111 下數位科技專案實作

期末報告

109033132 動機系 李臻茵

2023.07.13

Contents

*Github link: https://github.com/teresasa0731/Home_system_HOST_esp32

1 Idea generation & Product definition

透過這次修課的機會，希望可以網路傳輸協定與物聯網技術有基本的了解，故選擇學習開發一套居家系統，而這次的子專案則從建立整個系統的控制中心 System Host 開始，配合基本的三個功能來架設，分別為：

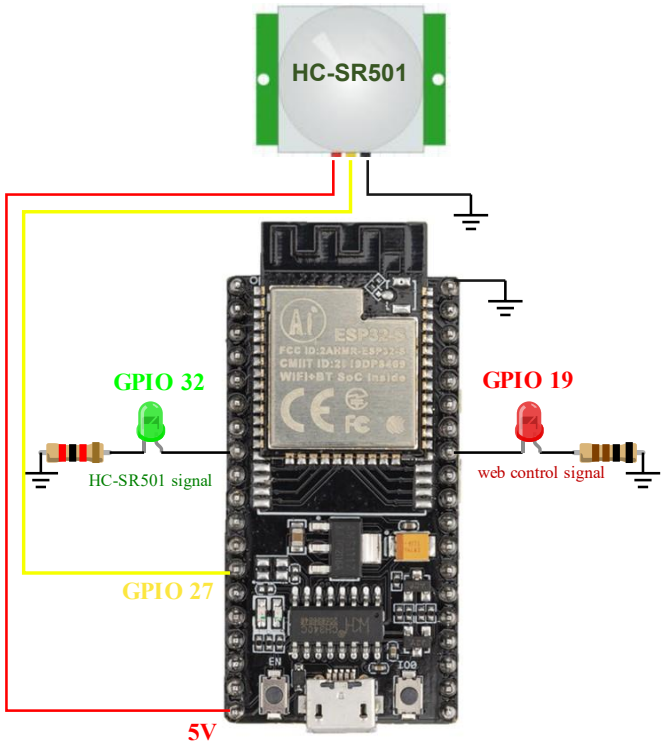
a. Control：以 LED 燈呈現對開關的控制訊號輸出。

b. Monitor：分別以 Switch Button+LED 燈呈現靜態訪問的訊號輸入；以 HC-SR501 來呈現及時狀態的偵測。

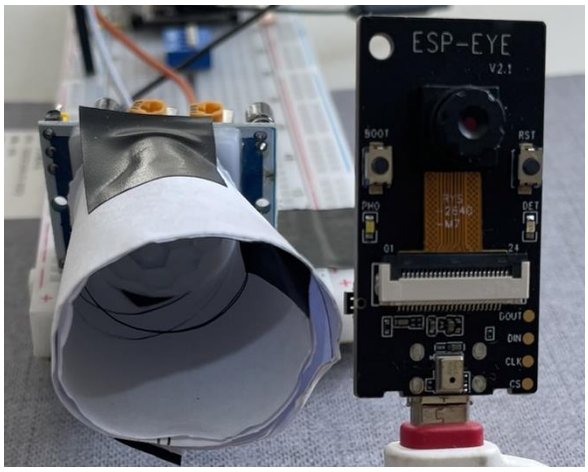
c. Live Streaming

為了將問題模型化，並將焦點放在網路的連線溝通上，以上三個功能都以最基本的元件模擬；同時為了將來能夠對系統有完整的控制，包括記憶體分配與特殊硬體模式等的設計，因此選擇直接使用 ESP-IDF (Espressif IoT Development Framework) 做為此次 ESP32 硬件開發的軟體環境。

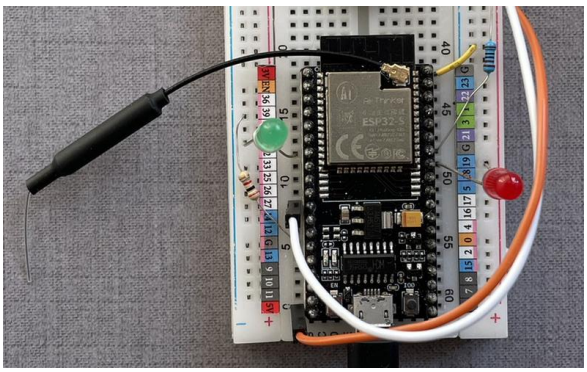
2 Hardware Structure



▲ Fig 1. Hardware circuit wiring diagram[1]



