PRODUCT NAME

TM58XX

TITLE

TM58XX 虛擬指令用法

APPLICATION NOTE

TM58 系列指令表

TM58 系列指令表:
M:Memory
b07:b0 ~ b7
ma:m,a
I:integer 整數
//
//////////////////////// 虛擬指令 ////////////////////////////////
//
<CBLOCK>
語法:CBLOCK [<運算式>]
<標號>[:<增量>][,<標號>[:<增量>]]
ENDC
例 1：定義區塊常數
```
CBLOCK 0x10
aa,bb,cc            ;aa=0x10,bb=0x11,cc=0x12
ENDC
BCM  aa,b0          ;BCM  0x10,0
```
例 2：定義常數
```
CBLOCK 0x10
aa:0ah,bb:0ch,cc:0fh ;aa=0ah,bb=0ch,cc=0fh
ENDC
BCM  aa,b0          ;BCM  0ah,0
BCM  bb,b0          ;BCM  0ch,0
BCM  cc,b0          ;BCM  0fh,0
```
例 3：定義常數
```
CBLOCK 0x10
aa,bb:0ch,cc:0fh    ;aa=10h,bb=0ch,cc=0fh
ENDC
BCM  aa,b0          ;BCM  10h,0
BCM  bb,b0          ;BCM  0ch,0
BCM  cc,b0          ;BCM  0fh,0
```
例 4：定義常數
```
CBLOCK 0x10
aa:0ah,bb,cc        ;aa=0ah,bb=10h,cc=11h
ENDC
BCM  aa,b0          ;BCM  0ah,0
BCM  bb,b0          ;BCM  10h,0
BCM  cc,b0          ;BCM  11h,0
```
例 5：定義常數
```
CBLOCK 0x10
aa,bb:0ah,cc        ;aa=10h,bb=0ah,cc=11h
ENDC
BCM  aa,b0          ;BCM  10h,0
BCM  bb,b0          ;BCM  0ah,0
BCM  cc,b0          ;BCM  11h,0
```

<__CONFIG>
語法：__CONFIG <運算式>
例：設定 config 值
```
  __CONFIG  0x20        ;config=0x20
```

<CONSTANT>
語法：CONSTANT <標號>=<運算式> [...,<標號>=<運算式>]
例：定義常數
```
      CONSTANT  aa=0x12,bb=14h,cc=1fh  ;aa=0x12,bb=14h,cc=1fh
      BCM  aa,b0            ;BCM       12h,0
      BCM  bb,b0            ;BCM       14h,0
      BCM  cc,b0            ;BCM       1fh,0
```

<DB>
語法：[<label>] DB <expr> [,<expr>,...<expr>]
例：填一個 Byte 資料.
```
      OpCode  Source
      0x3420  DB 't',' '   -> warring  (0x7420 & 0x3FFF)
      0x0805  DB 8,0x5
```

<#DEFINE>
語法：#DEFINE <名稱> [<字串>]
例：定義變數
```
      #DEFINE abc 0x30
      #DEFINE control 0x10,0x7

      BCM  abc,0            ;BCM       0x30,0
      BCM  control          ;BCM       0x10,0x7
```

<DT>
語法：[<標號>] DT <運算式> [,<運算式>,...,<運算式>]
例：向指定 ROM 單元填充一個 RETLA+Byte 指令.
```
        向 0f0h,0f1h,0f2h ROM 單元填充一個 RETLA+Byte 指令,指令碼分別
        爲 0x3c33,0x3c70,0x3c71
        Org     0f0h
      操作碼  原指令語句
      0x3C33  dt 33h,"pq"
      0x3C70
      0x3C71
```

<DW>
語法：[< 標號>] DW <運算式> [,<運算式>,...,<運算式>]
例: ：向指定 ROM 單元放入兩個位元組的資料
```
       向 0f0h,0f1h,0f2h ROM 單元放入兩個位元組的資料, 資料分別
        爲 0x3170,0x2c20,0x1245
        Org     0f0h
```

```
操作碼  原指令語句
0x3170  DW "p1 ,",1245
0x2C20
0x1245
```

