BASIC Commander® & InnoBASIC™ Workshop

補充指令手冊

版本 0.1

目錄

#DEFINE 相關指令

1. **#DEFINE *Parameter* {*Value*}** 使用者可以定義參數為固定常數值，或是定義該參數為有效

2. **#IFDEF *Parameter*** 判斷參數是否有效，若有效則執行下面的程式，直到#ENDIF, #ELSEIF, 或#ELSE

3. **#IFNDEF *Parameter*** 判斷參數是否有被定義，如果沒有定義，就執行下面的程式，直到#ENDIF

4. **#ELSEIF *Parameter*** 在有#IFDEF 起始的程式段落之後，判斷參數是否有效，若有效則執行下面的程式，直到#ENDIF, #ELSEIF, 或#ELSE

5. **#ELSE** 在有#IFDEF 起始的程式段落之後，判斷參數是否有效，若有效則執行下面的程式，直到#ENDIF

6. **#ENDIF** 在有#IFDEF 起始的程式段落之後，用來結束相關執行程式位置

Example 1: (根據#DEFINE 是否有設定，顯示不同值於終端視窗)

```
#DEFINE DEBUG_MSG
        'Set this line to show bVal1 and mark this to show bVal2
Sub Main()
    Dim bVal1, bVal2 As Byte
#IFDEF DEBUGMSG
    Debug bVal1
#ELSE
    Debug bVal2
#ENDIF
End Sub
```

Example 2: (根據#DEFINE 是否有設定，顯示不同值於終端視窗)

```
#DEFINE TEST        ‘ Mark this line the program will display “Normal Mode”
Sub Main()
#IFDEF TEST
        Debug “Test Mode”, CR
#ENDIF

#IFNDEF TEST
        Debug “Normal Mode”, CR
#ENDIF
End Sub
```

Example 3: (顯示#DEFINE 的參數值於終端視窗)

```
#DEFINE MOTOR_SPD_A 100
#DEFINE MOTOR_SPD_B 102
Sub Main()
        Debug “Motor A speed is “, MOTOR_SPD_A, CR     ‘ Show 100
        Debug “Motor B speed is “, MOTOR_SPD_B, CR     ‘ Show 102
End Sub
```

額外 DEBUG 顯示相關指令

1. **Debugfile *Item* {, *Item*}** 操作方式與 Debug 相同，差異為除了將訊息顯示在終端視窗之外，也同時會顯示訊息在指定的檔案中，在設定裡(工具/偏好設定/終端視窗)可以選擇檔案名稱，以及檔案擺放的位置

2. **DebuginFile Item** 操作方式與 Debugin 相同，差異為不是從終端視窗讀取輸入值，而是從指定的檔案中讀取輸入值，檔案名稱與位置可以在設定中(工具/偏好設定/終端視窗)調整

3. **Keyin *Item*** 與 Debugin 相同，會讀取終端視窗的輸入值，但不須按下"Enter"，就會繼續執行後續的程式

4. **Keyscan *Item*** **與 Keyin 相同**，會讀取終端視窗的輸入值，且不須按下"Enter"，而且不管使用者是否有輸入，都會繼續執行後面的程式，在沒有輸入的情況下，讀取到的值為數值 0

Example 1:

```
Sub Main()
    Dim I, j As Byte

    For I = 1 to 9
        For j = 1 to 9
            Debugfile %DEC3R I * j
        Next

        Debug CR
    Next
End Sub
```

Example 2:

```
Sub Main()
	Dim cKey As Byte

	Do
		Keyin cKey
		Debug %CHR cKey	‘ Show character on terminal screen
	Loop
End Sub
```

Example 3:

```
Sub Main()
	Dim cKey As Byte

	Do
		Keyscan cKey

		If cKey<>0 then
			Debug %CHR cKey	‘ Show character on terminal screen
		End If
	Loop
End Sub
```

SOUND指令

SOUND

SOUND Pin, Duration, Frequency

功能

操作於指定的引腳產生方波訊號

Pin 常數或變數值(0-23)，用來指定方波訊號產生的引腳。對一個24引腳BASIC Commander®，這個引腳值的範圍0~15。

Duration 常數或變數值(0~65535)用來定義產生訊號時間長度。單位為ms。

Frequency 常數或變數值(0~65535)用來定義方波頻率。單位為Hz。

範例

以下程式是以SOUND指令透過P.0引腳去驅動壓電式蜂鳴器發出低音Do，維持5s的例子。(音符Do，所須頻率為523Hz)

```
Sub main( )

    SOUND 0,5000,523

End Sub
```

I2C序列輸出入指令

I2COUT

I2COUT ***SDA, SCL, SlaveID,*** **{** ***Address {\|Length},*** **}** **[** ***OutputData…*** **]** **{** **,Err_Label** **}**

功能

SDA 常數或變數值(0~23)，用來指定產生I2C訊號的SDA引腳。初始時這個引腳會被設成輸入模式，當指令完成時會持續設為輸入模式。對一個24引腳BASIC commander®，這個引腳值的範圍為0~15.

