

MCN8032P 系列網路型軸卡

使用手冊

Version 1.0

2008. 6

著作權屬於和椿科技股份有限公司
版權所有

本使用手冊可能會因情況而修改，本公司將會於新版印刷時更新；但並不另行通知。

本公司依著作權法，享有及保留一切著作專屬之權利，未經本公司允許，不得已任何形式增減、改編、複製本使用手冊。

Aurotek 及 Aurotek 標誌為和椿科技股份有限公司所有

目 錄

1 簡介	1
1.1 概述	1
1.2 特點	2
2. 基板外型尺寸	3
3 安裝	4
3.1 開啓包裝	4
3.2 硬體設定及安裝	4
3.3 驅動程式安裝	5
3.4 函式庫安裝	9
3.4.1 用 Microsoft Visual C++開發應用程式時	9
3.5 軟體規格	9
3.5.1 作業環境	9
3.5.2 程式構成	9
4 接線	10
4.1 通信線連接	10
4.2 通信線	11
4.3 節點裝置 ID 設定	11
4.4 節點裝置	12
5 函式功能	13
5.2 函式說明	15
6 範例程式	34

1 簡介

1.1 概述

感謝您購買本公司之 MCN8032P 系列網路型軸卡。

MCN8032P/ MCN8016P/ MCN8008P 系列是專為 Panasonic RTEX 網路系統所開發的 PCI 網路控制卡，可以用來控制 Panasonic A4N 網路型伺服馬達，及和椿科技所提供的 I/O 模組，每一個伺服馬達或 I/O 模組各佔有一個 ID 位址，而每片網路控制卡因型號不同，分別可控制 32/16/8 個 ID 位址(節點)，如此一個 RTEX 網路便可構成一個自動化系統。

RTE 網路系統是一個環型(Ring)主-從架構的網路系統，採用專用的通信協定，可以達到即時通信效果。在通信的物理層則是採用與乙太網路相同的物理層控制 IC，以及標準的有遮蔽對絞網路線，因此最高通信速度可達 100Mbps，同時又具有省配線，及配線成本低廉等多項的優點。

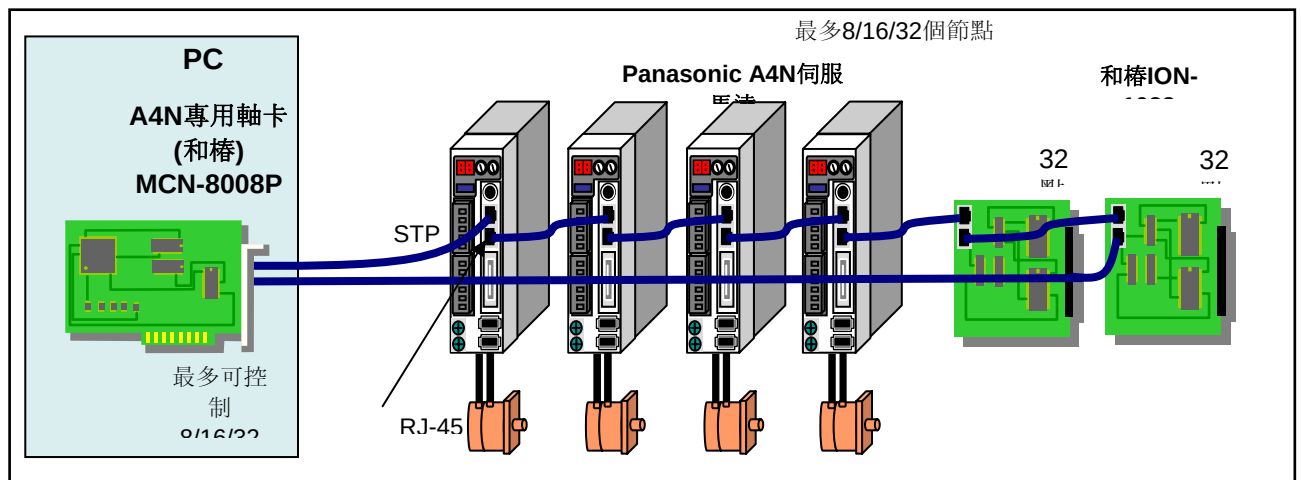


圖1.1 A4N網路架構

1.2 特點

Panasonic RTEX 伺服網路系統具有下列優點：

- 全雙工通信
- 通信速度可達 100Mbps
- 環型網路拓撲
- 最多可控制 32 個節點
- 可同時控制伺服馬達，及 I/O

MCN-8032P 系列控制卡除了具備 RTEX 伺服網路的優點外，在運動控制上更具備有直線補間，圓弧補間，以及為位置比較輸出等功能，可以滿足使用者在自動化設備運用的需求。

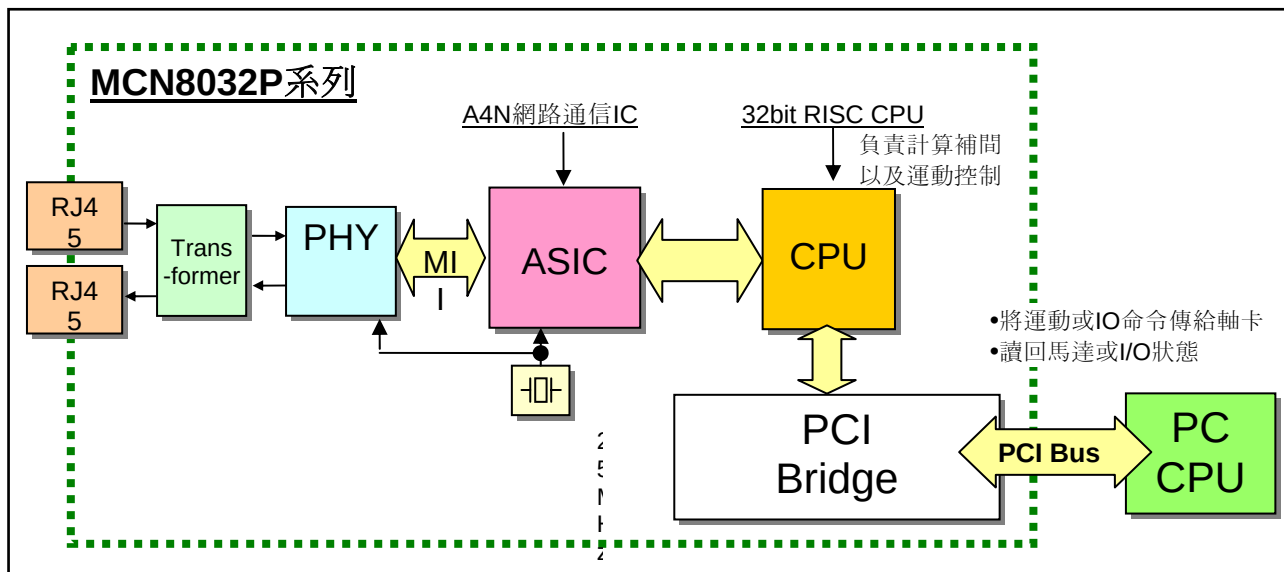
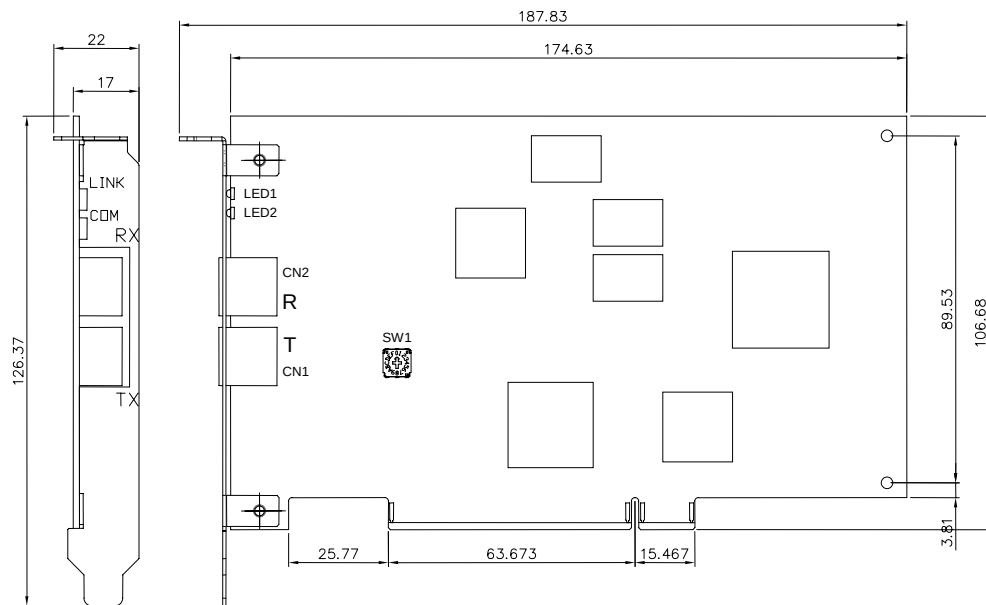


圖1.2 MCN8032P方塊圖

2. 基板外型尺寸



CN1(TX) : 傳送用接頭

CN2(RX) : 接收用接頭

LED1(LINK) : 連線狀態顯示燈
 綠色燈號：通信電纜連接正常
 熄燈：通信電纜沒有連接
 ※此燈號並非用來表示通信確立。

LED2(COM) : 通信狀態顯示燈
 綠色燈號：通信確立
 綠燈閃爍：通信準備中
 紅色燈號：通信異常
 熄燈：等待軟體開啓控制卡，或是控制卡異常

SW1 : 同一台電腦使用複數板時的 ID 設定開關，
 可以設定 ID 為 0~F。

3 安裝

本章說明本網路型軸卡的基本包裝及附件清單，以及標準的安裝步驟。請依照指示按照標準步驟安裝軟體及硬體。

3.1 開啓包裝

收到 MCN8032P / MCN8016P / MCN8008P 包裝後，請確認是否有下列的物品：

- ☒ MCN8032P / MCN8016P / MCN8008P 網路型運動控制卡
- ☒ 安裝光碟(包含驅動程式，DLL 函式庫，操作說明)

