

---

**PCX-4264/AC**  
**Isolated**  
**32 channel D/I**  
**and**  
**32 channel D/O**

Copy Right Notice

The information in this manual is subject to change without prior notice in order to improve reliability, design and function and does not represent a commitment on the part of the manufacturer. No part of this manual may be reproduced, copied, or transmitted in any form without the prior written permission of manufacturer.

Acknowledgment

Products mentioned in this manual are mentioned for identification purpose only. Product names appearing in this manual may or may not be registered trademarks or copyright of their respective companies

Printed Aug. 2002 Rev 1.0

---

---

# Table of Contents

<b>Chapter 1 Introduction</b> .....	<b>2</b>
1.1 Introduction .....	3
1.2 Features.....	3
1.3 Applications.....	3
1.4 Specifications .....	4
1.5 Software Supporting.....	5
1.6 Programming Library.....	5
<b>Chapter 2 Installation</b> .....	<b>6</b>
2.1 What You Have .....	7
2.2 Unpacking .....	7
2.3 Hardware Installation Outline.....	7
2.4 PCB Layout .....	8
2.5 Installation Procedures .....	9
2.6 Device Installation for Windows Systems .....	9
2.7 Connector Pin Assignment of PCX-4264/AC .....	10
2.8 Card number setting.....	11
<b>Chapter 3 Registers Format</b> .....	<b>12</b>
3.1 PCI PnP Registers .....	13
3.2 Digital Input Register Address Map .....	13
3.3 Digital Output Register .....	14
3.4 Reset control registers .....	14
<b>Chapter 4 Operation Theorem</b> .....	<b>16</b>
4.1 Isolated Digital Input Channels .....	17
4.2 Isolated Digital Output Channels .....	18
4.3 Edge Change Detection .....	19
4.4 Digital debounce.....	20
<b>Chapter 5 Libraries</b> .....	<b>22</b>
5.1 Libraries Installation .....	23
5.2 How to use the Functions in PCIDAQ.DLL .....	23
5.3 Summary of function calls .....	24
5.4 Open card .....	25
5.5 Get Card's ID: .....	26
5.6 Get Driver Version.....	27
5.7 Close card.....	28
5.8 Get PCI Bus and Slot number .....	29
5.9 Read digital input data.....	30

---

5.10 Set debounce time of digital inputs .....	31
5.11 Write data to digital output port .....	32
5.12 Read back digital output data.....	33
5.13 Set bit of digital output port .....	34
5.14 Reset bit of digital output port .....	35
5.15 Enable Interrupt .....	36
5.16 Disable Interrupt .....	38
5.17 Read Interrupt Status Register.....	39
5.18 Clear Interrupt Status Register.....	40
<b>Chapter 6 PDB-8068 Terminal board</b> .....	<b>41</b>

---

# Chapter 1

## Introduction

---

### 1.1 Introduction

The PCX-4264/AC is 64-CH high-density isolated digital input and/or output product. This I/O card is isolated up-to 5000 Vdc (excluding cables) for channel-to-computer isolation. It protects your computer against damage caused by accidental contact with high external voltage and eliminates troublesome ground loops.

The PCX-4264/AC fully implements the PCI local bus specification Rev 2.1. All bus relative configurations, such as base memory and interrupt assignment, are automatically controlled by BIOS software.

### 1.2 Features

The PCX-4264/AC Isolated digital I/O card provide the following advanced features:

- ◆ 32 Isolated digital Input channels (non-polarity input for PCX4264AC)
- ◆ 32 Isolated digital output channels
- ◆ High output driving capability, 500mA sink current on isolated output
- ◆ 5000 Vrms high voltage isolation
- ◆ External interrupt signal on DI channels
- ◆ Built-in digital debounce
- ◆ 68-pin SCSI-II connector (Pin compatible to PDB-8068 )(see page 40 )

### 1.3 Applications

- ◆ Laboratory and Industrial automation
- ◆ Watchdog timer
- ◆ Event counter
- ◆ Frequency counter and generator
- ◆ Low level pulse generator
- ◆ Time delay

## 1.4 Specifications

### ◆ Optical Isolated Input Channel

Numbers of Channel: 32 digital inputs

Input polarity: polarity sensitive for PCX-4264, and non-polarity for PCX-4264AC

Input Voltage: 0 - 24V dc

Logic H: 3~24V

Logic L: 0~2.4V

Input resistance: 4.7K $\Omega$  @ 0.5W

Isolated voltage: 5000 Vrms

Throughput: 10K Hz (0.1 ms)

### ◆ Optical Isolated Output Channel

Numbers of Channel: 32 digital outputs

Output type: Darlington transistors with common ground

Output voltage: 5V<sub>DC</sub> min, 90V<sub>DC</sub> maximum

Output Device:ULN2803(common ground)

Sink Current: Max. 500mA/ch for only one of the ULN2803 transistor is ON

Power Dissipation: 1.47W per ULN2803 device (8 channels)

Isolated voltage: 5000 V<sub>DC</sub>

### ◆ Interrupt Sources

Channel 0 to channel 7 of digital input channels

### ◆ Digital debounce

Software programmable from 10msec to 160 msec

### ◆ General Specifications

Connector: 68-pin SCSI-II connector

Operating temperature: 0°C ~ 60°C

Storage temperature: -20°C ~ 80°C

Humidity: 5 ~ 95%, non-condensing

Power Consumption: +5V @ 530 mA typical

Dimension: 170mm(W) x102mm (H)

## 1.5 Software Supporting

**Inlog** provides versatile software drivers and packages for users' different approach to built-up a system. We not only provide programming library such as DLL for many Windows systems, but also provide drivers for many software package such as LabVIEW™, InTouch™ and so on. All the software options are included in the provided CD.

