中華民國第 54 屆中小學科學展覽會

作品說明書

高職組　電子、電機及資訊科

最佳團隊合作獎

091010

創新架構之家庭自動化研製

學校名稱：國立旗山高級農工職業學校

作者：	指導老師：
職二 吳海若 職二 巫修全 職二 胡暄禾	林勝雄

關鍵詞：家庭自動化 、Raspberry PI、Arduino

作品名稱：創新架構之家庭自動化研製

摘要

家庭自動化的定義：利用各項電子零件系統或是可以程式化，協助人類處理家中事物時可以毋須直接接觸或是做出反應的一種機制。

以往製作此一系統必須花費不少精神在做系統整合，一部分是要做單晶片及連結一個伺服器去達成所要的功能，或者是利用單晶片去實踐整個系統，前者較為容易但是必須要建置一個體積較大的伺服器，且維護費用較高。後者體積較小但是程式碼撰寫則相對困難。

本系統是用 Raspberry PI 試圖完成家庭自動化的功能：偵測輸入信號(防盜系統)、控制輸出信號(用以模擬燈泡)，且可以用 PC 或是手機上網，便可以瞭解家庭中之各項資料。

壹、研究動機

於計算機概論課程中，討論有關電腦作業系統的分類問題，與指導老師討論此一議題，以往需要架設一台伺服器去實現此一功能，相對的所需成本較高。家庭自動化一直以來都是人們追求的夢想，舉凡簡單的家電監控，或是量測電能，更可以用來作為防盜的功能，若是家中有嬰幼兒或是年長的長者皆可以用來當成輔助角色，但目前家庭自動化都是用在較高級的住宅，一般的家庭要使用相對困難，再者以往要實作其工程相當浩大，費用也相對昂貴。

自從 Raspberry PI 基金會發佈 Raspberry PI 版本 A 之後，銷售量相當驚人，但是其軟體架設或是程式編寫門檻還是較高。

貳、研究目的

系統主要的目的就是實作輸出與輸入信號，由瀏覽器瀏覽網頁執行監控的目的。將來希望進一步結合一般的 Webcam，撰寫程式做影像辨認用以判斷是否有外物侵入，並進行電能監控、溫度、濕度、二氧化碳濃度且配合資料庫(SQLite)記錄相關資料等等功能，更可達成智慧家庭之功能。

本系統就是利用 Raspberry PI 架設網站並利用其內建的 GPIO(General Purpose Input Output)，可以讓使用者直接控制硬體，搭配 Python 可以撰寫更多程式，舉凡影像處理、資料庫等等程式，如此一來家庭自動化將更容易實現。

參、研究設備及器材

表 1 研究設備與器材

編號	設備及器材	規格	數量	備註
1	電烙鐵	40W	2	
2	烙鐵架		1	
3	吸錫器		2	
4	銲油		若干	
5	銲錫		若干	
6	個人電腦		2	
7	電源供應器	DC5V	2	
8	電源供應器	DC7.5V	2	
9	電源供應器	DC9V	2	
10	藍芽晶片模組	2.4GHz	2	
11	Arduino 電路板	ATMEGA328	10	
12	手機	LG-P350	2	
13	穩壓晶片	LM7805	10	
14	二極體	1N4001	20	
15	IC 腳座	28PIN	20	
16	IC 腳座	18PIN	20	
17	按鈕開關	8mm	20	
18	按鈕開關	TACT SW	20	
19	LED	5"紅色	20	
20	電源插座	公座	20	
21	電源插座	母座	20	
22	振盪晶體	16MHz	20	
23	振盪晶體	8MHz	20	
24	電容	22P	20	
25	電容	0.1uF	20	
26	電容	10uF	20	
27	牛角座	10PIN	20	
28	端子	2PIN	20	
29	通訊晶片	MAX232	5	

編號	設備及器材	規格	數量	備註
30	可變電阻	5K	10	
31	電阻	220Ω	10	
32	電阻	1kΩ	10	
33	電阻	10kΩ	10	
34	RS232 接頭	DB9 母座	10	
35	螺絲		10	
36	銅柱	3mm	10	
37	杜邦線	1PIN	20	
38	杜邦線	2PIN	20	
39	杜邦線	4PIN	20	
40	繼電器	6V	5	
41	電池	6V	5	
42	萬用電錶		2	
43	Rasberry PI		2	
44	壓克力	5mm 厚度	若干	
45	19 吋 LCD		1	
46	HDMI 導線		1	
47	CAT5 網路線		1	
48	鍵盤		1	
49	滑鼠		1	

肆、研究過程或方法

家庭自動化一般還是停留在早期的印象，都是針對各個電器用品的控制，卻無法達成系統整合，相對目前的各項 3C 產品遜色不少，因此就有很多使用者運用多方式去改善缺點。一般來說就是使用兩種方式，第一就是利用一般的 PC 架設伺服器，再將單晶片程式嵌入伺服器之網頁中，但 IO 還是只能利用 RS232 方式介面溝通。另一種就是直接使用單晶片轉寫其控制程式及 HTML 的程式，此一方式就是撰寫 CGI 程式，但此一方法工程浩大，就只能處理一般簡單網頁，但是本專題利用 Python 搭配 flask 是可以很容易實現動態網頁。

一、系統架構

如圖 2 所示，底下為各部份方塊描述如下：

(一)Raspberry PI 其內部必須架設 NTP server(ntpd)、internet superserver(inetd)、OpenBSD Secure Shell Server(sshd)、Web Server(lighttpd)、FTP Server(vsftpd)、Python、Flask 這些軟體主要的功能是，可以操控 Raspberry PI、架設網頁、上傳程式、動態網頁等等功能。

(二)Arduino：

Raspberry PI 因為輸出接腳只有 GPIO18 支接腳，為了擴充接腳與時下最知名 Arduino UNO 進行搭配。

(三)WIFI：

現在無線網路幾乎隨處可見，為了讓系統更加靈活我們會將整個系統加入 WIFI 模組，讓整個系統連接 Internet 更加容易。Raspberry PI 本身配備 2 個 USB 插槽使得 WIFI 連接更為方便。

(四)USB Camera：

為實作影像辨認，雖然 Raspberry PI 基金會已經生產一種無外殼的高階 CCD，但是一般時下 USB Camera 都可以適用，以便將來實作影像處理更加方便。

