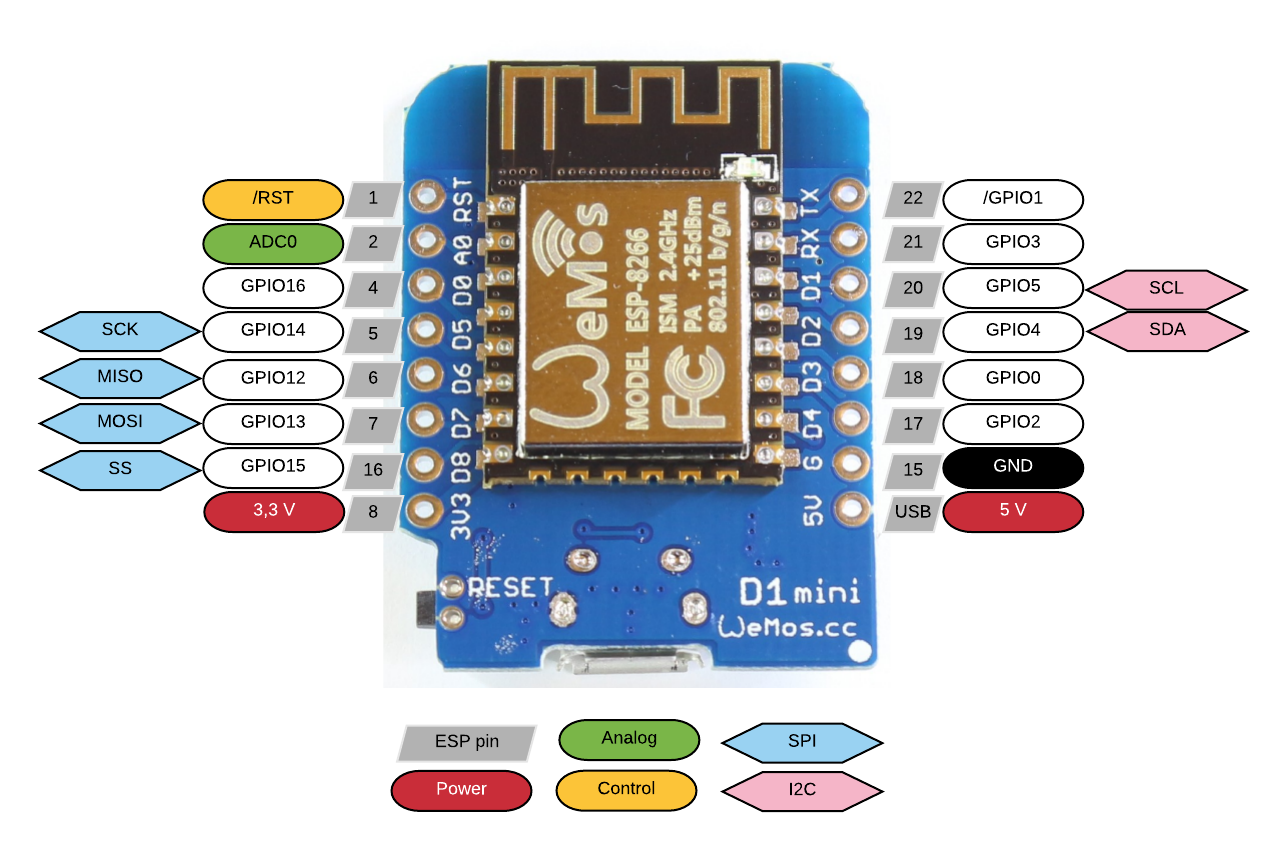
ESP01 & IR 實作

遠端控制一般遙控器

劉家豪



# 目標

擁有WIFI功能的單晶片日漸便宜的時代，適時地將一些控制設備更改為遠端遙控，將有以下優勢：

* 透過人性化界面而非簡單的LCD、TFT的介面操作遙控器，更為人性化。
* 透過遠端遙控，更能簡易的增加智慧居家的應用，如：室內溫度達多少時關閉冷／暖氣、監控家門狀況並遙控開關鐵門……。

遠端遙控一般遙控器將會是趨勢，故推動此項計畫。

# 計畫實作內容

遠端控制一般遙控器以下將對於此項計畫有詳盡的說明與計畫。

## 硬體

以下為此項計畫所需之硬體：

* ESP8266單晶片(型號D1 mini)
* USB to TTL (CP2102 Converter)
* IR接收
* IR發射

## 軟體技術

以下為此項計畫所需之軟體技術：

* 主要技術為C++其中包含以下幾點：
  + 單晶片將使用C++作為後臺，前台顯示HTML進行初次設定（使用ARDUINO IDE）
  + WEB SOCKET與使用者的手機／電腦進行資料傳輸
  + IR的訊號分析、分類
* HTML前台人性化界面
* Node.js後台

