

---

# *I-7565-H1 / I-7565-H2*

## *高效能 USB/CAN 轉換器*

### 使用手冊

#### Warranty

All products manufactured by ICP DAS are under warranty regarding defective materials for a period of one year from the date of delivery to the original purchaser.

#### Warning

ICP DAS assumes no liability for damages resulting from the use of this product. ICP DAS reserves the right to change this manual at any time without notice. The information furnished by ICP DAS is believed to be accurate and reliable. However, no responsibility is assumed by ICP DAS for its use, or for any infringements of patents or other rights of third parties resulting from its use.

#### Copyright

Copyright 2015 by ICP DAS. All rights are reserved.

#### Trademark

The names used for identification only may be registered trademarks of their respective companies.

---

# 目錄

<b>1. 簡介</b>	<b>5</b>
1.1 特色	6
1.2 規格	6
<b>2. 硬體</b>	<b>8</b>
2.1 方塊圖	8
2.2 CAN 埠接腳定義	9
2.3 硬體線路連接	10
2.4 終端電阻設定	11
2.5 Init / Normal 指撥開關	12
2.5.1 Firmware Update 模式	12
2.5.2 Firmware Operation 模式	13
2.6 LED 指示燈	14
2.7 線材選擇	15
<b>3. 驅動程式安裝</b>	<b>17</b>
3.1 自動安裝 I-7565-H1/H2 驅動程式	17
3.2 手動安裝 I-7565-H1/H2 驅動程式	18
3.3 驗證驅動程式安裝	21
3.4 移除 I-7565-H1/H2 驅動程式	22
<b>4. 軟體工具</b>	<b>23</b>
4.1 INI 檔案功能	23
4.2 連線功能	23
4.3 通訊功能	26
4.4 設定功能	32
4.4.1 模組設定功能	32
4.4.2 進階設定功能	36
4.4.3 額外設定功能	40
4.5 資料記錄功能	41
4.6 狀態列功能	43
<b>5. API 函式庫 -- VCI_CAN.dll</b>	<b>45</b>
5.1 API Library 概觀	45
5.2 API Library 功能表	46
5.3 使用者應用程式開發流程	48
5.4 Init Function	49
5.4.1 VCI_OpenCAN	49
5.4.2 VCI_CloseCAN	50
5.5 Module Config Function	51

5.5.1	VCI_Set_CANFID .....	51
5.5.2	VCI_Get_CANFID .....	53
5.5.3	VCI_Get_CANStatus .....	55
5.5.4	VCI_Clr_BufOverflowLED .....	57
5.5.5	VCI_Get_MODInfo .....	58
5.5.6	VCI_Rst_MOD .....	59
5.5.7	VCI_Set_MOD_Ex .....	60
5.6	Communication Function .....	61
5.6.1	VCI_SendCANMsg .....	61
5.6.2	VCI_RecvCANMsg .....	63
5.6.3	VCI_EnableHWCyclicTxMsg .....	65
5.6.4	VCI_DisableHWCyclicTxMsg .....	67
5.6.5	VCI_EnableHWCyclicTxMsgNo .....	68
5.6.6	VCI_EnableHWCyclicTxMsgNo_Ex .....	70
5.6.7	VCI_DisableHWCyclicTxMsgNo .....	72
5.7	Software Buffer Function .....	73
5.7.1	VCI_Get_RxMsgCnt .....	73
5.7.2	VCI_Get_RxMsgBufIsFull .....	74
5.7.3	VCI_Clr_RxMsgBuf .....	75
5.7.4	VCI_Get_TxMsgCnt .....	76
5.7.5	VCI_Clr_TxMsgBuf .....	77
5.7.6	VCI_Get_TxSentCnt .....	78
5.7.7	VCI_Clr_TxSentCnt .....	79
5.8	User Defined ISR Function .....	80
5.8.1	VCI_Set_UserDefISR .....	80
5.8.2	VCI_Clr_UserDefISR .....	81
5.8.3	VCI_Get_ISRCANData .....	81
5.9	Other Function .....	83
5.9.1	VCI_Get_DIIVer .....	83
5.9.2	VCI_DoEvents .....	84
5.10	Extended Function .....	85
5.10.1	VCI_OpenCAN_Ex .....	85
5.10.2	VCI_Get_CANBaud_BitTime .....	87
5.11	回傳代碼 .....	88
<b>6.</b>	<b>API 函式庫 -- mVCI_CAN.dll .....</b>	<b>89</b>
6.1	VC 專案 .....	89
6.2	VB 專案 .....	90
6.3	.Net 專案 .....	92

<b>7. 常問問題 (FAQ)</b> .....	<b>93</b>
7.1 模組連線問題 .....	93
Q1 : "Invalid port number" 錯誤訊息 ? .....	93
Q2 : "The device is not open" 錯誤訊息 ? .....	94
Q3 : "Device doesn't Exist" 錯誤訊息 ? .....	95
Q4 : "Could not set comm state" 錯誤訊息 ? .....	95
7.2 CAN 鮑率問題 .....	96
7.3 CAN 網路中發生 CAN ID 重覆問題 .....	98
7.4 電腦自動重新開機問題 .....	98
7.5 最大資料傳輸率 (fps)問題.....	98
7.6 資料遺失問題 .....	98
7.7 一台電腦能插多少模組的問題 .....	99
7.8 安裝驅動程式時間過久問題 .....	99
7.9 支援的 CAN Filter-ID 編號問題.....	100
7.10 其它問題.....	101
7.11 Windows 7 相關問題 .....	101
7.12 I-7565-H1/H2 模組無法接收 CAN 訊息封包.....	103
7.13 I-7565-H1/H2 有提供 LabVIEW Driver .....	104
7.14 如何調整 I-7565-H1/H2 之鮑率 Bit-Timing 參數值 ? .....	104
7.15 如何啓動 I-7565-H1/H2 之 CAN 錯誤封包訊息顯示功能 ? .....	104
7.16 新功能 - "OverWrite", 工具軟體 v1.09 版以上支援 ? .....	106
7.17 新功能 - "Symbolic", 工具軟體 v1.10 版以上支援 ? .....	107
7.18 如何使用 I-7565-H2 精確地傳送 CAN 訊息 ? .....	107
7.19 如何監聽 CAN 網路封包訊息，而不影響原本 CAN 網路通訊 ? .....	108
7.20 如何取得目前 CAN 網路之封包流量 ? .....	109
7.21 如何讓 I-7565-H1/H2 成爲 CAN 資料記錄器 ? .....	110
7.22 如何立即接收到指定之 CAN-ID 訊息資料 ? .....	111
7.23 API 函式庫是否支援 Visual Studio Express 免費開發軟體 ? .....	112
7.24 DotNet 範例在 Win7_x64 平台執行時，會出現"試圖載入格式錯誤 0x8007000B"或"System.NullReferenceException"錯誤訊息 ? .....	112
7.25 Windows 10 相關問題 .....	113
7.26 使用 VCI_Set_CANFID 函式，當寫設定參數錯誤時，可能造成模組無 法正常開啓 COM? .....	118
7.27 CAN 通訊記錄檔，如何轉成 Excel 檔開啓? .....	119
<b>8. Linux 平台使用手冊</b> .....	<b>121</b>
<b>9. 版本歷史</b> .....	<b>122</b>

# 1. 簡介

I-7565-H1 與 I-7565-H2 分別具備一個和二個 USB/CAN 高效率智能轉換通道。它們提供更優於 I-7565 的傳輸效能、符合 CAN 2.0A/2.0B 協定標準，與提供傳輸速率設定範圍 5Kbps 到 1Mbps，除此之外，它支援自訂速率傳輸速率。當 I-7565-H1/H2 連接到電腦時，電腦會自動載入相關的驅動程式(符合隨插即用的特性)。因此，使用者可輕易又快速的收集以及處理 CAN 控制網路上的資料。I-7565-H1/H2 目前應用場合相當廣泛，可運用在 CAN 匯流排監控、建構自動化、遠端資料收集、環境監控、實驗儀器與研究、工廠自動化...等。

USB/CAN 模組可運用在下列的應用架構上：

- (1) **I-7565-H1:** 提供 1 埠高效率智能 USB/CAN 轉換器。
- (2) **I-7565-H2:** 提供 2 埠高效率智能 USB/CAN 轉換器。



圖 1-1: I-7565-H1/H2 的應用



I-7565-H1 應用

I-7565-H2 應用

---

## 1.1 特色

- 符合 RoHS 無鉛製程規範。
- 完全相容於 USB 1.1/2.0 (全速-Full Speed)。
- 完全相容於 ISO 11898-2 標準。
- 支援 CAN2.0A 與 CAN2.0B 協定標準。
- 不需外接電源，直接由 USB 埠電源提供。
- 整合 1 個或 2 個 CAN 匯流排介面。
- 可選擇 CAN 鮑率範圍從 5Kbps 至 1Mbps，或是可自行設定鮑率。
- 支援 CAN ID 過濾設定功能。
- 支援 Listen Only Mode (LOM) 模式。(韌體 v1.05 以後版本)
- 支援 5 組硬體時鐘傳送 CAN 訊息，可提供較電腦更精準時鐘傳送訊息。(韌體 v1.05 以後版本)
- 支援 CAN Error Frame (包含 Bus/ArbitraionLost Error) 資訊顯示功能。(韌體 v1.07 以後版本)
- 支援 CAN 鮑率之 Bit-Timing (Tseg2) 設定功能。(韌體 v1.07 以後版本)
- 接收到之 CAN 訊息時間戳記，精準度可達 1 毫秒。
- 可透過 USB 更新韌體。
- 提供 Utility 工具，讓使用者更方便地進行模組設定與通訊測試。
- 提供 API 函式庫。
- 提供硬體唯一序號功能。(韌體 v1.04 以後版本)
- 提供 PWR / RUN / ERR 三種指示燈。
- 內建 120 歐姆終端電阻。
- 單一模組的最大資料流量為每秒 3000 個資料訊框(視使用者的電腦效能而定)。
- I-7565-H1 資料緩衝區提供 256 資料訊框；I-7565-H2 單一 CAN 埠的資料緩衝區提供 128 資料訊框。
- 內建 Watchdog 監測機制。
- 提供 Windows 2000/XP、Win7(32/64bit)、Linux 與 WinCE (即將發佈) 驅動程式。

## 1.2 規格

### [ USB ]

- 輸入埠：USB (USB Type B)
- 相容性：標準 USB 1.1 與 2.0

- 
- 驅動程式支援：Windows 2000/XP、Win7(32/64bit)、Linux 與 WinCE (即將發佈)。

### [ CAN ]

- CAN 埠的連接介面：
  - I-7565-H1：9-pin D-sub male (公)
  - I-7565-H2：10-pin 端子
- CAN 鮑率：5K ~ 1Mbps 或是由使用者自訂鮑率
- 隔離電壓：CAN 端提供 3000Vrms

### [ 模組 ]

- 體積：108mm x 72mm x 35mm (H x W x D)
- 工作溫度：-25 to 75°C (-13 to 167°F);
- 儲存溫度：-40 to 80°C (-40 to 176°F);
- 濕度：5 to 95%, 非冷凝。
- 指示燈號：
  - PWR LED – 電源
  - RUN LED – 通訊
  - ERR LED – 錯誤

### [ Utility 工具 / 軟體開發工具: ]

- 提供使用自訂的鮑率傳輸速率 / CAN ID 過濾設定。
- 簡易的傳輸 / 接收測試功能，並顯示每一個接收 CAN 訊息的時戳。
- 提供 CAN 訊息記錄功能。
- 可透過模組內部硬體高精度時鐘，準確地傳輸 CAN 訊息。
- 可遠端 檢查/重置 模組與取得目前 CAN 匯流排的訊息流量。
- 藉由我們提供的軟體開發工具，使用者可輕易地的開發系統。

### [ 應用場合 ]

- 工廠自動化。
- 建構自動化。
- 家電自動化。
- 控制系統。
- 監測系統。
- 車輛自動化。

## 2. 硬體

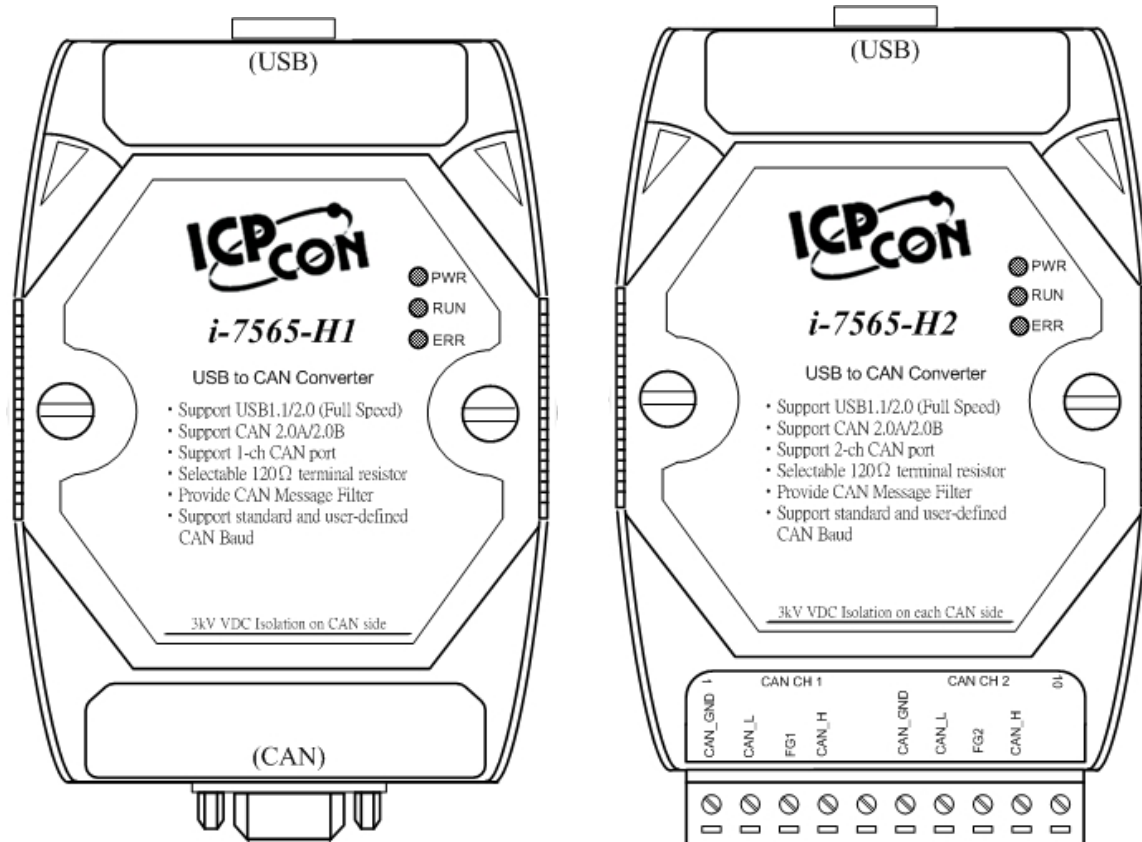


圖 2-1: I-7565-H1/H2 硬體外觀

### 2.1 方塊圖

圖 2-2 描述 I-7565-H1/H2 模組功能的方塊圖。CAN 埠具備 3000Vrms 隔離電壓保護。

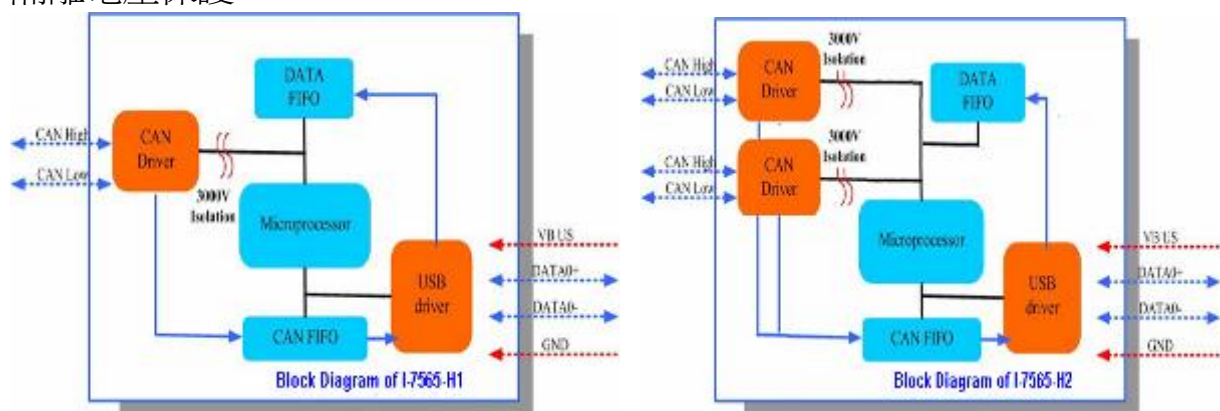


圖 2-2: I-7565-H1 / I-7565-H2 的方塊圖



## 2.2 CAN 埠接腳定義

表 1: I-7565-H1 之 CAN 埠的 DB9 公接頭

端子	2-wire CAN
1	Not Connect
2	<b>CAN Low</b>
3	<b>CAN Ground</b>
4	Not Connect
5	
6	<b>CAN Ground</b>
7	<b>CAN High</b>
8	Not Connect
9	

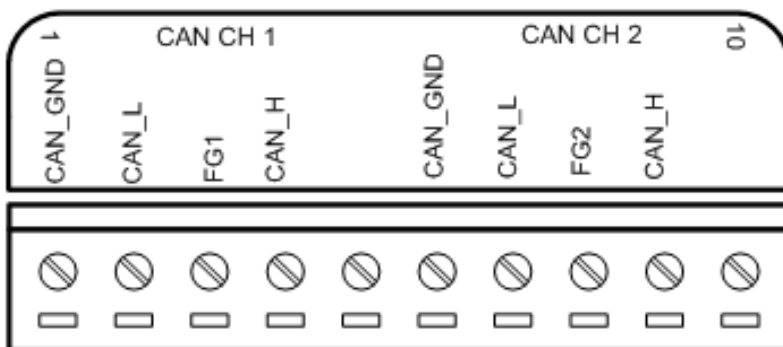
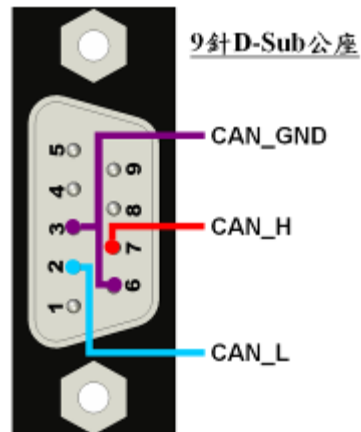


圖 2-3: I-7565-H2 之 CAN 埠 10-PIN 端子接腳定義

## 2.3 硬體線路連接

I-7565-H1(DB-9 公)CAN 埠的接腳定義完全符合 CANopen DS102 與 DeviceNet(規格書中的附錄 C)所規範的規格；I-7565-H1/H2 之間的硬體線路連接如圖 2-4 所示：

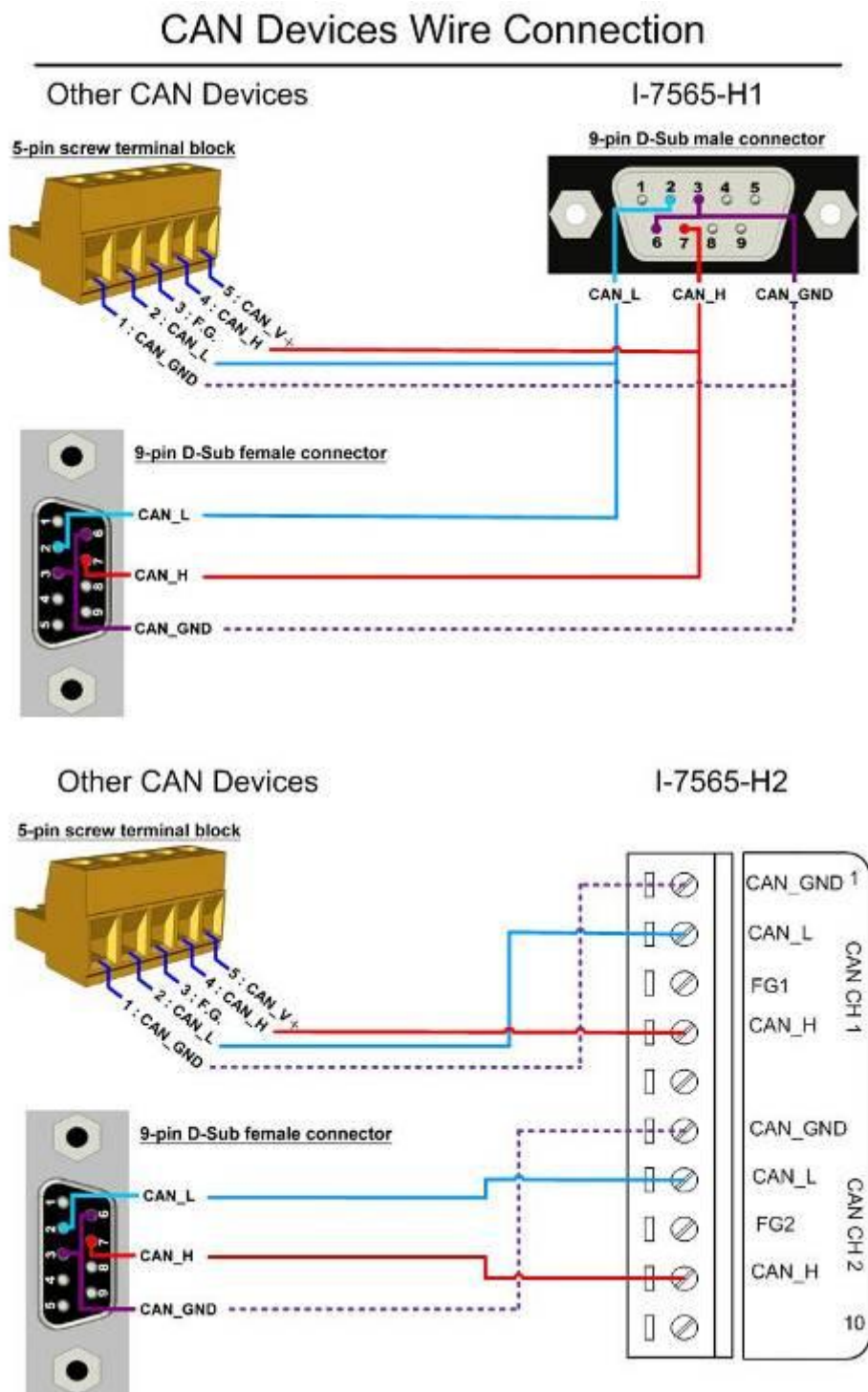


圖 2-4: CAN 硬體線路連接

## 2.4 終端電阻設定

根據 ISO 11898 的規範，使用 CAN 作為媒介傳輸時，必須在匯流排網路的終端裝上兩個終端電阻，如圖 2-6 所示：

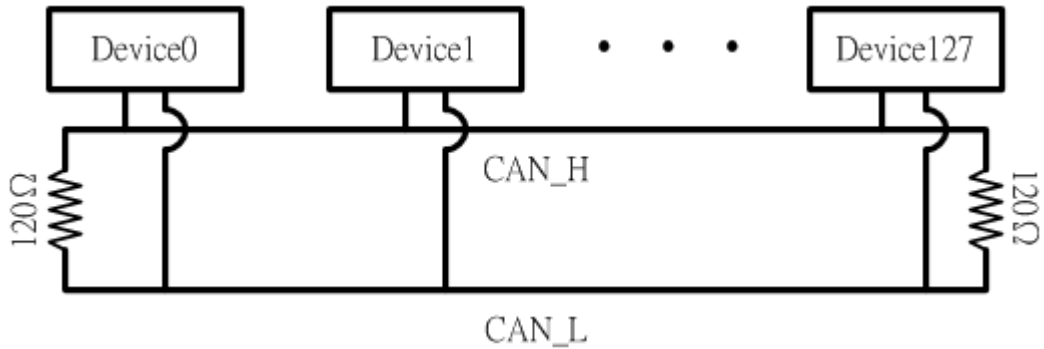


圖 2-6: 終端電阻

因此，I-7565-H1/H2 模組內提供一跳線，讓使用者選擇是否啓用終端電阻。如果使用者想使用終端電阻時，請打開 I-7565-H1/H2 的外殼，將 I-7565-H1 的 JP3 或是 I-7565-H2 的 JP3、JP4 短路，以啓用模組內部的 120Ω 終端電阻的設定(圖 2-7 所示)。(備註：終端電阻預設值為啓用)。

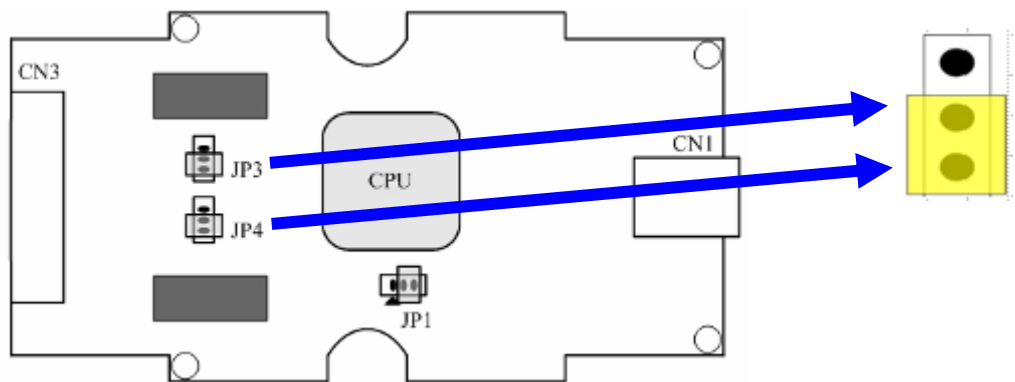


圖 2-7: 終端電阻跳線

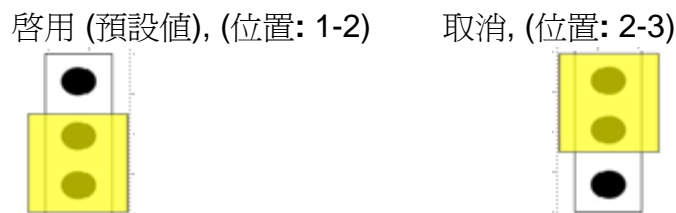


圖 2-8: JP3/JP4 跳線位置

## 2.5 Init / Normal 指撥開關

在 I-7565-H1/H2 模組的背面有一個指撥開關，它的用途是在設定模組 firmware operation 或是 firmware updating 兩種操作模式。接下來的步驟引導使用者如何設定操作模式。

### 2.5.1 Firmware Update 模式

請將指撥開關設定於“Init” (Initial)位置(圖 2-9 所示)後，並在再次重新啟動 I-7565-H1/H2 模組電源之後，它將進入“Firmware Update”模式。在這個模式中，電腦將 I-7565-H1/H2 模組視為一大量儲存體裝置，並且將自動顯示出如圖 2-10 的資料匣。

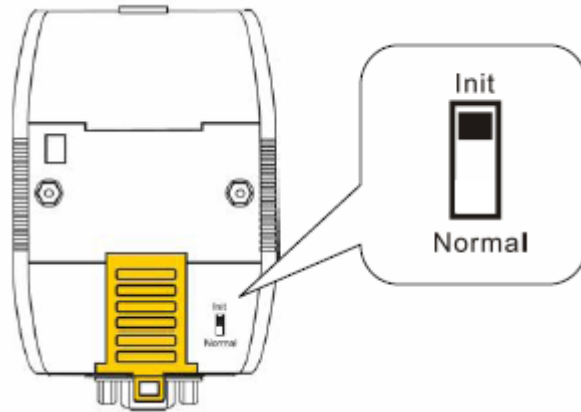


圖 2-9: 指撥開發的 Init 位置

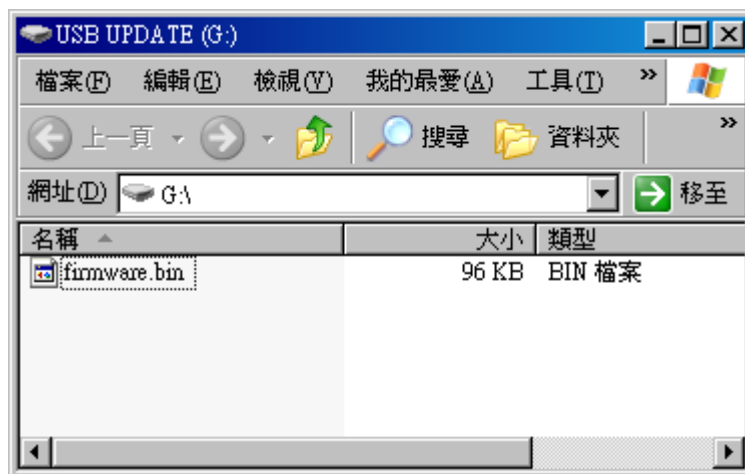


圖 2-10: USB 主儲存體裝置

使用者需要執行【Firmware\_Update\_Tool.exe】，並且依照下列各步驟，以完成韌體更新過程：

[1] 選擇【USB】介面與【USB Disk】。

- [2] 點擊【**Browser**】鍵選擇韌體檔案。(例如：**I7565H1\_v1.01.fw**)
- [3] 點擊【**Firmware Update**】鍵開發韌體更新程序。(韌體更新結果顯示在“Firmware Update”欄位內)

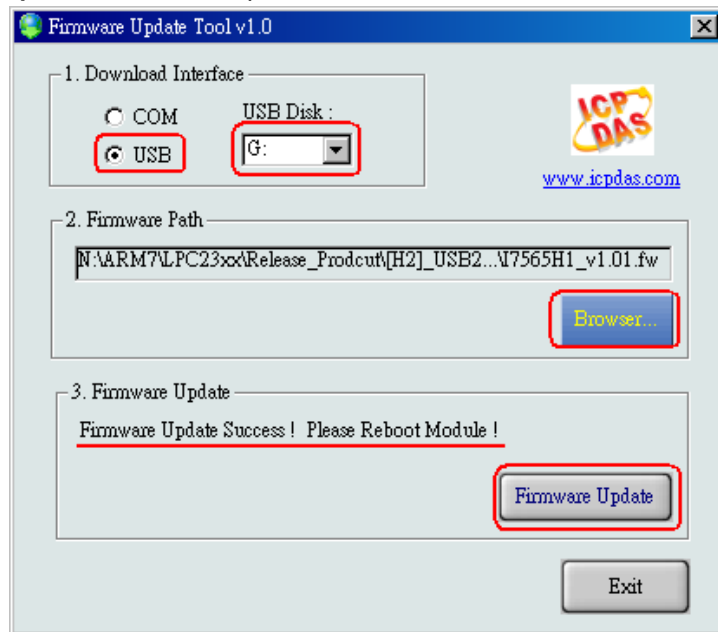


圖 2-11: Firmware\_Update\_Tool.exe 的執行畫面

Firmware\_Update\_Tool 程式可在下列網路下載：

[http://ftp.icpdas.com/pub/cd/fieldbus\\_cd/can/converter/i-7565-h1h2/software/tool](http://ftp.icpdas.com/pub/cd/fieldbus_cd/can/converter/i-7565-h1h2/software/tool)

### 2.5.2 Firmware Operation 模式

若要將模組執行運作模式時，使用者必須將指撥開關設定於“Normal”位置(圖 2-12)，並在再次重新啓動 I-7565-H1/H2 模組電源之後，它將進入“**Firmware Operation**”模式。在這個模式中，使用者可透過電腦的 USB 埠傳送/接收 CAN 訊息。

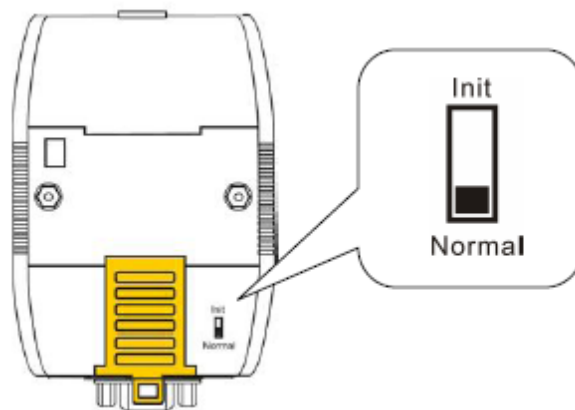


圖 2-12: 指撥開關的 Normal 位置

## 2.6 LED 指示燈

I-7565-H1/H2 提供三個指示燈讓使用者了解模組目前的運作情況。接下來，說明三個指示燈號的意義。圖 2-12 標示三個指示燈的位置。

### (1) PWR LED :

PWR LED 可以幫助使用檢查 I-7565-H1/H2 是否運作中。若模組處於“firmware operation”模式，則 PWR LED 顯示常亮。然而，當模組處於“firmware updating”模式時，該 PWR LED 將以每秒一次的頻率閃爍。

### (2) RUN LED :

RUN LED 顯示 I-7565-H1/H2 目前為傳送/接收 CAN 訊息的狀態。當傳送/接收到一筆 CAN 訊息時，RUN LED 將閃爍一次。I-7565-H2 的 CAN1 埠與 CAN2 埠共享同一 RUN LED。

### (3) ERR LED :

ERR LED 顯示目前是否有錯誤發生。ERR LED 在正常情況下是不亮的；反之，當發生 Bus-Off 錯誤發生時，ERR LED 將亮起，直到 Bus-Off 的錯誤情況被排除。若內建於 I-7565-H1/H2 CAN/USB 緩衝記憶體超載時、或是傳送 CAN 訊息失敗，則 ERR LED 將連續性的閃爍。I-7565-H2 的 CAN1 埠與 CAN2 埠共享同一 ERR LED。

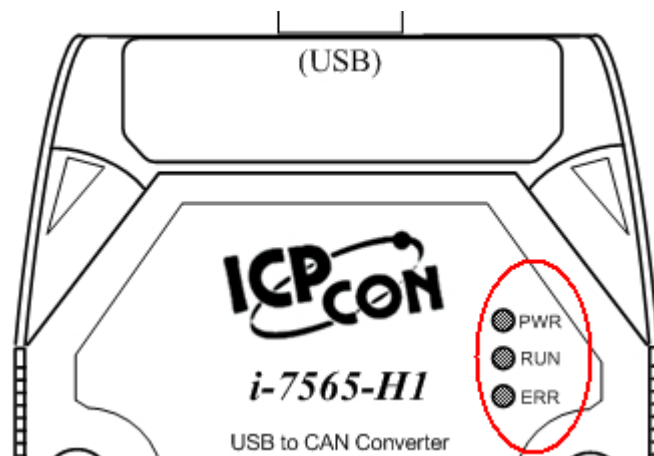


圖 2-13: I-7565-H1/H2 指示燈位置

表 2: I-7565-H1/H2 的指示燈意義

LED Name	I-7565-H1/H2 狀態	LED 狀態
所有指示燈	硬體初始化失敗	所有指示燈在重置後永遠恆亮
	硬體看門狗失效	所有指示燈以每 2 秒閃爍一次
	請聯絡我們	所有指示燈以每 300 毫秒輪流閃爍
PWR LED	Firmware Updating 模式	每秒閃爍一次
	Firmware Operation 模式	常亮
	電源關閉	常滅
RUN LED	傳輸中	閃爍一次
	Bus Idle	常滅
ERR LED	傳輸失敗	每 100 ms 閃爍一次
	緩衝區超載	每秒閃爍一次
	Bus-Off	常亮
	No Error	常滅

