Smarter. Greener. Together.

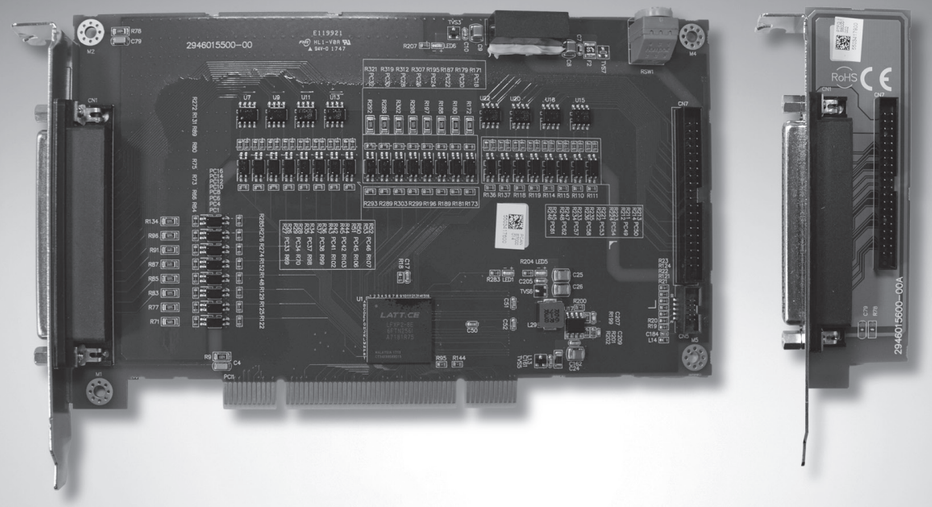
台達電子工業股份有限公司

機電事業群

33068 桃園市桃園區興隆路 18 號

TEL: 886-3-3626301

FAX: 886-3-3716301



台達 PCI-D122 程式開發手冊

www.deltaww.com

Smarter. Greener. Together.

序言

感謝您使用本產品，本使用手冊提供 PCI-D122-XND0 數位輸入/輸出控制卡(I/O 卡)產品程式開發的相關資訊。

本手冊內容包含

- API 回傳值與訊息描述
- 硬體初始化 Initial API 說明
- 數位輸入輸出控制 API 說明
- 中斷功能 Interrupt API 說明

I/O 卡系列產品特色

PCI-D 系列提供全系列的工業級資料擷取(DAQ)產品。並透過自家開發的 PAC 平台與軟體支援，達成整體的解決方案，滿足客戶對於 PCI 介面卡高速、高品質且節省成本並具有輸出保護功能的產品需求。

如何使用本操作手冊

您可視本手冊為學習使用I/O卡系列產品的參考資訊，手冊將告訴您如何使用I/O卡及撰寫簡單範例。在開始寫程式前，請先閱讀本手冊，了解函式庫如何使用與I/O卡的相關說明。

台達電子技術服務

如果您在使用上仍有問題，歡迎洽詢經銷商或本公司客服中心。

(此頁有意留為空白)

目錄

1 命令傳回值與訊息描述

1.1 錯誤傳回值一覽表 ····· 1-2

2 硬體初始化 Initial API

2.1 _PCI_DIO_open_card ····· 2-3
2.2 _PCI_DIO_close ····· 2-4
2.3 _PCI_DIO_get_card_no ····· 2-5
2.4 _PCI_DIO_get_fpga_version ····· 2-6
2.5 _PCI_DIO_get_hardware_id ····· 2-7
2.6 _PCI_DIO_get_DLL_path ····· 2-8
2.7 _PCI_DIO_get_DLL_version ····· 2-9

3 數位輸入輸出控制 API

3.1 _PCI_DIO_set_single_output ····· 3-3
3.2 _PCI_DIO_get_single_output ····· 3-4
3.3 _PCI_DIO_get_single_input ····· 3-5
3.4 _PCI_DIO_set_output ····· 3-6
3.5 _PCI_DIO_get_output ····· 3-7
3.6 _PCI_DIO_get_input ····· 3-8
3.7 _PCI_DIO_set_output_error_handle ····· 3-9
3.8 _PCI_DIO_get_output_error_handle ····· 3-10

