PIO-D144/D144U PIO-D168/D168U

User Manual

Version 3.0

November 2010

Warranty

All products manufactured by ICP DAS are warranted against defective materials for a period of one year from the date of delivery to the original purchaser.

Warning

ICP DAS assumes no liability for damages consequent to the use of this product. ICP DAS reserves the right to change this manual at any time without notice. The information furnished by ICP DAS is believed to be accurate and reliable. However, no responsibility is assumed by ICP DAS for its use, nor for any infringements of patents or other rights of third parties resulting from its use.

Copyright

Trademark

Names are used for identification only and may be registered trademarks of their respective companies.

Tables of Contents

1. Introduction

The PIO-D144U/D168U card is a new generation product provided by ICP DAS to meet RoHS compliance requirements. The new PIO-D144U/D168U card is designed as a drop-in replacement for the PIO-D144/D168, and users can directly replace the PIO-D144/D168 with the PIO-D144U/D168U without the need for software/driver modification.

The PIO-D144U/D168U universal PCI card supports the 3.3 V/5 V PCI bus, while the PIO-D144/D168 supports the 5 V PCI bus. These cards provide 144/168 TTL digital I/O lines, and these lines are grouped into eighteen 8-bit bi-directional ports. Every three ports on a connector are grouped and named as Port A (PA), Port B (PB) and Port C (PC), respectively. All ports are configured as inputs upon power-up or reset.

The PIO-D144U/D168U also includes an onboard Card ID switch. Once the Card ID is set, the board can be identified by the software when using more than one PIO-D144U/D168U cards in a single computer based on the ID.

The PIO-D144/D168 series cards supports various OS versions, such as Linux, DOS, Windows 98/NT/2000 and 32-bit/64-bit Windows XP/2003/Vista/7. DLL and Active X control together with various language sample programs based on Turbo C++, Borland C++, Microsoft C++, Visual C++, Borland Delphi, Borland C++ Builder, Visual Basic, C#.NET, Visual Basic.NET and LabVIEW are provided in order to help users quickly and easily develop their own applications.

1.1 Specifications

1.1.1 PIO-D144/PIO-D144U

Model Name	PIO-D144	PIO-D144U
Programmable Digital I/O		
Channels	144	
Digital Input		
Compatibility	5 V/TTL	
Input Voltage	Logic 0: 0.8 V max. Logic 1: 2.0 V min.	
Response Speed	1.2 MHz (Typical)	
Digital Output		
Compatibility	5 V/TTL	
Output Voltage	Logic 0: 0.4 V max. Logic 1: 2.4 V min.	
Output Capability	Sink: 64 mA @ 0.8 V Source: 32 mA @ 2.0 V	
Response Speed	1.2 MHz (Typical)	
General		
Bus Type	5 V PCI, 32-bit, 33 MHz	3.3 V/5 V Universal PCI, 32-bit, 33 MHz
Data Bus	8-bit	
Card ID	No	Yes (4-bit)
I/O Connectors	Female DB37 x 1 50-pin box header x 5	
Dimensions (L x W x D)	180 mm x 105 mm x 22 mm	
Power Consumption	1100 mA @ +5 V	
Operating Temperature	0 ~ 60 °C	
Storage Temperature	-20 ~ 70 °C	
Humidity	5 ~ 85% RH, non-condensing	

1.1.2 PIO-D168/PIO-D168U

Model Name	PIO-D168	PIO-D168U
Programmable Digital I/O		
Channels	168	
Digital Input		
Compatibility	5 V/TTL	
Input Voltage	Logic 0: 0.8 V max. Logic 1: 2.0 V min.	
Response Speed	1.2 MHz (Typical)	
Digital Output		
Compatibility	5 V/TTL	
Output Voltage	Logic 0: 0.4 V max. Logic 1: 2.4 V min.	
Output Capability	Sink: 64 mA @ 0.8 V Source: 32 mA @ 2.0 V	
Response Speed	1.2 MHz (Typical)	
General		
Bus Type	5 V PCI, 32-bit, 33 MHz	3.3 V/5 V Universal PCI, 32-bit, 33 MHz
Data Bus	8-bit	
Card ID	No	Yes (4-bit)
I/O Connectors	Female DB37 x 1 50-pin box header x 6	
Dimensions (L x W x D)	200 mm x 105 mm x 22 mm	
Power Consumption	1300 mA @ +5 V	
Operating Temperature	0 ~ 60 °C	
Storage Temperature	-20 ~ 70 °C	
Humidity	5 ~ 85% RH, non-condensing	

1.2 Features

- Supports the +5 V PCI bus for PIO-D144/D168.
- Supports the +3.3 V/+5 V PCI bus for PIO-D144U/D168U.
- PIO-D144(U): Five 50-pin flat cable connectors and one 37-pin connector.
- PIO-D168(U): Six 50-pin flat cable connectors and one 37-pin connector.
- Card ID function for PIO-D144U/D168U.
- Output status readback function.
- Each port consists of three 8-bit ports, PA, PB and PC in every connector.
- Each port can be independently configured as either DI or DO at the same time.
- PIO-D144/D144U board: 6 connectors = 6×3 ports = 6×3×8 bits =144 bits.
- PIO-D168/D168U board: 7 connectors = 7×3 ports = 7×3×8 bits =168 bits.
- 4 interrupt sources: P2C0, P2C1, P2C2 and P2C3.
- Connect directly to DB-24, DB-24R, DB-24PR, DB-24SSR, DB-24POR and other OPTO-22 compatible daughter boards.

1.3 Product Check List

The shipping package includes the following items:

- One PIO-D144/D168 series card
- One software utility PCI CD.
- One Quick Start Guide

It is recommended that you read the Quick Start Guide first. All the necessary and essential information is given in the Quick Start Guide, including:

- Where to get the software driver, demo programs and other resources.
- How to install the software.
- How to test the card.

Attention!

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

2. Hardware Configuration

2.1 Board Layout

The board layout of the PIO-D144/D168 series cards are shown below:

