

# 通訊網路實驗

Topic 2 - Lab 4  
紅外線發射與接收  
114學年度 第一學期

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

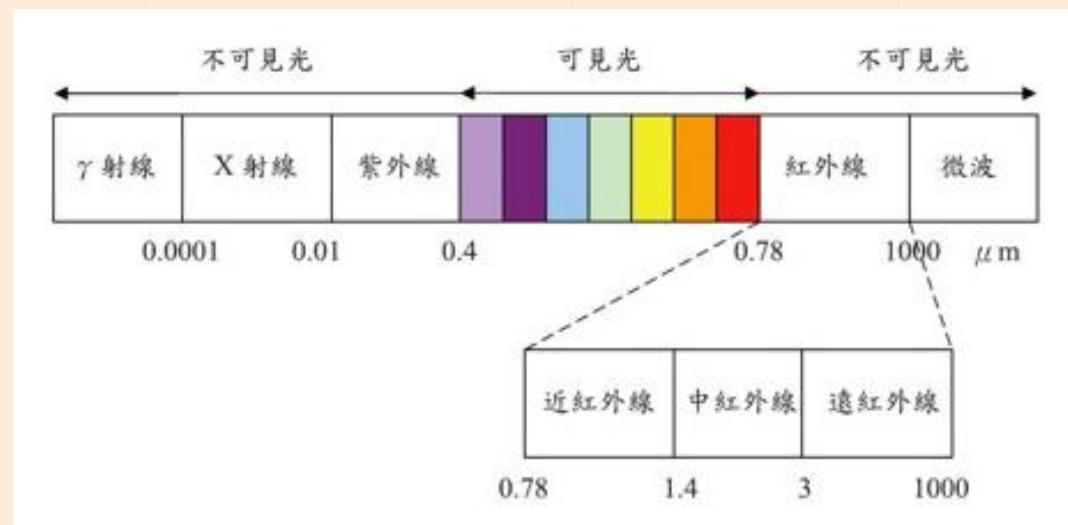
# 目錄

- 一、紅外線
- 二、GPIO
- 三、紅外線模組設定
- 四、實驗 / 結報內容

# 一、紅外線

# 紅外線介紹

- 紅外線是目前常見的一種無線通訊技術，普遍使用於家電以及玩具產品
  - ◆ 例如：電視、音響、冷氣機...等
- 紅外線因為波長 (8XX ~ 940 nm) 與可見光不同，所以肉眼無法看見



# 紅外線介紹

- 日常生活中常見的紅外線發射器，大部份都是在遙控器上頭，把電視機遙控器拿起來看一下就會看到。
- 人的眼睛雖然看不見紅外線，但是相機或手機的感光元件，可以感應紅外線，因此可以利用相機或手機的拍照功能，來檢測紅外線遙控器是否正常運作。
- 今天的實驗要利用紅外線遙控器錄製按鍵控制 LED 的亮與暗