## 2.7 線材選擇

CAN bus 上的訊號是以二條線之間電位差取得，可運作在隔離式雙絞線、未隔離式雙絞線或是排線上。CAN 高電位線及 CAN 低電位線以並聯的方式連通整個 CAN 網路系統，而在 CAN 高電位線及 CAN 低電位線之間則設有 120 歐姆的終端電阻。至於在線路類型、線路長度、終端電阻如何決定的部分，取決於 CAN bus 網路中傳送的鮑率，請參考下表 3。



圖 2-14: 未隔離型雙絞線 (UTP)

表 3: 線材的選擇

匯流排速度	線材類型	線材 阻抗/m	終端電阻	匯流排長度
50k bit/s at 1000m	0.75~0.8mm <sup>2</sup> 18AWG	70 mOhm	150~300 Ohm	600~1000m
100k bit/s at 500m	0.5~0.6 mm <sup>2</sup> 20AWG	< 60 mOhm	150~300 Ohm	300~600m
500k bit/s at 100m	0.34~0.6mm <sup>2</sup> 22AWG, 20AWG	< 40 mOhm	127 Ohm	40~300m
1000k bit/s at 40m	0.25~0.34mm <sup>2</sup> 23AWG, 22AWG	< 40 mOhm	124 Ohm	0~40m

附註：AWG 為一種用來量測電線的標準方法。



### 3. 驅動程式安裝

這個章節將說明如何安裝 I-7565-H1/H2 USB/CAN 轉換器在 Windows 2K/XP 及 Win7 平台的驅動程式。您可以從泓格科技公司網站下載最新的驅動程式安裝檔：

([http://ftp.icpdas.com/pub/cd/fieldbus\\_cd/can/converter/i-7565-h1h2/driver](http://ftp.icpdas.com/pub/cd/fieldbus_cd/can/converter/i-7565-h1h2/driver))。接下來，我們將引導您完成驅動程式的安裝：

#### 3.1 自動安裝 I-7565-H1/H2 驅動程式:

##### [ Step - 1 ]

先將I-7565-H1 or I-7565-H2 插入電腦後，Windows作業系統將偵測到該新的裝置，並在畫面上顯示“尋找新增硬體精靈”畫面提示您安裝已偵測到的USB裝置，如圖3-1所示，請直接點擊“取消”鍵，結束手動安裝I-7565-H1/H2驅動程式過程。

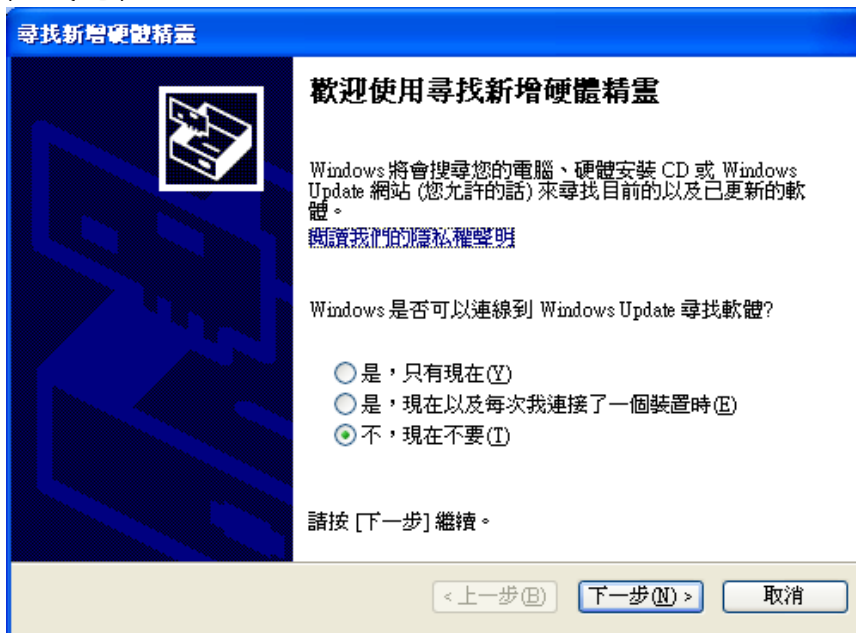


Figure 3-1: 新增硬體精靈(1)

##### [ Step - 2 ]

執行 “I7565H1H2\_DrvInst.exe” 程式，來啟動自動安裝I-7565-H1/H2驅動程式之程序，待出現如圖Figure 3-2畫面時，請點擊“繼續安裝”鍵，在自動安裝程式結束後，即會彈出如圖Figure 3-3之畫面。

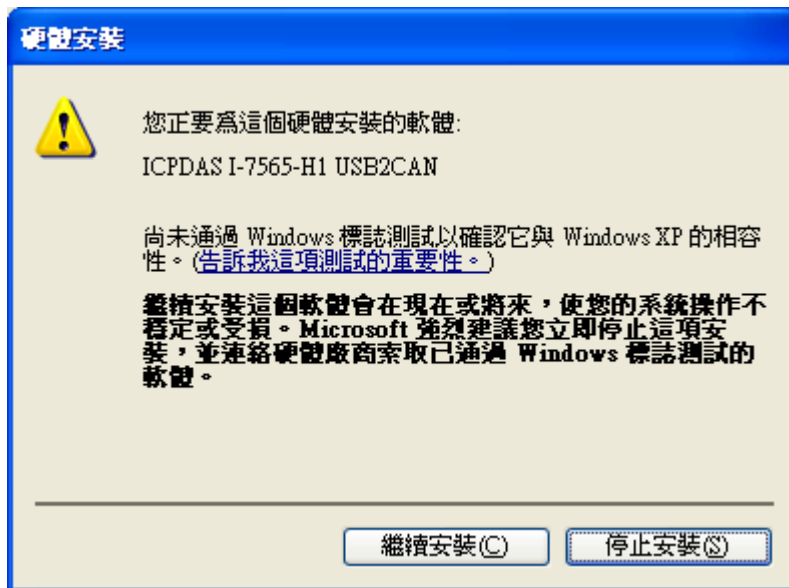


Figure 3-2: 新增硬體精靈 (2)

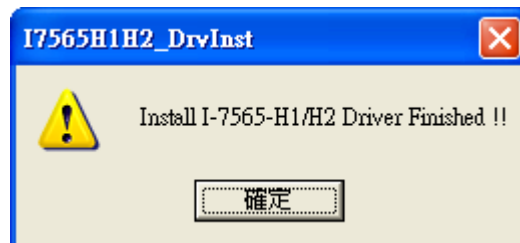


Figure 3-3: 自動安裝 I-7565-H1/H2 驅動程式完成

### 3.2 手動安裝 I-7565-H1/H2 驅動程式:

#### [ Step-1 ]

先執行“ICPUsbConverter\_DrvInst.exe”程式 (v1.2版以後之Driver名稱)，將I-7565-H1/H2之驅動程式檔案複製至系統中。

#### [ Step-2 ]

將I-7565-H1 or I-7565-H2 插入電腦後，Windows作業系統將偵測到該新的裝置，並在畫面上顯示“尋找新增硬體精靈”畫面提示您安裝已偵測到的USB裝置，如圖3-4所示，請選擇“不，現在不要(T)”選項並且點擊“Next”鍵。

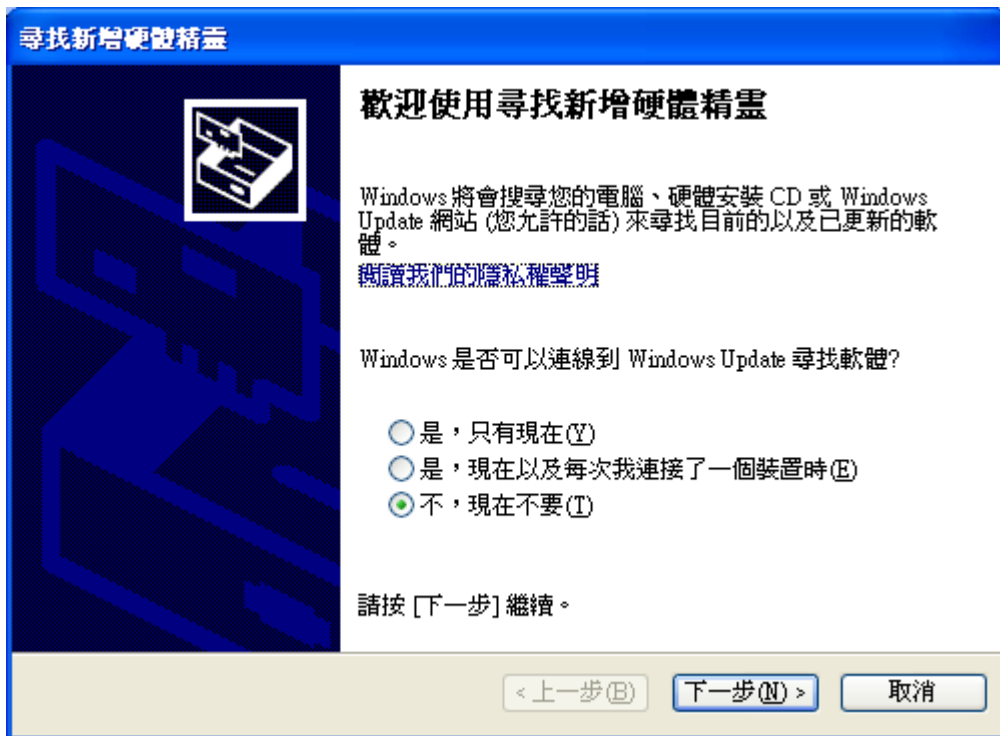


圖 3-4: 新增硬體精靈 (1)

### [ Step-3 ]

如圖3-5所示，請選擇“從清單或特定位置安裝(進階)(S)”選項並且點擊“下一步”。

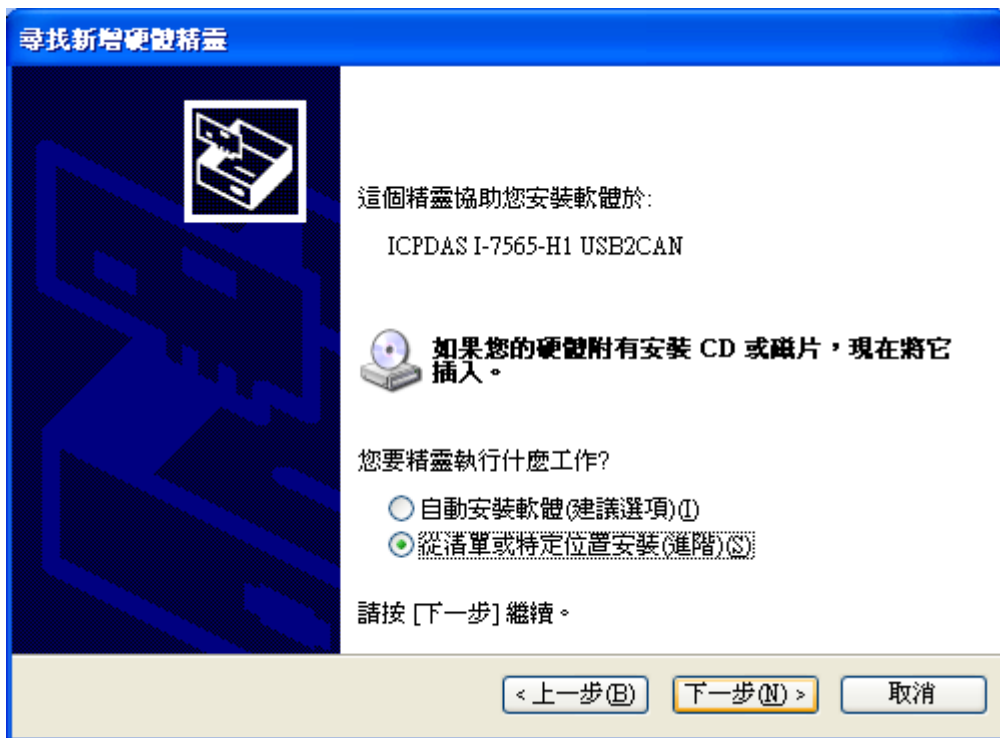


圖 3-5: 新增硬體精靈 (2)

### [ Step-4 ]

如圖3-6所示，請選擇“在這些位置中搜尋最好的驅動程式(S)”選項並且檢查“搜尋時包括這個位置(Q):”選項與點擊“瀏覽”鍵，指定I-7565-H1/H2驅動程式安裝檔案的位置 - C:\WINDOWS\inf，然後點擊“下一步”。

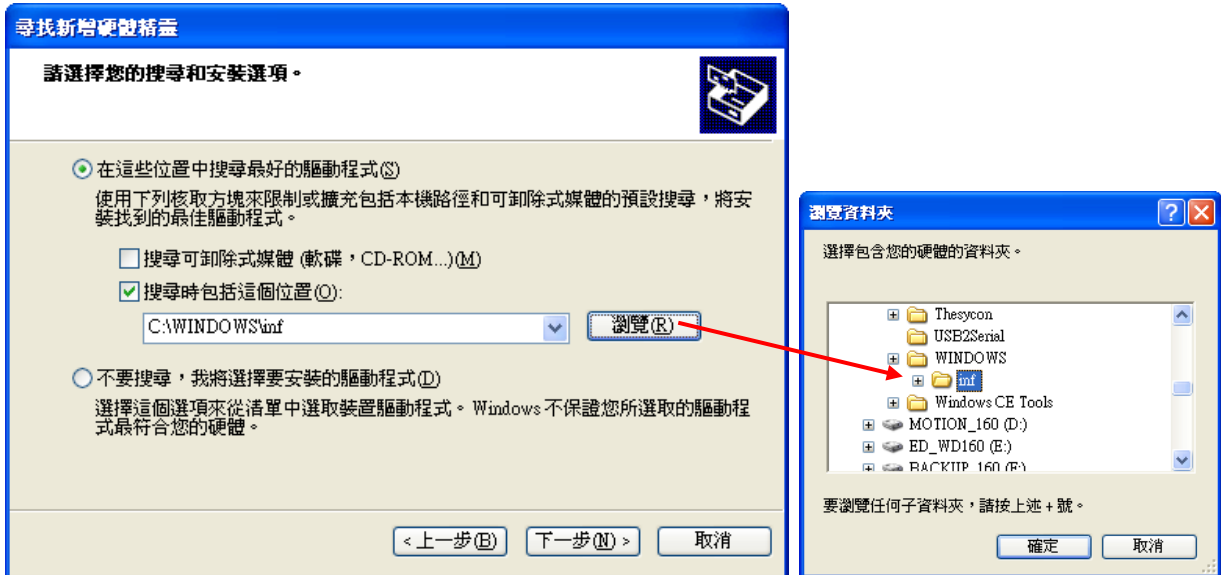


圖 3-6: 新增硬體精靈 (3)

### [ Step-5 ]

如圖3-7，請點擊“繼續安裝”鍵。

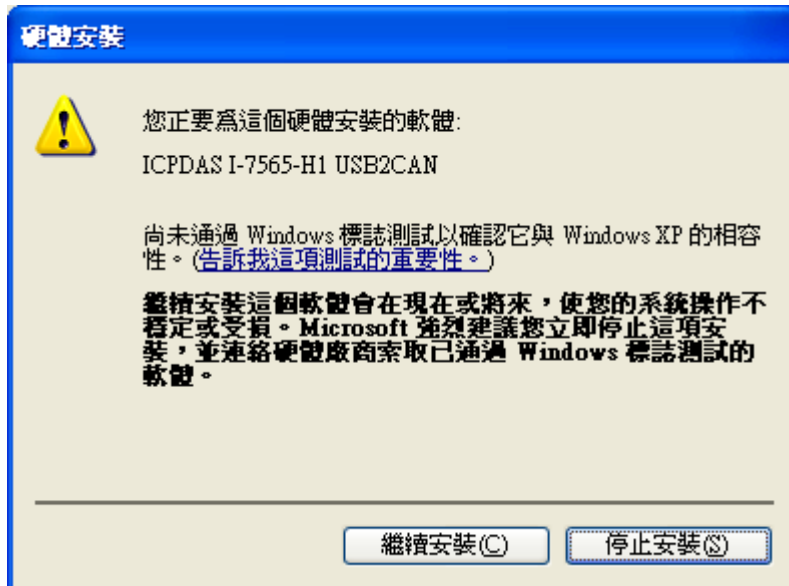


圖 3-7: 新增硬體精靈 (4)

### [ Step-6 ]

如圖3-8，請點擊“Finish”鍵，以完成I-7565-H1/H2裝置的驅動程式安裝。

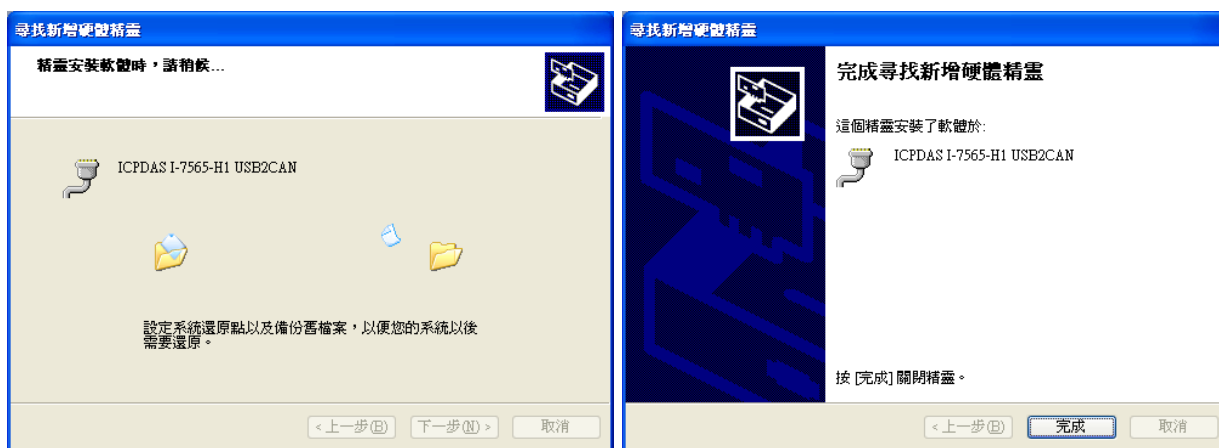


圖 3-8: 新增硬體精靈(5)

### 3.3 驗證驅動程式安裝：

這個章節說明如何驗證I-7565-H1/H2裝置驅動程式是否完全安裝。若安裝驅動程式成功，Windows系統將給予一個“虛擬COM埠”編號。請照片下述的步驟檢查。

點擊在桌面上我的【電腦滑鼠】右鍵後，選擇【內容】，並在彈出的視窗，點擊在【硬體】分頁中的【裝置管理員】。雙擊在**Ports (COM & LPT)**項目。如圖3-9，若裝置驅動程式安裝正確，使用者可在這裡找到“ICPDAS I-7565-H1 USB2CAN”或是“ICPDAS I-7565-H2 USB2CAN”裝置項目與Windows系統給予的“虛擬COM埠”編號 – **COM3**。

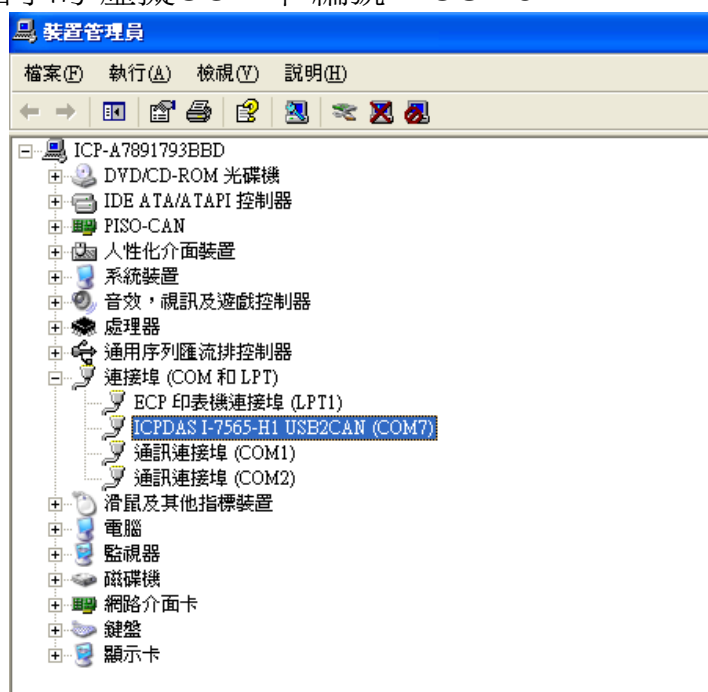


圖 3-9: 虛擬 COM 埠編號

### 3.4 移除 I-7565-H1/H2 驅動程式

請依照下述的步驟移除I-7565-H1/H2裝置驅動程式。

#### [ Step-1 ]

點擊在桌面上我的【電腦滑鼠】右鍵後，選擇【內容】，並在彈出的視窗，點擊在【硬體】分頁中的【裝置管理員】。雙擊在**Ports (COM & LPT)**項目。請找出“ICPDAS I-7565-H1 USB2CAN”或是“ICPDAS I-7565-H2 USB2CAN”裝置項目，並於該項目點擊滑鼠右鍵選擇“解除安裝”項目。

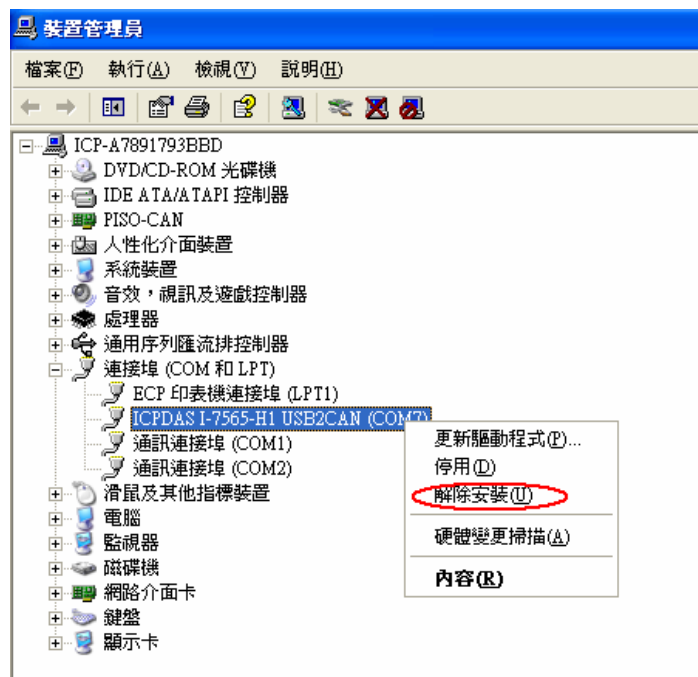


圖 3-10: 移除 I-7565-H1/H2 驅動程式(1)

#### [ Step-2 ]

如圖3-11，點擊“確定”鍵完成I-7565-H1/H2裝置驅動程式移除。在移除後，“ICPDAS I-7565-H1 USB2CAN”或是“ICPDAS I-7565-H2 USB2CAN”裝置項目將消失於**Ports (COM & LPT)**中。

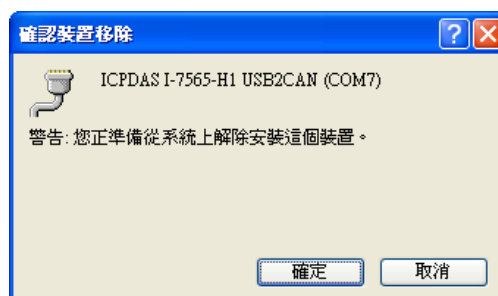


圖 3-11: 移除 I-7565-H1/H2 驅動程式 (2)

## 4. 軟體工具

我們提供的 I-7565-H1/H2 Utility 具有簡易的傳送/接收 CAN 訊息功能。在此同時，它也可以顯示每一個接收到的 CAN 訊息時戳，以利於分析。I-7565-H1/H2 Utility 可以從泓格科技公司網站上下載：

[http://ftp.icpdas.com/pub/cd/fieldbus\\_cd/can/converter/i-7565-h1h2/software/utility](http://ftp.icpdas.com/pub/cd/fieldbus_cd/can/converter/i-7565-h1h2/software/utility). 接下來，我們將說明如何使用 I-7565-H1/H2 Utility 的主要功能：

### 4.1 INI 檔案功能

每當使用者執行 I-7565-H1/H2 Utility 後，它會尋找與它相同路徑的 INI 檔案- **I-7565-H1H2 Utility.ini**，以載入初始設定值。若該 INI 檔案不存在時，Utility 會自動以預設值做為設定值。圖 4-1 說明該 INI 檔格式：

- [1] **COM** : 虛擬 COM 埠編號。(預設值：COM=1)
- [2] **TYPE** : 1: I-7565-H1；2: I-7565-H2。(預設值：TYPE=1)
- [3] **C1BR** : CAN1 鮑率。(預設值：C1BR=5K)
- [4] **C2BR** : CAN2 鮑率。(預設值：C2BR=Disable)
- [5] **C1EN** : CAN1 埠功能。(1: Enable; 0: Disable, 預設值: 1)
- [6] **C2EN** : CAN2 埠功能。(1: Enable; 0: Disable, 預設值: 0)
- [7] **C1LOM** : CAN1 埠 Listen Only 模式 (1: Enable; 0: Disable)
- [8] **C2LOM** : CAN2 埠 Listen Only 模式 (1: Enable; 0: Disable)
- [9] **SYMFILE**: 自動載入 Symbol 初始化檔 (I-7565-H1H2\_SymFile.ini)  
(1: Enable; 0: Disable) => Utility v1.10 版提供

```
COM=1 TYPE=1 C1BR=1000 C2BR=1000 C1EN=1 C1LOM=0 C2EN=1 C2LOM=0 SYMFILE=1
```

圖 4-1: I-7565-H1/H2 Utility 連線參數初始化檔

### 4.2 連線功能

當使用者執行 I-7565-H1/H2 Utility 後，它將顯示像圖 4-2-1 的 I-7565-H1/H2 連線參數畫面。我們列出各連線參數：

- [1] **Com Port** : 虛擬 COM 埠編號。
- [2] **Mod Name** : 模組名稱。
- [3] **CAN Port Enable** : 啟用 CAN 埠功能。(勾選: 啟用)  
(3.1) **Listen Only Mode** : 啟用 Listen Only 模式功能。(勾選: 啟用)
- [4] **CAN Baud Rate** : CAN 匯流排鮑率設定。

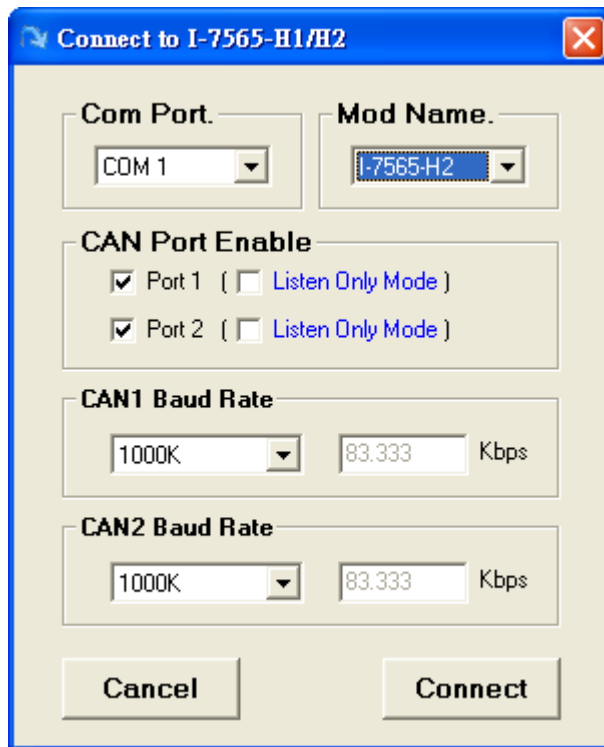


圖 4-2-1: I-7565-H1/H2 Utility 的連線參數畫面

### [ 注意 ]

#### 1. 自行定義的 CAN 鮑率設定：

若使用者想以自行定義的鮑率進行通訊時，在“I-7565-H1/H2 Utility”的”Connect to I-7565-H1/H2”畫面，使用者可選擇“**Defined**”項目並且在”Bard Rate”右邊的欄位輸入自訂的 CAN 鮑率值(例如：83.333)，如圖 Figure 4-2-2，然後，點擊“Connect”鍵以連線至 I-7565-H1/H2。

在韌體 v1.07 版，新增 CAN 鮑率之 Bit-Timing 取樣點 (即 Tseg2 值) 設定功能，Tseg2 值範圍可由 2~6，當應用在有電磁波干擾之 CAN bus 場合時 (如：馬達啓動產生干擾)，則可在相同 CAN 鮑率下，選擇 Tseg2 較大值來作通訊。



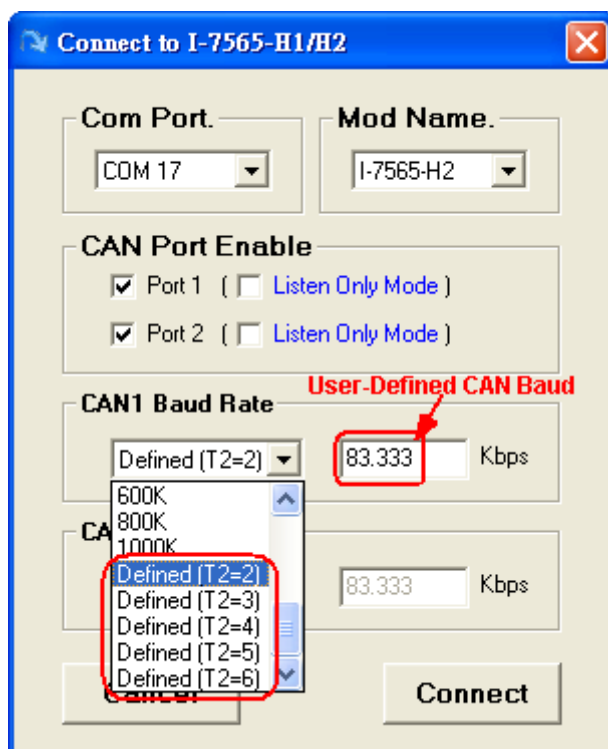


圖 4-2-2: 對 I-7565-H1/H2 自行定義 CAN 鮑率速率

## 2. “Listen Only Mode” (LOM)功能：

需搭配 I-7565-H1/H2 Utility v1.09 以及 FW v1.05 以後版本，通訊畫面如下圖所示。(在 LOM 模式下是無法傳送 CAN 訊息)

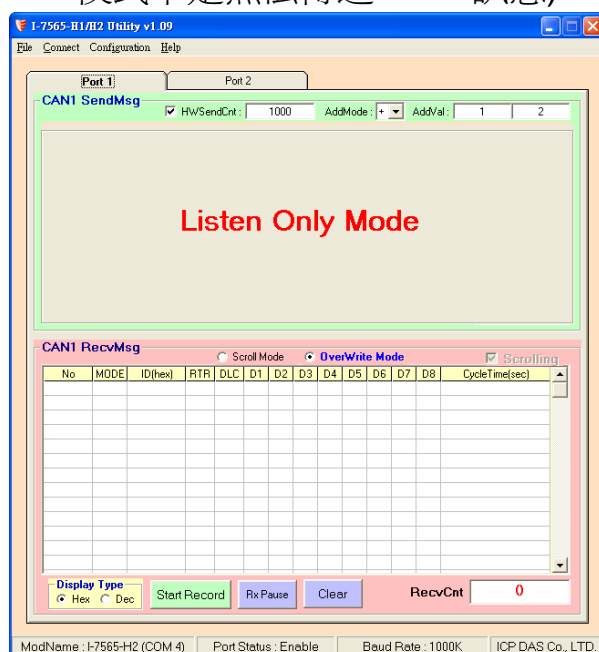


圖 4-2-3: I-7565-H1/H2 Utility 的 LOM 通訊畫面

在完成連線參數設定之後，點擊【Connect】鍵，以連線至 I-7565-H1/H2 模組。(附註：未連線之前，I-7565-H1/H2 並不會影響 CAN 通訊，

直到使用者成功連線至 I-7565-H1/H2)。使用者與 I-7565-H1/H2 斷線 (Disconnect)後，該 I-7565-H1/H2 上的 CAN 埠功能會立即失效；使用者可以也可以點擊選單中的【Connect】選項，並且選擇【Connect To I-7565-H1/H2】，以重新連線至 I-7565-H1/H2 (如圖 4-2-4 與圖 4-2-5 的“Disconnect”功能)。

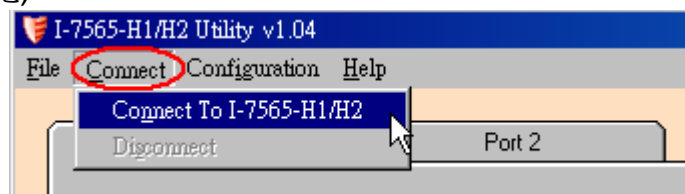


圖 4-2-4: “Connect To I-7565-H1/H2” 功能

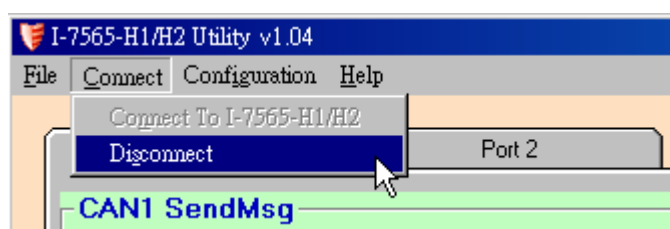


圖 4-2-5: “Disconnect” 功能

### 4.3 通訊功能

若成功連線至 I-7565-H1/H2 後，CAN bus 的通訊內容會顯示畫面上 (如圖 4-5)。

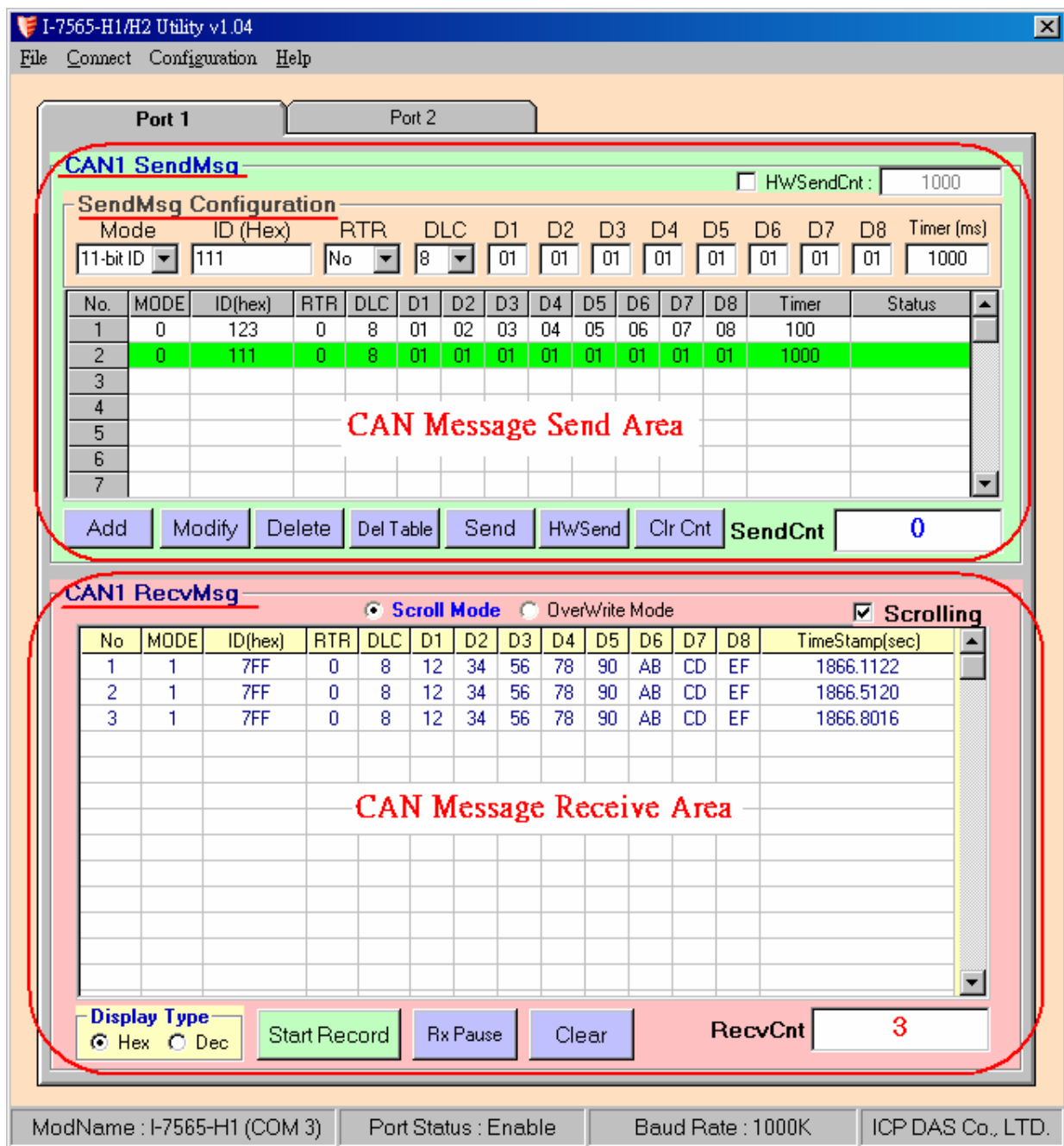


圖 4-5: I-7565-H1/H2 Utility 的通訊畫面

接下來，分別說明通訊畫面中的二個區塊：**SendMsg** 與 **RecvMsg**。此外，**“Port 1”** / **“Port 2”**分頁分別切換至 **CAN 1** / **CAN 2** 的通訊畫面。