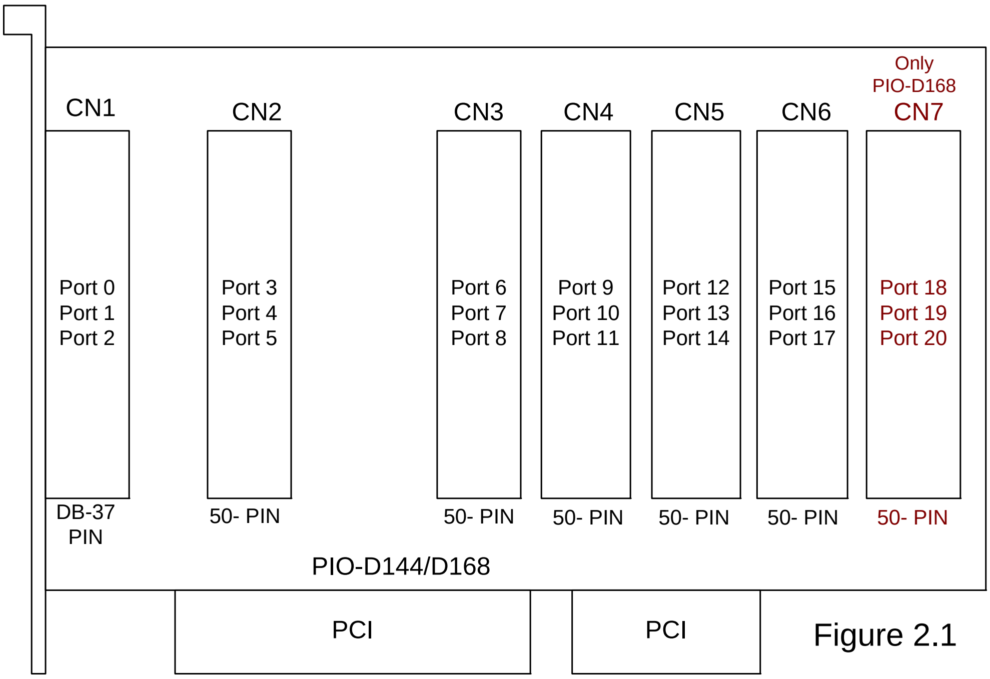


Figure 2.1

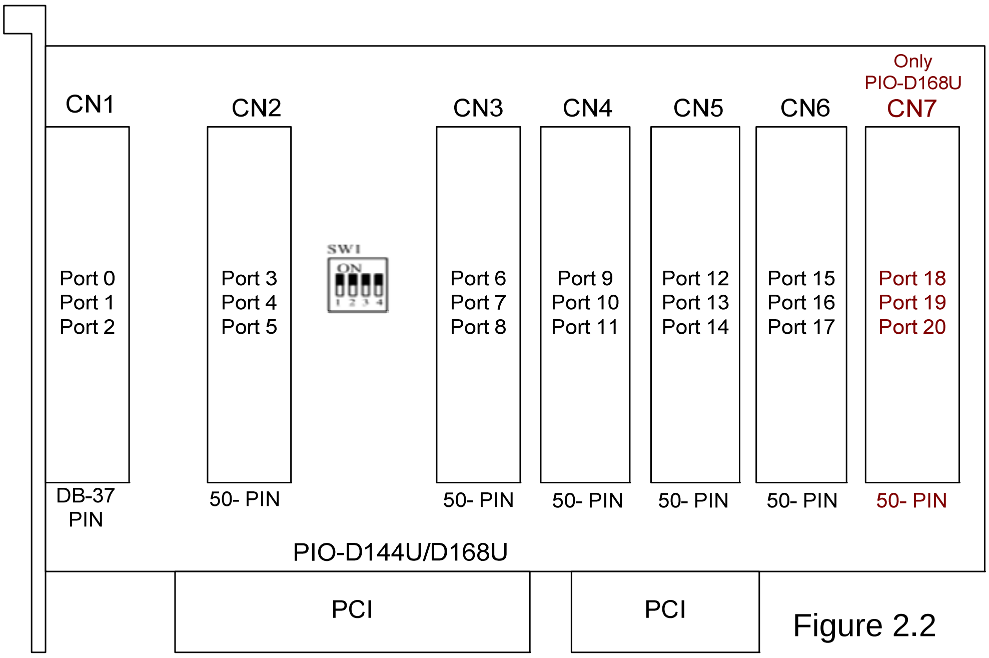


Figure 2.2

2.2 I/O Port Location

There are 18/21 8-bit I/O ports in the PIO-D144/D168 series. Every port can be independently configured as a D/I or D/O port. When the PC is first powered-on, all ports are set as Digital input ports. Therefore, the user needs to configure these ports as either digital input or output ports before using then in an application. Each I/O port is named as the following table and its location can be found in Figure 2.1 and 2.2.

Table 2.1

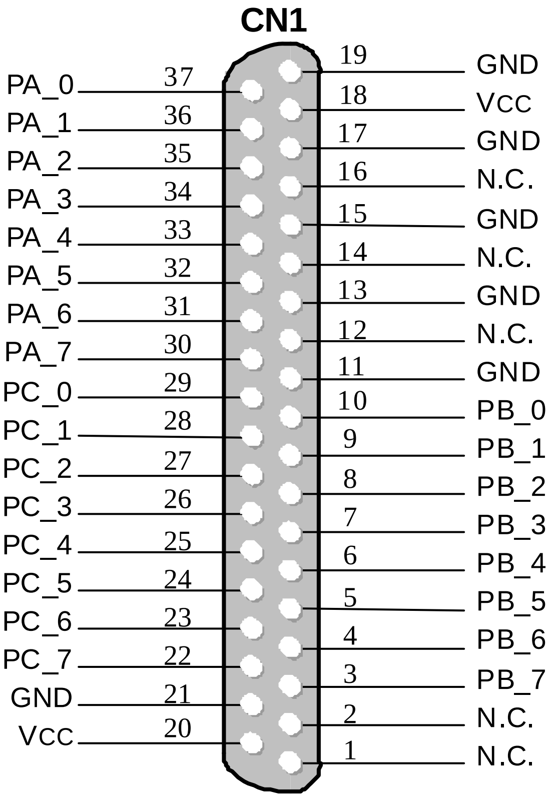
Connector of PIO-D144(U)/D168(U)	PA0 ~ PA7	PB0 ~ PB7	PC0 ~ PC7
CN1	Port0	Port1	Port2
CN2	Port3	Port4	Port5
CN3	Port6	Port7	Port8
CN4	Port9	Port10	Port11
CN5	Port12	Port13	Port14
CN6	Port15	Port16	Port17
CN7 (PIO-D168/168U Only)	Port18	Port19	Port20

Refer to Sec. 2.1 for details of board layout and I/O port location.

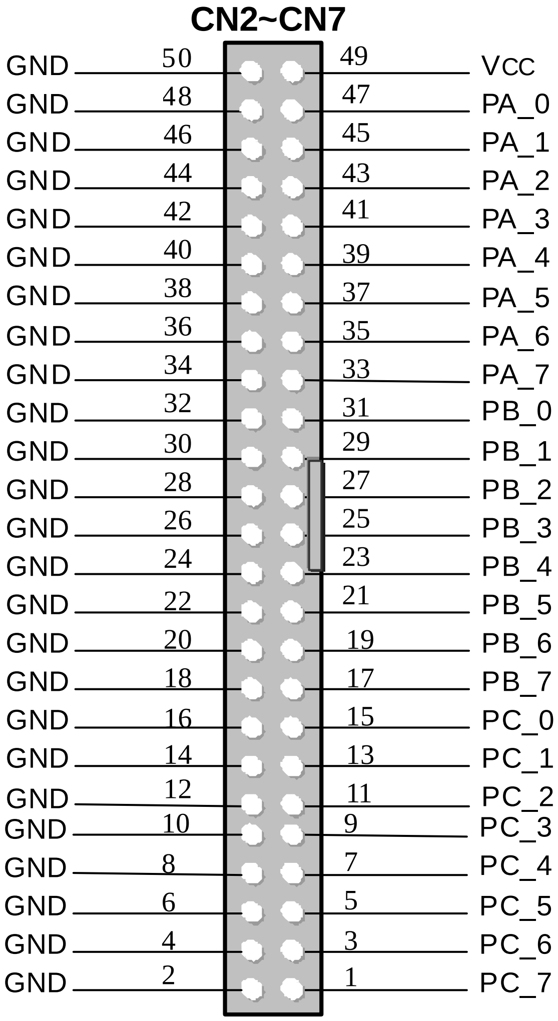
! **Note:**
P2C0, P2C1, P2C2, P2C3 interrupts and of the CN1 connector can be used as an interrupt signal source. Refer to Sec. 2.3 and 2.6 Pin Assignments for more information.

2.3 Pin Assignments

The Pin assignments for all PIO-D144/D168 series connectors are listed in Tables 2.2 and 2.3. All signal sources for each digital input or output pin (channel) are TTL compatible. **Note: CN7 (Port18~Port20) is only for the PIO-D168/D168U.**



37-PIN D-type female connector



50-PIN of flat-cable connector

Table 2.2 CN1: 37-PIN D-type female connector for Port0~Port2

Pin Number	Description	Pin Number	Description
1	N. C.	20	VCC
2	N. C.	21	GND
3	P1B7	22	P2C7
4	P1B6	23	P2C6
5	P1B5	24	P2C5
6	P1B4	25	P2C4
7	P1B3	26	P2C3
8	P1B2	27	P2C2
9	P1B1	28	P2C1
10	P1B0	29	P2C0
11	GND	30	P0A7
12	N.C.	31	P0A6
13	GND	32	P0A5
14	N.C.	33	P0A4
15	GND	34	P0A3
16	N.C.	35	P0A2
17	GND	36	P0A1
18	VCC	37	P0A0
19	GND	XXXXXXX	This pin not available

Table 2.3 CN2/3/4/5/6/7: 50-PIN of flat-cable connector for Port3~Port20

Pin Number	Description	Pin Number	Description
1	P5C7/ P8C7/ P11C7/ P14C7/ P17C7/ P20C7	2	GND
3	P5C6/ P8C6/ P11C6/ P14C6/ P17C6/ P20C6	4	GND
5	P5C5/ P8C5/ P11C5/ P14C5/ P17C5/ P20C5	6	GND
7	P5C4/ P8C4/ P11C4/ P14C4/ P17C4/ P20C4	8	GND
9	P5C3/ P8C3/ P11C3/ P14C3/ P17C3/ P20C3	10	GND
11	P5C2/ P8C2/ P11C2/ P14C2/ P17C2/ P20C2	12	GND
13	P5C1/ P8C1/ P11C1/ P14C1/ P17C1/ P20C1	14	GND
15	P5C0/ P8C0/ P11C0/ P14C0/ P17C0/ P20C0	16	GND
17	P4B7/ P7B7/ P10B7/ P13B7/ P16B7/ P19B7	18	GND
19	P4B6/ P7B6/ P10B6/ P13B6/ P16B6/ P19B6	20	GND
21	P4B5/ P7B5/ P10B5/ P13B5/ P16B5/ P19B5	22	GND
23	P4B4/ P7B4/ P10B4/ P13B4/ P16B4/ P19B4	24	GND
25	P4B3/ P7B3/ P10B3/ P13B3/ P16B3/ P19B3	26	GND
27	P4B2/ P7B2/ P10B2/ P13B2/ P16B2/ P19B2	28	GND
29	P4B1/ P7B1/ P10B1/ P13B1/ P16B1/ P19B1	30	GND
31	P4B0/ P7B0/ P10B0/ P13B0/ P16B0/ P19B0	32	GND
33	P3A7/ P6A7/ P9A7/ P12A7/ P15A7/ P18A7	34	GND
35	P3A6/ P6A6/ P9A6/ P12A6/ P15A6/ P18A6	36	GND
37	P3A5/ P6A5/ P9A5/ P12A5/ P15A5/ P18A5	38	GND
39	P3A4/ P6A4/ P9A4/ P12A4/ P15A4/ P18A4	40	GND
41	P3A3/ P6A3/ P9A3/ P12A3/ P15A3/ P18A3	42	GND
43	P3A2/ P6A2/ P9A2/ P12A2/ P15A2/ P18A2	44	GND
45	P3A1/ P6A1/ P9A1/ P12A1/ P15A1/ P18A1	46	GND
47	P3A0/ P6A0/ P9A0/ P12A0/ P15A0/ P18A0	48	GND
49	VCC	50	GND

2.4 Enable I/O Operation

When the PC is first powered-on, the operations of all digital I/O channels for each port are disabled. Note that the digital I/O channel for each port is enabled or disabled using the RESET\ signal. Refer to Sec. 3.3.1 for more information. The power-on states for all DI/O ports are as follows:

- D/I/O operations for each port are disabled.
- D/I/O ports are all configured as Digital input ports.
- D/O latch registers are all undefined. Refer to Sec. 2.5 for details.

