**微處理機實習報告**

* 指導老師：洪志宗
* 姓名：許展銘
* 班級：五專資訊四真
* 學期：109上

**題目**

* 使用Arduino UNO 的串列通訊（Arduino IDE 的序列埠監看視窗）來輸入指定的值，來控制LED燈的亮法



* 動作：當在序列埠監看視窗輸入：

“0”： 全部LED燈直接亮暗正向反向各一次

“1”： 第一個LED燈(Pin 3)亮一次

“2”： 第二個LED燈(Pin 5)亮一次

“3”： 第三個LED燈(Pin 6)亮一次

“4”： 第四個LED燈(Pin 9)亮一次

“5”： 全部的LED都用PWM控制且依序亮起

**所需材料**

|  |  |
| --- | --- |
| 材料名 | 數量 |
| Arduino UNO | 1個 |
| LED燈 | 4個 |
| 電阻(220Ω) | 4個 |
| 杜邦線 | 6條 |
| 麵包版 | 1個 |
| USB 2.0 Type A轉B線 | 1條 |

**程式碼**

**int LED[]={3,5,6,9}; //正向:用陣列定義LED燈的接腳(共4支腳)**

**int LED\_x[]={9,6,5,3}; //反向:用陣列定義LED燈的接腳(共4支腳)**

**int array\_num; //使用迴圈時,定義陣列的位置**

**int intput\_num=0; //預定窗口要輸入值**

**//PWM類比輸出(0 ==> 127 ==> 255)**

**void led\_pwm(int kx){**

**analogWrite(kx,0); //PWM值=0(0%)**

**delay(300); //delay 300ms**

