ARTISAN®
TECHNOLOGY GROUP

Your **definitive** source for quality pre-owned equipment.

Artisan Technology Group
(217) 352-9330 | sales@artisantg.com | artisantg.com

Full-service, independent repair center
with experienced engineers and technicians on staff.

We buy your excess, underutilized, and idle equipment
along with credit for buybacks and trade-ins.

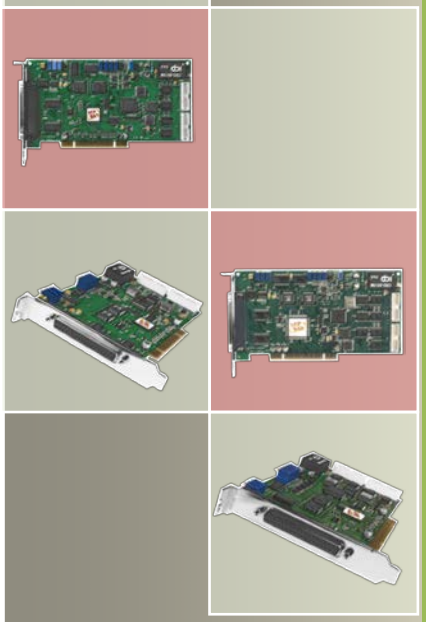
Custom engineering
so your equipment works exactly as you specify.

- Critical and expedited services
- In stock / Ready-to-ship
- Leasing / Rentals / Demos
- ITAR-certified secure asset solutions

Expert team | Trust guarantee | 100% satisfaction

All trademarks, brand names, and brands appearing herein are the property of their respective owners.

Find the *ICP DAS PCI-1202LU* at our website: *Click HERE*



PCI-1202/1602/180x

Series Card User Manual

Multi-Function Boards Version 4.9, Jan. 2017

SUPPORTS

Board includes PEX-1202L, PEX-1202H, PCI-1202L, PCI-1202H, PCI-1202LU, PCI-1202HU, PCI-1602, PCI-602F, PCI-1602U, PCI-1602FU, PCI-1800L, PCI-1800H, PCI-1800LU, PCI-1800HU, PCI-1802L, PCI-1802H, PCI-1802LU and PCI-1802HU.

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

Copyright © 2013 by ICP DAS. All rights are reserved.

TRADEMARK

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

CONTACT US

If you have any question, please feel to contact us. We will give you quick response within 2 workdays.

Email: service@icpdas.com, service.icpdas@gmail.com

TABLE OF CONTENTS

PACKING LIST **5**

RELATED INFORMATION **5**

1. INTRODUCTION **6**

1.1 FEATURES 8

1.2 THE BLOCK DIAGRAM 11

1.3 SPECIFICATIONS 12

1.3.1 PEX-1202/PCI-1202 Series *12*

1.3.2 PCI-1602 Series *14*

1.3.3 PCI-1800/PCI-1802 Series *16*

2. HARDWARE CONFIGURATION **18**

2.1 BOARD LAYOUT 18

2.2 JUMPER SETTING 21

2.2.1 JP1: A/D Input Type Selection *21*

2.2.2 J1: D/A Reference Voltage Selection *22*

2.2.3 D/I Port Setting (Pull-High/Low) *22*

2.3 CARD ID SWITCH 23

2.4 ANALOG INPUT SIGNAL CONNECTION 24

2.5 PIN ASSIGNMENTS 29

3. HARDWARE INSTALLATION **32**

4. SOFTWARE INSTALLATION **36**

4.1 DRIVER INSTALLING PROCEDURE 36

4.2 PNP DRIVER INSTALLATION 38

4.3 CONFIRM THE SUCCESSFUL INSTALLATION 41

5. TESTING PCI-1202/1602/180X CARD **42**

5.1 SELF-TEST WIRING 42

5.1.1 DIO Test Wiring *42*

5.1.2 Analog Input Test Wiring *43*

5.1.3 Analog Output Test Wiring *44*

5.2 EXECUTE THE TEST PROGRAM 46

6. I/O CONTROL REGISTER 50

6.1 How to Find the I/O Address 50
6.2 The Assignment of I/O Address 53
6.3 The I/O Address Map 54
6.4 BAR1: Timer Control 55
6.5 BAR2: Control Register 58
6.5.1 The Control Register 58
6.5.2 The Status Register 79
6.5.3 The A/D Software Trigger Register 80
6.6 BAR3: DI/DO Register 81
6.6.1 Digital Output/Digital Input 81
6.6.2 Card ID Register 82
6.7 BAR4: A/D and D/A Register 83

7. A/D CONVERSION OPERATION 86

7.1 The Configuration Code Table 86
7.2 The Unipolar/Bipolar 87
7.3 The Input Signal Range 87
7.4 The Settling Time 88
7.5 When to Delay the Settling Time 88
7.6 The AD Conversion Mode 89
7.7 The Fixed-channel Mode AD Conversion 91
7.8 The MagicScan Mode AD Conversion 92
7.8.1 The MagicScan Circular_Scan_Queue 93
7.8.2 The Digital Filter of MagicScan 94
7.8.3 The Digital Filter of MagicScan 94
7.8.4 The High/Low Alarm of MagicScan 95
7.8.5 The MagicScan Function 95
7.8.6 The MagicScan Thread 98

8. M_FUNCTION 100

8.1 Introduction 101

9. CONTINUOUS CAPTURE FUNCTION 105

9.1 General Purpose Functions 105
9.2 Functions for saving Data in PC Memory 109

10. CALIBRATION 110

10.1 A/D Calibration 110

10.2 D/A CALIBRATION 112

11. DEMO PROGRAM 114

11.1 DEMO PROGRAM FOR WINDOWS 114

11.2 DIAGNOSTIC PROGRAM 117

11.2.1 Power-on Plug&Play Test 117

11.2.2 Driver Plug&Play Test 117

11.2.3 D/O Test 118

11.2.4 D/A Test 118

11.2.5 A/D Test 118

12. PERFORMANCE EVALUATION 119

APPENDIX: DAUGHTER BOARD 120

A1. DB-37 and DN-37 120

A2. DB-8125 120

A3. DB-8225 121

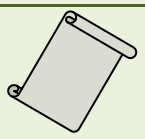
A4. DB-16P Isolated Input Board 121

A5. DB-16R Relay Board 122

A6. DB-24PR Power Relay Board 123

Packing List

The shipping package includes the following items:

	One multi-function card as follows: PEX-1202 series: PEX-1202L/ PEX-1202H PCI-1202 series: PCI-1202L/ PC-1202H/ PCI-1202LU/ PCI-1202HU PCI-1602 series: PCI-1602/ PCI-1602F/ PCI-1602U/ PCI-1602FU PCI-1800 series: PCI-1800L/ PCI-1800H/ PCI-1800LU/ PCI-1800HU PCI-1802 series: PCI-1802L/ PCI-1802H/ PCI-1802LU/ PCI-1802HU
	One printed Quick Start Guide
	One software utility CD
	One CA-4002 D-Sub Connect

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

Related Information

Product Page:

http://www.icpdas.com/root/product/solutions/pc_based_io_board/pci/pci-1202.html
http://www.icpdas.com/root/product/solutions/pc_based_io_board/pci/pci-1602.html
http://www.icpdas.com/root/product/solutions/pc_based_io_board/pci/pci-1800.html

Documentation and Software for UniDAQ SDK:

CD:\NAPDOS\PCI\UniDAQ\
http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/unidaq/

1. Introduction

Comparison Table

Model Name	Bus	D/A Channel	A/D Channel	A/D Sampling Rate	Software Trigger (Polling)	Pacer Scan (Magic Scan)	Post-trigger Pre-trigger Middle-trigger	FIFO Size
PEX-1202L	PCI Express	12-bit, 2-ch	12-bit, 32 S.E./ 16 Diff.	110 kS./s	✓	✓	✓	1 k
PCI-1202LU	Universal PCI							
PEX-1202H	PCI Express	12-bit, 2-ch	12-bit, 32 S.E./ 16 Diff.	44 kS./s	✓	✓	✓	1 k
PCI-1202HU	Universal PCI							
PCI-1602U	Universal PCI	12-bit, 2-ch	16-bit, 32 S.E./ 16 Diff.	100 kS./s	✓	✓	✓	8 k
PCI-1602FU	Universal PCI	12-bit, 2-ch	16-bit, 32 S.E./ 16 Diff.	200 kS./s	✓	✓	✓	8 k
PCI-1800LU	Universal PCI	12-bit, 2-ch	12-bit, 16 S.E./ 8 Diff.	330 kS./s	✓	✓	✓	1 k
PCI-1800HU	Universal PCI	12-bit, 2-ch	12-bit, 16 S.E./ 8 Diff.	44 kS./s	✓	✓	✓	1 k
PCI-1802LU	Universal PCI	12-bit, 2-ch	12-bit, 32 S.E./ 16 Diff.	330 kS./s	✓	✓	✓	8 k
PCI-1802HU	Universal PCI	12-bit, 2-ch	12-bit, 32 S.E./ 16 Diff.	44 kS./s	✓	✓	✓	8 k

General Description

The PCI-1800/1802(L/H) series are high performance, multifunction analog, digital I/O board for PC and compatible computers in a 5 V PCI slot. This series features a **continuous**, **330 k Samples/Sec.**, **gap-free** data acquisition under DOS. This family has the same features: one 12-bit 330 k AD converter, two 12-bit independent DA converters, 16-channel TTL compatible DI and 16-channel TTL compatible DO. The PCI-1800 series provides 16 single-ended or 8 differential inputs. The PCI-1802 series provides 32 single-ended or 16 differential inputs. Two DACs of this multifunction card are independent bipolar voltage output with jumper selectable voltage output range. The AD scan function of 1800 series is very amazing; we call it "**MagicScan**". It scans with two modes: **the fixed-channel mode** and **the channel-scan mode**, both modes can be up to 330 k samples per second. We also provide three trigger modes for this series: software trigger, pacer trigger and external trigger; each trigger mode uses "**MagicScan**" to perform the data acquisition. The external trigger can be programmed to one of the three trigger methods: pre-trigger, post-trigger and middle-trigger.

The PEX-1202(L/H) is new version of PCI-1202 series card they can be installed in PCI Express bus.
The PCI-1202/1800/1802(HU/LU) and PCI-1602(U/FU) are new version of PCI-1202/1800/1802(H/L) and PCI-1602(/F) they can be installed in 3.3 V, 5 V or 3.3 V/5 V Universal PCI-Bus. The PEX-1202(L/H), PCI-1202/1800/1802(H/L) and PCI-1602(/F) could be replaced with PCI-1202/1800/1802(HU/LU) and PCI-1602(U/FU) without modifying software.

The PEX/PCI-1202 series is very similar to PCI-1802 series. The different items between the PCI-1802 and **PEX/PCI-1202** are given as follows:

* A/D sampling rate is **110 k Samples/Sec**. **for PEX-1202(L) and PCI-1202(L/LU)**
* FIFO size is **1 k samples**

The PCI-1602(/U/FU) is very similar to PCI-1802(L). The different items between the PCI-1802 and **PCI-1602** are given as follows:

* A/D is **16-bit**
* A/D sampling rate is **200 k Samples/Sec**. **for PCI-1602(F/FU)**
* A/D sampling rate is **100 k Samples/Sec**. **for PCI-1602(U)**

1.1 Features

Interface:

- Supports the +5 V PCI bus for PCI-1202/1800/1802(L/H) and PCI-1602(/F) series card.
- Supports the +3.3 V/+5 V PCI bus for PCI-1202/PCI-1800/PCI-1802(LU/HU) and PCI-1602(U/FU) series card.
- Supports PCI Express x 1 for PEX-1202(L/H) series card.

Digital Input/Output:

- 16 channels TTL compatible D/I.
- 16 channels TTL compatible D/O.
- High Speed data transfer rate: refer to chapter 10.

Analog Output:

- 2 channels 12-bit DACs.
- Bipolar voltage output with ±5 V or ± 10 V by jumper selectable.
- High throughput: refer to chapter 10.
- 12-bit DAC output code for PEX-1202(L/H) and PCI-1202/1800/1802(L/H/LU/HU).

12-bit DAC Output Code

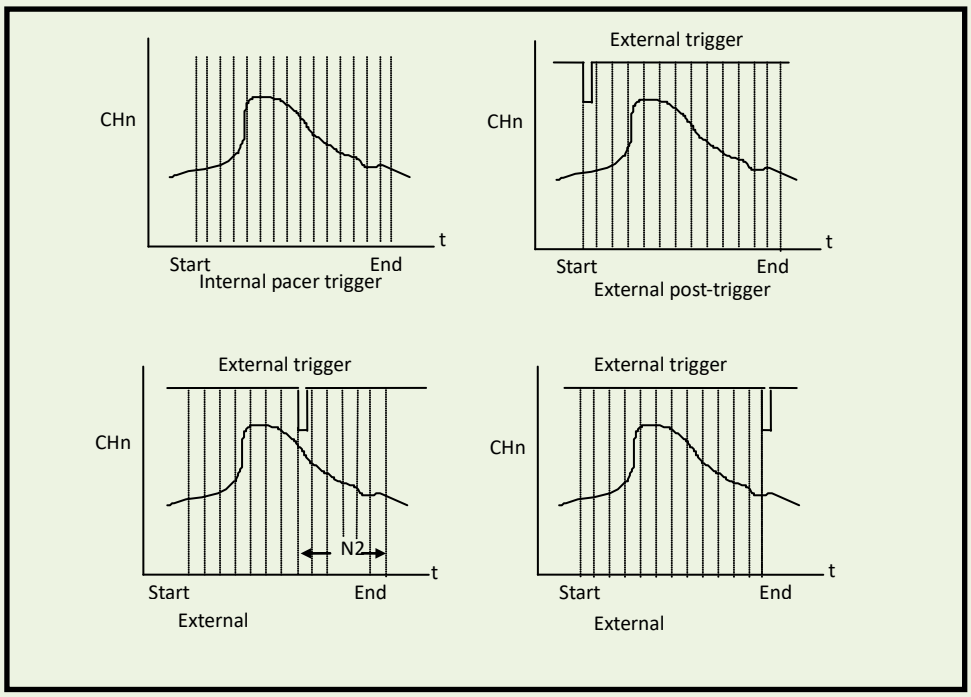
Data Input			Analog Output
MSB		LSB	
1111	1111	1111	+Vref (2047/2048)
1000	0000	0001	+Vref (1/2048)
1000	0000	0000	0 Volts
0001	1111	1111	-Vref (1/1024)
0000	0000	0000	-Vref (2048/2048)

Analog Input:

- 32 S.E./ 16 Diff. analog inputs for PEX-1202 and PCI-1202/1602/1802 series card.
- 16 S.E./8 Diff. analog inputs for PCI-1800 series card.
- A/D converter = 12-bit, 330 k Samples/Sec. for PCI-1800/1802(L/LU).
  A/D converter = 12-bit, 110 k Samples/Sec. for PEX-1202(L) and PCI-1202(L/LU).
  A/D converter = 12-bit, 44 k Samples/Sec. for PEX-1202(H) and PCI-1202/1800/1802(H/HU).
  A/D converter = 16-bit, 200 k Samples/Sec. for PCI-1602(F/FU).
  A/D converter = 16-bit, 100 k Samples/Sec. for PCI-1602(/U).

- Programmable input signal configuration.
- Provides “MagicScan” function.
- FIFO Size: 1 k samples for PEX-1202(L/H) and PCI-1202/1800(L/H/LU/HU).
  8 k samples for PCI-1602 (/F/U/FU) and PCI-1802(L/H/LU/HU).

- Three A/D trigger sources: software, Pacer and external trigger.
- Three external trigger modes: pre-trigger, middle-trigger and post-trigger.



- 12-bit ADC input voltages and output codes for PEX-1202 and PCI-1202/1800/1802 series.

12-bit ADC Input Voltages and Output Codes					
Analog Input	**Digital Output Binary Code**				**Hex Code**
	MSB			LSB	
+9.995 V	1111	1111	1111		FFF
+0 V	1000	0000	0000		800
-4.88 mV	0111	1111	1111		7FF
-10 V	0000	0000	0000		000

- 16-bit ADC input voltages and output codes for PCI-1602 series.

