元　　智　　大　　學

電機工程學系（所）

碩　士　論　文

無線紅外線額溫槍

Wireless Infrared Forehead Thermometer

研 究 生：邱彥錞

指導教授：趙燿庚 博士

中 華 民 國　一一0　年　五　月

元　　智　　大　　學

電機工程學系（所）

碩　士　論　文

無線紅外線額溫槍

Wireless Infrared Forehead Thermometer

研 究 生：邱彥錞

指導教授：趙燿庚 博士

中 華 民 國　一一0　年　五　月

無線紅外線額溫槍

Wireless Infrared Forehead Thermometer

研 究 生：邱彥錞 Student：Chiu Yen-Chun

指導教授：趙燿庚 Advisor：Chau Yaw-Geng

元智大學

電機工程學系(所)

碩士論文

A Thesis

Submitted to Department of Electrical Engineering
College of Electrical and Communication Engineering
Yuan Ze University
in Partial Fulfillment of the Requirements
for the Degree of
Master of Science
in
Electrical Engineering
June 2021

Chungli, Taiwan, Republic of China

中華民國 110 年 5 月

無線紅外線額溫槍

學生:邱彥錞　　指導教授: 趙燿庚 博士

元智大學電機工程學系（所）

中文摘要

透過論文，我們研究與實現用紅外線溫度計建立溫度監測的無線平台。我們使用紅外線溫度感測器 SEN0206 並與 ESP32 MCU 整合在電路板上。其中 ESP32 與其藍芽模組提供無線傳輸功能。

感測器量測溫度會發送到雲端服務 Thingspeak，Thingspeak 可做進一步分析。除溫度監控與分析外，還可以時下流行的 Line 即時訊息來傳送量測到的溫度。一旦溫度超過預設值，Line 可自動將感測器量到的溫度傳給手持裝置。本論文中開發的系統可應用在醫護或工業應用領域。

Wireless Infrared Forehead Thermometer

Student: Chiu Yen-Chun Advisor: Dr. Chau Yaw-Geng
Department of Electrical Engineering
Yuan Ze University

Abstract

In the thesis, we studied and implemented a wireless platform for temperature monitoring using an infrared thermometer. The thermometer sensor SEN0206 is used and integrated with an ESP32 MCU on a circuit board, where ESP32 with its Bluetooth module provides the wireless transmission function.

The sensed temperature is sent to the ThingSpeak cloud, where further analysis can be performed. In addition to temperature monitoring and analysis, the popular LINE instant massager is employed for real-time signaling of the measured temperature. With LINE notice, the sensed temperature can be sent to a mobile device once the temperature is crossing a preset threshold. The developed system can be applied to health caring or industrial areas.

.

誌謝

本身是電子工程系，自從考上元智電機工程系後就和其他電機系同學一樣，榮耀的成為電機工程系的一份子，和所有的同學一起努力求學，在電機系以來一直感謝所有的師長在課業上所給予的指導，而且對我們不懂的地方熱心的教導我們，在這裡我們也必須感謝所有的電機同學使我們並不感到孤單無助，在這裡我必須特別感謝趙燿庚老師, 他使學生有充分的空間和資源可以專心的 從事研究計畫，並在每個挑戰中提供最適切的指導與協助，學生才能完成這一篇專題，對於教授師恩，永銘於心。

目錄

圖目錄

第一章 序論

1.1 研究背景

隨著科技的進步，溫度對人們而言不再只是單純的幾個數字而 已了，它同時代表了你的身體狀況，是否發燒啦！或是今天要不 要多帶一件衣服啦！或者是今天的溫度不適合出去玩等等，溫度 對科學來說更是格外的重要，例如電容最高的耐熱溫度啦！高溫 爐的溫度夠不夠啊！或者是電腦的最大耐熱溫度是多少啦！只要所測量的溫度高個一兩度，再精密的儀器可能都會報銷的，而溫度計在我們的日常生活中也已經是非常常見的物品，從小時候的水銀溫度計，做自然科學實驗的酒精溫度計到市面上都有賣的電子溫度計，可見溫度是多麼的重要，而要怎麼正確又快速的量 出溫度就變成了一個難題，而有些人又會覺得市面上所販售的水 銀溫度計只能量體溫，而酒精溫度計只可以量室溫，而且類比式 的溫度感應器容易有誤差，電子溫度計也需要一分多鐘才測的出 數值，紅外線的溫度計卻是貴的不敢買下手，為了增加溫度計的 多功能性和方便性，為此我們必須去思考如何才能降低價格，增加準確率，和容易使用的溫度計。

1.2 研究目的

根據 COVID-19，其相關的抗流行病產品（如口罩，酒精，紅外溫度計和其他產品）已售罄。此外，市場上的紅外溫度計僅支持檢測實時溫度和實時讀取溫度的功能。然而，這些紅外線測溫儀沒支援互聯網與數據分析。同時，我還研究了市場上主流 MCU 供應商，例如 Nuvoton、Holtek、Sonix 和其他 MCU 供應商(ESP32)。

這些主流 MCU 供應商沒有提供 WIFI 功能。 但 Espressif ESP32 具有內置的 WIFI 功能。 這就是為什麼我想使用 ESP32 設計。

第二章 額溫槍簡介

溫度感測有許多的技術，例如熱敏電阻、電熱偶、DS1821、MLX90614 等方法，而在本研究將使用 MLX90614 紅外測溫度感測元件，红外線測溫技術是一種便捷、準確非接触式測溫技術。红外測溫可實現在其視場範圍内對難以接觸區域或危險區域進行連續、即時溫度檢測，有效降低測溫作業危險系數且具有體積小、精度高、可組網及實時性能好等優點。

採用數字式红外溫度傳感器 MLX90614 作為溫度檢測器件， 以 Espressif ESP32 MCU 為核心，OLED 顯示幕作為顯示器， 設計實現多點红外溫度測量系统。該系统優點為：測溫精度高；測量不影響溫度場的分布；非接觸式溫度測量，降低危險系數；響應時間短，易於實現動態測量。經過了 COVID-19 大家也都很清楚溫度對人們重要性，可以隨時知道自己身體狀況。

2.1.額溫槍材料

本專案使用材料(BOM)如下

1. MCU (Micro controller Unit)

a. Espressif ESP32 x 1 pcs

b. Its Module is WROOM-32S and model no is DFR0478

2. IR Temperature Sensor :

a. Sensor Model is SEN0206 with 3.3.V and connect by I2C x 1 pcs

b. Its senor IC is MLX90614 of Melexis

3. OLED Display

a. Its display size is 0.96 inch

b. Its driver IC is SSD1306 with 3.3V and connect by I2C x 1 pcs

4. Battery

a. Its type is lithium-ion rechargeable battery with 3.7V

b. Its model is 18650

2.2 額溫槍架構

系統架構如下圖

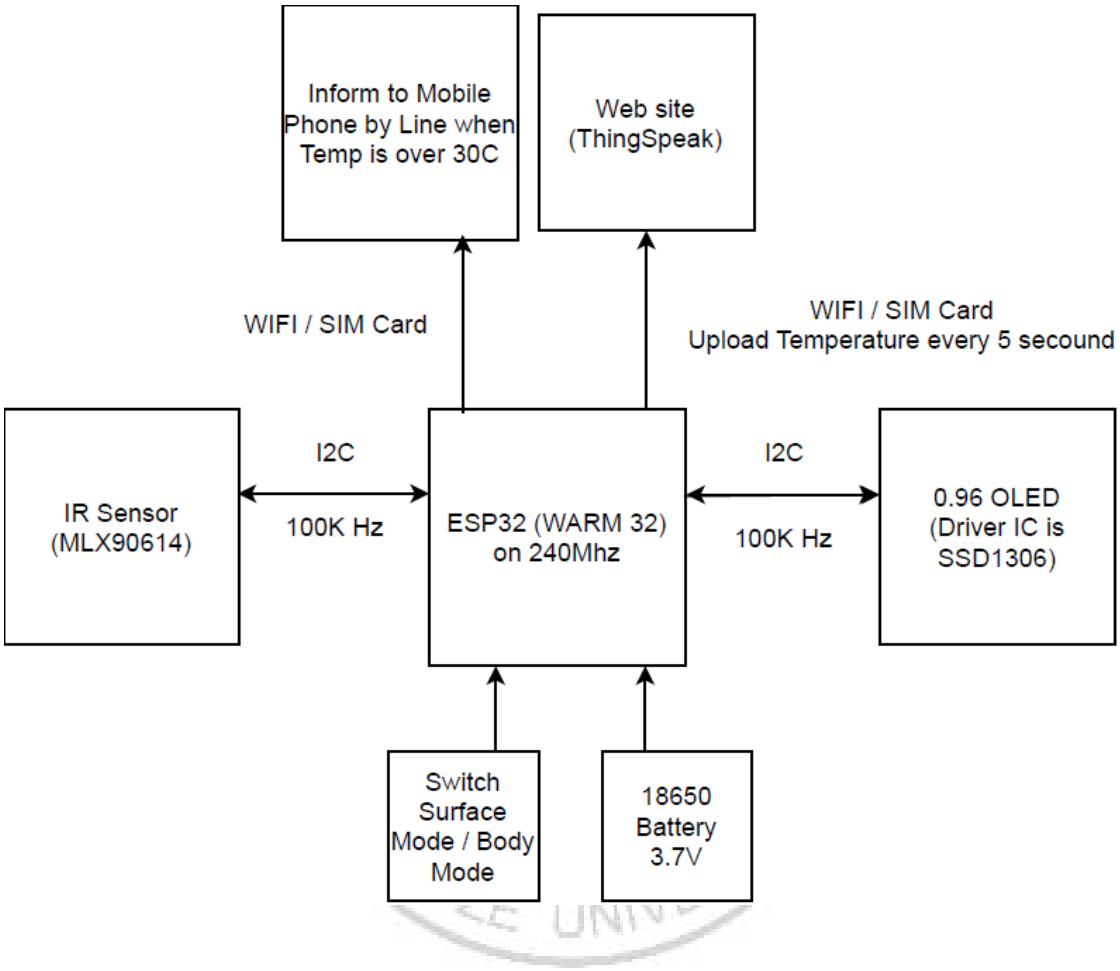


圖 1 ： 無線紅外線額溫槍系統方塊圖

2.3 技術原理

紅外線額溫槍是一種利用偵測物體表面的紅外線量進而達成溫度測量的溫度計。其原理就是用物體散發出的熱，換算成溫度。

將收集被測物體輻射的紅外能量並將能量聚集於檢測器上。經放大並顯示為溫度讀數。物體越熱，其分子就愈加活躍，紅外線量越大說明溫度越高,所以它所發出的紅外能量也就越多。

使用紅外線額溫槍不可離額頭太遠，一般規格顯示為 10cm 以內，這是因為單點式的感測器的點解析度會隨著距離越遠而變大（如下圖），因此測到的溫度誤差比較大。

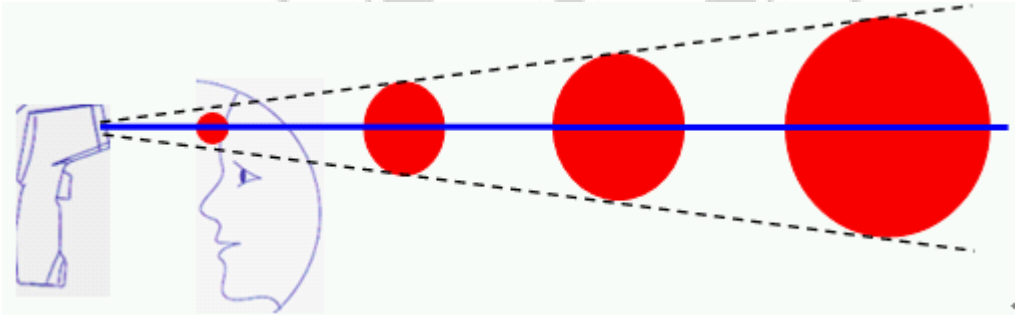


圖 2 ： 紅外線額溫槍量測距離

紅外線額溫槍是利用黑體輻射來量測物體表面散發出輻射熱的能量。任何物體只要溫度在絕對零度（-273.15℃）以上，就會發射出電磁輻射，而它所發射出來的電磁波頻譜和它本身的溫度有關。人類其實從史前時代就已經觀察到這個現象，當我們進入青銅時代、鐵器

時代，開始冶煉金屬時，被高溫熔融的金屬隨著溫度的不同，會發出紅色、橙色、黃色的光，這就是典型的黑體輻射。其實在室溫之下，所有的物體也都會發出這樣的電磁波，只是因爲溫度低，黑體輻射的波長很長，落在紅外線的範圍內，因此人眼看不到。

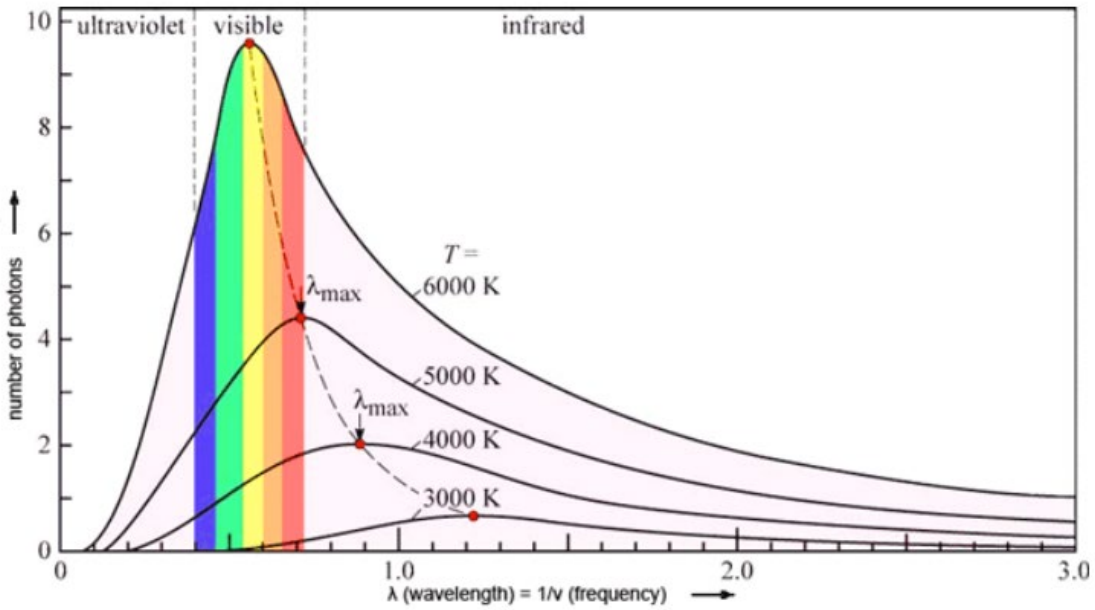


圖 3 ： 黑體輻射頻譜

圖 3 是很典型的黑體輻射頻譜，可以看到物體的溫度在絕對溫度 3000 度、4000 度、5000 度、6000 度時的發射頻譜變化。黑體輻射是個連續光譜，但它的頻譜會有一個高峰，隨著溫度上升，這個高峰的頻率會漸漸往短波長（也就是高頻率）的方向移動，藉由偵測這個頻譜的高峰，我們可以得知發射物體的溫度。

