如何在 Wincon 內 讀 / 寫 File 資料 ?

注意:
1. 假如在 File 內的相對位置並未儲存 浮點數(實數) , 但卻要從 File 內對該位置使用 F_REAF_F 來讀取 浮點數(實數) 可能會發生"Controller Fault 編號 117" . 請參考第 10.6 節的說明.
2.Wincon-8xx7/8xx6 支持 檔案存取 功能, 而 I-8xx7 & I-7188EG/XG 則沒有.
3. 若檔案位置在 '\CompactFlash\' 內 (例如 \CompactFlash\data1.txt), 則會儲存在 Compact Flah Disk 內, Wincon 關機後檔案仍然存在, 在 \CompactFlash 內 Read / Write 速度很慢.
4. 若檔案位置不在 '\CompactFlash\' 內 (例如 \Temp\data2.txt), 則會儲存在 RAM 內, Wincon 關機後檔案就消失,但優點是 RAM 檔案的 Read / Write 速度很快 (CompactFlash 的 Read / Write 很慢)

W-8xx7 支持以下的 **ISaGRAF 標準函式**

F_ROPEN	以 Binary 格式開啓已經存在的檔案以供 讀取 .
F_WOPEN	以 Binary 格式開啓已經存在的檔案以供 讀取 & 寫入 .
F_CLOSE	關閉已經開啓的檔案
F_EOF	測試是否 檔案指標 已達 檔案結尾
FA_READ	從檔案內讀出 1 個 binary 長整數 (4 bytes, signed).
FA_WRITE	寫 1 個 binary 長整數(4 bytes, signed) 到檔案內.
FM_READ	從檔案內讀出 1 個 字串.
FM_WRITE	寫 1 個字串到檔案內, 會附加 <CR> <LF> 符號於字串後方

W-8xx7 支持以下 **ICP DAS 提供的函式**

F_APPEND	附加一個檔案於另一個檔案的尾端.
F_COPY	覆製 1 個檔案.
F_CREAT	建立 1 個新的檔案.
F_DELETE	刪除 1 個檔案.
F_DIR	建立 1 個新資料夾 (目錄).
F_END	移動 檔案指標 到檔案的尾端.
F_SEEK	移動 檔案指標 到某個位置
F_READ_B	從檔案內讀出 1 個 binary 字元(0 - 255) (1 byte, unsigned).
F_WRIT_B	寫 1 個 binary 字元(0 -255) 到檔案內 (1 byte, unsigned).
F_READ_W	從檔案內讀出 1 個 binary 字組(-32768 to +32767) (2 byte, signed) .
F_WRIT_W	寫 1 個 binary 字組(-32768 to +32767) 到檔案內 (2 byte, signed).
F_READ_F	從檔案內讀出 1 個 binary 實數(例如 123.45, -2.15E-03, …) (4 byte).
F_WRIT_F	寫 1 個 binary 實數 到檔案內 (4 byte) .
F_WRIT_S	寫 1 個字串到檔案內, 不附加 <CR> <LF> 符號於字串後方.

第 11.3.6 節 或 以下路徑可找到 Wincon-8xx7 對於 檔案存取 的範例程式
Wincon CD-ROM: \napdos\isagraf\wincon\demo\ "wdemo_54.pia" , 55, 56, 51, 50, 1 或 2 或
ftp://ftp.icpdas.com/pub/cd/wincon_isagraf/napdos/isagraf/wincon/demo/ 內

10.5.1: 範例 Wdemo_51: 從檔案內讀取 10 個實數值, 共 10 行, 每行 1 個實數

本範例程式爲 Wdemo_51.pia 放於

W-8xx7 CD-ROM:\napdos\isagraf\wincon\demo\ 內 或

ftp://ftp.icpdas.com/pub/cd/wincon_isagraf/napdos/isagraf/wincon/demo/

若您 PC 上裝的 ISaGRAF 找不到 Msg_F , Msg_N , ARY_F_R, AFY_F_W 等函式, 請訪問 http://www.icpdas.com/products/PAC/i-8000/isagraf.htm 來下載 “ICP DAS utilities For ISaGRAF” , 解壓縮後, 執行其內的 setup.exe 來安裝新的 函式進去 ISaGRAF 內

本 Wdemo_51 範例程式 一開機 或 將 “RE_LOAD” 變數 再設爲 TRUE 時, 會再去讀 \CompactFlash\data51.txt 來更新 10 個 REAL 變數之值.
(在 CompactFlash 卡內操作檔案很花 CPU 時間, 讀/寫 完後請 將檔案 Close, 並且避免每個 PLC Scan 都在 Read / Write 檔案 , 不然 PLC Scan Time 會變很大)

要使這個範例仍夠正確運作, 請在 PC 用 NotePad (記事本) 編一個 “data51.txt” 文字檔, 共 10 行, 每行有 1 個實數值, 然後用 ftp 傳到 Wincon 的 \CompactFlash\ 路徑內.
“data51.txt” 文字檔 內容可如下:

```
2.345
999.03
-1.01
456.789
2
456.77
5.9E-12
32.3
45.1
33.3
```

變數宣告:

Name	***Type***	***Attribute***	***Description***
RE_LOAD	Bool	Internal	設爲 True 會去讀 File 一次, 宣告初值爲 TRUE
TMP	Bool	Internal	暫時使用的 Boolean 變數
File_name1	Message	Internal	長度 64, 初值爲 \CompactFlash\data51.txt
Msg1	Message	Internal	長度 128, 會顯示 File 處理狀態
str1	Message	Internal	長度 255, 用來儲存讀到 File 內的一行字串
F_VAL[0..9]	REAL	Internal	爲變數陣列, Dim 設爲 10. 讀到的 10 個 REAL 值
TMP_F	REAL	Internal	暫時使用的 REAL 變數
File1	Integer	Internal	File ID
ii	Integer	Internal	給 for 迴圈使用的 index

