

SYNTEC

OpenCNC MACRO 發展工具 操作手冊

By: SYNTEC

Date: 2015/05/28

Version: 9.40

此文件已轉移至 CF 維護，自 2015/06/22 起不再更新。

網址：

<http://confluence.syntecclub.com.tw/pages/viewpage.action?pageId=19793903>

版本更新記錄

項次	更改內容紀錄	更改日期	作者	版本
01	追加版本更新記錄	2004/12/07	吳達欽	V9.0
02	修正#1010 說明	2006/01/25	賴春億	V9.1
03	修正#1504 說明	2006/03/28	林宗賢	V9.2
04	修改文件格式	2010/07/10	Syntec	V9.3
05	新增#1461~1476 說明	2010/9/27	楊勝安	V9.4
05	1. 新增 MACRO 警報/提示說明 2. 修改文字描述	2010/09/29	張宏安	V9.5
06	1. 新增 MACRO XML 資料應用說明 2. 修改函數說明文字描述	2011/07/10	許晃源	V9.6
07	新增#1881~#1896 說明	2011/08/10	謝鎮陽	V9.7
08	修正 MSG 自訂提示觸發語法說明	2011/12/1	張宏安	V9.8
09	修正 Macro 自訂警報檔案位置	2011/12/27	李欣緯	V9.9
10	修正#20001~與#26001 說明	2012/01/05	張宏安	V9.10
11	1. 新增#1834~#1847 說明 2. 新增#1821、#1829、#1831 說明	2012/02/29	李柏瑩	V9.11
12	新增檔案讀取規則	2012/03/22	陳青杉	V9.12
13	新增『/』元件使用注意事項	2012/04/02	張宏安	V9.13
14	1. 英文說明中文化 2. 新增車床刀具補償變數對應 3. 取消檔案讀取規則	2012/06/14	陳丁銓 張宏安	V9.14
15	1. 新增 MSTG 等解譯處理順序 2. 新增登錄 G 碼巨集規格	2012/09/03	陳青杉	V9.15
16	1. 擴充#1504 使用規格	2012/11/21	陳青杉	V9.16
17	修改 SETDRAW 規格	2012/12/13	陳丁銓	V9.17
18	修改 SETDRAW 規格	2013/01/23	張介豪	V9.18
19	修改車床刀具補償變數對應	2013/05/03	高仰逸 張宏安	V9.19
20	1. 加強多國語言編輯檔案與目錄說明	2013/06/04	李朝修	V9.20
21	修改#1941~1976 的說明	2013/06/28	陳丁銓	V9.21
22	1. 修改#1504/#1512/#7001~說明 2. 新增系統變數型態說明	2013/07/07	張宏安	V9.22
23	新增#1826 說明	2013/10/29	李哲維	V9.23
24	1. 修改#1803~#1807 說明	2013/11/25	吳長壽	V9.24

25	1. 於函式表增加專案加密檢查函式	2013/12/20	洪培堯	V9.25
26	修改函數 PARAM 的內容，刪掉"特定"兩字	2013/12/23	陳丁銓	V9.26
27	修改#1881 的說明	2013/12/26	陳丁銓	V9.27
28	修改 MACRO GETARG 指令說明	2014/02/11	施禹廷	V9.28
29	新增 Macro 撰寫注意事項引數與區域變數關係(L, #12)	2014/02/14	簡文昱	V9.29
30	修改區域變數#1~#26 說明	2014/03/11	張宏安	V9.30
31	擴充銑床刀具補償變數說明	2014/06/26	張宏安	V9.31
32	修正函數 COS 的內容，#1 = -1	2014/07/09	高仰逸	V9.32
33	新增 Macro 指令與註解語法補充說明	2014/07/10	施禹廷	V9.33
34	擴充登錄 G 碼相關說明	2014/08/18	陳青杉	V9.34
35	新增#1517 說明	2014/08/29	王芝峰	V9.35
36	刪除模式群組變數及#1510	2015/3/23	李建明	V9.36
37	修改 9.3 節擴充 G 碼檔案命名規則內容	2015/4/14	何柏諺	V9.37
38	新增函數表 READDI、READDO 及 SETDO 功能說明	2015/4/20	謝旻翰	V9.38
39	修改#1822 說明	2015/04/30	王亭諭	V9.39
40	修改 STD 說明	2015/05/28	江禹安	V9.40

Contents

1	前言.....	1
2	檔案格式.....	1
3	指令格式(Block Format).....	2
4	運算子(Operator).....	4
5	語法說明.....	5
5.1	變數指定.....	5
5.2	GOTO	6
5.3	CASE	7
5.4	IF	8
5.5	REPEAT	9
5.6	WHILE.....	10
5.7	FOR.....	11
5.8	EXIT	12
5.9	程式註解(Comment)	13
6	函數表(Functions Listing).....	14
7	副程式呼叫.....	23
7.1	呼叫方式(Call Method).....	23
7.2	返回方式(Return Methods).....	24
8	變數規格.....	25
8.1	公用變數(Global variable)	25
8.2	區域變數(Local variables)	25
8.3	系統變數(System variables).....	26
8.3.1	程式資訊(Modal information, #1000~#1056)	26
8.3.2	操作控制/狀態變數(Operation control/status, #1500~#1624)	27
8.3.3	目前位置(Current position, #1301~#1476).....	28
8.3.4	運行狀態(Runtime state, #1800~#1976).....	29
8.3.5	模態變數(Modal variables, #1080~#3100).....	30
8.3.6	客制參數(Customer param., #4001~#5500)	31
8.3.7	介面訊號(Interface signals, #6001~#6032)	31
8.3.8	刀具補償(Tool compensation, #10000~#15288).....	31
8.3.9	工件座標系統(Workpiece coordinate system offset value, #20001~#20656).....	32
8.3.10	參考點位置(Reference point position, #26001~#26076)	33
9	擴充 G 碼使用說明.....	34
9.1	MACRO 讀取處理流程.....	34
9.2	MACRO 撰寫注意事項.....	35

9.3	擴充 G 碼檔案命名規則與儲存路徑.....	36
9.4	登錄 G 碼相關說明及規則.....	36
10	MACRO 自訂警報方式.....	38
10.1	MACRO 警報觸發語法.....	38
10.2	DOS 版本警報內容編輯說明.....	38
10.3	WinCE 版本警報內容編輯說明.....	38
11	MACRO 自訂提示方式.....	39
11.1	規格說明.....	39
11.2	MSG 自訂提示觸發語法.....	39
12	附件.....	41
12.1	Registry 資源分配表.....	41
12.2	基本 G 碼指令一覽表.....	42
12.3	MACRO 範例.....	44
12.4	MACRO XML 資料應用.....	47

1 前言

為增加控制器應用彈性，新代控制器提供 MACRO 程式編輯功能。當加工程式被宣告成 MACRO 格式後，該檔案將如同一般程式語言，可進行特定數學函數使用，如此一來，加工程式將不再只有單純具有的移動或補償指令功能，更能擁有邏輯判斷以及數學演算功能。

2 檔案格式

程式內容第一行需使用 ‘%’ 宣告為標題行，並加入關鍵字 ‘@MACRO’，否則該檔案將被視為一般 ISO 格式檔處理，因而本手冊中所描述之部分功能將無法使用，此外每一行結束需加**分號；**。

範例一：MACRO 格式檔

```
% @MACRO
IF @1 = 1 THEN
    G00 X100.;
ELSE
    G00 Y100.;
END_IF;
M30;
```

範例二：ISO 格式檔

```
% 這是標題行，可當作檔案用途說明，此行可有可無
G00 X100.;
G00 Y100.;
G00 X0;
G00 Y0;
M30;
```

3 指令格式(Block Format)

單行動作控制指令的撰寫格式敘述如下。

/	N	G	X	Y	Z	A	B	C	I	J	K	F	S	T	D	M
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

/	單節選擇性跳躍功能，需配合 PLC C41
N	單節次序碼，必須撰寫在該單節的第一碼位置
G	功能指定碼，需撰寫在 N 碼之後
X	X 軸的移動命令，或是擴充 G 碼的引數，需撰寫在 G 碼後
Y	Y 軸的移動命令，或是擴充 G 碼的引數，需撰寫在 G 碼後
Z	Z 軸的移動命令，或是擴充 G 碼的引數，需撰寫在 G 碼後
A	A 軸的移動命令，或是擴充 G 碼的引數，需撰寫在 G 碼後
B	B 軸的移動命令，或是擴充 G 碼的引數，需撰寫在 G 碼後
C	C 軸的移動命令，或是擴充 G 碼的引數，需撰寫在 G 碼後
I	X 方向的半徑命令，或是擴充 G 碼的引數，需撰寫在 G 碼後
J	Y 方向的半徑命令，或是擴充 G 碼的引數，需撰寫在 G 碼後
K	Z 方向的半徑命令，或是擴充 G 碼的引數，需撰寫在 G 碼後
F	單節進給速度，或是擴充 G 碼的引數
S	主軸旋轉速度，或是擴充 G 碼的引數
T	刀具選擇功能，或是擴充 G 碼的引數
D	刀具補償功能，或是擴充 G 碼的引數
M	輔助功能，或是擴充 G 碼的引數

解譯處理順序(1.最先~10.最後)：

1. 模態 G 碼(G15、G17、G70 等)、擴充 G 碼巨集(G73、G84 等)
2. M 碼巨集、T 碼巨集
3. S 碼
4. F 碼
5. H 碼
6. D 碼
7. T 碼
8. M 碼
9. B 碼
10. 差值 G 碼(G0、G1 等)、功能 G 碼(G4、G51、G68 等)

註：

1. 其餘未說明之格式由相關 G 碼以引數帶入
2. 一般在副程式中使用 GETARG 讀取傳入引數之情況，於主程式中可傳入之引數形式：

(1) 使用引數 D、E、H、I、J、K、L、M、P、Q、R、T

僅可以使用單一符號方式傳入引數值，例如「G101 X30. Y40. Z1=40. D50.;」亦即若在其後附帶參數將會引發警報，例如執行至「G101 X30. Y40. Z1=40. D1=50.;」出現警報

