

# 開發 eLogger HMI 加上 ISaGRAF SoftLogic 一起應用於 WP-8xx7, VP-2xW7 與 XP-8xx7-CE6 等 PAC 內

本文件版本為 1.01 版發布於 2009 年 11 月 25 日。 by [chun@icpdas.com](mailto:chun@icpdas.com)

eLogger 是泓格科技 (ICP DAS) 開發的 HMI 軟體工具, 具備簡單好用的特性, 可以放在 WP-8xx7 與 VP-2xW7 與日後即將出產的 XP-8xx7-CE6 內與 ISaGRAF SoftLogic 程序一起運行。

WP-8147/8447/8847 與 WP-8137/8437/8837 與 VP-25W7/23W7 從以下版本起開始支持 eLogger。  
WP-8xx7: 驅動版本第 1.16 版起。 VP-25W7/23W7: 驅動版本第 1.07 版起。

可由此下載 PAC 最新的驅動程式: <http://www.icpdas.com/products/PAC/i-8000/isagraf-link.htm>

本文件會隨 eLogger 軟體功能更新後 而 適時更新為新版本, 用戶可訪問以下網址取得最新的文件與範例程式. [www.icpdas.com](http://www.icpdas.com) > FAQ > Software > ISaGRAF > 中文 > FAQ-115 .

● 離線模擬程式  
● 線上除錯/控制/監看  
● 六種開放式軟邏輯程式語言(LD, ST, ..)  
**ISaGRAF**  
應用程式開發軟體

● 免費 HMI 應用工具  
● HMI 易學 & 易用  
**eLogger**  
HMI 人機介面開發工具

**ICP DAS**

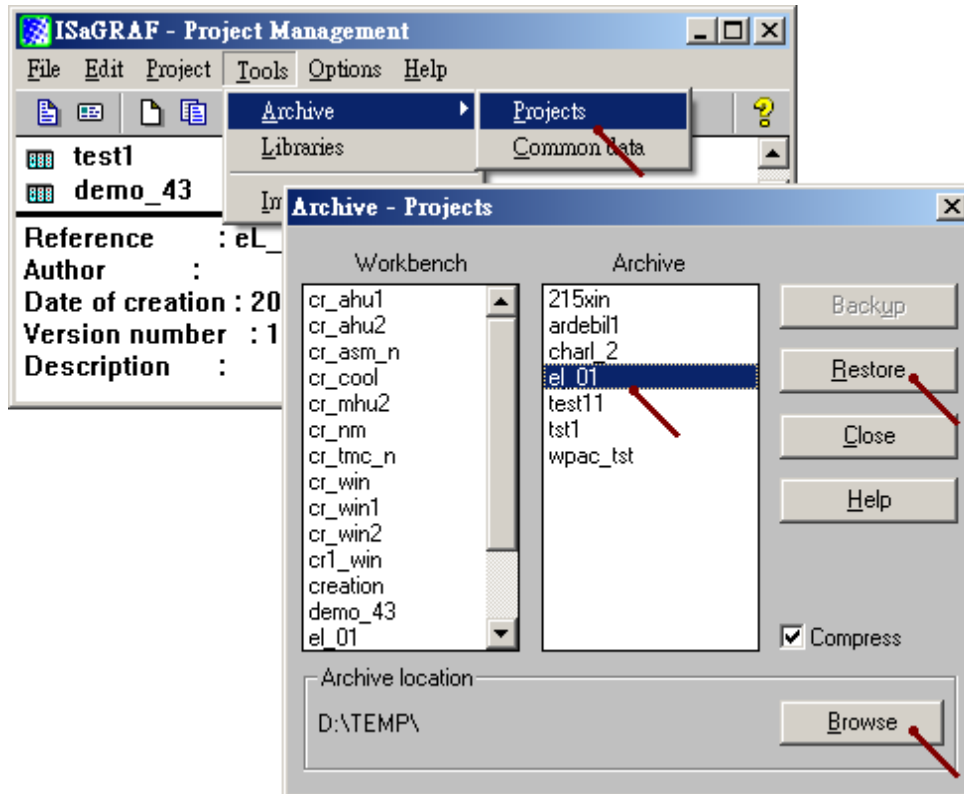
WP-8x47 / 8x37

VP-25W7 / 23W7

其他參考文件: 可由 [http://www.icpdas.com/products/PAC/i-8000/getting\\_started\\_manual.htm](http://www.icpdas.com/products/PAC/i-8000/getting_started_manual.htm) 取得  
ISaGRAF 進階使用手冊  
WinPAC ISaGRAF PAC 快速上手手冊  
VP-2xW7 快速上手手冊

## 1.1: 編寫 ISaGRAF SoftLogic 程式

本範例程式為 eL\_01.pia, 可從 [www.icpdas.com](http://www.icpdas.com) > FAQ > Software > ISaGRAF > FAQ-115 下載。若用戶已經熟悉 ISaGRAF 的使用方法, 可以直接將此 eL\_01.pia 回存到 PC / ISaGRAF 內, 之後將此 ISaGRAF 程式” eL\_01” 下載到 控制器內, 然後直接參考本文件 1.2 節來編寫 eLogger project.



若不清楚 ISaGRAF 要如何使用, 就必須操作本文件第 1.1 節的內容。

(不熟悉 ISaGRAF 的用戶, 建議先閱讀 “ISaGRAF 進階使用手冊 “ 第 2 章, 手冊為 PDF 格式, 名稱為 “chinese\_user\_manual\_i\_8xx7.pdf” 與 “chinese\_user\_manual\_i\_8xx7\_appendix.pdf” 燒錄在 WP-8xx7 與 VP-2xW7 的出貨 CD-ROM 內

WP-8xx7 CD-ROM: \Napdos\isagraf\wp-8xx7\chinese\_manu\

VP-2xW7 CD-ROM: \Napdos\isagraf\vp-25w7-23w7\chinese\_manu\

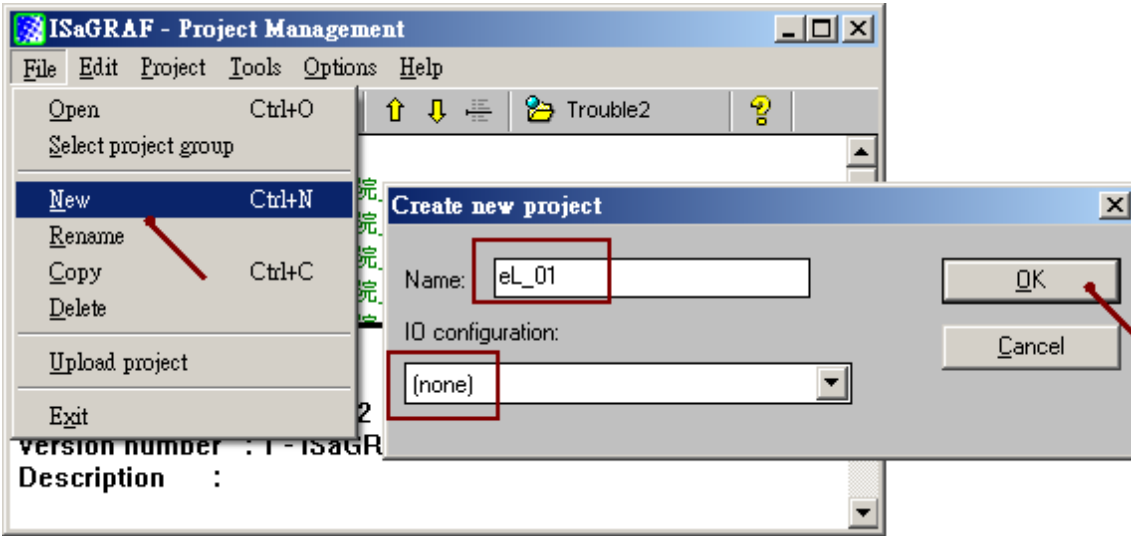
或可到 [http://www.icpdas.com/products/PAC/i-8000/getting\\_started\\_manual.htm](http://www.icpdas.com/products/PAC/i-8000/getting_started_manual.htm) 下載它)

若你的 PC 還未安裝 ISaGRAF, 請先安裝它. (用戶至少要購買一套 ISaGRAF-256 或 ISaGRAF-32 軟體, 請參訪 <http://www.icpdas.com/products/PAC/i-8000/isagraf.htm> 下方的 Ordering Information)

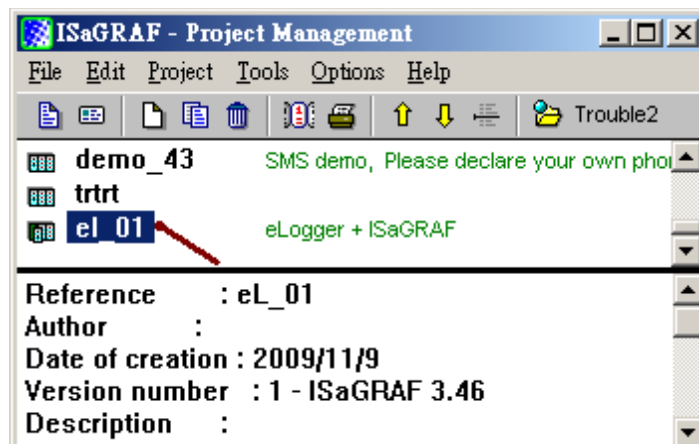
之後參考 “ISaGRAF 進階使用手冊 “ 第 1.1 節來安裝 ISaGRAF 且之後需參考 第 1.2 節來安裝 “ICP DAS utilities For ISaGRAF” .

## 1.1.1: 建立新 Project

將 ISaGRAF 運行起來,然後建立一個新的 project, 名稱可取為 “eL\_01”



之後雙擊該 Project 名稱來進入該 Project 內

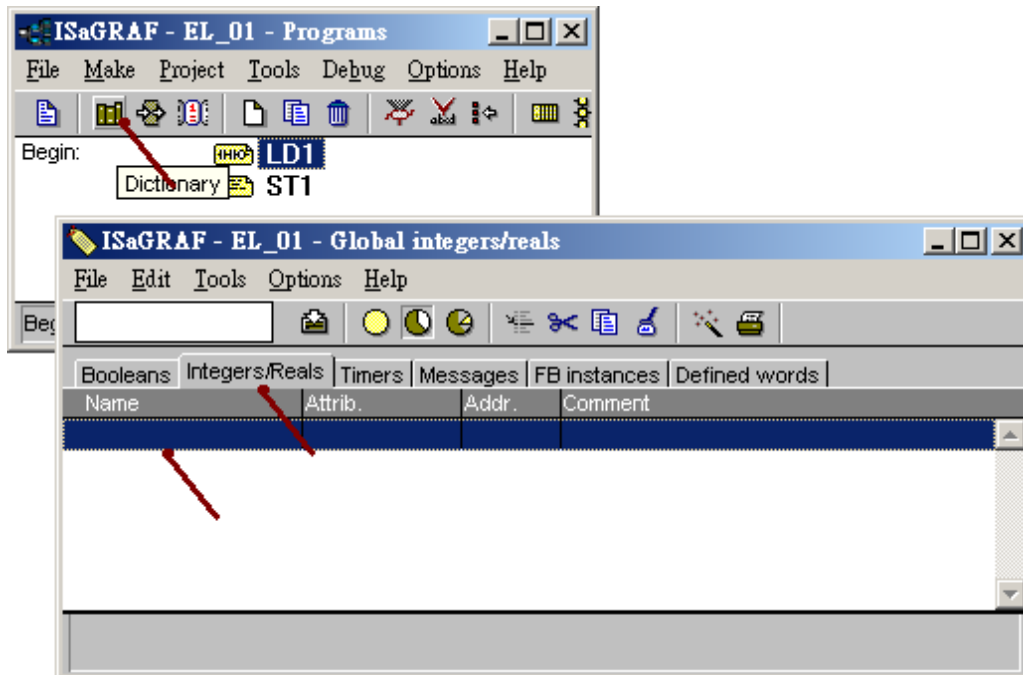


## 1.1.2: 建立 ISaGRAF 變數

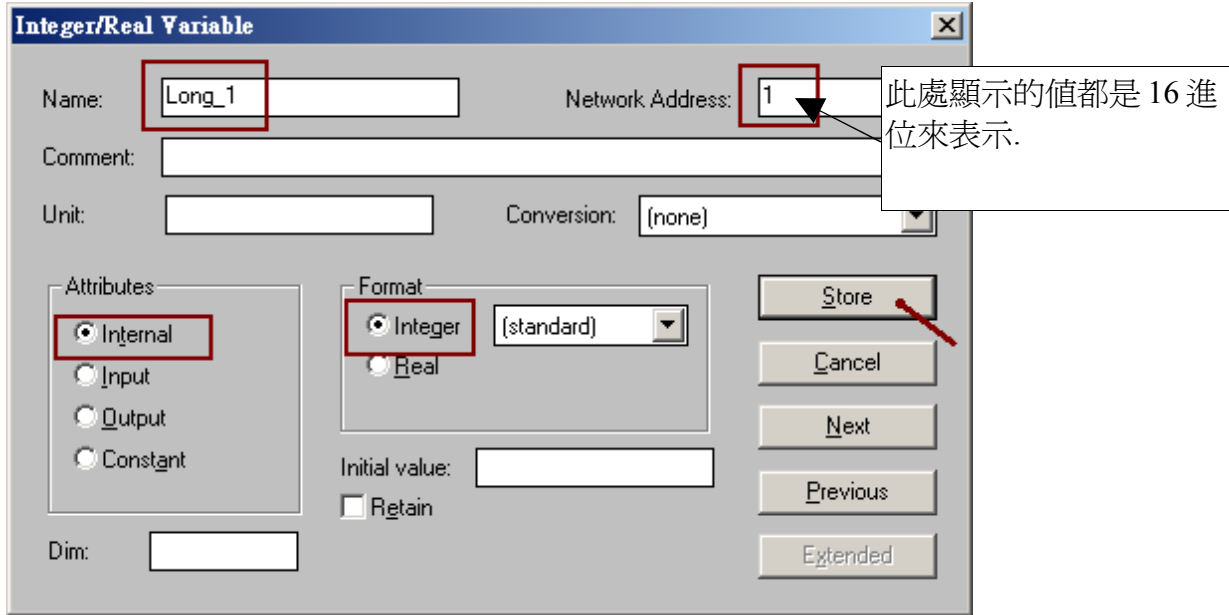
本範例會宣告以下的 ISaGRAF 變數

Name	Type	Attribution	NetWork Addr.	說明
Long_1	Integer	Internal	1	用來跟 eLogger 32-bit Long 溝通
Word_3	Integer	Internal	3	用來跟 eLogger 16-bit Integer 溝通
Word_4	Integer	Internal	4	用來跟 eLogger 16-bit Integer 溝通
Float_5	Real	Internal	5	用來跟 eLogger 32-bit Float 溝通
OUT_101	Boolean	<b>Output</b>	101	連接到 slot 1: I-87055W 的 DO1
OUT_102	Boolean	<b>Output</b>	102	連接到 slot 1: I-87055W 的 DO2
M1	Boolean	Internal	0	
DIR	Boolean	Internal	0	需指定初值為 True

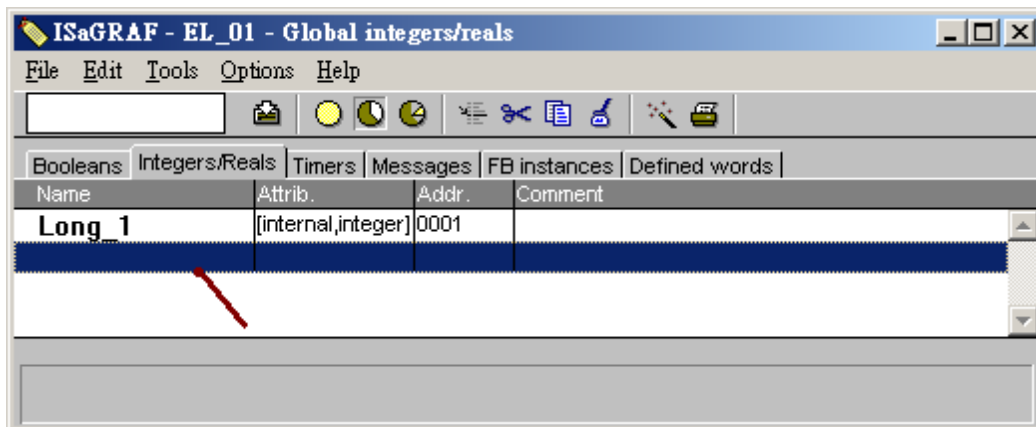
進入 Dictionary 內, 先點選 “Integers/Reals”, 然後雙擊下方的藍色區域來開啓 變數宣告畫面.



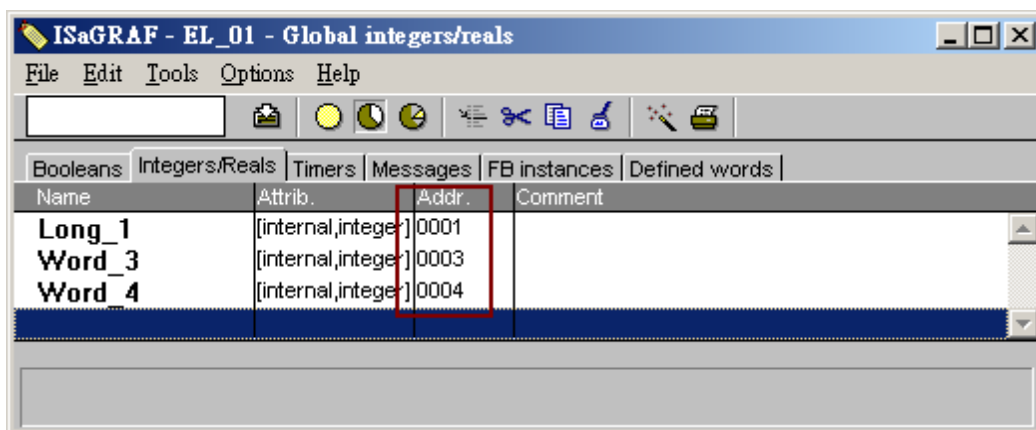
輸入變數名稱爲 “Long\_1”, Network Address 爲 1



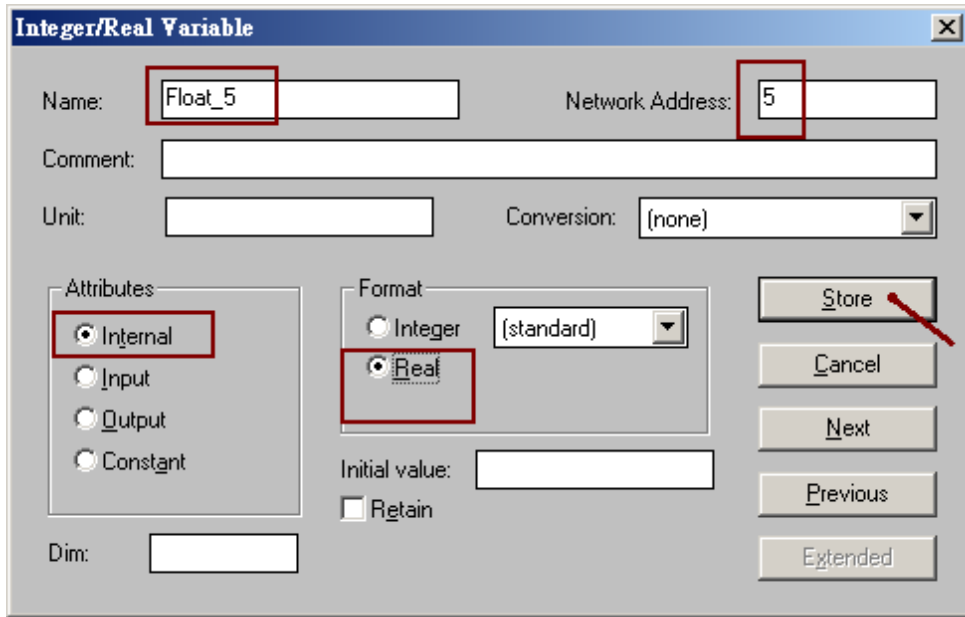
完成後點選 “Store”, 之後會出現下方的畫面. 之後採用相同方式來雙擊 下方區域來宣告出 Word\_3 與 Word\_4, 注意他們的 NetWork Address 分別是指定爲 3 與 4



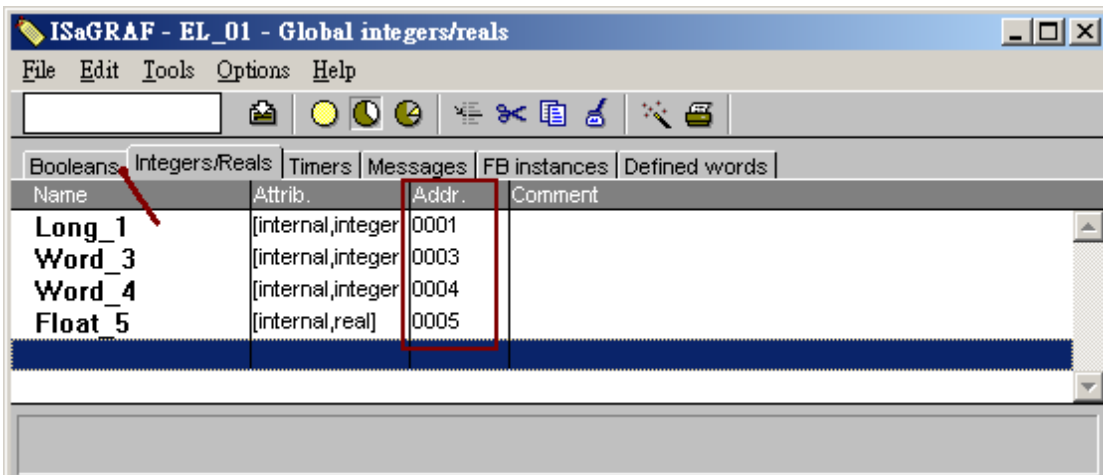
之後得到如下畫面.



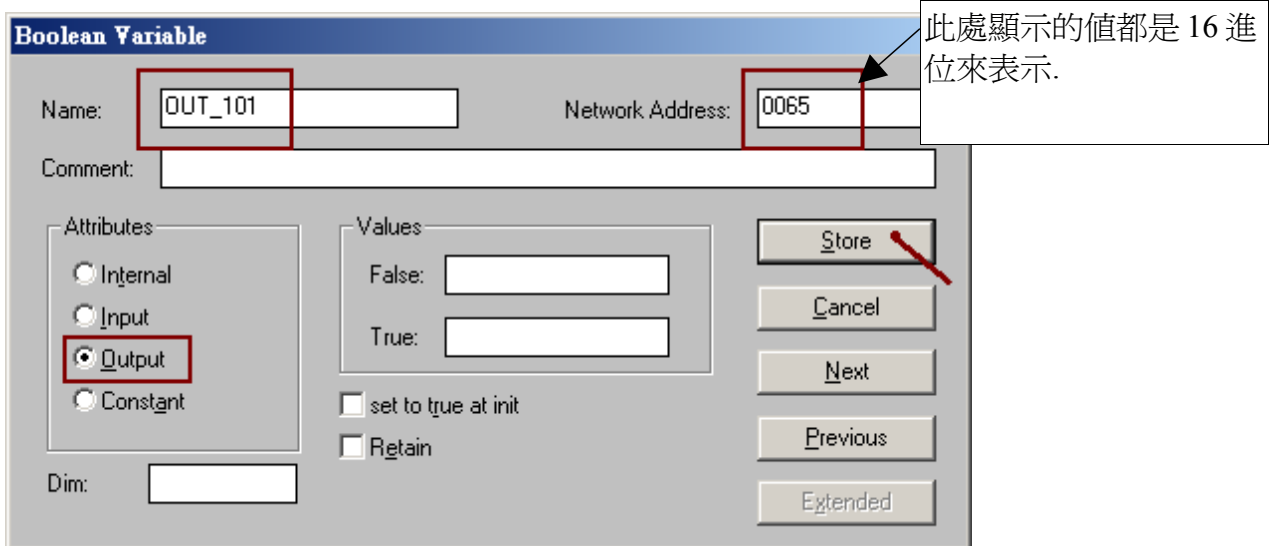
接下來用同方法來宣告 Float\_5 ,但要注意它是 REAL 型態 (非 Integer), 且 Network Addr 為 5.



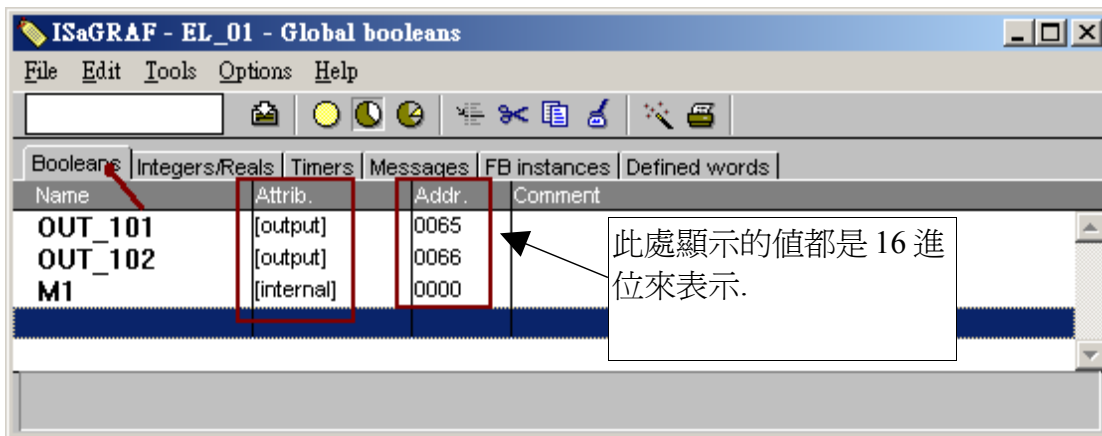
然後得到以下畫面. 接下來要宣告 Boolean 變數, 請點選 “Booleans” .



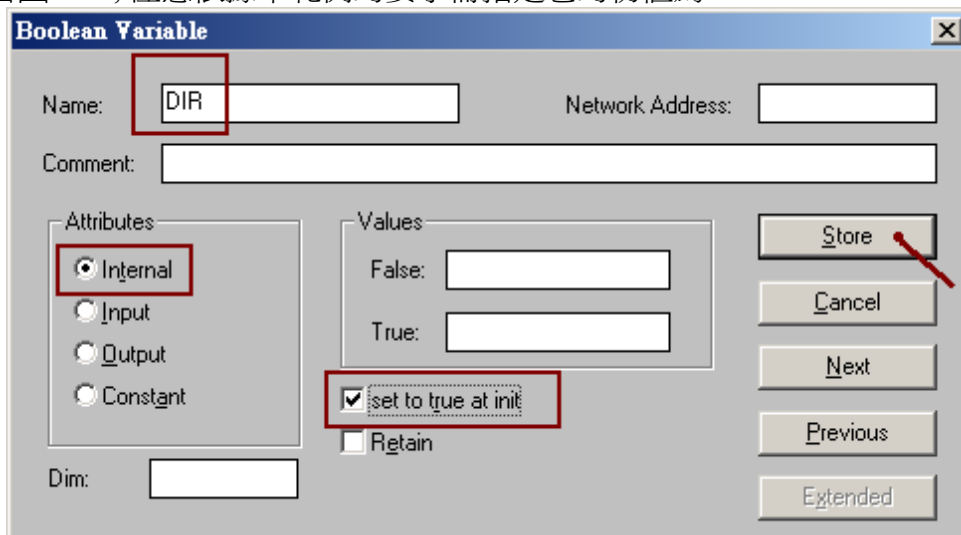
使用類似的方法宣告出 OUT\_101 與 OUT\_102 與 M1 , 注意 OUT\_101 與 OUT\_102 的屬性為 Output (Addr 分別是 101 與 102), 而 M1 的屬性為 Internal .



然後得到以下畫面

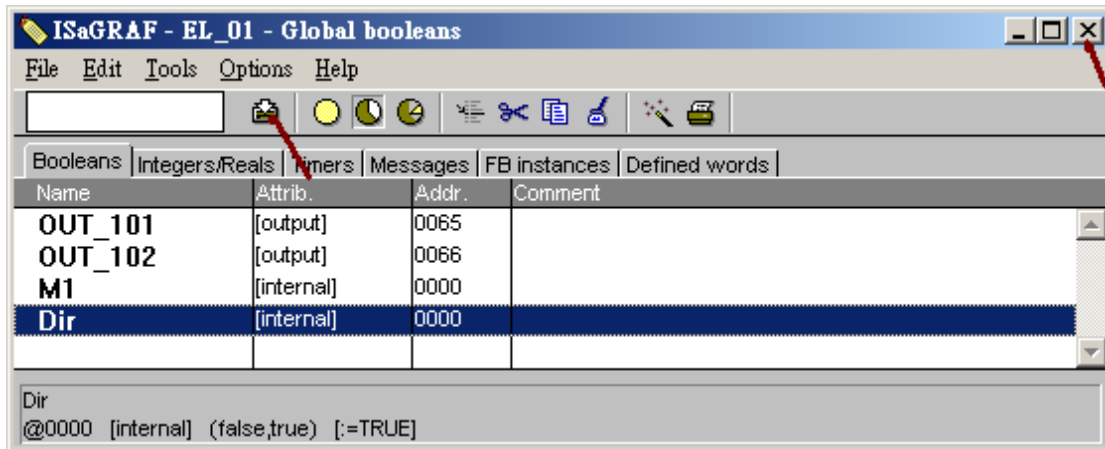


最後再宣告出 DIR, 注意依據本範例的要求需指定它的初值為 True.

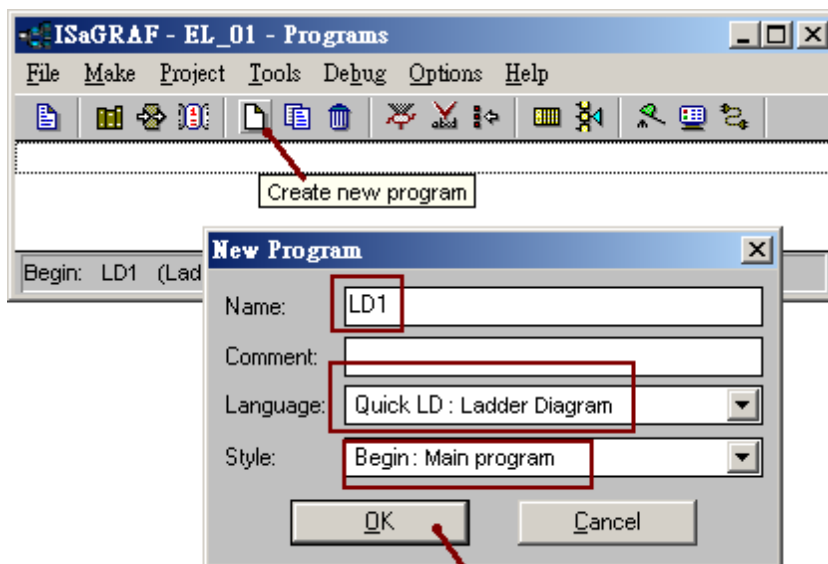


### 1.1.3: 編寫階梯圖程式 LD1

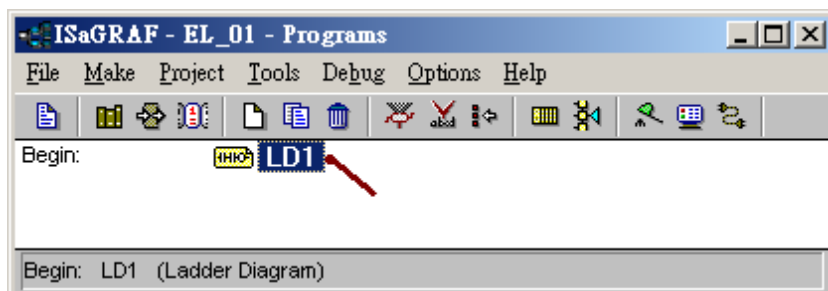
完成變數宣告後, 需儲存並離開該宣告視窗.