SCL 常數或變數值(0~23)，用來指定產生I2C訊號的SCL引腳。初始時這個引腳會被設成輸入模式，當指令完成時會持續設為輸入模式。對一個24引腳BASIC commander®，這個引腳值的範圍為0~15.

SlaveID 常數或變數值(0~127)，用來指定產生I2C Slave裝置所擁有的唯一ID。

Address 選擇性常數或變數值(0~4,294,967,295)，用來指定產生I2C Slave裝置端所需要的Memory Address。可透過***Length***來決定所需的Address長度(byte長度)

Length 選擇性常數或變數值(0~255)，用來指定所需的Address格式設定。

定義方式：

步驟一: 決定 Address 資料長度	可設定範圍0~4。單位：Byte 註：設定為0時，代表傳送完整Address資料。
步驟二: **決定先寫入High Byte或Low Byte。**	Low Byte First = 0 High Byte First = 128
步驟三: 請將前兩步驟的所得結果相加即為*Length數值*	***Length*** = results of step 1 to 2

Ex：若Length = 2，則資料長度為2Byte由Low Byte先寫入。

若Length = 2 + 128 ，則資料長度為2Byte由High Byte先寫入。

OutputData 一連串常數或變數的清單，用來指定所需要傳輸的資料

OutputData參數格式

Value{|L}	**Value**可以是1~4byte大小的任意常數或變數值，L則為0~255的常數或變數，若有L參數則可決定僅輸出Value參數部份數值(從low byte算起)，L=0時則會輸出完整Value數值，L>Value參數長度時則會以Value參數長度為準
String{|L}	**String**可以是使用者所宣告的任何字串變數，L則為0~255的常數或變數，String輸出會把0當做結束字元，String若有L參數則可決定僅輸出部份String長度(從第0個byte算起)，但也有可能先看到0而結束輸出，L=0時則會輸出完整String長度，L>String長度時則會以String長度為準
Array{|L}	**Array**可以是使用者所宣告的任何矩陣變數，L則為0~255的常數或變數，若有L參數則可決定僅輸出部份Array長度(從第0個byte算起)，L=0時則會輸出完整Array長度，L>Array長度時則會以Array長度為準
%Rep Value{|L}	**Value**可以是1~4byte大小的任意常數或變數值，L則為0~255的常數或變數，若有L參數則決定要將Value數值重複幾次輸出，L=0時等同於L=1會將Value數值輸出一次
TEXT	可直接字串輸出 例如 “Hello World!”

