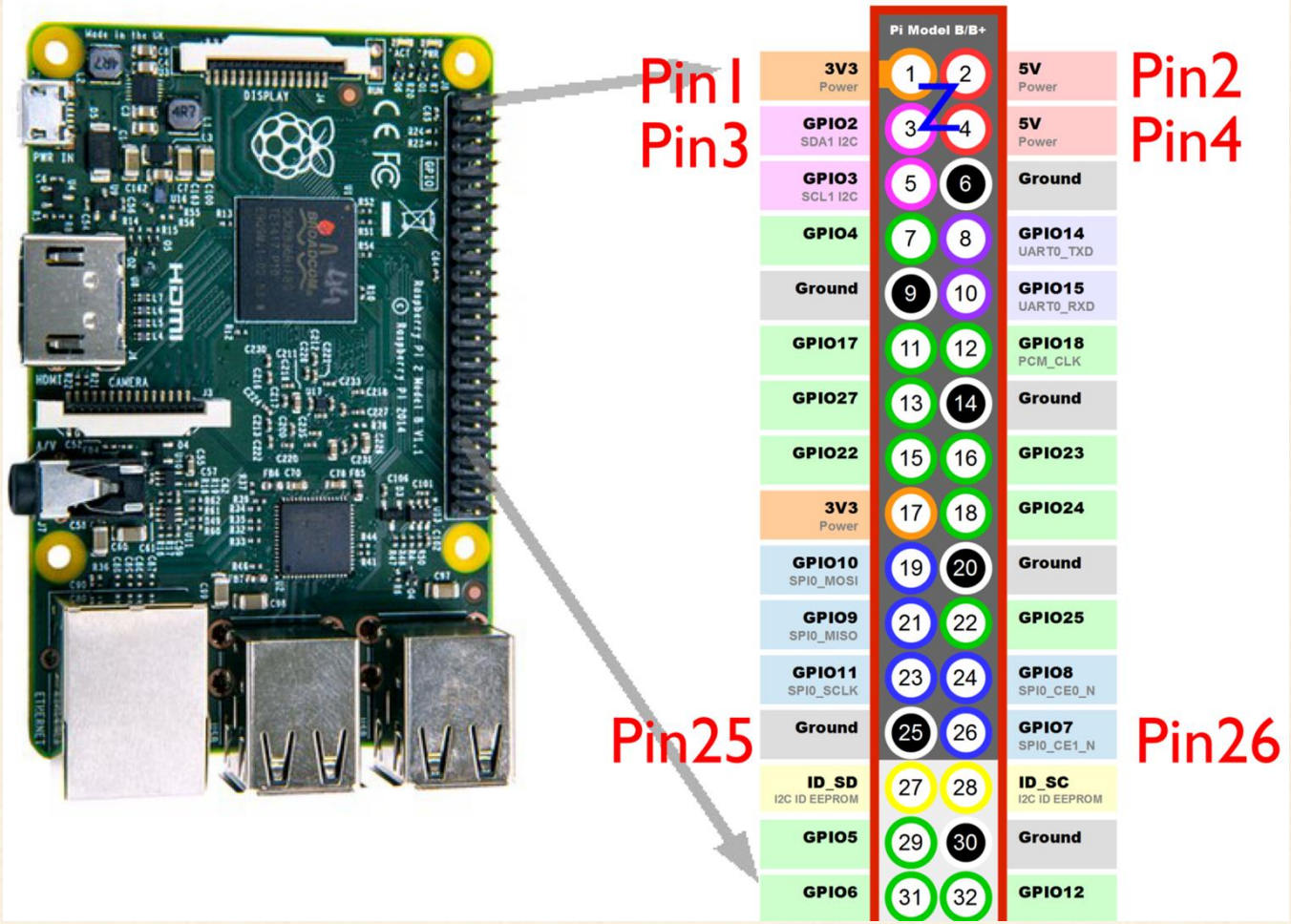
GPIO on Raspberry PI

- General-purpose input/output (GPIO)
  - 通用型之輸入輸出(General Purpose I/O)
  - PIN腳可設為輸入(GPI), 輸出(GPO), 輸入與輸出(GPIO)
    - 輸出：寫值到某根腳位
    - 輸入：從某根腳位讀值
  - 可產生用軟體控制的數位訊號



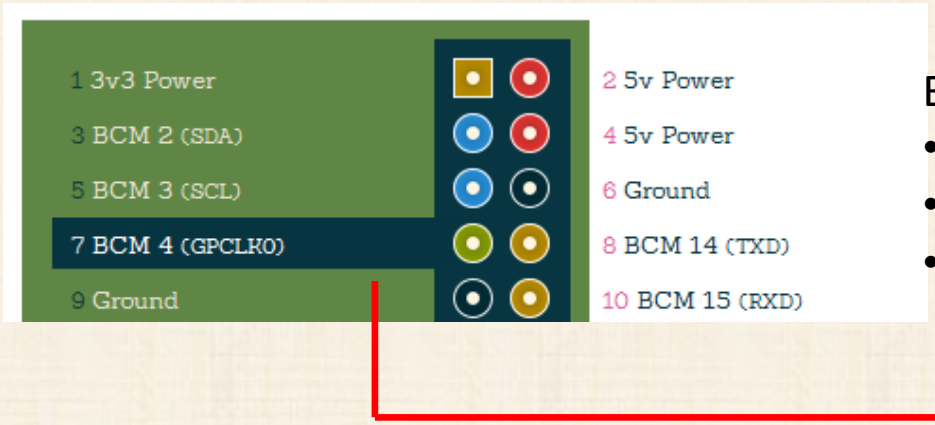
GPIO on Raspberry PI

- Z 字型的腳位編號



GPIO的腳位與應用

- 注意腳位的編號!!
  - Pin number != GPIO number
    - Physical numbering vs. GPIO numbering
  - Ex: Pin 7 = GPIO 4
  - Mode
    - GPIO.BOARD =>Pin number
    - GPIO.BCM =>GPIO number



BCM 4:
- Physical pin 7
- BCM pin 4
- Wiring Pi pin 7

Raspberry Pi B+ GPIO Header Details And Pinout - Raspberry Pi Spy (raspberrypi-spy.co.uk)

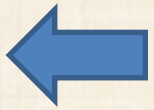
BUN LAB
Broadband Ubiquitous Networking Lab

GPIO的注意事項

- 在GPIO pin上不可以輸入超過3.3V
- GPIO PIN的電流上限是 3mA
- **不要拿金屬物體接觸GPIO PIN (會短路)**
  - 使用杜邦線, 針腳不要碰到板子
- 用GPIO PIN啟動PI時, 電壓不可以超過5V
  - 不太建議這樣用, 因為容易短路, 也不適合接太多周邊
- 供電腳位(3.3V)不可以超過50mA
- 供電腳位(5V)不可以超過250mA

控制 PI 的 GPIO

- 寫程式
  - C
  - C + wiringPi
  - C#
  - Ruby
  - Perl
  - **Python**
  - Scratch
  - Java Pi4J Library
  - Shell script



我們用這個

BUN LAB
Broadband Ubiquitous Networking Lab

Python 簡介

- 程式架構可大致分為這幾類
  1. 變數 , 物件 , 註解
  2. 模組
  3. 縮排
  4. 迴圈
  5. 條件判斷
  6. 函式

- PS. PI預設使用Python 2.7

(1). 變數，物件，註解

- 動態型別 (dynamic typing)

```
# 井號是註解

i = 3                       # 變數 i 指到數字物件 3
print i                     # 印出 i

i = [1, 2, 3, 4, 5]         # 變數 i 指到串列物件
print i[2]                  # 印出串列中第三個元素

i = “abcde”      # 變數 i 指到字串物件
print i[0]                  # 印出字串中第一個元素
```

(2). 模組

- 模組

```
# import MODULE
import RPi.GPIO

# import MODULE as ALIAS
import RPi.GPIO as GPIO
```

可以使用該模組的函式

```
GPIO.setmode(GPIO.BOARD)
GPIO.setup(12, GPIO.OUT)
```

(3). 縮排

- 用縮排取代大括號
- 程式碼的區塊是用縮排分隔
- 不使用 tab, 使用空白鍵
- 常見縮排為 4 個空白鍵

```
for i in xrange(start, stop[, step]) :
    process
```

前面有 4 個空白鍵

(4). 迴圈

- 自動迭代 (iterator); for loop

```
for i in xrange(start, stop[, step]) :
    process


for i in xrange(0, 11, 5) :
    process
```

前面有 4 個空白鍵

https://docs.python.org/2/tutorial/controlflow.html

(5). 條件判斷

- If statement

```
if condition_1:
    process_1
elif condition_2:
    process_2
else :
    process_3
process_4
```

```
if value > 20:
    print("lalala")
else:
    print(“XD")
```

https://docs.python.org/2/tutorial/controlflow.html

(6). 函式

- 自訂函式 (User-Defined Functions)

```
def function_name():
    process
def function_name(param_name):
    process
def function_name(param_name = 3):
    process
```

```
def f(x):
    return x**2 + 1
```

f(4) 可以得到 17

http://www.pitt.edu/~naraehan/python2/user_defined_functions.html

Python 小提示

- 如果有兩個Statement 要寫在同一行使用(;)分開

```
>>> print 'hello';print 'runoob';
hello
runoob
```

- 一個Statement 要分多行顯示則使用(\)

```
total = item_one + \
        item_two + \
        item_three
```

- Python的Statement自動默認換行，不換行需加(,)

```
# 不换行输出
print x,
print y,

# 不换行输出
print x,y
```

http://www.runoob.com/python/python-basic-syntax.html

Python 小提示

- Python保留字元，不可拿來當變數名、常數...

and	exec	not
assert	finally	or
break	for	pass
class	from	print
continue	global	raise
def	if	return
del	import	try
elif	in	while
else	is	with
except	lambda	yield

http://www.runoob.com/python/python-basic-syntax.html

BUN LAB
Broadband Ubiquitous Networking Lab

Linux常見指令

- $ ls
  - 查看當下目錄中的檔案
- $ mkdir 資料夾名稱
  - 新增資料夾
- $ cd 資料夾名稱(ex: cd Desktop)
  - 移動到Desktop中
- $ cd ..
  - 上一頁(資料夾)
- $ cd (空格)
  - 回主畫面 Home page (不是桌面)
- $ nano 檔案名稱 (ex: nano led.py)
  - 編輯led.py (如果沒有led.py，則會新建一個led.py)