如果有一種感測器，可感應這個範圍的遠紅外線，就可以用它來測量溫度。非接觸式的溫度計與熱像儀就是利用這種原理設計的，我們常用的測量範圍大概在絕對溫度兩百多度到六、七百度左右，這個範圍的黑體輻射落在數千到一萬多 nm 的遠紅外線（與一般紅外線遙控器 800-900 nm 的波長不同），稱之爲 far infrared（FIR）或是 long-wavelength infrared（LWIR）。

利用黑體輻射來測量溫度時，有一個很重要的參數叫做「發射率（ε_ Emissivity」，這個參數大概介於 0 到 1 之間，代表物體發射電磁波的能力與理想黑體之間的比值。理想的黑體發射率是 1，而所有其它的物質發射率都小於 1，一般來說，金屬的發射率很低，非金屬的發射率較高。金屬與非金屬之間的黑體輻射發射率有非常大的差異，以銀來說，它的發射率只有理想黑體的五十分之一，因此在同樣溫度之下，它所輻射出來的黑體輻射就只有理想黑體的五十分之一。 圖 4 說明常見物體發射率對照表

常見物體發射率對照表					
材料名稱	外觀	發射率	材料名稱	外觀	發射率
鋁	氟化	0.3	人體皮膚		0.98
鋁	拋光	0.02-0.04	石墨	氧化	0.20-0.60
黃銅	氟化	0.5	塑膠	不透明	0.95
黃銅	拋光	0.02-0.05	橡膠		0.95
黃金		0.01-0.10	塑膠		0.85-0.95
鐵	氧化	0.7	混凝土		0.95
鋼	氧化	0.70-0.90	水泥		0.96
石棉		0.95	土壤		0.90-0.98
石膏		0.80-0.90	灰泥		0.89-0.91
瀝青		0.95	磚		0.90-0.96
岩石		0.7	大理石		0.94
木材		0.90-0.95	紡織品		0.90
木炭	粉末	0.96	紙		0.95
碳		0.85	泥土		0.90
漆器	無光澤	0.97	沙子		0.92-0.96
碳膠		0.90	沙子		0.9
肥皂泡		0.75-0.80	玻璃		0.85-0.92
水		0.93	紡織品		0.95
雪		0.83-0.90	熱食物		0.95
冰		0.96-0.98	塑料		0.95
冷凍食品		0.95	油		0.94
陶瓷		0.95	鋼鐵		0.80
石灰石		0.98	羊毛	自然的	0.94
油漆		0.93	鉛	氧化	0.5
絕緣膠帶		0.95	黑色油漆		0.96

Note ： 發射率越低，影響測量結果越大。

圖 4 ： 常見物體發射率對照表

第三章 開發環境

這個章節說明開發無線紅線額溫槍使用的整合開發環境 IDE (Integrated Development Environment) 與如何將量測到的溫度透過 WIFI 上傳到網路並記錄每次儲存的溫度,同時判斷量測溫度是否超過指定溫度, 若量測溫度有超過指定溫度,就將量測溫度透過 Line Notify 傳送到手機端的 Line 讓使用者知道待測物/對方體溫有超過指定溫度,要進行後續處理。

3-1 MCU (ESP32-WROOM-32)

3.1.1 規格

•Quick Response(the top frequency is 400KHz)

•Operating voltage: 3.3V

•Input voltage: 3.3V~5.5V

•Support electric current of low power consumption: 10 μA

•Support maximum discharge current: 600mA@3.3V LDO

•Support maximum charge current: 500mA

•Support USB charging.

•Processer: Tensilica LX6 dual core processer

(One for high speed connection; one for independent programing).

•Frequency: 240MHz

•SRAM：520KB

•Flash：2MB (16Mb)

•Wi-Fi standard：FCC/CE/TELEC/KCC

•Wi-Fi protocol: 802.11 b/g/n/d/e/I/k/r (802.11n, high speed can reach to 150 Mbps), converge A-MPDU and A-MSDU, supporting 0.4us protecting interval.

•WIFI Frequency range: 2.4~2.5 GHz

•Bluetooth protocol: Comply with BR/EDR/BLE standard of Bluetooth v4.2•

•Bluetooth audio: the current under low power consumption of CVSD and SBC is 10μA

•Operating current: 80mA in average

•Support one-key downloading.

•Support Arduino and Micro Python

•On-chip clock: 40MHz crystal and 32.768 KHz crystal.

•Digital I/O: 10 (default setting of Arduino)

•Analog input: 5(default setting of Arduino)

•SPI: 1 (default setting of Arduino)

•I2C: 1 (default setting of Arduino)

•I2S: 1 (default setting of Arduino)

•LED_BUILTIN：D9

•Interface: FireBeetle series compatible

•Operating temperature: -40℃~+85℃

3.1.2 尺寸

•Pin Spacing: 2.54mm

•Mounting Hole Space: 24mm/53mm

•Mounting Hole Size: 29.00mm×58.00mm

•Thickness: 1.6mm

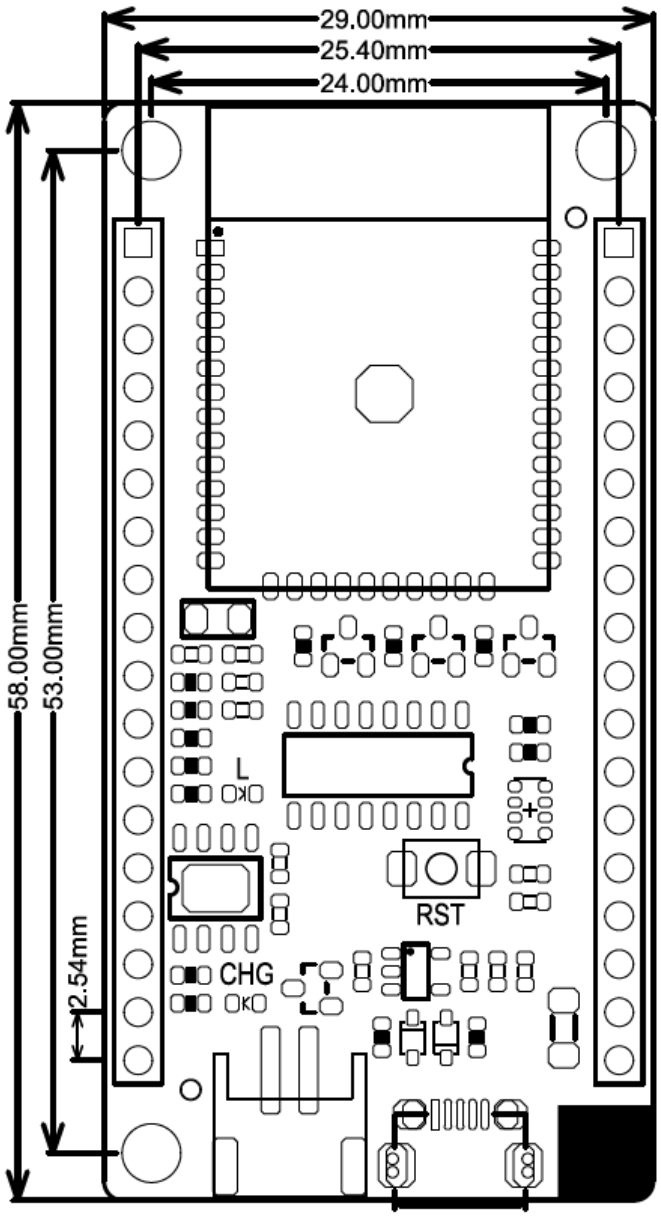


圖 5 ： ESP32 尺寸

3.1.3 接腳圖

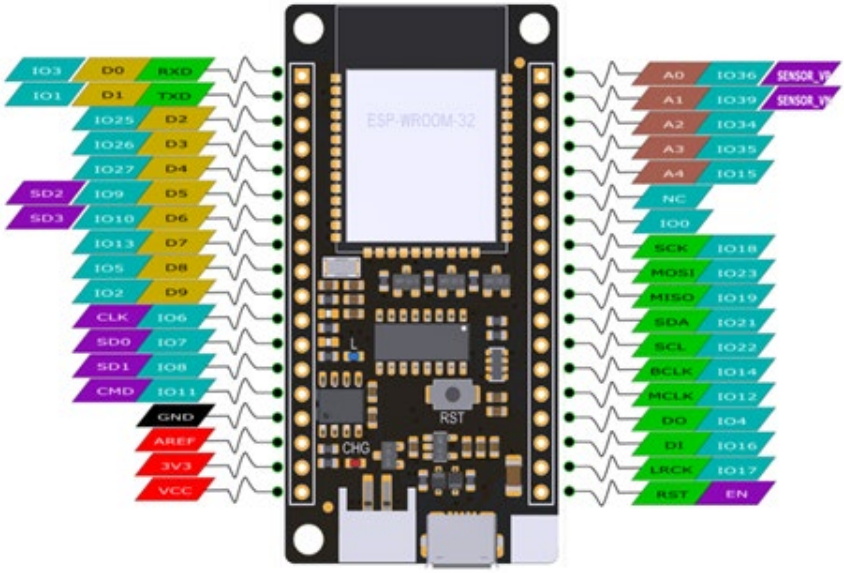


圖 6 ： ESP32 接腳圖

3.1.4 應用場景

•Generic Low-power IOT Sensor Hub

•Generic Low-power IOT Data Loggers

•Cameras for Video Streaming

•Over-the-top (OTT) Devices

•Speech Recognition

•Image Recognition

•Mesh Network

•Home Automation

•Smart Building

•Industrial Automation

•Smart Agriculture

•Audio Applications

•Health Care Applications

•Wi-Fi-enabled Toys

•Wearable Electronics

•Retail & Catering Applications

3-2 Arduino

3.2.1 簡介

Massimo Banzi 之前是義大利 Ivrea 一家高科技設計學校的老師，他的學生們經常抱怨找不到便宜好用的微處理機控制器。西元 2005 年， Massimo Banzi 跟 David Cuartielles 討論了這個問題，David Cuartielles 是一個西班牙籍晶片工程師，當時是這所學校的訪問學者。兩人討論之後，決定自己設計電路板，並引入了 Banzi 的學生 David Mellis 為電路板設計開發用的語言。兩天以後，David Mellis 就寫出了程式碼。又過了幾天，電路板就完工了。於是他們將這塊電路板命名為『Arduino』。

當初 Arduino 設計的觀點，就是希望針對『不懂電腦語言的族群』，也能用 Arduino 做出很酷的東西，例如：對感測器作出回應、閃爍燈光、控制馬達…等等。隨後 Banzi，Cuartielles，和 Mellis 把設計圖放到了網際網路上。他們保持設計的開放源碼(Open Source)理念，因為版權法可以監管開放原始碼軟體，卻很難用在硬體上，他們決定採用創用 CC 許可(Creative_Commons, 2013)。

創用 CC(Creative_Commons, 2013)是為保護開放版權行為而出現的類似 GPL1 的一種許可（license），來自於自由軟體 2 基金會(Free Software Foundation) 的 GNU 通用公共授權條款（GNU GPL)：在創用 CC 許可下，任何人都被允許生產電路板的複製品，且還能重新設計，甚至銷售原設計的複製品。你還不需要付版稅，甚至不用取得 Arduino 團隊的許可。

3.2.2 技術原理

1. 什麼是 Arduino？

Arduino 是基於開放原碼精神的一個開放硬體平臺，其語言和開發環境都很簡單。讓您可以使用它快速做出有趣的東西。

它是一個能夠用來感應和控制現實物理世界的一套工具，也提供一套設計程式的 IDE 開發環境，並可以免費下載。

Arduino 可以用來開發互動產品，比如它可以讀取大量的開關和感測器信號，並且可以控制各式各樣的電燈、電機和其他物理設備。也可以在運行時和你電腦中運行的程式（例如：Flash，Processing，MaxMSP）進行通訊。

圖 7 ： Ardunio+ESP32

2.Ardunio 特色如下

•基於創用 CC 開源的電路圖與程式設計

•Arduino 可使用 ICSP 線上燒入器，將 Bootloader 燒入新的 IC 晶片

•可依據 Arduino 官方網站，取得硬體的設計檔，加以調整電路板及元件，以符合自己實際設計的需求

•可簡單地與感測器，各式各樣的電子元件連接，如紅外線、超音波、熱敏電阻、光敏電阻、伺服馬達等。

•支援多樣的互動程式，如 Adobe Flash, Max/MSP, VVVV, Pure Data, C, Processing 等。

•使用低價格的微處理控制器 如 Microchip ATMEGA328P / ATMEGA168V 、Espressif ESP32 等 USB 介面，不需外接電源。另外有提供直流（DC 5V）電源輸入。

