CAN-8123/ CAN-8223/CAN-8423
CANopen 僕端(Slave)設備

使用手冊

保固條款

所有由泓格科技製造的產品，泓格科技皆提供對產品本身的一年保固，保固期由本公司交貨給原始訂購者的當天開始起算。

注意事項

泓格科技不對因使用本產品所引起的損害作任何的擔保，並保留在未公告的前提下，對本文件隨時進行修訂的權利。由泓格科技提供的這份文件被認定是正確且可信賴的，然而，泓格科技並不對這份文件的使用作任何的擔保，也不對因爲使用這份文件所引起的違反專利或對第三方的侵權負任何責任。

版權

商標

在本書中所引用的商標及產品名稱均屬原公司所有，書中引用純屬介紹之用，並無侵害之意。

目錄

1 介紹

1.1 概述

CANopen，是一種基於智能領域匯流排(intelligent field bus，如CAN bus)之上的通訊協定。其被用來發展具備高度彈性組態能力的標準嵌入式網路。

CANopen爲了因應不同需求，提供了多種標準化的通訊物件，像是適合用來傳輸即時(real-time)資料的Process Data Objects(PDO)、適合用來傳輸組態資料的Service Data Objects(SDO)、可進行網路管理的Network Management Objects(如NMT訊息與錯誤控制)，以及其他具有特殊功能的通訊物件(如Time Stamp、SYNC與EMCY訊息)等等。

現今，CANopen被使用在各種不同的應用領域，像是醫療設備、工程車輛、航海電子、公眾傳輸與建築自動化等等。

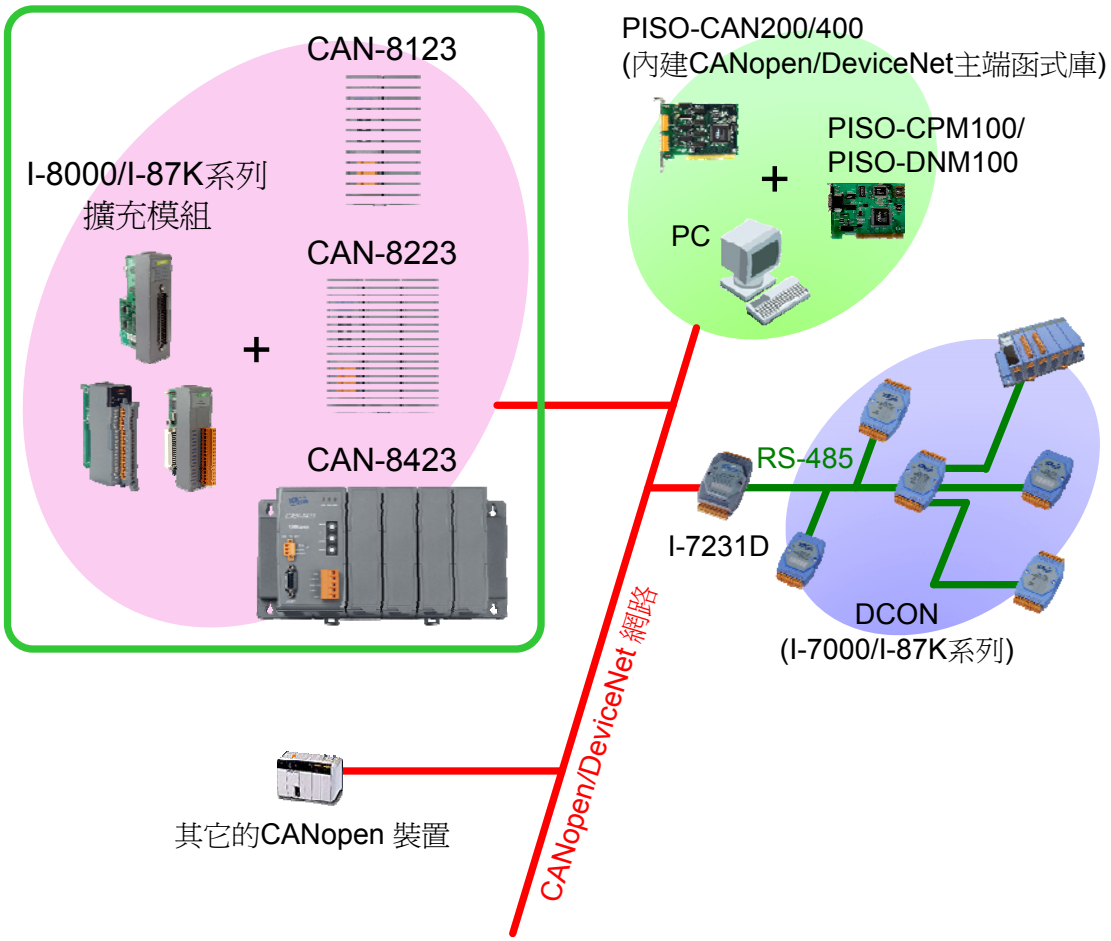
泓格科技針對 CANopen 協定中的僕端設備，推出了 CAN-8123/CAN-8223/CAN-8423 主控單元(main control unit)。

CAN-8123 具有 1 個擴充插槽(expansion slot)，CAN-8223 具有 2 個擴充插槽，CAN-8423 具有 4 個擴充插槽，使用者可以根據其實際的需求，選用 I-87K 或著 I-8000 系列的 DI/DO/AI/AO 擴充模組，安裝在擴充插槽上，擴充裝置的 I/O 通道，使裝置在應用上更具有彈性。

CAN-8123/CAN-8223/CAN-8423 的設計均遵循了 CANopen 的規範，如 DS-301 V4.01 和 DS-401 V2.1。因此裝置提供了像是動態 PDO、EMCY 物件、錯誤輸出值、循環(cyclic)與非循環(acyclic)同步等等不同的功能。

另外，我們也提供了 CAN 僕端工具程式(CAN Slave Utility)，讓使用者能夠根據裝置實際的配置，動態地建立 EDS 檔案。EDS 檔案的規範是基於 CANopen 的標準 DS-306 之上，並且能夠相容不同製造商所製造的 CANopen 主端介面(前提是這些 CANopen 主端支援 EDS 檔案)。

於本手冊中接下來的部份將詳細介紹 CAN-8123/CAN-8223/CAN-8423 的架構及常見應用。


CAN-8123
I-8000/I-87K系列
擴充模組
CAN-8223
CAN-8423
PISO-CAN200/400
(內建CANopen/DeviceNet主端函式庫)
PISO-CPM100/
PISO-DNM100
PC
RS-485
I-7231D
DCON
(I-7000/I-87K系列)
CANopen/DeviceNet 網路
其它的CANopen 裝置

1.2 硬體特徵

- 中央處理器(CPU)：80186, 80MHz
- Philips SJA1000 CAN 控制器(CAN controller)
- Philips 82C250 CAN 收發器(CAN transceiver)
- 靜態隨機存取記憶體(SRAM)：512K bytes
- 快閃記憶體(Flash Memory)：512K bytes
- 電子式可抹除可編程唯讀記憶體(EEPROM)：2k bytes
- 非揮發性隨機存取記憶體(NVRAM)：32 bytes
- 實時時鐘(Real Time Clock)
- 內建看門狗計時器(Watchdog Timer)
- 16-bit 計時器
- 電源指示燈(Power LED)、運行指示燈(RUN LED)、
  錯誤指示燈(ERR LED)
- 支援 1/2/4 個擴充 I/O 插槽
- CAN 端具有 2500 Vrms 的隔離
- 可利用跳線器決定啓用與否的 120Ω終端電阻
- CAN bus 之介面：ISO/IS 11898-2，內建具有光學隔離保護功能(optical isolators protection)的 5-pin 螺旋式終端(5-pin screw terminal)。
- 電源輸入：+10V_{DC} 到+30V_{DC}
- 功率消耗：不含擴充模組約 20W
- 操作溫度：-25˚C 到+75˚C
- 儲存溫度：-30˚C 到+85˚C
- 溼度：5%~95%

COM1

- RS-232：TXD,RXD,RTS,CTS,GND
- 連線速度：115200 bps
- 可透過配置工具(Configure tool)與裝置進行連線

1.3 CAN-8123/CAN-8223/CAN-8423 特性

- NMT 模式：支援作爲 NMT 僕端(Slave)
- 錯誤控制協定：支援節點守衛協定(Node Guarding Protocol)
- 節點 ID：可透過旋鈕來設定節點 ID
- PDO 數目：RxPDO 和 TxPDO 各支援 16 組
- PDO 模式：事件觸發、遠端要求、循環與非循環 SYNC
- PDO 映射：支援可調整的映射方式
- SDO 模式：支援作爲 SDO 伺服端
- SDO 之數目：RxSDO 和 TxSDO 各支援一組
- 緊急訊息：支援緊急訊息(EMCY message)的發送
- CANopen 版本：DS-301 v4.02
- 裝置描述文件：DS-401 v2.1
- 支援鮑率：10K、20K、50K、125K、250K、500K、800K 以及 1M bps。
- CAN-8423 最多可支援 4 組 I-8000 和 I-87K 系列的擴充模組，CAN-8223 最多支援 2 組，CAN-8123 最多支援 1 組。
- 提供便於使用的工具程式，可對 CAN-8123/CAN-8223/CAN-8423 上的 I-8000 和 I-87K 系列擴充模組進行設定。

1.4 工具程式特性

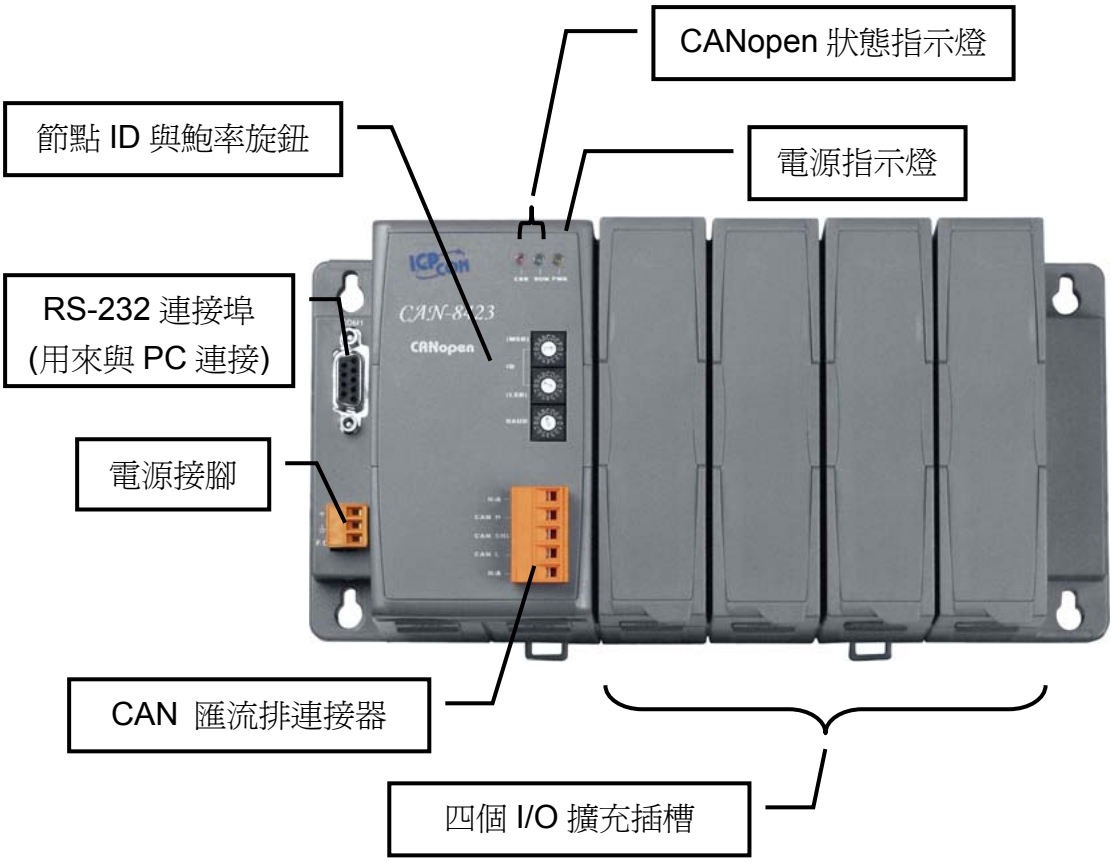
- 可設定 I-8000 或 I-87K 的 AI/AO 模組之參數
- 顯示 I-8000 和 I-87K 模組的組態
- 顯示應用程式(Application)與裝置物件(Device Object)的資訊
- 可顯示 RxPDO 映射、TxPDO 映射
- 動態建立 EDS 檔案

2 硬體規格

2.1 CAN-8123/CAN-8223 硬體結構

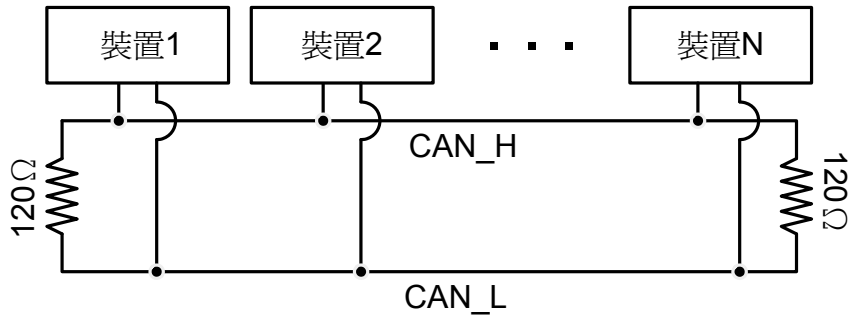
兩個 I/O 擴充插槽

2.2 CAN-8423 硬體結構



2.3 連線

爲了讓信號反射(reflection)在 CAN bus 上的影響降到最低，CAN bus 在網路的兩端必須要分別安裝一個終端電阻，如下圖所示：

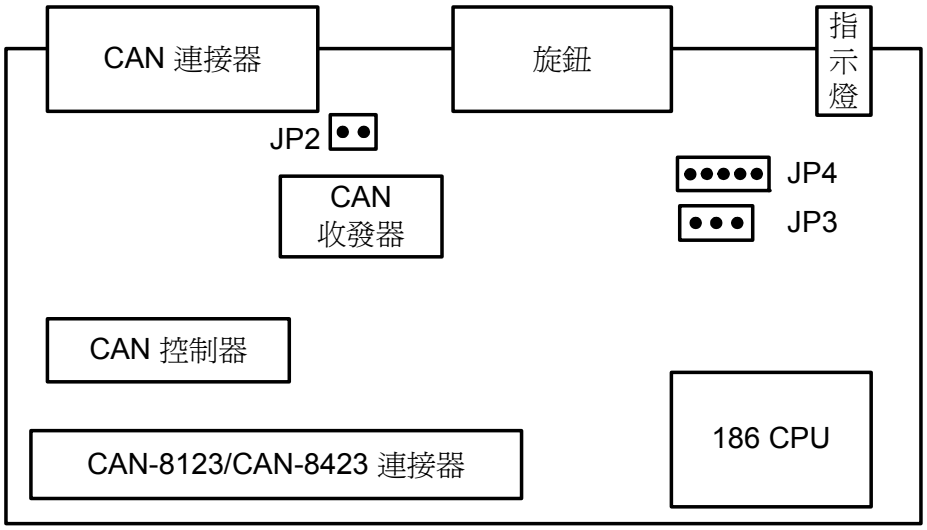


根據 ISO 11898-2 的規範，在上圖之中，每一個終端電阻應爲 120Ω(或著介於 108Ω~132Ω之間)。接線本身的電阻值應小於 70Ω/km。在開始架設 CAN bus 前，使用者應先檢查匯流排上的電阻值是否正常。

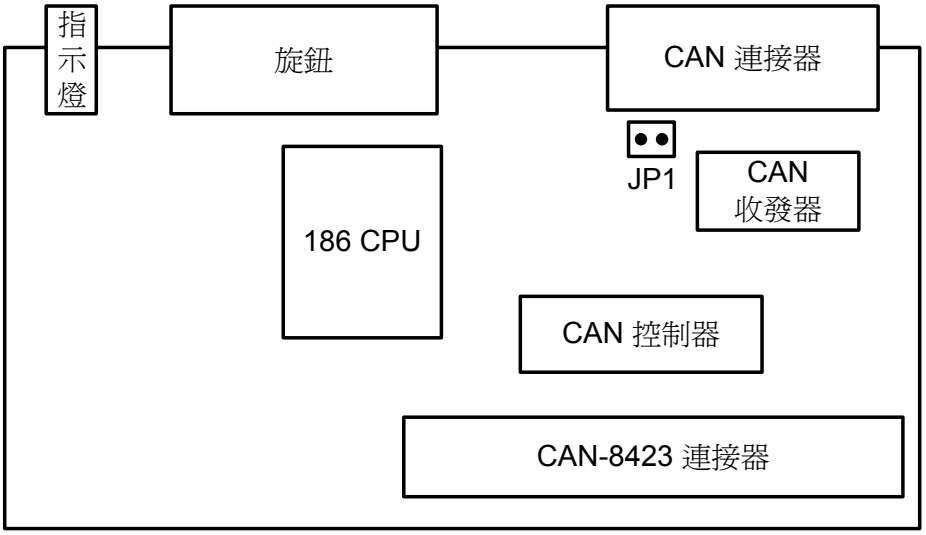
此外，爲了將長距離傳輸所引起的電壓損耗降到最低，實際使用的終端電阻值應比 ISO 11898-2 所規範的再高一些。下表的數據可作爲在架設 CAN bus 時的參考。

匯流排長度 (m)	匯流排纜線參數		終端電阻 (Ω)
	長度與電阻之關係 (Ω/km)	橫截面 (mm^2)	
0~40	70	0.25 (23AWG)~ 0.34 (22AWG)	124 (0.1%)
40~300	< 60	0.34 (22AWG)~ 0.6 (20AWG)	127 (0.1%)
300~600	< 40	0.5~0.6 (20AWG)	150~300
600~1K	< 20	0.75~0.8 (18AWG)	150~300

在 CAN-8123/CAN-8223/CAN-8423 內，本身便具有 120Ω的終端電阻可供使用。其中，CAN-8123/CAN-8223 的跳線器 JP2 便是用來調整終端電阻用的。JP2 的位置如下圖所示。



而 CAN-8423 則是利用跳線器 JP1 來調整終端電阻，它的位置如下圖所示。



決定終端電阻啓用和不啓用的跳線器連接方式如下所示。

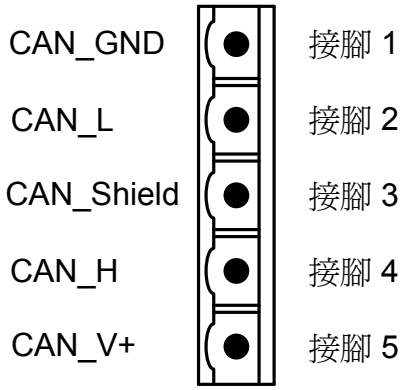
不啓用　　啓用

CAN bus 上所能運行的鮑率受到匯流排長度的限制。下表指出不同匯流排長，所能運行的鮑率。

鮑率(bit/s)	最大匯流排長度(m)
1 M	25
800 K	50
500 K	100
250 K	250
125 K	500
50 K	1000
20 K	2500
10 K	5000

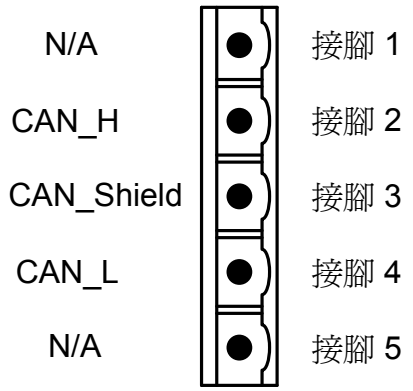
註：當匯流排的長度超過 1 公里時，就必需要在匯流排上，另外加裝橋接器或增益器。

CAN-8123/CAN-8223 上的匯流排連接器，各接腳所代表的意義，如下所示：



接腳編號	訊號	描述
1	CAN_GND	接地(0V)
2	CAN_L	CAN_L 匯流排線(顯性 low，dominant low)
3	CAN_SHLD	可選用的 CAN 遮蔽(Shield)
4	CAN_H	CAN_H 匯流排線(顯性 high，dominant high)
5	CAN_V+	CAN-8123/CAN-8223 電源輸入

CAN-8423 上的匯流排連接器，其接腳所代表的意義，如下所示：



接腳編號	訊號	描述
1	N/A	N/A
2	CAN_H	CAN_H 匯流排線(顯性 high，dominant high)
3	CAN_SHLD	選用的 CAN 遮蔽
4	CAN_L	CAN_L 匯流排線(顯性 low，dominant low)
5	N/A	N/A

2.4 電源指示燈

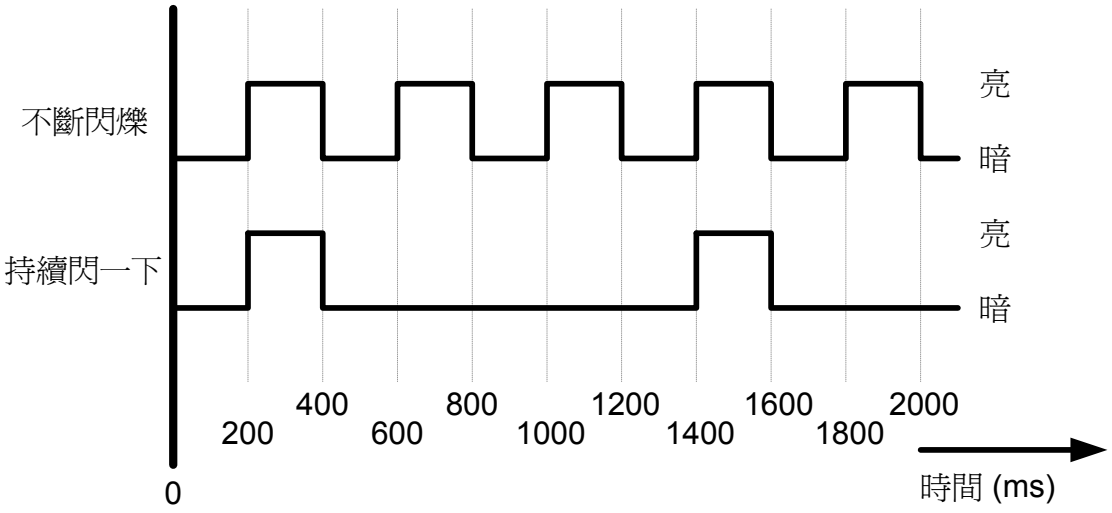
電源指示燈的顏色爲黃色。每一部CAN-8123/CAN-8223/CAN-8423的裝置均需要10~30伏特的直流電壓作爲電源輸入。CAN-8123/CAN-8223/CAN-8423的功率消耗與其擴充插槽上安裝的模組數量有關。若供應的電力足夠，則電源指示燈將會亮起。若供電後，電源指示燈無法亮起，使用者於此時可先檢查電源供應器是否正常作動，供電電壓是否正常。

2.5 CANopen 狀態指示燈

CAN-8123/CAN-8223/CAN-8423 提供兩個 CANopen 的指示燈，分別是錯誤指示燈(紅色)和運行指示燈(綠色)。錯誤指示燈和運行指示燈是在 CANopen 的規範內所定義的。這些指示燈會以不同的閃爍方式來對應不同的 CANopen 通訊事件。於下將解釋不同的指示燈閃爍方式所代表的意義。

2.5.1 運行指示燈

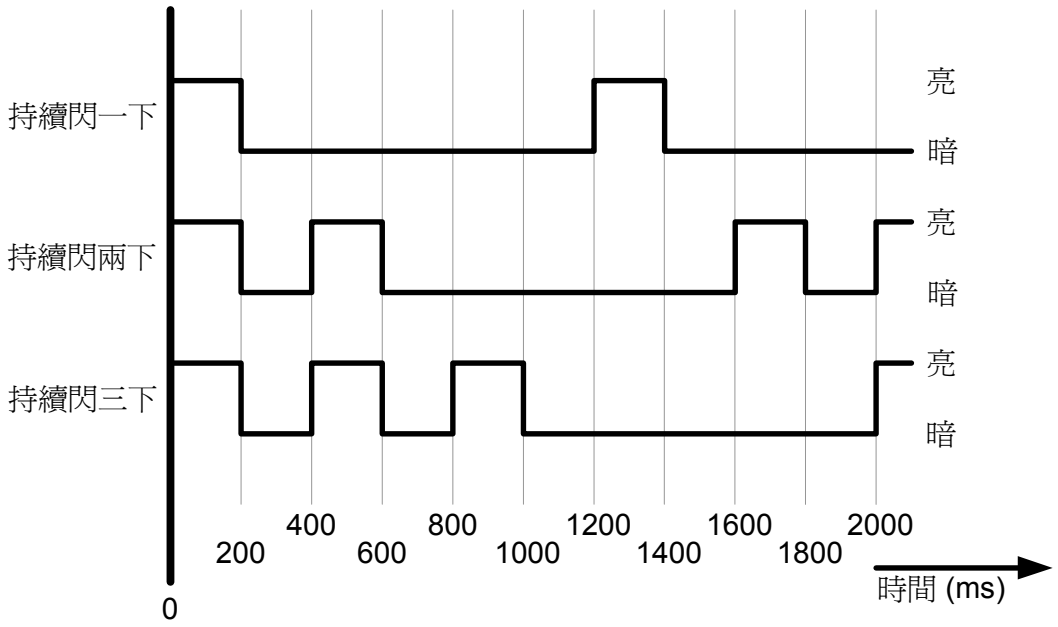
我們可以透過運行指示燈來了解裝置目前的網路狀態機制(network state mechanism)爲何。關於 CANopen 網路狀態機制的資訊，請參考 3.5.1 節。不同的指示燈閃爍方式和其對應的意義，將分別呈現於下圖與下表：



編號	CAN 運行指示燈	狀態	描述
1	持續閃一下	停止(stop)	裝置目前處於停止狀態
2	不斷閃爍	預操作(pre-operational)	裝置目前處於預操作狀態
3	恆亮	操作(operational)	裝置目前處於操作狀態

2.5.2 錯誤指示燈

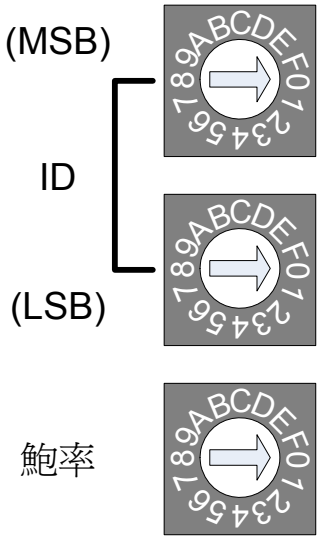
以下幾種狀況可能改變錯誤指示燈的燈號，CAN 物理層(physical layer)發生錯誤，裝置沒有收到某些特定的訊息(如 SYNC 訊息或 NMT 守衛訊息等)所引起的錯誤。每一種錯誤事件均有其不同的閃爍頻率，不同的指示燈閃爍方式和其對應的意義，將分別呈現於下圖與下表：



編號	錯誤指示燈	狀態	描述
1	不亮	沒有錯誤	裝置目前處於工作狀態
2	持續閃一下	已到達警告限制	至少有一個 CAN 控制器的錯誤計數器，已經到達或超過警告標準。(錯誤幀的數目太多)
3	持續閃兩下	錯誤控制事件	發生了守衛(guard)事件(NMT 僕端或 NMT 主端)或著心跳(heartbeat)事件(Heartbeat 消費者)。
4	持續閃三下	SYNC 錯誤	在預先設定的通訊週期內，沒有收到 SYNC 的訊息。(可參照物件字典中主索引爲 0x1006 的項目)
5	恆亮	匯流排關閉 (bus off)	CAN 控制器已經到達了匯流排關閉的條件。

註： 若同一時間內有多個錯誤出現，編號高者將會優先被顯示。舉例來說，若 NMT 錯誤(編號 3)和 SYNC 錯誤(編號 4)同時發生，則 SYNC 將會優先被顯示。

2.6 節點 ID 旋鈕與鮑率旋鈕



上面兩顆旋鈕是用來調整 CAN-8123/CAN-8223/CAN-8423 的節點 ID。其中 MSB 代表節點 ID 高半字節(high nibble)的部份，而 LSB 則是代表節點 ID 低半字節(low nibble)的部份。

舉例來說，假設 MSB 旋鈕被調到 3 的位置，而 LSB 的旋鈕被調到 2 的位置，那麼節點 ID 就會被設定爲 0x32，換算成十進制就是 50 (3x16+2=50)。

根據 CANopen 的規範，節點 ID 設定超過 0x7F(十進制的 127)是沒有意義的。若節點 ID 被設定超過 127，裝置的韌體就會自動將節點 ID 設定爲 1。

最下面的旋鈕(BAUD)是用來調整 CAN-8123/CAN-8223/CAN-8423 的鮑率。鮑率旋鈕所指的數值與實際鮑率的對應關係請參照下表：

旋鈕的數值	鮑率(K BPS)
0	10
1	20
2	50
3	125
4	250
5	500
6	800
7	1000

此外，若 CAN-8423 鮑率的旋鈕被調到 9，則 CAN-8423 就會進入起始狀態，CAN-8423 內的韌體中有關 CANopen 的部份就不會被執行。這時使用者可以執行工具程式，透過 RS-232 的連線埠與 CAN-8423 進行連線，以設定 CAN-8423 的參數或是建立 EDS 檔案。

由於 CAN-8123/CAN-8223 沒有 RS-232 的連線埠，如果使用者想要獲得 CAN-8123/CAN-8223 的 EDS 檔案，就只能透過離線的方式，直接執行工具程式以建立 EDS 檔案。

有關工具程式的使用方式以及如何建議 EDS 檔案，請參照第 4 章。

當 CAN-8123/CAN-8223/CAN-8423 開機時，CANopen 的韌體會先去檢查面板上面的旋鈕所指的數值。任何不正確的旋鈕設定方式，均會使得 CAN-8123/CAN-8223/ CAN-8423 的開機失敗。

2.7 支援模組

CAN-8123/CAN-8223/CAN-8423 支援了 I-8000/I-87K 系列中，許多種類的 DI/DO/AI/AO 擴充模組。

當使用者希望在 CANopen 的網路上使用這些擴充模組時，他們只需要把這些擴充模組裝到 CAN-8123/CAN-8223/CAN-8423 的 I/O 擴充插槽上即可。然後，CAN-8123/CAN-8223/CAN-8423 內建的 CANopen 韌體在會自動搜尋這些擴充模組，並在物件字典內自動組織這些擴充模組對應的物件。

下表爲 CAN-8123/CAN-8223/CAN-8423 所支援的擴充模組：

IO 型態	模組名稱	IO 型態	模組名稱
AI	I-87013/ I-87017/ I-87018 I-8017H	AO	I-8024 I-87022/ I-87024/ I-87026
DO	I-8037/ I-8041/ I-8056/ I-8057/ I-8060/ I-8064/ I-8065/ I-8066/ I-8068/ I-8069 I-87041/ I-87056/ I-87057/ I-87060/ I-87064/ I-87065/ I-87066/ I-87068/ I-87069	DI	I-8040/ I-8051/ I-8052/ I-8053/ I-8058/ I-87040/ I-87051/ I-87052/ I-87053/ I-87058/
DO&DI	I-8042/ I-8054/ I-8055/ I-8063 I-87042/ I-87054/ I-87055 I-87063/		

3 CANopen 系統

3.1 CANopen 介紹

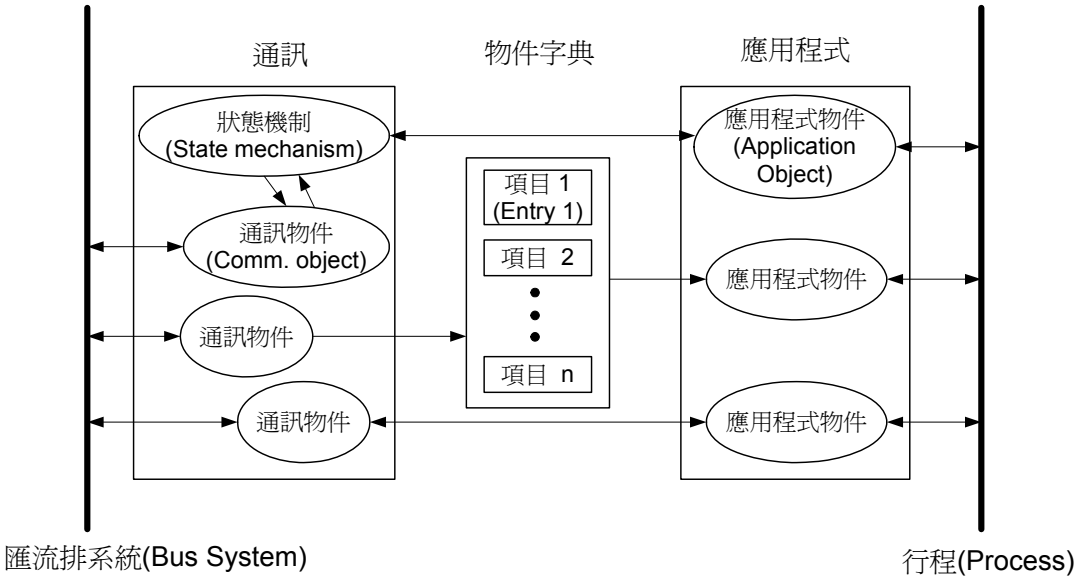
CANopen 是一種基於 CAN bus 之上的網路協定，且已經被使用在各種不同的應用中，像是交通工具，工業機械，自動化建築，醫療裝置，航海應用，飯店器具，實驗室設備與研究。

CANopen 內不只提供了訊息的廣播，同時也支援了節點間的點對點資料交換。CANopen 內所規範的網路管理功能，可簡化了專案的設計。此外，使用者還可以透過 CANopen 規範內的網路啓動(network start-up) 標準機制和錯誤管理(error management)標準機制，來對 CANopen 的網路進行實作與偵錯。

根據 CANopen 的裝置模型(device model)，任何 CANopen 的裝置均可有效率的存取 I/O 的數值，或著查詢同一網路內其他裝置的節點狀態。一般來說，一個 CANopen 的裝置可以大略區分爲三個部份。

- 通訊(Communication)
- 物件字典(Object Dictionary)
- 應用程式(Application)

下圖爲 CANopen 的裝置模型，其內每一部分的功能和概念，將於下描述。



通訊

裝置模型中的通訊部份內含數個不同功能的通訊物件(COB，Communication Object)，裝置和裝置之間就是使用這些 COB 來傳遞 CANopen 的訊息。這些 COB 分別爲 PDO(過程資料物件)、SDO(服務資料物件)、NMT(網路管理物件)以及 SYNC(同步物件)等。每一通訊物件均有其不同的用途，在使用時也必須依循其通訊模型。

以 PDO、SDO 和 NMT 爲例。SDO 可用來存取裝置物件字典內的各個項目，其通訊模型爲用戶端/伺服端(client/server)的架構(詳見 3.2 節)。相較於 SDO，PDO 通訊物件的傳輸協定沒有 header，也就是沒有額外的負擔(overhead)，速度較快，適合用來傳送與接收即時性的資料或 I/O 數值。PDO 的通訊模型乃是依循了生產者與消費者(producer/consumer)的架構，這種架構同時也被稱爲推挽式(Push/Pull)的模型(詳見 3.3 節)。NMT 的通訊物件則被用監控 CANopen 網路中，各節點的狀態，其遵循了主從式(master/slave)的架構(詳見 3.5 節)。

在 CAN-8123/CAN-8223/CAN-8423 上，不管使用了哪一種的 COB 來進行訊息的傳輸，其格式都必須符合 CAN 2.0 A 所定義的資料幀(data frame)規範，一般來說，資料幀的格式會如同下圖：

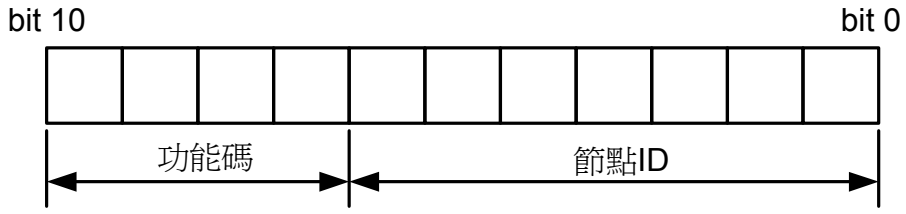
ID	RTR	資料長度	8-byte 資料

ID 欄位共有 11 個 bit，作爲匯流排上訊息優先權的仲裁機制之用。

RTR 欄位則僅有一個 bit，若某個訊息的 RTR 被設定爲 1，則這個訊息就是用作遠端傳送要求(RTR，Remote Transmit Requests)之用，此時，8 bytes 的資料欄位將不會被使用與傳送。

資料長度欄位包含了 4 個 bit 的資料，其代表著在 8 bytes 的資料欄位內，有效的資料數。最後一個欄位，也就是資料欄位，則用來存放訊息資料。值得特別注意的是，根據 CANopen 的規範，裝置在傳輸資料時，資料欄位內每一筆獨立的數值，是以低字節先傳的方式來傳送的(Small Endian)。

各 COB 在透過 data frame 進行傳送時，所使用的 Frame ID 又被稱爲 COB-ID。根據 CANopen 的規範，所有 COB 預設的 COB-ID 均是由 4 bit 的功能碼(Function Code)和 7 bit 的節點 ID(Node ID)組合而成的，其格式如下所示：



註： COB 預設的 COB-ID，是指在裝置在預設的情況之下(剛出廠，或著因爲電源重新啓動等原因，導致設定被清除掉)，裝置上各 COB 預設的 COB-ID。

COB-ID 被定義用來識別訊息的來源與目的地，也被用來區別訊息的用途，另外也決定了網路中了每一點，在傳送訊息時的優先等級。根據 CAN bus 的仲裁機制，COB-ID 數值較低的 CAN 訊息，有較高的優先等級可被傳送進入 CAN bus。

另外，根據 CANopen 的規範，有部分的 COB-ID 被保留給特定的通訊物件，或著留作更進一步的用途，使用者不能夠隨意的使用這些 COB-ID。於下表列出被保留的 COB-ID：

被保留的 COB-ID (hex)	被使用的物件
0	網路管理(NMT)
1	保留
80	同步(SYNC)
81~FF	緊急事件(EMERGENCY)
100	時戳(TIME STAMP)
101~180	保留
581~5FF	預設用來傳輸 SDO
601~67F	預設用來接收 SDO
6E0	保留
701~77F	網路管理(NMT)錯誤控制
780~7FF	保留

除了之前提到被保留的 COB-ID，使用者可以依據實際需求，使用剩下來的 COB-ID。

此處將 CANopen 協定中所有 COB 預設的 COB-ID 列於下表：

(Bit10~Bit7) (功能碼)	(Bit6~Bit0)	通訊物件名稱
0000	0000000	網路管理(NMT)
0001	0000000	同步(SYNC)
0010	0000000	時戳(TIME STAMP)
0001	節點 ID	緊急事件(EMERGENCY)
0011/0101/0111/1001	節點 ID	TxPDO1/2/3/4
0100/0110/1000/1010	節點 ID	RxPDO1/2/3/4
1011	節點 ID	用來傳輸的 SDO (TxSDO)
1100	節點 ID	用來接收的 SDO (RxSDO)
1110	節點 ID	網路管理(NMT)錯誤控制

註： CAN-8123/CAN-8223/CAN-8423 支援時戳以外的所有通訊物件。

物件字典

物件字典內蒐集了許多重要的資訊。這些資訊對裝置的行為有重要的影響，像是 I/O 通道中的資料、通訊的參數與網路的狀態。物件字典實質上是一群物件，而一個物件內可能有一到多個項目，另外這些項目能夠透過一事先定義(pre-defined)的方法經由網路被存取。

物件字典內的每一個項目均有其各自的作用(如通訊參數、裝置描述文件(device profile)等)、資料型別(如 8-bit 的整數、8-bit 的無號整數等)和存取類型(唯讀、唯寫等)。物件字典內的所有數值均以 16 進制來表示。

物件在物件字典中的位置，乃是透過一 16-bit 的主索引來定址，如果某個物件內有很多個項目，那這些項目還必須再透過一個 8-bit 的子索引來定址，如果某個物件內只有一個項目，那用主索引便可對這個項目定址。

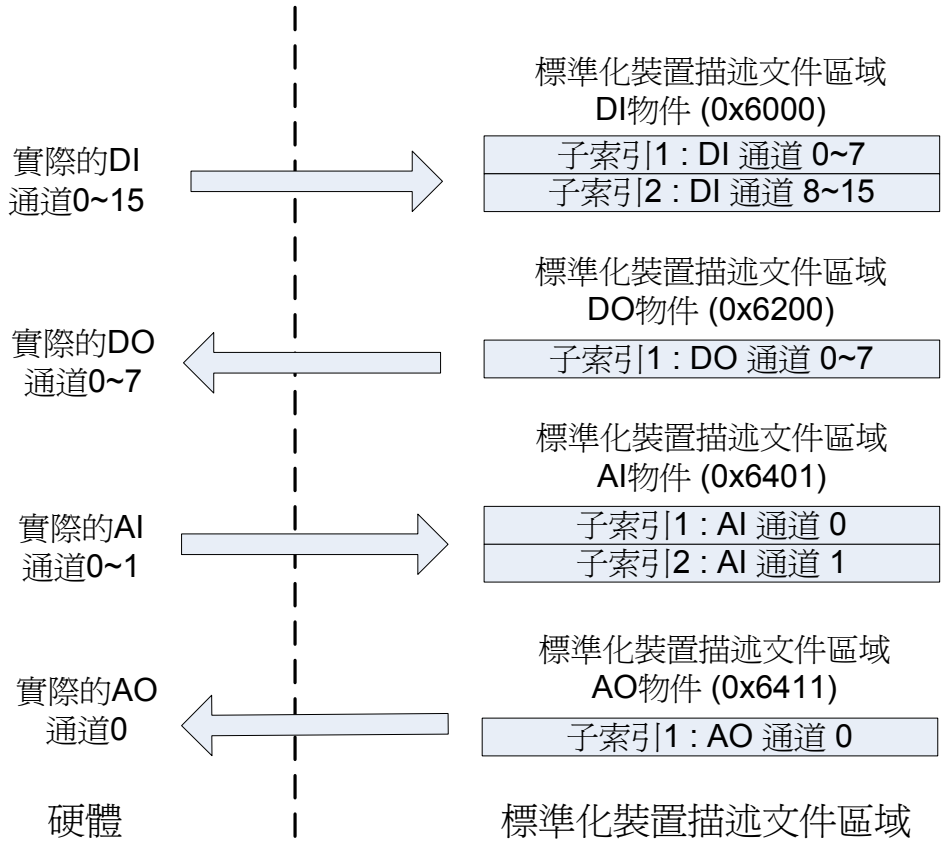
標準物件字典(standard object dictionary)之中，所有描述文件的位置如下：

主索引 (hex)	物件
0000	保留
0001-001F	靜態資料型別(Static Data Types)
0020-003F	複合資料型別(Complex Data Types)
0040-005F	製造商特定的複合資料型別 (Manufacturer Specific Data Types)
0060-007F	裝置描述文件特定的靜態資料型別
0080-009F	裝置描述文件特定的複合資料型別
00A0-0FFF	保留作更進一步的用途
1000-1FFF	通訊描述文件(communication profile)區域
2000-5FFF	製造商特定的描述文件區域 (Manufacturer Specific Profile Area)
6000-9FFF	標準化裝置描述文件區域 (Standardized Device Profile Area)
A000-BFFF	標準化介面描述文件區域 (Standardized Interface Profile Area)
C000-FFFF	保留作更進一步的用途

以標準化裝置描述文件區域爲例，假設一個 CANopen 的裝置有 16 個 DI、八個 DO、兩個 AI 以及一個 AO 的通道(channel)，這些通道的數值均會分別被對應到標準化裝置描述文件區域內的數個項目，如主索引 0x6000、0x6200、0x6401 和 0x6411 物件內的項目。

舉例來說，當 CANopen 的裝置抓取輸入通道的數值後，就會把這些數值存到主索引爲 0x6000 和 0x6401 物件內的項目。此外，裝置可分別輸出存在主索引爲 0x6200 和 0x6411 的物件內項目的數值到 DO 和 AO 通道上。

大致上的概念如下圖所示：



以下用 CAN-8423 爲例，若將一些 I-8000 或 I-87K 系列的模組裝在 CAN-8423 的 I/O 擴充插槽上，如下表：

模組名稱	安裝的擴充插槽編號	DO 通道數目	AO 通道數目	DI 通道數目	AI 通道數目
I-8063	0	4	0	4	0
I-87053	1	0	0	16	0
I-8053	3	0	0	16	0

當 CAN-8423 開機後，CAN-8423 會掃描安裝於其上所有模組的 I/O 通道。同時，這些 I/O 通道的數值也會分別被送到物件字典內特定的物件。

此外由於裝置在處理單一模組上的通道數值時，最小的資料單位爲 1 byte，如果單一模組上 DI 和 DO 通道的數值不滿 1 byte，這時候裝置會自動將其調整其長度爲 1 byte。

CAN-8423 會將 DI、DO、AI、AO 這些通道的 I/O 數值分別儲存到主索引值爲 0x6000、0x6200、0x6401 以及 0x6411 的物件項目。裝置在將 I/O 數值填入物件字典時，會遵循以下的規則：

- 被安裝在 CAN-8423 上的 I-8000/I87K 系列模組，較小擴充插槽編號上面的模組，其 I/O 通道的數值將會優先被放到物件字典。當 CAN-8423 處理完一個擴充模組上的所有 I/O 通道時，這時 CAN-8423 將會接著處理下一個編號的擴充插槽上面的擴充模組。
- 每一個類比通道的數值可以用 2 bytes 來儲存。
- 每一個數位通道用 1 bit 來儲存，若通道數目不滿 8 個，也就是數位通道數值不足 1 byte，裝置會自動將其補足(從 MSB 開始補 0)。

在套用了前述的規則之後，物件字典內相關物件的內容就會如下表所示：

主索引 / 子索引	0x6000 (DI 專用)	0x6200 (DO 專用)	0x6401 (AI 專用)	0x6411 (AO 專用)
0x00	9	1	9	4
0x01	DI0~DI3 (擴充插槽編號 0)	DO0~DO3 (擴充插槽編號 0)		
0x02	DI0~DI7 (擴充插槽編號 1)			
0x03	DI8~DI15 (擴充插槽編號 1)			
0x04	DI0~DI7 (擴充插槽編號 3)			
0x05	DI8~DI15 (擴充插槽編號 3)			

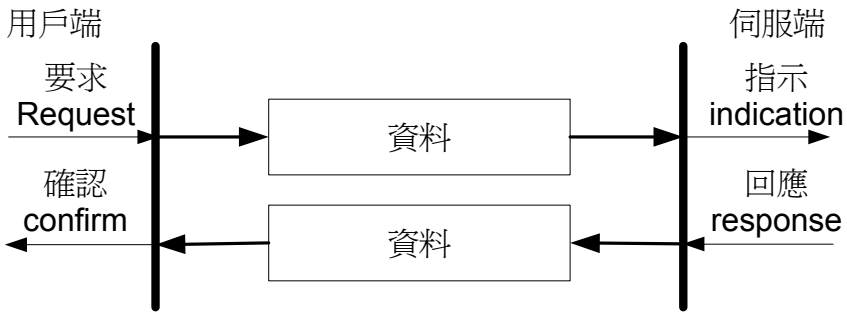
上表中的資料，也可以利用 CAN 僕端工具程式(CAN Slave Utility)來觀察。有關物件字典更細節的部份，以及 CAN 僕端工具程式的使用方式，可參照第 5 章和第 6 章。

應用程式

應用程式物件包含了裝置上，裝置與行程環境(process enviroment)之間互動的所有功能，其可被視爲是物件字典和實際行程(如 DI/DO/AI/AO)之間的橋樑。

3.2 SDO 介紹

SDO(Service Data Object，服務資料物件)的作用爲存取裝置物件字典的項目。藉由 SDO 的通訊方式，來建立兩個裝置之間點對點通訊的橋樑。SDO 的通訊模型依循著用戶端-伺服端(Client-Server)的關係，如下圖所示：



SDO 可分爲兩種，分別爲 RxSDO(Receive SDO)以及 TxSDO (Transmit SDO)，它們的 COB-ID 是分開的。這兩種 SDO 的區分方式，是從 CANopen 裝置的角度來看的。

