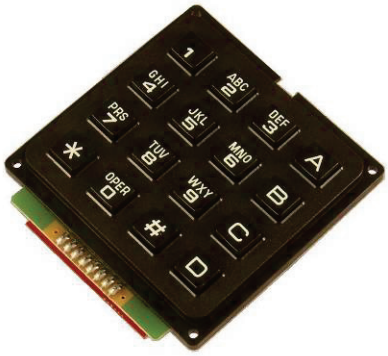
Keypad A

4x4　16 鍵按鍵輸入模組

版本: V2.0



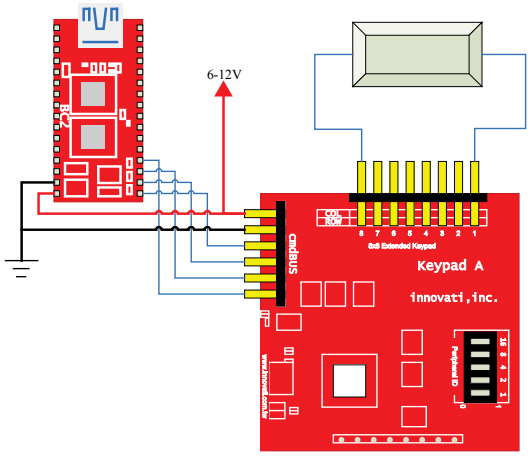
產品介紹: 利基 Keypad A 模組提供多樣化輸入功能，並且可透過簡單的聯接，直接由利基之 Basic Commander 操控各項應用。藉由設定不同的輸入模式，可以快速達到模擬各項日常生活常用的輸入介面，包括計算機的數字輸入，手機的英文輸入，工程十六位數的輸入，甚至自訂各鍵盤的設定回傳值。除了輸入選項，另外可自訂彈跳消除(debounce)時間，避免將機構的彈跳誤判為按鍵輸入，也可以設定自動重複輸入(auto-repeat)，讓按鍵久按時重複產生輸入。

應用方向:

- 搭配顯示 LCD，透過設定模式，可以快速模擬計算機功能。
- 密碼輸入可透過鍵盤，甚至設定英文大小寫的判斷。
- 自訂輸入可以藉由軟體設定，偵測不同按鍵，讓模組啟動多樣化的運作。
- 藉由設定久按模式，可將鍵盤當做八向鍵，作為有線遙控操縱器。
- 可透過 I2C 方式，下達指令。

產品特色:

- 4x4 輸入鍵盤，可以設定九種不同輸入模式。
  - 鍵值模式 (預設)
  - 十六進制模式
  - 數字模式
  - 大寫英文輸入
  - 小寫英文輸入
  - 記號模式
  - 計算機模式
  - 使用者自訂模式
  - 擴充鍵盤輸入模式
- 使用者可根據個人習性，設定防彈跳判斷值，避免重複輸入的發生。
- 藉由擴充腳位，產品能再外接按鍵擴充輸入
- 鍵盤判定透過程式設計不同，分為事件模式與輪詢模式。
- 久按鍵盤可設定重覆產生按鍵判定的速率。



連接方式: 直接將 ID 開關撥至欲設定的編號，再將 cmdBUS 連接至 Basic Commander 上對應的腳位，就可透過 Basic Commander 執行操作。如果需要新增其他按鈕，只需將按鈕與 8x8 Extended Keypad 連接，就可以擴充多達 64 個額外的按鍵。

產品規格:

