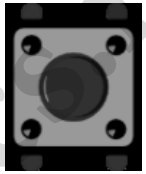
Chap 4
Arduino的I/O

使用的 I/O 裝置

LED 燈



按鈕開關

有源蜂鳴器

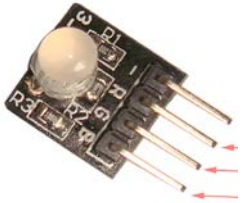
短腳為負極

長腳為正極

黑色膠封

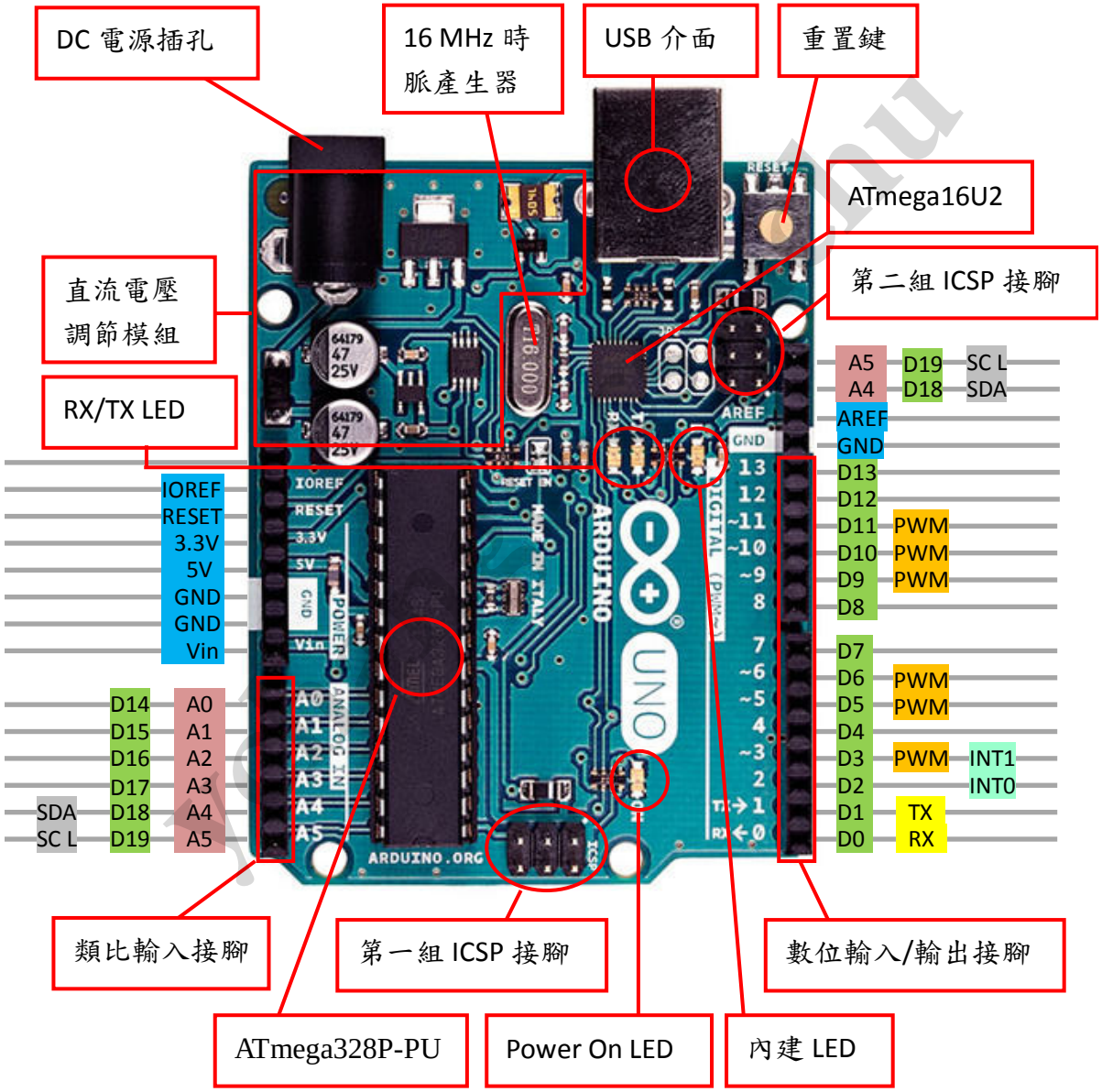
B50k 可變電阻

無源蜂鳴器

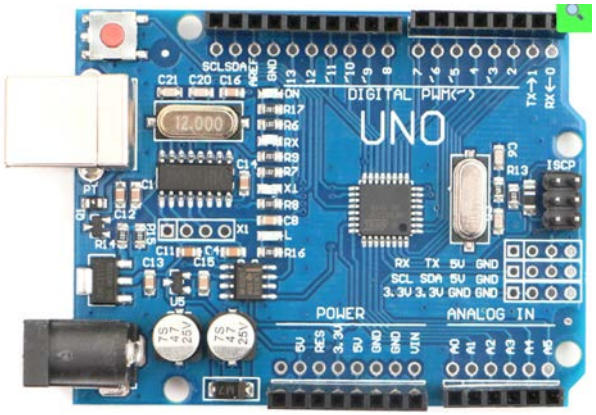


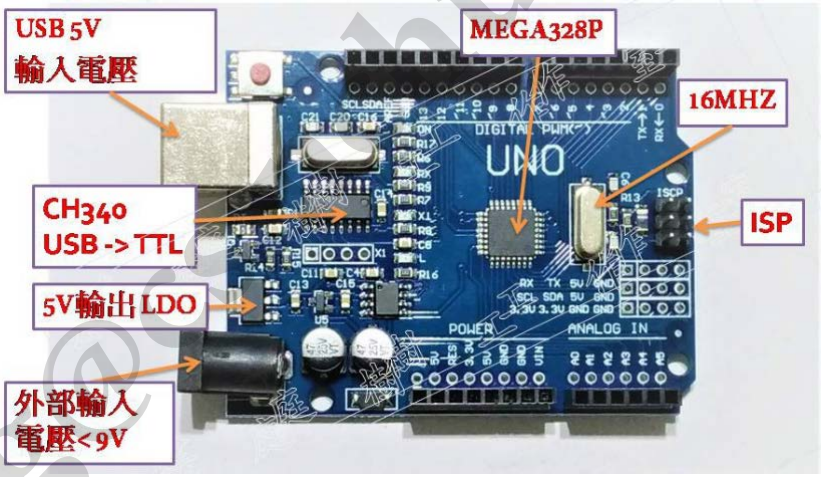
RGB LED 模組

Arduino UNO


DC 電源插孔
16 MHz 時脈產生器
USB 介面
重置鍵
ATmega16U2
第二組 ICSP 接腳
直流電壓調節模組
RX/TX LED
IOREF
RESET
3.3V
5V
GND
GND
Vin
A5 D19 SC L
A4 D18 SDA
AREF
GND
D13
D12
D11 PWM
D10 PWM
D9 PWM
D8
D7
D6 PWM
D5 PWM
D4
D3 PWM INT1
D2 INT0
D1 TX
D0 RX
D14 A0
D15 A1
D16 A2
D17 A3
SDA D18 A4
SC L D19 A5
類比輸入接腳
第一組 ICSP 接腳
數位輸入/輸出接腳
ATmega328P-PU
Power On LED
內建 LED

副廠Arduino Uno R3 經濟版





- 副廠Arduino Uno R3 經濟版，為了降低成本，ATMEGA328P晶片改為貼片封裝，ATmega16U2 改以 CH340G USB 晶片取代。並與原始Arduino Uno R3 100%相容。

基本 I/O 函式

- 基本 I/O 函式
  - pinMode( )
  - digitalRead( )
  - digitalWrite( )
  - analogRead( )
  - analogReference( )

pinMode( )

【描述】設定接腳輸入/輸出的模式
【語法】pinMode(pin, mode)
【參數】 pin：接腳的編號。
mode：可設定的模式有INPUT，INPUT_PULLUP，OUTPUT三種模式。
【傳回值】無
【範例】

```
1  void setup()
2  {
3    pinMode(13, OUTPUT);          //設定數位接腳 13 為 OUTPUT 模式
4  }
5
6  void loop()
7  {
8    digitalWrite(13, HIGH);       //將 HIGH 寫到(輸出)接腳 13
9    delay(1000);                  //延遲 1000ms=1s
10   digitalWrite(13, LOW);        //將 LOW 寫到(輸出)接腳 13
11   delay(1000);                  //延遲 1000ms=1s
12 }
```

pinMode( )