舉例來說，若使用者欲傳送一 SDO 訊息給 CAN-8123/CAN-8223/CAN-8423，則此 SDO 對裝置而言就是 RxSDO。因此，這個 SDO 訊息的 COB-ID 就必須是裝置的 RxSDO COB-ID。若 CAN-8123/ CAN-8223/CAN-8423 希望對外傳送一個 SDO 訊息，則此 SDO 對裝置而言就是 TxSDO。這個訊息的 COB-ID 就必須是裝置的 TxSDO COB-ID。

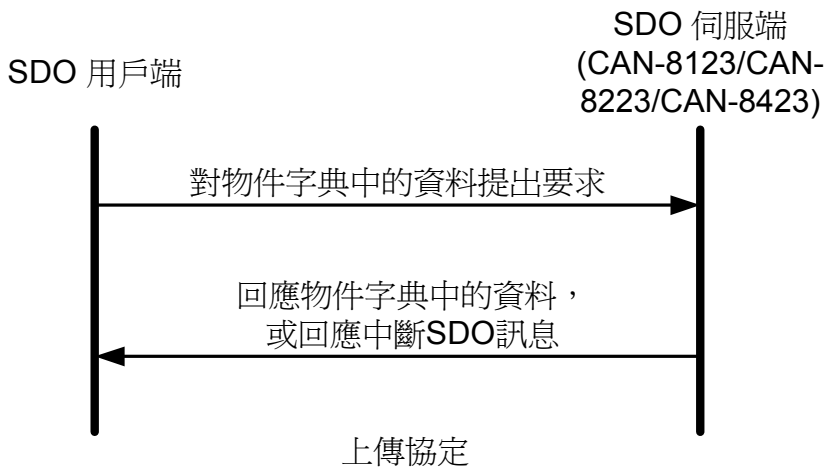
此外，所有 SDO 的傳輸，均需由 SDO 用戶端主動發起。而在 SDO 用戶端開始進行 SDO 傳輸之前，需先針對其需求選擇適當的 SDO 傳輸協定。

如果 SDO 用戶端必須要由 SDO 伺服端上的物件字典中獲取資訊，則需要使用上傳 SDO 協定(upload SDO protocol)或 SDO 區塊上傳協定(SDO block upload protocol)。前者適合用來傳輸較少量的資料，後者則適合用來傳輸較大量的資料。

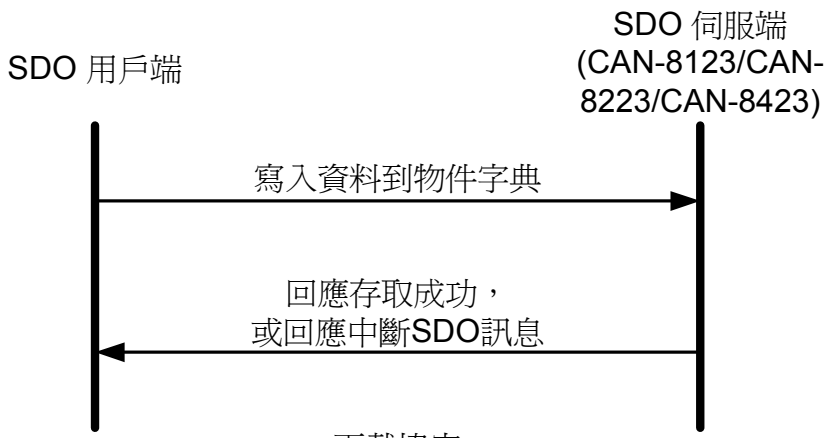
當 SDO 用戶端想要更改 SDO 伺服端的物件字典內容時，則可以透過下載 SDO 協定(download SDO protocol) 或 SDO 區塊下載協定(SDO block download protocol。這兩者之間的差異就如同於上傳 SDO 協定與 SDO 區塊上傳協定之間的差異。

此外，因爲物件字典中，不同的項目會有不同的存取類型，並不是所有物件字典的內容都能透過 SDO 的傳輸來進行存取。如果 SDO 用戶端試圖更動 SDO 伺服端物件字典中，不被允許存取的項目，這時 SDO 伺服端就會啓動異常中止 SDO 傳輸協定(Abort SDO Transfer Protocol)，以終止 SDO 的傳輸。

CAN-8123/CAN-8223/CAN-8423 僅支援作爲 SDO 伺服端。因此，它只能被動等待 SDO 用戶端發起 SDO 的傳輸。有關 CAN-8123/CAN-8223/CAN-8423 的上傳協定和下載協定，其大致上的概念可參考下圖：



上傳協定



下載協定

3.3 PDO 介紹

PDO 的通訊模式 (Communication Mode)

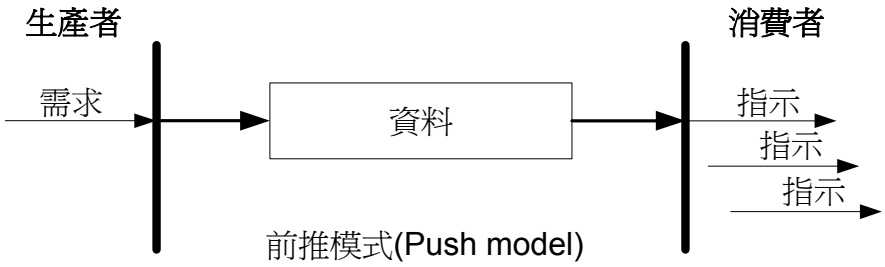
PDO(Process Data Object)常被用來傳輸即時性的資料，像是 DI、DO、AI、AO 等等。由於 CAN bus 的傳輸資料格式之限制，因此，透過 PDO 來傳輸資料，每一個 PDO 內最多僅能放入 8 bytes 的資料。另外，因爲 PDO 的訊息在傳輸時沒有傳輸協定上的額外負擔(overhead)，在資料的傳輸上比起其它 CANopen 的通訊物件都更有效率。

PDO 的傳送與接收通訊模型，乃是依據生產者/消費者(producer/consumer)的架構所實作的，這種通訊模式也可稱爲推挽式(push/pull)模型。

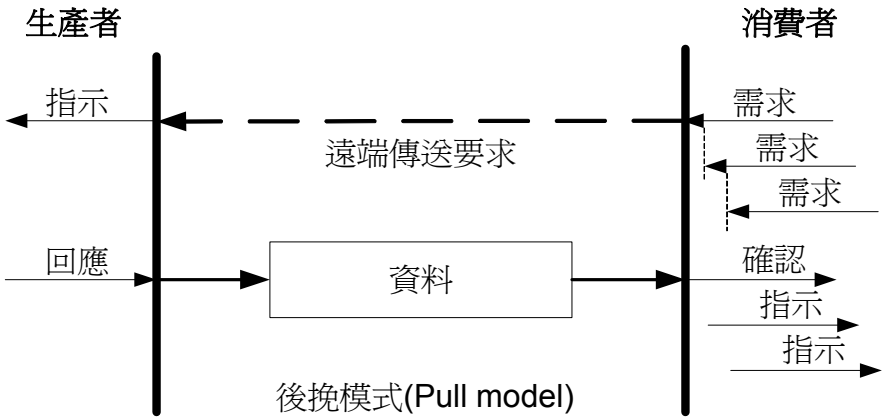
當 PDO 前推模式(push mode)開始進行時，必需有一個 CANopen 的裝置扮演 PDO 生產者的角色，而 PDO 消費者則可有可無。在 PDO 生產者送出 PDO 訊息之前，這一個 PDO 訊息必需要先在 CAN bus 上的仲裁中獲勝。接著，CAN bus 上的每一個 PDO 的消費者都會收到這個 PDO 訊息，並決定接受或著丟棄這個 PDO 訊息。

在 PDO 後挽模式(pull mode)中，會由一個 PDO 消費者先傳送一個遠端傳送要求(remote transmit request, RTR)給 PDO 生產者。然後 PDO 生產者根據這個遠端傳送要求的訊息，它會回覆一個具有相同 COB-ID 的 PDO 訊息給在 CAN bus 上的每一個 PDO 消費者。

PDO 的通訊架構圖如下所示：



前推模式(Push model)



PDO 和 SDO 一樣，也可分爲兩種，分別爲 RxPDO 以及 TxPDO，它們的 COB-ID 一樣是分開的。這兩種 PDO 的區分方式，也和 SDO 一樣，同樣是從 CANopen 裝置的角度來出發的。

裝置用來對外傳輸資料的 PDO 就是 TxPDO，其常被用來傳輸裝置上 DI/AI 通道的數值。而在裝置用來接收資料的 PDO，就是 RxPDO，其常被拿來改變裝置上 DO/AO 通道的數值。

以 CAN-8123/CAN-8223/CAN-8423 爲例，如果一個 PDO 生產者傳送了一個 PDO 訊息給 CAN-8123/CAN-8223/CAN-8423，這個訊息的 COB-ID 必需爲裝置的 RxPDO COB-ID，這是因爲從 CAN-8123/CAN-8223/CAN-8423 的角度來看，這動作是屬於 PDO 的接收。相對地，當 PDO 的消費者傳送遠端傳送要求(RTR)訊息給 CAN-8123/CAN-8223/CAN-8423，要求其對外傳送 TxPDO 時，此一訊息的 COB-ID 就必需使用 CAN-8123/CAN-8223/CAN-8423 的 TxPDO COB-ID，這是因爲對於 CAN-8123/CAN-8223/ CAN-8423 來說，回覆遠端傳送要求的動作是屬於 PDO 的傳送。

PDO 的觸發模式 (Trigger Modes)

對 PDO 生產者而言，PDO 訊息傳輸的觸發條件可以分爲三種。分別爲事件驅動(event driven)、時間驅動(timer driven)以及遠端要求(remote request)條件。以下將對這三種條件做描述。

事件驅動 (Event Driven)

PDO 的收送可以被一物件特定事件(object specific event)的發生所觸發。

以循環同步傳輸型態的 PDO 爲例，當裝置接收到某一特定數量的 SYNC 訊息時，便會觸發 PDO 的收送。

對非同步傳輸型態的 PDO 而言，裝置特定事件是由 CANopen 的規範，DS-401 v2.1 所定義的。根據規範，當被映射到 DI 通道的 TxPDO，其傳輸型態被設定爲非同步時，則只要 DI 通道數值發生變化，便會觸發裝置對外傳送此 TxPDO。

時間驅動 (Timer Driven)

PDO 的傳輸除了可以被裝置特定事件的發生所觸發之外，也可以設定成，若沒有事件發生的狀態持續一段特定時間後，便讓裝置對外傳送 PDO。

舉例來說，使用者可以設定 TxPDO 通訊參數內的事件計時器，則當經過了事件計時器內所記錄的特定時間間隔，且在這段時間內均未發生其他的事件，CAN-8123/CAN-8223/CAN-8423 就會對外傳送此 TxPDO。

遠端要求 (Remote Request)

如果 PDO 的傳輸型態被設定唯遠端傳送要求、非同步或非循環同步，則只有在接收到其他 PDO 消費者所傳輸的遠端傳送要求(RTR)之時，PDO 傳輸才會被觸發。

PDO 傳輸型態 (PDO Transmission Types)

一般來說，PDO 的傳輸型態可大略分爲兩種，分別爲同步和非同步。在同步傳輸型態下，必須要藉由 SYNC 訊息的接收，才會觸發 PDO 的收送。在非同步型態下，PDO 傳輸的觸發是與 SYNC 物件相互獨立的。

同步傳輸型態可再細分爲三種，分別爲非循環同步、循環同步以及唯遠端傳送要求同步。而非同步傳輸型態可再細分爲兩種，分別爲唯遠端傳送要求非同步和非同步。

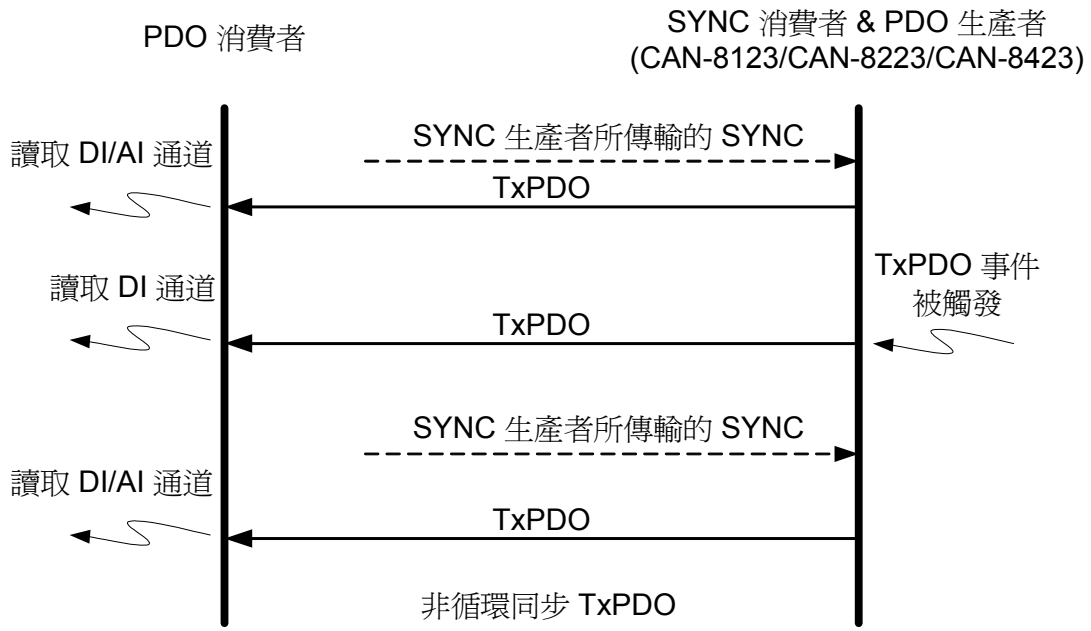
非循環同步(acyclic synchronous)

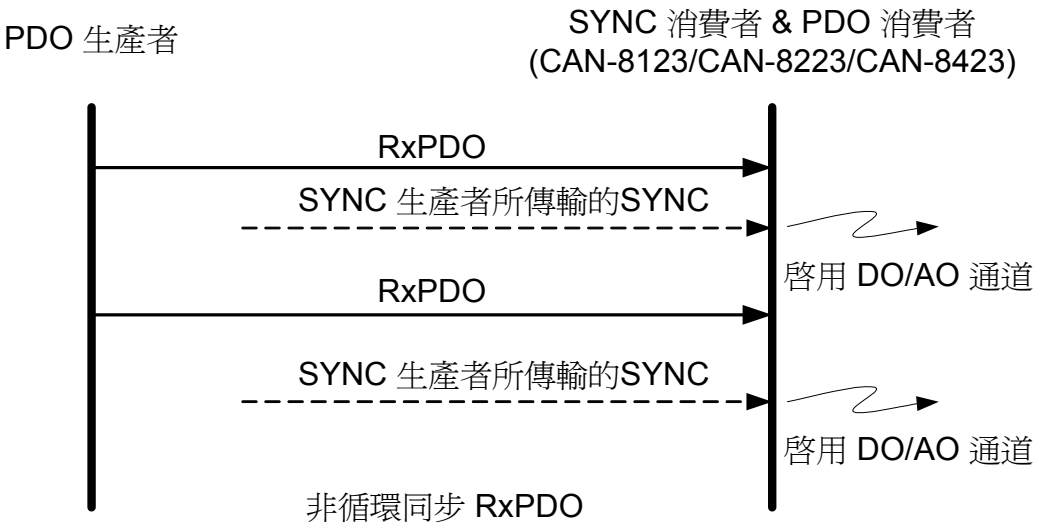
若某 TxPDO 的傳輸型態被設定爲非循環同步，則裝置上若先發生物件特定事件，接著又收到由 SYNC 生產者傳送的 SYNC 物件，此時裝置就會對外傳送此 TxPDO 訊息給 PDO 的消費者。

或著裝置收到此 TxPDO 的 RTR 訊息，此時裝置也會對外傳送此 TxPDO。

若某 RxPDO 的傳輸型態被設定爲非循環同步，則裝置必需要在接收到 SYNC 物件之後，才會啓用在接收到此 SYNC 物件之前所收到且尚未被啓用的 RxPDO 物件。每一次收到 SYNC 物件，裝置都會對尚未被啓用的 RxPDO 物件進行啓用。

關於非循環同步的 PDO 傳輸型態，使用者可參考下圖：





循環同步(cyclic synchronous)

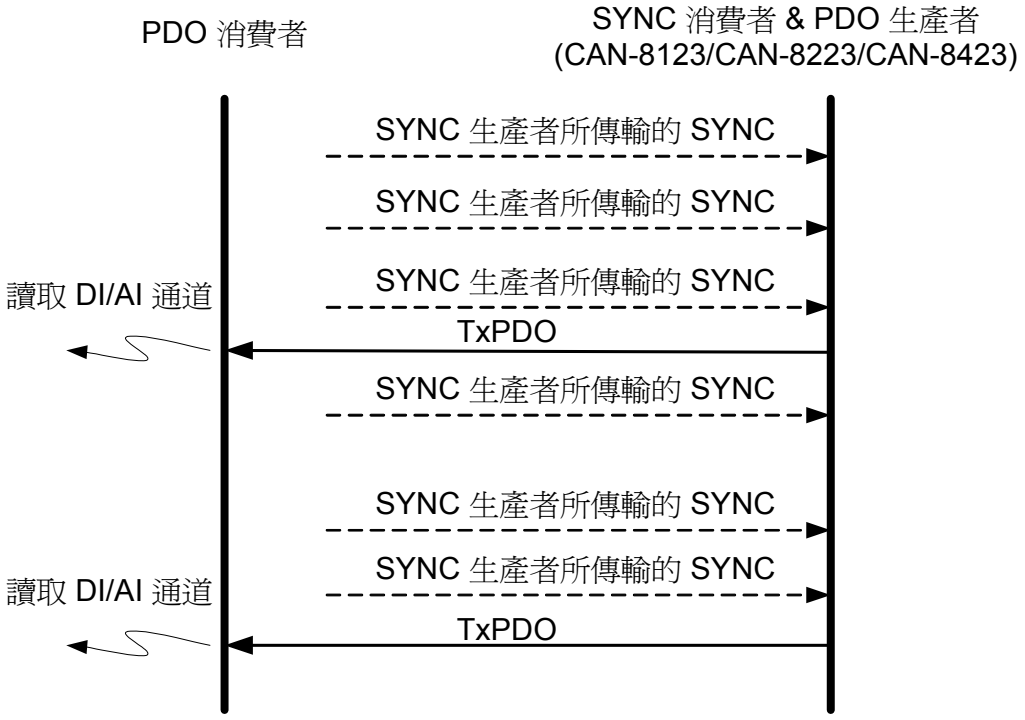
若某 TxPDO 的傳輸型態被設定爲循環同步，則裝置必須要接收到特定數量的 SYNC 物件之後，才會觸發此 TxPDO 的傳輸，SYNC 物件的特定數量最大可設定爲 240。

舉例來說，若某 TxPDO 被設定爲當接收到 3 個 SYNC 物件後便進行觸發，則 CAN-8123/CAN-8223/CAN-8423 將會在收到 3 個 SYNC 物件後，進行此 TxPDO 的傳送。

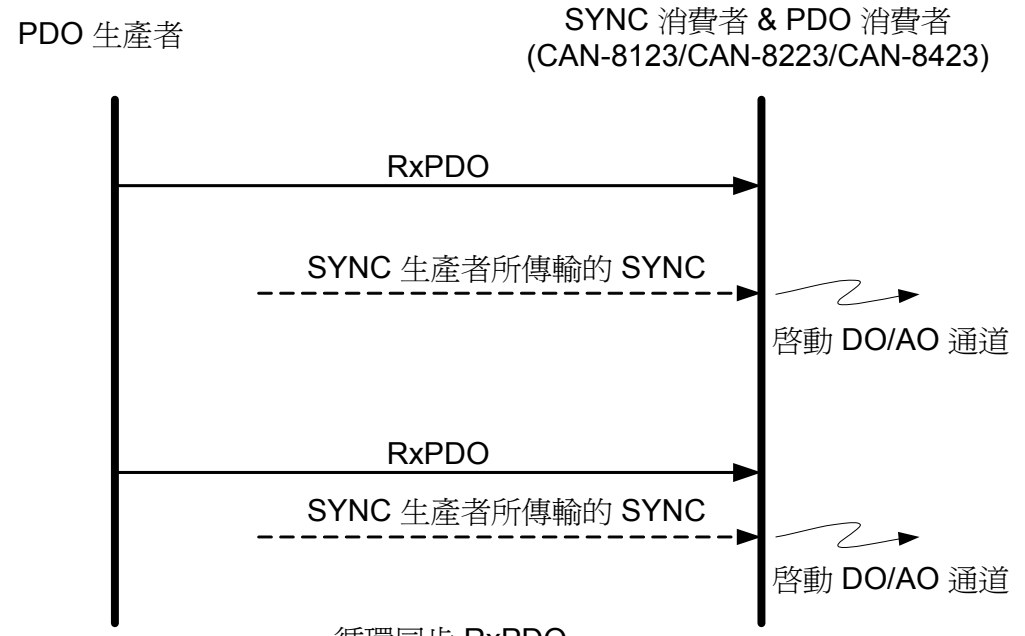
或著裝置收到此 TxPDO 的 RTR 訊息。此時裝置也會對外傳送此 TxPDO 訊息給 PDO 的消費者。

非循環同步與循環同步傳輸型態的 RxPDO 觸發方式相同，均是藉由接收 SYNC 物件來啓用 RxPDO，與收到的 SYNC 物件數目沒有關係。

關於循環同步的 PDO 傳輸型態，使用者可參考下圖：



循環同步 TxPDO



循環同步 RxPDO

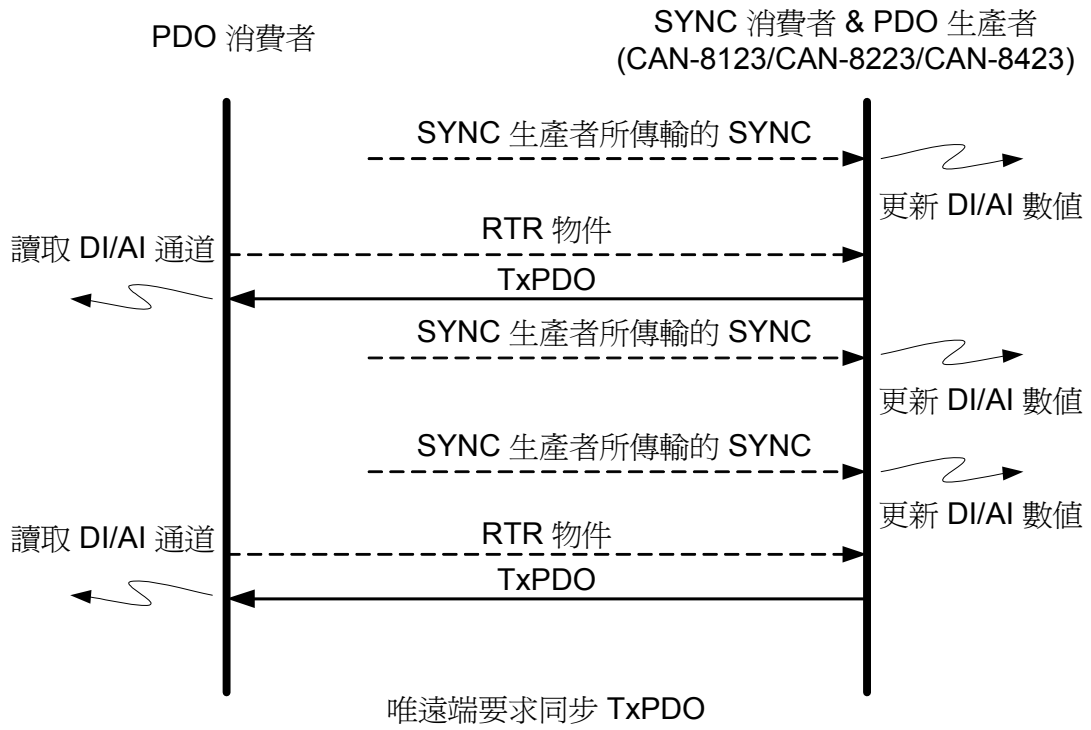
唯遠端傳送要求同步(RTR-only synchronous)

若某 TxPDO 的傳輸型態被設定爲唯遠端傳送要求同步，和其他傳輸型態不一樣的是，當裝置接收到 RTR 之後，裝置並不會去更新 TxPDO 裡面的數值，會直接就對外傳送此 TxPDO。此時 TxPDO 內記錄的有可能是過時的資料。

若使用者想要獲得有最新數值的 TxPDO，需先傳送 SYNC 訊息給裝置，讓裝置更新 TxPDO 內記錄的數值，再傳送此 TxPDO 的 RTR 訊息給裝置，則此時，裝置對外傳送的 TxPDO 就會有最新的數值。

舉例來說，若這個 TxPDO 內的內容映射到 DI/AI，則只有在裝置接收到 SYNC 物件後，裝置才會抓取最新的 DI/AI 數值到 TxPDO 之中。此時再傳送此 TxPDO 的 RTR 訊息給裝置，則裝置就會對外傳送最新數值的 TxPDO。

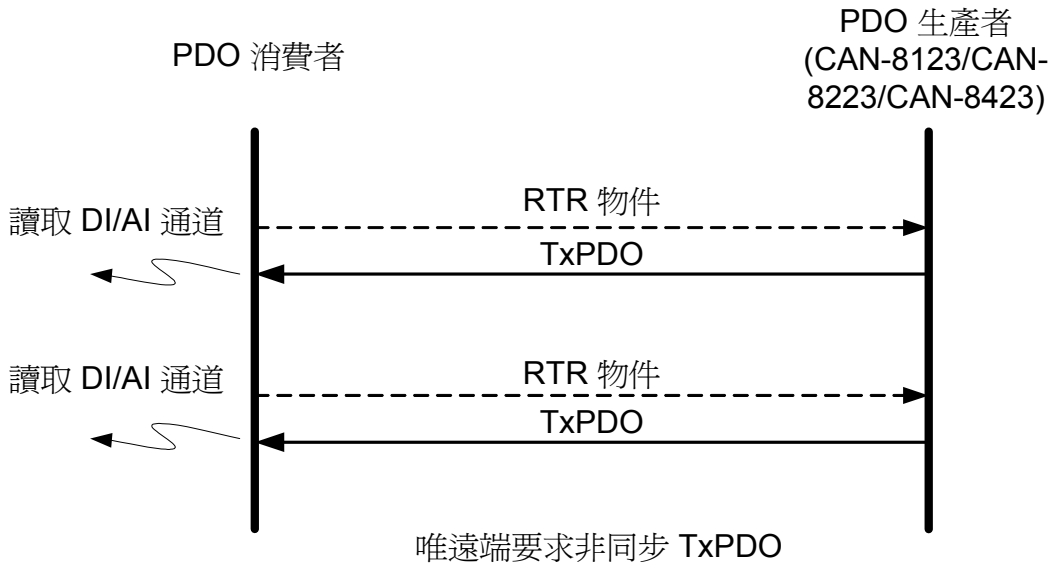
只有 TxPDO 能被設定成這種傳輸型態。關於唯遠端傳送要求同步的 PDO 傳輸型態，使用者可參考下圖：



唯遠端要求同步 TxPDO

唯遠端傳送要求非同步(RTR-only asynchronous)

只有 TxPDO 的傳輸型態能被設定爲唯遠端傳送要求非同步，對此傳輸型態的 TxPDO 來說，只有在裝置接收到此 TxPDO 的 RTR 訊息之後，裝置才會對外傳輸此 TxPDO。使用者可參考下圖：

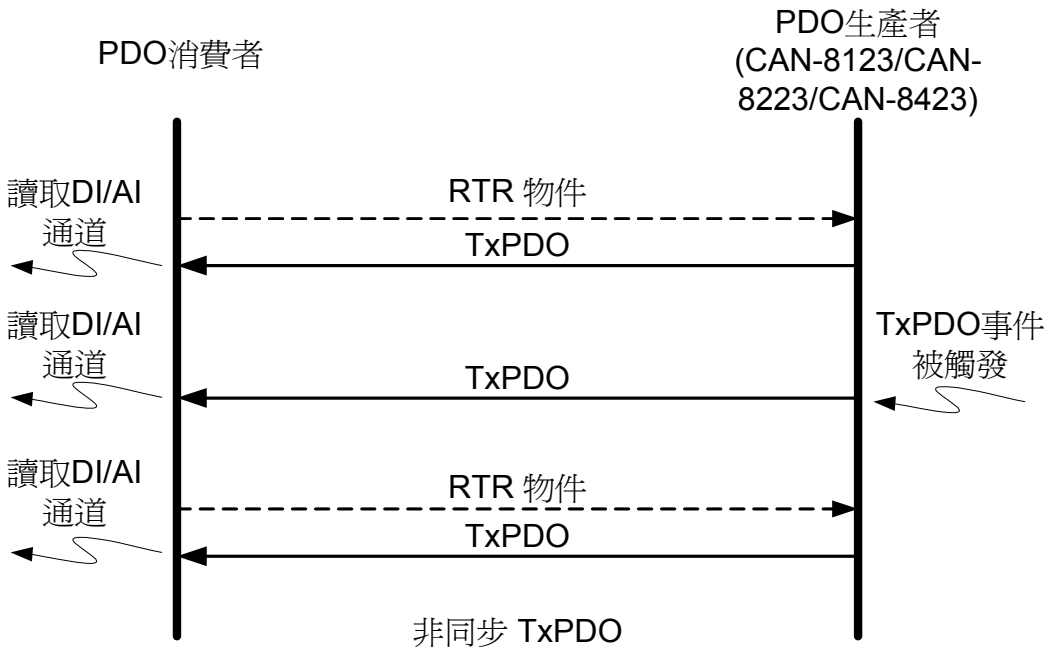


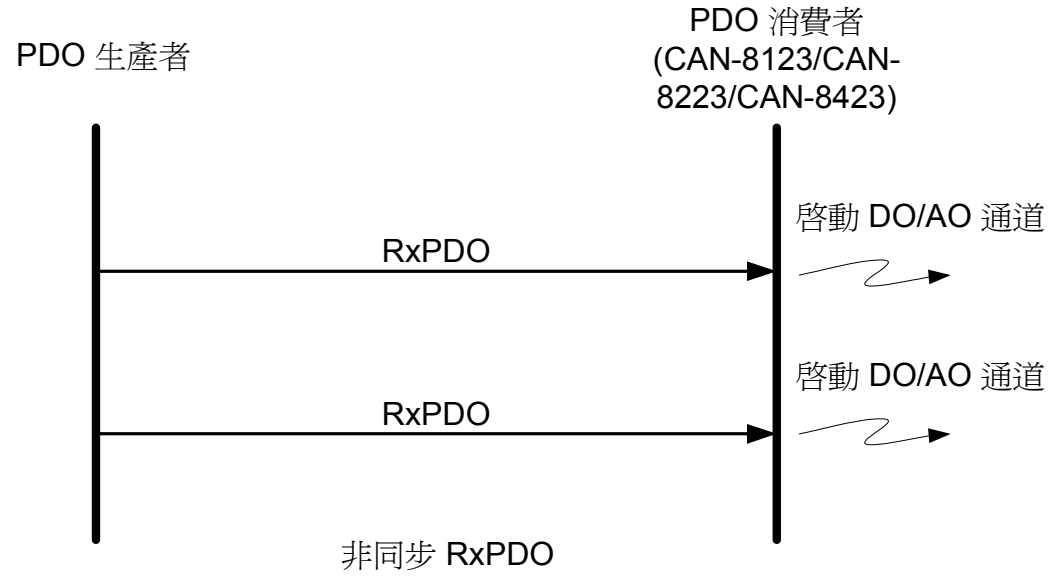
非同步(asynchronous)

若某 TxPDO 的傳輸型態被設定爲非同步，則 TxPDO 訊息的觸發可以透過以下兩種方式來達成，一爲裝置收到此 TxPDO 的 RTR 訊息，一爲先前提到過，裝置上特定事件的發生。

另外，若 RxPDO 的傳輸型態被設定爲非同步，則當裝置接收到 RxPDO 時，裝置便會直接啓用 RxPDO，不需要接收 SYNC 訊息。

而 CAN-8123/CAN-8223/CAN-8423 在開機之後，不管是 TxPDO 還是 RxPDO，其預設的傳輸型態就是非同步。有關非同步傳輸型態使用者可參考下圖：


PDO消費者
PDO生產者
(CAN-8123/CAN-
8223/CAN-8423)
讀取DI/AI
通道
RTR 物件
TxPDO
讀取DI/AI
通道
TxPDO
TxPDO事件
被觸發
讀取DI/AI
通道
RTR 物件
TxPDO
非同步 TxPDO


PDO 生產者
PDO 消費者
(CAN-8123/CAN-
8223/CAN-8423)
啓動 DO/AO 通道
RxPDO
啓動 DO/AO 通道
RxPDO
非同步 RxPDO

抑制時間 (Inhibit Time)

由於 CAN bus 的仲裁機制，在 CANopen 中，COB-ID 較小者，有著較高的傳輸優先權。

舉例來說，如果 CAN bus 上面有兩個節點，其中一個想要傳送 COB-ID 爲 0x181 的訊息，另外一個想要傳送 COB-ID 爲 0x182 的訊息。當這兩個節點同時傳送訊息到 CAN bus 上時，只有 COB-ID 爲 0x181 的訊息能夠被成功傳送，這是因爲它有著較高的傳輸優先權。COB-ID 爲 0x182 的訊息必須等到匯流排再次空閒(bus idle)，才能被傳送。這樣的仲裁機制可以擔保在訊息傳輸過程當中有衝突發生時，至少能夠有一個節點成功的傳輸訊息。

然而，如果 COB-ID 爲 0x181 的訊息不斷的被傳送，那 COB-ID 爲 0x182 的訊息將永遠沒有機會被傳送。換言之，如果較高優先權的訊息連續的被傳送，那較低優先權的 CAN 訊息就不可能會傳輸成功，這就是這種仲裁機制的缺點。

爲了避免這種狀況的發生，CANopen 的規範就對每一個 PDO 的物件定義了所謂的抑制時間(單位爲 100us)，當同一個 PDO 的物件在連續傳送的狀態下，PDO 訊息與 PDO 訊息之間的時間間隔，必須大於此 PDO 的抑制時間，如此就可避免上述情況的發生。

事件計時器 (Event Timer)

此參數只對 TxPDO 有作用。若某 TxPDO 的事件計時器不爲 0，且其爲非同步傳輸模式，則計時器便會開始倒數，當計時器倒數到 0 時，就會被視爲發生一次事件，並且此事件會觸發 TxPDO 的傳輸。事件計時器所記錄的數值，其時間單位爲 1ms。

PDO 映射參數 (PDO Mapping Parameter)

CANopen 裝置上的 PDO 映射參數，提供了 PDO 訊息與實際 I/O 資料之間的連結。而 PDO 訊息中每個 byte 所代表的意義(即 PDO 映射參數的內容)，又可以透過 SDO 訊息來加以改變。所有的 PDO 映射參數皆存放在通訊描述文件區域之中(Communication Profile Area，物件字典中的 1000-1FFF)。根據 CANopen 的規範 CiA DS-401，RxPDO 和 TxPDO 預設的映射參數具有下面的特性：

- 預設的 PDO 映射中應分別支援 4 組的 TxPDO 和 4 組的 RxPDO。

- 第 1 組 RxPDO 和 TxPDO 的映射，分別被對應到 DO 和 DI。

- 第 2、3、4 組的 RxPDO 映射被對應到 AO，第 2、3、4 組的 TxPDO 被對應到 AI。

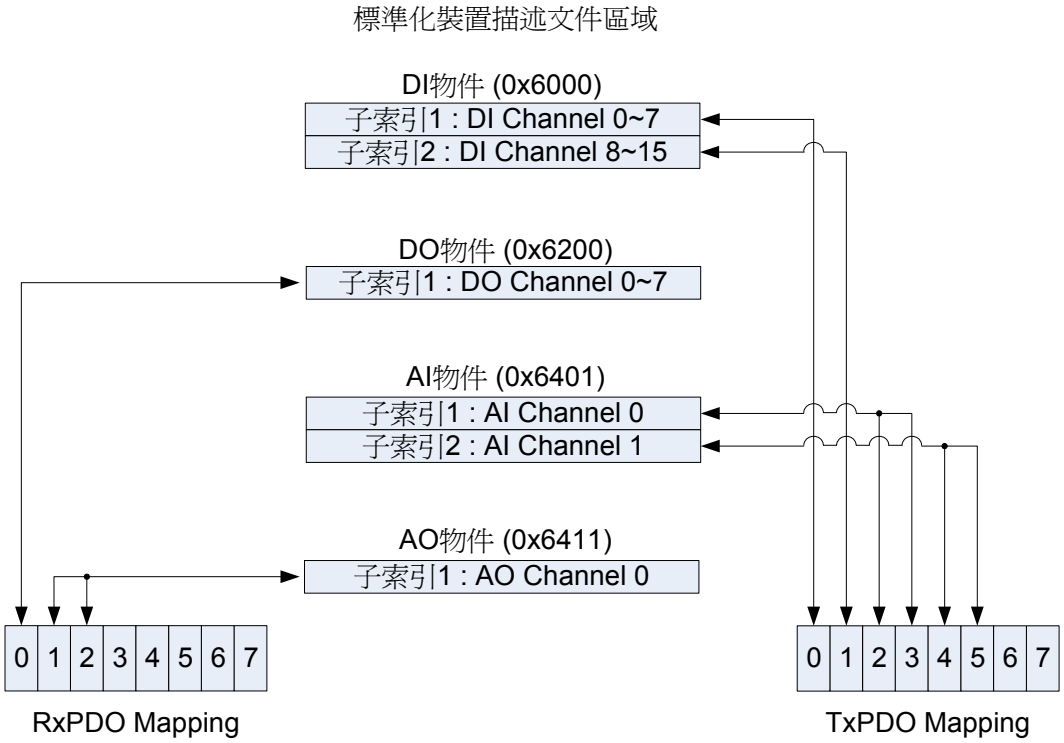
- 在預設的狀況下，前 8 個 DO 通道的物件會被映射到第 1 個 RxPDO，而前 8 個 DI 的物件會被映射到第 1 個 TxPDO(一個 DO/DI 物件的大小爲 1 byte，其內至多可含 8 個 DO/DI 的通道，而一個 RxPDO/TxPDO 內的資料欄位最多可容納 8 bytes，也就是 8 個 DO/DI 的物件)。若一個裝置所支援的 DO/DI 物件超過 8 個，則裝置就會從第 5 個 RxPDO/TxPDO 開始映射第 8 個以後的 DO/DI 物件，因爲第 2 個到第 4 個 RxPDO/TxPDO 預設是保留給 AO/AI 物件使用的。而在預設的情況下，前 12 個 AO 的物件會依序被映射到第 2、3、4 個 RxPDO，而前 12 個 AI 的物件會依序被映射到第 2、3、4 個 TxPDO (一個 AO/AI 物件的大小爲 2 byte，內含 1 個 AO/AI 的通道，而一個 RxPDO/TxPDO 內的資料欄位最多可容納 8 bytes，也就是 4 個 AO/AI 的物件)。若一個裝置所支援的 AO/AI 物件數量超過 12 個，則必須要從第 1 個未被映射的 RxPDO/TxPDO 開始映射第 12 個以後的 AO/AI 物件。

於下以例子來進一步說明，若在 CAN-8123/CAN-8223/CAN-8423 的物件字典之中，有 11 個 DO 和 13 個 AO 的物件。則在預設的情況下，前 8 個 DO 的物件會被映射到第 1 個 RxPDO，後 3 個 DO 的物件會被映射到第 5 個 RxPDO。前 4 個 AO 的物件會被映射到第 2 個 RxPDO，接下來的 8 個 AO 的物件將會被分別映射到第 3 個 RxPDO 和第 4 個 RxPDO，另外因爲第 5 個 RxPDO 已經用來映射 DO 了，故最後一個 AO 的物件會被映射到第 6 個 RxPDO。

在使用 PDO 進行通訊之前，針對此 PDO，生產者和消費者必須要有相同的 PDO 映射參數。一方面，PDO 生產者需要 PDO 的映射參數用來決定要傳送的 PDO 該填入哪些資料。另一方面，PDO 消費者需要 PDO 的映射參數才知道接收到的 PDO 訊息之中，每一個 byte 所代表的實際意義。

除了預設的 PDO 映射之外，使用者也可以根據實際使用的需求，自行定義 PDO 映射參數。

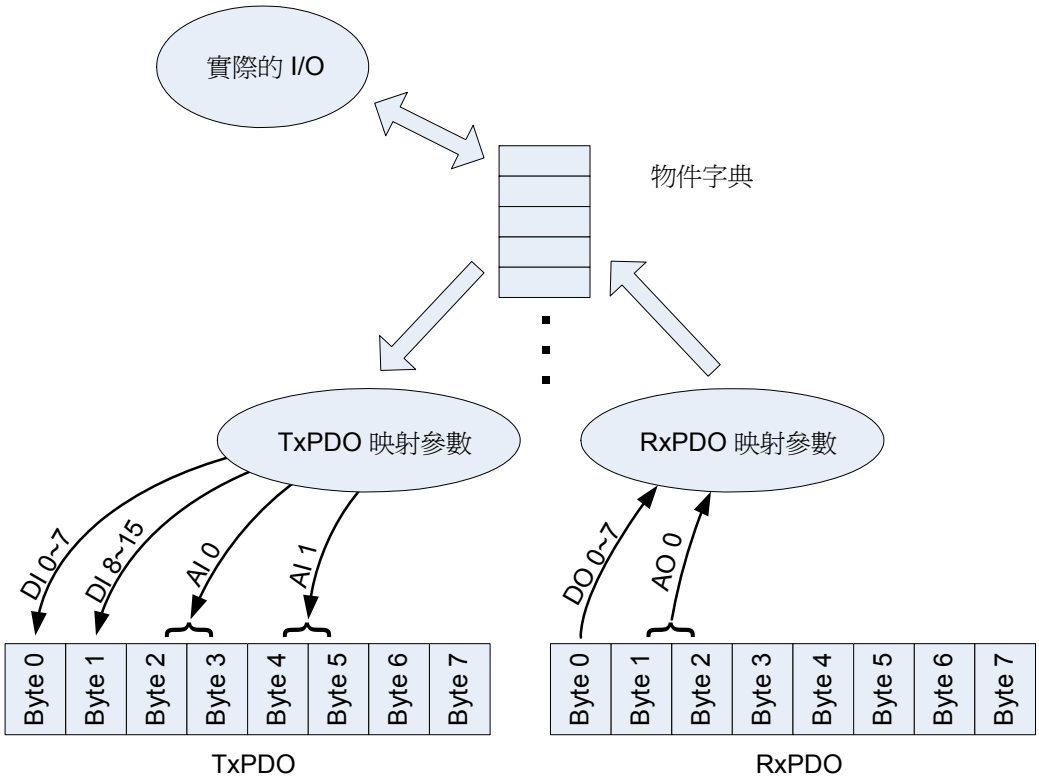
若一個 CANopen 的裝置有 16 個 DI、8 個 DO、2 個 AI 和 1 個 AO 通道，且這些輸入和輸出通道的數值已經分別被存入物件字典內的數個不同項目。除了預設的 PDO 映射之外，使用者也可以自行定義 PDO 映射參數，將這些通道的數值映射給使用者自定義的 PDO。於此處我們把這些通道的數值分別映射到 1 組 RxPDO 和 1 組 TxPDO 上，如下圖所示：



若某 CANopen 的裝置收到此 RxPDO 訊息，根據上圖所描述的 RxPDO 映射關係，第 1 個 byte 會被解譯成 DO 通道 0~7 的輸出值，其他兩個 byte 會被解譯成 AO 通道 0 的輸出值。在解譯完 RXPDO 訊息的資料後，裝置會輸出解譯後的資料到 DO 和 AO 的通道上。

而 TxPDO 訊息的傳送和 RxPDO 訊息的接收剛好相反，當 CANopen 裝置上發生觸發傳送此 TxPDO 的事件之後，裝置會依照上圖的 TxPDO 映射關係，把 DI 和 AI 通道的數值填到 TxPDO 之中，然後裝置再傳送 TxPDO 訊息給 PDO 的消費者。於此處，裝置會把 DI 通道 0~7 和 DI 通道 8~15 的數值填到 TxPDO 的前 2 byte，把 AO 通道 0 的數值填到第 3~4 byte，把 AO 通道 1 的數值填到第 5~6 byte。

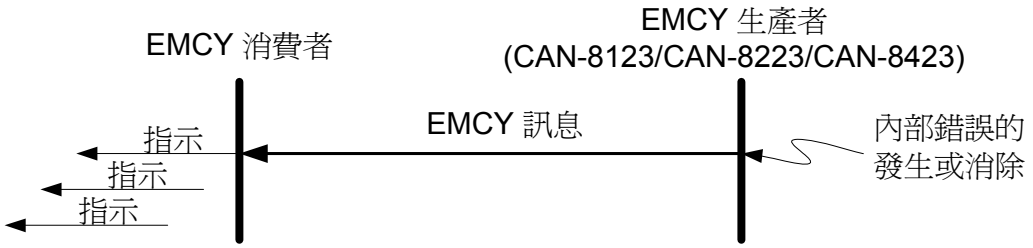
物件字典、PDO 映射參數和 PDO 訊息三者之間的關係，如下所示：


實際的 I/O
物件字典
TxPDO 映射參數
RxPDO 映射參數
DI 0~7
DI 8~15
AI 0
AI 1
DO 0~7
AO 0
Byte 0
Byte 1
Byte 2
Byte 3
Byte 4
Byte 5
Byte 6
Byte 7
TxPDO
RxPDO

3.4 EMCY 介紹

若裝置的錯誤狀態發生改變(發生內部錯誤，或錯誤被排除)，此時裝置就會對外傳送 EMCY 訊息。EMCY 的通訊模型爲生產者/消費者架構，在 CANopen 的裝置偵測到錯誤狀態發生改變之後，就會被傳送 EMCY 訊息給 EMCY 消費者(一錯誤事件，對應一 EMCY 訊息)。如果裝置的內部錯誤狀態沒有改變，那就裝置就不會傳送 EMCY 訊息。同一個 EMCY 訊息可多個 EMCY 消費者所接收。

CAN-8123/CAN-8223/CAN-8423 僅支援作爲 EMCY 訊息的生產者。有關 EMCY 訊息的傳輸行爲，使用者可參考下圖：



一個 EMCY 訊息包含 8 bytes 的資料，其被稱之爲緊急物件資料(emergency object data)，其結構如下表所示：

Byte	0	1	2	3	4	5	6	7
內容	Emergency 錯誤碼 (Emergency Error Code)		Error 暫存器 (Error register)	製造商特定的錯誤區域 (Manufacturer specific Error Field)				

我們將於 5.3 節對緊急物件資料中的各個欄位詳細進行說明。

以 CAN-8123/CAN-8223/CAN-8423 爲例，不管裝置內部發生了什麼錯誤，裝置都會對外傳送 EMCY 訊息。若重複發生相同的錯誤，針對此錯誤，裝置只會傳送一次 EMCY 訊息。然而，如果在 CAN-8123/ CAN-8223/CAN-8423 上出現了不同類型的新錯誤，這事件就會觸發裝置再次對外傳送 EMCY 訊息。

而裝置上只要有一個錯誤被排除，裝置便會對外傳送一個 EMCY 訊息，這訊息包含了“00 00”的 Emergency 錯誤碼，另外 Error 暫存器和製造商特定的錯誤區域這兩個欄位則會記錄裝置上尚未被排除的錯誤。

總而言之，使用者可以藉由檢查 EMCY 訊息，了解裝置上發生了什麼樣的錯誤，並且針對這些錯誤做出適當的處理。

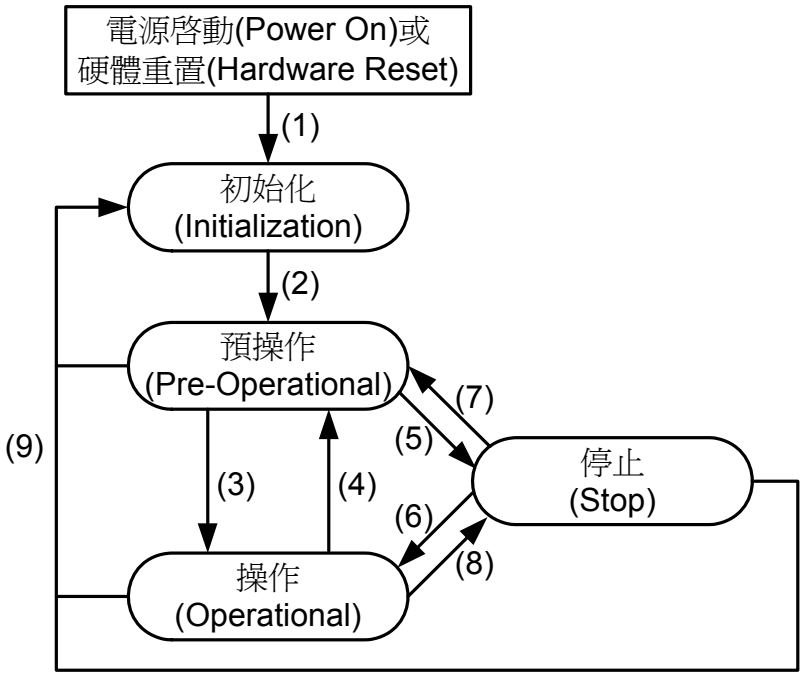
3.5 NMT 介紹

在 CANopen 內，所謂的 NMT 通訊物件，其通訊模型乃依循著節點導向(node-oriented)以及主從式(master-slave)的架構。在一個 CAN bus 的網路中，只有一個 CANopen 的裝置能夠作爲 NMT 主端，其他的節點則作爲 NMT 僕端。每一個 NMT 僕端都是獨一無二的，可藉由其節點 ID (1~127)來對其進行識別。

