黎明學報　第二十九卷　第二期　中華民國一〇七年七月三十一日
JOURNAL OF LEE-MING INSTITUTE OF TECHNOLOGY
VOL. 29, NO.2, JUL. 2018

機電整合微電腦Arduino自動化控制技術

楊吉仕
黎明技術學院創意產品設計系

yjs@mail.lit.edu.tw

摘要

本研究係結合「物聯網IoT」與「創客Maker」之精神，來運用「機電整合mechatronic integration」、「微電腦控制microcomputer controlling」、「程式設計programming designing」等技術，來達成「數位控制 digital controlling」、「藍牙通訊Bluetooth communication」、「馬達驅動motor driving」以及「感測器感知sensor perception」…等功能開發。

本機電整合微電腦Arduino自動化技術，列舉四項專題研究成品，包括：(1)手機藍牙遙控智慧家居開關盒、(2)黎明校歌蜂鳴器演奏、(3)超音波測距汽車電動尾門、(4)陀螺儀自動控制車燈投射角度裝置…等，上述四項範例基本上係用創意設計之思維，來進行Arduino微電腦單晶片之自動化機電整合，以及控制技術之規劃、設計、製作，學習所需之專業知識與技能，以達成培育自動化產業專業技術人才之目標。

*關鍵字：*Arduino微電腦單晶片、物聯網、創客、機電整合、微電腦控制、程式設計、數位控制、藍牙通訊、馬達驅動、感測器感知

1. 前言

本研究將針對四項創意專題範例進行相關關鍵技術探討，所含括技術類別分別述說如下：

(1)手機藍牙遙控智慧家居開關盒：包括藍牙通訊、手機積木方塊程式設計、繼電器模組控制…等技術。

(2)黎明校歌蜂鳴器演奏：包括蜂鳴器震盪發聲、音調與節拍演奏控制…等技術。

(3)超音波測距汽車電動尾門：包括超音波測距、線性電動缸正反轉驅動、近遠端極限開關保護…等技術。

(4)陀螺儀自動控制車燈投射角度裝置：包括陀螺儀角度感測、步進馬達控制…等技術。

2. 手機藍牙遙控之智慧家居開關盒

該專題乃利用手機方塊程式，利用HC-06藍牙通訊模組進行遠端繼電器模組之開關控制[1]，以達成交流110V電壓之家電啟閉[2]。

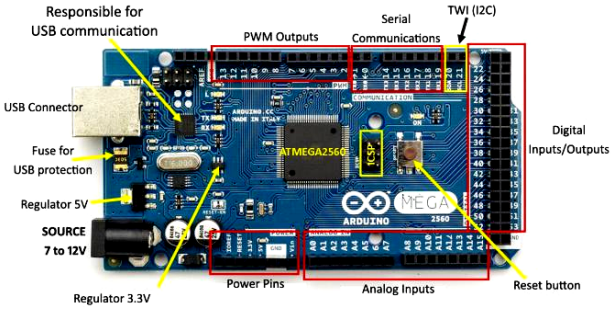
其目的在於製作物聯網遠端居家控制系統，利用App Inventor撰寫手機積木方塊程式，以連接藍芽通訊模組，發出自訂之控制指令，以設定Arduino微電腦板之定時時間；當定時時間到達後，則讓Arduino控制繼電器模組之啟閉，以達控制家電效果。此外，同時也利用Arduino內部時鐘程式讓LCD 1602液晶顯示模組呈現目前時間，也可利用盒上4顆按鈕來設定目前時間或家電啟閉時間；本專題成品能利用遠端手機或現場按鈕方式來設定家電自動啟閉，以提升住家環境控制自動化品質，並可享受物聯網的便利性。

2.1 智慧家居開關盒零組件介紹

手機藍牙遙控智慧家居開關盒之關鍵零組件包括Arduino Mega 2560微電腦板、藍牙通訊介面模組、文數字型液晶顯示模組、繼電器模組…等，零組件細節分述如後。

2.1.1 Arduino Mega 2560微電腦板

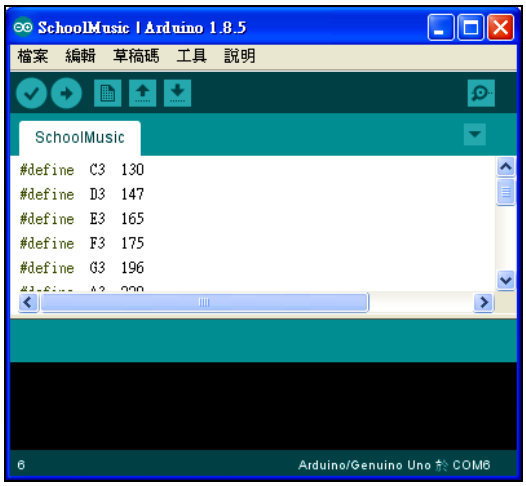
Arduino Mega 2560微電腦板(如圖一，購買價格約NT$280)其官方建議輸入直流電源之电壓範圍為7~12V，以16MHz時脈運作。該板有54根數位I/O接腳，通常標記0~53，可當作數位輸出或輸入腳來使用，其中又有15根腳可做PWM類比輸出；每根數位腳之最大輸出電流為40mA，均內建有20~50k歐姆提升電阻，一般出廠預設狀態為關閉。另外還有16根類比輸入腳，通常標記A0~A15，每根類比腳均可提供10位元或1024之數值解析感測能力。



圖一：Arduino Mega 2560微電腦板外觀圖。

不管使用何種型號款式之Arduino微電腦板，均得先到官網(https://www.arduino.cc/)下載Arduino IDE整合發展環境 (如圖二)，軟體安裝完成後尚有二項重要設定待完成，一是選擇Arduino微電腦板之型號款式，須至主功能列之「工具」⇨「開發版」⇨選擇Arduino之型號款式；另一設定則是選擇微電腦板

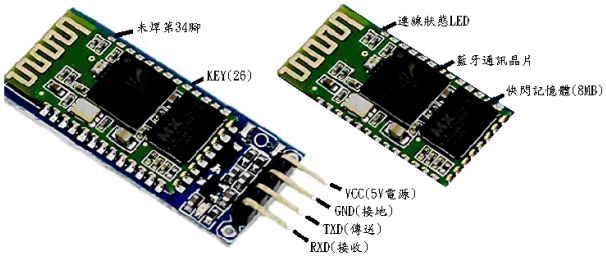
與電腦之間連線的通訊管道，須至「工具」⇨「序列埠」⇨選擇Arduino利用USB線來連接電腦之COM序列埠編號。當上述設定完成後，便可於IDE開發環境撰寫C語言程式碼，於撰寫完成後，按下「上傳」之向右箭頭圖案的圓形按鈕，便可於編譯通過之後，自動將程式上傳至Arduino微電腦板。



圖二：Arduino IDE整合發展環境。

2.1.2 HC-06藍牙通訊介面模組

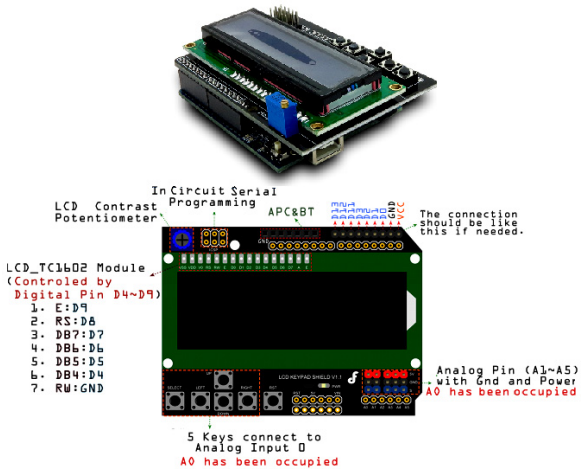
HC-06藍牙通訊介面模組(如圖三，購買價格約NT$135)，採用英國劍橋Cambridge Silicon Radio公司(CSR)的BC417143晶片，支援藍牙2.1+EDR規範，通訊距離10公尺。通常基本藍牙通訊晶片沒有引出接腳，而是在印刷電路板四周留下齒孔，廠商為方便連接上Arduino，模組會增加一塊底板，並於其上附帶直流電壓轉換IC，以利能夠使用Arduino之5V電源。



圖三：HC-06藍牙通訊介面模組外觀圖。

2.1.3 LCD 1602文數字型液晶顯示模組

LCD 1602僅是一塊液晶顯示幕，其與Arduino之間的傳統接線方式頗為複雜，因此通常會加購按鍵底板(keypad shield，如圖四，完整模組之購買價格約NT$200)來簡化接線(如表一之LCD 1602與Arduino接腳對照表)。LCD 1602可顯示2行文字，每行16個字，每個字可為英文字母、希臘字母、標點符號或數學符號，除了顯示資訊外，其它還包括資訊往左或往右捲動、顯示游標和LED背光調整…等功能。