Err_Label 用來指定當I2C資料傳輸錯誤時，強迫跳躍的程式標籤

範例程式:

```
Sub main()

    Dim  SDA          as  Byte
    Dim  SCL          as  Byte
    Dim  SlaveID      as  Byte
    Dim  Address1     as  Byte
    Dim  Address2     as  DWord
    Dim  Value1       as  byte
    Dim  Value2       as  Word
    Dim  Value4       as  DWord
    Dim  MyString     as  String*20
    Dim  MyArray(9)  as  byte = {0,1,2,3,4,5,6,7,8,9}
```

```
Dim   L1              as   byte
Dim   L2              as   byte
Dim   L3              as   byte

SDA=1    ‘將 SDA 設定在 Pin 1
SCL=2    ‘將 SCL 設定在 Pin 2
SlaveID=30
Address1=12
Address2=34562
L1=2
L2=5
L3=130
MyString=”Hello World!”

I2COUT   SDA, SCL, SlaveID, [Value1]        ‘輸出 1byte 的資料
I2COUT   SDA, SCL, SlaveID, [Value2]        ‘輸出 2byte 的資料
I2COUT   SDA, SCL, SlaveID, [Value4]        ‘輸出 4byte 的資料
I2COUT   SDA, SCL, SlaveID, [Value4|L1]     ‘輸出 2byte 的資料
I2COUT   SDA, SCL, SlaveID, [MyString]      ‘輸出”Hello World!”字串資料
I2COUT   SDA, SCL, SlaveID, [MyString|L2]   ‘輸出”Hello”字串資料
I2COUT   SDA, SCL, SlaveID, [MyArray]       ‘輸出{0,1,2,3,4,5,6,7,8,9}的資料
I2COUT   SDA, SCL, SlaveID, [MyArray |L2]   ‘輸出{0,1,2,3,4}的資料
I2COUT   SDA, SCL, SlaveID, [%Rep Value1|L1]
      ‘重複輸出 Value1 的資料兩次
I2COUT   SDA, SCL, SlaveID, [“Hello World!”]
      ‘輸出”Hello World!”字串資料
I2COUT   SDA, SCL, SlaveID, [1234]              ‘輸出 1234 (2 byte)資料
I2COUT   SDA, SCL, SlaveID, Address1, [Value1]
    ‘輸出 1byte 的 Address 和 1byte 的資料
I2COUT   SDA, SCL, SlaveID, Address2, [Value1]
    ‘輸出 4byte 的 Address 和 1byte 的資料
I2COUT   SDA, SCL, SlaveID, Address2|L1, [Value1]
    ‘輸出 2byte 的 Address(Low Byte First)  和 1byte 的資料
I2COUT   SDA, SCL, SlaveID, Address2|L3, [Value1]
    ‘輸出 2byte 的 Address(High Byte First)  和 1byte 的資料
I2COUT   SDA, SCL, SlaveID, [Value1] ,Err_Label
    ‘輸出 1byte 的資料及設定 I2C 傳輸錯誤強迫跳躍標籤
I2COUT   SDA, SCL, SlaveID, Address2|L1 ,[Value1] ,Err_Label
```

```
        ‘輸出 2byte 的 Address 和 1byte 的資料
        ‘及設定 I2C 傳輸錯誤強迫跳躍標籤
    I2COUT    SDA, SCL, SlaveID, [Value1, Value4, MyString|L2, MyArray, MyArray|L2]
        ‘可將資料合併輸出
    I2COUT    SDA, SCL, SlaveID, Address2, [Value1, Value2, 1234] ,Err_Label
        ‘可任意組合I2C傳輸所需設定

Err_Label:
    Debug    “I2C Error! “, CR, CR
    Do
    Loop

End sub
```

I2CIN

I2CIN ***SDA, SCL, SlaveID,*** **{** ***Address {\|Length},*** **}** **[** ***InputData…*** **]** **{** **,Err_Label** **}**

功能

SDA 常數或變數值(0~23)，用來指定產生I2C訊號的SDA引腳。初始時這個引腳會被設成輸入模式，當指令完成時會持續設為輸入模式。對一個24引腳BASIC commander®，這個引腳值的範圍為0~15.

SCL 常數或變數值(0~23)，用來指定產生I2C訊號的SCL引腳。初始時這個引腳會被設成輸入模式，當指令完成時會持續設為輸入模式。對一個24引腳BASIC commander®，這個引腳值的範圍為0~15.

SlaveID 常數或變數值(0~127)，用來指定產生I2C Slave裝置所擁有的唯一ID。

Address 選擇性常數或變數值(0~4,294,967,295)，用來指定產生I2C Slave裝置端所需要的Memory Address。可透過***Length***來決定所需的Address長度(byte長度)

Length 選擇性常數或變數值(0~255)，用來指定所需的Address格式設定。

定義方式：

步驟一: 決定 Address 資料長度	可設定範圍0~4。單位：Byte 註：設定為0時，代表傳送完整Address資料。
步驟二: **決定先寫入High Byte或Low Byte。**	Low Byte First = 0 High Byte First = 128
步驟三: 請將前兩步驟的所得結果相加即為*Length數值*	***Length*** = results of step 1 to 2

Ex：若Length = 2，則資料長度為2Byte由Low Byte先寫入。

若Length = 2 + 128 ，則資料長度為2Byte由High Byte先寫入。

InputData 一連串變數的清單，用來指定所需要接收的資料空間

InputData參數格式

Value{|L}	**Value**可以是1~4byte大小的變數值，L則為0~255的常數或變數，若有L參數則可決定僅接收Value參數部份數值(從low byte算起)，L=0時則會接收完整Value數值，L>Value參數長度時則會以Value參數長度為準
String{|L}	**String**可以是使用者所宣告的任何字串變數，L則為0~255的常數或變數，String接收會把0當做結束字元，String若有L參數則可決定僅接收部份String長度(從第0個byte算起)，但也有可能先看到0而結束輸出，L=0時則會接收完整String長度，L>String長度時則會以String長度為準
Array{|L}	**Array**可以是使用者所宣告的任何矩陣變數，L則為0~255的常數或變數，若有L參數則可決定僅接收部份Array長度(從第0個byte算起)，L=0時則會接收完整Array長度，L>Array長度時則會以Array長度為準
%Skip Value	**Value**可以是0~255(1byte)的常數或變數，可決定要跳過Value數值筆數的資料讀取

