高英高級工商職業學校

Kao Ying Industrial Commercial Vocational High School

教師專題研究（製作）報告

負載功率輸出大小控制-Arduino 及 APP 程式應用

老師姓名：林 俊 良 老師

科　　別：資訊科 科

中 華 民 國 １０４ 年 ２ 月

負載功率輸出大小控制-

Arduino 及 APP 程式應用

摘要

現在的社會中行動裝置普遍，每個人無時無刻都拿著智慧型手機在使用，從網路上的訊息了解，有很多人也用智慧型手機來遙控家電用品的應用，因此希望能藉由專題來探討應用智慧型手機的無線通訊來控制負載電路功率輸出大小，在這過程中，我們也將會結合 Arduino 的硬體應用及程式控制的設計、手機 APP 程式撰寫來對 Arduino 及 藍牙無線通訊作更深地瞭解，期望能增加新穎科技間的應用技能，藉著專題介紹科技生活的便利及教學上的延伸實用教材。並達到下列效果：

1. 在學習進階中，由單晶片微處理器電路，須自組零散週邊電路情形下，嘗試使用日漸普遍而整合性頗高的 Arduino 電路應用學習。14 個數位式輸出/入端，6 個類比式輸出/入端， USB 資料傳輸及連接上不同的電子裝置，例如 LED 燈、喇叭、馬達，然後再由控制器來驅動燈的亮滅、喇叭發聲、馬達運轉。
2. 手機上創作 App，開發設計工具與環境，有 Java 或者是 Objective C 的基礎就可產生有創意與主題性的 App。將學習如何以資訊雲端化與行動化，資通訊應用服務與數位內容的創新，實際體驗雲端應用的趨勢，不用攜帶書本又能在出外之間學習來獲得相應的學習效果。

目　錄

表目錄

圖目錄

壹、前言

一、製作動機

現在的學生大多擁有智慧型手機，科技通訊也一直在進步，結合科技和學生在校學習的課程需求；改善程式設計學習的實用性與應用科技硬體，希望利用智慧型手機 APP 程式來遙控家電用品，能藉由專題來探討智慧型手機的無線通訊來控制負載電路功率輸出大小，在過程中，學習 Arduino 的硬體各個接腳的功能與實際應用以及程式語法、控制方法的設計；在學習進階中，由單晶片微處理器電路，須自組零散週邊電路情形下，使用日漸普遍而整合性頗高的 Arduino 電路應用學習。另外利用手機 APP 程式撰寫工具環境開發來銜接控制 Arduino 及 藍牙無線通訊的學習並作更深地瞭解。所以將透通 PWM 脈波寬度調變，將類比信號轉換為脈波的技術，脈波的占空比會依類比信號的大小而改變轉速來實現負載功率輸出大小無線控制的目的。期望能增加新穎科技間的應用技能，藉著專題介紹科技生活的便利及教學上的延伸實用教材。

二、研究目的

本專題製作的主軸是利用行動裝置和 Arduino 微控器當成研發的工具，因智慧手機內有藍牙晶片，藉此可與 Arduino 微控器作無線傳輸溝通，就像遙控器一般的好用，當我們使用手機傳送指令時，藍牙晶片接收到指令便將指令傳送 Arduino 微控器，再以 Arduino 微控器去控制負載功率大小輸出而調整風扇轉速。

透過行動裝置 AppsduinoKitCmoreIO 副程式後，藍牙晶片接收到無線訊息的第二組數字，將會用 CmoreIO 來讀取第二組數字，當 CmoreIO 讀到 case 1 時，風扇將會開啟自然風轉速。當 CmoreIO 讀到 case 2 時，風扇將停止動作。當 CmoreIO 讀到 case 3 時，風扇將會啟動，並將風速訂在 50%。CmoreIO 讀到 case 4 時，風速則定速在 20%。CmoreIO 讀到 case 5 時，風速則定速在 35%。CmoreIO 讀到 case 6 時，風速則定速在 50%。CmoreIO 讀到 case 7 時，風速則定速在 65%。CmoreIO 讀到 case 8 時，風速則定速在 80%。CmoreIO 讀到 case 9 時，風速則定速在 100%。

三、製作架構

(一)、專題製作架構圖

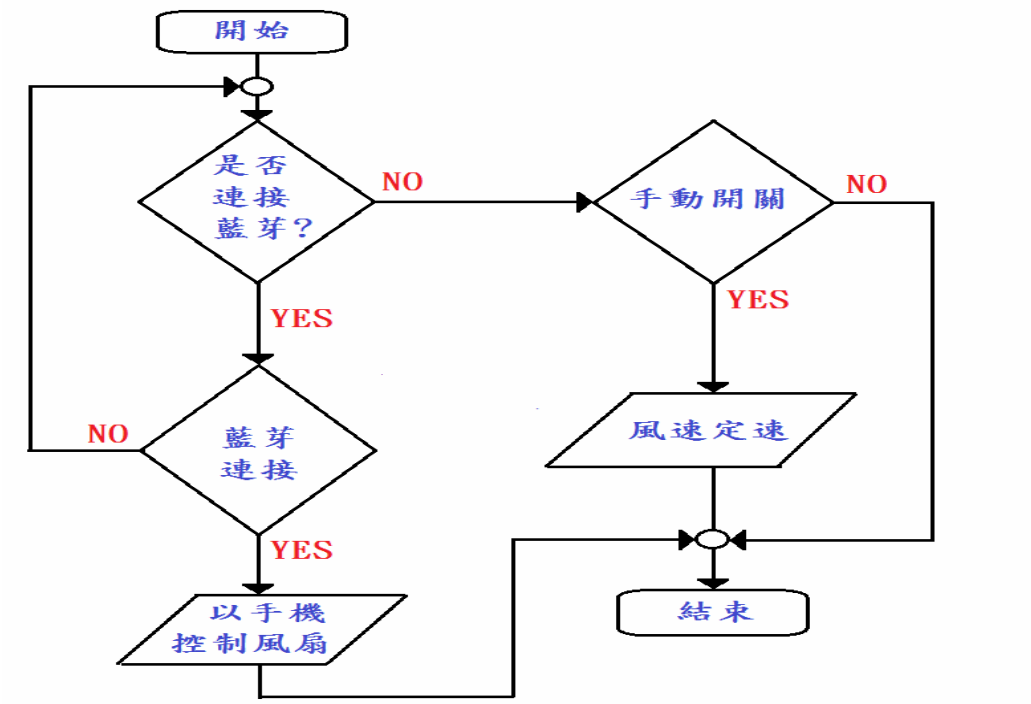

圖1 系統流程圖

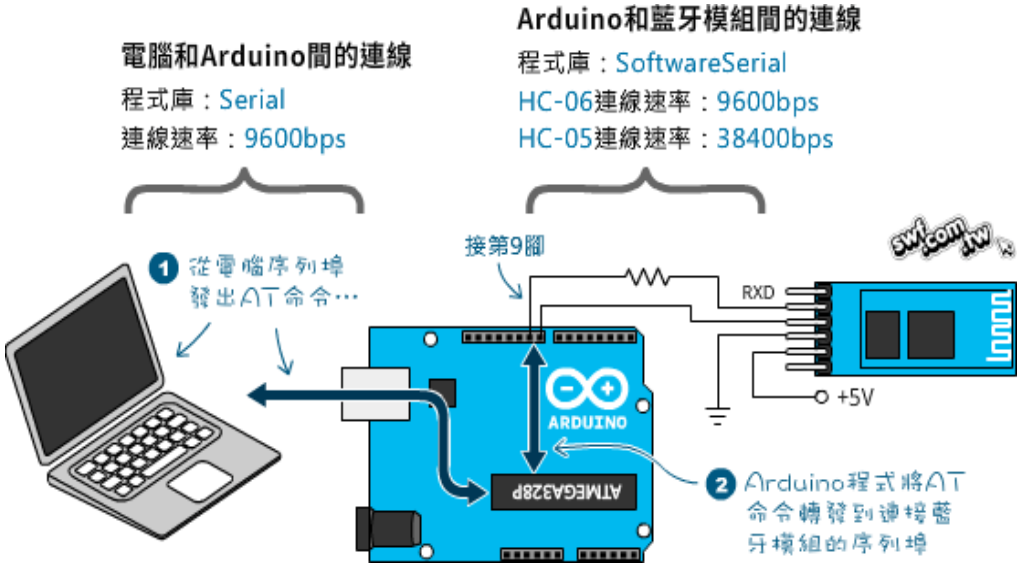

圖 2 專題製作架構圖

四、製作預期成效

本專題預計實做完成

(一)、實作 Arduino 開放原始碼的軟硬體平台，I/O 介面版，深入應用類似 Java、C 語言的 Processing/Wiring 開發環境。透過微處理機控制板的連結來接收如：感測器（ 聲音、光線 ）等傳送來的訊號控制 LED 燈、喇叭、馬達的應用。

(二)、實作手機上創作 App，認識 APP 開發設計工具與環境，產生有創意與主題性的 App。資通訊應用服務與數位內容的創新，學習如何以資訊雲端化與行動化之間學習來獲得相應的學習效果。

(三)、整合及實作手機 APP 程式撰寫來對 Arduino 及 藍牙無線通訊，增加科技產品間的結合應用技能，藉著專題介紹科技生活的便利及教學上的延伸實用教材效果。