MCN8032P 系列控制卡上有一些易受到靜電放電(ESD) 傷害的電子零件。如果沒有做好預防的措施，靜電放電可能會損壞積體電路和某些組成部分。在從抗靜電的塑膠袋中取出此卡片之前，你應該採取遵循預防靜電放電傷害：

- 用手接觸電腦機殼的金屬零件，或是利用一根接地線，導掉你的身體上累積的靜電。
- 在打開袋之前，將抗靜電的袋與電腦機殼的金屬零件接觸。
- 在從抗靜電的塑膠袋中取出此卡片時，只抓住卡片金屬部分。

● **在取出這張卡片之後，首先你應該：**

檢查這張卡片是否有外部損害(如鬆掉或損壞的零件，等等)的跡象，如果這張卡片有明顯損壞，請立即通知我們的服務部或者我們的本地銷售代表，避免將一張損壞的卡片安裝進你的電腦系統。

● **請特別注意下列幾點，以確保能妥善的安裝：**

- 避免卡片與易產生靜電的材料(例如塑膠，乙烯樹脂和發泡苯乙烯)接觸。
- 每當拿取這張卡片時，只拿取它的邊緣部份；請不要觸碰連接器或電子零件器件的金屬部份。

注意：

請妥善保管此抗靜電袋。當你需要運送此卡片時，請將卡片放入原先的抗靜電袋以避免靜電放電傷害。

3.2 硬體設定及安裝

請依照下列步驟安裝控制卡到電腦中：

Step 1: 關閉電腦電源，並拔下電腦的電源線。

Step 2: 打開電腦外殼，取下插槽套。

Step 3: 用手碰觸電腦金屬部分，消除身體可能的靜電。

Step 4: 設定控制卡的 ID 號碼。

Step 5: 將本製品插入電腦空的 PCI 擴充插槽。請渾柱控制卡的邊緣順著插槽插入，並確認控制卡已經穩固地插入插槽，請不要過度用力擠壓控制卡，否則可能會造成軸卡的損壞。

Step 6: 請將控制卡的固定金屬片用螺絲固定。如果未用螺絲固定，可能會造成控制卡掉落，控制卡短路故障或事誤動作等原因。

Step 7: 請依照原先順序蓋回電腦外殼。

注意：電腦硬體安裝作業請務必在切斷電源下進行。若未依照此程序可能會破壞電路零件等。

<使用複數控制卡時>

在同一台電腦上使用多張本控制卡時，爲了分辨 PCI 插槽上個別的控制卡，同時插上兩張卡以上時必須利用控制卡上的旋轉開關來設定控制卡的 ID 號碼，ID 號碼可設定 0~F，在同一台電腦中的控制卡不可設定相同的 ID 號碼。

3.3 驅動程式安裝

本驅動程式針對各種語言版本的 Windows 2000/XP 安裝過程均相同。本驅動程式可以同時支援 16 張控制卡。

安裝驅動程式必須以擁有 Administrator 權限的 USER 登錄才可安裝，如果使用其他權限等級的使用者登錄是無法正常安裝的，請注意這一點。

下面是以 Windows XP 爲例說明，基本上在 Windows 2000 的環境下操作亦相同。

1. 首先確認控制卡已經依照 3.2 項確實安裝製電腦主機中。
2. 開啓電腦電源，啓動 Windows XP。
3. 請以 Administrator 權限的 USER 登錄電腦。

電腦開啓後會出現「找到新硬體」安裝中的視窗，之後會接著出現「尋找新硬體精靈」的對話框，請接著按下一步

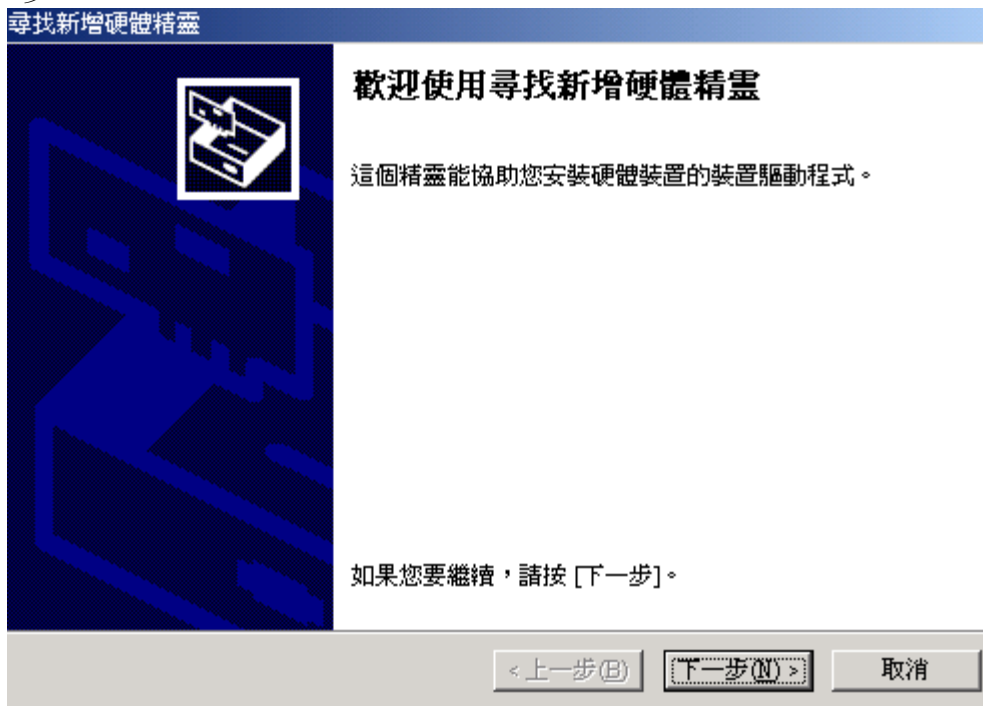


圖 3.1 新增硬體精靈

將 MCN-8032P 插入電腦的 PCI 插槽，開機登入 Windows 後，出現如圖 3.2 的畫面。按【下一步】。

尋找新增硬體精靈

安裝硬體裝置的驅動程式

裝置驅動程式是一個能在作業系統中啓用硬體裝置運作的軟體。



這個精靈將會完成裝置安裝：



其他 PCI 橋接器裝置

裝置驅動程式是能讓硬體裝置運作的軟體程式。Windows 需要驅動程式檔案來安裝新硬體。如果您要尋找驅動程式檔案，完成安裝的話，請按 [下一步]。

您要精靈執行什麼工作？

- ☒ 搜尋適當的裝置驅動程式檔案 (建議選項)(S)
- ☐ 請顯示這個裝置目前的驅動程式清單，讓我從清單中指定驅動程式(D)

< 上一步(B)

下一步(N) >

取消

圖 3.2a

尋找新增硬體精靈

尋找驅動程式檔

您要 Windows 從何處搜尋驅動程式檔案？



搜尋下列硬體裝置所需的驅動程式檔案：



其他 PCI 橋接器裝置

精靈會在電腦的驅動程式資料庫及以下您指定的搜尋位置中搜尋適當的驅動程式。

想要開始搜尋，請按 [下一步]。如果您將在磁片或是 CD-ROM 光碟機上搜尋，請先插入磁片或是 CD，再接 [下一步]。

選擇性搜尋位置：

- ☐ 軟式磁碟機(D)
- ☐ CD-ROM 光碟機(C)
- ☒ 指定位置(S)
- ☐ Microsoft Windows Update(M)

< 上一步(B)