3.可依照下面步驟將程式碼透過 Arduino 燒綠到 ESP32

Step 1. 在 Windows 安裝 Arduino (版本建議用 1.8.13)與 ESP32 Driver

Step 2. 執行 Arduino

Step 3. Arduino 一開始初始畫面如圖 8

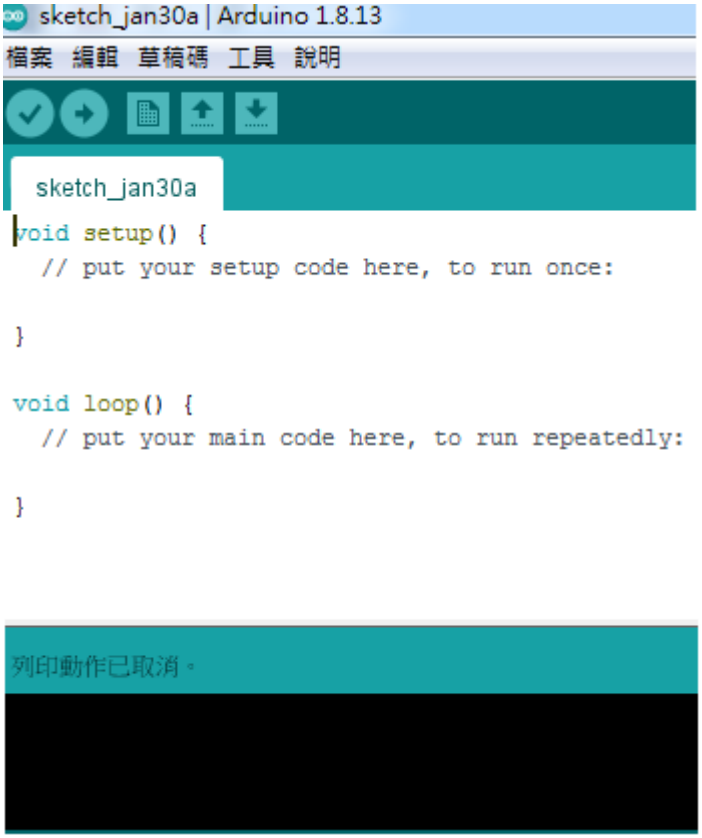


圖 8 ： Arduino 主畫面

Step 4. 選擇檔案→偏好設定如圖 9 and 圖 10。

圖 10 要輸入開發板網址，這樣開發板設定才會載入 Arduino

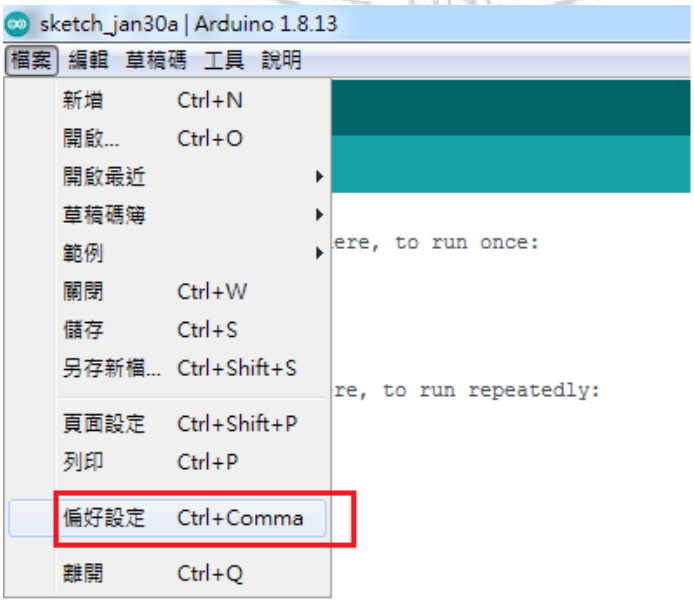


圖 9 ： Arduino 偏好設定

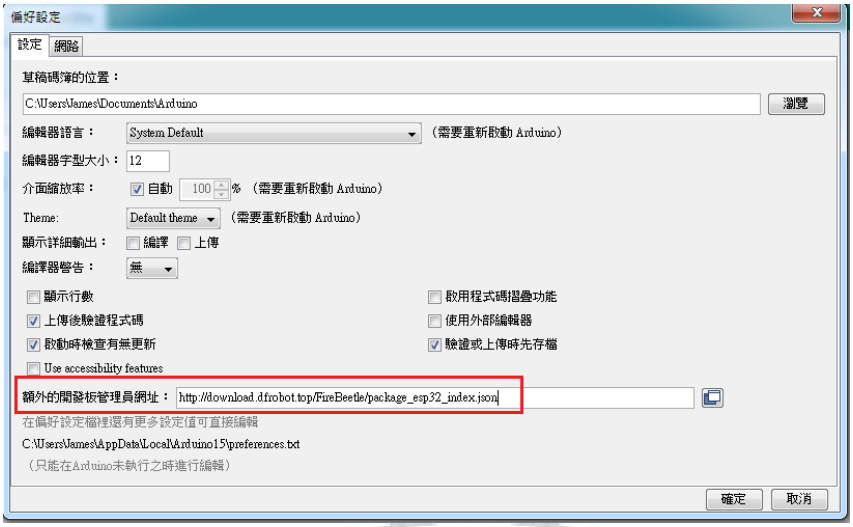


圖 10 ： Arduino 開發板網址

Step 5. 設定開發板型號。工具→開發板→ESP32 Arduino→FireBeetle-ESP32 如圖 11

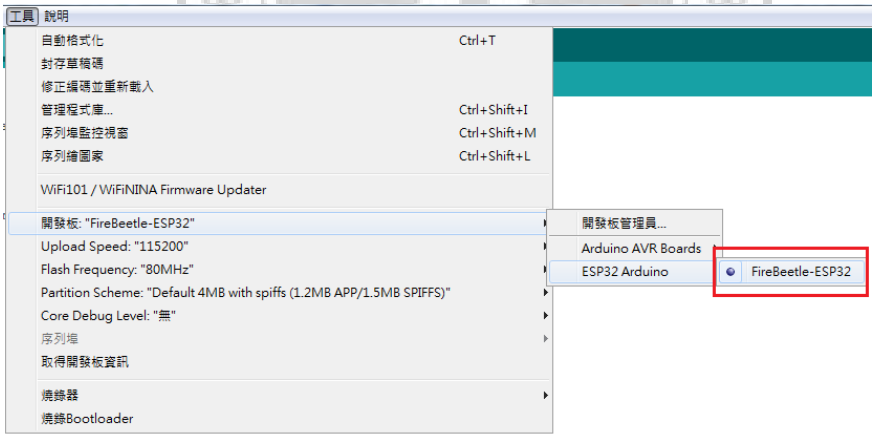


圖 11 ： 設定 ESP32 開發板

Step 6. 設定 Baud Rate。工具→Upload Speed→115200，如圖 12

圖 12 ： 設定 ESP32 Upload Speed

Step 7. 設定 Baud Rate。工具→Flash Frequency→80Mhz，如圖 13

圖 13 ： 設定 ESP32 Flash Frequency

Step 8. 設定 Partition Scheme。工具→Partition Scheme→Default 4MB with spiffs，如圖 14

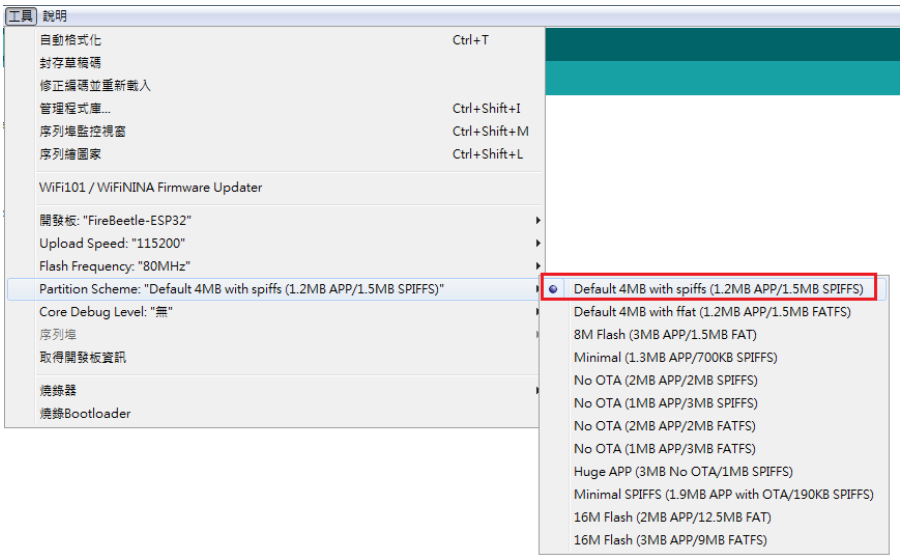


圖 14 ： 設定 ESP32 Partition Scheme

Step 9. ESP32 透過 Micro USB 連接 PC。在 Device Manager 確認 ESP32 COM Port 是否有被偵測。工具→COM Port→COM X，如圖 15

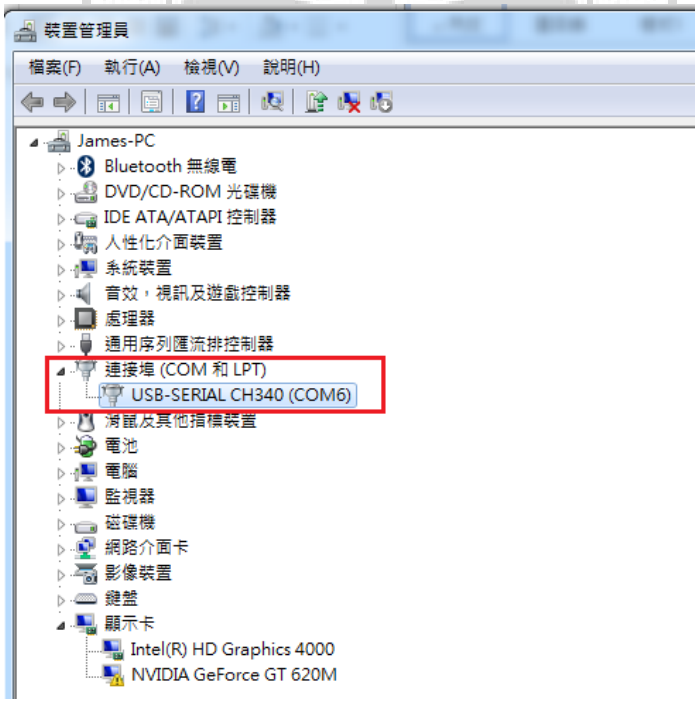


圖 15 ： 確認 ESP32 COM Port 有無偵測成功

Step 10. 開始編輯程式，程式編輯完成後，選擇”V”or 草稿碼→驗證/編譯，如圖 16

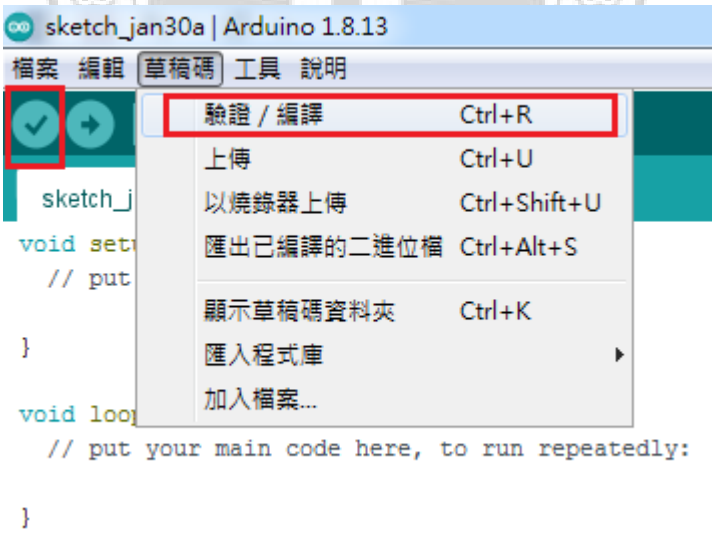


圖 16 ： 編譯程式確認與法有無錯誤

Step 11. 如果程式碼驗證無誤，會看到圖 17 編譯成功訊息

圖 17 ： 編譯程式成功訊息

Step 12.燒錄程式，選擇草稿碼→上傳　或者是按”->”就能燒錄程式碼，如圖 18

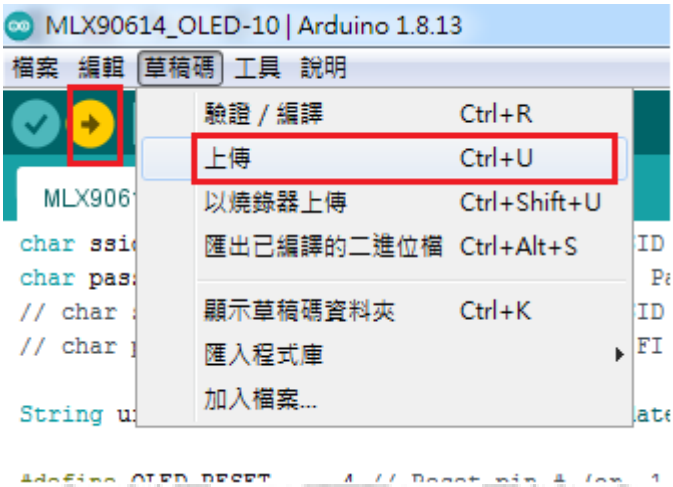


圖 18 ： 開始燒錄程式

程式燒錄成供訊息，如下

草稿碼使用了 885886 bytes（67%）的程式儲存空間。上限為 1310720 bytes。

全域變數使用了 40696 bytes（12%）的動態記憶體，剩餘 286984 bytes 給區域變數。上限為 327680 bytes 。

esptool.py v2.6

Serial port COM6

```
Connecting......
Chip is ESP32D0WDQ6 (revision 1)
Features: WiFi, BT, Dual Core, 240MHz, VRef calibration in
efuse, Coding Scheme None
MAC: b4:e6:2d:8d:4e:79
Uploading stub...
Running stub...
Stub running...
Configuring flash size...
Auto-detected Flash size: 8MB
Compressed 8192 bytes to 47...
Wrote 8192 bytes (47 compressed) at 0x0000e000 in 0.0
seconds (effective 5461.4 kbit/s)...
Hash of data verified.
Flash params set to 0x023f
Compressed 15856 bytes to 10276...
Wrote 15856 bytes (10276 compressed) at 0x00001000 in 0.9
seconds (effective 138.8 kbit/s)...
Hash of data verified.
```

Compressed 886000 bytes to 499695...

Wrote 886000 bytes (499695 compressed) at 0x00010000 in 44.3 seconds (effective 159.9 kbit/s)...

Hash of data verified.

Compressed 3072 bytes to 128...

Wrote 3072 bytes (128 compressed) at 0x00008000 in 0.0 seconds (effective 1445.7 kbit/s)...

Hash of data verified.

Leaving...

Hard resetting via RTS pin...