ST 程式:

```
if  RE_LOAD  then    (* 若將 RE_LOAD 設爲 TRUE , 會去讀 File 一次 *)

  RE_LOAD  :=  FALSE  ;

  File1 := f_wopen( File_name1 )  ;   (*  開啓 File 爲 可讀 / 可寫 模式 *)

  if  File1 = 0  then    (* 若回傳 0, 表示開啓 File 失敗 *)
   Msg1  :=  'Can not Open file '  +  File_name1 ;
   return ;   (* 無法開啓該 File, 用 Return 離開本 ST 程式 *)
  end_if  ;

  for  ii  :=  0  to  9  do   (* 共要讀 10 行 *)

    if   f_eof(File1)   =   TRUE    then    (* 偵測是否抵達 File 的 尾端 *)
      Msg1  :=  'Data number is not enough in '  +  File_name1 ;
      Exit  ;     (* 離開 for 迴圈 *)
    end_if  ;

    str1  :=  fm_read(File1)  ;   (* 讀取 File 內的一行字串 *)
    TMP_F  :=  str_real(str1) ;   (* 將字串轉成 實數 *)
    if  TMP_F  =  1.23E-20   then   (* 若傳回 1.23E-20 表示 格式錯誤  *)
      Msg1  :=  'The '  +  Msg(ii+1)  +  'th Data format is not correct !'  ;
      exit  ;     (* 離開 for 迴圈 *)
    end_if ;

    F_VAL[ii]  :=  TMP_F  ;   (* 前面 有轉換成功, 值存入 F_VAL[0..9] 內 *)

  end_for ;

  TMP  :=  f_close(File1)  ;   (* File 只要成功開啓過, 處理完就需 close *)
  If   ii=10   then   (* 10 行資料都讀到了 *)
    Msg1 := 'Read ' + File_name1 + ' Ok ' ;
  end_if ;

end_if  ;
```

10.5.2: 範例 Wdemo_54: 從檔案內讀取 20 個實數值, 共 4 行, 每行 5 個實數

本範例程式爲 Wdemo_54.pia 放於

W-8xx7 CD-ROM:\napdos\isagraf\wincon\demo\ 內 或

ftp://ftp.icpdas.com/pub/cd/wincon_isagraf/napdos/isagraf/wincon/demo/

若您 PC 上裝的 ISaGRAF 找不到 Msg_F , Msg_N , ARY_F_R, AFY_F_W 等函式, 請訪問 http://www.icpdas.com/products/PAC/i-8000/isagraf.htm 來下載 “ICP DAS utilities For ISaGRAF” , 解壓縮後, 執行其內的 setup.exe 來安裝新的 函式進去 ISaGRAF 內

本 Wdemo_54 範例程式 一開機 或 將 “RE_LOAD” 變數 再設爲 TRUE 時, 會再去讀 \CompactFlash\data54.txt 來更新 20 個 REAL 變數之值.
(在 CompactFlash 卡內操作檔案很花 CPU 時間, 讀/寫 完後請 將檔案 Close, 並且避免每個 PLC Scan 都在 Read / Write 檔案 , 不然 PLC Scan Time 會變很大)

要使這個範例仍夠正確運作, 請在 PC 用 NotePad (記事本) 編一個 “data54.txt” 文字檔, 共 4 行, 每行有 5 個實數值, 然後用 ftp 傳到 Wincon 的 \CompactFlash\ 路徑內.
“data54.txt” 文字檔 內容可如下:

```
23 ,  65.9 , 0.012 , 5.87 , 88.2
0.34 , 8.0005 , -2.0E8 ,  4.08 , 5.32E-6
2 ,  -7 ,  6666.8 , 456.07  ,  1.01
5  ,  6  , 7 , 8 ,  9
```

變數宣告:

Name	***Type***	***Attribute***	***Description***
RE_LOAD	Bool	Internal	設爲 True 會去讀 File 一次, 宣告初值爲 TRUE
TMP	Bool	Internal	暫時使用的 Boolean 變數
File_name1	Message	Internal	長度 64, 初值爲 \CompactFlash\data54.txt
Msg1	Message	Internal	長度 128, 會顯示 File 處理狀態
str1	Message	Internal	長度 255, 用來儲存讀到 File 內的一行字串
F_VAL[0..19]	REAL	Internal	爲變數陣列, Dim 設爲 20. 讀到的 20 個 REAL 值
NUM1	Integer	Internal	接收 Msg_F( ) 的回傳值, 若爲 -1 表示格式錯誤
File1	Integer	Internal	File ID
ii	Integer	Internal	給 for 迴圈使用的 index
jj	Integer	Internal	給 另一個 for 迴圈使用的 index

ST 程式:

```
if  RE_LOAD  then    (* 若將 RE_LOAD 設爲 TRUE , 會去讀 File 一次 *)

  RE_LOAD  :=  FALSE  ;

  File1 := f_wopen( File_name1 )  ;   (*  開啓 File 爲 可讀 / 可寫 模式 *)

  if  File1 = 0  then    (* 若回傳 0, 表示開啓 File 失敗 *)
   Msg1  :=  'Can not Open file '  +  File_name1 ;
   return ;   (* 無法開啓該 File, 用 Return 離開本 ST 程式 *)
  end_if  ;

  for  ii   :=   0   to   3   do   (* 共要讀 4 行 *)

    if   f_eof(File1)  =  TRUE  then    (* 偵測是否抵達 File 的 尾端 *)
      Msg1   :=   'There should be at least 4 rows in '  +  File_name1  + ' !!!'  ;
      Exit  ;   (* 離開 for 迴圈 *)
    end_if  ;

    str1  :=  fm_read(File1)  ;   (* 讀取 File 內的一行字串 *)

    NUM1  :=  Msg_F(str1 , 1)  ;   (* 轉換該字串爲 數個 REAL 值, 並存入 1 號 Float 陣列內 *)

    if   NUM1  <>  5   then   (* 轉換得到的 REAL 值數量不是 5 個, 缺資料 .若爲 -1 表示格式錯誤 *)
      Msg1 :=  'The '  + Msg(ii+1)  +  'th row data format is not correct or data number is not 5 !'  ;
      Exit  ;   (* 離開 for 迴圈 *)
    end_if  ;

    for  jj   :=   0  to   4   do
      (* 從 1 號 Float 陣列 1 到 5 位址內 取出轉換後的 REAL 值 存入 F_VAL[0..19] 內 *)
      F_VAL[ 5 * ii + jj]  :=  ARY_F_R( 1 ,  jj + 1 )  ;
    end_for ;

  end_for ;

  TMP   :=   f_close(File1)  ;   (* File 只要成功開啓過, 處理完就需 close *)

  If   ii = 4  then   (* 4 行資料都讀到了 *)
    Msg1  :=  'Read '  +  File_name1  +  ' Ok '  ;
  end_if  ;

end_if  ;
```

