EMP-S
使用者導引

版本 ：V.2.00

日期 ：2019.11

http://www.epcio.com.tw

目 錄

1 概論

EtherCAT 運動控制軟體平台(EMP-S)提供純軟體運動控制(Soft-Motion)的解決方案，使用者不須使用軸卡，即可透過乙太網路傳送資料封包與 EtherCAT 從站(伺服驅動器、I/O 等)進行資料交換，具備多軸運動控制能力(最多可支援 16 軸同動)；EtherCAT(Ethernet for Control Automation Technology；乙太網路控制自動化技術)最初為德國 Beckhoff 公司所研發基於乙太網路的開放架構通訊協議，並應用於自動化與工業控制領域的即時現場匯流排(Fieldbus)。EMP-S 具備 EtherCAT 通訊協議的分散式時鐘(Distributed Clock；DC)機制，可以保證各從站間同步誤差小於 1μs。使用者可利用 EMP-S 所提供的運動控制函式庫(Motion Control Command Library；MCCL)對伺服驅動器及周邊 I/O 下達相容於 CANOpen Over EtherCAT(CoE)通訊協定的運動命令、讀取目前命令位置、編碼器回授位置、EtherCAT I/O 狀態與原點復歸等，採用此架構不僅配線維護簡單、抗雜訊干擾能力高、安全可靠性高、節省成本，並可發展高速、高精度的自動化設備。

2 EMP-S 系統與軟體安裝、使用說明

2.1 系統需求與連接架構

A. 硬體需求：

雙核心以上 PC 或工業電腦 (須具備網路卡，規格參考表 1) 。

B. 軟體需求：

■ 作業系統環境
- ✓ WINDOWS 7
- ✓ WINDOWS 10

■ 開發環境
- ✓ Visual C++ (VC++)
- ✓ Visual C# (VC#)
- ✓RTX Runtime

表 1 支援 EtherCAT 網路卡驅動程式的晶片型號

Link Layer Name	**Controller / Device ID**
emllI8254x Intel Pro/1000	82540EM / 0x100E 82541EI / 0x1013 82541ER / 0x1078 82541GI / 0x1076 82541GI / 0x1077 82541PI / 0x107C 82545GM / 0x1026 82546EB / 0x1010 82546GB / 0x1079 82547EI / 0x1075 82547GI / 0x1019 82566DM / 0x104A 82566L / 0x10BD 82566MC / 0x104D

	82567V / 0x10CE 82567V / 0x1501 82567LM / 0x10DE 82567LM / 0x10F5 82571GB / 0x10A4 82571GB / 0x10BC 82572GI / 0x10B9 82572PI / 0x107D 82573 / 0x108C 82573E / 0x108B 82573L / 0x109A 82574(L) / 0x10D3 82575 / 0x10A7 82577LM / 0x10EA 82577LC / 0x10EB 82576 / 0x10C9 82576 ET2 / 0x1526 82578DM / 0x10EF 82578DC / 0x10F0 82579LM / 0x1502 82579V / 0x1503 82580 / 0x150E 82580 QF / 0x1527 82583V / 0x150C N1E5132 / 0x105E I350 / 0x1521 I210 / 0x1533 I210 CFL / 0x157B I211AT / 0x1539 I217LM / 0x153A I217V / 0x153B I218LM / 0x155A
emllI8255x Intel Pro/100	82551QM / 0x1059

	82555VE2 / 0x27DC 82557 / 0x1229 82557ER / 0x1209 82559ER / 0x2449 82562 / 0x1039 82801DB / 0x103A 82801EB / 0x1050 Pro/100/M / 0x1229 Pro/100/S / 0x1229 Pro/100/VE /0x1092
emllRTL8139 Realtek RTL8139	8139D / 0x8139 D-Link 8139D / 0x1300
emllRTL8169 Realtek Gigabit NIC (PCIe, PCI)	RTL8110 / 0x8169 RTL8111 / 0x8168 RTL8168 / 0x8168 RTL8169 / 0x8169 D-Link RTL8169 / 0x4300 RTL8169SC / 0x8167 RTL8103 / 0x8136

在 PC 或工業電腦上安裝 EMP-S 系統(RTX Runtime、EtherCAT 網路卡驅動程式、EcSever 與 MCCL 函式庫)，可透過乙太網路傳送資料封包與 EtherCAT 從站(如：伺服驅動器、I/O 模組)進行資料交換，如 Figure 2.1.1 所示為系統連接架構圖。

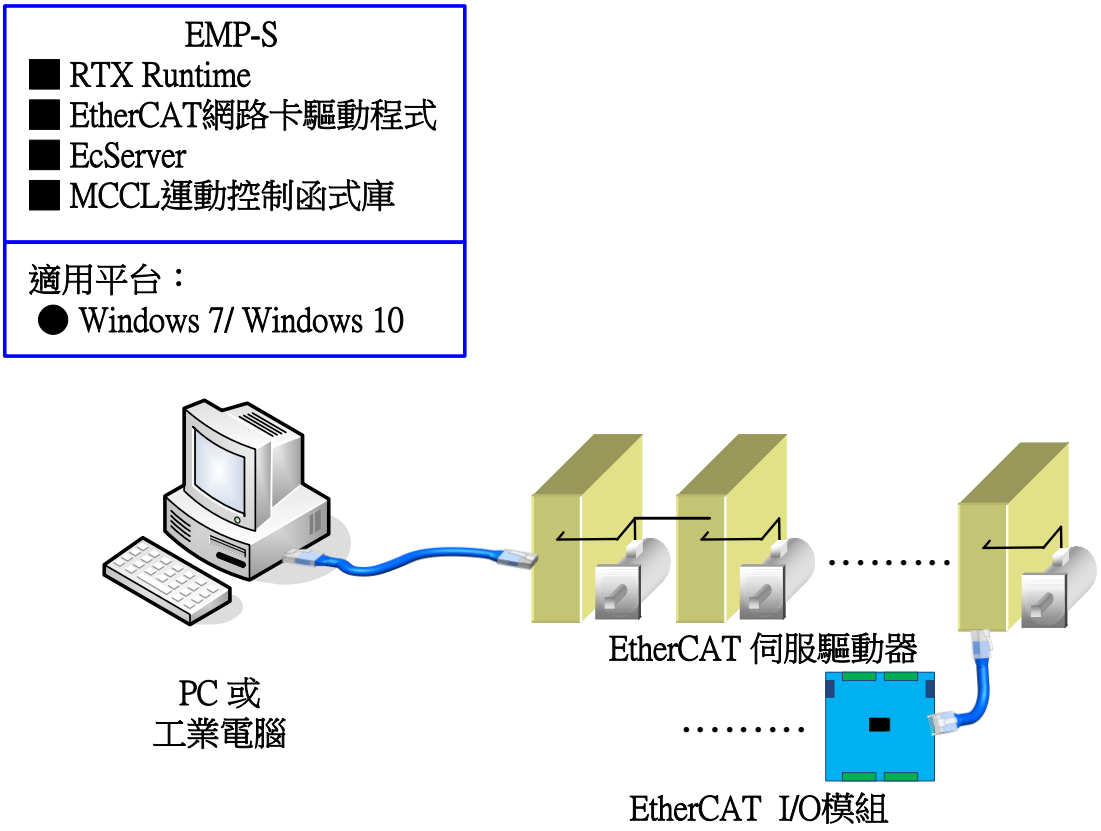


Figure 2.1.1 系統連接架構圖

2.2 32-bit Windows系統的軟體安裝步驟

1. 目錄與檔案架構

名稱	類型	說明
RTX	資料夾	存放 EcServer 應用程式和網路卡動態函式庫 EcServer.rtss emllRTL8169.rtdll emllI8254x.rtdll 請將檔案放置在特定目錄下（如：C:\RTX）
Win	資料夾	存放系統開發的相關目錄與檔案
dll	資料夾	存放 MCCL 動態函式庫 MCCL_Client.dll
include	資料夾	存放 VC++專案的標頭檔與 C#專案的 CS 檔 MCCL.h MCCL_RTX.h MCCL.cs
lib	資料夾	存放 VC++專案的靜態函式庫 MCCL_Client.lib
EcApp	資料夾	VC++專案範例程式
ECatDemoCode	資料夾	C#專案範例程式
EthercatSimple Demo_Csharp	資料夾	C#專案範例程式
Preset Tool	資料夾	存放 EMP Preset Tool 應用程式與 ESI 資料夾
ESI	資料夾	存放各驅動器廠牌的 ESI 檔案
MccMotion	資料夾	存放 EC_Config.xml 請將此目錄複製至 C:\
ENIFile.xml	xml 檔案	由 EMP Preset Tool 應用程式產生或自行產生的 ENI 檔案 請將此檔案放置在 C:\MccMotion
EC_Config.xml	xml 檔案	EtherCAT 組態設定檔 請將此檔案放置在 C:\MccMotion

2. 安裝RTX Runtime與線上註冊

- Step 1：進入主機板 BIOS 設定，確認 CPU Hyper-Threading（HT）功能為關閉狀態（Disabled）。
- Step 2：直接由安裝光碟進行安裝 RTX Runtime。
- Step 3：RTX 啟用註冊

開啟位置在『開始』→『所有程式』→『IntervalZero』→『RTX2012』→『Activate RTX』，在欄位【Enter your activation key】輸入合法啟動碼，點擊 Activate 按鈕後會透過網路連線至 IntervalZero 的授權伺服器，完成線上註冊(須具備可連線至網際網路)。

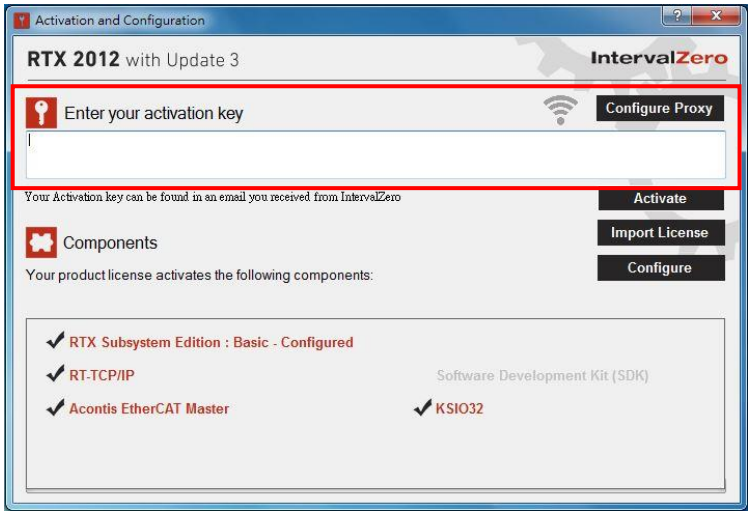


Figure 2.2.1 輸入 RTX Runtime 啟用碼

➢ Step 3：指定即時核心運作模式

點擊 Configure 按鈕進入 Configure 設定畫面，點選【Dedicated】模式，將 RTSS processors 設定為【1】，Windows processors 設定為【3】(此例 CPU 為四核心，指定一核心給 RTX 使用，其餘三核心歸 Windows 控管)，點擊 Done 按鈕。

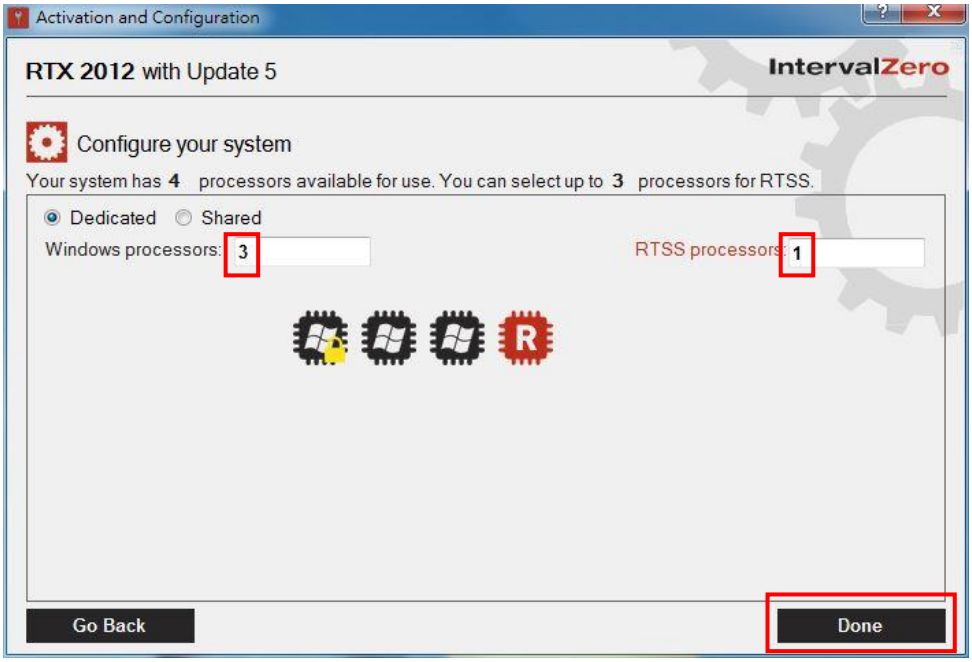


Figure 2.2.2 指定 Real-time 核心運作模式

➢ Step 4：設定完成後，需重新開機。

3. RTX 系統屬性設定與更新網路卡驅動程式

➢ Step 1：設定 RTX Properties

開啟位置在『開始』→『所有程式』→『IntervalZero』→『RTX2012』→『RTX Properties』。

將頁籤切換至『System』頁面，在『Behavior』點擊 Settings…按鈕，將欄位【HAL timer period (microseconds)】設定為【10】，點擊 OK 按鈕。

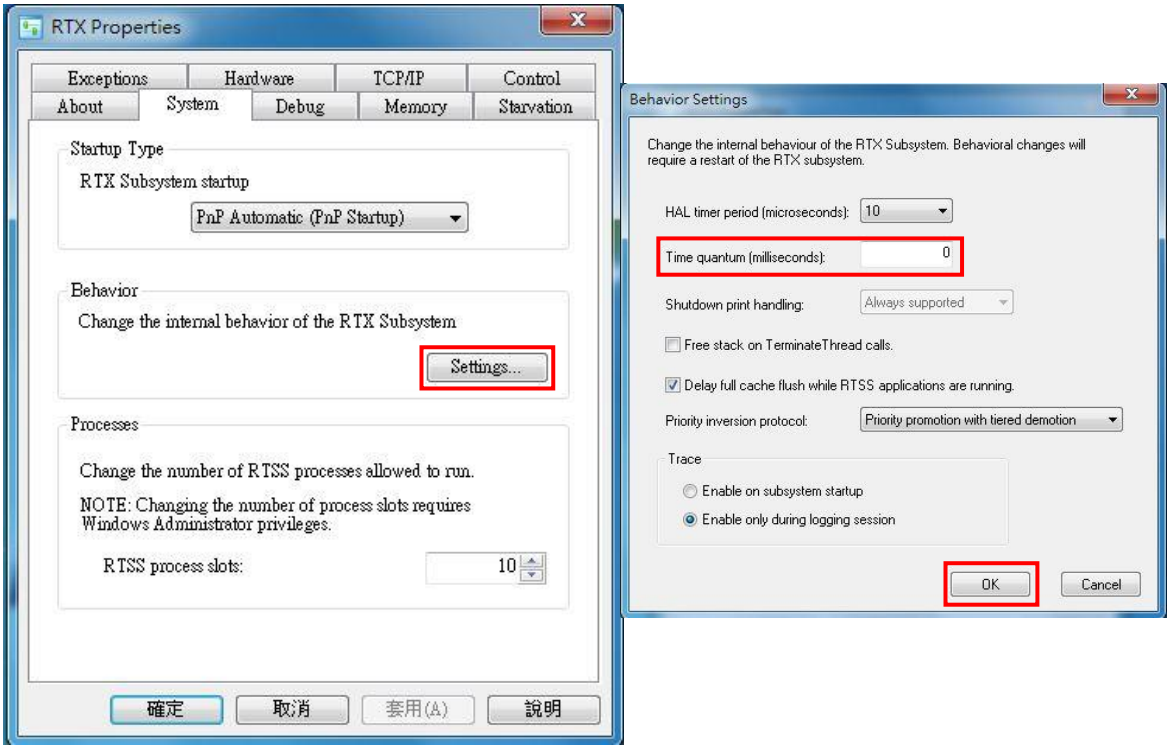


Figure 2.2.3 設定 HAL timer period

➢ Step 2：更新 RTX 網路卡驅動程式

1. 切換至『Hardware』頁面，在『Devices』點擊 Settings…按鈕，選取網路卡(Realtek PCI GBE Family Controller)的位置按下滑鼠右鍵點擊 Add RTX INF support 按鈕後，再點擊 Apply 按鈕。