(2) 使用引數 A、B、C、F、S、U、V、W、X、Y、Z

除可以使用單一符號方式傳入引數值之外，亦可附帶數字後傳入，例如「G101 X30. Y40. Z50.;」、「G101 X30. Y40. Z1=50.;」

(3) 以上所有單節動作指令之後，僅可使用數值或存值為數值之變數

否則可能會因程式解譯編碼限制而造成系統錯誤，此誤用情況屬於控制器能夠保護的範圍之外。

4 運算子(Operator)

運算子	符號	執行順序
括號	() []	1
函數賦值	Identifier (引數)	2
負號	-	3
補數	NOT	3
乘號	*	4
除號	/	4
模數	MOD	4
加號	+	5
減號	-	5
比較	<, >, <=, >=	6
等於	=	7
不等於	<>	8
布林運算”且”	&, AND	9
布林運算”互斥”	XOR	10
布林運算”或”	OR	11

備註：

請參閱以下範例說明，『/』元件使用上需注意分子與分母都是整數時，所得結果仍為整數。

$$2. / 2 = 0.5$$

$$1 / 2.0 = 0.5$$

$$1 / 2 = 0$$

$$(1 / 2) * 1.0 = 0$$

5

語法說明

5.1 變數指定

語法：<變數> := <敘述>;

說明：指定變數內容

範例一：直接定值

@1 := 123;

#1 := 456;

#10 := "12"; // 區域變數 #10 內容為 12

@10 := "12"; // 公用變數 @10 內容為 12849

PS：將字串存入公用變數中，控制器會若先進行 ASCII 轉碼，區域變數則不會，因此欲正確讀取公用變數中所儲存的字串內容，請使用 **SCANTEXT** 函數。

範例二：間接定值

#1 := 123;

@[#1] := 567; // @123=567

@[#1+7] := 890; // @130=890

5.2 GOTO

語法：**GOTO n;**

說明：跳到指定的 N 行號執行。假設程式中同時存在兩個 N 行號，則以該程式中第一個 N 行號為準。

範例一：

```
% @MACRO
#1 := 1;
#2 := 10;
G01 G90 X0. Y0. F1000;
IF( #1 = 1 ) THEN
    GOTO #2;
END_IF;
IF( #1 = 2 ) THEN
    GOTO 100;
END_IF;
N10
G01 G90 X50. Y0. F1000;
M30;
N100
G01 G90 X0. Y50. F1000;
M30;
```

5.3 CASE

語法：

CASE <條件變數> **OF**

 <整數>:

 <陳述列表>

 <整數>, <整數>:

 <陳述列表>

 <整數>, <整數>, 整數>:

 <陳述列表>

ELSE

 <陳述列表>

END_CASE;

說明：CASE 多條件判斷，根據條件變數內容，分別執行不同程式區塊。請注意變數內容需為**整數**型態。

範例：

```
% @MACRO
#1 := 1;
G01 G90 X0. Y0. F1000;
CASE #1 OF
1:
    X(1.0*#1) Y(1.0*#1);
2:
    X(2.0*#1) Y(2.0*#1);
3, 4, 5:
    X(3.0*#1) Y(3.0*#1);
ELSE
    X(4.0*#1) Y(4.0*#1);
END_CASE;
M30;
```

5.4 IF

語法：

```
IF <條件> THEN  
    <陳述列表>  
ELSEIF <條件> THEN  
    <陳述列表>  
ELSE  
    <陳述列表>  
END_IF;
```

說明：IF 條件判斷

範例：

```
% @MACRO  
#1 := 3.0;  
G01 G90 X0. Y0. F1000;  
IF #1 = 1 THEN  
    X(1.0*#1) Y(1.0*#1);  
ELSEIF #1 = 2 THEN  
    X(2.0*#1) Y(2.0*#1);  
ELSEIF #1 = 3 THEN  
    X(3.0*#1) Y(3.0*#1);  
ELSE  
    X(4.0*#1) Y(4.0*#1);  
END_IF;  
M30;
```

5.5 REPEAT

語法：

```
REPEAT
  <陳述列表>
UNTIL <條件>
END_REPEAT;
```

說明：REPEAT 迴圈控制

範例：

```
% @MACRO
#10 := 30.;
#11 := 22.5.;
#12 := #10/2;
#13 := #11/2;
#14 := 2.0;
#15 := 1.5;
G01 G90 X#12 Y#13 F1000;
REPEAT
  G00 X(#12+#14) Y(#13+#15);
  G01 X(#12+#14) Y(#13-#15);
  G01 X(#12-#14) Y(#13-#15);
  G01 X(#12-#14) Y(#13+#15);
  G01 X(#12+#14) Y(#13+#15);
  #14 := #14 + 2.0;
  #15 := #15 + 1.5;
UNTIL (#14 > #12) OR (#15 > #13) END_REPEAT;
M30;
```

備註：使用 REPEAT/WHILE/FOR 等迴圈功能時，應謹慎注意無窮迴圈問題，當此問題發生時，人機畫面將鎖死無法操作。建議在迴圈中適時加入 **SLPPE()** 函數，可在發生無窮迴圈時，仍可操作人機畫面以中止程式執行。

5.6 WHILE

語法：

```
WHILE <條件> DO  
    <陳述列表>  
END_WHILE;
```

說明：WHILE 迴圈控制

範例：

```
% @MACRO  
#10 := 30.;  
#11 := 22.5.;  
#12 := #10/2;  
#13 := #11/2;  
#14 := 2.0;  
#15 := 1.5;  
G01 G90 X#12 Y#13 F1000;  
WHILE (#14 <= #12) AND (#15 <= #13) DO  
    G00 X(#12+#14) Y(#13+#15);  
    G01 X(#12+#14) Y(#13-#15);  
    G01 X(#12-#14) Y(#13-#15);  
    G01 X(#12-#14) Y(#13+#15);  
    G01 X(#12+#14) Y(#13+#15);  
    #14 := #14 + 2.0;  
    #15 := #15 + 1.5;  
END_WHILE;  
M30;
```

備註：使用 REPEAT/WHILE/FOR 等迴圈功能時，應謹慎注意無窮迴圈問題，當此問題發生時，人機畫面將鎖死無法操作。建議在迴圈中適時加入 **SLPPE()函數**，可在發生無窮迴圈時，仍可操作人機畫面以中止程式執行。

5.7 FOR

語法：

```
FOR <變數 1> := 敘述 1> TO <敘述 2> BY <敘述 3> DO  
  <陳述列表>
```

```
END_FOR;
```

說明：FOR 迴圈控制

變數 1：控制迴圈次數的變數

敘述 1：迴圈計數的起始次數，可為數值或運算式

敘述 2：迴圈計數的終止次數，可為數值或運算式

敘述 3：迴圈計數每次的累加次數，可為數值或運算式

陳述列表：迴圈每次執行內容

範例：

```
% @MACRO  
#10 := 30;  
#11 := 22.5;  
#12 := #10/2;  
#13 := #11/2;  
#14 := 2.0;  
#15 := 1.5;  
G01 G90 X#12 Y#13 F1000;  
FOR #6 := 0 TO 3 BY 1.0 DO  
  G00 X(#12+#14) Y(#13+#15);  
  G01 X(#12+#14) Y(#13-#15);  
  G01 X(#12-#14) Y(#13-#15);  
  G01 X(#12-#14) Y(#13+#15);  
  G01 X(#12+#14) Y(#13+#15);  
  #14 := #14 + 2.0;  
  #15 := #15 + 1.5;  
END_FOR;  
M30;
```

備註：使用 REPEAT/WHILE/FOR 等迴圈功能時，應謹慎注意無窮迴圈問題，當此問題發生時，人機畫面將鎖死無法操作。建議在迴圈中適時加入 **SLPPE()函數**，可在發生無窮迴圈時，仍可操作人機畫面以中止程式執行。

5.8 EXIT

語法：**EXIT;**

說明：迴圈中斷，跳離迴圈控制

範例：

```
% @MACRO
#10 := 30.;
#11 := 22.5.;
#12 := #10/2;
#13 := #11/2;
#14 := 2.0;
#15 := 1.5;
#16 := 1.0;
G01 G90 X#12 Y#13 F1000;
FOR #6 := 0 TO 3 BY 1.0 DO
    IF((#14 = 4) & (#16 = 1)) THEN
        EXIT;
    END_IF;
    G00 X(#12+#14) Y(#13+#15);
    G01 X(#12+#14) Y(#13-#15);
    G01 X(#12-#14) Y(#13-#15);
    G01 X(#12-#14) Y(#13+#15);
    G01 X(#12+#14) Y(#13+#15);
    #14 := #14 + 2.0;
    #15 := #15 + 1.5;
END_FOR;
M30;
```

5.9 程式註解(Comment)

語法：

(* <陳述列表> *)

// <陳述列表>

說明：程式註解

範例一：單行註解

% @MACRO

G00 G90 X0. Y0.; // 移動回原點

M30;

範例二：區塊註解

% @MACRO

(*

此區塊為註解區，

不管內容為何並不會影響以下程式執行。

*)

G00 G90 X0. Y0.;

G00 G90 X10. Y0.;

G00 G90 X10. Y10.;

G00 G90 X0. Y10.;

G00 G90 X0. Y0.;

M30;

註：

若於符合註解語法以外的陳述列表區域添加屬於註解之文字，可能會因解譯編碼之限制造成系統錯誤，此誤用之情況屬於系統能夠保護的範圍之外。

6 函數表(Functions Listing)

功能	說明
ABS	計算某數值的絕對值 Ex: #10 := -1.1; #1 := ABS(#10); // #1 = 1.1 #2 := ABS(-1.2); // #2 = 1.2
ACOS	計算某數值的反餘弦值 Ex: #10 := 1; #1 := ACOS(#10); // #1 = 0 #2 := ACOS(-1); // #2 = 180
ALARM	觸發巨集警報 Ex: ALARM(300); // 觸發巨集第 300 號警報 ALARM(301, "ALARM 301 Content"); 警報內容有字串長度限制 中文：19 個字 英文：39 個字
ASIN	計算某數值的反正弦值 Ex: #10 := 1; #1 := ASIN(#10); // #1 = 90 #2 := ASIN(-1); // #2 = -90
ATAN	計算某數值的反正切值 Ex: #10 := 1; #1 := ATAN(#10); // #1 = 45 #2 := ATAN(-1); // #2 = -45
AXID	查詢軸名稱所對應的軸編號，當該軸名稱不存在時，回傳值為空白(VACANT, #0) Ex: 假設第六軸名稱為 Y2(Pr326=202)，第二軸名稱為 Y(Pr322=200) #1 := AXID(Y); // #1 = 2 #2 := AXID(Y2); // #2 = 6
CEIL	回傳大於或等於某數值的最小整數 Ex: #10 := 1.4; #1 := CEIL(#10); // #1 = 2 #2 := CEIL(1.5); // #2 = 2
CLOSE	關閉前面由 OPEN 函數所開啟的檔案，程式結束後該檔

