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

Arduino 常用感測模組控制

老師姓名：林 俊 良 老師

科　　別：資　訊　科

中　華　民　國　105　年　1　月

Arduino 常用感測模組控制

摘要

現在學生在微電腦實習課程中已經由傳統 8051 單晶片的電路製作控制及編寫組合語言的控制程式；普徧改為以 Arduino 控制板上來作各種電子電路模組的控制，例如 調控 LED 閃爍、控制各式開關元件、控制喇叭產生各種音調樂曲、控制馬達正反轉和速率、使用溫濕度感測器量測外界環境溫濕度、利用紅外線發射與接收器作為距離測量，還有利用 WiFi、 Bluetooth 等各種通訊模組作傳輸及無線控制等，就能利用 Arduino 做出自動控制應用，例如使用可變電阻控制燈光的明暗、光敏電阻作為室內照明的情境調控變化系統、利用溫度感測器調控電風扇的啟動和停止運轉、控制馬達的轉速及角度、利用紅外線組成遙控家電開關動作、利用伺服馬達製作機械手臂的操控，以及避開障礙的自走車、循線自走車或四軸飛行器等。再結合行動裝置的智慧型手機或平板電腦;即可用作更廣泛家電用品的遙控應用，因此希望能藉由專題來探討由基礎常用的感測模組控制,進而深入到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。

目　錄

表目錄

圖目錄

壹、前言

一、製作動機

學生在微電腦實習課程經由傳統 8051 單晶片的電路製作控制及編寫組合語言的控制程式;普徧改為以 Arduino 控制板來作各種電子電路模組的控制,例如 調控 LED 閃爍、控制各式開關元件、控制喇叭產生各種音調樂曲、控制馬達正反轉和速率、使用溫濕度感測器量測外界環境溫濕度、利用紅外線發射與接收器作為距離測量,還有利用 WiFi、 Bluetooth 等各種通訊模組作傳輸及無線控制等。

因此基本的單一模組功能及程式的控制撰寫技能必須紮實實作;方能在日後利用 Arduino 做出自動控制應用,例如使用可變電阻控制燈光的明暗、光敏電阻作為室內照明的情境調控變化系統、利用溫度感測器調控電風扇的啟動和停止運轉、控制馬達的轉速及角度、利用紅外線組成遙控家電開關動作、利用伺服馬達製作機械手臂的操控,以及避開障礙的自走車或四軸飛行器等。

因此希望能藉由專題來探討由基礎常用的感測模組個別單一和組合實作控制,進而深入到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、避開障礙的自走車循線自走車或四軸飛行器等,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。

二、研究目的

本專題製作目的是利用 Arduino 常用感測模組如大磁簧感測模組、三色 LED 模組、震動開關模組、光敏電阻模組、按鍵開關模組、傾斜開關模組、水銀開關模組、光遮斷器模組、5V 繼電器模組、無源蜂鳴器模組、有源蜂鳴器模組、溫度感測器模組、避障感測器模組、尋線感測器模組、紅外感測器接收模組、七彩自動閃爍 LED 模組作實作以獲得基本控制技能。進而深入到可以結合其它課程所學再應用智慧型手機的無線通訊來控制四軸飛行器、負載電路功率輸出大小控制馬達轉速、製作避開障礙及循線的自走車等。