文字編輯器-Nano

- Nano為Linux的文字編輯套件
  1. EX：$ nano 文件檔名.py
  2. 使用方式跟記事本一樣
  3. Ctrl + X ：退出(接著會問你是否要存檔)
  4. Ctrl + O：另存新檔

Ctrl+6	開始/結束選取
Alt+6	複製整列或選取範圍
Ctrl+K	剪下整列或選取範圍
Ctrl+U	貼上剪貼簿內容
Alt+Del	刪除整行
Alt+W	尋找
Alt+R	尋找並置換
Ctrl+S	存檔

執行Python

- 執行python程式：
  - `$ python filename.py`
  - 終止執行：Ctrl + C
  - 若牽涉到系統權限, 需要sudo開頭執行程式
    - Ex: `$ sudo python netexp.py`

開始實驗

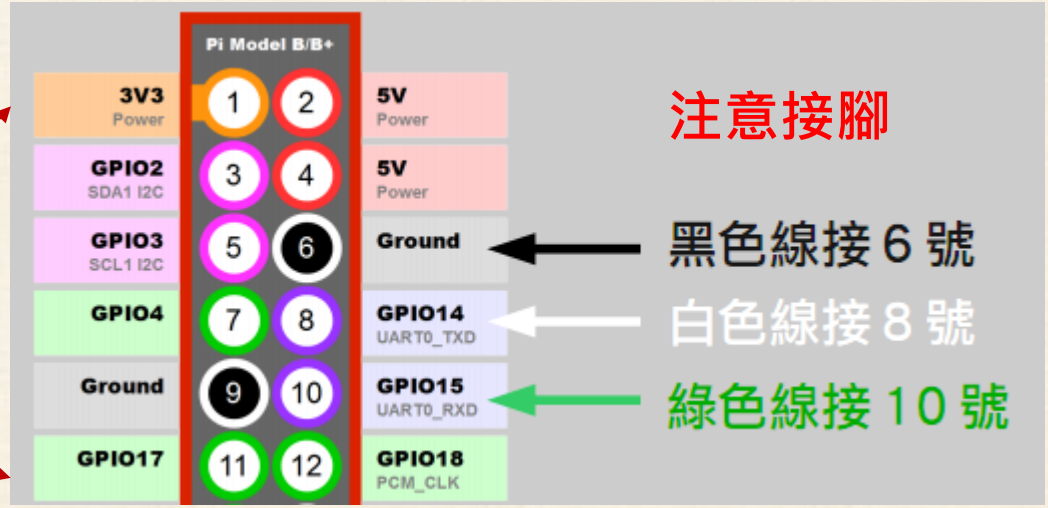
BUN LAB
Broadband Ubiquitous Networking Lab

注意事項

- 上電時勿以金屬碰觸任何腳位
- 務必先切斷電源
  - 插拔SD卡　　如果熱插拔SD卡，需要重灌系統
- 切斷電源前務必關機
  - `$ sudo shutdown -h now`
  - 等到**ACT**(綠)**不再閃爍**，只剩PWR(紅)長亮時，即可斷電

TTL序列連接

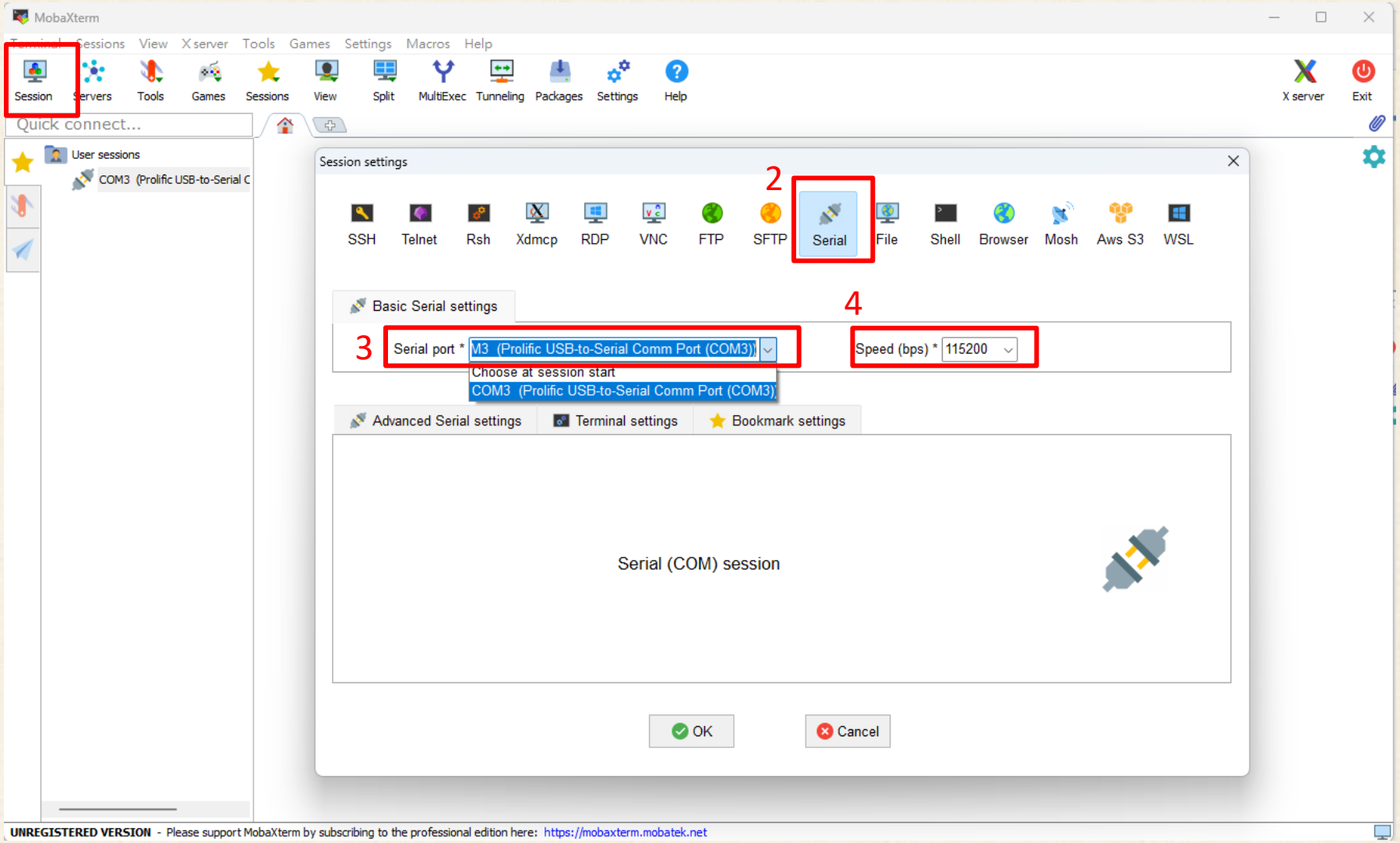
- 透過 USB 轉 TTL 序列傳輸線，就可以在不需要外接螢幕和鍵盤滑鼠的情況下登入 Raspberry Pi。
  1.確認SD卡已插到Raspberry PI
  2.將USB 轉 TTL 序列傳輸線，按下圖接上Rasp Pi
  3.接上電源開機
- **預設登入帳密**
  - ID: pi
  - PW: raspberry