功能	說明
	案亦會自動關閉。檔案關閉後，PRINT 函數就無效。 例： CLOSE(); // 關閉檔案
COS	計算某數值的餘弦值 Ex: #10 := 180; #1 := COS(#10); // #1 = -1 #2 := COS(-180); // #2 = -1
DBLOAD	從目前載入的 XML 資料表中，讀取指定索引的資料， 相關應用請參考附件。 例： DBOPEN("FLAT\\TAB01"); // 載入 FLAT\\TAB01 資料檔 DBLOAD(0); // 讀取第 0 筆資料 DBLOAD(1); // 讀取第 1 筆資料
DBOPEN	載入指定的 XML 資料表。XML 資料表路徑位於使用者 所指定的 GNCFILES 位置，相關應用請參考附件。 例： DBOPEN("Test"); // 載入 GNCFILES\\Test 資料檔 例： #1 = 51; DBOPEN("FLAT\\AB#1[3]ZZ"); // 載入 FLAT\\AB051ZZ 資料檔，[3]表示檔名為三 位有效數字
DRAWHOLE	依據刀具半徑以及 SETDRAW 函數所定義之顏色，在目 前位置畫一個圓(只在圖形模擬內有效，而不是真的在實 際軌跡加走一個圓)
DRAWMARK(形狀,尺寸,顏色)	以特定形狀、尺寸以及顏色，在目前位置畫一個標記， 使用放大縮小功能也不能改變此標記的大小(只在圖形 模擬內有效，而不是真的在實際軌跡加走一個標記)，其 中 形狀：0 圓形，1 方形，2 菱形 尺寸：單位為畫素
FLOOR	回傳小於或等於某數值的最大整數 Ex: #10 := 1.4; #1 := FLOOR(#10); // #1 = 1 #2 := FLOOR(1.5); // #2 = 1
GETARG	讀取呼叫者傳遞的引數 例： 假設 O0001 主程式內容為

功能	說明
	G101 X30. Y40. Z1=40. Z2=50.; G0101 擴充巨集程式使用 GETARG 讀取引數內容 #1 := GETARG(X); // 將 X 引數內容 30.存到#1 #2 := GETARG(Z1); // 將 Z1 引數內容 40.存到#2 #3 := GETARG(W); // 因 W 不存在，所以#3 內容為(VACANT, #0) 於主程式中可傳入之引數形式： 1. 使用引數 D、E、H、I、J、K、L、M、P、Q、R、T 僅可以使用單一符號方式傳入引數值 例： G101 X30. Y40. Z1=40. D50.; // 將引數 D 內容 50.傳入 亦即若在其後附帶參數將會引發警報 例： G101 X30. Y40. Z1=40. D1=50.; // 執行至此出現警報 2. 使用引數 A、B、C、F、S、U、V、W、X、Y、Z 除可以使用單一符號方式傳入引數值之外，亦可附帶數字後傳入 例： G101 X30. Y40. Z50.; // 將引數 Z 內容 50.傳入 G101 X30. Y40. Z1=50.; // 將引數 Z1 內容 50.傳入
GETTRAPARG	讀取 G66/G66.1 在 Trap 單節內的引數內容 例： 假設 O0001 主程式內容為 G66 P100 X100. Y100. G01 X20. O0100 副程式使用 GETTRAPARG 讀取引數內容 #1 := GETARG(X); // 將 X 引數內容 100.存到#1 #2 := GETTRAPARG(X); // 將 Trap 單節 X 引數內容 20.存到#2
MAX	決定兩輸入值的最大值 Ex: #10 := 1.2; #20 := 4.5; #1 := MAX(#10, #20); // #1 = 4.5 #2 := MAX(-1.2, -4.5); // #2 = -1.2
MIN	決定兩輸入值的最小值 Ex: #10 := 1.2; #20 := 4.5; #1 := MIN(#10, #20); // #1 = 1.2 #2 := MIN(-1.2, -4.5); // #2 = -4.5
MSG	自訂提示，詳情請參閱『MACRO 自訂提示方式』

功能	說明
	Ex: <pre>MSG(100); // 警報 ID MSG("鑽頭移失"); // 警報顯示內容 MSG(100, "鑽頭遺失"); // 提示 ID + 顯示內容</pre> 警報內容有字串長度限制 中文：19 個字 英文：39 個字
OPEN	開啟一文字檔，其檔名為所指定之名稱。需在檔案開啟後，PRINT 函數才有效。 若檔案名稱為“COM”時，表示打開 RS232 傳輸埠，其設定由系統 Pr3905 決定。 例： <pre>OPEN("PROBE.NC"); // 開啟 PROBE.NC 準備作資料輸出</pre> 例： <pre>OPEN("COM"); // 打開傳輸埠 PRINT("\p"); // 輸出'%'字元 FOR #1 = 1 TO 5000 DO #30 := #1 * 10.; PRINT("G01 X#30"); // 輸出 G01 X10.0 END_FOR; PRINT("\p"); // 輸出'%'字元 CLOSE(); // 關閉傳輸埠</pre>
PARAM	讀取系統參數的內容 Ex: <pre>#1 := PARAM(3204); // 讀取 Pr3204(PLC 掃描時間)之內容</pre>
POP	將堆疊(STACK)裡面的資料取出，依序由最上層一路取到最底層，使用者須注意堆疊裡面共有幾筆資料，有 5 筆資料就只能使用 5 次 POP。 Ex: <pre>PUSH(5); // 將數字 5 塞入堆疊中 #1 := POP(); // 取出堆疊中最上層數值(#1 = 5)</pre>
PRINT	此函數用來輸出字串，輸出字串中的變數名稱會被取代成該變數之內容。 字元‘\’為逃脫字元,特殊字元定義如下： <pre>'\\' 表示'\ '字元 '\@' 表示'@'字元 '\#' 表示'#'字元 '\p' 表示'%'字元</pre> 例： <pre>@53 = 20 ; #3 = 23.1 ; PRINT(« G01 X#3 Y@53 Z20.0 »);</pre>

功能	說明
	輸出結果為 G01 X23.100 Y20 Z20.0 ; 例： PRINT("G01 X(\@20^\@30) Y\#20/2.0"); 輸出結果為 G01 X(@20/@30) Y#20/2.0 ;
PUSH	將資料塞進堆疊裡(STACK)，最先 PUSH 進入控制器的數值會堆疊在最底層，並依序往上疊加。 Ex: PUSH(#1); // 將變數#1 塞入堆疊中 PUSH(#3); // 將數字#3 塞入堆疊中
RANDOM	產生一個隨機數 Ex: #1 := RANDOM();
READDI (I 點編號)	以 READDI、READDO 指令後方小括弧內的數字，決定所讀取的 I/O 點編號。 範例： @52 := READDI(31); // 把 I31 的值讀出來填到@52 #88 := READDO(11); // 把 O11 的值讀出來填到#88 G90 G10 L1000 P4000 R READDI(15); // 把 I15 的值讀出來填到 R4000 <u>注意事項：</u> 1. 有效版本始於 10.116.23。 2. 為了避免因預解問題造成 I/O 點讀取錯誤，系統內部會自行等待，因此在使用 READDI、READDO 指令時，會有減速到零的情形。 3. 讀取的 I/O 點編號範圍僅限於 0~511 間，若範圍錯誤，系統將發出【COR-138 I/O 點讀寫指令格式錯誤”警報】。
READDO(O 點編號)	
ROUND	回傳某數值完成四捨五入進位後的值 Ex: #10 := 1.4; #1 := ROUND(#10); // #1 = 1 #2 := ROUND(1.5); // #2 = 2
SCANTEXT	此函數用來讀取公用變數中所儲存的字串內容。由於將字串存入公用變數中，控制器會若先進行 ASCII 轉碼，因此若直接輸出將會得到錯誤的結果，故需使用此函數以取得正確字串。 例： % @MACRO @1:=”12”; #1:=SCANTEXT(1); OPEN(“NC”); PRINT(“@1”); PRINT(“#1”); CLOSE(); M30;

功能	說明																
	所得結果為 @1 = 12849 #1 = 12																
SETDO(O 點編號, O 點開或關)	以 SETDO 指令後方小括弧內的第一、二個數字，決定所設定的 O 點編號，以及該 O 點開或關(1：開；2：關)。 範例： SETDO(3, 1); // 設定 O3 on SETDO(8, 0); // 設定 O8 off <u>注意事項：</u> 1. 有效版本始於 10.116.23。 2. PLC 與 SETDO 避免不要混用，例如 MACRO 中以 SETDO 讓 O1 on，但卻在 PLC 中讓 O1 off。目前的自然規格是依照指令處理的順序後令蓋前令，但可能在應用上可能不好掌握，因此建議 PLC 與 SETDO 不要混用。 3. 設定的 O 點編號範圍僅限於 0~511 間，若範圍錯誤，系統將發出【COR-138 I/O 點讀寫指令格式錯誤”警報】。																
SETDRAW(路徑顏色) or SETDRAW(路徑顏色, 填充顏色, 刀具半徑)	定義圖形模擬的畫圖樣式： 1.路徑顏色：設定輪廓線的顏色，可使用 RGB 碼，或參考模擬參數設定裡面的顏色代碼來進行設定。 2.填充顏色：設定 DRAWHOLE 函數圓內填充的顏色，可使用 RGB 碼，或參考模擬參數設定裡面的顏色代碼來進行設定。 3.刀具半徑：設定 DRAWHOLE 函數圓半徑大小。 *影響包含有使用刀具半徑的 G 碼如 G01 常用 RGB 碼如下所示： <table border="0"> <tbody> <tr> <td>0: 0</td><td>1: 8388608</td></tr> <tr> <td>2: 32768</td><td>3: 8421376</td></tr> <tr> <td>4: 128</td><td>5: 8388736</td></tr> <tr> <td>6: 32896</td><td>7: 12632256</td></tr> <tr> <td>8: 8421504</td><td>9: 16711680</td></tr> <tr> <td>10: 65280</td><td>11: 16776960</td></tr> <tr> <td>12: 255</td><td>13: 16711935</td></tr> <tr> <td>14: 65535</td><td>15: 16777215</td></tr> </tbody> </table> <u>注意事項：</u> SETDRAW 會同時設定到路徑以及畫圓的顏色，所以若想让路徑以及畫圓的顏色有所區別，要記得在執行完 DRAWHOLE 之後，再下一次 SETDRAW，將路徑顏色改回來。	0: 0	1: 8388608	2: 32768	3: 8421376	4: 128	5: 8388736	6: 32896	7: 12632256	8: 8421504	9: 16711680	10: 65280	11: 16776960	12: 255	13: 16711935	14: 65535	15: 16777215
0: 0	1: 8388608																
2: 32768	3: 8421376																
4: 128	5: 8388736																
6: 32896	7: 12632256																
8: 8421504	9: 16711680																
10: 65280	11: 16776960																
12: 255	13: 16711935																
14: 65535	15: 16777215																