Err_Label 用來指定當I2C資料傳輸錯誤時，強迫跳躍的程式標籤

範例程式:

```
Sub main()
    Dim   SDA              as   Byte
    Dim   SCL              as   Byte
    Dim   SlaveID          as   Byte
    Dim   Address1         as   Byte
    Dim   Address2         as   DWord
    Dim   Value1           as   byte
    Dim   Value2           as   Word
    Dim   Value4           as   DWord
    Dim   MyString         as   String*20
    Dim   MyArray(9)       as   byte
    Dim   L1               as   byte
    Dim   L2               as   byte
    Dim   L3               as   byte

    SDA=1     ‘將 SDA 設定在 Pin 1
    SCL=2     ‘將 SCL 設定在 Pin 2
    SlaveID=30
    Address1=12
    Address2=34562
    L1=2
    L2=5
    L3=130

    I2CIN   SDA, SCL, SlaveID, [Value1]          ‘接收 1byte 的資料
    I2CIN   SDA, SCL, SlaveID, [Value2]          ‘接收 2byte 的資料
    I2CIN   SDA, SCL, SlaveID, [Value4]          ‘接收 4byte 的資料
    I2CIN   SDA, SCL, SlaveID, [Value4|L1]  ‘接收 2byte 的資料
    I2CIN   SDA, SCL, SlaveID, [MyString]        ‘接收完整字串資料
    I2CIN   SDA, SCL, SlaveID, [MyString|L2]     ‘僅接收 5byte 資料
    I2CIN   SDA, SCL, SlaveID, [MyArray]         ‘接收完整 Array 的資料
    I2CIN   SDA, SCL, SlaveID, [MyArray |L2]     ‘僅接收 5byte 資料
    I2CIN   SDA, SCL, SlaveID, [%Skip Value1]
        ‘略過 Value1 筆數的資料
    I2CIN   SDA, SCL, SlaveID, Address1, [Value1]
      ‘輸出 1byte 的 Address 和接收 1byte 的資料
    I2CIN   SDA, SCL, SlaveID, Address2, [Value1]
```

```
        ‘輸出 4byte 的 Address 和接收 1byte 的資料
    I2CIN    SDA, SCL, SlaveID, Address2|L1, [Value1]
        ‘輸出 2byte 的 Address(Low Byte First)  和接收 1byte 的資料
    I2CIN    SDA, SCL, SlaveID, Address2|L3, [Value1]
        ‘輸出 2byte 的 Address(High Byte First)  和接收 1byte 的資料
    I2CIN    SDA, SCL, SlaveID, [Value1] ,Err_Label
        ‘接收 1byte 的資料及設定 I2C 傳輸錯誤強迫跳躍標籤
    I2CIN    SDA, SCL, SlaveID, Address2|L1 ,[Value1] ,Err_Label
        ‘輸出 2byte 的 Address 和接收 1byte 的資料
        ‘及設定 I2C 傳輸錯誤強迫跳躍標籤
    I2CIN    SDA, SCL, SlaveID, [Value1, Value4, MyString|L2, MyArray, MyArray|L2]
        ‘可將資料合併接收
    I2CIN    SDA, SCL, SlaveID, Address2, [Value1, Value2] ,Err_Label
        ‘可任意組合I2C傳輸所需設定

Err_Label:
    Debug    “I2C Error! “, CR, CR
    Do
    Loop
End sub
```

UART序列輸出入指令

SEROUT

SEROUT *Tpin, Baudmode,* { *Pace,* } [*OutputData1, OutputData2…..*]

功能

Tpin 常數或變數值(0-23)，用來指定產生Uart訊號的引腳。初始時這個引腳會被設成輸出模式，當指令完成時會持續設為輸出模式。對一個24引腳BASIC commander®，這個引腳值的範圍為0~15.

Baudmode 常數或變數值(0-65535)，用來指定所需要的BaudRate資訊、傳輸模式以及even-parity檢查

如何決定***Baudmode***？

步驟一: 由 Baud Rate 決定所需 Period (bits 0 ~ 11)	Period = 4096- INT(2,000,000 / baud rate) 注意: INT為無條件取至整數 Baud rate最大值請勿超過80,000(bps) Baud rate最小值請勿小於500(bps)
步驟二: 是否需要位元檢查(bit 13)	8-bit/no-parity = 0 7-bit/even-parity = 8192
步驟三: 是否需反相輸出 (bit 14)	True (non-inverted) = 0 Inverted = 16384
步驟四: 選擇driven或是open模式輸出 (bit 15)	Driven = 0 Open = 32768
步驟五: 請將前四步驟的所得結果相加即為*Baudmode數值*	***Baudmode*** = results of step 1 to 4

Pace 常數或變數值(0-65535)，用來指定傳輸資料(byte)之間的延遲時間，單位為1ms.

