EzProg-I 工具使用手冊

(Version 4.5)

ICP DAS CO., LTD.

泓格科技股份有限公司

Warranty

All products manufactured by ICPDAS Inc. are warranted against defective materials for a period of one year from the date of delivery to the original purchaser.

Warning

ICPDAS Inc. assumes no liability for damages consequent to the use of this product. ICPDAS Inc. reserves the right to change this manual at any time without notice. The information furnished by ICPDAS Inc. is believed to be accurate and reliable. However, no responsibility is assumed by ICPDAS Inc. for its use, or for any infringements of patents or other rights of third parties resulting from its use.

Copyright

Trademark

The names used for identification only maybe registered trademarks of their respective companies.

License

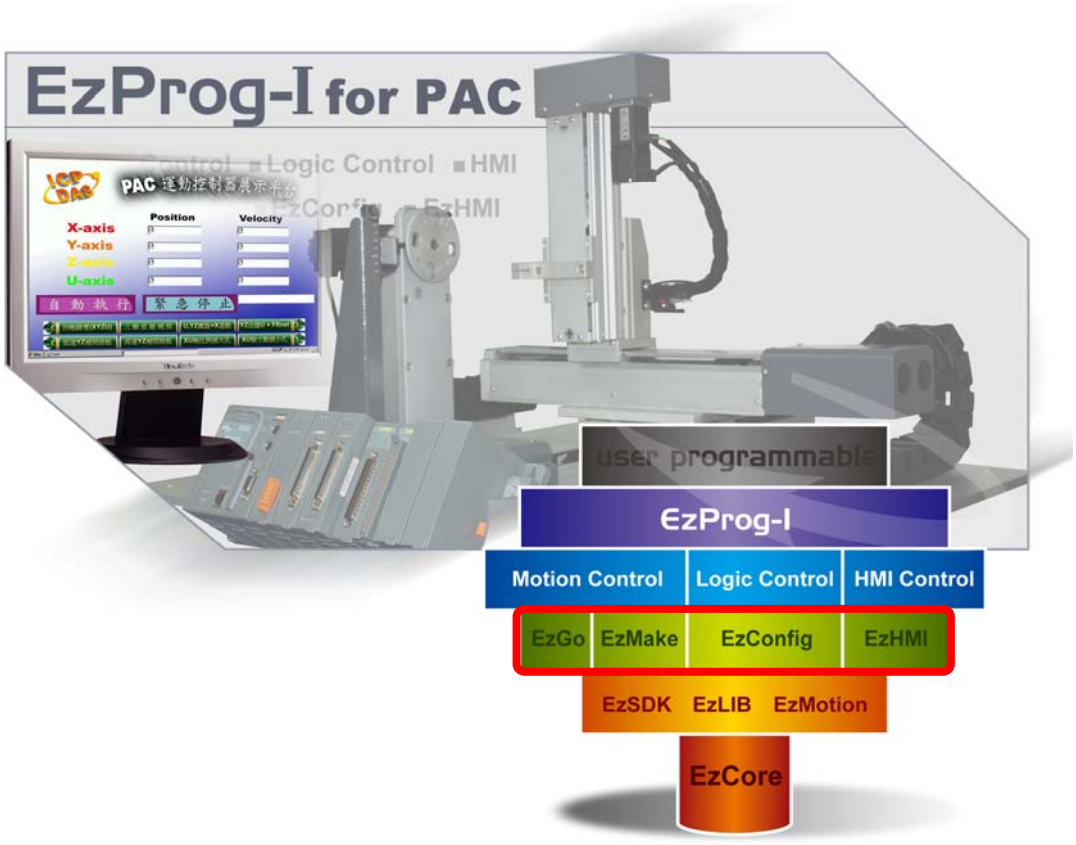
The user can use, modify and backup this software on a single machine. The user may not reproduce, transfer or distribute this software, or any copy, in whole or in part.

目錄

1前言

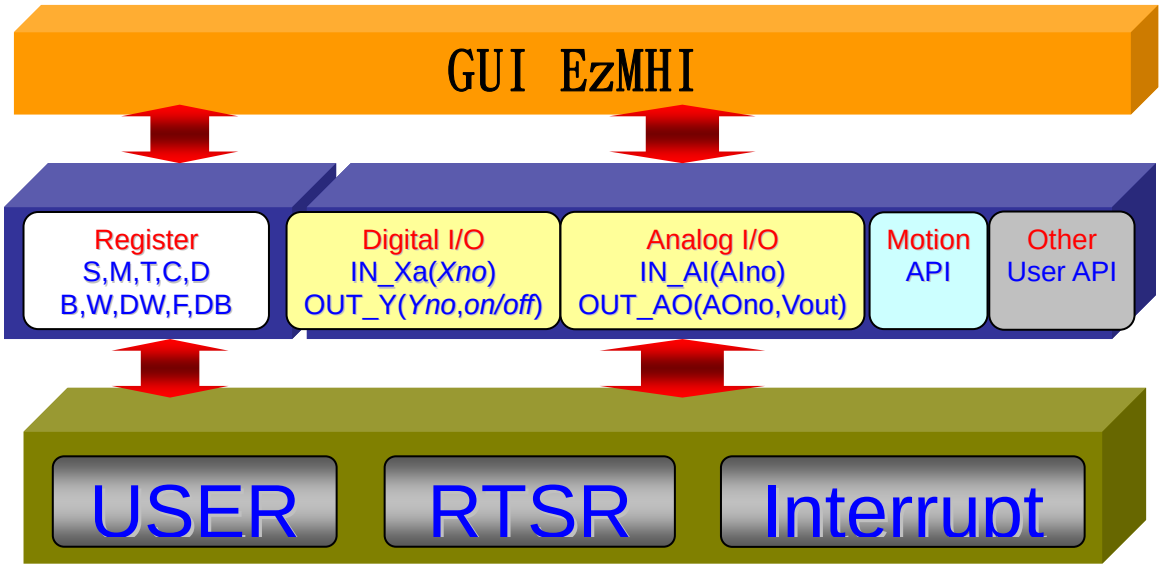
近年來工業控制器 PAC 的發展逐漸成熟，PAC 應用也越來越廣，使用者迅速增加所需功能，且自動化設備需求亦日益蓬勃，為協助新使用者對 PAC 控制器能輕鬆入門，因此 ICP DAS 泓格科技，設計此套 EzCore 軟體引擎，主要設計目標，是為了精簡工業自動化系統的開發時程。

EzCore 引擎主要工作方法目為在 eVC++ 平台輔以 EzProg-I 設計，來架構控制系統。整組套件包括了設定測試工具，HMI 輔助設計人機介面，我們架設了一個 Framework 包括了 IO、暫存器、斷電保持暫存器，利用 EzConfig 工具協助了系統規劃與設定與 IO 測試。Motion control 部分有 EzGo、EzMake 為協助測試與設計伺服馬達運動控制的工具。工程師可以更專注在控制邏輯的設計，並有效的縮短設計時程。



1.1 EzProg-I 使用架構說明

主要架構如下圖:



整個 EzProg-I 系統由三個部份組成:

1.上層 EzHMI:控制系統設計者可以透過 EzHMI ActiveX 物件的屬性表設定，就可以清鬆的設計人機操作介面，而這些 ActiveX 物件直接使用 EzCore 的 IO 與暫存器(變數) ，直接存取輸入與顯示狀態等等，其中還包括 UNICODE 多國語系支援。

2. 中間層 API: 軟硬體經過 EzConfig 的規劃後，EzProg-I(Framework)提供一系列統一的 API，系統開發設計者,很快就能熟悉與運用這些 API，大幅簡化以前客戶需針對不同的 DIO/AIO 模組使用不同的底層 API 的程序，很容易就可以存取到 EzCore 的變數與實體 IO 輸入輸出動作,例如 DO Y 的 on/off ，輸出 AO 的值，設定 D 暫存器的值等等。

3. 下層控制邏輯設計: 控制軟體實際運作程序，提供三種不同的控制軟體設計方法:

3.1 使用者自定程序:一般的執行程序，啟動後只執行一次。

3.2 定時執行程序: 類似 PLC 的 SCAN 執行方法，啟動後系統會按使用者設定時間定，定時執行。

3.3 硬體中斷程序:EzProg-I 經過適當的設定後，可以接受 DI 信號的中斷與 Motion 的中斷，並在此設計中斷的服務程序。

而這三種運作程序,一樣可以使用中間層的 API,很容易就可以存取到 EzCore 的變數與實體 IO 輸入輸出動作，例如 DO Y 的 on/off ，輸出 AO 的值，設定 D 暫存器的值等等，而漸漸達成控制系統的控制功能，期望系統開發者能達成同步與快速開發出易於管理與一定穩定品質的控制應用軟體。

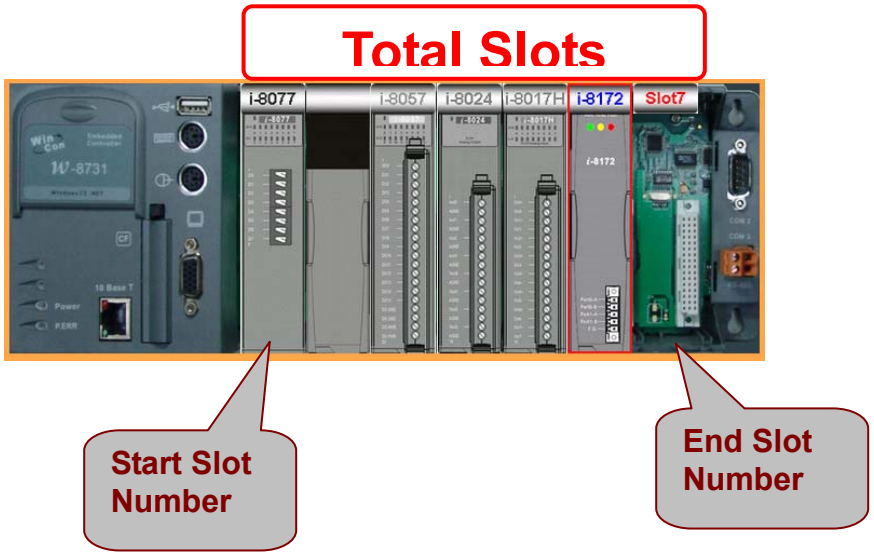
1.2 本手冊使用說明

PAC 系統路徑: 因 EzProg-I 在各平臺安裝位置會有所不同，為說明方便，我們會將 EzProg-I 的系統路徑統一稱為 EzProg_Path，在文中看見時請在此查閱。

WinCon W-8x8x-GM1: EzProg_Path = CompactFlash

PAC 支援 IO 模組數量: 因不同的 PAC 在 IO 模組數量也不同，為說明方便，我們會將不同的槽數列出如下，在使用時請在此查閱。

Model	Total Slot Numbers	Start Slot Number	End Slot Number
W-8731-GM1	7	1	7
W-8331-GM1	3	1	3



1.3 EzProg-I IO 與暫存器列表

EzProg-I IO 與暫存器的使用詳細說明，請參考本手冊 3.7 章節。

目前 EzProg-I 定義 IO 與暫存器的範圍如下表:

Register	Specification	Remarks	Register	Specification	Remarks
X	Local DI	X0000 ~ X0777	D	General	D0 ~ D4095
	Remote DI	X01000 ~ X07777		Retain	D4096 ~ D8191
Y	Local DO	Y0000 ~ Y0777	B	General	B0 ~ B1023
	Remote DO	Y01000 ~ Y07777		Retain	B1024 ~ B2047
AO	Local AO	AO000 ~ AO511	W	General	W0 ~ W1023
AI	Local AI	AI000 ~ AI511		Retain	W1024 ~ W2047
T	General	T0 ~ T299	DW	General	DW0 ~ DW4095
C	General	C0 ~ C511		Retain	DW4096 ~ DW8191
	Retain	C512 ~ C1023	F	General	F0 ~ F2047
M	General	M0 ~ M8191		Retain	F2048 ~ F4095
	Retain	M8192 ~ M16383	DB	General	DB0 ~ DB49
S	General	S0 ~ S8191	MSG	Retain	MSG0 ~ MSG249

下表列出 EzProg 中已指定用途的暫存器，請按照用途使用，而保留給系統特定用途的暫存器請勿任意使用以免造成不可預知的結果。

Register	Specification	Reserved	Register	Specification	Reserved
X	Local DI		D	General	D3900~D4095
	Remote DI			Retain	D8001~D8191
Y	Local DO		B	General	B700~B1023
	Remote DO			Retain	B2000~B2047
AO	Local AO		W	General	
AI	Local AI			Retain	
T	General		DW	General	
C	General			Retain	
	Retain		F	General	F1900~F1023
M	General	M7000~M8191		Retain	F4000~F4095
	Retain	M16001~M16383	DB	General	
S	General		MSG	Retain	

下表是 EzProg 中已指定用途的暫存器，請按照用途使用:

Register Number	Note
D8000	Muti-language
M16000	HMI:ColorEdit Input Control

2 EzHMI 介紹

2.1 EzHMI簡介

2.1.1 簡介

EzHMI是一套支援多國語系 HMI(Human Machine Interfcae)，以 UNICODE為標準WinCE OS的ActiveX，控制元件EzHMI 完全與 EzProg-I 整合，可以直接使用 EzProg-I 上既有的規劃資源(IO 與軟體暫存器)使用者可以利用它來開發，WinCE平台上相關監控與工業控制，應用系統的人機介面減少大量的 HMI 整合程序只要透過簡單的設定及些許程式碼的撰寫馬上就可以獲得，良好的而，人機介面ActiveX控制元件的特性運用也讓軟體的重複使用性更具有效率。

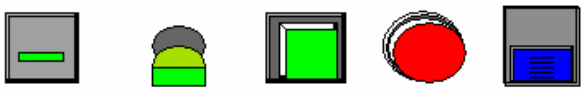
泓格科技推出PAC 的 EzHMI ActiveX控制元件的目地在於讓開發者可以更容易的做出實用透過，美觀的人機介面EzHMI而所需投入的資源也，程式碼部分也簡化了，開發者的系統不僅是畫面美化了，，更少您的心力可以專注於控制程序撰寫。

2.1.2 EzHMI包含元件

目前EzHMI Basic提供下列項物件(如下所述)。提供客戶使用，

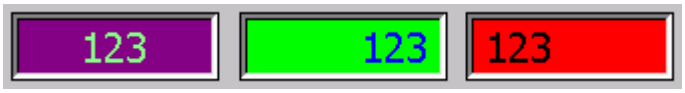
- **LED:** 提供使用者量測或工控應用上相關燈號或發光二極體(LED)。顯示與設定功能

- **SWITCH：** 提供使用者工業控制。應用上相關開關顯示與設定功能



- **Label：** 提供使用者顯示或監控各類工業控制數值。

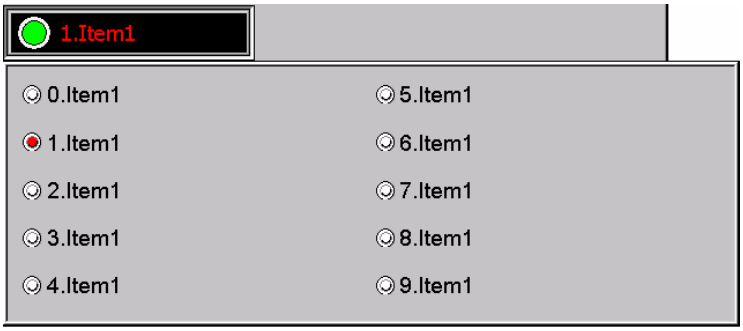
- **ColorEdit：** 提供使用者輸入介面，用於輸入各類工業控制數值。



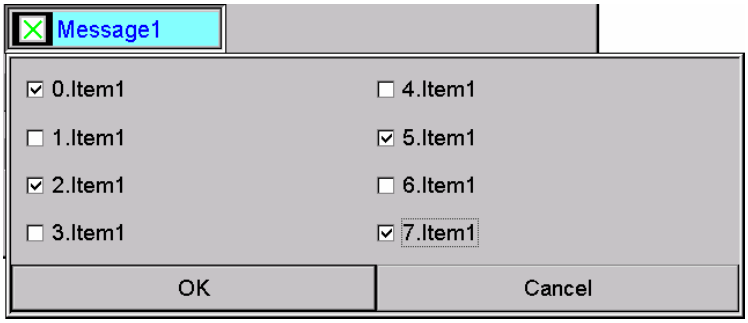
- **ButtonST：** 提供使用者帶圖示的按鈕，用於各類工業控制設定用途。

- **Image：** 提供使用者顯示與動態更換 BMP 圖檔。

- **ColorRadio：** 提供使用者多選一的項目選擇功能。

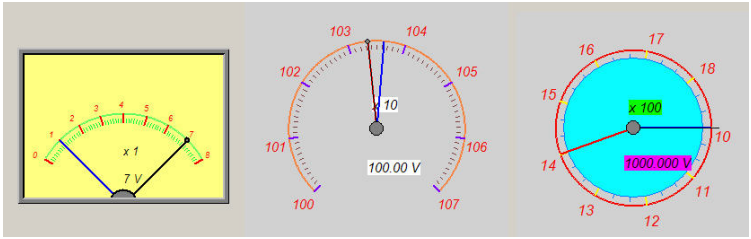


- **ColorCheck：** 提供使用者複選的項目選擇功能。

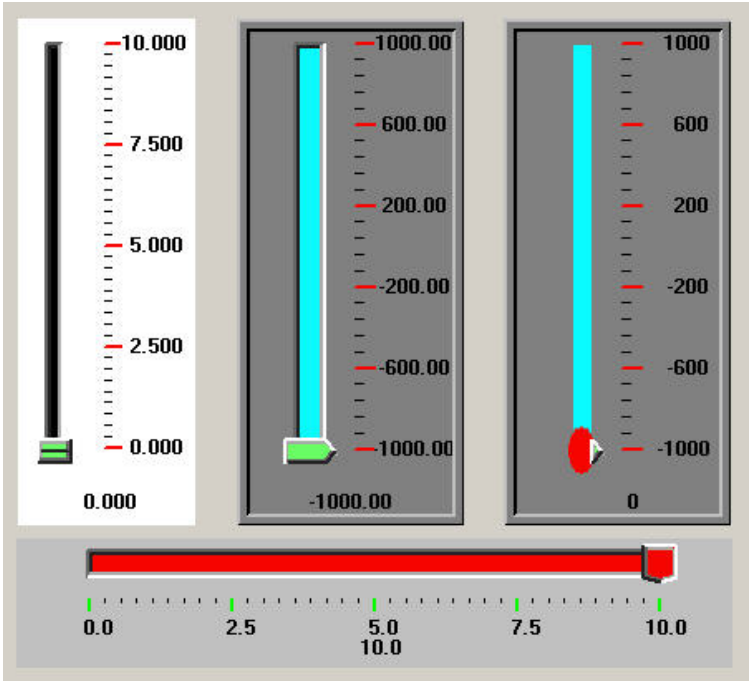


- **Position：** 提供使用者顯示 Motion 的指令位置實際位置速度等狀態。

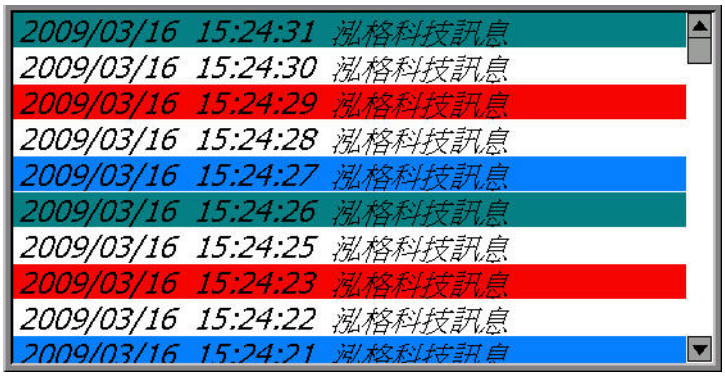
- **EzKnob：** 提供使用者輸入及顯示介面，用於輸入或監控各類工業控制數值。



- **EzSlider：** 提供使用者輸入及顯示介面，用於輸入或監控各類工業控制數值。



- **EzList：提供使用者以清單方式顯示或監控各類工業控制數值**



2.1.3 與 EzHMI 有關的特殊暫存器

D8000 : EzProg-I 提供了八國的 unicode 語系，讓使用者切換，
程式中可以利用 D8000(0~7)，動態切換所對應的文字。

M16000: 當 M16000 設為 true 時，ColorEdit 編輯時將彈出軟體輸入對話框，
當 M16000 設為 false 時，ColorEdit 編輯時將以鍵盤為輸入界面。

2.2安裝EzHMI

2.2.1 系統需求

- **程式開發端**
  作業系統 : WinXP
  硬體平台 : 一般桌上型電腦或筆記型電腦
  程式開發工具 : eVC++ 4.0 SP4

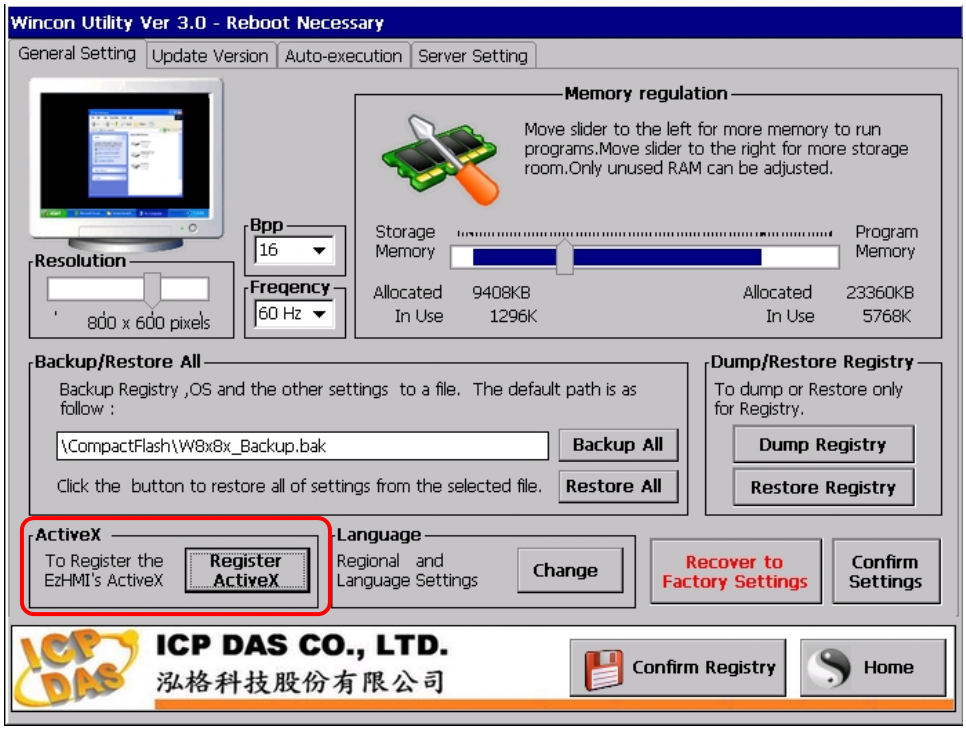
- **程式執行端**
  作業系統 : WinCE 5.0
  硬體平台 : 泓格科技WinCon/PAC 控制器

2.2.2 安裝程序

- **程式開發端(PC)**
  請參閱 EzProg-I_Getting Started 第三章 安裝 EzProg-I 在 PC 的開發環境

- **程式執行端 (PAC 一般出貨時都已經註冊過了，如果因使用者發現無法正常顯示才有需要手動註冊)**

可由 WinCon Utility 中註冊 :
開啟 WinCon Utility ，點擊[Register ActiveX] 按鈕會自動執行註冊



或是手動在PAC "\EzProg_Path\EzProg-I\EzHMI"，目錄確認下述檔案都存在:

RegistryOCX.bat、Regsvrce.exe、RegFlush.exe、LED.ocx、Swich.ocx、Label.ocx、ColorEdit.ocx、ButtonST.ocx、Image.ocx、ColorRadio.ocx、ColorCheck.ocx、Position.ocx、EzKnob.ocx、EzSlider.ocx、EzList.ocx

執行RegistryOCX.bat即可完成ocx在PAC上的註冊(如下圖) 。

若.ocx將泓格科技請，因版本更新需作更換或安裝時之"Machine Automation"CD內"

ezprog_i\xxxx\xxxx\EzProg-I\EzHMI"目錄內的上述檔案，複製於PAC "\EzProg_Path\EzProg-I\EzHMI"目錄下，

執行上述第二項的動作即可完成新版ocx在PAC。的註冊

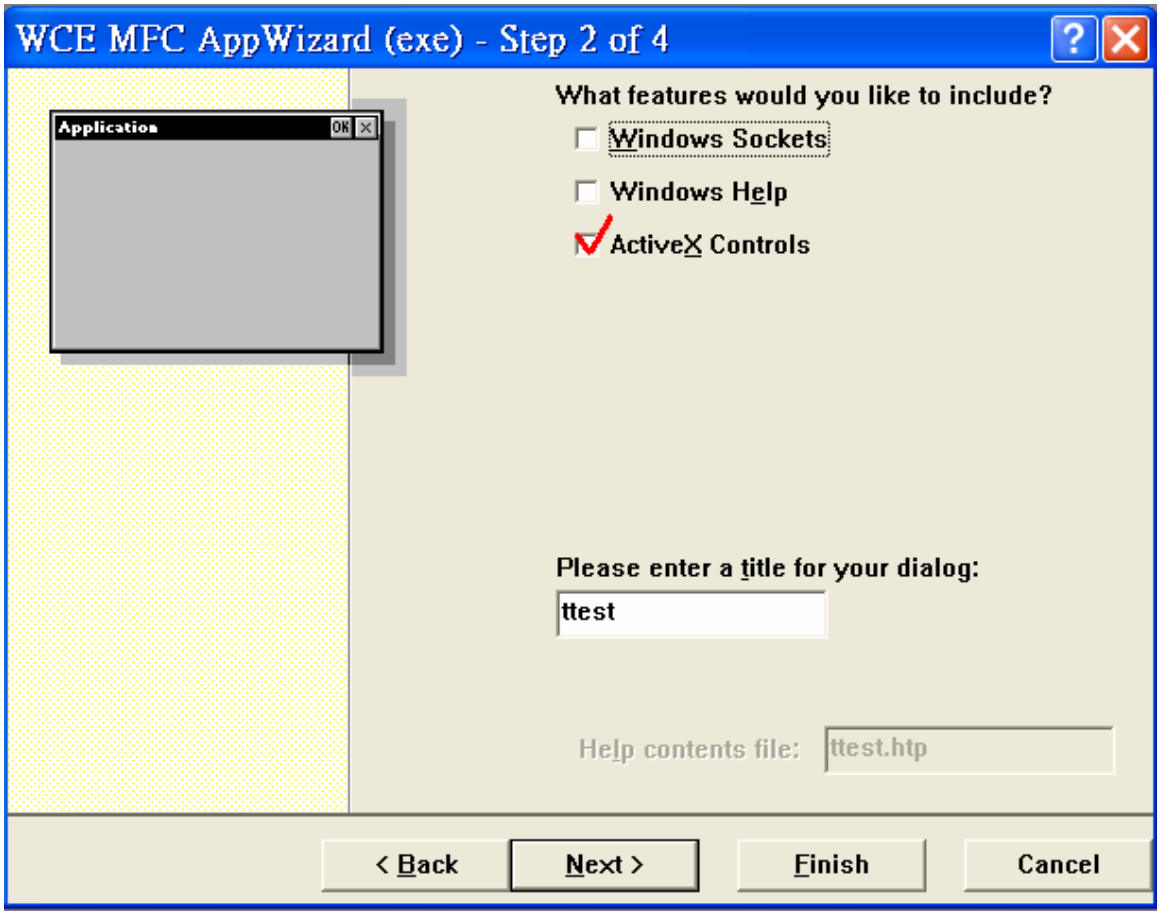
2.2.3 PC安裝檔案說明

- 程式開發端
  \安裝目錄\ICPDAS\EzProg-I\OCX：提供eVC++所使用的EzHMI ActiveX 。控制元件
  \安裝目錄\ICPDAS\ EzProg-I\Sample：eVC++ 。範例程式
  \安裝目錄\ICPDAS\ EzProg-I\Templat：開發樣版。程式
  \安裝目錄\ICPDAS\ EzProg-I\Manual：手冊位置。
  \安裝目錄\ICPDAS\ EzProg-I\Tutorial：指導。範例程式

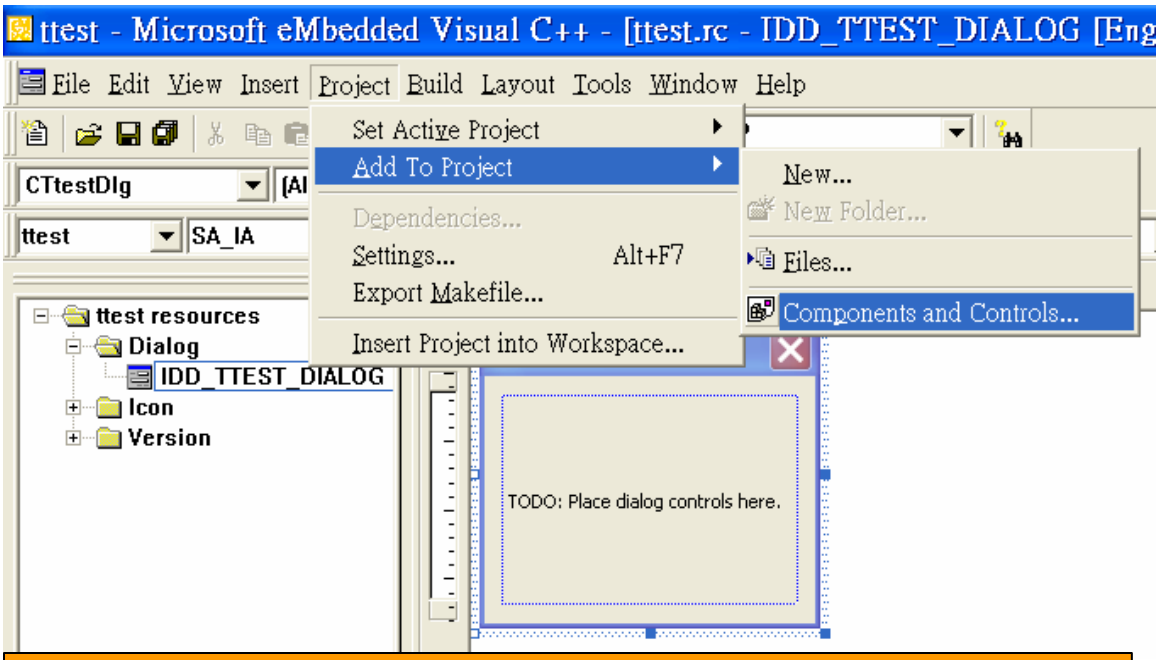
2.3加入ActiveX元件到開發專案中

如果您使用 EzTemplate 本章節可以略過，當您要自行建立專案時請按下列設定新的專案:

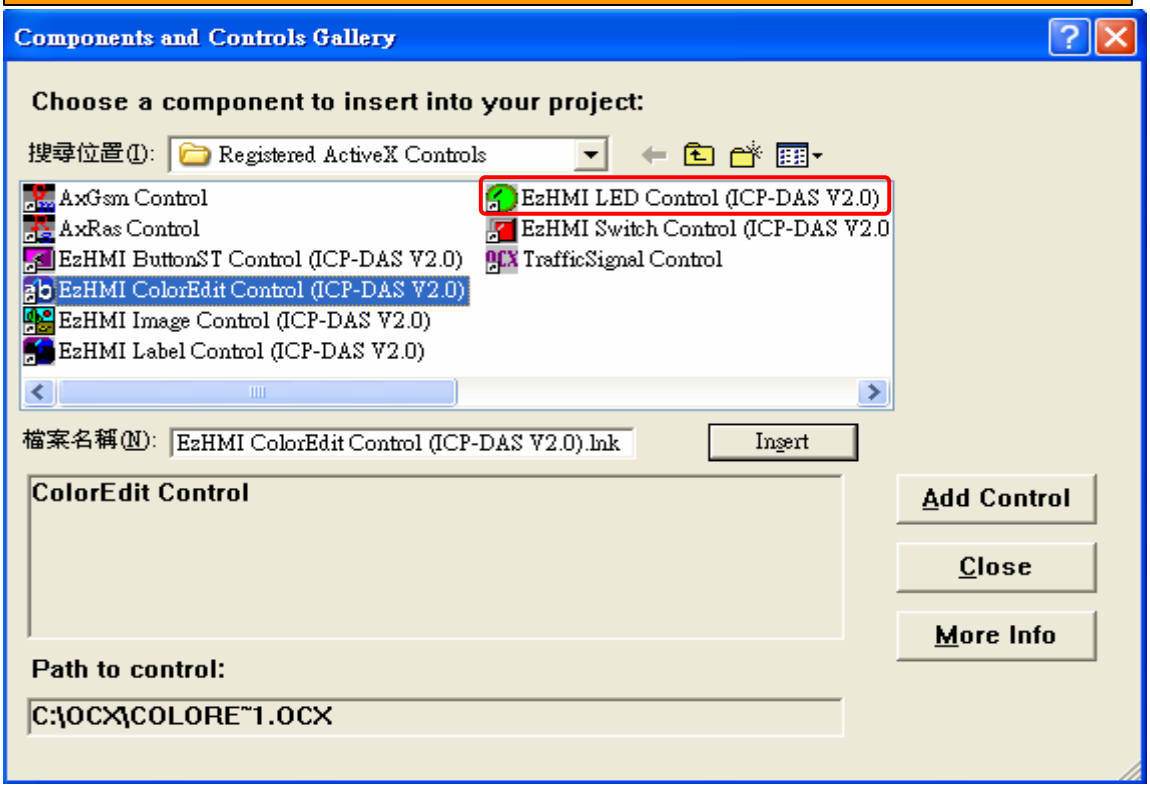
首先,如同平常開發MFC專案時開啟一個新的Project但是。,得勾選加入記ActiveX元件



