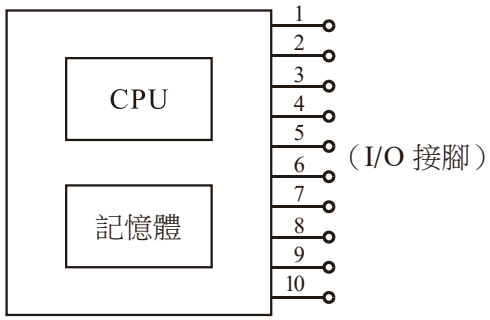
第1單元

準備工作

單晶片

單晶片是將 CPU、記憶單元整合在一個 IC 裡面，然後預留若干 I/O 接腳如下圖。這些 I/O 接腳可以讓使用者，以軟體程式的方式，指派爲低電位或高電位，進而控制所有負載的 ON 與 OFF。這動作看起來很簡單，但因爲單晶速度快，速度快就能有變化，就協助改善很多共業控制系統。其次，其體積很小，體積小就能嵌入原有的工業產品，所以又稱爲嵌入式控制。例如家電的冷氣、洗衣機、風扇、汽車的防鎖死煞車、胎壓偵測、自動駕駛等都已經使用單晶片協助控制，因爲都是透過軟體簡化硬體設備，這樣當然可縮短產品開發流程與降低硬體設備成本，所以學習單晶片已經是時代趨勢。其次，單晶片又稱單板電腦，因爲一小塊電路板，內部就含 CPU 與記憶單元，其功能就如同小型的電腦，已具有輸出入、決策、迴圈、陣列、輸出入等電腦所具有的功能，方塊示意圖如右：



單晶片模組

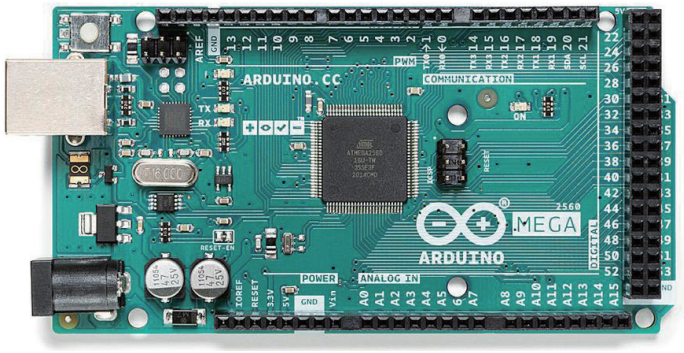
Arduino Maga 2560

目前常聽到的 Macro:Bit、8051、Arduino 等都是單晶片，Macro:Bit 比較著重在圖形開發介面，主要是來讓小學生玩簡單的聲光控制遊戲；8051 與 Arduino 都是使用文字式的開發工具，是眞實的嵌入式工業晶片，可眞實改善許多生活與工業控制，適合中學生以上與社會人士學習嵌入式系統控制。但 8051 已經過時，漸漸被 Arduino 取代，Arduino 官網是 www.arduino.cc。Arduino 之所以能異軍突起，我是認爲它主張開源，它的軟硬體都是開放（甚至使用者用 Arduino 所開發的程式也僅能公開），使用者可以站在巨人的肩膀，繼續接力開發新產品。其次，Arduino 輸出電流變大，可以直接驅動 LED，使的產品的電路也變的簡單。還有，Arduino 腳位也變多，這樣可以直接驅動一些需要重複掃描輸出的元件，例如，本書要介紹的四位數七段顯示器與點陣 LED，與傳統 8051 單晶相較，也變簡單了。Arduino 依照使用者不同的需求，開發很多版本的微控板，其中型號 Mega 2560 的 I/O 腳位共有 70 隻，因爲腳位多、電流也夠大，可直接驅動本書介紹的四位數七段與點陣 LED，這樣用來實作這些生活科技常用電路，其電路與軟體最爲省事，本書就選用 Mega 2560 介紹生活科技產品，下表是其技術規格（摘自 Arduino 官網）。

