



十速科技股份有限公司
tenx technology inc.

**Advance
Information**

TICE58

使用手册

Tenx reserves the right to change or discontinue this product without notice.

tenx technology, inc.

Preliminary

tenx technology, inc.

Rev 1.1, 2007/07/06

目录

产品简介	2
简介	2
系统配置	3
软件安装	4
快速开始	7
选单功能	13
主画面功能选单	13
主画面工具列	15
主画面窗口	16
环境设定窗口	21
帮助窗口	24
使用技巧	25
断点的使用	25
INCLUDE的使用	30
虚拟电路的使用	31
注意事项	35

产品简介

简介

为了缩短开发的时间和过程，十速科技特别开发出 IC 的仿真器---TICE58。TICE58 是由十速科技自行开发完成的，可支持 TM58 全系列的 IC。



(TICE58 外观)

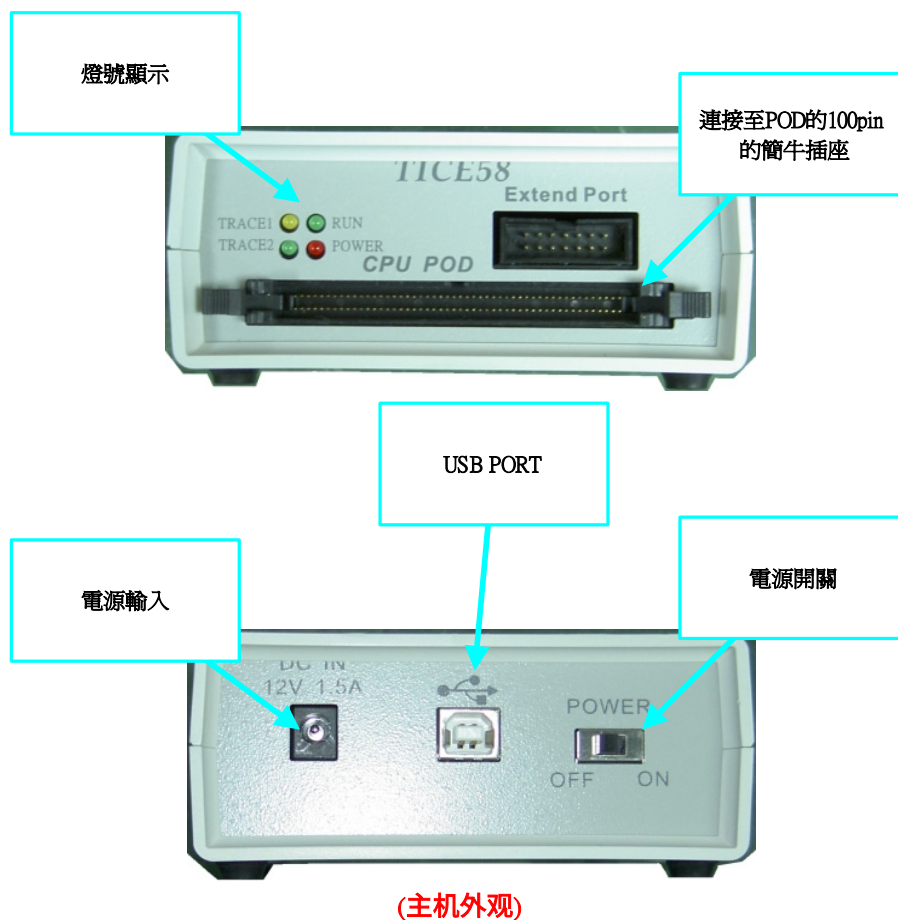
※系统要求:

- A、IBM PC 486 或以上。
- B、USB Port。
- C、64M 以上内存。
- D、WINDOWS 2000/XP 操作系统。

※TICE58 特点:

- A、用 USB 和计算机作联机，仿真速度快。
- B、使用内部或外部的振荡器。
- C、可设置断点。
- D、提供内部和外部的 RESET。
- E、提供硬件仿真和软件仿真。
- F、编辑、除错、组译都可在 TICE58 的软件完成。
- G、除了提供 AUTO (GO)和单步执行(STEP)外，还提供了 STEP INTO 的功能。

系统配置

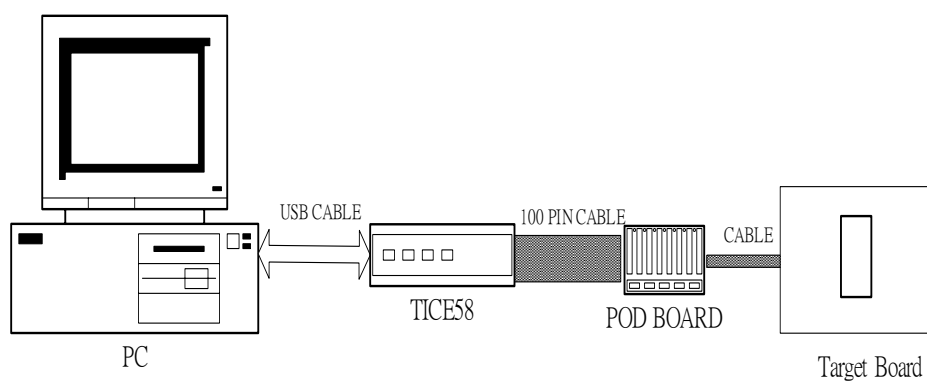


※用100pin的连接线连接TICE58主机和相应POD板。

※连接USB线以PC至TICE58主机。

※连接电源线。

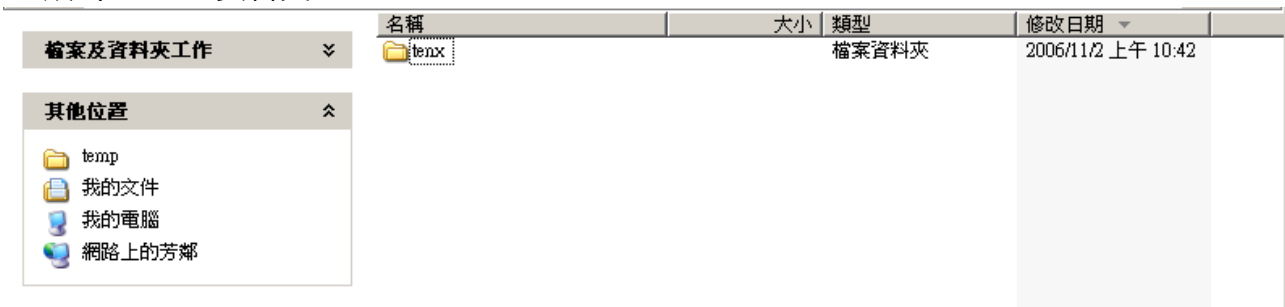
※打开电源及完成硬件安装。



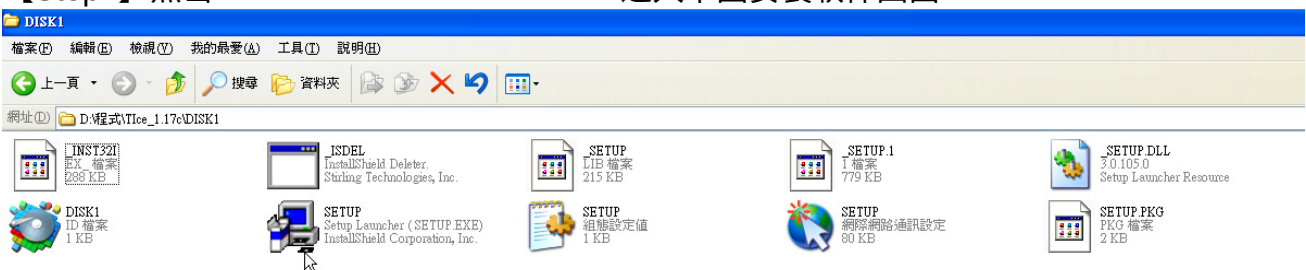
软件安装

【Step1】双击TICE58的软件解压缩，解压缩完成后，会产生一个TICE58的目录：

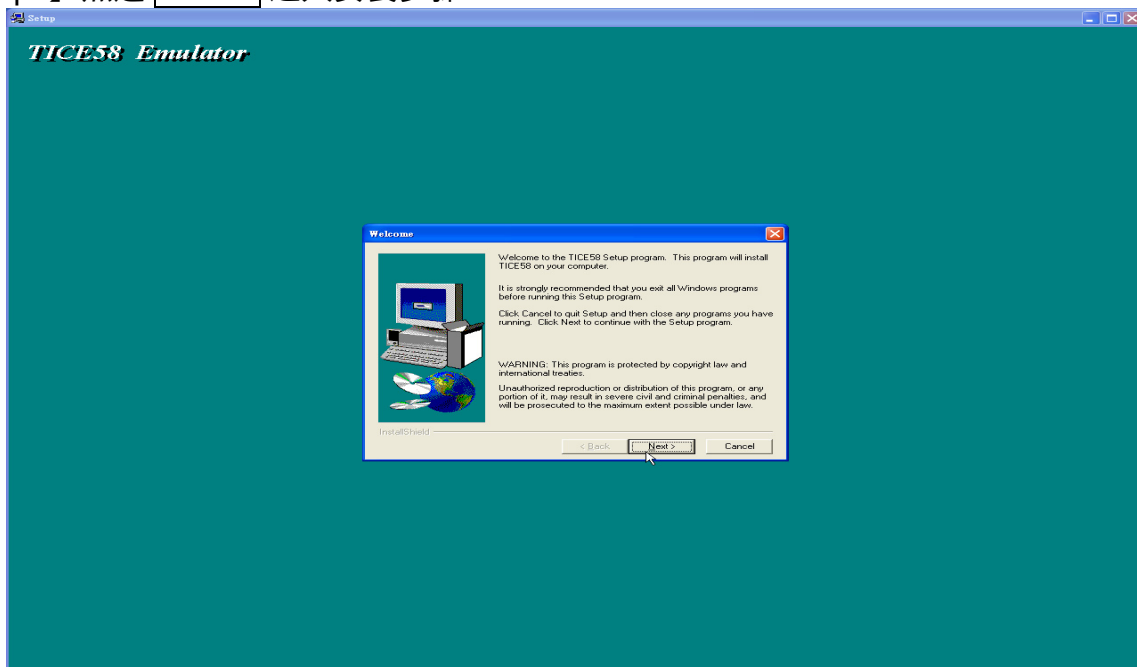
双击TICE58资料夹：



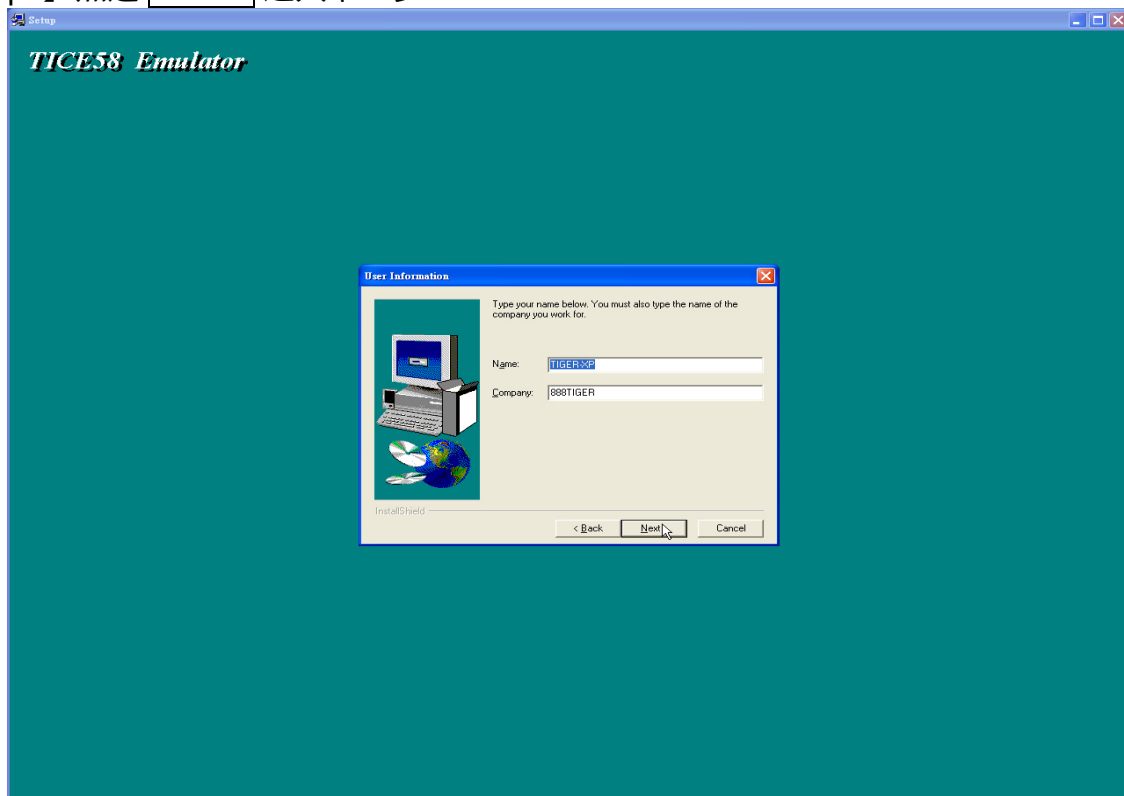
【Step2】点击  **SETUP** Setup Launcher (SETUP.EXE) InstallShield Corporation, Inc. 进入下图安装软件画面：