10.5.3: 範例 Wdemo_55: 從檔案內讀取 20 個整數值, 共 2 行, 每行 10 個整數

本範例程式爲 Wdemo_55.pia 放於

W-8xx7 CD-ROM:\napdos\isagraf\wincon\demo\ 內 或

ftp://ftp.icpdas.com/pub/cd/wincon_isagraf/napdos/isagraf/wincon/demo/

若您 PC 上裝的 ISaGRAF 找不到 Msg_F , Msg_N , ARY_F_R, AFY_F_W 等函式, 請訪問 http://www.icpdas.com/products/PAC/i-8000/isagraf.htm 來下載 “ICP DAS utilities For ISaGRAF” , 解壓縮後, 執行其內的 setup.exe 來安裝新的 函式進去 ISaGRAF 內

本 Wdemo_55 範例程式 一開機 或 將 “RE_LOAD” 變數 再設爲 TRUE 時, 會再去讀 \CompactFlash\data55.txt 來更新 20 個 Integer 變數之值.
(在 CompactFlash 卡內操作檔案很花 CPU 時間, 讀/寫 完後請 將檔案 Close, 並且避免每個 PLC Scan 都在 Read / Write 檔案 , 不然 PLC Scan Time 會變很大)

要使這個範例仍夠正確運作, 請在 PC 用 NotePad (記事本) 編一個 “data55.txt” 文字檔, 共 2 行, 每行有 10 個整數值, 然後用 ftp 傳到 Wincon 的 \CompactFlash\ 路徑內.
“data55.txt” 文字檔 內容可如下:

```
-1 , 1 , 2 , 3 , 4 , 5 , -6 , 7 , 8 , 9
100001 , 200002 , +300003 , 404 , -505 , 606 , 7007 , 8008 , 9009 , 10
```

變數宣告:

Name	***Type***	***Attribute***	***Description***
RE_LOAD	Bool	Internal	設爲 True 會去讀 File 一次, 宣告初值爲 TRUE
TMP	Bool	Internal	暫時使用的 Boolean 變數
File_name1	Message	Internal	長度 64, 初值爲 \CompactFlash\data55.txt
Msg1	Message	Internal	長度 128, 會顯示 File 處理狀態
str1	Message	Internal	長度 255, 用來儲存讀到 File 內的一行字串
N_VAL[0..19]	Integer	Internal	爲變數陣列, Dim 設爲 20. 讀到的 20 個 Integer 值
NUM1	Integer	Internal	接收 Msg_F( ) 的回傳值, 若爲 -1 表示格式錯誤
File1	Integer	Internal	File ID
ii	Integer	Internal	給 for 迴圈使用的 index
jj	Integer	Internal	給 另一個 for 迴圈使用的 index

ST 程式:

```
if  RE_LOAD  then    (* 若將 RE_LOAD 設爲 TRUE , 會去讀 File 一次 *)

  RE_LOAD  :=  FALSE  ;

  File1 := f_wopen( File_name1 )  ;   (*  開啓 File 爲 可讀 / 可寫 模式 *)

  if  File1 = 0  then    (* 若回傳 0, 表示開啓 File 失敗 *)
   Msg1  :=  'Can not Open file '  +  File_name1 ;
   return ;   (* 無法開啓該 File, 用 Return 離開本 ST 程式 *)
  end_if  ;

  for  ii   :=   0   to   1   do   (* 共要讀 2 行 *)

    if   f_eof(File1)  =  TRUE  then    (* 偵測是否抵達 File 的 尾端 *)
      Msg1   :=   'There should be at least 2 rows in '  +  File_name1  + ' !!!'  ;
      Exit  ;   (* 離開 for 迴圈 *)
    end_if  ;

    str1  :=  fm_read(File1)  ;    (* 讀取 File 內的一行字串 *)

    NUM1  :=  Msg_N(str1 , 2)  ;   (* 轉換該字串爲 數個 Integer 值, 並存入 2 號 Integer 陣列內 *)

    if   NUM1  <>  10   then   (* 轉換得到的 Integer 數量不是 10 個, 缺資料 .若爲 -1 表示格式錯誤 *)
      Msg1 :=  'The '  + Msg(ii+1)  +  'th row data format is not correct or data number is not 10 !'  ;
      Exit  ;    (* 離開 for 迴圈 *)
    end_if  ;

    for  jj   :=   0  to   9   do
      (* 從 2 號 Integer 陣列 1 到 10 位址內 取出轉換後的 Integer 值 存入 N_VAL[0..19] 內 *)
      N_VAL[ 10 * ii + jj]  :=  ARY_N_R( 2 ,  jj + 1 )  ;
    end_for ;

  end_for ;

  TMP   :=   f_close(File1)  ;   (* File 只要成功開啓過, 處理完就需 close *)

  If   ii = 2  then   (* 2 行資料都讀到了 *)
    Msg1  :=  'Read '  +  File_name1  +  ' Ok '  ;
  end_if  ;

end_if  ;
```