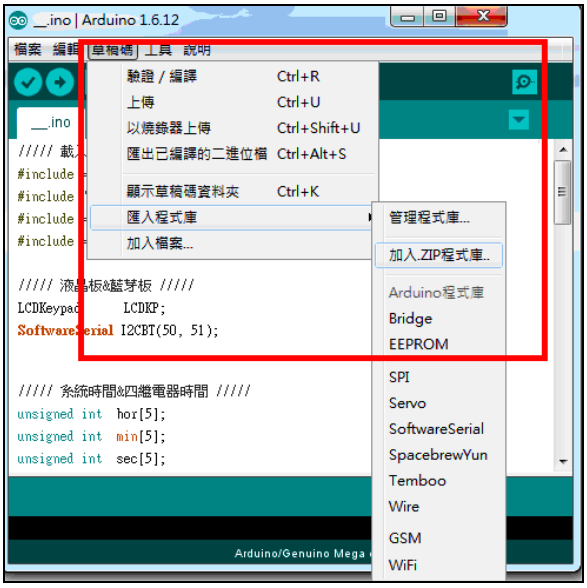


圖四：LCD 1602文數字型液晶顯示模組外觀圖。

表一：LCD 1602與Arduino接腳對照表。

Keypad 接腳	Arduino 接腳	說明
D4	數位輸出腳 4	LCD 資料接腳
D5	數位輸出腳 5	
D6	數位輸出腳 6	
D7	數位輸出腳 7	
RS	數位輸出腳 8	Register Selections
Enable	數位輸出腳 9	LCD 致能
A0	類比輸入腳 0	Key buttons 輸入(類比)
5V	5V 電源腳	
GND	GND 接地腳	

由於Arduino IDE整合發展環境本身沒有LCD 1602 Keypad Shield之I2C程式庫，亦即專屬之腳位定義程式庫，所以必須在網路上尋找名為LiquidCrystal_I2C之函式庫的ZIP壓縮檔，再於Arduino IDE整合環境下點選「草稿碼」⇨「匯入程式庫」⇨「加入.ZIP程式庫…」⇨選取該壓縮檔，即可在整合環境中擁有呼叫該函式庫之能力(如圖五)。



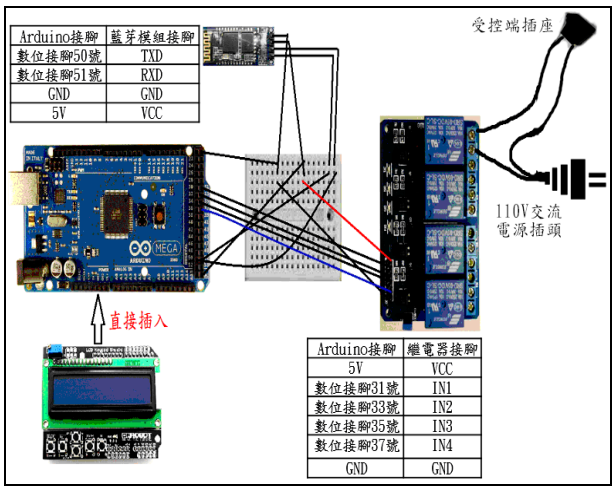
圖五：LCD 1602液晶顯示模組函式庫安裝畫面。

2.1.4 繼電器模組

繼電器模組(relay module，如圖六，購買價格約NT$80)主要功能係將「弱電」與「重電」隔離開來，亦即將「控制端」Arduino之直流5V控制訊號電壓與「受控端」家電之110V交流電壓隔離開來；「控制端」三個接點分別為5V電源腳Vcc、接地腳GND與訊號腳In；而「受控端」之單刀雙投開關有三個接點，分別為常閉接點NC、共地端COM及常開接點NO。

圖六：繼電器模組(單路)外觀圖。

購買繼電器模組時須注意四點：一是先決定模組之採購種類到底是單路、雙路、四路或八路，越多路者可控制之家電數目也就越多，使用者可視需求選擇適合之種類；二是再確認該繼電器「控制端」之輸入電壓是否為5V，如圖四藍色繼電器上紅框內所示之印刷字樣5VDC，切勿買錯，若誤拿12V或110V字樣之繼電器，未來Arduino之5V接腳將無能力提供足夠之電源電壓，會導致推不動「受控端」；三是勿接太多路繼電器，亦即一個數位輸出腳無法同時控制多路IN訊號腳，因為Arduino輸出腳供電能力為40mA，而每個繼電器模組之光耦合隔離電路之觸發電流至少需5mA；四則是要確定係正邏輯5V作動還是負邏輯0V作動，因為各家廠牌模組不盡相同，甚至有些模組上附有跳針，可讓使用者自由選擇正邏輯或負邏輯來觸發作動。



圖七：智慧家居開關盒整體配線圖。

2.2 智慧家居開關盒整體配線圖

在本智慧家居開關盒之專題製作成品的整體配線圖(如圖七)中，LCD 1602文數字型液晶顯示模組是直接利用基座keypad shield插在Arduino Mega 2560微電腦板之上，HC-06藍牙通訊介面模組以及四路繼電器模組則利用小塊麵包板與連接線相連，測試成功後再將麵包板更換為實際銲接之洞洞板即可。

2.3 藍牙通訊技術

HC-06藍牙通訊介面模組為手機與Arduino之間溝通的橋梁[3]，當手機App程式用藍牙發出定時指令，Arduino利用HC-06接收指令做出反應。因此，藍牙通訊技術可再細分成二項技術，一是用App Inventor撰寫發射端的手機藍芽通訊之積木方塊程式，二則是接收端的Arduino藍芽指令接收程式，細節分述如後。

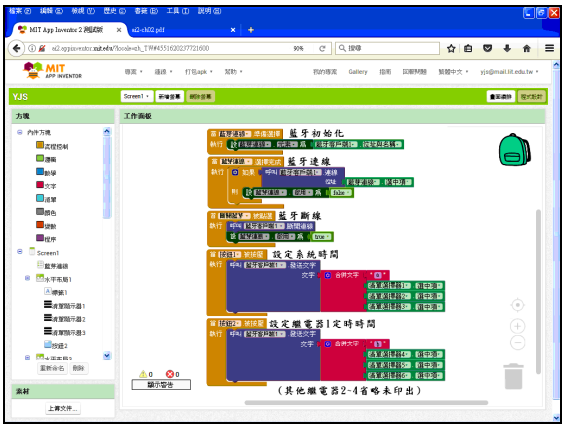
2.3.1 發射端手機藍芽通訊之積木方塊程式

首先須進入http://ai2.appInventor.mit.edu麻省理工學院網站[4]，登錄帳號後再新增一個專案；先於右上角選擇「畫面編排」模式(如圖八)，將左側「元件面板」之元件依序拖曳到中間「工作面板」之手機螢幕區，以便設計手機App畫面，例如：拖曳「通訊」類之不可視元件「藍牙客戶端」、「介面配置」類之可視元件「水平配置」、「使用者介面」類之可視元件「標籤」、「清單顯示器(元素字串之數字以逗號隔開，時為0~23、分與秒為0~59)」及「按鈕」…等元件，各個元件經過諸如：拖曳定位、尺寸調整、輸入文字、變更顏色…等微調後，專屬之手機定時App畫面之外觀設計便完成，其功能包含可設定系統時間或設定四個繼電器之定時時間。

圖八：App Inventor「畫面編排」模式。

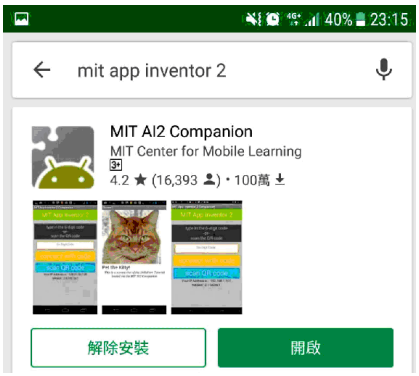
接著再於右上角選擇「程式設計」模式(如圖九)，將左側「方塊」之元件依序拖曳到中間「工作面板」之方塊積木程式撰寫區，依序加入例如：「藍牙連線」類棕色方塊之「當、準備選擇、執行」及「當、選擇完成、執行」、「藍牙連線」類綠色方塊之「設、藍牙連線、元素、為」及「設、藍牙連線、啟用、為」、「按鈕」類棕色方塊之「當、被壓下、執行」、「藍

牙客戶端」類紫色方塊之「呼叫、連線、地址」、「呼叫、發送文字、文字」...等程式，各個方塊經過諸如：拖曳定位、嵌入組合、下拉選項點選...等微調後，專屬之手機App程式功能便撰寫完成。



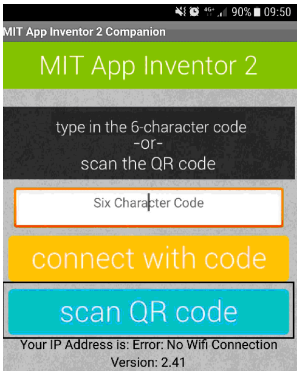
圖九：App Inventor「程式設計」模式。

當「畫面編排」與「程式設計」均完成後，便可進行手機App程式安裝，首先在手機內進入play商店，並搜尋關鍵字"MIT App Inventor"進行下載及安裝MIT AI2 Companion之App輔助軟體(如圖十)，安裝完成後，接著回到電腦App Inventor點選「打包apk並顯示二維條碼」(如圖十一)，在手機中再次執行安裝好的Companion程式，點選scan QR code功能(如圖十二)，再掃描電腦螢幕上的QR Code並點選安裝，則專屬自定之App藍芽通訊程式便成功安裝至手機。