下一步(N) >

取消

圖 3.2b

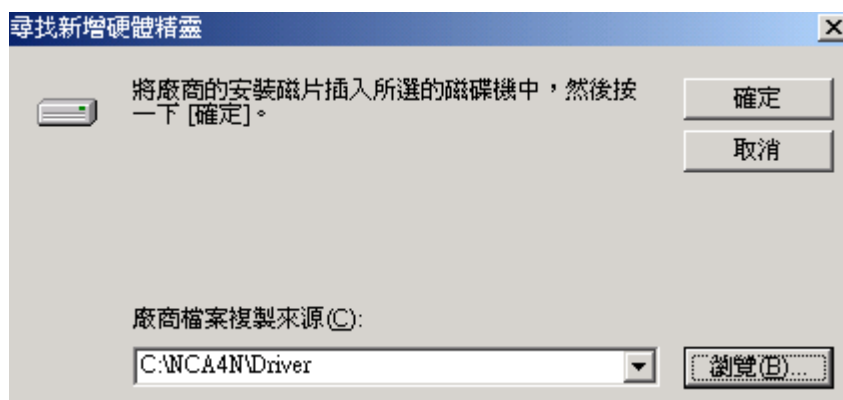


圖 3.3 指定搜尋位置

指定搜尋位置。若驅動程式位於 C:\NCA4N 之下，請選擇 C:\NCA4N\Driver。

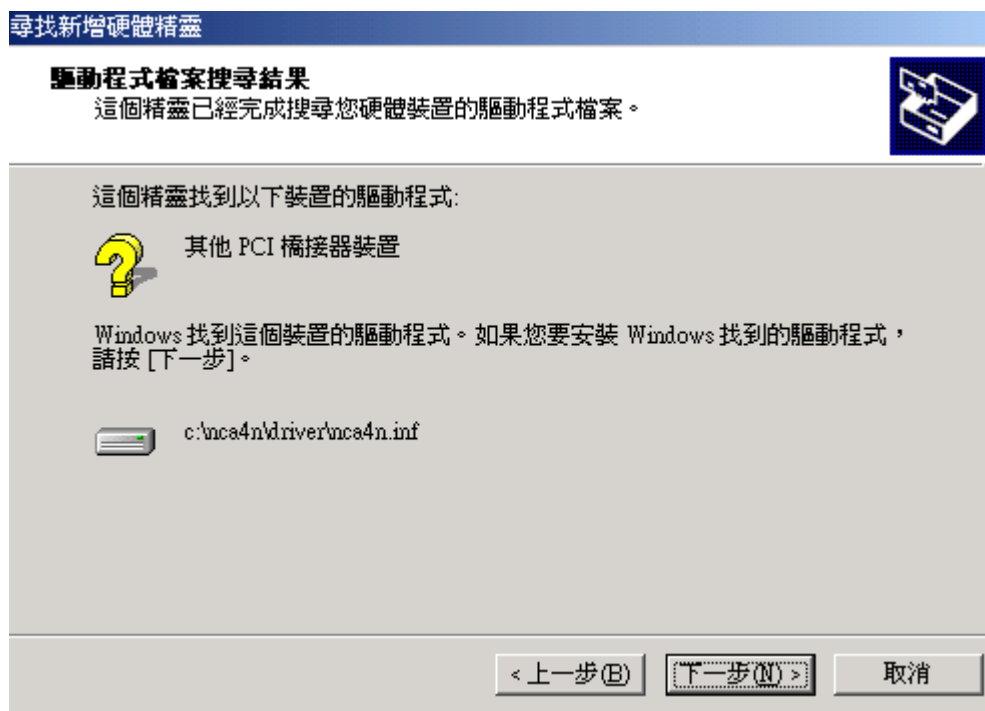


圖 3.4 找到驅動程式

Windows 會提醒找到驅動程式，按【下一步】，開始進行安裝。

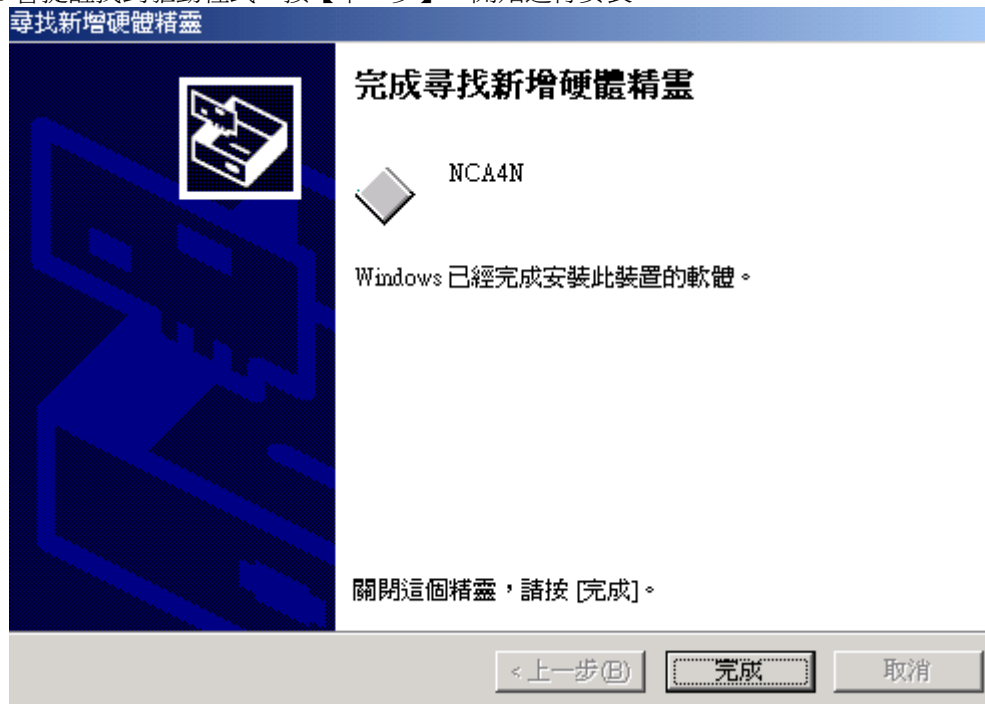


圖 3.5 新增硬體完成

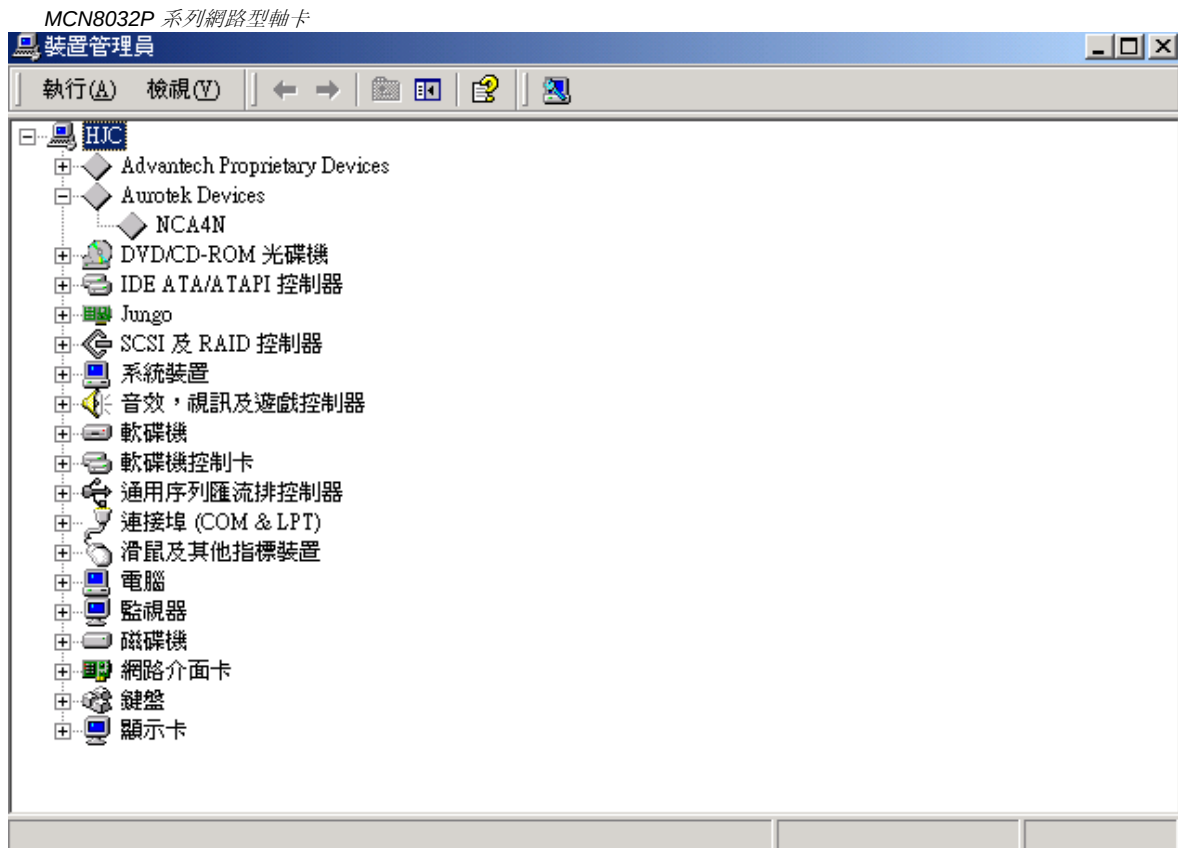


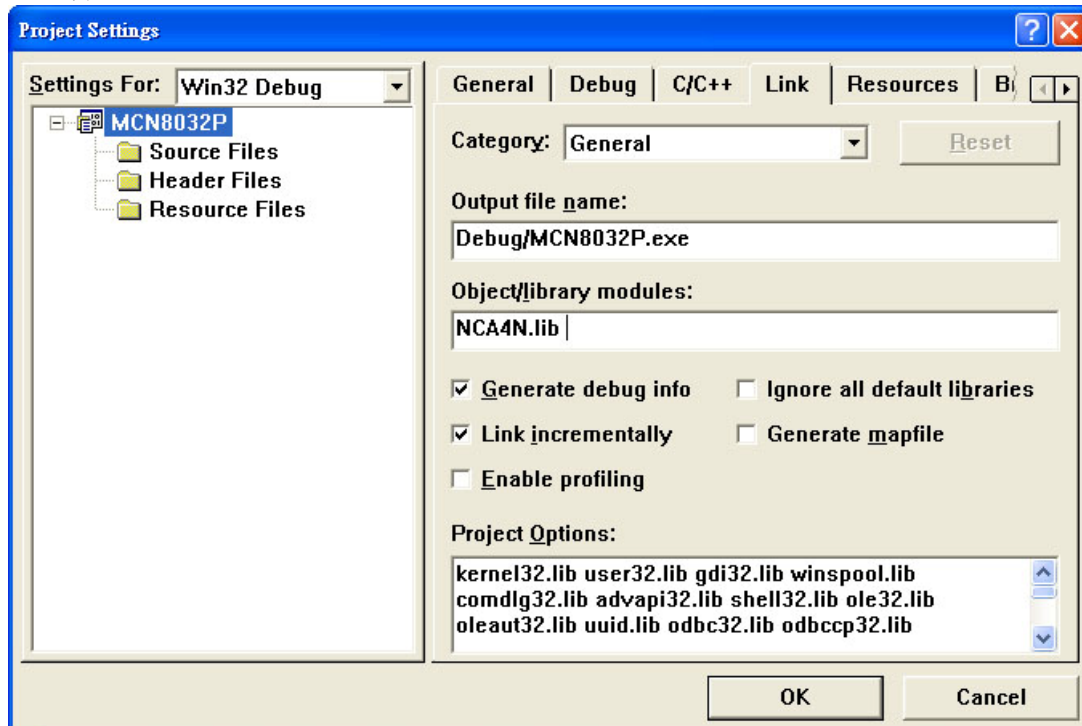
圖 3.6 確認驅動程式安裝正常

硬體安裝完成後，請點選【控制臺】→【系統】→【裝置管理員】，點選 Aurotek Devices，若出現 NCA4N，則安裝完成。

3.4 函式庫安裝

3.4.1 用 Microsoft Visual C++(以下稱 VC++)開發應用程式時

1. 請將 Lib 目錄夾中的 2 個檔案 NCA4N.lib 和 NCA4N.h 複製到開發程式所在的目錄夾下。
2. 請在 VC++ 的開發環境下將 NCA4N.h 登錄在使用的專案下使用。
3. 在「Project(專案)->Setting(設定)」打開對話盒，選擇 Link 頁面”Object/Library” 並指定包含 NCA4N.lib。