4 中斷 Interrupt API

4.1 _PCI_DIO_int_control ····· 4-3
4.2 _PCI_DIO_set_int_factor ····· 4-4
4.3 _PCI_DIO_get_int_factor ····· 4-5
4.4 _PCI_DIO_link_interrupt ····· 4-6
4.5 _PCI_DIO_get_int_status ····· 4-7
4.6 _PCI_DIO_get_int_io_control ····· 4-8
4.7 _PCI_DIO_get_input_trigger_count ····· 4-9
4.8 _PCI_DIO_reset_input_trigger_count ····· 4-10

(此頁有意留為空白)

1 命令傳回值與訊息描述

本章介紹 I/O 卡 API 函式命令回傳代碼、代碼(錯誤碼)標示名稱與錯誤問題的描述，使用者可利用錯誤代碼的描述進行問題排解。

1.1 錯誤傳回值一覽表 ····· 1-2

1.1 錯誤傳回值一覽表

1

當您在使用 I/O 卡的 API 函式庫時，API 函式會回傳一個錯誤代碼，如表 1.1.1。若該 API 函式傳回的代碼為 0 時，這表示此 API 函式已成功地被執行；反之，若此 API 函式回傳其它的錯誤代碼，這表示可能在操作程序或者是硬體連結方面產生問題。請依該錯誤代碼的描述進行問題排解。

表 1.1.1 PCI-D122-XND0 錯誤代碼表

錯誤傳回代碼 (十進制)	錯誤碼標示名稱	錯誤問題描述
0	ERR_NO_ERROR	API 成功運作
1	ERR_INIT_FAILED	硬體初始化失敗
2	ERR_NO_DEVICE_FOUND	尋找裝置錯誤，請確認輸入卡號
3	ERR_DEVICE_NOT_INITIALIZED	裝置尚未初始化，請確認開卡初始化程序
4	ERR_PORT_IO_READ	PCI 驅動程式讀取發生錯誤
5	ERR_PORT_IO_WRITE	PCI 驅動程式寫入發生錯誤
6	ERR_MEMORY_READ	I/O 卡記憶體讀取錯誤
7	ERR_MEMORY_WRITE	I/O 卡記憶體寫入錯誤
8 ~ 13	-	保留
14	ERR_Parameter	API 參數輸入錯誤
15	ERR_GetDLLPath	DLL 取得路徑錯誤

硬體初始化 Initial API

2

本章節說明如何使用應用程式介面(API)初始化 I/O 卡硬體裝置，並在不使用時釋放硬體裝置資源。

2.1 _PCI_DIO_open_card ····· 2-3
2.2 _PCI_DIO_close ····· 2-4
2.3 _PCI_DIO_get_card_no ····· 2-5
2.4 _PCI_DIO_get_fpga_version ····· 2-6
2.5 _PCI_DIO_get_hardware_id ····· 2-7
2.6 _PCI_DIO_get_DLL_path ····· 2-8
2.7 _PCI_DIO_get_DLL_version ····· 2-9

功能名稱	描述
_PCI_DIO_open_card	取得 I/O 卡裝置數量。
_PCI_DIO_close	關閉 I/O 卡，並且釋放硬體資源。
_PCI_DIO_get_card_no	取得 I/O 卡裝置卡號。
_PCI_DIO_get_fpga_version	取得 I/O 卡裝置 FPGA 版號。
_PCI_DIO_get_hardware_id	取得 I/O 卡版號。
_PCI_DIO_get_DLL_path	取得 PCI_DIO.dll 詳細路徑資料。
_PCI_DIO_get_DLL_version	取得 PCI_DIO.dll 版本資料。

2.1 _PCI_DIO_open_card

■ **格式**

I16 PASCAL _PCI_DIO_open_card(U8* nDeviceCount)

■ **目的**

初始化 I/O 卡並取得裝置數量。

■ **參數**

名稱	資料型別	單位	描述
nDeviceCount	U8*	數值單位	回傳控制系統數量

■ **範例**

I16 rt=0; //回傳錯誤代碼

U8 nDeviceCount=0;

rt = _PCI_DIO_open_card(&nDeviceCount);

2.2 _PCI_DIO_close

2

■ **格式**

void PASCAL _PCI_DIO_close(void)