Note:

可用 Arduino 內建序列埠（COM Port）觀看程式執行結果

3-3 Thingspeak

3.3.1 簡介

ThingSpeak 網站是一個專業物聯網網站，網址是 https://thingspeak.com/。它是一個專為物聯網而產生的應用程式平台，它允許使用者用網路設備即時地將數據上傳到雲端使之聚集在一起(成為資料)。而頻道(Channel)是 ThingSpeak 心臟之所在，您可以切換不同的頻道以存取不同的資料。圖１９是範例將將住家的光線數據上傳到雲端。

圖 19 ： Thingspeak 範例將住家的光線數據上傳到雲端

3.3.2 技術原理

ThingSpeak 是一種物聯網分析服務平台，可讓您 在雲端中收集與儲藏物聯網設備的資料數據。

物聯網設備發佈到 ThingSpeak 的數據，能以圖表即時呈現其時間與數據關係。通過在 ThingSpeak 中執行 MATLAB 程式碼，您可以對數據立即進行線上分析和處理。ThingSpeak 經常用於資料需要進行分析的物聯網系統原型和驗證。Thingspeak 系統架構可參閱如圖 20

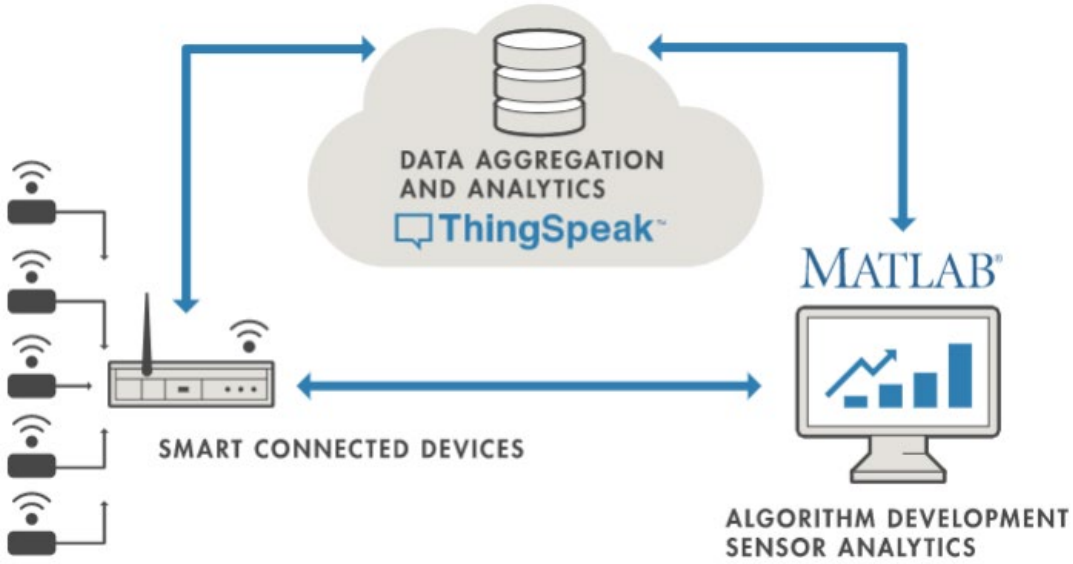


圖 20 ： Thingspeak 系統架構

請依照下面步驟就能透過 Thingspeak 達到數據上傳網路與數據分析

Step 1. 進入網站後，可以到主頁，先點選圖 21 紅框區，先建立一個帳號。

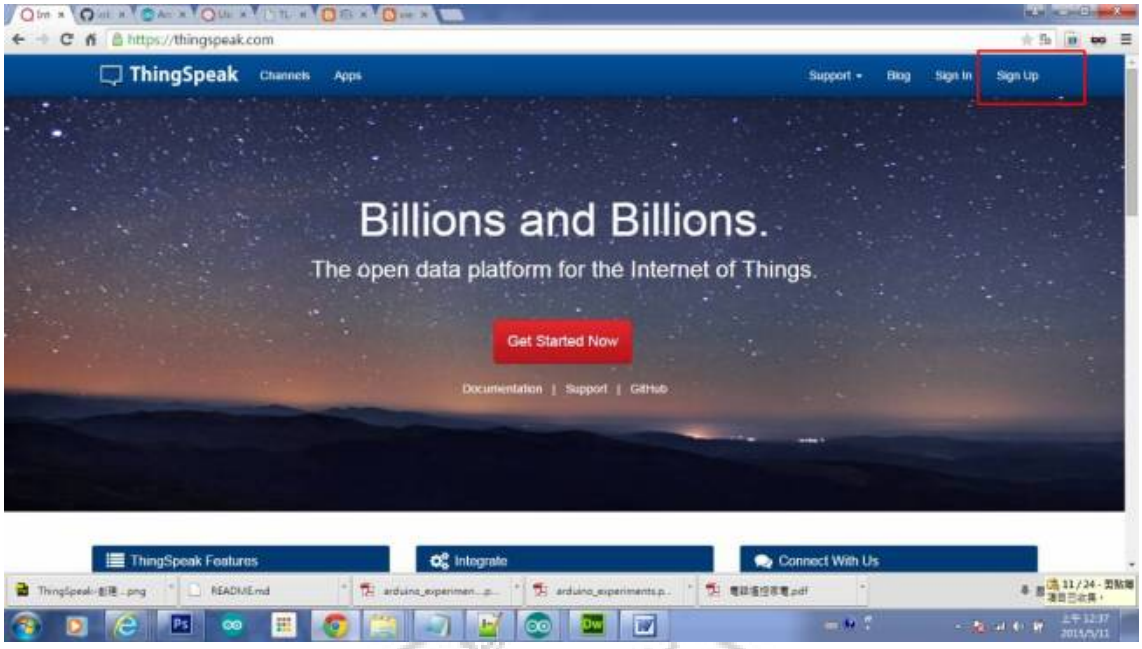


圖 21 ： ThingSpeak 網站主頁

Step 2. 依照圖 22 所示，將資料輸入完畢後，創建一個可以用的帳號。

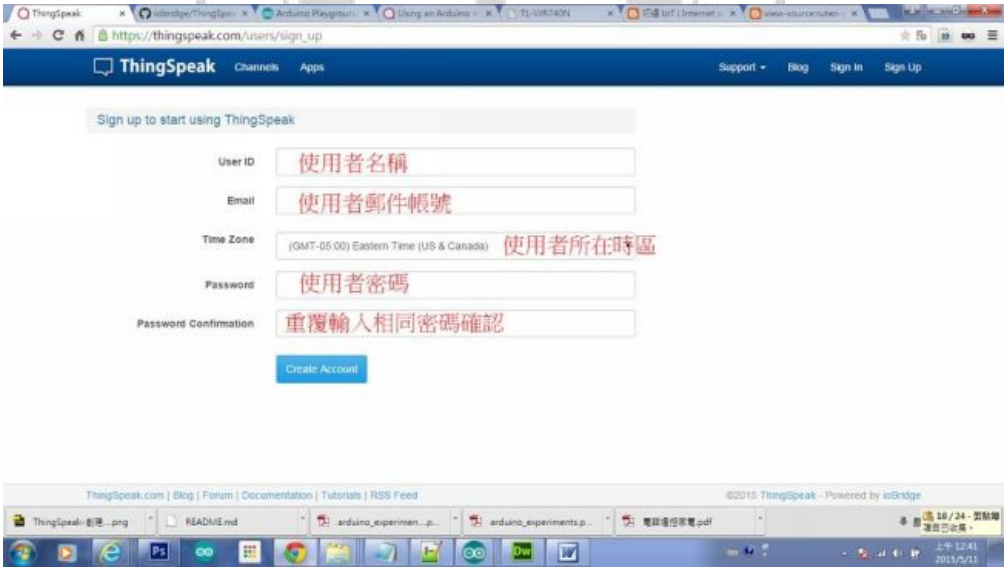


圖 22 ： 註冊 Thingspeak 帳號

Step 3. ThingSpeak 網站創建帳號完成後，會切換到圖 23 畫面

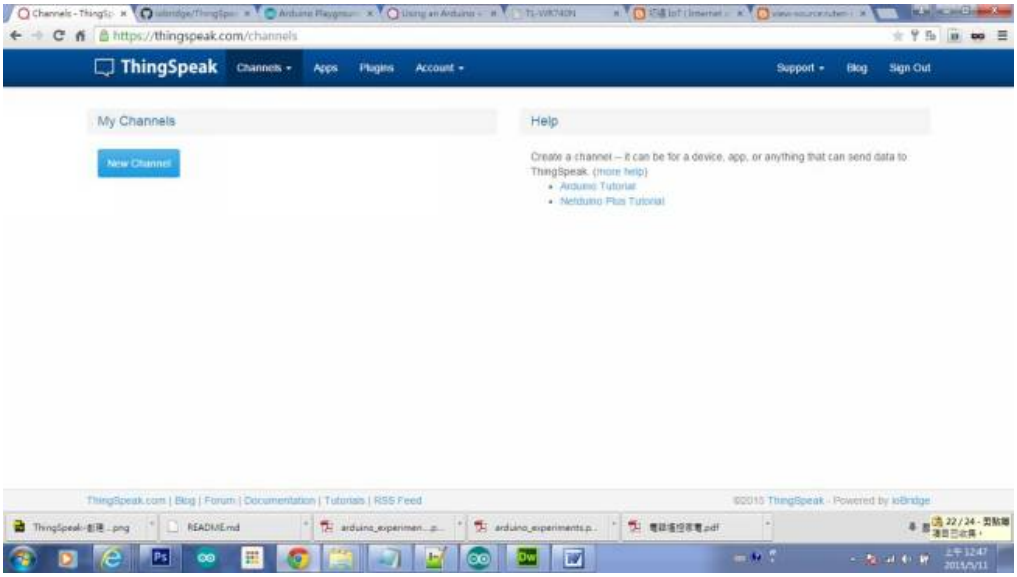


圖 23 ： ThingSpeak 網站創建帳號完成後畫面

Step 4.帳號登入後，先查到使用者目前 Channel

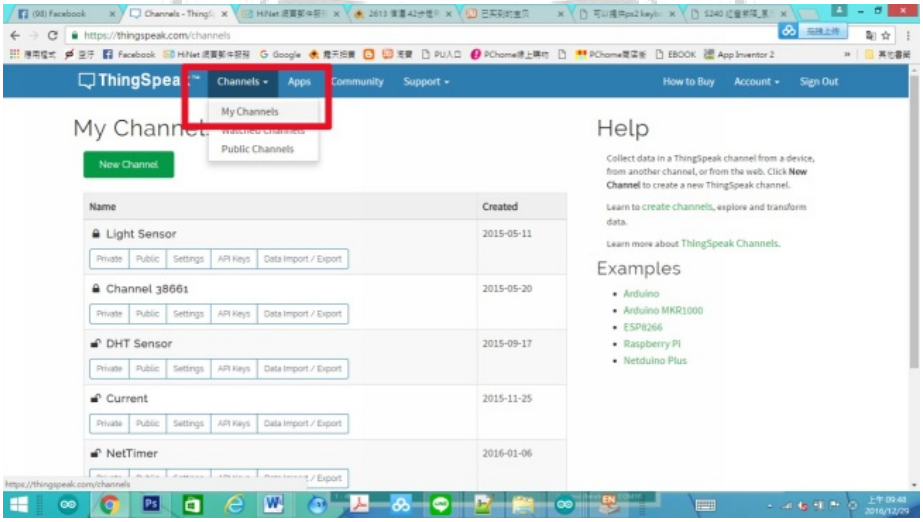


圖 24 ： 使用者目前 Channel

Step 5. 進入網站後，選擇 Channel 區後，由下圖紅框區所示，沒有任何東西，代表讀者需要先行創建新的 Channel

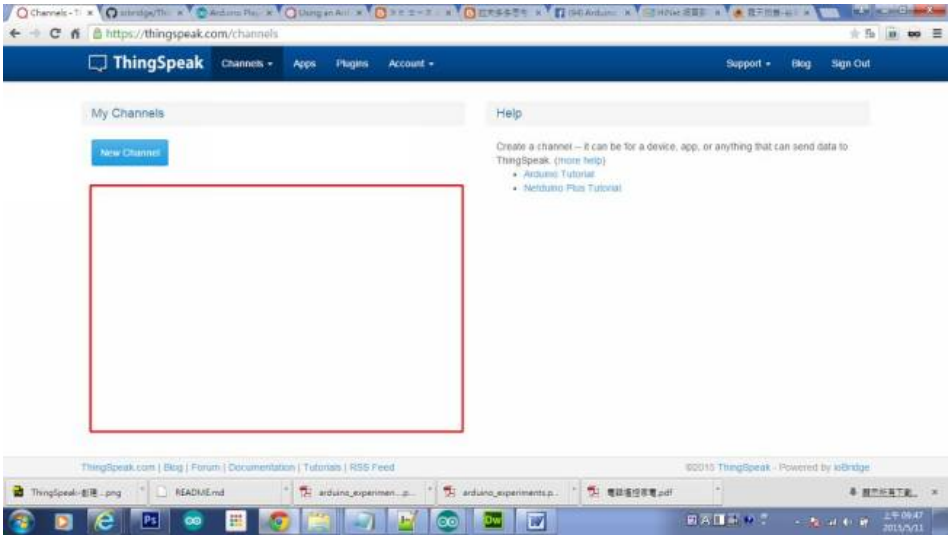


圖 25 ： 使用者尚未建立 Channel

Step 6. 點選 New Channel 選單來創建新的 Channel

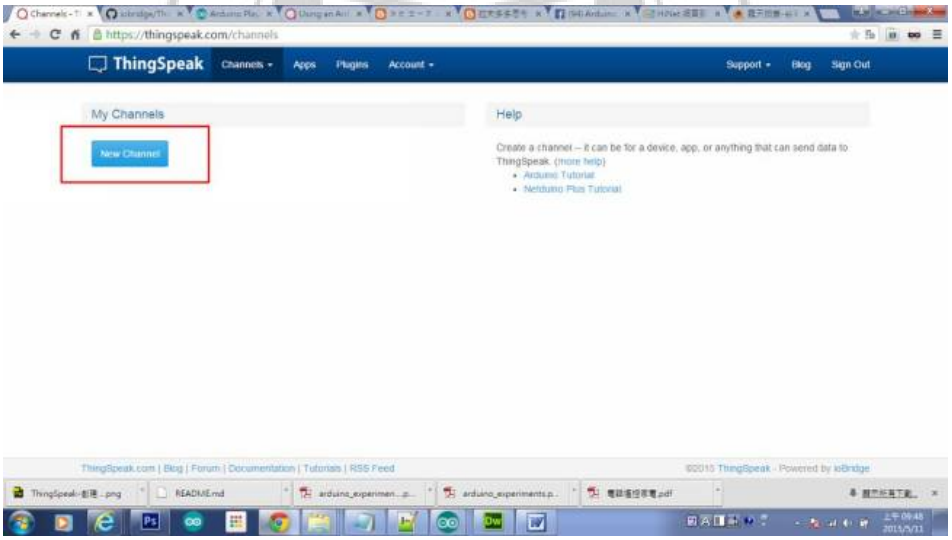


圖 26 ： 使用者建立 New Channel

Step 7. 設定 Channel，可參考如圖 27

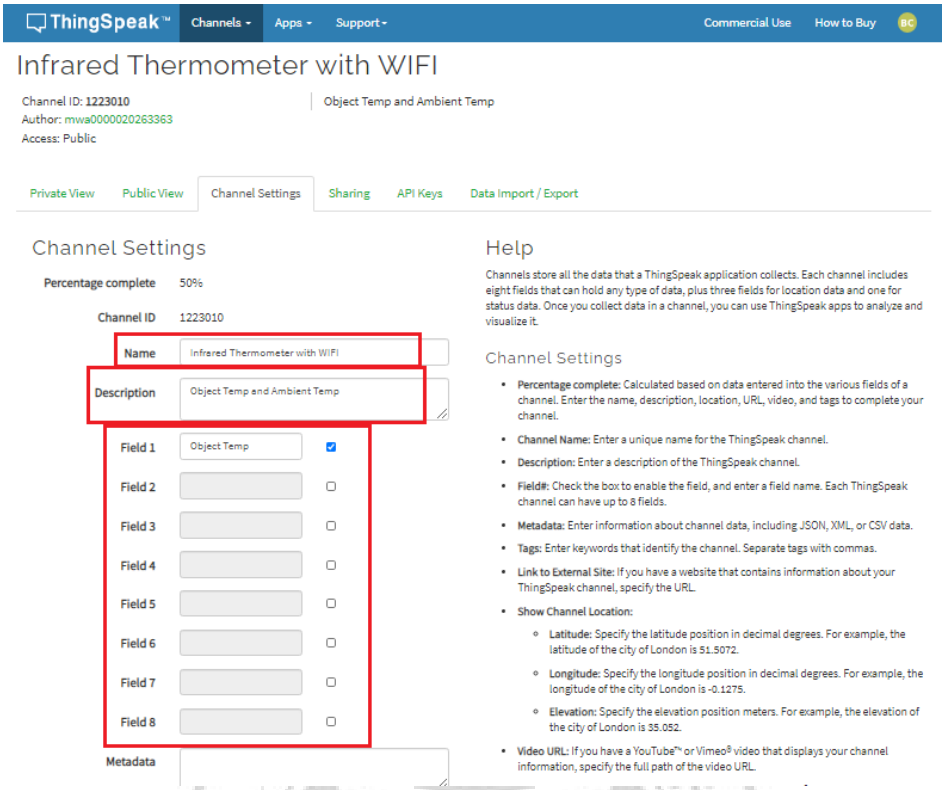


圖 27 ： 使用者設定 Channel

Step 8. Channel 設定完成後，儲存 Channel，如圖 28

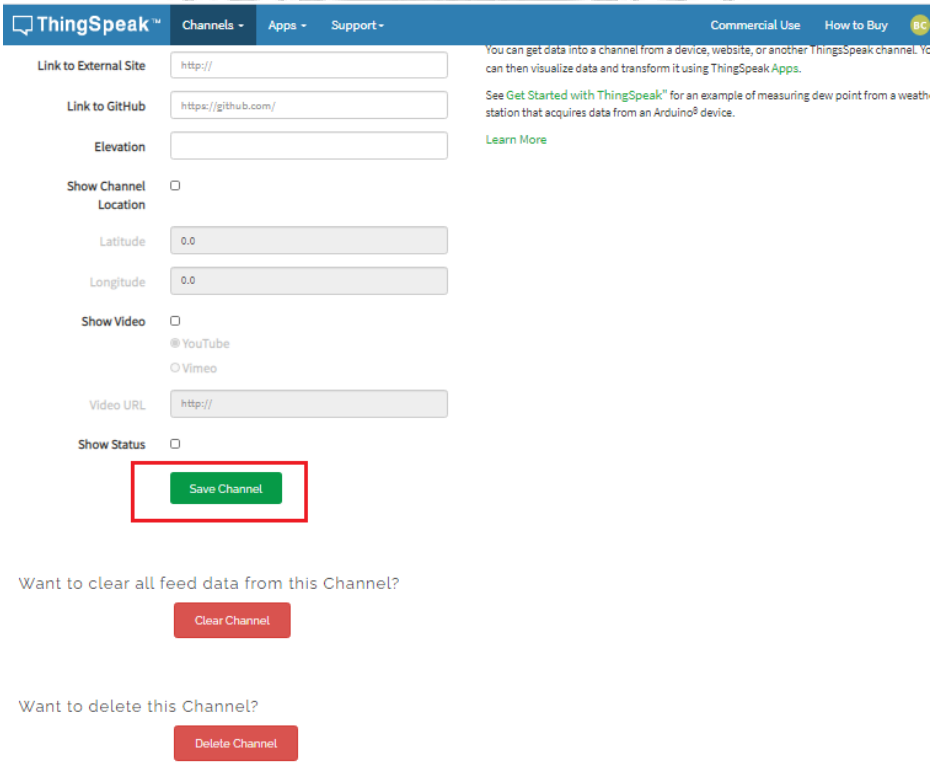


圖 28 ： 使用者儲存 Channel

Step 9. 完成 Channel 創建後，可以看到這個 Channel 的畫面，如圖 29

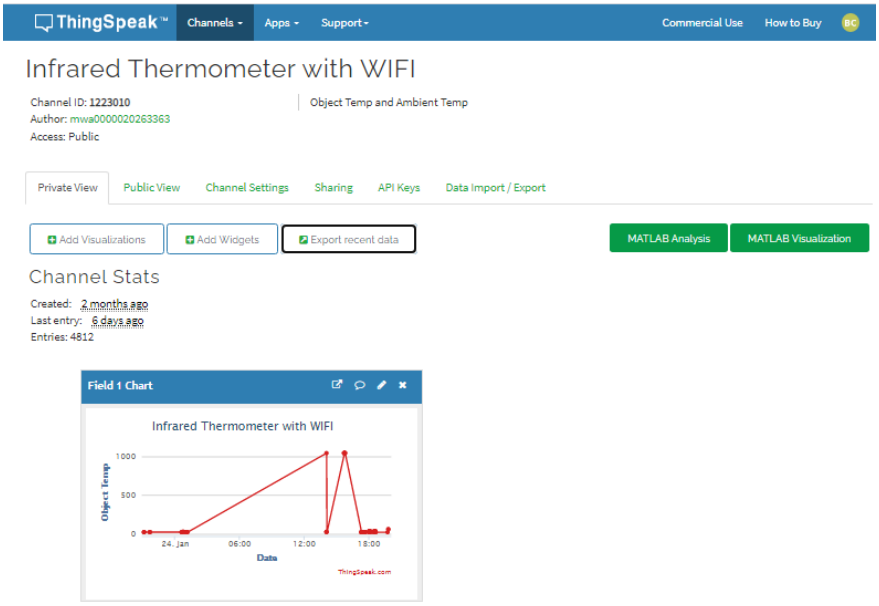


圖 29 ： Channel 完成畫面

Step 10. 使用者可將 Channel 分享給所有人/特定人/只有自己,如圖 30

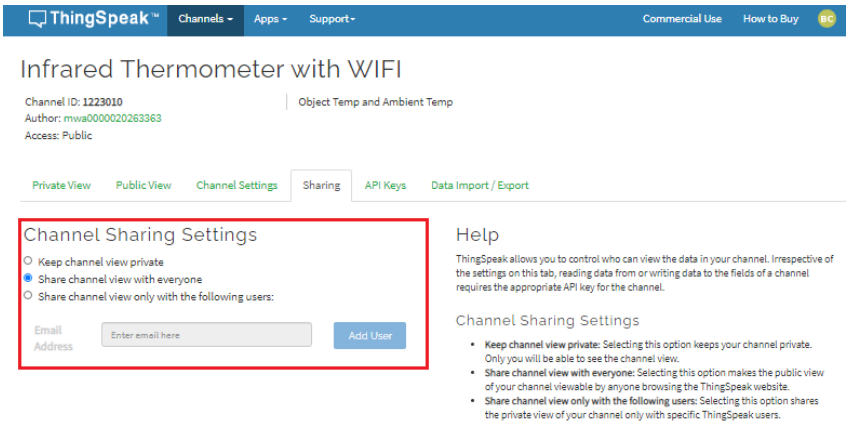


圖 30 ： 設定 Channel 分享

Step 11. 將欄位內資料的網址部份複製起來，格式如下

https://api.thingspeak.com/update?api_key=KE393NY4C7E4Y3UH&field1=0

其中 api_key＝KE393NY4C7E4Y3UH.....的亂碼是我們寫入該資料庫的密碼，避免閒雜人等亂更新我們的資料，這裡請使用您自己的 API Key 值，而 field1 則是要傳入資料庫的欄位，中間有「＆」符號是作為欄位串接用途，每一個變數之間必須有＆符號做區隔，此次內容我們要修改一個地方

ESP32 讀到的溫度值會每隔5秒鐘上傳一次到下面網頁