10.5.4: 範例 Wdemo_56: 將 1 ~ 255 個實數變數用 CompactFlash 卡來保存最終值

本範例程式爲 Wdemo_56.pia 放於

W-8xx7 CD-ROM:\napdos\isagraf\wincon\demo\ 內 或

ftp://ftp.icpdas.com/pub/cd/wincon_isagraf/napdos/isagraf/wincon/demo/

若您 PC 上裝的 ISaGRAF 找不到 Msg_F , Msg_N , ARY_F_R, AFY_F_W 等函式, 請訪問 http://www.icpdas.com/products/PAC/i-8000/isagraf.htm 來下載 “ICP DAS utilities For ISaGRAF” , 解壓縮後, 執行其內的 setup.exe 來安裝新的 函式進去 ISaGRAF 內

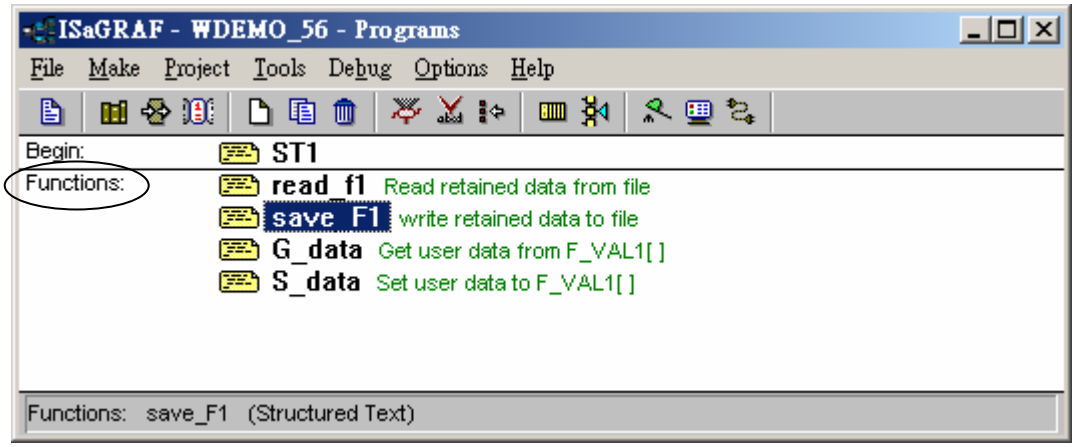
本 Wdemo_56 範例程式 一開機 會從 \CompactFlash\data56.txt 內讀取 1 ~ 255 個實數變數的 最終值, 若檔案不存在, 則指定 每個的 初值爲 0.0.
之後 只要任何一個數值有變, 就會將全部的 1 ~ 255 個新的值 存入 \CompactFlash\data56.txt 內. 檔案若不存在, 本範例程式會自動 建一個新的 \CompactFlash\data56.txt 檔案.

(在 CompactFlash 卡內 寫檔案 很花 CPU 時間, 讀/寫 完後請 將檔案 Close, 並且避免每個 PLC Scan 都在 Read / Write 檔案 , 不然 PLC Scan Time 會變很大
若值一直常常更改, 則不適合用 CompactFlash 卡來保存, 請參考第 10.1 節改使用 S256/512 來快速保存資料)

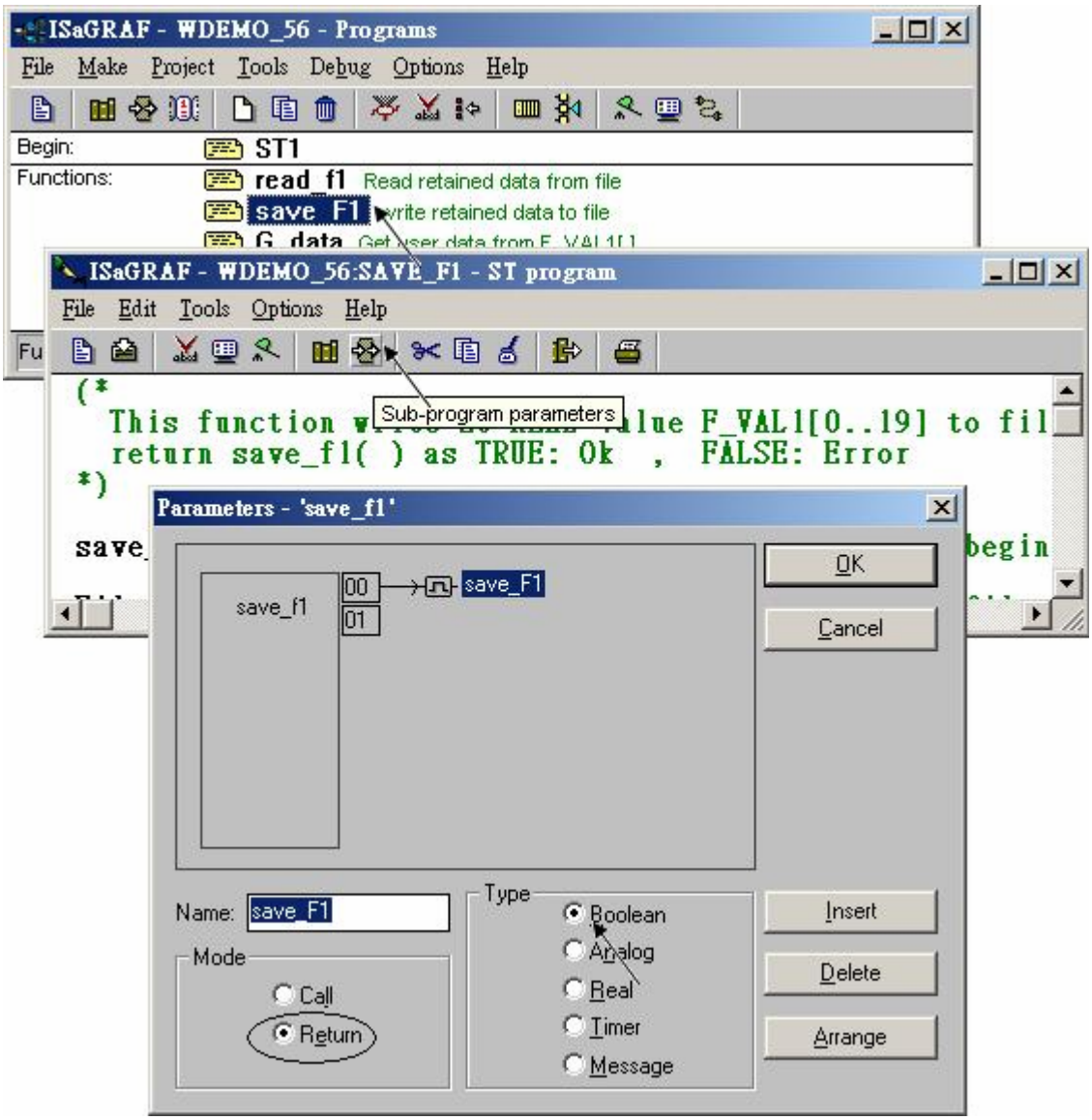
專案程式架構:

共有 5 個 ST 程式, 其中 reaf_f1 , save_f1 , G_data 與 S_data 爲 functions

重要:
1. 本範例程式 可以依照實際應用狀況來修改 常數變數 SIZE1 之值爲 1 到 255 之間的任一個值
2. 當 SIZE1 值有更改時, F_VAL1[ ] 與 Old_F_VAL1[ ] 的 ”Dim” 欄位也要改成同一個值, 同時, “G_data” 與 “S_data” 內的程式也需改成符合 User 需要的.
3. 使用 CompactFlash 卡來保存資料有個優點爲,這些資料的 File 可以預先在 PC 上編輯好, 再用 ftp 丟到 Wincon 內, 本例爲 \CompactFlash\data56.txt . 之後將 RE_LOAD 變數設爲 TRUE 一次, 它就會自動更新到 USER 的應用變數內.



本例, read_f1 , save_f1 , G_data 與 S_data 皆爲 functions, 其回傳值皆定義爲 Boolean 型態, 要宣告 functions 回傳值的型態, 如下 (第 15 章有詳細說明)



read_f1 與 save_f1 內有使用 區域變數 (Local variable):

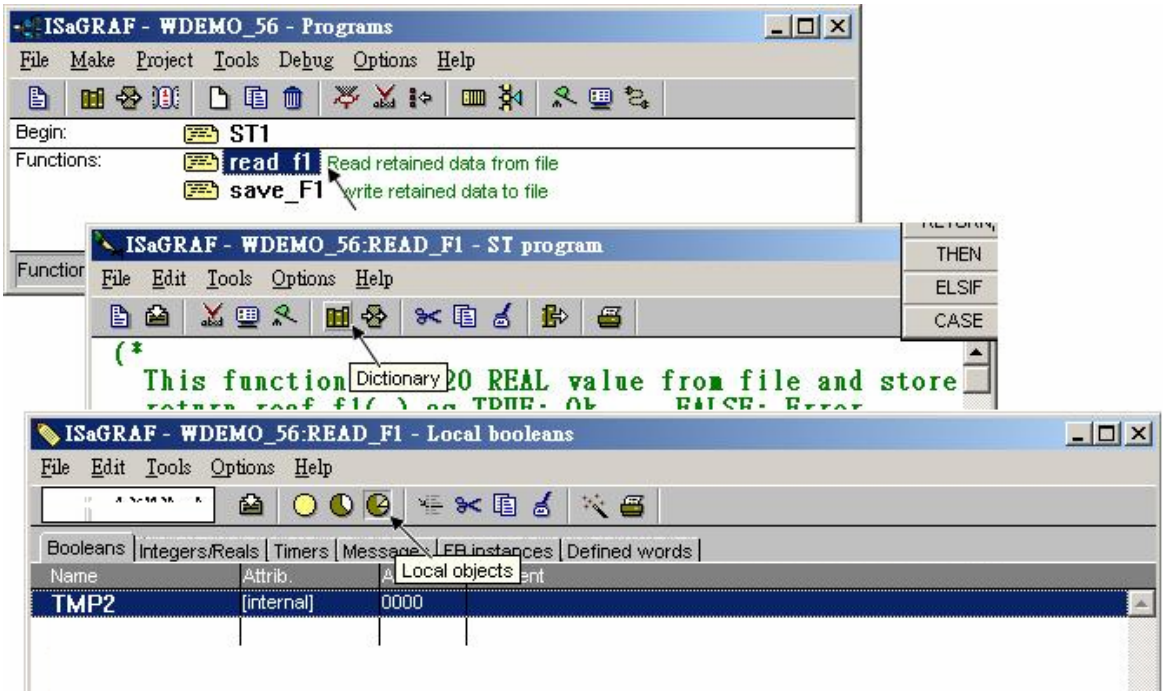
read_f1 程式的 區域變數 (Local variable) 如下:

Name	Type	Attribute	Description
TMP2	Bool	Internal	暫時使用的 Boolean 變數
ii2	Integer	Internal	給 for 迴圈使用的 index
jj2	Integer	Internal	給 for 迴圈使用的 index
num2	Integer	Internal	暫時使用的 Integer 變數

save_f1 程式的 區域變數 (Local variable) 如下:

Name	Type	Attribute	Description
TMP2	Bool	Internal	暫時使用的 Boolean 變數
ii2	Integer	Internal	給 for 迴圈使用的 index
jj2	Integer	Internal	給 for 迴圈使用的 index
num2	Integer	Internal	暫時使用的 Integer 變數

要宣告區域變數, 先雙擊 read_f1 程式進入, 之後 進入 Dictionary 內, 之後點選 “Local objects” 開始宣告 區域變數



全域變數 (Global variable)