圖十：play商店安裝MIT AI2 Companion手機畫面。

圖十一：打包專屬App的QR code。



圖十二：掃描專屬App的QR code進行下載與安裝。

2.3.2 接收端之Arduino藍芽指令接收程式

藍牙通訊介面模組HC-06因採I2C接線方式，需先以include巨集指令來含括SoftwareSerial及Wire二個函式庫，並主動宣告以生成一個I2CBT之藍牙物件，同時表明Arduino端接收腳RX及傳送腳TX，以分別反向對應藍芽模組端的TX及RX接腳。在setup()初始設定主函數內，需以9600之鮑率設定藍牙通訊介面模組，另需在loop()持續運行主函數內，宣告一字元組陣列變數cmd來接收手機端指令，以及宣告一整數變數size來記錄某指令之字元組長度。當不停地呼叫藍牙available()功能時，若字元組長度之傳回值大於0，表示有手機端指令進來了，則可用一簡單for迴圈，藉由藍牙read()功能逐個將字元組以char鑄型為字元，依次存到cmd指令陣列變數中，基礎骨架之程式碼片段如下所示：

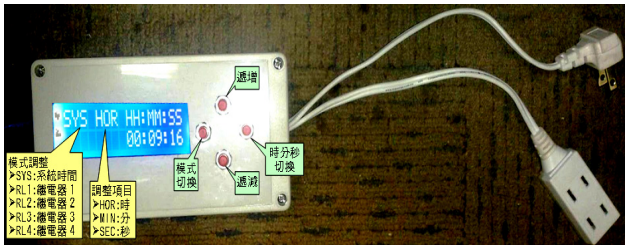
```
#include <SoftwareSerial.h>
#include <Wire.h>
SoftwareSerial I2CBT(RX, TX);

void setup()
{
   I2CBT.begin( 9600 );
}

void loop()
{
   byte  cmd[10];
   int   size;


   if((size=(I2CBT.available())) > 0)
   {
      for(int i=0 ; i<size ; i++)
      {
         cmd[i] = char(I2CBT.read());
      }
   }

   switch( cmd[0] )
   {
      case  97:            //a : 設定系統時間
      case  98:            //b : 設定繼電器時間
   }
}
```

值得一提的是，loop()主執行程序中，尚需每分每秒將系統時間之秒數進位，秒滿60則進位到分，分滿60則進位到時；若任一瞬間的系統時間之時、分、秒數值等於繼電器設定開啟家電的設定時間之時、

分、秒數值，則表示繼電器設定時間已到，可以供電到家電插座了[5]。

2.4 智慧家居開關盒之成品成果

所研究開發之手機藍牙遙控智慧家居開關盒，其成品實體之盒面左側為LCD 1602文數字型液晶顯示模組(如圖十三)，盒面右側為四顆硬體之設定按鈕，右邊則為交流110V電源插頭，以及繼電器1之受控插座，繼電器2~3受控插座則暫時省略不安裝。



圖十三：手機藍牙遙控智慧家居開關盒成品圖。

2.5 智慧家居開關盒之控制程式

茲將智慧家居開關盒之Arduino控制程式加上註解後，完整呈現如下：

```
/************ 指令開頭字母說明 ***************/
/* a(系統時間SYS)                             */
/* b(繼電器1 RL1)                             */
/* c(繼電器2 RL2)                             */
/* d(繼電器3 RL3)                             */
/* e(繼電器4 RL4)                             */
/*************** ASCII字母表 *****************/
/*  97a  98b  99c 100d 101e 102f 103g         */
/* 104h 105i 106j 107k 108l 109m 110n         */
/* 111o 112p 113q 114r 115s 116t 117u         */
/* 118v 119w 120x 121y 122z                   */
/*                                            */
/* 48-0 49-1 50-2 51-3 52-4                   */
/* 53-5 54-6 55-7 56-8 57-9                   */
/**********************************************/
///// 載入液晶與藍芽模組 /////
#include <LiquidCrystal.h>
#include "LCDKeypad.h"
#include <SoftwareSerial.h>
#include <Wire.h>

///// 液晶板&藍芽板物件 /////
LCDKeypad      LCDKP;
SoftwareSerial I2CBT(50, 51);

unsigned int  mode1=0, mode2=0;
///// 模式1 /////
#define SYS 0
#define RL1 1
#define RL2 2
#define RL3 3
#define RL4 4
///// 模式2 /////
#define HOR 0
#define MIN 1
#define SEC 2

///// 系統時間&四繼電器時間 /////
unsigned int  hor[5];
unsigned int  min[5];
unsigned int  sec[5];

void setup()
{
  ///// 液晶板設定 /////
  LCDKP.begin(16, 2);               //16字2列
  LCDKP.setCursor(0, 0);            //游標至左上角
  LCDKP.print("SYS HOR HH:MM:SS");  //上排顯示
                //0123456789012345

  for(int i=0 ; i<5 ; i++)      //1系統&4繼電器之時間歸零
  {
     hor[i] = 0;
     min[i] = 0;
     sec[i] = 0;
  }

  ///// 藍牙板設定 /////
  Serial.begin( 9600 );     //主板串列鮑率
   I2CBT.begin( 9600 );     //藍牙板之鮑率
  pinMode(37, OUTPUT);      //繼電器控制腳位為輸出
  pinMode(35, OUTPUT);
  pinMode(33, OUTPUT);
  pinMode(31, OUTPUT);
  digitalWrite(37, HIGH);  //先關閉四繼電器
  digitalWrite(35, HIGH);
  digitalWrite(33, HIGH);
  digitalWrite(31, HIGH);
}

///// 時間遞增溢位控制 /////
void incTime()
{
  sec[0]++;
  if(sec[0] == 60) { sec[0]=0;  min[0]++;
   if(min[0] == 60) { min[0]=0;  hor[0]++;
    if(hor[0] == 24) { hor[0]=0; }}}
}

///// 列印模式1標題 /////
void printMode1()
{
  LCDKP.setCursor(0, 0);
  switch( mode1 )
  {
     case SYS: LCDKP.print("SYS");  break;
     case RL1: LCDKP.print("RL1");  break;
     case RL2: LCDKP.print("RL2");  break;
     case RL3: LCDKP.print("RL3");  break;
     case RL4: LCDKP.print("RL4");
  }
}
///// 列印模式2標題 /////
void printMode2()
{
  LCDKP.setCursor(4, 0);
  switch( mode2 )
  {
     case HOR: LCDKP.print("HOR");  break;
     case MIN: LCDKP.print("MIN");  break;
     case SEC: LCDKP.print("SEC");
  }
}

///// 列印時間 /////
void printTime()
{
  LCDKP.setCursor(8,1);     //游標停於第2列

  char  s[9];
  sprintf(s, "%02i:%02i:%02i",
             hor[mode1],min[mode1],sec[mode1]);
  LCDKP.print( s );
}

void  readButton()
{
 int b = LCDKP.button();   //讀取按鍵
 switch( b )
 {
  case KEYPAD_LEFT :        //左鍵被押(切換模式1)
   mode1++;
   mode1 %= 5;
   break;
  case KEYPAD_RIGHT :       //右鍵被押(切換模式2)
   mode2++;
   mode2 %= 3;
   break;
  case KEYPAD_UP :          //上鍵被押(遞增時間)
   switch( mode2 )
   {
```

