NuDAQ® PCI-7442

64-CH Isolated Digital I/O Card

User's Manual

Manual Rev. 2.04

Revision Date: September 14, 2006

Part No: 50-11217-1010

Recycled Paper

Advance Technologies; Automate the World.

The information in this document 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.

In no event will the manufacturer be liable for direct, indirect, special, incidental, or consequential damages arising out of the use or inability to use the product or documentation, even if advised of the possibility of such damages.

Trademarks

NuDAQ, NuIPC, DAQBench are registered trademarks of ADLINK TECHNOLOGY INC.

Product names mentioned herein are used for identification purposes only and may be trademarks and/or registered trademarks of their respective companies.

Getting service from ADLINK

Customer satisfaction is our top priority. Please contact us should you require any service or assistance.

ADLINK TECHNOLOGY INC.

Web Site: http://www.adlinktech.com

Sales & Service: Service@adlinktech.com

TEL: +886-2-82265877

FAX: +886-2-82265717

Address: 9F, No. 166, Jian Yi Road, Chungho City, Taipei, 235 Taiwan

Please email or FAX this completed service form for prompt and satisfactory service.

Company Information	
Company/Organization	
Contact Person	
E-mail Address	
Address	
Country	
TEL	FAX:
Web Site	
Product Information	
Product Model	
Environment	OS: M/B: CPU: Chipset: BIOS:

Please give a detailed description of the problem(s):

Table of Contents

List of Tables

List of Figures

1 Introduction

The ADLINK PCI-7442 is a high-density isolated digital I/O card featuring 64 channels of digital input, 64 channels of digital output, and 32 TTL channels for PCI bus-based industrial applications.

The PCI-7442 provides 64 opto-isolated digital inputs/outputs. It provides a robust 1,250 V_{RMS} isolation protection which is suitable for most industrial applications. All of the opto-isolated digital inputs are identical non-polarity, meaning each digital input line is isolated respectively and suited to collect digital inputs even in noisy environments. The PCI-7442 also features a Change of State (COS) interrupt function. The COS function sends an interrupt when any of the inputs change its state, enabling users to efficiently handle these external events.

For PCI chassis with multiple PCI-7442 cards installed, the board ID design feature enables convenient identification of the cards through a switch jumper, allowing quick troubleshooting and maintenance.

1.1 Features

The PCI-7442 isolated digital I/O card has the following features:

- Universal 32-bit, 3.3/5 V PCI bus, plug-and-play
- High-density, opto-isolated 64-CH digital input and 64-CH digital output
- Programmable Change of State (COS) detection for all digital input channels
- Voltage protection of up to 28 V for isolated input
- Available dry contact input
- High-output driving capability
- 250 mA sink current on isolated output channels
- Digital output status read back function
- Digital output value retention after hot system reset
- Programmable power-up DO initial status
- Programmable safe DO status functions even when WDT interrupt occurs
- Watch dog timer counter prevents system from crashing
- 32-CH programmable TTL I/O function
- Board ID function

1.2 Applications

- Machine automation
- Industrial on/off control
- External relay driving
- Signal switching
- Laboratory automation

1.3 Specifications

Optical isolated digital input	
Input channels	64
Input type	Opto-isolated
Input Device	PC-3H4
Maximum input range	28 V, non-polarity
Digital logic levels	0 V – 28 V, non-polarity Input high voltage: 5 V – 28 V Input low voltage: 0 V – 1.5 V
Input resistance	2.4 kΩ at 0.5 W
Allowed input current	15 mA per channel (max)
Isolated voltage	1250 V_{RMS}
Input protection	ESD protection circuit (Forward direction)
Interrupt source	Change of State for 64 lines Watch dog timer counter
Data transfers:	Programmed I/O
Optical isolated digital output	
Output channels	64
Output type	Open drain power MOSFET driver
Output device	TPC8206
Output supply voltage	5 V – 40 V
Sink current	250 mA for all channel @ 60°C, 100% duty (300 mA max.)
Isolated voltage	1250 V_{RMS}
Data transfers	Programmed I/O
Isolated +5V power supply	
Output voltage	+5 V
Output current	100 mA maximum at 40°C
Programmable TTL I/O	
Number of I/O channels	32
Digital logic level	TTL / 3.3 V TTL
Current rating	4 mA (max) per channel
Data transfer	Programmed I/O

Watchdog timer	
Base clock available	10 MHz (fixed)
Counter-width	32-bit
Safety functions	
• Programmable power-up DO initial status	
• Programmable safety DO status function even during WDT interruption	
• Digital output value retention after hot system reset	
General specifications	
Dimensions	174.7 mm (L) x 106.7 mm (W), standard PCI
Bus	32-bit PCI bus
Operating temperature	0°C – 60°C
Storage temperature	-40°C – 80°C
Humidity	5 to 85% non-condensing
Power	
Power consumption	+5 V at 800 mA (typical)

1.4 Software support

ADLINK provides versatile software drivers and packages to address different approaches in building a system. We not only provide programming libraries such as DLLs for many Windows® - based systems, but also provide drivers for other software packages including LabVIEW®. All software options may be found in the ADLINK All-in-One CD.

Programming library

If you are writing you own programs, the following function libraries are available:

DOS Library

For Borland C/C++, and Visual C++, the functions descriptions are included in this user's guide.

PCIS-DASK

Included device drivers and DLL for Windows® 98/NT/2000/XP. A DLL is a binary compatible across Windows® 98/NT/2000/XP. That means all applications developed with PCIS-DASK are compatible across Windows® 98/NT/2000/XP. The developing environment can be VB, VC++, Delphi, BC5, or any Windows® programming language that allows calls to a DLL. The user's guide and function reference manual of PCIS-DASK are in the CD. Refer to the manual files in the All-in-One CD (\\Manual_PDF\Software\PCIS-DASK).

These software drivers are shipped with the board. Refer to the **Software Installation Guide** for installation procedures.

DAQ-LVIEW PnP: LabVIEW® Driver

DAQ-LVIEW PnP contains VIs that are used to interface with the LabVIEW® software package. DAQ-LVIEW PnP supports Windows® 95/98/NT/2000/XP. The LabVIEW® drivers are shipped free with the board. You can install and use them without a license. For more information about DAQ-LVIEW PnP, refer to the user's guide in the All-in-One CD.

DAQBenchTM: ActiveX Controls

It is recommended for programmers familiar with ActiveX controls and VB/VC++ programming to use the DAQBench™ ActiveX Control component library for developing applications. The DAQBench™ is designed under Windows® NT/98 environment. For more information about DAQBench™, refer to the user's guide in the All-in-One CD.

2 Getting started

This chapter provides information on the PCI-7442 card package, card layout, connectors, and pin assignments.

2.1 Unpacking checklist

Before unpacking, check the shipping carton for any damage. If the shipping carton and/or contents are damaged, inform your dealer immediately. Retain the shipping carton and packing materials for inspection. Obtain authorization from your dealer before returning any product to ADLINK.

Check if the following items are included in the package.

- PCI-7442 card
- ACL-10337 DB37F bracket
- ADLINK All-in-One CD
- User's manual

If any of the items is damaged or missing, contact your dealer immediately.

NOTE The packaging of OEM versions with non-standard configuration, functionality, or package may vary according to different configuration requests.

CAUTION 

The boards must be protected from static discharge and physical shock. Never remove any of the socketed parts except at a static-free workstation. Use the anti-static bag shipped with the product to handle the board. Wear a grounded wrist strap when servicing.

2.2 Card layout

Figure 2-1 shows the location of the PCI-7442 connectors, switch, and jumpers.

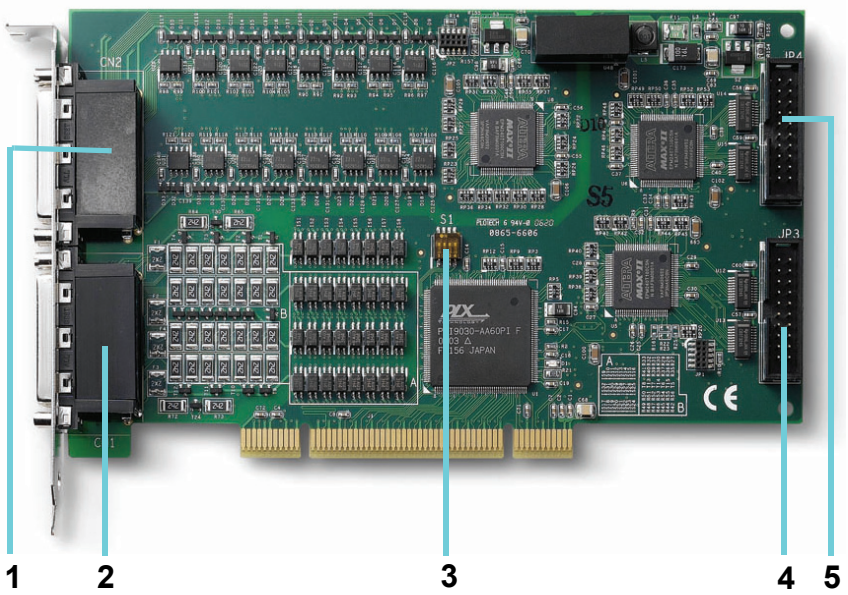


Figure 2-1: PCI-7442 layout

1	**CN2**	64-CH isolated digital output connector
2	**CN1**	64-CH isolated digital input connector
3	**S1**	Board ID DIP switch
4	**JP3**	16-CH (TTL0~16) TTL I/O connector
5	**JP4**	16-CH (TTL16~31) TTL I/O connector

Bracket layout

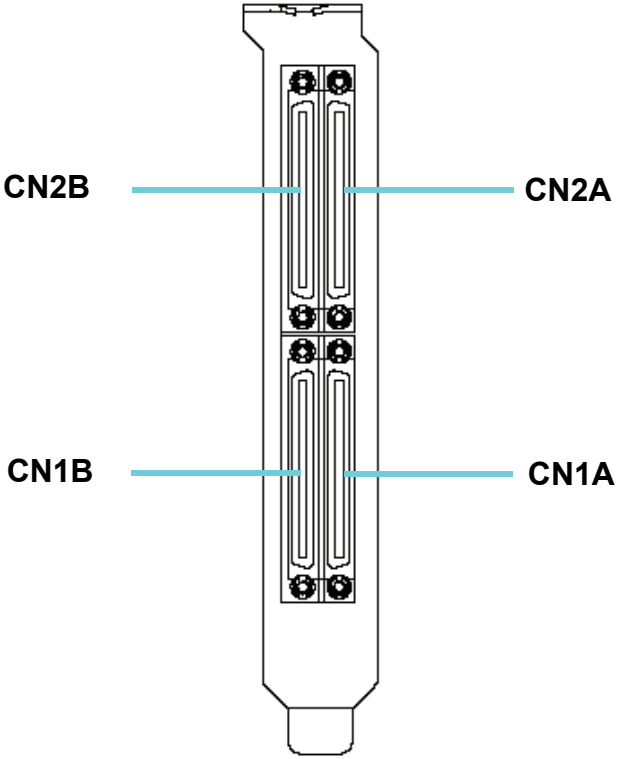
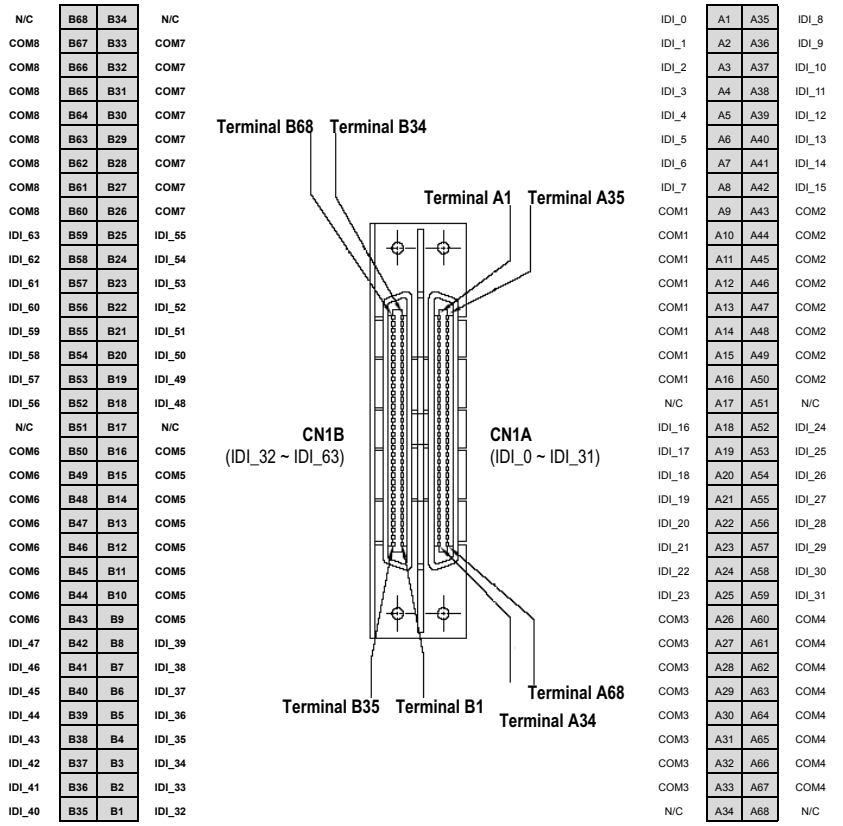


