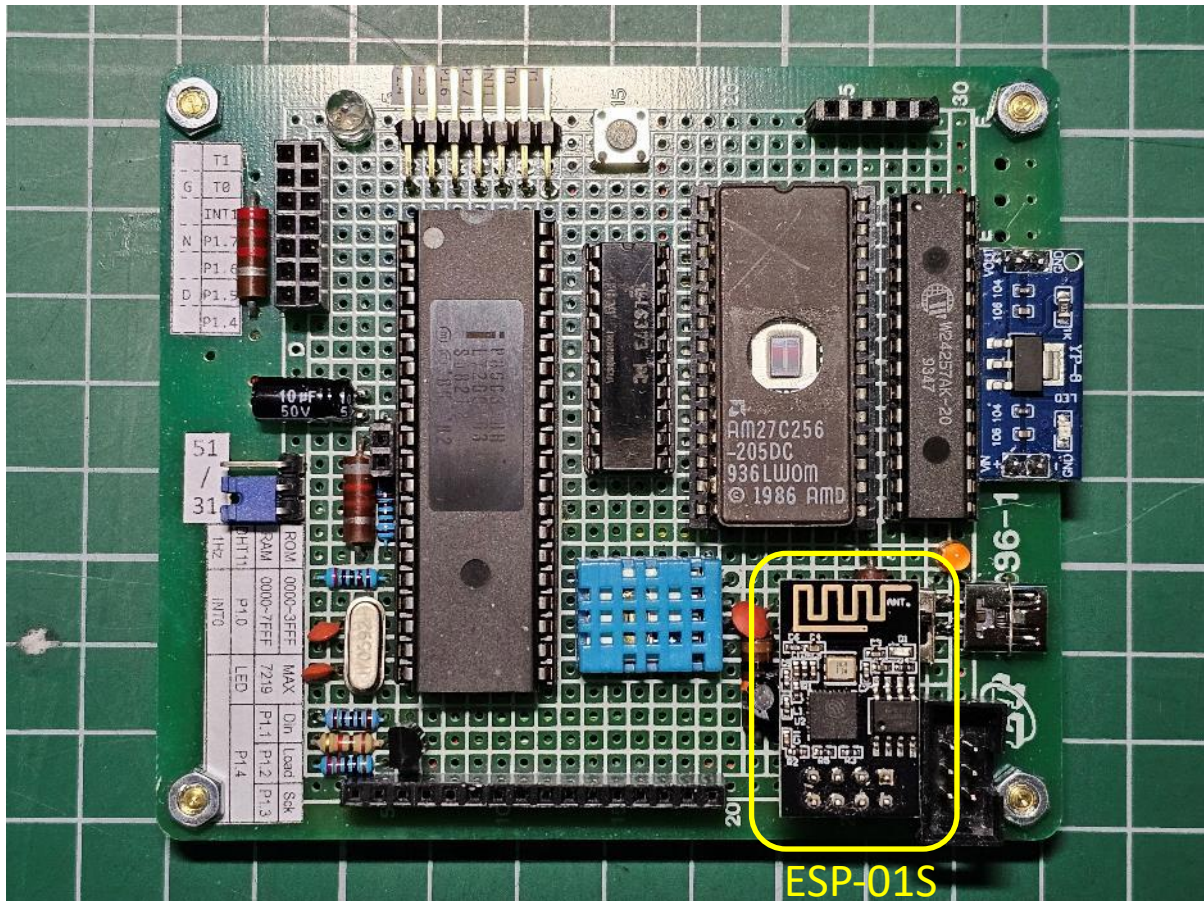


## 8031/51 單板微電腦 + ESP-01S

在 ESP8266、ESP32... 等等這些小板子蓬勃發展的年代，為了 IoT，微控制器能不能連上網就很重要了。MCS-51 和相容的系列單晶片都沒有連網的能力，那我們就來試試將 8031/51 加上一個 ESP-01S 讓它連上網路吧。



### 一、系統規劃

雖然買一個 NodeMCU 或類似的產品就解決了，GPIO(通用 IO 腳)夠用，又支援 WiFi、藍芽等，但我就是無聊啊，一定要讓 8031/51 連上網！

因為目的單純，這次我只需要一個簡單的系統來驗證 8031/51 與 ESP-01S 是否能溝通就好，所以使用一顆 AT89S51/52 加上 ESP-01S 就可以達成，但既然無聊嘛，那就把 ROM 和 SRAM 都順便加上了，讓 8031 也能快樂上網。

理論上兩者都有串列通訊埠，只要訊號正確就可以互相溝通，差別是 8031/51 是使用 +5V，ESP-01S 是使用 +3V，要分別供應電源，介面間電壓差別也要特別留意，不然 3V 的 ESP-01S 可不耐燒。

選擇 ESP-01S 的幾個原因，一是便宜，它連網的功能與一般 NodeMCU 一樣，都是 ESP8266 晶片。二是體積小巧，只拉出一組串列埠及兩個 GPIO 腳，不需要多，因為主要的 IO 腳可以給 8031/51 去做，它只要專心連網就好。很多人覺得 ESP-01S 很陽春不好用，我卻覺得和不能上網的單晶片搭配真是天生一對呀。


### 二、線路圖

(二)、這次沒有+5V 電源電路的部分(只是因為我手邊的洞洞板太小了)，直接由 USB 接頭供應，可以使用電腦 USB 或手機充電頭經 USB 線供應，但加上了 5V 轉 3V 的模組，給 ESP-01S 用。

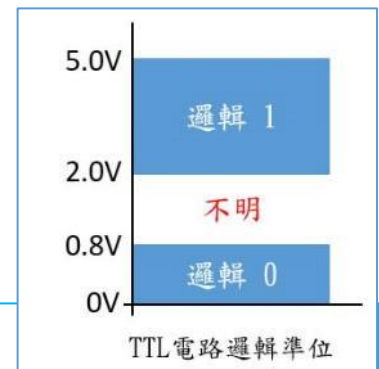
(三)、因為這次洞洞板較小，SRAM 使用了 W24257AK 這顆 IC，它和 62256 系列的接腳、容量都一樣，但是體積寬度少了一半，很節省空間。

(四)、因為只有 CPU、ROM、SRAM，沒有其它 IO 晶片，所以位址解碼電路就不需要了，只需要一顆 74LS373 來將 D0-D7 與 A0-A7 解開即可。