OutputData 一連串常數或變數的清單，用來指定所需要傳輸的資料

OutputData參數格式

Value{|L}	**Value**可以是1~4byte大小的任意常數或變數值，L則為0~255的常數或變數，若有L參數則可決定僅輸出Value參數部份數值(從low byte算起)，L=0時則會輸出完整Value數值，L>Value參數長度時則會以Value參數長度為準
String{|L}	**String**可以是使用者所宣告的任何字串變數，L則為0~255的常數或變數，String輸出會把0當做結束字元，String若有L參數則可決定僅輸出部份String長度(從第0個byte算起)，但也有可能先看到0而結束輸出，L=0時則會輸出完整String長度，L>String長度時則會以String長度為準
Array{|L}	**Array**可以是使用者所宣告的任何矩陣變數，L則為0~255的常數或變數，若有L參數則可決定僅輸出部份Array長度(從第0個byte算起)，L=0時則會輸出完整Array長度，L>Array長度時則會以Array長度為準
%Rep Value{|L}	**Value**可以是1~4byte大小的任意常數或變數值，L則為0~255的常數或變數，若有L參數則決定要將Value數值重複幾次輸出，L=0時等同於L=1會將Value數值輸出一次
TEXT	可直接字串輸出 例如 “Hello World!”

範例程式:

```
Sub main()

    Dim   Tpin          as   Byte
    Dim   Baudmode      as   Word
    Dim   Pace          as   Word
    Dim   Value1        as   byte
    Dim   Value2        as   Word
    Dim   Value4        as   DWord
    Dim   MyString      as   String*20
    Dim   MyArray(9)  as   byte = {0,1,2,3,4,5,6,7,8,9}
    Dim   L1            as   byte
    Dim   L2            as   byte
    #Define   BaudRate      38400      ‘可自行設定 值域必需要在 40000~500 之間
    #Define   ParityCheck   8192       ‘定義位元(Parity)檢查
```

```
    #Define   Inverted        16384      ‘定義反相輸出
    #Define   Open            32768      ‘定義 Open Drain 輸出


    Tpin=1    ‘將 UART 輸出設定在 Pin 1
    L1=2
    L2=5
    Pace=10
    MyString=”Hello World!”
    Baudmode= (4096 – (2000000\BaudRate))+ Inverted+ Open
          ‘設定傳輸模式為  38400bps,Inverted,Open,


    SEROUT    Tpin, Baudmode, [Value1]          ‘輸出 1byte 的資料
    SEROUT    Tpin, Baudmode, [Value2]          ‘輸出 2byte 的資料
    SEROUT    Tpin, Baudmode, [Value4]          ‘輸出 4byte 的資料
    SEROUT    Tpin, Baudmode, [Value4|L1]       ‘輸出 2byte 的資料
    SEROUT    Tpin, Baudmode, [MyString]        ‘輸出”Hello World!”字串資料
    SEROUT    Tpin, Baudmode, [MyString|L2]     ‘輸出”Hello”字串資料
    SEROUT    Tpin, Baudmode, [MyArray]         ‘輸出{0,1,2,3,4,5,6,7,8,9}的資料
    SEROUT    Tpin, Baudmode, [MyArray |L2]     ‘輸出{0,1,2,3,4}的資料
    SEROUT    Tpin, Baudmode, [%Rep Value1|L1]    ‘重複輸出 Value1 的資料兩次
    SEROUT    Tpin, Baudmode, [“Hello World!”]          ‘輸出”Hello World!”字串
資料
    SEROUT    Tpin, Baudmode, [1234]                    ‘輸出 1234 (2 byte)資料

    SEROUT    Tpin, Baudmode, [Value1, Value4, MyString|L2, MyArray,
MyArray|L2]
                              ‘可將資料合併輸出
    SEROUT    Tpin, Baudmode, Pace, [%Rep Value1|L1, “Hello World!”,1234]
                              ‘設定資料間隔10ms輸出
End sub
```

SERIN

SERIN *Rpin, Baudmode,* { *Timeout,* } [*InputData…*] { ,Plabel } {,Tlabel }

功能

Rpin 常數或變數值(0-23)，用來指定接收Uart訊號的引腳。初始時這個引腳會被設成輸入模式，當指令完成時會持續設為輸入模式。對一個24引腳BASIC commander®，這個引腳值的範圍為0~15.