OVERVIEW | TECH SPECS | DOCUMENTATION

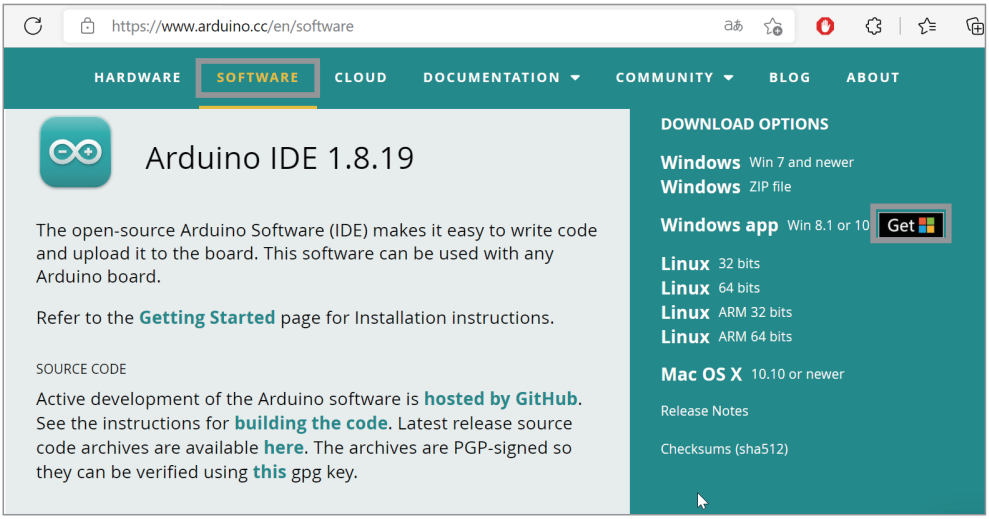
Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz
LED_BUILTIN	13
Length	101.52 mm

下圖是其微控板實體照片（摘自 Arduino 官網），中間黑色正方形 IC 就是 MEGA2560 單晶片，共有 70 隻 I/O 腳位，這些接腳都可以使用軟體的設定，分別指派當作數位輸出、數位輸入、或有上拉電阻 INPUT_PULL UP 的輸入等 3 種功能，這會在本書陸續介紹。



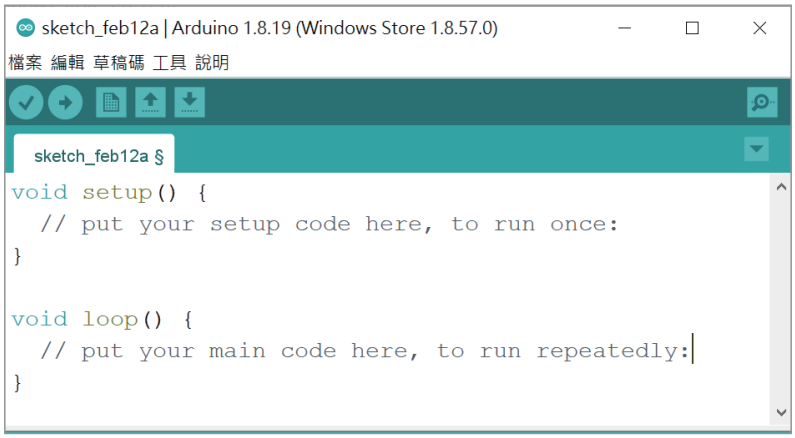
Arduino軟體下載與安裝

所有傳統的單晶片，都要自備類似記事本的編輯器，編寫程式、存檔、離開，然後使用其所提供的編譯程式，若有錯誤則要繼續開啓記事本修改，再編譯，直到完全正確。然後還要購買萬元燒錄器，將程式燒到單晶片。但是 Arduino 就神了，竟然有免費整合編輯視窗，整合以上編輯、編譯、燒錄（上傳）於單一視窗，此稱爲整合編輯程式（IDE, Integrated Development Environment 的縮寫）。請於 Arduino 官網點選『SOFTWARE/DOWNLOADS』，畫面如下，請點選『Windows app win8.1 or 10』，即可下載最新安裝執行檔（*.exe）。接著，請開啓檔案總管，至下載區按兩下該執行檔，即可安裝。



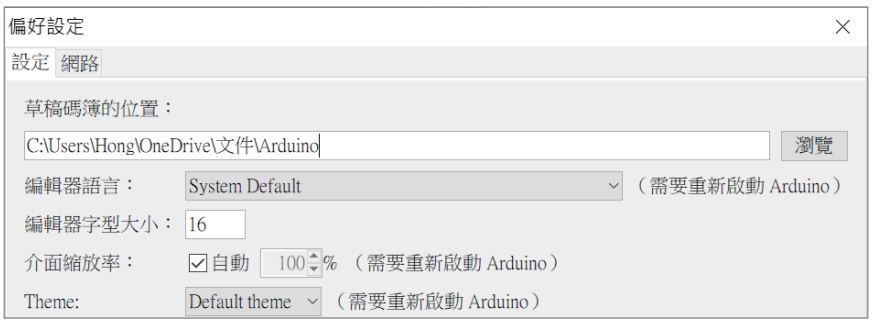
IDE畫面

下圖是開啓 Arduino IDE 畫面。



功能表的語言設定

理論上安裝後，功能表的語言會依據作業系統的語言自動完成設定，但若不是您所要的語言，請於功能表點選『檔案 / 偏好設定』，如下圖，即可在此畫面點選您要的語言，還有編輯器文字的大小等等。