**analogWrite(kx,127); //PWM值=127(50%)**

**delay(300); //delay 300ms**

**analogWrite(kx,255); //PWM值=255(100%)**

**delay(300); //delay 300ms**

**}**

**//亮指定的LED燈**

**void led\_view(int kv){**

**digitalWrite(kv,LOW); //LED燈亮(LOW)**

**delay(100); //delay 500ms**

**digitalWrite(kv,HIGH); //LED燈暗(HIGH)**

**delay(100); //delay 500ms**

**}**

**//LED燈全関**

**void led\_clear(int kc){**

**digitalWrite(kc,HIGH); //LED燈全関**

**}**

**//接腳&開始的設定**

**void setup() {**

**for(array\_num =0;array\_num<=4;array\_num++){**

**pinMode(LED[array\_num],OUTPUT); //把所有的LED燈全部設輸出**

**led\_clear(LED[array\_num]); //把LED燈全部関掉(HIGH)**

**}**

**Serial.begin(9600); //啟用串列通訊的鮑率:9600bps**

**}**

**//要做的動作**

**void loop() {**

**if(Serial.available()>0){**

**intput\_num = Serial.read(); //定義從窗口輸入的值**

**if(intput\_num=='0'){**

**Serial.write("Output 0 LED All ON \n");**

**//LED燈亮的位置順序:1,2,3,4**

**for ( array\_num =0 ; array\_num<4 ; array\_num++){**

**led\_view(LED[array\_num]);**

**delay(300); }**

**//LED燈亮的位置順序:4,3,2,1**

**for ( array\_num =0 ; array\_num < 4 ; array\_num++){**

**led\_view(LED\_x[array\_num]);**

**delay(300);**

**}**

**}else if(intput\_num == '1'){**

**Serial.write("Output 1 : LED 1 ON \n");**

**led\_view(LED[0]); //第一個LED燈亮**

**}else if(intput\_num == '2'){**

**Serial.write("Output 2 : LED 2 ON\n");**

**led\_view(LED[1]); //第二個LED燈亮**

**}else if(intput\_num == '3'){**

**Serial.write("Output 3 : LED 3 ON\n");**

**led\_view(LED[2]); //第三個LED燈亮**

**}else if(intput\_num == '4'){**