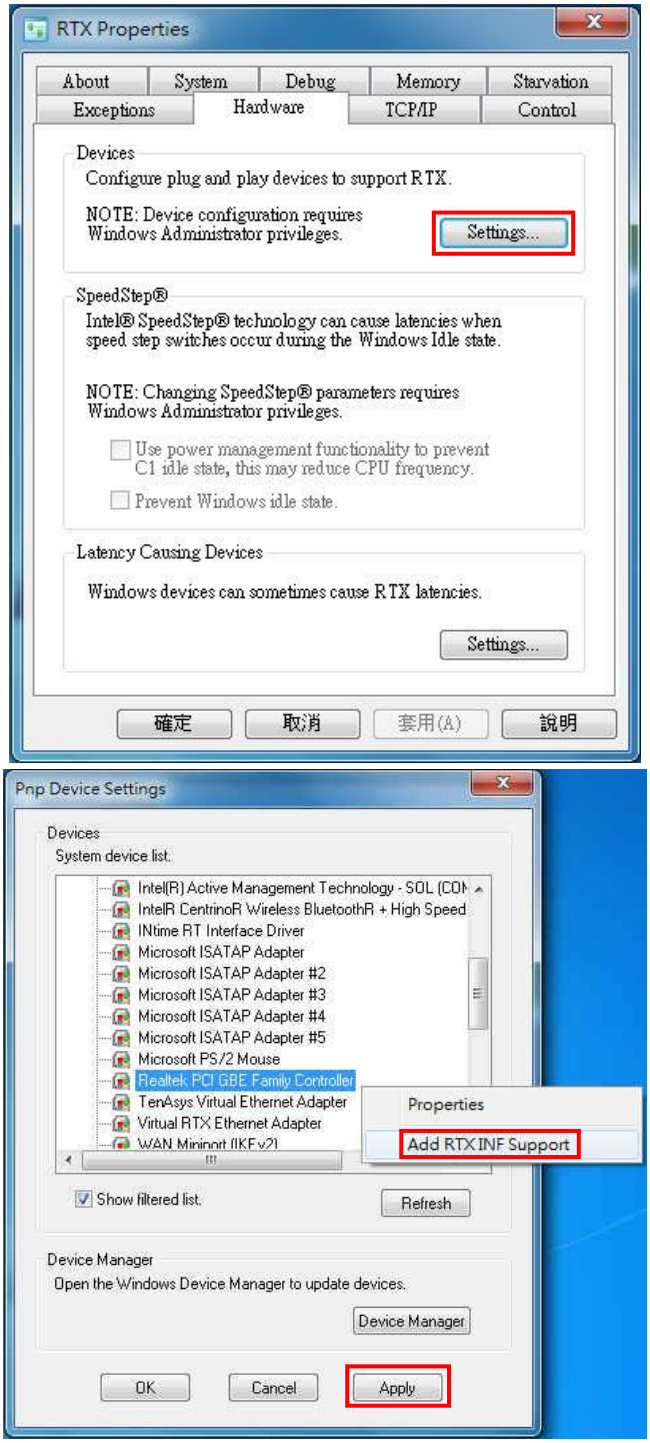


Figure 2.2.4 硬體裝置介面設定

2. 在『Devices Manager』點擊 Devices Manager 按鈕，進入裝置管理員後，選取網路卡(Realtek PCI GBE Family Controller)的位置，按下滑鼠右鍵點擊更新驅動程式軟體(P)…按鈕。

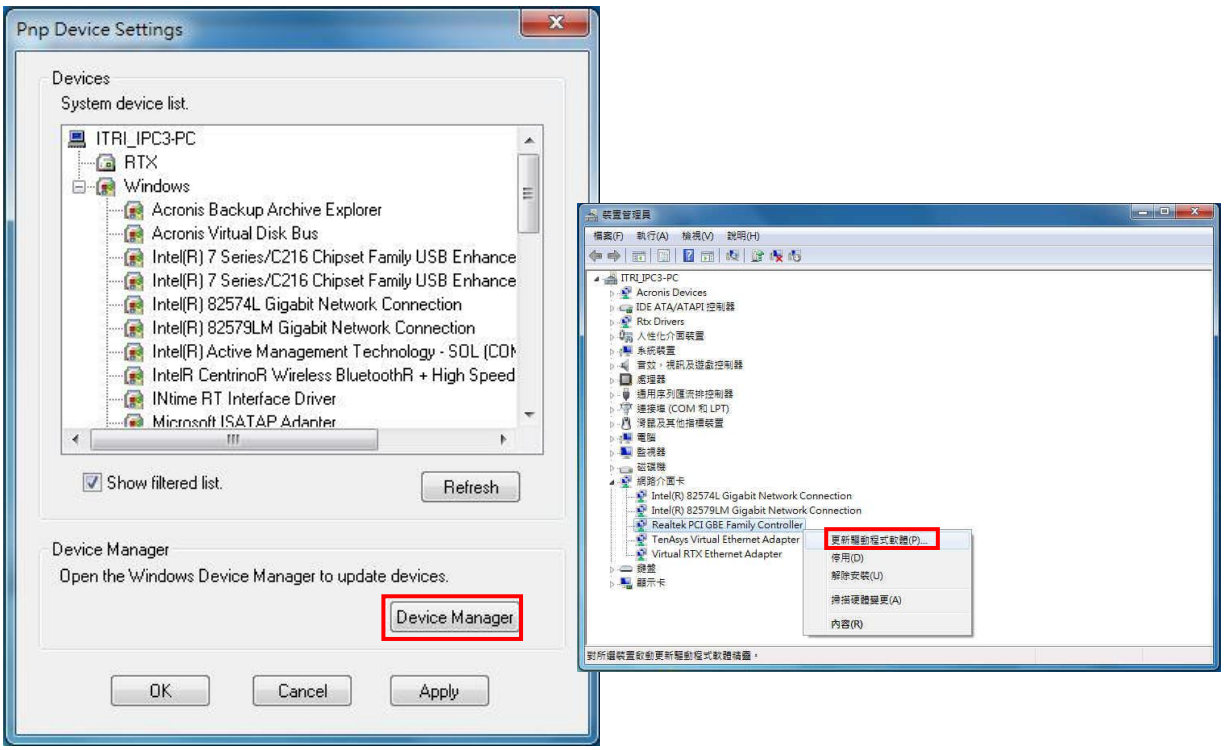


Figure 2.2.5 裝置管理員-更新驅動程式

3. 點擊瀏覽電腦上的驅動程式軟體(R)按鈕，再點擊讓我從電腦上的裝置驅動程式清單中挑選(L)按鈕。

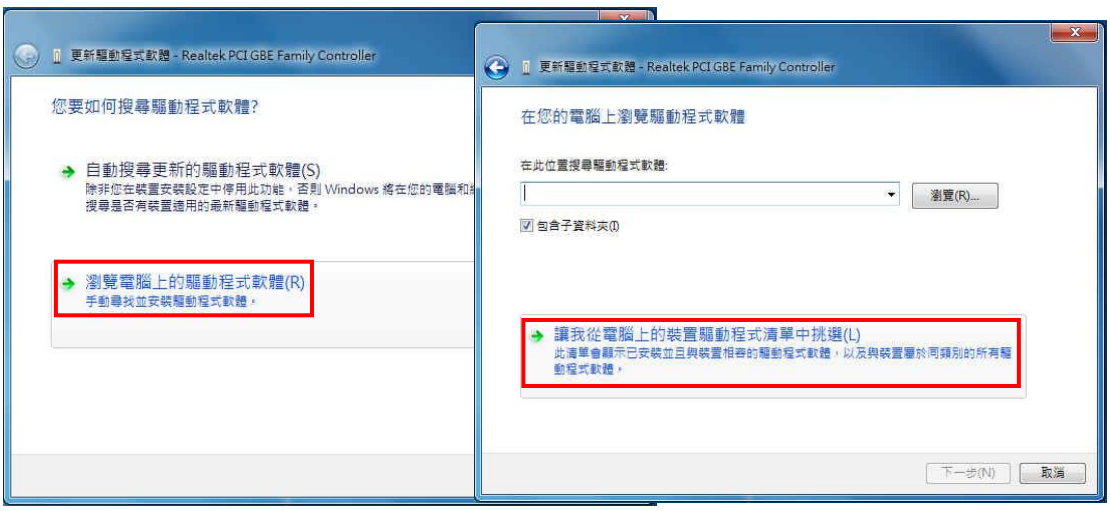


Figure 2.2.6 搜尋驅動程式

4. 從網路卡清單中點選【Realtek PCI GBE Family Controller RTX Supported】後，再點擊下一步(N)按鈕。

Figure 2.2.7 選取支援 RTX 網路卡驅動程式

5. 完成 RTX 網路卡驅動程式更新的畫面，如 Figure 2.2.8 所示。

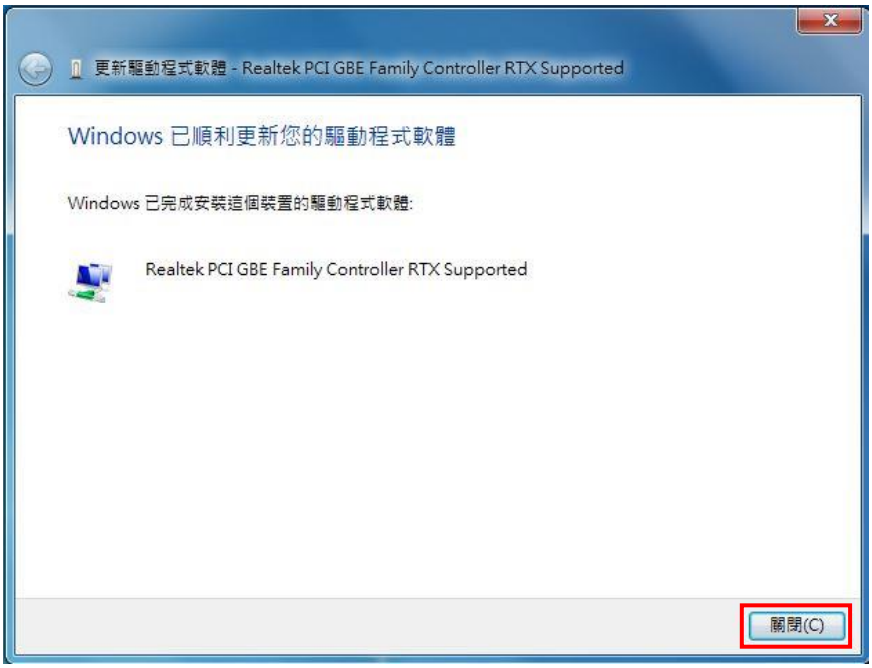


Figure 2.2.8 完成更新 RTX 網路卡驅動程式

6. 選取已掛載至 RTX 的網路卡(Realtek PCI GBE Family Controller)位置，按下滑鼠右鍵點擊 Properties 按鈕，在『Requested Resource』**取消**核選【Obtain line-based resources】，點擊 OK 按鈕。

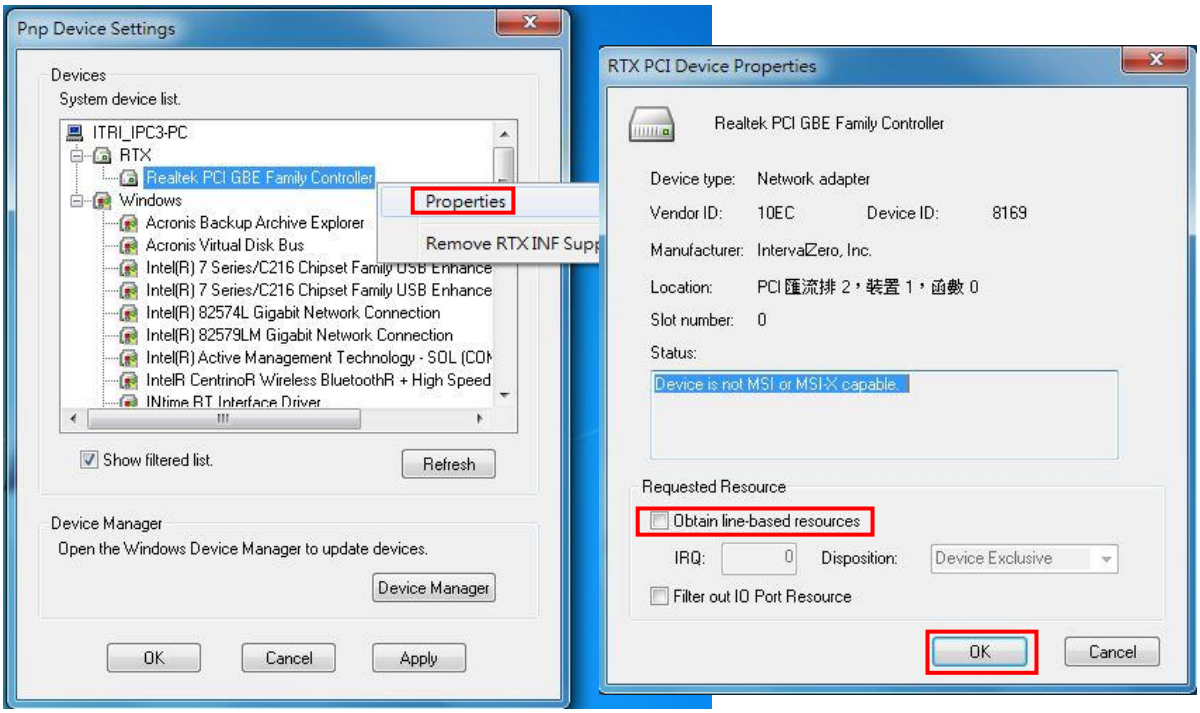


Figure 2.2.9 RTX 網路卡屬性設定

7. 完成以上設定，進行電腦關機後，必須將電源線予以**移除**再等待五秒後，重新將電源線接上進行開機，設定才會生效。

4. 掛載 EtherCAT 網路卡驅動程式：

使用者可以透過RTSS Task Manager應用程式確認EtherCAT網路卡驅動程式的檔案(emllRTL8169.rtdll或emllI8254x.rtdll)是否已掛載在RTX系統；若無，請自行掛載檔案，步驟如下：

