

單晶片控制實習--8051 C語言

CH2 μ Vision3 軟體操作介紹



下載KEIL C51

KEIL Tools by ARM ARM

Products **Download** Events Support Videos

Search Keil.com... Go

Product Information
Product Overview
Supported Microcont
Shows and Semina

Embedded Development Tools

ARM provides complete solutions for microcontroller software development. On this website, you can find

Downloads Request a Quote

KEIL Tools by ARM ARM

Products Download **Events** Support Videos

Search Keil.com... Go

Overview

Keil downloads include products and updates, example programs and various utilities you may use to learn about or extend the capabilities of your Keil development tools.

Product Downloads
Download current and previous versions of the Keil development tools.

File Downloads
Download example programs and various utilities which enable you to extend the capabilities of your Keil development tools.

KEIL Tools by ARM ARM

Products Download Events Support Videos

Search Keil.com... Go

Latest Versions

Download the latest Keil software products.

	MDK-ARM Version 4.71a (May 2013) Development environment for Cortex and ARM devices.
	C51 Version 9.51a (February 2013) Development tools for all 8051 devices
	C166 Version 7.52 (April 2013) Development tools for C166, XC166, & XC2000 MCUs.
	C251 Version 5.54 (February 2013) Development tools for all 8051 devices

KEIL Tools by ARM ARM

Products Download Events Support Videos

Search Keil.com... Go

Product Information
Software & Hardware Products
ARM Development Tools
C166 Development Tools
C51 Development Tools
C251 Development Tools
Debug Adapters
Evaluation Boards
Product Brochures
Newsletters

Device Database®
Device List

Compliance Testing
ISO/ANSI Compliance
1 Verification

Product Downloads

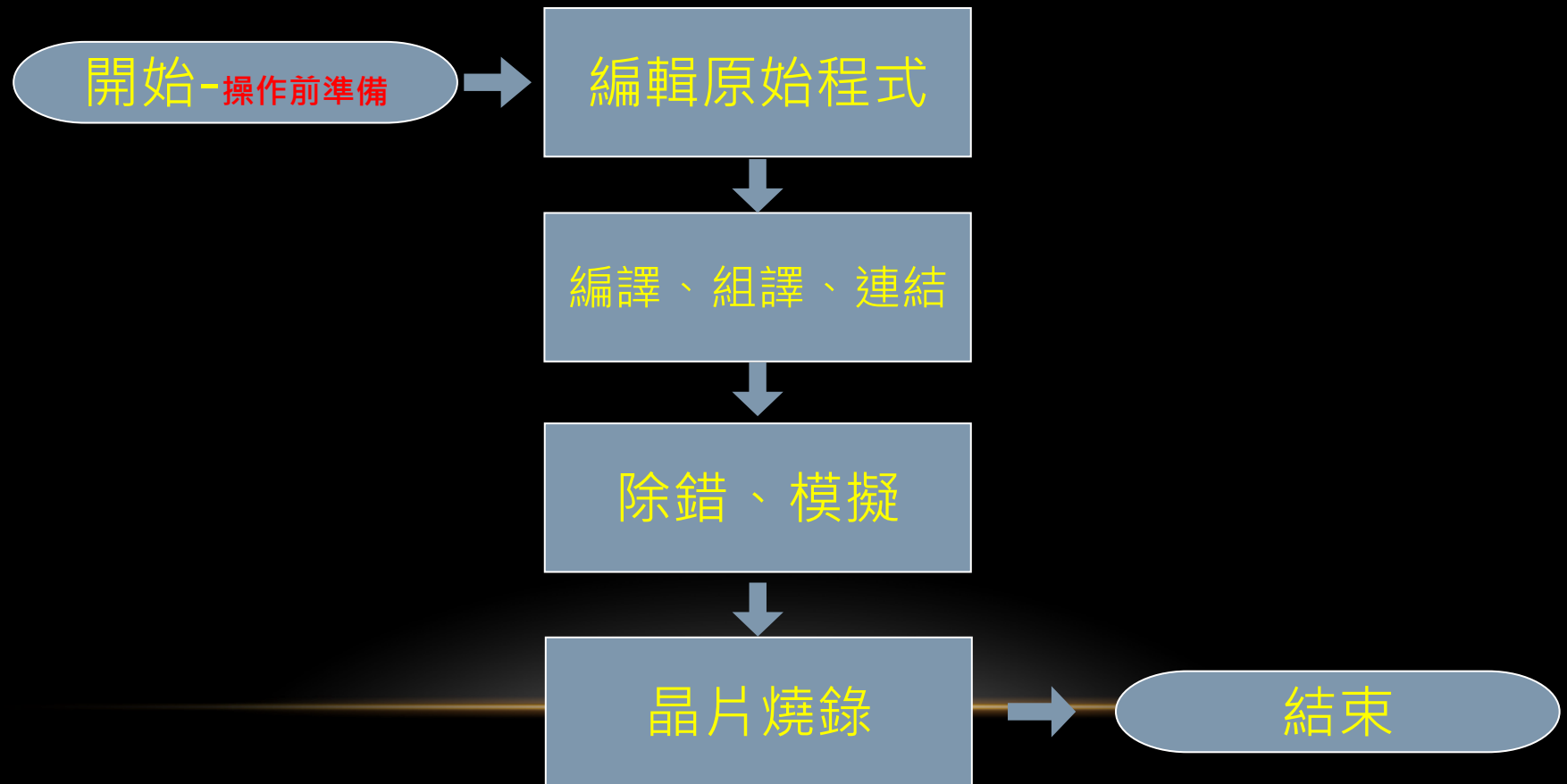
C51 Software
Development tools for Classic and Extended 8051 Microcontrollers
Version 9.51a
Complete the following form to download the Keil software development tools.

Enter Your Contact Information Below

First Name:
Last Name:
E-mail:
Company:
Address:
City:
State/Province:
Postal Code:
Country:

4. 填寫完資料即可下載

操作四大步驟



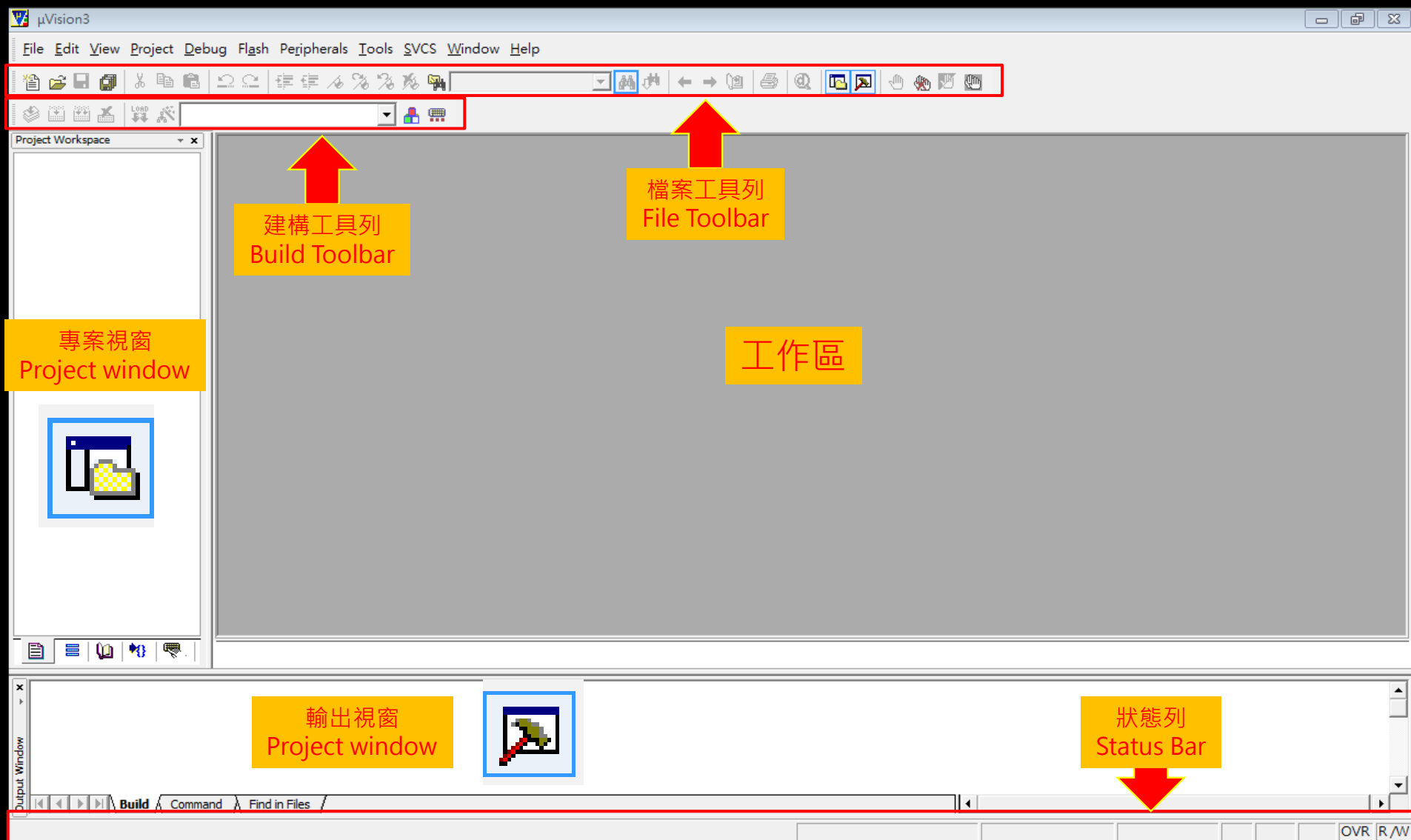
操作前準備-建立個人專案資料夾

- 以英文路徑建立個人專案資料夾

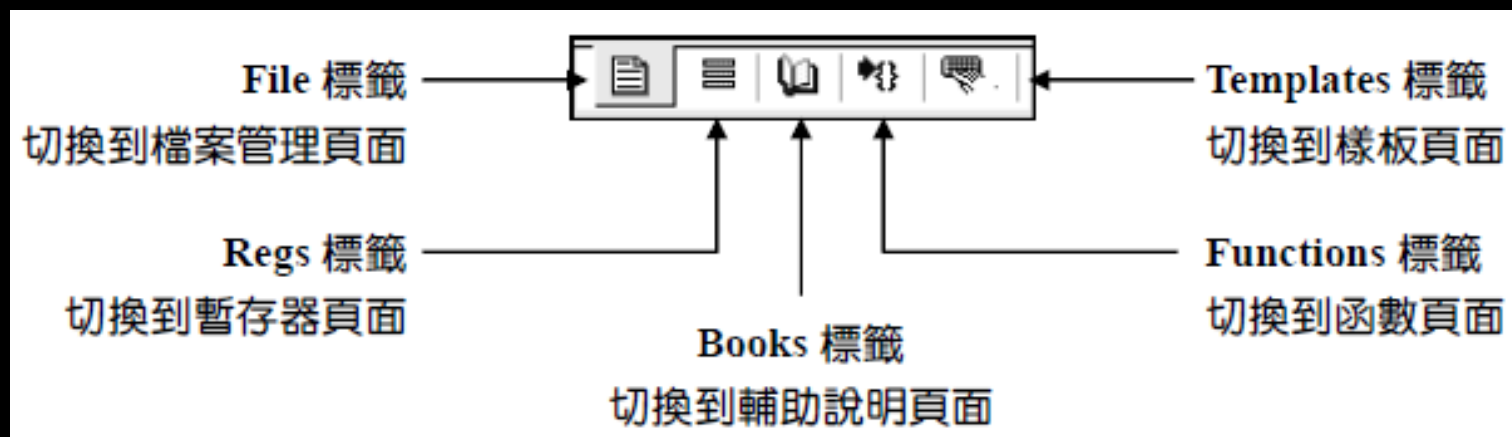
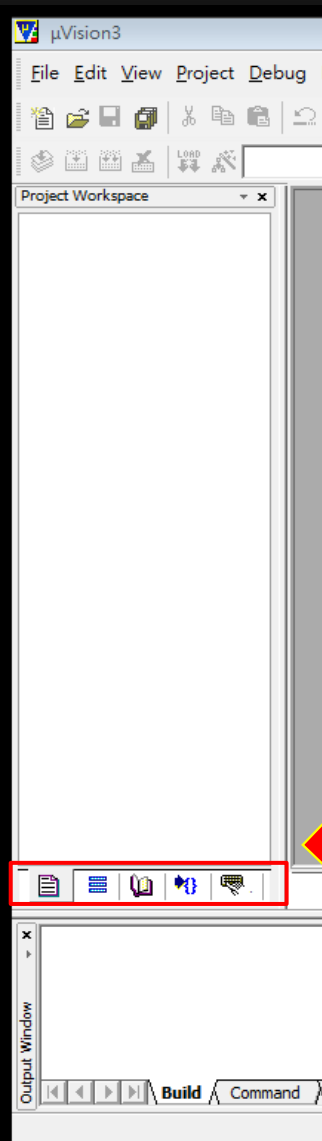
Ex : \8051\CH2\CH2-1\test.c

⋮
⋮

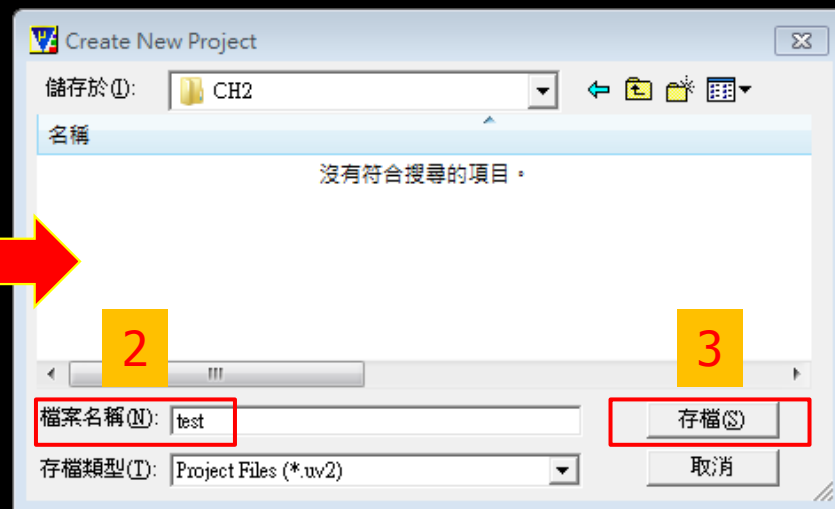
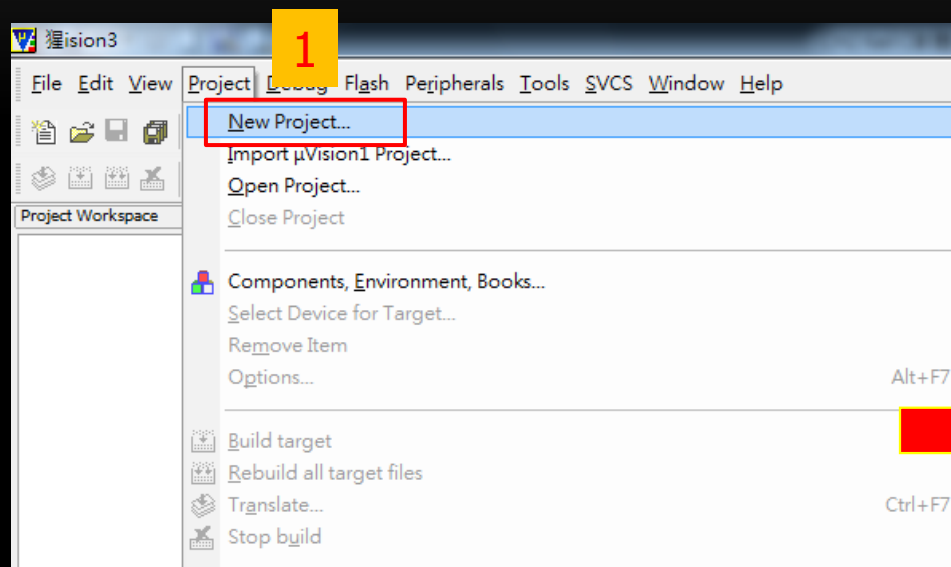
操作前準備-認識uVision3