https://thingspeak.com/channels/1223010/charts/1?bgcolor=%23ffffff&color=%23d62020&dynamic=true&results=60&type=line&update=05

3-4 Line Notify

3.4.1 簡介

LINE Notify 顧名思義就是通知屬性的服務，LINE Notify 是 LINE 在 2016 年推出的一個服務，只要進行登錄，就可以透過 LINE Notify 官方帳號來幫您推播通知訊息。

LINE Notify 是一種具有特定功能的 API，簡化了將訊息發送到 LINE 的流程。是使用 CURL 發送訊息，透過 LINE Notify 產生你自己的「personal access token」，你便能夠藉此來發送一個 HTTP POST 請求到 API 端點。在此任何可發送 HTTP 請求的方式皆可使用。

當使用者按下「發行權杖（Generate token）」按鈕時，會跳出一個設定畫面。在此畫面你可設定你的 Token 名稱，並指定要接收通知的目標。

簡單來說就是獲取 Line 群組 token，之後就可以廣播訊息到該群組。對於行銷或是訊息傳達是極大得方便，還可以配合 Line Bot 讓使用者客製化功能再由 Line Notify 推播達成功能串接。

3.4.2 技術原理

接著下面步驟會說明如何申請 LINE Notify 並將申請的 LINE Notify 放在 ESP32 程式碼裡面並與手機 Line 作連結。

Step 1. 申請 Line Notify 權杖

Line Notify 的網站申請開通服務，首先在瀏覽器中輸入網址：https://notify-bot.line.me/，並點選右上角的登入。

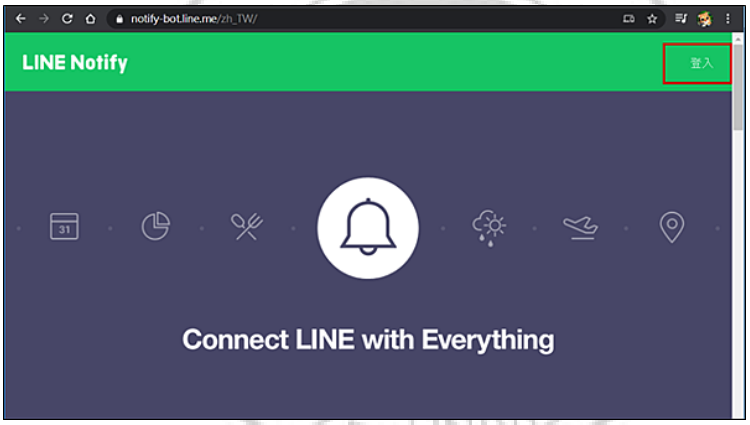


圖 31： Line Notify 登入主頁面

Step 2. Line 帳號密碼後，按下方的登入

圖 32 ： Line Notify 登入頁面

Step 3. 完成登入，點選右上角的登入帳號/個人頁面，即可進入「已連動服務」管理頁面

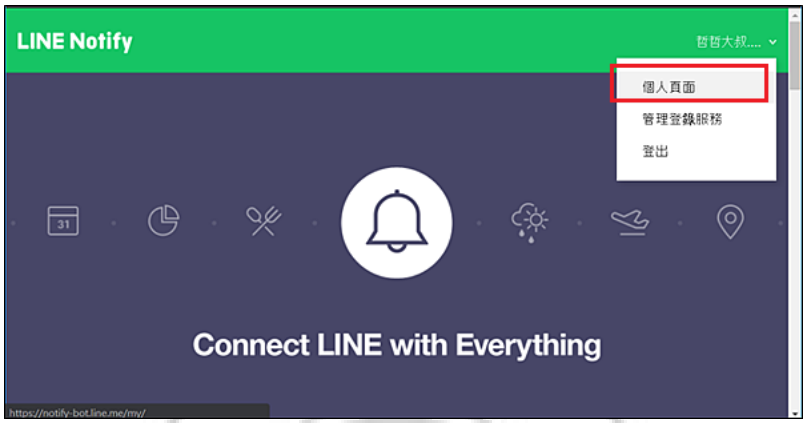


圖 33 ： 登入帳號/個人頁面

Step 4. 發行 Line Notify 權杖。

圖 34 ： 發行 Line Notify 權杖

Step 5. 在設定權杖頁面分別輸入

1. 名稱：未來發訊訊息通知時，會出現的名稱

2. 對象：要發送訊息的群組，此處練習時先選擇「透過 1 對 1

聊天接收 LINE Notify 通知」，也就是傳訊給自己，另外這裡你會發現在清單中找不到你某個特定的朋友，而只能選「群組」來發送訊息，這是因為我們目前使用的 Line 通知是免費的 Notify 功能，若您要通知特定的「人」則必須申請 Line Bot，這部份較為複雜，本研究先略過。完成輸入後，按下方的發行即可獲得一組密碼，此即為權杖。

圖 35 ： 設定 Line Notify 權杖

Step 6. 拿到權杖後，點選下方的複製按鈕將密碼複製起來，並貼在記事本上，避免遺失，若遺失也不用緊張，重新申請一次即可，不過就算申請的對象是同一個，每次的權杖密碼都是

不相同的，但功能是一樣的，後面申請的也不會使之前申請的權杖失效。

圖 36 ： 發行 Line Notify 權杖

Step 7. 確認申請完成，將會產生一個連動服務。

Step 8. 將申請到的 Line Notify 上傳到 API 測試網址 API Tester，確認 Line Notify 是否正常，網址是 https://apitester.com/ 並依照規定輸入 Line Notify 的相關設定如下圖。

1. 修改傳遞方式 POST
2. 輸入 Line Notify API 網站：
   https://notify-api.line.me/api/notify
3. 輸入傳遞的訊息內容：message=這是測試

4. 完成設定後後，按下下方藍色 Test 按鈕，即可在手機收到 LINE 傳來訊息。

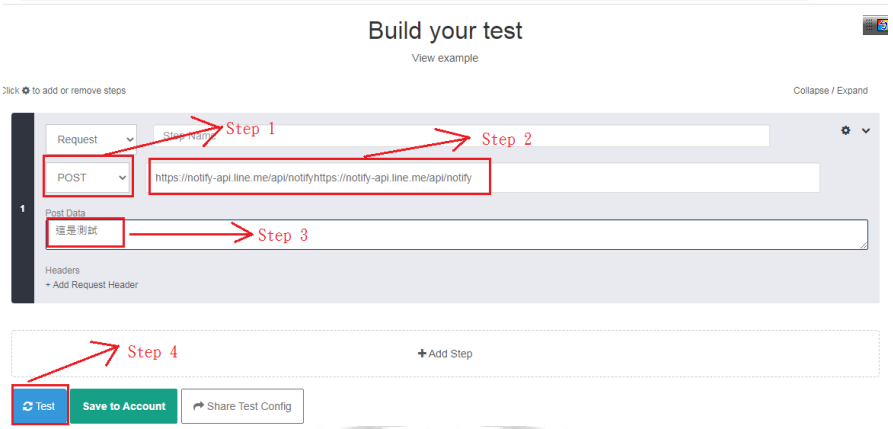


圖 37 ： API 測試網址確認 Line Notify 是否正常

5. API 測試網址回傳 Line Notify 測試結果

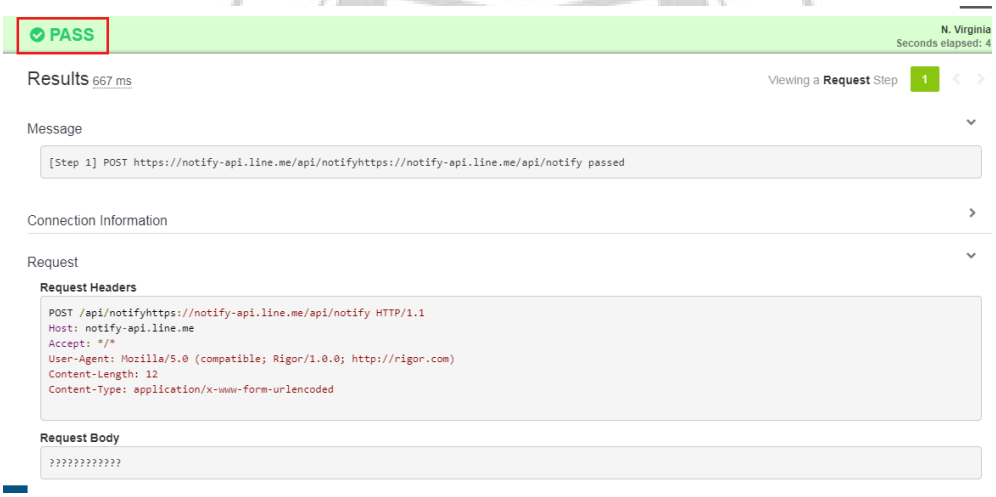


圖 38 ： API 測試網址回傳 Line Notify 測試結果

經由上面的測試可以發現，用 ESP32 傳 LINE 的方式就是將資料傳遞到 LINE 的 API 網站 https://notify-api.line.me/api/notify，並將訊息內容及權杖密碼以參數方