# —、GPIO

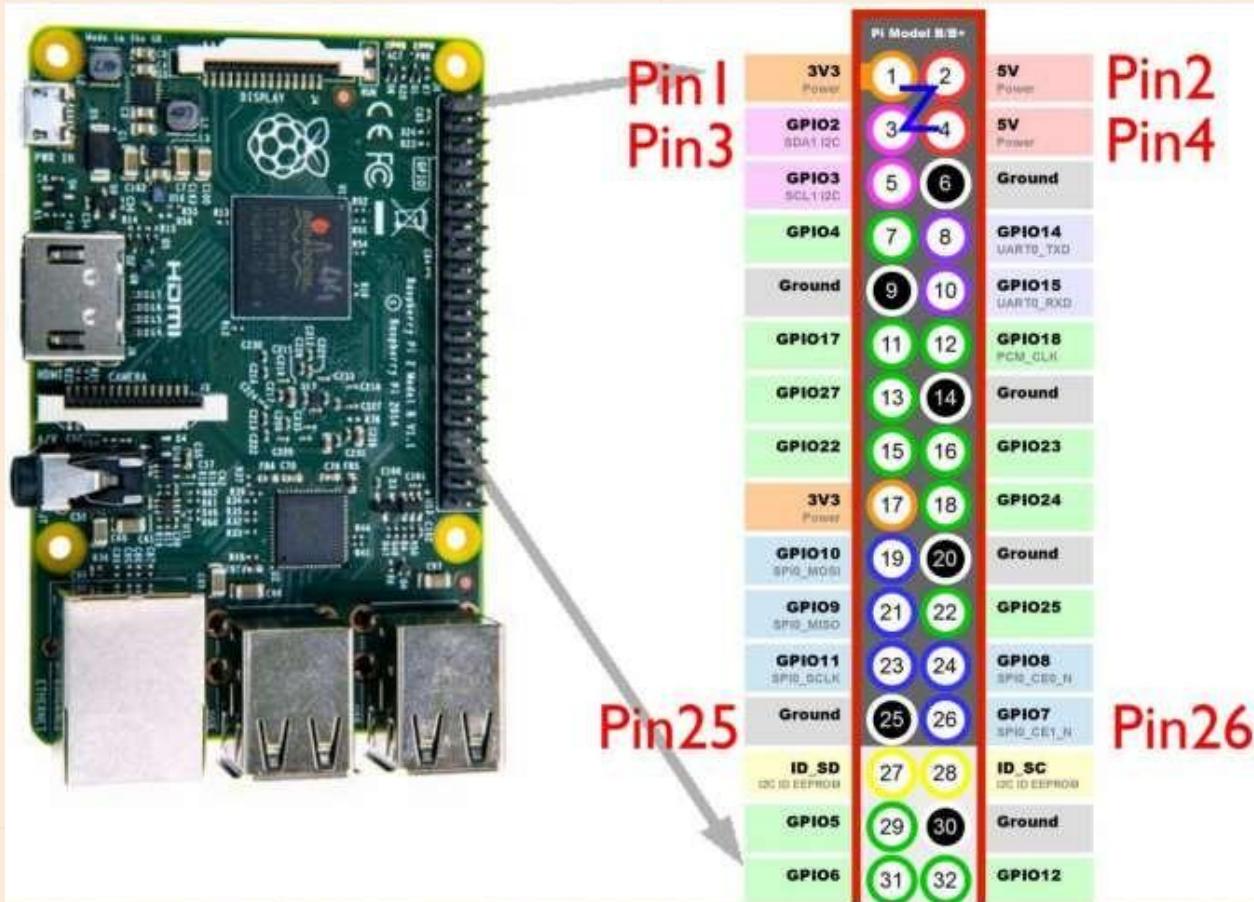
# GPIO on Raspberry Pi

- 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
  - ◆ 可用軟體控制的數位訊號



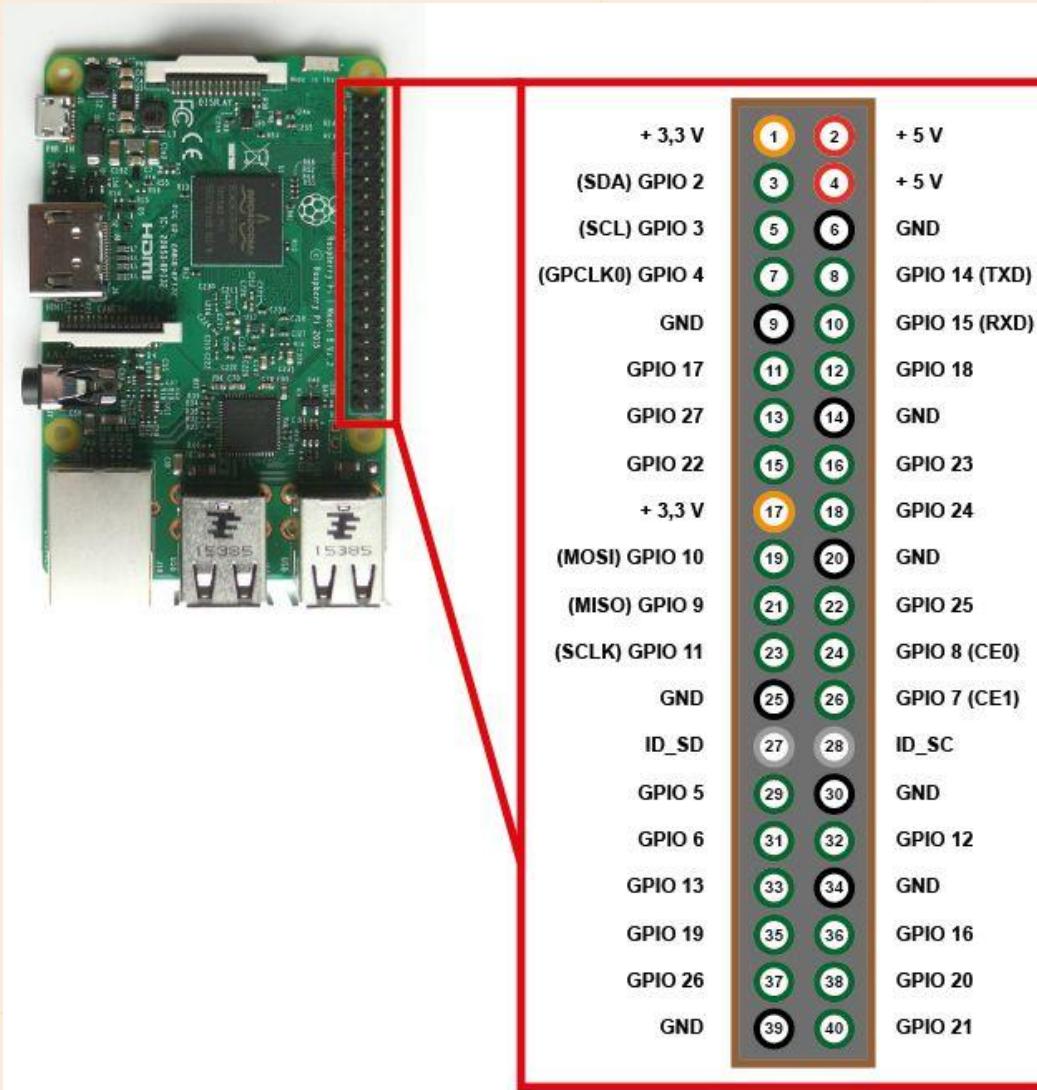
# GPIO on Raspberry Pi

- Z字型的腳位編號 (GPIO.BOARD)



# 腳位參考圖

- GPIO.BOARD
- GPIO.BCM

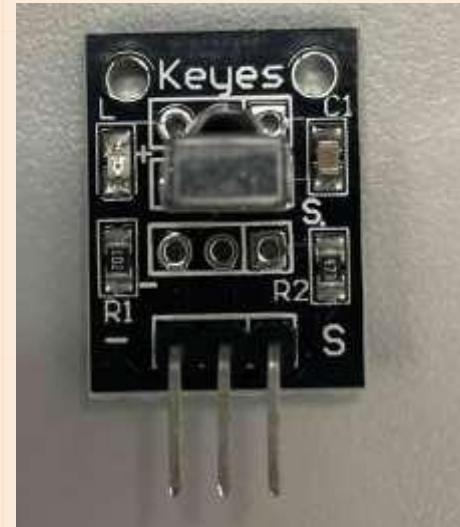


# 使用 GPIO 的注意事項

- 在 GPIO PIN 上不可以輸入超過 3.3 V
- GPIO PIN 的電流上限是 3 mA
- 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 5 V
  - ◆ 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 50 mA
- 供電腳位 (5V) 不可以超過 250 mA