單晶微控板使用步驟

Arduino 的單晶微控板使用步驟分別是插入微控板、點選開發板型號、點選通訊埠編號，分別說明如下：

1. **插入微控板**

   請依照指示，將微控板 USB 插頭插入電腦 USB 插座。請留意微控板右上角電源指示燈是否亮起。

2. **點選開發板型號**

   請點選功能表的『工具 / 開發板』即可點選您所使用的微控板型號。(請依照您的微控板型號點選，筆者是點選『Arduino/Genuino Mega or Mega 2560』，若您是 UNO 版，也是在此點選，因爲不同版本腳位數量不一樣，選錯了就無法正確編譯程式)

3. **點選通訊埠編號**

   請點選功能表的『工具 / 序列埠』即可點選您所使用的序列埠編號（備註：系統會出現可用編號 com1 或 com2,3,4,5 等等，讓您點選，有時候會同時出現很多個編號，請按照順序點選，直到可上傳 (或稱燒錄) 爲止。其次，有些電腦不會自動抓到通訊埠 (com1、com2…)，請到網路搜尋與下載『CH34x-install-Windows』，並安裝，直到出現通訊埠。)

4. **取得開發版資訊**

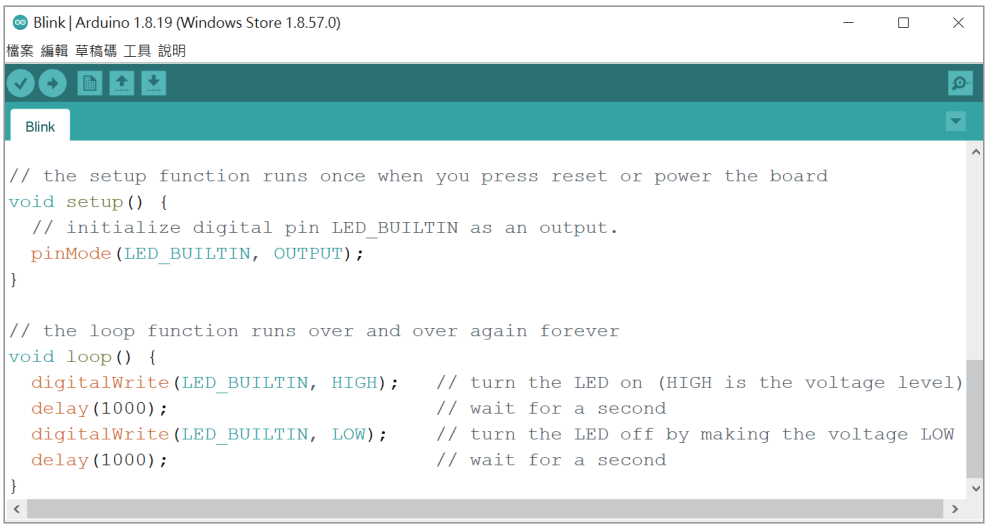
   請點選功能表的『工具 / 取得開發版資訊』即可取得您的微控板資訊。若出現下圖，才表示以上設定就緒，才能上傳程式。(實驗的中途，若改插入別人的微控板，那也要重覆以上步驟，直到看到下圖，才表示有抓到此微控板，才能上傳程式。其次，下圖左是原廠的微控板，若不是原廠，那可能就像下圖右，顯示『未知的開發板』。)

自我測試

Arduino 微控板有內植一顆 LED，腳位編號是 13，別名是 LED_BUILTIN。其次，整合開發環境安裝後，內部也含一個自我測試程式，這樣就可以測試此微控板是否已經安裝完成。進行單晶片控制都要這樣一步一步來，這樣當問題發生時，才能一步一步除錯，逐漸縮小錯誤範圍，並排除故障。以下說明如何自我測試：

1. 開啓自我測試程式

下圖是開啓測試程式 Blink（請點選功能表的『檔案 / 範例 /01.Basics/Blink』），其功能是直接使用微控板預植的 LED（不管什麼板 UNO、MEGA…等），都是腳位 13，以常數『LED_BUILTIN』表示），並令其明滅閃爍各一秒。