Baudmode 常數或變數值(0-65535)，用來指定所需要的BaudRate資訊、傳輸模式以及even-parity檢查

如何決定***Baudmode***？

步驟一: 由 Baud Rate 決定所需 Period (bits 0 ~ 11)	Period = 4096- INT(2,000,000 / baud rate) 注意: INT為無條件取至整數 Baud rate最大值請勿超過40,000(bps) Baud rate最小值請勿小於500(bps)
步驟二: 是否需要位元檢查(bit 13)	8-bit/no-parity = 0 7-bit/even-parity = 8192
步驟三: 是否需反相輸出 (bit 14)	True (non-inverted) = 0 Inverted = 16384
步驟四: 請將前三步驟的所得結果相加即為*Baudmode數值*	***Baudmode*** = results of step 1 to 3

Timeout 常數或變數值(0-65535)，用來指定接收每筆資料(byte)的最大等待時間，單位為1ms.

InputData 一連串變數的清單，用來指定所需要接收的資料空間

InputData參數格式

Value{|L}	**Value**可以是1~4byte大小的變數值，L則為0~255的常數或變數，若有L參數則可決定僅接收Value參數部份數值(從low byte算起)，L=0時則會接收完整Value數值，L>Value參數長度時則會以Value參數長度為準
String{|L}	**String**可以是使用者所宣告的任何字串變數，L則為0~255的

	常數或變數，String接收會把0當做結束字元，String若有L參數則可決定僅接收部份String長度(從第0個byte算起)，但也有可能先看到0而結束輸出，L=0時則會接收完整String長度，L>String長度時則會以String長度為準
Array{|L}	**Array**可以是使用者所宣告的任何矩陣變數，L則為0~255的常數或變數，若有L參數則可決定僅接收部份Array長度(從第0個byte算起)，L=0時則會接收完整Array長度，L>Array長度時則會以Array長度為準
%Skip Value	**Value**可以是0~255(1byte)的常數或變數，可決定要跳過Value數值筆數的資料讀取

利用不同的InputData存放空間接收資料時，將會在轉換不同InputData存放空間時耗費較多處理時間，因此在較快的BaudRate下，若是發送資料的一方在發送資(Byte與Byte之間)的延遲時間過短，有可能造成在雙方在同樣的BaudRate之下，也會發生資料傳輸錯誤的情形，解決方法如下:

1. **增加發送端Pace時間(byte與byte之間的延遲時間)**
2. **同時降低雙方BaudRate設定**
3. **將不同資料存放空間整合成一個資料存放空間**

PLabel 用來指定當位元檢查(parity check)錯誤時，強迫跳躍的程式標籤

TLabel 用來指定當等待時間逾期，強迫跳躍的程式標籤

範例程式:

```
Sub main()

    Dim   Rpin              as   Byte
    Dim   Baudmode          as   Word
    Dim   Timeout           as   Word
    Dim   Value1            as   byte
    Dim   Value2            as   Word
    Dim   Value4            as   DWord
    Dim   MyString          as   String*20
    Dim   MyArray(9)  as    byte
    Dim   L1                as   byte
    Dim   L2                as   byte
    #Define   BaudRate      38400      '可自行設定 值域必需要在 40000~500 之間
```

```
    #Define   ParityCheck    8192       ‘定義位元(Parity)檢查
    #Define   Inverted       16384      ‘定義反相輸入

    Rpin=1    ‘將 UART 輸入設定在 Pin 1
    L1=2
    L2=5
    Timeout=100
    Baudmode= (4096 – (2000000\BaudRate)) + ParityCheck + Inverted
        ‘設定傳輸模式為  38400bps,Inverted

    SERIN    Rpin, Baudmode, [Value1]            ‘接收 1byte 的資料
    SERIN    Rpin, Baudmode, [Value2]            ‘接收 2byte 的資料
    SERIN    Rpin, Baudmode, [Value4]            ‘接收 4byte 的資料
    SERIN    Rpin, Baudmode, [Value4|L1]  ‘接收 2byte 的資料
    SERIN    Rpin, Baudmode, [MyString]          ‘接收完整 MyString 字串資料
    SERIN    Rpin, Baudmode, [MyString|L2]       ‘僅接收 5byte 資料
    SERIN    Rpin, Baudmode, [MyArray]           ‘接收完整 myArray 資料
    SERIN    Rpin, Baudmode, [MyArray |L2]       ‘僅接收 5byte 的資料
    SERIN    Rpin, Baudmode, [%Skip Value1]      ‘略過 Value1 筆數的資料
    SERIN    Rpin, Baudmode, [Value1, Value4, MyString|L2, MyArray, MyArray|L2]
        ‘可將資料合併接收
    SERIN    Rpin, Baudmode, [Value2], Parity_Error_Label
        ‘設定位元檢查(Parity check)錯誤時  強迫跳躍的程式標籤
    SERIN    Rpin, Baudmode, Timeout, [Value2], Timeout_Label
        ‘設定等待資料逾時時  強迫跳躍的程式標籤
   SERIN    Rpin, Baudmode, Timeout, [Value2], Parity_Error_Label, Timeout_Label
        ‘可同時設定位元檢查錯誤及等待逾時的跳躍程式標籤

Parity_Error_Label:
    Debug “Parity Check Error!”,CR
Timeout_Label:
    Debug “Timeout!”,CR

                                ‘設定資料間隔10ms接收
End sub
```