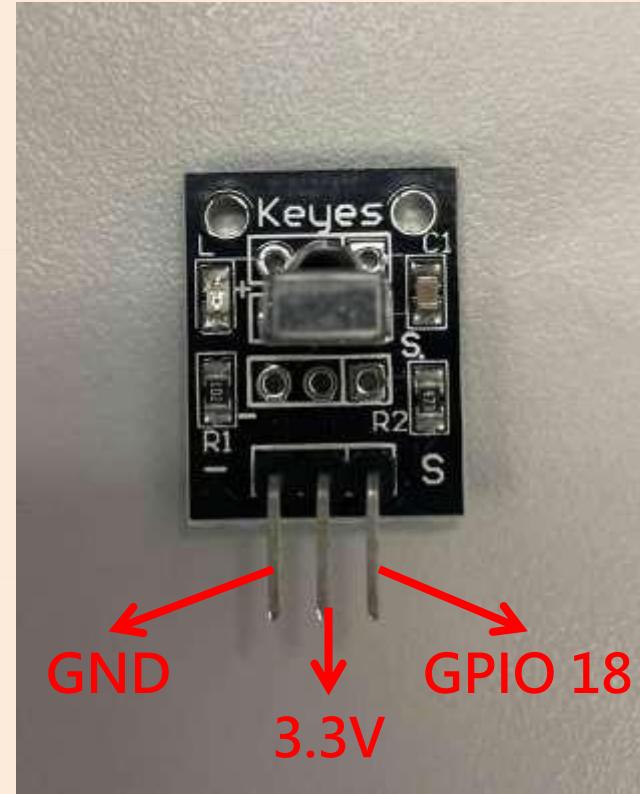
# 本次實驗材料

- Raspberry Pi
- LED燈 \* 2
- 遙控器
- KY-022 紅外線接收模組
- 杜邦線



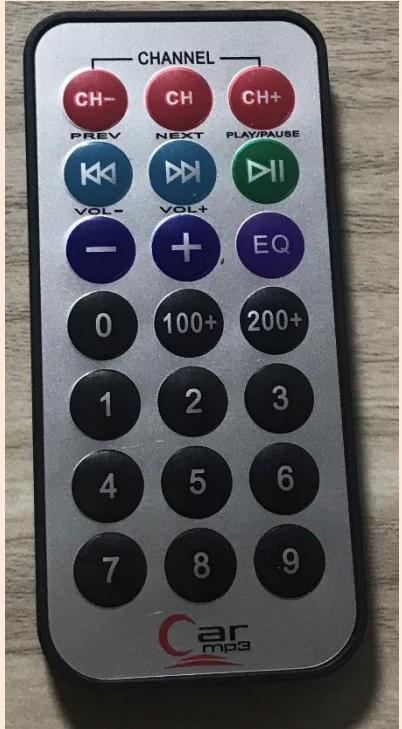
# 紅外線接收模組接線

- KY - 022
- 圖片中 最左邊的腳位 GND 接地
- 中間的接上 3.3V
- 最右邊的腳位接到 GPIO 18 (**Pin 12**)



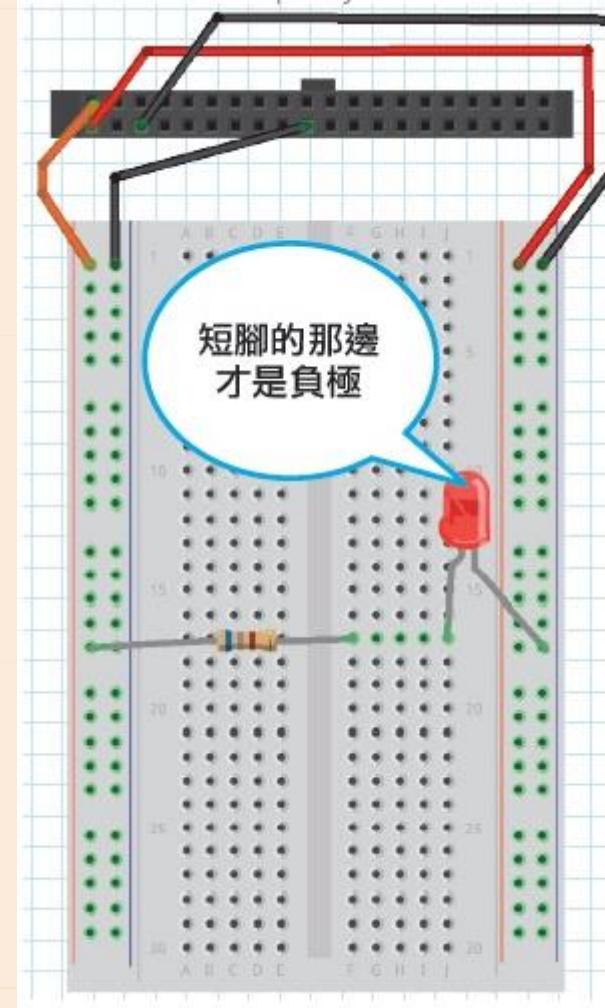
# 紅外線遙控器

- 請先**檢查遙控器是否裝上電池**



CR2025 電池

# LED 燈接線



# 三、紅外線模組設定

# 下載本次實驗函式庫

- lirc
  - ◆ sudo apt update
  - ◆ sudo apt install lirc
  - ◆ sudo apt-get install liblircclient-dev
- Python-GPIO
  - ◆ sudo pip install rpi.gpio

# 下載本次實驗函式庫

- Python-lirc
  - ◆ sudo python -m pip install cython
  - ◆ git clone <https://github.com/tompreston/python-lirc.git>
  - ◆ cd python-lirc/
  - ◆ python setup.py build
  - ◆ sudo python setup.py install
- 程式碼已放在E3

# 設定紅外線模組

- sudo nano /etc/modules
  - ◆ 編輯 modules 這個檔案

```
GNU nano 3.2          /etc/modules

# /etc/modules: kernel modules to load at boot time.
#
# This file contains the names of kernel modules that should be loaded
# at boot time, one per line. Lines beginning with "#" are ignored.

lirc_dev
lirc_rpi gpio_out_pin = 27 gpio_in_pin = 18

i2c-dev
```

加入這兩行，18 為接線的 GPIO 號碼

# 設定紅外線模組

- sudo nano /boot/config.txt
  - ◆ 加入 dtoverlay= gpio-ir, gpio\_pin=18
- sudo nano /etc/lirc/lirc\_options.conf
  - ◆ 找到這兩行

```
driver      = devinput
device      = auto
```
  - ◆ 修改成

```
driver      = default
device      = /dev/lirc0
```
- sudo reboot 重啟樹莓派

```
[all]
#dtoverlay=vc4-fkms-v3d

force_turbo=1
dtoverlay=pi3-miniuart-bt

dtoverlay= gpio-ir, gpio_pin=18

enable_uart=1
```