[1] 在 **“CAN1/2 SendMsg”** 區塊：

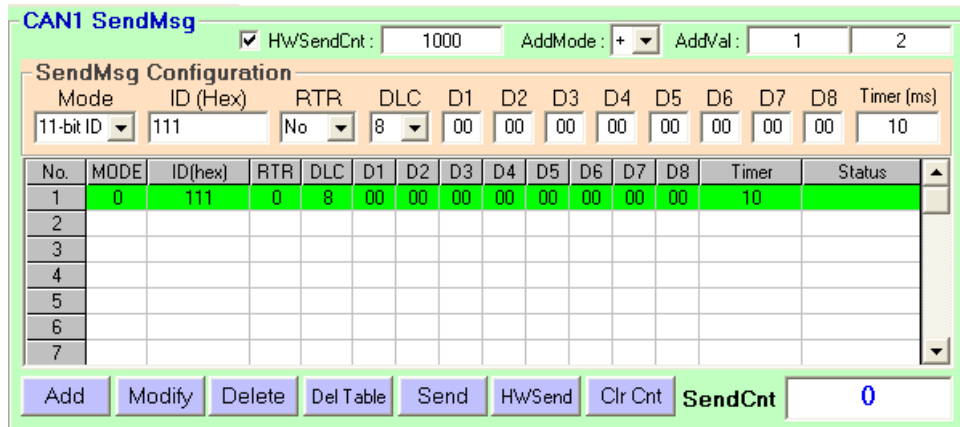


圖 4-6-1: 傳送 CAN 訊息區塊

<1> **“SendMsg Configuration”** 內容:

用來編輯 CAN 訊息的參數，使用者可以使用”Add”按鍵增加 CAN 訊息至 “CAN Message Send Area”內。

<2> **“Add”** 鍵：

它可新增一筆自行定義的 CAN 訊息至“CAN Message Send Area”的最後一列。

<3> **“Modify”** 鍵：

它可修改目前在“CAN Message Send Area”指定的 CAN 訊息內容，將“SendMsg Configuration” 位置中的 CAN 訊息參數更新至此。

<4> **“Delete”** 鍵：

它可刪除目前在 “CAN Message Send Area” 指定的 CAN 訊息。

<5> **“Del Table”** 鍵：

用來刪除目前在 “CAN Message Send Area” 內之所有的 CAN 訊息。

<6> **“Send”** 鍵：

它可傳送目前在” CAN Message Send Area”指定的 CAN 訊息。若”Timer” 欄位中的數值被設為 0 時，它只傳送一次；反之，它會依照 PC Timer 設定的延遲時間，並週期性地傳送該筆 CAN 訊息。

<7> **“HWSend”** 鍵：

它也可傳送目前在“CAN Message Send Area”指定的 CAN 訊息。如果 Timer 欄位中的數值被設為 0 時，它只傳送一次；反之，它會以模組的硬體 Timer 設定的延遲時間，傳送 CAN 訊息，提供比 PC Timer 更精準的傳送。若使用者想指定傳送的次數，在點擊【HWSend】鍵之前，先勾選如圖 4-6-1 畫面中

的【HWSendCnt】，並輸入傳送次數。

在“AddMode”欄位，是用來設定 CAN Data 傳送時之數值遞增模式，選項‘n’表示不啟動，選項‘+’表示以加法模式作遞增，選項‘x’表示以乘法模式作遞增；在“AddVal”欄位，是用來設定 CAN Data 傳送時之數值遞增大小，第一個欄位是給 CANL Data 使用，第二個欄位是給 CANH Data 使用。

<8> “Clr Cnt” 鍵：

它將重置在“CAN Message Send Area”內的【SendCnt】傳送計數值為零。

<9> “SendCnt” 欄位：

當傳送一筆 CAN 訊息時，【SendCnt】計數值將增加 1。

[2] 在“CAN1/2 RecvMsg” 區塊中：

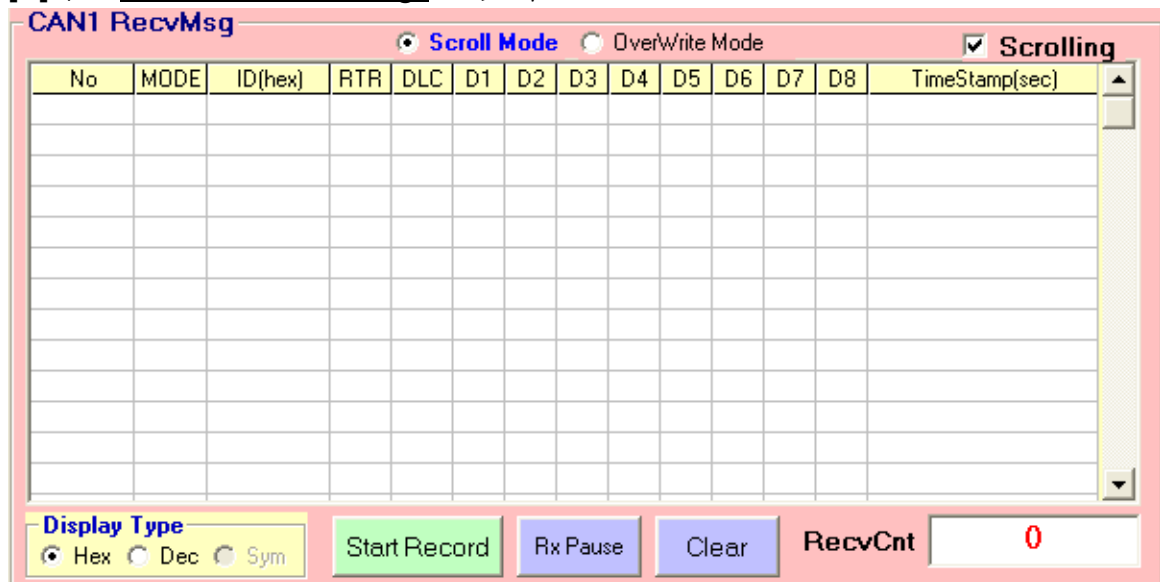


Figure 4-6-2: 接收 CAN 訊息區塊

<1> “Display Type” 選項：

Hex 選項：在“CAN Message Receive Area”中，以 16 進制格式顯示接到的 ID 及 Data 資料。

Dec 選項：在“CAN Message Receive Area”中，以 10 進制格式顯示接到的 ID 及 Data 資料。

Sym 選項：在“CAN Message Receive Area”中，以 Symbol 文字顯示接到的 ID 資料。(只有在 OverWrite 模式下才有支援且需先載入過 Symbol 設定檔，Utility\_v1.10 版提供)

以下為 Symbol 設定檔(\*.ini)之檔案內容格式範例：

**[CAN1Sym]**

SymNum=2  
ID1=0x100  
Name1=Engine Speed  
ID2=0x101  
Name2=Engine Temp.

**[CAN2Sym]**

SymNum=1  
ID1=0x200  
Name1=Motor Speed

**[CAN1Sym]** : 表示以下設定為 CAN1 之 Symbol 文字功能部份

**SymNum** : Symbol 文字之設定總數量

**ID1** : 第一組 CANID 之 Hex 值

**Name1** : 第一組 CANID 對應 Symbol 文字內容 (支援中文顯示)

⇒ 上述 Symbol 設定檔載入完成後，在 OverWrite 模式下，選擇”Sym”選項，在 CAN1 接收欄位中若收到 CANID=0x100 之 CANMsg，則在 ID 欄位中會改為顯示”Engine Speed”文字內容取代原來的 0x100，如圖 4-6-3 及圖 4-6-4 所示。

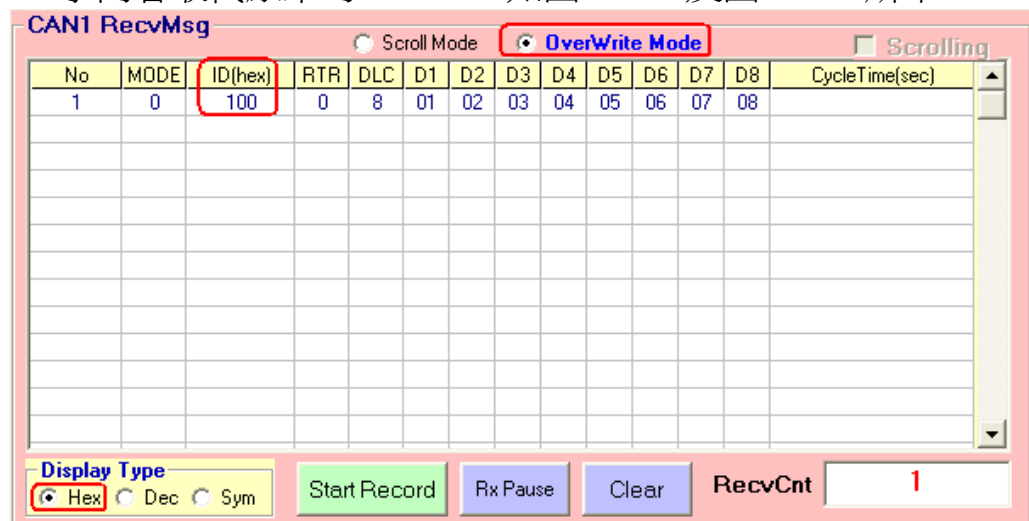


圖 4-6-3: OverWrite 模式之 Hex 顯示功能

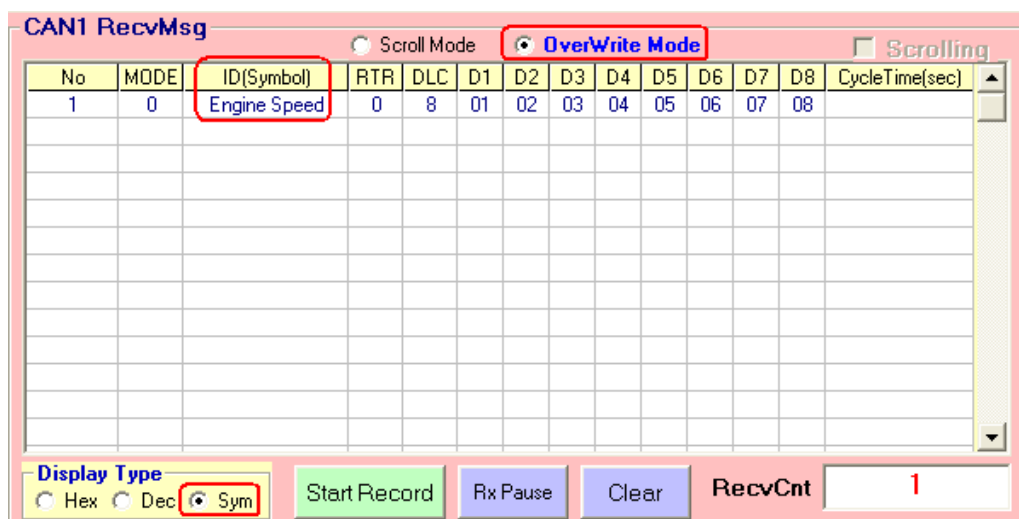


圖 4-6-4: OverWrite 模式之 Sym 顯示功能

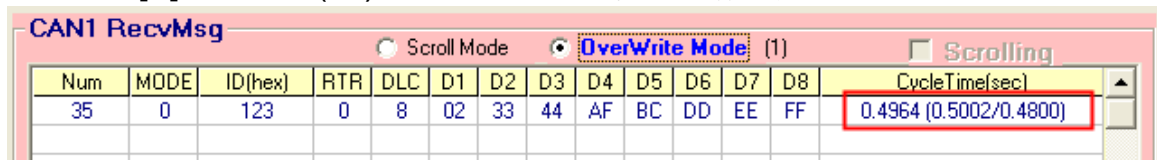
- <2> **“Start Record / Stop Record” 鍵：**  
 點擊了“Start Record”鍵之後，所有接收到的 CAN 訊息將以 ASCII 文字儲存於檔案內；反之，當點擊了“Stop Record”鍵後，它將停止記錄接收到的 CAN 訊息於檔案內。  
 檔案名稱會以“CAN1\_YYYYMMDD\_HHMMSS.txt”或是“CAN2\_YYYYMMDD\_HHMMSS.txt”(YYYY- 西元年份，MM-月，DD-日，HH-時，MM-分，SS-秒)的格式儲存於 Utility 執行檔的路徑下。
- <3> **“Rx Pause / Rx Start” 鍵：**  
 點擊了“Rx Pause”鍵後，將停止接收 CAN 訊息；反之，點擊“Rx Start”鍵，將開始接收 CAN 訊息。
- <4> **“Clear” 鍵：**  
 它將清除在“CAN Message Receive Area”的 CAN 訊息資料與重置“RecvCnt”接收計數值為零。
- <5> **“Scrolling” 選項：**  
 若“Scrolling”選項被勾選時，接收到的 CAN 訊息才會顯示在“CAN Message Receive Area”內，但是“RecvCnt”計數值還是會自動地更新。反之，若取消勾選後，“CAN Message Receive Area”將不會更新資料。
- <6> **“Scroll / OverWrite Mode” 選項：** (Utility v1.09 版本以後支援)  
**“Scroll Mode” 選項：**  
 接收到的 CAN 訊息會依序往下排列，顯示在 RecvTable 中。  
**“OverWrite Mode” 選項：**  
 接收到的 CAN 訊息若 MODE 及 ID 數值均相同時，則會被放在同一列之欄位中，其中“No”欄位會顯示相同 MODE 及 ID 之 CAN message 數量，“CycleTime”欄位會顯示此 CAN

message 之接收週期 (單位:秒)及 CAN 訊息之最長/最短間隔時間。下圖之 CycleTime 欄位說明:

[1] 0.4964 (秒) => CAN 訊息週期 (約 500ms)

[2] 0.5002 (秒) => CAN 訊息最長間隔時間

[3] 0.4800 (秒) => CAN 訊息最短間隔時間



The screenshot shows a window titled "CAN1 RecvMsg" with a table of received messages. The "OverWrite Mode" radio button is selected. The "CycleTime(sec)" column for the first message (Num 35) is highlighted with a red box and contains the value "0.4964 (0.5002/0.4800)".

Num	MODE	ID(hex)	RTR	DLC	D1	D2	D3	D4	D5	D6	D7	D8	CycleTime(sec)
35	0	123	0	8	02	33	44	AF	BC	DD	EE	FF	0.4964 (0.5002/0.4800)

圖 4-6-5: OverWrite 模式

## 4.4 設定功能

I-7565-H1/H2 Utility 提供兩種的配置功能：一是“Module Config”，另一是“Advanced Config”。使用者可以點擊列表中的“Configuration”項目，並如圖 4-7 選擇其中一項功能。

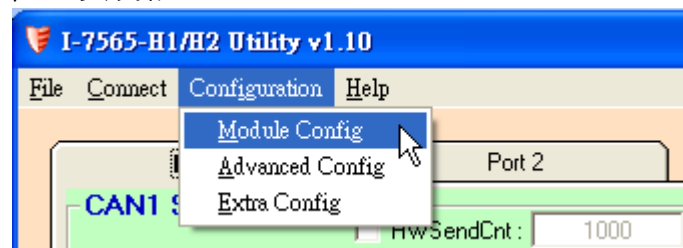


圖 4-7: I-7565-H1/H2 Utility 的設定功能

### 4.4.1 模組設定功能

下圖是“Module Config”的畫面，分為兩個區塊：一是“CAN Filter Setting”區塊，另一是“Config / Info Option”區塊(圖 4-8)。



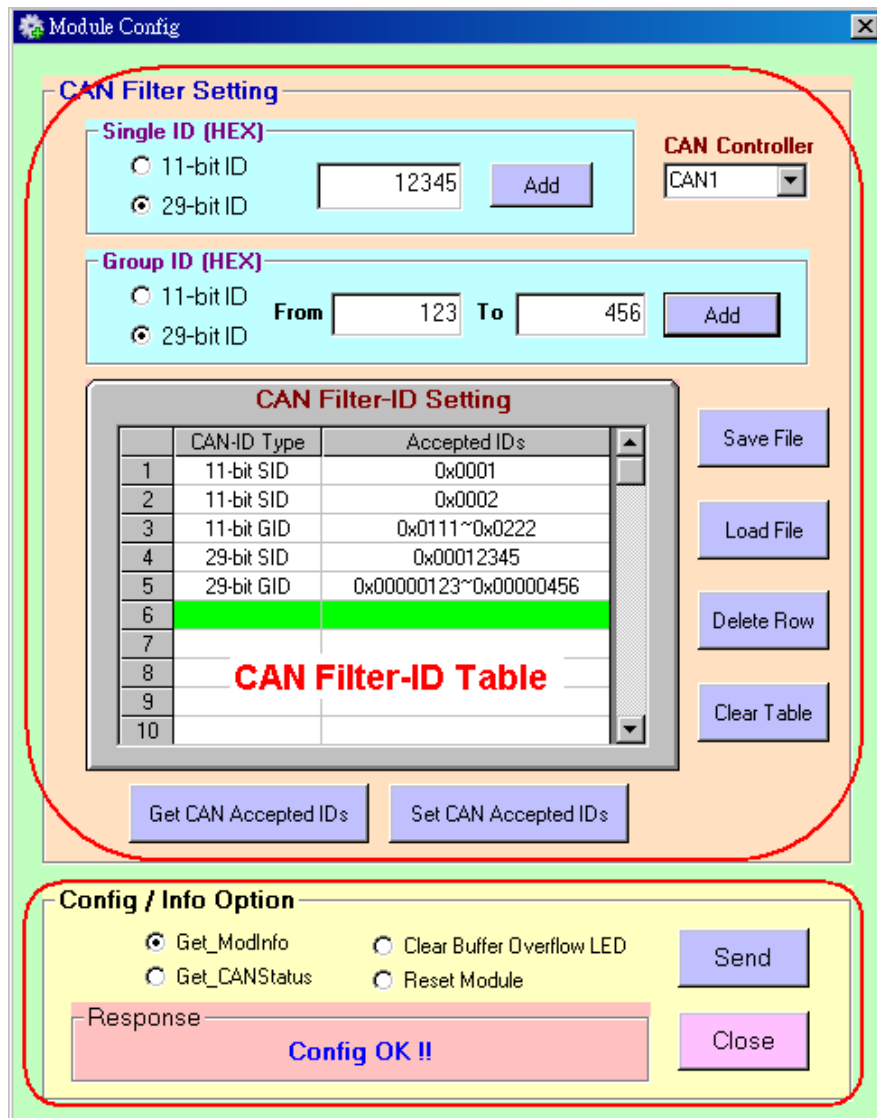


圖 4-8: I-7565-H1/H2 Utility 的模組設定功能

[1] “CAN Filter Setting” 區塊：

使用者若無設定任何的過濾規則時，所有的 CAN 訊息預設是允許接收。在“CAN Filter Setting”區塊中，使用者可以設定 I-7565-H1/H2 模組欲接收的 CAN ID。

<1> “Single ID” 項：

藉由點擊 “Add” 鍵增加指定的 CAN ID 至“CAN Filter-ID Table” 內，在“CAN Filter-ID Table” 中的 CAN ID 才會接收，其餘的會被過濾丟棄。

<2> “Group ID” 項：

藉由點擊 “Add” 鍵增加指定的 CAN ID 群組至“CAN Filter-ID Table” 內，在“CAN Filter-ID Table” 中的 CAN ID 才會接收，其餘的會被過濾丟棄。

<3> “CAN Controller” 下拉式選單：

選擇那個 CAN 埠需要進行設定。

<4> **“Get CAN Accepted IDs”** 鍵：

取得模組的 CAN 埠目前所設定的過濾 ID 資料，並顯示在“CAN Filter-ID Table”內。該命令結果也可以顯示在 “Config / Info Option” 區塊的“Response” 內。

<5> **“Set CAN Accepted IDs”** 鍵：

設定“CAN Filter-ID Table”中的 CAN ID 至指定的 CAN 埠上。該命令結果也可以顯示在 “Config / Info Option” 區塊的“Response” 內。

<6> **“Save File”** 鍵：

儲存 “CAN Filter-ID Table”中的內容至檔案。

<7> **“Load File”** 鍵：

載入檔案內容至“CAN Filter-ID Table”。

<8> **“Delete Row”** 鍵：

刪除 “CAN Filter-ID Table”列表中的項目。

<9> **“Clear Table”** 鍵：

清除 “CAN Filter-ID Table”列表中的所有項目。

[2] **“Config / Info Option”** 區塊：

Utility 針對 I-7565-H1/H2 模組提供幾個額外的功能，接下來，我們說明這些功能：

<1> **“Get\_ModInfo”** 選項：

取得模組資訊-“Module Name”(模組名稱)，“Firmware Version”(韌體版本)，以及“Hardware SN”(硬體唯一序號)。

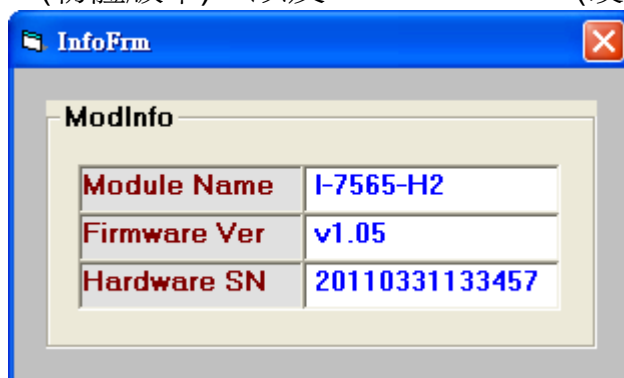


圖 4-9: 模組資訊

**[ 注意 ]**

1. “Hardware Serial Number” 功能需搭配 I-7565-H1/H2 v1.08 以及 firmware v1.04 以後版本。

<2> “Get\_CANStatus” 選項：  
取得指定的 CAN 埠的各項參數資訊。

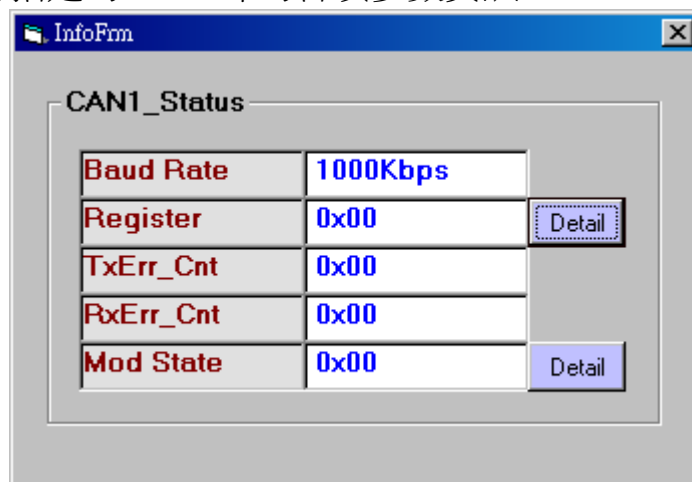


圖 4-10: CAN Status

在“Register”項目中，點擊旁邊的“Detail”鍵，它將顯示更多有關於 CAN 埠上的各暫存器資訊，若某項暫存器資訊內容顯示為 1 時，代表模組已偵測出該錯誤狀態的發生。

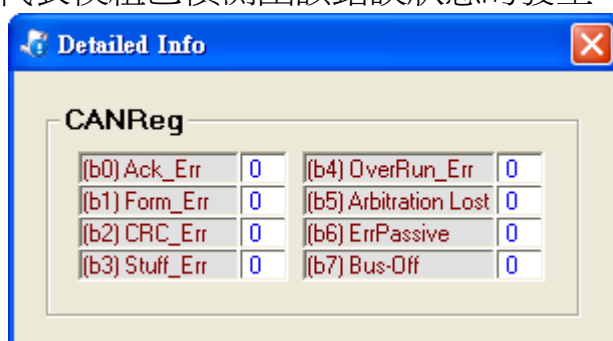


圖 4-11: CAN 埠之各暫存器資訊

點擊在“Mod State”項目旁的“Detail”鍵後，將顯示更多有關於模組狀態(如圖 4-12 所示)。若某項狀態內容顯示為 1 時，代表模組已偵測出狀態的改變。

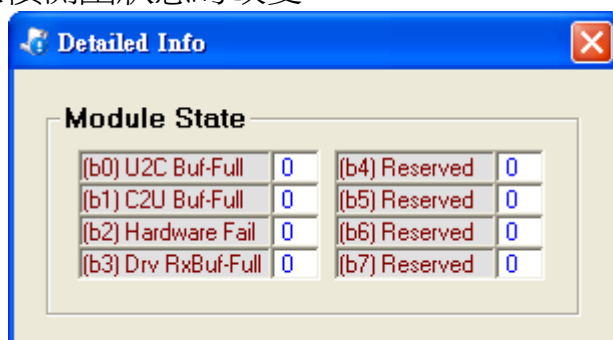


圖 4-12: 細部模組狀態資訊

- (1) U2C Buf-Full :  
模組 USB to CAN 硬體緩衝區溢出，發生過掉資料情形。
- (2) C2U Buf-Full :  
模組 CAN to USB 硬體緩衝區溢出，發生過掉資料情形。
- (3) Hardware Fail :  
模組某些硬體(如: CAN port...)初始化發生故障。
- (4) Drv RxBuf-Full :  
I-7565-H1/H2 Utility 軟體緩衝區溢出，發生過掉資料情形。

<3> **“Clear Buffer Overflow LED”** 選項：

當 CAN 與 USB 發生緩衝區溢位時，ERR LED 將以每秒一次的頻率閃爍，而這個按鍵則可用來清除 ERR LED 閃爍的狀態。

<4> **“Reset Module”** 選項：

遠端重置 I-7565-H1/H2 模組。

#### 4.4.2 進階設定功能

“Advanced Config”畫面如圖 4-13、圖 4-14 展示。

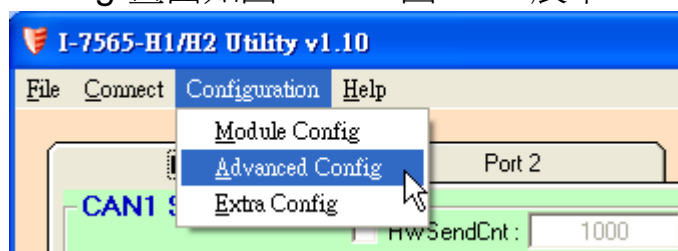


圖 4-13: I-7565-H1/H2 Utility 的進階設定功能

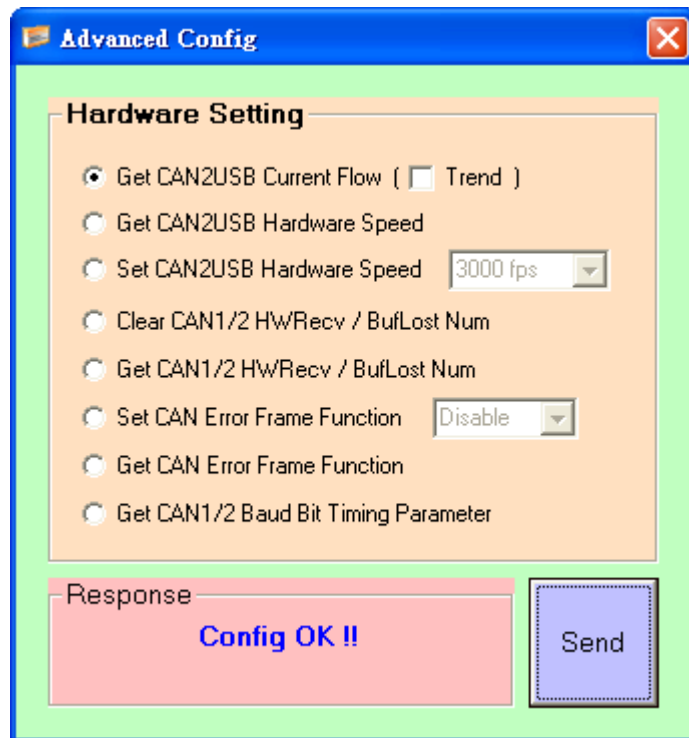


圖 4-14: I-7565-H1/H2 的進階設定

<1> “Get CAN2USB Current Flow” 選項：

取得目前在 I-7565-H1/H2 模組 CAN 埠上訊息的流量(單位: fps)。

假如“Trend”選項有被勾選，則會開啓 CAN bus 流量曲線圖畫面，如圖 4-14-1 所示。此功能畫面在 Utility v1.09 版本以後支援。

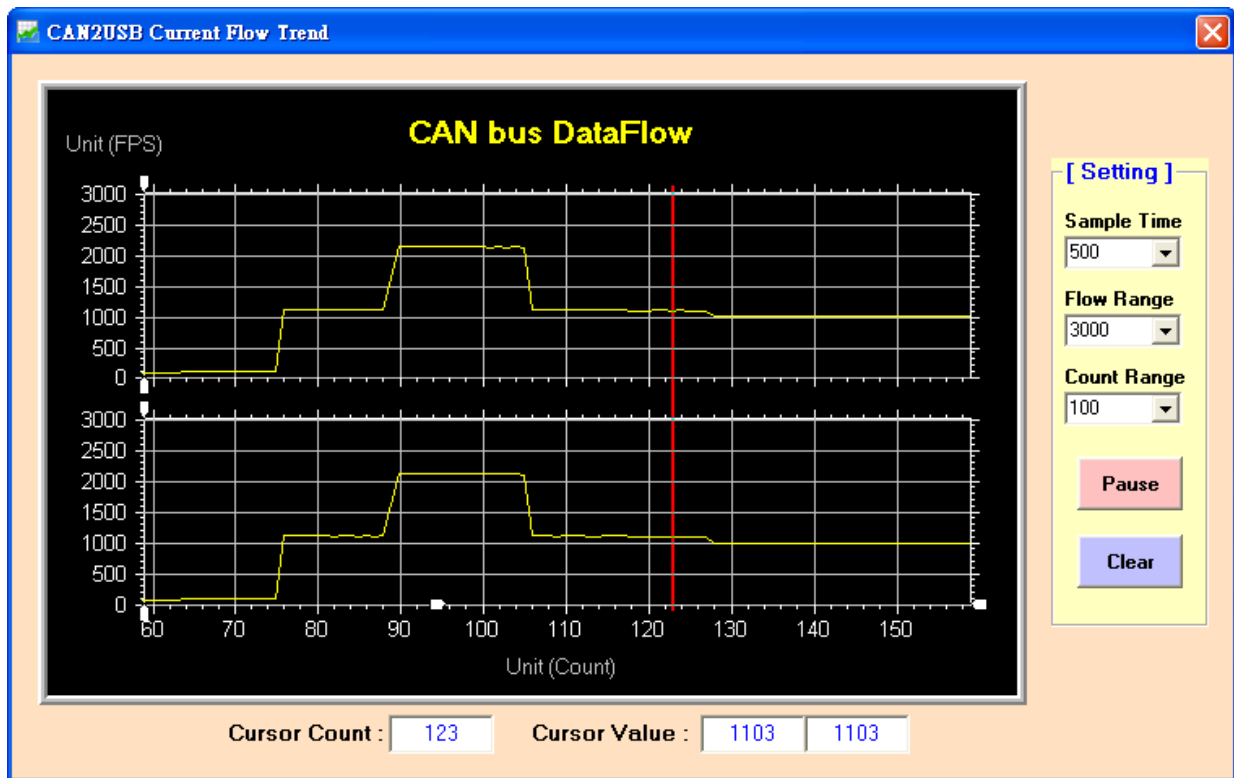


圖 4-14-1: CAN bus 流量曲線圖

<2> **“Get CAN2USB Hardware Speed”** 選項：

取得目前在 I-7565-H1/H2 模組 CAN 傳送至 USB 硬體傳輸速率的設定值。

<3> **“Set CAN2USB Hardware Speed”** 選項：

設定 I-7565-H1/H2 模組 CAN 傳送至 USB 硬體傳輸速率的設定值，其設定範圍介於 1000 fps 至 3000 fps，其設定的方法是使用者可使用“Get CAN2USB Current Flow”功能初步得知目前 CAN 訊息流量，再利用“Set CAN2USB Hardware Speed”功能設定比該 CAN 訊息流量大一點的速率。利用這個方法可在效能不佳的電腦上，降低 CAN 訊息遺失率。

<4> **“Clear CAN1/2 HWRecv / BufLost Num”** 選項：

清除模組硬體中之 CAN1 及 CAN2 所接收到及內部 Buffer 遺失掉之 CAN message 總數量。

<5> **“Get CAN1/2 HWRecv / BufLost Num”** 選項：

取得模組硬體中之 CAN1 及 CAN2 所接收到及內部 Buffer 遺失掉之 CAN message 總數量。

<6> **“Set CAN Error Frame Function”** 選項：

設定 I-7565-H1/H2 模組是否致能 CAN Error Frame 顯示功能，若此功能致能，則當 CAN bus 上有發生任何 Error 時，則在 CAN RecvMsg 接收區塊內會顯示 CAN Error Frame，如圖

4-14-2，其格式固定如下。

- (1) Mode=1 (29-bit)
- (2) ID=0xEEEEEEEE
- (3) RTR=0
- (4) DLC=8
- (5) D8=0xE1 (For CAN1), D8=0xE2 (For CAN2)
- (6) D0~D7=> CAN Error Frame 錯誤資訊

