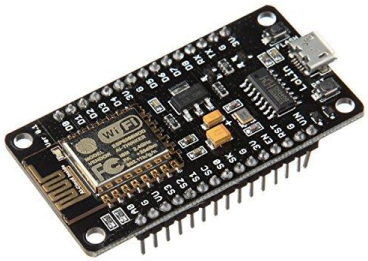
擷取中央氣象署氣象觀測資料的
ESP8266 NodeMCU V3 溫濕度計

雖然現在在手機上可以隨時查詢到氣象資料，但畢竟是一個地區的觀測結果，不表示自身所處環境的溫溼度狀況，例如冬天時氣象觀測為 21℃，但在室內實際測得為 23℃，而夏天時氣象觀測為 33℃，在室內實際上開冷氣候只有 27℃，而一般手機是不提供所處環境溫溼度的。

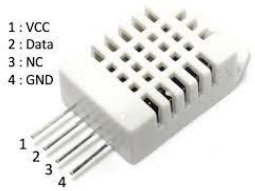
翻了一下工具箱，我有一個兩位數的 TM1637 驅動的七段顯示器模組，也有 0.96 吋的 OLED 顯示器，DHT22 溫溼度感測器也有了，還有一個 NodeMCU V3(ESP8266)，既然差不多的零件都有了，就來動手做一個能顯示所處環境溫溼度又能顯示中央氣象署資料的溫濕度計吧!



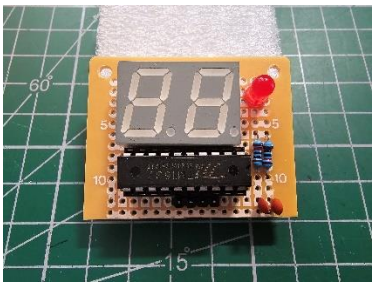
NodeMCU V3 (ESP8266)