▲ Fig 2-1. HC-SR501 with mask[2]



▲ Fig 2-2. Hardware circuit wiring diagram

[1] 修課同學提及之感測模組仍在測試中，尚未能展示。

[2] 透過增加錐狀遮罩(暫時以紙捲代替)來提高人體感測模組的指向性。

3 Software technology

前提及此次專題的重點是在學習網路傳輸的理論背景與應用，故在研究過由 esp32 的開發商 espressif 所提供的編譯環境 ESP-IDF 與範例後，先由單向訪問的 web server 技術開始建立(ver.1)，完成對周邊元件訊號的邏輯判斷後，加上 HTML 文本來將其架設成網站形式，開始透過網站做訪問。此版本中遇到的幾個瓶頸：

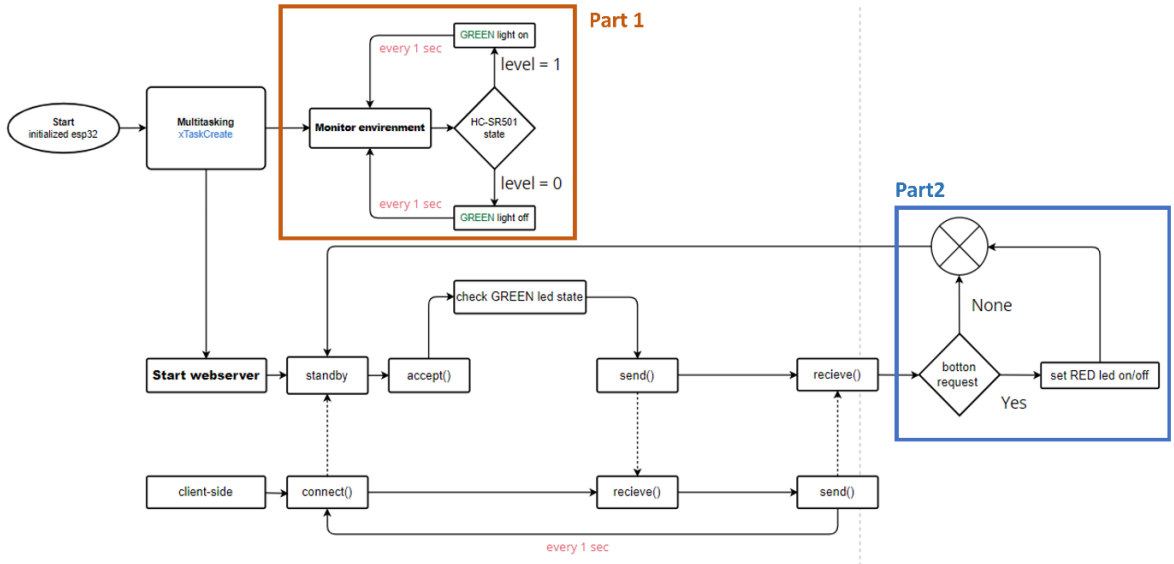
a. HTML 文本難以維護，因此階段的按鈕指令發送給伺服器是透過連接至不同位址做發送，但因網域時常更動而需更改文本內容，而此文本為內嵌於主程式中，故每次更新都須重新編譯(build)及燒錄(flash)，故在下一版加入 SPIFFS 檔案系統技術。