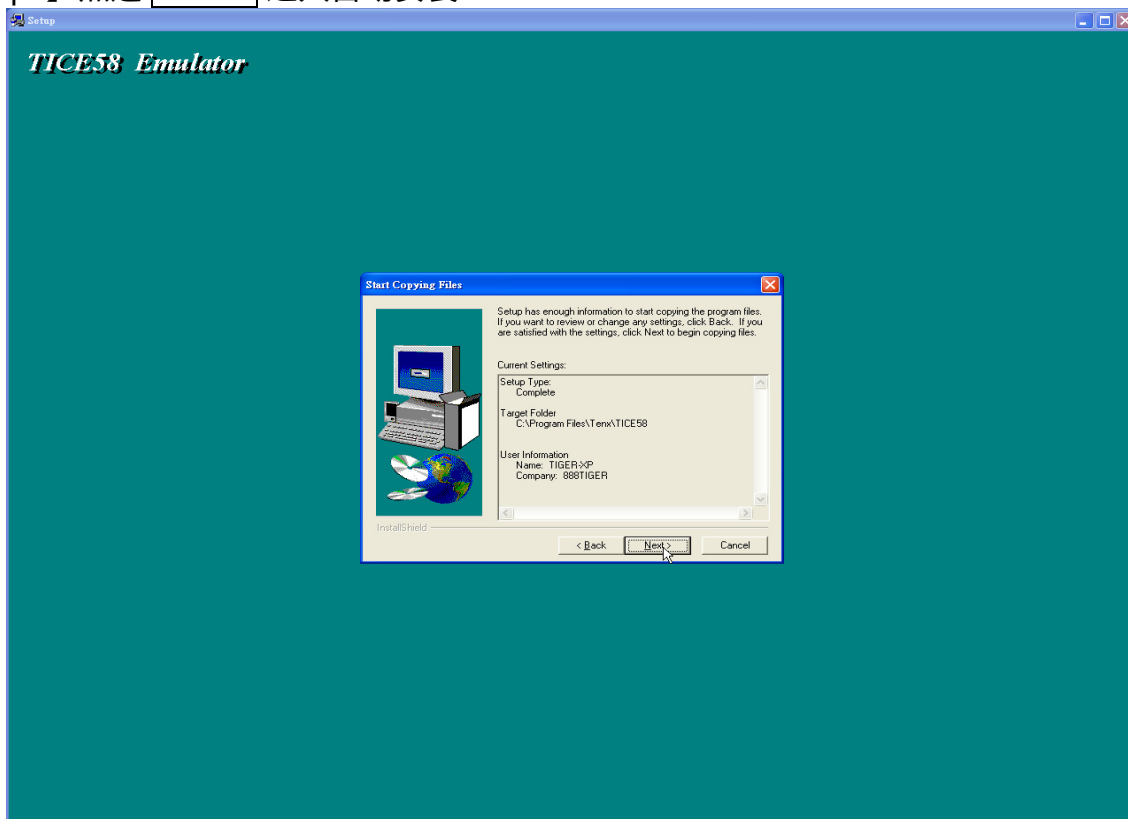
【Step3】点选 **NEXT** 进入安装步骤：



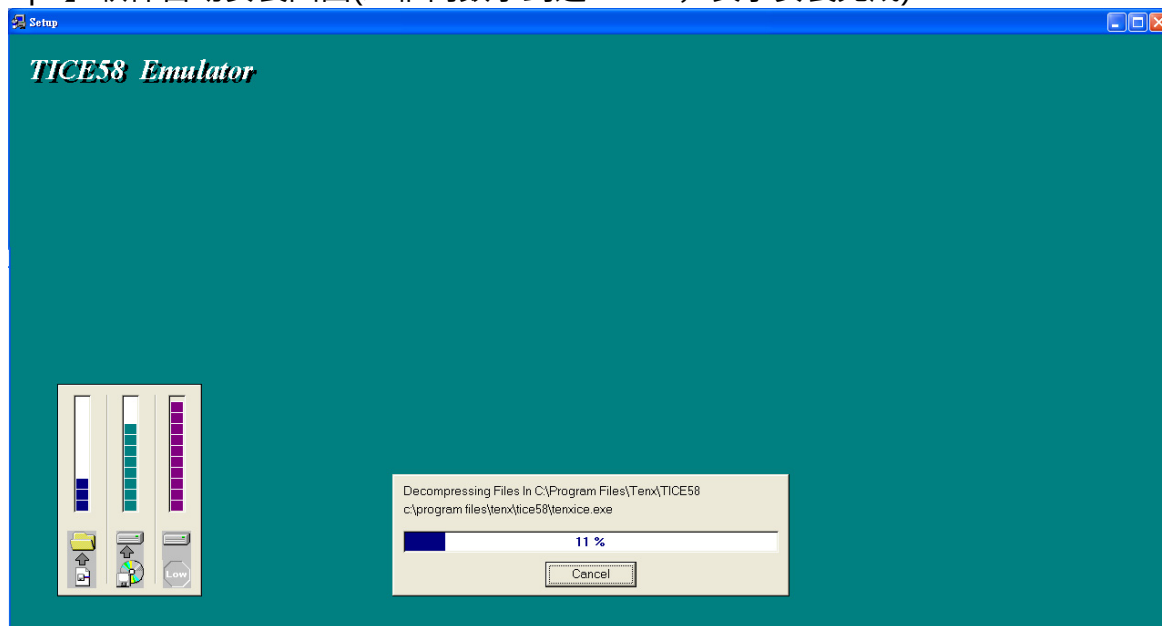
【Step4】点选 **NEXT** 进入下一步：



【Step5】点选 **NEXT** 进入自动安装：

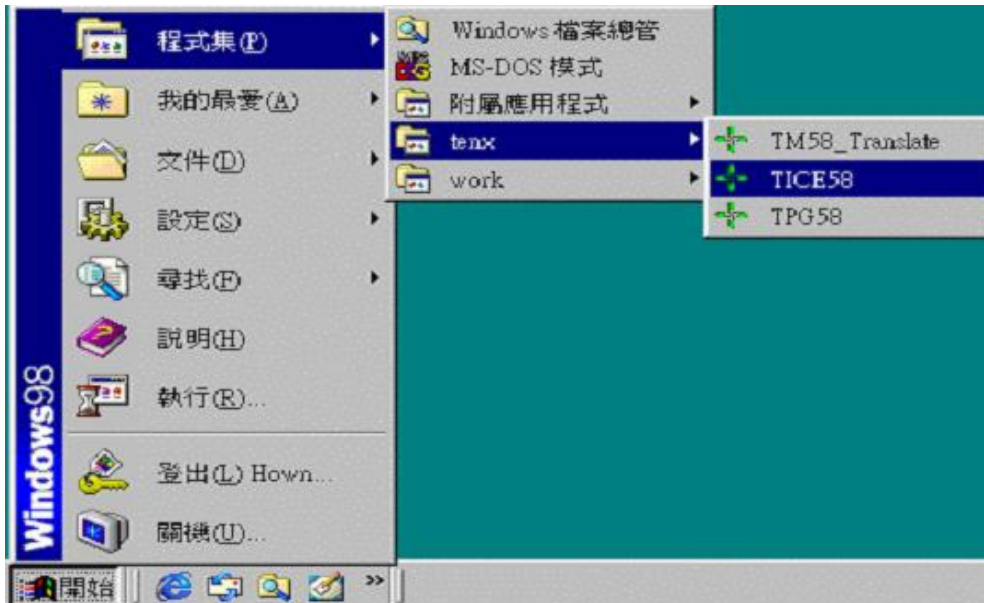


【Step6】 软件自动安装画面(红框内数字到达 100%，表示安装完成)：

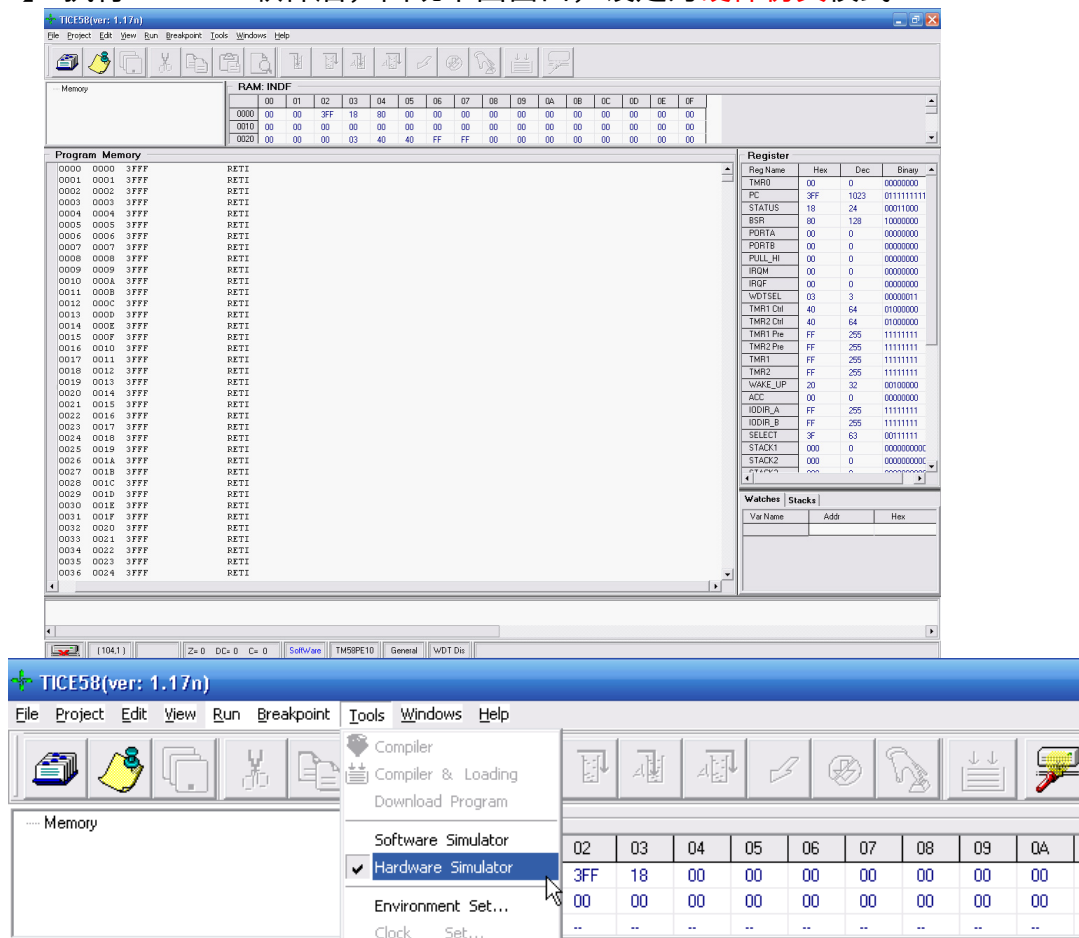


快速开始

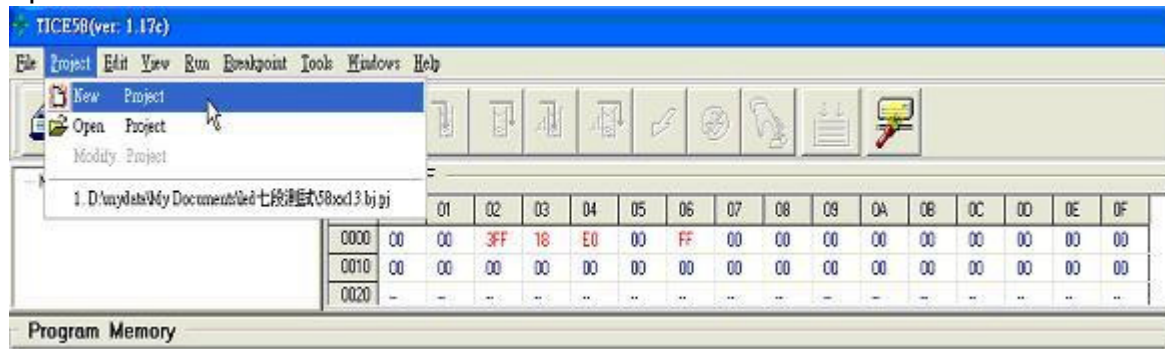
【Step1】执行 TICE58 程序：



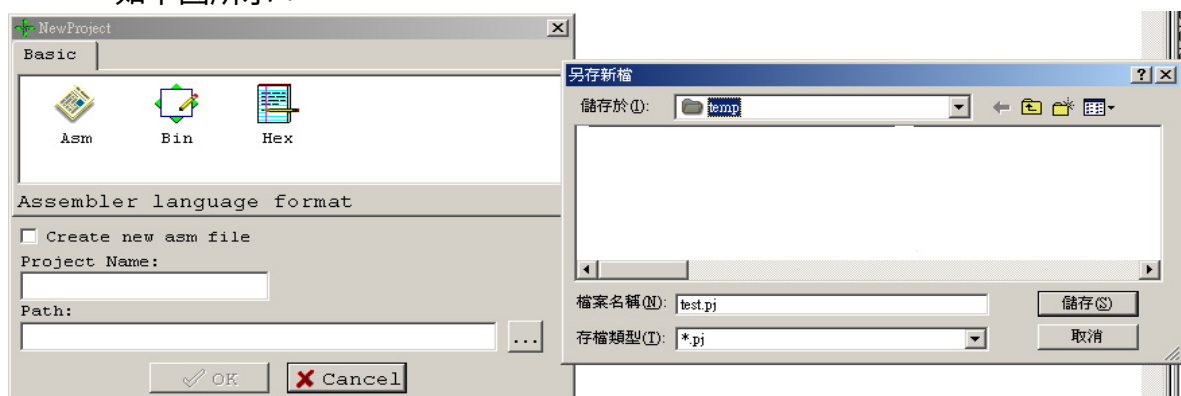
【Step2】执行 TICE58 软件后，出现下面窗口，设定为**硬件仿真**模式：



