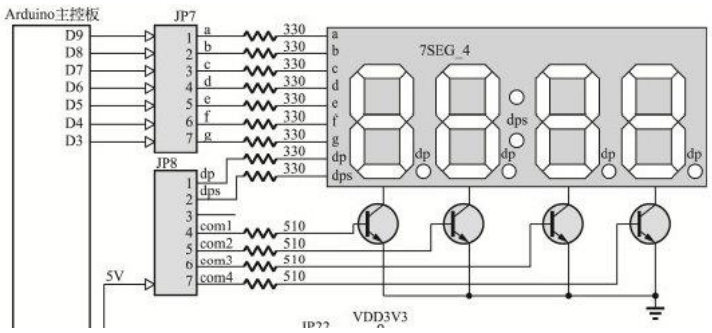
進階實習 1: 透過電腦端的 Serial Monitor 控制閃爍時間

```
 /* 5-4-1 透過電腦端的 Serial Monitor 控制閃爍時間 */
// Pin 10 接一個 LED 至 Arduino 板
// 宣告一個 led 的變數名稱，指定為 13
int led = 13;
// 宣告延遲時間的變數名稱，預設為 500 ms
int delaytime=500;
// 此 setup 程序只有在微控制器按 reset 時執行一次
void setup() {
  // 初始化指定的數位腳位為輸出模式
  pinMode(led, OUTPUT);
  // 設定串列埠的鮑率為 9600 bps
  Serial.begin(9600);
}
// 此 loop 程序會一直重覆執行
void loop() {
  // 判斷串列埠緩衝區有無資料
  if (Serial.available()) {
    delaytime=Serial.parseInt(); // 從串列埠緩衝區中讀取下一個有效的整數資料
    if (delaytime>0) {
      Serial.print(delaytime); // 在 Serial Monitor 中顯示訊息
      Serial.println(" ms");  //
    }
  }
  digitalWrite(led, HIGH);   // LED 滅 (led 輸出高準位，由於 LED 為低態動作的接法，故 led 輸出高準位時 LED 滅)
  delay(delaytime);          // 等待一段時間
  digitalWrite(led, LOW);    // LED 亮 (led 輸出低準位)
  delay(delaytime);          // 等待一段時間
}
```

進階實習 2:

四位元多工顯示器實驗 可透過 Serial Monitor 下顯示數字

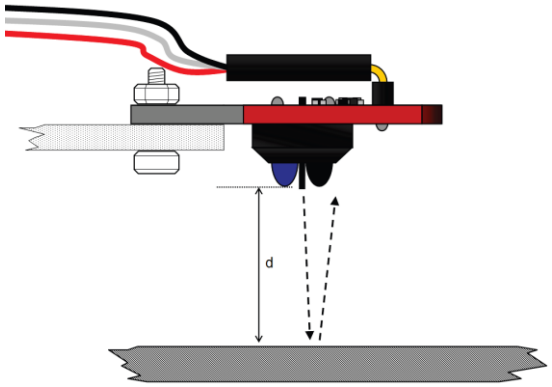


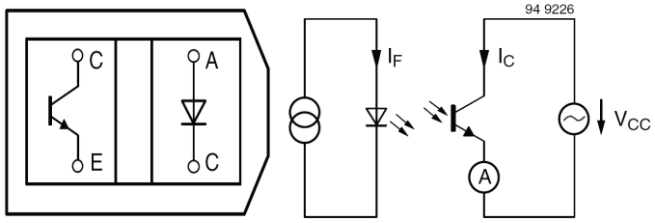

```
/*  四位元多工顯示器實驗 可透過 Serial Monitor 下顯示數字  */
// 七段控制腳陣列，分別對應 a～g 段
const int seg7[] = {11, 10, 9, 8, 7, 6, 5};
// 四位數掃描端控制腳陣列，對應千、百、十、個位數
const int scan[] = { 13, 12, 3, 2 };
// 顯示的數字
int Data[4];
int number = 2013;
// 宣告延遲時間的變數名稱，預設為 5 ms
int delaytime=5;
// 七節顯示器編碼表
char
TAB[]={ 0x3F,0x06,0x5B,0x4F,0x66,0x6D,0x7D,0x27,0x7F,0x67,0x00 };
int ii,jj;      // 迴圈用的變數
int ScanLine=0;
// 此 setup 程序只有在微控制器按 reset 時執行一次
void setup() {
  // 初始化指定的數位腳位為輸出模式
  for (ii=0; ii<7; ii++) {
    pinMode(seg7[ii], OUTPUT);
  }
  for (ii=0; ii<4; ii++) {
    pinMode(scan[ii], OUTPUT);
    digitalWrite(scan[ii], LOW);   // 掃描端電晶體 OFF
  }
  SaveData();
  // 設定串列埠的鮑率為 9600 bps
```

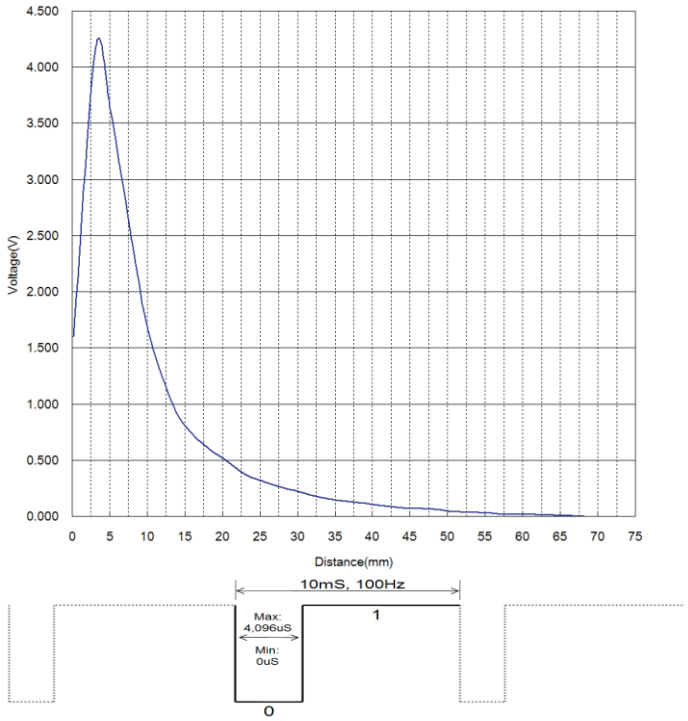
```
  Serial.begin(9600);
}
// 此 loop 程序會一直重覆執行
void loop() {
  // 判斷串列埠緩衝區有無資料
  if (Serial.available()) {
    number=Serial.parseInt(); // 從串列埠緩衝區中讀取下一個有效的整數資料
    Serial.print("number = ");     //
    Serial.println(number); // 在 Serial Monitor 中顯示訊息
    SaveData();
  }
  OutPort(TAB[Data[ii]]);
  digitalWrite(scan[ii], HIGH);   // 指定的掃描端電晶體 ON
  delay(delaytime);
  digitalWrite(scan[ii], LOW);   // 指定的掃描端電晶體 OFF
  ii=(ii+1) % 4;          // 換下一條掃描線, 並限制在 0~3
}
// 將指定值顯示在七段顯示器上，最低位元為 a，依序為 abcdefg
void OutPort(byte dat) {
  for (jj=0; jj<7; jj++) {
    if (dat % 2==1)   // 取出 dat 的最低位元
      digitalWrite(seg7[jj], HIGH); //若為 1 代表該段要亮, 輸出高準位
    else
    digitalWrite(seg7[jj], LOW); //若為 0 代表該段要滅, 輸出低準位
    dat=dat/2;   //除 2，進行下一位元的處理
  }
}
// 將欲顯示的數字 number 存入顯示陣列 Data 中
void SaveData(void) {
  int temp=number;
  Data[3]=temp % 10;      // 個位數
  temp=temp/10;
  Data[2]=temp % 10;
  temp=temp/10;
  Data[1]=temp % 10;
  Data[0]=temp/10;          // 千位數
}
```