- Step 1：啟動 RTSS Task Manager 應用程式

  開啟位置在『開始』→『所有程式』→『IntervalZero』→『RTX2012』→『Tools』→『RTSS Task Manager』，點擊 Start Task 按鈕後，在『Modes』點選【Register RTDLL】，再點擊 Browse…按鈕。

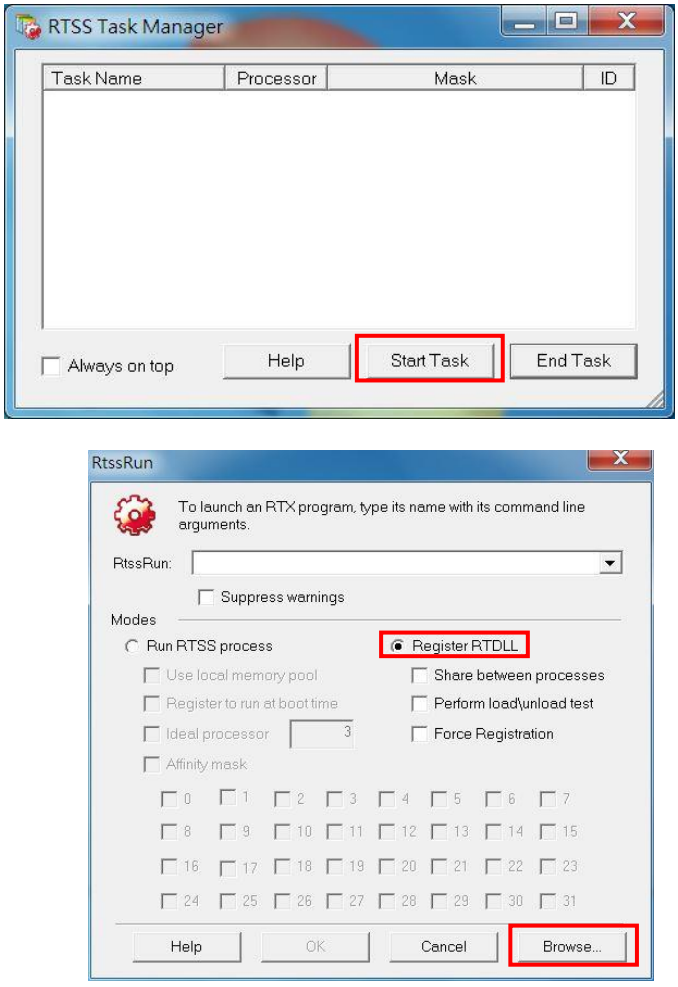


Figure 2.2.10 RTSS Task Manager

➢ Step 2：以 Realtek PCI 8169 為例，檔案為【RTX\emllRTL8169.rtdll】，點擊開啟舊檔(O)按鈕。

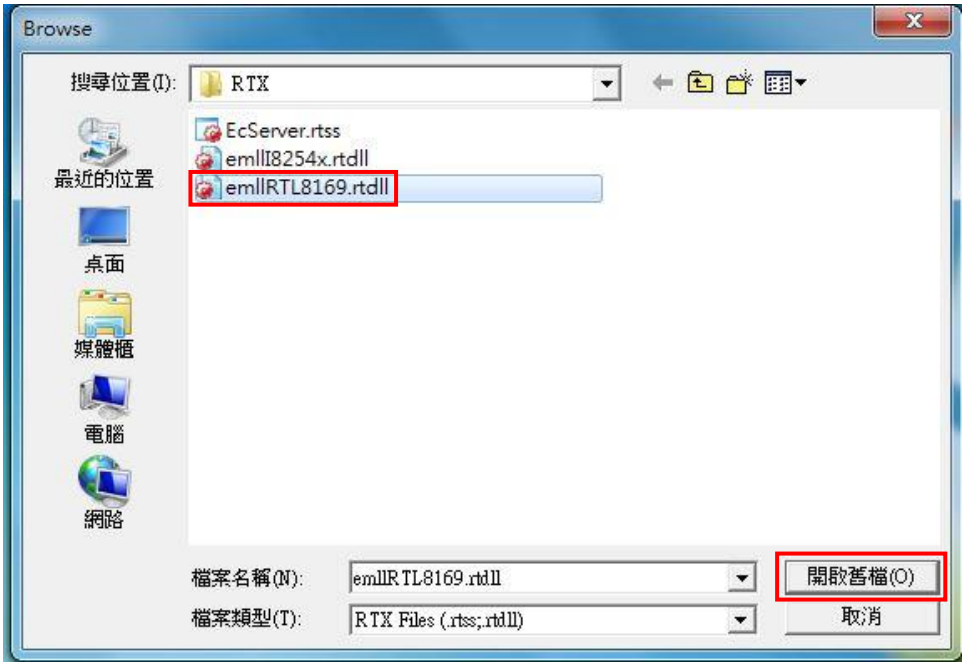


Figure 2.2.11 選取 Register RTDLL 檔案

➢ Step 3：點擊 OK 按鈕，即可將檔案掛載至 RTX 系統。

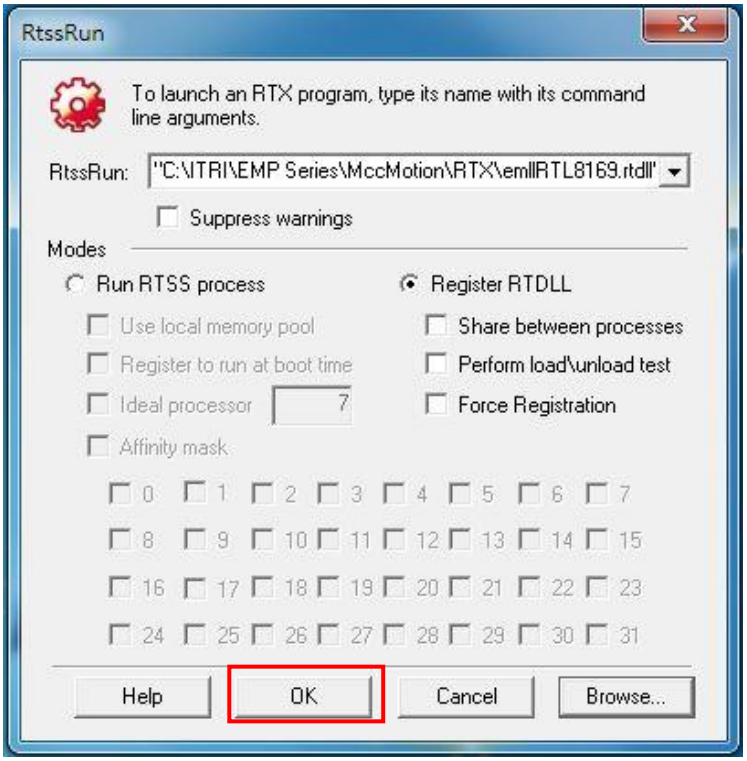


Figure 2.2.12 掛載 RTDLL 檔案

➢ Step 4：完成網路卡驅動程式掛載至 RTX 系統，如 Figure 2.2.13 所示。

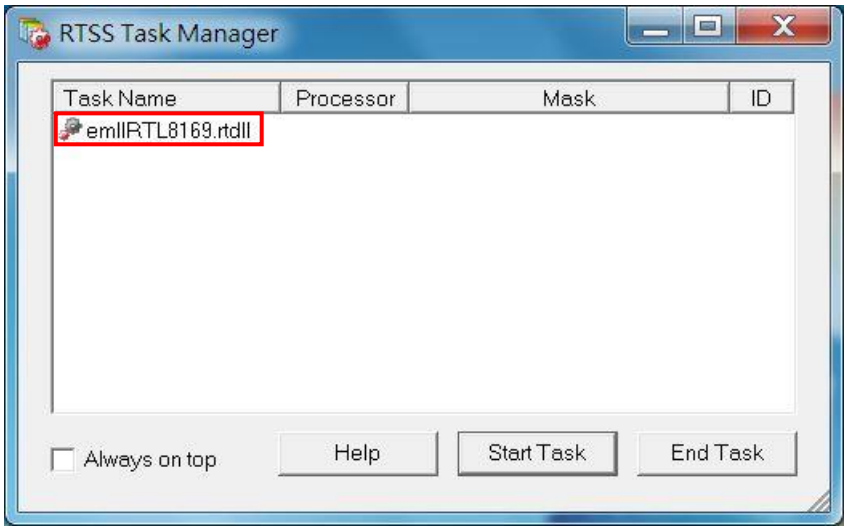


Figure 2.2.13 成功將檔案掛載至 RTX 系統

5. 手動啟動EcServer

➢ Step 1：啟動 RTSS Run 應用程式

開啟位置在『開始』→『所有程式』→『IntervalZero』→『RTX2012』→『Tools』→『RTSS Task Manager』，點擊Start Task按鈕後，在『Modes』點選【Run RTSS process】，再點擊Browse…按鈕。

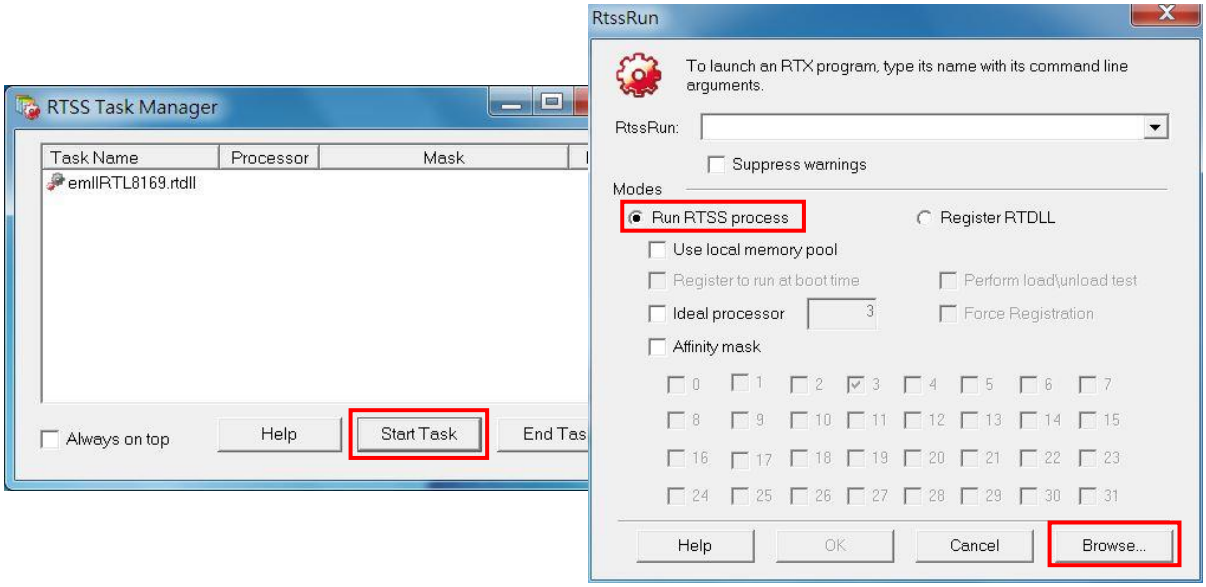


Figure 2.2.14 開啟 RTSS Task Manager

➢ Step 2：檔案為【RTX\EcServer.rtss】，點擊開啟舊檔(O)按鈕。

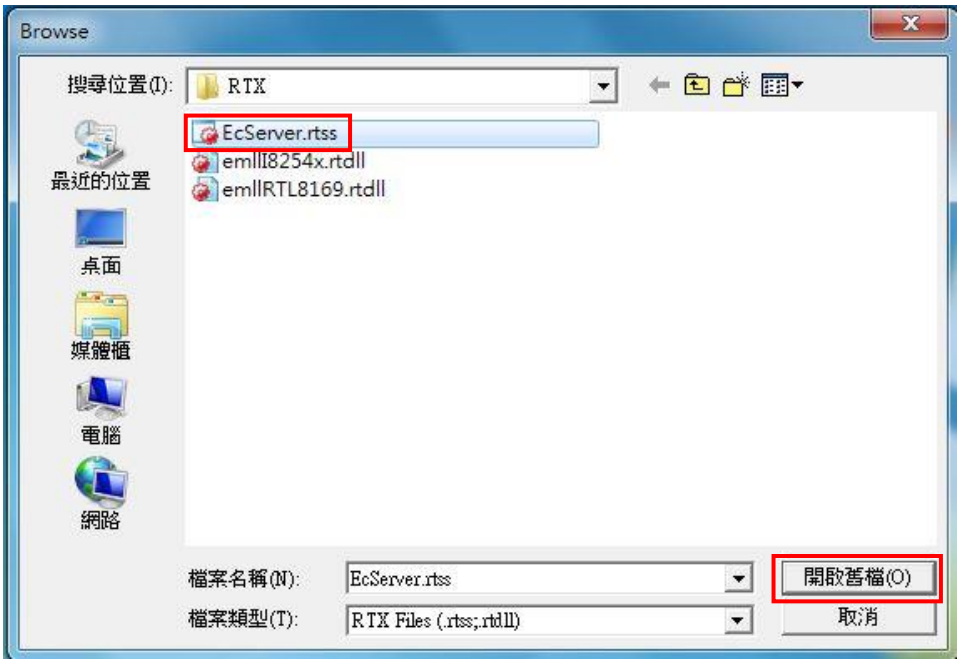


Figure 2.2.15 選取 EcServer 檔案

➢ Step 3：點擊 OK 按鈕，即可啟動 EcServer。

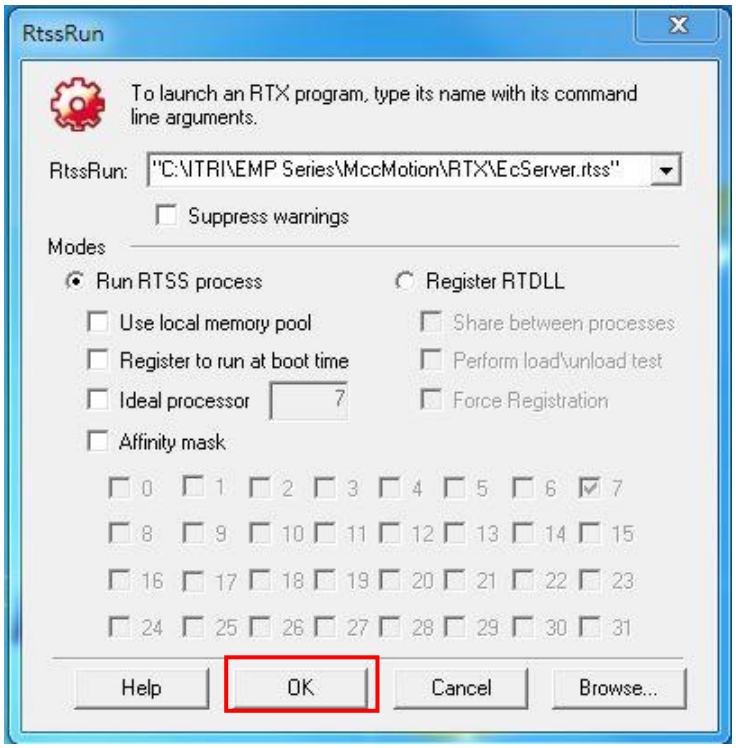


Figure 2.2.16 手動啟動 EcServer

➢ Step 4：完成 EcServer 啟動的畫面，如 Figure 2.2.17 所示。

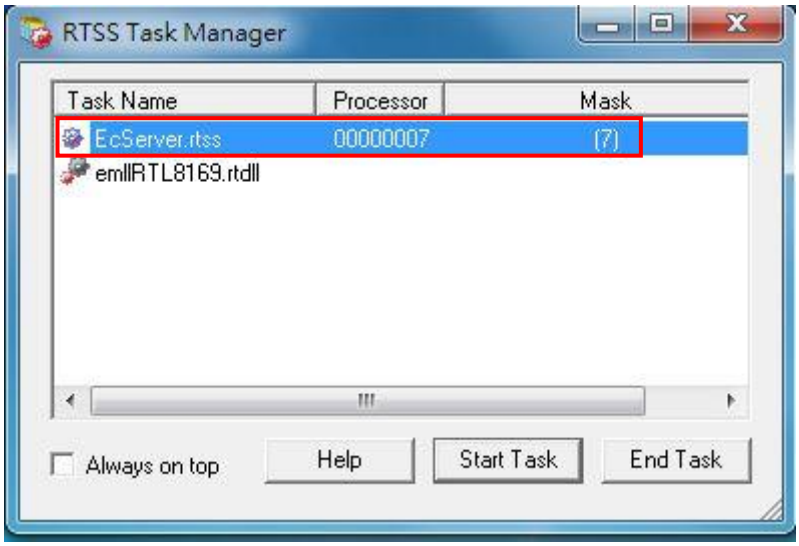


Figure 2.2.17 已成功啟動 EcServer

6. 手動終止/關閉EcServer：

EcServer在系統內僅能執行一次，若執行中發生不可預期之錯誤欲進行終止時，使用者可以透過RTSS Task Manager手動關閉EcServer，步驟如下：

