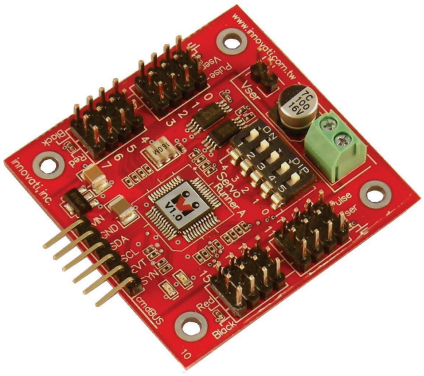
Servo Runner A

十六組伺服機輸出控制模組

版本: V2.0



產品介紹: 利基 Servo Runner A 模組可以一次控制十六個伺服機，並且提供整合好的指令，讓使用者可以直接使用速度或時間，決定伺服機的移動模式。設有多達 250 組記憶體可以儲存伺服機目標位置與移動方式(速度或時間)，讓各種動作輕易組合完成。

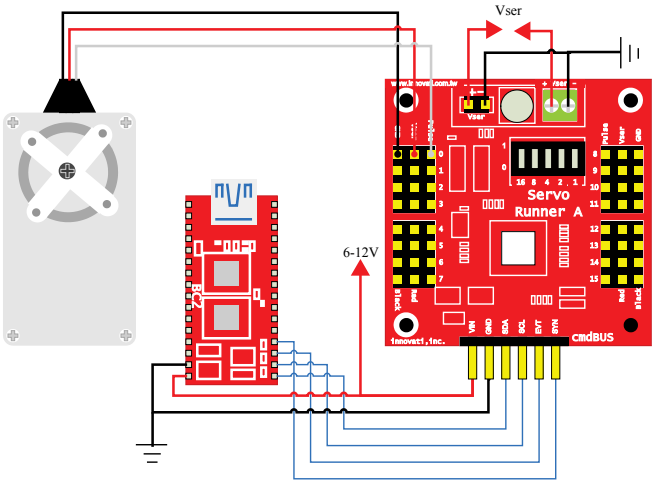
應用方向:

- 各種伺服機的操作與應用，包括機械手臂，機器人關節。

產品特色:

- 十六組伺服機輸出介面，可同時控制十六組伺服機。
- 可控制伺服機位置由 0.5 ms 至 2.5 ms。
- 軟體微調指令，不用機械拆裝，僅由軟體設定，就可以達到微調各個伺服機轉向角度的目的，可設定-128～127 μS。
- 程式可以設定伺服機轉向速度，使用者可根據需求設定多段的伺服機轉向速度。
- 使用者可以設定一個共同時間，讓各個伺服機在同時間達到不同的轉向角度。
- 內建 250 組伺服機記憶空間，每組可以儲存目前設定好的十六個伺服機目標位置，與速度或時間參數，在需要時直接呼叫，可以免去重覆設定的動作，也可以快速組合出多樣化的效果。
- 設有四組事件提醒，讓使用者可以在判斷動作完成後，自動進行下一項操作。各事件可以設定任意 1~16 個伺服機作為判斷依據。
- 各種狀態取得指令，使用者可以隨時確認伺服機是否動作完成，取得現在位置，目標位置，微調參數，或是設定的時間與速度值。
- 解析度可達 2μS。
- 可透過 I2C 方式，下達指令。

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。在模組上有以三個腳位為一組，共十六組伺服機連接座，提供各伺服機的控制訊號與電源，根據伺服機的對應腳位連接就可以動作(如右圖)，另外提供給伺服機的電源請在圖示中的電源輸入位置外加，需確定伺服機所需的電流與電壓，以免伺服機產生不正常動作，造成伺服機損毀。



產品規格:

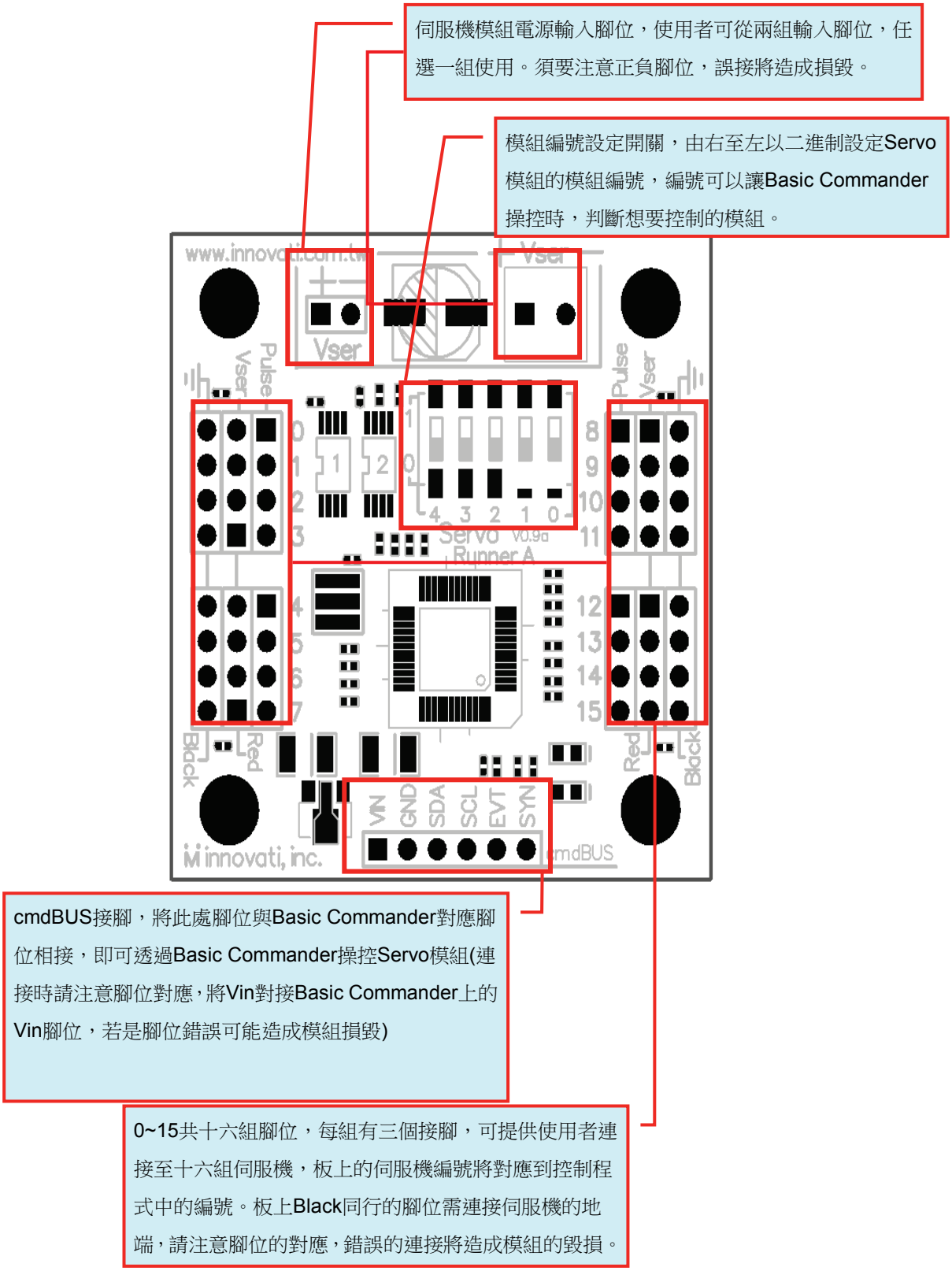


圖 1: 模組腳位與開關介紹

使用電流量: 7 mA (ServoRunnerA 模組未接上伺服機於 cmdBUS 之耗電量)

操作注意事項: 請確認所連接之伺服機所需之電壓範圍，與所需電流大小，選擇合適之

電源，由 Vser 連接正確之電源。
伺服機的 Pulse 腳位與模組連接須符合表 1 規範→(此為模組所能動作之範圍)

Symbol	Parameter	Test Conditions V_{IN}=7.5V	Conditions	Min.	Typ.	Max.	Unit
V_{OH}	I/O Port output high voltage	-	No loading	-	5	-	V
V_{OL}	I/O Port output low voltage	-	No loading	-	0	-	V
I_{OL}	I/O Port Sink Current	-	V_{load}=0.1V_{OH}	10	20	-	mA
I_{OH}	I/O Port Source Current	-	V_{load}=0.9V_{OH}	-5	-10	-	mA

表 1: ServoRunnerA 模組電流限制 (Test Temperature=25℃)

模組操作溫度　　0 ℃~ 70 ℃ (伺服機之操作溫度請於伺服機規格確認)
模組儲存溫度　　-50 ℃~125℃

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 Servo Runner 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 Servo Runner A 指令前，請先於程式開頭定義對應參數與編號，例:
Peripheral *ModuleName* As ServoRunnerA @ 0