## 計畫流程

以下將對於此計畫的使用流程進行簡介：

1. 使用者啟動單晶片，並透過手機／電腦WIFI搜尋單晶片AP。
2. 連上單晶片後，開啟網頁(192.168.0.1)進行初次設定，其中包含：
   1. WIFI帳號、密碼
   2. 使用者帳號、密碼
3. 重啟單晶片，ESP8266將透過設定的WIFI與後台(node.js)連線，並建立使用者資料。
4. 使用者從網頁(<http://www.???.com)登入後，將可以設定／>遙控家中設備。
5. 設定紅外線訊號為以下步驟：
   1. 設定設備名稱
   2. 設定功能名稱
   3. 對單晶片傳送、並重複驗證紅外線訊號
   4. 設定完成
6. 遙控家中設備為以下步驟：
   1. 點選設備
   2. 點擊功能

# 工作步驟及時間安排

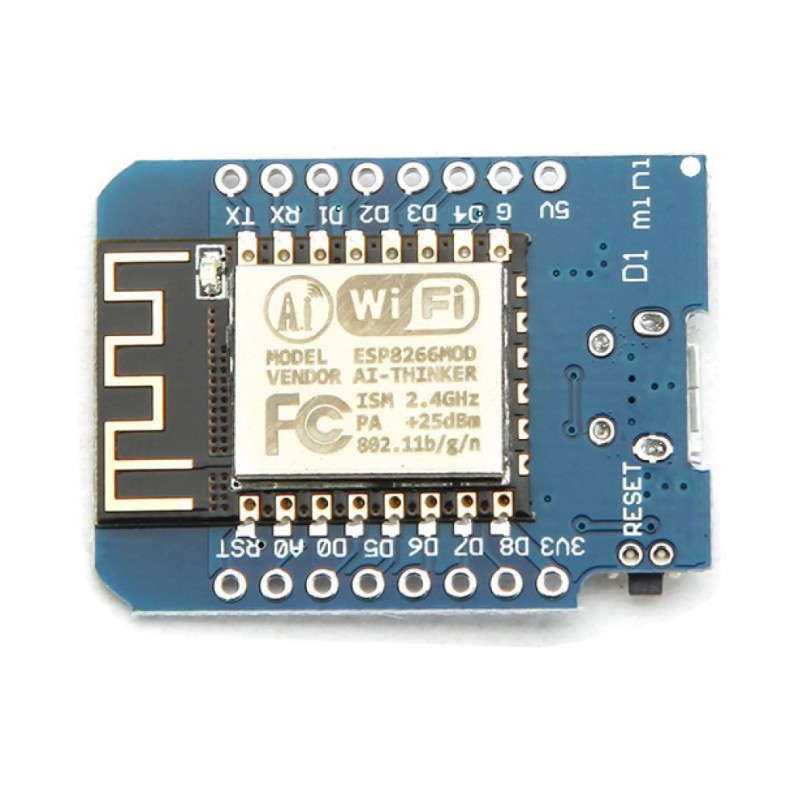
以下為此次計畫的工作步驟及時間安排：

|  |  |  |
| --- | --- | --- |
| 時間 | 工作內容 | 狀態 |
| 11/30 | ESP8266 HTTP SERVER測試  Web Socket Client 測試 | 成功 |
| 12/01 | IR設備購買  ESP8266 IR接收測試 | 成功 |
| 12/02 | ESP8266 IR發送測試  計劃書 | 成功 |
| 12/08 | ESP8266 嵌入HTML前台網頁  FS測試 |  |
| 12/15 | ESP8266 系統完成（初次設定、IR訊號接收並設定、IR訊號發射） |  |
| 12/23 | Node.js與前台網頁完成，並可以接受ESP8266的連線 |  |
| 01/13 | 測試並修正系統 |  |