三、製作架構

(一)、專題製作架構圖

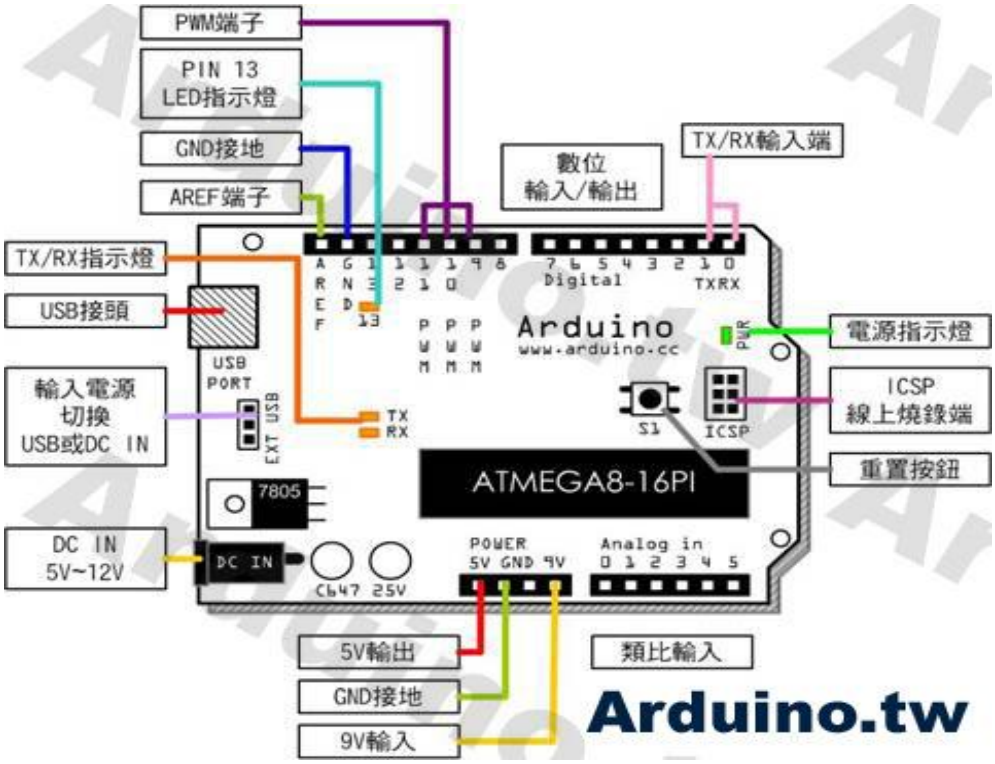

圖1 控制板架構圖

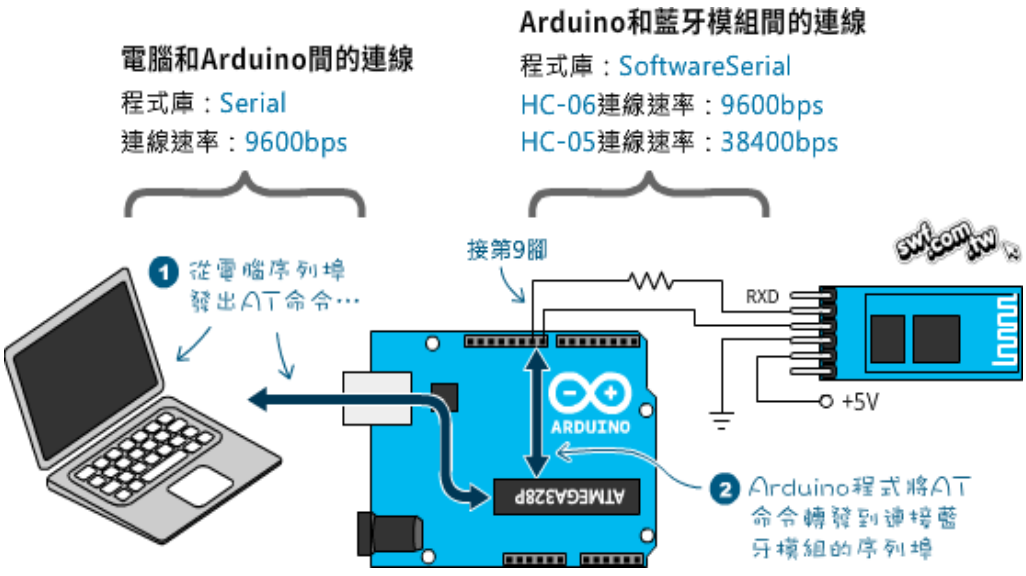

圖 2 專題製作架構圖

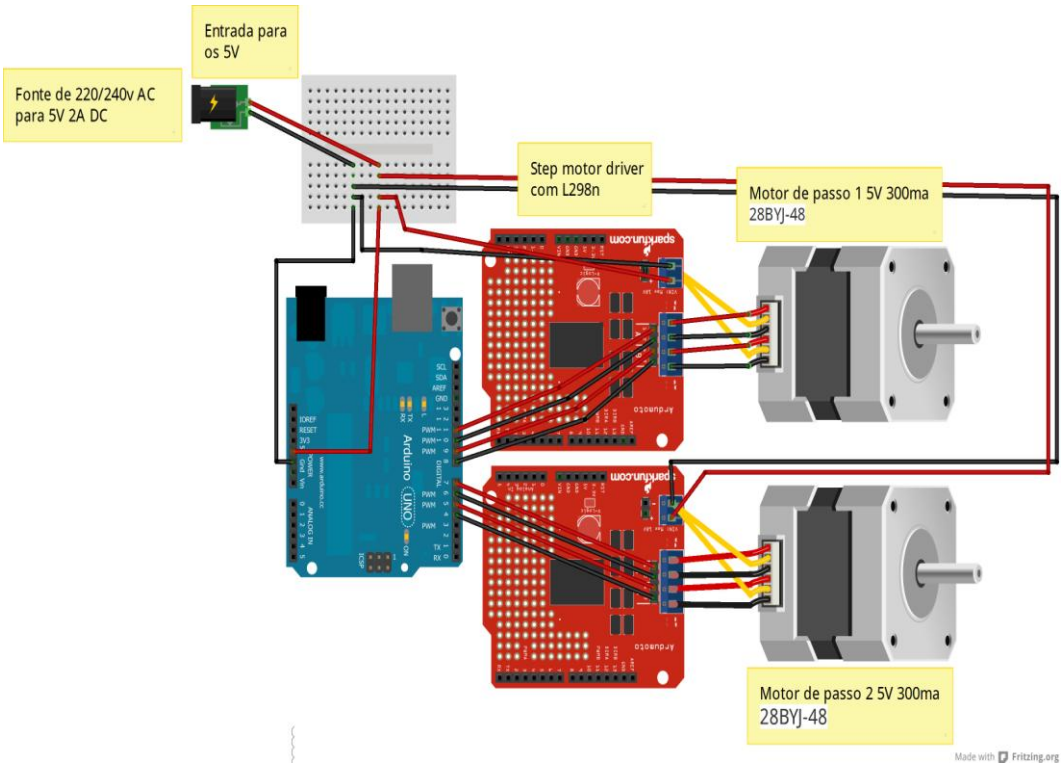

圖 3 馬達驅動架構圖

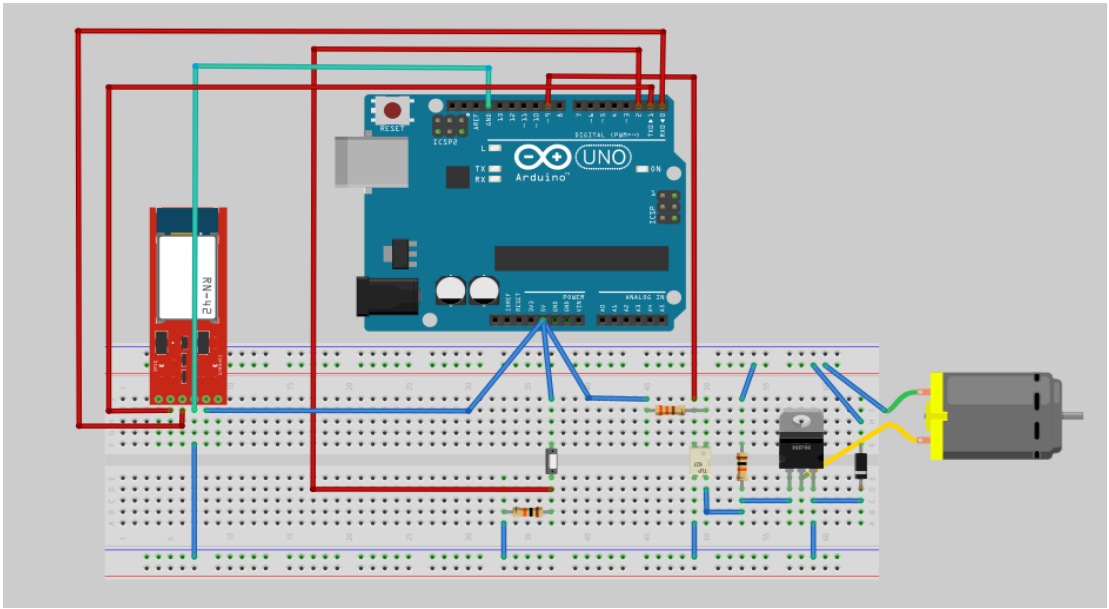

圖 4 Arduino 馬達接線電路

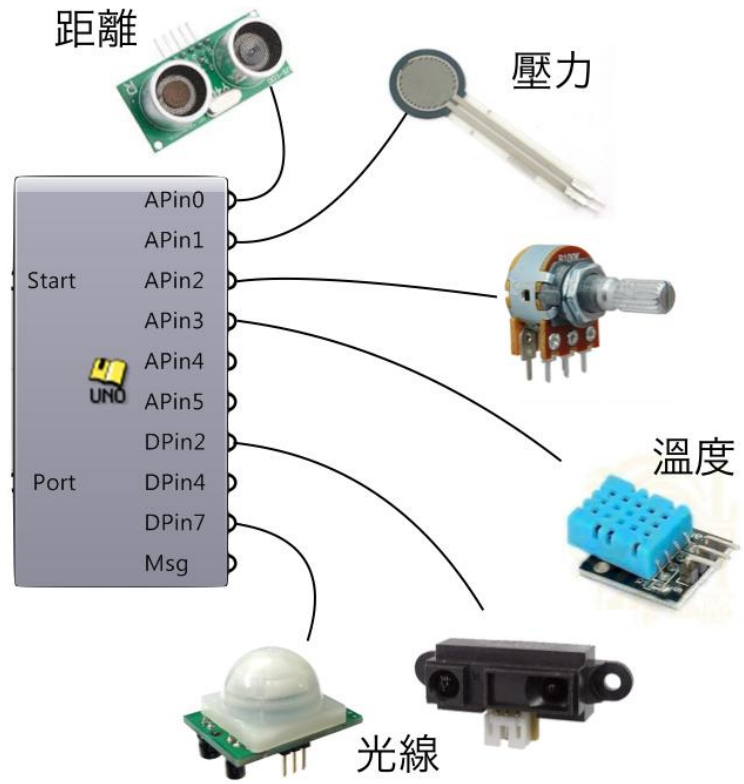

圖 5　Arduino 感測模組接線電路

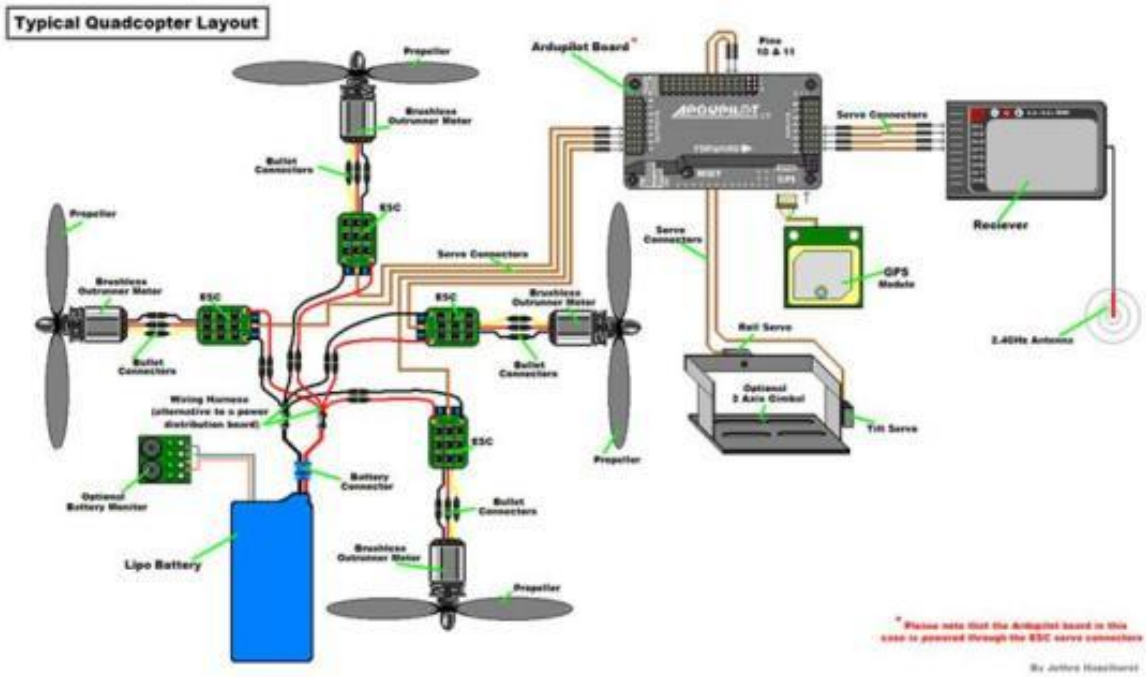

圖 6　Arduino 組件接線電路

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。

(四)、整合感測模組實作避障循線的自走車。

(五)、整合感測模組實作四軸飛行器。

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：

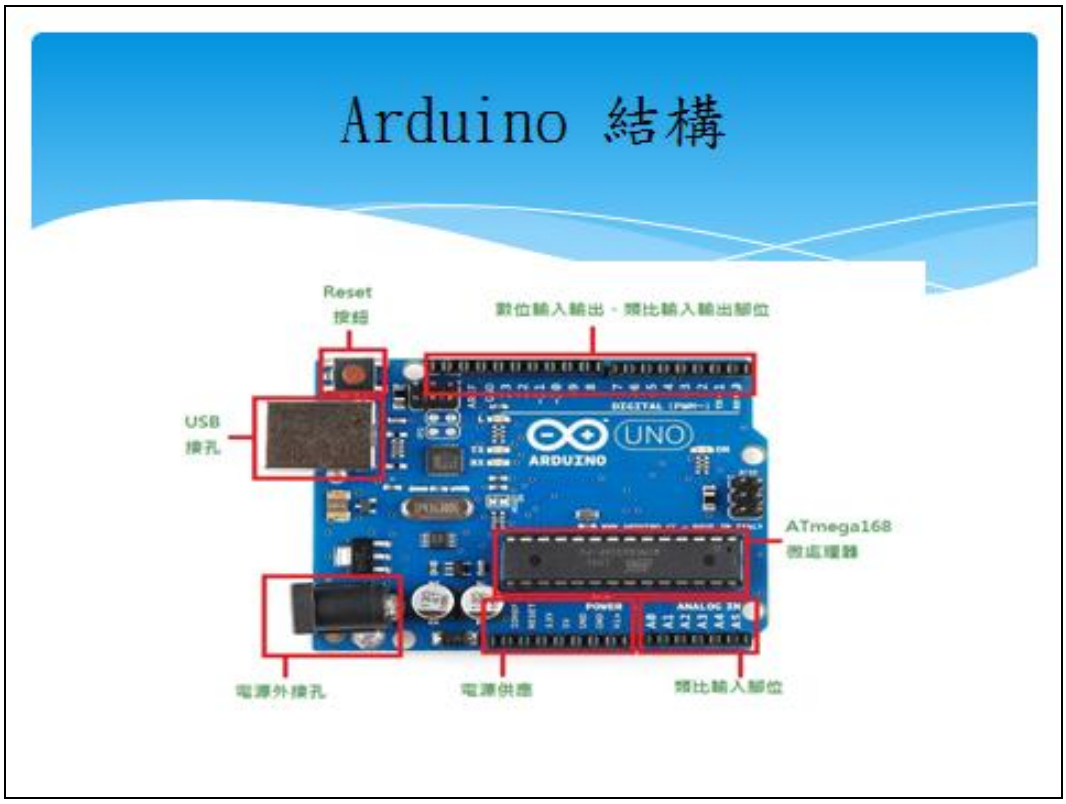

圖 7　Arduino 結構

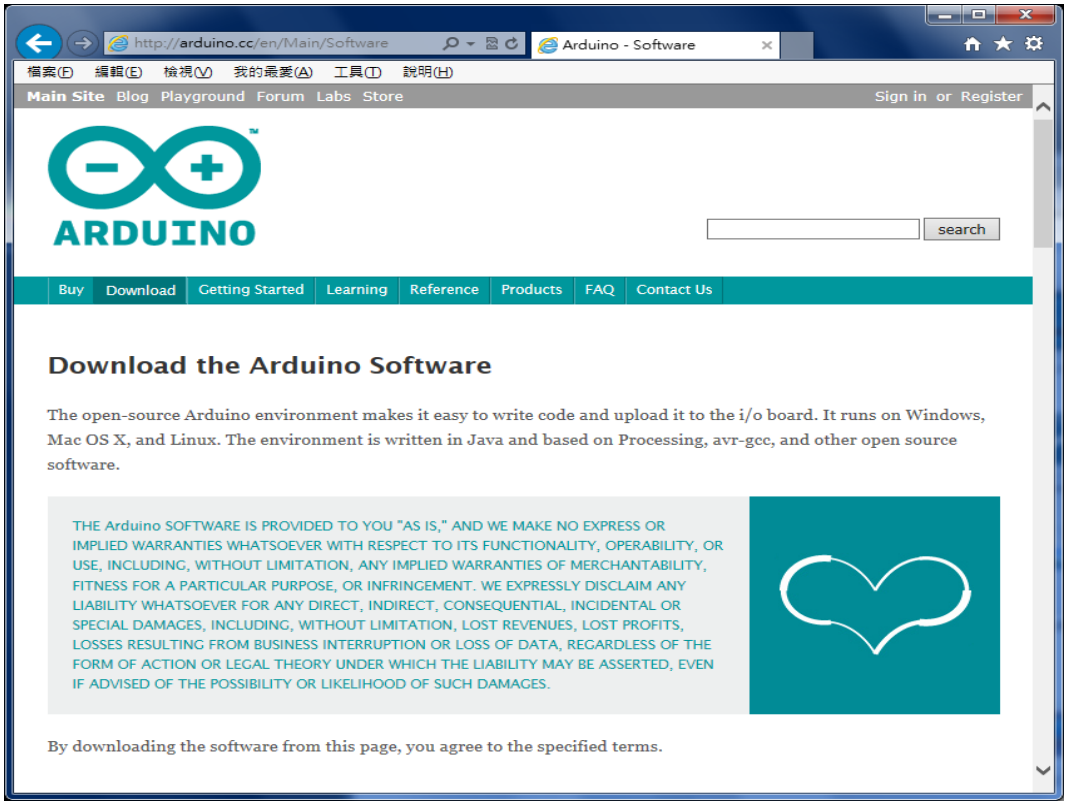

圖 8　Arduino 開發軟體

圖 9　Arduino 連接埠設定

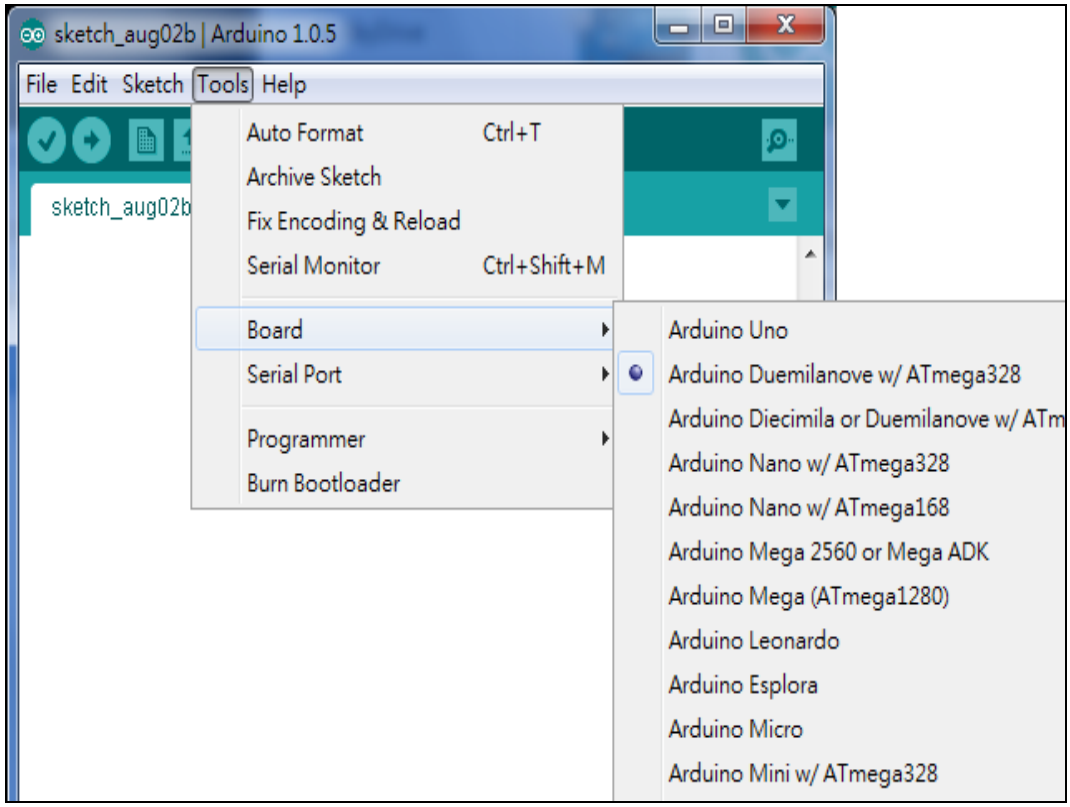