NMT 因應不同的用途支援了兩種協定，一爲模組控制協定(module control protocol)，一爲錯誤控制協定(error control protocol)。透過 NMT 模組控制協定，可以控制節點進入不同的狀態，像是起始(installing)、預操作(pre-operational)、操作(operational)、以及停止(stopped)等狀態。而錯誤控制協定則讓使用者能夠對網路中的遠端錯誤(remote error)進行偵測，可以用來確認節點的是否存活。

3.5.1 模組控制協定

在介紹模組控制協定(Module Control Protocol)之前，讓我們先將焦點放在 NMT 狀態機制(NMT state mechanism)的架構上。下圖列出了各 NMT 狀態(NMT state)之間的關係以及各 NMT 從端(NMT slave)其 NMT 狀態的轉換機制。



狀態機制(State Mechanism)圖

(1)	電源啓動或硬體重置後，裝置即自動進入初始化
(2)	初始化結束後即自動進入預操作狀態
(3),(6)	“啓動遠端節點(Start Remote Node)”指示(indication)
(4),(7)	“進入預操作狀態”指示
(5),(8)	“停止遠端節點(Stop Remote Node)”指示
(9)	“節點重置”或“通訊重置(Reset Communication)”指示

當初始化結束之後，裝置即進入預操作狀態。此時，裝置可以根據收到的模組控制協定訊息，切換到不同的 NMT 狀態。在不同的 NMT 狀態下，裝置可使用的通訊物件會有所差異。舉例來說，裝置僅能在操作狀態下進行 PDO 訊息的傳送和接收。在下表中將列出在不同 NMT 狀態下，CANopen 裝置可以使用的通訊物件。

	起始 (Installing)	預操作 (Pre-operational)	操作 (Operational)	停止 (Stopped)
PDO			O	
SDO		O	O	
SYNC		O	O	
Time Stamp		O	O	
EMCY		O	O	
Boot-Up	O			
NMT		O	O	O

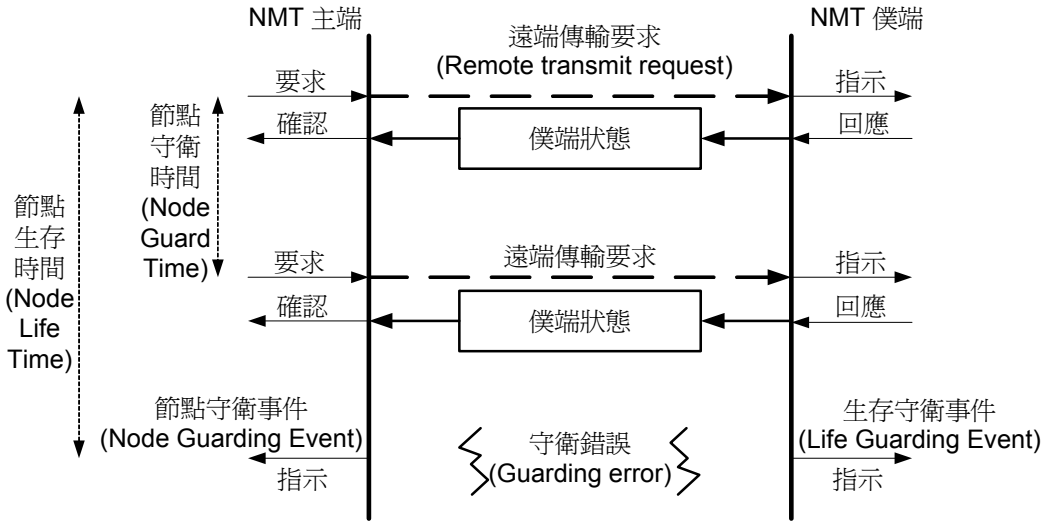
3.5.2 錯誤控制協定

CAN-8123/CAN-8223/CAN-8423 所支援的錯誤控制協定(Error Control Protocols)共有兩種，分別爲節點守衛協定(Node Guarding Protocol)與心跳產生者協定(Heartbeat Producer Protocol)。根據 CANopen 的規範，CANopen 的裝置在同一時間內僅能使用一種錯誤控制協定，同時使用兩種錯誤控制協定是不被允許的，使用者可以在實際的應用中，於 CAN-8123/CAN-8223/CAN-8423 上啓用這項功能。

下面將針對此兩種錯誤控制協定做進一步介紹。

節點守衛協定 (Node Guarding Protocol)

節點守衛協定的通訊模型依循著主僕(Master/Slave)的關係，使用者可以透過這個協定監測網路中 CANopen 節點的狀態。其通訊協定如下圖所示：



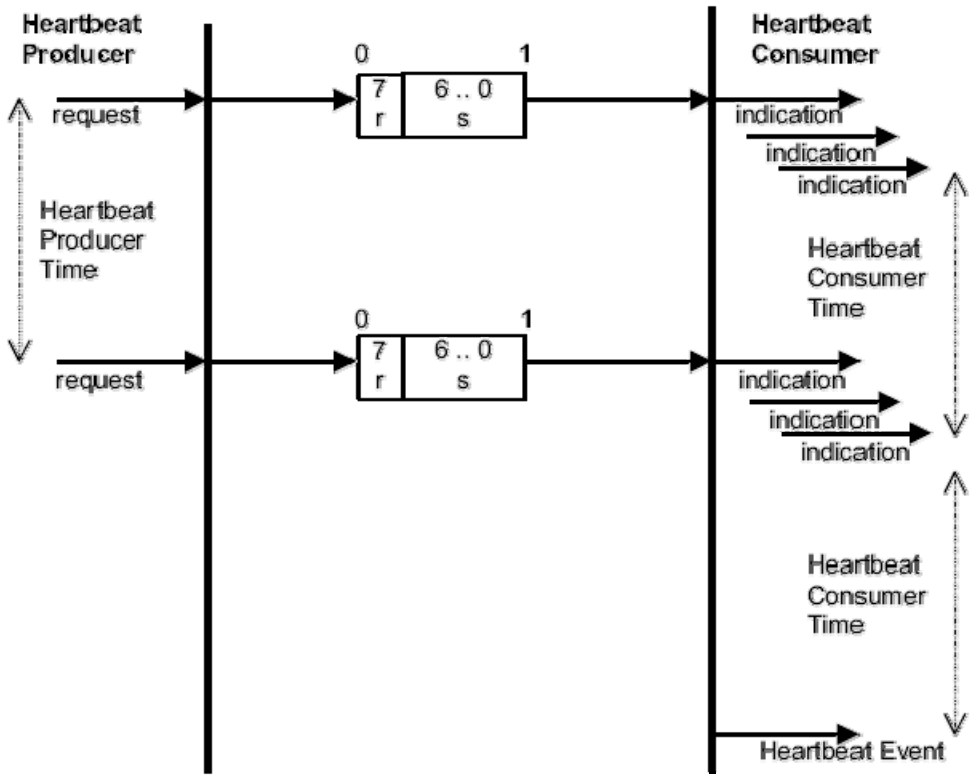
每隔一段固定的時間間隔，NMT 的主端就會去對每一個 NMT 僕端作輪詢(poll)的動作。此一時間間隔即所謂的節點守衛時間(Node Guard Time)，使用者可以對不同的 NMT 僕端，設定不同的節點守衛時間。

NMT 僕端收到主端的詢問之後，會回答 NMT 僕端目前所處的狀態，像是停止、操作或著預操作等狀態。節點生存時間(Node Life Time)爲節點守衛時間乘上生存時間係數(life time factor)。不同的 NMT 僕端，其生存時間係數同樣可以作不同的設定。如果 NMT 僕端在其節點生存時間內沒有收到 NMT 主端的詢問，那 NMT 僕端就會發生生存守衛事件(Life Guarding Event)，這也就代表著遠端節點錯誤(remote node error)的產生。

NMT 僕端裝置的物件字典內，主索引爲 0x6206 和 0x6443 的物件項目可用來控制 DO 和 AO 通道的錯誤模式(error mode)。當發生生存守衛事件時，若 0x6206 和 0x6443 的物件項目設定錯誤模式爲啓動(enable)，則 NMT 僕端裝置的 DO 和 AO 通道就會輸出記錄在主索引 x6207 和 0x6444 物件項目中的錯誤模式值(error mode value)，若 0x6206 和 0x6443 的物件項目設定錯誤模式爲抑制(disable)，則 NMT 僕端裝置的 DO 和 AO 通道就會保留原先的輸出值。更多有關 0x6206、0x6207、0x6443 和 0x6444 物件的資訊，請參照第 6 章。

心跳產生者協定 (Heartbeat Producer Protocol)

心跳協定的通訊模型依循著產生者與消費者(Master/Slave)的關係，使用者可以透過這個協定監測網路中 CANopen 節點的狀態。其通訊協定如下圖所示：



心跳協定是一種不需要遠端請求訊息(remote frames)的錯誤控制服務(Error Control Service)，一個心跳產生者會循環地產生心跳訊息，會有一個或多個心跳消費者(Heartbeat Consumer)接收這個訊號，而產生者與消費者之間的關係可以經由物件字典來做設定，心跳消費者在心跳消費時間(Heartbeat Consumer Time)內會保持心跳的接受，如果在心跳消費時間內沒有收到心跳的訊息，心跳消費者將會產生一個心跳事件。

3.6 EDS 檔案

EDS 檔案內記錄了 CANopen 僕端裝置的各種資訊，像是：

- 製造商的名稱、硬體的型號
- 軟硬體的版本
- EDS 檔案建立的日期
- 節點 ID，使用的鮑率
- 物件字典內，各物件的內容、用途與資料型態
- 溝通的方式
- …

當一個 CANopen 的僕端裝置新增到網路中時，CANopen 的主端可以依據 CANopen 僕端的 EDS 檔案，了解 CANopen 僕端的各種特性，像是裝置支援哪些功能，該如何與其進行溝通等等。

泓格科技提供了一套工具程式，可讓使用者動態建立 CAN-8123/CAN-8223/CAN-8423 的 EDS 檔案，有關此工具程式的使用方式，請參照第 4 章。

4 配置&入門指南

4.1 CAN 僕端工具程式概觀

CAN 僕端工具程式(Slave Utility)是爲了 CAN-8123/CAN-8223/CAN-8423 所設計的，其提供了以下的功能：

- 在不與裝置連線的情況下(離線模式)，由使用者手動選取安裝於 CAN-8123/CAN-8223/CAN-8423 上擴充插槽的 I-8000 和 I-87K 模組，以建立裝置的 EDS 檔案。

- 在與裝置連線的情況下(在線模式)，由工具程式掃描裝在 CAN-8423 上的 I-8000 和 I-87K 模組，並根據掃瞄結果建立 EDS 檔案。

- 在與裝置連線的情況下(在線模式)，設定 CAN-8423 上 I-8000 和 I-87K 模組的 AI/AO 通道之輸入輸出範圍。

- 顯示裝置上的重要資訊，像是 PDO 通訊參數，標準化的裝置物件以及在 CAN-8123/CAN-8223/CAN-8423 內由製造商所定義的特定物件的資訊。

註： CAN-8123/CAN-8223 僅支援離線模式。CAN-8423 則支援離線和在線兩種模式。

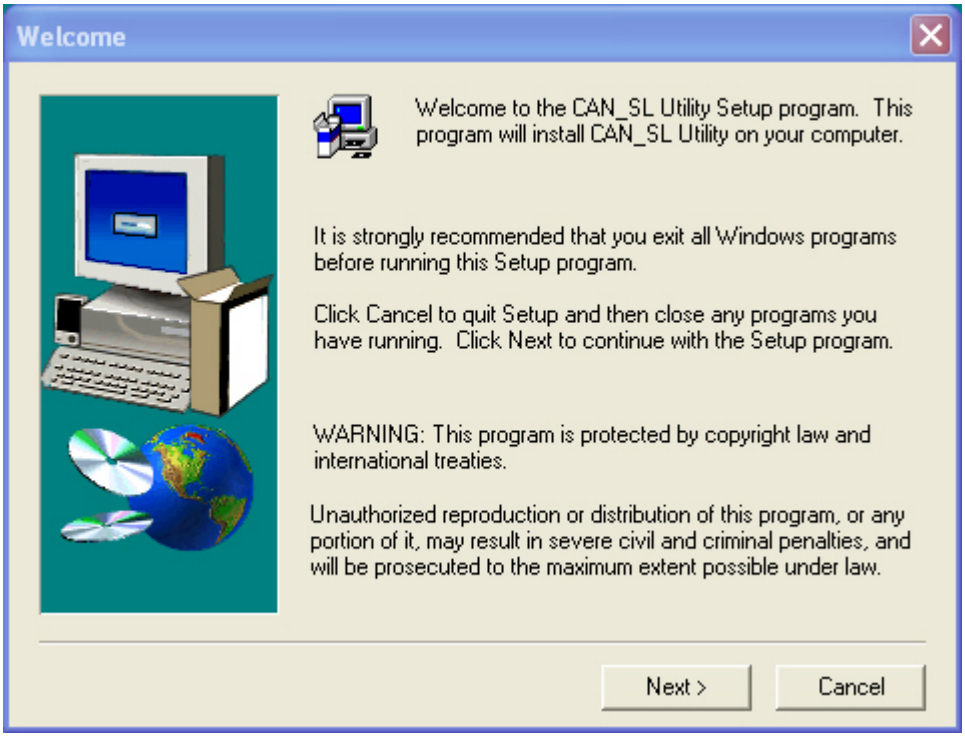
4.2 工具程式的安裝與移除

安裝 CAN 僕端工具程式

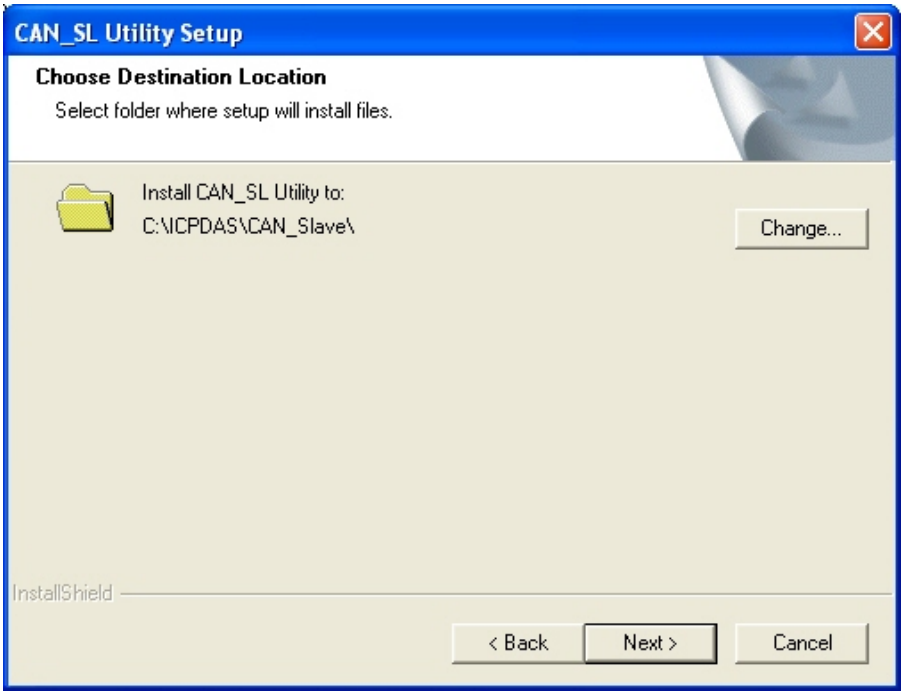
步驟 1：從底下的網址下載 CAN 僕端工具程式的安裝檔：
http://www.icpdas.com/products/Remote_IO/can_bus/can-8423.htm 或
http://www.icpdas.com/products/Remote_IO/can_bus/can-8123.htm
或著搜尋光碟片的路徑： \CANopen\Slave\CAN-8x23\Utility\

步驟 2：執行 CAN_SL_Setup.exe 藉以安裝 CAN 僕端工具程式。

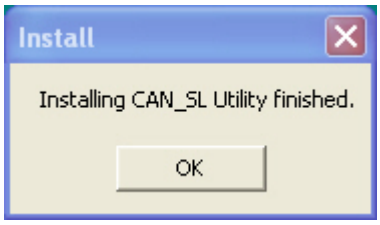
步驟 3：會出現 “Welcome” 視窗，表示即將開始進行安裝。



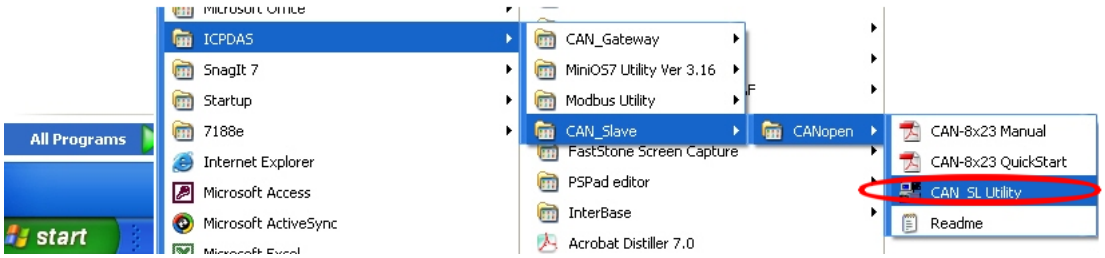
步驟 4：點選“Next”的按鈕，接著會跳出“Choose Destination Location”視窗，並要求選擇安裝路徑。



步驟 5：點選“Next”按鈕，接著便會開始安裝 CAN 僕端工具程式到系統之中。在安裝程序結束後，會出現一視窗提醒使用者 CAN 僕端工具程式已經安裝成功，如下圖。



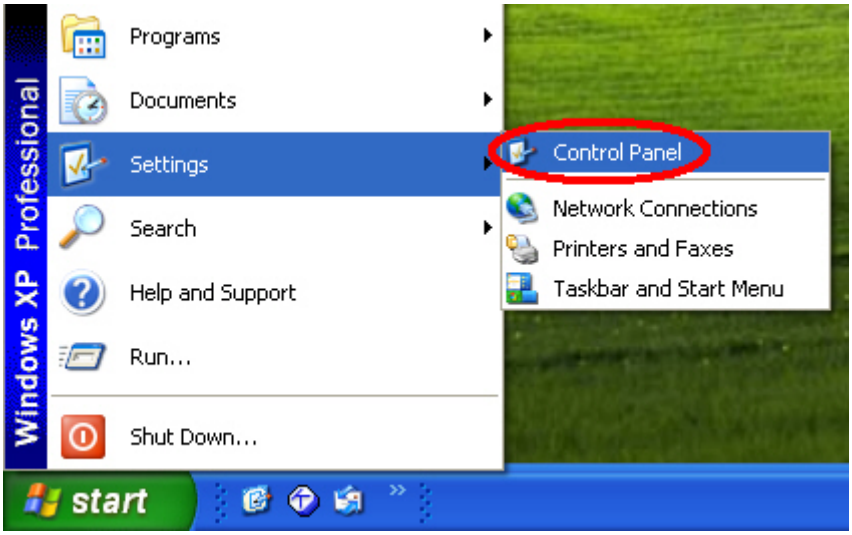
步驟 6：在安裝完 CAN 僕端工具程式之後，使用者便可以在程式集之中，找到 CAN 僕端工具程式的連結，如下圖：



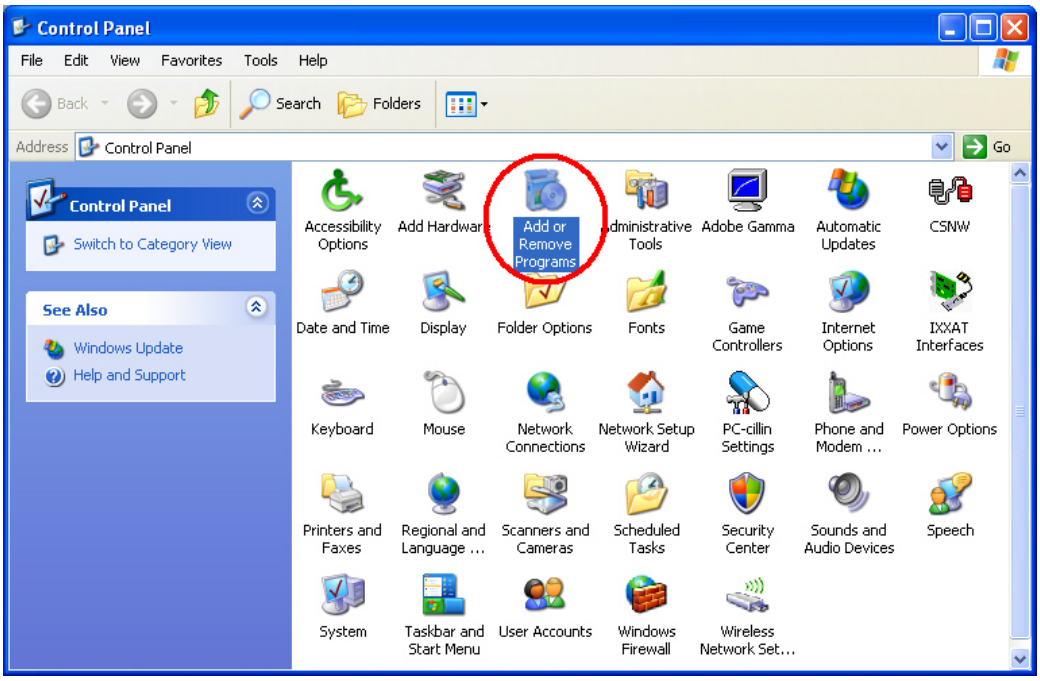
移除 CAN 僕端工具程式

使用者可以透過以下的方法移除 CAN 僕端工具程式。

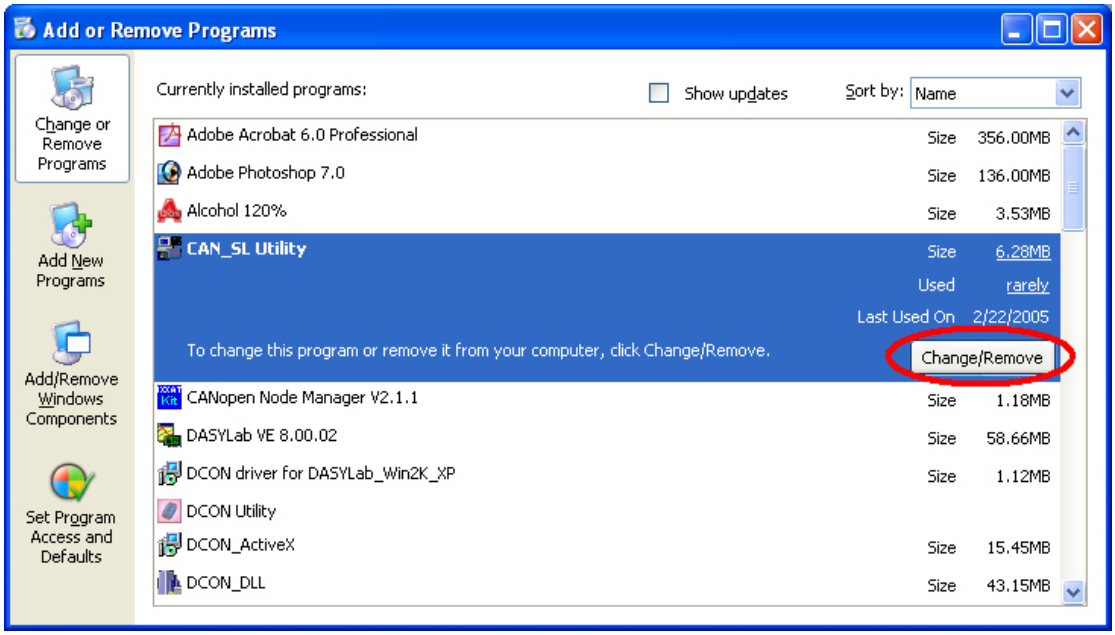
步驟 1：點選任務列上的 “Start” 按鈕，接著點選 “Setting” ，再點選 “Control Panel” ，如下圖所示：



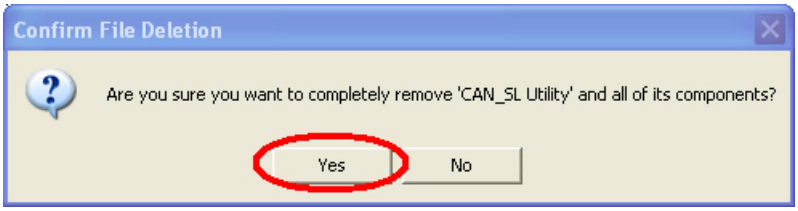
步驟 2：用滑鼠的左鍵雙擊 “Add/Remove” 圖示。



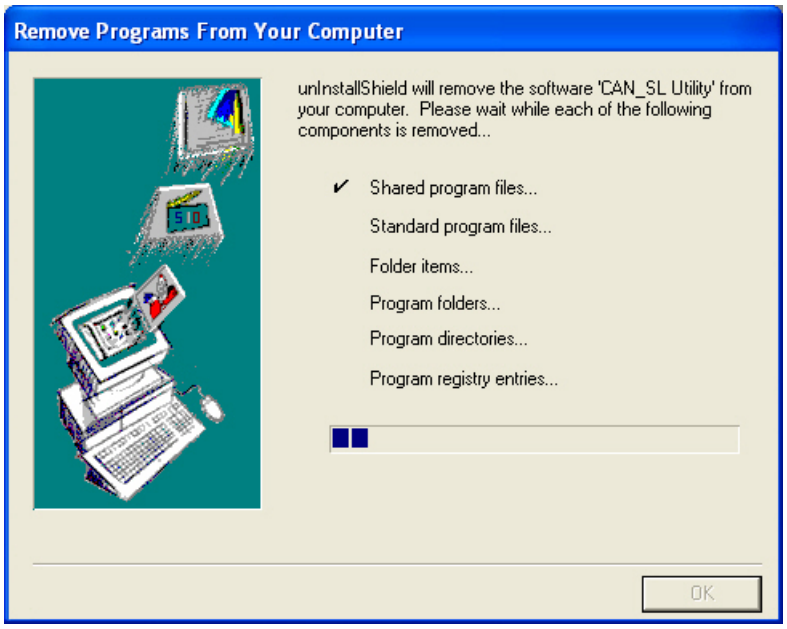
步驟 3：選取“CAN_SL Utility”的項目，並點選“Change/Remove”按鈕。



步驟 4：點選“Yes”按鈕，以移除 CAN 僕端工具程式。

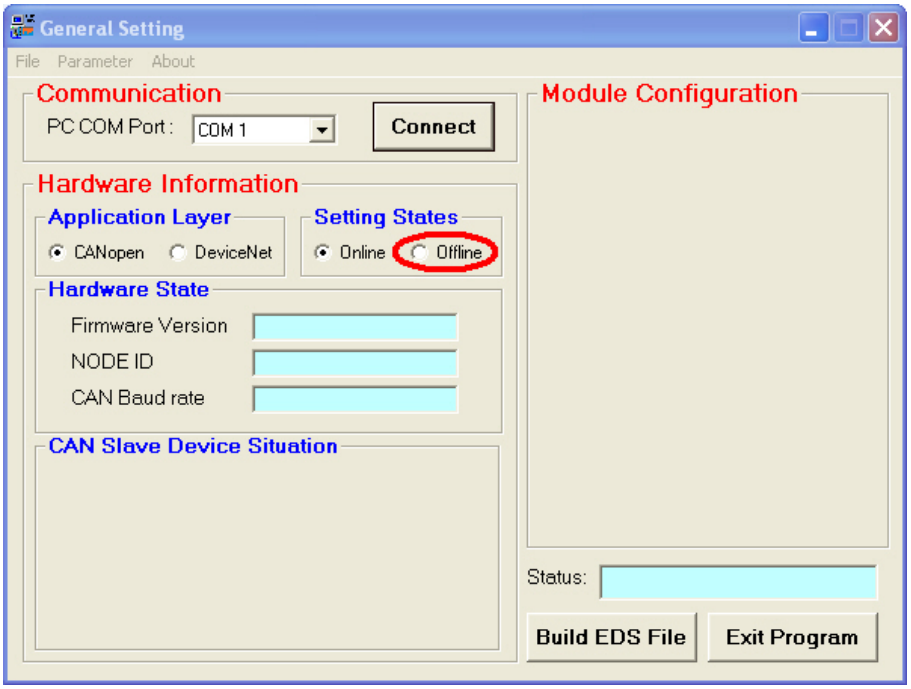


步驟 5：當移除完成時，請選取“OK”按鈕，以結束移除程序。



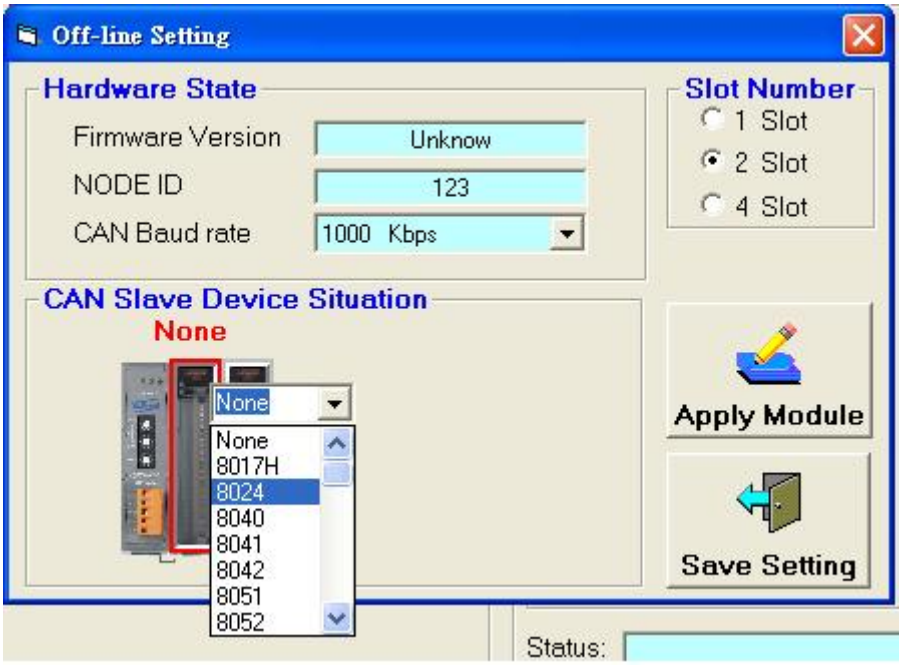
4.3 離線模式的配置方式

步驟 1：不需與裝置進行連線，逕行執行 CAN_SL Utility 即可，於此處設定“Application Layer”爲“CANopen”，“status”爲“offline”。

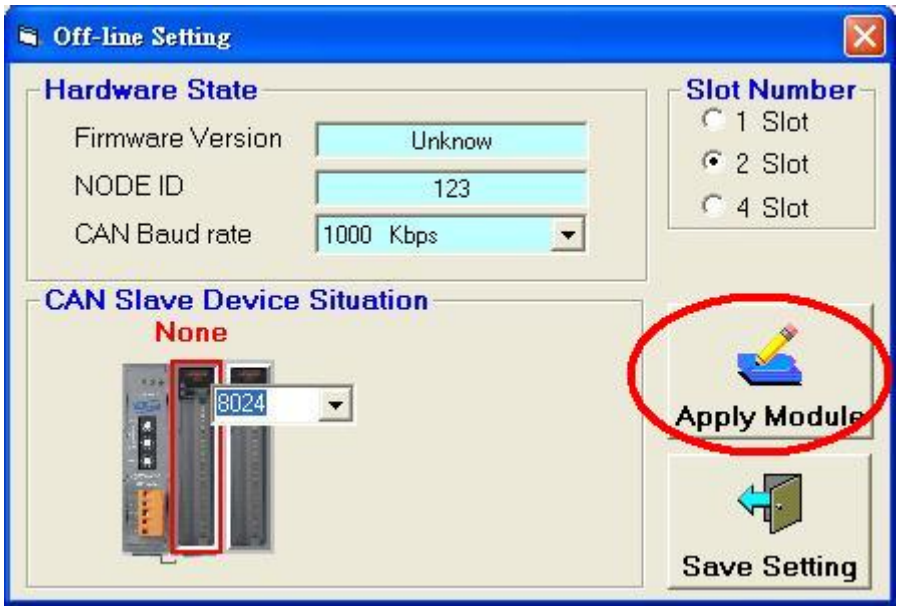


步驟 2：假設使用者所使用的 CAN 僕端裝置爲 CAN-8223，節點 ID 爲 123，鮑率爲 1000Kbps。於此處把相關的數值填到對應的欄位，接著選取“Slot Number 爲“2 Slot”。

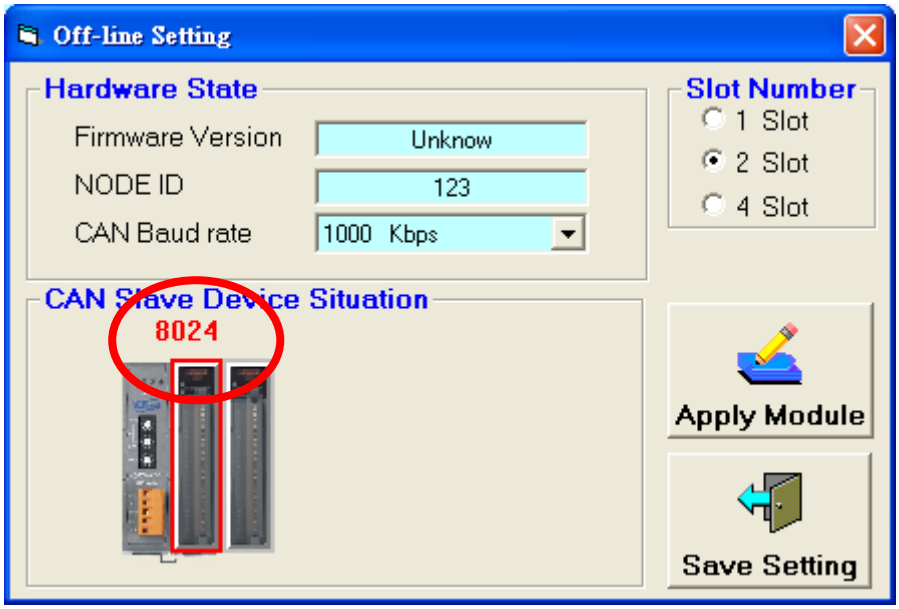
步驟 3：將滑鼠移動到“CAN Slave Device Situation”欄框內，依照實際狀況選擇擴充插槽圖示，並在跳出的捲軸中選取適當的模組。



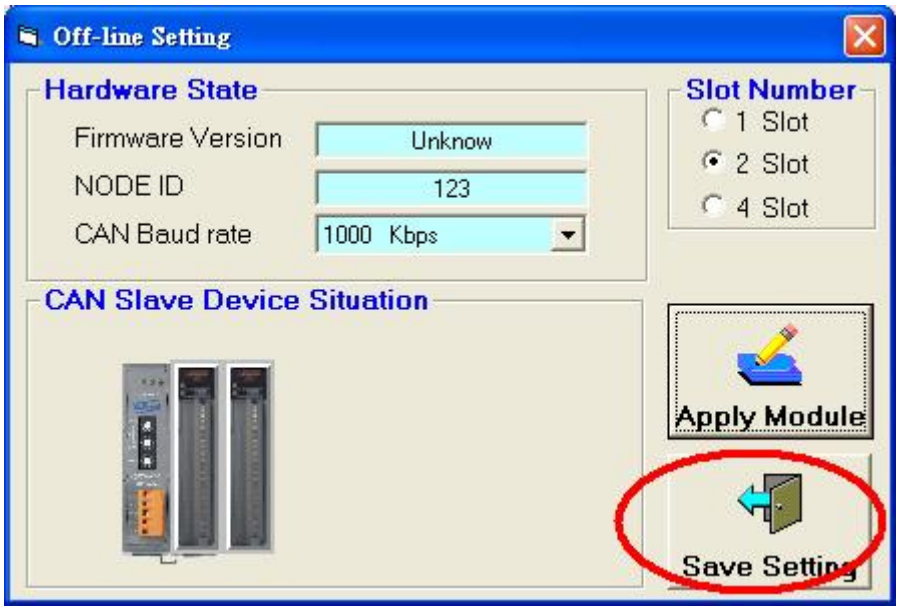
步驟 4：若擴充插槽 0 和擴充插槽 1 分別裝上了 I-8024 和 I-8042。此時，在左下方的圖示中，選取 8024，並按下“Apply Module”按鈕以儲存組態設定。



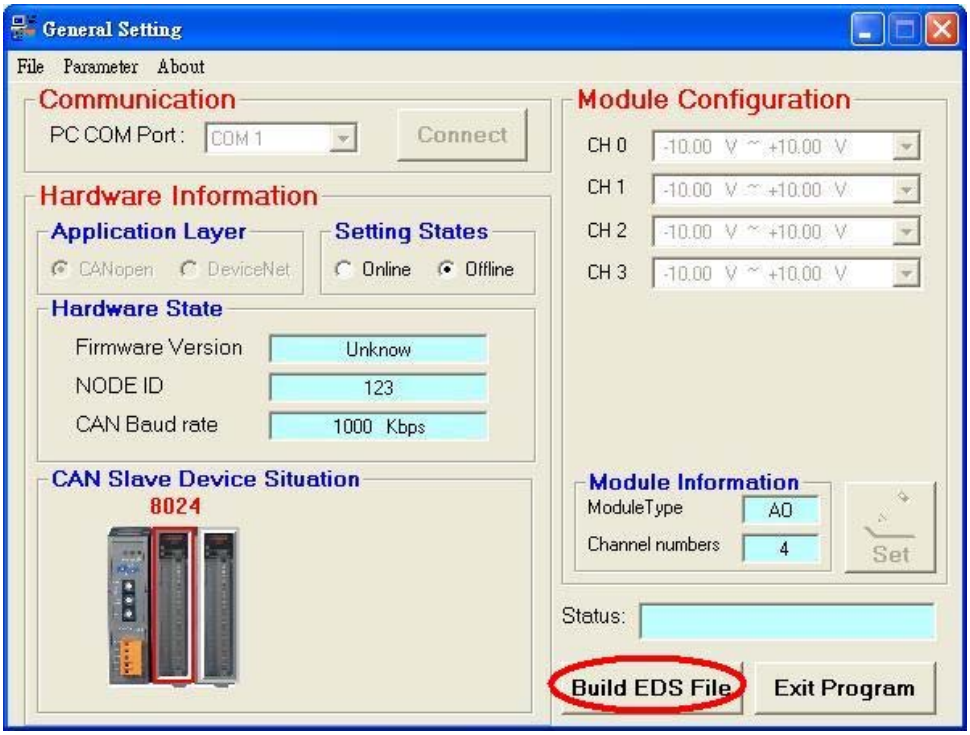
步驟 5：在結束組態之後，使用者可以將滑鼠游標移至“CAN Slave Device Situation”欄框內相對應的擴充插槽。若組態成功的話，使用者可以在擴充插槽的圖示上方看到正確的模組名稱。



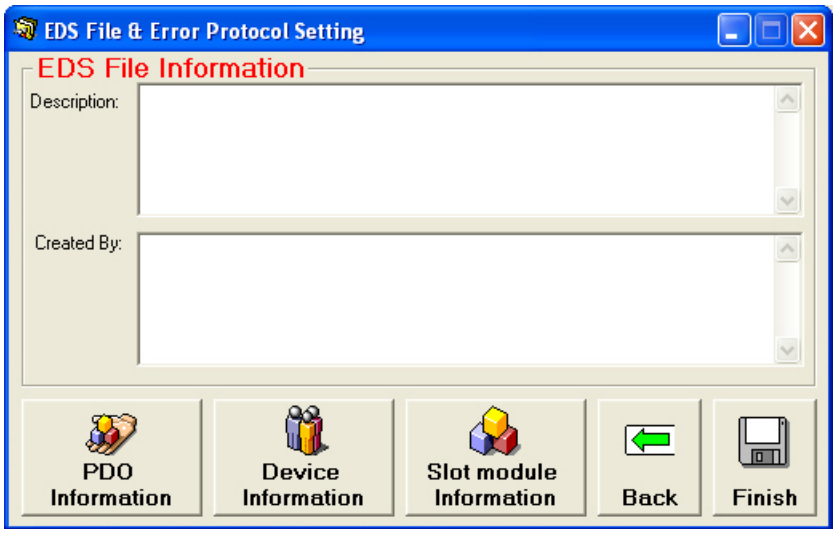
步驟 6：接著，針對擴充插槽 1 上面的 I-8042 模組，重複步驟 4~5。並且點選“Save Setting”按鈕以結束離線參數設定。



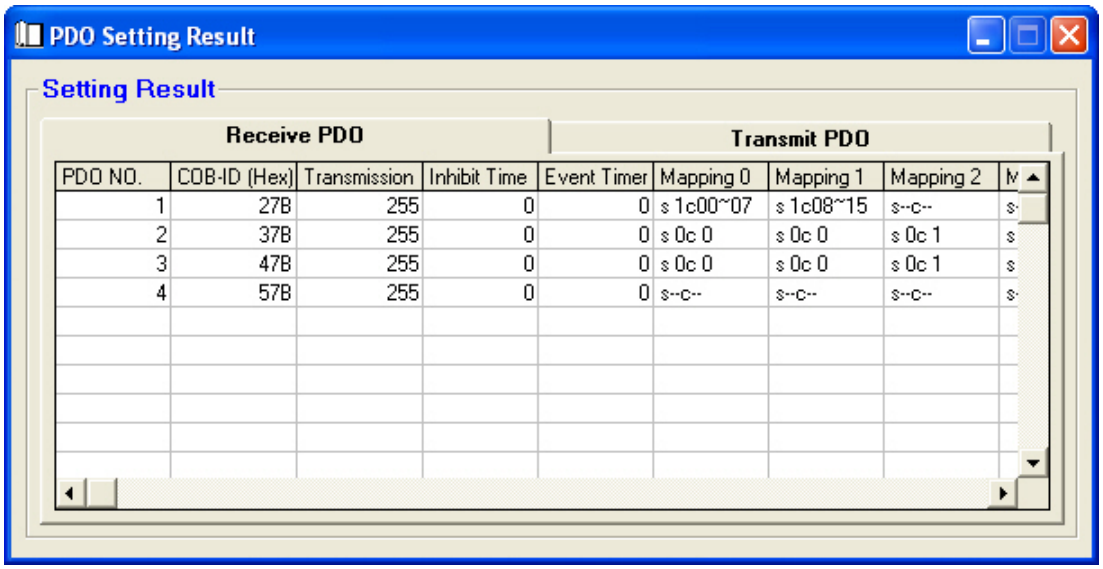
步驟 7：於“General Setting”視窗中，使用者可以點選個別擴充模組的圖示，檢視模組的名稱與“Module Information”欄框內的模組資訊。若點選的是類比模組，則還可以在“Module Congiguration”欄框內，檢視模組各通道的預設輸出入範圍。在確認所有設定後，點選“Build EDS File”按鈕以進入下一階段。

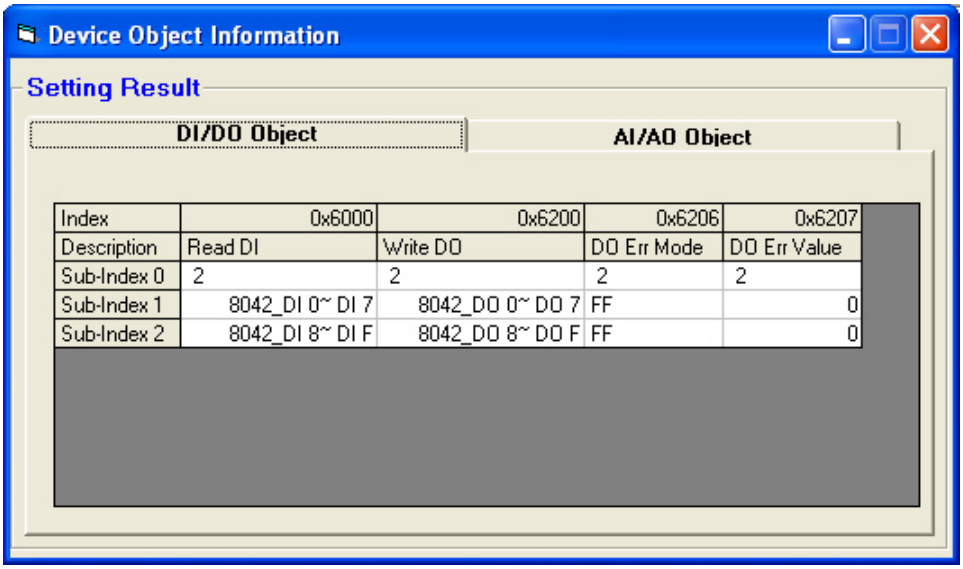


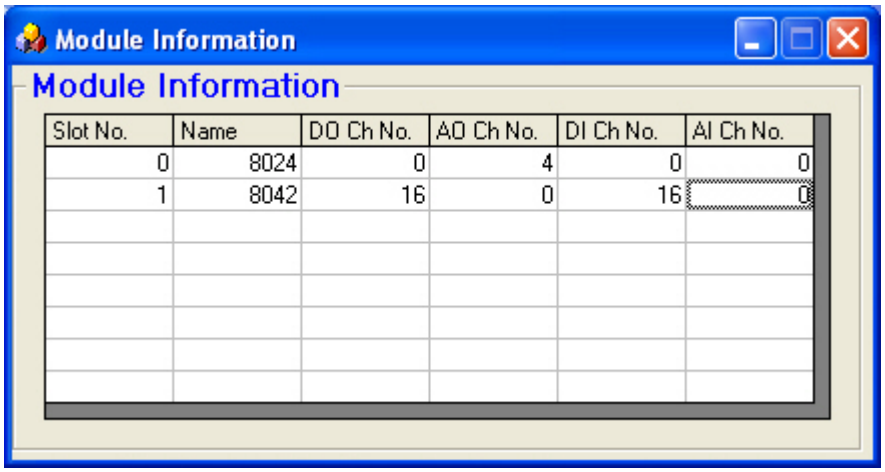
步驟 8：如果使用者需要在 EDS 檔案內做註記，可填在下圖中的“Description”和“Create By”欄位，如果使用者讓這兩個欄位保留空白，則在建立 EDS 檔案時，這兩個欄位的預設值就會分別爲“ICPDAS CANopen I/O Slave Device”和“ICPDAS”。



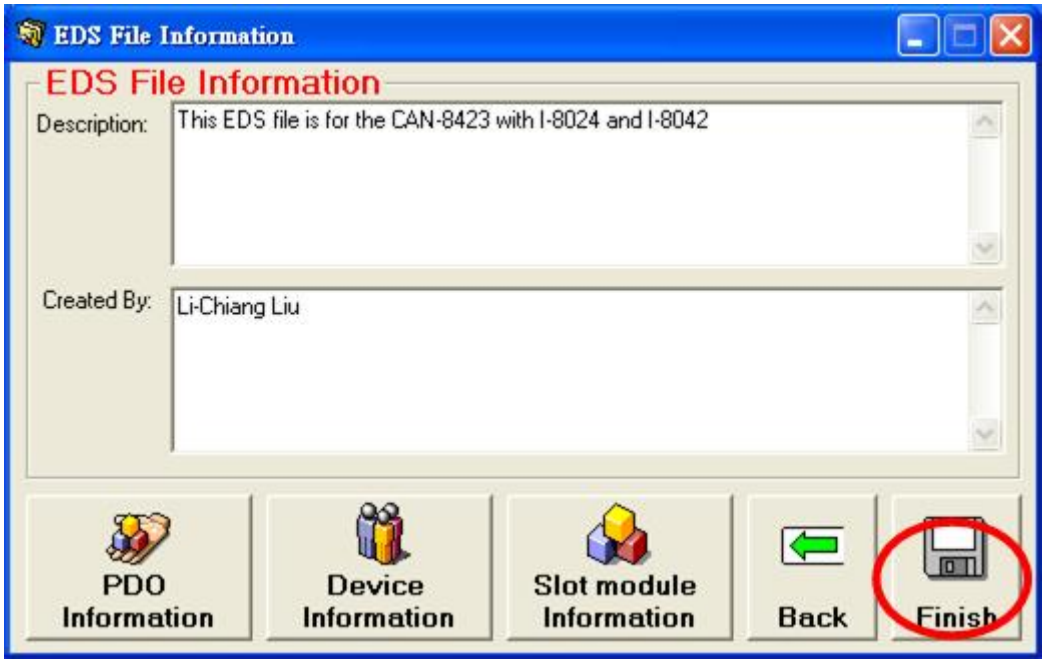
步驟 9：使用者可以分別點選“PDO Information”、“Device Information”和“Slot Module Information”的按鈕，以檢視相關資訊。操作畫面如下所示：