➢ 開啟位置在『開始』→『所有程式』→『IntervalZero』→『RTX2012』→『Tools』→『RTSS Task Manager』，選取【EcServer.rtss】點擊End Task按鈕。

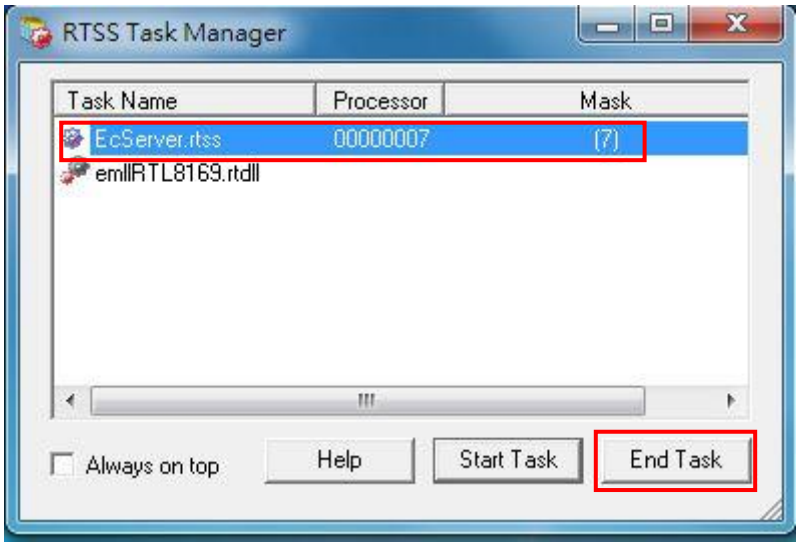


Figure 2.2.18 手動關閉 EcServer

7. 關閉RTX網路卡驅動程式的程序與關閉EcServer相同。

2.3 64-bit Windows系統的軟體安裝步驟

1. 目錄與檔案架構：

名稱	類型	說明
RTX	資料夾	存放 EcServer 應用程式和網路卡動態函式庫 EcServer.rtss EcMaster.rtdll emllRTL8169.rtdll emllI8254x.rtdll 請將檔案放置在特定目錄下（如：C:\RTX）或與專案執行檔同目錄
Win	資料夾	存放系統開發的相關目錄與檔案
dll	資料夾	存放 MCCL 動態函式庫 MCCL_Client.dll
include	資料夾	存放 VC++專案的標頭檔與 C#專案的 CS 檔 MCCL.h MCCL_RTX.h MCCL.cs
lib	資料夾	存放 VC++專案的靜態函式庫 MCCL_Client.lib
EcApp	資料夾	VC++專案範例程式
ECatDemoCode	資料夾	C#專案範例程式
EthercatSimple Demo_Csharp	資料夾	C#專案範例程式
Preset Tool	資料夾	存放 EMP Preset Tool 應用程式與 ESI 資料夾
ESI	資料夾	存放各驅動器廠牌的 ESI 檔案
MccMotion	資料夾	存放 EC_Config.xml 請將此目錄複製至 C:\
ENIFile.xml	xml 檔案	由 EMP Preset Tool 應用程式產生或自行產生的 ENI 檔案 請將此檔案放置在 C:\MccMotion
EC_Config.xml	xml 檔案	EtherCAT 組態設定檔 請將此檔案放置在 C:\MccMotion

2. 安裝RTX Runtime與線上註冊：

- Step 1：進入主機板 BIOS 設定，確認 CPU Hyper-Threading（HT）功能為關閉狀態（Disabled）。
- Step 2：直接由安裝光碟進行安裝 RTX Runtime。
- Step 3：RTX 啟用註冊

開啟位置在『開始』→『所有程式』→『RTX64 3.6 Runtime』→『Activation and Configuration』，在欄位【Enter your activation key】輸入合法啟動碼，點擊 Activate 按鈕後會透過網路連線至 IntervalZero 的授權伺服器，完成線上註冊(須具備可連線至網際網路)。

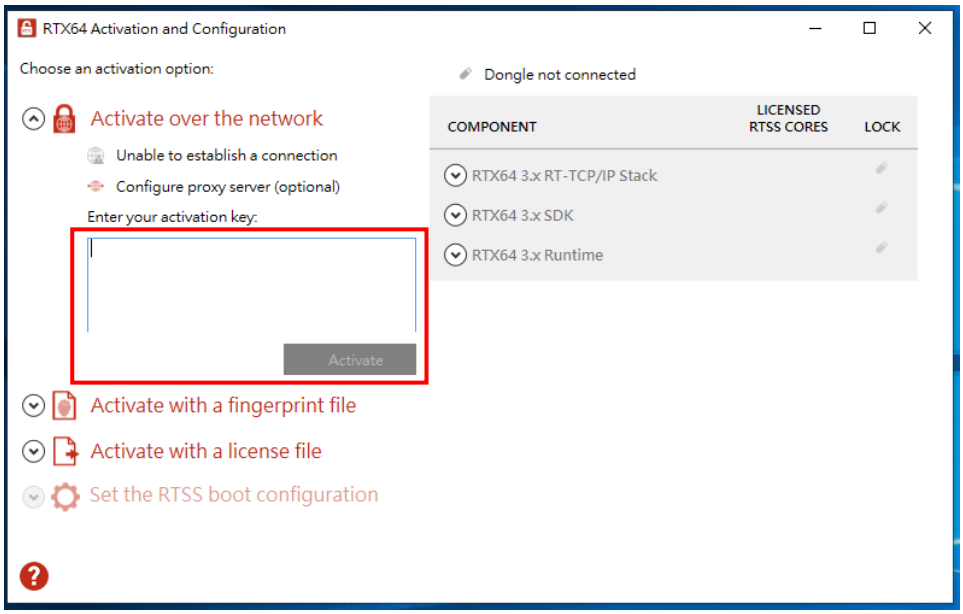


Figure 2.3.1 輸入 RTX Runtime 啟用碼

➢ Step 4：指定即時核心運作模式

點擊 Set the RTSS boot configuration 進入核心設定畫面，將 RTSS processors 設定為【1】，Windows processors 設定為【3】（此例 CPU 為四核心，指定一核心給 RTX 使用，其餘三核心歸 Windows 控管），點擊 Apply 按鈕。

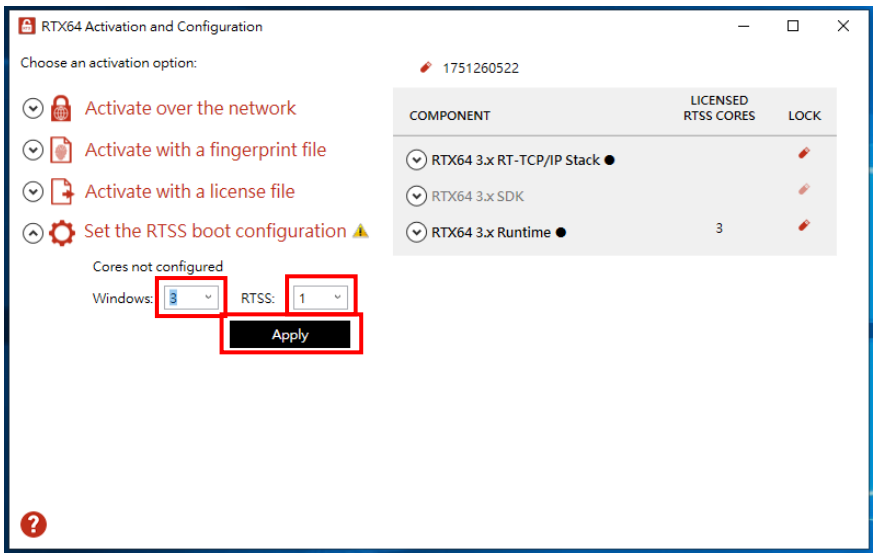


Figure 2.3.2 指定 Real-time 核心運作模式

➢ Step 5：設定完成後，需重新開機。

3. 更新RTX網路卡驅動程式：

1. 開啟『裝置管理員』,選取欲更新的網路卡： Intel(R) I210 Gigabit Network Connection #3 按下滑鼠右鍵點擊更新驅動程式(P) 按鈕。

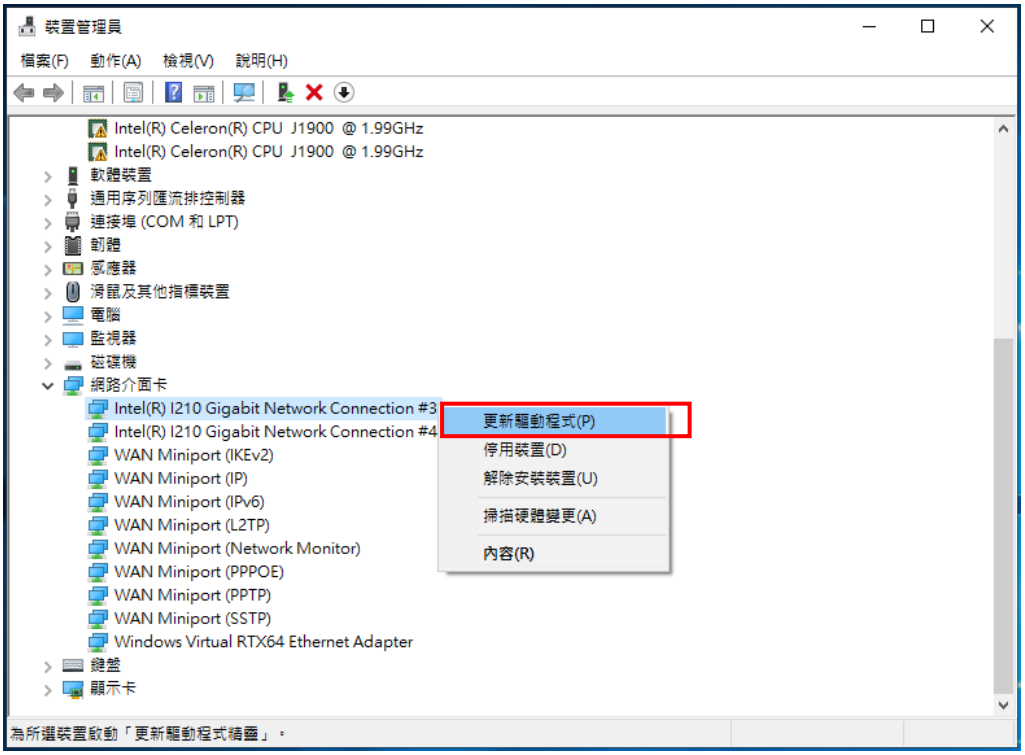


Figure 2.3.3 裝置管理員

2. 點擊瀏覽電腦上的驅動程式軟體(R)按鈕，再點擊瀏覽(R)按鈕去指定目錄為 C：\Program Files\IntervalZero\RTX64\inf，再點擊下一步(N)按鈕。

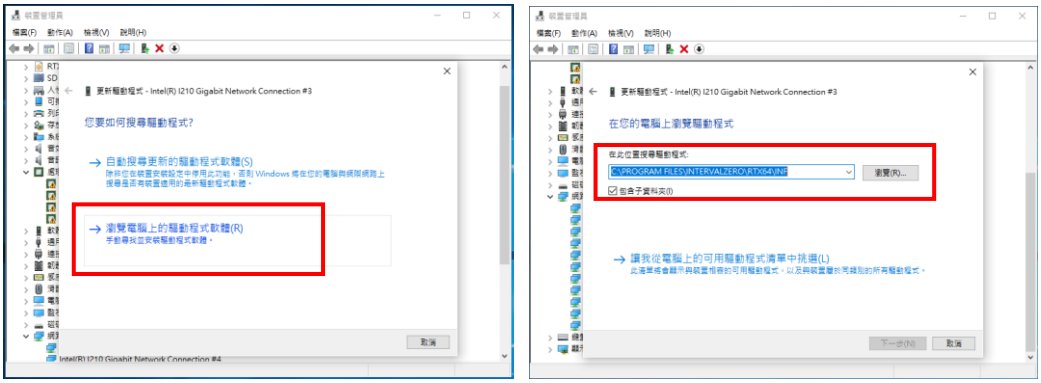


Figure 2.3.4 搜尋驅動程式

3. 從網路卡清單中點選【Intel I210 Copper-only Ethernet Controller(RTX64)】後，再點擊下一步(N)按鈕。

Figure 2.3.5 選取支援 RTX 網路卡驅動程式

4. 完成 RTX 網路卡驅動程式更新的畫面，如 Figure 2.3.6 所示。

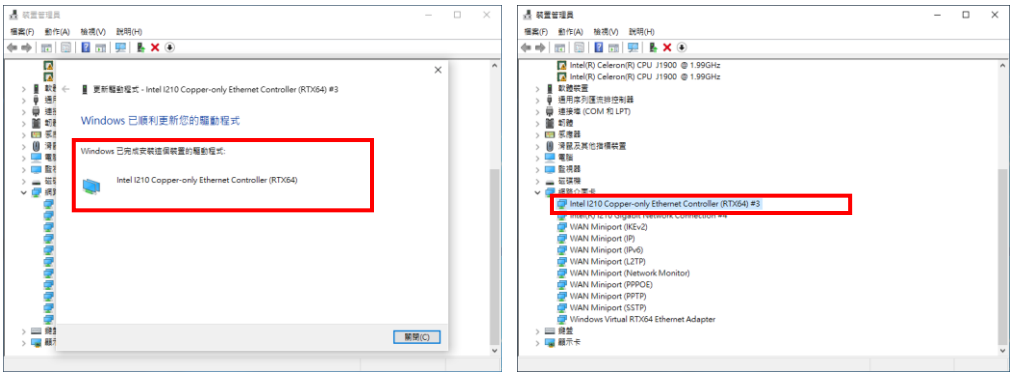


Figure 2.3.6 完成更新 RTX 網路卡驅動程式

5. 完成以上設定，進行電腦關機後，必須將電源線予以**移除**再等待五秒後，重新將電源線接上進行開機，設定才會生效。

4. 手動啟動 EcServer

➢ Step 1：啟動 RTSS Task Manager 應用程式

開啟位置在『開始』→『所有程式』→『RTX64 3.6 Runtime』→『RTSS Task Manager』，點擊Start Task按鈕後，再點擊Browse…按鈕。