本範例接下來要建立一個階梯圖程式 LD1 如下, 輸入程式名稱爲 “LD1”, 語法爲 “Quick LD : Ladder Diagram”, 型態爲 “Begin: Main program”, 再按下 OK.

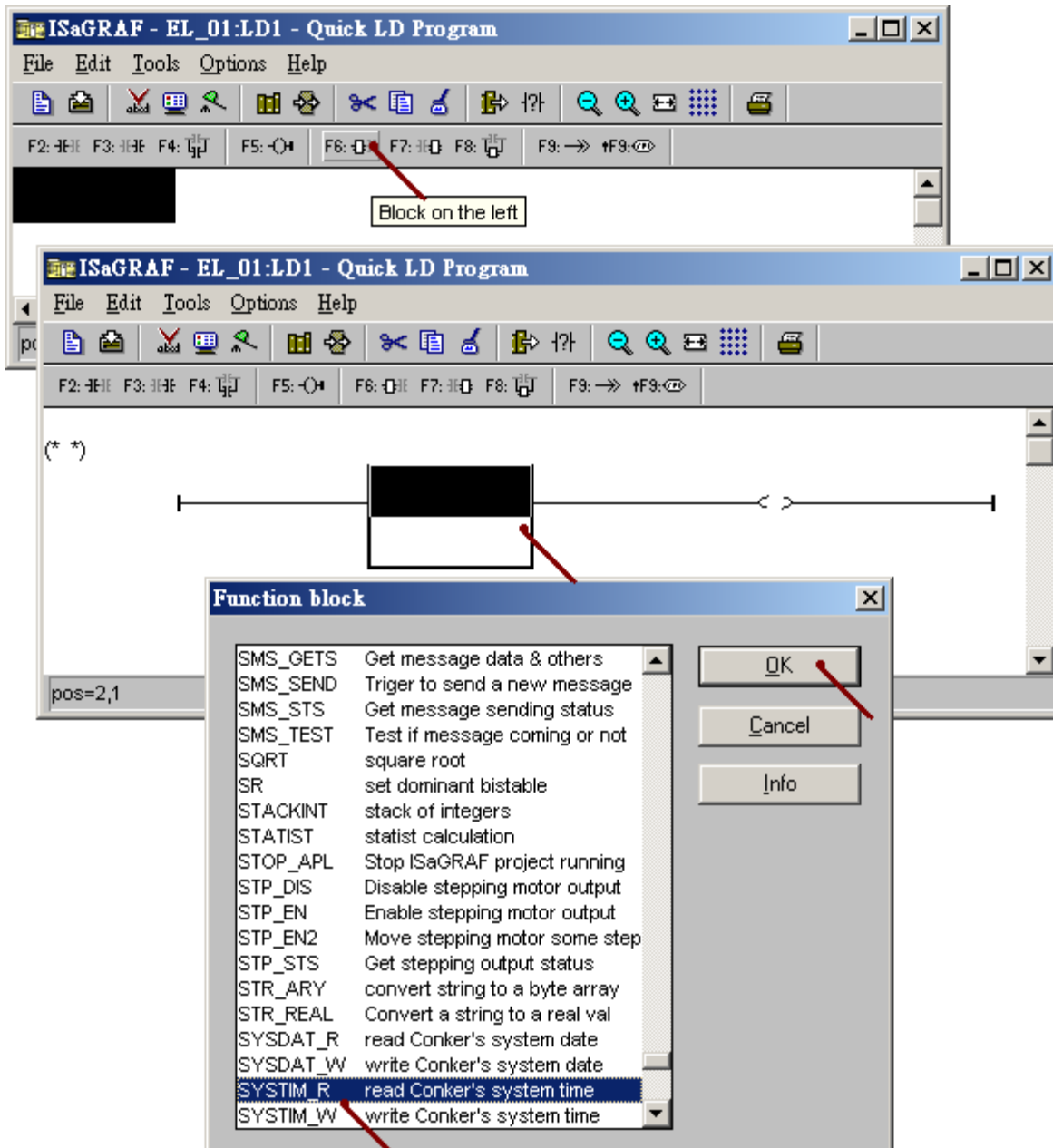


就會得到以下畫面, 請用 mouse 雙擊 “LD1”來編寫階梯圖程式.

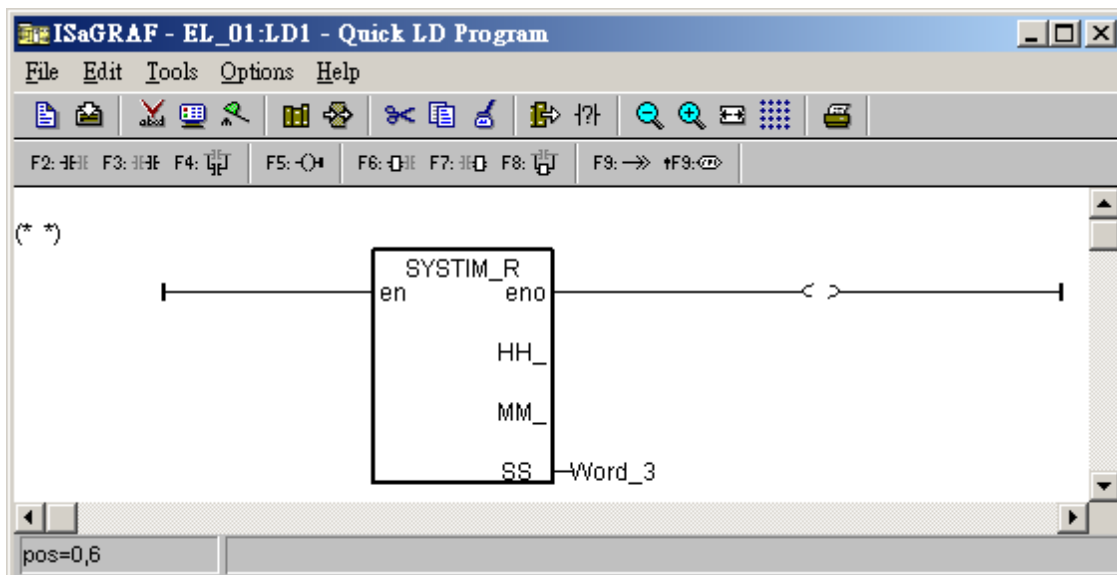
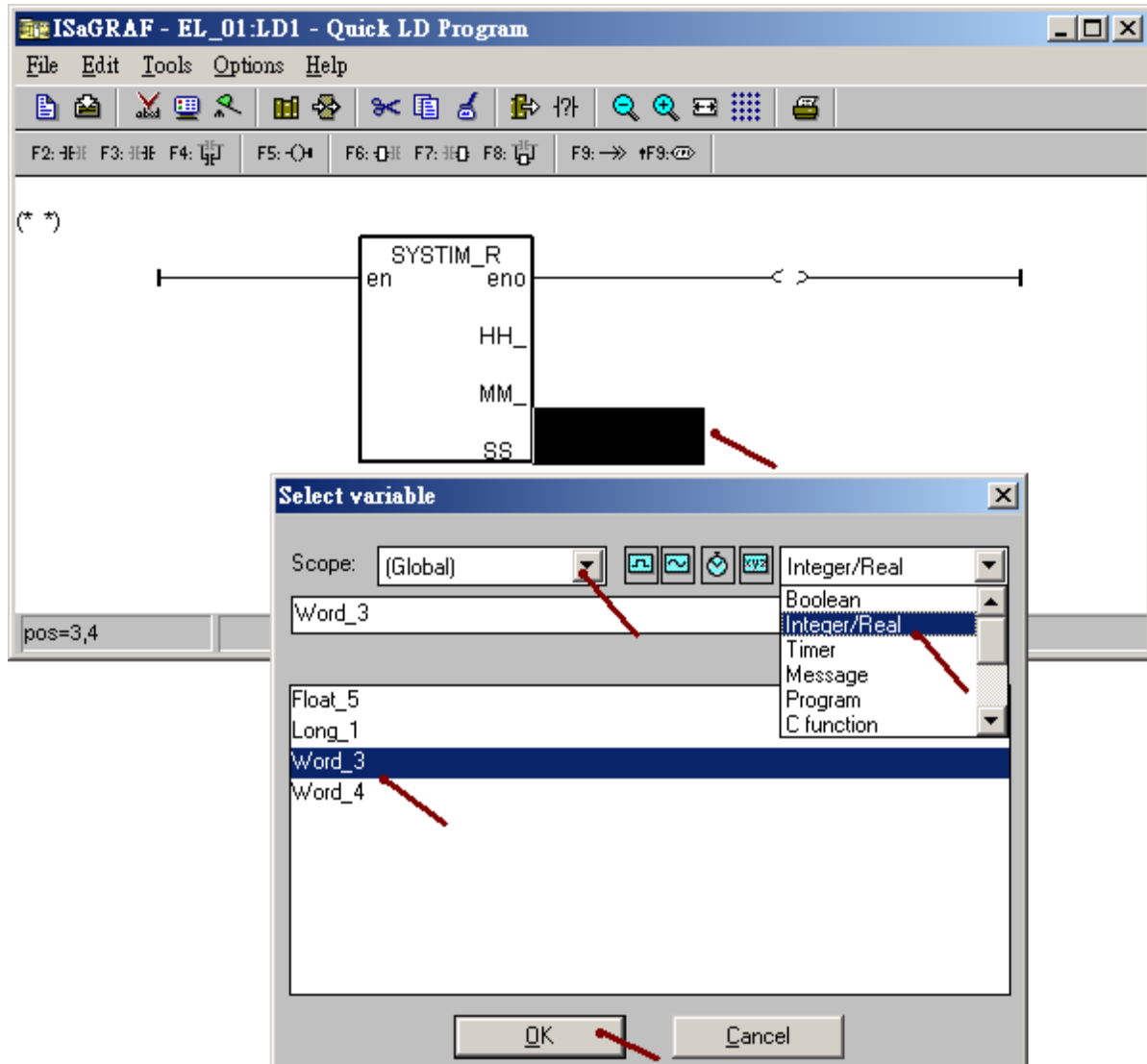




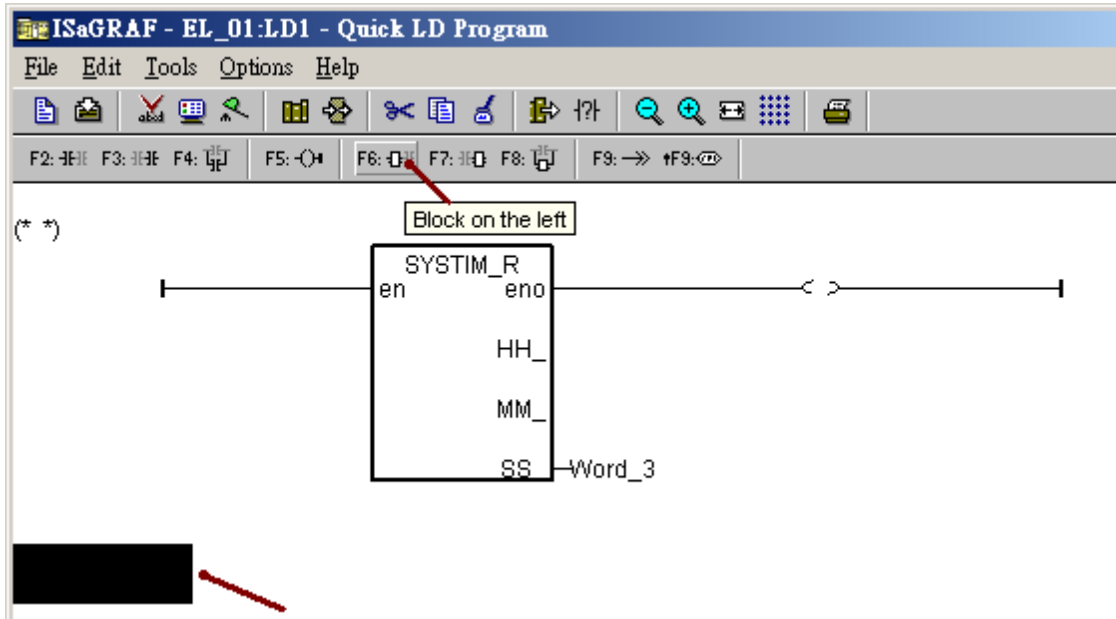
首先要新增一個 SYSTM\_R 方塊來取得 控制器的時間. 先點選 Block on the left , 之後在該空白方塊內用 Mouse 雙擊, 選取 SYSTM\_R 如下.



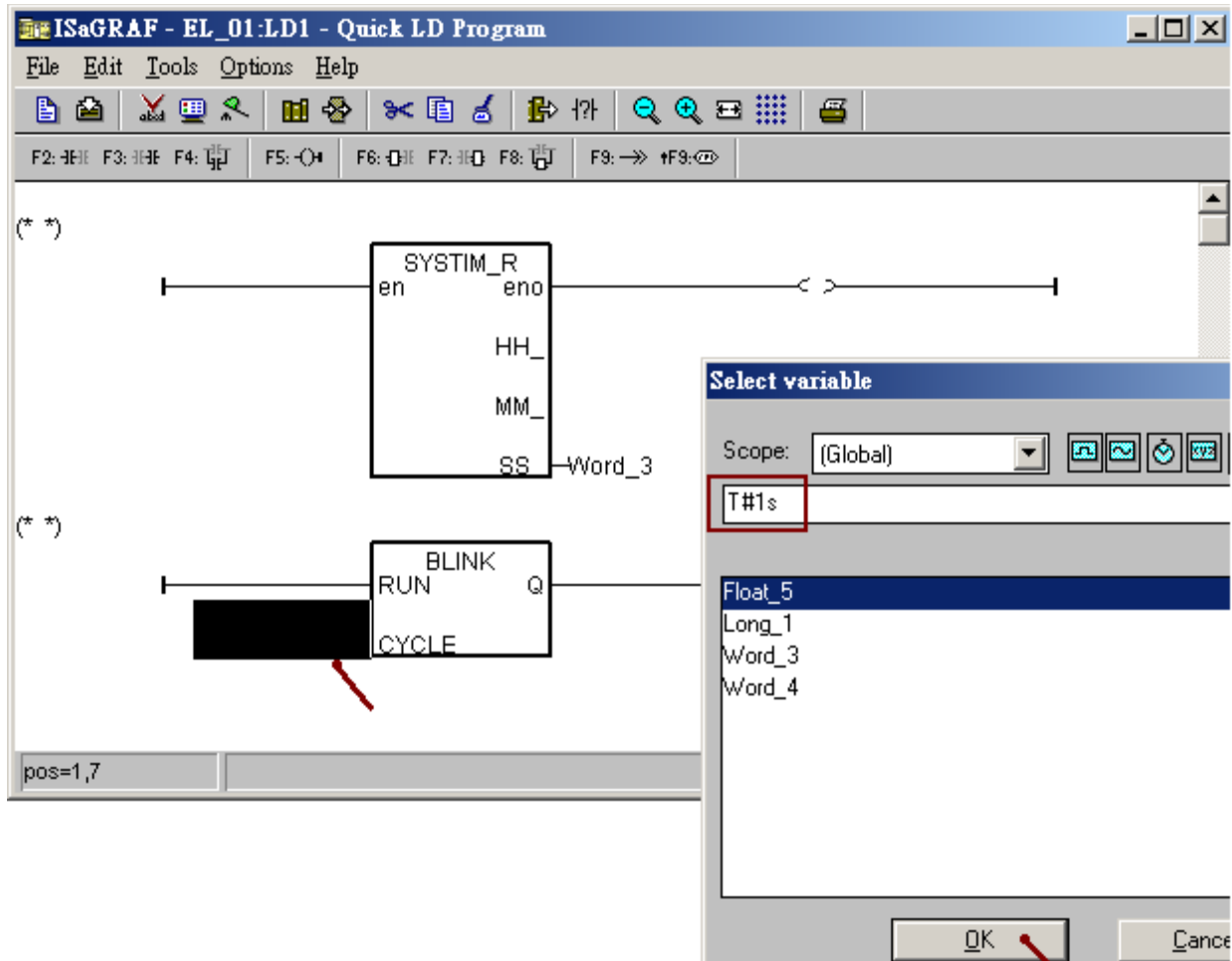
之後 Mouse 雙擊 SYSTIM\_R 的傳回參數 SS\_ 的右方那一格來選取 Word\_3 整數。



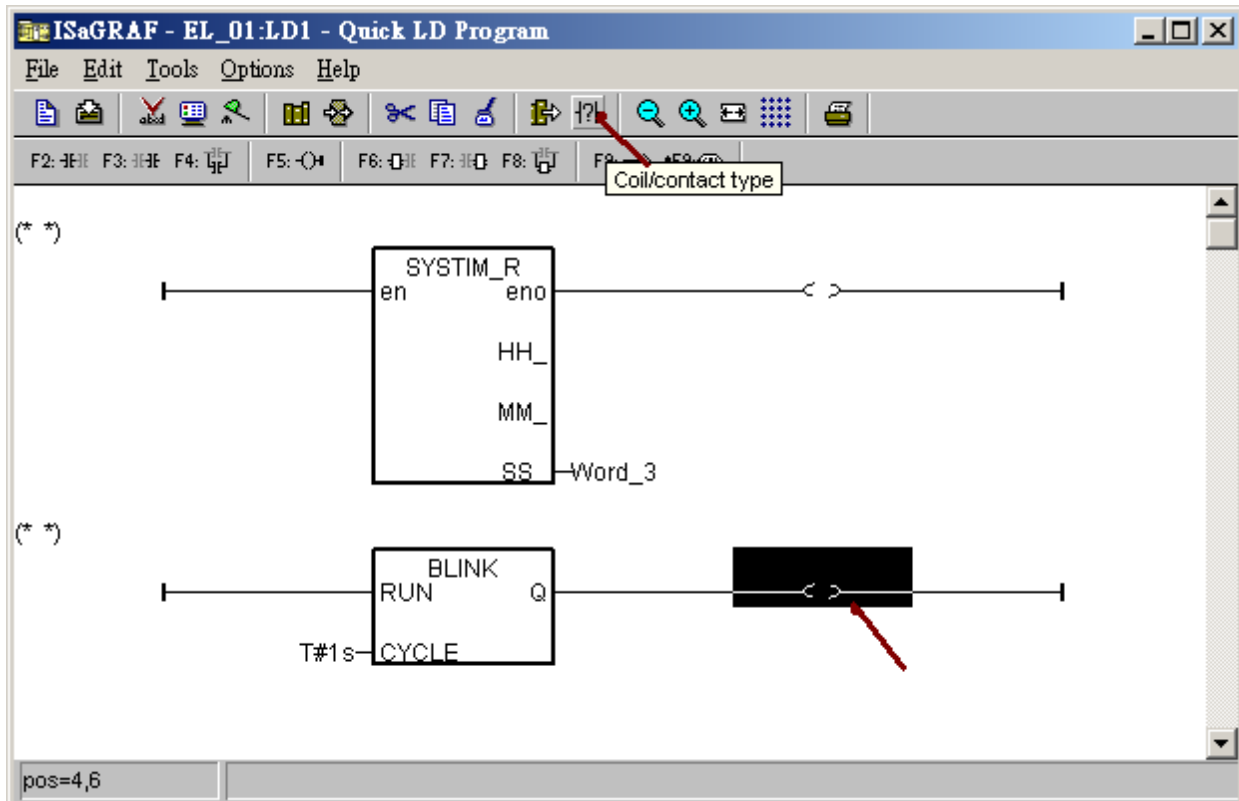
接下來本範例需要產生每秒一個 Pulse True 給 M1 變數, 先將郵標移到下方, 然後點選 Block on the left 來插入一個 Blink 方塊。



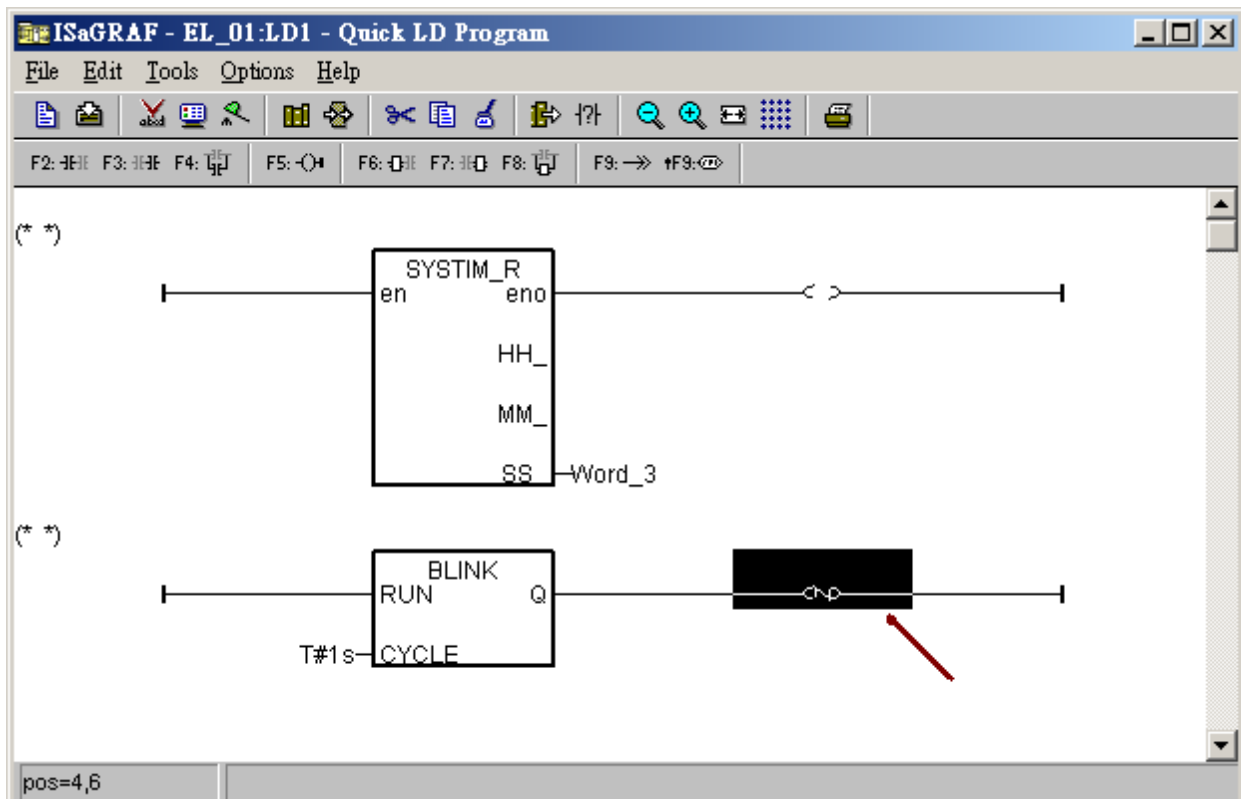
然後 Mouse 雙擊 BLINK 的輸入參數 “CYCLE”的左方那一格, 用鍵盤輸入 T#1S .

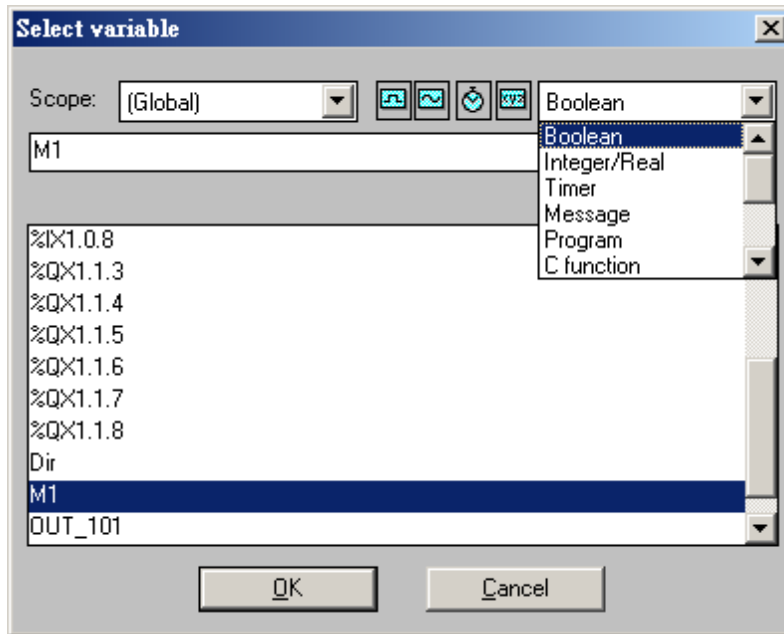


之後將郵標移到右方的輸出線圈, 點選 Coil / Contact type 數次來選取 N 線圈.

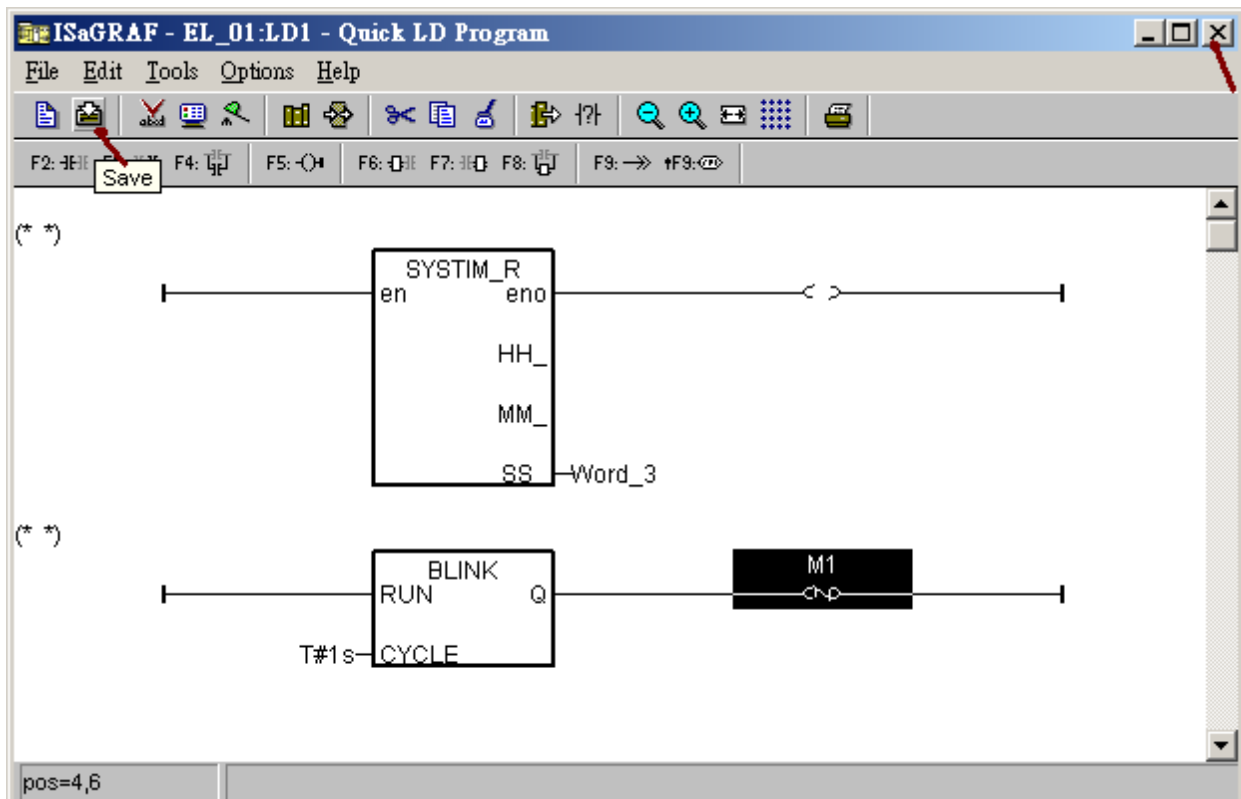


選好 N 線圈後, Mouse 對它雙擊 來輸入 M1 變數.



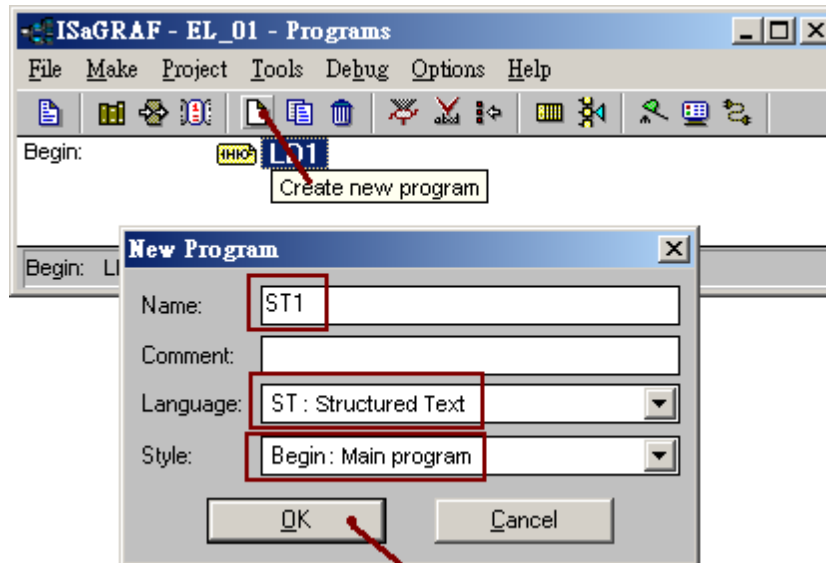


之後就完成了 LD1 程式, 按下 save 然後離開階梯圖畫面.

