Microcomputer Control

Lecture 6

Arduino程式撰寫與實驗 – 序列通訊(一)

Chih-Chiang Chen

ccchenevan@mail.ncku.edu.tw

2021

字串

- 字串 = 資料型態爲字元的一維陣列(簡稱字元陣列)
- 設定字串(即所謂的宣告字串) – 不設定初始值

char 字串(字元陣列)名稱[陣列大小(n)];

- 設定字串(即所謂的宣告字串) – 設定初始值

char 字串(字元陣列)名稱[陣列大小(n)] = 初始值;

其中初始值內容可爲

- 字元集合，並含有結束符號\0

{'s', 'w', 'e', 'e', 't', ' ', 'h', 'o', 'm', 'e', '\0'}　(此時陣列大小n需填入11或空格不填)

- 直接設定字串，並前後加上雙引號

"sweet home"　(此時陣列大小n需填入11或空格不填)

並列(平行)與序列(串列)通訊

- 微電腦和周邊裝置之間的連接有「並聯」和「串聯」兩種方式，分別稱爲「並列(平行)」介面和「序列(串列)」介面。
- 「介面」$\Longleftrightarrow$「橋梁、通道、路徑」
- 「並列」通訊介面可使得微電腦和周邊裝置一次接收/傳送8位元之資料。
  - 傳輸速度快
  - 線材成本高、施工費用高、佔用空間大
  - 個人電腦顯示卡PCI介面即爲並列介面
  - 容易受到雜訊干擾(線多)，不適合長距離傳輸
- 「序列」通訊介面可使得微電腦和周邊裝置一次接收/傳送1位元之資料。
  - 傳輸速度較慢
  - 線材成本低、佔用空間小
  - 個人電腦USB、HDMI/DVI顯示器介面、藍芽無線介面都爲序列介面
  - 較不容易受雜訊干擾(線少)
- Arduino控制板和電腦之間傳輸訊息主要是透過USB介面(USB訊號)(序列介面)，並執行序列(串列)埠通訊軟體/程式來交換訊息。

電腦 $\Longleftrightarrow$ ATmega16U2晶片 $\Longleftrightarrow$ Arduino控制板

USB訊號 (早期爲RS232訊號)　　TTL(Transistor-Transistor Logic)訊號

從Arduino控制板傳遞序(串)列訊息給電腦

- 透過啓用(編譯程式並燒錄)Arduino控制板上執行Serial通訊程式，並在電腦端啓用(直接打開)Arduino整合開發環境內建序(串)列通訊軟體(即序列傳輸埠監控視窗)，就可讓Arduino控制板與電腦能互相傳序列訊息。
- 啓用Arduino控制板之Serial通訊程式，並設定其資料傳輸速率(鮑率(Baud rate))

$$\boxed{\text{Serial.begin}(x);}$$

其中x代表鮑率(Baud rate)，即每秒傳送x個位元，其資料型態爲整數。一般將鮑率設定成9600(每個軟/硬體的可用鮑率有所不同)。

Arduino控制板 $\Longleftrightarrow$ Arduino整合開發環境內建序列通訊軟體(電腦端)

(也要設定鮑率 $=$ 9600) (預設鮑率 $=$ 9600)

從Arduino控制板傳遞序(串)列訊息給電腦

- 當啓用了Arduino控制板之Serial通訊程式，我們可透過下列函數來設定Arduino控制板傳送訊息給電腦：

```
Serial.print(information);
```

其中information之資料型態爲字元、整數或浮點數。當information被填入字串時且直接設定字串內容時，需在填入之內容前後加上雙引號；當information被填入字元時且直接設定字元內容時，需在填入之內容前後加上單引號。此外，當information爲浮點數時，可在information後面加上“, x”用以表示希望傳送小數點幾位(若不填，則預設爲2位)