The user must perform an initialization procedure before using these digital I/O ports. The recommended steps are as follows:

Step 1: Determine the address mapping of the PIO/PISO cards.
(Refer to Sec.3.1 for details)

Step 2: Enable all Digital I/O operations. (Refer to Sec. 3.3.1 for details).

Step 3: Select the controlled port. (Refer to Sec. 3.3.8 for details).

Step 4: Send initial value to the D/O latch register for the controlled port.
(Refer to Sec. 2.5 and Sec. 3.3.7 for details)

Step 5: Repeat Steps 3 and 4 to initialize the other D/O ports.

Step 6: Configure all Digital I/O ports to their expected D/I or D/O function. (Refer to Sec. 3.3.9 for details)

For more information regarding the initialization procedure for digital I/O ports, please refer to the DEMO1.C demo program.

2.5 D/I/O Architecture

The digital I/O control architecture for the PIO-D144/D168 series is illustrated in Figure 2.2. The operation method of control signal is described below.

- When the RESET\ signal is in the Low-state, if means that all D/I/O operations are disabled.
- When the RESET\ signal is in the High-state, if means that all D/I/O operations are enabled.
- If the D/I/O is configured as a D/I port, the port can accept digital input from external signal sources.
- If the D/I/O is configured as a D/O port, the digital output value can be read from the port.
- If the D/I/O is configured as a D/I port, sending data to the Digital input port will only change the D/O latch register. The latched data will be output when the port is configured as a digital output port and is activated right away.

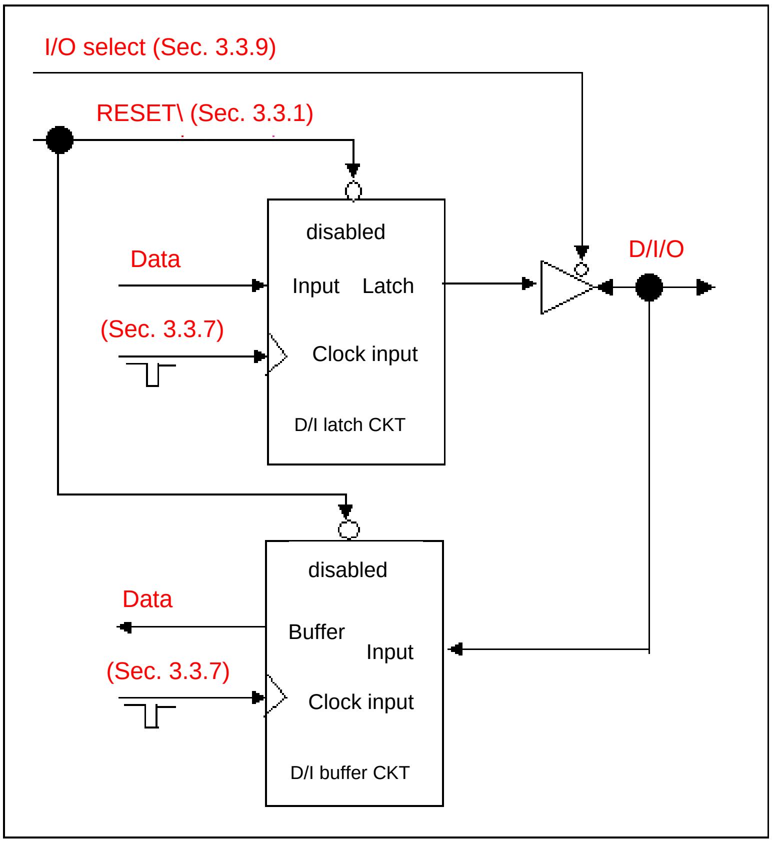


Figure 2.2

2.6 Interrupt Operation

The P2C0, P2C1, P2C2 and P2C3 interrupts pins of the CN1 connector can be used as an interrupt signal source. Refer to Sec. 2.1 for the location of the P2C0/P2C1/P2C2/P2C3 pins. The interrupt of the PIO-D144/D168 series **is level-triggered and is Active_High**. The interrupt signal can be programmed as either **inverted or non-inverted**. The procedure for how to configure the interrupt signal source is as follows:

1. Determine whether **the initial level is either High or Low from the signal source.**
2. If the initial state is High, select the **inverted** setting for the interrupt signal source (Section. 3.3.6). If the initial state is Low, select the **non-inverted** setting for the interrupt signal source (Section. 3.3.6)
3. Enable the interrupt function (Section. 3.3.4)
4. If the interrupt signal is active, the interrupt service routine will be automatically started.

Note that DEMO3.C and DEMO4.C are demo programs for a single interrupt source and DEMO5.C is a demo program for four interrupt sources in a DOS operating system. If only one interrupt signal source is used, the interrupt service routine does not need to identify the interrupt source. (Refer to DEMO3.C and DEMO4.C). However, if there is more than one interrupts source, the interrupt service routine needs to identify the active signals as follows: (Refer to DEMO5.C)

1. Read the new status of the interrupt signal source
2. Compare the new status with the old status to identify the active signals
3. If P2C0 is active, service P2C0 and non-inverter/inverted as the P2C0 signal
4. If P2C1 is active, service P2C1 and non-inverted/inverted as the P2C1 signal
5. If P2C2 is active, service P2C2 and non-inverted/inverted as the P2C2 signal
6. If P2C3 is active, service P2C3 and non-inverted/inverted the P2C3 signal
7. Save the new status to old status

Limitation: if the interrupt signal is too short, the new status may be the same as the old status. In this case, the interrupt signal must be held at active until the interrupt service routine is executed. This hold time is different for different OS versions and can be as short as micro-second or as a long as second. In general, 20 ms is enough for all types of OS.

2.7 Daughter Boards

2.7.1 DB-37

The DB-37 is a general-purpose daughter board for D-sub 37-pin devices, and is designed for easy wiring.

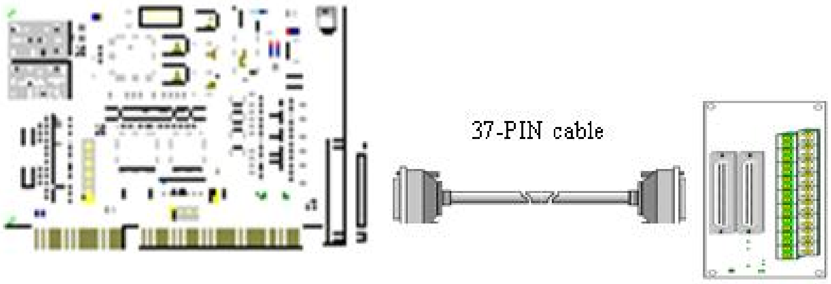


Figure 2.5

2.7.2 DN-37 and DN-50

The DN-37 is a general-purpose daughter board for the DB-37 using DIN-Rail Mounting. The DN-50 is designed for a 50-pin flat-cable header using DIN-Rail mounting. They are also designed for easy wiring.

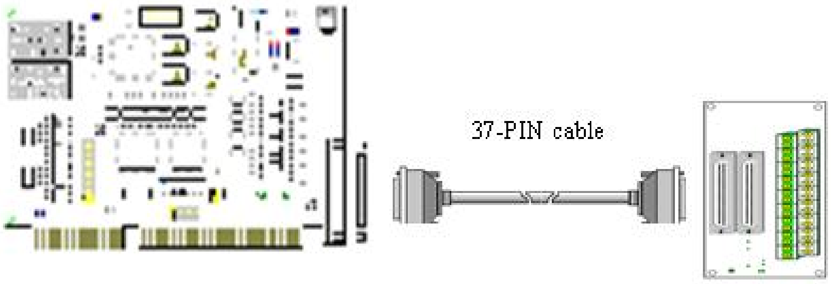


Figure 2.6

2.7.3 DB-8125

The DB-8125 is a general-purpose screw terminal board and is designed for easy wiring. The DB-8125 uses a single DB-37 header and two 20-pin flat-cable headers.