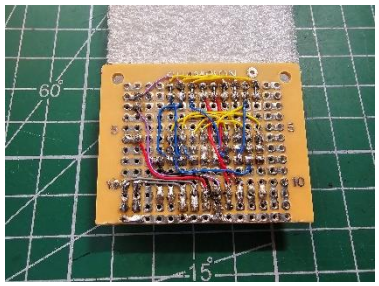
0.96" OLED

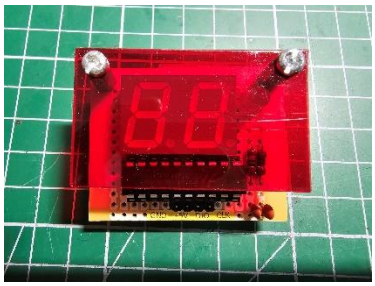
RGB LED



DHT22







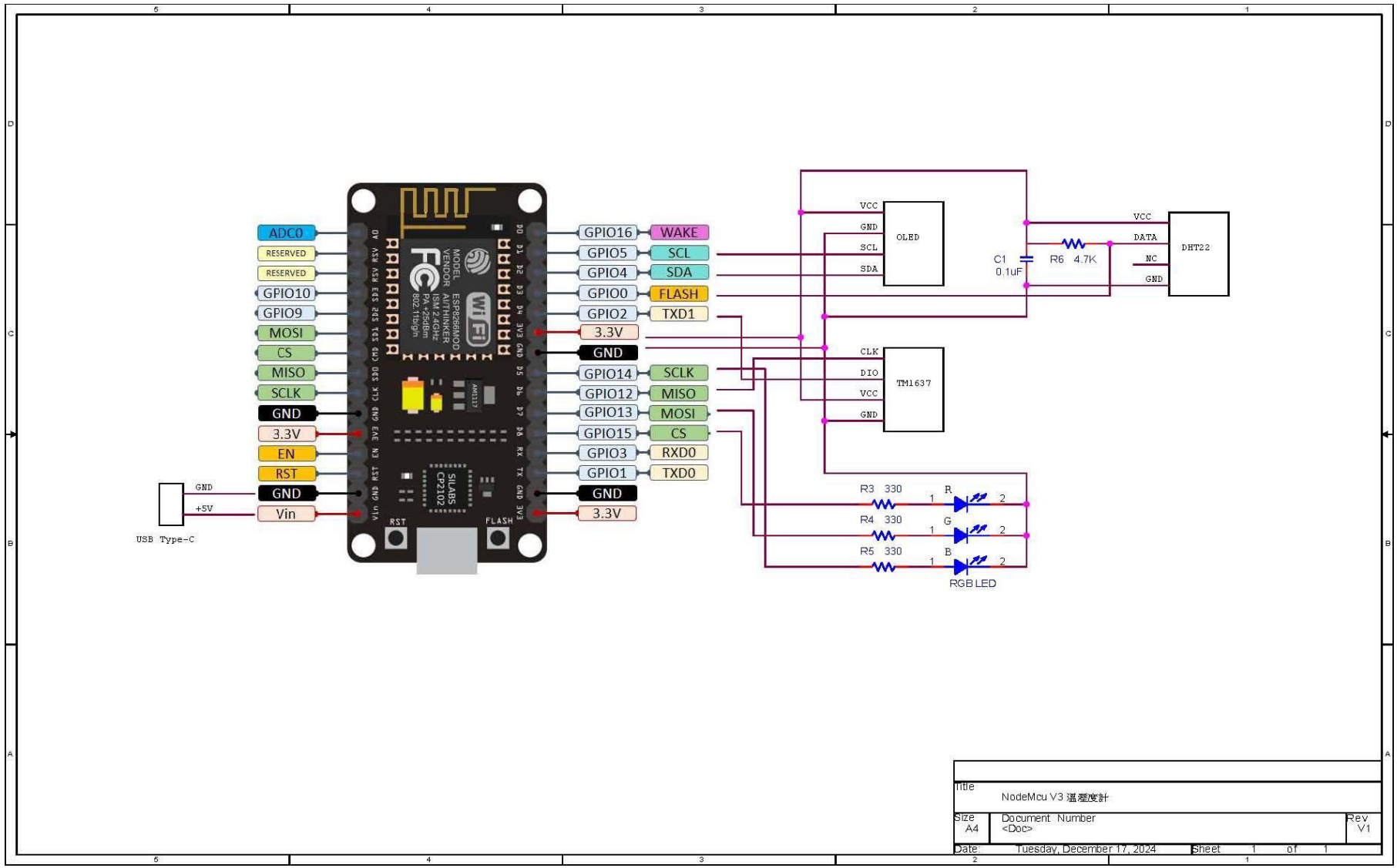
自製 TM1637 驅動的七段顯示器模組

一、線路圖：

由於元件並不多，所以線路也簡單。唯一要注意的是 ESP8266 有許多接腳有一些限制，不能隨意上拉或下拉，不然會無法正常燒錄或啟動。

所有元件都可以在 3.3V 下運作，但七段顯示器和 TM1637 在 5V 下會比較亮，而且只打算由 USB 供電，所以將 TM1637 模組接在 NodeMCU 的 5V 輸出，但 TM1637 的 CLK 和 DIO 兩個輸入因為是接在 ESP8266，所以這兩腳的上拉電阻要接在 3.3V，以免對 ESP8266 造成傷害。

● 線路圖及使用到的 GPIO 如下：



Title	NodeMcu V3 温湿度計	
Size A4	Document Number <Doc>	Rev V1
Date: Tuesday, December 17, 2024	Sheet 1 of 1	

NodeMCU V3 接腳使用注意事項：

標籤	通用輸入輸出	輸入值	輸出量	筆記
D0	GPIO16	沒有中斷	不支持PWM或I2C	開機時高 曾經從深度睡眠中醒來
D1	GPIO5	好	好	通常用作 SCL（I2C）
D2	GPIO4	好	好	通常用作 SDA（I2C）
D3	GPIO0	拉上來	好	連接到FLASH按鈕，如果拉至低電平則啟動失敗
D4	GPIO2	拉上來	好	開機時高 連接到板載LED，如果拉至低電平則啟動失敗
D5	GPIO14	好	好	SPI（SCLK）
D6	GPIO12	好	好	SPI（MISO）
D7	GPIO13	好	好	SPI（MOSI）
D8	GPIO15	拉至GND	好	SPI（CS） 如果拉高則引導失敗
RX	GPIO3	好	RX腳	開機時高
TX	GPIO1	TX引腳	好	開機時高 引導時調試輸出，如果拉至低電平則引導失敗
A0	ADC0	模擬輸入	X	