2. 驗證程式

按一下工具列的『驗證』按鈕  ，即可編譯程式。

3. 上傳程式

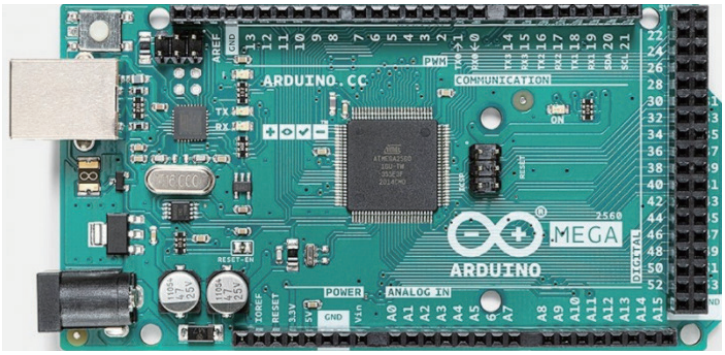
按一下工具列的『上傳』按鈕  ，即可上傳程式到微控板（上傳又稱爲燒錄）。上傳結束將會自動執行程式，請觀察微控板上所預植 LED 是否明滅（此顆 LED 就在腳位 13 旁）。

補充說明

1. setup() 函式僅在程式一開始時執行一次。所以通常放置執行程式的初始設定、或程式執行後只執行一次的指令。
2. pinMode(LED_BUILTIN, OUTPUT); 是指派編號 13 這隻腳的功能是輸出。
3. loop() 函式則會重複不斷的被執行，所以就放置一些需要反覆執行的指令。
4. digitalWrite(LED_BUILTIN, HIGH); 是指派腳位 13 爲高電位，請用三用電表量其電壓，將會有 5V，這樣就可以讓 LED 發光。
5. delay() 函式是程式延遲程式，單位是毫秒 ms，千分之一秒。因爲指派 LED 亮，總要停留一點時間，使用者才有機會看到。請自行調整時間，並觀察執行結果。
6. digitalWrite(LED_BUILTIN, LOW); 是指派腳位 13 爲低電位，請用三用電表量其電壓，將會有 0V，這樣就可以讓 LED 熄滅。
7. 雙斜線『//』稱爲註解，此爲單列註解，僅給人看，電腦不予編譯，沒有鍵入也沒關係。其次『/*』與『*/』之間的文字，也都是註解，此稱爲多列註解，這些註解都是給人看的，電腦不會編譯。

I/O腳位探索

Arduino Mega 2560 有 70 個 I/O 接腳，如下圖。



這些接腳都可單獨使用，那就使用腳位編號（例如，0,1,2,⋯ A1,A2⋯）。也可 8 個 1 組，使用暫存器名稱。例如，22,23,24,25, 26,27,28,29 這 8 隻腳暫存器名稱爲 PORTA，亦可以使用暫存器名稱 PORTA。其餘 PORT 名稱，如下表所示：

暫存器\位元	7	6	5	4	3	2	1	0
PORTA	29	28	27	26	25	24	23	22
PORTB	13	12	11	10	50	51	52	53
PORTC	30	31	32	33	34	35	36	37
PORTD	38				18	19	20	21
PORTE			3	2	5		1	0
PORTF	A7	A6	A5	A4	A3	A2	A1	A0
PORTG			4			39	40	41
PORTH		9	8	7	6		16	17
PORTI								
PORTJ						15	14	
PORTK	A15	A14	A13	A12	A11	A10	A9	A8
PORTL	42	43	44	45	46	47	48	49

以上這些接腳都可以使用軟體的設定，指派腳位功能。其次，腳位功能有三種，分別是數位輸出、數位輸入、或有上拉電阻 INPUT_PULL UP 的輸入等 3 種功能，分別說明如下：

指派腳位功能

Arduino 指派腳位功能有兩種方式，分別是單隻腳位的 pinMode 指派與 8 隻腳一起指派的 DDRA 指令（Data Direction of Port A 的縮寫）。例如，以下程式可使用 pinMode 指派腳位 13 作爲數位輸出。

```
pinMode(13, OUTPUT);
```

以下程式可指派腳位 29,28,27,26,25,24,23,22 作爲數位輸出。

```
    pinMode(29, OUTPUT);
    pinMode(28, OUTPUT);
    pinMode(27, OUTPUT);
    pinMode(26, OUTPUT);
    pinMode(25, OUTPUT);
    pinMode(24, OUTPUT);
    pinMode(23, OUTPUT);
    pinMode(22, OUTPUT);
```

因爲以上腳位 29 ～ 22 剛好是 PORTA 暫存器，所以以上程式亦可簡化爲

```
DDRA=B11111111;// 1是輸出，指派PORTA爲輸出
```

B 表示後續數字代表二進位，1 表示輸出，0 表示輸入。同理，若是

```
DDRA=B00000000;// 0是輸入，指派PORTA爲輸入
```

則指派 PORTA 爲輸入。也就是 pinMode 是配合腳位名稱，一次指派一個腳位的功能；DDRA 是配合暫存器名稱，一次指派 8 個腳位的功能，同理 PORTB、PORTC 就用 DDRB、DDRC 等指派其功能。

