

I-7565M-HS 使用手冊

版本 1.1.0, 2023 年 1 月



產品技術服務與使用資訊

I-7565M-HS

保固說明

泓格科技股份有限公司（ICP DAS）所生產的產品，均保證原始購買者對於有瑕疵之材料，於交貨日起保有為期一年的保固。

免責聲明

泓格科技股份有限公司對於因為應用本產品所造成的損害並不負任何法律上的責任。本公司保留有任何時間未經通知即可變更與修改本文件內容之權利。本文所含資訊如有變更，恕不予另行通知。本公司盡可能地提供正確與可靠的資訊，但不保證此資訊的使用或其他團體在違反專利或權利下使用。此處包涵的技術或編輯錯誤、遺漏，概不負其法律責任。

版權所有

版權所有 2023 泓格科技股份有限公司保留所有權利。

商標識別

本文件提到的所有公司商標、商標名稱及產品名稱分別屬於該商標或名稱的擁有者所有。

技術服務

如有任何問題，請與本公司客服聯絡，我們將盡速為您服務。

Email 信箱：service@icpdas.com

目錄

1. 簡介.....	6
1.1. 規格	7
1.2. 特色	9
2. 技術資料.....	10
2.1. 架構圖	10
2.2. 外觀	10
2.3. 接腳定義.....	11
2.4. LED 指示燈	12
2.5. 終端電阻設定.....	14
2.6. 接線圖	16
3. CAN 總線網絡部署.....	17
3.1. CAN 總線驅動能力	17
4. 軟體設定工具.....	18
4.1. 安裝 I-7565-HS Utility 設定工具	18
4.2. 設定 I-7565M-HS 模組	21
4.3. 開始使用 I-7565-HS Utility 設定工具	22
4.3.1 使用設定工具與模組連線	24
4.3.2 傳送 CAN 總線訊息	26
4.3.3 接收 CAN 總線訊息	29
4.3.4 設定 CAN 總線訊息過濾器	32
4.3.5 設定模組其他參數	35
5. API 函式庫.....	38
5.1. API 函式庫概觀	38

5.2.	API 函式庫功能表	40
5.3.	API 函式庫使用流程	43
5.4.	模組初始化功能函式	44
5.4.1	CANHS_ScanDevice	44
5.4.2	CANHS_ListDevice	45
5.4.3	CANHS_OpenDevice	46
5.4.4	CANHS_CloseDevice	47
5.5.	模組設定功能函式	48
5.5.1	CANHS_SetCANOPMode	48
5.5.2	CANHS_GetCANOPMode	49
5.5.3	CANHS_SetCANBaudSP	50
5.5.4	CANHS_GetCANBaudSP	51
5.5.5	CANHS_GetCANBaudSPEEP	52
5.5.6	CANHS_SetCANFilter	53
5.5.7	CANHS_GetCANFilter	55
5.5.8	CANHS_SetCANWhiteListMode	56
5.5.9	CANHS_GetCANWhiteListMode	57
5.5.10	CANHS_SetCANBlackListMode	58
5.5.11	CANHS_GetCANBlackListMode	59
5.5.12	CANHS_SetCANWhiteListCANID	60
5.5.13	CANHS_GetCANWhiteListCANID	61
5.5.14	CANHS_SetCANBlackListCANID	62
5.5.15	CANHS_GetCANBlackListCANID	63
5.5.16	CANHS_GetCANStatus	64
5.5.17	CANHS_ResetModule	66
5.6.	模組通訊功能函式	67

5.6.1	CANHS_SetCANTxMsg	67
5.6.2	CANHS_GetCANRxMsg	69
5.6.3	CANHS_SetCANHWSendMode	71
5.6.4	CANHS_GetCANHWSendMode	72
5.6.5	CANHS_SetCANHWSendMsg	73
5.6.6	CANHS_GetCANRxFramePerSec	75
5.7.	資料緩衝區功能函式	76
5.7.1	CANHS_GetCANRxMsgCount	76
5.7.2	CANHS_ClearCANRxBuf	77
5.7.3	CANHS_ClearCANTxBuf	78
5.8.	其他功能函式	79
5.8.1	CANHS_GetDllVersion	79
5.8.2	CANHS_GetFwVer	80
5.8.3	CANHS_GetModuleStatus	81
5.9.	函式回傳碼	82
6.	韌體更新	83
7.	附錄	87
7.1.	文件版本歷史紀錄	87
7.2.	機構	88
7.3.	CAN Status 暫存器	89
7.4.	CAN Error Counter 暫存器	90

1. 簡介

I-7565M-HS 是具有兩個 CAN 通道的高速 USB 轉 CAN 總線轉換器，它提供更優於其他 I-7565 系列產品的資料傳輸效能，每個 CAN 埠允許最高收送 15000 封包/秒的 CAN 總線訊息資料量、且符合 CAN 2.0A/2.0B 協定標準，並且提供使用者自定義的 CAN 總線通訊速率設定範圍 10Kbps 到 1Mbps。如果使用者需在 PC 或筆記型電腦上使用此模組時，在正確的使用 USB 介面連接電腦與 I-7565M-HS 後，電腦會自動載入相關的驅動程式。也因此，透過應用 I-7565M-HS，使用者可以更容易、更快速地進行 CAN 總線網路的數據收集和處理，合適的應用領域可以是 CAN 總線監控、樓宇自動化、遠端數據採集、環境控制與監控、實驗室設備與研究、工廠自動化等。

以下是這 I-7565M-HS 模組的應用架構。PC 可以是 CAN 主機、監視器或 HMI，透過 I-7565M-HS 轉換器來透過 CAN 總線網路訪問及控制 CAN 總線設備。這模組可能讓使用者輕鬆地使用具備 USB 介面的 PC 來與 CAN 總線設備進行通訊。



1.1. 規格

模組	I-7565M-HS
CAN 介面	
通道數	2
接頭	8-pin terminal-block
通訊速率	10 ~ 1000 kbps
終端電阻	指撥開關設定120 Ω 終端電阻
隔離	3000 V DC-DC隔離, 依UL1577規範2500 Vrms持續一分鐘 (光耦合)
規範	ISO-11898-2, 支援 CAN 2.0A 與 CAN 2.0B
CAN 訊息過濾功能	可透過 Utility 工具設定 CAN 訊息 ID 黑/白名單過濾器
接收緩衝區	512 筆
訊息收發流量	15000 fps (Tx + Rx)
USB 介面	
接頭	B-type公座 x 1
規範	USB 2.0 High Speed (480Mbps)
驅動程式	Built-in Windows XP/7/8/10
LED	
圓形 LED	Power, MS, CAN1, CAN2, CAN1_ST, CAN2_ST LEDs
電源	
電源供應	USB 介面
功耗	1.5 W (Max.)
機構	
安裝方式	壁掛式安裝
外殼	金屬
尺寸(長 x 寬 x 高)	111.0 mm x 102.0 mm x 27.0 mm (W x L x H)
環境	
操作溫度	-25 ~ 75 $^{\circ}\text{C}$
儲存溫度	-30 ~ 80 $^{\circ}\text{C}$

1.2. 特色

- 相容於USB 2.0 (High Speed)規範
- 相容於ISO 11898-2標準規範
- 支援 CAN2.0A及CAN2.0B 協定標準
- 不需外接電源，直接由USB埠電源提供
- 支援CAN速率範圍從10kbps至1000kbps
- 支援 CAN ID 訊息過濾設定功能
- 接收到之CAN訊息時間戳記，精準度可達±10微秒。
- 內建Watchdog監測機制
- 資料接收緩衝區支援512筆CAN資料訊框
- 模組單一CAN埠的最大總資料流量(傳送+接收)可達每秒15000個資料訊框 (視使用者的電腦效能而定)
- CAN總線內建120歐姆終端電阻 (可透過指撥開關調整)。
- 提供電源、CAN資料收送、CAN總線狀態等指示燈
- 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

2. 技術資料

2.1. 架構圖

下圖為 I-7565M-HS 功能架構圖，CAN 埠具備 3000Vrms 隔離電壓保護

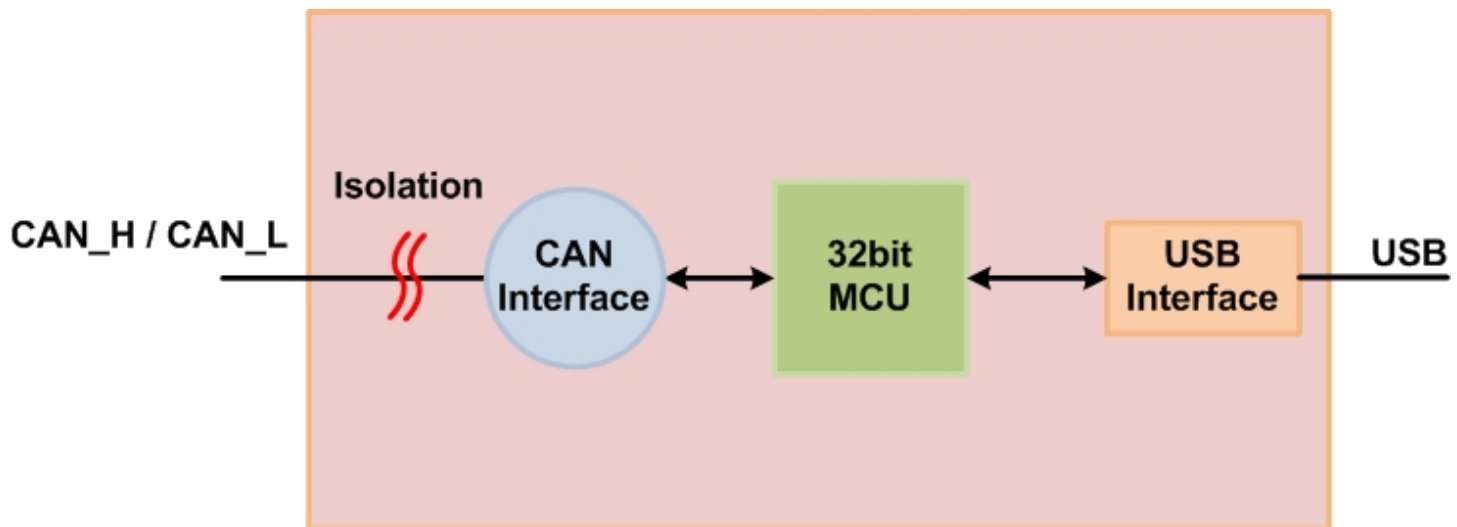


圖 2-1 I-7565M-HS 架構圖

2.2. 外觀

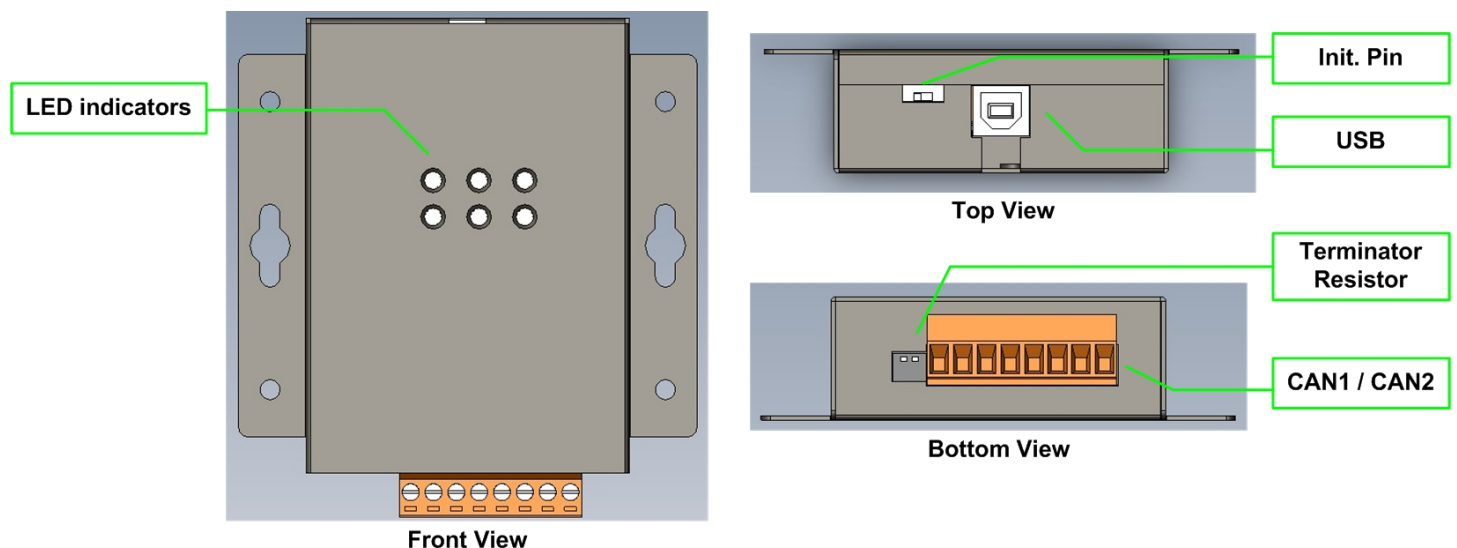


圖 2-2 I-7565M-HS 外觀

2.3. 接腳定義

I-7565M-HS 8-pin terminal block 端子接腳定義如下圖所示

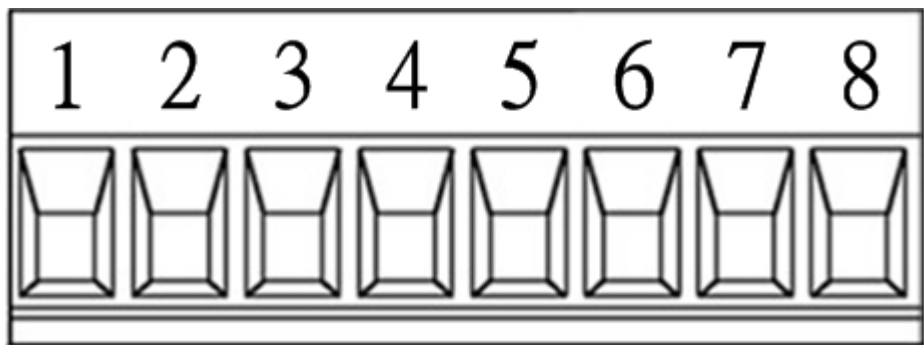


表 2-1 接腳定義

腳位	名稱	描述
1	CAN_GND	CAN1 埠的 CAN_Gnd 訊號腳位
2	CAN_L	CAN1 埠的 CAN_Low 訊號腳位
3	F.G.	大地接地腳位
4	CAN_H	CAN1 埠的 CAN_High 訊號腳位
5	CAN_GND	CAN2 埠的 CAN_Gnd 訊號腳位
6	CAN_L	CAN2 埠的 CAN_Low 訊號腳位
7	F.G.	大地接地腳位
8	CAN_H	CAN2 埠的 CAN_High 訊號腳位

電子電路設備容易受到不同程度的靜電放電 (ESD, Electro-Static Discharge) 現象所干擾，在大陸性氣候區會變得更糟。F.G.腳位提供了一個進行防靜電的接地路徑。因此，正確地連接 F.G.可以增強的 ESD 保護能力，並提高了模組的可靠性。

大地接地(F.G.)的接線並不是必要的，根據實際應用中，用戶可以修改佈線的配置。

2.4. LED 指示燈

I-7565M-HS 模組共有 6 個 LED 指示燈。一個電源指示燈、一個硬體狀態指示燈、四個 CAN 總線資料收送及狀態指示燈和指示燈。LED 的分配和描述如下所示。

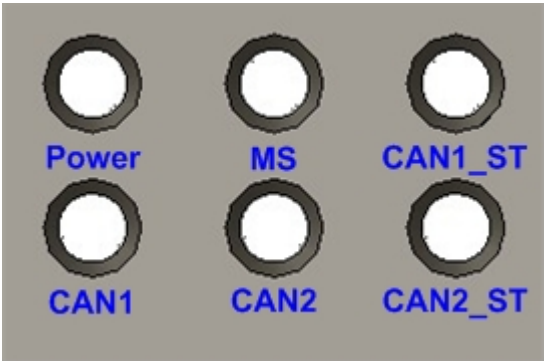


圖 2-3 I-7565M-HS LED 指示燈

表 2-2 LED 指示燈功能描述

LED 名稱	顏色	功能描述
Power	紅色	電源狀態，當 USB 端有電源供電時，此燈號會恆亮
MS	紅色	模組狀態 恆滅：無異常 恆亮：硬體故障
CAN1_ST	紅色	CAN 總現狀態. 恆滅：無異常 恆亮: CAN1 Bus Off 閃爍: CAN1 總線出現暫時性可回復之錯誤或緩衝區溢位
CAN2_ST	紅色	CAN Bus 狀態. 恆滅：無異常 恆亮: CAN2 Bus Off 閃爍: CAN2 總線出現暫時性可回復之錯誤或緩衝區溢位
CAN1	綠色	恆滅: CAN1 無訊息正在傳送 閃爍: CAN1 埠正再傳送訊息
CAN2	綠色	恆滅: CAN2 無訊息正在傳送 閃爍: T CAN1 埠正再傳送訊息

備註:

在韌體更新模式下:

模組 6 個 LED 指示燈會依 Power”, “MS”, “CAN1_ST”, “CAN2_ST”, “CAN2”, “CAN1” 的順序依順時針方向閃爍。

2.5. 終端電阻設定

為了盡量減少 CAN 總線上的反射效果，CAN 總線具有在兩端使用兩個終端電阻將總線終止。根據 ISO11898-2 規範，每一個終端電阻為 120Ω (或 $108\Omega\sim132\Omega$ 之間)。總線拓撲結構和這些終端電阻的位置，如下圖所示。

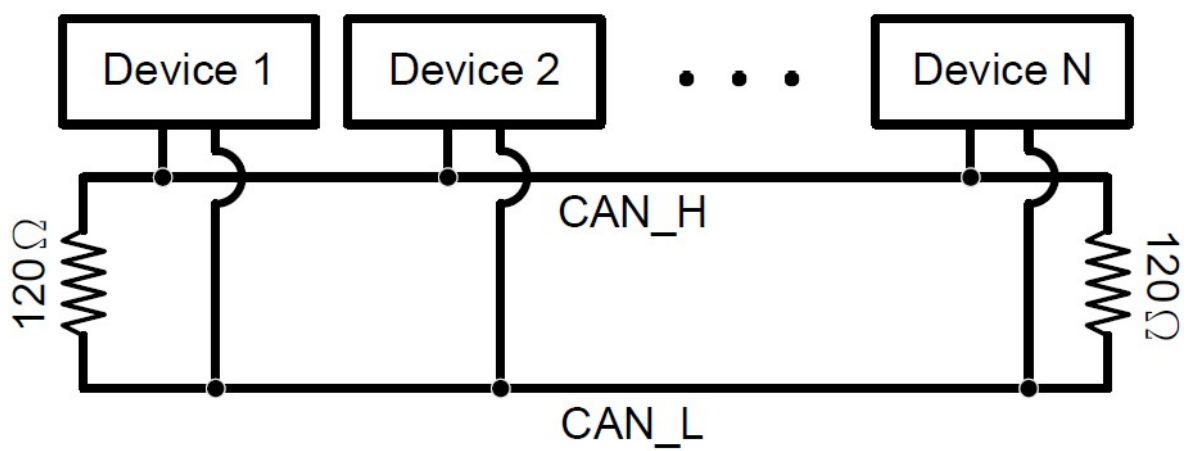


圖 2.4 CAN 總線網絡拓撲

I-7565M-HS 模組每個 CAN 埠均有一個內建的 120Ω 終端電阻，使用者可以透過調整指撥開關來決定是否啟用。終端電阻的指撥開關位於的模組的上方，如下圖所示。