■ **目的**

關閉 I/O 卡，並且釋放硬體資源。

■ **參數**

名稱	資料型別	單位	描述
N/A	N/A	N/A	N/A

■ **範例**

_PCI_DIO_close(void);

2.3 _PCI_DIO_get_card_no

■ **格式**

I16 PASCAL _PCI_DIO_get_card_no(U8 nDeviceNo, U16* nCardNo)

■ **目的**

取得 I/O 卡裝置卡號。

■ **參數**

名稱	資料型別	單位	描述
nDeviceNo	U8	編號單位	裝置序列
nCardNo	U16*	編號單位	取得 I/O 卡裝置卡號

■ **範例**

```
I16 rt;
U8 nDeviceNo;
U16 nCardNo;

rt = _PCI_DIO_get_card_no (nDeviceNo, &nCardNo);
```

2.4 _PCI_DIO_get_fpga_version

2

■ **格式**

I16 PASCAL _PCI_DIO_get_fpga_version(U8 nCardNo, U16* nVersion)

■ **目的**

取得 I/O 卡裝置 FPGA 版號。

■ **參數**

名稱	資料型別	單位	描述
nCardNo	U8	編號單位	I/O 卡裝置卡號
nVersion	U16*	編號單位	FPGA 版號

■ **範例**

```
I16 rt;
U8 nCardNo=0;
U16 nVersion;

rt = _PCI_DIO_get_fpga_version(nCardNo, &nVersion);
```

2.5 _PCI_DIO_get_hardware_id

■ **格式**

I16 PASCAL _PCI_DIO_get_hardware_id(U8 nCardNo, U16* nDeviceID)

■ **目的**

取得 I/O 卡版號。

■ **參數**

名稱	資料型別	單位	描述
nCardNo	U8	編號單位	I/O 卡裝置卡號
nDeviceID	U16*	編號單位	I/O 卡版號 ID

■ **範例**

```
I16 rt;
U8 nCardNo=0;
U16 nDeviceID;

rt = _PCI_DIO_get_hardware_id(nCardNo, &nDeviceID);
```

2.6 _PCI_DIO_get_DLL_path

2

■ **格式**

I16 PASCAL _PCI_DIO_get_DLL_path(I8* lpFilePath, U32 nSize, U32* nLength)

■ **目的**

取得 PCI_DIO.dll 詳細路徑資料。

■ **參數**

名稱	資料型別	單位	描述
lpFilePath	Char*	字元陣列單位	PCI_DIO.dll 詳細路徑資料字元
nSize	U32	數值單位	路徑資料擷取長度
nLength	U32*	數值單位	傳回路徑資料實際長度

■ **範例**

```
I16 rt;
Char lpFilePath =[50];
U32 nSize=50;
U32 nLength=0;

I16 status = _PCI_DIO_get_DLL_path(&lpFilePath, nSize, &nLength);
```

2.7 _PCI_DIO_get_DLL_version

■ **格式**

I16 PASCAL _PCI_DIO_get_DLL_version(I8 *lpBuf, U32 nSize, U32* nLength)

■ **目的**

取得 PCI_DIO.dll 版本資料。

■ **參數**

名稱	資料型別	單位	描述
lpBuf	Char*	字元陣列單位	PCI_DIO.dll 版本資料字元
nSize	U32	數值單位	版本資料擷取長度
nLength	U32*	數值單位	傳回版本資料實際長度

■ **範例**

```
I16 rt;
Char lpBuf =[50];
U32 nSize=50;
U32 nLength=0;

rt = _PCI_DIO_get_DLL_version(&lpBuf, nSize, &nLength);
```

(此頁有意留為空白)

2

數位輸入輸出控制 API

3

本章主要介紹數位輸入輸出控制(API)，可控制 I/O 卡 32 Input、32 Output 的腳位狀態與輸出狀態保持功能。

3.1 _PCI_DIO_set_single_output ····· 3-3
3.2 _PCI_DIO_get_single_output ····· 3-4
3.3 _PCI_DIO_get_single_input ····· 3-5
3.4 _PCI_DIO_set_output ····· 3-6
3.5 _PCI_DIO_get_output ····· 3-7
3.6 _PCI_DIO_get_input ····· 3-8
3.7 _PCI_DIO_set_output_error_handle ····· 3-9
3.8 _PCI_DIO_get_output_error_handle ····· 3-10