Figure 2-2: PCI-7442 bracket

2.3 DI (CN1) connector pin assignments



N/C	B68	B34	N/C
COM8	B67	B33	COM7
COM8	B66	B32	COM7
COM8	B65	B31	COM7
COM8	B64	B30	COM7
COM8	B63	B29	COM7
COM8	B62	B28	COM7
COM8	B61	B27	COM7
COM8	B60	B26	COM7
IDI_63	B59	B25	IDI_55
IDI_62	B58	B24	IDI_54
IDI_61	B57	B23	IDI_53
IDI_60	B56	B22	IDI_52
IDI_59	B55	B21	IDI_51
IDI_58	B54	B20	IDI_50
IDI_57	B53	B19	IDI_49
IDI_56	B52	B18	IDI_48
N/C	B51	B17	N/C
COM6	B50	B16	COM5
COM6	B49	B15	COM5
COM6	B48	B14	COM5
COM6	B47	B13	COM5
COM6	B46	B12	COM5
COM6	B45	B11	COM5
COM6	B44	B10	COM5
COM6	B43	B9	COM5
IDI_47	B42	B8	IDI_39
IDI_46	B41	B7	IDI_38
IDI_45	B40	B6	IDI_37
IDI_44	B39	B5	IDI_36
IDI_43	B38	B4	IDI_35
IDI_42	B37	B3	IDI_34
IDI_41	B36	B2	IDI_33
IDI_40	B35	B1	IDI_32

IDI_0	A1	A35	IDI_8
IDI_1	A2	A36	IDI_9
IDI_2	A3	A37	IDI_10
IDI_3	A4	A38	IDI_11
IDI_4	A5	A39	IDI_12
IDI_5	A6	A40	IDI_13
IDI_6	A7	A41	IDI_14
IDI_7	A8	A42	IDI_15
COM1	A9	A43	COM2
COM1	A10	A44	COM2
COM1	A11	A45	COM2
COM1	A12	A46	COM2
COM1	A13	A47	COM2
COM1	A14	A48	COM2
COM1	A15	A49	COM2
COM1	A16	A50	COM2
N/C	A17	A51	N/C
IDI_16	A18	A52	IDI_24
IDI_17	A19	A53	IDI_25
IDI_18	A20	A54	IDI_26
IDI_19	A21	A55	IDI_27
IDI_20	A22	A56	IDI_28
IDI_21	A23	A57	IDI_29
IDI_22	A24	A58	IDI_30
IDI_23	A25	A59	IDI_31
COM3	A26	A60	COM4
COM3	A27	A61	COM4
COM3	A28	A62	COM4
COM3	A29	A63	COM4
COM3	A30	A64	COM4
COM3	A31	A65	COM4
COM3	A32	A66	COM4
COM3	A33	A67	COM4
N/C	A34	A68	N/C

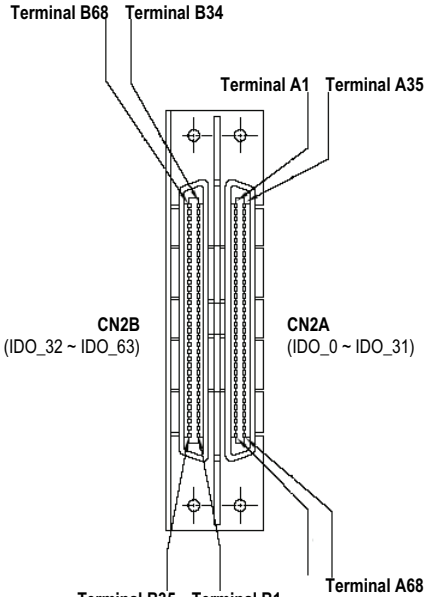
Figure 2-3: DI connector pin assignments

Legend

IDI_n	Isolated digital input channel n
COM1	Common junction for input channels 0 ~ 7
COM2	Common junction for input channels 8 ~ 5
COM3	Common junction for input channels 16 ~ 23
COM4	Common junction for input channels 24 ~ 31
COM5	Common junction for input channels 32 ~ 39
COM6	Common junction for input channels 40 ~ 47
COM7	Common junction for input channels 48 ~ 55
COM8	Common junction for input channels 56 ~ 63

2.4 DO (CN2) connector pin assignments

V5V	B68	B34	V5V
IGND	B67	B33	IGND
IGND	B66	B32	IGND
IGND	B65	B31	IGND
IGND	B64	B30	IGND
IGND	B63	B29	IGND
IGND	B62	B28	IGND
IGND	B61	B27	IGND
VDD8	B60	B26	VDD7
IDO_63	B59	B25	IDO_55
IDO_62	B58	B24	IDO_54
IDO_61	B57	B23	IDO_53
IDO_60	B56	B22	IDO_52
IDO_59	B55	B21	IDO_51
IDO_58	B54	B20	IDO_50
IDO_57	B53	B19	IDO_49
IDO_56	B52	B18	IDO_48
N/C	B51	B17	N/C
IGND	B50	B16	IGND
IGND	B49	B15	IGND
IGND	B48	B14	IGND
IGND	B47	B13	IGND
IGND	B46	B12	IGND
IGND	B45	B11	IGND
IGND	B44	B10	IGND
VDD6	B43	B9	VDD5
IDO_47	B42	B8	IDO_39
IDO_46	B41	B7	IDO_38
IDO_45	B40	B6	IDO_37
IDO_44	B39	B5	IDO_36
IDO_43	B38	B4	IDO_35
IDO_42	B37	B3	IDO_34
IDO_41	B36	B2	IDO_33
IDO_40	B35	B1	IDO_32



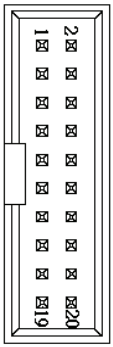
IDO_0	A1	A35	IDO_8
IDO_1	A2	A36	IDO_9
IDO_2	A3	A37	IDO_10
IDO_3	A4	A38	IDO_11
IDO_4	A5	A39	IDO_12
IDO_5	A6	A40	IDO_13
IDO_6	A7	A41	IDO_14
IDO_7	A8	A42	IDO_15
VDD1	A9	A43	VDD2
IGND	A10	A44	IGND
IGND	A11	A45	IGND
IGND	A12	A46	IGND
IGND	A13	A47	IGND
IGND	A14	A48	IGND
IGND	A15	A49	IGND
IGND	A16	A50	IGND
N/C	A17	A51	N/C
IDO_16	A18	A52	IDO_24
IDO_17	A19	A53	IDO_25
IDO_18	A20	A54	IDO_26
IDO_19	A21	A55	IDO_27
IDO_20	A22	A56	IDO_28
IDO_21	A23	A57	IDO_29
IDO_22	A24	A58	IDO_30
IDO_23	A25	A59	IDO_31
VDD3	A26	A60	VDD4
IGND	A27	A61	IGND
IGND	A28	A62	IGND
IGND	A29	A63	IGND
IGND	A30	A64	IGND
IGND	A31	A65	IGND
IGND	A32	A66	IGND
IGND	A33	A67	IGND
N/C	A34	A68	N/C

Figure 2-4: DO connector pin assignments

Legend

IDO_n Isolated digital output channel n
VDD1 common VDD junction for input channel 0-7
VDD2 common VDD junction for input channel 8-15
VDD3 common VDD junction for input channel 16-23
VDD4 common VDD junction for input channel 24-31
VDD5 common VDD junction for input channel 32-39
VDD6 common VDD junction for input channel 40-47
VDD7 common VDD junction for input channel 48-55
VDD8 common VDD junction for input channel 56-63
IGND Ground return path for isolated output channels
V5V Onboard un-regulated 5V power supply output
N/C No ConnectBoard ID (S1)

2.5 TTL I/O (JP3) connector pin assignments

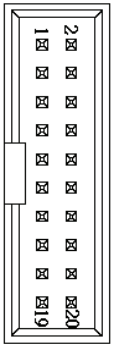


Pin	Function	Pin	Function
1	TTLIO_0	2	TTLIO_8
3	TTLIO_1	4	TTLIO_9
5	TTLIO_2	6	TTLIO_10
7	TTLIO_3	8	TTLIO_11
9	SGND	10	SGND
11	TTLIO_4	12	TTLIO_12
13	TTLIO_5	14	TTLIO_13
15	TTLIO_6	16	TTLIO_14
17	TTLIO_7	18	TTLIO_15
19	SGND	20	SGND

Table 2-1: TTL/IO connector (JP3) pin assignments

Legend
TTLIO_n TTL I/O channel n
SGND System ground

2.6 TTL I/O (JP4) connector pin assignments



Pin	Function	Pin	Function
1	TTLIO_16	2	TTLIO_24
3	TTLIO_17	4	TTLIO_25
5	TTLIO_18	6	TTLIO_26
7	TTLIO_19	8	TTLIO_27
9	SGND	10	SGND
11	TTLIO_20	12	TTLIO_28
13	TTLIO_21	14	TTLIO_29
15	TTLIO_22	16	TTLIO_30
17	TTLIO_23	18	TTLIO_31
19	SGND	20	SGND

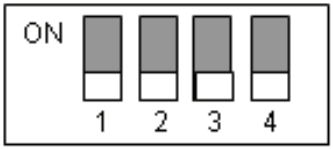
Table 2-2: TTL/IO (JP4) connector pin assignments

Legend
TTLIO_n TTL I/O channel n
SGND System ground

2.7 Board ID

When you plug two or more data acquisition cards in one system, it can take lots of efforts to identify one specific card. For easier identification, the PCI-7442 comes with a board ID function. According to a DIP switch configuration located in S1, you can assign a specific board ID to a designated card and access it correctly through simple software programming. For more details about board ID programming, refer to Chapter 5.

The table below shows all of the switch setting. **1** means DIP is at **ON** position; **0** means DIP is at **OFF**.



Note: 1 = ON, 0 = OFF
Default setting is **1111** or
Board ID = 0

Board ID	Switch No.			
	1	2	3	4
0	1	1	1	1
1	0	1	1	1
2	1	0	1	1
3	0	0	1	1
4	1	1	0	1
5	0	1	0	1
6	1	0	0	1
7	0	0	0	1
8	1	1	1	0
9	0	1	1	0
10	1	0	1	0
11	0	0	1	0
12	1	1	0	0
13	0	1	0	0
14	1	0	0	0
15	0	0	0	0

Table 2-3: Board ID settings

3 Operation theory

3.1 Isolated digital input

The PCI-7442 contains 64 opto-isolated digital input channels. The circuit diagram of the isolated input channel is shown in Figure 3-1.