操作前準備-認識uVision3

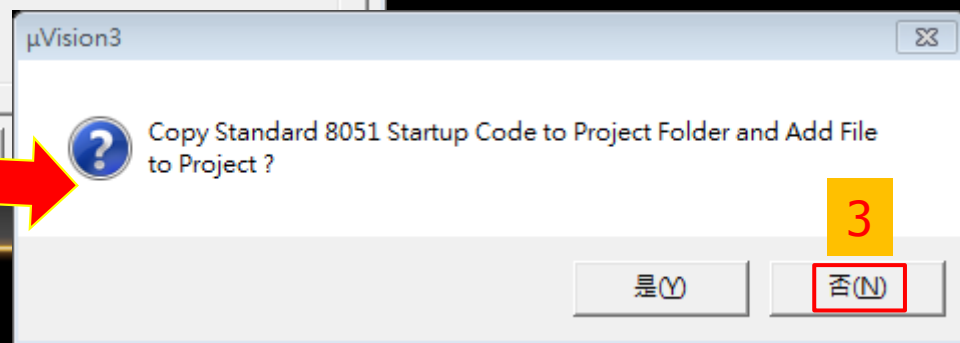
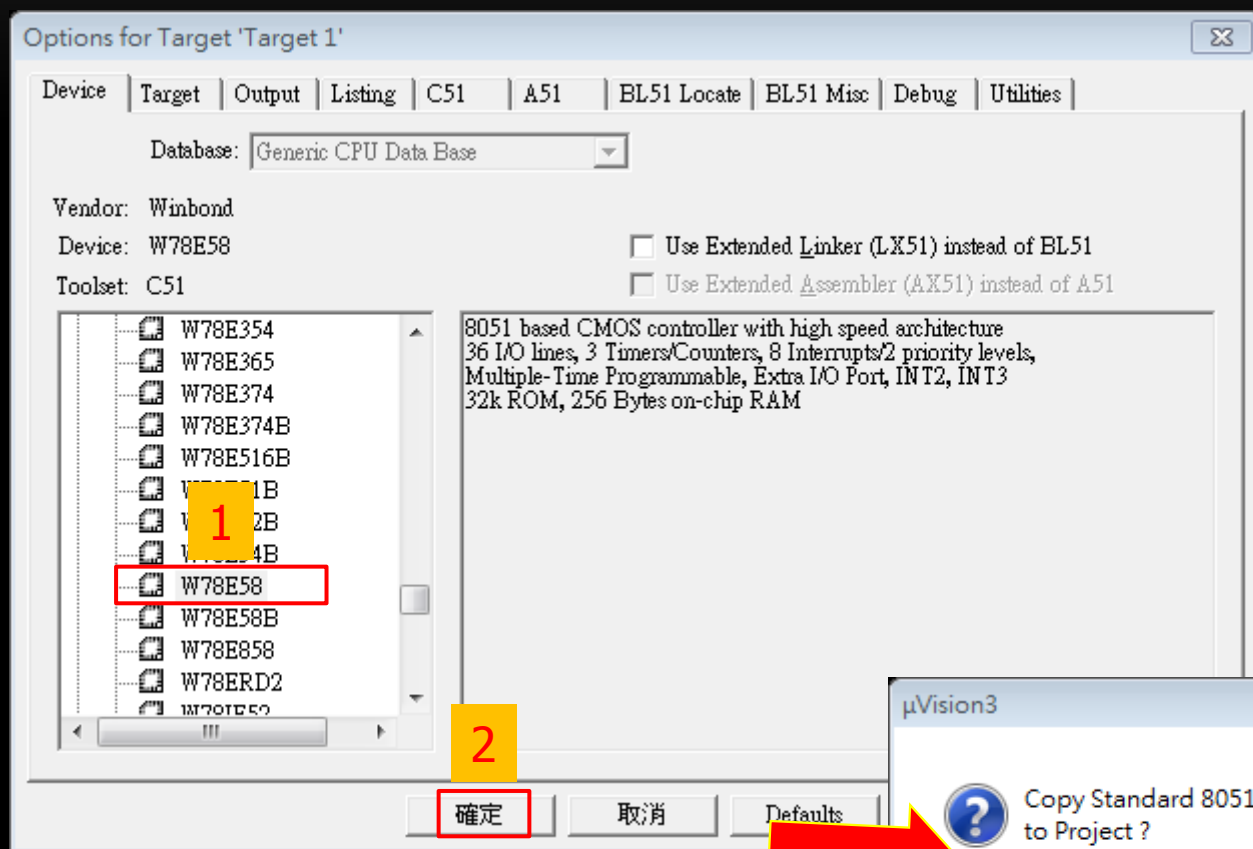