<ELSE>
語法：ELSE
例：條件彙編

```
abc equ 2
if abc==0
bcm 0,0                ;不編譯
else
bcm 1,1                ;編譯
endif
```

<END>
語法：END
例：程式結束語句,可寫可不寫。由該語句告訴編譯程序該語句後的程式不編譯。

```
.
.
END                    ;END 後語句不編譯
bcm  0,0
```

<ENDC>
語法：ENDC
例：定義區塊常數結尾，範例同 CBLOCK

```
CBLOCK
…..
….
ENDC
```

<ENDIF>
語法：ENDIF
例：判斷式結尾，範例同 ELSE

```
IF…..
…….
……
ENDIF
```

<ENDM>
語法：ENDM
例：巨集定義結束

```
Mymacro macro p1,p2
bcm p1,p2
exitm
```

```
    bcm 3,3
    endm
    Mymacro  2,1
```

<ENDW>
語法：ENDW
例：迴圈結尾

```
    variable i=0
    while i<2              ; bcm 2,0
    bcm 2,0                ; bcm 2,0
    i+=1;
    endw
```

<EQU>
語法：<標號> EQU <運算式>
例：定義變數

```
    abc equ 20             ;abc=20
```

<ERROR>
語法：ERROR <字串>
例：顯示錯誤資訊

```
    ERROR test message;在列表框(ListBox)上顯示錯誤資訊
```

<ERRORLEVEL>
語法：ERRORLEVEL <運算式>,10000　目前固定容許錯誤 10000 行
例：顯示錯誤及警告資訊等級

```
    ERRORLEVEL 0     ;0 = 錯誤列表, 0! = 錯誤及警告列表
```

<EXITM>
語法：EXITM
例：退出巨集,後面程式不編譯,範例同 ENDM

<FILL>
語法：<標號> FILL <運算式>,<計數值>
例：向指定的連續 ROM 單元填值,所填值爲運算式的內容，ROM 單元數由
計數值規定.

```
    Org     100h
Tabb：
    fill 0x2000,5          ;填 5 次 2000h(NOP)----向 100h,101h,102h,103h,104h ROM
                           ;單元填入 2000h(即 NOP 指令)
```

<IF>
語法：IF <運算式>
例：判斷式,範例同 ELSE

<IFDEF>
語法：IFDEF <標號>
例：條件彙編

```
#define deflabel
ifdef deflabel
bcm 3,3          ;編譯
else
bcm 1,1          ;不編譯
endif
```

<IFNDEF>
語法：IFNDEF
例：條件彙編

```
#define deflabel
ifndef deflabel
bcm 3,3          ;不編譯
else
bcm 1,1          ;編譯
endif
```

<#INCLUDE>
語法:：#INCLUDE <include_file>
例：引進包含文件

```
#INCLUDE <a.inc>   ;"< >" 預設路徑---編譯器(Compiler)路徑
INCLUDE "c:\a.inc"  ;指定路徑
INCLUDE "a.inc"     ;不指定路徑,a.inc 文件位於專案文件所在文件下
```

注：INCLUDE 關鍵字和其後運算式間應空一空格，INCLUDE 關鍵字可至頂也可不至頂端寫。

<LOCAL>
語法：LOCAL <運算式>[,<運算式>...]
例 1：用在局部範圍內,指定運算式的使用範圍

```
aa equ 2          ;全局定義(Global)
abc macro
local aa=3        ;局部定義(Local)
bcm  aa,0         ;bcm 3,0
endm
abc
```

例 2：用在內,指定運算式的使用範圍

```
aa equ 2          ;全局定義(Global)
abc macro cc
local aa=cc       ;局部定義(Local)
bcm  aa,0         ;bcm 9,0
endm
abc 09h
```