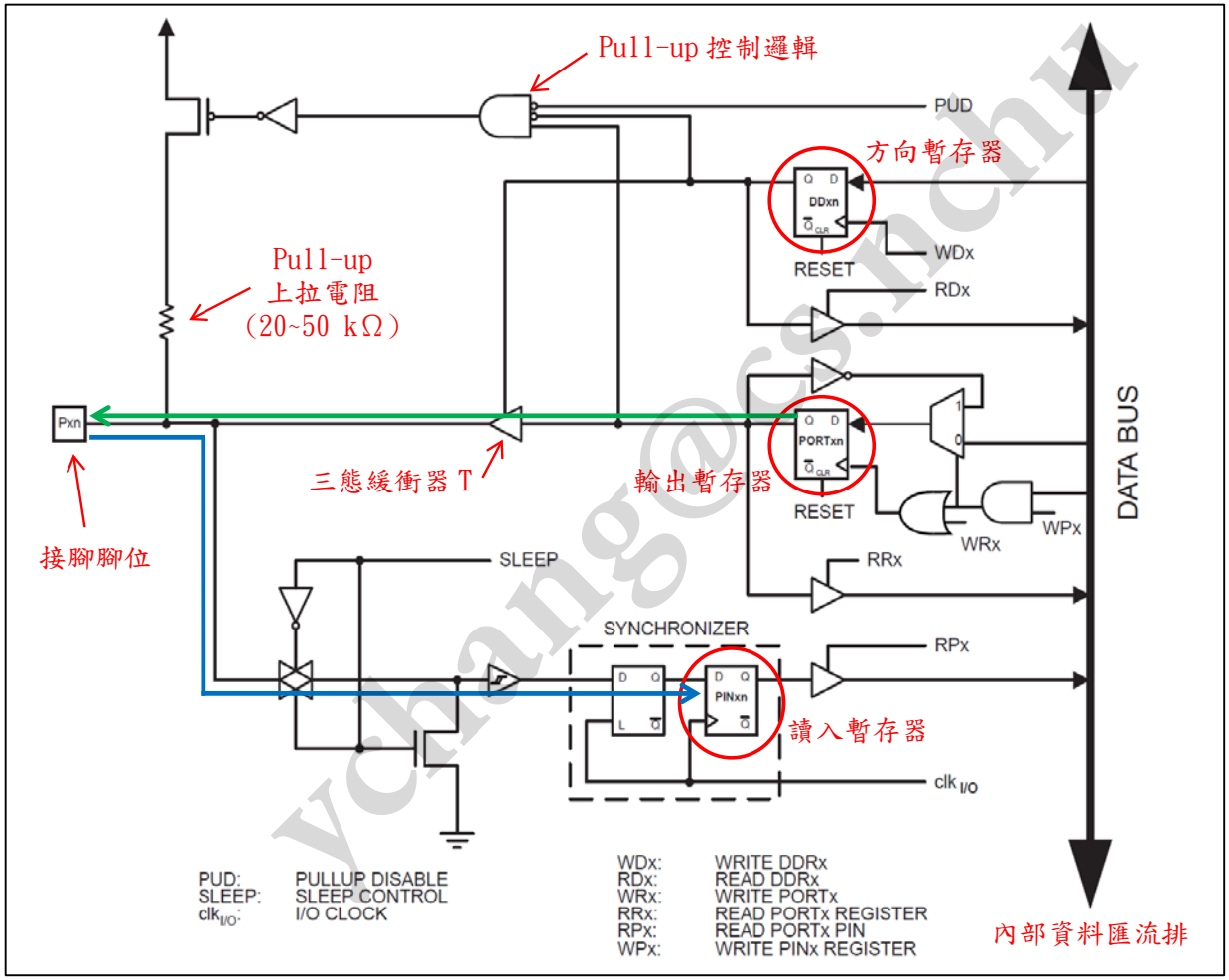
【說明】

(1) Arduino的接腳都是預設成INPUT模式，所以如果是輸入接腳，基本上不用pinMode設定也能直接使用；但如果是輸出接腳，就一定要用pinMode設定成OUTPUT模式，這樣才能輸出正確的訊號值。

(2) 大部分的Arduino開發板上接腳13都是固定接到內建的LED，屬於標準的輸出，所以要避免將接腳13設為INPUT模式。

(3) 與INPUT不同之處，INPUT_PULLUP會啟用Arduino開發板內建的上拉電阻，使得INPUT接腳在沒有任何訊號輸入或是浮接（floating）的狀態下，訊號固定拉升為HIGH，這樣才能避免讀取到不正確的值，所以如果是輸入接腳，一般都會直接使用INPUT_PULLUP模式來取代INPUT模式。

pinMode( )



ATmega328P
數位接腳的
內部電路

digitalRead( )

【描述】讀取數位輸入接腳的訊號值
【語法】digitalRead(pin)
【參數】pin：數位接腳的編號，以UNO而言，有0～13總共14支數位I/O接腳。
【傳回值】HIGH/LOW
【範例】

```
1  int ledPin = 13;    // 設定 LED 的接腳編號為 13
2  int inPin = 7;      // 設定按鈕的接腳編號為 7
3  int val = 0;        // 宣告 int 變數 val 來儲存 digitalRead 的傳回值
4
5  void setup()
6  {
7    pinMode(ledPin, OUTPUT);    //設定數位接腳 13 為 OUTPUT 模式
8    pinMode(inPin, INPUT);      //設定數位接腳 13 為 INPUT 模式
9  }
10
11 void loop()
12 {
     val = digitalRead(inPin);      //讀取接腳 7 的訊號值，存在 val
     digitalWrite(ledPin, val);     //將 val 變數的值寫(輸出)到 pin 13
   }
```

digitalRead( )

【說明】

(1) 參考圖2.7，對UNO而言，除了D0~D13為數位I/O接腳外，A0~A5的類比輸入接腳也可當成數位接腳來使用。

(2) 在此範例中若接腳7沒連接任何輸入裝置，則digitalRead()所讀到的值是隨機變動的，有可能是HIGH，也有可能是LOW。

(3) 建議如果是輸入接腳，一般都會直接使用INPUT_PULLUP模式來避免digitalRead()讀取到不確定的輸入值。

(4) 如果一個數位接腳設定成OUTPUT模式，也可使用analogRead( )函式讀取接腳數值。

digitalWrite( )