數位輸出

Arduino 腳位若當作數位輸出，使用手冊的說明如下：

> Properties of Pins Configured as OUTPUT
>
> Pins configured as OUTPUT with pinMode() are said to be in a low-impedance state. This means that they can provide a substantial amount of current to other circuits. Atmega pins can source (provide positive current) or sink (provide negative current) up to 40 mA (milliamps) of current to other devices/circuits. This is enough current to brightly light up an LED (don't forget the series resistor), or run many sensors, for example, but not enough current to run most relays, solenoids, or motors.
>
> Short circuits on Arduino pins, or attempting to run high current devices from them, can damage or destroy the output transistors in the pin, or damage the entire Atmega chip. Often this will result in a "dead" pin in the microcontroller but the remaining chip will still function adequately. For this reason it is a good idea to connect OUTPUT pins to other devices with 470Ω or 1k resistors, unless maximum current draw from the pins is required for a particular application.

大意是說，當輸出時，若設定爲 HIGH，則電壓有 5V（請留意有些微控板是 3.3V），且每隻腳位高電位的最大輸出電流是 20mA；設定爲 LOW 時，可承受或稱流入 40mA 的電流。其次，因爲 LED 只要 10mA(0.01A) 就很亮，所以直接使用高電位驅動 LED 可說綽綽有餘，但是驅動電流若太大，LED 也會燒毀，所以要加上限流電阻如下：

$$\frac{5-1.7}{0.01} = 330\,\Omega$$

以上 1.7(V) 稱爲 LED 的順向切入電壓，10mA=0.01A，這樣計算時，單位才一致。

指派電位

單晶片的優點是您可直接下指令，指派任何接腳爲 HIGH 或 LOW。指派的方式有兩種，分別是單隻腳位的 digitalWrite 指派與八位元的暫存器名稱 (如 PORTA) 整體指派。例如，以下程式，您可指派接腳 22 爲 HIGH。

```
digitalWrite(22, HIGH);
```

以下程式，您可指派其爲 LOW。

```
digitalWrite(22, LOW);
```

所有腳位都有一個暫存器名稱，還可使用暫存器名稱一起指派 8 隻腳的電位，例如，以下程式可快速指派 PORTA 的 8 個位元全爲 HIGH，PORTA 是暫存器名稱。

```
PORTA=B11111111;
```

以下程式可快速指派 PORTF 的 8 個位元輸出全爲 LOW。

```
PORTF=0;//0就0，當然不用再指派任何進位。
```

範例 1a

PORTA 腳位探索實習。（PORTA 共有 8 隻腳位，本書往後簡稱 PA）

1. 請鍵入以下程式，驗證、上傳。

```
void setup() {
  // put your setup code here, to run once:
    DDRA=B11111111;// 指派PA爲輸出
    PORTA=B11111111;//指派PA爲高電位
}void loop() {}//此函式雖然沒用到，但不能刪除
```

2. 請用三用電表，檢驗微控板的腳位 22, 23, 24, 25, 26, 27, 28, 29 是否爲高電位 5V。
3. 若沒三用電表，請依照附錄 A，焊接一個 LED 電位筆，然後負端先插入 Gnd，正端就可當作電位探測筆，LED 亮就代表有電壓 5V。
4. 鍵入以下程式，重新驗證、上傳，再檢驗微控板的腳位 22, 23, 24, 25, 26, 27, 28, 29 是否爲低電位 0V。

```
void setup() {
    DDRA=B11111111;// 指派PORTA爲輸出
    PORTA=0;//指派PORTA輸出低電位
}void loop() {}
```