## 初始設定

1. 硬體設定

首先準備ESP8266（D1 mini），ESP8266主要是指核心，而ESP 01S 則是指AI-Thinker模塊，下圖為ESP D1 mini的模塊：



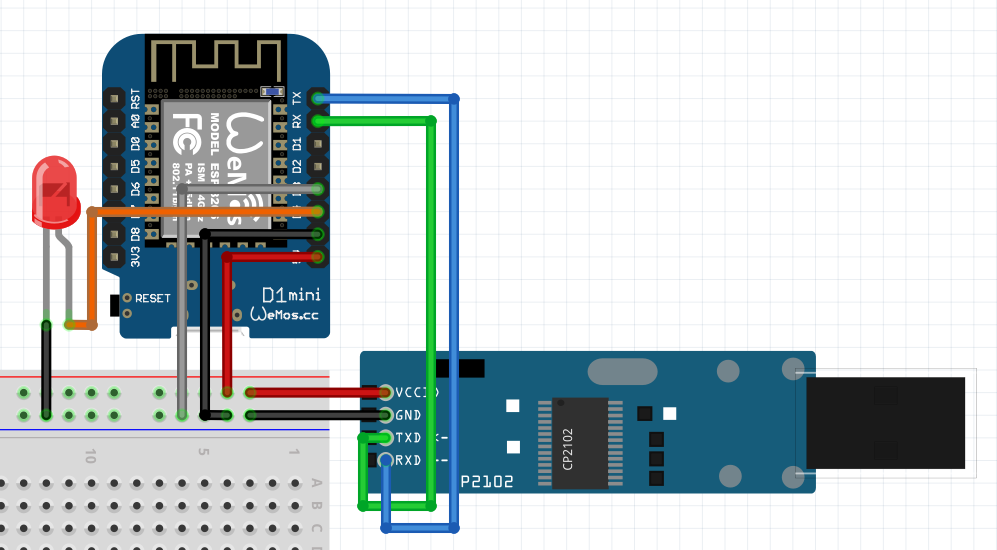
要燒入程式還需要USB to TTL(CP2102 Converter)的硬體，如下圖：



燒入模式接線（IO0接地）：

* ESP D1 mini TX 對 CP2102 Converter RX
* ESP D1 mini RX 對 CP2102 Converter TX
* ESP D1 mini 3.3V 接 3.3V
* ESP D1 mini GND 接 GND
* ESP D1 mini IO0 接GND

接線圖如下：



1. 軟體設定

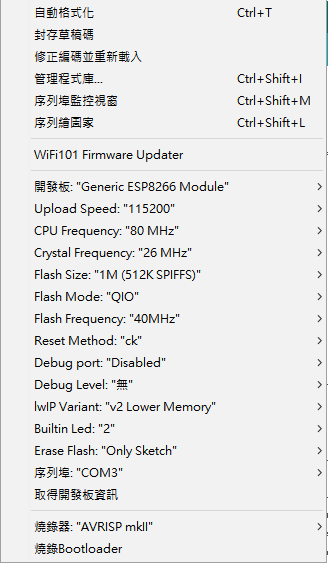
USB to TTL(CP2102 Converter)需要下載驅動程式，連結如下：<https://www.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers>

並透過 本機->管理->裝置管理員->連接埠（COM和LPT）->CP2102 USB to UART Bridge Controller ->右鍵點擊->Update Driver Sovtware…->Browse my Computer for driver software 中選取驅動程式。

由於Arduino IDE的玩家眾多，因此在此項計畫使用Arduino IDE來進行，以便查詢/獲取工具，而軟體設定如下：

* 1. 下載Arduino IDE
  2. 開啟Arduino IDE的設定，偏好設定->額外的開發板管理員網址，填入 <http://arduino.esp8266.com/stable/package_esp8266com_index.json>。
  3. 開啟工具->開發板管理員，進行更新。
  4. 透過需要燒入的型號及需求去設定Arduino IDE的資訊。