**Serial.write("Output 4 : LED 4 ON\n");**

**led\_view(LED[3]); //第四個LED燈亮**

**}else if(intput\_num == '5'){**

**Serial.write("Output 5 : ALL LED Using PWM Control\n");**

**for(array\_num =0;array\_num < 4;array\_num++){**

**led\_pwm(LED[array\_num]); //全部的LED都用PWM控制且依序亮起**

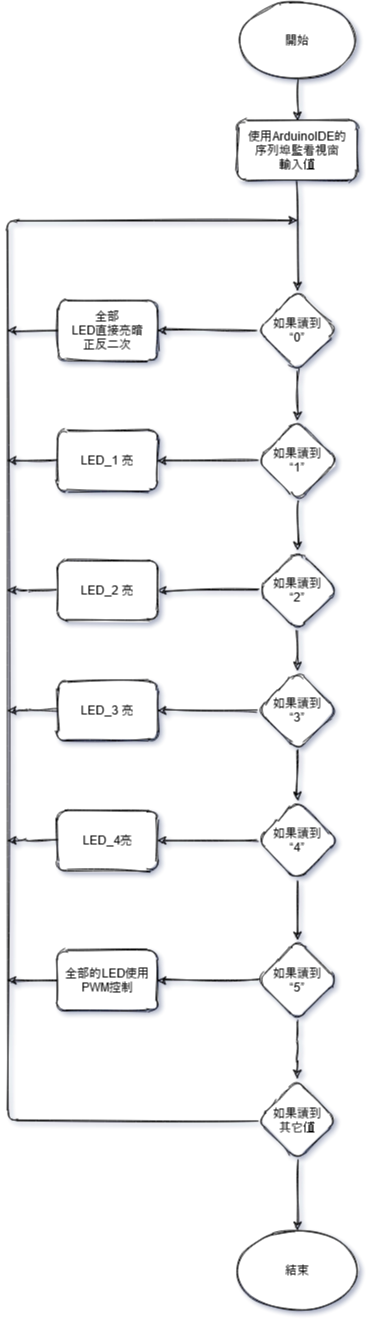
**}**

**}**

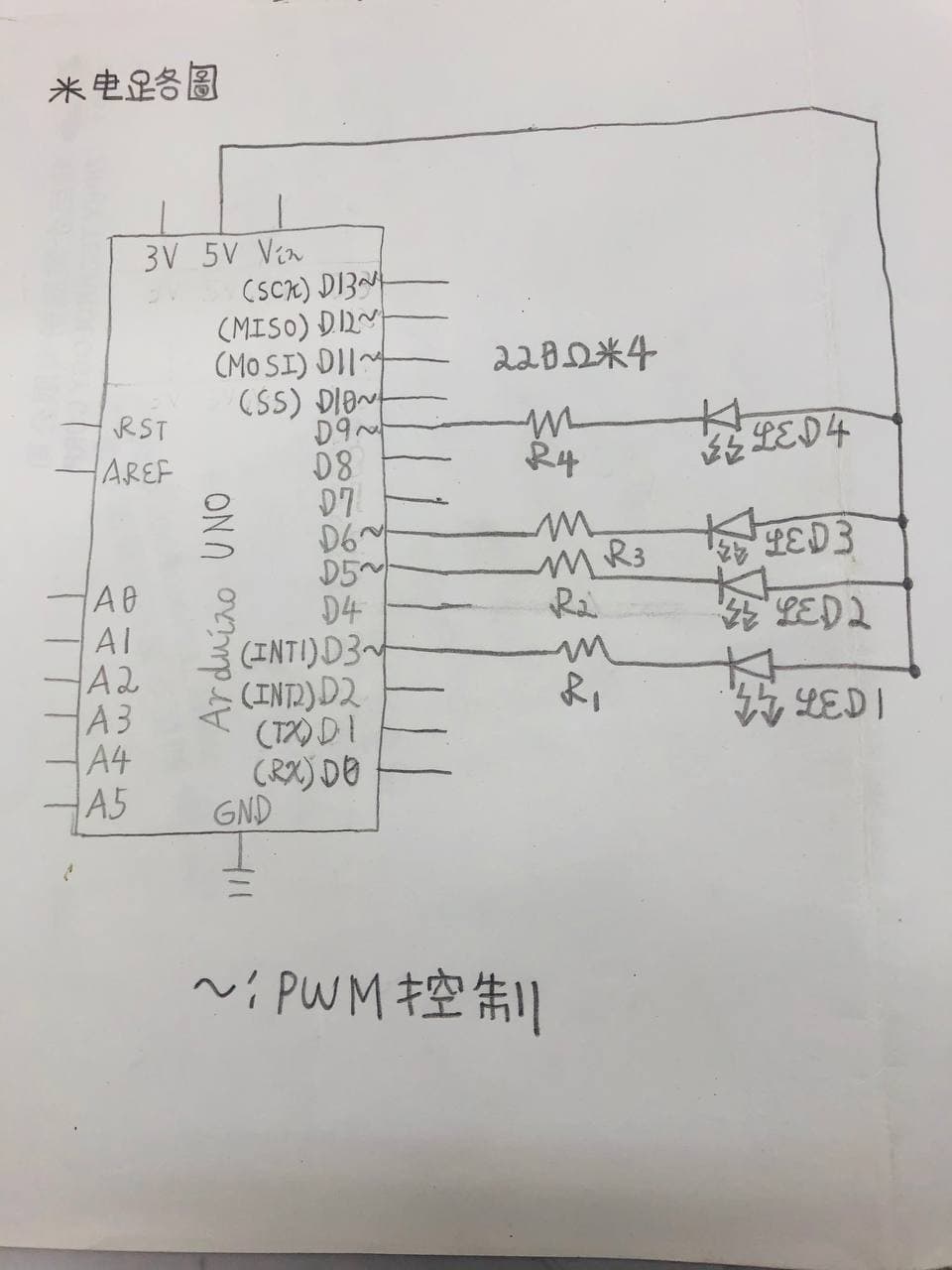
**}**

**}**

**流程圖**

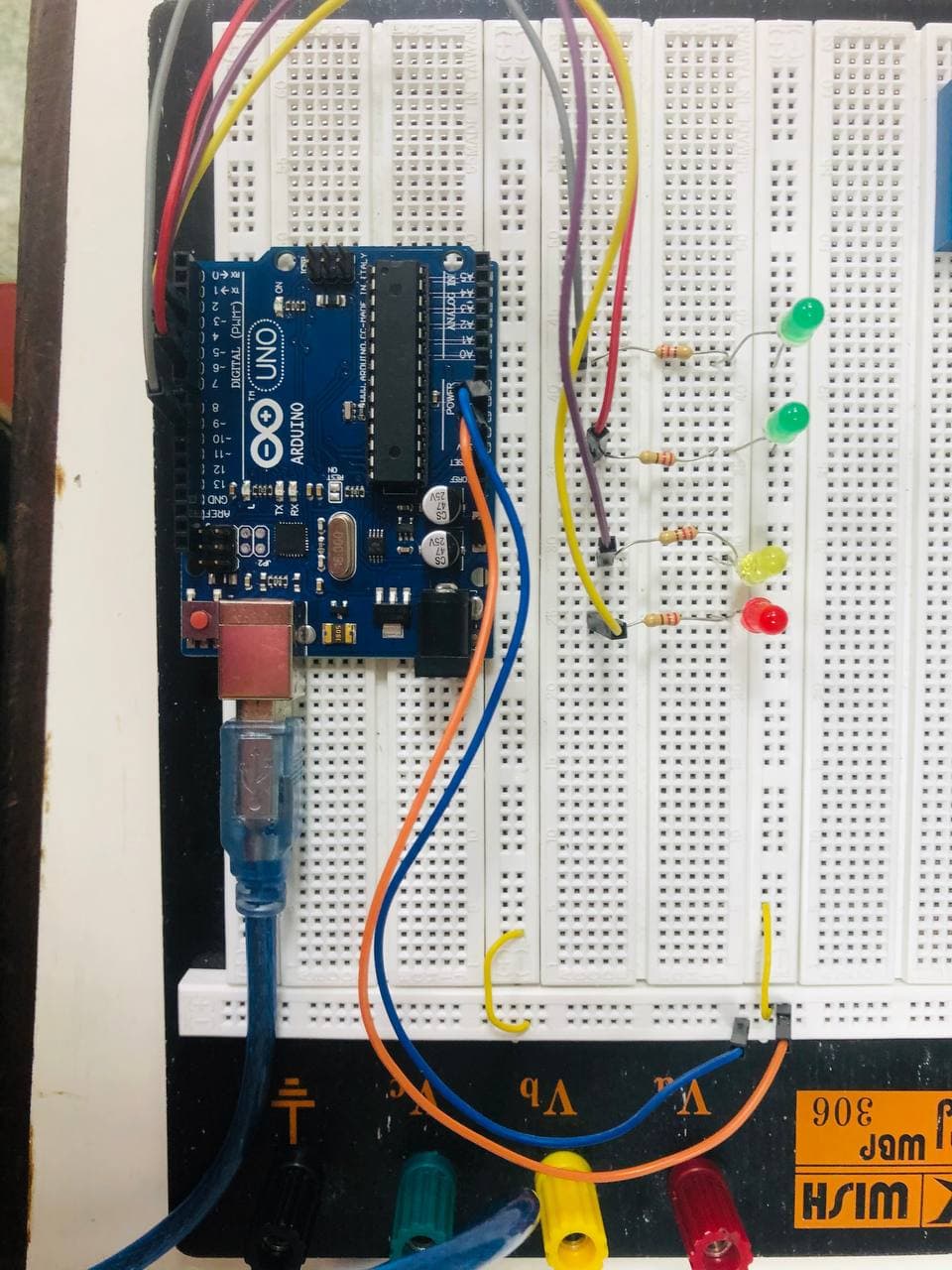


**電路圖**

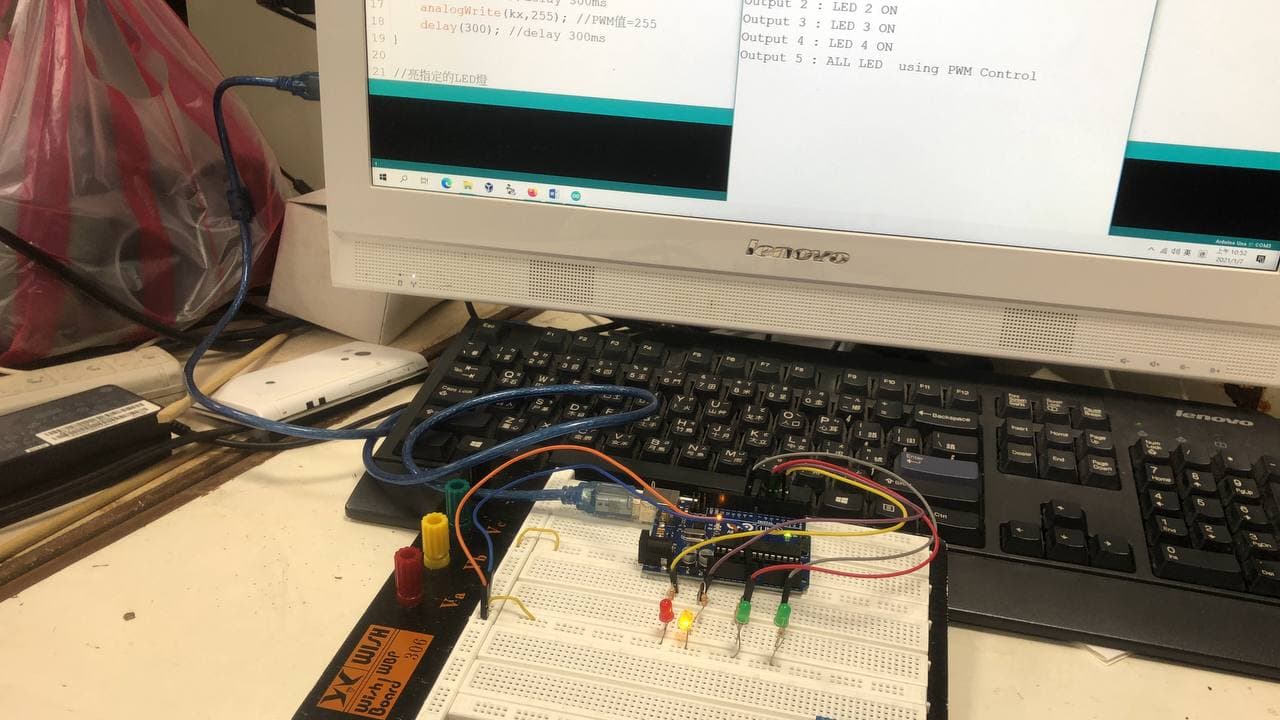


**實作過程**

* 手動接線



* 接線測試



\*實作影片: 微處理機實習-期未報告（用序列埠視窗來控制LED亮法）

https://youtu.be/-Dn1OWep-zY

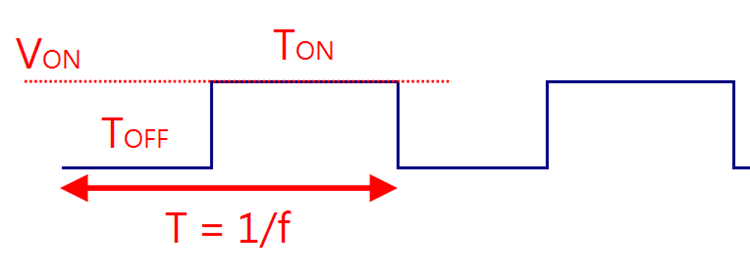
**使用原理**

* PWM（**Pulse-width modulation,**脈波寬度調變**)**

\* 是類比訊號轉換為脈波的一種技術

\* 通常我們可以用來調整燈光的亮度、馬達的轉速、RGB LED 的配色、螢幕亮度控制、喇叭的大小聲/聲音頻率等…