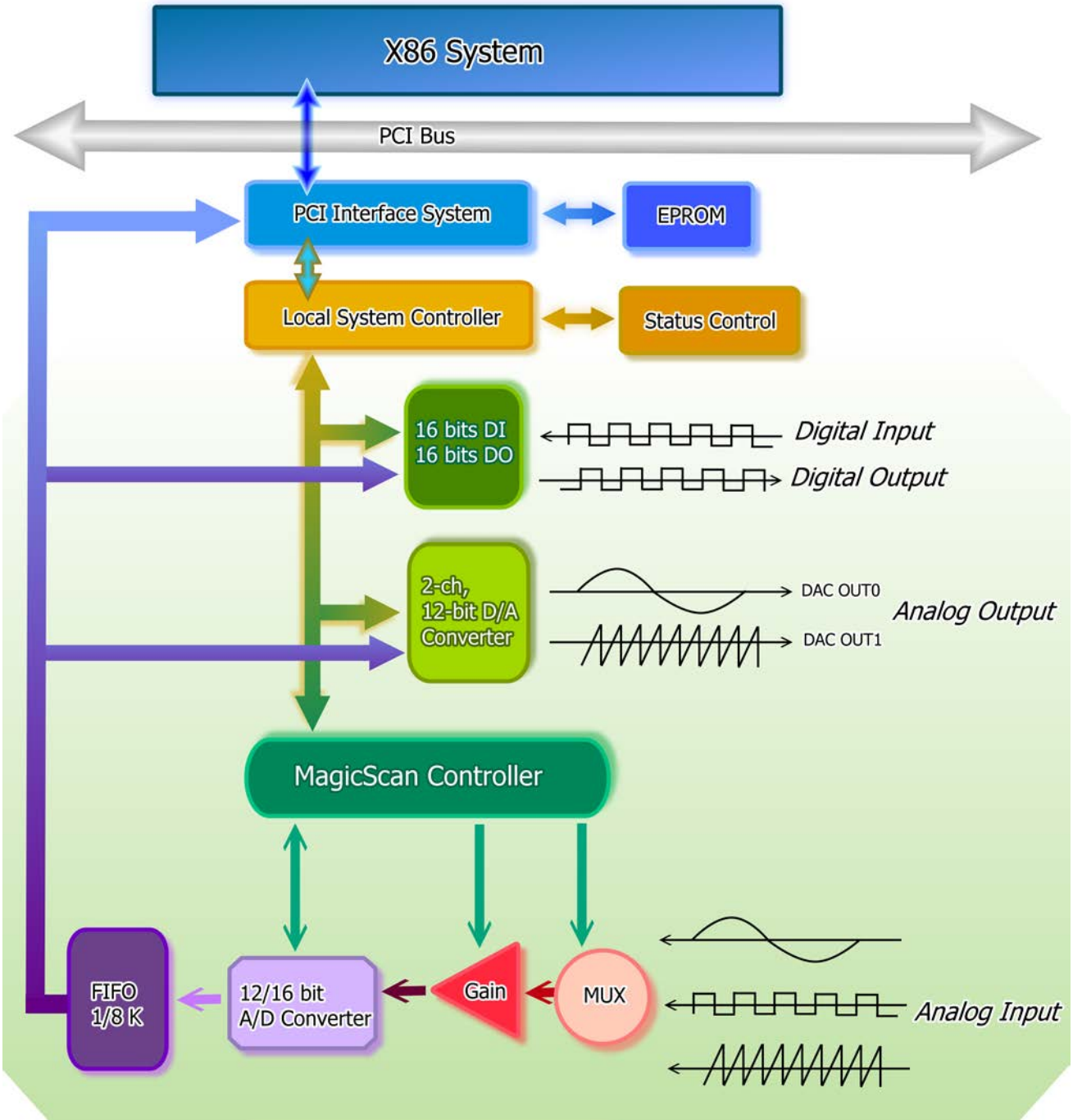
16-bit ADC Input Voltages and Output Codes					
Analog Input	**Digital Output Binary Code**				**Hex Code**
	MSB			LSB	
+9.99 V	0111	1111	1111	1111	7FFF
+0 V	0000	0000	0000	0000	0000
-305 µV	1111	1111	1111	1111	FFFF
-10 V	1000	0000	0000	0000	8000

Timer/Counter:

- Three 16-bit independent timers.
- Timer 0 is used as the internal A/D pacer trigger.
- Timer 1 is used as the external trigger.
- Timer 2 is used as the machine independent timer.

1.2 The Block Diagram

The block diagram of the PEX-1202 and PCI-1202/1602/1800/1802 series card is given as follows:



1.3 Specifications

1.3.1 PEX-1202/PCI-1202 Series

Model Name		PEX-1202L	PEX-1202H	PCI-1202LU	PCI-1202HU
Analog Input					
Channels		32 single-ended / 16 differential			
A/D Converter		12-bit, 8.5 µs conversion time			
Sampling Rate		110 kS./s	44 kS./s	110 kS./s	44 kS./s
FIFO Size		1024 samples			
Over Voltage Protection		Continuous ±35 V_{p-p}			
Input Impedance		10 MΩ/6 pF			
Trigger Modes		Software, Internal programmable pacer, External (5 V/TTL compatible)			
Data Transfer		Polling			
Accuracy		0.1 % of FSR ±1 LSB @ 25 °C, ± 10 V			
Zero Drift		± 4 ppm/°C of FSR			
Analog Output					
Channels		2-ch			
Resolution		12-bit			
Accuracy		0.06% of FSR ± 1 LSB @ 25 °C, ± 10 V			
Output Rang		Bipolar: ±5 V, ±10 V			
Output Driving		± 5 mA			
Slew Rate		8.33 V/µs			
Output Impedance		0.1 Ω max.			
Operating Mode		Software			
Digital Input					
Channels		16-ch			
Compatibility		5 V/TTL			
Input Voltage		Logic 0: 0.8 V max. / Logic 1: 2.0 V min.			
Response Speed		500 KHz		2.7 MHz (Typical)	
Digital Output					
Channels		16-ch			
Compatibility		5 V/CMOS		5 V/TTL	
Output Voltage	Logic 0	0.1 V max.		0.4 V max.	
	Logic 1	4.4 V min.		2.4 V min.	
Output Capability	Sink	6 mA @ 0.33 V		2.4 mA @ 0.8 V	
	Source	6 mA @ 4.77 V		0.8 mA @ 2.0 V	
Response Speed		500 KHz		2.7 MHz (Typical)	

Model Name	PEX-1202L	PEX-1202H	PCI-1202LU	PCI-1202HU
Timer/Counter				
Channels	3 (Independent x 1/Internal pacer x 1/External pacer x 1)			
Resolution	16-bit			
Compatibility	5 V/TTL			
Input Frequency	10 MHz max.			
Reference Clock	Internal: 8 MHz			
General				
Bus Type	PCI Express x1		3.3V/5V Universal PCI, 32-bit, 33MHz	
Data Bus	16-bit			
Card ID	Yes (4-bit)		Yes (4-bit) for Version 4.0 or above	
I/O Connector	Female DB37 x 1, Male 20-bit ribbon x 2			
Dimensions (L x W x D)	162 mm x 100 mm x 22 mm		195 mm x 97 mm x 22 mm	
Power Consumption	1300 mA @ + 3.3 V 0 mA @ +12 V		300 mA @ +5 V	
Operating Temperature	0 ~ 60 °C			
Storage Temperature	-20 ~ 70 °C			
Humidity	5 ~ 85% RH, non-condensing			

Analog Input Range								
Model Name	**PEX-1202L/PCI-1202LU (Low-Gain)**							
Gain	0.5		1	2		4		8
Bipolar (V)	± 10		± 5	± 2.5		± 1.25		± 0.625
Unipolar (V)	-		0 ~ 10	0 ~ 5		0 ~ 2.5		0 ~ 1.25
Sampling Rage Max.	110 kS./s							
Model Name	**PEX-1202H/PCI-1202HU (High-Gain)**							
Gain	0.5	1	5	10	50	100	500	1000
Bipolar (V)	± 10	± 5	± 1	± 0.5	± 0.1	± 0.05	± 0.01	± 0.005
Unipolar (V)	-	0~10	-	0 ~ 1	-	0 ~ 0.1	-	0 ~ 0.01
Sampling Rate Max.	44 kS./s				10 kS./s		1 kS./s	

1.3.2 PCI-1602 Series

Model Name	PCI-1602U	PCI-1602FU	PCI-1602	PCI-1602F
Analog Input				
Channels	32 single-ended / 16 differential			
A/D Converter	16-bit, 2 µs conversion time			
Sampling Rate	100 kS./s	200 kS./s	100 kS./s	200 kS./s
FIFO Size	8192 samples			
Over Voltage Protection	Continuous ±35 V_{p-p}			
Input Impedance	10 MΩ/6 pF			
Trigger Modes	Software, Internal programmable pacer, External (5 V/TTL compatible)			
Data Transfer	Polling			
Accuracy	0.01 % of FSR ±1 LSB @ 25 °C, ± 10 V			
Zero Drift	± 2 ppm/°C of FSR			
Analog Output				
Channels	2-ch			
Resolution	12-bit			
Accuracy	0.06% of FSR ± 1 LSB @ 25 °C, ± 10 V			
Output Rang	Bipolar: ±5 V, ±10 V			
Output Driving	± 5 mA			
Slew Rate	8.33 V/µs			
Output Impedance	0.1 Ω max.			
Operating Mode	Software			
Digital Input				
Channels	16-ch			
Compatibility	5 V/TTL			
Input Voltage	Logic 0: 0.8 V max. Logic 1: 2.0 V min.			
Response Speed	2.7 MHz (Typical)			
Digital Output				
Channels	16-ch			
Compatibility	5 V/TTL			
Output Voltage	Logic 0: 0.4 V max. Logic 1: 2.4 V min.			
Output Capability	Sink: 2.4 mA @ 0.8 V Source: 0.8 mA @ 2.0 V			
Response Speed	2.7 MHz (Typical)			

Model Name	PCI-1602U	PCI-1602FU	PCI-1602	PCI-1602F
Timer/Counter				
Channels	3 (Independent x 1/Internal pacer x 1/External pacer x 1)			
Resolution	16-bit			
Compatibility	5 V/TTL			
Input Frequency	10 MHz max.			
Reference Clock	Internal: 8 MHz			
General				
Bus Type	3.3 V/5 V Universal PCI, 32-bit, 33 MHz		5 V PCI bus, 32-bit, 33 MHz	
Data Bus	16-bit			
Card ID	Yes (4-bit)		No	
I/O Connector	Female DB37 x 1 Male 20-bit ribbon x 2			
Dimensions (L x W x D)	188 mm x 105 mm x 22 mm			
Power Consumption	350 mA @ +5 V			
Operating Temperature	0 ~ 60 °C			
Storage Temperature	-20 ~ 70 °C			
Humidity	5 ~ 85% RH, non-condensing			

Analog Input Range				
Model Name	**PCI-1602U/PCI-1602**			
Gain	1	2	4	8
Bipolar (V)	± 10	± 5	± 2.5	± 1.25
Sampling Rage Max.	100 kS./s			
Model Name	**PCI-1602FU/PCI-1602F**			
Gain	1	2	4	8
Bipolar (V)	± 10	± 5	± 2.5	± 1.25
Sampling Rate Max.	200 kS./s			

1.3.3 PCI-1800/PCI-1802 Series

Model Name	PCI-1802HU PCI-1802LU	PCI-1802H PCI-1802L	PCI-1800HU PCI-1800LU	PCI-1800H PCI-1800L
Analog Input				
Channels	32 single-ended / 16 differential		32 single-ended / 16 differential	
A/D Converter	12-bit, 3 µs conversion time			
Sampling Rate	330 kS./s for High Gain 44 kS./s for Low Gain			
FIFO Size	8192 samples		1024 samples	
Over Voltage Protection	Continuous ±35 V_{p-p}			
Input Impedance	10 MΩ/6 pF			
Trigger Modes	Software, Internal programmable pacer, External (5 V/TTL compatible)			
Data Transfer	Polling			
Accuracy	0.01 % of FSR ±1 LSB @ 25 °C, ± 10 V			
Zero Drift	± 2 ppm/°C of FSR			
Analog Output				
Channels	2-ch			
Resolution	12-bit			
Accuracy	0.06% of FSR ± 1 LSB @ 25 °C, ± 10 V			
Output Rang	Bipolar: ±5 V, ±10 V			
Output Driving	± 5 mA			
Slew Rate	8.33 V/µs			
Output Impedance	0.1 Ω max.			
Operating Mode	Software			
Digital Input				
Channels	16-ch			
Compatibility	5 V/TTL			
Input Voltage	Logic 0: 0.8 V max. Logic 1: 2.0 V min.			
Response Speed	2.7 MHz (Typical)			
Digital Output				
Channels	16-ch			
Compatibility	5 V/TTL			
Output Voltage	Logic 0: 0.4 V max. Logic 1: 2.4 V min.			
Output Capability	Sink: 2.4 mA @ 0.8 V Source: 0.8 mA @ 2.0 V			
Response Speed	2.7 MHz (Typical)			

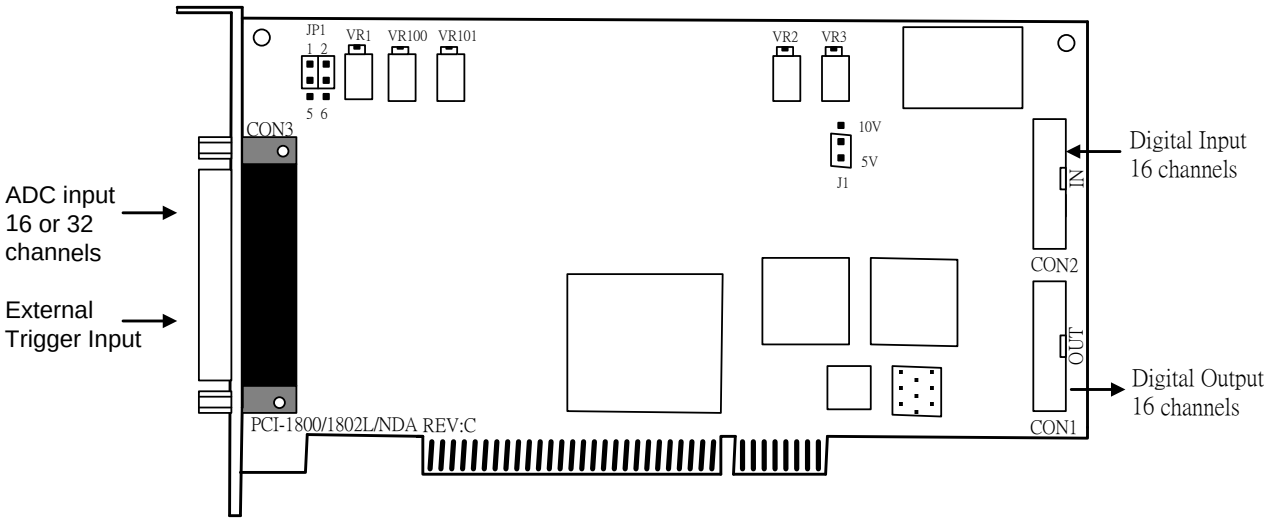
Model Name	PCI-1802HU PCI-1802LU	PCI-1802H PCI-1802L	PCI-1800HU PCI-1800LU	PCI-1800H PCI-1800L
Timer/Counter				
Channels	3 (Independent x 1/Internal pacer x 1/External pacer x 1)			
Resolution	16-bit			
Compatibility	5 V/TTL			
Input Frequency	10 MHz max.			
Reference Clock	Internal: 8 MHz			
General				
Bus Type	3.3 V/5 V Universal PCI, 32-bit, 33 MHz	5 V PCI bus, 32-bit, 33 MHz	3.3 V/5 V Universal PCI, 32-bit, 33 MHz	5 V PCI bus, 32-bit, 33 MHz
Data Bus	16-bit			
Card ID	Yes (4-bit)	No	Yes (4-bit)	No
I/O Connector	Female DB37 x 1 Male 20-bit ribbon x 2			
Dimensions (L x W x D)	200 mm x 105 mm x 22 mm			
Power Consumption	300 mA @ +5 V			
Operating Temperature	0 ~ 60 °C			
Storage Temperature	-20 ~ 70 °C			
Humidity	5 ~ 85% RH, non-condensing			