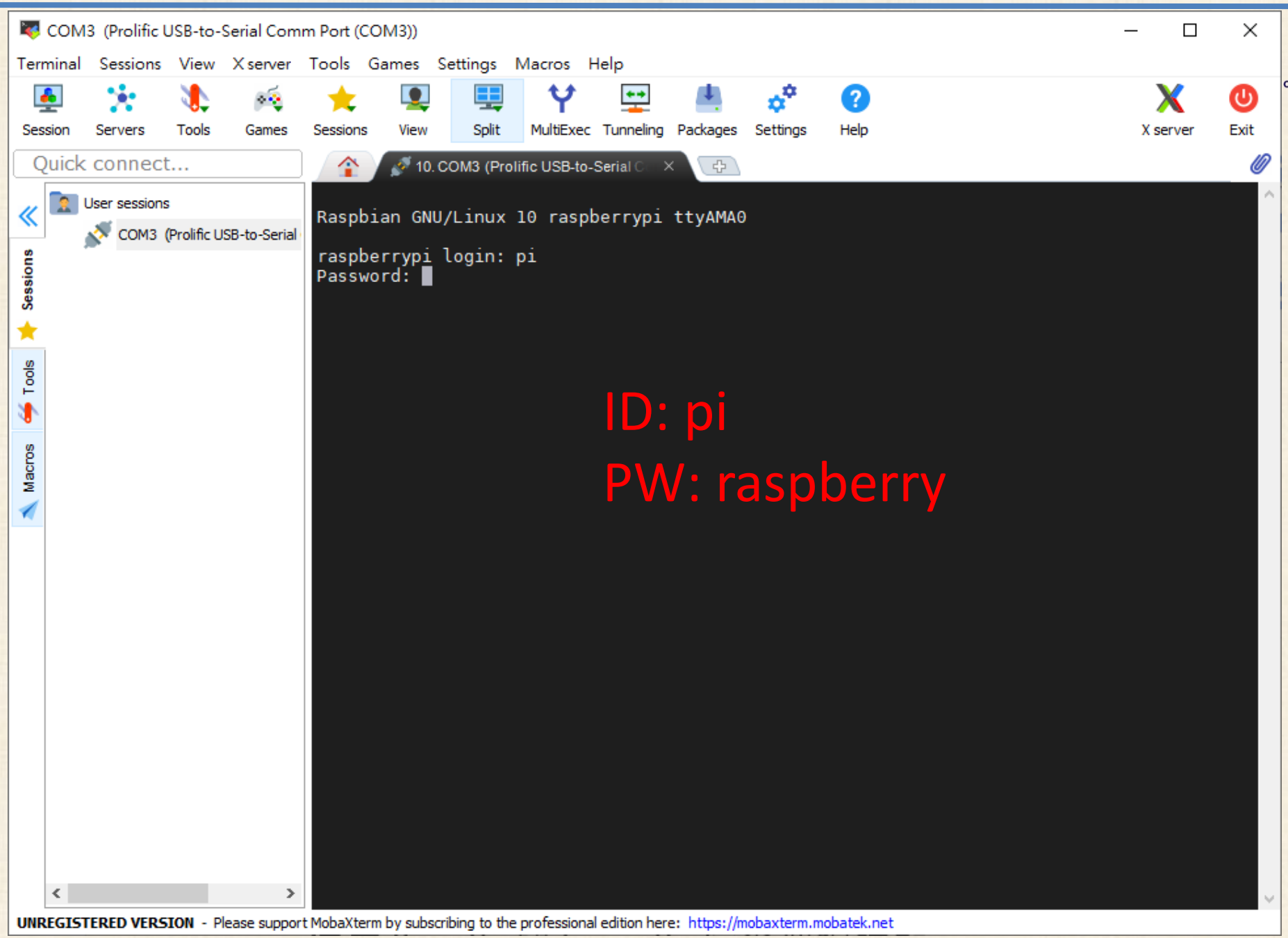


https://piepie.com.tw/1999/connect-to-raspberry-pi-via-serial

BUN LAB
Broadband Ubiquitous Networking Lab

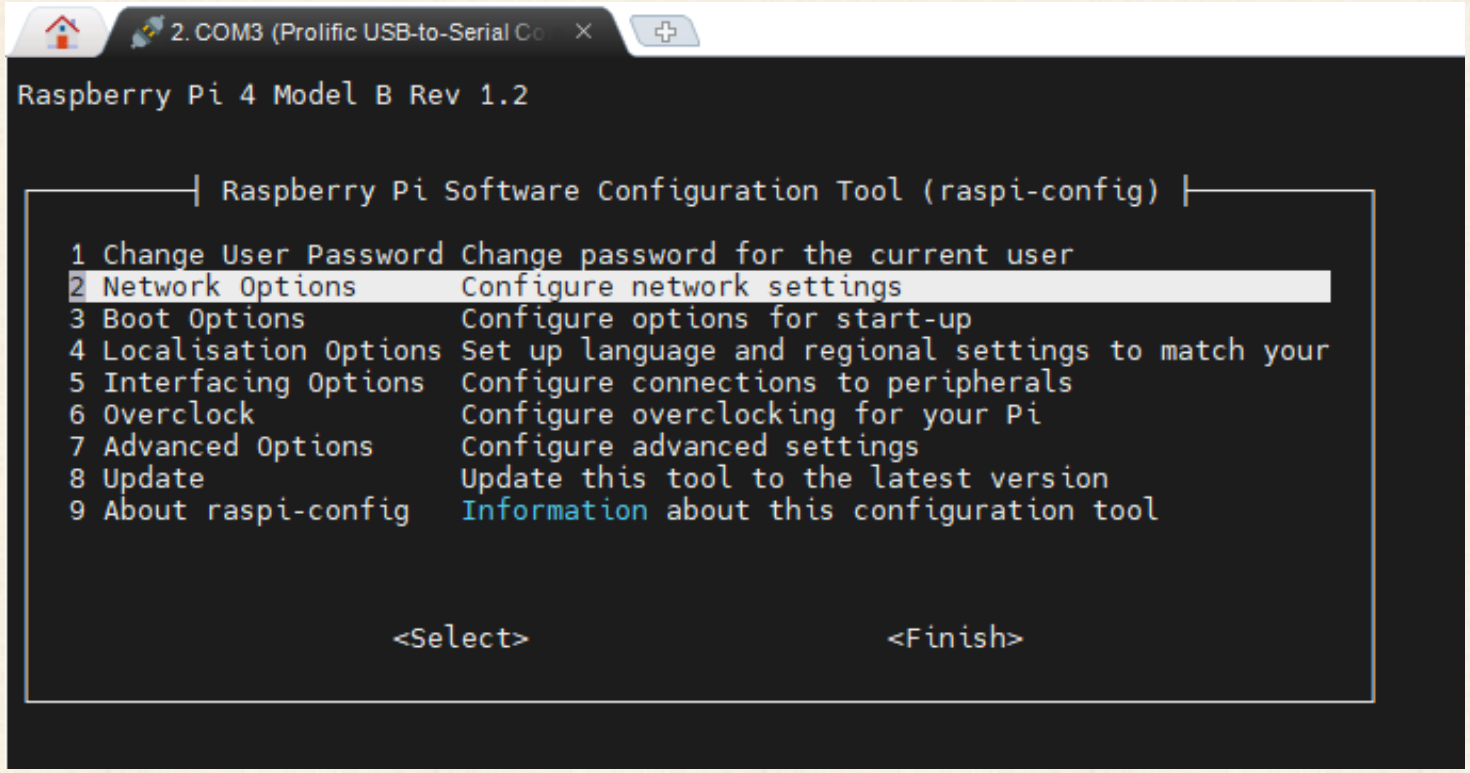
MobaXterm畫面




COM3 (Prolific USB-to-Serial Comm Port (COM3))
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help X server Exit
Quick connect...
User sessions
COM3 (Prolific USB-to-Serial
Sessions
Tools
Macros
10. COM3 (Prolific USB-to-Serial C
Raspbian GNU/Linux 10 raspberrypi ttyAMA0
raspberrypi login: pi
Password:
ID: pi
PW: raspberry
UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

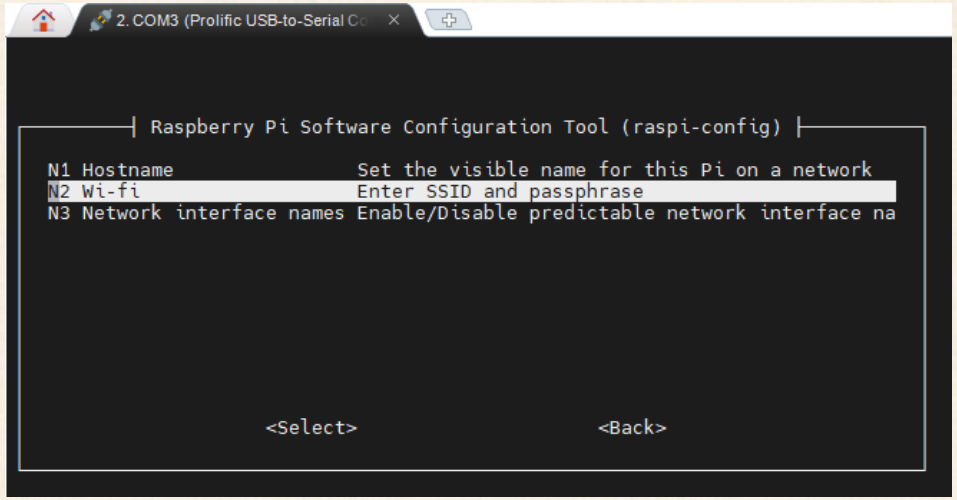
連接Wi-fi(1)

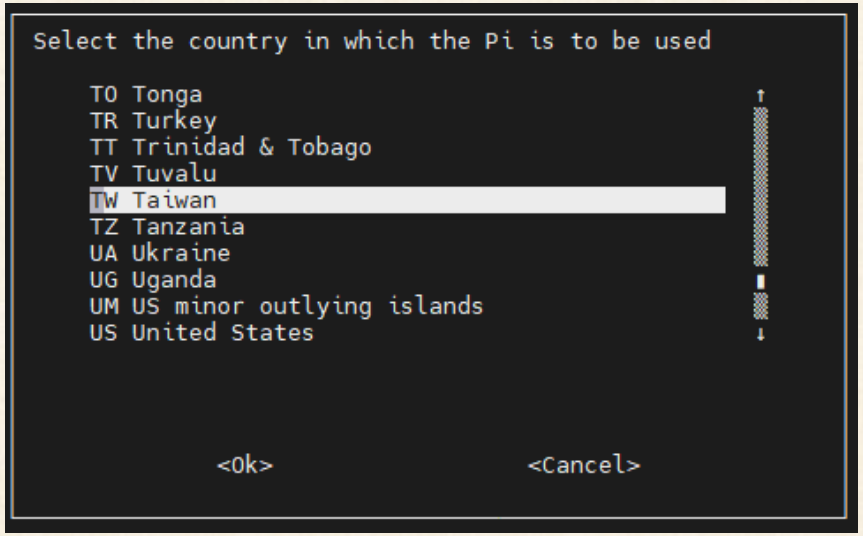
- $ sudo raspi-config
- 選擇Network Options



連接Wi-fi(2)

- 選擇Wi-fi





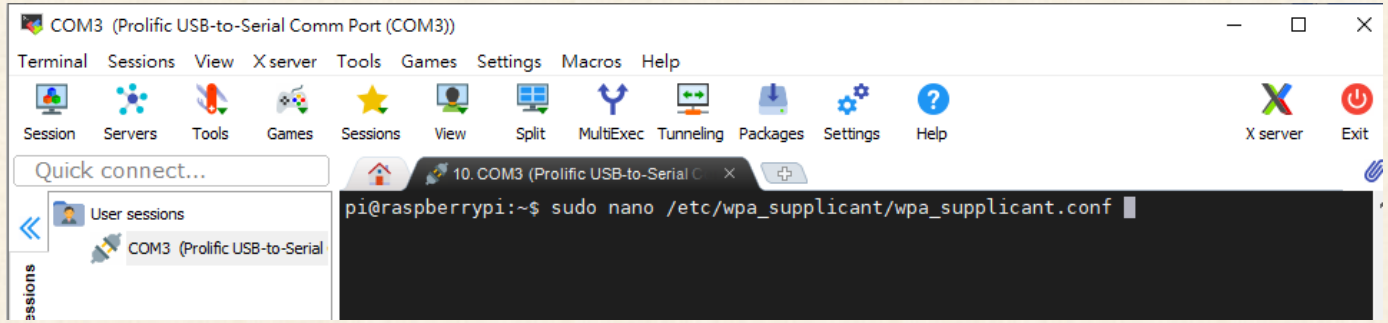
- 選擇國家：TW
- 輸入Wifi名稱(SSID)：Bun
- 輸入Wifi密碼：12345678
- 完成後，方向鍵→，<Finish>