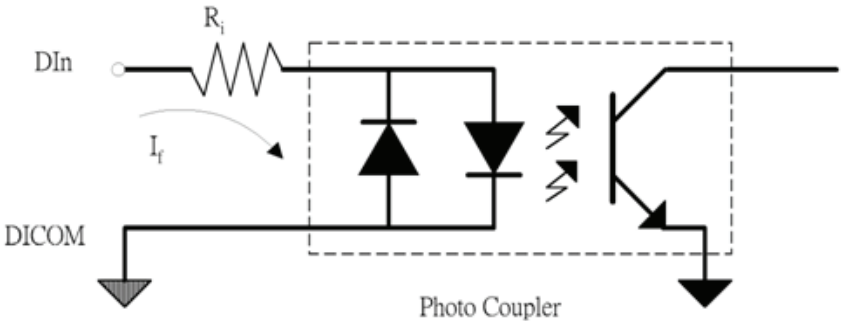


Figure 3-1: Photo coupler

The digital input is first routed through a photo-coupler (PC3H4), so that the connection are not polarly sensitive whether using positive or negative voltage.

The normal input voltage range for high state is from 5 V to 28 V. The PCI-7442 provides an isolated +5 V power (CN2B V5V) for dry contact input. When the external circuit has no voltage source (e.g. a switch), you may use the onboard +5V to respond the change of external circuit. The maximum output current of the onboard isolated power is 200 mA (at 40°C). Pay attention to the current consumption of the external circuit which should not exceed the limitation. The dry contact architecture is shown in Figure 3-2.

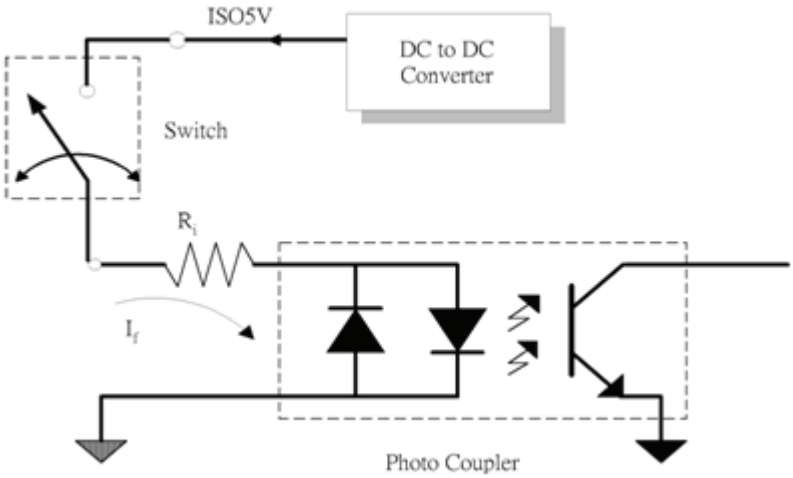


Figure 3-2: Dry contact

3.2 Change of State (COS) interrupt

Overview

The COS (Change of State) means either the input state (logic level) changes from low to high, or from high to low. The COS detection circuit will detect the edge of level change. In the PCI-7442 card, the COS detection circuit is applied to all the input channels. When any channel changes its logic level, the COS detection circuit generates an interrupt request to PCI controller.

COS detection

Figure 3-3 is an example of an 8-CH COS operation. All of the enabled DI channels' signal level change will be detected to generate the interrupt request.

While the interrupt request generates, the corresponding DI data will also be latched into the COS latch register. In our COS architecture, the DI data are sampled by a 33 MHz clock. It means the

pulse width of the digital input have to last longer than 31ns, or the COS latch register won't latch the correct input data. The COS latch register will be erased after clearing the interrupt request.

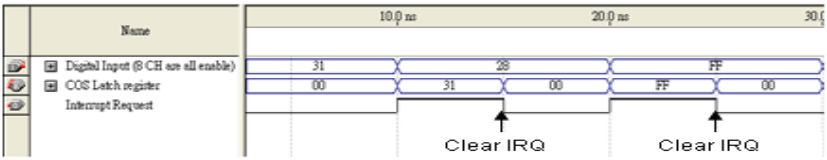


Figure 3-3: COS timing

COS detection architecture

The COS interrupt system is used in PCI-7442. COS interrupt occurs when the any of enabled DI line sense the status changes either from HIGH to LOW or from LOW to HIGH. The COS interrupt system can generate an interrupt request signal and the software can service this request with ISR. Note that there are two banks: bank 0 from DI0 to DI31 and bank 1 from DI32 to 63. These banks are cascaded together toward the same IRQ line via CPLD. Users can use commands to know which bank or which DI line has COS if it happens. Also, you can use commands to disable or enable the COS function of certain DI lines. The COS function for each is disabled by default. Refer to Figure 3-4 to know the architecture of COS detection.

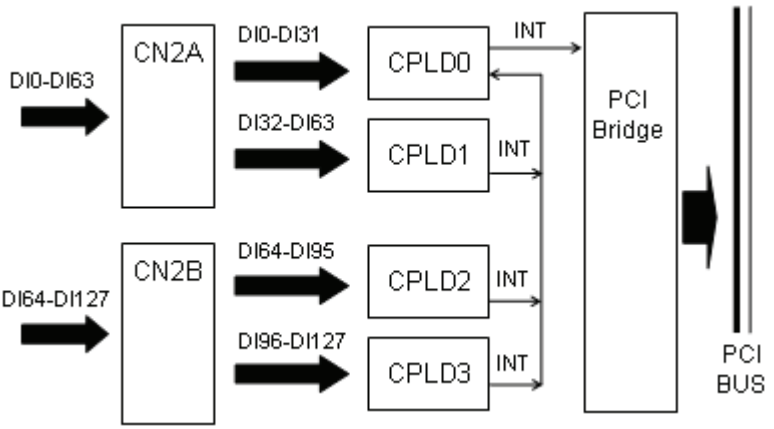


Figure 3-4: COS detection architecture

3.3 Programmable TTL Input/Output

The PCI-7442 card provides a 32-CH programmable TTL input/output. These channels are divided between two connectors: JP3 and JP4. You can change the direction of each TTL channel any time. The I/O voltage level suits with 5 V TTL level and 3.3 V TTL level. But the driving strength of each channel is 4 mA. Pay particular attention to the current consumption of the TTL channel.

3.4 Isolated digital output channels

The common ground connection of isolated digital output is shown in the figure below. When the isolated digital output goes **ON**, the sink current will be conducted through the power MOSFETs. When the isolated digital output goes **OFF**, no current is conducted to flow through the power MOSFETS. Take note that when the load is of an **inductance nature** such as a relay, coil or motor, the VDD pin must be connected to an external power source. The extra connection is utilized for the **fly-wheel diode** to form a current-release closed loop, so that the MOSFETs are protected from any high reverse voltage which can be generated by the inductance load when the output is switched from ON to OFF. In addition, you can read back the 64-CH IDO statuses to check if the statuses meet your purpose.

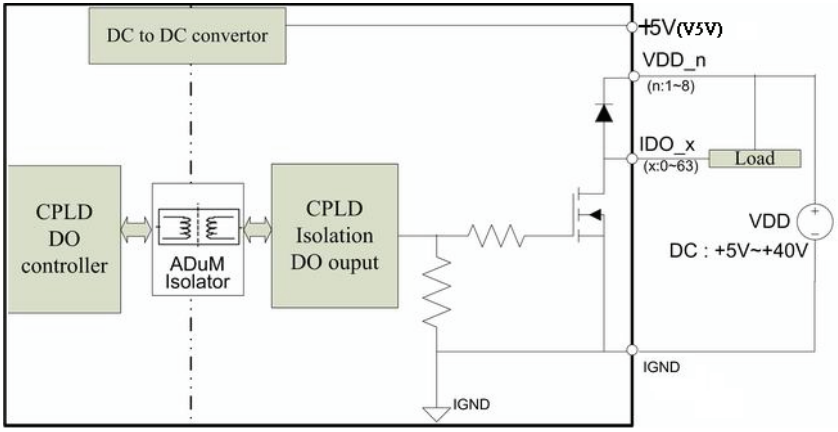


Figure 3-5: Common ground connection of isolated digital output

The PCI-7442 provides three special functions for safety measures. First, the PCI-7442 could automatically configure the 64-CH DO initial statuses when powering up. Second, you can direct the PCI-7442 to hold the DO statuses and avoid its power-up initial configuration state after a hot system reset. Third, you can direct the PCI-7442 to automatically configure the 64-CH DO safety statuses when a WDT interruption asserts.

3.5 Watch dog timer (WDT)

In safety-critical applications, you can enable the watch dog timer (WDT) function to automatically generate an interrupt signal, in case the operating system or the PCI-7442 card crashes. To access this function, you must first configure the watchdog timer overflow counter by windows API. Generally, the trigger source would come from the onboard 32-bit watchdog timer.

The WDT overflow interval can be programmed through API. You must reload the WDT counter value before enabling the WDT. After enabling the watchdog timer, you must periodically reload the timer value by software command. If the timer is not being reloaded within the specified interval, the WDT module generates an overflow interruption signal. When you enable the SafetyOut_Enable bit, the PCI-7442 would automatically configure the 64-CH DO safety statuses. This WDT function is disabled by default.

The WDT function operation flow is listed below.

1. Load the WDT counter value through the function **_7442_Bank2_WDT_Load()**, this function allows users to set WDT overflow time from 100 ns to 429.496 s.
2. Enable the WDT counter to count down through the function **_7442_WDTINT_Control()**, this function allows WDT function in the CPLD starting to count down. After enabling the function, you must reload WDT by step (1) before it overflows so that the card will work normally under WDT monitoring. When it overflows, a CPLD interrupt occurs.

When a WDT interrupt occurs, you must reload the WDT through **function_7442_Bank2_WDT_Load()** to clear the WDT carry out, and then clear the system interrupt through **function _7442_CLR_IRQ()**. After this, the WDT counts again, unless you disable the WDT counter through **function _7442_WDTINT_Control()**.

3.6 Interrupt architecture

The PCI-7442 has a powerful dual interrupt routing scheme including Change of State detection and interrupt sources on watch dog timer. these interrupts well can make you handle more complicated information from outside environment and release your computer from a heavy burden in dealing with digital input data. Note that the dual interrupts do not mean the card occupies two IRQ levels.

4 Register format

This chapter provides the detailed descriptions of the register formats intended for programmers who want to operate the PCI-7442 through low-level programming. This chapter is intended for users that have basic understanding of the PCI interface.