## 1.6 Programming Library

The provided CD includes the function libraries for many different operating systems, including:

- ◆ **DOS Library:** Borland C/C++ and Microsoft C++, the functions descriptions are included in this user's guide.
- ◆ **Windows 98/2000/NT/Me/XP DLL:** For VB, VC++, BC5, the functions descriptions are included in this user's guide.
- ◆ **Windows 98/2000/NT/Me/XP ActiveX:** For Windows's applications
- ◆ **LabVIEW ® Driver:** Contains the VIs, which are used to interface with NI's LabVIEW ® software package. Supporting Windows 95/98/NT/2000. The LabVIEW ® drivers are free shipped with the board.
- ◆ **InTouch Driver:** Contains the InTouch driver which support the Windows 98/2000/NT/XP. The The InTouch ® drivers are free shipped with the board.

---

# Chapter 2

## Installation

---

This chapter describes how to install the PCX-4264/AC card. Please follow the follow steps to install the PCX-4264/AC card.

### 2.1 What You Have

In addition to this *User's Manual*, the package includes the following items:

- ◆ PCX-4264/AC board
- ◆ Driver/utilities CD
- ◆ This user's manual

If any of these items is missing or damaged, contact the dealer from whom you purchased the product. Save the shipping materials and carton in case you want to ship or store the product in the future

### 2.2 Unpacking

Your PCX-4264/AC card contains sensitive electronic components that can be easily damaged by static electricity. The operator should be wearing an anti-static wristband, grounded at the same point as the anti-static mat. Inspect the card module carton for obvious damage. Shipping and handling may cause damage to your module. Be sure there are no shipping and handling damages on the module before processing.

After opening the card module carton, extract the system module and place it only on a grounded anti-static surface component side up. Again inspect the module for damage. Press down on all the socketed IC's to make sure that they are properly seated. Do this only with the module place on a firm flat surface.

### 2.3 Hardware Installation Outline

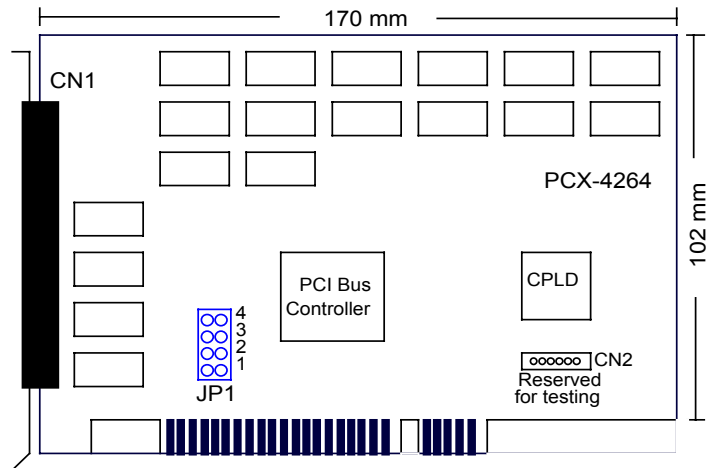
#### ◆ PCI configuration

The PCI cards are equipped with plug and play PCI controller, it can request base addresses and interrupt according to PCI standard. The system BIOS will install the system resource based on the PCI cards' configuration registers and system parameters (which are set by system BIOS). Interrupt assignment and memory usage (I/O port locations) of the PCI cards can be assigned by system BIOS only. These system resource assignments are done on a board-by-board basis. It is not suggested to assign the system resource by any other methods.

#### ◆ PCI slot selection

The PCI card can be inserted to any PCI slot without any configuration for system resource.

## 2.4 PCB Layout



Where

- CN1: Digital input/output connector
- JP1: Card number selection jumper
- CN2: Testing only, no used for user

### Installation Procedures

1. Turn off your computer.
2. Turn off all accessories (printer, modem, monitor, etc.) connected to your computer.
3. Remove the cover from your computer.
4. Setup jumpers on the card.
5. Before handling the PCI cards, discharge any static buildup on your body by touching the metal case of the computer. Hold the edge and do not touch the components.
6. Position the board into the PCI slot you selected.
7. Secure the card in place at the rear panel of the system.

### 2.5 Device Installation for Windows Systems

Once Windows 95/98/2000 has started, the Plug and Play function of Windows system will find the new PCX cards. If this is the first time to install PCX cards in your Windows system, you will be informed to input the device information source. Please refer to the ***“Software Installation Guide”*** for the steps of installing the device.