一、編輯原始程式-開啟新專案

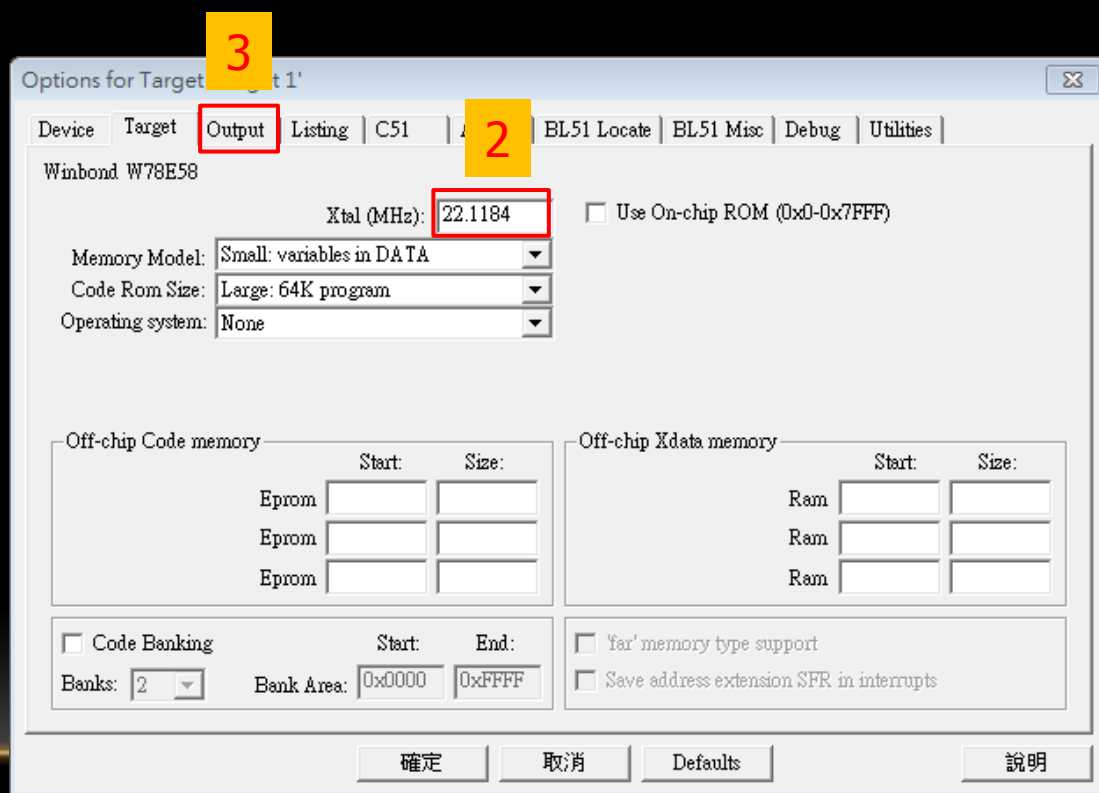
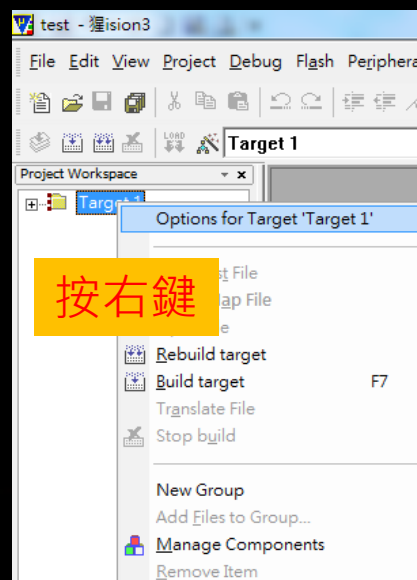
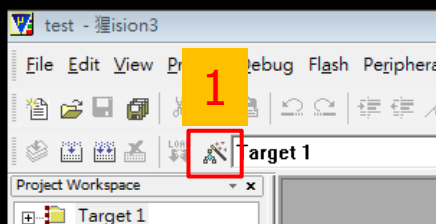


File Name : 8051\CH2\test.uv2

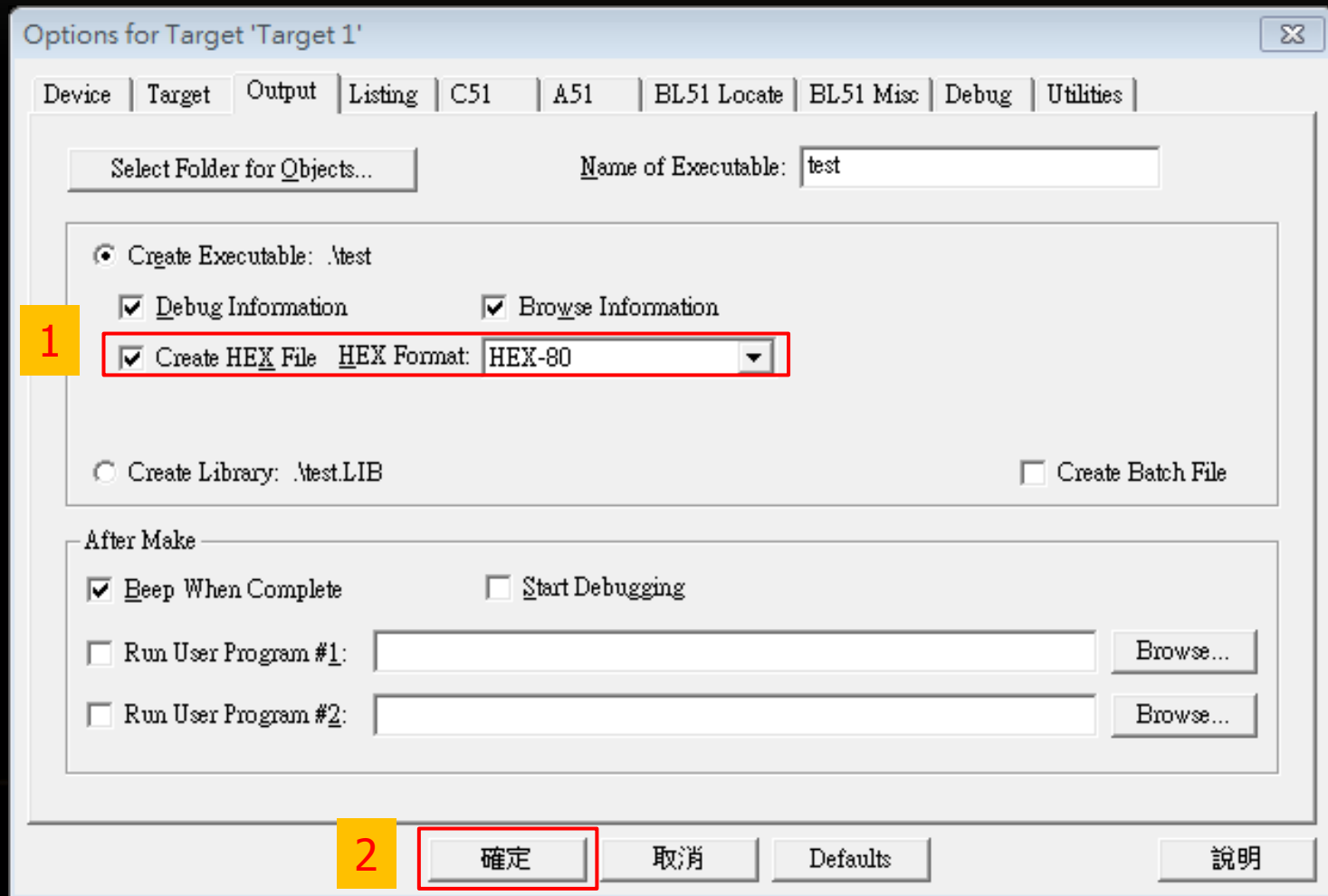
一、編輯原始程式-選擇晶片



一、編輯原始程式-設定晶片選項



一、編輯原始程式-設定晶片選項



一、編輯原始程式-開新檔案

1

2

3. 輸入XXX.C
(附屬檔名一定要打.C)