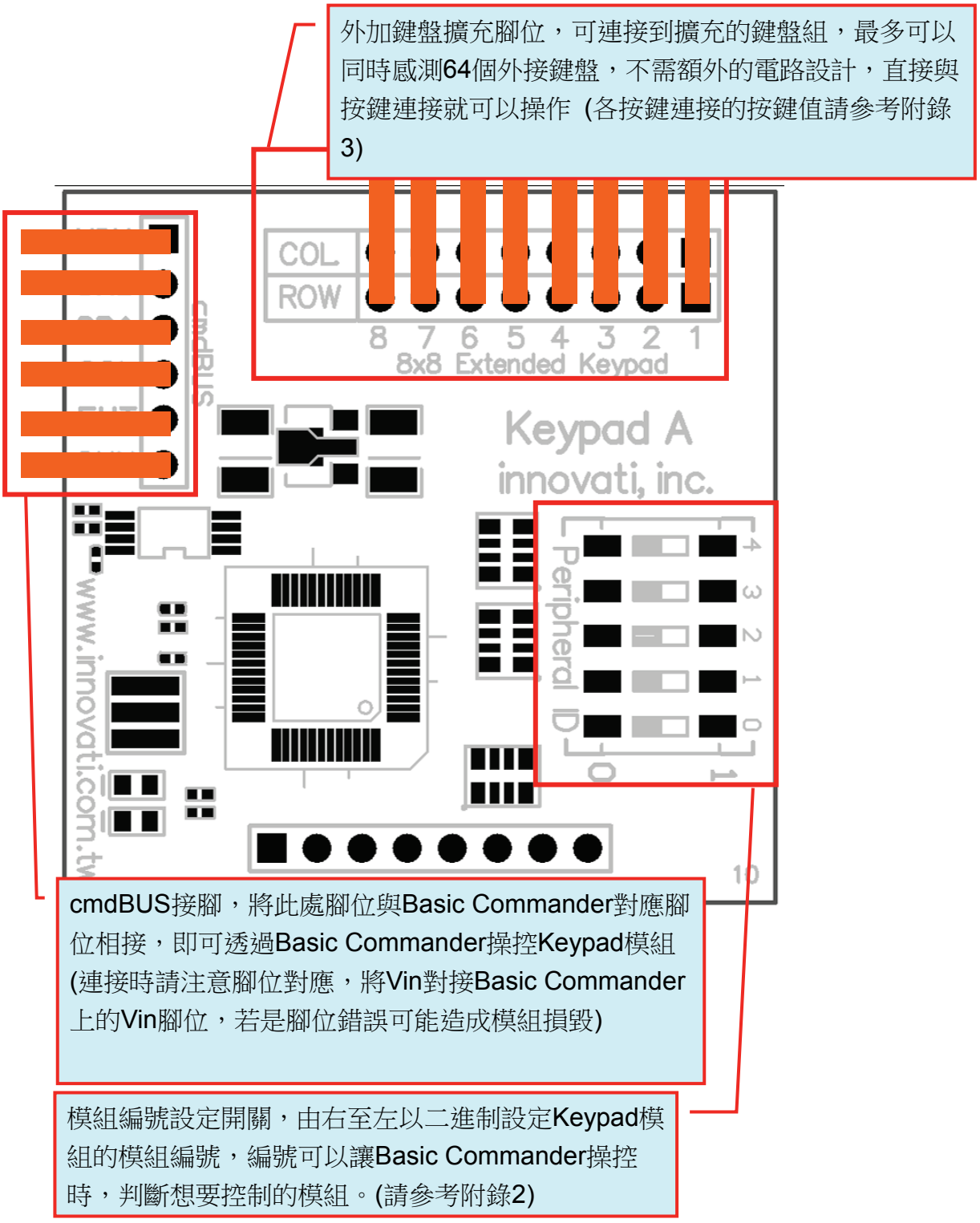


圖 1: 模組腳位與開關介紹

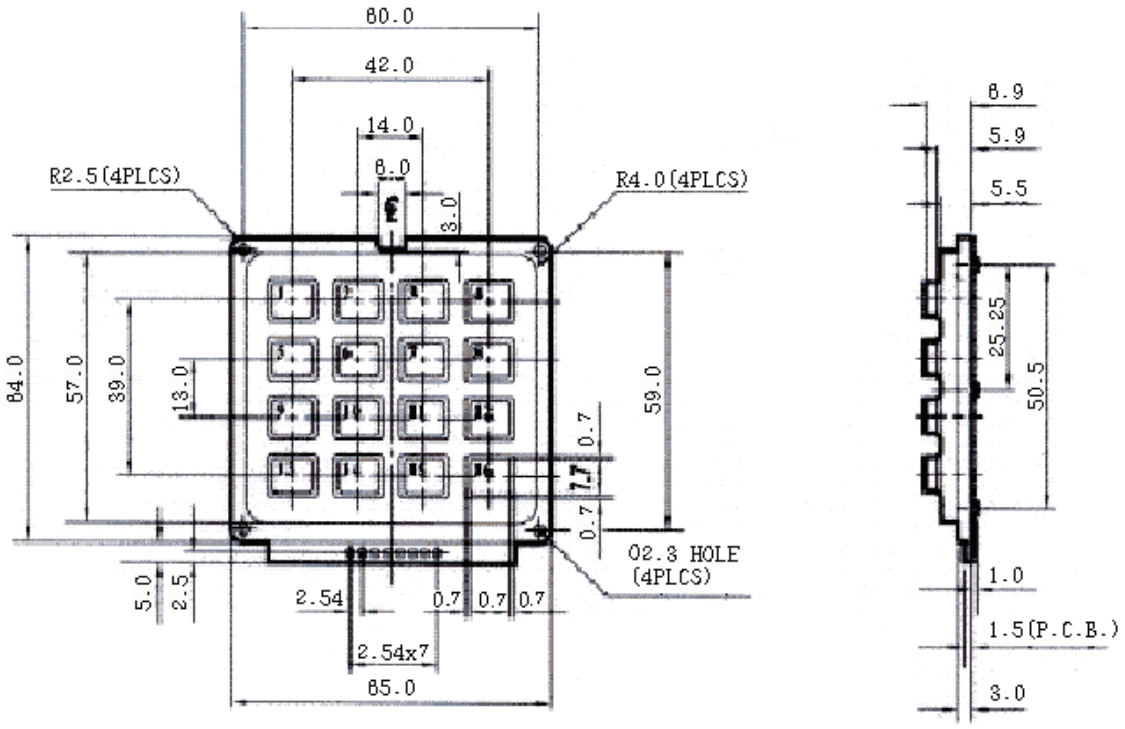


圖 2: Keypad 鍵盤規格 (單位 mm)

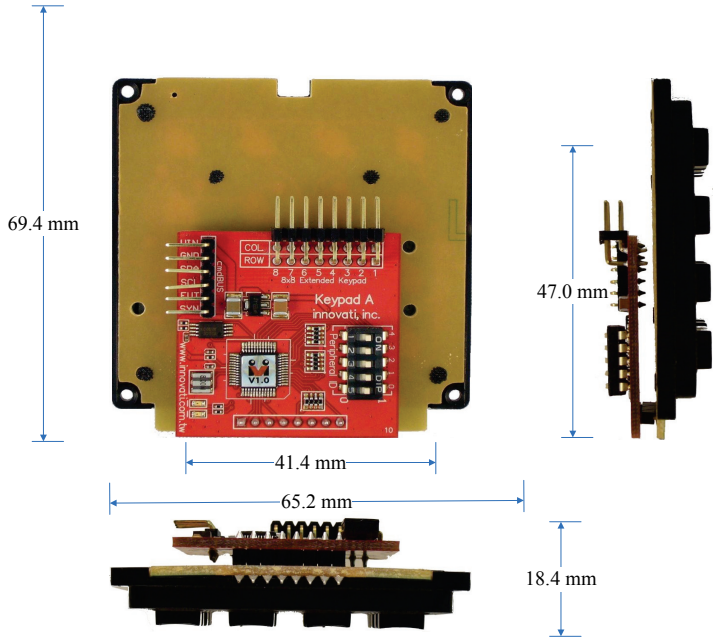


圖 3: Keypad 外觀

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{IN}	Conditions				
I_{IN}	Operating Current	7.5	—	—	6.5	—	mA

表 1: 工作電流特性 (於 25 ℃之環境)

操作注意事項:

按鍵可承受約 1,000,000 次的按壓動作。

操作溫度　　0 ℃~ 70 ℃

儲存溫度　　-20 ℃~ 80 ℃

模組下達指令的方式可分為兩種：cmdBUS、I2C 控制方式

cmdBUS 指令表:

下面的指令表是專供控制 Keypad A 模組的各種指令，必要輸入的指令名稱與參數，以粗底或粗斜體表示，粗體的文字在輸入時請不要更改，粗斜體的文字請自行定義適當格式的參數填入。輸入時請注意 innoBASIC Workshop 大寫與小寫會視為相同字。

在執行 Keypad A 指令前，請先於程式開頭定義對應參數與編號，例:

Peripheral *ModuleName* As KeypadA @ *ModuleID*

I2C 通訊協議(Protocol):