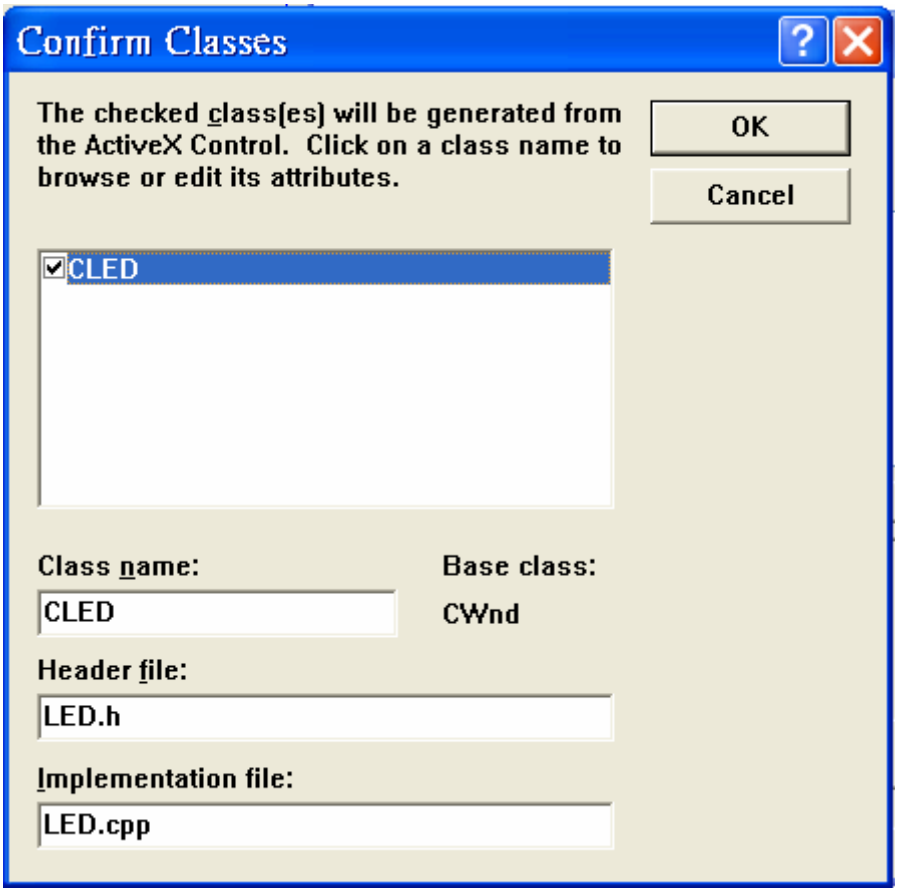
接著就是將ActiveX元件加入,依序選擇Project->Add to Project->Components and Controls Gallery

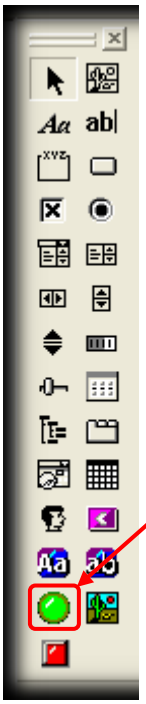


再來點選Add Control,然後找到OCX的位置選擇要加入的OCX(LED.ocx & Switch .ocx)加入成功後會出現如紅框的。Control,繼續依次點選control後按Inset



按下Insert後出現下圖,直接按下OK



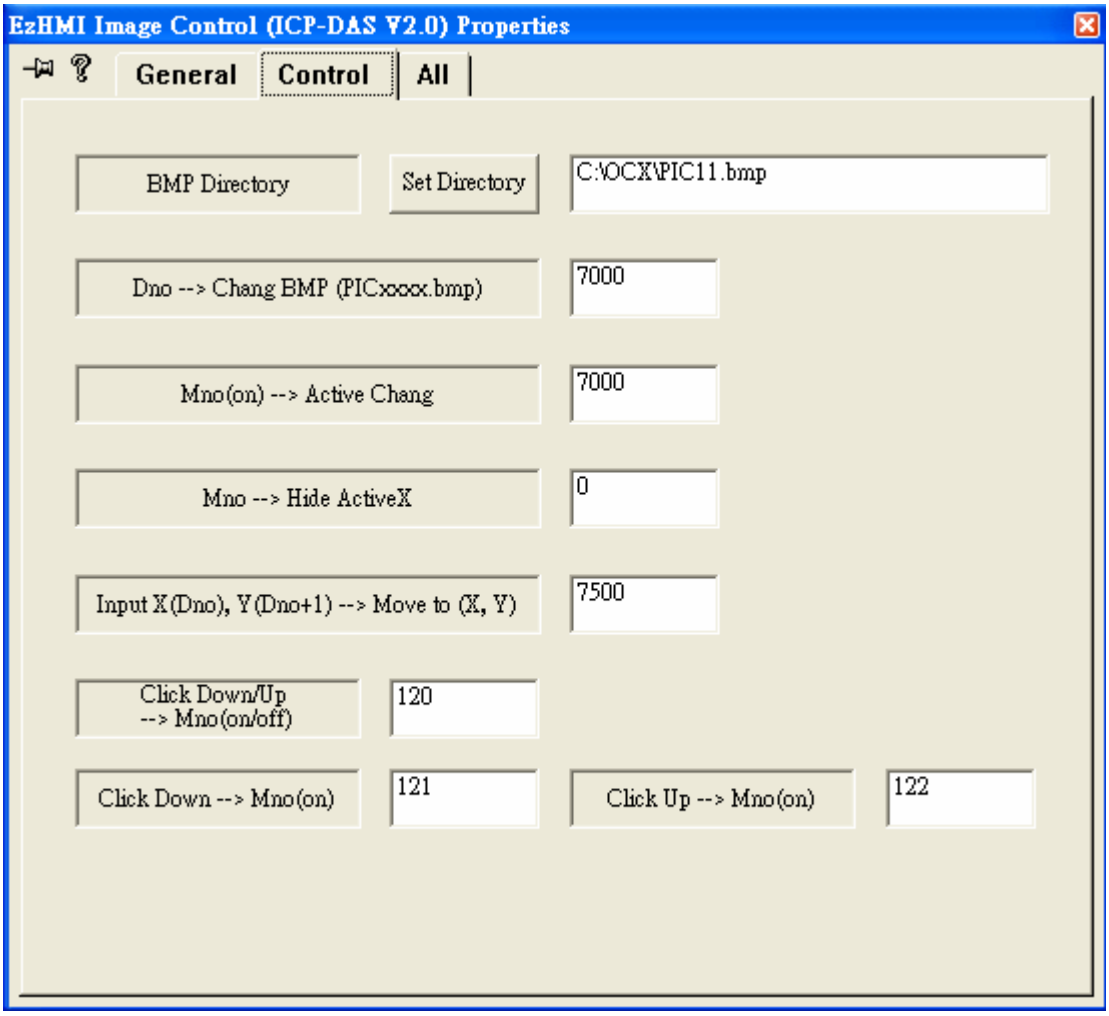


這樣就完成ActiveX元件加入，
使用來最後就依設計者需要。

2.4 EzHMI 控制元件的使用方法

一般而言對ActiveX可透過屬性頁的方式，控制元件屬性的設定方式設定：描述如下

當使用者將ActiveX件放至於所使用的程式編輯環境內時控制元(如：eVC++ 4.0之rc檔編輯器的Dialog Box物件)當使用者想改變此，您即會看到此物件顯示於表單上，ActiveX只需在該元件上使用滑鼠，控制元件的屬性時(Mouse)則該，的右鍵進行點選ActiveX下圖即以，此時即可進行屬性值的更動，控制元件的屬性頁即會顯示EzHMI之IMAGE。元件的屬性頁



2.5 EzHMI控制元件基本應用

2.5.1 直接顯示與控制 IO 狀態

EzHMI 不用寫介面程序經過簡單設定後就可以直接顯示與控制 IO 狀態，例如:

LED: 顯示 DI or DO 的 On/Off 狀態
SWITCH: 可以直接控制 DO On/Off
Lable: 可以直接數值顯示 AI/AO 狀態
EzKnob: 可以直接圖錶顯示 AI/AO 狀態
EzSlider: 可以直接 Mouse 控制 AO 輸出狀態
ColorEdit: 可以直接輸入控制 AO 輸出狀態
Position: 可以直接顯示 Motion 軸位置，速度等等。

2.5.2 直接顯示 EzProg-I 內部暫存器狀態

EzHMI 不用寫介面程序經過簡單設定後就可以直接顯示內容暫存器狀態，例如:

LED: 顯示 M， T(timer)， C(counter) 等接點的 On/Off 狀態
SWITCH: 顯示 M 接點 的 On/Off 狀態
Lable: 可以直接數值顯示 F(float)， D(long)， B(Byte)， MSG(Text) 等狀態
EzKnob: 可以直接圖錶顯示 F(float)， D(long) 等狀態。
EzList: 可以直接顯示 MSG(Text) 狀態

2.5.3 直接輸入到內部暫存器狀態

EzHMI 不用寫介面程序經過簡單設定後就可以直接輸入暫存器內容狀態，例如:

SWITCH: 可以直接控制 M 接點的 On/Off 狀態
ButtonST: 可以直接控制 M 接點的 On/Off 狀態
ColorEdit: 可以直接輸入數值 F(float)， D(long)， MSG(Text) 等狀態
EzKnob: 可以直接 Mouse 控制 F(float)， D(long) 等狀態。
EzSlider: 可以直接 Mouse 控制 F(float)， D(long) 等狀態。
ColorRadio: 可以直接 Mouse 多選一控制 D(long) 如 D8000(=0~7) 可以直接控制顯示語系狀態。
ColorCheck: 可以直接 Mouse 多選多控制 B(BYTE) 等狀態。

2.5.4 多國語系顯示支援

EzHMI 不用寫介面程序經過簡單事先設定後就可以依 D8000(=0~7)暫存器數值直接顯示相對應語系文字。

LED
SWITCH
Lable
ButtonST
ColorRadio
ColorCheck
其中 Lable 還可以配合 1000 筆多語系的文字訊息

2.5.5 支援大型輸入介面

ColorEdit: 會產生一大型螢幕輸入鍵盤可以直接輸入數值 F(float)，D(long)， MSG(Text) 等狀態， 可以設定為密文輸入， 數值可以限制上下限。

2.5.6 支援動態警告介面

例如:

Lable:	可以直接設定文字閃爍
Lable:	可以直接設定數值上下限，超過時會閃爍
Lable:	MSG 如以"#"開頭的文字會閃爍
EzKnob:	可以直接圖錶顯示設定數值上下限，超過時會閃爍

2.5.7 支援動態更換圖片介面

例如:

Image:	可以用程序設定顯示 BMP 圖及位置
ButtonST:	可以有效與無效時的 BMP 圖

2.5.8 物件可支援不同 Windows 字型

可以選擇其他 Windows CE 可以接受的 TrueType 字型
例如:
Lable
ColorEdit
ButtonST
ColorRadio
ColorCheck
Position

2.5.9 物件可支援不同顏色設定

EzHMI 大部分的物件都可以支援文字與背景顏色的設定，除了 Image 無文字顯示功能。

2.5.10 物件可以設定為 Enable 或 Disable

一般在設計人機操作程序時常常需要作防止錯誤操作的機制，以免錯誤操作造成意外或錯誤，因此會經常需要控制物件是否可以讓操作者可以使用，此功能可透過 EzHMI 簡單的設定 M 暫存器來完成。
SWITCH
ColorEdit
ButtonST
ColorRadio
ColorCheck
Position

2.6 EzHMI 與 EzCore 的完美結合

本章節將詳細介紹 EzHMI 的 ActiveX 如何與 EzCore 聯結，並且說明這些 ActiveX 透過 EzCore 的 Real I/O 和 Registers，如何設定屬性達到監控的目的。

以下的圖表列出了兩者相互關係的部份，如橘、藍兩色箭頭所示:

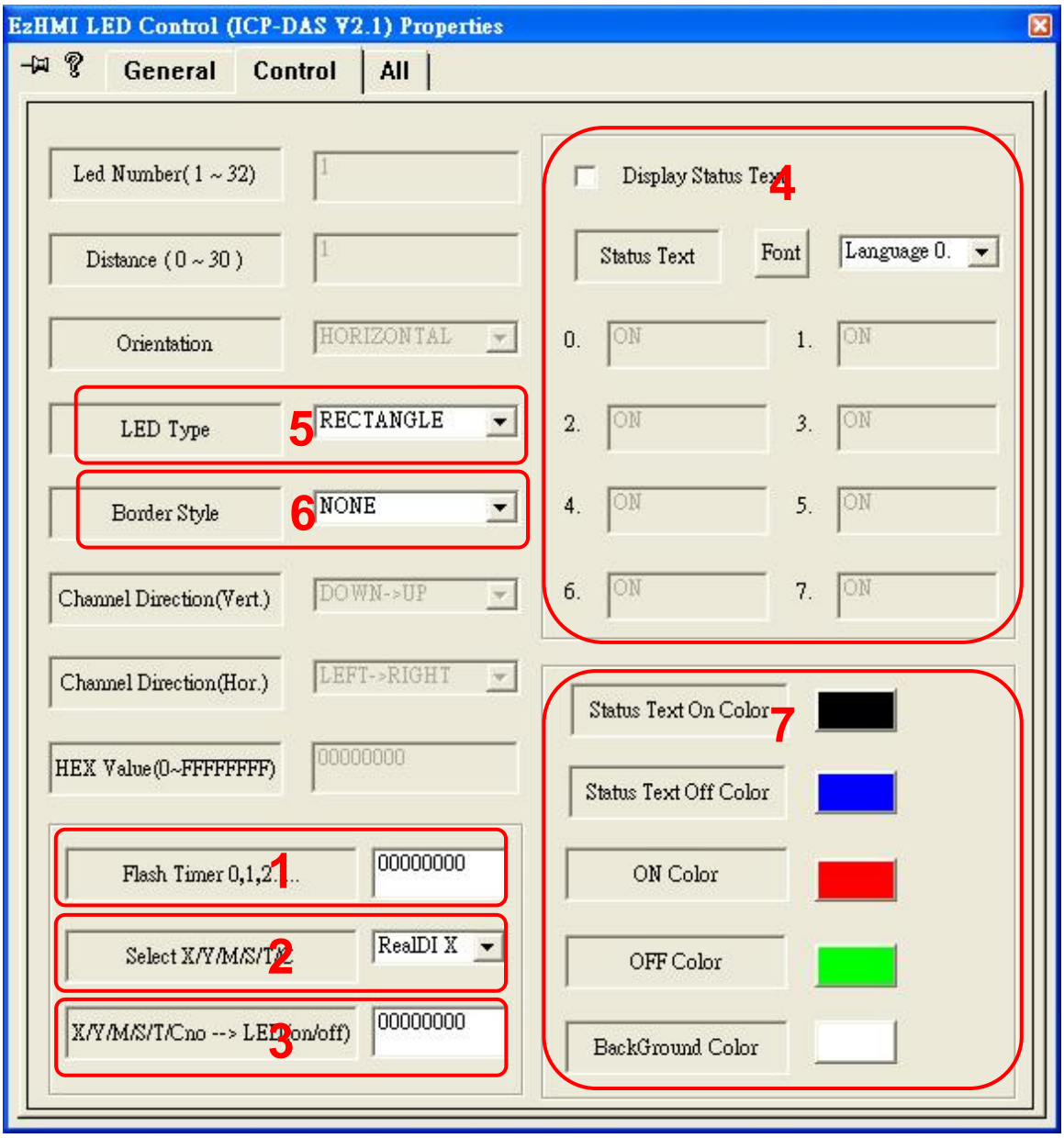
	LED	SWITCH	Label	ColorEdit	ButtonST	Image
D8000 multilingual(0~7)	↑	↑	↑		↑	
X0~X7777 : Real DI	↑					
Y0~Y7777 : Real DO	↑	←↑				
AO0~AO511 : Real AO			↑	←		
AI0~AI511 : Real AI			↑			
M0~M15999 AcitvieX Lamp(on/off)	↑	↑				
M0~M15999 AcitvieX Status(on/off)		←↑			←	
M0~M15999 AcitvieX Status(on)					←	
M0~M15999 AcitvieX Status(off)					←	
M0~M15999 DisableAcitvieX		↑		↑	↑	
M0~M15999 InitialChange ActiveX				↑		↑
D0~D79999			↑	←↑		
F0~F4095			↑	←↑		
B0~B2047						
MSG0~MSG249			↑	←↑		
T0~T299 Status(on/off)	↑					
C0~C1023 Status(on/off)	↑					

"↑": Register transfer to ActiveX
"←": ActiveX transfer to Register

	ColorRadio	ColorCheck	EzKnob	EzSlider	EzList	Position
D8000 multilingual(0~7)	↑	↑				
X0~X7777 : Real DI						
Y0~Y7777 : Real DO						
AO0~AO511 : Real AO			←↑	←↑		
AI0~AI511 : Real AI			↑	↑		
M0~M15999 AcitvieX Lamp(on/off)						
M0~M15999 AcitvieX Status(on/off)						
M0~M15999 AcitvieX Status(on)	←	←				
M0~M15999 AcitvieX Status(off)						
M0~M15999 DisableAcitvieX	↑	↑	↑	↑		↑
M0~M15999 InitialChange ActiveX	↑	↑	↑	↑	↑	
D0~D79999	←↑		←↑	←↑		
F0~F4095			←↑	←↑		
B0~B2047		←↑				
MSG0~MSG249					↑	
T0~T299 Status(on/off)						
C0~C1023 Status(on/off)						

"↑": Register transfer to ActiveX
"←": ActiveX transfer to Register

2.6.1 LED 聯結 EzCore屬性說明



1. **Flash Timer 0,1,2….. :**
   LED 燈號 on/off 的狀態來源，由內部計時器，在設定的時間內更新資料。時間單位為 50ms，0 表示不更新 EzCore 資料。

2. **Select X/Y/M/S/T/C :**
   選擇 LED 燈號 on/off 的狀態來源，是 Real DI/O 的 X/Y 或是 Register M、S(Step)、T(Timer)、C(Counter)。

3. **X/Y/M/Sno → LED(on/off) :**
   設定上述選定的 **EzCore** 元件號碼，**Register M/Sno** 設 **0** 表示忽略此項功能。

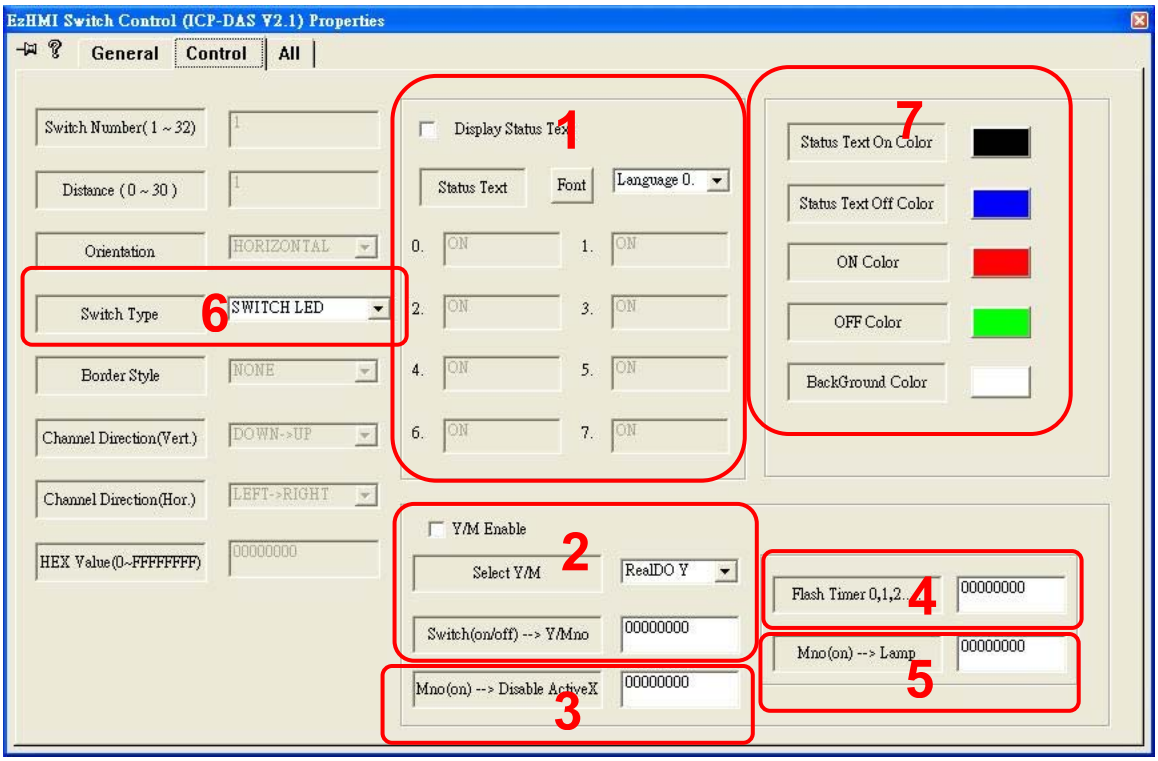
4. **Display StatusText:**。決定是否在物件的燈號上顯示文字
   Multi-Language :
   控件上的文字，提供了八國的 **unicode** 語系，讓使用者切換。
   程式中可以利用 **D8000(0~7)**，動態切換所對應的文字。
   如需使用本功能，需設定 **Flash Timer >= 1(50 ms)**。

5. **LED Type:**
   目前有方形，物件內燈號的外觀樣式選擇**(RECTANGLE)**圓形、**(ROUND)**及菱形**(DIAMOND)**。三種

6. **Board**
   Style: 目前，改變物件外框模式**NONE**、**RAISED**、**LOWERED**。三種

7. **Font :** 目前僅支援，文字字型設定**WinCE**上支援的。字型
   StatusText On Color: 當燈號值為**On(True or 1)**。時的文字顏色
   StatusText Off Color: 當燈號值為**Off(False or 0)**。時的文字顏色
   On Color: 當燈號值為**On(True or 1)**。燈號所顯示的顏色，時
   Off Color: 為當燈號值**Of(False or 0)**。燈號所顯示的顏色，時
   Background Color: 。燈號背景顏色設定

2.6.2 SWITCH 聯結 EzCore屬性說明



1. **Display StatusText:**。決定是否在物件的燈號上顯示文字
   Multi-Language :
   控件上的文字，提供了八國的 unicode 語系，讓使用者切換。
   程式中可以利用 D8000(0~7)，動態切換所對應的文字。
   如需使用本功能，需設定 Flash Timer >= 1。

2. **Y/M Enable :**
   致能 EzCore Y/M 的使用。
   Select Y/M :
   選擇 SWITCH 控制或顯示 Real DO Y、Register M 的狀態。
   Switch(on/off) → Y/Mno :
   設定上述選定的 EzCore 元件號碼，Register Mno 設 0 表示忽略此項功能。

3. **Mno(on) → Disable ActiveX :**
   設定使控件失效的 Register M 號碼，設 0 表示忽略此項功能。

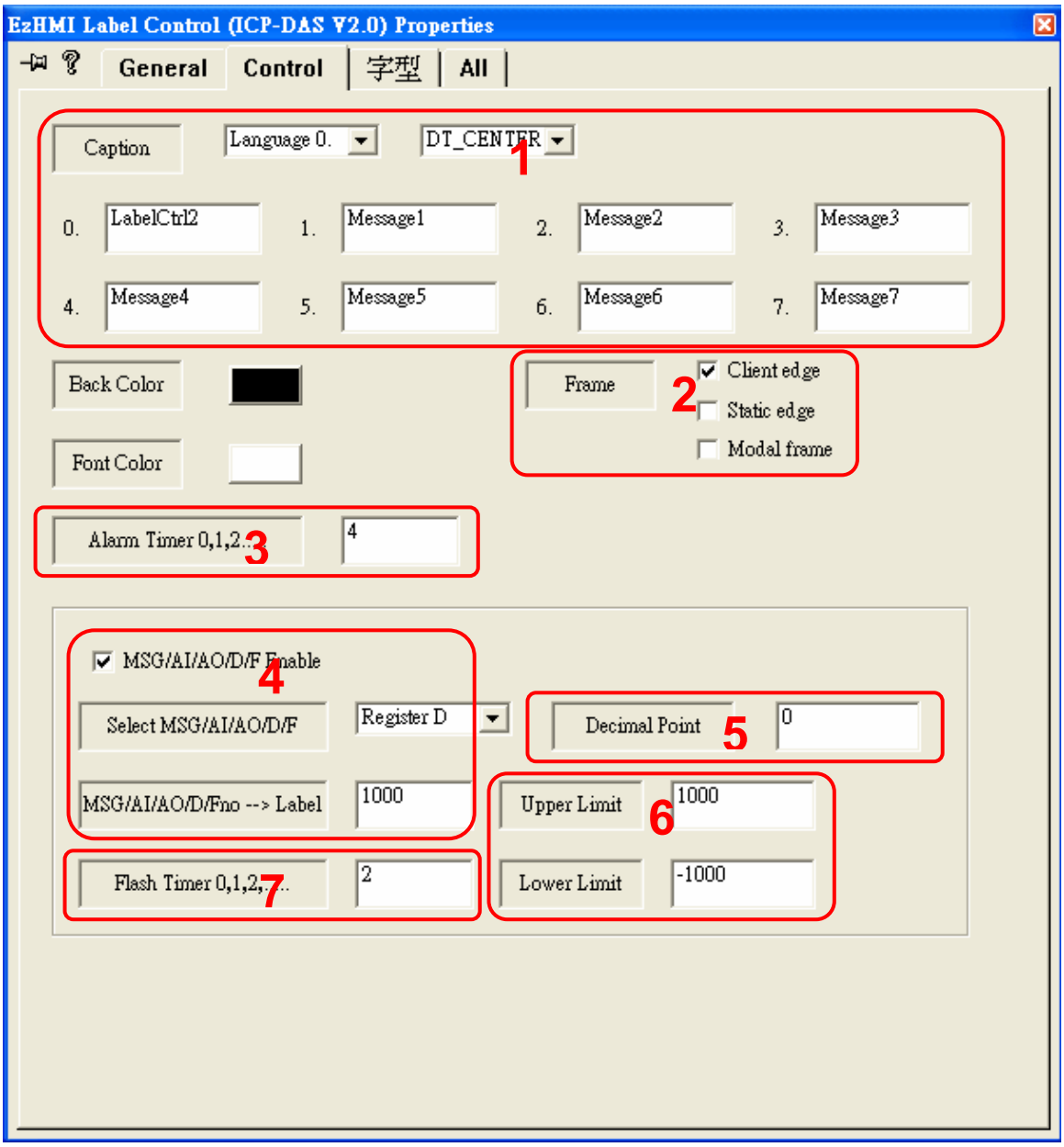
4. **Flash Timer 0,1,2..... :**
   SWITCH 燈號 **on/off** 的狀態來源，由內部計時器，在設定的時間內更新資料。時間單位為 **50ms**，**0** 表示不控制 **SWITCH** 的燈號**(Lamp)**及顯示 **Real DO Y**、**Register M** 的狀態**(**若 **Real DO Y** 或 **Register M** 在程式的邏輯控制中被更改狀態 **Switch** 也會自動更新顯示目前最新的狀態**)**。

5. **Mno(on) → Lamp :**
   設定 **Register M** 號碼，控制 **SWITCH-Lamp On** 或 **Off**。
   例**:** 參數如上屬性頁設定，當 **M1000: On**，然後燈號亮或滅。

6. **SWITCH** 控制元件內開關的外觀樣式有”**SWITCH ROUND**”、”**SWITCH RECTANGLE**”、”**SWITCH LEVER**”、”**SWITCH LED**”及”**SWITCH TOGGLE**”。

7. **Font :** 目前僅支援，文字字型設定**WinCE**上支援的。字型
   StatusText On Color: 當燈號值為**On(True or 1)**。時的文字顏色
   StatusText Off Color: 當燈號值為**Off(False or 0)**。時的文字顏色
   On Color: 當燈號值為**On(True or 1)**。所顯示的顏色燈號，時
   Off Color: 當燈號值為**Of(False or 0)**。燈號所顯示的顏色，時
   Background Color: 。燈號背景顏色設定

2.6.3 Label 聯結 EzCore屬性說明



1. **Multi-Language :**
   Label 的文字可以設定顯示位置，靠左、置中、靠右三個位置。
   控件上的文字，提供了八國的 unicode 語系，讓使用者切換。
   程式中可以利用 D8000(0~7)，動態切換所對應的文字。
   如需使用本功能，需設定 Flash Timer >= 1。

2. **Frame :**
   Client、Static、Modal 三種外框樣式選擇可複選。

3. **Alarm Timer 0,1,2….. :**
   設定警報的閃爍計時器，理想值為 4 (200ms)，時間單位為 50ms，0 表示關閉閃爍計時器。

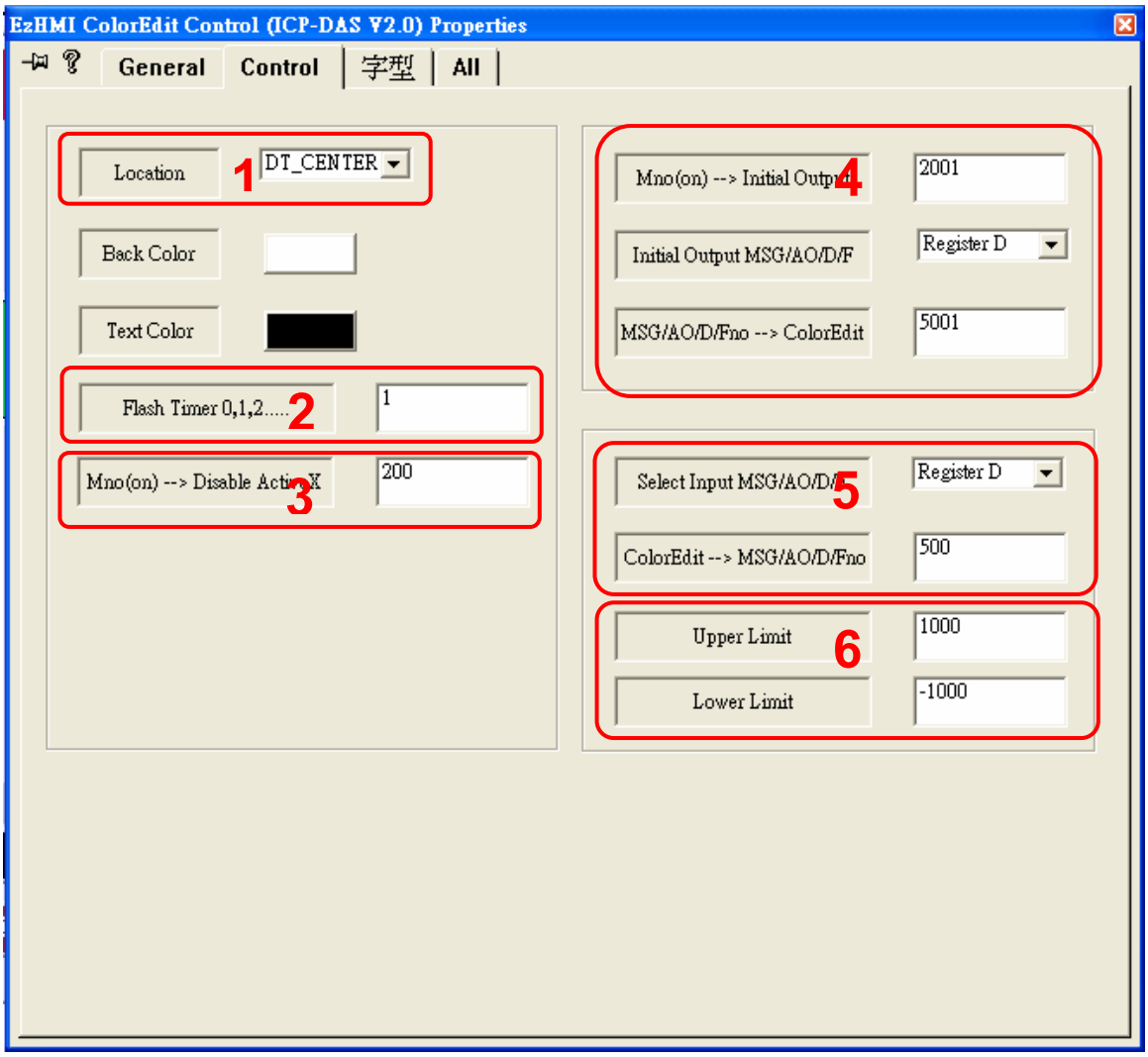
4. **MSG/AI/AO/D/F Enable :**
   致能 EzCore MSG/AI/AO/D/F 的使用。
   Select MSG/AI/AO/D/F :
   選擇傳送訊息 MSG 或是顯示 Real AI/O、Register D/F 的值。
   MSG/AI/AO/D/Fno → Label :
   設定上述選定的 EzCore 元件號碼，Label 將會顯示對應的資料，Register MSG/D/Fno 設 0 表示忽略此項功能。

5. **Decimal Point :**
   Select MSG/AI/AO/D/F 如果選擇 Register F(float)，必需設定小數位 (1~6)，設定 0 表示控件自行調整小數位。

6. **Upper Limit、Lower Limit :**
   如果選擇 Register F(float)或 Register D(long)，必需設定上下極限值，顯示的值如果不在設定的範圍內，將會啟動警報的閃爍計時器。

7. **Flash Timer 0,1,2….. :**
   Label 顯示的資料來源，由內部計時器，在設定的時間內更新資料。時間單位為 50ms，0 表示不更新 Label 顯示的資料。

2.6.4 ColorEdit 聯結 EzCore屬性說明



1. **Location :**
   ColorEdit 輸入的資料可以設定顯示位置，靠左、置中、靠右三個位置。

2. **Flash Timer 0,1,2….. :**
   ColorEdit 輸入資料的擷取速度，由內部計時器控制。時間單位為 50ms，0 表示 ColorEdit 不擷取輸入的資料。

3. **Mno(on) → Disable ActiveX :**
   設定使控件失效的 Register M 號碼，設 0 表示忽略此項功能。

4. **Mno(on) → Initial Output、Initial Output MSG/AO/D/F、MSG/AO/D/Fno → ColorEdit :**
   設定 Register M 號碼，控制上傳 ColorEdit 初始資料，設 0 表示忽略此項功能。
   設定 ColorEdit 初始顯示的資料形態，並設定 MSG/AO/D/F 的號碼。
   例: 參數如上屬性頁設定，當 M2001: On，然後 D5001 的值將上傳至 ColorEdit。

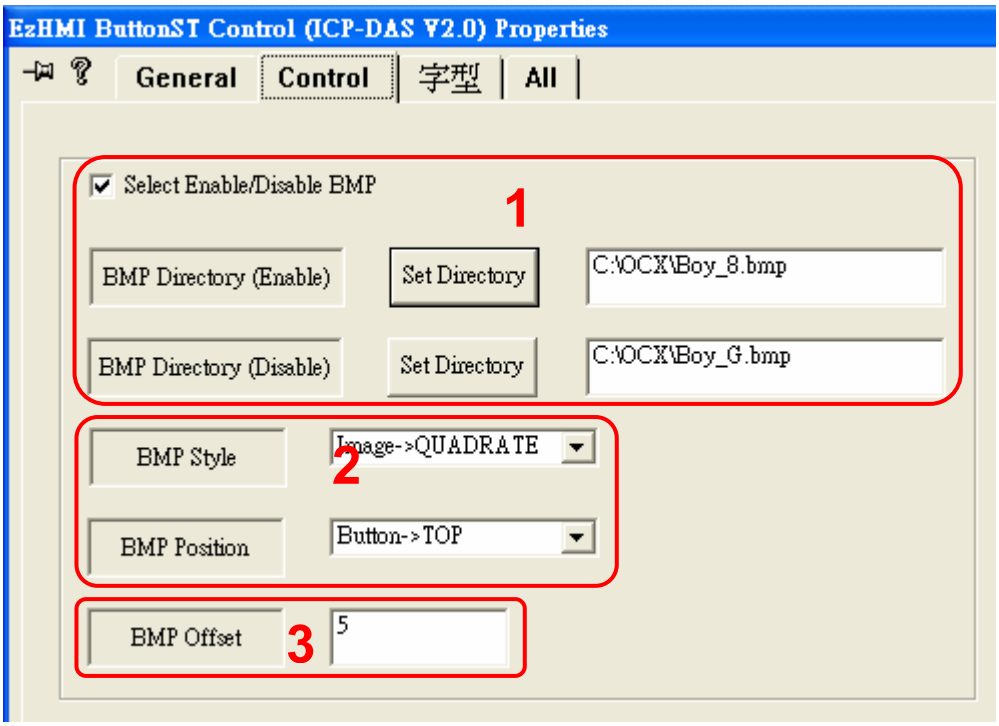
5. **Select Input MSG/AO/D/F、ColorEdit → MSG/AO/D/Fno :**
   - **選擇 Register F(float)或 Register D(long)，點擊 ColorEdit 時將彈出數字鍵盤(當 M16000 為 true 時)。**
   - **選擇 Register MSG，雙擊 ColorEdit 時將彈出英/數鍵盤。**
     MSG245~249 為 Password 輸入專用，輸入將顯示 ***** 保護。**