連接Wi-fi(備案)

1.輸入

```
$ sudo nano /etc/wpa_supplicant/wpa_supplicant.conf
```

(編輯設定檔)



2. 填寫SSID與密碼

(最下面新增空白列輸入)　　**自行輸入, 複製會有誤**

```
country=TW
network={
        ssid="Bun"
        psk="12345678"
}
```

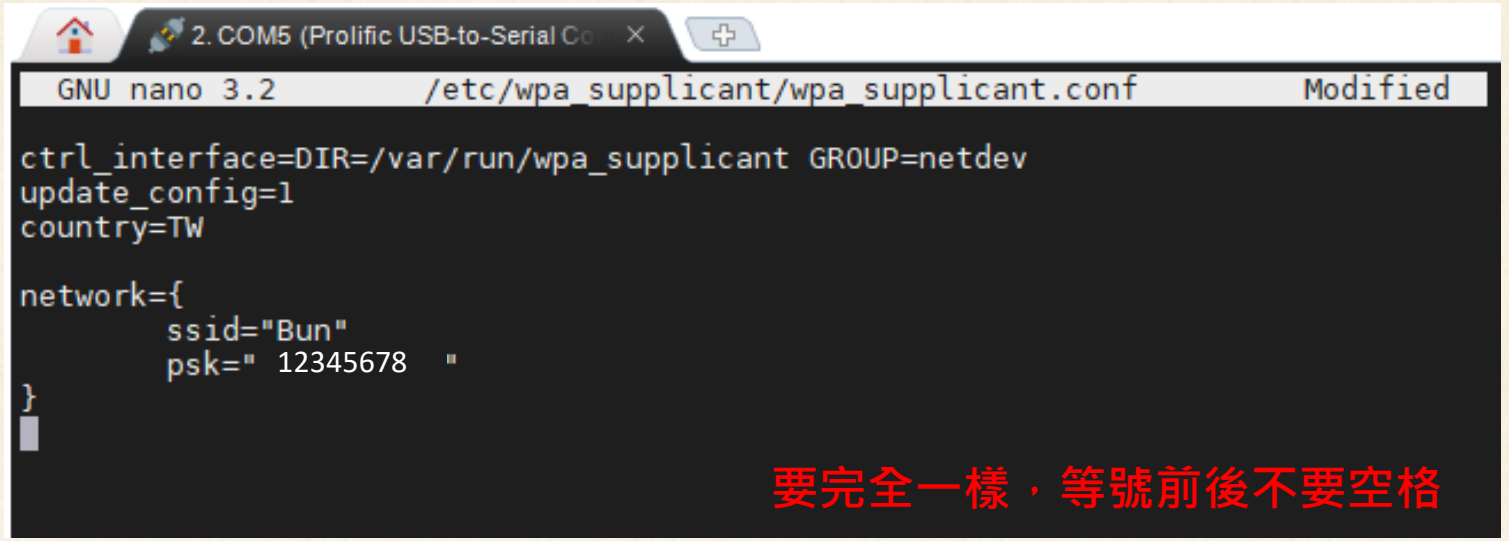
連接Wi-fi(備案)

2.文字編輯器 nano

- 編輯結束按 **ctrl + x** 離開
  - 若有變動, 會問你是否存檔, 輸入 **Y** 即可

3.重開樹梅派

- 輸入 $ sudo reboot



BUN LAB
Broadband Ubiquitous Networking Lab

確認連線狀態

輸入以下指令檢查連線狀態

- $ ifconfig：網路介面配置資訊
- $ iwconfig：針對無線網路配置資訊

```
pi@raspberrypi:~$ ifconfig
eth0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether b8:27:eb:d6:af:d1  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.50.130  netmask 255.255.255.0  broadcast 192.168.50.255
        inet6 fe80::200e:c37e:c9e9:77d5  prefixlen 64  scopeid 0x20<link>
        ether b8:27:eb:83:fa:84  txqueuelen 1000  (Ethernet)
        RX packets 325  bytes 50925 (49.7 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 122  bytes 16959 (16.5 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

```
pi@raspberrypi:~ $ iwconfig
eth0      no wireless extensions.

lo        no wireless extensions.

wlan0     IEEE 802.11  ESSID:"Bun"
          Mode:Managed  Frequency:2.462 GHz  Access Point: 18:D6:C7:FC:84:78
          Bit Rate=57.7 Mb/s   Tx-Power=31 dBm
          Retry short limit:7   RTS thr:off   Fragment thr:off
          Power Management:on
          Link Quality=63/70  Signal level=-47 dBm
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:1  Invalid misc:0   Missed beacon:0
```

0. 開始寫程式前

- 先安裝程式&相關函式庫
  - `$ sudo apt update`
  - `$ sudo apt-get update`
  - `$ sudo apt-get install -y python-dev python-pip python-rpi.gpio`

1. -dev (develop) 開發包
2. pip是一個以Python寫成的軟體包管理系統
3. rpi.gpio 為樹梅派的GPIO函式庫

1. 控制LED燈

- Python code (LED_blink.py)

