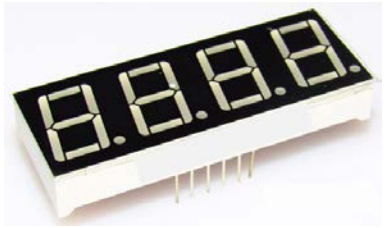
Chap 5
Arduino的I/O (2)

使用的 I/O 裝置

LED 燈

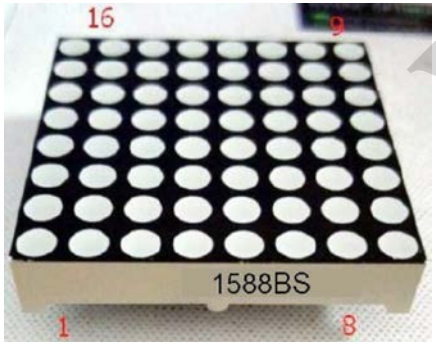
一位數七段

按鈕開關

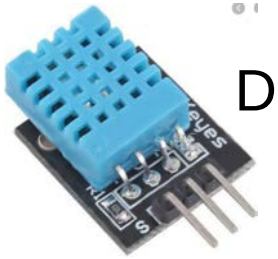


四位數七段

DS1302 RTC時鐘模組

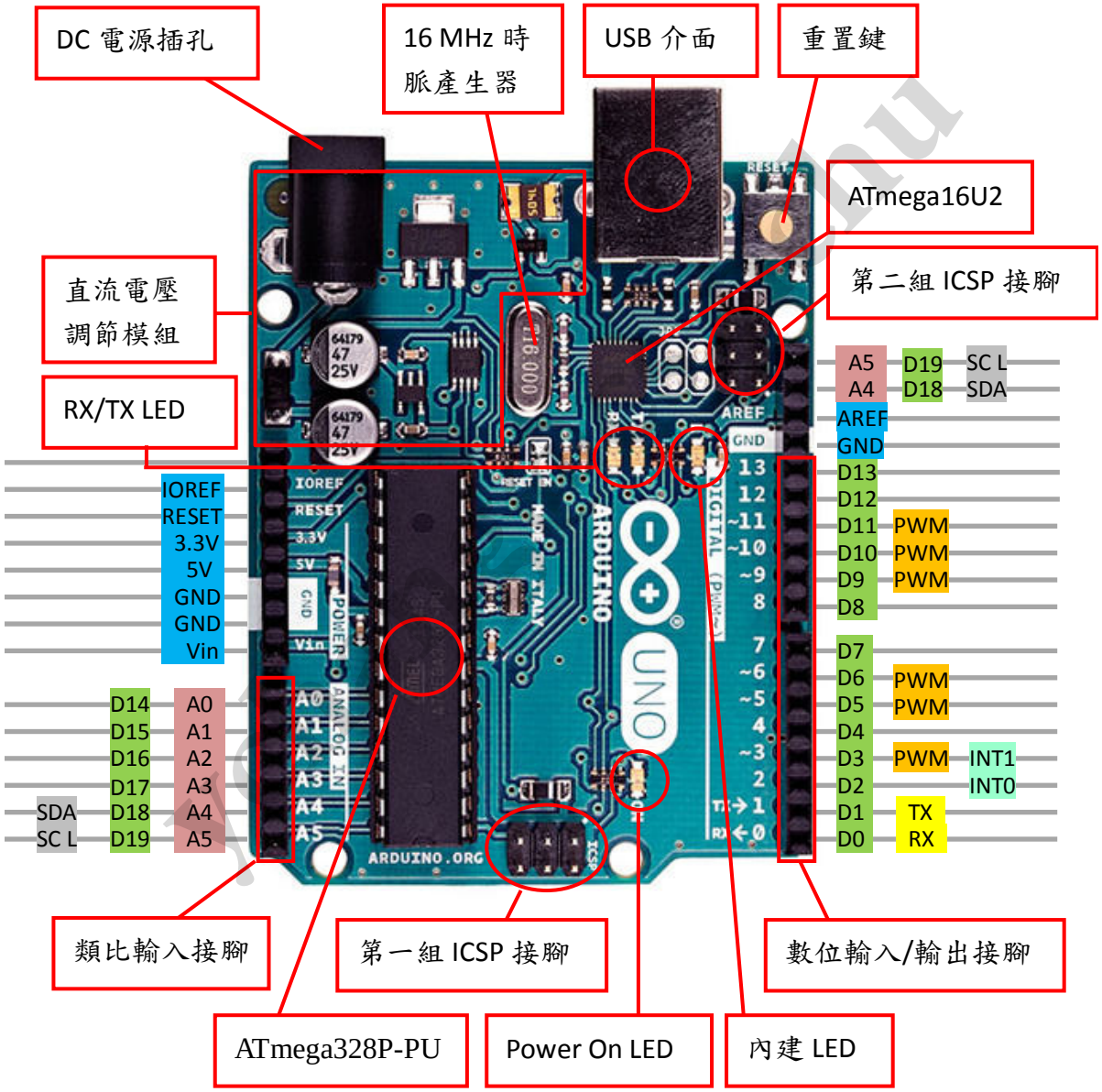


8x8 LED矩陣

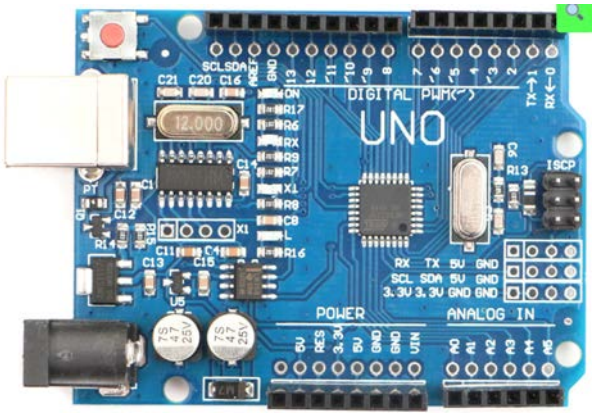


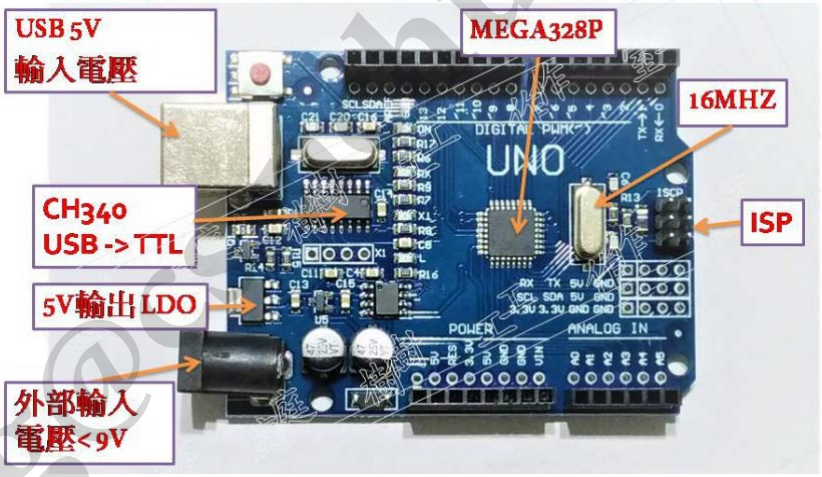
DHT11 溫濕度模組

