基於行動裝置的遠端恆溫系統監控

亞洲大學 資訊工程系
學生:沈東政、劉庭瑋、張簡彥佑
指導老師:陳永欽 教授

摘要

- 本研究以Arduino開發平台為基礎結合PID(比例–積分–微分控制器)與PWM(脈衝寬度調變)之恆溫監控系統，主要核心是使用PID為控制運算，將感測到的溫度經過PID運算後，再使用PWM改變電流輸出中之佔空比，控制交流供電的燈泡升溫，與直流供電的風扇降溫，以達到對升降溫裝置功率的控制。

- 運用IoT(物聯網)技術，將裝置與雲端技術結合，實現多台裝置的恆溫監控。透過我們所建置的操作系統，可同時監控分散在各地的裝置，只要環境裡有WiFi讓裝置能夠連上網路，就可以讓使用者即使在世界各地都能透過擁有上網功能的行動裝置進行監控。本系統會自動將接收到的溫度數據即時繪出溫度曲線圖，使用者能清楚了解當前以及過去的溫度變化。除此之外，系統會將收集到的溫度數據儲存於系統資料庫中，當需要使用時可以立刻匯出，便於未來對溫度變化作Big Data分析研究，或是將數據作進一步的應用。

章節架構

- 第一章　緒論
- 第二章　研究背景
- 第三章　系統設計
- 第四章　系統實作
- 第五章　結果討論
- 第六章　結論與未來展望

第一章 緒論

- 研究動機
- 恆溫監控應用

研究動機

- 氣候變遷(ex:溫室效應)使得大環境溫度發生異變，導致產業損失慘重
- 科技日新月異，功率越大，溫度越高，例如像機房同時有多台機器在運行，避免溫度過高，預防意外發生
- 有效降低儀器的耗損，延長使用壽命，讓儀器能在適當的溫度下，能發揮最大的效能

恆溫監控應用

植物栽培

機房

養殖業

實驗室 細胞培養

第二章 研究背景

- Arduino
- 物聯網

Arduino

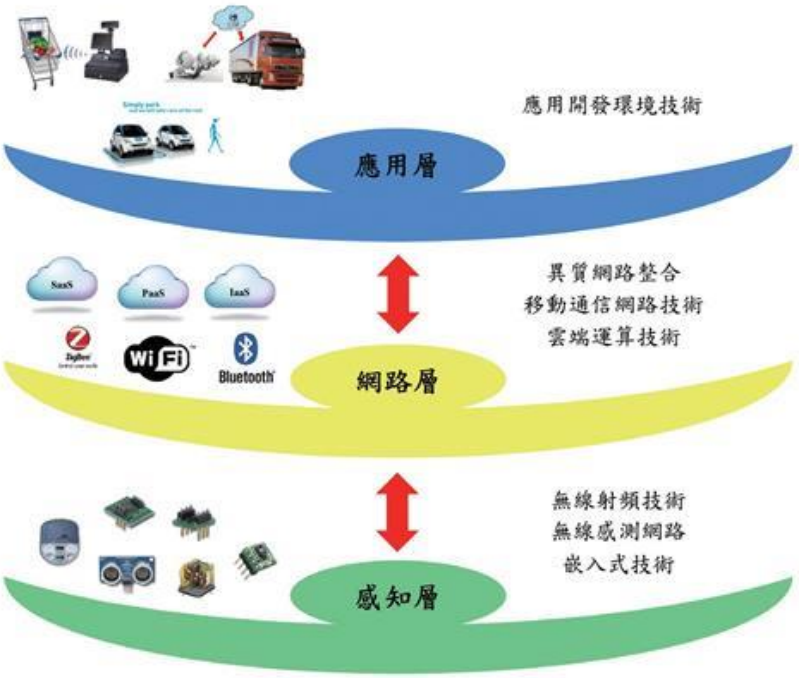
- Arduino是一種開放授權的開發平台，不僅軟體開放，連硬體都是開放的。Arduino 學習門檻簡單，不需要電子電機相關科系的背景，也可以很容易學會Arduino相關互動裝置的開發。

- 由於 Arduino 以公開共享為基礎，所以參考資料相當豐富，找出相關資料並依據自身的需求行調整，就可以在短時間內完成自己的創作。也因為Arduino 開發迅速，硬體成本低廉，Arduino是設計者(Maker)實現原型機的首選。

物聯網

- 過去的互聯網是人與人透過網路相互聯繫，因為互聯網的快速發展，進而延伸出互聯網與物件結合的應用，我們俗稱物聯網。物聯網讓人可透過網路取得物件的資訊外，物件與物件之間可以互通的網路環境。

- 物聯網可分為三個階層
  - 感知層(Device):溫度、濕度、紅外線、光度、壓力、音量等各式感測器
  - 網路層(Connect):將感知層收集到的資料透過網路層準確傳遞到應用層
  - 應用層(Manage):將資訊與各行業間的專業進行技術融合