【描述】將訊號值寫到指定的數位輸出接腳
【語法】digitalWrite(pin, value)
【參數】
pin：接腳的編號。
value：有HIGH/LOW二種訊號值。
【傳回值】無
【範例】

```
1  void setup()
2  {
3    pinMode(13, OUTPUT);           //設定數位接腳 13 為 OUTPUT 模式
4  }
5
6  void loop()
7  {
8    digitalWrite(13, HIGH);        //將 HIGH 寫到(輸出)接腳 13
9    delay(1000);                   //延遲 1000ms=1s
10   digitalWrite(13, LOW);         //將 LOW 寫到(輸出)接腳 13
11   delay(1000);                   //延遲 1000ms=1s
12 }
```

digitalWrite( )

【說明】

(1) 若接腳已明確的使用pinMode設定成OUTPUT模式，則digitalWrite()的HIGH會對應成5V的輸出電壓，而LOW會對應到0V的輸出電壓。

(2) 若接腳沒有設定成OUTPUT模式就直接使用digitalWrite()，則接腳會以預設的INPUT模式來處理，這時候digitalWrite()的HIGH不會有正確的5V輸出，反而是啟用接腳內建的上拉電阻，而LOW則是關閉接腳內建的上拉電阻。

(3) 以上說明，digitalWrite()要有正確的功能，一定要先用pinMode將接腳設定成OUTPUT模式。

範例 4-1: 按鈕開關 Button

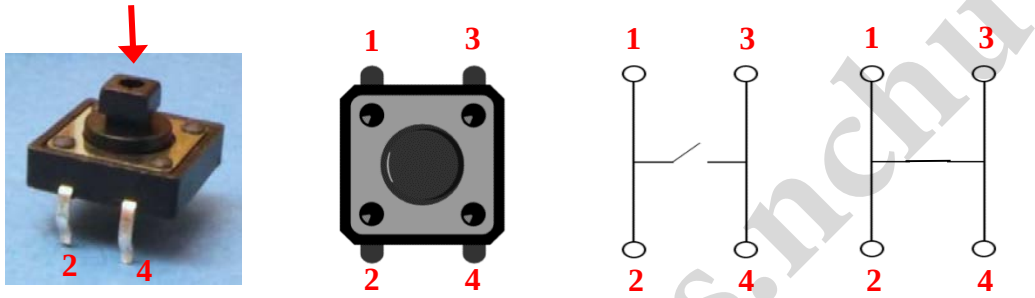
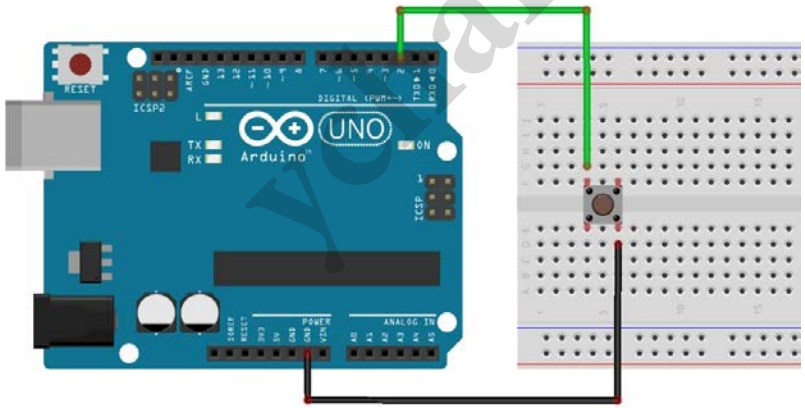
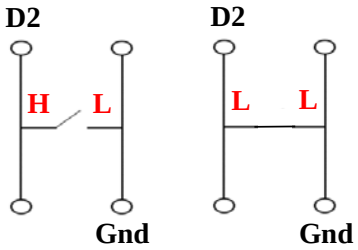


圖 5.3.1 按鈕開關 (a)實圖 (b)接腳圖 (c)放開狀態電路 (d)按壓狀態電路





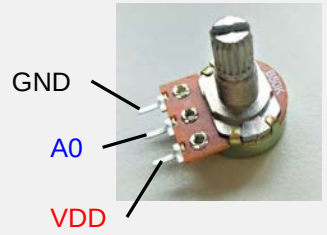
範例 4-1: 按鈕開關 Button

```
1  #define LedPin    13  //Uno 開發板上內建 LED 的接腳固定為 D13
2  #define ButtonPin 2   //指定按鈕開關的接腳為 D2
3
4  void setup() {
5    pinMode(LedPin,OUTPUT);           //設定 LED 接腳為輸出模式
6    pinMode(ButtonPin,INPUT_PULLUP); //設定 Button 接腳，並啟用內建的上拉電阻
7    digitalWrite(LedPin,LOW);         //初始 LED 為熄滅的狀態
8  }
9
10 void loop() {
11   if(digitalRead(ButtonPin)==LOW)  //讀取 Button 接腳的電位是否為 LOW
12     digitalWrite(LedPin,HIGH);     //若是，就代表按下開關，開啟 LED
13   else digitalWrite(LedPin,LOW);   //否則，關閉 LED
14 }
```

analogRead( )