【Step3】选择专案 → 开启专案：

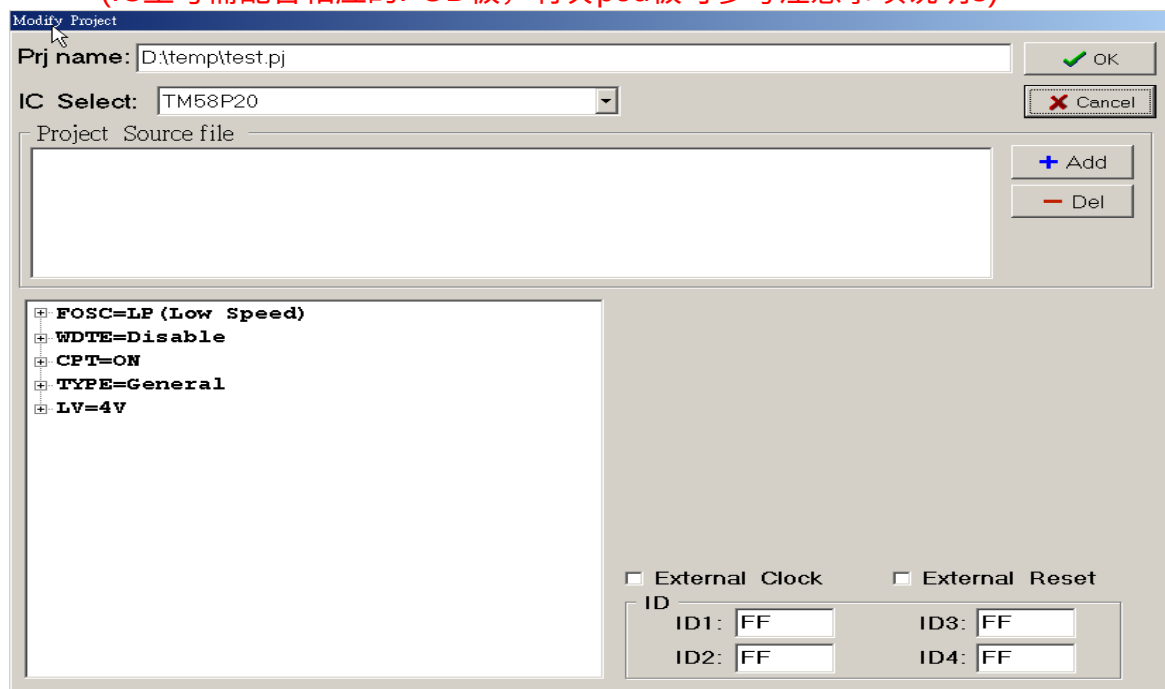


【Step4】选择路径，然后输入项目的名称(例如：“test.pj”)然后按下储存键即可，如下图所示：

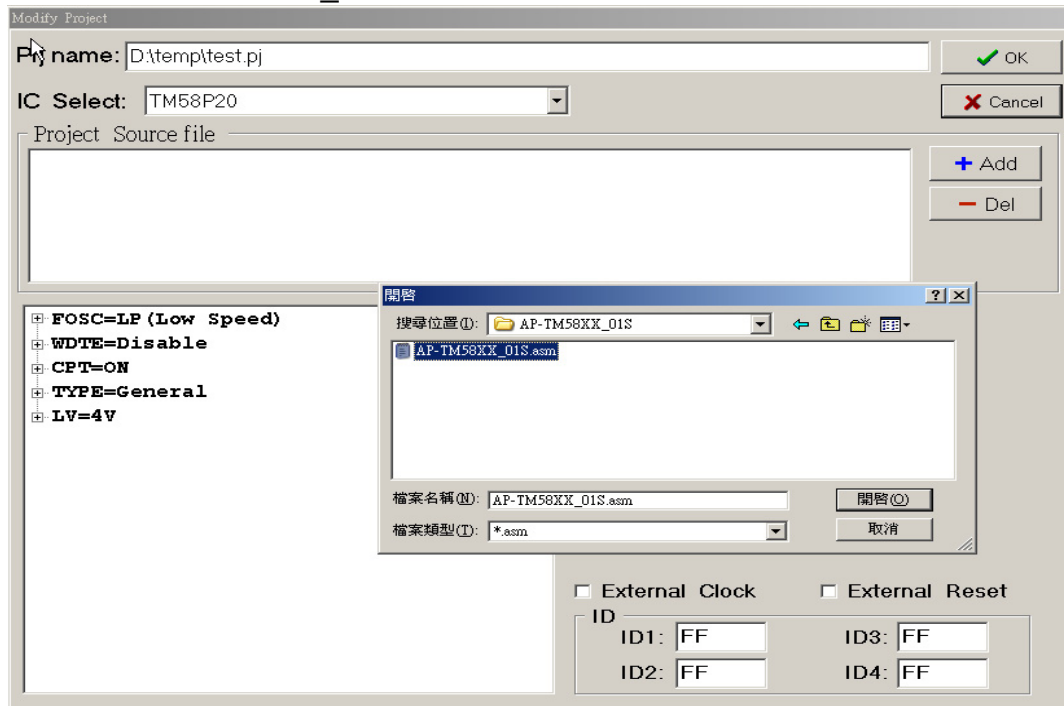


【Step5】按下OK键后，会出现选择IC型号、CONFIGURATION设定和加入Source Code的画面，如下图：

(IC型号需配合相应的POD板，有关pod板可参考注意事项说明3)

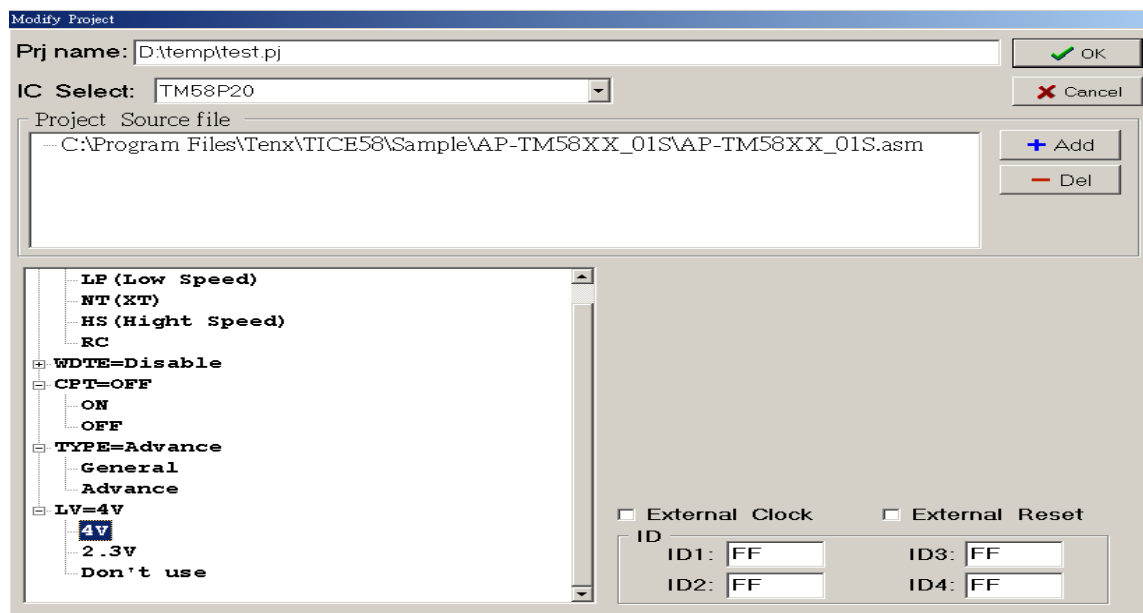


- 【Step6】 首先请先选择IC型号，再来点选 **+ Add**，加入Source Code(在此我们加入TICE58的example的档案，路径为C:\Program Files\Tenx\TICE58\example\AP-TM58XX_01S, <C: 会因安装不同而不同>) 选择AP-TM58XX_01S.asm后，按下开启。



- 【Step7】 结束加入Source Code后，会回到configuration 设定的画面，如下图。根据example的程序，我们的设定为：
 Type: Advanced
 FOSC: HS
 WATCHDOG: DISABLE

※ 设定完成后，按下 。



※ External Clock:

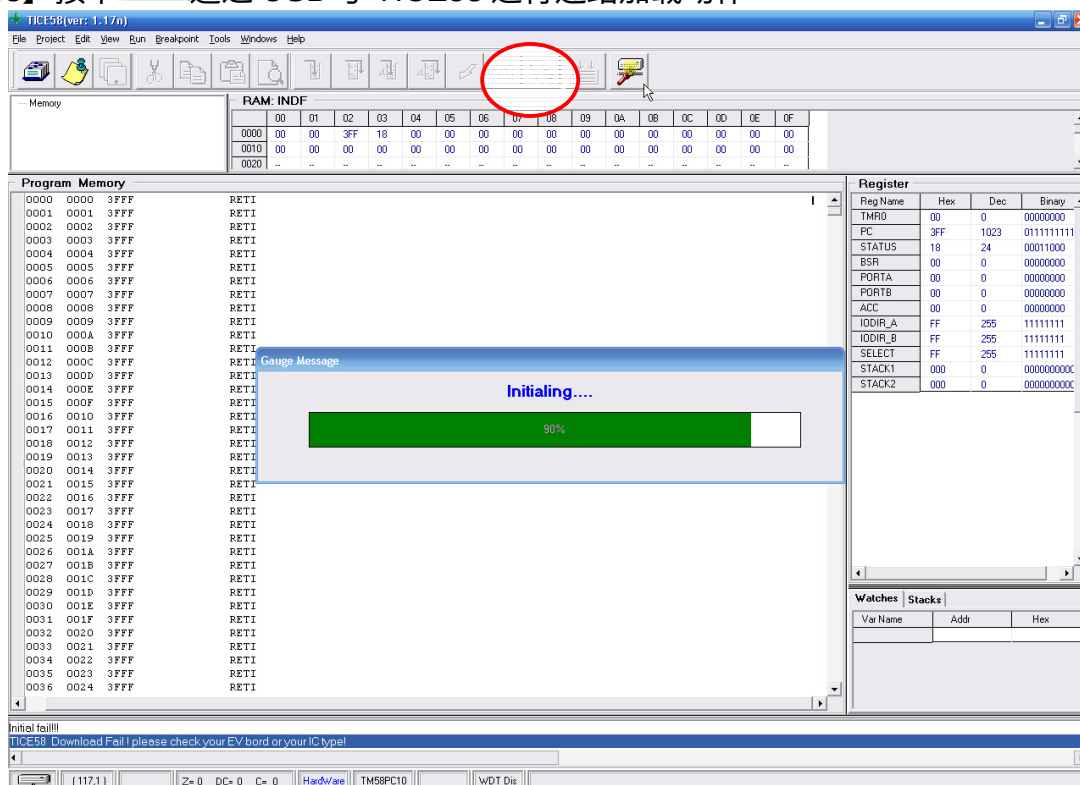
为选择使用 内部／外部 CLOCK，没打勾为选择内部 CLOCK(内部频率固定为 4MHz)
当选择使用外部 CLOCK 时，需调整 TM58POD 板上的 Y1、R1、C1 与 C2 的数值，
对应表如下：

Y1	R1	C1 C2
32.768K	4.7K	4.7n
455K	4.7K	680p
1M	27K	20p
2M	20K~27K	20p
4M	20K~27K	20p
8M	1K	20p
10M	1K~470	20p
12M	470	20p
14.374M	100	20p
16M	3.3	20p

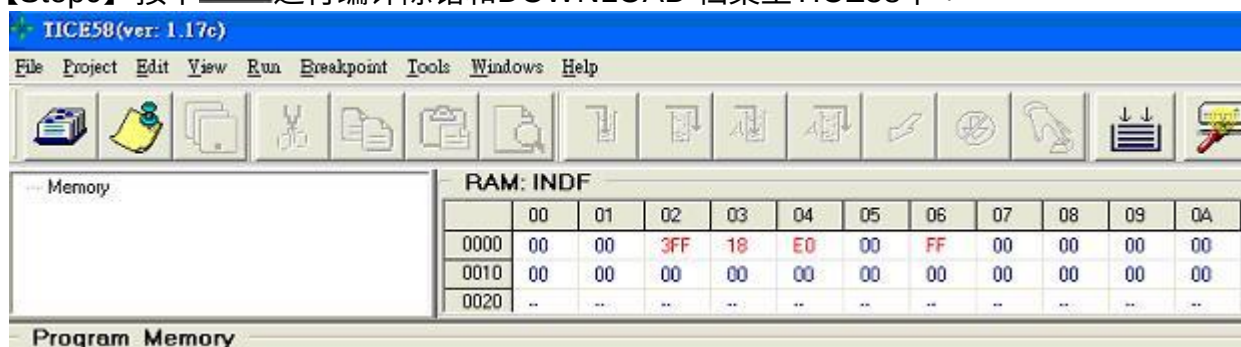
※ External Reset: 为选择使用 内部／外部 RESET，没打勾为选择内部 RESET。