No	MODE	ID(hex)	RTR	DLC	D1	D2	D3	D4	D5	D6	D7	D8	TimeStamp(sec)
37	1	EEEEEEEE	0	8	80	00	03	00	00	E8	00	E2	9406.1949
38	1	EEEEEEEE	0	8	80	00	03	00	00	F0	00	E2	9406.1949
39	1	EEEEEEEE	0	8	80	00	03	00	00	F8	00	E2	9406.1950
40	1	EEEEEEEE	0	8	84	00	03	00	00	7F	01	E2	9406.1950
41	1	EEEEEEEE	0	8	80	00	0A	00	00	08	00	E2	9406.2728
42	1	EEEEEEEE	0	8	80	00	0A	00	00	18	00	E2	9406.2729
43	1	EEEEEEEE	0	8	80	00	11	00	00	48	00	E2	9406.2729
44	1	EEEEEEEE	0	8	84	00	11	00	00	68	00	E2	9406.2729
45	1	EEEEEEEE	0	8	A0	00	11	00	00	88	00	E2	9406.2729
46	1	EEEEEEEE	0	8	80	00	06	00	00	90	00	E2	9406.2729
47	1	EEEEEEEE	0	8	80	00	0A	00	00	98	00	E2	9406.2730
48	1	EEEEEEEE	0	8	80	00	03	00	00	A0	00	E2	9406.2731
49	1	EEEEEEEE	0	8	80	00	03	00	00	A8	00	E2	9406.2731

Display Type:  Hex  Dec  Sym

Start Record Rx Pause Clear RecvCnt: 49

圖 4-14-2: CAN Error Frame 顯示

當點選所要查看之 CAN Error Frame 行數 (如 No=38)，即會彈出此 CAN Error Frame 之詳細錯誤資訊訊窗，如圖 4-14-3，包含仲裁 Error 及 Bus Error 之相關錯誤訊息。

CAN Message No : 38

Previous Next

[ Kind ]	[ Status ]	[ Dir ]	[ Type ]	[ Frame Bit ]
Arbi Lost	OFF	X	X	X
Bus Error	ON	Send	Bit	Start-Of-Frame

Tx Error Count : 240 Rx Error Count : 0 Bus-Off : OFF

圖 4-14-3: CAN Error Frame 詳細資訊

<7> “Get CAN Error Frame Function” 選項：

取得 I-7565-H1/H2 模組是否致能 CAN Error Frame 顯示功能。

<8> “Get CAN1/2 Baud Bit Timing Parameter” 選項：

取得 I-7565-H1/H2 模組之 CAN1/2 的 CAN Baud Bit-Timing 設定參數，如圖 4-14-4，當 I-7565-H1/H2 與其它 CAN bus 設備使用相同 CAN baud 但還是無法正常通訊時，此功能可方便用來檢查 CAN 設備彼此間 Baudrate Bit-Timing 之 sample point 設定是否相同。

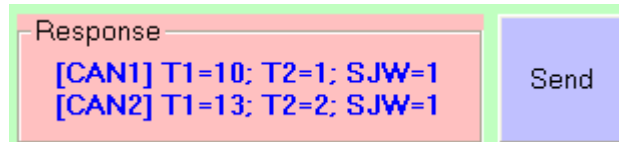


圖 4-14-4: CAN Baud Bit-Timing 設定參數

#### 4.4.3 額外設定功能

“Extra Config”畫面如圖 4-15、圖 4-16 展示。

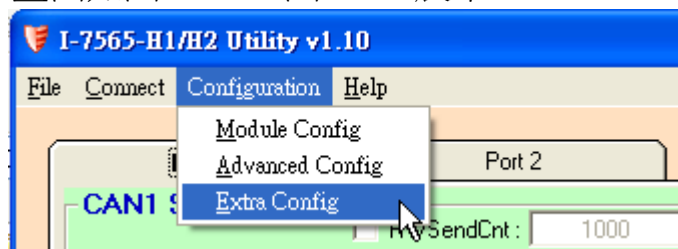


圖 4-15: I-7565-H1/H2 Utility 的額外設定功能

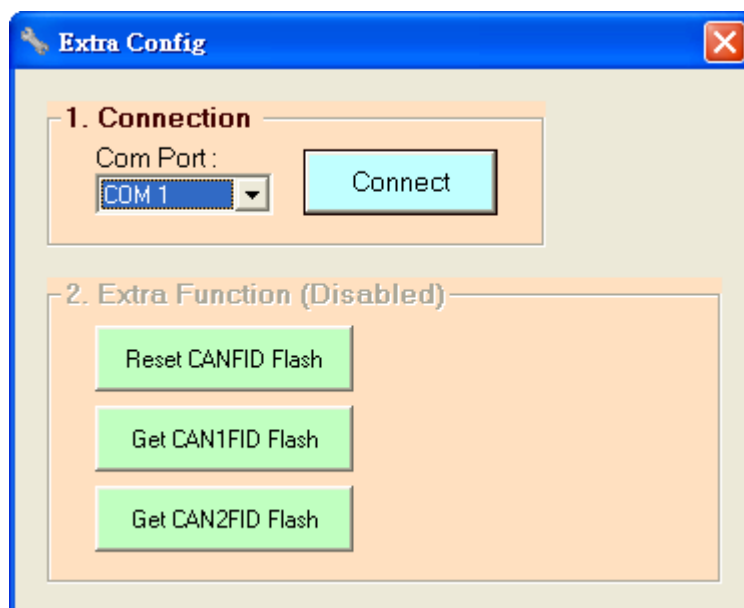


圖 4-16: I-7565-H1/H2 Utility 的額外設定功能畫面



請依照以下步驟來操作：

- (1) 選擇所要連線之 ComPort 號碼，再按下”Connect”鈕。
- (2) 連線成功後，即會致能”Extra Function”區之所有按鈕功能，按鈕功能說明如下：

**[1] Reset CANFID Flash 鈕：(Debug 用)**

=> 用來直接清除 CAN1/2 之 Filter-ID 所儲存之 Flash 內容。

**[2] Get CAN1FID Flash 鈕：(Debug 用)**

=> 用來顯示 CAN1 之 Filter-ID 所儲存之 Flash 內容。

**[3] Get CAN2FID Flash 鈕：(Debug 用)**

=> 用來顯示 CAN2 之 Filter-ID 所儲存之 Flash 內容。

## 4.5 資料記錄功能

在點選在選單中的「File」後，使用者可選擇執行資料記錄功能，如圖 4-15 所示。

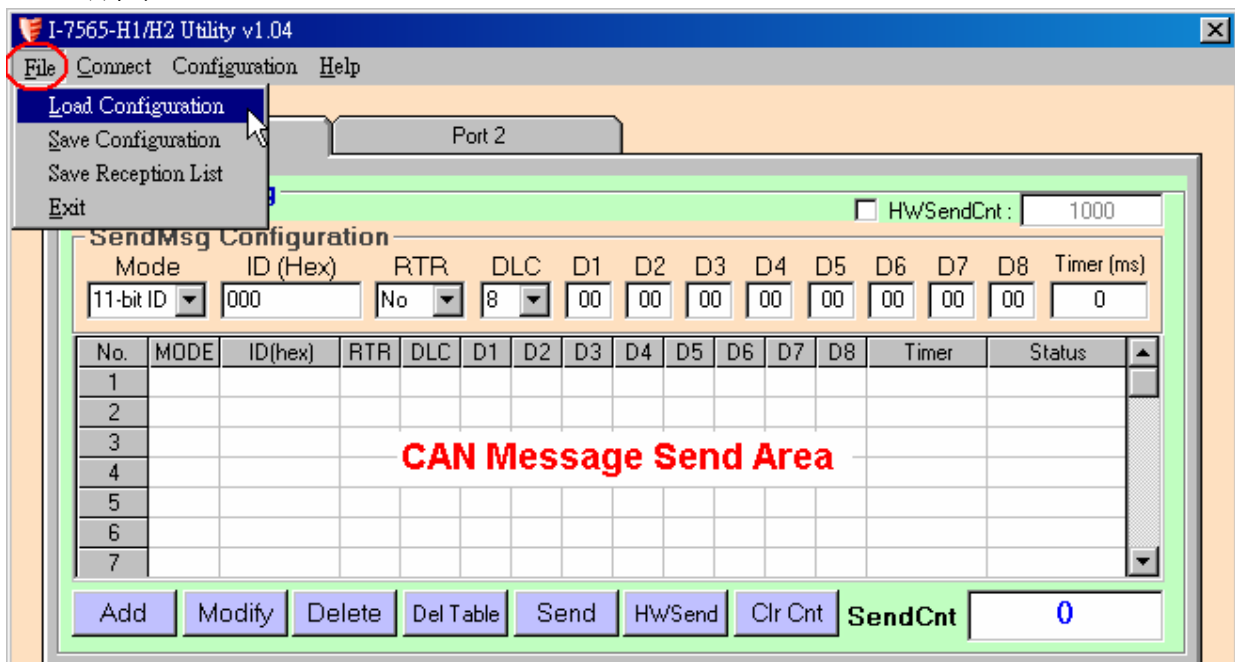


圖 4-15: I-7565-H1/H2 進階設定

### <1> “Load Configuration” 功能：

載入先前由 “CAN Send Message Configuration” 所記錄的文件檔至 “CAN Message Send Area” 內。(圖 4-16)



圖 4-16: 載入設定檔

<2> “Save Configuration” function :

儲存目前在“CAN Message Send Area”內的“CAN Send Message Configuration”設定值至指定的文字檔內。(圖 4-17)

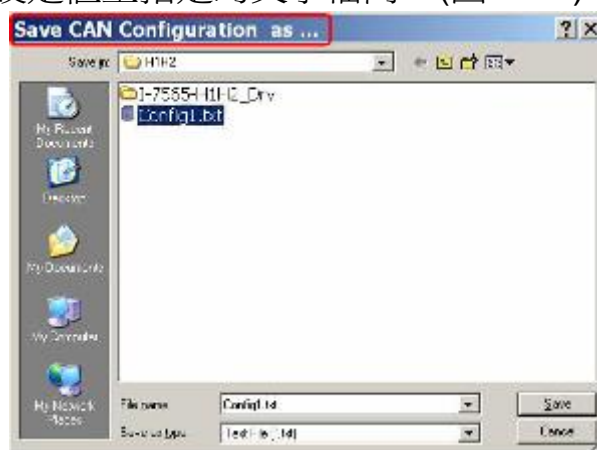


圖 4-17: 儲存設定檔

<3> “Load Reception List” 功能 :

載入先前由“CAN Message Receive Area”所記錄的文件檔至“CAN Message Receive Area”內。(圖 4-17-1)

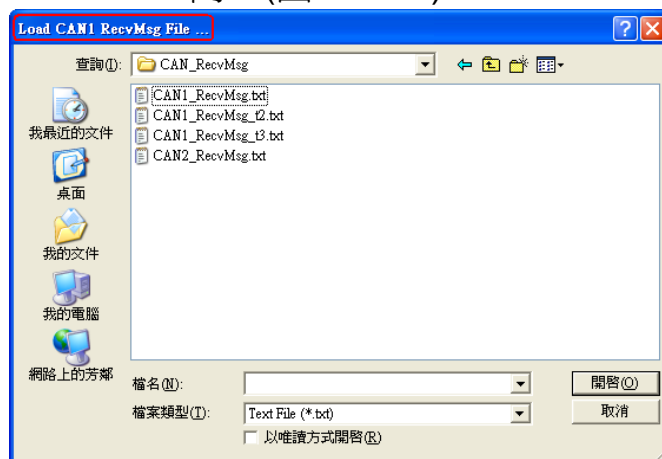


圖 4-17-1: Load Reception List 功能

<4> **“Save Reception List”** 功能：

儲存目前在“CAN Message Receive Area”中所有的 CAN 訊息至指定的文字檔中，並以 ASCII 編碼方式儲存。(圖 4-18)

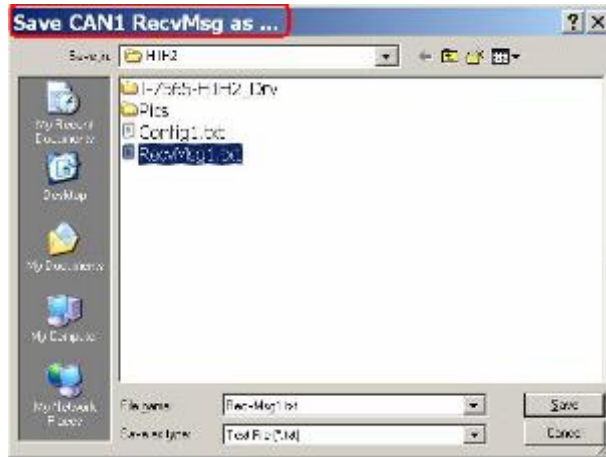


圖 4-18: Save Reception List 功能

<5> **“Load Symbol File”** 功能：

載入指定 Symbol 記錄檔(\*.ini)之 Symbolic CANID Name 資料至 Utility 中。(圖 4-18-1)

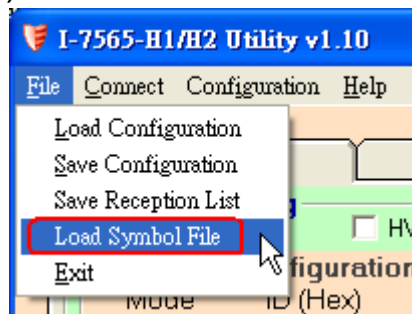


圖 4-18-1: Load Symbol File 功能

## 4.6 狀態列功能

標示目前與模組的連線與每一個 CAN 埠狀態。接下來，將細部說明 I-7565-H1/H2 Utility 的狀態列功能。

若與 I-7565-H1/H2 模組的連線未被建立時，該狀態列顯示的資訊如圖 4-19 所示。

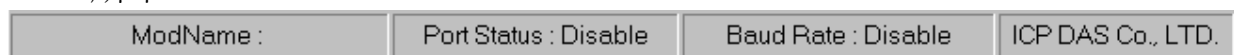


圖 4-19: I-7565-H1/H2 Utility 離線時的狀態列

當與 I-7565-H1/H2 模組成功建立連線後，該狀態列資訊顯示內容如圖

---

4-20 所示，可分為四個區塊：

- (1) **Module Name** => 標示已連線的模組名稱與其使用的虛擬 COM 埠。
- (2) **Port Status** => 標示 CAN 埠是否啓用。
- (3) **Baud Rate** => 標示 CAN 埠目前設定的鮑率。
- (4) **Company** => 標示 ICP DAS Co., LTD 字樣。

ModName : I-7565-H1 (COM 3)	Port Status : Enable	Baud Rate : 1000K	ICP DAS Co., LTD.
-----------------------------	----------------------	-------------------	-------------------

圖 4-20: I-7565-H1/H2 Utility 連線的狀態列

---

## 5. API 函式庫 -- VCI\_CAN.dll

使用者可透過 I-7565-H1/H2 應用程式介面函式庫-VCI\_CAN.dll，輕易地自行開發與 CAN bus 通訊的網路程式。VCI\_CAN 函式庫與展示範例可從泓格科技網頁上取得：

[http://ftp.icpdas.com/pub/cd/fieldbus\\_cd/can/converter/i-7565-h1h2/software/library](http://ftp.icpdas.com/pub/cd/fieldbus_cd/can/converter/i-7565-h1h2/software/library).

### 5.1 API Library 概觀

如圖 5-1 所示，所有由應用程式介面函式庫提供的函式，大致上可分為 5 個群組：

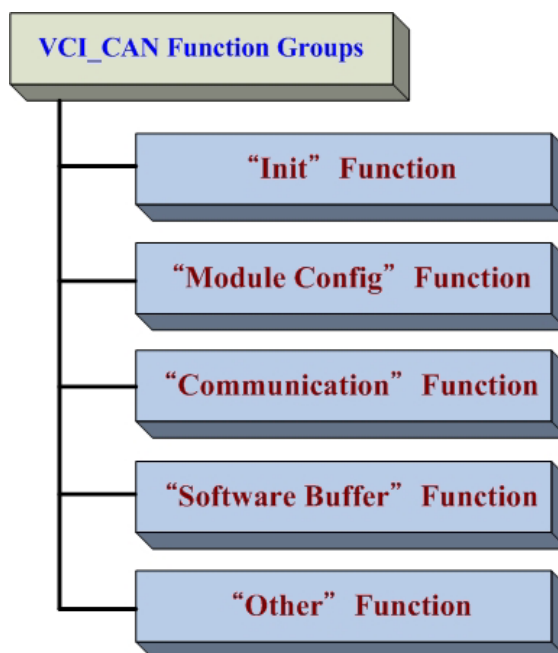


圖 5-1: VCI\_CAN 函式庫內之五個分類群組

#### [ Init Function ]

該群組內的函式功能為：啓用/取消 I-7565-H1/H2 模組 CAN 埠功能。

#### [ Module Config Function ]

該群組內的函式功能為：設定/取得 I-7565-H1/H2 模組參數與資訊 CAN 埠功能。

#### [ Communication Function ]

該群組內的函式功能為：透過 I-7565-H1/H2 模組傳送/接收 CAN 訊息功能。

### [ Software Buffer Function ]

當“VCI\_OpenCAN”功能成功啓用後，所有接收到的 CAN 訊息都將儲存於緩衝區內，以供使用者透過“VCI\_RecvCANMsg”功能取得。每一組 CAN 埠上的緩衝區容量有 65536 位元組。

### [ Other Function ]

該群組內的函式功能為：取得 VCI\_CAN 函式庫資訊或提供使用者撰寫程式的協助。

## 5.2 API Library 功能表

VCI\_CAN.dll 提供的函式庫列於下表：

表 5-1: “Init” 功能表

編號.	函式名稱	說明
1	VCI_OpenCAN	啓用 I-7565-H1/H2 之 CAN 埠功能。
2	VCI_CloseCAN	取消 I-7565-H1/H2 之 CAN 埠功能。

表 5-2: “Module Config” 功能表

編號.	函式名稱	說明
1	VCI_Set_CANFID	在指定的 CAN 埠上設定 CAN Filter-ID。
2	VCI_Get_CANFID	在指定的 CAN 埠上取得 CAN Filter-ID。
3	VCI_Get_CANStatus	取得指定的 CAN 埠狀態。
4	VCI_Clr_BufOverflowLED	清除指定的 CAN 埠緩衝區溢位狀態。
5	VCI_Get_MODInfo	取得模組資訊。
6	VCI_Rst_MOD	重置模組。

表 5-3: “Communication” 功能表

編號.	函式名稱	說明
1	VCI_SendCANMsg	在指定的 CAN 埠傳送 CAN 訊息。
2	VCI_RecvCANMsg	在指定的 CAN 埠接收 CAN 訊息。
3	VCI_EnableHWCyclicTxMsg	利用模組硬體時間，在指定的 CAN 埠上週斯地傳

		送 CAN 訊息。
4	VCI_DisableHWCyclicTxMsg	利用模組硬體時間，在指定的 CAN 埠上週斯地停止傳送 CAN 訊息。

表 5-4: “Software Buffer” 功能表

編號.	函式名稱	說明
1	VCI_Get_RxMsgCnt	取得指定 CAN 埠上已接收到的 CAN 訊息數量。 (該數量是指 Software buffer 所收到的數量)
2	VCI_Get_RxMsgBufIsFull	取得指定 CAN 埠上 Software buffer 狀態。
3	VCI_Clr_RxMsgBuf	清空指定的 CAN 埠 Software buffer。

表 5-5: “Other” 功能表

編號.	函式名稱	說明
1	VCI_Get_DllVer	取得 VCI_CAN 函式庫版本。
2	VCI_DoEvents	釋放 CPU 暫存資源。

### 5.3 使用者應用程式開發流程

下圖是使用者 CAN bus 網路程式開發基本控制流程圖。(圖 5-2)

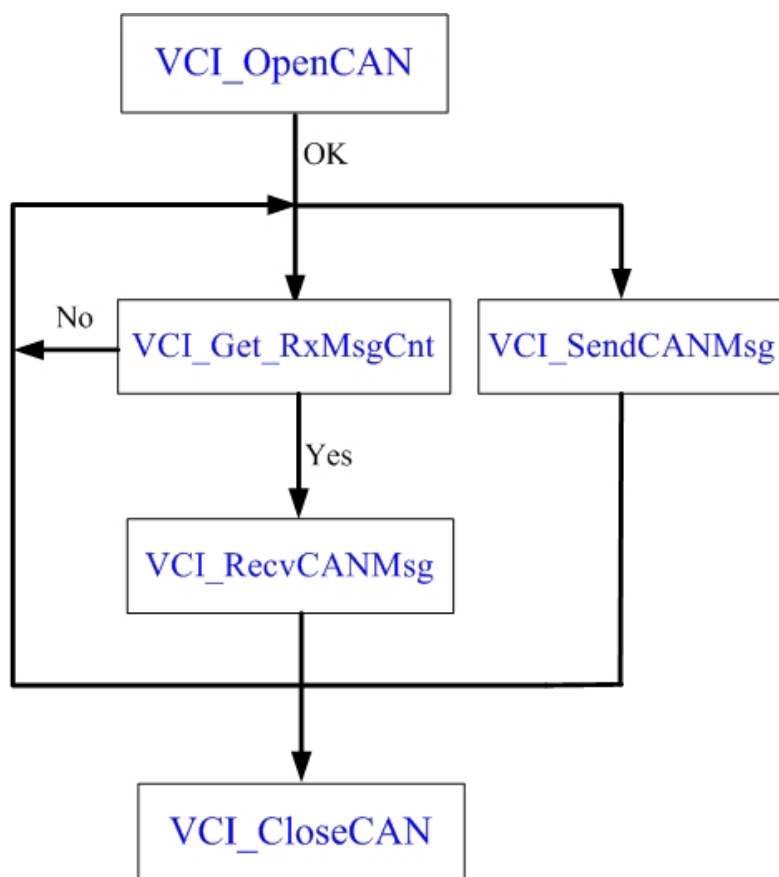


圖 5-2: API 函式庫(VCI\_CAN.dll)流程圖



---

## 5.4 Init Function

該群組內的函式功能為：啓用/取消 I-7565-H1/H2 模組 CAN 埠功能。

### 5.4.1 VCI\_OpenCAN

啓用指定的在 I-7565-H1/H2 模組上的 CAN 埠。在啓用後，使用者可使用“Communication”功能傳送 / 接收 CAN 訊息。

函式原型：

```
int VCI_OpenCAN (  
    PVCI_CAN_PARAM pCANPARAM  
);
```

參數：

**pCANPARAM**    *in*    為結構 **\_VCI\_CAN\_PARAM** 的指標，它是用來設定 CAN 埠通訊參數。

```
typedef struct _VCI_CAN_PARAM{  
    BYTE   DevPort;  
    BYTE   DevType;  
    DWORD  CAN1_Baud;  
    DWORD  CAN2_Baud;  
} _VCI_CAN_PARAM, *PVCI_CAN_PARAM;
```

**DevPort**    *in*    虛擬 COM 埠編號。

**DevType**    *in*    模組種類 (1: I-7565-H1; 2: I-7565-H2)。

**CAN1\_Baud**    *in*    CAN1 埠鮑率  
                  **0**: CAN1 埠除能；**Others**: CAN1 埠致能。

**CAN2\_Baud**    *in*    CAN2 埠鮑率  
                  **0**: CAN2 埠除能；**Others**: CAN2 埠致能。

回傳值：

回傳值為 0 時，代表成功；為其它值時，代表失敗。

範例：

```
int Ret;  
_VCI_CAN_PARAM pCANPARAM;  
  
pCANPARAM.DevPort        = 1;                    // Virtual com port = 1  
pCANPARAM.DevType        = 1;                    // I-7565-H1  
pCANPARAM.CAN1_Baud      = 250000;              // 250 Kbps  
pCANPARAM.CAN2_Baud      = 1000000;             // 1000K bps  
Ret = VCI_OpenCAN(&pCANPARAM);                // Enable CAN port
```

---

### 5.4.2 VCI\_CloseCAN

取消指定的在 I-7565-H1/H2 模組上的 CAN 埠。在取消後，模組將不干涉 CAN bus 網路的通訊。

#### 函式原型:

```
int VCI_CloseCAN (  
    BYTE DevPort  
);
```

#### 參數:

**DevPort** in 虛擬 COM埠編號。

#### 回傳值:

回傳值為0時，代表成功；為其它值時，代表失敗。

#### 範例:

```
int Ret;  
BYTE ComPort;  
  
ComPort = 1;  
Ret = VCI_CloseCAN(ComPort);           // Disable CAN port
```

---

## 5.5 Module Config Function

該群組內的函式功能為：設定/取得 I-7565-H1/H2 模組參數與資訊 CAN 埠功能。

### 5.5.1 VCI\_Set\_CANFID

設定 指定的 CAN 埠上的 Filter-ID。

#### 函式原型:

```
int VCI_Set_CANFID (  
    BYTE CAN_No,  
    PVCI_CAN_FID pCANFID  
);
```

#### 參數：

<b>CAN_No</b>	<i>in</i>	給予的CAN埠編號。
<b>pCANFID</b>	<i>in</i>	結構_VCI_CAN_FilterID的指標，用於設定CAN Filter-ID資料。

```
typedef struct _VCI_CAN_FilterID{  
    WORD SSFF_Num;  
    WORD GSFF_Num;  
    WORD SEFF_Num;  
    WORD GEFF_Num;  
    WORD SSFF_FID[512];  
    DWORD GSFF_FID[512];  
    DWORD SEFF_FID[512];  
    DWORD GEFF_FID[512];  
}_VCI_CAN_FilterID, *PVCI_CAN_FID;
```

<b>SSFF_Num</b>	<i>in</i>	單一11位元CAN Filter-ID數量。
<b>GSFF_Num</b>	<i>in</i>	多個11位元CAN Filter-ID數量。
<b>SEFF_Num</b>	<i>in</i>	單一29位元CAN Filter-ID數量。
<b>GEFF_Num</b>	<i>in</i>	多個29位元CAN Filter-ID數量。
<b>SSFF_FID[512]</b>	<i>in</i>	單一11位元CAN Filter-ID資料陣列。
<b>GSFF_FID[512]</b>	<i>in</i>	多個11位元CAN Filter-ID資料陣列。
<b>SEFF_FID[512]</b>	<i>in</i>	單一29位元CAN Filter-ID資料陣列。
<b>GEFF_FID[512]</b>	<i>in</i>	多個29位元CAN Filter-ID資料陣列。

#### 回傳值：

回傳值為0時，代表成功；為其它值時，代表失敗。

#### 範例：

```
int Ret;
```

---

```

BYTE CAN_No;
_VCI_CAN_FilterID pCANFID1;
//Single 11-bit Filter-ID
WORD SSFID[3]={0x0003, 0x0002, 0x0001};
//Group 11-bit Filter-ID
DWORD GSFID[2]={0x00300040, 0x00100020};
//Single 29-bit Filter-ID
DWORD SEFID[3]={0x00000013, 0x00000012, 0x00000011};
//Group 29-bit Filter-ID
DWORD GEFID[4]={0x00000300, 0x00000400, 0x00000100, 0x00000200};

CAN_No=1;
pCANFID1.SSFF_Num = sizeof(SSFID)/sizeof(WORD);
pCANFID1.GSFF_Num = sizeof(GSFID)/sizeof(DWORD);
pCANFID1.SEFF_Num = sizeof(SEFID)/sizeof(DWORD);
pCANFID1.GEFF_Num = sizeof(GEFID)/sizeof(DWORD);
memcpy(pCANFID1.SSFF_FID, SSFID, pCANFID1.SSFF_Num*2);
memcpy(pCANFID1.GSFF_FID, GSFID, pCANFID1.GSFF_Num*4);
memcpy(pCANFID1.SEFF_FID, SEFID, pCANFID1.SEFF_Num*4);
memcpy(pCANFID1.GEFF_FID, GEFID, pCANFID1.GEFF_Num*4);

Ret = VCI_Set_CANFID(CAN_No, &pCANFID1);    // Set CAN Filter-ID

```

## 5.5.2 VCI\_Get\_CANFID

This function is used to get CAN Filter-ID in the assigned CAN port.

### 函式原型:

```
int VCI_Get_CANFID (  
    BYTE CAN_No,  
    PVCI_CAN_FID pCANFID  
);
```

### 參數:

**CAN\_No**    *in*    指定的 CAN埠編號。  
**pCANFID**   *out*    結構\_VCI\_CAN\_FilterID的指標，用於接收CAN Filter-ID資料。

```
typedef struct _VCI_CAN_FilterID{  
    WORD SSFF_Num;  
    WORD GSFF_Num;  
    WORD SEFF_Num;  
    WORD GEFF_Num;  
    WORD SSFF_FID[512];  
    DWORD GSFF_FID[512];  
    DWORD SEFF_FID[512];  
    DWORD GEFF_FID[512];  
} _VCI_CAN_FilterID, *PVCI_CAN_FID;
```

**SSFF\_Num**    *in*    單一11位元CAN Filter-ID數量。  
**GSFF\_Num**    *in*    多個11位元CAN Filter-ID數量。  
**SEFF\_Num**    *in*    單一29位元CAN Filter-ID數量。  
**GEFF\_Num**    *in*    多個29位元CAN Filter-ID數量。  
**SSFF\_FID[512]**   *in*    單一11位元CAN Filter-ID資料陣列。  
**GSFF\_FID[512]**   *in*    多個11位元CAN Filter-ID資料陣列。  
**SEFF\_FID[512]**   *in*    單一29位元CAN Filter-ID資料陣列。  
**GEFF\_FID[512]**   *in*    多個29位元CAN Filter-ID資料陣列。

### 回傳值:

回傳值為0時，代表成功；為其它值時，代表失敗。

### 範例:

```
int Ret;  
BYTE CAN_No;  
_VCI_CAN_FilterID pCANFID;  
WORD SID11_EndNum=0, GID11_EndNum=0;  
WORD SID29_EndNum=0, GID29_EndNum=0;  
  
CAN_No=1;
```

---

```
Ret = VCI_Get_CANFID(CAN_No, &pCANFID); // Get CAN Filter-ID  
SID11_EndNum = CANFID.SSFF_Num;  
GID11_EndNum = CANFID.GSFF_Num;  
SID29_EndNum = CANFID.SEFF_Num;  
GID29_EndNum = CANFID.GEFF_Num;
```

### 5.5.3 VCI\_Get\_CANStatus

取得指定的 CAN 埠狀態。

函式原型：

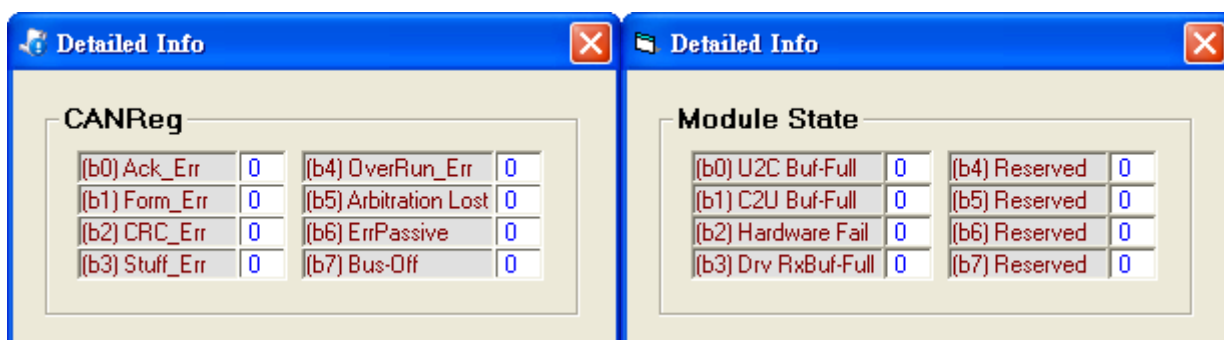
```
int VCI_Get_CANStatus (  
    BYTE CAN_No,  
    PVCI_CAN_STATUS pCANStatus  
);
```

參數：

**CAN\_No**     *in*     指定的 CAN埠編號。  
**pCANStatus**   *out*    結構\_VCI\_CAN\_STATUS的指標，用於接收CAN埠狀態資訊。

```
typedef struct _VCI_CAN_STATUS{  
    DWORD CurCANBaud;  
    BYTE  CANReg;  
    BYTE  CANTxErrCnt;  
    BYTE  CANRxErrCnt;  
    BYTE  MODState;  
    DWORD Reserved;  
} _VCI_CAN_STATUS, *PVCI_CAN_STATUS;
```

**CurCANBaud**   *in*     回傳指定的CAN埠鮑率。  
**CANReg**       *in*     回傳指定的CAN埠暫存器值。  
**CANTxErrCnt**   *in*     回傳指定的CAN埠傳送錯誤計數器。  
**CANRxErrCnt**   *in*     回傳指定的CAN埠接收錯誤計數器。  
**MODState**     *in*     回傳模組狀態。



“CANReg”之bit資訊

“MODState” 之bit資訊

回傳值：

回傳值為0時，代表成功；為其它值時，代表失敗。

範例：

---

```
Int Ret;
BYTE CAN_No, Module_State;
_VCI_CAN_STATUS CANSTA;

CAN_No=1;
Ret = VCI_Get_CANStatus(CAN_No, &CANSTA); // Get CAN port status
Module_State = CANSTA.MODState;
```



---

#### 5.5.4 VCI\_Clr\_BufOverflowLED

清除指定的 CAN 埠緩衝器錯誤狀態(此時，模組上的 ERR LED 以每秒一次的頻率閃爍)。

##### 函式原型：

```
int VCI_Clr_BufOverflowLED (  
    BYTE CAN_No,  
);
```

##### 參數：

**CAN\_No** *in* 指定的CAN埠編號。

##### 回傳值：

回傳值為0時，代表成功；為其它值時，代表失敗。

##### 範例：

```
Int Ret;  
BYTE CAN_No;  
  
CAN_No=1;  
Ret = VCI_Clr_BufOverflowLED(CAN_No); // Clear Buffer Overflow LED
```

---

### 5.5.5 VCI\_Get\_MODInfo

取得模組資訊。

#### 函式原型：

```
int VCI_Get_MODInfo (  
    PVCI_MOD_INFO pMODInfo  
);
```

#### 參數：

**pMODInfo** out 結構\_VCI\_MODULE\_INFO的指標，用於接收模組資訊。

```
typedef struct _VCI_MODULE_INFO{  
    char Mod_ID[12];  
    char FW_Ver[12];  
    char HW_SN[16];  
} _VCI_MODULE_INFO, *PVCI_MOD_INFO;
```

**Mod\_ID[12]** out 回傳模組名稱字串。

**FW\_Ver[12]** out 回傳模組韌體版本字串。

**HW\_SN[16]** out 回傳模組硬體唯一序號字串。

#### 回傳值：

回傳值為0時，代表成功；為其它值時，代表失敗。

#### 範例：

```
int Ret;  
char Module_ID[12], Firmware_Ver[12], Hardware_SN[16];  
_VCI_MODULE_INFO CAN_ModInfo;  
  
Ret = VCI_Get_MODInfo(&CAN_ModInfo); // Get module information  
sprintf(Module_ID, "%s", CAN_ModInfo.Mod_ID);  
sprintf(Firmware_Ver, "%s", CAN_ModInfo.FW_Ver);  
sprintf(Hardware_SN, "%s", CAN_ModInfo.HW_SN);
```

---

### 5.5.6 VCI\_Rst\_MOD

重置模組。

函式原型：

```
int VCI_Rst_MOD (  
    void  
);
```

參數：

**None**

回傳值：

回傳值為0時，代表成功；為其它值時，代表失敗。

範例：

```
int Ret;  
Ret = VCI_Rst_MOD();           // Reset Module
```

---

### 5.5.7 VCI\_Set\_MOD\_Ex

此擴充函式用來設定模組新增功能之相關參數，亦有保留陣列空間給未來新功能使用。

#### 函式原型：

```
int VCI_Set_MOD_Ex (  
    BYTE CfgData[512]  
);
```

#### 參數：

**CfgData[512]**    *in*    模組設定參數陣列。  
                  [Byte 0] : CAN1 Listen Only Function (0:Disable, 1:Enable)  
                  [Byte 1] : CAN2 Listen Only Function (0:Disable, 1:Enable)