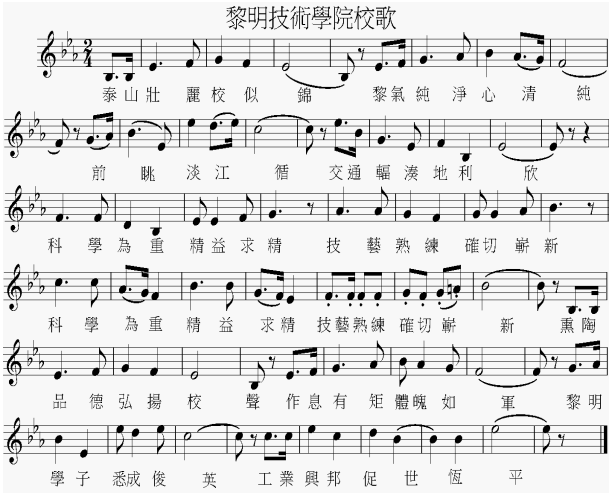
```
      case HOR: hor[mode1]++; hor[mode1]%=24; break;
      case MIN: min[mode1]++; min[mode1]%=60; break;
      case SEC: sec[mode1]++; sec[mode1]%=60; break;
    }
   case KEYPAD_DOWN :        //下鍵被押(遞減時間)
    switch( mode2 )
    {
       case HOR: hor[mode1]--;
                 if(hor[mode1]==-1) hor[mode1]=23;
                 break;
       case MIN: min[mode1]--;
                 if(min[mode1]==-1) min[mode1]=59;
                 break;
       case SEC: sec[mode1]--;
                 if(sec[mode1]==-1) sec[mode1]=59;
    }
  }
}

///// (每秒)按鍵偵測&動作 /////
void buttonListen()
{
  for(int i=0 ; i<5 ; i++) //每秒讀鍵5次
  {
    readButton();
    printMode1();
    printMode2();
    printTime();
    while(millis()%200 != 0); //等5分之1秒
  }
}

void loop()
{
  byte  cmd[20];
  int   size;

  while( 1 )
  {
    ///// 液晶板之畫面更新 /////
    incTime();                //遞增秒數
    printTime();              //液晶板列印時間
    buttonListen();           //液晶板讀取按鍵(每秒)

    ///// 藍芽板接收手機指令 /////
    if((size=(I2CBT.available())) > 0)
    {
      Serial.print( "input size = " );
      Serial.println( size );
      for(int i=0 ; i<size ; i++)
      {
        Serial.print(cmd[i]=char(I2CBT.read()));
        Serial.print(" ");
      }
    }

    switch( cmd[0] )
    {
      case  97:   //a：設定系統時間SYS
        hor[0] = (cmd[1]-48)*10 + (cmd[2]-48);
        min[0] = (cmd[3]-48)*10 + (cmd[4]-48);
        sec[0] = (cmd[5]-48)*10 + (cmd[6]-48);
        break;
      case  98:   //b：設定繼電器1時間RL1
        hor[1] = (cmd[1]-48)*10 + (cmd[2]-48);
        min[1] = (cmd[3]-48)*10 + (cmd[4]-48);
        sec[1] = (cmd[5]-48)*10 + (cmd[6]-48);
        break;
      case  99:   //c：設定繼電器2時間RL2
        hor[2] = (cmd[1]-48)*10 + (cmd[2]-48);
        min[2] = (cmd[3]-48)*10 + (cmd[4]-48);
        sec[2] = (cmd[5]-48)*10 + (cmd[6]-48);
        break;
      case 100:   //d：設定繼電器3時間RL3
        hor[3] = (cmd[1]-48)*10 + (cmd[2]-48);
        min[3] = (cmd[3]-48)*10 + (cmd[4]-48);
        sec[3] = (cmd[5]-48)*10 + (cmd[6]-48);
        break;
      case 101:   //e：設定繼電器4時間RL4
        hor[4] = (cmd[1]-48)*10 + (cmd[2]-48);
        min[4] = (cmd[3]-48)*10 + (cmd[4]-48);
        sec[4] = (cmd[5]-48)*10 + (cmd[6]-48);
        break;
```

```
    }
    if(hor[0]==hor[1] &&
       min[0]==min[1] &&
       sec[0]==sec[1]) digitalWrite(37, LOW);
    if(hor[0]==hor[2] &&
       min[0]==min[2] &&
       sec[0]==sec[2]) digitalWrite(35, LOW);
    if(hor[0]==hor[3] &&
       min[0]==min[3] &&
       sec[0]==sec[3]) digitalWrite(33, LOW);
    if(hor[0]==hor[4] &&
       min[0]==min[4] &&
       sec[0]==sec[4]) digitalWrite(31, LOW);
  }
}
```

3. 黎明校歌蜂鳴器演奏

以Arduino微電腦自撰程式來演奏音樂[6]，一般係指發出單音之曲目，因為Arduino內部計時器具有無法同時震盪輸出給任兩腳的缺點，很可惜沒辦法用Arduino 做和絃，故本研究暫時不討論和弦或立體聲之高階技術問題；通常演奏單音曲目，均採用紙盆喇叭或線圈蜂鳴器之零組件，並供應一定震盪頻率之電壓，使喇叭不停發出"喀搭"聲，隨著電壓變動頻率越高，則演奏音調越高，反之頻率越低，則演奏音調越低，因此鋼琴各個黑白琴鍵均有各自之音階頻率。

而本校校歌曲調抑揚頓挫、橫跨多音度，節拍輕快緊湊、快慢不一，故樂理上解析其音調與節拍頗為困難繁瑣(如圖十四)。



圖十四：黎明校歌之樂譜。

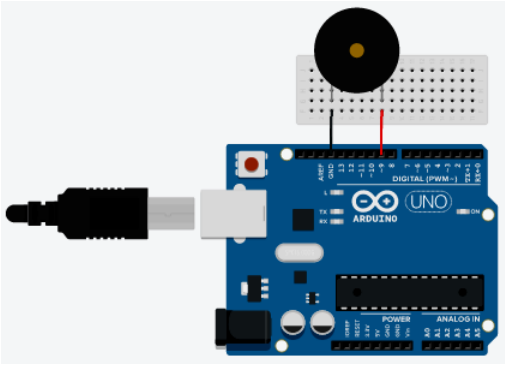
3.1 蜂鳴器零組件介紹

市面上所販售之蜂鳴器概分二種(如圖十五)；左圖為「有源蜂鳴器」，亦即是有固定震盪頻率之來源，外觀是澆封起來的，電壓極性有正負之分，且因有振盪電路之故，供應電源電壓值為恆定，其音調只能發出某一固定音調；右圖則為本研究所使用之「無源蜂鳴器」，亦即無震盪來源，需純粹仰賴外界輸入交變電壓訊號，其外觀明顯附有電路板，電壓極性無正負之分，且因當喇叭之故，輸入聲音訊號之電壓值為交變，因此才能夠發出多音頻之音調。

圖十五：(左)有源蜂鳴器　(右) 無源蜂鳴器。

3.2 蜂鳴器演奏之整體配線圖

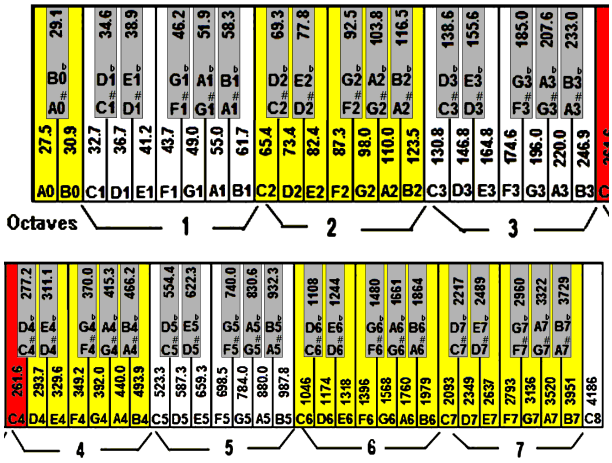
因只接一顆無源蜂鳴器，所以電路圖格外簡單(如圖十六)，只要將Arduino之GND腳連到蜂鳴器任一腳，再將Arduino第9號數位輸出腳連到蜂鳴器剩餘之另一腳，即完成接線。



圖十六：蜂鳴器演奏之整體配線圖。

3.3 蜂鳴器演奏之樂理簡介

欲讓蜂鳴器演奏單音調，首先須連續以特定頻率送出高電位5V及低電位0V各一次，使蜂鳴器能夠連續發出振動；若參考鋼琴各琴鍵之振動頻率(如圖十七)，則第四大調C4之中央Do音，其振動頻率可查得為261.6 Hz。



圖十七：鋼琴各琴鍵之振動頻率。

Arduino控制各個音階之振動頻率時，須先計算該音調每次高低電位間之延滯時間，也就是蜂鳴器振動一周期所包含高電位與低電位頻率作動之時間，再配合delayMicroseconds(微秒數 μs)之指令做精密延遲發聲，以上述C4音調為例，其發生高低電位程式碼如下所示，其週期時間計算方法如式1所示：

```
digitalWrite(pin, HIGH);
delayMicroseconds( period );
digitalWrite(pin, LOW);
delayMicroseconds(period );
```

$$Do\text{高及低電位之週期：}T_{C4}=\frac{1}{2\times f_{C4}}=\frac{1}{2\times 262\ \text{Hz}}\times 10^{6}=1908\ \mu s \qquad (1)$$

如懶得計算每個音符的高低電位作動間隔時間，則可以用內建函數tone(腳位,頻率,毫秒)之指令發聲，再以noTone(腳位)來噤聲。

在樂理方面，有二件事項必須注意，第一項是升降階之樂理，由降記號$b^{b}{}_{b}$坐落五線譜之位置，得知Si、Mi及La三個音符皆須降半階，也就是無論高八度還是低八度之B、E及A之音符群，都要由白鍵左移到旁邊的黑鍵，唯一例外是降半階音符前方如果又擺了一個升記號♮，則該音符降升互相抵銷。

第二項則是節拍之樂理，其中$\frac{2}{4}$表示「以四分音符為1拍、每小節有2拍」，意思是各小節內所有音符加總起來為2拍；此外不管音符正擺或倒擺，全音符𝅝為4拍、二分音符𝅗𝅥為2拍、四分音符♩為1拍、八分音符♪為½拍、十六分音符𝅘𝅥𝅯則為¼拍，任二音符尾部羽毛可互相連結成雙音符，若音符頭部右側帶黑點則須再加上本身½拍數；又四分休止符𝄽停頓1拍、八分休止符𝄾停頓½拍，休止符亦需一併納入小節內總拍數之計算。