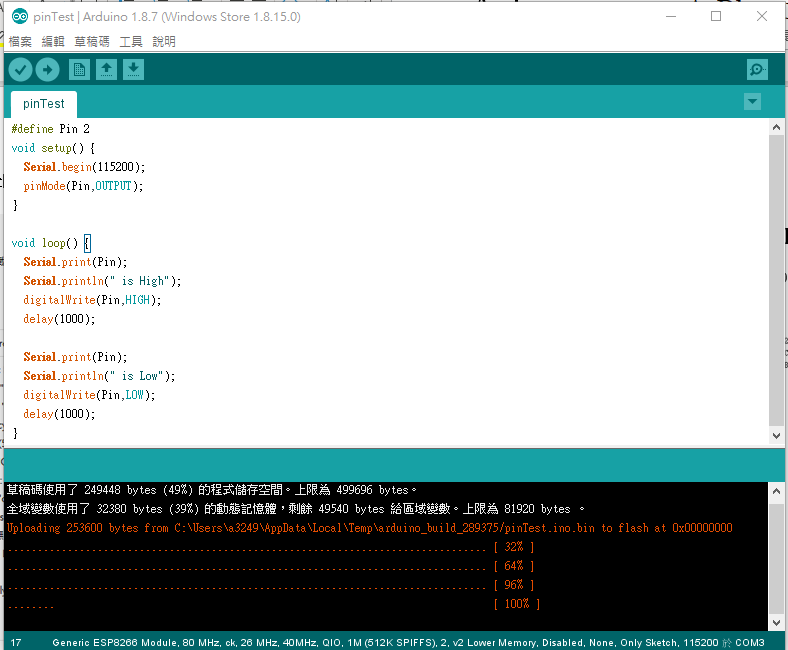
由於此項計畫將會使用File System，因此設定如下圖：



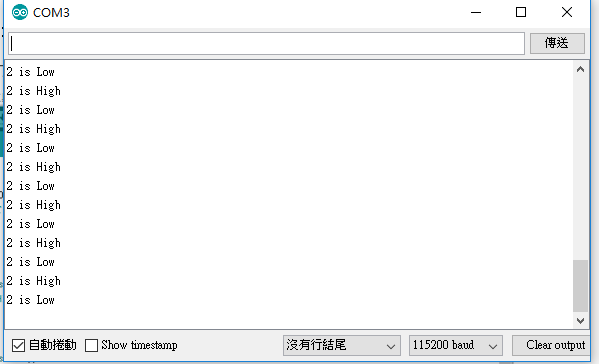
1. 燒入程式

將以下程式進行燒入：

|  |
| --- |
| #define Pin 2  void setup() {  Serial.begin(115200);  pinMode(Pin,OUTPUT);  }  void loop() {  Serial.print(Pin);  Serial.println(" is High");  digitalWrite(Pin,HIGH);  delay(1000);    Serial.print(Pin);  Serial.println(" is Low");  digitalWrite(Pin,LOW);  delay(1000);  } |

燒入結果如下：

Serial的輸出如下：



## ESP8266 Web Server測試

Web Server主要用於在使用者初次設定資料時，透過網頁介面設定WiFi

ESP8266的WiFi在網路上有許多Lib可以使用，在本次計畫則使用ESP8266WiFi.h，作者為liebman及devyte，可以透過GitHub上進行下載.zip檔（<https://github.com/esp8266/Arduino>），下載完成後，請透過Arduino IDE->草稿碼->匯入程式庫->加入.zip程式庫，匯入此Lib。

點選Arduino IDE->檔案->範例->ESPWebServer->HelloServer，將以下程式進行修改：

1. 第六行：

const char\* ssid = "........";

"........" 更改為您的WIFI ID。

1. 第七行：

const char\* password = "........";

"........"更該為您的WIFI密碼。

1. 第九行：

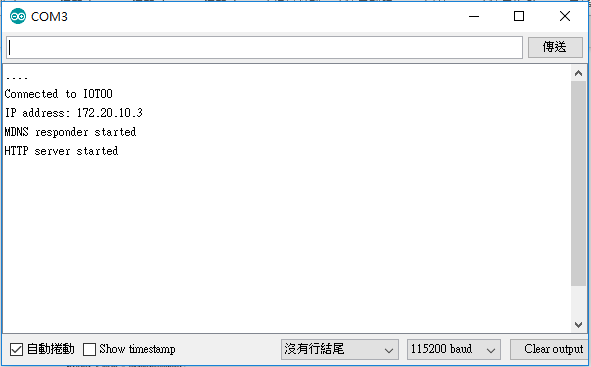
const int led = 13;

更改為 const int led = 2;

並將IO02接上Led燈。

接著寫入程式。

透過Serial中獲得網址，如下圖：



透過手機連至與ESP8266相同的WIFI，並在Chrome中輸入172.20.10.3:80，獲得以下結果：



## ESP8266 Web Socket Client 測試

Web Socket Client 主要用於讓使用者能透過Socket的方式，在外遠端此系統。

在此測試，將使用iPhone當熱點，及使用APP－Socket Tester進行測試，點選Arduino IDE->檔案->範例->ESPWiFi->WiFiClient，將以下程式進行修改：

1. 第十一行：

const char\* ssid = " your-ssid";

" your-ssid" 更改為您的WIFI ID。

1. 第十二行：

const char\* password = " your-password";

" your-password"更該為您的WIFI密碼。

1. 第十四行：

const char\* host = "data.sparkfun.com";