功能	說明
	Ex : %@MACRO #3:=SETDRAW(#1,#2,#18); // 使用#3 記錄原本路徑顏色，#2 定義填充顏色，#18 定義畫圓半徑 DRAWHOLE(); SETDRAW(#3); // 畫圓後將路徑顏色改回 M99;
SIGN	回傳某數值的正負號，負數為-1，正數為1，0 為 0 Ex: #10 := 4; #1 := SIGN(#10); // #1 = 1 #2 := SIGN(-4); // #2 = -1 #3 := SIGN(0); // #3 = 0
SIN	計算某數值的正弦值 Ex: #10 := 90; #1 := SIN(#10); // #1 = 1 #2 := SIN(-90); // #2 = -1
SLEEP	暫時放棄此次巨集循環的執行權，一般用於配合迴圈 (FOR, WHILE...)使用，以避免無窮迴圈使用而造成人機 卡死 Ex: SLEEP();
SQRT	計算某數值的平方根值 Ex: #10 := 4; #1 := SQRT(#10); // #1 = 2 #2 := SQRT(9); // #2 = 3
STD(引數 1,引數 2)	根據參數 17 基本單位長度單位(BLU)，將數值轉成 IU 的單位。 1. 引數 1 為欲改變單位之數值 2. 引數 2 為標準化單位，一般使用#1600，#1600 設定 值來自參數 17 Ex: //當 Pr17=2, BLU = 0.001mm #9 := 100; #10 :=STD(#9,#1600); // #9 為 100.000，#10 為 0.1mm
STDAX(引數 1,引數 2)	將數值轉成對應軸向的標準單位其中 引數 1 為某數值 引數 2 為對應軸像的名稱 Ex: #24 := STDAX(#24,X);

功能	說明
	#3 := STDAX(#3,A);
STKTOP	將堆疊(STACK)裡面的資料複製出來。 Ex: PUSH(5); // 將數字 5 塞入堆疊中 PUSH(6); // 將數字 6 塞入堆疊中 PUSH(7); // 將數字 7 塞入堆疊中 #1 := STKTOP[0]; // #1 = 7 #2 := STKTOP[1]; // #2 = 6 #3 := STKTOP[2]; // #3 = 5
SYSVAR(軸群識別碼, 系統變數碼)	讀取特定軸群中的的系統變數，其中 軸群識別碼：1 為第一軸群組，2 為第二軸群組，以此類推。 系統變數碼：欲讀取的系統變數 Ex: #1 := SYSVAR(1, 1000); // 讀取第一軸群的系統變數#1000(差值模式)
TAN	計算某數值的正切值 Ex: #10 := 45; #1 := TAN(#10); // #1 = 1 #2 := TAN(-45); // #2 = -1
WAIT	系統停止預解，直到 WAIT 前面指令執行完畢。 例： % @MACRO N1 G10 L1000 P50 R1; // R50 內容填成 1 N2 G94 G01 X100. F1; // 假設此時 Reset 系統 N3 WAIT(); N4 G10 L1000 P50 R0; // R50 內容填成 0 N5 M30; 假設在 N2 單節移動中重置系統，由於 N3 存在 WAIT 關係，因此系統於 N3 停止預解，故重置後 R50 內容仍是 1。
CHKMN(“機械廠代碼”)	檢查機械廠代碼，TRUE：一致 FALSE：不符 例： % @MACRO #51 := CHKMN(“5566”); // #51 之值為檢查結果 IF #51=0 THEN // 若檢查代碼不符發出警報 ALARM(501, “The manufacturer code is invalid.”); END_IF (目標版本:10.116.6A)
CHKSN(“序號”)	檢查控制器序號，TRUE：一致 FALSE：不符 例： % @MACRO #52 := CHKSN(“M9A0001”); // #52 之值為檢查結果

功能	說明
	IF #52=0 THEN //若檢查代碼不符發出警報 ALARM(502, "The serial number is invalid."); END_IF (目標版本:10.116.6A)
CHKMT(“機床屬性”)	檢查機床屬性，TRUE：一致 FALSE：不符 例： %@MACRO #53 := CHKMT("MILL"); // #53 之值為檢查結果 IF #53=0 THEN //若檢查代碼不符發出警報 ALARM(503, "The machine type is invalid."); END_IF (目標版本:10.116.6A)
CHKMI(“機型”)	檢查控制器機型，TRUE：一致 FALSE：不符 機型除了 SUPER 系列在此需輸入 s 以外，其他機型皆按照型號輸入，如 10B 則輸入 10B，11A 則輸入 11A。 例： %@MACRO #54 := CHKMI("s"); // #54 之值為檢查結果 IF #54=0 THEN //若檢查代碼不符發出警報 ALARM(504, "The hardware type is invalid."); END_IF (目標版本:10.116.6A)
STR2INT(“字串”)	轉換數字字串為整數型態 例 1： %@MACRO @1:="5555"; #1:= SCANTEXT(1); // #1 = 字串 5555 #2:= STR2INT("#1"); // #2 = 5555 例 2： %@MACRO #1:=STR2INT("100"); // #1 = 100 例 3： %@MACRO #1:=STR2INT("123.456"); // #1 = 123 注意：只要有文字的部分即為不合法之輸入

7 副程式呼叫

7.1 呼叫方式(Call Method)

Syntax	Description	Examples
M98 P_ H_ L_	呼叫副程式 P_ 副程式名稱 H_ 起始序號 L_ 重覆次數	範例： M98 P10 L2; 說明： 呼叫 O0010 兩次
G65 P_ L_	呼叫單一巨集程式 P_ 副程式名稱 L_ 重覆次數	範例： G65 P10 X10.0 Y10.0; 說明： 呼叫 O0010 一次
G66 P_ L_	使用移動指令來呼叫巨集程式 P_ 副程式名稱 L_ 重覆次數	範例： G66 P10 X10.0 Y10.0; X20. Y20. 說明： X20.與 Y20.移動指令會呼叫 O0010
G66.1 P_ L_	每個單節皆會呼叫巨集程式 P_ 副程式名稱 L_ 重覆次數	範例： G66.1 P10 X10.0 X20. G04 X2.; M30; 說明： X20, G04 X2.與 M30 每一單節指令都會呼叫 O0010
G_ L_	呼叫擴充 G 碼 L_ 重覆次數	範例： G128 L3 X1.0; 說明： 呼叫 G0128 三次
T_	呼叫副程式 T0000 來變換刀具，在副程式內的 T 碼為普通 T 碼，將不會再呼叫 T0000	範例： T3; 說明： 呼叫 T0000
M_	呼叫 M 碼巨集程式，在巨集內的 M 碼為普通 M 碼，將不會再呼叫 M 碼巨集	M13 A_ B_ C_; 呼叫 M0013 之巨集程式。 使用前必需先在 Pr3601~內登錄。

■ 上述之 L 引數未指定時，系統內定預設值為 1。

7.2 返回方式(Return Methods)

Syntax	Description	Examples
M99	返回主程式	M99;
M99 P_	返回主程式特定序號 P_ 欲到達的 序號 編號	M99 P100; 回到父程式序號 N100 位置
M99 Q_	返回主程式特定行號 Q_ 欲到達的 行號 編號	M99 Q100; 回到父程式行號 100 行位置
G67	取消 G66	G67;

8 變數規格

MACRO 可使用的變數主要可分為區域變數(Local variable, #1~#400)、系統變數(System variable, #1000~#31986)，以及公用變數(Global variable, @1~@165535)三種。不同型態的變數將有其不同的生命週期，以及讀寫規則，以下各章節將有更詳細說明。

8.1 公用變數(Global variable)

編號	說明	規則	型態
@0	空白 VACANT	R	-
@1~@400	一般運算用變數	R/W	Double
@401~@655	對應 Registry R1~R255	R/W	Long
@656~@1999	有記憶性變數(關機後仍存在)	R/W	Double
@10000~@14095	對應 Registry R0~R4095	R/W	Long
@60000~@79999	擴充公用變數(請配合 No3813 使用，此外僅適用於 CE 系統)	R/W	Double
@100000~@165535	對應 Registry R0~R65535	R/W	Long
備註	■ 所有公用變數除非遇到關機，否則並無生命週期之限制。 ■ @1~@400 關機後若仍需要記憶的話，請利用 No3811 設定之。 ■ DOS 僅支援 R0~R7999，因此使用對應公用變數時需特別注意。		

8.2 區域變數(Local variables)

編號	說明	規則	型態
#0	空白 VACANT	R	-
#1~#26	系統內定用來作為巨集引數使用之區域變數	R	Double
#27~#400	給巨集程式使用之區域變數	R/W	Double
備註	■ MACRO 中所使用的區域變數，其生命週期僅限於該 MACRO 執行時，當執行完畢離開時，該區域變數將自動回復成 VACANT 狀態。 ■ 副程式可與其父程式共用區域變數，其生命週期於父程式執行完畢時也跟著結束。 ■ 巨集引數對照表，請參閱【MACRO 撰寫注意事項】。		