圖 10　Arduino 主機板和微處理器型號設定

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

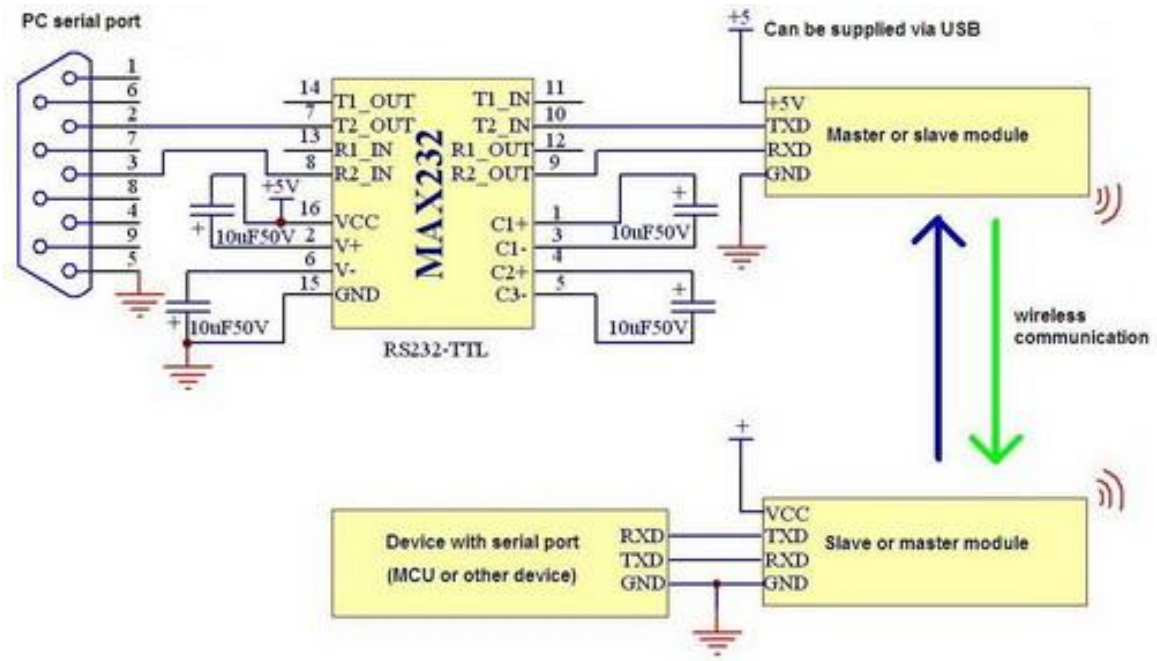

圖 11　藍牙電路

(四)、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。

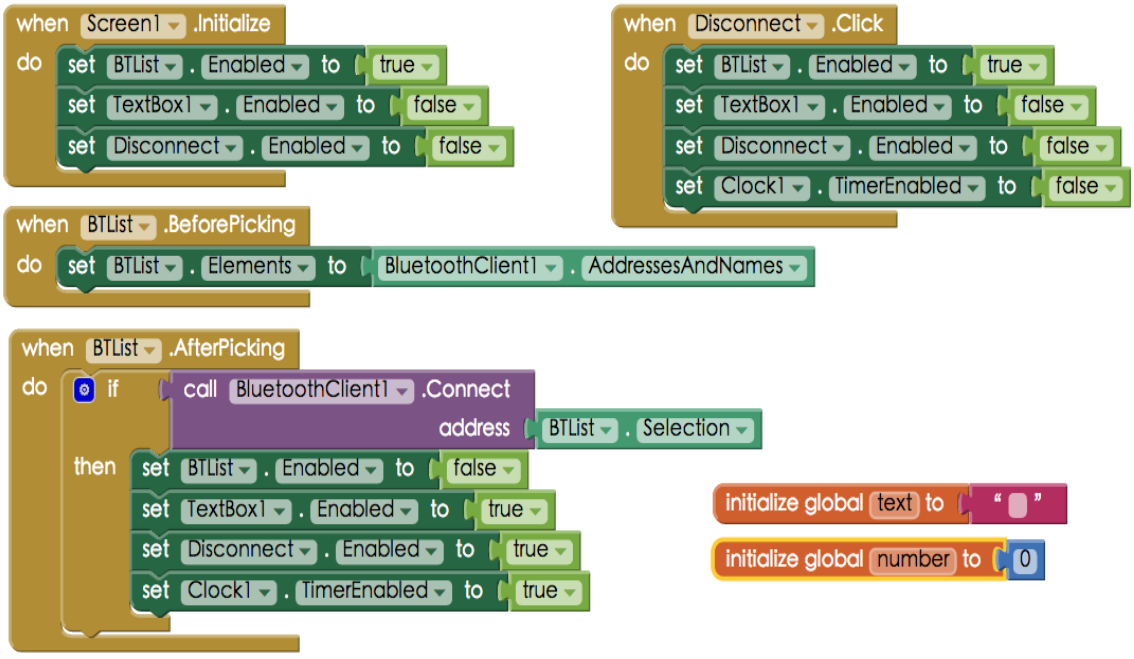

圖 12　Android AI2 藍牙通訊

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	NxtColorSensor NXT顏色感測器
PasswordTextBox 密碼輸入	NxtLightSensor NXT光感測器
Slider 拖動條	NxtSoundSensor NXT聲音感測
Spinner 下拉式選單	

圖 13　Android AI2 PWM 控制

參、專題製作過程或方法

(一)、製作設備及器材

常用感測模組：

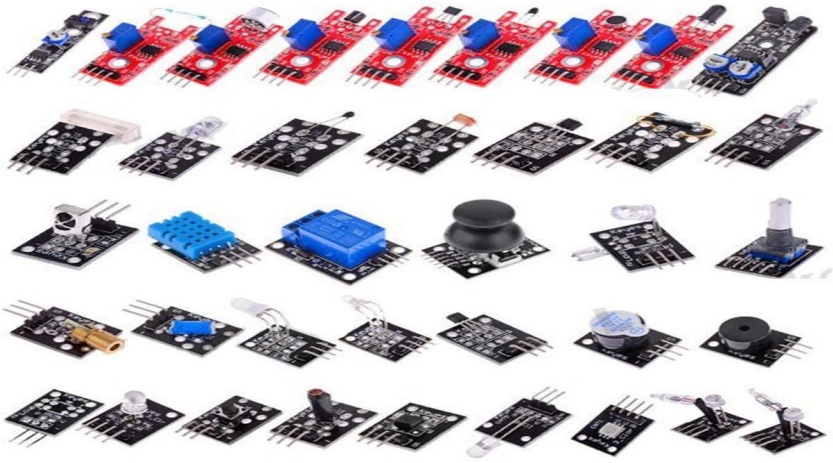

圖 14 常用感測模組

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
12. 常用感測模組	週邊感測電路

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:

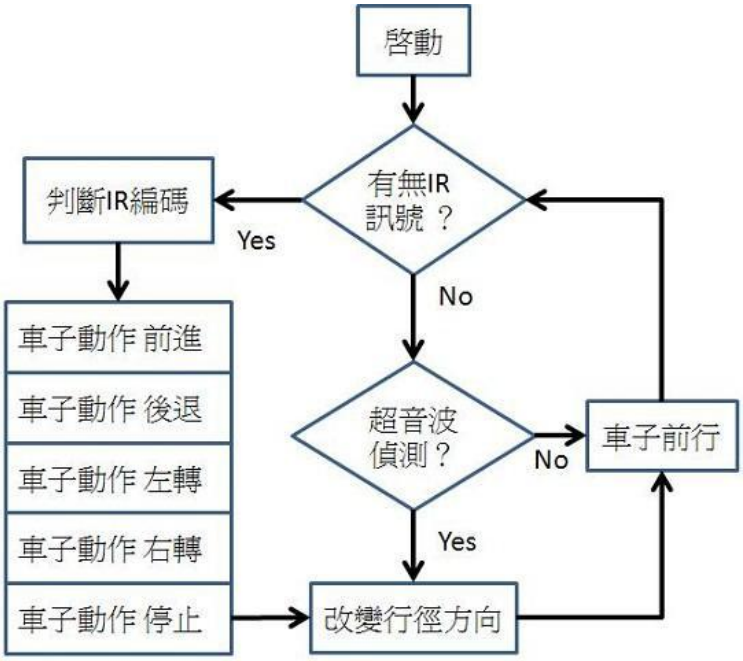

圖 15　外接馬達控制流程

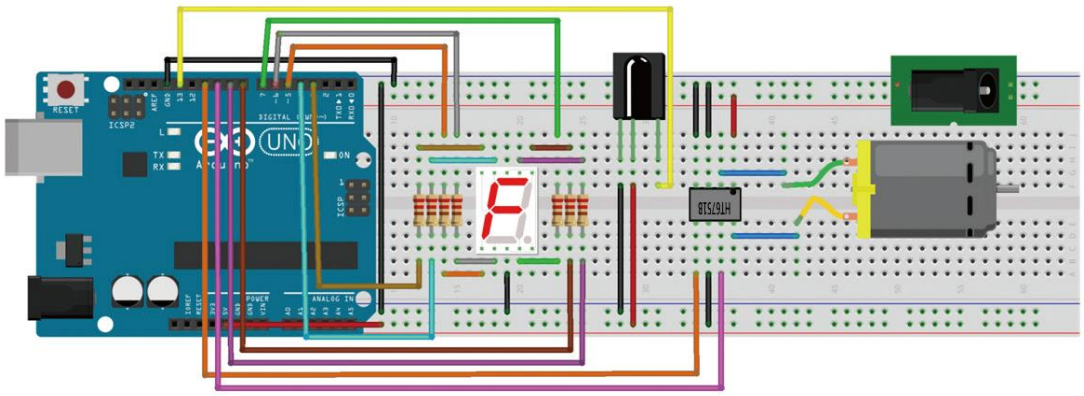

圖 16 直流馬達控制

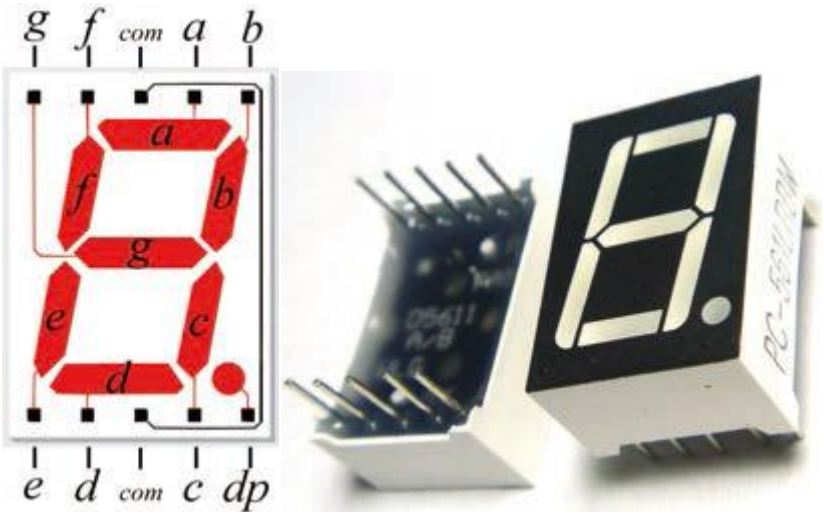

圖 17 七段顯示器各段名稱與腳位

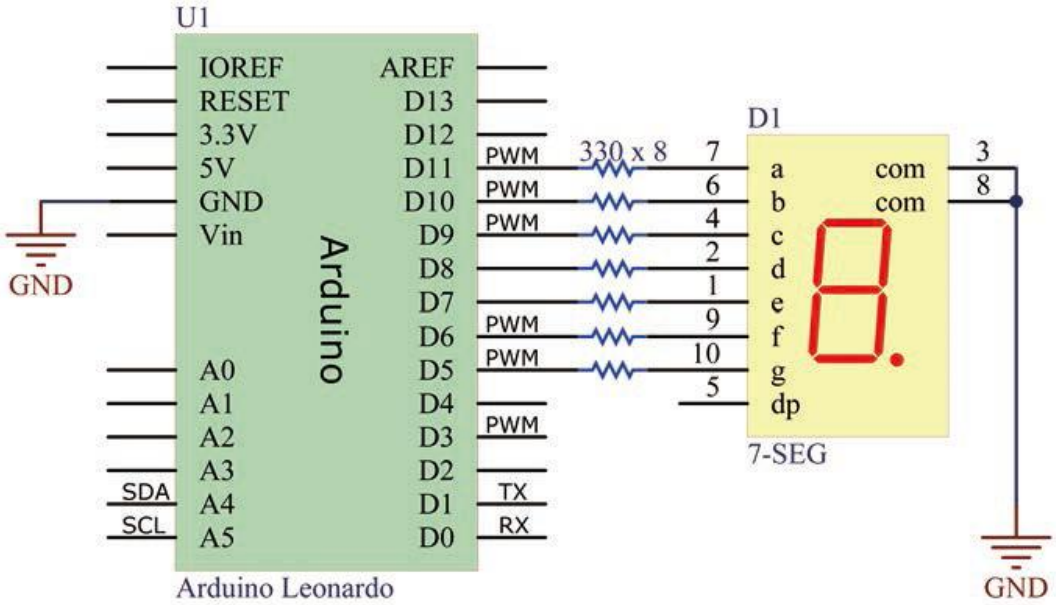

圖 18 七段顯示器硬體電路圖

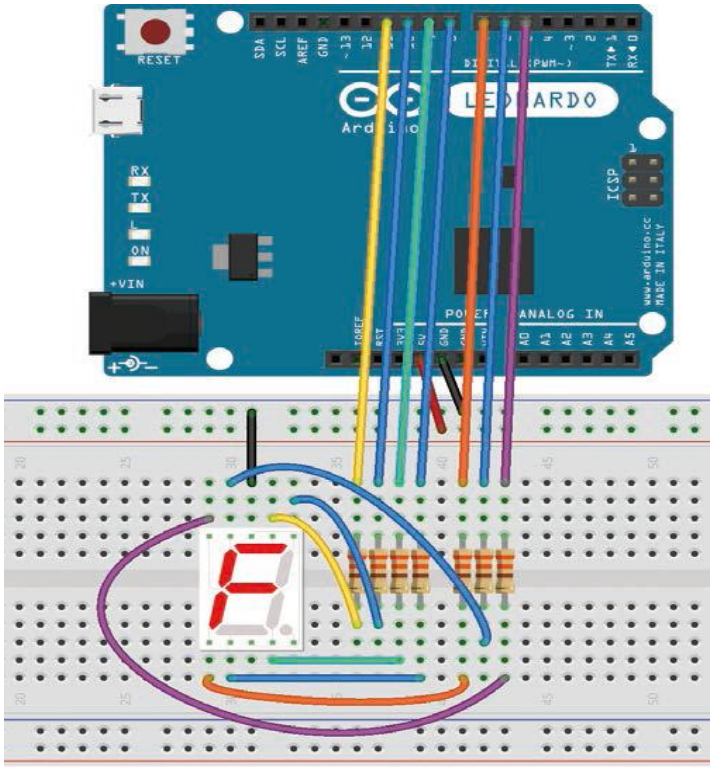

圖 19 麵包板參考接線圖

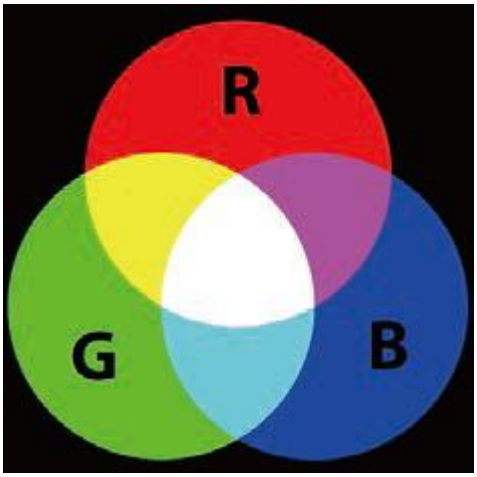

圖 20 三色 LED 光調色盤

R 紅色、G 綠色、B 藍色的RGB LED。RGB LED 有四支腳，其中最長的腳要接地，其他三支腳分別控制R、G、B 色彩。

如果我們將三個LED 的亮度調成：

1. 紅(0)、綠(0)、藍(0)：光呈現出來的顏色是黑色，黑色在LED 光源的真正意義為關閉所有三種顏色。
2. 紅(255)、綠(0)、藍(0)：光呈現出來的顏色是紅色。