3.4 黎明校歌蜂鳴器演奏之控制程式

遵守前節所述音調與節拍之樂理規則，首先定義白鍵與黑鍵之音調頻率以及節拍，之後便可以開始解析樂譜上各個音符，整首曲子含休止符經計算後共有118個音符，故宣告音調陣列note[118]及節拍陣列beat[118]如下：

```
///// 白鍵 /////
#define  C3   131
#define  D3   147
#define  E3   165
#define  F3   175
#define  G3   196
#define  A3   220
#define  B3   247

#define  C4   262
#define  D4   294
#define  E4   330
#define  F4   349
#define  G4   392
#define  A4   440
#define  B4   494

#define  C5   523
#define  D5   587
#define  E5   659
#define  F5   698
#define  G5   784
#define  A5   880
#define  F5   988
```

```
///// 黑鍵 /////
#define  D3b  139
#define  E3b  156
#define  G3b  185
#define  A3b  208
#define  B3b  233

#define  D4b  277
#define  E4b  311
#define  G4b  370
#define  A4b  415
#define  B4b  466

#define  D5b  554
#define  E5b  622
#define  G5b  740
#define  A5b  831
#define  B5b  932

const int  spk=9;

///// 音調陣列 /////
int  note[118] = {
  B3b,B3b,E4b,F4,G4,F4,E4b,B3b,0,  E4b,F4,G4,A4b,B4b,A4b,G4,F4,F4,0,
//泰 山 壯 麗校似  錦    黎 氣純 淨 心  清  純
  G4,A4b,B4b,E4b,E5b,D5,E5b,C5,C5,0,  E5b,B4b,G4,E4b,F4,B3b,E4b,E4b,0,0,
//前眺 北 市 淡  江  循   交 通 輻 湊 地 利  欣
  F4,F4,D4,B3b,  E4b,E4b,F4,G4,0,  A4b,A4b,G4,F4,  G4,G4,A4b,B4b,0,
//科學為 重  精 益 求精  技 藝 熟練  確切 嶄 新
  C5,C5,A4b,G4,F4,  B4b,B4b,G4,F4,E4b,  F4,F4,F4,F4,G4,F4,G4,A4,B4b,B4b,0,
//科學  為 重  精 益  求 精  技藝熟練 確切  嶄   新
  B3b,B3b,E4b,F4,G4,F4,E4b,B3b,0,  E4b,F4,G4,A4b,B4b,A4b,G4,F4,F4,0,
//熏 陶 品 德弘揚 校 聲    作 息有 矩 體 魄 如 軍
  G4,A4b,B4b,E4b,E5b,D5,E5b,C5,C5,0,  C5,C5,E5b,C5,D5,B4b,B4b,B4b,E5b,E5b ,0};
//黎明 學 子 悉 成 俊  英    工業 興 邦促   世  恆   平

///// 節拍陣列 /////
int  beat[118] = {
  3,  1,  6,  2,4,4,   8,2,2,   3,  1,6,  2,  4,   3,1,8,2,2,
//泰 山 壯 麗校似  錦    黎 氣純 淨 心  清  純
  3,1,  6,  2,  4,   3,1,  8,2,2,  3,  1,  6,  2,  4,  4,  8,2,2,4,
//前眺 北 市 淡  江  循   交 通 輻 湊 地 利  欣
  6,2,4,  4,   2,  4,  2,6,2,  6,  2,  4,4,   2,4,  2,  6,2,
//科學為 重  精 益 求精  技 藝 熟練  確切 嶄 新
  6,2,  3,1,4,   6,  2,   3,1,4,   3,1,2,2,  2,2,  2,2,  8,2,2,
//科學  為 重  精 益  求 精  技藝熟練 確切  嶄   新
  3,  1,  6,  2,4,4,  8,  2,2,   1,  3,6,  2,  2,  4,  2,  8,2,2,
//熏 陶 品 德弘揚 校 聲    作 息有 矩 體 魄 如 軍
  3,1,  4,  4,  2,  4,  2,  8,2,2,  3,1,  4,  4,4,   4,4,4,  8,2,2};
//黎明 學 子 悉 成 俊  英    工業 興 邦促   世  恆   平

void setup()
{
  pinMode(spk, OUTPUT);
}

void myTone(int freq, int duration)
{
  tone(spk, freq, duration);
  delay( duration );
  noTone( spk );
}

void loop()
{
   for(int i=0 ; i<118 ; i++) myTone(note[i], beat[i]*100);
}
```

在自定發聲函數myTone(頻率,毫秒)的內容中，原本僅有發聲指令tone(腳位,頻率,毫秒)這一行，但因為Arduino送出該發聲指令便馬上執行下一行，無暇檢查是否確實執行完畢，因此本研究特別增加delay(毫秒)指令來等待發聲結束，否則後續重疊會發生無聲或者頻率混亂之雜音狀況，最後再補上noTone(腳位) 讓上一音符噤聲，以免干擾下一音符的發聲。

4. 超音波測距汽車電動尾門

第三項專題研究成品，係把傳統之汽車後尾門進行創意改造，利用繼電器模組、線性電動缸、超音波感測器、無源蜂鳴器、啟動開關及極限開關...等零組件，改裝成能夠自動啟閉、具超音波感測障礙物保護防撞功能之智慧電動尾門。

在操作控制邏輯上，第一種係處於正常操作狀態下，壓一下啟動開關，因內側極限開關被按壓，因此微型電動缸便頂出尾門，直至外側極限開關被按壓，尾門頂出動作便停止；若再壓一下啟動開關，因外側極限開關被按壓，因此微型電動缸便把尾門縮回，直至內側極限開關被按壓，尾門縮回動作便停止。

第二種係處於異常操作狀態下，起初碰觸起動開關令尾門開啟，若尾門打開時超音波檢出障礙物，則微型電動缸便停止頂出，尾門開啟動作暫停，同時蜂鳴器發出警戒旋律；一旦障礙物被排除，則蜂鳴器噤聲，尾門開啟動作繼續；反之，若障礙物無法被排除，則須再次觸摸啟動開關，以便讓微型電動缸縮回，以令尾門關上，俾利車主重新移車至其他安全地點。

4.1 超音波測距汽車電動尾門之零組件介紹

因繼電器模組及無源蜂鳴器在前面章節已介紹過，茲僅就超音波感測器及線性電動缸做細部介紹。

4.1.1 HC-SR04超音波感測器

HC-SR04超音波感測器(如圖十八)，有兩顆金屬圓盒，分別是左側標示T之發射頭與右側標示R之接收頭，分別對應四個腳位中之觸發腳Trig與回音腳Echo，剩下二腳為感測器5V電源腳Vcc及接地腳GND。欲超音波感測器模組工作時，假想與模擬蝙蝠發射與接收之機制一般，首先須以Arduino數位腳於極短時間內(如1毫秒)、對觸發腳輸出5V電壓後馬上斷電，再於接收腳接收到信號電壓之後，利用pulseIn(接腳, 上緣/下緣觸發電壓準位)之指令來得到超音波回彈所需時間duration，其基礎骨架之程式碼片段如下所示[7]：

圖十八：HC-SR04超音波感測器。

```
///// 超音波初始化 /////
pinMode(trig, OUTPUT);
pinMode(echo, INPUT );

///// 超音波測距 /////
digitalWrite(trig, HIGH);
delayMicroseconds(1000);
digitalWrite(trig, LOW);
duration = pulseIn(echo, HIGH);
distance = (duration / 2) * 0.034;
```

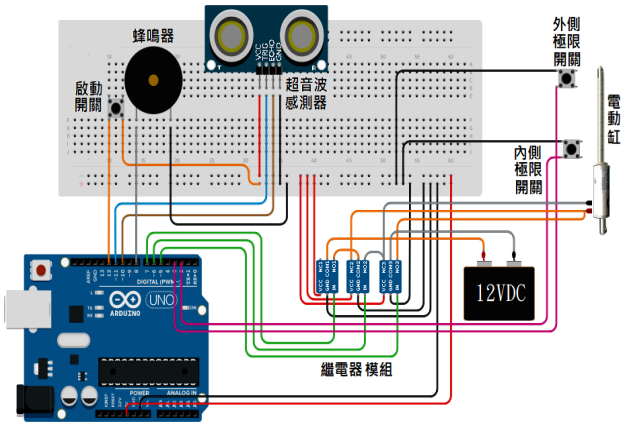
知道超音波回彈所需時間之後，才能利用音速傳遞公式(如式2)推算出距離distance，舉例來說在氣溫15℃時，套用音速公式為331+0.6×15 = 340 m/sec = 34000 cm/sec = 0.034 cm/μs，又因為音波是來回往返2倍距離之故，所以把回彈所需時間代入公式之前，還要先再除以2，才能把開啟尾門時之實際障礙物距離真實反映出來。

音速公式：$v = 331 + 0.6 \cdot t℃ \quad (m/sec)$ (2)

4.1.2 線性電動缸

線性電動缸(又稱推桿缸，如圖十九)通常使用12V或24V直流無刷馬達，左側為本研究所使用之微型線性電動缸，其優點是筆型體積不占空間、缺點是功率甚少超出100W，且伸縮時兩端需於插銷處固定，否則推桿會原地打轉導致無法伸縮，且市售商品甚少附有兩端保護之極限開關；右側則是附帶減速齒輪之重負載型線性電動缸，因馬達被獨自分離在旁，因轉動扭力經過減速齒輪機構放大之故，最大推力或拉力通常能輕易超過100Kg，高級品則會附帶近端及遠端內外極限，以防止過度伸縮造成減速齒輪或螺牙伸縮機構之損傷；在使用線性電動缸時，僅需將供電電源之正負極顛倒，即可進行正轉或反轉(即伸長或縮短)。