進階實習 3: TCRT5000 紅外反射式光電傳感器/反射型光電開關/尋跡小車專用

TCRT5000反射式紅外線偵測元件，來偵測從平面反射回來的類比訊號，並可透過A2P Converter轉換成脈波寬度訊號輸出，可測量約0~60mm之距離，使用者可依據不同環境進行調整,可辨別灰階型顏色表面。







A2P Converter脈波輸出圖

數位輸出是輸出固定電壓值，只有0與1的變化，每10 ms(ms = 1/1000 sec)會根據電壓值大小，改變輸出電壓維持在0(低電位)的時間，如上圖。根據接收到的電壓值不同，輸出波形在低電位的時間，會從最小0 us，到最大4096 us(us = 1/1000 ms)，解析度為2uS。

以5V為例，每增加約2.4 mV就會將在低電位的時間增加2 us，相反的，每減少約2.4 mV就會減少輸出在低電位的時間2 us。例如給予A2P Converter 類比電壓3V的輸入值，低電位時間(脈波寬度)會約在2458uS。計算方式如下:

(3(輸入類比電壓(V)) / 5(供電電壓(V))) x 4096 (us) = 2457.6 (us) ≒ 2458 (us)

公式: 距離 y=27.66x(adc) E –0.586

程式碼:

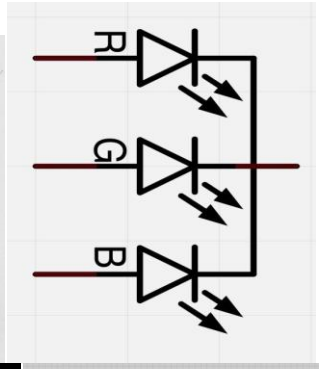
```
/* 紅外線測距模組 TCRT5000 的測距程式 */
#include <math.h>          // 引入數學函式庫
void setup() {
  Serial.begin(9600);
}
void loop() {
  float cm;
  int adc = analogRead(A0);        //讀入指定類比腳位的 ADC 值
  if (adc>0) {
    cm = 27.66 * pow(adc, -0.586);// 讀值與距離的曲線轉換公式
    Serial.print(adc);             // 顯示測得的 ADC 值
    Serial.print("\t");
    Serial.print(cm);              // 顯示測得的距離值
    Serial.println(" cm");
  }
  delay(200);                      // 延遲 0.2 秒
}
```

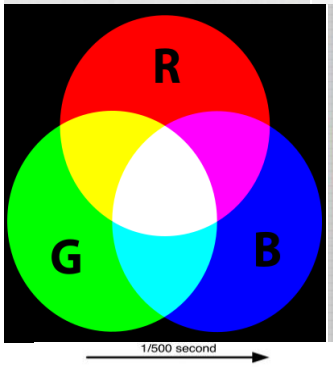
進階實習 4:

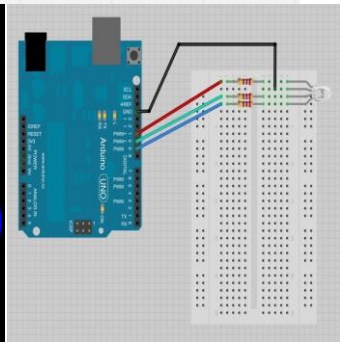
```
//使用 IRremote 程式庫解析紅外線遙控接收值
#include <IRremote.h>
int RECV_PIN = 11;  //宣告紅外線接收 pin 第 11 腳
IRrecv  irrecv(RECV_PIN) ;  //宣告紅外線接收物件 irrecv
decode_results  results ;  //宣告紅外線接收值的變數 results

void setup() {
  Serial.begin(9600); // 設定串列埠的鮑率為 9600 bps
  irrecv.enableIRIn();//啟動紅外線功能
}
void loop() {
  //解析紅外線接收值,若 decode()傳回 true,代表有接收到新資料
  if (irrecv.decode(&results)) {
    //儲存接收到的變數
    Serial.println(results.value, HEX);
    //解讀解析後的數值,並以十六進位格式輸出
    irrecv.resume();
    //準備接收下一筆資料
  }
}
```

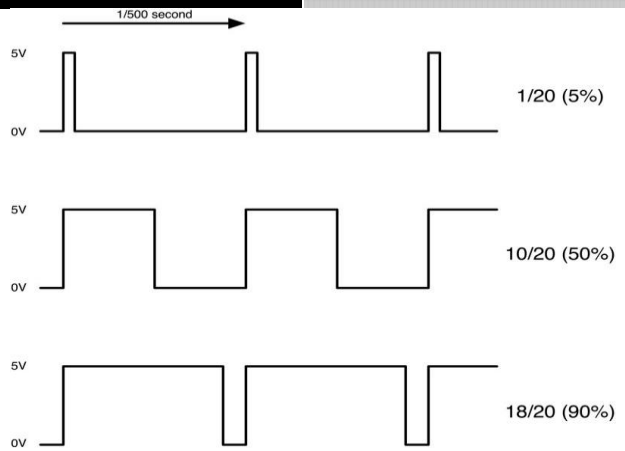
紅外線接收器接腳	Arduino 接腳
GND (-)	接到 GND
Vcc (+ 或 V+)	接到 +5V 電源
Vout (或 OUT)	接到 pin11。你可以接到其它 Digital pin，但程式要配合修改