3. 紅(0)、綠(255)、藍(0)：光呈現出來的顏色是綠色。

4. 紅(0)、綠(0)、藍(255)：光呈現出來的顏色是藍色。

5. 紅(255)、綠(255)、藍(0)：光呈現出來的顏色是黃色。

6. 紅(0)、綠(255)、藍(255)：光呈現出來的顏色是青色。

7. 紅(255)、綠(0)、藍(255)：光呈現出來的顏色是紫色。

8. 紅(255)、綠(255)、藍(255)：光呈現出來的顏色是白色。

0 表示全暗，255 表示全亮，因此每個顏色的燈亮度可由0~255 來做調整，只要將三個燈混合使用，就可以調出各種不同顏色的燈源，

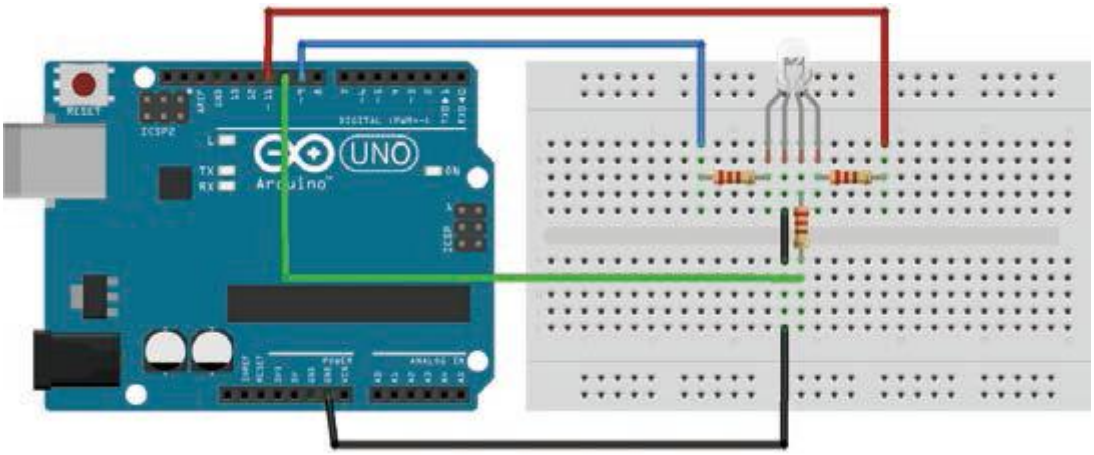

圖 21 三色 LED 接線圖

三色 LED 控制程式碼

```
01 int redPin = 11; // 宣告紅色LED 接腳
02 int greenPin = 10; // 宣告綠色LED 接腳
03 int bluePin = 9; // 宣告藍色LED 接腳
04 void setup()
05 {
06 pinMode(redPin, OUTPUT); // 設定紅色LED 接腳
07 pinMode(greenPin, OUTPUT); // 設定綠色LED 接腳
08 pinMode(bluePin, OUTPUT); // 設定藍色LED 接腳
09 }
10 void loop()
11 {
```

```
12 analogWrite(redPin,0); // 全暗
13 analogWrite(greenPin,0);
14 analogWrite(bluePin,0);
15 delay(1000);
16 analogWrite(redPin,255); // 紅
17 analogWrite(greenPin,0);
18 analogWrite(bluePin,0);
19 delay(1000);
20 analogWrite(redPin,0); // 綠
21 analogWrite(greenPin,255);
22 analogWrite(bluePin,0);
23 delay(1000);
24 analogWrite(redPin,0); // 藍
25 analogWrite(greenPin,0);
26 analogWrite(bluePin,255);
27 delay(1000);
28 analogWrite(redPin,255); // 白
29 analogWrite(greenPin,255);
30 analogWrite(bluePin,255);
31 delay(1000);
32 }
```