# 確認模組是否正常運作

- 重新啟動後，輸入 `sudo /etc/init.d/lircd status`

```
● lircd.service - Flexible IR remote input/output application support
  Loaded: loaded (/lib/systemd/system/lircd.service; enabled; vendor preset: enabled)
  Active: active (running) since Mon 2021-07-05 09:54:38 BST; 2h 37min ago
    Docs: man:lircd(8)
          http://lirc.org/html/configure.html
  Main PID: 465 (lircd)
     Tasks: 2 (limit: 2200)
   Memory: 1.9M
      CGroup: /system.slice/lircd.service
              └─465 /usr/sbin/lircd --nodaemon

Jul 05 09:59:51 raspberrypi lircd[465]: lircd-0.10.1[465]: Notice: accepted...ircd
Jul 05 09:59:51 raspberrypi lircd-0.10.1[465]: Notice: accepted new client o...rcd
Jul 05 09:59:51 raspberrypi lircd[465]: lircd-0.10.1[465]: Info: removed client
Jul 05 09:59:51 raspberrypi lircd-0.10.1[465]: Info: removed client
Jul 05 09:59:53 raspberrypi lircd[465]: lircd-0.10.1[465]: Notice: accepted...ircd
Jul 05 09:59:53 raspberrypi lircd-0.10.1[465]: Notice: accepted new client o...rcd
Jul 05 09:59:53 raspberrypi lircd[465]: lircd-0.10.1[465]: Info: removed client
Jul 05 09:59:53 raspberrypi lircd-0.10.1[465]: Info: removed client
Jul 05 10:01:14 raspberrypi lircd[465]: lircd-0.10.1[465]: Info: removed client
Jul 05 10:01:14 raspberrypi lircd-0.10.1[465]: Info: removed client
Hint: Some lines were ellipsized, use -l to show in full.
```

- 在 Active 顯示 `active (running)` 後，便能夠測試紅外線模組是否能夠接收遙控器的訊號了

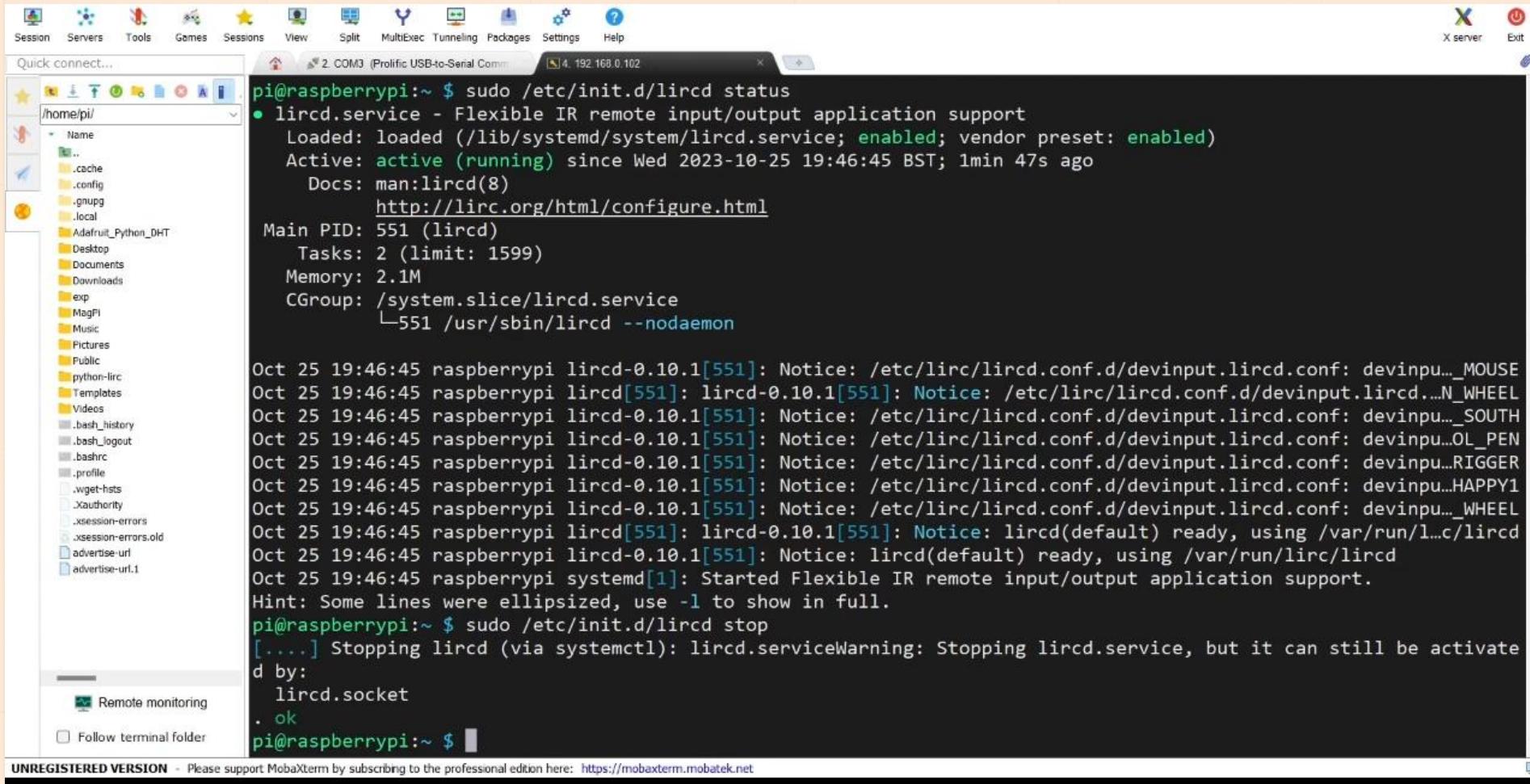
# 確認模組是否正常運作

- 首先，先輸入指令 `sudo /etc/init.d/lircd stop`
- 接著再輸入 `mode2 -d /dev/lirc0`
- 最後按下遙控器上任意按鈕，會得到類似下面圖，表示接收器能夠正常收到訊號

```
Using driver default on device /dev/lirc1
Trying device: /dev/lirc1
Using device: /dev/lirc1
pulse 9298
space 4443
pulse 675
space 468
pulse 676
space 466
pulse 675
space 466
pulse 674
space 469
pulse 673
space 469
pulse 676
space 467
pulse 676
space 467
pulse 675
space 466
pulse 678
space 1573
pulse 675
space 1575
pulse 652
space 1598
pulse 654
space 1597
pulse 653
space 1599
pulse 653
space 1600
pulse 679
space 1573
pulse 652
space 1598
```

# 確認模組是否正常運作

- 此為示範影片，可開啟.ppt檔觀看



pi@raspberrypi:~ \$ sudo /etc/init.d/lircd status

- lircd.service - Flexible IR remote input/output application support

  Loaded: loaded (/lib/systemd/system/lircd.service; enabled; vendor preset: enabled)

  Active: active (running) since Wed 2023-10-25 19:46:45 BST; 1min 47s ago

    Docs: man:lircd(8)  
          <http://lirc.org/html/configure.html>

    Main PID: 551 (lircd)

      Tasks: 2 (limit: 1599)