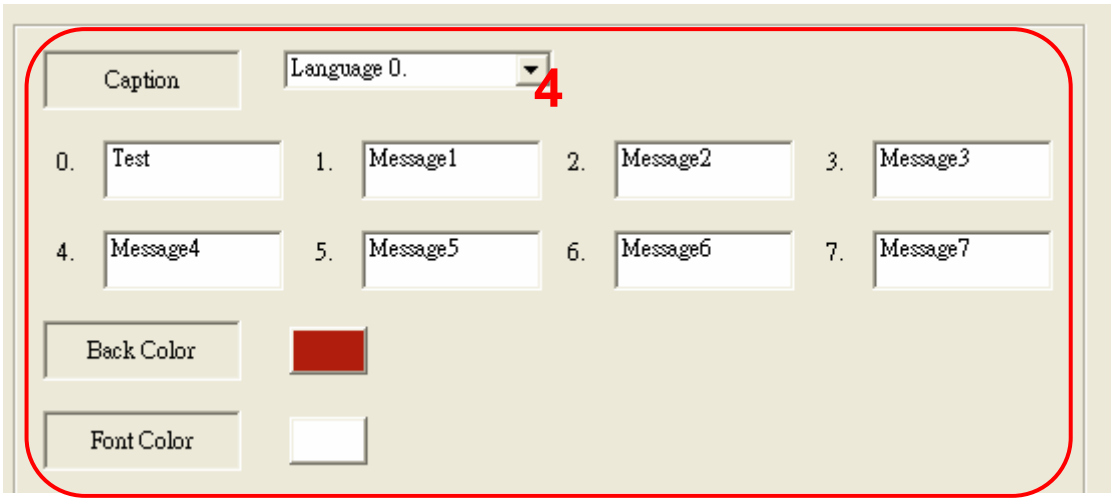
   ColorEdit 輸入資料擷取後，將傳入設定的 MSG/AO/D/Fno 中，Register MSG/D/Fno 設 0 表示忽略此項功能。

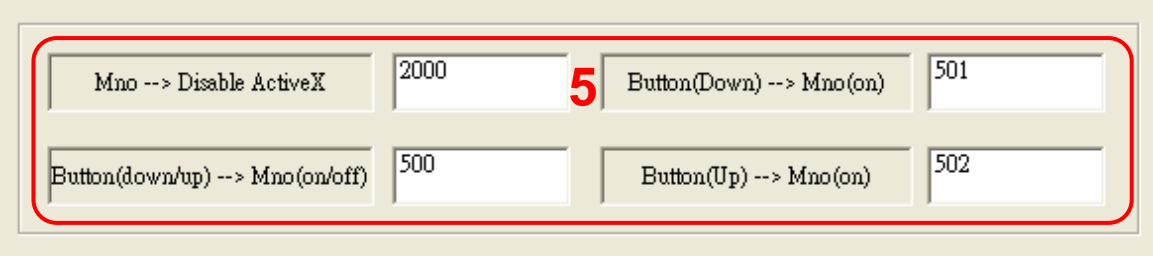
6. **Upper Limit、Lower Limit :**
   如果選擇 Register F(float)或 Register D(long)，必需設定輸入鍵盤的上下極限值，鍵盤輸入的值如果不在設定的範圍內，會自動代入安全值。
   如果選擇 Register F(float)，那麼 Upper Limit 所設定的小數位，將是鍵盤輸入容許最大的小數位。

2.6.5 ButtonST 聯結 EzCore屬性說明

由於 BMP 檔案大小(越小越好)會影響顯示效果，使用 BMP 前請先經適當處理(解析度與顏色深度 2,8,16,24 Bit)，如此可以獲得良好效果與效能。







1. **Select Enable/Disable BMP :**
   ButtonST 上的 BMP 圖，可以選擇顯示 Enable 或 Disable 時的個別圖檔。
   Set Directory :
   可以使用此按鈕去選取儲存在 PC 上的 BMP 圖檔。
   注意: 在 PAC 上"\EzProg_Path\EzProg-I\EzHMI\BMP\"的固定路徑中，必需存在同名相對應的圖檔。
   如沒輸入檔名或查無此檔，ButtonST 將不秀圖。

2. **BMP Style :**
   ButtonST 上的圖示，可以選擇全圖、半圖、或正方圖。
   BMP Position :
   圖示的位置也有，靠左貼、或置上貼兩個選擇。

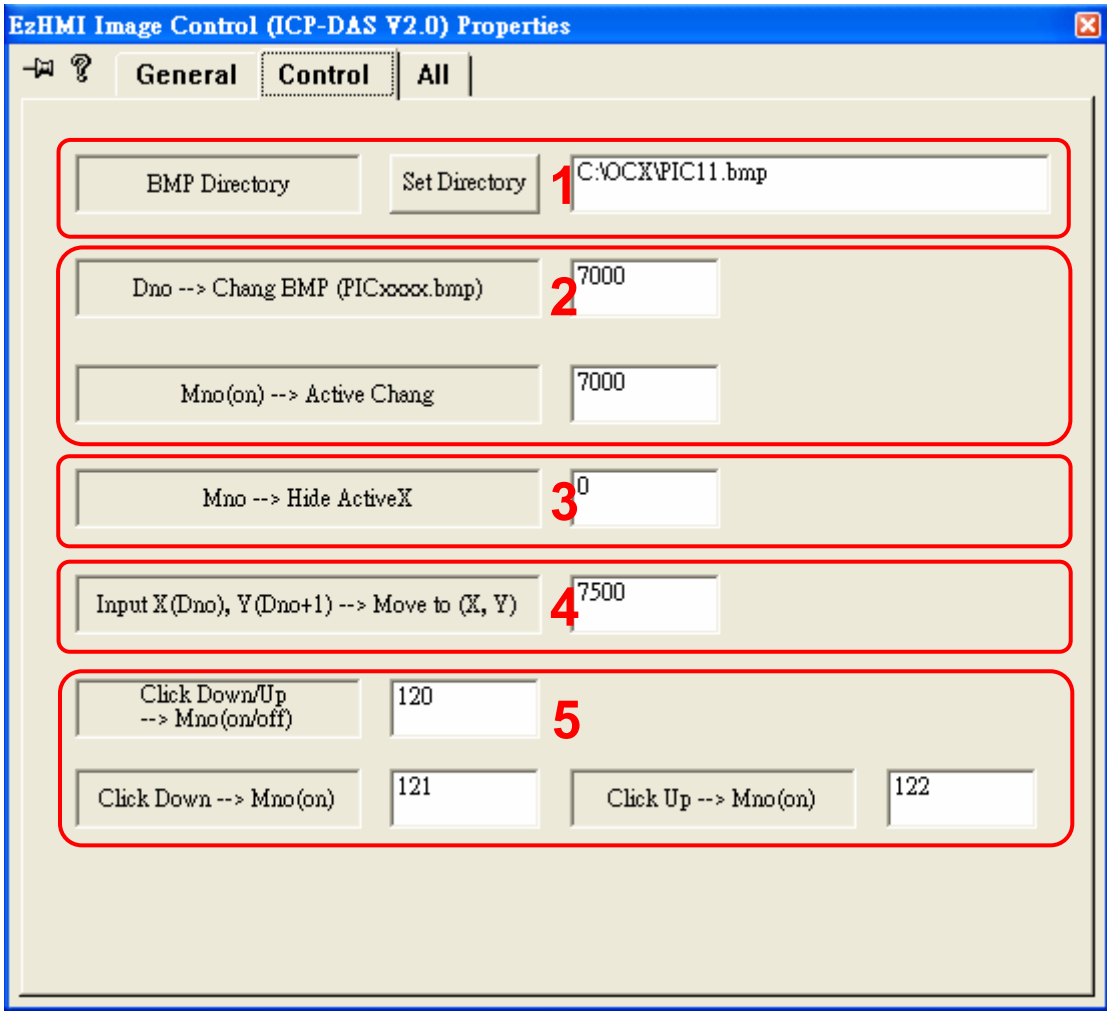
3. **BMP Offset :**
   ButtonST 上的 BMP 圖，預留板邊大小，單位 Pixel。

4. **Multi-Language :**
   控件上的文字，提供了八國的 unicode 語系，讓使用者切換。
   程式中可以利用 D8000(0~7)，動態切換所對應的文字。
   Back Color :
   ButtonST 上文字的背景顏色。
   Font Color :
   ButtonST 上文字的顏色。

5. **Mno → Disable ActiveX :**
   設定使控件失效的 Register M 號碼，設 0 表示忽略此項功能。
   Button (down/up) → Mno (on/off) :
   設定的 Register M，隨著 ButtonST 按鍵下即 On 壓彈起即 Off，設 0 表示忽略此項功能。
   Button (Down) → Mno (on) :
   ButtonST 按鍵下壓，設定的 Register M → On，設 0 表示忽略此項功能。
   Button (Up) → Mno (on) :
   ButtonST 按鍵彈起，設定的 Register M → On，設 0 表示忽略此項功能。

2.6.6 Image 聯結 EzCore屬性說明

由於 BMP 檔案大小(越小越好)會影響顯示效果，使用 BMP 前請先經適當處理(解析度與顏色深度 2,8,16,24 Bit)，如此可以獲得良好效果與效能。



1. **Set Directory :**
   可以使用此按鈕去選取儲存在 PC 上的 BMP 圖檔。
   注意: 在 PAC 上"\EzProg_Path\EzProg-I\EzHMI\BMP\"的固定路徑中，必需存在同名相對應的圖檔。
   如沒輸入檔名或查無此檔，Image 將不顯示圖片。

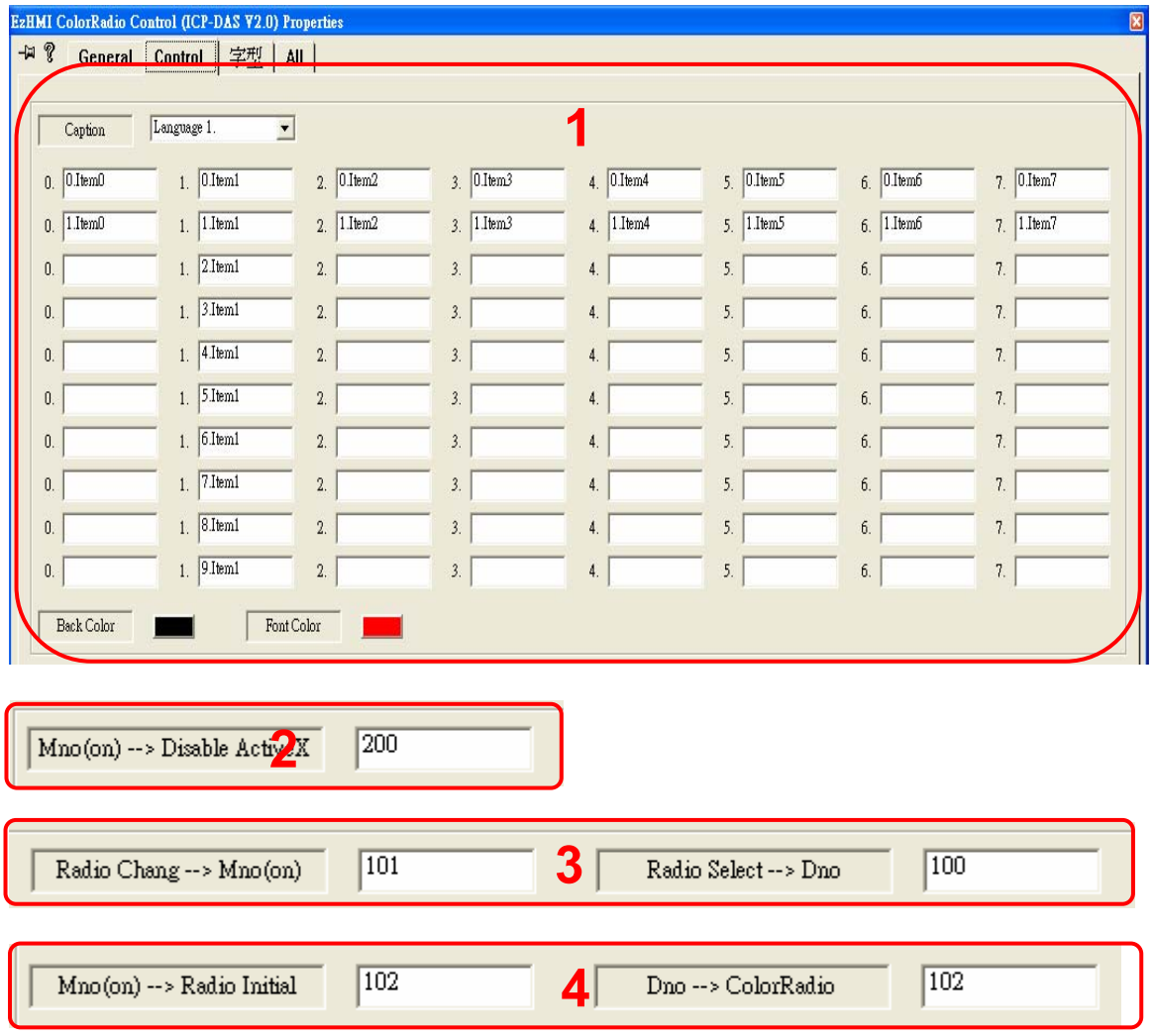
2. **Dno → Chang BMP (PICxxxx.bmp) :**
   更換 Image 上的 BMP 圖檔。
   Mno(on) → Active Chang :
   Mno: On，即進行圖檔更換，設 0 表示忽略此項功能。
   例: 如上屬性頁設定，如果 D7000 = 11，M7000: On，
   然後 Image 將會換上在 PAC 上的圖檔如下，
   "\EzProg_Path\EzProg-I\EzHMI\BMP\PIC11.bmp"。

3. **Mno → Hide ActiveX :**
   設定使控件隱藏的 Register M 號碼，設 0 表示忽略此項功能。

4. **Input X(Dno), Y(Dno+1) → Move to (X, Y):**
   X 座標: D7500 的值，Y 座標: D7501 的值。Dno 設 0 表示忽略此項功能。

5. **Click Down/Up → Mno (on/off) :**
   設定的 M120，隨著 Click 按下即 On，上即 Off，設 0 表示忽略此項功能。
   Click Down → Mno (on) :
   Click 按下，設定的 M121 → On，設 0 表示忽略此項功能。
   Click Up → Mno (on) :
   Click 上，即設定的 M122 → On，設 0 表示忽略此項功能。

2.6.7 ColorRadio 聯結 EzCore屬性說明



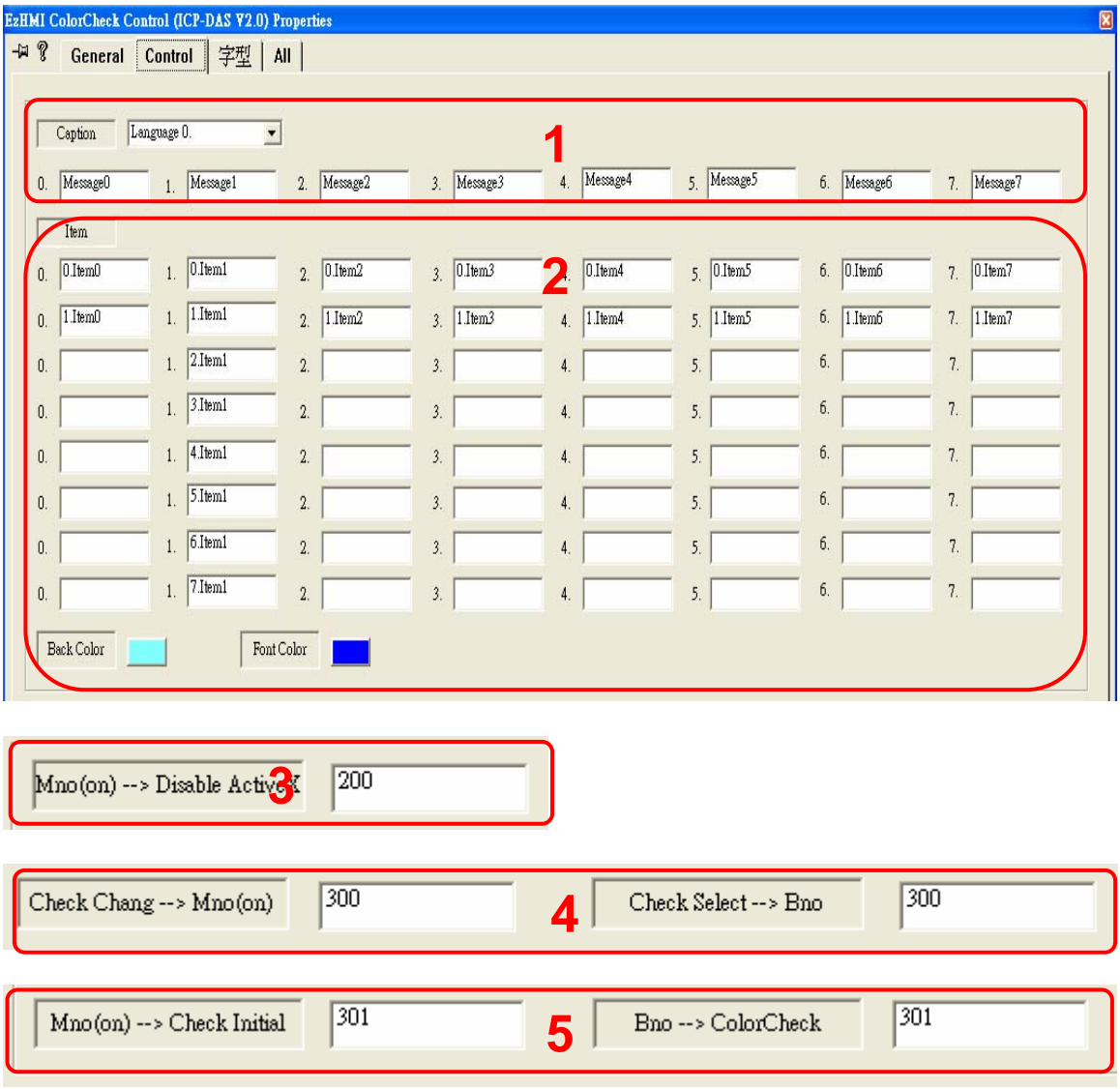
1. **Multi-Language :**
   控件上的文字，提供了八國的 unicode 語系，讓使用者切換。
   程式中可以利用 D8000(0~7)，動態切換所對應的文字。
   每一個語系皆有 10 個多選一的 Radioe 功能選項，不使用請空白。
   Back Color :
   ColorRadio 上文字的背景顏色。
   Font Color :
   ColorRadio 上文字的顏色。

2. **Mno(on) → Disable ActiveX :**
   設定使控件失效的 Register M 號碼，設 0 表示忽略此項功能。

3. **Radio Chang → Mno(on) :**
   當使用者更改了 ColorRadio 的選項時，設定的 Register M → On，設 0 表示忽略此項功能。
   Radio Select → Dno :
   當使用者更改了 ColorRadio 的選項時，將所選項目之值(0~9)，傳入設定的 Register D，設 0 表示忽略此項功能。

4. **Mno(on) Radio Initial、Dno → ColorRadio :**
   設定 Register M 號碼，控制上傳 ColorRadio 初始資料，設 0 表示忽略此項功能。
   例: 參數如上屬性頁設定，當 M102: On，然後 D102 的值(0~9)將上傳至 ColorRadio。

2.6.8 ColorCheck 聯結 EzCore屬性說明



1. **Multi-Language :**
   控件上的文字，提供了八國的 **unicode** 語系，讓使用者切換。
   程式中可以利用 **D8000(0~7)**，動態切換所對應的文字
   (Message0 ~ Message7)。

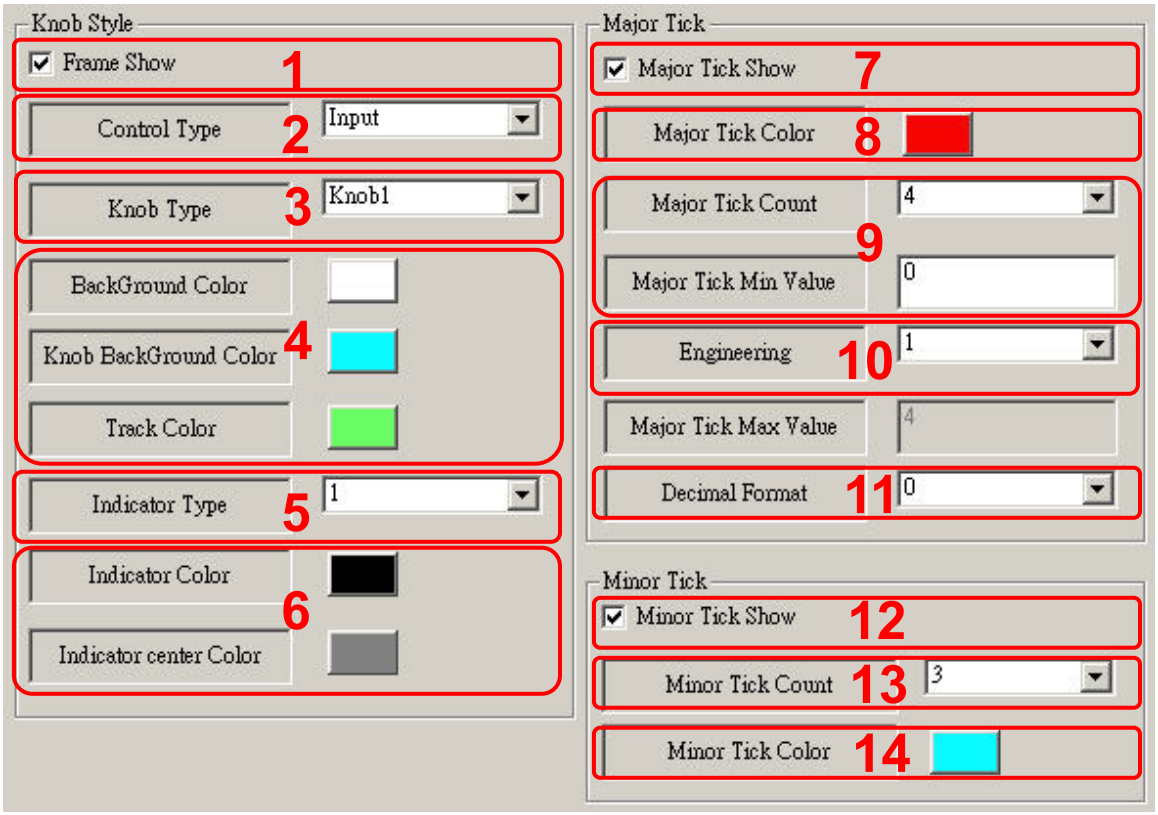
2. **Item :**
   每一個語系皆有 **8** 個複選的 **Check** 功能選項，不使用請空白。
   Back Color :
   ColorRadio 上文字的背景顏色。
   Font Color :
   ColorRadio 上文字的顏色。

3. **Mno(on) → Disable ActiveX :**
   設定使控件失效的 **Register M** 號碼，設 **0** 表示忽略此項功能。

4. **Check Chang → Mno(on) :**
   當使用者更改了 **ColorCheck** 的選項時，設定的 **Register M → On**，設 **0** 表示忽略此項功能。
   Check Select → Bno :
   當使用者更改了 **ColorCheck** 的選項時，將所選項目之值**(0x00~0xFF)**，傳入設定的 **Register B**，設 **0** 表示忽略此項功能。

5. **Mno(on) Check Initial、Bno → ColorCheck :**
   設定 **Register M** 號碼，控制上傳 **ColorCheck** 初始資料，設 **0** 表示忽略此項功能。
   例: 參數如上屬性頁設定，當 **M301: On**，然後 **B301** 的值**(0x00~0xFF)** 將上傳至 **ColorCheck**。

2.6.9 EzKnob 聯結 EzCore屬性說明



1. **Frame Show :**
   選擇是否顯示邊框。

2. **Control Type :**
   選擇作為輸入或用作顯示模式，選擇 “Output” 時聯結 EzCore Input 的屬性無效；選擇 “Input” 時若在程式中邏輯控制有改變 Register 的數值，EzKnob 會自動更新顯示目前最新的數值。

3. **Knob Type :**
   選擇 Knob 圖形。

4. **BackGround Color 、Knob BackGround Color 、TrackColor :**
   設定 EzKnob 背景顏色及軌道顏色。
   Knob BackGround Color 只針對設定 ”Knob3” 內圓部份的顏色。

5. **Indicator Type :**
   選擇指針的粗細。

6. **Indicator Color、Indicator Center Color :**
   設定指針及指針軸心的顏色。

7. **Major Tick Show :**
   選擇是否顯示大刻度。

8. **Major Tick Color :**
   設定刻度的顏色。

9. **Major Tick Count、Major Tick Min Value :**
   選擇刻度數量及設定刻度的最小值，接著就會在 **Major Tick Max Value** 看到刻度的最大值是多少。
   Major Tick Count 最大值為 **10**，若用手動輸入大於 **10** 之數值並無法保證圖形的準確度。

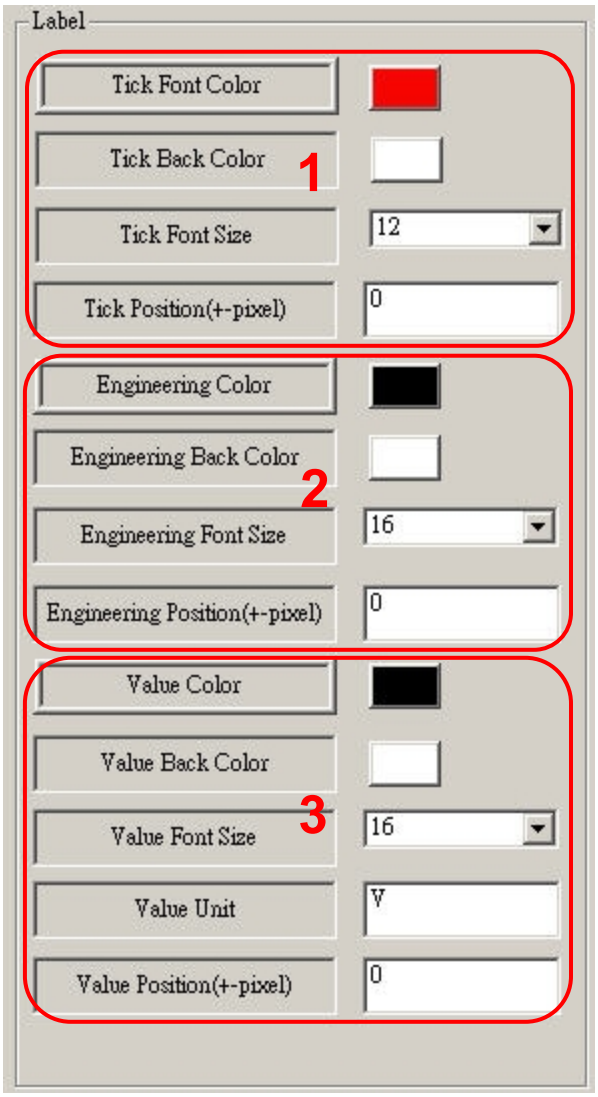
10. **Engineering :**
    選擇刻度數值的倍率，**EzKnob** 實際輸入或顯示的值是刻度值乘上倍率。

11. **Decimal Format :**
    選擇顯示 **Value** 所需的小數位數。

12. **Major Tick Show :**
    選擇是否顯示小刻度。

13. **Minor Tick Count :**
    選擇小刻度的數量。
    Minor Tick Count 最大值為 **10**，若用手動輸入大於 **10** 之數值並無法保證圖形的準確度。

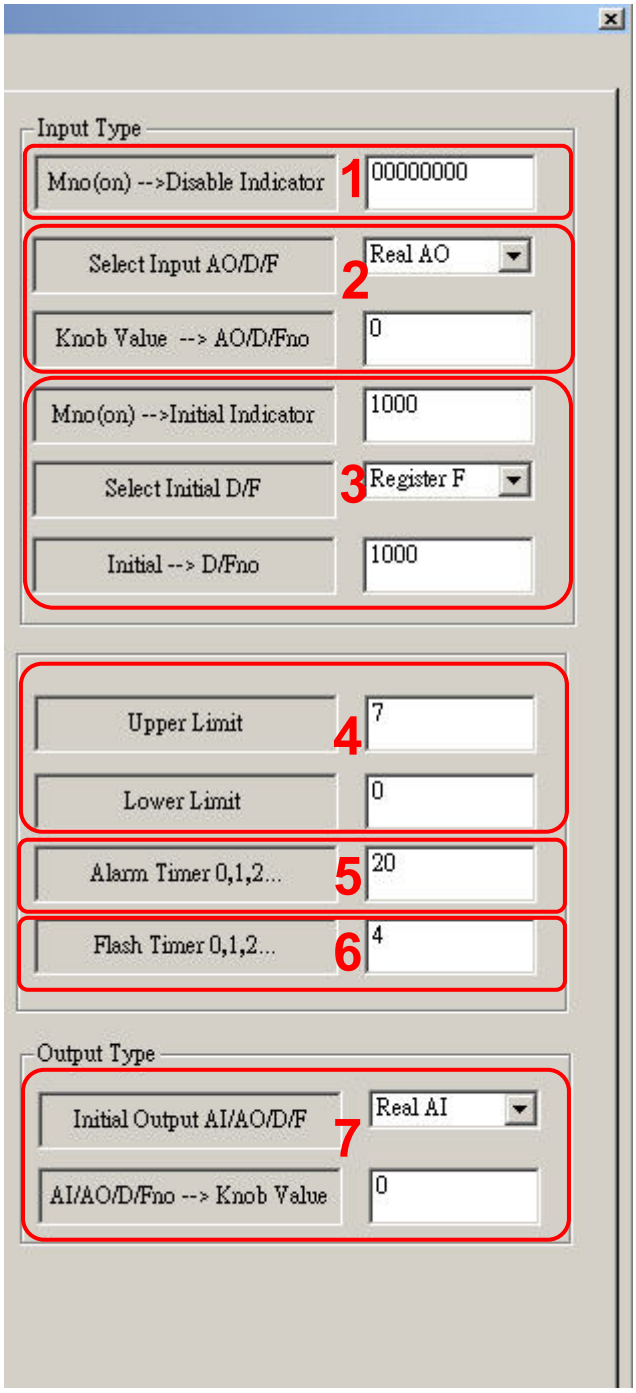
14. **Major Tick Color :**
    設定小刻度的顏色。



1. **Tick Font Color、Tick Back Color、Tick Font Size、TickPosition(+–pixel) :**
   設定顯示刻度值的文字顏色、文字背景顏色、字體大小、文字位置左右微調。

2. **Engineering Color、Engineering Back Color、Engineering Font Size、Engineering Position(+–pixel) :**
   設定顯示倍率的文字顏色、文字背景顏色、字體大小、文字位置左右微調。

3. **Value Color、Value Back Color、Value Font Szie、Value Unit、Value Position(+–pixel) :**
   設定顯示數值的文字顏色、文字背景顏色、字體大小、數值單位、文字位置左右微調。


Input Type
Mno(on) -->Disable Indicator
1
00000000
Select Input AO/D/F
Real AO
2
Knob Value --> AO/D/Fno
0
Mno(on) -->Initial Indicator
1000
Select Initial D/F
Register F
3
Initial --> D/Fno
1000
Upper Limit
4
7
Lower Limit
0
Alarm Timer 0,1,2...
5
20
Flash Timer 0,1,2...
6
4
Output Type
Initial Output AI/AO/D/F
Real AI
7
AI/AO/D/Fno --> Knob Value
0

1. **Mno(on) → Disable Indicator :**
   設定使控件滑鼠移動指針功能失效的 **Register M** 號碼，設 **0** 表示忽略此項功能。

2. **Select Input AO/D/F、Knob Value → AO/D/Fno :**
   滑鼠移動指針而得的數值輸入擷取後，將傳入設定的 **AO/D/Fno** 中，**Register D/Fno** 設 **0** 表示忽略此項功能。

3. **Mno(on) → Initial Indicator 、Select Initial D/F、Initial → D/Fno :**
   設定 **Register M** 號碼，控制上傳 **EzKnob** 初始資料，設 **0** 表示忽略此項功能。
   設定 **EzKnob** 初始顯示的資料形態，並設定 **D/F** 的號碼。
   例: 參數如上屬性頁設定，當 **M1000: On**，然後 **F1000** 的值將上傳至 **EzKnob**。

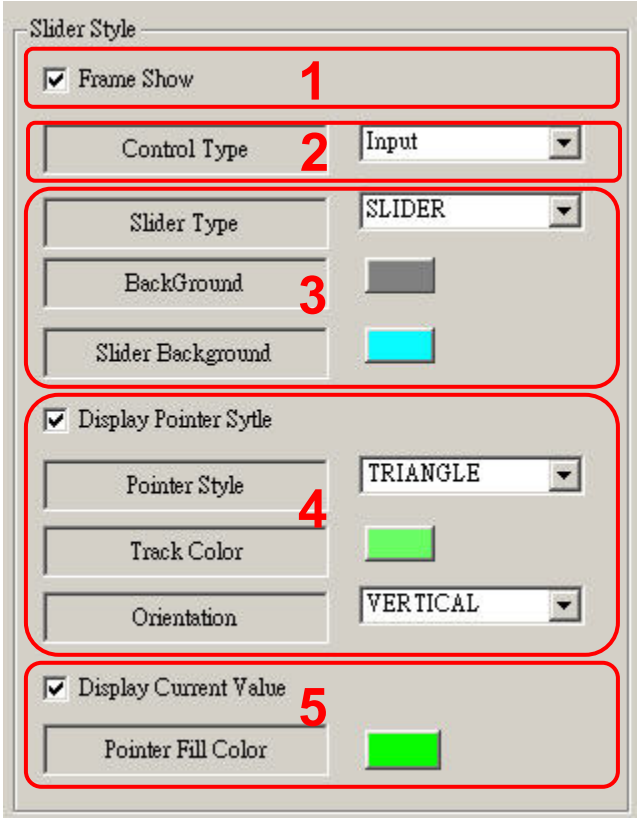
4. **Upper Limit、Lower Limit :**
   設定上下極限值，輸入或顯示的值如果不在設定的範圍內，將會啟動警報的閃爍計時器。