注意：此兩個檔案是對應 VC++6.0 以後的版本。

3.5 軟體規格

3.5.1 作業環境

對應 OS：Windows2000、WindowsXP

對應言語：Microsoft Visual C++ 6.0 之後版本

3.5.2 程式構成

程式構成種類	目錄夾	檔案名	說明
驅動程式	Driver	NCA4N.SYS	Windows2000,XP 用裝置驅動程式
		NCA4N.DLL	動態連結函式庫
函式庫	Lib	NCA4N.LIB	為使用 NCA4N.DLL 的函式庫，VC++專用
		NCA4N.H	為使用 NCA4N.DLL 的標頭檔，VC++專用
VC++範例程式	Sample	一式	控制本卡的 VC++6.0 範例程式

備考：此說明省略了只在驅動程式設定時所使用的檔案。

4 接線

4.1 通信線連接

RTEX 網路為一環型網路系統，整個系統的通信連接必須構成一個迴路(Loop)，先由系統的 Master 裝置(軸卡)的傳送埠(Tx)需連線到第一個裝置(如驅動器，或 I/O 模組)的接收埠(Rx)，之後再用第一個裝置的傳送埠連接第二個裝置的接收埠…如此串聯下去，最後再由最後一個裝置的傳送埠連接回 Master 裝置的接收埠。請參考下圖：

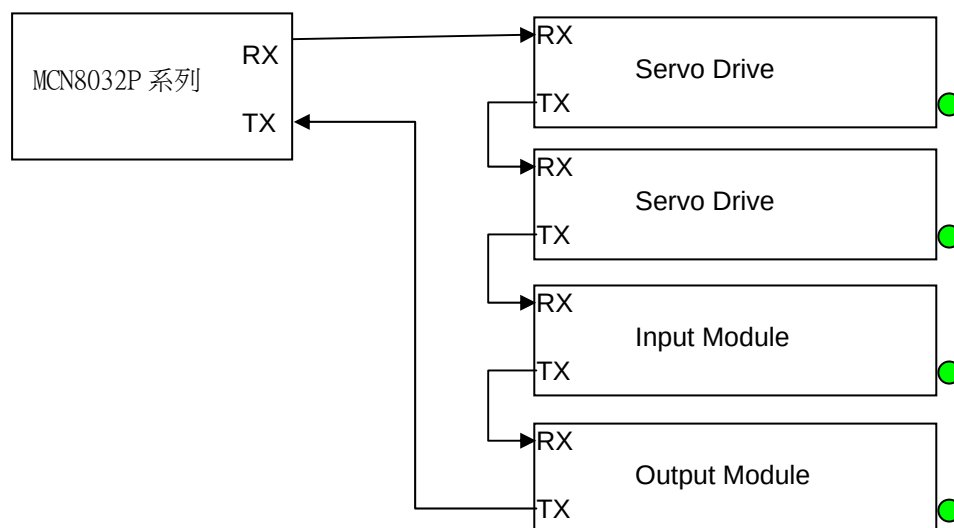


圖 4.1、通信線連接接線圖

連接正確的話，請確認每個裝置的 LINK 燈是亮綠燈，如果斷線或是接線不正確則不會亮燈，如圖 4.2。此例中第三個節點裝置的 LINK 不亮，此時應檢查第二個裝置與第三個裝置間的連線。

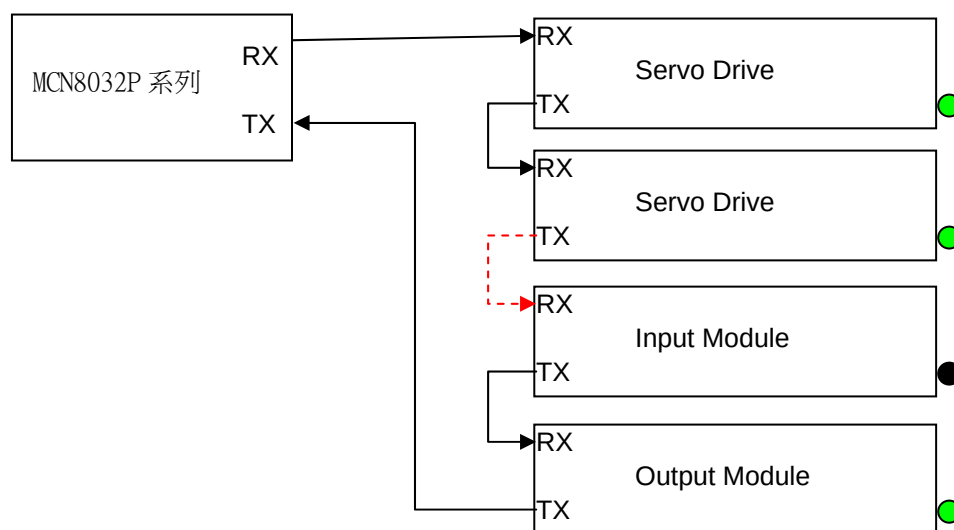


圖 4.2 通信線連接確認

4.2 通信線

1. 請使用符合Category 5e 或以上，有遮蔽之對絞線來製作通線線。
2. 遮蔽線如果沒有兩端都接地，有可能會使得通信線的EMC特性劣化，因此製作通信線時請將遮蔽線連接到接頭的金屬部分。
3. 圖4.3所標示的線色是依照TIA/EIA568B。
4. 請注意通信線的長度，節點與節點間不可超過60公尺，整個網路的總長度不可超過200公尺。

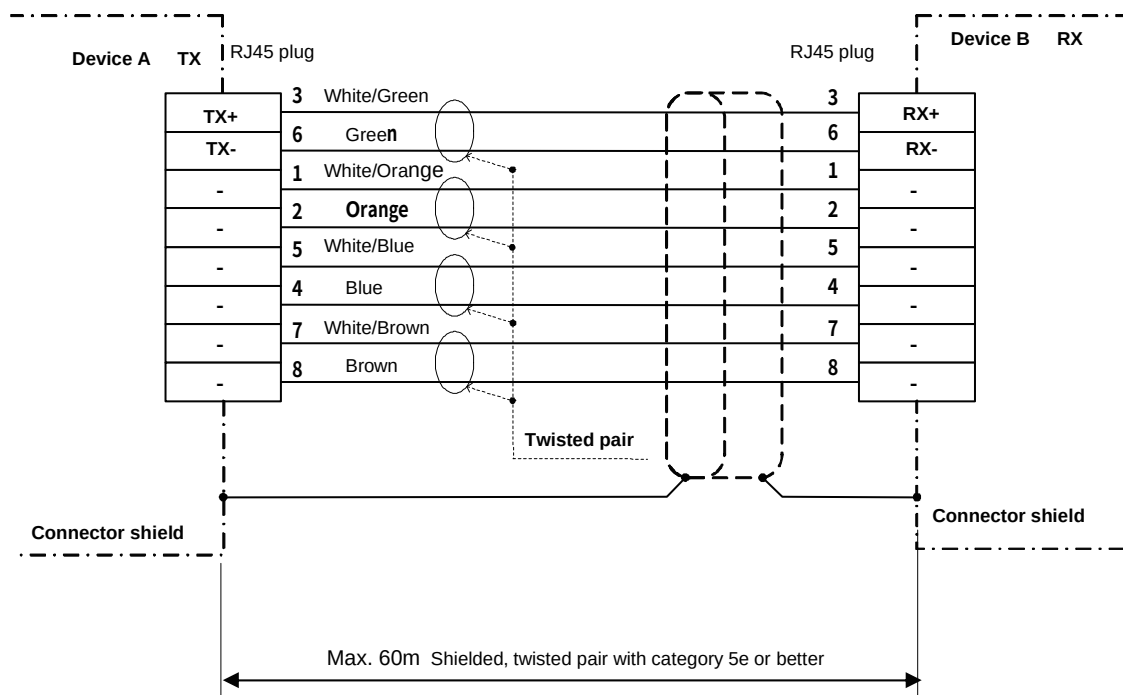


圖 4.3 通信線連接圖

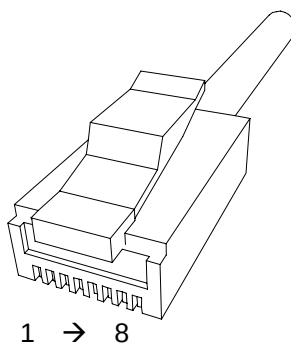


圖 4.4 RJ45 接頭腳位

4.3 節點裝置 ID 設定

在同一個網路中，每一個節點裝置(如伺服驅動器，I/O 模組)均須設定一唯一的 ID 值，請調整節點裝置的旋轉開關來設定 ID。如圖 4.5 所示，共有兩個旋鈕。左邊的旋鈕為十位數，右邊的旋鈕為個位數。可設定的數值由 0~31(視軸卡可控制節點數而定)。