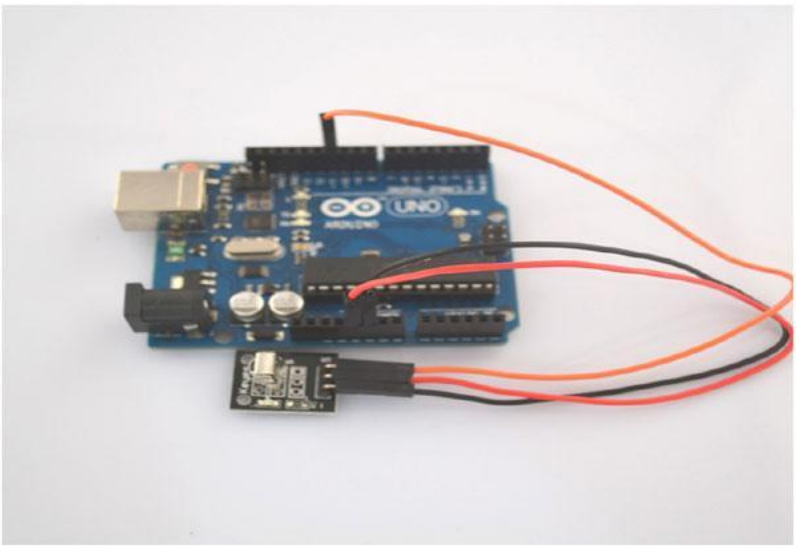

圖 22 循線模組圖

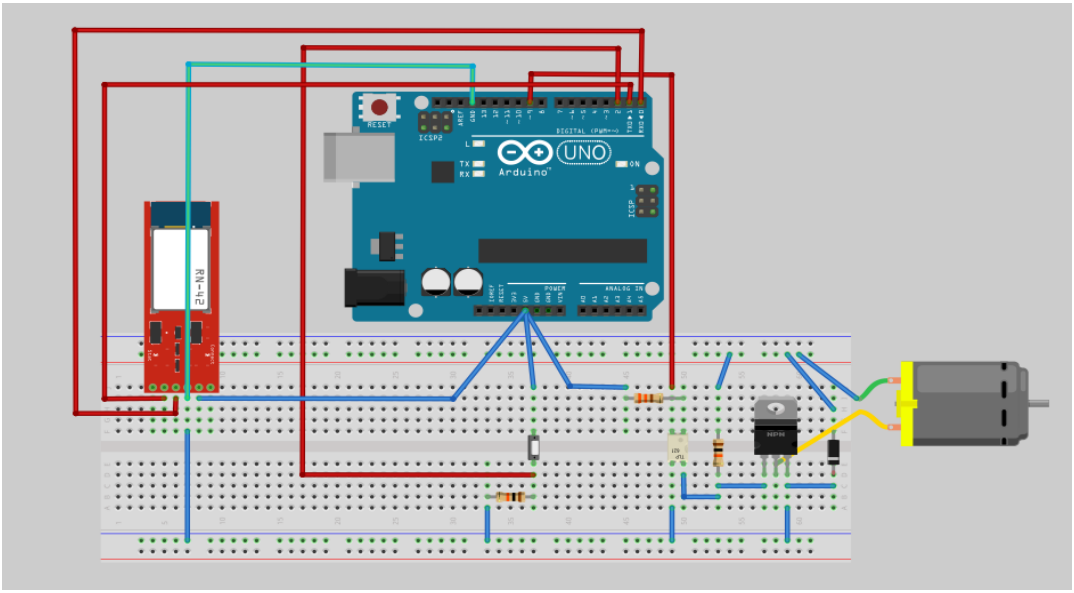

圖 23　馬達控制電路圖

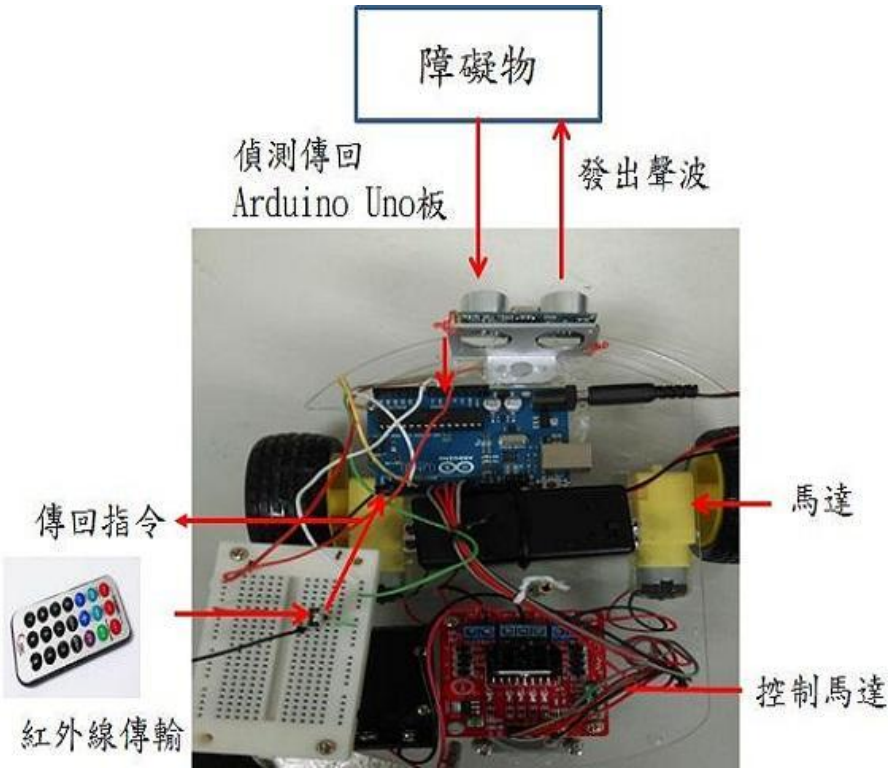

圖 24　自走車感測模組圖

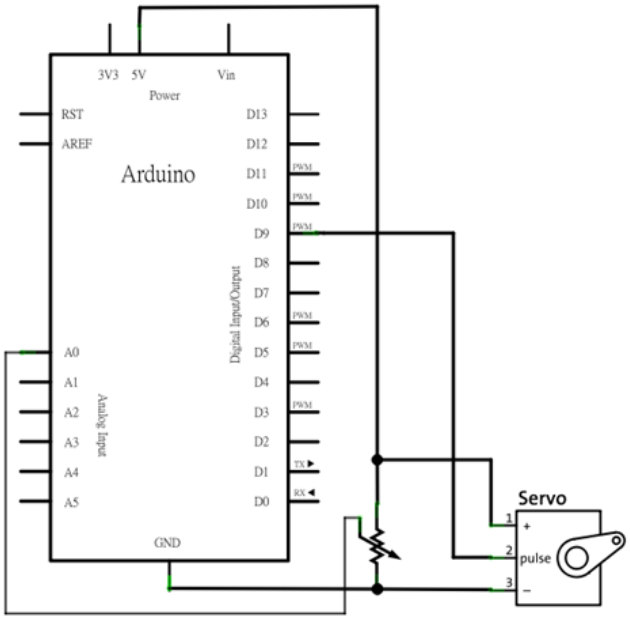

圖 25　伺服馬達接線圖

輸出電壓是由 on 和 off 時間的平均 值

output_voltage =(on_time / off_time) * max_voltage

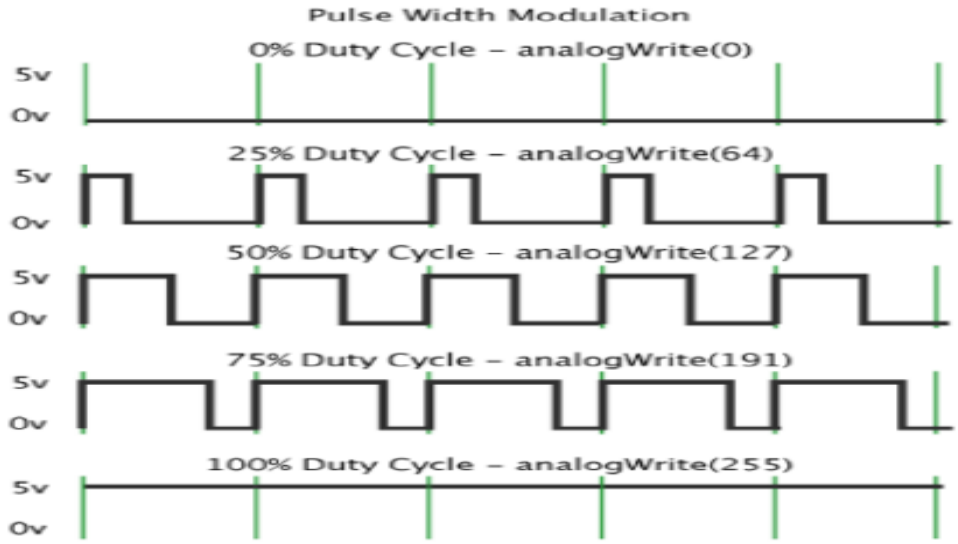

圖 26 PWM 輸出圖

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，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。

微電腦控制所使用的介面一般都是低電壓的直流電，如果遇到要控制的元件或電器設備電壓不同或屬於大電力的，就必須要有一隔離裝置來阻隔兩個不同電力，避免發生互相干擾甚至燒毀的情形發生。這個可以讓小電力控制大電力的介面開關，常用的是繼電器（Relay）和固態電驛（SSR）。

現今電氣設備均標榜微電腦控制或智慧型控制等等，使用的就是如同 Arduino 這樣的微控制器，也就是說，當我們要製作一個微電腦控制的電器設備時，就必須要有微控制器，而因微控制器一般都是使用低電壓的直流電源，所以如果要控制大電力的馬達（motors）、電暖爐（heaters）、燈泡（bulbs）等設備的開和關時，就必須使用繼電器或固態電驛來做介面了。

- 紅外線遙控原理

紅外線是目前最常見的一種無線通訊，普遍使用在家電以及玩具產品，如電視、音響、錄放影機、冷氣機、DVD、MP3 Player、遙控車等。紅外線遙控之所以被大量採使用，主要是因為紅外線裝置體積小、成本低、耗電少及硬體設計容易。下圖是紅外線發射器（Transmitter 或稱 IR LED）和接收器（Receiver）常見外觀，一般來說，紅外線遙控系統由發射器和接收器這兩部份組成。

紅外線是不可見光，其實生活中充滿了紅外線，只是我們看不到。紅外線主要來自太陽，不過很多物體也會發射紅外線，例如燈泡、蠟燭、中央空調等，甚至人體也會散發紅外線。人體所發出的紅外線的量是可以偵測的，耳溫槍就是利用這個道理測量人的體溫。有這麼多紅外線光源，當然會對遙控造成干擾，所以得做一些預防措施確保通訊正確。

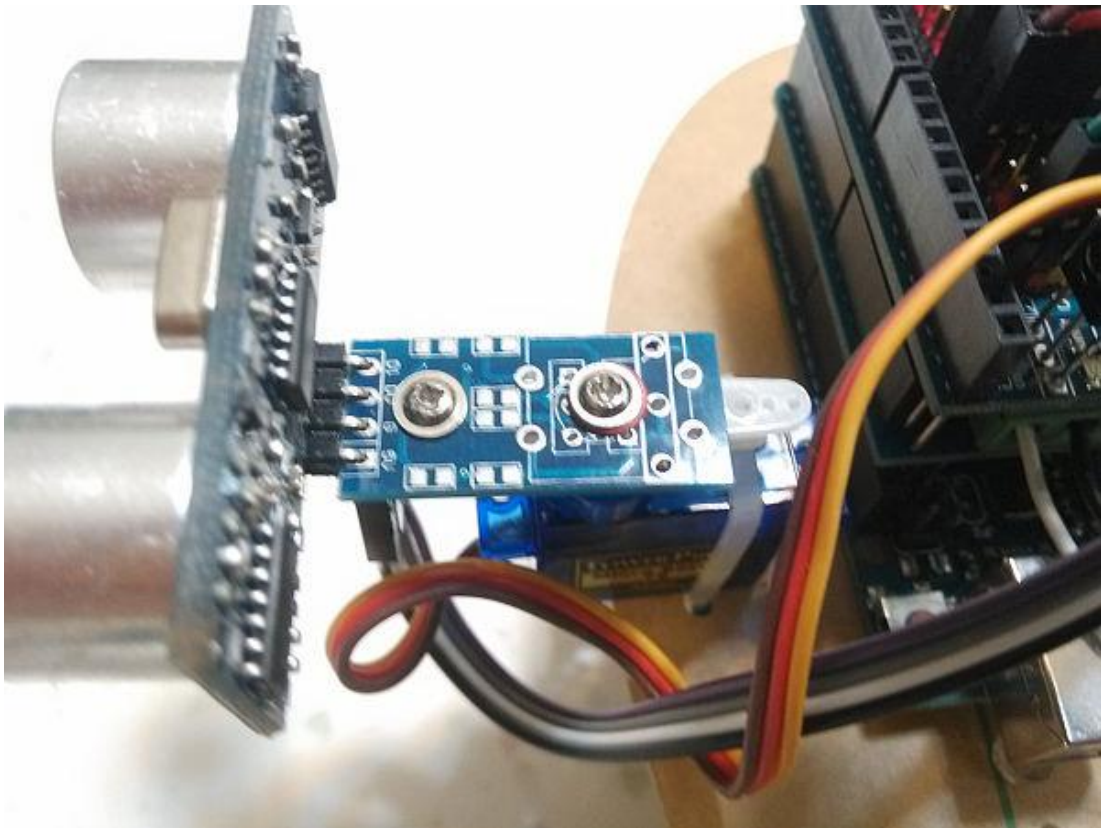

圖 27 超音波模組接線圖

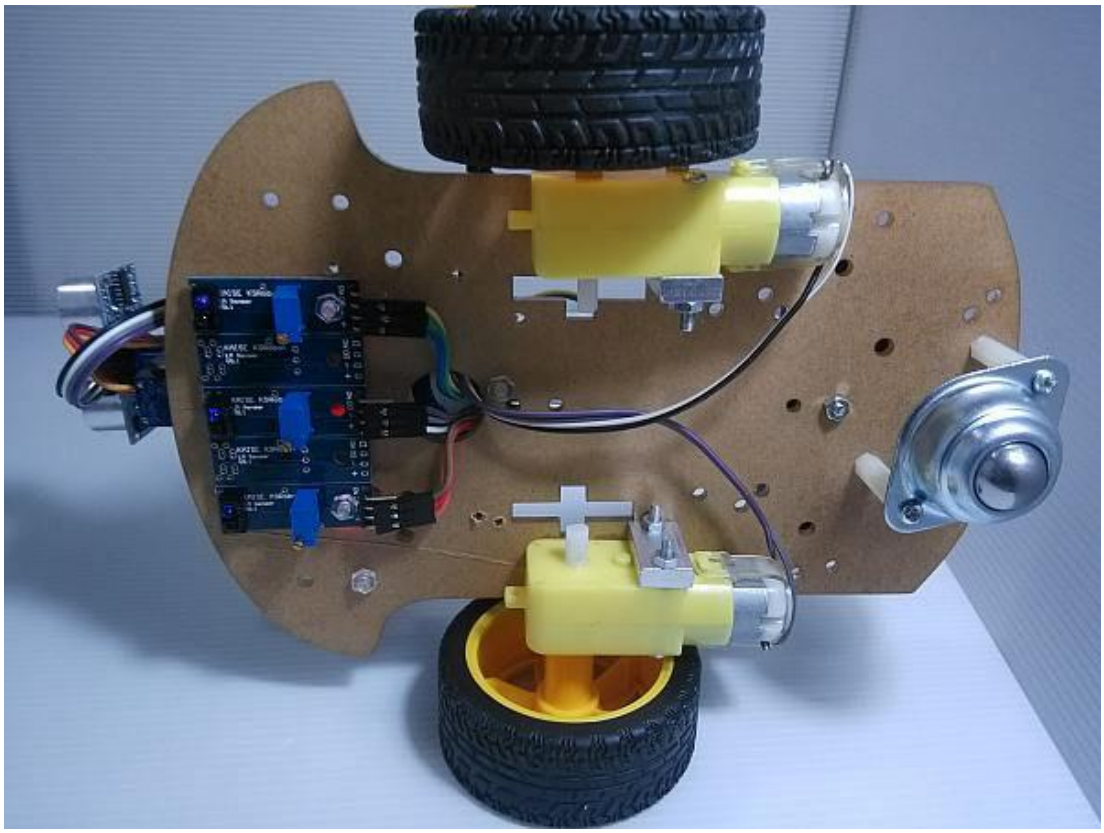

圖 28 紅外線模組接線圖

（二）超音波感測器

超音波感測器是由超音波發射器、接收器和控制電路所組成。當它被觸發的時候，會發射一連串 40 kHz 的聲波並且從離它最近的物體接收回音。

超音波測量距離的方法，是測量聲音在感測器與物體之間往返經過的時間。由於超音波從發射到返迴是兩段距離，因此在計算時必須將結果除以 2 才是正確的物體距離。

超音波感測器主要應用在機器人或自走車避障、物體測距等。

超音波模組HC-SR04有四支腳:電源及接地外，一支腳是Trig，功能為啟 動發射超音波，頻率約為40KHz，另一支腳為Echo回音，為接收到回 返的招音波指示。

測距離時，給予HC-SR04的Trig一個,至少10μS寬度的脈波，啟動HCSR04發出超音波，發射後，Echo會由低電位變為高電位，直到接收到 反射回來的超音波，Echo由高電位回復為低電位，