注：LOCAL 關鍵字和局部變數 aa 間應空一空格。

<MACRO>
語法：<LOCAL> MACRO [<arg>,...<arg>]
例：巨集定義,範例同 ENDM
注：巨集名應至頂寫,LOCAL 和 MACRO 關鍵字間應空一空格。

<ORG>
語法：[<標號>] ORG <運算式>
例：定義程式在 ROM 中的起始位置

```
ORG     200H
NOP                 ;200h ROM 單元內容爲 2000h(nop)
```

<RADIX>
語法：RADIX HEX
例：定義數制

```
RADIX HEX
Clrm    10          ;清除 10h  RAM 單元
RADIX DEC
Clrm    10          ;清除 0ah  RAM 單元
RADIX OCT
a equ 10            ;a=08h
```

<RES>
語法：[<標號>] RES <ROM 單元數>
例：指定下一程式位置

```
ab res 3            ;空 3 行
bcm 2,2
```

<SET>
語法： <變數名> SET <運算式>
例:設定變數值

```
area set 0          ;area=0
```

注：set 關鍵字和運算式之間應空一空格。
　　變數名和 set 關鍵字之間亦應至少空一空格。

<#UNDEFINE>
語法： #UNDEFINE <label>
例：解除條件彙編

```
#define  deflabel
#undefine deflabel
ifdef deflabel
bcm 3,3             ;不編譯
else
```

```
    bcm 1,1                 ;編譯
    endif
```

<VARIABLE>

語法：VARIABLE <變數名>[=<運算式>][,<變數名>[=<運算式>]...]

例：定義變數

```
    variable a=3
```

<WHILE>

語法:WHILE <運算式>

```
    :
    :
    ENDW
```

例：迴圈,範例同 ENDM

<STREQU>

語法：1.[<label>] STREQU "String"
2.[<label>] STREQU label Macro
3.[<label>] STREQU %(Arithmetic)

例：字串替換.

```
    variable var=3
    Str  STREQU "BCM 3,1"
    Str1 STREQU Str
    Str2 STREQU %(var+5)

    Str                 ;BCM 3,1
    Str1                ;BCM 3,1
    BCM  Str2,1         ;BCM 8,1
```

<CATSTR>

語法:[<label>] CATSTR "String1","String2"

例：字串相加.

```
    Txt1  STREQU "01"
    Txt2  STREQU "3"
    Str   CATSTR "01","23"

    BCM  Str,1          ;BCM 0123,1
```

<SUBSTR>

語法：1.[<label>] SUBSTR "String1",START,[LENGTH]
2.[<label>] SUBSTR label Macro1,START,[LENGTH]

例：取字串部份.

```
    Txt1  STREQU  "12345"
    Str   SUBSTR  "332211",4,2
    Str1  SUBSTR  Txt1,3,2

    BCM  Str,1          ;BCM 21,1
```

```
    BCM  Str1,1              ;BCM 34,1
```

<INSTR>

語法：1.[<Value>] INSTR START,"String1","SubString"
　　2.[<Value>] INSTR START,label Macro1,"SubString"

例：尋找字串位置.

```
    Txt1   STREQU  "12345"
    Txt2   STREQU  "5"
    Value  INSTR  1,Txt1,"4"
    Value2 INSTR  1,Txt1,Txt2

    BCM Value,0              ;BCM 4,0
    BCM Value2,0             ;BCM 5,0
```

<SIZESTR>

語法：1.[<Value>] SIZESTR "String1"
　　2.[<Value>] SIZESTR label Macro1

例：求字串長度.

```
    Txt1   STREQU "564"
    Value  SIZESTR "012345"
    Value1 SIZESTR Txt1
    Value2 SIZESTR Value

    BCM  Value,0             ;BCM 6,0
    BCM  Value1,0            ;BCM 3,0
    BCM  Value2,0            ;BCM 6,0
```

<SMTVAR>

語法：1.SMTVAR "String",[1,10,1],[1,10,2]
　　2.SMTVAR "String",[1,10,1],[2,2,0]
　　3.SMTVAR "String",[1,10,1]