Name	***Type***	***Attribute***	***Description***
SIZE1	Integer	Constant 常數	User 要保存的資料數量, 可以是 1 ~ 255 若改變此值, 需一併更改 F_VAL1[ ] 與 Old_F_VAL1[ ] 內宣告的 Dim 欄位爲相同值 本例設 SIZE1 爲 17
num_row1	Integer	Internal	檔案內資料共幾行, 此值由程式用 SIZE1 自動算出 本範例一行可放 10 個實數資料
Last_num1	Integer	Internal	檔案內最後一行資料的 資料有幾個 此值由程式用 SIZE1 自動算出
RE_LOAD	Bool	Internal	宣告初值爲 TRUE, 所以一開機會讀 File 一次 若程式運行中 又設此值爲 TRUE, 會再去讀 File 一次
TMP	Bool	Internal	暫時使用的 Boolean 變數
Data_Ok1	Bool	Internal	TRUE 表是讀 File Ok
Flag_to_save	Bool	Internal	Controller 要存 File 時, 會自動設它爲 TRUE
File_name1	Message	Internal	長度 64, 初值爲 \CompactFlash\data56.txt
Msg1	Message	Internal	長度 128, 會顯示 File 處理狀態
str1	Message	Internal	長度 255, 操作 File 字串時會用到
F_VAL1[0..19]	REAL	Internal	爲變數陣列, Dim 欄位設成與 SIZE1 之值相同
Old_F_VAL1 [0..19]	REAL	Internal	爲變數陣列, Dim 欄位設成與 SIZE1 之值相同 爲 F_VAL1[ ] 的舊值
NUM1	Integer	Internal	接收 Msg_F( ) 的回傳值, 若爲 -1 表示格式錯誤
File1	Integer	Internal	File ID
ii	Integer	Internal	給 for 迴圈使用的 index
jj	Integer	Internal	給 另一個 for 迴圈使用的 index
Data1 ~ Data5 與 Data06 ~ Data17	REAL	Internal	用來模擬 (仿真) 爲 User Data 變數, 本例因爲 SIZE1 值爲 17, 所以共有 17 個資料變數 User 實際的應用可以每個都使用不同的變數名稱. 若有更改名稱, G_data 與 S_data 內程式也要修改

ST 程式 ST1:

```
-------------------------------------------------------------------------------------------------------------------
if  RE_LOAD  then   (* 一開機, RE_LOAD 初值爲 TRUE, 會去讀 File 一次 *)

   RE_LOAD  :=  FALSE  ;   (* 進來之後需馬上將 RE_LOAD 設爲 FALSE *)

   (* 根據 SIZE1 之值, 計算出 共有幾行資料, 與最後一行有幾個 REAL 資料 *)
   num_row1  :=  SIZE1 / 10  ;
   last_num1  :=  SIZE1 - 10 * num_row1 ;
   if   last_num1  <>  0  then
      num_row1  :=  num_row1 + 1 ;
   else
      last_num1 := 10 ;
   end_if ;

   TMP  :=  read_F1( )  ;   (* 呼叫 read_f1( ) 來讀  SIZE1 個 資料到 F_VAL1[ ] 內 *)

   if  TMP = FALSE  then   (* read_f1( ) 若回傳 FALSE, 表示讀取失敗 *)

      for   ii  :=  0   to    SIZE1 - 1   do
         F_VAL1[ii]  :=  0.0  ;   (* 讀取失敗將  SIZE1 個初值設爲 0.0 *)
      end_for ;

      Data_Ok1  :=  FALSE  ;    (* 將 Data_Ok1 設爲 FALSE 來表示 “讀取失敗” *)
      Msg1  :=  'File : '  +  File_name1  +  ' not exist or data error ! or File is open now'  ;

   Else    (* 若讀取 File 成功 *)

      Data_Ok1 := TRUE ;    (* 將 Data_Ok1 設爲 TRUE 來表示 “讀取成功” *)
      Msg1  :=  'Get Retained data from file Ok '  ;

   end_if;

  (* 一開機時 要更新 Old_F_VAL1[  ] 之值 與 F_VAL1[  ] 相同 *)
   for  ii  :=  0   to    SIZE1 - 1    do
      Old_F_VAL1[ii]  :=  F_VAL1[ii]  ;
   end_for ;

   TMP  :=  G_data( )  ;   (* 每次從 File 內讀資料後 , 需更新 值 到 User Data 變數 內 *)

end_if  ;

(* 每個 PLC Scan 都需 把 User Data 變數值 更新到 F_VAL1[  ] 內 *)
TMP  :=  S_data( )  ;
```

```
(* 在每個 PLC scan 內判斷 這 SIZE1 個資料是否有任一個值有更改 *)
for   ii   :=   0   to     SIZE1 - 1    do

  if    Old_F_VAL1[ii]   <>   F_VAL1[ii]    then      (* 新/舊值不同, 表示有更改過 *)

     Flag_to_save  :=  TRUE  ;     (* 將 Flag 設爲 TRUE 來準備要 寫資料到 File 內*)
     Old_F_VAL1[ii]   :=  F_VAL1[ii] ;     (* 值不同時, 要更新 舊值 *)

  end_if  ;

end_for  ;

if  Flag_to_save  then     (* 若 Flag 被設爲 TRUE, 寫 SIZE1 個資料到 File 內 *)

  TMP  := save_f1( )  ;   (* 呼叫 save_f1( ) 來寫 資料 *)

  if   TMP  =  FALSE    then   (* save_f1( ) 回傳 FALSE, 表示寫 File 失敗,可能 File 被打開,沒關 *)
     Msg1  :=  'Can not save data to file. May be file is open now by Wincon screen ! '  ;
  Else        (*  寫 File 成功, 將 Flag 清除爲 FALSE *)
     Flag_to_save  :=  FALSE  ;
  end_if  ;

end_if  ;
```

ST functions 程式 – G_data :

(* 更新 User Data 變數之值 ,若 SIZE1 值有改, 或 User Data 變數名稱有改, 以下的 Code 也要修改 *)

```
Data1    :=  F_VAL1[0] ;
Data2    :=  F_VAL1[1] ;
Data3    :=  F_VAL1[2] ;
Data4    :=  F_VAL1[3] ;
Data5    :=  F_VAL1[4] ;
Data06   :=  F_VAL1[5] ;
Data07   :=  F_VAL1[6] ;
Data08   :=  F_VAL1[7] ;
Data09   :=  F_VAL1[8] ;
Data10   :=  F_VAL1[9] ;
Data11   :=  F_VAL1[10] ;
Data12   :=  F_VAL1[11] ;
Data13   :=  F_VAL1[12] ;
Data14   :=  F_VAL1[13] ;
Data15   :=  F_VAL1[14] ;
Data16   :=  F_VAL1[15] ;
Data17   :=  F_VAL1[16] ;
G_data  :=  TRUE   ;    (* function 回傳  TRUE *)
```