Analog Input Range								
Model Name	**PCI-1802(L/LU)/PCI-1800(L/LU) (Low-Gain)**							
Gain	0.5	1	2	4	8			
Bipolar (V)	± 10	± 5	± 2.5	± 1.25	± 0.625			
Unipolar (V)	-	0 ~ 10	0 ~ 5	0 ~ 2.5	0 ~ 1.25			
Sampling Rage Max.	330 kS./s							
Model Name	**PCI-1802(H/HU)/PCI-1800(H/HU) (High-Gain)**							
Gain	0.5	1	5	10	50	100	500	1000
Bipolar (V)	± 10	± 5	± 1	± 0.5	± 0.1	± 0.05	± 0.01	± 0.005
Unipolar (V)	-	0~10	-	0 ~ 1	-	0 ~ 0.1	-	0 ~ 0.01
Sampling Rate Max.	44 kS./s				10 kS./s		1 kS./s	

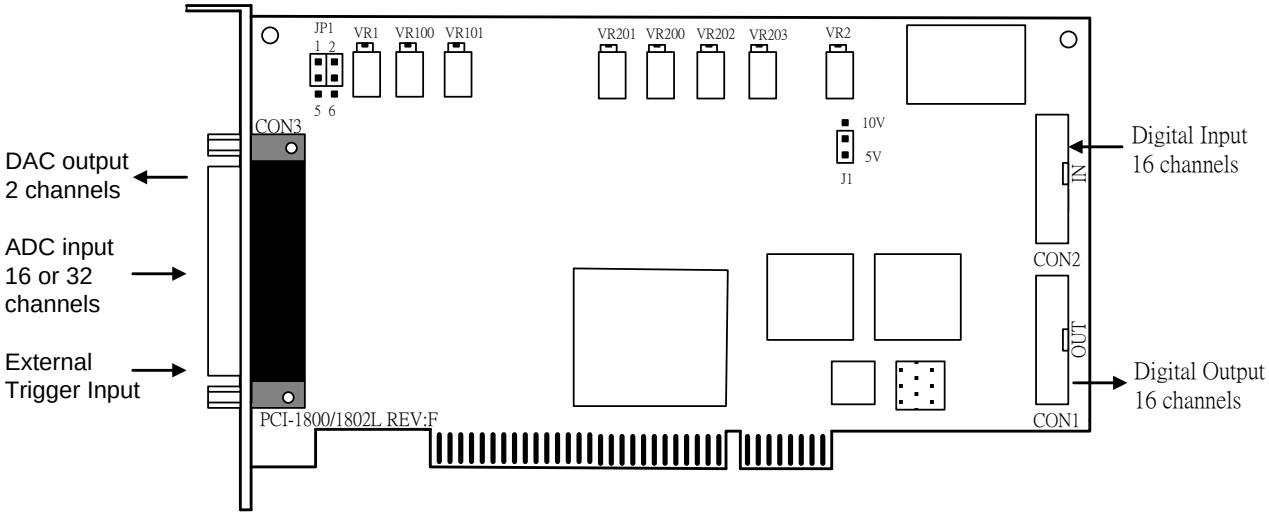
2. Hardware Configuration

2.1 Board Layout

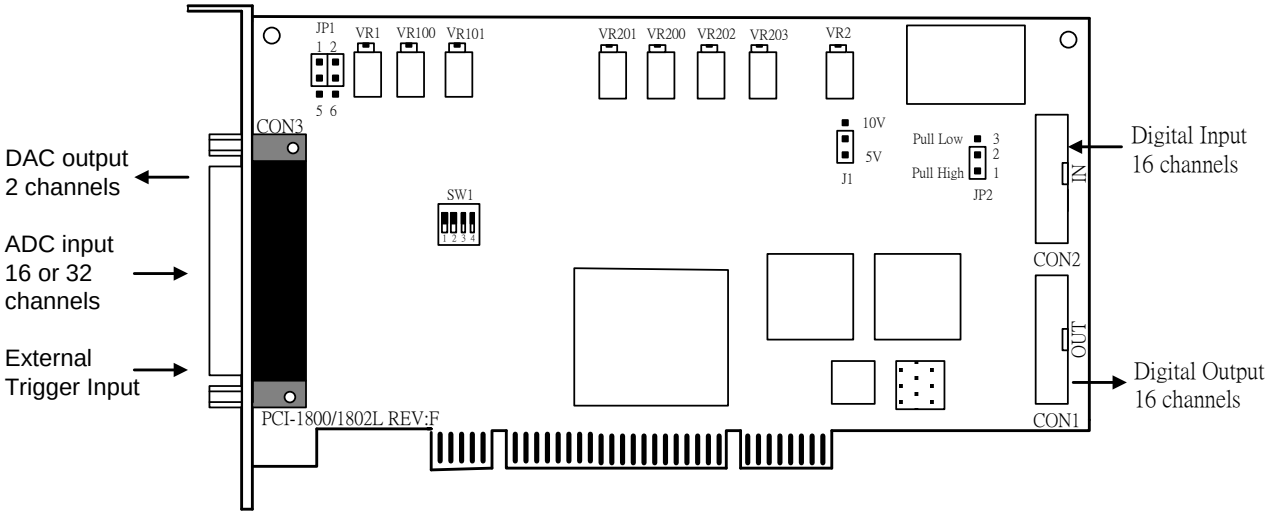
- PCI-180x(H/L)/NDA Board Layout.



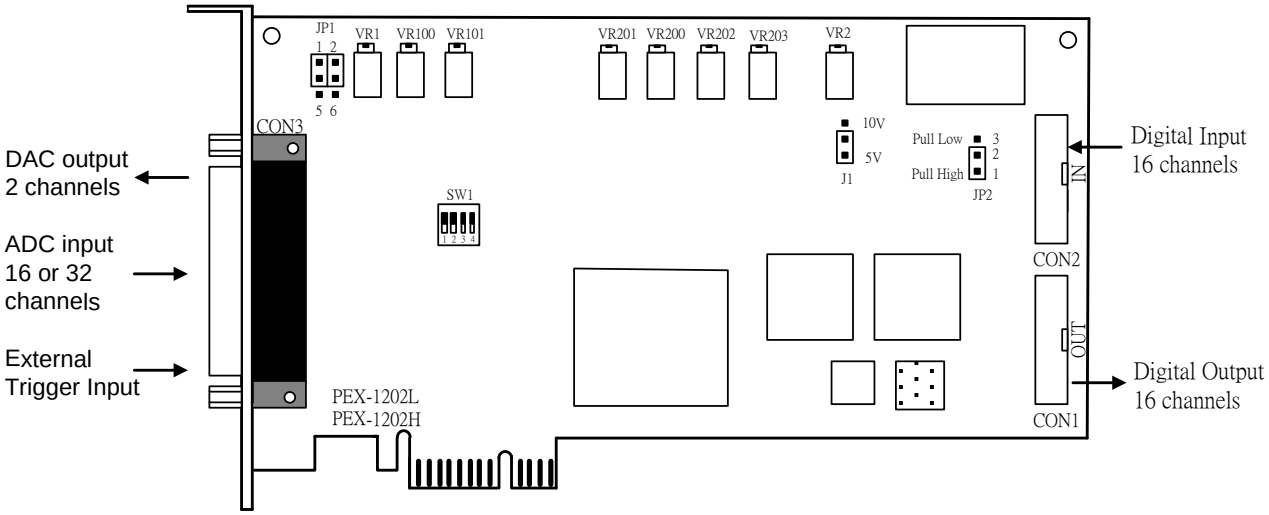
- PCI-1202/1800/1802(H/L) Board Layout.



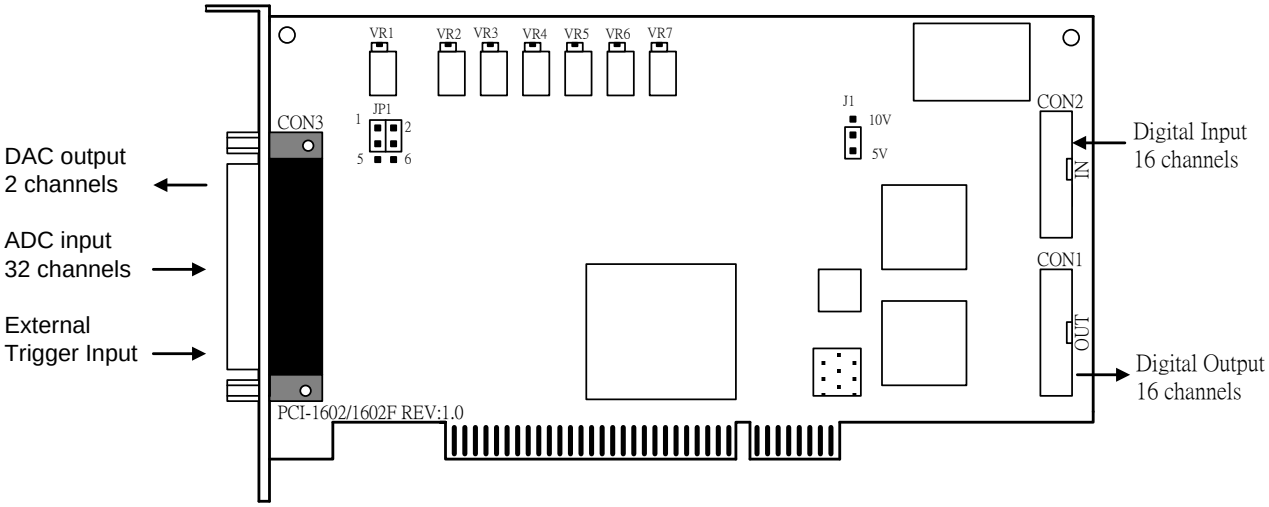
PCI-1202/1800/1802(HU/LU) Board Layout.



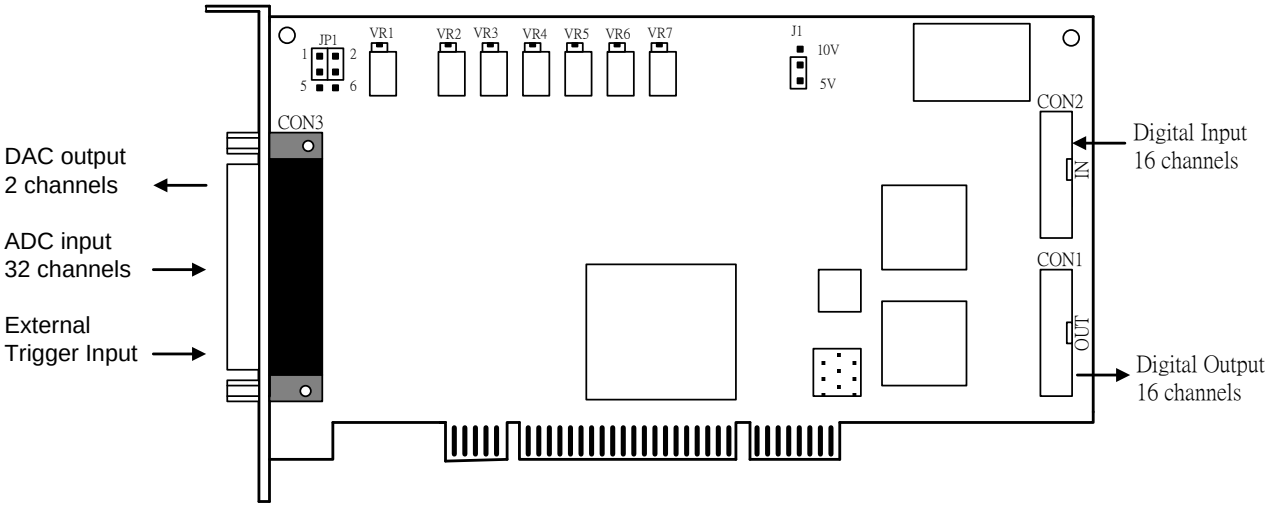
PEX-1202(H/L) Board Layout.



PCI-1602(/F) Board Layout.



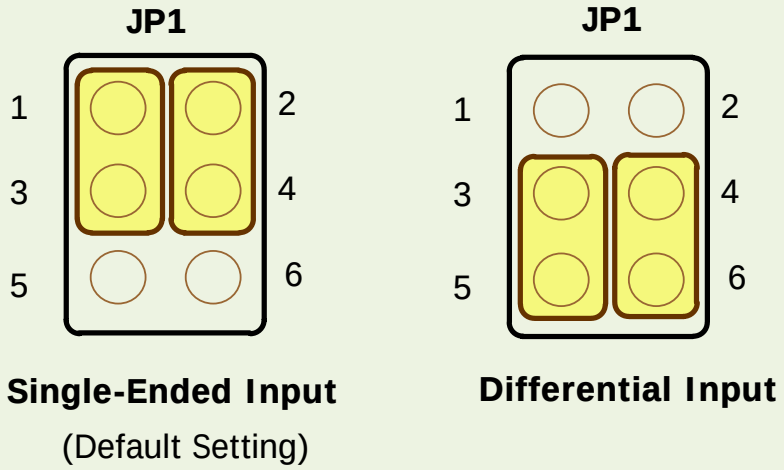
PCI-1602(U/FU) Board Layout.



2.2 Jumper Setting

2.2.1 JP1: A/D Input Type Selection

This jumper is used to select the analog input type. For single-ended inputs, connect pin1, 3 and pin2, 4. For differential inputs, pin3, 5 and pin4, 6 should be connected. The configuration is illustrated in the figure below.



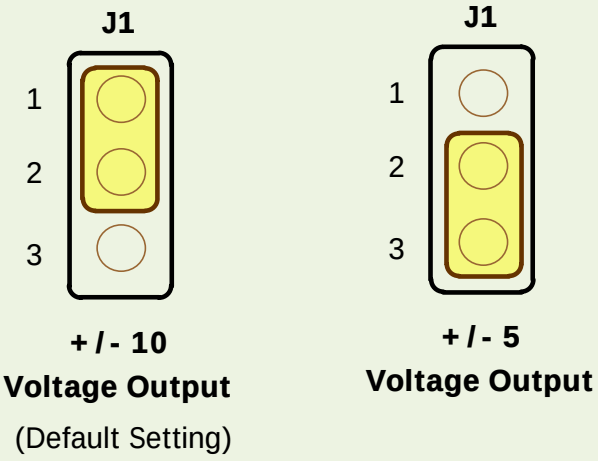
Single-Ended Input
(Default Setting)

Differential Input

For detailed information about the single-ended and differential input wiring, please refer to *Sec. 2.4 Analog input signal connection.*

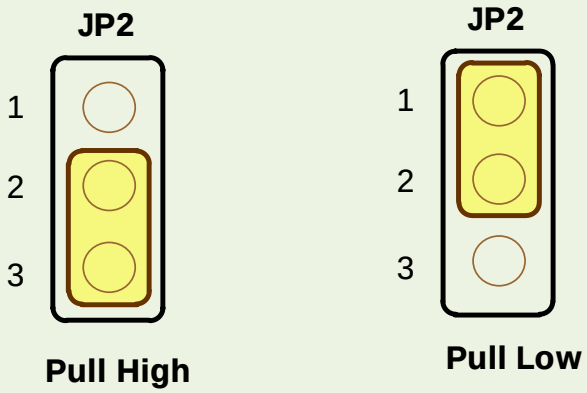
2.2.2 J1: D/A Reference Voltage Selection

J1 is used to select the internal D/A output reference voltage. To select the ±10 V voltage output, the pin 1&2 should be connected. To select the ±5 V voltage output, the pin 2&3 should be connected. The configuration is illustrated in the figure below.



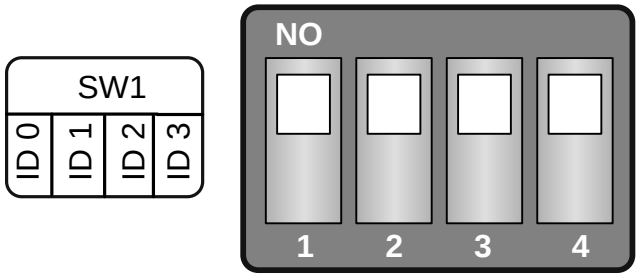
2.2.3 D/I Port Setting (Pull-High/Low)

This DI ports can be pull-high or pull-low that is selected by JP2. The pull-high/low jumpers of the card allow user to predefine the DI status instead of floating when the DI channels are unconnected or broken. The configuration is illustrated in the figure below. ***Note!! This function only supports PEX-1202(L/H), PCI-1602(U/FU) and PCI-1202/18001802 (LU/HU)***.