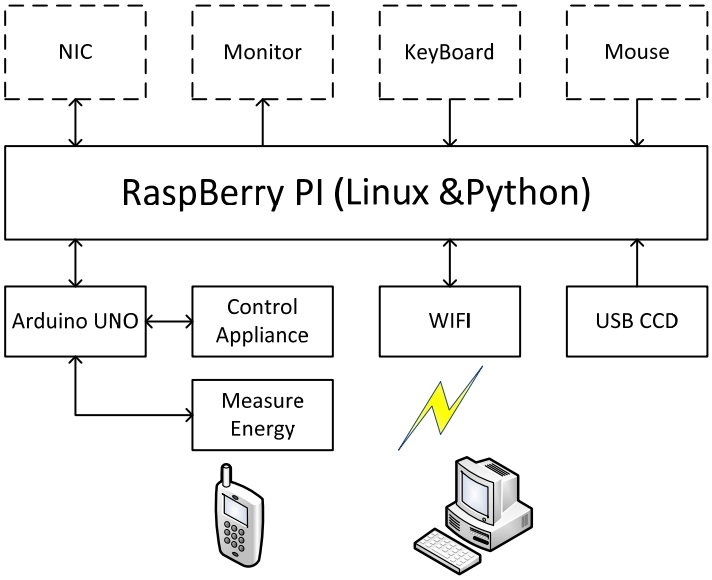

圖 1 系統架構圖

二、硬體架構：

(一)Raspberry PI

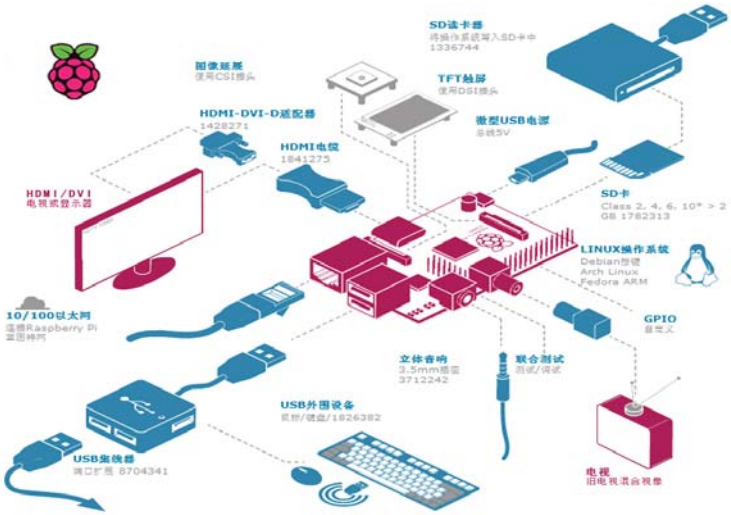

圖 3 Raspberry PIB 型板的宏觀結構圖

Raspberry PI 就相當於是一塊電腦主機板。居於長方形板面中心的是一塊 BCM2835 晶片，其上包括了一個 ARM1176JZF-S 700MHZ 的處理器（可以超頻至 1GHZ），和一塊 VideoCore IV GPU，還有 512M RAM（A 型板記憶體為 256M）。板子上面沒有內置的 Flash ROM，但是提供一個 SD 插槽，用戶可以將自己的 SD 卡插入作為硬碟使用。SD 插槽的旁邊是一個 Micro-USB 電源插孔，由此引入 5V 的電源，也可以通過 GPIO 介面的 2 號（VCC）和 3 號（GND）引腳提供 5V 電源。在 SD 插槽的另一邊是 26 針的擴展介面，其中 2 腳可以接入或給出 5V 電源，1 腳給出 3.3V 電源，第 12 腳可以提供 PWM，另外包括一組 UART，一組 SPI，一組 I2C，8 個 GPIO 引腳。擴展介面還算豐富。

兩種顯示介面方式。其一是 HDMI，HDMI 相容的電視或者顯示器可以直接通過這個介面獲得輸出的視頻信號，VGA 顯示器可以通過 HDMI-VGA 轉換器得到 VGA 信號。在 HDMI 相反方向是一個 RCA 介面，可以輸出類比影像信號供舊式電視使用。非常顯眼的可以看到疊在一起的兩個 USB 介面（A 型板只有一個 USB 介面），可以用來連接滑鼠和鍵盤，總是會遇到 USB 介面不夠用的時候，這是就需要我們通過 USB-HUB 來進行擴展。在 USB 介面旁邊是 10/100 M 乙太網介面。另外有三個不常用的 IO。一是，USB 口和 RCA 介面之間的聲音/測試介面；二是，板上面的類似於擴展介面的 JTA 介面；三是，TFT 觸摸控制螢幕。

核心晶片 BCM2835：嵌入式多媒體應用處理器 BCM2835 是一種低成本，高畫質多媒體應用處理器，適用需要高品質多媒體性能的移動和嵌入式應用設備。產品設計充分考慮了電池使用效率，並進行了相關優化。BCM2835 使用博通公司的第四代 VideoCore 技術來使能應用程式中的各種影像處理。

1.低功耗 ARM1176JZ-F 應用處理器

2.雙核第四代 VideoCore 多媒體協同處理器

3.1080P30 幀每秒全高畫質視編碼/解碼

4.高性能視頻輸出。1080P60 幀每秒持續高解析度 LCD、HDMI 輸出

5.低功耗，高性能 OpenGL-ES1.1/2.0VideoCoreGPU

6.先進的影像感測器流水線（ISP）長達 20 萬圖元的攝像頭高達 220 萬圖元每秒。

下面是該晶片的內部結構圖：

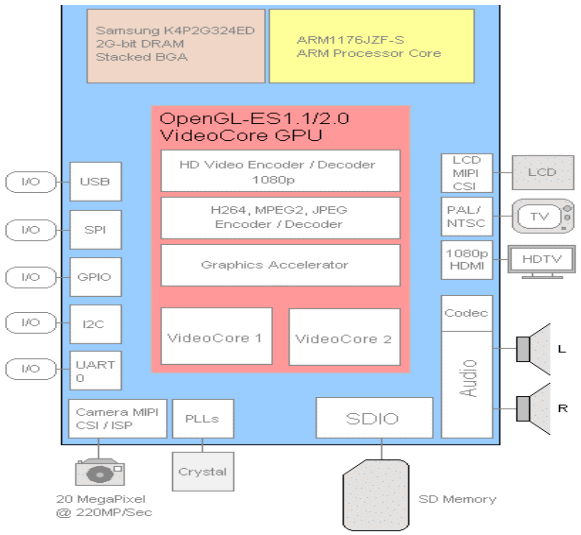

圖4 BCM2835內部結構圖

總之，Raspberry PI 的外部介面仍算豐富，處理器、存儲設備不夠強大。但因其低廉的價格（B 型板35美元，A 型板25美元），在8051單片機和個人電腦之間為我們提供了一個理想的嵌入式開發平臺。

軟體方面，毫無疑問選擇 GNU/Linux 平臺。做為開源軟體平臺，GNU/Linux

給予用戶以免費或者較低的價格獲取作業系統和軟體的程式碼的權力。這使得用戶可以根據自己的需要編輯、修改程式碼，分發自己的程式碼，鼓勵人們互相分享。其開放、共用的精神受到了許多科技人員的喜愛。如下是維基百科中關於 Debain 的描述：

Debian 是由 GPL 和其他自由軟體許可協定授權的自由軟體組成的作業系統，由 Debian 計畫（Debian Project）組織維護。Debian 計畫是一個獨立的、分散的組織，由3000志願者組成，接受世界多個非盈利組織的資金支持，Software in the Public Interest 提供支援並持有商標作為保護機構。Debian 以其堅守 Unix 和自由軟體的精神，以及其給予用戶的眾多選擇而聞名。現時 Debian 包括了超過37,500個套裝軟體並支持12個電腦系統結構（i386、amd64、arm、mips、IBM 等等）。Debian 是一個大的系統組織框架，在這個框架下有多種不同作業系統核心的分支計畫，主要為採用 Linux 核心的 Debian GNU/Linux 系統，其他還有採用 GNU Hurd 核心的 Debian GNU/Hurd 系統、採用 FreeBSD 核心的 Debian GNU/kFreeBSD 系統，以及採用 NetBSD 核心的 Debian GNU/NetBSD 系統。甚至還有應用 Debian 的系統架構和工具，採用 OpenSolaris 核心構建而成的 Nexenta OS 系統。在這些 Debian 系統中，以採用 Linux 核心的 Debian GNU/Linux 最為著名。眾多的 Linux 發佈版，例如 Ubuntu、Knoppix 和 Linspire 及 Xandros 等，都基於 Debian GNU/Linux。