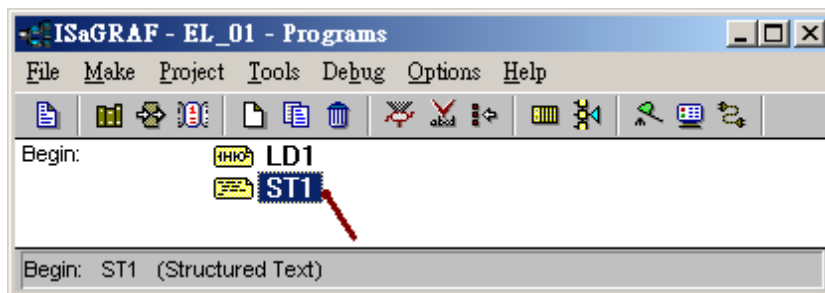


## 1.1.4: 編寫 Structured Text 程式 ST1

接下來要建立 ST1 程式如下。



然後 mouse 雙擊 ST1 來進入該程式內。



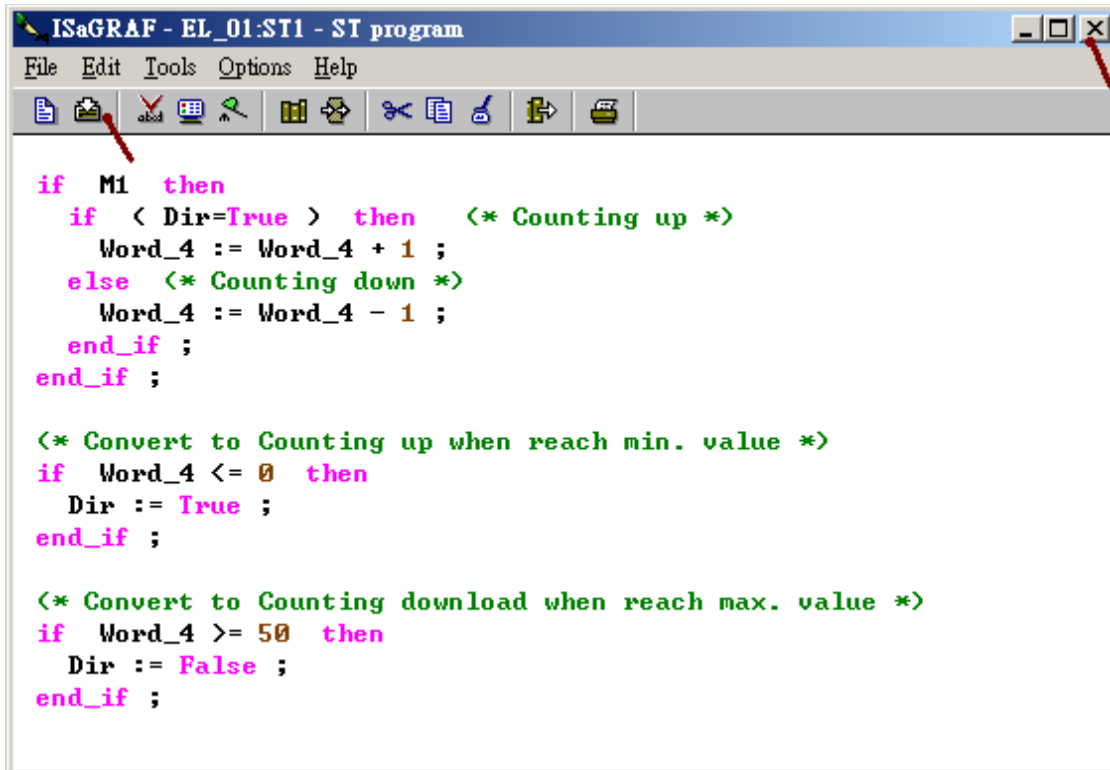
請輸入以下程式碼。

```
if M1 then
  if ( Dir=True ) then
    Word_4 := Word_4 + 1 ; (* Counting up *)
  else
    Word_4 := Word_4 - 1 ; (* Counting down *)
  end_if;
end_if;

if Word_4 <= 0 then
  Dir := True ; (* reach Min. value, change to counting up *)
end_if;

if Word_4 >= 50 then
  Dir := False ; (* reach Max. value, change to counting down *)
end_if;
```

之後得到以下畫面, save 後離開.



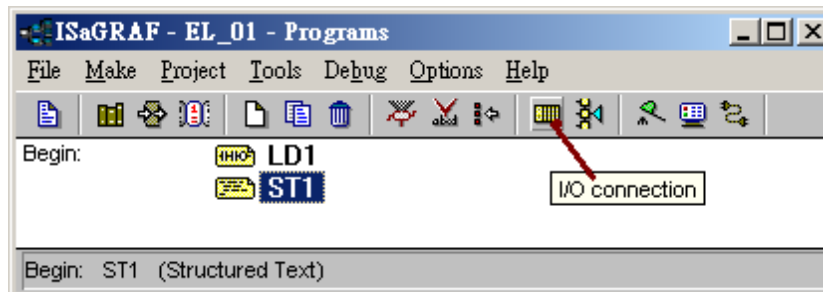
```
ISaGRAF - EL_01:ST1 - ST program
File Edit Tools Options Help
if M1 then
  if < Dir=True > then  (* Counting up *)
    Word_4 := Word_4 + 1 ;
  else (* Counting down *)
    Word_4 := Word_4 - 1 ;
  end_if ;
end_if ;

(* Convert to Counting up when reach min. value *)
if Word_4 <= 0 then
  Dir := True ;
end_if ;

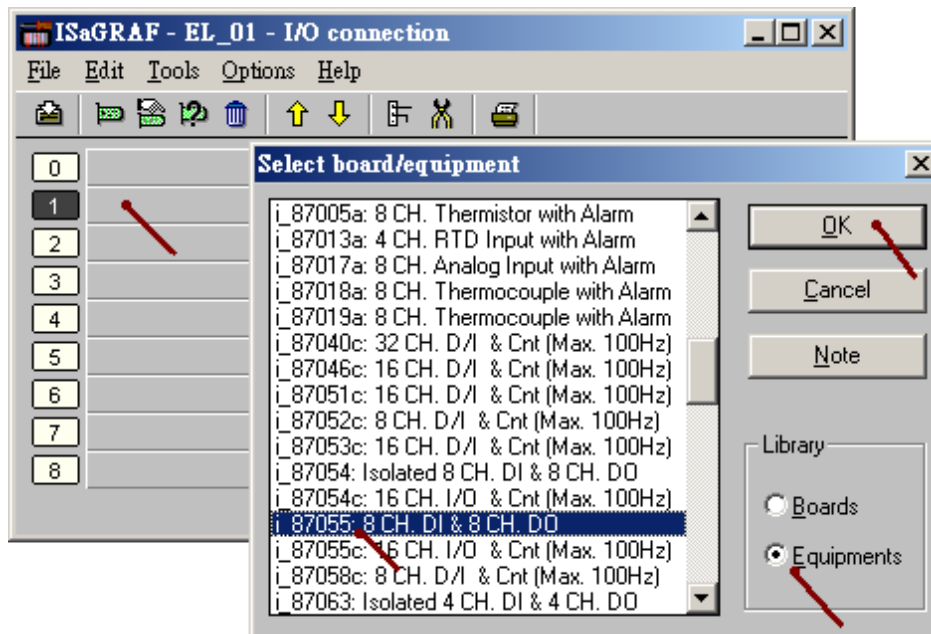
(* Convert to Counting down when reach max. value *)
if Word_4 >= 50 then
  Dir := False ;
end_if ;
```

### 1.1.5: 進行 I/O 連結

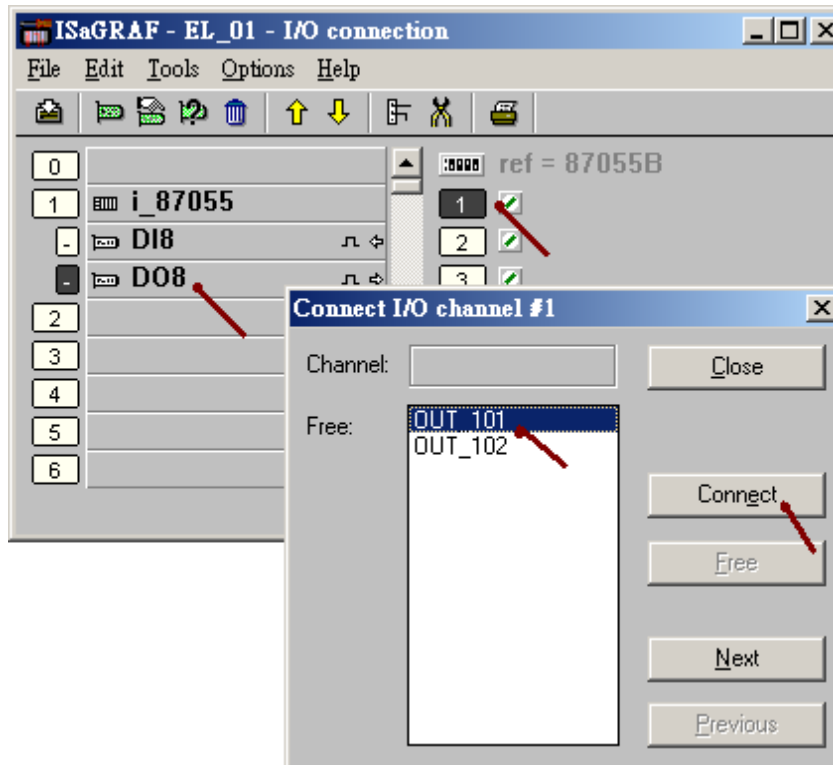
本範例有宣告 2 個屬性為 Output 的 Boolean 變數, 名稱爲 OUT\_101 與 OUT\_102, 必需對它們進行 I/O 連結, 本例是將 1 塊 I-87055W 板卡插在 WP-8447 的 slot 1 上 (最左邊的 I/O slot 號碼是 slot 0)。請點選 I/O connection 來進行 I/O 連結。



然後用 mouse 雙擊 Slot 1 來插入 i\_87055 . (若找不到 “Equipments” 內的 i\_87055 表示 “ICP DAS Utilities For ISaGRAF” 並未安裝, 請參考 “ISaGRAF 進階使用手冊 “ 第 1.2 節的說明)

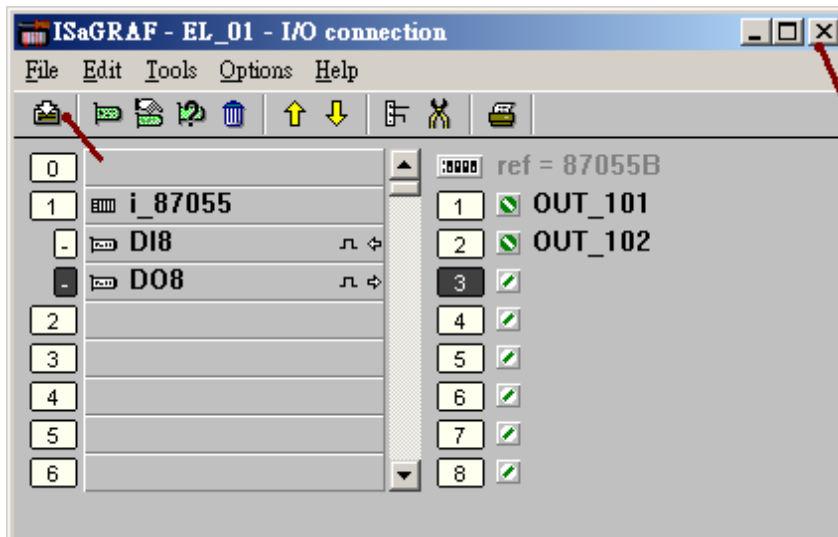


接下來將 Boolean 輸出變數連接在 Channel 1 與 2 上. (若找不到 OUT\_101 與 OUT\_102 表示該變數並未宣告其屬性為 Output, 請參考本文件 1.1.2 節將 OUT\_101 與 OUT\_102 改成宣告為 Output 屬性)



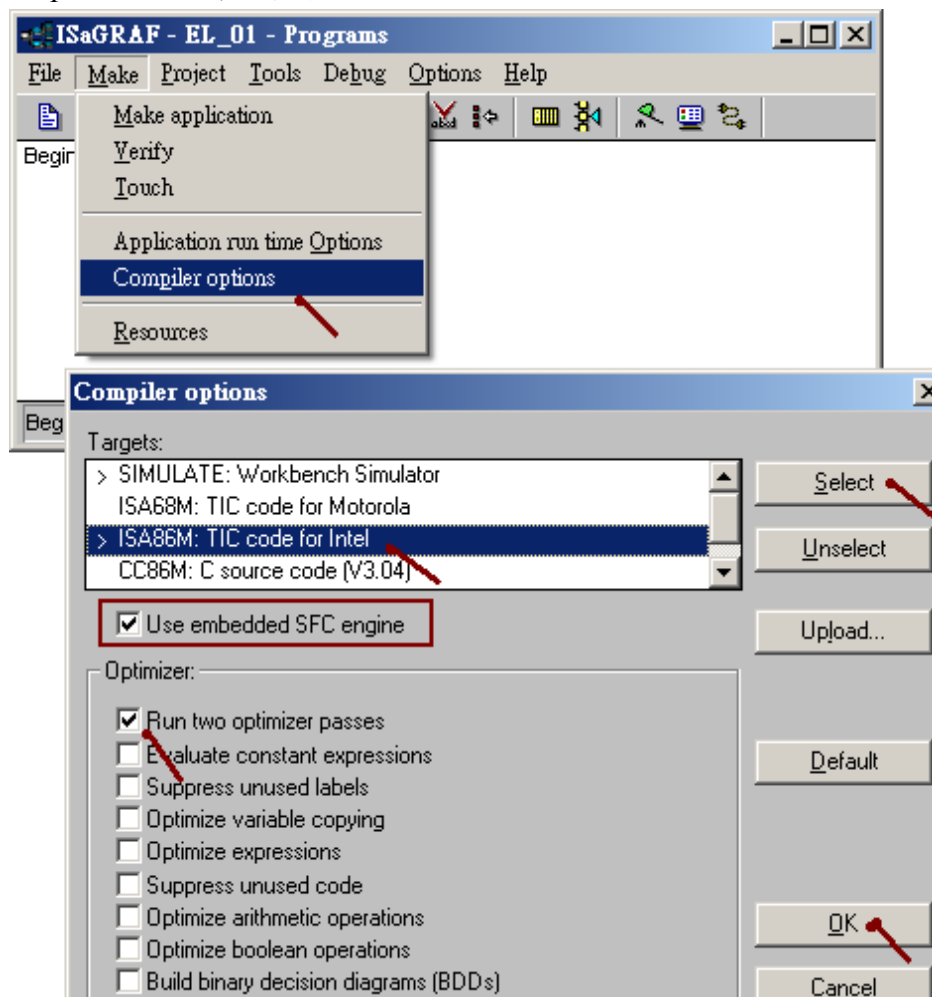


完成後 Save 並離開該畫面.

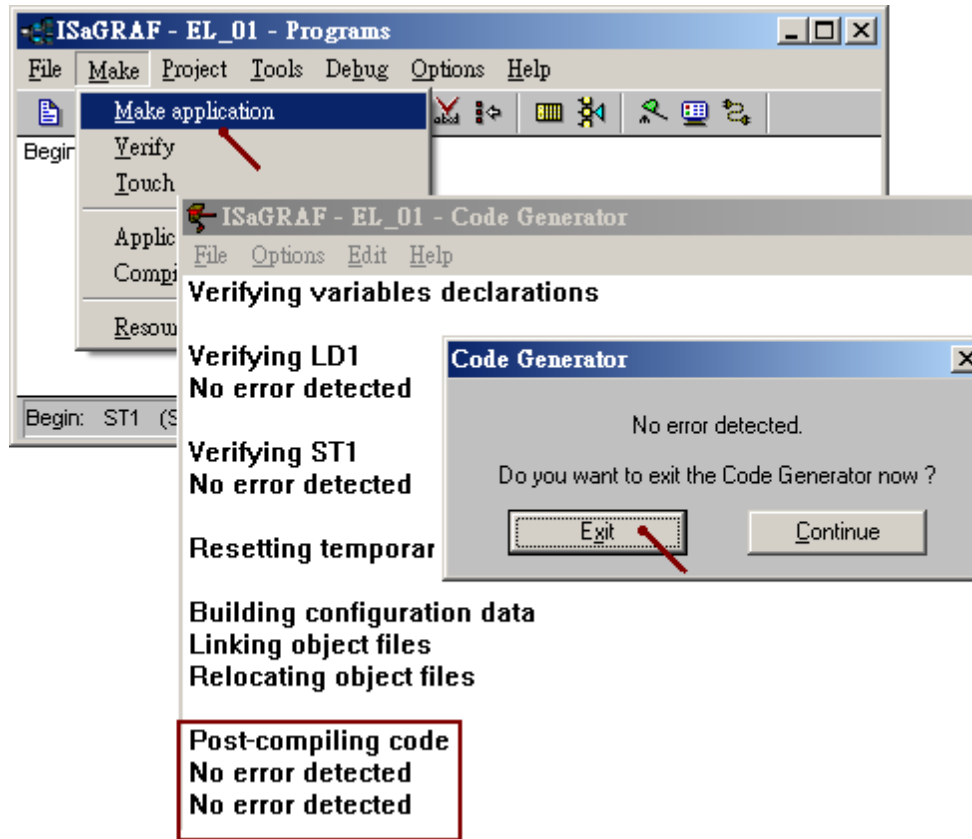


### 1.1.6: 編譯 ISaGRAF Project

在進行編譯前要先設好編譯選項. 一定要選取 “ISA86M: TIC code for Intel” 與 “Use embedded SFC engine” 與 Optimizer 的第一個項目.



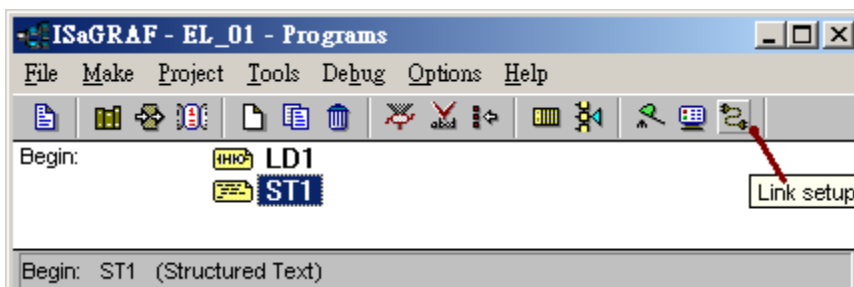
設好 Compiler Option 後進行編譯如下。



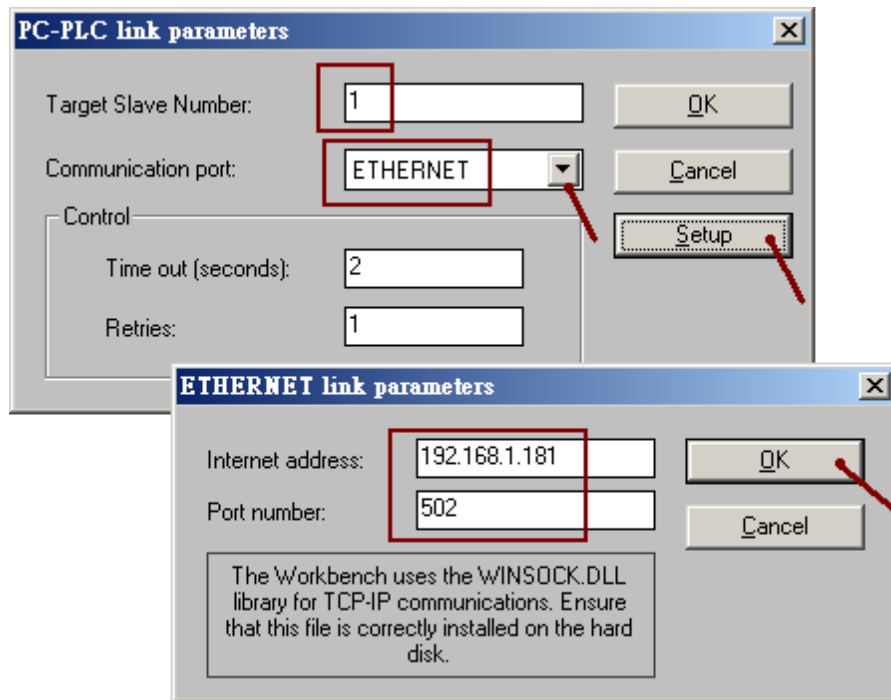
若都沒有 Error 發生, 恭喜你, 成功了!

### 1.1.7: 下載 ISaGRAF Project 到 WP-8xx7 內

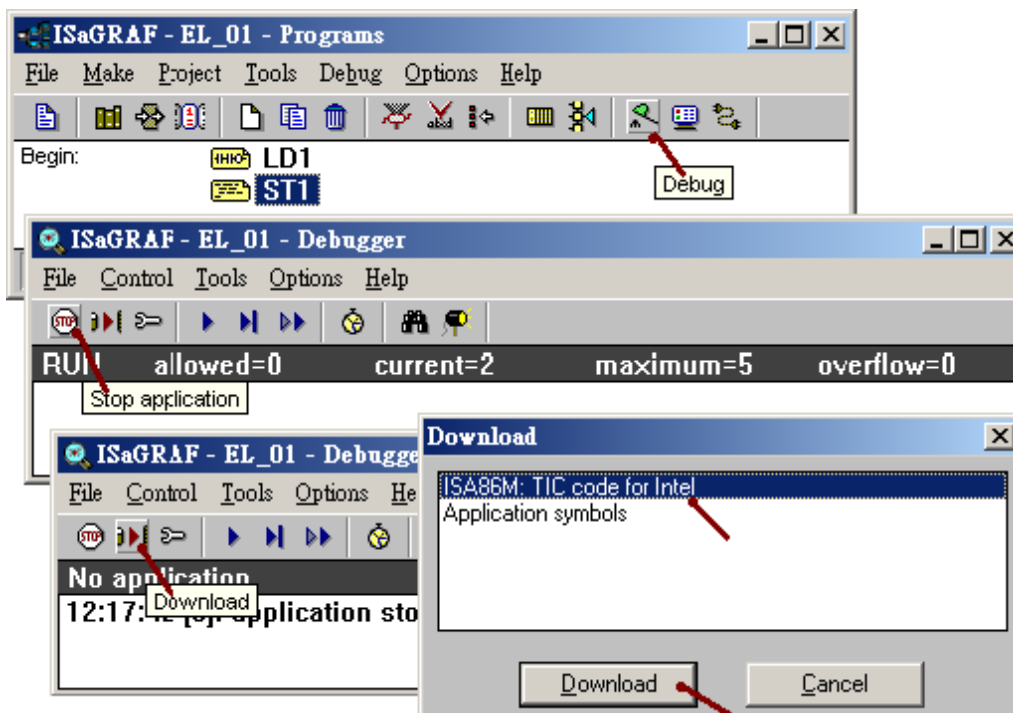
接下來需設定 Link Setup.



WP-8xx7 內定是由 Ethernet 來下載程式. 需設好要連到那台控制器的 IP 與 Port Number (需為 502).



之後按下 Debug, 若有連上控制器, 則先 Stop Application 然後再 Download “TIC code for Intel”.



做到這裏, ISaGRAF project 的 編寫, 編譯 與 下載到控制器內 就算完成了!

## 1.2: 編寫 eLogger HMI 畫面

當運作 eLogger RunTime 出現問題時, 請參考本文件第 1.3 節來解決.

eLogger 區分為 Developer 與 RunTime, Developer 指的是安裝在 PC 上的 eLogger 開發工具, 客戶可以使用它來設計出 HMI 畫面與圖形控制元件. RunTime 指的是安裝在 WP-8xx7 或 VP-2xW7 或 XP-8xx7-CE6 等控制器內的 eLogger RunTime 程式, 當用戶完成 PC 的 eLogger HMI 畫面後, 必需將此 eLogger HMI project 丟進 控制器內, 讓 eLogger RunTime 來運行它, 之後它就能在控制器所連接的 VGA 螢幕上顯示出該 HMI 畫面 .

### 1.2.1: 安裝 eLogger Developer 到 PC 與 安裝 eLogger RunTime 到控制器內

若你的 PC 並未安裝 eLogger Developer, 請至以下網址下載最新版的 eLogger Developer.

[ftp://ftp.icpdas.com/pub/cd/winpac-8xx7/napdos/elogger/elogger\\_developer/](ftp://ftp.icpdas.com/pub/cd/winpac-8xx7/napdos/elogger/elogger_developer/)

(此路徑內容日後會適時更新為最新版本).之後安裝它到 PC 上.

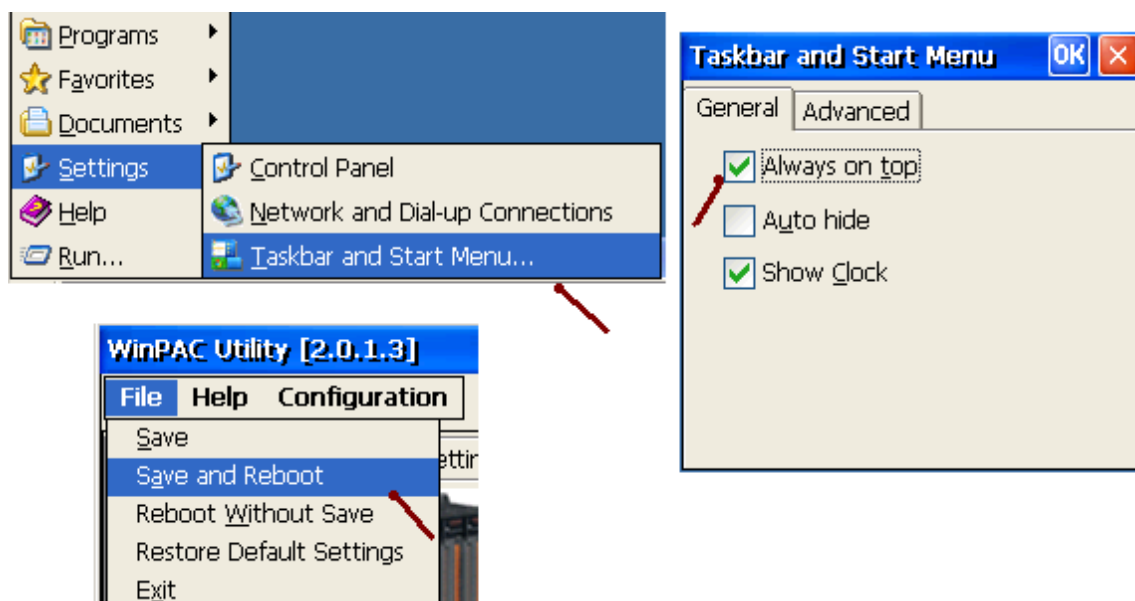
接下來若你的 WP-8xx7 或 VP-2xW7 並未安裝 eLogger RunTime, 請至

[ftp://ftp.icpdas.com/pub/cd/winpac-8xx7/napdos/elogger/elogger\\_runtime/](ftp://ftp.icpdas.com/pub/cd/winpac-8xx7/napdos/elogger/elogger_runtime/)

下載最新版的 eLogger RunTime. (此路徑內容日後會適時更新為最新版本)

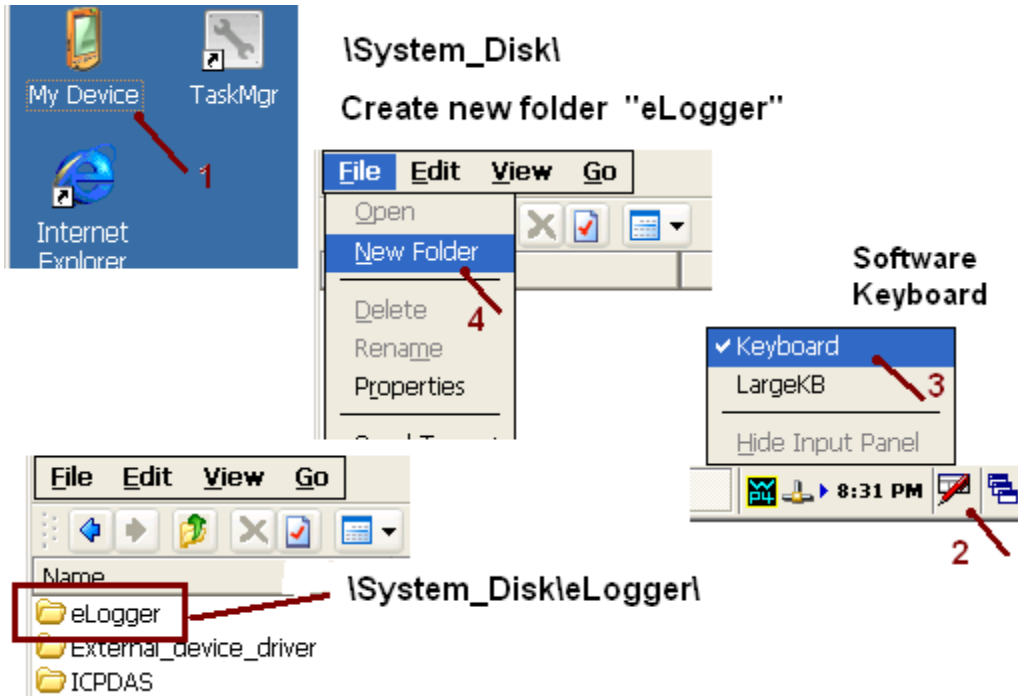
在設計階段時, 需將 Taskbar 顯示出來, 如下.

將 WP-8xx7 的 VGA port 接上一台 VGA Monitor, USB port 接上一個 Mouse. 然後設定控制器的 Taskbar 為 “Always on top”, 不可勾選 “Auto hide”. 然後運行 WinPAC utility (或 ViewPAC Utility)將此設定存到 registry 內, 控制器會自動 reset 一次.

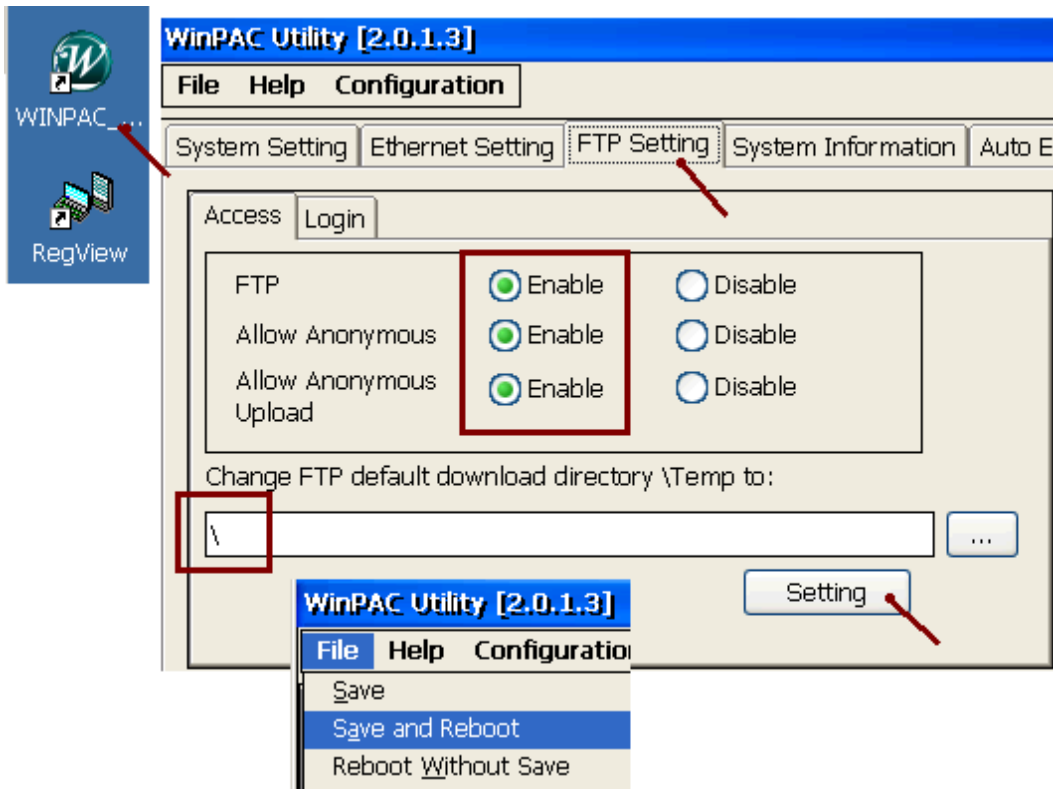


如何安裝 eLogger RunTime:

請先進入控制器螢幕上的 My Device, 進入 \System\_Disk\ 內建立一個 eLogger 子目錄



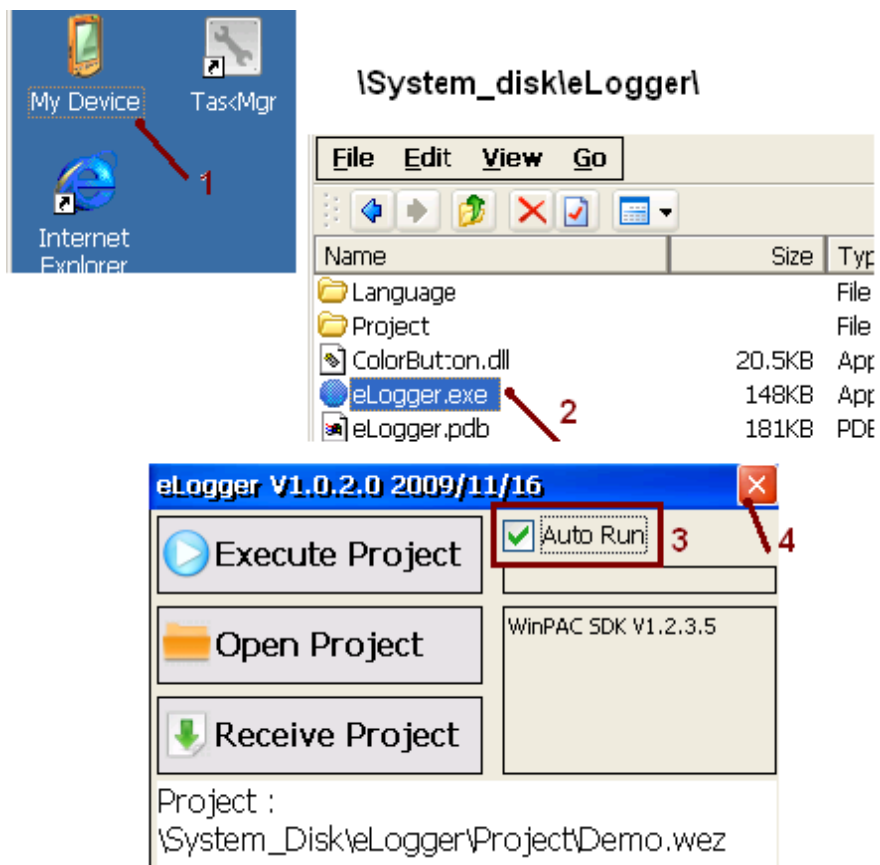
再將 eLogger RunTime 所有的 file 傳送到 WP-8xx7 或 VP-2xW7 的 \System\_Disk\ eLogger\ 內, 可用 PC 的 ftp 工具程式來傳送, 先確定 WinPAC Utility > FTP Setting 內有 Enable FTP 與 其路徑是設在 “\”. 若不是請設好它, 並按下 Setting, 然後要運行 File > Save and Reboot 將此設定儲存.