4

Save As

儲存於(O): CH2

名稱

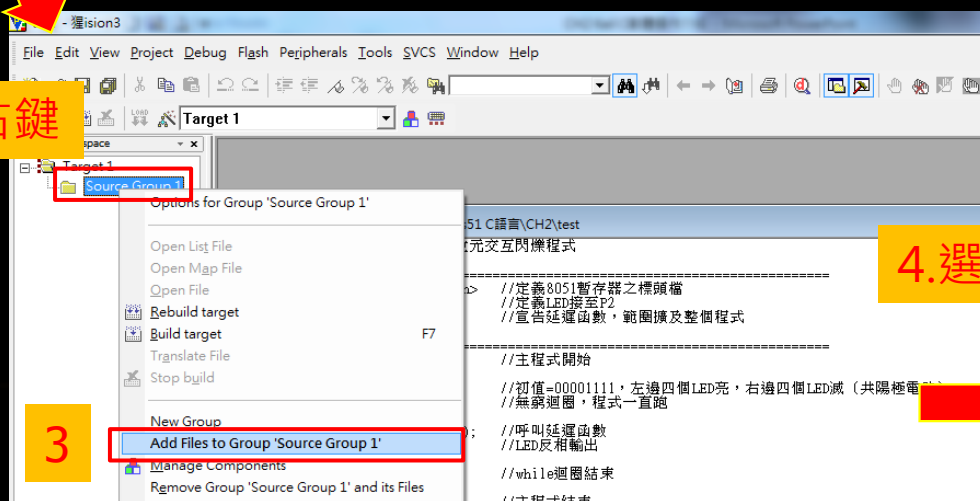
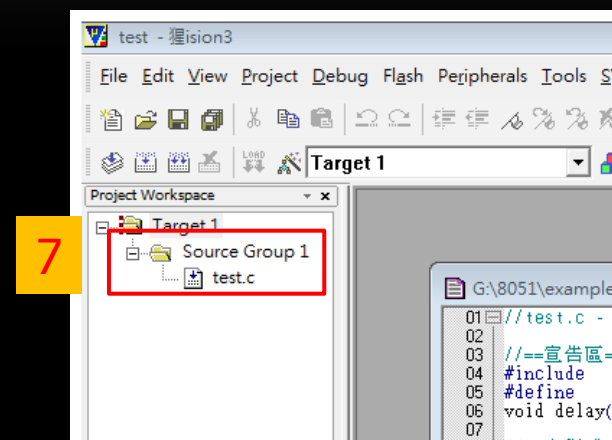
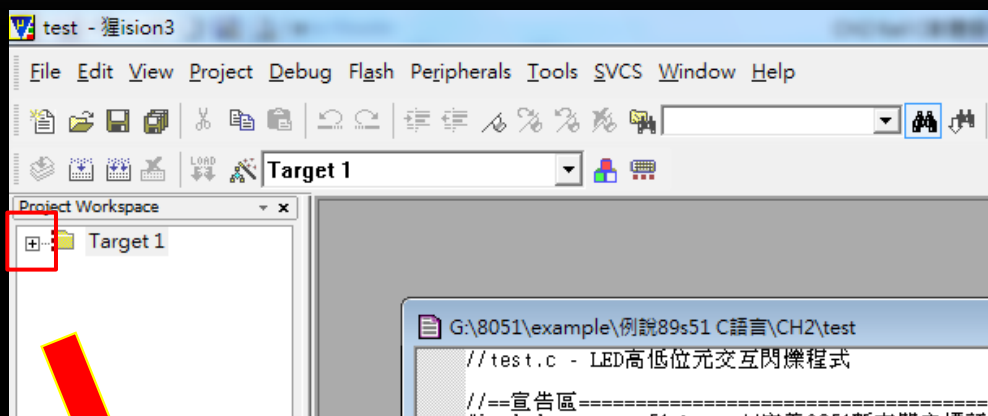
- test.Opt
- test.plg
- test
- test_Opt.Bak
- test_Uv2.Bak

檔案名稱(N): test.c

存檔(S)

取消

一、編輯原始程式-加入專案



一、編輯原始程式-編輯程式

```
01 //test.c - LED高低位元交互閃爍程式
02
03 //==宣告區=====
04 #include <reg51.h> //定義8051暫存器之標頭檔
05 #define LED P2 //定義LED接至P2
06 void delay(int); //宣告延遲函數，範圍擴及整個程式
07
08 //==主程式=====
09 main() //主程式開始
10 {
11     LED=0x0f; //初值=00001111，左邊四個LED亮，右邊四個LED滅（共陽極電路）
12     while(1) //無窮迴圈，程式一直跑
13     {
14         delay(200); //呼叫延遲函數
15         LED=~LED; //LED反相輸出
16
17     } //while迴圈結束
18
19 } //主程式結束
20
21 //==延遲函數=====
22 void delay(int x) //延遲函數開始，傳入x數值，不傳回數值
23 {
24     int i,j; //宣告整數變數i,j
25     for(i=0;i<x;i++) //計數x次，延遲x*5mS
26         for(j=0;j<1730;j++); //計數1730次，延遲5mS
27 } //延遲函數結束
28
```

二、編譯及連結

1

```
01 //test.c - LED高低位元交互閃爍程式
02
03 //==宣告區=====
04 #include <reg51.h> //定義8051暫存器之標頭檔
05 #define LED P2 //定義LED接至P2
06 void delay(int); //宣告延遲函數，範圍擴及整個程式
07
08 //==主程式=====
09 main() //主程式開始
10 {
11     LED=0x0f; //初值=00001111，左邊四個LED亮，右邊四個LED滅（共陽極電路）
12     while(1) //無窮迴圈，程式一直跑
13     {
14         delay(200); //呼叫延遲函數
15         LED=~LED; //LED反相輸出
16     } //while迴圈結束
17
18
19 } //主程式結束
20
21 //==延遲函數=====
22 void delay(int x) //延遲函數開始，傳入x數值，不傳回數值
23 {
24     int i,j; //宣告整數變數i,j
25     for(i=0;i<x;i++) //計數x次，延遲x*5ms
26     { //計數1730次，延遲5ms
27         for(j=0;j<1730;j++)
28     }
```

漏打; for(j=0;j<1730;i++);

TEST.C(27): error C141: syntax error near } 第27行語法錯誤

Build target 'Target 1'
compiling test.c...
target not created

二、編譯及連結

The screenshot shows the Keil IDE interface. The main window displays the source code for `test.c`, which is a C program for controlling LEDs. The code includes headers, defines the LED pin, and implements a delay function and a main loop that toggles the LED state.

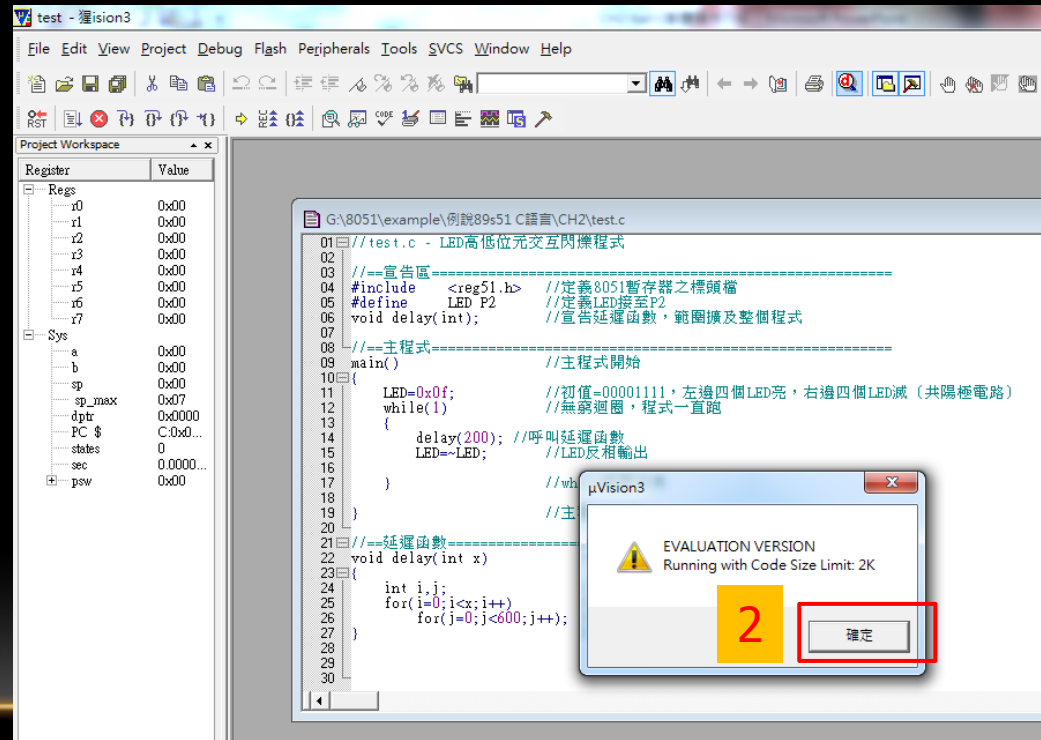
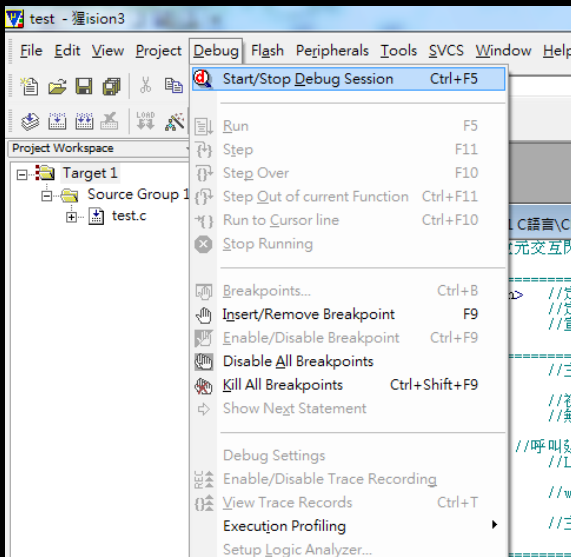
```
01 //test.c - LED高低位元交互閃爍程式
02
03 //==宣告區=====
04 #include <reg51.h> //定義8051暫存器之標頭檔
05 #define LED P2 //定義LED接至P2
06 void delay(int); //宣告延遲函數，範圍擴及整個程式
07
08 //==主程式=====
09 main() //主程式開始
10 {
11     LED=0x0f; //初值=00001111，左邊四個LED亮，右邊四個LED滅（共陽極電路）
12     while(1) //無窮迴圈，程式一直跑
13     {
14         delay(200); //呼叫延遲函數
15         LED=~LED; //LED反相輸出
16     } //while迴圈結束
17 } //主程式結束
18
19
20
21 //==延遲函數=====
22 void delay(int x) //延遲函數開始，傳入x數值，不傳回數值
23 {
24     int i,j; //宣告整數變數i,j
25     for(i=0;i<x;i++) //計數x次，延遲x*5mS
26         for(j=0;j<1730;j++); //計數1730次，延遲5mS
27 } //延遲函數結束
28
29
30
```