8.3 系統變數(System variables)

8.3.1 程式資訊(Modal information, #1000~#1056)

編號	說明	規則	型態
#1000	插值模式, 00/01/02/03/33/34/35	R/ W	Long
#1002	平面選擇模式, 17/18/19	R	Long
#1004	絕對/增量命令模式, 90/91	R	Long
#1006	行程限制模式, 22/23	R	Long
#1008	加工進給模式, 94/95	R	Long
#1010	英/公制模式, 70/71	R	Long
#1012	刀具半徑補償模式, 40/41/42	R	Long
#1014	刀長補償模式, 43/44/49	R	Long
#1016	縮放模式, 50/51	R	Long
#1018	主軸速度模式, 96/97	R	Long
#1020	切削進給控制模式, 61/62/63/64	R	Long
#1022	旋轉模式, 68/69	R	Long
#1024	主軸轉速變動偵測模式, 25/26	R	Long
#1026	極座標插值模式, 12.1/13.1	R	Long
#1028	極座標命令模式, 15/16	R	Long
#1030	刀具半徑補償之刀號, D 碼	R	Long
#1032	刀長補償之刀號, H 碼	R	Long
#1034	主軸轉速值, S 碼	R	Long
#1036	刀具號碼, T 碼	R	Long
#1038	輔助功能, M 碼	R	Long
#1040	目前工件座標號碼, G54: #1040=1; G55: #1040=2; G56: #1040=3...	R	Long
#1042	程式次序碼, N 碼	R	Long
#1044	G66.1 的上一單節插值模式, 可能為 4(00/01/02/03)或者空白(M_S_T_F_)	R	Long
#1046	進給量, F 碼	R	Double
#1048	目前行號	R	Long
#1050	程式開始的次序碼	R	Long
#1052	程式開始的行號	R	Long
#1054	主軸運轉狀態(M03/M04/M05)	R	Long
#1056	Program sequence number after corner processing, N Code	R	Long
#1062	系統核心變數	-	-

8.3.2 操作控制/狀態變數(Operation control/status, #1500~#1624)

編號	說明	規則	型態
#1500	寧靜(Quiet)模式，系統僅記錄程式座標，但不進行動程規劃 0：關閉；1：開啟。	R/W	Long
#1502	- 位元 0：當 PLC C40 開啟時，是否執行單步執行控制？ 0：執行單步執行功能，此為預設值，系統重置時會自動回復； 1：不執行單步執行功能。 位元 1：系統是否要在 MST 碼未完成前，繼續往下執行後續程式內容？(暫不提供) 0：要等待，此為預設值，系統重置時會自動回復； 1：不等待，此時 PLC S30(DEN)不會輸出。	R/W	Long
#1504	- 位元 1：暫停(Feedhold)與軸向/主軸百分比更動(Override)是否有作用？ 0：加工中暫停與軸向/主軸百分比更動有效[預設值]； 1：加工中暫停無效，並且軸向/主軸百分比固定為 100%。 - 位元 2：暫停(Feedhold)是否有作用？ 0：加工中暫停有效[預設值]； 1：加工中暫停無效。 - 位元 3：快動百分比更動(Override)是否有作用？ 0：加工中快動百分比更動有效[預設值]； 1：加工中快動百分比固定為 100%。 - 位元 4：切削百分比更動(Override)是否有作用？ 0：加工中切削百分比更動有效[預設值]； 1：加工中切削百分比固定為 100%。 - 位元 5：主軸百分比更動(Override)是否有作用？ 0：加工中主軸百分比更動有效[預設值]； 1：加工中主軸百分比固定為 100%。 註：#1504.2~#1504.5 於 114.51 之後版本始有效。	R/W	Long
#1505	- 位元 1：保留 - 位元 2：保留 - 位元 3：保留 - 位元 4：保留 - 位元 5：主軸進給百分比(Override)鎖定 0：加工中主軸進給百分比更動有效[預設值]； 1：加工中主軸進給百分比鎖定為當時之設定。 註：#1505 值與#1504 同時使用時，將以#1505 為主。此功能有效版本始於 10.114.56E/10.116.0E/10.116.5。	R/W	Long
#1506	圖形模擬模式，系統解譯 NC 程式的兩種狀態 0：一般解譯模式，系統用以進行動程規劃差值等功能使用 1：圖形模擬模式，系統用以取得程式大小範圍時使用	R	Long
#1507	圖型模擬是否關閉旗標 (只影響圖型模擬) 0：恢復/啟動圖型模擬 1：關閉圖型模擬 (10.116.x 功能)	W	Long
#1508	目前執行該巨集程式的軸群代碼 1：第一軸群；2：第二軸群；3：第三軸群；4：第四軸群	R	Long
#1512	G04.1 同步等待旗標 0：一般非同步等待狀態；1：同步等待中狀態	R	Long

編號	說明	規則	型態
	註：10.116 後不再提供此功能。		
#1514	G33 螺牙車削進行旗標 0：一般車削狀態；1：進退刀車削狀態；2：拉刀車削狀態車削狀態	R/W	Long
#1517	系統使用變數	R/W	Long
#1600	線性軸最小輸入單位(條/微米/次微米)，對應 Pr17(控制精度, BLU)	R	Long
#1602	旋轉軸最小輸入單位(10 毫度/1 毫度/0.1 毫度)，對應 Pr17(控制精度, BLU)	R	Long
#1604	U/V/W 是否視為 X/Y/Z 軸之增量命令模式 0：視為一般 U/V/W 軸之命令模式 1：視為 X/Y/Z 軸之增量命令模式	R	Long
#1606	巨集程式堆疊內的元素數量	R	Long
#1608	G31 跳脫指令對應的 C Bit 是否被觸發 0：尚未觸發；1：已觸發	R	Long
#1610	主軸定位停止角度	R/W	Long
#1612	預設工件座標號碼, G54: #1040=1; G55: #1040=2; G56: #1040=3...	R/W	Long
#1614	加工主軸預設轉速，對應 Pr1871~P1876	R/W	Long
#1616	斷刀點執行之程式次序號	R	Long
#1618	斷刀點執行之程式行號	R	Long
#1620	目前程式次序碼	R	Long
#1622	目前程式行號	R	Long
#1624	目前有效主軸號碼	R	Long

8.3.3 目前位置(Current position, #1301~#1476)

編號	說明	規則	型態
#1301~ #1316	各軸向於單節終點時的程式座標	R	Double
#1321~ #1336	各軸向目前的機械座標，移動中不可讀取	R	Double
#1341~ #1356	各軸向目前的程式座標	R	Double
#1361~ #1376	G31 跳脫指令對應的 C Bit 被觸發時，各軸向所對應的程式座標	R	Double
#1381~ #1396	各軸向的刀長補償值	R	Double
#1401~ #1403	上一個圓弧指令的中心向量(I, J, K)	R	Double
#1404~ #1406	刀具向量座標	R	Double
#1411~ #1419	XYZABCUVW 軸向於單節終點時的工件座標，對應關係如下： 1411(X); 1412(Y); 1413(Z) 1414(A); 1415(B); 1416(C)	R	Double

編號	說明	規則	型態
	1417(U); 1418(V); 1419(W)		
#1421~ #1436	Current encoder position of in workpiece coordinate machine	R	Double
#1441~ #1456	G31 跳脫指令對應的 C Bit 被觸發時，各軸向所對應的機械座標	R	Double
#1461~ #1476	執行暫停點回歸(Pr3852)時，各軸的偏移量	R	Double

8.3.4 運行狀態(Runtime state, #1800~#1976)

編號	說明	規則	型態
#1800	剛性攻牙於旋轉軸上的追隨誤差(毫度)	R	Double
#1801	剛性攻牙於 Z 軸上的追隨誤差(微米)	R	Double
#1802	剛性攻牙於 Z 軸上的最大追隨誤差(微米)	R	Double
#1803	第二組剛性攻牙於 Z 軸上的最大追隨誤差(微米) 有效版本 10.114.16~10.116.5	R	Double
#1804	第三組剛性攻牙於 Z 軸上的最大追隨誤差(微米) 有效版本 10.114.16~10.116.5	R	Double
#1805	第四組剛性攻牙於 Z 軸上的最大追隨誤差(微米) 有效版本 10.114.16~10.116.5	R	Double
#1806	第五組剛性攻牙於 Z 軸上的最大追隨誤差(微米) 有效版本 10.114.16~10.116.5	R	Double
#1807	第六組剛性攻牙於 Z 軸上的最大追隨誤差(微米) 有效版本 10.114.16~10.116.5	R	Double
#1815	教導功能旗標 0：關閉；1：開啟。	R	Double
#1816	教導功能的進給速度設定(IU/min)	R/ W	Double
#1820	靜音(Mute)模式，系統將進行動程規劃，但不會有實際命令輸出， 需配合 G10 L1100 指令作設定。 0：關閉；1：開啟。	R/ W	Double
#1821	累積切削長度	R/ W	Double
#1822	切削進給速度命令 F(mm/min)	R/ W	Double
#1823	主軸轉速命令(RPM)	R/ W	Double
#1824	有效切削控制模式, G61/G62/G63/G64	R	Double
#1825	有效差值模式, 00/01/02/03/04/33/34/35	R	Double
#1826	HPCC 模式(選配功能) 0：關閉；非 0：開啟 人機通過#1826 確定在加工監控頁面是否顯示 HPCC 有效版本 10.116.0I, 10.116.6B (含) 後	R	Double
#1827	有效工件座標號碼, G54: #1040=1; G55: #1040=2; G56: #1040=3...	R	Double
#1828	目前運作的參數下，系統預估的加工誤差 (BLU，當 Pr3808 = 0 時，不會顯示)	R	Double