※ ID: 此选项中有四个，是让客户自行设定 ID，默认值为 FF。

【Step8】按下  透过 USB 与 TICE58 进行连结加载动作：



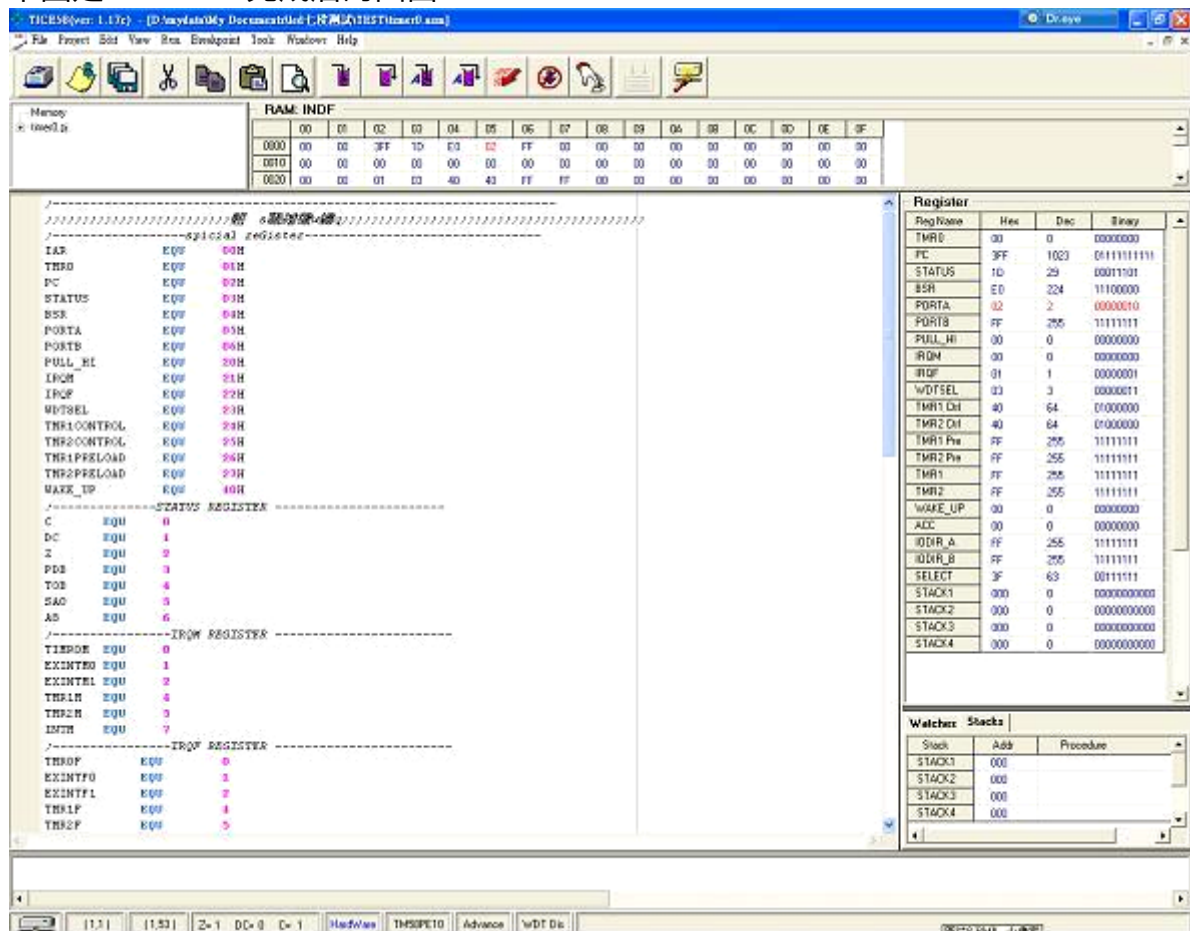
【Step9】按下  进行编译除错和DOWNLOAD 档案至TICE58中：



※ 如果程序编译没有错误且完成程序加载的话，即可按全速执行  开始执行，或是按下单步执行键行单步  的功能。

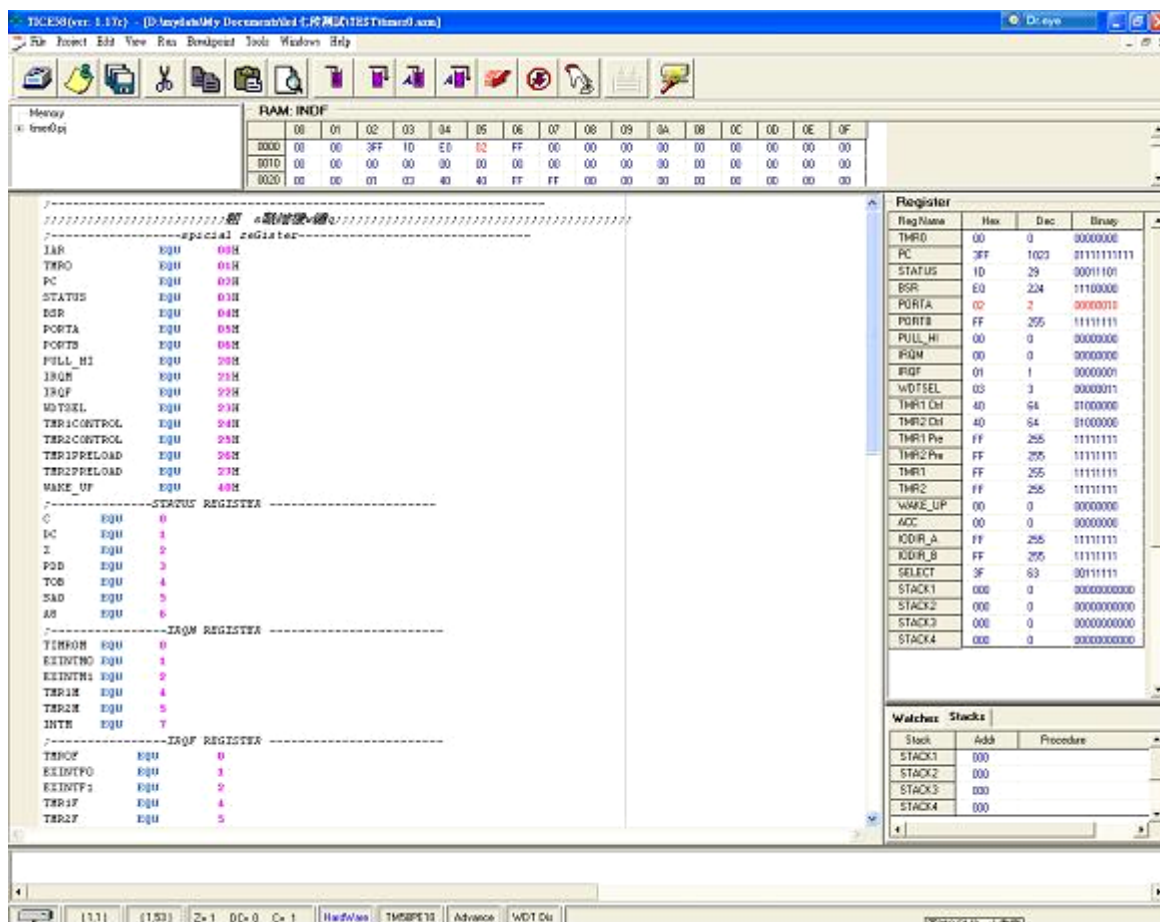
※ 如果在Download时发生错误，则需按下 ，重新Download。

下图是Download完成后的画面：



选单功能

主画面功能选单



File:

- Create new file：建立一个新的 ASM 档案。
- Open file：开启旧的档案。
- Save file：储存目前的档案。
- Save as file：另存新档。
- Save all file：储存项目中所有的档案。
- Close all file：关闭项目中所有的档案。
- Printer：打印档。
- Printer set：打印设定
- Exit：关闭 TICE58 的程序。

Project:

- New project：建立新的项目。
- Open project：开启旧的专案
- Modify project：修改项目中设定，包括 Configuration 的设定、ic 型号变更.....。

Edit:

- Copy : 复制。
- Cut : 剪下。
- Paste : 贴上。
- Find : 寻找。
- Replace : 取代。

View:

- Register : 显示 Register 的窗口。
- Watches and Stacks : 显示 Watches 和 Stacks 的窗口。
- Message : 显示讯息的窗口。
- RAM : 显示内存的窗口。
- Circuit Output(只有在软件仿真时有用) : 请参考使用技巧 4.3
- Remote Controlled(只有在软件仿真和时有用)

RUN:

- Free run : 全速执行程序。
- Auto step into → 自动单步执行(所有程序)。
- Auto step over : 自动单步执行(只执行程序)。
- Step over : 单步执行(所有程序)。
- Setp into : 单步执行(只执行程序)。
- Run to cursor : 程序只执行到光标处。
- Stop : 停止。
- Warm reset : 重置。

Break Point:

- Break Point List : 断点清单。

Tools:

- Compiler : 编辑
- Compiler & Loading (只有在硬件仿真时有用) : 编辑完成后,
下载程序到 EV。
- Download Program (只有在硬件仿真时有用) : 下载程序到 EV。
- Software Simulator : 使用软件仿真, 不需 POD 板。(部份功能无法使用)
- Hardware Simulator : 使用硬件仿真, 必须连接 POD 板。
- Environment Set : 操作环境设定。

主画面工具列

按键图标	功能键	备注	
		开启旧专案	开启上一次或是之前的项目
		建立新的来源档	建立新的来源档(如 .asm)
		储存项目中所有的档案	储存项目中所有的档案
		剪下	剪下
		贴上	贴上
		寻找	寻找
	F7	单步执行 (所有程序)	单步执行所有的程序
	F8	单步执行 (只执行主程序)	单步执行, 不执行 CALL 指令 (只执行主程序)
	F10	自动单步执行 (所有程序)	自动单步执行所有的程序
	F11	自动单步执行 (只执行主程序)	自动单步执行所有的程序, 但不执行 CALL 指令
	F9	执行程序	全速执行程序
	F3	停止	停止执行
	F4	重置	将程序回到最初的起始位置
		执行 Compiler 跟 Download	执行 Compiler,并在没有任何错误的情况下 Download 档案到 ICE。
		重新 Initial ICE	重新 Initial ICE。

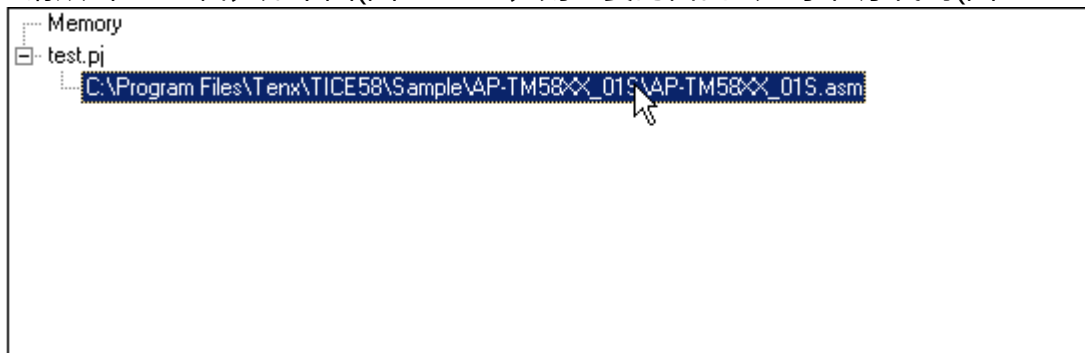
主画面窗口

【项目档案】

此窗口主要可分为来源文件的显示或是 Memory 格式的显示。使用者可依自己的习惯设定显示的信息。

a. 程序代码(*.asm)的显示

请双击.asm 档，如下图(图 3.3.1-1，则主要的窗口会显示程序代码(图 3.3.1-2)



(图 3.3.1-1)

```

;=====
;=====
org    7f5h    ;重定向量地址
• goto    main
org    7feh    ;中断副程式入口位址
• goto    extint
org    000h    ;程式在ROM中的起始地址
;
;
;=====
;tmr0时间处理副程式
;=====
readtime:
• movm    tmr0,w        ;读取tmr0的值,结果放入累加器中
• xoram    time_temp,w    ;tmr0的值与time_temp值相异或,结果放入累加器中
• movam    time_buf        ;将tmr0与time_temp异或的结果送入time_buf中
• xoram    time_temp,f    ;tmr0的值送入time_temp中
• btms    time_buf,7        ;判断128*64us/1000=8.192ms是否已到
• ret        ;8.192ms未到,则退出副程式
• btmsc    del_flag,0        ;8.192ms已到,判断是否处于遮罩外中断状态
• incm    del_cnt    ;处于遮罩外中断状态,则del_cnt值加1
• incm    count1        ;count1值加1

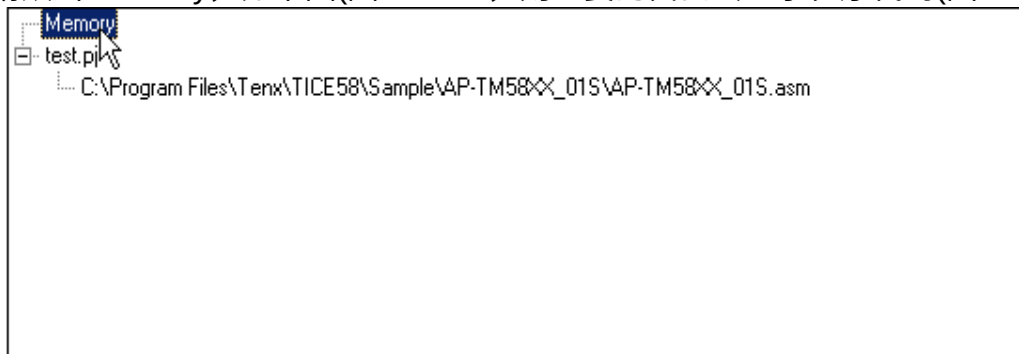
• movla    3dh        ;判断61*8.192ms=499.712ms约0.5S是否已到
• xoram    count1,w
• btms    status,z
• ret        ;499.712ms未到,则退出副程式
• clrm    count1    ;499.712ms已到,清count1
• btmsc    led_flag,led1    ;判断led1控制标志位元是否?1?
• goto    turn_on_led1    ;led1控制标志位元?1,转去开led1
• btmsc    led_flag,led2    ;led1控制标志位元不?1,判断led2控制标志位元是否?1?