自我練習

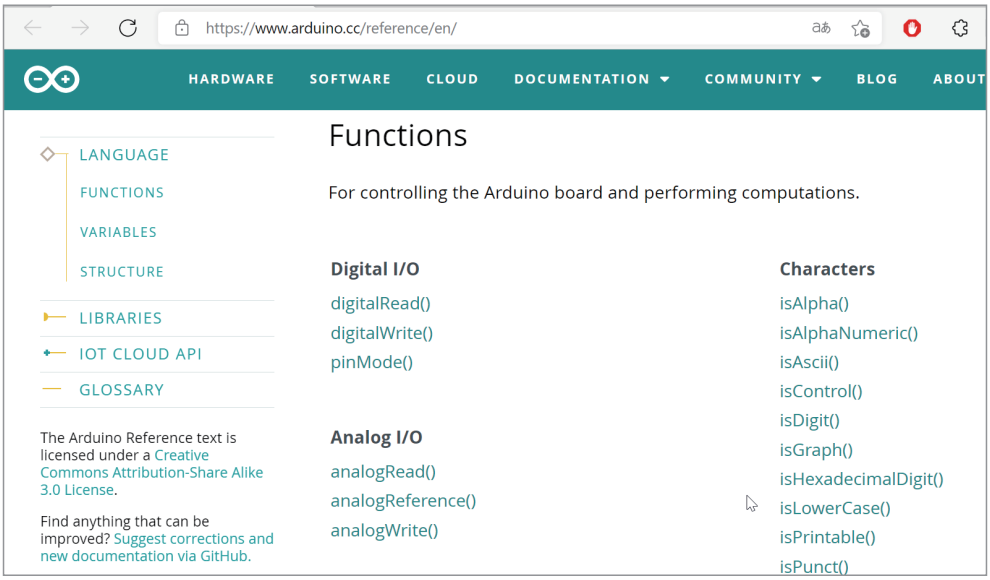
1. 請鍵入以下程式，並驗證 PORTA、PORTB、PORTC 對應腳位是否爲高電位。

```
void setup() {
  // put your setup code here, to run once:
    DDRA=B11111111;//指派PA爲輸出
    PORTA=B11111111;//設定PA0～7均爲高電位
    DDRB=B11111111;//指派PB爲輸出
    PORTB=B11111111;//設定PB0～7均爲高電位
    DDRC=B11111111;
    PORTC=B11111111;
} void loop() {}
```

2. 請探索 PORTF、PORTK、PORTL 腳位在哪裡？並寫程式檢驗這些腳位電壓是否能隨著您的程式而變化。

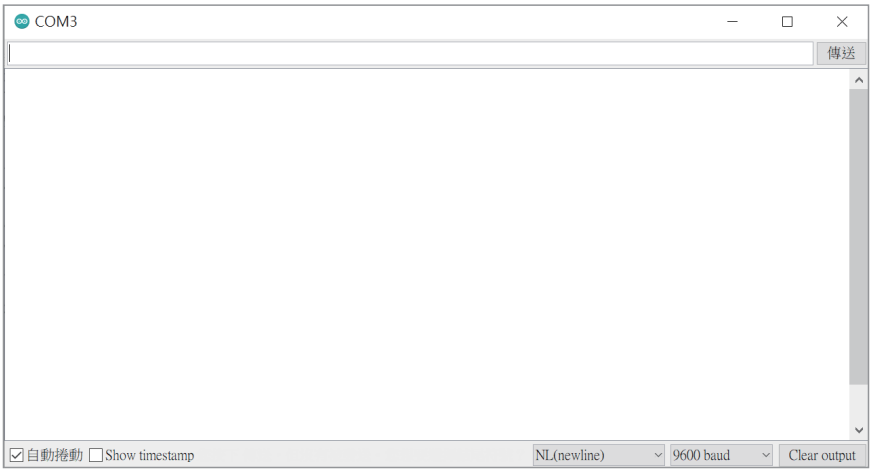
線上使用手冊

點選 Arduino IDE 功能表的『說明 / 參考文件』畫面如下圖（電腦請先連線），這就是 Arduino 所提供的軟體指令手冊，不論指令分類、指令解說、範例等都很詳細，請讀者自行探索。



程式的輸出入與序列埠監控視窗

人類號稱萬物之靈，眼睛、鼻子等五官與手腳是人類的輸出入設備，所有程式語言都是幫助人類處理事情的工具，當然也有輸出入設備。輸出入也是所有程式設計的第一步，本單元先介紹 Arduino 的序列埠監控視窗。序列埠監控視窗如下圖所示：(點選『工具』/『序列埠視窗』)



它可以讓我們用電腦的鍵盤與螢幕和微控板雙向溝通。此一功能是以往其他單晶片所沒有的功能，因爲透過此序列埠監控視窗，不僅可以學習 Arduino 語言，學習 Arduino 所能完成的所有程式設計工作。例如，以下程式可以學習 Arduino 的運算子與數學函式，因爲程式執行後，可在序列埠視窗出現執行結果。

```
void setup() {
    Serial.begin(9600);
    Serial.println(4+2);
    Serial.println(4>=2);
    Serial.println(pow(2,3));
} void loop() {}
```

其次，於開發 Arduino 程式中，從感測器所讀到的值、或運算結果要從輸出元件輸出前，都可將結果先行輸出，先行測試此段程式邏輯與硬體接線是否正確。例如，以下程式，可以將指撥開關的輸入值，先行輸出於電腦螢幕的序列埠視窗，這樣可以確認此部分硬體接線是否正確。

```
a=digitalRead(37);//讀取腳位37的電壓值
Serial.println(a);//於序列埠視窗輸出電壓值
```