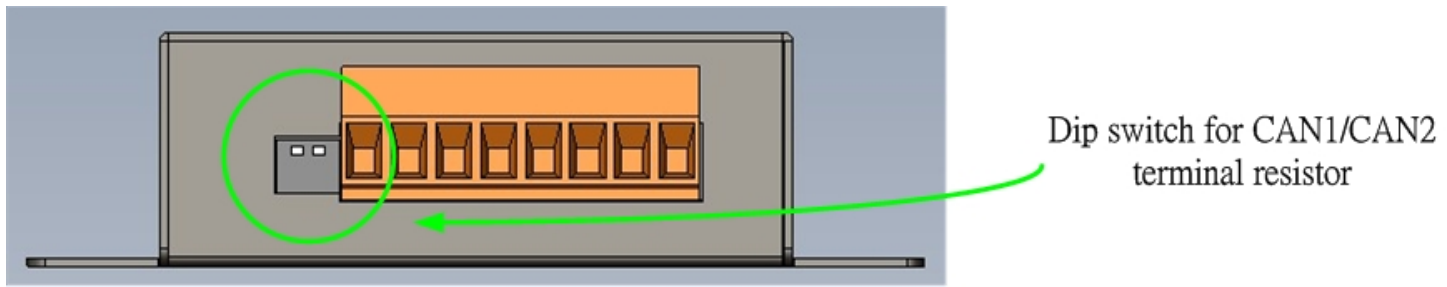
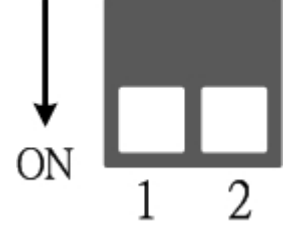


圖 2-5 I-7565M-HS 終端電阻指撥開關位址

下圖為終端電阻指撥開關啟用及停用的位置圖，預設為啟用狀態。

表 2-3 終端電阻對應之指撥開關位置調整說明

	開關位址	描述
	1	ON: 啟用 CAN1 終端電阻 (預設) OFF: 停用 CAN1 終端電阻
	2	ON: 啟用 CAN2 終端電阻 (預設) OFF: 停用 CAN2 終端電阻

一般來說，如果您的應用架構如下，我們建議您啟用終端電阻。

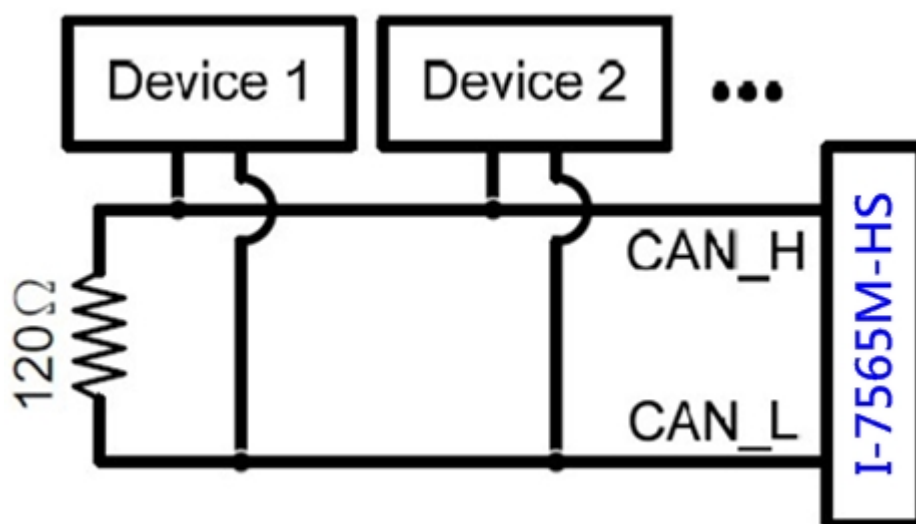


圖 2-6 應用-啟用終端電阻

如果您的應用程序為下圖之架構，則沒有必要啟用終端電阻。

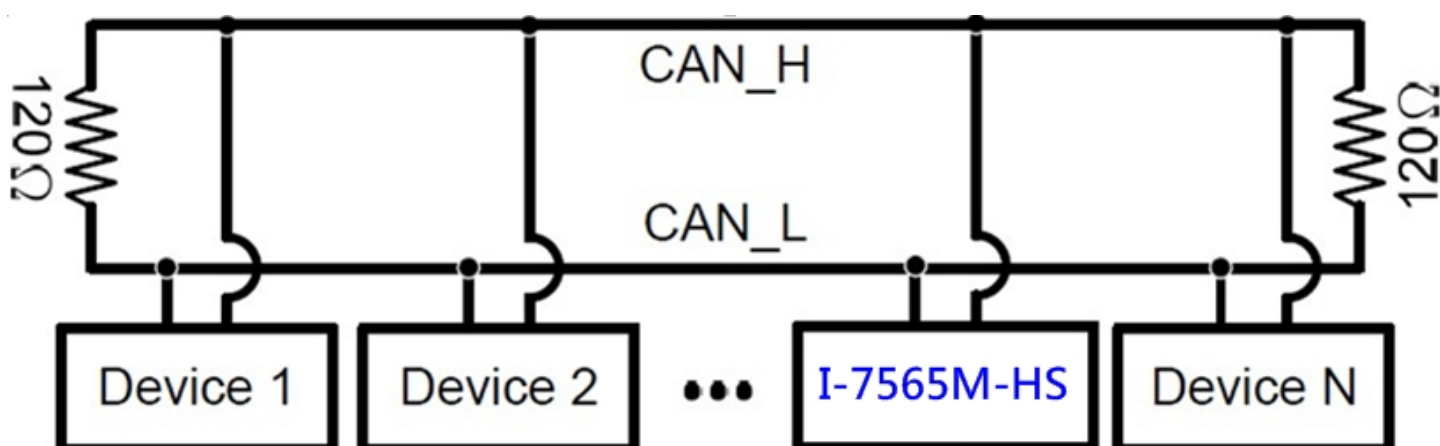


圖 2-7 應用-停用終端電阻

2.6. 接線圖

I-7565M-HS 模組的 USB 及 CAN 總線接線圖如下圖所示。

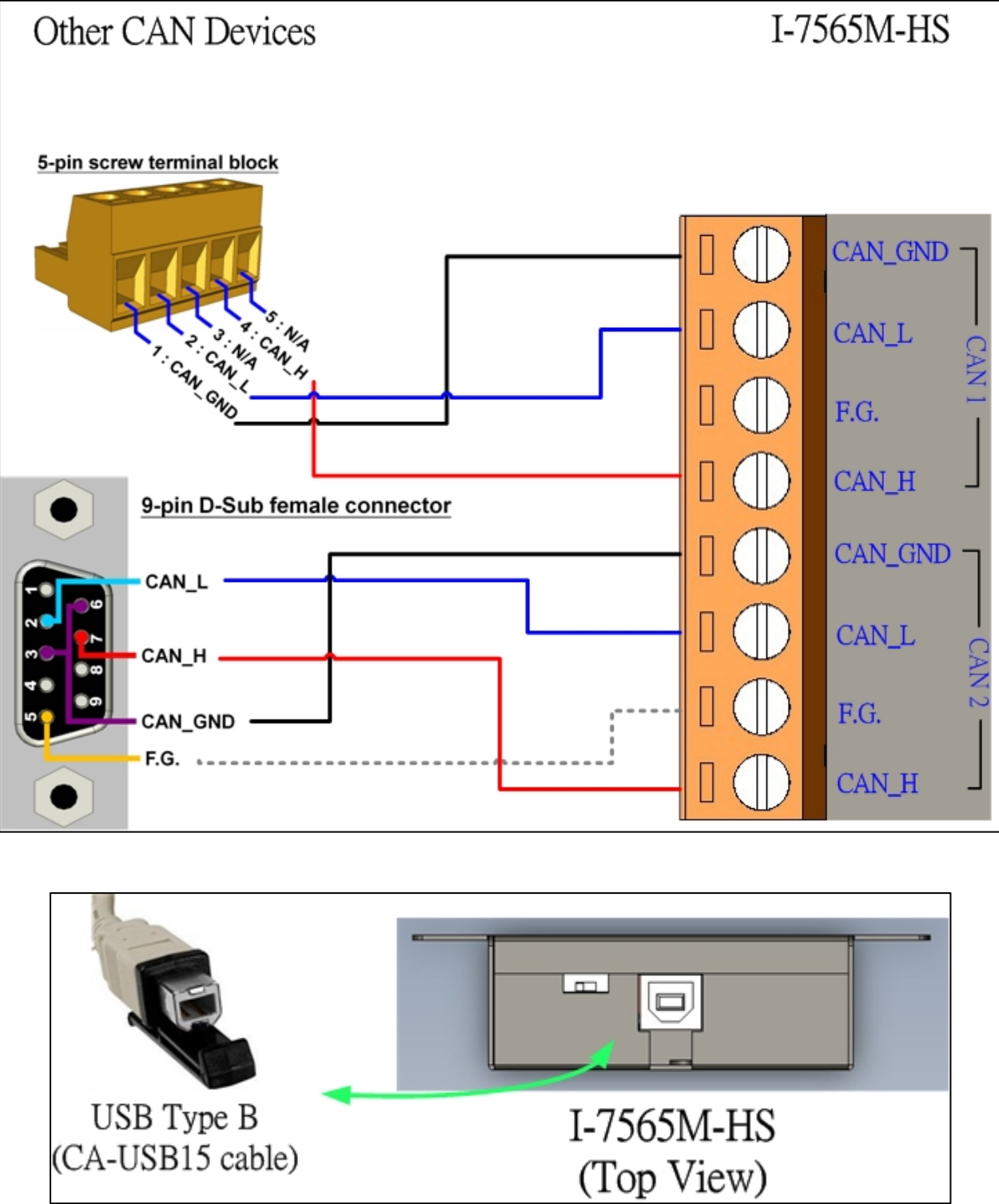


圖 2-8 I-7565M-HS USB 及 CAN 總線接線圖

3. CAN 總線網絡部署

3.1. CAN 總線驅動能力

在介紹 I-7565M-HS 模組的 CAN 總線驅動能力之前，必須先假定銅電纜的某些特徵。阻值參數為 120Ω 阻抗和毫秒/線延遲，直流參數如下面的表格所述。

表 3-1 CAN 總線導線截面建議的阻值

導線截面[mm ²]	阻值[Ω/km]
~0.25 (AWG23)	< 90
~0.5 (AWG20)	< 50
~0.8 (AWG18)	< 33
~1.3 (AWG16)	< 20

如上所述的條件下，用戶可以參考下表來得知按照 ISO11898-2 規範，每個段中的最大的節點數以及使用不同類型的導線時，最大數據段長度為何。

表 3-2 驅動能力

導線截面[mm ²]	在特定節點數量情況下的最大節段長度[m]			
	16 節點	32 節點	64 節點	100 節點
~0.25 (AWG23)	< 220	< 200	< 170	< 150
~0.5 (AWG20)	< 390	< 360	< 310	< 270
~0.8 (AWG18)	< 590	< 550	< 470	< 410
~1.3 (AWG16)	< 980	< 900	< 780	< 670

4. 軟體設定工具

我們提供的 I-7565-HS 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。接下來，我們將說明如何使用 I-7565-HS Utility 的主要功能。

4.1. 安裝 I-7565-HS Utility 設定工具

步驟 1: 取得 I-7565M-HS 設定工具

軟體路徑如下:

<https://www.icpdas.com/tw/download/show.php?num=3963&model=I-7565M-HS>

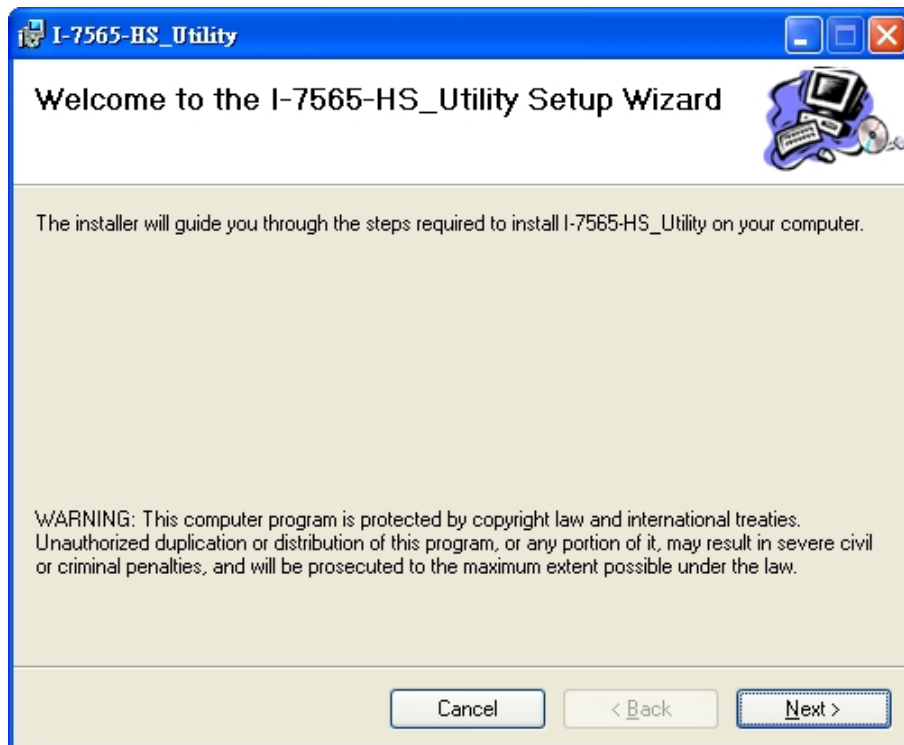
步驟 2: 安裝 .NET Framework 3.5 元件

The I-7565M-HS 工具需要使用到 .NET Framework 3.5 元件。在執行“Setup.exe”檔案後，如果你的電腦沒有安裝此元件的話，設定程式將會開始由網路安裝此必備的元件。