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

<table>
<tr><th>指令格式</th><th>指令功能</th></tr>
<tr><td colspan="2">伺服機位置設定相關指令</td></tr>
<tr><td>CmdBUS：
SetPos (SerID, Pos)
I2C：
MID+91+CheckSum1
+SerID+Pos_L+Pos_H
+CheckSum2+Dummy</td><td rowspan="2">以 SerID 指定所要操作的伺服機，請輸入 0~15 之間的整數值，設定該伺服機所要到達的位置為 Pos，若設定 Run 則立刻開始動作 (Pos 請輸入 499~2500 之間的整數值，單位為 μS，若設定超過此範圍的值，這次設定的命令將不被執行)</td></tr>
<tr><td>CmdBUS：
SetPosAndRun(SerID, Pos)
I2C：
MID+88+CheckSum1
+SerID+Pos_L+Pos_H
+CheckSum2+Dummy</td></tr>
</table>

CmdBUS： **SetPosSpd(*SerID*, *Pos*, *Spd*)** **I2C：** **MID+92+CheckSum1** **+SerID+Pos_L+Pos_H** **+Spd_L+Spd_H** **+CheckSum2+Dummy**	以 ***SerID*** 指定所要操作的伺服機，請輸入 0~15 之間的整數值，設定伺服機所要到達的位置為 ***Pos***，***Pos*** 請輸入 499~2500 之間的整數值，並且設定伺服機以 ***Spd*** 移動伺服機，若設定 **Run** 則立刻開始動作(***Spd*** 請輸入 0~65535 之間的整數值，0 代表全速，值越大移動越快，單位為 μS/S)
CmdBUS： **SetPosSpdAndRun(*SerID*, *Pos*, *Spd*)** **I2C：** **MID+89+CheckSum1** **+SerID+Pos_L+Pos_H** **+Spd_L+Spd_H** **+CheckSum2+Dummy**	
CmdBUS： **SetPosTime(*SerID*, *Pos*, *Time*)** **I2C：** **MID+93+CheckSum1** **+SerID+Pos_L+Pos_H** **+Time_L+Time_H** **+CheckSum2+Dummy**	以 ***SerID*** 指定所要操作的伺服機，設定伺服機所要到達的位置為 ***Pos***，***Pos*** 請輸入 499~2500 之間的整數值，並且設定伺服機需要以固定速度，花 ***Time*** 所設定的時間到達指定位置，若設定 ***Time*** 則立刻開始動作(***Time*** 請輸入 0~65535 之間的整數值，***Time*** 若設為 0 則會以全速移動，***Time*** 的單位為毫秒[ms]，如果設定時間太短，伺服機就以全速移動)
CmdBUS： **SetPosTimeAndRun(*SerID*,*Pos*, T*ime*)** **I2C：** **MID+90+CheckSum1** **+SerID+Pos_L+Pos_H** **+Time_L+Time_H** **+CheckSum2+Dummy**	
伺服機啟動相關指令	
CmdBUS： **Run1Servo(*SerID*)** **I2C：** **MID+94+CheckSum1** **+SerID+CheckSum2+Dummy**	根據所設定的 ***SerID***，請輸入 0~15 之間的整數值，啟動各伺服機執行所設定的動作，若起動伺服機時，只進行位置的設定，而未設定速度或時間，伺服機會用最快的速度動作

CmdBUS： **Run2Servo(*SerID*,*SerID1*)** **I2C：** **MID+95+CheckSum1** **+SerID+SerID1** **+CheckSum2+Dummy** **CmdBUS：** **Run3Servo(*SerID*,*SerID1*,*SerID2*)** **I2C：** **MID+96+CheckSum1** **+SerID+SerID1+SerID2** **+CheckSum2+Dummy** **CmdBUS：** **Run4Servo(*SerID*,*SerID1*,*SerID2*,*SerID3*)** **I2C：** **MID+97+CheckSum1** **+SerID+SerID1** **+SerID2+SerID3** **+CheckSum2+Dummy** **CmdBUS：** **Run5Servo(*SerID*,*SerID1*,*SerID2* ***,SerID3*,*SerID4*)** **I2C：** **MID+98+CheckSum1** **+SerID+SerID1+SerID2** **+SerID3+SerID4** **+CheckSum2+Dummy** **CmdBUS：** **Run6Servo(*SerID*,*SerID1*,*SerID2* ***SerID3*,*SerID4*,*SerID5*)** **I2C：** **MID+99+CheckSum1** **+SerID+SerID1+SerID2** **+SerID3+SerID4+SerID5** **+CheckSum2+Dummy**	