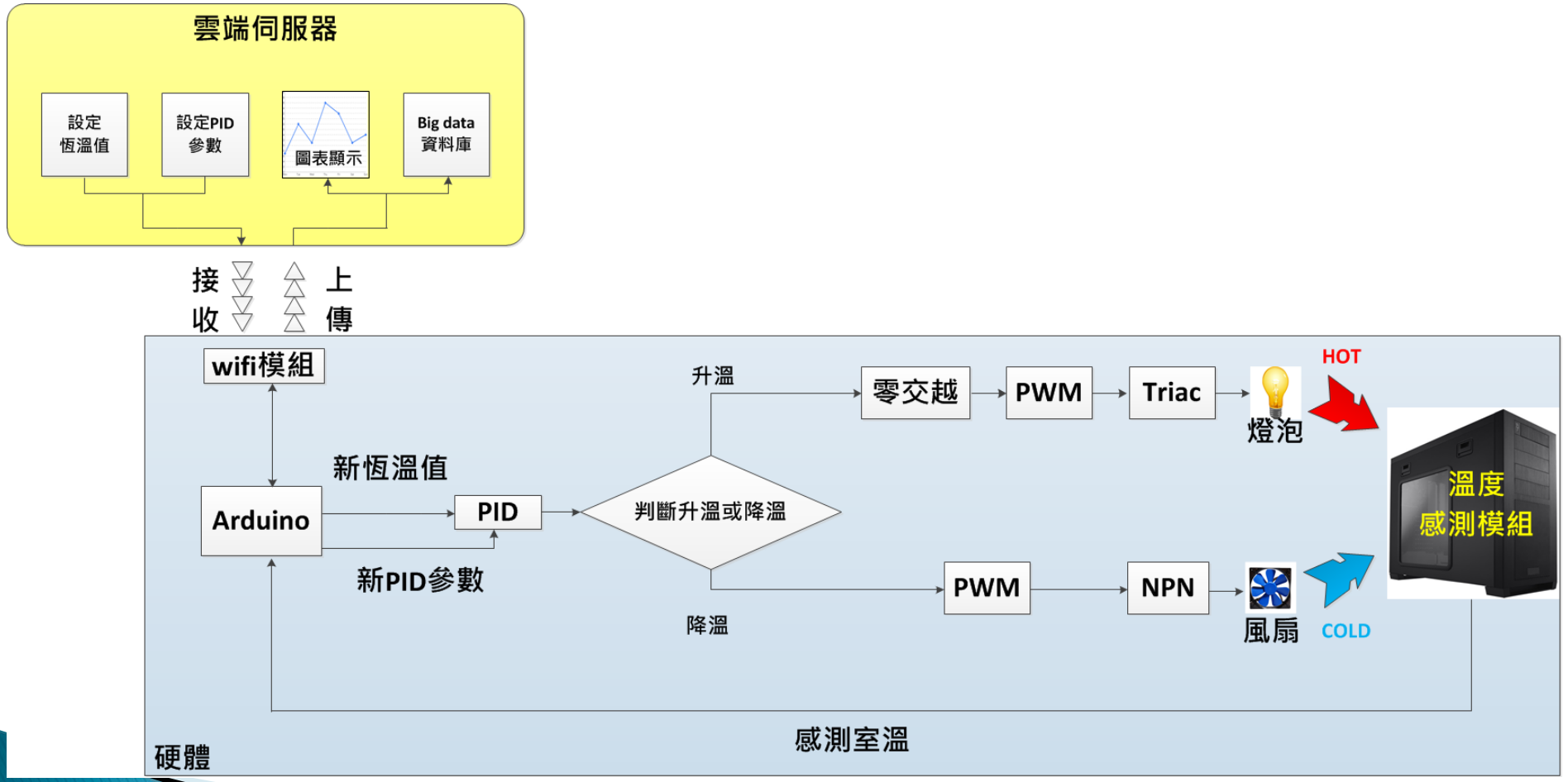
物聯網應用

- 老人看護系統

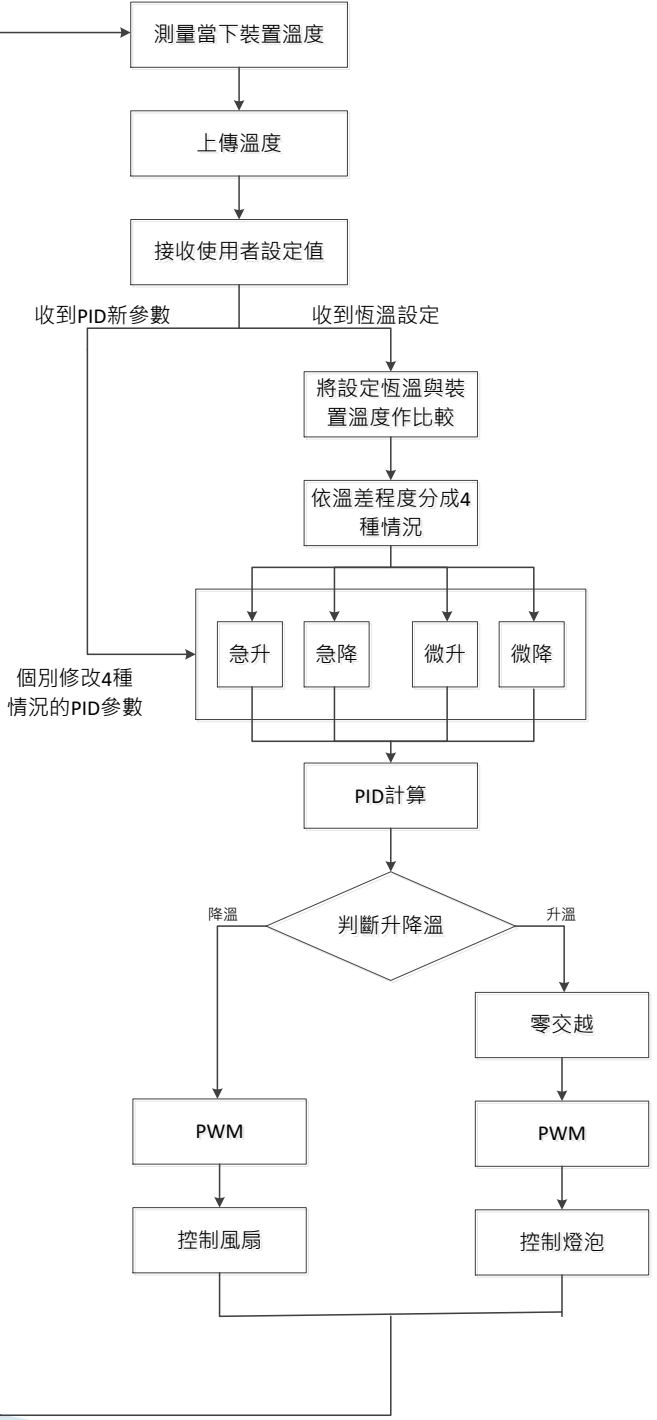
第三章 系統設計

- PID控制器
- PWM調變

系統架構

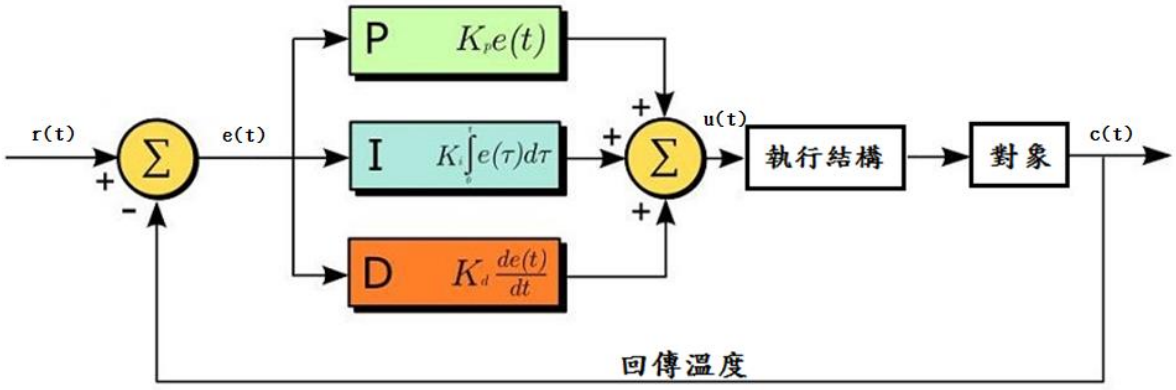

雲端伺服器
設定
恆溫值
設定PID
參數
圖表顯示
Big data
資料庫
接收
上傳
wifi模組
Arduino
新恆溫值
新PID參數
PID
判斷升溫或降溫
升溫
降溫
零交越
PWM
Triac
燈泡
HOT
PWM
NPN
風扇
COLD
溫度
感測模組
感測室溫
硬體

程式流程圖


測量當下裝置溫度
上傳溫度
接收使用者設定值
收到PID新參數
收到恆溫設定
將設定恆溫與裝置溫度作比較
依溫差程度分成4種情況
急升
急降
微升
微降
個別修改4種情況的PID參數
PID計算
判斷升降溫
降溫
升溫
零交越
PWM
PWM
控制風扇
控制燈泡

PID控制器

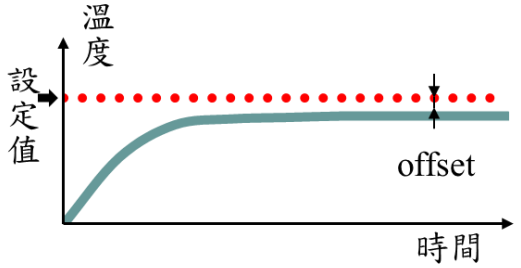
- PID控制器是一種在工業控制中常見的自動控制物件，主要運作原理為透過誤差值的大小調整輸出功率，縮小誤差，並自動持續運算，最終理想狀態為誤差值=0



- P(比例控制)
  - P 控制器的輸出與輸入誤差訊號成比例關係，但 僅有比例控制時，系統輸出會上下震盪並存在一穩態誤差(offset)。

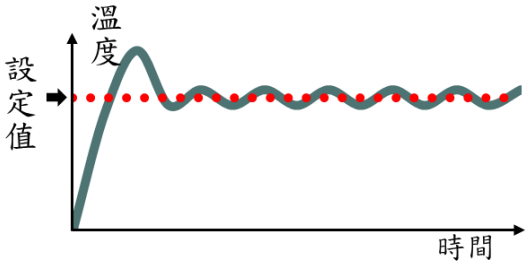
- I(積分控制)
  - I 控制器的輸出與輸入誤差訊號的積分成正比關係，如果在進入穩態後存在穩態誤差，為了消除穩態誤差(offset)，在控制器中必須引入積分項。

    (消除P控制後的穩態誤差)



- D(微分控制)
  - D 控制器的輸出與輸入誤差訊號的微分（即誤差的變化率）成正比關係，只要加入微分項，即可預測誤差變化的趨勢，比例微分的控制器，就能夠提前讓克服誤差的控制作用等於零，甚至為負值，從而避免了被控量的嚴重地沖過頭。

    (預測並消除系統未來的誤差)



傳統PID轉為數位PID

- $e(n) = r(n) - c(n)$　　誤差值=設定值-回傳值

- $u(n) = K_P\left\{e(n) + \frac{T}{T_I}\sum_{i=1}^{n} e(n) + \frac{T_D}{T}[e(n) - e(n-1)]\right\} + u_0$　　=> 傳統PID基本公式