## 2.6 Connector Pin Assignment of PCX-4264/AC

The pin assignment of the 68 pins SCSI-II connector is an isolated signal connector, 4264/AC's pin assignment is as shown in Figure 2.6

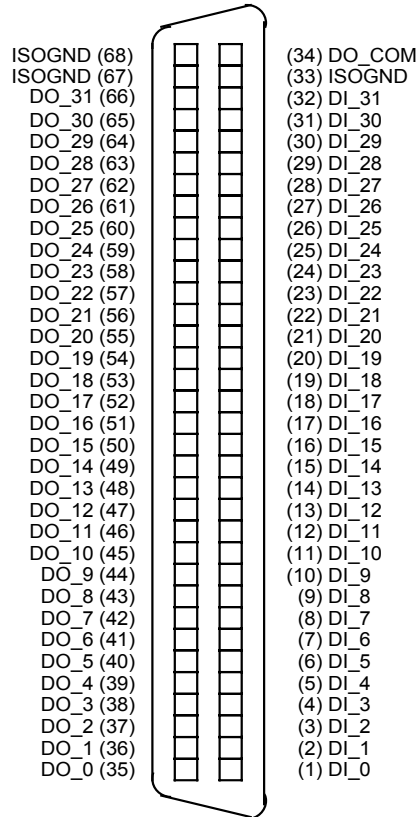


Figure 2.6 Pin Assignment of PCX-4264/AC connector CN1

**Legend:**

- ◆ **DI\_n**: Isolated digital input channel #n
- ◆ **DO\_n**: Isolated digital output channel #n
- ◆ **DO\_COM**: Power input signal for fly-wheel diode of DO channels
- ◆ **ISOGND**: Ground return path of isolated input and output channels

## 2.7 Card number setting

Maximum four PCX-4264/AC cards can be installed in system simultaneously with each has a unique card number.

A jumper called “JP1” (see page 8) on the card is used to set the card number starts from 1 to 4

JP1	Card number
	1 (default setting)
	2
	3
	4

## Chapter 3

# Registers Format

This information is quite useful for the programmers who wish to handle the card by low-level programming. However, we suggest user have to understand more about the PCI interface then start any low-level programming. In addition, the contents of this chapter can help users understand how to use software driver to manipulate this card.

### 3.1 PCI PnP Registers

There are two types of registers: PCI Configuration Registers (PCR) and Peripheral Interface Bus (PIB). The PCR, which is compliant to the PCI-bus specifications, is initialized and controlled by the plug & play (PnP) PCI BIOS..

The PCI bus controller Tiger 100/320 is provided by Tigerjet Network Inc. ([www.tjnet.com](http://www.tjnet.com) ). For more detailed information of PIB, please visit Tigerjet technology's web site to download relative information. It is not necessary for users to understand the details of the PIB if you use the software library. The PCI PnP BIOS assigns the base address of the PIB. The assigned address is located at offset 14h of PIB .

The 4264/AC board registers are in 32-bit width. But only lowest byte (bit0~bit7) is used. The users can access these registers by only 32-bit I/O or 8-bit I/O instructions. The following sections show the address map, including descriptions and their offset addresses relative to the base address.

### 3.2 Digital Input Register Address Map

There are 32 isolated digital input channels on PCX-4264/AC, each bit of based address is corresponding to a signal on the digital input channel.

**Address:** BASE + 0xc0~ BASE + 0xcc (port 0 ~ port 3)

**Attribute:** read only

Bit	Port	7	6	5	4	3	2	1	0
Base+0xc0	0	DI_7	DI_6	DI_5	DI_5	DI_3	DI_2	DI_1	DI_0
Base+0xc4	1	DI_15	DI_14	DI_13	DI_12	DI_11	DI_10	DI_9	DI_8
Base+0xc8	2	DI_23	DI_22	DI_21	DI_20	DI_19	DI_18	DI_17	DI_16
Base+0xcc	3	DI_31	DI_30	DI_29	DI_28	DI_27	DI_26	DI_25	DI_24

### 3.3 Digital Output Register

There are total 32 digital output channels on the PCX-4264/AC, each bit of based address is corresponding to a signal on the digital output channel.

**Address:** BASE + 0xd0 ~ BASE +0xdc (port 0 ~ port 3)

**Attribute:** write /read

Bit	Port	7	6	5	4	3	2	1	0
Base+0xd0	0	DO_7	DO_6	DO_5	DO_5	DO_3	DO_2	DO_1	DO_0
Base+0xd4	1	DO_15	DO_14	DO_13	DO_12	DO_11	DO_10	DO_9	DO_8
Base+0xd8	2	DO_23	DO_22	DO_21	DO_20	DO_19	DO_18	DO_17	DO_16
Base+0xdc	3	DO_31	DO_30	DO_29	DO_28	DO_27	DO_26	DO_25	DO_24

### 3.4 Reset control registers

The PCX-4264/AC is in inactive state when the system power on, and should be activated by set bit 0 of this register to "1" state

**Address:** Base + 0x00

**Attribute:** Write only

Bit	7	6	5	4	3	2	1	0	State
Base+0x00	0	0	0	0	0	0	0	0	Inactive (reset) state (Default)
Base+0x00	0	0	0	0	0	0	0	1	Active state

♦ Bit 0 of this register should be set to "1" before using PCX-4264/AC

# Chapter 4 Operation Theorem

## 4.1 Isolated Digital Input Channels

The isolated digital input is open collector transistor structure. The input voltage range form 0V to 24V and input resistor is 4.7K ohms. The connection between outside signal and PCX-4264/AC is shown in Figure 4-1. and Figure 4-2