#### 回傳值：

回傳值為0時，代表成功；為其它值時，代表失敗。

#### 範例：

```
_VCI_CAN_PARAM pCANPARAM;  
BYTE Mod_CfgData[512];  
  
//Listen Only Mode Setting  
Mod_CfgData[0] = 1; //CAN1 => 0:Disable, 1:Enable  
Mod_CfgData[1] = 0; //CAN2 => 0:Disable, 1:Enable  
VCI_Set_MOD_Ex(Mod_CfgData);  
  
//Open CAN  
pCANPARAM.DevPort = 1;  
pCANPARAM.DevType = I7565H2;  
pCANPARAM.CAN1_Baud = 1000000;  
pCANPARAM.CAN2_Baud = 1000000;  
Ret = VCI_OpenCAN(&pCANPARAM);
```

---

## 5.6 Communication Function

該群組內的函式功能為：透過 I-7565-H1/H2 模組傳送/接收 CAN 訊息功能。

### 5.6.1 VCI\_SendCANMsg

利用指定的 CAN 埠傳送 CAN 訊息。

#### 函式原型：

```
int VCI_SendCANMsg (  
    BYTE CAN_No,  
    PVCI_CAN_MSG pCANMsg  
);
```

#### 參數：

<b>CAN_No</b>	<i>in</i>	指定的CAN埠編號。
<b>pCANMsg</b>	<i>in</i>	結構_VCI_CAN_MSG的指標，用於設定CAN訊息參數。
		<pre>typedef struct _VCI_CAN_MSG{     BYTE Mode;     BYTE RTR;     BYTE DLC;     BYTE Reserved;     DWORD ID;     DWORD TimeL;     DWORD TimeH;     BYTE Data[8]; }_VCI_CAN_MSG, *PVCI_CAN_MSG;</pre>
<b>Mode</b>	<i>in</i>	CAN訊息模式。0: 11-bit(2.0A)；1: 29-bit(2.0B)
<b>RTR</b>	<i>in</i>	CAN訊息RTR旗標。0: No RTR；1: RTR
<b>DLC</b>	<i>in</i>	CAN訊息資料長度。(0~8)
<b>ID</b>	<i>in</i>	CAN訊息ID。
<b>TimeL</b>	<i>in</i>	CAN訊息時戳。(低雙字元組)
<b>TimeH</b>	<i>in</i>	CAN訊息時戳。(高雙字元組)
<b>Data[8]</b>	<i>in</i>	CAN訊息資料陣列。

#### 回傳值：

回傳值為0時，代表成功；為其它值時，代表失敗。

#### 範例：

```
int Ret;  
BYTE CAN_No;  
_VCI_CAN_MSG CAN_SendMsg;
```

---

```
CAN_No=1;
CAN_SendMsg.Mode = 1;
CAN_SendMsg.RTR = 0;
CAN_SendMsg.ID      = 0x1;
CAN_SendMsg.DLC    = 8;
CAN_SendMsg.Data[0]= 0x12;
CAN_SendMsg.Data[1]= 0x34;
CAN_SendMsg.Data[2]= 0x56;
CAN_SendMsg.Data[3]= 0x78;
CAN_SendMsg.Data[4]= 0x90;
CAN_SendMsg.Data[5]= 0xAB;
CAN_SendMsg.Data[6]= 0xCD;
CAN_SendMsg.Data[7]= 0xEF;
Ret = VCI_SendCANMsg(CAN_No, &CAN_SendMsg); // Send CAN Msg
```

## 5.6.2 VCI\_RecvCANMsg

接收從自指定的 CAN 埠之軟體緩衝區(Software buffer)的 CAN 訊息。

### 函式原型：

```
int VCI_RecvCANMsg (  
    BYTE CAN_No,  
    PVCI_CAN_MSG pCANMsg  
);
```

### 參數：

<b>CAN_No</b>	<i>in</i>	指定的CAN埠編號。
<b>pCANMsg</b>	<i>out</i>	結構_VCI_CAN_MSG的指標，用於設定CAN訊息參數。

```
typedef struct _VCI_CAN_MSG{  
    BYTE Mode;  
    BYTE RTR;  
    BYTE DLC;  
    BYTE Reserved;  
    DWORD ID;  
    DWORD TimeL;  
    DWORD TimeH;  
    BYTE Data[8];  
}_VCI_CAN_MSG, *PVCI_CAN_MSG;
```

<b>Mode</b>	<i>out</i>	CAN訊息模式。0: 11-bit(2.0A)； 1: 29-bit(2.0B)
<b>RTR</b>	<i>out</i>	CAN訊息RTR旗標。0: No RTR； 1: RTR
<b>DLC</b>	<i>out</i>	CAN訊息資料長度。(0~8)
<b>ID</b>	<i>out</i>	CAN訊息ID。
<b>TimeL</b>	<i>out</i>	CAN訊息時戳。(低雙字元組)
<b>TimeH</b>	<i>out</i>	CAN訊息時戳。(高雙字元組)
<b>Data[8]</b>	<i>out</i>	CAN訊息資料陣列。

### 回傳值：

回傳值為0時，代表成功；為其它值時，代表失敗。

### 範例：

```
int Ret, i;  
BYTE CAN_No;  
BYTE CANMsg_Mode, CANMsg_RTR, CANMsg_DLC, CANMsg_Data[8];  
DWORD CANMsg_ID, CANMsg;  
Double CANMsg_Time;  
_VCI_CAN_MSG CAN_RecvMsg;  
  
CAN_No=1;  
Ret = VCI_RecvCANMsg(CAN_No, &CAN_RecvMsg); // Recv CAN Msg  
CANMsg_Mode = CAN_RecvMsg.Mode;
```

---

```
CANMsg_RTR = CAN_RecvMsg.RTR;  
CANMsg_ID = CAN_RecvMsg.ID;  
CANMsg_DLC = CAN_RecvMsg.DLC;  
CANMsg_Time = (double)(CAN_RecvMsg.TimeH*pow(2.0,32.0))+  
                (double)((double)CAN_RecvMsg.TimeL/10000);  
For(i=0; i< CANMsg_DLC; i++){  
    CANMsg_Data[i] = CAN_RecvMsg.Data[i]  
}
```



### 5.6.3 VCI\_EnableHWCyclicTxMsg

利用模組硬體時鐘，在指定的 CAN 埠傳送 CAN 訊息，可提供較電腦更精準時鐘傳送訊息。

在韌體 v1.05 版以後，支援 5 組硬體時鐘(No:0~4)來傳送 CAN 訊息，此函式功能預設使用硬體時鐘編號 0 來傳送 CAN 訊息。

#### 函式原型：

```
int VCI_EnableHWCyclicTxMsg (  
    BYTE CAN_No,  
    PVCI_CAN_MSG pCANMsg,  
    DWORD TimePeriod,  
    DWORD TransmitTimes  
);
```

#### 參數：

<b>CAN_No</b>	<i>in</i>	指定的CAN埠編號。
<b>pCANMsg</b>	<i>in</i>	結構_VCI_CAN_MSG的指標，用於設定CAN訊息參數。 <pre>typedef struct _VCI_CAN_MSG{     BYTE Mode;     BYTE RTR;     BYTE DLC;     BYTE Reserved;     DWORD ID;     DWORD TimeL;     DWORD TimeH;     BYTE Data[8]; }_VCI_CAN_MSG, *PVCI_CAN_MSG;</pre>
<b>Mode</b>	<i>in</i>	CAN訊息模式。0: 11-bit(2.0A)； 1: 29-bit(2.0B)
<b>RTR</b>	<i>in</i>	CAN訊息RTR旗標。0: No RTR； 1: RTR
<b>DLC</b>	<i>in</i>	CAN訊息資料長度。(0~8)
<b>ID</b>	<i>in</i>	CAN訊息ID。
<b>TimeL</b>	<i>in</i>	CAN訊息時戳。(低雙字元組)
<b>TimeH</b>	<i>in</i>	CAN訊息時戳。(高雙字元組)
<b>Data[8]</b>	<i>in</i>	CAN訊息資料陣列。
<b>TimePeriod</b>	<i>in</i>	設定模組硬體時鐘時間週期，以利傳送CAN訊息。若該值為零時，該函式將失效。
<b>TransmitTimes</b>	<i>in</i>	設定傳送CAN訊息次數。若該值為零時，它代表CAN訊息將以TimePeriod值永遠地傳送。

#### 回傳值：

回傳值為0時，代表成功；為其它值時，代表失敗。

---

## 範例：

```
int Ret;
BYTE CAN_No;
_VCI_CAN_MSG CAN_SendMsg;

CAN_No=1;
CAN_SendMsg.Mode = 1;
CAN_SendMsg.RTR = 0;
CAN_SendMsg.ID = 0x1;
CAN_SendMsg.DLC = 8;
CAN_SendMsg.Data[0]= 0x12;
CAN_SendMsg.Data[1]= 0x34;
CAN_SendMsg.Data[2]= 0x56;
CAN_SendMsg.Data[3]= 0x78;
CAN_SendMsg.Data[4]= 0x90;
CAN_SendMsg.Data[5]= 0xAB;
CAN_SendMsg.Data[6]= 0xCD;
CAN_SendMsg.Data[7]= 0xEF;

//以10毫秒的週期，傳送200次的CAN訊息後停止。
Ret = VCI_EnableHWCyclicTxMsg(CAN_No, &CAN_SendMsg, 10, 200);

//以10毫秒的週期性傳送CAN訊息。
Ret = VCI_EnableHWCyclicTxMsg(CAN_No, &CAN_SendMsg, 10, 0);
```

---

#### 5.6.4 VCI\_DisableHWCyclicTxMsg

停止模組硬體時鐘(預設編號 0)之 CAN 訊息傳送。

##### 函式原型：

```
int VCI_DisableHWCyclicTxMsg (  
    void  
);
```

##### 參數：

None

##### 回傳值：

回傳值為0時，代表成功；為其它值時，代表失敗。

##### 範例：

```
int Ret;  
Ret = VCI_DisableHWCyclicTxMsg(); // Disable module hardware timer
```

### 5.6.5 VCI\_EnableHWCyclicTxMsgNo

利用模組硬體時鐘，在指定的 CAN 埠傳送 CAN 訊息，可提供較電腦更精準時鐘傳送訊息。

在韌體 v1.05 版以後，支援 5 組硬體時鐘(No:0~4)來傳送 CAN 訊息，此函式功能可用來指定硬體時鐘編號來傳送 CAN 訊息。

#### 函式原型：

```
int VCI_EnableHWCyclicTxMsgNo (  
    BYTE CAN_No,  
    BYTE Mode,  
    BYTE RTR,  
    BYTE DLC,  
    DWORD ID,  
    BYTE Data[8],  
    DWORD TimePeriod,  
    DWORD TransmitTimes,  
    BYTE HW_TimerNo  
);
```

#### 參數：

<b>CAN_No</b>	<i>in</i>	指定的CAN埠編號。
<b>Mode</b>	<i>in</i>	CAN訊息模式。0: 11-bit(2.0A)； 1: 29-bit(2.0B)
<b>RTR</b>	<i>in</i>	CAN訊息RTR旗標。0: No RTR； 1: RTR
<b>DLC</b>	<i>in</i>	CAN訊息資料長度。(0~8)
<b>ID</b>	<i>in</i>	CAN訊息ID。
<b>Data[8]</b>	<i>in</i>	CAN訊息資料陣列。
<b>TimePeriod</b>	<i>in</i>	設定模組硬體時鐘時間週期，以利傳送CAN訊息。若該值為零時，該函式將失效。
<b>TransmitTimes</b>	<i>in</i>	設定傳送CAN訊息次數。若該值為零時，它代表CAN訊息將以TimePeriod值永遠地傳送。
<b>HW_TimerNo</b>	<i>in</i>	指定的硬體時鐘編號。

#### 回傳值：

回傳值為0時，代表成功；為其它值時，代表失敗。

#### 範例：

```
int Ret;  
BYTE CAN_No;  
BYTE Mode, RTR, DLC, Data[8], HW_TimerNo;  
DWORD ID;  
  
CAN_No=1;
```

---

```
Mode    = 1;  
RTR     = 0;  
ID      = 0x1;  
DLC     = 8;  
Data[0] = 0x12;  
Data[1] = 0x34;  
Data[2] = 0x56;  
Data[3] = 0x78;  
Data[4] = 0x90;  
Data[5] = 0xAB;  
Data[6] = 0xCD;  
Data[7] = 0xEF;
```

//以10毫秒的週期，傳送200次的CAN訊息後停止，藉由使用硬體時鐘編號1。

```
HW_TimerNo = 1;
```

```
Ret = VCI_EnableHWCyclicTxMsgNo(CAN_No, Mode, RTR, DLC, ID, Data, 10,  
200, HW_TimerNo);
```

### 5.6.6 VCI\_EnableHWCyclicTxMsgNo\_Ex

利用模組硬體時鐘，在指定的 CAN 埠傳送 CAN 訊息，可提供較電腦更精準時鐘傳送訊息。

在韌體 v1.05 版以後，支援 5 組硬體時鐘(No:0~4)來傳送 CAN 訊息，此函式功能可用來指定硬體時鐘編號及調整 CAN Data 數值內容來傳送 CAN 訊息。

#### 函式原型：

```
int VCI_EnableHWCyclicTxMsgNo_Ex (  
    BYTE CAN_No,  
    BYTE Mode,  
    BYTE RTR,  
    BYTE DLC,  
    DWORD ID,  
    BYTE Data[8],  
    DWORD TimePeriod,  
    DWORD TransmitTimes,  
    BYTE HW_TimerNo  
    BYTE AddMode,  
    DWORD DLAddVal,  
    DWORD DHAddVal  
);
```

#### 參數：

<b>CAN_No</b>	<i>in</i>	指定的CAN埠編號。
<b>Mode</b>	<i>in</i>	CAN訊息模式。0: 11-bit(2.0A)； 1: 29-bit(2.0B)
<b>RTR</b>	<i>in</i>	CAN訊息RTR旗標。0: No RTR； 1: RTR
<b>DLC</b>	<i>in</i>	CAN訊息資料長度。(0~8)
<b>ID</b>	<i>in</i>	CAN訊息ID。
<b>Data[8]</b>	<i>in</i>	CAN訊息資料陣列。
<b>TimePeriod</b>	<i>in</i>	設定模組硬體時鐘時間週期，以利傳送CAN訊息。若該值為零時，該函式將失效。
<b>TransmitTimes</b>	<i>in</i>	設定傳送CAN訊息次數。若該值為零時，它代表CAN訊息將以TimePeriod值永遠地傳送。
<b>HW_TimerNo</b>	<i>in</i>	指定的硬體時鐘編號。
<b>AddMode</b>	<i>in</i>	傳送CAN Data之數值遞增模式(0:加法模式; 1:乘法模式)
<b>DLAddVal</b>	<i>in</i>	CANL Data遞增數值 (每次傳送)
<b>DHAddVal</b>	<i>in</i>	CANH Data遞增數值 (每次傳送)

#### 回傳值：

回傳值為0時，代表成功；為其它值時，代表失敗。

---

範例：

```
int Ret;
BYTE CAN_No;
BYTE Mode, RTR, DLC, Data[8], HW_TimerNo;
DWORD ID;

CAN_No=1;
Mode = 1;
RTR = 0;
ID = 0x1;
DLC = 8;
Data[0]= 0x0;
Data[1]= 0x0;
Data[2]= 0x0;
Data[3]= 0x0;
Data[4]= 0x0;
Data[5]= 0x0;
Data[6]= 0x0;
Data[7]= 0x0;

//以10毫秒的週期，傳送200次的CAN訊息後停止，藉由使用硬體時鐘編號1。
//CANL Data每次傳送數值遞增1 (採加法模式)
//CANH Data每次傳送數值遞增2 (採加法模式)
HW_TimerNo = 1;
Ret = VCI_EnableHWCyclicTxMsgNo_Ex(CAN_No, Mode, RTR, DLC, ID, Data,
10, 200, HW_TimerNo, ADDITION_MODE, 1, 2);
```

---

### 5.6.7 VCI\_DisableHWCyclicTxMsgNo

停止指定模組硬體時鐘編號 0~4 之 CAN 訊息傳送。

#### 函式原型：

```
int VCI_DisableHWCyclicTxMsgNo (  
    BYTE HW_TimerNo  
);
```

#### 參數：

**HW\_TimerNo**    *in*    指定的硬體時鐘編號。

#### 回傳值：

回傳值為0時，代表成功；為其它值時，代表失敗。

#### 範例：

```
int Ret;  
BYTE HW_TimerNo;  
  
//停止硬體時鐘編號1之CAN訊息傳送  
HW_TimerNo = 1;  
Ret = VCI_DisableHWCyclicTxMsgNo(HW_TimerNo);
```



---

## 5.7 Software Buffer Function

當“VCI\_OpenCAN”功能成功啓用後，所有接收到的 CAN 訊息都將儲存於緩衝區內，以供使用者透過“VCI\_RecvCANMsg”功能取得。每一組 CAN 埠上的緩衝區容量有 65536 位元組。

### 5.7.1 VCI\_Get\_RxMsgCnt

此函式用來取得 API 函式庫之軟體緩衝區內尚未被使用者程式接收之 CAN 訊息數量。

函式原型：

```
int VCI_Get_RxMsgCnt (  
    BYTE CAN_No,  
    DWORD* RxMsgCnt  
);
```

參數：

**CAN\_No**    *in*    指定的CAN埠編號。  
**RxMsgCnt**   *out*    該指標用於取得軟體緩衝區接收CAN訊息計數值。

回傳值：

回傳值為0時，代表成功；為其它值時，代表失敗。

範例：

```
int Ret;  
BYTE CAN_No;  
DWORD RxMsgCnt;  
  
CAN_No=1;  
Ret = VCI_Get_RxMsgCnt(CAN_No, &RxMsgCnt);
```

---

### 5.7.2 VCI\_Get\_RxMsgBufIsFull

取得指定的 CAN 埠軟體接收緩衝區的状态。若該緩衝區滿載時，它代表可能有些 CAN 訊息已遺失。

#### 函式原型：

```
int VCI_Get_RxMsgBufIsFull (  
    BYTE CAN_No,  
    BYTE* Flag  
);
```

#### 參數：

<b>CAN_No</b>	<i>in</i>	指定的CAN埠編號。
<b>Flag</b>	<i>out</i>	該指標用於取得軟體緩衝區状态。若該值為0時，代表軟體緩衝區尚未滿載；反之亦然，若該值為1時，它代表軟體緩衝區已滿。

#### 回傳值：

回傳值為0時，代表成功；為其它值時，代表失敗。

#### 範例：

```
int Ret;  
BYTE CAN_No;  
BYTE RxSoftBufFull_Flag;  
  
CAN_No=1;  
Ret = VCI_Get_RxMsgBufIsFull(CAN_No, &RxSoftBufFull_Flag);
```

---

### 5.7.3 VCI\_Clr\_RxMsgBuf

清空指定 CAN 埠之軟體接收緩衝區內尚未被使用者程式接收之所有 CAN 訊息。

#### 函式原型：

```
int VCI_Clr_RxMsgBuf (  
    BYTE CAN_No,  
);
```

#### 參數：

**CAN\_No** *in* 指定的CAN埠編號。

#### 回傳值：

回傳值為0時，代表成功；為其它值時，代表失敗。

#### 範例：

```
int Ret;  
BYTE CAN_No;  
  
CAN_No=1;  
Ret = VCI_Clr_RxMsgBuf(CAN_No);
```

---

### 5.7.4 VCI\_Get\_TxMsgCnt

取得 API 函式庫之軟體發送緩衝區內尚未被送出之 CAN 訊息數量。

#### 函式原型：

```
int VCI_Get_TxMsgCnt (  
    BYTE CAN_No,  
    DWORD* TxMsgCnt  
);
```

#### 參數：

**CAN\_No**    *in*    指定的CAN埠編號。  
**RxMsgCnt**    *out*    該指標用於取得軟體緩衝區內尚未被送出之CAN訊息數量。

#### 回傳值：

回傳值為0時，代表成功；為其它值時，代表失敗。

#### 範例：

```
int Ret;  
BYTE CAN_No;  
DWORD TxMsgCnt;  
  
CAN_No=1;  
Ret = VCI_Get_TxMsgCnt(CAN_No, &TxMsgCnt);
```

---

### 5.7.5 VCI\_Clr\_TxMsgBuf

清空指定 CAN 埠之軟體發送緩衝區內尚未被送出的所有 CAN 訊息。

#### 函式原型：

```
int VCI_Clr_TxMsgBuf (  
    BYTE CAN_No,  
);
```

#### 參數：

**CAN\_No**    *in*    指定的CAN埠編號。

#### 回傳值：

回傳值為0時，代表成功；為其它值時，代表失敗。

#### 範例：

```
int Ret;  
BYTE CAN_No;  
  
CAN_No=1;  
Ret = VCI_Clr_TxMsgBuf (CAN_No);
```

---

### 5.7.6 VCI\_Get\_TxSentCnt

取得 API 函式庫之已送出的所有 CAN 訊息數量。

#### 函式原型：

```
int VCI_Get_TxSentCnt (  
    BYTE CAN_No,  
    DWORD* TxSentCnt  
);
```

#### 參數：

**CAN\_No**    *in*    指定的CAN埠編號。  
**TxSentCnt**    *out*    該指標用於取得已送出的所有CAN訊息數量。

#### 回傳值：

回傳值為0時，代表成功；為其它值時，代表失敗。

#### 範例：

```
int Ret;  
BYTE CAN_No;  
DWORD TxMsgCnt;  
  
CAN_No=1;  
Ret = VCI_Get_TxSentCnt(CAN_No, &TxSentCnt);
```

---

### 5.7.7 VCI\_Clr\_TxSentCnt

清空指定 CAN 埠之已發送的所有 CAN 訊息計數值。

#### 函式原型：

```
int VCI_Clr_TxSentCnt (  
    BYTE CAN_No,  
);
```

#### 參數：

**CAN\_No** *in* 指定的CAN埠編號。

#### 回傳值：

回傳值為0時，代表成功；為其它值時，代表失敗。

#### 範例：

```
int Ret;  
BYTE CAN_No;  
  
CAN_No=1;  
Ret = VCI_Clr_TxSentCnt (CAN_No);
```

---

## 5.8 User Defined ISR Function

該群組內的函式功能為：當收到使用者所設定之 CAN-ID message 時，立即會啟動使用者自訂義之函式功能。

### 5.8.1 VCI\_Set\_UserDefISR

用來設定使用者自訂義 Callback 函式及符合觸發功能之 CAN message 條件。

#### 函式原型：

```
int VCI_Set_UserDefISR (  
    BYTE ISRNo,  
    BYTE CAN_No,  
    BYTE* Mode  
    DWORD CANID,  
    void (*UserDefISR)()  
);
```

#### 參數：

<b>ISRNo</b>	<i>in</i>	指定的ISRNo編號。(合法值: 0~7)
<b>CAN_No</b>	<i>in</i>	指定的CAN埠編號。(0: 表示所有 Port 均符合)
<b>Mode</b>	<i>in</i>	指定的CAN Message Mode。(2: 表示所有 Mode 均符合)
<b>CANID</b>	<i>in</i>	指定的CAN Message ID。(0: 表示所有 ID 均符合)
<b>*UserDefISR</b>	<i>in</i>	指定的使用者自訂義函式之函式指標。

#### 回傳值：

回傳值為0時，代表成功；為其它值時，代表失敗。

#### 範例：

```
int Ret;  
  
/* UserDefISR (MyTestISR0) 將會被觸發當收到任一個 CANMsg 時 */  
Ret=VCI_Set_UserDefISR(ISRNO_0, ISR_CANPORT_ALL, ISR_CANMODE_ALL,  
ISR_CANID_ALL, MyTestISR0);  
  
/* The UserDefISR (MyTestISR1) 將會被觸發只有當 CAN1 收到 11-bit 且  
CANID=0x100 之 CANMsg */  
Ret=VCI_Set_UserDefISR(ISRNO_1, CAN1, MODE_11BIT, 0x100, MyTestISR1);
```

#### [ 注意事項 ]

1. 使用者自訂義函式之內容需愈簡單愈好 (即執行時間愈短愈好)且符合條



---

件之 **CAN message** 出現頻率愈慢愈好，否則可能會發生自訂義函式執行次數遺失之情形。

### 5.8.2 VCI\_Clr\_UserDefISR

用來移除使用者自訂義函式符合條件之 **Callback** 函式觸發功能。

函式原型：

```
int VCI_Clr_UserDefISR (  
    BYTE ISRNo,  
);
```

參數：

**ISRNo** *in* 指定的ISRNo編號。(合法值: 0~7)

回傳值：

回傳值為0時，代表成功；為其它值時，代表失敗。

範例：

```
/* 移除使用者自訂義之ISRNO_0及ISRNO_1之功能 */  
VCI_Clr_UserDefISR(ISRNO_0);  
VCI_Clr_UserDefISR(ISRNO_1);
```

### 5.8.3 VCI\_Get\_ISRCANData

用來取得使用者自訂義函式符合條件時之最新 **CAN Message Data**。

函式原型：

```
int VCI_Get_ISRCANData (  
    BYTE ISRNo,  
    BYTE* DLC,  
    BYTE Data[8]  
);
```

參數：

**ISRNo** *in* 指定的ISRNo編號。(合法值: 0~7)  
**DLC** *out* 回傳符合UserDefISR條件之CAN Data有效長度。  
**Data[8]** *out* 回傳符合UserDefISR條件之CAN Data資料內容。

---

### 回傳值：

回傳值為0時，代表成功；為其它值時，代表失敗。

### 範例：

```
BYTE ISR1_CANDataLen;  
BYTE ISR1_CANData[8]={0};  
  
VCI_Get_ISRCANData(ISRNO_1, &ISR1_CANDataLen, ISR1_CANData);
```

---

## 5.9 Other Function

該群組內的函式功能為：取得 VCI\_CAN 函式庫資訊或提供使用者撰寫程式的協助。

### 5.9.1 VCI\_Get\_DllVer

取得 VCI\_CAN 函式庫版本編號。

#### 函式原型：

```
DWORD VCI_Get_DllVer (  
    void  
);
```

#### 參數：

*None*

#### 回傳值：

回傳VCI\_CAN函式庫版本。高位元組為主要版本編號、低位元組為次要版本編號。

#### 範例：

```
DWORD DllVer;  
char VCI_DllVer[10];  
  
DllVer = VCI_Get_DllVer();  
sprintf(VCI_DllVer, "v%lu.%02lu", (DllVer>>8)&0xFF, DllVer&0xFF);
```

---

### 5.9.2 VCI\_DoEvents

釋放 CPU 暫時使用的資源。

函式原型：

```
void VCI_DoEvents (  
    void  
);
```

參數：

None

回傳值：

None

範例：

```
VCI_DoEvents();
```

---

## 5.10 Extended Function

該群組內的函式功能為：擴充 I-7565-H1/H2 模組之功能。

### 5.10.1 VCI\_OpenCAN\_Ex

此函式功能與 VCI\_OpenCAN( ) 相同，但它額外提供可設定 CAN 鮑率之 Bit-Timing 取樣點 (即 Tseg2 值)功能，當應用在有電磁波干擾之 CAN bus 場合時 (如：馬達啟動產生干擾)，則可在相同 CAN 鮑率下，選擇 Tseg2 較大值來作通訊。

函式原型：

```
int VCI_OpenCAN_Ex (  
    PVCI_CAN_PARAM_EX pCANPARAMEx  
);
```

參數：

**pCANPARAMEx** in 為結構\_VCI\_CAN\_PARAM\_EX的指標，它是用來設定CAN埠通訊參數及Tseg2參數值。

```
typedef struct _VCI_CAN_PARAM_EX{  
    BYTE DevPort;  
    BYTE DevType;  
    DWORD CAN1_Baud;  
    DWORD CAN2_Baud;  
    BYTE CAN1_T2Val;  
    BYTE CAN2_T2Val;  
    BYTE Reserved[32];  
}_VCI_CAN_PARAM_EX, *PVCI_CAN_PARAM_EX;
```

**DevPort** in 虛擬 COM埠編號。  
**DevType** in 模組種類 (1: I-7565-H1; 2: I-7565-H2)。  
**CAN1\_Baud** in CAN1埠鮑率  
0: CAN1埠除能；Others: CAN1埠致能。  
**CAN2\_Baud** in CAN2埠鮑率  
0: CAN2埠除能；Others: CAN2埠致能。  
**CAN1\_T2Val** in CAN1之Tseg2值。  
**CAN2\_T2Val** in CAN2之Tseg2值。  
**Reserved** in 保留。

回傳值：

回傳值為0時，代表成功；為其它值時，代表失敗。

範例：

---

```
Int Ret;
_VCI_CAN_PARAM_EX pCANPARAM;

pCANPARAM.DevPort = 1;           // Virtual com port = 1
pCANPARAM.DevType = 1;          // I-7565-H1
pCANPARAM.CAN1_Baud = 250000;   // 250 Kbps
pCANPARAM.CAN2_Baud = 1000000;  // 1000K bps
pCANPARAM.CAN1_T2Val = 2;       // CAN1 Tseg2 = 2
pCANPARAM.CAN2_T2Val = 3;       // CAN2 Tseg2 = 3
Ret = VCI_OpenCAN_Ex(&pCANPARAM); // Enable CAN port
```

---

### 5.10.2 VCI\_Get\_CANBaud\_BitTime

此函式用來取得指定 CAN 通道之 CAN 鮑率 Bit-Timing 之參數值，包含 Tseg1, Tseg2 及 SJW 數值，當 CAN bus 通訊有問題時，可用來檢查是否與其它 CAN 設備之鮑率 Bit-Timing 參數值相同。

函式原型：

```
int VCI_Get_CANBaud_BitTime (  
    BYTE CAN_No,  
    BYTE* T1Val,  
    BYTE* T2Val,  
    BYTE* SJWVal  
);
```

參數：

<b>CAN_No</b>	<i>in</i>	指定的CAN埠編號。
<b>T1Val</b>	<i>out</i>	該指標用於取得指定CAN埠之Tseg1值。
<b>T2Val</b>	<i>out</i>	該指標用於取得指定CAN埠之Tseg2值。
<b>SJWVal</b>	<i>out</i>	該指標用於取得指定CAN埠之SJW值。

回傳值：

回傳值為0時，代表成功；為其它值時，代表失敗。

範例：

```
int Ret;  
BYTE CAN_No;  
BYTE T1Val=0, T2Val=0, SJWVal=0;  
  
CAN_No=1;  
Ret = VCI_Get_CANBaud_BitTime (CAN_No, &T1Val, &T2Val, &SJWVal);
```

---

## 5.11 回傳代碼

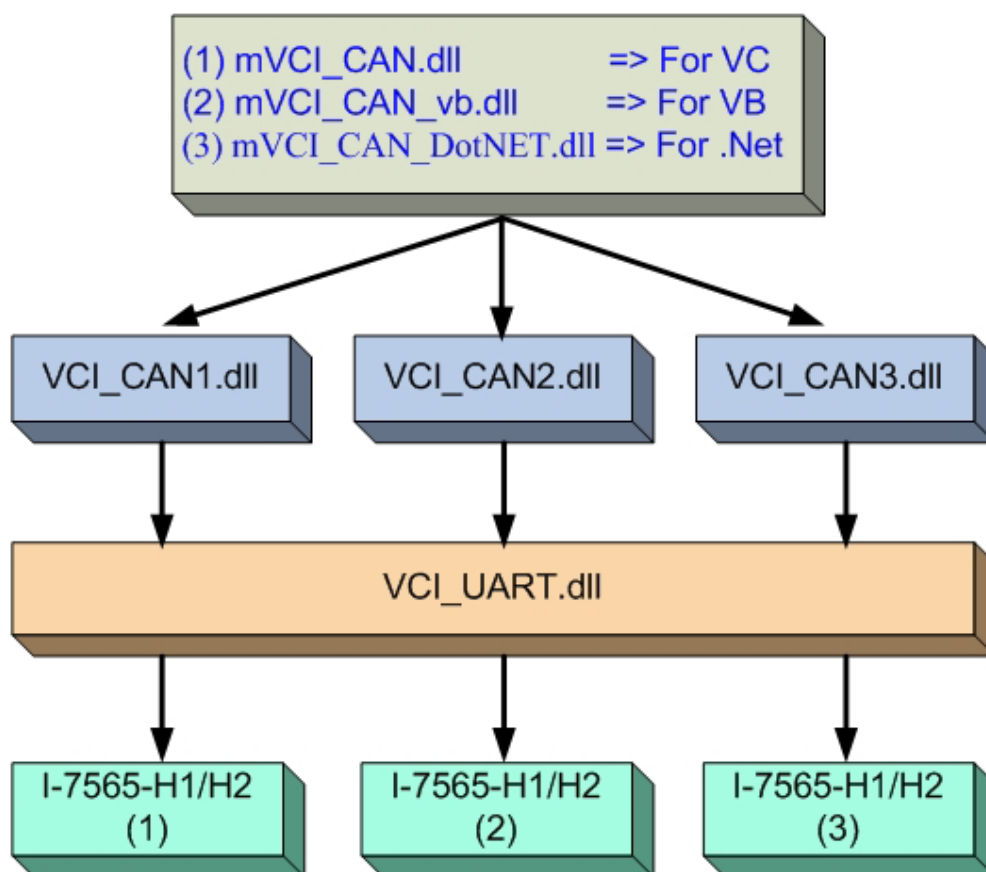
顯示在執行 VCI\_CAN 函式庫各函式的回傳代碼，可依下表得知各錯誤代碼意義。

<b>No_Err</b>	<b>0</b>	<b>//No Error</b>
<b>DEV_ModName_Err</b>	<b>1</b>	<b>//The Module Name Error</b>
<b>DEV_ModNotExist_Err</b>	<b>2</b>	<b>//The Module doesn't exist in this Port</b>
<b>DEV_PortNotExist_Err</b>	<b>3</b>	<b>//The Port doesn't Exist</b>
<b>DEV_PortInUse_Err</b>	<b>4</b>	<b>//The Port is in Used</b>
<b>DEV_PortNotOpen_Err</b>	<b>5</b>	<b>//The Port doesn't Open</b>
<b>CAN_ConfigFail_Err</b>	<b>6</b>	<b>//CAN Hardware Init Fail</b>
<b>CAN_HARDWARE_Err</b>	<b>7</b>	<b>//CAN Hardware Init Fail</b>
<b>CAN_PortNo_Err</b>	<b>8</b>	<b>//The Device doesn't support this CAN Port</b>
<b>CAN_FIDLength_Err</b>	<b>9</b>	<b>//The CAN Filter-ID Number exceed Max Number</b>
<b>CAN_DevDisconnect_Err</b>	<b>10</b>	<b>//The Connection of device is broken</b>
<b>CAN_TimeOut_Err</b>	<b>11</b>	<b>//The Config Command Timeout</b>
<b>CAN_ConfigCmd_Err</b>	<b>12</b>	<b>//The Config Command doesn't support</b>
<b>CAN_ConfigBusy_Err</b>	<b>13</b>	<b>//The Config Command is busy</b>
<b>CAN_RxBufEmpty</b>	<b>14</b>	<b>//The CAN Receive Buffer is empty</b>
<b>CAN_TxBufFull</b>	<b>15</b>	<b>//The CAN Send Buffer is full</b>
<b>CAN_UserDefISRNo_Err</b>	<b>16</b>	<b>//The User Defined ISRNo. Error</b>
<b>CAN_HWSendTimerNo_Err</b>	<b>17</b>	<b>//The HW SendTimer No. Error</b>