```
1  #!/usr/bin/env python
2  import RPi.GPIO as GPIO
3  import time
4
5  GPIO.setmode(GPIO.BOARD)
6  LED_PIN = 12
7  GPIO.setup(LED_PIN, GPIO.OUT)
8
9  while True:
10     print("LED is on")
11     GPIO.output(LED_PIN, GPIO.HIGH)
12     time.sleep(1)
13     print("LED is off")
14     GPIO.output(LED_PIN, GPIO.LOW)
15     time.sleep(1)
16
17 GPIO.cleanup()
```

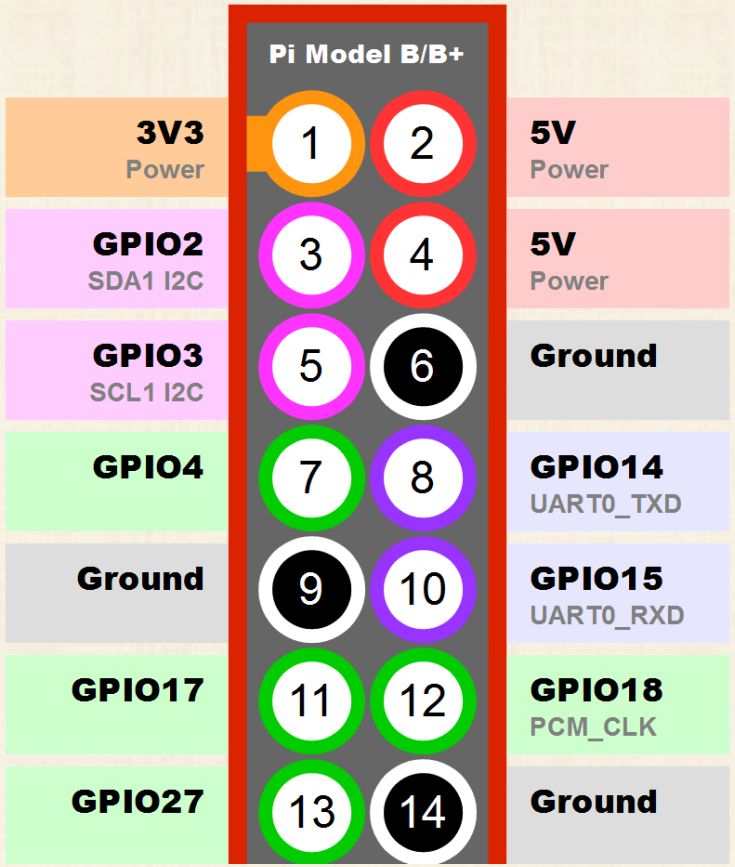
載入函式庫

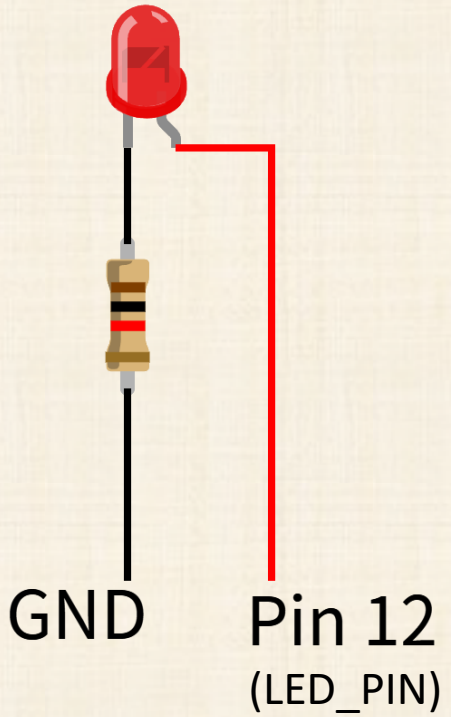
GPIO.BOARD:
按照腳位的順序編號(Pin Num)
使用 pin 12的接腳

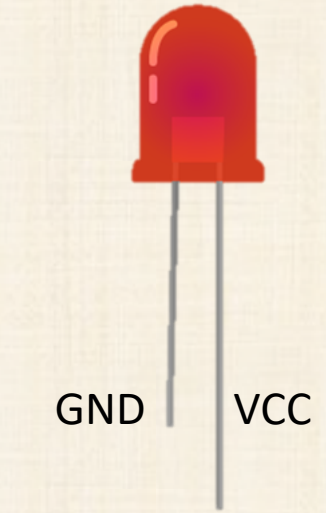
判斷、控制
縮排要統一

1. 控制LED燈

- 電路圖


Pi Model B/B+
3V3
Power
1
2
5V
Power
GPIO2
SDA1 I2C
3
4
5V
Power
GPIO3
SCL1 I2C
5
6
Ground
GPIO4
7
8
GPIO14
UART0_TXD
Ground
9
10
GPIO15
UART0_RXD
GPIO17
11
12
GPIO18
PCM_CLK
GPIO27
13
14
Ground


GND
Pin 12
(LED_PIN)


GND
VCC

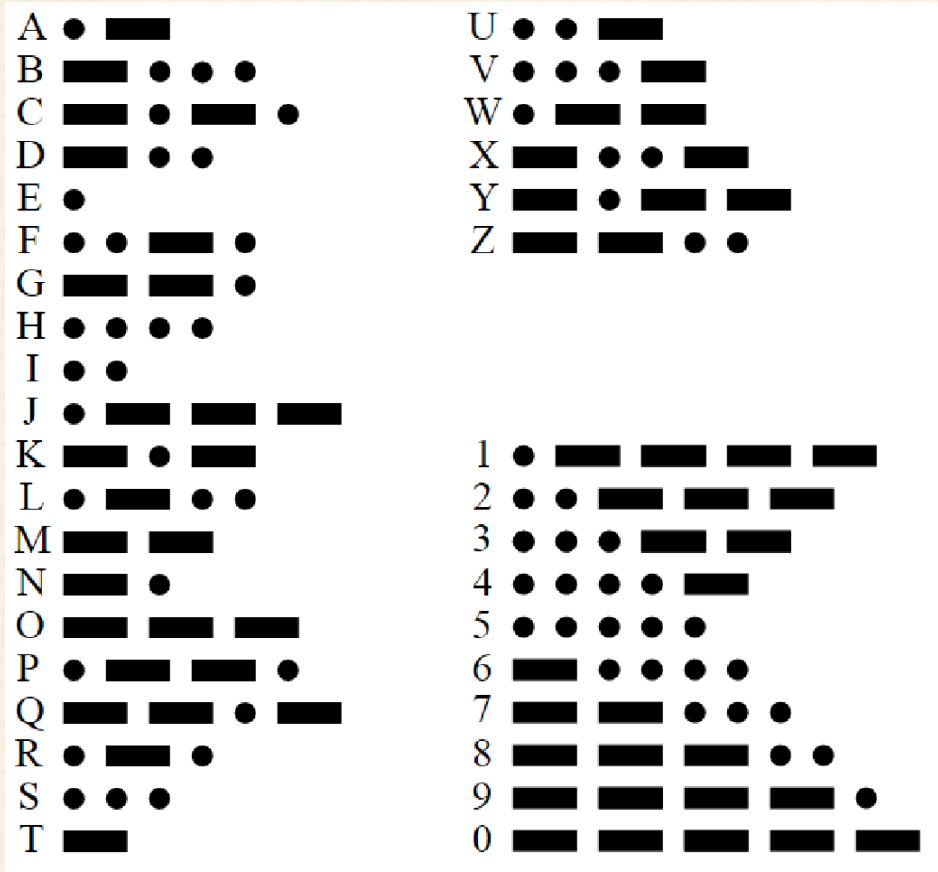

1K:棕黑紅

Q1

▢ 用LED產生SOS的摩斯密碼(循環顯示)

1. The length of a dot is one unit.
2. A dash is three units.
3. The space between parts of the same letter is one unit.
4. The space between letters is three units.
5. The space between words is seven units.

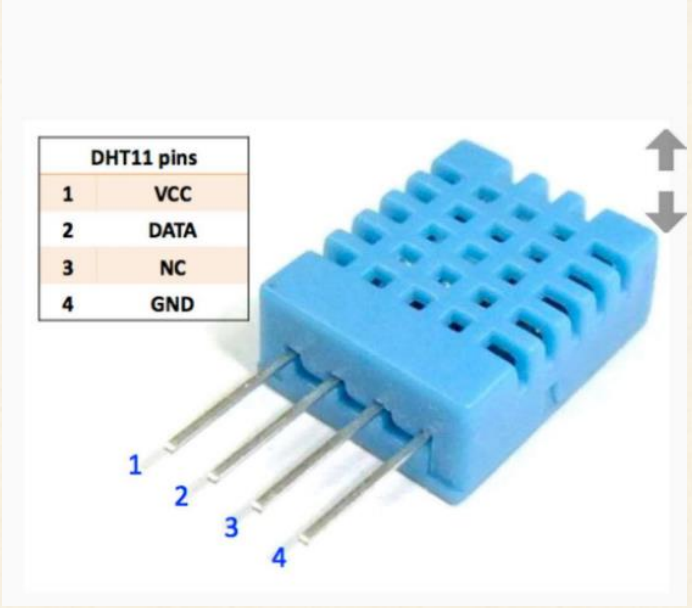
PS. 設定 one unit = 0.1 sec

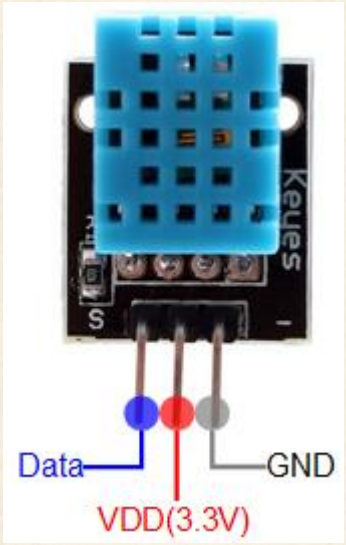


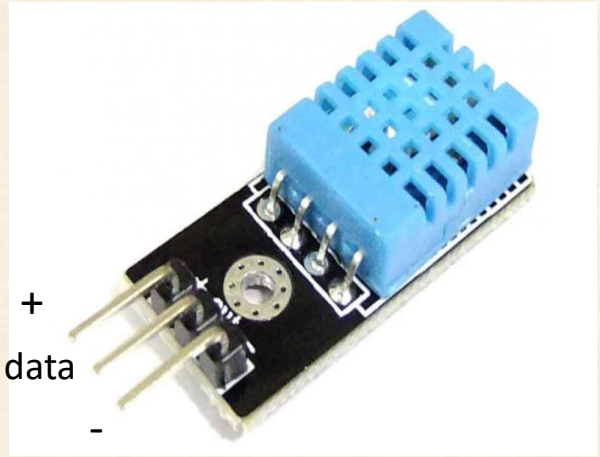
https://en.wikipedia.org/wiki/Morse_code

2. 溫濕度sensor

- DHT11
  - 溫濕度感應器
  - 溫度: 0~50℃, 誤差±2 ℃
  - 濕度: 20~90%RH, 誤差±5%
  - 使用三個腳位: Data , VCC , GND (out, + , -)
  - 供電電壓：3.3～5.5V







注意針腳順序不一樣!!插錯會燒掉

2. 溫濕度sensor

- Adafruit的模組與範例程式

  Source code: https://github.com/adafruit/Adafruit_Python_DHT

1. 安裝編譯程式

```
$ sudo apt-get update
$ sudo apt-get install git-core build-essential python-dev
```

2. 下載source code

```
$ git clone https://github.com/adafruit/Adafruit_Python_DHT.git
```

3. 安裝Adafruit_Python_DHT程式

```
$ cd Adafruit_Python_DHT   (移動至路徑)
$ sudo python setup.py install
```

2. 擷取溫濕度資訊

- 執行範例程式(後面要接2個參數)
  - `$ cd examples`
  - `$ sudo ./AdafruitDHT.py 11 4`
    - 使用 DHT 11 感應器 (另有DHT 22, DHT 2302)
    - Data腳位連接 GPIO 4 (也就是Pin 7)
    - 輸入指令，得到資訊: Temp=26.0*C Humidity=37.0%

```
pi@raspberrypi ~ $ cd Adafruit_Python_DHT/examples/
pi@raspberrypi ~/Adafruit_Python_DHT/examples $ sudo ./AdafruitDHT.py 11 4
Temp=26.0*  Humidity=37.0%
```

- 調整 溫度 & 濕度 的顯示方式
  - 觀察AdafruitDHT.py，可改為自己想要的格式

Code第一行是環境變數, 需保留:
#!/usr/bin/python

2. 擷取溫濕度資訊

- AdafruitDHT.py:

執行方式 $ sudo ./AdafruitDHT.py 11 4

```
#!/usr/bin/python
import sys
import Adafruit_DHT

sensor_args = { '11': Adafruit_DHT.DHT11,
                '22': Adafruit_DHT.DHT22,
                '2302': Adafruit_DHT.AM2302 }
if len(sys.argv) == 3 and sys.argv[1] in sensor_args:
    sensor = sensor_args[sys.argv[1]]
    pin = sys.argv[2]
else:
    print('usage: sudo ./Adafruit_DHT.py [11|22|2302] GPIOpin#')
    print('example: sudo ./Adafruit_DHT.py 2302 4 - Read from an AM2302 connected to GPIO #4')
    sys.exit(1)

humidity, temperature = Adafruit_DHT.read_retry(sensor, pin)

if humidity is not None and temperature is not None:
    print('Temp={0:0.1f}*  Humidity={1:0.1f}%'.format(temperature, humidity))
else:
    print('Failed to get reading. Try again!')
    sys.exit(1)
```

環境變數，一定要

載入函式庫

定義.py後的參數

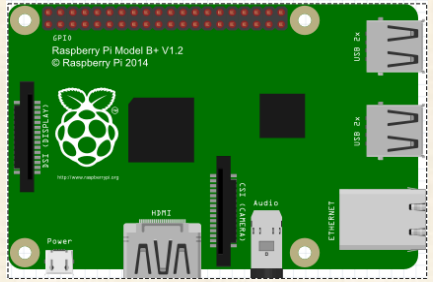
導入.py後的2個輸入參數
(Adafruit_DHT/common.py)

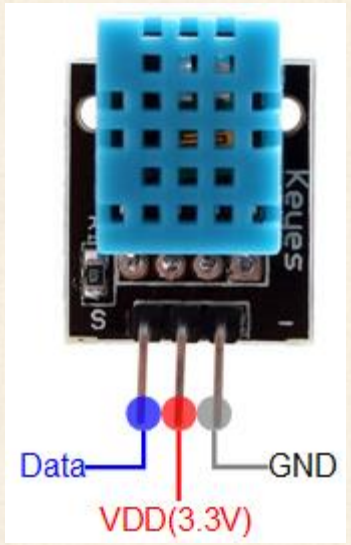
回傳數值

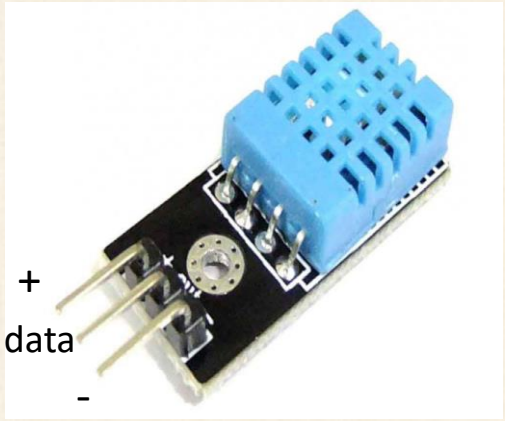
Q2

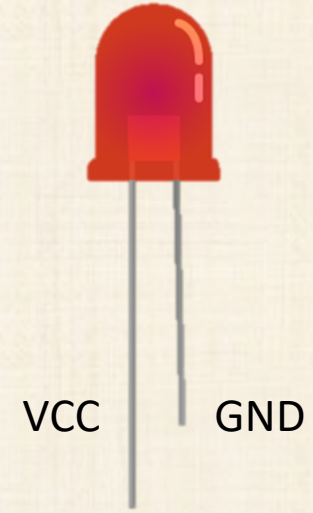
- 設計一個溫度警示燈
  - 輸入門檻值(ex: 27度)
  - 每秒輸出當前溫度及濕度
  - 當溫度大於門檻值時，開啟LED燈號