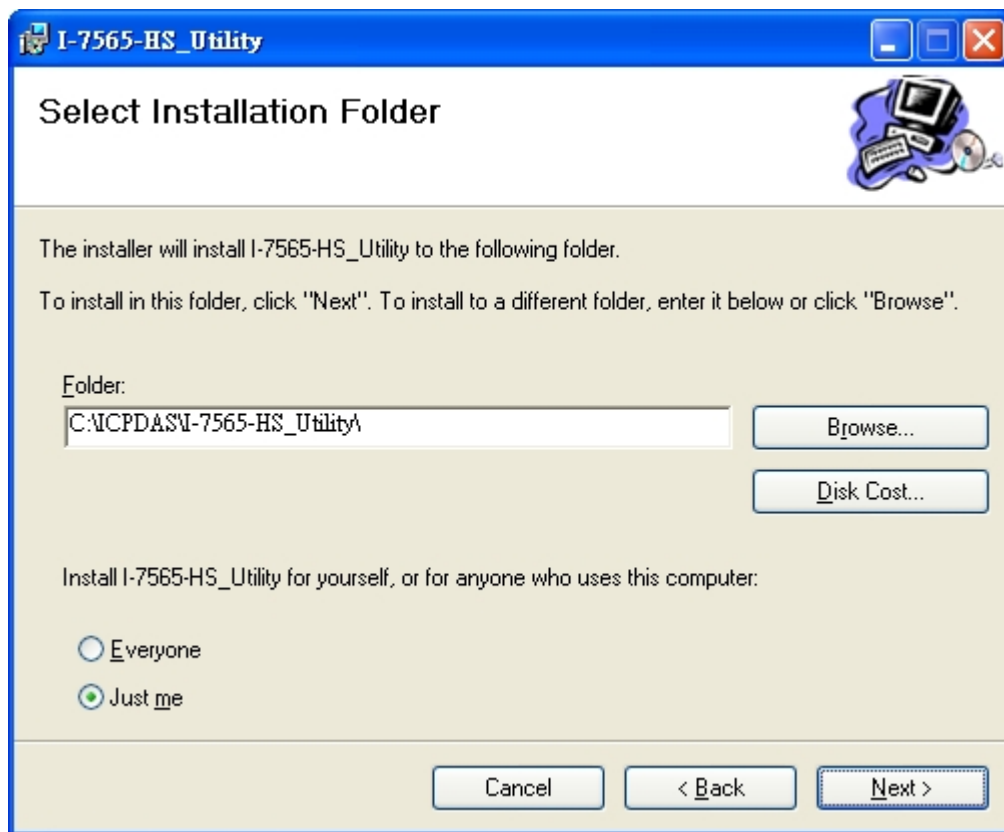
步驟 3: 安裝設定工具

在安裝完 .Net Framework 元件後軟體將會繼續安裝設定工具。

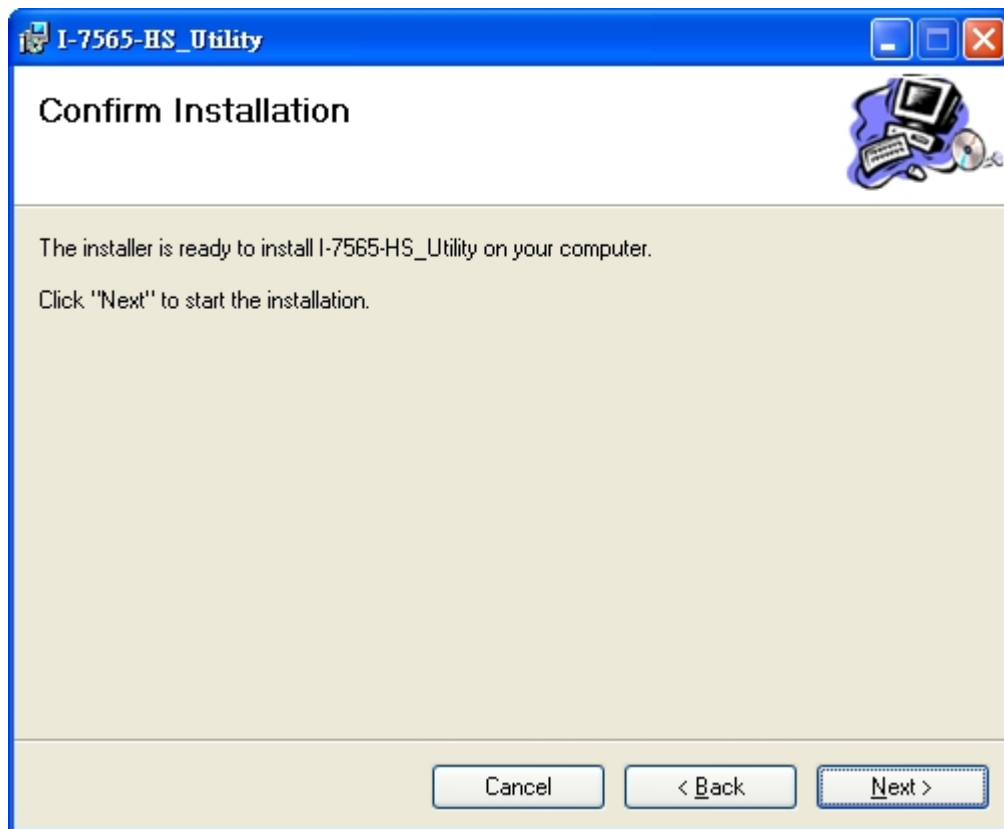
1. 點擊“Next”按鈕，執行下一步動作。



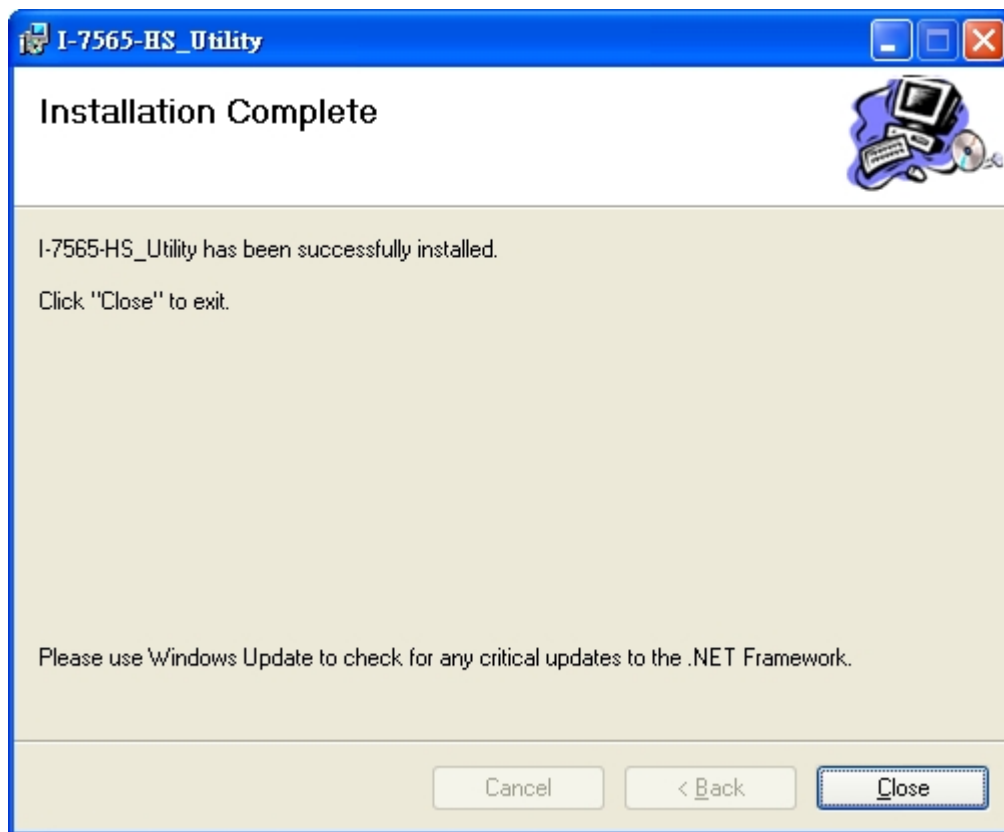
2. 選擇 I-7565-HS Utility 安裝路徑，並點擊“Next 按鈕，執行下一步動作。



3. 確認安裝。點擊 “Next” 按鈕，開始安裝。



4. 安裝完成。點擊 “Close” 按鈕退出。



4.2. 設定 I-7565M-HS 模組

安裝完 I-7565-HS Utility 工具後，請按照以下步驟設置 Utility 工具和 I-7565M-HS 模組之間的通信。

步驟 1: 在產品盒內可以找到 CA-USB15 纜線，使用此覽線來連接模組與 PC 端的 USB 埠。

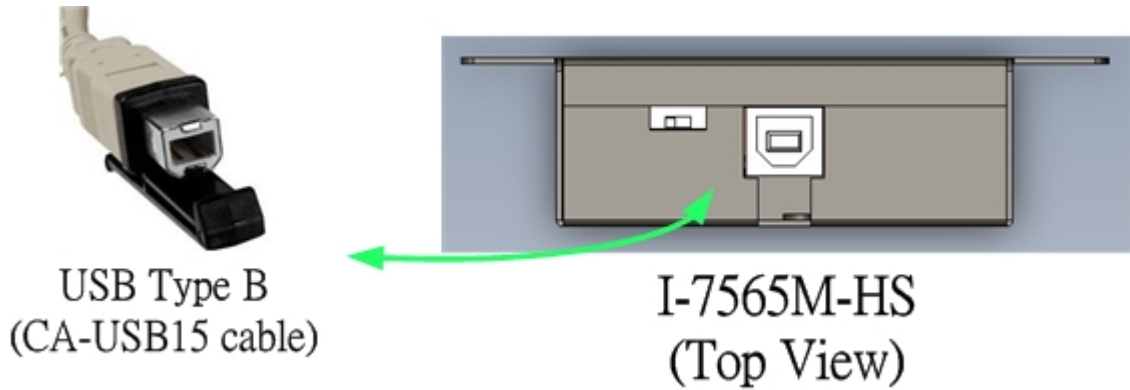


圖 4-1 模組 USB 接線

步驟 2: 執行 I-7565-HS Utility 工具程式。

4.3. 開始使用 I-7565-HS Utility 設定工具

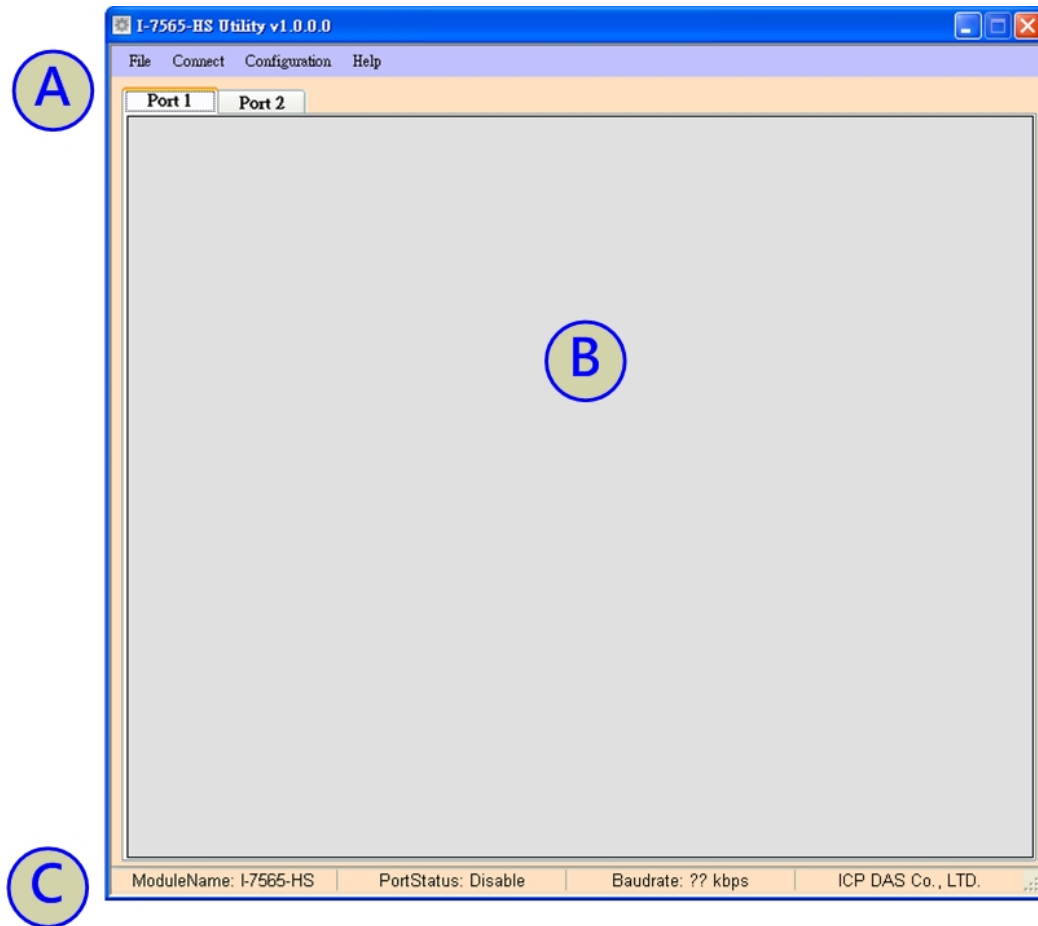


圖 4-2 I-7565-HS Utility 設定工具主畫面

A 選單工具列

[File]

載入/儲存所有設定在“CAN 總線訊息傳送框”上的相關訊息 以及 儲存所有在“CAN 總線訊息接收框”上的所有訊息。

[Connect]

與模組連線或者斷線

[Configuration]

開啟“Module Configuration”訊息框，用來設定模組的 CAN ID 訊息過濾器及其他功能。

[Help]

關於設定工具的版本資訊。

- B 傳送/接收訊息框。當與模組連線後，此訊息框會分成上/下兩個框架，上方用來傳送 CAN 總線訊息，下方用來顯示模組所接收到的 CAN 訊息。
- C 狀態列。當與模組連線後，使用者可於此工具列讀到 CAN 埠的相關設定資訊。

4.3.1 使用設定工具與模組連線

當一開始執行設定工具時，設定工具會自動掃描所有可使用的 I-7565M-HS 模組，且將掃描到的模組條列在設定工具”Connection to ...”訊息框上的”Module Name”欄位。使用者可以透過重新執行”Connect to ...”功能來重新掃描新增之 I-7565M-HS 模組。

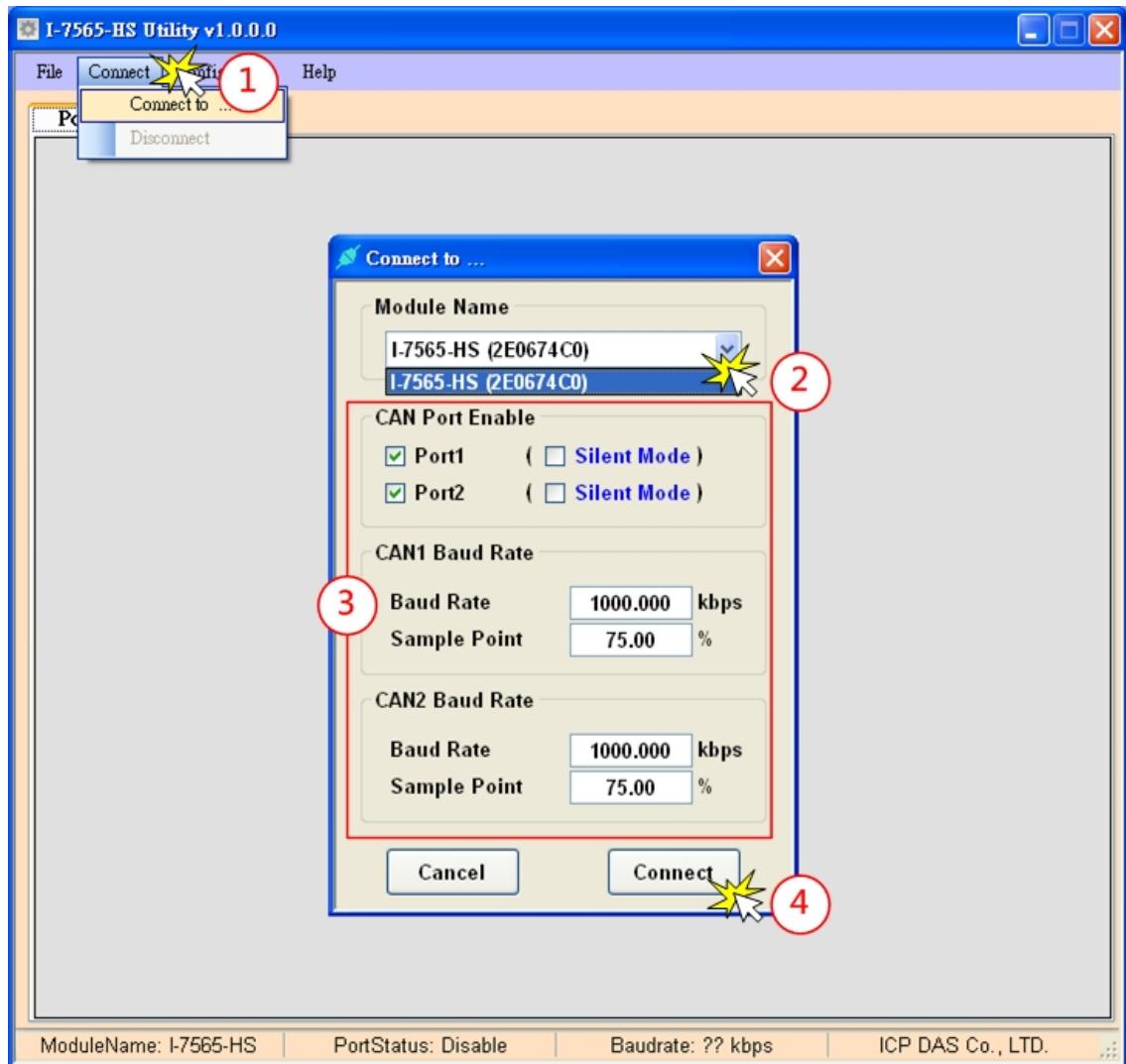


圖 4-3 I-7565-HS Utility 射鏡工具之”Connect to ...”畫面

在與模組連線前，使用者可以設定模組的 CAN 埠操作模式及 CAN 總線的通訊速率等參數。請參考下列步驟來設定模組的相關連線設定參數。

步驟 1: 點選”Connect to ...”項目，開啟”Connect”設定畫面

步驟 2: 選擇欲連線之 I-7565M-HS 模組

步驟 3: 使用者可以在“CAN Port Enable”及“CAN1/CAN2 Baud Rate”欄位 設定 CAN 埠操作模式及 CAN 總線的通訊速率等參數，詳細功能條列如下:

[CAN Port Enable]

“Port Enable”： 啟用/停用 CAN1/CAN2 埠功能

“Silent Mode”： 設定 CAN 埠進入監聽模式。當 CAN 埠為監聽模式時，此 AN 埠將只會接收 CAN 總線訊息而不會發出 CAN Ack 訊息至 CAN 總線上。

[CAN Baudrate]

“Baud Rate”： 欲使用之 CAN 總線速率。數值範圍為 10 ~ 1000 kbps。

“Sample Point”： CAN baud rate bit timing 取樣點。

步驟 4: 按下“Connect”按鈕，開始使用上述 CAN 總線之設定來傳送/接收 CAN 訊息。

4.3.2 傳送 CAN 總線訊息

經由使用設定工具，使用者可以透過 I-7565M-HS 模組來傳送 CAN 總線訊息。在成功與模組連線後，CAN 總線通訊功能將會顯示在設定工具上，如下圖所示。

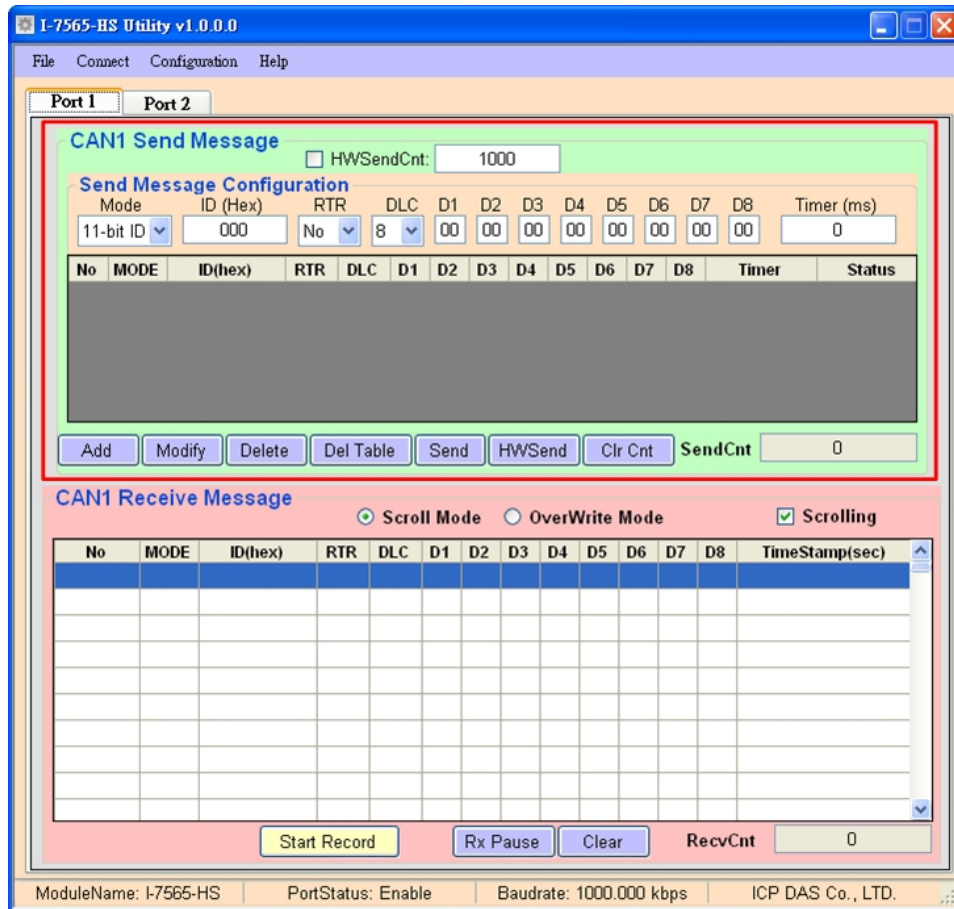


圖 4-4 I-7565-HS Utility 之 CAN1/2 資料通訊畫面

上圖為設定工具之“Communication”畫面，可區分成兩個區塊，上方為“Send Message”區塊，下方為“Receive Message”區塊。使用者可透過“Port 1” / “Port 2”標籤，來切換 CAN1/CAN2 的“Communication”設定畫面。次外，使用者可透過“CAN Message Send Area”畫面來傳送 CAN 訊息。