編號	說明	規則	型態
#1829	多組高速高精參數之選擇	R/W	Double
#1831	加工主軸耦合模式, G51.2 / G113 / G114.1 / G114.3	R	Double
#1832	絕對/增量命令模式, 90/91	R	Double
#1833	插值模式, 00/01/02/03/33/34/35	R	Double
#1834	平面選擇模式, 17/18/19	R	Double
#1835	絕對/增量命令模式, 90/91	R	Double
#1836	行程限制模式, 22/23	R	Double
#1837	加工進給模式, 94/95	R	Double
#1838	英/公制模式, 70/71	R	Double
#1839	刀具半徑補償模式, 40/41/42	R	Double
#1840	刀長補償模式, 43/44/49	R	Double
#1841	縮放模式, 50/51	R	Double
#1842	主軸速度模式, 96/97	R	Double
#1843	切削進給控制模式, 61/62/63/64	R	Double
#1844	旋轉模式, 68/69	R	Double
#1845	主軸轉速變動偵測模式, 25/26	R	Double
#1846	極座標插值模式, 12.1/13.1	R	Double
#1847	極座標命令模式, 15/16	R	Double
#1851	系統核心變數	-	-
#1881 ~ #1896	各軸向的手輪偏置量 各軸群的偏置量需獨立設定，若要用人機畫面上的手輪偏置來改設定，只能改到第一軸群的部份	R/W	Double
#1901 ~ #1916	各軸向的 G92 座標系偏置量	R/W	Double
#1930	G92 座標系的旋轉偏置量	R/W	Double
#1931 ~ #1933	G92 座標系的旋轉偏置向	R/W	Double
#1941 ~ #1956	各軸向的第三軟體行程正極限(IU)	R/W	Double
#1961 ~ #1976	各軸向的第三軟體行程負極限(IU)	R/W	Double

8.3.5 模態變數(Modal variables, #1080~#3100)

編號	說明	規則	型態
#1080~ #1099	電控人員使用之模態變數(系統斷電時消失)	R/W	Double
#2001~ #2100	系統內部使用之模態變數(系統重置時消失)	R/W	Double
#3001~	電控人員使用之模態變數(系統重置時消失)	R/W	Double

編號	說明	規則	型態
#3100			
備註	模態變數生命週期不受限於單一 MACRO，因此可用於跨 MACRO 之變數存取。		

8.3.6 客制參數(Customer param., #4001~#5500)

編號	說明	規則	型態
#4001~ #4100	系統內部使用之客制參數(對應參數 4001~4100)	R	Double
#5001~ #5500	電控人員使用之客制參數(對應參數 5001~5500)	R	Double
備註	參閱 EMC6_C005_擴充參數使用說明文件以開啟#5001~顯示		

8.3.7 介面訊號(Interface signals, #6001~#6032)

編號	說明	規則	型態
#6001~ #6032	MLC 介面訊號，C101~C132/ S101~S132 Ex: @1 := #6001; // 指派 C101 狀態給@1。若 C101 On，則@1=1，反之則為 0 #6001 := @2; // 指派@2 內容給 S101。若@2=1，則 S101 On，反之 S101 Off	R/W	Double
備註			

8.3.8 刀具補償(Tool compensation, #10000~#15288)

銑床刀表				
編號	刀長補償(H)		刀徑補償(D)	
	幾何補償	磨耗補償	幾何補償	磨耗補償
0	#11000	#10000	#13000	#12000
1	#11001	#10001	#13001	#12001
...	...	...	...	...
96	#11096	#10096	#13096	#12096
...	...	...	...	...
800	#11800	#10800	#13800	#12800
備註	- 第 0 號刀所有對應補償皆為 0 - 標準人機僅提供 96 把刀輸入畫面 - 上述變數型態皆為 Double			

車床刀表						
編號	刀長補償(H)		刀徑補償(D)			
	幾何補償	磨耗補償	幾何補償	磨耗補償	刀鼻方向	刀鼻角度
1	#11001(1 st) #11002(2 nd)	#10001(1 st) #10002(2 nd)	#13003	#12003	#14003	#15003

	#11003(3 rd) #11401(4 th) #11402(5 th) #11403(6 th)	#10003(3 rd) #10401(4 th) #10402(5 th) #10403(6 th)				
2	#11004(1 st) #11005(2 nd) #11006(3 rd) #11404(4 th) #11405(5 th) #11406(6 th)	#10004(1 st) #10005(2 nd) #10006(3 rd) #10404(4 th) #10405(5 th) #10406(6 th)	#13006	#12006	#14006	#15006
...	...	...	...	...	...	...
96	#11286(1 st) #11287(2 nd) #11288(3 rd) #11686(4 th) #11687(5 th) #11688(6 th)	#10286(1 st) #10287(2 nd) #10288(3 rd) #10686(4 th) #10687(5 th) #10688(6 th)	#13288	#12288	#14288	#15288
備註	■ 第 0 號刀所有對應補償皆為 0 ■ 系統目前僅提供前六軸具刀補功能，且需設定為線性軸 ■ 上述變數型態皆為 Double					

8.3.9 工件座標系統(Workpiece coordinate system offset value, #20001~#20656)

編號	說明	規則	型態
#20001~ #20016	外偏座標系統偏移值	R/W	Double
#20021~ #20036	G54(G54P1)座標系統偏移值	R/W	Double
#20041~ #20056	G55(G54P2)座標系統偏移值	R/W	Double
...	...	R/W	Double
#20121~ #20136	G59(G54P6)座標系統偏移值	R/W	Double
#20141~ #20156	G59.1(G54P7)座標系統偏移值	R/W	Double
...	...	R/W	Double
#20301~ #20316	G59.9(G54P15)座標系統偏移值	R/W	Double
#20321~ #20336	G54P16 座標系統偏移值	R/W	Double
...	...	R/W	Double
#20641~ #20656	G54P32 座標系統偏移值	R/W	Double
備註	各組座標系統可對應 16 個軸向		

8.3.10 參考點位置(Reference point position, #26001~#26076)

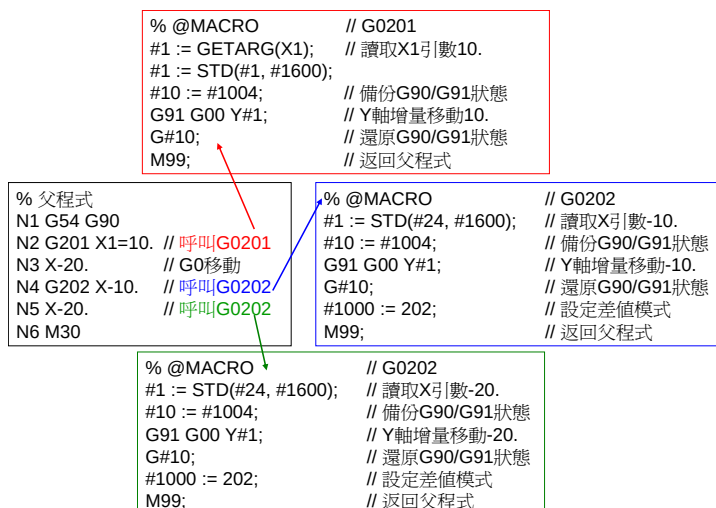
編號	說明	規則	型態
#26001~ #26016	各軸第 1 參考點位置	R/W	Double
#26021~ #26036	各軸第 2 參考點位置，對應 Pr2801~Pr2816	R/W	Double
#26041~ #26056	各軸第 3 參考點位置，對應 Pr2821~Pr2836	R/W	Double
#26061~ #26076	各軸第 4 參考點位置，對應 Pr2841~Pr2856	R/W	Double
備註	■ 各組參考點位置可對應 16 個軸向 ■ 第 1 參考點位置即為原點		

9 擴充 G 碼使用說明

控制器內建的基本 G 碼可能無法滿足各產業應用之需求，此時使用者將可自行撰寫 MACRO 作為擴充 G 碼，以滿足其自身需求。以下將就控制器 MACRO 處理流程，以及撰寫注意事項作說明。

9.1 MACRO 讀取處理流程

- N1：設定座標系為 G54，以及採取絕對量 G90 移動
- N2：呼叫 G0201 巨集，並透過 GETARG 函數將 X1 引數帶入。
 - 進入 G0201 後，將 X1 引數儲存於區域變數#1。
 - 以#10 備份 G90/G91 狀態
 - Y 軸以 G00 增量移動 10mm
 - 還原 G90/G91 狀態
 - 返回父程式。
- N3：由於離開 G0201 前最後一個差值模式為 G00(#1000=0)，故此單節表示 X 軸將以 G00 增量移動-20mm。
- N4：呼叫 G0202 巨集，並透過#24 將 X 引數帶入。
 - 進入 G0202 後，分別將 X 引數儲存於區域變數#1。
 - 以#10 備份 G90/G91 狀態
 - Y 軸以 G00 增量移動-10mm
 - 還原 G90/G91 狀態
 - 返回父程式。
- N5：由於離開 G0202 前將差值模式儲存為 202，故執行此單節時，系統將會再次呼叫 G0202。
- N6：程式結束。



9.2 MACRO 撰寫注意事項

- MACRO 內請多使用區域變數(Local Variables, #1 ~ #400)，而不要使用公用變數(Global Variables, @1 ~ @165535)。因為 MACRO 執行時，使用者的資料是透過引數(A_, B_, ..., Z_, X1=_, Y1=_, ...)傳入，由公用變數傳遞不符合使用習慣。
- 下表為引數與區域變數關係；針對擴充位址，例如 X1 引數的讀取方式，則請使用 GETARG(address)函數。

Address	Variable Number	Address	Variable Number	Address	Variable Number
A	#1	J	#5	T	#20
B	#2	K	#6	U	#21
C	#3	L	#12	V	#22
D	#7	M	#13	W	#23
E	#8	P	#16	X	#24
F	#9	Q	#17	Y	#25
H	#11	R	#18	Z	#26
I	#4	S	#19	X1	GETARG(X1)