```
門檻值 = input()
while(Ture):
    讀取及顯示現在溫溼度
    if 現在溫溼度 > 門檻值:
        LED燈亮
    else:
        LED燈暗
    等1秒
```



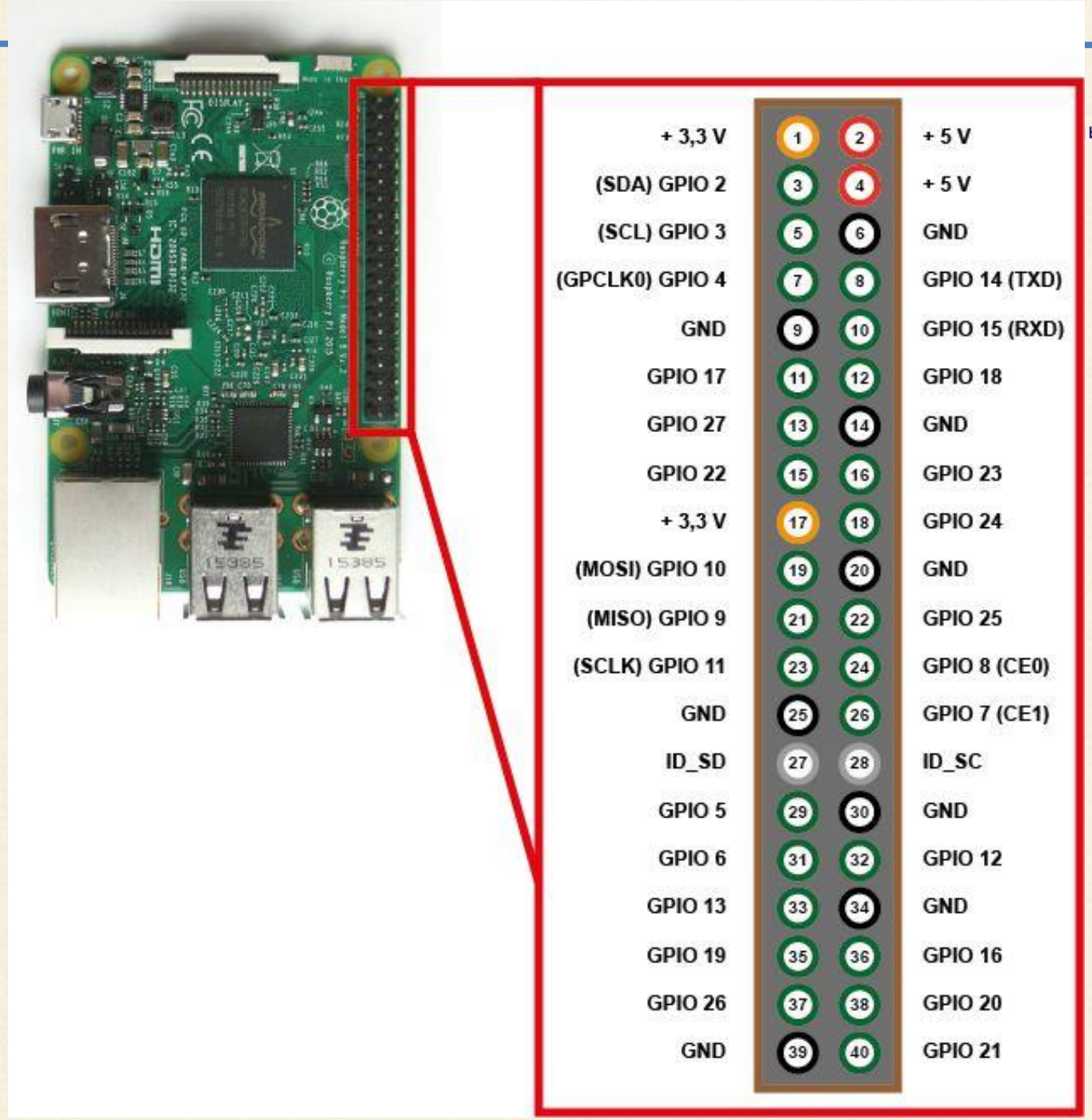






注意針腳順序不一樣!!插錯會燒掉

腳位參考圖

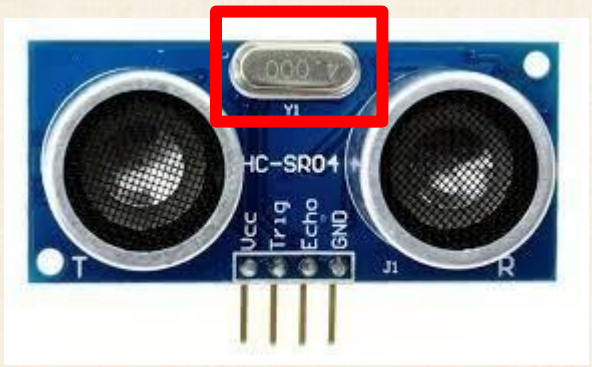


BUN LAB
band Ubiquitous Networking Lab

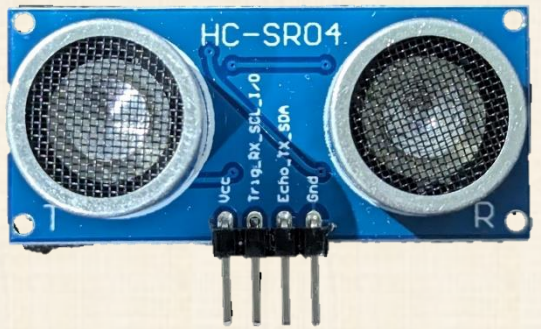
3.超音波感測器

- HC-SR04
- 由超音波發射器、接收器和控制電路所組成。當它被觸發的時候，會發射 40 kHz 的聲波並測量回音。
- Rpi 可透過 Trigger 接腳通知 HC-SR04 進行偵測，而 Echo 接腳用來接收偵測的結果。

VCC (電源輸入)
Trig (發射)
Echo (接收)
GND (接地)

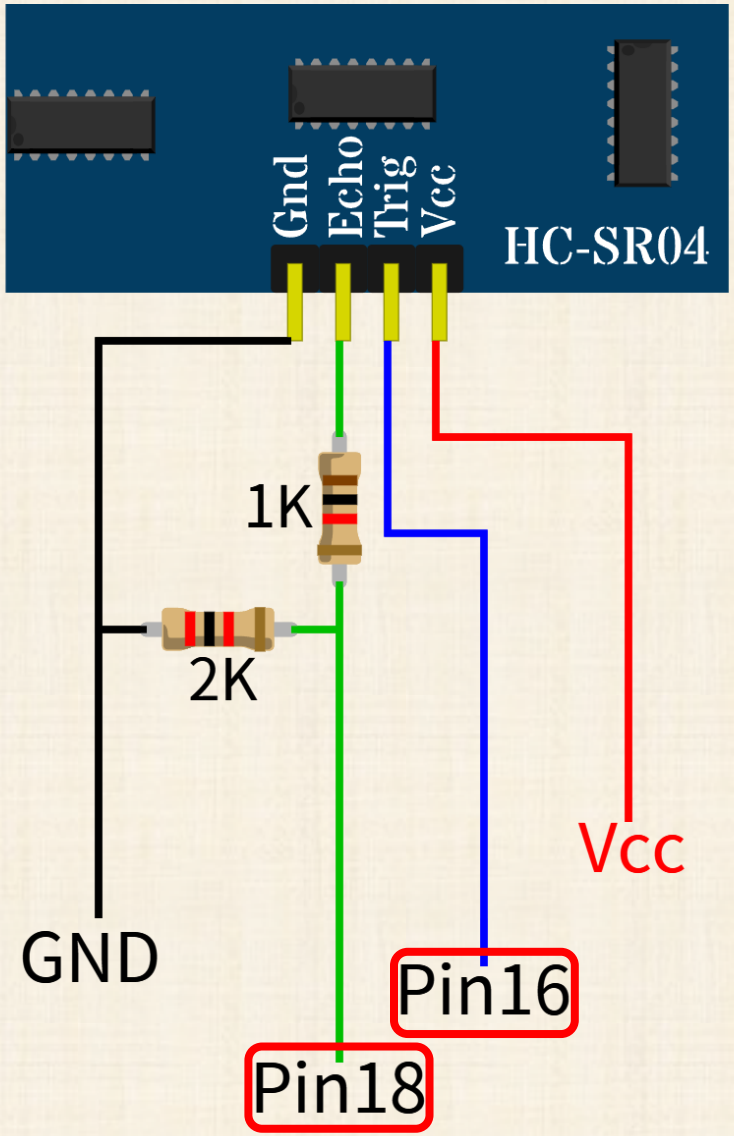


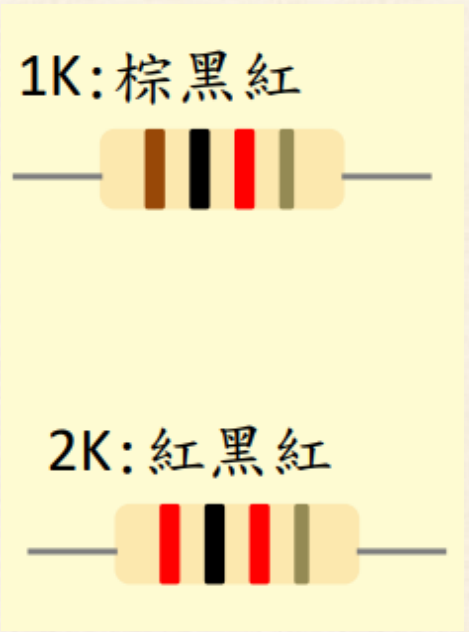
5v



3.3v

電路圖

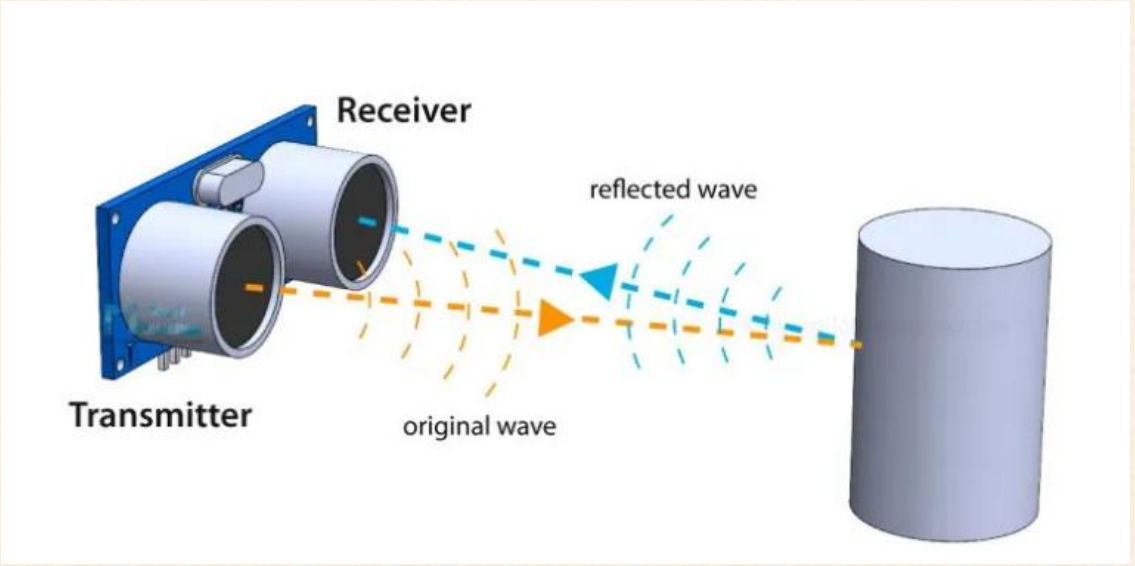

Gnd
Echo
Trig
Vcc
HC-SR04
1K
2K
GND
Vcc
Pin16
Pin18


1K：棕黑紅
2K：紅黑紅

測量距離