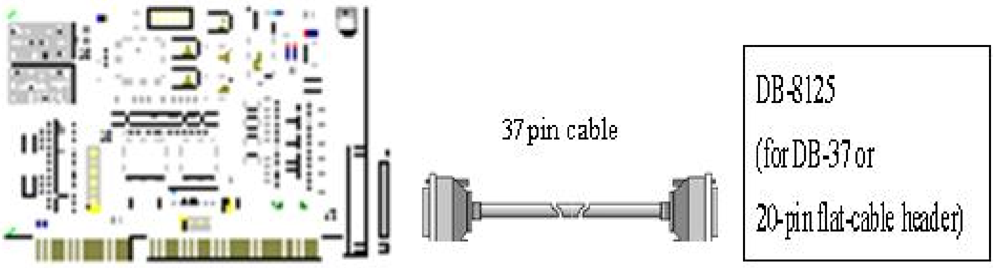


Figure 2.7

2.7.4 ADP-37/PCI & ADP-50/PCI

The ADP-37/PCI and ADP-50/PCI is an extender for a 50-pin header. One side of the ADP-37/PCI and ADP-50/PCI can be connected to a 50-pin header and the other side can be mounted on the PC chassis, as shown in the following figure. Note that the ADP-37/PCI is a 50-pin header to DB-37 extender and the ADP-50/PCI is a 50-pin header to a 50-pin header extender.

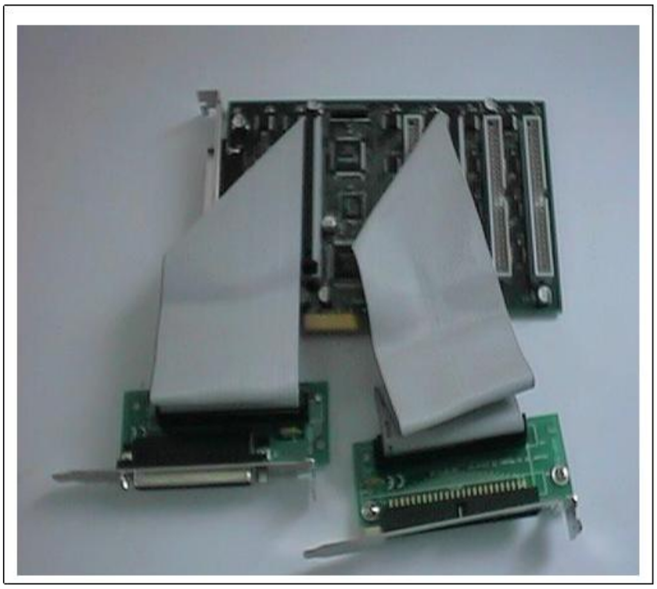


Figure 2.8

2.7.5 DB-24P, DB-24PD Isolated Input Board

The DB-24P is a 24-channel isolated digital input daughter board. The optically isolated inputs of the DB-24P consist of a bi-directional optocoupler with a resistor for current sensing. The DB-24P can be used to sense DC signals from TTL levels up to 24 V, or use the DB-24P to sense a wide range of AC signals. This board can also be used to isolate the computer from large common-mode voltage, ground loops and transient voltage spikes that often occur in industrial environments, as shown in Figure 2.7. Table 2.4 is comparison of the DB-24P and DB-24PD.

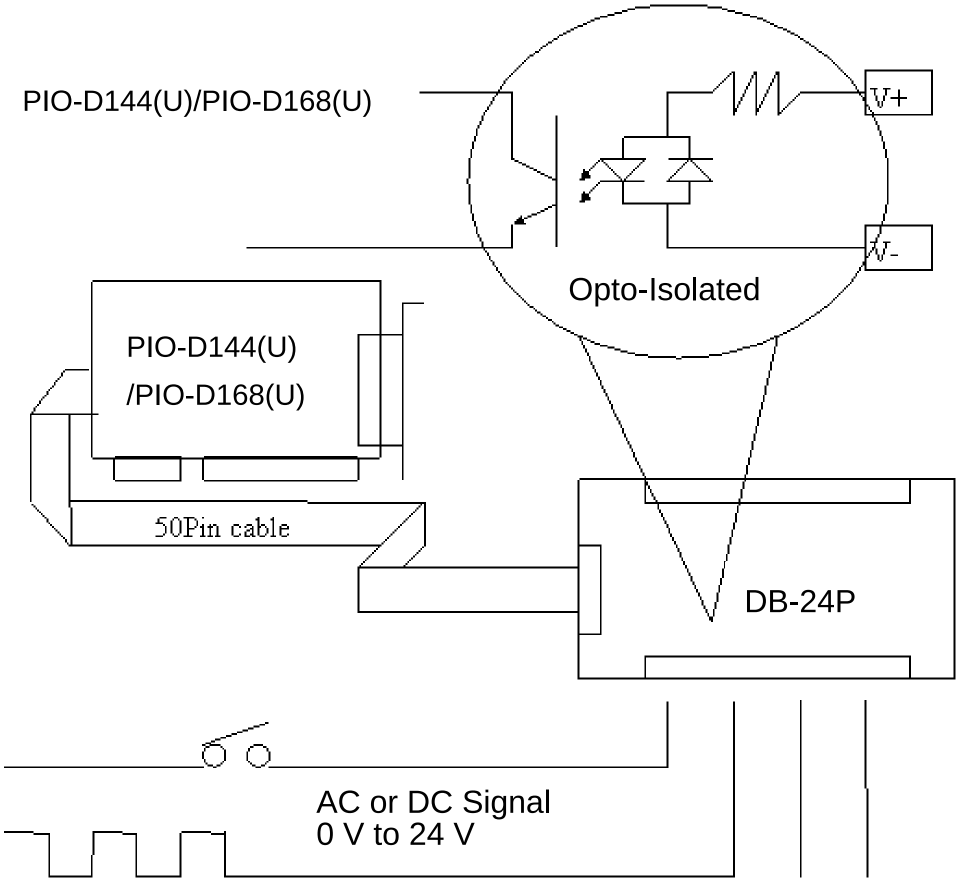


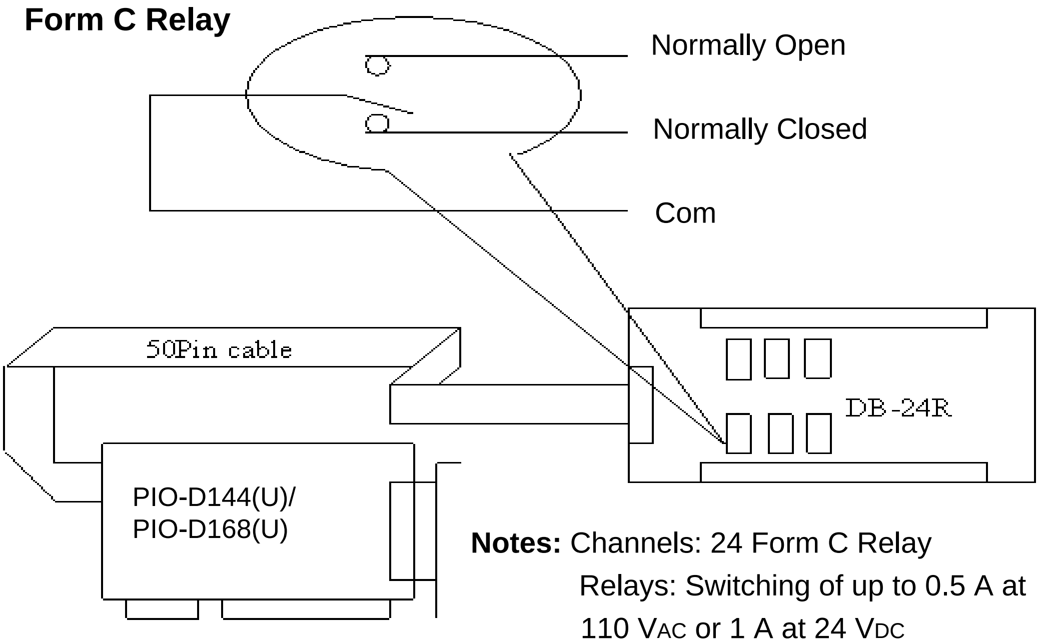
Figure 2.7

Table 2.4

	DB-24P	DB-24PD
50-pin flat-cable header	Yes	Yes
D-sub 37-pin header	No	Yes
Other specifications	Same	

2.7.6 DB-24R, DB-24RD Relay Board

The DB-24R is a 24-channel relay output board consisting of 24 Form-C relays for efficient programmable control of the load switching. The relays are powered by applying a 12 V/24 V voltage signal to the appropriate relay channel on the 50-pin flat-cable connector. There are 24 enunciator LEDs for each relay channel and the LED is light when their associated relay is activated. The control scheme is illustrated in Figure 2.8 below. Table 2.5 gives a comparison and Table 2.6 gives a description of the daughter boards used in this application.