如果某些引腳被拉低或拉高，可以防止ESP8266引導。下表顯示了BOOT上以下引腳的狀態：

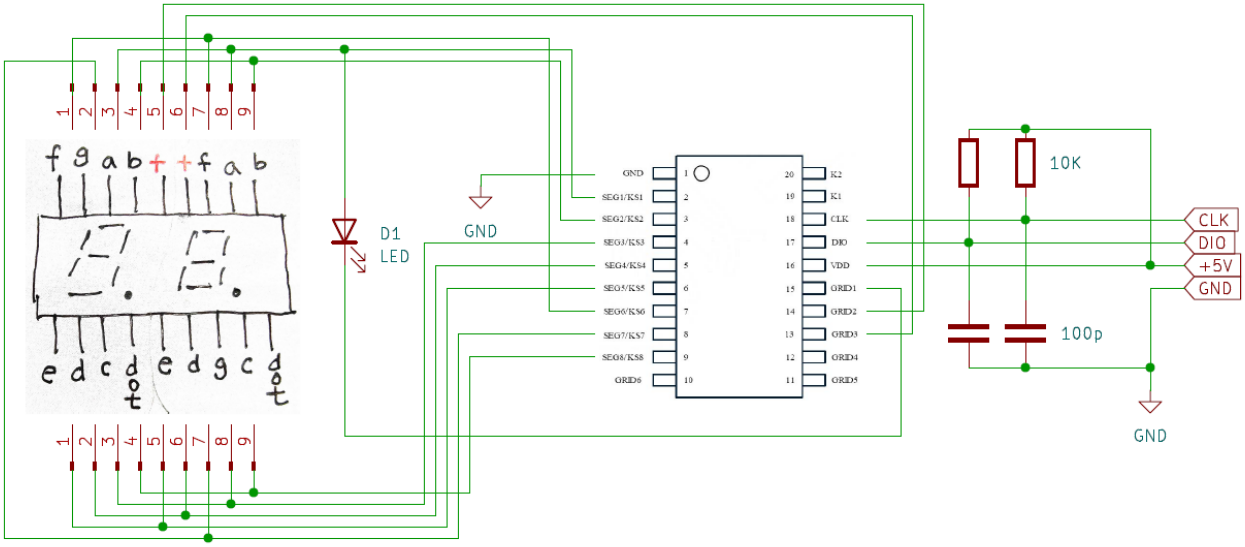
- **GPIO16**：引腳在BOOT處為高電平
- **GPIO0**：如果拉低則啟動失敗
- **GPIO2**：引腳在啟動時為高電平，如果拉低則啟動失敗
- **GPIO15**：如果拉高則啟動失敗
- **GPIO3**：引腳在啟動時為高電平
- **GPIO1**：引腳在啟動時為高電平，如果拉低則啟動失敗
- **GPIO10**：引腳在BOOT處為高電平
- **GPIO9**：引腳在BOOT處為高電平

資料來源：https://honeststore.com.tw/esp8266-pin-out/

● TM1637 模組線路圖：

TM1637 可以控制 6 位數的七段顯示器及 16 位的掃描按鍵輸入，是很方便小巧的顯示控制 IC，七段顯示器必須是共陽極的，CLK 及 DIO 輸入的上拉電阻不可以省略，否則會顯示不正常或不顯示，此圖將上拉電阻接在+5V，實際上已改成接在 3.3V。