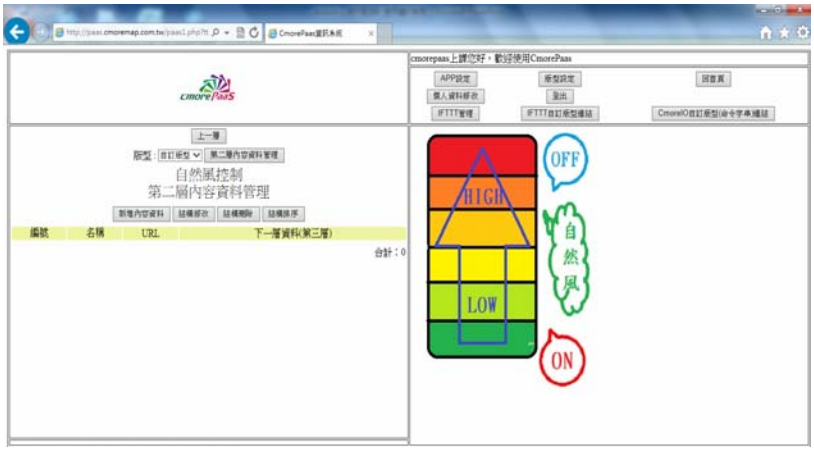

圖 3　手機 APP 程式撰寫

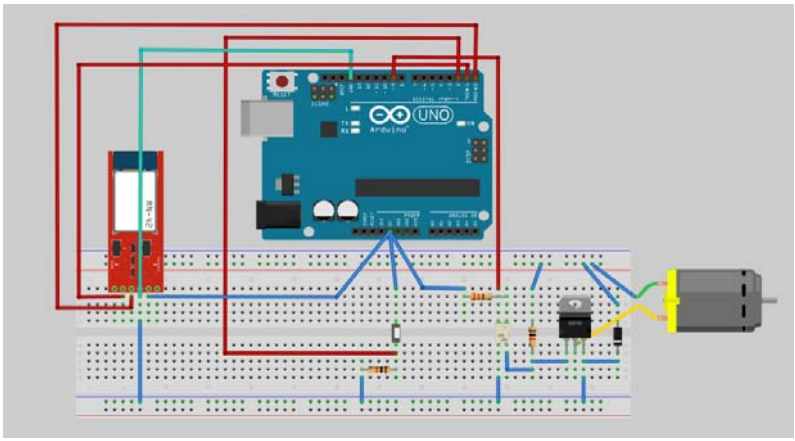

圖 4　Arduino 接線電路

貳、理論探討

一、研究分析:

(一)、Android 系統簡介:

網路巨人 Google 於 2007 年 11 月 5 日推出 Android，Android 是一套建構在 Linux 核心(Linux Kernel)之上的智慧型手機作業系統，Android 作業系統的核心屬於 Linux 核心的一個分支，具有典型的 Linux 排程和功能，除此之外，Google 為了能讓 Linux 在行動裝置上良好的運行，對其進行了修改和擴充。

Android 去除了 Linux 中的本地 X Window System，也不支援標準的 GNU 庫，這使得 Linux 平台上的應用程式移植到 Android 平台上變得困難。

2008 年，Patrick Brady 於 Google I/O 演講「Anatomy & Physiology of an Android」，並提出的 Android HAL 架構圖。HAL 以*.so 檔的形式存在，可以把 Android framework 與 Linux kernel 隔開，這種中介層的方式使得 Android 能在行動裝置上獲得更高的執行效率。這種獨特的系統結構被 Linux 核心開發者 Greg Kroah-Hartman 和其他核心維護者稱讚。

Google 還在 Android 的核心中加入了自己開發製作的一個名為「wakelocks」的行動裝置電源管理功能，該功能用於管理行動裝置的電池效能，但是該功能並沒有被加入到 Linux 核心的主線開放和維護中，因為 Linux 核心維護者認為 Google 沒有向他們展示這個功能的意圖和代碼。

2010 年 2 月 3 日，由於 Google 在 Android 核心開發方面和 Linux 社群方面開發的不同步，Linux 核心開發者 Greg Kroah-Hartman 將 Android 的驅動程式從 Linux 核心「狀態樹」(「staging tree」) 上除去。2010 年 4 月，Google 宣布將派遣 2 名開發人員加入 Linux 核心社群，以便重返 Linux 核心。

2010 年 9 月，Linux 核心開發者 Rafael J. Wysocki 添加了一個修復程式，使得 Android 的「wakelocks」可以輕鬆地與主線 Linux 核心合併。2011 年，Linus Torvalds 說:「Android 的核心和 Linux 的核心將最終回歸到一起，但可能不會是 4-5 年。」在 Linux 3.3 中大部分代碼的整合完成。]

在早期的 Android 應用程式開發中，通常通過在 Android SDK（Android 軟體開發包）中使用 Java 作為編程語言來開發應用程式。開發者亦可以通過在 Android NDK（Android Native 開發包）中使用 C 語言或者 C++語言來作為編程語言開發應用程式。同時 Google 還推出了適合初學者編程使用的 Simple 語言，該語言類似微軟公司的 Visual Basic 語言。此外，Google 還推出了 Google App Inventor 開發工具，該開發工具可以快速地構建應用程式，方便新手開發者。

Android 作業系統使用了沙箱（sandbox）機制，所有的應用程式都會先被簡單地解壓縮到沙箱中進行檢查，並且將應用程式所需的權限提交給系統，並且將其所需權限以清單的形式展現出來，供用戶檢視。

例如一個第三方瀏覽器需要「連接網路」的權限，或者一些軟體需要撥打電話，發送簡訊等權限。用戶可以根據權限來考慮自己是否需要安裝，用戶只有在同意了應用程式權限之後，才能進行安裝[。

由於，Android 在軟體版本授權上是採用 Apache Software License 2.0 的開放原始碼方案，因此在這個版權協議之下，智慧型手機製造商可免費地安裝 Android 作業系統至其生產製造的硬體之中，有效地降低了軟體的採購成本。

對於智慧型手機製造商來說，透過免費取得作業系統而降低軟體採購成本是一項很大的誘因，所以吸引眾多廠商投入生產、銷售或者研發 Android 作業系統的相關軟硬體產品與服務。

Android 以新秀之姿在短短五年的時間就有此成績，不只對於旗下合作的智慧型手機製造商具鼓舞作用，也會促使 Android 應用程式的開發者，投入更多時間與精力去開發兼具功能性與創新性的應用程式。

Android 在應用程式開發上，採取免費、開放的策略。開發者不僅可以免費地下載安裝 Android SDK (Android 的軟體開發工具包)進行應用程式的開發。

更重要地是，人們可以使用多數程式設計師所熟悉的 Java 程式語言進行應用程式的編寫。因為這兩個特點，促使為數眾多的 Java 程式設計師蜂擁至 Android 應用程式的開發行列。

(二)、什麼是 APP？

App 是「Application」的縮寫，就是手機「應用程式」「應用軟體」的意思。近年來 App 這個字眼充斥在我們生活中，原因是智慧型手機越來越普及。智慧型手機的 App 就如同電腦中的各種軟體，只是作業系統不同、內容相對簡單罷了。

「智慧型手機」(Smart Phone)這個說法主要是針對「功能手機」(Feature phone)，作為一個區隔而來，並非這個手機有多麼「智慧(Smart)」。

智慧型手機作業系統(OS)的四巨頭為 Apple 的 IOS、Google 的 Android 、Microsoft 的 Windows Mobile (最近推出 Win8 RT)、RIM 的 Black Berry ，其他還有 Nokia (Symbian) 、Palm 等。不同作業系統的應用程式無法互通，因為內部程式指令與運作方式各有不同。

傳統的功能手機的操作模式與內容是被手機廠限制固定的，不能由使用者隨意安裝移除軟體。而智慧型手機有提供一個平台空間，讓使用者可以隨意安裝和移除應用軟體 (App) ，就像電腦那樣。因此智慧型手機可定義為：擁有開放系統環境，允許第三方自行研發 App，並提供使用者自由運用的手機。