進階實習 5: RGB LEDs 控制實習










```
/*
Adafruit Arduino - Lesson 3. RGB LED
*/
int redPin = 11;
int greenPin = 10;
int bluePin = 9;
 //uncomment this line if using a Common Anode LED
//#define COMMON_ANODE

void setup()
{
  pinMode(redPin, OUTPUT);
  pinMode(greenPin, OUTPUT);
  pinMode(bluePin, OUTPUT);
}

void loop()
{
  setColor(255, 0, 0);  // red
  delay(1000);
  setColor(0, 255, 0);  // green
  delay(1000);
  setColor(0, 0, 255);  // blue
  delay(1000);
  setColor(255, 255, 0);  // yellow
  delay(1000);
  setColor(80, 0, 80);  // purple
  delay(1000);
  setColor(0, 255, 255);  // aqua
  delay(1000);
}

void setColor(int red, int green, int blue)
{
  #ifdef COMMON_ANODE
    red = 255 - red;
    green = 255 - green;
    blue = 255 - blue;
  #endif
  analogWrite(redPin, red);
  analogWrite(greenPin, green);
  analogWrite(bluePin, blue);
}
```

進階實習 6: 蜂鳴器實習

無源(8~16 歐姆)

有源(幾百歐姆)

有源蜂鳴器直接接上額定電源(新的蜂鳴器在標籤上都有註明)就可以連續發聲，而無源蜂鳴器則和電磁揚聲器一樣，需要接在音頻輸出電路中才能發聲。程式 1

```
1.  int buzzer= 10 ; //設置控制蜂鳴器的數字 IO 腳
2.  void   setup ( )
3.  {
4.    pinMode ( buzzer, OUTPUT ) ;
    //設置數字 IO 腳模式，OUTPUT 為輸出
5.  }
6.  void   loop ( )
7.  {
8.    unsigned   char i,j;   //定義變量
9.    while ( 1 )
10.   {
11.     for ( i= 0 ;i< 80 ;i++ ) //輸出一個頻率的聲音
12.     {
13.       digitalWrite ( buzzer, HIGH ) ; //發聲音
14.       delay ( 1 ) ; //延時 1ms
15.       digitalWrite ( buzzer, LOW ) ; //不發聲音
16.       delay ( 1 ) ; //延時 ms
17.     }
18.     for ( i= 0 ;i< 100 ;i++ ) //輸出另一個頻率的聲音
19.     {
20.       digitalWrite ( buzzer, HIGH ) ; //發聲音
21.       delay ( 2 ) ; //延時 2ms
22.       digitalWrite ( buzzer, LOW ) ; //不發聲音
23.       delay ( 2 ) ; //延時 2ms
24.     }
25.   }
26. }
```

程式 2:使用 tone()函式

```
const int speaker=10;
void setup()
{}
void loop()
{
  for(int i=0;i<10;i++)
  {
    tone(speaker,1000); //輸出 1000hz 音高
    delay(50);
    tone(speaker,500);   //輸出 500hz 音高
    delay(50);
  }
  noTone(speaker);//靜音 2 秒
  delay(2000);
}
```

// tone()用法

- tone(pin, frequency)
- tone(pin, frequency, duration)

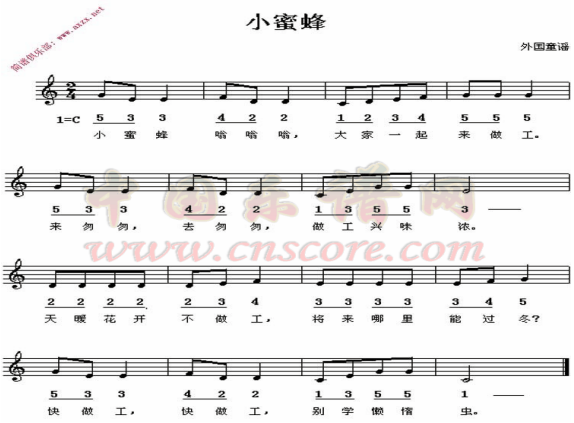
1. pin 是連接揚聲器的引腳
2. frequency 是輸出頻率，頻率越低，音頻就越低。
3. duration 音頻輸出持續時間

程式 3: 使用 tone()函式播放音符

```
const int speaker=10;
const int
toneTable[8]={523,587,659,694,784,880,988,104
6};
void setup()
{}
void loop()
{
  for(int i=0;i<8;i++)
  {
    tone(speaker,toneTable[i]);
    delay(500);
```

```
  }
  noTone(speaker);
}
```

程式 4: 使用 tone()函式播放小蜜蜂




```
const int speaker=10;
char toneName[]="CDEFGAB";
unsigned int
frequency[7]={523,587,659,694,784,880,988};
char
beeTone[]="GEEFDDCDEFGGGGEEFDDCEGGED
DDDDEFEEEEEFGGEEFDDCEGGC"; //音符頻率表
byte
beeBeat[]={1,1,2,1,1,2,1,1,1,1,1,1,2,1,1,2,1,1,2,1,1,
1,1,4,


1,1,1,1,1,1,2,1,1,1,1,1,1,2,1,1,2,1,1,2,1,1,1,1,4};  //
音符表
const int beeLen=sizeof(beeTone);  //小蜜蜂音
符總數
unsigned long tempo=180; //每分鐘 180 拍
int i,j;
void setup()
{
}
void loop()
{
  for(i=0;i<beeLen;i++)
      playTone(beeTone[i],beeBeat[i]); //播放小蜜
蜂
  delay(3000); //每隔 3 秒重複播放
}
```

```
void playTone(char toneNo,byte beatNo) //播放
函式


{
  unsigned long
  duration=beatNo*60000/tempo;
  //計算每拍時間(毫秒)
  for(j=0;j<7;j++)
  {
    if(toneNo==toneName[j])  //查音符表
    {
      tone(speaker,frequency[j]); //播放音符
      delay(duration); //此音符節拍
      noTone(speaker); //靜音
    }
  }
}
```

進階實習 7:

4x4 鍵盤實習讀取鍵盤輸入值

Arduino	4x4 Keypad
D2	0
D3	1
D4	2
D5	3
D6	4
D7	5
D8	6
D9	7

安裝 Arduino Keypad 鍵盤庫

- 下載 Arduino Keypad 鍵盤庫
- 將下載了的文件（keypad.zip）解壓至 Arduino 軟件的 libraries 文件夾
- 首先必須安裝 Arduino Keypad 鍵盤庫（Keypad library）,Arduino Keypad 鍵盤庫可以從 Arduino Playground 下載。Arduino Keypad 鍵盤庫讓你讀取矩陣式鍵盤而不用編寫複雜的代碼，此鍵盤庫可以讀取 3x4， 4x4 以及各種矩陣結構的鍵盤