（三）控制流程：

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%。以 Arduino 控制板搭配藍牙無線通訊模組，來做為手機遠端遙控用；當使用者按下手機 APP 程式上的 4 個控制鍵後，該鍵碼立即由手機上的藍牙傳至 Arduino 上，再藉由 Arduino 上的程式，解析出該鍵碼，以便驅動直流馬達進控制遙控車動作。

- Android 智慧型手機：負責傳遞遙控之控制訊號給車體。
- Arduino 控制板：結合 Arduino 和 L298N 馬達驅動晶片的一塊整合板，可以驅動兩顆直流馬達(電流最大到 1.2A) 並利用 PWM 訊號來控制馬達轉速。
- 藍牙模組：透過藍牙通訊協定，連接 Android 手機及 Arduino 控制的介面。
- 遙控車主體：主要透過控制板訊號的 HIGH、LOW 輸出和變化，控制馬達正反轉，遙控車透過控制板加上藍牙無線 通訊模組的配合，以藍牙模組收到手機訊號，透過控制板上的 L298N 放大電路來擴大電流，使驅動直流馬達運轉，來控制動作。
- 避障裝置：透過在遙控車前方裝設的超音波感測器，距離的感測，可以在偵測到前方障礙物時，不需要人為的操作就可以自行轉彎，達到避障的效果，也可以避免在人為操作不當時，撞到障礙物，而使載體受到損壞的機制。

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

}
```

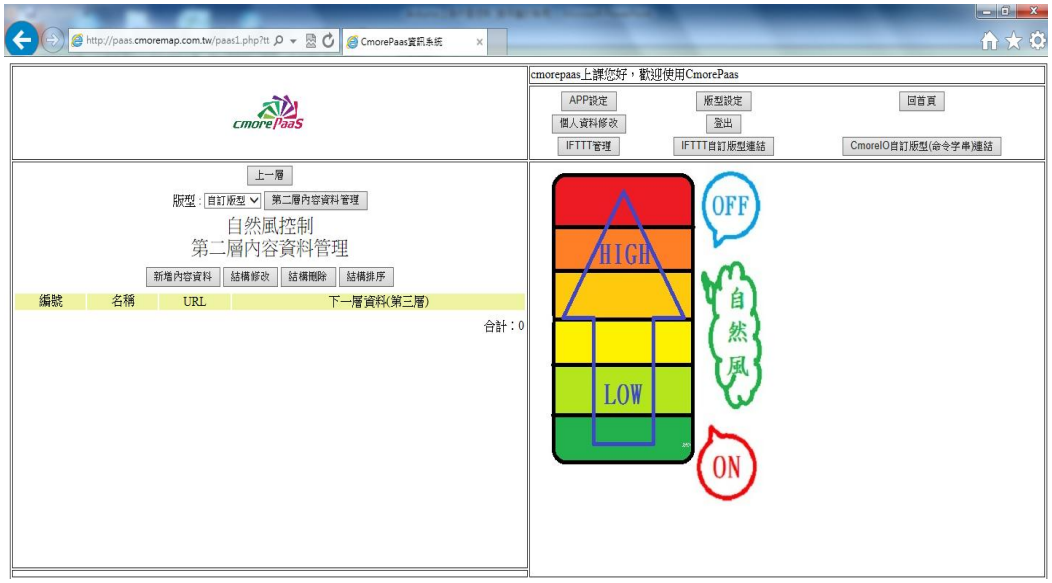

圖 29 手機功率操作模式

RC 伺服馬達(Radio Controlled Servo Motor) 大部份是透過 PWM (Pulse Width Modulation, 脈波寬度調變)來控制，Arduino 裏內建了 Servo Library 讓事情變得很簡單，就算你對 PWM 不熟，也可以很輕鬆地控制伺服馬達。底下的程式碼示範 Servo Library 的使用方法(Servo.pde):

```
01 // 引用 Servo Library
02 #include <Servo.h>
03
04 // 建立一個 Servo 物件
05 Servo myservo;
06 // 旋轉角度
07 int value = 0;
08
09 void setup()
10 {
11   myservo.attach(9);  // Servo 接在 pin 9
12 }
13
14 void loop()
15 {
16   if (value == 0)
17     value = 180;
18   else
19     value = 0;
20
21   // 叫 Servo 旋轉角度:
22   //    myservo.write(0) 是叫 Servo 旋轉到 0 度的位置
```