2.3 Card ID Switch

The PEX-1202(L/H), PCI-1202/1800/1802(LU/HU) and PCI-1602(U/FU) has a Card ID switch (SW1) with which users can recognize the board by the ID via software when using two or more PEX-1202 and PCI-1202/1602/1800/1802 series cards in one computer. The default Card ID is 0x0. For detail SW1 Card ID settings, please refer to Table 2.1.



(Default Settings)

Table 2.1 (*) Default Settings; OFF → 1; ON → 0

Card ID (Hex)	1 ID0	2 ID1	3 ID2	4 ID3
(*) 0x0	ON	ON	ON	ON
0x1	**OFF**	ON	ON	ON
0x2	ON	**OFF**	ON	ON
0x3	**OFF**	**OFF**	ON	ON
0x4	ON	ON	**OFF**	ON
0x5	**OFF**	ON	**OFF**	ON
0x6	ON	**OFF**	**OFF**	ON
0x7	**OFF**	**OFF**	**OFF**	ON
0x8	ON	ON	ON	**OFF**
0x9	**OFF**	ON	ON	**OFF**
0xA	ON	**OFF**	ON	**OFF**
0xB	**OFF**	**OFF**	ON	**OFF**
0xC	ON	ON	**OFF**	**OFF**
0xD	**OFF**	ON	**OFF**	**OFF**
0xE	ON	**OFF**	**OFF**	**OFF**
0xF	**OFF**	**OFF**	**OFF**	**OFF**

2.4 Analog Input Signal Connection

The PEX-1202 and PCI-1202/1602/1800/1802 series can measure single-ended or differential-type analog input signal. Some analog signals can be measured in both modes. However, some analog signals only can be measured in one or the other. The user must decide which mode is suitable for measurement.

In general, there are 4 different analog signal connection methods (shown from **Figure 2.4-1** to **Figure 2.4-5**). The connection in **Figure 2.4-1** is suitable for grounded analog input signals. The **Figure 2.4-3** connection is used to measure more channels than in **Figure 2.4-1**, but it is only suitable for analog signals that large than 1 V. The connection in **Figure 2.4-4** is suitable for thermocouple and the **Figure 2.4-5** connection is suitable for floating analog input signals.

Note: In Figure 2.4-4 the maximum common mode voltage between the analog input source and the AGND is 70 V_{p-p}, so the user must take care that the input signal is under this specification first. If the common mode voltage is over 70 V_{p-p}, the input multiplexer will be permanently damaged!

The simple way to select your input signal connection configuration is listed below.

1. **Grounded source input signal → see Figure 2.4-1**
2. **Thermocouple input signal → see Figure 2.4-4**
3. **Floating source input signal → see Figure 2.4-5**
4. **If Vin > 1 V, the gain<=10 and more channels are needed → see Figure 2.4-3**
5. **Current source input signal → see Figure 2.4-6**

If you are unsure of the characteristics of your input signal, follow these test step:

1. **Step1 : Try and record the measurement result of the Figure 2.4-1**
2. **Step2 : Try and record the measurement result of the Figure 2.4-5**
3. **Step3 : Try and record the measurement result of the Figure 2.4-3**
4. **Compare the three results and select the best one**

Figure2.4-1: Differential input with grounded source (Right way)

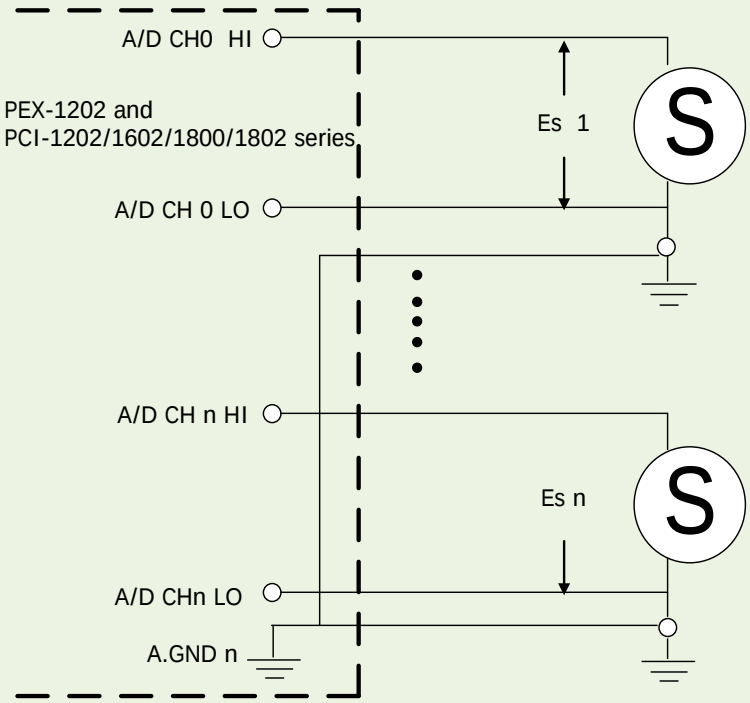
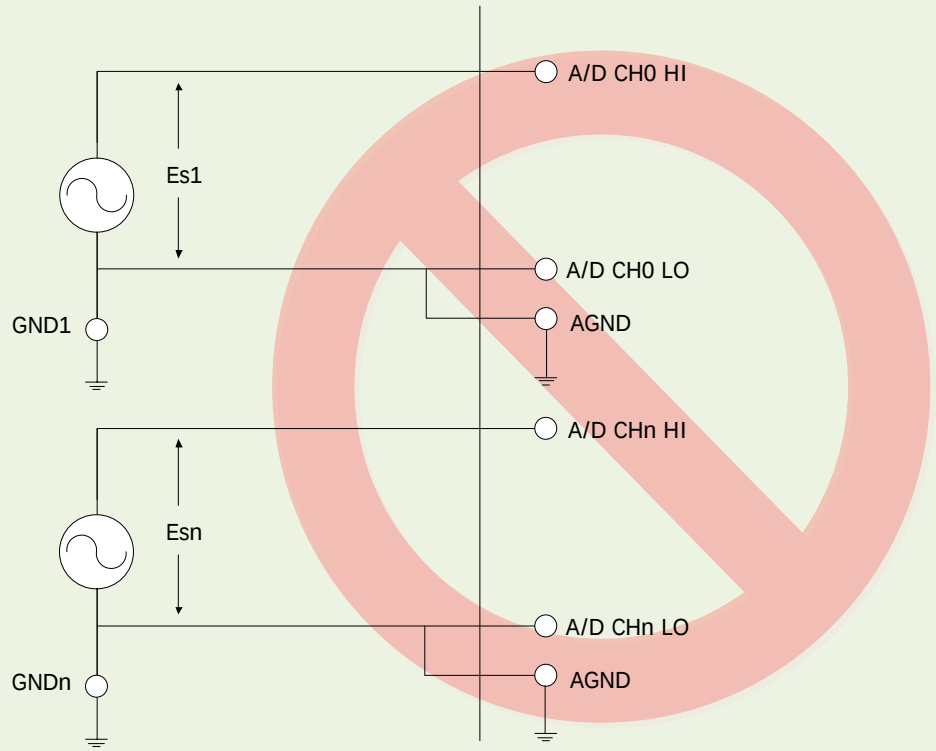
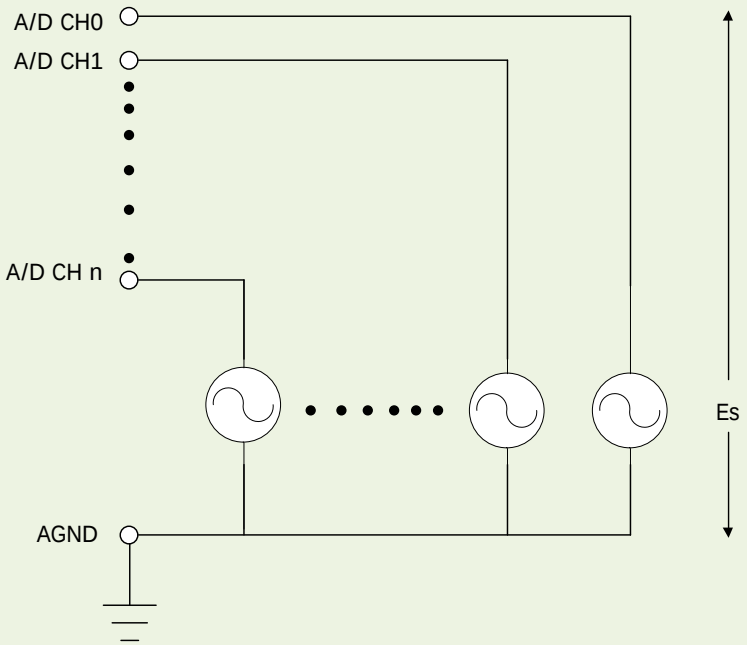


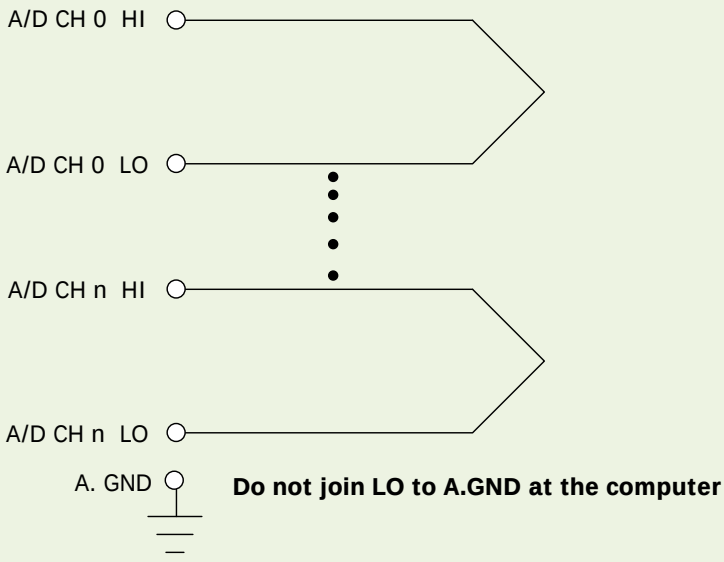
Figure2.4-2: Grounded loop input (Wrong way)



■ **Figure 2.4-3:** Single-ended input with floating signal source



■ **Figure 2.4-4:** Differential input with floating thermocouple signal



Note: If the input signal is not thermocouple, the user should use an oscilloscope to measure common mode voltage of Vin before connecting to PEX-1202 and PCI-1202/1602/1800/1802 series card. Don't use a voltage meter or multi-meter.

CAUTION: In **Figure 2.4-4**, the maximum common mode voltage between the analog input source and the AGND is 70 Vp-p. Make sure that the input signal is under specification first! If the common mode voltage is over 70 Vp-p, the input multiplexer will be permanently damaged.

Figure 2.4-5: Differential input with floating signal source

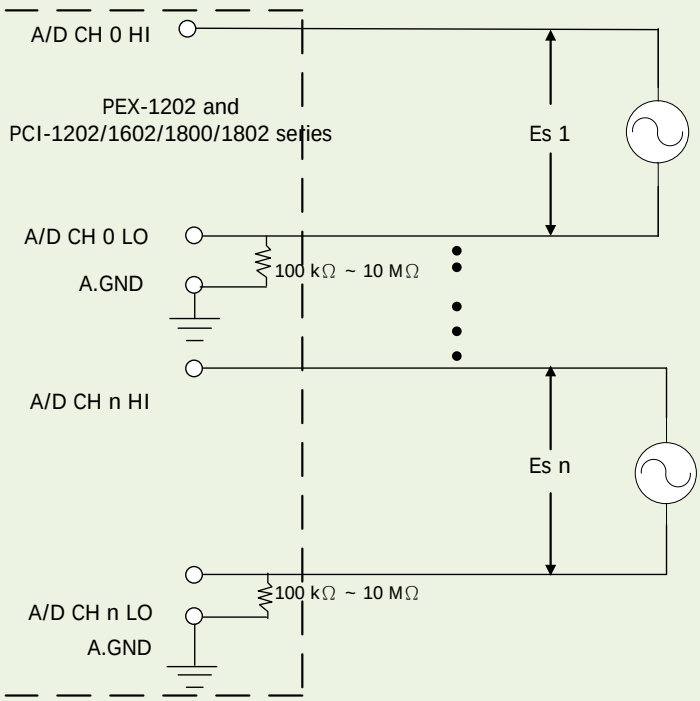
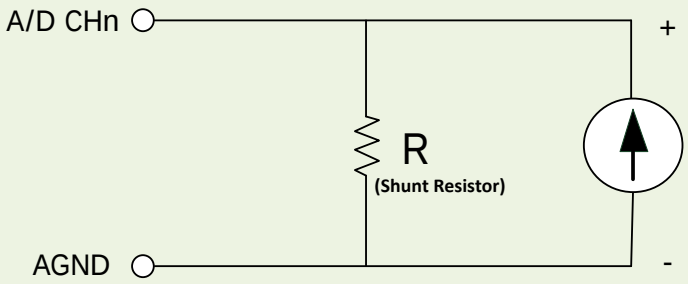


Figure 2.4-6: Connecting to a 4 ~ 20 mA Source



Example: A 20 mA source current through a 125 Ω resistor (e.g. 125 Ω, 0.1% DIP Resistors) between + and – terminals and the board will read a 2.5 V_{DC} voltage. You can use the **I = V/R** (Ohm's law) to calculate what value the source current should have.

Current (I) = Voltage (V) / Resistance (R)

= 2.5 V / 125 Ω

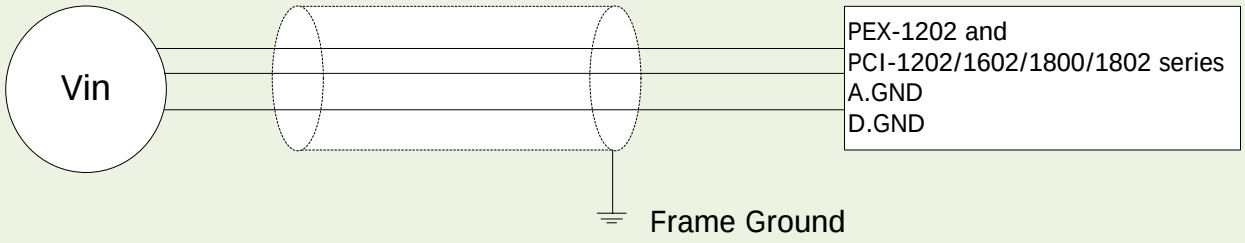
= 0.02 A

= 20 m

Signal Shielding

Signal shielding connections in **Figure 2.4-1** to **Figure 2.4-6** are all the same

- Use a single-point connection to **frame ground (not A.GND or D.GND)**



2.5 Pin Assignments

- CON1 and CON2: **Digital output and digital input** connector for PEX-1202 and PCI-1202/1602/1800/1802 series card.