      Memory: 2.1M

      CGroup: /system.slice/lircd.service  
          └─551 /usr/sbin/lircd --nodaemon

Oct 25 19:46:45 raspberrypi lircd-0.10.1[551]: Notice: /etc/lirc/lircd.conf.d/devinput.lircd.conf: devinpu...\_MOUSE

Oct 25 19:46:45 raspberrypi lircd-0.10.1[551]: Notice: /etc/lirc/lircd.conf.d/devinput.lircd...N\_WHEEL

Oct 25 19:46:45 raspberrypi lircd-0.10.1[551]: Notice: /etc/lirc/lircd.conf.d/devinput.lircd.conf: devinpu...\_SOUTH

Oct 25 19:46:45 raspberrypi lircd-0.10.1[551]: Notice: /etc/lirc/lircd.conf.d/devinput.lircd.conf: devinpu...OL\_PEN

Oct 25 19:46:45 raspberrypi lircd-0.10.1[551]: Notice: /etc/lirc/lircd.conf.d/devinput.lircd.conf: devinpu...RIGGER

Oct 25 19:46:45 raspberrypi lircd-0.10.1[551]: Notice: /etc/lirc/lircd.conf.d/devinput.lircd.conf: devinpu...HAPPY1

Oct 25 19:46:45 raspberrypi lircd-0.10.1[551]: Notice: /etc/lirc/lircd.conf.d/devinput.lircd.conf: devinpu...\_WHEEL

Oct 25 19:46:45 raspberrypi lircd-0.10.1[551]: Notice: lircd(default) ready, using /var/run/lirc/lircd

Oct 25 19:46:45 raspberrypi lircd-0.10.1[551]: Notice: lircd(default) ready, using /var/run/lirc/lircd

Oct 25 19:46:45 raspberrypi systemd[1]: Started Flexible IR remote input/output application support.

Hint: Some lines were ellipsized, use -l to show in full.

pi@raspberrypi:~ \$ sudo /etc/init.d/lircd stop

[...] Stopping lircd (via systemctl): lircd.serviceWarning: Stopping lircd.service, but it can still be activate

d by:

  lircd.socket

  . ok

pi@raspberrypi:~ \$

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

# 錄製遙控器按鈕進 Pi

- 輸入指令 `irrecord -d /dev/lirc0 ~/lircd.conf`
- 輸入 config 檔的名稱，在此以 `LED_test` 為範例

```
Enter name of remote (only ascii, no spaces) :LED_test
Using LED_test.lircd.conf as output filename
```

- 接著就可以按照終端機上指示的步驟做錄製

```
Now start pressing buttons on your remote control.

It is very important that you press many different buttons randomly
and hold them down for approximately one second. Each button should
generate at least one dot but never more than ten dots of output.
Don't stop pressing buttons until two lines of dots (2x80) have
been generated.

Press RETURN now to start recording.
.....
Got gap (39909 us)}
```

※錄製按鍵過程中，遙控器上每一個鈕都要按到。按一次會出現一個點，建議一次按一個按鈕，可以稍微按快一點。

# 錄製遙控器按鈕進 Pi

- 錄製總共會有兩個步驟
- 結束之後會需要為按鍵取名，這裡需要4個按鈕
  - ◆ 統一命名為KEY\_# (1~4)

```
please enter the name for the next button (press <ENTER> to finish recording)
KEY_1

Now hold down button "KEY_1".

please enter the name for the next button (press <ENTER> to finish recording)
KEY_2

Now hold down button "KEY_2".

please enter the name for the next button (press <ENTER> to finish recording)
KEY_3

Now hold down button "KEY_3".

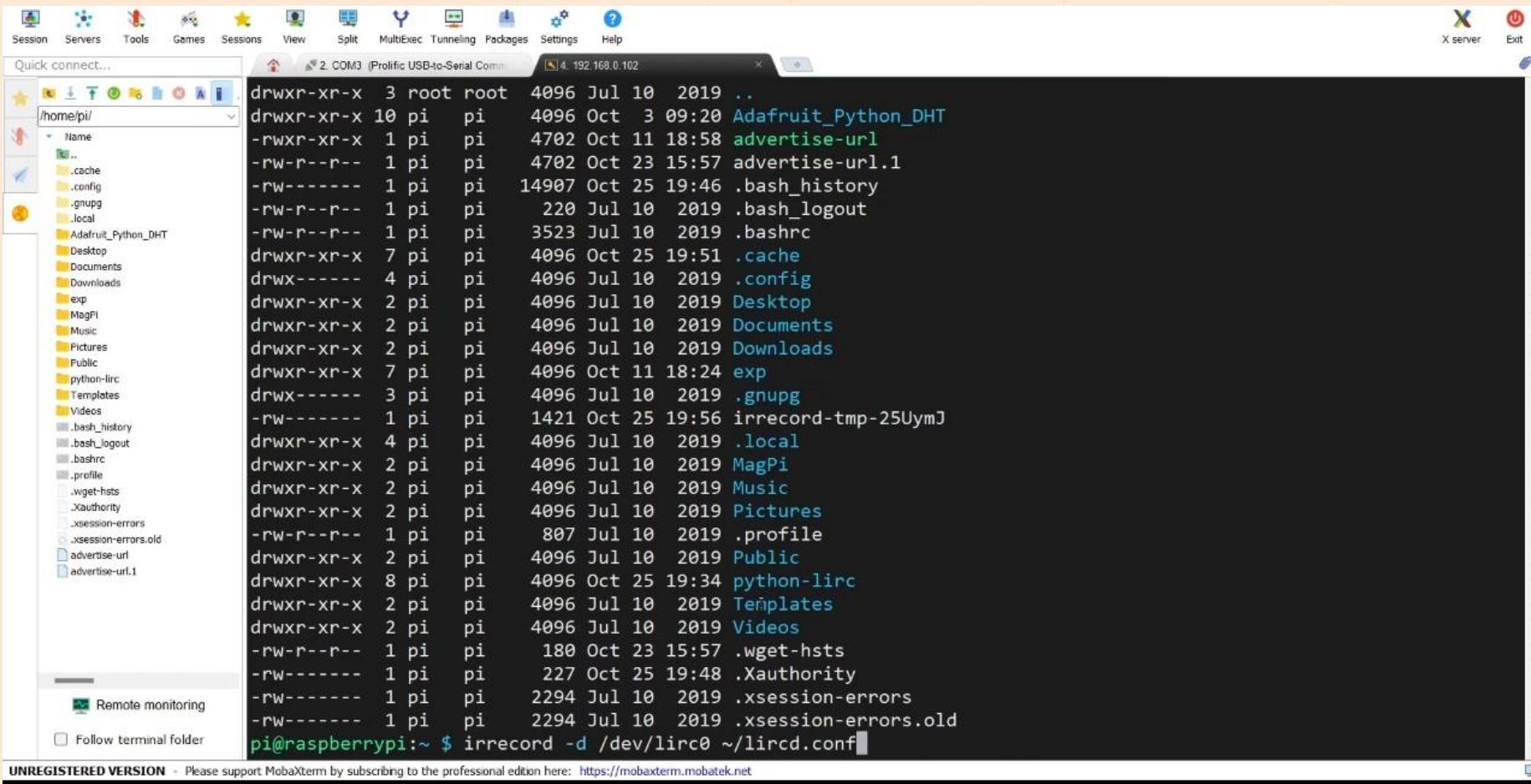
please enter the name for the next button (press <ENTER> to finish recording)
KEY_4

Now hold down button "KEY_4".
```