The bottom console window shows the build output for Target 1:

```
Build target 'Target 1'
compiling test.c...
linking...
Program Size: data=9.0 xdata=0 code=68
creating hex file from "test"...
"test" - 0 Error(s), 0 Warning(s).
```

三、除錯及模擬-啟動除錯及模擬畫面

1



三、除錯及模擬-模擬視窗

The screenshot displays a development environment with two main windows. On the left is the 'Project Workspace' window, which contains a 'Register' table. On the right is a code editor window showing the source code for a C program named 'test.c'.

除錯及模擬工具列

Register	Value
Regs	
r0	0x00
r1	0x00
r2	0x00
r3	0x00
r4	0x00
r5	0x00
r6	0x00
r7	0x00
Sys	
a	0x00
b	0x00
sp	0x07
sp_max	0x07
dptr	0x0000
PC \$	C:0x0...
states	389
sec	0.0003...
psw	0x00








```
G:\8051\example\例說89s51 C語言\CH2\test.c
01 //test.c - LED高低位元交互閃爍程式
02
03 //==宣告區==
04 #include <reg51.h> //定義8051暫存器之標頭檔
05 #define LED P2 //定義LED接至P2
06 void delay(int); //宣告延遲函數，範圍擴及整個程式
07
08 //==主程式==
09 main() //主程式開始
10 {
11     LED=0x0f; //初值=00001111，左邊四個LED亮，右邊四個LED滅（共陽極電路）
12     while(1) //無窮迴圈，程式一直跑
13     {
14         delay(200); //呼叫延遲函數
15         LED=~LED; //LED反相輸出
16     } //while迴圈結束
17 } //主程式結束
18
19
20
21 //==延遲函數==
22 void delay(int x) //延遲函數開始，傳入x數值，不傳回數值
23 {
24     int i,j; //宣告整數變數i,j
25     for(i=0;i<x;i++) //計數x次，延遲x*5ms
26         for(j=0;j<600;j++); //計數600次，延遲5ms
27 } //延遲函數結束
28
29
30
```

暫存器頁面

三、除錯及模擬-工具列介紹



重要

	本按鈕的功能是重置(RESET)CPU，同時讓程式從頭開始執行。
	本按鈕的功能是全速執行程式。
	本按鈕的功能是停止程式之執行。
	本按鈕的功能是單步執行，每按一下執行一個指令，若遇到函數(副程式)，則跳入該函數，同樣一步一步執行函數裡的指令。
	本按鈕的功能是單步執行，每按一下執行一個指令，若遇到函數，則直接執行完成該函數。
	本按鈕的功能是完成當時所執行的函數，跳出該函數，返回主程式。
	本按鈕的功能是執行到文字插入點(文字游標，即 I 形游標)所在的那一列指令，所以，在按本按鈕之前，先將文字插入點移至指定的那一列。

三、除錯及模擬-工具列介紹



	本按鈕的功能是開啓 STARUP.A51 視窗，以展示程式相對應的組合語言執行狀況。
	本按鈕的功能是啓用/停用追蹤記錄。
	本按鈕的功能是顯示追蹤記錄，並開啓 反組譯視窗 (Disassembly) 。
	本按鈕的功能是開啓 反組譯視窗 (Disassembly) 。
	本按鈕的功能是開啓 監視視窗 (Watch) ，在右下方。

三、除錯及模擬-工具列介紹



	本按鈕的功能是開啓指令碼包含率視窗(Code Coverage)。
	本按鈕的功能是開啓串列埠視窗(Serial #1)。
	本按鈕的功能是開啓記憶體視窗(Memory)，在下方中間。
	本按鈕的功能是開啓效能分析器視窗(Prefermance Analyzer)。
	本按鈕的功能是開啓邏輯分析儀視窗(Logic Analyzer)，以進行時序(波形)分析。
	本按鈕的功能是開啓符號視窗(Symbols Analyzer)，與專案視窗共用同一個位置。
	本按鈕的功能是開啓工具箱視窗(Toolbox)，提供進行除錯分析時，更新顯示之用。

三、除錯及模擬-周邊操作 peripherals

1

The screenshot displays the Proteus simulation environment. The 'Peripherals' menu is open, with 'I/O-ports' selected, showing a list of ports (Port 0 to Port 4). A red arrow points to this menu with the text '可顯示各port的狀態'. The main window shows a C program for an 8051 microcontroller. The code includes a header file, defines the LED port (P2), and contains a main function that initializes LEDs and a delay function. A yellow box with the number '2' is placed over the code editor. In the bottom right, a 'Parallel Port 2' dialog box is open, showing the port configuration for Port 2. The 'P2' and 'Pins' fields are both set to '0xFF', and all 8 bits are checked.

```
01 // test.c
02
03 //==宣告區
04 #include <reg51.h> //定義8051暫存器之標頭檔
05 #define LED P2 //定義LED接至P2
06 void delay(int); //宣告延遲函數，範圍擴及整個程式
07
08 //==主程式
09 main() //主程式開始
10 {
11     LED=0x0f; //初值=00001111，左邊四個LED亮，右邊四個LED滅（共陽極電路）
12     while(1) //無窮迴圈，程式一直跑
13     {
14         delay(200); //呼叫延遲函數
15         LED=~LED; //LED反相輸出
16     } //while迴圈結束
17 } //主程式結束
18
19
20
21 //==延遲函數
22 void delay(int x) //延遲函數開始，傳入x數值，不傳回數值
23 {
24     int i,j; //宣告整數變數i,j
25     for(i=0; i<x; i++) //計數x次，延遲x*5mS
26         for(j=0; j<600; j++) //計數600次，延遲5mS
27             ; //延遲函數結束
28
29
30
```