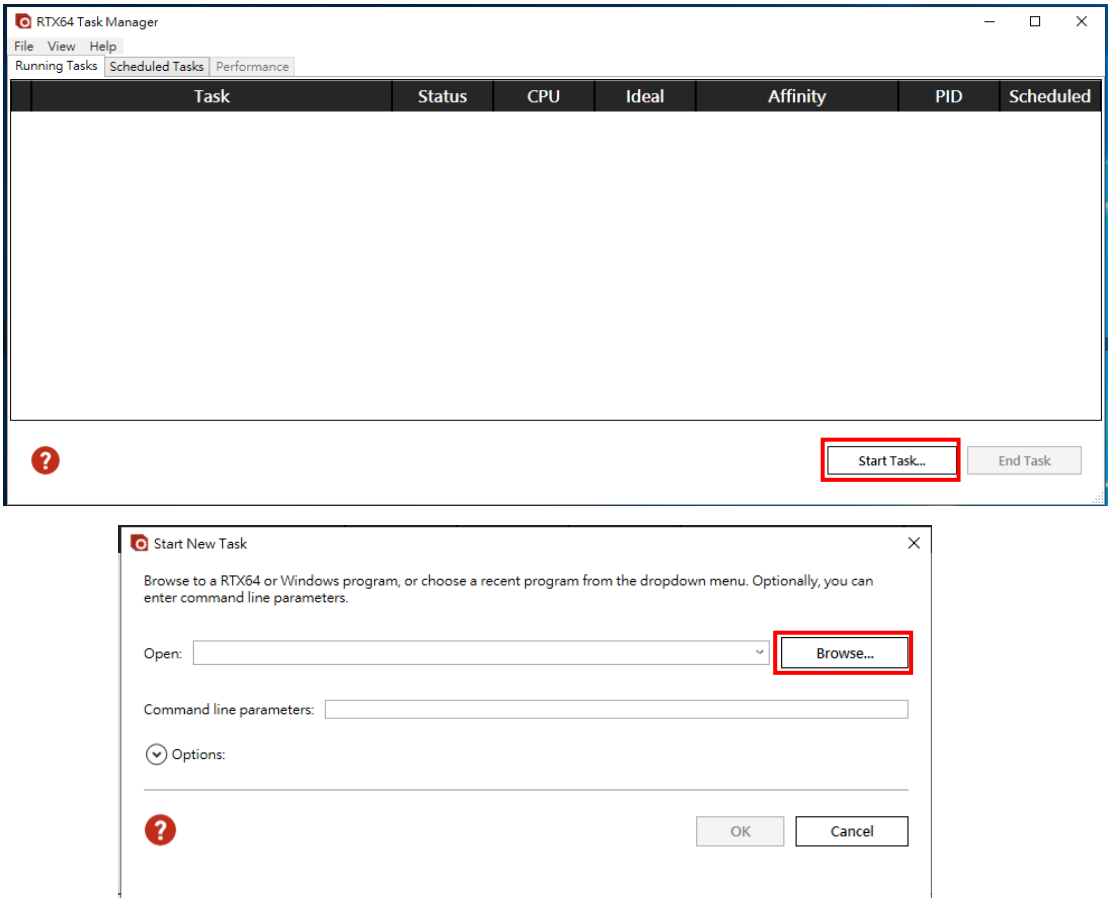


Figure 2.3.7 開啟 RTSS Task Manager

➢ Step 2：檔案為【RTX\EcServer.rtss】，點擊開啟(O)按鈕。

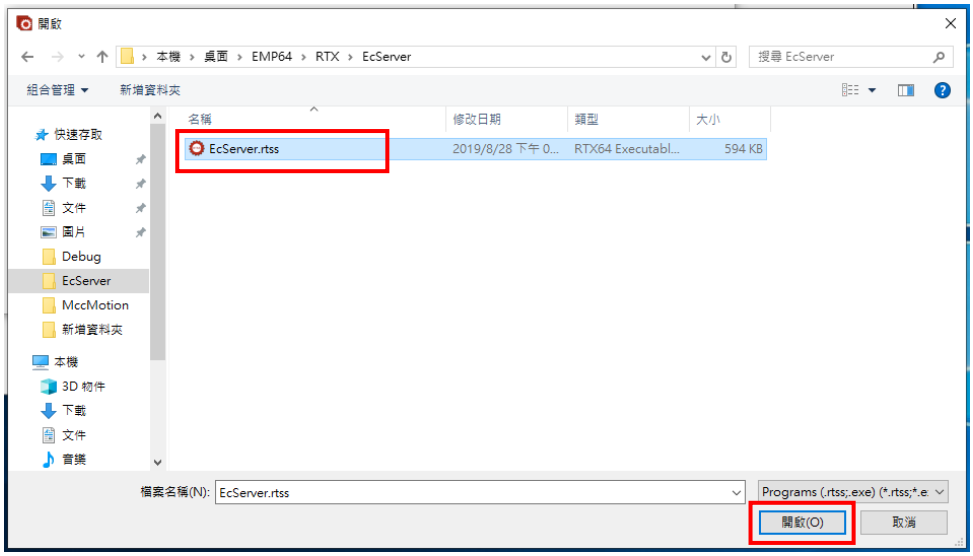


Figure 2.3.8 選取 EcServer 檔案

➢ Step 3：點擊 OK 按鈕，即可啟動 EcServer。

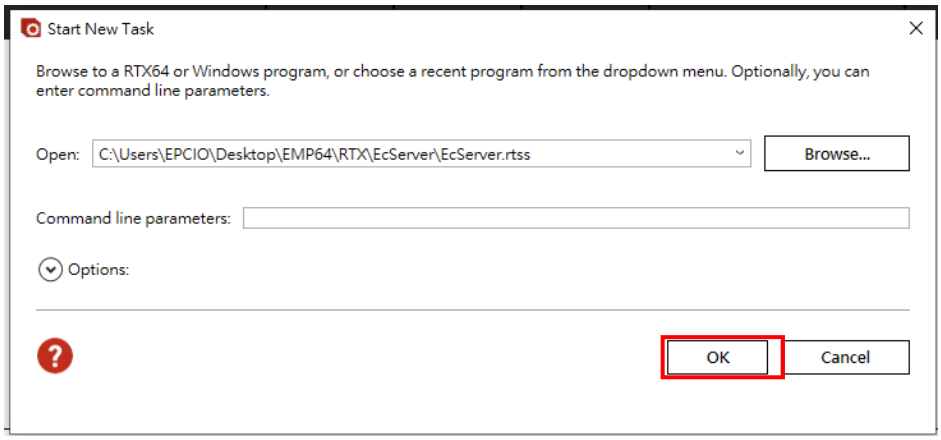


Figure 2.3.9 手動啟動 EcServer

➢ Step 4：完成 EcServer 啟動的畫面。

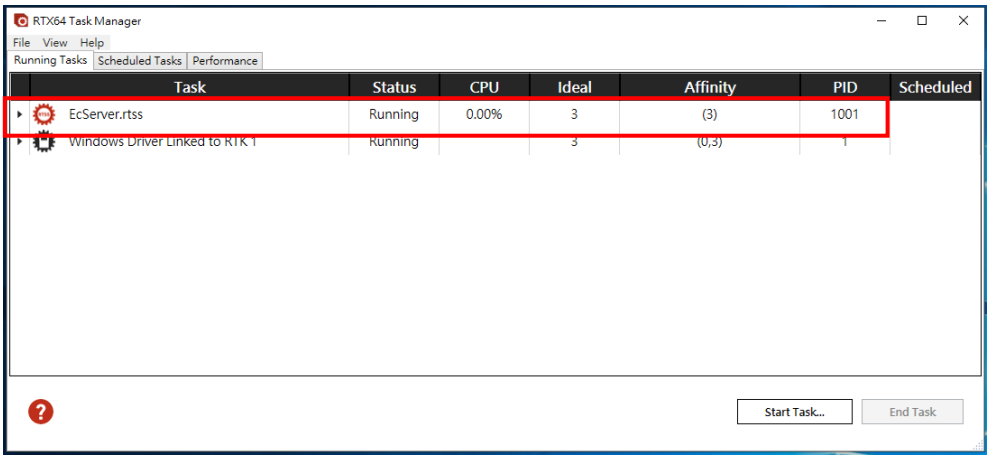


Figure 2.3.10 已成功啟動 EcServer

5. 卸載 RTX 網路卡驅動程式

點選欲卸載的 RTX 網路卡【Intel I210 Copper-only Ethernet Controller(RTX64) #3】，按下滑鼠右鍵點擊解除安裝裝置(U) 按鈕，勾選刪除此裝置的驅動程式軟體，再點擊解除安裝按鈕。

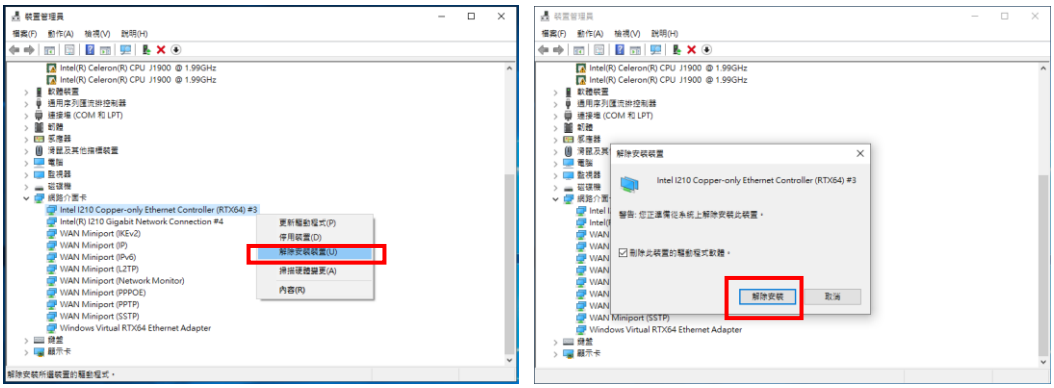


Figure 2.3.11 卸載網路卡驅動程式

6. 手動終止/關閉 EcServer

EcServer在系統內僅能執行一次，若執行中發生不可預期之錯誤欲進行終止時，使用者可以透過RTSS Task Manager手動關閉EcServer，步驟如下：

- 開啟位置在『開始』→『所有程式』→『RTX64 3.6 Runtime』→『RTSS Task Manager』，選取【EcServer.rtss】點擊 End Task 按鈕。

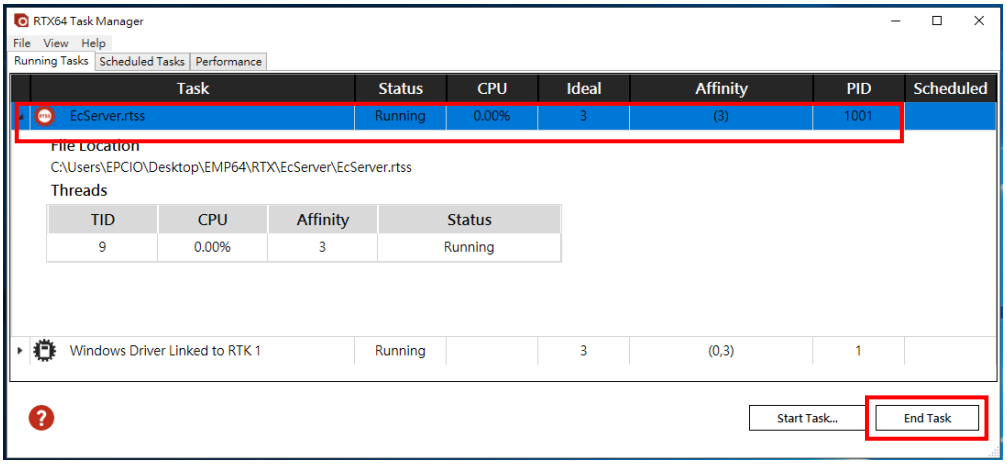


Figure 2.3.12 手動關閉 EcServer

2.4 Preset Tool應用程式與產生ENI檔案

Preset Tool 應用程式可自動探索 EtherCAT 網路拓樸結構，讀取每一從站的 ESI(EtherCAT Slave Information)，並自動架構出符合 EMP-S 需求之 ENI(EtherCAT Network Information)。

1. 應用程式操作步驟：

- Step 1：請確認驅動器的網路線連結至 PC 或工業電腦的網路卡（請勿使用已掛載 RTX 的網路卡）。
- Step 2：請安裝 WinPcap 軟體，下載網址：https://www.winpcap.org/。
- Step 3：將驅動器廠商提供的 ESI 檔案放至 Preset Tool\ESI 資料夾。
- Step 4：執行 Preset Tool\EMP_PresetTool v2.1.1.exe，下圖為執行輸入的畫面：

- Step 5：請輸入 PDO 模式：1 / 2 / 3（預設，請輸入 **1**）

■ 模式 1：

RxPDO	6040, 607A
TxPDO	6041, 6064, 60FD

■ 模式 2：

RxPDO	6040, 607A
TxPDO	6041, 6064, 60FD, 606B, 606C, 6074, 6077, 60F4

■ 模式 3：

RxPDO	6040, 607A, 60FF, 6071
TxPDO	6041, 6064, 60FD, 606B, 606C, 6074, 6077, 60F4

此時應用程式會列出掃瞄到的驅動器數量與名稱，並產生 ENI，檔名為 **ENIFile.xml**（自動存放至 C:\ MccMotion 目錄）。

注意：

- 應用程式將自動掃瞄連結於 Windows 網路卡的 EtherCAT 驅動器，無法掃瞄連結於 RTX 網路卡的驅動器。
- 支援 Sanyo、Delta、Yaskawa 及 Panasonic 的 EtherCAT 驅動器，其他廠牌驅動器持續開發中。
- 預設 DC Cycle Time 為 1ms。

2. 相關檔案及資料夾說明：

名稱	類型	說明
Preset Tool	資料夾	存放 EMP Preset Tool 應用程式
ESI	資料夾	存放各驅動器廠牌的 ESI 檔案
MccMotion	資料夾	存放 EC_Config.xml 請將此目錄複製至 C:\
Preset Tool v2.1.1.exe	exe 檔案	主執行檔
ENIFile	xml 檔案	產生的 ENI 檔案 請將此檔案放置在 C:\MccMotion
EC_Config.xml	xml 檔案	EtherCAT 組態設定檔 請將此檔案放置在 C:\MccMotion