程式 1

```
#include   <Keypad.h>
const byte ROWS = 4; // Four rows
const byte COLS = 4; // Four columns
//Define the keymap

char keys[ROWS][COLS] = {
{'1','2','3','A'},
{'4','5','6','B'} ,
{'7','8','9','C'},
{'*','0','#','D'}
};
//// Connect keypad ROW0, ROW1, ROW2 and ROW3 to these Arduino pins.

byte rowPins[ROWS] = {6,7,8,9};
// Connect keypad COL0, COL1, COL2 and COL3 to these Arduino pins.

byte colPins[COLS] = { 2,3,4,5 }; //connect to column pinouts
// Create the Keypad

Keypad keypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS );

void setup(){ Serial.begin(9600); }
  void loop()
  { char key = keypad.getKey();
 if (key != NO_KEY){ Serial.println(key); } }
```

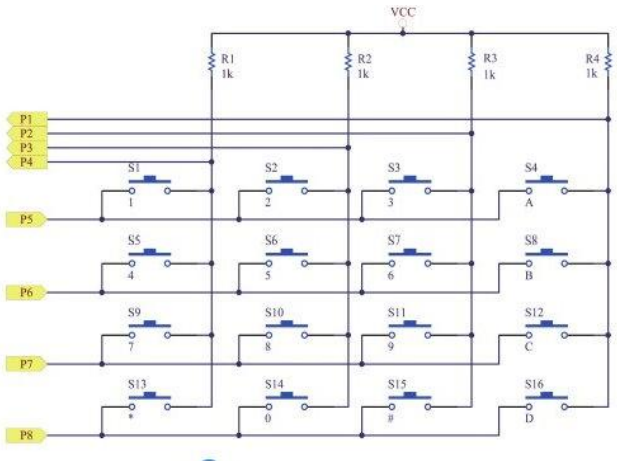
程式 2

```
/* 4x4 矩陣鍵盤 */
const int numRows = 4; //定義 4 行
const int numCols = 4; //定義 4 列
const int debounceTime = 20; //去抖動時間長度
const char keymap[numRows][numCols]= {
{ '1','2','3','+' },
{'4','5','6','-' },
{'7','8','9','X' },
{'*','0','#','/' }
};
//鍵值，可以按需要更改
const int rowPins[numRows] = {6,7,8,9}; //設置硬體對應的引腳
const int colPins[numCols] = {2,3,4,5 };
//初始化功能
void setup(){
Serial.begin(9600);
for(int row = 0; row < numRows; row++){
    pinMode(rowPins[row],INPUT);
    digitalWrite(rowPins[row],HIGH);}
for(int column = 0; column < numCols; column++){
      pinMode(colPins[column],OUTPUT);
      digitalWrite(colPins[column],HIGH);
 }
}
//主迴圈
void loop() {
      // 添加其他的程式，迴圈運行
      char key = getkey();
      if(key !=0){
        Serial.print("Got key ");   //串口列印鍵值
        Serial.println(key);
      }
  }
//讀取鍵值程式
char getkey(){
      char key = 0;
      for(int column = 0;column < numCols; column++){
        digitalWrite(colPins[column],LOW);
        for(int row = 0 ;row < numRows; row++){
          if(digitalRead(rowPins[row]) == LOW){
          //是否有按鍵按下
       delay(debounceTime);
       while(digitalRead(rowPins[row])== LOW) ;
           //等待按鍵釋放
             key = keymap[row][column];
          }
```

```
    }
    digitalWrite(colPins[column],HIGH);
    //De-active the current column
  }
  return key;
}
```

進階實習 8: 4x4 鍵盤電子琴實習

```
#include <Keypad.h>
const byte ROWS = 4 ; // Four rows
const byte COLS = 4 ; // Four columns

//Define the keymap
char keys [ ROWS ] [ COLS ] = {
{ '1' , '2' , '3' , 'A' } ,
{ '4' , '5' , '6' , 'B' } ,
{ '7' , '8' , '9' , 'C' } ,
{ '*' , '0' , '#' , 'D' }
} ;

//// Connect keypad ROW0, ROW1,
   ROW2 and ROW3 to these Arduino pins.
byte rowPins [ ROWS ] = { 6 , 7 , 8 , 9 } ;

// Connect keypad COL0, COL1, COL2
   and COL3 to these Arduino pins.
byte colPins [ COLS ] = { 2 , 3 , 4 , 5 } ; //c
   onnect to column pinouts

// Create the Keypad 物件
Keypad
   keypad = Keypad ( makeKeymap ( keys )
   , rowPins , colPins , ROWS , COLS ) ;

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

void loop ( ) {
char key = keypad. getKey ( ) ;
if ( key != NO_KEY ) {
delay ( 50 ) ; //act as debounce
beep ( ) ;
Serial . println ( key ) ;
}
}

#define   SPEAKER_PIN   10
void beep ( ) {
tone ( SPEAKER_PIN , 2000 , 90 ) ;
delay ( 20 ) ;
noTone ( SPEAKER_PIN ) ;
}
```



1. 鍵盤掃描取值原理

(1) 行列式鍵盤一般分為掃描端與資料端，如圖所示，其中 P5～P8 為掃描信號輸入端，P1～P4 為資料輸出端，為使輸出準位明確，P1～P4 需接提昇電阻。

(2) 掃描端會由控制電路依序送出掃描訊號至 P5～P8，分別為 0111、1011、1101、1110 不斷的重複產生。

(3) 平時按鍵都沒有按下時，P1~P4 所讀取之數值為 1111，當掃描端 P5 ～ P8 為 0111 時，表示此時要偵測的按鍵為 S1、S2、S3、S4，按 S5 ～ S16 是不會有作用的；如果此時 S1 按鍵被按下，則 P1 ～ P4 所偵測之訊號將變為 1110；S2 按鍵被按下，則 P1～P4 所偵測之訊號將變為 1101；S3 按鍵被按下，則 P1～P4 所偵測之訊號將變為 1011；

S4 按鍵被按下，則 P1 ~ P4 所偵測之訊號將變為 0111。

(4) 同理，當掃描端 P5～P8 為 1011 時，表示此時要偵測的按鍵為 S5、S6、S7、S8，按 S1 ~ S4 以及 S9 ~ S16 一樣不會有作用。依此方法不斷依序掃描，即可以判斷此 16 個按鍵是否有按下。