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

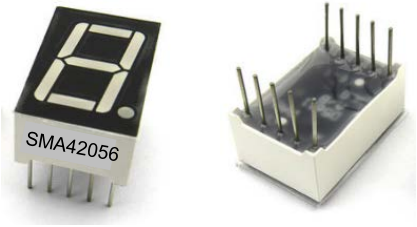
副廠Arduino Uno R3 經濟版

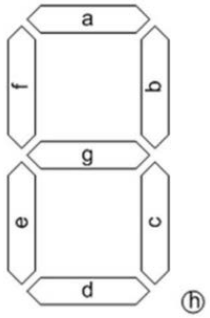


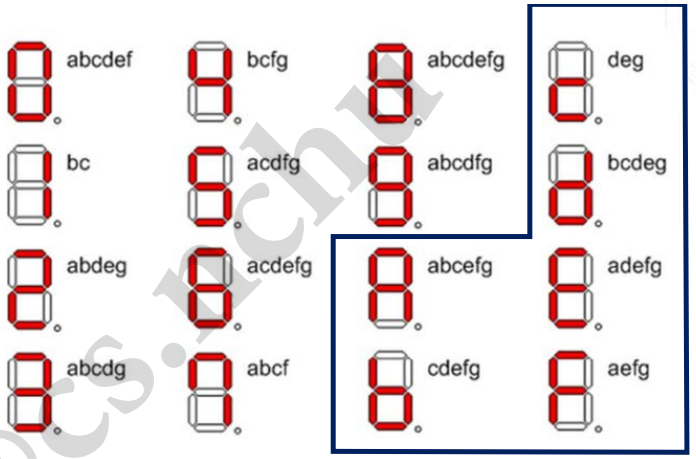


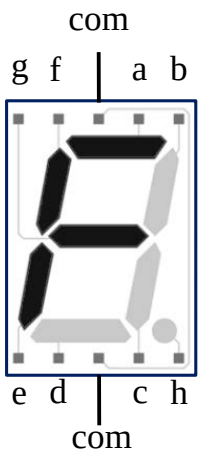
- 副廠Arduino Uno R3 經濟版，為了降低成本，ATMEGA328P晶片改為貼片封裝，ATmega16U2 改以 CH340G USB 晶片取代。並與原始Arduino Uno R3 100%相容。