"data.sparkfun.com"更改為您的手機IP。

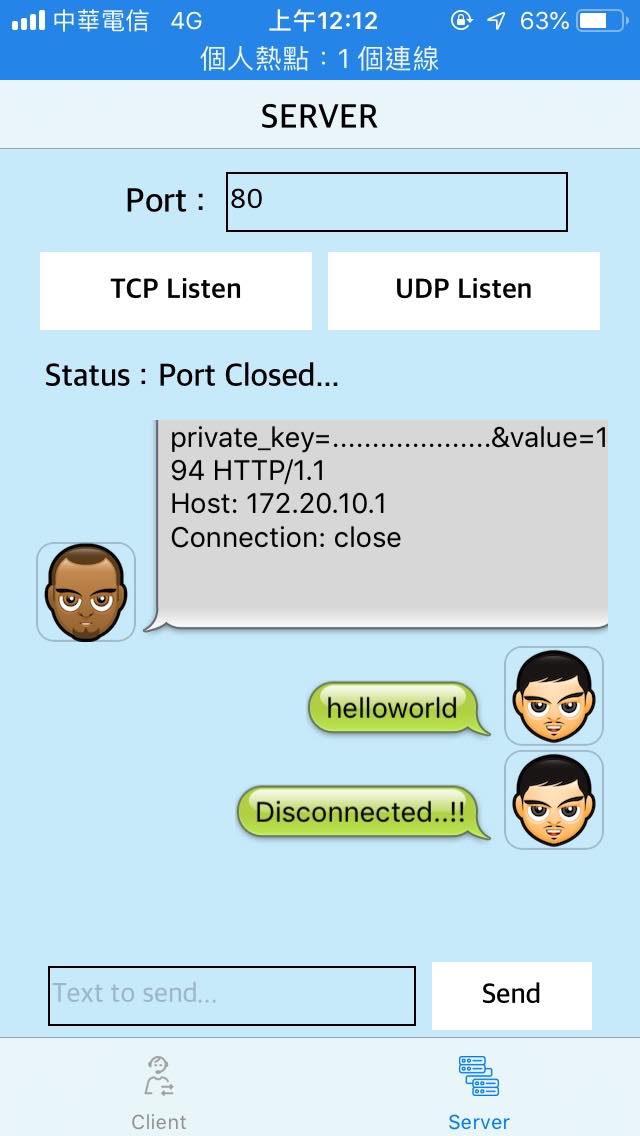
1. 第八十九行：

String line = client.readStringUntil('\r');

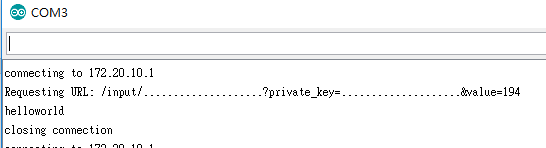
'\r'更改為' '

接著寫入程式。

在Socket Tester中開啟Server，Port設定為80，確認ESP8266連線後，Send出”helloworld “的字串，如下圖：



在Serial中將會獲得訊息如下圖：



## ESP8266 IR接收測試

透過IR接收來記錄使用者的遙控器訊號，並透過發射來傳送資料。

ESP8266的IR接收/發射，並無法使用Arduino IRremote.h此Lib，主因在於Arduino與Esp的暫存器不同，因此在本次計畫則使用IRremoteESP8266.h，作者為crankyoldgit，可以透過GitHub上進行下載.zip檔（<https://github.com/markszabo/IRremoteESP8266>），下載完成後，請透過Arduino IDE->草稿碼->匯入程式庫->加入.zip程式庫，匯入此Lib。

點選Arduino IDE->檔案->範例-> IRremoteESP8266->IRrecvDumpV2，將以下程式進行修改：

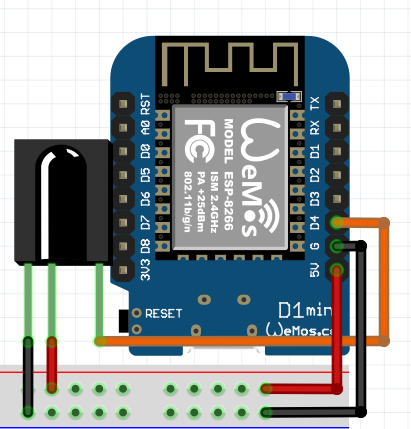
* 1. 第四十四行：

const uint16\_t kRecvPin = 14;

更改為 const uint16\_t kRecvPin = 2;