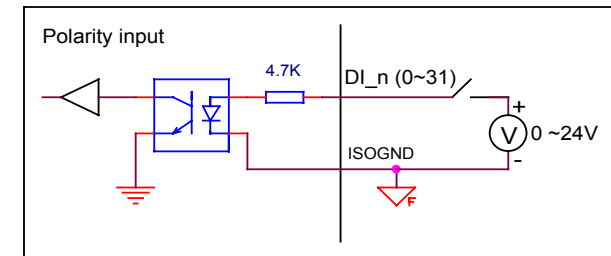


Figure 4-1 Isolated digital inputs of PCX-4264

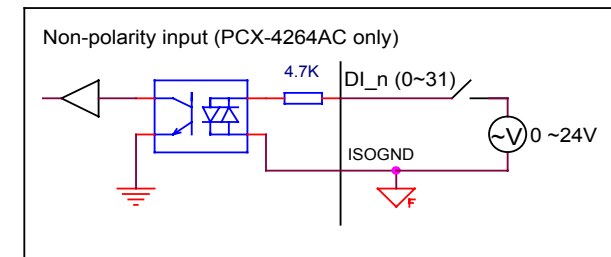


Figure 4-2 Non-polarity isolated digital inputs of PCX-4264AC

**Note:** The digital input connections of PCX-4264AC are not polarity sensitive whether used on AC or DC voltage.

### 4.2 Isolated Digital Output Channels

On PCX-4264/AC, the DO\_COM pin is used as “fly-wheel” diode, which can protect the driver if the loading is inductance loading such as relay, motor or solenoid. If the loading is resistance loading such as resistor or LED, the connection to fly-wheel diode is not necessary.

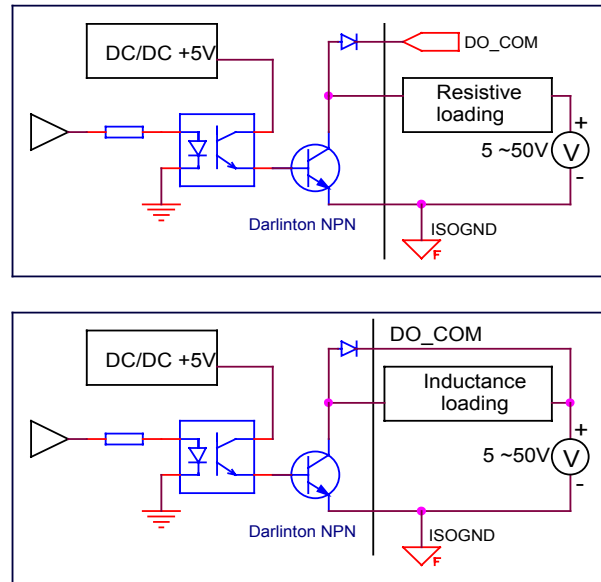
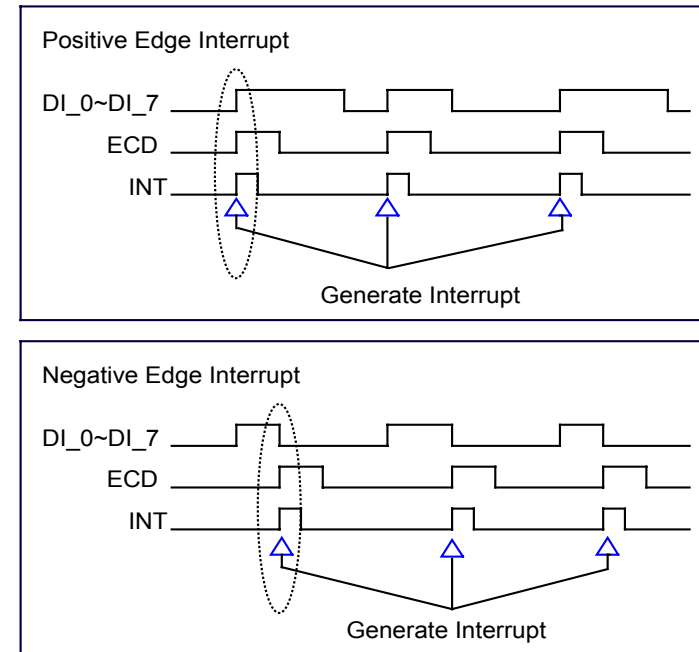


Figure 4-3 isolated digital outputs

Please note that when the loading is as “inductance type loading” such as relay, coil or motor, the DO\_COM pin must be connected to the external power source. The extra connection is to utilize the ‘fly-wheel diode’ to form a current-release closed loop, so that the transistor won’t be destroyed by the reverse high voltage which is generated by the inductance load when the output switches from “ON” to “OFF”.

### 4.3 Edge Change Detection

The ECD (Edge Change Detection) detection circuit is used to detect the edge of level change. In the PCX-4264/AC, the detection circuit is applied to 8 input channels (DI\_0 ~ DI\_7). If channel is programmed to be positive edge or negative edge interrupt mode, the ECD detection circuit generate an interrupt request, when the signal inputs are changed from low to high level or high to low level respectively