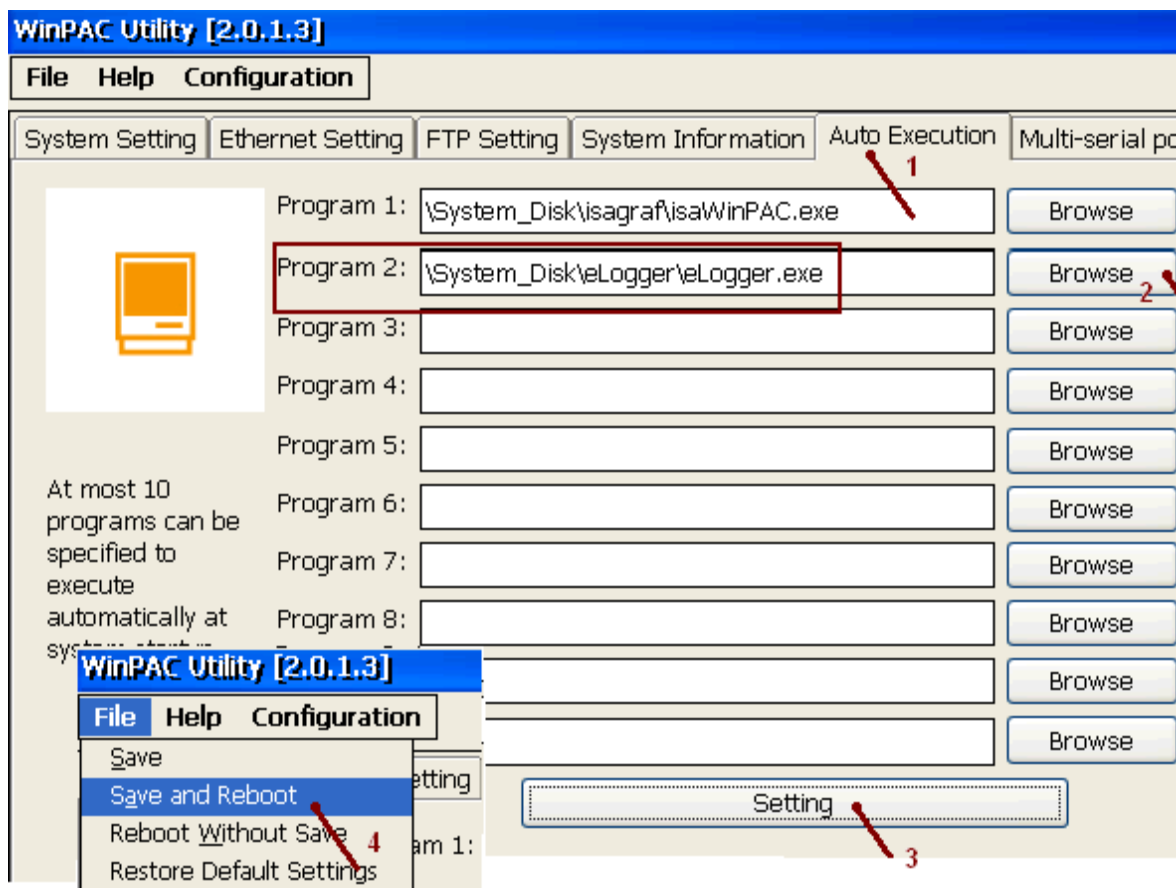
之後在 PC 上運行 Internet Explorer, 輸入控制器的 IP, 例如 **ftp://192.168.1.181**  
然後進入 \System\_Disk\eLogger\, 從 PC 上將 eLogger RunTime 的所有 file 都 Copy 進去 控制器的 \System\_Disk\eLogger\ 路徑內.



之後點選控制器內的 My Device 進入 \System\_Disk\eLogger\ 內, 將 eLogger.exe 運行起來.  
然後勾選 Auto Run, 再離開 eLogger.exe .



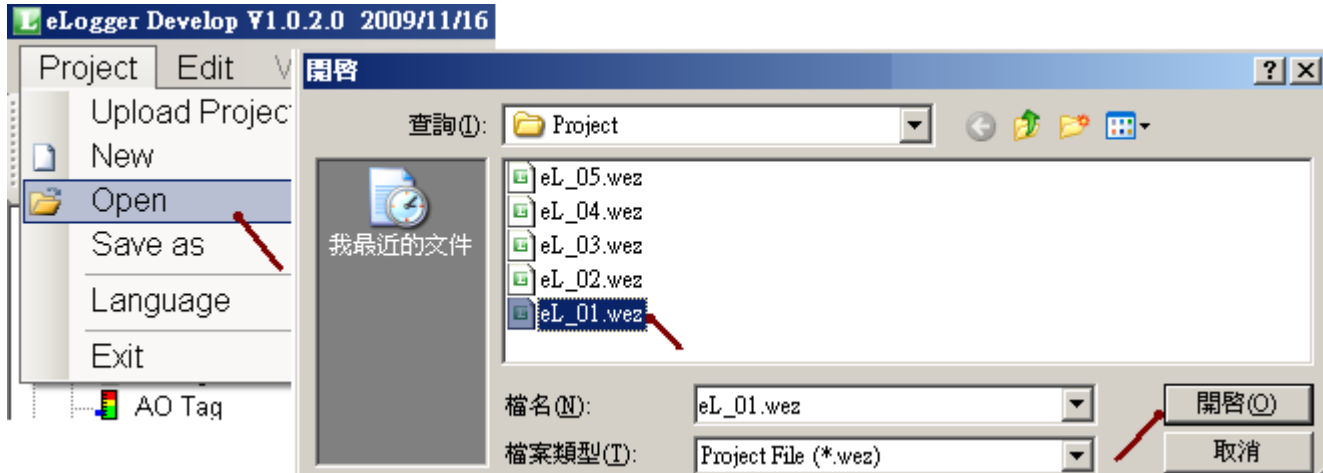
接下來運行 WinPAC Utility (或 ViewPAC Utility) 設定 “Auto Execution” 的第 2 個為 “\System\_Disk\Logger\Logger.exe” (WP-8xx7 與 VP-2xW7 的第 1 個 Auto Execution 需是各別的 ISaGRAF driver), 之後要運行 File > Save and Reboot 將此設定存入 Registry 內。



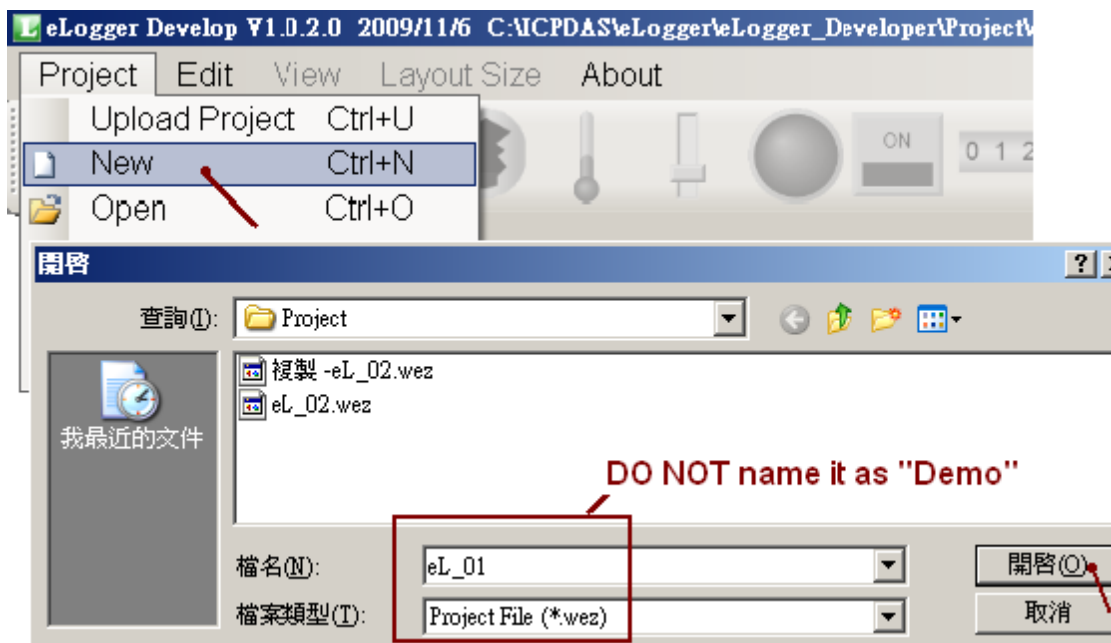
若安裝 eLogger Run Time 發生問題, 請參考 1.3 節來解決問題.

## 1.2.2: 建立 eLogger Project

若 user 想直接開啓本範例 “eL\_01.wez”請將從 FAQ-115 上取得的 “eL\_01.wez” ~ “eL\_06.wez” 複製到 PC 的 c:\ICPDAS\eLogger\Logger\_Developer\Project\ 內, 然後運行 eLogger Developer, 點選 Project > Open 來開啓它。



若要建立新的 Project, 則運行 Project > New 來建立一個 “eL\_01” project. (不可命名為 Demo )

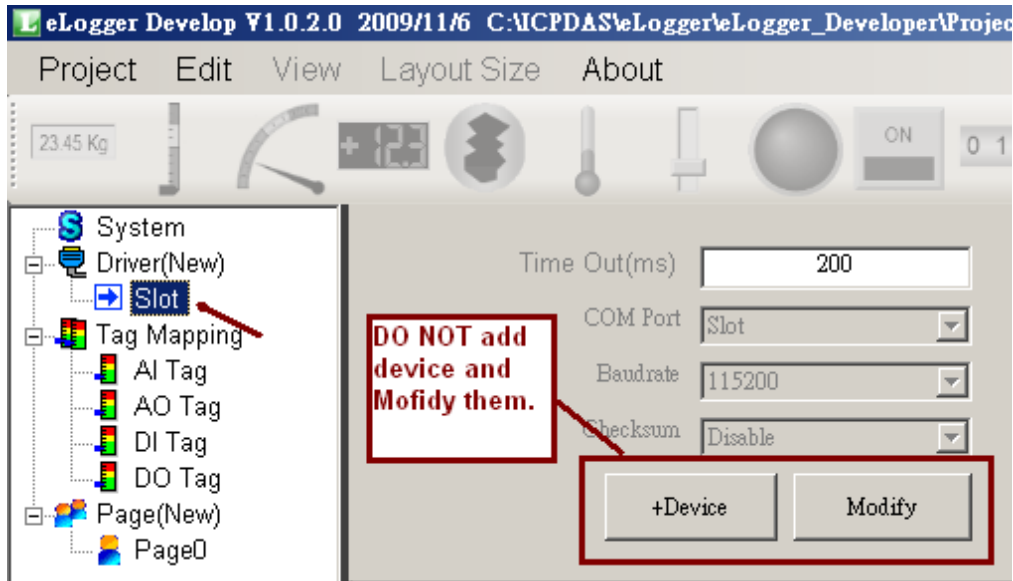


當 eLogger 是要與 ISaGRAF SoftLogic 一起在控制器內運行時, 請不要勾選 “Local Database” 這個項目, Sampling Time 建議是設成 1 秒。



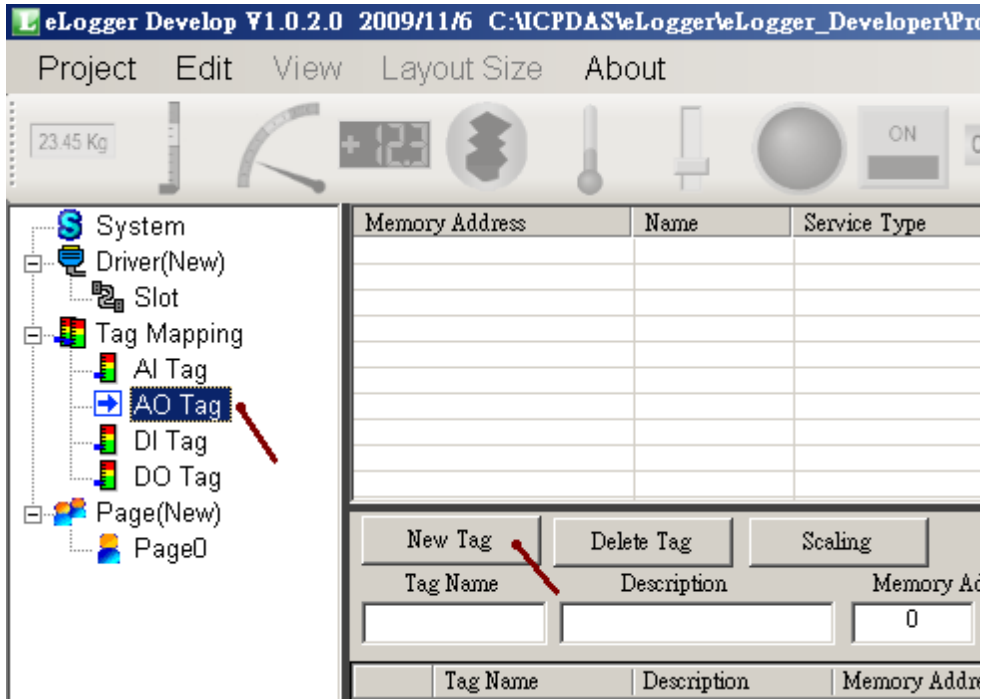


並且不要 新增任何 Device 與 Modify device. (eLogger 與 ISaGRAF SoftLogic 一起運行時, eLogger 不可啓用 Driver, 要維持在無 Driver 的狀態)



### 1.2.3: 宣告 eLogger 資料 Tag

eLogger 要與 ISaGRAF 的變數溝通, 只能使用 AO Tag 區 與 DO Tag 區 (eLogger 的 AI Tag 與 DI Tag 無法跟 ISaGRAF 的變數溝通).



上圖先點選 AO Tag , 之後點選 “New Tag” 來新增 數個 AO Tag 資料.

輸入要新增 4 個 AO Tag 資料.



之後先點選 第一個資料欄, 然後在上方分別輸入 Tague Name 為 “Long\_1”, Memory Add 為 1, Data Type 為 “32-bit Signed Long”, Gain 為 1, Offset 為 0 .

(依據 ISaGRAF 變數的定義, 32-bit Integer 與 Real 必須占用 2 個 Addr 號碼, 此例 Long\_1 的 Addr 為 1, 所以 2 號就不能再給其它 Tag 資料使用, 請參考 “ISaGRAF 進階使用手冊 4.2 節的說明” )

Tag Name	Description	Memory Add	Data Type	Gain	Offset
Long_1	32-bit long	1	32-bit Signed Long	1	0
AO1	AO1	(null)	16-bit Signed Inte...	1	0
AO2	AO2	(null)	16-bit Signed Inte...	1	0
AO3	AO3	(null)	16-bit Signed Inte...	1	0

然後依序輸入第 2 , 3 與第 4 個 AO Tag 資料如下

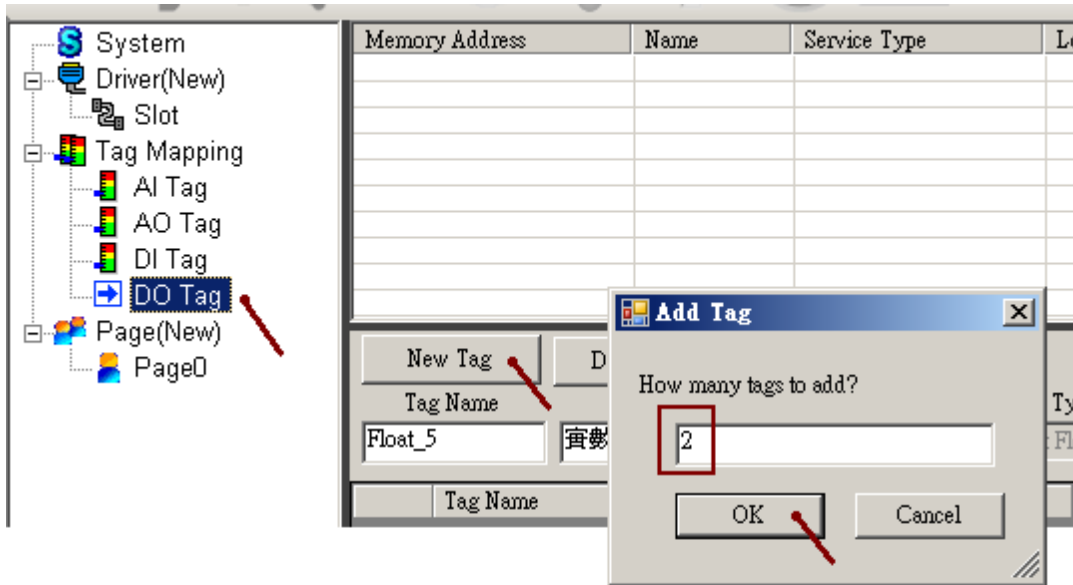
Tag Name	Memory Add	Data Type	Gain	Offset
Word_3	3	16-bit Signed Integer	1	0
Word_4	4	16-bit Signed Integer	1	0
Float_5	5	32-bit Flaot	1	0

最後得到以下畫面.

Tag Name	Description	Memory Add	Data Type	Gain	Offset
Float_5	實數 (Real)	5	32-bit Float	1	0
Long_1	32-bit long	1	32-bit Signed Long	1	0
Word_3	秒 (Seconds)	3	16-bit Signed Inte...	1	0
Word_4		4	16-bit Signed Inte...	1	0
Float_5	實數 (Real)	5	32-bit Float	1	0

接下來點選 DO Tag 來輸入 2 個 DO Tag 資料, eLogger Tag 名稱可以跟 ISaGRAF 的變數名稱不同, 因為它們彼此是以 Memory Addr (ISaGRAF 內是稱呼為 Network Address) 來識別的.

Tag Name	Memory Add	Data Type	Gain	Offset
DO_101	101	-	-	-
DO_102	102	-	-	-

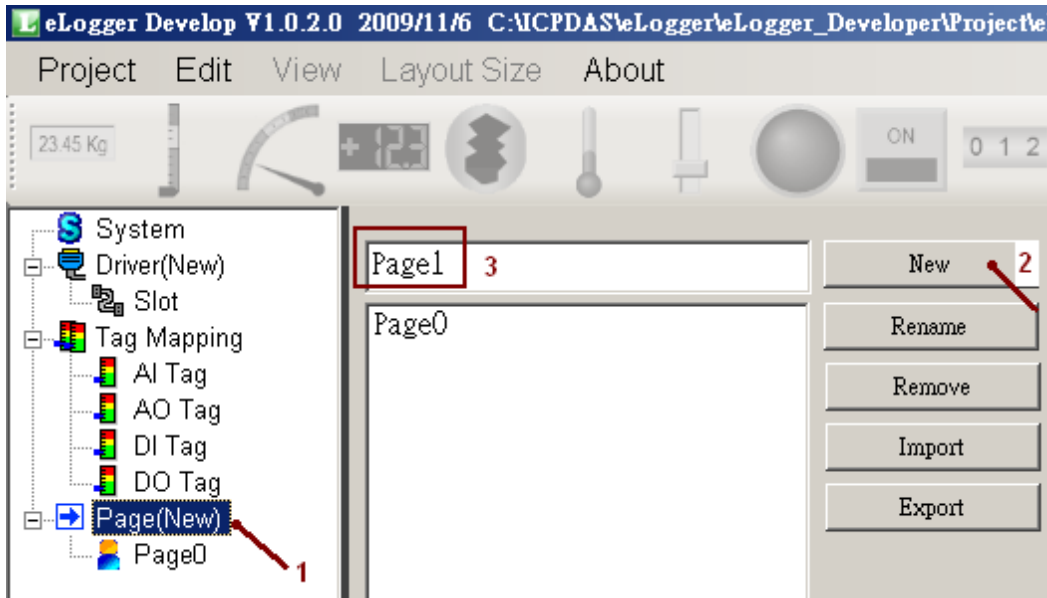


Tag Name	Description	Memory Add	Data Type
DO_102		102	32-bit Fl
DO_101		101	
DO_102		102	

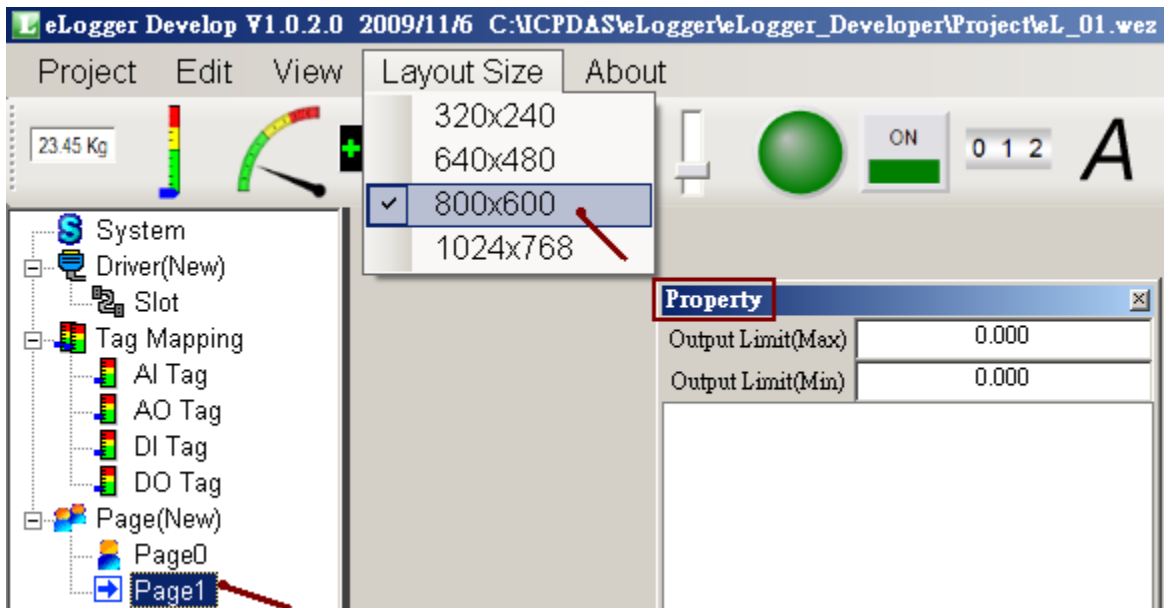
## 1.2.4: 建立與編輯 eLogger 頁面

eLogger Run Time 運行後先顯示的是 Page 0 (可以改名稱), 本範例採用 2 個頁面, Page0 與 Page1.

要新增第 2 頁, 先點選 左方的 Page(New), 然後點選右方的 New, 之後可以變更新增頁面的名稱 (也可不更改)

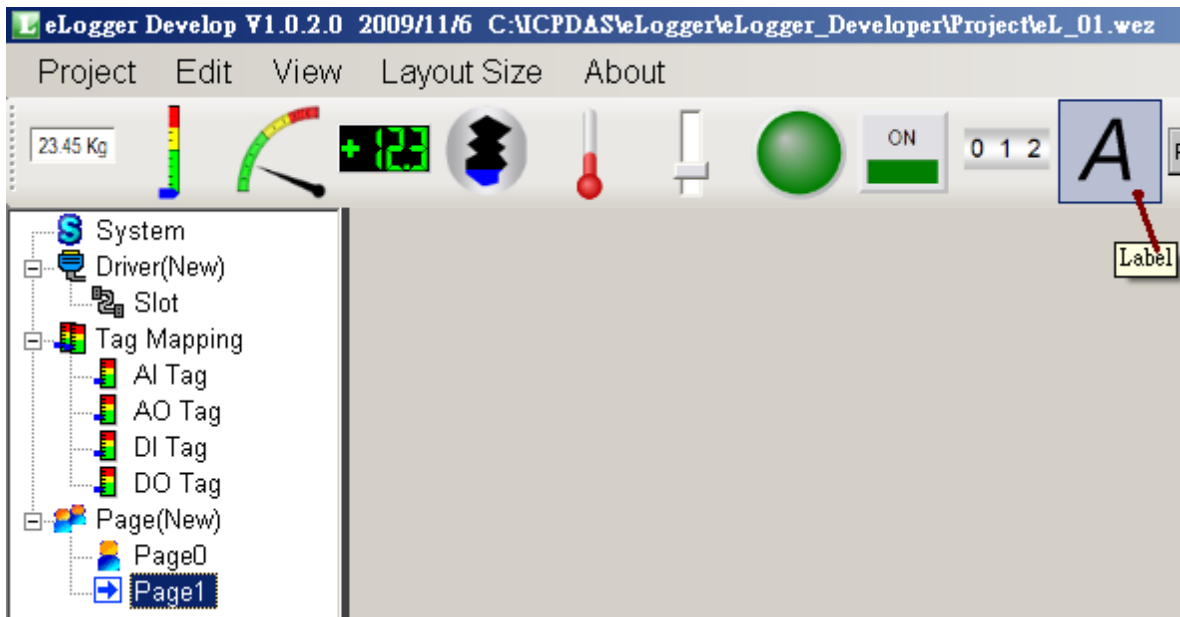


之後點選 左方的 Page1 來編輯第 2 個頁面. 請先選好你的 控制器的 VGA 所設定使用的解稀度 (WP-8x47 最大是 800 x 600, WP-8x37 則可到 1024 x 768, VP-25W7 最大是 640 x 480, VP-23W7 最大是 320 x 240, VP-23W7 的螢幕沒有 Touch 功能, VP-25W7 才有), 圖形元件的 Property 視窗會自動彈出來, 也可以點選 View > Component Properties 來開啓 Property 視窗.

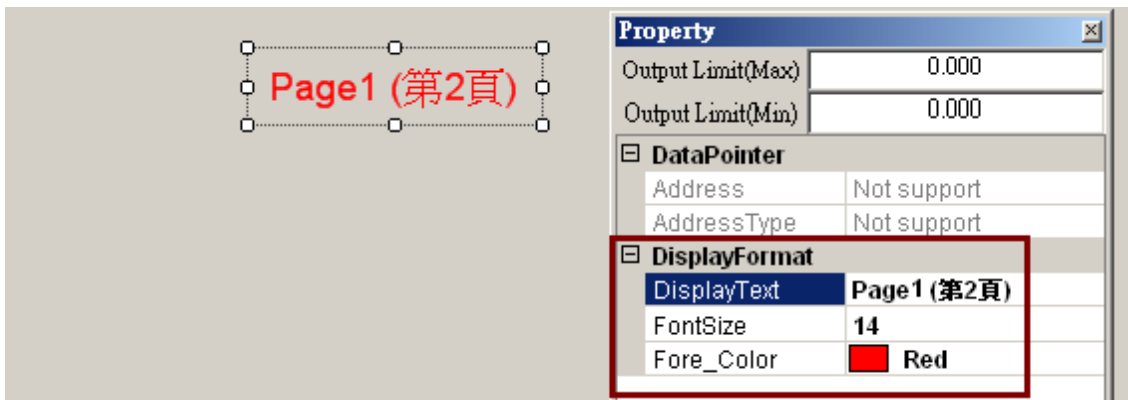


### 1.2.4.1: 新增 Label 元件

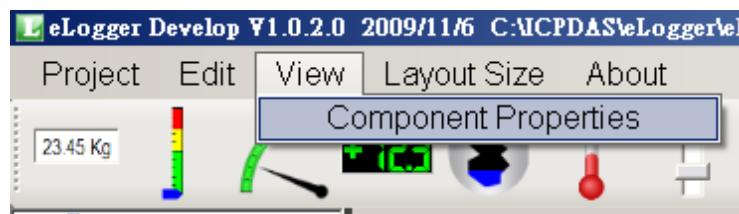
接下來本範例需在第 2 頁 (名稱 Page1) 新增一個 Label 元件。



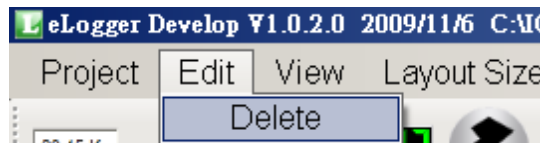
將該元件拖移至適當位置, 並在 Property 視窗內更改所要顯示的內容與 size 與顏色。



若 Property 視窗不小心被關閉, 可以點選 View > Component Properties 來開啓它。



若因編輯發生錯誤, 想刪除某元件時, 可以先點選該元件後, 再點選 Edit > Delete .