\* 一般轉換後脈波的週期固定，但脈波的工作週期**(Duty Cycle)**會依類比訊號的大小而改變



\* 切換週期 T，通常我們會用頻率 f (Hz) 來表達

\* Duty Cycle = Ton/Toff (%)，為 ON 的時間與 OFF相除的百分比

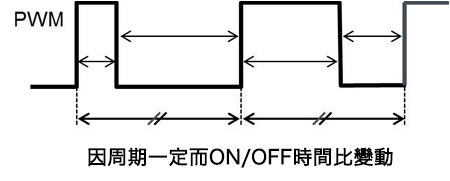
\*模擬出的電壓 V = Von \* duty cycle (%)

🡺可以知道 duty cycle 越高模擬出的電壓越高，當完全沒有 OFF 的時候，duty cycle = 100%

\*V = Von 為最大可輸出的電壓，這時候電燈會最亮

\*優點: 由於頻率一定，故有可預測即將產生之開關雜訊、濾波處理容易等

\*缺點:由於頻率一定，重負載時和輕負載時之開關次數皆相同，自我消耗電流不變，故輕負載時其開關損失受到控制，效率降低



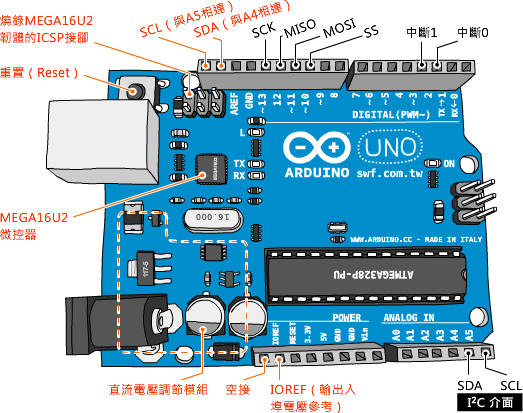
**\*** 數值：**0**～**255**（**8**位元範圍）🡺 所對應電壓**0**～**5V**

\* PWM的值大小對應：

|  |  |  |
| --- | --- | --- |
| 數值 | 寬度 | LED亮度 |
| 大 | 寬 | 暗 |
| 小 | 窄 | 亮 |

\* Arduino UNO可支持的PWM輸出接腳：[3,5,6,9,10,11,13]

\* Arduino 上的“**～**”符號 ＝＝》 即腳位支援PWM輸出

****

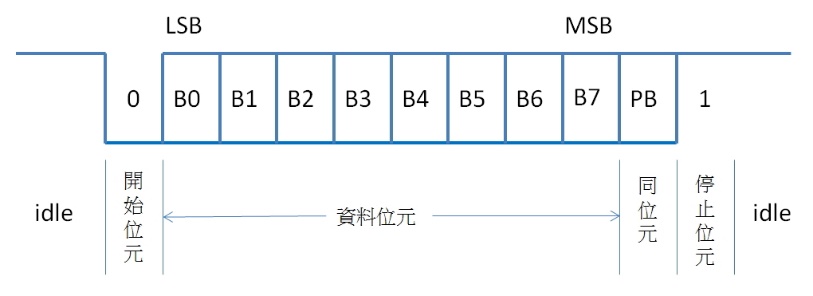
* 串列通訊(Serial communication)