4.1 I/O address map

The PCI-7442 registers are all 16-bit wide and can only be accessed using 16-bit I/O instructions. The isolated digital input/output control is by accessing registers mentioned in this chapter. Table 4-1 outlines the register map, including descriptions and offset addresses corresponding to the base address

Offset	Write	Read
0X00h	-	-
0X02h	-	Bank0 Isolated DI A
0X04h	-	Bank0 Isolated DI B
0X06h	Bank0 COS Interrupt Control	Interrupt Status/Bank0 COS INT Control Read Back
0X08h	Bank0 COS Setup A	Bank0 COS Latch A
0X0Ah	Bank0 COS Setup B	Bank0 COS Latch B
0X0Ch	Bank0 TTL IO Setup	Bank0 TTL IO Status Read Back
0X0Eh	Bank0 TTL IO Digital Output	Bank0 TTL IO Digital Input
0X40h	-	-
0X42h	-	Bank1 Isolated DI A
0X44h	-	Bank1 Isolated DI B
0X46h	Bank1 COS Interrupt Control	Bank1 COS Interrupt Control Read Back
0X48h	Bank1 COS Setup A	Bank1 COS Latch A
0X4Ah	Bank1 COS Setup B	Bank1 COS Latch B
0X4Ch	Bank1 TTL IO Setup	Bank1 TTL IO Status Read Back
0X4Eh	Bank1 TTL IO Digital Output	Bank1 TTL IO Digital Input
0X80h	Isolated DO A	Bank2 Isolated DO Read back Start
0X82h	Isolated DO B	Isolated DO Read Back A
0X84h	Isolated DO C	Isolated DO Read back B
0X86h	Isolated DO D	Isolated DO Read back C
0X88h	Bank2 Isolated DO Send Out	Isolated DO Read back D
0X8Ah	WDT Interrupt Control/	WDT Interrupt Control Read Back/
0X8Ch	Power-Up Initial DO Status Setup A	Power-Up Initial DO Status Read Back Start
0X8Eh	Power-Up Initial DO Status Setup B	Power-Up Initial DO Status Read Back A
0X90h	Power-Up Initial DO Status Setup C	Power-Up Initial DO Status Read Back B
0X92h	Power-Up Initial DO Status Setup D	Power-Up Initial DO Status Read Back C
0X94h	WDT Load Config A	Power-Up Initial DO Status Read Back D
0X96h	WDT Load Config B	WDT Overflow DO Safety Status Read Back Start
0X98h	WDT Overflow DO Safety Status Setup A	WDT Overflow DO Safety Status Read Back A
0X9Ah	WDT Overflow DO Safety Status Setup B	WDT Overflow DO Safety Status Read Back B
0X9Ch	WDT Overflow DO Safety Status Setup C	WDT Overflow DO Safety Status Read Back C
0X9Eh	WDT Overflow DO Safety Status Setup D	WDT Overflow DO Safety Status Read Back D

Table 4-1: I/O registers map

4.2 Bankn (n=0~1) isolated digital input register A, B

There are 64 isolated inputs on a PCI-7442 card. The statuses of the 64 lines can be read from the four isolated input registers. Each bit corresponds to each channel. The bit value 0 means that the input is ON and 1 means that the input is OFF.

- ▷ Bank0 Isolated Digital Input Register A
- ▷ Address: BASE + 0x02h
- ▷ Attribute: Read

7	6	5	4	3	2	1	0
IDI_COS_Latch_Data [7..0]							
15	14	13	12	11	10	9	8
IDI_COS_Latch_Data [15..8]							

- ▷ Bank0 Isolated Digital Input Register B
- ▷ Address: BASE + 0x04h
- ▷ Attribute: Read

7	6	5	4	3	2	1	0
IDI_COS_Latch_Data [23..16]							
15	14	13	12	11	10	9	8
IDI_COS_Latch_Data [31..24]							

- ▷ Bank1 Isolated Digital Input Register A
- ▷ Address: BASE + 0x42h
- ▷ Attribute: Read

7	6	5	4	3	2	1	0
IDI_COS_Latch_Data [39..32]							
15	14	13	12	11	10	9	8
IDI_COS_Latch_Data [47..40]							

- ▷ Bank1 Isolated Digital Input Register B
- ▷ Address: BASE + 0x44h
- ▷ Attribute: Read

7	6	5	4	3	2	1	0
IDI_COS_Latch_Data [55..48]							
15	14	13	12	11	10	9	8
IDI_COS_Latch_Data [63..56]							

- ▶ IDI[n]: isolated digital input channel n, n = 0 ~ 63
- ▶ 0: The input is ON
- ▶ 1: The input is OFF
- ▶ The initial value is 1

4.3 Bank0, Bank1 COS interrupt control registers

There are two different interrupt modes in PCI-7442. In default, both interrupt modes are disabled. You can write registers mentioned later to enable them. In the first mode, you must enable the COS (Change of State) interrupt function to monitor the statuses of enabled input channels and whenever the statuses change from 0 to 1 or 1 to 0. In another mode, you can enable watch dog timer (WDT) counter and let it count down. The interrupt asserts when the WDT counter counts to zero.

After processing the interrupt request event, you have to clear the interrupt request in order to handle another interrupt request. Noteworthy is the fact that it takes time for a system to clear the interrupt. That is, any COS interrupt or WDT interrupt comes before the previous interrupt is still not cleared is neglected. To clear the interrupt request, write 1 to the corresponding bit. The WDT INT control registers are discussed later on this chapter.

The COS interrupt is enabled by two registers (Bank0 COS Interrupt Control Register and Bank1 COS Interrupt Control Register). Because the 64 digital inputs are divided among two 32-bit on board buses, every 32 inputs is connected to a CPLD. When you enable Bank0 COS interrupt, the first CPLD (CPLD0) produces interrupt signal while the first 32-bit inputs IDI[31..0] have change of state. When you enable Bank1 COS interrupt, the second CPLD (CPLD1) will produce interrupt signal while the second 32-bit inputs IDI[63..32] have change of state.

- ▷ Bank0 COS Interrupt Control Register
- ▷ Address: BASE + 0x06h
- ▷ Attribute: Write

7	6	5	4	3	2	1	0
---							COS_CLR0
15	14	13	12	11	10	9	8
---							COS0_Enable

COS_CLR0 (bit 0): Write 1 to clear Bank0 COS interrupt

- ▶ 1: clear the Bank0 COS interrupt
- ▶ 0: no effects (default)

COS0_Enable (bit 8): Bank0 IDI[31..0] COS interrupt control

- 1: enable
- 0: disable (default)

  - Bank1 COS Interrupt Control Register
  - Address: BASE + 0x46h
  - Attribute: Write

7	6	5	4	3	2	1	0
---							COS_CLR1
15	14	13	12	11	10	9	8
---							COS1_Enable

COS_CLR0 (bit 0): Write 1 to clear Bank1 COS interrupt

- 1: clear the Bank0 COS interrupt
- 0: no effects (default)

COS0_Enable (bit 8): Bank1 IDI[31..0] COS interrupt control

- 1: enable
- 0: disable (default)

4.4 Interrupt status Bank0, Bank1 COS INT control read back

When any Bank0 - Bank1 COS interrupts occur, these registers provide information for you to recognize the interrupt status and the interrupt setup condition read back.

- ▷ Interrupt Status/Bank0 COS INT Control Read Back Register
- ▷ Address: BASE + 0x06h
- ▷ Attribute: Read

7	6	5	4	3	2	1	0
---						COS1_INT_Status	COS0_INT_Status
15	14	13	12	11	10	9	8
COS0_EN_Status	---						

COS0_INT_Status (bit 0): Bank0 DI[31..0] COS interrupt status register1: Bank0 COS interrupt asserts

- ▶ 0: Bank0 COS interrupt de-asserts (default)

COS1_INT_Status (bit 1): Bank1 DI[63..32] COS interrupt status register

- ▶ 1: Bank1 COS interrupt asserts
- ▶ 0: Bank1 COS interrupt de-asserts (default)

COS0_EN_Status (bit 15): Bank0 DI[31..0] COS interrupt enable status

- ▶ 1: Bank0 COS interrupt enabled
- ▶ 0: Bank0 COS interrupt disabled (default)

- ▷ Bank1 COS INT Control Read Back Register
- ▷ Address: BASE + 0x46h
- ▷ Attribute: Read

7	6	5	4	3	2	1	0

15	14	13	12	11	10	9	8
COS1_EN_Status	---						

COS1_EN_Status (bit 15): Bank1 DI[63..32] COS interrupt enable status

- ▶ 1: Bank1 COS interrupt enabled
- ▶ 0: Bank1 COS interrupt disabled (default)

4.5 Bankn (n = 0~1) COS setup register A, B

The PCI-7442 provides a Change of State (COS) interrupt function on any one of digital input channel. This function allows you to monitor the status of input channels. Because these digital input channels are divided among two 32-bits banks you can monitor the digital input channels by setting two Bank COS Setup registers.

By enabling the COS Setup registers, it will generate an interrupt when the corresponding channel changes its state. For detailed information, refer to Section 3.2.

- ▷ Bank0 COS Setup Register A
- ▷ Address: BASE + 0x08h
- ▷ Attribute: Write

7	6	5	4	3	2	1	0
IDI_COS_EN [7..0]							
15	14	13	12	11	10	9	8
IDI_COS_EN [15..8]							

- ▷ Bank0 COS Setup Register B
- ▷ Address: BASE + 0x0Ah
- ▷ Attribute: Write

7	6	5	4	3	2	1	0
IDI_COS_EN [23..16]							
15	14	13	12	11	10	9	8
IDI_COS_EN [31..24]							

- ▷ Bank1 COS Setup Register A
- ▷ Address: BASE + 0x48h
- ▷ Attribute: Write

7	6	5	4	3	2	1	0
IDI_COS_EN [39..32]							
15	14	13	12	11	10	9	8
IDI_COS_EN [47..40]							

- ▷ Bank1 COS Setup Register B
- ▷ Address: BASE + 0x4Ah
- ▷ Attribute: Write

7	6	5	4	3	2	1	0
IDI_COS_EN [55..48]							
15	14	13	12	11	10	9	8
IDI_COS_EN [63..56]							

IDI_COS_EN [n]: Change of State function Enable of IDI channel n, n = 0 ~ 63

- ► 0: disable monitoring channel n COS interrupt (default)
- ► 1: enable monitoring channel n COS interrupt

4.6 Bankn (n=0~1) COS latch register A, B

When COS occurs, the Bank0 - Bank1 COS latch registers will also latch the DI[31..0], DI[63..32] data respectively. Once you clear the interrupt request, the COS latch register automatically clears. Since you can simply read these registers to know the statuses after interrupts, these registers free the CPU from the overwhelming task of constantly polling all inputs, enabling it to handle other tasks.

- ▷ Bank0 COS Latch Register A
- ▷ Address: BASE + 0x08h
- ▷ Attribute: Read

7	6	5	4	3	2	1	0
IDI_COS_Latch_Data [7..0]							
15	14	13	12	11	10	9	8
IDI_COS_Latch_Data [15..8]							

- ▷ Bank0 COS Latch Register B
- ▷ Address: BASE + 0x0Ah
- ▷ Attribute: Read

7	6	5	4	3	2	1	0
IDI_COS_Latch_Data [23..16]							
15	14	13	12	11	10	9	8
IDI_COS_Latch_Data [31..24]							

- ▷ Bank1 COS Latch Register A
- ▷ Address: BASE + 0x48h
- ▷ Attribute: Read

7	6	5	4	3	2	1	0
IIDI_COS_Latch_Data [39..32]							
15	14	13	12	11	10	9	8
IDI_COS_Latch_Data [47..40]							

- ▷ Bank1 COS Latch Register B
- ▷ Address: BASE + 0x4Ah
- ▷ Attribute: Read

7	6	5	4	3	2	1	0
IDI_COS_Latch_Data [55..48]							
15	14	13	12	11	10	9	8
IDI_COS_Latch_Data [63..56]							

IDI_COS_Latch_Data [n]: COS latch data register of DI channel n, n = 0 - 63

- ▶ 0: The input is ON
- ▶ 1: The input is OFF
- ▶ The initial value is 1

4.7 Bankn (n=0~1) TTL IO setup register

The PCI-7442 provides an extra 32-channel TTL I/O function for optional applications. These TTL I/O channels are divided among two 16-bits banks. These channels are divided between two connectors, JP3, JP4. Users can choose the direction of each TTL channel any time by set up the two bank TTL IO setup register.

- ▷ Bank0 TTL IO Setup Register
- ▷ Address: BASE + 0x0ch
- ▷ Attribute: Write

7	6	5	4	3	2	1	0
TTLIO_SETUP [7..0]							
15	14	13	12	11	10	9	8
TTLIO_SETUP [15..8]							

- ▷ Bank1 TTL IO Setup Register
- ▷ Address: BASE + 0x4ch
- ▷ Attribute: Write

7	6	5	4	3	2	1	0
TTLIO_SETUP [23..16]							
15	14	13	12	11	10	9	8
TTLIO_SETUP [31..24]							

TTLIO_SETUP[n]: Setup TTL I/O Channel n Direction, n=0~31

- ▶ 0: setup the TTL I/O channel direction is Input (default)
- ▶ 1: setup the TTL I/O channel direction is Output

4.8 Bankn (n=0~1) TTL IO status read back register

When you set up the direction of TTL I/O channels, the statuses of setting can be read back through TTL IO Status Read Back Register in each back. You can read back the I/O direction statuses to check if the directions meet your need.