CmdBUS： **Run7Servo(*SerID,SerID1,SerID2 SerID3,SerID4,SerID5,SerID6*)** **I2C：** **MID+100+CheckSum1 +SerID+SerID1+SerID2 +SerID3+SerID4+SerID5+SerID6 +CheckSum2+Dummy**	
CmdBUS： **Run8Servo(*SerID,SerID1,SerID2 SerID3,SerID4,SerID5,SerID6,SerID7*)** **I2C：** **MID+101+CheckSum1 +SerID+SerID1+SerID2+SerID3 +SerID4+SerID5+SerID6+SerID7 +CheckSum2+Dummy**	
CmdBUS： **Run9Servo(*SerID,SerID1,SerID2 SerID3,SerID4,SerID5,SerID6 ,SerID7,SerID8*)** **I2C：** **MID+102+CheckSum1 +SerID+SerID1+SerID2+SerID3 +SerID4+SerID5+SerID6+SerID7 +SerID8+CheckSum2+Dummy**	
CmdBUS： **Run10Servo(*SerID,SerID1,SerID2 SerID3,SerID4,SerID5,SerID6 ,SerID7,SerID8,SerID9*)** **I2C：** **MID+103+CheckSum1 +SerID+SerID1+SerID2+SerID3 +SerID4+SerID5+SerID6+SerID7 +SerID8+SerID9+CheckSum2+Dummy**	

CmdBUS： **Run11Servo(*SerID,SerID1,SerID2* *SerID3,SerID4,SerID5,SerID6* *,SerID7,SerID8,SerID9,SerID10*)**	
I2C： **MID+104+CheckSum1** **+SerID+SerID1+SerID2+SerID3** **+SerID4+SerID5+SerID6+SerID7** **+SerID8+SerID9+SerID10** **+CheckSum2+Dummy** **CmdBUS：** **Run12Servo(*SerID,SerID1,SerID2* *SerID3,SerID4,SerID5,SerID6* *,SerID7,SerID8,SerID9,SerID10* *SerID11*)**	
I2C： **MID+105+CheckSum1** **+SerID+SerID1+SerID2+SerID3** **+SerID4+SerID5+SerID6+SerID7** **+SerID8+SerID9+SerID10+SerID11** **+CheckSum2+Dummy** **CmdBUS：** **Run13Servo(*SerID,SerID1,SerID2* *SerID3,SerID4,SerID5,SerID6* *,SerID7,SerID8,SerID9,SerID10* *SerID11,SerID12*)**	
I2C： **MID+106+CheckSum1** **+SerID+SerID1+SerID2+SerID3** **+SerID4+SerID5+SerID6+SerID7** **+SerID8+SerID9+SerID10+SerID11** **+SerID12+CheckSum2+Dummy** **CmdBUS：** **Run14Servo(*SerID,SerID1,SerID2* *SerID3,SerID4,SerID5,SerID6* *,SerID7,SerID8,SerID9,SerID10* *SerID11,SerID12,SerID13*)**	

I2C： **MID+107+CheckSum1** **+SerID+SerID1+SerID2+SerID3** **+SerID4+SerID5+SerID6+SerID7** **+SerID8+SerID9+SerID10+SerID11** **+SerID12+SerID13+CheckSum2+Dummy** **CmdBUS：** **Run15Servo(*SerID,SerID1,SerID2*** ***SerID3,SerID4,SerID5,SerID6*** ***,SerID7,SerID8,SerID9,SerID10*** ***SerID11,SerID12,SerID13,SerID14*)**	
I2C： **MID+108+CheckSum1** **+SerID+SerID1+SerID2+SerID3** **+SerID4+SerID5+SerID6+SerID7** **+SerID8+SerID9+SerID10+SerID11** **+SerID12+SerID13+SerID14** **+CheckSum2+Dummy** **CmdBUS：** **RunAllServo()**	
I2C： **MID+109+CheckSum1+Dummy**	

Run1ServoWithEventA(*SerID*) **…** **Run15ServoWithEventA(*SerID1*, …, *SerID15*)** **RunAllServoWithEventA()**	根據所設定的 ***SerID***，請輸入 0~15 之間的整數值，啟動各伺服機執行所設定的動作，並且在動作完成後，產生事件 A
Run1ServoWithEventB(*SerID*) **…** **Run15ServoWithEventB(*SerID1*, …, *SerID15*)** **RunAllServoWithEventB()**	根據所設定的 ***SerID***，請輸入 0~15 之間的整數值，啟動各伺服機執行所設定的動作，並且在動作完成後，產生事件 B
Run1ServoWithEventC(*SerID*) **…** **Run15ServoWithEventC(*SerID1*, …, *SerID15*)** **RunAllServoWithEventC()**	根據所設定的 ***SerID***，請輸入 0~15 之間的整數值，啟動各伺服機執行所設定的動作，並且在動作完成後，產生事件 C
Run1ServoWithEventD(SerID) **…** **Run15ServoWithEventD(*SerID1*, …, *SerID15*)** **RunAllServoWithEventD()**	根據所設定的 ***SerID***，請輸入 0~15 之間的整數值，啟動各伺服機執行所設定的動作，並且在動作完成後，產生事件 D
伺服機停止相關指令	
CmdBUS： **Pause1Servo(*SerID*)** **I2C：** **MID+197+CheckSum1** **+SerID+CheckSum2+Dummy** **CmdBUS：** **Pause2Servo(*SerID,SerID1*)** **I2C：** **MID+198+CheckSum1** **+SerID+SerID1** **+CheckSum2+Dummy** **CmdBUS：** **Pause3Servo(*SerID,SerID1,SerID2*)** **I2C：** **MID+199+CheckSum1** **+SerID+SerID1+SerID2** **+CheckSum2+Dummy**	根據所設定的 ***SerID***，請輸入 0~15 之間的整數值，暫停各伺服機正在執行的動作