【描述】讀取類比輸入接腳的訊號值
【語法】analogRead(pin)
【參數】pin：類比接腳的編號，以UNO而言，有A0～A5總共6支類比I/O接腳。
【傳回值】0~1023的整數值
【範例】

```
1  //讀取可變電阻(電位器)的輸出值
2  const int potPin=A0;
3  void setup() {
4    pinMode(potPin,INPUT);
5    Serial.begin(9600);
6  }
7
8  void loop() {
9  int val;
10 float volt;
11   val=analogRead(potPin);  //讀取 A0，傳回 0-1023
12   volt=val*0.00488;        //轉換成電壓值 5v/1024=0.00488v
13   Serial.print(val);  Serial.print(" => "); //印出結果
14   Serial.print(volt,1);   //小數點 1 位
15   Serial.println('V');
16   delay(1000);            //延遲 1 秒
17 }
```



digitalRead( )

【說明】

(1) 類比輸入接腳編號A0～A5是Arduino預設的常數，其數值對應為14～19。

(2) 因為Arduino開發板上的類比數位轉換器（ADC）是10-bit，代表0~5V的電壓會對應到0~1023個整數值，所以解析度等於5V/1024，約為每單位4.9mV。

(3) 根據官網說明，Arduino讀取類比訊號的時間，一次讀取大約是100微秒（microsecond），所以最高的讀取頻率為每秒10000次。

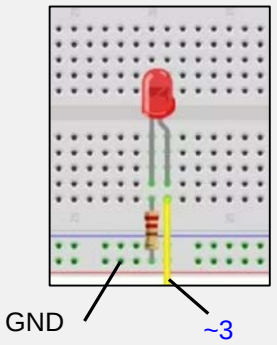
(4) 如果類比輸入接腳沒有連接任何裝置，則analogRead()的傳回值會是外在因素所造成的波動值，例如其他類比訊號的輸入、或是手靠近板子的距離遠近等。

(5) 如果數位接腳不夠用，A0～A5也可以改設定為數位輸出，例如 pinMode(A0, OUTPUT); 然後可以使用 digitalWrite(A0, HIGH/LOW); 寫出資料，但是這樣混合使用，在讀取其他類比腳位時會產生雜訊干擾，建議還是盡量避免。

analogWrite( )

【描述】使用duty cycle的值，將PWM的訊號波形輸出到指定的接腳
【語法】analogWrite(pin, value)
【參數】 pin：接腳的編號，以UNO而言，有3、5、6、9、10、11。
value：0~255的整數值，可表示PWM的duty cycle。
【傳回值】無
【範例】

```
1  //LED 燈的亮度控制
2  const int ledPin=3;      //LED輸出接腳
3  void setup() {
4    pinMode(ledPin,OUTPUT);
5    Serial.begin(9600);
6  }
7
8  void loop() {
9  int val;
10   for(val=0;val<256;val=val+15)  {  //漸亮
11     analogWrite(ledPin, val);
12     delay(100);  //間隔0.2秒
13   }
14   for(val=255;val>=0;val=val-15)  {  //漸暗
15     analogWrite(ledPin, val);
16     delay(100);  //間隔0.2秒
17   }
18 }
```



analogWrite( )

【說明】

(1) digitalWrite()只可輸出0V或5V二種電壓值，相較之下analogWrite()可以輸出0~5V中的任一電壓值，達到類比輸出的效果。

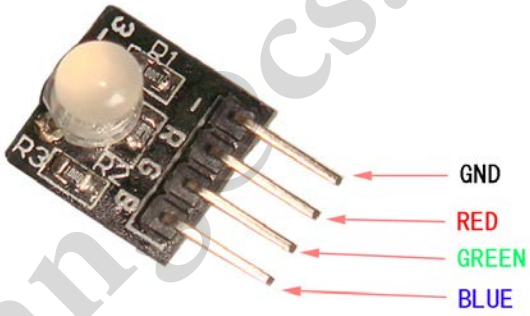
(2) Duty cycle（佔空比）表示在一個周期內，工作時間佔整個週期時間的比值，假設value=128，則duty cycle=128/255約為50%，輸出電壓=5Vx50%=2.5V。

(3) 在大部分的Arduino開發板，只要微控晶片是ATmega168或是ATmega328P系列的，包含UNO（請參考圖2.7），analogWrite()都可以使用在接腳3、5、6、9、10、11正常工作。

(4) 執行analogWrite()之後，指定的接腳就會持續輸出一個穩定的方波，其值由第二個參數value來決定，直到在相同的接腳執行到下一個analogWrite()，digitalRead()或是digitalWrite()時才會停止。