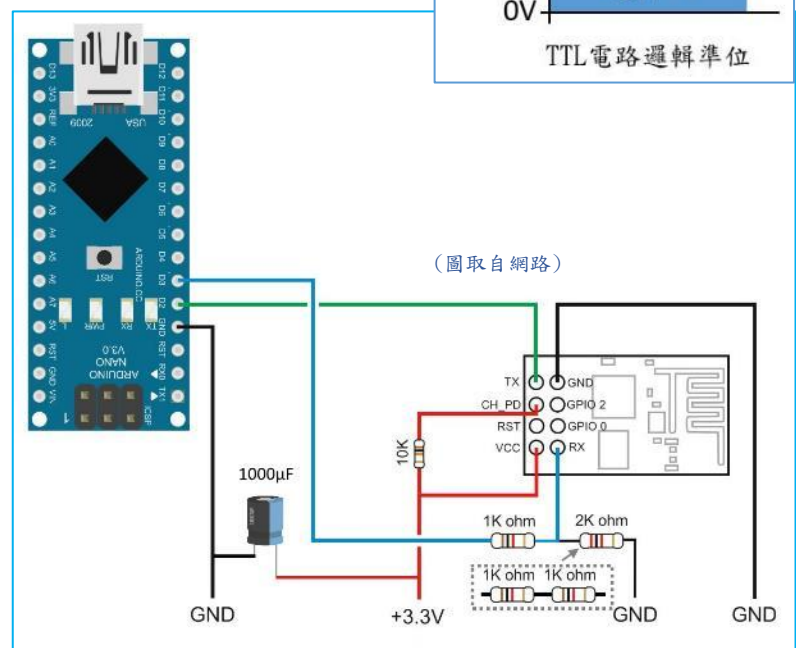
(五)、8031/51 與 ESP-01S 相互透過串列埠溝通，但兩者訊號的電壓不同，對 TTL 位準的 8031/51 來說低於 0.8V 者皆視為邏輯 0，高於 2.0V 至 5.0V 之間皆視為邏輯 1，但若電壓落在 0.8V 至 2.0V 之間，就無法判斷到底是 0 還是 1 了。ESP-01S 的 Tx 送到 8031/51 的 Rx 的訊號介於 0V 至 3V 之間，這讓 8031/51 可以辨識送來的是 0 還是 1，沒有問題，可以直接接，但 8031/51 的 Tx 送到 ESP-01S 的 Rx 就不能直接接了，8031/51 在送出邏輯 0 時還好，但送出邏輯 1 時高達 5V，嬌嫩的 ESP-01S 可受不了，得減壓至 3V 才行。



電壓範圍 (V)	邏輯狀態
0V ~ 0.8V	邏輯 0
0.8V ~ 2.0V	不明
2.0V ~ 5.0V	邏輯 1



一開始想得很單純，照著網路說的(如右圖)用兩個電阻分壓即可得到 3V，實際上這招在 8031/51 是行不通的，它沒有足夠的電流來驅動獲得正確的分壓，理論上 Arduino 也是使用 ATMEL 的 ATmega328p 晶片，和 AT89S51 應該差不多，但電氣特性還是有差，後來還是乖乖加上一個電晶體開關來轉換電平才解決，當然也有人說直

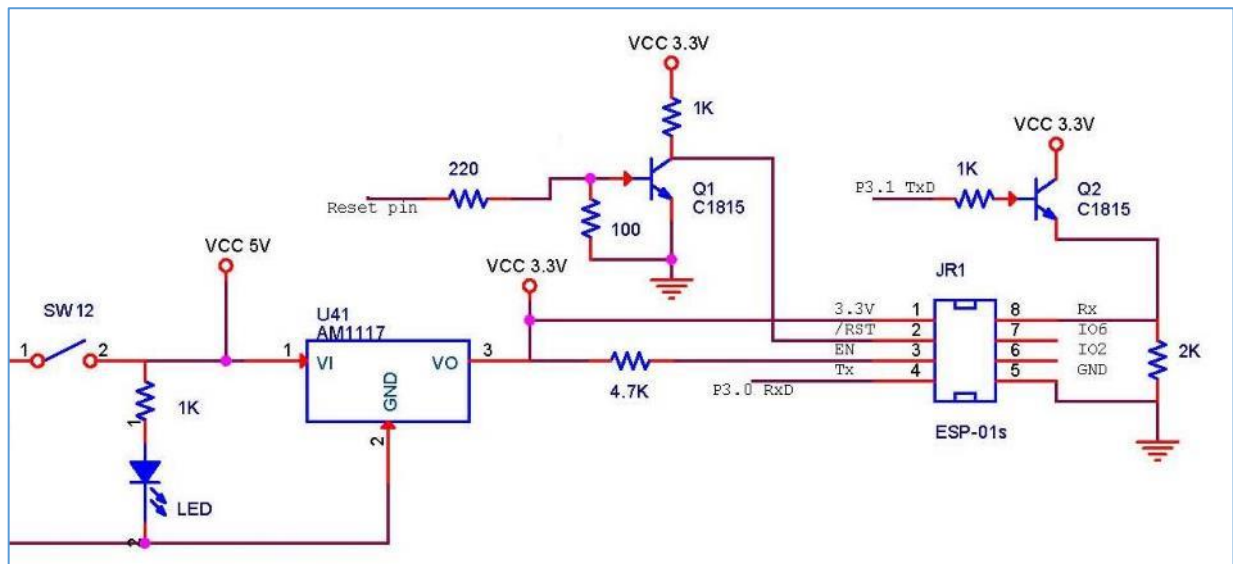


接給它 5V 灌下去也還好，但不能因為 ESP-01S 便宜就這樣摧殘它啊。

(六)、我為了將 8051 及 ESP-01S 的 RESET 接在一起共用一個按鈕，沒注意一個是高電位觸發(8051)，一個是低電位觸發(ESP-01S)，結果 ESP-01S 其實一直沒動作(處在 RESET 狀態)，搞了好久不知道它為什麼不發訊號，一直查程式卻找不出問題，這就是不好好研究晶片手冊的結果。後來也是用一個電晶體當反向器來解決。RESET 在 USPISP 插入後又有問題，從 USBISP 出來的 RESET 腳有一個固定 0.6V 的電壓，導致 ESP-01S 又被 RESET 了，過程中一直不懂為何用單純的 USB 供電可以動作，用 USBISP 供電就不行，原來是這個原因，後來在電晶體 B 極和地之間接一個電阻，確保無訊號時必須低於晶體開關電壓就行了。