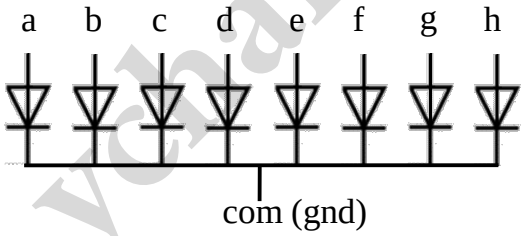
5-1 七段顯示器

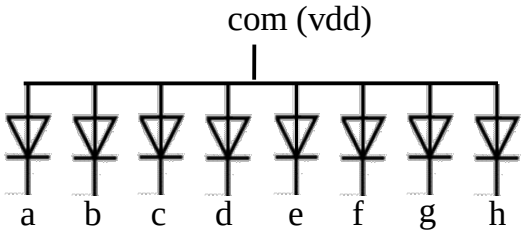












(a)共陰極電路 **(b)共陽極電路**

圖 5.5.2 七段顯示器的種類

5-1 一位數七段顯示器

表 5.5.1 共陰極七段顯示器數字 0~9 的編碼

	a	b	c	d	e	f	g	h
0	1	1	1	1	1	1	0	0
1	0	1	1	0	0	0	0	0
2	1	1	0	1	1	0	1	0
3	1	1	1	1	0	0	1	0
4	0	1	1	0	0	1	1	0
5	1	0	1	1	0	1	1	0
6	1	0	1	1	1	1	1	0
7	1	1	1	0	0	1	0	0
8	1	1	1	1	1	1	1	0
9	1	1	1	1	0	1	1	0

```
const byte d0=B11111100;  // 數字0的定義，B表示二進位的格式
if(d0&B10000000) digitalWrite(2, HIGH); else digitalWrite(2, LOW);  //輸出a段的值1
if(d0&B01000000) digitalWrite(3, HIGH); else digitalWrite(3, LOW);  //輸出b段的值1
if(d0&B00100000) digitalWrite(4, HIGH); else digitalWrite(4, LOW);  //輸出c段的值1
if(d0&B00010000) digitalWrite(5, HIGH); else digitalWrite(5, LOW);  //輸出d段的值1
if(d0&B00001000) digitalWrite(6, HIGH); else digitalWrite(6, LOW);  //輸出e段的值1
if(d0&B00000100) digitalWrite(7, HIGH); else digitalWrite(7, LOW);  //輸出f段的值1
if(d0&B00000010) digitalWrite(8, HIGH); else digitalWrite(8, LOW);  //輸出g段的值0
if(d0&B00000001) digitalWrite(9, HIGH); else digitalWrite(9, LOW);  //輸出h段的值0
```

因為以上的程式碼具有高度的規律性，我們可以使用迴圈的方式改寫如下，可大幅的減少程式碼的數量

```
for(i=0; i<8; i++)  {
  mask=B10000000>>i;  //將1右移到正確的位置
  if(d0&mask) digitalWrite(2+i, HIGH); else digitalWrite(2+i, LOW);  //從a段到h段
}
```

5-1 一位數七段顯示器

```
/*** seg7.h ***/
const byte seg7_digit[10]={ B11111100,  //數字 0
                            B01100000,  //數字 1
                            B11011010,  //數字 2
                            B11110010,  //數字 3
                            B01100110,  //數字 4
                            B10110110,  //數字 5
                            B10111110,  //數字 6
                            B11100100,  //數字 7
                            B11111110,  //數字 8
                            B11110110}; //數字 9
int seg7_x1_FirstPin;  //一位數七段顯示器的第一隻接腳
int seg7_x4_FirstPin;  //四位數七段顯示器的第一隻接腳

//*** 初始化一位數七段顯示器 8 隻連續的接腳
void seg7_x1_init(int pin) {
seg7_x1_FirstPin=pin;  //設定一位數七段顯示器的第一隻接腳
for(int i=0; i<8; i++) pinMode(seg7_x1_FirstPin+i, OUTPUT);
}

//*** 一位數七段顯示器顯示數字
void seg7_x1_display(int num) {
byte mask;

  for(int i=0; i<8; i++)  {
    mask=B10000000>>i;  //將 1 右移到正確的位置
    if(seg7_digit[num] & mask) digitalWrite(seg7_x1_FirstPin+i, HIGH);
    else digitalWrite(seg7_x1_FirstPin+i, LOW);
  }
}
```