例：快速聲明.

```
  SMTVAR "M%I equ J",[1,10,1],[1,10,2]  ;M1=1,M2=3,M3=5,M4=7,
                                        ;M5=9,M6=9,..,M10=9
  SMTVAR "N%I equ J",[1,10,1],[2,2,0]   ;N1=2,N2=2,..,N10=2
  SMTVAR "O%I equ J",[1,10,1]           ;O1=1,O2=2,O3=3,O4=4,..,O10=10

    BCM M3,0               ;BCM 5,0
    BCM N6,0               ;BCM 2,0
    BCM O7,0               ;BCM 7,0
```

<IFB>

語法：1.IFB "String"
　　2.IFB Str

例:：條件彙編,判斷字串是否爲空白字串

```
    a STREQU " "
    b STREQU "abc"
    IFB b
```

```
    bcm 1,0                 ;不編譯
    ENDIF
    IFB a
    bcm 2,0                 ;編譯
    ENDIF
    IFB "  "
    bcm 1,0                 ;編譯
    ENDIF
    IFNB b
    bcm 1,0                 ;編譯
    ENDIF
```

<IFNB>

語法：1.IFNB "String"
　　2.IFNB Str

例：條件彙編,判斷字串是否不爲空白字串,範例同 IFB

<IFSE>

語法：1.IFSE "String1","String2"
　　2.IFSE "String1",Str2
　　3.IFSE Str1,"String2"
　　4.IFSE Str1,Str2

例:：條件彙編, 判斷字串是否相等(區分大小寫)

```
abc STREQU "ac"
    IFSE abc,"acc"
    BCM 1,0                 ;不編譯
    ENDIF
    IFNSE abc,"acc"
    BCM 1,0                 ;編譯
    ENDIF
```

<IFNSE>

語法：1.IFNSE "String1","String2"
　　2.IFNSE "String1",Str2
　　3.IFNSE Str1,"String2"
　　4.IFNSE Str1,Str2

例:：條件彙編, 判斷字串是否不等(區分大小寫) ,範例同 IFSE

<IFCSE>

語法：1.IFCSE "String1","String2"
　　2.IFCSE "String1",Str2
　　3.IFCSE   Str1,"String2"
　　4.IFCSE   Str1,Str2

例:：條件彙編, 判斷字串是否相等(不區分大小寫)

```
    abc STREQU "ac"
    IFCSE abc,"acc"
    BCM 1,0                 ;不編譯
    ENDIF
```

```
IFNCSE abc,"acc"
BCM 1,0              ;編譯
ENDIF
```

<IFNCSE>

語法：1.IFNCSE "String1","String2"
2.IFNCSE "String1",Str2
3.IFNCSE Str1,"String2"
4.IFNCSE Str1,Str2

例:：條件彙編, 判斷字串是否不等(不區分大小寫),範例同 IFCSE

<ELSEIF>

語法：ELSEIF <運算式>

例:：條件彙編,否則,判斷運算式是否成立

```
a EQU 2
IF a<4
BCM 1,0              ;編譯
ELSEIF a<5
BCM 2,0              ;不編譯
ELSEIF a<=4
BCM 3,0              ;不編譯
ELSE
BCM 4,0              ;不編譯
ENDIF
```

<ELSEIFB>

語法：1.ELSEIFB "String"
2.ELSEIFB Str

例:：條件彙編,否則,判斷字串是否爲空白字串

```
str STREQU "asd"
IFB str
BCM 1,0              ;不編譯
ELSEIFB "test"
BCM 2,0              ;不編譯
ELSEIFB ""
BCM 3,0              ;編譯
ELSE
BCM 4,0              ;不編譯
ENDIF
```