```
import RPi.GPIO as GPIO
import time

GPIO.setwarnings(False)

      聲速
v=343
TRIG = 16
E = 18

print '1'
GPIO.setmode(GPIO.BOARD)
GPIO.setup(TRIG,GPIO.OUT)
GPIO.setup(E,GPIO.IN)
GPIO.output(TRIG,GPIO.LOW)
def measure():
        GPIO.output(TRIG, GPIO.HIGH)
        time.sleep(0.00001)
        GPIO.output(TRIG, GPIO.LOW)
        pulse_start = 0
        pulse_end =0
        while GPIO.input(E) == GPIO.LOW:
                pulse_start = time.time()
        while GPIO.input(E) == GPIO.HIGH:
                pulse_end = time.time()

        t = pulse_end-pulse_start
        d=t*v
        d=d/2
        return d*100


while(1):
        print measure()
        time.sleep(1)
GPIO.cleanup()
```

Using the ***Trig*** pin we send the ultrasound wave from the transmitter, and with the ***Echo*** pin we listen for the reflected signal.



https://howtomechatronics.com/tutorials/arduino/ultrasonic-sensor-hc-sr04/

Q3

- 距離感測警示燈
  - 當距離 >20 cm，LED燈暗
  - 當距離在 10 ~ 20cm 之間， LED燈快速閃爍
  - 當距離 <10 cm，持續亮燈

Q4 距離感測警示燈2.0

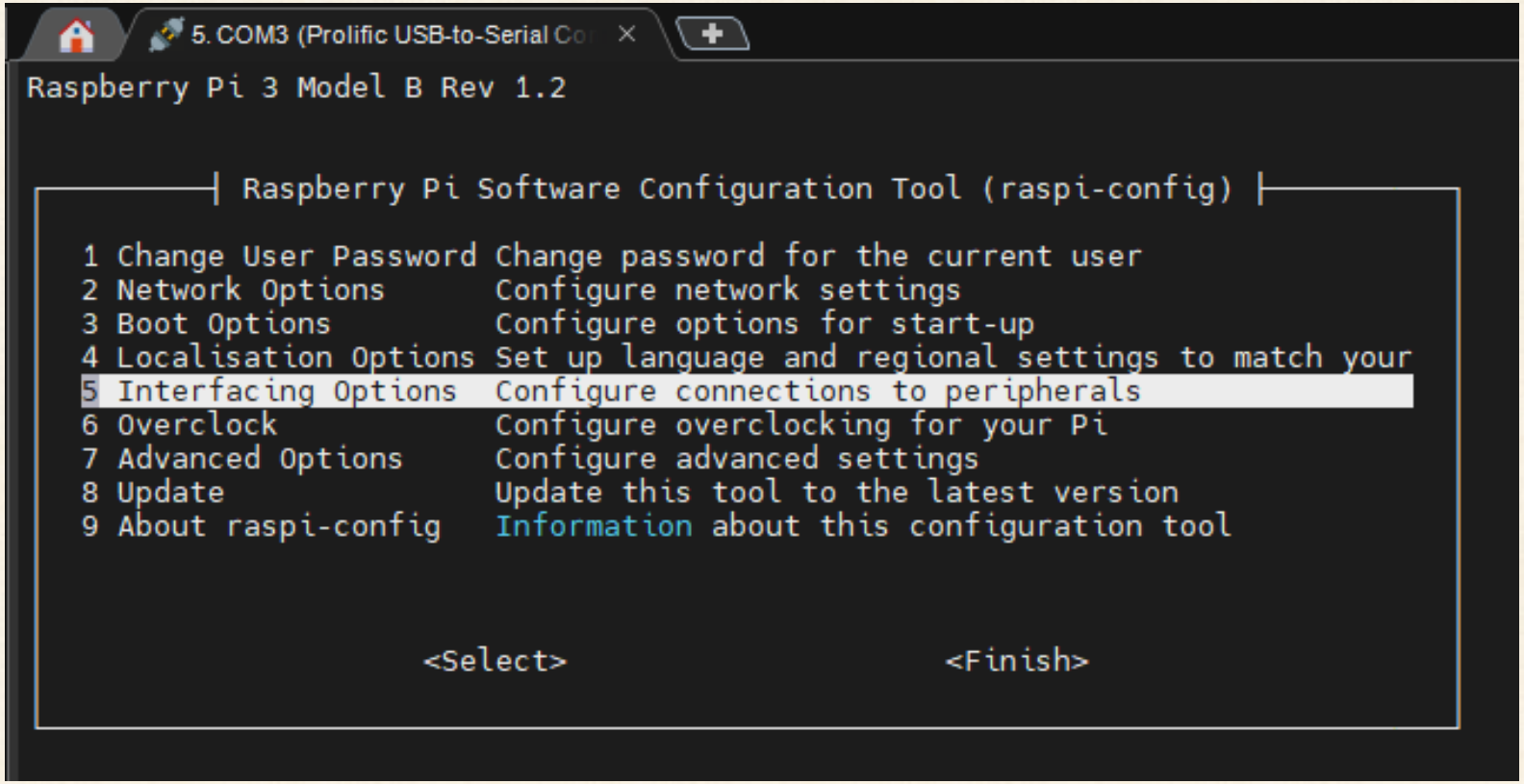
- 將**Q3**速度調整為V=331+0.6×T
  - 溫度 T :由溫溼度感測器測量
- 持續(每秒)偵測溫度及距離，控制LED亮暗
- 要print出當前溫度、公式、結果

```
pi@raspberrypi: ~/lab1
pi@raspberrypi:~/lab1 $ python Q4_temp_dist.py
====================
Temp:25.0*C
V=331+0.6*25.0
 =346.0
Distance:40.5cm
====================
Temp:25.0*C
V=331+0.6*25.0
 =346.0
Distance:38.4cm
====================
Temp:25.0*C
V=331+0.6*25.0
 =346.0
Distance:50.7cm
```