CmdBUS： **Pause4Servo(*SerID,SerID1,SerID2,SerID3*)**	
I2C： **MID+200+CheckSum1** **+SerID+SerID1** **+SerID2+SerID3** **+CheckSum2+Dummy**	
CmdBUS： **Pause5Servo(*SerID,SerID1,SerID2* ***,SerID3,SerID4*)**	
I2C： **MID+201+CheckSum1** **+SerID+SerID1+SerID2** **+SerID3+SerID4** **+CheckSum2+Dummy**	
CmdBUS： **Pause6Servo(*SerID,SerID1,SerID2* ***SerID3,SerID4,SerID5*)**	
I2C： **MID+202+CheckSum1** **+SerID+SerID1+SerID2** **+SerID3+SerID4+SerID5** **+CheckSum2+Dummy**	
CmdBUS： **Pause7Servo(*SerID,SerID1,SerID2* ***SerID3,SerID4,SerID5,SerID6*)**	
I2C： **MID+203+CheckSum1** **+SerID+SerID1+SerID2** **+SerID3+SerID4+SerID5+SerID6** **+CheckSum2+Dummy**	

CmdBUS： **Pause8Servo(*SerID,SerID1,SerID2 SerID3,SerID4,SerID5,SerID6,SerID7*)**	
I2C： **MID+204+CheckSum1 +SerID+SerID1+SerID2+SerID3 +SerID4+SerID5+SerID6+SerID7 +CheckSum2+Dummy**	
CmdBUS： **Pause9Servo(*SerID,SerID1,SerID2 SerID3,SerID4,SerID5,SerID6 ,SerID7,SerID8*)**	
I2C： **MID+205+CheckSum1 +SerID+SerID1+SerID2+SerID3 +SerID4+SerID5+SerID6+SerID7 +SerID8+CheckSum2+Dummy**	
CmdBUS： **Pause10Servo(*SerID,SerID1,SerID2 SerID3,SerID4,SerID5,SerID6 ,SerID7,SerID8,SerID9*)**	
I2C： **MID+206+CheckSum1 +SerID+SerID1+SerID2+SerID3 +SerID4+SerID5+SerID6+SerID7 +SerID8+SerID9+CheckSum2+Dummy**	
CmdBUS： **Pause11Servo(*SerID,SerID1,SerID2 SerID3,SerID4,SerID5,SerID6 ,SerID7,SerID8,SerID9,SerID10*)**	
I2C： **MID+207+CheckSum1 +SerID+SerID1+SerID2+SerID3 +SerID4+SerID5+SerID6+SerID7 +SerID8+SerID9+SerID10**	

+CheckSum2+Dummy **CmdBUS：** **Pause12Servo(*SerID,SerID1,SerID2 SerID3,SerID4,SerID5,SerID6 ,SerID7,SerID8,SerID9,SerID10 SerID11*)**	
I2C： **MID+208+CheckSum1 +SerID+SerID1+SerID2+SerID3 +SerID4+SerID5+SerID6+SerID7 +SerID8+SerID9+SerID10+SerID11 +CheckSum2+Dummy** **CmdBUS：** **Pause13Servo(*SerID,SerID1,SerID2 SerID3,SerID4,SerID5,SerID6 ,SerID7,SerID8,SerID9,SerID10 SerID11,SerID12*)**	
I2C： **MID+209+CheckSum1 +SerID+SerID1+SerID2+SerID3 +SerID4+SerID5+SerID6+SerID7 +SerID8+SerID9+SerID10+SerID11 +SerID12+CheckSum2+Dummy** **CmdBUS：** **Pause14Servo(*SerID,SerID1,SerID2 SerID3,SerID4,SerID5,SerID6 ,SerID7,SerID8,SerID9,SerID10 SerID11,SerID12,SerID13*)**	
I2C： **MID+210+CheckSum1 +SerID+SerID1+SerID2+SerID3 +SerID4+SerID5+SerID6+SerID7 +SerID8+SerID9+SerID10+SerID11 +SerID12+SerID13+CheckSum2+Dummy**	