如果各項動作均完成且無誤，則可以點選“Finish”按鈕以建立 EDS 檔案。



註： 1. 於離線模式下建立 EDS 檔案時，畫面所顯示的類比模組之輸入/輸出範圍，是模組在出廠時的預設值。但是模組實際的輸入/輸出範圍乃是根據在裝置內儲存的設定來決定的，也就是 EDS 檔案內所記錄的輸入/輸出範圍，有可能和模組實際的輸入/輸出範圍不同。

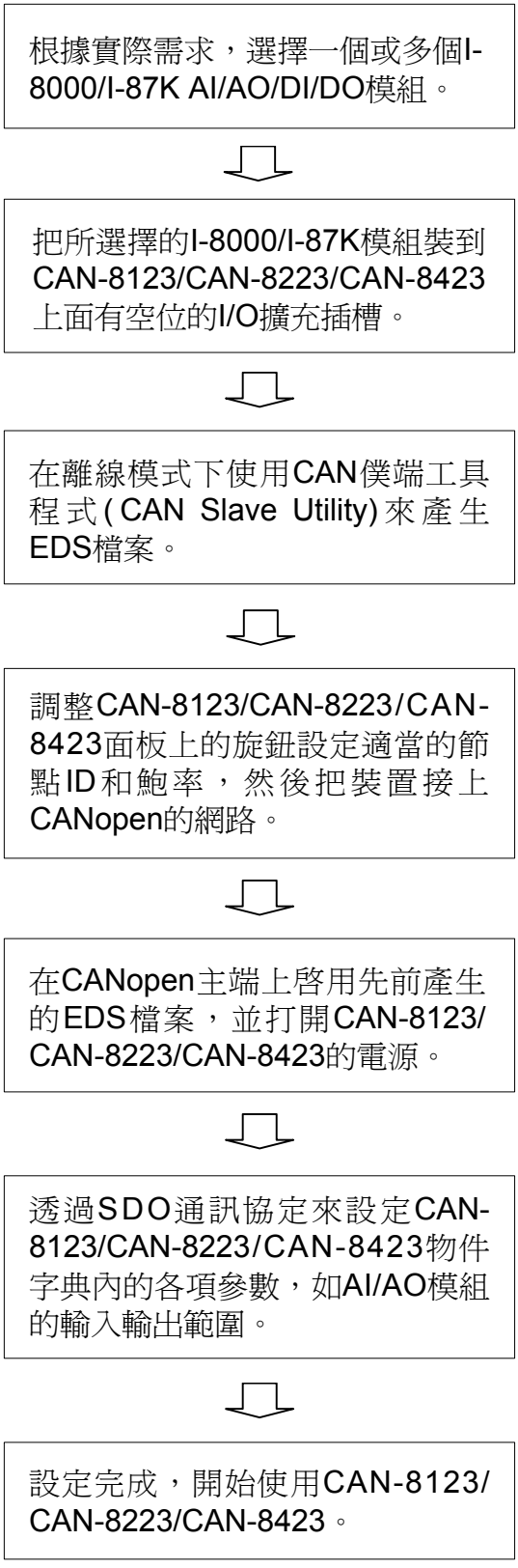
此外，使用者還可以利用 SDO 協定來調整裝置上 AI/AO 模組的輸入輸出範圍，以符合實際應用的需求。有關 AI/AO 模組的輸入輸出範圍，以及如何使用 SDO 協定對其進行修改，請參照 5.5 和 6.2 節。

2. EDS 檔案在建立完成之後，會放在和 CAN 僕端工具程式執行檔相同的目錄下，預設的路徑是“C:\ICPDAS\CAN_Slave\CANopen\”，EDS 檔案的檔案名稱格式如下所示：

COPn□.eds

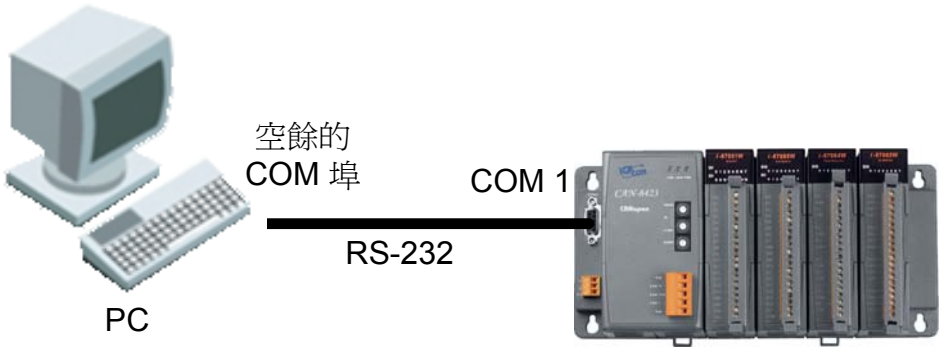
其中□爲在步驟 2 所設定的節點 ID，若節點 ID 被設定爲 12，則 EDS 檔案的名字就會是“COPn12.eds”。

4.4 離線模式配置流程圖

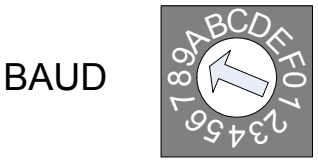


4.5 在線模式的配置方式

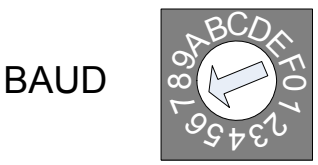
若要於在線模式下，使用 CAN 僕端工具程式與 CAN-8423 進行連線，則必須先確定 CAN-8423 端的 COM1 埠有連接到 PC 端空餘的 COM 埠，可參考下圖。我們於下圖中假設，CAN-8423 的擴充插槽 0、1、2、3 上面，分別裝上了 I-87018、I-87057、I-8024 和 I-87017 的擴充模組，節點 ID 設定爲 1，。



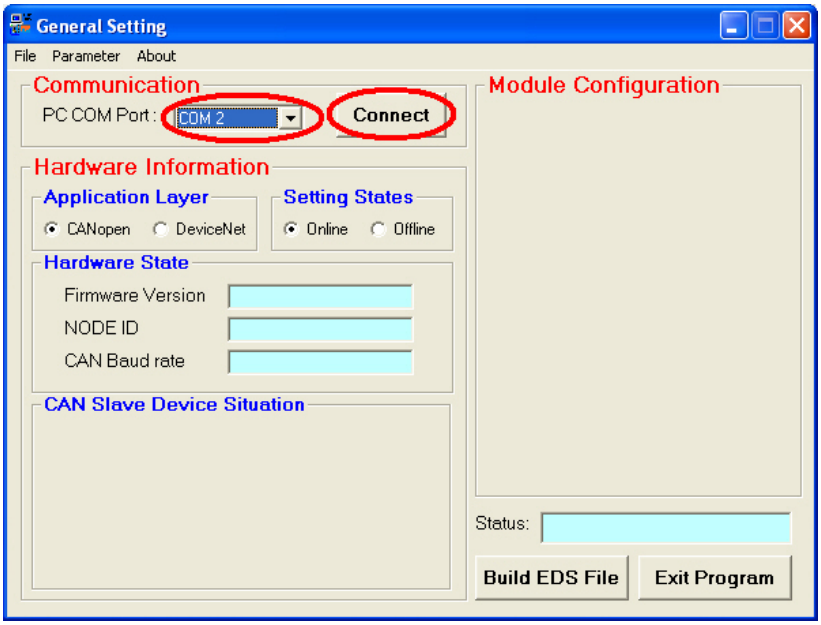
步驟1：將CAN-8423的電源關閉，接著調整CAN-8423的 “Baud” 旋鈕到 “9” 的位置，接著再打開CAN-8423。



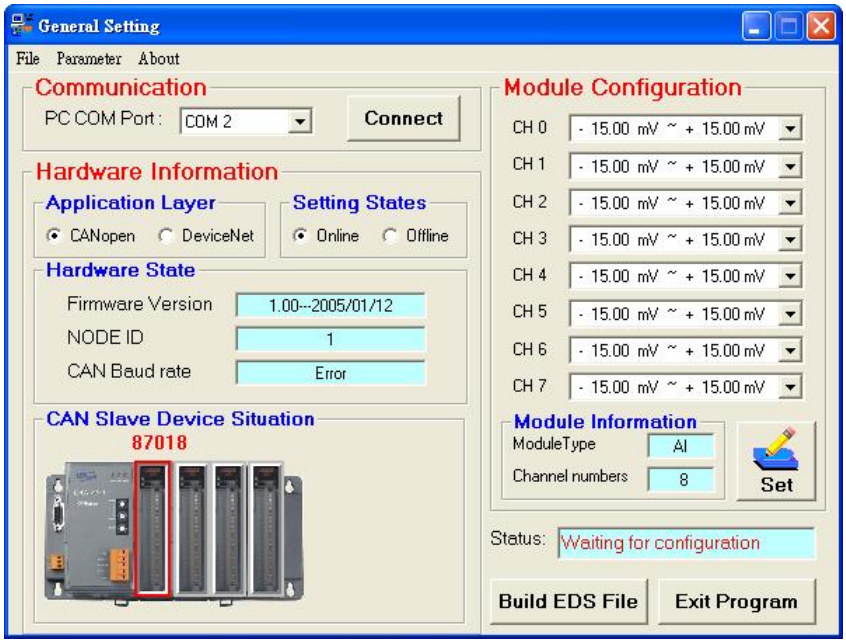
步驟2：於此處我們假設CAN-8423所使用的鮑率是1000Kbps，因此我們必需再將 “Baud” 的旋鈕調整到 “7” 的位置。



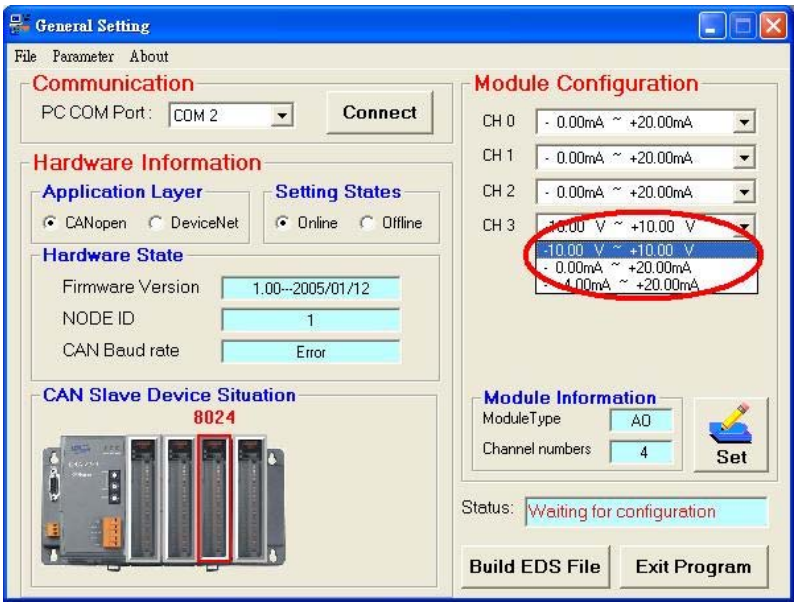
步驟 3：執行 PC 端上的 CAL_SL.exe，則會出現下圖。並依據實際狀況，在下圖中的“Communication”欄框內挑選連接 CAN-8423 的 PC 端 COM 埠，於此處我們假設使用 PC 端的 COM2 埠來連接 CAN-8423。接著點選“Connect”按鈕，以獲取儲存在 CAN-8423 內的資訊。



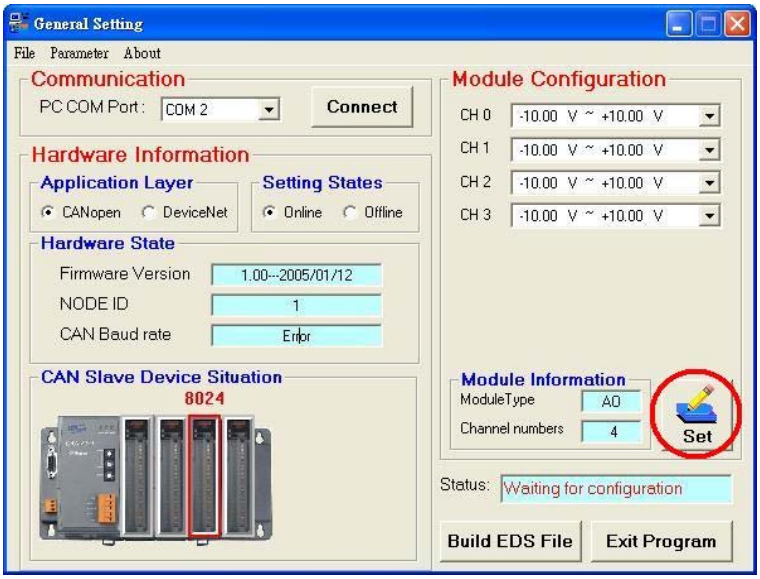
步驟 4：接著，使用者可以移動滑鼠的游標到“CAN Slave Device Situation”欄框中，CAN-8423 的擴充模組上，以獲取模組的相關資訊。



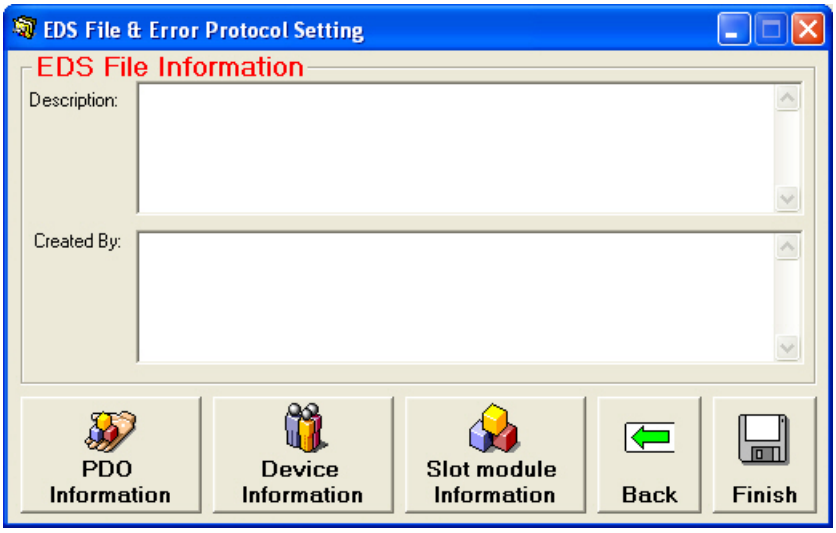
步驟 5：點選第 3 個擴充插槽上的擴充模組，並在“Module Configuration”欄框內選取適當的 AO 通道輸出範圍。於此處的示範中，我們選擇的範圍爲 -10.00V ~ +10.00V。另外，由於 I-8024 模組本身的特性，只要改變其中某一個通道的輸出範圍，其餘通道的輸出範圍也會跟著一起改變。



步驟 6：在選擇完輸出範圍之後，點選“Set”按鈕以儲存相關參數的設定。若已完成所有擴充模組的組態，點選“Build EDS File”按鈕以進入下一階段。

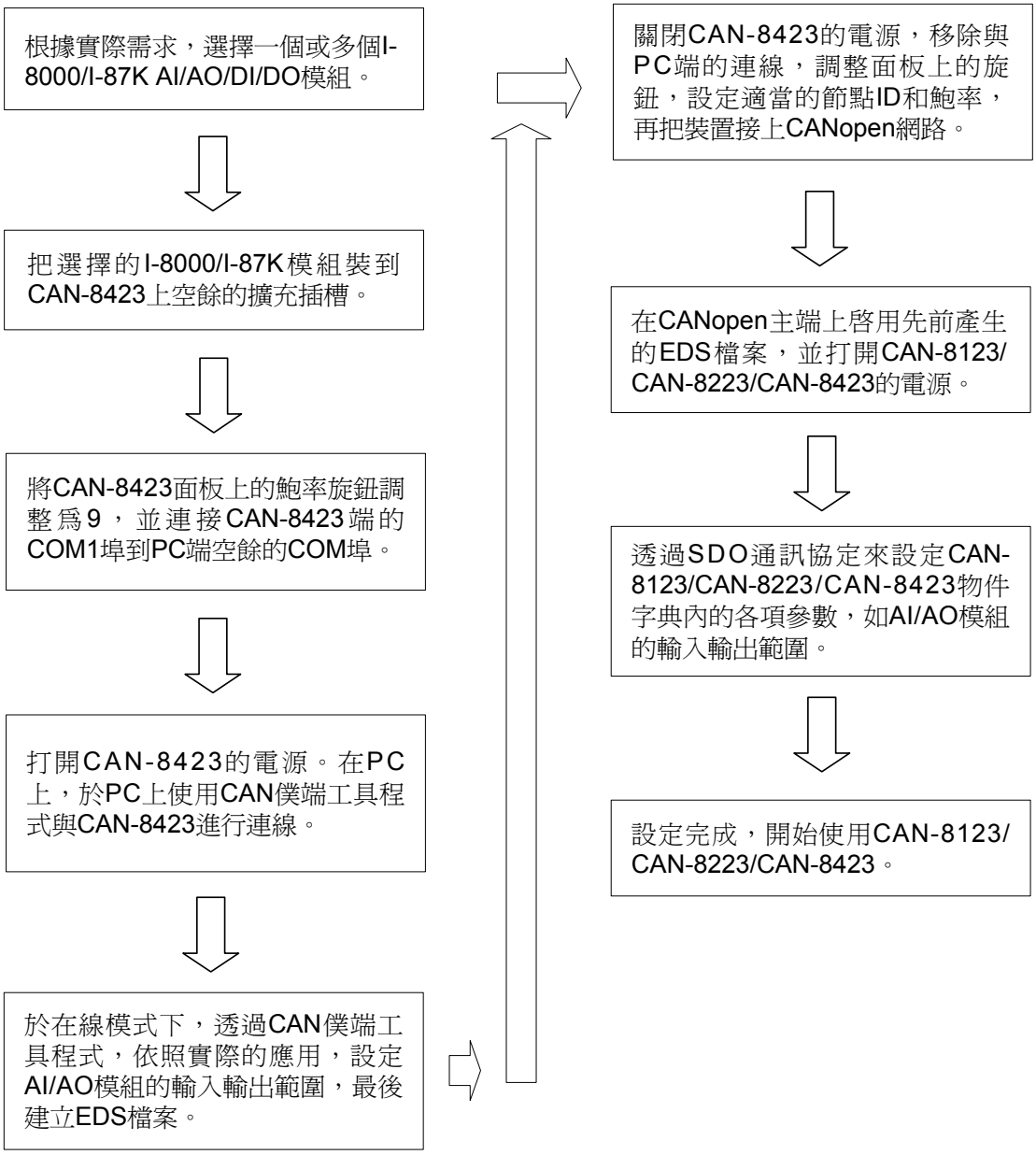


步驟 7：再來，使用者可以檢視 “EDS FIle Information” 視窗，並且在 “Description” 和 “Create By” 欄位中填入適當的資料。同時，使用者也可以點選視窗中的按鈕以檢視 CANopen 物件資訊和模組資訊，如 4.3 節的步驟 8 和步驟 9。



步驟 8：在確認設定沒有問題後，用者可以點選 “Finish” 按鈕以建立 EDS 檔案。值得一提的是，先前於步驟 5 所做的設定，在這個步驟完成之後才會寫入 CAN-8423 之中。

4.6 在線模式配置流程圖

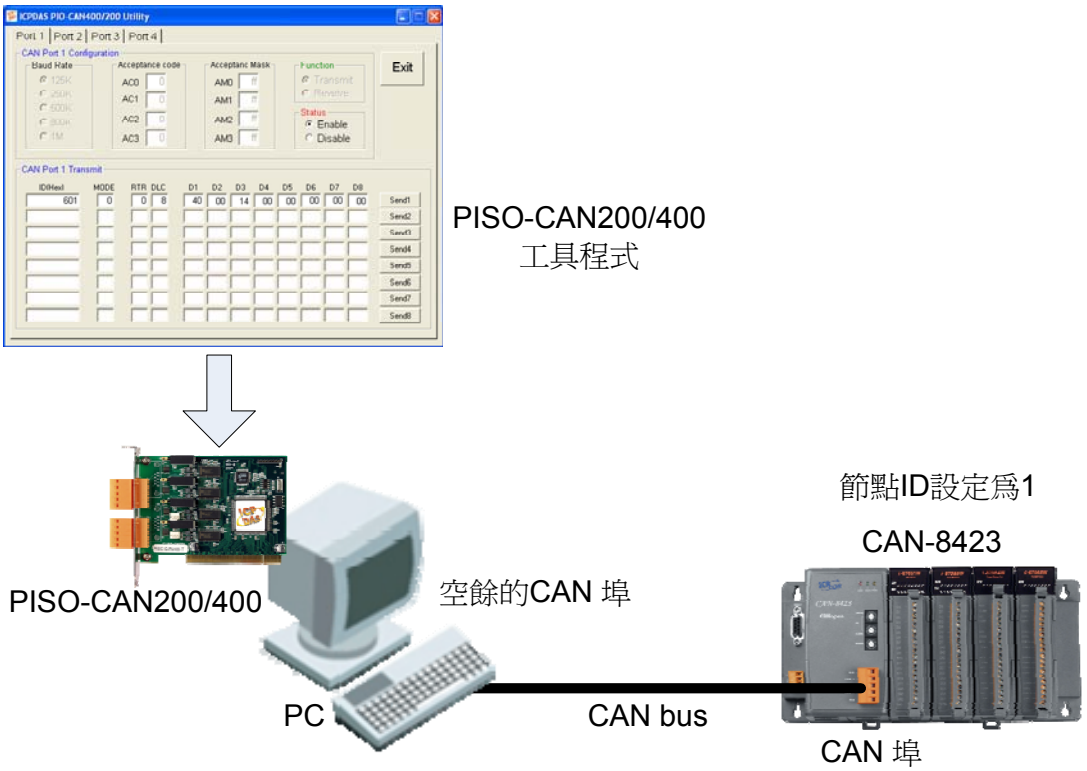


5 CANopen 通訊集

於這一章內將介紹數種 CANopen 的通訊協定(communication protocol)，並利用實例來示範如何使用這些協定。因爲 CAN-8123/CAN-8223 的通訊方式在使用上與 CAN-8423 十分類似，因此本章所提到的例子僅針對 CAN-8423。另外假設本章範例內的 CAN-8423，其節點 ID 被設定爲 1。

爲了實作這些例子，我們必須有一個 CAN 的介面，來和 CAN bus 上面的 CAN-8123/CAN-8223/CAN-8423 進行訊息的溝通，於此處，我們採用了型號爲 PISO-CAN200/400 的 CAN 介面卡。PISO-CAN200/400 是一個具有 2/4 CAN 埠的 PCI 介面卡，其提供了易於使用的工具程式，可在 PC 端透過 PISO-CAN200/400 對 CAN bus 上的裝置傳送或接收 CAN 2.0A/2.0B 的訊息。

本章內各範例所使用的的硬體架構如下所示：



註：有關 PISO-CAN200/400 工具程式的使用方式，請參考 PISO-CAN200/400 的使用者手冊。

5.1 SDO 通訊集

5.1.1 上傳 SDO 協定

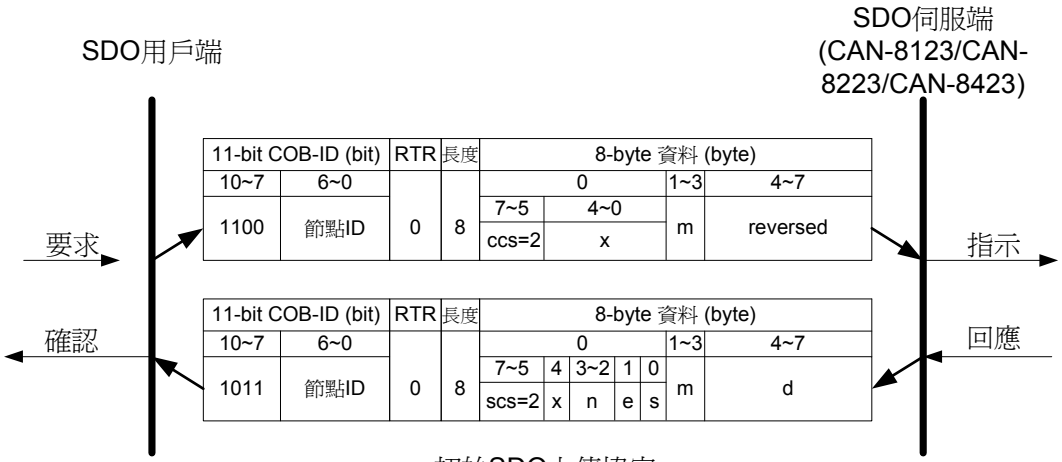
上傳 SDO 協定(Upload SDO Protocol)可被分爲兩個部份，包括初始 SDO 上傳協定，和上傳 SDO 區段協定。底下將針對這兩種協定進行描述。

初始 SDO 上傳協定 (Initiate SDO Upload Protocol)

在傳輸 SDO 區段(SDO segments)之前，用戶端(client)和伺服端(server)必須先利用初始 SDO 上傳協定來進行溝通。SDO 用戶端可以利用初始 SDO 上傳協定告訴 SDO 伺服端，其想要請 SDO 伺服端上傳的物件爲何。

另外，由於初始 SDO 上傳協定也可以同時夾帶 4 bytes 的資料進行傳輸。因此，如果 SDO 用戶端要求 SDO 伺服端上傳的物件，其資料長度小於或等於 4 bytes，則僅使用初始 SDO 上傳協定便可以完成資料的上傳，也就是不需進行上傳 SDO 區段協定(Upload SDO segment protocol)。

有關初始 SDO 上傳協定的細節，使用者可參考下列圖表：



初始SDO上傳協定
(Initiate SDO Upload Protocol)

ccs ：用戶端命令識別符(client command specifier)
2：初始上傳要求(initiate upload request)

scs ：伺服端命令識別符(server command specifier)
2：初始上傳回應(initiate upload response)

n ：只有當 **e**=1 且 **s**=1 此欄位才有意義，否則 **n**=0。
若此欄位有意義，則 **n** 代表 **d** 欄位內沒有資料的 byte 數目，即第 8-**n** 個 byte 到第 7 個 byte 內，沒有節段資料(segment data)。

e ：傳輸形式(transfer type)
0：正規傳輸(normal transfer)
1：附件傳輸(expedited transfer)
若 **e**=1，即代表欲傳輸物件的資料量小於或等於 4 bytes，僅需使用初始 SDO 上傳協定即可傳送完畢。
若 **e**=0，就必需要進行上傳 SDO 區段協定。

s ：資料量指示符(size indicator)
0：代表幀(frame)內沒有資料大小的資訊。
1：代表幀(frame)內有資料大小的資訊。

m ：多工器(multiplexor)
其代表欲傳輸 SDO 物件內含的資料，在物件字典的主索引和子索引。前兩個 byte 代表主索引，後一個 byte 代表子索引。

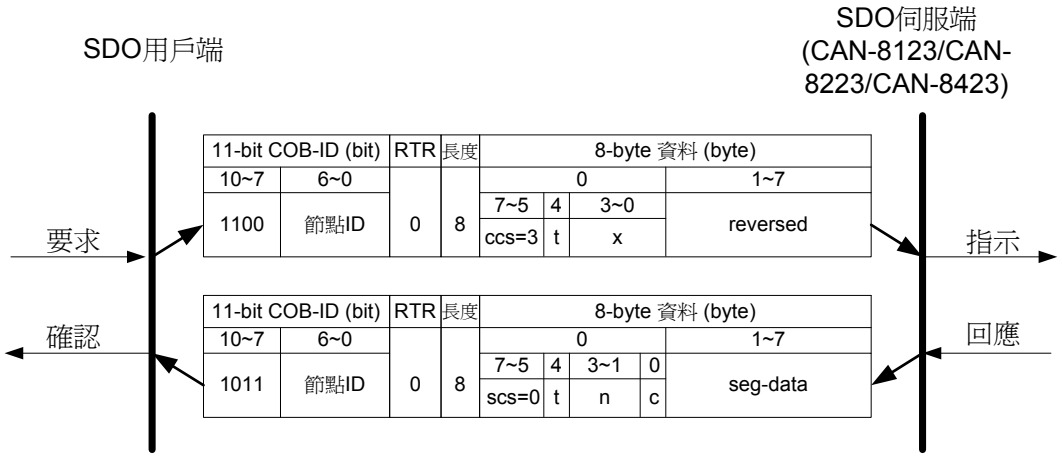
d ：資料
e=0，**s**=0：表示 **d** 被保留，留待進一步的使用。
e=0，**s**=1：**d** 包含了欲上傳 byte 的數目，其中 byte 4 內含最低有效位元(least significant bit)，byte 7 內含最高有效位元(most significant bit)。
e=1，**s**=1：**d** 的內容爲欲上傳的資料，其長度爲 4-**n**。資料編碼的方式取決於主索引和子索引在物件字典中參照項目的資料型別。
e=1，**s**=0：**d** 的內容爲未標示長度的上傳資料。

x ：沒有被使用，數值永爲 0。

reserved ：保留，留待進一步的使用，數值永爲 0。

上傳 SDO 區段協定 (Upload SDO Segment Protocol)

當欲上傳的資料大小超過 4 bytes，除了初始 SDO 上傳協定，此時還需要透過上傳 SDO 區段協定來進行資料的上傳。當初始 SDO 上傳協定結束之後，SDO 用戶端便開始利用上傳 SDO 區段協定，請 SDO 伺服端上傳資料。有關上傳 SDO 區段協定的細節如下所示：



上傳SDO區段協定
(Upload SDO Segment Protocol)

ccs ：用戶端命令識別符(client command specifier)
3：上傳節段要求(upload segment request)

scs ：伺服端命令識別符(server command specifier)
0：上傳節段回應(upload segment response)

t ：交替位元(toggle bit)
對每一連續的上傳節段而言，每一個節段的交替位元均需與其上一個或下一個節段的交替位元不同。而第 1 個節段的交替位元必須設定爲 0，另外要求訊息和回應訊息的交替位元必須相同。

c ：用來指出是否還有節段要被上傳。
0：還有節段等待上傳。
1：已經沒有節段需要上傳。

seg-data ：其內爲欲上傳的節段資料，一次最多可上傳 7 bytes。資料編碼的方式取決於初始 SDO 上傳協定內，主索引和子索引在物件字典中參照項目的資料型別。

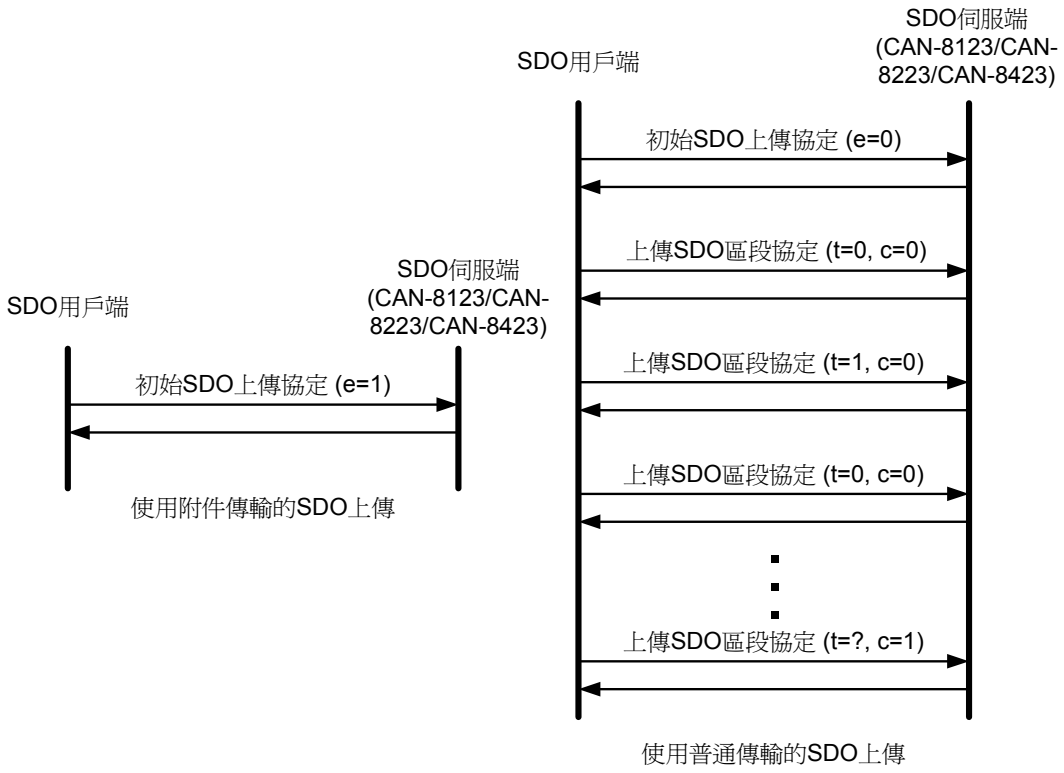
n ：**n** 內含 **seg-data** 欄位內沒有節段資料的 byte 數目，即第 8-**n** 個 byte 到第 7 個 byte 內，沒有節段資料(segment data)。如果 **n**=0，代表節段的大小沒有被指示。

x ：沒有被使用，數值永爲 0。

reserved ：保留，留待進一步的使用，數值永爲 0。

SDO 上傳範例

下圖爲實際利用上傳 SDO 協定上傳資料的過程：



底下將根據上圖，用實際的範例對附件傳輸(expedited transfer)和正規傳輸(normal transfer)的細節做更深入的描述。同時也會示範如何取得儲存在物件字典內的資料。

舉例來說，若使用者想要知道物件字典中主索引爲 0x1400 的物件其子索引數目，則使用者可以透過初始 SDO 上傳協定，要求 SDO 伺服端上傳主索引爲 0x1400 且子索引爲 0x00 的物件項目。若使用者想要知道製造商裝置名稱(manufacturer device name)，則可以透過初始 SDO 上傳協定和上傳 SDO 區段協定，由 SDO 伺服端上傳主索引爲 1008 的物件給使用者。

- **附件傳輸 (expedited transfer)的範例**

步驟 1：傳送如下的 SDO 訊息給 CAN-8423，以取得在物件字典的通訊描述文件區域內，主索引爲 0x1400 且子索引爲 0x00 的項目。有關此項目的作用，使用者可以參照第 6 章。

初始 SDO 上傳協定 第 67 頁

11-bit COB-ID (601)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	40	00	14	00	00	00	00	00

SDO 用戶端 → **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 2

m ： 00 14 00

因爲低字組會先被傳送，所以第 1 個 byte“00”代表 0x1400 的低字組，而第 2 個 byte“14”代表 0x1400 的高字組，至於最後一個 byte“00”則代表子索引 0x00。

步驟 2：CAN-8423 將會回應主索引爲 0x1400 且子索引爲 0x00 的物件項目內所儲存的資料。

初始 SDO 上傳協定 第 67 頁

11-bit COB-ID (581)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	4F	00	14	00	02	00	00	00

SDO 用戶端 ← **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 2

n ： 3

e ： 1

s ： 1

m ： 00 14 00

d ： 02 00 00 00

因爲 **n=3**，所以只有第 4 個 byte 會有資料，於此處其值爲 0x02。

- **正規傳輸 (normal transfer)範例**

步驟 1：傳送 SDO 訊息給 CAN-8423(對 CAN-8423 而言，此 SDO 訊息爲 RxSDO)，以取得儲存在物件字典的通訊描述文件區域中，主索引爲 0x1008，子索引爲 0x00 的物件項目。關於主索引爲 0x1008 的物件，其內容將於第 6 章再行介紹。

初始 SDO 上傳協定 第 67 頁

11-bit COB-ID (601)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	40	08	10	00	00	00	00	00

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 2
m ： 08 10 00

步驟 2：CAN-8423 會回應 SDO 訊息以告知使用者，CAN-8423 預計上傳給使用者的資料量(byte)。

初始 SDO 上傳協定 第 67 頁

11-bit COB-ID (581)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	41	08	10	00	09	00	00	00

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 2
n ： 0
e ： 0
s ： 1
m ： 08 10 00
d ： 09 00 00 00
因爲 **e**=0 且 **s**=1，所以 **d** 代表 CAN-8423 預計上傳給使用者的資料量(byte)。而“09”代表資料長度中的最低的字組，因此“09 00 00 00”這 4 個 byte 就代表 CAN-8423 將上傳 9 個 bytes 給使用者。

步驟 3：於此處傳送以下訊息，要求 CAN-8423 開始傳送資料。

上傳 SDO 節段協定 第 69 頁

11-bit COB-ID (601)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	60	00	00	00	00	00	00	00

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 3
t ： 0

步驟 4：CAN-8423 會回應主索引 0x1008，子索引 0x00 的項目內，第 0 個 byte 到第 7 個 byte。

上傳 SDO 節段協定 第 69 頁

11-bit COB-ID (581)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	00	43	41	4E	2D	38	34	32

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 0
t ： 0
n ： 0
c ： 0
seg-data ： 43 41 4E 2D 38 34 32

使用者若是查閱第 6 章，可發現主索引 0x1008，子索引 0x00 的項目，其資料型態爲“VISIBLE_STRING”。因此，使用者在接收到這個物件之後，可根據 ASCII 表來對 **seg-data** 內的資料做轉換，此處轉換之後，這 7 個字元分別爲“CAN-842”。

步驟 5：於此處傳送以下訊息，要求 CAN-8423 傳送剩餘的資料。

上傳 SDO 節段協定 第 69 頁

11-bit COB-ID (601)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	70	00	00	00	00	00	00	00

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 3
t ： 1

步驟 6：接收由 SDO 伺服端(CAN-8423)傳送來的剩餘資料。

上傳 SDO 節段協定 第 69 頁

11-bit COB-ID (581)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	1B	33	00	00	00	00	00	00

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 0
t ： 1
n ： 5
c ： 1
seg-data ： 33 00 00 00 00 00 00

因爲 **n=5**，也就是說在 **data** 欄位內，只有前 3 個 bytes 的資料有意義，而 0x1B 爲上傳 SDO 節段協定的標頭，扣掉這個 byte，剩下的 0x33 和 0x00 才是資料。對照 ASCII 表，0x33 和 0x00 分別代表字元“3”和字元“Null”。

5.1.2 SDO 區塊上傳協定

SDO 區塊上傳協定(SDO Block Upload Protocol)可分爲三個部份，分別爲初始 SDO 區塊上傳協定、SDO 區塊上傳協定以及終止 SDO 區塊上傳協定。底下將詳細描述這三個協定的架構。

初始 SDO 區塊上傳協定 (Initiate SDO Block Upload Protocol)

SDO 區塊上傳協定通常被用來作大量的資料傳輸。在進行 SDO 區塊上傳協定時，首先必須要進行初始 SDO 區塊上傳協定。初始 SDO 區塊上傳協定的細節如下所示：

初始SDO區塊上傳協定
(Initiate SDO Block Upload Protocol)

ccs ：用戶端命令識別符(client command specifier)
5：區塊上傳(block upload)

scs ：伺服端命令識別符(server command specifier)
6：區塊上傳(block upload)

cs ：用戶端子指令(client subcommand)
0：初始上傳要求(initiate upload request)
3：開始上傳(start upload)

ss ：伺服端子指令(server subcommand)
0：初始上傳回應(initiate upload response)

m ：多工器(multiplexor)
其內含 SDO 所傳輸的資料，其主索引和子索引。

cc ：用戶端循環冗餘檢查支援(client CRC support)
cc=0：用戶端不支援對資料產生 CRC。
cc=1：用戶端支援對資料產生 CRC。.

sc ：伺服端循環冗餘檢查支援(client CRC support)
sc=0：伺服端不支援對資料產生 CRC。
sc=1：伺服端支援對資料產生 CRC。

pst ：協定切換臨界值(Protocol Switch Threshold)，單位爲 byte，其用來決定是否要切換爲 SDO 傳輸協定(SDO transfer protocol)。
pst=0：不允許進行傳輸協定的切換。
pst>0：若欲上傳的資料量(byte)小於或等於 pst，則伺服端可以改傳送上傳 SDO 協定的伺服端回應訊息給 SDO 用戶端，通知 SDO 用戶端，欲上傳的資料將改用上傳 SDO 協定來傳送。

s ：資料量指示符(size indicator)
0：代表幀(frame)內沒有資料集(Data Set)大小的資訊。
1：代表幀(frame)內有資料集(Data Set)大小的資訊。

size ：欲上傳的資料量大小(byte)
s=0：**size** 欄位被保留，留待進一步的使用，數值永爲 0。
s=1：**size** 欄位內包含了欲上傳 byte 的數目，其中 byte 4 內含最低有效位元(least significant bit)，byte 7 內含最高有效位元(most significant bit)。

blksize ：每一個區塊內包含的區段數目(0 < **blksize** < 128)。

x ：沒有被使用，數值永爲 0。

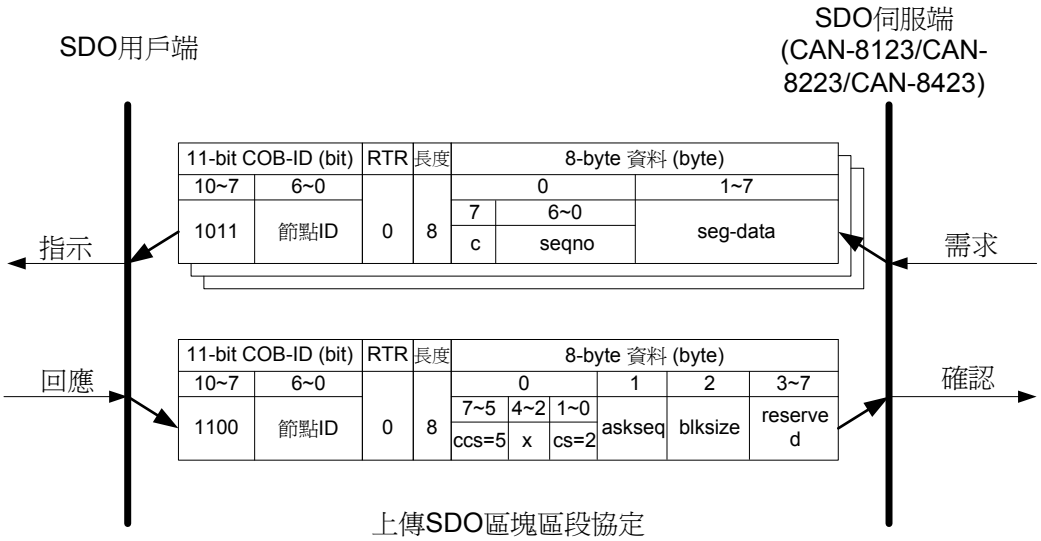
reserved ：保留，留待進一步的使用，數值永爲 0。

上傳 SDO 區塊區段協定 (Upload SDO Block Segment Protocol)

在初始 SDO 區塊上傳協定結束後，SDO 伺服端便開始透過上傳 SDO 區塊區段協定來回覆資料。

每一個區塊(block)至少要有一個區段(segment)，最多可內含 127 個區段。一個區段內的資料量可以爲 1 byte ~ 7 bytes。進行一次上傳 SDO 區塊區段協定僅能傳送一個 Block。

上傳 SDO 區塊區段協定的細節如下所示：



上傳SDO區塊區段協定
(Upload SDO Block Segment Protocol)

ccs ：用戶端命令識別符(client command specifier)
5：區塊上傳(block upload)

cs ：用戶端子指令(client subcommand)
2：區塊上傳回應(block upload response)

c ：此欄位用來指出是否仍有區段等待被上傳。
0：尚有區段需要被上傳。
1：已經沒有區段需要被上傳，準備執行”終止 SDO 區塊上傳” (End SDO block upload)協定。

seqno ：區段序號(0 < **seqno** < 128)

seg-data ：一個區段最多可以內含 7 個 bytes 的資料。

ackseq ：註記最近一次的區塊上傳內，最後一個成功接收區段的序號。若 **ackseq** 被設定爲 0，則表示從編號爲 1 的區段開始，用戶端就沒有接收到。此時伺服端必需要重送所有的區段。

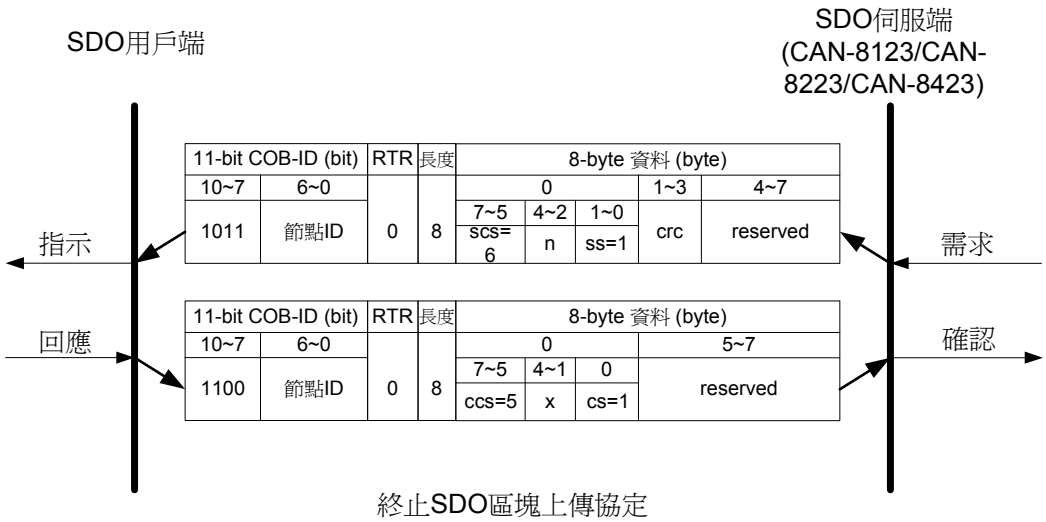
blksize ：用來讓用戶端告知伺服端，在下一個要傳送的區塊內，應擺放的區段個數。(0 < **blksize** < 128)

x ：沒有被使用，數值永爲 0。

reserved ：保留，留待進一步的使用，數值永爲 0。

終止 SDO 區塊上傳協定 (End SDO Block Upload Protocol)