Pin Assign-ment	Terminal No.		Pin Assign-ment
DO 0	01	02	DO 1
DO 2	03	04	DO 3
DO 4	05	06	DO 5
DO 6	07	08	DO 7
DO 8	09	10	DO 9
DO 10	10	12	DO 11
DO 12	12	14	DO 13
DO 14	14	16	DO 15
GND	16	18	GND
+5V	18	20	+12V

CON1

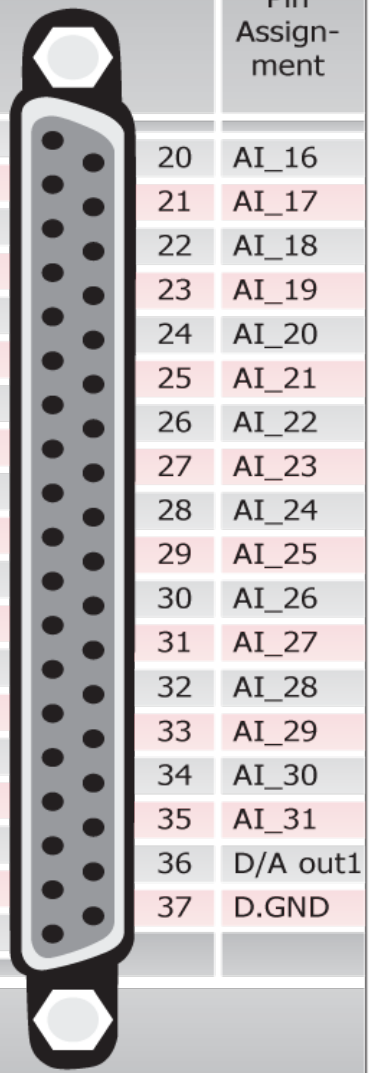
Pin Assign-ment	Terminal No.		Pin Assign-ment
DI 0	01	02	DI 1
DI 2	03	04	DI 3
DI 4	05	06	DI 5
DI 6	07	08	DI 7
DI 8	09	10	DI 9
DI 10	11	12	DI 11
DI 12	13	14	DI 13
DI 14	15	16	DI 15
GND	17	18	GND
+5V	19	20	+12V

CON2

- CON3: **Single-ended input** pin assignment as follows:

PEX-1202(L/H)
PCI-1202/1802(L/H/LU/HU)
PCI-1602(/F/U/FU)

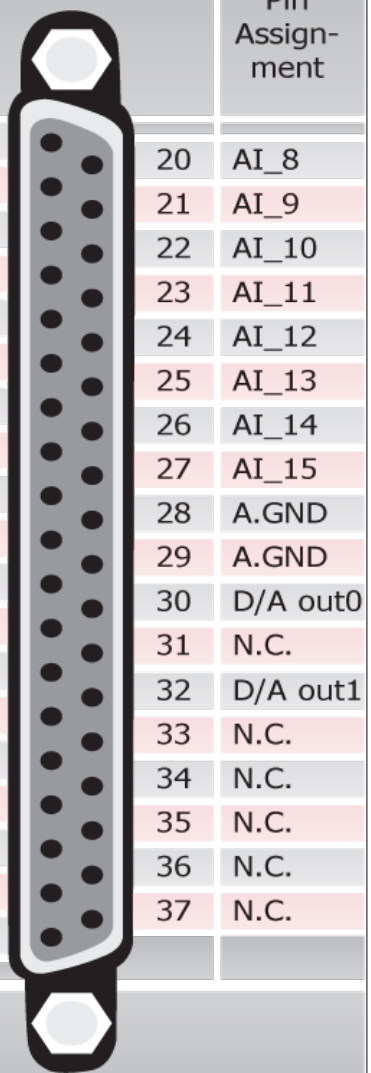
Pin Assign-ment	Terminal No.		Terminal No.	Pin Assign-ment
AI_0	01		20	AI_16
AI_1	02		21	AI_17
AI_2	03		22	AI_18
AI_3	04		23	AI_19
AI_4	05		24	AI_20
AI_5	06		25	AI_21
AI_6	07		26	AI_22
AI_7	08		27	AI_23
AI_8	09		28	AI_24
AI_9	10		29	AI_25
AI_10	11		30	AI_26
AI_11	12		31	AI_27
AI_12	13		32	AI_28
AI_13	14		33	AI_29
AI_14	15		34	AI_30
AI_15	16		35	AI_31
A.GND	17		36	D/A out1
D/A out0	18		37	D.GND
Ext_Trg	19			



CON3

PCI-1800(L/H/LU/HU)

Pin Assign-ment	Terminal No.		Terminal No.	Pin Assign-ment
AI_0	01		20	AI_8
AI_1	02		21	AI_9
AI_2	03		22	AI_10
AI_3	04		23	AI_11
AI_4	05		24	AI_12
AI_5	06		25	AI_13
AI_6	07		26	AI_14
AI_7	08		27	AI_15
A.GND	09		28	A.GND
A.GND	10		29	A.GND
N.C.	11		30	D/A out0
N.C.	12		31	N.C.
+12V out	13		32	D/A out1
A.GND	14		33	N.C.
D.GND	15		34	N.C.
N.C.	16		35	N.C.
Ext_Trig	17		36	N.C.
N.C.	18		37	N.C.
+5V out	19			

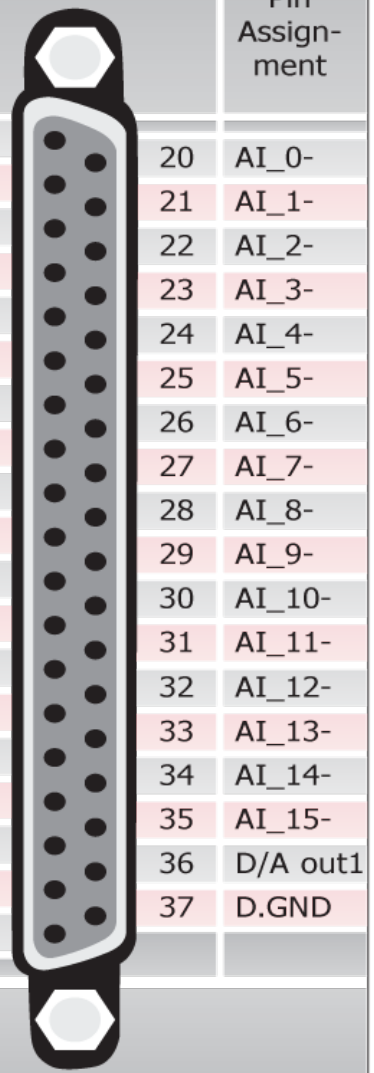


CON3

- CON3: **Differential input** pin assignment as follows:

PEX-1202(L/H)
PCI-1202/1802(L/H/LU/HU)
PCI-1602(/F/U/FU)

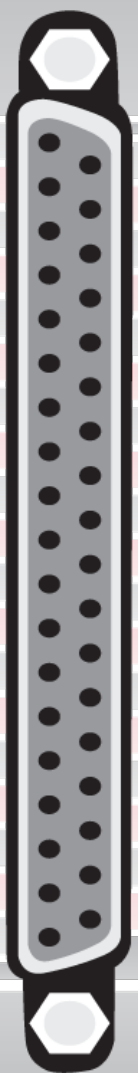
Pin Assign-ment	Terminal No.		Pin Assign-ment
AI_0+	01	20	AI_0-
AI_1+	02	21	AI_1-
AI_2+	03	22	AI_2-
AI_3+	04	23	AI_3-
AI_4+	05	24	AI_4-
AI_5+	06	25	AI_5-
AI_6+	07	26	AI_6-
AI_7+	08	27	AI_7-
AI_8+	09	28	AI_8-
AI_9+	10	29	AI_9-
AI_10+	11	30	AI_10-
AI_11+	12	31	AI_11-
AI_12+	13	32	AI_12-
AI_13+	14	33	AI_13-
AI_14+	15	34	AI_14-
AI_15+	16	35	AI_15-
A.GND	17	36	D/A out1
D/A out0	18	37	D.GND
Ext_Trg	19		



CON3

PCI-1800(L/H/LU/HU)

Pin Assign-ment	Terminal No.		Pin Assign-ment
AI_0+	01	20	AI_0-
AI_1+	02	21	AI_1-
AI_2+	03	22	AI_2-
AI_3+	04	23	AI_3-
AI_4+	05	24	AI_4-
AI_5+	06	25	AI_5-
AI_6+	07	26	AI_6-
AI_7+	08	27	AI_7-
A.GND	09	28	A.GND
A.GND	10	29	A.GND
N.C.	11	30	D/A out0
N.C.	12	31	N.C.
+12V out	13	32	D/A out1
A.GND	14	33	N.C.
D.GND	15	34	N.C.
N.C.	16	35	N.C.
Ext_Trig	17	36	N.C.
N.C.	18	37	N.C.
+5V out	19		



CON3

3. Hardware Installation

Note!!

It's recommended to install driver first, since some operating system (such as Windows 2000) may ask you to restart the computer again after driver installation. This reduces the times to restart the computer.

To install your PEX-1202 and PCI-1202/1602/1800/1802 series card, complete the following steps:

Step 1: Installing DAQ card driver on your computer first.



For detailed information about the driver installation, please refer to Chapter 4 Software Installation.

Step 2: Configuring Card ID by the SW1 DIP-Switch.



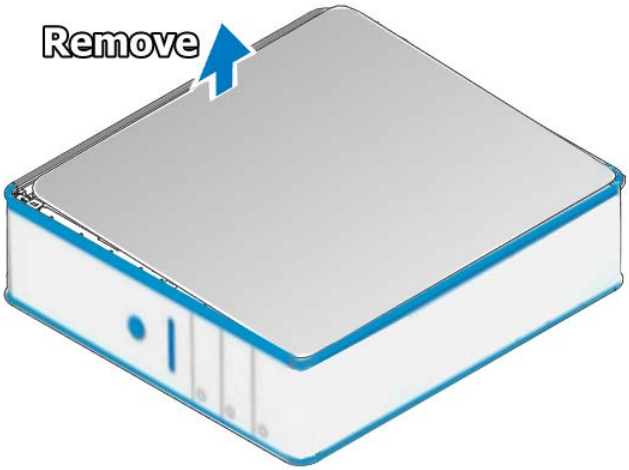
For detailed information about the card ID (SW1), please refer to Sec. 2.3 Car ID Switch.

Note!! The card ID function only supports PEX-1202(L/H), PCI-1602(U/FU) and PCI-1202/1800/1802 (LU/HU).

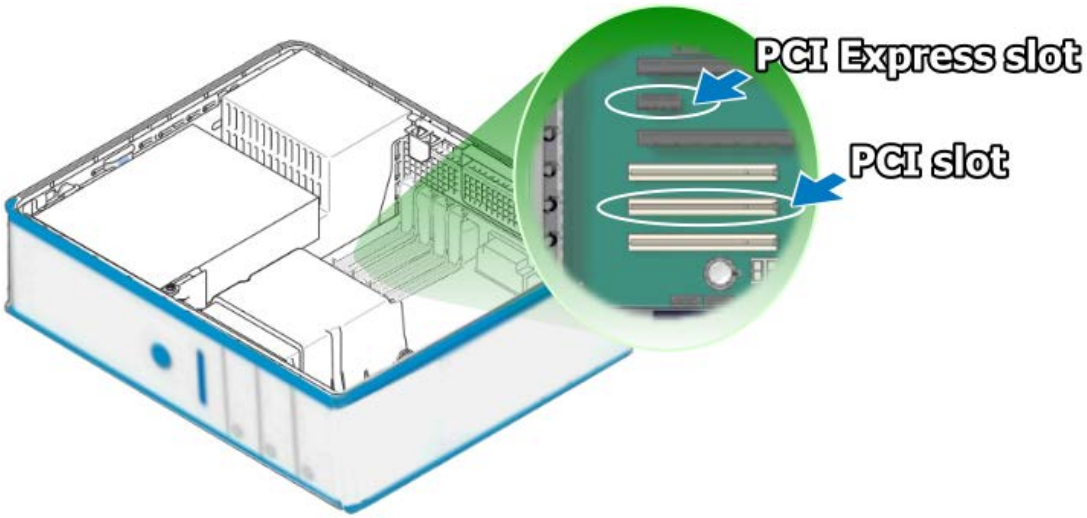


Step 3: Shut down and power off your computer.

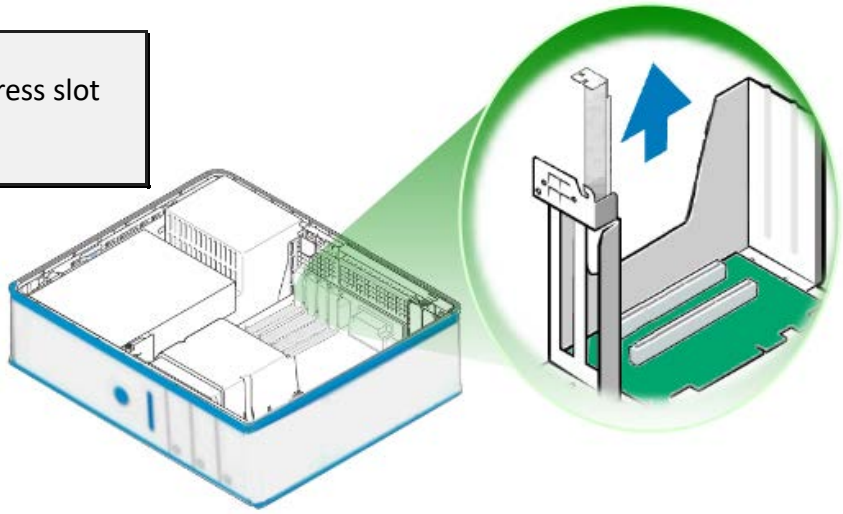
Step 4: Remove all covers from the computer.

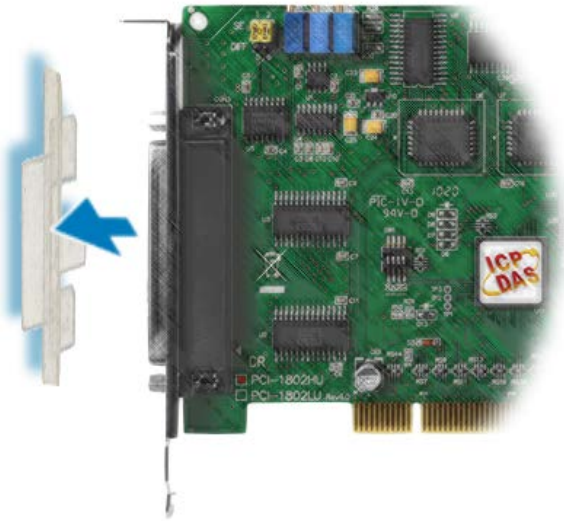


Step 5: Select an empty PCI/PCI Express slot.



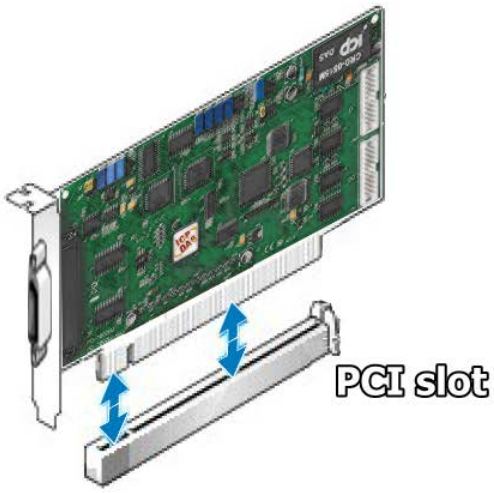
Step 6: Remove the PCI/PCI Express slot cover form the PC.

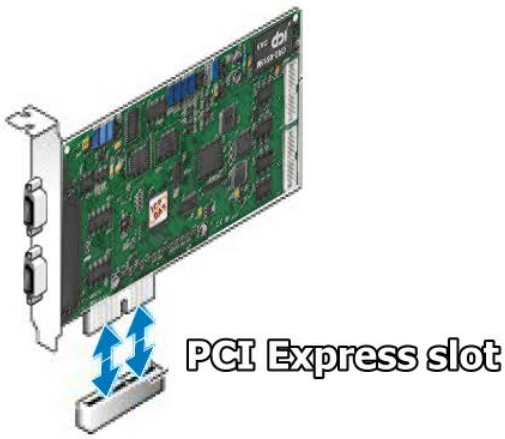


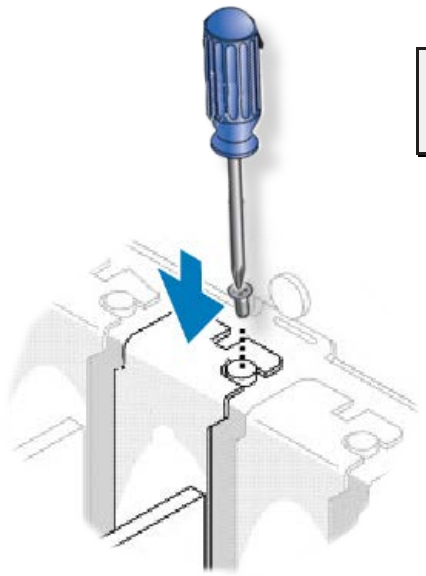


Step 7: Remove the connector cover form the DAQ card.

Step 8: Carefully insert your DAQ card into the PCI/PCI Express slot.

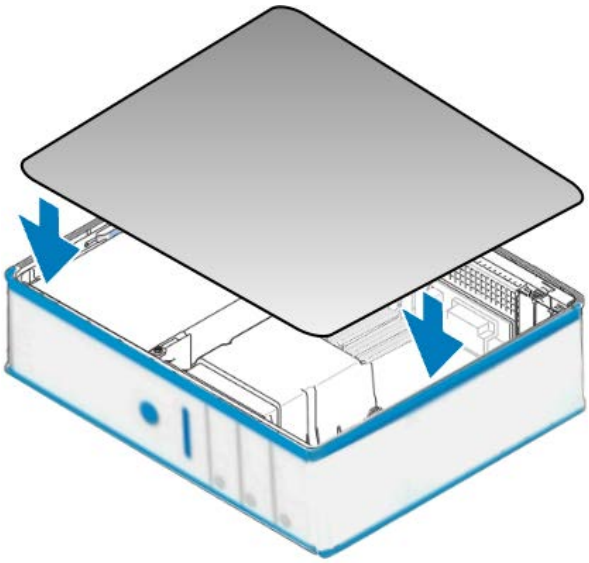






Step 9: Tighten the captive Phillips screw.