- ▷ Bank0 TTL IO Status Read Back Register
- ▷ Address: BASE + 0x0ch
- ▷ Attribute: Read

7	6	5	4	3	2	1	0
TTLIO_Status_RB[7..0]							
15	14	13	12	11	10	9	8
TTLIO_Status_RB[15..8]							

- ▷ Bank1 TTL IO Status Read Back Register
- ▷ Address: BASE + 0x4ch
- ▷ Attribute: Read

7	6	5	4	3	2	1	0
TTLIO_Status_RB[23..16]							
15	14	13	12	11	10	9	8
TTLIO_Status_RB[31..24]							

TTLIO_ Status_RB [n]: Read back the TTL I/O Channel n Direction status, n =0~31

- ▶ 0: the TTL I/O channel direction status is Input (default)
- ▶ 1: setup the TTL I/O channel direction status is Output

4.9 Bankn (n=0~1) TTL IO digital output register

The PCI-7442 provides an extra 32-channel TTL I/O function for optional applications. These TTL I/O channels are divided among two 16-bits banks. When the I/O direction setting is output, users can send out data through the TTL I/O output channel. **1** means that the corresponding TTL channel output is high. The setting **0** means that it is low.

- ▷ Bank0 TTL IO Digital Output Register
- ▷ Address: BASE + 0x0Eh
- ▷ Attribute: Write

7	6	5	4	3	2	1	0
TTLIO_DO[7..0]							
15	14	13	12	11	10	9	8
TTLIO_DO[15..8]							

- ▷ Bank1 TTL IO Digital Output Register
- ▷ Address: BASE + 0x4Eh
- ▷ Attribute: Write

7	6	5	4	3	2	1	0
TTLIO_DO[23..16]							
15	14	13	12	11	10	9	8
TTLIO_DO[31..24]							

TTLIO_DO[n]: Output Data of TTL I/O Channel n, n =0~31

- ▶ 0: Output Data of TTL I/O is low (default).
- ▶ 1: Output Data of TTL I/O is high.

4.10 Bankn (n=0~1) TTL IO digital input register

The PCI-7442 provides an additional 32-channel TTL I/O function for optional applications. These TTL I/O channels are divided among two 16-bits banks. When the I/O direction setting is input, users can read data through the TTL I/O input channel. **1** means that the corresponding TTL channel input data is high. The setting **0** means that it is low.

- ▷ Bank0 TTL IO Digital Input Register
- ▷ Address: BASE + 0x0Eh
- ▷ Attribute: Read

7	6	5	4	3	2	1	0
TTLIO_DI[7..0]							
15	14	13	12	11	10	9	8
TTLIO_DI[15..8]							

- ▷ Bank1 TTL IO Digital Input Register
- ▷ Address: BASE + 0x4Eh
- ▷ Attribute: Read

7	6	5	4	3	2	1	0
TTLIO_DI[23..16]							
15	14	13	12	11	10	9	8
TTLIO_DI[31..24]							

TTLIO_DO[n]: Output Data of TTL I/O Channel n, n =0~31

- ► 0: Input Data of TTL I/O is low (default).
- ► 1: Input Data of TTL I/O is high.

4.11 Bank2 Isolated Digital Output Register A~D and Bank2 Isolated DO Send Out Start

There are 64 isolated digital outputs on each PCI-7442 board. These lines are divided between two output connectors, CN2A and CN2B. They are controlled by four 16-bit registers in bank2. Each digital output line is controlled by each bit of the four control registers. You must send out the corresponding DO output data, then send out the start command to the bank2, finally. Then the 64-bit DO data will be sent out at the same time. The output device type is Open Drain Power MOSFET driver. The setting **1** means that the corresponding PowerMOS is **ON**. The setting **0** means that the PowerMOS is **OFF**.

- ▷ Bank2 Isolated Digital Output Register A
- ▷ Address: BASE + 0x80h
- ▷ Attribute: Write

7	6	5	4	3	2	1	0
IDO[7..0]							
15	14	13	12	11	10	9	8
IDO[15..8]							

- ▷ Bank2 Isolated Digital Output Register B
- ▷ Address: BASE + 0x82h
- ▷ Attribute: Write

7	6	5	4	3	2	1	0
IDO[23..16]							
15	14	13	12	11	10	9	8
IDO[31..24]							

- ▷ Bank2 Isolated Digital Output Register C
- ▷ Address: BASE + 0x84h
- ▷ Attribute: Write

7	6	5	4	3	2	1	0
IDO[39..32]							
15	14	13	12	11	10	9	8
IDO[47..40]							

- ▷ Bank2 Isolated Digital Output Register D
- ▷ Address: BASE + 0x86h
- ▷ Attribute: Write

7	6	5	4	3	2	1	0
IDO[55..48]							
15	14	13	12	11	10	9	8
IDO[63..56]							

IDO[n]: Isolated digital output channel n, n = 0 - 63

- ▶ 0: the corresponding PowerMOS is Turn OFF
- ▶ 1: the corresponding PowerMOS is Turn ON
- ▶ The initial value is 0

Bank2 Isolated DO Send Out Start does not need any register value. Users only need to send out the address (BASE + 0x88h) in **Write** mode after setting up all 64-bit channel output data. When the bank2 receives the **Start** command, the 64-bit DO data will be sent out at the same time. Users can check if the DO send procedure is finished by getting nDO_SendReady flag status. See Section 3.4 for details

- ▷ Bank2 Isolated DO Send Out Start
- ▷ Address: BASE + 0x88h
- ▷ Attribute: Write

7	6	5	4	3	2	1	0
--							
15	14	13	12	11	10	9	8
--							

4.12 Bank2 Isolated DO Read Back Start and Bank2 Isolated DO Read Back Register A~D

The isolated DO statuses can be read back from the Bank2 DO read back register. When users want to read back the 64-bit IDO statuses, users must send the Read Back Start command to the bank2 first. Then, you can read back isolated DO Read Back Registers A~D if DO read back procedure is standby. If the output PowerMOS is ON, the corresponding bit value is **1**. If the output PowerMOS is OFF, the corresponding bit value is **0**.

- ▷ Bank2 Isolated DO Read Back Start
- ▷ Address: BASE + 0x80h
- ▷ Attribute: Read

7	6	5	4	3	2	1	0
--							
15	14	13	12	11	10	9	8
--							

Bank2 Isolated DO Read Back Start doesn't need any register value. Users only need to send out the address (BASE + 0x80h) in **Read** mode before reading back all 64-bit channel output data. When the bank2 receives the Start command, the 64-bit DO data read back procedure will start to proceed. Users can check if the DO Read Back procedure is finished by get nDO_RBReady flag status. Refer to Section 3.4 for more information.

- ▷ Bank2 Isolated DO Read Back Register A
- ▷ Address: BASE + 0x82h
- ▷ Attribute: Read

7	6	5	4	3	2	1	0
IDO[7..0]							
15	14	13	12	11	10	9	8
IDO[15..8]							

▷ Bank2 Isolated DO Read Back Register B

▷ Address: BASE + 0x84h

▷ Attribute: Read

7	6	5	4	3	2	1	0
IDO[23..16]							
15	14	13	12	11	10	9	8
IDO[31..24]							

▷ Bank2 Isolated DO Read Back Register C

▷ Address: BASE + 0x86h

▷ Attribute: Read

7	6	5	4	3	2	1	0
IDO[39..32]							
15	14	13	12	11	10	9	8
IDO[47..40]							

▷ Bank2 Isolated DO Read Back Register D

▷ Address: BASE + 0x88h

▷ Attribute: Read

7	6	5	4	3	2	1	0
IDO[55..48]							
15	14	13	12	11	10	9	8
IDO[63..56]							

IDO[n]: Isolated digital output channel n, n = 0 - 63

- 0: the corresponding PowerMOS is **OFF**
- 1: the corresponding PowerMOS is **ON**
- The initial value is 0

4.13 Bank2 WDT INT Control/Hot-Reset Hold Control Register

There are two different interrupt modes in PCI-7442. In default, both interrupt modes are disabled. The first mode is COS INT function. Refer to Section 3.2 for details. In the second mode, you can enable the watch dog timer (WDT) counter and let it count down. The interrupt asserts when the watch dog timer counter counts to zero. You can control WDT enable and clear WDT INT by setting two bits (WDT_Enable, WDT_INT_CLR) in Bank2 WDT INT Control/Hot-Reset Hold Control Register.

The PCI-7442 provides some special function for safety industrial applications. When WDT interrupt asserts, users can choose if system send out Safety DO value to prevent some unknown damage by setting the **SafetyOut_Enable** bit. Besides, when the system goes through a unexpected or normal hot system reset without turning off the system power, you can choose whether to allow the PCI-7442 board to retain the original DO values before the system hot reset, or allow the PCI-7442 board to enter the power-up initial procedure to send out the default initial DO values which you configured. Refer to Section 3.4 for details. By setting the **Hot_Reset_Hold_Enable** bit, users can enable **Hot_Reset_Hold** function anytime. This function is useful for some unstable environment.

- ▷ Bank2 WDT INT Control/Hot-Reset Hold Control Register
- ▷ Address: BASE + 0x8Ah
- ▷ Attribute: Write

7	6	5	4	3	2	1	0
---				SafetyOut_ Enable	WDT_INT_CLR	WDT_Enable	Hot_Reset_ Hold_Enable
15	14	13	12	11	10	9	8
--							

Hot_Reset_Hold_Enable (bit 0): Enable Hot-System-Reset DO Hold function

- ▶ 0: disable function (default)
- ▶ 1: enable Hot-System Reset DO Hold function

WDT_Enable (bit 1): Bank2 WDT interrupt Enable control

- 0: disable WDT (default)
- 1: enable WDT

WDT_INT_CLR(bit2): write 1 to clear WDT interrupt

- 0: no effect (default)
- 1: clear WDT interrupt

SafetyOut_Enable(bit3): Write 1 to Enable Send Out Safety DO status function while WDT INT asserts

- 0: Disable function, DO status will be hold, not change while WDT INT asserts (default)
- 1: Enable Send Out Safety DO status function while WDT INT asserts

4.14 Bank2 WDT INT control RB/ Hot-reset hold control RB register

When the WDT interrupt asserts, this register provides information to recognize the WDT interrupt status and the WDT setup condition read back. You may also get information about the hot-reset-hold function enable setting from the DO-safety-value sendout function enable setting, the DO send-out, RB action ready flags, and flash R/W action ready flag.

- ▷ Bank2 WDT INT Control RB / Hot-Reset Hold Control RB Register
- ▷ Address: BASE + 0x8Ah
- ▷ Attribute: Read

7	6	5	4	3	2	1	0
---	nAction_ Ready	nDO_Se ndReady	nDO_RB Ready	SafetyOut_ Enable_Status	WDT_INT_ Status	WDT_Enable _Status	Hot_Reset_Hold _Enable_Status
15	14	13	12	11	10	9	8

Hot_Reset_Hold_Enable_Status (bit 0): Read back Hot-System Reset DO Hold function Enable Status

- ▶ 0: function is disable (default)
- ▶ 1: The Hot-System Reset DO Hold function is enabled

WDT_Enable_Status (bit1): Read back Bank2 WDT interrupt Enable Status

- ▶ 0: The WDT interrupt is disable (default)
- ▶ 1: The WDT interrupt is enable

WDT_INT_Status (bit2): Read Back the WDT interrupt Status if the interrupt asserts

- ▶ 0: The WDT interrupt doesn't assert (default)
- ▶ 1: The WDT interrupt asserts

SafetyOut_Enable_Status (bit3): Read back the Send Out Safety DO status function Enable Status

- ▶ 0: function is disable (default)
- ▶ 1: the Send Out Safety DO status function is Enable

nDO_RBReady (bit4): detect flag for users to know if DO RB data could be read

- 0: Ready for users to read (default)
- 1: not ready, please hold

nDO_SendReady (bit5): detect flag for users to know if DO Data Sending is finished

- 0: finished, ready for users to access next step
- 1: not finished, please hold

nAction_Ready (bit6): detect flag for users to know if Flash Data W/R is finished

- 0: finished, ready for users to access next step (default)
- 1: not finished, please hold

4.15 Bank2 Power-up DO Setup Register A~D

When the system enters the power up status, PCI-7442 can enter the initial procedure which sends out the default initial value to 64-CH digital outputs. You can configure the power-up default DO values and store them in the flash. Thus, the DO would not enter an unknown status when the system power is turned on.