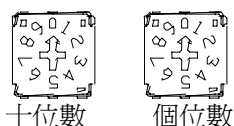


圖 4.5 節點裝置的 ID 設定鈕

4.4 節點裝置

目前可以選用搭配本卡的節點裝置有 Panasonic A4N 系列網路型伺服馬達，以及本公司的 32 點輸入模組 MCN-9321，32 點輸出模組 MCN-9322。未來我們還會陸續開發步進模組，A/D 模組，D/A 模組…等，敬請期待。

5 函式功能

5.1 函數表

函數名	說 明
Nca4n_Open	開啓控制卡
Nca4n_Close	關閉控制卡
Nca4n_CloseAll	關閉全部的控制卡
Nca4n_ServoReset	重置控制卡以及所有連接的伺服驅動器
Nca4n_AlarmReset	重置控制卡以及所有連接的伺服驅動器的警報訊息
Nca4n_GetCardVersion	讀取控制卡的軟體版本
Nca4n_GetMaxAxis	讀取控制卡可控制的最大軸數
Nca4n_GetConnectAxis	讀取控制卡可連接的軸數
Nca4n_GetLogicPos	讀取指定軸的邏輯位置
Nca4n_GetRealPos	讀取指定軸的實際位置
Nca4n_GetSpeed	讀取指定軸的實際速度 單位：脈波 / 秒
Nca4n_GetTorque	讀取指定軸的實際轉矩 單位：0.1% / bit
Nca4n_GetAxisStatus	讀取指定軸的軸狀態
Nca4n_ReadParameter	讀取指定的伺服驅動器的指定參數所設定的值
Nca4n_WriteParameter	將參數值寫入指定的伺服驅動器的指定參數中
Nca4n_EepromWriteParameter	將現在(暫時的)伺服驅動器的參數存入伺服驅動器的 EEPROM 內
Nca4n_ServoOn	指定伺服驅動器 SERVO ON
Nca4n_ServoOff	指定伺服驅動器 SERVO OFF
Nca4n_GainChange	切換指定軸伺服驅動器的增益
Nca4n_TorqueLimitChange	切換指定軸的轉矩極限
Nca4n_ExOut	指定伺服驅動器的輸出做 ON/OFF
Nca4n_SetStartSpeed	設定指定軸的初速度
Nca4n_SetDriveSpeed	設定指定軸的驅動速度
Nca4n_SetUpRate	設定指定軸的加速度
Nca4n_SetDownRate	設定指定軸的減速度
Nca4n_SetOriginSpeed1	設定指定軸的搜索原點感應器速度
Nca4n_SetOriginSpeed2	設定指定軸的搜尋到原點感應器之，再搜索 Z 相的速度
Nca4n_AbsMove	指定軸進行絕對值移動
Nca4n_RelMove	指定軸進行相對值移動
Nca4n_TriggerAbsMove	讓指定軸以絕對值移動，但在每指定脈波數到達後指定輸出點輸出一觸發信號
Nca4n_TriggerRelMove	讓指定軸以相對值移動，但在每指定脈波數到達後指定輸出點輸出一觸發信號
Nca4n_SetLineMoveAxis	指定要執行直線補間的軸。
Nca4n_ResetLineMoveAxis	用來解除 SetLineMoveAxis 所指定的直線補間軸
Nca4n_StartLineMove	指定的軸直線補間
Nca4n_CirMove	圓補間移動
Nca4n_ArcMove	弧補間移動

函數名	說 明
Nca4n_SetFirstCirFollowAxis	爲了在一主軸和多副軸的組合約時進行圓補間移動的登記最初的第 1 組的主軸和副軸。
Nca4n_SetFirstArcFollowAxis	當一個主軸和多個副軸的組合時進行圓補間移動時，用來設定最初的第 1 組的主軸和副軸。
Nca4n_SetNextFollowAxis	設定第二個以後的副軸。
Nca4n_StartFollow	同時執行圓或弧補間移動
Nca4n_SlowStop	讓指定軸減速停止
Nca4n_InstantlyStop	讓指定軸立即停止
Nca4n_GetDriveEnd	讀回指定軸的是否移動結束
Nca4n_SetPosition	變更實際位置
Nca4n_Homing	指定軸的原點覆歸
Nca4n_GetHomingStatus	原點覆歸的結束確認
Nca4n_SetSoftLimit	設定軟體極限
Nca4n_SetSoftLimitUse	設定使用/不使用軟體極限
Nca4n_SetHardLimitUse	設定使用/不使用硬體極限
Nca4n_Input	取得輸入模組 (MCN-9321)的輸入資料
Nca4n_Output	寫進到輸出模組 (MCN-9322)的輸出資料
Nca4n_BitOutput	設定輸出模組 (MCN-9322) 的位元是 ON / OFF

5.2 函式說明

函數	BOOL Nca4n_Open(short BoardNo)
說明	開啓控制卡
輸入	控制卡號碼 0-15
傳回值	TRUE: 開啓正常、FALSE: 開啓失敗
使用例	<pre> if ==FALSE) { MessageBox(" NCA4N Board Open Error ! " ," Error" ,MB _ OK) ; EndDialog(0) ; } </pre>

函數	BOOL Nca4n_Close(short BoardNo)
說明	關閉控制卡
輸入	short BoardNo: 控制卡號碼 0-15
傳回值	TRUE: 正常關閉、FALSE: 關閉失敗
使用例	Nca4n_Close(0);

函數	BOOL Nca4n_CloseAll(void)
說明	關閉全部的控制卡
輸入	沒有
傳回值	TRUE: 正常關閉、FALSE: 關閉失敗
使用例	Nca4n_CloseAll();

函數	short Nca4n_ServoReset(short BoardNo)
說明	重置控制卡以及所有連接的伺服驅動器
輸入	short BoardNo: 控制卡號碼 0-15
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓
使用例	Ret = Nca4n_ServoReset(0);

函數	short Nca4n_AlarmReset(short BoardNo)
說明	重置控制卡以及所有連接的伺服驅動器的警報訊息
輸入	short BoardNo: 控制卡號碼 0-15
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的板沒有開啓
使用例	Nca4n_AlarmReset(0);

函數	short Nca4n_GetCardVersion(short BoardNo, char *Version)
說明	讀取控制卡的韌體版本
輸入	short BoardNo: 控制卡號碼 0-15 char *Version: 用來接收傳回文字列的緩衝指標(最小 5 byte)
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的板沒有開啓
使用例	char Version[5]; (0, Version); 結果: Version[] = " 1.00"

函數	short Nca4n_GetMaxAxis(short BoardNo)
說明	讀取控制卡可控制的最大軸數
輸入	short BoardNo: 控制卡號碼 0-15
傳回值	下面之外: 最大控制軸數 (8,16,32) -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓
使用例	short MaxAxis; MaxAxis =(0) ;

函數	short Nca4n_GetConnectAxis(short BoardNo)
說明	讀取控制卡可連接的軸數
輸入	short BoardNo: 控制卡號碼 0-15
傳回值	下面之外: : 接續軸數 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓
使用例	short ConnectAxis; ConnectAxis =(0) ;

函數	short Nca4n_GetLogicPos(short BoardNo, short UnitNo, long *Pos)
說明	讀取指定軸的邏輯位置
輸入	short BoardNo: 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31 long *Pos: 用於接受傳回邏輯位置的緩衝指標
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	long Pos; Nca4n_GetLogicPos(0, 0, &Pos);

函數	short Nca4n_GetRealPos(short BoardNo, short UnitNo, long *Pos)
說明	讀取指定軸的實際位置
輸入	short BoardNo: 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31 long *Pos: 用於接受傳回實際位置的緩衝指標
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	long Pos;

Nca4n_GetRealPos(0, 0, &Pos);

函數	short Nca4n_GetSpeed(short BoardNo, short UnitNo, long *Speed)
說明	讀取指定軸的實際速度 單位：脈波 / 秒
輸入	short BoardNo: 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31 long *Speed: 用於接受傳回實際速度的緩衝指標
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	long Speed; Nca4n_GetSpeed(0, 0, &Speed);

函數	short Nca4n_GetTorque(short BoardNo, short UnitNo, long *Torque)
說明	讀取指定軸的實際轉矩 單位：0.1% / bit
輸入	short BoardNo: 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31 long *Torque : 用於接受傳回實際轉矩的緩衝指標
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	long Torque; Nca4n_GetTorque(0, 0, &Torque); 結果：Torque=100→10.0%

函數	short Nca4n_GetAxisStatus(short BoardNo, short UnitNo, long *Stat)
說明	讀取指定軸的軸狀態 Bit17 : -軟體極限 8 : CW / CCW 極限感應器狀態*2 Bit 16 : +軟體極限 7 : Servo ON 狀態 15 : 非常停止 6 : Servo Ready 狀態 14 : 外部輸入 4 狀態 5 : 報警信號狀態 13 : 外部輸入 3 狀態 4 : 警告發生 12 : 外部輸入 2 狀態 3 : 轉矩限制 11 : 外部輸入 1 狀態 2 : 原點覆歸完畢 10 : HOME 感應器狀態*1 1 : 未使用 9 : CW / CCW 極限感應器狀態*2 0 : 位置決定完畢 *1 動作方向設定 Pr.42 「近原點感應器邏輯位準設定」的設定值而定 *2 位元 9,8 是代表 CW 或 CCW 是依據參數 Pr.43「動作方向設定」的設定值而定 感應器信號，活動時候一套（輸入光電耦合器 OFF 活動）
輸入	short BoardNo: 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31 long *Stat : 對用於儲藏狀態的緩衝指標
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	<pre>long Stat; Nca4n_GetAxisStatus(0, 0, &Stat);</pre>