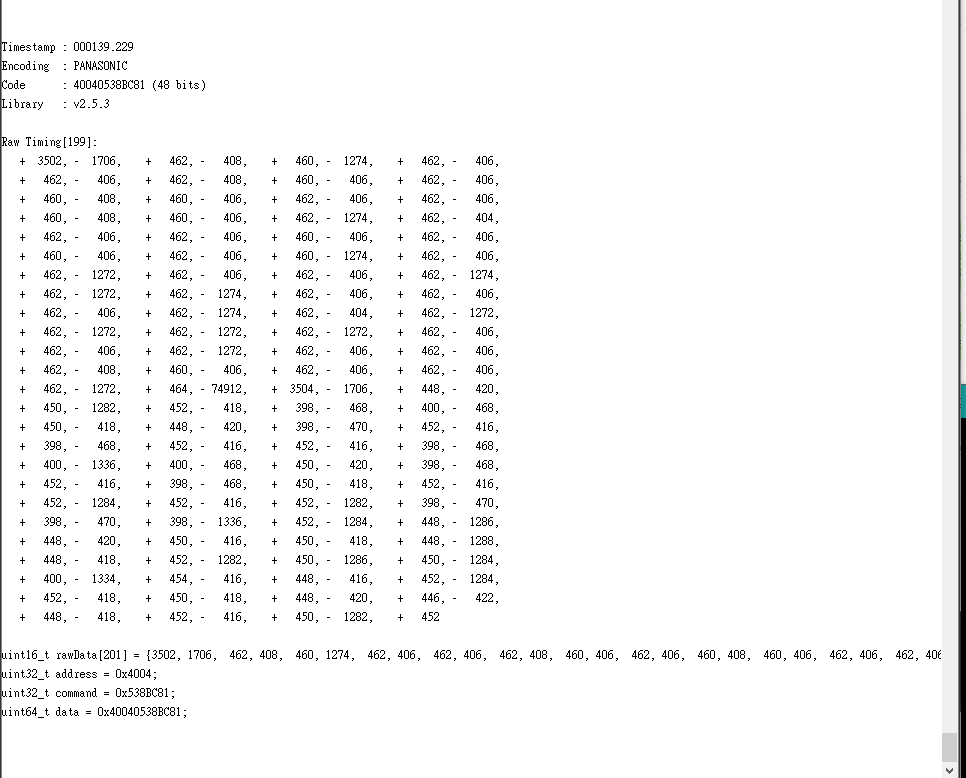
接著寫入程式。

硬體部分的接線圖如下：



請注意IR接收器可能因型號腳位會有所不同。

使用手邊的遙控器對IR接收器進行測試，本次測試使用的是Pansonic藍芽音響的紅外線遙控器開關鍵，測試結果如下：

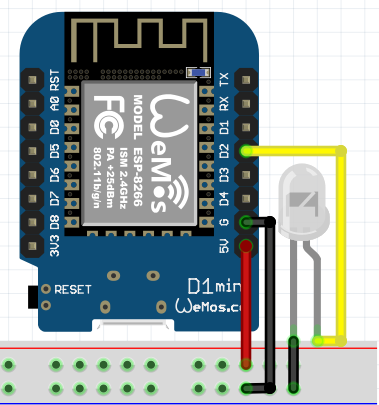


未來可以透過IR發射器，發射出一樣的rawData造成一樣的訊號。

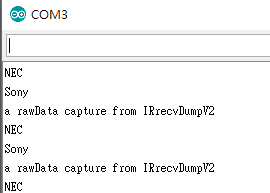
## ESP8266 IR發射測試

點選Arduino IDE->檔案->範例-> IRremoteESP8266->IRGCSendDemo，將程式直接寫入。

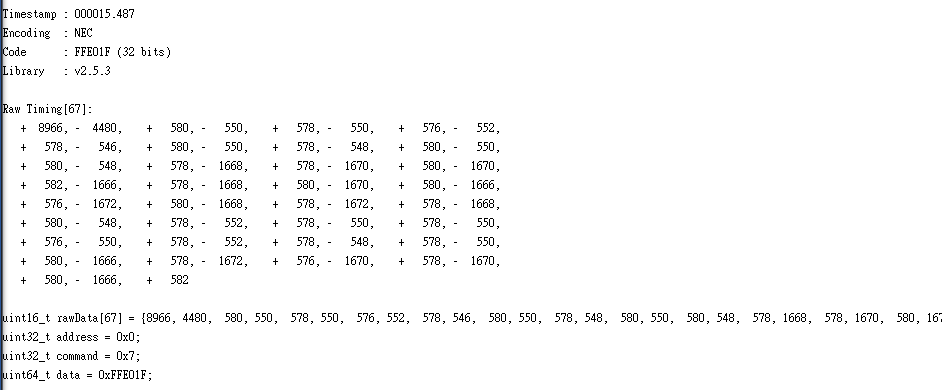
硬體部分接線圖如下：



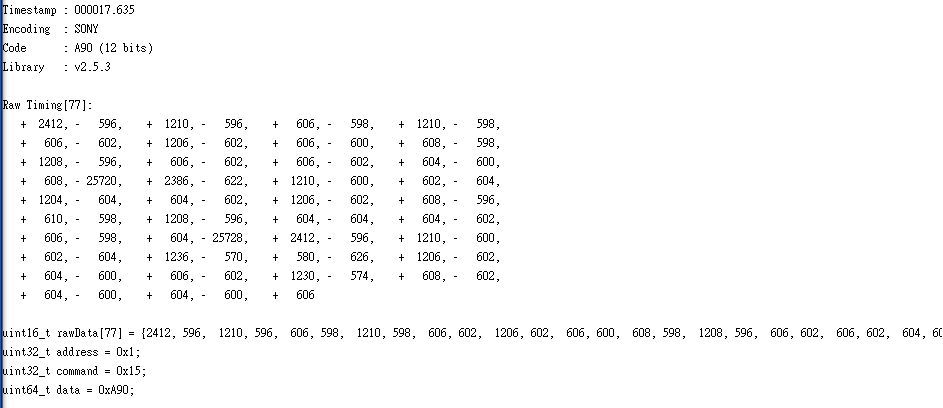
發射出去的資料LOG如下。



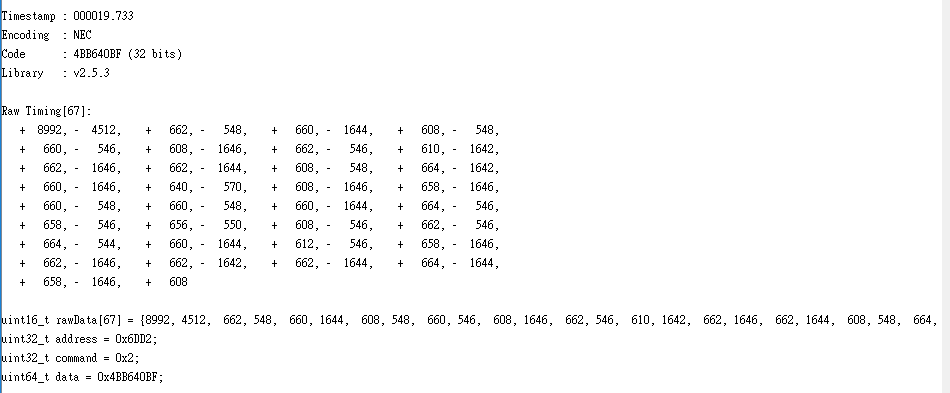
使用之前的接收程式接收此IR訊號結果分別如下：



NEC的訊號



Sony的訊號

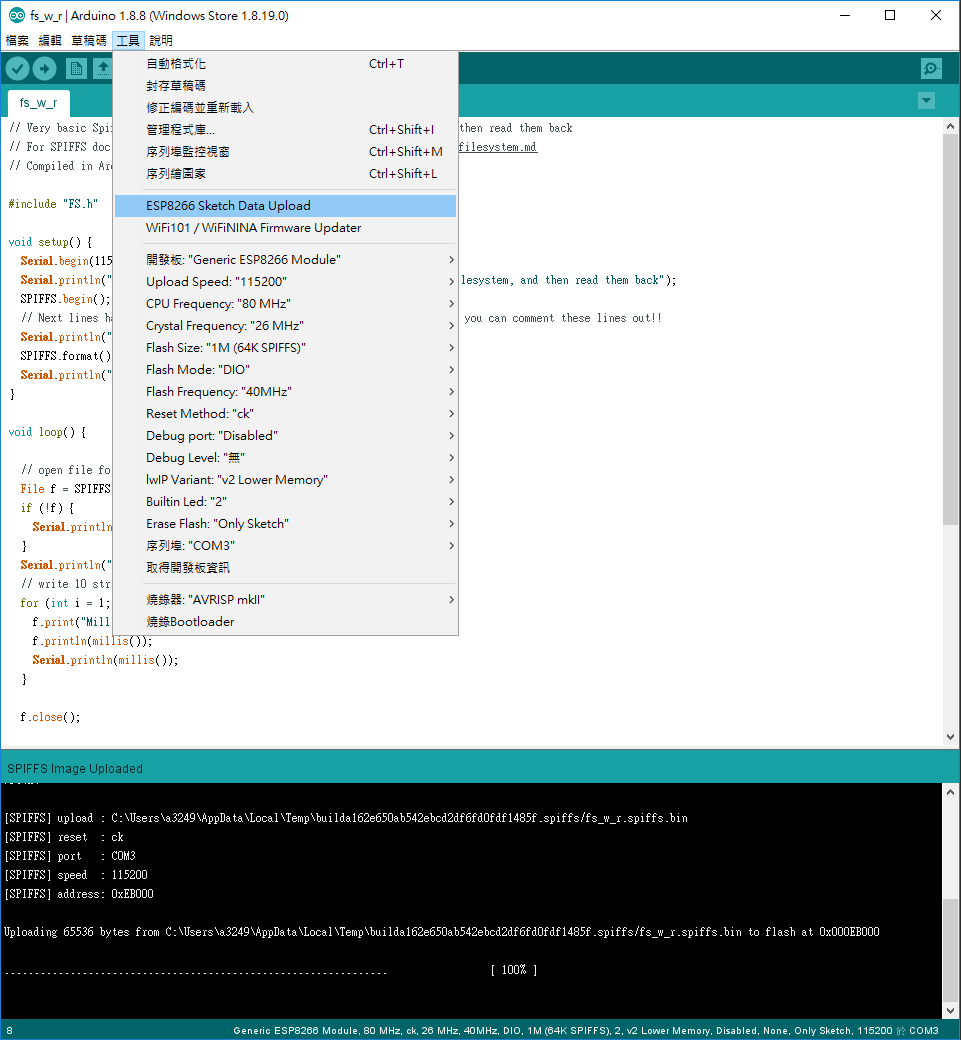


直接傳遞的資料

## File System

主要用於使用者初次設定WiFi 後，不希望每次重啟系統時，需再次設定網路，因此需要File System，主要設定如下。

在此系統的初始設定上有些複雜，特別要注意的是Arduino IDE不要透過Windows