因傳統公式計算繁複，造成程式執行的負擔，所以代入n與$n-1$，並將兩式相減，得以將計算簡化

- $u(n-1) = K_P\left\{e(n-1) + \frac{T}{T_I}\sum_{i=1}^{n-1} e(n) + \frac{T_D}{T}[e(n-1) - e(n-2)]\right\} + u_0$

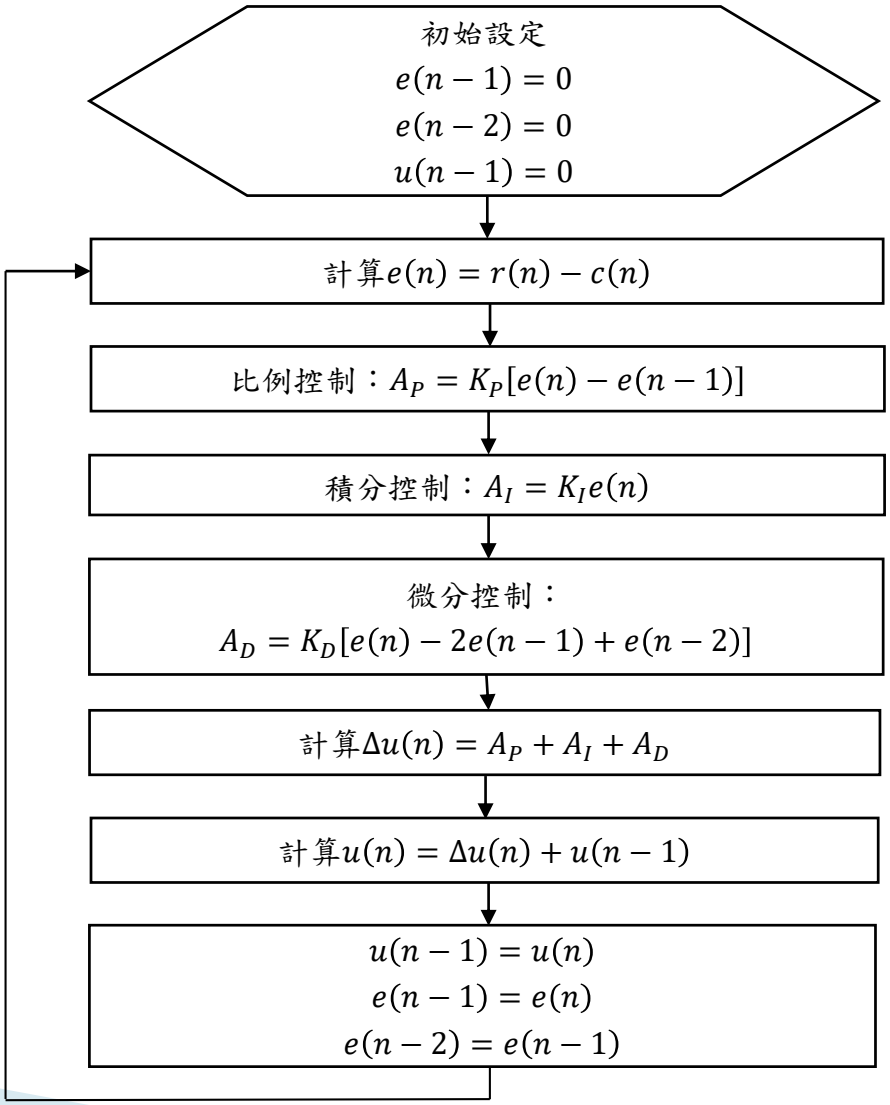
- $\Delta u(n) = u(n) - u(n-1) = K_P[e(n) - e(n-1)] + K_I e(n) + K_D[e(n) - 2e(n-1) + e(n-2)]$

兩式相減後得出的$\Delta u(n)$公式，再將$u(n-1)$向後移

- $u(n) = K_P[e(n) - e(n-1)] + K_I e(n) + K_D[e(n) - 2e(n-1) + e(n-2)] + \mathrm{u(n-1)}$

透過簡化，得出的$u(n)$，將較於傳統公式，簡化後的公式避免掉繁複所造成的負擔，此為數位PID更適合程式開發使用

數位PID流程圖


初始設定
$e(n-1)=0$
$e(n-2)=0$
$u(n-1)=0$
計算$e(n)=r(n)-c(n)$
比例控制：$A_P=K_P[e(n)-e(n-1)]$
積分控制：$A_I=K_Ie(n)$
微分控制：
$A_D=K_D[e(n)-2e(n-1)+e(n-2)]$
計算$\Delta u(n)=A_P+A_I+A_D$
計算$u(n)=\Delta u(n)+u(n-1)$
$u(n-1)=u(n)$
$e(n-1)=e(n)$
$e(n-2)=e(n-1)$

PWM調變

- PWM(脈衝寬度調變)
  - 一般脈波會有固定的週期，PWM是透過調整脈波之佔空比(Duty cycle)來控制電壓輸出的大小，佔空比越大，電壓越大，燈泡的亮度就越大