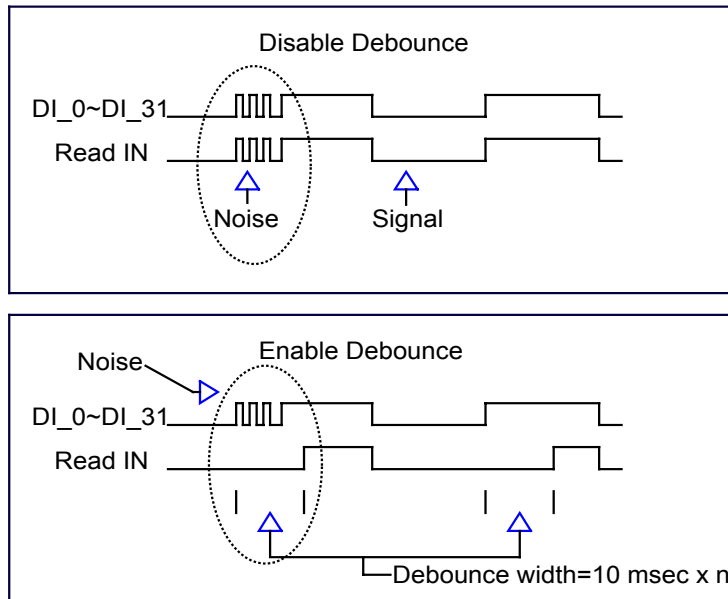
#### 4.4 Digital debounce

Each digital input channel has a programmable digital debounce for eliminating unexpected signals and noise from the card circuitry. The user can set different digital debouncing parameters for each input channel in different applications. The following is a functional description of the digital debounce.

1. When a digital debounce is enabled, the PCXI-4264/AC will sample the signals at the enabled input channel at a 10 ms sampling rate.
2. When a high or low signal is present at a digital input channel whose digital debounce function is enabled, the signal will be filtered out as noise unless it lasts for an effective period.
3. The effective period is determined by multiplying the sampling rate (10 ms) by the sampling number (1 ~ 15) chosen by the user, i.e.

$$\text{Effective period} = \text{Sampling number} \times 10 \text{ ms.}$$

4. See Sec 5.10 (page 30) to more detail using of debounce function



---

## Chapter 5 Libraries

---

This chapter describes the software library for operating this card. Only the functions in DOS library and Windows DLL are described. Please refer to the PCIDAQ function reference manual, which included in Inlog CD, for the descriptions of the Windows 98/NT/2000 DLL functions.

### 5.1 Libraries Installation

Please refer to the "**Software Installation Guide**" for the detail information about how to install the software libraries for DOS, or Windows 98 DLL, or PCIDAQ for Windows 98/NT/2000.

The device drivers and DLL functions of Windows 98/NT/2000 are included in the PCIDAQ. The Inlog CD also includes the detail examples and readme files

### 5.2 How to use the Functions in PCIDAQ.DLL

#### ◆ VC++6.0:

1. Add file './Include/PCIDAQ.H' in your project
2. In link page of menu project| setting, add './LIB/PCIDAQ.LIB' in the blank of Objects/Library Modules
3. Add this sentence "#include './Include/PCIDAQ.H' " to the head of your main file.

#### ◆ Visual BASIC :

1. Add file './Include/Declare.bas' in your project.

#### ◆ Delphi :

2. Add file './Include/Declare.pas' in your project
3. Add this sentence "uses Declare;" in the head of your unit.pas

#### ◆ C++Builder:

4. Add file './Include/PCIDAQ.H' and './Lib/PCIDAQ\_CB.lib' to your project
5. Add this sentence "#include './Include/PCIDAQ.H' " to head of your main file.

**Note:** For more information, please refer to program in directory './Example/'

### 5.3 Summary of function calls

Function	Description	page
Open card	Initial PCX-4264/AC card before using	24
Get Card's ID	Get PCI ID code of PCX-4264/AC	25
Get Driver Version	Get version number of PCIDAQ.DLL	26
Close card	Close PCX-4264/AC card before terminating program	27
Get PCI Bus and Slot number	Get PCI bus and slot number occupied by PCX-4264/AC	28
Read digital input data	Read digital input port data (8-bit)	29
Set debounce time	Set debounce timer of digital input signals	30
Write data to digital output port	Write data (8-bit) to digital output port	31
Read back digital output data	Read back current value of digital output port	32
Set bit of digital output port	Activate a bit of digital output port (output transistor ON)	33
Reset bit of digital output port	De-activate a bit of digital output port (output transistor OFF)	34
Enable interrupt	Enable interrupt by input(DI_0 ~ DI_7)	35
Disable interrupt	Disable digital input interrupt	37
Read interrupt status	Read channels which generate interrupt	38
Clear interrupt status register	Clear interrupt status register	39

### 5.4 Open card

#### Description:

Because the PCX-4264/AC is PCI bus architecture and meets the plug and play design, the IRQ and base address are assigned by system BIOS directly. PCX-4264/AC cards have to be initialized by this function before calling other functions.

#### Syntax:

##### C/C++(DOS)

```
WORD D_4264_Open(WORD cardNo);
```

##### C/C++ (Windows)

```
WORD W_4264_Open(WORD *ExistedCards);
```

##### Visual BASIC(Windows)

```
W_4264_Open (ByRef ExistedCards As Long) As Long
```

##### Delphi

```
W_4264_Open(var ExistedCards:Integer):Integer;
```

#### Argument:

cardNo: card number set by jumper on card (DOS only)

existCards: The number of installed PCX-4264/AC cards. (Windows only)

This returned value shows how many PCI-4264/AC cards are installed in your system.

#### Return Code:

Error Code (Please refer to PCIDAQ.H or DOSDAQ.H)

## 5.5 Get Card's ID:

### Description:

Get the cards number which is set by jumper on cards.