Store下載，否則在很多軟體的改動上會遇到很多問題，像新增tools使用了最高管理員卻權限不足；請透過以下網址下載軟體，並解壓縮至Arduino/tools中<https://github.com/esp8266/arduino-esp8266fs-plugin/releases/download/0.1.3/ESP8266FS-0.1.3.zip>，並重啟Arduino IDE；之後在每個新的Arduino 專案中只需新增Data的資料夾，並把需要的文件放置Data中，從IDE中透過工具->ESP8266 Sketch Data Upload上傳至單晶片中，並透過IDE下方Terminal觀察結果，如下圖。



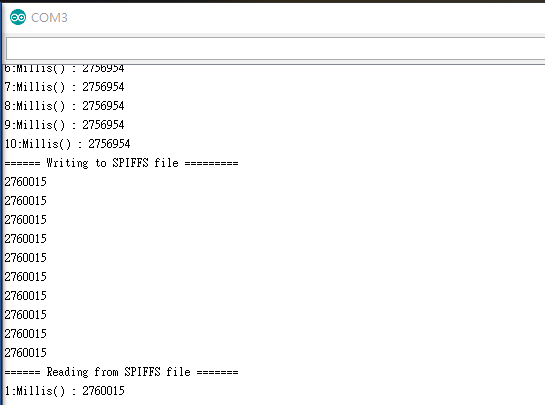
有幾個常見問題，補充在附錄1中。

接下來可以進行第一項網頁、系統測試，本次參考的資料（<https://swf.com.tw/?p=905>），透過此內容，可以成功建置Web Server於ESP中。

下一項測試為File System的讀取、寫入測試，透過以下程式進行測試：

|  |
| --- |
| #include "FS.h"  void setup() {  Serial.begin(115200);  SPIFFS.begin();  Serial.println("Please wait 30 secs for SPIFFS to be formatted");  SPIFFS.format();  Serial.println("Spiffs formatted");  }  void loop() {  File f = SPIFFS.open("/f.txt", "w");  if (!f) {  Serial.println("file open failed");  }  Serial.println("====== Writing to SPIFFS file =========");  for (int i = 1; i <= 10; i++) {  