CmdBUS： **Pause15Servo(*SerID,SerID1,SerID2 SerID3,SerID4,SerID5,SerID6 ,SerID7,SerID8,SerID9,SerID10 SerID11,SerID12,SerID13,SerID14*)** **I2C：** **MID+211+CheckSum1 +SerID+SerID1+SerID2+SerID3 +SerID4+SerID5+SerID6+SerID7 +SerID8+SerID9+SerID10+SerID11 +SerID12+SerID13+SerID14 +CheckSum2+Dummy** **CmdBUS：** **PauseAllServo()** **I2C：** **MID+212+CheckSum1+Dummy**	
CmdBUS： **Stop1Servo(*SerID*)** **I2C：** **MID+213+CheckSum1 +SerID+CheckSum2+Dummy** **CmdBUS：** **Stop2Servo(*SerID,SerID1*)** **I2C：** **MID+214+CheckSum1 +SerID+SerID1 +CheckSum2+Dummy** **CmdBUS：** **Stop3Servo(*SerID,SerID1,SerID2*)** **I2C：** **MID+215+CheckSum1 +SerID+SerID1+SerID2 +CheckSum2+Dummy**	根據所設定的 ***SerID***，請輸入 0~15 之間的整數值，停止各伺服機正在執行的動作，同時停止供應給伺服機電流，此時伺服機若受外力移動，將改變位置

CmdBUS： **Stop4Servo(*SerID,SerID1,SerID2,SerID3*)**	
I2C： **MID+216+CheckSum1** **+SerID+SerID1** **+SerID2+SerID3** **+CheckSum2+Dummy**	
CmdBUS： **Stop5Servo(*SerID,SerID1,SerID2* ***,SerID3,SerID4*)**	
I2C： **MID+217+CheckSum1** **+SerID+SerID1+SerID2** **+SerID3+SerID4** **+CheckSum2+Dummy**	
CmdBUS： **Stop6Servo(*SerID,SerID1,SerID2* ***SerID3,SerID4,SerID5*)**	
I2C： **MID+218+CheckSum1** **+SerID+SerID1+SerID2** **+SerID3+SerID4+SerID5** **+CheckSum2+Dummy**	
CmdBUS： **Stop7Servo(*SerID,SerID1,SerID2* ***SerID3,SerID4,SerID5,SerID6*)**	
I2C： **MID+219+CheckSum1** **+SerID+SerID1+SerID2** **+SerID3+SerID4+SerID5+SerID6** **+CheckSum2+Dummy**	

CmdBUS： **Stop8Servo(*SerID,SerID1,SerID2 SerID3,SerID4,SerID5,SerID6,SerID7*)**	
I2C： **MID+220+CheckSum1 +SerID+SerID1+SerID2+SerID3 +SerID4+SerID5+SerID6+SerID7 +CheckSum2+Dummy**	
CmdBUS： **Stop9Servo(*SerID,SerID1,SerID2 SerID3,SerID4,SerID5,SerID6 ,SerID7,SerID8*)**	
I2C： **MID+221+CheckSum1 +SerID+SerID1+SerID2+SerID3 +SerID4+SerID5+SerID6+SerID7 +SerID8+CheckSum2+Dummy**	
CmdBUS： **Stop10Servo(*SerID,SerID1,SerID2 SerID3,SerID4,SerID5,SerID6 ,SerID7,SerID8,SerID9*)**	
I2C： **MID+222+CheckSum1 +SerID+SerID1+SerID2+SerID3 +SerID4+SerID5+SerID6+SerID7 +SerID8+SerID9+CheckSum2+Dummy**	
CmdBUS： **Stop11Servo(*SerID,SerID1,SerID2 SerID3,SerID4,SerID5,SerID6 ,SerID7,SerID8,SerID9,SerID10*)**	
I2C： **MID+223+CheckSum1 +SerID+SerID1+SerID2+SerID3 +SerID4+SerID5+SerID6+SerID7 +SerID8+SerID9+SerID10 +CheckSum2+Dummy**	