為了使更廣泛的使用者能控制模組，提供了部份指令的通訊協議讓使用者應用。透過通訊規格，使用者可使用 I2C 通訊協議為模組下達命令。

通訊協議常見的封包如下：

MID：模組 ID 編號，空間大小為 Byte 的變數。對應於硬體的指播開關。

CID：命令 ID 編號，空間大小為 Byte 的變數。依不同命令而改變。

CheckSum1：驗證位元_1，空間大小為 Byte 的變數。

定義方式：255 – (MID * 2) – CID

CheckSum2：驗證位元_2，空間大小為 Byte 的變數。

定義方式：255 – (CheckSum1~CheckSum2 之間的變數總和)

CheckSum3：驗證位元_3，空間大小為 Byte 的變數。

定義方式：255 – MID-(MID~CheckSum3 之間的變數總和)

Dummy：虛設位元，可為任意變數。空間大小為 Byte 的變數。

於通訊規格每筆資料空間大小階為 Byte，若資料空間大小超過一個 Byte 時，需將資料拆開，並由 Low Byte 開始傳送。

Ex:傳送資料 Temp 為一筆空間大小為 Word 的資料,則需將 Temp 拆開,分為 Temp_L、Temp_H，並且先傳送 Temp_L。

Ex1 模組編號為 2，命令編號為 153，傳送參數 Byte 為 100，通訊協議為 MID+CID+CheckSum1+Byte+CheckSum2+Dummy 則：

MID = 2

CID = 153

CheckSum1 = 255 – (2*2) – 153 = 98

Byte = 100

CheckSum2 = 255 – 100

Dummy = 0~255 之間的任意數

Ex2 模組編號為 2，命令編號為 153，傳送參數 Temp 為 511，通訊協議為 MID+CID+CheckSum1+Temp_L+Temp_H+CheckSum2+Dummy 則：

MID = 2

CID = 153

CheckSum1 = 255 – (2*2) – 153 = 98

Temp_L = 255，Temp_H = 1

CheckSum2 = 255 – Temp_L–Temp_H = 255

Dummy = 0~255 之間的任意數

指令格式	指令功能
CmdBUS： **ClearKeyBuffer ()** **I2C：** **MID+101+CheckSum1+Dummy**	清除按鍵儲存器中的所有按鍵值
DisableBufferFullEvent()	解除儲存器溢位事件通知
DisableKeyPressedEvent()	解除按鍵動作事件通知
EnableBufferFullEvent()	啟動儲存器溢位事件通知
EnableKeyPressedEvent()	啟動按鍵動作事件通知
GetCustomTable(*CustomTable*)	取得 CustomTable 所儲存的按鍵值，***CustomTable*** 需要提供可儲存 16 個值的陣列，每個值會回傳 0~255 之間著整數值
GetCustomTableIndex(*Index*)	取得目前所使用的 CustomTable 的編號，***Index*** 會回傳 0~15 之間的整數值 *1
GetDebounceTime(*Time*)	取得設定的防彈跳值，***Time*** 會回傳 0~255 之間的整數值
CmdBUS： ***Status* = GetKeyID(*KeyID*)** **I2C：** **Out：** **MID+90+CheckSum1+Dummy** **In：** **MID+Status+KeyID+CheckSum3**	取得鍵盤狀態存放於 ***Status***，並取得按鍵值存放於 ***KeyID*** 中，***Status*** 有三種狀態，0 代表沒有按鍵觸發，1 代表最後按鍵值，2 代表暫時的按鍵值，***Status*** 會在某些模式下出現 2 的狀態，代表使用者還可以再按下按鍵切換不同回傳值，請參考 **SetKeypadMode** 的說明，***KeyID*** 會回傳 0~255 之間的整數值 *1
CmdBUS： **GetKeypadMode(*Mode*)** **I2C：** **Out：** **MID+89+CheckSum1+Dummy** **In：** **MID+Mode+CheckSum3**	取得目前設定的鍵盤模式，***Mode*** 會回傳 0~8 之間的整數值