```

(图 3.3.1-2)

b. 机器码的显示

请双击 Memory，如下图(图 3.3.1-3，则主要的窗口会显示程序代码(图 3.3.1-4)



(图 3.3.1-3)

Program Memory					
	0078	004E	2000		NOP
	0079	004F	0B06		BSM PORTB, LED7
	0080	0050	3101		MOVLA 01H
	0081	0051	2088		MOVAM LED_FLAG
	0082	0052	3F7F		RET
→	0083	0053	2189	MAIN	CLRM COUNT1
	0084	0054	218A		CLRM DEL_CNT
	0085	0055	218F		CLRM DEL_FLAG
	0086	0056	3102		MOVLA 02H
	0087	0057	2088		MOVAM LED_FLAG
	0088	0058	3101		MOVLA 01H
	0089	0059	2086		MOVAM PORTB
	0090	005A	2100		CLRA
	0091	005B	2006		IODIR PORTB
	0092	005C	2085		MOVAM PORTA
	0093	005D	310F		MOVLA 0FH
	0094	005E	2005		IODIR PORTA
	0095	005F	3105		MOVLA 05H
	0096	0060	2002		SELECT
	0097	0061	3121		MOVLA B'00100001'
	0098	0062	20A0		MOVAM WAKEUP
	0099	0063	3184		MOVLA B'10000100'
	0100	0064	20A1		MOVAM IRQM
	0101	0065	2301		MOVW TMRO, W
	0102	0066	208D		MOVAM TIME_TEMP
	0103	0067	3600	START	CALL READTIME
	0104	0068	0521		BTMSC IRQM, EXINTM
	0105	0069	3A67		GOTO START
	0106	006A	0005		BTMSS PORTA, 0
	0107	006B	3A67		GOTO START

(图 3.3.1-4)

【RAM】

显示 IC RAM 的信息，对应位置的意义请参考每颗 IC 的 DataSheet。

RAM: PC																
	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
0000	00	78	53	1C	00	0F	01	FF	02	00	00	00	00	00	00	00
0010	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0020	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
0030	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0040	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
0050	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0060	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
0070	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

※ 在 Debug 中，如需要修改，只需在欲修改的位置，点两下(如下图)后，输入数值，按下 **OK** 键即可。

Address: 12 H

Hex

上图会显示要修改的位置

【Register】

显示 IC Register 的信息，每个 Register 的意义请参考每颗 IC 的 DataSheet。

Register			
Reg Name	Hex	Dec	Binary
TMR0	78	120	01111000
PC	053	83	00001010011
STATUS	1C	28	00011100
BSR	00	0	00000000
PORTA	0F	15	00001111
PORTB	01	1	00000001
PORTC	FF	255	11111111
ACC	00	0	00000000
IODIR_A	FF	255	11111111
IODIR_B	FF	255	11111111
IODIRC	FF	255	11111111
SELECT	FF	255	11111111
STACK1	000	0	000000000000
STACK2	000	0	000000000000
STACK3	000	0	000000000000
STACK4	000	0	000000000000

※ 在 Debug 中，如需要修改，只需在欲修改的位置，点两下(如下图)后，输入数值，按下 **OK** 键即可。

Address: IODIR_A

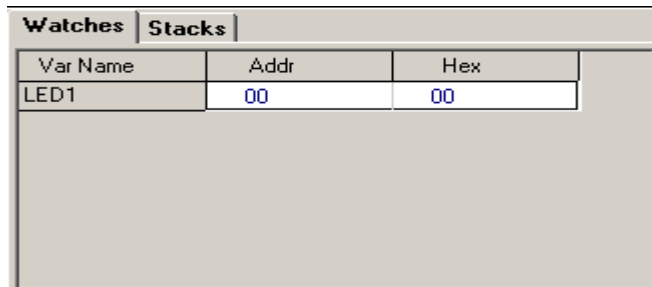
Dec

上图会显示要修改的 Register

【Watches/Stacks】

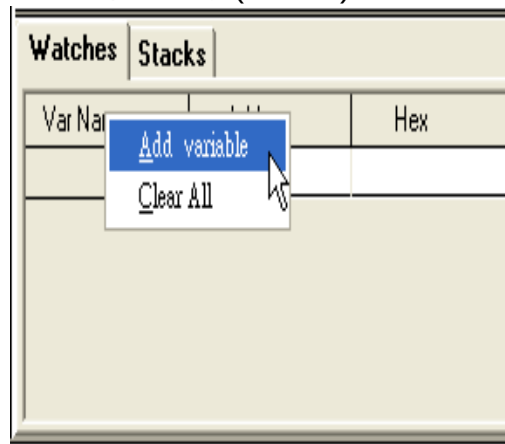
此窗口主要分成 Watches 和 Stacks。

1. Watches 是让使用者方便追踪程序中定义的变量数值的变化。

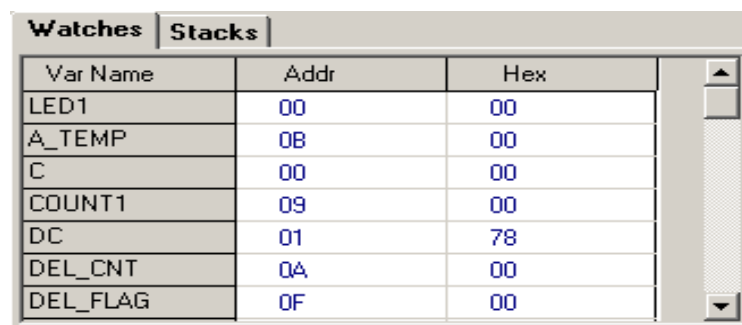
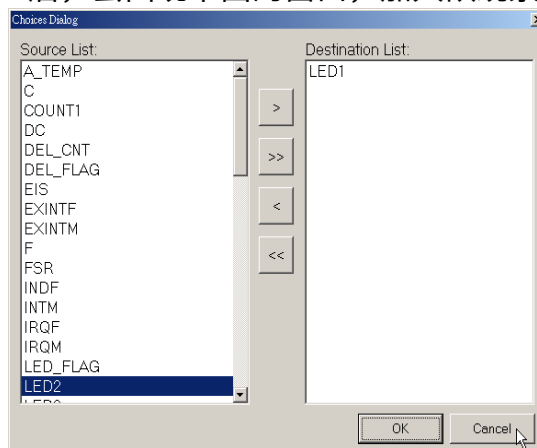


※ 加入方法如下：

- (1). 在 Var Name 的位置按鼠标的右键(如下图)。



- (2). 点选 Add variable 后, 会出现下面的窗口, 加入欲观察的变量, 按 **ok** 键。



2. Stacks: 是让使用者观察堆栈内数值。

Stacks		
Stack	Addr	Procedure
STACK1	000	
STACK2	000	
STACK3	000	
STACK4	000	

【操作讯息】

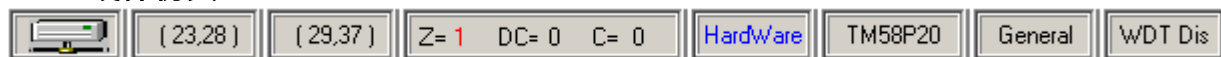
显示操作、编辑.....等相关讯息。

```
Building C:\Program Files\Tenx\TICE58\Sample\AP-TM58XX_01S\AP-TM58XX_01S.Hex , AP-TM58XX_01S.Bin
Build completed successfully.
```

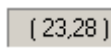
【其它讯息】

如下图所示

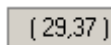
1. 硬件仿真



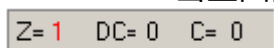
→ 代表和 TICE58 的主机联机。



→ 当主画面是在机器码的显示下，光标的位置。

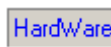


→ 当主画面是在程序代码的显示下，光标的位置。



→ 各个旗标缓存器的值。

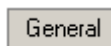
(旗标缓存器的意义，请参考 IC 的 DataSheet)



→ 目前是使用硬件仿真。



→ 目前所要模拟的 ic 型号。



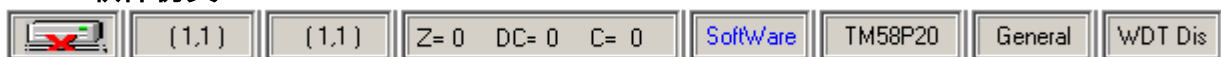
→ TM58P20 Configuration 中 TYPE 所设定的值。

(每颗 ic 都提供不同的 Configuration，所以请参考 ic 的 DataDheet)