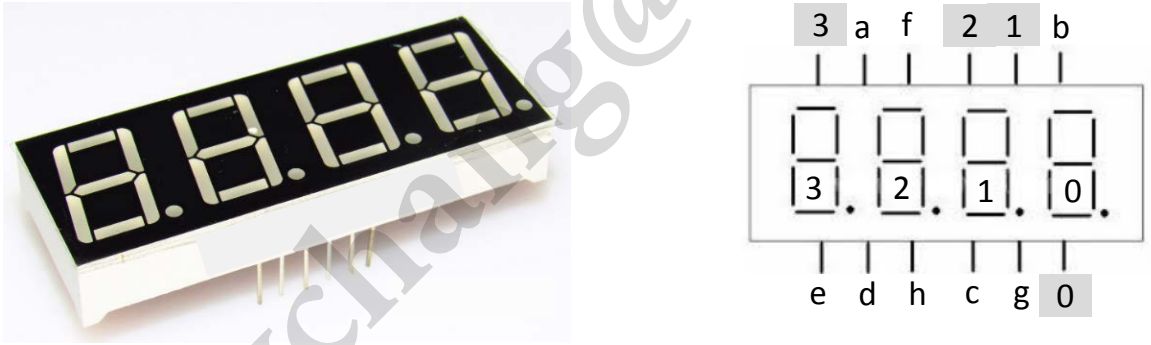
範例 5-1

- 使用一位數的七段顯示器來計數按鈕開關按下的次數，以按一下就加1的方式顯示，從數字0開始一直到數字9，然後又從0開始。

```
1  /*** 一位數七段顯示器範例 ***/
2  #include  “seg7.h”
3  #define Button 10  //指定按鈕開關的接腳為 D10
4  int num=0, flag=0;
5
6  void setup() {
7    pinMode(Button,INPUT_PULLUP); //設定 Button 接腳，並啟用內建的上拉電阻
8    seg7_x1_init(2);      //初始化七段顯示器，第一隻接腳為 D2，到 D9
9    seg7_x1_display(0);  //顯示數字 0
10 }
11
12 void loop() {
13   if(digitalRead(Button)==LOW)  //讀取 Button 接腳的電位是否為 LOW
14     { num=++num%10; flag=1; }   //若是，就代表按下開關，num+1 後取 10 的餘數
15   if(flag==1) {
16     seg7_x1_display(num);  //顯示數字 num
17     flag=0;
18   }
19 }
20
```

5-1 四位數七段顯示器

- 一顆七段顯示器需要8隻接腳控制顯示，四顆豈不是要32隻接腳，可是Arduino UNO所有數位接腳的總量也才14隻，即使全部用上了也不夠，那要如何顯示四個數字呢？

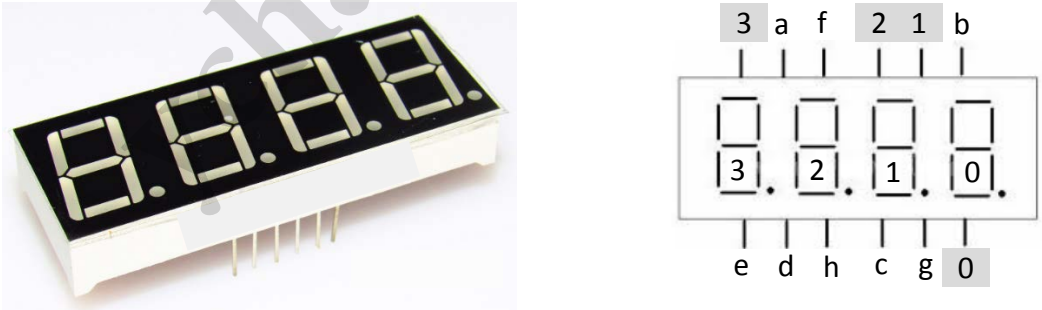


(a) (b)