- 當它顯示 "Now hold down button KET\_#" 時，請按住你要設定的那一顆按鈕

# 錄製遙控器按鈕進 Pi

- 此為示範影片，可開啟.ppt檔觀看



The screenshot shows a MobaXterm window with two panes. The left pane is a file browser showing the contents of the /home/pi directory. The right pane is a terminal window displaying the command 'irrecord -d /dev/lirc0 ~/lircd.conf' being run as root. The terminal output lists numerous files and their permissions and timestamps.

```
pi@raspberrypi:~ $ irrecord -d /dev/lirc0 ~/lircd.conf
```

File browser content:

- ..
- Name
- ..
- .cache
- .config
- .gnupg
- .local
- Adafruit\_Python\_DHT
- Desktop
- Documents
- Downloads
- exp
- MagPi
- Music
- Pictures
- Public
- python-lirc
- Templates
- Videos
- .bash\_history
- .bash\_logout
- .bashrc
- .profile
- .wget-hsts
- .Xauthority
- .xsession-errors
- .xsession-errors.old
- advertise-url
- advertise-url.1

Terminal output (partial):

```
drwxr-xr-x 3 root root 4096 Jul 10 2019 ..
drwxr-xr-x 10 pi pi 4096 Oct 3 09:20 Adafruit_Python_DHT
-rw-r--r-- 1 pi pi 4702 Oct 11 18:58 advertise-url
-rw-r--r-- 1 pi pi 4702 Oct 23 15:57 advertise-url.1
-rw----- 1 pi pi 14907 Oct 25 19:46 .bash_history
-rw-r--r-- 1 pi pi 220 Jul 10 2019 .bash_logout
-rw-r--r-- 1 pi pi 3523 Jul 10 2019 .bashrc
drwxr-xr-x 7 pi pi 4096 Oct 25 19:51 .cache
drwx----- 4 pi pi 4096 Jul 10 2019 .config
drwxr-xr-x 2 pi pi 4096 Jul 10 2019 Desktop
drwxr-xr-x 2 pi pi 4096 Jul 10 2019 Documents
drwxr-xr-x 2 pi pi 4096 Jul 10 2019 Downloads
drwxr-xr-x 7 pi pi 4096 Oct 11 18:24 exp
drwx----- 3 pi pi 4096 Jul 10 2019 .gnupg
-rw----- 1 pi pi 1421 Oct 25 19:56 irrecord-tmp-25UymJ
drwxr-xr-x 4 pi pi 4096 Jul 10 2019 .local
drwxr-xr-x 2 pi pi 4096 Jul 10 2019 MagPi
drwxr-xr-x 2 pi pi 4096 Jul 10 2019 Music
drwxr-xr-x 2 pi pi 4096 Jul 10 2019 Pictures
-rw-r--r-- 1 pi pi 807 Jul 10 2019 .profile
drwxr-xr-x 2 pi pi 4096 Jul 10 2019 Public
drwxr-xr-x 8 pi pi 4096 Oct 25 19:34 python-lirc
drwxr-xr-x 2 pi pi 4096 Jul 10 2019 Templates
drwxr-xr-x 2 pi pi 4096 Jul 10 2019 Videos
-rw-r--r-- 1 pi pi 180 Oct 23 15:57 .wget-hsts
-rw----- 1 pi pi 227 Oct 25 19:48 .Xauthority
-rw----- 1 pi pi 2294 Jul 10 2019 .xsession-errors
-rw----- 1 pi pi 2294 Jul 10 2019 .xsession-errors.old
```

# 錄製遙控器按鈕進 Pi

- 錄製結束後須將檔案拷貝到 config 檔裡面
  - ◆ 輸入以下指令: `sudo cp ~/<your-name>.lircd.conf /etc/lirc/lircd.conf`
  - ◆ `your-name` 記得改成自己命名的檔案名稱!!!
- 重啟 lircd 服務: `sudo /etc/init.d/lircd restart`
- 最後在 terminal 輸入 `irw`, 並對著接收器按下剛剛錄製的四顆按鈕
  - ◆ 錄製成功的話會像以下圖片一樣顯示按下的按鈕

```
0000000000ff7a85 00 KEY_3 LED_test
0000000000ff7a85 01 KEY_3 LED_test
0000000000ff10ef 00 KEY_4 LED_test
0000000000ff10ef 01 KEY_4 LED_test
0000000000ff30cf 00 KEY_1 LED_test
0000000000ff30cf 00 KEY_1 LED_test
0000000000ff30cf 01 KEY_1 LED_test
0000000000ff18e7 00 KEY_2 LED_test
0000000000ff18e7 01 KEY_2 LED_test
0000000000ff7a85 00 KEY_3 LED_test
0000000000ff7a85 01 KEY_3 LED_test
```

# 若無法顯示按鈕-解決方法

- 查看.conf檔案：`sudo nano /etc/lirc/lircd.conf`



The diagram illustrates the correction of a configuration file error. On the left, a code block shows four key definitions under 'codes'. The fourth key, 'KEY\_4', has two identical code assignments: '0x00FF10EF' followed by '0xE8393D00'. This second assignment is highlighted with a red box. A red arrow points from this section to the right, where the corrected code block is shown. In the corrected version, the second code entry for 'KEY\_4' has been removed, leaving only the first assignment.