式夾帶給網站，網站收到後就會將訊息傳遞給權杖所對應的使用者。

第四章 紅外線感測器 MLX90614

4-1 簡介

溫度測量可分為接觸式和非接觸式，接觸式測溫只能測量被測物體與測溫感測器達到熱平衡後的溫度，所以回應時間長，且極易受環境溫度的影響；而紅外線測溫是非接觸式且根據被測物體的紅外輻射能量來確定物體的溫度，不與被測物體接觸，不影響被測物體溫度場，並且溫度解析度高、回應速度快、穩定性好等特點。近年來，非接觸紅外測溫在醫療，環境監測、家庭自動化、汽車電子、航空和軍事上得到越來越廣泛的應用。

DFRobot 最新推出 Melexis MLX90614 紅外線測溫模組-SEN0206，通過探測物體紅外輻射能量的大小和波長的分佈來檢測物體的表面溫度。紅外測溫器由光學系統、光電探測器、信號放大器和信號處理及輸出等部分組成。

光學系統彙聚其視場內的目標紅外輻射能量，視場的大小由測溫儀的光學零件及其位置確定。

紅外線能量聚焦在光電探測器上並轉變為相應的電信號。該信號經過放大器和信號處理電路，並按照儀器內的演算法和目標發射率校

正後轉變為被測目標的溫度值。MLX90614 出廠自帶校準，並且在信號調節晶片中使用了先進的低噪音放大器，一枚 17-bit ADC 以及功能強大的 DSP 元件，從而實現高精度溫度測量。

4-2 技術原理

1. Melexis MLX90614 紅外線測溫模組-SEN0206 特徵如下

- Factory calibrated IR thermometer with linear digital output
- Small size TO-39 can, easy to integrate
- Low cost, competitive prices
- Standard calibration in wide temperature range: -40 to 125 °C for ambient temperature -70 to 380 °C for object temperature
- Better than 0.5 °C accuracy in the range 0-50 °C
- 0.02 °C readout resolution possible
- High refresh rate • Easy emissivity correction
- Continuous temperature readout through PWM (Pulse Width Modulated) output

・ 2-wire SMBus/I2C compatible interface for reading temperatures and sensor reconfiguration

・ Building block for sensor network with up to 100 thermometers

・ High reliability and long-term stability

・ Excellent ESD/EMC characteristics

・ Available for 3 and 5V applications, Easy to adapt for voltage sources in range 6-24V

・ Power saving mode for battery operation

・ Traceability through unique ID number in non-volatile memory

・ Single or Dual zone thermometer version

・ RoHS compliant

2. 目前 Melexis MLX90614 普遍應用在下面場景

・ High precision contact-less temperature measurement

・ Thermal comfort sensor for Mobile Air Conditioning control system

・ Windshield defogging

- Automotive blind angle detection
- Temperature sensing element for residential, commercial and industrial building air conditioning
- Presence detector
- Fire/heat detection, alarm systems;
- Gas sensing
- Mobile telephones
- Industrial temperature control;
- Temperature control in laser printers and copiers
- Home appliances: microwave ovens, cooking stove, heater, hair dryer
- Healthcare
- Sensor grid for multi-zone temperature control
- Thermal relay/alert

3. MLX90614 本身可以量測 Object Temperature (待測物溫度)與 Ambient Temperatire(環境溫度), 支援量測攝氏與華氏溫度，量測誤差為±0.5C，將讀到溫度值透過 I2C 回傳到 ESP32。

4. 要將 MLX90614 Library 新增到 Ardunio，可透過下面 2 種方式將 MLX90614 Library 新增到 Ardunio，分別是”加入 ZIP 程式庫” 與“管理程式庫”

1. 加入 ZIP 程式庫 ： 匯入 MLX90614 Library ZIP file 到 Arduino
2. 管理程式庫 ： 搜尋 MLX90614 Library，然後接著安裝 Library

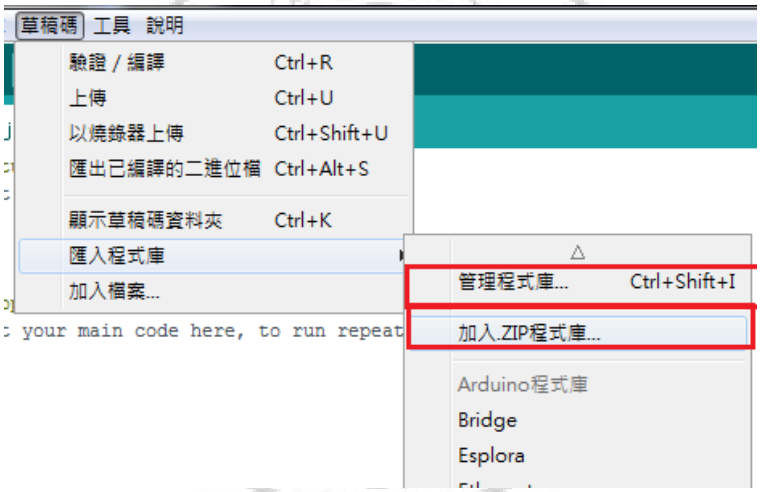


圖 39 ： 如何新增 MLX90614 Library

5.將 MLX90614 程式碼加入 Ardunio 裡面，可依照下面步驟

Step 1. 宣告 MLX90614 Library

```
#include <Adafruit_MLX90614.h>
```

Step 2. 初始化 MLX90614 _MLX90614::begin 在 setup() 裡面

```
void setup()
```

```
      {
          Wire.begin(21,22,100000); //I2C pin: SCL pin=22,
SDA pin=21, I2C frequency=100K hz)
          Serial.begin(115200);
          Objtemp1 = threshold1 +
mlx.readTemp(MLX90614_TOBJ1);
          // This mode is for surface mode
          Objtemp2 = threshold2 +
mlx.readTemp(MLX90614_TOBJ1);
// This mode is for body mode
  init_display();
  Serial.println("MLX90614 infra-red temperature sensor
test");                                  //Initalation IR sensor-
MLX90614
  mlx.begin();
//Initalation IR sensor-MLX90614
       }
}
```

Step 3.IR Temperature Sensor-MLX90614 透過 I2C 跟 ESP32 溝通。

用傳送/接收 I2C 方法從 MLX90614 讀取溫度並回傳到 ESP32

本專案設計二種量測溫度模式分別是物體表面溫度 與 人體表面溫度

A.首先介紹如何量測物體表面溫度並回傳到 ESP32

A-1. 初始化 MLX90614 I2C

```
mlx.begin();
```

A-2. 傳送 I2C, MLX90614 I2C 位址是 0x5A

```
Wire.beginTransmission(0x5A);
```

A-3. Data 寫入到 MLX90614 RAM Address-0x07

```
Wire.write(0x07);這個 address 是用來儲存量測到的表面溫度
```

A-4. 停止傳送 I2C Data

```
Wire.endTransmission(false);
```

A-5. ESP32 向 MLX90614 請求 3 個 Byte 資料

```
Wire.requestFrom(0x5A, 3);
```

A-6. 將 MLX90614 量到溫度回傳到 ESP32 與同時輸出到 OLED Display

```
result = Wire.read();
result |= Wire.read() << 8;
```

A-7. 每次 MLX90614 量到數值不能直接當做實際量測溫度，需要透過下面公式換算，並將換算過數值回傳到 ES32 與同步顯示在 OLED

Display 溫度換算公式如下

temp = result*0.02-273.15;

Note:

1. 0.02 : 是 MLX90614 計算溫度 resolution
2. 273.15 : 是絕對溫度

Exampl 1. 假設實際量測溫度=26.5

Temp=(Result *0.02)-273.15 →(Result * 0.02)=273.15+26.5

Result=(273.15+26.5) / 0.02= 14983 (0x3A87H)

Example 2. 假設實際量測溫度=21.8

Temp=(Result *0.02)-273.15 →(Result * 0.02)=273.15+21.8

Result=(273.15+21.8) / 0.02= 14748 (0x399CH)

B.接著介紹如何量測人體表面溫度並回傳到 ESP32

B-1. 同 A-1

B-2. 同 A-2

B-3. 同 A-3

B-4. 同 A-4

B-5. 同 A-5

B-6. 同 A-6

B-7. 每次 MLX90614 量到數值不能直接當做實際量測溫度，需要透過下面公式換算，並將換算過數值回傳到 ES32 與同步顯示在 OLED

Display 溫度換算公式如下

temp = result*0.02-273.15;

Objtemp2 = temp + threshold;

Note:

1.Threshold = 人體量測溫度 Offset

經過與市面額溫槍做校正，發現 ESP32 實際量測值會差 4 度 C。所以需要將實際量測到溫度在加 4 度

C 就會是人體表面溫度

2. 0.02 : 是 MLX90614 計算溫度 resolution

3. 273.15 : 是絕對溫度

Exampl 1. 假設實際量測溫度=36.6

首先，36.6- offset 4C = 32.6C

Temp=(Result *0.02)-273.15 →(Result *

0.02)=273.15+32.6

Result=(273.15+32.6) / 0.02= 15288 (0x3BB8H)

第五章 OLED Display

5-1 簡介

OLED 是 Organic Light-emitting Diode，意思是「有機發光二極體」，它指的是 OLED 顯示裝置中，每個 pixel 所使用的技術；LCD 則是 liquid-crystal display，也就是「液晶顯示器」的意思，它的「D」是 display 顯示器的意思。OLED 顯示器的每一個 pixel 裡都有一個小小的發光二極體，當你讓它流過足夠大的電流時，它就會發光，發光的顏色和二極體的材料有關，而發光的亮度則與流過它的電流大小有關。

SSD1306 是香港晶門科技（Solomon Systech）所設計的小型 OLED 顯示器驅動控制器，它可以控制最大 128x64 個 pixel 的單色 OLED 面板，內建產生 OLED 所需驅動電壓的 charge pump 電路，以及內建記憶顯示狀態的 frame buffer 記憶體。它有傳統的 6800-like CPU memory bus 界面，也有 I^2C（以下寫作 I2C） 和 SPI 界面。SSD1306 系統架構如圖

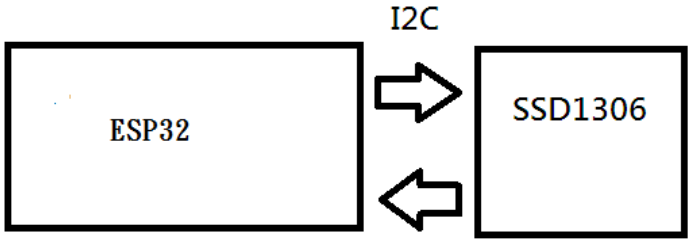


圖 40 ： SSD1306 系統架構

接著進行 Config 檔設定，路徑為 arduino-nightly\libraries\Adafruit_SSD1306-master\Adafruit_SSD1306.h，可設定解析度 128*64 or 128*32

5-2 技術原理

只要在這 SSD1306 and ESP32 之間連接 4 條線，分別是 VCC=3.3V、GND、SCL、SDA。接下來會示範使用 Arduino library 來驅動這個 OLED 顯示器，並將 MLX90614 讀到溫度透過 I2C 同時顯示在 OLED。

我們先到 Arduino IDE Library Manager 中搜尋並新增這兩個 library：「Adadruit_SSD1306」以及「Adafruit_GFX_Library」。熟悉 Arduino 的讀者對 Adafruit 這家公司應該不會太陌生，這是一間位於紐約的開源硬體公司，供應各種 maker 需要的模組、電子零件，也提供許多開發工具和開源的 library，我們這次使用的這兩個 library 就是由 Adafruit 提供的。

圖 41 ： Arduino SSD1306Library

Adadruit_SSD1306 是用來驅動 SSD1306 的底層 library，它支援 I2C 和 SPI 界面與 SSD1306 通訊，並提供了控制 SSD1306 的基本功能如 register 和 frame buffer 讀寫等。Adafruit_GFX_Library

則是在顯示裝置上繪圖的核心 library，它提供了如顯示文字、劃線、幾何圖形等功能。

當我們建立 OLED 顯示器物件時，需要傳 5 個參數給它，分別是：顯示器的水平方向點數、垂直方向點數、指向 I2C 通訊界面的指標、reset 接腳的 GPIO 編號與 SSD1306 通訊時的 I2C bus 速度以及與 SSD1306 通訊完了之後的 I2C bus 速度。要在程式碼中 include 所有需要 library：

```
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
```

除上面介紹的兩個 library 外，我們還需要一個叫 Wire 的 library，這是 Adruino 標準的 I2C 通訊 library，不過我們不會直接使用它，當我們告訴 Adafruit_SSD1306 我們要用 I2C 當作與 SSD1306 通訊的界面之後，Adafruit_SSd1306 會自動去呼叫 Wire 相關的功能。

接下來，在 setup() 中要執行的第一個指令，就是偵測 OLED 是否有接觸不良？程式碼如下

```
// Detect OLED installtion ok or not and Address 0x3C for
```

128x32

```
    if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C))
     {
       Serial.println(F("SSD1306 allocation failed"));
       for(;;); // Don't proceed, loop forever
      }
```

若上述程式沒有發現OLED接觸不良，就會開始做SSD1306 初始化，程式碼如下。

```
void init_display()
{
  display.clearDisplay();
  display.setRotation(2);    //OLED顯示轉向，可自行決定方向
  display.setTextSize(2);    // 設定文字大小
  display.setTextColor(1); // 1:OLED預設的顏色(這個會依該OLED的顏色來決定)
  display.setCursor(0,0);         // Start at top-left corner
  display.print("YZU_Brian");  //OLED 顯示" YZU_Brian"
  display.display();            //將內容更新到OLED
```

```
  delay(1000);
}
```