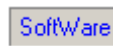


→ WatchDog 的功能关闭。

2. 软件仿真



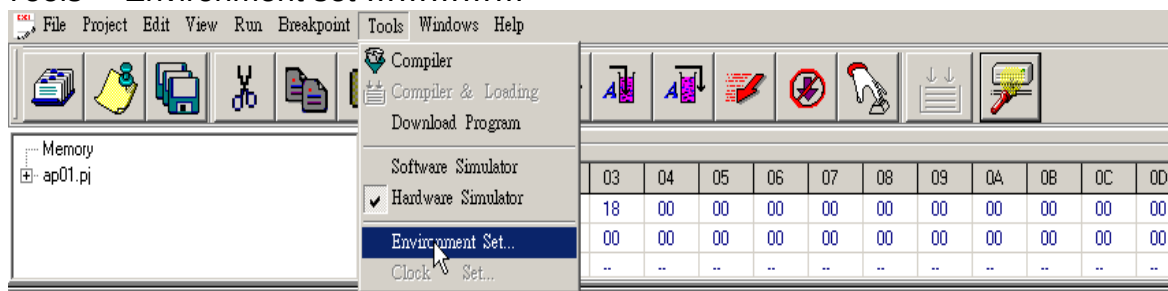
→ 无法和 TICE58 的主机联机。



→ 目前是使用软件仿真。

环境设定窗口

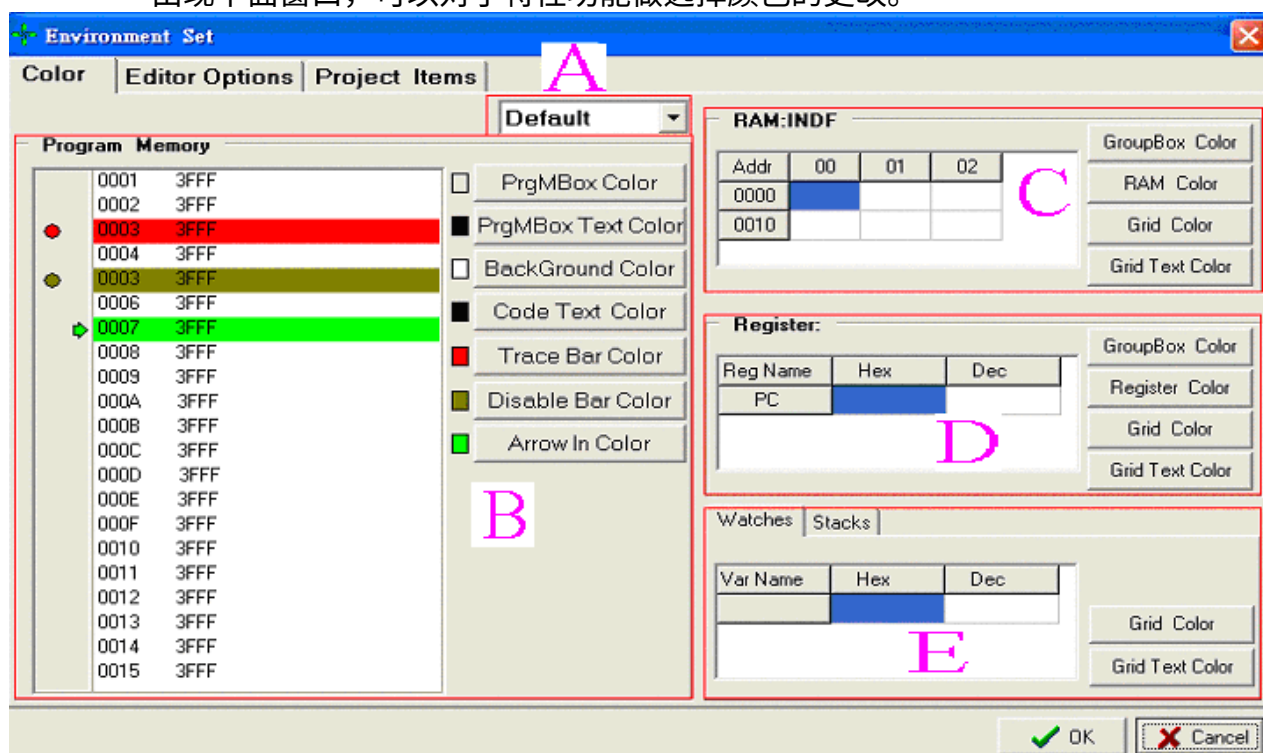
※ 在 Tools→ Environment set



1. Color

出现下面窗口，可以对于特性功能做选择颜色的更改。

出现下面窗口，可以对于特性功能做选择颜色的更改。



A: 设定主画面背景颜色

B: PrgMBox Color 设定外框颜色

PrgMBox Text Color 设定上方文字颜色

BackGround Color 设定程序框背影颜色

Code Text Color 设定程序颜色

Trace Bar Color 设定 enable 时断点的颜色

Disable Bar Color 设定 disable 时断点的颜色

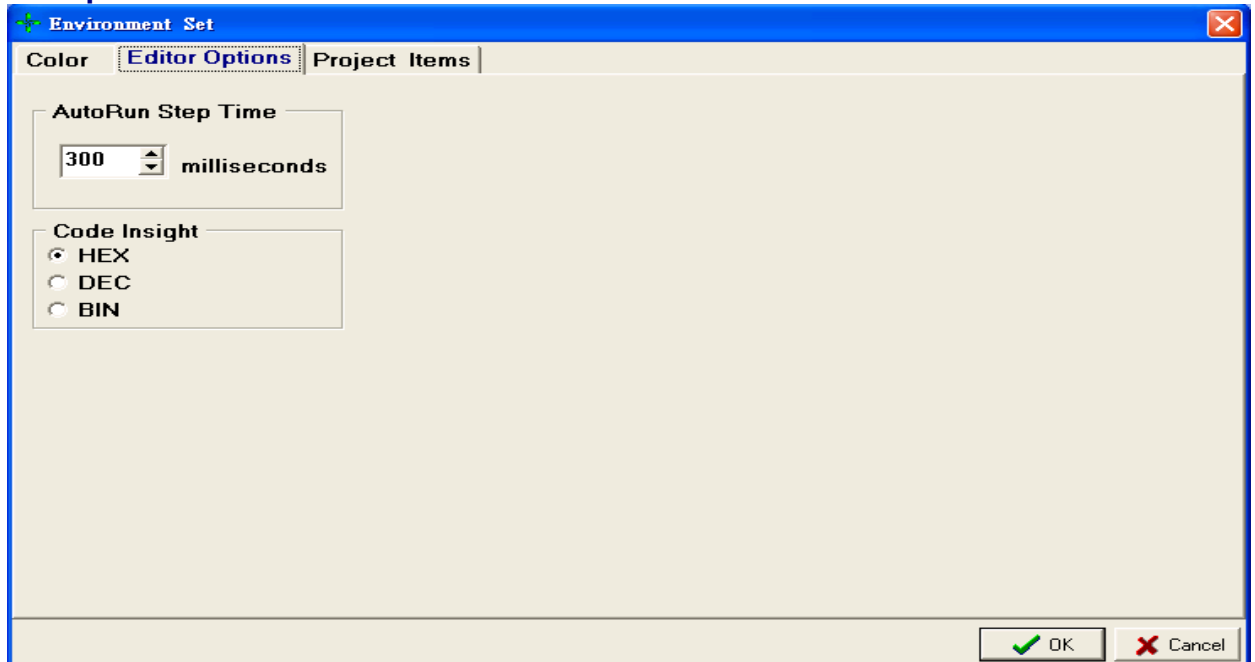
Arrow In Color 设定程序执行到某地址的颜色

C: 调整 RAM 窗口颜色

D: 调整 Register 窗口颜色

E: 调整 Watches 和 Stacks 窗口颜色

2. Editor Options



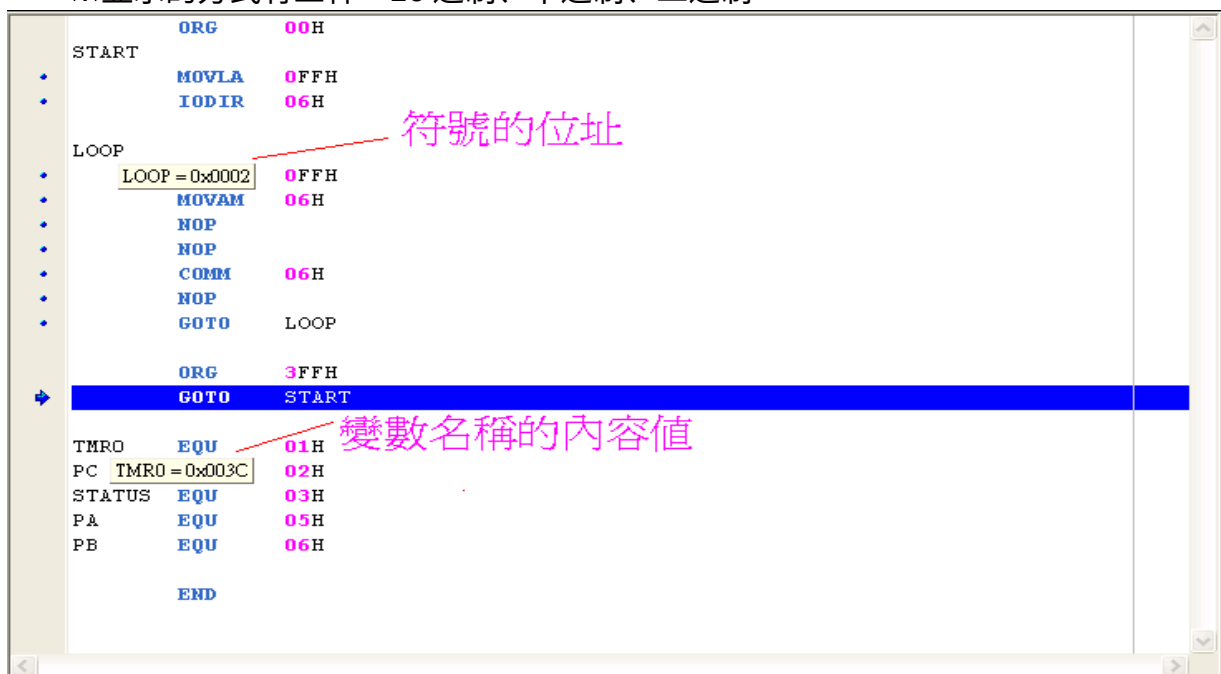
※AutoRun Setup Time:

设定当执行自动单步时，从一行指令跳至下一行指令的时间(单位:毫秒)。

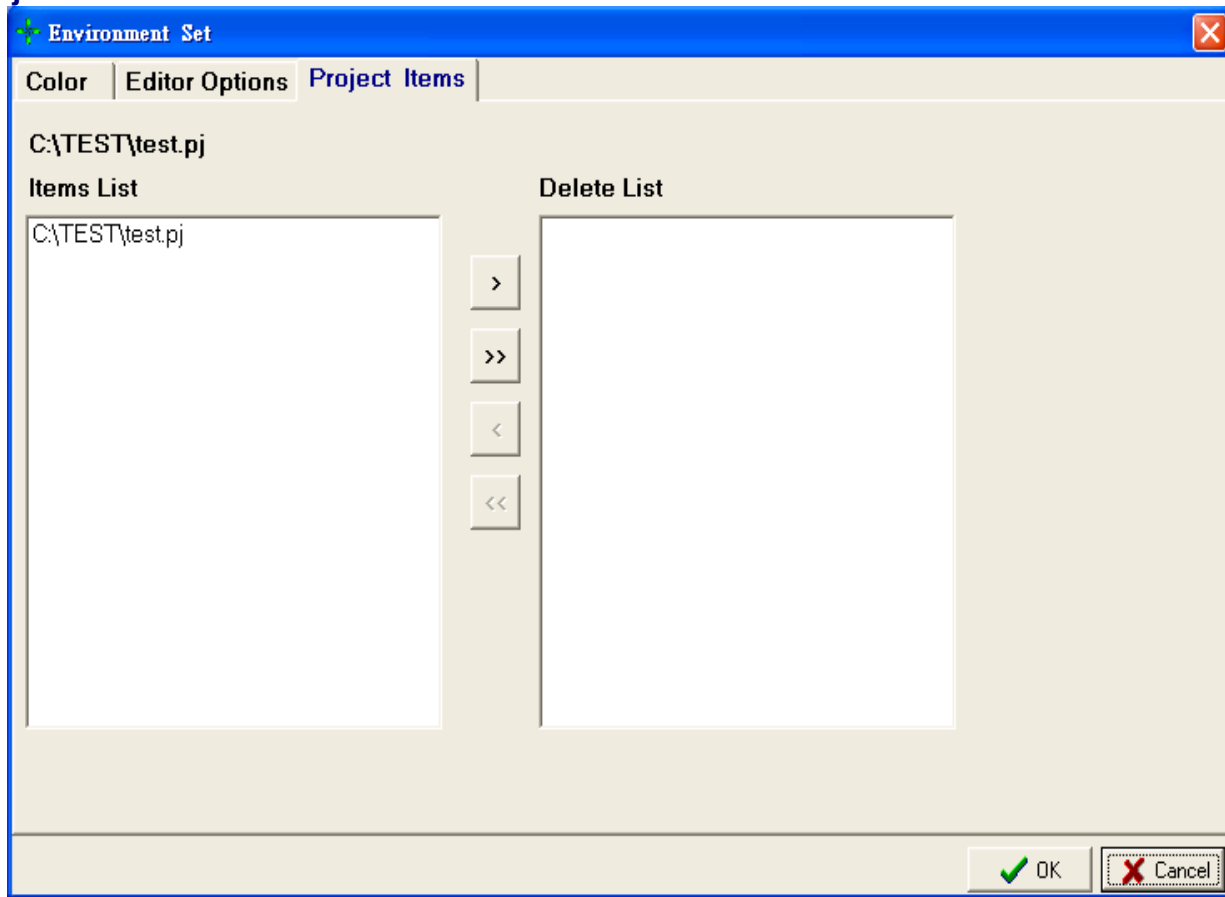
※Code Insight:

此选项只有在主画面是在来源窗口下有用，用于显示符号的地址和变量名称的内容值。

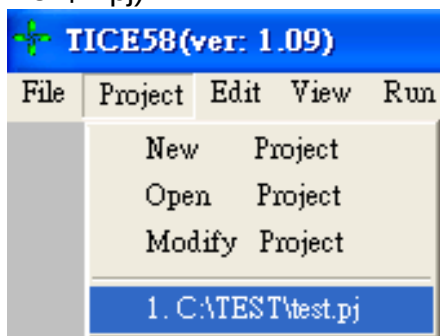
※显示的方式有三种：16 进制、十进制、二进制



3. Project items



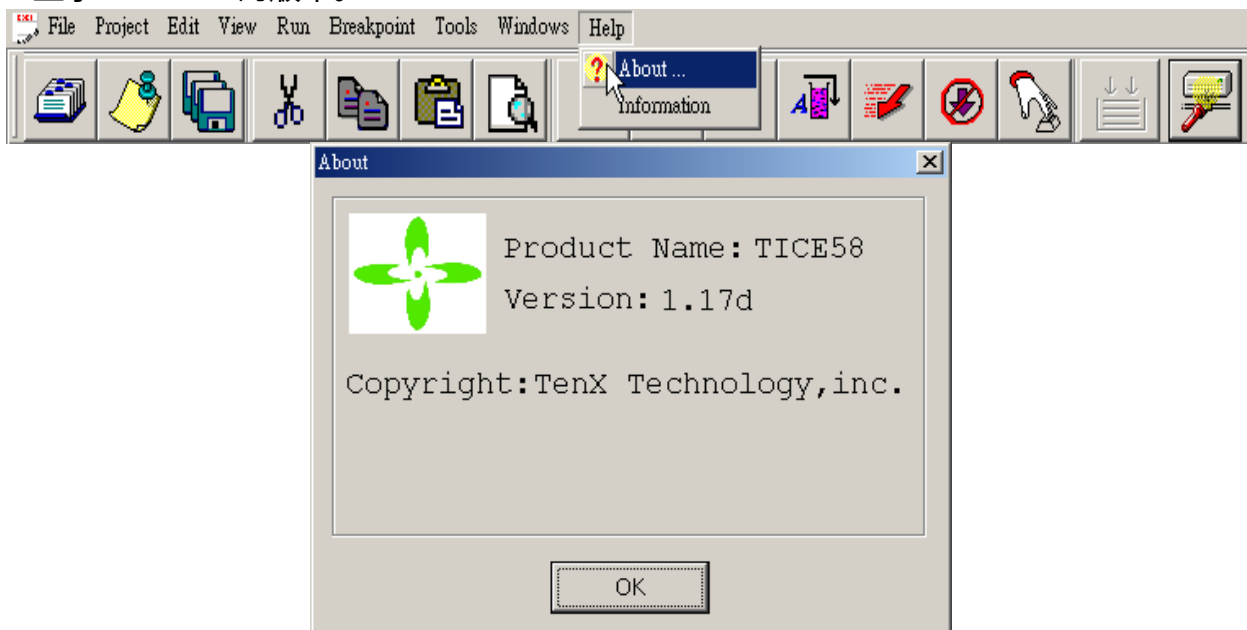
※ 此选项是在于当每次开启不同的 project 时，会在“Project”(如下图)中增加一快速开启的选择，当有些不想再用的*.pj 时，可由此处移除(在 Project 中最多能纪录 5 个*.pj)



帮助窗口

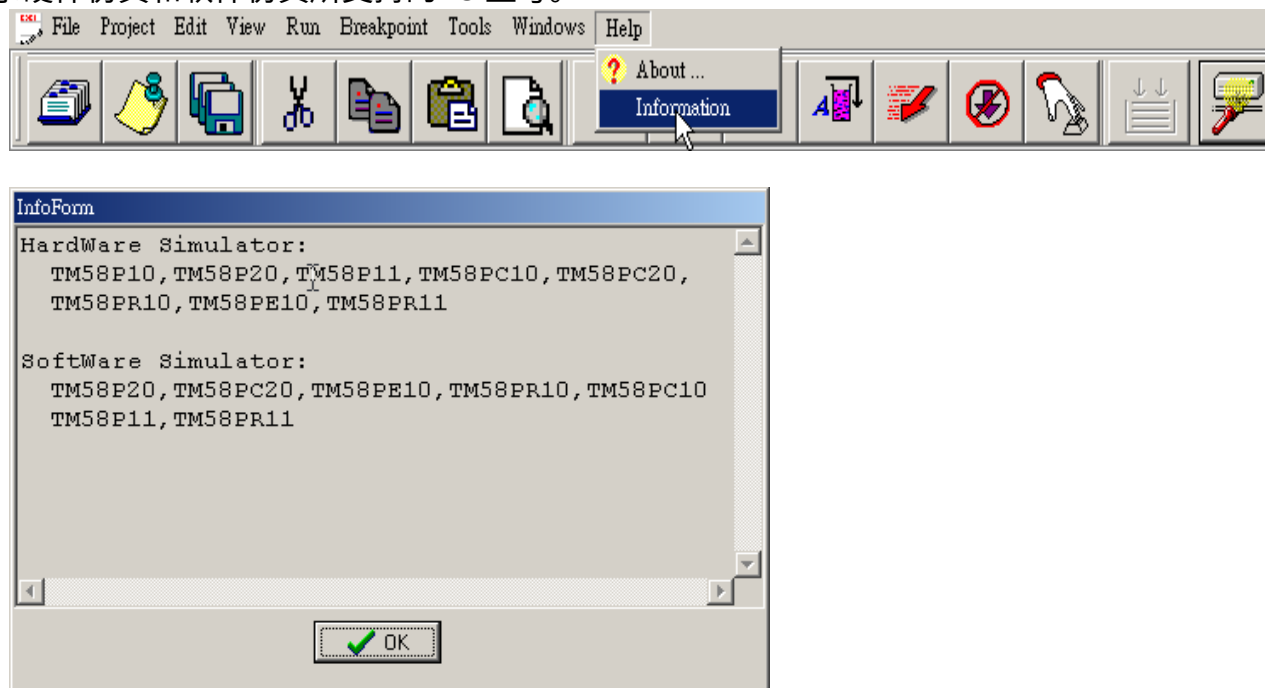
1. About

显示 TOCE58 的版本。



2. Information

显示硬件仿真和软件仿真所支持的 IC 型号。



使用技巧

断点的使用

为了方便使用者进行 Debug，TICE58 的软件提供数种让程序中断的方法，分别如下：

- 使用 F2 键
- 使用断点
- 使用 Run to Cursor
- 使用 Break Point List
- 主画面(断点发生时)