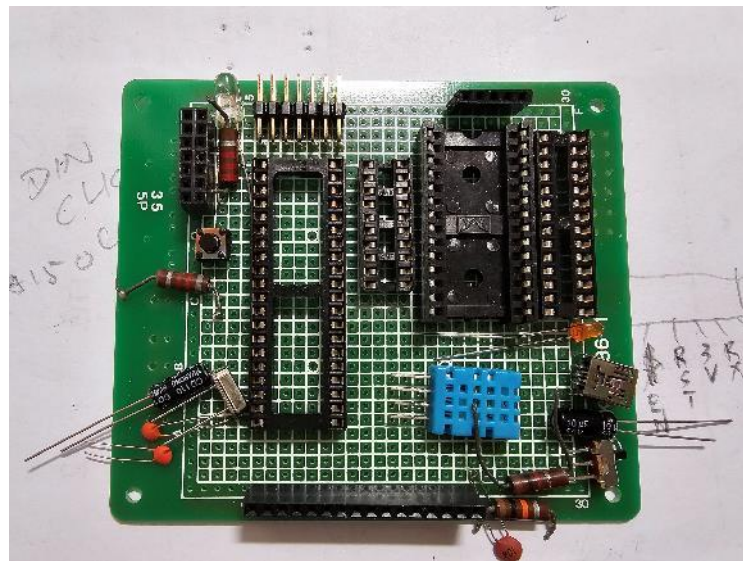
RESET 電路和 8031/51 Tx 轉換電路如下：



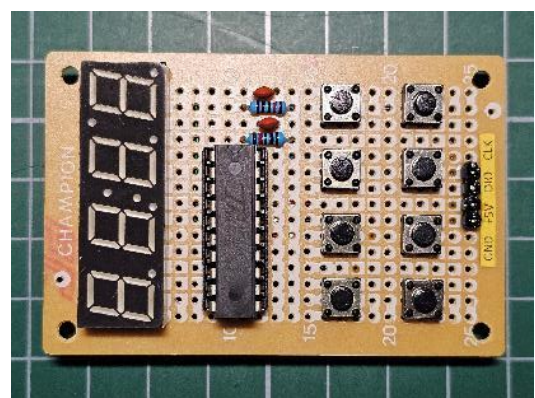
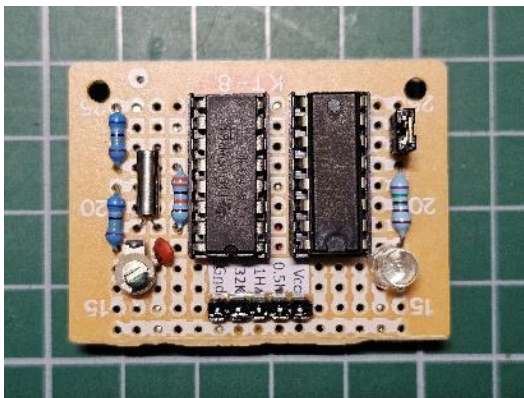


### 三、零件布置

這次手邊的洞洞板比較小，那就往上發展吧，只把顯示介面的排針座做出來，TM1637 的七段顯示器模組往上安裝。另外使用了一個之前做的 1Hz 訊號產生器模組，當作這個系統的定時中斷輸入，也是只留了排針，有需要時再插上。



試排一下零件，看如何安排較方便。



使用之前做的現成的模組(CD4060 1Hz、TM1637 七段顯示器)

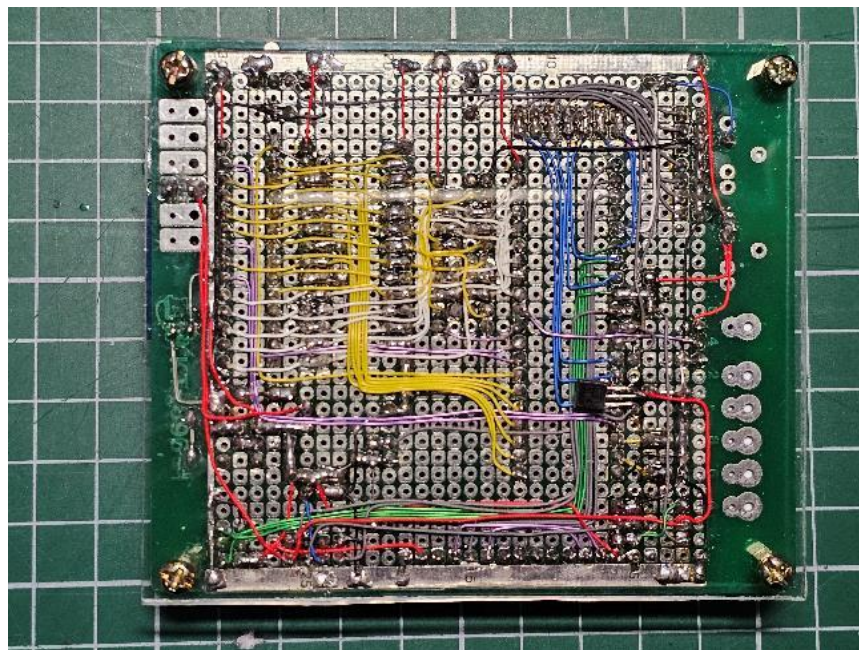


## 四、動手實作

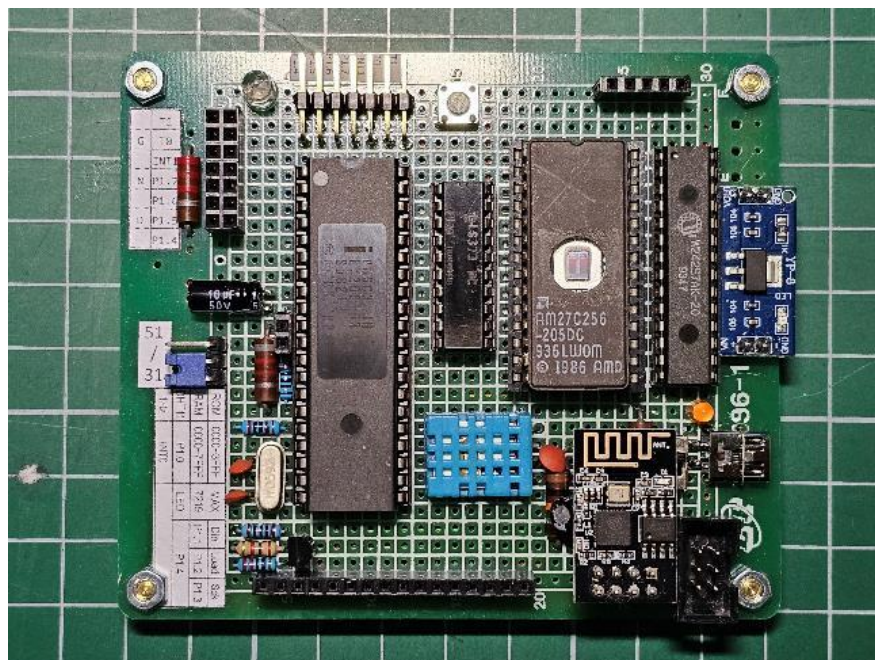
這裡分為兩個部分，一個是 8031/51 的板子，一個是 ESP-01S 的程式。