進階實習 9 : RGB LED 控制實習

RGB LED 測試程式 1

```
/* 三色 LED 的顯示測試 */
const byte led[]={ 6,5,3 };   // 宣告紅綠藍三個 LED 的腳位
const byte led_RGB[6][3] = {   // 宣告不同顏色的 RGB 數值
  {0,204,255},          // 0 天藍色
  {255,255,255},   // 1 白色
  {0,255,0},          // 2 綠色
  {255,127,0},      // 3 橘色
  {255,0,0},          // 4 红色
  {153,50,205} };   // 5 紫色
int ii;                  // 計數值變數
int color=0;        // 顏色索引值

void setup() {
  for (ii=0; ii<2; ii++)
    pinMode (led[ii], OUTPUT);    // 指定 LED 為輸出埠
}

void loop() {
  for (ii=0; ii<3; ii++)   // 依序輸出指定顏色索引值的 RGB 值
      analogWrite(led[ii],led_RGB[color][ii]);   // 輸出 PWM
  delay(1000);             // 延遲 1 秒
  color=(color+1) % 6; //顏色索引值加 1,並取出除以 6 的餘數
}
```

// RGB LED - 自動顏色循環變化 程式 2

```
// Matthew L Beckler
// matthew at mbeckler dot org

int redPin = 6;              // 紅色 LED 腳位
int greenPin = 5;   // 綠色 LED 腳位
int bluePin = 10;   // 藍色 LED 腳位

int redVal=0;
int greenVal=0;
int blueVal=0;

void setup()
{
  pinMode(redPin, OUTPUT);
  pinMode(greenPin, OUTPUT);
  pinMode(bluePin, OUTPUT);
}

void loop()
{
  // start out at black (all off)
  color_morph(redVal,    1);      // transition to red
  color_morph(greenVal, 1);      // transition to yellow
  color_morph(redVal,    0);      // transition to green
  color_morph(blueVal,   1);      // transition to aqua
  color_morph(redVal,    1);      // transition to white
  color_morph(greenVal, 0);      // transition to violet
  color_morph(redVal,    0);      // transition to blue
  color_morph(blueVal,   0);      // transition to black (all off)
}

// This function updates the LED outputs.
void update()
```

```
{
  analogWrite(redPin, redVal);
  analogWrite(greenPin, greenVal);
  analogWrite(bluePin, blueVal);
}

// This function updates one of the color variables
// either getting brighter or getting dimmer.
// It also updates the outputs and delays for 10 milliseconds.
void color_morph(int &value, int get_brighter)
{
  for (int i = 0; i < 255; i++)
  {
    if (get_brighter)
      value++;
    else
      value--;

    update();
    delay(10);
  }
}
```

進階實習 11：

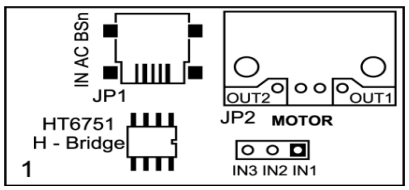
```
/* 紅外線遙控器 DC 直流馬達控制
紅外線 IRremote 模組取自 Ken Shirriff http://arcfn.com*/
#include <IRremote.h>
#define IN1          10   // 馬達控制信號輸入端
#define IN2          11   // 馬達控制信號輸入端
#define RECV_PIN 2     // IR
byte   speedvalue;
unsigned   long code;
int on = 0;
char dir = IN1;                // forward
unsigned   long   last = millis();
IRrecv irecv(RECV_PIN);     // 繼承 IRrecv 物件來接收紅外線訊號
decode_results   results;     // 解碼結果將放在 results 變數裏

void setup()
{
  pinMode(IN1, OUTPUT);  //  IN1  設定為 OUTPUT
  pinMode(IN2, OUTPUT);  //  IN2  設定為 OUTPUT
  Serial.begin(9600);   // 啟用串列埠
  irecv.enableIRIn();   // 啟動紅外線解碼
}
void loop() {
  if (irecv.decode(&results)) {
    // 若解碼成功，收到一組紅外線訊號
    // IR 接收訊號的間隔若大於 0.25 秒，
    // 才會對馬達進行切換，並顯示接收到的紅外線訊號
    if (millis() - last > 250) {
      code=results.value;
      if (code==0xFF6897) {        // off motor '0'
          speedvalue=0;
          on=0;
      }
      else if (code==0xFF30CF) {      //          on motor '1'
          speedvalue=255;
          on=1;
      }
      else if (on==1) {
          if (code==0xFFA857) {          //     + motor
              speedvalue+=20;
              if                (speedvalue>255) speedvalue=255;
          }
          else  if  (code==0xFFE01F)  {      // - motor
              speedvalue-=20;
              if                (speedvalue<50) speedvalue=50;
```

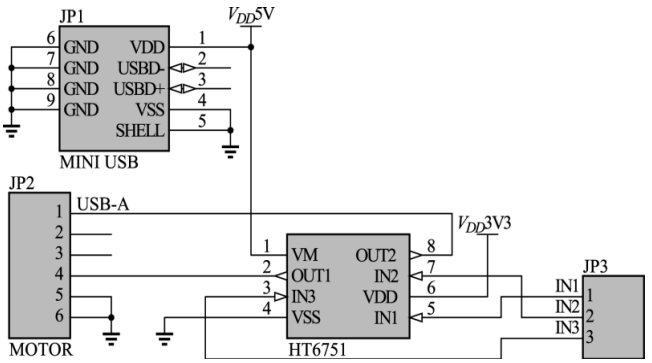
```
            }
            else if (code==0xFF906F) {      // 正反轉切換
                digitalWrite(IN1, LOW);
                digitalWrite(IN2, LOW);
                dir = (dir==IN1) ? IN2 : IN1;
            }
        }
        Serial.print("code=");
        Serial.print(code);
        Serial.print("\t");
        Serial.println(speedvalue);
        analogWrite(dir, speedvalue);
    }
    last = millis();
    irrecv.resume();      // 繼續接收下一組紅外線訊號
  }
}
```

進階實習 12 :

//1. 直流馬達驅動電路

由 MINI USB(JP1)提供+5V，經 H 型橋式驅動 IC（HT6751）來驅動連接 USB-A（JP2）的直流馬達（風扇）。





TH6751 A

IN 1	IN 2	IN 3	Function	MOS On	MOS Off
0	1	1	Motor1 forwar d	P1/N2	P2/N1
1	0	1	Motor1 revers e	P2/N1	P1/N2
0	0	1	Motor1 brake	N1/N2	P1/P2
1	1	1	Standby mode	¾	P1/P2/N1/N2
0	1	0	Motor2 forwar d	P2/(N3)	N2/(P3)
1	0	0	Motor2 revers e	N2/(P3)	P2/(N3)
0	0	0	Motor2 brake	N2/(N3)	P2/(P3)