Confirm the PEX-1202 and PCI-1202/1602/1800/1802 series card is mounted on the motherboard.



Step 10: Replace the computer cover.

Step 11: Power on the computer.



Follow the prompt message to finish the Plug&Play steps, please refer to Chapter 4 Software Installation.

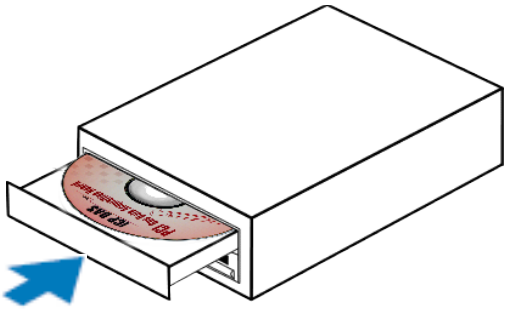
4. Software Installation

The PEX-1202 and PCI-1202/1602/1800/1802 series card can be used in DOS, Linux and Windows 98/NT/2K and 32-bit/64-bit Windows XP/2003/Vista/7/8. This chapter shows you the detail steps to install these drivers. The recommended installation procedure for **Windows** is given in Sections 4.1 to 4.3.

4.1 Driver Installing Procedure

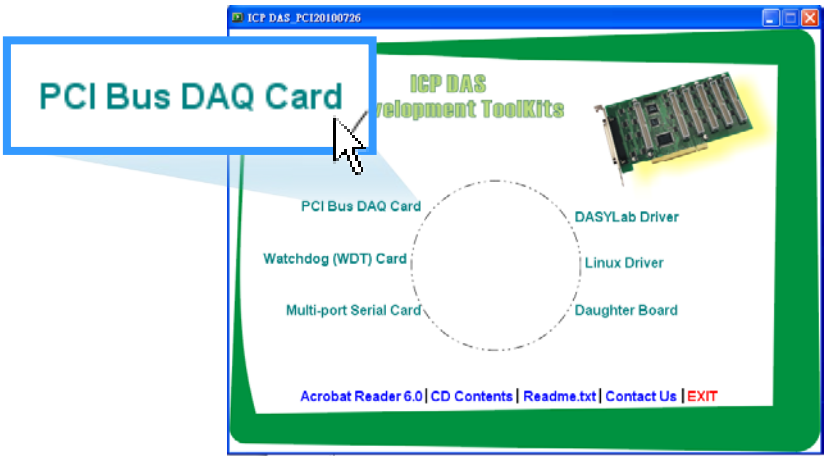
Follow these steps:

Step 1: Run the companion CD.



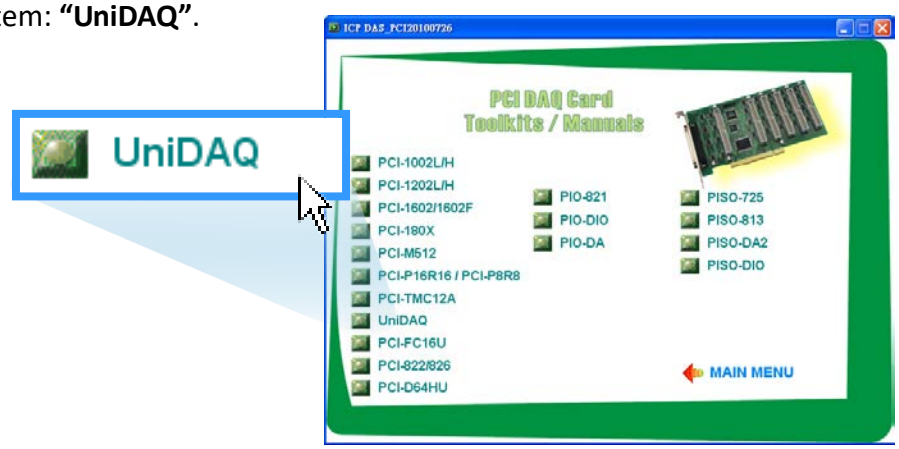
Insert the companion CD into the CD-ROM driver and wait a few seconds until the installation program starts automatically. If it does not start automatically for some reason, then please double-click the file **\NAPDOS\AUTO32.EXE** on the CD.

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

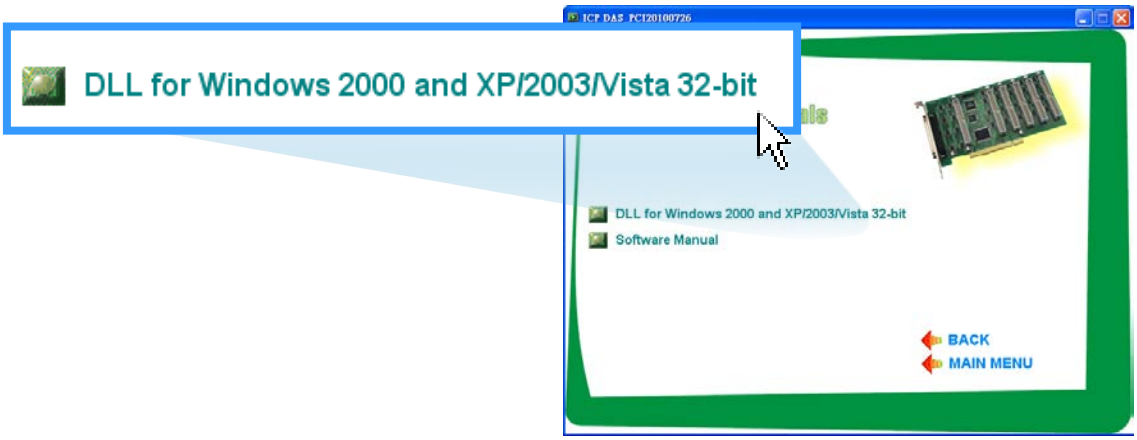


Step 3: Please install the appropriate driver for your OS.

1. Click the item: **“UniDAQ”**.



2. Click the item: **“DLL for Windows 2000 and XP/2003/Vista 32-bit”**.



3. Double-Click **“UniDAQ_Win_Setup_x.x.x.x_xxxx.exe”** file in the **“Driver”** folder.

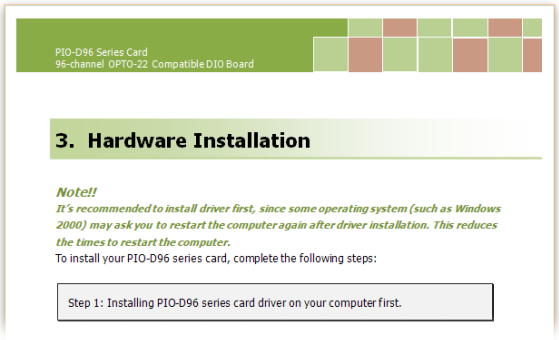


4. Click the **"Next>"** button to start the installation.
5. Check your DAQ Card is or not on supported list, Click the **"Next>"** button.
6. Select the installed folder, the default path is C:\ICPDAS\UniDAQ, confirm and click the **"Next>"** button.
7. Check your DAQ card on list, then click the **"Next>"** button.
8. Click the **"Next>"** button on the **Select Additional Tasks** window.
9. The demo program can be obtained from the following link and then click the **"Next>"** button.
10. Select **"No, I will restart my computer later"** and then click the **"Finish"** button.

For detailed information about the UniDAQ driver installation, please refer to UniDAQ DLL Software Manual. The user manual is contained in: CD:\NAPDOS\PCI\UniDAQ\Manual\
http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/unidaq/manual/

4.2 PnP Driver Installation

Step 1: Turn off the computer and install the DAQ card into the computer.



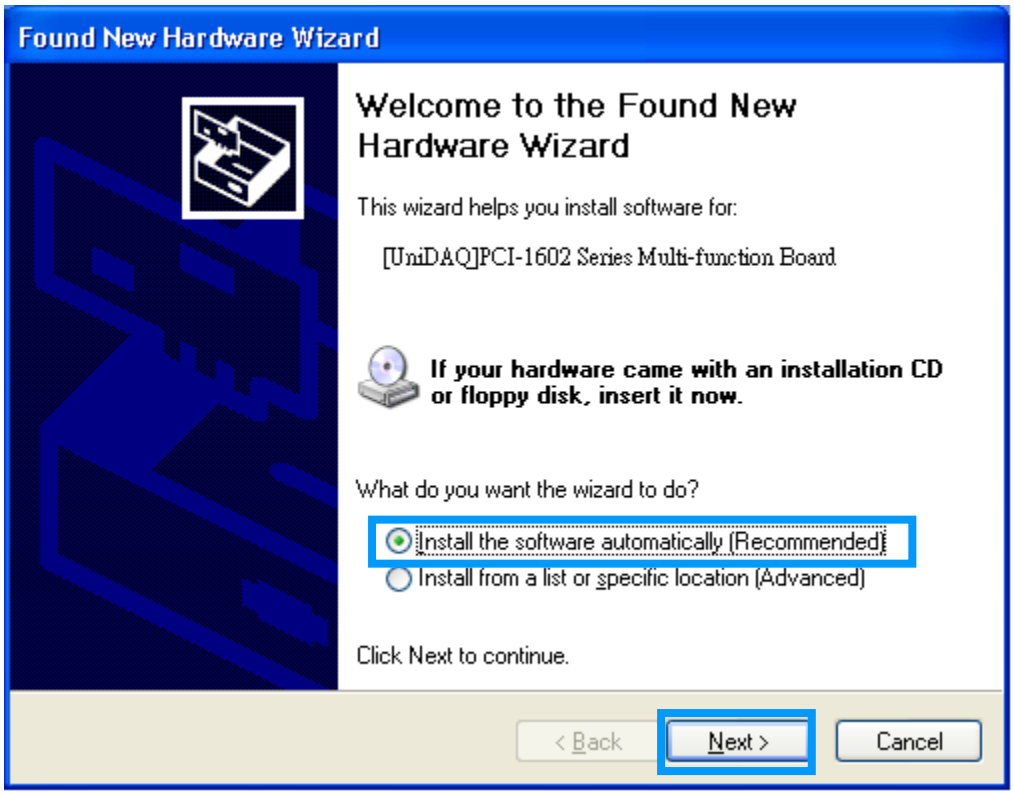
For detailed information about the hardware installation of PEX-1202 and PCI-1202/1602/1800/1802 series card, please refer to Chapter 3 Hardware Installation.

Step 2: Power on the computer and system should find the new card and then continue to finish the Plug&Play steps.

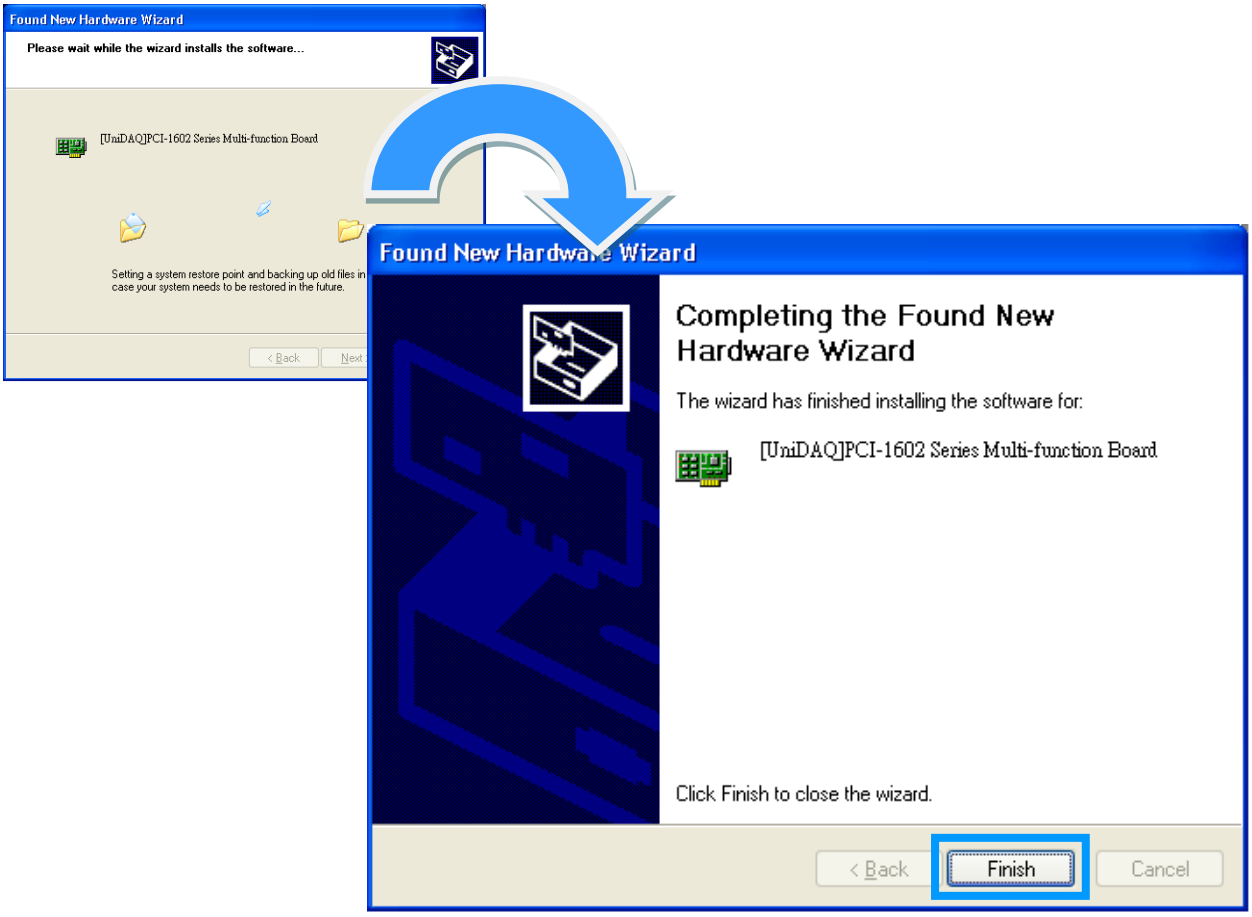
Note: Some Windows OS will load the driver automatically to complete the installation at boot.



Step 3: Select **"Install the software automatically [Recommended]"** and click the **"Next>"** button.



Step 4: Click the **"Finish"** button.



Step 5: Windows pops up **"Found New Hardware"** dialog box again.



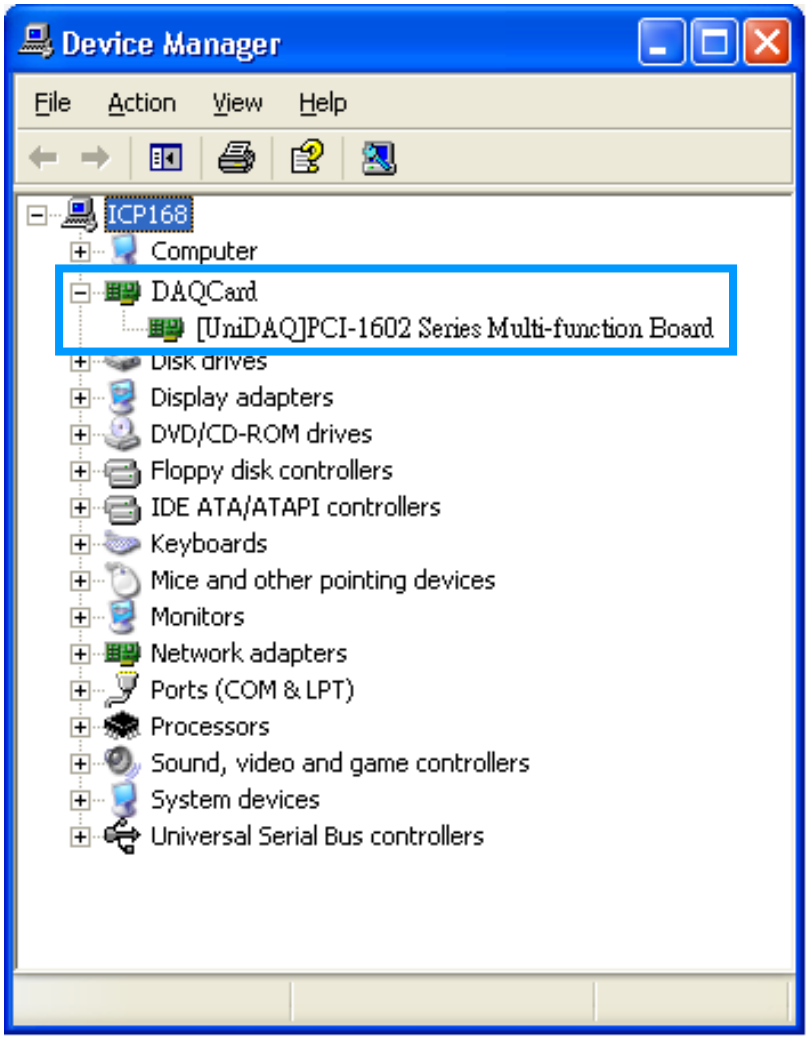
4.3 Confirm the Successful Installation

Make sure the PEX-1202 and PCI-1202/1602/1800/1802 series card installed is correct on the computer as follows:

Step 1: Select the **"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 PEX-1202 and PCI-1202/1602/1800/1802 series card which listed correctly or not, as illustrated below.