b. 感測模組的訊號有即時性問題，這裡的做法是每 100ms 做輪詢，其過度消耗 CPU 的資源，故二版使用 web socket 的方式架構雙向溝通的伺服器。

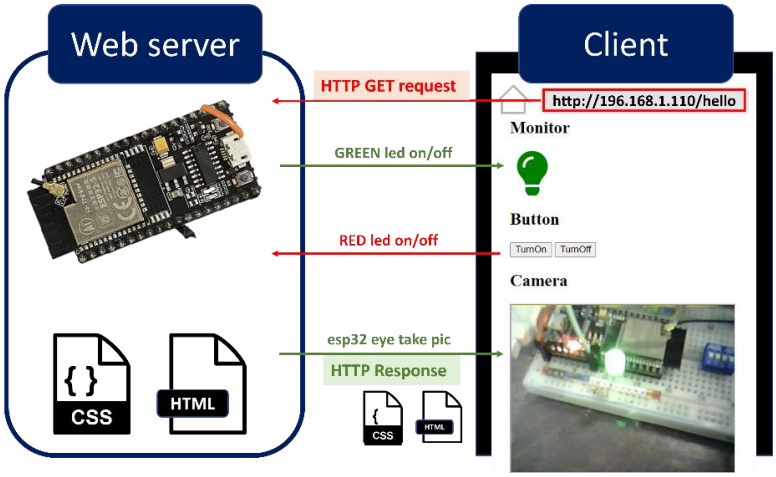
3.1 version 1

將硬體端 esp32 當 static web server，由 web(client 端)對伺服器發起 HTTP request 進行單向訪問，伺服器經由網路接受 HTTP 請求後，透過 HTTP 協定傳送回復給客戶端。

3.1.1 Architecture



▲ Fig 3-1. Program Flow chart[3]



◂ Fig 3-2. Server-Client connection

[3] 主程式碼請參照資料夾 systemHost_Ver1/main.c

3.1.2 Key Details

(a) Multitasking

因當 CPU 進入 server 狀態後可視為在主程式架設好參數後，到 function 中服務，故為了同時能接受人體感測模組的訊號，對其做 multitasking 來分割事件。

```
xTaskCreate(Demo_Task, "Demo_Task", 4096, NULL, 10, &myTaskHandle);
server = start_webserver();
```

(b) HTML 文本

網站的部分希望能對前提及的三個功能提供資訊參考：

a. 人體感測模組狀態

在硬體上以綠燈做為訊號顯示

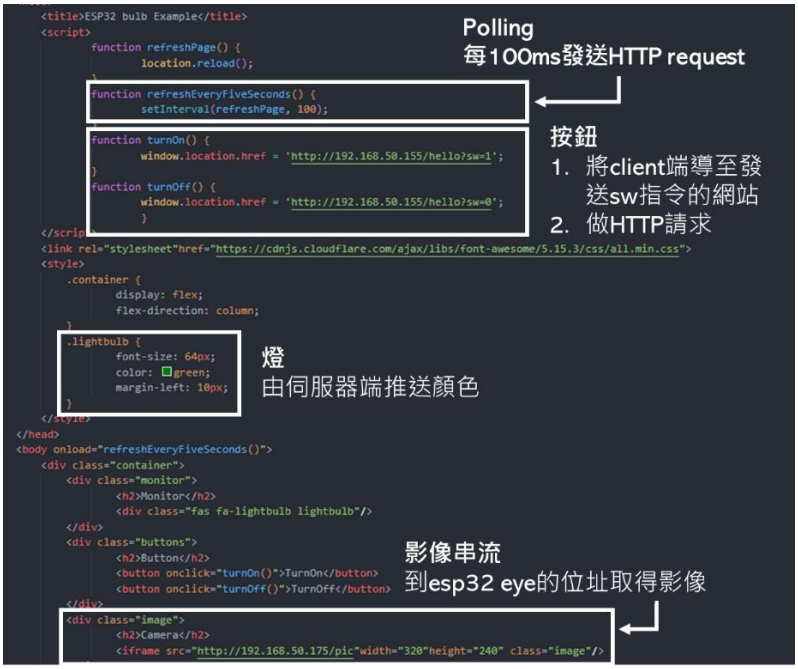
b. 透過網站按鈕控制硬體狀態

此處為控制紅燈做訊號顯示

c. 來自 esp32eye 的影像資訊[4]

目前先以照片的形式取代，因其易過熱

◂Fig 4. Website



◂ Fig 4. Ver1.html[5]

[4] esp32 eye 程式來源為官方提供之範本(https://github.com/espressif/esp-iot-solution)，可見資料夾 pc_server/