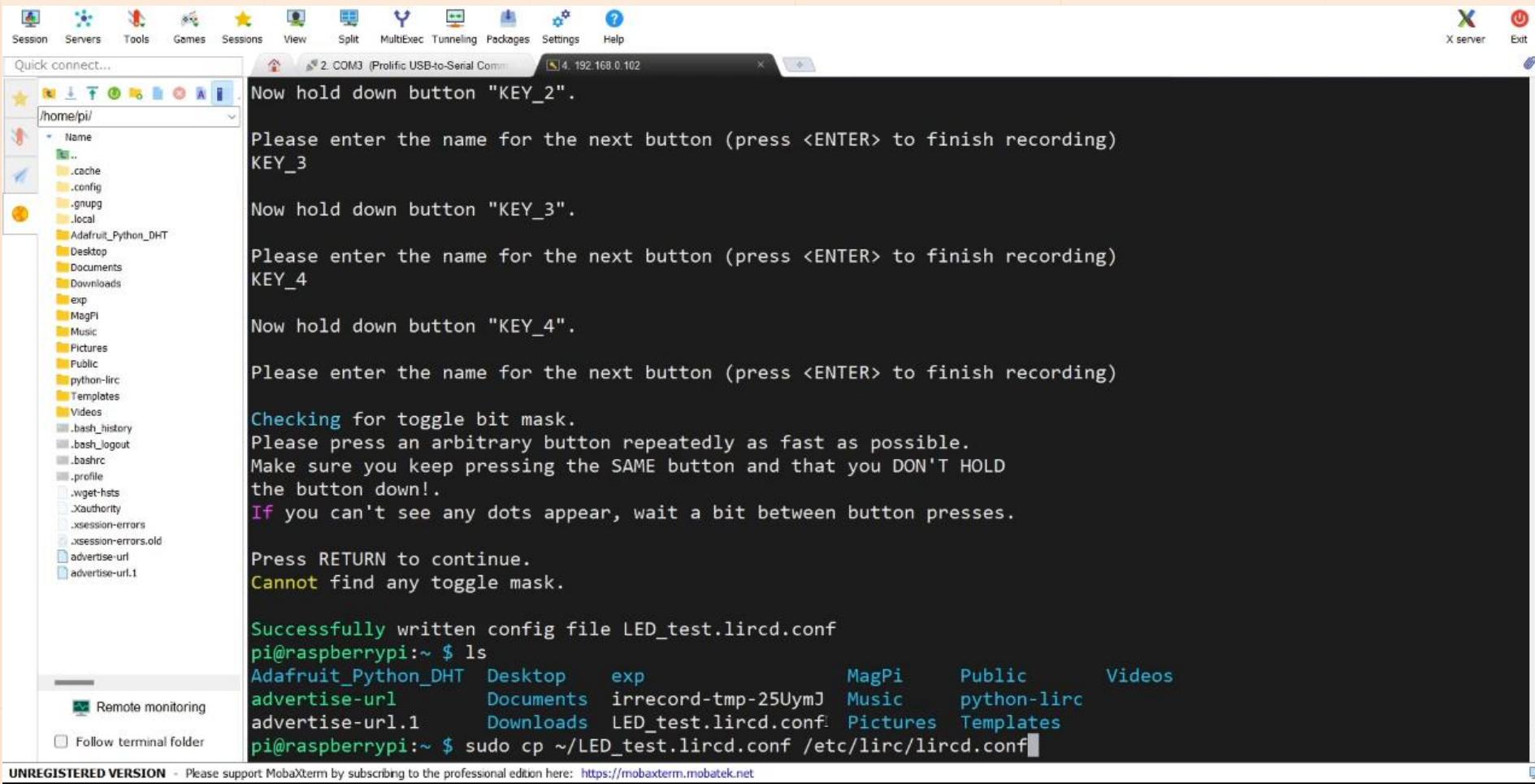
```
begin codes
  KEY_1      0x00FF30CF 0xE8393D00
  KEY_2      0x00FF18E7 0xE8393D00
  KEY_3      0x00FF7A85 0xE8393D00
  KEY_4      0x00FF10EF 0xE8393D00
end codes
```

```
begin codes
  KEY_1      0x00FF30CF
  KEY_2      0x00FF18E7
  KEY_3      0x00FF7A85
  KEY_4      0x00FF10EF
end codes
```

- 如果你的 codes 出現了兩行，請刪除第二行 (框起來的部分)
- 完成之後重啟lircd服務，並再試一次指令 `irw`
- 若還是沒有反應，則重新錄製一次

# 若無法顯示按鈕-解決方法

- 此為示範影片，可開啟.ppt檔觀看



Now hold down button "KEY\_2".  
Please enter the name for the next button (press <ENTER> to finish recording)  
KEY\_3  
Now hold down button "KEY\_3".  
Please enter the name for the next button (press <ENTER> to finish recording)  
KEY\_4  
Now hold down button "KEY\_4".  
Please enter the name for the next button (press <ENTER> to finish recording)  
Checking for toggle bit mask.  
Please press an arbitrary button repeatedly as fast as possible.  
Make sure you keep pressing the SAME button and that you DON'T HOLD  
the button down!.  
If you can't see any dots appear, wait a bit between button presses.  
Press RETURN to continue.  
Cannot find any toggle mask.  
Successfully written config file LED\_test.lircd.conf  
pi@raspberrypi:~ \$ ls  
Adafruit\_Python\_DHT Desktop exp MagPi Public Videos  
advertise-url Documents irrecord-tmp-25UymJ Music python-lirc  
advertise-url.1 Downloads LED\_test.lircd.conf Pictures Templates  
pi@raspberrypi:~ \$ sudo cp ~/LED\_test.lircd.conf /etc/lirc/lircd.conf

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

# 四、實驗 / 結報內容

# 如何控制 PI 的 GPIO

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

我們使用這個

# 如何用 Python 控制 LED 燈

- Python code (控制 LED 閃爍)

```
1 #!/usr/bin/env python
2 import RPi.GPIO as GPIO
3 import time
4
5 GPIO.setmode(GPIO.BCM)
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.BCM:  
按照腳位的順序編號

使用NO. 12的pin腳

判斷、控制

# 如何用 Python 控制紅外線接收

- 工作目錄切換至 /home/pi/
  - ◆ 輸入指令 sudo cd ~ 或 sudo cd /home/pi/
- 使用指令 sudo nano .lircrc 以創立一個 .lircrc 檔案，檔案內容如下
  1. prog : .py 檔的名字
  2. button : 設置按鈕的名字
  3. config : 當按下按鈕後傳送給程式的訊息
  - ◆ 每一個區塊都必須用 begin 、 end 包裹起來