ST functions 程式 – S_data :

(* 更新 F_VAL1[ ] 之值 , 若 SIZE1 值有改, 或 User Data 變數名稱有改, 以下的 Code 也要修改 *)

```
F_VAL1[0]  :=  Data1;
F_VAL1[1]  :=  Data2;
F_VAL1[2]  :=  Data3;
F_VAL1[3]  :=  Data4;
F_VAL1[4]  :=  Data5;
F_VAL1[5]  :=  Data06;
F_VAL1[6]  :=  Data07;
F_VAL1[7]  :=  Data08;
F_VAL1[8]  :=  Data09;
F_VAL1[9]  :=  Data10;
F_VAL1[10]  :=  Data11;
F_VAL1[11]  :=  Data12;
F_VAL1[12]  :=  Data13;
F_VAL1[13]  :=  Data14;
F_VAL1[14]  :=  Data15;
F_VAL1[15]  :=  Data16;
F_VAL1[16]  :=  Data17;
S_data  :=  TRUE ;    (* function 回傳  TRUE *)
```

ST functions 程式 read_f1 :

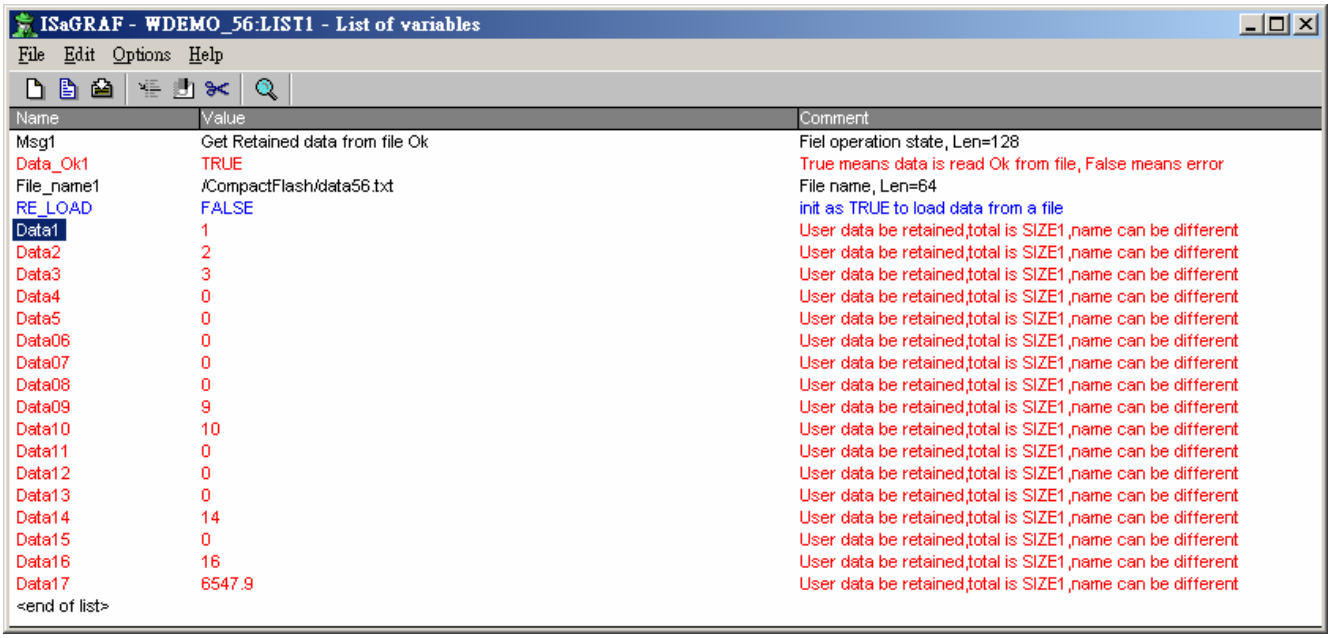
```
(*  這個 function 從 File 內讀出 SIZE1 個實數並存入 F_VAL1[ ]內
    成功: reaf_f1( ) 回傳 TRUE , 失敗回傳 FALSE *)
read_f1 := FALSE ;  (* 一開始先預設爲失敗 *)
File1 := f_wopen( File_name1 ) ;  (* 開啓 File 爲 可讀 / 可寫 模式 *)

if File1 = 0 then  (* 開啓失敗, 通常爲 File 不存在 *)
 return ;  (* 離開此程式 *)
end_if ;

(* File 開啓成功 , 讀資料 *)
for ii2 := 0 to num_row1 – 1 do  (* 共 num_row1 行, 每行 10 個 REAL 值 *)

  if f_eof( File1 ) = TRUE then  (* 偵測是否抵達檔案尾端 *)
    exit ;  (* 若是則離開 for 迴圈 *)
  end_if ;

  str1 := fm_read( File1 ) ;  (* 從 File 內讀出一行字串 *)
  NUM1 := Msg_F( str1 , 1 ) ; (* 將該字串轉換爲數個 REAL 值 , 並存放於 1 號 Float 陣列內 *)

  (* 若爲 最後一行資料數量是否正確 ? 若非最後一行, 資料數量是否爲 10 個 ? *)
  if ( ( ii2 = num_row1 – 1 ) and ( NUM1 <> last_num1 ) ) or
     ( ( ii2 <> num_row1 – 1 ) and ( NUM1 <> 10 ) ) then
    exit ;  (* 數量不對, 離開 for 迴圈 *)
  end_if ;

  if ii2 = num_row1 - 1 then  (* 若爲最後一行, 設 num2 爲 最後一行的資料數量 *)
    num2 := last_num1 ;
  else  (* 若不是最後一行, 設資料數量 num2 爲 10 個 *)
    num2 := 10 ;
  end_if ;

 (* 轉換成功, 將此行的 REAL 值 存入 F_VAL1[ ] 內 *)
  for jj2 := 0 to num2 - 1 do
    F_VAL1[ 10*ii2 + jj2 ] := ARY_F_R( 1 , jj2 + 1 ) ;
  end_for ;
end_for ;

TMP2 := f_close( File1 ) ;  (* File 只要成功開啓過, 處理完就需 close *)
If ii2 = num_row1 then
   read_F1 := TRUE ;  (* num_row1 行資料都已讀到, read_f1( ) 回傳 TRUE *)
end_if ;
```