3

功能名稱	描述
_PCI_DIO_set_single_output	設置 I/O 卡輸出單一 bit 腳位狀態。
_PCI_DIO_get_single_output	取得 I/O 卡輸出單一 bit 腳位狀態。
_PCI_DIO_get_single_input	取得 I/O 卡輸入單一 bit 腳位狀態。
_PCI_DIO_set_output	設置 I/O 卡輸出腳位狀態。
_PCI_DIO_get_output	取得 I/O 卡輸出腳位狀態。
_PCI_DIO_get_input	取得 I/O 卡輸入腳位狀態。
_PCI_DIO_set_output_error_handle	設置 DIO 保持輸出狀態。
_PCI_DIO_get_output_error_handle	取得 DIO 保持輸出狀態的設定。

註：本章節 API 功能需在進行第二章初始化流程完成後才可正常使用。

3.1 _PCI_DIO_set_single_output

■ **格式**

I16 PASCAL _PCI_DIO_set_single_output(U8 nCardNo, U16 nbit, U16 onoff)

■ **目的**

設置 I/O 卡輸出單一 bit 腳位狀態。

■ **參數**

名稱	資料型別	單位	描述
nCardNo	U8	編號單位	I/O 卡裝置卡號
nbit	U16	編號單位	操作的 Bit (0 ~ 31) 如欲操作 Bit 0，bitno = 0 如欲操作 Bit 1，bitno = 1 如欲操作 Bit 2，bitno = 2 以此類推。
onoff	U16	數值單位	0：關閉 1：開啟

■ **範例**

```
I16 rt;
U8 nCardNo=0;
U16 nbit=0;//第一點輸出 ON
U16 onoff=1;

rt = _PCI_DIO_set_single_output(nCardNo, nbit, onoff);
```

註：I/O 卡輸出輸入響應速度

DIO 裝置	響應速度
Output	1 kHz
Input	1 kHz

3.2 _PCI_DIO_get_single_output

3

■ 格式

I16 PASCAL _PCI_DIO_get_single_output(U8 nCardNo, U16 nbit, U16* onoff)

■ 目的

取得 I/O 卡輸出單一 bit 腳位狀態。

■ 參數

名稱	資料型別	單位	描述
nCardNo	U8	編號單位	I/O 卡裝置卡號
nbit	U16	編號單位	操作的Bit (0 ~ 31) 如欲操作Bit 0，bitno = 0 如欲操作Bit 1，bitno = 1 如欲操作Bit 2，bitno = 2 以此類推。
onoff	U16*	數值單位	0：關閉 1：開啟

■ 範例

```
I16 rt;
U8 nCardNo=0;
U16 nbit=0;//第一點輸出 ON
U16 onoff;

rt = _PCI_DIO_set_single_output(nCardNo, nbit, &onoff);
```

3.3 _PCI_DIO_get_single_input

■ **格式**

I16 PASCAL _PCI_DIO_get_single_input(U8 nCardNo, U16 nbit, U16* onoff)

■ **目的**

取得 I/O 卡輸入單一 bit 腳位狀態。

■ **參數**

名稱	資料型別	單位	描述
nCardNo	U8	編號單位	I/O 卡裝置卡號
nbit	U16	編號單位	操作的Bit (0 ~ 31) 如欲操作Bit 0，bitno = 0 如欲操作Bit 1，bitno = 1 如欲操作Bit 2，bitno = 2 以此類推。
onoff	U16*	數值單位	0：關閉 1：開啟

■ **範例**

```
I16 rt;
U8 nCardNo=0;
U16 nbit=0;//第一點輸入
U16 onoff;

rt = _PCI_DIO_get_single_input(nCardNo, nbit, &onoff);
```

3.4 _PCI_DIO_set_output

3

■ 格式

I16 PASCAL _PCI_DIO_set_output(U8 nCardNo, U16 nport, U8 value)

■ 目的

設置 I/O 卡輸出腳位狀態。

■ 參數

名稱	資料型別	單位	描述
nCardNo	U8	編號單位	I/O 卡裝置卡號
nport	U16	編號單位	Port 0 ~ 3 0：Port 0 1：Port 1 2：Port 2 3：Port 3 Port 以 8 bit 為單位
value	U8	數值單位	輸出資料 (ex: 0xF, =bit0 ~ bit3 為 ON 表示輸出)