You can program the 64-CH power-up default DO values through accessing the Bank2 Power-up DO Setup Register A~D. After accessing Bank2 Power-up DO Setup Register D, it could take 0.5s to finish writing flash procedure. Users can check if the procedure is finish or not by nAction_Ready flag. Refer to Section 3.4 for details.

- ▷ Bank2 Power-up DO Setup Register A
- ▷ Address: BASE + 0x8Ch
- ▷ Attribute: Write

7	6	5	4	3	2	1	0
IDO[7..0]							
15	14	13	12	11	10	9	8
IDO[15..8]							

- ▷ Bank2 Power-up DO Setup Register B
- ▷ Address: BASE + 0x8Eh
- ▷ Attribute: Write

7	6	5	4	3	2	1	0
IDO[23..16]							
15	14	13	12	11	10	9	8
IDO[31..24]							

- Bank2 Power-up DO Setup Register C
- Address: BASE + 0x90h
- Attribute: Write

7	6	5	4	3	2	1	0
IDO[39..32]							
15	14	13	12	11	10	9	8
IDO[47..40]							

- Bank2 Power-up DO Setup Register D
- Address: BASE + 0x92h
- Attribute: Write

7	6	5	4	3	2	1	0
IDO[55..48]							
15	14	13	12	11	10	9	8
IDO[63..56]							

IDO[n]: Isolated digital output channel n, n = 0 - 63

- 0: the corresponding PowerMOS is **OFF**
- 1: the corresponding PowerMOS is **ON**
- The initial value is 0

4.16 Bank2 Power-up DO Read Back Start and Bank2 Power-up DO Read Back Register A~D

You can read back the configured power-up initial DO values that are stored in the onboard flash. First, you have to send out the Power-up DO Read Back Start command to the bank2. Then, the flash read procedure starts in about 50 ms. The finished flag can be checked by nAction_Ready flag. Refer to Section 3.4 for details. After the Read Back procedure, you can read back the 64-bit Bank2 Power-up DO Read Back Register A~D.

- ▷ Bank2 Power-up DO Read Back Start
- ▷ Address: BASE + 0x8Ch
- ▷ Attribute: Read

7	6	5	4	3	2	1	0
--							
15	14	13	12	11	10	9	8
--							

Bank2 Power-up DO Read Back Start does not need any register value. You only need to send out the address (BASE + 0x8Ch) in **Read** mode before reading back all 64-bit channel output initial data. When the bank2 receives the **Start** command, the flash read procedure starts after about 50 ms. Users can check if the procedure is finished by getting nAction_Ready flag status. Refer to Section 3.4 for details.

- ▷ Bank2 Power-up DO Read Back Register A
- ▷ Address: BASE + 0x8Eh
- ▷ Attribute: Read

7	6	5	4	3	2	1	0
IDO[7..0]							
15	14	13	12	11	10	9	8
IDO[15..8]							

- ▷ Bank2 Power-up DO Read Back Register B
- ▷ Address: BASE + 0x90h
- ▷ Attribute: Read

7	6	5	4	3	2	1	0
IDO[23..16]							
15	14	13	12	11	10	9	8
IDO[31..24]							

- ▷ Bank2 Power-up DO Setup RegisterC
- ▷ Address: BASE + 092h
- ▷ Attribute: Read

7	6	5	4	3	2	1	0
IDO[39..32]							
15	14	13	12	11	10	9	8
IDO[47..40]							

- ▷ Bank2 Power-up DO Setup Register D
- ▷ Address: BASE + 0x94h
- ▷ Attribute: Read

7	6	5	4	3	2	1	0
IDO[55..48]							
15	14	13	12	11	10	9	8
IDO[63..56]							

IDO[n]: Isolated digital output channel n, n = 0 - 63

- ▶ 0: the corresponding PowerMOS is OFF
- ▶ 1: the corresponding PowerMOS is ON
- ▶ The initial value is 0

4.17 Bank2 WDTimer Load Config Register A~B

The PCI-7442 provides an onboard, 32-bit counter width watch dog timer (WDT) with 10 MHz clock. The WDT counter loads the 32-bit value written in the two 16-bit WDT Load Config Register A~B. The corresponding hexadecimal value you set determines the overflow time of WDT counter. The overflow time is calculated by the value that you set multiplied 100 ns. The timer interval is from 0 to 429.496 seconds.

- ▷ Bank2 WDT Load Config Register A
- ▷ Address: BASE + 0x94h
- ▷ Attribute: Write

7	6	5	4	3	2	1	0
WDT_Load [7..0]							
15	14	13	12	11	10	9	8
WDT_Load [15..8]							

- ▷ Bank2 WDT Load Config Register B
- ▷ Address: BASE + 0x96h
- ▷ Attribute: Write

7	6	5	4	3	2	1	0
WDT_Load [23..16]							
15	14	13	12	11	10	9	8
WDT_Load [31..24]							

WDT_Load[n]: Counter value determined by each binary bit n, n = 0 - 31 For example, if you write 1 to each bit, then the maximum time interval is obtained.

NOTE You must first load counter value to WDT before enabling the WDT counter to avoid unexpected interruptions.

4.18 Bank2 WDTSafety DO Setup A~D

When WDT interrupt asserts, you can set the system to send out Safety DO value by setting the **SafetyOut_Enable** bit. See Section 3.4 for details. This function is useful for industrial applications with high safety concerns. When WDT INT asserts, the system process may halt or may be offline. Thus, this function can prevent untoward damage. You can configure the default 64-CH safety DO values that are stored in the onboard flash chip. When WDT interrupt asserts and the SafetyOut_Enable bit is enabled, PCI-7442 enters the safety DO procedure which sends out the default safety value to 64-CH digital outputs.

You can program the 64-CH safety default DO values by accessing the **Bank2 WDTSafety DO Setup A~D**. After accessing Bank2 WDTSafety DO Setup D, it takes 0.5 s to finish writing the flash procedure. You can check if the procedure is finished or not by **nAction_Ready** flag.

- ▷ Bank2 WDTSafety DO Setup Register A
- ▷ Address: BASE + 0x98h
- ▷ Attribute: Write

7	6	5	4	3	2	1	0
IDO[7..0]							
15	14	13	12	11	10	9	8
IDO[15..8]							

- ▷ Bank2 WDTSafety DO Setup Register B
- ▷ Address: BASE + 0x9Ah
- ▷ Attribute: Write

7	6	5	4	3	2	1	0
IDO[23..16]							
15	14	13	12	11	10	9	8
IDO[31..24]							

- ▷ Bank2 WDTSafety DO Setup Register C
- ▷ Address: BASE + 0x9Ch
- ▷ Attribute: Write

7	6	5	4	3	2	1	0
IDO[39..32]							
15	14	13	12	11	10	9	8
IDO[47..40]							

- ▷ Bank2 WDTSafety DO Setup Register D
- ▷ Address: BASE + 0x9Eh
- ▷ Attribute: Write

7	6	5	4	3	2	1	0
IDO[55..48]							
15	14	13	12	11	10	9	8
IDO[63..56]							

IDO[n]: Isolated digital output channel n, n = 0 - 63

- ▶ 0: the corresponding PowerMOS is OFF
- ▶ 1: the corresponding PowerMOS is ON
- ▶ The initial value is 0

4.19 Bank2 WDTSafety DO Read Back Start and Bank2 WDTSafety DO Read Back A~D

You can read back the configured Safety DO values that are stored in the flash on board. First, you have to send out the WDT Safety DO Read Back command to Bank2. Then, the flash read procedure proceed after about 50 ms. The finished flag can be checked by nAction_Ready flag. After the Read Back procedure, you can read back the 64-bit Bank2 WDTSafety DO Read Back A~D.

- Bank2 WDTSafety DO Read Back Start
- Address: BASE + 0x96h
- Attribute: Read

7	6	5	4	3	2	1	0
--							
15	14	13	12	11	10	9	8
--							

Bank2 WDTSafety DO Read Back Start does not need any register value. You only need to send out the address (BASE + 0x86h) in **Read** mode before reading back all 64-bit channel output safety data. When Bank2 receives the Start command, the flash read procedure proceeds after about 50 ms. You may check if the procedure is finished by getting nAction_Ready flag status.

- Bank2 WDTSafety DO Read Back Register A
- Address: BASE + 0x98h
- Attribute: Read

7	6	5	4	3	2	1	0
IDO[7..0]							
15	14	13	12	11	10	9	8
IDO[15..8]							

▷ Bank2 WDTSafety DO Read Back Register B

▷ Address: BASE + 0x9Ah

▷ Attribute: Read

7	6	5	4	3	2	1	0
IDO[23..16]							
15	14	13	12	11	10	9	8
IDO[31..24]							

▷ Bank2 WDTSafety DO Read Back Register C

▷ Address: BASE + 0x9Ch

▷ Attribute: Read

7	6	5	4	3	2	1	0
IDO[39..32]							
15	14	13	12	11	10	9	8
IDO[47..40]							

▷ Bank2 WDTSafety DO Read Back Register D

▷ Address: BASE + 0x9Eh

▷ Attribute: Read

7	6	5	4	3	2	1	0
IDO[55..48]							
15	14	13	12	11	10	9	8
IDO[63..56]							

IDO[n]: Isolated digital output channel n, n = 0 - 63

- ▶ 0: the corresponding PowerMOS is OFF
- ▶ 1: the corresponding PowerMOS is ON
- ▶ The initial value is 0

4.20 Handling PCI controller registers

The PCI-7442 adopts the PLX PCI-9030 PCI bus controller. You should notice some registers when you attempt to handle the card via low-level programming. The interrupt control register (INTCSR; 0x4Ch) of PCI-9030 takes charge of all interrupt information from local bus to PCI bus. When you want to develop your own interrupt function driver, both interrupt registers in PCI-9030 and in PCI-7442 have to work together. For detailed information about the interrupt control register in PCI-9030, refer to the PCI-9030 databook.

In the PCI-7442 software function library, we provide simple and easy-to-use functions to handle interrupt procedures. Using these functions, you don't need handle the interrupt register in the PCI controller. It is recommended that you use these functions instead of developing interrupt functions. Refer to Chapter 5 for more information on the PCI-7442 function library.

5 C/C++ DOS libraries

5.1 Programming guide

Naming convention

The NuDAQ PCI card software driver are using full names to represent the functions' real meaning. The naming conventions are:

`_{hardware_model}_{action_name}. e.g. _7442_Initial().`

All PCI-7442 functions have 7442 as {hardware_model}.

Data types

Some data types are defined in the **ACL_PCI.H**. The NuDAQ card library uses these data types. It is recommended that you use these data types in your application programs. The following table shows the data type names and their ranges.

Type Name	Description	Range
U8	8-bit ASCII character	0 to 255
I16	16-bit signed integer	-32768 to 32767
U16	16-bit unsigned integer	0 to 65535
I32	32-bit signed long integer	-2147483648 to 2147483647
U32	32-bit unsigned long integer	0 to 4294967295
F32	32-bit single-precision floating-point	-3.402823E38 to 3.402823E38
F64	64-bit double-precision floating-point	-1.797683134862315E308 to 1.797683134862315E309
Boolean	Boolean logic value	TRUE, FALSE

5.2 _7442_Initial

@ Description

The PCI-7442 card is initialized according to the card number. Because it supports PCI bus architecture and meets the plug and play specifications, the IRQ and base_address (pass-through address) are directly assigned by the system BIOS. The PCI-7442 card must be initialized by this function before using other functions.