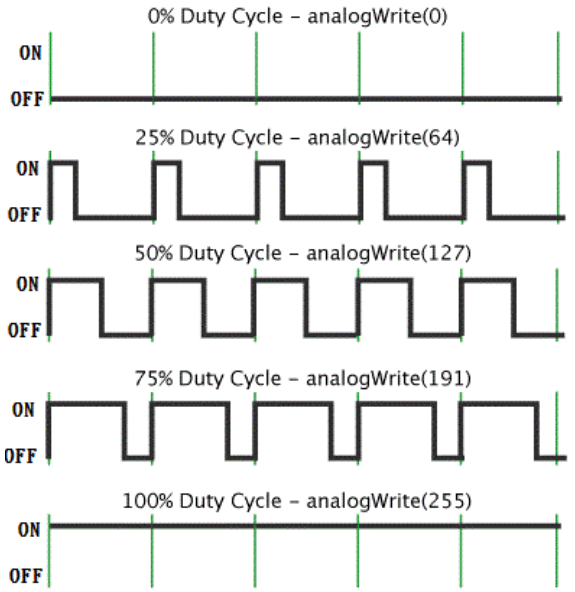
Ex: 如果原始電壓為12V

透過PWM佔空比調節

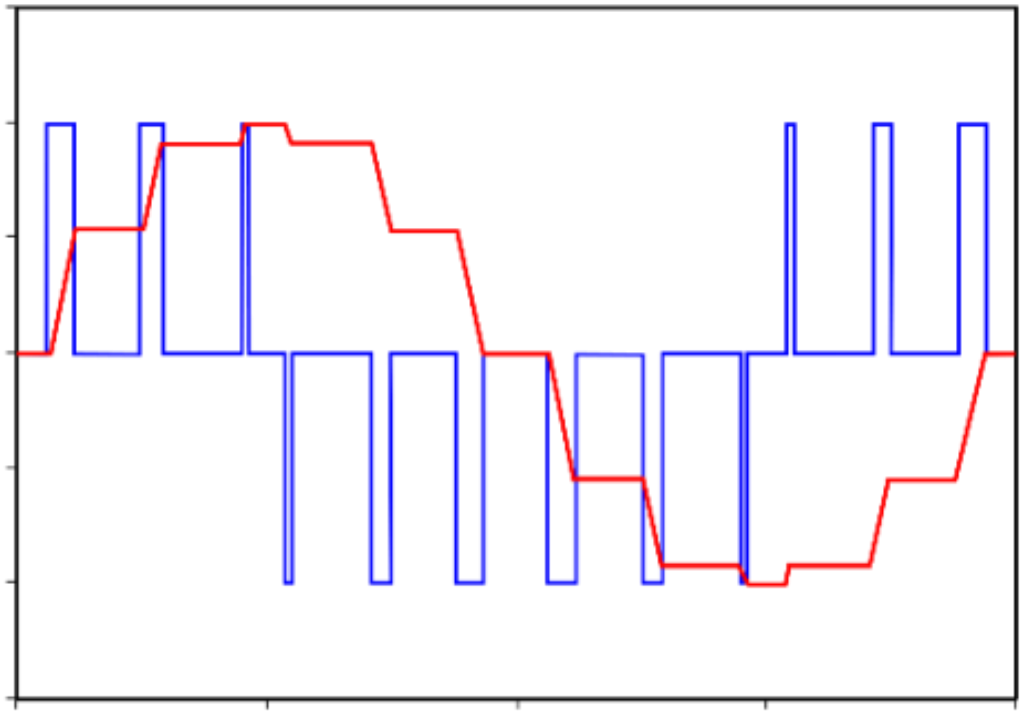
佔空比25%=>3V

佔空比75%=>9V

佔空比100%=>12V



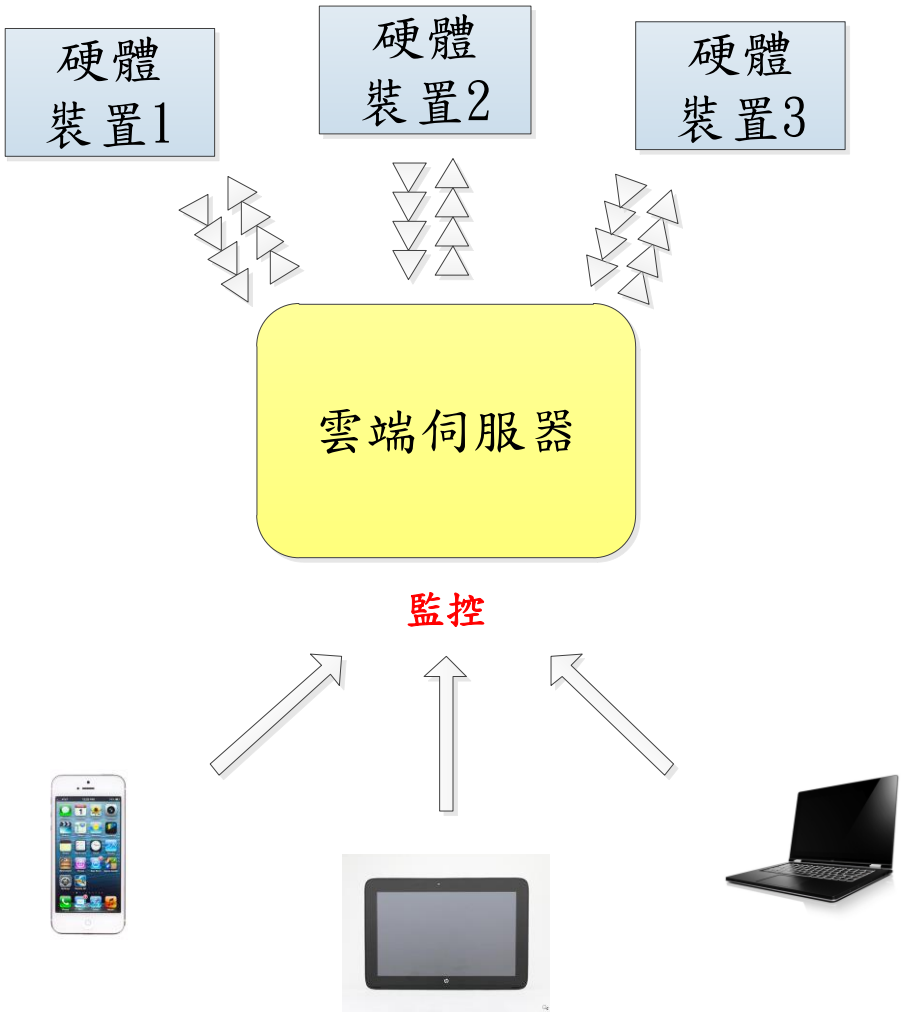
- PWM佔空比與升降溫速率示意圖



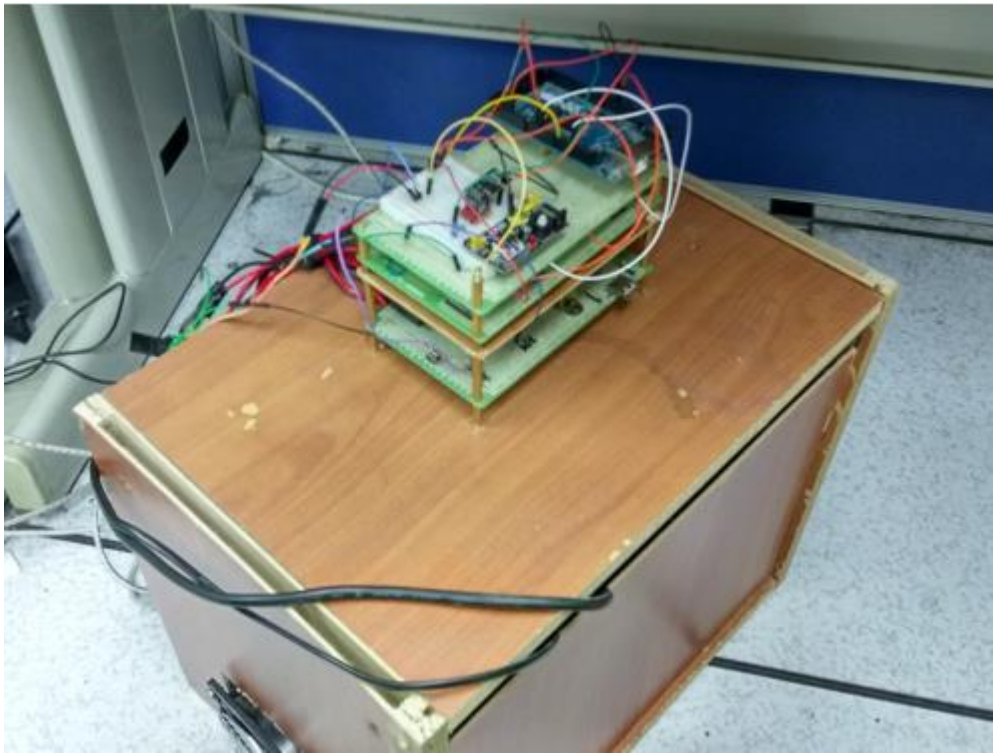
第四章 系統實作

- 系統架構圖
- 多點監控架構圖
- 程式流程圖
- 硬體照片
- 燈泡電路實作
- 風扇電路實作
- 使用者介面