■ 範例

```
I16 rt;
U8 nCardNo=0;
U16 nport=0;//bit0 ~ bit7
U8 value=0xFF;//bit0 ~ bit7  為 ON

rt = _PCI_DIO_set_output(nCardNo, nport, value);
```

註：I/O 卡輸出輸入響應速度

DIO 裝置	響應速度
Output	1 kHz
Input	1 kHz

3.5 _PCI_DIO_get_output

■ **格式**

I16 PASCAL _PCI_DIO_get_output(U8 nCardNo, U16 nport, U8* value)

■ **目的**

取得 I/O 卡輸出腳位狀態。

■ **參數**

名稱	資料型別	單位	描述
nCardNo	U8	編號單位	I/O 卡裝置卡號
nport	U16	編號單位	Port 0 ~ 3 0：Port 0 1：Port 1 2：Port 2 3：Port 3 Port 以 8 bit 為單位
value	U8*	數值單位	輸出資料 (ex: 0xF, =bit0 ~ bit3 為 ON 表示輸出)

■ **範例**

```
I16 rt;
U8 nCardNo=0;
U16 nport=0;//bit0 ~ bit7
U8 value;

rt = _PCI_DIO_get_output(nCardNo, nport, &value);
```

3.6 _PCI_DIO_get_input

3

■ 格式

I16 PASCAL _PCI_DIO_get_input(U8 nCardNo, U16 nport, U8* value)

■ 目的

取得 I/O 卡輸入腳位狀態。

■ 參數

名稱	資料型別	單位	描述
nCardNo	U8	編號單位	I/O 卡裝置卡號
nport	U16	編號單位	Port 0 ~ 3 0：Port 0 1：Port 1 2：Port 2 3：Port 3 Port 以 8-bit 為單位
value	U8*	數值單位	輸入狀態 (ex: 0xF, =bit0 ~ bit3 表示輸入狀態為 ON)

■ 範例

```
I16 rt;
U8 nCardNo=0;
U16 nport=0;//bit0 ~ bit7
U8 value;

rt = _PCI_DIO_get_output(nCardNo, nport, &value);
```

3.7 _PCI_DIO_set_output_error_handle

■ **格式**

I16 PASCAL _PCI_DIO_set_output_error_handle(U8 nCardNo, U16 onoff)

■ **目的**

設置 DIO 保持輸出狀態。

■ **參數**

名稱	資料型別	單位	描述
nCardNo	U8	編號單位	I/O 卡裝置卡號
onoff	U16	數值單位	0：關閉 1：開啟

■ **範例**

```
I16 rt;
U8 nCardNo=0;
U16 onoff =1;

rt = _PCI_DIO_set_output_error_handle(nCardNo, onoff);
```

3.8 _PCI_DIO_get_output_error_handle

3

■ 格式

I16 PASCAL _PCI_DIO_get_output_error_handle(U8 nCardNo, U16* onoff)

■ 目的

取得 DIO 保持輸出狀態的設定。

■ 參數

名稱	資料型別	單位	描述
nCardNo	U8	編號單位	I/O 卡裝置卡號
onoff	U16*	數值單位	0：關閉 1：開啟

■ 範例

```
I16 rt;
U8 nCardNo=0;
U16 onoff;

rt = _PCI_DIO_get_output_error_handle(nCardNo, &onoff);
```

中斷 Interrupt API

4

本章主要介紹如何設定 I/O 卡中斷操作(API)，包含 DI 中斷功能啟動與解除、中斷觸發模式、讀取中斷達成次數、讀取中斷觸發的事件值及預設中斷處理程序等功能。
本 I/O 卡提供一組 PCI 中斷功能，可指定四支腳位(DI0、DI1、DI2、DI3)作為中斷觸發來源。