5. **Alarm Timer 0,1,2….. :**
   設定警報的閃爍計時器，理想值為 **4 (200ms)**，時間單位為 **50ms**，**0** 表示關閉閃爍計時器。

6. **Flash Timer 0,1,2….. :**
   - **EzKnob** 顯示的資料來源，由內部計時器，在設定的時間內更新資料。時間單位為 **50ms**，**0** 表示不更新 **EzKnob** 顯示的資料。
   - 由 **EzKnob** 輸入的數值，由內部計時器，在設定的時間內更新至 **AO/D/F**。時間單位為 **50ms**，**0** 表示不更新輸入的數值至 **AO/D/F**。

7. **Initial Output AI/AO/D/F、AI/AO/D/Fno → Knob Value:**
   設定 **EzKnob** 初始顯示的資料形態，並設定 **AI /AO/D/F** 的號碼。
   例: 參數如上屬性頁設定，**AI** 第 **0 channel** 的值將上傳至 **EzKnob**。

2.6.10 EzSlider 聯結 EzCore屬性說明



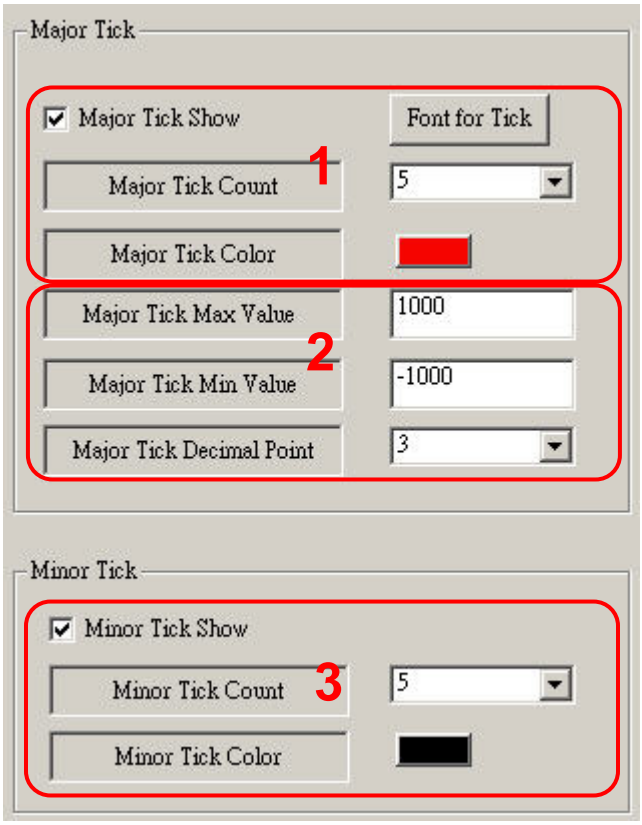
1. **Frame Show :**
   選擇是否顯示邊框。

2. **Control Type :**
   選擇作為輸入或用作顯示模式，選擇 “Output” 時聯結 EzCore Input 的屬性無效；選擇 “Input” 時若在程式中邏輯控制有改變 Register 的數值，EzSlider 會自動更新顯示目前最新的數值。

3. **Slider Type、BackGround、Slider Background :**
   選擇圖形樣式、背景顏色及圖形顏色。

4. **Display Pointer Style、Pointer Style、Track Color、Orientation :**
   選擇是否顯示游標、選擇游標樣式、設定軌道顏色、設定 EzSlider 為直立式或水平式(水平只對”SLIDER”圖形樣式有效”)。

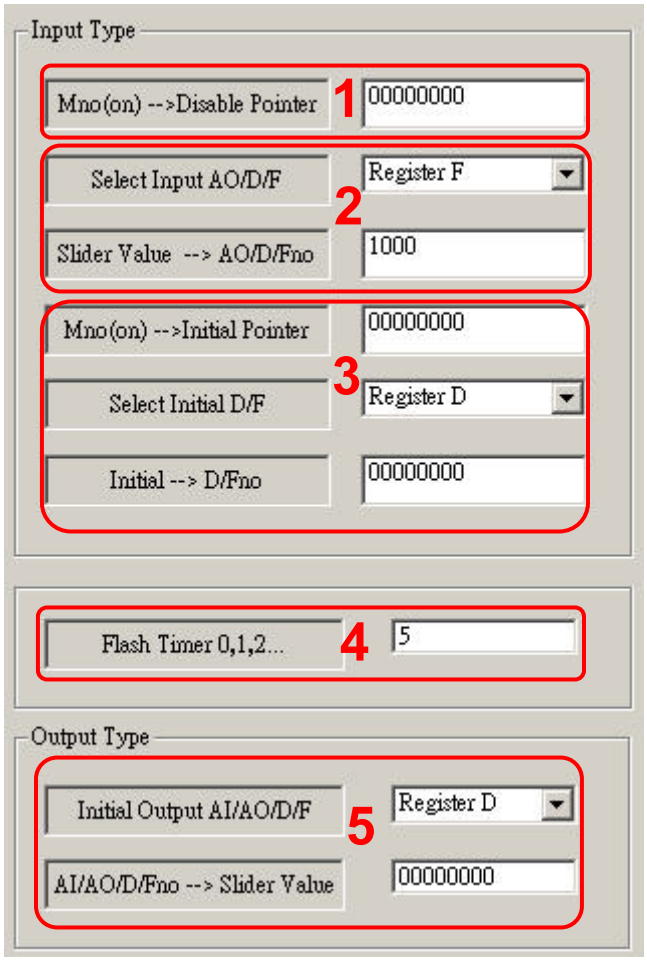
5. **Display Current Value、Pointer Fill Color :**
   選擇是否顯示數值、設定游標顏色。



1. **Major Tick Show 、Font for Tick、Major Tick Count、Major Tick Color:**
   選擇是否顯示刻度、設定刻度字型、選擇刻度數量、設定刻度顏色。
   Major Tick Count 最大值為 **10**，若用手動輸入大於 **10** 之數值並無法保證圖形的準確度。

2. **Major Tick Max Value、Major Tick Min Value、Major Tick Decimal Point :**
   設定刻度最大及最小值、選擇顯示的小數位數。

3. **Minor Tick Show、Minor Tick Count、Minor Tick Color**
   選擇是否顯示小刻度、選擇小刻度格數、設定小刻度顏色。
   Minor Tick Count 最大值為 **9**，若用手動輸入大於 **9** 之數值並無法保證圖形的準確度。


Input Type
Mno(on) -->Disable Pointer
1
00000000
Select Input AO/D/F
Register F
2
Slider Value --> AO/D/Fno
1000
Mno(on) -->Initial Pointer
00000000
3
Select Initial D/F
Register D
Initial --> D/Fno
00000000
Flash Timer 0,1,2...
4
5
Output Type
Initial Output AI/AO/D/F
Register D
5
AI/AO/D/Fno --> Slider Value
00000000

1. **Mno(on) → Disable Poniter :**
   設定使控件滑鼠移動指針功能失效的 **Register M** 號碼，設 **0** 表示忽略此項功能。

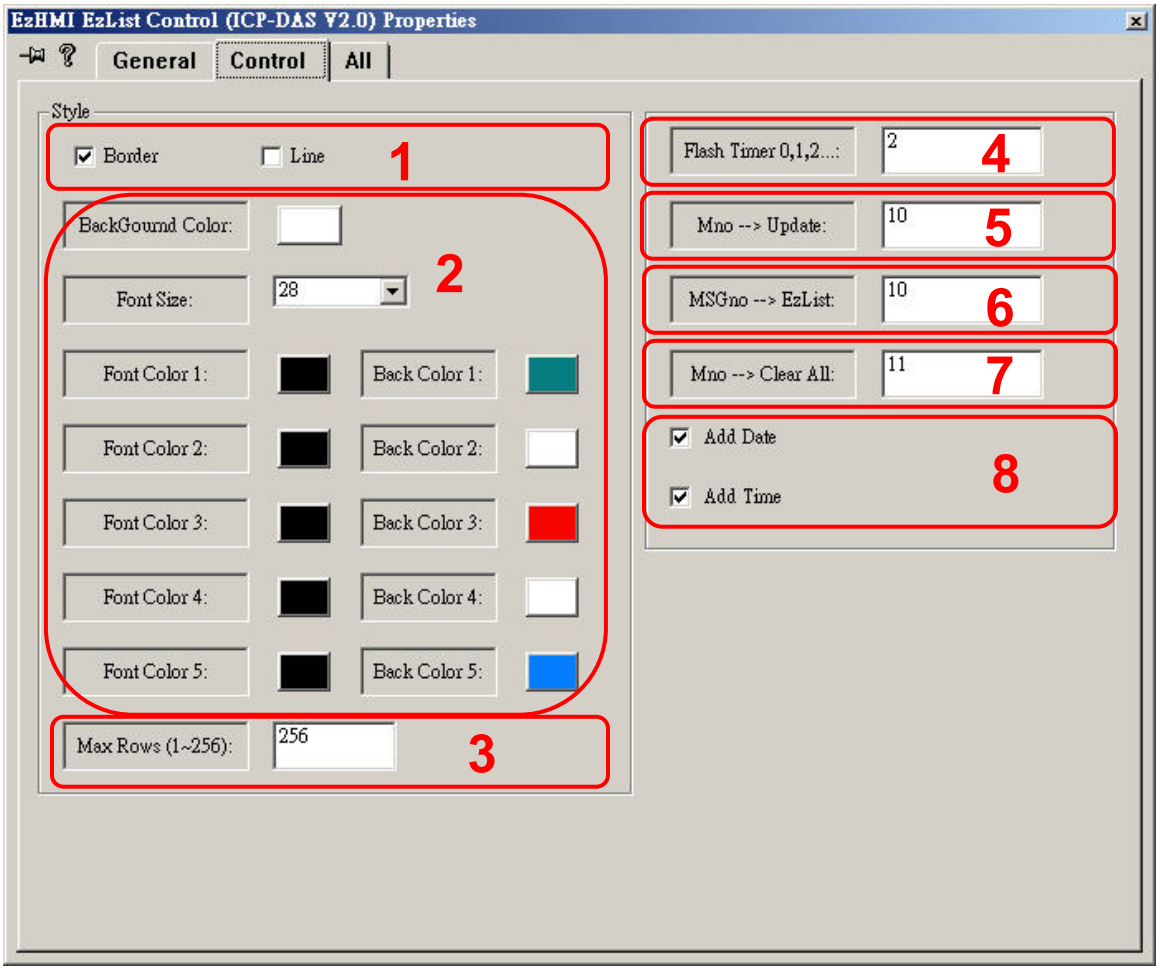
2. **Select Input AO/D/F、Slider Value → AO/D/Fno :**
   滑鼠移動指針而得的數值輸入擷取後，將傳入設定的 **AO/D/Fno** 中，**Register D/Fno** 設 **0** 表示忽略此項功能。

3. **Mno(on) → Initial Poniter 、Select Initial D/F、Initial → D/Fno :**
   設定 **Register M** 號碼，控制上傳 **EzSlider** 初始資料，設 **0** 表示忽略此項功能。
   設定 **EzSlider** 初始顯示的資料形態，並設定 **D/F** 的號碼。

4. **Flash Timer 0,1,2….. :**
   - **EzSlider** 顯示的資料來源，由內部計時器，在設定的時間內更新資料。時間單位為 **50ms**，**0** 表示不更新 **EzSlider** 顯示的資料。
   - 由 **EzSlider** 輸入的數值，由內部計時器，在設定的時間內更新至 **AO/D/F**。時間單位為 **50ms**，**0** 表示不更新輸入的數值至 **AO/D/F**。

5. **Initial Output AI/AO/D/F、AI/AO/D/Fno → Slider Value:**
   設定 **EzSlider** 初始顯示的資料形態，並設定 **AI /AO/D/F** 的號碼。

2.6.11 EzList 聯結 EzCore屬性說明



1. **Border 、Line :**
   選擇是否顯示外框及列分隔線。

2. **BackGround Color :**
   設定 EzList 背景顏色。

   Font Size :
   選擇文字大小。

   Font color 1~5 :
   選擇文字顏色，共可設定五種。

   Back Color 1~5 :
   選擇每一列的文字背景顏色，共可設定五種。

3. **Max Rows(1~256) :**
   設定 **EzList** 可顯示的最大行數，可設定的範圍為 **1** 到 **256** 。

4. **Flash Timer 0，1，2….. :**
   EzList 顯示的資料來源，由內部計時器，在設定的時間內更新資料。時間單位為 **50ms**，**0** 表示不更新 **EzList** 顯示的資料。

5. **Mno → Update :**
   設定 **Register M** 號碼，控制是否將 **Register MSG** 的內容顯示在 **EzList** 上，設 **0** 表示忽略此項功能。

6. **MSGno → EzList :**
   設定要顯示在 **EzList** 上的 **Register MSG** 號碼。

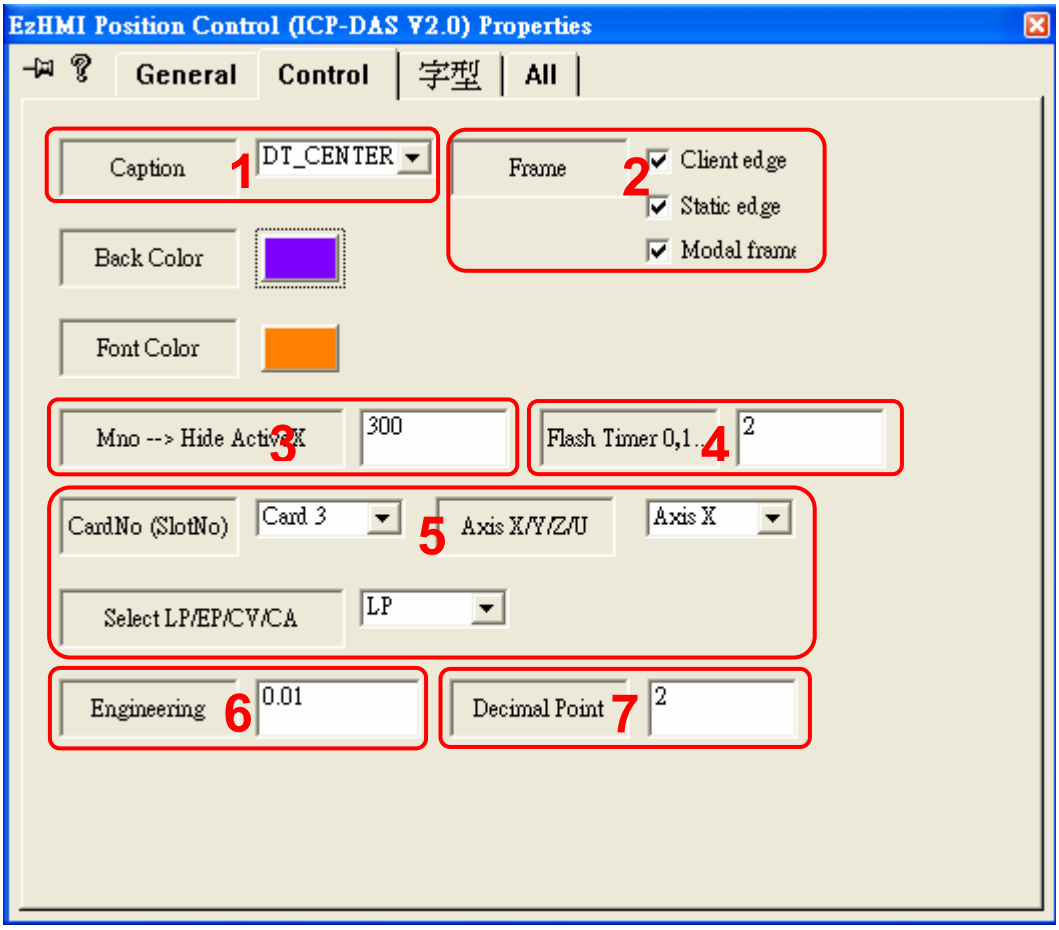
7. **Mno → Clear All :**
   設定 **Register M** 號碼，控制是否要將 **EzList** 目前顯示的文字清除掉。

8. **Add Date :**
   選擇是否在 **MSG** 文字前顯示日期，依目前 **PAC** 系統內所設定的日期為準。

   Add Time :
   選擇是否在 **MSG** 文字前顯示時間，依目前 **PAC** 系統內所設定的時間為準。

2.7 與 EzCore 結合的進階應用

2.7.1 Position 聯結 Motion 屬性說明



1. **Caption :**
   Position 的文字可以設定顯示位置，靠左、置中、靠右三個位置。

2. **Frame :**
   Client、**Static**、**Modal** 三種外框樣式選擇可複選。

3. **Mno(on) → Hide ActiveX :**
   設定使控件隱藏的 **Register M** 號碼，設 **0** 表示忽略此項功能。

4. **Flash Timer 0,1,2….. :**
   Position 顯示的資料來源，由內部計時器，在設定的時間內更新資料。時間單位為 **50ms**，**0**、**1** 皆表示 **50ms** 更新 **Position** 顯示的資料。

5. **CardNo (SlotNo) :**
   選擇 **Card1~Card7**，**Position** 欲顯示的模組卡號，目前支援。
   Axis X/Y/Z/U :
   選擇 **Axis_X**、**Axis_Y**、**Axis_Z**、**Axis_U**，任一軸。
   Select LP/EP/CV/CA :
   選擇 **Position** 欲顯示邏輯位置、編碼器位置、速度或加速度資訊。

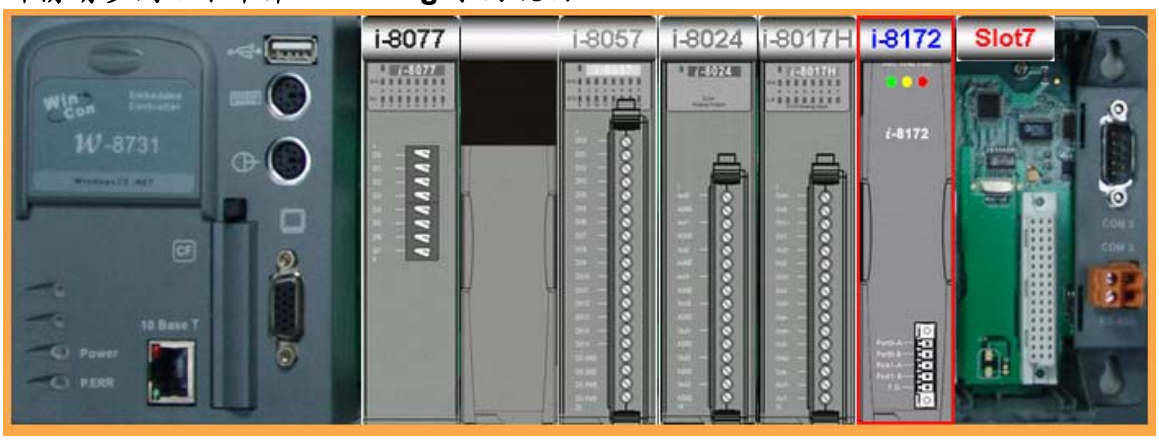
6. **Engineering :**
   Position 顯示值可以乘上一個工程單位(**double**)。

7. **Decimal Point :**
   設定 **Position** 顯示值需幾個小數位(**1~14**)，設定 **0** 表示控件自行調整小數位。

3 EzConfig 介紹與應用

3.1 使用 EzConfig 規劃工具

開始使用 EzProg-I 開發前，請先使用 EzConfig 規劃工具，規劃完成並存檔，詳情請參閱如下章節 EzConfig 手冊說明



Slot1： 

EzConfig 有支援之模組以灰階表示。

Slot2： 
表示 EzConfig 未支援之模組。

Slot6： 

EzConfig 有支援之模組，並且選取設定中。

Slot7： 

Scan 不到任何模組。

以下列出目前 **EzConfig** 支援模組，**i8092F,i8094F (only support FRnet)**

模組	形態	支援	函式庫	備註
i8017-G	DEV_TYPE_AI17	◎	WinConSDK.DLL	8 channels AI
i8024-G	DEV_TYPE_AO24	◎	WinConSDK.DLL	4 channels AO
i8037-G	DEV_TYPE_DO_16	◎	WinConSDK.DLL	
i8040-G	DEV_TYPE_DI_32	◎	WinConSDK.DLL	
i8041-G	DEV_TYPE_DO_32	◎	WinConSDK.DLL	
i8042-G	DEV_TYPE_DI16DO16	◎	WinConSDK.DLL	
i8048-G	DEV_TYPE_DI_8	◎	WinConSDK.DLL	Only for slot 1
i8051-G	DEV_TYPE_DI_16	◎	WinConSDK.DLL	
i8052-G	DEV_TYPE_DI_8	◎	WinConSDK.DLL	
i8053-G	DEV_TYPE_DI_16	◎	WinConSDK.DLL	
i8054-G	DEV_TYPE_DI8DO8	◎	WinConSDK.DLL	
i8055-G	DEV_TYPE_DI8DO8	◎	WinConSDK.DLL	
i8056-G	DEV_TYPE_DO_16	◎	WinConSDK.DLL	
i8057-G	DEV_TYPE_DO_16	◎	WinConSDK.DLL	
i8058-G	DEV_TYPE_DI_8	◎	WinConSDK.DLL	
i8060-G	DEV_TYPE_DO_8	◎	WinConSDK.DLL	
i8063-G	DEV_TYPE_DI8DO8	◎	WinConSDK.DLL	
i8064-G	DEV_TYPE_DO_8	◎	WinConSDK.DLL	
i8065-G	DEV_TYPE_DO_8	◎	WinConSDK.DLL	
i8066-G	DEV_TYPE_DO_8	◎	WinConSDK.DLL	
i8068-G	DEV_TYPE_DO_8	◎	WinConSDK.DLL	
i8069-G	DEV_TYPE_DO_8	◎	WinConSDK.DLL	
i8077-G	DEV_TYPE_DI8DO8	◎	WinConSDK.DLL	
i8172-G	DEV_TYPE_FR8172	◎	WinConSDK.DLL	
i8092F	DEV_TYPE_8092F	◎	i8092.DLL	Card No=SLOT No
i8094F	DEV_TYPE_8094F	◎	i8094.DLL	Card No=SLOT No

PS: 其餘未列出的模組，依然可以使用其專用的函式庫，在 eVC++的平台上混合應用不受影響。

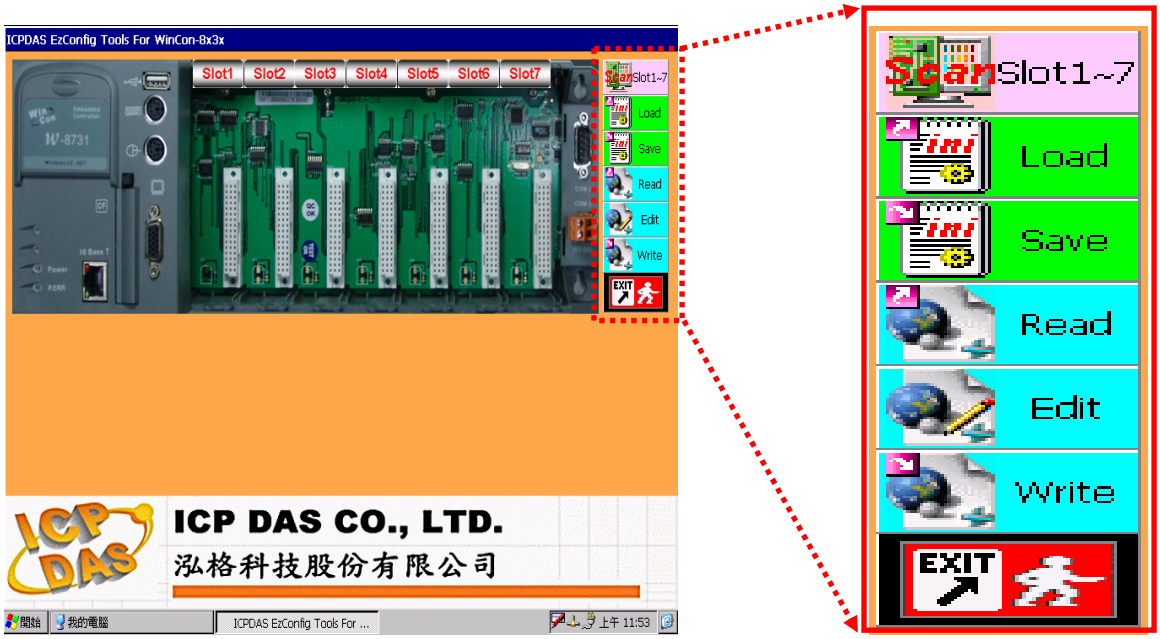
3.2 開啟 EzConfig

請按開始->程式集->ICPDAS->EzProg-I->EzConfig 如下圖:

出現 EzConfig 主畫面，請參考 3.3 如下章節:

3.3 EzConfig 主功能簡介

EzConfig 的主畫面，如下圖:



：自動掃瞄所有插槽上的模組

：載入先前已儲存舊的規劃檔，DEVICE.INI、XXX.NOT

：儲存目前規劃資料在 DEVICE.INI、XXX.NOT

：將 IO_TABLE.XML 檔案中的 Val、NOTE，更新到 EzConfig

：編輯所有 Register(T、C、M、B、W….)的預設值、NOTE

：將目前規劃資料輸出到，IO_TABLE.XML、IO_TABLE.TXT

：離開 EzConfig

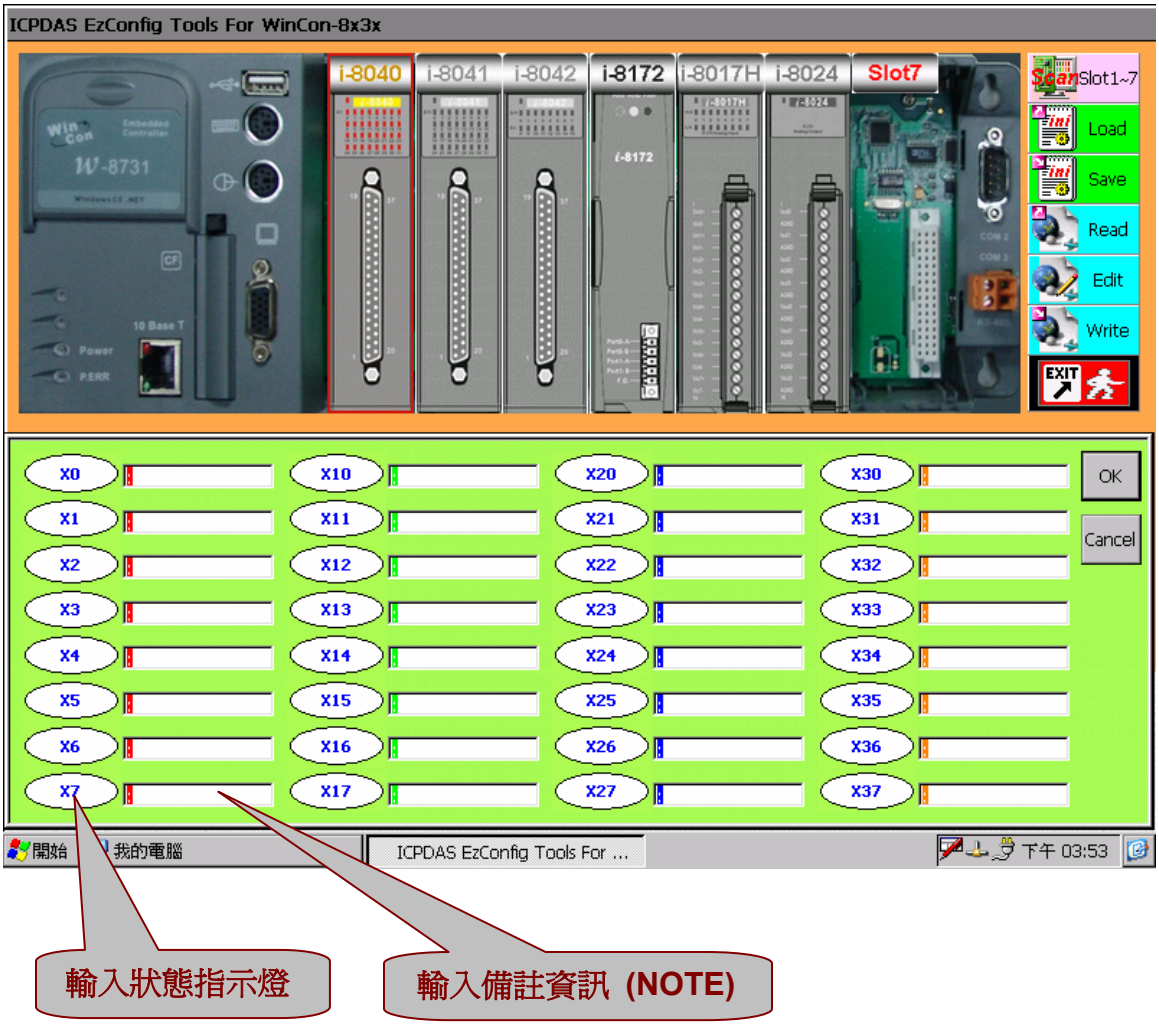
3.4 Scan_Button 功能說明

如下圖所示，按下 Scan 按鈕，將重新規劃所有插槽上的模組

將插槽上的模組偵測與顯示出來

3.4.1 Config DI 功能說明

單擊 i8040 模組圖示，會即時顯示 DI 狀態(ON/OFF),並可以在各 DI 點右側編輯框內編輯 NOTE(備註)

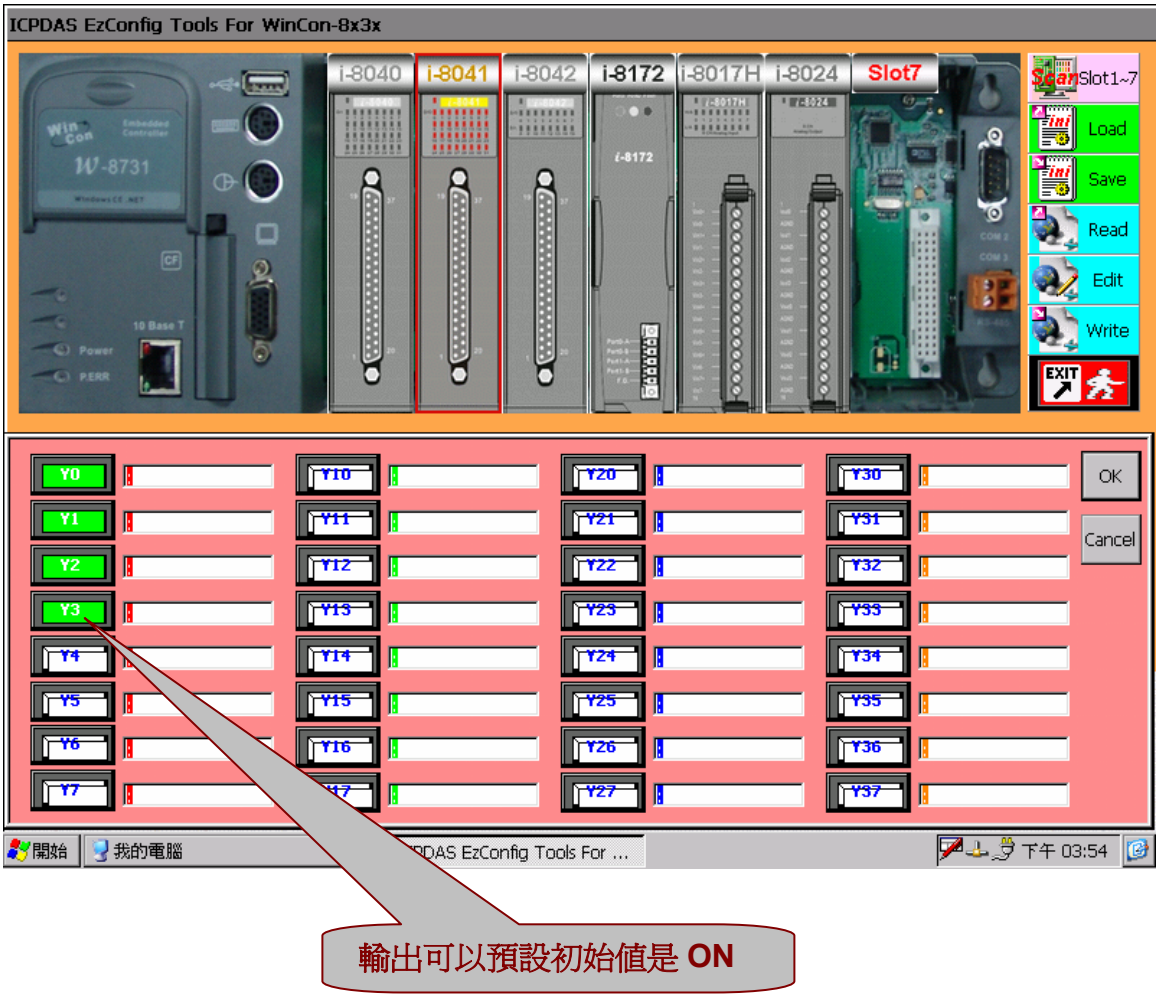


輸入點自動以 8 進制方式編號:

X00~X07
X10~X17
X20~X27
X30~X37

3.4.2 Config DO 功能說明

單擊 i8041 模組圖示，左邊按鈕可以用來測試 DO 輸出，並可作為系統開始之預設值，並可以在各 DO 點右側編輯框內編輯 NOTE(備註)



輸出點自動以 8 進制方式編號:

Y00~Y07
Y10~Y17
Y20~Y27
Y30~Y37

3.4.3 Config DI/O 功能說明

單擊 i8042 模組圖示，DI DO 各分別與 3.4.1，3.4.2 可以測試 DI 和 DO，並可以在各 DIO 點右側編輯框內編輯 NOTE(備註)

輸入點自動以 8 進制方式編號:

X40~X47

X50~X57

輸出點自動以 8 進制方式編號:

Y40~Y47

Y50~Y57

3.4.4 Config FRnet (Port :Group)

單擊 i8172 模組圖示，由於 Frnet 有相當多的 IO 點數，並可以選定 DI/O 的 Port 和 Group 是否要規劃使用，如有勾選時表示要規劃使用

輸入點自動以 8 進制方式編號:

X1000~X1007
X1010~X1017
X1020~X1027
X1030~X1037

輸出點自動以 8 進制方式編號:

Y1000~Y1007
Y1010~Y1017
Y1020~Y1027
Y1030~Y1037

3.4.5 Config FRnet DI 功能說明

單擊 1:08 紅色按鈕，會即時顯示 FRnet DI 狀態(ON/OFF),並可以在各 DI 點右側編輯框內編輯 NOTE(備註)

1:08 → 1 : Port 1，08 : Group(DI 08)

1:08 輸入點編號:

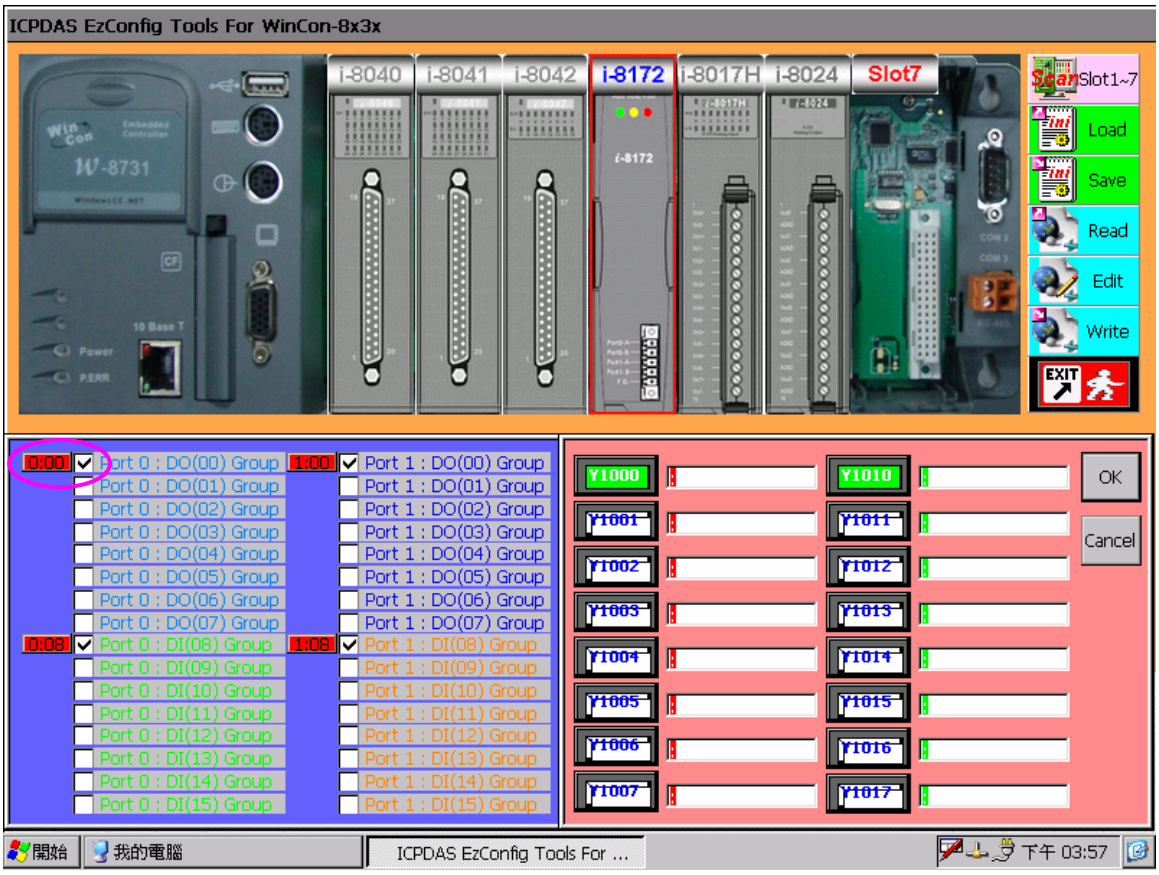
X1020~X1027

X1030~X1037

3.4.6 Config FRnet DO 功能說明

單擊 0:00 紅色按鈕，左邊按鈕可以用來測試 FRnet DO 輸出，並可作為系統開始之預設值，並可以在各 DO 點右側編輯框內編輯 NOTE(備註)

0:00 → 0 : Port 0，00 : Group(DO 00)

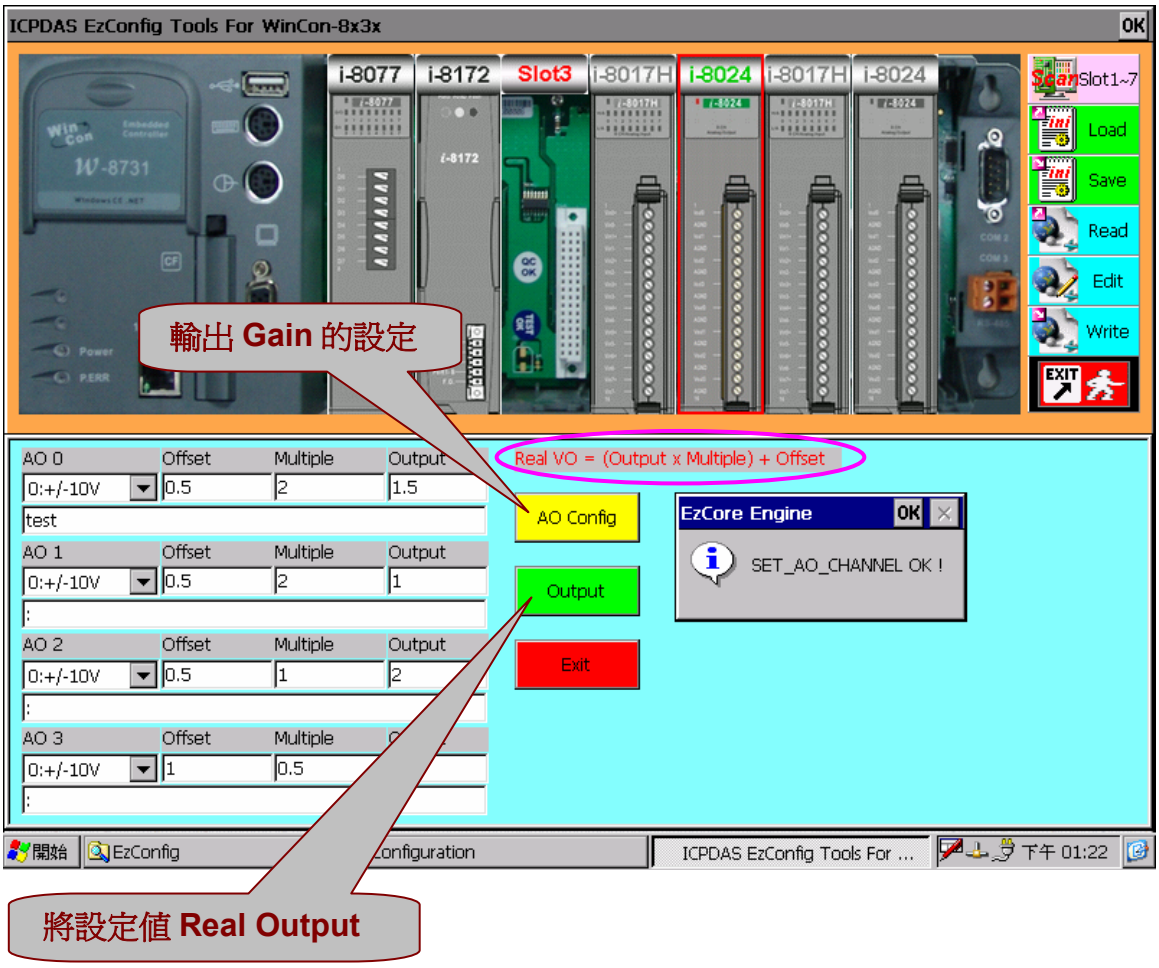


0:00 輸出點編號:

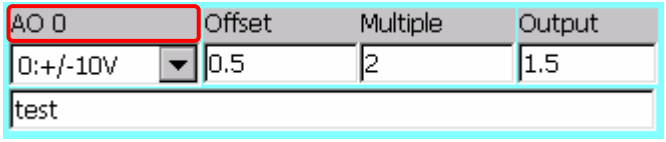
Y1000~Y1007

Y1010~Y1017

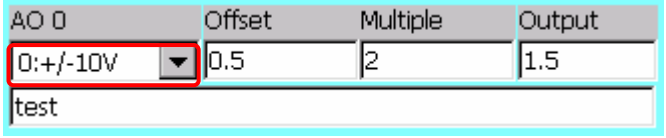
3.4.7 Config AO 功能說明



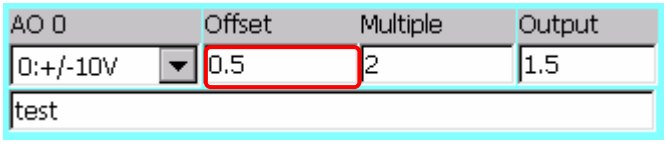
1. 選擇 AO 通道，這是 SCAN Slot 後自動產生的(按模組順序排列)。



2. 選擇 AO 通道的輸入範圍及種類(Gain Mode)。

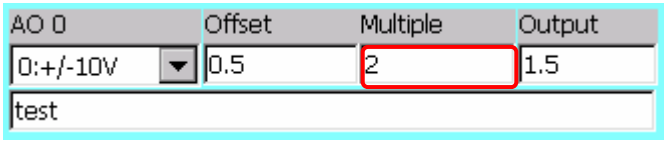


3. 設定 AO 通道的偏差值，可作為軟體校正值(Offset 單位 Volt)。



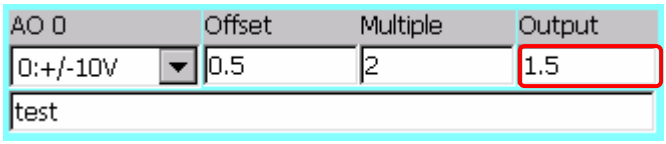
表示補正硬體 0.5Volt。

4. 設定 AO 通道的倍率，可作為軟體工程單位轉換(Multiplet 單位 V/工程單位)。



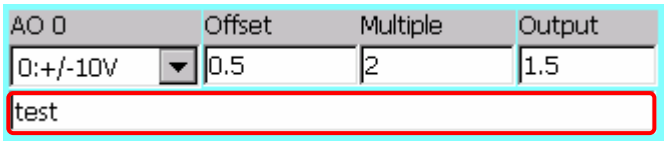
表示 1 個工程單位= 2Volt。

5. 設定 AO 通道的預設輸出值(Output 工程單位)。



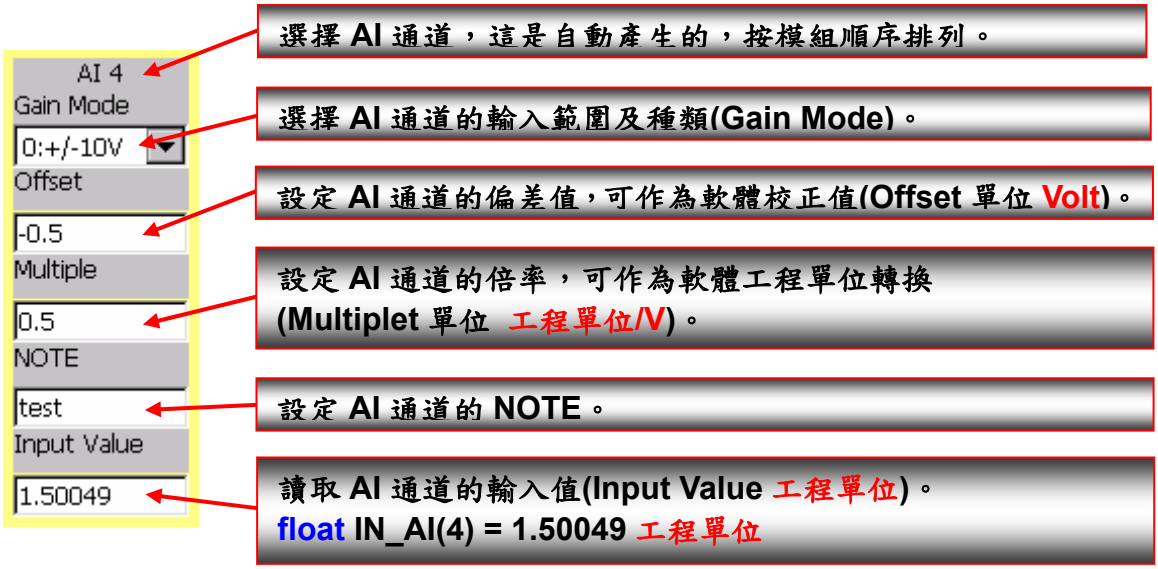
表示輸出 1.5 個工程單位，OUT_AO(0, 1.5)。

6. 設定 AO 通道的 NOTE。



例如上述工程單位為公斤，輸出 1.5Kg，設 1 Kg 代表電壓 2V，並補正 0.5V。
實際硬體 AO 輸出 ：Real VO = (Output x Multiple) + Offset
(1.5 Kg x 2 V/Kg) + 0.5V = 3.5V
實際輸出電壓 3.5 Volt

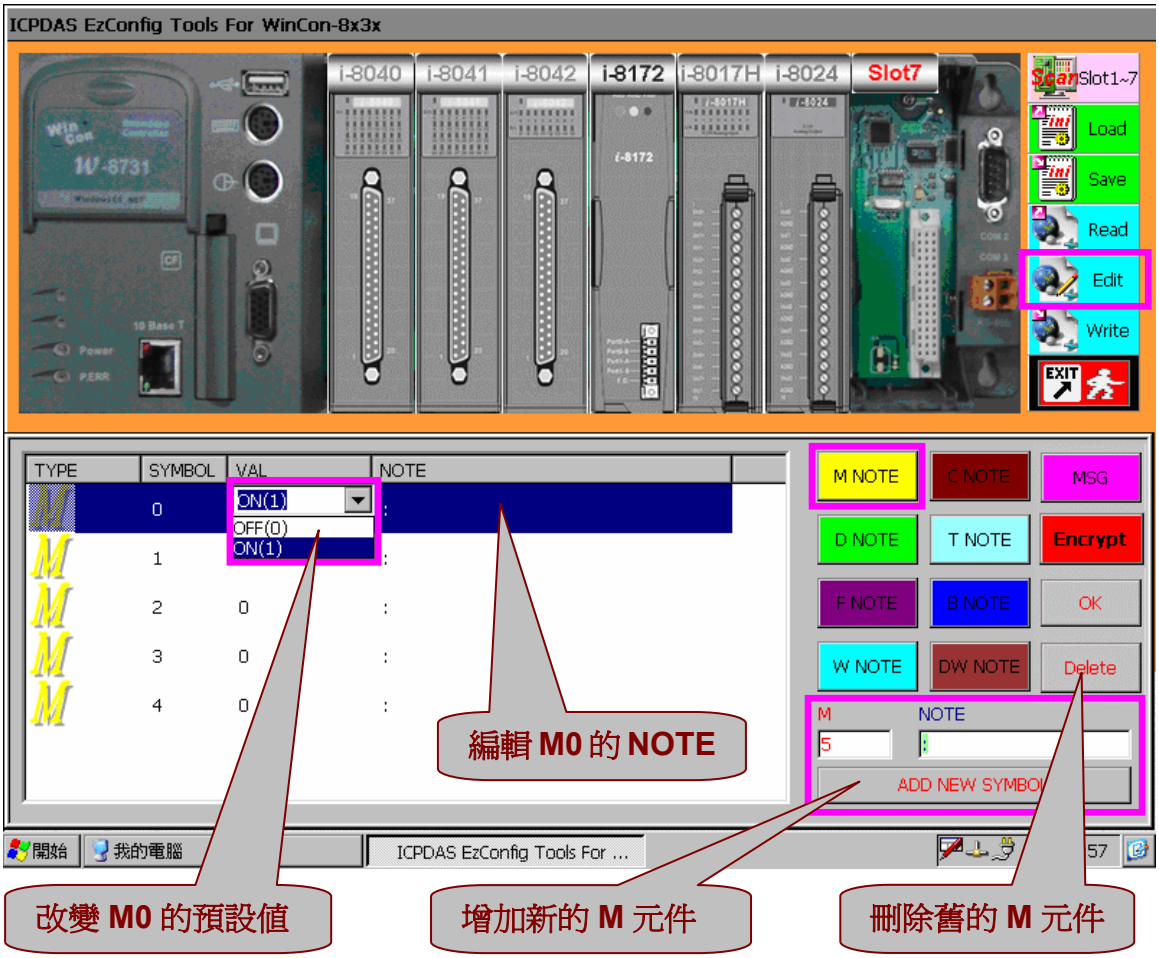
3.4.8 Config AI 功能說明



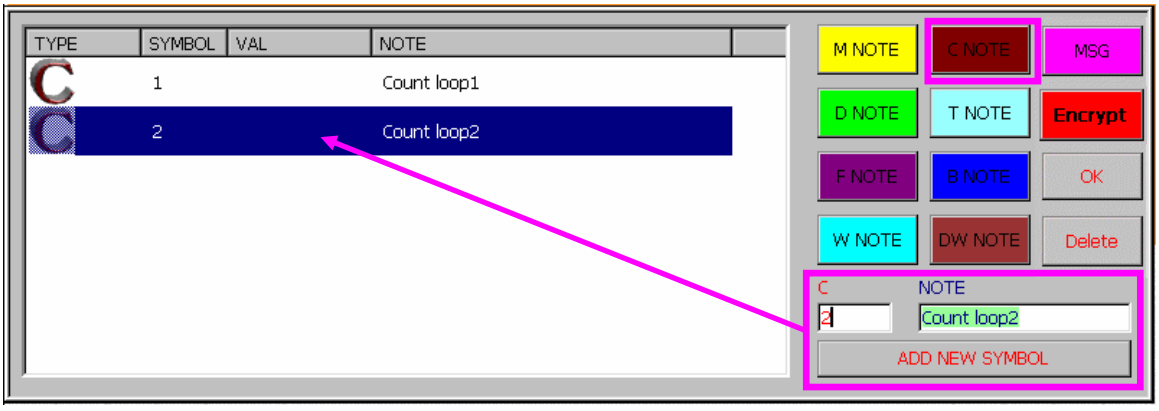
請配合 3.4.7 章節的 AO0 輸出，AI4 輸入電壓 3.5 Volt。
例如上述工程單位為公斤，設 0.5 Kg 代表電壓 1V，並補正-0.5V。
輸入值: Input Value = (Real VI + Offset) x Multiple
(3.5V + (-0.5V)) x 0.5 Kg/V = 1.5Kg
得到輸入值 1.5Kg

3.5 Edit_Button 功能說明

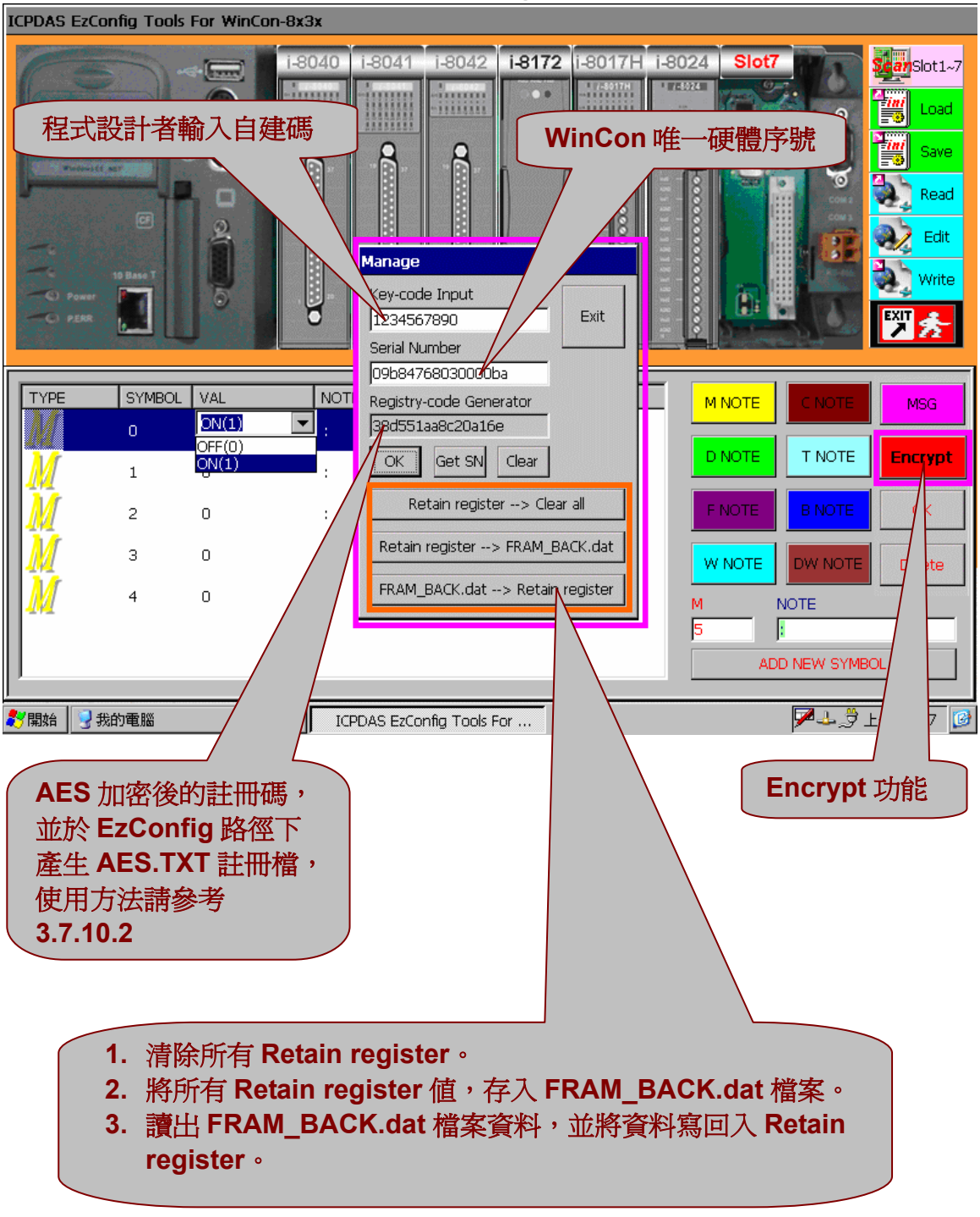
- 編輯所有 Rigister : M、D、F、W、C、T、B、DW、MSG 的預設值、NOTE。



C(計數器)和 T(計時器)無預設值設定



- **AES 加密功能，及 Retain register 的管理。**



3.6 Write_Button 功能說明

作完規劃後可以將資料存入 CF 卡中供日後使用，輸出文字檔為:
(\EzProg_Path\EzProg-I\EzConfig\IO_Table.TXT)，
可以用文字編輯器，或用 Excel 開啟轉換。
亦同時將規劃資料輸出 XML 檔為:
(\EzProg_Path\ EzProg-I\EzConfig \IO_Table.XML)，
可以使用 Office 2003 Excel 直接開啟。

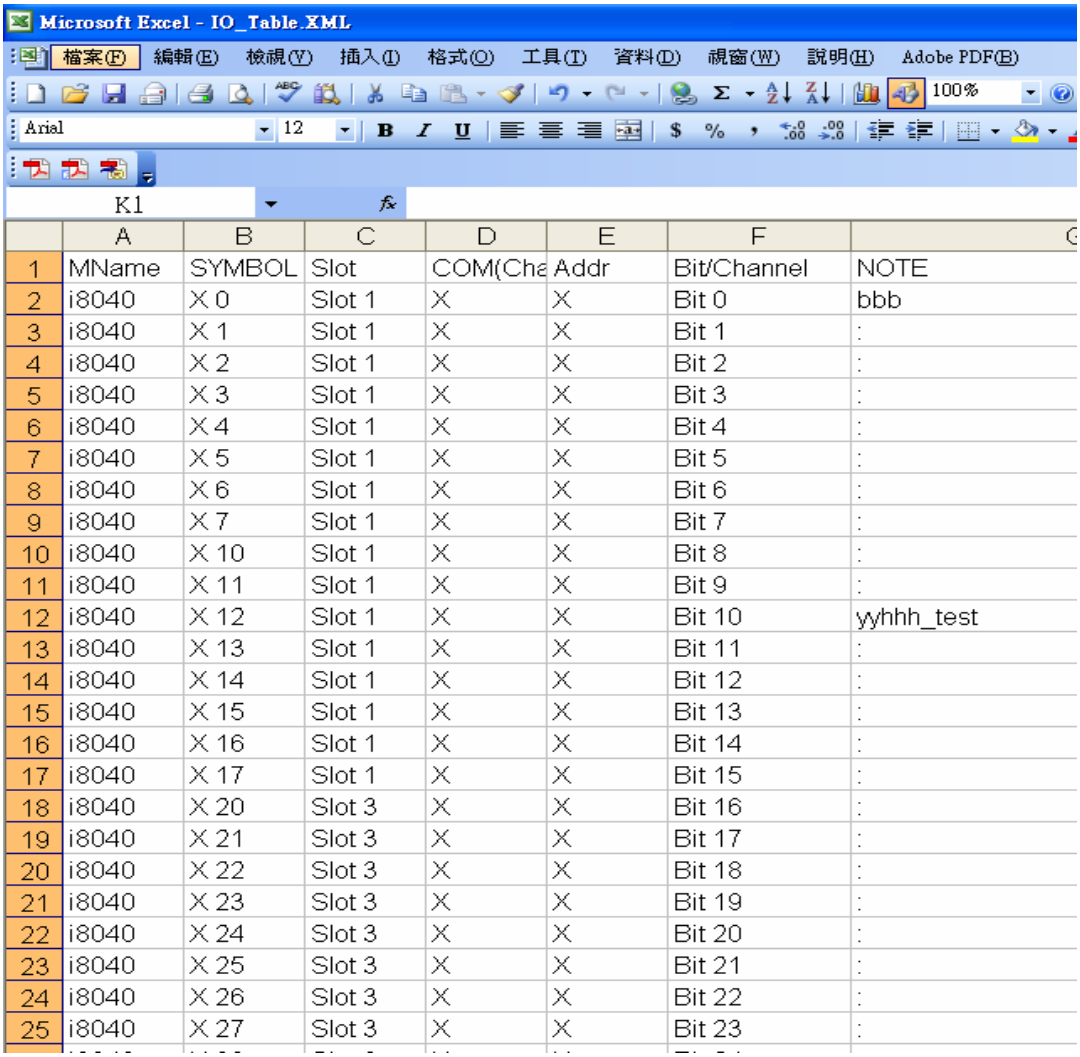
IO_Table.TXT

標準文字檔資料，可供其它應用程式讀檔運用

MName ,SYMBOL ,Slot ,COM(Channel) ,Addr ,Bit/Channel ,Val ,Note
i8040,X 0 ,Slot1 ,X,X,Bit 0,X,bbb
i8040,X 1 ,Slot1 ,X,X,Bit 1,X,:
i8040,X 2 ,Slot1 ,X,X,Bit 2,X,:
i8040,X 3 ,Slot1 ,X,X,Bit 3,X,:
i8040,X 4 ,Slot1 ,X,X,Bit 4,X,:
i8040,X 5 ,Slot1 ,X,X,Bit 5,X,:
i8040,X 6 ,Slot1 ,X,X,Bit 6,X,:
i8040,X 7 ,Slot1 ,X,X,Bit 7,X,:
……
……

IO_Table.XML

可直接在 Excel 編輯 IO_Table.XML，重新 Upload 更新 Val 的值或 NOTE 備註資料



	A	B	C	D	E	F	G
1	MName	SYMBOL	Slot	COM(Cha	Addr	Bit/Channel	NOTE
2	i8040	X 0	Slot 1	X	X	Bit 0	bbb
3	i8040	X 1	Slot 1	X	X	Bit 1	:
4	i8040	X 2	Slot 1	X	X	Bit 2	:
5	i8040	X 3	Slot 1	X	X	Bit 3	:
6	i8040	X 4	Slot 1	X	X	Bit 4	:
7	i8040	X 5	Slot 1	X	X	Bit 5	:
8	i8040	X 6	Slot 1	X	X	Bit 6	:
9	i8040	X 7	Slot 1	X	X	Bit 7	:
10	i8040	X 10	Slot 1	X	X	Bit 8	:
11	i8040	X 11	Slot 1	X	X	Bit 9	:
12	i8040	X 12	Slot 1	X	X	Bit 10	yyhhh_test
13	i8040	X 13	Slot 1	X	X	Bit 11	:
14	i8040	X 14	Slot 1	X	X	Bit 12	:
15	i8040	X 15	Slot 1	X	X	Bit 13	:
16	i8040	X 16	Slot 1	X	X	Bit 14	:
17	i8040	X 17	Slot 1	X	X	Bit 15	:
18	i8040	X 20	Slot 3	X	X	Bit 16	:
19	i8040	X 21	Slot 3	X	X	Bit 17	:
20	i8040	X 22	Slot 3	X	X	Bit 18	:
21	i8040	X 23	Slot 3	X	X	Bit 19	:
22	i8040	X 24	Slot 3	X	X	Bit 20	:
23	i8040	X 25	Slot 3	X	X	Bit 21	:
24	i8040	X 26	Slot 3	X	X	Bit 22	:
25	i8040	X 27	Slot 3	X	X	Bit 23	:

3.7 使用 EzCore 的應用程式庫

EzCore 除內建的功能外，還可以依客戶需求選用各種型式的 i8K IO 模組，如數位輸入(DI)、數位輸出(DO)、類比輸入(AI)、類比輸出(AO)、多軸運動控制(Motion)、高速分散式 DIO(FRNET)，而這些模組經適當規劃後(EzConfig 自動規劃)，在 EzCore 中我們就可以用統一的方法，輕易的使用它們，縮短專案規劃的時間，亦可達到設計標準化的目的。

EzProg-定義的 IO 與暫存器請參閱 1.3 章節

3.7.1 數位及類比控制 DI/O AI/O

3.7.1.1 輸出 DO

- **void OUT_Y(WORD** *DOno*, **bool** *Flag*)

功能: DO Y 點輸出。

參數:		
	DOno	DO 號碼: 0000 ~ 7777(8 進位)
	Flag	true 輸出 ON，false 輸出 OFF

回應: 無

範例:

3.7.1.2 讀回 DO (a 接點)狀態

- **bool GET_Ya(WORD** *DOno*)

功能: 取回 DO 輸出 Ya 接點值。

參數:		
	DOno	DO 號碼: 0000 ~ 7777(8 進位)

回應:		
	true	Ya 輸出狀態為 ON
	false	Ya 輸出狀態為 OFF

範例:

3.7.1.3 讀回 DO (b 接點)狀態

- **bool GET_Yb(WORD** *DOno*)

功能: 取回 DO 輸出 Yb 接點值。

參數:		
	DOno	DO 號碼: 0000 ~ 7777(8 進位)

回應:		
	true	Yb 狀態為 ON
	false	Yb 狀態為 OFF

範例:

3.7.1.4 讀回 DI (a 接點)狀態

- **bool IN_Xa(WORD *DIno*)**

功能: 取回 DI X 輸入 a 接點值。

參數:	*DIno*	DI 號碼: 0000 ~ 7777(8 進位)
回應:	true	Xa 狀態為 ON
	false	Xa 狀態為 OFF

範例:

3.7.1.5 讀回 DI (b 接點)狀態

- **bool IN_Xb(WORD *DIno*)**

功能: 取回 DI X 輸入 b 接點值。

參數:	*DIno*	DI 號碼: 0000 ~ 7777(8 進位)
回應:	true	Xb 狀態為 ON
	false	Xb 狀態為 OFF

範例:

3.7.1.6 輸出 AO

- **void OUT_AO(WORD** *AOno*, **float** *Vout*)

功能: **AO** 點輸出。

參數: ***AOno*** **AO** 號碼: **0 ~ 511**
Vout 輸出值

回應: 無

範例:

3.7.1.7 讀回 AO 輸出值

- **float GET_AO(WORD** *AOno*)

功能: 取回 **AO** 輸出值。

參數: ***AOno*** **AO** 號碼: **0 ~ 511**

回應: 輸出值

範例:

3.7.1.8 讀回 AI 輸入值

- **float IN_AI(WORD** *AIno***)**

功能: 取回 **AI** 輸入值。

參數: ***AIno*** **AI 號碼: 0 ~ 511**

回應: **AI 輸入值**

範例:

3.7.2 計時器功能 Timer

3.7.2.1 設定 Timer 計時器

- **void SET_T(BYTE** *Tno*, **bool** *Flag*, **DWORD** *ms*)

功能: 啟動或關閉計時器。

參數:		
	Tno	**TIMER 號碼: 0 ~ 299**
	Flag	**true 啟動，false 關閉**
	ms	**TIMER 設定時間: 1 ~ 4,294,967,295 ms**

回應: 無

範例:

3.7.2.2 讀回 Timer 倒數計時值

- **DWORD GET_T(BYTE** *Tno*)

功能: 讀回 Timer 倒數計時值。

參數:		
	Tno	**TIMER 號碼: 0 ~ 299**
回應:		**TIMER 倒數計時值: 0 ~ 4,294,967,295 ms**

範例:

3.7.2.3 讀回 Timer (a 接點)狀態

- **bool GET_Ta(BYTE *Tno*)**

功能: 讀回 Timer a 接點狀態。

參數:	*Tno*	TIMER 號碼: 0 ~ 299
回應:	true	Ta 狀態為 ON
	false	Ta 狀態為 OFF

範例:

3.7.2.4 讀回 Timer (b 接點)狀態

- **bool GET_Tb(BYTE *Tno*)**

功能: 讀回 Timer b 接點狀態。

參數:	*Tno*	TIMER 號碼: 0 ~ 299
回應:	true	Tb 狀態為 ON
	false	Tb 狀態為 OFF

範例:

3.7.3 計數器功能 Counter

3.7.3.1 設定 Counter 計數器

- **void SET_C(WORD** *Cno*, **bool** *Flag*, **DWORD** *COUNT*)

功能: 啟動或下數計數器。

參數:		
	Cno	**Counter 號碼: 0 ~ 511、512 ~ 1023 為斷電保持型**
	Flag	**true 啟動，false 下數**
	COUNT	**設定 COUNT 數: 1 ~ 4,294,967,295**

回應: 無

範例:

3.7.3.2 重置 Counter 計數器

- **void RESET_C(WORD** *Cno*)

功能: 重置計數器。

參數:		
	Cno	**Counter 號碼: 0 ~ 511、512 ~ 1023 為斷電保持型**

回應: 無

範例:

3.7.3.3 讀回 Counter 倒數計數值

- **DWORD GET_C(WORD *Cno*)**

功能: 讀回 Counter 倒數計數值。

參數:	*Cno*	Counter 號碼: 0 ~ 511、512 ~ 1023 為斷電保持型
回應:		Counter 倒數計數值: 0 ~ 4,294,967,295

範例:

3.7.3.4 讀回 Counter (a 接點)狀態

- **bool GET_Ca(WORD *Cno*)**

功能: 讀回 Counter 計數器 a 接點，Ca 是否 ON。

參數:	*Cno*	Counter 號碼: 0 ~ 511、512 ~ 1023 為斷電保持型
回應:	true	Ca 狀態為 ON
	false	Ca 狀態為 OFF

範例:

3.7.3.5 讀回 Counter (b 接點)狀態

- **bool GET_Cb(WORD *Cno*)**

功能: 查詢 Counter 計數器 b 接點，Cb 是否 ON。

參數:	***Cno***	**Counter 號碼: 0 ~ 511、512 ~ 1023 為斷電保持型**
回應:	**true**	**Cb 狀態為 ON**
	false	**Cb 狀態為 OFF**

範例:

3.7.4 步進程序 Step 功能

3.7.4.1 設定 Step 旗標

- **void SET_S(WORD** *Sno***)**

功能: 設定 S 步進旗標 On 狀態。

參數: *Sno* S 旗標號碼: 0 ~ 8191

回應: 無

範例:

3.7.4.1 清除 Step 旗標

- **void RST_S(WORD** *Sno***)**

功能: 清除 S 步進旗標為 Off 狀態。

參數: *Sno* S 旗標號碼: 0 ~ 8191

回應: 無

範例:

3.7.4.2 讀回 Step 旗標狀態

- **bool GET_S(WORD** *Sno***)**

功能: 取回 S 旗標 On/Off 狀態。

參數: *Sno* S 旗標號碼: 0 ~ 8191

回應: true S 狀態為 ON
false S 狀態為 OFF

範例:

3.7.5 軟體旗標功能 M

3.7.5.1 設定 M 旗標值

- **void** SET_M(WORD *Mno*, **bool** *Flag*)

功能: 設定 M 旗標 On/Off 狀態。

參數:

	Mno	M 旗標號碼: 0 ~ 8191、8192 ~ 16383 為斷電保持型 M16000 ~ M16383 為系統內定使用 M16000 為 ColorEdit 控件輸入介面切換， true=控件內部鍵盤。
	Flag	true ON，false OFF

回應: 無

範例:

3.7.5.2 讀回 M (a 接點)

- **bool** GET_Ma(WORD *Mno*)

功能: 取回 M 旗標 a 接點 On/Off 狀態。

參數:

	Mno	M 旗標號碼: 0 ~ 8191、8192 ~ 16383 為斷電保持型

回應:

true	Ma 狀態為 ON
false	Ma 狀態為 OFF

範例:

3.7.5.2 讀回 M (b 接點)

- **bool** GET_Mb(WORD *Mno*)

功能: 取回 M 旗標 b 接點 On/Off 狀態。

參數:

	Mno	M 旗標號碼: 0 ~ 8191、8192 ~ 16383 為斷電保持型

回應:

true	Mb 狀態為 ON
false	Mb 狀態為 OFF

範例:

3.7.6 一般暫存器功能 D

3.7.6.1 設定 D 一般暫存器值

- **void SET_D(WORD** *Dno*, **long** *Val*)

功能: 指定 D 暫存器值。

參數: ***Dno*** D 暫存器號碼: 0 ~ 4095、4096 ~ 8191 為斷電保持型
D8000 ~ D8191 為系統內定使用:
D8000 : 0 ~ 7 為八國語系選擇
D8100 : 為 FRAM W/R ERROR

Val 帶符號 32 位元值: -2,147,483,648 ~ 2,147,483,647

回應: 無

範例:

3.7.6.2 讀回 D 一般暫存器值

- **long GET_D(WORD** *DNo*)

功能: 取回 D 暫存器值。

參數: ***DNo*** D 暫存器號碼: 0 ~ 4095、4096 ~ 8191 為斷電保持型

回應: D 暫存器值: -2,147,483,648 ~ 2,147,483,647

範例:

3.7.7 資料暫存器功能 B、W、DW、F

3.7.7.1 設定 B 暫存器值

- **void SET_B(WORD** *Bno*, **BYTE** *data*)

功能: 指定 8 位元資料不帶符號 B 暫存器值。

參數: ***Bno*** B 暫存器號碼: 0 ~ 1023、1024 ~ 2047 為斷電保持型
data 8 位元資料: 0 ~ 255

回應: 無

範例:

3.7.7.2 讀回 B 暫存器值

- **BYTE GET_B(WORD** *Bno*)

功能: 取回 8 位元資料不帶符號 B 暫存器值。

參數: ***Bno*** B 暫存器號碼: 0 ~ 1023、1024 ~ 2047 為斷電保持型

回應: B 暫存器 8 位元資料: 0 ~ 255

範例:

3.7.7.3 設定 W 暫存器值

- **void SET_W(WORD** *Wno*, **WORD** *data*)

功能: 指定 16 位元資料不帶符號 W 暫存器值。

參數:	*Wno*	W 暫存器號碼: 0 ~ 1023、1024 ~ 2047 為斷電保持型
	data	16 位元資料: 0 ~ 65,535

回應: 無

範例:

3.7.7.4 讀回 W 暫存器值

- **WORD GET_W(WORD** *Wno*)

功能: 取回 16 位元資料不帶符號 W 暫存器值。

參數:	*Wno*	W 暫存器號碼: 0 ~ 1023、1024 ~ 2047 為斷電保持型
回應:		W 暫存器 16 位元資料: 0 ~ 65,535

範例:

3.7.7.5 設定 DW 暫存器值

- **void SET_DW(WORD** *DWno*, **DWORD** *data*)

功能: 指定 32 位元資料不帶符號 DW 暫存器值。

參數: ***DWno*** DW 暫存器號碼: 0 ~ 4095、4096 ~ 8191 為斷電保持型
data 32 位元資料: 0 ~ 4,294,967,295

回應: 無

範例:

3.7.7.6 讀回 DW 暫存器值

- **DWORD GET_DW(WORD** *DWno*)

功能: 取回 32 位元資料不帶符號 DW 暫存器值。

參數: ***DWno*** DW 暫存器號碼: 0 ~ 4095、4096 ~ 8191 為斷電保持型

回應: DW 暫存器 32 位元資料: 0 ~ 4,294,967,295

範例:

3.7.7.7 設定 F 暫存器值

- **void SET_F(WORD** *Fno*, **float** *data*)

功能: 指定 F 浮點暫存器值。

參數:	*Fno*	F 暫存器號碼: 0 ~ 2047、2048 ~ 4095 為斷電保持型
	data	7 位數浮點資料: 3.4E +/- 38 (7 digits)

回應: 無

範例:

3.7.7.8 讀回 F 暫存器值

- **float GET_F(WORD** *Fno*)

功能: 取回 F 浮點暫存器值。

參數:	*Fno*	F 暫存器號碼: 0 ~ 2047、2048 ~ 4095 為斷電保持型
回應:		F 浮點暫存器值: 3.4E +/- 38 (7 digits)

範例:

3.7.8 資料區塊暫存器功能 DB

	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	.....	1	0
	bit	bit	bit	bit	bit	bit	bit	bit	bit	bit	bit	bit	bit	bit	bit	bit	bit	bit	bit	bit	bit	bit	bit	bit	bit	bit	bit	bit	bit	bit	bit	bit	.....		
DB	BYTE(15)								BYTE(14)								BYTE(13								BYTE(12)								.....		
	WORD(7)																WORD(6)																.....		
	DWORD(3)																																.....		

3.7.8.1 設定 1 位元資料到 DB

- **void DB_SETx1(BYTE** *DBno*, **BYTE** *Num*, **bool** *data*)

功能: 移動 1 位元資料區塊，到指定的 DB 暫存器。

參數:

	DBno	**DB 暫存器號碼: 0 ~ 49**
	Num	**區段號碼: 0 ~ 127**
	data	**1 位元資料: 0，1**

回應: 無

範例:

3.7.8.2 讀回 DB 暫存器 1 位元資料

- **bool DB_GETx1(BYTE** *DBno*, **BYTE** *Num*)

功能: 從 DB 暫存器讀取 1 位元資料。

參數:

	DBno	**DB 暫存器號碼: 0 ~ 49**
	Num	**區段號碼: 0 ~ 127**

回應: **DB 暫存器 1 位元資料: 0，1**

範例:

3.7.8.3 設定 8 位元資料到 DB

- **void DB_SETx8(BYTE** *DBno*, **BYTE** *Num*, **BYTE** *data*)

功能: 移動 8 位元資料區塊，到指定的 DB 暫存器。

參數:

	DBno	**DB 暫存器號碼: 0 ~ 49**
	Num	**區段號碼: 0 ~ 15**
	data	**8 位元資料: 0 ~ 255**

回應: 無

範例:

3.7.8.4 讀回 DB 暫存器 8 位元資料

- **BYTE DB_GETx8(BYTE** *DBno*, **BYTE** *Num*)

功能: 從 DB 暫存器讀取 8 位元資料。

參數:

	DBno	**DB 暫存器號碼: 0 ~ 49**
	Num	**區段號碼: 0 ~ 15**

回應: **DB 暫存器 8 位元資料: 0 ~ 255**

範例:

3.7.8.5 設定 16 位元資料到 DB

- **void DB_SETx16(BYTE** *DBno*, **BYTE** *Num*,**WORD** *data*)

功能: 移動 16 位元資料區塊，到指定的 DB 暫存器。

參數:		
	DBno	**DB 暫存器號碼: 0 ~ 49**
	Num	**區段號碼: 0 ~ 7**
	data	**16 位元資料: 0 ~ 65,535**

回應: 無

範例:

3.7.8.6 讀回 DB 暫存器 16 位元資料

- **WORD DB_GETx16(BYTE** *DBno*, **BYTE** *Num*)

功能: 從 DB 暫存器讀取 16 位元資料。

參數:		
	DBno	**DB 暫存器號碼: 0 ~ 49**
	Num	**區段號碼: 0 ~ 7**

回應: **DB 暫存器 16 位元資料: 0 ~ 65,535**

範例:

3.7.8.7 設定 32 位元資料到 DB

- **void DB_SETx32(BYTE** *DBno*, **BYTE** *Num*, **DWORD** *data*)

功能: 移動 32 位元資料區塊，到指定的 DB 暫存器。

參數:		
	DBno	**DB 暫存器號碼: 0 ~ 49**
	Num	**區段號碼: 0 ~ 3**
	data	**32 位元資料: 0 ~ 4,294,967,295**

回應: 無

範例:

3.7.8.8 讀回 DB 暫存器 32 位元資料

- **DWORD DB_GETx32(BYTE** *DBno*, **BYTE** *Num*)

功能: 從 DB 暫存器讀取 32 位元資料。

參數:		
	DBno	**DB 暫存器號碼: 0 ~ 49**
	Num	**區段號碼: 0 ~ 3**

回應: **DB 暫存器 32 位元資料: 0 ~ 4,294,967,295**

範例:

3.7.8.9 左移 DB 暫存器位元

- **void DB_SL(BYTE** *DBno*, **BYTE** *Shift*)

功能: 位移 DB 暫存器位元。

參數: ***DBno*** DB 暫存器號碼: 0 ~ 49
Shift ←左移位元數: 1 ~ 128

回應: 無

範例:

3.7.8.10 右移 DB 暫存器位元

- **void DB_SR(BYTE** *DBno*, **BYTE** *Shift*)

功能: 位移 DB 暫存器位元。

參數: ***DBno*** DB 暫存器號碼: 0 ~ 49
Shift →右移位元數: 1 ~ 128

回應: 無

範例:

3.7.9 訊息資料讀寫

EzCore 訊息可以讓系統交換訊息,可以用於應用程序與人機界面做溝通。

3.7.9.1 訊息資料寫入

- **long SET_MSG(WORD** *MSGno*,**TCHAR** *UMSG[30]*);

功能: 訊息資料寫入。

參數:

	MSGno	訊息號碼，0 ~ 249 皆為斷電保持型 MSG245~249 為 Password 輸入專用
	UMSG[30]	欲寫入訊息內容，限 30 個字

回應:

0	執行成功
其他值	請參閱附錄一錯誤碼表

範例:

```
//1.直接用 UNICODE 字串傳入
RET= SET_MSG(100, _T("泓格科技訊息"));

//2.或者用 CString 傳入
CString CS= _T("泓格科技訊息");
TCHAR UMSG[30];
swprintf(UMSG, CS); //使用 swprintf( ); 將 CString 轉 TCHAR
                    //CS 的字數不可超過 UMSG 陣列數量
RET= SET_MSG(101, UMSG);

//3.在訊息開頭加入時間碼→ 10:20:55泓格科技訊息
CEzLIB EzLIB;
TCHAR tcTime[15];
EzLIB.Get_Time(tcTime);
CString HMSm(tcTime);
CString CS= _T(" 泓格科技訊息");
CS= HMSm + CS;
TCHAR UMSG[30];
swprintf(UMSG, CS);
ret= SET_MSG(102, UMSG);

//4.在訊息開頭加入日期碼→ 2007/04/26 泓格科技訊息
TCHAR tcDate[15];
EzLIB.Get_Date(tcDate);
CString YMD(tcDate);
CString CS1= _T(" 泓格科技訊息");
CS1= YMD + CS1;
TCHAR UMSG1[30];
swprintf(UMSG1, CS1);
ret= SET_MSG(103, UMSG1);
```

```
//5.在訊息開頭加入日期和時間碼→ 2007/04/26 10:20:55 泓格科技訊息
TCHAR tcDT[30];
EzLIB.Get_DT(tcDT, true, true, false, true, true, true, true);
CString YMWDHMSm(tcDT);
CString CS2= _T(" 泓格科技訊息");
CS2= YMWDHMSm + CS2;
TCHAR UMSG2[30];
swprintf(UMSG2, CS2);
ret= SET_MSG(104, UMSG2);
```

3.7.9.2 訊息資料讀取

- **long GET_MSG(WORD** *MSGno*,**TCHAR** *UMSG[30]*);

功能: 訊息資料讀取。

參數:	*MSGno*	訊息號碼，0 ~ 249 皆為斷電保持型 MSG245~249 為 Password 輸入專用
	UMSG[30]	欲讀取訊息內容 30 個字
回應:	0	執行成功
	其他值	請參閱附錄一錯誤碼表

範例:
```
TCHAR UMSG[30];
RET= GET_MSG(100, UMSG);
CString CS(UMSG);  //使用 CString( ); 將 TCHAR 轉 CString
```

3.7.9.3 讀取多語系文字檔寫到 MSG 訊息

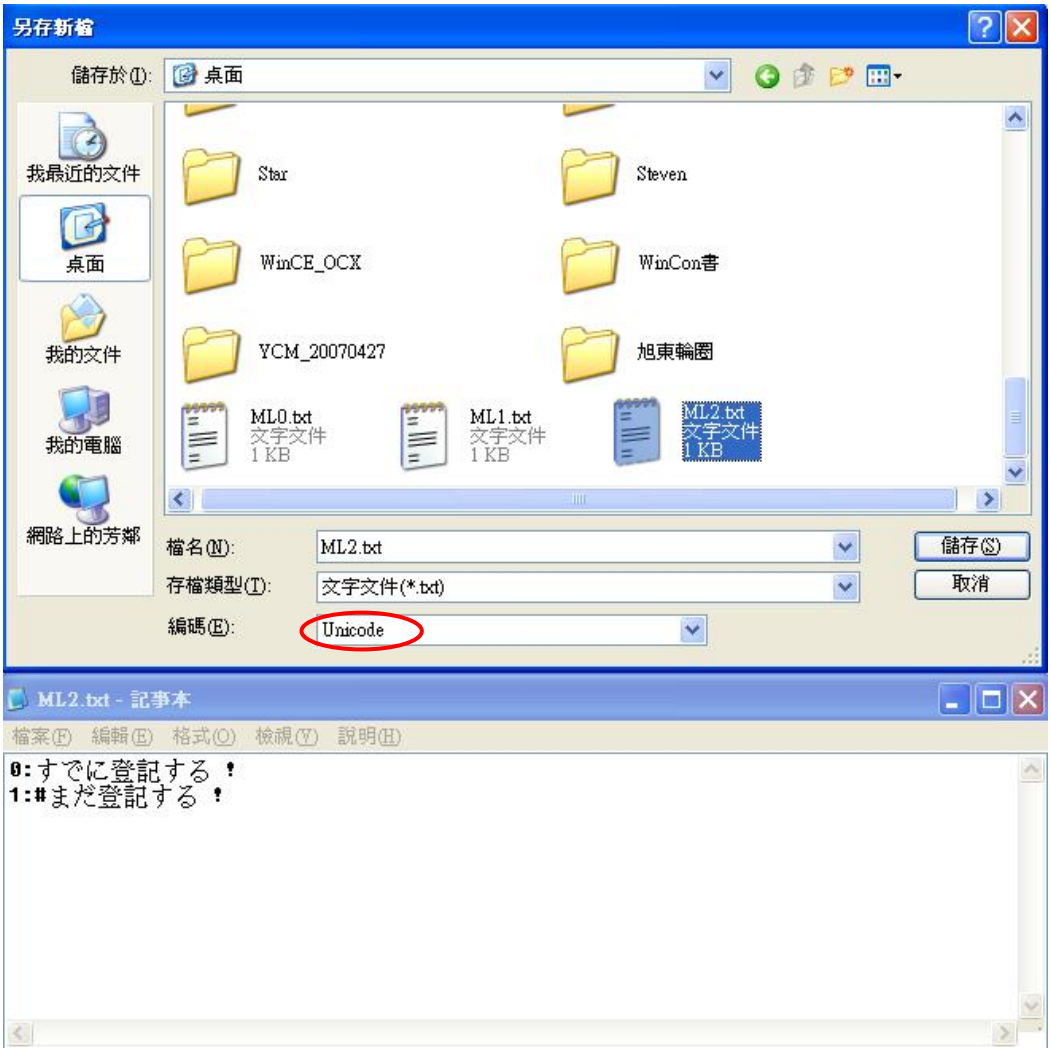
EzCore 提供多國語系訊息，可以讓系統依不同語系使用不同訊息，可以用於應用程序與人機界面多國語系切換用，當 D8000 被修改時，稍後立即自行變更。

而多國語系檔案用 Unicode 分別存為 ML0.txt，ML1.txt ~ML7.txt，分別對應 Multi-Language 0~7 八個語系文字檔，檔案需事先置於 PAC 如下路徑中: EzProg_Path\EzProg-I\EzHMI\ML\ML0~7.txt，其每一語系可以包括 0~999 則訊息，每一個訊息上限 30 個字，格式如下述。

ML0.txt ➔
0:系統初始化 OK !!
1:使用者登入成功 !!

ML1.txt ➔
0:System Initial OK !!
1:User login OK !!

":" 前數字為訊息號碼 (*MSGFno)*
":" 後 30 個字為訊息內容，每則以 CrLf 為終止



● **long SET_MSGF(WORD** *MSGno*, **WORD** *MSGFno*);

功能: 多國語系訊息資料讀取到訊息資料(MSG)。

參數:

	MSGno	訊息號碼，**0 ~ 249** 皆為斷電保持型
	MSGFno	多國語系訊息號碼，**0 ~ 999**

回應:

	0	執行成功
	其他值	請參閱附錄一錯誤碼表

範例:

```
RET= SET_MSGF(100, 0);
//將多國語系訊息檔 MLn.txt 的第 0 則訊息; Set 到 MSG100 號訊息中
```

3.7.9.4 讀取多語系文字檔資料

● **long GET_MSGF(WORD** *MSGFno*,**TCHAR** *UMSG[30]*);

功能: 讀取多國語系訊息。

參數:

	MSGFno	多國語系訊息號碼，**0 ~ 999**
	UMSG[30]	欲讀取訊息內容 **30** 個字

回應:

	0	執行成功
	其他值	請參閱附錄一錯誤碼表

範例:

```
TCHAR UMSG[30];
RET= GET_MSGF(0, UMSG); //將 MLn.txt 的第 0 則訊息; 傳到 UMSG
CString CS(UMSG);  //使用 CString( ); 將 TCHAR 轉 CString
```

3.7.10 使用 AES 加密的系統保全

PAC EzCore 內建一方便的 AES 加密機制功能，軟體開發者可以輕易的保護合法軟體，而此機制是利用軟體開發者指定的金鑰(AES_KEY)作加解密，並結合 PAC 的唯一硬體序號做認證。

本章節提供兩種簡易的方法自由選擇使用，即可操作 EzCore AES 的認證，並在 EzConfig(3.5 章節)中提供方便的操作介面(依輸入 PAC 的唯一硬體序號及加密金鑰產生註冊碼)，輕易達成系統保全功能保護著作財產權。

3.7.10.1 將註冊碼註冊到 EzCore 系統中

long REGISTRY_KEY(TCHAR *REG[20]*)

功能: 將軟體開發者透過提供 EzConfig 所產生的註冊碼，合法授權給 End-user。再由 end-user 輸入給應用程式，然後應用程式可以使用 REGISTRY_KEY( )對 EzCore 註冊。

參數:	*REG[20]*	1.軟體開發者提供的註冊碼 16 個字串指標 2.或由 MSGno 傳入 EzConfig/Edit/Manage→Registry-codeGenerator
回應:	0	輸入語法正確
	其他值	請參閱附錄一錯誤碼表

範例:
```
REGISTRY_KEY(_T("05386f8e9a7b6fa7"));
REGISTRY_KEY(_T("MSG249"));
```

3.7.10.2 將註冊檔(AES.txt)註冊到 EzCore 系統中

long REGISTRY_FILE()

功能: 將軟體開發者透過提供 EzConfig 所產生的註冊檔，路徑在 \EzProg_Path\EzProg-I\EzConfig\AES.txt，合法授權給 End-user。然後應用程式可以使用 REGISTRY_FILE( )對 EzCore 註冊。

回應:	0	輸入語法正確
	其他值	請參閱附錄一錯誤碼表

範例:
```
REGISTRY_FILE();
```

3.7.10.3 檢查 EzCore 系統中註冊碼是否合法

long CHECK_KEY(TCHAR *AES_KEY[20]***)**

功能: 軟體開發者在應用程式中的任何位置，可隨時下 **CHECK_KEY( )**，利用 **EzCore** 自動檢查註冊碼是否合法。

參數:	***AES_KEY[20]***	軟體開發者個人或軟體的加密金鑰 **16** 個字串指標 **EzConfig/Edit/Manage→Key-code Input**
回應:	**0**	檢查正確(註冊碼合法授權)
	其他值	請參閱附錄一錯誤碼表

範例: **long RET= CHECK_KEY(_T("1234567812345678"));**

4 EzGo 介紹

4.1 EzGo 簡介

4.1.1 簡介

隨著 EzProg-I 的發表，已經提供使用者在 PAC 更方便的開發軸控程式，為了讓使用者在未使用過 PAC 的前提下先了解軸控部分的控制方式，或讓使用者用來檢查錯誤，因此誕生了 EzGo，目前 EzGo 已經可以支援 i8092(F) ，i8094(F) 的測試與設定儲存，8094A(H)等模組只有測試作用。

4.1.2 功能說明

EzGo 操作畫面共分四頁。

1. 第一頁是主頁，主要是限定使用者需先進第二頁註冊軸卡和設定參數後再來決定進入第三頁或第四頁測試。

2. 第二頁是基本設定頁，除了註冊軸卡外，還可以針對軸卡的幾個基本設定項來作設定，設定完儲存後(only for i8092(F) ，i8094(F))使用者可以用"LOAD_CONFIG()"來讀取及重新將這些設定恢復成當初的設定狀態。

3. 第三頁是基本操作頁，可以針對軸卡的基本直線動作做測試，也可以用來確認軸卡動作是否有步數錯誤或位置錯誤。

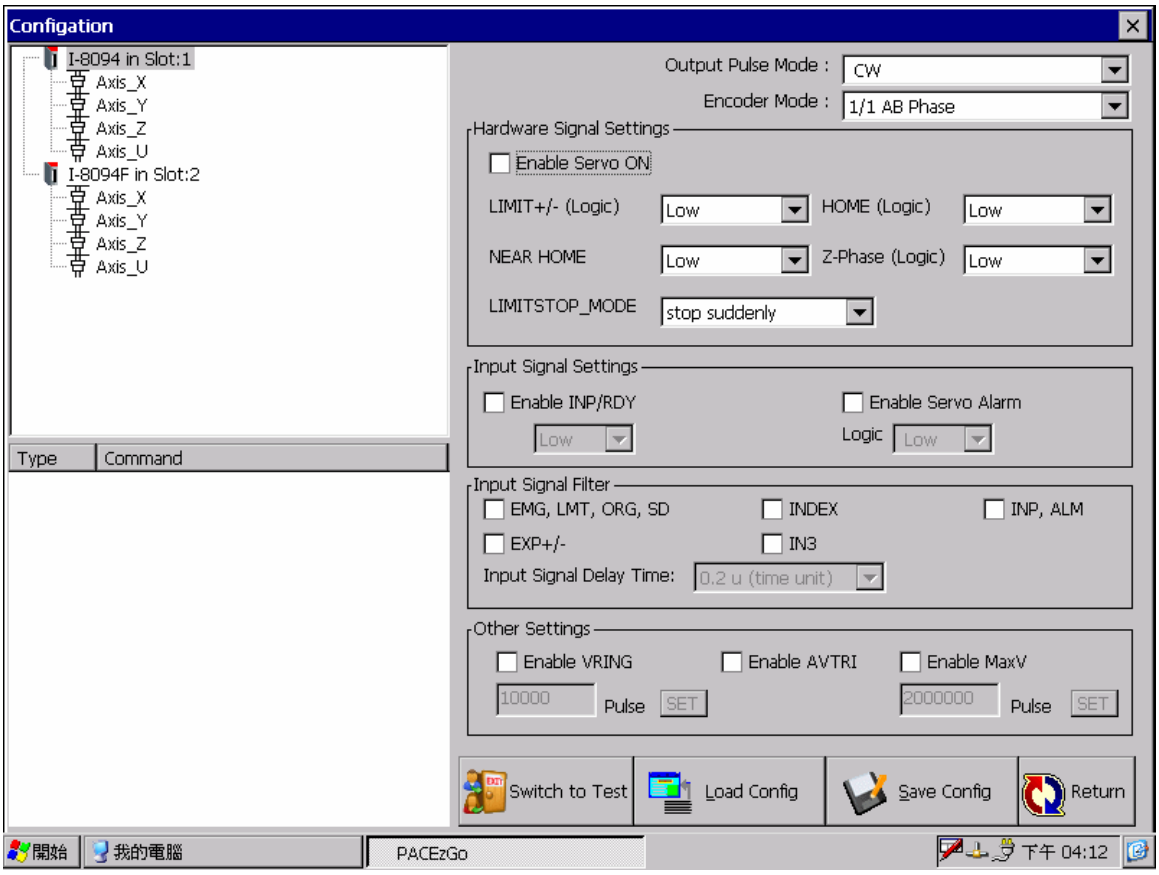
4. 第四頁是進階動作頁，使用者可以在這邊操作多一點的動作變化。

4.2 操作說明

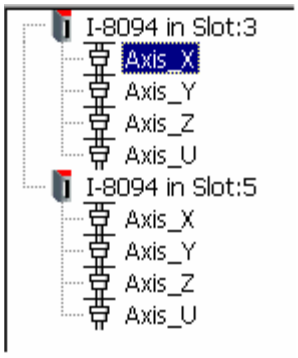
4.2.1 首頁

首頁主要是使用者用來選擇要進入的頁面，第一次進入會限制使用者只能進入設定頁,因為要使用軸卡必須先註冊以及作初始設定,等軸卡註冊後就不再限制使用者只能進入設定頁,**Basic_Operation & Advanced_Features** 這兩個按鈕也會由黑白轉彩色。

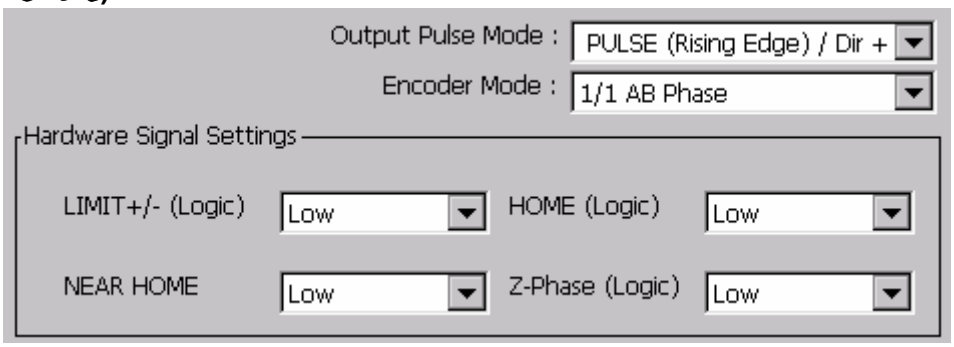
4.2.2 屬性設定頁



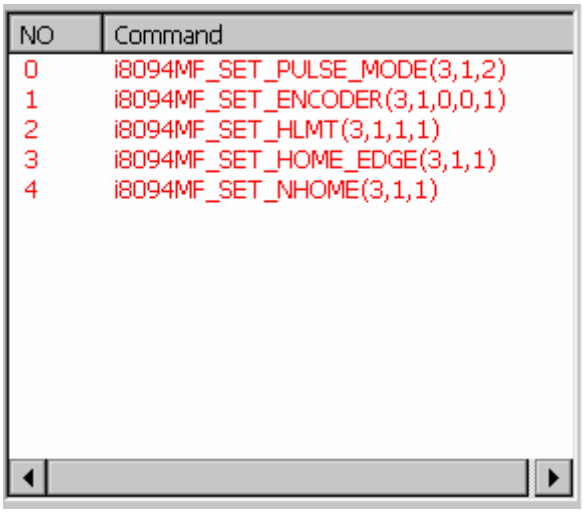
進入屬性設定頁後，請先點擊要設定的軸，點擊後被選擇項會反白，如下圖:



接著，開始在各個設定選擇需要的選項，操作需要的幾個設定(如下圖)，其他設定可以依照需要加入，或者按下最下方的 **LOAD** 鍵，讀取之前的設定(如果以前有做過設定)。



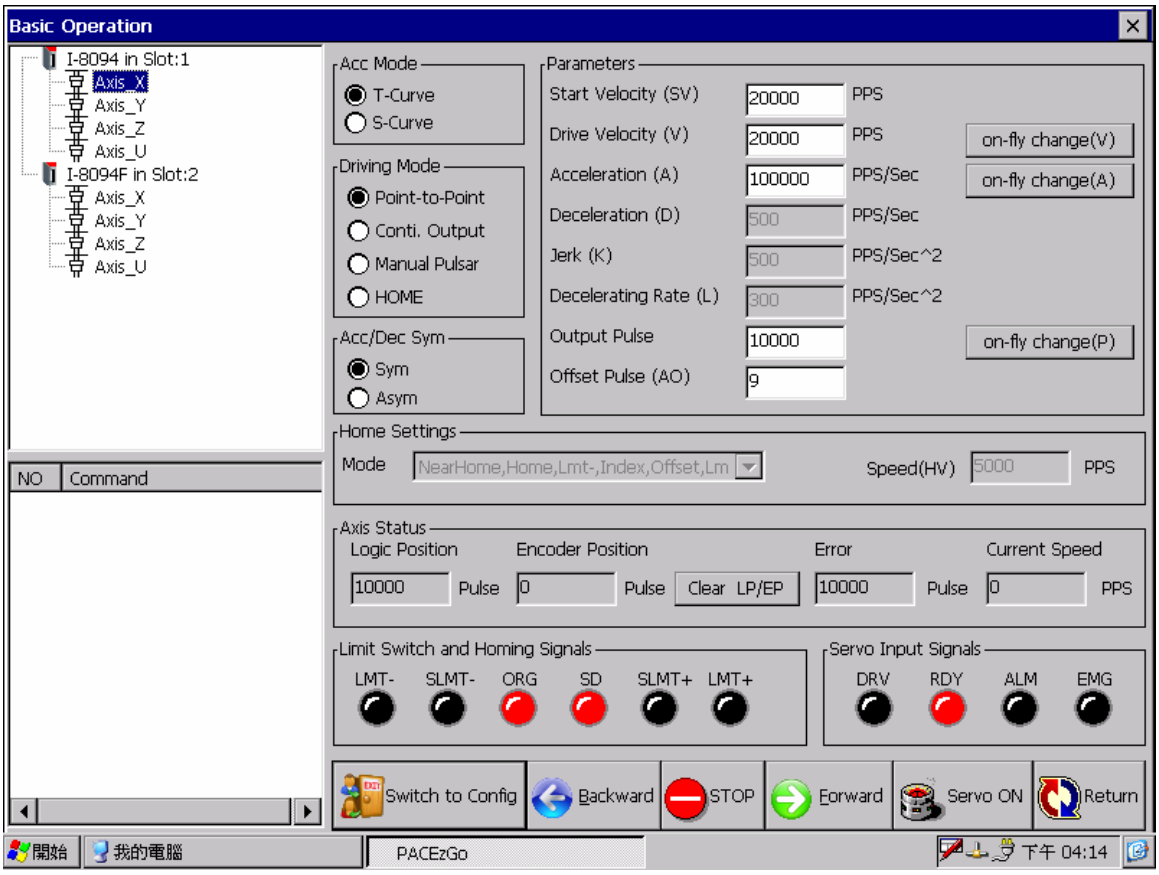
EzGo 會依所選擇的設定發送命令,畫面左下方可以看到所有發出的指令(如下圖)



所有設定設完後可以按 

鍵將設定儲存到 **CF** 卡的 **ini** 檔，**\EzProg_Path\EzProg-I\EzGo\I8094_Config.ini**，接著到基本操作頁測試，或者不儲存直接到基本設定頁測試，按不按 **save** 在 **switch to test** 的動作執行後都會再問一次。

4.2.3 基本操作頁

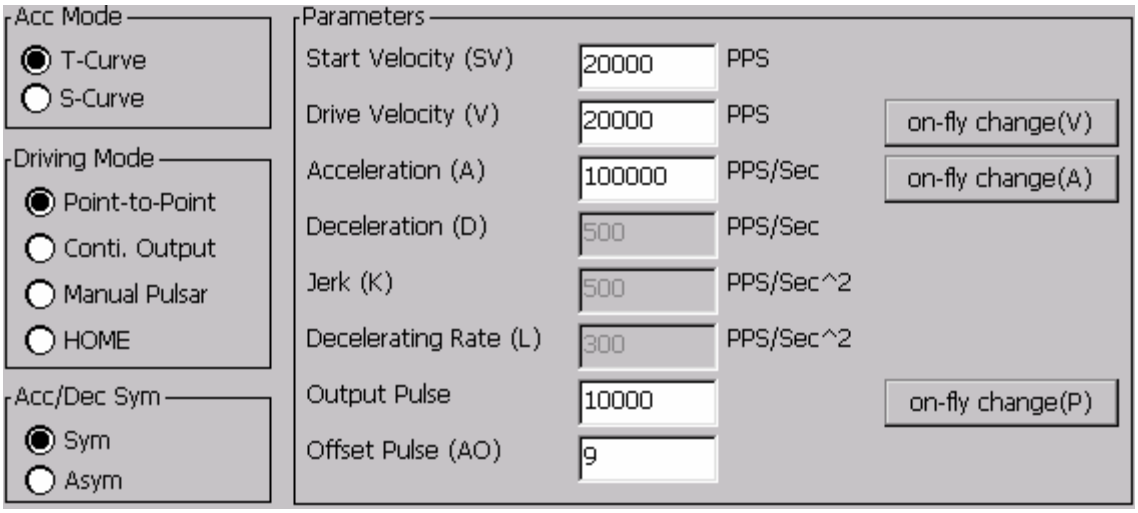


這個頁面主要是讓使用者測試基本直線動作和簡單的偵錯檢查，首先由左上角開始。

- 第一步: 先選擇要動作的卡和軸，基本上此頁會跟隨上一頁最後所選擇的項目。(不過要先提醒使用者，在這頁操作中，要更改動作軸只能在同一張卡，而且每一軸中的參數需已設定；因為要使用軸卡時須先註冊軸卡和設定參數，而在這套軟體中，設計是只能在第一頁設定參數，因此，註冊軸卡也只在第一頁，所以在這一頁更改卡號是無效的。)