5. Testing PCI-1202/1602/180x Card

This chapter can give you the detail steps about self-test. In this way, user can confirm that PEX-1202 and PCI-1202/1602/1800/1802 series card well or not. Before the self-test, you must complete the hardware and driver installation. For detailed information about the hardware and driver installation, please refer to Chapter 3 Hardware Installation and Chapter 4 Software Installation.

5.1 Self-Test Wiring

5.1.1 DIO Test Wiring

1. Prepare for device:

   ☑ One CA-2002 (optional) cable.

2. Use the CA-2002 to connect the CON1 with CON2 on board.

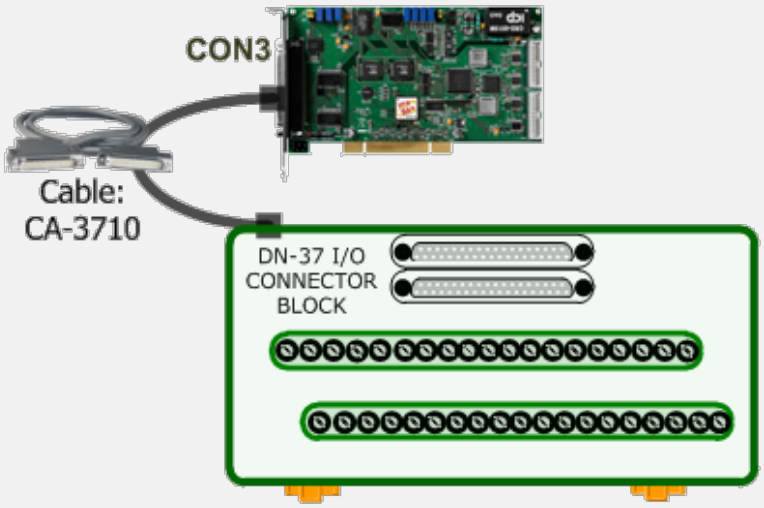


5.1.2 Analog Input Test Wiring

1. Prepare for device:

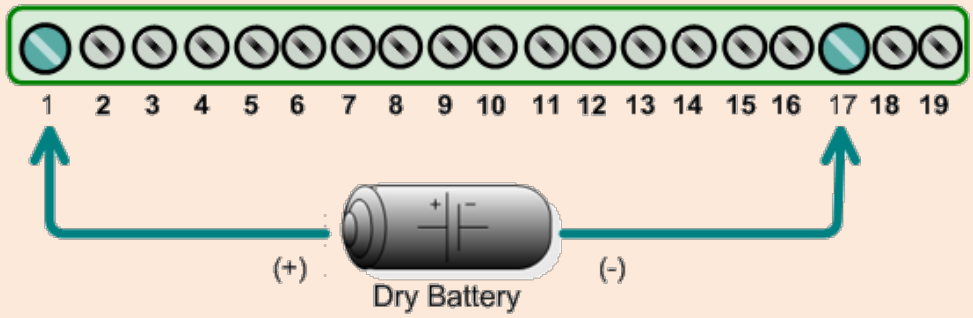
   ☑ One DN-37 (optional) wiring terminal board.
   ☑ One CA-3710 (optional) cable.
   ☑ Provide a stable signal source. (for example, dry battery)

2. Connect a DN-37 to the CON3.



3. Wire the signal source to A/D channel_0, and then keep set the **JP1** jumper to **Single-Ended** (refer to Sec. 2.2.1), and wire the signals as follows:

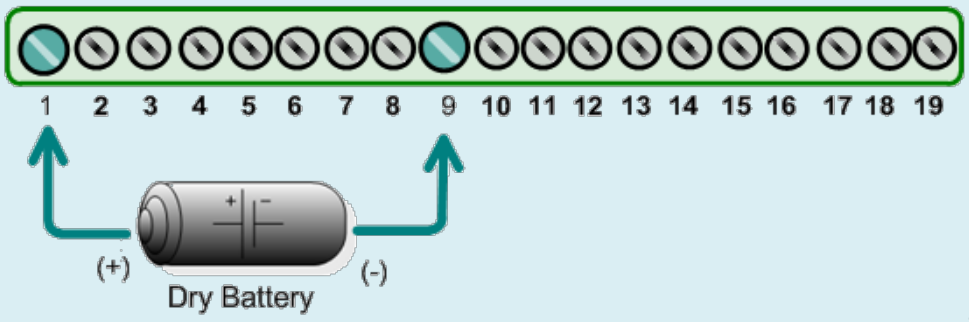
- **The PEX-1202 and PCI-1202/1602/1802 Series :**
  Connect the **AI_01 (Pin01)** to **signal positive(+)**
  Connect the **A.GND(Pin17)** to **signal negative(-)**



- **The PCI-1800 Series :**

  Connect the **AI_01 (Pin01)** to **signal positive(+)**

  Connect the **A.GND(Pin09)** to **signal negative(-)**



5.1.3 Analog Output Test Wiring

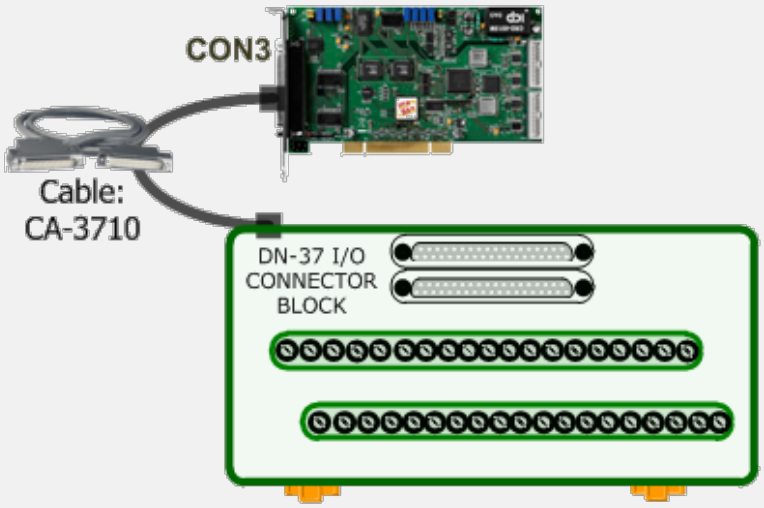
1. Prepare for device:

   ☑ One DN-37 (optional) wiring terminal board.

   ☑ One CA-3710 (optional) cable.

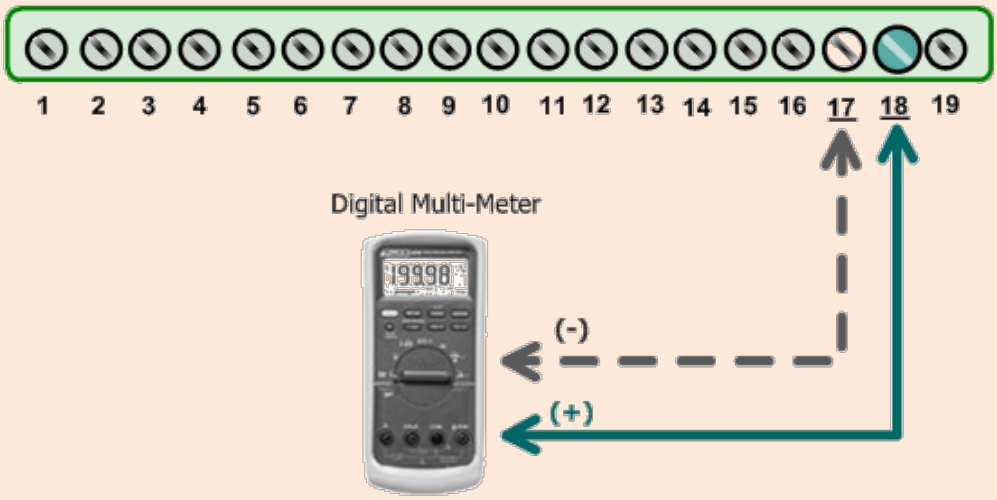
   ☑ Digital Multi-Meter.

2. Connect a DN-37 to the CON3.

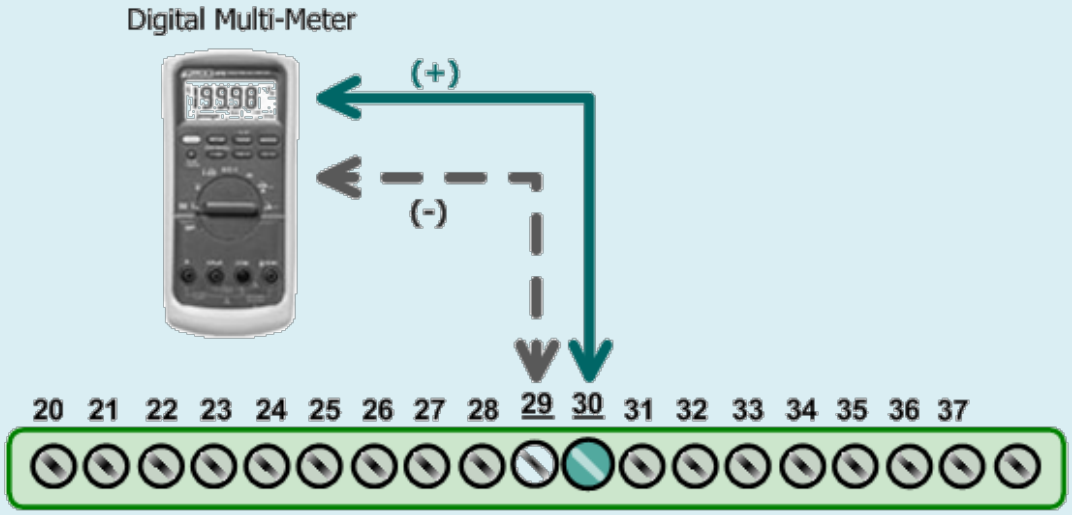


3. Wire the digital multi-meter to D/A channel_0, and then keep set the **J1** jumper to **±10 V voltage** (refer to Sec. 2.2.2), and wire the signals as follows:

- **The PEX-1202 and PCI-1202/1602/1802 Series :**
  Connect the **positive probe (+) of Multi-meter** to D/A_out0 (Pin 18).
  Connect the **negative probe (-) of Multi-meter** to A.GND (Pin 17).



- **The PCI-1800 Series :**
  Connect the **positive probe (+) of Multi-meter** to D/A_out0 (Pin 30).
  Connect the **negative probe (-) of Multi-meter** to A.GND (Pin 29).

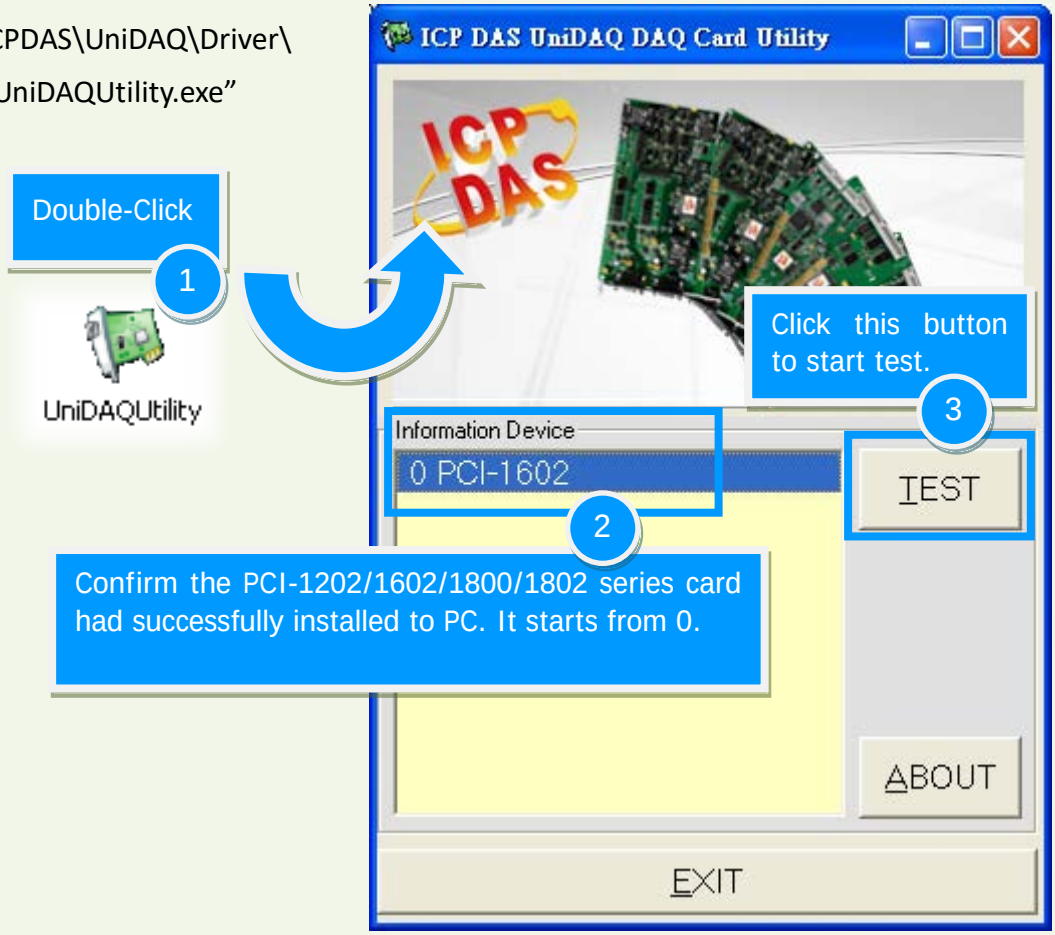


5.2 Execute the Test Program

1. Execute the UniDAQ Utility Program. The UniDAQ Utility.exe will be placed in the default path after completing installation.