Notes: Channels: 24 Form C Relay
Relays: Switching of up to 0.5 A at
110 V_{AC} or 1 A at 24 V_{DC}

Figure 2.8

Table 2.5

	DB-24R	DB-24RD
50-pin flat-cable header	Yes	Yes
D-sub 37-pin header	No	Yes
Other specifications	Same	

Table 2.6

DB-24R, DB-24RD	24 × Relay (120 V, 0.5 A)
DB-24PR,DB-24PRD	24 × Power Relay (250 V, 5 A)
DB-24POR	24 × Photo MOS Relay (350 V, 01 A)
DB-24SSR	24 × SSR (250 V_{AC}, 4 A)
DB-24C	24 × O.C. (30 V, 100 mA)
DB-16P8R	16 × Relay (120 V, 0.5 A) + 8 × isolated input

2.7.7 DB-24PR, DB-24POR, DB-24C

Table 2.7

DB-24PR	24 × Power relay, 5 A/250 V
DB-24POR	24 × Photo MOS relay, 0.1 A/350 V_{AC}
DB-24C	24 × Open Collector, 100 mA per channel, 30 V max.

The DB-24PR is a 24-channel power relay output board consisting of 8 Form-C and 16 Form-A electromechanical relays for efficient programmable control of the load switching. The contact of each relay can allow a 5 A current load at 250 V_{AC}/30 V_{DC}. The relay is powered by applying a 5 voltage signal to the associated relay channel on the 20-pin flat-cable connector (only uses 16 relays) or 50-pin flat-cable connector (OPTO-22 compatible, for DIO-24 series). 24 enunciator LEDs for indicating the status of for each relay and the corresponding LED is light when their associated relay is activated. To avoid overloading your PC's power supply, this board needs a +12 V_{DC} or +24 V_{DC} external power supply, as shown in Figure 2.9.

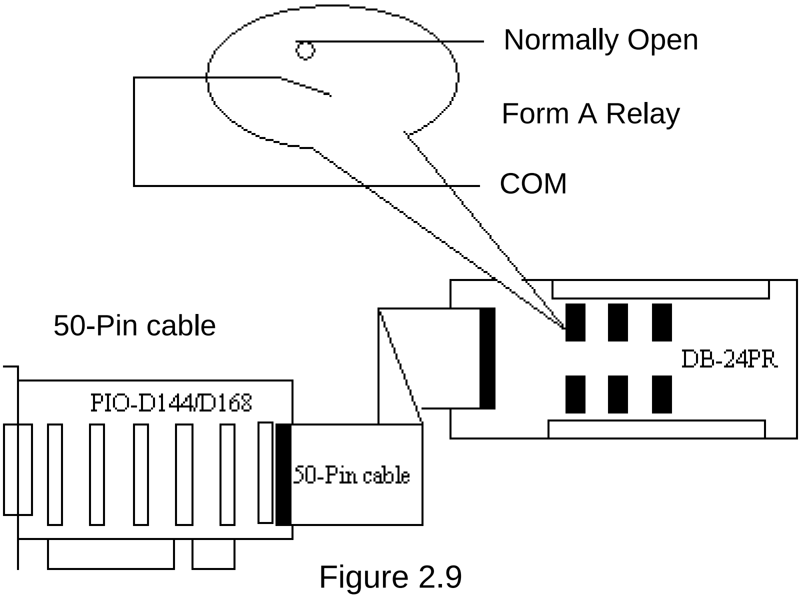


Figure 2.9

Notes:

1. **A 50-Pin connector (OPTO-22 compatible) is used, for DIO-24 / 48 / 144, PIO-D144, PIO-D96, PIO-D56, PIO-D48, PIO-D24, PIO-D168**
2. **A 20-Pin connector for 16-channel digital output is used for A-82X, A-62X, DIO-64, ISO-DA16 / DA8,**
3. **Channels: 16 Form A Relay, 8 Form C Relay.**
4. **Relays: Switching of up to 5 A at 110 V_{AC} / 5 A at 30 V_{DC}.**

2.7.8 Daughter Board Comparison Table

Table 2.9 lists a comparison for the daughter board applications using PIO/PISO series cards.

Table 2.9

	20-pin flat-cable	50-pin flat-cable	D-sub 37-pin
DB-37	No	No	**Yes**
DN-37	No	No	**Yes**
ADP-37/PCI	No	**Yes**	**Yes**
ADP-50/PCI	No	**Yes**	No
DB-24P	No	**Yes**	No
DB-24PD	No	**Yes**	**Yes**
DB-16P8R	No	**Yes**	**Yes**
DB-24R	No	**Yes**	No
DB-24RD	No	**Yes**	**Yes**
DB-24C	**Yes**	**Yes**	**Yes**
DB-24PRD	No	**Yes**	**Yes**
DB-24POR	**Yes**	**Yes**	**Yes**
DB-24SSR	No	**Yes**	**Yes**

3. I/O Control Register

3.1 How to Find the I/O Address

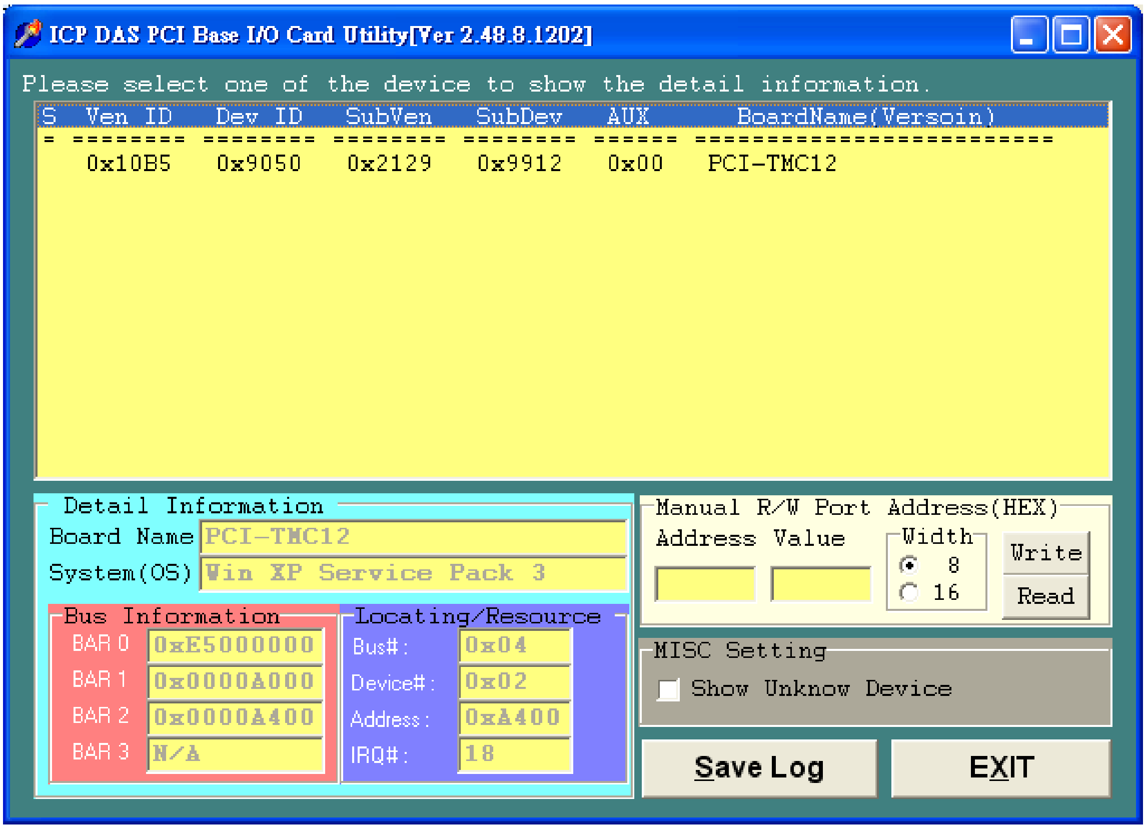
The Plug & Play BIOS will assign an appropriate I/O address to each PIO/PISO series card during the power-on stage. The fixed IDs of the PIO-D144 and PIO-D168 series cards are as following:

For PIO-D144/D144U			
Rev 1.0~Rev 3.0		**Rev 4.0 or above**	
Vendor ID	0xE159	Vendor ID	0xE159
Device ID	0x0002	Device ID	0x0001
Sub-vendor ID	0x80	Sub-vendor ID	0x5C80 0x1C80
Sub-device ID	0x01	Sub-device ID	0x01
Sub-aux ID	0x00	Sub-aux ID	0x00

For PIO-D168/D168A/D168U			
PIO-D168A		**PIO-D168(U)**	
Vendor ID	0xE159	Vendor ID	0xE159
Device ID	0x0002	Device ID	0x0001
Sub-vendor ID	0x80	Sub-vendor ID	0x9880
Sub-device ID	0x01	Sub-device ID	0x01
Sub-aux ID	0x50	Sub-aux ID	0x50

The PIO_PISO.EXE utility program will detect and present all information for PIO/PISO cards installed in the PC, as shown in the following figure. Details of how to identify the PIO series cards of ICPDAS data acquisition boards based on the **Sub-vendor**, **Sub-device** and **Sub-Aux ID** are given in Table 3-1.