\*串列通訊指的是兩交換信息者的資料流動,而其資料流動的方式是一個位元接著一個位元有順序地由一個方向至另一個方向流動

\*串列埠的通訊方式:一個位元組被拆成一個接著一個的位元而傳送出去，接到此電位訊息的一方，再將此一個一個的位元組合成原來的位元組，如此形成一個位元組的完整傳送

\* LSB(Least Significant Bit): 最低有效位

MSB(Most Significant Bit): 最高有效位



\* 傳輸資料的格式:

起始位元(固定是1 bit)+傳送字元(8 bit)+同位位元+停止位元(1/1.5/2 bit, 一般為1個位元)

\*同位位元是用來檢查所傳送資料的正確性的一種核對碼，又分成奇同位(Odd)及偶同位(Even Parity)兩種，分別是檢查字元碼中1的數目是奇數或是偶數

\*如果採用不同的資料位元數,同位元檢查,停止位元，則每次傳輸的位元組中的位元數都不相同

※如: 此時最小的傳輸單元是以10位元為單位且採用19200bps的傳輸速度，每一秒便可傳輸19200/10=1920(Bytes)的資料

\*串列通訊埠所傳送的資料是字元形態，工業界使用的有ASCII字元碼及JIS字元碼

|  |  |  |
| --- | --- | --- |
| 字元碼 | 資料位元 | 使用制式的地區 |
| ASCII | 8位元 | 歐美 |
| JIS | 7位元 | 日本 |

\* 鮑率(baud rate): 調變速率,指的是有效數據訊號調變載波的速率，即單位時間內載波調變狀態變化的次數

(1)在Arduino上的鮑率跟電腦(序列埠監控視窗)這邊的鮑率設定必須一樣🡺否則無法互相理解雙方所傳過來的資料且難以溝通

(2) 串列傳輸速率的單位是每秒比特數(bps)，常用的單位還有

每秒千比特數Kbps，每秒兆比特數Mbps。

(3)串列典型的傳輸串列傳輸速率600bps,1200bps,2400bps,4800bps,9600bps，19200bps，38400bps

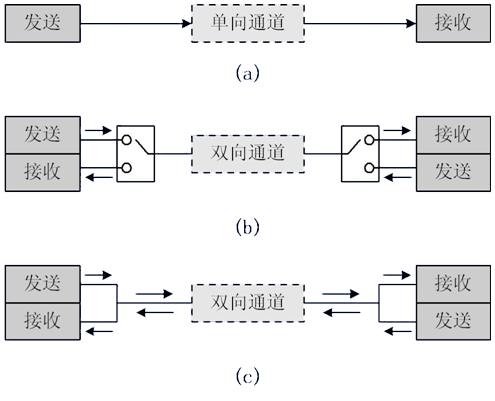
\* 串列埠的工作模式:

(1) 單工（Simplex Communication）: 資料流動只有一個方向

(2) 全雙工（Half Duplex）: 同時具有兩個方向的傳輸能力

(3) 半雙工（Full Duplex）: 資料流動是雙向，但同一時間只能一個方向行進, 實際上是一種切換方向的單工通訊；它不需要獨立的接收端和傳送端，兩者可以合併一起使用一個埠

\*在其它參數都一樣的情況下，全雙工比半雙工傳送速率要快，效率要高



\*串列埠(Serial Port)的通訊在微控制器中是很常用的一種方式，可以用來連接兩個晶片，或是直接跟電腦進行資料交換

\*在Arduino上同樣具備Tx(傳送),Rx(接收)的Serial Port功能，其中一組會建立在板子上(D0[TX]&D1[RX])的USB連接埠上🡺不必另外買RS232-USB轉接線就可以直接跟電腦進行連線

**心得**

上了這堂課後,我在這堂課得知Arduino算一個微處理機且可知道微處理機在実際上可以做出很多事(如:串列通訊,類比輸出),且出的作業讓我們查課本以外的資料(如:RS232,RS422,RS485的差異),