4.1 _PCI_DIO_int_control ····· 4-3
4.2 _PCI_DIO_set_int_factor ····· 4-4
4.3 _PCI_DIO_get_int_factor ····· 4-5
4.4 _PCI_DIO_link_interrupt ····· 4-6
4.5 _PCI_DIO_get_int_status ····· 4-7
4.6 _PCI_DIO_get_int_io_control ····· 4-8
4.7 _PCI_DIO_get_input_trigger_count ····· 4-9
4.8 _PCI_DIO_reset_input_trigger_count ····· 4-10

4

功能名稱	描述
_PCI_DIO_int_control	設置中斷功能。
_PCI_DIO_set_int_factor	設置中斷模式。
_PCI_DIO_get_int_factor	取得中斷模式設置狀態。
_PCI_DIO_link_interrupt	設定處理程序，當中斷觸發達成就進入此程序。
_PCI_DIO_get_int_status	取得中斷觸發狀態。
_PCI_DIO_get_int_io_control	取得中斷功能狀態(啟動/關閉)。
_PCI_DIO_get_input_trigger_count	取得中斷觸發次數。
_PCI_DIO_reset_input_trigger_count	重置中斷觸發次數。

註：本章節 API 功能需在進行第二章初始化流程完成後才可正常使用。

4.1 _PCI_DIO_int_control

■ **格式**

I16 PASCAL _PCI_DIO_int_control(U8 nCardNo, U16 enable)

■ **目的**

設置中斷功能。

■ **參數**

名稱	資料型別	單位	描述
nCardNo	U8	編號單位	I/O 卡裝置卡號
enable	U16	數值單位	0：關閉 1：開啟

■ **範例**

```
I16 rt;
U8 nCardNo=0;

U16 enable =1;//中斷功能啟動


rt = _PCI_DIO_int_control(nCardNo, enable);
```

4.2 _PCI_DIO_set_int_factor

4

■ **格式**

I16 PASCAL _PCI_DIO_set_int_factor(U8 nCardNo, U16 int_factor)

■ **目的**

設置中斷模式。

■ **參數**

名稱	資料型別	單位	描述
nCardNo	U8	編號單位	I/O 卡裝置卡號
int_factor	U16	數值單位	中斷模式 0：正緣觸發 Rising Edge 1：負緣觸發 Falling Edge 2：轉態觸發 Change Edge 輸入資料 DI 腳位以 4 個 bit 為一組對應 (ex: int_factor = 0x1111, 二進位表示法： 0x 0001 0001 0001 0001 DI3 DI2 DI1 DI0 DI0：負緣觸發 DI1：負緣觸發 DI2：負緣觸發 DI3：負緣觸發 若設定 DI0 ~ DI3 為轉態觸發模式 ex: int_factor = 0x2222)

■ **範例**

```
I16 rt;
U8 nCardNo=0;

U16 int_factor =0x1111;//中斷模式為負緣觸發


rt = _PCI_DIO_set_int_factor(nCardNo, int_factor);
```

4.3 _PCI_DIO_get_int_factor

■ **格式**

I16 PASCAL _PCI_DIO_get_int_factor(U8 nCardNo, U16* int_factor)

■ **目的**

取得中斷模式設置狀態。

■ **參數**

名稱	資料型別	單位	描述
nCardNo	U8	編號單位	I/O 卡裝置卡號
int_factor	U16*	數值單位	中斷模式 0：正緣觸發 Rising Edge 1：負緣觸發 Falling Edge 2：轉態觸發 Change Edge 輸入資料 DI 腳位以 4 個 bit 為一組對應 (ex: int_factor = 0x1111, 二進位表示法： 0x 0001 0001 0001 0001 DI3 DI2 DI1 DI0 DI0：負緣觸發 DI1：負緣觸發 DI2：負緣觸發 DI3：負緣觸發 若 ex: int_factor 為 0x2222 DI0 ~ DI3 為轉態觸發模式)

■ **範例**

```
I16 rt;
U8 nCardNo=0;
U16 int_factor;

rt = _PCI_DIO_get_int_factor(nCardNo, &int_factor);
```

4.4 _PCI_DIO_link_interrupt

4

■ **格式**

I16 PASCAL _PCI_DIO_link_interrupt(U8 nCardNo, void (__stdcall *callbackAddr)(U8 nCardNo))