### Syntax:

#### C/C++(DOS)

```
void D_4264_GetCardsID(WORD *CardsIDArray);
```

#### C/C++(Windows)

```
WORD W_4264_GetCardsID(WORD *CardsIDArray);
```

#### Visual BASIC(Windows)

```
W_4264_GetCardsID(ByRef CardsIDArray As Long) As Integer
```

#### Delphi

```
W_4264_GetCardsID(var CardsIDArray:Word):Word;
```

### Argument:

CardsIDArray : This array return card number(1,2,3,4)which is set by jumper on card. You should define a 4 elements array, then pass the array's pointer to this function.

### Return Code:

Error Code (Please refer to PCIDAQ.H or DOSDAQ.H)

## 5.6 Get Driver Version

### Description:

The PCX-4264/AC card is driven by PCIDAQ. DLL driver. This function returns the version of PCIDAQ.DLL driver

### Syntax:

#### C/C++(DOS):

```
void D_4264_Version(char *version);
```

#### C/C++ (Windows)

```
Int W_4264_Version(void);
```

#### Visual BASIC(Windows)

```
W_4264_Version() As Long
```

#### Delphi

```
W_4264_Version():Integer;
```

### Argument:

Version: This string return the version of DOSDAQ.DLL (DOS)

### Return Code:

Error Code (Please refer to PCIDAQ.H or DOSDAQ.H)

## 5.7 Close card

### Description:

The IRQ and base address of PCX-4264/AC ( pass-through address) are assigned by system BIOS directly. This function should be called to release all system resource before terminate application program

### Syntax:

#### C/C++(DOS)

```
WORD D_4264 _Close(WORD cardNo);
```

#### C/C++ (Windows)

```
Void W_4264 _Close(void);
```

#### Visual BASIC(Windows)

```
W_4264 _Close ()
```

#### Delphi

```
W_4264 _Close ();
```

### Argument:

cardNo : card number to select borad (1,2,3,4),It's set by jumper on card

### Return Code:

Error Code (Please refer to PCIDAQ.H or DOSDAQ.H)

## 5.8 Get PCI Bus and Slot number

### Description:

Get the PCI bus and slot number occupied by PCX-4264/AC card

### Syntax:

#### C/C++(DOS)

```
WORD D_4264 _GetBusSlot(WORD cardNo, WORD *bus,WORD *slot);
```

#### C/C++ (Windows)

```
WORD W_4264 _GetBusSlot(WORD cardNo, WORD *bus,WORD *slot);
```

#### Visual BASIC(Windows)

```
W_4264 _GetBusSlot (ByVal cardNo As Long, ByRef bus As Long, ByRef slot As Long) As Long
```

#### Delphi

```
W_4264 _GetBusSlot(cardNo:Integer;var bus:Integer;var slot:Integer):Integer;
```

### Argument:

cardNo : card number to select borad (1,2,3,4),It's set by jumper on card

bus: return PCI bus Number

slot: return PCI slot Number of the bus

### Return Code:

Error Code (Please refer to PCIDAQ.H or DOSDAQ.H)

## 5.9 Read digital input data

### Description:

This function is used to read data from digital input port. You can get 8-bit input data from PCX-4264/AC by calling this function.

### Syntax:

#### C/C++(DOS)

```
WORD D_4264_Read_Di(WORD cardNo,WORD portNo,WORD *DiData);
```

#### C/C++ (Windows)

```
WORD W_4264_Read_Di(WORD cardNo,WORD portNo,WORD *DiData);
```

#### Visual BASIC(Windows)

```
W_4264_Read_Di(ByVal cardNo As Long, ByVal portNo As Long, ByRef DiData As Long) As Long
```

#### Delphi

```
W_4264_Read_Di(cardNo:Integer;portNo:Integer;var DiData:Integer): Integer;
```

### Argument:

cardNo : card number to select board (1,2,3,4),It's set by jumper on card

portNo : Digital Input port number (0 ~ 3)

Didata : return digital input data

### Return Code:

Error Code (Please refer to PCIDAQ.H or DOSDAQ.H)

## 5.10 Set debounce time of digital inputs

### Description:

The all digital input channels (DI\_0 ~ DI\_31) are grouped into 4 ports, each port can has an individually programmable digital debounce circuit which can filter the bounce of input signals

### Syntax:

#### C/C++(DOS)

```
WORD D_4264_Set_DebounceMode (WORD cardNo,WORD DebounceMode);
```

#### C/C++ (Windows)

```
WORD W_4264_Set_DebounceMode (WORD cardNo,BYTE DebounceMode);
```

#### Visual BASIC(Windows)

```
W_4264_Set_DebounceMode (ByVal cardNo As Long, ByVal DebounceMode As Long) As Long
```

#### Delphi

```
W_4264_Set_DebounceMode (cardNo:Integer;DebounceMode:Integer):Integer;
```

### Argument:

cardNo : card number to select borad (1,2,3,4),It's set by jumper on card

DebounceMode :

Bit0 : =1/0 Enable/Disable Port 0 (DI\_0 ~ DI\_7) debounce

Bit1 : =1/0 Enable/Disable Port 1 (DI\_8 ~ DI\_15) debounce

Bit2 : =1/0 Enable/Disable Port 2 (DI\_16 ~ DI\_23) debounce

Bit3 : =1/0 Enable/Disable Port 3 (DI\_24 ~ DI\_31) debounce

Bit4~Bit7 : 0001~1111 debounce time factor.

for example :

Bit4~Bit7=0010 debounce time=10ms\*2=20ms

Bit4~Bit7=0101 debounce time=10ms\*5=50ms

### Return Code:

Error Code (Please refer to PCIDAQ.H or DOSDAQ.H)

## 5.11 Write data to digital output port

### Description:

This function is used to write data (byte) to output port. You can send 8-bit output data to PCX-4264/AC by calling this function.