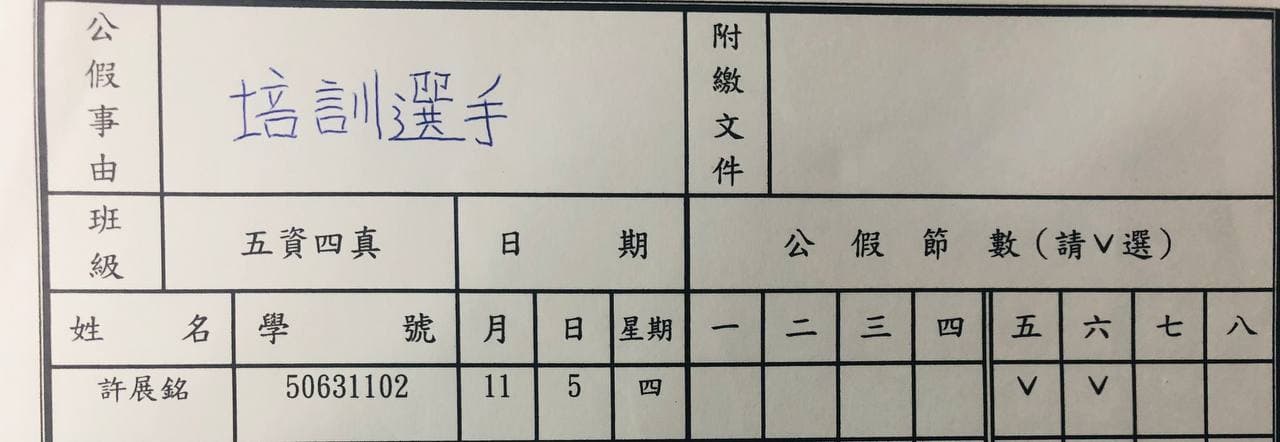
還有老師會講一些課本沒有的東西,

如:

1. 指撥開関要設輸入時可寫[INPUT\_PULLUP],就可使用Arduino內建的提升電阻,來逹到節省外接電阻,且能縮小整個電路面積
2. 如果LED燈只亮一顆,可只用一顆電阻,但要亮不是一顆,全部的LED燈都要有電阻,不然電流和亮度會不穏
3. 係數調整: 5.0/1023.0 🡺 実數輸出, 5/1023 🡺整數輸出

**附錄:公假和加分項目**

* 2020/11/05:請公假



* 2020/11/07:去台大參加[2020第25屆大專校院資訊應用服務創新競賽]比賽



**來源**

* **第一章基本通訊介紹:** [**http://www.ccut.edu.tw/adminSection/ace/downloads/ch01.pdf**](http://www.ccut.edu.tw/adminSection/ace/downloads/ch01.pdf)
* **Arduino 串列埠測試(UART) - 小狐狸事務所:** **http://yhhuang1966.blogspot.com/2015/09/arduino.html**
* **[arduino學堂]傳輸資料吧!序列埠傳輸篇:** [**https://home.gamer.com.tw/creationDetail.php?sn=4687675**](https://home.gamer.com.tw/creationDetail.php?sn=4687675)
* **使用Arduino實作Serial串列通訊的經驗談:** [**https://hanmajor.blogspot.com/2014/08/arduinoserial.html**](https://hanmajor.blogspot.com/2014/08/arduinoserial.html)
* **開關式穩壓器的基本：PWM和PFM:** [**https://techweb.rohm.com.tw/knowledge/dcdc/dcdc\_sr/dcdc\_sr01/897**](https://techweb.rohm.com.tw/knowledge/dcdc/dcdc_sr/dcdc_sr01/897)
* **鮑率- 維基百科，自由的百科全書 – Wikipedia:** [**https://zh.wikipedia.org/wiki/%E6%B3%A2%E7%89%B9%E7%8E%87**](https://zh.wikipedia.org/wiki/%E6%B3%A2%E7%89%B9%E7%8E%87)
* **[Arduino] 脈衝寬度調變 (PWM) 與Arduino -Pulse Width Modulation:**

[**http://thats-worth.blogspot.com/2014/04/arduino-pwm-arduino-pulse-width.html**](http://thats-worth.blogspot.com/2014/04/arduino-pwm-arduino-pulse-width.html)

* **【STM32】串口通信基本原理（超基础、详细版）**

[**http://mcu.eetrend.com/content/2019/100044596.html**](http://mcu.eetrend.com/content/2019/100044596.html)

* **串列通訊基礎知識**

[**https://www.accus.com.tw/msg/msg19.html**](https://www.accus.com.tw/msg/msg19.html)

* **何謂串列通訊**

**http://web.nuu.edu.tw/~tzen/VBITF\_ch08.pdf**