最後將 MLX90614 讀到溫度，透過” display.print();”將實際讀取溫度同步顯示在 OLD 上面，程式碼如下。

```
    Wire.beginTransmission(0x5A);  //Start to send MLX90614
     I2C address-0x5A
     Wire.write(0x07); // Write I2C data to MLX90614 address-
     0x07 (Ogject Temperature)
     Wire.endTransmission(false);   // I2C stop transmission
     Wire.requestFrom(0x5A, 3);
//向已知地址 slave 獲取連續 3 個數據，這時候需要注意，數據只是存起來了，並沒有真正返回
     result = Wire.read();          //Receive DATA
     result |= Wire.read() << 8;    //Receive DATA
     uint8_t pec = Wire.read();
     temp =  result*0.02-273.15;    //Measure Surface
Temperature
     Objtemp1 = temp;
     Serial.println(Objtemp1,1);
```

```
Serial.println(" *C");
display.setCursor(0,4);
display.print(Objtemp1,1);
```

第六章 線路圖

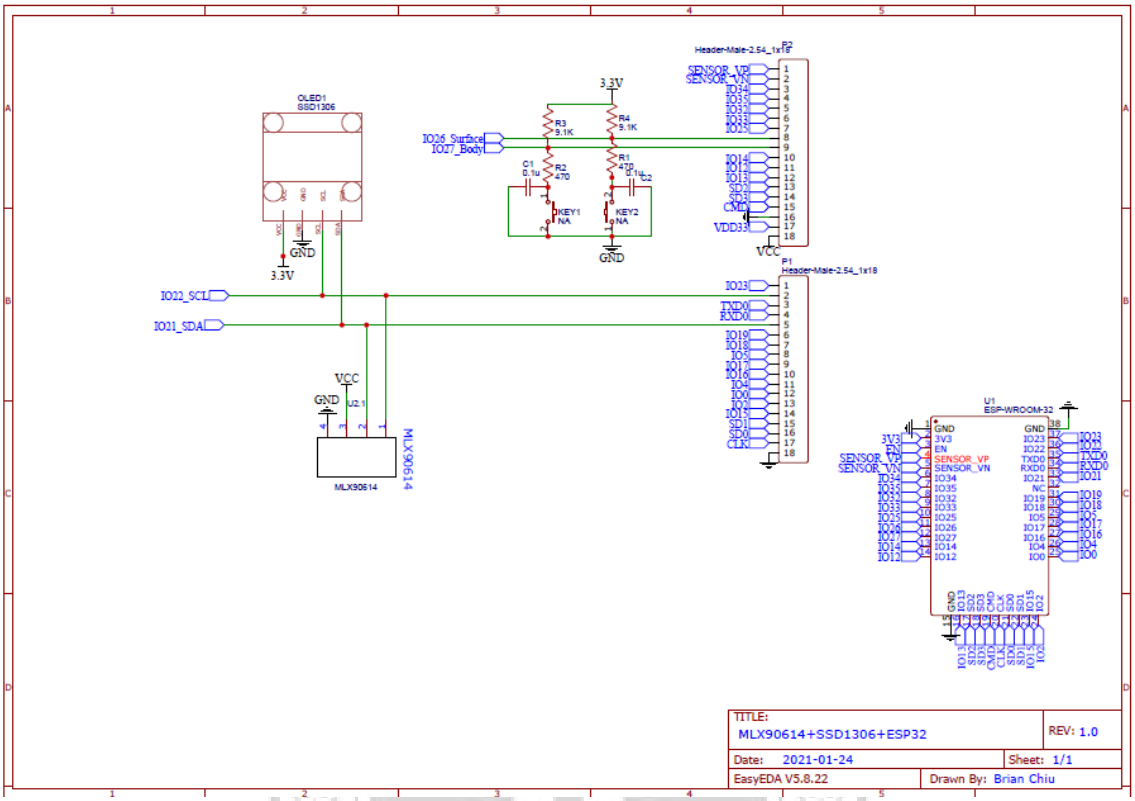


圖 42 ： 線路圖

圖 42 線路圖由下面 4 個零組件組成

1. ESP32
2. IR Sensor MLX906141 by I2C
3. 0.96 吋 OLD Diosplay（Controller is SSD1306 by I2C）
4. 按鍵 x2 顆（Port 26-物體表面溫度 ； Port 27-人體表面屋度）
5. Pull High 線路 ： 9.1 K ohm x2
6. Pull down 線路 ： 470 ohm x2 + 0.1uF x2

按鍵用來切換讀取量測物體表面溫度與人體表面溫度。讀取溫度值透過 I2C 從 MLX9014 讀出後，同時透過 I2C 顯示在 OLED Display。接著透過 WIFI 將 MLX90614 讀到溫度值上傳到 Thingspeak 上面做數據紀錄。當量測人體表面溫度值超過 30°C，量測溫度值會透過 Line Notify 傳送到手機 Line 以達到推播通知效果。當量測物體表面溫度值超過 20°C，量測溫度值會透過 Line Notify 傳送到手機 Line 以達到推播通知效果。

第七章 程式碼Flow與程式碼

1. 量測人體表面溫度程式碼Flow如圖43

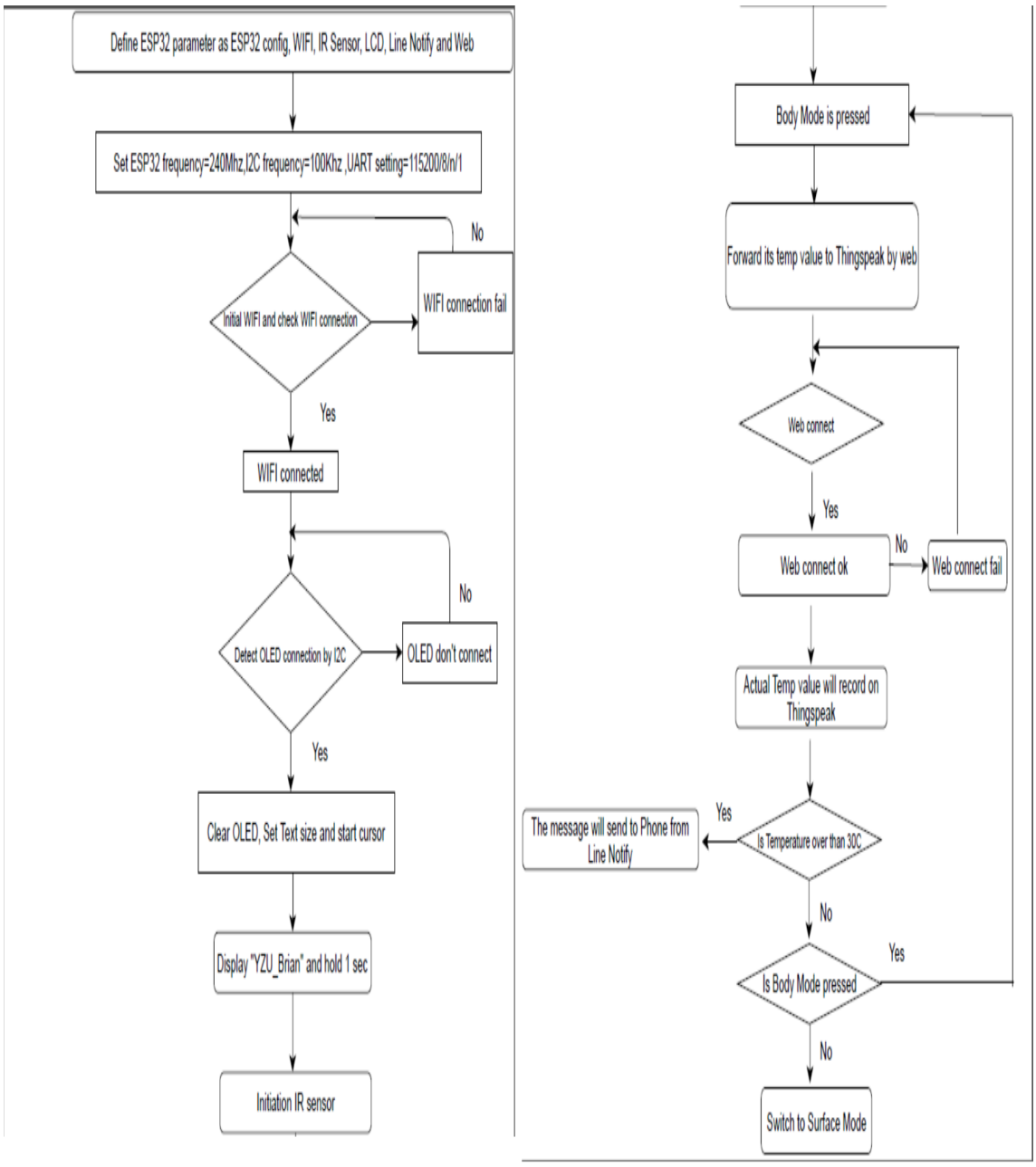


圖43： 量測人體表面溫度程式碼Flow

2.說明程式碼，如下

2-1. Include 需要執行Library

```
#include <Wire.h>
#include <WiFi.h>
#include <WiFiClient.h>
#include <WiFiClientSecure.h>
#include "Arduino.h"
#include "esp32-hal-cpu.h"
#include <Adafruit_MLX90614.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <HTTPClient.h>
```

2-2. 設定Line Notify Token

```
String Linetoken =
"vg0ZKjA0aThL3tuWUAGG73xaes7WiD9e0yFplu0K5vx";
//Your Line Token
```

2-3. 設定量測溫度相關參數

```
int threshold = 4;
//MLX90614 Calibration for Body mode (Surface + 4.0C=Body)
float temp,Objtemp1,Objtemp2;
bool setCpuFrequencyMhz(uint32_t cpu_freq_mhz);
uint16_t result;
```

2-4. 定義 OLD 解析度=128 * 32 、量測表面溫度與人體溫度 Mode

```
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 32
//Set OLED pixel=128 * 32
#define pin2 26
#define pin3 27
```

2-5. 設定網路連線物件、Line Notify Address、WIFI password

```
WiFiClientSecure client; //網路連線物件
char host[] = "notify-api.line.me"; //LINE Notify API web
address
// Your WiFi credentials.
```

```
// Set password to "" for open networks.
char ssid[] = "HITRON-6E00";  //Home WIFI SSID
char password[] = "0921294757";  //Home WIFI Password
```

2-6. 設定上傳溫度數據網頁與宣告 MLX90614

```
String url =
"https://api.thingspeak.com/update?api_key=KE393NY4C7E4Y3UH&
field1=25";          //Set Thingspeak chanel
#define OLED_RESET     4 // Reset pin # (or -1 if sharing
Arduino reset pin)                    //Set OLED
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire,
OLED_RESET);                          //Set OLED
Adafruit_MLX90614 mlx = Adafruit_MLX90614();
```

2-7.設定 I2C 接腳與 I2C Frequency=100Khz、設定 IO26=Pin 2_表面溫度、IO27=Pin 3_人體溫度，WIFI 初始化與確認 WIFI 連線有無成功、初始化 OLED SSD1306 and 初始化 MLX90614

```
void setup()
{
```

```
  setCpuFrequencyMhz(240);
//Set CPU clock to 240MHz fo example
  Wire.begin(21,22,100000);
//I2C pin: SCL pin=22, SDA pin=21, I2C frequency=50K hz
(Default is 100K hz)
  Serial.begin(115200);
  pinMode(pin2,INPUT_PULLUP);
  pinMode(pin3,INPUT_PULLUP);


  // Initial WIFI
  WiFi.begin(ssid, password);


  while (WiFi.status() != WL_CONNECTED) //Wait for
connection ok
  {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
```

```
  Serial.println("WiFi connected"); //WIFI cooection ok
  // Detect OLED installtion ok or not and Address 0x3C for
128x32
    if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C))
     {
       Serial.println(F("SSD1306 allocation failed"));
       for(;;); // Don't proceed, loop forever
      }
  init_display();
  Serial.println("MLX90614 infra-red temperature sensor
test");                          //Initalation IR sensor-
MLX90614
  mlx.begin();
}
```

2-8.一直判斷是做表面溫度量測 還是人體溫度量測。量到溫度後，每 5 秒鐘上傳讀取溫度到 Thingspeak。若量測溫度超過 30C，手機會收到 Line Notify 通知，並上傳到 Thingspeak。

```
void loop()
```

```
{
   Mode:while (digitalRead(pin2) == LOW)
        {
          Serial.println("IO26 is pressed");
          Serial.println(digitalRead(pin2));
          display.clearDisplay();
          delay(1000);
          get_surface();
          delay(500);
          unit_display();
          web_surface();
          Line_surface();

          if (digitalRead(pin2) == LOW) continue;
          if (digitalRead(pin2) == HIGH) break;
        }
          goto BODY;

BODY: while (digitalRead(pin3) == LOW)
```

```
        {
          Serial.println("IO27 is pressed");
          Serial.println(digitalRead(pin3));
          display.clearDisplay();
          delay(1000);
          get_body();
          delay(500);
          unit_display();
          web_body();
          Line_body();
          if (digitalRead(pin3) == LOW) continue;
          if (digitalRead(pin3) == HIGH) break;
        }
          goto Mode;
}
```