(三)、Arduino 系統簡介：

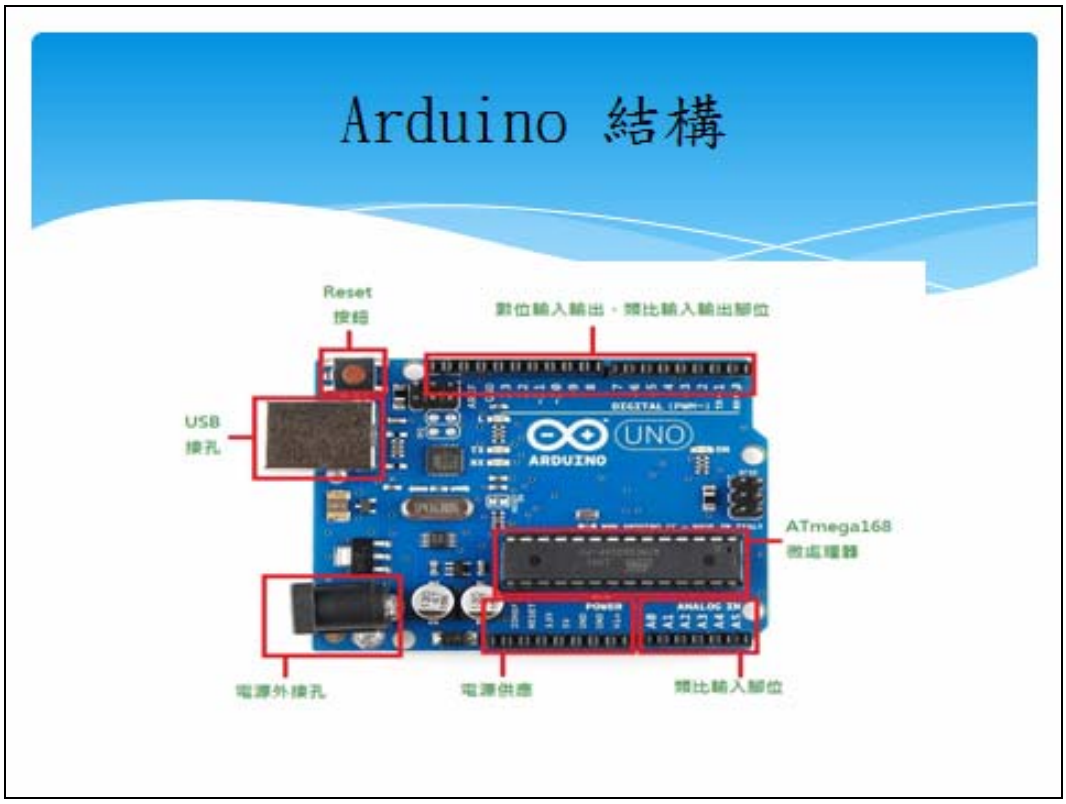

圖 5　Arduino 結構

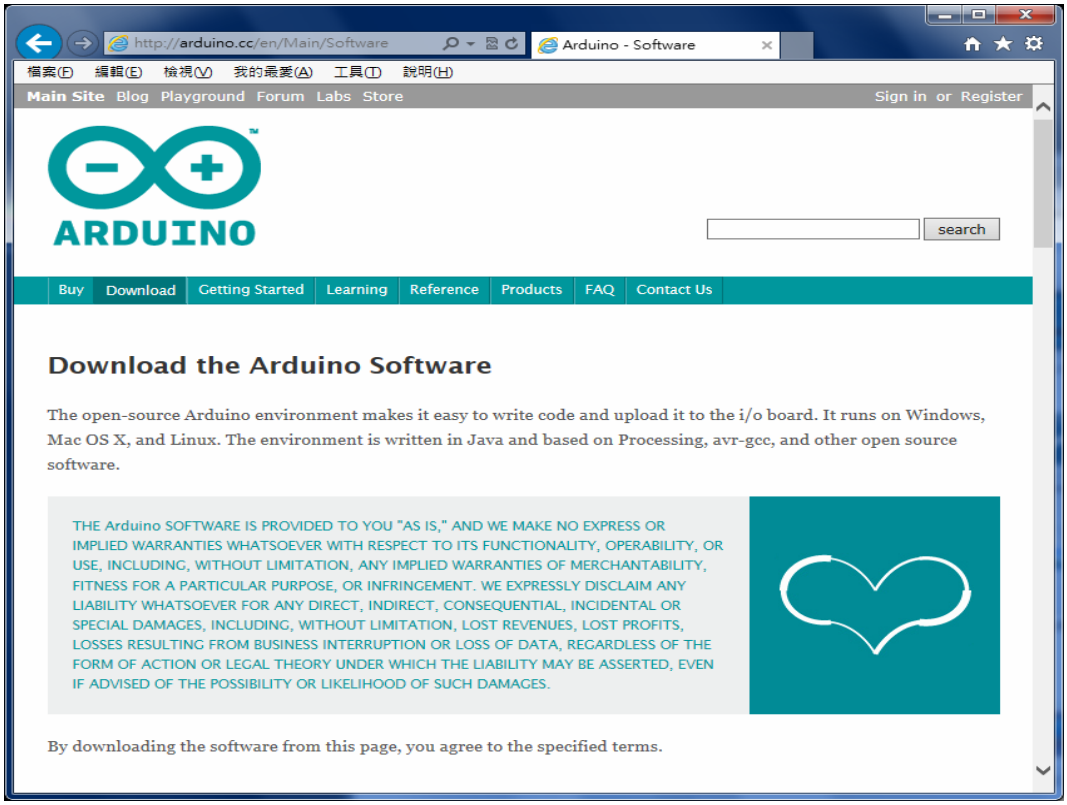

圖 6　Arduino 開發軟體

圖 7　Arduino 連接埠設定

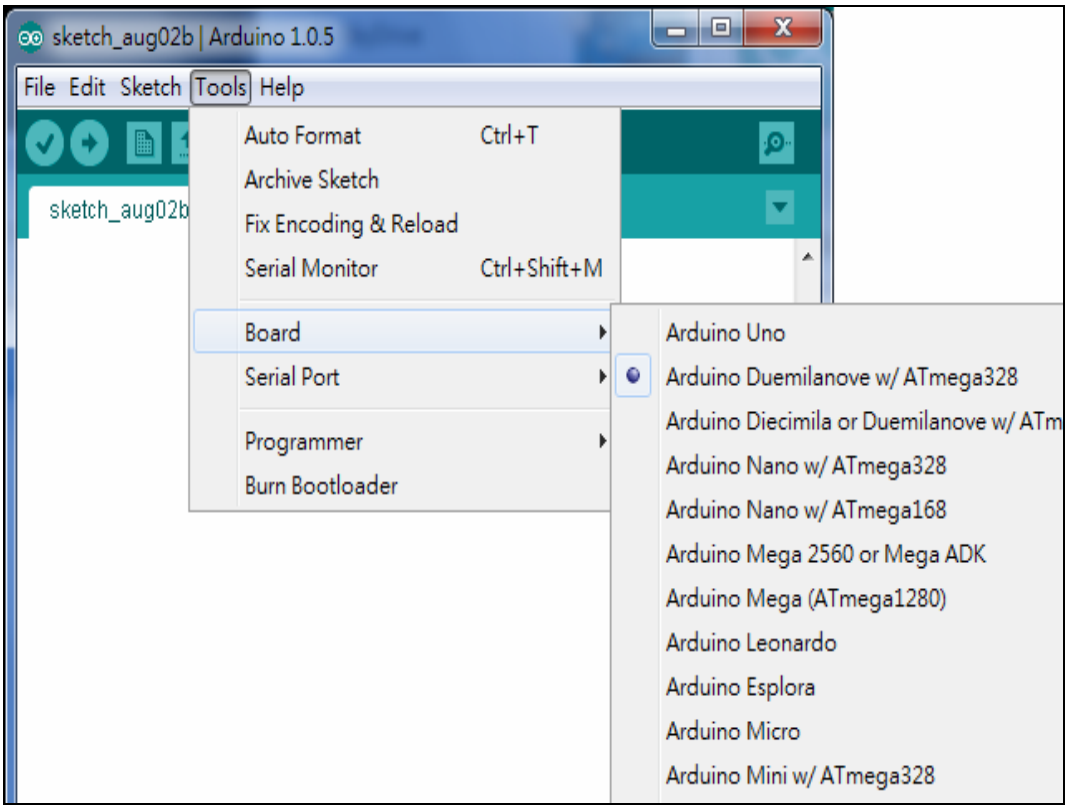

圖 8　Arduino 主機板和微處理器型號設定

表 1　Arduino 常用函式表

說　明	語　法	註　解
RS232/COM 設定包率	Serial.begin(9600);	9600 可依需求做改變
RS232/COM 輸出	Serial.print("Hello : ");	
RS232/COM 換行輸出	Serial.println("A");	
RS232/COM 輸入	char c = Serial.read();	
RS232/COM 資料輸入	if(Serial.available() > 0) { }	
延遲（毫秒）	delay(1); delay(1000);	延遲 1 毫秒 延遲 1 秒
延遲（微渺）	delayMicroseconds(1); delayMicroseconds(1000000);	延遲 1 微秒 延遲 1 秒
設定 Pin 為輸出	pinMode 5, OUTPUT);	設定第 5 隻腳為輸出
輸出 Pin 高電位	digitalWrite(5,HIGH);	設定第 5 隻腳為高電位
輸出 Pin 低電位	digitalWrite(5,LOW);	設定第 5 隻腳為低電位
設定 Pin 為輸入	pinMode(7, INPUT);	
讀取 Pin 電位	boolean b = digitalRead(7);	讀取第 7 隻腳(TRUE/FALSE)
輸出類比	analogWrite(5,255);	第 5 隻腳輸出，0-255 為一脈衝的時間(頻率不變)
讀取類比	analogRead(0);	讀取第 0 隻腳的類比訊號