### Syntax:

#### C/C++(DOS)

```
WORD D_4264_Write_Do(WORD cardNo,WORD portNo,WORD Data);
```

#### C/C++ (Windows)

```
WORD W_4264_Write_Do(WORD cardNo,WORD portNo,WORD Data);
```

#### Visual BASIC(Windows)

```
W_4264_Write_Do (ByVal cardNo As Long, ByVal portNo As Long, ByVal Data As Long) As Long
```

#### Delphi

```
W_4264_Write_Do(cardNo:Integer;portNo:Integer;Data:Integer):Integer;
```

### Argument:

cardNo : card number (1,2,3,4),It's set by jumper on card

portNo : Digital port number (0 ~ 3)

Data : Data be written to output port

### Return Code:

Error Code (Please refer to PCIDAQ.H or DOSDAQ.H)

## 5.12 Read back digital output data

### Description:

This function is used to read current data of output port. You can read back 8-bit output data of PCX-4264/AC by calling this function.

### Syntax:

#### C/C++(DOS)

```
WORD D_4264_Read_Do(WORD cardNo,WORD portNo,WORD *DoData);
```

#### C/C++ (Windows)

```
WORD W_4264_Read_Do(WORD cardNo,WORD portNo,WORD *DoData);
```

#### Visual BASIC(Windows)

```
W_4264_Read_Do (ByVal cardNo As Long, ByVal portNo As Long, ByRef DoData As Long) As Long
```

#### Delphi

```
W_4264_Read_Do(cardNo:Integer;portNo:Integer;var DoData:Integer):Integer;
```

### Argument:

cardNo : card number (1,2,3,4),It's set by jumper on card

portNo : Do port number (0 ~ 3)

Data : return current output data

### Return Code:

Error Code (Please refer to PCIDAQ.H or DOSDAQ.H)

### 5.13 Set bit of digital output port

#### Description:

Set one digital output bit to **short** state (output transistor ON).

#### Syntax:

##### C/C++(DOS)

```
WORD D_4264_Set_Do_Bit (WORD cardNo,WORD portNo, WORDbitNo);
```

##### C/C++ (Windows)

```
WORD W_4264_Set_Do_Bit (WORD cardNo,WORD portNo, WORDbitNo);
```

##### Visual BASIC(Windows)

```
W_4264_Set_Do_Bit(ByVal cardNo As Long, ByVal portNo As Long, ByVal  
bitNo As Long) As Long
```

##### Delphi

```
W_4264_Set_Do_Bit (cardNo:Integer;portNo:Integer;  
bitNo:Integer):Integer;
```

#### Argument:

cardNo : card number (1,2,3,4),It's set by jumper on card

portNo : Digital output port number (0 ~ 3)

bitNo :bit number(0 to 7)

#### Return Code:

Error Code (Please refer to PCIDAQ.H or DOSDAQ.H)

### 5.14 Reset bit of digital output port

#### Description:

Set one digital output bit to **open** state (output transistor OFF)

#### Syntax:

##### C/C++ (DOS)

```
WORD D_4264_Reset_Do_Bit(WORD cardNo,WORD portNo, WORD  
bitNo);
```

##### C/C++ (Windows)

```
WORD W_4264_Reset_Do_Bit(WORD cardNo,WORD portNo, WORD  
bitNo);
```

##### Visual BASIC(Windows)

```
W_4264_Reset_Do_Bit (ByVal cardNo As Long, ByVal portNo As Long,  
ByVal bitNo As Long) As Long
```

##### Delphi

```
W_4264_Reset_Do_Bit(cardNo:Integer;portNo:Integer;  
bitNo:Integer):Integer;
```

#### Argument:

cardNo : card number to select board (1,2,3,4),It's set by jumper on card

portNo : Doport number (0 ~ 3)

bitNo :channel Number(0 to 7)

#### Return Code:

Error Code (Please refer to PCIDAQ.H or DOSDAQ.H)

## 5.15 Enable Interrupt

### Description:

Enable Interrupt of digital inputs

### Syntax:

#### C/C++(DOS)

```
WORD D_4264_IntEnable( WORD cardNo,WORD IntMode,WORD
IntMask,*UserIntServiceRoutine());
```

#### C/C++ (Windows)

```
WORD W_4264_IntEnable(WORD cardNo,WORD IntMode,WORD
IntMask,User_Interrupt_HANDLER userIntServiceRoutine);
```

#### Visual BASIC(Windows)

```
W_4264_IntEnable(ByVal cardNo As Long, ByVal IntMode As Long,
IntMask as Long,ByVal userIntServiceRoutine As Long) As Long
```

#### Delphi

```
W_4264_IntEnable(cardNo:Integer;IntMode:Integer;IntMask:Integer;userInt
ServiceRoutine:Pointer):Integer;
```