CATCAN AI Servo 相關指令

CATCAN 智能伺服機函數分類

- 資訊函數：這些函數可回報伺服機相關資訊。如：溫度、電壓、運動速度、目前位置、目的地位置。

Catcan_GetCurrentPos(SDA,SCL,ID,Position) 取得目前的座標的 ADC 值

參數說明

SDA：為 SDA Pin，為 Basic Commander 中的 Pin 名。
SCL：為 SCLP Pin，須為 Basic Commander 中的 Pin 名。
ID：為 SmartServo 的 ID（10 進位）。
Position：為 SmartServo 的運動目的做標範圍為 0-1024，但若是啟動了邊界保護機制，則就僅能在限定的邊界內運動超過伺服機則不會接受該任務，會以邊界為主。

```
Sub main()

Dim Position As Word

        Do
        Catcan_SetPos(12,13,16,0)
        Pause 1000

        Catcan_SetPos(12,13,16,1024)
        Pause 1000
        Loop
End Sub
```

Catcan_GetCurrentPos(SDA,SCL,ID,Position) 取得目前的座標參數值

參數說明

SDA：為 SDA Pin，為 Basic Commander 中的 Pin 名。
SCL：為 SCLP Pin，須為 Basic Commander 中的 Pin 名。
ID：為 SmartServo 的 ID（10 進位）。

Position：為輸出用參數，將變數帶入此欄位之後呼叫此參數可將數伺服機目前的位置傳出。

```
Sub main()

Dim Position As Word

        Do
        Catcan_SetPos(12,13,16,0)
        Catcan_GetCurrentPos(12,13,16,Position)
        Debug "Current position and the variable I is ",    Position
        Debug CR
        Pause 1000

        Catcan_SetPos(12,13,16,1024)
        Catcan_GetCurrentPos(12,13,16,Position)
        Debug "Current position and the variable I is ",    Position
        Debug CR
        Pause 1000
        Loop
End Sub
```

Catcan_GetDestinedPos(SDA,SCL,ID,Position) 取得目的地座標參數值

參數說明

SDA：為 SDA Pin，為 Basic Commander 中的 Pin 名。

SCL：為 SCLP Pin，須為 Basic Commander 中的 Pin 名。

ID：為 SmartServo 的 ID（10 進位）。

Position：將資料傳輸至參數，可供輸出使用。

```
Sub main()
Dim Position As Word

        Do
        Catcan_SetPos(12,13,16,0)
        Catcan_GetDestinedPos(12,13,16,Position)
        Debug "Destine position and the variable I is ",    Position
        Debug CR
```

```
        Pause 1000

        Catcan_SetPos(12,13,16,1024)
        Catcan_GetDestinedPos(12,13,16,Position)
        Debug "Destine position and the variable I is ",    Position
        Debug CR
        Pause 1000
        Loop
End Sub
```

Catcan_GetSpeed(SDA,SCL,ID,Speed) 取得目前 Servo 的運動速度相對值

參數說明

SDA：為 SDA Pin，為 Basic Commander 中的 Pin 名。
SCL：為 SCLP Pin，須為 Basic Commander 中的 Pin 名。
ID：為 SmartServo 的 ID（10 進位）。
Speed：將資料傳輸至參數，可供輸出使用。

```
Sub main()

Dim Speed As Word

        Do
        Catcan_SetPos(12,13,16,0)
        Catcan_GetSpeed(12,13,16,Speed)
        Debug "Servo Speed is ",    Speed
        Debug CR
        Pause 1000

        Catcan_SetPos(12,13,16,1024)
        Catcan_GetSpeed(12,13,16,Speed)
        Debug "Servo Speed    ",    Speed
        Debug CR
        Pause 1000
        Loop
End Sub
```