### (一)、8031/51 板：

1. 如同前兩次的實作，這裡應該沒什麼問題，細心一些不要焊錯線，不要冷焊，不要短路就好。
2. 我沒有加上電源電路，直接使用 USB 的 5V 供電，然後使用現成的 AM1117 3V 降壓模組來提供 5V 轉 3V 的功能給 ESP-01S 用。
3. 為了驗證連網功能，加上了一個 DHT11 溫溼度感應器，預計每隔一段時間由 8031/51 透過 ESP-01S 向伺服器端發送溫溼度數據。
4. (1). 完成接線的背面，追加的電晶體沒地方塞，只好焊在背面了。

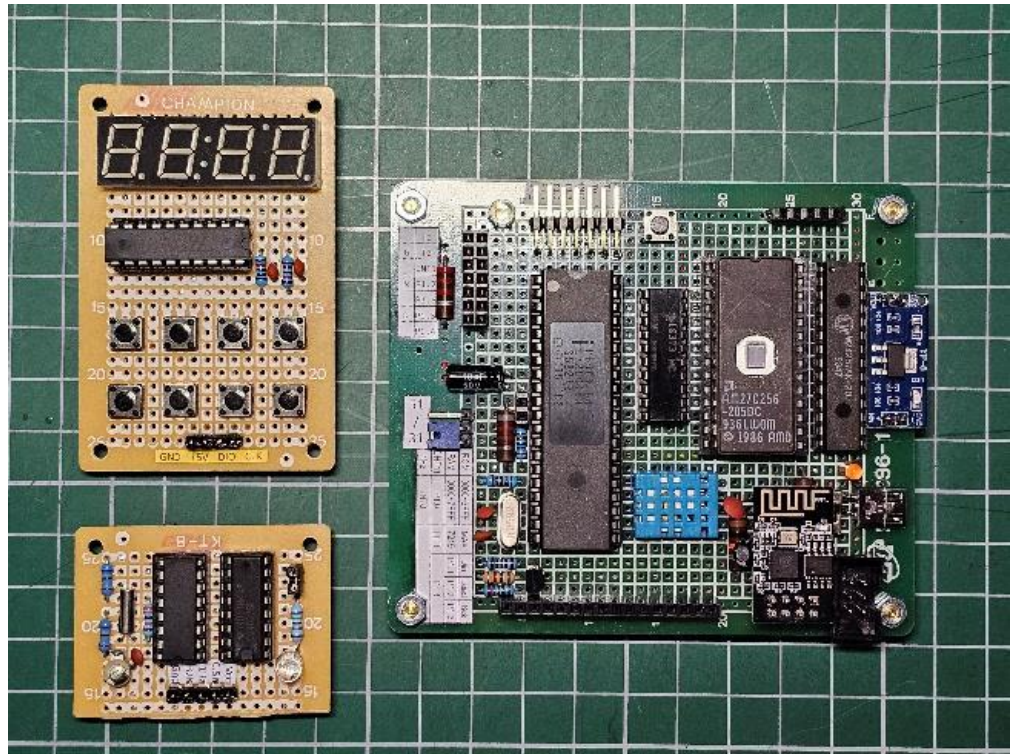