为了方便使用者进行 Debug，TICE58 的软件提供数种让程序中断的方法，分别如下：

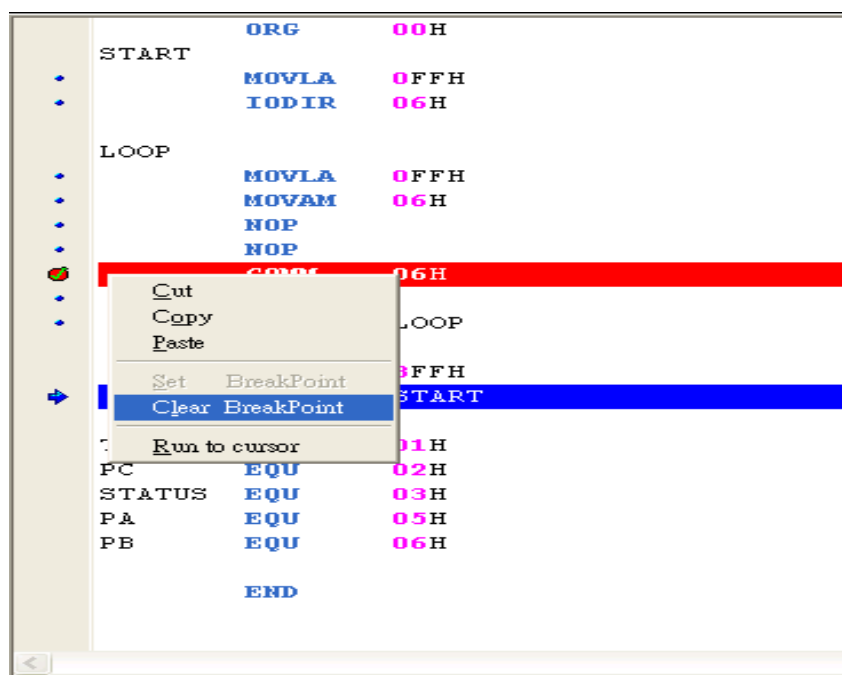
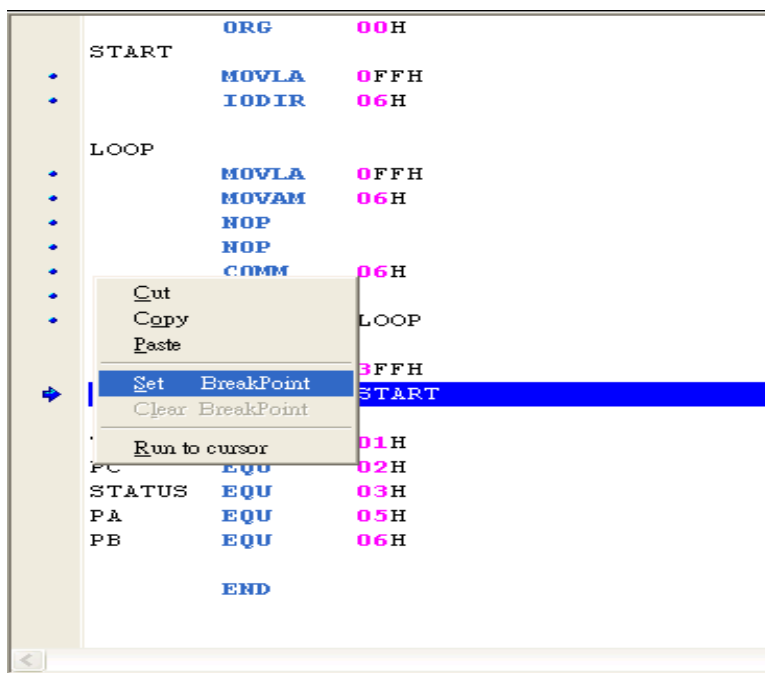
- 使用 F2 键
- 使用断点
- 使用 Run to Cursor
- 使用 Break Point List
- 主画面(断点发生时)

```

cleara
iodir portb ;將portb口設?輸出態
movam porta ;porta口輸出?0
movla 0fh ;
iodir porta ;將porta口設?輸入態
movla 05h ;
select ;tmr0採用內部時鐘計數,預分頻器分配給tmr0,
;分頻比?1:64
movla b'00100001' ;
movam wakeup ;設定porta,0口?外中斷輸入口
movla b'10000100' ;
movam irqm ;開放外中斷功能
movm tmr0,w ;
movam time_temp ;讀tmr0的值結果送入time_temp中
;
;
start:;主迴圈
call readtime ;讀tmr0時間處理副程式
btmsc irqm,exintm ;判斷外中斷功能是否關閉
goto start ;未關閉外中斷功能,返回start處執行
btmsc porta,0 ;外中斷功能關閉,判斷按鍵是否?起。
goto start ;按鍵未?起,返回start處執行
bsm del_flag,0 ;按鍵?起,置控制延時標誌del_flag
movla 05h
xoram del_cnt,w
btmsc status,z ;判斷延時消抖時間5*8.192ms=40.96ms 到否
goto start ;延時消抖時間未到,返回start處執行
clrm del_cnt ;延時消抖時間到,清del_cnt
btmsc porta,0 ;再次確認按鍵是否真的?起
goto start ;按鍵未?起,返回start處執行
bcm del_flag,0 ;按鍵確實?起,清控制延時標誌del_flag
bcm irqf,exintf ;清外中斷請求標誌位元
bsm irqm,exintm ;開放外中斷功能
goto start ;返回start處執行
;
;
;=====
;外中斷回應副程式
;=====

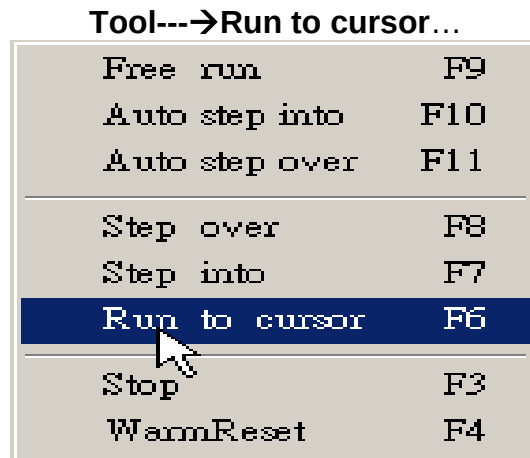
```

使用鼠标右键设置，在要设断点的位置按鼠标右键，选择“Set BreakPoint”设断点，如要取消，将鼠标移至要取消的地方按右键后选择“Clear BreakPoint”取消。



【使用 Run to Cursor】

Run to cursor 就是让程序执行到光标所在的位置。



※ 将光标移到要停止的程序代码上(红色箭头所指)，如下：

```

movla    b'00100001'    ;
movam    wakeup          ;設定porta,0口?外中斷輸入口 ←
movla    b'10000100'    ;
movam    irqm            ;開放外中斷功能

```

※ 执行 run to cursor 或是 F6，则程序执行到光标所在的位置就会停止，如下：
(红色箭头所指的程序将会变成蓝色)

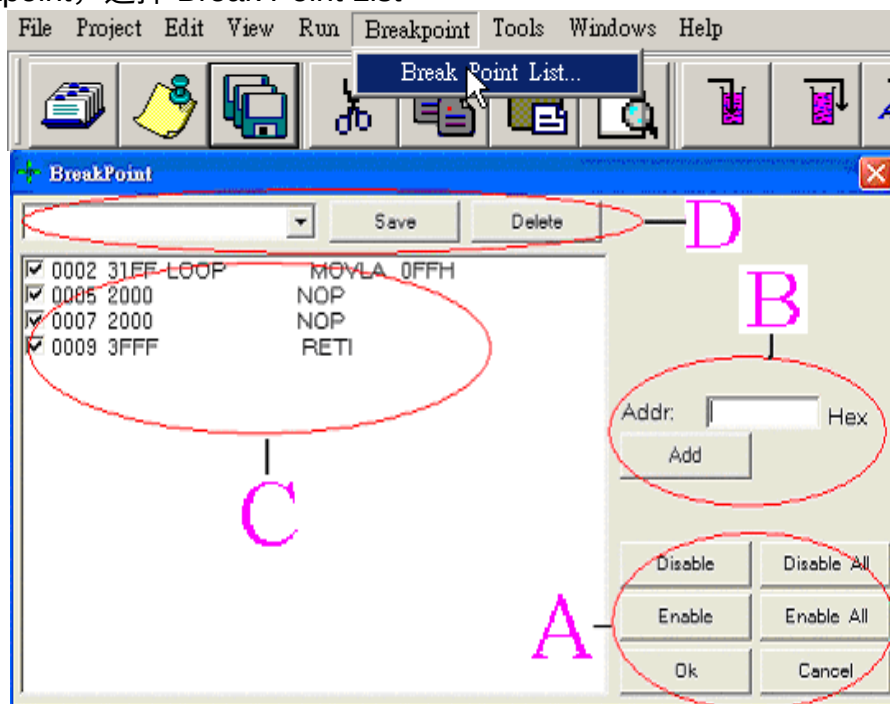
```

• movla    05h            ;
• select                                ;tmr0採用內部時鐘計數,預分頻器分配給tmr0,
;分頻比?1:64
• movla    b'00100001'    ;
• ← movam    wakeup          ;設定porta,0口?外中斷輸入口 ←
• movla    b'10000100'    ;
• movam    irqm            ;開放外中斷功能

```

【使用 Break Point List】

点选 Breakpoint, 选择 Break Point List :



A、控制用。

B、输入要设置断点的地址数值后按 。

C、所有设置断点地址数据。

D、将设置断点的位置储存起来，方便使用者设置多组不同的断点，储存的名称可随使用者自行定义，会将储存的数据至于 *.pj 的档案中。

【主画面(断点发生时)】

A、在来源文件的显示下：

```

• nop
• nop
• bsm      portb, led7      ;打開燈led7
• movla   01h              ;給led_flag付值01h,使其指向燈led1
• movam   led_flag         ;退出副程式
• ret
/
/
=====
;MAIN PROGRAM
;=====
main: ;初始化
• clrm    count1          ;清count1
• clrm    del_cnt         ;清del_cnt
• clrm    del_flag        ;清del_flag
• movla   02h              ;
• movam   led_flag        ;給led_flag付值02h,使其指向燈led2
• movla   01h              ;
• movam   portb            ;打開燈led1
• clra    ;
• iodir   portb            ;將portb口設?輸出態
• movam   porta           ;porta口輸出?0
• movla   0fh              ;
• iodir   porta           ;將porta口設?輸入態
• movla   05h              ;
• select  ;tmo0採用內部時鐘計數,預分頻器分配給tmo0,
/
• movla   b'001000001'    ;分頻比?1:64
/
• movam   wakeup          ;設定porta,0口?外中斷輸入口
• movla   b'10000100'    ;
/
• movam   irqm, w         ;開放外中斷功能
• movm    tmo0, w         ;
• movam   time_temp       ;讀tmo0的值結果送入time_temp中
/
/
start: ;主迴圈
• call    readtime        ;讀tmo0時間處理副程式
• btmsc   irqm, exintm    ;判斷外中斷功能是否關閉
• goto    start           ;未關閉外中斷功能,返回start處執行

```

※ 红色为设置的断点，灰色为设置失败的断点，蓝色为程序现在运行的位置。

B、在机器码的显示下：

0078	004E	2000	NOP
0079	004F	0B06	BSM PORTB, LED7
0080	0050	3101	MOVLA 01H
0081	0051	2088	MOVAM LED_FLAG
0082	0052	3F7F	RET
0083	0053	2189	CLRM COUNT1
0084	0054	218A	CLRM DEL_CNT
0085	0055	218F	CLRM DEL_FLAG
0086	0056	3102	MOVLA 02H
0087	0057	2088	MOVAM LED_FLAG
0088	0058	3101	MOVLA 01H
0089	0059	2086	MOVAM PORTB
0090	005A	2100	CLRA
0091	005B	2006	IODIR PORTB
0092	005C	2085	MOVAM PORTA
0093	005D	310F	MOVLA 0FH
0094	005E	2005	IODIR PORTA
0095	005F	3105	MOVLA 05H
0096	0060	2002	SELECT
0097	0061	3121	MOVLA B'001000001'
0098	0062	20A0	MOVAM WAKEUP
0099	0063	3184	MOVLA B'10000100'
0100	0064	20A1	MOVAM IRQM
0101	0065	2301	MOVAM TMRO, W
0102	0066	208D	MOVAM TIME_TEMP
0103	0067	3600	CALL READTIME
0104	0068	0521	BTMSC IRQM, EXINTM
0105	0069	3A67	GOTO START
0106	006A	0005	BTMSS PORTA, 0
0107	006B	3A67	GOTO START
0108	006C	080F	BSM DEL_FLAG, 0
0109	006D	3105	MOVLA 05H
0110	006E	2B0A	XORAM DEL_CNT, W
0111	006F	0103	BTMSS STATUS, Z
0112	0070	3A67	GOTO START
0113	0071	218A	CLRM DEL_CNT
0114	0072	0005	BTMSS PORTA, 0
0115	0073	3A67	GOTO START
0116	0074	0C0F	BCM DEL_FLAG, 0
0117	0075	0D22	BCM IRQF, EXINTF