(二)Arduino UNO

Paspberry PI 的 IO 在某些場合會數量不足，因此我們選擇 Arduino UNO，此電路板目前相當風行，於是利用此一電路板實現本系統所需達成之功能。完整電路圖如 5 所示，基本功能描述如下：

1.微控制器：ATmega328
2.工作電壓：5V
3.輸入電壓(建議)：7-12V
4.輸入電壓：6-20V
5.數位輸入腳位量：14(含 PWM 輸出腳位量 6)
6.類比輸入腳位量：6
7.每一腳位 DC 電流供應：40mA
8.3.3V 腳位 DC 電流供應：50mA
9.Flash 記憶體 32KB 其中 0.5KB 提供 Bootloader 使用
10.SRAM：32KB
11.EEPROM：1KB
12.脈波速度：16MHz

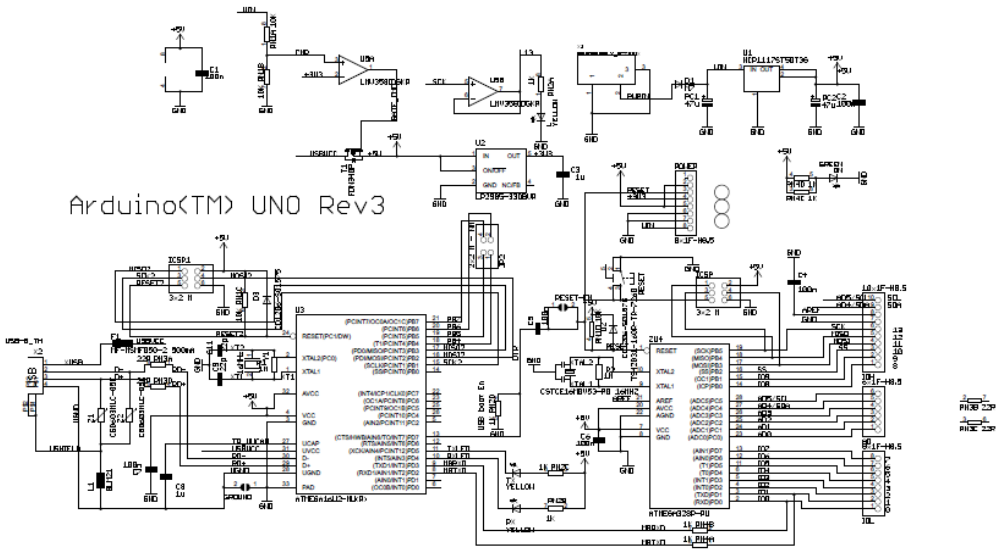

圖 6 Arduino UNO 完整電路

(三)基本輸出輸入

目前的進度只是實現 Raspberry PI 用網頁呈現輸出與輸入的狀態，因此只製作基本輸出與輸入的電路，輸入則利用 10k 歐姆電阻與按鈕開關串聯，再取出信號用以判斷高電位還是低電位。

輸出信號則用 LED 串聯一個 1K 電阻，用 LED 燈號來判斷是否動作與否，圖 7 為基本輸出輸入電路圖。

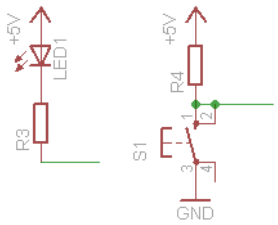

圖 7 基本輸出與輸入電路圖

三、軟體部分：

程式撰寫分成理兩個模組進行：Python 程式執行、flask 網頁執行。一般 Linux web server 之 port 80 是用以執行 web page，本專題除此功能外，展示三個測試程式，用以模擬家電用品。當需要測試 IO 時，則是在 server 端執行 python 程式，之後再利用 web page 將 python 之變數傳至 web page，如此一來便可以用網頁顯示伺服器時間及控制 Input 與 Output。

當控制 output 時採用切不同網頁之作法，採用 port 5000。且使用者選擇不同 output 時可以切換到不同網頁，採用輸入變數的方式切換。@app.route("/<changePin>/<action>") 是使用者以網頁觀看後所按下之網址，將可以在指定網只看到結果改變，同時也確定可以讓 LED 燈 ON 或是 OFF。

當測試 Input 時，主網頁是顯示目前伺服器之時間，當在瀏覽器輸入

@app.route("/readPin/<pin>")，則可以顯示所讀取腳位知信號為 HIGH 或 LOW。

伍、研究結果

目前執行底下幾個程式：顯示目前系統時間、由網頁讀取開關狀態、由網頁顯示 LED 燈狀態。以上程式確實可行。

一、顯示目前系統時間、由網頁讀取開關狀態：

(一)於樹莓派路徑/home/pi 輸入 python gpio.py，可於 http://192.168.11.2:5000/顯示出伺服器日期及時間，顯示結果如圖 8 所示。