- (2). 正面。AM1117 模組有點跑到板子外面去了。

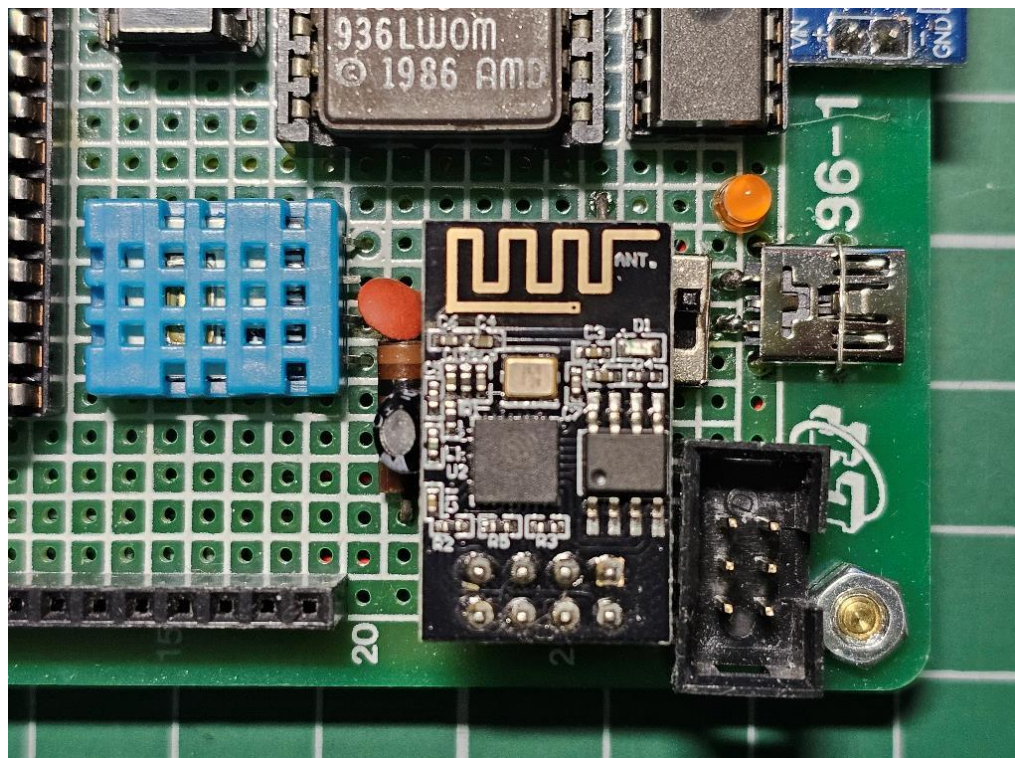




(3). 主板加兩個模組。

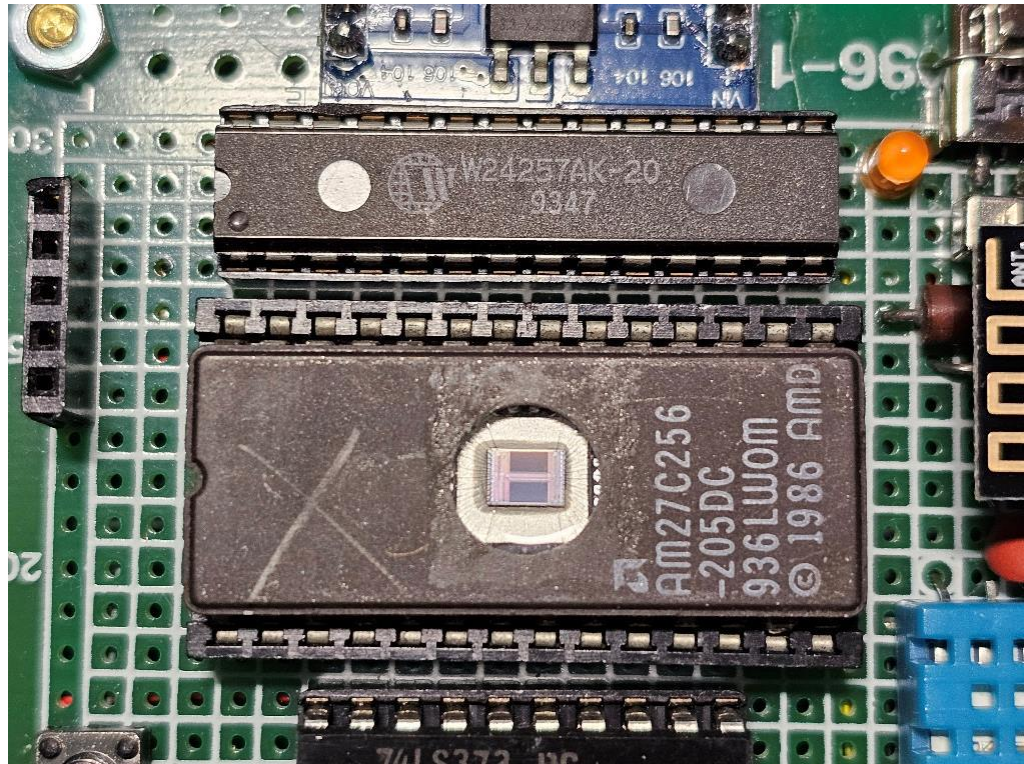


(4). 小巧的 ESP-01S 插在板子上。

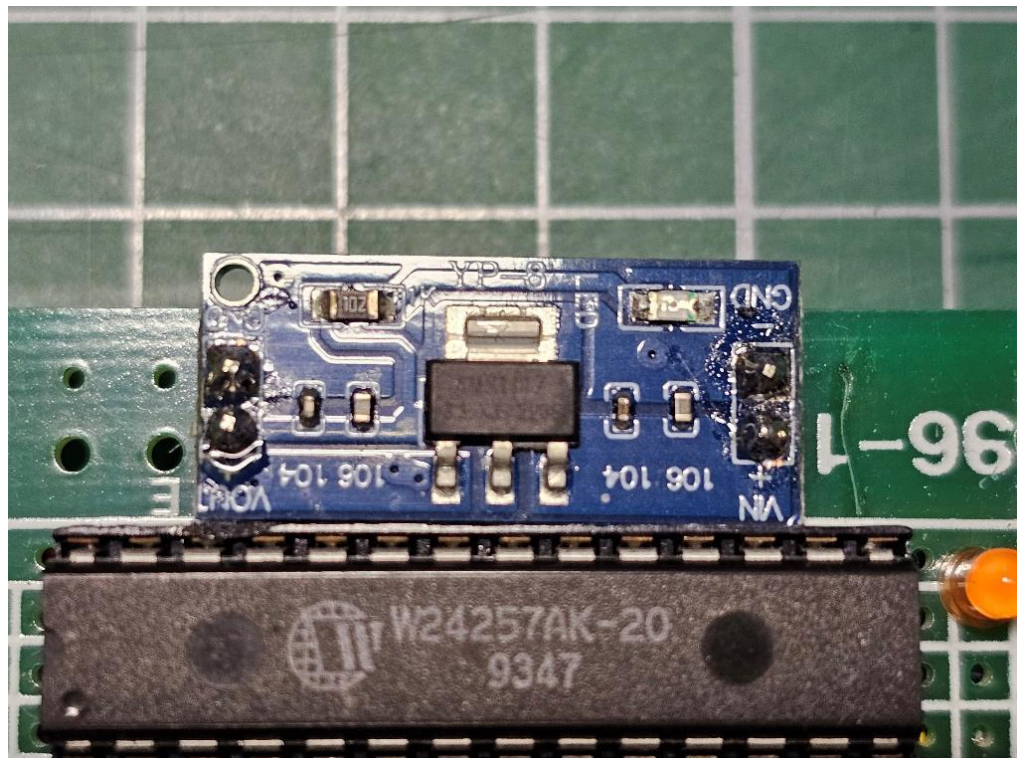




(5).ROM 和 SRAM，看得出來容量一樣的 RAM，W24257AK 硬是瘦身一半。

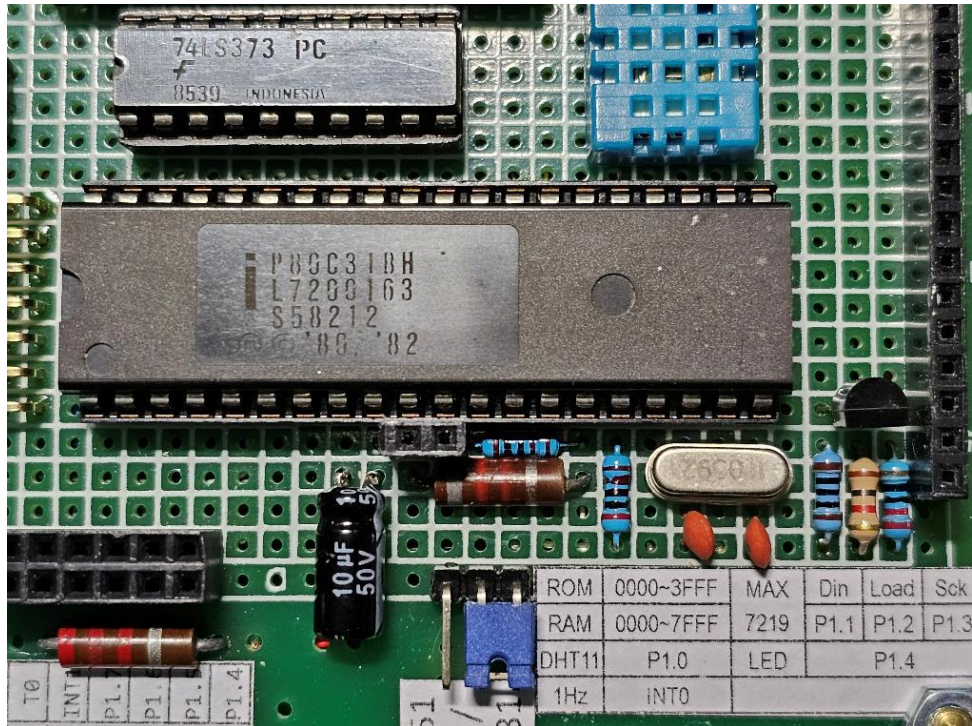


(6).AM1117 其實和常用的 7805 是一樣的三支腳。