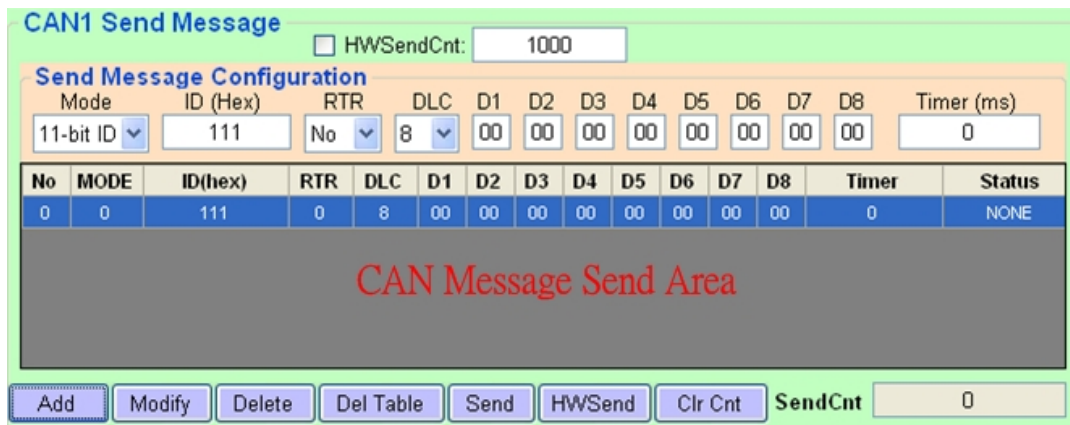


圖 4-5 I-7565-HS Utility 設定工具之“Send Message”畫面

[Send Message]區塊:

<1> “Send Message Configuration” 訊息框:

此訊息框用來給使用者編輯欲傳送之 CAN 總線訊息參數，使用者可以使用“Add”/“Modify”/“Delete”按鈕來新增/修改/刪除“CAN Message Send Area”中的 CAN 訊息。

<2> “Add” 按鈕:

此按鈕功能在由“Send Message Configuration”欄位新增一筆 CAN 訊息至“CAN Message Send Area”欄位的最後一行。

<3> “Modify” 按鈕:

此按鈕功能在於使用“Send Message Configuration”上的 CAN 訊息來替換“CAN Message Send Area”所選擇之行數訊息。

<4> “Delete” 按鈕:

此按鈕功能在於刪除一筆“CAN Message Send Area”中所選擇之訊息。

<5> “Del Table” 按鈕：

此按鈕功能在於刪除“CAN Message Send Area”欄位中所有的訊息。

<6> “Send” 按鈕:

此按鈕功能在於傳送“CAN Message Send Area”欄位所選擇的訊息。當“Timer”欄位為 0 時，表示只送出一筆訊息。當“Timer”不為 0 時，表示會依設定值，周期性的傳送該筆訊息。

<7> “HWSend” 按鈕:

此按鈕功能在於透過模組內部的硬體時間計數器，將“CAN Message Send Area”欄位中所選擇之訊息傳送至 CAN 總線上。當選擇之訊息的“Timer”欄位為 0 時，表示只傳送一次。當“Timer”欄位非 0 時，此訊息將會透過模組內部的硬體時間計數器來傳送。使用此“HWSend”按鈕所傳送的訊息時間間隔將會比使用“Send”按鈕還要即時。當使用者要傳送固定次數之訊息時，則需要啟用“HWSendCnt”且輸

入要傳送的筆數後，在按下“HWSend”按鈕時，模組就會開始進行傳送。

<8> “Clr Cnt” 按鈕:

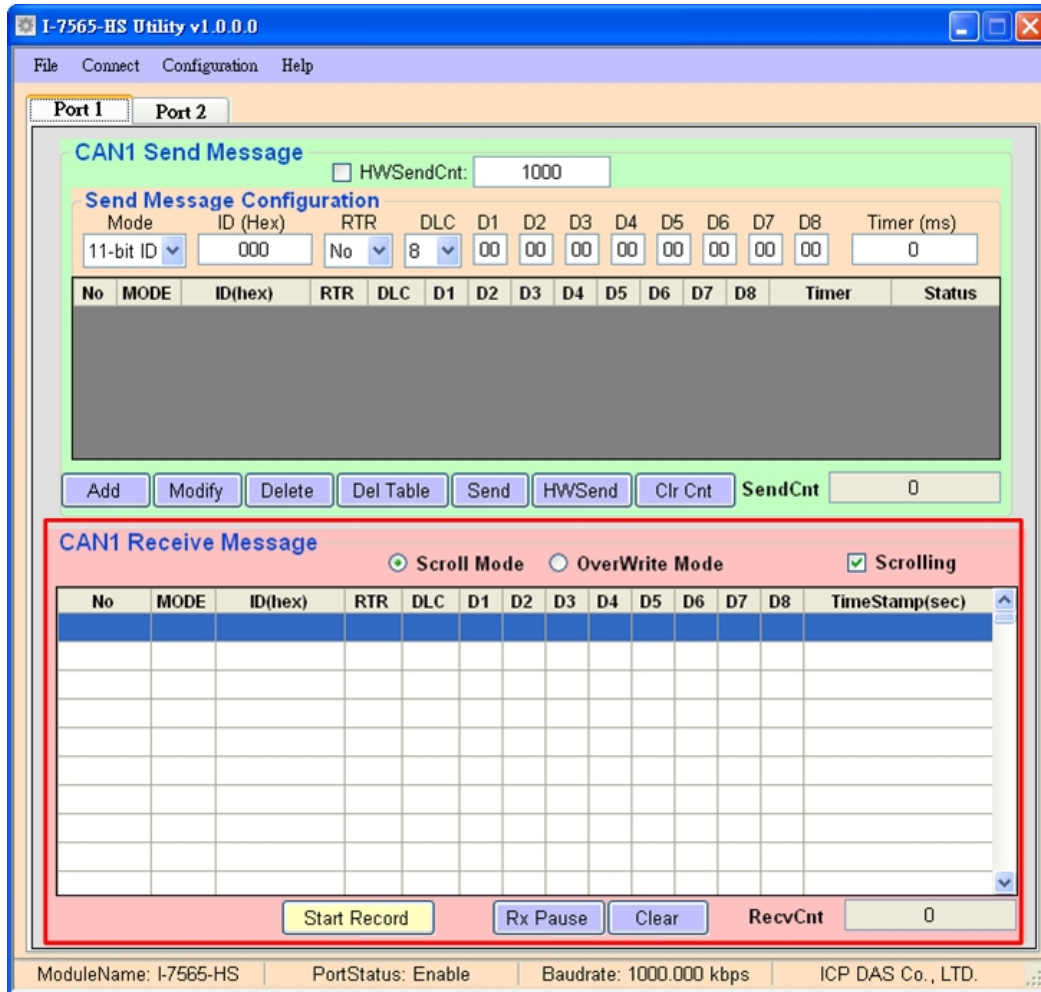
此按鈕功能在於將“CAN Message Send Area”欄位中“SendCnt”參數所計數之傳送次數歸 0。

<9> “SendCnt” 欄位參數:

當 CAN 總線訊息透過設定工具成功傳送時，此參數將會遞增 1。

4.3.3 接收 CAN 總線訊息

經由使用設定工具，使用者可以透過 I-7565M-HS 模組來接收 CAN 總線訊息。在成功與模組連線後，CAN 總線通訊功能將會顯示在設定工具上，如下圖所示。



在與 I-7565M-HS 設備連線後，模組 CAN 總線所接收到的資料或異常訊息都會顯示在“CAN Message Receive Area”訊息框上。

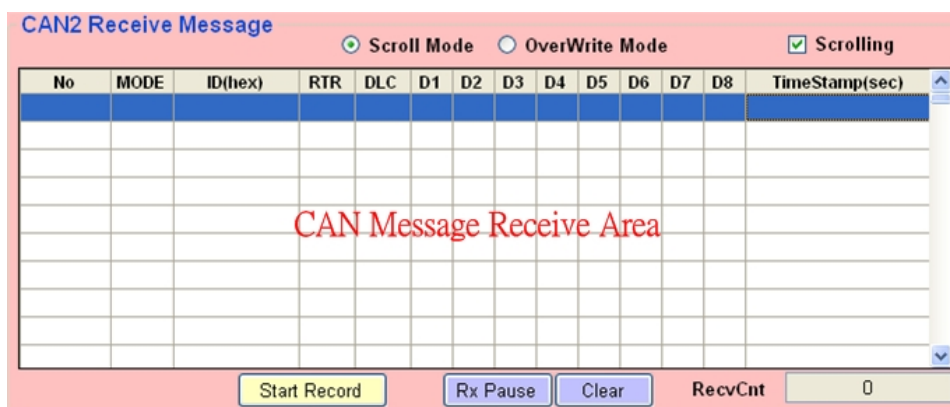


圖 4-6 I-7565-HS Utility 設定工具之“Receive Message”畫面

[Receive Message]區塊:

<1> “Start Record / Stop Record” 按鈕:

當使用者按下 “Start Record” 按鈕時，原先顯示在 “CAN Message Receive Area” 的 CAN 訊息將會以 ASCII 文本的格式存成檔案。當按下 “Stop Record” 按鈕時，則會停止紀錄。CAN1 埠所儲存的檔案名稱將會是 “CAN1_YYMMDD_HHMMSS.txt”，而 CAN2 埠的檔名則是 “CAN2_YYMMDD_HHMMSS.txt”，每個檔案最大儲存之檔案大小為 200MB，超過時則會在另存成一份檔案。

<2> “Rx Start / Rx Pause” 按鈕:

當按下 “Rx Start” 按鈕時，設定工具將會開始接收 CAN 訊息
當按下 “Rx Pause” 按鈕時，設定工具將會停止接收 CAN 訊息

<3> “Clear” 按鈕:

此按鈕功能在於清除所有 “CAN Message Receive Area” 上的訊息且將 “RecvCnt” 數值歸 0。

<4> “Scrolling” 核取鈕:

當 “Scrolling” 核取鈕有被核取時，“CAN Message Receive Area” 上的訊息會自動地以捲動方式呈現新的訊息，當此按鈕沒被核取時，則訊息畫面不會自動呈現新的訊息。

<5> “Scroll / OverWrite Mode” 選項:

“Scroll Mode” :

此模式下，每筆接收到的訊息將會依序顯示在 “CAN Message Receive Area” 畫面上。

“Overwrite Mode” :

此模式下，當接收到的訊息MODE及ID相同時，該筆資料會被擺在 “CAN Message Receive Area” 上的同一行。“Num” 欄位將會顯示接收到的資料筆

數，而“CycleTime”欄位則會顯示接收到此訊息的周期平均值/最大及最小值。

“CycleTime”欄位之功能描述如下：

[1] 0.4972 (Sec) => CAN訊息平均周期(約500ms)

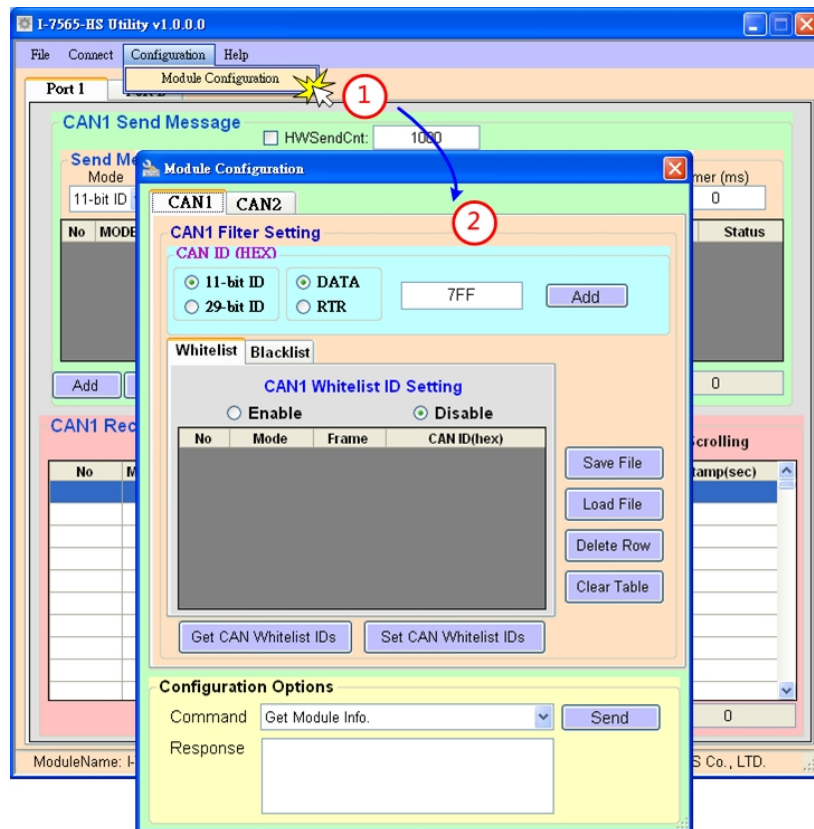
[2] 0.6536 (Sec) => CAN訊息周期最大值

[3] 0.4972 (Sec) => CAN 訊息周期最小值

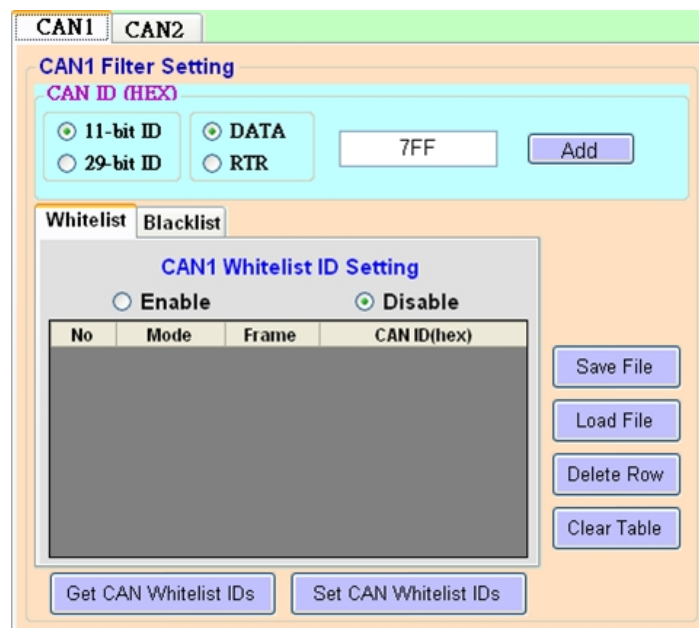
CAN1 Receive Message													
☐ Scroll Mode ☒ OverWrite Mode ☒ Scrolling													
Num	MODE	ID(hex)	RTR	DLC	D1	D2	D3	D4	D5	D6	D7	D8	CycleTime(sec)
59	0	123	No	8	11	22	33	44	55	66	77	88	0.4972 (0.6536/0.4972)

4.3.4 設定 CAN 總線訊息過濾器

經由使用設定工具，使用者可以設定模組的 CAN 總線訊息過濾功能。



在點選“Module Configuration”項目後，使用者可以在“Whitelist”訊息框設定欲接收的 CAN 訊息，以及在“Blacklist”訊息框設定不想接收的 CAN 訊息。“CAN1” / “CAN 2” 頁籤是用來切換 CAN1 / CAN2 埠的訊息過濾畫面。



[CAN ID (HEX)] 區塊:

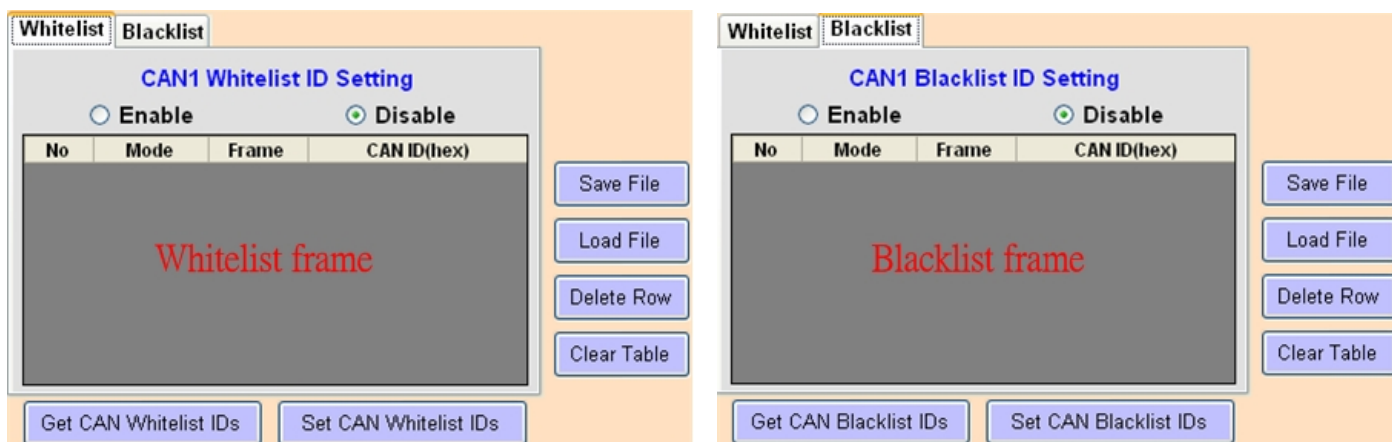


The interface for adding a CAN ID. It features two radio buttons for '11-bit ID' (selected) and '29-bit ID'. To the right are two more radio buttons for 'DATA' (selected) and 'RTR'. A text input field contains the value '7FF'. An 'Add' button is located to the right of the input field.

<1> “Add” 按鈕:

此按鈕功能在於新增一 “11-bit ID 或 29-bit ID” 及 “DATA 或 RTR” 格式的 CAN 訊息至 “Whitelist/Blacklist” 區塊。

[Whitelist/Blacklist] 區塊:



Two side-by-side configuration panels for CAN1. The left panel is titled 'CAN1 Whitelist ID Setting' and the right is 'CAN1 Blacklist ID Setting'. Both panels have 'Whitelist' and 'Blacklist' tabs at the top. Each panel contains an 'Enable' (radio button) and a 'Disable' (radio button, selected) option. Below the options is a table with columns: 'No', 'Mode', 'Frame', and 'CAN ID(hex)'. The table area is labeled 'Whitelist frame' on the left and 'Blacklist frame' on the right. To the right of each table are buttons for 'Save File', 'Load File', 'Delete Row', and 'Clear Table'. At the bottom of each panel are buttons for 'Get CAN Whitelist IDs' and 'Set CAN Whitelist IDs' (left) and 'Get CAN Blacklist IDs' and 'Set CAN Blacklist IDs' (right).