終止 SDO 區塊上傳協定乃是用來結束 SDO 區塊上傳，此協定的細節如下所示：



終止SDO區塊上傳協定
(End SDO Block Upload Protocol)

ccs ：用戶端命令識別符(client command specifier)
5：區塊上傳(block upload)

scs ：伺服端命令識別符(server command specifier)
6：區塊上傳(block upload)

cs ：用戶端子指令(client subcommand)
1：終止區塊上傳要求(end block upload request)

ss ：伺服端子指令(server subcommand)
1：終止區塊上傳回應(end block upload response)

n ：用來註記最後一次傳輸的區塊內的最後一個區段，沒有內含資料的 byte 數目。其中，未包含區段資料的部份爲第 8-**n** 個 byte 到第 7 個 byte。

crc ：對所有資料集合(data set)所做的 16bit 循環冗餘檢查(CRC，Cyclic Redundancy Checksum)碼。

用來計算 CRC 的多項式： $x^{16}+x^{12}+x^5+1$

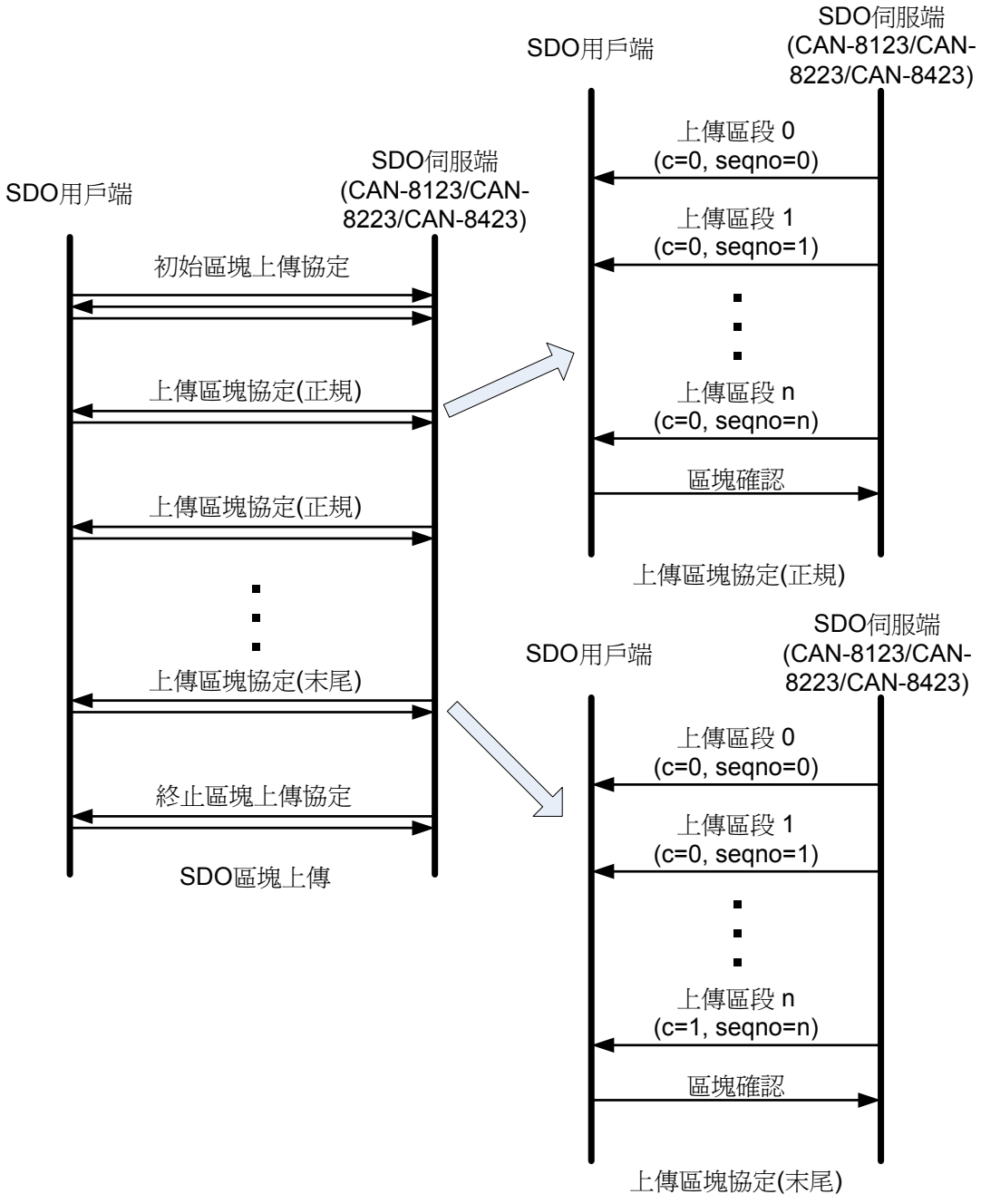
只有在進行初始 SDO 區塊上傳協定時，**cc** 和 **sc** 都被設定成 1，**crc** 欄位才會被啓用，否則的話 **crc** 欄位的值就會被設定爲 0。而 CAN-8123/CAN-8223/CAN-8423 目前並不支援 CRC 檢查機制。

x ：沒有被使用，數值永爲 0。

reserved ：保留，留待進一步的使用，數值永爲 0。

SDO 區塊上傳範例

SDO 區塊上傳大致的過程如下圖所示：



根據上圖，底下將使用範例來對 SDO 區塊上傳協定作進一步的描述。(存取物件字典中，主索引爲 0x1008，子索引爲 0x00 的物件項目)

步驟 1：開始進行初始 SDO 區塊上傳協定，用戶端要求 CAN-8423 使用 SDO 區塊上傳的方式來傳送資料。

初始 SDO 區塊上傳協定 第 76 頁

11-bit COB-ID (601)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	A0	08	10	00	7F	00	00	00

SDO 用戶端 → **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 5
cc ： 0
cs ： 0
m ： 08 10 00
blksize ： 7F
每一個區塊內含 127 個區段。
pst ： 00

步驟 2：CAN-8423 透過初始 SDO 區塊上傳協定回覆訊息給用戶端，以確認用戶端所提出的要求。

初始 SDO 區塊上傳協定 第 76 頁

11-bit COB-ID (581)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	C2	08	10	00	09	00	00	00

SDO 用戶端 ← **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 6
sc ： 0
s ： 1
ss ： 0
m ： 08 10 00
size ： 09 00 00 00
CAN-8123 會利用 SDO 區塊上傳協定傳送 9 bytes 資料給用戶端。

步驟 3：用戶端傳送此訊息以結束初始 SDO 區塊上傳協定，並通知 CAN-8423 開始進行資料傳輸。

初始 SDO 區塊上傳協定 第 76 頁

11-bit COB-ID (601)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	A3	00	00	00	00	00	00	00

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 5
cs ： 3

步驟 4：開始進行上傳 SDO 區塊區段協定，CAN-8423 把欲傳送資料的前 7 個 bytes 填到區段中，接著傳送此區段給用戶端。

上傳 SDO 區塊區段協定 第 78 頁

11-bit COB-ID (581											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	1	43	41	4E	2D	38	34	32

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

c ： 0
seqno ： 1
seg-data ： 43 41 4E 2D 38 34 32

步驟 5：CAN-8423 把資料中尚未傳完的部份繼續填到區段中，並傳輸給用戶端。

上傳 SDO 區塊區段協定 第 78 頁

11-bit COB-ID (581)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	82	33	00	00	00	00	00	00

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

c ： 1
seqno ： 2
seg-data ： 33 00 00 00 00 00 00
因爲欲傳輸的資料只有 9 bytes，所以這個區段會是最後一個區段，並且不是整個 **seg-data** 欄位的資料都有意義。CAN-8423 會在進行結束區塊上傳協定時，告訴 SDO 用戶端，這一個節段的有效資料長度。詳情請見步驟 7 的 **n** 欄位。

步驟 6：然後，使用者會傳送下面訊息，告知 CAN-8423 已收到它傳輸的資料。上傳 SDO 區塊區段協定於此時結束。

上傳 SDO 區塊區段協定 第 78 頁

11-bit COB-ID (601)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	A2	02	7F	00	00	00	00	00

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 5
cs ： 2
ackseq ： 2
blksize ： 7F

步驟 7：當 CAN-8423 收到用戶端傳來的確認訊息後，它就會傳送以下訊息以進行終止 SDO 區塊上傳協定。

終止 SDO 區塊上傳協定 第 80 頁

11-bit COB-ID (581)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	D5	00	00	00	00	00	00	00

SDO 用戶端 ← **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 6

n ： 5

這一個欄位的數值是用來指出，由 CAN-8423 所傳送的最後一個區段內(步驟 5)，哪些 byte 沒有意義。於此處，沒有意義的 byte 爲第 3 個到第 7 個。

ss ： 1

crc ： 00 00

步驟 8：用戶端傳送以下訊息以結束終止 SDO 區塊上傳協定。

終止 SDO 區塊上傳協定 第 80 頁

11-bit COB-ID (601)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	A1	00	00	00	00	00	00	00

SDO 用戶端 → **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 5

cs ： 1

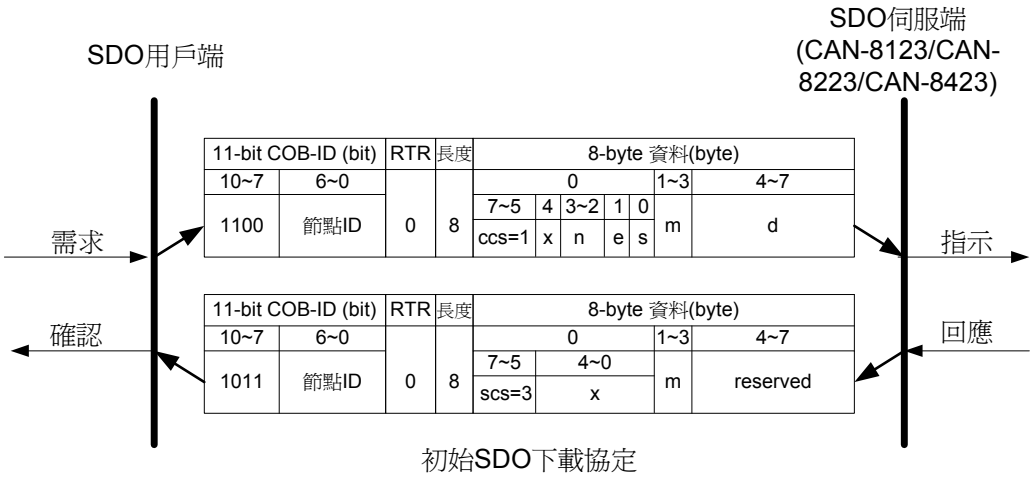
5.1.3 下載 SDO 協定

下載 SDO 協定(Download SDO Protocol)與上傳 SDO 協定十分類似，其通訊協定僅有部份參數與上傳 SDO 協定不同。

下載 SDO 協定也可被分爲兩個部份，包括初始 SDO 下載協定和下載 SDO 區段協定。若欲下載的資料長度小於 4 bytes，則僅使用初始 SDO 下載協定即可完成資料的下載，若欲下載的資料長度大於 4 bytes，除了初始 SDO 下載協定，還必須進行下載 SDO 區段協定，才能把資料下載完畢。

底下將針對這兩種協定進行描述。

初始 SDO 下載協定 (Initiate SDO Download Protocol)



初始SDO下載協定
(Initiate SDO Download Protocol)

ccs ：用戶端命令識別符(client command specifier)
1：初始下載要求(initiate download request)

scs ：伺服端命令識別符(server command specifier)
3：初始下載回應(initiate download response)

n ：只有當 **e**=1 且 **s**=1 此欄位才有意義，否則 **n**=0。
若此欄位有意義，則 **n** 代表 **d** 欄位內沒有資料的 byte 數目，即第 8-**n** 個 byte 到第 7 個 byte 內，沒有節段資料(segment data)。

e ：傳輸型式(transfer type)
0：正規傳輸(normal transfer)
1：附件傳輸(expedited transfer)
若 **e**=1，即代表欲傳輸物件的資料量小於或等於 4 bytes，僅需使用初始 SDO 下載協定即可傳送完畢。
若 **e**=0，就必需要進行下載 SDO 區段協定。

s ：資料量指示符(size indicator)
0：代表幀(frame)內沒有資料集(Data Set)大小的資訊。
1：代表幀(frame)內有資料集(Data Set)大小的資訊。

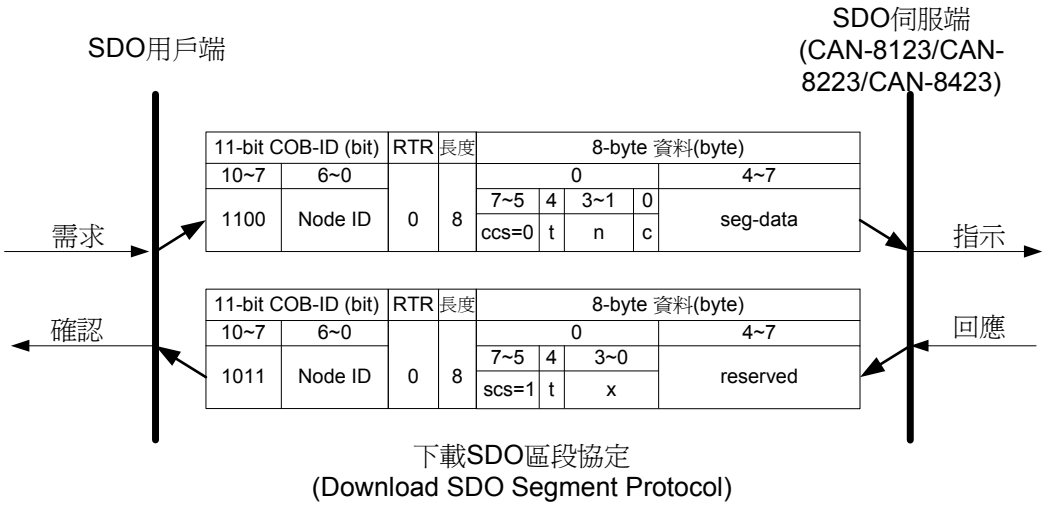
m ：多工器(multiplexor)
其代表欲傳輸 SDO 物件其內含的資料在物件字典的主索引和子索引。前兩個 byte 代表主索引，後一個 byte 代表子索引。

d ：資料
e=0，**s**=0：表示 **d** 被保留，留待進一步的使用。
e=0，**s**=1：**d** 包含了欲上傳 byte 的數目，其中 byte 4 內含最低有效位元(least significant bit)，byte 7 內含最高有效位元(most significant bit)。
e=1，**s**=1：**d** 的內容爲欲上傳的資料，其長度爲 4-**n**。資料編碼的方式取決於主索引和子索引在物件字典中參照項目的資料型別。
e=1，**s**=0：**d** 的內容爲未標示長度的上傳資料。

x ：沒有被使用，數值永爲 0。

reserved ：保留，留待進一步的使用，數值永爲 0。

下載節段協定 (Download Segment Protocol)



下載SDO區段協定
(Download SDO Segment Protocol)

ccs ：用戶端命令識別符(client command specifier)
0：下載節段要求(download segment request)

scs ：伺服端命令識別符(server command specifier)
1：下載節段回應(download segment response)

seg-data ：其內爲欲下載的節段資料，一次最多可下載 7 bytes。資料編碼的方式取決於初始 SDO 下載協定內，主索引和子索引在物件字典中參照項目的資料型別。

n ：**n** 內含 **seg-data** 欄位內沒有節段資料的 byte 數目，即第 8-**n** 個 byte 到第 7 個 byte 內，沒有節段資料(segment data)。如果 **n** = 0，代表節段的大小沒有被指示。

c ：用來指出是否還有節段要被上傳。
0：還有節段等待上傳。
1：已經沒有節段需要上傳。

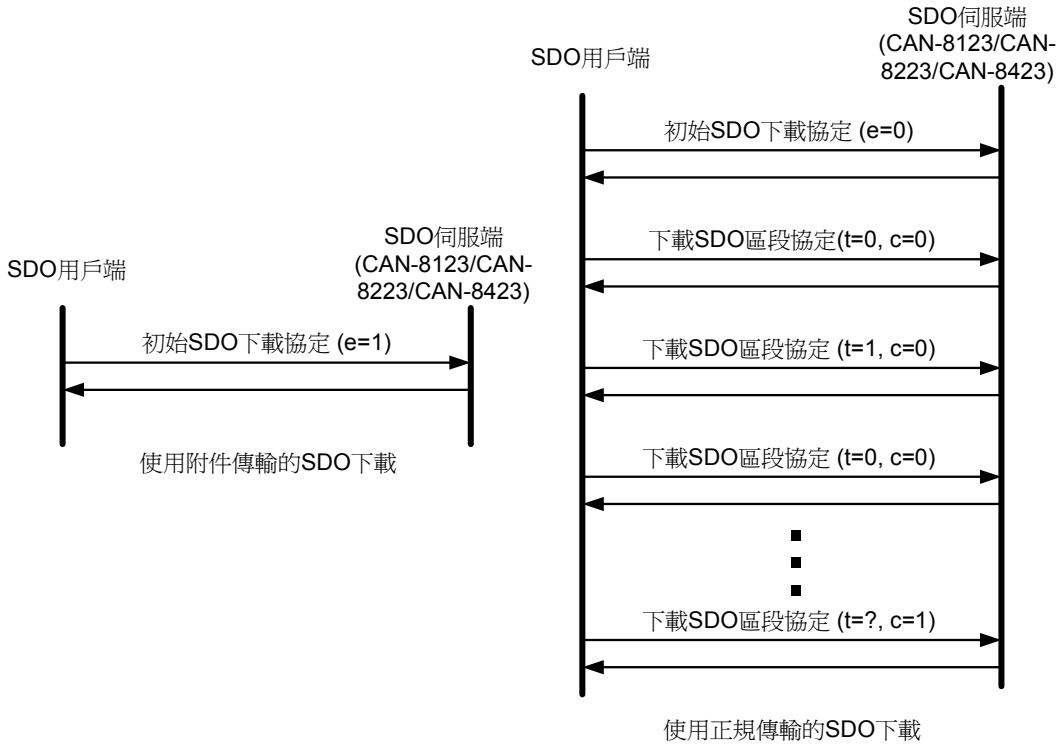
t ：交替位元(toggle bit)
對每一連續的下載節段而言，每一個節段的交替位元均需與其上一個或下一個節段的交替位元不同。而第 1 個節段的交替位元必須設定爲 0，另外要求訊息和回應訊息的交替位元必須相同。

x ：沒有被使用，數值永爲 0。

reserved ：保留，留待進一步的使用，數值永爲 0。

SDO 下載範例

底下將使用實際的例子來示範如何用下載 SDO 協定來進行資料的下載：



由於在 CAN-8123/CAN-8223/CAN-8423 內，所有可被寫入的物件項目，其資料長度均不超過 4 bytes，因此底下僅示範如何使用附件傳輸。

- **附件傳輸 (expedited transfer)的範例**

步驟 1：傳送 Rx SDO 訊息給 CAN-8423，以更改在通訊描述文件區域內，主索引爲 0x1400 且子索引爲 0x02 的物件項目，於此處將此物件項目的值更改爲 5。另外假設 CAN-8423 的節點 ID 被設定爲 1。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (601)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2F	00	14	02	05	00	00	00

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 1
n ： 3
e ： 1
s ： 1
m ： 00 14 02
d ： 05 00 00 00
因爲 **n=3**，所以只有第 4 個 byte 會有資料，於此處其值爲 0x05。

步驟 2：CAN-8423 將會回覆此訊息以結束資料的下載。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (601)											RT R	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	00	14	02	00	00	00	00

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 3
M ： 00 14 00

在結束資料的下載後，使用者可以使用前面所介紹過的上傳 SDO 通訊協定來把寫入的資料再讀出來比對，以確定資料是否正確地下載至 CAN-8423。

5.1.4 SDO 區塊下載協定

SDO 區塊下載協定(SDO Block Download Protocol)很類似於 SDO 區塊上傳協定。SDO 區塊下載協定共可分爲三個部份，分別爲初始 SDO 區塊下載協定(Initiate SDO Block Download protocol)、SDO 區塊下載協定(SDO Block Download protocol)以及終止 SDO 區塊下載協定(End SDO Block Download protocol)。

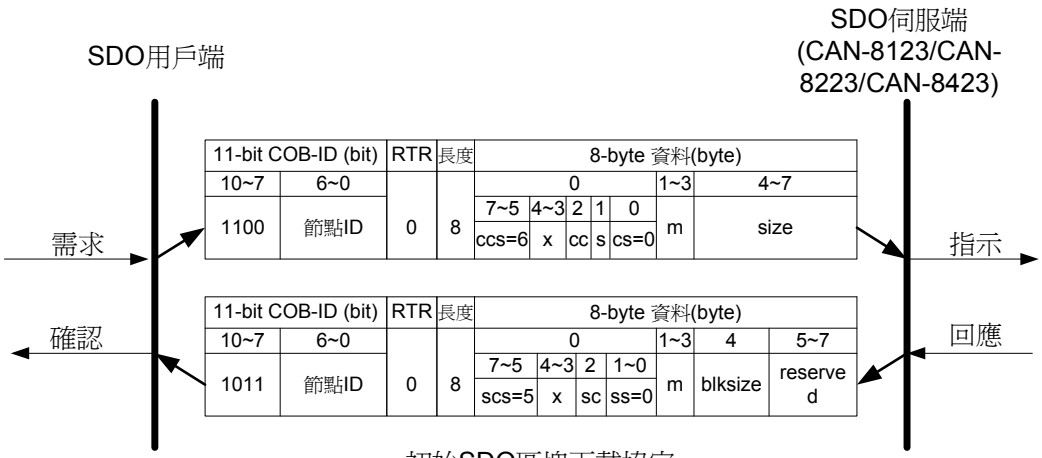
初始 SDO 下載協定爲 SDO 區塊下載協定的起始協定，透過此協定，SDO 伺服端和 SDO 用戶端可以互相溝通以準備必要的資訊。

在初始 SDO 下載協定結束後，便可以進行 SDO 區塊區段下載協定，讓 SDO 用戶端傳送資料給 SDO 伺服端。

最後，SDO 用戶端和 SDO 伺服端必須透過終止 SDO 區塊下載協定來結束 SDO 區塊下載協定。

底下將詳細描述這三個協定的架構。

初始 SDO 區塊下載協定 (Initiate SDO Block Download Protocol)



初始SDO區塊下載協定
(Initiate SDO Block Download Protocol)

ccs ：用戶端命令識別符(client command specifier)
6：區塊下載(block download)

scs ：伺服端命令識別符(server command specifier)
5：區塊下載(block download)

s ：資料量指示符(size indicator)
0：代表幀(frame)內沒有資料集(Data Set)大小的資訊。
1：代表幀(frame)內有資料集(Data Set)大小的資訊。

cs ：用戶端子指令(client subcommand)
0：初始下載要求(initiate download request)

ss ：伺服端子指令(server subcommand)
0：初始下載回應(initiate download response)

cc ：用戶端循環冗餘檢查支援(client CRC support)
cc=0：用戶端不支援對資料產生 CRC。
cc=1：用戶端支援對資料產生 CRC。.

sc ：伺服端循環冗餘檢查支援(client CRC support)
sc=0：伺服端不支援對資料產生 CRC。
sc=1：伺服端支援對資料產生 CRC。

m ：多工器(multiplexor)
其內含 SDO 所傳輸的資料，其主索引和子索引。

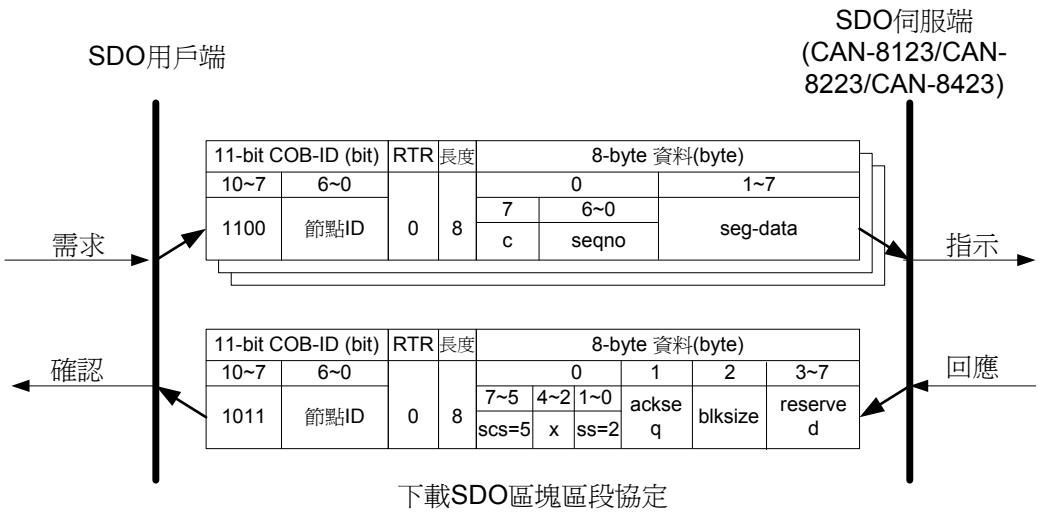
size ：欲下載的資料量大小(byte)
s=0：**size** 欄位被保留，留待進一步的使用，數值永爲 0。
s=1：**size** 欄位內包含了欲下載 byte 的數目，其中 byte 4 內含最低有效位元(least significant bit)，byte 7 內含最高有效位元(most significant bit)。

blksize ：每一個區塊內包含的區段數目(0 < **blksize** < 128)。

x ：沒有被使用，數值永爲 0。

reserved ：保留，留待進一步的使用，數值永爲 0。

下載SDO 區塊區段協定 (Download SDO Block Segment Protocol)



下載SDO區塊區段協定
(Download SDO Block Segment Protocol)

scs ：伺服端命令識別符(server command specifier)
5：區塊下載(block download)

ss ：伺服端子指令(server subcommand)
0：初始下載回應(initiate download response)

c ：此欄位用來指出是否仍有區段等待被上傳。
0：尚有區段需要被下載。
1：已經沒有區段需要被下載，準備執行"終止 SDO 區塊下載" (End SDO block download)協定。

seqno ：區段序號(0 < **seqno** < 128)

seg-data ：一個區段最多可以內含 7 個 bytes 的資料。

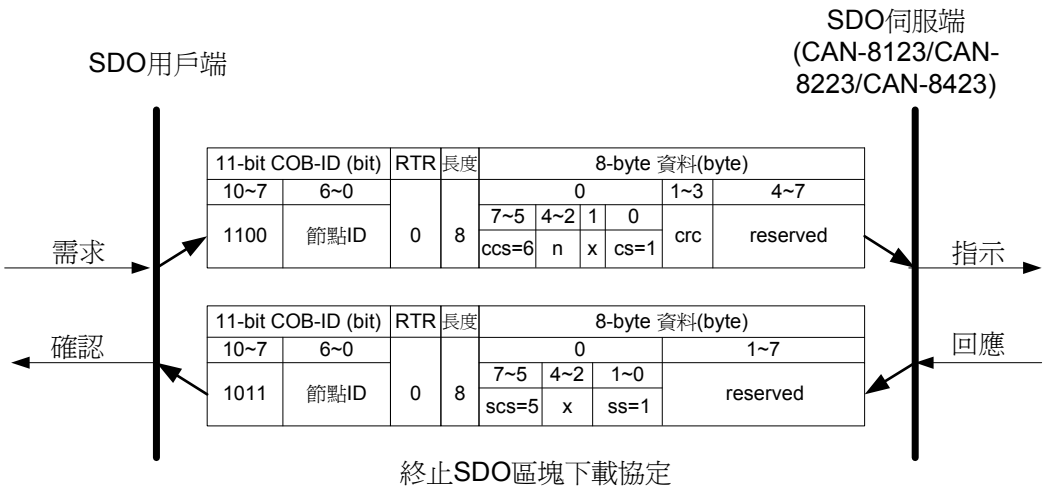
ackseq ：註記最近一次的區塊下載內，最後一個成功接收區段的序號。若 **ackseq** 被設定爲 0，則表示從編號爲 1 的區段開始，用戶端就沒有接收到。此時伺服端必需要重送所有的區段。

blksize ：用來讓用戶端告知伺服端，在下一個要傳送的區塊內，應擺放的區段個數。(0 < **blksize** < 128)

x ：沒有被使用，數值永爲 0。

reserved ：保留，留待進一步的使用，數值永爲 0。

終止 SDO 區塊下載協定 (End SDO Block Download Protocol)



終止SDO區塊下載協定
(End SDO Block Download Protocol)

ccs ：用戶端命令識別符(client command specifier)
6：區塊下載(block download)

scs ：伺服端命令識別符(server command specifier)
5：區塊下載(block download)

cs ：用戶端子指令(client subcommand)
1：終止區塊下載要求(end block download request)

ss ：伺服端子指令(server subcommand)
1：終止區塊下載回應(end block download response)

n ：用來註記最後一次傳輸的區塊內的最後一個區段，沒有內含資料的 byte 數目。其中，未包含區段資料的部份爲第 8-**n** 個 byte 到第 7 個 byte。

crc ：對所有資料集合(data set)所做的 16bit 循環冗餘檢查(CRC，Cyclic Redundancy Checksum)碼。

用來計算 CRC 的多項式： $x^{16}+x^{12}+x^5+1$

只有在進行初始 SDO 區塊下載協定時，**cc** 和 **sc** 都被設定成 1，**crc** 欄位才會被啓用，否則的話 **crc** 欄位的值就會被設定爲 0。而 CAN-8123/CAN-8223/CAN-8423 目前並不支援 CRC 檢查機制。

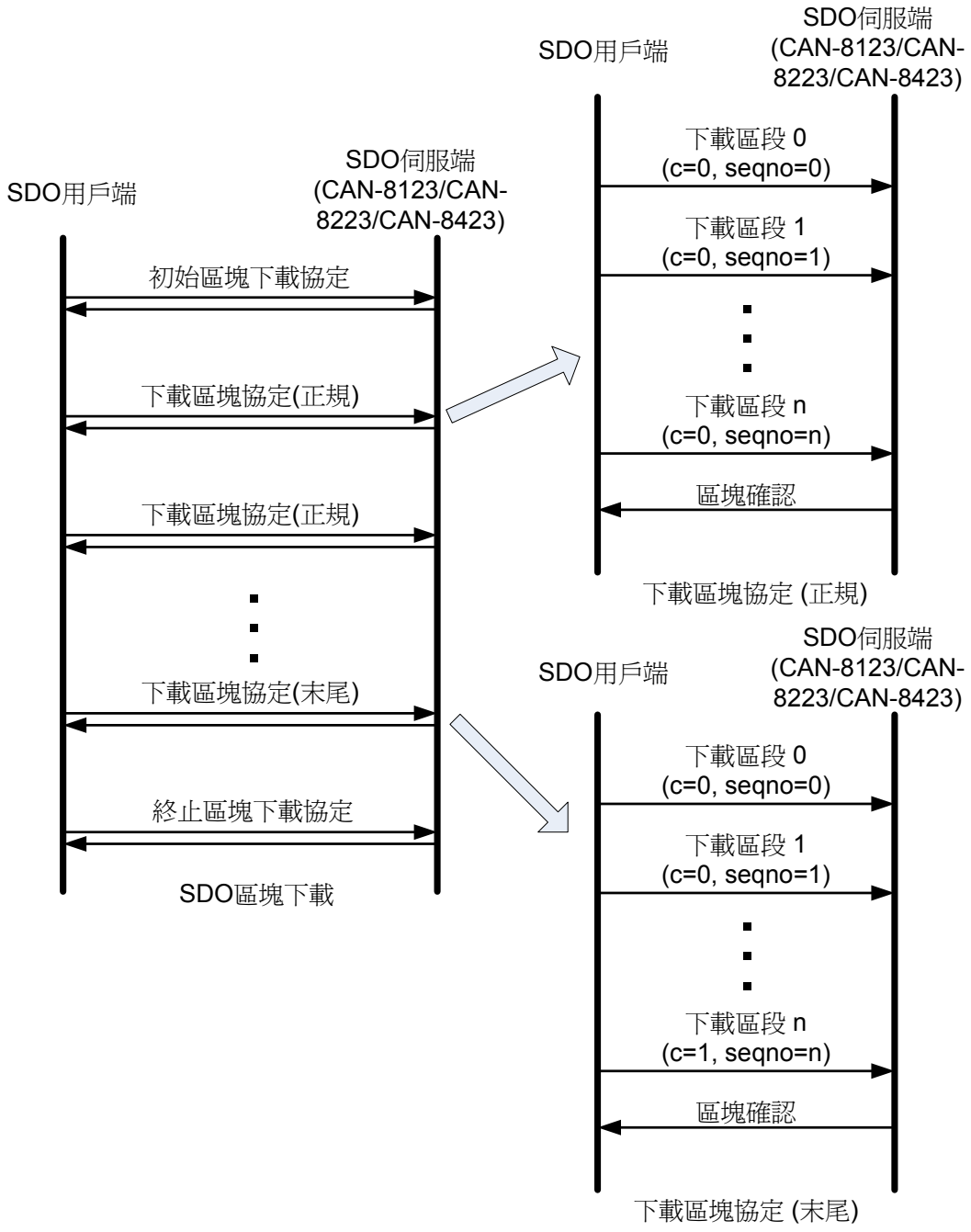
X ：沒有被使用，數值永爲 0。

reserved ：保留，留待進一步的使用，數值永爲 0。

SDO 區塊下載範例

底下將利用例子來示範，如何利用 SDO 區塊下載協定將物件字典中主索引爲 0x1400，子索引爲 0x02 的項目之值變更爲 5。

SDO 區塊下載協定的執行過程如下所示：



步驟 1：於下將透過 SDO 區塊下載協定更改其物件字典中，主索引爲 0x1400，子索引爲 0x02 的項目內容，在此之前，必須先利用初始 SDO 區塊下載協定傳送初始訊息給 CAN-8423。

初始 SDO 區塊下載協定 第 92 頁

11-bit COB-ID (601)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	C0	00	14	02	00	00	00	00

SDO 用戶端 → **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 6
cc ： 0
s ： 0
cs ： 0
m ： 00 14 02
size ： 00 00 00 00
因爲 **s** 欄位的值爲 0，所以 **size** 欄位於此處沒有被使用。

步驟 2：CAN-8423 在收到用戶端的訊息後，便會回傳一訊息以結束初始 SDO 區塊下載協定。接著，SDO 用戶端便可以開始下載資料到 CAN-8423 的物件字典中，主索引爲 0x1400，子索引爲 0x02 的項目。

初始 SDO 區塊下載協定 第 92 頁

11-bit COB-ID (581)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	A0	00	14	02	7F	00	00	00

SDO 用戶端 ← **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 5
sc ： 0
s ： 0
ss ： 0
m ： 00 14 02
blksize ： 7F

步驟 3：SDO 用戶端開始利用下載 SDO 區塊區段協定傳輸資料到 CAN-8423 的物件字典內主索引爲 0x1400，子索引爲 0x02 的物件項目。若欲傳輸的資料長度小於一個區塊所能傳送的最大資料長度的話，則僅需要進行一次下載 SDO 區塊區段協定即可。

下載 SDO 區塊區段協定 第 94 頁

11-bit COB-ID (601)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	81	05	00	00	00	00	00	00

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

c ： 1
seqno ： 1
seg-data ： 05 00 00 00 00 00 00
因爲傳輸資料量的關係，此一節段就是最後一個節段，所以 **seg-data** 欄位內不一定全部的資料都有意義。而使用者於後在進行終止 SDO 區塊下載協定時，會告訴 CAN-8423，目前這一個訊息的 **seg-data** 欄位，究竟哪些資料是有意義的。詳情請參照步驟 5 內的 **n** 欄位。

步驟 4：CAN-8423 回應此訊息給使用者，以告知使用者訊息的傳送是成功還是失敗。如果訊息傳送失敗了，則這一個區塊就必須再傳送一遍。當 CAN-8423 向 SDO 用戶端回報最後一個區塊傳送成功，此時也就代表著下載 SDO 區塊區段協定的結束。

下載 SDO 區塊區段協定 第 94 頁

11-bit COB-ID (601)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	A2	01	7F	00	00	00	00	00

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423**

scs ： 5
ss ： 2
ackseq ： 01
blksize ： 7F

步驟 5：透過終止 SDO 區塊下載協定，SDO 用戶端傳送終止訊息給 CAN-8423。

終止 SDO 區塊下載協定 第 95 頁

11-bit COB-ID (601)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	D9	00	00	00	00	00	00	00

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423**

ccs ： 6

n ： 6

這一個欄位的數值是用來指出，由 SDO 用戶端所傳送的最後一個區段內(步驟 4)，哪些 byte 沒有意義。於此處，沒有意義的 byte 爲第 2 個到第 7 個，也就是只有前兩個 bytes 有意義。

cs ： 1

crc ： 00 00

步驟 6：CAN-8423 回覆 SDO 用戶端傳來的中止訊息，此時 SDO 區塊下載協定正式結束。

終止 SDO 區塊下載協定 第 95 頁

11-bit COB-ID (601)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	A1	00	00	00	00	00	00	00

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423**

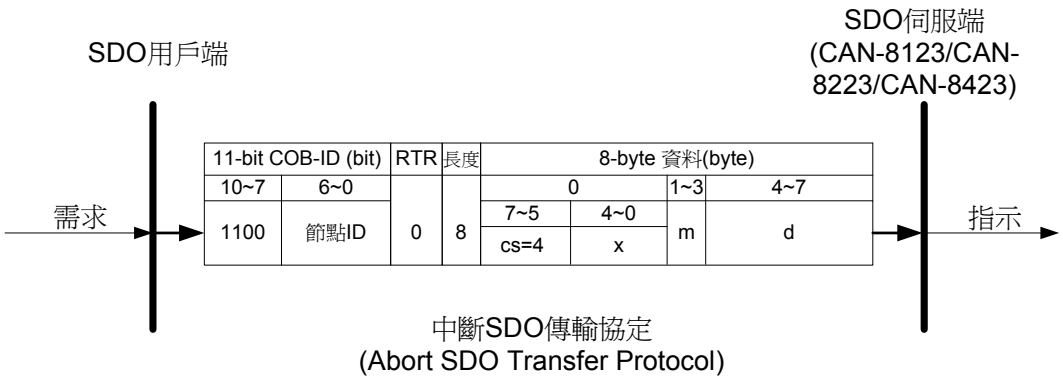
scs ： 5

ss ： 1

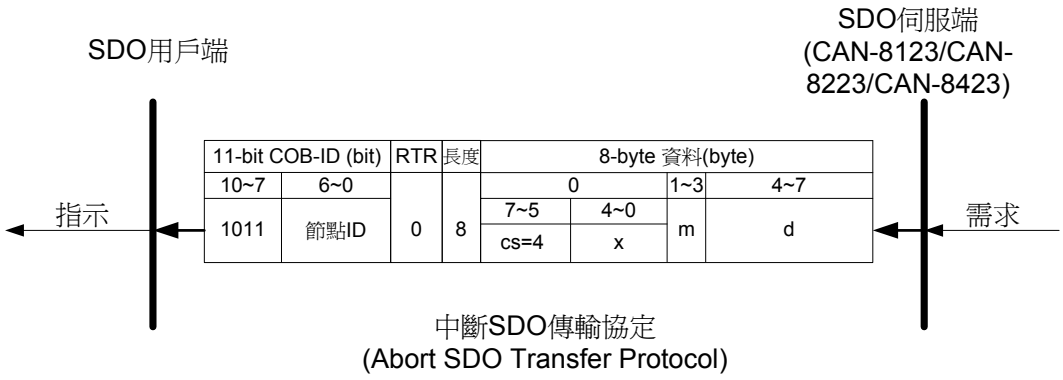
5.1.5 中斷 SDO 傳輸協定

在某些情況下，SDO 用戶端和 SDO 伺服端會需要中斷 SDO 的傳輸。舉例來說，像是想要更改的物件項目是唯讀的，或著要存取的物件項目根本不存在，亦或是用戶端和伺服端基於其它理由不想完成 SDO 的傳輸。當這些情況發生時，SDO 用戶端和 SDO 伺服端均可以主動的透過中斷 SDO 傳輸協定，傳送中斷訊息以中斷 SDO 的傳輸。

底下將介紹中斷 SDO 傳輸協定(Abort SDO Transfer Protocol)：



中斷SDO傳輸協定
(Abort SDO Transfer Protocol)



中斷SDO傳輸協定
(Abort SDO Transfer Protocol)

cs ：命令識別符(command specifier)
4：中斷傳輸要求(abort transfer request)

x ：沒有被使用，數值永爲 0。

m ：多工器(multiplexor)
其代表欲中斷 SDO 物件其內含的資料在物件字典的主索引和子索引。前兩個 byte 代表主索引，後一個 byte 代表子索引。

d ：內含 4 bytes 的中斷碼(abort code)，其代表中斷的原因。

於此處將 SDO 的中斷碼整理如下表：

中斷碼(Abort code)	描述
0503 0000h	交替位元(Toggle bit)沒有變化。
0504 0000h	SDO 協定逾時(timed out)。
0504 0001h	用戶端/伺服端的命令識別符(command specifier)無效或著無法解譯。
0504 0002h	無效的區塊大小。(僅限區塊模式)
0504 0003h	無效的序號(sequence number)。(僅限區塊模式)
0504 0004h	CRC 錯誤。(僅限區塊模式)
0504 0005h	記憶體不足。
0601 0000h	物件的存取不被支援。(不可被讀寫)
0601 0001h	嘗試讀取唯寫(write only)的物件。
0601 0002h	嘗試寫入唯讀(read only)的物件。
0602 0000h	物件字典內不存在此物件。
0604 0041h	物件無法映射(mapped)給 PDO。
0604 0042h	被映射的物件數量和長度超過 PDO 所能承載的負荷。
0604 0043h	一般的參數設定相容性問題。
0604 0047h	一般的裝置內部相容性問題。
0606 0000h	基於硬體錯誤導致的存取失敗。
0607 0010h	服務參數(service parameter)長度不符引起的資料型別不符。
0607 0012h	服務參數(service parameter)長度過長引起的資料型別不符。
0607 0013h	服務參數(service parameter)長度過短引起的資料型別不符。
0609 0011h	子索引不存在。
0609 0030h	欲寫入的參數，其數值超出範圍。
0609 0031h	欲寫入的參數數值過高。
0609 0032h	欲寫入的參數數值過低。
0609 0036h	最大數值小於最小數值。
0800 0000h	一般性的錯誤。
0800 0020h	資料無法傳送或著儲存到應用程式(application)。
0800 0021h	因爲本地端控制(local control)，資料無法傳送或著儲存到應用程式。
0800 0022h	因爲目前的裝置狀態，資料無法傳送或著儲存到應用程式。
0800 0023h	物件字典在動態產生時發生錯誤，或著物件字典不存在(若物件字典的產生必須藉由載入檔案來完成，萬一檔案載入發生發生錯誤，那麼物件字典就無法被建立)。

<u>中斷 SDO 傳輸範例</u>

在物件字典中，主索引爲 0x1008 的物件，沒有子索引 0x01 的項目。因此，若使用者想要透過 SDO 協定來讀取主索引爲 0x1008 且子索引爲 0x01 的物件項目，此時 CAN-8123/CAN-8223/CAN-8423 就會回應中斷 SDO 傳輸訊息。於下將詳細描述這個過程：

步驟 1：SDO 用戶端透過初始 SDO 上傳協定，傳送 RxSDO 訊息給 CAN-8423，表示希望讀取物件字典中主索引爲 0x1008 且子索引爲 0x01 的物件項目。

初始 SDO 上傳協定 第 67 頁

11-bit COB-ID (601)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	40	08	10	01	00	00	00	00

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423**

Ccs ： 2
M ： 08 10 01

步驟 2：CAN-8423 會回應中斷 SDO 訊息給 SDO 用戶端。

中斷 SDO 傳輸協定 第 100 頁

11-bit COB-ID (581)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	80	08	10	01	11	00	09	06

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423**

cs ： 4
m ： 08 10 01
d ： 11 00 09 06
因爲低字組會被先傳送，所以接收下來的資料在轉換之後，正確的順序應該爲"06 09 00 11"。若是查找上一頁的中斷碼表格，將會發現中斷碼代表的意義爲"子索引不存在"。

5.2 PDO 通訊集

5.2.1 PDO COB-ID 參數

每一個 PDO 在物件字典內都會有其對應的 PDO 通訊參數(PDO communication parameter)，在使用 PDO 之前，必須要先查詢物件字典中，PDO 通訊參數物件內的 COB-ID 項目(子索引 0x01)。COB-ID 項目內記錄了 PDO 在傳輸時會使用的 COB-ID，其共有 32 bits，而此處將每一個 bit 所代表的意義整理如下表：

Bit 編號	值	代表的意義
31 (MSB)	0	PDO 存在 (此 PDO 是有效的，valid)
	1	PDO 不存在 (此 PDO 是無效的，invalid)
30	0	此 PDO 允許 RTR 的傳輸方式
	1	此 PDO 不允許 RTR 的傳輸方式
29	0	11-bit ID (CAN 2.0 A)
	1	29-bit ID (CAN 2.0 B)
28-11	0	若 bit 29=0，此欄位的數值便爲 0
	x	若 bit 29=1：則此欄位就是 29bits COB-ID 內的第 28~11 bits
10-0 (LSB)	x	COB-ID 內的第 10~0 bits

註： CAN-8123/CAN-8223/CAN-8423 僅支援 CAN 2.0 A，也就是僅支援 11 bit 的 COB-ID。

此處將預設的 PDO COB-ID 參數整理如下表：

PDO 的編號	預設的 PDO COB-ID	
	Bit10~Bit7 (功能碼)	Bit6~Bit0
TxPDO1	0011	節點 ID
TxPDO2	0101	節點 ID
TxPDO3	0111	節點 ID
TxPDO4	1001	節點 ID
RxPDO1	0100	節點 ID
RxPDO2	0110	節點 ID
RxPDO3	1000	節點 ID
RxPDO4	1010	節點 ID

註：
1. 除了在 3.1 節內，提到被保留的 COB-ID 使用者不可以拿來自行定義之外，其他的 COB-ID 使用者均可以拿來定義爲 PDO 的 COB-ID。當使用者欲自行定義 PDO 的 COB-ID 時，必須小心避免在同一節點 (同一裝置) 上，某 COB-ID 被不同 COB 重複使用的情形。

2. 若 PDO 爲有效狀態(bit 31=0)，則此時 PDO 的 COB-ID 參數便不允許被更改。細節可見第 136 頁的範例。

5.2.2 傳輸型態

PDO 通訊參數內含數個具有不同作用的參數，其中子索引爲 0x02 的參數爲傳輸型態(transmission type)，而每一個 PDO 均可對其設定傳輸型態。我們可以透過傳輸型態來了解此 PDO 在傳送與接收時的特性。

舉例來說，若使用者設定第 1 個 TxPDO 的傳輸型態爲 0，則 CANopen 的裝置便會利用非循環同步的方式來進行第 1 個 TxPDO 的傳輸。此處將不同傳輸型態與其對應的 PDO 特性之關係整理如下表。

傳輸型態	PDO 傳輸方式				
	同步	非同步	循環	非循環	唯遠端傳送要求
0		O	O		
1-240	O		O		
241-251	----------reversed----------				
252			O		O
253				O	O
254				O	
255				O	

註：
1. TxPDO 的傳輸型態若是 1-240，則代表需要接收到這麼多個 SYNC 物件才能夠觸發 TxPDO 的傳送。RxPDO 的傳輸型態若是 0-240，則僅需要一個 SYNC 物件的接收便可以啓用在此之前收到且尚未被啓用的 RxPDO 物件，與傳輸型態的數字大小無關。
2. 只有 TxPDO 的傳輸型態可以被設定爲 252 和 253。傳輸型態若被設定爲 252，則代表裝置在接收到 SYNC 物件時，才會更新 TxPDO 內的資料。傳輸型態若被設定爲 253，則在接收到 RTR 訊息時，裝置才會更新 TxPDO 內的資料。TxPDO 若是被設定爲這兩種型態，則只有在接收到此 TxPDO 的 RTR 訊息時，裝置才會對外傳送 TxPDO。
3. 傳輸型態若是被設定爲 254 和 255，便可以使用事件計時器(event timer)來觸發 TxPDO 的傳送。另外若某 DI 被映射到某個 PDO，當此 DI 的值被變更時，也會觸發其對應 TxPDO 的傳送。對 RxPDO 而言，若是傳輸型態被設定爲 254 或 255，則在接收到 RxPDO 之後，就必須立即啓用此 RxPDO。

5.2.3 PDO 通訊規則

根據 CANopen DS-301 的規範，與 PDO 有關的物件乃存放於物件字典中主索引 0x1400 到 0x1BFF 之間。而在 CAN-8123/CAN-8223/CAN-8423 內，其 RxPDO 的通訊參數(communication parameter)位於物件字典中主索引 0x1400 到 0x140F 之間，其 RxPDO 的映射參數(mapping parameter)位於物件字典中主索引 0x1600 到 0x160F 之間，其 TxPDO 的通訊參數位於物件字典中主索引 0x1800 到 0x180F 之間，其 TxPDO 的映射參數位於物件字典中主索引 0x1A00 到 0x1A0F 之間。此外，每一個 PDO 的通訊參數物件均會對應到一個映射參數物件，兩者之間爲一對一的關係。

更進一步的來說，譬如第 1 組 RxPDO 通訊參數存放於物件字典中主索引爲 0x1400 的地方，而其相對應的映射參數便會存放於物件字典中主索引爲 0x1600 的地方，可依序推得而主索引 0x1401 和 0x1601 爲一對，主索引 0x1402 和 0x1602 爲一對・・・等。TxPDO 的通訊參數和映射參數的關係同樣依循這樣的關係，譬如第 1 個 TxPDO 通訊參數存放於物件字典中主索引爲 0x1800 的地方，而其相對應的映射參數便會存放於物件字典中主索引爲 0x1A00 的地方，可依序推得而主索引 0x1801 和 0x1A01 爲一對，主索引 0x1802 和 0x1A02 爲一對・・・等。在使用者開始利用 PDO 對實際的 I/O 通道作存取前，必須要先取得 PDO 的通訊參數和映射參數。