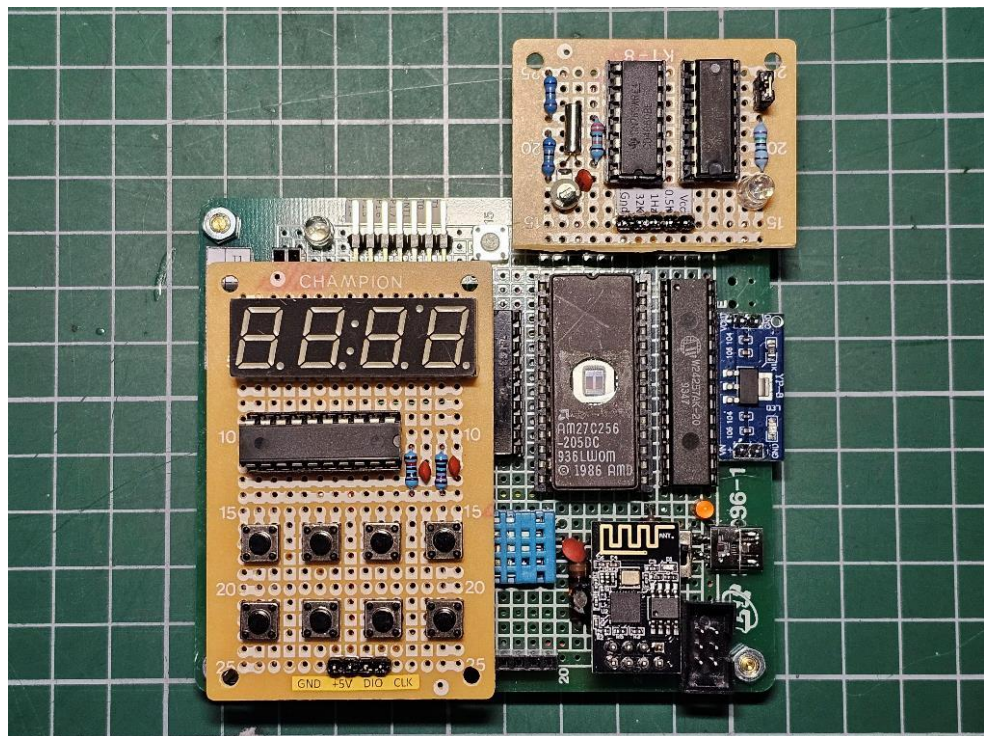




(7.). CPU 和 RESET 電路、震盪晶體電路。



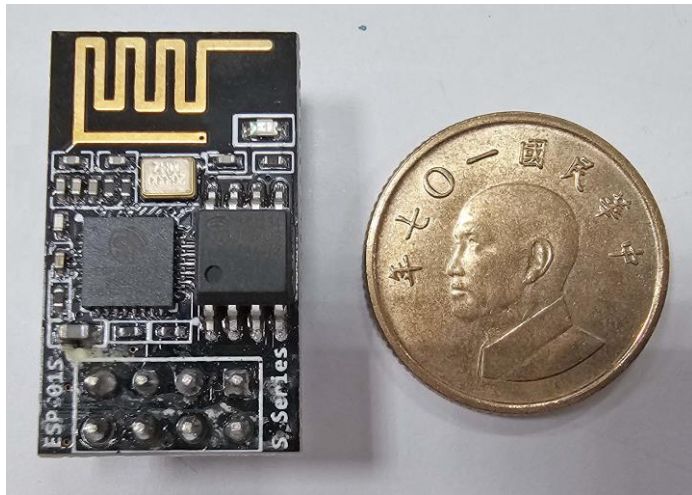
(8). 組合起來的樣子。



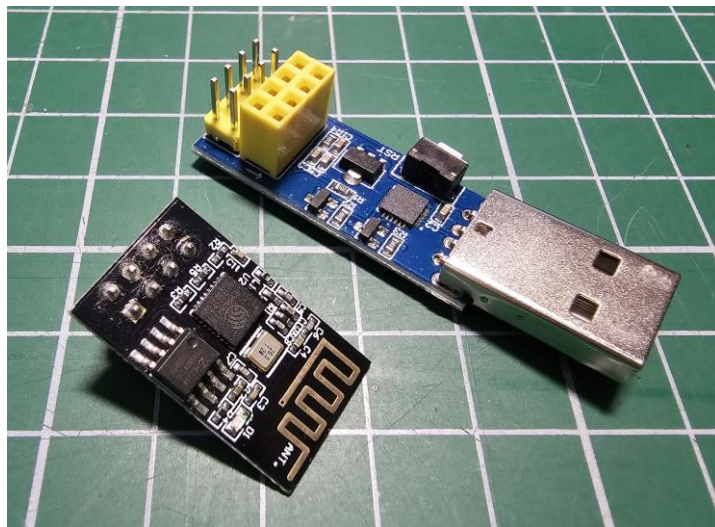
(二)、ESP-01S 的部分：

1. 這部分不需要做什麼，只是 ESP-01S 需要加上一個 USB 的介面才能跟 PC 溝通，可以購買現成的裝置，一個就可以用很久。然後可以用 Arduino IDE 來寫程式。ESP-01S 的大小只有一元硬幣大麼大。