Parallel Port 2

Port 2

P2: 0xFF 7 Bits 0

Pins: 0xFF

三、除錯及模擬-周邊操作 peripherals

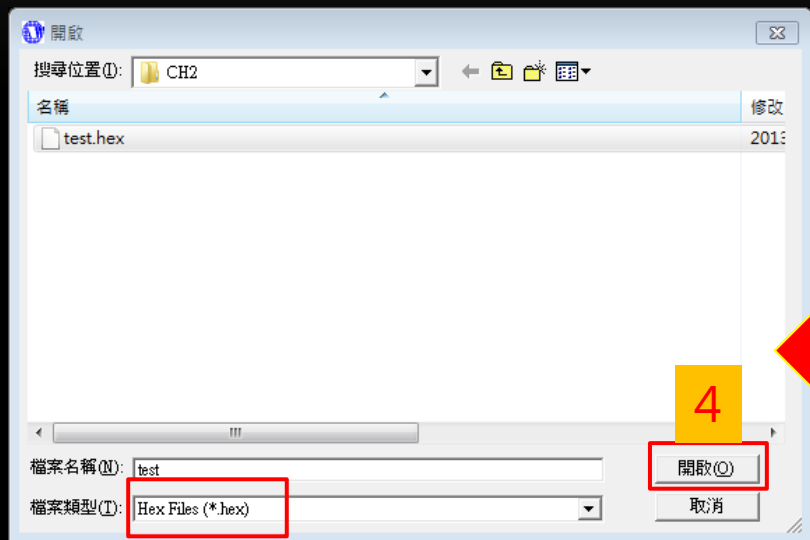
1. 按下RUN

```
01 //test.c - LED高低位元交互閃爍程式
02
03 //==宣告區==
04 #include <reg51.h> //定義8051暫存器之標頭檔
05 #define LED P2 //定義LED接至P2
06 void delay(int); //宣告延遲函數，範圍擴及整個程式
07
08 //==主程式==
09 main() //主程式開始
10 {
11     LED=0x0f; //初值=00001111，左邊四個LED亮，右邊四個LED滅（共陽極電路）
12     while(1) //無窮迴圈，程式一直跑
13     {
14         delay(200); //呼叫延遲函數
15         LED=~LED; //LED反轉
16     }
17 }
18
19 //==延遲函數==
20
21 void delay(int x)
22 {
23     int i,j;
24     for(i=0;i<x;i++)
25     {
26         for(j=0;j<600;j++);
27     }
28 }
29
30
```