同理，當切換到物體表頓量測模式，量測溫度超過 20C，手機收到 Line Notify 通知，同時量測溫度會上傳到 Thingspeak，如圖 44。

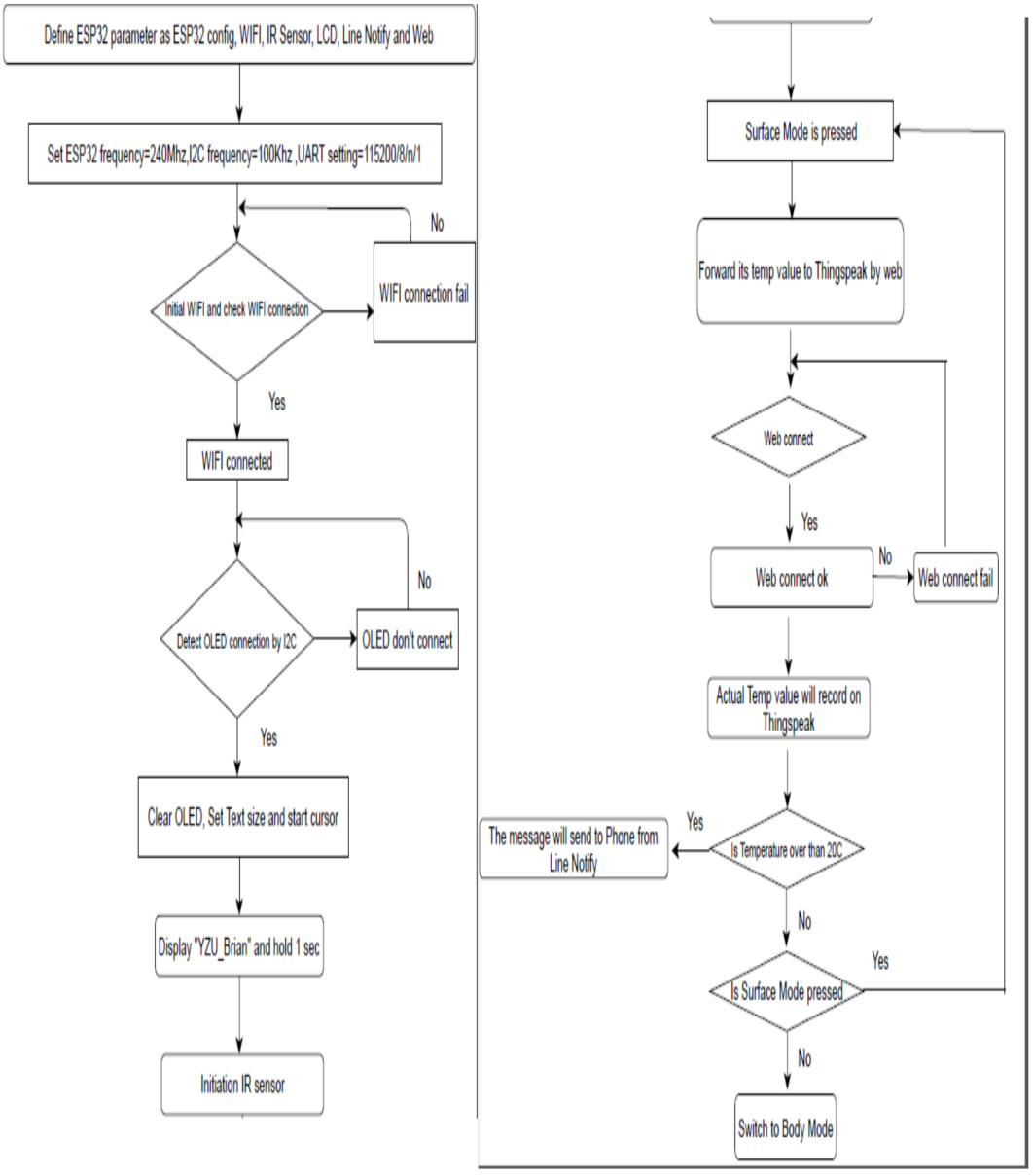


圖 44 ： 量測物體表面溫度程式碼 Flow

第八章 實驗結果

完成程式碼編譯與燒錄後，將手邊市售額溫槍型號 GP-300 與 ESP32 做 correction 驗證，得到下面 Table 驗證結果

Distance=5 cm, Unit : C	洞洞板	書桌架子牆角	NB_Enter key	牆壁	手心	額頭	抱枕	馬克杯	馬克杯裡面的水
Thermometer(Model-GP-300)	21.5	20.8	26.7	20.3	36.1	36.3	20.6	21.7	19.9
ESP32+MLX90614	21.8	21.3	26.5	20.4	35.8	36.6	20.6	21.1	19.8

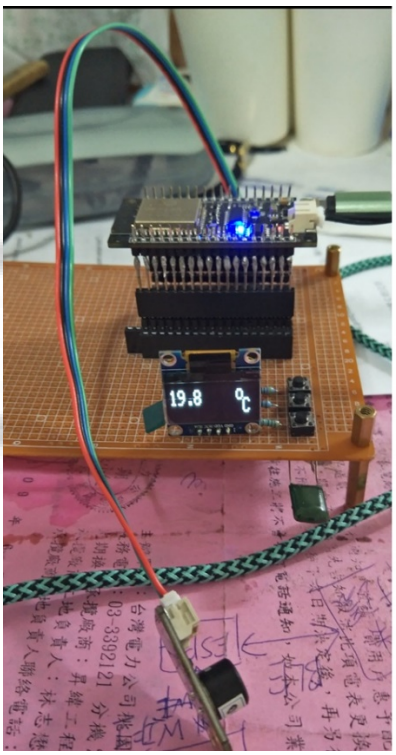


圖 45 : ESP32+MLX90614+OLED Display

當量測溫度超過 30C ESP32 會透過 Line Notify 發送通知到手機,如圖 45

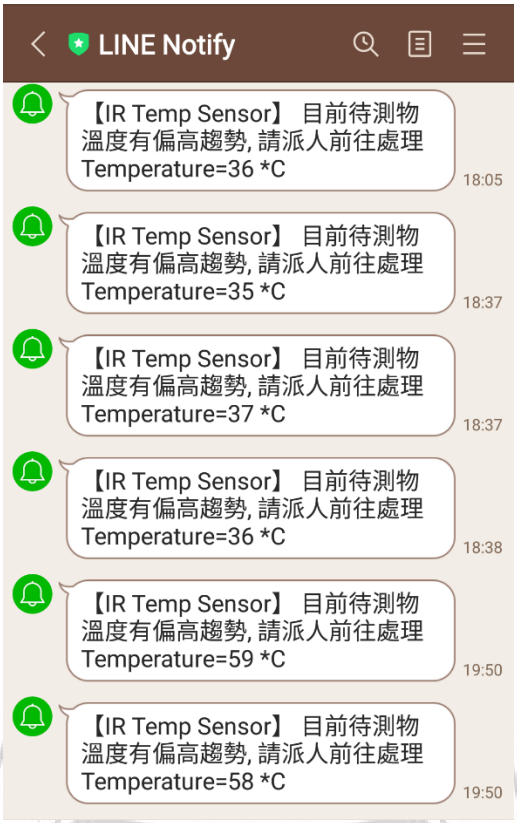
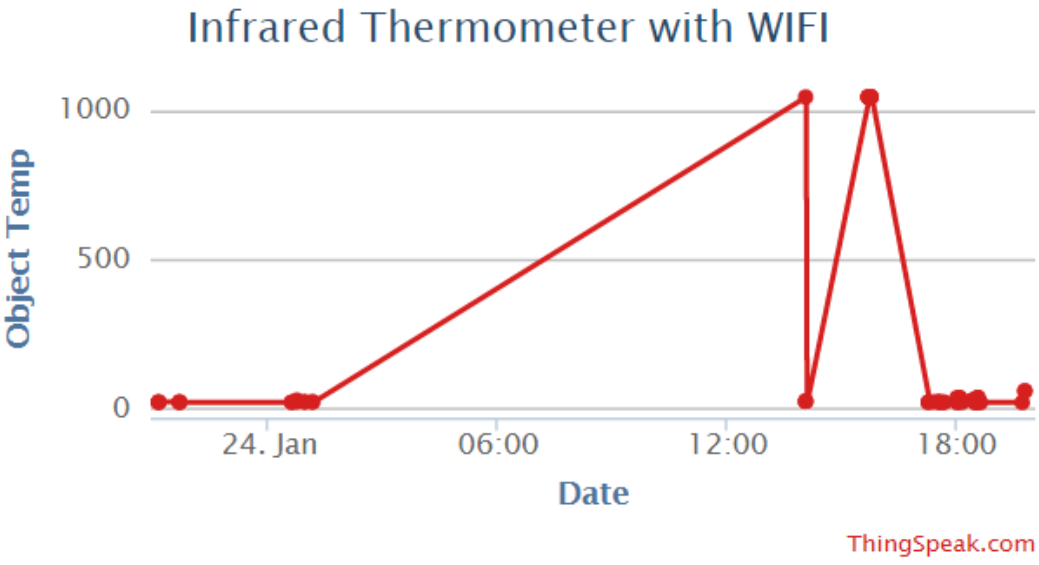


圖 46 ： Line Notify 通知

Thingspeak 上傳量測結果，如圖 46



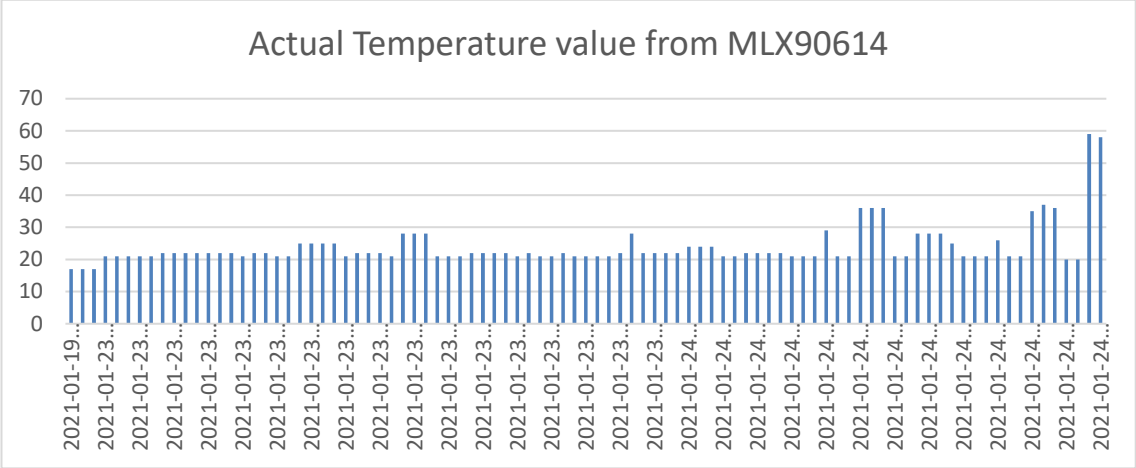


圖 47 ： 量測溫度上傳 Thingspeak

第九章結論

本專案所使用 MLX90614+ESP32 與市售額溫槍(GP-300)量測溫度差距在±0.5 度內，二者之間的待測物表面溫度與人體溫度差距並無顯著差異，透過 WIFI 上傳量測溫度數據，並將量測溫度統計成大數據分析。使用者可隨時知道溫度量測結果有無異常。同時也可免除病人因接觸式量測而感染的機會。

此研究專案未來展望: 現有系統架構還可以在延伸出將現有實體按鍵移轉到智慧型手機上面，變成透過手機 APP- Thunkable 控制量測物體表面溫度或者量測人體表面溫度，並將紅外線 Sensor 溫度讀值透過 Bluetooth BLE function 傳到使用者手機，接著透過 WIFI 上傳到雲端服務 Thingspeak 並用 MATLAB Analysis App 做線上分析，最後將線上分析結果透過推播服務,如 Line Notify, E-Mail, Wechat 等傳送到使用者手機上。

Reference

[1] Arduino Library https://www.arduino.cc/reference/en/

[2] Arduino 快速入門教學全集

https://blog.jmaker.com.tw/arduino-tutorials/

[3] Arduino Wiki https://en.wikipedia.org/wiki/Arduino

[4] Arduino ESP32 實作範例

http://wangwangtc.blogspot.com/2017/09/arduino.html?m=1

[5] ESP32 Wiki https://en.wikipedia.org/wiki/ESP32

[6] ESP32 official web site

https://www.espressif.com/en/products/socs/esp32

[7] 人體紅外線測溫儀器（作者 : 賴永成）

http://www.irpro.com.tw/Application_02.htm

[8] 中科電子報_全民量體溫，額溫槍體溫測量技術知多少？

https://ctsphub.org.tw/news/info_more?id=6325b6f5995e4f7b861

37a34b254b1bf

[9] 產品操作手冊_紅外線測溫槍 AT8380 系列

http://atlantis.tw/upload_file/img_fckeditor/file/%E8%AA%AA%

E6%98%8E%E6%9B%B8/%E6%95%B8%E4%BD%8D%E6%BA%AB%E5%BA%A6%E8%A8

%88/%E7%B4%85%E5%A4%96%E7%B7%9A%E6%B8%AC%E6%BA%AB%E6%A7%8D%E8%AA%AA%E6%98%8E%E6%9B%B8%20AT8380.pdf

[10] DFROBOT ESP32 Model-DFR0478 SPEC

https://wiki.dfrobot.com/FireBeetle_ESP32_IOT_Microcontroller(V3.0)__Supports_Wi-Fi_&_Bluetooth__SKU__DFR0478

[11] 痞客邦 第四篇 ESP32 數位讀取(感測)digitalRead

https://youyouyou.pixnet.net/blog/post/120257245-%e7%ac%ac%e5%9b%9b%e7%af%87-esp32%e6%95%b8%e4%bd%8d%e8%ae%80%e5%8f%96%28%e6%84%9f%e6%b8%ac%29digitalread

[12] 痞客邦 第三篇 ESP32 數位輸出 digitalWrite

https://youyouyou.pixnet.net/blog/post/120253747-%e7%ac%ac%e4%b8%89%e7%af%87-esp32%e6%95%b8%e4%bd%8d%e8%bc%b8%e5%87%badigitalwrite

[13] Thingspeak Wiki

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

[14] Thingspeak 物聯網平台（作者 : 弘道國中 潘建宏）

http://web.htjh.tp.edu.tw/B4/106iot/ThingSpeak%E7%89%A9%E8%81%AF%E7%B6%B2%E5%B9%B3%E5%8F%B0.pdf、

[15] 【物聯網系統開發】THINGSPEAK 平台基本篇

https://www.techbang.com/posts/51161-internet-system-development-thingspeak-platform-review

[16] 痞客邦_第十四篇 ESP32 資料庫存取 ThingSpeak 圖表製作

https://youyouyou.pixnet.net/blog/post/120275941-%E7%AC%AC%E5%8D%81%E5%9B%9B%E7%AF%87-esp32-wifi-server%E7%B6%B2%E9%A0%81%E4%BC%BA%E6%9C%8D%E5%99%A8%28%E9%81%A0%E7%AB%AF%E6%BE%86%E8%8A%B1

[17] 自建 Line Notify 通知

https://www.oxxostudio.tw/articles/201806/line-notify.html

[18] 痞客邦_第十三篇 ESP32 LINE 通知：倉庫溫度異常機器人

https://youyouyou.pixnet.net/blog/post/120275932

[19] API Tester web site

https://apitester.com/

[20] iT 邦幫忙_如何在 Line Bot 免費推播訊息- Line Notify

https://ithelp.ithome.com.tw/articles/10229814

[21] Melexis MLX90614 feature description

https://www.sparkfun.com/datasheets/Sensors/Temperature/SEN-09570-MLX90614.pdf

[22] 紅外線溫測原理簡介

http://www.infrared.com.tw/uploadfiles/280/Catalog/News/ir-principles_thermometry_infrared.pdf

[23] Arduino 與 MLX90614 I2C 主從之間通信

https://www.twblogs.net/a/5c0ce721bd9eee5e41831684

[24] IR Thermometer MLX90614 Module-SEN0206 SPEC

https://wiki.dfrobot.com/IR_Thermometer_Sensor_MLX90614_SKU__SEN0206

[25] OLED Driver IC-SSD1306 description

https://kknews.cc/zh-tw/news/jkyb6v6.html

[26] 小型 OLED 顯示裝置原理與應用 part 1

https://makerpro.cc/2020/09/principle-and-application-about-oled-part1/

[27] 小型 OLED 顯示裝置原理與應用 part 2

https://makerpro.cc/2020/10/principle-and-application-about-oled-part2/

[28] 小型 OLED 顯示裝置原理與應用 part 4

https://makerpro.cc/2020/10/principle-and-application-about-oled-part4/