雖然市面上有現成的 TM1637 模組可選購，但自己組裝可以深入了解此 IC 動作的原理，有助之後的學習發展。

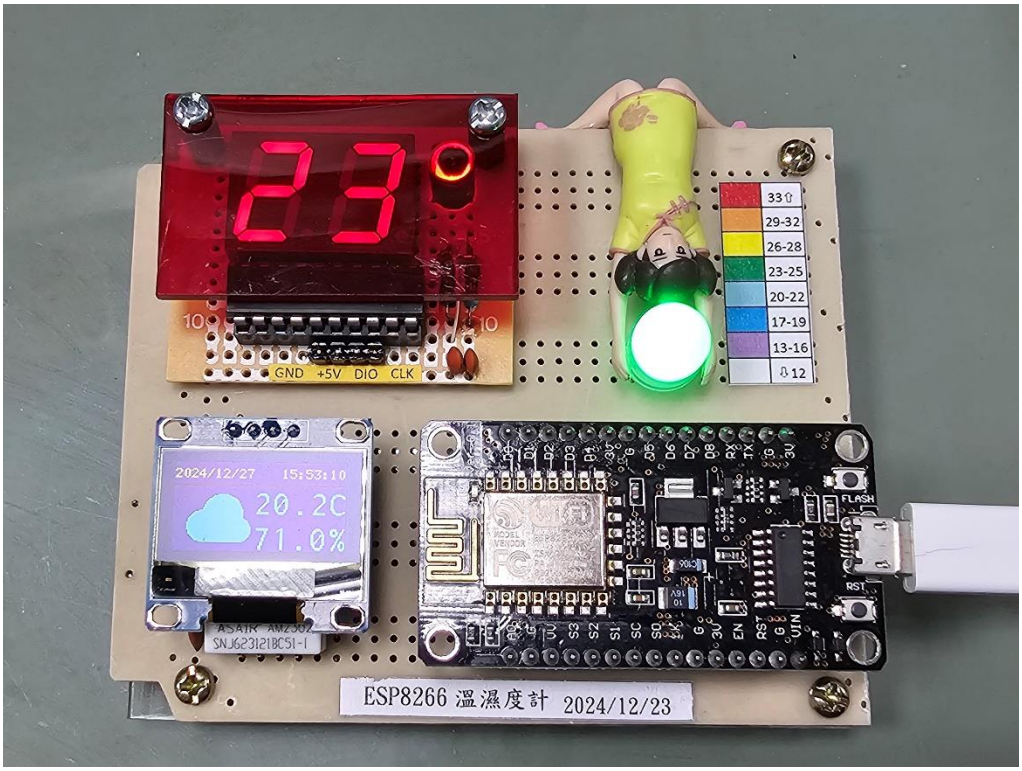


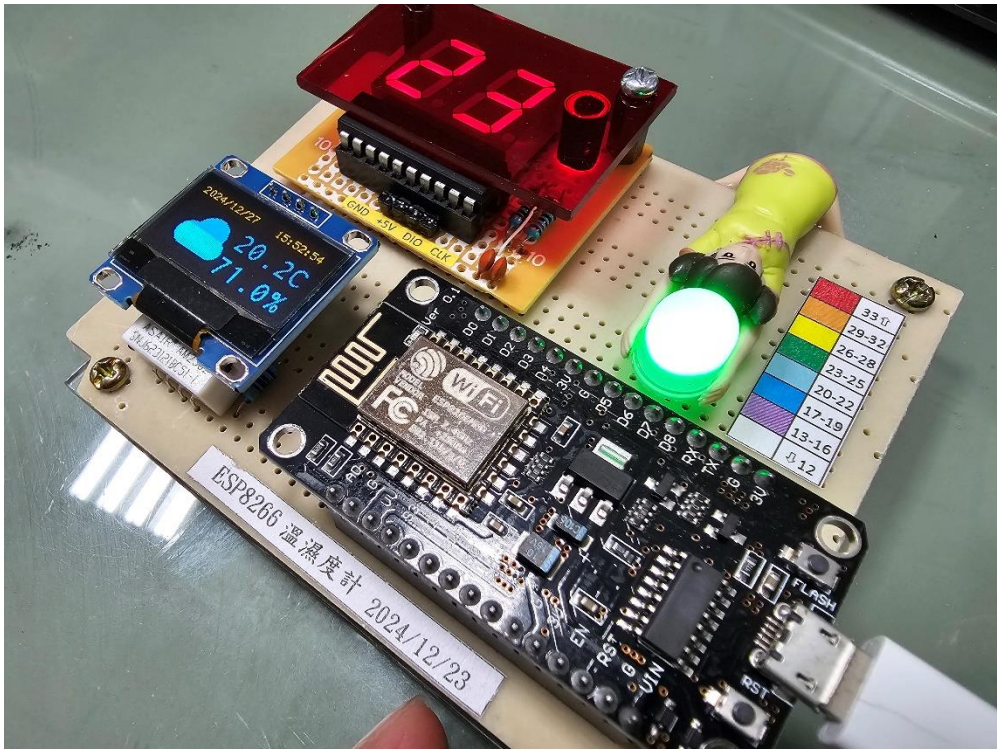
使用Arduino TM1637函式庫，要指定顯示位數，GRID1在最左邊，GRID6在最右邊。
這裡最高位給了LED，十位數為GRID2，個位數為GRID3。

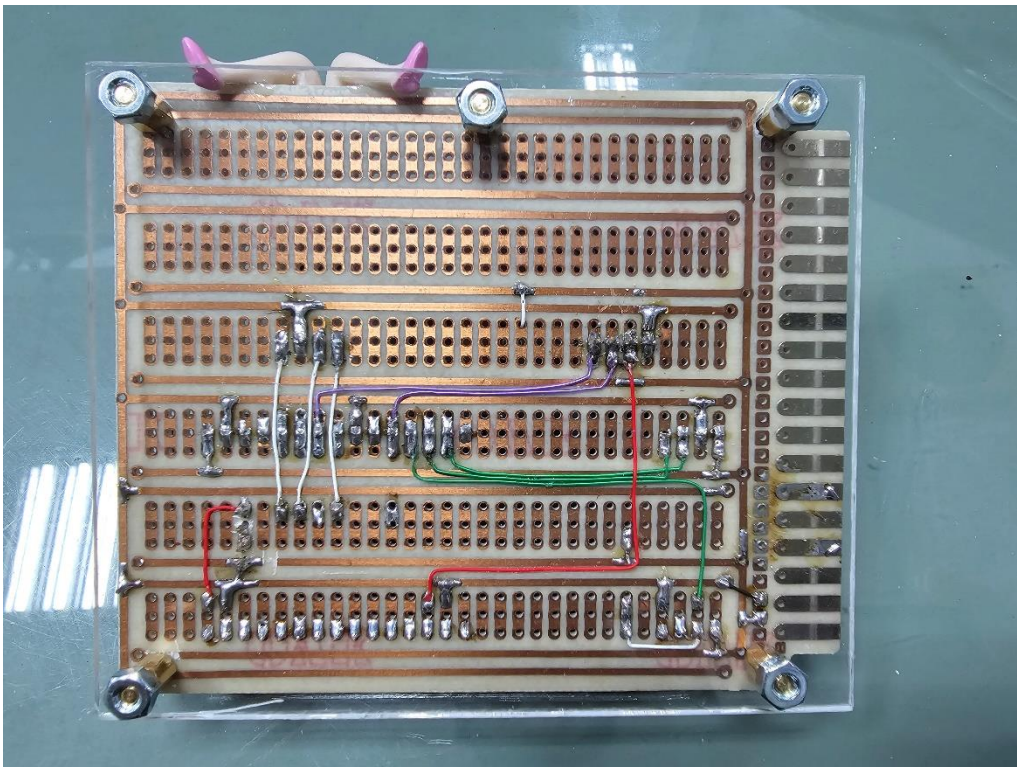
DIO為串列輸入，故最先進入的數字會被推到最前面(GRID1)。
TM1637最多為6位數。