The PIO_PISO.exe utility is located on the CD as below and is useful for all PIO/PISO series cards.
CD:\NAPDOS\PCI\Utility\Win32\PIO_PISO\
http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/utility/win32/pio_piso/



```
Please select one of the device to show the detail information.
S  Ven ID    Dev ID    SubVen    SubDev    AUX       BoardName(Versoin)
= ========  ========  ========  ========  ======  =========================
   0x10B5    0x9050    0x2129    0x9912    0x00    PCI-TMC12
```

Detail Information

Board Name: PCI-TMC12
System(OS): Win XP Service Pack 3

Bus Information		Locating/Resource	
BAR 0	0xE5000000	Bus# :	0x04
BAR 1	0x0000A000	Device# :	0x02
BAR 2	0x0000A400	Address :	0xA400
BAR 3	N/A	IRQ# :	18

Manual R/W Port Address(HEX)
Address Value Width ● 8 ○ 16

MISC Setting
□ Show Unknow Device

Figure 3.1

Table 3-1

PIO/PISO series card	Description	Sub_sendor ID	Sub_device ID	Sub_aux ID
PIO-D168(U)	168 * DIO	9880	01	50
PIO-D168A	168 * DIO	80	01	50
PIO-D144(REV4.0)	144 * D/I/O	80 (5C80)	01	00
PIO-D96	96 * D/I/O	80	01	10
PIO-D64 (REV2.0)	64 * D/I/O	80 (4080)	01	20
PIO-D56	24 * D/I/O + 16 * D/I+16*D/O	80	01	40
PIO-D48	48 * D/I/O	80	01	30
PIO-D24	24 * D/I/O	80	01	40
PIO-821	Multi-function	80	03	10
PIO-DA16	16 * D/A	80	04	00
PIO-DA8	8 * D/A	80	04	00
PIO-DA4	4 * D/A	80	04	00
PISO-C64	64 * isolated D/O (**Current sinking**)	80	08	00
PISO-A64	64 * isolated D/O **(Current sourcing)**	80	08	50
PISO-P64	64 * isolated D/I	80	08	10
PISO-P32C32	32 * isolated D/O (**Current sinking**) + 32 * isolated D/I	80	08	20
PISO-P32A32	32 * isolated DO (**Current sourcing**) + 32 * isolated D/I	80	08	70
PISO-P8R8	8 * isolated D/I + 8 * 220 V relay	80	08	30
PISO-P8SSR8AC	8 * isolated D/I + 8 * SSR /AC	80	08	30
PISO-P8SSR8DC	8 * isolated D/I + 8 * SSR /DC	80	08	30
PISO-730	16 * DI + 16 * D/O +16 * isolated D/I + 16 * isolated D/O (**Current sinking**)	80	08	40
PISO-730A	16 * DI + 16*D/O + 16 * isolated D/I + 16 * isolated D/O (**Current sourcing**)	80	08	80
PISO-813	32 * isolated A/D	80	0A	00
PISO-DA2	2 * isolated D/A	80	0B	00

! **Note:**
If the board is a different version, it may have different Sub IDs, but no matter which version of the board you select, the same function calls are provided.

3.2 The Assignment of I/O Addresses

The Plug & Play BIOS will assign an appropriate I/O address to the PIO/PISO series card. If there is only one PIO/PISO board, the board will be identified as card_0. If there are two PIO/PISO boards in the system, it is very difficult to identify which board is card_0. The software driver can support a maximum of 16 boards. Therefore, the user can install 16 PIO/PSIO series cards in one PC system. Details of how to locate and identify card_0 and card_1 are provided below:

The simplest way to identify which card is card_0 is to use the wSlotBus and wSlotDevice functions as follows:

Step 1: Remove all PIO-D144/D168 series cards from the PC.

Step 2: Install a single PIO-D144/D168 series card into the PCI_slot1 in the PC, then run PIO_PISO.EXE and record the wSlotBus1 and wSlotDevice1 information.

Step 3: Remove all PIO-D144/D168 series cards from the PC.

Step 4: Install a single PIO-D144/D168 series cards into the PCI_slot2 in the PC, then run PIO_PISO.EXE and record the wSlotBus2 and wSlotDevice2 information.

Step 5: Repeat Steps 3 and 4 for all PCI_slots and record all wSlotBus and wSlotDevice information.

The records may be similar to the following table:

Table 3-2

PC's PCI slot	wSlotBus	wSlotDevice
Slot_1	**0**	**0x07**
Slot_2	**0**	**0x08**
Slot_3	**0**	**0x09**
Slot_4	**0**	**0x0A**
PCI-BRIDGE		
Slot_5	**1**	**0x0A**
Slot_6	**1**	**0x08**
Slot_7	**1**	**0x09**
Slot_8	**1**	**0x07**

The above procedure is used to record all the wSlotBus and wSlotDevice information for the PC. These values will be mapped to this PC's physical slots and this mapping will not be changed for any PIO/PISO cards. Therefore, this information can be used to identify the specific PIO/PISO card using the following steps:

Step 1: Using the wSlotBus and wSlotDevice information from Table 3-2, enter the board number into the PIO_GetConfigAddressSpace(…) function to get the information for a specific card information, especially wSlotBus and wSlotDevice details.

Step 2: Identify the specific PIO/PISO card by comparing the data of the wSlotBus and wSlotDevice from step 1.

! **Note that, normally, the card that is installed in slot 0 is card0 and the card installed in slot1 is card1 for PIO/PISO series cards.**

3.3 The I/O Address Map

The I/O address of the PIO/PISO series card is automatically assigned by the main board ROM BIOS. The I/O address can also be re-assigned by the user, but **It is strongly recommended that the I/O address is not changed by user. The Plug & Play BIOS will assign an appropriate I/O address to each PIO/PISO series card**. The I/O addresses of the PIO-D144/D168 series cards are as follows, and are based on the base address of each card.

Table 3-3

Address	Read	Write
wBase+0	Reserved	RESET\ control register
wBase+2	Reserved	Aux control register
wBase+3	Aux data register	Aux data register
wBase+5	Reserved	INT mask control register
wBase+7	Aux pin status register	Reserved
wBase+0x2a	Reserved	INT polarity control register
wBase+0xc0	Read D/O Readback	Write 8-bit data to the D/O port
	Read 8-bit data from the D/I port	
wBase+0xc4	Reserved	Select the active I/O port
wBase+0xc8	Reserved	I/O Port 0-5 direction control
wBase+0xcc	Reserved	I/O Port 6-11 direction control
wBase+0xd0	Reserved	I/O Port 12-17 direction control
wBase+0xd4	Reserved	I/O Port 18-20 direction control **(only for PIO-D168 series)**
wBase+0xf0	Read Card ID	Reserved

Note: Wbase+0xd4 is only for PIO-D168 series.

3.3.1 RESET\ Control Register

(Write): wBase+0

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	RESET\

When the PC is first powered-on, the RESET\ signal is in the Low-state. **This will disable all D/I/O operations.** The user has to set the RESET\ signal to the High-state before performing any D/I/O commands.

```
outp(wBase,1);        /*RESET\=High → all D/I/O operations are now enabled*/
outp(wBase,0);        /*RESET\=Low → all D/I/O operations are now disabled*/
```

3.3.2 AUX Control Register

(Write): wBase+2

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Aux7	Aux6	Aux5	Aux4	Aux3	Aux2	Aux1	Aux0

Aux?=0→ this Aux is used as a D/I

Aux?=1→ this Aux is used as a D/O

When the PC is first powered-on, all Aux signals are in the Low-state. All Aux are designed as D/I operations for all PIO/PISO series cards.

3.3.3 AUX Data Register

(Read/Write): wBase+3

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Aux7	Aux6	Aux5	Aux4	Aux3	Aux2	Aux1	Aux0

When the Aux is used as D/O operations, the output state is controlled by this register. This register is designed for use with future extensions. Therefore, do not use this register.