@ Syntax

```
U16 _7442_Initial (U16 *existCards, PCI_INFO
     *Info)
```

@ Argument

`existCards`	The number of installed PCI-7442 cards. The returned value shows how many PCI-7442 cards are installed in your system.
`info`	A structure to memorize the PCI bus plug and play initialization information which is decided by PnP BIOS. The PCI_INFO structure is defined in ACL_PCI.H. The base I/O address and the interrupt channel number is stored in pciinfo which is for reference.

@ Return Code

```
ERR_NoError,  ERR_PCIBiosNotExist,
     ERR_BoardNoInit, ERR_InvalidBoardNumber
```

5.3 _7442_DI_Bankn, n=0~1

@ Description

These functions read data from a digital input port. There are two DI banks (n: 0 ~ 1) on the PCI-7442, each has 32-bit digital inputs. You can get 64 input data by using these two functions.

@ Syntax

```
U16 _7442_DI_Bank0 (U16 boardID1, U32 *diData)
U16 _7442_DI_Bank1 (U16 boardID1, U32 *diData)
```

@ Argument

boardID	Board ID to the specific board
diData	return 32-bit value from each digital input port

@ Return Code

```
ERR_BoardIDNotFinded, ERR_BoardNoInit,
     ERR_NoError
```

5.4 _7442_COSETUP_Bankn, n=0~1

@ Description

These functions enable the COS channel for each port.

@ Syntax

```
U16  _7442_COSETUP_Bank0(U16  boardID1, U32
     COS_Enable_Data)
U16  _7442_COSETUP_Bank1(U16  boardID1, U32
     COS_Enable_Data)
```

@ Argument

boardID1	Board ID to the specific board
COS_Enable_Data	32-bit COS channel enable 1 - enables the corresponding channel 0 - disables the corresponding channel

@ Return Code

```
ERR_BoardIDNotFinded, ERR_BoardNoInit,
     ERR_NoError
```

5.5 _7442_COSINT_Control

@ Description

This function controls the COS interrupt source of PCI-7442. For details on interrupt sources, refer to Section 3.2.

@ Syntax

```
U16  _7442_COSINT_Control(U16  boardID1, U16
     COS1_Enable,
U16  COS0_Enable)
```

@ Argument

boardID1	Board ID to the specific board
COS0_Enable	Bank0 COS interrupt function enable (1)/disable (0)
COS1_Enable	Bank1 COS interrupt function enable (1)/disable (0)

@ Return Code

```
ERR_BoardIDNotFinded, ERR_BoardNoInit,
     ERR_NoError
```

5.6 _7442_COSLatch_Bankn, n=0~1

@ Description

These functions latch digital input data for each port after COS interrupt occurs.

@ Syntax

```
U16  _7442_COSLatch_Bank0(U16  boardID1, U32
     *COS0_Latch_Data)
U16  _7442_COSLatch_Bank1(U16  boardID1, U32
     *COS1_Latch_Data)
```

@ Argument

boardID1	Board ID to the specific board
COS0_Latch_Data	Digital input data of bank0 when COS occurs. This register is erased when clearing IRQ.
COS1_Latch_Data	Digital input data of bank1 when COS occurs. This register is erased when clearing IRQ.

@ Return Code

```
ERR_BoardIDNotFinded, ERR_BoardNoInit,
     ERR_NoError
```

5.7 _7442_Bankn_TTLIO_Setup, n=0~1

@ Description

These functions set the directions of the 32-bit TTL I/O channels.

@ Syntax

```
U16  _7442_Bank0_TTLIO_Setup (U16  boardID1, U16
     Bank0_TTLIO_Setup)
U16  _7442_Bank1_TTLIO_Setup (U16  boardID1, U16
     Bank1_TTLIO_Setup)
```

@ Argument

boardID1	Board ID to the specific board
Bank0_TTLIO_Setup	Set the TTL I/O channels direction of bank0 0 - set the corresponding TTL I/O channel direction to input (default) 1 - set the corresponding TTL I/O channel direction to output
Bank1_TTLIO_Setup	Setup TTL I/O Channels Direction of bank1 0 - set the corresponding TTL I/O channel direction to input (default) 1 - set the corresponding TTL I/O channel direction to output

@ Return Code

```
ERR_BoardIDNotFinded, ERR_BoardNoInit,
     ERR_NoError
```

5.8 _7442_Bankn_TTLIO_Status, n=0~1

@ Description

These functions read back the direction statuses of 32-bit TTL I/O channels.

@ Syntax

```
U16  _7442_Bank0_TTLIO_Status(U16  boardID1, U16
     *Bank0_TTLIO_Status)
U16  _7442_Bank1_TTLIO_Status(U16  boardID1, U16
     *Bank1_TTLIO_Status)
```

@ Argument

`boardID1`	Board ID to the specific board
`Bank0_TTLIO_Status`	Read back the TTL I/O Channels Direction statuses of bank0 0 - the corresponding TTL I/O channel direction status is input (default) 1 - the corresponding TTL I/O channel direction status is output
`Bank1_TTLIO_Status`	Read back the TTL I/O Channels Direction statuses of bank1 0 - the corresponding TTL I/O channel direction status is input (default) 1 - the corresponding TTL I/O channel direction status is output

@ Return Code

```
ERR_BoardIDNotFinded, ERR_BoardNoInit,
     ERR_NoError
```

5.9 _7442_Bankn_TTLIO_DI, n=0~1

@ Description

These functions read data from 32-CH TTL I/O input ports. The PCI-7442 board has two banks, each of which has a 16-bit TTL I/O. You can get 32-bit TTL input data using these functions.

@ Syntax

```
U16  _7442_Bank0_TTLIO_DI(U16  boardID1, U16
     *Bank0_TTLIO_DI)
U16  _7442_Bank1_TTLIO_DI(U16  boardID1, U16
     *Bank1_TTLIO_DI)
```

@ Argument

boardID1	Board ID to the specific board
Bank0_TTLIO_DI	return the 16-CH TTL input data of bank0
Bank1_TTLIO_DI	return the 16-CH TTL input data of bank1

@ Return Code

```
ERR_BoardIDNotFinded, ERR_BoardNoInit,
     ERR_NoError
```

5.10 _7442_Bankn_TTLIO_DO, n=0~1

@ Description

These functions write data to 32-CH TTL I/O output ports. The PCI-7442 board has two banks, each of which has a 16-bit TTL I/O. You can write a 32-bit TTL output data by using these functions.

@ Syntax

```
U16  _7442_Bank0_TTLIO_DO (U16  boardID1, U16
     Bank0_TTLIO_DO)
U16  _7442_Bank1_TTLIO_DO (U16  boardID1, U16
     Bank1_TTLIO_DO)
```

@ Argument

boardID1	Board ID to the specific board
Bank0_TTLIO_DO	16-bit value that will be written on each TTL output port of bank0
Bank1_TTLIO_DO	16-bit value that will be written on each TTL output port of bank1

@ Return Code

```
ERR_BoardIDNotFinded, ERR_BoardNoInit,
     ERR_NoError
```

5.11 _7442_Bank2_DO

@ Description

These functions write data to digital output ports which can be used to turn on/off the output power MOSFET. You can control all 64 channels by using this function. 0 means that the corresponding PowerMOS is OFF; 1 means that the corresponding PowerMOS is ON.

@ Syntax

```
U16  _7442_Bank2_DO(U16  boardID1, U32
     Bank2_DO_DataA ,
U32  Bank2_DO_DataB)
```

@ Argument

boardID1	Board ID to the specific board
Bank2_DO_DataA	the low 32-bit output data which will be written to corresponding digital output port
Bank2_DO_DataB	the high 32-bit output data which will be written to corresponding digital output port

@ Return Code

```
ERR_BoardIDNotFinded, ERR_BoardNoInit,
     ERR_NoError
```

5.12 _7442_Bank2_DORB

@ Description

These functions read data back from each digital output port. There are 64-bit digital outputs on the PCI-7442. You can get back all DO data by using this function. 0 means that the corresponding PowerMOS is OFF; 1 means that the corresponding PowerMOS is ON.

@ Syntax

```
U16  _7442_Bank2_DORB (U16  boardID1, U32
     *Bank2_DO_DataA ,
U32  *Bank2_DO_DataB)
```

@ Argument

boardID1	Board ID to the specific board
Bank2_DO_DataA	the low 32-bit output data which will be read back from corresponding digital output port
Bank2_DO_DataB	the high 32-bit output data which will be read back from corresponding digital output port

@ Return Code

```
ERR_BoardIDNotFinded, ERR_BoardNoInit,
     ERR_NoError
```

5.13 _7442_Bank2_WDT_Load

@ Description

This function clears the watch dog timer counter zero state and reloads the 32-bit count value (1 ~ 4,294,967,295 [232-1]). The overflow time is calculated by the value that you set multiplied 100 ns. The timer interval is from 0 to 429.496 seconds.

@ Syntax

```
U16  _7442_Bank2_WDT_Load (U16  boardID1, U32
     WDT_Load_Value)
```

@ Argument

boardID1	Board ID to the specific board
WDT_Load_Value	the 32-bit value from 1 ~ 4,294,967,295(232-1) that will be written to WDT reload value. It must not be zero, else it will return this error message: ERR_WDTNotSet

@ Return Code

```
ERR_BoardIDNotFinded, ERR_BoardNoInit,
     ERR_WDTNotSet, ERR_NoError
```

5.14 _7442_WDTINT_Control

@ Description

This function controls the WDT interrupt source of the PCI-7442 and the Safety-DO configuration function. For more details on WDT interrupt sources, refer to Section 3.5.

@ Syntax

```
U16  _7442_WDTINT_Control(U16  boardID1, U16
     WDT_Enable,
U16  Safety_out_Enable)
```

@ Argument

boardID1	Board ID to the specific board
WDT_Enable	Bank2 WDT interrupt Enable control. 0 disable WDT function (default); 1 enable WDT function
Safety_out_Enable	Write 1 to Enable Send Out Safety DO status function while WDT INT asserts; Write 0 disable function, DO status will be hold, not change while WDT INT asserts (default)

@ Return Code

```
ERR_BoardIDNotFinded, ERR_BoardNoInit,
     ERR_NoError
```

5.15 _7442_GET_IRQ_Status

@ Description

This function gets the interrupt status of PCI-7442.

@ Syntax

```
U16  _7442_GET_IRQ_Status( U16  boardID1, U16
     *COS1_EN_Status,
          U16  *COS0_EN_Status, U16
     *COS1_Status, U16  *COS0_Status,
U16  *WDT_EN_Status, U16  *WDT_INT_Status ,
U16  *LINTi1_Status , U16  *LINTi2_Status);
```

@ Argument

boardID1	Board ID to the specific board
COS0_EN_Status	Bank0 DI[31..0] COS interrupt enable status 1: Bank0 COS interrupt enabled 0: Bank0 COS interrupt disabled (default)
COS1_EN_Status	Bank1 DI[63..32] COS interrupt enable status 1: Bank1 COS interrupt enabled 0: Bank1 COS interrupt disabled (default)
COS0_Status	Bank0 DI[31..0] COS interrupt status register 1: Bank0 COS interrupt asserts 0: Bank0 COS interrupt de-asserts (default)
COS1_Status	Bank1 DI[63..32] COS interrupt status register 1: Bank1 COS interrupt asserts 0: Bank1 COS interrupt de-asserts (default)
WDT_EN_Status	Read back Bank2 WDT interrupt Enable Status 0: The WDT interrupt is disabled (default) 1: The WDT interrupt is enabled
WDT_INT_Status	Read Back the WDT interrupt Status if the interrupt asserts 0: The WDT interrupt does not assert (default) 1: The WDT interrupt asserts
LINTi1_Status	Read Back the PLX PCI-controller interrupt 1 (LINTi1) Status if the interrupt asserts 0: The LINTi1 interrupt does not assert (default) 1: The LINTi1 interrupt asserts
LINTi1_Status	Read Back the PLX PCI-controller interrupt 2 (LINTi2) Status if the interrupt asserts 0: The LINTi2 interrupt does not assert (default) 1: The LINTi2 interrupt asserts

@ Return Code

ERR_BoardIDNotFinded, ERR_BoardNoInit,
ERR_NoError

5.16 _7442_CLR_IRQ

@ Description

This function clears the PCI-7442 interrupt request.