<1> “Enable/Disable” 選項:

啟用或停用所選擇的 CAN1/CAN2 埠的 CAN 訊息過濾黑白名單功能。

<2> “Get CAN Whitelist IDs/Get CAN Blacklist IDs” 按鈕:

由模組取得 CAN 訊息黑白名單的功能設定值。

<3> “Set CAN Whitelist IDs/Set CAN Blacklist IDs” 按鈕:

將 Whitelist/Blacklist 訊息框上的 CAN 訊息黑白名單設定值寫入模組。

<4> “Save File” 按鈕:

將 Whitelist/Blacklist 訊息框上的 CAN 訊息黑白名單設定值寫入一 ini 檔案。

<5> “Load File” 按鈕:

由所選擇的 ini 檔案，讀取 CAN 訊息黑白名單設定值並將此設定值載入至

Whitelist/Blacklist 訊息框上。

<6> **“Delete Row”** 按鈕:

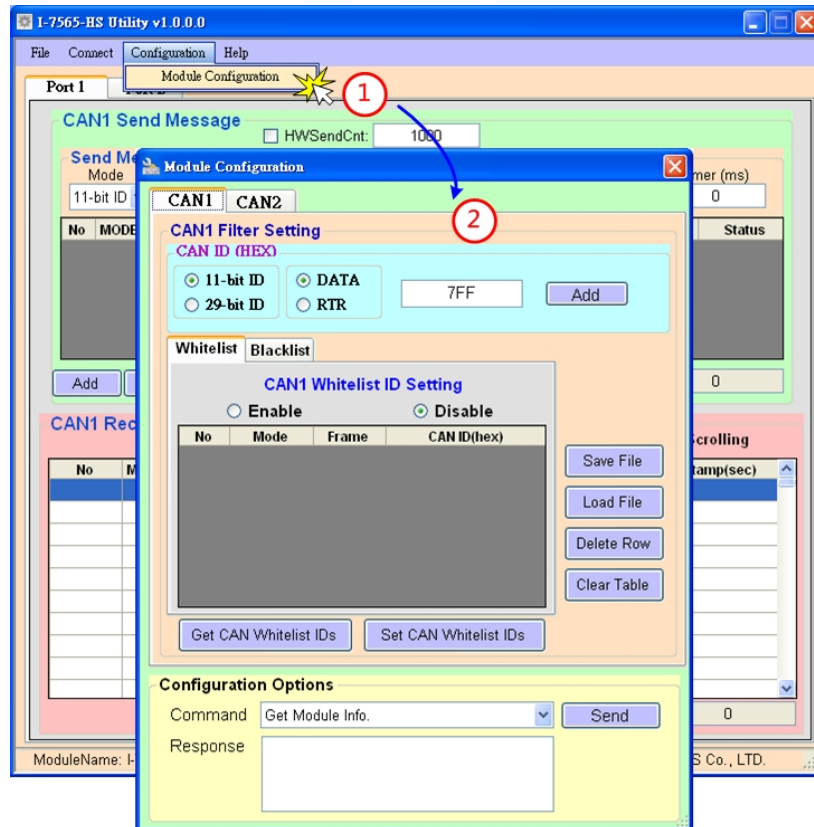
由 Whitelist/Blacklist 訊息框上，刪除一筆所選擇的 CAN ID 欄位訊息。

<7> **“Clear Table”** 按鈕:

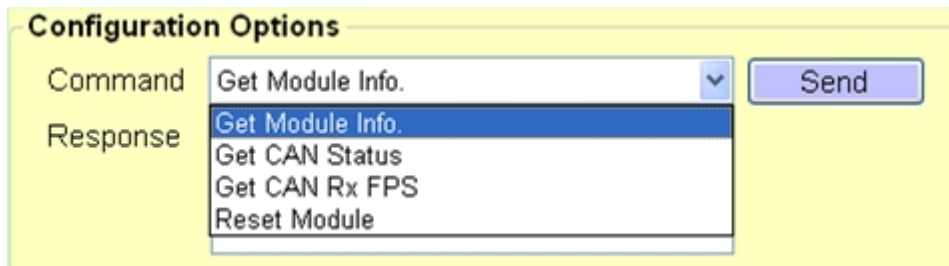
清除 Whitelist/Blacklist 訊息框上所有的 CAN ID 欄位訊息。

4.3.5 設定模組其他參數

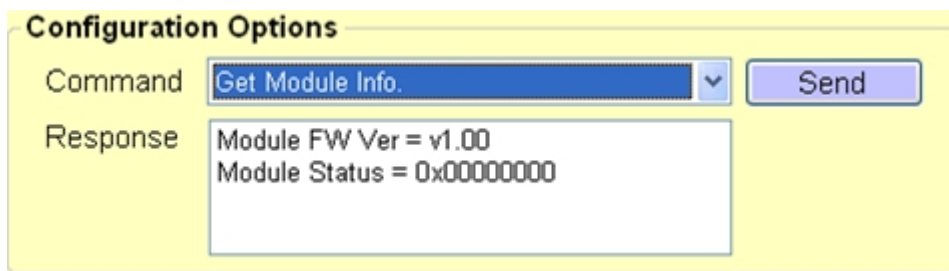
經由使用設定工具，使用者可以透過命令來讀取模組的其他設定參數功能。



由“Configuration Options”區塊中，使用者可以經由傳送設定命令來讀取模組狀態(韌體版本、硬體狀態)、CAN 狀態(CAN 總線狀態、錯誤筆數、緩衝區狀態)、CAN 總線的流量以及重置模組。設定工具上的“CAN1” / “CAN 2”頁籤是用來切換 CAN1/CAN2 埠。



[Get Module Info.] 命令:



Configuration Options

Command: **Get Module Info.** [Send]

Response: Module FW Ver = v1.00
Module Status = 0x00000000

<1> “Module FW Ver” 選項:

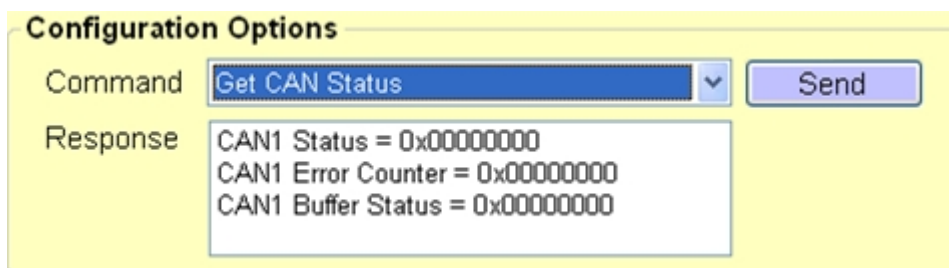
v1.00 表示模組的韌體版本為 1.00。

<2> “Module Status” 選項:

0x: 表示此數值為 16 進制數值

位元	數值	功能描述
0		模組的USB初始化狀態
	0	初始化USB正常
	1	初始化USB異常
31:1	-	保留

[Get CAN Status] 命令:



Configuration Options

Command: **Get CAN Status** [Send]

Response: CAN1 Status = 0x00000000
CAN1 Error Counter = 0x00000000
CAN1 Buffer Status = 0x00000000

<1> “CAN Status” 選項:

0x: 表示此數值為 16 進制數值

詳細資訊請參考附錄 7.3 的“CAN Status”定義說明。

<2> “CAN Error Counter” 選項:

0x: 表示此數值為 16 進制數值

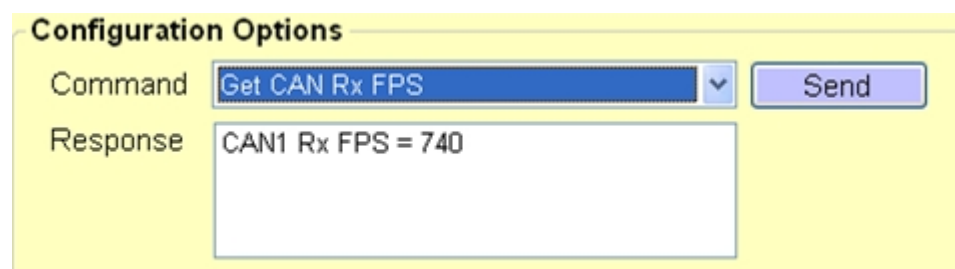
詳細資訊請參考附錄 7.4 的“CAN Error Counter”定義說明。

<3> “CAN Buffer Status” 選項:

0x: 表示此數值為 16 進制數值

位元	符號	數值	功能描述
0	RX		CAN1/CAN2埠接收緩衝區的狀態
		0	接收緩衝區狀態正常
		1	接收緩衝區資料溢位
1	TX		CAN1/CAN2埠傳送緩衝區的狀態
		0	傳送緩衝區狀態正常
		1	傳送緩衝區資料溢位
31:2		-	保留

[Get CAN Rx FPS] 命令:



Configuration Options

Command: Get CAN Rx FPS Send

Response: CAN1 Rx FPS = 740

<1> “CAN Rx FPS” 選項:

CAN1/CAN2 埠總線上接收到的資料流量(筆數/每秒)。

[Reset Module] 命令:

重置模組。當送出此命令給模組後，設定工具的“Module Configuration”畫面將會被關閉。

5. API 函式庫

使用者可透過 I-7565M-HS 應用程式介面函式庫, CAN_HS.dll 及 CAN_HS_x64.dll (於 Windows .Net 開發環境上, 需使用 CAN_HS_Net.dll 或 CAN_HS_Net_x64.dll) , 輕易地自行開發與 CAN Bus 通訊的網路程式。CAN_HS 函式庫與展示範例可從泓格科技網頁上取得：

函式庫路徑如下:

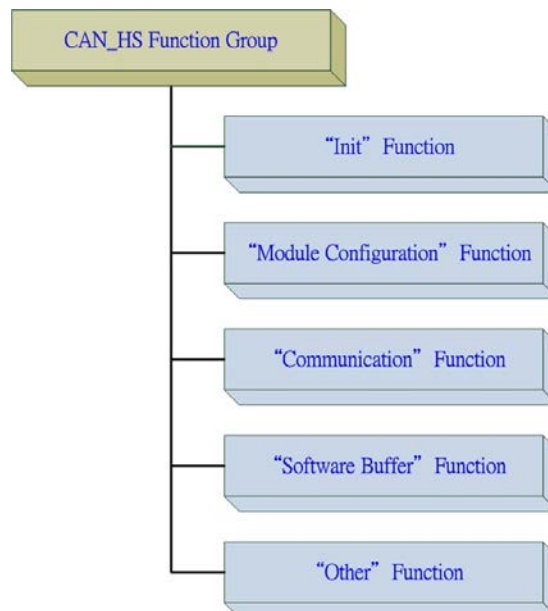
<https://www.icpdas.com/tw/download/show.php?num=3961&model=I-7565M-HS>

範例程式路徑如下:

<https://www.icpdas.com/tw/download/show.php?num=3960&model=I-7565M-HS>

5.1. API 函式庫概觀

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：



[Init Function]

該群組函式功能為搜尋及啟用/停用所有可正常使用的 I-7565M-HS 模組。

[Module Configuration Function]

該群組函式功能為讀取/設定 I-7565M-HS 模組參數及資訊。

[Communication Function]

該群組函式功能為透過 I-7565M-HS 模組來傳送及接收 CAN 總線訊息。

[Software Function]

所有的 CAN 總線傳送/接收的訊息均會先存放在 CAN_HS 函式庫的軟體緩衝區上。以便使用者透過此群組功能的函式來取得。

[Other Function]

該群組函式功能為取得 CAN_HS 函式庫資訊或提供使用者撰寫程式上的協助。

5.2. API 函式庫功能表

CAN_HS 提供的函式庫詳列於下表。

“Init” 功能表		
編號	函式名稱	功能描述
1	CANHS_ScanDevice	搜尋電腦上所有支援的設備
2	CANHS_ListDevice	將所有搜尋到的設備 pid/vid 資訊條列至資料表上
3	CANHS_OpenDevice	經由所選擇之設備 pid/vid 資訊來開啟設備
4	CANHS_CloseDevice	停用所選擇之設備

“Module Configuration” 功能表		
編號	函式名稱	功能描述
1	CANHS_SetCANOPMode	在指定的 CAN 埠上，設定 CAN 啟用/監聽等操作模式
2	CANHS_GetCANOPMode	在指定的 CAN 埠上，取得模組 CAN 啟用/監聽等操作模式
3	CANHS_SetCANBaudSP	在指定的 CAN 埠上，設定模組實際的 CAN 鮑率及取樣點設定值
4	CANHS_GetCANBaudSP	在指定的 CAN 埠上，取得模組實際的 CAN 鮑率及取樣點設定值
5	CANHS_GetCANBaudSPEEP	在指定的 CAN 埠上，取得模組 EEPROM 中的 CAN 鮑率及取樣點設定值
6	CANHS_SetCANFilter	在指定的 CAN 埠上，設定模組的硬體 CAN ID 過濾器設定參數
7	CANHS_GetCANFilter	在指定的 CAN 埠上，取得模組的硬體 CAN ID 過濾器設定參數
8	CANHS_SetCANWhiteListMode	在指定的 CAN 埠上，設定模組韌體的 CAN ID 白名單過濾器操作模式
9	CANHS_GetCANWhiteListMode	在指定的 CAN 埠上，取得模組韌體的 CAN ID 白名單過濾器操作模式

10	CANHS_SetCANBlackListMode	在指定的 CAN 埠上，設定模組韌體的 CAN ID 黑名單過濾器操作模式
11	CANHS_GetCANBlackListMode	在指定的 CAN 埠上，取得模組韌體的 CAN ID 黑名單過濾器操作模式
12	CANHS_SetCANWhiteListCANID	在指定的 CAN 埠上，設定模組韌體的 CAN ID 白名單過濾器設定值
13	CANHS_GetCANWhiteListCANID	在指定的 CAN 埠上，取得模組韌體的 CAN ID 白名單過濾器設定值
14	CANHS_SetCANBlackListCANID	在指定的 CAN 埠上，設定模組韌體的 CAN ID 黑名單過濾器設定值
15	CANHS_GetCANBlackListCANID	在指定的 CAN 埠上，取得模組韌體的 CAN ID 黑名單過濾器設定值
16	CANHS_GetCANStatus	在指定的 CAN 埠上，取得 CAN 總線的狀態

“Communication” 功能表		
編號	函式名稱	功能描述
1	CANHS_SetCANTxMsg	傳送一筆 CAN 訊息至所選擇的 CAN 埠的傳送緩衝區中
2	CANHS_GetCANRxMsg	由所選擇的 CAN 埠的接收緩衝區中，取得一筆接收到的 CAN 訊息
3	CANHS_SetCANHWSendMode	啟用/停用模組在指定的 CAN 埠上，利用模組硬體時鐘，周期性的傳送 CAN 訊息
4	CANHS_GetCANHWSendMode	讀取模組在指定的 CAN 埠上，利用模組硬體時鐘，周期性傳送 CAN 訊息的啟用/停用模式
5	CANHS_SetCANHWSendMsg	設定模組在指定的 CAN 埠上，利用模組硬體時鐘，周期性傳送 CAN 訊息的資料內容
6	CANHS_GetCANRxFramePerSec	取得模組在指定的 CAN 埠上，CAN 總線的訊息接收流量值

“Software Buffer” 功能表

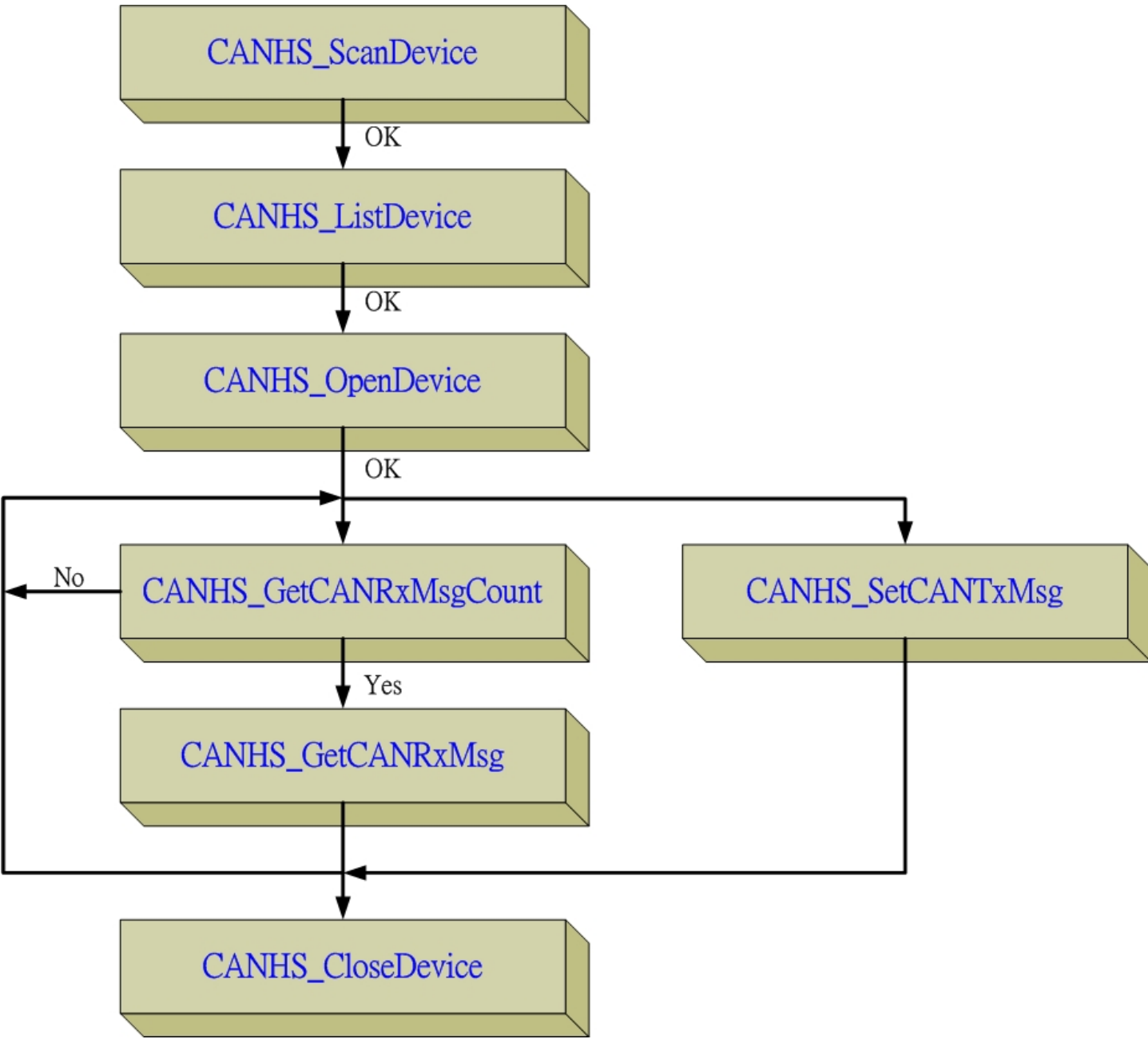
編號	函式名稱	功能描述
1	CANHS_GetCANRxMsgCount	取得模組在指定的 CAN 埠上，接收緩衝區內的 CAN 訊息筆數
2	CANHS_ClearCANRxBuf	清除模組在指定的 CAN 埠上，接收緩衝區內的所有 CAN 訊息
3	CANHS_ClearCANTxBuf	清除模組在指定的 CAN 埠上，傳送緩衝區內的所有 CAN 訊息

“Other” 功能表

編號	函式名稱	功能描述
1	CANHS_GetDllVersion	取得函式庫的版本資訊
2	CANHS_GetFwVer	取得模組的韌體版本資訊
3	CANHS_GetModuleStatus	取得模組的硬體狀態資訊
4	CANHS_ResetModule	重置模組

5.3. API 函式庫使用流程

下圖是使用者使用 CAN_HS 函式庫來開發 CAN 總線程式的基本控制流程圖。



5.4. 模組初始化功能函式

該群組函式功能為搜尋及啟用/停用所有可正常使用的 I-7565M-HS 模組。

5.4.1 CANHS_ScanDevice

此函式功能為搜尋電腦上所有可正常使用的 I-7565M-HS 模組。

語法:

C++

```
int CANHS_ScanDevice(void);
```

C#

```
Int32 CANHS_ScanDevice();
```

參數:

無。

回傳值:

回傳值為 0 時，代表成功；為其它值時，代表失敗。

5.4.2 CANHS_ListDevice

此函式功能在於將所有搜尋到的 I-7565M-HS 設備的 PID (product ID) · BID (board ID) 資訊條列至 PID/VID 資料表上。函式庫最多支援 8 個 I-7565M-HS 設備。

語法:

C++

```
BYTE CANHS_ListDevice(WORD* o_wPID, DWORD* o_dwBID);
```

C#

```
Byte CANHS_ListDevice(UInt16[] o_wPID, UInt32[] o_dwBID);
```

參數:

***o_wPID**

[out] 此指標指向一個最多可存放 8 筆搜尋到的模組 PID(product ID) 資訊的資料陣列。

***o_dwBID**

[out] 此指標指向一個最多可存放 8 筆搜尋到的模組 BID(board ID) 資訊的資料陣列。

回傳值:

回傳函式庫所搜尋得的模組個數。

5.4.3 CANHS_OpenDevice

此函式功能在於經由所選擇之設備 pid/vid 資訊來開啟設備。當設備成功開啟時，使用者會得到一組設備編號，使用者可使用此設備編號來使用“Communication”函式庫群組內的 CAN 資料收送函式。

語法:

C++

```
int CANHS_OpenDevice(WORD *o_wDevice_id, WORD i_wpid, DWORD i_wbid);
```

C#

```
Int32 CANHS_OpenDevice(out UInt16 o_wDevice_id, UInt16 i_wpid, UInt32 i_wbid);
```

參數:

***o_wDevice_id**

[out] 此指標指向一個可存放 經由模組 PID (product ID)/ BID (board ID)所成功開啟的設備編號參數位址。

i_wpid

[in] 欲開啟的 I-7565M-HS 設備 PID (product ID)。

i_wbid

[in] 欲開啟的 I-7565M-HS 設備 BID (board ID)。

回傳值:

回傳值為 0 時，代表成功；為其它值時，代表失敗。

5.4.4 CANHS_CloseDevice

此函式在停用所選擇之 I-7565M-HS 設備。在改設被關閉後，該設備在函式庫所使用之資源將會被釋放掉。

語法:

C++

```
int CANHS_CloseDevice(WORD i_wDevice_id);
```

C#

```
Int32 CANHS_CloseDevice(UInt16 i_wDevice_id);
```

參數:

i_wDevice_id

[in] 指定的 I-7565M-HS 設備編號。

回傳值:

回傳值為 0 時，代表成功；為其它值時，代表失敗。

5.5. 模組設定功能函式

該群組函式功能為讀取/設定 I-7565M-HS 模組的操作模式、CAN 總線通訊參數、CAN 總線訊息過濾等設定參數及資訊。

5.5.1 CANHS_SetCANOPMode

此功能函式在指定的 CAN 埠上，設定 CAN 埠啟用/監聽等操作模式。

語法:

C++
<pre>int CANHS_SetCANOPMode(WORD i_wDevice_id, BYTE i_byCANPort, WORD i_wEnable, WORD i_wMode);</pre>
C#
<pre>Int32 CANHS_SetCANOPMode(UInt16 i_wDevice_id, Byte i_byCANPort, UInt16 i_wEnable, UInt16 i_wMode);</pre>

參數:

i_wDevice_id

[in] 指定的 I-7565M-HS 設備編號。

i_byCANPort

[in] 指定的設備 CAN 埠編號。

i_wEnable

[in] 啟用/停用所指定的 CAN 埠功能。

0: 停用, 1: 啟用。

i_wMode

[in] 設定所指定的 CAN 埠功能為一般/監聽模式。

0: 一般模式, 可以正常收送 CAN 總線訊息。

1: 監聽模式, 只可以接收 CAN 總線訊息

回傳值:

回傳值為 0 時，代表成功；為其它值時，代表失敗。

5.5.2 CANHS_GetCANOPMode

此功能函式在指定的 CAN 埠上，讀取 CAN 埠啟用/監聽等操作模式。

語法:

C++
<pre>int CANHS_GetCANOPMode(WORD i_wDevice_id, BYTE i_byCANPort, WORD *o_wEnable, WORD *o_wMode);</pre>
C#
<pre>Int32 CANHS_GetCANOPMode(UInt16 i_wDevice_id, Byte i_byCANPort, out UInt16 o_wEnable, out UInt16 o_wMode);</pre>

參數:

i_wDevice_id

[in] 指定的 I-7565M-HS 設備編號。

i_byCANPort

[in] 指定的設備 CAN 埠編號。

****o_wEnable***

[out] 取得所指定的 CAN 埠上的啟用/停用設定功能。

0: 停用, 1: 啟用.

****o_wMode***

[out] 取得所指定的 CAN 埠上的一般/監聽模式設定功能。

0: 一般模式, 可以正常收送 CAN 總線訊息。

1: 監聽模式, 只可以接收 CAN 總線訊息

回傳值:

回傳值為 0 時，代表成功；為其它值時，代表失敗。

5.5.3 CANHS_SetCANBaudSP

此函式功能在指定的 CAN 埠上，設定模組實際欲使用的 CAN 鮑率及取樣點設定值。

語法:

C++
<pre>int CANHS_SetCANBaudSP(WORD i_wDevice_id, BYTE i_byCANPort, DWORD i_dwBR, DWORD i_dwSP);</pre>
C#
<pre>Int32 CANHS_SetCANBaudSP(UInt16 i_wDevice_id, Byte i_byCANPort, UInt32 i_dwBR, UInt32 i_dwSP);</pre>

參數:

i_wDevice_id

[in] 指定的 I-7565M-HS 設備編號。

i_byCANPort

[in] 指定的設備 CAN 埠編號。

i_dwBR

[in] 所指定的 CAN 埠鮑率設定值。

單位: bps (bit per second), 數值 1000000 表示 1000000 bps.

i_dwSP

[in] 所指定的 CAN 埠鮑率樣點設定值。

單位: 0.01%, 數值 7500 表示 75.00%

回傳值:

回傳值為 0 時，代表成功；為其它值時，代表失敗。

5.5.4 CANHS_GetCANBaudSP

此函式功能在取得目前所指定的 CAN 埠上，模組當下使用的 CAN 鮑率及取樣點設定值。

語法:

C++
<pre>int CANHS_GetCANBaudSP(WORD i_wDevice_id, BYTE i_byCANPort, DWORD *o_dwBR, DWORD *o_dwSP);</pre>
C#
<pre>Int32 CANHS_GetCANBaudSP(UInt16 i_wDevice_id, Byte i_byCANPort, out UInt32 o_dwBR, out UInt32 o_dwSP);</pre>

參數:

i_wDevice_id

[in] 指定的 I-7565M-HS 設備編號。

i_byCANPort

[in] 指定的設備 CAN 埠編號。

****o_dwBR***

[out] 取得目前所指定的 CAN 埠上，模組當下使用的 CAN 鮑率設定值。

單位: bps (bit per second), 數值 1000000 表示 1000000 bps.

****o_dwSP***

[out] 取得目前所指定的 CAN 埠上，模組當下使用的 CAN 鮑率取樣點設定值。

單位: 0.01%, 數值 7500 表示 75.00%

回傳值:

回傳值為 0 時，代表成功；為其它值時，代表失敗。

5.5.5 CANHS_GetCANBaudSPEEP

此函式功能在取得目前所指定的 CAN 埠上，模組存放在 EEPROM 的 CAN 鮑率及取樣點設定值。

語法:

C++
int CANHS_GetCANBaudSPEEP(WORD i_wDevice_id, BYTE i_byCANPort, DWORD *o_dwBR, DWORD *o_dwSP);
C#
Int32 CANHS_GetCANBaudSPEEP(UInt16 i_wDevice_id, Byte i_byCANPort, out UInt32 o_dwBR, out UInt32 o_dwSP);

參數:

i_wDevice_id

[in] 指定的 I-7565M-HS 設備編號。

i_byCANPort

[in] 指定的設備 CAN 埠編號。

****o_dwBR***

[out] 取得目前所指定的 CAN 埠上，模組 EEPROM 中的 CAN 鮑率設定值。
單位: bps (bit per second), 數值 1000000 表示 1000000 bps.

****o_dwSP***

[out] 取得目前所指定的 CAN 埠上，模組 EEPROM 中的 CAN 鮑率取樣點設定值。
單位: 0.01%, 數值 7500 表示 75.00%

回傳值:

回傳值為 0 時，代表成功；為其它值時，代表失敗。

5.5.6 CANHS_SetCANFilter

此函式功能在指定的 CAN 埠上，設定模組的硬體 CAN ID 過濾器設定參數。

語法:

C++
<pre>int CANHS_SetCANFilter(WORD i_wDevice_id, BYTE i_byCANPort, BYTE i_byMode, DWORD i_dwMask, DWORD i_dwArbitration);</pre>
C#
<pre>Int32 CANHS_SetCANFilter (UInt16 i_wDevice_id, Byte i_byCANPort, Byte i_byMode, UInt32 i_dwMask, UInt32 i_dwArbitration);</pre>

參數:

i_wDevice_id

[in] 指定的 I-7565M-HS 設備編號。

i_byCANPort

[in] 指定的設備 CAN 埠編號。

i_byMode

[in] 設定所指定的 CAN 埠的 CAN ID 訊息模式。可以設定成 2.0A (11-bit CAN ID), 或 2.0B (29-bit CAN ID)。

0: 2.0A (11-bit CAN ID)

1: 2.0B (29-bit CAN ID)

i_dwMask

[in] CAN 訊息 ID 遮罩位元。需對應 [***i_dwArbitration***] 參數使用。位元值為 0 表示不需要比對 [***i_dwArbitration***] 參數中相對應的 CAN ID 位元。位元值為 1 表示此 CAN 訊息 ID 需與 [***i_dwArbitration***] 參數中相對應的 CAN ID 位元相同；不相同的話表是將過濾掉此 CAN 訊息 ID。

i_dwArbitration

[in] CAN 訊息 ID 仲裁位元。要用來做為過濾比對用的 CAN 訊息 ID。

[範例]

1. 所有的 CAN ID 均不過濾，全部接收。

[i_byMode] = 0

[i_dwMask] = 0x000

[i_dwArbitration] = 0x000

2. 除了 0x123 CAN ID 外，其他的 CAN ID 均被過濾掉，均不接收。

[i_byMode] = 0

[i_dwMask] = 0x7FF

[i_dwArbitration] = 0x123

3. 除了 0x100 ~ 0x10F 範圍之 CAN ID 外，其他的 CAN ID 均被過濾掉，均不接收。

[i_byMode] = 0

[i_dwMask] = 0x7F0

[i_dwArbitration] = 0x100

回傳值:

回傳值為 0 時，代表成功；為其它值時，代表失敗。

5.5.7 CANHS_GetCANFilter

此函式功能在指定的 CAN 埠上，取得模組的硬體 CAN ID 過濾器設定參數。

語法:

C++

```
int CANHS_GetCANFilter(WORD i_wDevice_id, BYTE i_byCANPort, BYTE  
*o_byMode, DWORD *o_dwMask, DWORD *o_dwArbitration);
```

C#

```
Int32 CANHS_GetCANFilter (UInt16 i_wDevice_id, Byte i_byCANPort, Byte  
*o_byMode, UInt32 *o_dwMask, UInt32 *o_dwArbitration);
```

參數:

i_wDevice_id

[in] 指定的 I-7565M-HS 設備編號。

i_byCANPort

[in] 指定的設備 CAN 埠編號。

****o_byMode***

[in] Mode of CAN ID in the assigned CAN port of the I-7565M-HS device.
0: 2.0A (11-bit CAN ID)
1: 2.0B (29-bit CAN ID)

****o_dwMask***

[out] CAN 訊息 ID 遮罩位元。需對應 [***o_dwArbitration***] 參數使用。位元值為 0 表示不需要比對[***o_dwArbitration***]參數中相對應的 CAN ID 位元。位元值為 1 表示此 CAN 訊息 ID 需與[***o_dwArbitration***]參數中相對應的 CAN ID 位元相同；不相同的話表是將過濾掉此 CAN 訊息 ID。

****o_dwArbitration***

[out] CAN 訊息 ID 仲裁位元。要用來做為過濾比對用的 CAN 訊息 ID。

回傳值:

回傳值為 0 時，代表成功；為其它值時，代表失敗。

5.5.8 CANHS_SetCANWhiteListMode

此函式功能在指定的 CAN 埠上，啟用/停用模組韌體 CAN ID 白名單過濾功能。當啟用此功能時，模組只會接收白名單上有支援的 CAN 訊息。

語法:

C++
<pre>int CANHS_SetCANWhiteListMode(WORD i_wDevice_id, BYTE i_byCANPort, BYTE i_byMode);</pre>
C#
<pre>Int32 CANHS_SetCANWhiteListMode(UInt16 i_wDevice_id, Byte i_byCANPort, Byte i_byMode);</pre>

參數:

i_wDevice_id

[in] 指定的 I-7565M-HS 設備編號。

i_byCANPort

[in] 指定的設備 CAN 埠編號。

i_byMode

[in] 啟用/停用模組韌體 CAN ID 白名單過濾功能。

0: 停用模組韌體 CAN ID 白名單過濾功能。

1: 啟用模組韌體 CAN ID 白名單過濾功能。

回傳值:

回傳值為 0 時，代表成功；為其它值時，代表失敗。

5.5.9 CANHS_GetCANWhiteListMode

此函式功能在指定的 CAN 埠上，取得模組韌體 CAN ID 白名單過濾功能的啟用/停用設定值。

語法:

C++
<pre>int CANHS_GetCANWhiteListMode(WORD i_wDevice_id, BYTE i_byCANPort, BYTE *o_byMode);</pre>
C#
<pre>Int32 CANHS_GetCANWhiteListMode(UInt16 i_wDevice_id, Byte i_byCANPort, out Byte o_byMode);</pre>

參數:

i_wDevice_id

[in] 指定的 I-7565M-HS 設備編號。

i_byCANPort

[in] 指定的設備 CAN 埠編號。

****o_byMode***

[out] 此指標指向模組韌體 CAN ID 白名單過濾功能的啟用/停用設定值。

0: 模組韌體 CAN ID 白名單過濾功能為停用狀態。

1: 模組韌體 CAN ID 白名單過濾功能為啟用狀態。

回傳值:

回傳值為 0 時，代表成功；為其它值時，代表失敗。

5.5.10 CANHS_SetCANBlackListMode

此函式功能在指定的 CAN 埠上，啟用/停用模組韌體 CAN ID 黑名單過濾功能。當啟用此功能時，模組會過濾掉黑名單上所列之 CAN 訊息。

語法:

C++
<pre>int CANHS_SetCANBlackListMode(WORD i_wDevice_id, BYTE i_byCANPort, BYTE i_byMode);</pre>
C#
<pre>Int32 CANHS_SetCANBlackListMode(UInt16 i_wDevice_id, Byte i_byCANPort, Byte i_byMode);</pre>

參數:

i_wDevice_id

[in] 指定的 I-7565M-HS 設備編號。

i_byCANPort

[in] 指定的設備 CAN 埠編號。

i_byMode

[in] 啟用/停用模組韌體 CAN ID 黑名單過濾功能。

0: 停用模組韌體 CAN ID 黑名單過濾功能。

1: 啟用模組韌體 CAN ID 黑名單過濾功能。

回傳值:

回傳值為 0 時，代表成功；為其它值時，代表失敗。

5.5.11 CANHS_GetCANBlackListMode

此函式功能在指定的 CAN 埠上，取得模組韌體 CAN ID 黑名單過濾功能的啟用/停用設定值。

語法:

C++

```
int CANHS_GetCANBlackListMode(WORD i_wDevice_id, BYTE i_byCANPort, BYTE *o_byMode);
```

C#

```
Int32 CANHS_GetCANBlackListMode(UInt16 i_wDevice_id, Byte i_byCANPort, out Byte o_byMode);
```

參數:

i_wDevice_id

[in] 指定的 I-7565M-HS 設備編號。

i_byCANPort

[in] 指定的設備 CAN 埠編號。

****o_byMode***

[out] 此指標指向模組韌體 CAN ID 黑名單過濾功能的啟用/停用設定值。

0: 模組韌體 CAN ID 黑名單過濾功能為停用狀態。

1: 模組韌體 CAN ID 黑名單過濾功能為啟用狀態。

回傳值:

回傳值為 0 時，代表成功；為其它值時，代表失敗。

5.5.12 CANHS_SetCANWhiteListCANID

此函式功能在指定的 CAN 埠上，設定欲使用之 CAN ID 至模組韌體 CAN ID 白名單過濾器中。當模組韌體 CAN ID 白名單過濾器功能有啟用時，模組接收到的 CAN 訊息 ID 如果與模組韌體 CAN ID 白名單過濾器中的 CAN ID 相符時，模組將會接收此筆接收到的 CAN 訊息。

語法:

C++

```
int CANHS_SetCANWhiteListCANID(WORD i_wDevice_id, BYTE i_byCANPort, WORD i_wCIDNum, DWORD *i_dwCID);
```

C#

```
Int32 CANHS_SetCANWhiteListCANID(UInt16 i_wDevice_id, Byte i_byCANPort, UInt16 i_wCIDNum, [In,Out] UInt32[] i_dwCID);
```

參數:

i_wDevice_id

[in] 指定的 I-7565M-HS 設備編號。

i_byCANPort

[in] 指定的設備 CAN 埠編號。

i_wCIDNum

[in] 在指定的 CAN 埠上，欲設定至模組韌體 CAN ID 白名單過濾器的 CAN ID 數量。模組每個 CAN 埠最多支援 128 組 CAN ID。當此參數設定為 0 時，表示清除模組韌體 CAN ID 白名單過濾器內的所有 CAN ID。

****i_dwCID***

[in/out] 此指標指向在指定的 CAN 埠上，欲設定至模組韌體 CAN ID 白名單過濾器的 CAN ID 陣列。

回傳值:

回傳值為 0 時，代表成功；為其它值時，代表失敗。

5.5.13 CANHS_GetCANWhiteListCANID

此函式功能在指定的 CAN 埠上，取得模組韌體 CAN ID 白名單過濾器中所設定的 CAN ID。

語法:

C++
<pre>int CANHS_GetCANWhiteListCANID(WORD i_wDevice_id, BYTE i_byCANPort, WORD *o_wCIDNum, DWORD *o_dwCID);</pre>
C#
<pre>Int32 CANHS_GetCANWhiteListCANID(UInt16 i_wDevice_id, Byte i_byCANPort, out UInt16 o_wCIDNum, [In, Out] UInt32[] o_dwCID);</pre>

參數:

i_wDevice_id

[in] 指定的 I-7565M-HS 設備編號。

i_byCANPort

[in] 指定的設備 CAN 埠編號。

****o_wCIDNum***

[out] 此指標指向在指定的 CAN 埠上，模組韌體 CAN ID 白名單過濾器中所設定的 CAN ID 總數。

****o_dwCID***

[in/out] 此指標指向在指定的 CAN 埠上，模組韌體 CAN ID 白名單過濾器中所設定的 CAN ID 陣列初始位址。

回傳值:

回傳值為 0 時，代表成功；為其它值時，代表失敗。

5.5.14 CANHS_SetCANBlackListCANID

此函式功能在指定的 CAN 埠上，設定欲使用之 CAN ID 至模組韌體 CAN ID 黑名單過濾器中。當模組韌體 CAN ID 黑名單過濾器功能有啟用時，模組接收到的 CAN 訊息 ID 如果與模組韌體 CAN ID 黑名單過濾器中的 CAN ID 相符時，模組將會捨棄並且過濾掉此筆接收到的 CAN 訊息。

語法:

C++

```
int CANHS_SetCANBlackListCANID(WORD i_wDevice_id, BYTE i_byCANPort, WORD i_wCIDNum, DWORD *i_dwCID);
```

C#

```
Int32 CANHS_SetCANBlackListCANID(UInt16 i_wDevice_id, Byte i_byCANPort, UInt16 i_wCIDNum, [In,Out] UInt32[] i_dwCID);
```

參數:

i_wDevice_id

[in] 指定的 I-7565M-HS 設備編號。

i_byCANPort

[in] 指定的設備 CAN 埠編號。

i_wCIDNum

[in] 在指定的 CAN 埠上，欲設定至模組韌體 CAN ID 黑名單過濾器的 CAN ID 數量。模組每個 CAN 埠最多支援 128 組 CAN ID。當此參數設定為 0 時，表示清除模組韌體 CAN ID 黑名單過濾器內的所有 CAN ID。

****i_dwCID***

[in/out] 此指標指向在指定的 CAN 埠上，欲設定至模組韌體 CAN ID 黑名單過濾器的 CAN ID 陣列。

回傳值:

回傳值為 0 時，代表成功；為其它值時，代表失敗。

5.5.15 CANHS_GetCANBlackListCANID

This function is used to get CAN IDs from blacklist table of CAN message ID software filter in the assigned CAN port of the I-7565M-HS device.