3.3.4 INT Mask Control Register

(Write): wBase+5

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	0	0	0	EN3	EN2	EN1	EN0

EN0=0→ Disable P2C0 of the CN1 as an interrupt signal (Default).
EN0=1→ Enable P2C0 of the CN1 as an interrupt signal

```
outp(wBase+5,0);          /* Disable interrupt                              */
outp(wBase+5,1);          /* Enable interrupt P2C0                          */
outp(wBase+5,0x0f);       /* Enable interrupt P2C0, P2C1, P2C2, and P2C3 */
```

3.3.5 Aux Status Register

(Read): wBase+7

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Aux7	Aux6	Aux5	Aux4	Aux3	Aux2	Aux1	Aux0

Aux0=P2C0, Aux1=P2C1, Aux2=P2C2, Aux3=P2C3, Aux4~7=Aux-ID. Refer to the DEMO5.C file for more information. Aux0~3 are used as interrupt sources. The interrupt service routine must read this register to identify the interrupt source. Refer to Sec. 2.6 for more information.

3.3.6 Interrupt Polarity Control Register

(Write): wBase+0x2a

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	0	0	0	INV3	INV2	INV1	INV0

This register provides a function that is used to control whether the interrupt signal source is inverted or non-inverted. A detailed application example is shown below.

- INV0=1→ select the non-inverted signal from P2C0
- INV0=0→ select the inverted signal from P2C0

```
outp(wBase+0x2a,0);      /* Select the inverted input from P2C0/1/2/3      */
outp(wBase+0x2a,0x0f);   /* Select the non-inverted input from P2C0/1/2/3 */
outp(wBase+0x2a,1);      /* Select the non-inverted input from P2C0        */
                         /* Select the inverted input from P2C1/2/3        */
outp(wBase+0x2a,3);      /* Select the non-inverted input from P2C0/1      */
                         /* Select the inverted input from P2C2/3          */
```

Refer to Sec. 2.6 and the DEMO5.C file for more information.

3.3.7 Read/Write 8-bit Data Register

(Read/Write): wBase+0xc0

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
D7	D6	D5	D4	D3	D2	D1	D0

The PIO-D144/D168 series contains 18/21 8-bit I/O ports, and each I/O port can be configured as either a D/I or a D/O port. The user can send/receive digital data to/from this register for digital input or output. Note that all ports are set as D/I ports when the PC is first powered-on.

```
outp(wBase+0xc0,Val);    /* Write to a D/O port                         */
Val=inp(wBase+0xc0);     /* Read from a D/I port or read a D/O Readback */
```

! **Note: Ensure that the I/O port configuration is set to either D/I or D/O before attempting to read/write from the to data register. (Refer to sec.3.3.9)**

3.3.8 Active I/O Port Control Register

(Read/Write): wBase+0xc4

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
D7	D6	D5	D4	D3	D2	D1	D0

The PIO-D144/D168 series contains 18/21 8-bit I/O ports, but only one I/O port can be activated at a time. The following example is a demonstration of how to activate the port of the PIO series card.

```
outp(wBase+0xc4,0);       /*  I/O Port0 is now active   */
outp(wBase+0xc4,1);       /*  I/O Port1 is now active   */
outp(wBase+0xc4,17);      /*  I/O Port17 is now active */
outp(wBase+0xc4,20);      /*  I/O Port20 is now active */
```

3.3.9 I/O Selection Control Register

(Write): wBase+0xc8

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	0	Port5	Port4	Port3	Port2	Port1	Port0

(Write): wBase+0xcc

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	0	Port11	Port10	Port9	Port8	Port7	Port6

(Write): wBase+0xd0

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	0	Port17	Port16	Port15	Port14	Port13	Port12

(Write): wBase+0xd4

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	0	0	0	0	Port20	Port19	Port18

Port?=1→ This port is used as a D/I port
Port?=0→ This port is used as a D/O port

These registers provide the function for configuring the digital input/output ports of the PIO/PISO series cards. Each I/O port can be programmed as either a D/I or a D/O port. Note that all ports are set as D/I ports when the PC is first powered-on. The location of the I/O ports and a configuration example is presented below.

Table 3-4

PIO-D144(U)/D168(U) connector	PA0 ~ PA7	PB0 ~ PB7	PC0 ~ PC7
CN1	Port0	Port1	Port2
CN2	Port3	Port4	Port5
CN3	Port6	Port7	Port8
CN4	Port9	Port10	Port11
CN5	Port12	Port13	Port14
CN6	Port15	Port16	Port17
CN7 (PIO-D168(U) Only)	Port18	Port19	Port20

```
outportb(wBase+0xc8,0);        /* Port0 to Port5 are all D/O ports    */
outportb(wBase+0xcc,0x3f);     /* Port6 to Port11 are all D/I ports   */
outportb(wBase+0xd0,0x38);     /* Port12 to Port14 are all D/O ports  */
                               /* Port15 to Port17 are all D/I ports  */
```

3.3.10 Read Card ID Register

(Read): wBase+0xf0

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	0	0	0	ID3	ID2	ID1	ID0

```
wCardID = inportb(wBase+0xF0) ;     /* Read Card ID   */
```

! **Note: The Card ID function is only supported by the PIO-D144U and PIO-D168U model (Ver1.0 or above)**

4. Software Installation

The PIO-D144 and PIO-D168 series can be used in DOS and Windows 98/ME/NT/2K and 32-bit/64-bit Windows XP/2003/Vista/7. The recommended installation procedure for windows is given in Sec. 4.1 ~ 4.2. Or refer to Quick Start Guide (CD:\NAPDOS\PCI\PIO-DIO\Manual\QuickStart\).
http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/pio-dio/manual/quickstart/

4.1 Software Installing Procedure

- UniDAQ SDK driver (32-bit/64-bit Windows XP/2003/Vista/7):

Step 1: Insert the companion CD into the CD-ROM drive and after a few seconds the installation program should start automatically. If it doesn't start automatically for some reason, double-click the **AUTO32.EXE** file in the **NAPDOS** folder on this CD.

Step 2: Click the item: "**PCI Bus DAQ Card**".

Step 3: Click the item: "**UniDAQ**".

Step 4: Click the item: "**DLL for Windows 2000 and XP/2003/Vista 32-bit**".

Step 5: Double-Click "**UniDAQ_Win_Setup_x.x.x.x_xxxx.exe**" file in the **Driver** folder.

- Windows driver (Windows 98/NT/2K and 32-bit Windows XP/2003/Vista/7):

Step 1: Insert the companion CD into the CD-ROM drive and after a few seconds the installation program should start automatically. If it doesn't start automatically for some reason, double-click the **AUTO32.EXE** file in the **NAPDOS** folder on this CD.

Step 2: Click the item: "**PCI Bus DAQ Card**".

Step 3: Click the item: "**PIO-DIO**".

Step 4: Click the item "**DLL and OCX for Windows 98/NT/2K/XP/2003**".

Step 5: Double-Click "**PIO_DIO_Win_vxxxx.exe**" file in the **Driver** folder.

The setup program will then start the driver installation and copy the relevant files to the specified directory and register the driver on your computer. The directory where the drive is stoned is different for different windows versions, as shown below.

- **Windows 64-bit Windows XP/2003/Vista/7:**

The UniDAQ.DLL file will be copied into the C:\WINNT\SYSTEM32 folder
The NAPWNT.SYS and UniDAQ.SYS files will be copied into the
C:\WINNT\SYSTEM32\DRIVERS folder

! **For more detailed UniDAQ.DLL function information, please refer to UniDAQ SDK user manual** (CD:\NAPDOS\PCI\UniDAQ\Manual\)**.**
http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/unidaq/maunal/

- **Windows NT/2K and 32-bit Windows XP/2003/Vista/7:**

The PIODIO.DLL file will be copied into the C:\WINNT\SYSTEM32 folder
The NAPWNT.SYS and PIO.SYS files will be copied into the
C:\WINNT\SYSTEM32\DRIVERS folder

- **Windows 95/98/ME:**

The PIODIO.DLL and PIODIO.Vxd files will be copied into the
C:\Windows\SYSTEM folder

! **For more detailed PIODIO.DLL function information, please refer to "PIO-DIO DLL Software Manual.pdf**(CD:\NAPDOS\PCI\PIO-DIO\Manual\)**".**
http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/pio-dio/manual/

4.2 PnP Driver Installation

Power off the computer and install the PIO-D144 and PIO-D168 series cards. Turn on the computer and Windows 98/Me/2K and 32-bit/64-bit Windows XP/2003/Vista/7 should automatically defect the new PCI device(s) and then ask for the location of the driver files for the hardware. If a problem is encountered during installation, refer to the PnPinstall.pdf file for more information.