以下程式，可以先將運算結果先行輸出，這樣可以檢查此軟體運算是否正確，然後再輸出於 PORTF。

```
b=a+3//處理
Serial.println(b);//於序列埠視窗輸出電壓值
PORTF=b;
```

Serial物件

要讓微控板與電腦溝通，那就要透過 Serial 物件，請開啓 Arduino 參考文件 (點選『說明』/『參考文件』/『Serial』)。Serial 的 Function 如下圖所示。這些方法可以讓我們啓用序列

埠，然後進行輸出文數字、輸入字元、輸入字串與數值，分別說明如下。(補充說明：以下這些函式，在舊式函式導向程式設計稱爲『函式』，但在目前物件導向的程式設計領域，函式已經改稱爲『方法』)

Functions

If (Serial)
available()
availableForWrite()
begin()
end()
find()
findUntil()
flush()
parseFloat()
parseInt()
peek()
print()
println()
read()
readBytes()
readBytesUntil()
setTimeout()
write()
serialEvent()

啓用序列埠

要使用序列埠，首先要使用 begin() 方法啓用序列埠，先規定雙方傳輸速度，本例規定 9600，程式如下：

```
Serial.begin(9600);
```

然後請點選『工具 / 序列埠監控視窗』進入『序列埠監控視窗』點選相同的速度，如下圖：

輸出

要在序列埠輸出文數字，要用 print() 或 println() 方法，兩者的差別是 println() 輸出後換行歸位，print() 則沒有。例如，以下程式，可輸出『GwoshengGwosheng』，如下右圖。

`Serial.print("Gwosheng");` `Serial.print("Gwosheng");`	GwoshengGwosheng

以下程式可輸出兩列 Gwosheng，如下圖右。

`Serial.println("Gwosheng");` `Serial.println("Gwosheng");`	Gwosheng Gwosheng

若是 print 或 println 方法內接變數，則會轉印出此變數的內容。例如，以下程式，可將變數所代表的內容輸出至電腦螢幕。

```
int a=3;//規定資料的儲存空間，宣告a爲整數變數
String b="Gwosheng";//規定資料的儲存空間，宣告b爲字串變數
Serial.println(a);//3
Serial.println(b);//Gwosheng
```

以下程式可輸出『a=3』。有加雙引號『""』的是字串，沒加雙引號的是變數，遇到字串就直接輸出，遇到變數就往前找此變數所儲存的值，並輸出。

```
int a=3;//規定資料的儲存空間，宣告a爲整數變數
Serial.print("a=");//字串就直接輸出a=
Serial.print(a);//變數，輸出變數的值3
```

範例 1b

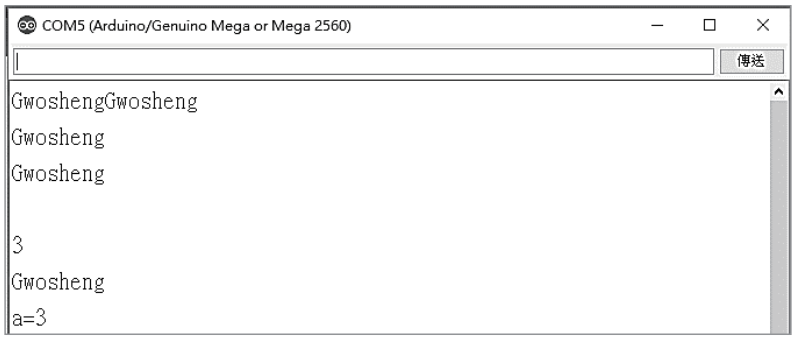
示範以上程式。

操作步驟

本例只要插入 Arduino 微控板就好，不用準備任何電路。

執行結果

程式『驗證』、『上傳』後，請開啓『序列埠監控視窗』（功能表的『工具 / 序列埠監控視窗』）。



程式列印

雙斜線『//』稱爲註解，註解後的文字，僅給人看，電腦不予編譯，初學者可以不用打。

```
void setup() {
    // put your setup code here, to run once:
    Serial.begin(9600);
    Serial.print("Gwosheng");
    Serial.print("Gwosheng");
    Serial.println();//only linefeed
    Serial.println("Gwosheng");
    Serial.println("Gwosheng");
    Serial.println();//only linefeed
    int a=3;//規定資料的儲存空間，請看第十單元
    String b="Gwosheng";//規定資料的儲存空間，請看第十單元
    Serial.println(a);
    Serial.println(b);
    Serial.print("a=");
    Serial.print(a);
}void loop() {}//此函式雖然沒有放程式，但也不能刪除
```