<table>
<tr><td>GetRepeatRate(Rate)</td><td>取得目前所設定的重複輸入判定速率值，Rate 會回傳 0~255 之間的整數值</td></tr>
<tr><td>GetRepeatTime(Time)</td><td>取得目前所設定的重複輸入判定時間值，Time 會回傳 0~255 之間的整數值</td></tr>
<tr><td>LoadCustomTable(Index)</td><td>根據 Index 值，取出儲存於 EEPROM 中的使用者自訂輸入，Index 請輸入 0~15 之間的整數值</td></tr>
<tr><td>SaveCustomTable (Index)</td><td>將目前所設定的使用者自訂輸入儲存於 Index 所指定的位址，Index 請輸入 0~15 之間的整數值，共十六個自訂鍵盤值可設定</td></tr>
<tr><td>SetCustomTable(KeyID)</td><td>將陣列 KeyID 所對應的各設定值，訂為自訂鍵盤回傳值，KeyID 需要提供設定共 16 個值的陣列，每個值可輸入 0~255 之間的整數值，KeyID 中的設定值將依序對應到鍵盤上的 1, 2, 3, A, 4, 5, 6, B, 7, 8, 9, C, *, 0, #, D</td></tr>
<tr><td>SetDebounceTime(Time)</td><td>將 Time 設定為防彈跳值，可輸入 0~255 之間的整數值，單位為 10 ms。設定只對外接按鈕有效，不會影響到模組所含的鍵盤</td></tr>
<tr><td>CmdBUS：
SetKeypadMode(Mode)
I2C：
MID+88+CheckSum1
+Mode+CheckSum2+Dummy</td><td>設定鍵盤輸入模式，Mode 可設定為 0~8，分別代表的鍵盤操作如下: *2
0: 鍵值模式 (按鍵傳回順序的 Key ID 值)
<table>
<tr><td>0</td><td>1</td><td>2</td><td>3</td></tr>
<tr><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td>8</td><td>9</td><td>10</td><td>11</td></tr>
<tr><td>12</td><td>13</td><td>14</td><td>15</td></tr>
</table>
1: 十六進制模式 (按鍵傳回 0~F 值)
<table>
<tr><td>1</td><td>2</td><td>3</td><td>A</td></tr>
<tr><td>4</td><td>5</td><td>6</td><td>B</td></tr>
<tr><td>7</td><td>8</td><td>□</td><td>C</td></tr>
<tr><td>F</td><td>0</td><td>E</td><td>D</td></tr>
</table>
2: 數字模式 (按鍵傳回 0~9 之 ASCII 碼值)
<table>
<tr><td>1</td><td>2</td><td>3</td><td>(11)</td></tr>
<tr><td>4</td><td>5</td><td>6</td><td>(12)</td></tr>
<tr><td>7</td><td>8</td><td>9</td><td>(13)</td></tr>
<tr><td>*</td><td>0</td><td>#</td><td>(14)</td></tr>
</table></td></tr>
</table>

<table>
<tr>
<td></td>
<td>3: 大寫英文輸入(按鍵傳回 0~9 之 ASCII 碼值)
<table>
<tr><td>!?</td><td>ABC</td><td>DEF</td><td>F1</td></tr>
<tr><td>GHI</td><td>JKL</td><td>MNO</td><td>F2</td></tr>
<tr><td>PQRS</td><td>TUV</td><td>WXYZ</td><td>F3</td></tr>
<tr><td>,</td><td>space</td><td>.</td><td>F4</td></tr>
</table>
4: 小寫英文輸入(按鍵傳回 0~9 之 ASCII 碼值)
<table>
<tr><td>!?</td><td>abc</td><td>def</td><td>F1</td></tr>
<tr><td>ghi</td><td>jkl</td><td>mno</td><td>F2</td></tr>
<tr><td>□qrs</td><td>tuv</td><td>wxyz</td><td>F3</td></tr>
<tr><td>,</td><td>space</td><td>.</td><td>F4</td></tr>
</table>
5: 記號模式(按鍵傳回 0~9 之 ASCII 碼值)
<table>
<tr><td>! ?</td><td>% @ $</td><td>+ - =</td><td>F1</td></tr>
<tr><td>/ \ _</td><td>( ) &</td><td>< > |</td><td>F2</td></tr>
<tr><td>: ; `</td><td>[ ] ^</td><td>‘ “ “</td><td>F3</td></tr>
<tr><td>, *</td><td>space</td><td>. #</td><td>F4</td></tr>
</table>
6: 計算機模式(按鍵傳回 0~9 之 ASCII 碼值)
<table>
<tr><td>1</td><td>2</td><td>3</td><td>/</td></tr>
<tr><td>4</td><td>5</td><td>6</td><td>*</td></tr>
<tr><td>7</td><td>8</td><td>9</td><td>-</td></tr>
<tr><td>.</td><td>0</td><td>=</td><td>+</td></tr>
</table>
7: 使用者自訂模式(按鍵傳回 0~9 之 ASCII 碼值)
<table>
<tr><td>自訂 0</td><td>自訂 1</td><td>自訂 2</td><td>自訂 3</td></tr>
<tr><td>自訂 4</td><td>自訂 5</td><td>自訂 6</td><td>自訂 7</td></tr>
<tr><td>自訂 8</td><td>自訂 9</td><td>自訂 10</td><td>自訂 11</td></tr>
<tr><td>自訂 12</td><td>自訂 13</td><td>自訂 14</td><td>自訂 15</td></tr>
</table>
8: 擴充鍵盤輸入模式
<table>
<tr><td>240</td><td>241</td><td>242</td><td>243</td></tr>
<tr><td>244</td><td>245</td><td>246</td><td>247</td></tr>
<tr><td>248</td><td>249</td><td>250</td><td>251</td></tr>
<tr><td>252</td><td>253</td><td>254</td><td>255</td></tr>
</table>
*3</td>
</tr>
<tr>
<td>SetRepeatRate (Rate)</td>
<td>將 Rate 設定為重複輸入判定速率值，可以輸入 0~255 之間的整數值，單位為 10 ms *4</td>
</tr>
</table>