圖十九：線性電動缸 (左)微型　(右)重負載型。

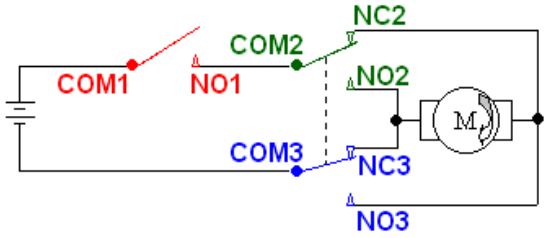


圖二十：電動尾門之整體配線圖。

4.2 超音波測距汽車電動尾門之整體配線圖

本專題研究之整體配線圖(如圖二十)採用Arduino UNO微電腦版，周邊零件搭配一顆啟動開關(2腳：接第12號數位輸入腳、接地GND)、一顆超音波感測器(4腳：5V電源、觸發TRIG接第11號數位輸出腳、回音ECHO接第10號數位輸入腳、接地GND)、三顆繼電器模組(3訊號IN分別接到第7、6、5號數位輸出腳)、一顆蜂鳴器(2腳：第8號數位輸出腳、接地GND)、二顆極限開關(內側極限2腳：第2號數位輸入腳、接地GND；外側極限2腳：第3號數位輸入腳、接地GND)。

此外為能順利伸縮微型電動缸，利用前述之三顆繼電器模組來製作馬達正反轉電路(如圖二十一)，整體配線圖左邊繼電器1係控制馬達啟閉，中間及右邊之繼電器2與繼電器3構成一雙刀雙投(DPDT)之閘刀開關；當繼電器1作動時，NO1開接點導電，馬達開始運轉；當繼電器2與繼電器3均無作動時，NC2及NC3閉接點導電，馬達正轉；當繼電器2與繼電器3均作動時，NO2及NO3開接點導電，馬達反轉。



圖二十一：電動缸之馬達正反轉電路圖。

4.3 電動尾門之控制程式

因不正常的尾門啟閉可能導致人員傷亡，本程式碼須嚴格區隔微型電動缸之各類運動狀態，除了確保目前動作是否完成外，也有利於規畫下一動作，例如尾門位置pos、馬達運動方向dir、作動狀態mod三個類型參數：馬達位置有分為近端CLOSED、中間MIDDLE及遠端EXTEND；馬達運動方向有分為前進FOREWARD、後退BACKWARD及停止STOPWARD；超音波檢測障礙物之作動狀態有分為安全SAFE及干涉COLLIDE等。

程式在每次執行迴圈loop()一開始，便偵測啟動開關btn、內側極限開關in及外側極限開關out之三個開關之按押狀態，並依照下列原則作動，其中項次(1)與(2)為按押啟動開關之馬達驅動反應、項次(3)與(5)為抵達目的位置之馬達停止反應、項次(4)為開啟尾門時遇到障礙物之反應、項次(6)則為超音波測距並設定安危狀態：

(1) 若按鈕啟動開門、且馬達停止、且處在近端位置、且避障為安全狀態，則馬達正轉將尾門頂出。
(2) 若按鈕啟動關門、且馬達停止、且處在遠端位置、且避障為安全狀態，則馬達反轉將尾門縮回。
(3) 若頂到外側極限開關、且馬達運動方向為前進、且之前處在中間位置，則令馬達停止。
(4) 若按鈕啟動關門避障、且馬達運動方向為前進、

且之前處在中間位置、且避障為危險狀態，則令馬達反轉將尾門縮回以求躲避。

(5) 若頂到內側極限開關、且馬達運動方向為後退、且之前處在中間位置，則令馬達停止。

(6) 若之前處在中間位置、且馬達運動方向為前進，則偵測障礙物距離；若距離小於5公分，則馬上停止繼電器1之馬達運轉，並將避障狀態設為危險；反之，若距離大於5公分，則馬上啟動繼電器1之馬達運轉，並將避障狀態設為安全。

```
int  btn, in, out;

///// 定義位置 /////
#define  CLOSED  1
#define  MIDDLE  2
#define  EXTEND  3
int  pos = CLOSED;

///// 定義方向 /////
#define  FOREWARD 4
#define  BACKWARD 5
#define  STOPWARD 6
int  dir = STOPWARD;

///// 定義狀態 /////
#define  SAFE     7
#define  COLLIDE  8
int  mod = SAFE;

const int start   = 12;   //啟動開關腳
const int trig    = 11;   //超音波觸發腳
const int echo    = 10;   //超音波回音腳
const int relay1  =  7;   //馬達啟閉繼電器腳
const int relay2  =  6;   //正反轉繼電器一腳
const int relay3  =  5;   //正反轉繼電器二腳
const int speaker =  8;   //喇叭腳
const int inner   =  2;   //關門內極限開關腳
const int outter  =  3;   //開門外極限開關腳

float  distanceOld, distanceNew, duration;

void setup()
{
  ///// 超音波初始化 /////
  pinMode (  trig, OUTPUT);
  pinMode (  echo, INPUT );
  Serial.begin(9600);
  ///// 啟動按鈕初始化 /////
  pinMode ( start, INPUT );  digitalWrite(start, HIGH);
  ///// (負邏輯)繼電器初始化 /////
  pinMode (relay1, OUTPUT);  digitalWrite(relay1, HIGH);
  pinMode (relay2, OUTPUT);  digitalWrite(relay2, HIGH);
  pinMode (relay3, OUTPUT);  digitalWrite(relay3, HIGH);
  ///// 內、外極限開關初始化 /////
  pinMode ( inner, INPUT );  digitalWrite( inner, HIGH);
  pinMode (outter, INPUT );  digitalWrite(outter, HIGH);
  ///// 喇叭初始化 /////
  pinMode(speaker, OUTPUT);
}

void loop()
{
   btn = digitalRead( start  );
   in  = digitalRead( inner  );
   out = digitalRead( outter );

   ///// 按鈕啟動開門(馬達正轉) /////
   if(btn==LOW && dir==STOPWARD && pos==CLOSED && mod==SAFE)
   {
      dir = FOREWARD;
      pos = MIDDLE;
      digitalWrite(relay1, LOW);
      digitalWrite(relay2, LOW);
      digitalWrite(relay3, LOW);
   }
   ///// 按鈕啟動關門(馬達反轉) /////
   if(btn==LOW && dir==STOPWARD && pos==EXTEND && mod==SAFE)
   {
      dir = BACKWARD;
      pos = MIDDLE;
      digitalWrite(relay1, LOW);
      digitalWrite(relay2, HIGH);
      digitalWrite(relay3, HIGH);
   }
   ///// 開門觸到外極限開關(馬達停止) /////
   if(out==LOW && dir==FOREWARD && pos==MIDDLE)
   {
      dir = STOPWARD;
      pos = EXTEND;
      digitalWrite(relay1, HIGH);
   }
   ///// 危險狀態關門(馬達反轉) /////
   if(btn==LOW && dir==FOREWARD && pos==MIDDLE && mod==COLLIDE)
   {
      dir = BACKWARD;
      pos = MIDDLE;
      mod = SAFE;
      digitalWrite(relay1, LOW);
      digitalWrite(relay2, HIGH);
      digitalWrite(relay3, HIGH);
   }
   ///// 關門觸到內極限開關(馬達停止) /////
   if(in==LOW && dir==BACKWARD && pos==MIDDLE)
   {
      dir = STOPWARD;
      pos = CLOSED;
      digitalWrite(relay1, HIGH);
   }

   ///// 超音波測距 /////
   if(pos==MIDDLE && dir==FOREWARD)
   {
      digitalWrite(trig, HIGH);
      delayMicroseconds(1000);
      digitalWrite(trig, LOW);
      duration = pulseIn(echo, HIGH);
      distanceNew = (duration / 2) * 0.34;
      distanceNew = 0.5 * distanceOld + 0.5 * distanceNew;
      Serial.println(distanceNew);
      delay( 250 );

      if(distanceNew < 5)
      { digitalWrite(relay1,HIGH);  mod=COLLIDE; }
      else
      { digitalWrite(relay1,LOW );  mod=SAFE   ; }
   }

   ///// 超音波避障發聲程式 /////
   if(mod == COLLIDE)
   {
      tone(speaker, 440, 250);   delay(100);
      tone(speaker, 494, 250);   delay(100);
      tone(speaker, 523, 250);   delay(100);
   }
   else noTone( speaker );
}
```

5. 陀螺儀自動控制車燈投射角度裝置

目前市售高級房車，多半都有配備車頭大燈之手控電動調整，以利調整大燈對於地面投射之角度；本研究係利用陀螺儀機動感測車身前後傾角，再以步進馬達動態即時補償陡升或陡降地勢之偏差投射角。此外，為防止Arduino數位輸出腳推動步進馬達之電流過低、或車頭大燈燈座轉動慣性過重，造成步進馬達有收到驅動電壓卻扭矩不足因而失步，Arduino不直接接線至步進馬達，而是兩者間加了一片步進馬達驅動板，以確保驅動電流足夠，讓步進馬達具有足夠的正反轉扭矩。因受限於本款GY-801陀螺儀具有角度累積誤差之缺點，程式特色在於一開始通電後，會進行短暫數秒的角速度誤差值歸零之自我校正動作之外，還額外增加一馬達擺向微調開關，以利駕駛在長