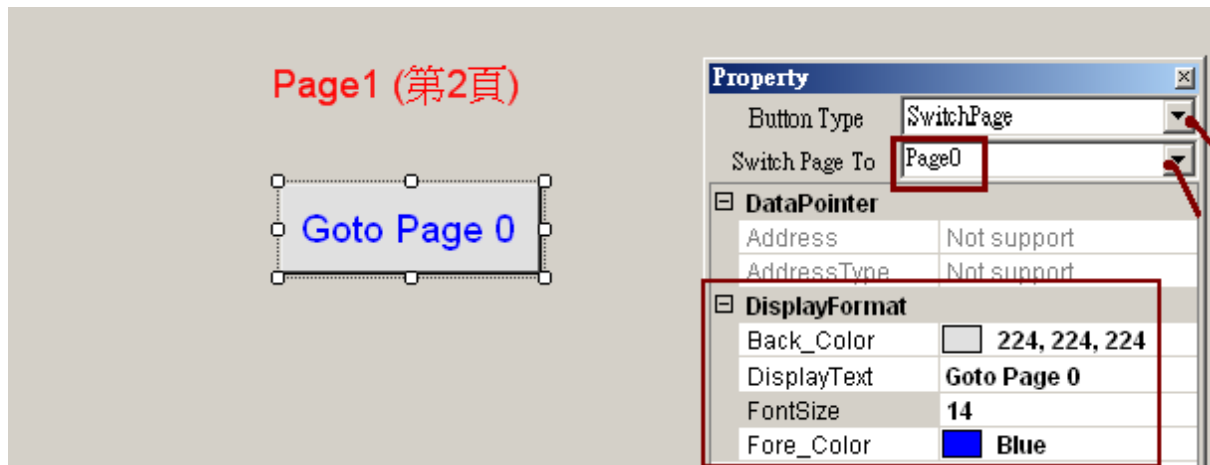


### 1.2.4.2: 新增 SwitchPage 元件

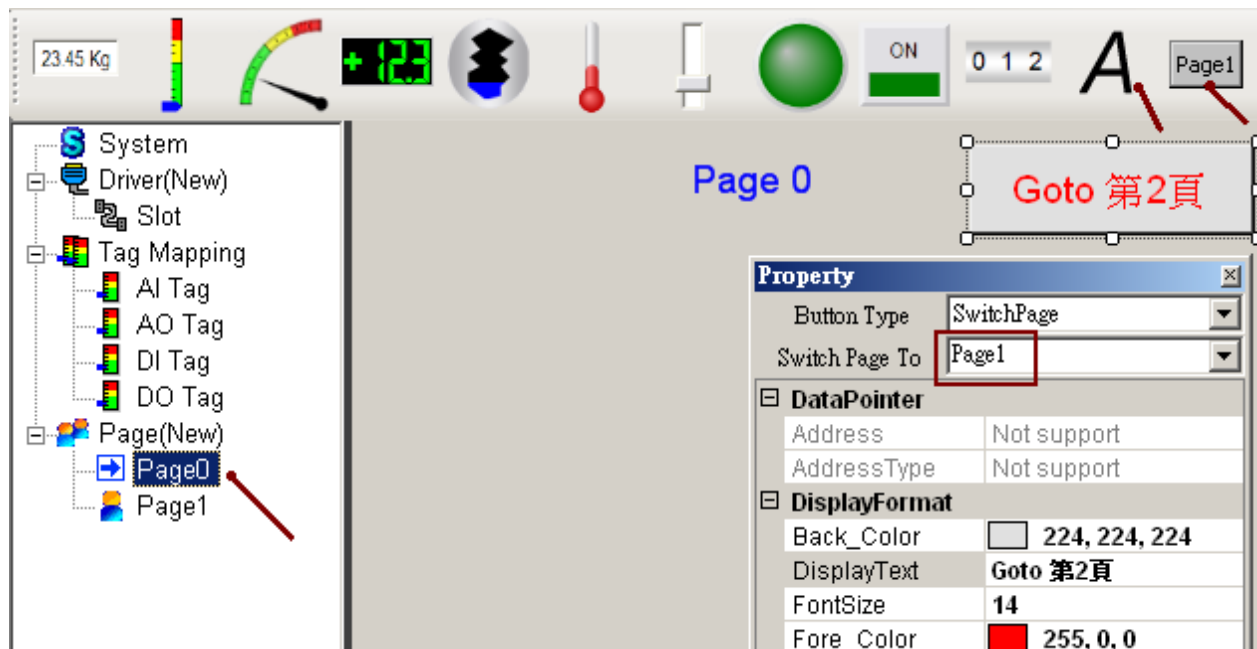
在 page 1 內點選 Button 來新增一個 SwitchPage 元件.



輸入 Button Type 為 “SwitchPage”，輸入 Switch Page To “為 “Page0”。之後輸入適當的 DisplayText, FontSize 與 Fore\_Color。



接下來請點選左方的 Page0, 採用同樣的方式在 Page 0 新增一個 Label 與 SwitchPage 元件

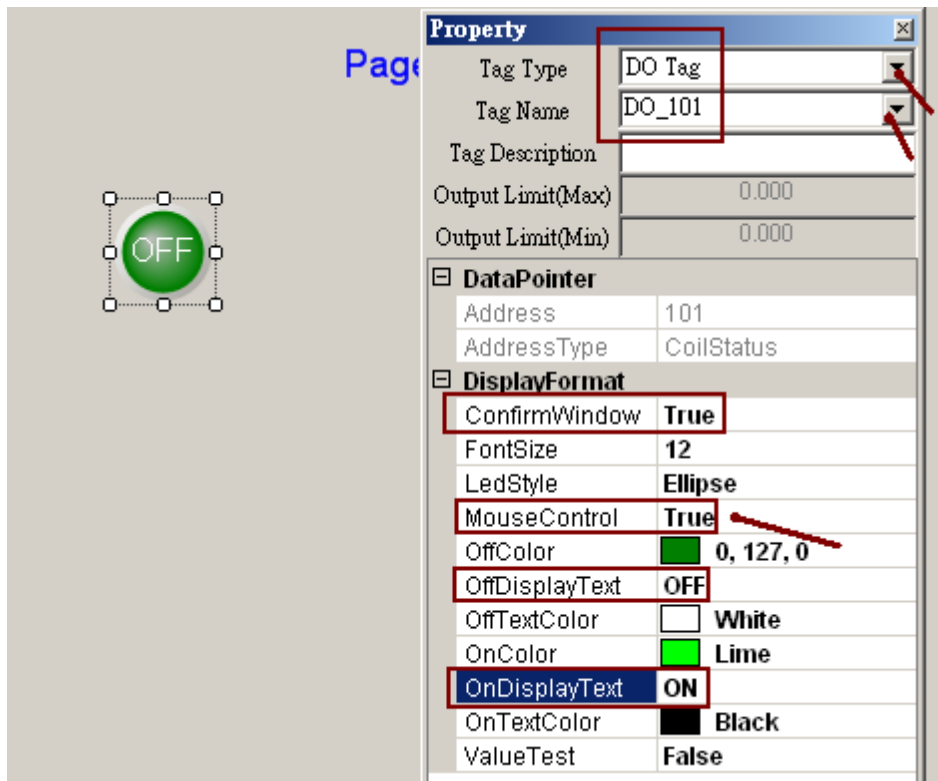


### 1.2.4.3: 新增 LED 元件

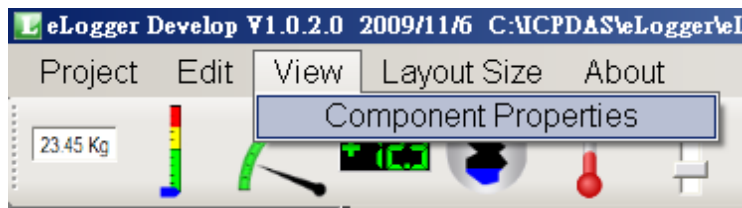
在 page0 內, 點選 LED 來新增一個 LED 元件.



本 LED 元件是要去控制 ISaGRAF 的 OUT\_101 變數 (在 eLogger 內的相對應的 DO Tag 為 DO\_101), 請選取 Tag Type 為 “DO Tag”, Tag Name 為 “DO\_101”, ConfirmWindow 為 “True” (表示要先詢問過後才可輸出), MouseControl 為 “True” (True 表示允許對該 Tag 輸出它的值, False 表示只能讀值, 不能去輸出), 之後設好其它適當的 Text 與顏色.

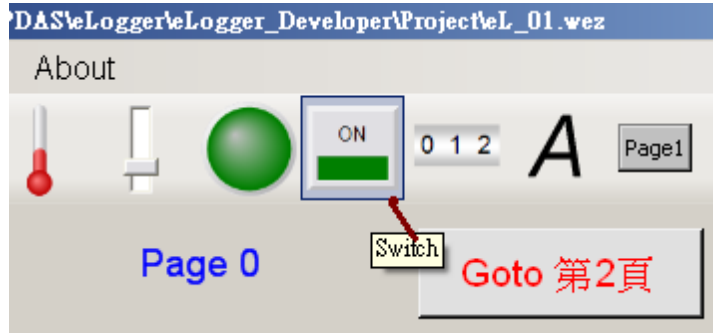


若 Property 視窗 不小心被關閉, 可以點選 View > Component Properties 來開啓它.

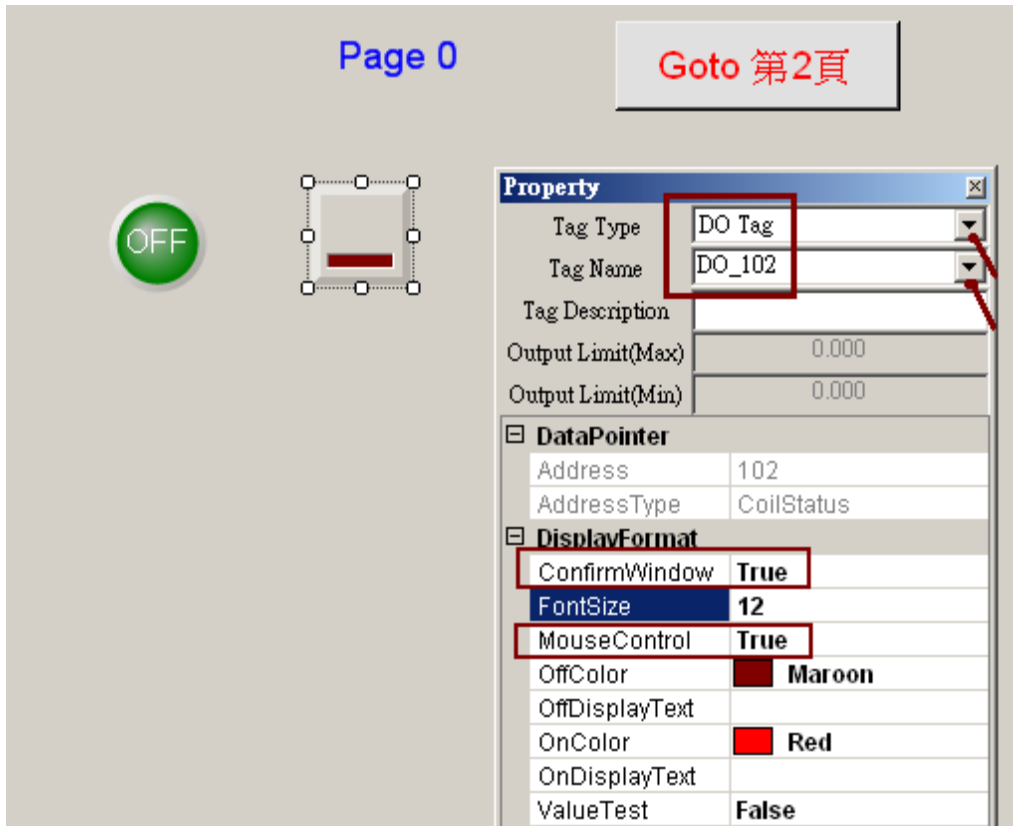


### 1.2.4.4: 新增 Switch 元件

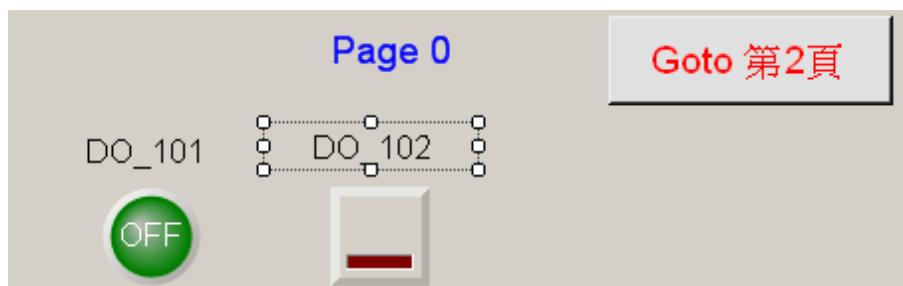
在 Page0 內, 點選 Switch 來新增一個 Switch 元件



設定 Tag Type 爲 “DO Tag”, Tag Name 爲 “DO\_102”, ConfirmWindow 爲 “True” (表示要先詢問過後才可輸出), MouseControl 爲 “True” (True 表示允許對該 Tag 輸出它的值, False 表示只能讀值, 不能去輸出), 之後設好其它適當的 Text 與顏色.



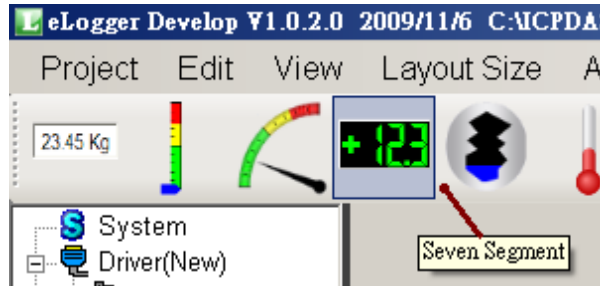
然後可再新增 2 個 Label 來說明剛才新增的 LED 與 switch 元件的用途 (參考 1.2.4.1 節).



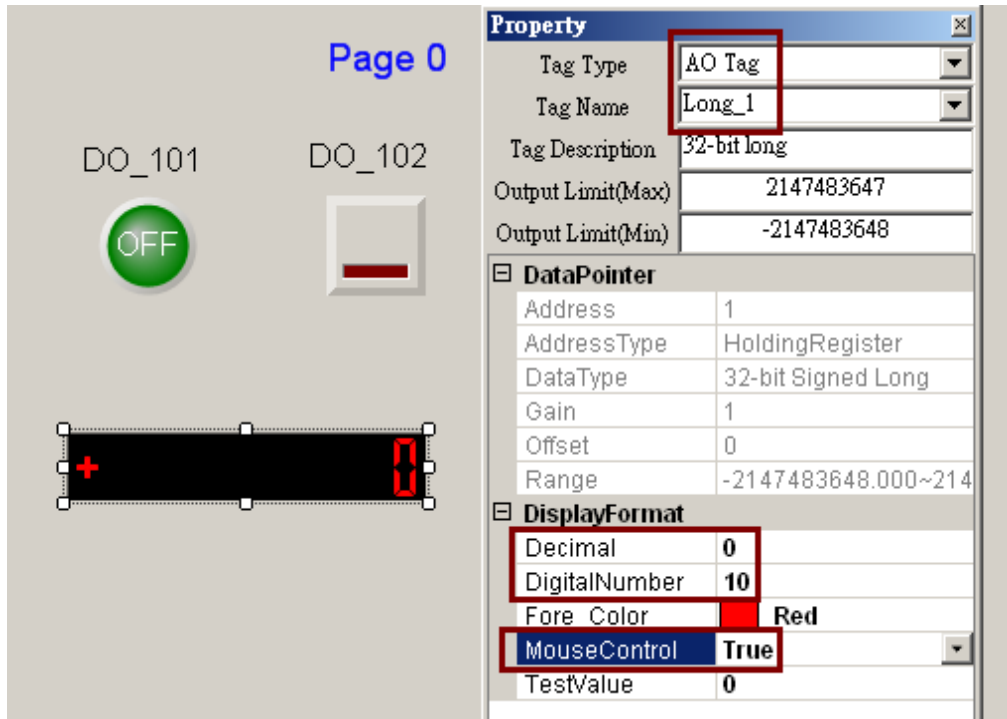


### 1.2.4.5: 新增 Seven Segment 元件

點選 Seven Segment 可以新增一個 Seven Segment 數字元件。



將 Tag type 設為 “AO\_Tag”, Tag Name 設為 “Long\_1”, 設定 MouseControl 為 “True” (True 表示允許對該 Tag 輸出它的值, False 表示只能讀值, 不能去輸出), 本 long\_1 資料型態為一個 32-bit long, 因此不需要小數位數, 所以將 Decimal 設為 0, 而 DigitalNumber 設為 10 (可以是 1 ~ 24), 若用戶的應用只能限定現場操作人員只能輸入某範圍內的值, 請另外設定好適當的 Output Limit(Max) 與 Output Limit (Min) .



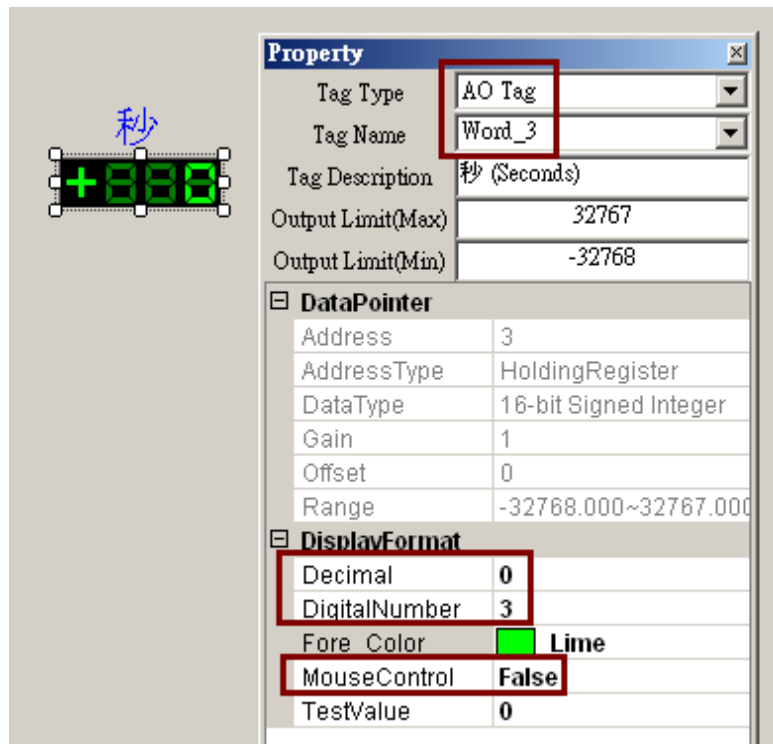
然後新增一個 Label 來說明 該元件的用途。



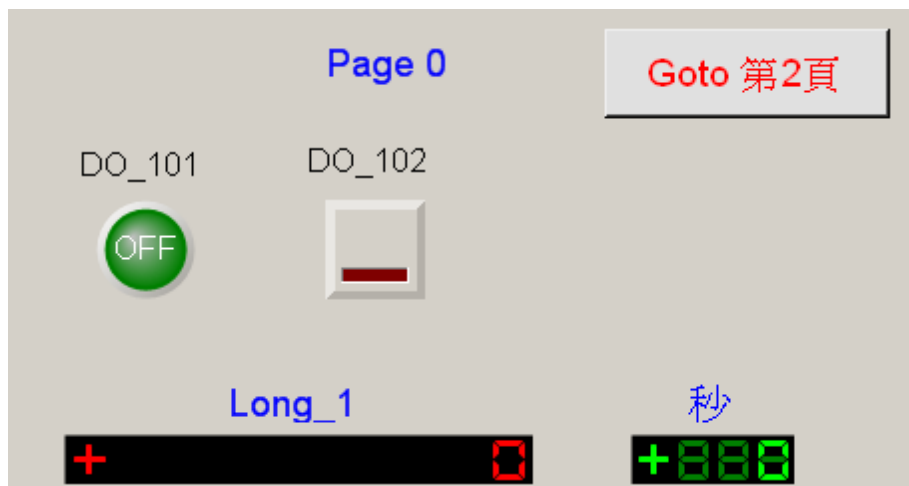
若因編輯發生錯誤, 想刪除某元件時, 可以先點選該元件後, 再點選 Edit > Delete .



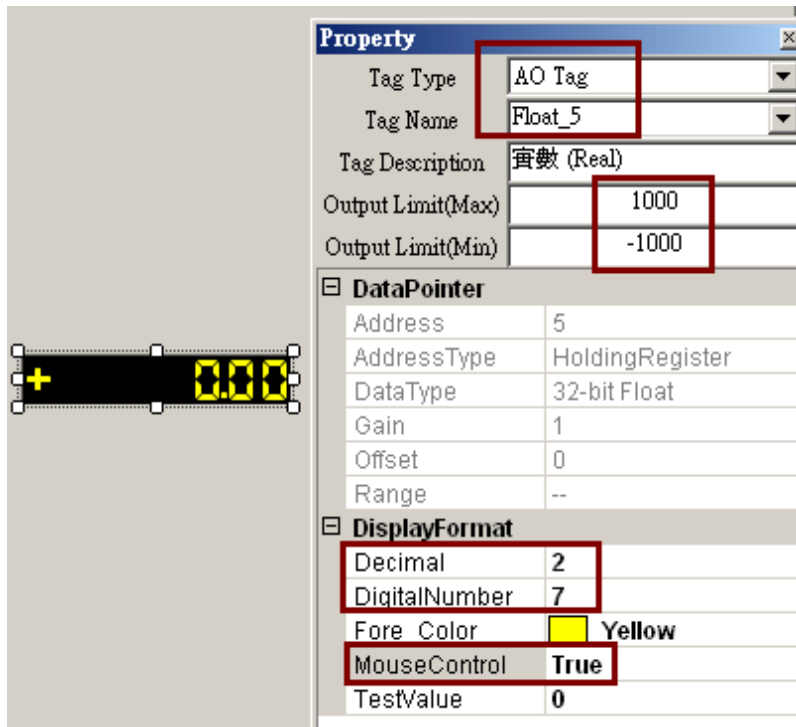
接下來使用類似的方法再新增一個 Seven Segment 元件, 設定 Tag Type 為 “AO Tag”, Tag Name 為 “Word\_3” (本例此值是讀取 WP-8xx7 控制器的時間的秒數值), 設定 MouseControl 為 “False” (True 表示允許對該 Tag 輸出它的值, False 表示只能讀值, 不能去輸出), Decimal 為 0, DigitalNumber 為 3.



然後得到以下 Page0 畫面.



接下來再以相同方式新增一個 Seven Segment 元件. 設定 Tag Type 為 “AO Tag”, Tag Name 為 “Float\_5”, 設定 MouseControl 設為 True (True 表示允許對該 Tag 輸出它的值, False 表示只能讀值, 不能去輸出), DigitalNumber 為 7, Decimal 為 2 (顯示 2 位小數), 限定操作者只能輸入 -1000 ~ +1000, 所以設定 Output Limit(Max)為 1000, Output Limit(Min)為 -1000 .

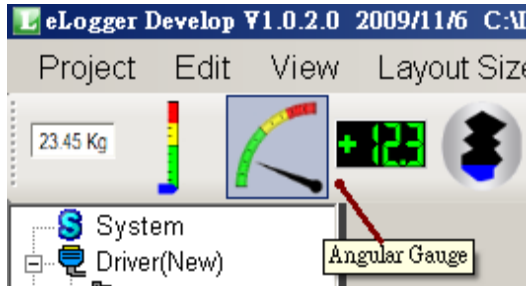


再新增一個 Label 來說明它的用途, 然後得到以下畫面.

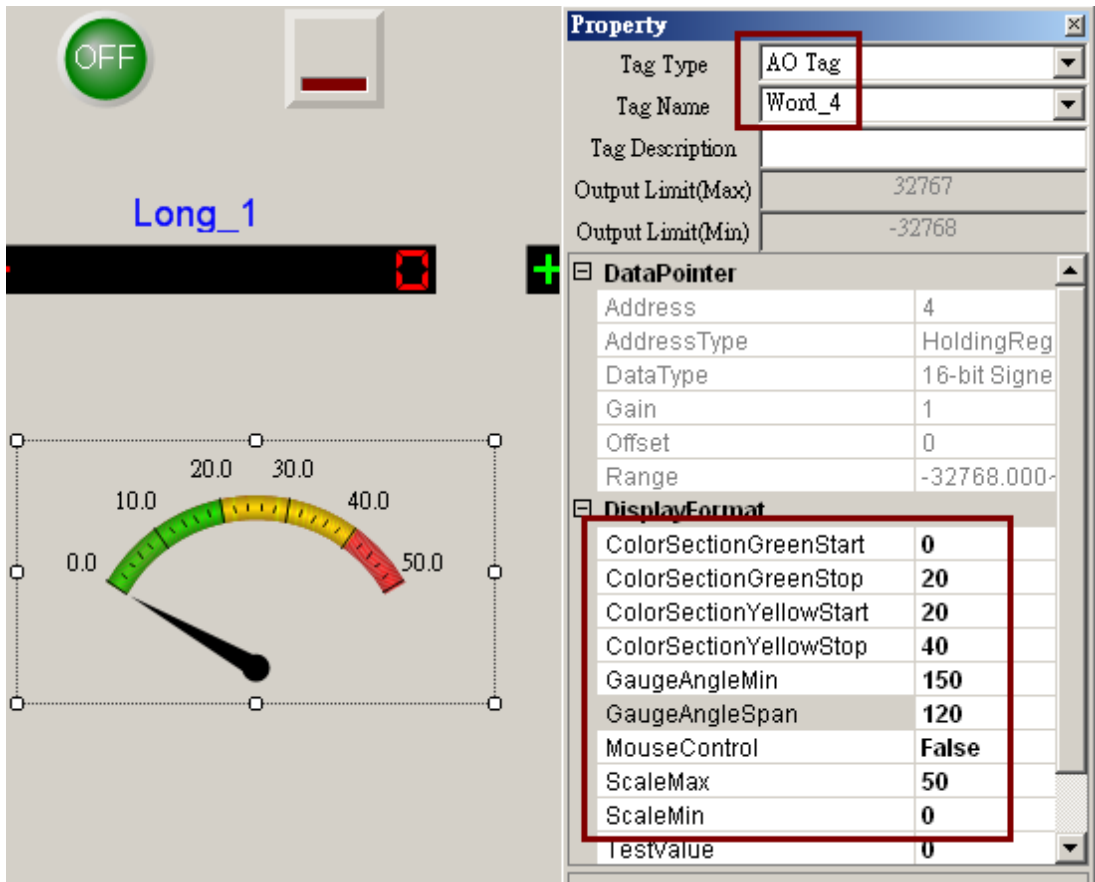


### 1.2.4.6: 新增 Angular Gauge 元件

點選 Angular Gauge 來新增一個 Angular Gauge 元件.

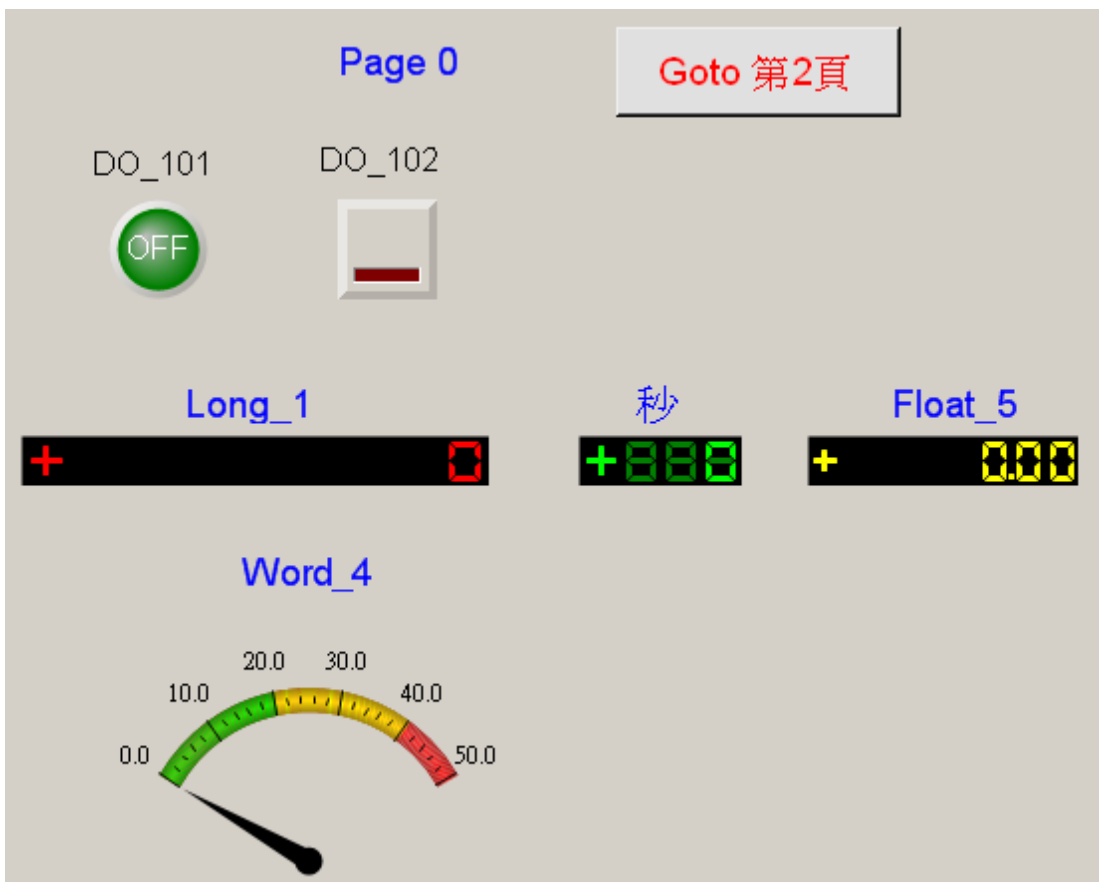


先設定 Tag Type 為 “AO Tag”, Tag Name 為 “Word\_4”, MouseControl 為 False, ScaleMin 為 0, Scale Max 為 50 (本範例 ISaGRAF 程式會將此值輸出在 0 ~ 50 範圍內), 然後設定 GaugeAngleMin 為 150 度 (即最小值是在 150 度的位置, 反時針方向), GaugeAngleSpan 設為 120 度 (即整個錶頭展開是 120 度), 設定 ColorSectionGreenStart 為 0, ColorSectionGreenStop 為 20, ColorYellowSectionStart 為 20, ColorYellowSectionStop 為 40.



然後再新增 一個 Label 來說明它. 最後得到以下畫面.

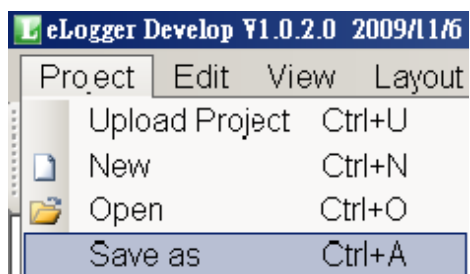
Page 0:



Page 1:

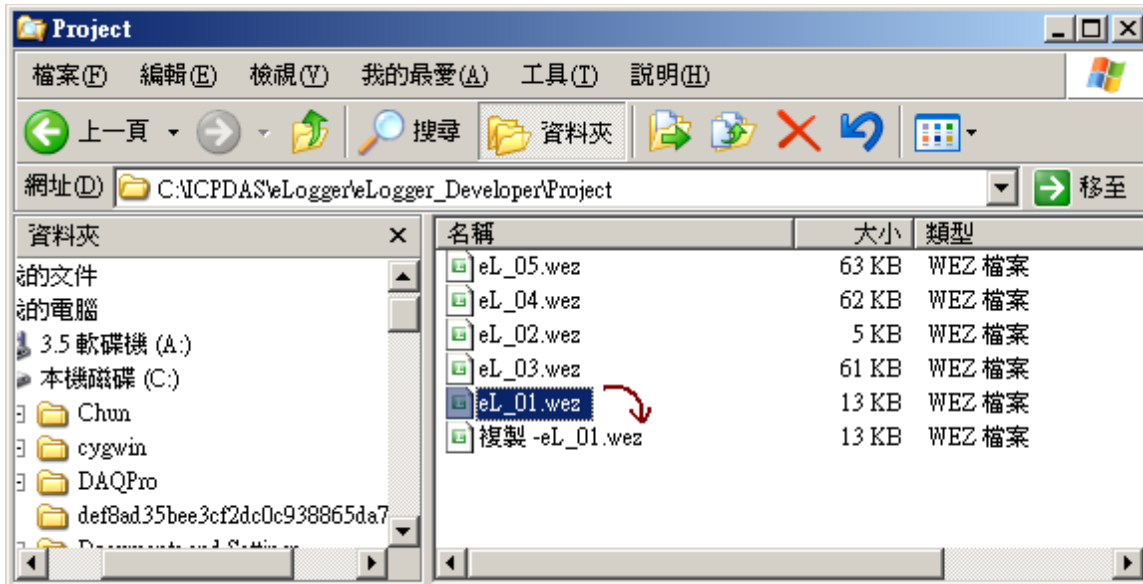


Save Project.