- 由於模式變數(Modal Variables, #2001 ~ #2100; #3001 ~ #3100)在系統重置時，才會回復成 VACANT 狀態，因此可應用於多個 MACRO 間進行資料交換之時機，以節省共用資源之使用。
- MACRO 若需要內定初始值，可多加利用使用者參數(Custom Parameter, #4001 ~ #4100, #5001 ~ #5100)。
- 執行 MACRO 時，如需改變模式 G 碼(G91/G90, G40/G41/G42, ..., etc)狀態，請在一進入 MACRO 時，即備份當前的狀態，待離開 MACRO 前再回復原有狀態。
- 當離開 MACRO 後，仍想保留此 MACRO 插值模式(#1000)，建議在離開 MACRO 程式前，將#1000 指定為該 MACRO 號碼。爾後只要是軸向位移指令單節，系統將自動呼叫此 MACRO 程式，而不用再次指定。當然此 MACRO 之插值模式在遇到 G00/G01/G02/G03/G31/G33，或#1000 內容變動時，將被自動解除。
- 對於長度或角度的引數，在運算前請使用 STD()函數將單位標準化，以符合工具機使用習慣。
- 不可以改變座標系統設定，G92/G54/G52 等與座標系統相關指令不使用，否則圖形模擬功能將失去參考意義。
- 系統在執行加工時，對於 MACRO 程式會事先預解，因此 MACRO 執行速度會超前 G/M 碼指令，故當變數指定或資料讀取須與 G/M 碼指令發出排程同步時，請於變數指定或資料讀取前加入 WAIT()指令，以確保動作正確性。
- MACRO 程式最後需加”M99;”，才能返回到父程式。
- 請盡量在程式內容中多加入註解，養成良好習慣，增加程式可讀性可幫助後續維護及問題排除。

9.3 擴充 G 碼檔案命名規則與儲存路徑

■ 擴充 G 碼命名規格

目前僅支援 G200~G999 擴充 G 碼，範例如下

G200 所對應之 MACRO，其檔名為 G0200；

G200.1 所對應之 MACRO，其檔名為 G200001；

G200.10 所對應之 MACRO，其檔名為 G200010；

G200.100 所對應之 MACRO，其檔名為 G200100，以上檔名均無副檔名。

■ 擴充 G 碼程式儲存路徑

DOS 系統：C:\CNC\MACRO

CE 系統：DiskC:\OpenCNC\MACRO

9.4 登錄 G 碼相關說明及規則

■ 登錄 G 碼使用規格

目前只提供 G00/G01/G02/G03 可以自訂 MACRO 取代。

參數 3701~3710 設定：

0：不啟動 G00/G01/G02/G03 自訂 MACRO 取代功能。

-1：表 G00 以自訂 MACRO(G0000)取代；

1：表 G01 以自訂 MACRO(G0001)取代；

2：表 G02 以自訂 MACRO(G0002)取代；

3：表 G03 以自訂 MACRO(G0003)取代；

4：表 G53 以自訂 MACRO(G0053)取代；

■ 注意事項

1. G 碼巨集中執行之 G 碼指令皆為一般 G 碼功能。
2. 舊規格 G 碼巨集中以 G900000 執行 G00 之功能仍可使用。
3. 登錄 G 碼巨集與一般差值模式一樣有繼承功能(G53 除外)，例如執行
G00 X100.
Y100.
其中 Y100.也會執行 G00 巨集並將 Y100.引數帶入。
4. G 碼巨集在解譯時與一般差值 G 碼無異，可完全取代之。
5. 若 G00 巨集中有使用 G01 或其他差值模式且有需要繼承 G00 差值模式，務必在 G00 巨集結束前再下一次 G00 將差值模式換回 G00 否則差值模式將改變。
6. 此功能不支援車床 G7.1、G12.1 和 G72~G75 等車削循環；不支援 A、R、C 功能。
7. G53 除了非插值模式之外，使用方法與其餘相同。

■ 相容性異動

1. 10.114.50 之後，登錄 G 碼呼叫巨集功能，其 G 碼巨集在加工程式中的解譯順序與一般差值 G 碼完全相同；

- 10.114.49 以前，有些規格未定義明確因此可能造成新、舊版本的差異。
2. 原先在 G 碼巨集中將差值模式改為 900000 (#1000 := 900000) 的特殊用法，10.114.50 之後將不再支援。
 3. 10.116.16A、10.116.17 之後，提供 G53 以自訂 MACRO(G0053) 取代

10 MACRO 自訂警報方式

10.1 MACRO 警報觸發語法

```
%@MACRO
ALARM(xxx);    // xxx 為警報編號
M30;
```

10.2 DOS 版本警報內容編輯說明

- 檔案路徑：
 - 繁簡中文：C:\CNC\EXE\APPDATA.RES\CNCCHI.STR
 - 英文版：C:\CNC\EXE\APPDATA.RES\CNCENG.STR
 - 其他：C:\CNC\EXE\APPDATA.RES\CNCLOC.STR
- 內容格式：**24xxx**="1;MSG=**警報內容**"，其中 **xxx** 為警報編號，請選擇未使用過的號碼作為自訂警報編號，並請注意識別代號為 **24**。
- 檔案範例：
 - CNCCHI.STR 範例：


```
24003="1;MSG=最大圓弧弦長小於或等於 0"
```
 - CNCENGSTR 範例：


```
24003="1;MSG= max arc length can not be negative"
```

10.3 WinCE 版本警報內容編輯說明

- 檔案路徑：
 - 中文版：DiskC\OCRes\CHT\String\AlarmMacro_CHT.Xml
 - 英文版：DiskC\OCRes\Common\String\AlarmMacro_Com.Xml
 - 通則：DiskC\OCRes**L**\String\AlarmMacro_**L**.Xml，其中 **L** 為各語系之名稱
- 檔案格式：<Message ID="AlarmMsg::**Macro**::ID=**xxx**" Content="警報內容" />，其中 **xxx** 為警報編號，請選擇未使用過的號碼作為自訂警報編號，並請注意識別字母為 **Macro**。警報內容的字串長度可容納 48 個英文字，或是 31 個中文字，多餘的字串會超出警報視窗。
- 檔案範例：
 - CusMacroAlarmMsg_CHT.Xml 範例：


```
<Message ID="AlarmMsg::Macro::ID=3" Content="最大圓弧弦長小於或等於 0" />
```
 - CusMacroAlarmMsg_Common.Xml 範例：


```
<Message ID="AlarmMsg::Macro::ID=3" Content="max arc length can not be negative" />
```

11 MACRO 自訂提示方式

11.1 規格說明

- 有別於 MACRO 警報發生時，使用者必須重置系統才可消除警報狀態及其畫面；MSG 自訂提示僅需按下"ESC"，即可將畫面消除，因此可用於單純提示狀態，但須注意當程式結束時，該提示將自動消失。

11.2 MSG 自訂提示觸發語法

- MSG(100);

//警報 ID



- MSG("鑽頭移失");

//警報顯示內容



- MSG(100, "鑽頭遺失");

//警報 ID+顯示內容



警報內容有字串長度限制

中文：19 個字

英文：39 個字

12 附件

12.1 Registry 資源分配表

編號	說明	巨集程式或人機介面	階梯程式	位元(Bit)存取	永久保存
R0~R39	CNC 系統介面區	唯讀	依下表定義之	是	否
R40~R49	PLC 警報訊息區				
R50~R80	使用者自行定義	讀/寫	讀/寫	是	否
R81~R100	對應到系統參數 3401~3420	唯讀	唯讀	是	否
R101~R102	刀具狀態	讀/寫	讀/寫	是	是
R103~R255	使用者自行定義	讀/寫	讀/寫	是	是
R256~R511					否
R512~R639	CNC 系統介面區	唯讀	依下表定義之	是	否
R640~R1023				否	
R1024~R1032	銑床自動分中/自動對刀佔用，車床為使用者自行定義區	讀/寫	讀/寫	否	否
R1033~R2049	使用者自行定義	讀/寫	讀/寫	否	否
R2050~R2090	Pr3810=0 為使用者自行定義	讀/寫	讀/寫	否	否
	Pr3810=1 為多個 M Code 同時執行功能佔用	讀			
R2091~R2099	使用者自行定義	讀/寫	讀/寫	否	否
R2100~R3100	Pr3228=0 為使用者自行定義	讀/寫	讀/寫	否	否
	Pr3228=1 為刀具管理功能佔用				是
R3101~R4095	使用者自行定義	讀/寫	讀/寫	否	否
R4096~R5999	CNC 系統介面區	唯讀	依下表定義之	否	否
R6000~R7999	使用者自行定義(Fram)	讀/寫	讀/寫	否	是
R8000~R8255	CNC 系統介面區	讀/寫	唯讀	否	是
R8256~R9999	CNC 系統介面區	唯讀	唯讀	否	否
R10000~R10999	CNC 系統加值應用記憶區	讀/寫	讀/寫	否	是
R11000~R14999	CNC 系統介面區	唯讀	唯讀	否	否
R15000~R19999	CNC 系統加值應用非記憶區	讀/寫	讀/寫	否	否
R20000~R65535	使用者自行定義	唯讀	讀/寫	否	否

※ DOS 僅支援 R0~R7999

12.2 基本 G 碼指令一覽表

項 目	功 能 名 稱
G00	直線快速定位
G01	直線補間、切削進給
G02	圓弧補間(順時鐘)
G03	圓弧補間(逆時鐘)
G04	暫停指定時間
G10	可程式資料輸入
G15	取消極座標命令
G16	極座標命令
G17	設定XY工作平面
G18	設定ZX工作平面
G19	設定YZ工作平面
G28	原點回歸
G29	從參考點復歸
G30	任意參考點回歸
G31	跳躍功能
G33	螺牙切削
G40	刀具半徑補償消除
G41	刀具半徑左補償
G42	刀具半徑右補償
G43	刀具長度正補償
G44	刀具長度負補償
G49	刀具長度補償取消
G50	取消放大縮小
G51	放大縮小
G52	設定區域座標(子工件坐標)
G53	機械座標定位
G54	設定工件座標系統
G55	設定第二工件座標系統
G56	設定第三工件座標系統
G57	設定第四工件座標系統
G58	設定第五工件座標系統
G59	設定第六工件座標系統
G65	單一巨集程式呼叫
G66	模式巨集程式呼叫
G67	模式巨集程式呼叫取消
G67	取消座標旋轉
G68	座標旋轉
G70	英制單位加工
G71	公制單位加工
G90	絕對位置輸入方式

項 目	功 能 名 稱
G91	相對位置輸入方式
G92	加工位置座標系統設定
G94	每分鐘進給量(mm/min.)
G95	每轉進給量(mm/rev.)
G96	等表面切削速度
G97	等表面切削速度取消