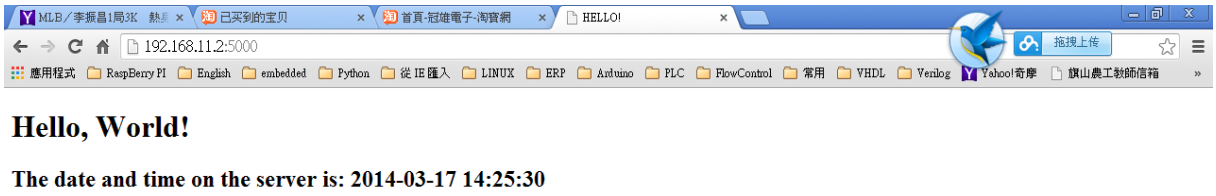

圖 8 顯示出伺服器日期及時間

(二)測試輸入信號於 http://192.168.11.2:5000/readPin/24 結果如圖 9 網頁所示，腳位 24 為高電位。

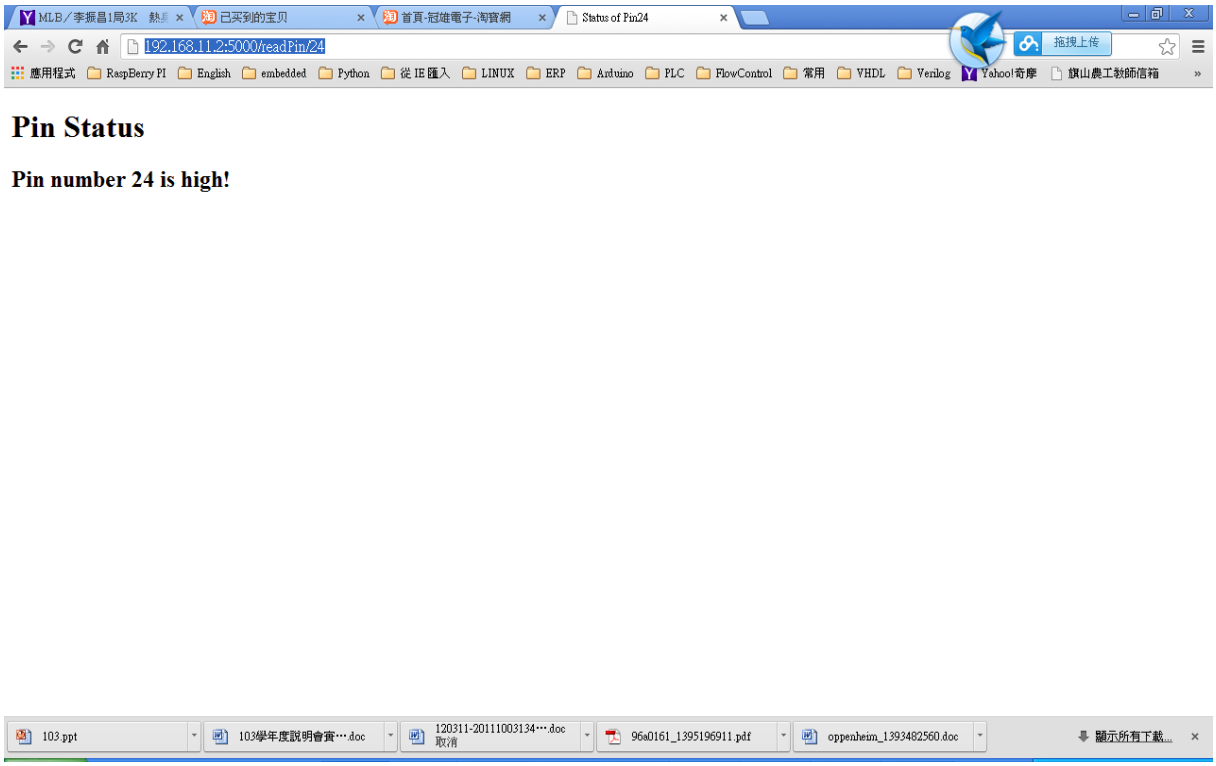

圖 9 網頁所示腳位 24 為高電位

(三)測試輸入信號於 http://192.168.11.2:5000/readPin/25 結果如圖 10 網頁所示，腳位 25 為高電位。

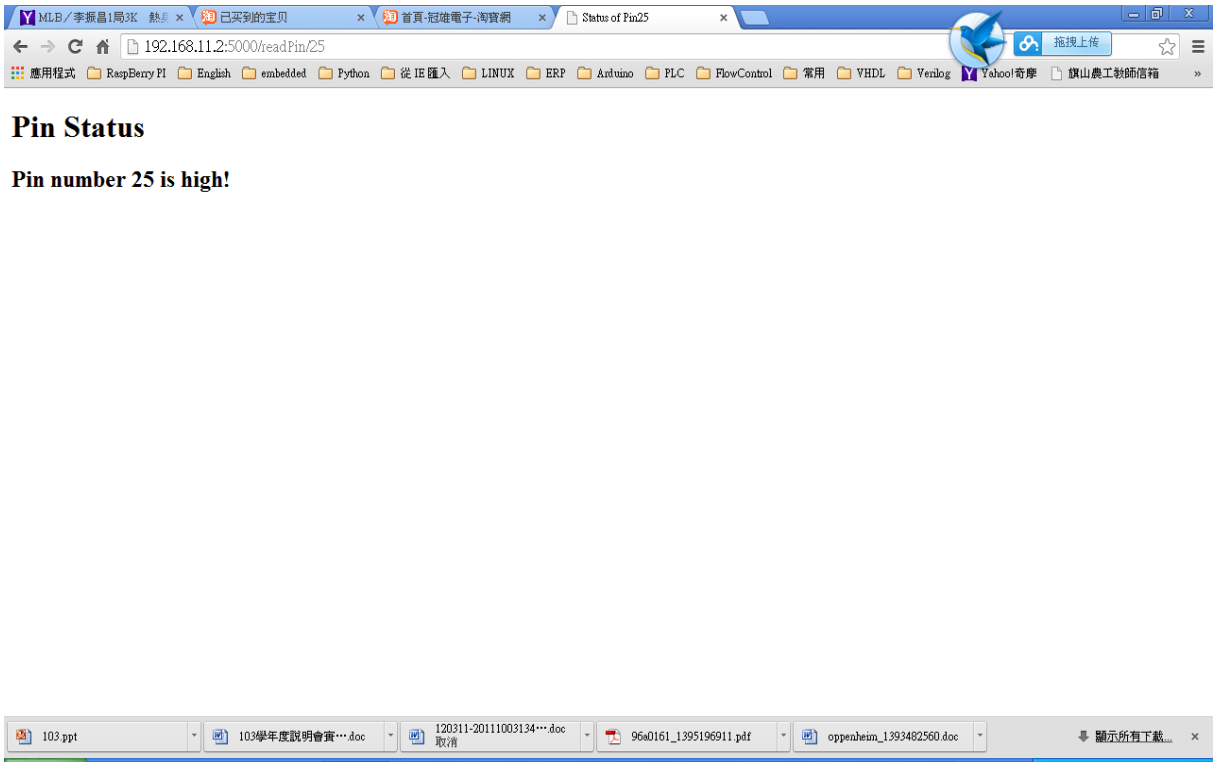

圖 10 網頁所示腳位 25 為高電位

(四)測試輸入信號於 http://192.168.11.2:5000/readPin/8 結果如圖 11 網頁所示，腳位