CmdBUS： **Stop12Servo(*SerID,SerID1,SerID2 SerID3,SerID4,SerID5,SerID6 ,SerID7,SerID8,SerID9,SerID10 SerID11*)**	
I2C： **MID+224+CheckSum1 +SerID+SerID1+SerID2+SerID3 +SerID4+SerID5+SerID6+SerID7 +SerID8+SerID9+SerID10+SerID11 +CheckSum2+Dummy**	
CmdBUS： **Stop13Servo(*SerID,SerID1,SerID2 SerID3,SerID4,SerID5,SerID6 ,SerID7,SerID8,SerID9,SerID10 SerID11,SerID12*)**	
I2C： **MID+225+CheckSum1 +SerID+SerID1+SerID2+SerID3 +SerID4+SerID5+SerID6+SerID7 +SerID8+SerID9+SerID10+SerID11 +SerID12+CheckSum2+Dummy**	
CmdBUS： **Stop14Servo(*SerID,SerID1,SerID2 SerID3,SerID4,SerID5,SerID6 ,SerID7,SerID8,SerID9,SerID10 SerID11,SerID12,SerID13*)**	
I2C： **MID+226+CheckSum1 +SerID+SerID1+SerID2+SerID3 +SerID4+SerID5+SerID6+SerID7 +SerID8+SerID9+SerID10+SerID11 +SerID12+SerID13+CheckSum2+Dummy**	

CmdBUS： **Stop15Servo(*SerID,SerID1,SerID2 SerID3,SerID4,SerID5,SerID6 ,SerID7,SerID8,SerID9,SerID10 SerID11,SerID12,SerID13,SerID14*)** **I2C：** **MID+227+CheckSum1 +SerID+SerID1+SerID2+SerID3 +SerID4+SerID5+SerID6+SerID7 +SerID8+SerID9+SerID10+SerID11 +SerID12+SerID13+SerID14 +CheckSum2+Dummy** **CmdBUS：** **StopAllServo()** **I2C：** **MID+228+CheckSum1+Dummy**	
伺服機狀態與記憶相關指令	
Get1ServoReadyStatus(*SerID*, *Status*) **…** **Get15ServoReadyStatus(*SerID1*, …, *SerID15*, *Status*)** **GetAllServoReadyStatus(*Status*)**	取得 ***SerID*** 指定之伺服機狀態，請輸入 0~15 之間的整數值，並將狀態值存放於 ***Status*** 中 (若所指定的伺服機中，有任一個還沒有到達指定位置，所傳回的狀態將會是 0，當所有伺服機皆到達指定位置，傳回狀態則為 1)
GetNowPos (*SerID*, *Pos*)	取得 ***SerID***，請輸入 0~15 之間的整數值，所指定之伺服機現在位置，存放於 ***Pos*** 中，***SerID*** 請輸入 0~15 之間的整數值，***Pos*** 會回傳 499~2500 之間的整數值
GetPos(*SerID*, *Pos*)	取得 ***SerID***，請輸入 0~15 之間的整數值，所指定之伺服機目標位置，存放於 ***Pos*** 中，***SerID*** 請輸入 0~15 之間的整數值，***Pos*** 會回傳 499~2500 之間的整數值
GetPosOffset(*SerID*, *Offset*)	取得 ***SerID***，請輸入 0~15 之間的整數值，所指定之伺服機微調值，存放於 ***Offset*** 中 (***Offset*** 單位為微秒[μS]，範圍可從-128 到 127)
GetSpdAndTime(*SerID*, *Type*, *Value*)	取得 ***SerID***，請輸入 0~15 之間的整數值，所指定伺服機所設定的移動方式 ***Type***，與設定值 ***Value***，若是所設定的移動方式為

	速度，則 ***Type*** 取回值為 1，若為時間則 ***Type*** 為 0，***Value*** 會回傳 0~65535 之間的整數值
CmdBUS： **LoadFrame(*FrameID*)** **I2C：** **MID+196+CheckSum1** **FrameID+CheckSum2+Dummy**	讀取 ***FrameID*** 所指定記憶位址的紀錄值，作為現在伺服機的目標位置與移動方式，***FrameID*** 請輸入 0~249 之間的整數值
CmdBUS： **LoadOffset()** **I2C：** **MID+230+CheckSum1+Dummy**	將 EEPROM 中儲存的 Offset 值，取代現有的設定值
CmdBUS： **SaveFrame(*FrameID*)** **I2C：** **MID+195+CheckSum1** **FrameID+CheckSum2+Dummy**	將目前所設定的各伺服機位置，儲存於 ***FrameID*** 所指定的記憶體中，***FrameID*** 請輸入 0~249 之間的整數值
CmdBUS： **SaveOffset()** **I2C：** **MID+229+CheckSum1+Dummy**	將現在設定的 Offset 值，儲存到 EEPROM 中
CmdBUS： **SetPosOffset(*SerID*, *Offset*)** **I2C：** **MID+193+CheckSum1** **SerID+Offset+CheckSum2+Dummy**	設定 ***SerID***，請輸入 0~15 之間的整數值，所指定的伺服機微調值為 ***Offset***，***Offset*** 請輸入-128~127 之間的整數值

模組提供應用事件:

事件名稱 (Event)	啟動條件
ServoPosReadyEventA	執行 **Run1ServoWithEventA** 類別的指令(**1** 可為 **1~15** 或 **All**)，當所有指定伺服機都到達指定位置，就會啟動此事件
ServoPosReadyEventB	執行 **Run1ServoWithEventB** 類別的指令(**1** 可為 **1~15** 或 **All**)，當所有指定伺服機都到達指定位置，就會啟動此事件
ServoPosReadyEventC	執行 **Run1ServoWithEventC** 類別的指令(**1** 可為 **1~15** 或 **All**)，當所有指定伺服機都到達指定位置，就會啟動此事件
ServoPosReadyEventD	執行 **Run1ServoWithEventD** 類別的指令(**1** 可為 **1~15** 或 **All**)，當所有指定伺服機都到達指定位置，就會啟動此事件

範例程式:

```
'       範例程式中的伺服機位置是以多數伺服機的範圍設定，
'       請根據所使用的伺服機可設定的位置做調整，以免造成伺服機毀損
Peripheral mySer As ServoRunnerA @ 0                '       設定模組編號為 0

Dim EventEnd As Byte                                '       儲存事件處理完成的判斷參數
Dim i As Byte                                       '       儲存迴圈的判斷值
Dim SerStatus As Byte                               ‘       儲存 Servo 的 Status 值


Sub Main()                                          '       主程式
        mySer.SetPosOffset(0, 0)                    '       設定 Servo0 的微調值為 0
        mySer.SetPosAndRun(0, 1500)                 '       啟動 Servo0 移動到 1500 的位置
        Pause 1000                                  '       暫停一段時間讓伺服機移動到指定位置

        mySer.SetPos(0, 2200)                       '       指定 Servo0 目標位置為 2200
        mySer.SaveFrame(0)                          '       將目前設定的伺服機動作存到 Frame0 中
        mySer.Run1Servo(0)                          '       讓 Servo0 開始動作
        Pause 500

        mySer.SetPosSpdAndRun(0, 700, 1000)         '       啟動 Servo0 以速度 1000 移動到 700 的位置
        Pause 2000
        mySer.SetPosTimeAndRun(0, 2200, 1000)       '       啟動 Servo0 花一秒的時間移動到 2200 的位置
        Pause 1000

        EventEnd=0
        mySer.SetPosTime(0, 700, 1000)              '       設定 Servo0 花一秒的時間移動到 700 的位置
        mySer.SaveFrame(1)                          '       將目前設定的伺服機動作存到 Frame1 中
        mySer.Run1ServoWithEventA(0)                '       啟動 Servo0 並於動作結束產生 EventA

        Do
                Pause 1
        Loop Until EventEnd=1

'       下面的迴圈反覆讀取 Frame0 與 Frame1 的設定值，再啟動 Servo0 執行
'       Frame0 儲存的位置為 2200，Frame1 儲存的位置為 700
'       Servo0 將在這兩個位置間來回移動四次
        For i=0 To 3
                mySer.LoadFrame(1)                  '       讀取儲存於 Frame1 的設定值
                mySer.Run1Servo(0)
                Pause 1000
```

```
            mySer.LoadFrame(0)                          '       讀取儲存於 Frame0 的設定值
            mySer.Run1Servo(0)
            Pause 1000
        Next

        mySer. SetPosAndRun(0, 1500)
'       下面的迴圈反覆執行讀取 Status 的動作，
'       在確定動作結束後，才會跳出迴圈
        Do
            mySer.Get1ServoReadyStatus(0, SerStatus)‘       讀取 Servo0 的狀態存於 SerStatus 中
        Loop Until SerStatus>0

End Sub

Event mySer.ServoPosReadyEventA()
        mySer.SetPosAndRun(0, 2200)
        Pause 1000
        EventEnd=1
End Event
```

附錄

1. 已知問題:
   - V1.0 版本用 GetPos 取得目標位置，須在設定位置後，相隔 1 毫秒以上，否則可能取得錯誤位置。
   - V1.0 版本使用 SaveFrame 儲存位置時，所設定的位置需為偶數，若為奇數則取回值將出現錯誤。
   - V1.0 版本連續執行 SaveFrame 作儲存動作時，各儲存動作間須有 5 毫秒以上之間隔，否則可能無法正確完成所有的儲存命令。
   - V1.1 之前版本僅能儲存 120 組動作。
   - V1.1 之前版本沒有提供 SaveOffset 與 LoadOffset 指令。
   - V1.1 之前版本不會將 Offset 值存在 EEPROM 中。
   - V1.1 之前版本在讀取儲存的動作值時，會有讀取到錯誤值的現象。

   ※版本標示於模組上的雷射貼紙

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