- EtherCAT 組態檔(EC_Config.xml)設定：

  以記事本開啟EC_Config.xml進行內容的確認，如下所示：

標籤名稱	說明	範例
LinkLayer	網路卡裝置型號、編號與模式 (以 RTL8169 為例)	<LinkLayer> -rtl8169 1 1 </LinkLayer>
ENIFileName	ENI 檔案路徑 (存放路徑與 ENI 檔名)	<ENIFileName> C:\MccMotion\ENIFile.xml <\ENIFileName>
TotalDriverCount	Slave 驅動器數量 (成功連線到的 EtherCAT 裝置數量)	<TotalDriverCount> 8 </TotalDriverCount>

3. 錯誤訊息代碼與處理方式

錯誤代碼與訊息	發生原因	處理方式
Error[-3]: No ESI file in the Folder	找不到 ESI 檔案	確認驅動器的 ESI 檔案放置於 ESI 資料夾
Error[-4]: ESI Folder not found	找不到 ESI 目錄	1. 建立 ESI 資料夾、ESI 檔案 2. 重新安裝 SETUP.EXE
Error [-5]: Slave not found, please check EtherCAT connection!	找不到 EtherCAT 從站	1. 確認驅動器的網路線連結至電腦的網路卡 2. 確認驅動器電源開啟
Error [-6]: Slaves do not match with ESI!	未正確配對 Slave 與 ESI 檔案	檢查 ESI 檔案是否匹對 Slave 型號

3 開發專案環境設定

使用者可藉由 PC 或工業電腦進行程式開發、編譯及執行應用程式；支援 Visual C++、C#.NET、VB.NET 等開發環境，以下將以 Microsoft Visual Studio 2010 C++與 C#.NET 的專案環境設定為例。

3.1 Microsoft Visual Studio 2010 C++專案環境設定

➢ Step 1：開啟 Visual Studio 2010，功能表選單中點選【檔案】→【新增】→【專案】→【Visual C++】→【MFC】→【MFC 應用程式】，鍵入專案名稱：【DemoCPlusApp1】，點擊確定按鈕。

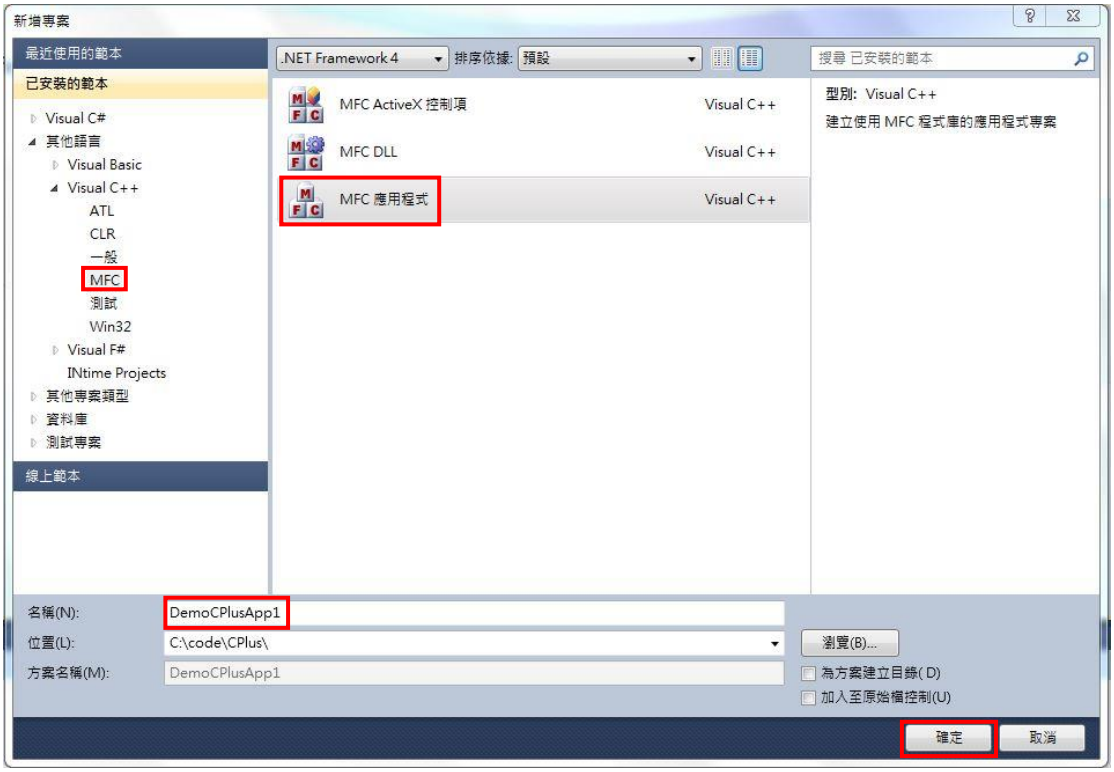


Figure 3.1.1 C++專案範本設定

➢ Step 2：將頁籤切換至『應用程式類型』頁面，點選【對話方塊式(D)】，在『MFC 的使用』點選【使用 MFC 的靜態函式庫(E)】，點擊完成按鈕。

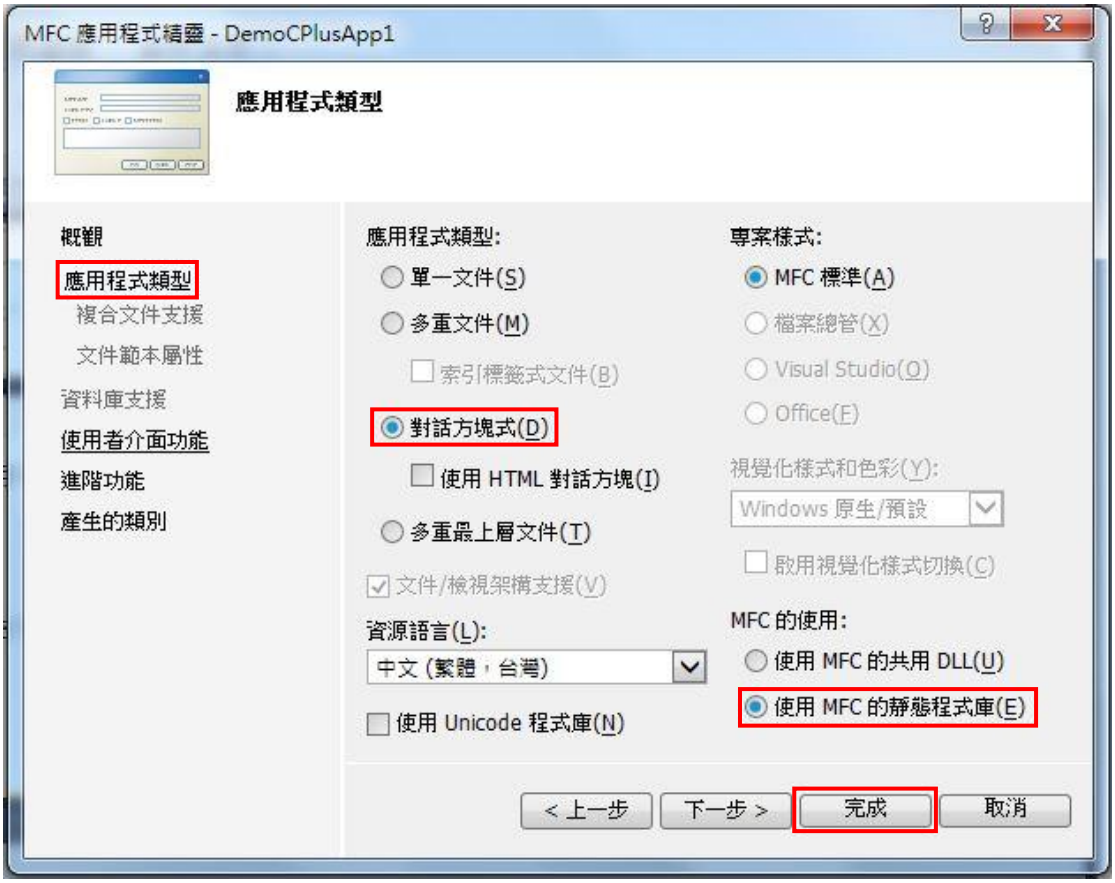


Figure 3.1.2 C++專案環境設定

註：若勾選【使用Unicode程式庫(N)】，需注意在GUI顯示時要使用寬字元(wide character)。

➢ Step 3：複製標頭檔，將 Win\include 資料夾複製至專案的 include 資料夾，包含：MCCL.h 與 MCCL_RTX.h。

Figure 3.1.3 include 資料夾的檔案

➢ Step 4：複製匯入函式庫(import library)，將 MCCL_Client.lib 複製至專案的 lib 資料夾。

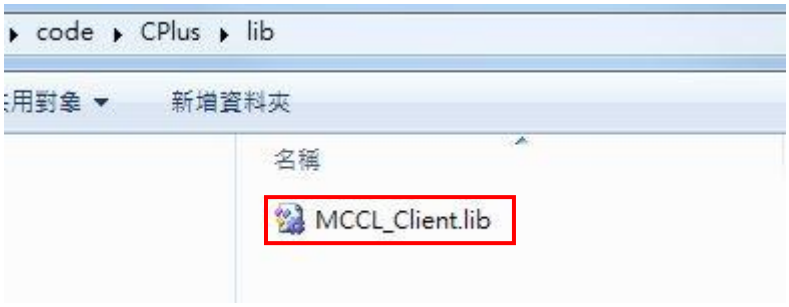
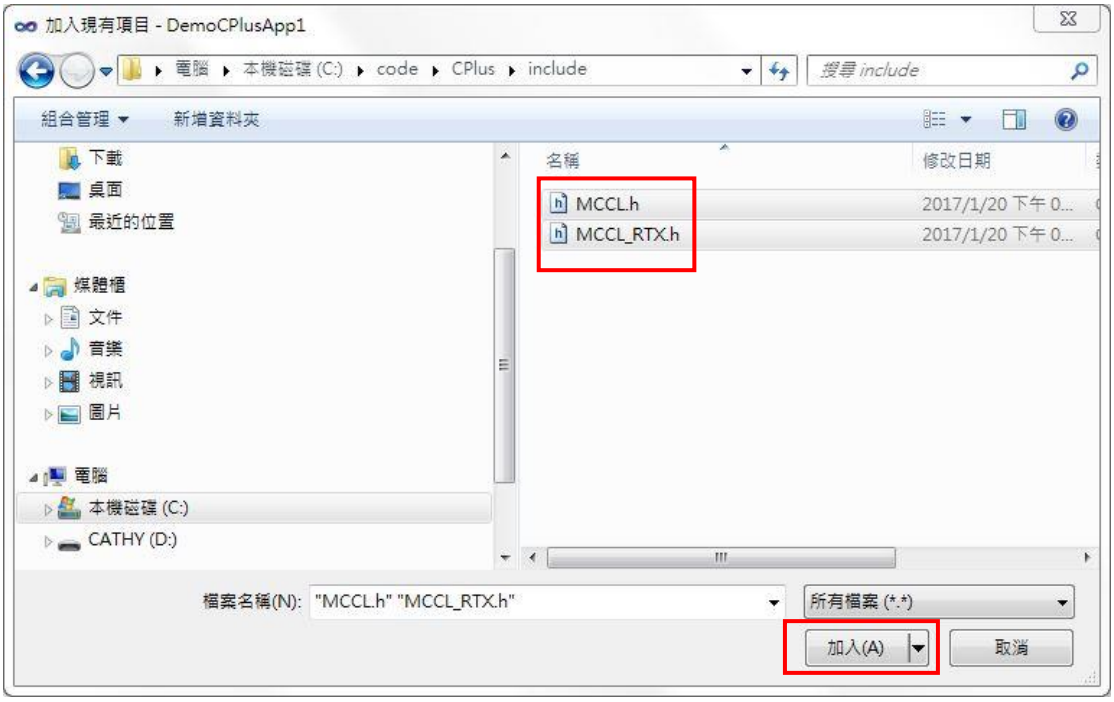


Figure 3.1.4 匯入函式庫

➢ Step 5：複製動態函式庫與 EcServer 應用程式，將 MCCL_Client.dll、EcServer.rtss 複製至專案執行檔的資料夾。若為 64-bit 系統，請將網路卡動態函式庫：EcMaster.rtdll、emllRTL8169.rtdll、emllI8254x.rtdll 一併放入。

Figure 3.1.5 動態函式庫與 EcServer 應用程式

➢ Step 6：點擊功能表的【專案(P)】→【加入現有項目(G)…】，選取 MCCL.h 與 MCCL_RTX.h 再點擊加入(A)按鈕。



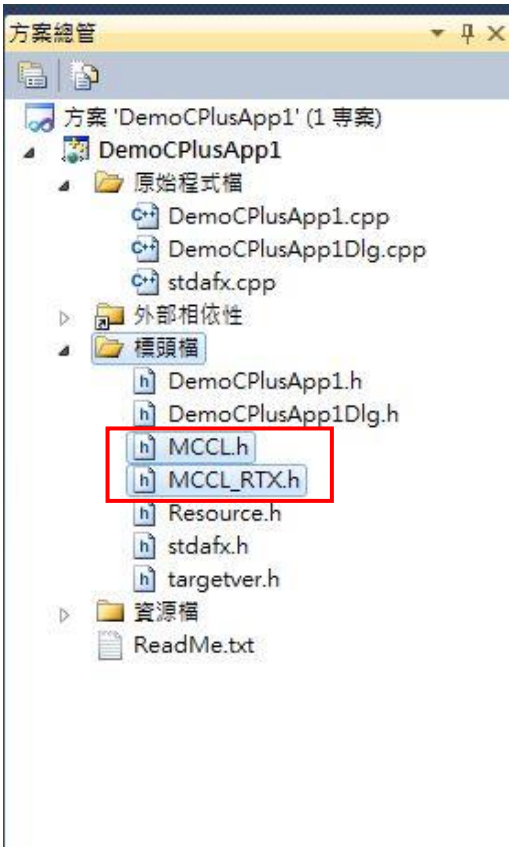


Figure 3.1.6 專案加入現有項目(.h)

➢ Step 7：點擊功能表的【專案(P)】→【屬性(P)】進行專案屬性設定，點擊【組態屬性】→【連結器】→【一般】→在欄位【其他程式庫目錄】鍵入【..\lib】。