二、實作

裁了一小塊洞洞板，零部件配置完還空了一個區域，就加個杯緣子美化一下吧。接線沒甚麼困難，使用 OK 線焊一下就好。







三、中央氣象署資料：

	33⇧
	29-32
	26-28
	23-25
	20-22
	17-19
	13-16
	⇩12

我希望用七段顯示器顯示目前環境溫濕度，在 OLED 顯示氣象署的觀測資料，這樣有個比較，然後依照我自己的感覺將溫度區分成若干段，以 RGB LED 顯示相對區段的顏色。

由於需取得中央氣象署的資料，關於中央氣象署 OPEN DATA 的取用方式在另一篇單獨說明，請參閱 https://www.hlbh.hlc.edu.tw/resource/openfid.php?id=38959

比較麻煩的是中央氣象署對天氣狀況描述似乎沒有什麼統一標準，例如在回傳的 json 檔中是「"Weather":"陰"」，在氣象署的「預報 XML 產品預報因子欄位中文說明表」中的「預報產品天氣描述代碼表」裡並沒有"陰"，只有"陰天"，多個"天"字，所以沒辦法照著 json 檔的資料去對應到相對的描述和圖示。還好我也只是使用 Arduino 的 U8g2 庫的 u8g2_font_open_iconic_weather _6x_t 圖示檔，它只有幾個簡單的圖示如下，

(看起來是"陰"、"多雲"、"晴(夜間)"、"雨"、(星星是甚麼意思?)、"晴")

所以我自己歸納一下取得的 json 檔可能出現的描述，去顯示相對應的圖就好。

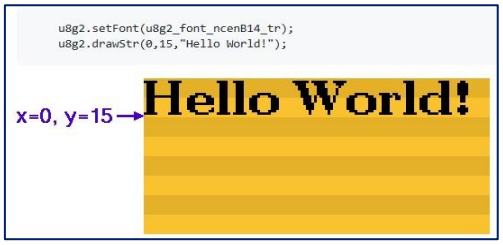
四、結語及注意事項：

一開始以為所有 GPIO 都是一樣的，就隨便配置了，而部分原件是需要上拉的，而 RGB LED 因為是共陰，也相當於下拉了，結果又是開不了機又是無法上傳一堆問題，還是查了 ESP8266 GPIO 相關資訊才解決。

中央氣象署的 json 檔資料我只擷取了「觀測」部分的「花蓮市」的資料，所以回傳資料還蠻少的，很容易理解。

DHT22 藏在 OLED 下面，還好 OLED 運作時並不發熱，應該對溫度感測影響不大啦。

OLED 的 X、Y 座標一開始有誤解，以為定位是指左上角，正確應是左下角，如圖：



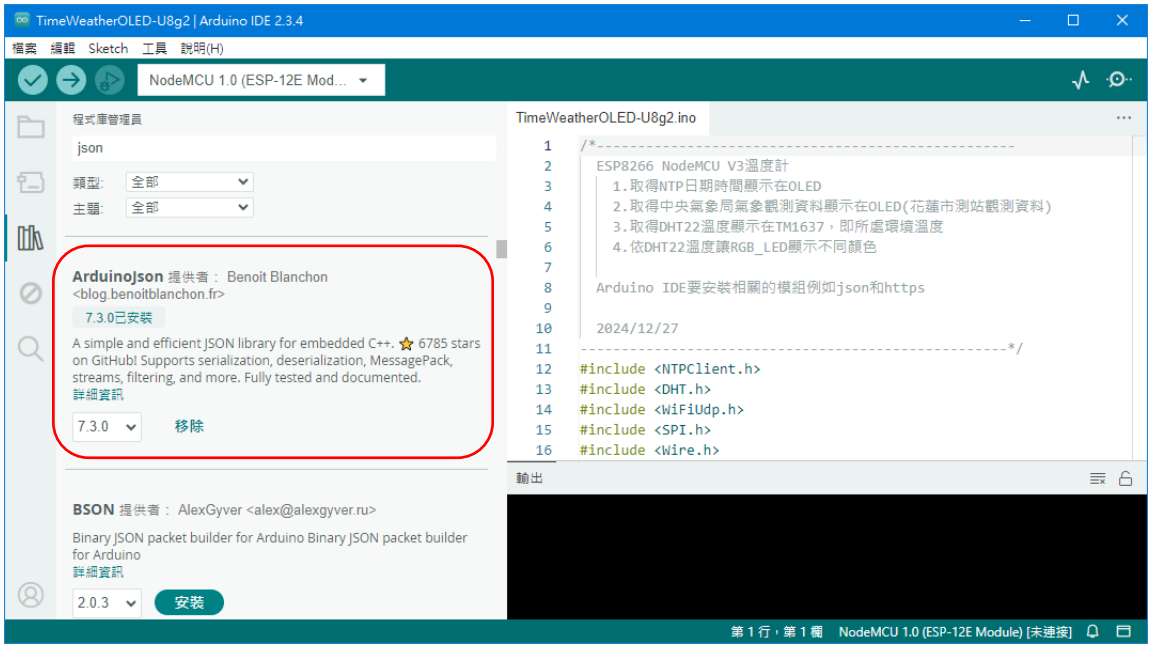
中央氣象署的溫度到小數第一位，濕度只有整數，為了 OLED 畫面整齊，我依然將濕度轉為浮點數顯示，只是小數位永遠是 0。