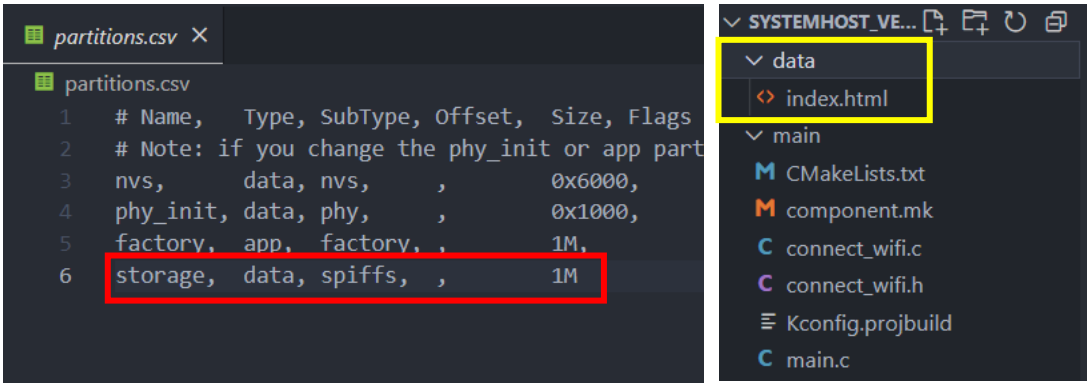
[5] 可見資料夾 systemHost_Ver1/ ver1.html

3.2 version 2

改成使用 web socket + SPIFFS 的模式，除了將發送文檔打包而非內嵌於程式碼，在對 HTML 文本進行優化時能獨立處理，也實現 client 與 server 的雙向溝通機制，可以即時的交換資訊，不用等待 client 端發出請求，或使用佔用資源的輪詢方式來實現，同時因 Handshake 後會保持連線狀態，所以每次通訊時所發出的請求就可省略掉一些關於連線狀態的資訊，提高連線效率。

3.2.1 Key Details

(a) SPIFFS



可以看到在 partition table 中，相較於舊版多要了一塊 flash 的空間作為 SPIFFS 的儲存位置(紅框處)，就可將推送資訊放於額外的空間，在實作時也不需重新編譯，只需燒錄更動的檔案即可(以此次專案而言即可以單獨優化 HTML 文本)。

(b) web socket vs web server

在功能 2(monitor)中，舊版的觸發方式來自於 client 端頻繁的發送請求，同時因監測的功能是分割在 task2(前所提及的 multitasking)，故當 client 發送請求時，其實並非感測器當下的狀態，加上此次選擇的感測器訊號並非非常穩定，故在此部份是第一版比較缺失的地方；但透過 web socket 的方式，可以透過訊號觸發來讓伺服器端向用戶端推送資訊，且舊版若是在不同平台上操作，儘管接收同一伺服器的資訊，在未更新網頁前是不會收到最新的狀態的，改成用 web socket 後就可以同步刷新網站資訊。

3.2.2 Incomplete Verification[6]

因功能 2 的推送觸發仍在修正，故舊第二版預期能達到的目標而言，剩伺服器端在某些條件下(此專案即為當感測器觸發時要改變網頁端綠燈的狀態)主動發送資訊此功能未完善，其他諸如獨立 HTML 文本、建立 web socket 連線等皆已完成。

[6] 此版本因 web socket 的程式推送仍在修正，目前 1.5 版本較為完善(可見資料夾 systemHost_Ver1.5/)

4 Future work

(a) 完善網頁架構

考量到未來欲監控的電器，須對網頁做動態調整。

目前還無法將 esp32 eye 的影像同步在 web socket 的網頁上更新，僅 ver.1 能以 polling 詢問 esp32 eye 伺服器的影像，故仍需再研究。

(b) 硬件方面修正與設計

完成 ver.2 後即可開始時做硬體的部分，大致如如何將市電控制與開發板結合、設計相應應用場景的模式等。

(c) 影像處理

學習影像辨識，設計並取得所需資訊(ex.人體偵測/環境辨識等)，與居家系統的應用情境結合