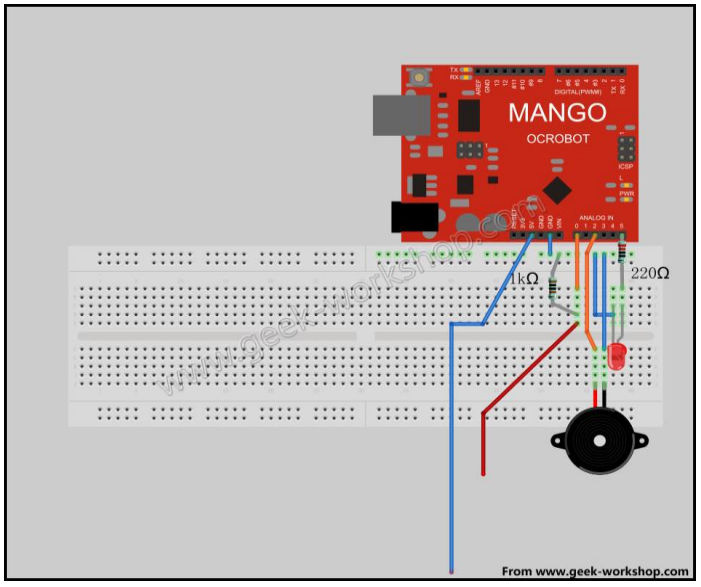
電動車行進速度與方向控制

```
const int in1Pin=2;
const int in2Pin=3;
const int in3Pin=4;
const int in4Pin=5;
const int enA=9;
const int enB=10;
void setup()
{
  Serial.begin(9600);
  Serial.println("press '0'~'9' : setup speed");
  Serial.println("press 'S' : stop");
  Serial.println("press 'F' :   front");
  Serial.println("press 'B' :   back");
  Serial.println("press 'R' : turn right");
  Serial.println("press 'L' : turn left");
  pinMode(in1Pin,OUTPUT);
  pinMode(in2Pin,OUTPUT);
  pinMode(in3Pin,OUTPUT);
  pinMode(in4Pin,OUTPUT);
  analogWrite(enA,50);
  analogWrite(enB,50);
}
void loop()
{
  if(Serial.available())
```

```
{
  char key=Serial.read();
  Serial.print("key=");
  Serial.println(key);
  if(key>='0' && key<='9')
  {
    int speed=map(key,'0','9',50,250);
    analogWrite(enA,speed);
    analogWrite(enB,speed);
  }
  else if(key=='S'||key=='s')
  {
    digitalWrite(in1Pin,LOW);
    digitalWrite(in2Pin,LOW);
    digitalWrite(in3Pin,LOW);
    digitalWrite(in4Pin,LOW);
  }
  else if(key=='F'||key=='f')
  {
    digitalWrite(in1Pin,LOW);
    digitalWrite(in2Pin,HIGH);
    digitalWrite(in3Pin,HIGH);
    digitalWrite(in4Pin,LOW);
  }
  else if(key=='B'||key=='b')
  {
    digitalWrite(in1Pin,HIGH);
    digitalWrite(in2Pin,LOW);
    digitalWrite(in3Pin,LOW);
    digitalWrite(in4Pin,HIGH);
  }
  else if(key=='R'||key=='r')
  {
    digitalWrite(in1Pin,LOW);
    digitalWrite(in2Pin,HIGH);
    digitalWrite(in3Pin,LOW);
    digitalWrite(in4Pin,LOW);
  }
  else if(key=='L'||key=='l')
  {
    digitalWrite(in1Pin,LOW);
    digitalWrite(in2Pin,LOW);
    digitalWrite(in3Pin,HIGH);
    digitalWrite(in4Pin,LOW);
  }
 }
}
```

進階實習 13：

製作訊線水位報警器模型控制實習

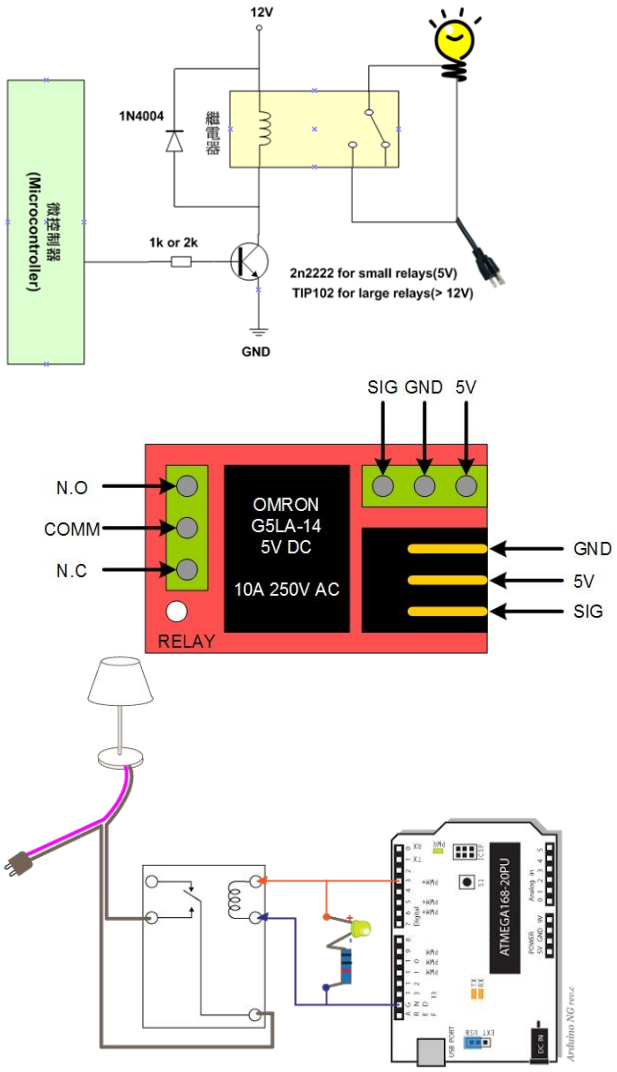


程式

```
/*水位報警器，到達設定水位，進行指示燈和聲音報警*/
 void setup()
{
  pinMode(A5,OUTPUT);  //讓類比口 A5 作為數位口輸出
}
 void loop()
{
 int n=analogRead(A0);
 if (n>=1)
 {
  digitalWrite(A5, HIGH);
  pinMode(A2,OUTPUT);   //蜂鳴器頻響  0.5 秒
  tone(A2,800);
  delay(500);
```

```
  pinMode(A2,INPUT);
  digitalWrite(A5, LOW);
  delay(500);
 }
}
```