```
23  //    myservo.write(180) 是叫 Servo 旋轉到 180 度的位置
    myservo.write(value);
24
```

圖 30　紅外線控制器

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。

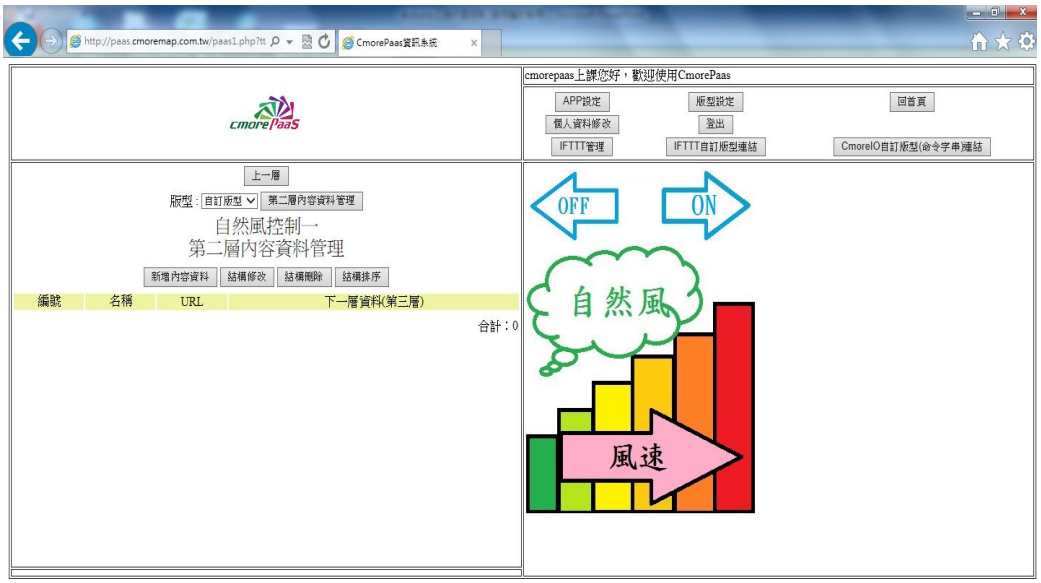

圖 31 手機設計版面

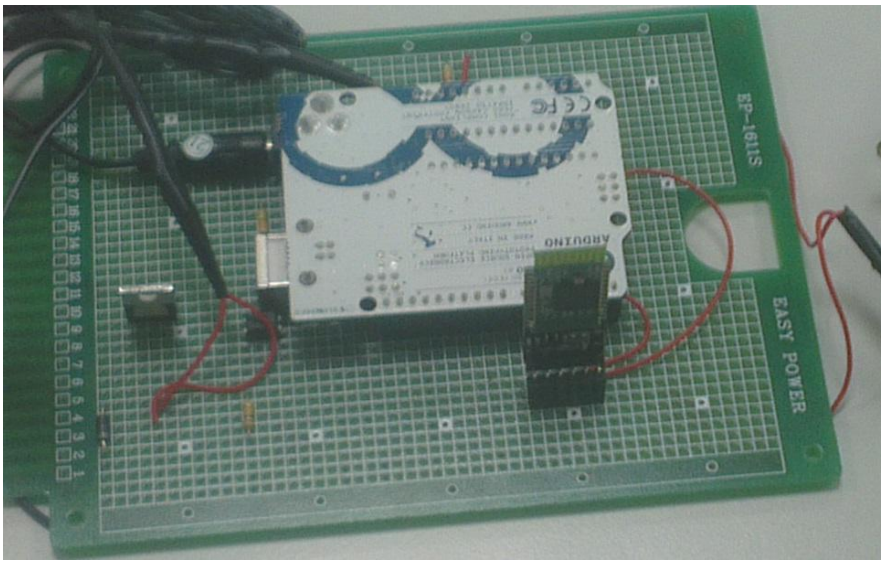

圖 32 實體電路

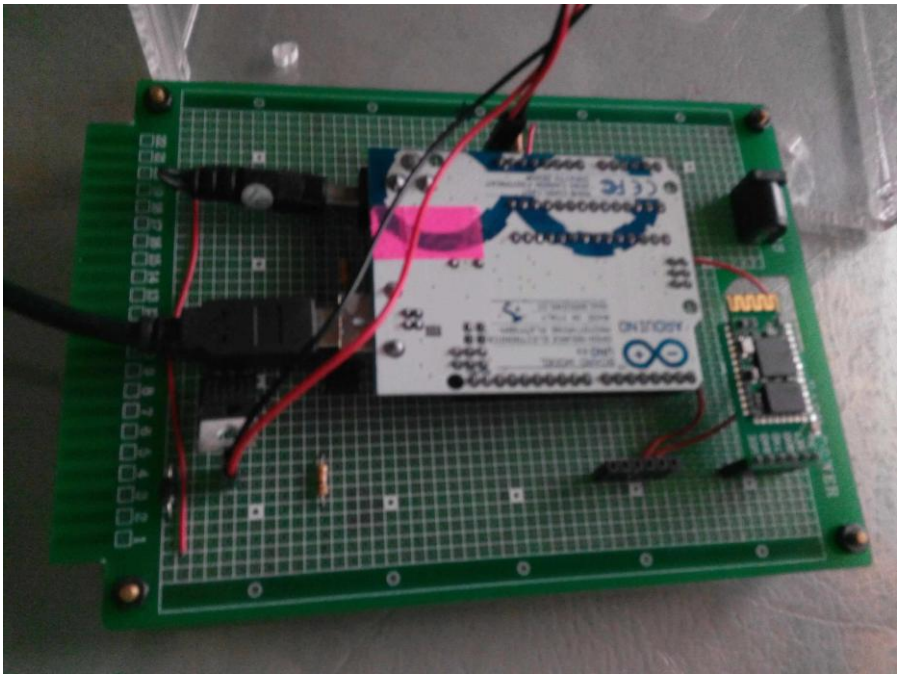

圖 33　Arduino 之成果

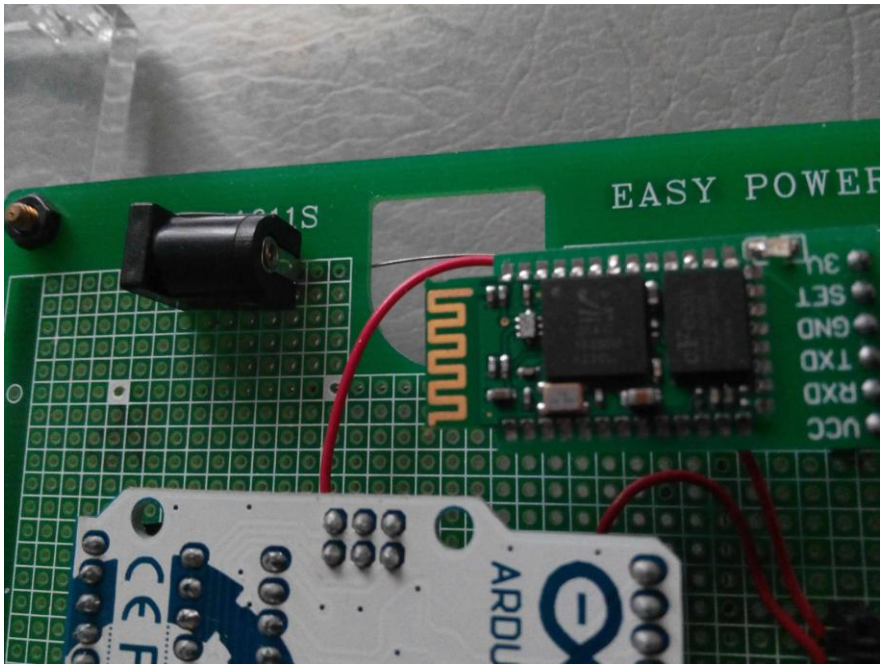

圖 34 **藍牙傳輸**

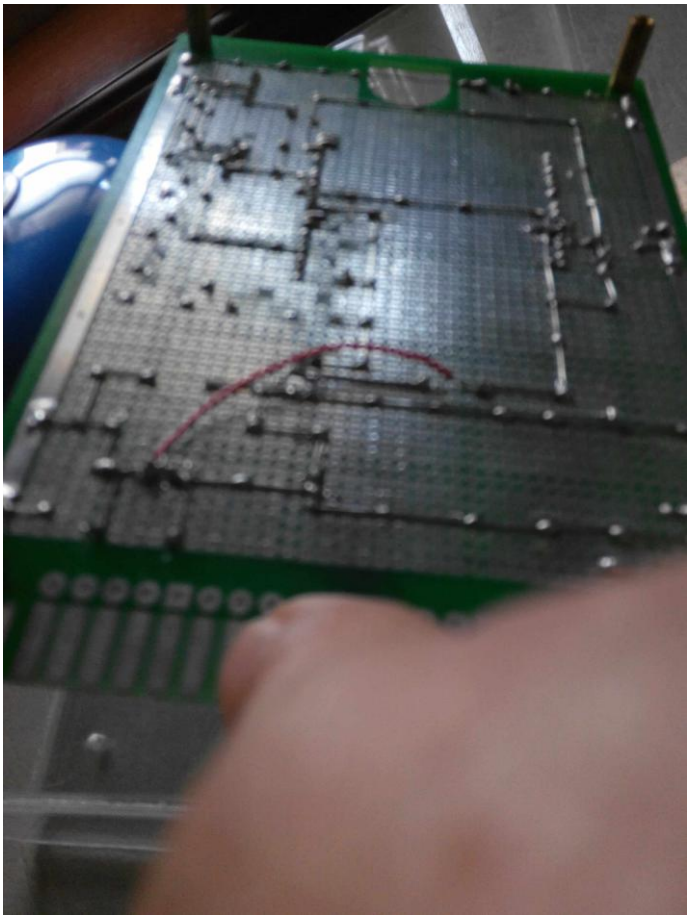

圖 35　**電路焊接**

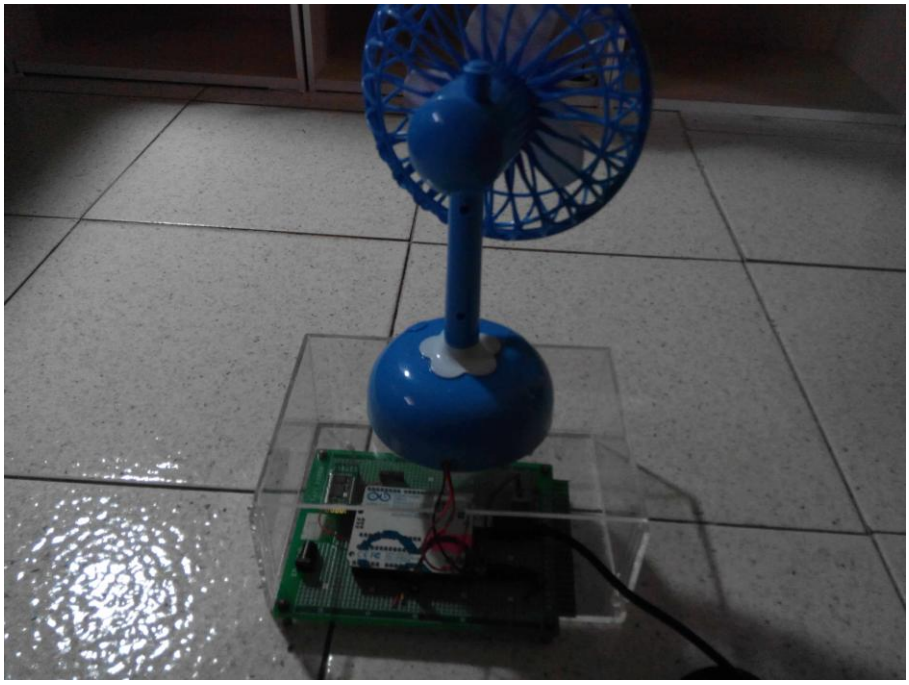

圖 36 **負載功率輸出成品**

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 微控板去控制四軸馬達的電子器件轉速。

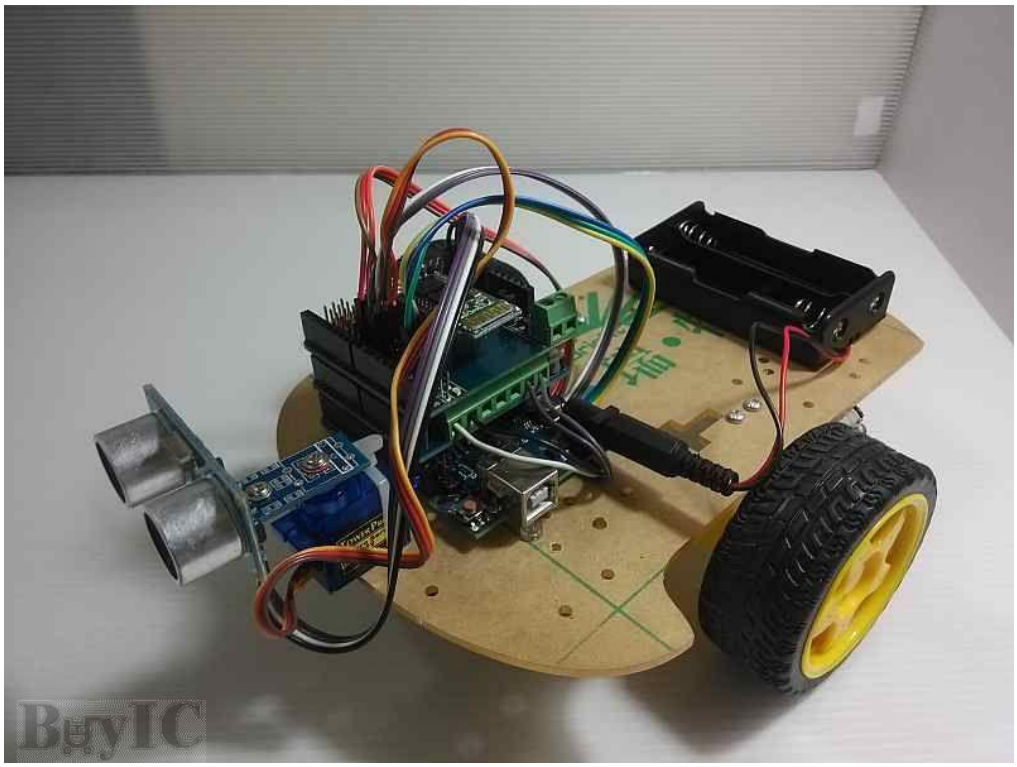

圖 37 二軸馬達控制

圖 38 四軸馬達控制

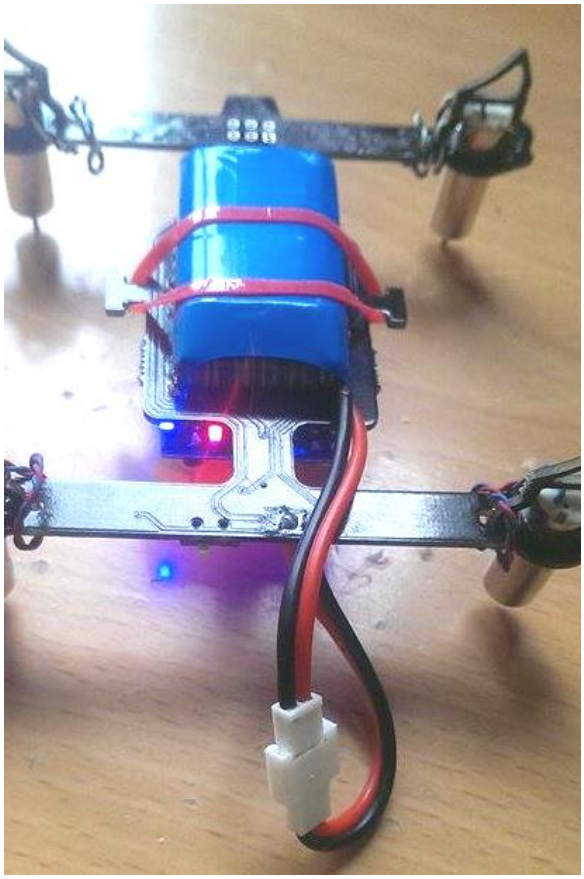

圖 39 四軸馬達藍牙無線控制

圖 40 APP 藍牙無線控制畫面

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。

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_resource/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
6. 網址 **appinventor**.mit.edu/
7. Arduino http://www.arduino.cc/
8. Arduino.TW 台灣使用者社 http://arduino.tw/index.php
9. 維基百科 Arduino http://zh.wikipedia.org/wiki/Arduino
10. Cooper Maa 的 Arduino 筆記
11. http://coopermaa2nd.blogspot.tw/search/label/Arduino