8 為高電位。

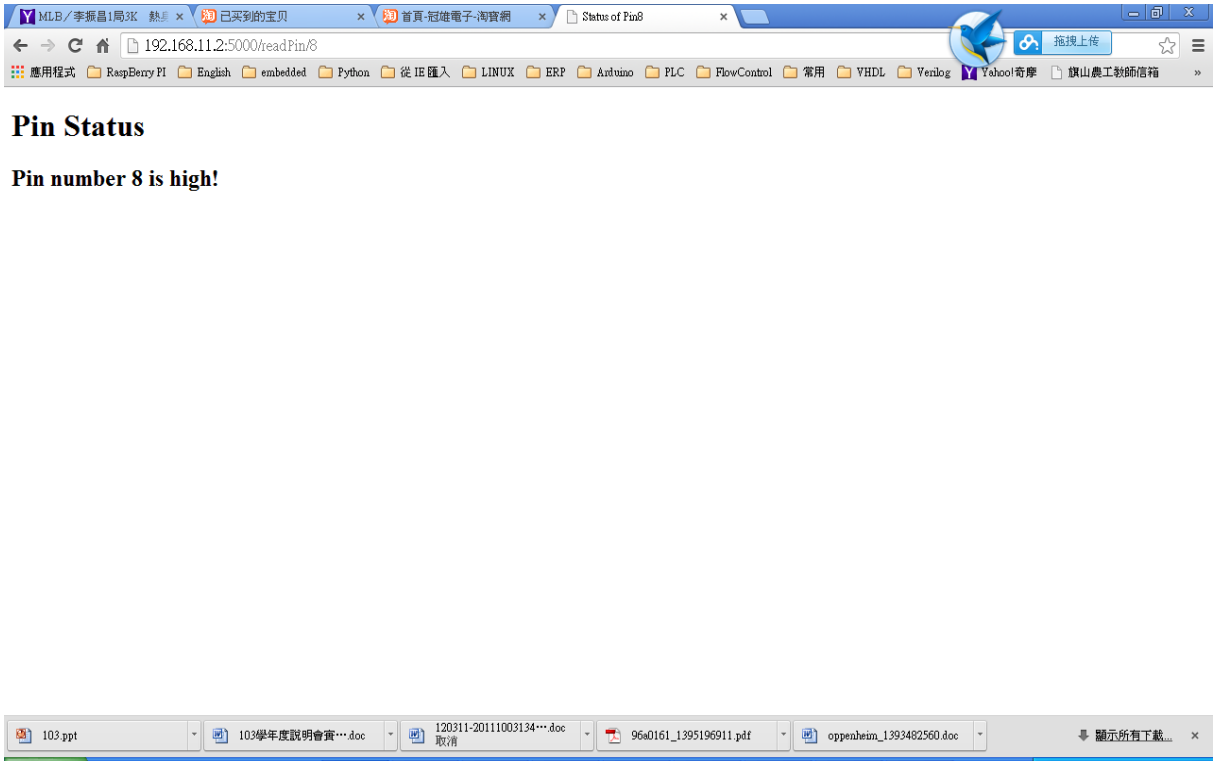

圖 11 網頁所示腳位 8 為高電位

(五)測試輸入信號於 http://192.168.11.2:5000/readPin/7 結果如圖 12 網頁所示，腳位 7 為高電位。

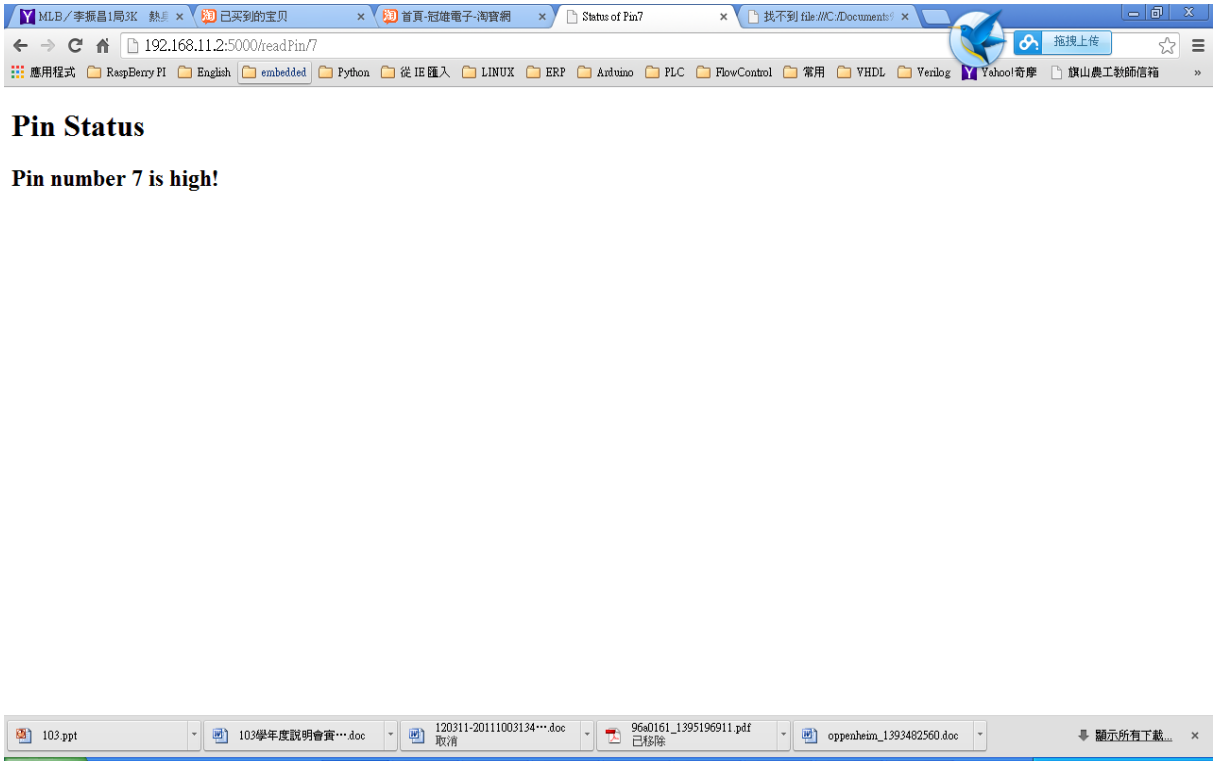

圖 12 網頁所示腳位 7 為高電位

二、測試輸出信號：

(一)首先在 raspberry pi 路徑/home/pi/weblamp 輸入 python weblamp.py 於 PC 或是手機之瀏覽器輸入 http://192.168.11.2:5000/顯示 lamp14 及 lamp15 可以控制 ON 與 OFF。如圖 13 所示。