U8g2 庫有許多字型檔可選，顯示指令也方便，不錯用，也可以使用 SSD1306 庫來顯示，記得 Arduino 要安裝相應的模組。

ESP8266 沒有內建 RTC 模組，為了顯示時間，每秒都要要求 NTP 資料，如果使用 ESP32 可以減少跟 NTP 的通訊次數。

五、相關模組：

- <WiFiClientSecureBearSSL.h>在 ESP8266WiFi 模組裡。
- Arduinojson 模組：



參考程式：

```
/*-----------------------------------------------------
  ESP8266 NodeMCU V3 溫度計
    1.取得 NTP 日期時間顯示在 OLED
    2.取得中央氣象局氣象觀測資料顯示在 OLED(花蓮市測站觀測資料)
    3.取得 DHT22 溫度顯示在 TM1637(即所處環境溫度)
    4.依 DHT22 溫度讓 RGB_LED 顯示不同顏色

  Arduino IDE 要安裝相關的模組例如 json 和 https

  2024/12/27
-----------------------------------------------------*/
#include <NTPClient.h>
#include <DHT.h>
#include <WiFiUdp.h>
#include <SPI.h>
#include <Wire.h>
#include <TM1637Display.h>
#include <time.h>
#include <U8g2lib.h>
#include <String.h>
#include <ArduinoJson.h>    //解析 json 的模組
#include <ESP8266WiFi.h>
#include <ESP8266WiFiMulti.h>
#include <ESP8266HTTPClient.h>
#include <WiFiClientSecureBearSSL.h>  //中央氣象局 opendata 是 https 連接
#include <WiFiClient.h>

//RGB_LED pin
#define R 15
#define G 13
#define B 14

//設定 OLED
U8G2_SSD1306_128X64_NONAME_F_HW_I2C u8g2(U8G2_R0, U8X8_PIN_NONE);
String msg;  // 要顯示在螢幕的訊息字串

// 設定 DHT22
#define DHT_SENSOR_PIN  0
#define DHT_SENSOR_TYPE DHT22
DHT dht_sensor(DHT_SENSOR_PIN, DHT_SENSOR_TYPE);
float URLtemp, URLhumi, humi, tempC;
int icon;

// 設定 TM1637
#define CLK 12
#define DIO 2
TM1637Display displayTM(CLK, DIO);  //設定傳輸腳位
int initflag = 1;
```

```
const char* ssid1     = "Your WiFi AP name 1";
const char* password1 = "password here 1";
const char* ssid2     = "Your WiFi AP name 2";
const char* password2 = "password here 2";
const char* ssid3     = "Your WiFi AP name 3";
const char* password3 = "password here 3";

//事先取得中央氣象局 API Key，在其網站可取得連接字串如下
String API_URL = "https://opendata.cwa.gov.tw/api/v1/rest/datastore/O-A0003-
001?Authorization=CWA-00000000-0000-0000-0000-000000000000&StationId=466990";
HTTPClient https;
WiFiClient client;
ESP8266WiFiMulti WiFiMulti;

// Define NTP Client to get time
const long utcOffsetInSeconds = 28800;
WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP, "pool.ntp.org", utcOffsetInSeconds);
// Extract date and time components
int currentYear;
byte month = 0;
byte day = 0;
byte hour = 0;
byte minute = 0;
byte second = 0;
String MM, DD, hh, mm, ss;   //供 OLED 顯示日期時間時個位數補"0"用

//顯示 RGB_LED 顏色
void showRGB(int r, int g, int b){
  analogWrite(R, r);
  analogWrite(G, g);
  analogWrite(B, b);
}

//取得中央氣象局資料，注意是 https
String openWeather(){
  String payload = "";

  std::unique_ptr<BearSSL::WiFiClientSecure> client(new
BearSSL::WiFiClientSecure);
  client->setInsecure();

  if(https.begin(*client, API_URL)){
    int httpCode = https.GET();
    if (httpCode > 0) {
      payload = https.getString();   //取得回應本體
      //Serial.printf("回應本體：%s\n", payload.c_str());
    }
```

```
    else {
      Serial.println("HTTP 請求出錯了～");
    }
    https.end();
    return payload;
  }
  return payload;
}

//自 HTTP 回應內容取出溫濕度及氣象
void parseWeather(String json)
{
  //轉成 json 物件，本例中央氣象局回應本體約 2137 字，此處 buffer 留 2200byte 應該夠了
  DynamicJsonDocument doc(2200);

  deserializeJson(doc, json);
  JsonObject weather = doc["records"]["Station"][0]["WeatherElement"]; //資料路徑