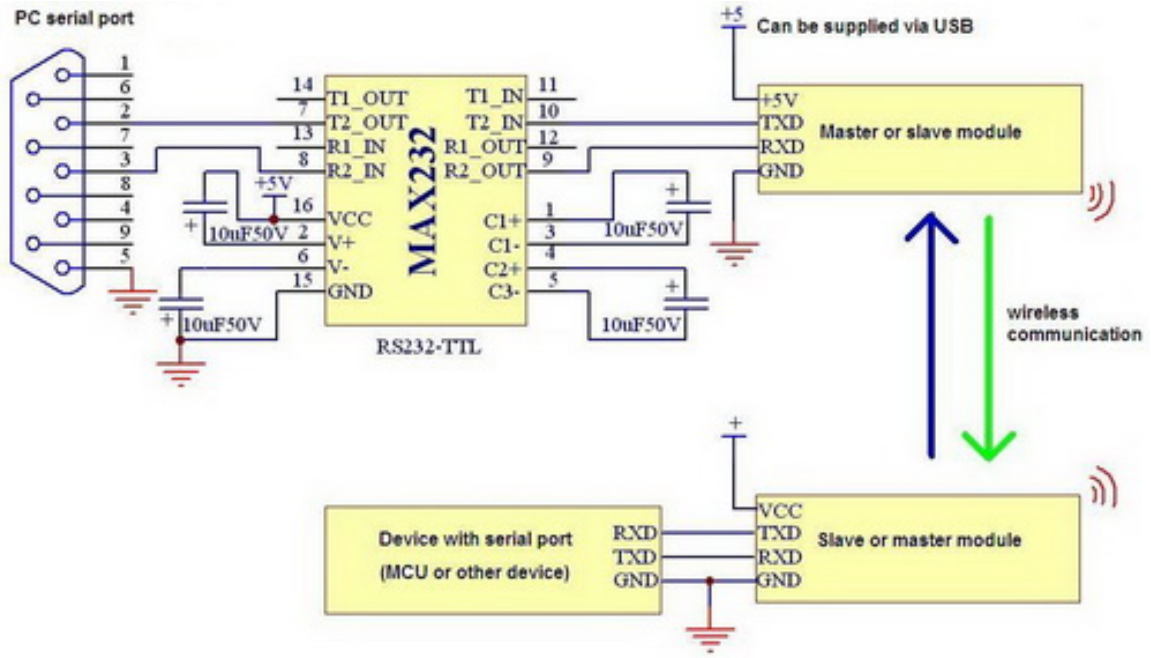

圖 9　藍牙電路

(五)、App Inventor2 環境建置

Google App Inventor2 為全雲端的開發環境，動作皆在瀏覽器上完成，開發設計資料全放在 MIT App Inventor2 伺服器上。

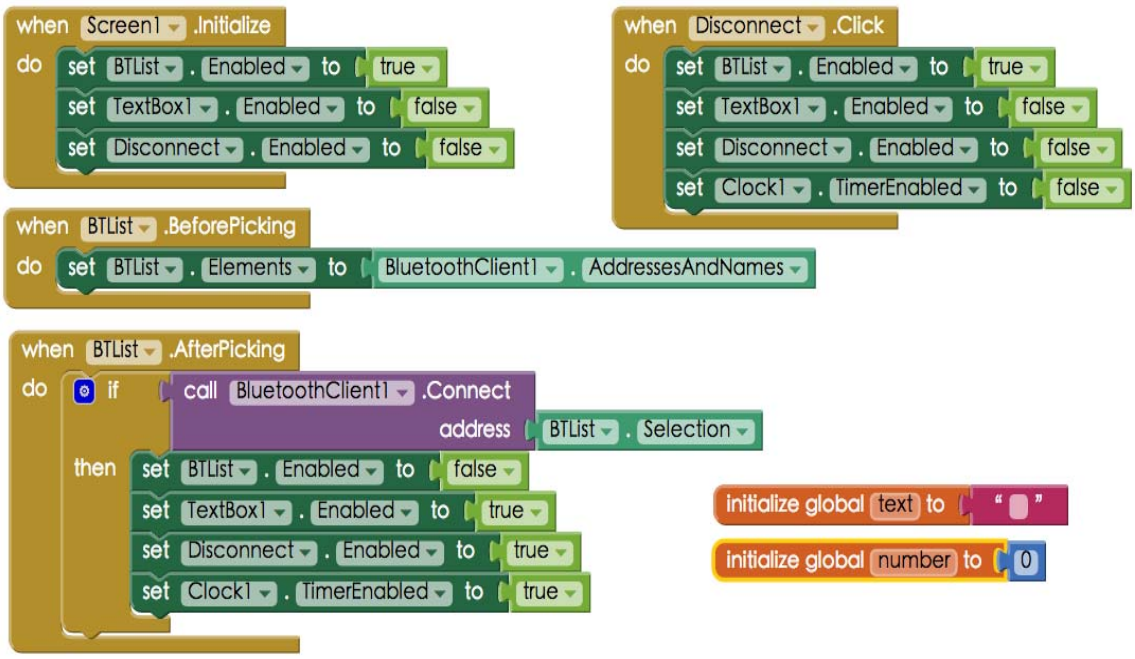

圖 10　Android AI2 藍牙通訊

表 2　Android AI2 函式

CheckBox 檢查方塊	ActivityStarter 活動啓動器
DatePicker 日期選取元件	BluetoothClient 藍牙用戶端
Image 圖片	BluetoothServer 藍牙伺服器
Lable 標籤	Web 網路
ListPicker 選取清單	LEGO MINDSTORMS
ListView 清單檢視	NxtDirectCommands NXT直接控制指令
Notifier 通知	
PasswordTextBox 密碼輸入	NxtColorSensor NXT顏色感測器
Slider 拖動條	NxtLightSensor NXT光感測器
Spinner 下拉式選單	NxtSoundSensor NXT聲音感測

圖 11　Android AI2 PWM 控制

參、專題製作過程或方法

一、製作設備及器材

表 3　專題製作使用儀器（軟體）設備一覽表

儀器（軟體）／設 備 名 稱	應　用　說　明
1.個人電腦	專題報告撰寫、電路圖繪製及專題成品測試
2.Google Chrome	操作 APP Inventor 2
3.WIN XP	作業系統
4.Arduino 控制板	微處理機控制板
5.藍牙通訊控制介面	無線傳輸通訊控制
6.Microsoft Office Word	製作專題報告
7.智慧行動手機或平板電腦	操作 Android 系統控制 APP 程式
8.數位相機	紀錄整個專題製作流程
9.噴墨印表機	列印專題相關資料
10.APP Inventor 2	程式設計
11. Arduino 工具軟體	程式設計
11. Cmore Cloud	程式設計

二、製作方法與步驟

Arduino 部分之軟體與硬體部分的製作，分別敘述如下:

(一)電路設計

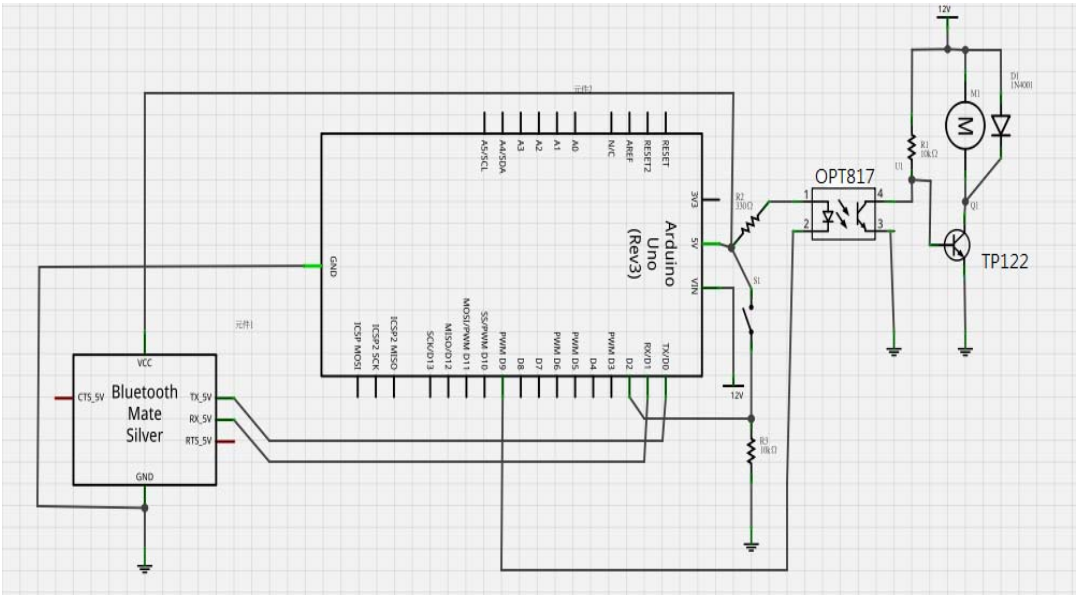

圖 12　電路圖

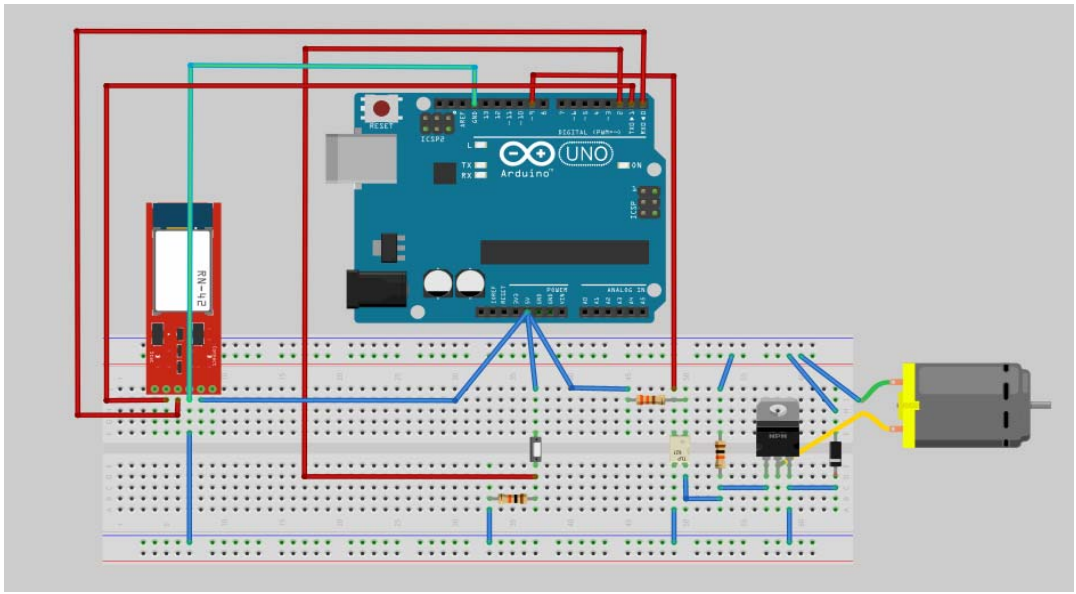

圖 13　接線圖

輸出電壓是由 on 和 off 時間的平均 值

output_voltage =(on_time / off_time) * max_voltage

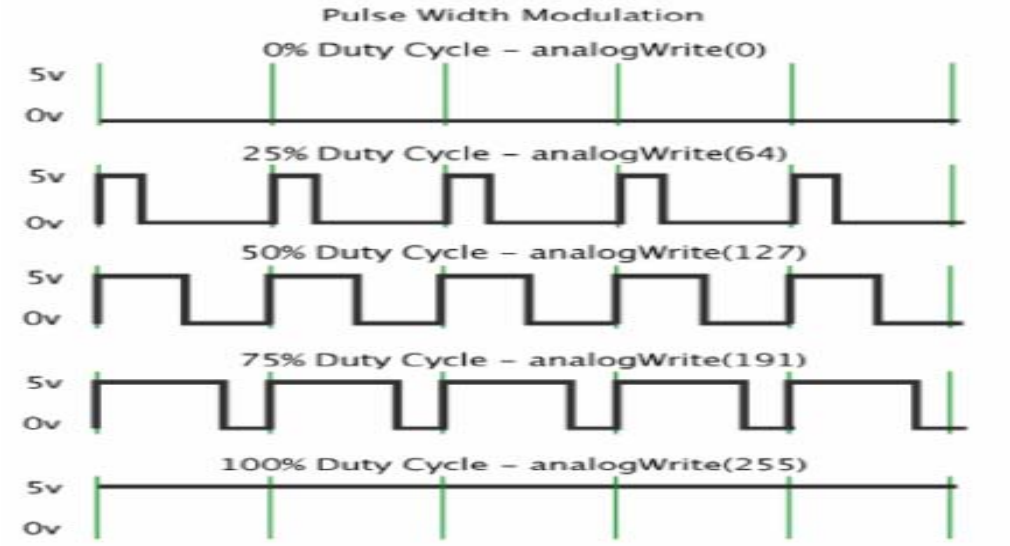

圖 14 PWM 輸出圖

OPT817 是顆光耦合器，它的特色是利用發射光以及光的接收來啟動電路，如果馬達部分的電路燒掉不會影響到 Arduino 硬體電路的部分。

OPT817 這顆光耦合器是利用 D9 來控制，假如 D9 輸出值為零，將會啟動光耦合器，那光耦合器第四支接腳將會倒通，便直接接地，故馬達不會啟動。假如 D9 輸出值為一，光耦合器將不會動作，那電晶體 B 腳將會觸發，馬達就會啟動。

手動開關是利用 D2 來控制。藍牙的部分，TX 是接到 DO 的腳位，RX 則是接到 D1 的腳位。

(二) 藍牙通訊資料

圖15　藍牙通訊

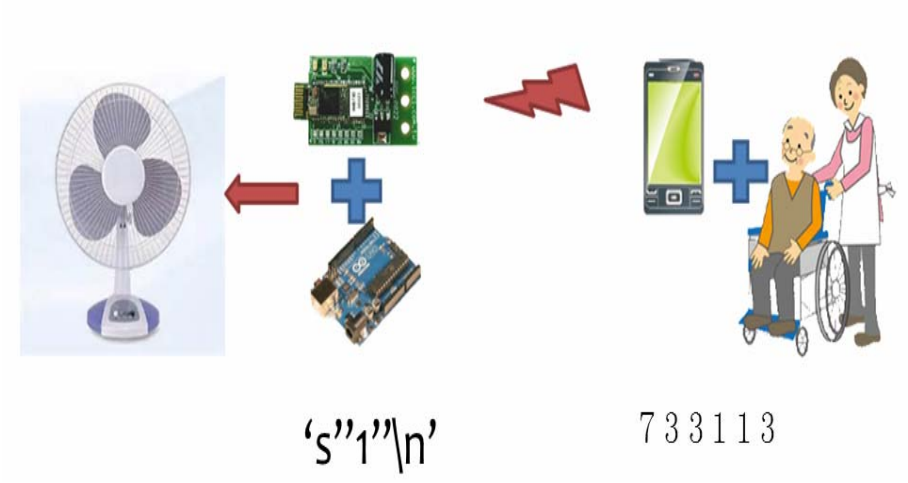

圖 16　藍牙通訊圖

使用智慧型手機去控制風扇，當按下不同的按鍵時，手機的藍牙將會傳送不同的指令給藍牙晶片接收，然後再以 Arduino 微控器去控制風扇。

從智慧型手機按下 ON 鍵時，智慧手機藉由藍牙晶片，發’733113’一串 16 進制的數字，它是 ASCII 碼，73 代表的就是‘s’，31 代表的字為’1’，13 代表的就是’\n’。Arduino 端之藍牙晶片接收後，由 Arduino 控制電風扇風量的變化。

藍牙部分：

當我們載入藍牙資料庫時，程式將會宣告一個藍牙元件，我們將設定藍牙鮑率為 57600。當藍牙接收到’s’時，我們將啟動 AppsduinoKitCmoreIO()副程式。

（三）風速控制程式：

當我們啟動 AppsduinoKitCmoreIO 副程式後，藍牙晶片將會接收到第二組數字，然後將會用 CmoreIO 來讀取第二組數字，當 CmoreIO 讀到 case 1 時，風扇將會開啟自然風轉速。當 CmoreIO 讀到 case 2 時，風扇將停止動作。當 CmoreIO 讀到 case 3 時，風扇將會啟動，並將風速訂在 50%。當 CmoreIO 讀到 case 4 時，風速則定速在 20%。當 CmoreIO 讀到 case 5 時，風速則定速在 35%。當 CmoreIO 讀到 case 6 時，風速則定速在 50%。當 CmoreIO 讀到 case 7 時，風速則定速在 65%。當 CmoreIO 讀到 case 8 時，風速則定速在 80%。當 CmoreIO 讀到 case 9 時，風速則定速在 100%。

（四）Cmore Cloud 開發步驟流程

4.1 CmoreCloud 開發步驟

Cmore Cloud 開發步驟共分為九步，詳如以下所示:

1. 登錄網站 CmoreCould 操作平台後，左邊的資料管理為工作區，右邊的手機部分為預覽區。

圖 17　CmoreCould 操作平台

2. 設定 IFTTT 管理之資料新增

IF 要設定無線裝置，無線裝置要設定 BT 藍芽的 IFTTT 資料。

圖 18　資料新增介面

傳送命令理要選擇 Hex 來傳送 16 進制參數，例如：傳送的參數為 733113，73 在 ASCII 碼裡是小寫 s、39 顯示 9、13 顯示 Enter，而 Arduino 讀取參數後會發出指令為 s9 Enter 使風扇運轉自然風。傳送命令採字串或 Hex，要依照藍芽控制端規格選擇，IF 一定要選擇為無線裝置，裝置 ID 一定要設定。