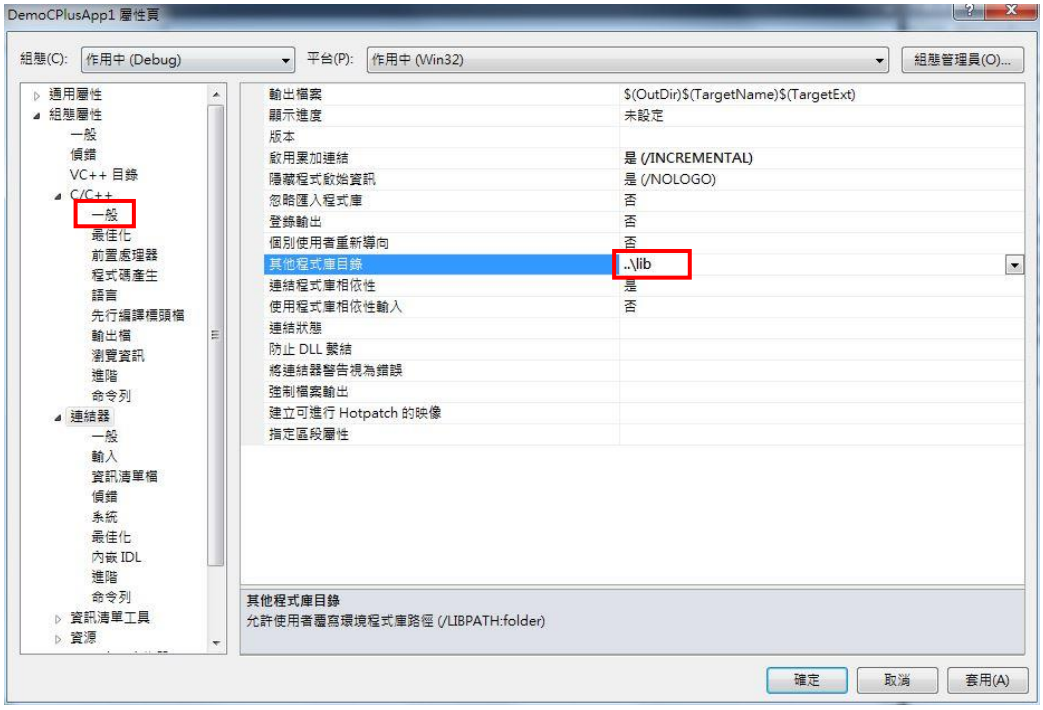


Figure 3.1.7 其他程式庫目錄設定

➢ Step 8：點擊【組態屬性】→【C/C++】→在欄位【其他 Include 目錄】鍵入【..\include】。

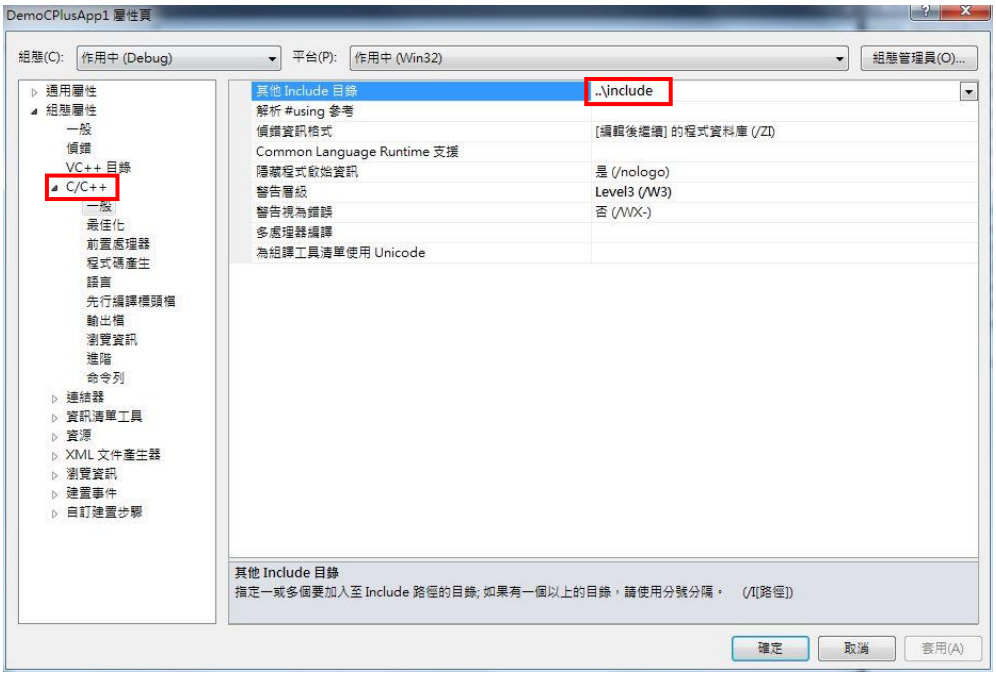


Figure 3.1.8 其他 Include 目錄設定

➢ Step 9：點擊【連結器】→【輸入】→在欄位【其他相依性】鍵入【MCCL_Client.lib】，依序點擊套用(A)、確定按鈕。

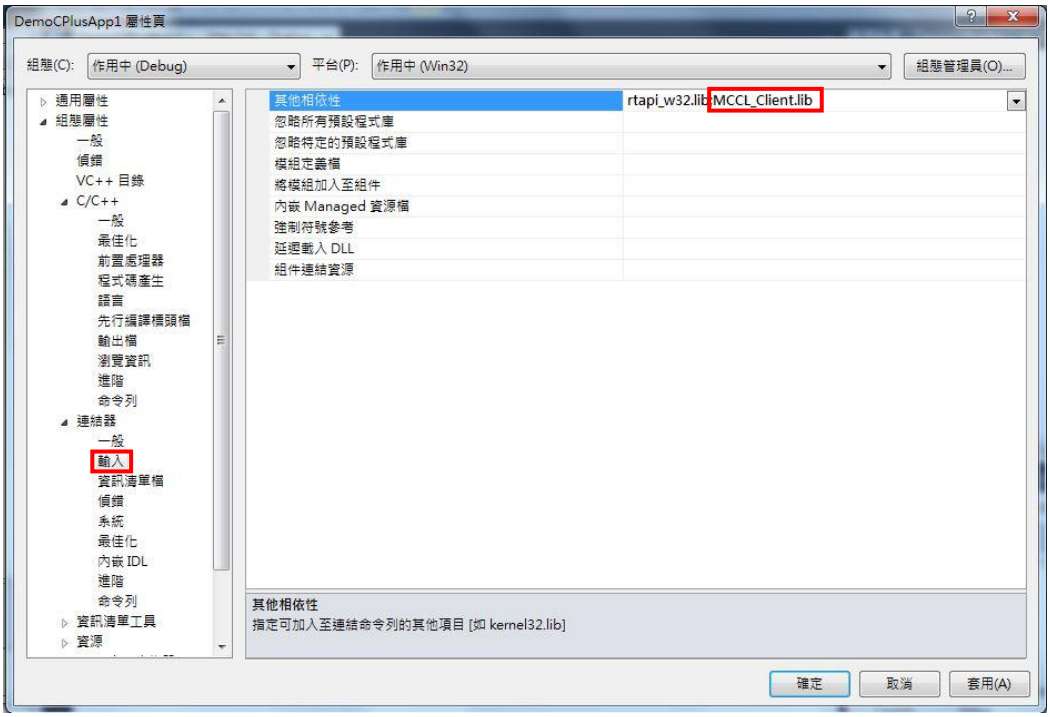


Figure 3.1.9 其他相依性設定

➢ Step 10：在原始程式碼鍵入須引用的標頭檔 MCCL.h 與 MCCL_RTX.h，即可開始使用 MCCL 開發程式。

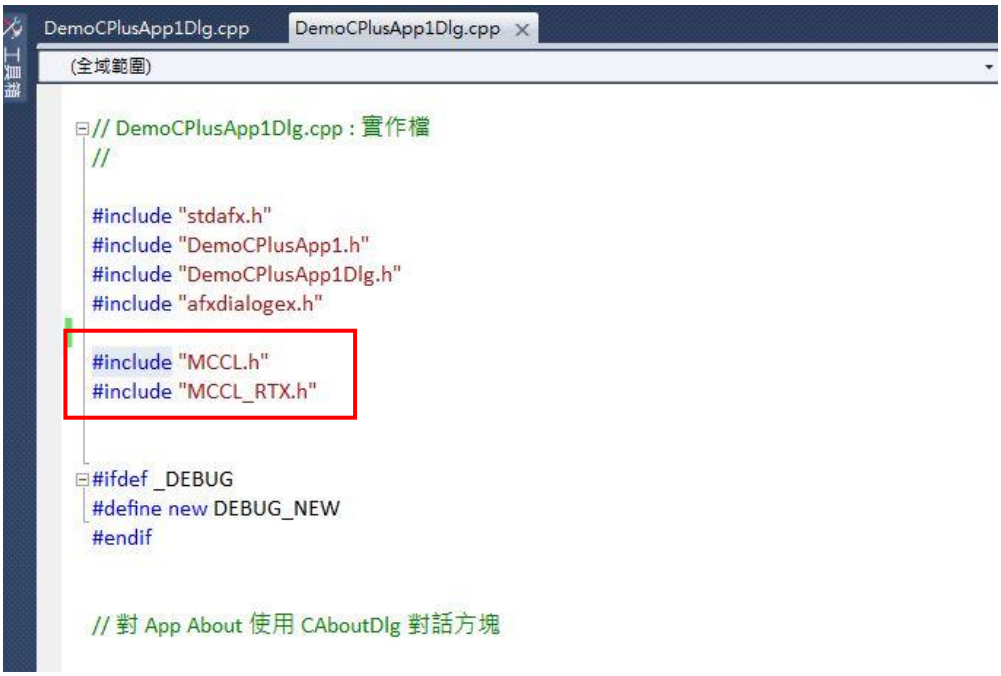
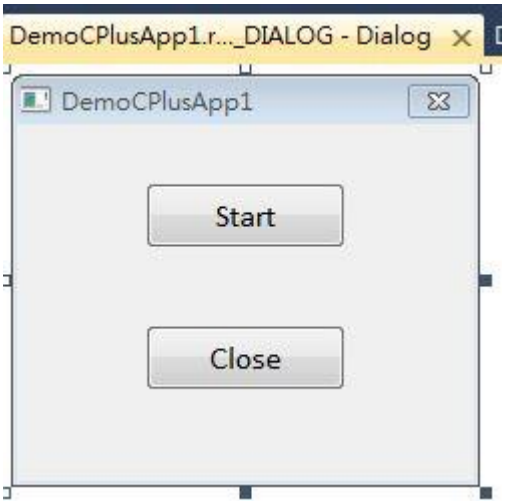


Figure 3.1.10 引用標頭檔

➢ Step 11：在人機介面上新增兩按鈕：Start 與 Close 按鈕。

Start 為呼叫 MCC_StartEcServer()、MCC_RtxInit()，進行啟動 EcServer、建立與 EcServer 通訊初始化的功能，後續可直接呼叫 MCCL 函式；Close 為呼叫 MCC_RtxClose()以關閉與 EcServer 之通訊。




```
void CDemoCPlusApp1Dlg::OnBnClickedBtnStart()
{
    BOOL nStartEcServer = MCC_StartEcServer(1);
    if(nStartEcServer==1)
    {
        int nRet=MCC_RtxInit(MAX_NUM_OF_DEIVERS);
    }
    else
        AfxMessageBox("Starting EcServer is Fail!",MB_OK);
}

void CDemoCPlusApp1Dlg::OnBnClickedButton3()
{
    MCC_RtxClose();
}
```

Figure 3.1.11 人機介面與程式碼

透過以上設定，應用程式才能透過 EcServer 呼叫 MCCL 下達運動命令、讀取目前命令位置、EtherCAT I/O 狀態與原點復歸等等，MCCL 的函式用法請參考【IMP Series 運動控制函式庫參考手冊】。

3.2 Microsoft Visual Studio 2010 C#.NET專案環境設定

➢ Step 1：開啟 Visual Studio 2010，功能表選單中點選【檔案】→【新增】→【專案】→【Visual C#】→【Windows】→【Windows Form 應用程式】，鍵入專案名稱：【DemoCSharpApp1】，點擊確定按鈕。

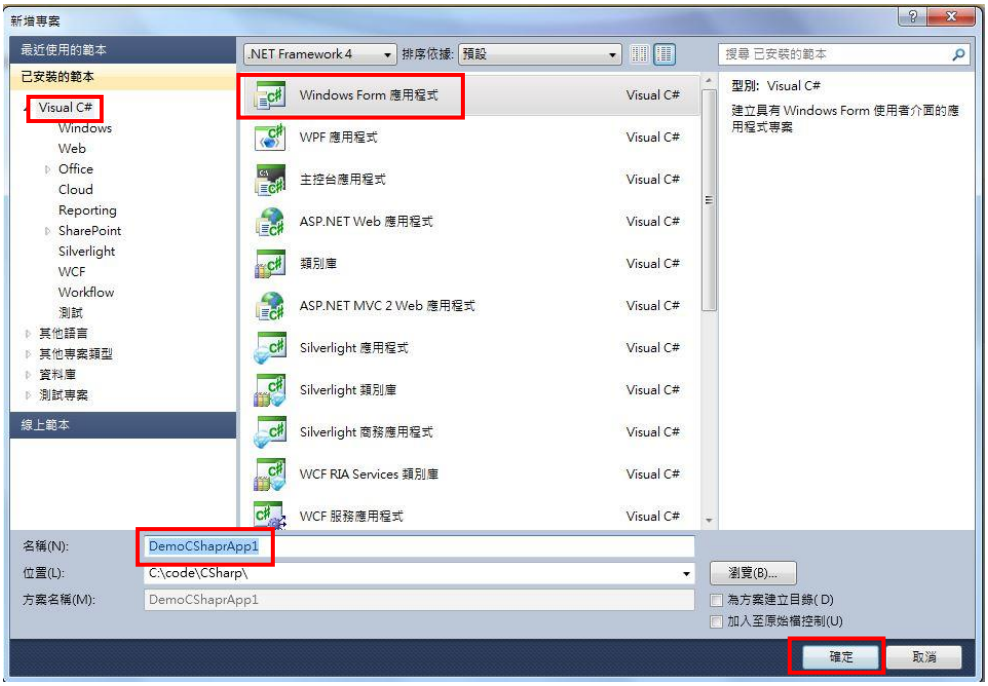


Figure 3.2.1 C#.NET 專案範本設定

➢ Step 2：複製動態函式庫與 EcServer 應用程式，將 MCCL_Client.dll、EcServer.rtss 複製至專案執行檔的資料夾。若為 64-bit 系統，請將網路卡動態函式庫：EcMaster.rtdll、emllRTL8169.rtdll、emllI8254x.rtdll 一併放入。

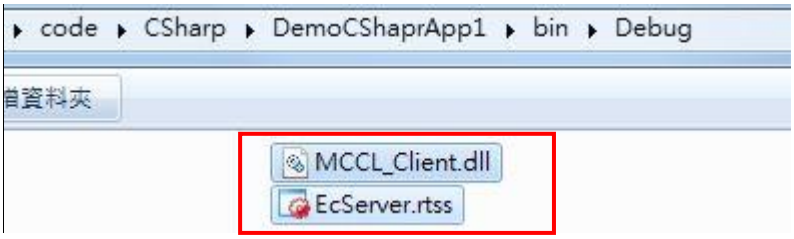


Figure 3.2.2 複製動態函式庫與 EcServer 應用程式

➢ Step 3：將 MCCL.cs 複製至專案原始碼的資料夾，在專案上點擊右鍵【Add】→【Existing Item…】，選取 MCCL.cs 後，點擊 Add 按鈕。