多點監控架構圖



硬體照片



燈泡

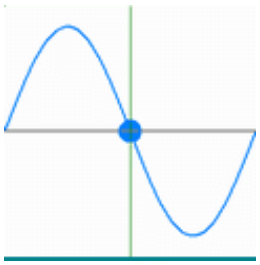
溫度感測器

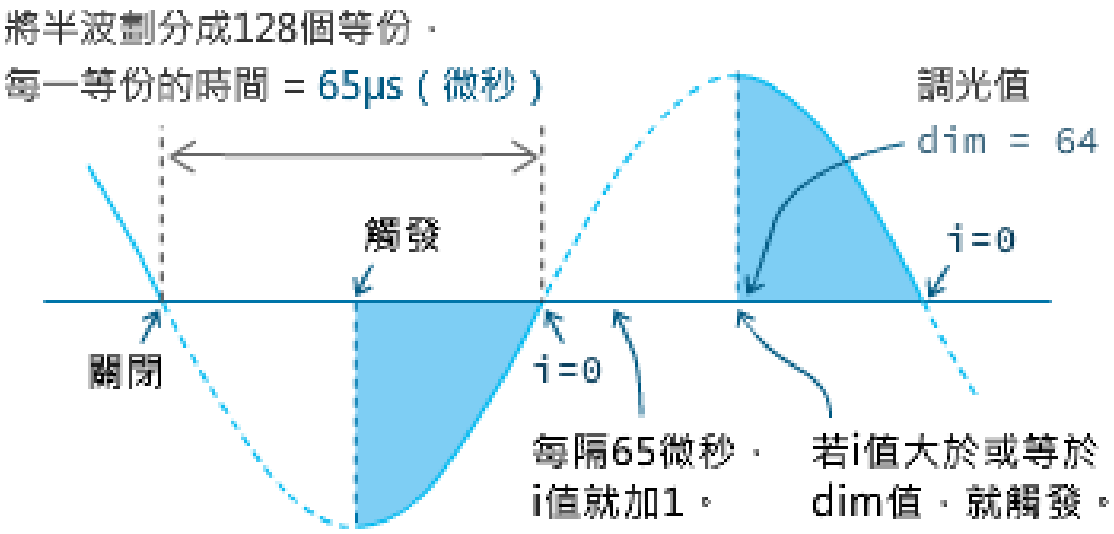
WiFi模組

風扇

零交越

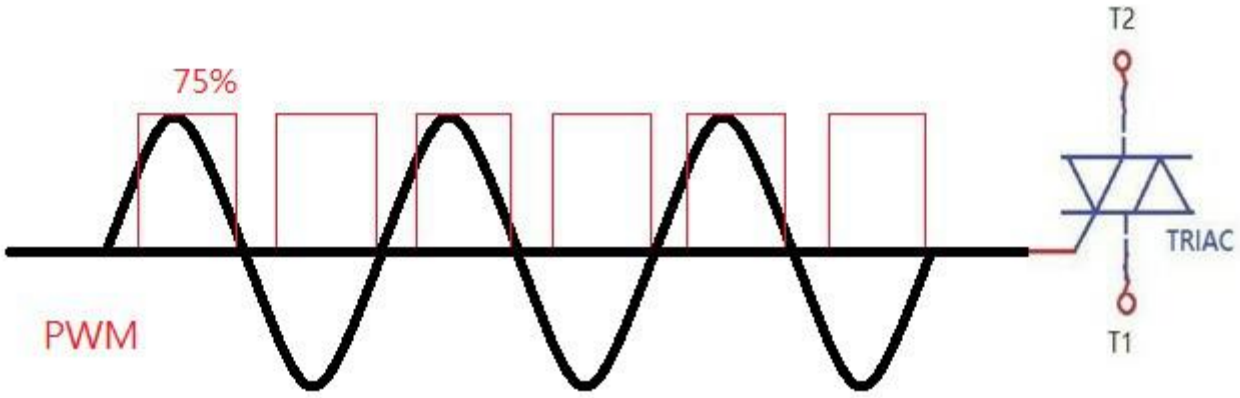
- 因為本研究外部電源使用110V的交流電，如果要使用PWM來調節輸出電壓，必須找出交流電的零交越點作為起始點，到下一個零交越點為結束點，每個半波為一個區間，這樣PWM調節出來的電壓才會是正確的



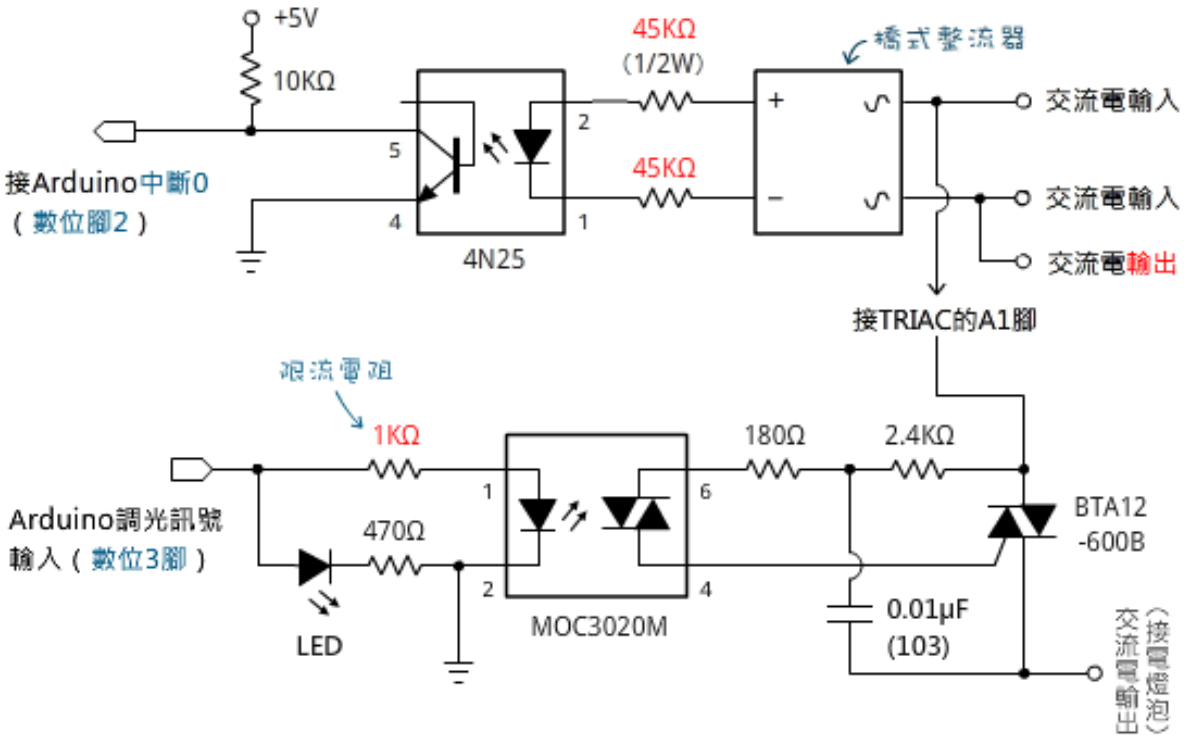


燈泡(交流電)