```
import RPi.GPIO as GPIO
import time

GPIO.setwarnings(False)


v=343
TRIG = 16
E = 18

print '1'
GPIO.setmode(GPIO.BOARD)
GPIO.setup(TRIG,GPIO.OUT)
GPIO.setup(E,GPIO.IN)
GPIO.output(TRIG,GPIO.LOW)
def measure():
        GPIO.output(TRIG, GPIO.HIGH)
        time.sleep(0.00001)
        GPIO.output(TRIG, GPIO.LOW)
        pulse_start = 0
        pulse_end =0
        while GPIO.input(E) == GPIO.LOW:
                pulse_start = time.time()
        while GPIO.input(E) == GPIO.HIGH:
                pulse_end = time.time()

        t = pulse_end-pulse_start
        d=t*v
        d=d/2
        return d*100


while(1):
        print measure()
        time.sleep(1)
GPIO.cleanup()
```

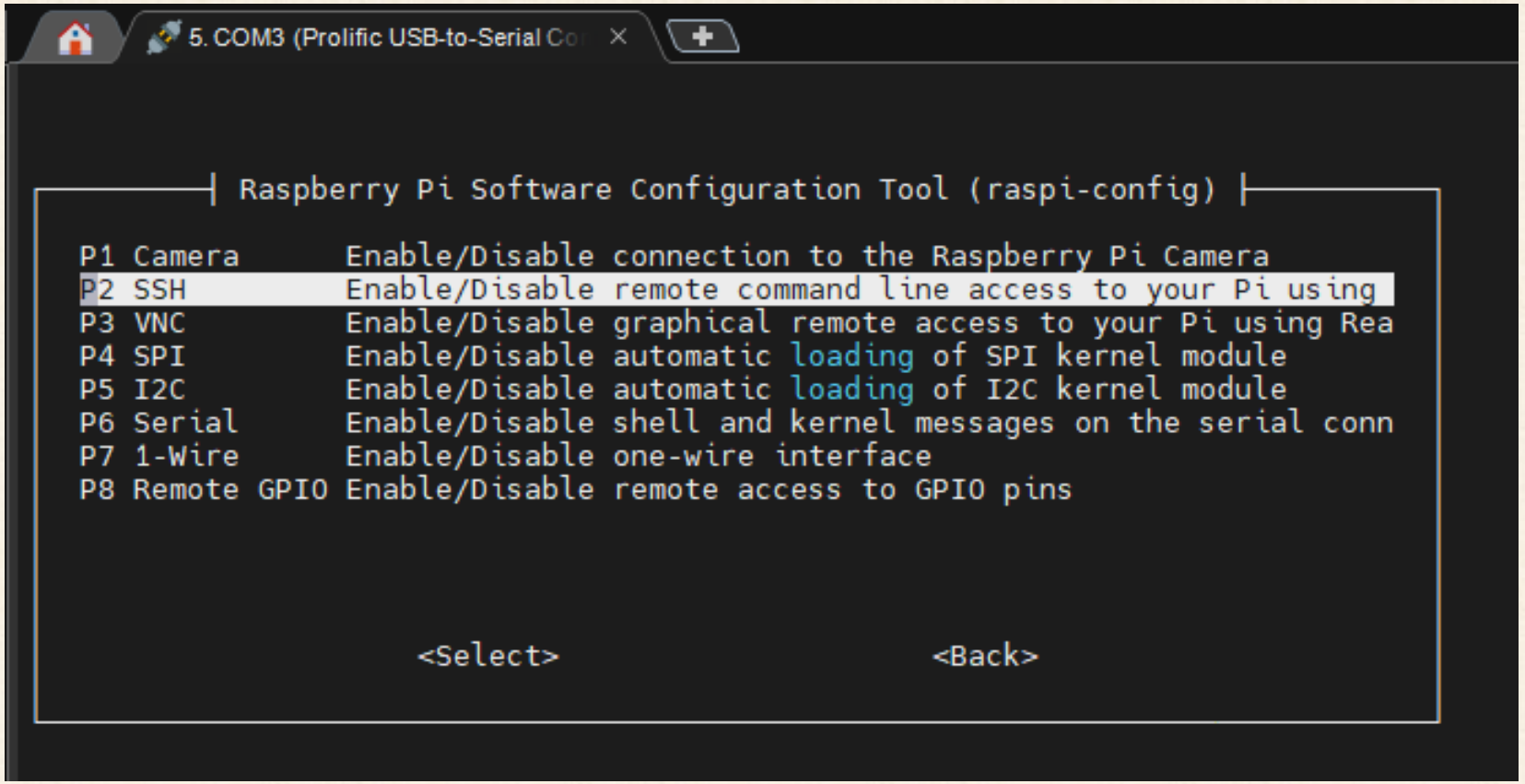
補充-SSH無線連線(1)

- $ sudo raspi-config
- 選擇Interfacing Options



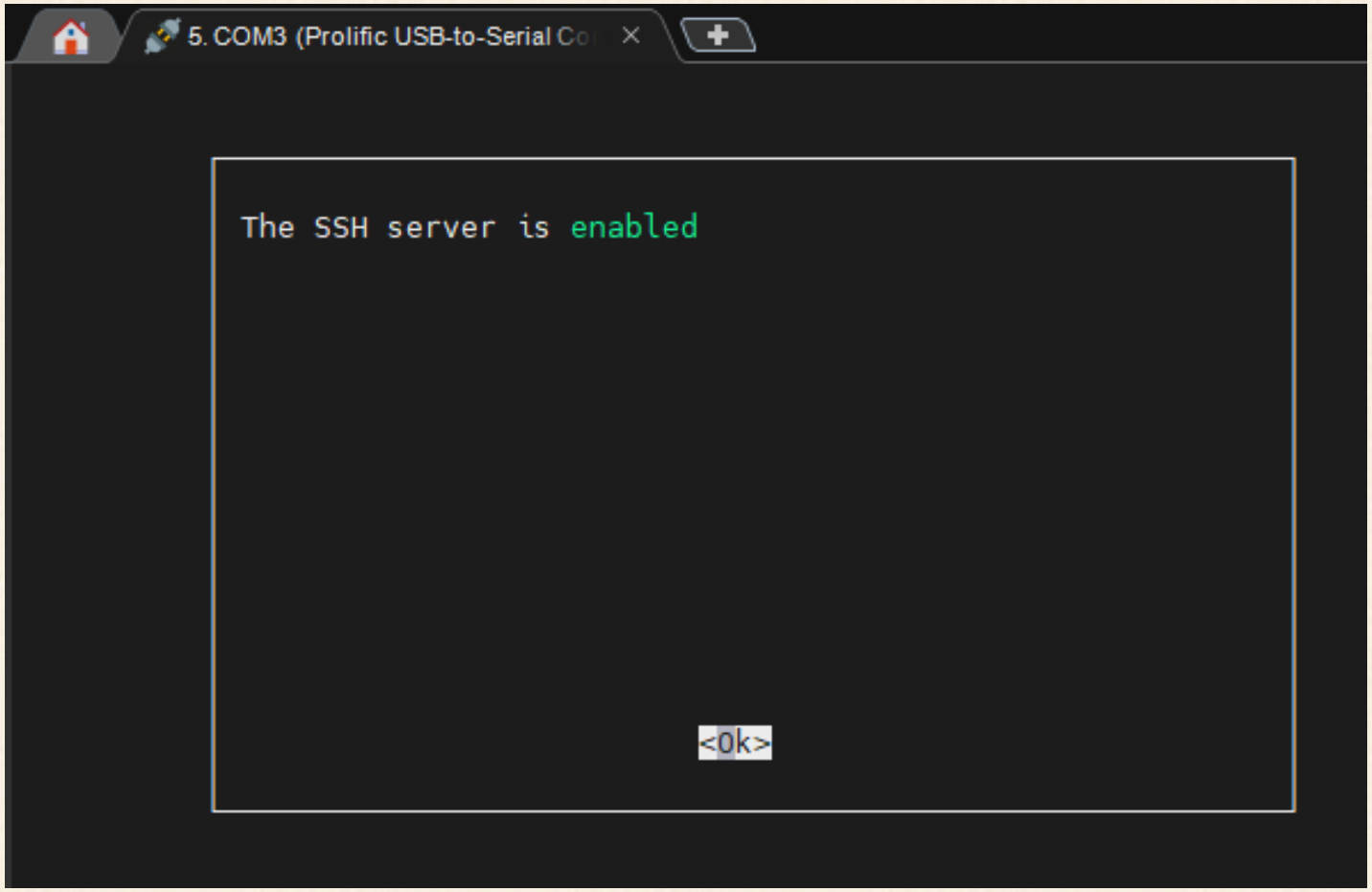
補充-SSH無線連線(2)

- 選擇SSH



補充-SSH無線連線(3)

- <Yes> <Ok> <Finish>



補充-SSH無線連線(4)

- 利用TTL序列連接時，先透過ifconfig查看Pi板IP，及Pi板及電腦是否接上相同Wi-Fi

- 可用 MobaXterm 或 Terminal 或 **VSCode**
  - MobaXterm:
    - Session > SSH > Remote host輸入Pi板IP
  - **VSCode** / Terminal:
    - `$ ssh pi@Pi板IP`

帳號:pi

密碼:raspberry

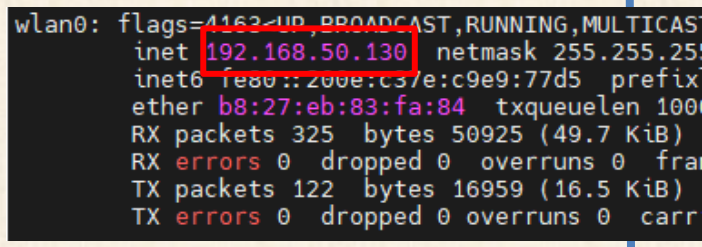
BUN LAB
Broadband Ubiquitous Networking Lab

補充-固定IP

- **修改/etc/dhcpcd.conf**
  - `$ sudo nano /etc/dhcpcd.conf`
  - 加入以下內容

```
interface wlan0                              #無線網路
static ip_address=192.168.X.X                #想固定的IP
static routers=192.168.Y.Y                   #無線基地台地址
static domain_name_servers=192.168.Y.Y       #無線基地台地址
```

  - Ctrl+O 存檔，Ctrl+X 跳出 nano
- **重開機**
  - `$ sudo reboot`



HINT:請固定為AP給的IP，避免重複

```
$ ifconfig
ip_address=192.168.50.130
routers=192.168.50.1
domain_name_servers=192.168.50.1
```

補充- raspi-config update

- `$ sudo raspi-config`
- 選擇8 Update
- DeBug” There was an error running option 8 Update ”
  - `$ sudo apt update`
  - 同意*2
  - `$ sudo raspi-config`
  - 再次選擇Update

Reference

- From 台灣樹莓派
  - 官網
    - https://piepie.com.tw/
  - github程式碼
    - https://github.com/raspberrypi-tw
- [VScode]Remote Development using SSH
  - https://code.visualstudio.com/docs/remote/ssh