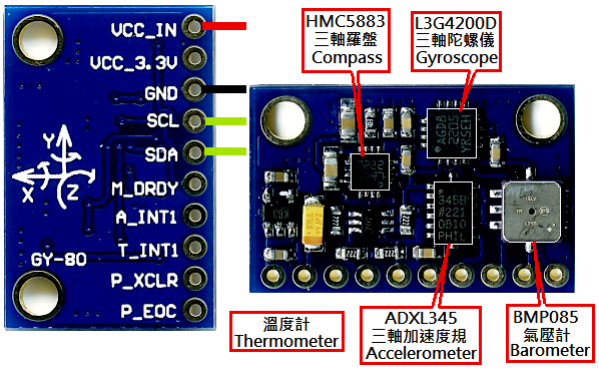
時間使用之後，還能人為調整投射角度之累積誤差。

5.1 自動控制車燈之零組件介紹

陀螺儀自動控制車燈投射角度裝置，其關鍵零組件包括最重要之九軸IMU陀螺儀模組GY-801，具有3軸磁場、3軸加速度、3軸角加速度陀螺儀、1大氣壓及1溫度感測；另外還有零組件步進馬達28BYJ-48-5V及ULN2003步進馬達驅動器之組合。

5.1.1 九軸IMU陀螺儀模組

陀螺儀模組(如圖二十二，購買價格約NT$350)，其不但功能眾多(如表二)、價格低廉，微機電功能具有11個物理參數之感測能力。



圖二十二：九軸IMU陀螺儀模組零件外觀圖。

表二：九軸IMU陀螺儀模組功能對照表。

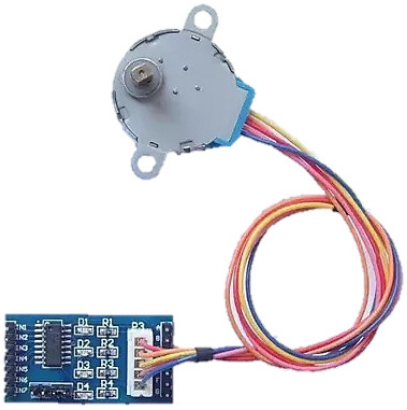
功能	執掌晶片	軸維度	物理意義	單位
電子羅盤 Magnetometer	HMC5883	3	地磁強度	高斯[CGS 制] (Gauss)
陀螺儀 Gyroscope	L3G4200	3	角速度值	度每秒 (dps)
加速度計 Accelerometer	ADXL345	3	加速度值	重力加速度 g (1024 分割)
氣壓計 Barometer	BMP085	1	氣壓值	大氣壓 (atm)
溫度計 Thermometer		1	氣溫值	攝氏 (℃)

在與Arduino硬體搭配使用時，僅須將陀螺儀模組之VCC_IN及GND連到Arduino之5V與地極，並將陀螺儀模組之串列時脈腳SCL及串列資料腳SDA連到Arduino之類比腳A5及A4，便可簡易完成硬體接線設定。又因為陀螺儀模組採用此種I2C接線方式，因此須預先下載函式庫ZIP壓縮檔，需先至GitHub公司網頁(網址：https://github.com/muggn/GY80)，點選右上角Clone or download綠色按鈕，以下載檔名為GY80-master.zip之陀螺儀函式庫壓縮檔；接著進入Arduino IDE整合發展環境內，由主功能表「Sketch」⇨「Include Library」⇨「Add .ZIP Library...」點選剛剛下載之壓縮檔，便完成陀螺儀函式庫之程式環境安裝。接著便可使用基礎骨架之程式碼片段做應用了，如下所示：

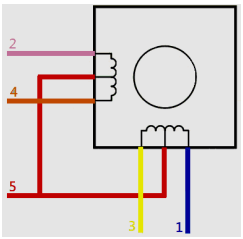
```
宣告：
  #include <GY80.h>
  GY80  sensor = GY80();
初始化設定：
  sensor.begin();
讀各項感測值：
  GY80_scaled  val = sensor.read_scaled;
  float ax = val.a_x,       //X軸加速度
        ay = val.a_y,       //Y軸加速度
        az = val.a_z;       //Z軸加速度
  float mx = val.m_x,       //X軸磁場強度
        my = val.m_y,       //Y軸磁場強度
        mz = val.m_z;       //Z軸磁場強度
  float gx = val.g_x,       //X軸角速度
        gy = val.g_y,       //Y軸角速度
        gz = val.g_z;       //Z軸角速度
  float  p = val.p,         //氣壓
         t = val.t;         //氣溫
```

5.1.2 步進馬達與步進馬達驅動器

眾所周知步進馬達有三種驅動方式[8]，分別為1相激磁、2相激磁及1-2相激磁，其三種驅動方式之解析角度比值分別為1:1:0.5，解析角度越小越好，扭矩比值為1:2:1.5，扭矩越大越好；本研究所採用之28BYJ-48-5V步進馬達(外觀如圖二十三、內部繞組如圖二十四)為二相四極五線(2-Phase 4-Pole 5-Wire Stepper Motor)構裝，五條電線顏色分別為藍、粉紅、黃、橘及紅，其中紅色線為共地端，減速齒輪比為1/64，使用5V脈波電壓驅動；若一相激磁走一步經過四次輪詢後，實測值一圈360°為512步，故單步之解析角度為360°/512=0.703125°，其最高轉速可達每分鐘80 rpm (revolution per minute)。



圖二十三：28BYJ-48-5V步進馬達組零件外觀圖。

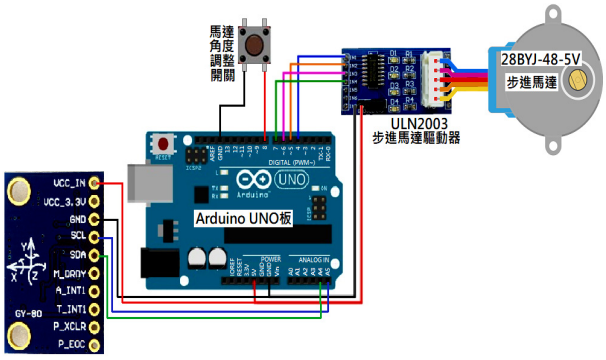


圖二十四：28BYJ-48-5V步進馬達繞組線路圖。

驅動時因Arduino數位輸出腳送出訊號的電流不足，其輸出電流能力僅40mA，故利用ULN2003步進馬達驅動器送出較大的驅動電流以避免馬達失步。ULN2003A晶片採用大電流共射極達林頓(Darlington)電晶體積體電路，7組達林頓電路之電流能力各為500mA，輸出電壓最高甚至可達50V。

5.2 車燈投射角度裝置之整體配線圖

本電動車燈之整體配線圖(如圖二十五)，其關鍵零組件包括陀螺儀模組、步進馬達、步進馬達驅動器以及角度歸零開關；茲先將陀螺儀模組之電源VCC_IN及GND分別接到Arduino板之5V及GND，再將陀螺儀模組之SCL及SDA分別接到Arduino板之A5及A4；角度歸零開關之二極分別接到Arduino板之8號數位輸出腳及GND;步進馬達之排線公頭則插入有防呆功能之母座；步進馬達驅動器之電源接到Arduino板之5V及GND之後，將驅動板之IN1、IN2、IN3、IN4四個訊號腳分別接到Arduino板之4號、5號、6號及7號數位輸出腳，即完成接線工作。



圖二十五：車燈投射角度裝置之整體配線圖。

5.3 車燈投射角度裝置之控制程式

使用GY-801陀螺儀具有四項先天缺陷；第一點是靜置不動之陀螺儀，所回報之角速度並不為零，表示存在先天初始誤差，若後續讀值不給予補償，將不可能產生準確角速度；第二點因須分分秒秒動態讀取感測之角速度，再利用時間積分將瞬時角速度積分成為相對角度差，也因此造成最終角度具有很大之累積誤差；第三點若陀螺儀以極緩慢速度轉動，會造成角速度零讀值，此困擾無解；第四點若陀螺儀遭受大力敲擊，將會跑出超大讀值，此先天困擾亦無解。

因陀螺儀具有上述先天缺陷，本研究使用程式手法克服部分缺陷；針對第一點初始誤差之缺陷，本程式在Arduino設定函數setup()中，增加5秒之初始自我校正功能，此時要求陀螺儀須在絕對靜止狀態下，於5秒內讀取角速度之連續漸變平均值，作為角速度之歸零校正值；針對第二點角度累積誤差之缺陷，本程式建議未來可以朝自動補正方式進行，亦即先將motorRun()傳入相對轉動角度改為傳入絕對轉動角度(0~360度)，再隨時偵測角速度讀值是否為零，表示車體俯仰變化趨緩，則可令車燈回復正常投射角度，但若要真正解決此問題則須引入二顆裝設於車體及車燈之陀螺儀，或是買更高檔之陀螺儀來徹底擺脫此缺陷；針對第四點之先天缺陷，僅能在機構上增加避震裝置來降低衝擊力對陀螺儀之影響；至於第三點缺陷則無解，僅能換較好之感測器了。