  String Weather = (String)weather["Weather"];   //取得天氣狀況
  Serial.printf("天氣：%s\n", Weather);
  URLtemp = (float)weather["AirTemperature"];    //取得觀測溫度
  URLhumi = (float)weather["RelativeHumidity"];  //取得相對溼度
  Serial.printf("攝氏溫度：%.1f\n", URLtemp);
  Serial.printf("濕    度：%.1f\n", URLhumi);

  // 使用 U8g2 庫，取得圖示數值 code:           陰  多雲  晴(夜間)  雨  星星  晴
  //u8g2_font_open_iconic_weather_6x_t -->  64   65    66    67   68   69
  //不知氣象局有多少種天氣狀況描述，反正 U8g2 庫的圖示只有幾個，就大概就好
  if(Weather.indexOf("雨") != -1)    //"雨"最優先，只要有這個字就是雨的圖形了
    icon = 67;
  else if(Weather.indexOf("多雲") != -1){   //"多雲"還有點日頭，先用
    icon = 65;
  }
  else if(Weather.indexOf("陰") != -1){    //剩下有"陰"這個字的就用這個圖
    icon = 64;
  }
  else if(Weather.indexOf("晴") != -1){    //單獨出現"晴"這個字時得看看是日間還夜間
    if(hour >= 18 && hour <= 4)
      icon = 66;     //晴天夜間使用月亮符號
    else
      icon = 69;     //晴天白日使用太陽符號
  }
  else
    icon = 65;      //如果都不是，那就都歸到多雲吧
}

void setup(){
  Serial.begin(115200);          //設置串列傳輸速率
  dht_sensor.begin();            // initialize the DHT sensor
  displayTM.setBrightness(7);    //TM1637 的亮度設定，從 0~7 最強
```

```
  //未取得 DHT22 數據前先顯示 88，全亮順便當 LED 檢測
  displayTM.showNumberDec(88, 1, 3, 0);
  u8g2.begin();

  //連接無線基地台，會嘗試三個常用基地台
  WiFi.mode(WIFI_STA);
  int tryN = 0;
  while(1){
    WiFi.begin(ssid1, password1);
    while (WiFi.status() != WL_CONNECTED && tryN<20) {
      Serial.print(".");
      tryN++;
      delay(500);
    }
    if(tryN<20) break;   //連接成功，離開迴圈，後面不用試了

    tryN= 0;
    WiFi.begin(ssid2, password2);
    while (WiFi.status() != WL_CONNECTED && tryN<20) {
      Serial.print(".");
      tryN++;
      delay(500);
    }
    if(tryN<20) break;   //連接成功，離開迴圈，後面不用試了

    tryN= 0;
    WiFi.begin(ssid3, password3);
    while (WiFi.status() != WL_CONNECTED && tryN<20) {
      Serial.print(".");
      tryN++;
      delay(500);
    }
    if(tryN<20) break;   //連接成功，離開迴圈，不用再試了
  }
  Serial.print("IP 位址：");
  Serial.println(WiFi.localIP());

  //設定 UDP 客戶端，以便能夠向 NTP 伺服器發送和接收數據包。
  //準備 NTP 客戶端的內部計時器，確保可以定期向 NTP 伺服器請求時間更新。
  timeClient.begin();
}

void loop() {
  timeClient.update();
  unsigned long epochTime = timeClient.getEpochTime();
  String DATE, TIME;
  currentYear = 1970;

  // Convert epoch time to year, month, day, etc.
```

```
  unsigned long days = epochTime / 86400;
  unsigned long remainingSeconds = epochTime % 86400;
  hour = remainingSeconds / 3600;
  remainingSeconds = remainingSeconds % 3600;
  minute = remainingSeconds / 60;
  second = remainingSeconds % 60;

  // Calculate current year
  while((days >= 365 && !(currentYear % 4 == 0 && (currentYear % 100 != 0 ||
currentYear % 400 == 0))) || (days >= 366 && (currentYear % 4 == 0 &&
(currentYear % 100 != 0 || currentYear % 400 == 0)))) {
    days -= (currentYear % 4 == 0 && (currentYear % 100 != 0 || currentYear %
400 == 0)) ? 366 : 365;
    currentYear++;
  }

  // Calculate month and day
  const byte daysInMonth[] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
  for (month = 0; month < 12; month++) {
    byte dim = daysInMonth[month];
    if (month == 1 && currentYear % 4 == 0 && (currentYear % 100 != 0 ||