- 第二步: 在畫面的左上角選擇加減速模式(**I-8094H 手冊-6.1.1**)，接著往下選擇驅動模式，分別是點對點驅動、連續驅動、手搖輪驅動、歸零，再來選擇加減速曲線是否要對稱。

- 第三步: 當第二步完成；使用者可以發現中間原本不能輸入的項目有幾個項目可以輸入了，中間這部份會依照使用者選擇的驅動模式來開放需要的設定項目，使用者依照要驅動的速度設定完畢後即可。



如果使用者選擇的是歸零;則要設定的只有 **Home Settings** 這個框框內項目。

- 第四步: 先按下 

；接著可以按下 

或是 

來驅動馬達，連續驅動要停止時按下 

。如果是手搖輪驅動則只要 

即可。歸零的話， 

鍵會變成 

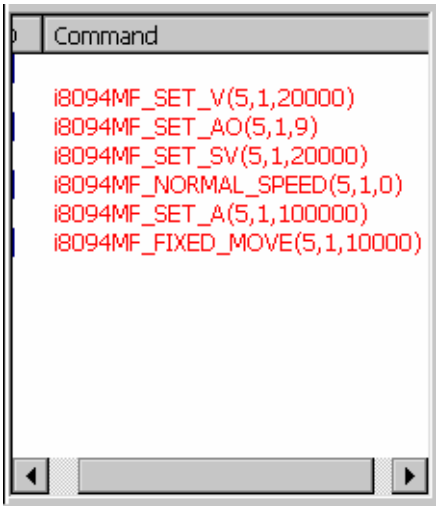
，按下即可歸零。

PS:如果一開始該軸的 **LP & EP** 不為零,需要清除為零只要按下 

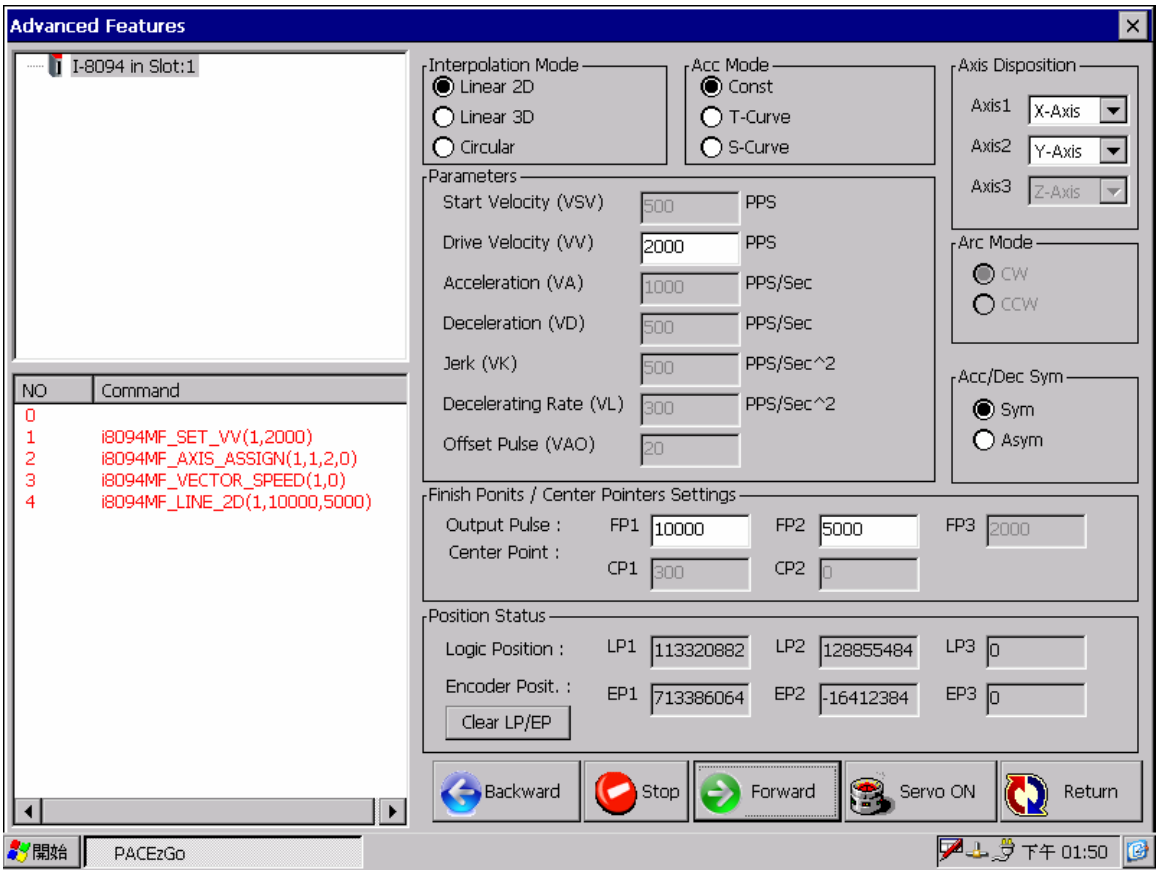
即可。

至於狀態列及燈號列則是顯示當時狀態,譬如邏輯位置、編碼器位置、失步(LP-EP)、現行速度、是否有碰到極限或到達原點、驅動狀態、準備狀態或者有 **Alarm**、**EMG**(緊急停止)...等等。

另外，驅動馬達所有的命令也會顯示在左下角的狀態列。



4.2.4 進階展示頁

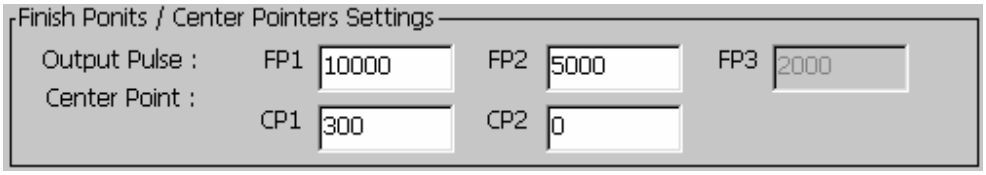


這一頁以展示的功能居多，顯示軸卡做出 2D、3D、圓弧等補間的動作。操作與上一頁類似，設定所需操作即可。

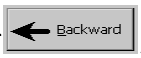
- 第一步: 在左上方選擇要操作的卡號，一開始會依照使用者在第二、三頁所選擇的卡，而且不能在此更換卡號，原因同第三頁所說。

- 第二步: 依序選擇加減速模式、要動作的軸、CW/CCW、對稱與否，中間設定部份就會依照選擇開啟，中間基本設定輸入完畢之後；接著設定每軸要驅動的步數。

- 第三步: 在步數輸入處鍵入所以要驅動步數，或著圓弧中心點。



- 第四步，按下 

，接著按下 

或是 

驅動(如果是圓弧

動作則是  或是 )。

PS:在這頁一進來一樣會顯示該卡被選定各軸的 LP/EP，如果使用者有清除 LP/EP 的需要，一樣可以按  **來達成。**

在 position status 的地方，會顯示邏輯位置及編碼器位置。同樣的，動作的命令會顯示在左下角的顯示列內。

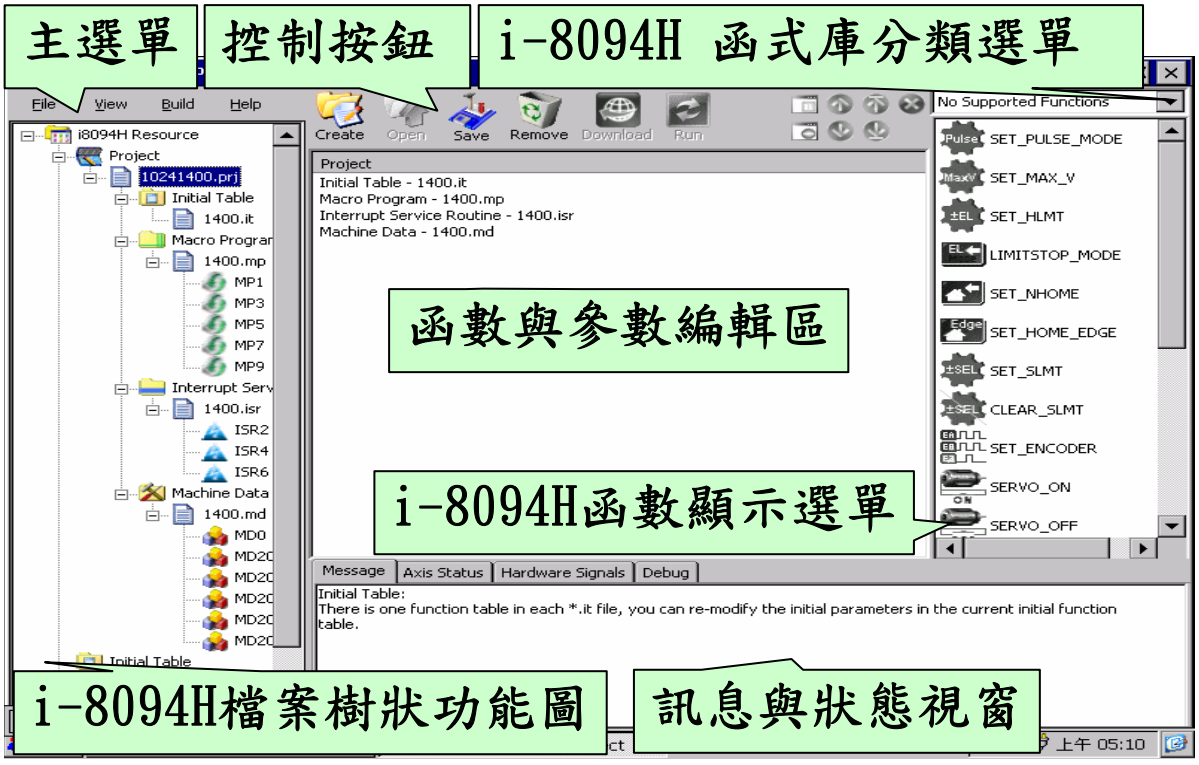
5 EzMake 介紹

5.1 EzMake 功能簡介

EzMake 顧名思義為容易使用、簡單操作的軟體工具，其軟體操作、程式編輯、與運動控制命令模擬，需搭配 i-8094H 四軸智慧型運動控制模組來使用。EzMake 最主要的功能是免除使用者需要使用 eVC 撰寫運動控制程式這項耗時費力的工作，幫助使用者縮短運動控制程式開發的時程。關於運動控命令依照功能分類可選擇不同的圖形化介面的 API 函數，可友善且快速的編輯函數的參數。EzMake 的模組資源樹狀圖上可針對 i-8094H 模組建立四種不同的檔案類型(依軟體規劃順序可分為初始化表單檔案、巨集程式檔案、中斷服務常式檔案，機械資料檔案)，就不同檔案類型皆可以簡單管理並容易編寫使用者所自訂的初始化表單檔案、巨集程式檔案、中斷服務常式檔案或機械資料檔案。編寫後可下載至 FRAM 後做初步的執行與驗證。以下 5.2~5.5 節分別就四種檔案內容詳述之。

5.1.1 主功能視窗介紹

視窗主畫面由左至右分別為選單(File, View, Build, Help)，程式檔案樹狀顯示圖，控制按鈕，顯示表單，運控函式庫分類選項，運動函數顯示表單；視窗主畫面下方則為訊息和狀態顯示選單，依功能性可分為 4 個子頁：訊息、4 軸位置速度狀態、4 軸硬體訊號，和除錯模式訊息。



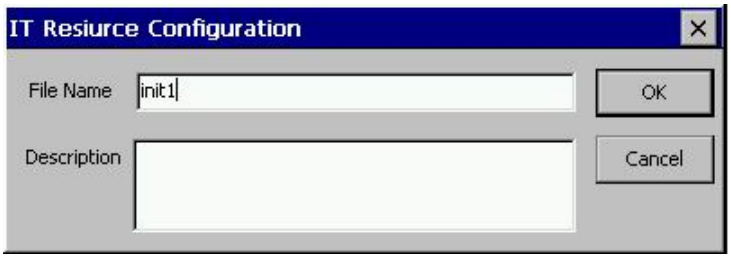
5.2 初始化表單檔案 (IT Files, Initial Table Files)

初始化表單檔案為管理 i-8094H 初始化函數的檔案，使用者可以經由新增初始化表單檔案，然後設定或修改檔案中參數，其後只要對初始化表單檔案做下載(Download)的動作即完成了 i-8094H 模組的初始化設定。初始化表單檔案中的參數設定由 18 個初始化 API 函數所構成，詳細內容請請參閱 5-6 節。

5.2.1 新增初始化表單檔案

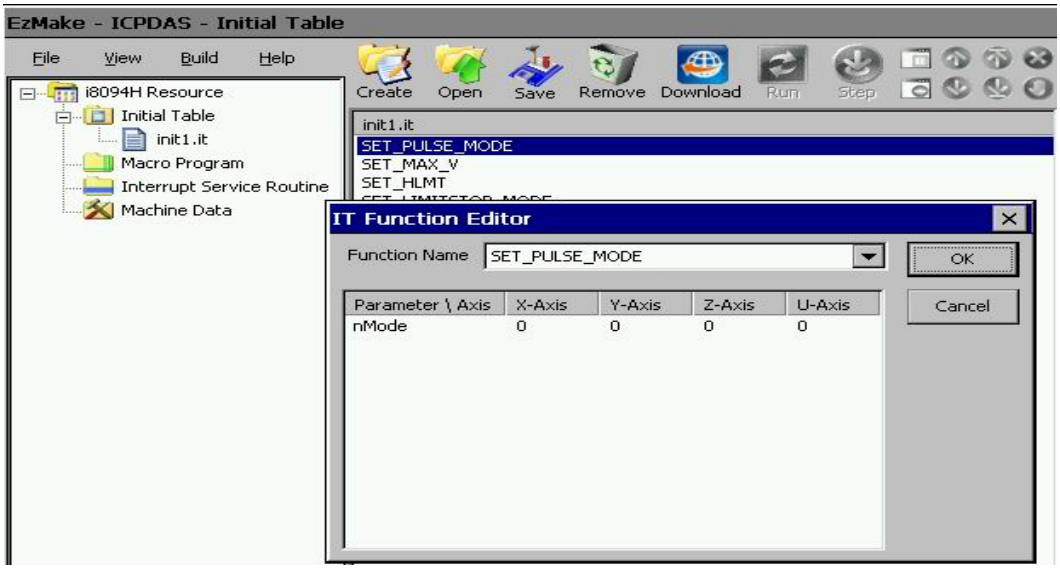
使用者可於主畫面中的"i8094H Resource"樹狀圖中點選"Initial Table"後至 "File → Create" 或直接對主畫面上方的快速控制按鈕列的 Create 按鈕作點選的動作，之後會出現"Initial Table Resource Configuration"的對話視窗，直接輸入檔案名稱和檔案描述，待確認無誤後按下"OK"按鈕，即產生*.it 的檔案於 Initial Table 的樹狀節點中。

備註：在 EzMake 中限制最多同時開啟 5 個初始化表單檔案。



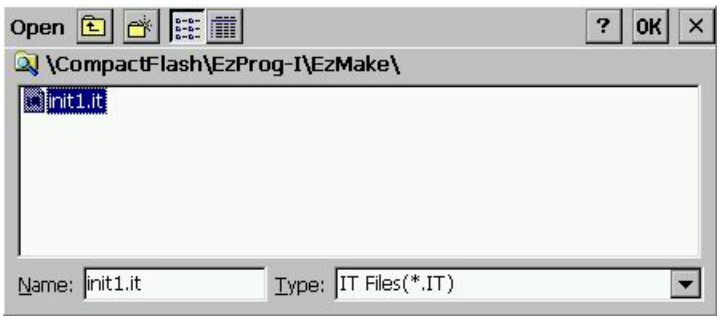
5.2.2 修改初始化表單檔案

使用者可於主畫面中的"i8094H Resource"樹狀圖中點選"*.it"的檔案後至 "File → Create" 或直接對主畫面上方的快速控制按鈕列的 Create 按鈕作點選的動作，之後會出現"IT Resource Configuration"的對話視窗，可修改檔案名稱或檔案描述，待確認無誤後按下"OK"按鈕後，則會同步更新*.it 的檔案於 Initial Table 的樹狀節點中。



5.2.3 開啟初始化表單檔案

使用者可於主畫面中的”i8094H Resource”樹狀圖中點選”*.it”的檔案後至 ”File → Open” 或直接點選快速控制按鈕列的 Open 按鈕，之後會出現檔案開啟的對話視窗，請至以下路徑\EzProg_Path\EzProg-I\EzMake 選擇副檔名為.it 的檔案，待確認無誤後按下”OK”按鈕後，則會加入所選取*.it 的檔案於 Initial Table 的樹狀節點中。

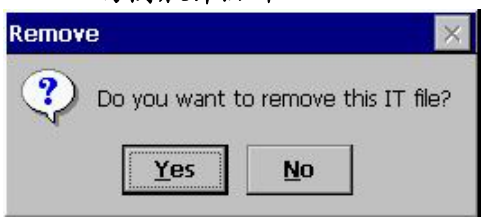


5.2.4 修改初始化表單檔案中的內容

使用者可於主畫面中的”i8094H Resource”樹狀圖中點選新增後的”*.it”檔案後至 ”File → Create” 或直接對主畫面上方的快速控制按鈕列的 Create 按鈕作點選的動作，之後會出現”Initial Table Editor”的對話視窗，可以修改檔案名稱和檔案備註，待確認無誤後按下”OK”按鈕後，*.it 的檔案資料即更新完成。

5.2.5 移除初始化表單檔案

使用者可於主畫面中的”i8094H Resource”樹狀圖中點選”*.it”的檔案後至 ”File → Remove” 或直接點選快速控制按鈕列的 Remove 按鈕，之後會出現詢問是否刪除的對話視窗，待確認無誤後按下”OK”按鈕後，則會刪除*.it 的檔案於 Initial Table 的樹狀節點中。



5.2.6 下載初始化表單檔案

使用者設定完成初始化表單之後可於主畫面中的”i8094H Resource”樹狀圖中點選”*.it”的檔案後至”Build → Download”將檔案下載 i-8094H 模組中，或直接點選快速控制按鈕列的 Download 按鈕將檔案下載 i-8094H 模組中。出現檔案下載成功訊息後即完成初始化表單檔案的下載。

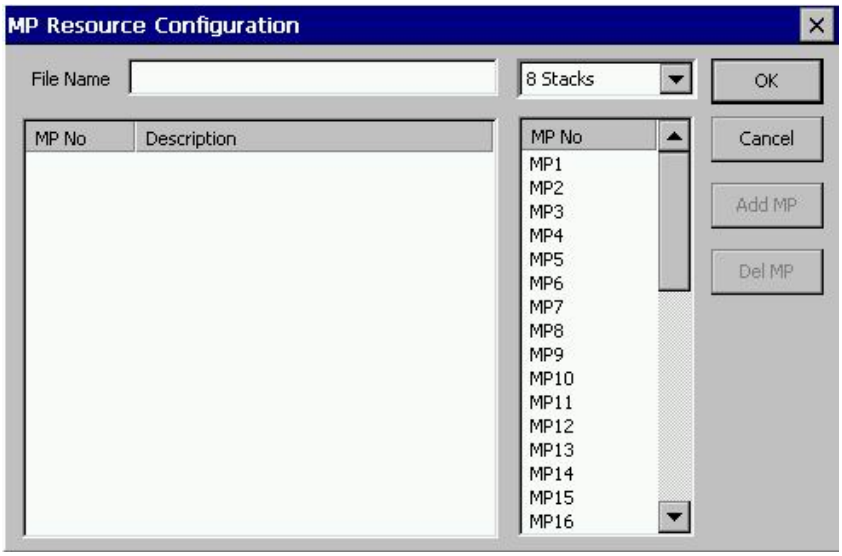
5.3 巨集程式檔案 (MP Files, Macro Program Files)

巨集程式檔案為可以使用 i8094H API 函數互相組合為巨集副程式的程式檔案，使用者可以經由新增巨集程式檔案，選擇檔案中的 MP 副程式號碼，選擇要編輯的 MP 副程式後便可以選擇所需要的 i8094H API 加入你的 MP 副程式中。其後只要對巨集程式檔案做下載(Download)的動作即完成了巨集程式載入於 i-8094H 模組的動作。待成功載入後，可以按下"Run"或"Step"以自動執行或單步執行的方式去驗證所編輯的 MP 副程式。巨集程式檔案總共可以使用 96 個 API 函數，詳細內容請參閱 5-6 節。

5.3.1 新增巨集程式檔案

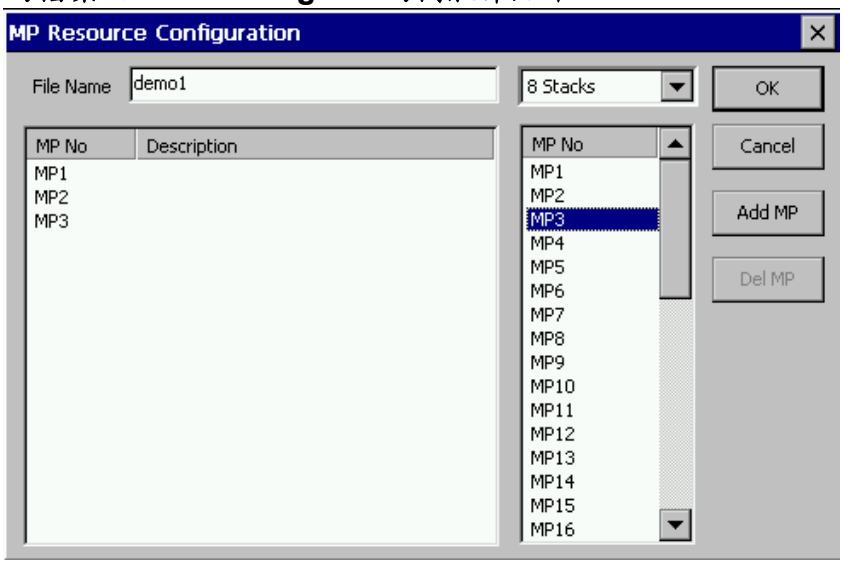
使用者可於主畫面中的"i8094H Resource"中點選"Macro Program"後至 "File → Create" 或直接對主畫面上方的快速控制按鈕列的 Create 按鈕作點選的動作，之後會出現"Macro Program Resource Configuration"的對話視窗，直接輸入檔案名稱，待確認無誤後按下"OK"按鈕後，則會產生*.mp 的檔案於 Macro Program 的樹狀節點中。

備註：在 EzMake 中限制最多同時開啟 5 個巨集程式檔案。



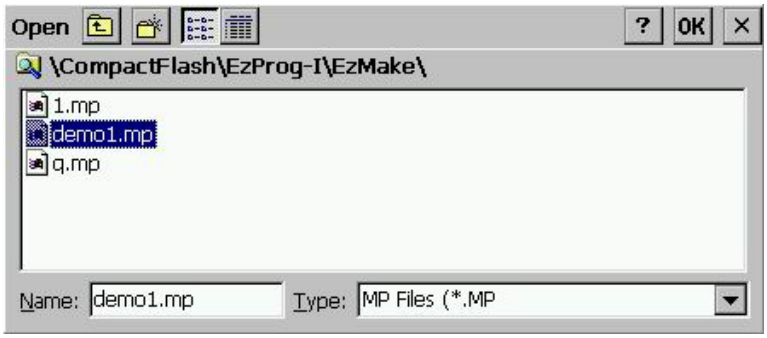
5.2.2 修改巨集程式檔案

使用者可於主畫面中的”i8094H Resource”樹狀圖中點選”*.mp”的檔案後至 ”File → Create” 或直接對主畫面上方的快速控制按鈕列的 Create 按鈕作點選的動作，之後會出現”MP Resource Configuration”的對話視窗，可修改檔案名稱或使用的 MP 副程式編號或檔案描述，待確認無誤後按下”OK”按鈕後，則會同步更新*.mp 的檔案於 Macro Program 的樹狀節點中。



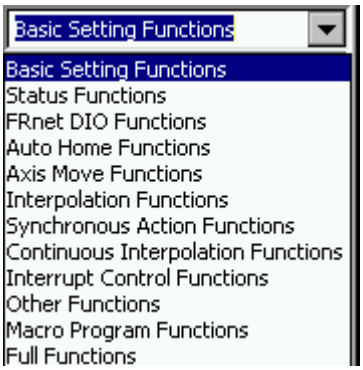
5.2.3 開啟巨集程式檔案

使用者可於主畫面中的”i8094H Resource”樹狀圖中點選”*.mp”的檔案後至 ”File → Open” 或直接點選快速控制按鈕列的 Open 按鈕，之後會出現檔案開啟的對話視窗，請至以下路徑\EzProg_Path\EzProg-I\EzMake 選擇副檔名為.mp 的檔案，待確認無誤後按下”OK”按鈕後，則會加入所選取*.mp 的檔案於 Macro Program 的樹狀節點中。

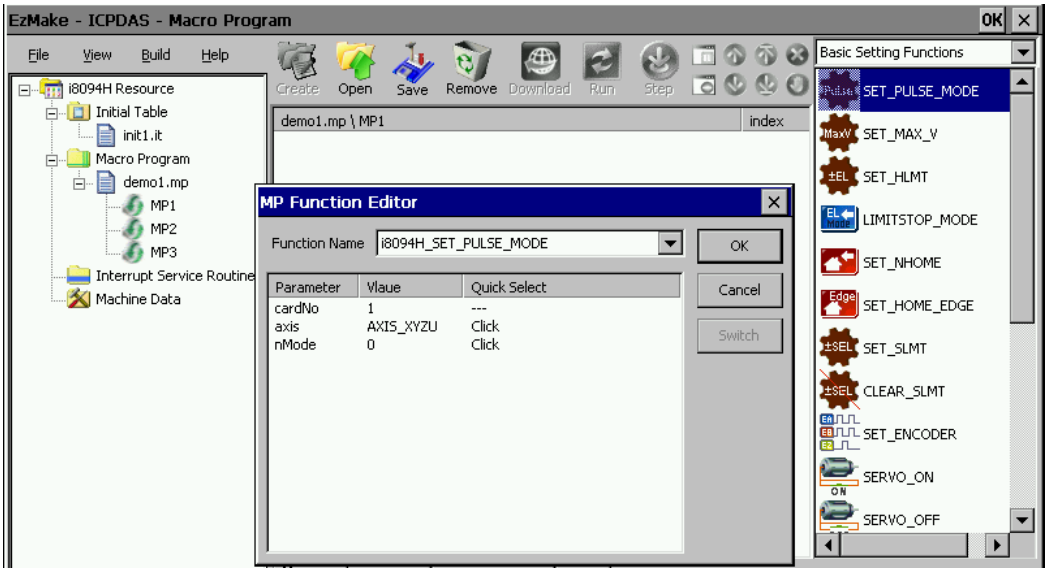


5.2.4 編輯巨集程式檔案中的內容

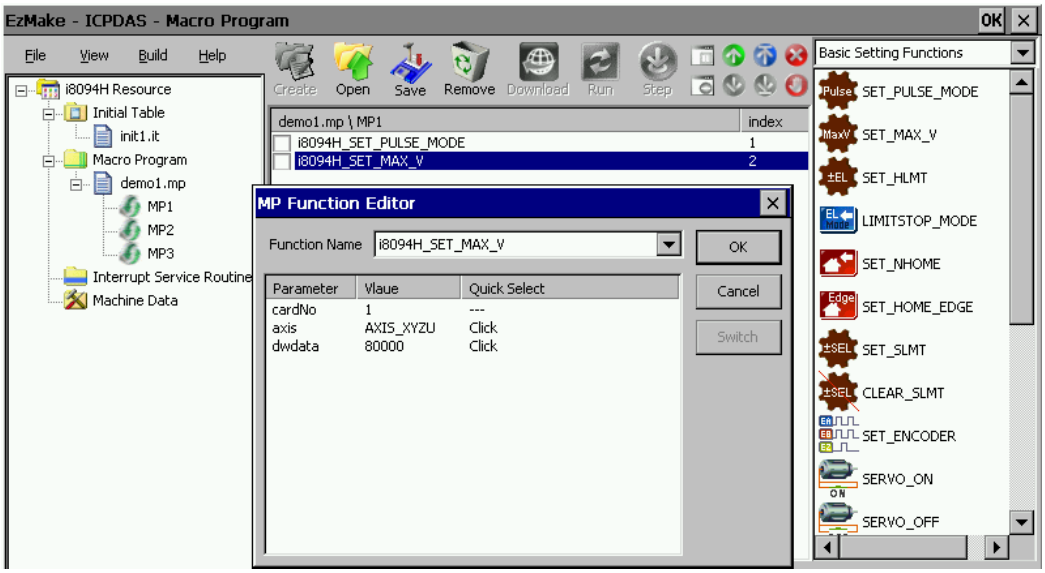
使用者可選擇運動函數的 Combo 控制項切換不同種類的運動函數。另外可於主畫面中的"i8094H Resource"樹狀圖中點選新增後的"*.mp"檔案後至 "File → Create" 或直接對主畫面上方的快速控制按鈕列的 Create 按鈕作點選的動作，之後會出現"Macro Program Editor"的對話視窗，可以修改檔案名稱和使用 MP 編號和檔案備註，待確認無誤後按下"OK"按鈕後，*.mp 的檔案資料即更新完成。



※MP 檔案所支援的函數分爲 11 種：初始化基本設定函數(Basic Setting Functions)，運動狀態讀取/設定函數(Status Functions)，FRnet 串列 DIO 函數(FRnet DIO Functions)，自動原點返回函數(Auto Home Functions)，基本軸控函數(Axis Move Functions)，多軸補間函數(Interpolation Functions)，同步運動函數(Synchronous Action Functions)，多軸連續補間函數(Continuous Interpolation Functions)，中斷控制函數(Interrupt Control Functions)，其他函數(Other Functions)，巨集指令函數(Macro Program Functions)。



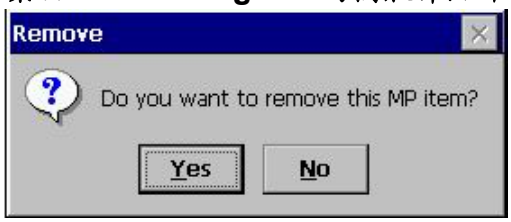
※由加入於編輯區內的函數上雙擊即出現函數編輯對話視窗，可重新修改參數設定，待確認無無後按下"OK"按鈕後即完成更新。



※由加入於編輯區內的函數上雙擊即出現函數編輯對話視窗，可重新修改參數設定，待確認無無後按下"OK"按鈕後即完成更新。

5.2.5 移除巨集程式檔案

關於移除巨集程式檔案方面，使用者可於主畫面中的"i8094H Resource"樹狀圖中點選"*.mp"的檔案後至 "File → Remove" 或直接點選快速控制按鈕列的 Remove 按鈕。只要選擇欲刪除的*.mp 或 MP 副程式後按下"Remove"會之後會出現詢問是否刪除的對話視窗，待確認無誤後按下"OK"按鈕後，則會刪除*.mp 的檔案於 Macro Program 的樹狀節點中。



5.2.6 下載巨集程式檔案

使用者設定完成巨集程式檔案之後可於主畫面中的”i8094H Resource”樹狀圖中點選”*.mp”的檔案後至”Build → Download”將檔案下載 i-8094H 模組中，或直接點選快速控制按鈕列的 Download 按鈕將檔案下載 i-8094H 模組中。出現檔案下載成功訊息後即完成巨集程式檔案的下載。

5.2.7 執行巨集程式檔案

使用者下載完成巨集程式檔案之後，可選擇自動執行(“Run”)或單步執行(“Step”)來驗證巨集程式檔案的運動控制程式和基本邏輯設定。操作方式可於主畫面中的”i8094H Resource”樹狀圖中點選使用者建立*.mp 檔案中的 MP 號碼副程式後至”Build →Run/Step ”，或直接點選快速控制按鈕列的 Run 或 Step 按鈕，出現詢問是否要執行的對話視窗，待確認無誤後按下”OK”按鈕後，程式開始進入 Debug 模式中。

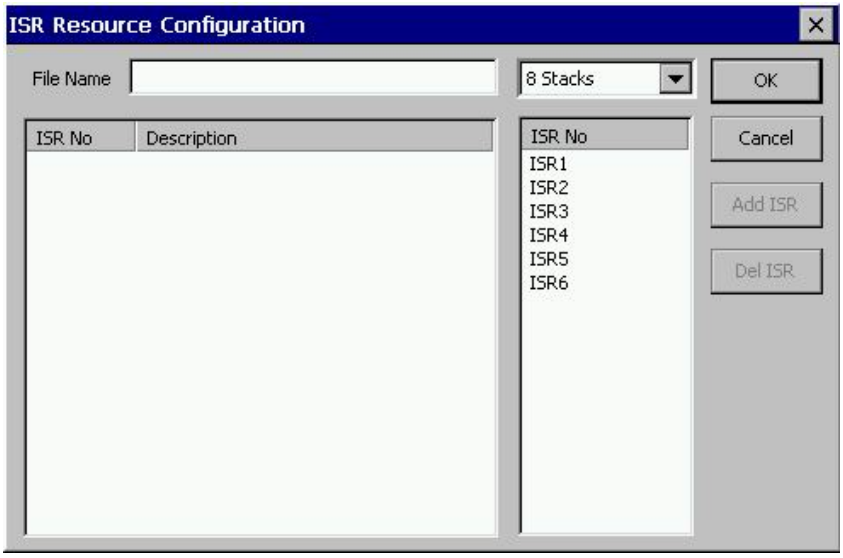
注:目前暫時取消單步執行部分

5.4 中斷服務常式檔案 (ISR Files, Interrupt Service Routine Files)

中斷服務常式檔案為管理 i8094H API 函數中可以使用於中斷服務常式中的檔案，總共可以使用 50 個 API 函數，詳細請參閱 5-6 節。

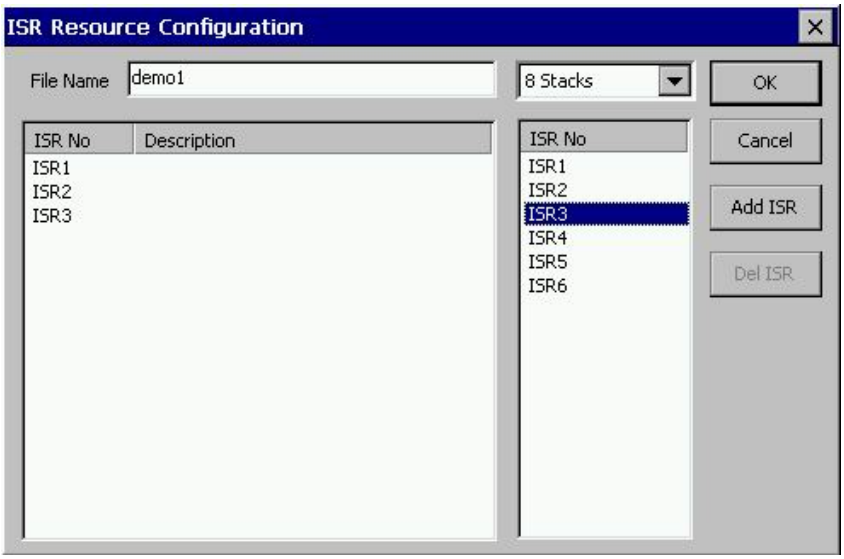
5.2.1 新增中斷服務常式檔案

使用者可於主畫面中的"i8094H Resource"中點選"Interrupt Service Routine"後至 "File → Create" 或直接對主畫面上方的快速控制按鈕列的 Create 按鈕作點選的動作，之後會出現"Interrupt Service Resource Configuration"的對話視窗，直接輸入檔案名稱，待確認無誤後按下"OK"按鈕後，則會產生*.isr 的檔案於 Interrupt Service Routine 的樹狀節點中。