函數	short Nca4n_ReadParameter(short BoardNo, short UnitNo, short ParaNo, long *Data)
說明	讀取指定的伺服驅動器的指定參數所設定的值
輸入	short BoardNo: 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31 short ParaNo : 參數號碼 long *Data : 對用於儲藏設定值的緩衝指標
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	long Data; Nca4n_ReadParameter(0, 0, 0x42, &Data); 讀取近原點的感應器的邏輯位準設定值

函數	short Nca4n_WriteParameter(short BoardNo, short UnitNo, short ParaNo, long Data)
說明	將參數值寫入指定的伺服驅動器的指定參數中
輸入	short BoardNo: 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31 short ParaNo : 參數號碼 long Data: 寫進的設定值
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	Nca4n_WriteParameter(0, 0, 0x42, 1) ; 近原點感應器的邏輯設成 A 接點(常開)

函數	short Nca4n_EepromWriteParameter(short BoardNo, short UnitNo)
說明	將現在(暫時的)伺服驅動器的參數存入伺服驅動器的 EEPROM 內
輸入	short BoardNo: 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	Nca4n_EepromWriteParameter(0, 0);

函數	short Nca4n_ServoOn(short BoardNo, short UnitNo)
說明	指定伺服驅動器 SERVO ON
輸入	short BoardNo: 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31, 指定為-1 的話, 做全部軸伺服會 SERVO ON
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	Nca4n_ServoOn(0, 0); 指定模組編號 0, SERVO ON Nca4n_ServoOn(0, -1); 指定全部軸 SERVO ON

函數	short Nca4n_ServoOff(short BoardNo, short UnitNo)
說明	指定伺服驅動器 SERVO OFF
輸入	short BoardNo: 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31, 指定為-1 的話, 做全部軸伺服會 SERVO OFF
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	Nca4n_ServoOff(0, 0); 指定模組編號 0, SERVO OFF Nca4n_ServoOff(0, -1); 指定全部軸 SERVO OFF

函數	short Nca4n_GainChange(short BoardNo, short UnitNo, short GainNo)
說明	切換指定軸伺服驅動器的增益 此函數只在伺服驅動器參數 Pr.31 “增益切換模式” 的值設定為 2 時才有效
輸入	short BoardNo: 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31 short GainNo : 切換增益號碼 1 = 第 1 增益設定, 2 = 第 2 增益設定
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -5(PARAMETER _ ERR) : 參數設定值是範圍以外 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	Nca4n_GainChange(0, 0, 2);

函數	short Nca4n_TorqueChange(short BoardNo, short UnitNo, shoart TorqueNo)
說明	切換指定軸的轉矩極限 只有在伺服驅動器參數的 Pr.03 “轉矩極限選擇” 的設定值為 3 時有效
輸入	short BoardNo : 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31 short TorqueNo: 更換轉矩極限 1 = 第 1 轉矩極限 2 = 第 2 轉矩極限
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -5(PARAMETER _ ERR) : 參數的設定值是範圍以外 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	Nca4n_TorqueChange(0, 0, 2);

函數	short Nca4n_ExOut(short BoardNo, short UnitNo, short OutNo, short Mode)
說明	指定伺服驅動器的輸出做 ON/OFF
輸入	short BoardNo: 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31 short OutNo: 欲控制的輸出號碼 1 或 2 short Mode: 輸出狀態 1(EX _ OUT _ ON)=輸出 ON; 0(EX _ OUT _ OFF)=輸出 OFF
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -5(PARAMETER _ ERR) : 參數的設定值是範圍以外 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	Nca4n_ExOut(0, 0, 1, EX _ OUT _ ON); 輸出 1 設定為 ON

函數	short Nca4n_SetStartSpeed(short BoardNo, short UnitNo, long Speed)
說明	設定指定軸的初速度 初速度並不會被記憶在控制卡中，因此每當電源投入時必要再設定。
輸入	short BoardNo: 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31 long Speed: 初速度 pulse / sec
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	Nca4n_SetStartSpeed(0, 0, 1000);

函數	short Nca4n_SetDriveSpeed(short BoardNo, short UnitNo, long Speed)
說明	設定指定軸的驅動速度 驅動速度並不會被記憶在控制卡中，因此每當電源投入時必要再設定。
輸入	short BoardNo: 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31 long Speed: 驅動速度 pulse / sec
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	Nca4n_SetDriveSpeed(0, 0, 100000);

函數	short Nca4n_SetUpRate(short BoardNo, short UnitNo, long Rate)
說明	設定指定軸的加速度 加速度並不會被記憶在控制卡中，因此每當電源投入時必要再設定。
輸入	short BoardNo: 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31 long Rate : 加速度 pps / msec
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	Nca4n_SetUpRate(0, 0, 1000);

函數	short Nca4n_SetDownRate(short BoardNo, short UnitNo, long Rate)
說明	設定指定軸的減速度 減速度並不會被記憶在控制卡中，因此每當電源投入時必要再設定。
輸入	short BoardNo: 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31 long Rate : 減速度 pps / msec
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	Nca4n_SetDownRate (0, 0, 1000);

函數	short Nca4n_SetOriginSpeed1(short BoardNo, short UnitNo, long Speed)
說明	設定指定軸的搜索原點感應器速度 搜尋速度並不會被記憶在控制卡中，因此每當電源投入時必要再設定。
輸入	short BoardNo: 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31 long Speed: 搜尋速度 pulse / sec
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	Nca4n_SetOriginSpeed1(0, 0, 10000);

函數	short Nca4n_SetOriginSpeed2(short BoardNo, short UnitNo, long Speed)
說明	設定指定軸的搜尋到原點感應器之，再搜索 Z 相的速度 搜尋速度並不會被記憶在控制卡中，因此每當電源投入時必要再設定。
輸入	short BoardNo: 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31 long Speed: 搜索速度 pulse / sec
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	Nca4n_SetOriginSpeed2(0, 0, 1000);

函數	short Nca4n_AbsMove(short BoardNo, short UnitNo, long Position)
說明	指定軸進行絕對值移動
輸入	short BoardNo : 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31 long Position: 目標位置 pulse
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -6(AXIS _ BUSY) : 該軸動作中 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	Nca4n_AbsMove(0, 0, 10000);

函數	short Nca4n_RelMove(short BoardNo, short UnitNo, long Position)
說明	指定軸進行相對值移動
輸入	short BoardNo : 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31 long Position: 目標位置 pulse
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -6(AXIS _ BUSY) : 該軸動作中 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	Nca4n_RelMove(0, 0, 10000);

函數	short Nca4n_TriggerAbsMove(short BoardNo, short UnitNo, long Position, long IntervalPls, short OutTime, short OutNo)
說明	讓指定軸以絕對值移動，但在每指定脈波數到達後指定輸出點輸出一觸發信號
輸入	short BoardNo : 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31 long Position: 目標位置 pulse long IntervalPls: 間隔 pulse (1~) short OutTime: 輸出 ON 時間 msec (1~) short OutNo: 輸出點號碼 1 2
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -6(Axis _ BUSY) : 軸動作中 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	Nca4n_TriggerAbsMove(0, 0, 10000, 100, 5, 1); 從現在位置開始到位置 10000 脈波的位置，每 100 脈波第 1 點輸出 ON 並持續 5msec。

函數	short Nca4n_TriggerRelMove(short BoardNo, short UnitNo, long Position, long IntervalPls, short OutTime, short OutNo)
說明	讓指定軸以相對值移動，但在每指定脈波數到達後指定輸出點輸出一觸發信號
輸入	short BoardNo : 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31 long Position: 目標位置 pulse long IntervalPls: 間隔 pulse (1~) short OutTime: 輸出 ON 時間 msec (1~) short OutNo: 輸出號碼 1 2
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -6(Axis _ BUSY) : 該軸動作中 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	Nca4n_TriggerRelMove(0, 0, 10000, 100, 5, 1);

函數	short Nca4n_SetLineMoveAxis(short BoardNo, short UnitNo, long Position, short PosType)
說明	指定要執行直線補間的軸。這命令只是用來指定進行直線補間的移動軸，在 StartLineMove 被執行之前軸並不會移動。 另外在 StartLineMove 以及 ResetLineMoveAxis 被執行之前此指定一直有效。
輸入	short BoardNo : 控制卡號碼 0-15

	short UnitNo : 模組號碼 0-31 long Position: 目標位置 pulse short PosType: 絕對 / 相對 0(ABS _ PLS):絕對設定值移動, 1 (RELPLS):相對值移動
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -6(AXIS _ BUSY) : 軸動作中 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器

函數	short Nca4n_ResetLineMoveAxis(short BoardNo)
說明	用來解除 SetLineMoveAxis 所指定的直線補間軸。
輸入	short BoardNo : 控制卡號碼 0-15
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓

函數	short Nca4n_StartLineMove(short BoardNo)
說明	開始執行被 SetLineMoveAxis 指定的軸直線補間。 此命令實行時, 被指定的直線補間軸會被解除。
輸入	short BoardNo : 控制卡號碼 0-15
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -7(LINE _ AXIS _ ERR) : 直線補間軸沒有被指定
使用例	Nca4n_SetLineMoveAxis (0, 0, 10000, ABS _ PLS); 指定 1 號軸 (絕對值移動) Nca4n_SetLineMoveAxis (0, 2, 20000, ABS _ PLS); 指定 3 號軸 (絕對值移動) Nca4n_ResetLineMoveAxis (0); 取消 1、3 號軸的指定 Nca4n_SetLineMoveAxis (0, 3, -5000, REL _ PLS); 指定 4 號軸 (相對值移動) Nca4n_SetLineMoveAxis (0, 5, 10000, ABS _ PLS); 指定 6 號軸 (絕對值移動) Nca4n_StartLineMove (0); 4、6 號軸執行直線補間移動

直線補間能最大同時 32 軸進行。

也可執行多個直線補間組的移動, 例如, 1、3、4 軸一組執行直線補間, 而 6、7 軸一組執行直線補間。在此狀況下, 請依照順序指定 1,3,4 軸後, 執行直線補間移動, 再指定 6,7 軸後, 執行直線補間移動。

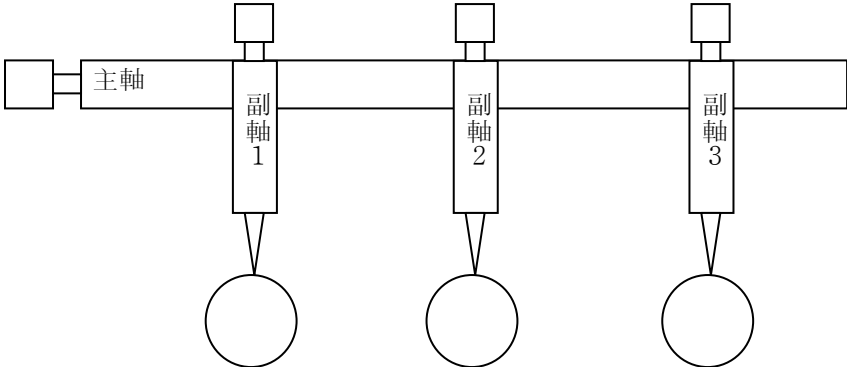
函數	short Nca4n_CirMove(short BoardNo, short UnitNo1, long PassPos1A, long PassPos1B short UnitNo2, long PassPos2A, long PassPos2B)
說明	圓補間移動, 同時最多可執行 16 組的圓補間 移動方式以現在位置, 第 1 經過點, 以及第 2 經過點的順序自動決定的。
輸入	short BoardNo : 控制卡號碼 0-15 short UnitNo1 : 圓補間第 1 軸 0-31 long PassPos1A : 第 1 軸經過點 1 pulse long PassPos1B : 第 1 軸經過點 2 pulse short UnitNo2 : 圓補間第 2 軸 0-31 long PassPos2A : 第 2 軸經過點 1 pulse long PassPos2B : 第 2 軸經過點 2 pulse
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -6(AXIS _ BUSY) : 軸動作中 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	Nca4n_CirMove(0, 0, 500, -500, 1, 500, 500); 執行 1、2 號圓補間

函數	<code>short Nca4n_ArcMove(short BoardNo, short UnitNo1, long PassPos1, long EndPos1 short UnitNo2, long PassPos2, long EndPos2)</code>
說明	弧補間移動，同時最多可執行 16 組的弧補間 移動方式以現在位置，第 1 經過點，以及結束點的順序自動決定的。
輸入	short BoardNo : 控制卡號碼 0-15 short UnitNo1 : 弧補間第 1 軸 0 - 3 1 long PassPos1 : 第 1 軸經過點 pulse long EndPos1 : 第 1 軸結束點 pulse short UnitNo2 : 弧補間第 2 軸 0 - 3 1 long PassPos2 : 第 2 軸經過點 pulse long EndPos2 : 第 2 軸結束點 pulse
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -6(Axis _ BUSY) : 軸動作中 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	<code>Nca4n_ArcMove(0, 0, 500, -500, 1, 500, 500);</code> 執行 1、2 號弧補間

函數	short Nca4n_SetFirstCirFollowAxis(short BoardNo, short UnitNo1, long PassPos1A, long PassPos1B short UnitNo2, long PassPos2A, long PassPos2B)
說明	爲了在一主軸和多副軸的組合約時進行圓補間移動的 登記最初的第 1 組的主軸和副軸。
輸入	short BoardNo : 控制卡號碼 0-15 short UnitNo1 : 圓補間軸 1 0-31 long PassPos1A : 透過點 1 pulse long PassPos1B : 透過點 2 pulse short UnitNo2 : 圓補間軸 2 0-31 long PassPos2A : 透過點 1 pulse long PassPos2B : 透過點 2 pulse
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -6(Axis _ BUSY) : 軸動作中 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器

函數	short Nca4n_SetFirstArcFollowAxis(short BoardNo, short UnitNo1, long PassPos1, long EndPos1 short UnitNo2, long PassPos2, long EndPos2)
說明	當一個主軸和多個副軸的組合時進行圓補間移動時，用來設定最初的第 1 組的主軸和副軸。
輸入	short BoardNo : 控制卡號碼 0-15 short UnitNo1 : 圓弧補間軸 1 0 - 31 long PassPos1: 透過點 pulse long EndPos1 : 結束點 pulse short UnitNo2 : 圓弧補間軸 2 0 - 31 long PassPos2: 透過點 pulse long EndPos2 : 結束點 pulse
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -6(Axis _ BUSY) : 軸動作中 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器

函數	short Nca4n_SetNextFollowAxis(short BoardNo, short UnitNo)
說明	設定第二個以後的副軸。 圓補間和弧補間都共用此函數。 第二副軸之後的所有跟隨軸通過點等數值都是根據第一副軸的數值基礎之上自動計算。 儘管副軸的現在位置不同，全部的副軸都是從現在位置開始來描繪圓弧。
輸入	short BoardNo : 控制卡號碼 0-15 short UnitNo : 圓弧補間跟隨軸 0-31
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -6(AXIS _ BUSY) : 軸動作中 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器 -11(FOLLOW _ ERR) : 最初的第 1 組沒有被登記

函數	short Nca4n_StartFollow(short BoardNo)
說明	SetFirstCirFollowAxis 以及 SetFirstArcFollowAxis 和 SetNextFollowAxis 設定之後，開始以設定數據為基礎同時執行圓或弧補間移動
輸入	short BoardNo : 控制卡號碼 0-15
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -11(FOLLOW _ ERR) : 沒有設定最初的第 1 組
使用例	Nca4n_SetFirstCirFollowAxis(0, 0, 500, -500, 1, 500, 500); 設定第 1 組 Nca4n_SetNextFollowAxis(0, 2); 設定第 2 組 Nca4n_SetNextFollowAxis(0, 3); 設定第 3 組 Nca4n_StartFollow(0); 同時圓補間移動開始
 The diagram illustrates a mechanical system where a single horizontal main axis (主軸) is coupled to three separate slave axes (副軸 1, 副軸 2, 副軸 3). Each slave axis consists of a vertical rectangular component mounted on the main axis, with a circular disk or wheel attached below it via a conical pivot point.	

函數	short Nca4n_SlowStop(short BoardNo, short UnitNo)
說明	讓指定軸減速停止
輸入	short BoardNo : 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	Nca4n_SlowStop(0, 0);

函數	short Nca4n_InstantlyStop(short BoardNo, short UnitNo)
說明	讓指定軸立即停止
輸入	short BoardNo : 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	Nca4n_InstantlyStop(0, 0);

函數	short Nca4n_GetDriveEnd(short BoardNo, short UnitNo)
說明	讀回指定軸的是否移動結束
輸入	short BoardNo : 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31
傳回值	0(NO _ ERR) : 移動結束 -6(AXIS _ BUSY) : 移動中 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	If (Nca4n_GetDriveStat(0, 0) == 0) 1 號軸移動結束？

函數	short Nca4n_SetPosition(short BoardNo, short UnitNo, long Position)
說明	變更實際位置
輸入	short BoardNo : 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31 long Position: 新的實際位置 pulse
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器 -6(Axis _ BUSY) : 軸動作中
使用例	Nca4n_SetPosition(0, 0, 0); 重置(reset)實際位置

函數	short Nca4n_Homing(short BoardNo, short UnitNo, short HomeMode)
說明	進行指定軸的原點覆歸 1:HomeLS+Z 相，先朝 CCW 方向搜索 HomeLS，直到該信號 ON 後，再以 CW 方向搜索 Z 相 2:CCW LS+Z 相，先朝 CCW 方向搜索 CCW LS，直到該信號 ON 後，再以 CW 方向搜索 Z 相 3:CW LS+Z 相，先朝 CW 方向搜索 CW LS，直到該信號 ON 後，再以 CCW 方向搜索 Z 相
輸入	short BoardNo : 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31 short HomeMode: 原點覆歸型式 1 : HomeLS+ Z 相 , 2 : CCW LS+ Z 相 , 3 : CW LS+ Z 相
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器 -5(PARAMETER _ ERR) : 參數的設定值是範圍以外 -6(Axis _ BUSY) : 軸動作中
使用例	Nca4n_Homing(0, 0, 1);

函數	short Nca4n_GetHomingStatus(short BoardNo, short UnitNo)
說明	進行原點覆歸的結束確認
輸入	short BoardNo : 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31
傳回值	0(HOMING _ NOW) : 原點覆歸中 1(HOMING _ FINISH) : 原點覆歸正常結束 -8(HOMING _ ERR) : 原點覆歸異常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	Nca4n_Homing (0, 0, 1) ; 軸 1 原點覆歸 Nca4n_Homing (0, 1, 1) ; 軸 2 原點覆歸 Nca4n_Homing (0, 2, 1) ; 軸 3 原點覆歸 hile(Nca4n_GetHomingStatus(0, 0) != HOMING_FINISH);