### 1.2.4.7: 完成 eLogger Project 後建議將 Project File 備份起來

當設計完一個 eLogger Project, 為了安全建議將該 Project File 複製一份, 完成的 eLogger Project file 會放在 C:\ICPDAS\Logger\Logger\_Developer\Project\ 內, 請將對應的 Project file “\*.wze” 複製一份, 以保安全.

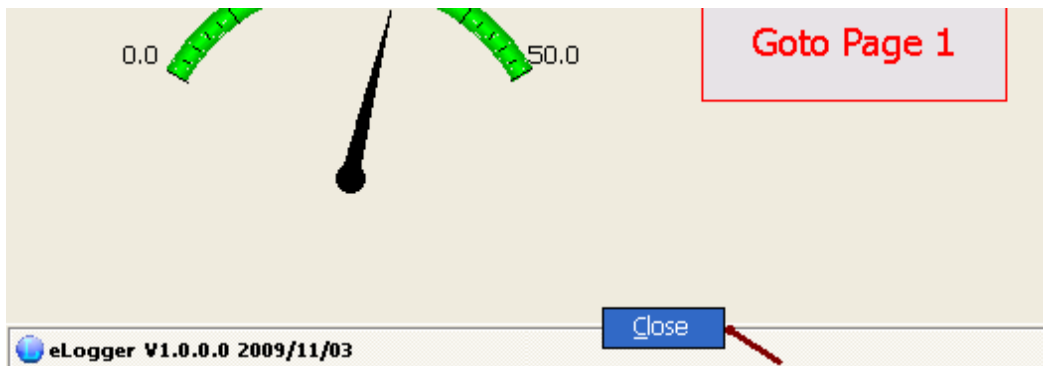


## 1.2.5: 將 eLogger 畫面丟到控制器內

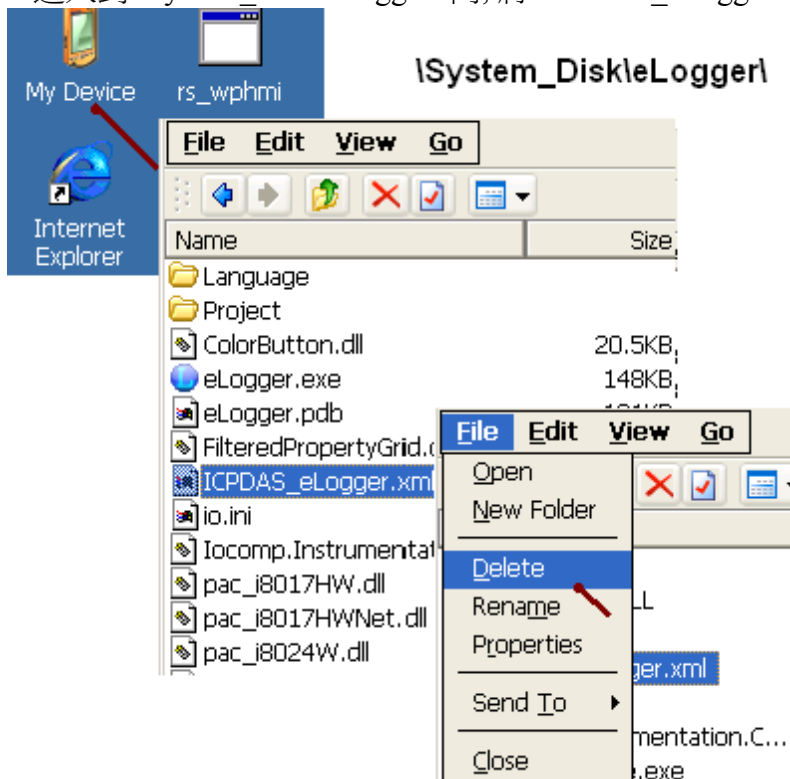
本範例硬體採用一台 WP-8447 + slot 1: I-87055W. 軟體則是採用 ISaGRAF SoftLogic 與 eLogger HMI. 請先將 ISaGRAF 程式先下載到 WP-8447 內 (參考本文件 1.1.7 節 或 1.1 節)

控制器內必需已經安裝好 eLogger RunTime 與做好相關設定 (參考本文件 1.2.1 節的說明) 然後將它開機, ISaGRAF 與 eLogger RunTime 會自動運行起來.

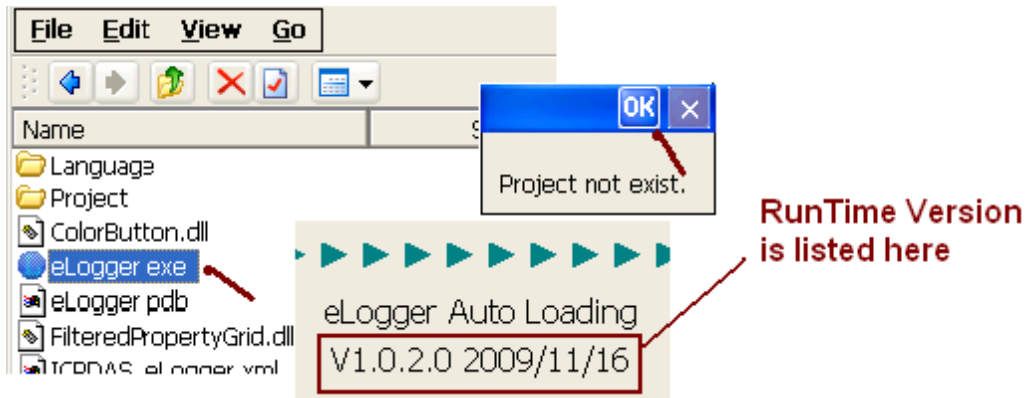
要下載 eLogger 畫面前, 得先將目前控制器正在運行的 eLogger 畫面 close 掉, 請用 Mouse 在下方 Taskbar 上的 “eLogger” 上按右鍵來 close (若你使用的是 Touch Monitor, 比如 VP-25W7, 那可能沒辦法按 Mouse 右鍵, 此時請將 WP-8xx7/VP-2xW7 控制器關機, 然後將 Rotary Switch 切換到 1 再開機, 然後再進行下一個步驟, 此時 eLogger 不會進行 Auto-Execution, 當 eLogger 畫面已經成功丟進控制器後, 記得再將 Rotart Switch 切換為 0, 重新開機).



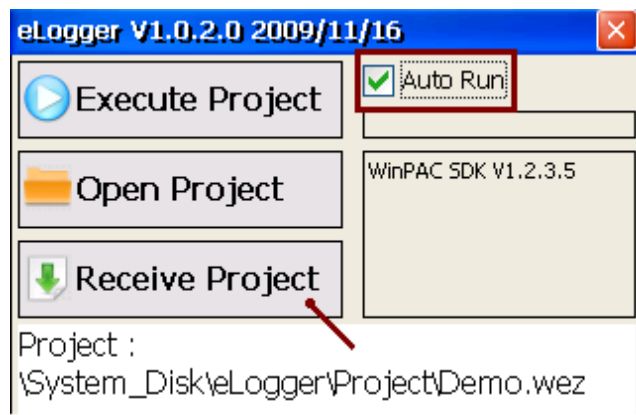
然後雙擊 My Device 進入到 \System\_Disk\eLogger\ 內, 將 ICPDAS\_eLogger.xml 這個 file 刪掉.



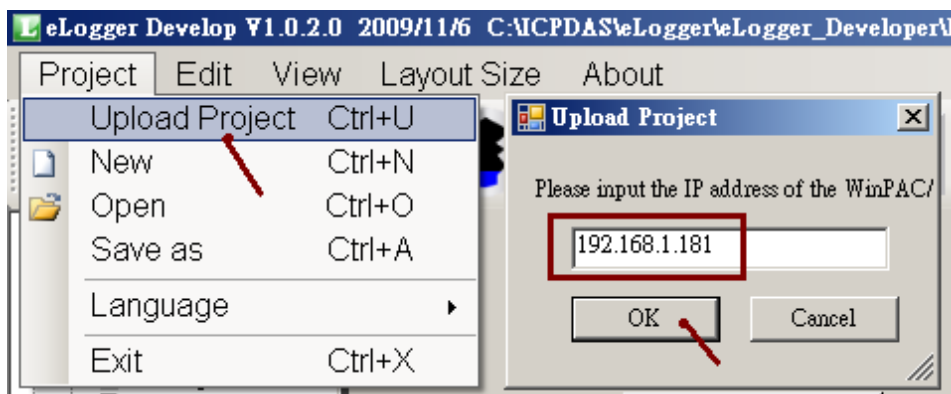
之後再將 eLogger.exe 運行起來 (用 Mouse 雙擊它), 此時會找不到 eLogger 畫面 (因為前一頁把 ICPDAS\_eLogger.xml 這個 file 刪掉了)



上方按下 OK 後會出現如下視窗. 那個 “Auto Run” 是在 1.2.1 節內已經勾選過的, 不要去改變它. 請按下 “Receive Project” 來等待 PC / eLogger Developer 那邊丟 eLogger 畫面過來.

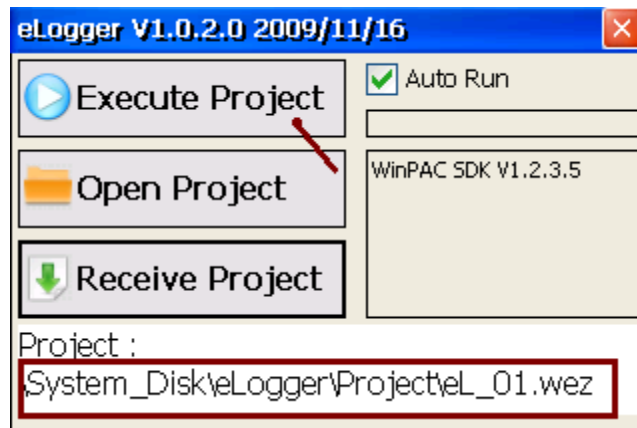


點選 PC 上的 eLogger Developer 的 Project > Upload Project, 輸入控制器的 IP, 然後按下 OK 來將 eLogger 畫面丟進控制器內.

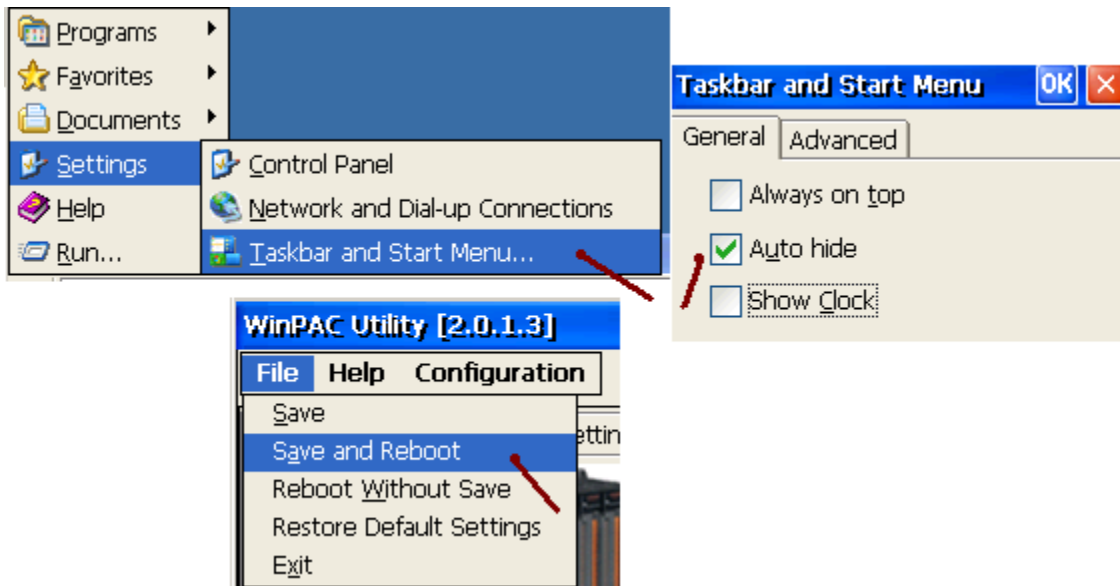




若成功, eLogger Run Time 那邊會顯示出剛才丟進去的 eLogger Project 名稱 (本例為 eL\_01.wez) 然後請按下 Execute Project 把新設計的 eLogger 畫面運作起來, 把控制器 reset 也可以讓新設計的 eLogger 畫面運作起來 (若 reset 控制器後 eLogger 畫面並沒有運作起來, 那表示 eLogger 並沒有設成 Auto-Execution 與 Auto-Run, 請參考本文件內的第 1.2.1 節的最後 2 個步驟來設定)。



當 Project 經過驗證與試機後都 OK 了, 要長時間實機運行時, 可以取消 Taskbar 的顯示. 如此 eLogger 畫面就會採用全螢幕模式顯示. 然後 運行 WinPAC Utility (或 ViewPAC Utility) 的 File > Save and Reboot 將此設定存入 Registry 內.

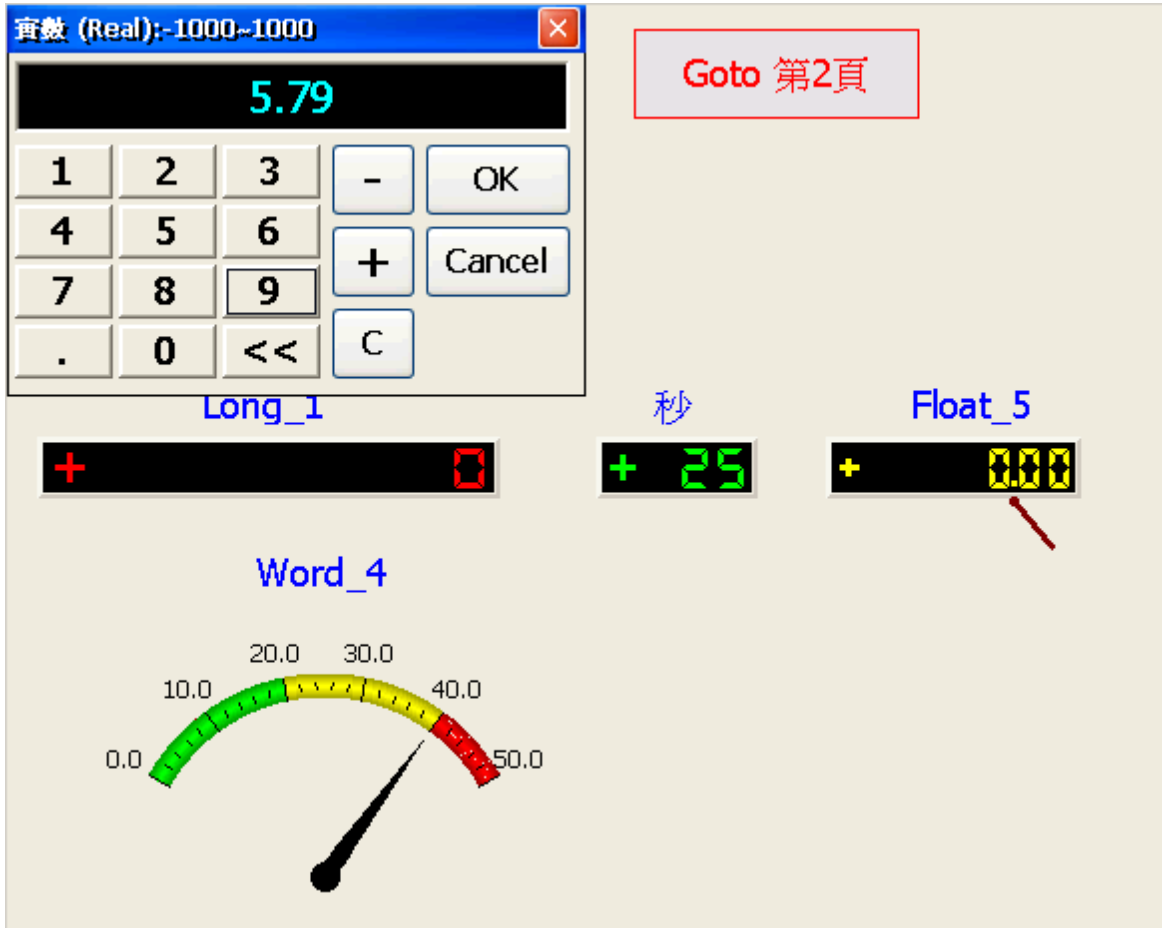


但是若仍是處於設計與測試階段, 建議還是顯示出 Taskbar 比較好.

當使用全螢幕顯示 eLogger 畫面後 (即不顯示 Taskbar), 若想再變更 eLogger 畫面, 請參考 本文件 1.3 節來處理.

## 1.2.6: 測試 eLogger HMI

當 WP-8xx7, VP-2xW7 (或日後推出的 XP-8xx7-CE6) 已經將 eLogger 畫面接收到並 Auto-Run 起來後, 本範例 “eL\_01” 會在 控制器的 VGA 上顯示如下, 你可以試著去點選比如 “Float\_5” 輸入一個數值來測試它, 並且用 ISaGRAF 連上該 控制器來觀察輸入的資料是否正確.

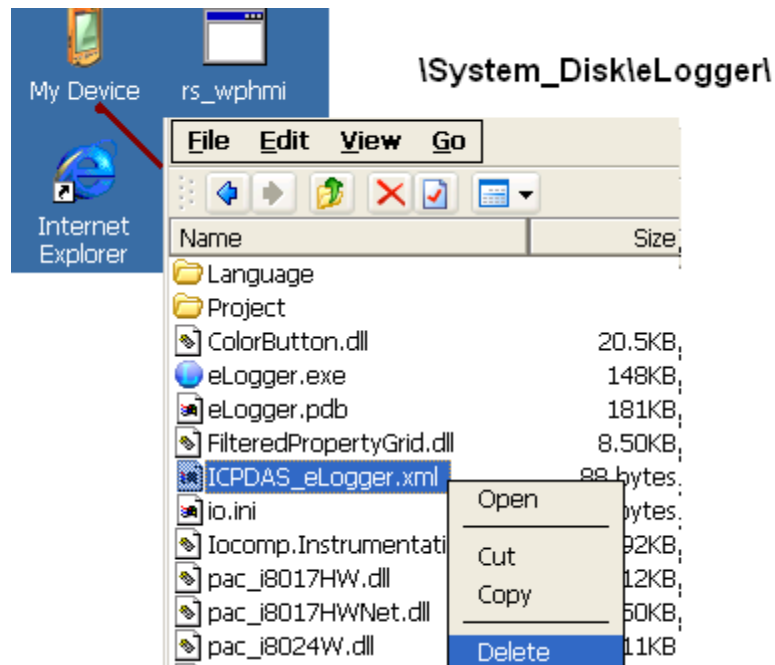


## 1.3: eLogger Run Time 問題解決

當 eLogger Run Time 安裝出現問題, 或採用 全螢幕顯示 eLogger 畫面 而無法 關閉 eLogger Run Time 時, 請將 控制器關機, 切換 面板上的 Rotary Switch 到 1 位置, 重新開機.

此時開機後 ISaGRAF 與 eLogger 都不會 自動 Run 起來.

接下來可以到 \System\_Disk\eLogger\ 內將 ICPDAS\_eLogger.xml 刪掉 (或者將它改名稱), 這樣 eLogger 自動 Run 起來後就會找不到 Project, 用戶就可再下載變更過的 Project. (若要重新安裝 eLogger Run Time, 請先刪除 \System\_Disk\eLogger\ 內的所有 file, 然後參考 1.2.1 節的說明來重新安裝它)



將 \System\_Disk\eLogger\ICPDAS\_eLogger.xml 刪除後, 關機, 再將 Rotary Switch 切到 0, 再開機. 之後 eLogger 畫面就不會自動 Run 起來 (因為找不到 eLogger project)

## 1.4: 其它 eLogger 進階功能

### 1.4.1: 設定 Gain 與 Offset 來做資料轉換

很多場合需要顯示的是 工程應用值 而非整數資料值, 比如 I-8017HW 這塊板卡, 當使用 Rang 設定為 8 時, 它在 ISaGRAF 內量測到的值是 (-32768 ~ +32767) 為一個 16-bit 有正負號的整數, 用來表示 (-10 ~ +10) Volt, 這時若想在 eLogger 畫面上顯示出 -10 ~ +10, 就必須設定 Gain 與 Offset.

**注意:** 當 Gain 是設為 1 與 Offset 是設為 0 時, 表示不進行資料轉換

轉換式為  $Y = [\text{Gain}] * X + [\text{Offset}]$

將 (X0 ~ X1) 轉換成 (Y0 ~ Y1) 那

$$\begin{aligned}\text{Gain} &= (Y1 - Y0) / (X1 - X0) \\ \text{Offset} &= (X1 * Y0 - X0 * Y1) / (X1 - X0)\end{aligned}$$

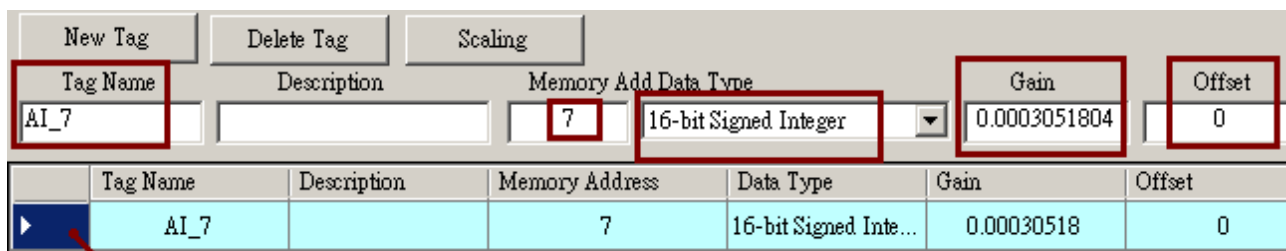
比如要將 (-32768 ~ 32767) 轉換成 (-10 ~ 10), 那

$$X0 = -32768, X1 = 32767, Y0 = -10, Y1 = 10$$

於是可算出

$$\begin{aligned}\text{Gain} &= 0.0003051804 \quad (\text{Gain 即使值很小, 也不能忽略}) \\ \text{Offset} &= 0.0001525902 \quad (\text{Offset 的值太小, 通常就會忽略改設為 0})\end{aligned}$$

所以若 AI\_7 這個 Tag 是讀取 I-8017HW 某個 Channel 的值, 那就可以設定 Gain 與 Offset 如下. (注意 Data\_Type 是 “16-bit Signed Integer”)



Tag Name	Description	Memory Address	Data Type	Gain	Offset
AI_7		7	16-bit Signed Integer	0.0003051804	0

所以可以如下頁在 eLogger 的畫面上顯示轉換後的值.

在 Property 視窗內 請先將 “TestValue” 設成轉換後的最大值來測試, 本例為 10.

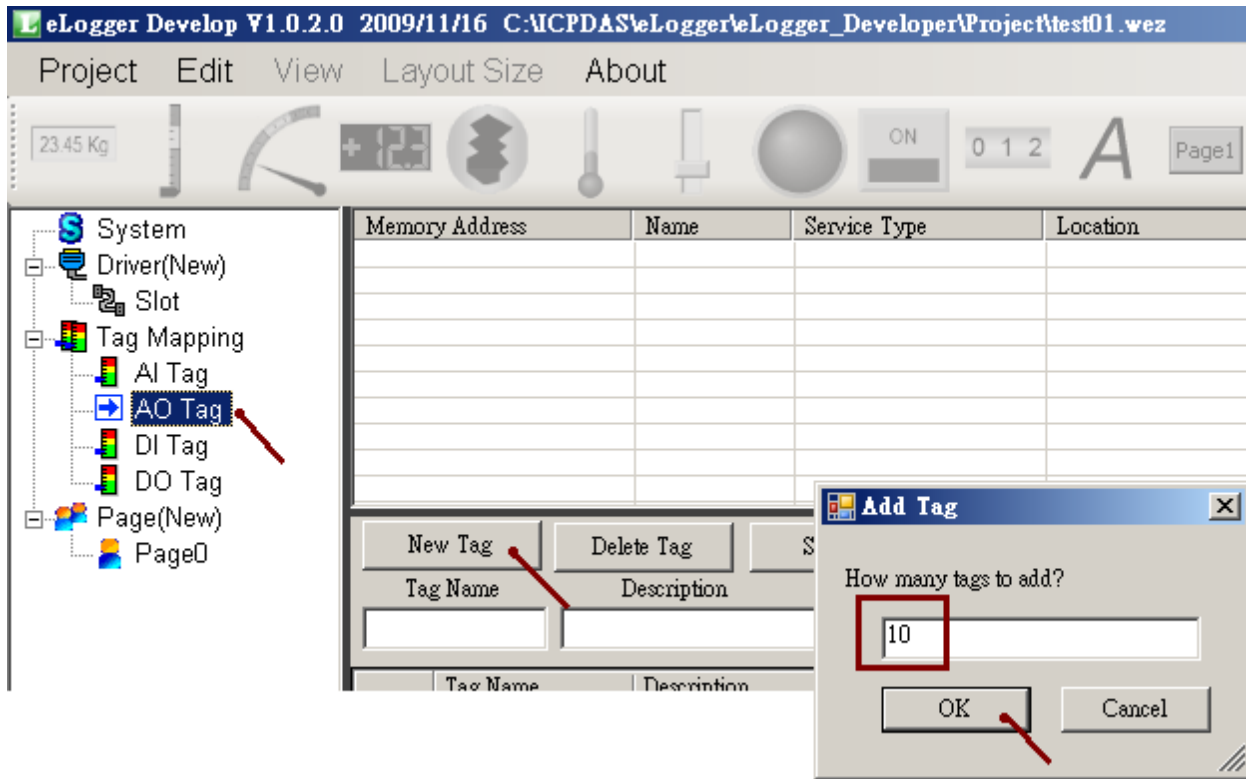
The screenshot shows the ISaGRAF software interface. On the left, there is a panel with a red button labeled "第2頁". Below it, a digital display labeled "Float\_5" shows the value "123.46" in yellow on a black background. Below that, another digital display shows the value "10.00" in green on a black background. On the right, the "Property" window is open, showing the configuration for an AO Tag named "AI\_7". The "Tag Type" is "AO Tag" and the "Tag Name" is "AI\_7". The "Output Limit(Max)" is "10.00" and the "Output Limit(Min)" is "-10.0". The "DataPointer" section shows "Address" as "7", "AddressType" as "HoldingRegister", "DataType" as "16-bit Signed Integer", "Gain" as "0.00030518", "Offset" as "0", and "Range" as "-10.000~10.000". The "DisplayFormat" section shows "Decimal" as "2", "DigitalNumber" as "4", "Fore\_Color" as "0, 255, 0", "MouseControl" as "False", and "TestValue" as "10". The "TestValue" section at the bottom of the property window contains the text "TestValue" and "Input the value to test the display."

Property	
Tag Type	AO Tag
Tag Name	AI_7
Tag Description	
Output Limit(Max)	10.00
Output Limit(Min)	-10.0
DataPointer	
Address	7
AddressType	HoldingRegister
DataType	16-bit Signed Integer
Gain	0.00030518
Offset	0
Range	-10.000~10.000
DisplayFormat	
Decimal	2
DigitalNumber	4
Fore_Color	0, 255, 0
MouseControl	False
TestValue	10

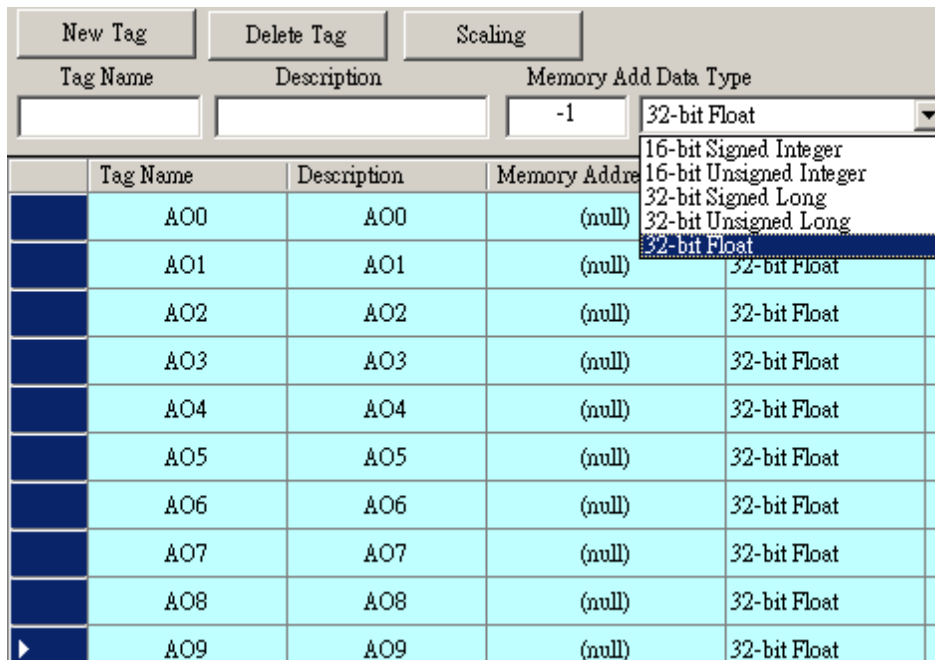
**TestValue**  
Input the value to test the display.