currentYear % 400 == 0))
      dim = 29;
    if (days >= dim) {
      days -= dim;
    } else {
      break;
    }
  }
  day = days + 1;

  //印出日期時間，若為個位數則補上一個 0
  Serial.print("Date: ");
  Serial.print(currentYear);
  Serial.print("-");
  if (month < 9){
    Serial.print("0");
    MM = "0";
  }
  else{
    MM = "";
  }
  Serial.print(month + 1);
  Serial.print("-");
  if (day < 10){
    Serial.print("0");
    DD = "0";
  }
  else{
```

```
  DD = "";
}
Serial.print(day);
Serial.print(" ");
Serial.print("Time: ");
if (hour < 10){
  Serial.print("0");
  hh = "0";
}
else{
  hh = "";
}
Serial.print(hour);
Serial.print(":");
if (minute < 10){
  Serial.print("0");
  mm = "0";
}
else{
  mm = "";
}
Serial.print(minute);
Serial.print(":");
if (second < 10){
  Serial.print("0");
  ss = "0";
}
else{
  ss = "";
}
Serial.print(second);
Serial.println();

//每分鐘在 TM1637 更新溫度一次
if(second == 0 or initflag == 1){  //僅第一次開進入時不管秒數都要執行一次顯示
  initflag = 0;    //之後要 0 秒時才進入讀取 DHT22
  humi  = dht_sensor.readHumidity();     // read humidity
  tempC = dht_sensor.readTemperature();  // read temperature
  if (isnan(tempC) || isnan(humi)) {
    Serial.printf("DHT22 Read ERROR!\n");
    displayTM.showNumberDec(99, 0, 3, 0);   //若 DHT22 讀取錯誤則顯示 99
    //在 OLED 第一行顯示 DHT22 錯誤
    u8g2.clearBuffer();
    u8g2.setFont(u8g2_font_6x12_tr);     //width=6, height=12
    msg = "DHT22 Error!";
    u8g2.drawStr(0, 12, msg.c_str());
  }
  else{
    Serial.printf("%4.1fC\t", tempC);
```

```
      Serial.printf("%4.1f%c\n", humi, '%');
      displayTM.showNumberDec(int(tempC), 1, 3, 0);

      String payload = openWeather();  //取得中央氣象局資料
      if (payload != ""){
        parseWeather(payload);         //處理回傳的 json 檔，更新相關數據
      }
    }
  }
  if(second >=50)   //50~59 秒時顯示 DHT22 濕度
    displayTM.showNumberDec(int(humi), 0, 3, 0);

  //在 OLED 顯示日期時間
  u8g2.clearBuffer();
  u8g2.setFont(u8g2_font_6x12_tr);     //width=6, height=12，較小字型
  msg = (String)currentYear + '/' + MM + (String)(month + 1) + '/' + DD +
(String)day;
  u8g2.drawStr(0, 12, msg.c_str());
  msg = hh + (String)hour + ':' + mm + (String)minute + ':' + ss +
(String)second;
  u8g2.drawStr(80, 12, msg.c_str());
  //在 OLED 顯示氣象資料
  char dd[6];
  u8g2.setFont(u8g2_font_inr16_mf);    //width=14, height=26，較大字型
  sprintf(dd, "%4.1fC", URLtemp);
  u8g2.drawStr(60, 37, dd);
  sprintf(dd, "%4.1f%c", URLhumi, '%');
  u8g2.drawStr(60, 63, dd);
  //在 OLED 顯示天氣圖示
  u8g2.setFont(u8g2_font_open_iconic_weather_6x_t);
  u8g2.drawGlyph(10, 64, icon);    //參數: (x, y, icon_number)
  u8g2.sendBuffer();

  //依溫度顯示 RGB_LED 顏色，我自己訂的溫度與顏色的對應
  if(tempC >= 33)
    showRGB(100, 0, 0);     //紅
  else if(tempC >= 29 && tempC < 33)
    showRGB(100, 15, 0);    //橙
  else if(tempC >= 26 && tempC < 29)
    showRGB(100, 40, 0);    //黃
  else if(tempC >= 23 && tempC < 26)
    showRGB(0, 100, 0);     //綠
  else if(tempC >= 20 && tempC < 23)
    showRGB(10, 20, 100);   //淺藍
  else if(tempC >= 17 && tempC < 20)
    showRGB(0, 0, 100);     //藍
  else if(tempC >= 13 && tempC < 17)
    showRGB(40, 5, 80);     //紫
  else
```

```
    showRGB(70, 70, 70);    //白

  delay(1000);
}
```