圖 19　資料新增介面

3. 新增按鈕

第三步驟為新增按鈕，依照步驟打上名稱，選項部分不用修改。

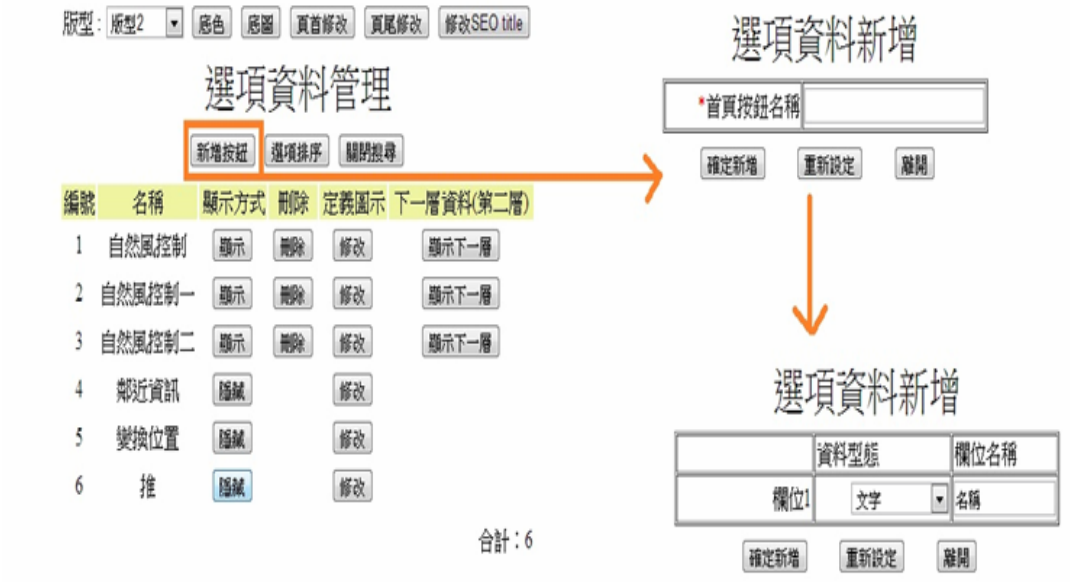

圖 20　新增按鈕

4. 第二層-自訂版型

第四步驟要在工作區點選顯示下一層，版型選擇自訂版型。

圖 21　版型設定

5. 上傳版型

上傳版型依照規格而設定，手機規格需要寬 320 點 X 高 425 點的圖片。

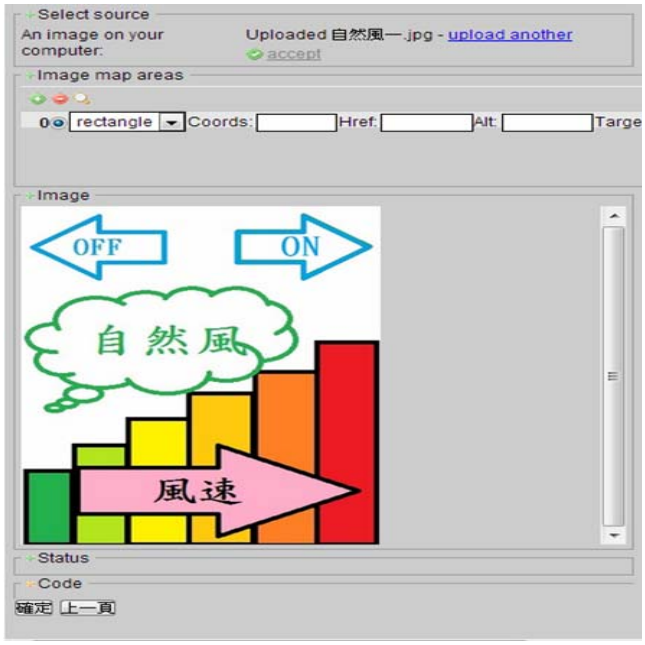

圖 22　上傳版型

6. 選取按鈕

第六步驟在介面圖上選取該控制的地方新增按鈕。

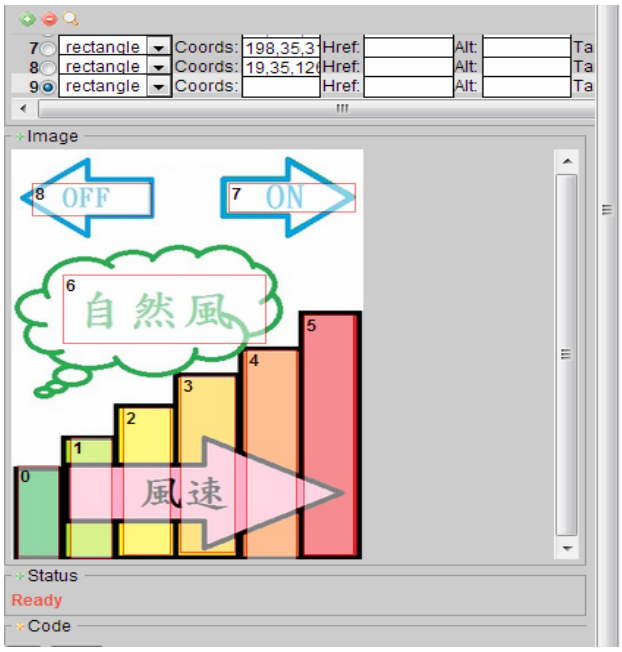

圖 23　選取按鈕

7. 開啟管理 IFTTT 命令連接

開啟 IFTTT 命令字串連結，以下圖表需要的是它的命令字串來傳送指令。

IFTTT自訂版型(命令字串)連結

無線裝置 All

編號	主題	IF	that	命令字串
9	風速6	無線裝置	傳送字串或命令	ifttt_android(9)
8	風速5	無線裝置	傳送字串或命令	ifttt_android(8)
7	風速4	無線裝置	傳送字串或命令	ifttt_android(7)
6	風速3	無線裝置	傳送字串或命令	ifttt_android(6)
5	風速2	無線裝置	傳送字串或命令	ifttt_android(5)
4	風速1	無線裝置	傳送字串或命令	ifttt_android(4)
3	ON	無線裝置	傳送字串或命令	ifttt_android(3)
2	OFF	無線裝置	傳送字串或命令	ifttt_android(2)
1	自然風	無線裝置	傳送字串或命令	ifttt_android(1)

圖 24　IFTTT 命令連接

8. IFTTT 自訂版型(命令字串)連結

把命令字串逐一的複製進去該框框的對照號碼的按鈕，要填入 Href，Alt 則顯示空白，設定好按確定

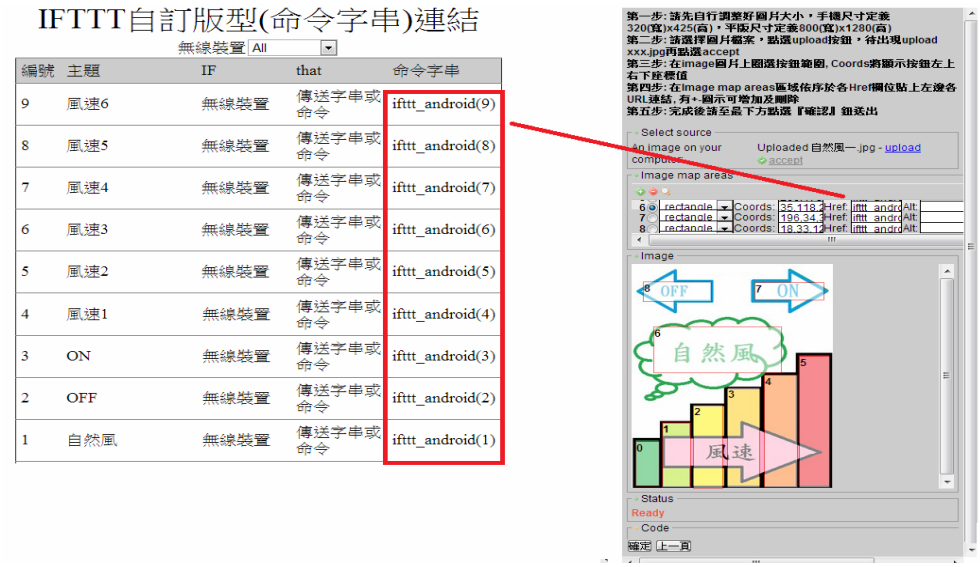

圖 25　IFTTT 自訂版型(命令字串)連結

9. APP 設定確定打包

在 APP 設定會顯示裝置的 QR Code。

圖 26　QR Code

三、專題製作

藍牙傳輸設定：

AppsduinoKitCmoreIO()副程式。

```
#include <MeetAndroid.h>  // include Android Bluetooth Library header
MeetAndroid AppsduinoBot;
//-------------------------------------------------
void setup()
{
  Serial.begin(57600);
   AppsduinoBot.registerFunction(AppsduinoKitCmoreIO, 's');
}
```

風速控制程式：

```
void AppsduinoKitCmoreIO(byte flag, byte numOfValues)
{
  int randomvalue = AppsduinoBot.getInt();
      CmoreIO(randomvalue) ; // execute the related function
}
void CmoreIO(int value)
{
  switch(value) {
    case 1 :
       flag=1; //開啟自然風
       break ;
    case 2 :
       outputValue=0;
       flag=0;
       break ;
```

```
    case 3 :
    if(outputValue ==0)
       outputValue=50;
       flag=0;
       break ;
case 4 :
       outputValue=20;
       flag=0;
       break ;
    case 5 :
       outputValue=35;
       flag=0;
       break ;
case 6 :
       outputValue=50;
       flag=0;
       break ;
    case 7 :
       outputValue=65;
       flag=0;
       break ;
    case 8 :
       outputValue=80;
       break ;
     case 9 :
       outputValue=100;
       flag=0;
       break ;
         }
       }
```

手動開關指令

當我們手動將開關調為 ON 時，此時的風扇將啟動並且保持風力在 80%。

```
buttonState = digitalRead(buttonPin);
  if (buttonState == HIGH)
 {       outputValue=80;   }
```

控制電路程式：

```
   Tx Cmd :
           's1' : toggle green led
           's2' : toggle Red led
           's3' : toggle Blue led
           's4' : play melody(Buzzer demo)
           'p100' ~ 'p800' : setup Cds Threshold Level
   Rx message :
            1. temperature
            2. CDS value
            3. Battery Voltage
            4. Cds Threshold
*/

#include <MeetAndroid.h>  // include Android Bluetooth Library header
#include <Timer.h>
#include <SoftwareSerial.h>
//SoftwareSerial mySerial(10, 11); // RX, TX


//-------------------------------------------------------------
//--------------- Android Bluetooth Test ------------
MeetAndroid AppsduinoBot;
```

```
//-----------------------------------------------
void setup()
{
  pinMode(led, OUTPUT);
  pinMode(buttonPin, INPUT);
  //----------------------------
  // Configure your bluetooth module to this baud rate : 57600 bps
  Serial.begin(57600);

  while(!Serial) ;
//  Serial.println("Goodnight moon!");

//  tick.every(1000, OneSecTimerHandler);// read temperature/humidity
}

//--------------- Main Program -----------------------
void loop()
{
  //-----------------------------------
//  tick.update(); // check timer event
 buttonState = digitalRead(buttonPin);
  if (buttonState == HIGH) {
    outputValue=80;
  }
  else {
  AppsduinoBot.receive(); // you need to keep this in your loop() to receive
events
  if(flag ==1)
   {
    outputValue+= Wstep;
```

```
    if(outputValue >100)
    {
      outputValue=100;
      Wstep=-2;
    }
    if(outputValue <20)
    {
      outputValue=20;
      Wstep=2;
    }
   }
 }
 digitalWrite(led, HIGH);   // turn the LED on (HIGH is the voltage level)
 delay(outputValue);                // wait for a second
 digitalWrite(led, LOW);    // turn the LED off by making the voltage LOW
 delay(100-outputValue);                // wait for a second

}
```

自然風指令

當 flag ==1 時，風扇將自動開啟為自然風，風速在 20%~100%之間，當風速為最大值 100%時，將每次-2 直到 20%為止，當風速為 20%時，將每次+2 到 100%為止。

```
if(flag ==1)
   {
    outputValue+= Wstep;
    if(outputValue >100)
       {
          outputValue=100;
          Wstep=-2;
       }
     if(outputValue <20)
       {
           outputValue=20;
          Wstep=2;
       }
    }
  digitalWrite(9, HIGH);
  delay(outputValue);
  digitalWrite(9, LOW);
  delay(100-outputValue);
 }
```