圖 5.5.3 共陰極四位數七段顯示器及其腳位

5-1 四位數七段顯示器

- 其實這個問題的解法也不難，我們還是一次只顯示一個數字，從左到右（或從右到左）輪流顯示，只要二次掃描的間隔時間夠短，通常是小於1/16秒，就會因為眼睛視覺暫留的效應，而達到四個數字同時顯示的視覺效果，再次強調這不是真正的同時顯示，只是快速輪動所造成的視覺假象。
- 經由以上的解釋，除了原來8段LED的控制針腳a，b，c，d，e，f，g，h之外，還需要4隻針腳來控制哪一顆七段顯示器要動作，所以總共會有12隻針腳，如圖5.5.3(b)所示，其中0，1，2，3針腳分別控制四顆七段顯示器，因為是共陰極的電路，所以邏輯1代表致能，而邏輯0則為禁用。



(a) (b)

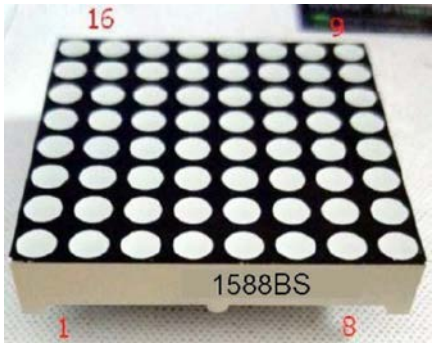
圖 5.5.3 共陰極四位數七段顯示器及其腳位

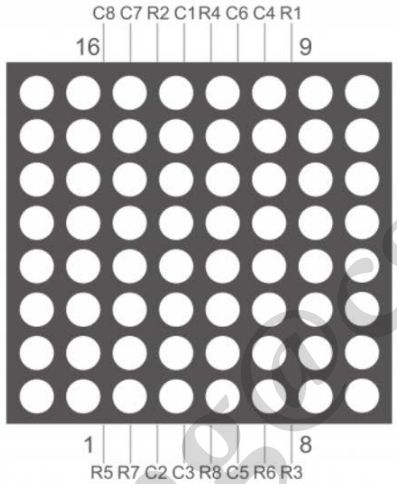
5-1 四位數七段顯示器

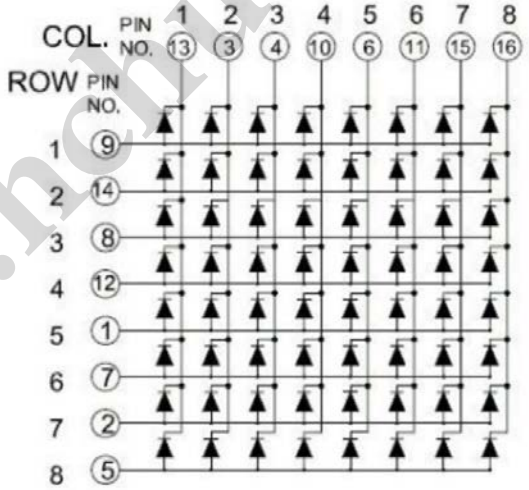
- 因為擴充了四位數的七段顯示器，所以我們必須把下列的程式碼加入seg7.h才夠使用，特別注意在數字顯示的函式中，必須指定四顆當中要在哪一顆顯示

```
1  //*** 初始化四位數七段顯示器 12 隻連續的接腳
2  void seg7_x4_init(int pin) {
3  seg7_x4_FirstPin=pin;  //設定四位數七段顯示器的第一隻接腳
4  for(int i=0; i<12; i++) pinMode(seg7_x4_FirstPin+i, OUTPUT);
5  }
6
7  //*** 四位數七段顯示器顯示數字
8  void seg7_x4_display(int digit, int num) {
9  byte mask;
10
11   //先關掉四顆七段顯示器
12   for(int i=8; i<12; i++) digitalWrite(seg7_x4_FirstPin+i, LOW);
13   //再致能指定的七段顯示器
14   switch(digit) {
15     case 0: digitalWrite(seg7_x4_FirstPin+8, HIGH); break;
16     case 1: digitalWrite(seg7_x4_FirstPin+9, HIGH); break;
17     case 2: digitalWrite(seg7_x4_FirstPin+10, HIGH); break;
18     case 3: digitalWrite(seg7_x4_FirstPin+11, HIGH); break;
19   }
20   //顯示數字
21   for(i=0; i<8; i++)  {
22     mask=B10000000>>i;
23     if(seg7_digit[num] & mask) digitalWrite(seg7_x4_FirstPin+i, HIGH);
     else digitalWrite(seg7_x4_FirstPin+i, LOW);
   }
}
```

5-2 8x8 LED 矩陣







- 程式碼5-2.1
- 程式碼5-2.2