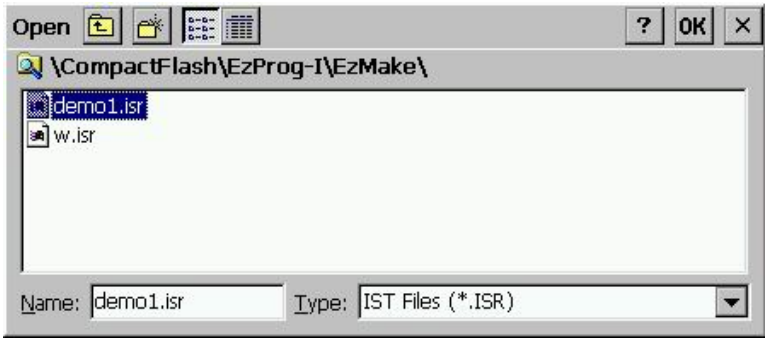
5.2.2 修改中斷服務常式檔案

使用者可於主畫面中的"i8094H Resource"樹狀圖中點選"*.isr"的檔案後至 "File → Create" 或直接對主畫面上方的快速控制按鈕列的 Create 按鈕作點選的動作，之後會出現"ISR Resource Configuration"的對話視窗，可修改檔案名稱或檔案描述，待確認無誤後按下"OK"按鈕後，則會同步更新*.isr 的檔案於 Interrupt Service Routine 的樹狀節點中。



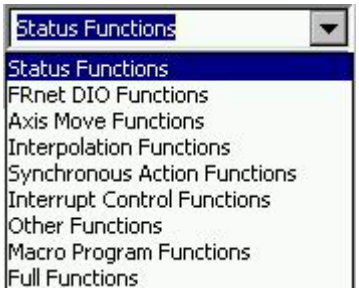
5.2.3 開啟中斷服務常式檔案

使用者可於主畫面中的"i8094H Resource"樹狀圖中點選"*.isr"的檔案後至 "File → Open" 或直接點選快速控制按鈕列的 Open 按鈕，之後會出現檔案開啟的對話視窗，請至以下路徑\EzProg_Path\EzProg-I\EzMake 選擇副檔名為*.isr 的檔案，待確認無誤後按下"OK"按鈕後，則加入所選取*.isr 的檔案於 Interrupt Service Routine 的 i8094H Resource 樹狀節點中。

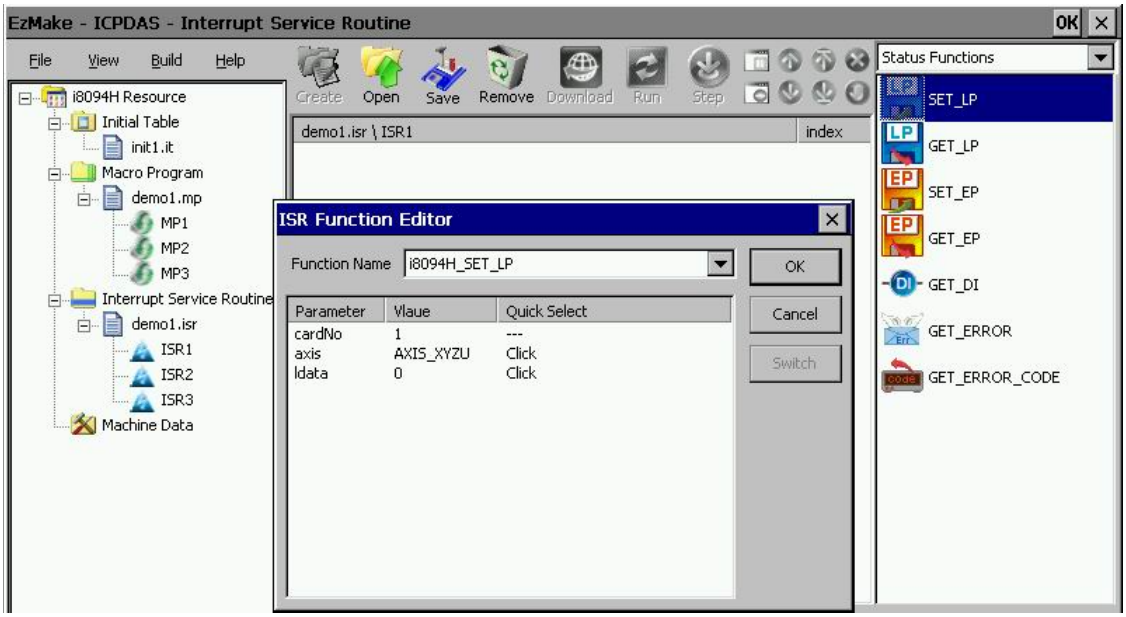


5.2.4 編輯中斷服務常式檔案中的內容

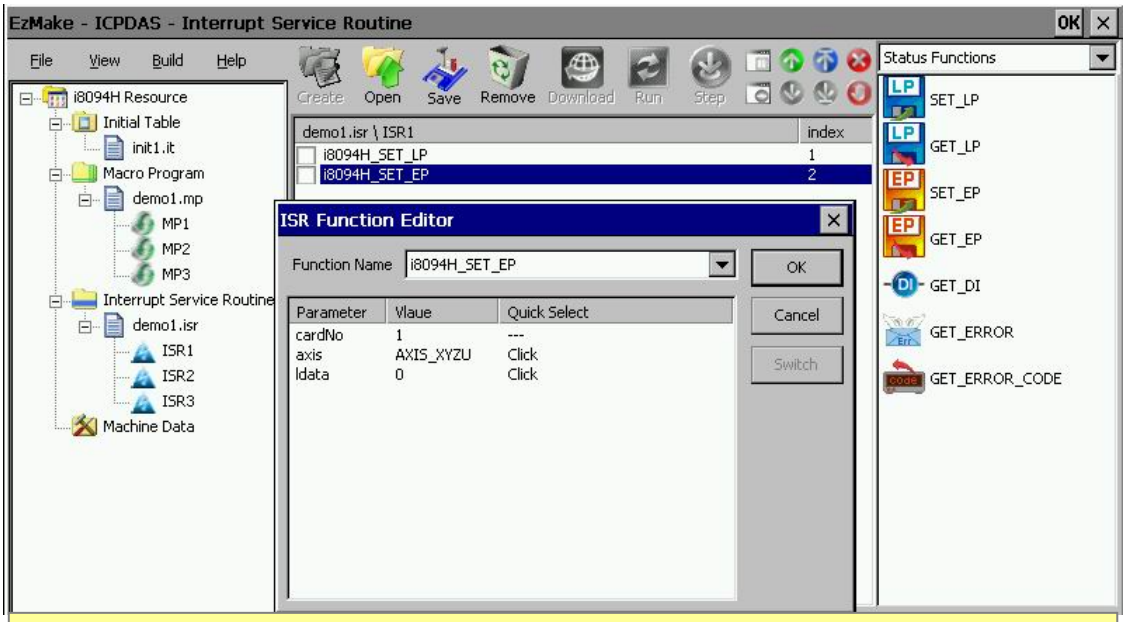
使用者可於主畫面中的"i8094H Resource"樹狀圖中點選新增後的"*.isr"檔案後至 "File → Create" 或直接對主畫面上方的快速控制按鈕列的 Create 按鈕作點選的動作，之後會出現"Interrupt Service Routine Editor"的對話視窗，可以修改檔案名稱和使用 ISR 編號和檔案備註，待確認無誤後按下"OK"按鈕後，*.isr 的檔案資料即更新完成。



※ISR 檔案所支援的函數分爲 8 種：運動狀態讀取/設定函數(Status Functions)，FRnet 串列 DIO 函數(FRnet DIO Functions)，基本軸控函數(Axis Move Functions)，多軸補間函數(Interpolation Functions)，同步運動函數(Synchronous Action Functions) ，中斷控制函數(Interrupt Control Functions)，其他函數(Other Functions)，巨集指令函數(Macro Program Functions)。



※由加入於編輯區內的函數上雙擊即出現函數編輯對話視窗，可重新修改參數設定，待確認無無後按下"OK"按鈕後即完成更新。



※由加入於編輯區內的函數上雙擊即出現函數編輯對話視窗，可重新修改參數設定，待確認無無後按下"OK"按鈕後即完成更新。

關於移除中斷服務常式檔案方面，使用者可於主畫面中的"i8094H Resource"樹狀圖中點選"*.isr"的檔案後至 "File → Remove" 或直接點選快速控制按鈕列的 Remove 按鈕。只要選擇欲刪除的*.isr 或 ISR 副程式後按下"Remove"會出現詢問是否刪除的對話視窗，待確認無誤後按下"OK"按鈕後，則會刪除*.isr 的檔案於 Interrupt Service Routine 的樹狀節點中。

5.2.6 載入中斷服務常式檔案

使用者設定完成初始化表單之後可於主畫面中的"i8094H Resource"樹狀圖中點選"*.isr"的檔案後至"Build → Download"將檔案下載 i-8094H 模組中，或直接點選快速控制按鈕列的 Download 按鈕將檔案下載 i-8094H 模組中。出現檔案下載成功訊息後即完成中斷服務常式檔案的下載。

5.2.7 執行中斷服務常式檔案

使用者下載完成中斷服務常式檔案之後，可選擇執行("Run")來驗證中斷服務常式檔案的運動控制程式和基本邏輯設定。操作方式可於主畫面中的"i8094H Resource"樹狀圖中點選使用者建立*.isr 檔案中的 ISR 號碼副程式後至"Build →Run"，或直接點選快速控制按鈕列的 Run 按鈕，出現詢問是否要執行的對話視窗，待確認無誤後按下"OK"按鈕後，程式開始進入 Debug 模式中。

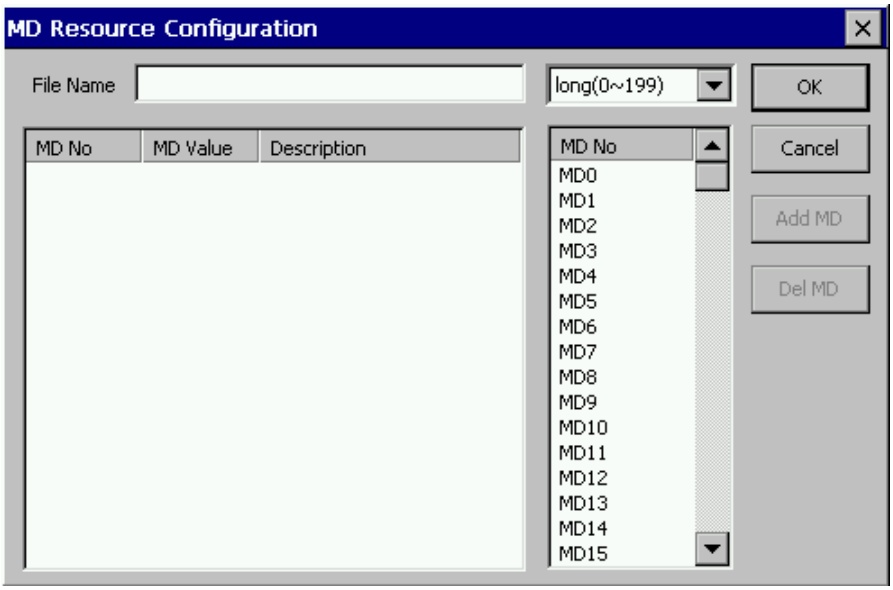
5.5 機械資料檔案 (MD Files, Machine Data Files)

機械資料檔案為管理機械平台、機械設備，或其他受控體的一些相關機械參數或運動控制相關常數，每一個機械資料檔案總共可以定義 2048 筆資料 (MD0~MD2047)，每筆資料可包含 MD 號碼、資料值、備註等。

5.5.1 新增機械資料檔案

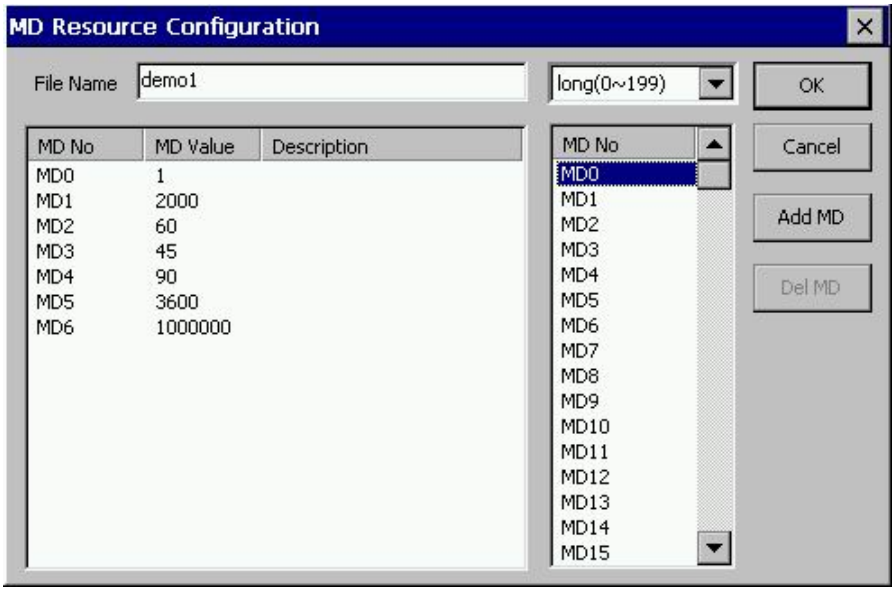
使用者可於主畫面中的"i8094H Resource"中點選"Machine Data"後至 "File → Create" 或直接對主畫面上方的快速控制按鈕列的 Create 按鈕作點選的動作，之後會出現"Machine Data Resource Configuration"的對話視窗，直接輸入檔案名稱，待確認無誤後按下"OK"按鈕後，則會產生*md 的檔案於 Machine Data 的樹狀節點中。

備註：在 EzMake 中限制最多同時開啟 5 個初始化表單檔案。



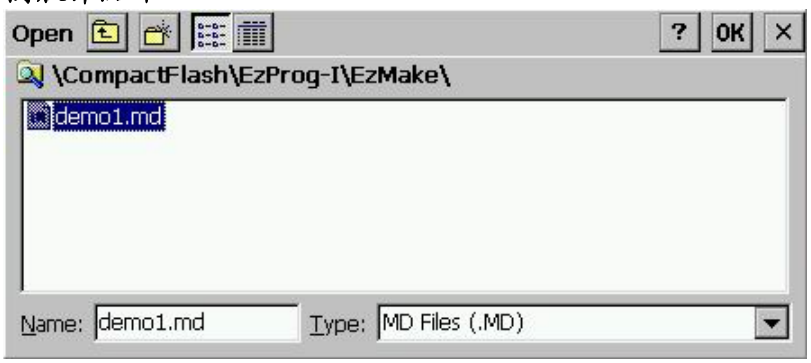
5.5.2 修改機械資料檔案

使用者可於主畫面中的”i8094H Resource”樹狀圖中點選”*.md”的檔案後至 ”File → Create” 或直接對主畫面上方的快速控制按鈕列的 Create 按鈕作點選的動作，之後會出現”MD Resource Configuration”的對話視窗，可修改檔案名稱或使用的 MD 編號或檔案描述，待確認無誤後按下”OK”按鈕後，則會同步更新*.md 的檔案於 Machine Data 的樹狀節點中。



5.5.3 開啟機械資料檔案

使用者可於主畫面中的”i8094H Resource”樹狀圖中點選”*.md”的檔案後至 ”File → Open” 或直接點選快速控制按鈕列的 Open 按鈕，之後會出現檔案開啟的對話視窗，請至以下路徑\EzProg_Path\EzProg-I\EzMake 選擇副檔名為.md 的檔案，待確認無誤後按下”OK”按鈕後，則會加入所選取*.md 的檔案於 Machine Data 的樹狀節點中。



5.5.3 編輯機械資料檔案中的內容

使用者可於主畫面中的”i8094H Resource”樹狀圖中點選新增後的”*.md”檔案後至 ”File → Create” 或直接對主畫面上方的快速控制按鈕列的 Create 按鈕作點選的動作，之後會出現”Machine Data Editor”的對話視窗，可以修改檔案名稱、使用 MD 編號、MD 資料數值和檔案備註，待確認無誤後按下”OK”按鈕後，*.md 的檔案資料即更新完成。

5.5.4 移除機械資料檔案

使用者可於主畫面中的”i8094H Resource”樹狀圖中點選”*.md”的檔案後至 ”File → Remove” 或直接點選快速控制按鈕列的 Remove 按鈕，之後會出現詢問是否刪除的對話視窗，待確認無誤後按下”OK”按鈕後，則會刪除*.md 的檔案於 Machine Data 的樹狀節點中。

5.5.5 載入機械資料檔案

使用者設定完成初始化表單之後可於主畫面中的”i8094H Resource”樹狀圖中點選”*.md”的檔案後至”Build → Download”將檔案下載 i-8094H 模組中，或直接點選快速控制按鈕列的 Download 按鈕將檔案下載 i-8094H 模組中。出現檔案下載成功訊息後即完成機械資料檔案的下載。

5.6 專案管理(Prj Files,Project Files)

5.2~5.5 是開啟或創建單一類型檔案的方式,如果使用者一次需要處理多個類型檔案的話,可以選擇以專案方式開啟,或者一開始就以專案方式創建專案的各類型檔案。

5.6.1 開啟專案

專案開啟方式和開啟其他單一類型檔案一樣,只要點選 Project 後在控制按鈕或主選單選擇『OPEN』即可開啟畫面,再選擇要開啟『.prj』檔案的位置就完成開啟,『.prj』檔案開啟後會依序放在 Project 內。(圖 5.6.1)

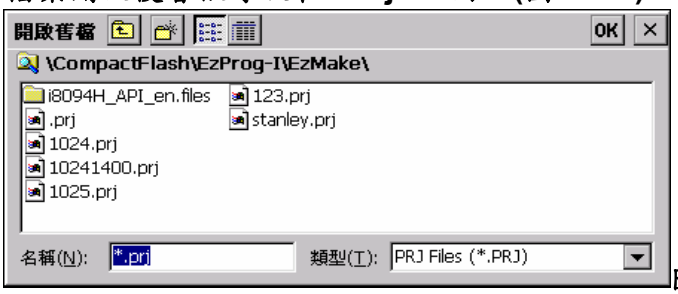


圖 5.6.1 專案開啟

5.6.2 建立新專案

當選擇 Project 然後在 Menu 或控制按鈕按下『Create』後可以開啟 Project 畫面(圖 5.6.2)。首先在 Project Name 輸入名稱，接著可以在四個類型名稱前勾選要不要選擇該類型,再來決定要在新專案內創該類型的新檔案或是開該類型的舊檔案,都勾選完後按下 OK,如果是選擇開舊檔案,該檔案或直接開啟在 Projet 的該類型資料夾下,如果是新建,則步驟就如 5.2~5.5 的步驟一樣。

另外一種開啟方式需要說明一下，在使用者已經開啟一個專案的情況下，而該專案當初創建時並未將所有的類型加入,這時候如果又想將未加入之類型加入,雖然一樣是用 Create 的方式,但是,已經開啟的類型還是必須要勾選,而且後面的方式要點選 OPEN。

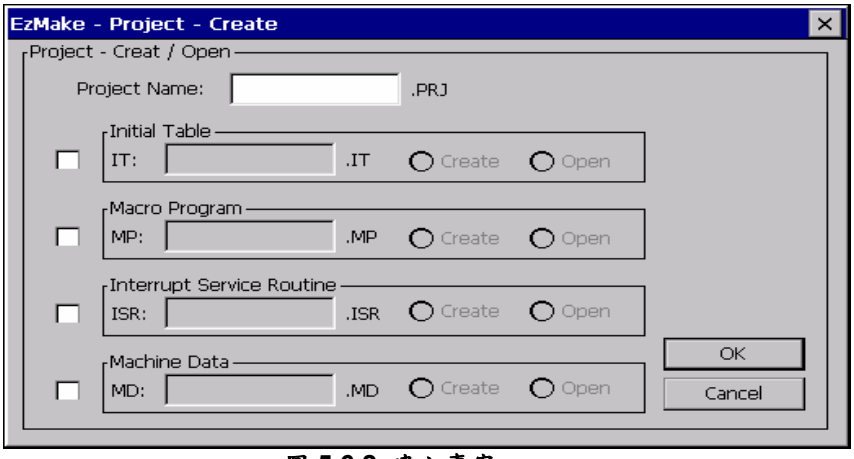


圖 5.6.2 建立專案

5.6.3 移除專案

專案的移除和其他類型檔案的移除一樣,只要點選『xxx.prj』然後再 Menu 或控制按鈕按下 Remove 即可將專案移除。

5.6.4 修改專案

點選專案名稱然後再 Menu 或控制按鈕選擇 Create 可以對該專案進行修改(圖 5.6.4)。修改內容為決定四類檔案的包含與否以及是否要重新創新類型檔案或開啟其他舊檔案。選擇開其他舊檔案的話是點選 Open 的選項,按下 OK 後該類型檔案自動開啟在 Project 的該類型資料夾下。選擇建立新檔案的話是點選 Create 的選項,按下 OK 後見新檔案過程和 5.2~5.5 一樣。

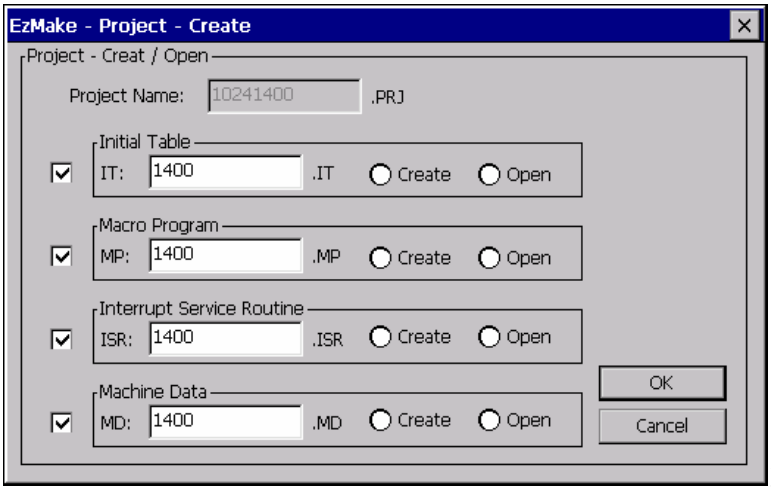


圖 5.6.4 修改專案內容

專案開啟或者建立之後，為了避免開啟過多的同類型檔案，即不能對各類型資料夾做新增或開啟該類型檔案的動作，要改變所包括的檔案只能以修改專案方式來做。

5.6.5 儲存專案

專案建立設定好各部份後要儲存的話，如果要儲存所有類型的檔案則先點選專案名稱 XXXX.prj 然後在 Menu 或控制按鈕按下 SAVE 即可儲存所有的檔案，另外，使用者也可以對單一類型檔案進行儲存,方法如各類型檔案內所述。

5.7 運動控制函數

EzMake 提供的運動控制函數需搭配 i-8094H 模組來使用，依功能性區分可分為 11 類：一、基本設定函數；二、運動狀態讀取/設定函數；三、串列式 DIO 函數；四、原點返回函數；五、基本軸控函數；六、多軸補間函數；七、同步運動函數；八、多軸連續補間函數；九、中斷控制函數；十、其他功能函數；十一、巨集指令函數。以上各函數分述於 5.6.1~5.6.11 小節。

備註：IT 檔案支援 18 個運動控制函數，MP 支援 96 個運動控制函數，ISR 支援 50 個運動控制函數。

5.7.1 初始化基本設定函數

圖示	函數名稱	函數說明	IT	MP	ISR
	i8094H_SET_PULSE_MODE	包含，設定軸之輸出模式CW/CCW 或 PULSE/DIR。及正方向定義，	◎	◎	
	i8094H_SET_MAX_V	設定軸之輸出最高速度PPS 影響，限制: 。反之越大，速度解析度越高，最高速度越小	◎	◎	
	i8094H_SET_HLMT	設定軸之"前後極限"。開關觸發邏輯	◎	◎	
	i8094H_LIMITSTOP_MODE	設定碰觸"前後極限"。處理模式	◎	◎	
	i8094H_SET_NHOME	設定軸之"近原點"。開關觸發邏輯	◎	◎	
	i8094H_SET_HOME_EDGE	設定軸之"原點"。開關觸發邏輯	◎	◎	
	i8094H_SET_SLMT	設定軸之"前後軟體極限"。功能	◎	◎	
	i8094H_CLEAR_SLMT	取消軸之"前後軟體極限"。功能	◎	◎	
	i8094H_SET_ENCODER	。設定軸之編碼器輸入參數	◎	◎	
	i8094H_SERVO_ON	。設定軸驅動器伺服啟動	◎	◎	
	i8094H_SERVO_OFF	。設定軸驅動器伺服關閉	◎	◎	
	i8094H_SET_ALARM	之驅動器異常設定軸(ALARM)。輸入參數	◎	◎	
	i8094H_SET_INPOS	。動器定位完成輸入參數設定軸之驅	◎	◎	
	i8094H_SET_FILTER	。設定軸之輸入數位濾波項目及濾波時間參數	◎	◎	
	i8094H_VRING_ENABLE	指定軸啟動為環狀計數器。	◎	◎	
	i8094H_VRING_DISABLE	。指定軸關閉環狀計數器功能	◎	◎	
	i8094H_AVTRI_ENABLE	設定軸開啟預防三角形速度曲線的功能。	◎	◎	
	i8094H_AVTRI_DISABLE	設定軸關閉預防三角形速度的功能。	◎	◎	

5.7.2 運動狀態讀取/設定函數

圖示	函數名稱	函數說明	IT	MP	ISR
	i8094H_SET_LP	。設定軸之目前指令邏輯位置		◎	◎
	i8094H_GET_LP	。讀取軸目前之指令邏輯位置		◎	◎
	i8094H_SET_EP	設定軸之目前ENCODER。回授位置		◎	◎
	i8094H_GET_EP	讀取軸目前之ENCODER。回授位置		◎	◎
	i8094H_GET_DI	。輸入點狀態讀取軸之		◎	◎
	i8094H_GET_ERROR	。讀取軸運動有無錯誤發生		◎	◎
	i8094H_GET_ERROR_CODE	。讀取各軸之錯誤碼		◎	◎

5.7.3 FRnet 串列式 DIO 函數

圖示	函數名稱	函數說明	IT	MP	ISR
	i8094H_FRNET_IN	讀取FRnet的數位輸入資料。		◎	◎
	i8094H_FRNET_OUT	寫入FRnet的數位輸出資料。		◎	◎

5.7.4 原點返回函數

圖示	函數名稱	函數說明	IT	MP	ISR
	i8094H_SET_HV	設定軸之原點返回。速度		◎	
	i8094H_HOME _LIMIT	設定軸之Limit 。開關當原點開關		◎	
	i8094H_SET_HOME_MODE	設定軸原點返回。方法及參數		◎	
	i8094H_HOME _START	設定軸開始執行原點返回。		◎	

5.7.5 基本軸控函數

圖示	函數名稱	函數說明	IT	MP	ISR
	i8094H_NORMAL_SPEED	設定速度模式。		◎	◎
	i8094H_SET_SV	設定軸之初始速度。		◎	◎
	i8094H_SET_V	設定軸之定速度。		◎	◎
	i8094H_SET_A	設定軸之加速度。		◎	◎

	i8094H_SET_D	設定軸之減速度。		◎	◎
	i8094H_SET_K	設定軸之輸出加速度變化率。		◎	◎
	i8094H_SET_L	設定軸之輸出減速度變化率。		◎	◎
	i8094H_SET_AO	於固定脈波數運動控制時，至目標前保留低速輸出 Offset Pulse 數。		◎	◎
	i8094H_FIXED_MOVE	執行設定軸固定步數輸出。		◎	◎
	i8094H_SET_PULSE	在單軸固定步數輸出時，可於途中改變輸出步數，但無法改變方向。		◎	◎
	i8094H_CONTINUE_MOVE	執行設定軸連續脈波輸出。		◎	◎

5.7.6 多軸補間函數

圖示	函數名稱	函數說明	IT	MP	ISR
	i8094H_AXIS_ASSIGN	設定補間軸。		◎	◎
	i8094H_VECTOR_SPEED	設定向量加減速模式。		◎	◎
	i8094H_SET_VSV	設定軸之向量初始速度。		◎	◎
	i8094H_SET_VV	設定軸之向量定速度。		◎	◎
	i8094H_SET_VA	設定軸之向量加速度。		◎	◎
	i8094H_SET_VD	設定軸之向量減速度。		◎	◎
	i8094H_SET_VK	設定軸之輸出向量加速度變化率。		◎	◎
	i8094H_SET_VL	設定軸之輸出向量減速度變化率。		◎	◎
	i8094H_SET_VAO	於固定脈波數運動控制時，至目標前保留低速輸出 Offset Pulse 數。		◎	◎
	i8094H_LINE_2D	執行二軸直線補間。		◎	◎
	i8094H_LINE_3D	執行三軸直線補間。		◎	◎
	i8094H_ARC_CW	執行二軸順時針圓弧補間。		◎	◎
	i8094H_ARC_CCW	執行二軸逆時針圓弧補間。		◎	◎
	i8094H_CIRCLE_CW	執行二軸順時針圓形補間。		◎	◎
	i8094H_CIRCLE_CCW	執行二軸逆時針圓形補間。		◎	◎

5.7.7 同步運動函數

圖示	函數名稱	函數說明	IT	MP	ISR
	i8094H_SYNC_ACTION	同步運動條件的設定。		◎	◎
	i8094H_SET_COMPARE	設定位置比較器的值，將會使軟體極限功能失效。		◎	◎
	i8094H_GET_LATCH	讀取同步位置閂鎖值。		◎	◎
	i8094H_SET_PRESET	選擇同步資料設定方式(各同步運動軸無法單獨設定)。		◎	◎
	i8094H_SET_OUT	輸出脈波設定。		◎	◎

5.7.8 多軸連續補間函數

圖示	名稱	說明	IT	MP	ISR
	i8094H_RECTANGLE	執行二軸矩形補間。		◎	
	i8094H_LINE_2D_INITIAL	二軸直線連續補間初始設定(對稱 T 曲線加減速)。		◎	
	i8094H_LINE_2D_CONTINUE	執行二軸直線連續補間。		◎	
	i8094H_LINE_3D_INITIAL	三軸直線連續補間初始設定(對稱 T 曲線加減速)。		◎	
	i8094H_LINE_3D_CONTINUE	執行三軸直線連續補間。		◎	
	i8094H_MIX_2D_INITIAL	二軸直線和圓弧連續補間初始設定。		◎	
	i8094H_MIX_2D_CONTINUE	執行二軸直線和圓弧連續補間。		◎	
	i8094H_HELIX_3D	執行螺旋運動(定速)。		◎	
	i8094H_RATIO_INITIAL	比例運動初始設定(對稱 T 曲線加減速)。		◎	
	i8094H_RATIO_2D	執行比例連續運動。		◎	

5.7.9 中斷控制函數

圖示	名稱	說明	IT	MP	ISR
	i8094H_ENABLE_INT	開啟中斷功能。		◎	
	i8094H_DISABLE_INT	關閉中斷功能。		◎	
	i8094H_INTFACTOR_ENABLE	設定觸發中斷條件因子。		◎	
	i8094H_INTFACTOR_DISABLE	解除觸發中斷條件因子。		◎	

5.7.10 其他功能函數

圖示	名稱	說明	IT	MP	ISR
	i8094H_DRV_START	指定軸開始動作。		◎	◎
	i8094H_DRV_HOLD	指定軸運動暫停。		◎	◎
	i8094H_STOP_SUDDENLY	指定軸之輸出立即(緊急)停止。		◎	
	i8094H_STOP_SLOWLY	指定軸之輸出減速停止。		◎	
	i8094H_VSTOP_SUDDENLY	指定補間軸之輸出立即(緊急)停止。		◎	
	i8094H_VSTOP_SLOWLY	指定補間軸之輸出減速停止。		◎	
	i8094H_CLEAR_STOP	故障排除後，清除停止狀態。		◎	
	i8094H_INTP_END	改做單軸運動或改變補間運動座標系，請於動作前下此指令。		◎	

5.7.11 巨集指令函數

圖示	名稱	說明	IT	MP	ISR
	i8094H_MP_CALL	i8094H 將執行指定編號的巨集程式。		◎	
	i8094H_MP_SET_VAR	設定變數值 VARn。		◎	
	i8094H_MP_SET_RVAR	設定變數值 VARn 為函式的返回值。		◎	
	i8094H_MP_VAR_CALCULATE	變數四則運算，*varNo* (+-*/) *varNo1* = *varNo2*。		◎	
	i8094H_MP_FOR	程序執行循環命令。		◎	
	i8094H_MP_NEXT	需搭配 i8094H_MP_FOR 使用。		◎	
	i8094H_MP_IF	程序條件敘述式。		◎	
	i8094H_MP_ELSE	需搭配 i8094H_MP_IF 使用。		◎	
	i8094H_MP_IF_END	結束 i8094H_MP_IF 程序。		◎	
	i8094H_MP_TIMER	程序延時命令。		◎	
	i8094H_MP_STOP_WAIT	等待軸完全停止，MP 的程序才能再繼續執行。		◎	
	i8094H_MP_SET_RINT	使用者定義 RINT 的設定，當 MP 執行到此 Function，軸卡便會對 Controller 發出 0x04 的中斷。		◎	

5.8 全域參數資料表 (bVAR/VAR Table)

全域參數資料表為針對使用者常用或特殊的參數將其存在模組中的 FRAM 中，為斷電保持的資料，重新開機後資料會記錄最後所輸入或改變的參數。關於全域參數資料表的使用，可於 EzMake 主畫面中的選單””Build→bVAR/VAR Table”開啟此視窗來編輯。全域參數資料表依資料型態分為兩個子頁，第一個子頁資料型態為 BYTE，設定 bVAR0~bVAR127 共 128 筆資料。第二個子頁資料型態為 long，設定 VAR0~bVAR511 共 512 筆資料。設定或修改資料後，如果要在 MP/ISR 副程式使用需將 Control Flag 設定為 used，待確認無誤後按下 OK 即完成設定。