* 模式一

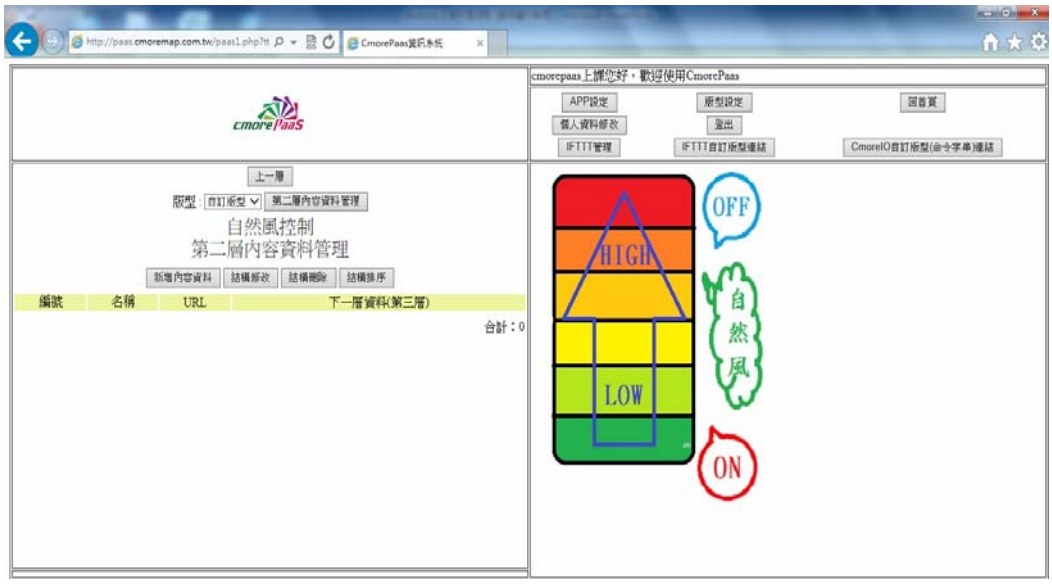

圖 27 手機功率操作模式一

* 模式二

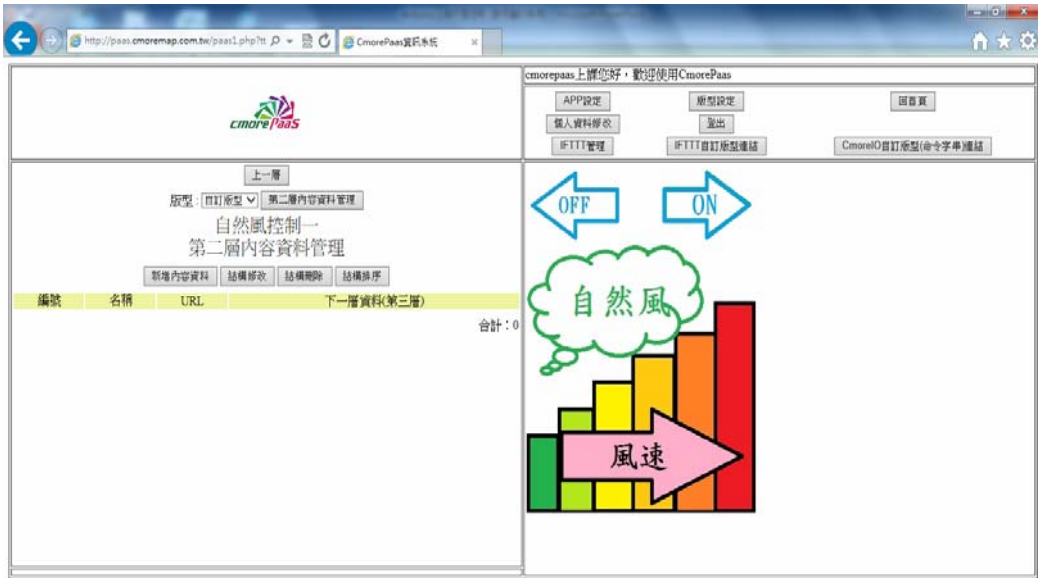

圖 28 手機功率操作模式二

* 模式三

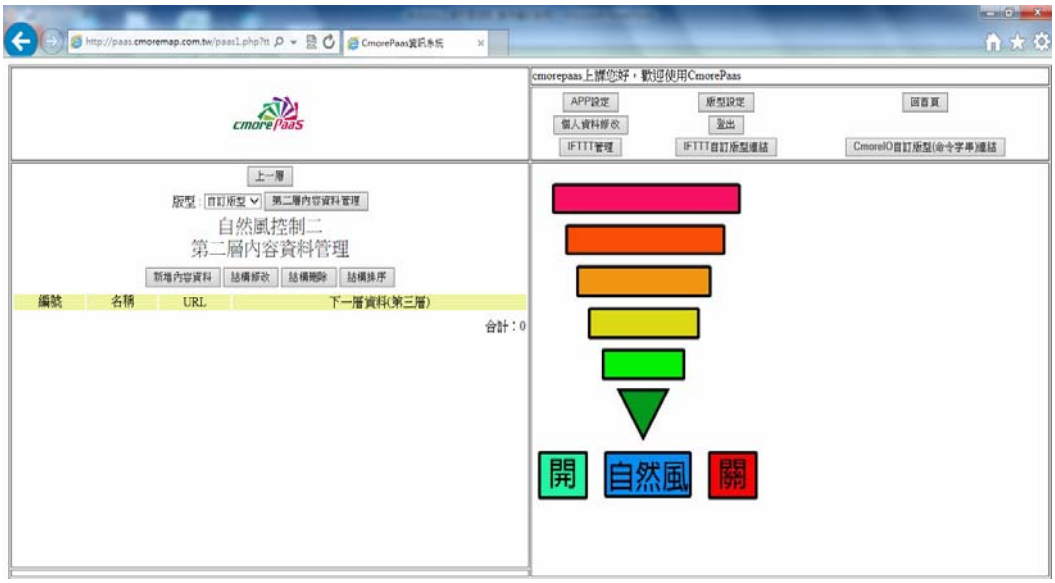

圖 29 手機功率操作模式三

四、製作成果

本專題使用 Android 智慧型手機操控程式，控制風扇的開關及轉速，智慧型手機之操作畫面。智慧型手機上所示有 9 個控制按鍵，透過藍芽晶片接收並傳遞至 Arduino 微控板。藍芽晶片安裝到電路板上，藍芽晶片接收到指令再傳送給 Arduino 微控板，再利用 Arduino 微控板去控制風扇的開關及轉速。