進階實習 14 :如何透過繼電器控制外部電路



進階實習 15 :Arduino 電腦開關 LED

程式 1:

```
void setup()
{
  Serial.begin(9600);
  Serial.print("Ready");
  pinMode(3,OUTPUT);
}
void loop()
{
  if ( Serial.available())
{
  char ch = Serial.read(); //接收由電腦端傳來的字元
  Serial.print(ch);
  switch(ch) {
   case 't':
   digitalWrite(3,HIGH); //將 PIN3 輸出高電位
   delay(1000);
   break;
  case 'r':
   digitalWrite(3,LOW); //將 PIN3 輸出高電位
   delay(1000);
   break;
  }
  }
  }
```

程式 2:

```
/* 紅外線遙控 + 繼電器控制 12V 風扇 */
#include <IRremote.h>        // 引用 IRRemote 函式庫
const int irReceiverPin = 4;      // 紅外線接收器 OUTPUT 訊號接在 pin 2
int motor = 2;
IRrecv irrecv(irReceiverPin);     // 定義 IRrecv 物件來接收紅外線訊號
decode_results results;     // 解碼結果將放在 decode_results 結構的 result 變數裏
void setup()
{
  Serial.begin(9600);         //開啟 Serial port, 通訊速率為 9600 bps
  irrecv.enableIRIn();        //啟動紅外線解碼
  pinMode(motor, OUTPUT);
}
void loop()
{
  if (irrecv.decode(&results)){ //解碼成功,收到一組紅外線訊號
          Serial.print("Protocol: "); //印到 Serial
```

```
port
        // 判斷紅外線協定種類
            switch(results.decode_type) {
        case NEC:
          Serial.print("NEC");
          break;
        case SONY:
          Serial.print("SONY");
          break;
        case RC5:
          Serial.print("RC5");
          break;
        case RC6:
          Serial.print("RC6");
          break;
        default:
          Serial.print("Unknown encoding");
    }
    Serial.print("    irCode: ");
    Serial.print(results.value);      // 紅外線編碼
      if (results.value == 16724175) {   //遙控器上的 0
        digitalWrite(motor, LOW);
    }
    if (results.value == 16738455) { //遙控器上的 1
         digitalWrite(motor, HIGH);
    }
    Serial.print(",    bits: ");
    Serial.println(results.bits);   // 紅外線編碼位元數
    delay(500);
  irrecv.resume();              // 繼續收下一組紅外線訊號
  }
}
```

程式 3:

```
/*   用繼電器控制 12V 風扇(使用紅外線遙控) */
const int irReceiver = 2;           // 紅外線接收器
const int relayPin = 13;            // 繼電器(Relay)
int relayState = 0;                 // 繼電器狀態
void setup()
{
  Serial.begin(9600);   //開啟 Serial port,通訊速率為 9600 bps
  pinMode(irReceiver, INPUT); // 把 irReceiver 接腳設置為 INPUT
  pinMode(relayPin, OUTPUT); // 把 relayPin 設置成 OUTPUT
}
void switchRelay()
{
  if (relayState == 0)
    relayState = 1; // 把繼電器狀態改為 ON
  else
    relayState = 0; // 把繼電器狀態改為 OFF
        digitalWrite(relayPin, relayState);   // 讓繼電器作動, 切換開關
  Serial.print("Relay status: "); // 把繼電器的狀態印到 Serial Port
  Serial.println(relayState);
}
void loop()
{
  int irStatus; // 讀取 irReceiver 的狀態
  irStatus = digitalRead(irReceiver);
  // 檢查 irReceiver 是否有收到紅外線訊號
  // 有的話,ir_status 會是 0 (因為 Receiver 會把訊號反向, 所以 0 代表有收到訊號)
  if (irStatus == 0)   {
    switchRelay(); // 切換繼電器開關
    delay(500); // 稍候 0.5 秒,以免風扇切換太頻繁
  }
}
```

實習 16: App 讓 Arduino 與 Android 建立藍芽通訊

下載 Blueterm App 下載 BTCOM App

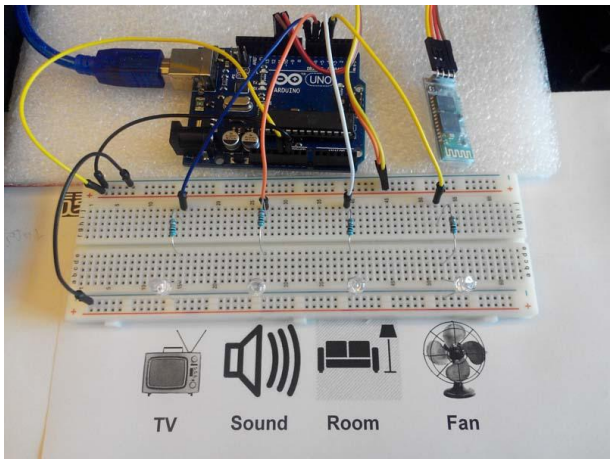
Arduino 與藍牙(hc-05)接線

Arduino	藍芽模組	備註
3.3V	VCC	注意電源不可接錯
GND	GND	注意電源不可接錯
RXD	TXD	
TXD	RXD	

程式一: 藍芽手機遙控 led 亮滅

```
// BluetermDemo 先燒錄程式再接上藍牙
void setup() {
  // 確定你的藍芽模組 baud rate 設定是 9600
  Serial.begin(9600);
  pinMode(13, OUTPUT);
}
void loop() {
  if(Serial.available()){
    // 讀出第 1 個字元
    unsigned char charreceived = Serial.read();
        switch(charreceived){
      case '1':
        digitalWrite(13, HIGH);
        Serial.println("   Arduino Led On");
        break;
      case '0':
        digitalWrite(13, LOW);
        Serial.println("   Arduino Led Off");
        break;
      default:
        break;
    }
   //清掉後續的字元&#65292;我們只要第 1 個字元
    Serial.flush();
  }
  delay(10);
}
```