ST functions 程式 save_f1 :

```
-------------------------------------------------------------------------------------------------------------------------
(*   這個 function 寫 SIZE1 個實數 F_VAL1[ ] 到 File 內,
     成功: save_f1( ) 回傳 TRUE   ,   失敗回傳 FALSE *)

save_f1 := FALSE ;   (* 一開始先預設爲失敗 *)

File1  :=  f_creat( File_name1 )  ;  (* 建立一個新 File, 若檔案已存在, 資料會刪除 *)

if  File1 =  0 then
   return  ;    (* 建立一個新 File 失敗, 離開此程式 *)
end_if  ;

(*建立一個新 File 成功, 準備寫資料進去 *)

for  ii2  :=  0    to     num_row1 - 1     do   (* 共 num_row1 行資料要寫入 *)

 str1   :=   ' '  ;   (* 設每行字串初值爲 1 個空格 *)

 if    ii2   =  num_row1 - 1   then   (* 若爲最後一行, 設 num2 爲 最後一行的資料數量 *)
    num2 := last_num1 ;
 else       (* 若不是最後一行, 設資料數量 num2 爲 10 個 *)
    num2  :=  10  ;
 end_if  ;

  (* 將 REAL 資料 寫成 字串格式 *)
 for   jj2  :=  0   to   num2 - 2    do        (* 不含最後一行 *)
   str1  :=   str1   +   REAL_STR( F_VAL1[ 10 * ii2   +  jj2 ]  )  +  ' , '  ;
 end_for ;

(* 最後一行, 結尾需爲 <CR> <LF> 字元 *)
 str1   :=   str1   +   REAL_STR( F_VAL1[  10  *  ii2   +   9  ]  )   +  '$0D$0A'   ;

 TMP2  :=  f_writ_s( File1  ,  str1 )  ;   (* 將此行字串寫入 File 內 *)

end_for ;

TMP2  :=  f_close( File1 )  ;   (* File 只要成功開啓過, 處理完就需 close *)

save_f1 := TRUE ;   (* num_row1 行資料都已寫入, save_f1( ) 回傳 TRUE *)
-------------------------------------------------------------------------------------------------------------------------
```

本 wdemo_56 範例如何測試？

1. 將 wdemo_56 用 ISaGRAF 下載到 W-8xx7 後，會出現 Spy list 視窗如下



Name	Value	Comment
Msg1	Get Retained data from file Ok	Fiel operation state, Len=128
Data_Ok1	TRUE	True means data is read Ok from file, False means error
File_name1	/CompactFlash/data56.txt	File name, Len=64
RE_LOAD	FALSE	init as TRUE to load data from a file
Data1	1	User data be retained,total is SIZE1,name can be different
Data2	2	User data be retained,total is SIZE1,name can be different
Data3	3	User data be retained,total is SIZE1,name can be different
Data4	0	User data be retained,total is SIZE1,name can be different
Data5	0	User data be retained,total is SIZE1,name can be different
Data06	0	User data be retained,total is SIZE1,name can be different
Data07	0	User data be retained,total is SIZE1,name can be different
Data08	0	User data be retained,total is SIZE1,name can be different
Data09	9	User data be retained,total is SIZE1,name can be different
Data10	10	User data be retained,total is SIZE1,name can be different
Data11	0	User data be retained,total is SIZE1,name can be different
Data12	0	User data be retained,total is SIZE1,name can be different
Data13	0	User data be retained,total is SIZE1,name can be different
Data14	14	User data be retained,total is SIZE1,name can be different
Data15	0	User data be retained,total is SIZE1,name can be different
Data16	16	User data be retained,total is SIZE1,name can be different
Data17	6547.9	User data be retained,total is SIZE1,name can be different
<end of list>		

請更改 任何一個 USER Data 變數之值, 改了後, 它就會自動存入 \CompactFlash\data56.txt 內
所以你可以在 Wincon 的螢幕上開啓 此檔, 會發現它的資料有跟著變動. (請不要一直開著此檔, 要關掉它, 不然之後 資料有變它會寫不進去, 但只要有關掉, 資料就會更新)

2. 將 Wincon 關機後 約 5 秒 再開機, 開完機後您會看到 這些 USER Data 變數之值 會是您最後輸入給它的值.

3. 在 PC 上用 NotePad (記事本) 編輯一個 data56.txt 檔類似如下,

1.1 , 2.2 , 3.3 , 4.4 , 5.5 , 6.66 , 7.77 , 8.88 , 9.99 , 10.01
0.01 , 0.02 , 0.03 , 0.04 , 0.05 , 0.06 , 0.07

將此 data56.txt 檔 用 ftp 傳到 Wincon 的 \CompactFlash\ 目錄 內. 之後在 ISaGRAF 的 Spy list 視窗, 雙擊 RE_LOAD, 把它的值設爲 TRUE. 您會發現 USER Data 變數之值 會更新成 上面的值.

10.5.5: 每 0.05 秒記錄 i-8017H 的 1 ~ 4 個電壓值於 RAM Disk 內的一個檔案, 連續記錄 1~10 分鐘, 之後可在 PC 上開 M.S. Excel 來看 1 ~ 4 條趨勢圖

請參考第 11.3.6 節的說明 (可記錄最快達 20Hz 的資料).

第 11.3.10 節 的方法 可記錄最快達 100Hz的資料. (或 www.icpdas.com – FAQ – Software – ISaGRAF – 057)