範例 4-2: 全彩RGB LED 模組

- 利用RGB三種不同顏色的輸入，混合出全彩的效果。
  - R 紅色: 0~255
  - G 綠色: 0~255
  - B 藍色: 0~255



- 程式碼

analogReference( )

【描述】設定analogRead()讀取類比輸入訊號時的參考電壓
【語法】analogReference(type)
【參數】type：參考電壓的種類，有DEFAULT、INTERNAL、INTERNAL1V1、INTERNAL2V56、EXTERNAL等5種。
【傳回值】無
【範例】
【說明】

(1) DEFAULT：使用預設參考電壓，在5V的開發板就是5V，在3.3V的開發板就是3.3V。
INTERNAL：使用內建的（built-in）參考電壓，在ATmega168或是ATmega328P微控器上為1.1V，而在ATmega8微控器上則為2.56V。（Arduino Mega無法使用）
INTERNAL1V1：使用內建的參考電壓1.1V。（僅在Arduino Mega上使用）
INTERNAL2V56：使用內建的參考電壓2.56V。（僅在Arduino Mega上使用）
EXTERNAL：使用從AREF接腳輸入的外部參考電壓，範圍限制在0~5V。

(2) 改變類比輸入的參考電壓之後，開始幾次analogRead讀取的值會有不準確的情況發生，之後就會恢復正確的讀取。

(3) 使用EXTERNAL外部參考電壓時，不要使用低於0V或是高於5V的電壓，因為這樣可能會導致Arduino開發板永久性的毀損。

tone( )

【描述】在指定的接腳輸出duty cycle為50%的方波，也就是HIGH，LOW各佔週期的一半。
【語法】tone(pin, frequency, [duration])
【參數】pin：接腳編號。
frequency：方波的頻率，資料型別為unsigned int。
duration：持續輸出方波的時間，資料型別為unsigned long，單位millisecond。此參數可有可無，若沒設定，則會不斷的輸出方波，直到叫用noTone()才會停止。
【傳回值】無
【範例】

```
1  #define pin 13
2  void setup()
3  {
4    pinMode(pin,OUTPUT);    //設定接腳 13 為 OUTPUT 模式
5  }
6
7  void loop()
8  {
9    tone(pin,31,2000);      //輸出頻率 31Hz 的方波，持續 2 秒鐘
10   delay(3000);            //延遲 3000ms=3s
11 }
```

tone( )

【說明】

(1) 此範例的效果會看到Uno板上的LED燈會快速的閃爍2秒之後，暫停1秒鐘，再開始快速閃爍2秒，如此不停的循環。

(2) 經測試，tone() 的頻率一定要≧31 Hz才有作用。

(3) tone() 函式是屬於非阻塞式（non-block）的執行，即使有設定持續（duration）時間，在呼叫tone()之後也會立刻返回，繼續執行下一道指令，所以此範例第10行的delay時間若≦2000ms就會看不出暫停的效果。

(4) tone() 主要應用在蜂鳴器或喇叭，使其發出聲音或音樂。

(5) 在Uno開發板上，因為tone() 函式的功能必須要使用到定時器Timer2，而Timer2同時也負責D3與D11接腳PWM的產生，所以在使用tone() 的時候會影響到這二個接腳PWM的輸出，使用者要自己避免。

noTone( )

【描述】在指定的接腳停止輸出方波。
【語法】noTone(pin)
【參數】pin：接腳編號。
【傳回值】無
【範例】

```
1  #define pin 13
2  void setup()
3  {
4    pinMode(pin,OUTPUT);    //設定接腳 13 為 OUTPUT 模式
5  }
6
7  void loop()
8  {
9    tone(pin,31);  delay(2000);    //輸出頻率 31Hz 的方波，並持續 2 秒鐘
10   noTone(pin);   delay(2000);    //停止輸出方波 2 秒鐘
11 }
```