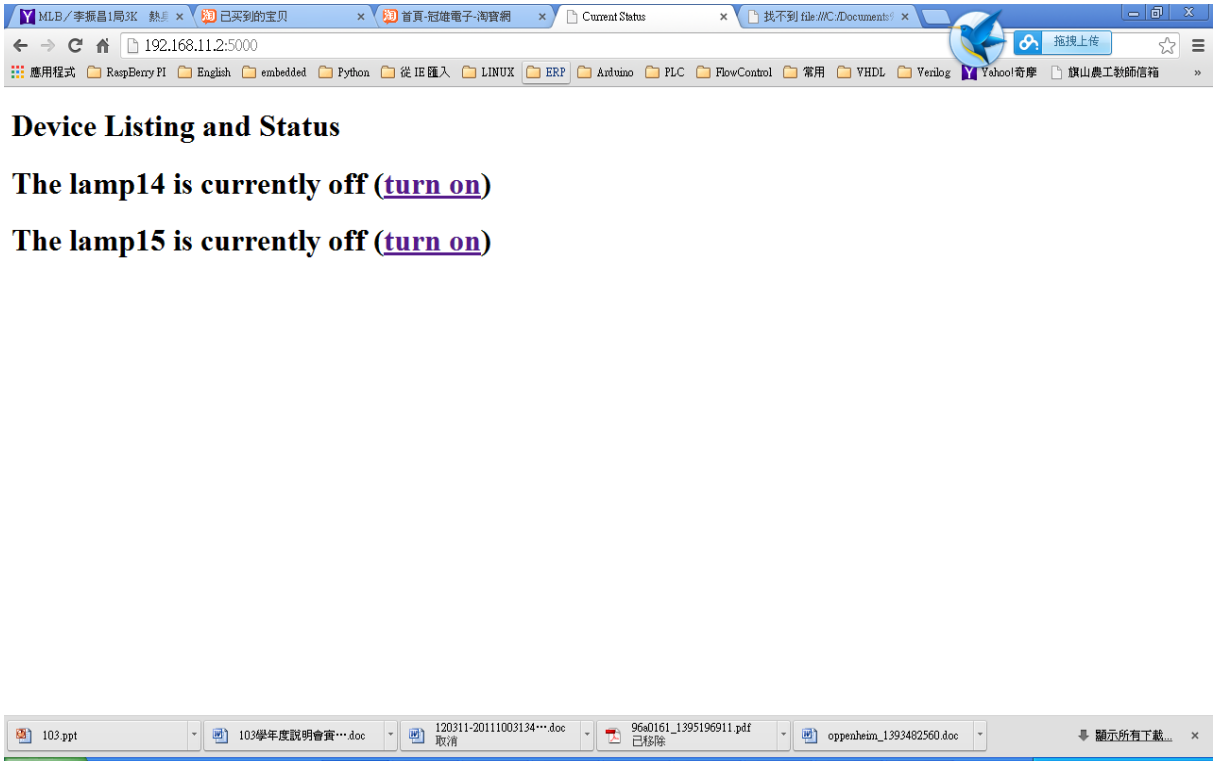

圖 13 lamp14 及 lamp15 控制狀態

(二)控制 Lamp14 ON 及實際顯示於 LED 如圖 14 及 15 所示。

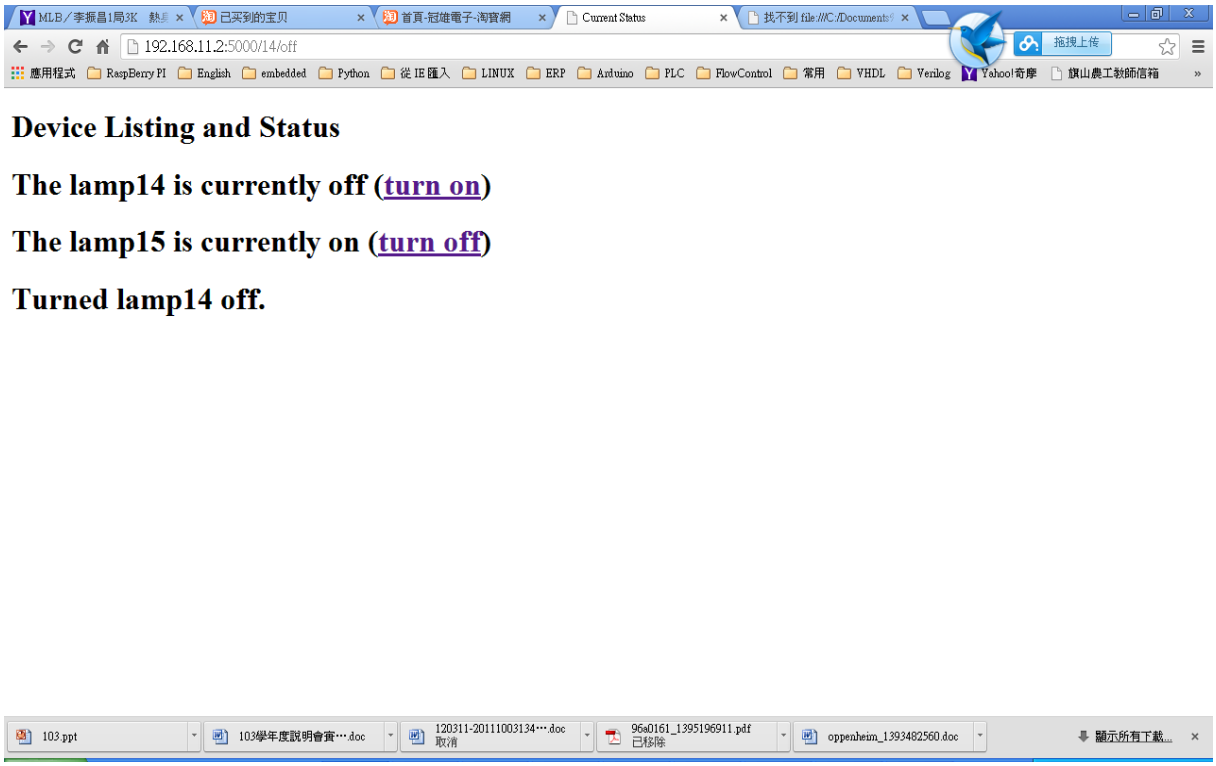

圖 14Lamp14 ON

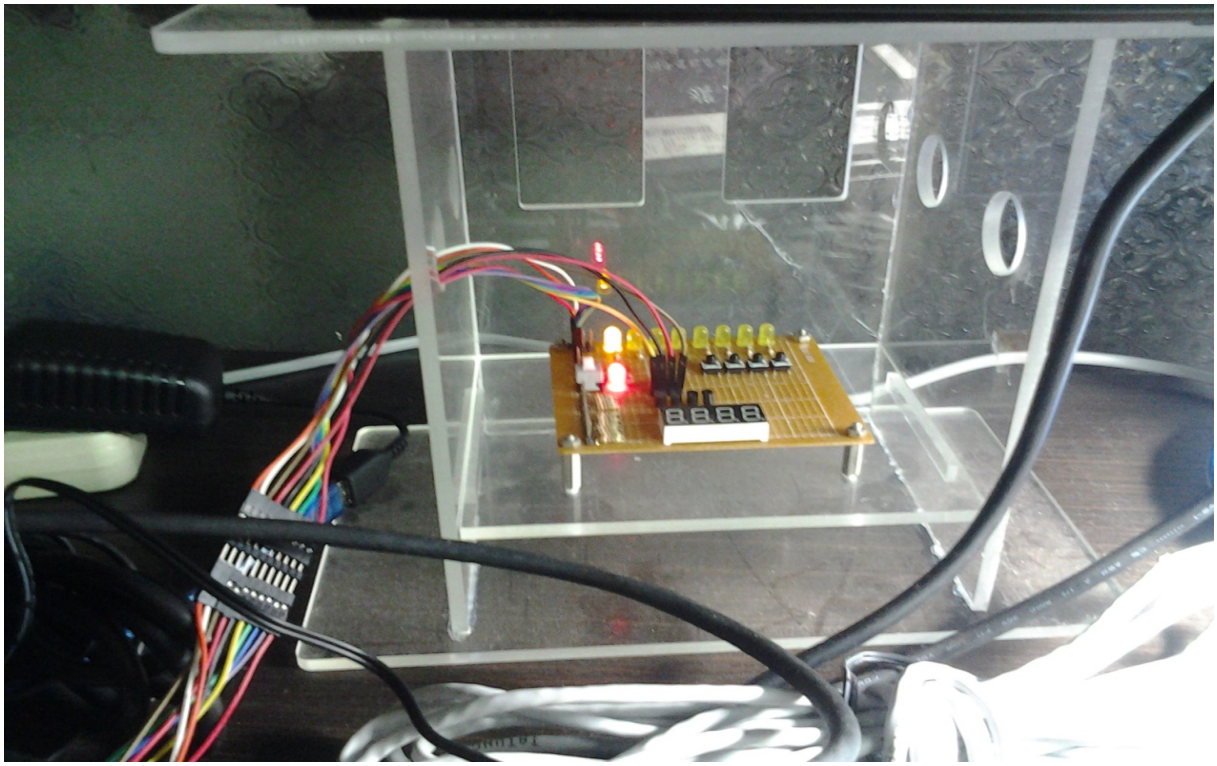

圖 15 實際顯示於 LED

(三)控制 Lamp15 ON 及實際顯示於 LED 如圖 16 及 17 所示。

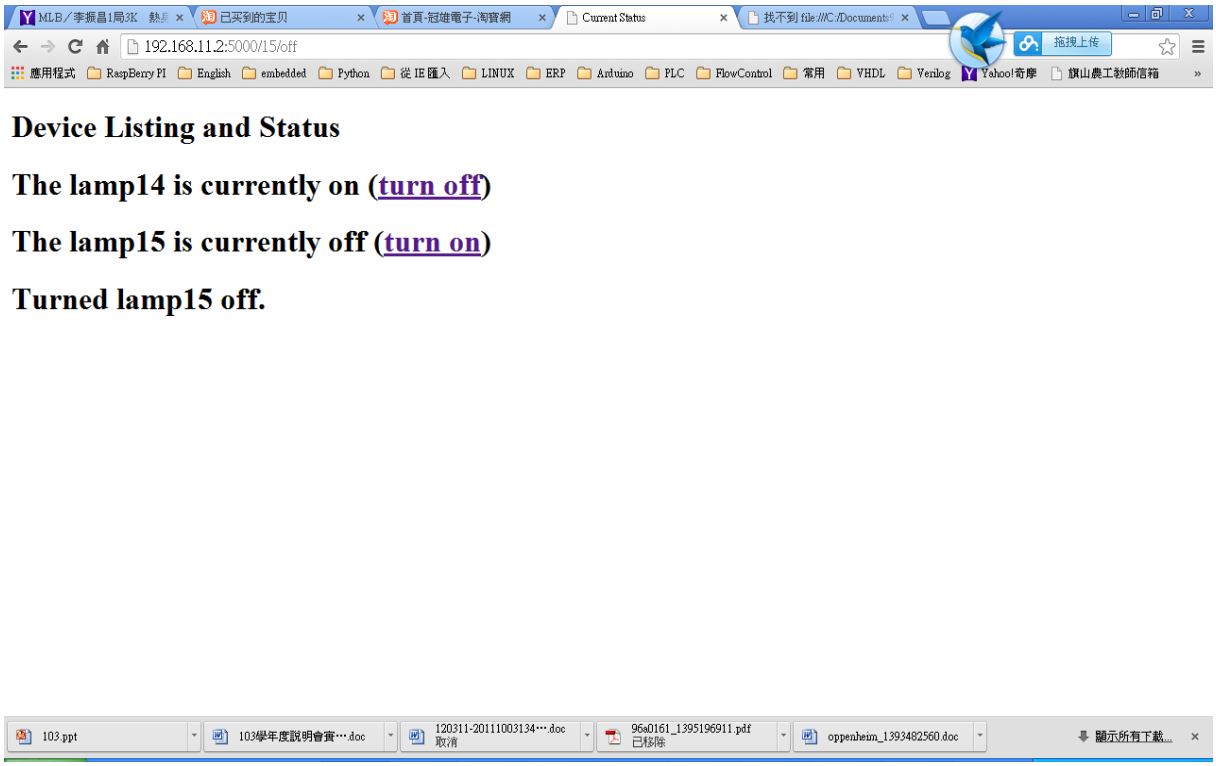

圖 14Lamp15 ON

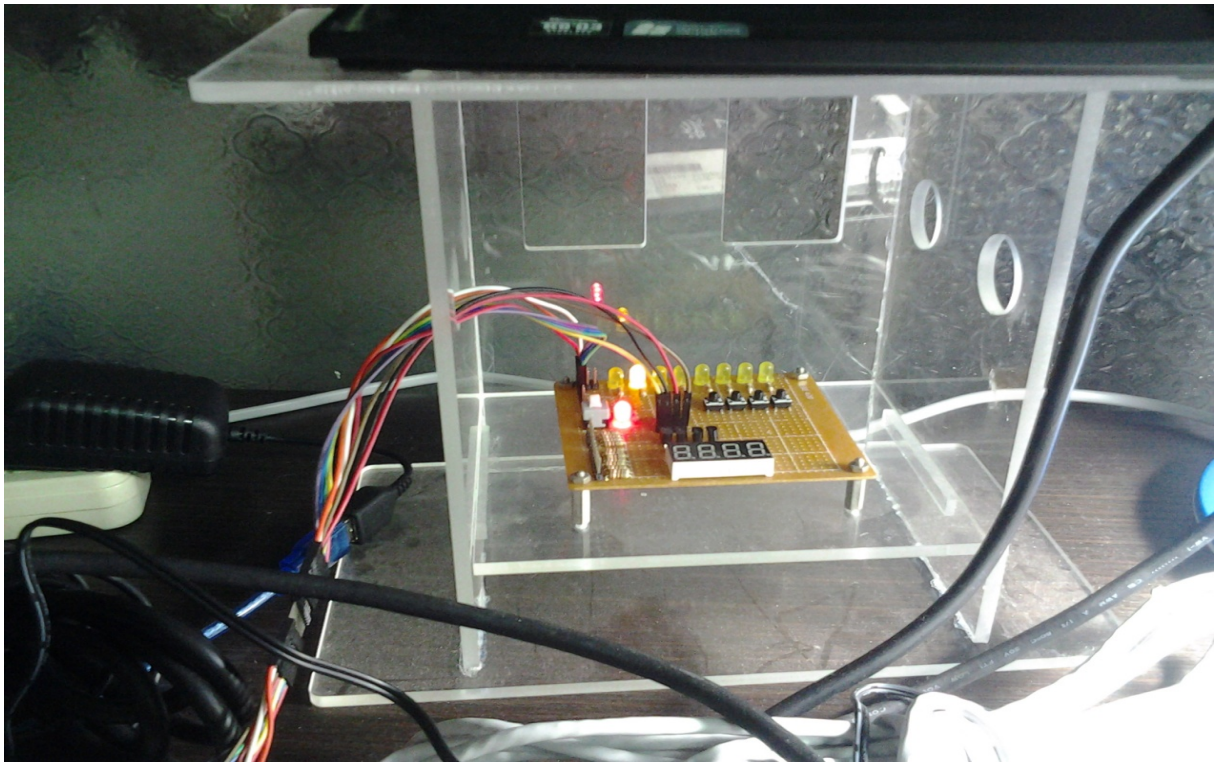

圖 17 實際顯示於 LED

(四)模擬房子之成品如圖 18 所示

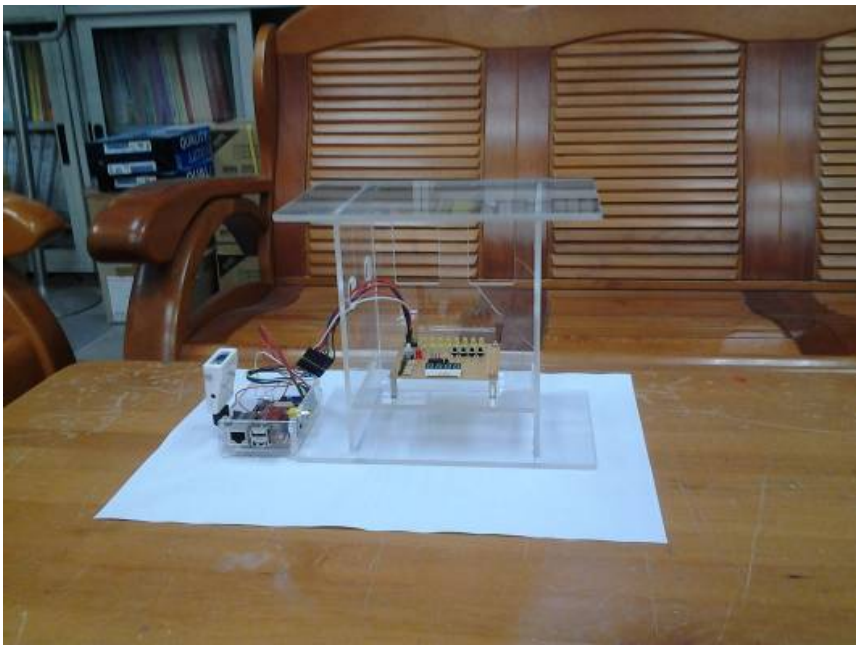

圖 18 模擬房子之成品

陸、討論與結論

因本專題所需之相關知識相對較廣泛，從 Raspberry pi、Linux 作業系統、web server、ftp server、ssh、python 及 HTML 這些知識我們三個人分工合作下，可以各司其職，因時間緊迫目前僅完成模擬之硬體電路。相信在短時間內應該可以大幅改善本專題之效能。

由以上 3 個程式可知，Raspberry PI 搭配 Python 及 HTML 確實可以達成家庭自動化，不用再添加複雜的軟硬體，只要專心開發 Python 程式及搭配 HTML 即可利用無遠弗屆的 Internet 達成家庭自動化。往後希望可以繼續針對溫度、濕度、電能量測、防盜、影像辨認及資料庫等議題深入研究，也更希望在更成熟之後可以商品化，為家庭自動化提供一份心力。

柒、參考資料及其他

[1]L. A. Zadeh,“Fussy Sets,” Informat. Contr., Vol. 8, pp. 338-353, 1965.

[2]L. A. Zadeh, “Fuzzy Algorithms,” Inform. Control, Vol. 12, pp. 94-102, 1968.

[3]廖仲文，模型車智慧型導航控制器，國立中央大學資訊及電子工程研究所碩士論文，1994。

[4]沈良寰，自走式機器人在未知環境之路徑規劃與執行，國立交通大學控制工程研究所碩士論文，1994。

[5]張世傑，智慧型車庫停車之設計、研製與實現，國立成功大學電機系碩士論文，1997。

[6]蘇裕記，以 FPGA 晶片實現智慧型車庫停車控制系統，國立成功大學電機系碩士論文，2000。

[7]陳意翔，具智慧型停車功能車型機器人之設計與研製，國立成功大學電機系碩士論文，2002。
[8]J. P. Laumond, P. E. Jacobs, M. Taix, and R. M. Murray, “A Motion Planner for Nonholonomic Mobile Robots,” IEEE Trans. on Robotics and Automation, Vol. 10, No. 5, pp. 577-593, 1994.
[9]H. Miyata, M. Ohki, Y. Yokouchi, and M. Ohkita*, “Control of The Autonomous Mobile Robot DREAM-1 for A Parallel Parking,” Mathematics and Computers in Simulation, Vol. 14, pp. 129 –138, 1996.