語法:

C++
<pre>int CANHS_GetCANBlackListCANID(WORD i_wDevice_id, BYTE i_byCANPort, WORD *o_wCIDNum, DWORD *o_dwCID);</pre>
C#
<pre>Int32 CANHS_GetCANBlackListCANID(UInt16 i_wDevice_id, Byte i_byCANPort, out UInt16 o_wCIDNum, [In,Out] UInt32[] o_dwCID);</pre>

參數:

i_wDevice_id

[in] 指定的 I-7565M-HS 設備編號。

i_byCANPort

[in] 指定的設備 CAN 埠編號。

****o_wCIDNum***

[out] 此指標指向在指定的 CAN 埠上，模組韌體 CAN ID 黑名單過濾器中所設定的 CAN ID 總數。

****o_dwCID***

[in/out] 此指標指向在指定的 CAN 埠上，模組韌體 CAN ID 白名單過濾器中所設定的 CAN ID 陣列初始位址。

回傳值:

回傳值為 0 時，代表成功；為其它值時，代表失敗。

5.5.16 CANHS_GetCANStatus

此函式功能在於取得模組指定的 CAN 埠上，CAN 總線狀態、CAN 總線傳送/接收的錯誤發生次數、以及傳送/接收資料緩衝區是否溢位等狀態。

語法:

C++

int CANHS_GetCANStatus(WORD i_wDevice_id, BYTE i_byCANPort, DWORD *o_dwCANStatus, DWORD *o_dwErrCnt, DWORD *o_dwBufStatus);

C#

Int32 CANHS_GetCANStatus(UInt16 i_wDevice_id, Byte i_byCANPort, out UInt32 o_dwCANStatus, out UInt32 o_dwErrCnt, out UInt32 o_dwBufStatus);

參數:

- i_wDevice_id**

[in] 指定的 I-7565M-HS 設備編號。
- i_byCANPort**

[in] 指定的設備 CAN 埠編號。
- * o_dwCANStatus**

[out] 指定的 CAN 埠的 CAN 總線狀態，詳細內容請參考附錄 7.3 “CAN Status” 定義。
- * o_dwErrCnt**

[out] 指定的 CAN 埠的 CAN 總線傳送/接收的錯誤發生次數，詳細內容請參考附錄 7.4 “CAN Error Counter” 定義。
- *o_dwBufStatus**

[out] 指定的 CAN 埠的傳送/接收資料緩衝區是否溢位等狀態。

位元	符號	數值	功能描述
0	RX		CAN1/CAN2埠接收緩衝區的狀態
		0	接收緩衝區狀態正常
		1	接收緩衝區資料溢位

1	TX		CAN1/CAN2埠傳送緩衝區的狀態
		0	傳送緩衝區狀態正常
		1	傳送緩衝區資料溢位
31:2		-	保留

回傳值:

回傳值為 0 時，代表成功；為其它值時，代表失敗。

5.5.17 CANHS_ResetModule

此函式功能在於重置 I-7565M-HS 模組。

語法:

C++
Int CANHS_ResetModule(WORD i_wDevice_id);
C#
Int32 CANHS_ResetModule(UInt16 i_wDevice_id);

參數:

i_wDevice_id

[in] 指定的 I-7565M-HS 設備編號。

回傳值:

回傳值為 0 時，代表成功；為其它值時，代表失敗。

5.6. 模組通訊功能函式

該群組函式功能為透過 I-7565M-HS 模組來傳送及接收 CAN 總線訊息。

5.6.1 CANHS_SetCANTxMsg

此函式功能在於傳送一筆 CAN 訊息至所選擇的 CAN 埠的傳送緩衝區中。

語法:

C++
int CANHS_SetCANTxMsg(WORD i_wDevice_id, BYTE i_byCANPort, BYTE i_byMode, DWORD i_dwID, BYTE i_byRTR, BYTE i_byDlen, BYTE *i_byData);
C#
Int32 CANHS_SetCANTxMsg(UInt16 i_wDevice_id, Byte i_byCANPort, Byte i_byMode, UInt32 i_dwID, Byte i_byRTR, Byte i_byDlen, [In, Out] Byte[] i_byData);

參數:

i_wDevice_id

[in] 指定的 I-7565M-HS 設備編號。

i_byCANPort

[in] 指定的設備 CAN 埠編號。

i_byMode

[in] 欲傳送的 CAN 訊息種類。

0: 2.0A, 11-bit CAN ID

1: 2.0B, 29-bit CAN ID

i_dwID

[in] 欲傳送的 CAN 訊息 ID 參數。

合理範圍:

2.0A mode ➔ 0x000 ~ 0x7FF

2.0B mode ➔ 0x00000000 ~ 0x1FFFFFFF

i_byRTR

[in] 欲傳送的 CAN 訊息 RTR (Remote Transmission Request) 參數。

0: no RTR

1: RTR

i_byDlen

[in] 欲傳送的 CAN 訊息 DLC 資料長度參數。

合理範圍: 0 ~ 8。

****i_byData***

[in/out] 該指標用於取得欲傳送的 8 位元組 CAN 資料參數。

回傳值:

回傳值為 0 時，代表成功；為其它值時，代表失敗。

5.6.2 CANHS_GetCANRxMsg

此函式功能在於由所選擇的 CAN 埠的接收緩衝區中，取得一筆接收到的 CAN 訊息。

語法：

C++
<pre>int CANHS_GetCANRxMsg(WORD i_wDevice_id, BYTE i_byCANPort, BYTE* o_byType, BYTE* o_byMode, DWORD* o_dwID, BYTE* o_byRTR, BYTE* o_byDlen, BYTE *o_byData, DWORD *o_dw_TimeStamp_s, DWORD *o_dw_TimeStamp_us);</pre>
C#
<pre>Int32 CANHS_GetCANRxMsg(UInt16 i_wDevice_id, Byte i_byCANPort, out Byte o_byType, out Byte o_byMode, out UInt32 o_dwID, out Byte o_byRTR, out Byte o_byDlen, [In, Out] Byte[] o_byData, out UInt32 o_dw_TimeStamp_s, out UInt32 o_dw_TimeStamp_us);</pre>

參數：

i_wDevice_id

[in] 指定的 I-7565M-HS 設備編號。

i_byCANPort

[in] 指定的設備 CAN 埠編號。

****o_byType***

[out] 接收到的訊息格式。

0: 接收到一筆 2.0A/2.0B CAN 訊息。

2: 表示接收到一筆 CAN 錯誤事件訊息。

錯誤事件訊息格式：

Mode: 1 (2.0B CAN 訊息格式)

ID: 0xEEEEEEEE

RTR: 0 (No RTR)

Dlen: 0x08

Data: D0~D3 → CAN Bus status in little-endian format

(請參考附錄 7.3 的 “CAN Status” 定義)

D4~D7 → CAN Bus error counter in little-endian format
(請參考附錄 7.4 的 “CAN Error Counter” 定義)

***o_byMode**

[out] CAN 訊息種類。
0: 2.0A, 11-bit CAN ID
1: 2.0B, 29-bit CAN ID

***o_dwID**

[out] 接收到的 CAN 訊息 ID 參數。
2.0A CAN ID 訊息 → 0x000 ~ 0x7FF
2.0B CAN ID 訊息 → 0x00000000 ~ 0x1FFFFFFF
CAN 錯誤事件訊息 → 0xEEEEEE

***o_byRTR**

[out] 接收到的 CAN 訊息 RTR (Remote Transmission Request) 參數。
0: no RTR
1: RTR

***o_byDlen**

[in] 接收到的 CAN 訊息 DLC 資料長度參數。
合理範圍: 0 ~ 8。

***o_byData**

[in/out] 該指標用於取得接收到的 CAN 訊息 8 位元組 CAN 資料參數。

***o_dw_TimeStamp_s**

[out] 接收到此訊息時的時間戳記
單位: 秒

***o_dw_TimeStamp_us**

[out] 接收到此訊息時的時間戳記
單位: 微秒

回傳值:

回傳值為 0 時，代表成功；為其它值時，代表失敗。

5.6.3 CANHS_SetCANHWSendMode

此函式用來啟用/停用模組在指定的 CAN 埠上，利用模組硬體時鐘，周期性的傳送 CAN 訊息。使用模組硬體時鐘來傳送 CAN 訊息會比使用電腦端的軟體時鐘還要實時。

語法:

C++
<pre>int CANHS_SetCANHWSendMode(WORD i_wDevice_id, BYTE i_byCANPort, BYTE i_byMode);</pre>
C#
<pre>Int32 CANHS_SetCANHWSendMode(UInt16 i_wDevice_id, Byte i_byCANPort, Byte i_byMode);</pre>

參數:

i_wDevice_id

[in] 指定的 I-7565M-HS 設備編號。

i_byCANPort

[in] 指定的設備 CAN 埠編號。

i_byMode

[in] 啟用/停用模組在指定的 CAN 埠上，利用模組硬體時鐘，周期性的傳送 CAN 訊息

0: 停用模組硬體時鐘，周期性的傳送 CAN 訊息

1: 啟用模組硬體時鐘，周期性的傳送 CAN 訊息

回傳值:

回傳值為 0 時，代表成功；為其它值時，代表失敗。

5.6.4 CANHS_GetCANHWSendMode

此函式用來取得模組在指定的 CAN 埠上，利用模組硬體時鐘，周期性傳送 CAN 訊息的啟用/停用模式。

語法:

C++
<pre>int CANHS_GetCANHWSendMode(WORD i_wDevice_id, BYTE i_byCANPort, BYTE *o_byMode);</pre>
C#
<pre>Int32 CANHS_GetCANHWSendMode(UInt16 i_wDevice_id, Byte i_byCANPort, out Byte o_byMode);</pre>

參數:

i_wDevice_id

[in] 指定的 I-7565M-HS 設備編號。

i_byCANPort

[in] 指定的設備 CAN 埠編號。

****o_byMode***

[out] 該指標用於取得目前是停用或啟用模組硬體時鐘傳送 CAN 訊息。

0: 表示目前為停用模組硬體時鐘傳送 CAN 訊息功能。

1: 表示目前為啟用模組硬體時鐘傳送 CAN 訊息功能。

回傳值:

回傳值為 0 時，代表成功；為其它值時，代表失敗。

5.6.5 CANHS_SetCANHWSendMsg

此函式用來設定模組在指定的 CAN 埠上，利用模組硬體時鐘，周期性傳送 CAN 訊息的資料內容。

語法:

```
C++
int CANHS_SetCANHWSendMsg(WORD i_wDevice_id, BYTE i_byCANPort, BYTE i_byMode, DWORD i_dwID, BYTE i_byRTR, BYTE i_byDlen, BYTE *i_byData, DWORD i_dwTimer, DWORD i_dwCounter);
```

```
C#
Int32 CANHS_SetCANHWSendMsg(UInt16 i_wDevice_id, Byte i_byCANPort, Byte i_byMode, UInt32 i_dwID, Byte i_byRTR, Byte i_byDlen, [In, Out] Byte[] i_byData, UInt32 i_dwTimer, UInt32 i_dwCounter);
```

參數:

i_wDevice_id

[in] 指定的 I-7565M-HS 設備編號。

i_byCANPort

[in] 指定的設備 CAN 埠編號。

i_byMode

[in] CAN 訊息種類。

0: 2.0A, 11-bit CAN ID

1: 2.0B, 29-bit CAN ID

i_dwID

[in] CAN 訊息 ID 參數。

合理範圍:

2.0A mode → 0x000 ~ 0x7FF

2.0B mode → 0x00000000 ~ 0x1FFFFFFF

i_byRTR

[in] CAN 訊息 RTR (Remote Transmission Request)參數。

0: no RTR

1: RTR

i_byDlen

[in] CAN 訊息 DLC 資料長度參數。

合理範圍: 0 ~ 8。

****i_byData***

[in/out] 該指標用於取得欲傳送的 8 位元組 CAN 資料參數。

i_dwTimer

[in] 使用模組硬體時鐘傳送該筆 CAN 訊息的時間周期。

單位: 100 微秒

i_dwCounter

[in] 使用模組硬體時鐘傳送該筆 CAN 訊息的次數。

回傳值:

回傳值為 0 時，代表成功；為其它值時，代表失敗。

5.6.6 CANHS_GetCANRxFramePerSec

此函式功能在於取得模組在指定的 CAN 埠上，CAN 總線的訊息接收流量值。

語法:

C++

```
int CANHS_GetCANRxFramePerSec(WORD i_wDevice_id, BYTE i_byCANPort, WORD *o_wRxFPS);
```

C#

```
Int32 CANHS_GetCANRxFramePerSec(UInt16 i_wDevice_id, Byte i_byCANPort, out UInt16 o_wRxFPS);
```

參數:

i_wDevice_id

[in] 指定的 I-7565M-HS 設備編號。

i_byCANPort

[in] 指定的設備 CAN 埠編號。

**o_wRxFPS*

[out] 取得指定的 CAN 埠每秒鐘所接收到的 CAN 訊息量，單位為 FPS。

回傳值:

回傳值為 0 時，代表成功；為其它值時，代表失敗。

5.7. 資料緩衝區功能函式

所有的 CAN 總線傳送/接收的訊息均會先存放在 CAN_HS 函式庫的軟體緩衝區上。以便使用者透過此群組功能的函式來取得。

5.7.1 CANHS_GetCANRxMsgCount

此函式功能在於取得所指定的 I-7565M-HS 模組 CAN 埠接收緩衝區內的 CAN 訊息數量。

語法:

C++
<pre>int CANHS_GetCANRxMsgCount(WORD i_wDevice_id, BYTE i_byCANPort, DWORD *o_dwCount);</pre>
C#
<pre>Int32 CANHS_GetCANRxMsgCount(UInt16 i_wDevice_id, Byte i_byCANPort, out UInt32 o_dwCount);</pre>

參數:

i_wDevice_id

[in] 指定的 I-7565M-HS 設備編號。

i_byCANPort

[in] 指定的設備 CAN 埠編號。

****o_dwCount***

[out] 取得指定的 CAN 埠接收緩衝區內的 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。

5.7.2 CANHS_ClearCANRxBuf

此函式功能在於清除模組在指定的 CAN 埠上，接收緩衝區內的所有 CAN 訊息。

語法:

C++

```
int CANHS_ClearCANRxBuf(WORD i_wDevice_id, BYTE i_byCANPort);
```

C#

```
Int32 CANHS_ClearCANRxBuf(UInt16 i_wDevice_id, Byte i_byCANPort);
```

參數:

i_wDevice_id

[in] 指定的 I-7565M-HS 設備編號。

i_byCANPort

[in] 指定的設備 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。

5.7.3 CANHS_ClearCANTxBuf

此函式功能在於清除模組在指定的 CAN 埠上，傳送緩衝區內的所有 CAN 訊息。

語法:

C++

```
int CANHS_ClearCANTxBuf(WORD i_wDevice_id, BYTE i_byCANPort);
```

C#

```
Int32 CANHS_ClearCANTxBuf(UInt16 i_wDevice_id, Byte i_byCANPort);
```

參數:

i_wDevice_id

[in] 指定的 I-7565M-HS 設備編號。

i_byCANPort

[in] 指定的設備 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。

5.8. 其他功能函式

該群組函式功能為取得 CAN_HS 函式庫資訊或提供使用者撰寫程式上的協助。

5.8.1 CANHS_GetDllVersion

此函式用來取得 CAN_HS 函式庫的版本。

語法:

C++

```
DWORD CANHS_GetDllVersion(void);
```

C#

```
UInt32 CANHS_GetDllVersion();
```

參數:

None.