## 1.4.2: 宣告多個名稱類似與格式相同的 eLogger Tag

在 eLogger 內可一次新增多個 Tag, 然後再一次宣告為類似的名稱與相同的格式, 如下。



先使用 Mouse 搭配 “Shift” 鍵來選取這些 Tag, 然後就可設定為相同的格式。



也可設定為類似的名稱,與類似的 “Description”。

New Tag		Delete Tag		Scaling	
Tag Name	Description	Memory Address	Add	Data Type	
VAL		-1		32-bit Float	
Tag Name	Description	Memory Address		Data Type	
VAL_0	AO0	(null)		32-bit Float	
VAL_1	AO1	(null)		32-bit Float	
VAL_2	AO2	(null)		32-bit Float	
VAL_3	AO3	(null)		32-bit Float	
VAL_4	AO4	(null)		32-bit Float	
VAL_5	AO5	(null)		32-bit Float	
VAL_6	AO6	(null)		32-bit Float	
VAL_7	AO7	(null)		32-bit Float	
VAL_8	AO8	(null)		32-bit Float	
▶ VAL_9	AO9	(null)		32-bit Float	

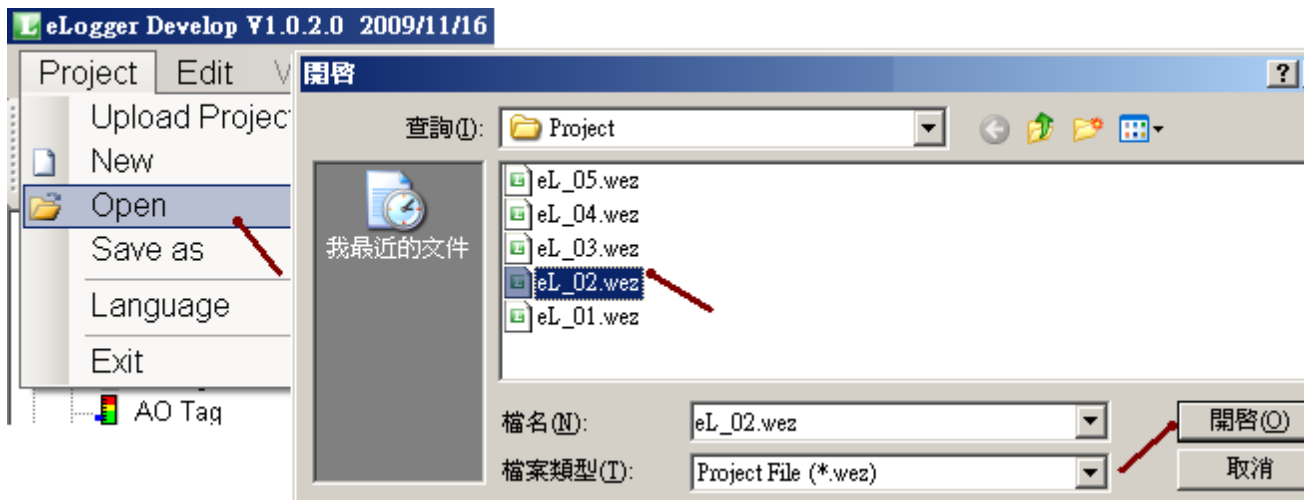
New Tag		Delete Tag		Scaling	
Tag Name	Description	Memory Address	Add	Data Type	
VAL	REAL vale	-1		32-bit Float	
Tag Name	Description	Memory Address		Data Type	
VAL_0	REAL vale_0	(null)		32-bit Float	
VAL_1	REAL vale_1	(null)		32-bit Float	
VAL_2	REAL vale_2	(null)		32-bit Float	
VAL_3	REAL vale_3	(null)		32-bit Float	
VAL_4	REAL vale_4	(null)		32-bit Float	
VAL_5	REAL vale_5	(null)		32-bit Float	
VAL_6	REAL vale_6	(null)		32-bit Float	
VAL_7	REAL vale_7	(null)		32-bit Float	
VAL_8	REAL vale_8	(null)		32-bit Float	
▶ VAL_9	REAL vale_9	(null)		32-bit Float	

## 1.5: 一些實用的 eLogger + ISaGRAF 範例程式

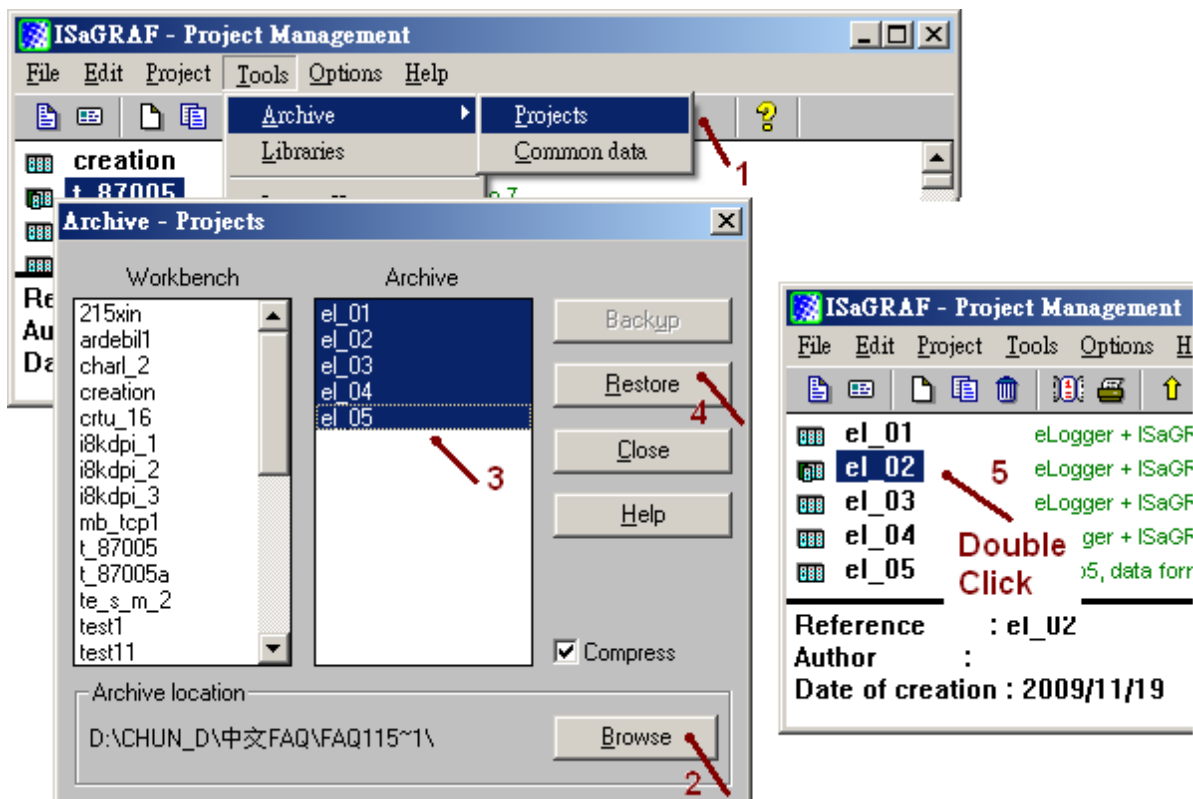
本節所介紹的 ISaGRAF 與 eLogger 範例程式可由以下網址取得。

[www.icpdas.com](http://www.icpdas.com) > FAQ > Software > ISaGRAF > 中文 > FAQ-115

若 user 想直接開啓 eLogger 範例程式來參考, 請將從 FAQ-115 上取得的 “eL\_01.wez” ~ “eL\_06.wez” 複製到 PC 的 C:\ICPDAS\eLogger\eLogger\_Developer\Project\ 內, 然後運行 eLogger Developer, 點選 Project > Open 來開啓它。



若 user 想直接開啓 ISaGRAF 範例程式來參考, 請運行 ISaGRAF 將 “eL\_01.pia” ~ “eL\_06pia” 回存到 ISaGRAF 內. 之後用 Mouse 雙擊來開啓 project.





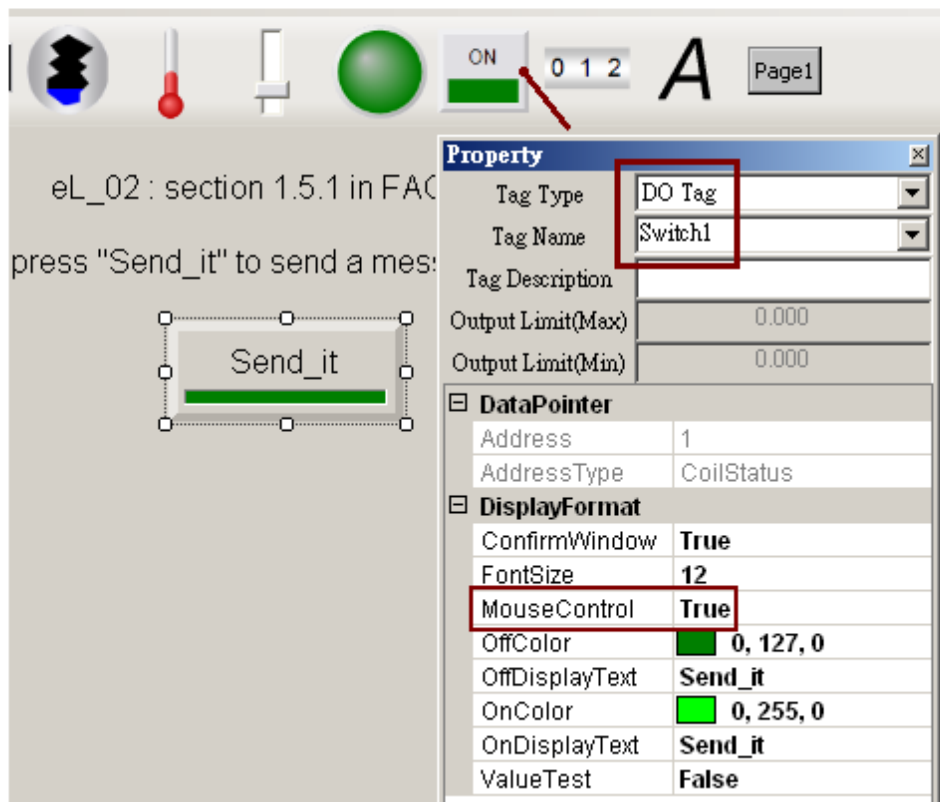
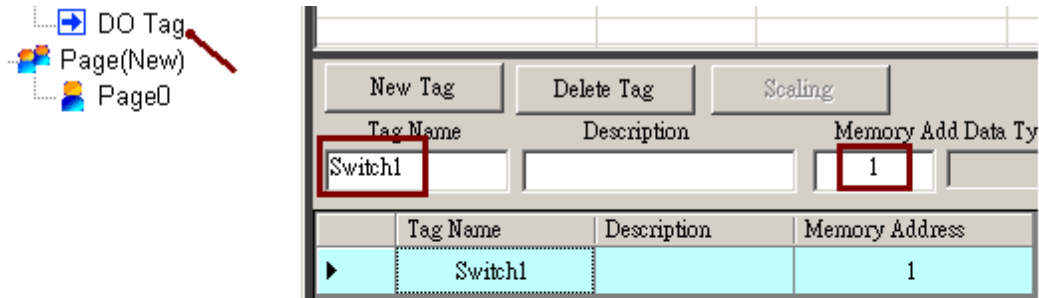
## 1.5.1: 如何在 eLogger+ ISaGRAF 內使用一個可瞬間 ON 之後再 OFF 的控制按鈕

ISaGRAF 範例程式爲： eL\_02.pia

eLogger 範例程式爲： eL\_02.wez

本範例當 Switch 元件被按下並設爲 ON 後, ISaGRAF 程式會從 WP-8447 的 COM1: RS-232 送出目前的控制器日期與時間一次, 可以在 PC 上 Run Hyper-Terminal 開啓一個 RS-232 port (9600,8,N,1) 來接收此日期與時間資料. 之後 eLogger 上該 Switch 元件會自動還原回 OFF 狀態.

Elogger Developer:

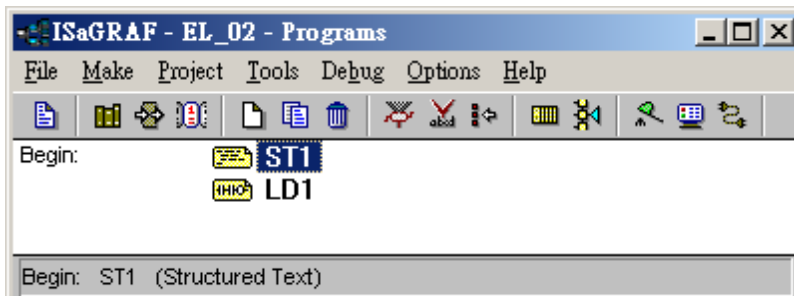


ISaGRAF :

變數宣告:

Name	Type	Attribution	NetWork Addr.	說明
Switch1	Boolean	Internal	1	用來跟 eLogger Switch 溝通
TMP	Boolean	Internal	0	
INIT	Boolean	Internal	0	宣告初值為 TRUE
STR1	Message	Internal	0	宣告 Max. Length 為 64
Year1	Integer	Internal	0	取得 年
Month1	Integer	Internal	0	取得 月
Day1	Integer	Internal	0	取得 日
WeekDay1	Integer	Internal	0	取得 星期幾
Hour1	Integer	Internal	0	取得 時
Minute1	Integer	Internal	0	取得 分
Second1	Integer	Internal	0	取得 秒

Project :

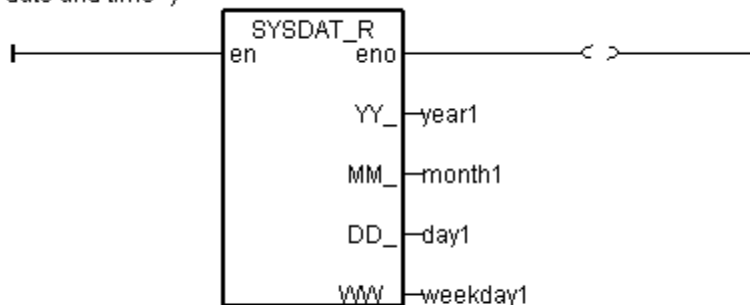


ST1 程式:

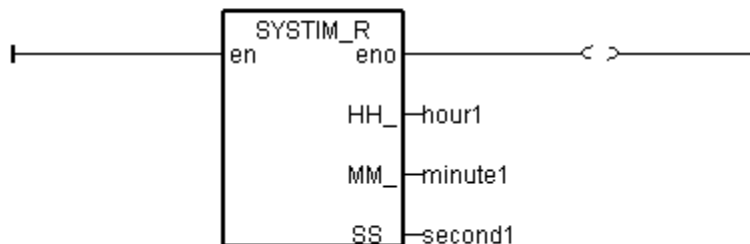
```
(* 在第一個 Scan 內開啓 COM1 串口爲 9600,8,N,1,  
"INIT" 必需宣告其初值爲 True *)  
if INIT then  
  INIT := False ; (* 只在第一個 Scan 內 run 一次 *)  
  TMP := COMOPEN( 1 , 9600 , 8 , 0 , 1 ) ;  
end_if ;  
  
(* 當 eLogger HMI 上將 "Switch1" 設爲 True , 從 COM1 送出一個字串 *)  
if switch1 then  
  
  (* 轉換日期與時間爲字串 , 比如 'Feb/18/2010,13:25:45' *)  
  str1 := time_str( year1 , month1 , day1 , weekday1 , hour1 , minute1 , second1 ) ;  
  str1 := str1 + MSG('$0D$0A') ; (* 在字串尾部加上 <CR><LF> *)  
  
  (* 送出該字串到 COM1 *)  
  TMP := COMSTR_W( 1 , str1 ) ;  
  
  switch1 := False ; (* 完成時,將 "Switch1" 設回 False *)  
  
end_if ;
```

LD1 程式 :

(\* get PAC 's date and time \*)



(\* \*)

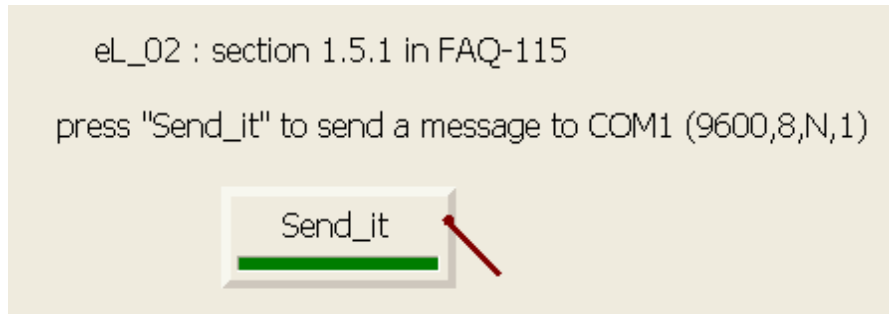


如何測試：

請準備一條 RS-232 Cable 連接 WP-8xx7 的 COM1 與 PC 的一個 RS-232 port (請參考 WP-8xx7 快速上手手冊的附錄 A.5), 然後將 WP-8xx7 開機, PC 則是運行 Hyper-Terminal 開啓 RS-232 port 爲 9600, 8, N, 1.

WP-8xx7 - eLogger HMI :

按下 Send\_it 按鈕並設爲 ON 來從 COM1 送出一個字串.



PC – Hyper Terminal (9600, 8, N, 1) :



## 1.5.2: 從一個 file 讀出或存入應用參數, File 格式為實數 20 筆, 每列 10 筆, 共 2 列

ISaGRAF 範例程式為: eL\_03.pia

eLogger 範例程式為: eL\_03.wez

另外有個 eL\_04 範例程式與 eL\_03 類似, 但資料格式為 整數.

ISaGRAF 範例程式為: eL\_04.pia

eLogger 範例程式為: eL\_04.wez

另外有個 eL\_05 範例程式與 eL\_03 類似, 資料格式也是實數, 但可以儲存多組應用參數到不同的 file 內

ISaGRAF 範例程式為: eL\_05.pia

eLogger 範例程式為: eL\_05.wez

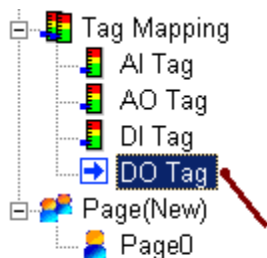
本範例可以在 WP-8xx7 上的 eLogger HMI 畫面來輸入 20 個應用參數值, 這些值會被存放在 \System\_Disk\ 內的 Working\_Real.txt 檔. 該檔為一文字檔, 共有 2 列資料, 每列資料各存放 10 個實數值, 類似如下:

23, 65.9, 0.12, 5.87, 88.2, 0.34, 8.5, -2.08, 4.08, 5.32  
2, -7, 6666.8, 456.07, 1.01, 5, 6, 7, 8, 9

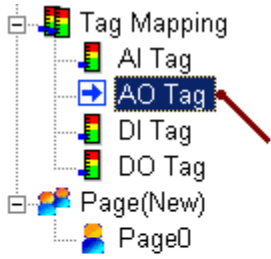
eLogger HMI 畫面上會使用 20 個 Seven Segment 來顯示 目前工作中的 20 個應用參數值 (Address 為 101, 103, 105, ..., 139, MouseContrl 要設為 FALSE, 只能讀, 不能寫入)於上方. 另外下方則是 使用 20 個 Seven Segment 來顯示與輸入 新修改的值 (Address 為 201, 203, 205, ..., 239, MouseContrl 要設為 TRUE, 可以讀, 也可以寫入).

另外還有 3 個 Switch, Address 分別是 1, 2, 3, 用來 從 檔案 RE\_LOAD 資料 與 RE\_Store 資料到 檔案 與 顯示目前資料是否 OK.

Elogger Developer:



Tag Name	Description	Memory Address
RE_LOAD	Load data from file	1
RE_Store	Store new data to ...	2
Read_Data_OK	TRUE: Data is rea...	3



Tag Name	Description	Memory Address	Data Type
Working_VAL_0		101	32-bit Float
Working_VAL_1		103	32-bit Float
Working_VAL_2		105	32-bit Float
...			
Working_VAL_18		137	32-bit Float
Working_VAL_19		139	32-bit Float
TMP_VAL_0		201	32-bit Float
TMP_VAL_1		203	32-bit Float
TMP_VAL_2		205	32-bit Float
...			
TMP_VAL_18		237	32-bit Float
TMP_VAL_19		239	32-bit Float

**Property**

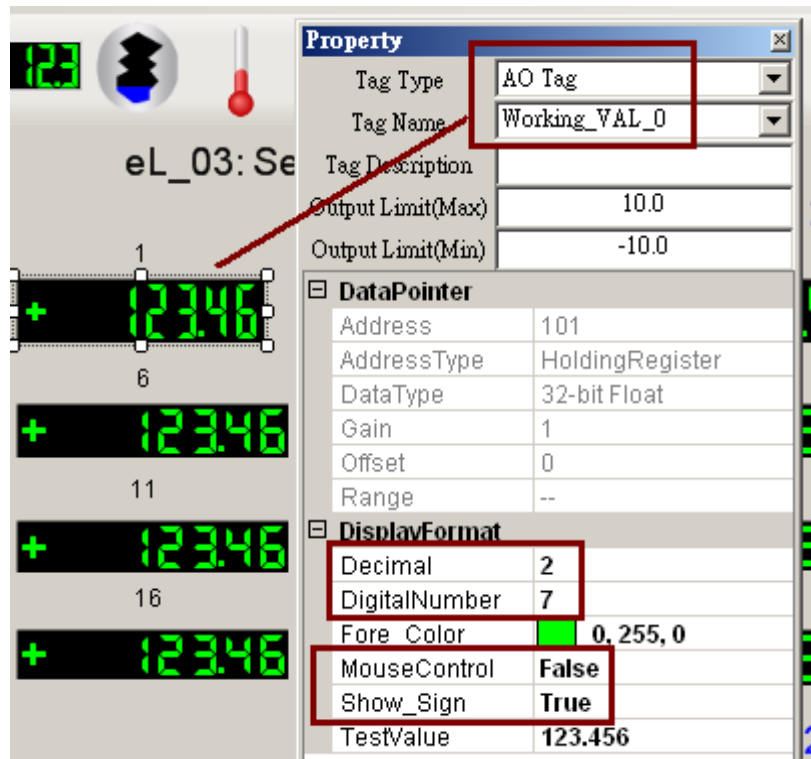
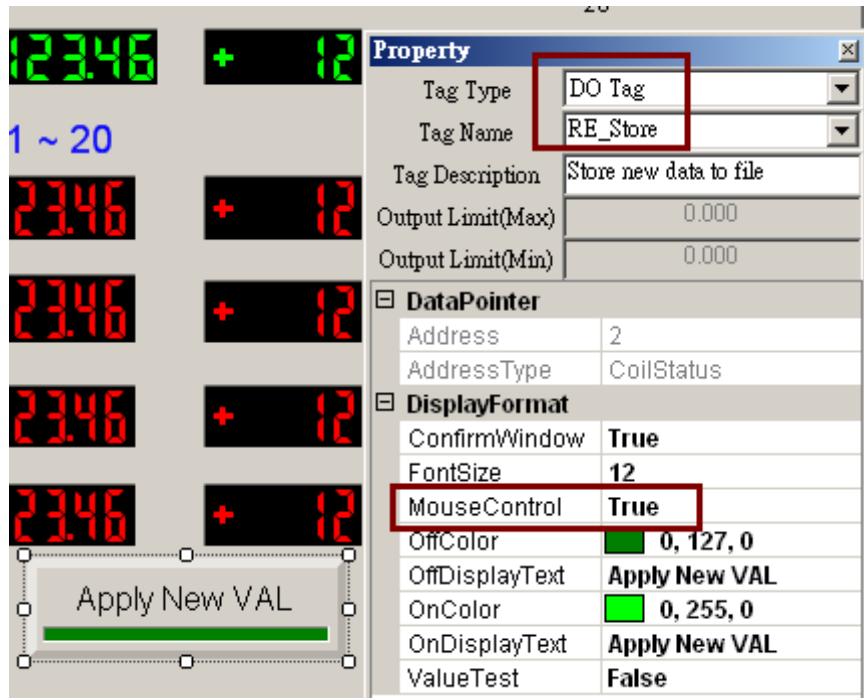
Tag Type	DO Tag
Tag Name	Read_Data_OK
Tag Description	TRUE: Data is ready
Output Limit(Max)	0.000
Output Limit(Min)	0.000
<b>DataPointer</b>	
Address	3
AddressType	CoilStatus
<b>DisplayFormat</b>	
ConfirmWindow	True
FontSize	14
LedStyle	Ellipse
MouseControl	False
OffColor	255, 255, 255
OffDisplayText	Data Error
OffTextColor	255, 0, 0
OnColor	255, 255, 255
OnDisplayText	Data OK
OnTextColor	0, 0, 255
ValueTest	False

The image shows a vertical stack of five digital displays. The top display shows 'Data Error' in red text. The middle three displays show '12346' in green text. The bottom display shows '12346' in red text. Each display is flanked by '+' signs.

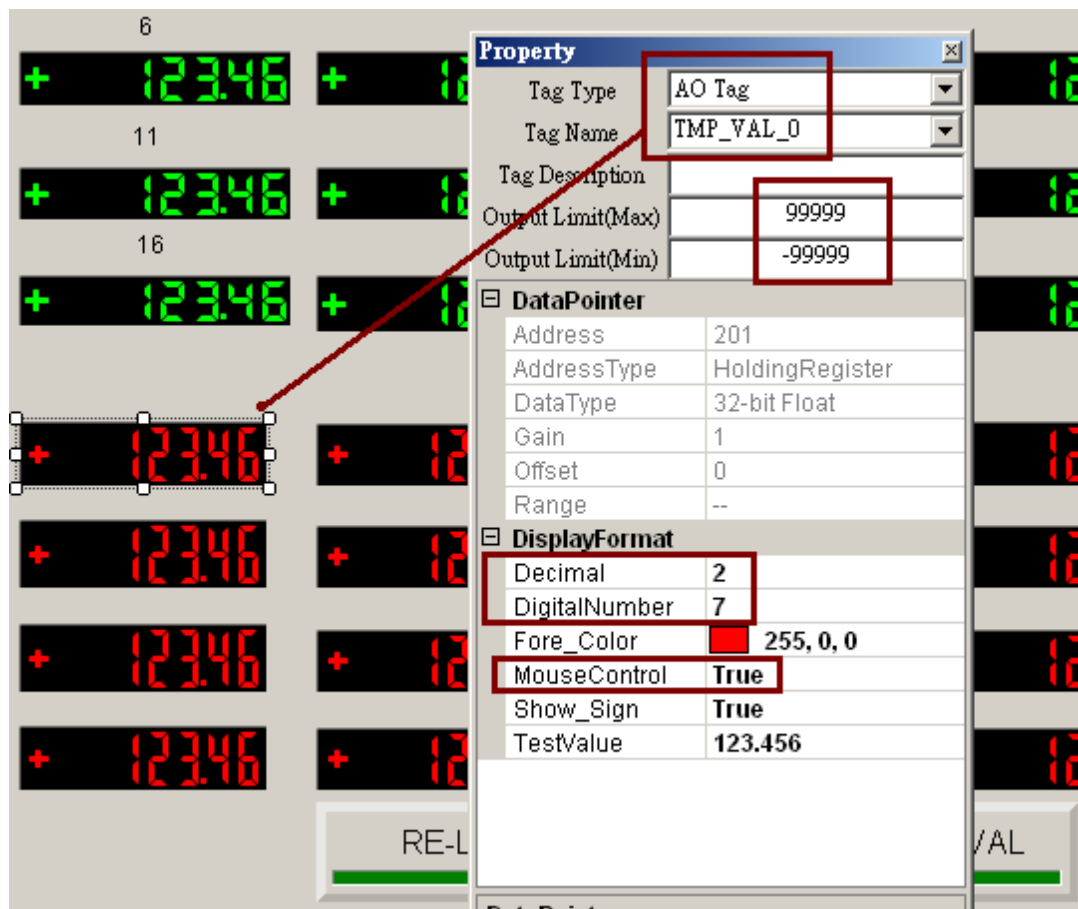
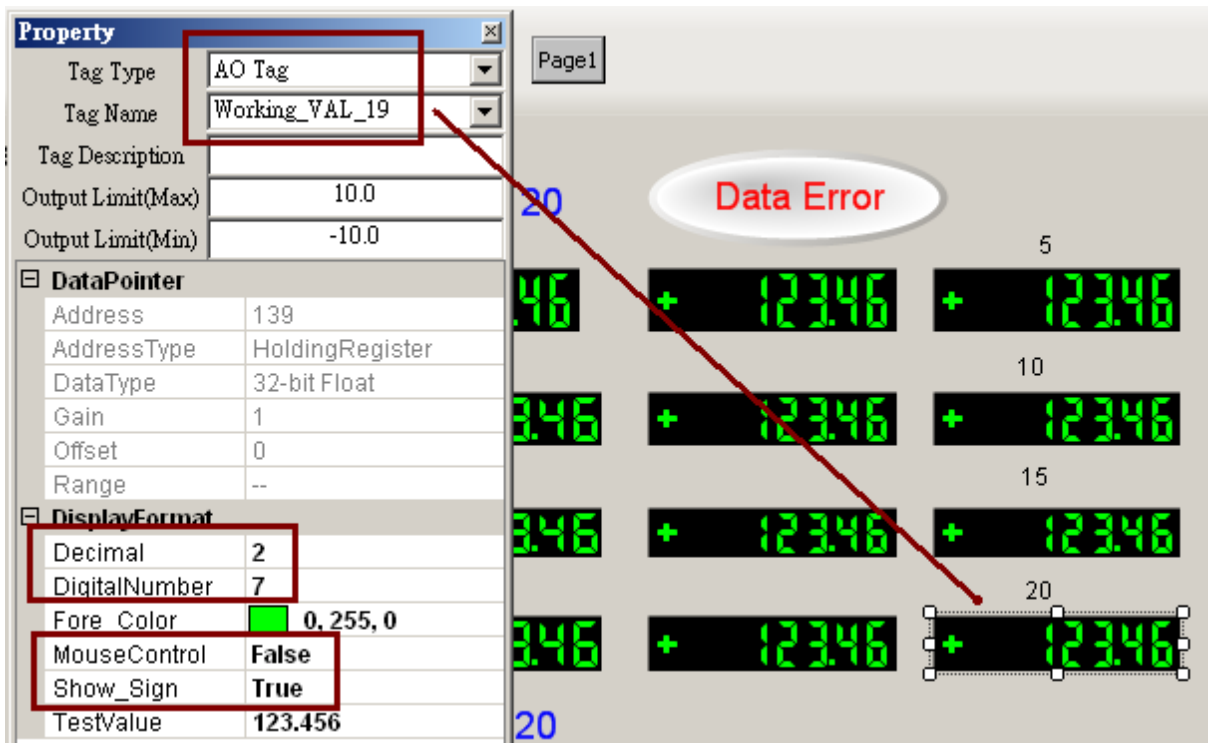
**Property**

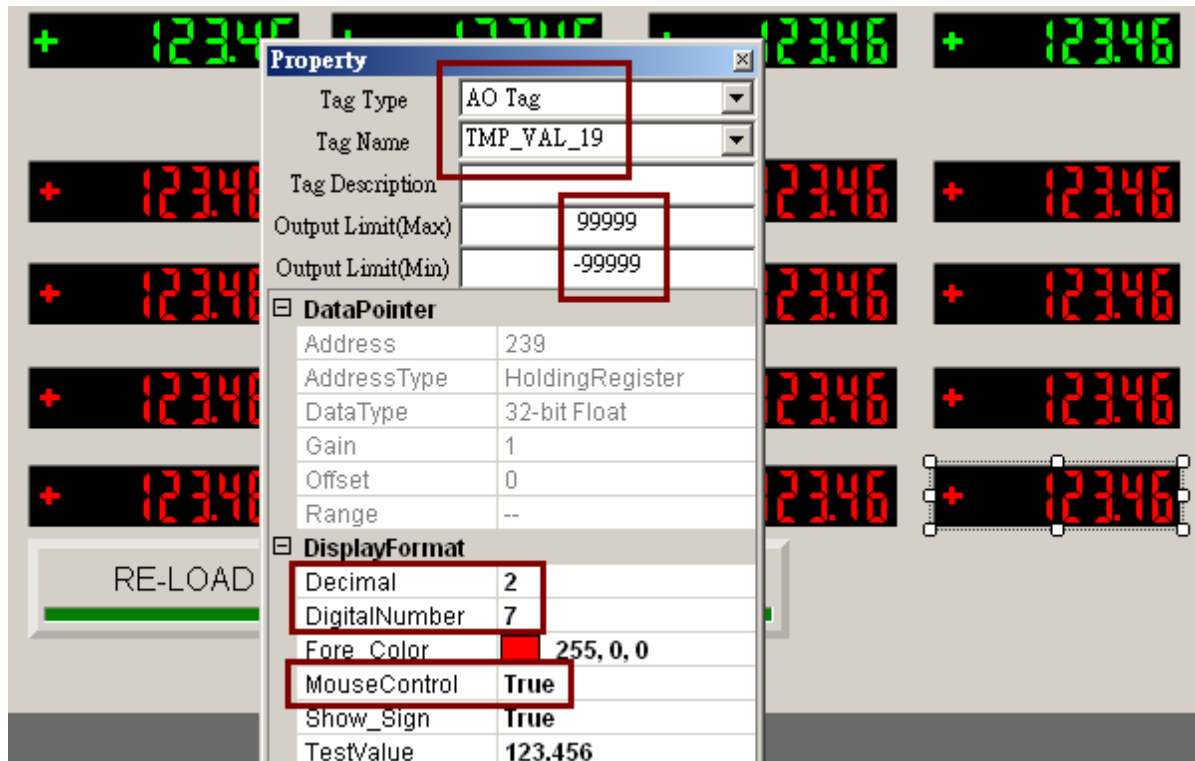
Tag Type	DO Tag
Tag Name	RE_LOAD
Tag Description	Load data from file
Output Limit(Max)	0.000
Output Limit(Min)	0.000
<b>DataPointer</b>	
Address	1
AddressType	CoilStatus
<b>DisplayFormat</b>	
ConfirmWindow	True
FontSize	12
MouseControl	True
OffColor	0, 127, 0
OffDisplayText	RE-LOAD
OnColor	0, 255, 0
OnDisplayText	RE-LOAD
ValueTest	False

The image shows a vertical stack of five digital displays. The top four displays show '12346' in red text. The bottom display shows 'RE-LOAD' in green text. Each display is flanked by '+' signs. A label 'RE-LOAD' with a green bar is visible at the bottom left of the display area.