f.print("Millis() : ");  f.println(millis());  Serial.println(millis());  }  f.close();  f = SPIFFS.open("/f.txt", "r");  if (!f) {  Serial.println("file open failed");  }  Serial.println("====== Reading from SPIFFS file =======");  for (int i = 1; i <= 10; i++) {  String s = f.readStringUntil('\n');  Serial.print(i);  Serial.print(":");  Serial.println(s);  }  delay(3000);  } |

寫入成功後，透過Serial觀察結果，若成功會獲得如下圖的資料。



# 實作

## 初始化設定伺服器

下圖為遠端控制遙控器的初始化流程圖。



下圖為初始化設定伺服器的UML圖。



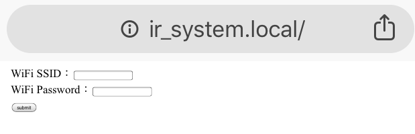
透過\_data這個結構來控制需要設定的資料內容，目前Data的UML圖如下。



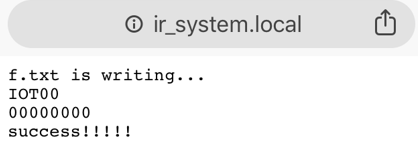
使用者在第一次開機或是重設定資料時，會在開啟熱點及伺服器，使用者透過Web介面進行初始化設定，如下圖所示。



WiFi中搜尋熱點

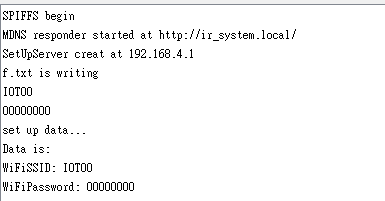


透過Web介面進行初始化設定



設定成功的畫面

使用者也能透過Serial端口獲得相關資訊，如下圖。



當使用者久按Reset按鈕時，會進行覆寫，並重啟SetUpDataSystem，如下圖。