此外，PDO 的通訊只能在 NMT 的操作(operational)狀態下使用，若使用者要使用 PDO 來進行資料的傳輸，可以透過 NMT 模組控制協定(NMT module control protocol)，傳送模組控制訊息給 CAN-8123/CAN-8223/CAN-8423，要求裝置改變 NMT 狀態爲操作狀態。詳細的內容將於 5.3 節內作介紹。

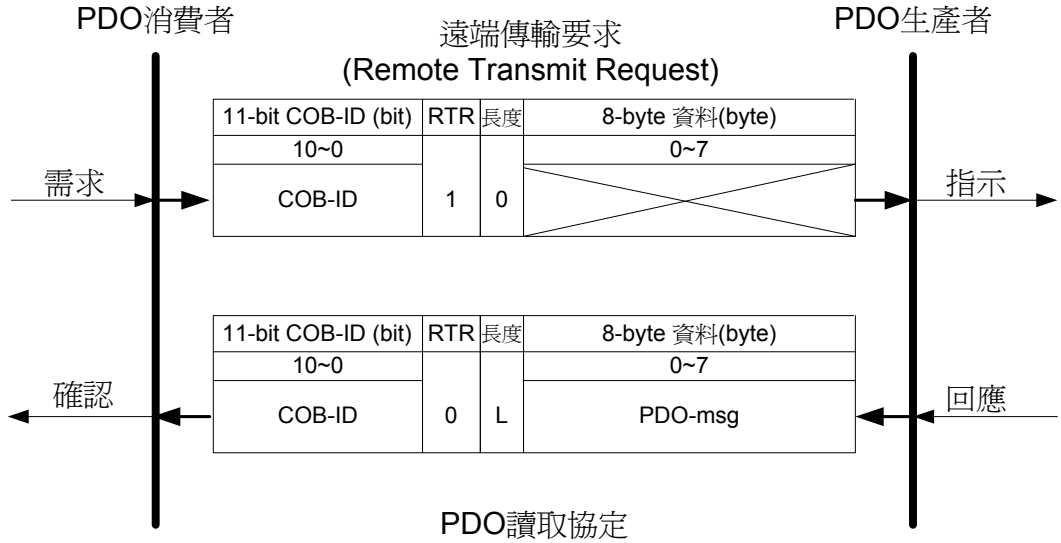
附帶一提，透過 PDO 來傳送訊息，PDO 內的資料長度必須和其對應 PDO 映射參數內所記載的資料長度相吻合，當 PDO 消費者收到 PDO 訊息時，會根據此 PDO 的 COB-ID 來查找相對應的 RxPDO 映射參數。若此 PDO 內的資料長度(假設爲 L bytes)大於其映射參數所記載的長度(假設爲 n bytes)，則 PDO 消費者只會取前 n bytes 來使用，其餘部份則丟棄。若此 PDO 內的資料長度小於其映射參數所記載的長度，則 PDO 消費者將不會處理這個 PDO，並且會發出一個錯誤碼爲 8210h 的 EMCY(Emergency)訊息給 PDO 的生產者。

註： 關於 EMCY 訊息，請參照 5.3 節。

底下爲 PDO 的通訊協定：

PDO寫入協定
(Write PDO Protocol)

COB-ID ：預設的 PDO COB-ID，或是使用者定義的 PDO COB-ID。
L ：PDO 訊息所使用的資料長度(bytes)。
PDO-msg ：即時性的資料，或著可以用作 PDO 映射的資料。



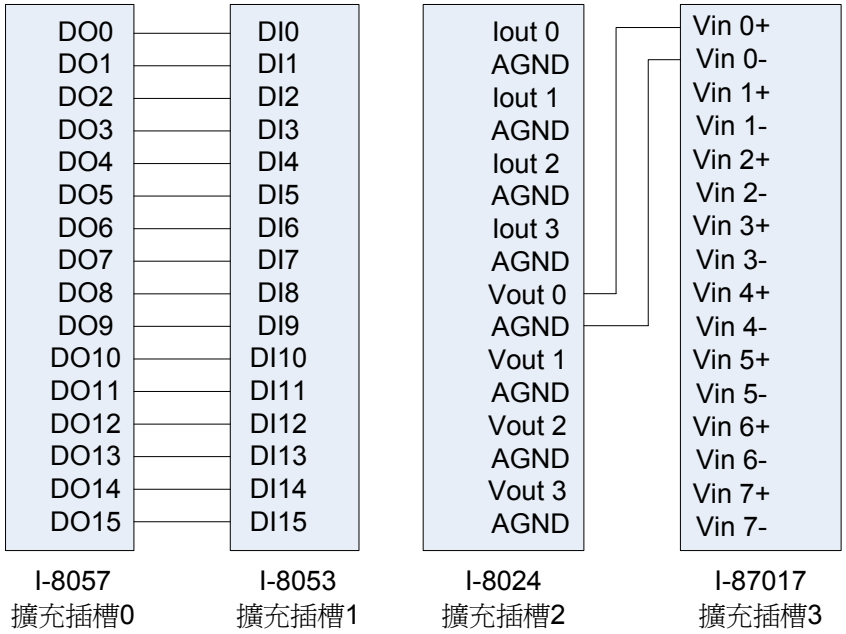
PDO讀取協定
(Read PDO Protocol)

COB-ID ：預設的 PDO COB-ID，或是使用者定義的 PDO COB-ID。
L ：PDO 訊息所使用的資料長度(bytes)。
PDO-msg ：即時性的資料，或著可以拿來作 PDO 映射的資料。

註： 上圖中，部分欄位內爲交叉符號，即代表此訊息沒有交叉符號的部份。

PDO 通訊範例

在開始敘述 PDO 的通訊範例之前，此處假設依序安裝了 4 個 I-8000 的擴充模組在 CAN-8423 的四個擴充插槽上，分別爲 I-8057、I-8053、I-8024 以及 I-87017。並將這些模組的輸出輸入埠連接如下圖：



利用 CAN-8423 面板上的旋鈕，設定裝置的節點 ID 爲 1，設定 CAN bus 的鮑率爲 125Kbps。再透過 CAN 的僕端工具程式設定 I-8024 和 I-87017 的輸入/輸出範圍爲 -10V ~ +10V。

於使用 CAN 僕端工具程式的過程當中，可以看到類似底下幾張圖的畫面。(CAN-8123/CAN-8223 不能夠於在線模式下，來設定裝置上通道的輸入/輸出範圍，不過使用者可以另外透過 SDO 協定來完成這件事，詳情請參照 5.5 節)

PDO NO.	COB-ID (Hex)	Transmission	Inhibit Time	Event Timer	Mapping 0	Mapping 1	Ma
1	201	255	0	0	s 0c00~07	s 0c08~15	s--c
2	301	255	0	0	s 2c 0	s 2c 0	s 2
3	401	255	0	0	s--c--	s--c--	s--c
4	501	255	0	0	s--c--	s--c--	s--c

▪▪▪▶

▪▪▪▶

Mapping 2	Mapping 3	Mapping 4	Mapping 5	Mapping 6	Mapping 7
s--c--	s--c--	s--c--	s--c--	s--c--	s--c--
s 2c 1	s 2c 1	s 2c 2	s 2c 2	s 2c 3	s 2c 3
s--c--	s--c--	s--c--	s--c--	s--c--	s--c--
s--c--	s--c--	s--c--	s--c--	s--c--	s--c--

RxPDO 資訊 (RxPDO 映射表)

PDO NO.	COB-ID (Hex)	Transmission	Inhibit Time	Event Timer	Mapping 0	Mapping 1	Ma
1	181	255	0	0	s 1c00~07	s 1c08~15	s--c
2	281	255	0	0	s 3c 0	s 3c 0	s 3(
3	381	255	0	0	s 3c 4	s 3c 4	s 3(
4	481	255	0	0	s--c--	s--c--	s--c

▪▪▪▶

▪▪▪▶

Mapping 2	Mapping 3	Mapping 4	Mapping 5	Mapping 6	Mapping 7
s--c--	s--c--	s--c--	s--c--	s--c--	s--c--
s 3c 1	s 3c 1	s 3c 2	s 3c 2	s 3c 3	s 3c 3
s 3c 5	s 3c 5	s 3c 6	s 3c 6	s 3c 7	s 3c 7
s--c--	s--c--	s--c--	s--c--	s--c--	s--c--

TxPDO 資訊 (TxPDO 映射表)

註： Transmission 欄位內的數值爲傳輸型態(transmission type)，於 CAN-8123/CAN-8223/CAN-8423 內所有 PDO 的傳輸型態預設均爲 255。

Mapping 0 代表 PDO 第 0 個 byte 的映射，而 Mapping 1 則代表 PDO 第 1 個 byte 的映射，依此類推。

s 代表擴充模組。s0 是指安裝在 0 號擴充插槽上面的擴充模組，s1 則是指安裝在 1 號擴充插槽上面的擴充模組，依此類推。

c 代表通道(channel)。c 0 是指通道 0，c00~07 則是指通道 0~通道 7，依此類推。

另外 s--c--代表這個 byte 沒有映射到任何實體的模組，不具意義。

有關 DI/DO/AI/AO 通道的數值如何存放在物件字典中，以及這些數值存放到物件字典之後與 PDO 映射的關係，請參照 3.1 和 3.3 節。

Index	0x6000	0x6200	0x6206	0x6207
Description	Read DI	Write DO	DO Err Mode	DO Err Value
Sub-Index 0	2	2	2	2
Sub-Index 1	8053_DI 0~ DI 7	8057_DO 0~ DO 7	FF	0
Sub-Index 2	8053_DI 8~ DI F	8057_DO 8~ DO F	FF	0

Index	0x6401	0x6411	0x6443	0x6444
Description	Read AI	Write AO	AO Err Mode	AO Err Value
Sub-Index 0	8	4	4	4
Sub-Index 1	87017_AI 0	8024_AO 0	1	0 V
Sub-Index 2	87017_AI 1	8024_AO 1	1	0 V
Sub-Index 3	87017_AI 2	8024_AO 2	1	0 V
Sub-Index 4	87017_AI 3	8024_AO 3	1	0 V
Sub-Index 5	87017_AI 4			
Sub-Index 6	87017_AI 5			
Sub-Index 7	87017_AI 6			
Sub-Index 8	87017_AI 7			

標準化裝置描述文件區域的資訊

在這一節內，將會利用實際的例子來示範如何使用 PDO 通訊的各種功能，於下會介紹的項目有：

- 利用非同步的 PDO 來存取數位 I/O 和類比 I/O。
- 利用事件計時器來取得輸入的數值。
- 非循環同步 RxPDO 的功能展示。
- 非循環同步 TxPDO 的功能展示。
- 循環同步 TxPDO 的功能展示。
- 同步、唯遠端傳輸要求 TxPDO 的功能展示。
- 非同步、唯遠端傳輸要求 RxPDO 的功能展示。
- 針對 DI/AI/DO/AO 通道的動態 PDO 映射。

另外假設於範例中所使用的通訊物件，其 COB-ID 均爲預設值。

- **變更 CAN-8423 的操作狀態**

在開始用例子來示範 PDO 通訊的各項功能之前，必須先更改裝置的 NMT 狀態，因爲 PDO 傳輸只能在 NMT 操作狀態(operational state)下進行。

步驟 0：CAN-8423 在接收到以下的訊息時，若無異常，便會更改其 NMT 狀態爲 NMT 操作狀態。

NMT 模組控制協定 第 159 頁

11-bit COB-ID (000)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0	0	0	0	8	01	01	00	00	00	00	00	00

NMT 主端 ⟶ **NMT 僕端 (CAN-8123/CAN-8223/CAN-8423)**

cs ： 1
節點 ID ： 1

註：關於 NMT，請參照 3.5 節和 5.4 節。

- **存取數位 I/O 和類比 I/O**

步驟 1：如果要更改 I-8057 的數位輸出值爲 0x1234，使用者可以傳送物件字典中的第 1 組 RxPDO 給 CAN-8423，如下：

PDO 通訊協定 第 107 頁

11-bit COB-ID (201)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
0	1	0	0	0	0	0	0	0	0	1	0	8	34	12	00	00	00	00	00	00

PDO 生產者 → **PDO 消費者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x201

L ： 8

PDO-msg ： 34 12 00 00 00 00 00 00

根據裝置預設的 PDO 映射，可以知道第 1 組 RxPDO 內只有 2 bytes 的資料。即使此處將資料長度 L 設定爲 8，CAN-8423 在收到這個 PDO 之後，依據這個 PDO 的映射參數，還是會把後 6 個 bytes 丟掉，僅處理前 2 個 bytes 的資料。而第 1 個 byte 的內容爲 I-8057 上通道 DO0~DO7 的數值，第 2 個 byte 的內容爲 I-8057 上通道 DO8~DO15 的數值。

在 CAN-8423 上，第一組 RxPDO 預設的傳輸型態爲 255，所以 CAN-8423 在收到這個 PDO 之後，便會立即對 I-8057 的 DO 通道值作更新。

步驟 2：I-8053/I-8057 的 DI/DO 通道特性如下，DI/DO 通道的數值若爲 1，則代表 DI/DO 通道目前爲低邏輯準位，I/O 模組上的 LED 燈號爲 ON，DI/DO 通道的數值若爲 0，則代表 DO 爲高邏輯準位，I/O 模組上的 LED 燈號爲 OFF。

根據 CANopen DS-401，DO 通道輸出預設值爲 0，故上個步驟執行完之後，I-8057 的 DO 值會由 0x0000 變爲 0x1234。即代表其 DO2、DO4、DO5、DO9、DO12 的 LED 燈號會由 OFF 轉 ON，且其輸出由高邏輯準位變爲低邏輯準位，其餘 DO 通道則反之。

此時 I-8053 的 DI2、DI4、DI5、DI9、DI12 的 LED 燈號會變由 OFF 轉 ON，且會測量到 DI 數值爲由 0x0000 變爲 0x1234。

I-8053 的 DI 發生變化，連帶會使得 I-8053 儲存到物件字典內的 DI 通道數值也會跟著改變，由於 I-8053 的 16 個 DI 通道是映射給第 1 個 TxPDO，且此 PDO 的傳輸型態爲 255，所以當 CAN-8423 一偵測到通道狀態的改變，就會立即傳送第 1 組 TxPDO 給 PDO 消費者，如下：

PDO 通訊協定 第 107 頁

11-bit COB-ID (181)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	1	1	0	0	0	0	0	0	1	0	2	34	12						

PDO 消費者 ← **PDO 生產者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x181
L ： 2
PDO-msg ： 34 12 00 00 00 00 00 00
因爲此處的資料長度設定爲 2，因此只有前 2 bytes 是有效的。

步驟 3：使用者可以透過傳送如下的第 2 組 RxPDO 訊息，使 I-8024 的通道 AO0 輸出 5V。(根據 CANopen DS-401，AO 通道輸出預設值爲 0x0000)

PDO 通訊協定 第 107 頁

11-bit COB-ID (301)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
0	1	1	0	0	0	0	0	0	0	1	0	8	FF	3F	00	00	00	00	00	00

PDO 生產者 ⟶ **PDO 消費者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x301

L ： 8

PDO-msg ： FF 3F 00 00 00 00 00 00

此處 8 bytes 的資料分別映射到 I-8024 的 4 個 AO 通道，每個 AO 通道佔用 2 bytes，依序分別爲通道 AO0、AO1、AO2、AO3。

因爲 CAN-8123/CAN-8223/CAN-8423 在處理類比通道數值時，是使用 2 補數/16 進制的數值表示法，因此使用者必須要透過轉換公式，將實際的類比通道數值(通道值 $_{float}$)轉換爲裝置能夠處理的格式(通道值 $_{hex}$)。

根據附錄 B 的轉換表，可知 I-8024 預設的通道值 $_{float}$ 和通道值 $_{hex}$ 範圍分別爲-10V~10V 和 0x8000(-32768)~0x7FFF(32767)。若我們欲透過 AO0 輸出 5V，透過下面的轉換公式：

$$\begin{aligned} \text{通道值}_{hex} &= \left[\frac{\text{通道值}_{float} - \text{最小通道值}_{float}}{\text{最大通道值}_{float} - \text{最小通道值}_{float}}\right] * \\ &\quad \left[\text{最大通道值}_{hex} - \text{最小通道值}_{hex}\right] + \text{最小通道值}_{hex} \\ &= \left[\frac{5\text{V} - (-10\text{V})}{10\text{V} - (-10\text{V})}\right] * \left[32767 - (-32768)\right] + (-32768) \\ &= 16383.25 \approx 16383 = 0\text{x}3\text{FFF} \end{aligned}$$

可知 PDO 訊息的前 2 bytes 需填入“FF”和“3F”。因爲此處其餘 AO 通道的輸出欲設定爲 0V，所以剩下 6 bytes 便填入“000000”。

有關轉換公式的進一步內容，請參照 6.3 節。

步驟 4：若 AI 通道所量測到的數值發生變化，此時 CAN-8423 並不會自動傳送 AI 對應的 TxPDO(第 2 組 TxPDO)給 PDO 消費者。因此，使用者需要傳送第 2 組 TxPDO 的 RTR 訊息，要求 CAN-8423 傳送 AI 通道的數值給使用者。

PDO 通訊協定 第 107 頁

11-bit COB-ID (281)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
0	1	0	1	0	0	0	0	0	0	1	1	0								

PDO 消費者 ⟶ **PDO 生產者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x281

步驟 5：CAN-8423 在收到 RTR 訊息之後，就會傳送第 2 組 TxPDO 給使用者。

PDO 通訊協定 第 107 頁

11-bit COB-ID (281)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
0	1	0	1	0	0	0	0	0	0	1	0	8	00	40	FD	FF	FD	FF	FD	FF

PDO 消費者 ⟵ **PDO 生產者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x281
L ： 8
PDO-msg ： 00 40 FD FF FD FF FD FF

這 8 bytes 分別代表爲 I-87017 的 AI0、AI1、AI2、AI3 通道所量到的 AI 數值，根據低字組先傳的原則，可知 “00 40” 代表 AI0 量到的數值爲 0x4000(16384)。據附錄 B 的轉換表，可知 I-87017 預設的通道值 $_{float}$ 和通道值 $_{hex}$ 範圍分別爲-10V~10V 和-32768~32767。透過轉換公式，可以將 0x4000 轉換爲實際的通道數值，如下：

$$\text{通道值}_{float} = \left[\frac{\text{通道值}_{hex} - \text{最小通道值}_{hex}}{\text{最大通道值}_{hex} - \text{最小通道值}_{hex}}\right] * (\text{最大通道值}_{float} - \text{最小通道值}_{float}) + (\text{最小通道值}_{float})$$

$$= \left[\frac{16384 - (-32768)}{32767 - (-32768)}\right] * (10\text{V} - (-10\text{V})) + (-10\text{V}) \approx 5.001\text{V}$$

- **事件計時器(Event Timer)功能展示**

步驟 6：此處利用 SDO 存取物件字典內主索引爲 0x1801 且子索引爲 0x05 的項目，將第 2 組 TxPDO 的事件計時器數值設定爲 1000。由於事件計時器內數值的單位爲毫秒，所以此處的 1000 即代表 1 秒。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (601)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2B	01	18	05	E8	03	00	00

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 1
n ： 2
e ： 1
s ： 1
m ： 01 18 05
d ： E8 03 00 00
16 進制的 0x03E8 即代表十進制的 1000。此外因爲 **n=2**，所以最後 2 bytes 的內容“0000”不具任何意義。

步驟 7：如果 SDO 傳輸成功，CAN-8423 會回覆以下訊息以結束 SDO 的通訊。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (581)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	01	18	05	00	00	00	00

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 3
m ： 01 18 05

步驟 8：在事件計時器的數值被改變之後，第 2 組的 TxPDO 便會每 1 秒傳送 1 次，也就是通道 AI0 所量測到的數值每 1 秒會傳送 1 次。底下是第 1 次接收到的第 2 組 TxPDO 訊息。

PDO 通訊協定 第 107 頁

11-bit COB-ID (281)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	1	0	1	0	0	0	0	0	0	1	0	8	00	40	FD	FF	FF	FF	FF	FF

PDO 消費者 ← **PDO 生產者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x281
L ： 8
PDO-msg ： 00 40 FD FF FF FF FF FF

步驟 9：底下是第 2 次接收到的第 2 組 TxPDO 訊息。

PDO 通訊協定 第 107 頁

11-bit COB-ID (281)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	1	0	1	0	0	0	0	0	0	1	0	8	06	40	FF	FF	FF	FF	FF	FF

PDO 消費者 ← **PDO 生產者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x281
L ： 8
PDO-msg ： 06 40 FF FF FF FF FF FF

0x4006 在轉換爲通道值 $_{float}$ 後，相當於 5.002V。此處 AI 的數值會出現小幅度變動的原因，可能是因爲雜訊的干擾或著其他的因素。類比輸出輸入，數值變動的範圍須參考硬體本身的規格。

步驟 10：底下是第 3 次接收到的第 2 組 TxPDO 訊息。

PDO 通訊協定 第 107 頁

11-bit COB-ID (281)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	1	0	1	0	0	0	0	0	0	1	0	8	00	40	FF	FF	FD	FF	FF	FF

PDO 消費者 ⟵ **PDO 生產者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x281
L ： 8
PDO-msg ： 00 40 FF FF FD FF FF FF

步驟 11：將事件計數器的數值設定回 0，以結束事件計數器的功能測試。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (601)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2B	01	18	05	00	00	00	00

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 1
n ： 2
e ： 1
s ： 1
m ： 00 18 05
d ： 00 00 00 00

11-bit COB-ID (581)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	00	18	05	00	00	00	00

SDO 用戶端 ← **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 3

m ： 00 18 05

- **非循環同步 RxPDO 的功能展示 (傳輸型態爲 0)**

步驟 12：設定第 1 組 RxPDO 的傳輸型態爲 0。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (601)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2F	00	14	02	00	00	00	00

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 1
n ： 3
e ： 1
s ： 1
m ： 00 14 02
d ： 00

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (581)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	00	14	02	00	00	00	00

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 3
m ： 00 14 02

步驟 13：使用第 1 組 RxPDO 通知 CAN-8423，準備將 I-8057 的 DO 數值變更爲 0x5678。

PDO 通訊協定 第 107 頁

11-bit COB-ID (201)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
0	1	0	0	0	0	0	0	0	0	1	0	8	78	56	00	00	00	00	00	00

PDO 生產者 → **PDO 消費者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x201
L ： 8
PDO-msg ： 78 56 00 00 00 00 00 00

步驟 14：在進行上個步驟之後，I-8057 的 DO 通道數值並不會馬上改變，這是因爲此時第 1 組的 RxPDO，其傳輸型態被設定爲 0 (非循環同步)。如果要使 I-8057 的 DO 通道數值變更爲 0x5678，使用者可傳送一 SYNC 物件給 CAN-8423，以觸發前一個步驟所傳輸的 RxPDO，如下：

SYNC 通訊協定

11-bit COB-ID (80)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
0	0	0	1	0	0	0	0	0	0	0	0	0								

SYNC 生產者 → **SYNC 消費者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x80

SYNC 訊息的格式是固定的，如上所示。每一台裝置的 SYNC 訊息，其 COB-ID 均可依據實際需求，透過 SDO 通訊協定做修改，使用者可參考本手冊中有關 SDO 下載的部份。此外 SYNC 的通訊協定爲生產者/消費者的架構。

步驟 15：當先前傳送的第 1 組 RxPDO 被 SYNC 物件觸發之時，I-8057 的 DO 數值便會立刻改變爲 0x5678，因此 I-8053 的 DI 數值也會跟著改變，此時使用者就會收到 CAN-8423 傳來的第 1 組 TxPDO。

PDO 通訊協定 第 107 頁

11-bit COB-ID (181)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
0	0	1	1	0	0	0	0	0	0	1	0	2	78	56						

PDO 消費者 ← **PDO 生產者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x181
L ： 2
PDO-msg ： 78 56 00 00 00 00 00 00

步驟 16：把第 1 組 RxPDO 的傳輸型態設回 255。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (601)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2F	00	14	02	FF	00	00	00

SDO 用戶端 → **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 1
n ： 3
e ： 1
s ： 1
m ： 00 14 02
d ： FF 00 00 00

11-bit COB-ID (581)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	00	14	02	00	00	00	00

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 3

m ： 00 14 02

- **非循環同步 TxPDO 的功能展示 (傳輸型態爲 0)**

步驟 17：設定第 1 組 TxPDO 的傳輸型態爲 0。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (601)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2F	00	18	02	00	00	00	00

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 1
n ： 3
e ： 1
s ： 1
m ： 00 18 02
d ： 00 00 00 00

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (581)											RTR	資料	8-byte 資料(byte)							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	00	18	02	00	00	00	00

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 3
m ： 00 18 02

步驟 18：利用第 1 組 RxPDO 將 I-8057 的 DO 值變更爲 0x90AB。

PDO 通訊協定 第 107 頁

11-bit COB-ID (201)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
0	1	0	0	0	0	0	0	0	0	1	0	8	AB	90	00	00	00	00	00	00

PDO 生產者 ⟶ **PDO 消費者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x201
L ： 8
PDO-msg ： AB 90 00 00 00 00 00 00

步驟 19：因爲第 1 組 TxPDO 的傳輸型態被設定爲 0，此時即使 DI 測量到的數值發生變化，CAN-8423 也不會傳送第 1 組的 TxPDO 給使用者。此時若是 CAN-8423 收到 SYNC 訊後，則第 1 組 TxPDO 的傳送就會被觸發。

SYNC 通訊協定

11-bit COB-ID (80)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
0	0	0	1	0	0	0	0	0	0	0	0	0								

SYNC 生產者 ⟶ **SYNC 消費者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x80

步驟 20：在 CAN-8423 收到 SYNC 訊息之後，便會觸發第 1 組 TxPDO 的傳送，此時使用者便可以接收到由 CAN-8423 傳來的第 1 組 TxPDO。

PDO 通訊協定 第 107 頁

11-bit COB-ID (181)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	1	1	0	0	0	0	0	0	1	0	2	AB	90						

PDO 消費者 ← **PDO 生產者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x181
L ： 2
PDO-msg ： AB 90 00 00 00 00 00 00

步驟 21：再傳送一次 SYNC 訊息給 CAN-8423。

SYNC 通訊協定

11-bit COB-ID (80)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	0	1	0	0	0	0	0	0	0	0	0								

SYNC 生產者 → **SYNC 消費者 (CAN-8123/CAN-8223/CAN-8423)**

SYNC COB-ID ： 0x80

步驟 22：若此時 DI 的值沒有發生變化，則 CAN-8423 即使收到 SYNC 訊息，也不會觸發第一組 TxPDO 的傳送。如果此時第 1 組 TxPDO 的傳輸型態被設定爲 1，則不管 DI 的值有沒有被改變，CAN-8423 在收到 SYNC 訊息之後，都會觸發第一組 TxPDO 的傳送。傳輸型態 0 或 1，這兩者最大的差別就在於 SYNC 訊息對其的觸發行爲。

- **循環同步 TxPDO 的功能展示 (傳輸型態爲 3)**

步驟 23：設定第 1 組 TxPDO 的傳輸型態爲 3。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (601)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2F	00	18	02	03	00	00	00

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 1
n ： 3
e ： 1
s ： 1
m ： 00 18 02
d ： 03 00 00 00

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (581)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	00	18	02	00	00	00	00

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 3
m ： 00 18 02

步驟 24：利用第 1 組 RxPDO 將 I-8057 的 DO 值變更爲 0xCDEF。

PDO 通訊協定 第 107 頁

11-bit COB-ID (201)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
0	1	0	0	0	0	0	0	0	0	1	0	8	EF	CD	00	00	00	00	00	00

PDO 生產者 → **PDO 消費者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x201
L ： 8
PDO-msg ： EF CD 00 00 00 00 00 00

步驟 25：於此傳送 3 個 SYNC 訊息給 CAN-8423，因爲目前第 1 組 TxPDO 的傳輸型態被設定爲 3，所以 CAN-8423 在收到 3 個 SYNC 訊息，便會觸發第 1 組 TxPDO 的傳送。

SYNC 通訊協定

11-bit COB-ID (80)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
0	0	0	1	0	0	0	0	0	0	0	0	0								

SYNC 生產者 → **SYNC 消費者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x80

步驟 26：CAN-8423 在收到 3 個 SYNC 訊息之後，便會觸發第 1 組 TxPDO 的傳送，此時使用者便可以收到由 CAN-8423 所傳來的第 1 組 TxPDO。

11-bit COB-ID (181)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
0	0	1	1	0	0	0	0	0	0	1	0	2	EF	CD						

PDO 消費者 ← **PDO 生產者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x181
L ： 2
PDO-msg ： EF CD 00 00 00 00 00 00

- **同步、唯遠端傳輸要求 TxPDO 的功能展示 (傳輸型態 252)**

步驟 27：設定第 1 組 TxPDO 的傳輸型態爲 252，若 TxPDO 被設定爲此一傳輸型態，則在收到 SYNC 訊息時，才會更新 PDO 的內容，並且在收到此 TxPDO 的 RTR 訊息之後，才會觸發此 TxPDO 的傳送。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (601)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2F	00	18	02	FC	00	00	00

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 1
n ： 3
e ： 1
s ： 1
m ： 00 18 02
d ： FC 00 00 00

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (581)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	00	18	02	00	00	00	00

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 3
m ： 00 18 02

步驟 28：利用第 1 組 RxPDO 將 I-8057 的 DO 值變更爲 0x1234。

PDO 通訊協定 第 107 頁

11-bit COB-ID (201)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	1	0	0	0	0	0	0	0	0	1	0	8	34	12	00	00	00	00	00	00

PDO 生產者 ⟶ **PDO 消費者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x201
L ： 8
PDO-msg ： 34 12 00 00 00 00 00 00

步驟 29：此處傳送第 1 組 TxPDO 的 RTR 訊息給 CAN-8423，要求其回傳第一組 TxPDO。因爲第 1 組 TxPDO 的傳輸型態被設定爲 252，所以必須要等到 CAN-8423 收到 RTR 訊息之後，第 1 組 TxPDO 的傳送才會被觸發。

PDO 通訊協定 第 107 頁

11-bit COB-ID (181)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	1	1	0	0	0	0	0	0	1	1	0								

PDO 消費者 ⟶ **PDO 生產者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x181

步驟 30：使用者在收到 CAN-8423 傳來的第 1 組 TxPDO 訊息之後，可以跟 I-8053 面板上的 LED 燈號做比較，於此處會發現 PDO 內記錄的爲更新前的 DO 數值。

PDO 通訊協定 第 107 頁

11-bit COB-ID (181)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
0	0	1	1	0	0	0	0	0	0	1	0	2	EF	CD						

PDO 消費者 ← **PDO 生產者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x181
L ： 2
PDO-msg ： EF CD 00 00 00 00 00 00

步驟 31：傳送 SYNC 訊息給 CAN-8423，以更新第一組 TxPDO 內記錄的值。

SYNC 通訊協定

11-bit COB-ID (80)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
0	0	0	1	0	0	0	0	0	0	0	0	0								

SYNC 生產者 → **SYNC 消費者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x80

步驟 32：此處再傳送一次第 1 組 TxPDO 的 RTR 訊息給 CAN-8423，要求其回傳第一組 TxPDO。

PDO 通訊協定 第 107 頁

11-bit COB-ID (181)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
0	0	1	1	0	0	0	0	0	0	1	1	0								

PDO 消費者 ⟶ **PDO 生產者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x181

步驟 33：此時使用者在收到 CAN-8423 傳來的第 1 組 TxPDO 訊息之後，就會發現 PDO 內記錄的 DO 數值和 I-8053 面板上的 LED 燈號相符合。

PDO 通訊協定 第 107 頁

11-bit COB-ID (181)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
0	0	1	1	0	0	0	0	0	0	1	0	2	34	12						

PDO 消費者 ⟵ **PDO 生產者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x181
L ： 2
PDO-msg ： 34 12 00 00 00 00 00 00

- **非同步、唯遠端傳輸要求 TxPDO 的功能展示 (傳輸型態 253)**

步驟 34：設定第 1 組 TxPDO 的傳輸型態爲 253。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (601)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2F	00	18	02	FD	00	00	00

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 1
n ： 3
e ： 1
s ： 1
m ： 00 18 02
d ： FD 00 00 00

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (581)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	00	18	02	00	00	00	00

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 3
m ： 00 18 02

步驟 35：利用第 1 組 RxPDO 將 I-8057 的 DO 值變更爲 0x5678。

PDO 通訊協定 第 107 頁

11-bit COB-ID (201)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	1	0	0	0	0	0	0	0	0	1	0	8	78	56	00	00	00	00	00	00

PDO 生產者 ⟶ **PDO 消費者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x201
L ： 8
PDO-msg ： 78 56 00 00 00 00 00 00

步驟 36：因爲第 1 組 TxPDO 的傳輸型態被設定爲 253，所以 CAN-8423 僅在收到此 TxPDO 的 RTR 訊息之後，才會更新 TxPDO 的內容與對外傳送。於下，先傳送第 1 組 TxPDO 的 RTR 訊息給 CAN-8423，CAN-8423 在收到 RTR 訊息之後會回傳第一組 TxPDO，內容爲 I-8053 最新的 DI 數值。

PDO 通訊協定 第 107 頁

11-bit COB-ID (181)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	1	1	0	0	0	0	0	0	1	1	0								

PDO 消費者 ⟶ **PDO 生產者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x181

PDO 通訊協定 第 107 頁

11-bit COB-ID (181)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	1	1	0	0	0	0	0	0	1	0	2	78	56						

PDO 消費者 ⟵ **PDO 生產者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x181
L ： 2
PDO-msg ： 78 56 00 00 00 00 00 00

步驟 37：把第 1 組 TxPDO 的傳輸型態設回 255。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (601)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2F	00	18	02	FF	00	00	00

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 1
n ： 3
e ： 1
s ： 1
m ： 00 18 02
d ： FF 00 00 00

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (581)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	00	18	02	00	00	00	00

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 3
m ： 00 18 02

- **對 DI/AI/DO/AO 通道使用動態 PDO 映射**

步驟 38：底下將示範使用者要如何建立新的 TxPDO，於下將建立第 5 組的 TxPDO(使用者可自行選用空餘的 TxPDO)。首先必須決定 TxPDO 的 COB-ID，即設定此 TxPDO 的通訊參數(主索引 0x1804)的 COB-ID 項目(子索引 0x01)，另外需確認此項目的第 31 個 bit 爲 1，才可以更改此項目，若此 bit 爲 0，則無法進行更改，必須將此項目的第 31 個 bit 設定爲 1，才可以進行接下來的修改動作，在修改 COB-ID 項目的時後，需再把第 31 個 bit 設定爲 0。(可參考 5.2.1 節)

使用者在設定 COB-ID 時，必須確定所使用的 COB-ID 不是 CANopen 規範保留的 COB-ID，也沒有被裝置上的其他通訊物件使用，或著和網路中其他裝置的通訊物件衝突。因爲第 5 組 TxPDO 的通訊參數的 COB-ID 項目，其第 31 bit 預設爲 1，所以可以直接修改通訊參數的 COB-ID 項目，於此將其設定爲 0x182。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (601)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	23	04	18	01	82	01	00	00

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 1
n ： 0
e ： 1
s ： 1
m ： 04 18 01
d ： 82 01 00 00

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (581)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	04	18	01	00	00	00	00

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 3
m ： 04 18 01

步驟 39：於此處設定第 5 組 TxPDO 的映射參數(主索引 0x1A04)。在開始設定 PDO 的映射參數之前，首先必須要確認物件字典中主索引爲 0x1A04 且子索引爲 0x00 的項目之數值爲 0，如果此數值不爲 0，則對映射參數所做的設定便會失敗。於此處，由於主索引爲 0x1A05 的物件並沒有被使用過，其子索引 0x00 項目的數值預設便爲 0，所以可以直接對此 PDO 的映射參數作設定。於下將設定第 5 組 TxPDO 映射參數的子索引 0x01。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (601)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	23	04	1A	01	08	01	00	60

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 1
n ： 0
e ： 1
s ： 1
m ： 04 1A 01
d ： 08 01 00 60

“08 01 00 60”即代表 PDO 映射參數(0x1A04)的子索引 0x01 映射到主索引爲 0x6000 且子索引爲 0x01 的項目，其長度爲 8 bits。也就是說第 5 組 TxPDO 訊息的前 2 bytes 會映射到 I-8053 的 DI0~DI7，使用者可以參照本節一開始所提過的“標準化裝置描述文件區域”一圖。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (581)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	04	1A	01	00	00	00	00

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 3
m ： 04 1A 01

步驟 40：於下設定第 5 組 TxPDO 映射參數(主索引 0x1A04)的子索引 0x02 和 0x03。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (601)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	23	04	1A	02	08	02	00	60

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 1
n ： 0
e ： 1
s ： 1
m ： 04 1A 02
d ： 08 02 00 60

“08 02 00 60”即代表 PDO 映射參數(0x1A04)的子索引 0x02 映射到主索引爲 0x6000 且子索引爲 0x02 的項目，其長度爲 8 bits。也就是說第 5 組 TxPDO 訊息的第 3~4 bytes 會映射到 I-8053 的 DI8~DI15。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (581)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	0	1	1	1	0	1	1	1	0	1	0	8	60	04	1A	02	00	00	00	00

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 3
m ： 04 1A 02

11-bit COB-ID (601)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	23	04	1A	03	10	01	01	64

SDO 用戶端 → **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 1
n ： 0
e ： 1
s ： 1
m ： 04 1A 03
d ： 10 01 01 64

“10 01 01 64”即代表 PDO 映射參數(0x1A04)的子索引 0x03 映射到主索引爲 0x6401 且子索引爲 0x01 的項目，其長度爲 16 bits。也就是說第 5 組 TxPDO 訊息的第 5~6 bytes 會映射到 I-87017 的 AI0，使用者可以參照本節一開始所提過的“標準化裝置描述文件區域”一圖。此外在 CAN-8123/CAN-8223/CAN-8423 內，所有類比通道的數值其長度皆爲 2 bytes。

11-bit COB-ID (581)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	04	1A	03	00	00	00	00

SDO 用戶端 ← **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 3
m ： 04 1A 03

步驟 41：設定 PDO 映射參數的最後一個步驟就是在其子索引 0x00 的地方填入此 PDO 內含的映射個數，於此處在主索引 0x1A04 且子索引爲 0x00 的地方填入 3。3 代表第 5 組的 TxPDO 內含 3 個映射，分別指向主索引爲 0x6000 且子索引爲 0x01 處、主索引爲 0x6000 且子索引爲 0x02 處、主索引爲 0x6401 且子索引爲 0x01 處。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (601)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2F	04	1A	00	03	00	00	00

SDO 用戶端 → **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 1
n ： 3
e ： 1
s ： 1
m ： 05 1A 00
d ： 03 00 00 00

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (581)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	04	1A	00	00	00	00	00

SDO 用戶端 ← **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 3
m ： 04 1A 00

步驟 42：底下將示範使用者要如何建立新的 RxPDO 通訊物件，於此處使用第 5 組的 RxPDO(使用者可自行選用空餘的 RxPDO)。RxPDO 的設定方式和 TxPDO 相同，於此處便不再贅述。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (bit)											RTR	資料長度	8-byte 資料(byte)							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	23	04	14	01	02	02	00	00

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 1
n ： 0
e ： 1
s ： 1
m ： 04 14 01
d ： 02 02 00 00
將第 5 組的 RxPDO COB-ID 設定爲 0x202。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (bit)											RTR	資料長度	8-byte 資料(byte)							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	04	14	01	00	00	00	00

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 3
m ： 04 14 01

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (601)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	23	04	16	01	08	01	00	62

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 1
n ： 0
e ： 1
s ： 1
m ： 04 16 01
d ： 08 01 00 62

“08 01 00 62” 即代表 PDO 映射參數(0x1604)的子索引 0x01 映射到主索引爲 0x6200 且子索引爲 0x01 的項目，其長度爲 8 bits。也就是說第 5 組 RxPDO 訊息的前 2 bytes 會映射到 I-8057 的 DO0~DO7。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (581)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	04	16	01	00	00	00	00

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 3
m ： 04 16 01

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (601)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	23	04	16	02	08	02	00	62

SDO 用戶端 → **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 1
n ： 0
e ： 1
s ： 1
m ： 04 16 02
d ： 08 02 00 62

“08 02 00 62”即代表 PDO 映射參數(0x1604)的子索引 0x02 映射到主索引爲 0x6200 且子索引爲 0x02 的項目，其長度爲 8 bits。也就是說第 5 組 RxPDO 訊息的第 3~4 byte 會映射到 I-8057 的 DO0~DO7。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (581)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	04	16	02	00	00	00	00

SDO 用戶端 ← **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 3
m ： 04 16 02

11-bit COB-ID (601)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	23	04	16	03	10	01	11	64

SDO 用戶端 → **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 1
n ： 0
e ： 1
s ： 1
m ： 04 16 03
d ： 10 01 11 64

“10 01 11 64”即代表 PDO 映射參數(0x1604)的子索引 0x03 映射到主索引爲 0x6411 且子索引爲 0x01 的項目，其長度爲 16 bits。也就是說第 5 組 RxPDO 訊息的第 5~6 bytes 會映射到 I-8024 的 AO0。

11-bit COB-ID (581)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	04	16	03	00	00	00	00

SDO 用戶端 ← **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 3
m ： 04 16 03

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (601)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2F	04	16	00	03	00	00	00

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 1
n ： 3
e ： 1
s ： 1
m ： 04 16 00
d ： 03 00 00 00
3 代表第 5 組的 RxPDO 內含 3 個映射。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (581)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	04	16	00	00	00	00	00

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 3
m ： 04 16 00

步驟 43：透過底下 PDO 訊息的傳送，通知 CAN-8423 分別將 I-8057 的 DO0~DO15 和 I-8024 的 AO0 設定爲 0x90AB 和 0V。

PDO 通訊協定 第 107 頁

11-bit COB-ID (202)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	1	0	0	0	0	0	0	0	1	0	0	8	AB	90	00	00	00	00	00	00

PDO 生產者 ⟶ **PDO 消費者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x202

PDO-msg ： AB 90 00 00 00 00 00 00

這個 PDO 訊息只有前 4 bytes 爲有效資料。第 1~2 bytes 代表 I-8057 的 DO0~DO15 要設定爲 0x90AB，第 3~4 bytes 代表 I-8024 的 AO0 要設定爲 0x0000。

步驟 44：因爲 DI 的數值發生了變化，所以使用者於此時會接收到第 1 組和第 5 組的 TxPDO。

PDO 通訊協定 第 107 頁

11-bit COB-ID (181)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	1	1	0	0	0	0	0	0	1	0	2	AB	90						

PDO 消費者 ⟵ **PDO 生產者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x181

L ： 2

PDO-msg ： AB 90 00 00 00 00 00 00

“AB 90”代表 I-8053 的 DI0~DI15 目前的數值爲 0x90AB。

11-bit COB-ID (182)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	1	1	0	0	0	0	0	1	0	0	4	AB	90	FF	FF				

PDO 消費者 ← **PDO 生產者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x182

L ： 4

PDO-msg ： AB 90 FF 3F 00 00 00 00

前 2 bytes 代表 I-8053 的 DI0~DI15 目前的數值爲 0x90AB。第 3~4 bytes 代表 I-87017 的 AI0 目前的數值爲 0xFFFF，將此值轉換之後可以得知 AI0 目前測量到的電壓爲-0.001V。

5.3 EMCY 通訊集

5.3.1 EMCY COB-ID 參數

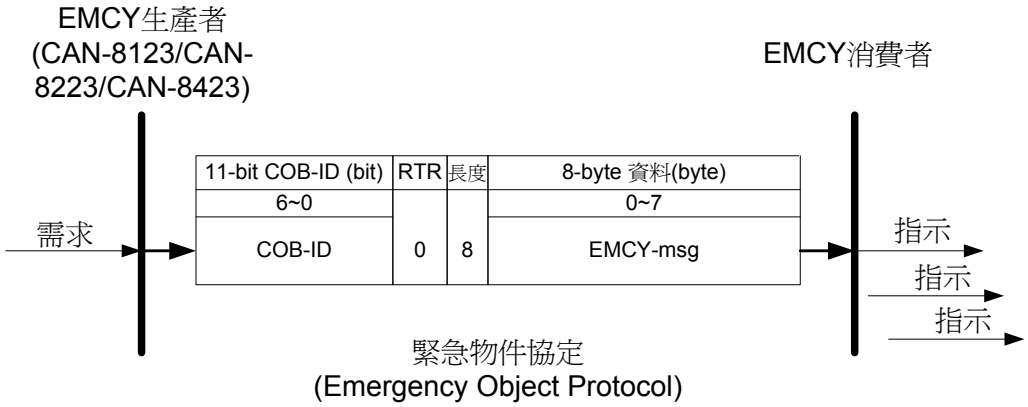
EMCY 訊息的 COB-ID 和 PDO 訊息的 COB-ID 十分類似，使用者可以直接使用裝置開機之後的預設值，或著透過 SDO 訊息來對其作更改。EMCY 訊息的 COB-ID 儲存在物件字典中主索引爲 0x1014 的物件內，其資料格式如下表所示：

Bit 編號	值	代表的意義
31 (MSB)	0	EMCY 存在 (EMCY 爲有效的，valid)
	1	EMCY 不存在 (EMCY 是無效的，invalid)
30	0	被保留 (此 bit 的值永爲 0)
29	0	11-bit ID (CAN 2.0A)
	1	29-bit ID (CAN 2.0B)
28-11	0	若 bit 29=0，此欄位的數值便爲 0
	x	若 bit 29=1：則此欄位就是 29bits COB-ID 內的第 28~11 bits
10-0 (LSB)	x	COB-ID 內的第 10~0 bits

在使用 EMCY 訊息之前，必須先確認 0x1014 這個物件的 bit 31 爲 0，也就是這個裝置上的 EMCY 訊息是有效的。

5.3.2 EMCY 通訊協定

當裝置內產生某些特定的內部錯誤，亦即發生 EMCY 事件，此時便會觸發 EMCY 訊息的傳送。在物件字典內主索引 0x1003 的物件中，會記錄裝置上曾經發生過的 EMCY 事件，使用者可以透過檢視這個物件來了解裝置發生錯誤的記錄。CAN-8123/CAN-8223/CAN-8423 在主索引 0x1003 的物件內，最多可儲存 5 筆記錄，由子索引 0x01 到子索引 0x05，其儲存的方式爲先進先出，最近的 EMCY 事件會儲存在子索引 0x01。底下將介紹傳送 EMCY 訊息所使用的通訊協定：



COB-ID ： 使用者可以自行定義 EMCY 訊息所使用的 COB-ID，定義的方式和 PDO 相同。EMCY 訊息預設的 COB-ID 爲 4-bit 的功能碼“0001”加上 7-bit 的節點 ID。

EMCY-msg ： 內含緊急物件資料(emergency object data)，此欄位所記錄的資料可以用來識別裝置上發生錯誤的種類。

緊急物件資料的資料格式如下表所示：

Byte	0	1	2	3	4	5	6	7
內容	緊急錯誤碼 (emergency error code)		錯誤暫存器 (error register)	製造商特定錯誤區域 (manufacturer specific error field)				

錯誤暫存器長度爲 8-bit，其內每一 bit 所代表的意義如下表所示：

Bit 編號	代表的意義
0	一般性的錯誤
1	電流錯誤
2	電壓錯誤
3	溫度錯誤
4	通訊錯誤(過載，或著裝置正處於錯誤狀態)
5	於裝置描述文件內定義
6	保留(數值永爲 0)
7	製造商自定義

CAN-8123/CAN-8223/CAN-8423 僅支援上表中的 bit 0、bit 4 和 bit 7。

針對 CAN-8123/CAN-8223/CAN-8423，緊急物件資料內不同的內容所代表的意義如下表所示：

緊急錯誤碼	錯誤暫存器	製造商特定錯誤區域		代表的意義
		前 2 bytes	後 3 bytes	
00 00	00	00 00	00 00 00	錯誤重置，或著沒有錯誤發生
10 00	81	00 01	00 00 00	CAN 控制器發生錯誤
50 00	81	00 02	00 00 00	EEPROM 存取錯誤
50 00	81	00 03	00 00 00	COM 埠存取錯誤
81 10	11	00 04	00 00 00	軟體的 Rx 緩衝區過載
81 10	11	00 05	00 00 00	軟體的 Tx 緩衝區過載
81 10	11	00 06	00 00 00	CAN 控制器過載
81 30	11	00 07	00 00 00	生存守衛事件錯誤
81 40	11	00 08	00 00 00	裝置已由 bus off 狀態下復原
82 10	11	00 09	00 00 00	PDO 資料長度錯誤
FF 00	80	00 0A	00 00 00	裝置需要重新啓動節點或著重新啓動通訊

裝置在產生 EMCY 訊息之後，會把 EMCY 訊息之中的緊急錯誤碼和製造商特定錯誤區域的前 2 bytes 存到主索引爲 0x1003 且子索引爲 0x01 的項目之中，如下：

