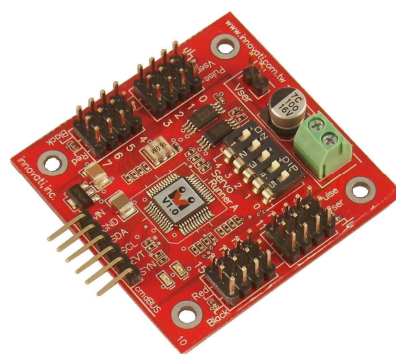


Servo Runner A 模組

十六組伺服機輸出控制

版本: V1.0



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，決定伺服機的移動效率。設有多達 120 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。

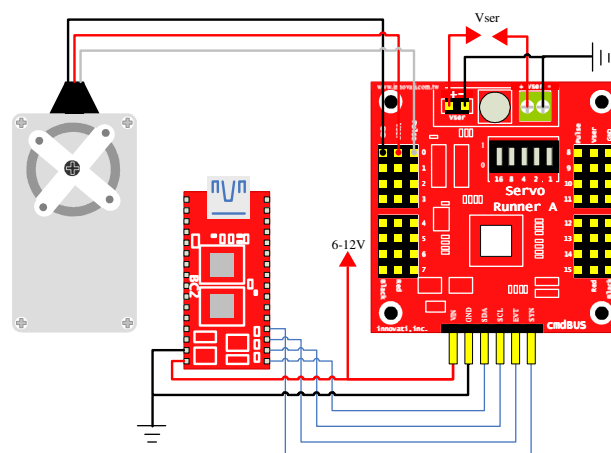
應用方向:

- 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 μ S。
- 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 120 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 各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 2 μ S。

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:

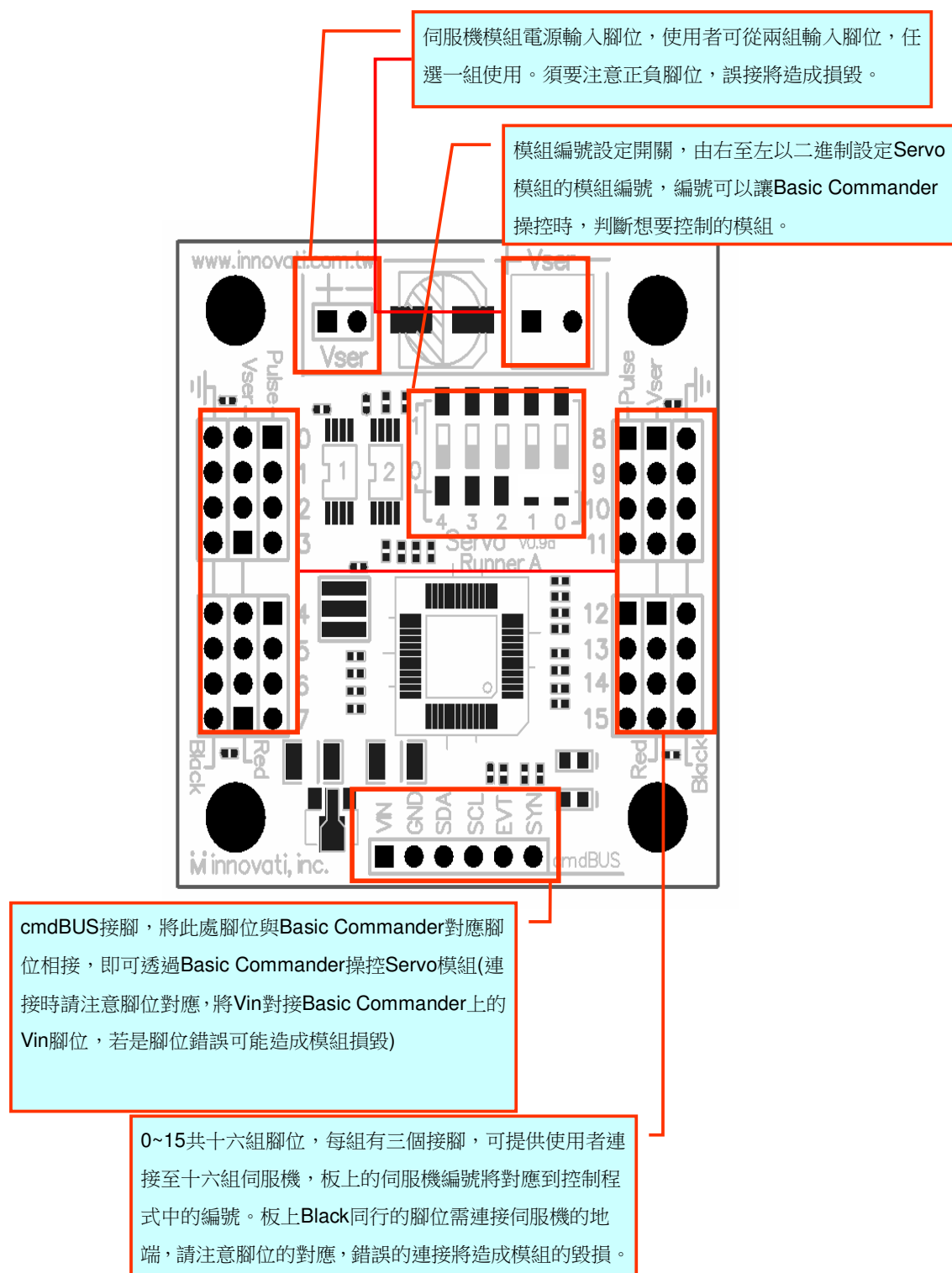


圖 1: 模組腳位與開關介紹

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		Min.	Typ.	Max.	Unit
		V _{IN} =7.5V	Conditions				
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℃

指令表: (指令格式中，各參數以”，”分隔開，並以”參數型態 參數”方式表列)

指令格式	指令功能
伺服機位置設定相關指令	
SetPos (Byte SerID, Word Pos)	以 SerID 指定所要操作的伺服機，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 可設定範圍為 499~2500，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)
SetPosAndRun(Byte SerID, Word Pos)	
SetPosSpd(Byte SerID, Word Pos, Word Spd)	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，並且設定伺服機以 Spd 值移動伺服機，若設定 Run 則立刻開始動作(Spd 值為 0 代表全速，值越大移動越快，可輸入值最大為 2000，單位為 μS/S)
SetPosSpdAndRun(Byte SerID, Word Pos, Word Spd)	
SetPosTime(Byte SerID, Word Pos, Word Time)	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，並且設定伺服機需以固定速度，花 Time 所設定的時間到達指定位置，若設定 Run 則立刻開始動作(Time 可設定範圍為 0~65535，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)
SetPosTimeAndRun(Byte SerID, Word Pos, Word Time)	
伺服機啟動相關指令	
Run1Servo(Byte SerID)	根據所設定的 SerID，啟動各伺服機執行所設定的動作，若啟動伺服機時，只進行位置的設定，而未設定速度或時間，伺服機將以最快速度動作 (SerID 之範圍為 0~15，若設定超出範圍的值，則該設定無效)
...	
Run15Servo(Byte SerID1, ..., Byte SerID15)	
RunAllServo()	

Run1ServoWithEventA(Byte SerID)	根據所設定的 SerID，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
...	
Run15ServoWithEventA(Byte SerID1, ..., Byte SerID15)	
RunAllServoWithEventA()	
Run1ServoWithEventB(Byte SerID)	根據所設定的 SerID，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
...	
Run15ServoWithEventB(Byte SerID1, ..., Byte SerID15)	
RunAllServoWithEventB()	
Run1ServoWithEventC(Byte SerID)	根據所設定的 SerID，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
...	
Run15ServoWithEventC(Byte SerID1, ..., Byte SerID15)	
RunAllServoWithEventC()	
Run1ServoWithEventD(Byte SerID)	根據所設定的 SerID，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
...	
Run15ServoWithEventD(Byte SerID1, ..., Byte SerID15)	
RunAllServoWithEventD()	
伺服機停止相關指令	
Pause1Servo(Byte SerID)	根據所設定的 SerID，暫停各伺服機正在執行的動作
...	
Pause15Servo(Byte SerID1, ..., Byte SerID15)	
PauseAllServo()	
Stop1Servo (Byte SerID)	根據所設定的 SerID，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
...	
Stop15Servo (Byte SerID1, ..., Byte SerID15)	
StopAllServo ()	
伺服機狀態與記憶相關指令	
Get1ServoReadyStatus(Byte SerID, Byte Status)	取得 SerID 指定之伺服機狀態，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)
...	
Get15ServoReadyStatus(Byte	

SerID1, ..., Byte SerID15, Byte Status)	
GetAllServoReadyStatus(Byte Status)	
GetNowPos (Byte SerID, Word Pos)	取得 SerID 所指定之伺服機現在位置，存放於 Pos 中
GetPos(Byte SerID, Word Pos)	取得 SerID 所指定之伺服機目標位置，存放於 Pos 中 (*1)
GetPosOffset(Byte SerID, Short Offset)	取得 SerID 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(Byte SerID, Byte Type, Word Value)	取得 SerID 所指定伺服機所設定的移動方式存於 Type， 所設定的參數值存於 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)
LoadFrame(Byte FrameID)	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 可儲存範圍為 0~119)
SaveFrame(Byte FrameID)	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 (*2, *3)
SetPosOffset(Byte SerID, Short Offset)	設定 SerID 所指定的伺服機微調值為 Offset 所設定的值

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































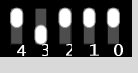

附錄

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。

※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