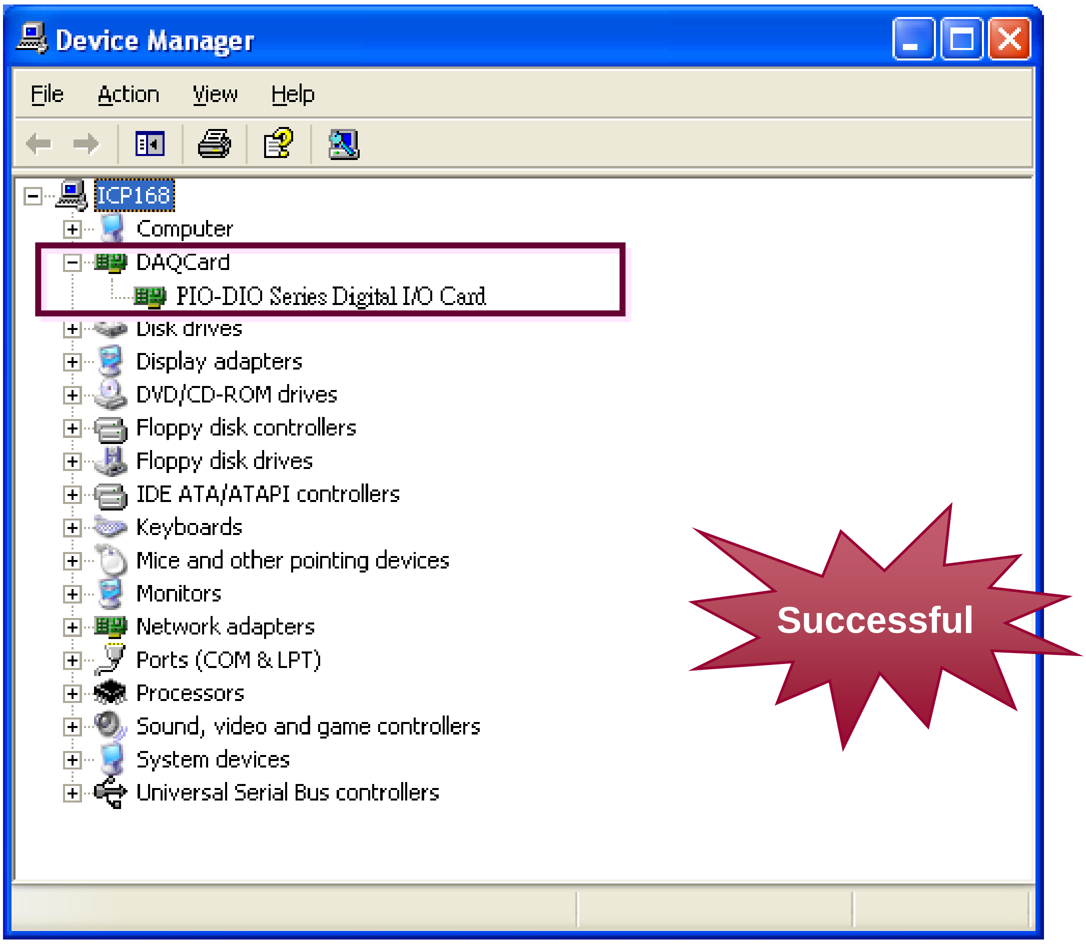
4.3 Confirm the Successful Installation

Make sure the PIO-D144 and PIO-D168 series card installed are correct on the computer as follows:

Step 1: Select **"Start"** → **"Control Panel"** and then double click the **"System"** icon on Windows.

Step 2: Click the **"Hardware"** tab and then click the **"Device Manager"** button.

Step 3: Check the PIO-D144 or PIO-D168 series card which listed correctly or not, as illustrated below.



5. Demo Programs for Windows

None of the demo programs will function correctly if the DLL driver is not properly installed. During the DLL driver installation process, the Install Shield software will register the correct kernel driver to the operating system and copy the DLL driver and demo programs to the correct position based on the driver software package you have selected (Win98/ME/NT/2K and 32-bit Windows XP/2003/Vista/7). After the drivers are installed, the relevant demo programs, development libraries and declaration header files for the different development environments will be available in the following locations.

The demo program is contained in:

CD:\NAPDOS\PCI\PIO-DIO\DLL_OCX\Demo\
http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/pio-dio/dll_ocx/demo/

- BCB 4 → for Borland C^{++} Builder 4
  PIODIO.H → Header files
  PIODIO.LIB → Linkage library for BCB only

- Delphi4 → for Delphi 4
  PIODIO.PAS → Declaration files

- VB6 → for Visual Basic 6
  PIODIO.BAS → Declaration files

- VC6 → for Visual C^{++} 6
  PIODIO.H → Header files
  PIODIO.LIB → Linkage library for VC only

- VB.NET2005 → for VB.NET2005
  PIODIO.vb → Visual Basic Source files

- CSharp2005 → for C#.NET2005
  PIODIO.cs → Visual C# Source files

A list of available demo programs is as follows:
DO: D/O for CN1
DIO: D/I/O for CN5 and CN6
DIO2: D/I/O for all Ports
INT: Interrupt for P2C0

Appendix A. Related DOS Software

A1. Where is the related software

The related DOS software and demos are located on the CD as below:

CD:\NAPDOS\PCI\PIO-DIO\dos\
http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/pio-dio/dos/

- TC*.* → for Turbo C 2.xx or above
  TC\LIB\*.* → for TC library
  TC\DEMO\*.* → for TC demo programs
  TC\DIAG\*.* → for TC diagnostic programs

  TC\LIB\PIO.H → TC declaration file
  TC\\LIB\TCPIO_L.LIB → TC large model library file
  TC\\LIB\TCPIO_H.LIB → TC huge model library file

- MSC*.* → for MSC 5.xx or above
  MSC\LIB\PIO.H → MSC declaration file
  MSC\\LIB\MSCPIO_L.LIB → MSC large model library file
  MSC\\LIB\MSCPIO_H.LIB → MSC huge model library file

- BC*.* → for BC 3.xx or above
  BC\LIB\PIO.H → BC declaration file
  BC\\LIB\BCPIO_L.LIB → BC large model library file
  BC\\LIB\BCPIO_H.LIB → BC huge model library file

The list of demo programs:
DEMO1: D/O for CN1
DEMO2: D/O for CN1 ~ CN6
DEMO3: Interrupt of P2C0 (Initial low and active high)
DEMO4: Interrupt of P2C0 (Initial high and active low)
DEMO5: 4 interrupt sources

A2. DOS LIB Functions

A2-1. ErrorCode and ErrorString Code Table

Table A.1

Error Code	Error ID	Error String
0	NoError	OK (No error)
1	Driver HandleError	Error opening the device driver
2	DriverCallError	An error occurred while calling the driver functions
3	FindBoardError	Can't find the board on the system
4	TimeOut	Timeout
5	ExeedBoardNumber	Invalid board number (Valid range: 0 to TotalBoards -1)
6	NotFoundBoard	Can't detect the board on the system

A2-2. PIO_DriverInit

- **Description:**

  This function is used to detect all PIO/PISO series card in the system and is implemented based on the PCI Plug & Play mechanism. The function will locate/identify all PIO/PISO series cards installed in this system and save the resource information in the library.

- **Syntax:**

  WORD **PIO_DriverInit**(WORD ***wBoards**, WORD **wSubVendorID**,
  WORD **wSubDeviceID**, WORD **wSubAuxID**)

- **Parameters:**

WBoards	[Output]	The number of boards found in this PC
wSubVendorID	[Input]	SubVendor ID of the board
wSubDeviceID	[Input]	SubDevice ID of the board
wSubAuxID	[Input]	SubAux ID of the board

- **Returns:**

  Refer to "Table A.1".

A2-3. PIO_GetConfigAddressSpace

- **Description:**

  This function can be used to save the resource information all PIO/PISO cards installed in the system. The application program can then control all the functions of the PIO/PISO series card directly.

- **Syntax:**

  WORD **PIO_GetConfigAddressSpace**(**wBoardNo**,*__wBase__,*__wIrq__, **wSubVendor**, *__wSubDevice__, *__wSubAux__, *__wSlotBus__, *__wSlotDevice__)

- **Parameters:**

wBoardNo	[Input]	The board number
wBase	[Output]	The base address of the board
wIrq	[Output]	The IRQ number that the board using
wSubVendor	[Output]	Sub Vendor ID
wSubDevice	[Output]	Sub Device ID
wSubAux	[Output]	Sub Aux ID
wSlotBus	[Output]	Slot Bus number
wSlotDevice	[Output]	Slot Device ID

- **Returns:**

  Refer to "Table A.1".

A2-4. PIO_GetDriverVersion

- **Description:**

  This function is used to obtain the version number of PIODIO driver.

- **Syntax:**

  WORD **PIO_GetDriverVersion**(WORD *__wDriverVersion__)

- **Parameters:**

wDriverVersion	[Output]	wDriverVersion address

- **Returns:**

  Refer to "Table A.1".

A2-5. ShowPIOPISO

- **Description:**

  This function can be used to display a text string indicating the special Sub_ID. This text string is the same as that defined in PIO.H.

- **Syntax:**

  WORD **ShowPIOPISO**(**wSubVendor**, **wSubDevice**, **wSubAux**)

- **Parameters:**

wSubVendor	[Input]	SubVendor ID of the board
wSubDevice	[Input]	SubDevice ID of the board
wSubAux	[Input]	SubAux ID of the board

- **Returns:**

  Refer to "Table A.1".