圖 30 手機實體版面

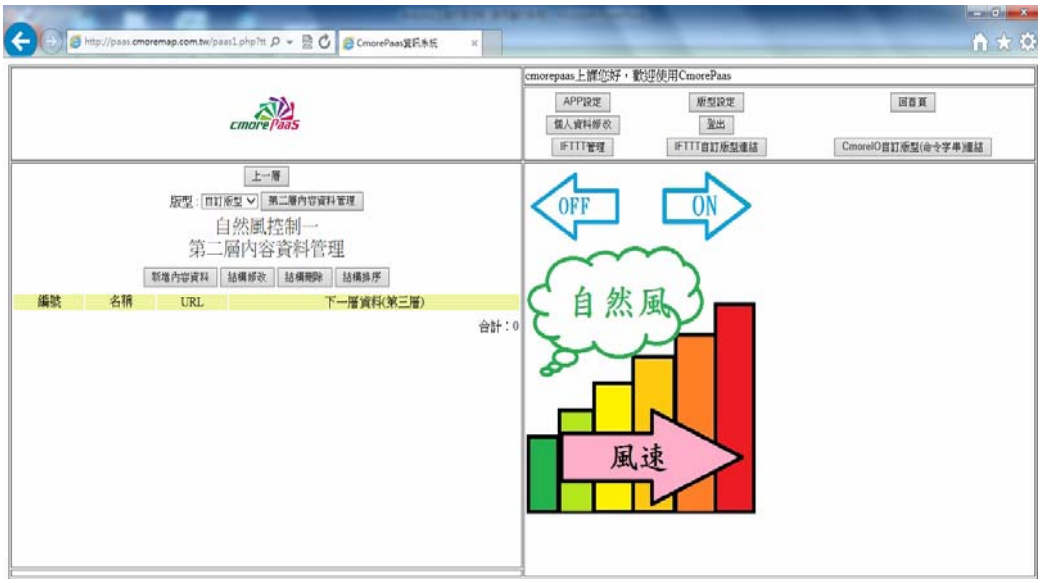

圖 31 手機設計版面

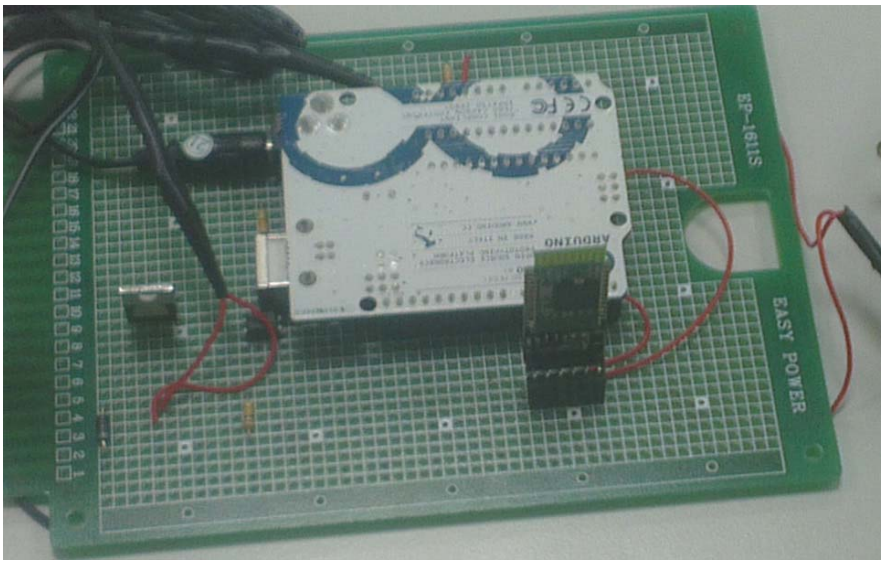

圖 32 實體電路

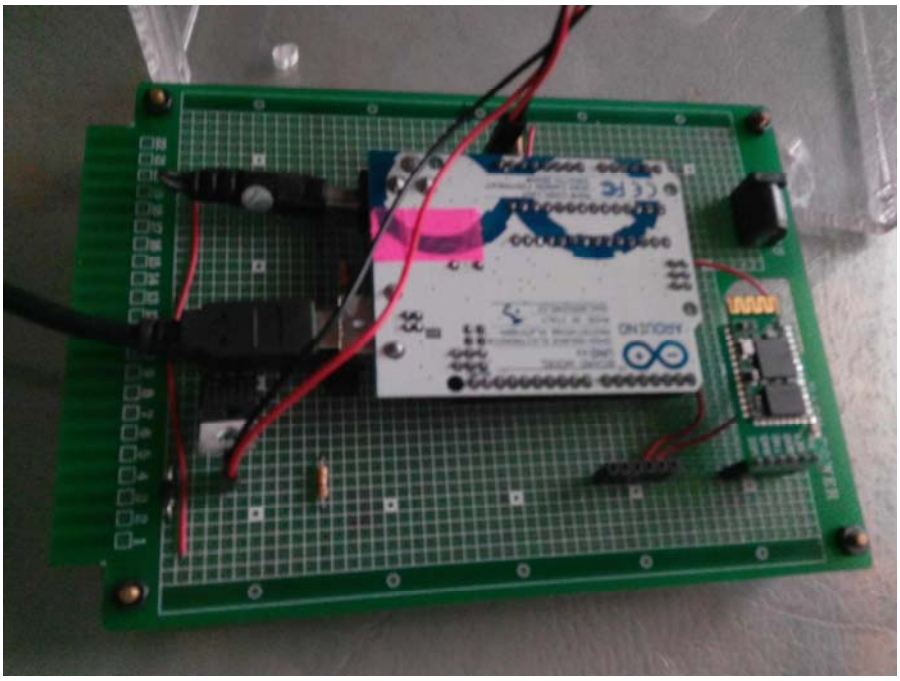

圖 33　Arduino 之成果

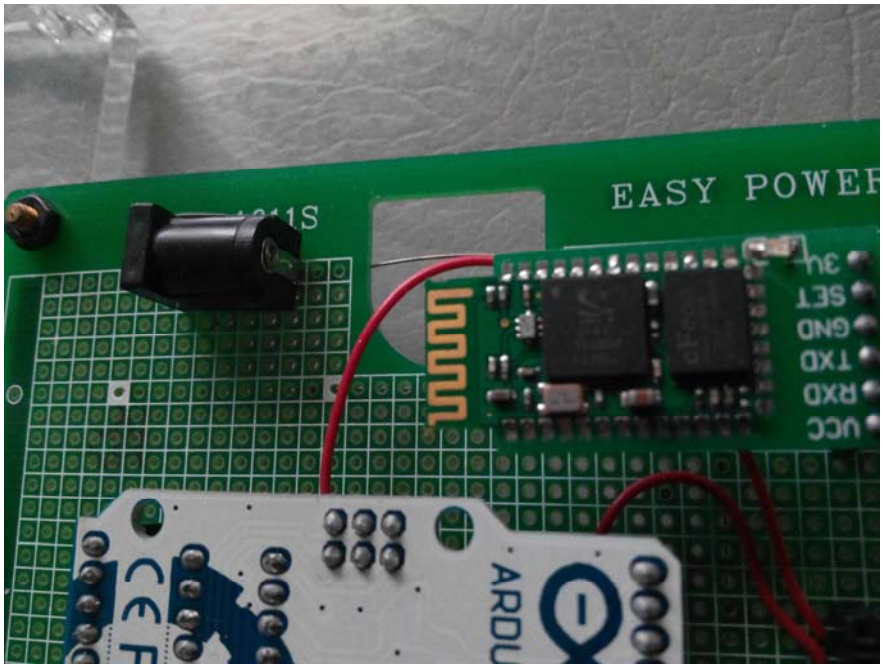

圖 34 **藍牙傳輸**

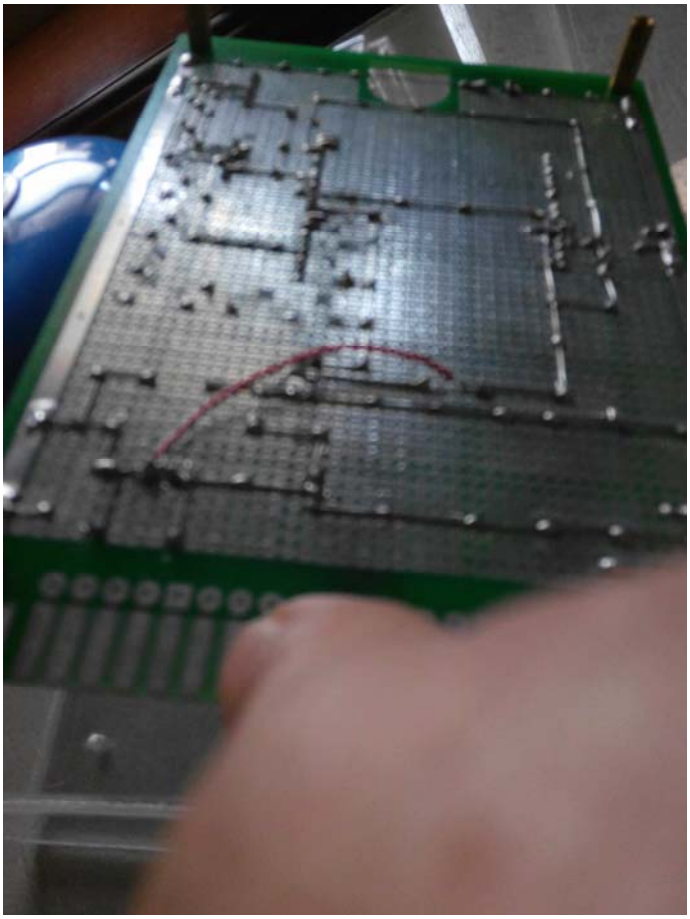

圖 35　電路焊接

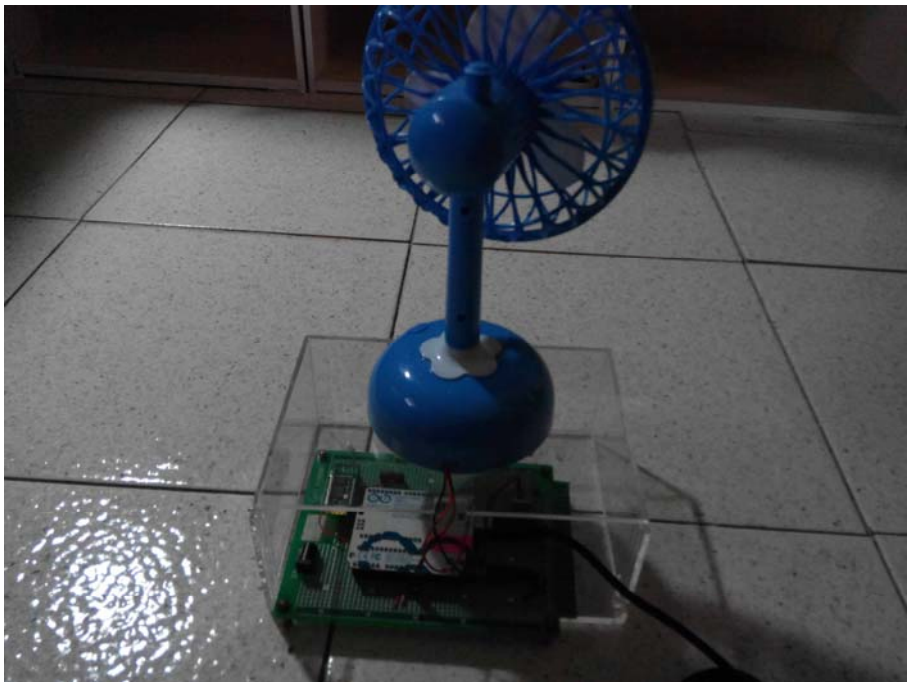

圖 36 負載功率輸出成品

肆、製作結論與建議

一、結論

本專題是使用智慧型手機來控制自然風扇，當我們開啟手機內 App 程式後，手機的藍芽模組便會跟電路上的藍芽晶片作連接，然後將可以進一步進到風扇操控介面，此時，當我們按下任何按鍵，手機藍芽將會傳送不同的指令到藍芽晶片，然後再傳送到 Arduino 微控器，讓 Arduino 微控器去控制風扇。

學習 Arduino 的硬體各個接腳的功能與實際應用以及程式語法、控制方法的設計；在學習進階中，由單晶片微處理器電路，須自組零散週邊電路情形下，使用日漸普遍而整合性頗高的 Arduino 電路應用學習。另外利用手機 APP 程式撰寫工具環境開發來銜接控制 Arduino 及 藍牙無線通訊的學習並作更深地瞭解。

二、建議

設計智慧型手機 APP 時，選用開發工具項目繁多易使設計難度增加；尤其是介面及控制指令語法的應用不熟悉，以致錯誤產生及進度緩慢。在設定藍牙模組的連線速率上初始以 9600 色鮑率作測試連線，才不致發生無法找到藍牙模組的通訊失敗情形，而未達到遠端的操控。

參考文獻

1. 鄧文淵，2014，手機應用程式設計超簡單－App Inventor 2 初學特訓班，臺北市：碁峰出版社。

2. 體感創作 DNA-『想』與『做』間的拔河及結合，作者：陳光雄版次：2011 年 11 月版

3. http://elesson.tc.edu.tw/md221/pluginfile.php/4151/mod_r

   esource/content/1/arduino.pdf-Arduino 基礎教學

4. http://www.embeda.com.tw/tw/?p=2874 專題繪圖軟體 Fritzing 教學

5. http://tw.myblog.yahoo.com/w047/article?mid=3909- Fritzing 元件教學

網址 appinventor.mit.edu/