Byte：	3	2	1	0
	製造商特定錯誤區域的前 2 bytes		緊急錯誤碼	

另外在主索引 0x1001 的物件內，則會記錄錯誤暫存器的內容。使用者可以查閱物件字典內的這兩個地方，以檢視裝置上發生錯誤的記錄，進一步了解 CAN-8123/CAN-8223/CAN-8423 爲何發出緊急訊息。

EMCY 通訊範例

底下範例所使用的硬體架構與 PDO 通訊範例相同，使用者可參照 5.2.3 節。

步驟 1：爲了產生 EMCY 事件，此處傳送一 PDO 訊息給 CAN-8423，設定此 PDO 的 COB-ID 爲 0x201，資料長度爲 0x01。

PDO 通訊協定 第 107 頁

11-bit COB-ID (201)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	1	0	0	0	0	0	0	0	0	1	0	1	00							

PDO 生產者 ⟶ **PDO 消費者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x201
L ： 1
PDO-msg ： 00

步驟 2：CAN-8423 在收到上一個步驟的 PDO 訊息之後，根據這個 PDO 的 COB-ID，會判斷這個 PDO 爲第 1 組 RxPDO，且會因爲其內所記錄的資料長度和 PDO 映射參數內所記錄的不吻合，所以會對外回應一緊急訊息，如下。

EMCY 通訊協定 第 149 頁

11-bit COB-ID (81)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	0	1	0	0	0	0	0	0	1	0	8	10	82	11	09	00	00	00	00

EMCY 消費者 ⟵ **EMCY 生產者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x81
EMCY-msg ： 10 82 11 09 00 00 00 00
前 2 bytes 爲緊急錯誤碼。第 3 個 byte 爲錯誤暫存器，數值“11”代表通訊錯誤或一般性的錯誤。後 5 bytes 爲製造商特定錯誤區域。這 8 bytes 的緊急物件資料所代表的意義爲裝置所接收到的 PDO 資料長度與 PDO 映射參數內所記錄的資料長度不吻合。

步驟 3：於此處，使用者可以傳送以下的 SDO 訊息，以讀取裝置的物件字典內主索引爲 0x1003 且子索引爲 0x01 的項目，使用者可以獲得儲存在這個項目內的緊急錯誤碼和製造商特定錯誤區域的前 2 bytes。

初始 SDO 上傳協定 第 67 頁

11-bit COB-ID (601)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	40	03	10	01	00	00	00	00

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 2
m 03 10 01

步驟 4：CAN-8423 會回應以下訊息，內含主索引爲 0x1003 且子索引爲 0x01 的項目。其內容會和最近一次所收到的 EMCY 訊息相吻合。

初始 SDO 上傳協定 第 67 頁

11-bit COB-ID (581)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	43	03	10	01	10	82	09	00

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 2
n ： 0
e ： 1
s ： 1
m ： 03 10 01
d ： 10 82 09 00

步驟 5：於此處，傳送以下的 SDO 訊息，讀取裝置的物件字典內主索引爲 0x1001 的物件，以檢查此處錯誤暫存器的數值和最近一次收到的 EMCY 訊息是否吻合。

初始 SDO 上傳協定 第 67 頁

11-bit COB-ID (601)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	40	01	10	00	00	00	00	00

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 2
m ： 01 10 00

步驟 6：CAN-8423 所回傳的 SDO 訊息中，錯誤暫存器的數值爲 0x11，代表通訊錯誤或一般性的錯誤，與最近一次所收到的 EMCY 訊息相吻合。

初始 SDO 上傳協定 第 67 頁

11-bit COB-ID (581)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	4F	01	10	00	11	00	00	00

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 2
n ： 3
e ： 1
s ： 1
m ： 01 10 00
d ： 11 00 00 00

步驟 7：於此步驟中，傳送一 COB-ID 爲 0x201 的 RxPDO 訊息給 CAN-8423，並於此訊息中註記資料長度爲 0x02，此訊息的作用爲使 DO0~DO15 輸出 0x0000。因爲此 PDO 的資料格式是正確的，CAN-8423 在收到這個訊息之後，會發出一錯誤重置的 EMCY 訊息，代表先前的錯誤已被排除。

另外因爲 DO0~DO15 的值原本就是 0x0000，DI0~DI15 所測量到的數值沒有變化，所以此時並不會觸發 CAN-8423 傳送第 1 組的 TxPDO。

PDO 通訊協定 第 107 頁

11-bit COB-ID (201)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
0	1	0	0	0	0	0	0	0	0	1	0	2	00	00						

PDO 消費者 ⟵ **PDO 生產者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x201
L ： 2
PDO-msg ： 00 00 00 00 00 00 00 00

EMCY 通訊協定 第 149 頁

11-bit COB-ID (81)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
0	0	0	1	0	0	0	0	0	0	1	0	8	00	00	00	00	00	00	00	00

EMCY 消費者 ⟵ **EMCY 生產者 (CAN-8123/CAN-8223/CAN-8423)**

EMCY-msg ： 00 00 00 00 00 00 00 00
錯誤重置 EMCY 訊息的資料欄位爲“00 00 00 00 00 00 00 00”，即代表 CAN-8423 目前沒有任何的錯誤。

步驟 9：於此處，傳送以下的 SDO 訊息，以讀取裝置的物件字典內主索引爲 0x1003 且子索引爲 0x01 的項目，使用者可以檢視儲存在這個項目內的緊急錯誤碼和製造商特定錯誤區域的前 2 bytes，其內容會和最近一次所收到的 EMCY 訊息相吻合。

初始 SDO 上傳協定 第 67 頁

11-bit COB-ID (601)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	40	03	10	01	00	00	00	00

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 2
m 03 10 01

初始 SDO 上傳協定 第 67 頁

11-bit COB-ID (581)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	43	03	10	01	00	00	00	00

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 1
n ： 0
e ： 1
s ： 1
m ： 03 10 01
d ： 00 00 00 00

步驟 10：於此處，傳送以下的 SDO 訊息，以讀取裝置的物件字典內主索引爲 0x1003 且子索引爲 0x02 的項目，使用者可以檢視儲存在這個項目內的緊急錯誤碼和製造商特定錯誤區域的前 2 bytes，其內容會和上上一次所收到的 EMCY 訊息相吻合。

初始 SDO 上傳協定 第 67 頁

11-bit COB-ID (601)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	40	03	10	02	00	00	00	00

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 2
m ： 03 10 02

初始 SDO 上傳協定 第 67 頁

11-bit COB-ID (581)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	43	03	10	02	10	82	09	00

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 1
n ： 0
e ： 1
s ： 1
m ： 03 10 02
d ： 10 82 09 00

步驟 11：於此處，傳送以下的 SDO 訊息，讀取裝置的物件字典內主索引爲 0x1001 的物件，使用者會發現目前錯誤暫存器的數值爲 0x00，和最近一次收到的 EMCY 訊息相吻合。

初始 SDO 上傳協定 第 67 頁

11-bit COB-ID (601)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	40	01	10	00	00	00	00	00

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 2
m 01 10 00

初始 SDO 上傳協定 第 67 頁

11-bit COB-ID (581)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	4F	01	10	00	00	00	00	00

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

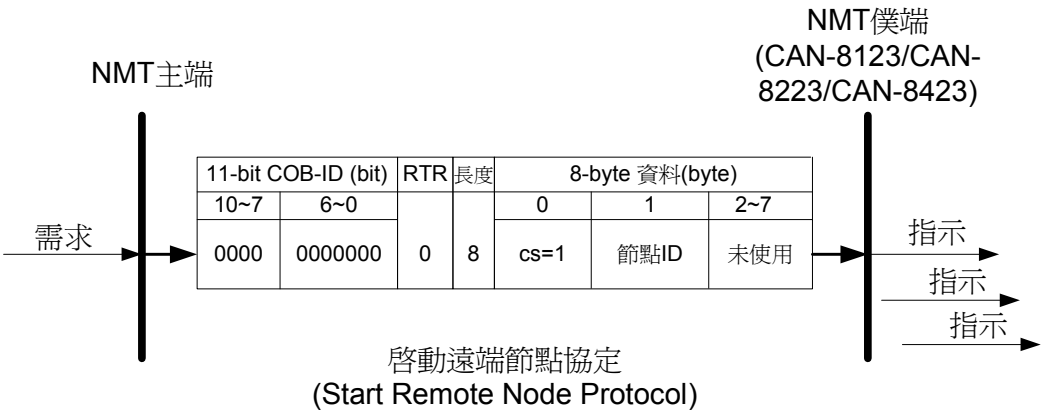
ccs ： 1
n 2
e 1
s 1
m ： 01 10 00
d ： 00 00 00 00

5.4 NMT 通訊集

5.4.1 模組控制協定

NMT 主端可以利用模組控制協定(Module Control Protocol)來改變 NMT 僕端的 NMT 狀態，底下將詳細介紹模組控制協定的細節，並在本節的末尾利用範例來說明模組控制協定的使用方式。

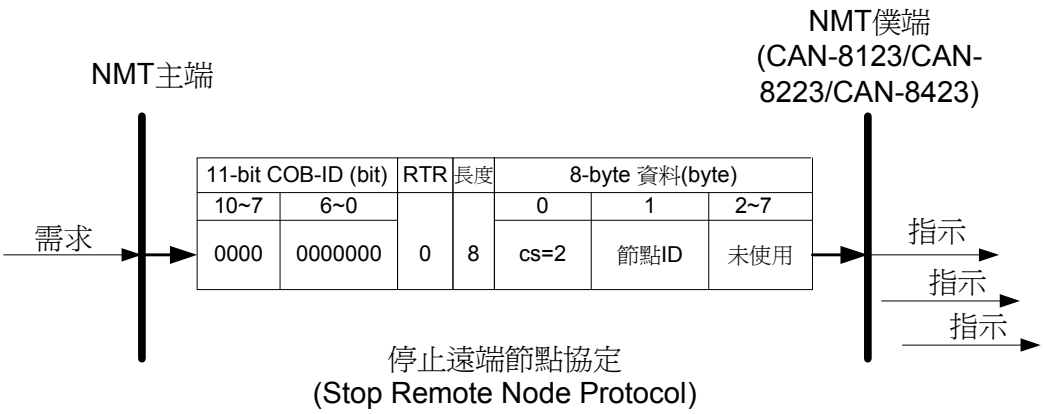
<u>啓動遠端節點協定 (Start Remote Node Protocol)</u>



cs ：NMT 命令識別符
1：啓動(start)

節點 **ID** ：NMT 僕端裝置的節點 ID。

停止遠端節點協定 (Stop Remote Node Protocol)

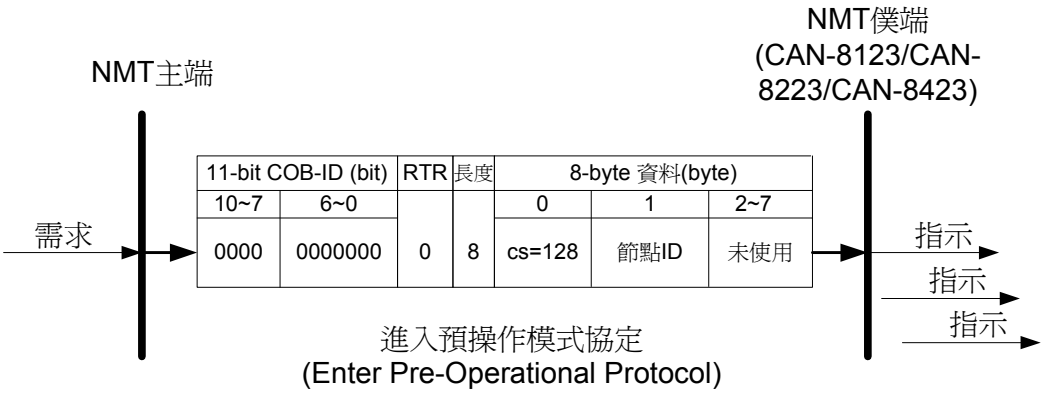


11-bit COB-ID (bit)		RTR	長度	8-byte 資料(byte)		
10~7	6~0			0	1	2~7
0000	0000000	0	8	cs=2	節點ID	未使用

cs ：NMT 命令識別符
2：停止(stop)

節點 **ID** ：NMT 僕端裝置的節點 ID。

進入預操作狀態協定 (Enter Pre-Operational Protocol)

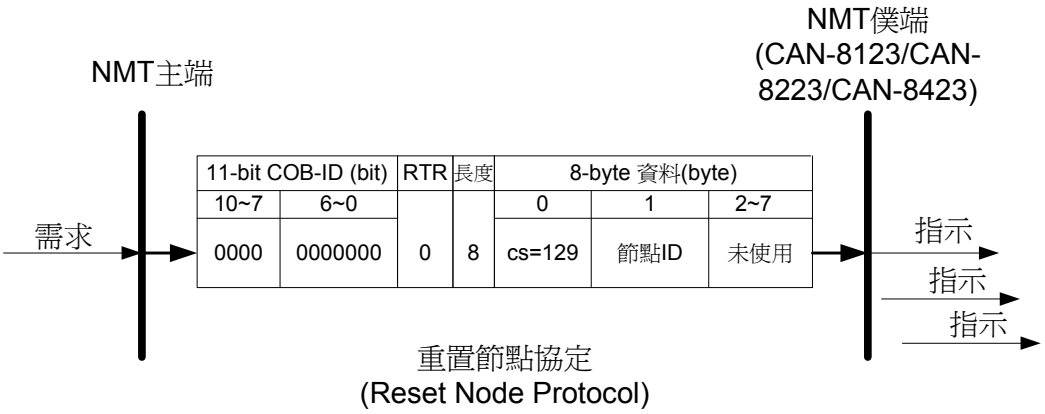


11-bit COB-ID (bit)		RTR	長度	8-byte 資料(byte)		
10~7	6~0			0	1	2~7
0000	0000000	0	8	cs=128	節點ID	未使用

cs ：NMT 命令識別符
128：進入預操作(PRE-OPERATIONAL)狀態

節點 **ID** ：NMT 僕端裝置的節點 ID。

重置節點協定 (Reset Node Protocol)

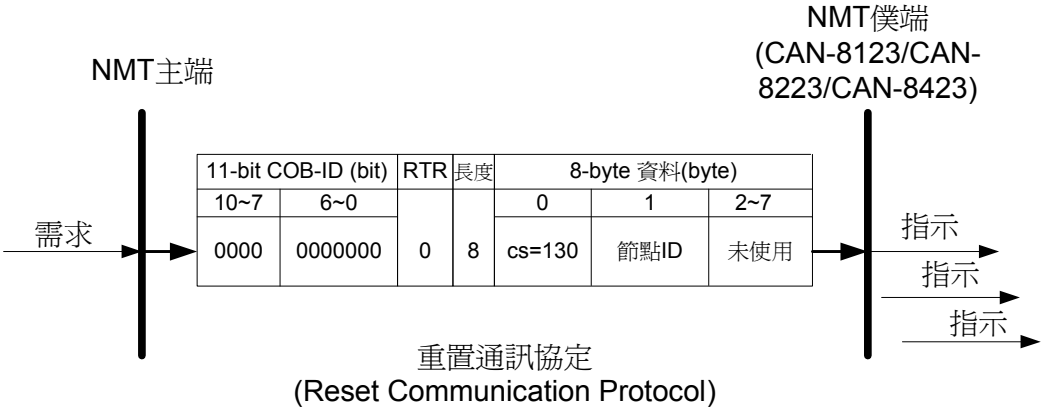


11-bit COB-ID (bit)		RTR	長度	8-byte 資料(byte)		
10~7	6~0			0	1	2~7
0000	0000000	0	8	cs=129	節點ID	未使用

cs ：NMT 命令識別符
129：重置(reset)節點

節點 **ID** ：NMT 僕端裝置的節點 ID。

重置通訊協定 (Reset Communication Protocol)



11-bit COB-ID (bit)		RTR	長度	8-byte 資料(byte)		
10~7	6~0			0	1	2~7
0000	0000000	0	8	cs=130	節點ID	未使用

cs ：NMT 命令識別符
130：重置通訊(Reset_Communication)

節點 **ID** ：NMT 僕端裝置的節點 ID。

模組控制協定範例 (Module Control Protocol Example)

步驟 1：關閉 CAN-8423。

步驟 2：啓動 CAN-8423。若無異常，CAN-8423 在結束初始化(initialization)後，會自動進入預操作(Pre_Operational)狀態。此時使用者可以看到 CAN-8423 面板上面的運行指示燈約每秒閃爍兩次。

步驟 3：傳送底下的訊息給 CAN-8423，透過 NMT 模組控制協定要求 CAN-8423 進入操作狀態(operational state)。

NMT 模組控制協定 第 159 頁

11-bit COB-ID (000)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0	0	0	0	8	01	01	00	00	00	00	00	00

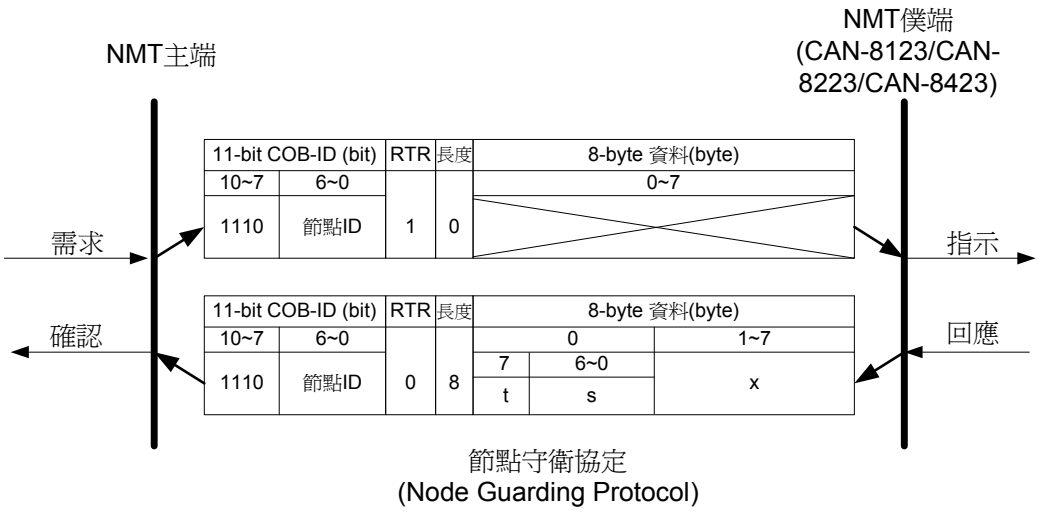
NMT 主端 ⟶ **NMT 僕端 (CAN-8123/CAN-8223/CAN-8423)**

cs ： 1
節點 **ID** ： 5

5.4.2 錯誤控制協定

透過錯誤控制協定，我們可以檢查網路當中的 CANopen 裝置是否存活(運作是否正常)。在物件字典之中，主索引爲 0x100C 的物件記錄了節點守衛時間(node guard time)，主索引爲 0x100D 的物件記錄了生存時間係數(life time factor)。而節點生存時間(node life time)爲節點守衛時間乘上生存時間係數。

CAN-8123/CAN-8223/CAN-8423 支援了 NMT 節點守衛協定的僕端功能，裝置在收到了具有特定 COB-ID 的遠端要求訊息(即守衛要求訊息，guarding request message)之後，便會根據節點守衛時間開始倒數，若裝置沒有在此時間內再次收到守衛要求訊息，裝置便會對外發出 EMCY 訊息。錯誤控制協定的細節如下所示：



t ：交替位元(toggle bit)
對 CANopen 的 NMT 僕端而言，在進行節點守衛協定的時候，每一次回覆訊息的交替位元均需與其上一個回覆訊息的交替位元不同。而在節點守衛協定開始進行時，NMT 僕端第 1 次回覆的訊息，其交替位元必須設定爲 0。

s ：NMT 僕端的狀態
4：停止(STOPPED)狀態
5：操作(OPERATIONAL)狀態
127：預操作(PRE-OPERATIONAL)狀態

reserved ：保留，留待進一步的使用，數值永爲 0。

<u>錯誤控制協定範例</u>

底下範例所使用的硬體架構與 PDO 通訊範例相同，使用者可參照 5.2.3 節。

步驟 1：移除 CAN-8423 的電源，再重新把 CAN-8423 接上電源。這時候 CAN-8423 會自動進入預操作狀態。

步驟 2：此處透過初始 SDO 下載協定，把節點守衛時間設定爲 250。節點守衛時間位於物件字典中主索引爲 0x100C 之處。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (601)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2B	0C	10	00	FA	00	00	00

SDO 用戶端 → **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 1
n ： 2
e ： 1
s ： 1
m ： 0C 10 00
d ： FA 00 00 00

步驟 3 : 如果沒有錯誤的話，CAN-8423 會回覆以下訊息以結束初始 SDO 下載。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (581)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	0C	10	00	00	00	00	00

SDO 用戶端 ← **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 3
m ： 0C 10 00

步驟 4：此處透過初始 SDO 下載協定，把生存時間係數設定爲 4。生存時間係數位於物件字典中主索引爲 0x100D 之處。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (601)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2F	0D	10	00	04	00	00	00

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 1
n ： 3
e ： 1
s ： 1
m ： 0D 10 00
d ： 04 00 00 00

步驟 5：如果沒有錯誤的話，CAN-8423 會回覆以下訊息以結束初始 SDO 下載。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (581)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	0D	10	00	00	00	00	00

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 3
m ： 0D 10 00

步驟 6：傳送以下的守衛要求訊息以開始節點守衛協定。這時候的生存時間爲 1 秒，也就是 1000 毫秒(節點守衛時間 × 生存時間係數 = 250 × 4 = 1000)。

NMT 節點守衛協定 第 163 頁

11-bit COB-ID (701)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	1	1	0	0	0	0	0	0	0	1	1	0	00	00	00	00	00	00	00	00

NMT 主端 ⟶ **NMT 僕端 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x701

步驟 7：接著，使用者可以接收到如下訊息，其內記錄了 CAN-8423 所回報的 NMT 狀態。

NMT 節點守衛協定 第 163 頁

11-bit COB-ID (701)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	1	1	0	0	0	0	0	0	0	1	0	8	7F	00	00	00	00	00	00	00

NMT 主端 ⟵ **NMT 僕端 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x701
t ： 1
s 7F
數值 7F 代表 CAN-8423 目前的 NMT 狀態爲預操作狀態。

步驟 8：因爲生存時間現在被設定爲 1 秒，所以傳送守衛要求訊息給 CAN-8423 的時間間隔必須在 1 秒之內。如果 CAN-8423 收到守衛要求訊息的時間間隔超過 1 秒，CAN-8423 就會發生生存守衛事件，對外傳送 EMCY 訊息，此時輸出通道的數值會根據物件字典中主索引爲 0x6206、0x6207、0x6443 和 0x6444 的內容作改變。

如果此處停止傳送守衛要求訊息給 CAN-8423，它就會發出以下訊息。

11-bit COB-ID (81)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
0	0	0	1	0	0	0	0	1	0	1	0	8	30	81	11	07	00	00	00	00

EMCY 消費者 ◄ **EMCY 生產者 (CAN-8123/CAN-8223/CAN-8423)**

EMCY-msg ： 30 81 11 07 00 00 00 00

前 2 bytes“30 81”，代表緊急錯誤碼。第 3 個 byte“11”代表 CAN-8423 錯誤暫存器的數值。後 5 bytes “07 00 00 00 00” 爲製造商特定錯誤區域的數值。根據緊急錯誤碼，我們可以知道這個 EMCY 訊息被傳送是因爲發生了生存守衛事件。

5.5 CAN-8123/CAN-8223/CAN-8423 的特殊功能

類比模組的輸入輸出範圍項目

泓格科技在 CAN-8123/CAN-8223/CAN-8423 物件字典內的製造商特定描述文件區域，定義了數個項目，用來記錄裝置上類比模組的輸入/輸出範圍碼。使用的物件之主索引範圍爲 0x2004 到 0x200B。這 8 個物件分別依序對應到裝置上的擴充插槽 0 到擴充插槽 7。若裝置上面任何一個插槽有安裝模組，則裝置的物件字典中就會有與擴充插槽相對應的物件。若裝置上面一個模組都沒有安裝，那物件字典中就不會有這些物件。

以 CAN-8123 爲例，如果使用者在編號 0 的擴充插槽上安裝了模組，那它的物件字典內就會有 0x2004 的物件。以 CAN-8423 爲例，如果使用者在裝置上任何一個插槽安裝了模組，那它的物件字典內就會有 0x2004、0x2005、0x2006 和 0x2007 的物件。

如果主索引 0x2004 到 0x200B 的物件存在，但相對應編號擴充插槽上面裝的不是類比模組，則物件子索引 0 數值就會是 0。如果主索引 0x2004 到 0x200B 物件存在，但在相對應編號擴充插槽上面裝的是類比模組，則相對應物件的子索引 0 的數值就會是該類比模組的通道數目，同時也代表這個物件的子索引的最大定址範圍。如果子索引 0x01 存在的話，那其內會記錄相對應模組上的類比通道 1 的輸入/輸出範圍碼，如果子索引 0x02 存在的話，那其內會記錄相對應模組上的類比通道 2 的輸入/輸出範圍碼，以此類推到物件字典中的最後一個子索引(模組上的最後一個類比通道)。

舉例來說，假設 CAN-8423 上的四個擴充插槽(插槽 0、1、2、3)分別依序裝上 I-8057、I-8053、I-8024、I-87017。在 CAN-8423 開機之後，它的物件字典內就會有主索引 0x2004、0x2005、0x2006、0x2007 的物件。因爲前兩個插槽裝的是數位的模組，所以前兩個物件子索引 0x00 的值會是 0，後兩個物件子索引 0x00 的數值就不是 0，因爲 I-8024 的有四組 AO 通道，所以第三個物件子索引 0x00 的數值會是 4，因爲因爲 I-87017 有八組 AI 通道，所以第四個物件子索引 0x00 的數值會是 8。在預設的情況下，第三個物件(主索引 0x2006)子索引 0x01~0x04 的數值會是 0，也就是 I-8024 的範圍碼會是 0，根據附錄 B，我們可以知道 I-8024 的範圍碼 0 代表 AO 通道的範圍被設定爲-10V~+10V。在預設的情況下，第四個物件(主索引 0x2007)子索引 0x01~0x08 的數值會是 8，也就是 I-87017 的範圍碼會是 8，根據附錄 B，我們可以知道 I-87017 的範圍碼 8 代表 AI 通道的範圍被設定爲-10V~+10V。

有關各類比模組的範圍碼與實際輸出輸入範圍的對應關係，使用者可以參考附錄 B 以取得更詳細的資料。

於下我們將透過一個簡單的範例來展示如何讀取、修改範圍碼，以及範圍碼修改後，類比模組所受到的影響。這裡所使用的硬體配置請參考第五章一開始的部份。

步驟 1：首先傳送如下 NMT 訊息，使 CAN-8423 的 NMT 狀態變更爲操作狀態。

NMT 模組控制協定 第 159 頁

11-bit COB-ID (000)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0	0	0	0	8	01	01	00	00	00	00	00	00

NMT 主端 ⟶ **NMT 僕端 (CAN-8123/CAN-8223/CAN-8423)**

cs ： 1
節點 ID ： 1

步驟 2：傳送以下的 SDO 訊息以讀取 I-8024 的 AO 通道 0 之範圍碼。

初始 SDO 上傳協定 第 67 頁

11-bit COB-ID (601)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	40	06	20	01	00	00	00	00

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 2
m ： 06 20 01

步驟 3：CAN-8423 會回應如下的 SDO 訊息。若 I-8024 處於預設狀態的話，SDO 訊息裡面記載的數值就會是 0，也就是 I-8024 的 AO 通道 0 之範圍碼爲 0，輸出範圍爲-10V~+10V。

初始 SDO 上傳協定 第 67 頁

11-bit COB-ID (581)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	4F	06	20	01	00	00	00	00

SDO 用戶端 ← **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 2
n ： 3
e ： 1
s ： 1
m ： 06 20 01
d ： 00 00 00 00
因爲 **n=3**，只有第 4 個 byte 是有效的。因此 CAN-8423 透過這個 SDO 訊息回傳的數值就是 00。

步驟 4：傳送以下的 SDO 訊息以讀取 I-87017 的 AI 通道 0 之範圍碼。

初始 SDO 上傳協定 第 67 頁

11-bit COB-ID (601)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	40	07	20	01	00	00	00	00

SDO 用戶端 → **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 2
m ： 07 20 01

步驟 5：CAN-8423 會回應如下的 SDO 訊息。若 I-87017 處於預設狀態的話，SDO 訊息裡面記載的數值就會是 8，也就是 I-87017 的 AI 通道 0 之範圍碼爲 8，輸入範圍爲-10V~+10V。

初始 SDO 上傳協定 第 67 頁

11-bit COB-ID (581)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	4F	07	20	01	08	00	00	00

SDO 用戶端 ← **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 2
n ： 3
e ： 1
s ： 1
m ： 07 20 01
d ： 08 00 00 00
因爲 **n=3**，只有第 4 個 byte 是有效的。因此 CAN-8423 透過這個 SDO 訊息回傳的數值就是 00。

步驟 6：於此處，傳送如下的第 2 組 RxPDO 訊息給 CAN-8423，改變 I-8024 的 AO0 輸出值爲 7V。

PDO 通訊協定 第 107 頁

11-bit COB-ID (301)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	1	1	0	0	0	0	0	0	0	1	0	8	98	59	00	00	00	00	00	00

PDO 生產者 ⟶ **PDO 消費者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x301

L ： 8

PDO-msg ： 98 59 00 00 00 00 00 00

此處 8 bytes 的資料分別映射到 I-8024 的 4 個 AO 通道，每個 AO 通道佔用 2 bytes，依序分別爲通道 AO0、AO1、AO2、AO3。

若我們欲透過 AO0 輸出 7V，透過下面的轉換公式：

$$\text{通道值}_{\text{hex}} = \left[\frac{7\text{V} - (-10\text{V})}{10\text{V} - (-10\text{V})}\right] * [32767 - (-32768)] + (-32768)$$
$$= 22936.75 \approx 22937 = 0\text{x}5998$$

可知 PDO 訊息的前 2 bytes 需填入“98”和“59”。此處其餘 AO 通道的輸出欲設定爲 0V，所以剩下 6 bytes 便填入“00 00 00”。

有關轉換公式的細節，可參照 5.2.3 節和 6.3 節。

步驟 7：此時 AI0 所量測到的數值會因 AO0 改變而改變，但 CAN-8423 並不會自動傳送 AI0 對應的 TxPDO(第 2 組 TxPDO)給 PDO 消費者。因此，使用者需要利用 PDO 要求訊息要求 CAN-8423 傳送第 2 組 TxPDO，以讀取 AI0 的數值。

PDO 通訊協定 第 107 頁

11-bit COB-ID (281)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
0	1	0	1	0	0	0	0	0	0	1	1	0								

PDO 消費者 → **PDO 生產者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x281

步驟 8：CAN-8423 在收到 RTR 訊息之後，就會回傳第 2 組 TxPDO，如下。

PDO 通訊協定 第 107 頁

11-bit COB-ID (281)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
0	1	0	1	0	0	0	0	0	0	1	0	8	80	59	FD	FF	FD	FF	FD	FF

PDO 消費者 ← **PDO 生產者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x281
L ： 8
PDO-msg ： 80 59 FD FF FD FF FD FF
這 8 bytes 分別代表爲 I-87017 的 AI0、AI1、AI2、AI3 所量到的數值，此處 AI0 所測量到的數值爲 0x5980 (22912)。

根據附錄 B 的轉換表，可知 I-87017 預設的通道值 $_{float}$ 和通道值 $_{hex}$ 範圍分別爲-10V~10V 和 0x8000(-32768)~0x7FFF(32767)。透過下面的轉換公式，可以將 0x5980 轉換爲通道值 $_{float}$，如下：

$$通道值_{float} = \left[\frac{22912-(-32768)}{32767-(-32768)}\right] * (10V-(-10V)) + (-10V)$$
$$\approx 6.992V$$

有關轉換公式的細節，可參照 5.2.3 節和 6.3 節。

步驟 9：傳送如下的 SDO 訊息給 CAN-8423，要求更改其物件字典內主索引 0x2007 子索引 0x01 之項目，於此處要求此項目的數值更改爲 9。也就是把 I-87017 的 AI0 範圍碼改爲 9，輸入範圍改爲-5V 到+5V。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (601)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2F	07	20	01	01	00	00	00

SDO 用戶端 ⟶ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

ccs ： 1
n ： 3
e ： 1
s ： 1
m ： 07 20 01
d ： 09 00 00 00

步驟 10：CAN-8423 在數值更改完畢之後，會回復如下的 SDO 訊息。於此處，使用者可以再利用初始 SDO 上傳協定把數值讀出來比對。

初始 SDO 下載協定 第 87 頁

11-bit COB-ID (581)											RTR	資料長度	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	07	20	01	00	00	00	00

SDO 用戶端 ⟵ **SDO 伺服端 (CAN-8123/CAN-8223/CAN-8423)**

scs ： 3
m ： 07 20 01

步驟 11：此處利用如下的 PDO 要求訊息，通知 CAN-8423 回傳第 2 組 TxPDO，以讀取 AI0 的數值。

PDO 通訊協定 第 107 頁

11-bit COB-ID (281)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
0	1	0	1	0	0	0	0	0	0	1	1	0								

PDO 消費者 ⟵ **PDO 生產者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x281

步驟 12：CAN-8423 會回傳如下的第 2 組 TxPDO。

PDO 通訊協定 第 107 COB-ID (281)											RTR	資料	8-byte 資料							
10	9	8	7	6	5	4	3	2	1	0		長度	0	1	2	3	4	5	6	7
0	1	0	1	0	0	0	0	0	0	1	0	8	FF	7F	FF	FF	FD	FF	FF	FF

PDO 消費者 ⟵ **PDO 生產者 (CAN-8123/CAN-8223/CAN-8423)**

COB-ID ： 0x281

L ： 8

PDO-msg ： FF 7F FF FF FD FF FF FF

這 8 bytes 分別代表爲 I-87017 的 AI0、AI1、AI2、AI3 所量到的數值，此處 AI0 所測量到的數值爲 0x7FFF，0x7FFF 是輸入的上限值，也就是 AI0 所測量到的數值爲目前輸入範圍的最大值，即爲 5V。

雖然 I-8024 的 AO0 輸出是 7V，但是 7V 已經超出了 AI0 的輸入範圍 (-5V~5V)，所以 AI0 於此處就只能量到 5V。

6 CAN-8123/CAN-8223/CAN-8423 的物件字典

6.1 通訊描述文件區域

底下會列出在 CAN-8123/CAN-8223/CAN-8423 的物件字典中，於通訊描述文件區域內的各個項目。爲方便閱讀，此處將所有通訊描述文件區域內的項目區分爲數個表格，分別爲：“一般通訊項目”、“SDO 通訊項目”、“RxPDO 通訊項目”、“RxPDO 映射項目”、“TxPDO 通訊項目”以及“TxPDO 映射項目”。在預設區裡面的“---”代表這個項目沒有定義預設值，或著在某些條件下，才會由 CAN-8123/CAN-8223/CAN-8423 內建的韌體來指定預設值。在表格內，如果數字的後方帶有“h”字樣，即代表這個數字是以 16 進位來表示的。

一般通訊項目(General Communication Entries)

主索引	子索引	描述	型態	屬性	預設值
1000h	0h	裝置型態	UNSIGNED 32	唯讀	---
1001h	0h	錯誤暫存器	UNSIGNED 8	唯讀	---
1003h	0h	“預設錯誤區 ”子索引最大定址範圍	UNSIGNED 8	唯讀	FEh
	1h	實際的錯誤 (最新的)	UNSIGNED 32	唯讀	---
	...	...	...	...	---
	5h	實際的錯誤 (最舊的)	UNSIGNED 32	唯讀	---
1005h	0h	SYNC 訊息的 COB-ID	UNSIGNED 32	可讀寫	80h
1008h	0h	製造商所定義的裝置名稱	VISIBLE_STRING	唯讀	CAN-8123/ CAN-8223/ CAN-8423/
1009h	0h	製造商所定義的硬體版本	VISIBLE_STRING	唯讀	---
100Ah	0h	製造商所定義的軟體版本	VISIBLE_STRING	唯讀	---
100Ch	0h	守衛時間	UNSIGNED 16	可讀寫	0
100Dh	0h	生存時間係數	UNSIGNED 8	可讀寫	0
1014h	0h	EMCY 訊息的 COB-ID	UNSIGNED 32	可讀寫	80h+節點 ID
1015h	0h	EMCY 訊息的抑制時間	UNSIGNED 16	可讀寫	0
1018h	0h	“識別物件”子索引最大定址範圍	UNSIGNED 8	唯讀	1
	1h	供應商的 ID	UNSIGNED 32	唯讀	---

註： 1. 主索引爲 0x1000 物件內的項目，其資料格式如下：

附加資訊		一般資訊
bit 31~ bit 24	bit 23 ~ bit16	bit 15 ~ bit 0
特定功能碼	I/O 功能碼	裝置描述文件碼

在 CAN-8123/CAN-8223/CAN-8423 上，特定功能碼的數值會保持 0。I/O 功能碼則定義了 CAN-8123/CAN-8223/CAN-8423 是哪一種性質的裝置，功能碼內的 bit 16、17、18、19 分別代表著裝置是否有 DI、DO、AI、AO 的通道。舉例來說，如果 bit 16 爲 1，這就代表著 CAN-8123/CAN-8223/CAN-8423 有 DI 的通道，如果 bit 16 和 bit 17 爲 1，這就代表著 CAN-8123/CAN-8223/CAN-8423 有 DI 和 DO 的通道。功能碼內的 bit 20 到 bit 19 的數值會保持爲 0。一般資訊內記錄的爲裝置描述文件碼，CAN-8123/CAN-8223/CAN-8423 的裝置描述文件碼爲 0x191(0x191=401)，代表裝置支援 CANopen 的裝飾描述文件規範 DS-401。

2. 有關主索引 0x1001 和主索引 0x1003 的物件，請參照 5.3.2 節。

3. 主索引 0x1005 的物件裡面存放的是 SYNC 的 COB-ID。如果要傳 SYNC 訊息給 CAN-8123/CAN-8223/CAN-8423，就必須使用裝置上所記錄的 SYNC COB-ID，其格式如下表所示：

位元編號	數值	代表的意義
31 (MSB)	x	這個 bit 沒有意義，無須理會
30	0	裝置不會產生 SYNC 訊息
	1	裝置會產生 SYNC 訊息
29	0	11-bit ID (CAN 2.0A)
	1	29-bit ID (CAN 2.0B)
28-11	0	如果 bit 29=0，則這 18 個 bit 就爲 0
	x	如果 bit 29=1，則這 18 個 bit 就是 COB-ID 的 bit 28 到 bit 11
10-0 (LSB)	x	是 COB-ID 的 bit 10 到 bit 0

CAN-8123/CAN-8223/CAN-8423 不支援產生 SYNC 訊息，也不支援 29 bits 的 COB-ID，所以 bit 29、30、31 的數值就會保持爲 0。

4. 物件字典內，主索引 0x1008、0x1009、0x100A 的物件，記錄的是製造商於物件字典內記錄的產品資訊。使用者可以對照 ASCII 碼，利用查表的方式來瞭解裝置的產品資訊。

5. 有關主索引 0x100C、0x100D 的物件，請參照 5.4.2 節。主所引 0x100C 的物件，其數值範圍爲 0~32767。主所引 0x100D 的物件，其數值範圍爲 0~255。

6. 有關主索引 0x1014 的物件，請參照 5.3.1 節。

7. 物件字典內，主索引 0x1015 的物件，記錄的是 EMCY 的抑制時間。兩個 EMCY 訊息傳輸的時間間隔必須大於抑制時間。這個物件的功能很類似 PDO 的抑制時間 (PDO 通訊參數物件中的子索引 0x04)。透過抑制時間的設定，可以有效的避免因頻繁傳送 EMCY 訊息，導致 CAN bus 負載過重的情形。在 CAN-8123/CAN-8223/CAN-8423 上，這個物件的數值範圍爲 0 到 32767，單位爲 ms。

SDO 通訊項目(SDO Communication Entries)

主索引	子索引	描述	型態	屬性	預設值
1200h	0h	“伺服 SDO 參數”子索引最大定址範圍	UNSIGNED 8	唯讀	2
	1h	RxSDO 的 COB-ID。(用戶端到伺服端)	UNSIGNED 32	唯讀	600h+節點 ID
	2h	TxSDO 的 COB-ID。(伺服端到用戶端)	UNSIGNED 32	唯讀	580h+節點 ID

RxPDO 通訊項目(RxPDO Communication Entries)

主索引	子索引	描述	型態	屬性	預設值
1400h	0h	第 1 組 “RxPDO 通訊參數” 子索引的最大定址範圍	UNSIGNED 8	唯讀	2
	1h	第 1 組 RxPDO 的 COB-ID	UNSIGNED 32	可讀寫	200h+節點 ID
	2h	第 1 組 RxPDO 的傳輸型態	UNSIGNED 8	可讀寫	FFh
1401h	0h	第 2 組 “RxPDO 通訊參數” 子索引的最大定址範圍	UNSIGNED 8	唯讀	5
	1h	第 2 組 RxPDO 的 COB-ID	UNSIGNED 32	可讀寫	300h+節點 ID
	2h	第 2 組 RxPDO 的傳輸型態	UNSIGNED 8	可讀寫	FFh
1402h	0h	第 3 組 “RxPDO 通訊參數” 子索引的最大定址範圍	UNSIGNED 8	唯讀	5
	1h	第 3 組 RxPDO 的 COB-ID	UNSIGNED 32	可讀寫	400h+節點 ID
	2h	第 3 組 RxPDO 的傳輸型態	UNSIGNED 8	可讀寫	FFh
1403h	0h	第 4 組“RxPDO 通訊參數” 子索引的最大定址範圍	UNSIGNED 8	唯讀	5
	1h	第 4 組 RxPDO 的 COB-ID	UNSIGNED 32	可讀寫	500h+節點 ID
	2h	第 4 組 RxPDO 的傳輸型態	UNSIGNED 8	可讀寫	FFh
1404h	0h	第 5 組 “RxPDO 通訊參數” 子索引的最大定址範圍	UNSIGNED 8	唯讀	5
	1h	第 5 組 RxPDO 的 COB-ID	UNSIGNED 32	可讀寫	80000000h
	2h	第 5 組 RxPDO 的傳輸型態	UNSIGNED 8	可讀寫	FFh
…	…	…	…	…	…
140Fh	0h	第 16 組 “RxPDO 通訊參數” 子索引的最大定址範圍	UNSIGNED 8	唯讀	5
	1h	第 16 組 RxPDO 的 COB-ID	UNSIGNED 32	可讀寫	8000 0000h
	2h	第 16 組 RxPDO 的傳輸型態	UNSIGNED 8	可讀寫	FFh

RxPDO 映射項目(RxPDO Mapping Entries)

主索引	子索引	描述	型態	屬性	預設值
1600h	0h	第1組 "RxPDO映射參數" 子索引的最大定址範圍	UNSIGNED 8	唯讀	8
	1h	對 1h 到 8h 的數位輸出通道作寫入	UNSIGNED 8	可讀寫	6200 0108h
	2h	對 9h 到 10h 的數位輸出通道作寫入	UNSIGNED 8	可讀寫	6200 0208h
	3h	對 11h 到 18h 的數位輸出通道作寫入	UNSIGNED 8	可讀寫	6200 0308h
	4h	對 19h 到 20h 的數位輸出通道作寫入	UNSIGNED 8	可讀寫	6200 0408h
	5h	對 21h 到 28h 的數位輸出通道作寫入	UNSIGNED 8	可讀寫	6200 0508h
	6h	對 29h 到 30h 的數位輸出通道作寫入	UNSIGNED 8	可讀寫	6200 0608h
	7h	對 31h 到 38h 的數位輸出通道作寫入	UNSIGNED 8	可讀寫	6200 0708h
	8h	對 39h 到 40h 的數位輸出通道作寫入	UNSIGNED 8	可讀寫	6200 0808h
1601h	0h	第2組 "RxPDO映射參數" 子索引的最大定址範圍	UNSIGNED 8	唯讀	4
	1h	對 1h 的類比輸出通道作寫入	UNSIGNED 16	可讀寫	6411 0110h
	2h	對 2h 的類比輸出通道作寫入	UNSIGNED 16	可讀寫	6411 0210h
	3h	對 3h 的類比輸出通道作寫入	UNSIGNED 16	可讀寫	6411 0310h
	4h	對 4h 的類比輸出通道作寫入	UNSIGNED 16	可讀寫	6411 0410h
1602h	0h	第3組 "RxPDO映射參數" 子索引的最大定址範圍	UNSIGNED 8	唯讀	4
	1h	對 5h 的類比輸出通道作寫入	UNSIGNED 16	可讀寫	6411 0510h
	2h	對 6h 的類比輸出通道作寫入	UNSIGNED 16	可讀寫	6411 0610h
	3h	對 7h 的類比輸出通道作寫入	UNSIGNED 16	可讀寫	6411 0710h
	4h	對 8h 的類比輸出通道作寫入	UNSIGNED 16	可讀寫	6411 0810h
1603h	0h	第4組 "RxPDO映射參數" 子索引的最大定址範圍	UNSIGNED 8	唯讀	4
	1h	對 9h 的類比輸出通道作寫入	UNSIGNED 16	可讀寫	6411 0910h
	2h	對 Ah 的類比輸出通道作寫入	UNSIGNED 16	可讀寫	6411 0A10h
	3h	對 Bh 的類比輸出通道作寫入	UNSIGNED 16	可讀寫	6411 0B10h
	4h	對 Ch 的類比輸出通道作寫入	UNSIGNED 16	可讀寫	6411 0C10h

1604h	0h	第5組 ”RxPDO映射參數” 子索引的最大定址範圍	UNSIGNED 8	唯讀	---
	1h	---	---	可讀寫	---
	…	…	…	…	…
	---	---	---	可讀寫	---
…	…	…	…	…	…
160Fh	0h	第 16 組 ”RxPDO 映射參數” 子索引的最大定址範圍	UNSIGNED 8	唯讀	---
	1h	---	---	可讀寫	---
	…	…	…	…	…
	---	---	---	可讀寫	---

註： 1h 到 8h 的數位輸出通道，指的是裝置上第 1 組 DO 通道到第 8 組 DO 通道，如果 CAN-8123/CAN-8223/CAN-8423 上的 DI 模組是 I-8057 的話，那第 1 組 DO 通道到第 8 組 DO 通道，指的就是 I-8057 上的 DO0 到 DO7。依此類推，類比輸出通道亦同。

TxPDO 通訊項目(TxPDO Communication Entries)

主索引	子索引	描述	型態	屬性	預設值
1800h	0	第1組 "TxPDO通訊參數" 子索引的最大定址範圍	UNSIGNED 8	唯讀	5
	1	第 1 組 TxPDO 的 COB-ID	UNSIGNED 32	可讀寫	180h+Node-ID
	2	第 1 組 TxPDO 的傳輸型態	UNSIGNED 8	可讀寫	FFh
	3	第 1 組 TxPDO 的抑制時間	UNSIGNED 16	可讀寫	0
	4	此項目被保留	---	---	---
	5	第 1 組 TxPDO 的事件計時器	UNSIGNED 16	可讀寫	0
1801h	0	第2組 "TxPDO通訊參數" 子索引的最大定址範圍	UNSIGNED 8	唯讀	5
	1	第 2 組 TxPDO 的 COB-ID	UNSIGNED 32	可讀寫	280h+Node-ID
	2	第 2 組 TxPDO 的傳輸型態	UNSIGNED 8	可讀寫	FFh
	3	第 2 組 TxPDO 的抑制時間	UNSIGNED 16	可讀寫	0
	4	此項目被保留	---	---	---
	5	第 2 組 TxPDO 的事件計時器	UNSIGNED 16	可讀寫	0
1802h	0	第3組 "TxPDO通訊參數" 子索引的最大定址範圍	UNSIGNED 8	唯讀	5
	1	第 3 組 TxPDO 的 COB-ID	UNSIGNED 32	可讀寫	380h+Node-ID
	2	第 3 組 TxPDO 的傳輸型態	UNSIGNED 8	可讀寫	FFh
	3	第 3 組 TxPDO 的抑制時間	UNSIGNED 16	可讀寫	0
	4	此項目被保留	---	---	---
	5	第 3 組 TxPDO 的事件計時器	UNSIGNED 16	可讀寫	0
1803h	0	第4組 "TxPDO通訊參數" 子索引的最大定址範圍	UNSIGNED 8	唯讀	5
	1	第 4 組 TxPDO 的 COB-ID	UNSIGNED 32	可讀寫	480h+Node-ID
	2	第 4 組 TxPDO 的傳輸型態	UNSIGNED 8	可讀寫	FFh
	3	第 4 組 TxPDO 的抑制時間	UNSIGNED 16	可讀寫	0
	4	此項目被保留	---	---	---
	5	第 4 組 TxPDO 的事件計時器	UNSIGNED 16	可讀寫	0