- 特殊字串符號(須被涵在雙引號內)：
  - \n：換行
  - \t：定位(相當於按一下電腦的tab)
- 等效函數：(小數點設定方法仍可用)

```
Serial.println(information);
```

||

```
Serial.print(information);
Serial.print("\n");
```

實驗

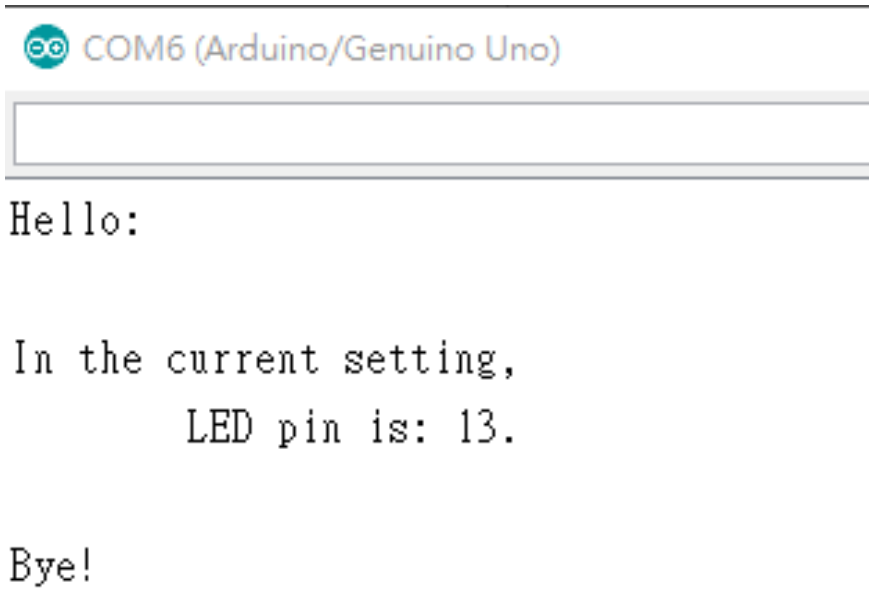
- **Source Code:**

```
(1)   const byte ledPin = 13;
(2)   char str1[ ] = “Hello”;
(3)   char str2 = ‘:’;
(4)
(5)   void setup() {
(6)     Serial.begin(9600);
(7)     Serial.print(str1);
(8)     Serial.println(str2);
(9)     Serial.println();
(10)    Serial.print(“In the current setting,\n”);
(11)    Serial.print(“\tLED pin is: ”);
(12)    Serial.print(ledPin);
(13)    Serial.print(‘.’);
(14)    Serial.println();
(15)    Serial.print(“\nBye!”);
(16)  }
(17)
(18)  void loop() {
(19)  }
```

實驗

- **Principle:**
  1. 程式碼(1): 設定ledPin爲常數byte資料類型態並令爲13
  2. 程式碼(2): 設定str1爲char資料類型態並令爲(字串)Hello
  3. 程式碼(3): 設定str2爲char資料類型態並令爲(字元):
  4. 程式碼(5): 開始setup函數
  5. 程式碼(6): 透過Serial.begin函數啓用Arduino控制板之Serial通訊程式，並設定其資料傳輸速率爲9600
  6. 程式碼(7): 透過Serial.print函數設定Arduino控制板傳送訊息str1給電腦，其中str1爲字串
  7. 程式碼(8): 透過Serial.println函數設定Arduino控制板傳送訊息str2給電腦，其中str2爲字元
  8. 程式碼(9): 透過Serial.println函數設定Arduino控制板傳送訊息(空白)給電腦
  9. 程式碼(10): 透過Serial.print函數設定Arduino控制板傳送(字串)訊息In the current setting, \n給電腦
  10. 程式碼(11): 透過Serial.print函數設定Arduino控制板傳送(字串)訊息\tLED pin is:給電腦
  11. 程式碼(12): 透過Serial.print函數設定Arduino控制板傳送訊息ledPin給電腦
  12. 程式碼(13): 透過Serial.print函數設定Arduino控制板傳送(字元)訊息.給電腦
  13. 程式碼(14): 透過Serial.println函數設定Arduino控制板傳送訊息(空白)給電腦
  14. 程式碼(15): 透過Serial.print函數設定Arduino控制板傳送(字串)訊息\nBye!給電腦
  15. 程式碼(16): 結束setup函數
  16. 程式碼(18): 開始loop函數
  17. 程式碼(19): 結束loop函數

實驗


COM6 (Arduino/Genuino Uno)
Hello:
In the current setting,
LED pin is: 13.
Bye!