12.3 MACRO 範例

取材自車床 G21 車床車牙模式循環巨集，其規格請參考車床程式使用手冊

```
% @MACRO
// WHEN NO X(U) Z(W) ARGUMENTS, THEN LOAD MODAL INFORMATION
// ELSE DO THREAD CUTTING
// #8(E)  LEAD COUNT PER INCH
// #18(R) TAPER AMOUNT
// #2070  RECORDED TAPER AMOUNT
// #2071  RECORDED Z AXIS AMOUNT

// PROCESS NO ARGUMENT CASE WHICH MAY CAUSE FROM MODAL
RESTORE
IF( #21 = #0 AND #23 = #0 AND #24 = #0 AND #26 = #0 ) THEN
    M99;
END_IF;

// PROCESS TAPER
IF( #1000 <> 21 ) THEN
    // WHEN FIRST ENTRY, CLEAR TAPER MODAL STATE
    #2070 := 0;
    #2071 := 0;
    #2072 := #0;
END_IF;

IF( #18 <> #0 ) THEN
    // THERE ARE TAPER ADDRESS, RECORD IT INTO MODAL VARIABLE
    #2070 := #18;
ELSE
    // NO TAPER ADDRESS APPEAR, INHERIT MODAL STATE
    #18 := #2070;
END_IF;

IF( #9 <> #0 ) THEN
    // THERE ARE LEAD ADDRESS, RECORD IT INTO MODAL VARIABLE
    #2072 := #9;
ELSE
    // NO LEAD ADDRESS APPEAR, INHERIT MODAL STATE
    #9 := #2072;
END_IF;

IF( #23 <> #0 OR #26 <> #0 ) THEN
    // WHEN THERE ARE W OR Z ADDRESS APPEAR, THEN
    // CHECK WHICH KIND OF COMMAND BEEN ADDRESS, AND
    // SAVE IT INTO MODAL VARIABLE
    IF( #26 <> #0 ) THEN
        // Z ADDRESS
        #2071 := #26 - #1303;
    ELSE
        // W ADDRESS
        #2071 := #23;
    END_IF;
ELSE
    // WHEN THERE ARE NO Z/W ADDRESS, THEN INHERIT IT FROM
    // MODAL VARIABLE
```

```
#26 := #2071 + #1303;
END_IF;

// PROCESS E ADDRESS
IF( #8 <> #0 AND #9 = #0 ) THEN
  IF( #1008 = 94 ) THEN
    // FEED PER MINUTE, CALCULATE MM/MIN = LEAD * SPINDLE
SPEED
    #9 := (25.4 * #1034) / #8;
    #2072 := #9;
  ELSE
    // FEED PER REVOLUTION, CALCULATE MM/REV = LEAD
    #9 := 25.4 / #8;
    #2072 := #9;
  END_IF;
END_IF;

// STANDARDIZE ARGUMENT
#9 := STD(#9,#1600);
#21 := STD(#21,#1600);
#23 := STD(#23,#1600);
#24 := STD(#24,#1600);
#26 := STD(#26,#1600);
#18 := STD(#18,#1600);

// working variable
// #31      chamfer start point relative to block end X
// #32      chamfer block X-direction displacement
// #33      chamfer amount
// #36      thread head number iterative count
// #37      thread start angle

// READ CHAMFER AMOUNT
#33 := (#4043 * #9) / 10.0;

// COPY X,Z INFORMATION INTO U,W

// PROCESS X ADDRESS
IF( #24 <> #0 ) THEN
  #21 := #24 - #1301;
END_IF;

// PROCESS Z ADDRESS
IF( #26 <> #0 ) THEN
  #23 := #26 - #1303;
END_IF;

// process H addresss, the head number
IF( #11 <> #0 ) THEN
  #11 := ROUND(#11);
ELSE
  // set default head number
  #11 := 1;
END_IF;

// CALCULATE CHAMFER START POINT RELATIVE TO BLOCK END POINT
IN X
```

```
#31 := (SIGN(#23) * #33 * 2 * #18)/#23;

// CALCULATE CHAMFER BLOCK X-DIRECTION DISPLACEMENT
#32 := -SIGN(#21)*#33*2;

FOR #36:=1 TO #11 DO

    // calculate thread start angle
    #37 := (360.0 / #11) * (#36 - 1);

    G00 U( #21 + #18*2 );
    G33 U-(#18*2-#31) W(#23-SIGN(#23)*#33) Q#37 F(#9*#11);
    G33 U#32 W(SIGN(#23)*#33) Q#37;
    G00 U-#21-#32-#31;
    G00 W-#23;

END_FOR;

// SET INTERPOLATION MODE TO 21
#1000 := 21;

// RETURN
M99;
```

12.4 MACRO XML 資料應用

- MACRO 可以使用特殊的函數讀取 xml 檔案，分別是 DBLOAD 和 DBOPEN，DBOPEN 用來載入 xml 資料檔案，DBLOAD 用來讀取資料內容。
- 應用範例：下圖為某產機客制人機畫面，該畫面將自行產生一 xml 檔案，以記錄相關加工資料，之後在巨集規劃動程時，同步讀取該 xml 內容，以作為規劃依據。

尺寸編輯		模式關閉		前座外徑		12.98	線徑	1.80	方向	右旋	材質	SW-A	▼	間隙	直軸	切刀	一般
No	X 間隙	Y 送線	Z 右旋	A 上切	B 左旋	C 下切	R	K	T	起始速度							
1	0.00	17.63	12.98	267.54													
2	2.26	21.34	13.20	300.87													
3	2.26	91.19	13.20	443.29													
4	0.00	21.55	12.98	116.95													
5	-6.05	21.16	12.98	150.00													

- 該客制人機首先將使用者設定內容輸出成 xml 檔案，其語法格式定義如下，並且將該 xml 檔案存放於使用者所指定的 GNCFILES 路徑(參閱 Pr3219 說明)中：

```
<?xml version="1.0" encoding="UTF-16"?>
<CycleFile>
  <Cycle Name="Cycle_HerdonProg"> ←第一筆資料開頭
    <Field Name="Col_Y" Value="17.63"/>
    <Field Name="Col_Z" Value="12.98"/>
    <Field Name="Col_X" Value="0.00"/>
    <Field Name="Col_A" Value="267.54"/>
  </Cycle> ←第一筆資料結尾
  <Cycle Name="Cycle_HerdonProg"> ←第二筆資料開頭
    <Field Name="Col_Y" Value="21.34"/>
    <Field Name="Col_Z" Value="13.20"/>
    <Field Name="Col_X" Value="2.26"/>
    <Field Name="Col_A" Value="300.87"/>
  </Cycle> ←第二筆資料結尾
  <Cycle Name="Cycle_HerdonProg"> ←第三筆資料開頭
    <Field Name="Col_Y" Value="91.19"/>
    <Field Name="Col_Z" Value="13.20"/>
    <Field Name="Col_X" Value="2.26"/>
    <Field Name="Col_A" Value="443.29"/>
  </Cycle> ←第三筆資料結尾
  <Cycle Name="Cycle_HerdonProg"> ←第四筆資料開頭
    <Field Name="Col_Y" Value="21.55"/>
    <Field Name="Col_Z" Value="12.98"/>
    <Field Name="Col_X" Value="0.00"/>
    <Field Name="Col_A" Value="116.95"/>
```

```

</Cycle>                                ←第四筆資料結尾
<Cycle Name="Cycle_HerdonProg">        ←第五筆資料開頭
  <Field Name="Col_Y" Value="21.16"/>
  <Field Name="Col_Z" Value="12.98"/>
  <Field Name="Col_X" Value="-6.05"/>
  <Field Name="Col_A" Value="150.00"/>
</Cycle>                                ←第五筆資料結尾
</CycleFile>

➤ 使用者需自行編撰 xml 資料結構設定檔，用來宣告當巨集利用 DBLOAD 函
  數讀取資料時，要將所讀取的相關資料，存放至哪些對應變數中。其語法
  格式定義如下，並且要將此設定檔儲存於 OCRes\Common\Schema\中。
  <?xml version="1.0" encoding="UTF-16"?>
  <Schema>
    <Cycle name="Cycle_HerdonProg">
      <Field>
        <Name>Col_X</Name>
        <InputStorage>@1200</InputStorage>  ←Col_X 存入的變數中
        <InputFormat>Variant</InputFormat>
        <DefaultValue></DefaultValue>
      </Field>
      <Field>
        <Name>Col_Y</Name>
        <InputStorage>@1201</InputStorage>  ←Col_Y 存入的變數中
        <InputFormat>Variant</InputFormat>
        <DefaultValue></DefaultValue>
      </Field>
      <Field>
        <Name>Col_Z</Name>
        <InputStorage>@1202</InputStorage>  ←Col_Z 存入的變數中
        <InputFormat>Variant</InputFormat>
        <DefaultValue></DefaultValue>
      </Field>
      <Field>
        <Name>Col_A</Name>
        <InputStorage>@1203</InputStorage>  ←Col_A 存入的變數中
        <InputFormat>Variant</InputFormat>
        <DefaultValue></DefaultValue>
      </Field>
      <Field>
        <Name>Col_B</Name>
        <InputStorage>@1204</InputStorage>  ←Col_B 存入的變數中
        <InputFormat>Variant</InputFormat>
        <DefaultValue></DefaultValue>
      </Field>
      <Field>
        <Name>Col_C</Name>
        <InputStorage>@1205</InputStorage>  ←Col_C 存入的變數中

```

```
<InputFormat>Variant</InputFormat>
<DefaultValue></DefaultValue>
</Field>
</Cycle>
</Schema>
```

► MACRO 範例

```
// 載入 GNCFILES\Test 資料數，總共 5 筆，因此@1:=5;
@1:=DBOPEN("Test");
```

```
// 載入第 1 筆資料，DBLOAD 引數為 0
// @1200=0.00 @1201=17.63 @1202=12.98 @1203=267.54
DBLOAD( 0 );
```

```
// 載入第 2 筆資料，DBLOAD 引數為 1
// @1200=2.26 @1201=21.34 @1202=13.20 @1203=300.87
DBLOAD( 1 );
```