2. 必須有一個 ESP-01S 專用的 USB 模組才能和 PC 溝通。



3. 將 ESP-01S 插入 USB 模組後再插入 PC 的 USB 即可。





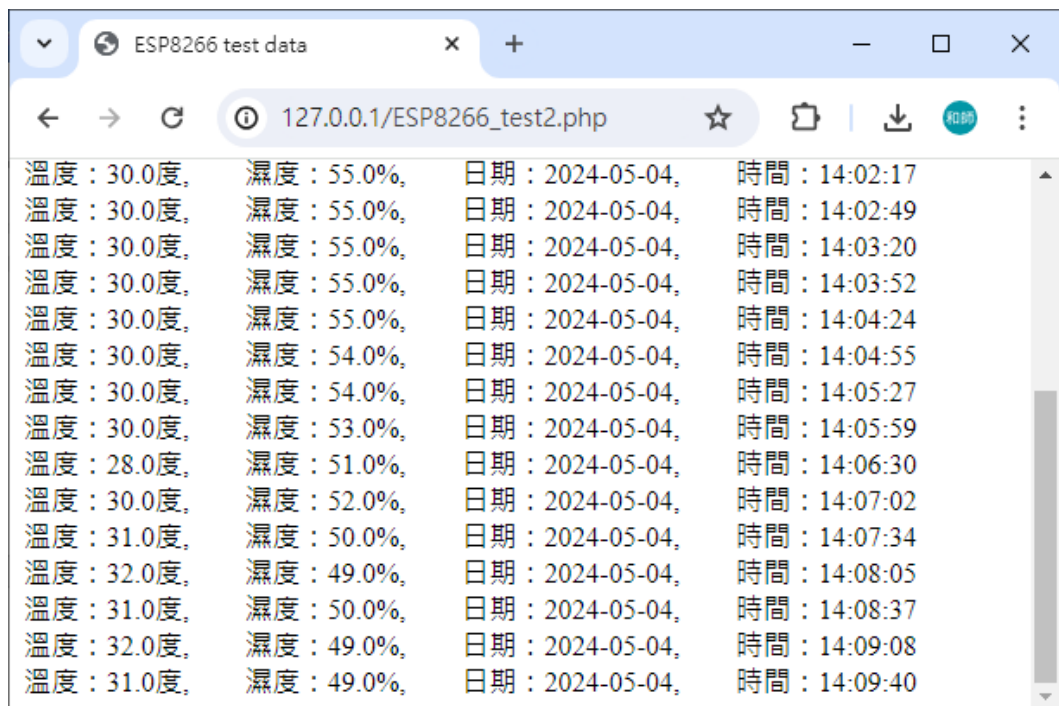
## 五、程式及測試

由於 8031/51 板子和 ESP-01S 是分開寫各自的程式的，所以要先預想好兩邊溝通的流程和傳送的數據。

(一)、ESP-01S 使用 Arduino IDE 來撰寫程式，我讓它連上我的測試伺服器的網頁，然後等待 8031/51 送溫溼度的數據來，便將數據發送到伺服器，就這麼單純的動作。基本上 ESP-01S 也是透過它那拉出來的一組串列通訊埠跟 PC 溝通，所以撰寫完程式後可以使用 Arduino IDE 的序列埠監控窗來試著發送一些數據給 ESP-01S，然後看看網頁伺服器是否有收到，也可以使用 RealTerm 軟體來模擬當作 8031/51 的串列埠和 ESP-01S 溝通看看。



如果都 OK 了，可以刪除在和 Arduino IDE 溝通時的一堆 Serial.println() 指令，因為插到板子上和 8031/51 溝通時是無效的指令，我的程式 8031/51 的部分目前並不打算處理 ESP-01S 發送過來的訊息。下圖是網頁伺服器接收到的數據。



溫度：30.0度,	濕度：55.0%,	日期：2024-05-04,	時間：14:02:17
溫度：30.0度,	濕度：55.0%,	日期：2024-05-04,	時間：14:02:49
溫度：30.0度,	濕度：55.0%,	日期：2024-05-04,	時間：14:03:20
溫度：30.0度,	濕度：55.0%,	日期：2024-05-04,	時間：14:03:52
溫度：30.0度,	濕度：55.0%,	日期：2024-05-04,	時間：14:04:24
溫度：30.0度,	濕度：54.0%,	日期：2024-05-04,	時間：14:04:55
溫度：30.0度,	濕度：54.0%,	日期：2024-05-04,	時間：14:05:27
溫度：30.0度,	濕度：53.0%,	日期：2024-05-04,	時間：14:05:59
溫度：28.0度,	濕度：51.0%,	日期：2024-05-04,	時間：14:06:30
溫度：30.0度,	濕度：52.0%,	日期：2024-05-04,	時間：14:07:02
溫度：31.0度,	濕度：50.0%,	日期：2024-05-04,	時間：14:07:34
溫度：32.0度,	濕度：49.0%,	日期：2024-05-04,	時間：14:08:05
溫度：31.0度,	濕度：50.0%,	日期：2024-05-04,	時間：14:08:37
溫度：32.0度,	濕度：49.0%,	日期：2024-05-04,	時間：14:09:08
溫度：31.0度,	濕度：49.0%,	日期：2024-05-04,	時間：14:09:40

(二)、8031/51 的部分，我讓它每隔一段時間讀取一次 DHT11 的數據，然後發送給 ESP-01S，並顯示在 TM1637 的七段顯示器上，就是這麼簡單。間隔的計時由 CD4060 1Hz 模組每秒觸發 INTO 一次來計算（單晶片的中斷使用很常見，要會），DHT11 的手冊說每次的讀取不可以太接近，所以我訂個 15~30 秒讀取 DHT11 一次。

由於 TM1637 使用 IIC 介面，傳輸訊號時需要較精確的 CLK，DHT11 只有一支訊號腳，讀取資料時也要精確的時間計算，和 ESP-01S 溝通的串列埠的 Baud rate 也要精確計時，這些都要用到 8031/51 的 Timer，所以要好好分工一下，不然很容易互相干擾。

(三)、所以整個系統的程式分為三個部分：

1. ESP-01S：要用 Arduino IDE 的 C 語言。
2. 8031/51：要用 8051 的 C 語言。
3. 網頁伺服器：我用 PHP(其它網頁後端語言都可以)寫一個接收並處理、顯示數據的網頁。



## 六、心得

系統整合的難度高於單純的獨立系統，在這個專案裡需要動用多方的能力，除了硬體的理解、設計和製作，還有各種語言和軟體使用。我使用了自己架設的網頁伺服器(Appserv 套件)和 PHP 來處理 ESP-01S 送來的數據，只要 ESP-01S 和我的伺服器在同一個網域即可(在家裡很方便，都連上家裡的無線 IP 分享器即可)。