【說明】

(1) 此範例也可達到LED燈快速的閃爍2秒之後，暫停2秒鐘，再開始快速閃爍2秒的效果。

蜂鳴器 Buzzer

(a)有源蜂鳴器

(b) 無源蜂鳴器

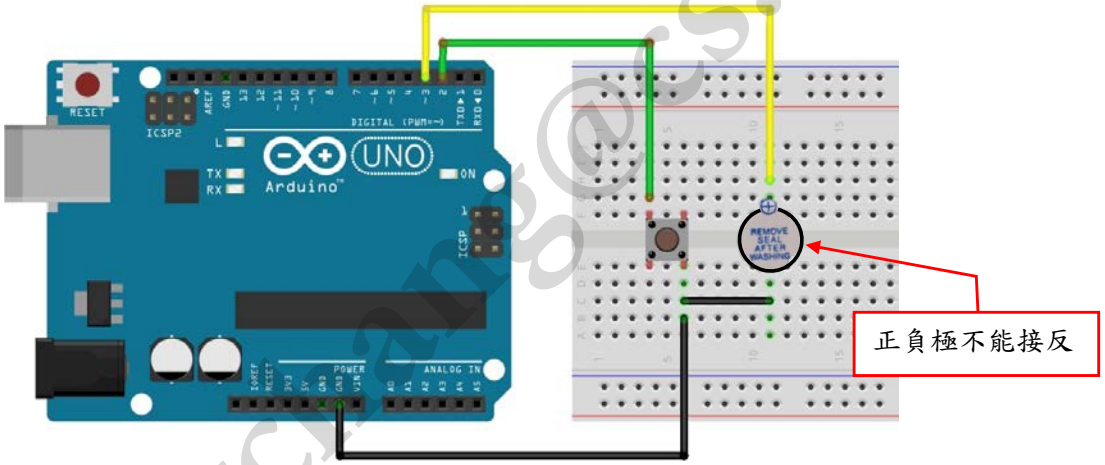
圖 5.4.1 蜂鳴器的種類

表 5.4.1 有源與無源蜂鳴器的差異

	有源蜂鳴器	無源蜂鳴器
尺寸	因為內含震盪電路，所以高度較高，為 9mm	高度略低，為 8mm
針腳	針腳一長一短，有正負極之分，長腳為正極，短腳為負極	針腳長度一樣，無正負極之分
有無膠封	針腳處有黑色膠封	無膠封，可直接看到 IC 電路
價格	稍貴	較便宜
功能	簡單易用，只能發出單音	使用方法較為複雜，能發出不同的音高，變化較多

範例 4-3: 有源蜂鳴器

- 使用有源蜂鳴器編輯三首不同節奏的音樂，配合按鈕開關，以按一下就換一首的方式呈現，特別注意音樂開始播放就不會被中斷，一直到結束才會停止。

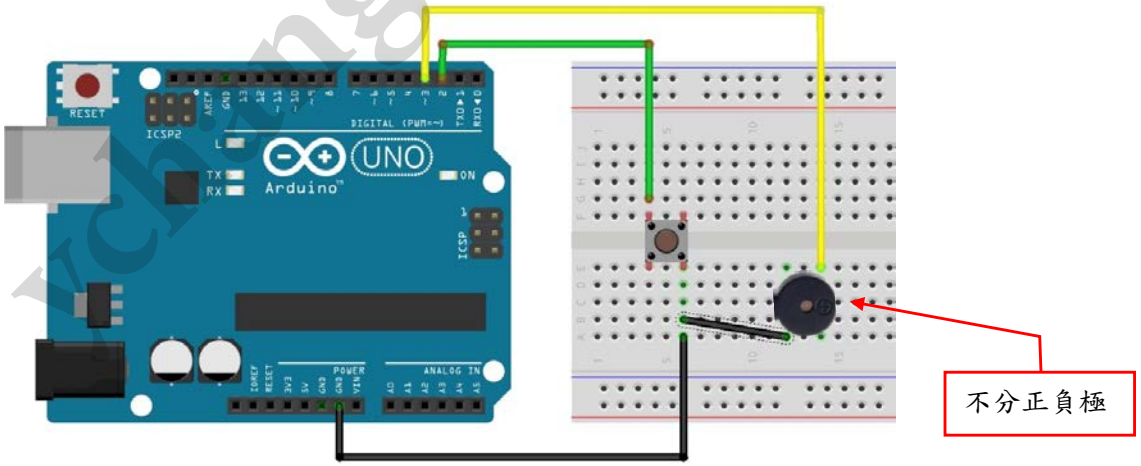


- 程式碼

範例 4-4: 無源蜂鳴器

- 編輯以下三首不同的兒歌片段，使用無源蜂鳴器演奏，配合按鈕開關，以按一下換一首的方式呈現，特別注意音樂開始播放就不會被中斷，一直到結束才會停止。

小蜜蜂	| 5 3 3 - | 4 2 2 - | 1 2 3 4 | 5 5 5 - |	速度 140 拍/分鐘
蝴蝶	| 1 12 3 3 | 21 23 1 - | 3 34 5 5 | 43 45 3 - |	速度 80 拍/分鐘
望春風	| 1· 1 2 4 | 5 45 6 - | i· 6 65 4 | 5 - - - |	速度 60 拍/分鐘



- 程式碼

範例 4-4: 無源蜂鳴器

- 構成聲音的三要素：音高，音量與音色

(1) 音高（Pitch）：指聲音的高低，在聲波的表現就是頻率，頻率越高，聲音就越高，計量單位為赫茲（Hz）。人類的聽覺可接收的頻率範圍大約落在20Hz到20KHz之間，超過此範圍的統稱「超音波」，實際上聽覺頻率的範圍因人而異，有人遲鈍，也有人會特別靈敏，但隨著年齡的增長，可聽到的頻率範圍一定是逐漸縮小。