捌、程式列表

1.Gpio.py

```
from flask import Flask, render_template
import datetime
import RPi.GPIO as GPIO
app = Flask(__name__)

GPIO.setmode(GPIO.BCM)

@app.route("/")
def hello():
    now = datetime.datetime.now()
    timeString = now.strftime("%Y-%m-%d %H:%M:%S")
    templateData = {
        'title' : 'HELLO!',
        'time': timeString
        }
    return render_template('main.html', **templateData)

@app.route("/readPin/<pin>")
def readPin(pin):
    try:
        GPIO.setup(int(pin), GPIO.IN)
        if GPIO.input(int(pin)) == True:
            response = "Pin number " + pin + " is high!"
        else:
            response = "Pin number " + pin + " is low!"
    except:
        response = "There was an error reading pin " + pin + "."

    templateData = {
```

```
        'title' : 'Status of Pin' + pin,
        'response' : response
        }

    return render_template('pin.html', **templateData)


if __name__ == "__main__":
    app.run(host='0.0.0.0', port=5000, debug=True)
```

2. weblamp.py

```
import RPi.GPIO as GPIO
from flask import Flask, render_template, request
app = Flask(__name__)

GPIO.setmode(GPIO.BCM)

pins = {
    14 : {'name' : 'lamp14', 'state' : GPIO.LOW},
    15 : {'name' : 'lamp15', 'state' : GPIO.LOW}
    }

for pin in pins:
    GPIO.setup(pin, GPIO.OUT)
    GPIO.output(pin, GPIO.LOW)

@app.route("/")
def main():
    for pin in pins:
        pins[pin]['state'] = GPIO.input(pin)
    templateData = {
        'pins' : pins
        }
    return render_template('main-weblamp.html', **templateData)

@app.route("/<changePin>/<action>")