■ **目的**

設定處理程序，當中斷觸發達成就進入此程序。

■ **參數**

名稱	資料型別	單位	描述
nCardNo	U8	編號單位	I/O 卡裝置卡號
callbackAddr	void	處理程序	預設之中斷處理程序

■ **範例**

```
//建立 CallBack Function

void _stdcall CallBack(U8 CardNo)
{

    //寫入處理程序指令

}


//上述 Function 建立後開始設定中斷處理程序 API

I16 rt;
U8 nCardNo=0;

//設定設定處理程序，並將 CallBack Function  名稱引數寫入 API

rt = _PCI_DIO_link_interrupt(nCardNo, CallBack);

//設定完成，在中斷觸發達成時，自動進入 CallBack 函式中執行程序指令
```

4.5 _PCI_DIO_get_int_status

■ **格式**

I16 PASCAL _PCI_DIO_get_int_status(U8 nCardNo, U16 *int_status)

■ **目的**

取得中斷觸發狀態。

■ **參數**

名稱	資料型別	單位	描述
nCardNo	U8	編號單位	I/O 卡裝置卡號
int_status	U16*	數值單位	0: Clear 1: Trigger bit0: DI0 中斷觸發狀態 bit1: DI1 中斷觸發狀態 bit2: DI2 中斷觸發狀態 bit3: DI3 中斷觸發狀態 (ex: int_status = 0x1, DI0 觸發 int_status = 0x2, DI1 觸發 int_status = 0x4, DI2 觸發 int_status = 0x8, DI3 觸發) 目前 int_status 僅定義 bit0 ~ bit3 資料，int_status 狀態在完成 Callback 處理程序後會自動清除為 0

■ **範例**

```
I16 rt;
U8 nCardNo=0;
U16 int_status;

rt = _PCI_DIO_get_int_status(nCardNo, &int_status);
```

4.6 _PCI_DIO_get_int_io_control

4

■ **格式**

I16 PASCAL _PCI_DIO_get_int_io_control(U8 nCardNo, U16 *io_control)

■ **目的**

取得中斷功能狀態(啟動/關閉)。

■ **參數**

名稱	資料型別	單位	描述
nCardNo	U8	編號單位	I/O 卡裝置卡號
io_control	U16*	數值單位	0：關閉 1：開啟 bit0: DI0 中斷功能 ON/OFF bit1: DI1 中斷功能 ON/OFF bit2: DI2 中斷功能 ON/OFF bit3: DI3 中斷功能 ON/OFF (ex: io_control = 0x3, DI0 中斷功能開啟 DI1 中斷功能開啟 DI2 中斷功能關閉 DI3 中斷功能關閉) 目前 io_control 僅定義 bit0 ~ bit3 資料

■ **範例**

```
I16 rt;
U8 nCardNo=0;
U16 io_control;

rt = _PCI_DIO_get_int_io_control(nCardNo, &io_control);
```

4.7 _PCI_DIO_get_input_trigger_count

■ **格式**

I16 PASCAL _PCI_DIO_get_input_trigger_count(U8 nCardNo, U16 nbit, U16 *counter)

■ **目的**

取得中斷觸發次數。

■ **參數**

名稱	資料型別	單位	描述
nCardNo	U8	編號單位	I/O 卡裝置卡號
nbit	U16	編號單位	0: DI0 通道 1: DI1 通道 2: DI2 通道 3: DI3 通道
counter	U16*	數值單位	中斷觸發達成次數

■ **範例**

```
I16 rt;
U8 nCardNo=0;

U16 nbit=0;//DI0 通道

U16 counter;

rt = _PCI_DIO_get_input_trigger_count(nCardNo, nbit, &counter);
```

4.8 _PCI_DIO_reset_input_trigger_count

4

■ **格式**

I16 PASCAL _PCI_DIO_reset_input_trigger_count(U8 nCardNo, U16 nbit)

■ **目的**

重置中斷觸發次數。

■ **參數**

名稱	資料型別	單位	描述
nCardNo	U8	編號單位	I/O 卡裝置卡號
nbit	U16	編號單位	0: DI0 通道 1: DI1 通道 2: DI2 通道 3: DI3 通道

■ **範例**

```
I16 rt;
U8 nCardNo=0;

U16 nbit=0;//DI0 通道


rt = _PCI_DIO_reset_input_trigger_count(nCardNo, nbit);
```