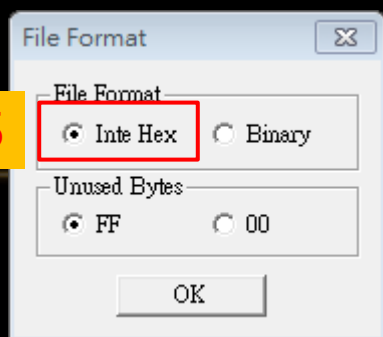
2. port2的狀態隨時改變

此狀態預設值為打勾

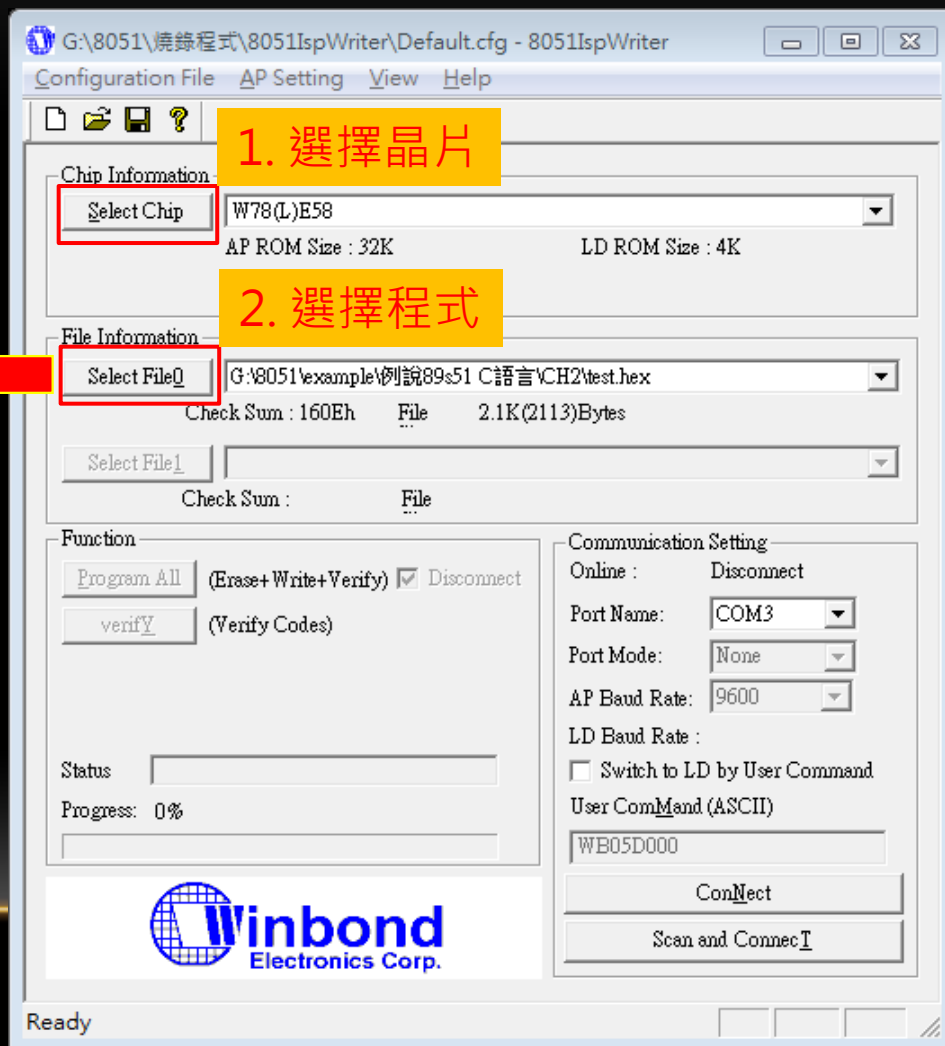
四、晶片燒錄-開啟燒錄程式



3. 選擇16進制檔案



5

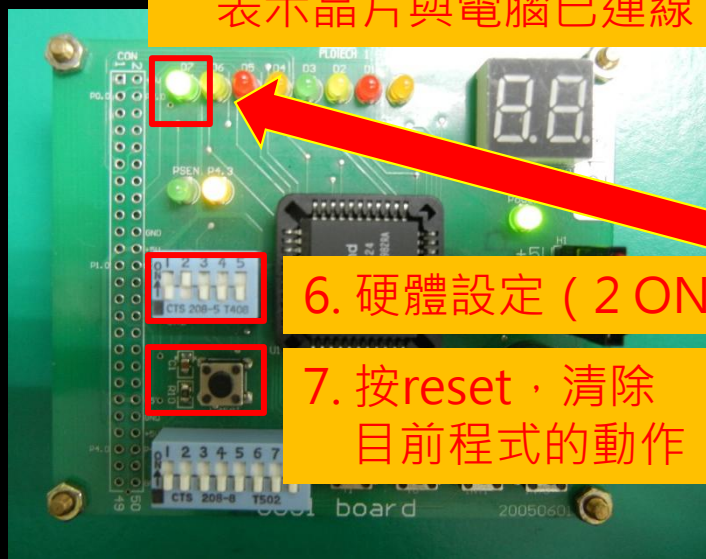
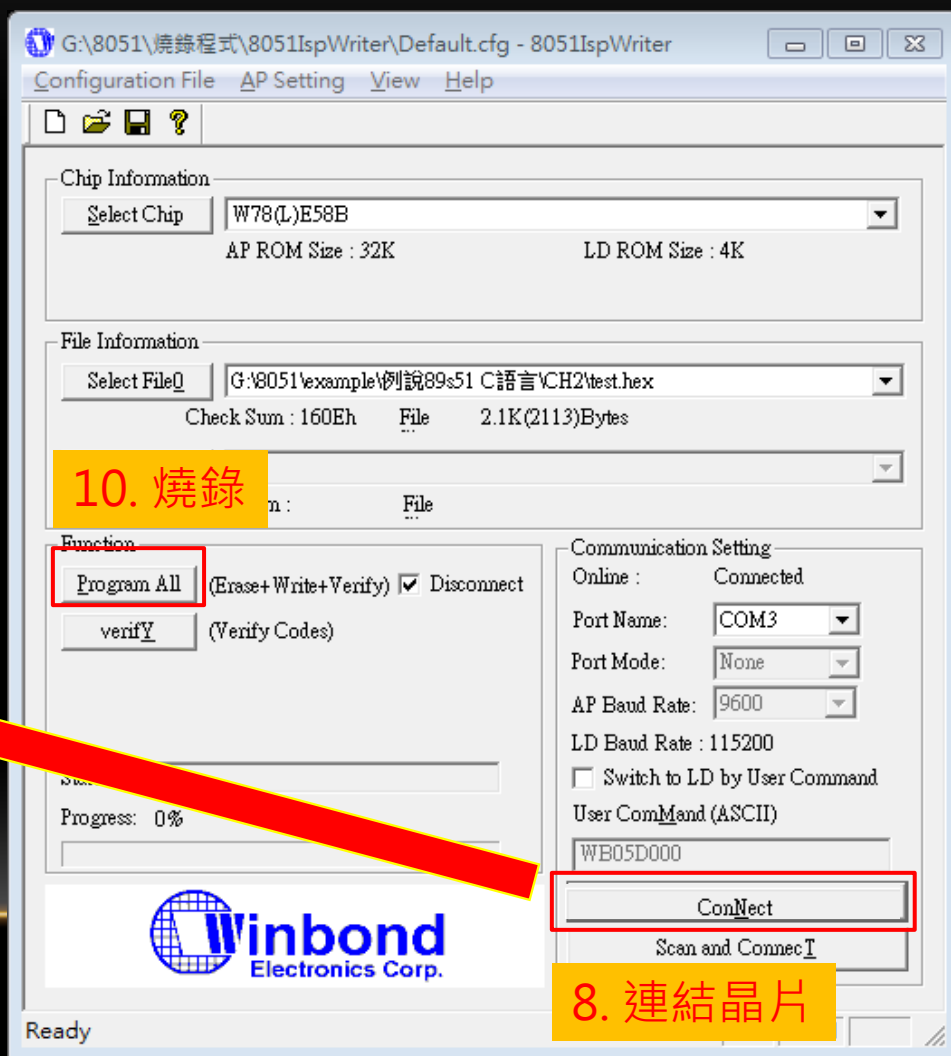


四、晶片燒錄-連結晶片及燒錄

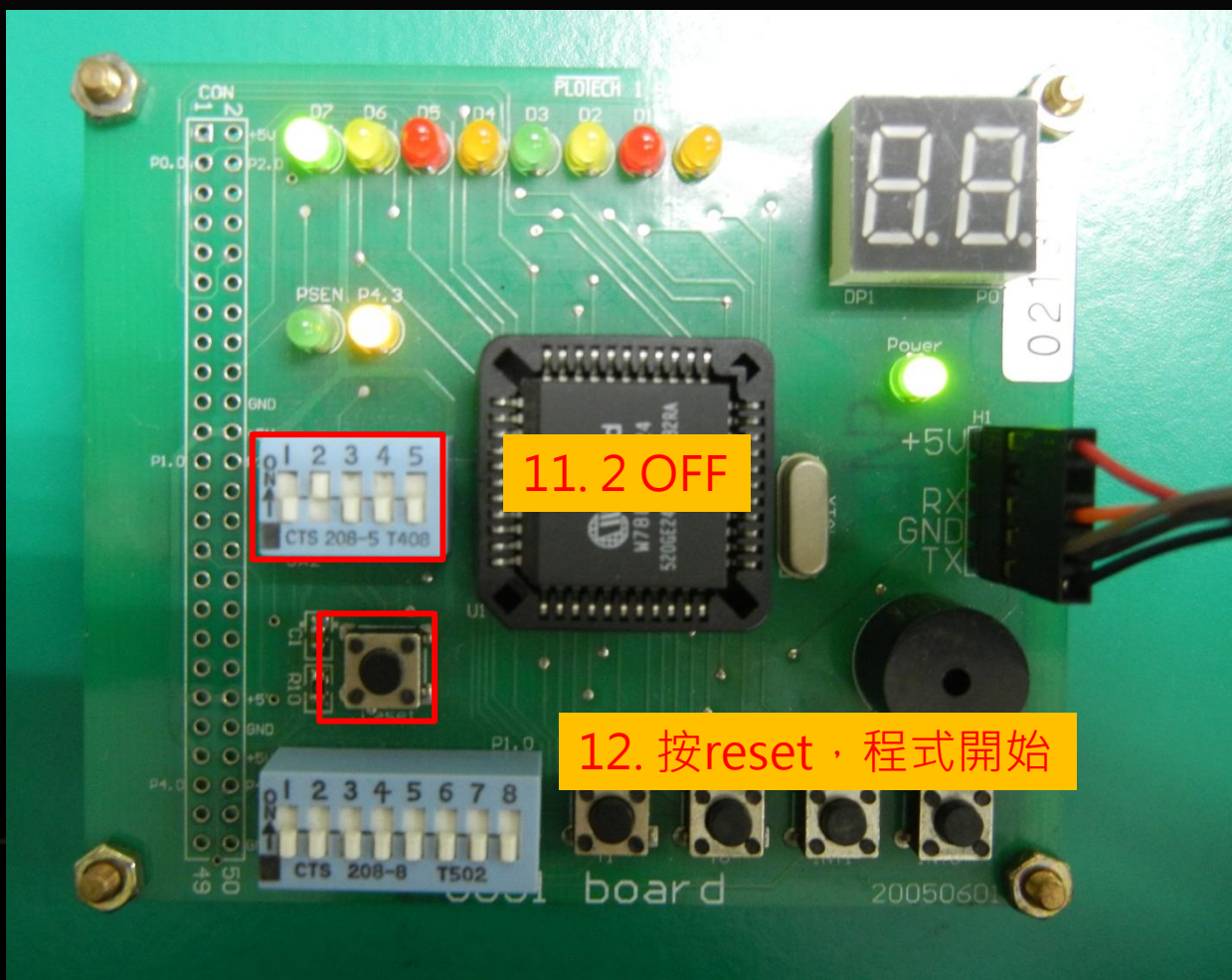
9. 按下按鈕後，燈會亮起，表示晶片與電腦已連線

6. 硬體設定 (2 ON)

7. 按reset，清除目前程式的動作



四、晶片燒錄-連結晶片及燒錄



四、晶片燒錄-燒錄完成