最後詳列車燈投射角度裝置之完整程式如下：

```
#include  <GY80.h>

#define  UPON   1              //馬達上揚
#define  DOWN  -1              //馬達下擺

const int  buton =  8;         //電動調整開關
const int  step1 =  4;         //步進相位A+腳
const int  step2 =  5;         //步進相位A-腳
const int  step3 =  6;         //步進相位B+腳
const int  step4 =  7;         //步進相位B-腳

GY80         compass = GY80();//陀螺儀
GY80_scaled  val;             //陀螺儀讀值

int          control = UPON;  //手動控制馬達初始方向
int          isPress;         //手動控制按鈕值
unsigned long  tNow, tDif, tPrv, tLim;
       //時間：現在、相差、前一、限制
float          zroX, dpsX, degX, degP;
   //(角)速度：歸零、角速、角度、前一

///// 馬達驅動(二相激磁) /////
void  motorRun(int dir, float  deg)
{
   float  resolution = 360.0/512.0;    //算步進馬達解析度
   int     steps     = deg/resolution; //總共步數計算

   if(dir == UPON)                     //馬達上揚
   {
      for(int s=1 ; s<=steps ; s++)
      {
         digitalWrite(step1, HIGH);
         digitalWrite(step2, HIGH);
         digitalWrite(step3, LOW );
         digitalWrite(step4, LOW );   delay( 2 );
         digitalWrite(step1, LOW );
         digitalWrite(step2, HIGH);
         digitalWrite(step3, HIGH);
         digitalWrite(step4, LOW );   delay( 2 );
         digitalWrite(step1, LOW );
         digitalWrite(step2, LOW );
         digitalWrite(step3, HIGH);
         digitalWrite(step4, HIGH);   delay( 2 );
         digitalWrite(step1, HIGH);
         digitalWrite(step2, LOW );
         digitalWrite(step3, LOW );
         digitalWrite(step4, HIGH);   delay( 2 );
      }
   }
   else if(dir == DOWN)                //馬達下擺
   {
      for(int s=1 ; s<=steps ; s++)
      {
         digitalWrite(step1, LOW );
         digitalWrite(step2, LOW );
         digitalWrite(step3, HIGH);
         digitalWrite(step4, HIGH);   delay( 2 );
         digitalWrite(step1, LOW );
         digitalWrite(step2, HIGH);
         digitalWrite(step3, HIGH);
         digitalWrite(step4, LOW );   delay( 2 );
         digitalWrite(step1, HIGH);
         digitalWrite(step2, HIGH);
         digitalWrite(step3, LOW );
         digitalWrite(step4, LOW );   delay( 2 );
         digitalWrite(step1, HIGH);
         digitalWrite(step2, LOW );
         digitalWrite(step3, LOW );
         digitalWrite(step4, HIGH);   delay( 2 );
      }
   }
}
```

```
void setup()
{
   pinMode(buton, INPUT);           //開關腳位設定
   digitalWrite(buton, HIGH);       //開關腳高電位

   pinMode(step1, OUTPUT);          //馬達腳位設定輸出
   pinMode(step2, OUTPUT);
   pinMode(step3, OUTPUT);
   pinMode(step4, OUTPUT);

   compass.begin();                 //羅盤初始化
   Serial.begin( 9600 );            //通訊字幕視窗連線鮑率

   ////////////////////////////////
   ///// 開機靜止5秒找-初始角速度 /////
   ////////////////////////////////
   tLim = micros();                 //限制時間
   for(zroX=0 ; ; )                 //進入無窮迴圈
   {
      tNow = micros();              //現在時間
      tDif = tNow - tLim;           //差異時間
      if(tDif < 5000000)            //5秒之靜止初始角速度
      {
         val = compass.read_scaled();  //讀陀螺儀
         zroX = 0.9*zroX + 0.1*val.g_x;//取連續漸變平均值
      }
      else break;                   //5秒後離開(校正完畢)
   }
   Serial.print("校正角速度:");
   Serial.println(zroX, 2);
}

void loop()
{
   isPress = digitalRead( buton ); //取得按鈕電壓

   ///// 按鈕手動控制馬達(按鈕有壓) /////
   if(isPress == LOW)
   {
      if(control == UPON)
      {
         digitalWrite(step1, HIGH);
         digitalWrite(step2, HIGH);
         digitalWrite(step3, LOW );
         digitalWrite(step4, LOW );   delay( 2 );
         digitalWrite(step1, LOW );
         digitalWrite(step2, HIGH);
         digitalWrite(step3, HIGH);
         digitalWrite(step4, LOW );   delay( 2 );
         digitalWrite(step1, LOW );
         digitalWrite(step2, LOW );
         digitalWrite(step3, HIGH);
         digitalWrite(step4, HIGH);   delay( 2 );
         digitalWrite(step1, HIGH);
         digitalWrite(step2, LOW );
         digitalWrite(step3, LOW );
         digitalWrite(step4, HIGH);   delay( 2 );
      }
      else if(control == DOWN)
      {
         digitalWrite(step1, LOW );
         digitalWrite(step2, LOW );
         digitalWrite(step3, HIGH);
         digitalWrite(step4, HIGH);   delay( 2 );
         digitalWrite(step1, LOW );
         digitalWrite(step2, HIGH);
         digitalWrite(step3, HIGH);
         digitalWrite(step4, LOW );   delay( 2 );
         digitalWrite(step1, HIGH);
         digitalWrite(step2, HIGH);
         digitalWrite(step3, LOW );
         digitalWrite(step4, LOW );   delay( 2 );
         digitalWrite(step1, HIGH);
         digitalWrite(step2, LOW );
         digitalWrite(step3, LOW );
         digitalWrite(step4, HIGH);   delay( 2 );
      }

      if(digitalRead(buton) == HIGH)  //二次馬達反向
      {
            if(control == UPON) control = DOWN;
         else if(control == DOWN) control = UPON;
      }
   }

   ///// 陀螺儀自動控制馬達(按鈕沒壓) /////
   else if(isPress == HIGH)
   {
      tNow = micros();              //現在時間
      tDif = tNow - tPrv;           //時間差異
      tPrv = tNow;                  //前一時間

      val = compass.read_scaled(); //羅盤讀動態值
      dpsX = val.g_x - zroX;    //真正角速度=讀取值-校正值
                                   //由角速度計算角度
      degX = degX + dpsX * (tDif/1000000.0);
                                   //瞬間轉動角差來驅動馬達
      if(degX-degP > 0) motorRun(DOWN,  degX-degP);
      else              motorRun(UPON, -degX+degP);
      Serial.println(degX-degP, 3);//列印瞬間轉動角差
      degP = degX;              //更新上次角度(以利下次相減)
   }
}
```

6. 結果與討論

本研究係以Arduino機電整合創客設計進行學生專題指導，報告中列舉手機藍牙遙控智慧家居開關盒、黎明校歌蜂鳴器演奏、超音波測距汽車電動尾門以及陀螺儀自動控制車燈投射角度裝置...等四項專題成品之完成開發所有細節，相信此教材對於培養學生具備之零件選用、電路規劃、程式設計以及成品製作將有莫大助益，以期讓學生能夠習得藍牙通訊技術、手機積木方塊程式設計技術、繼電器模組控制技術、蜂鳴器震盪發聲技術、音調與節拍演奏控制技術、超音波測距技術、線性電動缸正反轉驅動技術、近遠端極限開關保護技術、陀螺儀角度感測技術以及步進馬達控制技術...等專業技能。

7. 參考文獻

1. 張義和，智慧居家，9-2節藍牙控制繼電器實習，新文京開發出版，新北市，台灣，2017。
2. 陳明熒，Arduino語音互動專題製作與應用，第7章Arduino小家電控制及互動調光器，松崗科技，台北，台灣，2016。
3. 趙英傑，超圖解 Arduino 互動設計入門(第3版)，第14章手機藍牙遙控機器人製作，旗標科技，台北，台灣，2016。
4. 郭恆鳴，專題製作-Arduino+App Inventor2，單元2-App Inventor2的簡介，全華圖書，新北市，台灣，2017。
5. 艾迪諾，Arduino全能微處理機實習-強效解析，4-8節-計時器設計，全華圖書，新北市，台灣，2015。
6. 楊明豐，Arduino最佳入門與應用:打造互動設計輕鬆學(第2版)，第11章聲音控制實例，碁峰資訊，台北，台灣，2015。
7. 周忠信，Arduino初學完全指南，第10章超音波距離感測器實習，碁峰資訊，台北，台灣，2016。
8. 程兆龍，Arduino微控智學創新(第二版)，第九章玩伺服器與步進馬達，新文京開發出版，新北市，台灣，2017。

Electromechanical Integration of Arduino Microcomputer Automation Control Technology

Chi-Shih Yang
Department of Innovative Product Design,
Lee-Ming Institute of Technology

yjs@mail.lit.edu.tw

Abstract

The research combines the sprits of IoT and Maker to utilitize the technologies of mechatronic integration, microcomputer controlling and programming design. The developed functions can be achieved by digital controlling, Bluetooth communication, motor driving and sensor perception.

The electromechanical integration of microcomputer Arduino automation technology enumerates four special research products. It includes: (1)mobile phone Bluetooth remote controlling switch box of the smart home, (2)buzzer performance of Lee-Ming school song, (3)ultrasonic ranging electric car tailgate, (4)Gyro automatic control headlight projection angle device...etc. The above four paradigms basically use the idea of creative design to achieve the electromechanical integration by Arduino single chip microcomputer automation, as well as the technological planning, designing, manufacturing. It is to learn the required professional knowledge and skills to achieve the goal of nurturing professional and technical personnel in the automation industry.

Keywords: Arduino single chip microcomputer, IoT, Maker, Mechatronic integration, Microcomputer controlling, Programming design, Digital controlling, Bluetooth communication, Motor driving, Sensor perception