## 6. API 函式庫 -- mVCI\_CAN.dll

mVCI\_CAN 函式庫主要用來可以在一個程式中同時控制多個 I-7565-H1/H2 模組，以下為其函式庫連結流程圖：



I-7565-H1/H2 函式庫連結流程圖

mVCI\_CAN 函式庫採用”物件導向”觀念，使用前需先建立 I-7565-H1/H2 模組之物件，之後即可針對各物件來控制對應的模組，以下為 mVCI\_CAN 函式庫之基本使用步驟：

### 6.1 VC 專案

#### (1) 必要檔案：

[1] 複製"mVCI\_CAN.h" 和 "mVCI\_CAN.lib" 檔案至 VC 專案資料夾

- 
- 中。(無需使用到 VCI\_CAN.h 和 VCI\_CAN.lib)
- [2] 複製"mVCI\_CAN.dll", "VCI\_CAN.dll", "VCI\_Uart.dll"三個檔案至 VC 專案下之 Debug 或 Release 資料夾中。

(2) 程式使用方式：

- [1] 加入"mVCI\_CAN.h" 和"mVCI\_CAN.lib"檔案至 VC 專案中。
- [2] 宣告"CMVCI\_CAN"類別之全域物件。  
(如: CMVCI\_CAN I7565H1H2\_Mod[2];)
- [3] 對每個"CMVCI\_CAN"類別之物件，執行 InitDLL()函式。  
(如: I7565H1H2\_Mod[0].InitDLL();)
- [4] 在執行完 InitDLL()函式均成功後，則每一個"CMVCI\_CAN"類別之物件即分別表示為一個 I-7565-H1/H2 模組，之後使用者即可針對不同物件來操作不同的模組。  
(如: I7565H1H2\_Mod[0].mVCI\_OpenCAN();)
- [5] 詳細使用方式，可參考 I-7565-H1/H2 之 VC Demo3 範例。

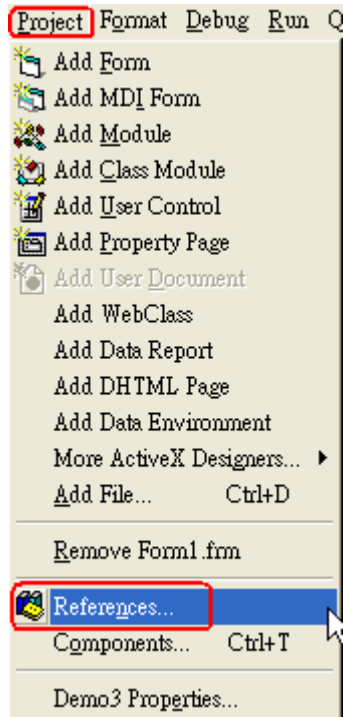
## 6.2 VB 專案

(1) 必要檔案：

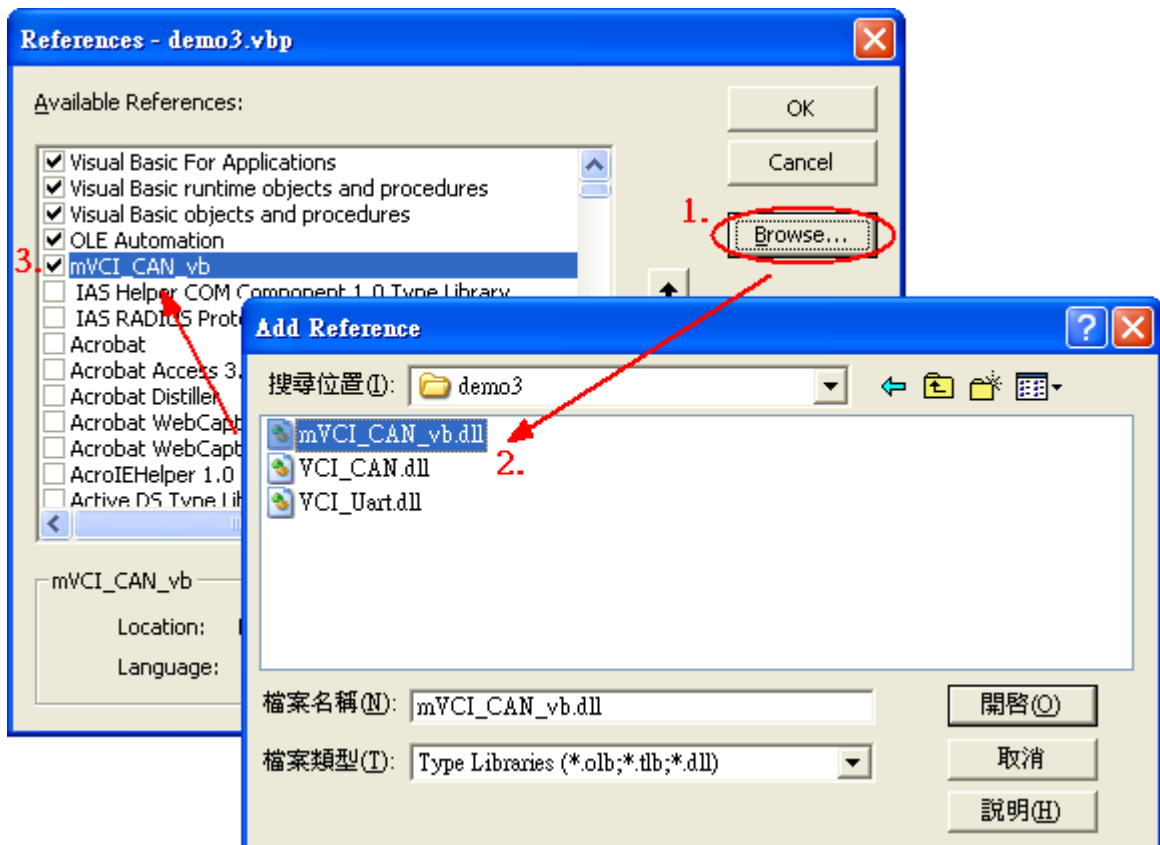
- [1] 複製以下 5 個檔案 - "mVCI\_CAN\_vb\_Register.bat"、  
"mVCI\_CAN\_vb.dll"、"VCI\_CAN.dll"、"VCI\_UART.dll"和"I-7565-H1H2\_Lib.bas"至 VB 專案資料夾中。
- [2] 執行"mVCI\_CAN\_vb\_Register.bat"檔案來註冊"mVCI\_CAN\_vb.dll"資訊至 Windows 系統中。

(2) 程式使用方式：

- [1] 加入"mVCI\_CAN\_vb"參考至 VB 專案中，請依照以下步驟來完成：
- (1) 點選"Project/References..."選項
- (2) 按下"Browser..."鈕，選擇"mVCI\_CAN\_vb.dll"檔案，之後  
"mVCI\_CAN\_vb"參考即會被加入至 VB 專案中。



"Project/References..." option



"mVCI\_CAN\_vb" reference

[2] 宣告"CMVCI\_CAN"類別之全域變數。  
(如: Private I7565H1H2\_Mod(1) As CMVCI\_CAN)

- 
- [3] 對每個"MVCI\_SDK"類別之變數，建立物件並執行 InitDLL()函式。  
(如: Set I7565H1H2\_Mod(0) = New CMVCI\_CAN  
I7565H1H2\_Mod(0).InitDL() )
  - [4] 在執行完 InitDLL()函式均成功後，則每一個"MVCI\_SDK"類別之物件即分別表示為一個 I-7565-H1/H2 模組，之後使用者即可針對不同物件來操作不同的模組。  
(如: I7565H1H2\_Mod(0).mVCI\_OpenCAN())
  - [5] 詳細使用方式，可參考 I-7565-H1/H2 之 VB Demo3 範例。

### 6.3 .Net 專案

#### (1) 必要檔案：

- [1] 複製"mVCI\_CAN\_DotNET.dll", "VCI\_CAN.dll", "VCI\_Uart.dll"三個檔案至.Net 專案下之 Debug 或 Release 資料夾中。

#### (2) 程式使用方式：

- [1] 加入"mVCI\_CAN\_DotNET.dll"檔案至.Net 專案之參考中。
- [2] 輸入"using mVCI\_CAN\_DotNET;"至.Net 專案之開頭。
- [3] 宣告"MVCI\_SDK"類別之全域變數。  
(如: MVCI\_SDK[] I7565H1H2\_Mod = new MVCI\_SDK[2];)
- [4] 對每個"MVCI\_SDK"類別之變數，建立物件並執行 InitDLL()函式。  
(如: I7565H1H2\_Mod[0] = New MVCI\_SDK();  
I7565H1H2\_Mod[0].InitDLL(); )
- [5] 在執行完 InitDLL()函式均成功後，則每一個"MVCI\_SDK"類別之物件即分別表示為一個 I-7565-H1/H2 模組，之後使用者即可針對不同物件來操作不同的模組。  
(如: I7565H1H2\_Mod[0].mVCI\_OpenCAN\_NoStruct();)
- [6] 詳細使用方式，可參考 I-7565-H1/H2 之 .Net Demo2 範例。

---

## 7. 常問問題 (FAQ)

### 7.1 模組連線問題

成功安裝 I-7565-H1/H2 模組驅動程式後，Windows 系統將自動給予該虛擬 COM 埠一個編號，然後，使用者可利用該 COM 埠編號，並透過“I-7565-H1/H2 Utility”連線至 I-7565-H1/H2 模組，以利完成通訊。

#### Q1 : ”Invalid port number” 錯誤訊息？

當使用者試著打開虛擬 COM 埠時，顯示”Invalid port number”，如圖 7.1-1 的錯誤訊息，請依照以下可能情形來排除問題。

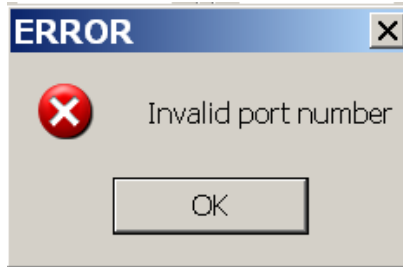


圖 7.1-1: 不存在的 COM 埠編號

- (1) 該 COM 埠不存在，且再次檢查 COM 埠編號。
- (2) 若該虛擬 COM 埠編號值大於 COM16 時，使用者需從 I-7565-H1H2 utility 資料夾內，複製新版的“**MSCOMM32.OCX**”檔案至“C:\WINDOWS\system32\”資料匣下，並取代舊有的版本與重新註冊該 MSCOMM32.ocx(在命令提示字元中，輸入 regsvr32.exe “C:\WINDOWS\system32\ MSCOMM32.ocx”指令)。
- (3) 系統有其它 ComPort 設備和 I-7565-H1/H2 模組使用相同的 ComPort 號碼，請在”裝置管理員”中，將 I-7565-H1/H2 模組之虛擬 ComPort 強制改成其它沒有使用的 ComPort 號碼後，並重新啓動電腦，再重新連線。使用者可執行在 I-7565-H1/H2 Utility v1.11 資料夾內之“Show\_Hidden\_Device.bat”檔，來開啓裝置管理員，接著勾選「檢視 / 顯示隱藏裝置」，即可在“連接埠 (COM 和 LPT)”項目中顯示系統所有 ComPort 使用情形，如圖 7.1-1-1。

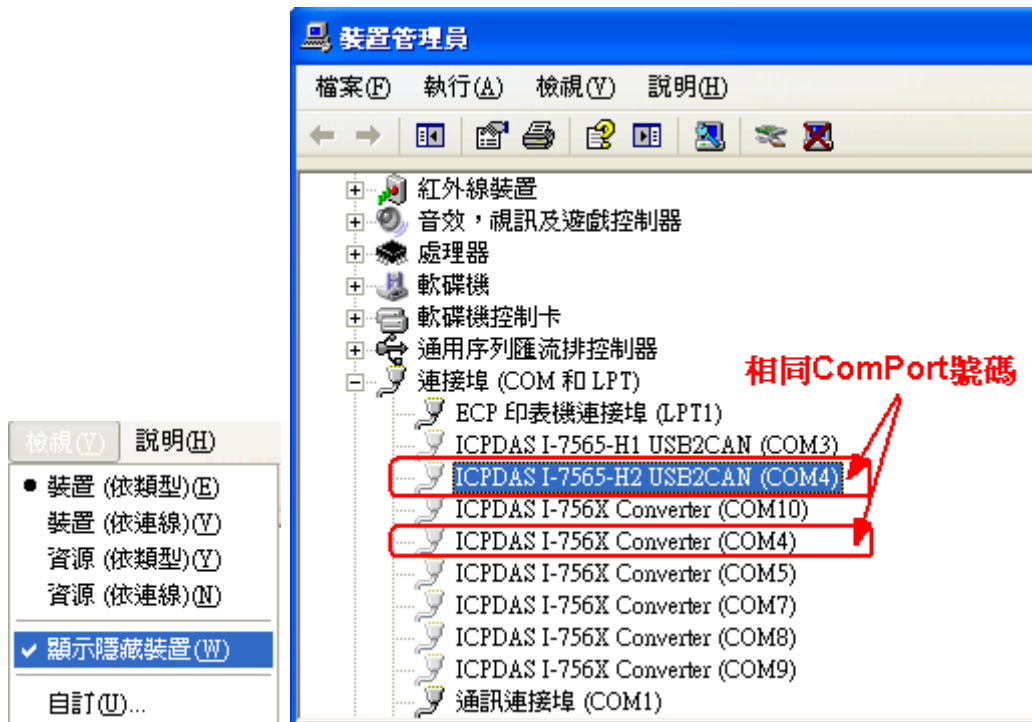


圖 7.1-1-1: 顯示隱藏裝置

若以上方法仍然無法連線時，請檢查 I-7565-H1/H2 是否完成驅動程式安裝或是虛擬 COM 埠的選擇是否正確。

## Q2 : "The device is not open" 錯誤訊息 ?

當使用者打開虛擬 COM 埠時，顯示"The device is not open"，如圖 7.1-2 的錯誤訊息時，表示該 COM 埠已被其它的程式所佔用，例如：VxComm Utility。請在 VxComm Utility 中，"UnMap"相同的 COM 埠編號，並且點選"Restart Driver"功能，如圖 7.1-3 所示。完成上述動作之後，請重置 I-7565-H1/H2 模組，並重試連線至 I-7565-H1/H2。



圖 7.1-2: 該裝置啓用失敗

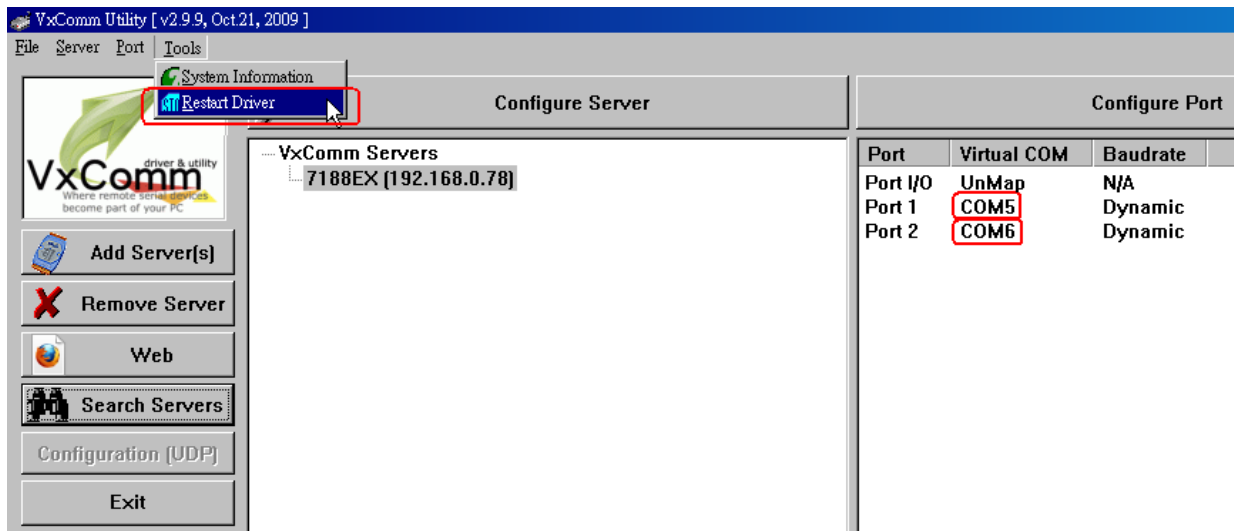


圖 7.1-3: VxComm Utility 的虛擬 COM 埠

### Q3 : "Device doesn't Exist" 錯誤訊息 ?

當使用者打開虛擬 COM 埠時，顯示"Device doesn't Exist !! Please Check Port No. !!"，如圖 7.1-4 的錯誤訊息時，表示該 COM 埠已被其它的程式所佔用。請在"裝置管理員"中，將 I-7565-H1/H2 模組之虛擬 ComPort 強制改成其它沒有使用的 ComPort 號碼後，並重新啓動電腦，再重新連線。

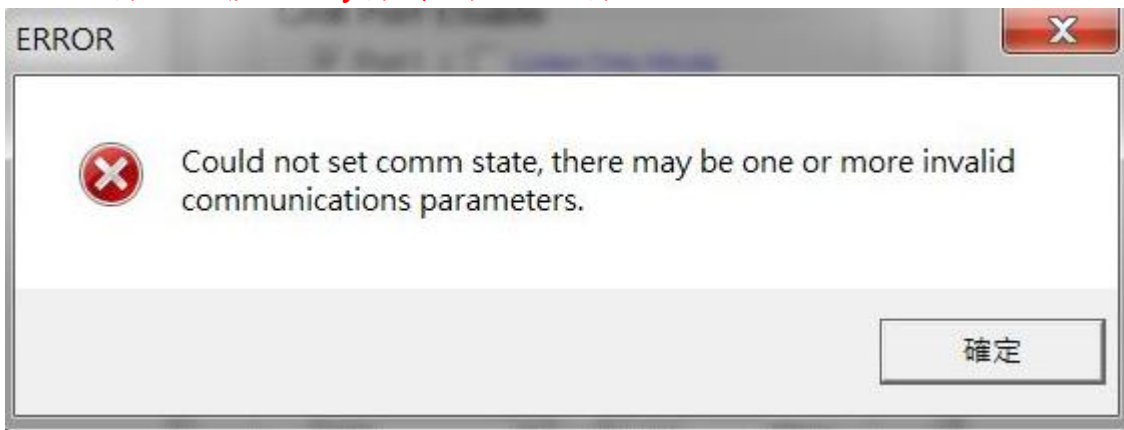


圖 7.1-4: Device doesn't Exist

### Q4 : "Could not set comm state" 錯誤訊息 ?

當使用者打開虛擬 COM 埠時，顯示"Could not set comm state"，如圖 7.1-5 所示，請執行"Extra Config"功能畫面之"Reset CANFID Flash"按鈕功能 (參考 4.4.3 節說明)，清除 CAN1/2 之 Filter-ID 所儲存之 Flash 空間，即可正常連線。**注意，此功能需配合 => 韌體版本為**

v1.06 版以上及 utility 版本為 v1.10 版以上。



(圖 7.1-5 “Could not set comm state”錯誤訊息)

## 7.2 CAN 鮑率問題

### (1) CAN 鮑率錯誤：

若 I-7565-H1/H2 模組所設定的鮑率與 CAN bus 網路上其它裝置設定的鮑率不同，並且在傳送第一筆 CAN 訊息後，I-7565-H1/H2 模組上的 RUN LED 將以每 100ms 頻率閃爍(這是因為鮑率設定不一致所導致的)，此時，使用者可以透過“I-7565-H1/H2 Utility”取得 I-7565-H1/H2 狀態，以協助使用者了解模組目前的情形。

### (2) 自行定義的 CAN 鮑率設定：

若使用者想以自行定義的鮑率進行通訊時，在“I-7565-H1/H2 Utility”的“Connect to I-7565-H1/H2”畫面，使用者可選擇“**Defined**”項目並且在“Baud Rate”右邊的欄位輸入自訂的 CAN 鮑率值(例如：83.333)，如圖 Figure 7.2-1，然後，點擊“Connect”鍵以連線至 I-7565-H1/H2。



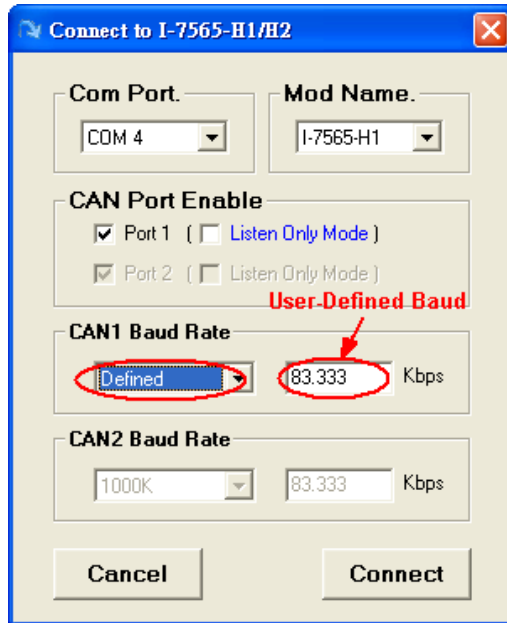


圖 7.2-1: 對 I-7565-H1/H2 自行定義 CAN 鮑率速率

(3) I-7565-H1/H2 自訂 CAN 鮑率的設定方法與 SJA1000 晶片相同：

若使用者使用 I-7565-H1/H2 模組與其它附有 SJA1000 晶片的 CAN 裝置並且使用自訂 CAN 鮑率進行通訊時，附有 SJA1000 晶片的 CAN 裝置，使用者需要選擇正確的 CAN 通訊參數(**BTR0 & BTR1**)，以利通訊：

- (1) “**Samples**”值為 1。
- (1) “**SJW**”值越小越好。(1 是最佳)
- (2) “**Tseg2**”值越小越好。(1 是最佳)
- (3) “**Tseg1**”值越大越好。

根據上述的四個規則，使用者可以選擇正確的 BTR0 與 BTR0 值。例如：若使用者想使用鮑率為 83.333 Kbps 時，此時，使用者應該選擇 BTR0=05、BTR1=1C。如圖 7.2-2 中，BTR0=05、BTR1=1C、TSEG1=13與 TSEG2=2 為此鮑率的最佳參數。

BTR0(hex)	BTR1(hex)	Samples	Spl%	TSEG1	TSEG2	BRP	SJW	Max. Bus(m)	Kbps	Osc. Tol(%)
0F	12	1	66	3	2	16	1	516	83.3333	.2809
0B	14	1	75	5	2	12	1	652	83.3333	.2101
07	18	1	83	9	2	8	1	788	83.3333	.1397
05	1C	1	87	13	2	6	1	856	83.3333	.1046
0B	23	1	62	4	3	12	1	516	83.3333	.211
4B	23	1	62	4	3	12	2	379	83.3333	.4219
07	27	1	75	8	3	8	1	697	83.3333	.1401
47	27	1	75	8	3	8	2	606	83.3333	.2801
05	20	1	91	13	2	6	1	700	83.3333	.1040

圖 7.2-2: SJA1000 自訂鮑率

---

### 7.3 CAN 網路中發生 CAN ID 重覆問題

當有兩個以上的 CAN 裝置發送相同的 CAN ID 時，會導致整個 CAN 網路通訊異常，相對的 I-7565-H1/H2 也會因為異常的 CAN 網路而無法發送或接收 CAN 訊息。若要讓 CAN 網路能回到正常狀態，使用者應該要先解決 CAN ID 重覆的問題，接著再重新啟動 I-7565-H1/H2。完成上述兩步驟後才能讓 I-7565-H1/H2 再度收送 CAN 訊息。

### 7.4 電腦自動重新開機問題

若使用者在使用 I-7565-H1/H2 模組一段時間後，電腦會自動重新開機時，請更新 Windows 系統的“Service Pack of Windows”至最新版本。例如：若使用者使用 Windows XP，請更新至 SP3 以解決這個問題。

### 7.5 最大資料傳輸率 (fps)問題

I-7565-H1/H2 最大資料傳輸率可達每秒 3000 資料訊框(frame)。但在一些處理速度較慢的電腦上，若以每秒 3000 筆速率接收資料訊框時，可能會造成遺失的問題。為此，我們在 I-7565-H1/H2 Utility 提供一可自行調整資料傳輸速率的功能(Advanced Config)，以解決高速傳輸模式下，造成資料訊框遺失的問題。值得注意的是，硬體資料傳輸率不可低於目前 CAN bus 的傳輸速率，否則資料遺失的問題將發生 I-7565-H1/H2 上。

### 7.6 資料遺失問題

有下列兩種情況會導致資料遺失的問題：

(1)由API函式庫提供的軟體接收緩衝區發生溢位：

它代表著使用者的程式不能再接收由軟體緩衝區的CAN封包。因此，使用應該調整最佳的通訊策略。

(2)硬體接收緩衝區溢位：

在接收封包的PC端，其中斷發生的延遲時間過長，造成硬體緩衝區溢位。這樣的問題解決方式是提升PC執行效能或是降低CAN上其它節點的傳輸速率。

## 7.7 一台電腦能插多少模組的問題

理論上，沒有這個限制。PC支援多個I-7565-H1/H2模組同步傳送作業，但其通訊效率端看PC硬體效能。

## 7.8 安裝驅動程式時間過久問題

若使用者按照第 3 章的步驟安裝 I-7565-H1/H2 的驅動程式，且安裝時間超過 2 分鐘以上時，請依照下述步驟解決問題：

- (1) 複製“I-7565-H1H2.inf”檔案至“C:\WINDOWS\inf”路徑下。
- (2) 複製“usbser.sys”檔案至“C:\WINDOWS\system32\drivers\”路徑。
- (3) 在完成上述兩個步驟後，請再依照第三章的步驟重新手動安裝 I-7565-H1/H2 的驅動程式，但操作方式與下述方式不同：如圖 7.8-1，請選擇“不要搜尋，我將選擇要安裝的驅動程式(D)”選項，並點擊“下一步”鍵。

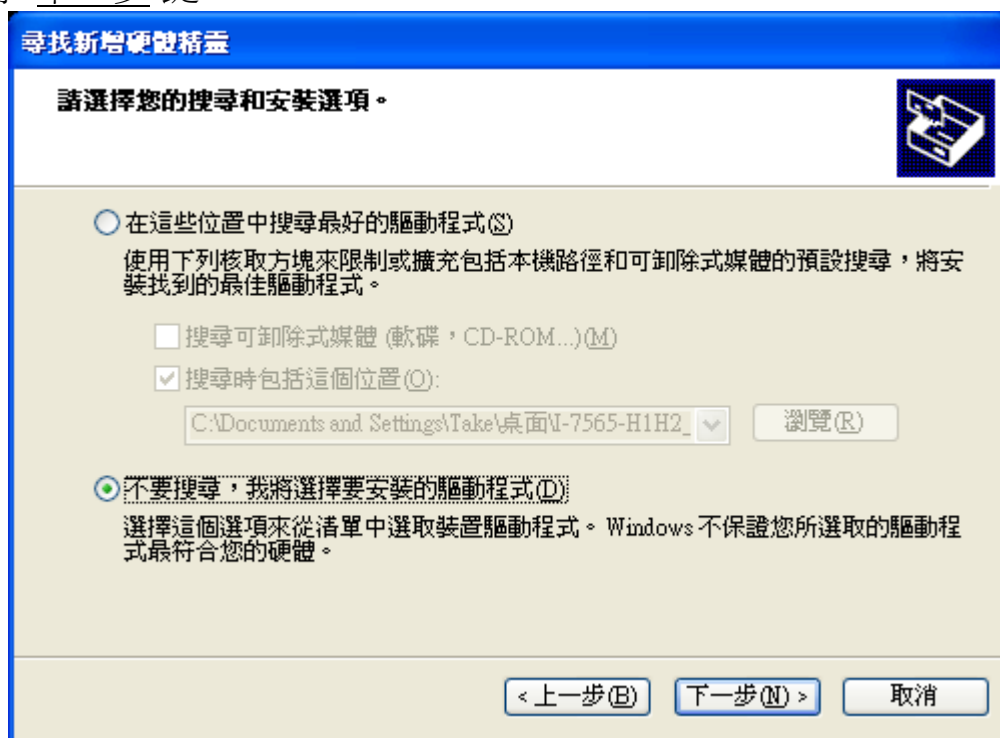


圖 7.8-1: I-7565-H1/H2 的驅動程式安裝(1)

- (4) 當畫面出現如同圖 7.8-2 時，點擊“下一步”鍵，其餘的步驟與第三章的說明步驟相同。

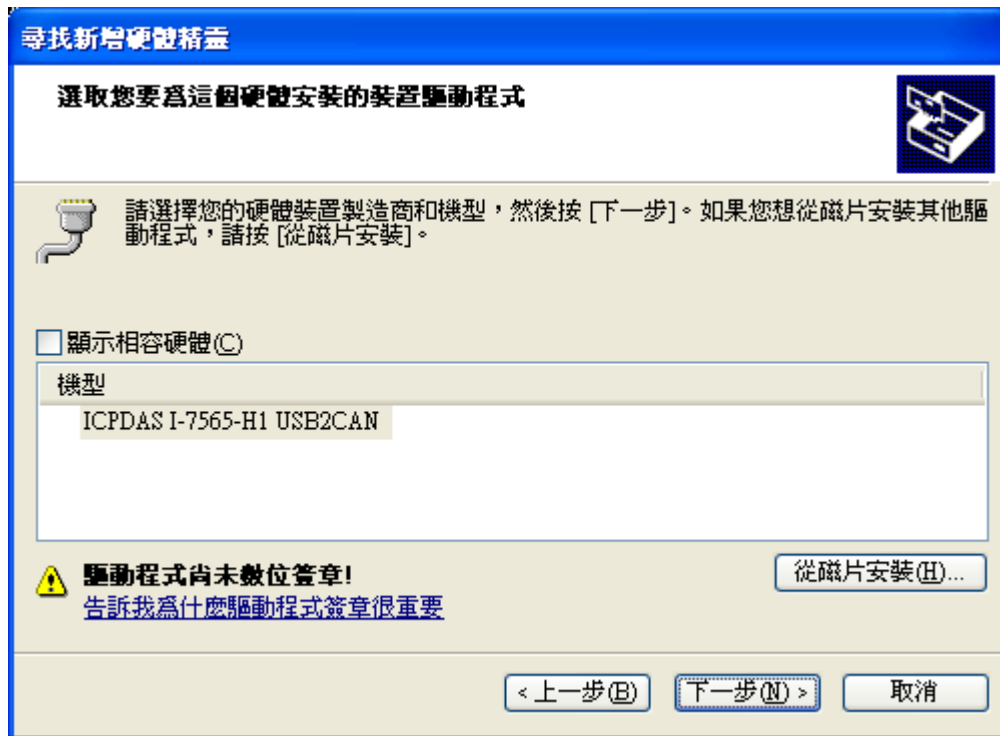


圖 7.8-2: I-7565-H1/H2 的驅動程式安裝(2)

## 7.9 支援的 CAN Filter-ID 編號問題

I-7565-H1/H2 的 CAN Filter-ID 總容量提供 440 WORD。下表說明每一個不同種類所佔用的大小。

	大小 (單位: WORD)
<b>11-bit Single ID</b>	1
<b>11-bit Group ID</b>	2
<b>29-bit Single ID</b>	2
<b>29-bit Group ID</b>	4

表 7.9-1: 不同種類的 CAN Filter-ID 所佔用大小

根據表 7.9-1，下列的表格說明 I-7565-H1/H2 支援的 CAN Filter-ID 規則數量。

	<b>I-7565-H1 ( CAN Port )</b>	<b>I-7565-H2 ( Each CAN Port )</b>
<b>11-bit Single ID</b>	440/1 = <b>440</b>	<b>220</b>

<b>11-bit Group ID</b>	440/2 = <b>220</b>	<b>110</b>
<b>29-bit Single ID</b>	440/2 = <b>220</b>	<b>110</b>
<b>29-bit Group ID</b>	440/4 = <b>110</b>	<b>55</b>

表 7.9-2: 每一個不同種類 CAN Filter-ID 規則數量

## 7.10 其它問題

基本上，接下來的錯誤種類也許會發生。例如，以 CAN 網路作為傳輸媒介時，其終端電阻未設置、與 CAN Bus 上的其它節點設定不同鮑率時...等等的設定。

## 7.11 Windows 7 相關問題

### 7.11.1 如何在 Windows 7 64-bit (x64) 中正確安裝 I-7565-H1/H2 之 Driver 及順利執行 I-7565-H1/H2 之 Utility ?

(1) 在 Windows 7 64-bit 之 OS 中，無法直接使用”自動安裝 driver”方式來執行，需採用手動方式來安裝，請依照以下步驟：

[1] 先執行”**ICPUsbConverter\_DrvInst\_v1.2.exe**” (在 v1.2 版以後之 driver 已支援”**驅動程式數位簽章**”認證)，將必要 driver 檔案安裝至 C:\WINDOWS\inf 路徑下。

[2] 將 I-7565-H1/H2 模組接至 PC，並採用手動方式來安裝 driver，請參考以下步驟：(或請參考 3.2 節之步驟)。

<1> 解壓縮”I-7565-H1H2\_DrvFile\_v1.4.rar”檔案

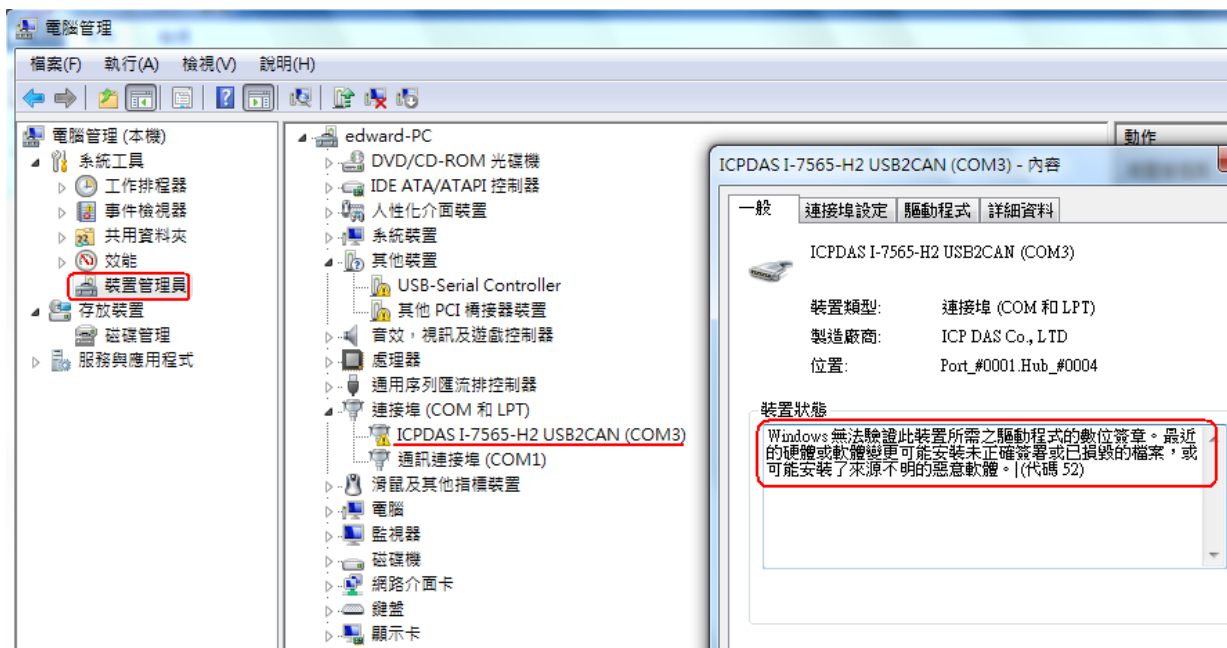
<2> 複製”I-7565-H1H2\_DrvFile\_v1.4”資料夾至桌面