### Argument:

cardNo : card number (1,2,3,4),It's set by jumper JP1 on card

IntMode: Interrupt mode of input port 0 (DI\_0 ~ DI\_7)

bit0=1/0:Rising/Falling edge Interrupt of DI\_0

bit1=1/0:Rising/Falling edge Interrupt of DI\_1

bit2=1/0:Rising/Falling edge Interrupt of DI\_2

bit3=1/0:Rising/Falling edge Interrupt of DI\_3

bit4=1/0:Rising/Falling edge Interrupt of DI\_4

bit5=1/0:Rising/Falling edge Interrupt of DI\_5

bit6=1/0:Rising/Falling edge Interrupt of DI\_6

bit7=1/0:Rising/Falling edge Interrupt of DI\_7

IntMask : Interrupt mask of channels.

bit0=1/0:Enable/Disable Interrupt of DI\_0

bit1=1/0:Enable/Disable Interrupt of DI\_1

bit2=1/0:Enable/Disable Interrupt of DI\_2

bit3=1/0:Enable/Disable Interrupt of DI\_3

bit4=1/0:Enable/Disable Interrupt of DI\_4

bit5=1/0:Enable/Disable Interrupt of DI\_5

bit6=1/0:Enable/Disable Interrupt of DI\_6

bit7=1/0:Enable/Disable Interrupt of DI\_7

userIntServiceRoutine: User Interrupt service routine pointer will be called when interrupt occurs.

for C++:

```
void userIntServiceRoutine(WORD CardNo,WORD IntStatus);
```

for VB :

```
Sub UserInterruptRutine(ByVal CardNo As Long, ByVal IntStatus As Long)
```

for Delphi :

```
procedure userIntServiceRoutine (CardNo:Word;IntStatus:Word);StdCall;
```

This function will pass CardNo and IntStatus parameters to user's service routine when interrupt occurred.

Where IntStatus :

bit0=1 Interrupt from DI\_0

bit1=1 Interrupt from DI\_1

bit2=1 Interrupt from DI\_2

bit3=1 Interrupt from DI\_3

bit4=1 Interrupt from DI\_4

bit5=1 Interrupt from DI\_5

bit6=1 Interrupt from DI\_6

bit7=1 Interrupt from DI\_7

### Return Code:

Error Code (Please refer to PCIDAQ.H or DOSDAQ.H)

## 5.16 Disable Interrupt

### Description:

Disable interrupt of input

### Syntax:

#### C/C++(DOS)

```
WORD D_4264_IntDisable (WORD cardNo);
```

#### C/C++ (Windows)

```
Void W_4264_IntDisable (WORD cardNo);
```

#### Visual BASIC(Windows)

```
W_4264_IntDisable (ByVal cardNo As Long)
```

#### Delphi

```
W_4264_IntDisable (cardNo:Integer);
```

### Argument:

cardNo : card number (1,2,3,4),It's set by jumper on card

### Return Code:

Error Code (Please refer to PCIDAQ.H or DOSDAQ.H)

## 5.17 Read Interrupt Status Register

### Description:

Read the digital channel number which generate interrupt (DOS only)

### Syntax:

#### C/C++(DOS)

```
WORD D_4264_Read_IntStatus(WORD cardNo,WORD *IntStatus);
```

### Argument:

cardNo: card number to select board (1,2,3,4),It's set by jumper on card

IntStatus:

Bit0=1 DI\_0 Interrupt / Bit0=0 DI\_0 no Interrupt

Bit1=1 DI\_1 Interrupt / Bit1=0 DI\_1 no Interrupt

Bit2=1 DI\_2 Interrupt / Bit2=0 DI\_2 no Interrupt

Bit3=1 DI\_3 Interrupt / Bit3=0 DI\_3 no Interrupt

Bit4=1 DI\_4 Interrupt / Bit4=0 DI\_4 no Interrupt

Bit5=1 DI\_5 Interrupt / Bit5=0 DI\_5 no Interrupt

Bit6=1 DI\_6 Interrupt / Bit6=0 DI\_6 no Interrupt

Bit7=1 DI\_7 Interrupt / Bit7=0 DI\_7 no Interrupt

*DI\_n (see sec 2.6 page 10)*

### Return Code:

Error Code (Please refer to DOSDAQ.H)

## 5.18 Clear Interrupt Status Register

### Description:

Clear interrupt status register

### Syntax:

#### C/C++(DOS)

```
WORD D_4264_Clear_IntStatus(WORD cardNo);
```

#### C/C++ (Windows)

```
WORDW_4264_Clear_IntStatus(WORD cardNo);
```

#### Visual BASIC(Windows)

```
W_4264_Clear_IntStatus(ByVal cardNo As Long) As Long
```

#### Delphi

```
W_4264_Clear_IntStatus(cardNo:Integer):Integer;
```

### Argument:

cardNo : card number to select board (1,2,3,4), It's set by jumper on card

### Return Code:

Error Code (Please refer to PCIDAQ.H or DOSDAQ.H)

## Chapter 6

# PDB-8068 Terminal board

PDB-8068 digital input/output termination board features one DIN socket for easy maintenance, wiring, and installation. It provides 68 channels that are accessed through a SCSI-68 connector.

Each terminal pin is in series with 0 ohms resistor to relative pin on the DIN connector. These resistors can be changed to the desired value to meet the requirement of your applications