```
begin
    prog = LED
    button = KEY_1
    config = on
end

begin
    prog = LED
    button = KEY_2
    config = off
end
```

# 如何用 Python 控制紅外線接收

- 創建 Python 檔案

```
1 import lirc
2 import RPi.GPIO as GPIO
3
4 GPIO.setmode(GPIO.BCM)
5 GPIO.setwarnings(False)
6 PIN =          #LED_1_PIN
7 GPIO.setup(PIN, GPIO.OUT)
8 PIN_2 =         #LED_2_PIN
9 GPIO.setup(PIN_2, GPIO.OUT)
10
11 sockid = lirc.init('LED', '.lircrc', verbose = True)
12
13 while True:
14     inf = lirc.nextcode()
15     """ Edit Here """
16
17 lirc.deinit()
```

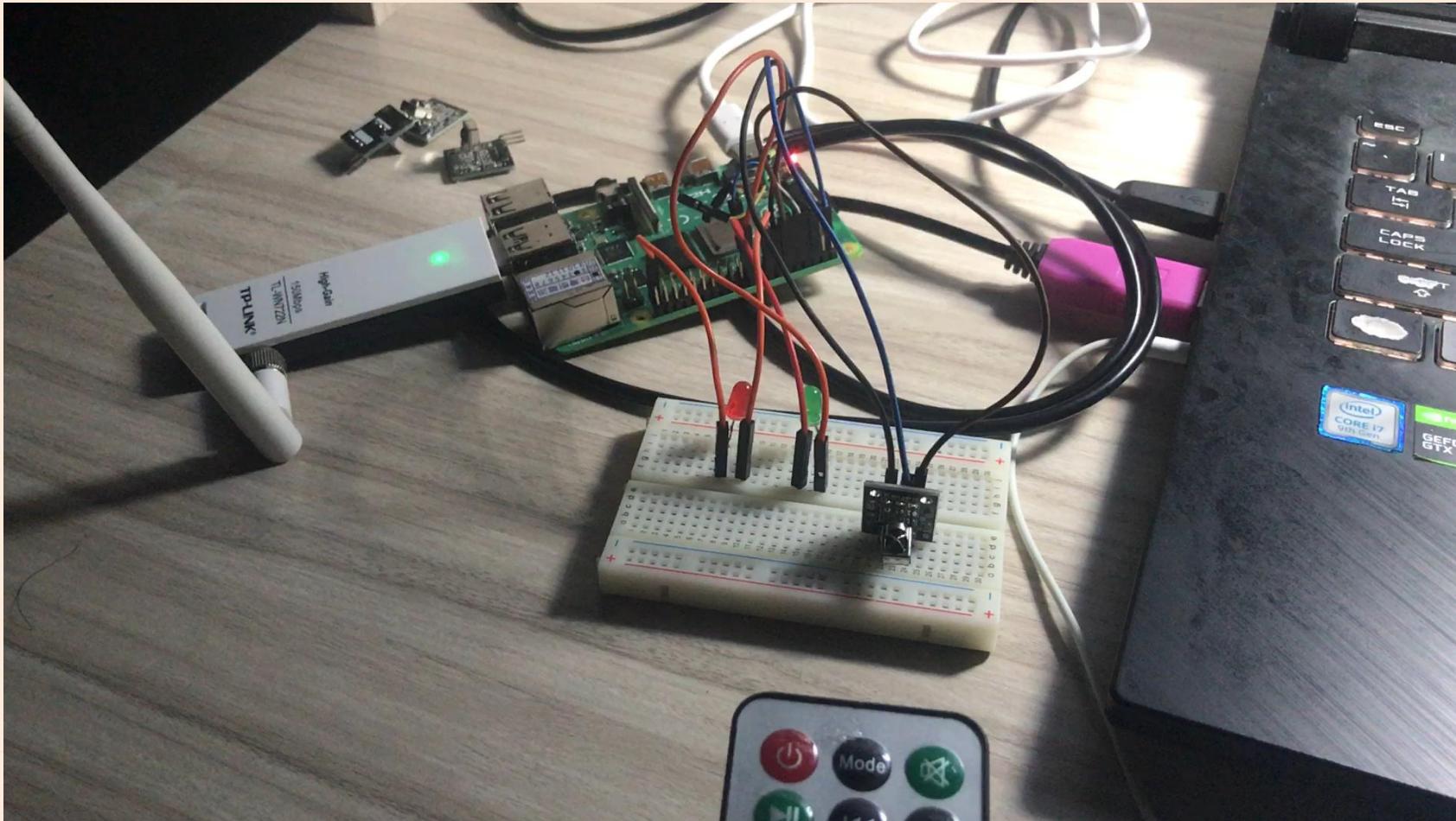
- lirc.nextcode() 為等待接收遙控器的訊號
  - 請嘗試使用接收到的值 (變數 inf) 來控制 LED 的亮暗
  - inf 的格式為 [ u 'xxxx' ] → 編碼 : unicode

# 本次實驗 Demo

- Q1: 請錄製四個按鈕以控制 2 個 LED 燈的亮暗
  - ◆ LED\_1\_ON
  - ◆ LED\_2\_ON
  - ◆ LED\_1\_OFF
  - ◆ LED\_2\_OFF
- 接收遙控器發送的訊號，並根據按鈕使 LED 燈採取相應的行為（亮、暗）

# 本次實驗 Demo

- 此為Demo結果的影片，可開啟.ppt檔觀看



# 本次結報內容

- 1. 在使用紅外線技術傳輸時，可能受到哪些因素影響而無法正常運作？
- 2. 為避免受環境中相同波長的電磁波干擾，一般會在紅外線傳輸訊號時加上載波，試問市面上常見的載波頻率範圍為何？具體加入載波的方式是什麼？請詳細說明。
- 3. 紅外線技術與藍牙技術有什麼差別？各自有什麼優缺點？(越詳細越高分)
- 4. 紅外線技術可以應用在哪些領域？請詳細說明。
- 5. 本次實驗心得，你學到了什麼東西？

# 評分標準 & 注意事項

- 出席 30 %
- Demo 30 %
- 結報 40 %
- 請繳交 .pdf 檔，檔名為 學號\_姓名\_Labx.pdf
- 遲交一天，結報分數扣20%，以此類推

# Reference

- <https://github.com/tompreston/python-lirc/issues/28>
- [https://clover.coex.tech/en/ir\\_sensors.html#install](https://clover.coex.tech/en/ir_sensors.html#install)
- <https://www.jianshu.com/p/eefb1e5d9a23>