<3> 採用手動方式安裝 I-7565-H1/H2 驅動程式

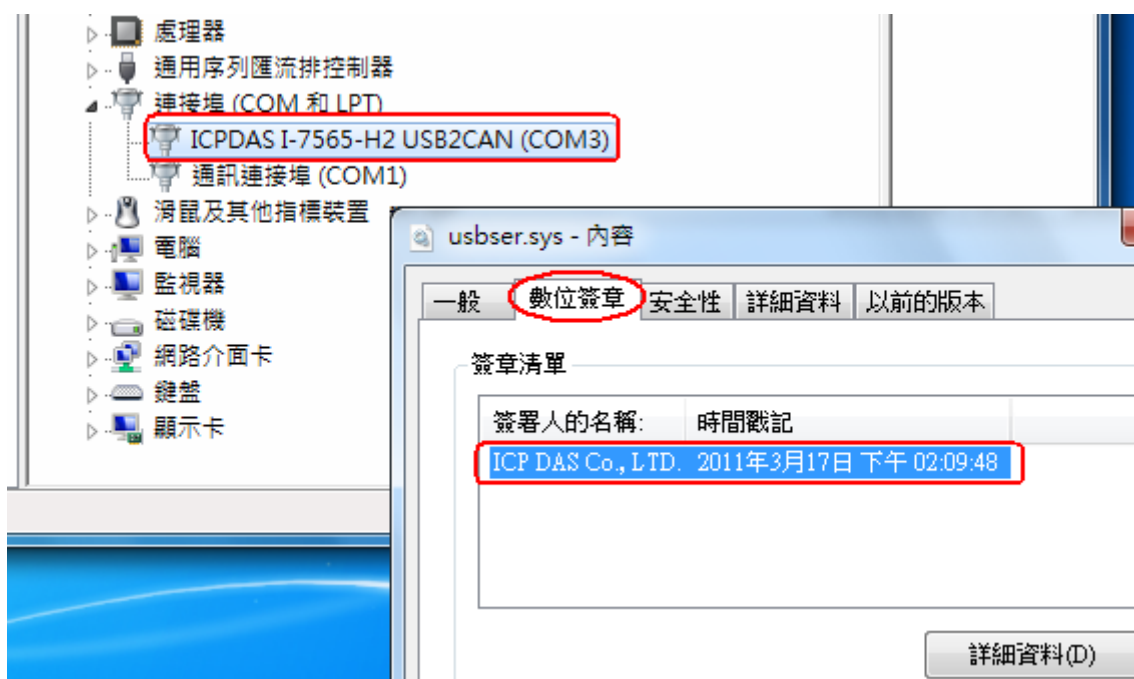
**(重要: 驅動程式檔案位置=>選擇”I-7565-H1H2\_DrvFile\_v1.4”資料夾)**

(2) 在 driver 順利安裝完成後，由於 driver v1.2 以後已支援”驅動程式數位簽章”認證，因此在”裝置管理員”中，I-7565-H1/H2 之 Virtual COM Driver 圖示上並不會多出一個”!” (錯誤代碼 52)，錯誤畫面如圖 7.11-1，若安裝 v1.2 以前之版本，請先解除安裝，再重新安裝最新版本之 Driver 即可正常顯示，如圖 7.11-2。

(3) 第一次執行 I-7565-H1/H2 Utility 時，記得要先使用”**系統管理員**”身份來執行，步驟如圖 7.11-3，否則會有”檔案元件未正確註冊”之錯誤訊息出現，如圖 7.11-4，在第一次執行成功後，之後再執行 Utility 時，直接點二下即可順利執行。



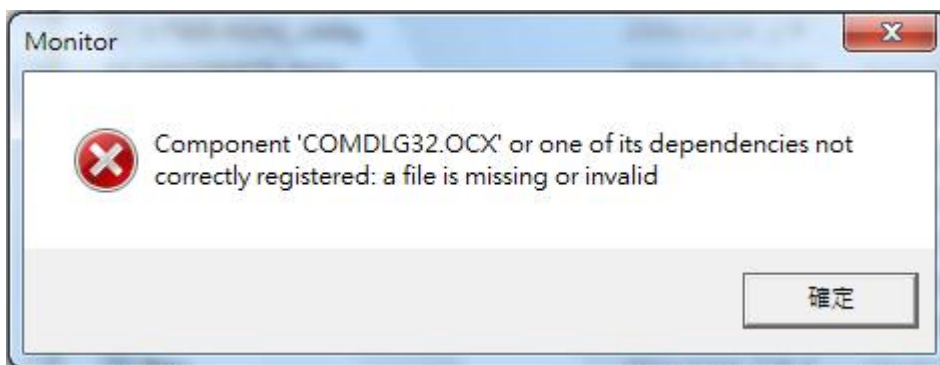
(圖 7.11-1 無驅動程式數位簽章認證之錯誤)



(圖 7.11-2 具驅動程式數位簽章認證)



(圖 7.11-3 使用”系統管理員”身份執行)



(圖 7.11-4 檔案元件未正確註冊之錯誤訊息)

## 7.12 I-7565-H1/H2 模組無法接收 CAN 訊息封包

1. 請檢查 I-7565-H1/H2 模組以下項目：
  - (1) CAN\_H 與 CAN\_L 腳位與設備是否接反。
  - (2) CAN bus 鮑率設定值與設備是否相同。
  - (3) I-7565-H1/H2 模組之 Filter-ID 功能是否有設定啓動。
  - (4) CAN\_H 與 CAN\_L 腳位之電阻值是否約為 60 歐姆。
  - (5) 週邊是否有強電磁波干擾源 (如: 馬達)。
2. 可調整 CAN 鮑率之 Bit-Timing (即 Tseg2 值 => 韌體 v1.07 支援)，選擇 Tseg2 較大值，再作通訊測試。(作法: 可參考 FAQ 7.14)
3. 可啓用 "CAN Error Frame" 功能 (韌體 v1.07 支援)，檢查 CAN bus 網路上是否有 Error Frame 產生。(作法: 可參考 FAQ 7.15)
4. I-7565-H1/H2 模組 CAN1/CAN2 之 CAN Filter-ID 功能是否已有設定。
5. 若以上方式均無法成功，請執行 Utility 之 "Configuraiton" -> "Extra

Config”功能畫面之”Reset CANFID Flash”按鈕功能 (參考 4.4.3 節說明)，清除 CAN1/2 之 Filter-ID 所儲存之 Flash 空間。(注意，此功能需配合 => 韌體版本為 v1.06 版以上及 utility 版本為 v1.10 版以上)

### 7.13 I-7565-H1/H2 有提供 LabVIEW Driver

使用者可至 ICP DAS 網站下載 LabVIEW 8.x 函式庫 及範例，如下：  
[ftp://ftp.icpdas.com/pub/cd/fieldbus\\_cd/can/converter/i-7565-h1h2/software/library/win2k\\_xp/](ftp://ftp.icpdas.com/pub/cd/fieldbus_cd/can/converter/i-7565-h1h2/software/library/win2k_xp/)

### 7.14 如何調整 I-7565-H1/H2 之鮑率 Bit-Timing 參數值？

在 I-7565-H1/H2 之韌體 v1.07 及工具軟體 v1.13 版以上，已有提供以下功能：

- (1) 可調整 CAN 通訊鮑率之 Bit-Timing (即 Tseg2 值)，如圖 14-1 所示。
- (2) 顯示 I-7565-H1/H2 內部目前之 Bit-Timing 設定值，如圖 14-2 所示。

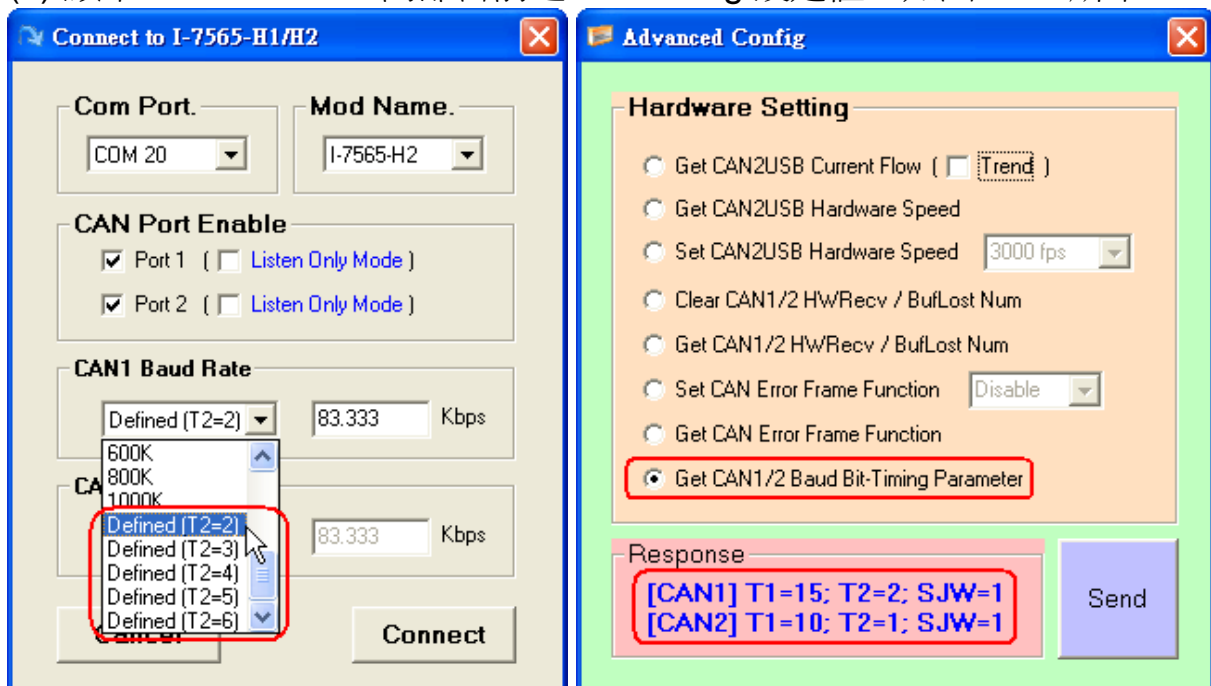


圖 7-14-1 T2 值調整

圖 7-14-2 取得 T1, T2 及 SJW 設定值

### 7.15 如何啓動 I-7565-H1/H2 之 CAN 錯誤封包訊息顯示功能？

在 I-7565-H1/H2 之韌體 v1.07 及工具軟體 v1.12 版以上，已有提供 CAN 接收錯誤封包之訊息顯示功能，參考以下步驟：

- (1) 執行 I-7565-H1/H2 之 Utility，並連線至 I-7565-H1/H2 模組。
- (2) 至”Advanced Config”功能畫面，選擇”Set CAN Error Frame Function”



選項為”Enable”，再按下”Send”鈕，設定至模組，如圖 7-15-1。

- (3) 若 CAN 網路有錯誤發生，會在 CAN RecvMsg 欄位顯示 CAN 錯誤訊息封包，可點選所要查看之錯誤行數，開啓詳細錯誤資訊，如圖 7-15-2。

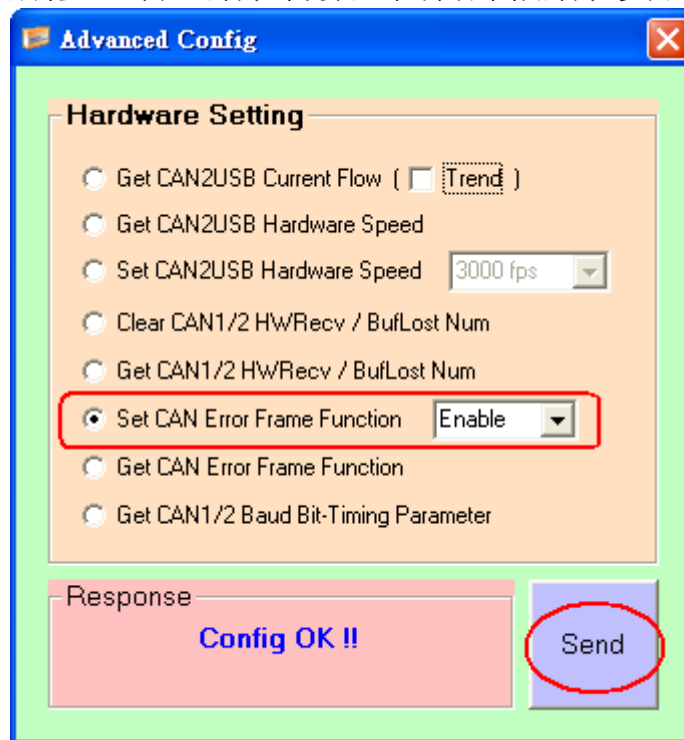


圖 7-15-1 啓動 CAN 錯誤封包之訊息顯示功能

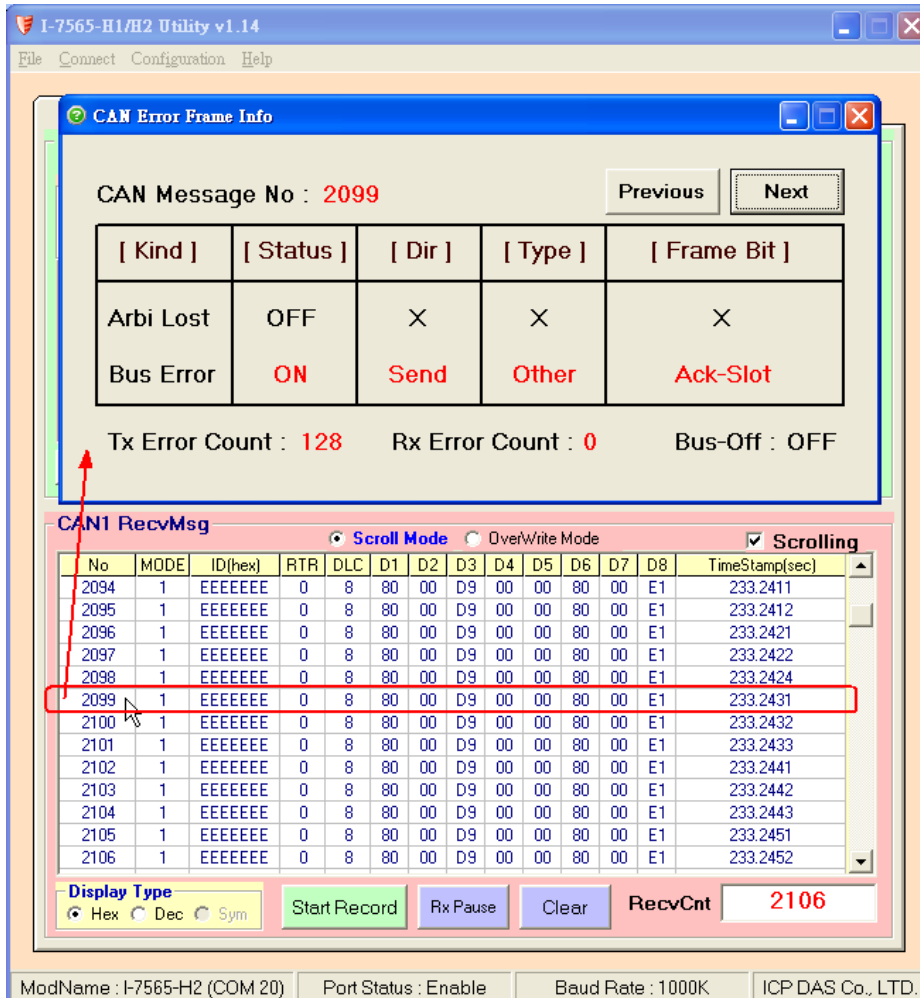


圖 7-15-2 CAN 錯誤封包詳細資訊

## 7.16 新功能 - “OverWrite”, 工具軟體 v1.09 版以上支援？

在工具軟體 v1.09 版以上，在 CAN 接收訊息欄位中有提供 “OverWrite” 功能選項，主要可將接收到的 CAN 訊息，當其 MODE 及 ID 內容均相同時，則會被放在同一列之欄位中，其中“Num”欄位會顯示接收到之相同 MODE 及 ID 的 CAN message 數量，而在“CycleTime”欄位會顯示此種 CAN message 之接收週期 (單位:秒)。

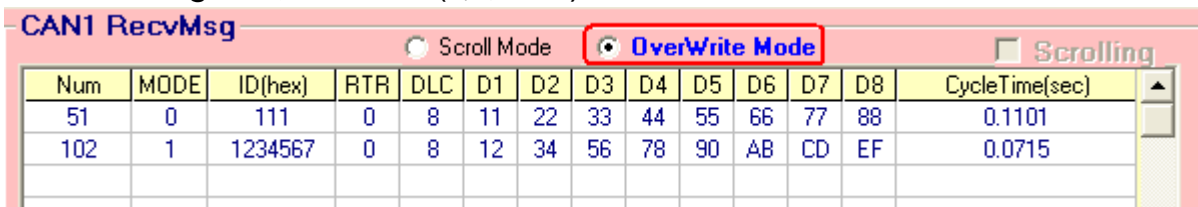


圖 7-16-1 “OverWrite” 接收功能畫面

## 7.17 新功能 - “Symbolic”, 工具軟體 v1.10 版以上支援？

在工具軟體 v1.10 版以上，在 CAN 接收訊息欄位之”Display Type”選項中有提供”Sym”功能 (目前僅支援在”OverWrite”顯示模式下-參考 7.16 說明)，主要可將接收到的 CAN 訊息，將其 ID 數值內容轉換顯示為指定文字內容。(需先執行”Load Symbol File”功能)

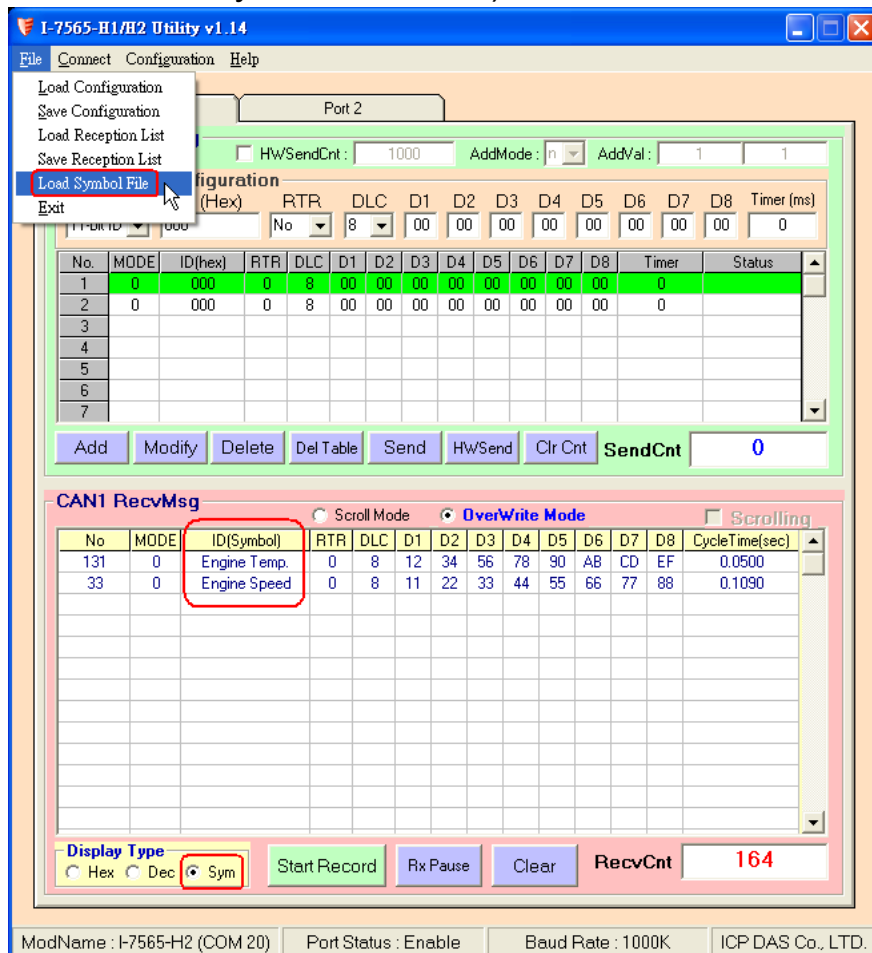


圖 7-17-1 “Sym” 接收功能畫面

## 7.18 如何使用 I-7565-H2 精確地傳送 CAN 訊息？

### (1) 透過 I-7565-H1/H2 軟體工具:

先加入所要傳送之 CAN 訊息，再按下”HWSend”鈕，即可開始持續傳送，若要傳送指定數量，則可勾選”HWSendCnt”選項，並輸入所要傳送之數量。

### (2) 透過 I-7565-H1/H2 之 API 函式庫:

在 I-7565-H1/H2 所提供 API 函式庫中，可透過使用

**VCI\_EnableHWCyclicTxMsgNo()**函式 (v1.08 以上支援) 來達成。

## 7.19 如何監聽 CAN 網路封包訊息，而不影響原本 CAN 網路通訊？

請使用“Listen Only”功能來達成。

### (1) 透過 I-7565-H1/H2 軟體工具:

在連線畫面中，先勾選“Listen Only Mode”選項 (如圖 7-19-1)，再按下“Connect”鈕進行連線，連線成功後，即會在 SendMsg 欄位內改為顯示“Listen Only Mode”訊息 (如圖 7-19-2)。

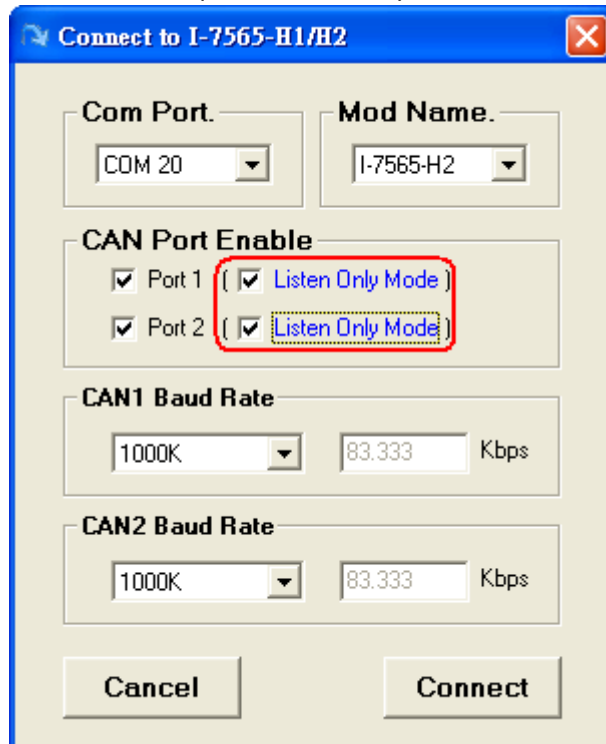


圖 7-19-1 “Listen Only Mode”選項畫面

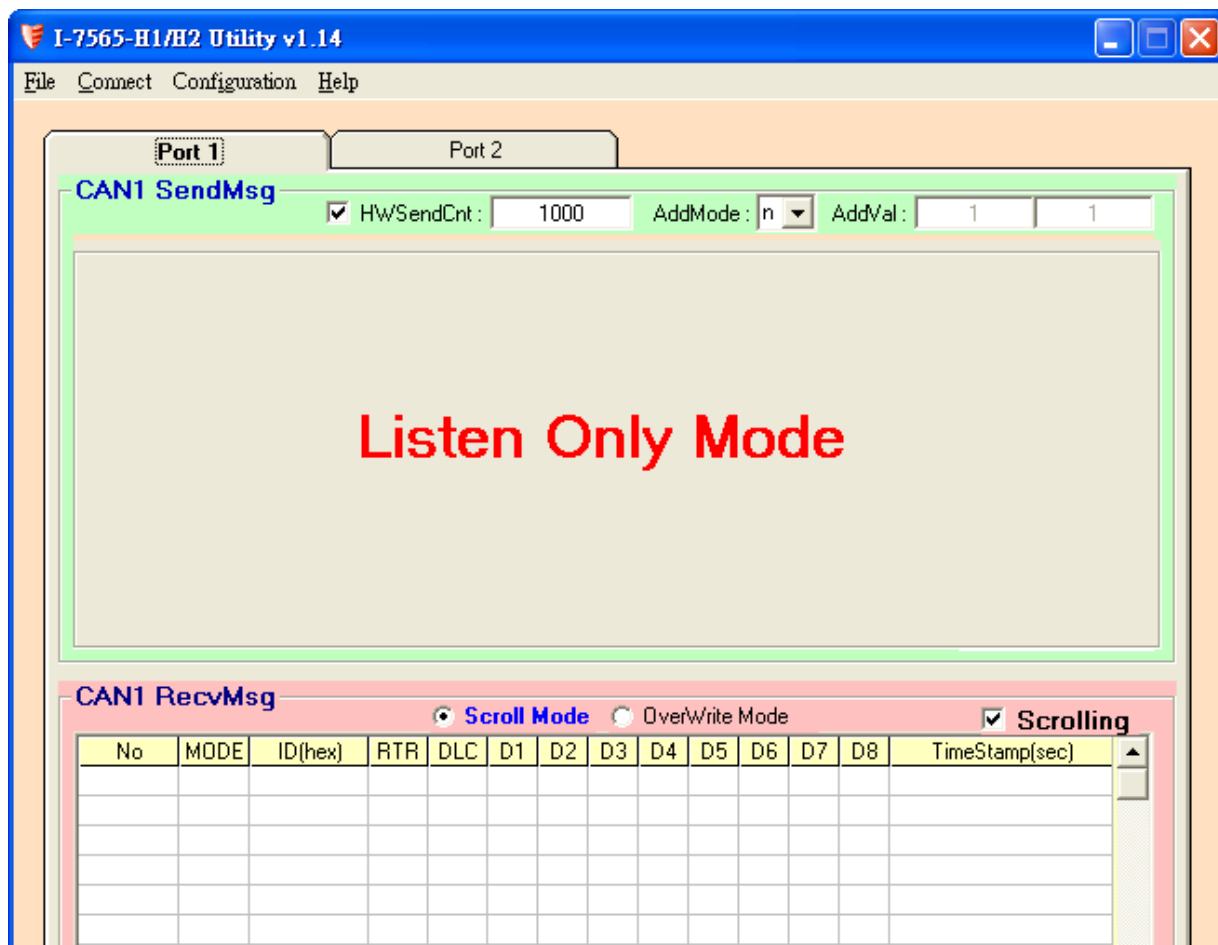


圖 7-19-2 "Listen Only Mode" 操作畫面

(2) 透過I-7565-H1/H2之API函式庫:

在I-7565-H1/H2所提供API函式庫中，可透過使用VCI\_Set\_MOD\_Ex()函式 (v1.10以上支援) 來達成。

## 7.20 如何取得目前 CAN 網路之封包流量？

在 I-7565-H1/H2 軟體工具之"Advanced Config"功能畫面中，有提供取得目前 CAN 網路通訊流量之功能選項 – [Get CAN2USB Current Flow](#)，如圖 7-20-1，勾選後，按下"Send"鈕即可開啓 CAN 網路通訊即時流量圖，如圖 7-20-2。

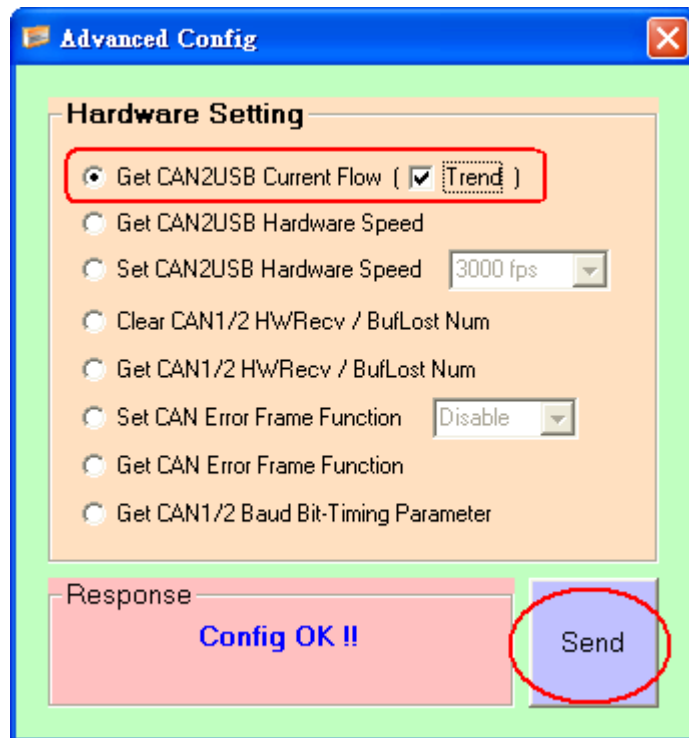


圖 7-20-1 "Get CAN2USB Current Flow" 選項畫面

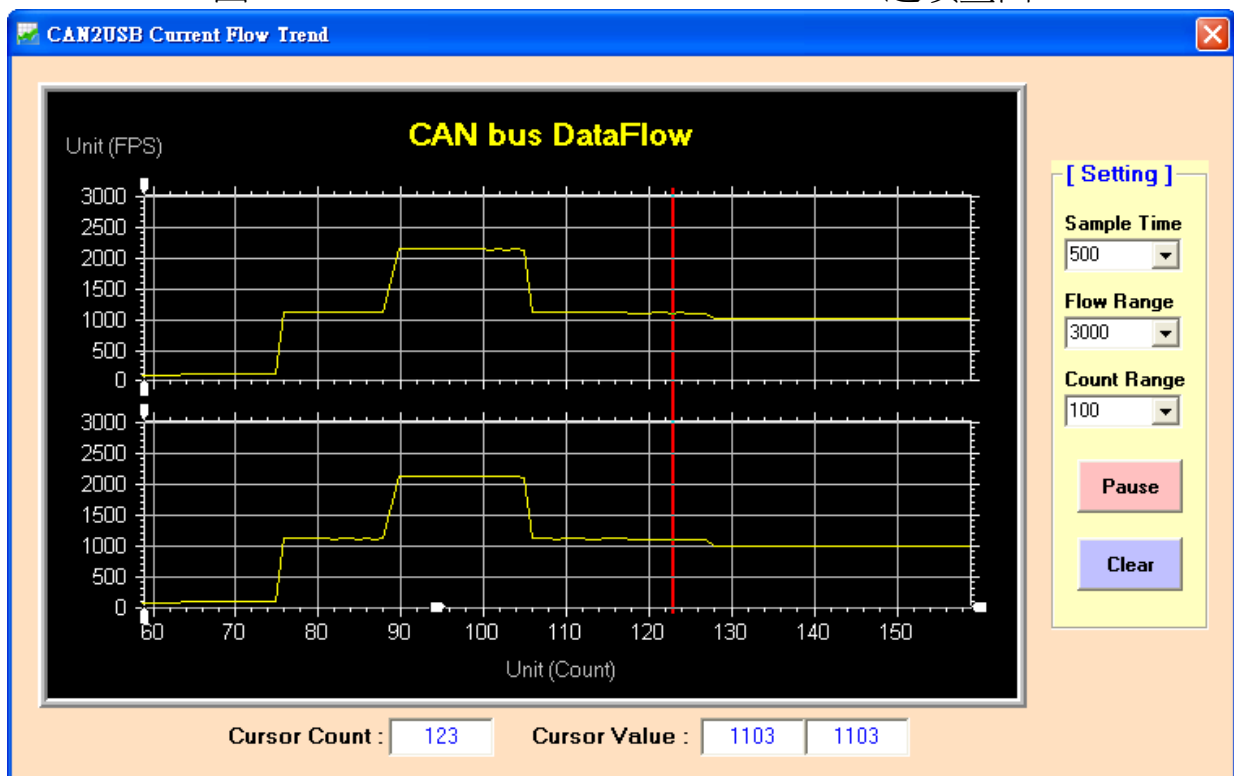


圖 7-20-2 "CAN bus DataFlow" 畫面

## 7.21 如何讓 I-7565-H1/H2 成爲 CAN 資料記錄器？

在 I-7565-H1/H2 軟體工具之“RecvMsg”的 CAN 資料接收區中，可透過使用“Start Record”鈕功能來達成，如圖 7-21-1。當點選“Start Record”鈕

時，使用者可選擇此次所要儲存至記錄檔之筆數(0:表示為無限制)，接著將會自動產生一個 CAN 資料記錄檔來儲存所接收到之所有 CAN 訊息，並以目前 PC 之日期及時間作為檔名，如 CAN1\_20130102\_100339.txt，此記錄檔會儲存在與 I-7565-H1/H2 軟體工具之同一資料夾內。

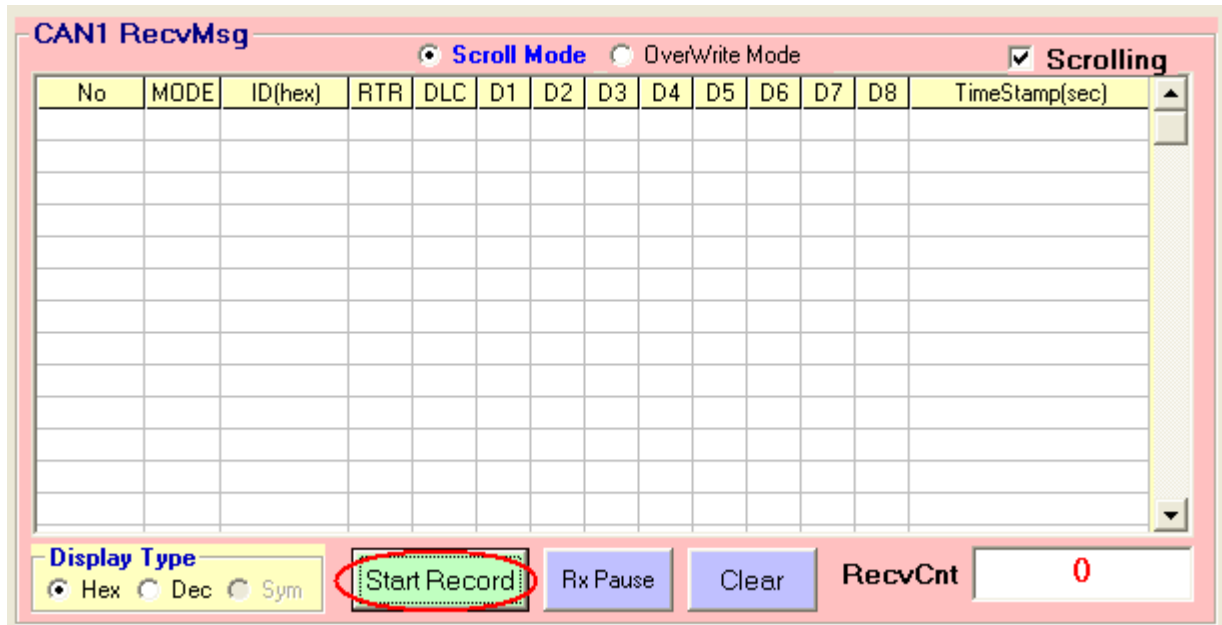


圖 7-21-1 "Start Record"鈕

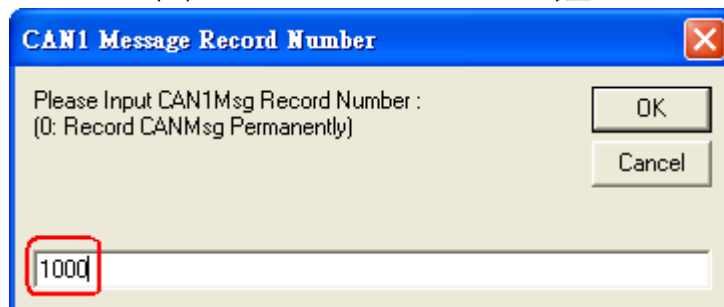


圖 7-21-2 設定 CAN 訊息之記錄筆數

## 7.22 如何立即接收到指定之 CAN-ID 訊息資料？

在 I-7565-H1/H2 函式庫中，可透過使用“VCI\_Set\_UserDefISR”函式來達成。

例如：使用者希望立即收到 CAN1 訊息之 Mode=11bit, ID=0x100 的資料，請參考以下範例碼：

- (1) 執行 VCI\_OpenCAN() 函式，來開啓 I-7565-H1/H2 模組之 CAN 通道。
- (2) 執行 VCI\_Set\_UserDefISR(1, CAN1, MODE\_11BIT, 0x100, MyTestISR1) 函式。
- (3) 當使用者程式收到指定之 CAN 訊息時，即會立刻執行 MyTestISR1 函式一次，因此可在 MyTestISR1 函式中，使用 VCI\_Get\_ISRData 函式來取得指定 CAN 訊息之資料。