def action(changePin, action):
    changePin = int(changePin)
    deviceName = pins[changePin]['name']
    if action == "on":
        GPIO.output(changePin, GPIO.HIGH)
        message = "Turned " + deviceName + " on."
    if action == "off":
        GPIO.output(changePin, GPIO.LOW)
        message = "Turned " + deviceName + " off."
    if action == "toggle":
        GPIO.output(changePin, not GPIO.input(changePin))
        message = "Toggled " + deviceName + "."

    for pin in pins:
        pins[pin]['state'] = GPIO.input(pin)
```

```
    templateData = {
        'message' : message,
        'pins' : pins
    }

    return render_template('main-weblamp.html', **templateData)

if __name__ == "__main__":
    app.run(host='0.0.0.0', port=5000, debug=True)
```

3. weblamp.html

```
<!DOCTYPE html>
<head>
    <title>Current Status</title>
</head>

<body>
<h1><span style="font-size:24pt;">Device Listing and Status</span></h1>

    <h1><span style="font-size:24pt;">{% for pin in pins %}</span></h1>

<h1><span style="font-size:24pt;">The {{ pins[pin].name }}
    {% if pins[pin].state == true %}
        is currently on (<a href="/{{pin}}/off">turn off</a>)
    {% else %}
        is currently off (<a href="/{{pin}}/on">turn on</a>)
    {% endif %}</span>
        </h1>
    <h1><span style="font-size:24pt;">{% endfor %}

    {% if message %}</span></h1>

<h1><span style="font-size:24pt;">{{ message }}</span></h1>
    <h1><span style="font-size:24pt;">{% endif %}</span></h1>


</body>
</html>
```

【評語】091010

1. 本研究主題並不具創新性，但系統實作完整，實用性佳。

2. 三位組員分工明確，各司其職，但整合度高，團隊合作精神佳。