(2) 音量（Loudness）：即聲音的大小，在聲波的表現就是振幅，振幅愈大代表音量愈大，計量單位為分貝（dB）。

(3) 音色（Timbre）：聲音的特色，在聲波的表現就是波形的特徵，不同的發聲體會有不同的聲波特徵。例如：鋼琴與小提琴就有截然不同的音色。

我們直接引用的結論就是（1）方波的頻率可決定音高，而（2）方波的佔空比（duty cycle）則可決定音量。雖然我們可以使用PWM的方式來控制蜂鳴器的音高與音量，但是對初學者而言，比較簡單的方式還是使用Arduino內建的函式tone()，tone()可以很簡單的產生週期性的方波，使用者可以改變頻率跟持續輸出的時間，但唯一的限制，就是產生方波的佔空比（duty cycle）固定為50%，這是不能改變的，因此使用tone()所產生的音量一定會保持不變，不能有大小聲的變化，但這不會影響我們要控制的音高旋律。

pulseIn( )

【描述】在指定的接腳上量測目標脈衝（HIGH或LOW）持續的時間。舉例而言，如果目標脈衝為HIGH，則pulseIn()會等待接腳上的電位出現LOW轉HIGH之後開始計時，直到訊號再變回LOW才停止計時，此為一個完整的脈衝，最後傳回HIGH持續的時間（微秒）。

【語法】pulseIn(pin, value, [timeout])

【參數】pin：接腳編號。

value：目標脈衝，HIGH或LOW。

timeout：設定監聽時間，若監聽時間內沒有出現目標脈衝或是沒有量測到一個完整的脈衝，則傳回0值。此參數若沒設定，則預設監聽時間為1秒鐘。

【傳回值】傳回0或目標脈衝持續的時間，單位是微秒（microsecond）。

【範例】

```
1  #define pin 0
2  unsigned long oldVal=0,newVal=0;
3  void setup()
4  {
5    Serial.begin(9600);  //設定串列埠傳輸速率為 9600 bps
6    pinMode(pin,INPUT);  //設定接腳 0(Rx)為 INPUT 模式
7  }
8  void loop()
9  {
10   newVal=pulseIn(pin,HIGH);     //監聽 D0 接腳上的 HIGH 脈衝
11   if(oldVal!=newVal) {          //若 HIGH 持續時間有改變才印出數值
12     Serial.println(newVal);
13     oldVal=newVal;
14   }
15 }
```

pulseIn( )

【說明】

(1) 在Uno開發板上，因為D0固定是UART串列傳輸的接收（Rx）接腳，所以此範例我們將D0設為監聽接腳，只要在PC端串列埠監控視窗的傳送框輸入文數字，就可讀取目標脈衝，不需要連接其它的輸入設備。

(2) 根據官方網頁上的說明，pulseIn()使用的時機取決於設計者的經驗，通常在目標脈衝持續太長的情況下較容易發生錯誤，有效的脈衝持續時間介於10微秒到3分鐘之間。

(3) pulseIn() 是屬於阻塞式（block）的函式，一旦執行就要等到目標脈衝出現或是監聽時間結束（timeout）才會返回程式，執行下一道指令。

(4) pulseIn() 常應用在偵測按鈕時間的長短，短按或長按，以決定不同的執行動作

pulseInLong( )

【描述】由於pulseIn() 在量測長脈衝時會出現較大的誤差，相比之下，pulseInLong()更適合使用在長脈衝的量測。因為pulseInLong()會使用到micros()時間函式，所以一定要在允許中斷發生的情況下才能使用pulseInLong()。

【語法】pulseInLong(pin, value, [timeout])

【參數】

pin：接腳編號。

value：目標脈衝，HIGH或LOW。

timeout：設定監聽的時間，此為非必要參數。若監聽時間內沒有出現目標脈衝或是脈衝長度太長，則會停止讀取並回傳0值。若沒設定，則預設是1秒鐘。

【傳回值】傳回0或目標脈衝持續的時間，單位是微秒（microsecond）。

shiftIn( )

【描述】在指定的接腳上以一次一個bit的方式接收資料，可以選擇從最高（MSB）或是最低（LSB）位元開始接收資料，直到接收滿8個bit（等於一個位元組），再傳回完整的byte。

【語法】shiftIn(dataPin, clockPin, bitOrder)

【參數】

dataPin：讀取資料的接腳。

clockPin：時脈接腳。

bitOrder：讀取bit的順序，MSBFIRST表示從最高位元開始讀取，而LSBFIRST則是從最低位元開始讀取。

【傳回值】讀取到的byte

。

shiftOut( )

【描述】在指定的接腳上以一次一個bit的方式輸出資料，可以選擇從最高（MSB）或是最低（LSB）位元開始傳送，直到指定的位元組（8個bit）傳送完畢。

【語法】shiftOut(dataPin, clockPin, bitOrder, value)

【參數】

dataPin：輸出bit的接腳。

clockPin：時脈接腳。

bitOrder：輸出bit的順序，MSBFIRST表示從最高位元開始輸出，而LSBFIRST則是從最低位元開始輸出。

value：要輸出的位元組

【傳回值】無