函數	short Nca4n_SetSoftLimit(short BoardNo, short UnitNo, long PsoftLmt, long NsoftLmt)
說明	設定軟體極限
輸入	short BoardNo : 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31 long PsoftLmt: +旁邊軟體極限 pulse long NsoftLmt: -旁邊軟體極限 pulse
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	Nca4n_SetSoftLimit(0, 0, 10000, -10000);

函數	short Nca4n_SetSoftLimitUse(short BoardNo, short UnitNo, short Mode)
說明	設定使用/不使用軟體極限
輸入	short BoardNo : 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31 short Mode : 1(SET _ ON) : 軟體極限有效; 0(SET _ OFF) : 軟體極限無效
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	Nca4n_SetSoftLimit(0, 0, SET _ ON); 軸 1 軟體極限有效

函數	short Nca4n_SetHardLimitUse(short BoardNo, short UnitNo, short P _ LS, short N _ LS)
說明	設定使用/不使用硬體極限
輸入	short BoardNo : 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31 short P _ LS : 1(SET _ ON) : +硬體極限有效; 0(SET _ OFF) : +硬體極限無效 short N _ LS : 1(SET _ ON) : -硬體極限有效; 0(SET _ OFF) : -硬體極限無效
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤

	-10(MODULE _ ERR) : 被指定的模組號碼不是伺服驅動器
使用例	Nca4n_SetHardLimit(0, 0, SET _ ON, SET _ OFF) ; 軸 1 +LS 有效-LS 無效

函數	short Nca4n_Input(short BoardNo, short UnitNo, long* Data)
說明	取得輸入模組 (MCN-9321)的輸入資料
輸入	short BoardNo : 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31 long *Data: 對用於儲藏輸入數據的緩衝指標 1=輸入 ON ; 0=輸入 OFF
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -10(MODULE _ ERR) : 被指定的單位號碼不是輸入模組
使用例	Nca4n_Input(0, 0,&InpBuf);

函數	short Nca4n_Output(short BoardNo, short UnitNo, long Data)
說明	寫進到輸出模組 (MCN-9322)的輸出資料
輸入	short BoardNo : 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31 long Data: 輸出數據 1=輸入 ON ; 0=輸入 OFF
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -10(MODULE _ ERR) : 被指定的單位號碼不是輸出模組
使用例	OutBuf l= 5; Nca4n_Output(0, 0, OutBuf); Bit0,2 輸出 ON

函數	short Nca4n_BitOutput(short BoardNo, short UnitNo, short Bit, short Data)
說明	設定輸出模組 (MCN-9322) 的位元是 ON / OFF
輸入	short BoardNo : 控制卡號碼 0-15 short UnitNo : 模組號碼 0-31 short Bit : ON / 做 OFF 的位號碼 0-31 short Data: 1(SET _ ON) : 輸出 ON、0(SET _ OFF) : 輸出 OFF
傳回值	0(NO _ ERR) : 正常結束 -1(CARD _ NO _ ERR) : 控制卡號碼指定錯誤 -2(CARD _ OPEN _ ERR) : 被指定的控制卡沒有開啓 -3(UNIT _ NO _ ERR) : 模組號碼指定錯誤 -10(MODULE _ ERR) : 被指定的單位號碼不是輸出模組
使用例	Nca4n_BitOutput(0, 0, 3, SET _ ON) ; Bit3 的輸出 ON

6 範例程式

本程式是利用 VC++ 6.0 所寫成的，使用其他程式語言開法亦可參考此架構修改。

```

/***** Board Open *****/
if ( Nca4n_Open( 0 ) == FALSE ) {                                     // Board ID=0 Open
MessageBox( "NCA4N Board Open Error !", "Error", MB_OK );
EndDialog( 0 );
}

/***** Set Speed *****/
Nca4n_SetStartSpeed( 0, 0, 1000 );                                   // Set Axis1 Start Speed (pulse/sec)
Nca4n_SetUpRate( 0, 0, 100 );                                       // Set Axis1 UpRate (pps/msec)
Nca4n_SetDownRate( 0, 0, 100 );                                     // Set Axis1 DownRate (pps/msec)
Nca4n_SetStartSpeed( 0, 1, 1000 );                                   // Set Axis2 Start Speed (pulse/sec)
Nca4n_SetUpRate( 0, 1, 100 );                                       // Set Axis2 UpRate (pps/msec)
Nca4n_SetDownRate( 0, 1, 100 );                                     // Set Axis2 DownRate (pps/msec)
Nca4n_SetStartSpeed( 0, 2, 1000 );                                   // Set Axis3 Start Speed (pulse/sec)
Nca4n_SetUpRate( 0, 2, 100 );                                       // Set Axis3 UpRate (pps/msec)
Nca4n_SetDownRate( 0, 2, 100 );                                     // Set Axis3 DownRate (pps/msec)

/***** Servo ON *****/
Nca4n_ServoOn( 0, 0 );                                              // Axis1 Servo ON
Nca4n_ServoOn( 0, 1 );                                              // Axis2 Servo ON
Nca4n_ServoOn( 0, 2 );                                              // Axis3 Servo ON

/***** Homing *****/
Nca4n_SetOriginSpeed1( 0, 0, 30000 );                               // Set Axis1 Home Sensor Search Speed
Nca4n_SetOriginSpeed2( 0, 0, 1000 );                               // Set Axis1 Z Phase Search Speed
Nca4n_SetOriginSpeed1( 0, 1, 30000 );                               // Set Axis2 Home Sensor Search Speed
Nca4n_SetOriginSpeed2( 0, 1, 1000 );                               // Set Axis2 Z Phase Search Speed
Nca4n_SetOriginSpeed1( 0, 2, 30000 );                               // Set Axis3 Home Sensor Search Speed
Nca4n_SetOriginSpeed2( 0, 2, 1000 );                               // Set Axis3 Z Phase Search Speed
Nca4n_Homing( 0, 2, 1 );                                           // Axis3(Z) Homing
while ( Nca4n_GetHomingStatus( 0, 2 ) != HOMING_FINISH );        // Wait Axis3 Homing Finish
Nca4n_Homing( 0, 0, 1 );                                           // Axis1(X) Homing
Nca4n_Homing( 0, 1, 1 );                                           // Axis2(Y) Homing
while ( ( Nca4n_GetHomingStatus( 0, 0 ) != HOMING_FINISH )        // Wait Axis1 Homing Finish
|| ( Nca4n_GetHomingStatus( 0, 1 ) != HOMING_FINISH ) );          // Wait Axis2 Homing Finish

/***** Absolute Move *****/
Nca4n_SetDriveSpeed( 0, 0, 100000 );                               // Set Axis1 Moving Speed
Nca4n_SetDriveSpeed( 0, 1, 100000 );                               // Set Axis2 Moving Speed
Nca4n_SetDriveSpeed( 0, 2, 50000 );                                // Set Axis3 Moving Speed
Nca4n_AbsMove( 0, 0, 10000 );                                       // Axis1 Move to 10000pulse position
Nca4n_AbsMove( 0, 0, 20000 );                                       // Axis2 Move to 20000pulse position
Nca4n_AbsMove( 0, 0, -5000 );                                       // Axis3 Move to -5000pulse position
while ( ( Nca4n_GetDriveEnd( 0, 0 ) == AXIS_BUSY )                // Wait Axis1 Move Finish
|| ( Nca4n_GetDriveEnd( 0, 1 ) == AXIS_BUSY )                      // Wait Axis2 Move Finish
|| ( Nca4n_GetDriveEnd( 0, 2 ) == AXIS_BUSY ) );                  // Wait Axis3 Move Finish

/***** Line Move *****/
Nca4n_SetDriveSpeed( 0, 0, 100000 );                               // Set Axis1 Moving Speed
Nca4n_SetDriveSpeed( 0, 1, 100000 );                               // Set Axis2 Moving Speed
Nca4n_SetLineMoveAxis( 0, 0, 50000, ABS_PLS );                     // Axis1 Registration
Nca4n_SetLineMoveAxis( 0, 1, 70000, ABS_PLS );                     // Axis2 Registration

```

```

Nca4n_StartLineMove( 0 );
while ( ( Nca4n_GetDriveEnd( 0, 0 ) == AXIS_BUSY )
  || ( Nca4n_GetDriveEnd( 0, 1 ) == AXIS_BUSY ) );

/***** Circle Move *****/
Nca4n_SetDriveSpeed( 0, 0, 50000 );
Nca4n_SetDriveSpeed( 0, 1, 50000 );
Nca4n_CirMove( 0, 0, 5000, 0, 1, 5000, 10000 );
while ( ( Nca4n_GetDriveEnd( 0, 0 ) == AXIS_BUSY )
  || ( Nca4n_GetDriveEnd( 0, 1 ) == AXIS_BUSY ) );

/***** Servo OFF *****/
Nca4n_ServoOff( 0, 0 );
Nca4n_ServoOff( 0, 1 );
Nca4n_ServoOff( 0, 2 );

/***** Board Close *****/
Nca4n_Close( 0 );

```

```

// Line Move Start
// Wait Axis1 Move Finish
// Wait Axis2 Move Finish

// Set Axis1 Moving Speed
// Set Axis2 Moving Speed
// Axis1,2 Circle Move Start
// Wait Axis1 Move Finish
// Wait Axis2 Move Finish

// Axis1 Servo OFF
// Axis2 Servo OFF
// Axis3 Servo OFF

```

經銷商：

製造商：

Aurotek[®]

和椿科技股份有限公司

台北市內湖區洲子街60號1樓

TEL: +886-2-8752-3311 FAX: +886-2-8752-3347

<http://www.robot.com.tw>