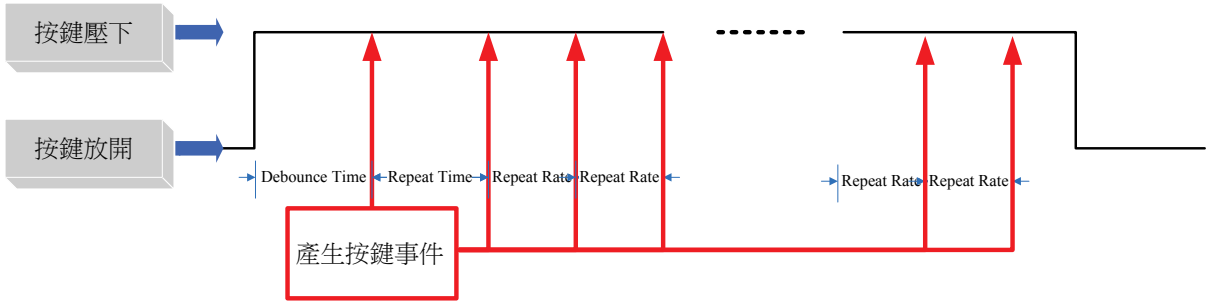
SetRepeatTime(*Time*)	將 ***Time*** 設定為重複輸入判定時間值，可以輸入 0~255 之間的整數值，單位為 10 ms *4

*1:在未設定 Repeat 時，在按鍵被按下後執行 **GetKeyID**，則 ***Status*** 就會被清為 0，即使按鍵仍在按下的狀態，若是連續按下多個按鍵，而未執行 **GetKeyID** 時，按鍵值會保留，當按鍵數量超過可儲存的容量時(32 個按鍵值)，後面按下的按鍵值將被忽略

*2:在模式 3~5，當按下按鍵時，會先將按鍵的狀態值設為 2，如果使用者在此時再按下按鍵，就會依模式所訂的按鍵順序切換不同回傳值，等到隔了 Repeat Time 所設定的時間，才會將狀態轉為 1；如果使用者在按鍵狀態 2 時，再按下不同按鍵，就會先將狀態 2 的按鍵狀態改回 1 回傳，再將最後按下的按鍵狀態改為 2

*3:擴充按鍵相對應的按鍵 ID 值，請參考附錄 3

*4: Repeat Time 與 Repeat Rate 的設定效果如下



在按鈕按下不放後，相隔 Repeat Time 設定的時間，會再產生一次按鍵壓下的事件，而後每隔 Reapeat Rate 所設定的時間，產生同樣的事件直到按鈕放開為止

模組提供應用事件:

事件名稱 (Event)	啟動條件
KeyBufferFullEvent	在執行 **EnableBufferFullEvent** 後，若是模組所儲存的按鍵被按下事件超過儲存容量時。(模組最多儲存 32 個按鍵值)
KeyPressedEvent	在執行 **EnableKeyPressedEvent** 後，若是有模組上的按鍵被按下，就會啟動。

範例程式:

```
Peripheral myKeypad As KeypadA @ 0                  '   設定模組編號為 0

Dim PressKeyD As Byte                               '   儲存是否按下 D 鍵
Dim KeyStatus As Byte                               '   判斷是否有取得鍵值
Dim KeyID As Byte                                   '   儲存取得的鍵值
Dim RepeatTime As Byte                              '   儲存所取得的 RepeatTime 設定值
Dim RepeatRate As Byte                              '   儲存所取得的 RepeatRate 設定值
Dim DebounceTime As Byte                            '   儲存所取得的 DebounceTime 設定值
Dim RepeatCount As Byte                             '   儲存重複按鍵次數
Dim CustomTable(15) As Byte                         '   儲存自訂回傳鍵值的陣列
Dim i As Byte                                       '   儲存迴圈計數次數

Sub Main()
    Debug CLS

KEYID_CHECK:
'   下面的迴圈會反覆執行，並根據 KeyStatus 判斷到按鍵被按下，
'   顯示所按下的按鍵訊息在 Terminal Window，
'   若判斷到所按下的按鍵是"D"，就會跳出迴圈
'   此部份使用 Event 判斷鍵值
    myKeypad.SetKeypadmode(0)                       '   設定鍵盤模式為 0，預設模式
    Pause 100
    myKeypad.GetKeyID(KeyID)                        '   取得開始的按鍵狀態
    RepeatCount=0
    KeyStatus=0
    myKeypad.EnableKeyPressedEvent()                '   啟動按鍵壓下事件
    Debug "按下按鍵 D 離開迴圈", CR
    PressKeyD=0

    Do
    Loop Until PressKeyD>0

REPEAT_CHECK:
    myKeypad.GetRepeatTime(RepeatTime)              '   取得系統初始設定的 RepeatTime
    myKeypad.GetRepeatRate(RepeatRate)              '   取得系統初始設定的 RepeatRate

'   顯示系統初始設定的 Repeat Time
    Debug "Repeat Time 目前設定為  ", RepeatTime, " * 10 ms...", CR
'   顯示系統初始設定的 Repeat Rate
    Debug "Repeat Rate 目前設定為  ", RepeatRate, " * 10 ms...", CR
```

```
        myKeypad.SetRepeatTime(50)                      '       設定 RepeatTime 為  500 ms
        myKeypad.SetRepeatRate(2)                       '       設定 RepeatRate 20 ms
        RepeatCount=1

'       下面的迴圈會反覆執行，並根據 KeyStatus 判斷到按鍵被按下。
'       持續按壓超過半秒，就會重複產生按鍵被按下的事件。(因為設定了 RepeatTime)
'       每隔  20 ms 就會再產生一次按鍵的事件。(因為設定了定 RepeatRate)
        Debug "請按住任意鍵", CR

        Do
        Loop Until RepeatCount>100

        myKeypad.DisableKeyPressedEvent()               '       關閉按鍵壓下事件

'       同時設定 RepeatTime 與 RepeatRate 為 0，使長按不產生新按鍵訊息
        myKeypad.SetRepeatTime(0)
        myKeypad.SetRepeatRate(0)

CUSTOM_TABLE:
        For i=0 To 15
                CustomTable(i)=100+i
        Next

        myKeypad.SetCustomTable(CustomTable)            '       將自訂按鍵回傳值設定為 100~115
        myKeypad.SaveCustomTable(0)                     '       將自訂值存到編號 0 的位置
        myKeypad.SetKeypadmode(7)                       '       設定鍵盤模式為 7，自訂模式
        Pause 100
        myKeypad.GetKeyID(KeyID)
        KeyStatus=0

'       下面的迴圈會反覆執行，並根據 KeyStatus 判斷到按鍵被按下，
'       顯示所按下的按鍵訊息在 Terminal Window，
'       若判斷到所按下的按鍵是"D"，就會跳出迴圈
'       可以看到回傳的 Key ID 值已經變為設定的自訂鍵值