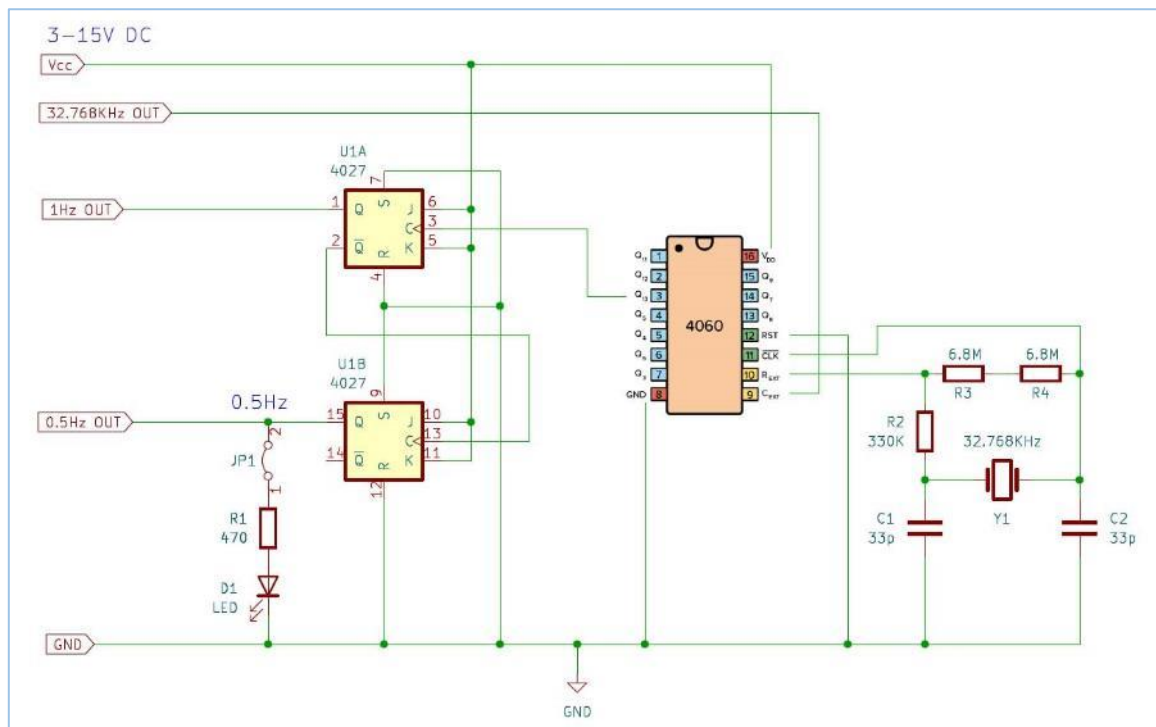
雖然 ESP8266、ESP32 和其它各種板子已經整合得很好也超方便使用了，但這個專案，嗯，就是沒事找事做嘛~，我就一定要讓 8051 上網啊，從硬體到軟體，滿滿的成就感啊。



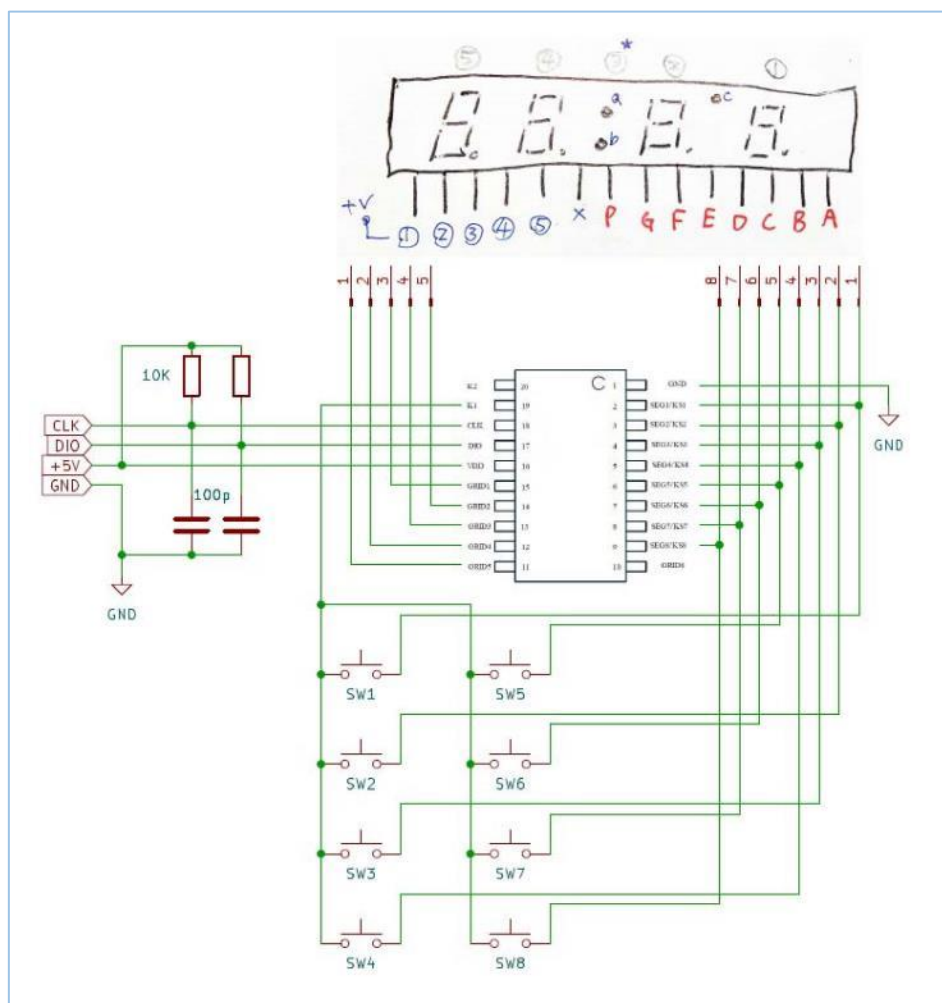


Date: Saturday, May 04, 2024	Sheet 1 of 1
------------------------------	--------------





CD4060 + CD4027 組成的 1Hz 訊號產生器



TM1637 七段顯示器模組