- 由於燈泡是使用交流電，所以必須使用Triac(三極交流開關)控制元件，PWM控制Triac的導通來調整燈泡的亮度



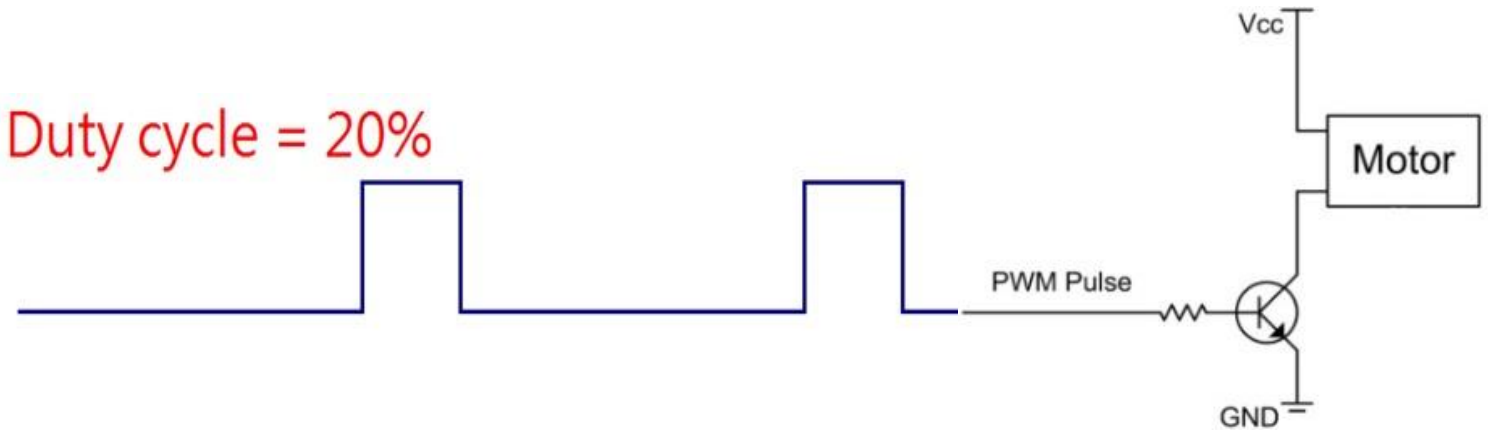
交流電調光電路



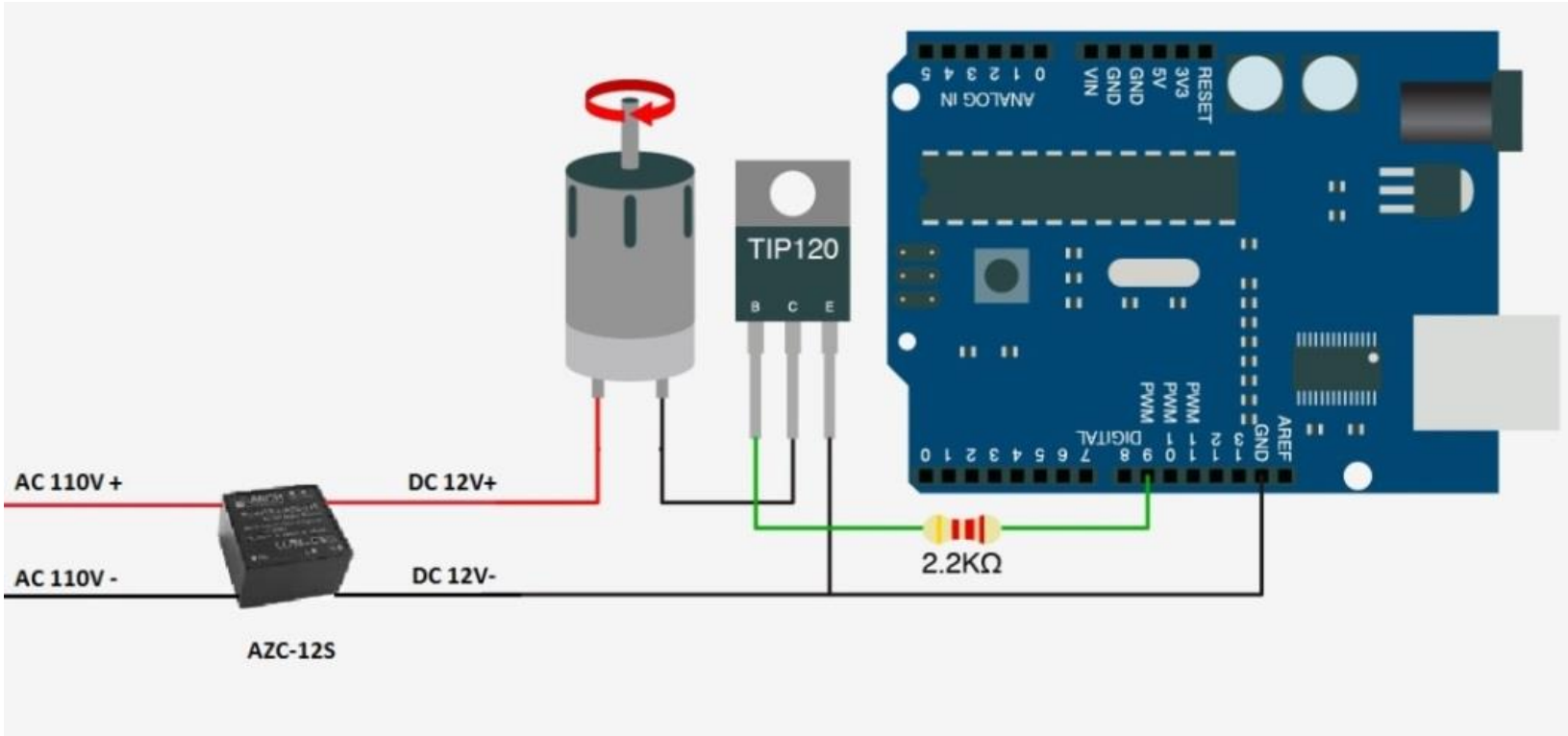
- 上半部為零交越點檢測電路
- 下半部為TRIAC(BTA12-600B)控制電路

風扇(直流電)

- 本研究所使用的風扇為12V的直流供電，所以須將外部電源110V交流電透過電源模組轉換成12V的直流電供風扇作使用。Arduino一樣使用PWM透過電晶體去調節風扇



風扇電路



- 使用AZC-12S電源模組轉換電源，而PWM藉由控制TIP120電晶體的導通來調節風扇

MySQL資料庫

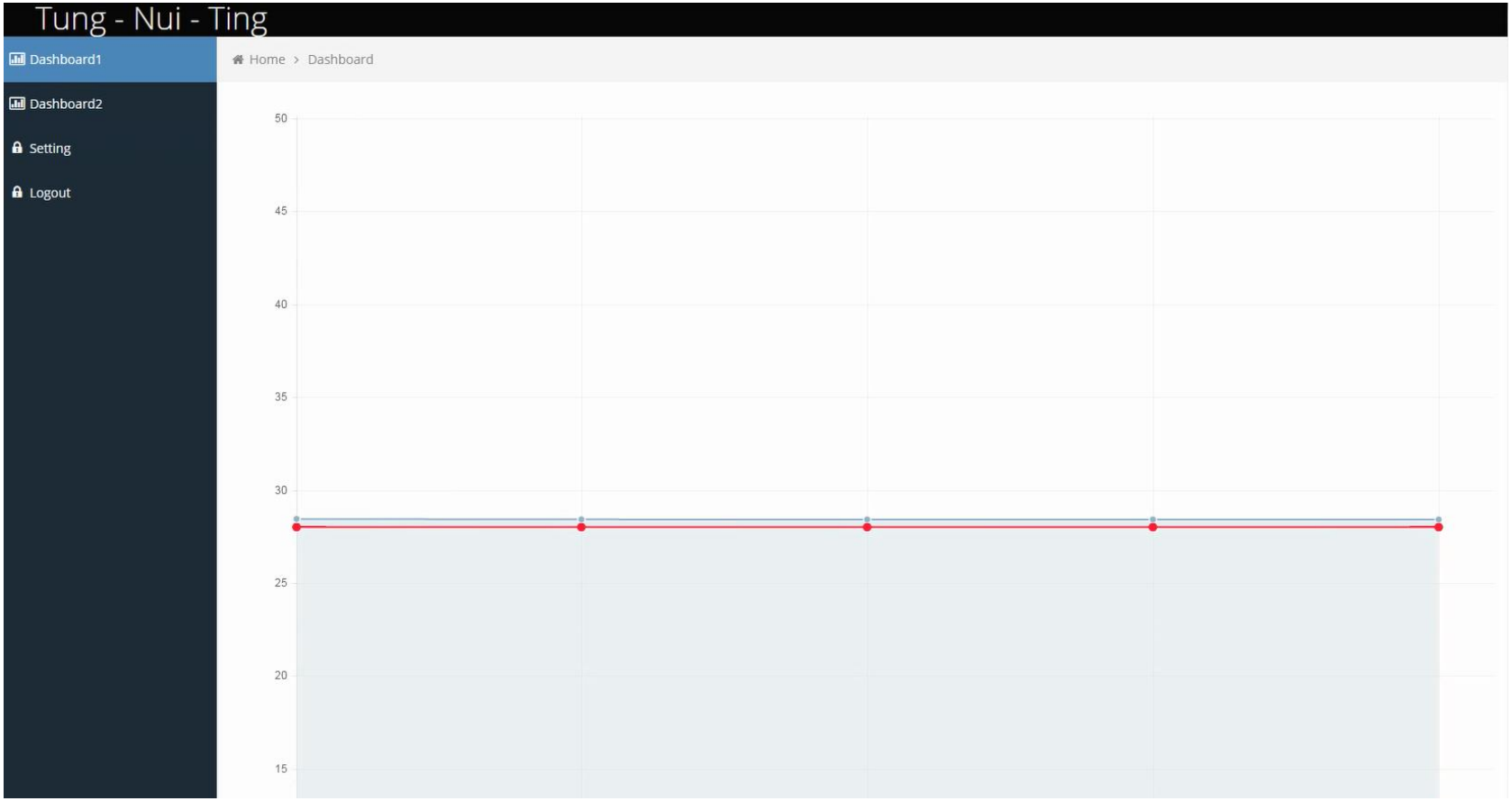
- 本研究配合使用者介面設計了一個架構，資料庫系統使用phpmyadmin使用者介面套件。
- 始MySQL更為方便管理，每個裝置都有相對應的溫度控制表、歷史資料與管理員帳密，方便資料的區分與整齊性。