程式二: 用手機 App 遙控家電開關




```
//使用 BTCOM  App
#include <SoftwareSerial.h>
#define MAX_BTCMDLEN 128
#define BTSW_TV   7  // pin7 當 BlueTooth Switch for TV
#define BTSW_Sound  6  // pin6 當 BlueTooth Switch for Sound
#define BTSW_Room   5  // pin5 當 BlueTooth Switch for Room
#define BTSW_Fan    4  // pin4 當 BlueTooth Switch for Fan
// 建立一個軟體模擬的序列 port
SoftwareSerial BTSerial(10,11); //
```

```
HC-06/TX-->PIN10(Arduino/RX),
HC-06/RX-->PIN11(Arduino/TX);
byte btCmdBuff[MAX_BTCMDLEN]; // up to 128 bytes BT command received from the Android system
int btCmdLen = 0; // received BT command length

void setup() {
  //Serial.begin(9600); // serial port for debugging
   BTSerial.begin(9600); // 建立藍芽軟體串列埠
   pinMode(BTSW_TV,OUTPUT);    // 設定電視開關
   pinMode(BTSW_Sound,OUTPUT); // 設定音響開關
   pinMode(BTSW_Room,OUTPUT);  // 設定房間開關
   pinMode(BTSW_Fan,OUTPUT);   // 設定風扇開關
   digitalWrite(BTSW_TV,LOW);
   digitalWrite(BTSW_Sound,LOW);
   digitalWrite(BTSW_Room,LOW);
   digitalWrite(BTSW_Fan,LOW);
}

void loop() {
   listenBTCmd();
   //showBTCmd(); // for debugging
   executeBTCmd();
}

void executeBTCmd() {
   char cmd[MAX_BTCMDLEN];
   if ( btCmdLen>0 ) {
      sprintf(cmd,"%s",btCmdBuff);
      //Serial.println(cmd); 返回零表示二字串相同

      if ( strcmp("tv,on",cmd)==0 ) {
         digitalWrite(BTSW_TV,HIGH);
      }
      if ( strcmp("tv,off",cmd)==0 ) {
         digitalWrite(BTSW_TV,LOW);
      }
      if ( strcmp("sound,on",cmd)==0 ) {
         digitalWrite(BTSW_Sound,HIGH);
      }
      if ( strcmp("sound,off",cmd)==0 ) {
         digitalWrite(BTSW_Sound,LOW);
      }
      if ( strcmp("room,on",cmd)==0 ) {
         digitalWrite(BTSW_Room,HIGH);
      }
      if ( strcmp("room,off",cmd)==0 ) {
         digitalWrite(BTSW_Room,LOW);
      }
      if ( strcmp("fan,on",cmd)==0 ) {
         digitalWrite(BTSW_Fan,HIGH);
      }
      if ( strcmp("fan,off",cmd)==0 ) {
         digitalWrite(BTSW_Fan,LOW);
      }
   }
}

void listenBTCmd() {
   char tmp;
   btCmdLen = 0;
   memset(btCmdBuff,0,MAX_BTCMDLEN);
   while( BTSerial.available()>0 ) {
      if( (tmp=BTSerial.read())=='O' ){
         btCmdLen = 0;
      }
      btCmdBuff[(btCmdLen++)%MAX_BTCMDLEN] = tmp;
   }
}

void showBTCmd() {
   char cmd[MAX_BTCMDLEN];
```

```
    if ( btCmdLen>0 ) {
        sprintf(cmd,"%s",btCmdBuff);
        Serial.println(cmd);
    }
}
```

程式三:

```
// 用 Android 手機藍牙遙控機器人
#include <SoftwareSerial.h>      // 引用「軟體序列埠」程式庫
SoftwareSerial BT(3, 2);      // 設定軟體序列埠(接收腳, 傳送腳)

char command;                    // 接收序列埠值的變數
const byte EA = 6;               // 馬達 A 的致能接腳
const byte IA = 7;               // 馬達 A 的正反轉接腳
const byte EB = 5;               // 馬達 B 的致能接腳
const byte IB = 4;               // 馬達 B 的正反轉接腳

// 設定 PWM 輸出值
const byte speed = 130;

void stop() {                    // 馬達停止
  analogWrite(EA, 0);            // 馬達 A 的 PWM 輸出
  analogWrite(EB, 0);            // 馬達 B 的 PWM 輸出
}
void forward() {                 // 馬達轉向：前進
  analogWrite(EA, speed);        // 馬達 A 的 PWM 輸出
  digitalWrite(IA, HIGH);
  analogWrite(EB, speed);        // 馬達 B 的 PWM 輸出
  digitalWrite(IB, HIGH);
}
void backward() {                // 馬達轉向：後退
  analogWrite(EA, speed);        // 馬達 A 的 PWM 輸出
  digitalWrite(IA, LOW);
  analogWrite(EB, speed);        // 馬達 B 的 PWM 輸出
  digitalWrite(IB, LOW);
}
void turnLeft() {                // 馬達轉向：左轉
  analogWrite(EA, speed);        // 馬達 A 的 PWM 輸出
  digitalWrite(IA, LOW);         // 馬達 A 反轉
  analogWrite(EB, speed);        // 馬達 B 的 PWM 輸出
  digitalWrite(IB, HIGH);
}
void turnRight() {               // 馬達轉向：右轉
  analogWrite(EA, speed);        // 馬達 A 的 PWM 輸出
  digitalWrite(IA, HIGH);
  analogWrite(EB, speed);        // 馬達 B 的 PWM 輸出
  digitalWrite(IB, LOW);         // 馬達 B 反轉
}

void setup() {
```

```
  BT.begin(9600);                  // 啟動軟體序列埠

  pinMode(IA, OUTPUT);             // 馬達 A 的致能腳位
  pinMode(IB, OUTPUT);             // 馬達 B 的致能腳位
  stop();                          // 先停止馬達
}

void loop() {
  if (BT.available() > 0) {
    command = BT.read();

    switch (command) {
    case 'w' :       // 接收到 'w'，前進
      forward();
      break;
    case 'x' :       // 接收到 'x'，後退
      backward();
      break;
    case 'a' :       // 接收到 'a'，左轉
      turnLeft();
      break;
    case 'd' :       // 接收到 'd'，右轉
      turnRight();
      break;
    case 's' :       // 接收到 's'，停止馬達
      stop();
      break;
    }
  }
}

//需要安裝 BTRobotControl.apk
```

參考資料

1.arduino 官方網站 http://arduino.cc/

2.最簡單的互動設計 Arduino 一試就上手 碁峰 孫駿榮、吳明展、盧聰勇

3.超圖解 Arduino 互動設計入門(第二版) 旗標 趙英傑

4. Arduino 互動設計專題與實戰 碁峰 柯博文

5. Arduino 最佳入門與應用 碁峰 楊明豐

6. Arduino 自造指南 碁峰 原文作者 :John Boxall 譯者：曾繁勛,陳麒元,趙涵捷