1804h	0	第5組 "TxPDO 通訊參數" 子索引的最大定址範圍	UNSIGNED 8	唯讀	5
	1	第 5 組 TxPDO 的 COB-ID	UNSIGNED 32	可讀寫	80000000h
	2	第 5 組 TxPDO 的傳輸型態	UNSIGNED 8	可讀寫	FFh
	3	第 5 組 TxPDO 的抑制時間	UNSIGNED 16	可讀寫	0
	4	此項目被保留	---	---	---
	5	第 5 組 TxPDO 的事件計時器	UNSIGNED 16	可讀寫	0
…	…	…	…	…	…
180Fh	0	第 16 組 "TxPDO 通訊參數" 子索引的最大定址範圍	UNSIGNED 8	唯讀	5
	1	第 16 組 TxPDO 的 COB-ID	UNSIGNED 32	可讀寫	80000000h
	2	第 16 組 TxPDO 的傳輸型態	UNSIGNED 8	可讀寫	FFh
	3	第 16 組 TxPDO 的抑制時間	UNSIGNED 16	可讀寫	0
	4	此項目被保留	---	---	---
	5	第 16 組 TxPDO 的事件計時器	UNSIGNED 16	可讀寫	0

TxPDO 映射項目(TxPDO Mapping Entries)

主索引	子索引	描述	型態	屬性	預設值
1A00h	0h	第 1 組 "TxPDO 映射參數" 子索引的最大定址範圍	UNSIGNED 8	唯讀	8
	1h	讀取 1h 到 8h 的數位輸入通道數值	UNSIGNED 8	可讀寫	6000 0108h
	2h	讀取 9h 到 10h 的數位輸入通道數值	UNSIGNED 8	可讀寫	6000 0208h
	3h	讀取 11h 到 18h 的數位輸入通道數值	UNSIGNED 8	可讀寫	6000 0308h
	4h	讀取 19h 到 20h 的數位輸入通道數值	UNSIGNED 8	可讀寫	6000 0408h
	5h	讀取 21h 到 28h 的數位輸入通道數值	UNSIGNED 8	可讀寫	6000 0508h
	6h	讀取 29h 到 30h 的數位輸入通道數值	UNSIGNED 8	可讀寫	6000 0608h
	7h	讀取 31h 到 38h 的數位輸入通道數值	UNSIGNED 8	可讀寫	6000 0708h
	8h	讀取 39h 到 40h 的數位輸入通道數值	UNSIGNED 8	可讀寫	6000 0808h
1A01h	0h	第 2 組 "TxPDO 映射參數" 子索引的最大定址範圍	UNSIGNED 8	唯讀	4
	1h	讀取 1h 的類比輸入通道數值	UNSIGNED 16	可讀寫	6401 0110h
	2h	讀取 2h 的類比輸入通道數值	UNSIGNED 16	可讀寫	6401 0210h
	3h	讀取 3h 的類比輸入通道數值	UNSIGNED 16	可讀寫	6401 0310h
	4h	讀取 4h 的類比輸入通道數值	UNSIGNED 16	可讀寫	6401 0410h
1A02h	0h	第 3 組 "TxPDO 映射參數" 子索引的最大定址範圍	UNSIGNED 8	唯讀	4
	1h	讀取 5h 的類比輸入通道數值	UNSIGNED 16	可讀寫	6401 0510h
	2h	讀取 6h 的類比輸入通道數值	UNSIGNED 16	可讀寫	6401 0610h
	3h	讀取 7h 的類比輸入通道數值	UNSIGNED 16	可讀寫	6401 0710h
	4h	讀取 8h 的類比輸入通道數值	UNSIGNED 16	可讀寫	6401 0810h
1A03h	0h	第 4 組 "TxPDO 映射參數" 子索引的最大定址範圍	UNSIGNED 8	唯讀	4
	1h	讀取 9h 的類比輸入通道數值	UNSIGNED 16	可讀寫	6401 0910h
	2h	讀取 Ah 的類比輸入通道數值	UNSIGNED 16	可讀寫	6401 0A10h
	3h	讀取 Bh 的類比輸入通道數值	UNSIGNED 16	可讀寫	6401 0B10h
	4h	讀取 Ch 的類比輸入通道數值	UNSIGNED 16	可讀寫	6401 0C10h

1A04h	0h	第 5 組 ”TxPDO 映射參數” 子索引的最大定址範圍	UNSIGNED 8	唯讀	---
	1h	---	---	可讀寫	---
	…	…	…	…	…
	---	---	---	可讀寫	---
…	…	…	…	…	…
1A0Fh	0h	第 16 組 ”TxPDO 映射參數” 子索引的最大定址範圍	UNSIGNED 8	唯讀	---
	1h	---	---	可讀寫	---
	…	…	…	…	…
	---	---	---	可讀寫	---

註： 1h 到 8h 的數位輸入通道，指的是裝置上第 1 組 DI 通道到第 8 組 DI 通道，如果 CAN-8123/CAN-8223/CAN-8423 上的 DI 模組是 I-8053 的話，那第 1 組 DI 通道到第 8 組 DI 通道，指的就是 I-8053 上的 DI0 到 DI7。依此類推，類比輸入通道亦同。

6.2 製造商特定描述文件區域

CAN-8123/CAN-8223/CAN-8423 除了實作 CANopen 規範內的功能之外，另外還額外實做了一些特殊的功能。如主索引 0x2004 到 0x200B 的物件，其內記錄了裝置上，擴充模組的類比通道輸入/輸出範圍碼。舉例來說，如果 CAN-8423 的任一擴充插槽上有安裝擴充模組，則 CAN-8423 的物件字典內就會有主索引 0x2004 到 0x2007 的物件。有關這些物件的相關細節，請參閱 5.5 節。

類比模組的輸入/輸出範圍碼項目

主索引	子索引	描述	型態	屬性	預設值
2004h	0h	”類比模組的輸入/輸出範圍碼” 子索引的最大定址範圍	UNSIGNED 8	唯讀	依編號 0 的擴充插槽上的 AI/AO 模組之通道數而決定
	1h	AI/AO 通道 0 的輸入/輸出範圍碼	UNSIGNED 16	可讀寫	---
	…	…	…	…	…
2005h	0h	”類比模組的輸入/輸出範圍碼” 子索引的最大定址範圍	UNSIGNED 8	唯讀	依編號 1 的擴充插槽上的 AI/AO 模組之通道數而決定
	1h	AI/AO 通道 0 的輸入/輸出範圍碼	UNSIGNED 8	可讀寫	---
	…	…	…	…	…
…	…	…	…	…	…
200Bh	0h	”類比模組的輸入/輸出範圍碼” 子索引的最大定址範圍	UNSIGNED 8	唯讀	依編號 16 的擴充插槽上的 AI/AO 模組之通道數而決定
	1h	AI/AO 通道 0 的輸入/輸出範圍碼	UNSIGNED 8	可讀寫	---
	…	…	…	…	…

6.3 標準化裝置描述文件區域

當 CAN-8123/CAN-8223/CAN-8423 接上電源，CAN-8123/CAN-8223/CAN-8423 內建的韌體便會根據其上擴充模組的通道種類、通道數目等，自動產生相對應的的裝置描述文件項目。

爲了方便觀察，於底下將這些項目劃分成四類，分別爲“數位輸入裝置項目”、“數位輸出裝置項目”、“類比輸入裝置項目”、“類比輸出裝置項目”，如下面的四個表格所示：

數位輸入裝置項目

主索引	子索引	描述	型態	屬性	預設值
6000h	0h	”數位輸入(8 bit)讀取”子索引的最大定址範圍	UNSIGNED 8	唯讀	8
	1h	讀取 1h 到 8h 的數位輸入通道數值	UNSIGNED 8	唯讀	---
	…	…	…	…	…

數位輸出裝置項目

主索引	子索引	描述	型態	屬性	預設值
6200h	0h	”數位輸出(8 bit)寫入”子索引的最大定址範圍	UNSIGNED 8	唯讀	---
	1h	對 1h 到 8h 的數位輸出通道作寫入	UNSIGNED 8	可讀寫	---
	…	…	…	…	…
6206	0h	”數位輸出(8 bit)錯誤模式”子索引的最大定址範圍	UNSIGNED 8	可讀寫	---
	1h	設定 1h 到 8h 的數位輸出錯誤模式	UNSIGNED 8	可讀寫	0
	…	…	…	…	…
6207	0h	”數位輸出(8 bit)錯誤值”子索引的最大定址範圍	UNSIGNED 8	可讀寫	---
	1h	設定 1h 到 8h 的數位輸出錯誤值	UNSIGNED 8	可讀寫	0
	…	…	…	…	…

註： 當裝置偵測到 bus-off 或著節點守衛錯誤，又或著裝置的 NMT 狀態變成停止，此時 CAN-8123/CAN-8223/CAN-8423 就會去檢查主索引 0x6206 的物件，若其內的某個項目的某個 bit 被設定爲 1，則對應的 DO 通道在此時就會輸出記錄在主索引 0x6207 物件內相對應項目中的"數位輸出錯誤值"，如果 0x6206 內的某個項目的某個 bit 被設定爲 0，則對應的 DO 通道在此時就會保持原本的輸出值。

舉例來說，若裝置上有 1h 到 8h 的 DO，且裝置上物件字典內主索引 0x6206 和 0x6207 的物件，其內子索引 0x01 的項目分別被設定爲 0x31 和 0xF8，也就是二進制的 0011 0001 和 1111 1000。當裝置偵測到 bus-off 或節點守衛錯誤，又或著裝置的 NMT 狀態變成停止時，根據主索引 0x6206 物件內子索引 0x01 項目之數值 0x31，可知裝置上的 DO5、DO4、DO0 會被設定爲"數位輸出錯誤值"，又根據主索引 0x6207 物件內子索引 0x01 項目之數值 0xF8，可知 DO5、DO4、DO 的”數位輸出錯誤值”分別爲"1"、"1"、"0"，至於其他的數位輸出通道則會保持原有的數值。

類比輸入裝置項目

主索引	子索引	描述	型態	屬性	預設值
6401h	0h	”類比輸入(16 bit)讀取”子索引的最大定址範圍	UNSIGNED 8	唯讀	8
	1h	讀取 1h 的類比輸入通道數值	UNSIGNED 16	唯讀	---
	...	...	...	...	...

註： 因爲 CAN-8123/CAN-8223/CAN-8423 是使用 16 進制(hex))的數值表示法來儲存類比通道的數值，所以在把 AI 的數值從裝置上的物件字典讀取出來之後，使用者還必須要透過下面的轉換公式，才能夠把 AI 的輸入值轉爲實際的通道數值，也就是把通道值 $_{hex}$ 轉爲通道值 $_{float}$，如下：

$$通道值_{float}=\left[\frac{通道值_{hex}-最小通道值_{hex}}{最大通道值_{hex}-最小通道值_{hex}}\right]*(最大通道值_{float}-最小通道值_{float})+(最小通道值_{float})$$

通道值 $_{hex}$ 就是存放在裝置物件字典裏面的通道數值。通道值 $_{float}$ 就是經過轉換後，以 10 進制方式表示的實際通道數值。最大通道值 $_{hex}$ 和最小通道值 $_{hex}$ 就是通道數值用 2 補數 16 進制表示的數值上下限。最大通道值 $_{float}$ 和最小通道值 $_{float}$ 就是通道數值用 10 進制表示的數值上下限。使用者可以觀看附錄 B，來查找各模組所支援的最大通道值 $_{hex}$、最小通道值 $_{hex}$、最大通道值 $_{float}$ 和最小通道值 $_{float}$。

舉例來說，如果 CAN-8123/CAN-8223/CAN-8423 上安裝了 I-87017，且從物件字典之中，取得 I-87017 的某個 AI 通道數值爲 0x1234(+4660)，透過附錄 B 的轉換表，可知 I-87017 預設的通道值 $_{float}$ 和通道值 $_{hex}$ 範圍分別爲-10V~10V 和 0x8000(-32768) ~0x7FFF(32767)。接著再利用轉換公式，便可求出實際的通道數值，如下：

$$通道值_{float}=\left[\frac{4660-(-32768)}{32767-(-32768)}\right]*[10V-(-10V)]+(-10V)\approx 1.422V$$

此外，當類比輸入通道的數值太大，超出了模組所能量測的範圍，則物件字典內所記錄的類比通道數值就會是通道值 $_{hex}$ 的上限，反之就是通道值 $_{hex}$ 的下限。

類比輸出裝置項目

主索引	子索引	描述	型態	屬性	預設值
6411h	0h	”類比輸出(16 bit)寫入”子索引的最大定址範圍	UNSIGNED 8	唯讀	---
	1h	寫入 1h 的類比輸出通道數值	UNSIGNED 16	可讀寫	---
	...	...	...	...	...
6443	0h	“類比輸出(16 bit)錯誤模式" 子索引的最大定址範圍	UNSIGNED 8	可讀寫	---
	1h	設定 1h 的類比輸出錯誤模式	UNSIGNED 16	可讀寫	0
	...	...	...	...	---
6444	0h	“類比輸出(16 bit)錯誤值”子索引的最大定址範圍	UNSIGNED 8	可讀寫	---
	1h	設定 1h 的類比輸出錯誤值	UNSIGNED 16	可讀寫	0
	...	...	...	...	---

註： 1. 當使用者想調整 AO 通道數值的時候，同樣必須先將實際的通道數值(通道值 $_{float}$)轉換爲裝置能夠處理的格式(通道值 $_{hex}$)。轉換公式如下：

$$通道值_{hex} = \left[\frac{通道值_{float} - 最小通道值_{float}}{最大通道值_{float} - 最小通道值_{float}}\right] * [最大通道值_{hex} - 最小通道值_{hex}] + 最小通道值_{hex}$$

使用者可以翻閱附錄 B，查找各模組所支援的通道值範圍。

2. 當裝置偵測到 bus-off 或著節點守衛錯誤，又或著裝置的 NMT 狀態變成停止，此時 CAN -8123/CAN-8223/CAN-8423 就會去檢查主索引 0x6443 的物件，若其內的某個項目的某個 bit 被設定爲 1，則對應的 AO 通道在此時就會輸出記錄在主索引 0x6444 物件內相對應項目中的"類比輸出錯誤值"，若此 bit 爲 0，則對應的 AO 通道在此時就會保持原本的輸出值。

舉例來說，若裝置上有一編號 1h 的 AO 通道，，且裝置上物件字典內主索引 0x6443 和 0x6444 的物件，其內子索引 0x01 的項目分別被設定爲 0x01 和 0x0000。當裝置偵測到 bus-off 或節點守衛錯誤，又或著裝置的 NMT 狀態變成停止時，根據主索引 0x6443 物件內子索引 0x01 項目之數值 0x01，可知編號 1h 的 AO 通道會被設定爲"類比輸出錯誤值"，又根據主索引 0x6443 物件內子索引 0x01 項目之數值 0x0000，可知該 AO 通道的”類比輸出錯誤值”爲"0"。

6.4 計數器/頻率模組的物件 (I-8080 and I-8084W 專用)

主索引	子索引	描述	型態	屬性	PDO映射	預設值
3000h	0h	“讀取 32-bit 計數器/頻率”子索引的最大定址範圍	UNSIGNED 8	唯讀	否	---
	1h	讀取第一個通道的計數器/頻率	UNSIGNED 32	唯讀	可	---
	...	...	...	...		...
3001h	0h	“讀取 16-bit 計數器溢位位元”子索引的最大定址範圍	UNSIGNED 8	唯讀	否	---
	1h	讀取第一個通道的計數器溢位位元	UNSIGNED 16	唯讀	可	---
	...	...	...	...		---
3002h	0h	“重置計數器數據爲 0”子索引的最大定址範圍	UNSIGNED 8	唯讀	否	---
	1h	重置第一個通道的計數器數據	UNSIGNED 8	唯寫	可	---
	...	...	...	...		---
3003h	0h	“設定 XorRegister 爲 0 或 1”子索引的最大定址範圍	UNSIGNED 8	唯讀	否	---
	1h	第一個通道的 XorRegister 數值	UNSIGNED 8	可讀寫	否	---
	...	...	...	...		---
3004h	0h	“設定頻率模式爲 0, 1 或 2”子索引的最大定址範圍	UNSIGNED 8	唯讀	否	---
	1h	第一個通道的頻率模式	UNSIGNED 8	可讀寫	否	---
	...	...	...	...		---
3005h	0h	“設定頻率的更新時間”(單位-豪秒)子索引的最大定址範圍	UNSIGNED 8	唯讀	否	---
	1h	第一個通道頻率的更新時間	UNSIGNED 16	可讀寫	否	---
	...	...	...	...		---
3006h	0h	“啓用或停用低通濾波器”子索引的最大定址範圍	UNSIGNED 8	唯讀	否	---
	1h	第一個通道的低通濾波器	UNSIGNED 8	可讀寫	否	---
	...	...	...	...		---
3007h	0h	“低通濾波器的更時間”(單位-微秒)子索引的最大定址範圍	UNSIGNED 8	唯讀	否	---
	1h	第一個通道的低通濾波器時間	UNSIGNED 16	可讀寫	否	---
	...	...	...	...		---

爲了配置 0x3000 到 0x3007 的物件索引，你可能要參數化這個計數器模組。物件索引 0x3000 紀錄了每一個通道的計數數值，每一個子索引都對應到一個通道，使用者可以使用物件索引 0x2004 ~ 0x2007 的部分來決定計數的方式，請參考附件 B 以取得更多的詳細資訊。如果使用者選擇 Dir/Pulse、Up/Down 或 AB phase 這幾種計數模式，可能會有負數的計數值，因此使用者需要自行將 UNSIGNED 32 格式轉成 SIGNED 32 的格式。物件 0x3001 是紀錄物件 0x3000 的溢位次數，因此一個通道的總計數值是由物件 0x3000 與物件 0x3001 所組合的結果，例如，假設使用者選擇 Dir/Pulse、Up/Down 或 AB phase 其中一種模式，會有兩種情形，如果物件 0x3000 的子索引 1 是 16384，而物件 0x3001 的子索引 1 是 1，那麼總計數值則爲 1*0x80000000+16384=2147500032，如果物件 0x3000 的子索引 1 爲-8192，而物件 0x3001 的子索引 1 爲-1，那麼總計數值則爲(-1)*0x80000000-8192=-2147491840。如果使用者選擇 Frequency 或是 Up 模式，那就只會有一種情形，如果物件 0x3000 的子索引 1 是 16384，而物件 0x3001 的子索引 1 是 1，那麼總計數值爲 1*0x100000000+16384=4294983680。請注意，Frequency 模式是不會有溢位數值的。

物件 0x3002 可以幫助使用者清除計數數值的，只要在此物件中寫入 0，就可以清除紀錄在物件 0x3000 與 0x3001 中的數值了。物件 0x3003 可以設定計數器的計數觸發準位，1 爲高觸發準位(high active)，0 爲低觸發準位(low active)。物件 0x3004 則是用來設定模組的更新頻率模式的，0 爲自動模式，1 爲高頻模式，2 爲低頻模式，這三種更新頻率模式是由「頻率的更新時間」這個物件 0x3005 決定的，如果使用者要手動設定頻率的更新時間，可以使用物件 0x3005 來達到此目的。物件 0x3006 是決定使用者是否使用低通濾波器，1 是啓用，0 是停用。物件 0x3007 是用來設定會被用於低通濾波器上的脈衝寬度，假如訊號的脈衝寬度比此數值小，該訊號將會被採納進去。

6.5 脈波調變模組的物件 (I-8088W 專用)

主索引	子索引	描述	型態	屬性	PDO 映射	預設值
3100h	0h	“開始輸出脈波”子索引的最大定址範圍	UNSIGNED 8	唯讀	否	---
	1h	開始第一個通道的脈波輸出	UNSIGNED 8	可讀寫	可	---
	...	...	...	...		...
3101h	0h	“設定 16-bit 的 Burst Count”子索引的最大定址範圍	UNSIGNED 8	唯讀	否	---
	1h	設定第一個通道的 Burst Count	UNSIGNED 16	可讀寫	否	---
	...	...	...	...		---
3102h	0h	“設定 32-bit 的脈波頻率”子索引的最大定址範圍	UNSIGNED 8	唯讀	否	---
	1h	設定第一個通道的脈波頻率	UNSIGNED 32	可讀寫	否	---
	...	...	...	...		---
3103h	0h	“設定 1 ~ 999 (‰)比的 Pulse Duty”子索引的最大定址範圍	UNSIGNED 8	唯讀	否	---
	1h	設定第一個通道的 Pulse Duty	UNSIGNED 16	可讀寫	否	---
	...	...	...	...		---
3104h	0h	“硬體觸發的啓用與停用”子索引的最大定址範圍	UNSIGNED 8	唯讀	否	---
	1h	設定第一個通道爲硬體觸發	UNSIGNED 8	可讀寫	否	---
	...	...	...	...		---
3105h	0h	“同步通道的啓用與停用”子索引的最大定址範圍	UNSIGNED 8	唯讀	否	---
	1h	設定第一個通道爲同步通道	UNSIGNED 8	可讀寫	否	---
	...	...	...	...		---
3106h	0h	“開始同步脈波輸出”子索引的最大定址範圍	UNSIGNED 8	唯讀	否	---
	1h	開始第一個 PWM 模組的同步脈波輸出(插槽號碼最小的那個 PWM 模組)	UNSIGNED 8	可讀寫	可	---
	...	...	...	...		---

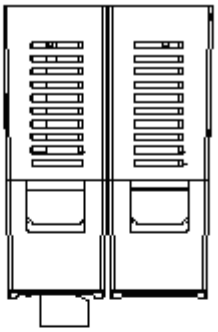
爲了配置 0x3100 到 0x3106 的物件索引，你可能要參數化這個脈波調變模組。物件 0x3100 可以控制模組的每一個通道要開始或是停止脈波的輸出，每一個子索引都對應到一個通道，使用者可使用物件 0x2004 ~ 0x2007 來決定每個插槽的 PWM 的方式，更多詳細的資訊請參考附件 B。假如使用者選擇 Burst Counting 模式，那麼就必須要設定物件 0x3101 以便決定使用者需要輸出多少脈波，使用者可以設定物件 0x3101 爲 1 ~ 65535 次，並使用物件 0x3100 來開始或停止脈波輸出，每一次當物件 0x3100 設定爲 1 的時候，該通道就會輸出所指定的脈波數目一輪，例如，使用者設定通道 0 爲 Burst Counting 模式，並且設定 0x3101 的子索引 1 爲 100，當使用者設定物件 0x3100 的子索引 1 爲 1 的時候，該通道將會輸出 100 次的脈波。如果使用者選擇 Continue Counting 模式，那麼物件 0x3101 就沒有作用了，此時當使用者設定 0x3100 的子索引 1 爲 1 時，該通道將會開始連續不斷地輸出脈波，直到該物件被設定成 0。如果想要更改脈波的頻率，可以設定物件 0x3102，可設定的數值從 100 ~ 5000000，單位爲 0.1 赫茲(也就是 10 赫茲 ~ 500k 赫茲)。

物件 0x3103 是 pulse duty 的千分比，如果將此物件設定爲 300，意思就是脈波的高準位佔了整個脈波寬度的 300‰，而低準位則佔了 700‰。物件 0x3104 可以設定將 PWM 模組的 DI 腳位當成硬體觸發腳位。當設定物件 0x3104 的子索引 2 爲 1 時，意思是該 DI 腳位將會喪失 DI 的功能，並且變成硬體的觸發腳位，此時如果有一個 5V ~ 30V 間的訊號進來該 DI 腳位，該 DI 通道所對應的脈波通道將會開始輸出，直到該 DI 訊號被清除爲止。

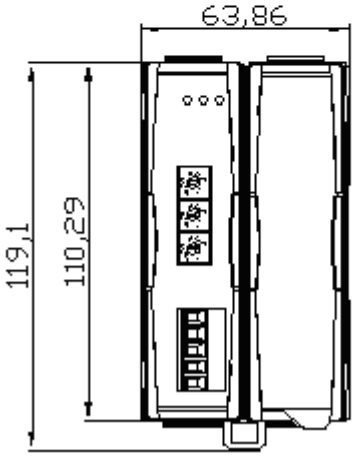
物件 0x3105 和 0x3106 可以控制模組的所有脈波通道同步輸出，如果使用者想要讓第 0 到第 3 的脈波通道同步輸出脈波，可以設定物件 0x3105 的子索引 1 ~ 4 爲 1，並將其他子索引設定爲 0，然後設定物件 0x3106 的子索引 1 爲 1，如此這四個通道(通道 0 ~ 3)將會同時開始輸出脈波(它們的第一個由低到高的邊緣的時間會相同，而接下來的周期則會根據物件 0x3102 所設定的頻率有所不同)。請注意，物件 0x3106 中，不一樣的子索引代表著不一樣的 PWM 模組，假如有兩個 PWM 模組在同一個 CAN-8x23 上，那麼最大的子索引號碼就是 2，其中子索引 1 是指插槽號碼較小的那個 PWM 模組，子索引 2 則是插槽號碼比較大的那個 PWM 模組。

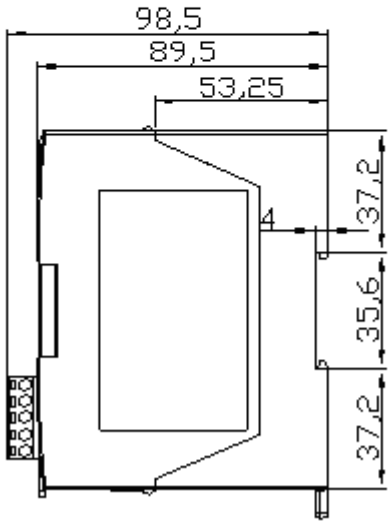
附錄 A：機構圖

6.6 A.1 CAN-8123 機構圖

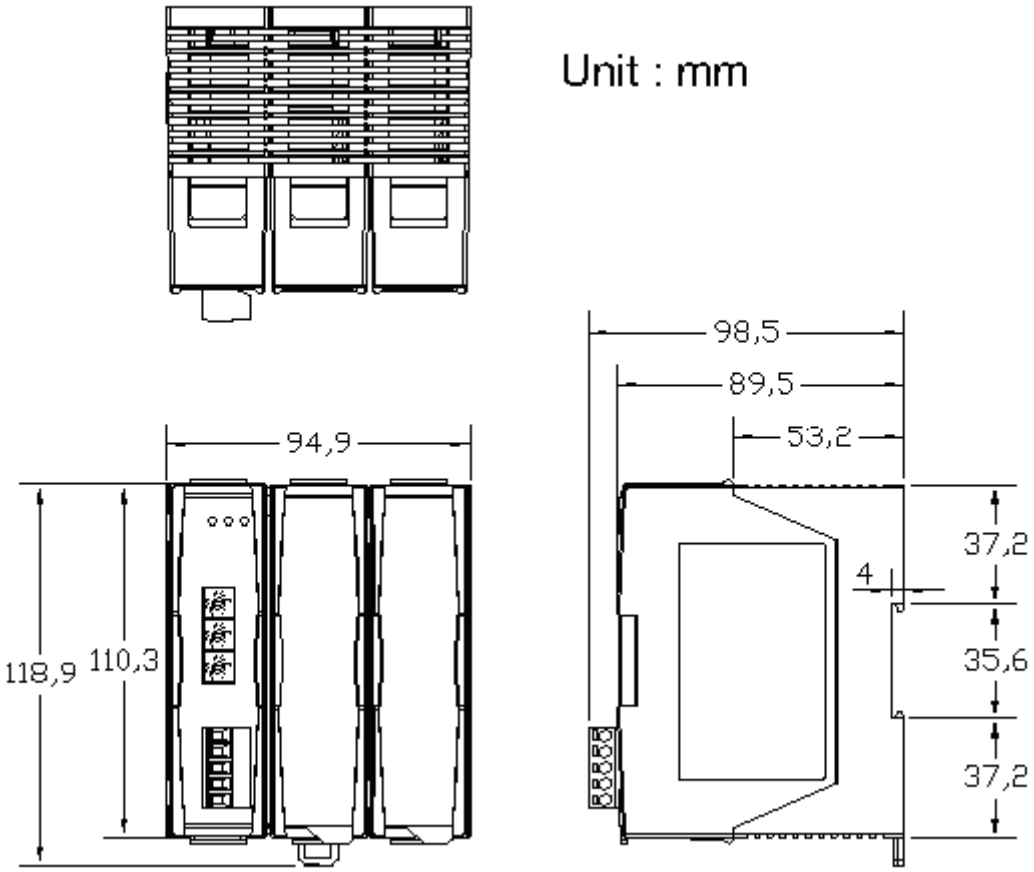


Unit :mm

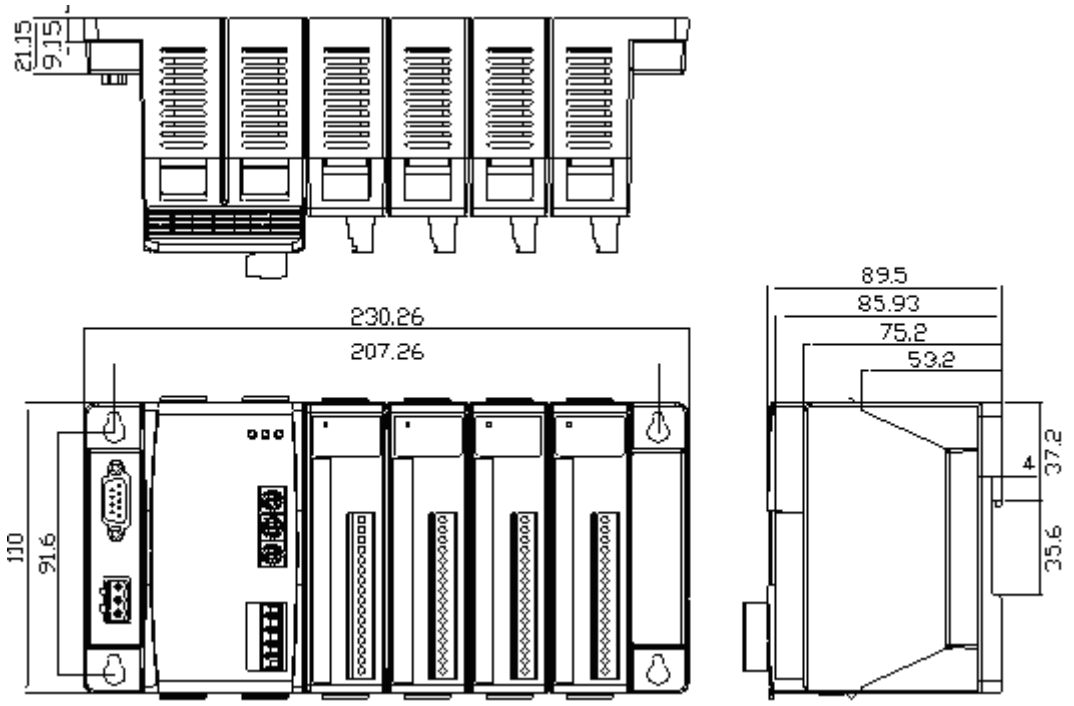




6.7 A.2 CAN-8223 機構圖



6.8 A.3 CAN-8423 機構圖



附錄 B：I/O 模組的支援與 Type Code 轉換表

In order to look up the configuration parameters of each slot module more quickly, the transformation table has separated into several parts according to the name of slot module. They are given below.

I-87K module	I-8K module
I-87013	I-8017HS/I-8017HW
I-87015/I-87015P	I-8024/I-8024W
I-87017/I-87017R/I-87017W/I-87017RW	I-8050
I-87017RC	I-8080/I-8084W
I-87018/I-87018RW/I-87018W/I-87018Z	I-8088W
I-87019RW	
Thermocouple of I-87018/I-87019 Series	
I-87022	
I-87024/I-87024W	
I-87026	
I-87024C/I-87028C	

註： 有關這些模組的接線方式，使用者可參照各模組的使用手冊，或是泓格科技的網站。

I-87013/ I-87015 熱敏電組型態定義				回到列表
範圍碼 (Hex)	熱敏電阻型態	資料格式	最大數值	最小數值
20 (預設)	Platinum 100 a = 0.00385	輸入範圍	+100.00℃	-100.00℃
		2 的補數 HEX	7FFF	8000
21	Platinum 100 a = 0.00385	輸入範圍	+100.00℃	+000.00℃
		2 的補數 HEX	7FFF	0000
22	Platinum 100 a = 0.00385	輸入範圍	+200.00℃	+000.00℃
		2 的補數 HEX	7FFF	0000
23	Platinum 100 a = 0.00385	輸入範圍	+600.00℃	+000.00℃
		2 的補數 HEX	7FFF	0000
24	Platinum 100 a = 0.003916	輸入範圍	+100.00℃	-100.00℃
		2 的補數 HEX	7FFF	8000
25	Platinum 100 a = 0.003916	輸入範圍	+100.00℃	+000.00℃
		2 的補數 HEX	7FFF	0000
26	Platinum 100 a = 0.003916	輸入範圍	+200.00℃	+000.00℃
		2 的補數 HEX	7FFF	0000
27	Platinum 100 a = 0.003916	輸入範圍	+600.00℃	+000.00℃
		2 的補數 HEX	7FFF	0000
28	Nickel 120	輸入範圍	+100.00℃	-080.00℃
		2 的補數 HEX	7FFF	999A
29	Nickel 120	輸入範圍	+100.00℃	+000.00℃
		2 的補數 HEX	7FFF	0000
2A	Platinum 1000 a = 0.00385	輸入範圍	+600.00℃	-200.00℃
		2 的補數 HEX	7FFF	D556
2B*1	Cu 100 a = 0.00421	輸入範圍	+150.00℃	-020.00℃
		2 的補數 HEX	7FFF	EEEF
2C*1	Cu 100 a = 0.00421	輸入範圍	+200.00℃	-000.00℃
		2 的補數 HEX	7FFF	0000
2D*1	Cu 1000 a = 0.00421	輸入範圍	+150.00℃	-020.00℃
		2 的補數 HEX	7FFF	EEEF

2E*2	Pt 100 a = 0.00385	輸入範圍	+200.00℃	-200.00℃
		2 的補數 HEX	7FFF	8000
2F*2	Pt 100 a = 0.003916	輸入範圍	+200.00℃	-200.00℃
		2 的補數 HEX	7FFF	8000
80*2	Pt 100 a = 0.00385	輸入範圍	+600.00℃	-200.00℃
		2 的補數 HEX	7FFF	D556
81*2	Pt 100 a = 0.003916	輸入範圍	+600.00℃	-200.00℃
		2 的補數 HEX	7FFF	D556

注意：
* 1：2B、2C、和 2D 只適用於 I-87015.
* 2：2E、2F、80、和 81 只適用於韌體版本爲 A1.10 (含)以後的 I-87015，以及韌體版本爲 B1.3 (含)以後的 I-87013。

I-87017 系列 型態 08 到 0D 定義 (不含 I-87017RC) 回到列表

範圍碼(Hex)	輸入範圍	資料格式	滿刻度	負數滿刻度
08 (預設)	-10V to +10V	輸入範圍	+10.000 V	-10.000 V
		2 的補數 HEX	7FFF	8000
09	-5V to +5V	輸入範圍	+5.0000 V	-5.0000 V
		2 的補數 HEX	7FFF	8000
0A	-1V to +1V	輸入範圍	+1.0000 V	-1.0000 V
		2 的補數 HEX	7FFF	8000
0B	-500mV to +500mV	輸入範圍	+500.00 mV	-500.00 mV
		2 的補數 HEX	7FFF	8000
0C	-150mV to +150mV	輸入範圍	+150.00 mV	-150.00 mV
		2 的補數 HEX	7FFF	8000
0D*1	-20mA to +20mA	輸入範圍	+20.000 mA	-20.000 mA
		2 的補數 HEX	7FFF	8000

注意：
*1：當 I-87017 和 I-87017R 連接到電流源並設型態爲 0D 時，需額外並接一個 125Ω 的精密電阻。

I-87017RC 型態 07 到 1A 定義				回到列表
範圍碼(Hex)	**輸入範圍**	**資料格式**	**滿刻度**	**負數滿刻度**
07	-4mA to +20mA	輸入範圍	+04.000 mA	+20.000 mA
		2 的補數 HEX	7FFF	8000
0D (預設)	-20mA to +20mA	輸入範圍	+20.000 mA	-20.000 mA
		2 的補數 HEX	7FFF	8000
1A	+0A to +20mA	輸入範圍	+00.000 mA	+20.000 mA
		2 的補數 HEX	7FFF	8000

注意：

1. I-87017RC 的每一個通道均已內建 125Ω 的精密電阻，因此當連接到電流源的時候不需要再外加其他電阻。

I-87018 系列 型態 00 到 06 定義				回到列表
範圍碼(Hex)	**輸入範圍**	**資料格式**	**滿刻度**	**負數滿刻度**
00	-15mV to +15mV	輸入範圍	+15.000 mV	-15.000 mV
		2 的補數 HEX	7FFF	8000
01	-50mV to +50mV	輸入範圍	+50.000 mV	-50.000 mV
		2 的補數 HEX	7FFF	8000
02	-100mV to +100mV	輸入範圍	+100.00 mV	-100.00 mV
		2 的補數 HEX	7FFF	8000
03	-500mV to +500mV	輸入範圍	+500.00 mV	-500.00 mV
		2 的補數 HEX	7FFF	8000
04	-1V to +1V	輸入範圍	+1.0000 V	-1.0000 V
		2 的補數 HEX	7FFF	8000
05 (預設)	-2.5V to +2.5V	輸入範圍	+2.5000 V	-2.5000 V
		2 的補數 HEX	7FFF	8000
06*1	-20mA to +20mA	輸入範圍	+20.000 mA	-20.000 mA
		2 的補數 HEX	7FFF	8000

注意：

*1：當 I-87018 和 I-87018R 連接到電流源並設型態爲 06 時，需額外並接一個 125Ω 的精密電阻。

I-87019R 型態 00 到 19 定義				回到列表
範圍碼(Hex)	輸入範圍	資料格式	滿刻度	負數滿刻度
00	-15mV to +15mV	輸入範圍	+15.000 mV	-15.000 mV
		2 的補數 HEX	7FFF	8000
01	-50mV to +50mV	輸入範圍	+50.000 mV	-50.000 mV
		2 的補數 HEX	7FFF	8000
02	-100mV to +100mV	輸入範圍	+100.00 mV	-100.00 mV
		2 的補數 HEX	7FFF	8000
03	-500mV to +500mV	輸入範圍	+500.00 mV	-500.00 mV
		2 的補數 HEX	7FFF	8000
04	-1V to +1V	輸入範圍	+1.0000 V	-1.0000 V
		2 的補數 HEX	7FFF	8000
05	-2.5V to +2.5V	輸入範圍	+2.5000 V	-2.5000 V
		2 的補數 HEX	7FFF	8000
06	-20mA to +20mA 需外加 125Ω電阻	輸入範圍	+20.000 mA	-20.000 mA
		2 的補數 HEX	7FFF	8000
08 (預設)	-10V to +10V	輸入範圍	+10.000 V	-10.000 V
		2 的補數 HEX	7FFF	8000
09	-5V to +5V	輸入範圍	+5.0000 V	-5.0000 V
		2 的補數 HEX	7FFF	8000
0A	-1V to +1V	輸入範圍	+1.0000 V	-1.0000 V
		2 的補數 HEX	7FFF	8000
0B	-500mV to +500mV	輸入範圍	+500.00 mV	-500.00 mV
		2 的補數 HEX	7FFF	8000
0C	-150mV to +150mV	輸入範圍	+150.00 mV	-150.00 mV
		2 的補數 HEX	7FFF	8000
0D	-20mA to +20mA 需外加 125Ω電阻	輸入範圍	+20.000 mA	-20.000 mA
		2 的補數 HEX	7FFF	8000

I-87018/ 87018R/ 87019R 熱電偶型態定義				回到列表
範圍碼(Hex)	熱電偶型態	資料格式	最大值	最小值
0E	J Type	輸入範圍	+760.00℃	-210.00℃
		2 的補數 HEX	7FFF	DCA2
0F	K Type	輸入範圍	+1372.0℃	-0270.0℃
		2 的補數 HEX	7FFF	E6D0
10	T Type	輸入範圍	+400.00℃	-270.00℃
		2 的補數 HEX	7FFF	A99A
11	E Type	輸入範圍	+1000.0℃	-0270.0℃
		2 的補數 HEX	7FFF	DD71
12	R Type	輸入範圍	+1768.0℃	+0000.0℃
		2 的補數 HEX	7FFF	0000
13	S Type	輸入範圍	+1768.0℃	+0000.0℃
		2 的補數 HEX	7FFF	0000
14	B Type	輸入範圍	+1820.0℃	+0000.0℃
		2 的補數 HEX	7FFF	0000
15	N Type	輸入範圍	+1300.0℃	-0270.0℃
		2 的補數 HEX	7FFF	E56B
16	C Type	輸入範圍	+2320.0℃	+0000.0℃
		2 的補數 HEX	7FFF	0000
17	L Type	輸入範圍	+800.00℃	-200.00℃
		2 的補數 HEX	7FFF	E000
18	M Type	輸入範圍	+100.00℃	-200.00℃
		2 的補數 HEX	4000	8000
19	L Type DIN43710	輸入範圍	+900.00℃	-200.00℃
		2 的補數 HEX	7FFF	E38F

I-87022 類比輸出型態定義				回到列表
範圍碼(Hex)	輸出範圍	資料格式	最大值	最小值
0	0 to 20mA	輸出範圍	20.000 mA	00.000 mA
		16 進位數值	FFF	000
1	4 to 20mA	輸出範圍	20.000 mA	04.000 mA
		16 進位數值	FFF	000
2 (預設)	0 to 10V	輸出範圍	10.000 V	00.000 V
		16 進位數值	FFF	000

I-87024 類比輸出型態定義				回到列表
範圍碼(Hex)	輸出範圍	資料格式	最大值	最小值
30	0 to 20mA	輸出範圍	+20.000 mA	+00.000 mA
		2 的補數 HEX	0x7FFF	0
31	4 to 20mA	輸出範圍	+20.000 mA	+04.000 mA
		2 的補數 HEX	0x7FFF	0
32	0 to 10V	輸出範圍	+10.000 V	+00.000 V
		2 的補數 HEX	0x7FFF	0
33 (預設)	-10 to 10V	輸出範圍	+10.000 V	-10.000 V
		2 的補數 HEX	0x7FFF	0x8000
34	0 to 5V	輸出範圍	+05.000 V	+00.000 V
		2 的補數 HEX	0x7FFF	0
35	-5 to 5V	輸出範圍	+05.000 V	-05.000 V
		2 的補數 HEX	0x7FFF	0x8000

I-87026 類比輸出型態定義				回到列表
範圍碼(Hex)	輸出範圍	資料格式	最大值	最小值
0	0 to 20mA	輸出範圍	20.000 mA	00.000 mA
		16 進位數值	FFFF	0000
1	4 to 20mA	輸出範圍	20.000 mA	04.000 mA
		16 進位數值	FFFF	0000
2 (預設)	0 to 10V	輸出範圍	10.000 V	00.000 V
		16 進位數值	FFFF	0000

I-87024C/I-87028C 類比輸出型態定義				回到列表
範圍碼(Hex)	輸出範圍	資料格式	最大值	最小值
0 (預設)	0 to 20mA	輸出範圍	+20.000 mA	+00.000 mA
		16 進位數值	FFF	000
1	4 to 20mA	輸出範圍	+20.000 mA	+04.000 mA
		16 進位數值	FFF	000

I-8024/I-8024W 類比輸出型態定義				回到列表
範圍碼(Hex)	輸出範圍	資料格式	最大值	最小值
0 (預設)	-10 to 10V	輸出範圍	+10.000 V	-100.000 V
		16 進位數值	7FFF	8000
1	0 to 20mA	輸出範圍	+20.000 mA	+00.000 mA
		16 進位數值	7FFF	8000

I-8017HS/I-8017HW 類比輸入型態定義				回到列表
範圍碼(Hex)	輸入範圍	資料格式	最大值	最小值
0 (預設)	-10 to 10V	輸入範圍	+10.000 V	-10.000 V
		16 進位數值	1FFF	2000
1	-5 to 5V	輸入範圍	+5.000 V	-5.000 V
		16 進位數值	1FFF	2000
2	-2.5V to +2.5V	輸入範圍	+2.500 V	-2.5000 V
		16 進位數值	1FFF	2000
3	-1.25V to +1.25V	輸入範圍	+1.250 V	-1.250 V
		16 進位數值	1FFF	2000
4	-20 mA to +20 mA	輸入範圍	+20.000 mA	-20.000 mA
		16 進位數值	1FFF	2000

I-8080/ 8084W 計數輸入型態定義				回到列表
範圍碼(Hex)	計數類型	通道數目	最大值	最小值
0 *1	Dir/Pulse Counter	4	2147483647	-2147483648
			1FFFFFFF	80000000
1 *1	Up/Down Counter	4	2147483647	-2147483648
			1FFFFFFF	80000000
2	Frequency	8	-	-
			-	-
3 (預設)	Up Counter	8	4294967295	0
			FFFFFFFF	00000000
4 *1 (I-8084W)	AB Phase	4	2147483647	-2147483648
			1FFFFFFF	80000000

注意：

*1：所有參數的子索引以及輸入通道仍然是 8，但是子索引 1 與子索引 2 相等，子索引 3 與子索引 4 相等，其後依此類推。

I-8088W 數位脈波調變輸出型態定義				回到列表
範圍碼(Hex)	計數類型	通道數目	最大值	最小值
0	Burst Counter	8	65535	1
			FFFF	1
1 (預設)	Continue Counter	8	---	---
			---	---

附錄 C：I-8050 數位輸出入模組的定義方式

I-8050是一個可調整的數位輸出入模組，使用者可以決定哪一個通道要當做數位輸入，哪一個通道要當做數位輸出。在CAN-8x23裡面，使用者可以利用設定物件0x2004 ~ 0x2007的型態碼來達到此一目的，物件0x2004、0x2005、0x2006、與0x2007分別代表插在CAN-8x23上的插槽0、插槽1、插槽2、與插槽3的模組。

例如 I-8050 模組插在 CAN-8423 的插槽 0 上，使用者可以透過設定物件 0x2004 來決定 I-8050 的通道型態，物件 0x2004 的子索引 1 可以控制第 0 ~ 7 的通道型態，而子索引 2 可以控制第 8 ~ 15 的通道型態，假如此物件中的某一個位元被設定成 1，代表該位元相對應的通道將當作 DI 使用，所以如果使用者設定子索引 1 爲 0x77，表示只有第 3 和第 7 通道是 DO 通道，其餘則爲 DI 通道，I-8050 的所有通道的出廠預設型態均爲 DI 通道。

回到列表

附錄 D：專有名詞中英對照

爲避免在翻譯時，對原文專有名詞文義上的扭曲，於下按照中文筆劃順序列出較常見專有名詞的中英對照，方便讀者查詢。

2 補數/16 進制	2's complement hex
子索引	sub-index
工具程式	utility
主索引	index
主端	master
生產者	producer
用戶端	client
同步	synchronous
在線	on-line
字組	byte
字節	octet
位元	bit
伺服端	server
即時	real-time
物件	object
物件字典	object dictionary
非同步	asynchronous
非循環	acyclic
映射	mapping
消費者	consumer
訊息	message
埠	port
組態/設定	configuration
終端電阻	terminal resistance
通訊協定	communication protocol
通訊物件	COB (Communication Object)
通道	channel
幀	frame

循環	cyclic
描述文件	profile
項目	entry
匯流排	bus
節點	node
跳線器	jummper
僕端	slave
遠端傳送要求	RTR (Remote Transmission Request)
數位輸入	digital input
數位輸出	digital output
鮑率	baud rate
擴充插槽	expansion slot
擴充模組	slot module
離線	off-line
類比輸入	analog input
類比輸出	analog output
操作	operational
預操作	pre-operational
停止	stop

附錄 E：英文縮寫對照表

爲求文章通順，部份英文名詞是以縮寫的方式來呈現，於下按照英文字母順序列出這些專有名詞的縮寫對照表，方便讀者查詢。

AI	Analog Intput
AO	Analog Output
COB	Communication Object
DI	Digital Input
DO	Digital Output
DS	Draft Standard
EDS	Electronic Data Sheet
EMCY	Emergency
I/O	Input/Output
LSB	Least Significant Bit
MSB	Most Significant Bit
NMT	Network Management
PDO	Process Data Object
RTD	Resistance Temperature Detector
RTR	Remote Transmit Request
Rx	Receive
SDO	Service Data Object
SYNC	Synchronous
Tx	Transmit