## [ 注意 ]

(1)在 MyTestISR1 函式中不能花費太長時間，否則可能造成指定 CAN 訊息遺失之情形。

## 7.23 API 函式庫是否支援 Visual Studio Express 免費開發軟體？

I-7565-H1/H2 之 .Net API 函式庫可支援 Visual Studio Express 免費開發軟體，使用方式和 I-7565-H1/H2 所提供之 Visual Studio .Net 版本 Demo 之函式用法完全相同。

## 7.24 DotNet 範例在 Win7\_x64 平台執行時，會出現"試圖載入格式

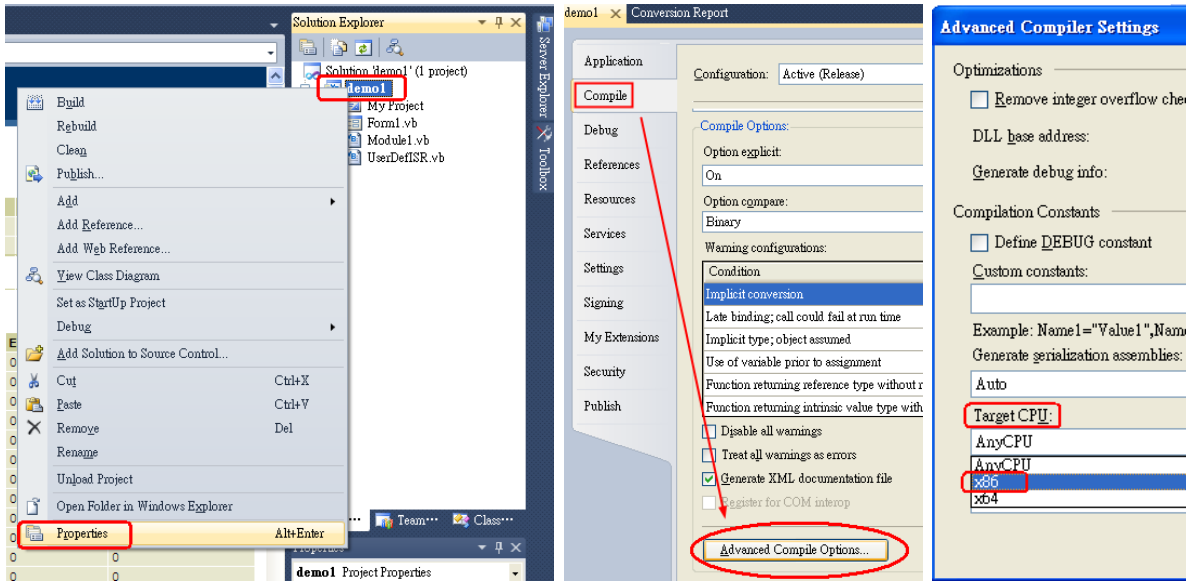
錯誤 0x8007000B"或"System.NullReferenceException"錯誤訊息？

1. 錯誤訊息如下。



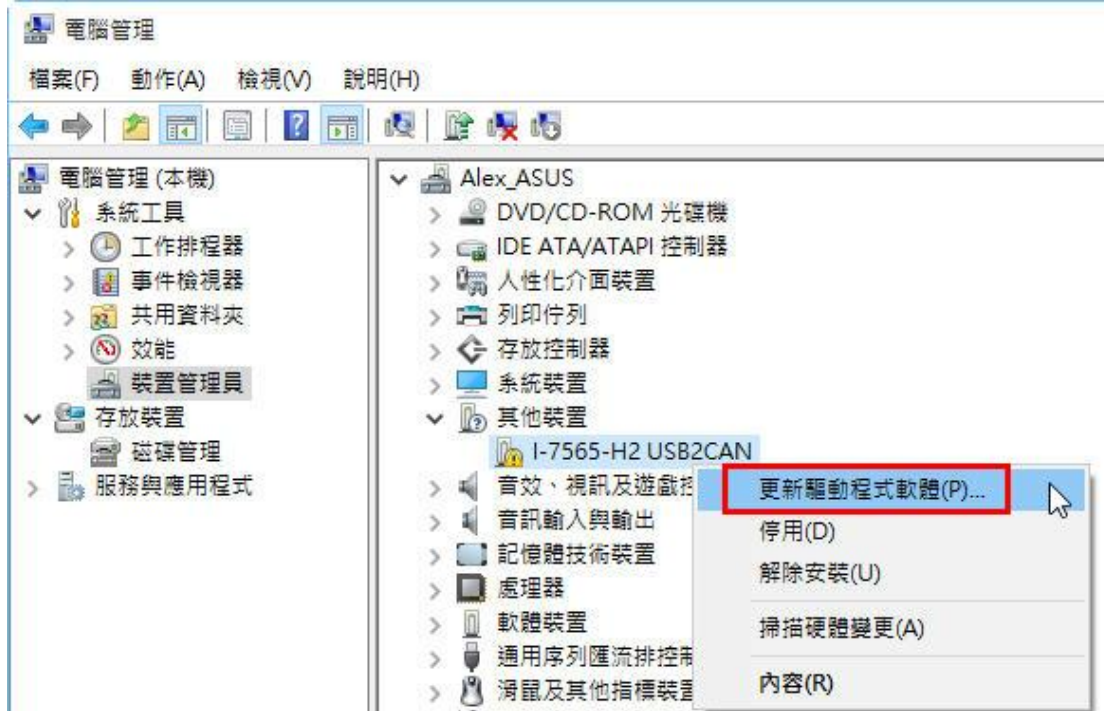
2. 解法: 將 DotNet Demo 專案之 CPU 選項，由"AnyCPU"改為"x86"，並重新編譯即可正常執行。

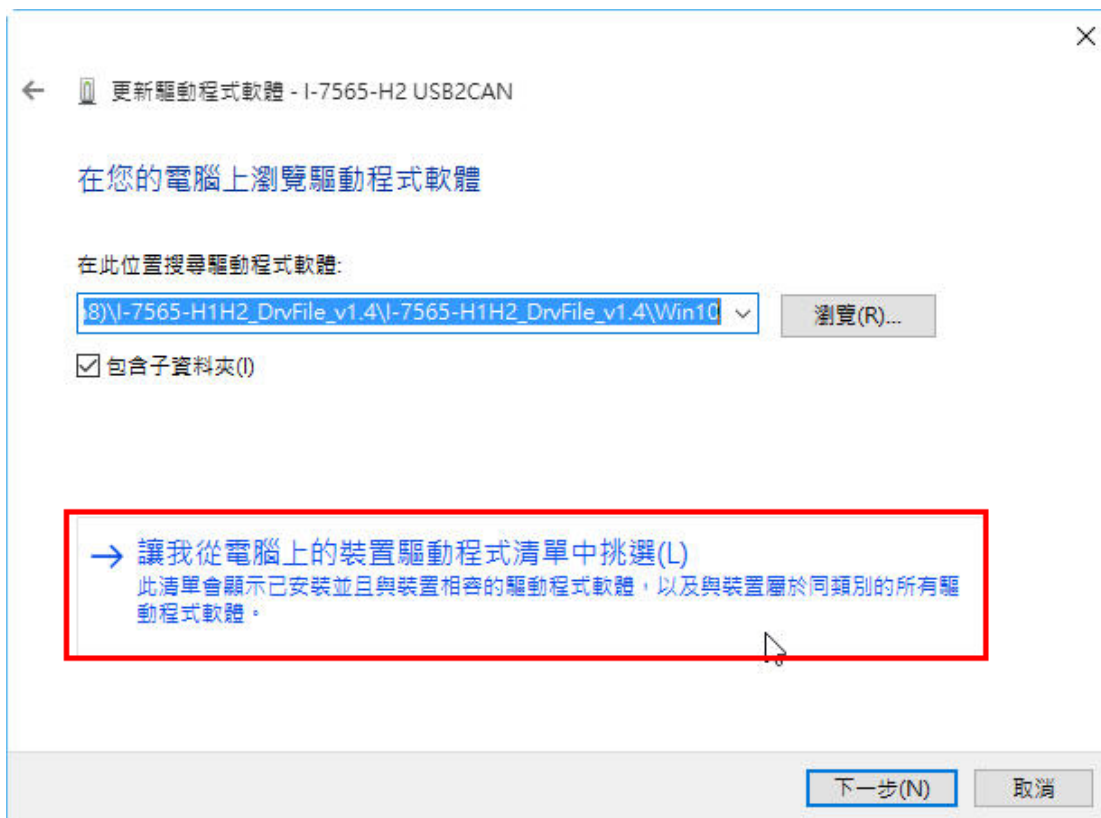
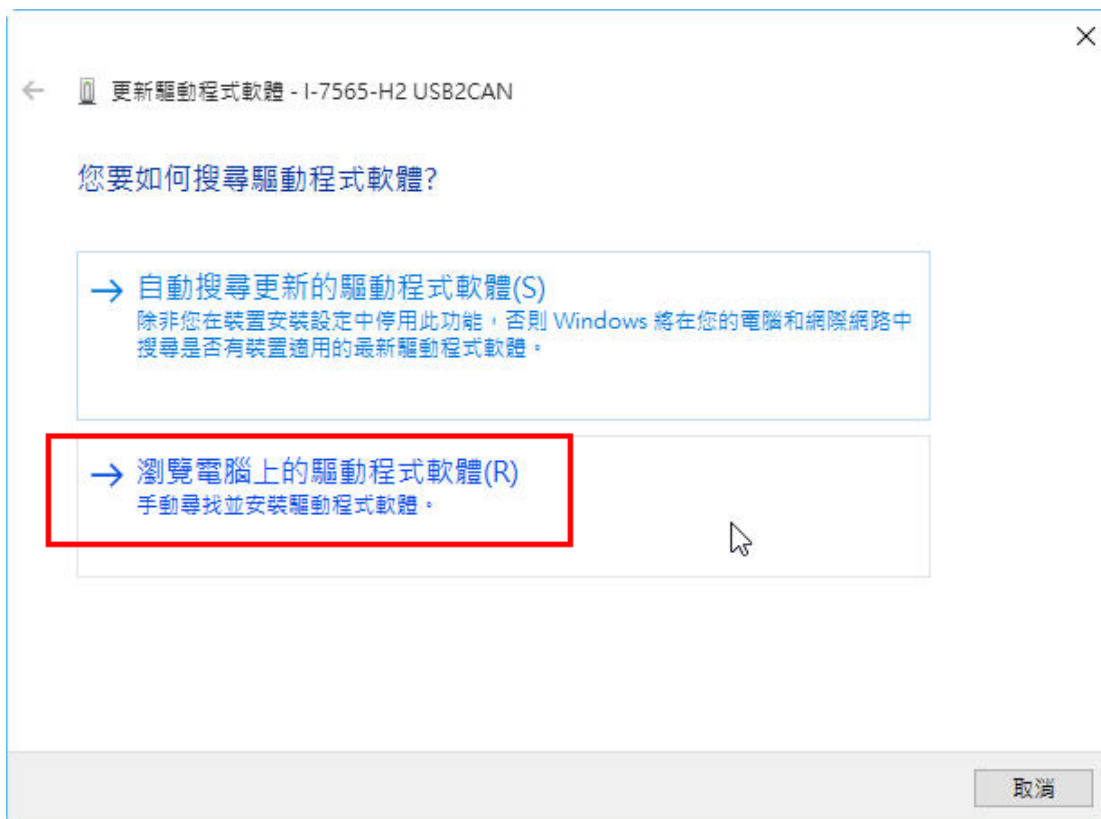




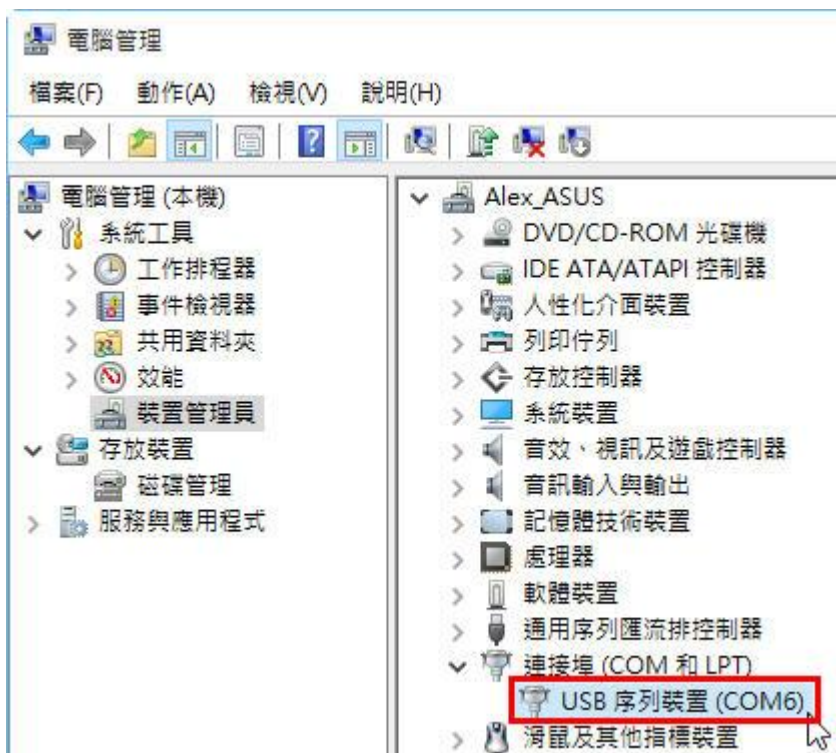
## 7.25 Windows 10 相關問題

### 7.25.1 如何在 Windows 10 中正確安裝 I-7565-H1/H2 之 Driver ?







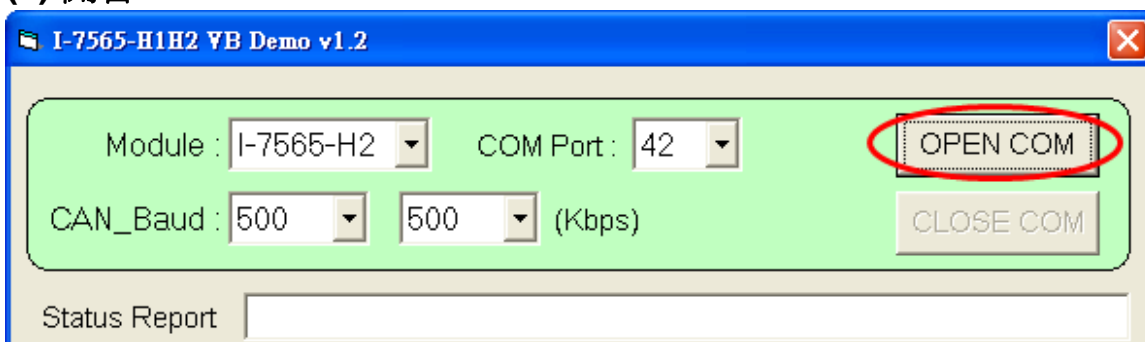


### 7.25.2 如何在 Windows 10 中使用 I-7565-H1/H2 之 Utility ?

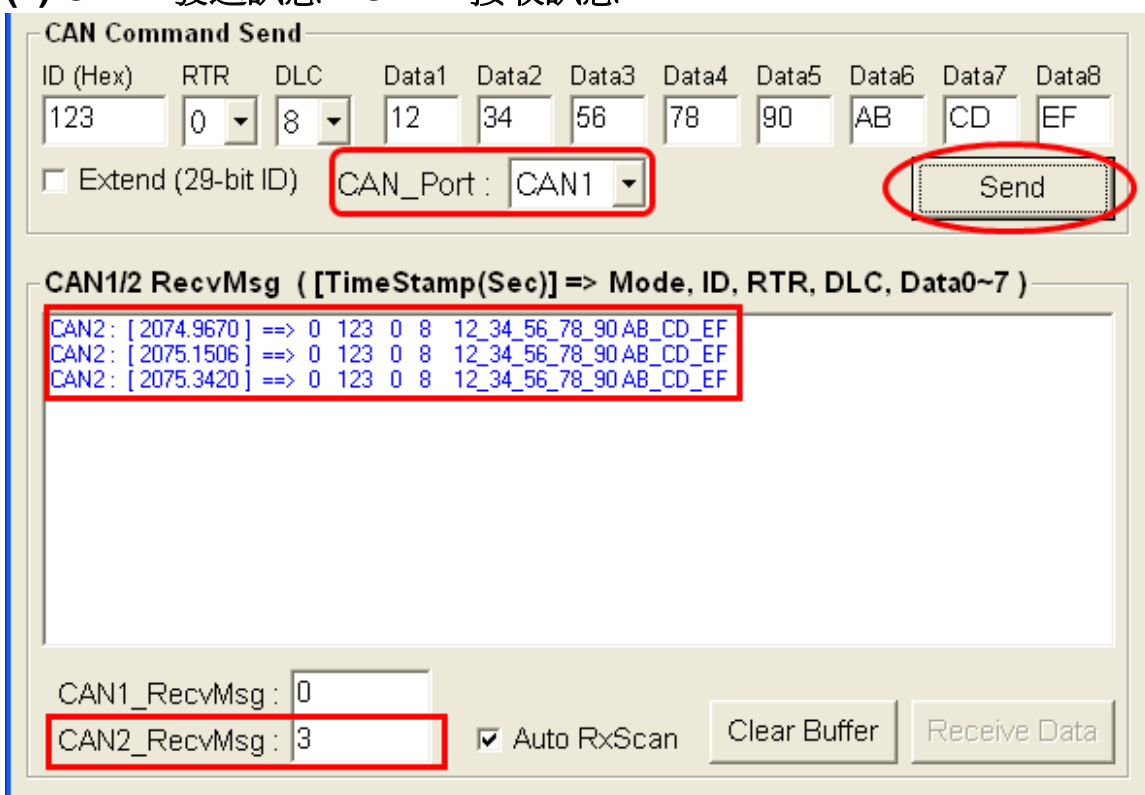
1. 在 Windows 10 平台，請執行"I-7565-H1H2\_Utility\_Win10.exe"。

2. 亦可使用 **VC\_Demo1** 或 **VB\_Demo1** 來測試 CAN 訊息之發送及接收。  
(採用 I-7565-H2 模組，並將 CAN1 及 CAN2 對接作測試)

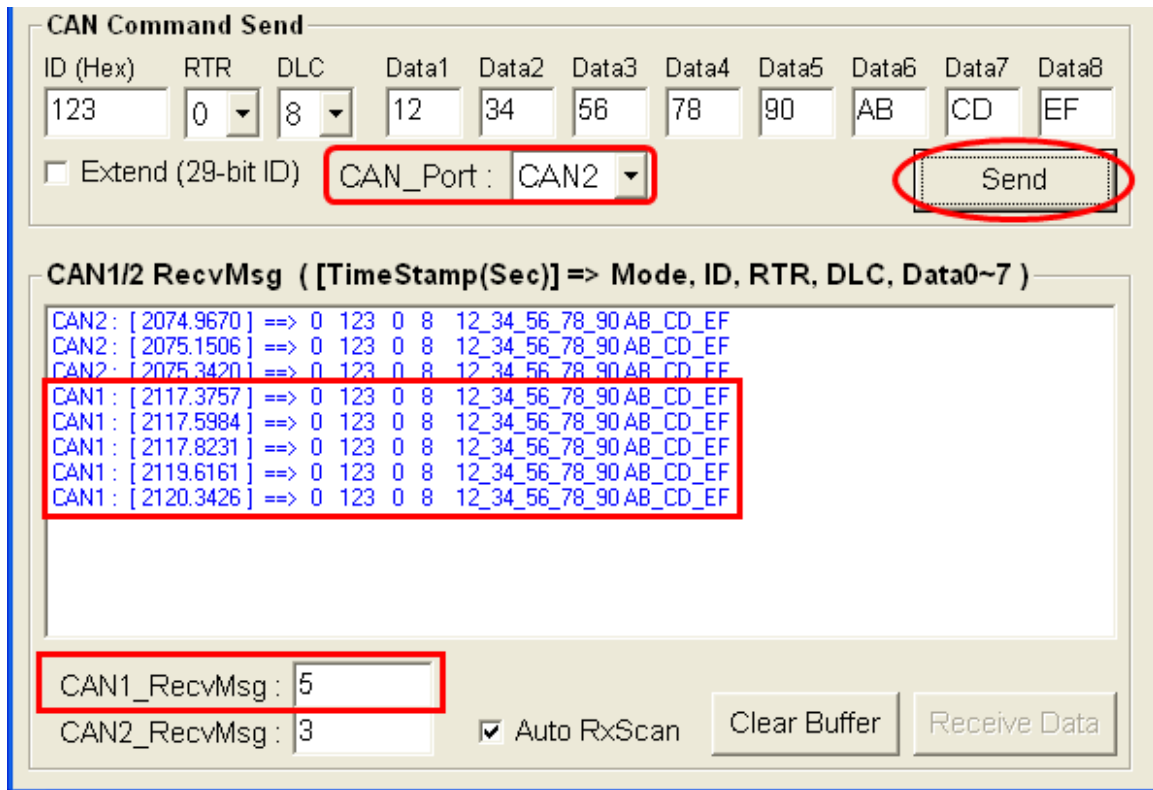
**(1) 開啓 ComPort :**



**(2) CAN1 發送訊息，CAN2 接收訊息:**



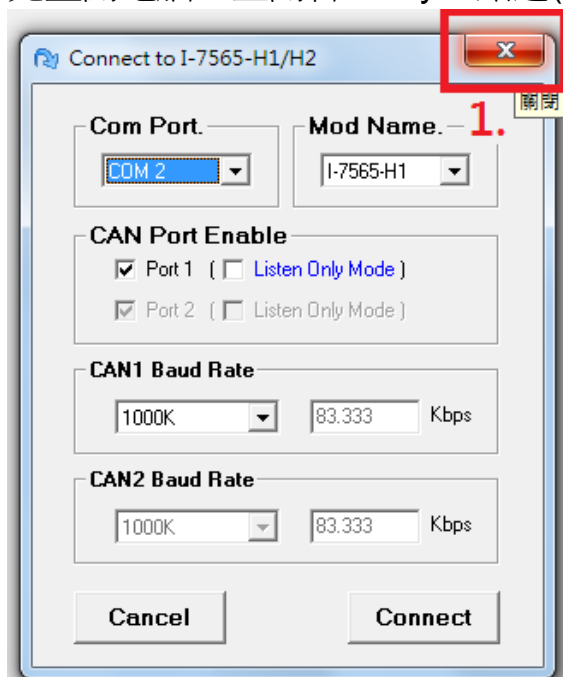
**(3) CAN2 發送訊息，CAN1 接收訊息:**



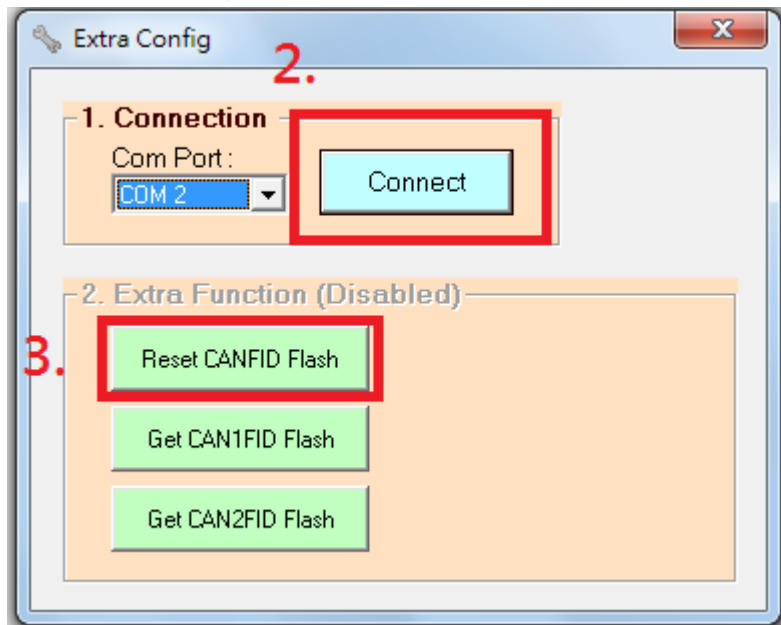
## 7.26 使用 VCI\_Set\_CANFID 函式，當寫設定參數錯誤時，可能造成模組無法正常開啓 COM?

A26: (2017/12/20)

1. 先重開電腦，並開啓 utility，點選(x)關閉連線畫面。



2. 點擊 Configuration -> Extra Config 選項。
3. 選擇 Com Port，並按下 Connect 鈕。
4. 點擊 Reset CANID Flash 鈕，來清除 I-7565-H1/H2 模組內部之 CAN Filter-ID 設定參數。

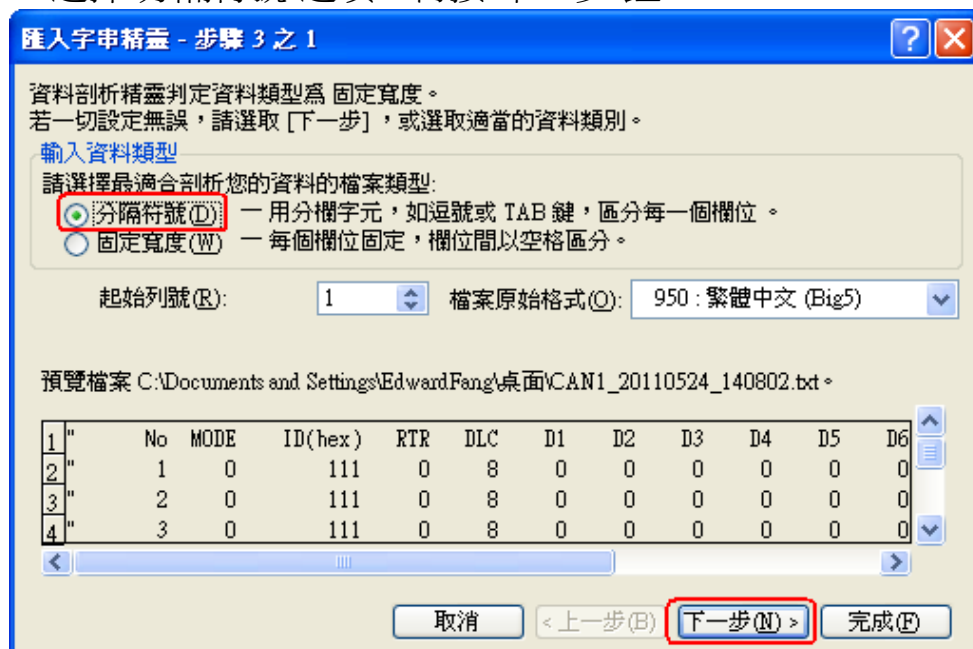


5. 至連線畫面，即可與 I-7565-H1/H2 模組正常連線。

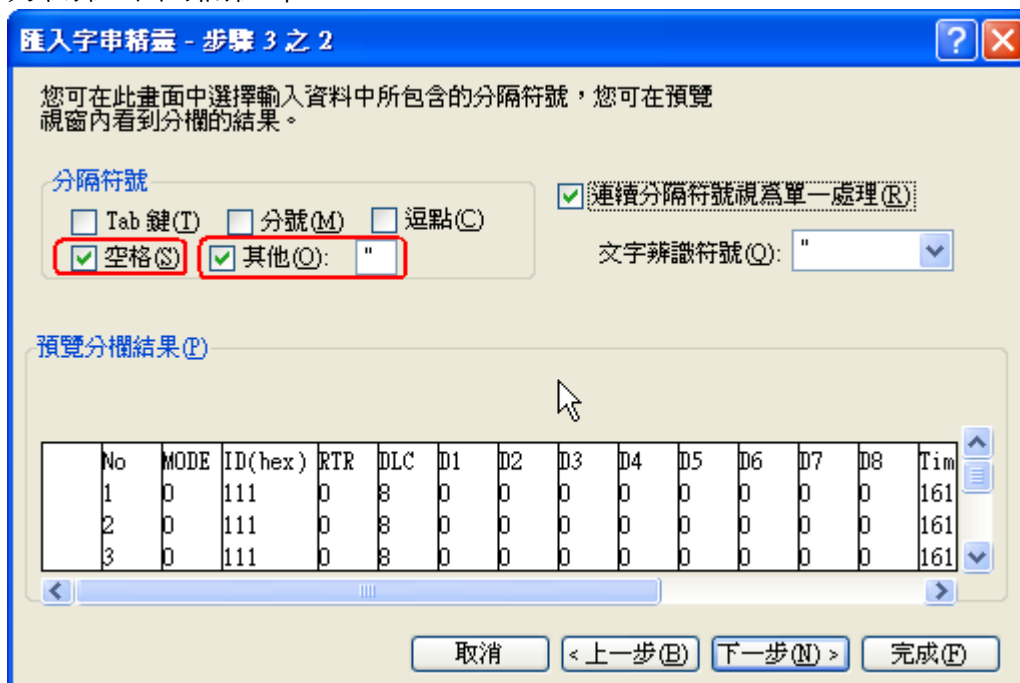
## 7.27 CAN 通訊記錄檔，如何轉成 Excel 檔開啓?

A27: (2017/12/20)

1. 先開啓 Excel 程式，直接選擇 TXT 記錄檔(\*.txt)，並開啓。
2. 選擇”分隔符號”選項，再按”下一步”鈕。



3. 勾選“空格”及“其它”(在右方空格,輸入”),再按入”完成”鈕,即可將所有資料分開在不同欄位中。



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1		No	MODE	ID(hex)	RTR	DLC	D1	D2	D3	D4	D5	D6	D7	D8	TimeStamp(sec)	
2		1	0	111	0	8	0	0	0	0	0	0	0	0	1612.7703	
3		2	0	111	0	8	0	0	0	0	0	0	0	0	1612.9509	
4		3	0	111	0	8	0	0	0	0	0	0	0	0	1613.1157	
5		4	0	111	0	8	0	0	0	0	0	0	0	0	1613.5251	
6		5	0	111	0	8	0	0	0	0	0	0	0	0	1613.7434	
7																
8																



---

## 8. Linux 平台使用手冊

可至 [ftp://ftp.icpdas.com/pub/cd/fieldbus\\_cd/can/converter/i-7565-h1h2/manual/linux/](ftp://ftp.icpdas.com/pub/cd/fieldbus_cd/can/converter/i-7565-h1h2/manual/linux/)，下載 Linux 平台使用手冊。

## 9. 版本歷史

版本編號	作者	日期	更動說明
1.0	Wayne	2010/03/01	1. 中文初版發佈
1.2	Wayne	2010/04/07	1. Utility 更新至 v1.04 2. 新增自動安裝驅動程式功能 3. 提供無結構型態之 API 函式庫(VCI_CAN Lib v1.04)
1.3	Edward	2010/11/29	1. 新增同時控制多個模組之 API 函式庫 – “mVCI_CAN” v1.00 2. Utility 更新至 v1.07 3. VCI_CAN 函式庫更新至 v1.06 4. VCI_UART 函式庫更新至 v1.01
1.4	Edward	2010/12/08	1. 新增”mVCI_CAN_vb.dll”來支援 VB 多個模組同時控制之功能
1.5	Edward	2011/03/17	1. 新增”使用者自訂義函式”功能 2. 新增”硬體唯一序號”功能 (VCI_CAN 函式庫更新至 v1.07) 3. Driver 版本更新至 v1.2 (加入驅動程式數位簽章認證)，且名稱改爲 ICPUsbConverter_DrvInst (整合 I-7567 模組) 4. Utility 版本更新至 v1.08 5. VCI_CAN 函式庫 v1.073 版加入 VCI_Get_ISRCANData 函式
1.6	Edward	2011/05/25	新功能需搭配以下軟體版本： <b>FW: v1.05 / Utility: v1.09 / APILib: v1.08</b> 1. 提供”Listen Only Mode”功能 2. HWSendTimer 數量由 1 組增加至 5 組 3. 加入”AddMode”及”AddVal”選項至 HWSendTimer 功能中 4. Utility 提供 CAN bus Flow Trend 畫面功能 5. Utility 之 CAN RecvMsg Table 提供”Scroll”及”OverWrite”接收顯示模式選項

1.7	Edward	2011/08/19	<p>新功能需搭配以下軟體版本：  <b>FW: v1.06 / Utility: v1.10</b></p> <ol style="list-style-type: none"> <li>Utility 加入 Arbitration Lost 錯誤欄位顯示功能</li> <li>Utility 加入 Extra Config 功能選項</li> <li>Utility 加入 "Load SymbolFile" 功能</li> <li>Utility 之 CAN RecvMsg Table 的 Display Type 加入 "Sym" 接收顯示模式選項</li> </ol>
1.8	Edward	2012/07/27	<p>新功能需搭配以下軟體版本：  <b>FW: v1.07 / Utility: v1.12 / APILib: v1.09</b></p> <ol style="list-style-type: none"> <li>加入 CAN Error Frame 資訊顯示功能</li> <li>在 "使用者自訂定義 CAN Baud" 功能中，加入可設定 Bit-Timing 之取樣點 (即 Tseg2 值)</li> <li>Utility 加入以下三個選項功能： <ol style="list-style-type: none"> <li>Set CAN Error Frame Function</li> <li>Get CAN Error Frame Function</li> <li>Get CAN1/2 Baud Bit-Timing Parameter</li> </ol> </li> </ol>
1.9	Edward	2012/11/15	<ol style="list-style-type: none"> <li>Utility_v1.13 加入以下功能： <ol style="list-style-type: none"> <li>Load Reception List 功能</li> </ol> </li> <li>APILib_v1.10: <ol style="list-style-type: none"> <li>加入 "VCI_Set_MOD_Ex()" 函式</li> </ol> </li> </ol>
2.0	Alan	2015/04/29	美化 I-7565-H2 的接線圖與接腳配置圖
2.1	Edward	2015/06/16	<ol style="list-style-type: none"> <li>增加常問問題(FAQ)內容</li> <li>新增 Linux 平台使用手冊下載連結</li> </ol>
2.2	Edward	2016/06/06	<ol style="list-style-type: none"> <li>增加 5.7.4 ~ 5.7.7 節說明</li> <li>增加 FAQ 7.25 (Windows 10 相關問題)</li> <li>更新 Linux API 函式庫至 v0.3.6</li> </ol>
2.3	Edward	2016/07/21	<ol style="list-style-type: none"> <li>改善 VC_Demo1 及 VB_Demo1 功能</li> <li>Utility_v1.16，在 OverWrite 模式加入 CAN 訊息之最長/最短間隔時間</li> </ol>
2.4	Edward	2017/12/20	<ol style="list-style-type: none"> <li>擴充 7.12 說明</li> <li>增加 FAQ 7.26 及 7.27</li> </ol>