自我練習

1. 請於序列埠監控視窗輸出自己的座號與名字，本例輸出座號後換行歸位，再輸出姓名。
2. 同上題，請在同一列輸出自己座號與名字。

■ ※ 輸入（不常用，教學時數少時，可先跳過）

Arduino 可以分別使用 read()、readString() 及 parseInt() 等方法讀取使用者於鍵盤所輸入字元、字串與數字，分別說明如下：

字元輸入

要讓序列埠輸入字元（僅一個英文字母或數字視爲字元），也是要同上範例，先使用 begin() 啓動序列埠，再使用 read() 方法讀入字元，程式如下：

```
char c=Serial.read();//宣告c 爲char型態變數，請看第十單元
```

available() 方法

大部分程式語言的鍵盤輸入功能，遇到輸入指令，都會停留在此指令，癡癡的等待使用者輸入資料，直到使用者按『Eenter』，再繼續往下執行。但是 Arduino 有特殊原因，它必須兼顧很多輸入設備，所以不會也不能等待任何輸入設備。若特別需要它一定要原地等待使用者鍵盤輸入，那就要使用 available() 方法，因爲 available() 方法可判斷使用者是否已經輸入，available() 的語法如下：

```
Serial.available()
```

1. 也就是 available() 傳回大於等於『1』，才表示使用者有輸入，例如，以下程式，會請單晶原地等待您輸入，若鍵盤緩衝區沒有資料，就重複迴圈，也就是原地癡癡的等待。while 稱爲迴圈指令，請看第 8 單元

```
void setup() {
    Serial.begin(9600);
}
void loop() {
    while(Serial.available() ==0) {}//若未輸入，則一直在此等待。
    char c=Serial.read();
    Serial.println(c);
}
```

2. 請輸入以下程式（沒有使用 available() 函式），並觀察執行結果。

```
void setup() {
    Serial.begin(9600);
}
void loop() {
    char c;
    c=Serial.read();
    Serial.println(c);
}
```

3. 以下左右程式也不同。請自行鍵入、執行，並觀察結果。下圖左是正確的，while 迴圈有加兩個 {}；下圖右是錯誤的，c =Serial.read() 會屬於 while 迴圈，因爲 while 沒有大括號，那此迴圈會包含與執行一個敘述，這都在第 8 單元詳細介紹。

```
void setup() {
    Serial.begin(9600);
}
void loop() {
    char c;
     while(Serial.available() ==0)
{}
    c =Serial.read();
    Serial.println(c);
    Serial.println("aa");
}
```

```
void setup() {
    Serial.begin(9600);
}
void loop() {
    char c;
    while(Serial.available() ==0)
       c =Serial.read();
    Serial.println(c);
    Serial.println("aa");
}
```

字串輸入

readString() 可讀取字串（一連串的字元稱爲字串），例如，以下程式可讀取一個字串，然後輸出字串。

```
void setup() {
    Serial.begin(9600);
}
void loop() {
    while(Serial.available() ==0) {}
    String c=Serial.readString();
    Serial.println(c);
}
```

自我練習

請輸入以上程式，執行程式，輸入字串，並觀察執行結果。

數字輸入

Mega 2560 還可直接輸入數字，例如，以下程式可輸入一個整數。

```
int a=Serial.parseInt();
```

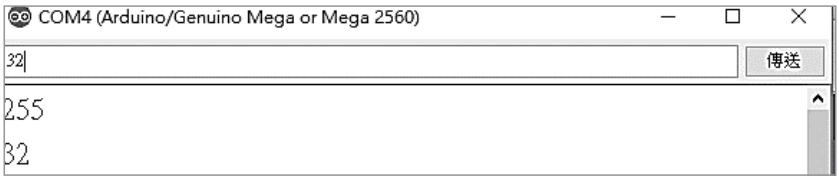
int 是變數的資料型態，表示 a 可儲存 -32768 ～ 32767 的整數，將於下一單元進一步介紹。

範例 1c

示範以上程式。

輸出結果

請開啓序列埠監控視窗（功能表的『工具 / 序列埠監控視窗』）。



程式列印

```
void setup() {
    Serial.begin(9600);
}
void loop() {
    int a;
    while(Serial.available() ==0) {}//程式在此等待使用者輸入
    a=Serial.parseInt();
    Serial.println(a);
}
```

自我練習

1. 有了 Serial 鍵盤與螢幕輸出入方法，那也可把 Arduino 拿來作爲學習 C 語言程式設計的工具，寫出 C 語言能做的程式。例如，請寫一程式，可以輸入兩個整數、計算其和，且輸出結果（提示：每輸入一個整數，就要一個 while 迴圈等待。其次，往後各章還有很多這方面的題目，可用來學習 C 語言）。