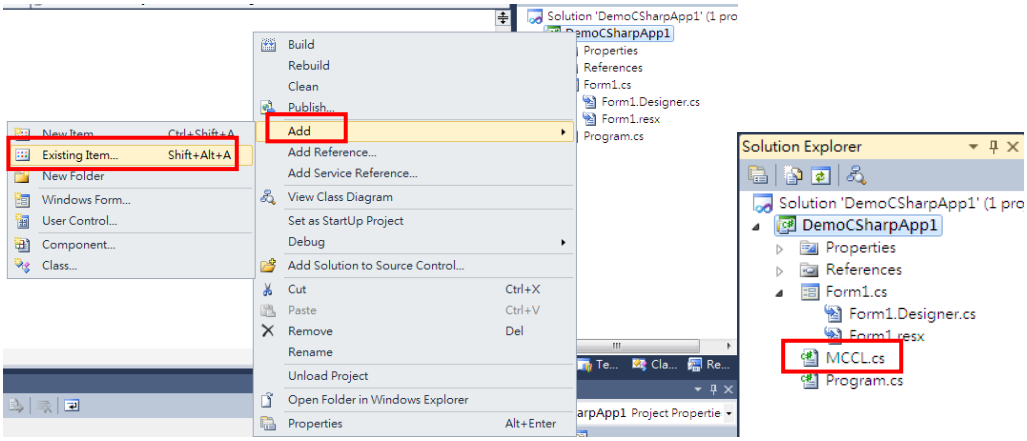


Figure 3.2.3 加入檔案

➢ Step 4：在原始程式碼鍵入須引用的命名空間 EtherCATSeries，即可開始使用 MCCL 開發程式。

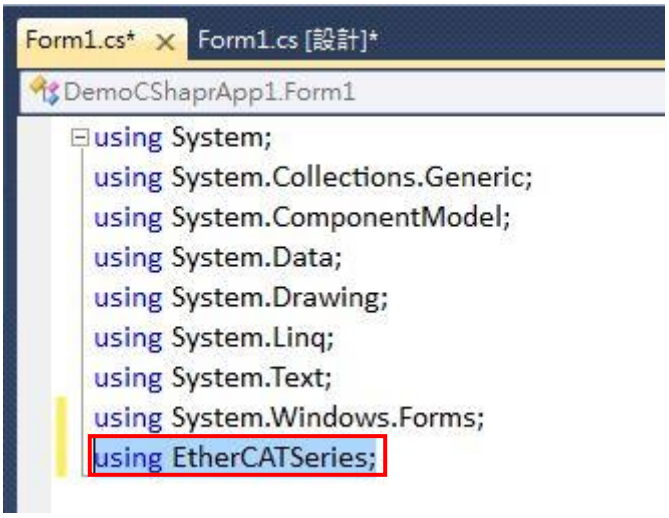
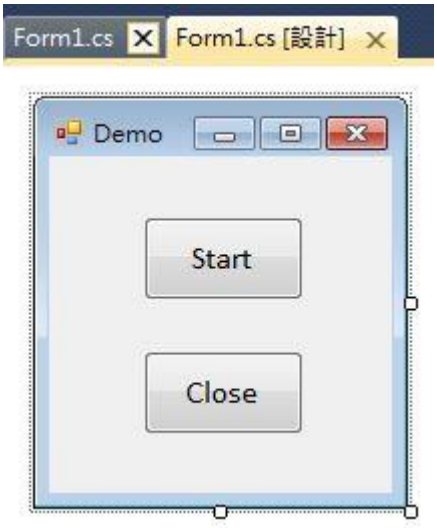


Figure 3.2.4 引用命名空間

➢ Step 5：在人機介面上新增兩按鈕：Start 與 Close 按鈕。

Start 為呼叫 MCC_StartEcServer()、MCC_RtxInit()，進行啟動 EcServer、建立與 EcServer 通訊初始化的功能，後續可直接呼叫 MCCL 函式；Close 為呼叫 MCCL.MCC_RtxClose()以關閉與 EcServer 之通訊。




```
int MAX_NUM_OF_DEIVERS = 8;
private void btStart_Click(object sender, EventArgs e)
{
    int nStartEcServer = MCCL.MCC_StartEcServer(1);
    if (nStartEcServer == 1)
    {
        int nRet = MCCL.MCC_RtxInit(MAX_NUM_OF_DEIVERS);
    }
    else
        MessageBox.Show("Starting EcServer is Fail!");
}

private void btClose_Click(object sender, EventArgs e)
{
    MCCL.MCC_RtxClose();
}
```

Figure 3.2.5 人機介面與程式碼

透過以上設定，應用程式才能透過 EcServer 呼叫使用 MCCL 下達運動命令、讀取目前命令位置、EtherCAT I/O 狀態與原點復歸等，MCCL 的函式用法請參考【IMP Series 運動控制函式庫參考手冊】。

4 EtherCAT 相關函式與操作說明

A. 系統功能

---

1. BOOL MCC_StartEcServer (

int *SleepSec*

)

Description 啟動 EcServer 應用程式(檔名：EcServer.rtss)，檔案存放位置須與開發的執行檔同一目錄(若為 64-bit 系統，網路卡動態函式庫：EcMaster.rtdll、emllRTL8169.rtdll、emllI8254x.rtdll 亦放置在此目錄)。

Parameters	*nSleepSec*	啟動 EcServer 應用程式的延遲秒數
Return Value	0	失敗
	1	成功

---

2. BOOL MCC_StartEcServerEx (

LPCTSTR *lpEcServerPath*,

int *SleepSec*

)

Description 啟動 EcServer 應用程式。

Parameters	*lpEcServerPath*	存放欲啟動 EcServer 應用程式的絕對路徑與檔案名稱，如：”C:\\RTX\\EcServer.rtss”，(若為 64-bit 系統，網路卡動態函式庫：EcMaster.rtdll、emllRTL8169.rtdll、emllI8254x.rtdll 亦放置在相同目錄)
	nSleepSec	啟動 EcServer 應用程式的延遲秒數

Return Value	0	失敗
	1	成功

---

3. int MCC_RtxInit(

int *nAxis*

)

Description	建立與 EcServer 之通訊。	
Parameters	*nAxis*	串接 EtherCAT 驅動器軸數
Return Value	0	成功
	2	RTX 初始化失敗

---

4. int MCC_RtxClose()

Description	關閉且停止與 EcServer 之通訊。	
Return Value	0	成功

---

5. BOOL MCC_EcatEnableWatchDog(

int *nExpireTime*

)

Description	設定系統 Watch Dog 時間，必須在 MCC_RTXInit()之後呼叫；EcServer 會判斷 Windows 程式在 *nExpireTime* 時間內無回應後終止。	
Parameters	*nExpireTime*	Watch Dog 的倒數計時時間，單位為 1ms
Return Value	1	成功
	0	失敗，MCC_RTXInit()尚未呼叫

---

6. BOOL MCC_EcatDisableWatchDog()

Description	取消設定系統 Watch Dog 時間，必須在 MCC_EcatEnableWatchDog()之後呼叫。	
Return Value	1	成功
	0	失敗，未呼叫 MCC_EcatEnableWatchDog()

---

7. int MCC_CreateGroup(

int *xMapToCh*,

int *yMapToCh*,

int *zMapToCh*,

int *uMapToCh*,

int *vMapToCh*,

int *wMapToCh*,

int *aMapToCh*,

int *bMapToCh*,

int *xMapToCh1*,

int *yMapToCh1*,

int *zMapToCh1*,

int *uMapToCh1*,

int *vMapToCh1*,

int *wMapToCh1*,

int *aMapToCh1*,

int *bMapToCh1*,

int *nCardIndex*

)

Description　此函式用來建立一個新的運動群組。

在呼叫 MCCL 中與運動群組有關的函式（如：MCC_Line）之前，必須先利用此函式建立群組，並得到新建立群組之編號，作為其傳入參數（之一）。

此函式必須在初始化 MCCL（MCC_InitSystem）前始可呼叫；又，在第一次呼叫此函式前請先呼叫 MCC_CloseAllGroups。注意，任兩運動軸，不可對應到同一實體輸出 Channel。

Parameters　*xMapToCh~bMapToCh1*　指定此 Group 中 X~B1 軸所對應之實體輸出 Channel(0 ~ 15)

nCardIndex　指定此群組所對應之運動控制卡編號(0 ~ 5)

若運動軸並不對應到實體軸，則須傳入 AXIS_INVALID(-1)

Return Value　大於等於 0　新建立之群組編號

小於 0　失敗，傳回值的意義請參考 **IV.函式傳回值**

---

8. BOOL MCC_EcatReset(

DWORD *dwSlaveId*

)

Description　重置 EtherCAT 驅動器。

Parameters　*dwSlaveId*　EtherCAT Slave ID

Return Value　1　成功

0　失敗

B. 原點復歸功能

1. int MCC_EcatSetHomeAxis(

BYTE *byAxisX*, BYTE *byAxisY*, BYTE *byAxisZ*, BYTE *byAxisU*,

BYTE *byAxisV*, BYTE *byAxisW*, BYTE *byAxisA*, BYTE *byAxisB*,

BYTE *byAxisX1*, BYTE *byAxisY1*, BYTE *byAxisZ1*, BYTE *byAxisU1*,

BYTE *byAxisV1*, BYTE *byAxisW1*, BYTE *byAxisA1*, BYTE *byAxisB1*

)

Description	設定欲執行原點復歸的運動軸。	
Parameters	*byAxisX~ byAxisB1*	欲進行原點復歸的運動軸設定值為 1，不執行原點復歸動作的運動軸，須設定為 0
Return Value	1	成功
	0	失敗

---

2. int MCC_EcatSetHomeMode(

int *nMode*,

int *nChannel*

)

Description	設定各軸原點復歸模式。	
Parameters	*nMode*	原點復歸使用模式，請依照 Home Sensor、實際配線與搭配的所選的 EtherCAT 驅動器廠牌而定(請參考該驅動器手冊進行配線與模式設定)
	nChannel	EtherCAT 驅動器的輸出 Channel(0 ~ 15)
Return Value	1	成功
	0	失敗

3. int MCC_EcatSetHomeZeroSpeed(

int *nZeroSpeed*,

int *nChannel*

)

Description	設定各軸原點復歸時尋找零點(Zero)的速度。	
Parameters	*nZeroSpeed*	尋找零點(Zero)的速度
	nChannel	EtherCAT 驅動器的輸出 Channel(0 ~ 15)
Return Value	1	成功
	0	失敗

---

4. int MCC_EcatSetHomeSwitchSpeed(

int *nSwitchSpeed*,

int *nChannel*

)

Description	設定各軸原點復歸時尋找原點開關(Switch)的速度。	
Parameters	*nSwitchSpeed*	尋找原點開關(Switch)的速度
	nChannel	EtherCAT 驅動器的輸出 Channel(0 ~ 15)
Return Value	1	成功
	0	失敗

---

5. int MCC_EcatHome()

Description	開始執行原點復歸。使用此函式時，可配合呼叫 MCC_EcatGetGoHomeStatus()，來檢查原點復歸是否完成。	
Return Value	1	成功
	0	失敗

6. BOOL MCC_EcatGetGoHomeStatus()

Description	在呼叫 MCC_EcatHome()後，使用此函式可檢視是否已完成原點復歸動作。	
Return Value	1	完成原點復歸動作
	0	尚未完成原點復歸動作

---

7. BOOL MCC_EcatAbortHome()

Description	當呼叫 MCC_EcatHome()後，使用此函式可停止原點復歸的動作。	
Return Value	1	成功
	0	失敗

■ **EtherCAT 原點復歸相關函式使用順序：**

1. MCC_EcatSetHomeAxis()
2. MCC_EcatSetHomeMode()
3. MCC_EcatSetHomeZeroSpeed()
4. MCC_EcatSetHomeSwitchSpeed()
5. MCC_EcatHome()
6. 可以使用 MCC_EcatGetGoHomeStatus() 取得目前原點復歸的狀態，使用 MCC_EcatAbortHome()停止原點復歸的動作

C. IO功能

---

1. int MCC_EcatSetOutput(

DWORD *dwSlaveId*,

DWORD *dwOutData*

)

Description	設定 ITRI EtherCAT IO 模組的 16 點輸出訊號狀態值。	
Parameters	*dwSlaveId*	EtherCAT IO 模組的 Slave ID
	dwOutData	指定 EtherCAT IO 模組 16 點輸出訊號狀態(bit 0 ~ bit 15 分別代表第 0 點到第 15 點的狀態)
Return Value	1	成功
	0	失敗

---

2. int MCC_EcatGetOutput(

DWORD *dwSlaveId*,

DWORD **pdwOutData*

)

Description	讀取 ITRI EtherCAT IO 的 16 點輸出訊號狀態值。	
Parameters	*dwSlaveId*	EtherCAT IO 模組 Slave ID
	pdwOutData	指向一 DWORD 值，用來存放 EtherCAT IO 模組的 16 點輸出訊號狀態(bit 0 ~ bit 15 分別代表第 0 點到第 15 點的狀態)
Return Value	1	成功
	0	失敗

3. int MCC_EcatGetInput(

DWORD *dwSlaveId*,

DWORD **pdwInData*

)

Description	讀取 ITRI EtherCAT IO 模組的 16 點輸入訊號狀態值。	
Parameters	*dwSlaveId*	EtherCAT IO 模組的 Slave ID
	pdwInData	指向一 DWORD 值，用來存放 IO 模組的 16 點輸入訊號狀態(bit 0 ~ bit 15 分別代表第 0 點到第 15 點的狀態)
Return Value	1	成功
	0	失敗

---

4. int MCC_EcatSetOutputEnqueue(

DWORD *dwSlaveId*,

DWORD *dwOutData*

)

Description	設定 ITRI EtherCAT IO 模組的 16 點輸出訊號狀態值存放至運動命令佇列(Queue)中。成功呼叫此函式將增加運動命令的庫存數目。	
Parameters	*dwSlaveId*	EtherCAT IO 模組的 Slave ID
	dwOutData	指定 IO 模組 16 點輸出訊號狀態(bit 0 ~ bit15 分別代表第 0 點到第 15 點的狀態)，可同時輸出
Return Value	大於或等於零	MCCL 給予此運動命令的編碼
	小於零	失敗，傳回值的意義請參考 **IV.函式傳回值**

D. SDO功能

---

1. int MCC_EcatCoeSdoUpload(

DWORD *dwSlaveId*,

WORD *wObIndex*,

BYTE *byObSubIndex*,

BYTE* *pbyData*,

DWORD *dwDataLen*,

DWORD* *pdwOutDataLen*

)

Description	執行 EtherCAT SDO 讀取命令。	
Parameters	*dwSlaveId*	EtherCAT Slave ID
	wObIndex	欲讀取的 CoE Object 之 Index
	byObSubIndex	欲讀取的 CoE Object 之 Sub Index
	pbyData	指向一 BYTE 值，用來存放欲讀取資料之指標
	dwDataLen	欲讀取資料之長度
	pdwOutDataLen	指向一 DWORD 值，用來存放實際讀取資料之長度
Return Value	0	成功
	非零	失敗，傳回值的意義請參考 EC-Master_ClassB.pdf 4.2 Error Codes

---

2. int MCC_EcatCoeSdoDownload(

DWORD *dwSlaveId*,

WORD *wObIndex*,

BYTE *byObSubIndex*,

BYTE* *pbyData*,

DWORD *dwDataLen*

)

Description	執行 EtherCAT SDO 寫入命令。	
Parameters	*dwSlaveId*	EtherCAT Slave ID
	wObIndex	欲寫入的 CoE Object 之 Index
	byObSubIndex	欲寫入的 CoE Object 之 Sub Index
	pbyData	指向一 BYTE 值，用來存放欲寫入資料之指標
	dwDataLen	欲寫入資料之長度
Return Value	0	成功
	非零	失敗，傳回值的意義請參考 EC-Master_ClassB.pdf 4.2 Error Codes

---

3. BOOL MCC_EcatSetInverterPowerOn(

BOOL *bEnable*

)

Description	啟用(宏盛客製化功能)。	
Parameters	*bEnable*	1 表示啟用，0 表示關閉
Return Value	1	成功
	0	失敗