※ 红色为设置的断点，绿色为程序现在运行的位置。

<注：颜色部份非固定，可由使用者另行定义>

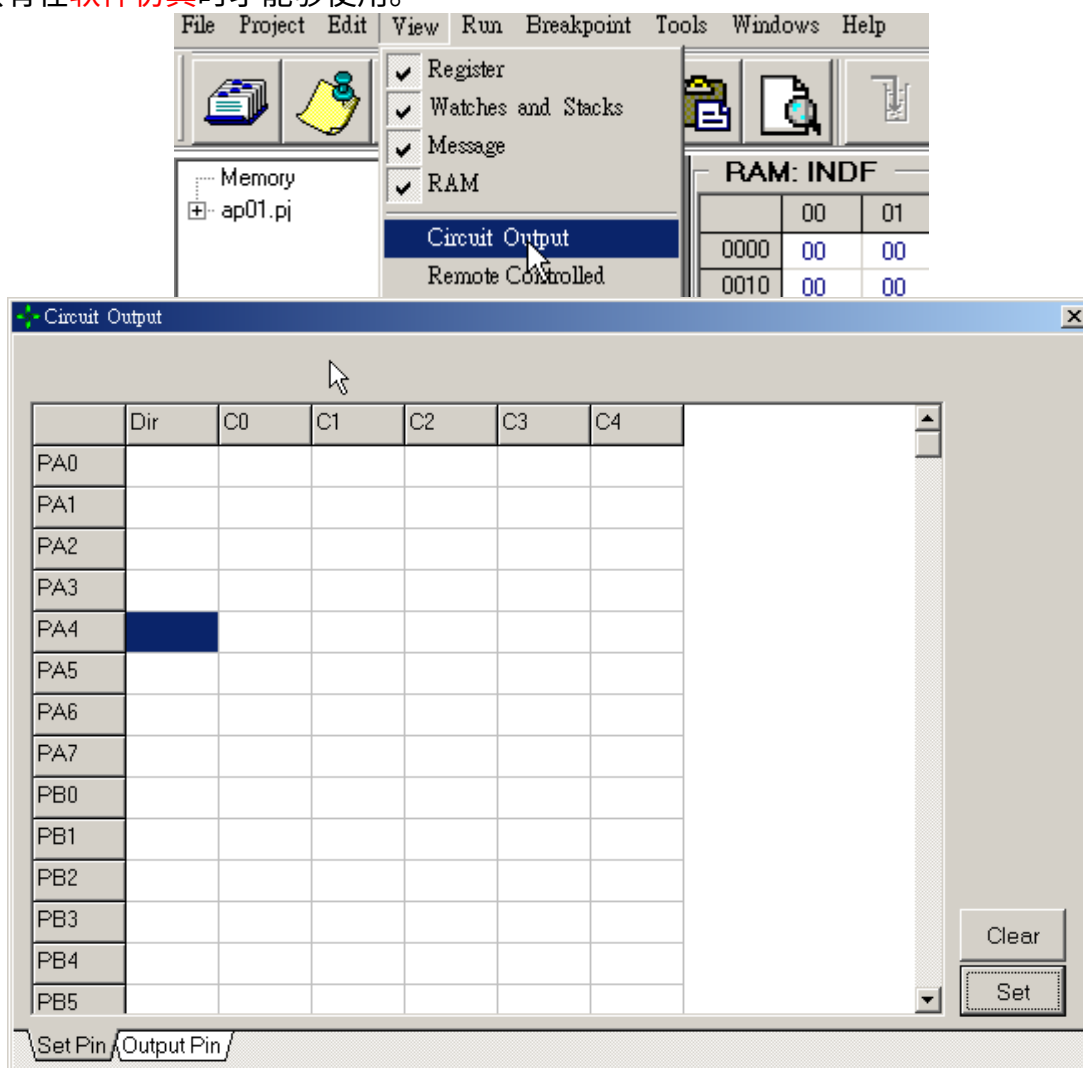
INCLUDE 的使用

使用include的指令，可缩短主程序对程序代码，方便阅读。

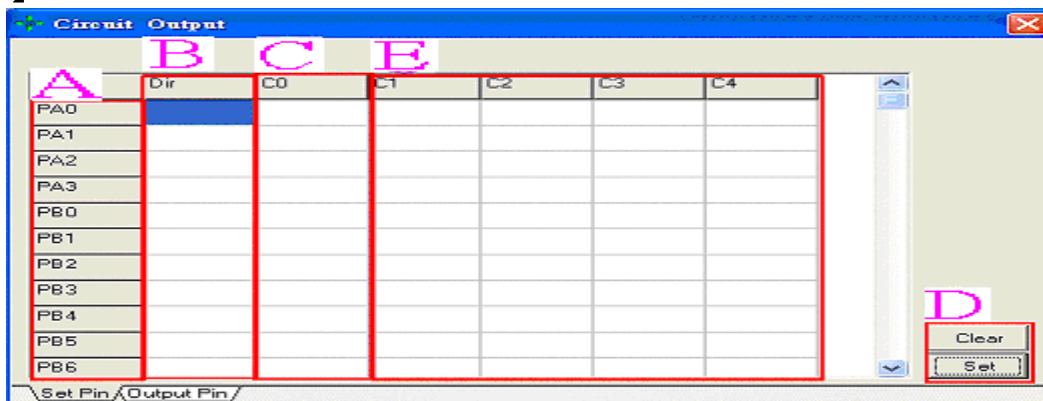
```
;osc1,osc2:clock pin(0)
;reseth0:pa0 be normal i/o pin(0)
;reseth1:reseth1(pa5) be system reset pin(1)
;fosc1~fosc0:nt(01)--8MHz
;lv1~lv0:don't use(11)
;-----
sec_acc      equ      09h
second       equ      0ah
count_buf    equ      0bh
;-----
#include "D:\tenx\1706\Tx58pe10.inc"
    org      3fff
    movla    0x7f
    org      3feh
    lgoto    tmr1_int
    org      000h
    movam    0x41
    lgoto    main
    org      010h                ;pc---16,page:0
;-----
led_table:
    addam    pc,f
    retla    00h      ,"0"
    retla    01h      ,"1"
    retla    02h      ,"2"
    retla    03h      ,"3"
    retla    04h      ,"4"
    retla    05h      ,"5"
    retla    06h      ,"6"
    retla    07h      ,"7"
    retla    08h      ,"8"
    retla    09h      ,"9"
    retla    0ah      ,"10"
    retla    0bh      ,"11"
    retla    0ch      ,"12"
    retla    0dh      ,"13"
    retla    0eh      ,"14"
    retla    0fh      ,"15"
    retla    10h      ,"16"
```

虚拟电路的使用

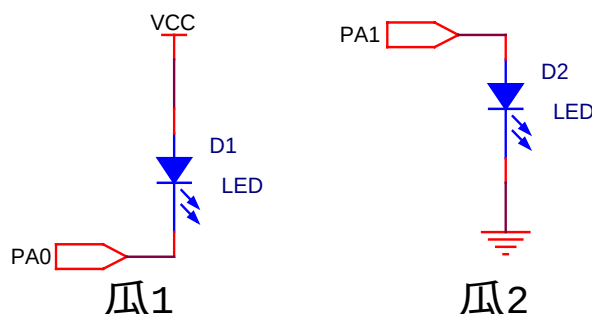
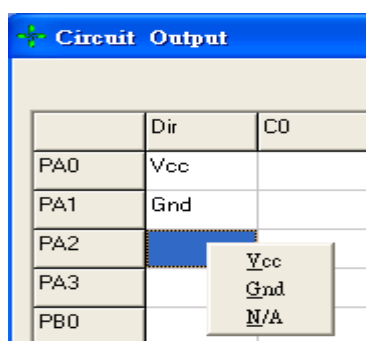
此功能只有在**软件仿真**时才能够使用。



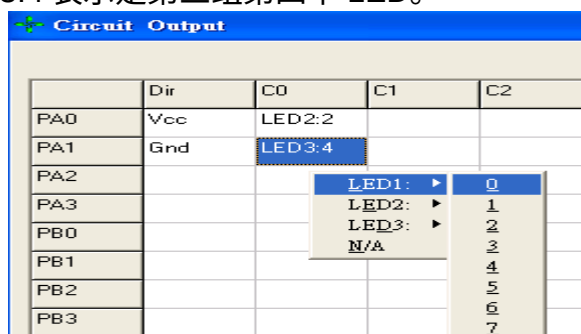
【设定 Pin】



- A、IO PIN 脚：此处会根据所选不同型号的 IC 而不同。
- B、IO 脚方向：设定仿真 LED 的接法，按鼠标右键选择，“N/A”选项表示不接，Vcc 与 Gnd 接法请参考图一与图二。



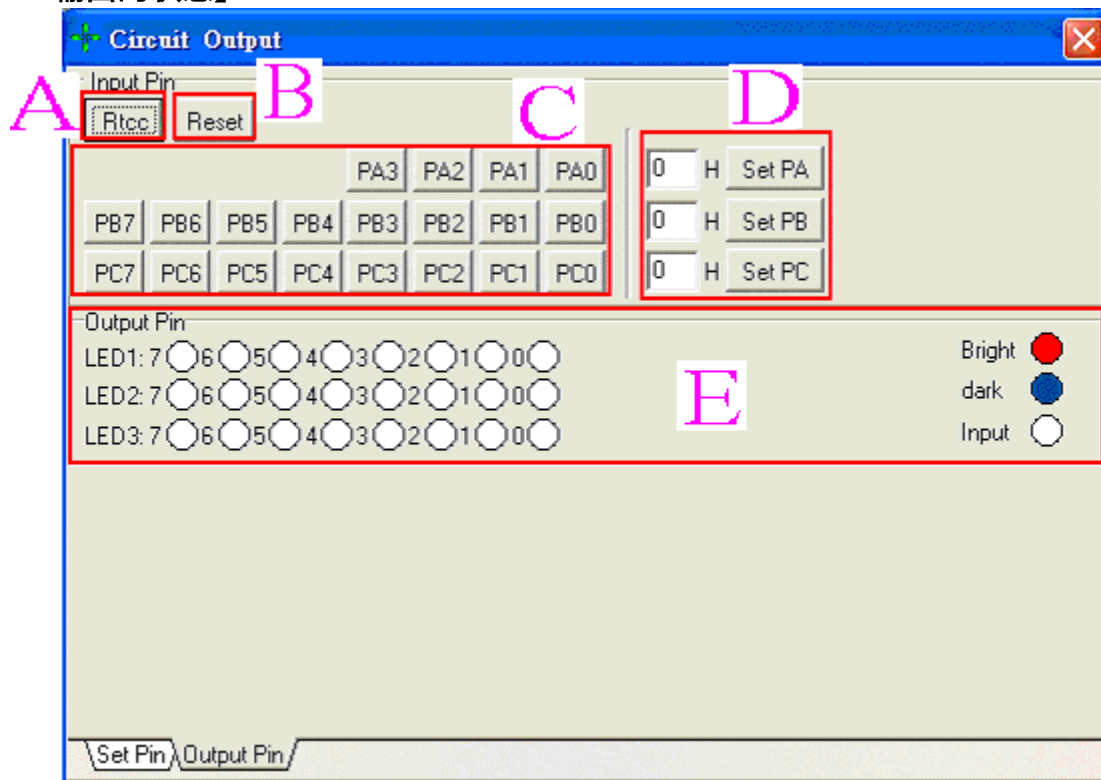
- C、C0：在要设定的位置按鼠标右键设定 LED 的位置，共有三组 LED 可以选择，每组 LED 有 8 个可选，如下图所示，LED2:2 表示是第二组第二个 LED；LED3:4 表示是第三组第四个 LED。



- D、当设定完成后，按 **Set** 键储存，如果设定完成后未按此键，则设定完成也不会有作用，要清除只需按 **Clear** 键就会将所有设定清除。

- E、此四格暂时保留未用。

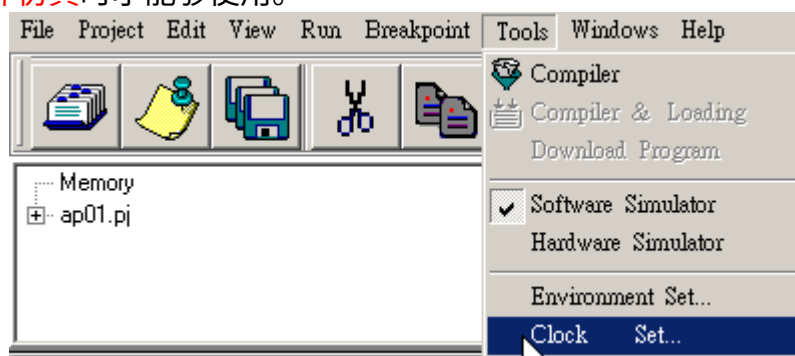
【观察 PIN 输出的状态】



- A、Rtcc：外部 clock 仿真信号，按此键一下 TMR0 加 1。
- B、Reset 键，按下此键，程序会 Reset 回初始状态。
- C、此框中的各键，是用于当程序有应用到 Wake-Up 或 Interrupt 时，当按下程序中所设定 Wake-Up 或 Interrupt 键时，就会产生相对应的动作，按键的有效与否是根据不同 IC 型号，而不是每个按键皆有效的，详细资料请参考各 IC SPEC。
- D、用于设定 I/O port “In/Out 数值”动作，I/O 的 In/Out 状态由程序设定，在白框中输入 16 进制值后按下 Set，将所输入的数值由对应的 I/O port 输出/输入。
- E、模拟 LED Output 动作，红色表示点亮，蓝色表示熄灭。

【频率设定】

此功能只有在**软件仿真**时才能够使用。



※点选频率设定会出现.

Frequency：此选项为设定仿真时的 Clock 频率

RTCC：为仿真外加 TRM Clock 频率，当没打勾表示为未使用

INRC Clock：当选择的 IC 有使用到 Internal RC 时，会多一窗口设定

Internal RC 值，若选择的 IC 无 Internal RC 的功能时，则不会出现。

注意事项

【说明1】

☐ **External Clock** 当选择internal / external 频率时，如选择内部振荡时，不需将其位置“√”。

☐ **External Reset** 当选择internal / external 频率时，如选择外部振荡时，需将其位置“√”。

【说明2】

在编译除错时，将会自动储存*.ASM档案，所以*.ASM档案不可设定成“只读”档，如果设定为“只读”档，将会在编译除错时，发生错误。

【说明3】

目前 TICE58 可以支持三种 POD。

1. TM58POD：支援 TM58P11、TM58PC05、TM58P10/PC10 和 TM58P20/PC20。



2. TM58POD V1.0：支援 TM58RC10、TM58PE10 和 TM58PR10。



3. TM58POD V3.0：支援 TM58PC10A 和 TM58PC20A。