Catcan_GetCurrent(SDA,SCL,ID,Current) 取得目前 Servo 的附載電流相對值

參數說明

SDA：為 SDA Pin，為 Basic Commander 中的 Pin 名。

SCL：為 SCLP Pin，須為 Basic Commander 中的 Pin 名。

ID：為 SmartServo 的 ID（10 進位）。

Current：取得伺服機上的附載。

執行下列的程式，然後將伺服裝上附載，在有附載的情況下可以發現電流質會變化。

※注意事項：因為伺服馬達需要再付載足夠大的情況下才會啟動馬達，因此載體需要有一定重量的負載。

```
Sub main()

Dim Current As Word

        Do
        Catcan_GetCurrent(12,13,16,Current)
        Debug "Servo Speed is ",    Current
        Debug CR
        Pause 1000
        Catcan_GetCurrent(12,13,16,Current)
        Debug "Servo Speed    ",    Current
        Debug CR
        Pause 1000
        Loop
End Sub
```

Catcan_GetVoltage(SDA,SCL,ID,Voltage) 取得目前 Servo 的電壓相對質

參數說明

SDA：為 SDA Pin，為 Basic Commander 中的 Pin 名。

SCL：為 SCLP Pin，須為 Basic Commander 中的 Pin 名。

ID：為 SmartServo 的 ID（10 進位）。

Voltage：取得伺服機上的電壓值。

```
Sub main()

Dim Voltage As Word

        Do
        Catcan_GetCurrent(12,13,16,Voltage)
        Debug "Servo Voltage is ",   Voltage
        Debug CR
        Pause 1000

        Catcan_GetCurrent(12,13,16,Voltage)
        Debug "Servo Voltage   ",   Voltage
        Debug CR
        Pause 1000
        Loop
End Sub
```

- 運動函數：設定目的地，啟動/解除伺服機內的 PWM 輸出。

Catcan_SetPos(SDA,SCL,ID,Position) 設定 SmartServo 的目的作標 ADC 值

參數說明
SDA：為 SDA Pin，為 Basic Commander 中的 Pin 名。
SCL：為 SCLP Pin，須為 Basic Commander 中的 Pin 名。
ID：為 SmartServo 的 ID（10 進位）。
Position：目的地的座標。

```
Sub main()

        Do
        Catcan_SetPos(12,13,16,200)
        Pause 1000

        Catcan_SetPos(12,13,16,1200)
        Pause 1000
        Loop
End Sub
```

Catcan_EnablePWM(SDA,SCL,ID)　　啟動馬達輸出
Catcan_DisablePWM(SDA,SCL,ID)　　啟動馬達輸出
SDA：為 SDA Pin，為 Basic Commander 中的 Pin 名。
SCL：為 SCLP Pin，須為 Basic Commander 中的 Pin 名。
ID：為 SmartServo 的 ID（10 進位）。
※注意事項：智能伺服起動時會以當時角度為初始位置，因此若在 PWM 為輸出使用者將感受不到變化。

```
Sub main()

        Do
        Catcan_EnablePWM(12,13,16) 'Motor with Power
        Pause 2000

        Catcan_DisablePWM(12,13,16,1200) 'Motor without Power
        Pause 2000
        Loop
End Sub
```

- 工程函數：設定伺服機的參數 ID（識別碼），掃描線上伺服機 ID。

Catcan_ChangeAddress(SDA,SCL,ID,NewID)
Catcan_EnablePWM(SDA,SCL,ID)　　啟動馬達輸出
Catcan_DisablePWM(SDA,SCL,ID)　　啟動馬達輸出

參數說明
SDA：為 SDA Pin，為 Basic Commander 中的 Pin 名。
SCL：為 SCLP Pin，須為 Basic Commander 中的 Pin 名。
ID：原有為 SmartServo 的 ID（10 進位）。
NewID：新為 SmartServo 的 ID（10 進位）。
※注意事項：使用此功能後，後續的控制須採用 NEW_ID 參數。

```
Sub main()

Dim ID As Word

        Catcan_ChangeAddress(12,13,16,18)
```

```
        Pause 500

End Sub
```

Status=Catcan_ScanServoID(SDA,SCL,ID)
SDA：為 SDA Pin，為 Basic Commander 中的 Pin 名。
SCL：為 SCLP Pin，須為 Basic Commander 中的 Pin 名。

```
Sub main()

Dim ID As Word

        Do
        Catcan_ScanServoID(12,13,ID)
        Debug "Servo ID is ",   ID
        Debug CR
        Pause 500
        Loop
End Sub
```