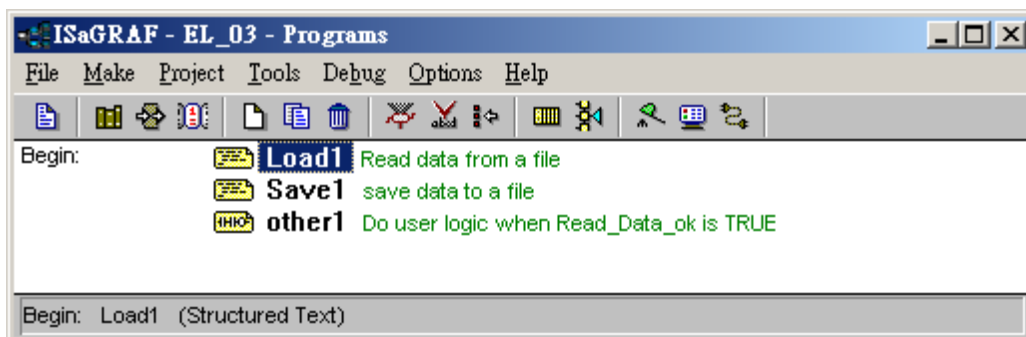


ISaGRAF :

變數宣告: (關於 ISaGRAF 變數陣列的說明,請參考 FAQ-039)

Name	Type	Attrib.	NetW. Addr.	說明
TMP	Boolean	Internal	0	
RE_LOAD	Boolean	Internal	1	設 TRUE 來讀 File
RE_Store	Boolean	Internal	2	設 TRUE 來存 File
INIT	Boolean	Internal	0	需宣告 初值為 TRUE
Read_Data_OK	Boolean	Internal	3	True: 資料正確, False: 資料錯誤.
Msg1	Message	Internal	0	需宣告 Len 為 128
str1	Message	Internal	0	需宣告 Len 為 255
TMP_file_name	Message	Internal	0	需宣告 Len 為 128
File1	Integer	Internal	0	
Working_ReaL_VAL[0..19]	REAL	Internal	101	目前工作中的應用參數 為一個 ISaGRAF 變數陣列 需宣告第一個 Addr. 為 101
TMP_Real_VAL[0..19]	REAL	Internal	201	為一個 ISaGRAF 變數陣列需宣告 第一個 Addr. 為 201
NUM1	Integer	Internal	0	
ii	Integer	Internal	0	Index of “for” loops
jj	Integer	Internal	0	Index of “for” loops
Data_Cnt	Integer	Internal	0	
Row_Cnt	Integer	Internal	0	有幾列資料, 本例需宣告初值為 2
Data_Cnt_in_Row	Integer	Internal	0	每列有幾個資料, 本例需宣告初值 為 10

Project :



Load1 程式 :

```
(* 第一個 PLC Scan 內的動作, INIT 的初值需宣告為 TRUE *)
if INIT then

  (* 指定 NetWork Addr 給變數陣列 *)
  (* 指定 Working_Real_VAL[0..19] 的 netWork addr 為 101 , 103, 105, ..., 139 *)
  TMP := S_MB_ADR( 101 , 20 , 1 ) ;
  (* 指定 TMP_Real_VAL[0..19] 的 netWork addr 為 201 , 203, 205, ..., 239 *)
  TMP := S_MB_ADR( 201 , 20 , 1 ) ;

  (* 設定要儲存目前 Working parameters 的 file 路徑與名稱 *)
  TMP_file_name := '\System_Disk\Working_Real.txt' ;
  RE_LOAD := True ; (* PAC 剛開機,要驅動去從 File 讀出 Working parameters 資料 *)

end_if ;

(* 若 RE_LOAD 被設為 TRUE, 開啓 file 並讀資料 *)
if RE_LOAD then

  RE_LOAD := FALSE ; (* 還原為 FALSE *)
  Read_Data_ok := False ; (* 一開始先設為資料錯誤 *)
  Data_Cnt := 0 ; (* 一開始先設為還沒讀到資料 *)

  File1 := f_wopen( TMP_file_name ) ; (* 開啓 file 為可讀可寫模式 *)

  if File1 = 0 then

    (* 開啓 file 失敗, 離開本 ST 程式 *)
    Msg1 := 'Can not Open file ' + TMP_file_name ;
    INIT := False ; (* 離開本 ST 程式前要將 INIT 設為 False *)
    return ;

  end_if ;

  (* 讀出 file 內的 2 列資料, 每列有 10 個實數, 總共 20 個實數資料 *)
  for ii := 0 to ( Row_Cnt - 1 ) do

    if f_eof(File1) = TRUE then (* 測試是否已抵達 file 尾端 *)

      (* 已抵達 file 尾端, 表示沒資料了 *)
      Msg1 := 'There should be at least ' + MSG(Row_Cnt)+ ' rows in ' +TMP_file_name+ ' !!!' ;
      exit ; (* 離開這個 for loop *)

    end_if ;
```

```

str1 := fm_read(File1); (* 從 file 內讀出一列字串(Message) *)
(* 轉換字串內容為數個實數資料, 並將這些資料存放到 1 號 Float Array 內 *)
NUM1 := Msg_F( str1 , 1 );
if NUM1 <> Data_Cnt_in_Row then (* 本例一行需有 10 個實數資料 *)
(* 若非 10 個即為錯誤, 可能是資料並非實數或資料數量太多或不夠 *)
Msg1 := 'The data format of No.' + Msg( ii + 1 ) + ' row is not correct or data number is not ' + MSG( Data_Cnt_in_Row );
exit ; (* 離開這個 for loop *)
end_if;
Data_Cnt := Data_Cnt + Data_Cnt_in_Row; (* 正確! 累加 Data_Cnt *)
(* 將此 10 個實數由 1 號 Float Array 內讀出並存入 TMP_Real_VAL[ ] 陣列內 *)
for jj := 0 to ( Data_Cnt_in_Row - 1 ) do
(* 這些資料原先是放在 1 號 Float array 內的 addr. 1 到 10 *)
TMP_Real_VAL[ Data_Cnt_in_Row * ii + jj ] := ARY_F_R( 1 , jj + 1 );
end_for ;
end_for ;
(* File 有開啓過,處理完後就必須要使用 f_close()來關閉它 *)
TMP := f_close(File1);
(* 已讀出正確的 20 個資料 *)
if ( Data_Cnt = Row_Cnt * Data_Cnt_in_Row ) then
Msg1 := 'Read ' + TMP_file_name + ' Ok ';
Read_Data_Ok := True ; (* 設資料狀態為 『資料正確』 *)
(* 若本 scan 為第 1 個 scan 需將 TMP_Real_Val[ ] 值複製到 Working_ReaL_Val[ ]內 *)
if INIT then
for ii := 0 to ( Data_Cnt - 1 ) do
Working_ReaL_VAL[ii] := TMP_Real_VAL[ii];
end_for ;
end_if ;
end_if ;
end_if ;
INIT := False ; (* 設為 False, 表示不再是第 1 個 PLC scan 了 *)

```

Save1 程式:

```
(* 當 RE_Store 被設為 True 時, 將資料存入 file 內 *)
(* 此 "RE_Store"之值可以被 eLogger HMI 來設為 True *)
if RE_Store then

  RE_Store := False ; (* 還原回 False *)
  Read_Data_ok := False ; (* 一開始先設資料狀態為錯誤 *)
  Data_Cnt := 0 ; (* 一開始先設為 0 *)

  TMP := f_delete( TMP_file_name ) ; (* 刪除舊 file *)

  (* 建立一個新 file 來寫資料 *)
  File1 := f_creat( TMP_file_name ) ;

  (* 建立新 file 發生錯誤, 離開本 ST 程式 *)
  if File1 = 0 then
    MSG1 := 'Can not Create a new file - ' + TMP_file_name + ' !' ;
    return ; (* 離開本 ST 程式 *)
  end_if ;

  (* file 格式為 2 列資料, 每列有 10 個實數, 總共 20 個實數資料 *)
  for ii := 0 to ( Row_Cnt - 1 ) do

    str1 := " ; (* 一開始先設為空字串 *)

    (* 將資料轉為字串, 1 列有 10 個實數資料 *)
    for jj := 0 to ( Data_Cnt_in_Row - 2 ) do
      str1 := str1 + Real_Str( TMP_ReaL_VAL[Data_Cnt_in_Row * ii + jj] ) + ',' ;
    end_for ;
    (* 最後 1 個資料尾端要加上 <CR><LF> *)
    str1 := str1 + Real_Str( TMP_ReaL_VAL[Data_Cnt_in_Row * ii + jj] ) + '$0D$0A' ;

    (* 將此列字串寫入 file *)
    TMP := F_writ_S( File1 , str1 ) ;
    if TMP = False then
      MSG1 := 'Write data to file - ' + TMP_file_name + ' failed !' ;
      exit ; (* 寫入錯誤, 離開這個 for loops *)
    end_if ;
    Data_Cnt := Data_Cnt + Data_Cnt_in_Row ; (* 正確! 累加 Data_Cnt *)

  end_for ;
```

```
TMP := f_close(File1); (* File 有開啓過,處理完後就必須要使用 f_close( )來關閉它 *)
```

```
(* 資料儲存正確後... *)
```

```
if ( Data_Cnt = Row_Cnt * Data_Cnt_in_Row ) then
```

```
  Msg1 := 'Write ' + TMP_file_name + ' Ok ';
```

```
  Read_Data_ok := True; (* 設狀態為 『資料正確』 *)
```

```
(* 複製 TMP_Real_Val[ ] 值到 Working_ReaL_Val[ ] *)
```

```
  for ii := 0 to ( Data_Cnt - 1 ) do
```

```
    Working_ReaL_VAL[ii] := TMP_ReaL_VAL[ii];
```

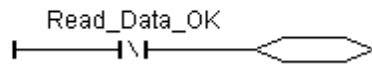
```
  end_for;
```

```
end_if;
```

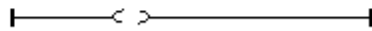
```
end_if;
```

Other1 程式：

```
(* 若 Read_OK 為 False,表示應用參數 "Working_ReaL_VAL[]" 之值還不正確,就 return *)
```



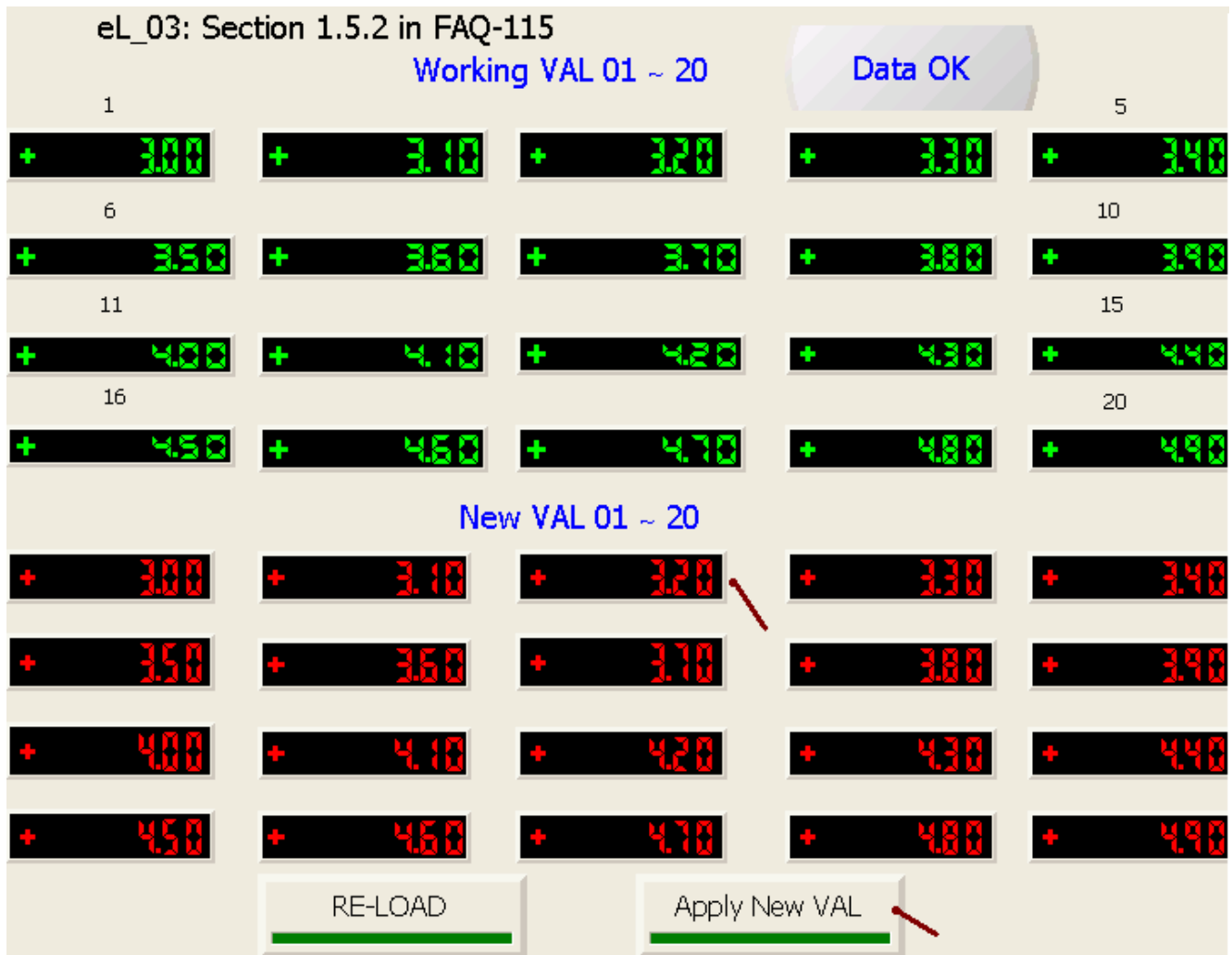
```
(* 若 Read_OK 為 True,表示 "Working_ReaL_VAL[]" 是正確的,可繼續Run下方新增加的程式... *)
```



如何測試：

你可以更改下方 New VAL 區域內的一些數值, 然後按下 Apply New Val 按鈕並設為 ON, 之後上方的 Working VAL 區就會變更為 剛剛所設定的值. 若 PC / ISaGRAF 有連上 WP-8xx7, 也可以看到相對應的數值已經改變為新輸入的值.

接下來將 WP-8xx7 關機 後 再開機, 可以看到 開完機時, Working VAL 區已經是套用為新輸入的數值.





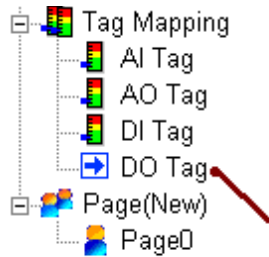
### 1.5.3: 讀取與變更 Controller 的日期 / 時間 與 進行時間控制

ISaGRAF 範例程式為： eL\_06.pia

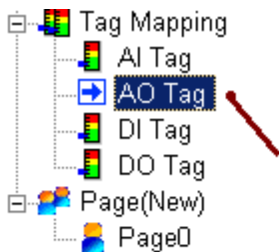
eLogger 範例程式為： eL\_06.wez

本範例可以顯示目前控制器的日期與時間, 並可以變更它, 另外可以在週一到週五 09:00 ~ 18:00 將 OUT1 輸出為 ON, 週六至週日 09:00 ~ 12:00 將 OUT2 輸出為 ON.

Elogger Developer:



Tag Name	Description	Memory Address	Data Type
Set_Time	Set as New Date / Time	101	
Refresh_Time	Refresh as Curren...	102	



Tag Name	Description	Memory Address	Data Type
Year1		1	16-bit Signed Inte...
Month1		2	16-bit Signed Inte...
Day1		3	16-bit Signed Inte...
Hour1		4	16-bit Signed Inte...
Minute1		5	16-bit Signed Inte...
Second1		6	16-bit Signed Inte...
W_Year1		11	16-bit Signed Inte...
W_Month1		12	16-bit Signed Inte...
W_Day1		13	16-bit Signed Inte...
W_Hour1		14	16-bit Signed Inte...
W_Minute1		15	16-bit Signed Inte...
W_Second1		16	16-bit Signed Inte...

06: Section 1.5.3 in FAQ-115

ent Day / Month / Year and

123

ew Day / Month / Year and

123

refresh as Current Date / time

Property	
Tag Type	AO Tag
Tag Name	Day1
Tag Description	
Output Limit(Max)	32767
Output Limit(Min)	-32768
DataPointer	
Address	3
AddressType	HoldingRegister
DataType	16-bit Signed Integer
Gain	1
Offset	0
Range	-32768.000~32767.000
DisplayFormat	
Decimal	0
DigitalNumber	2
Fore_Color	0, 255, 0
MouseControl	False
Show_Sign	False
TestValue	123.456

06: Section 1.5.3 in FAQ-115

Day / Month / Year and

123

/ Day / Month / Year and

123

refresh as Current Date / time

Property	
Tag Type	AO Tag
Tag Name	Year1
Tag Description	
Output Limit(Max)	32767
Output Limit(Min)	-32768
DataPointer	
Address	1
AddressType	HoldingRegister
DataType	16-bit Signed Integer
Gain	1
Offset	0
Range	-32768.000~32767.000
DisplayFormat	
Decimal	0
DigitalNumber	4
Fore_Color	0, 255, 0
MouseControl	False
Show_Sign	False
TestValue	123.456

Property

Tag Type	AO Tag
Tag Name	W_Day1
Tag Description	
Output Limit(Max)	31.0
Output Limit(Min)	1.00
<b>DataPointer</b>	
Address	13
AddressType	HoldingRegister
DataType	16-bit Signed Integer
Gain	1
Offset	0
Range	-32768.000~32767.000
<b>DisplayFormat</b>	
Decimal	0
DigitalNumber	2
Fore_Color	0, 255, 0
MouseControl	True
Show_Sign	False
TestValue	123.456

Property

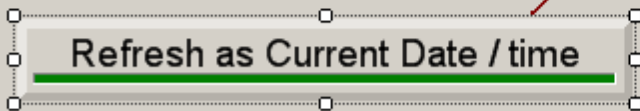
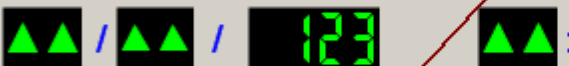
Tag Type	AO Tag
Tag Name	W_Year1
Tag Description	
Output Limit(Max)	2079
Output Limit(Min)	2009
<b>DataPointer</b>	
Address	11
AddressType	HoldingRegister
DataType	16-bit Signed Integer
Gain	1
Offset	0
Range	-32768.000~32767.000
<b>DisplayFormat</b>	
Decimal	0
DigitalNumber	4
Fore_Color	0, 255, 0
MouseControl	True
Show_Sign	False
TestValue	123.456

eL\_06: Section 1.5.3 in FAQ-115

Current Day / Month / Year and Time



Set New Day / Month / Year and Time



Property

Tag Type	DO Tag
Tag Name	Refresh_Time
Tag Description	Refresh as Current Date / Time
Output Limit(Max)	0.000
Output Limit(Min)	0.000
DataPointer	
Address	102
AddressType	CoilStatus
DisplayFormat	
ConfirmWindow	False
FontSize	14
MouseControl	True
OffColor	0, 127, 0
OffDisplayText	Refresh as Current Da
OnColor	0, 255, 0
OnDisplayText	
ValueTest	False

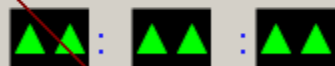
Property

Tag Type	DO Tag
Tag Name	Set_Time
Tag Description	Set as New Date / Time
Output Limit(Max)	0.000
Output Limit(Min)	0.000
DataPointer	
Address	101
AddressType	CoilStatus
DisplayFormat	
ConfirmWindow	True
FontSize	14
MouseControl	True
OffColor	0, 127, 0
OffDisplayText	Set Date / Time
OnColor	0, 255, 0
OnDisplayText	Set Date / Time
ValueTest	False

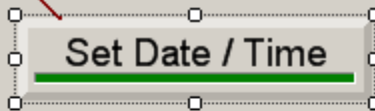
Time

A digital display showing the time. It consists of two pairs of green triangles for the hour and minute, separated by a colon.

Set Time



Set

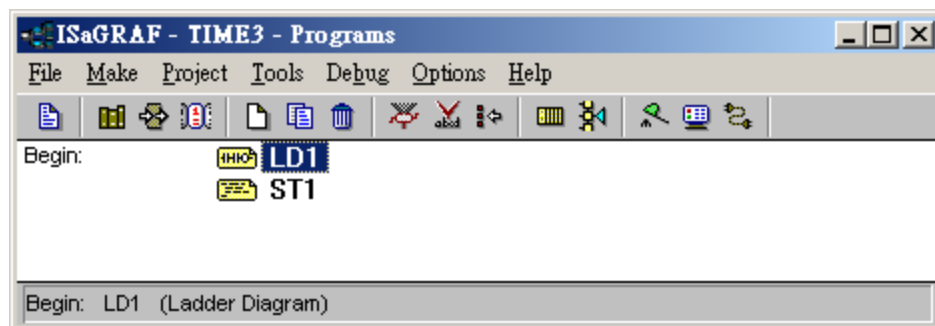


ISaGRAF:

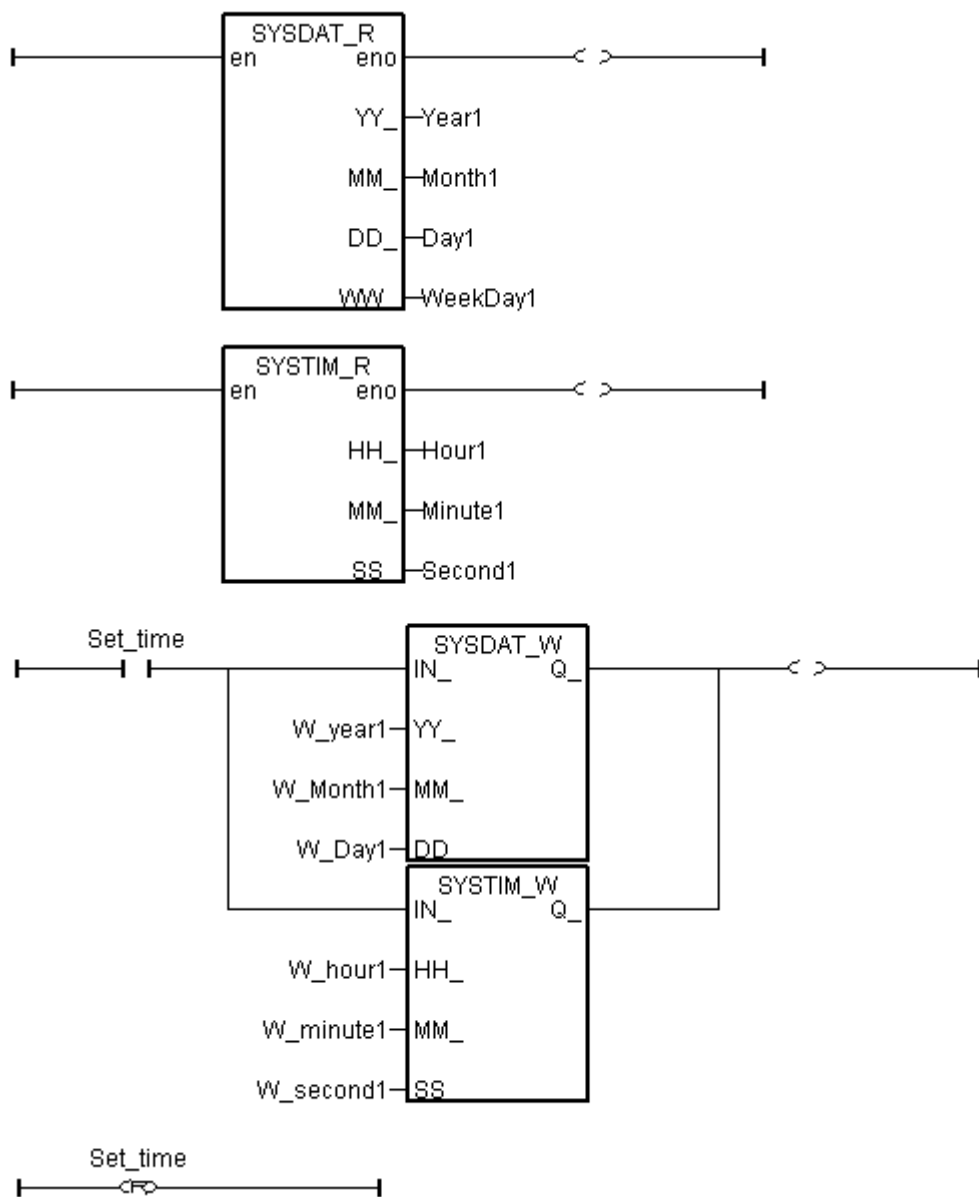
變數宣告:

Name	Type	Attrib.	NetW. Addr.	說明
Set_time	Boolean	Internal	101	設為 True 來變更日期與時間
Refresh_Time	Boolean	Internal	102	刷新為目前日期 / 時間
OUT1	Boolean	Output	0	連到 I-87055W 的 CH.1
OUT2	Boolean	Output	0	連到 I-87055W 的 CH.2
time_val	Integer	Internal	0	單位為分 0 ~ 1439 表示 00:00 ~ 23:59
Year1	Integer	Internal	1	取得 年
Month1	Integer	Internal	2	取得 月
Day1	Integer	Internal	3	取得 日
Hour1	Integer	Internal	4	取得 時
Minute1	Integer	Internal	5	取得 分
Second1	Integer	Internal	6	取得 秒
WeekDay1	Integer	Internal	0	取得 星期幾
W_Year1	Integer	Internal	11	新的 年
W_Month1	Integer	Internal	12	新的 月
W_Day1	Integer	Internal	13	新的 日
W_Hour1	Integer	Internal	14	新的 時
W_Minute1	Integer	Internal	15	新的 分
W_Second1	Integer	Internal	16	新的 秒

Project :



LD1 程式：



ST1 程式:

```
(* 單位為秒, 0 ~ 86399
Time_val := 3600 * Hour1 + 60* Minute1 + Second1 ;
*)

(* 單位為分, 0 ~ 1439 *)
Time_val := 60 * Hour1 + Minute1 ;

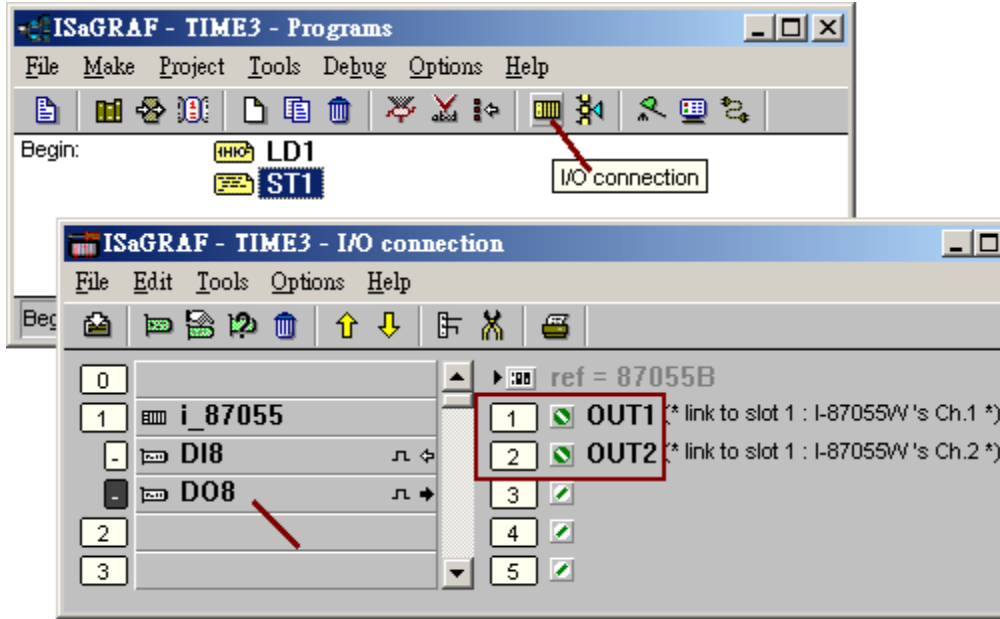
OUT1 := False ; (* 一開始先設為 False *)
OUT2 := False ;

(* 週一至週五 9:00 到 18:00 將 OUT1 設為 True *)
if (WeekDay1>=1) and (WeekDay1<6) and
  (Time_val >= 540) and (Time_val < 1080) then
  OUT1 := True ;
end_if ;

(* 週六至週日 9:00 到 12:00 將 OUT2 設為 True *)
if (WeekDay1=6) and (WeekDay1=7) and
  (Time_val >= 540) and (Time_val < 720) then
  OUT2 := True ;
end_if ;

(* 刷新為目前的日期與時間 *)
if Refresh_Time then
  Refresh_Time := False ; (* 還原回 False *)
  W_Year1 := Year1 ;
  W_Month1 := Month1 ;
  W_Day1 := Day1 ;
  W_Hour1 := Hour1 ;
  W_Minute1 := Minute1 ;
  W_Second1 := Second1 ;
end_if ;
```

I/O 連結:



如何測試:

點選下方的 日期或時間 來輸入新的日期與時間, 然後按下 Set Dat / Time 按鈕來變更它. 之後可看到上方的日期 / 時間就會變更為新的 日期 / 時間 .