'       此處使用 Polling 的方式取得按鍵狀態與按鍵值，並未啟動按鍵事件
        Debug "按下按鍵 D 離開迴圈", CR

        Do
                If myKeypad.GetKeyID(KeyID)<>0 Then
                        Select Case KeyID
                        Case 100 : Debug "按下按鍵  1! (回傳值為 100)", CR
```

```
                Case 101 : Debug "按下按鍵  2! (回傳值為 101)", CR
                Case 102 : Debug "按下按鍵  3! (回傳值為 102)", CR
                Case 103 : Debug "按下按鍵  A! (回傳值為 103)", CR
                Case 104 : Debug "按下按鍵  4! (回傳值為 104)", CR
                Case 105 : Debug "按下按鍵  5! (回傳值為 105)", CR
                Case 106 : Debug "按下按鍵  6! (回傳值為 106)", CR
                Case 107 : Debug "按下按鍵  B! (回傳值為 107)", CR
                Case 108 : Debug "按下按鍵  7! (回傳值為 108)", CR
                Case 109 : Debug "按下按鍵  8! (回傳值為 109)", CR
                Case 110 : Debug "按下按鍵  9! (回傳值為 110)", CR
                Case 111 : Debug "按下按鍵  C! (回傳值為 111)", CR
                Case 112 : Debug "按下按鍵  *! (回傳值為 112)", CR
                Case 113 : Debug "按下按鍵  0! (回傳值為 113)", CR
                Case 114 : Debug "按下按鍵  #! (回傳值為 114)", CR
                Case 115 : Debug "按下按鍵  D! (回傳值為 115)", CR
                End Select

                If KeyID=115 Then
                    Goto KEYID_CHECK
                End If
            End If
        Loop
End Sub

Event myKeypad.KeyPressedEvent()
        KeyStatus=myKeypad.GetKeyID(KeyID)          '       將取得的鍵值放入參數 KeyID 中

        If RepeatCount>100 Then
            Return
        Elseif RepeatCount>0 Then
            RepeatCount+=1
            Debug "累計按鍵數  ", RepeatCount, CR
        Elseif RepeatCount=0 Then
            Select Case KeyID
            Case 0 : Debug "按下按鍵  1!", CR
            Case 1 : Debug "按下按鍵  2!", CR
            Case 2 : Debug "按下按鍵  3!", CR
            Case 3 : Debug "按下按鍵  A!", CR
            Case 4 : Debug "按下按鍵  4!", CR
            Case 5 : Debug "按下按鍵  5!", CR
            Case 6 : Debug "按下按鍵  6!", CR
            Case 7 : Debug "按下按鍵  B!", CR
```

```
            Case 8 : Debug "按下按鍵  7!", CR
            Case 9 : Debug "按下按鍵  8!", CR
            Case 10 : Debug "按下按鍵  9!", CR
            Case 11 : Debug "按下按鍵  C!", CR
            Case 12 : Debug "按下按鍵  *!", CR
            Case 13 : Debug "按下按鍵  0!", CR
            Case 14 : Debug "按下按鍵  #!", CR
            Case 15 : Debug "按下按鍵  D!", CR : PressKeyD=1
            End Select
        End If
End Event
```

附錄

1. 已知問題:
   - 在切換模式時，事件的判定會重設。如果在按鍵按住的情況下切換模式，即使已經關閉 Repeat 的相關設定，仍會收到一個按鍵壓下的事件。

2. 模組編號開關對應編號表:

	0		8		16		24
	1		9		17		25
	2		10		18		26
	3		11		19		27
	4		12		20		28
	5		13		21		29
	6		14		22		30
	7		15		23		31

3. 模式 8，擴充按鍵回傳之按鍵值列表

	ROW 1	ROW 2	ROW 3	ROW 4	ROW 5	ROW 6	ROW 7	ROW 8
COL 1	0	1	2	3	4	5	6	7
COL 2	8	9	10	11	12	13	14	15
COL 3	16	17	18	19	20	21	22	23
COL 4	24	25	26	27	28	29	30	31
COL 5	32	33	34	35	36	37	38	39
COL 6	40	41	42	43	44	45	46	47
COL 7	48	49	50	51	52	53	54	55
COL 8	56	57	58	59	60	61	62	63