使用者介面

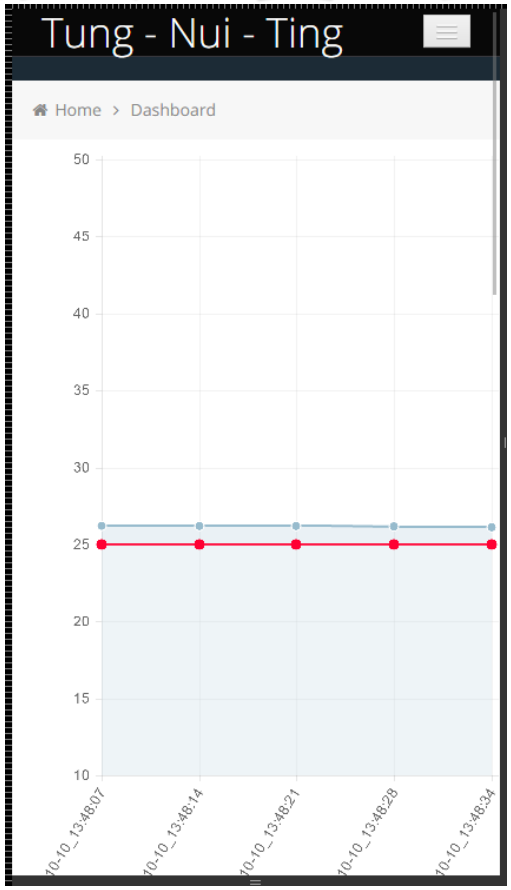
- 上方可切換位於不同位置的裝置，圖表將會顯示此裝置的溫度走勢圖，以及數字顯示當前溫度。

監控溫度(電腦)

滑鼠移到圖表點上會顯示出數值



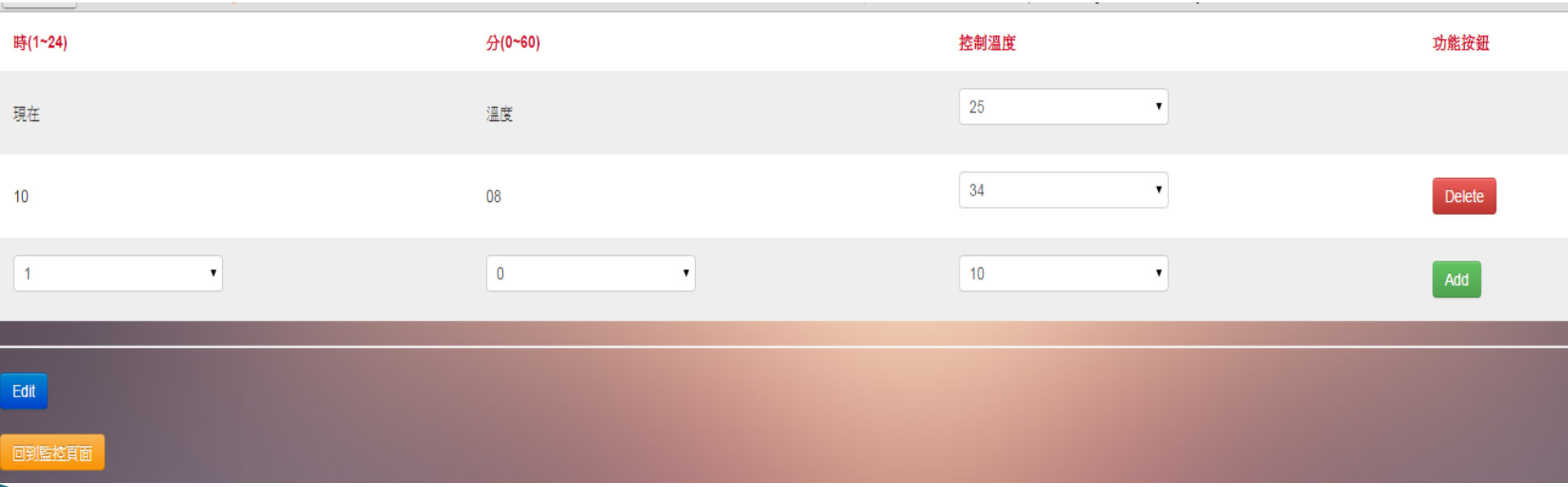
監控溫度(手機)



也可以隨手機螢幕大小跟著修正圖表顯示於手機螢幕的比例

設定恆溫溫度與預約排程溫度設定

- 點擊設定恆溫溫度後會來到下方這個頁面，最上方會依照使用者設定，顯示當前的恆溫值，讓使用者方便參考已設定的值進行修正或是微調



設定溫度與排程介面(電腦)

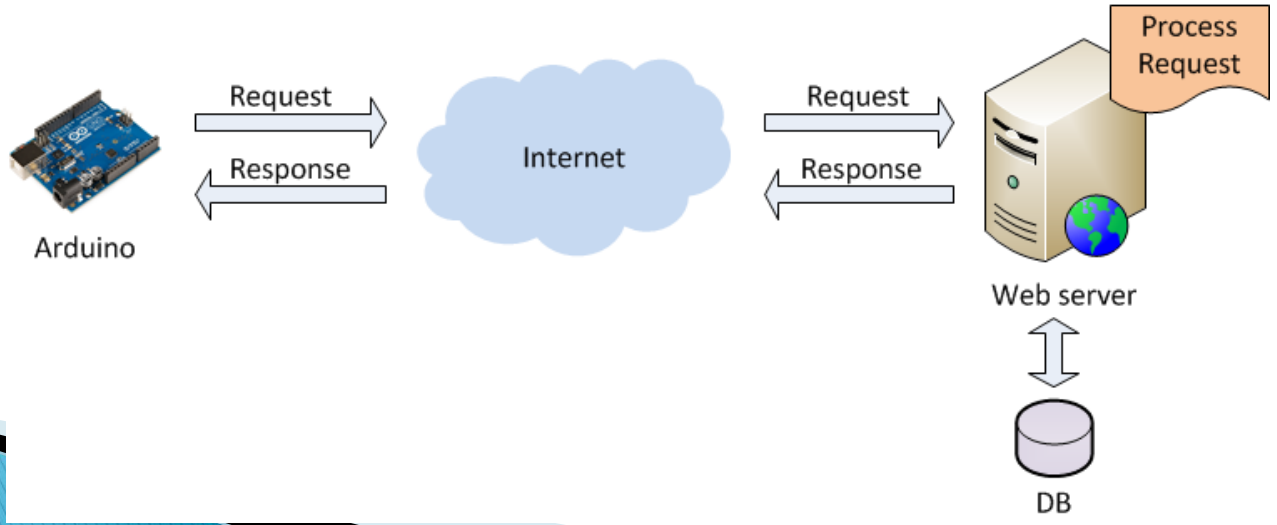
預約恆溫設定

- 此功能是提供使用者能夠事先預約所要的恆溫設定值，設定好改變溫度的時間以及想要改變的恆溫值，系統將在到達所設定之時間後自動變更設定值，達到一個智能預約的效果

設定溫度介面與排程(手機)

資料傳遞

- 此恆溫控制系統是一個不受任何環境所限制的完善系統，只要環境中有WiFi能提供裝置進行連線即可使用。Arduino對雲端伺服器提出傳遞請求(Request)，當伺服器收到請求後將會回傳請求回應(Response)給Arduino，使用此技術來達到資料傳遞的效果。