@ Syntax

```
U16  _7442_CLR_IRQ(U16  boardID1, U16  WDT_CLR,
U16  COS_CLR1, U16  COS_CLR0)
```

@ Argument

boardID1	Board ID to the specific board
WDT_CLR	Clear Watch Dog Timer interrupt request
COS_CLR0	Clear Bank0 COS interrupt request
COS_CLR1	Clear Bank1 COS interrupt request

@ Return Code

```
ERR_BoardIDNotFinded, ERR_BoardNoInit,
     ERR_NoError
```

5.17 _7442_Bank2_Powerup_Setup

@ Description

When the system enters a power up status, the PCI-7442 can enter the initial procedure which sends out the default initial value to 64-CH digital outputs. You can configure the power-up default DO values and store them in the flash by using this function. Thus, the DO will not enter an unknown status when the system is turned on. You can program the 64-CH power-up default DO values through accessing the Bank2 Power-up DO Setup Register (16-bit) A~D in turn by calling the _7442_Bank2_Powerup_Setup() function. See Section 3.4 for details. After accessing Bank2 Power-up DO Setup Register D, it takes about 0.5 s to finish writing flash procedure. Users can check if the procedure is finished or not by checking nAction_Ready flag.

@ Syntax

```
U16 _7442_Bank2_Powerup_Setup(U16  boardID1,
 U32  Bank2_Powerup_DataA ,
U32  Bank2_Powerup_DataB )
```

@ Argument

boardID1	Board ID to the specific board
Bank2_Powerup_DataA	the low 32-bits power-up initial DO value which will be written to flash
Bank2_Powerup_DataB	the high 32-bits power-up initial DO value which will be written to flash

@ Return Code

```
ERR_BoardIDNotFinded, ERR_BoardNoInit,
     ERR_NoError
```

5.18 _7442_Bank2_Powerup_RB

@ Description

You can read back the configured the power-up initial DO values that are stored in the flash on board through the function _7442_Bank2_Powerup_RB. The flash read procedure starts after about 50 ms. The finished flag can be checked by nAction_Ready flag. After the Read Back procedure, you can read back the 64-bit Bank2 Power-up DO Read Back Register (16bit) A~D.

@ Syntax

```
U16  _7442_Bank2_Powerup_RB(U16  boardID1,
U32  *Bank2_Powerup_DataA ,
U32  *Bank2_Powerup_DataB)
```

@ Argument

boardID1	Board ID to the specific board
Bank2_Powerup_DataA	the low 32-bits power-up initial DO value which will be read back from flash
Bank2_Powerup_DataB	the high 32-bits power-up initial DO value which will be read back from flash

@ Return Code

```
ERR_BoardIDNotFinded, ERR_BoardNoInit,
     ERR_NoError
```

5.19 _7442_Bank2_WDTsafety_Setup

@ Description

When WDT interrupt asserts, users can choose if system send out Safety DO value to prevent some unknown damage by setting the **SafetyOut_Enable** bit. See Section 3.4 for details. This function is useful for some industrial applications with high safety requirements. When WDT INT asserts, the system process could halt or be offline. Thus, this function can prevent untoward damage. You may configure the default 64-CH safety DO values which are stored in flash through the function _7442_Bank2_WDTsafety_Setup(). When WDT interrupt asserts and the SafetyOut_Enable bit is enabled, PCI-7442 can enter the safety DO procedure which sends out the default safety value to 64-CH digital outputs. You can program the 64-CH safety default DO values through accessing the Bank2 WDTSafety DO Setup (16-bit) A~D. After accessing Bank2 WDTSafety DO Setup D, it takes 0.5 s to finish writing flash procedure. You can check if the procedure is finished or not by nAction_Ready flag.

@ Syntax

```
U16 _7442_Bank2_WDTsafety_Setup (U16 boardID1,
U32  Bank2_WDTsafety_DataA ,
U32  Bank2_WDTsafety_DataB)
```

@ Argument

boardID1	Board ID to the specific board
Bank2_WDTsafety_DataA	the low 32-bits safety DO value which will be written to flash
Bank2_WDTsafety_DataB	the high 32-bits safety DO value which will be written to flash

@ Return Code

```
ERR_BoardIDNotFinded, ERR_BoardNoInit,
     ERR_NoError
```

5.20 _7442_Bank2_WDTsafety_RB

@ Description

With this function, you can read back the configured Safety DO values that are stored in the onboard flash. The flash read procedure starts after about 50 ms. The finished flag can be checked by nAction_Ready flag. After the Read Back procedure, you can read back the 64-bit Bank2 WDTSafety DO Read Back(16bit) A~D.

@ Syntax

```
U16 _7442_Bank2_WDTsafety_RB (U16  boardID1,
                                    U32
     *Bank2_WDTsafety_DataA ,
U32  *Bank2_WDTsafety_DataB)
```

@ Argument

boardID1	Board ID to the specific board
Bank2_WDTsafety_DataA	the low 32-bits safety DO value which will be read from flash
Bank2_WDTsafety_DataB	the high 32-bits safety DO value which will be read from flash

@ Return Code

```
ERR_BoardIDNotFinded, ERR_BoardNoInit,
     ERR_NoError
```

5.21 _7442_HotReset_Control

@ Description

When the system goes through a unexpected or normal hot system reset without turning off the system power, you may either choose to retain the original DO values held before resetting or allow the board to enter the power-up initial procedure and send out the default initial DO values that you configured. By setting the **Hot_Reset_Hold_Enable** bit through the _7442_HotReset_Control function, you can enable the Hot-Reset-Hold function anytime. This function is particularly useful for some unstable environment.

@ Syntax

```
U16  _7442_HotReset_Control(U16  boardID1, U16
     HotReset_Enable)
```

@ Argument

boardID1	Board ID to the specific board
HotReset_Enable	Enable Hot-System-Reset DO Hold function 0: disable function (default) 1: enable function

@ Return Code

```
ERR_BoardIDNotFinded, ERR_BoardNoInit,
     ERR_NoError
```

5.22 _7442_GET_HotReset_Status

@ Description

This function reads back the Hot_Reset_Hold_Enable bit status. You can check if the Hot-Reset-Hold function of the PCI-7442 board is enabled through the _7442_GET_HotReset_Status function.

@ Syntax

```
U16 _7442_GET_HotReset_Status(U16 boardID1, U16
    *HotReset_Status)
```

@ Argument

boardID1	Board ID to the specific board
HotReset_Status	Read back Hot-System Reset DO Hold function Enable Status 0: function is disabled (default) 1: function is enabled

@ Return Code

```
ERR_BoardIDNotFinded, ERR_BoardNoInit,
    ERR_NoError
```

5.23 _7442_GET_Safetyout_Status

@ Description

This function reads back the SafetyOut_Enable_Status bit status.

@ Syntax

```
U16  _7442_GET_Safetyout_Status(U16  boardID1,
     U16  *Safetyout_Status)
```

@ Argument

boardID1	Board ID to the specific board
Safetyout_Status	Read back the Send Out Safety DO status function Enable Status 0: function is disabled (default) 1: function is enabled

@ Return Code

```
ERR_BoardIDNotFinded, ERR_BoardNoInit,
     ERR_NoError
```

5.24 _7442_GET_nActionReady_Flag

@ Description

The nAction_Ready flag is used to detect if the Flash Data Write/ Read procedure is finished or not. You can get the information on the nAction_Ready flag through the _7442_GET_nActionReady_Flag function.

@ Syntax

```
U16 _7442_GET_nActionReady_Flag(U16  boardID1,
     U16  *nActionReady_Flag)
```

@ Argument

boardID1	Board ID to the specific board
nActionReady_Flag	detect flag for users to know if Flash Data W/R is finished 0: finished, ready for users to access next step (default) 1: not finished, please hold

@ Return Code

```
ERR_BoardIDNotFinded, ERR_BoardNoInit,
     ERR_NoError
```

5.25 _7442_GET_nDO_SendReady

@ Description

The nDO_SendReady flag detects if the DO Data Sending procedure is finished or not. Users can get the information about the nDO_SendReady flag through the _7442_GET_nDO_SendReady function.

@ Syntax

```
U16  _7442_GET_nDO_SendReady(U16  boardID1, U16
     *nDO_SendReady)
```

@ Argument

boardID1	Board ID to the specific board
nDO_SendReady	detect flag for users to know if DO Data Sending procedure is finished 0: finished, ready for users to access next step 1: not finished, please hold

@ Return Code

```
ERR_BoardIDNotFinded, ERR_BoardNoInit,
     ERR_NoError
```

5.26 _7442_GET_nDO_RBReady

@ Description

The nDO_RBReady flag detects if the DO Data reading back procedure is finished or not. Users can get the information about the nDO_RBReady flag through the _7442_GET_nDO_RBReady function.

@ Syntax

```
U16  _7442_GET_nDO_RBReady(U16  boardID1, U16
     *nDO_RBReady)
```

@ Argument

boardID1	Board ID to the specific board
nDO_RBReady	detect flag for users to know if DO Data reading back procedure is finished 0: finished, ready for users to access next step 1: not finished, please hold

@ Return Code

```
ERR_BoardIDNotFinded, ERR_BoardNoInit,
     ERR_NoError
```

Warranty Policy

Thank you for choosing ADLINK. To understand your rights and enjoy all the after-sales services we offer, please read the following carefully.

1. Before using ADLINK's products please read the user manual and follow the instructions exactly. When sending in damaged products for repair, please attach an RMA application form which can be downloaded from: http://rma.adlinktech.com/policy/.
2. All ADLINK products come with a limited two-year warranty, one year for products bought in China:
   - The warranty period starts on the day the product is shipped from ADLINK's factory.
   - Peripherals and third-party products not manufactured by ADLINK will be covered by the original manufacturers' warranty.
   - For products containing storage devices (hard drives, flash cards, etc.), please back up your data before sending them for repair. ADLINK is not responsible for any loss of data.
   - Please ensure the use of properly licensed software with our systems. ADLINK does not condone the use of pirated software and will not service systems using such software. ADLINK will not be held legally responsible for products shipped with unlicensed software installed by the user.
   - For general repairs, please do not include peripheral accessories. If peripherals need to be included, be certain to specify which items you sent on the RMA Request & Confirmation Form. ADLINK is not responsible for items not listed on the RMA Request & Confirmation Form.

3. Our repair service is not covered by ADLINK's guarantee in the following situations:
   - Damage caused by not following instructions in the User's Manual.
   - Damage caused by carelessness on the user's part during product transportation.
   - Damage caused by fire, earthquakes, floods, lightening, pollution, other acts of God, and/or incorrect usage of voltage transformers.
   - Damage caused by unsuitable storage environments (i.e. high temperatures, high humidity, or volatile chemicals).
   - Damage caused by leakage of battery fluid during or after change of batteries by customer/user.
   - Damage from improper repair by unauthorized ADLINK technicians.
   - Products with altered and/or damaged serial numbers are not entitled to our service.
   - This warranty is not transferable or extendible.
   - Other categories not protected under our warranty.
4. Customers are responsible for shipping costs to transport damaged products to our company or sales office.
5. To ensure the speed and quality of product repair, please download an RMA application form from our company website: http://rma.adlinktech.com/policy. Damaged products with attached RMA forms receive priority.

If you have any further questions, please email our FAE staff: service@adlinktech.com.