<ELSEIFNB>

語法：1.ELSEIFNB "String"
2.ELSEIFNB Str

例:：條件彙編,否則,判斷字串是否不爲空白字串,範例同 ELSEIFB

<ELSEIFSE>

語法：1.ELSEIFSE "String1","String2"
2.ELSEIFSE "String1",Str2
3.ELSEIFSE Str1,"String2"
4.ELSEIFSE Str1,Str2
例:：條件彙編,否則,判斷兩字串是否相同(有大小寫之分)

```
str1 STREQU "teststr"
str2 STREQU "abc"
IFSE str1,"acc"
BCM 1,0              ;不編譯
ELSEIFSE "abc",str1
BCM 2,0              ;不編譯
ELSEIFSE str1,str2
BCM 3,0              ;不編譯
ELSEIFSE str2,"abc"
BCM 4,0              ;編譯
ELSEIFNSE str1,str2
BCM 5,0              ;不編譯
ENDIF
```

<ELSEIFNSE>
語法：1.ELSEIFNSE "String1","String2"
2.ELSEIFNSE "String1",Str2
3.ELSEIFNSE Str1,"String2"
4.ELSEIFNSE Str1,Str2
例:：條件彙編,否則,判斷兩字串是否不相同(無大小寫之分),範例同 ELSEIFSE

<ELSEIFCSE>
語法：1.ELSEIFCSE "String1","String2
2.ELSEIFCSE "String1",Str2
3.ELSEIFCSE Str1,"String2"
4.ELSEIFCSE Str1,Str2
例:：條件彙編,否則,判斷兩字串是否相同(有大小寫之分)

```
str1 STREQU "TESTSTR"
str2 STREQU "TESTSTR"
IFCSE str1,"acc"
BCM 1,0              ;不編譯
ELSEIFCSE "abc",str1
BCM 2,0              ;不編譯
ELSEIFCSE str1,str2
BCM 3,0              ;編譯
ELSEIFCSE str2,"abc"
BCM 4,0              ;不編譯
ELSEIFNCSE str1,str2
BCM 5,0              ;不編譯
ENDIF
```

<ELSEIFNCSE>

語法：1.ELSEIFNCSE "String1","String2"
2.ELSEIFNCSE "String1",Str2
3.ELSEIFNCSE Str1,"String2"
4.ELSEIFNCSE Str1,Str2
例:：條件彙編,否則,兩字串是否不相同(無大小寫之分),範例同 ELSEIFCSE

<ELSEIFDEF>
語法：ELSEIFDEF <運算式>
例：條件彙編,否則,判斷運算式是否有定義

```
#DEFINE deflabel
IFDEF def
BCM 1,0              ;不編譯
ELSEIFDEF deflabel
BCM 2,0              ;編譯
ELSEIFNDEF deflabel
BCM 3,0              ;不編譯
ELSE
BCM 4,0              ;不編譯
ENDIF
```

<ELSEIFNDEF>
語法：ELSEIFNDEF<運算式>
例:：條件彙編,否則,判斷運算式是否未定義。範例同 ELSEIFDEF

//
////////////////////// 運算符號 ////////////////////////////////
//
HIGH: 取 HIGH Byte
LOW: 取 LOW Byte
UPPER: 取 HIGH Byte 上的 LOW Byte
++： 數值加 1
-- ： 數值減 1
~ ： 異或(XOR)運算
! ： 取反(NOT)運算
(： 左括弧
) ： 右括弧
* ： 乘運算
/ ： 除運算
% ： 求餘數運算
+ ： 加運算
- ： 減運算
>>： 右移
<<： 左移
< ： 小於
> ： 大於

<= ：小於等於
>= ：大於等於
== ：等於
!= ：不等於
& ： 位”與”(AND)運算
^ ： 位”異或”(XOR)運算
| ： 位”或”(OR)運算
&& ： 位元組”與”(AND)運算
|| ： 位元組”或”(OR)運算
= ： 付值
+= ： 加上並附值
-= ： 減去並附值
*= ： 乘上並附值
/= ： 除去並附值
%= ： 求余後並附值
<<= ： 左移後並附值
>>= ： 右移後並附值
^= ： 位 XOR 後並附值
&= ： 位 AND 後並附值
|= ： 位 OR 後並附值
$ ： 取 PC 值