第五章 結果討論

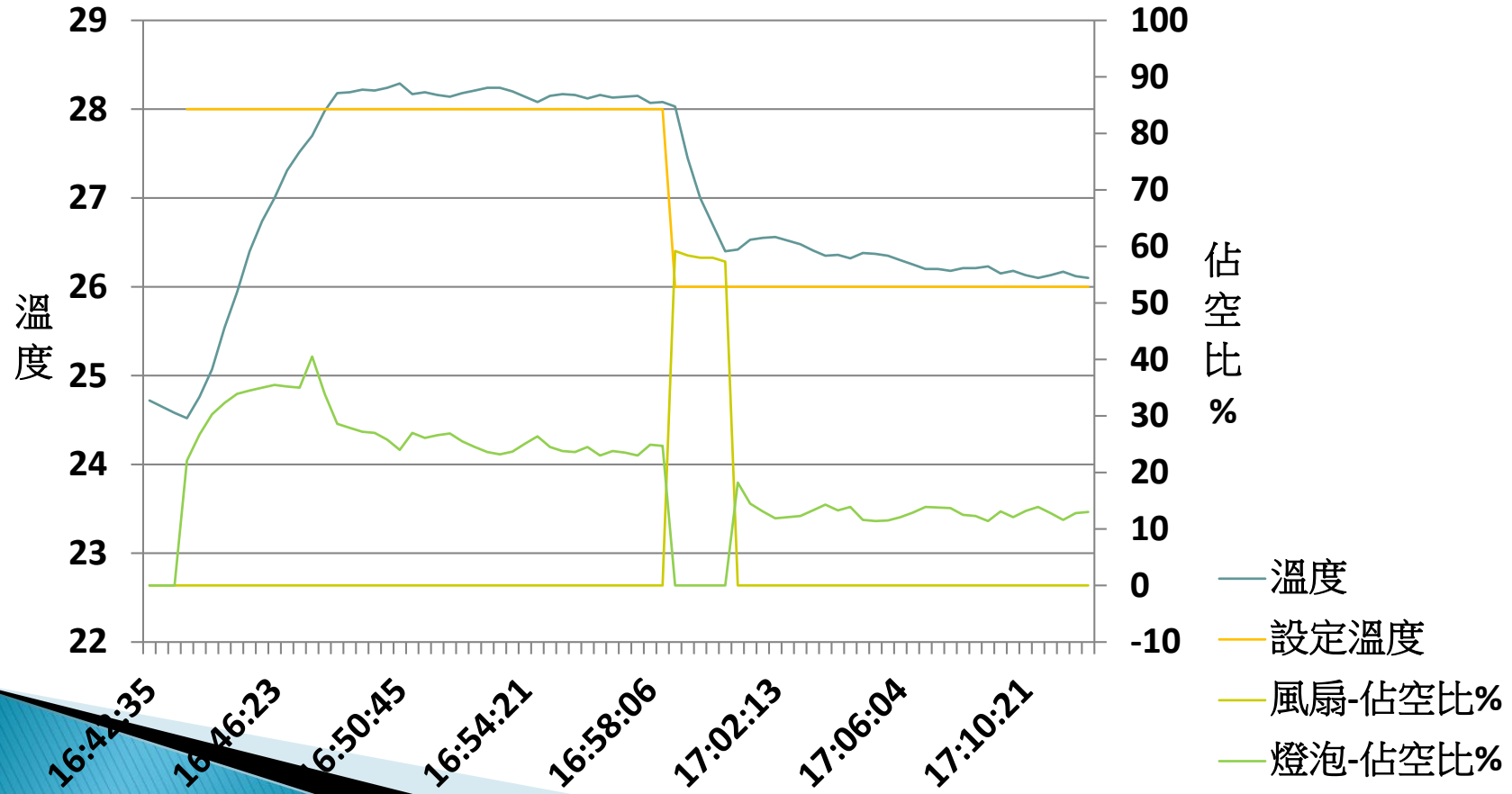
- 系統測試
- 功能展示

系統測試

- 為了測試系統效能並瞭解PID參數優化前後所造成的影響，我們選擇在相同環境以及條件相符的情況下所測出的兩組數據，起始溫度於24度以下，設定上升恆溫至28度，溫度上升的取樣時間為15分鐘，15分鐘後設定為下降恆溫至26度，同樣也是取樣時間15分鐘，讓兩者的條件相符，來比較出優化與否的差異

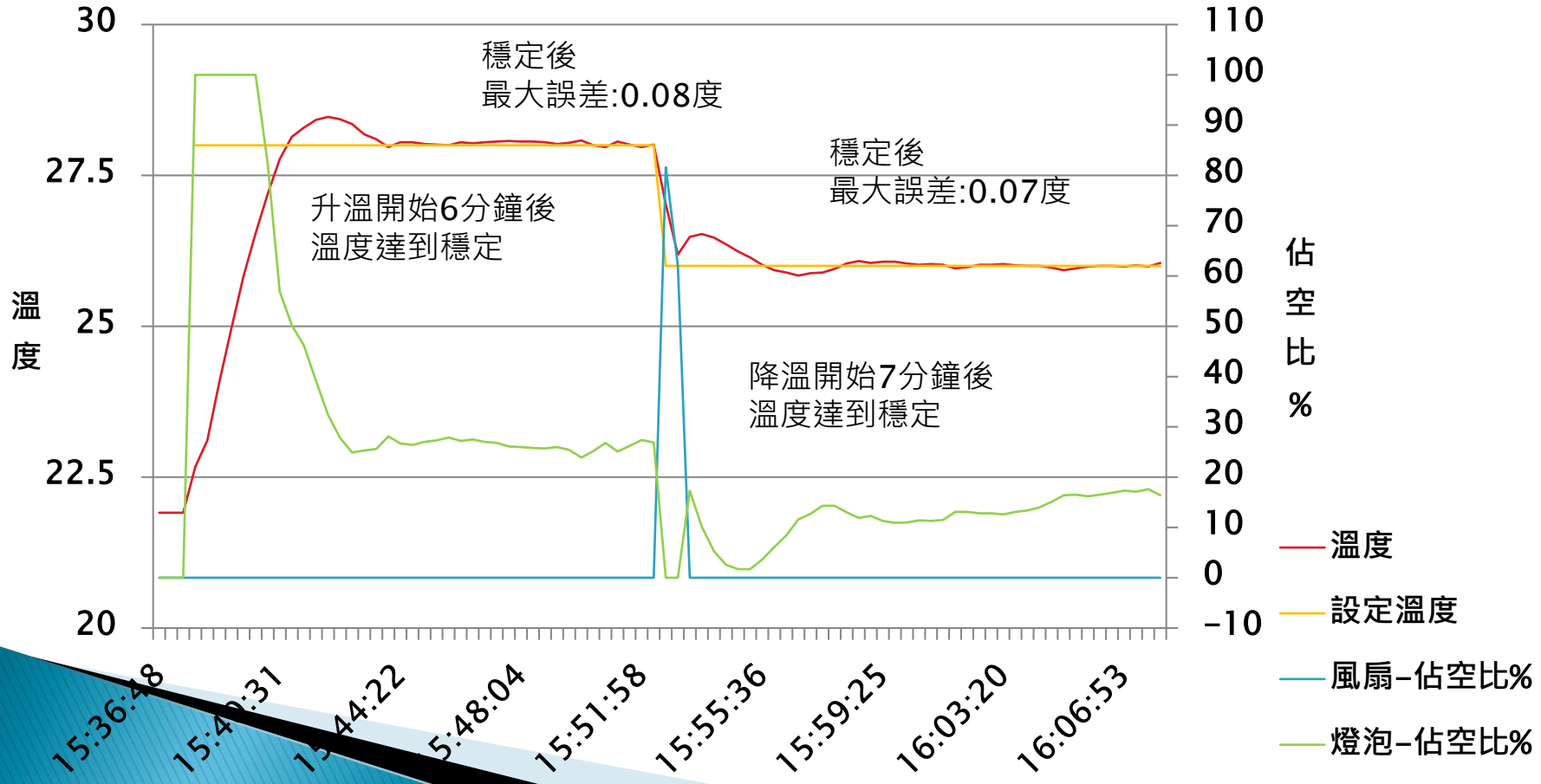
溫度數據(優化前)

- PID參數優化前，升降溫速度慢，趨近穩定速度過慢，誤差過大，穩定性不佳



溫度數據(優化後)

- PID參數優化後，能快速達到使用者的設定溫度，穩定性極佳，穩定後的誤差值不超過±0.1度



第六章 結論與未來展望

- 結論
- 未來展望

結論

- 本系統將PID控制器運算，PWM控制調節，作為Arduino程式核心的主軸，幾乎已經是個溫度智能調節的處理晶片。
- 接著讓原本只能單體處理的Arduino與雲端技術結合，使用者可以同時監控數台Arduino，並且個別進行設定，而且是透過簡單明瞭的控制頁面，讓就算不了解Arduino、不會程式語言的使用者，也一樣能享受此系統帶來的便利性，儘管是要進行細部設定也不需要擔心。

未來展望

- 現在系統的運算主要是靠各個裝置上的Arduino作處理，相對的Arduino效能不能太低。未來可以嘗試使用一台運算主機，將運算的部分集中到主機上，此方法減少了Arduino運算的需求，使用低階Arduino開發版即可達成，有機會大幅降低裝置成本。也因為使用運算主機，未來如果需要增加新的功能，或是將數據作進一步的應用，只要將功能寫入運算主機即可完成。

感謝各位評審的聆聽