回傳值:

回傳 CAN_HS 函式庫的版本。

數值 1000000 (十進制) ➔ 表示 CAN_HS 函式庫的版本為 v1.0.0.0

數值 1000113 (十進制) ➔ 表示 CAN_HS 函式庫的版本為 v1.0.1.13

5.8.2 CANHS_GetFwVer

此函式用來取得 I-7565M-HS 模組的韌體版本。

語法:

C++

```
Int CANHS_GetFwVer(WORD i_wDevice_id, WORD* o_wFwVer);
```

C#

```
Int32 CANHS_GetFwVer(UInt16 i_wDevice_id, out UInt16 o_wFwVer);
```

參數:

i_wDevice_id

[in] 指定的 I-7565M-HS 設備編號。

**o_wFwVer*

[out] 模組韌體版本。

數值 100 (十進制) → 韌體版本: v1.00

回傳值:

回傳值為 0 時，代表成功；為其它值時，代表失敗。

5.8.3 CANHS_GetModuleStatus

此函式用來取得 I-7565M-HS 模組的硬體狀態。

語法:

C++

Int CANHS_GetModuleStatus(WORD i_wDevice_id, DWORD* o_dwStatus);

C#

Int32 CANHS_GetModuleStatus(UInt16 i_wDevice_id, out UInt32 o_dwStatus);

參數:

i_wDevice_id
[in] 指定的 I-7565M-HS 設備編號。

*** o_dwStatus**
[out] I-7565M-HS 設備的硬體狀態。

位元	數值	功能描述
0		模組的USB初始化狀態
	0	初始化USB正常
	1	初始化USB異常
31:1	-	保留

回傳值:

回傳值為 0 時，代表成功；為其它值時，代表失敗。

5.9. 函式回傳碼

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

回傳碼 (16 進制)	描述
0x0	No error
0x1	OP field of the configuration command error
0x2	FC field of the configuration command error
0x3	DL field of the configuration command error
0x4	Fail to write data into device
0x10001	Invalid device
0x10002	Device already in used
0x10003	Device not exist
0x10004	Get device information error
0x10005	Invalid USB package size
0x10006	Write file fail
0x10007	USB Tx buffer overflow
0x1000A	Exceed maximum supported USB device
0x1000B	USB device not open
0x10100	Communication timeout
0x10101	Invalid CAN port number
0x10102	No data in CAN received buffer
0x10103	CAN transmitted buffer overflow
0x10104	Exceed maximum supported CAN filter IDs

6. 韌體更新

請參考下列步驟來更新模組的韌體。

步驟 1: 將 I-7565M-HS 模組上方的 "Init." 開關調整至 "ON" 位置，然後使用產品盒中的 "CA-USB15" USB 線連接模組及使用者的電腦(PC)。

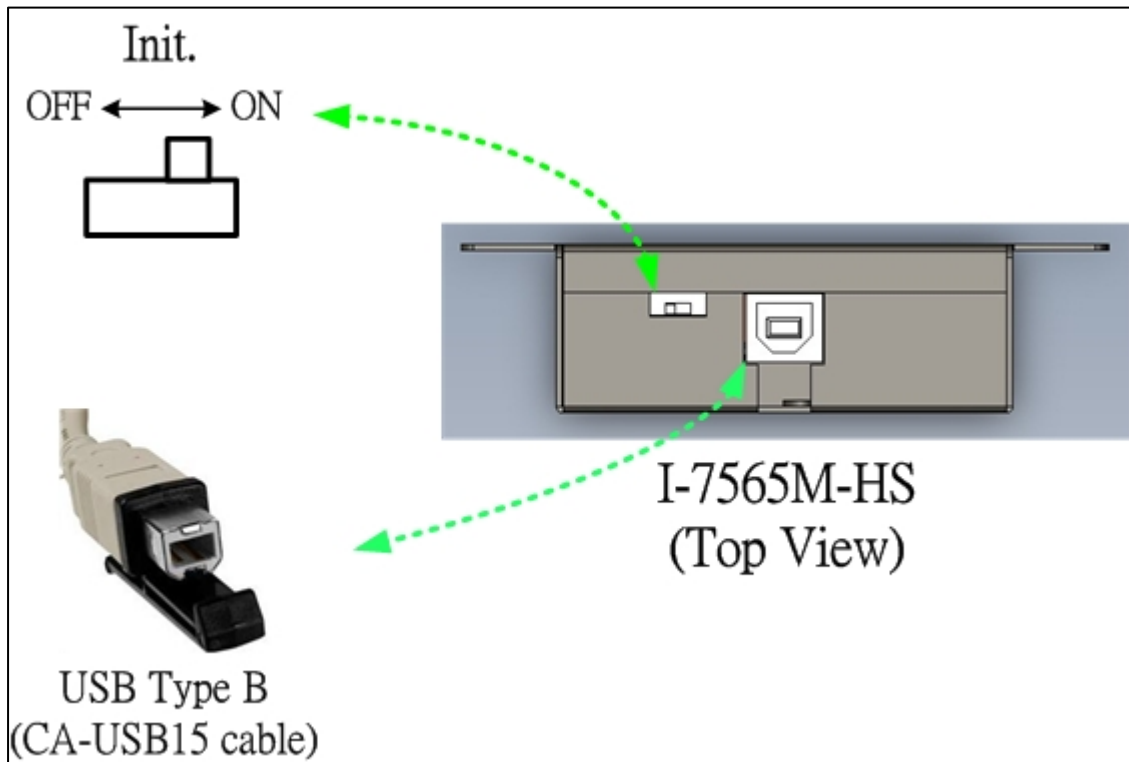
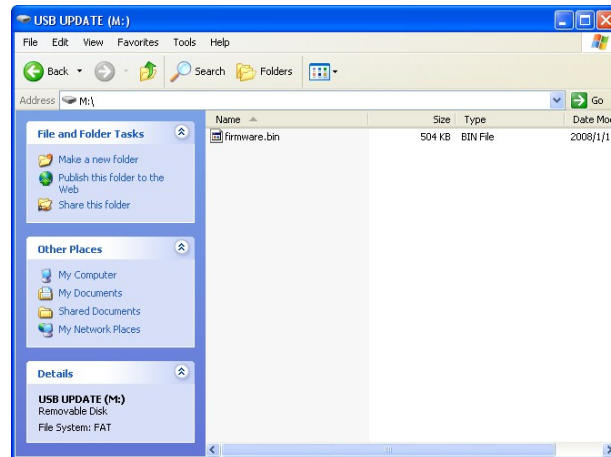


圖 6-1 指撥開關調整及 USB 接線

步驟 2: 然後模組將會進入“韌體更新模式”，在此模式下，模組 LED 燈將會依 Power, MS, CAN1_ST, CAN2_ST, CAN2, CAN1 的順序輪循環閃爍。同時，在 PC 上會虛擬出一 USB 磁碟槽，如下圖所示(範例: M:)\。使用者可透過此磁碟槽來更新模組韌體。



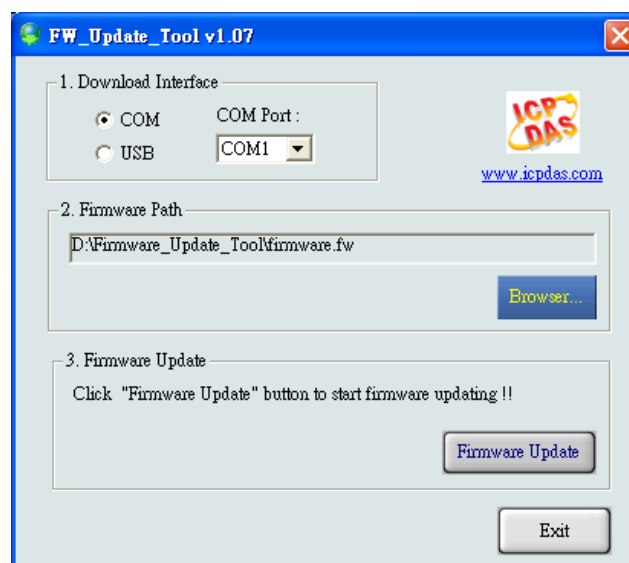
步驟 3: 取得模組更新用的軟體: “Firmware Update Tool”及模組韌體。
模組更新用的軟體路徑如下:

<https://www.icpdas.com/tw/download/show.php?num=3963&model=I-7565M-HS>

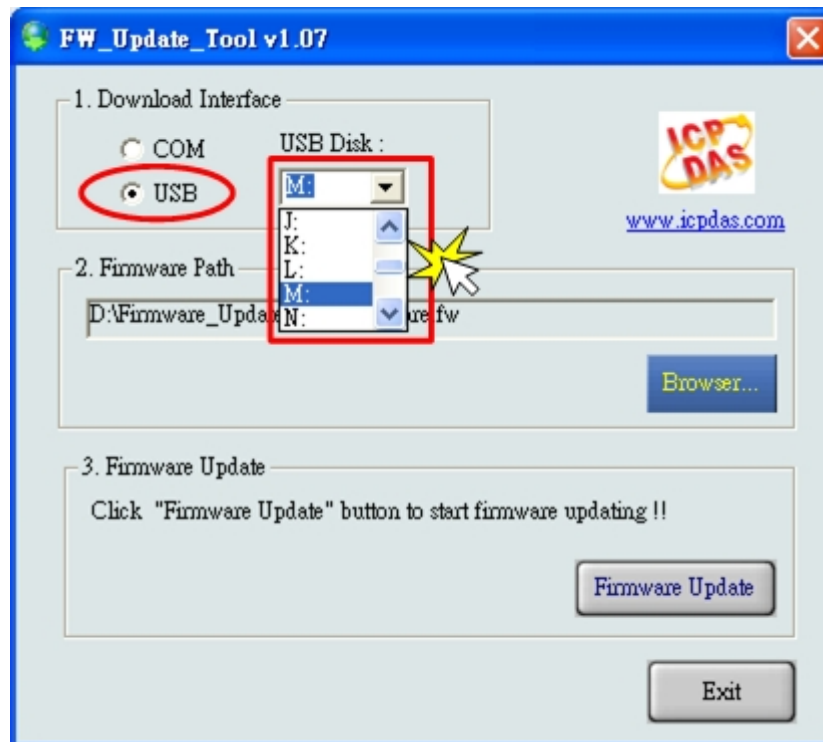
模組韌體路徑如下:

<https://www.icpdas.com/tw/download/show.php?num=3957&model=I-7565M-HS>

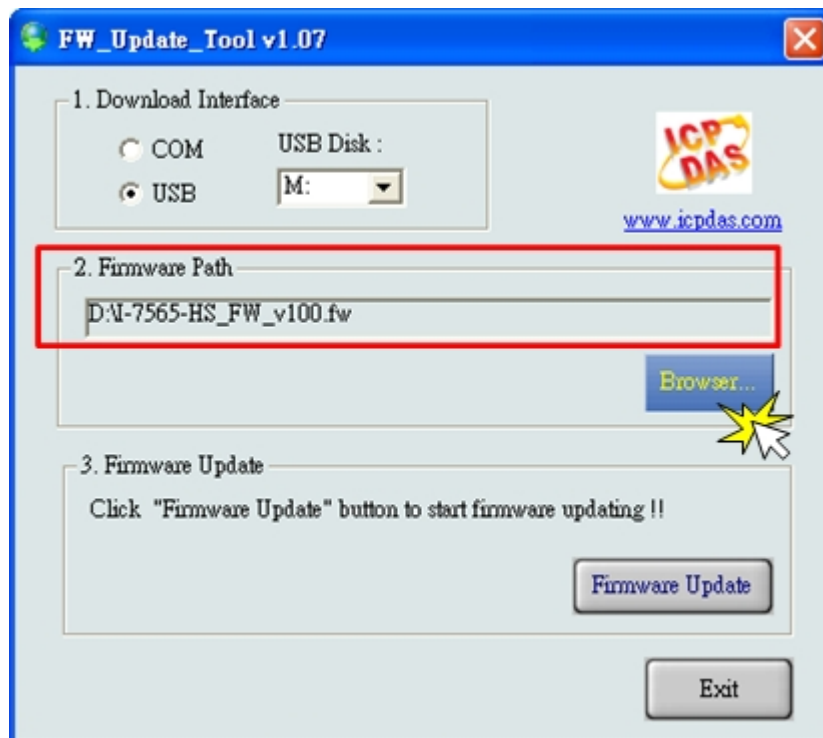
步驟 4: 執行韌體更新程式” Firmware Update Tool”。



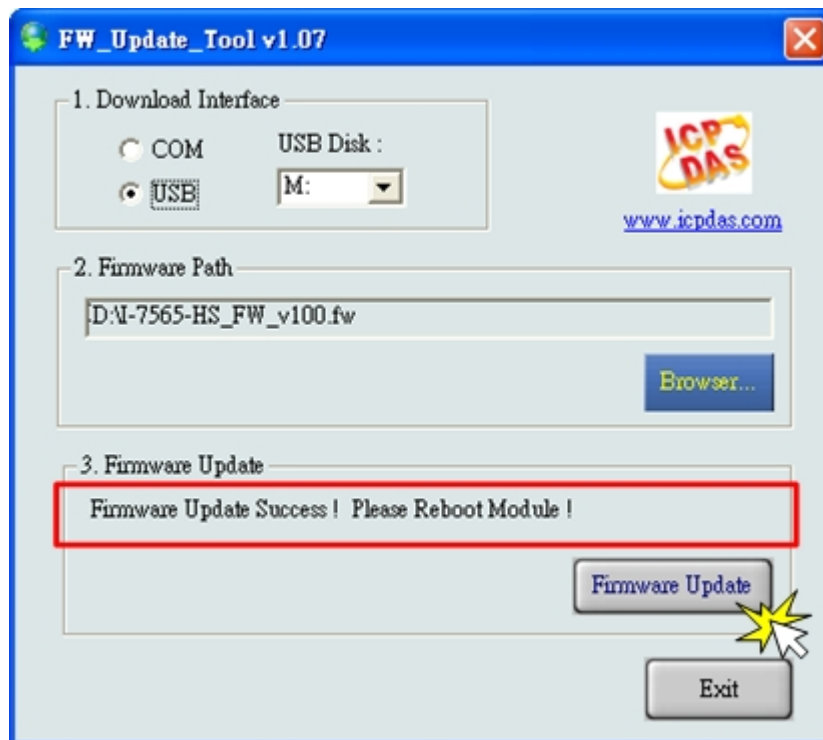
步驟 5: 選擇 PC 端連接至模組所使用的 USB 磁碟槽。



步驟 6: 按下”Browser...”按鈕，選取所以要更新用的模組韌體(*.fw)。



步驟 7: 按下”Firmware Update”按鈕開始更新模組韌體。更新成功之後，在”3. Firmware Update”訊框中會顯示” Firmware Update Success! Please Reboot Module!”。



步驟 8: 將模組”Init.”指撥開關位置調整至”OFF”位置。

步驟 9: 將模組重新啟動並按下”Exit”按鈕關閉程式。

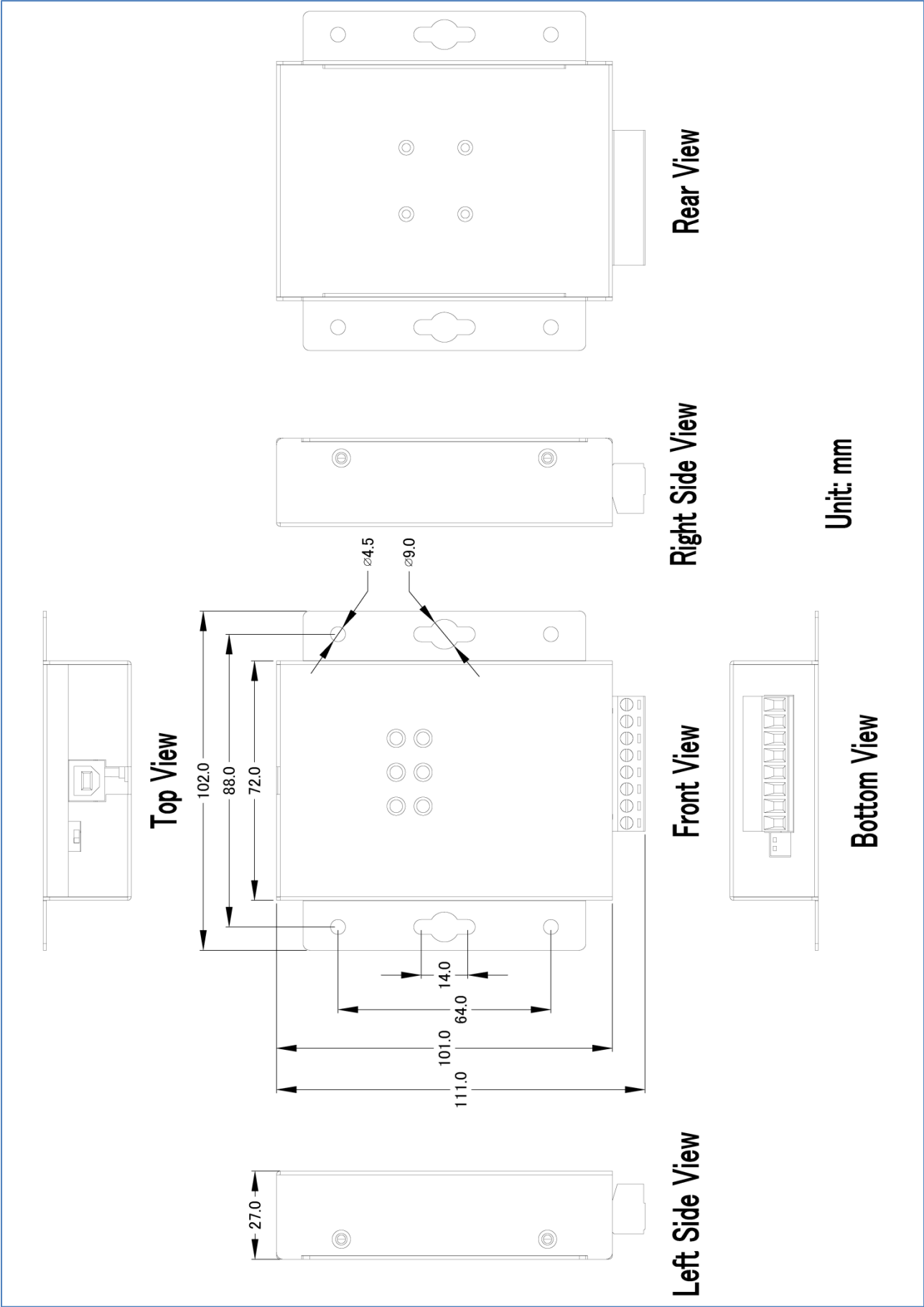
7. 附錄

7.1. 文件版本歷史紀錄

本章節描述此文件的版本修改紀錄。下表為版本歷史紀錄。

版本	日期	描述
1.0.0	Jun. 2018	初版
1.1.0	Jan. 2023	修正 dll, demo, utility 及 firmware 檔案 下載連結

7.2. 機構



7.3. CAN Status 暫存器

位元	符號	數值	描述
2:0	LEC		Last error code Type of the last error to occur on the CAN bus. The LEC field holds a code which indicates the type of the last error to occur on the CAN bus.
		0x0	No error.
		0x1	Stuff error: More than 5 equal bits in a sequence have occurred in a part of a received message where this is not allowed.
		0x2	Form error: A fixed format part of a received frame has the wrong format.
		0x3	AckError: The message this CAN core transmitted was not acknowledged.
		0x4	Bit1Error: During the transmission of a message (with the exception of the arbitration field), the device wanted to send a HIGH/recessive level (bit of logical value '1'), but the monitored bus value was LOW/dominant.
		0x5	Bit0Error: During the transmission of a message (or acknowledge bit, or active error flag, or overload flag), the device wanted to send a LOW/dominant level (data or identifier bit logical value '0'), but the monitored Bus value was HIGH/recessive.
		0x6	CRCErrror: The CRC checksum was incorrect in the message received.
		0x7	Unused: No CAN bus event was detected
3	TXOK		Transmitted a message successfully.
		0	No message has been successfully transmitted.
		1	A message has been successfully transmitted.
4	RXOK		Received a message successfully
		0	No message has been successfully received
		1	A message has been successfully received independent of the result of acceptance filtering.
5	EPASS		Error passive
		0	The CAN controller is in the error active state.
		1	The CAN controller is in the error passive state as defined in the CAN 2.0 specification.
6	EWARN		Warning status
		0	Both error counters are below the error warning limit of 96.
		1	At least one of the error counters in the Error Counter Register has reached the error warning limit of 96.
7	BOFF		Busoff status
		0	The CAN module is not in busoff state.
		1	The CAN controller is in busoff state.
31:8	-	-	Reserved

7.4. CAN Error Counter 暫存器

位元	符號	數值	描述
7:0	TEC		Transmit error counter Current value of the transmit error counter (maximum value 255)
14:8	REC		Receive error counter Current value of the receive error counter (maximum value 127).
15	RP		Receive error passive
		0	The receive counter is below the error passive level.
		1	The receive counter has reached the error passive level as defined in the CAN2.0 specification.
31:16	-	-	Reserved