Default Path: C:\ICPDAS\UniDAQ\Driver\

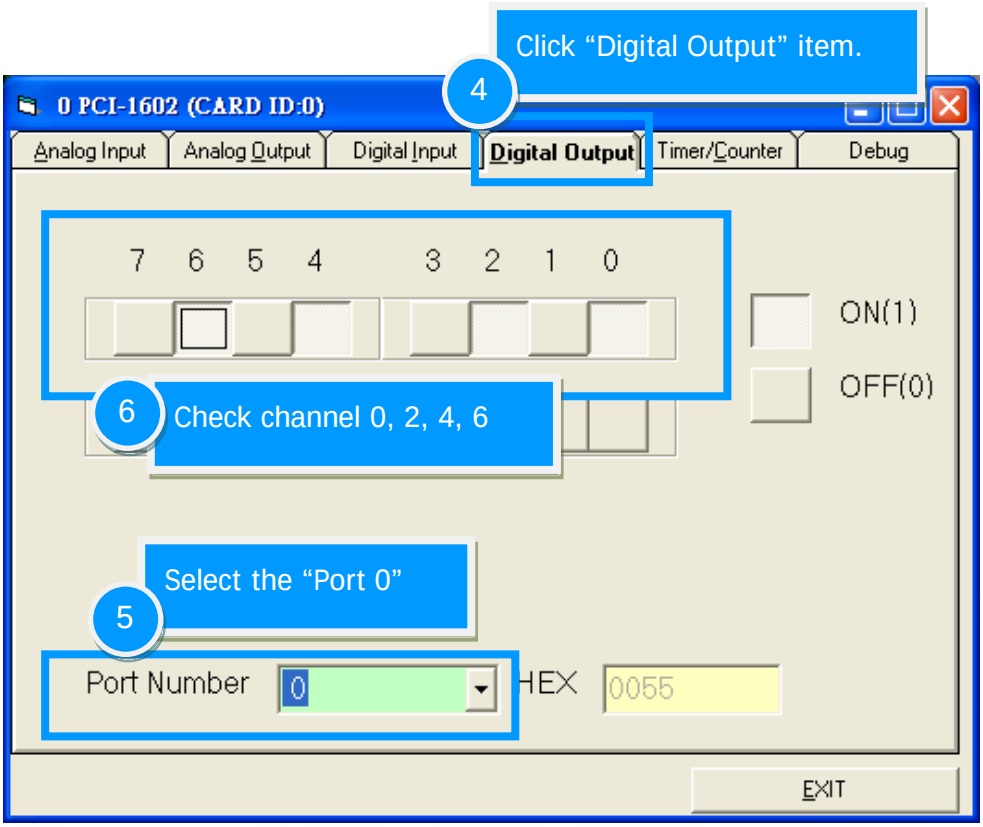
Double click the "UniDAQUtility.exe"

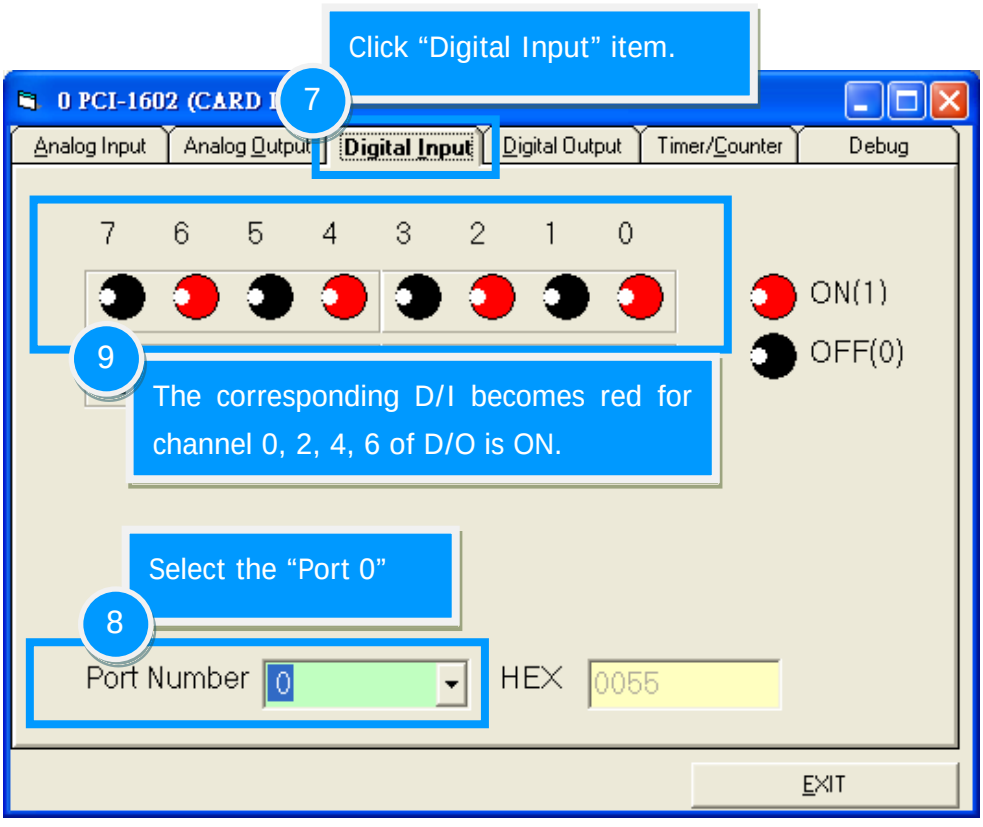


Note!!

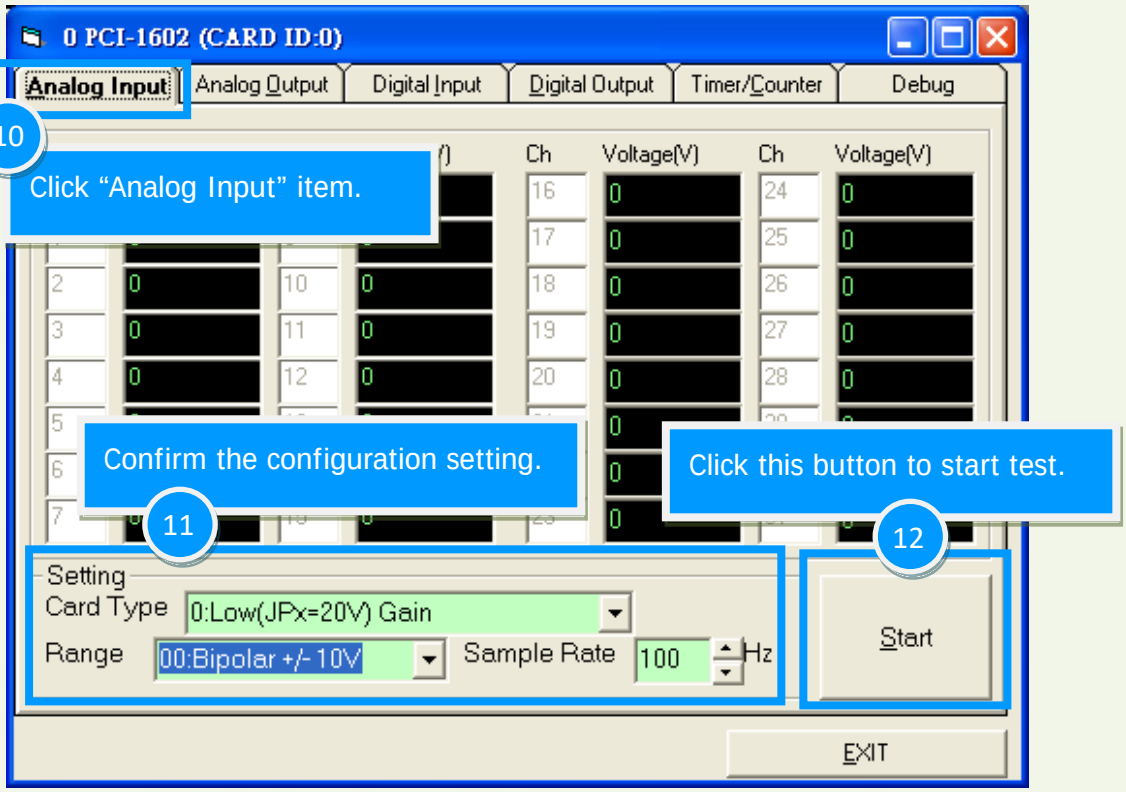
The PEX-1202L/H software is fully compatible with the PCI-1202 series software.

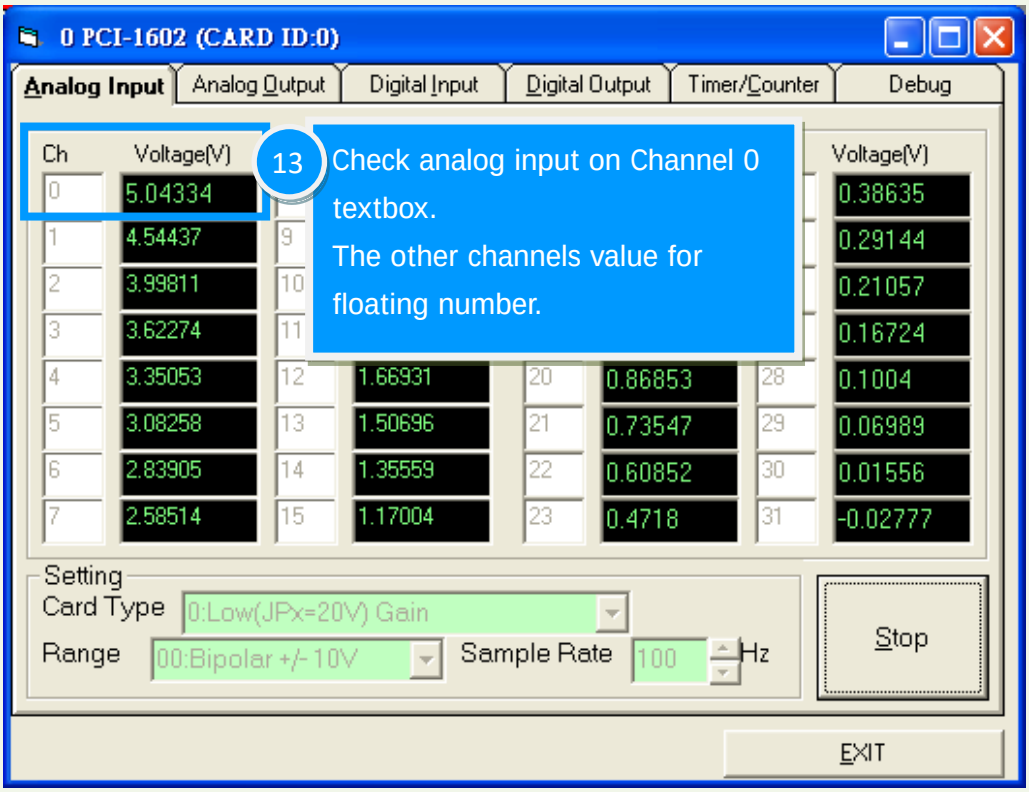
2. Get **Digital Output/Input Function** test result.



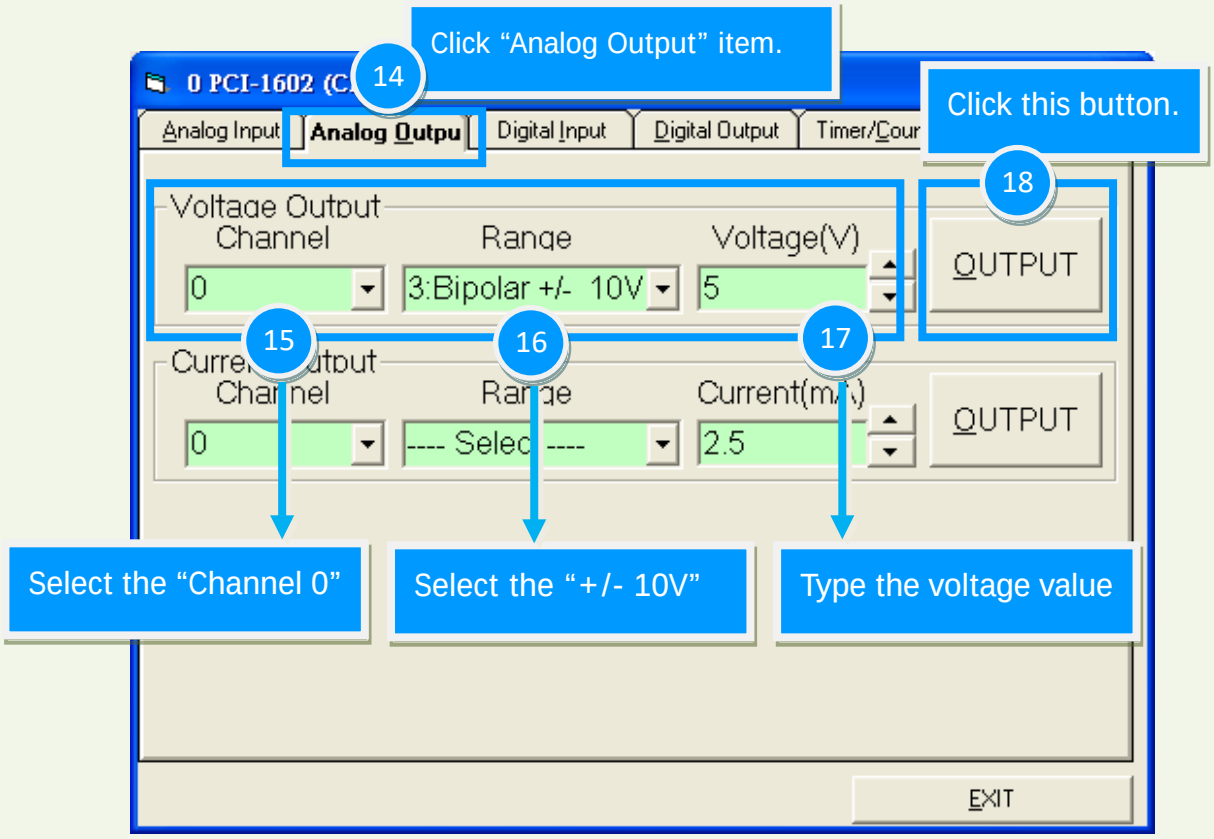


3. Get **Analog Input Function** test result.

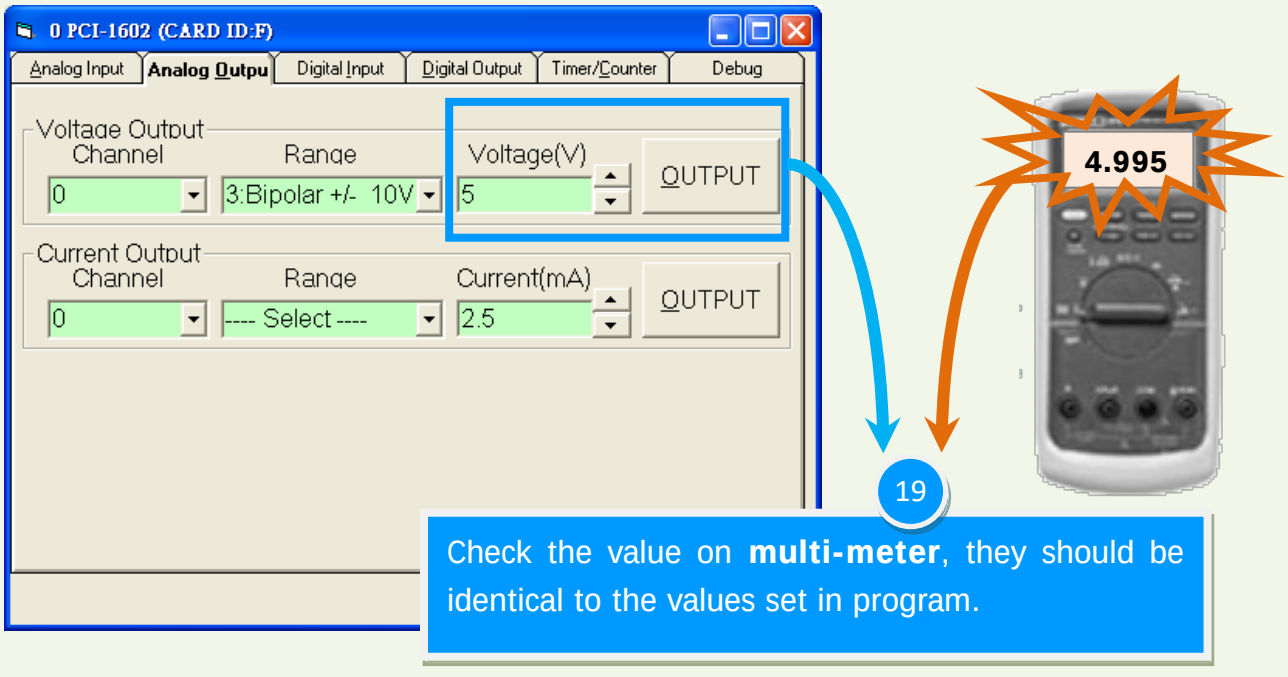




4. Get **Analog Output Function** test result.



The value read on meter may be a little difference from the DA value because of the resolution limit of meter or the measurement error.



6. I/O Control Register

6.1 How to Find the I/O Address

The plug&play BIOS will assign a proper I/O address to every PEX-1202 and PCI-1202/1602/1800/1802 series card in the power-on stage. The fixed IDs for the PEX-1202 and PCI-1202/1602/1800/1802 series card are given as follows:

Table 6-1:

PEX-1202/ PCI-1202 Series			
Vendor ID	0x1234	Sub-Vendor ID	0x0000
Device ID	0x5672	Sub-Device ID	0x0000
		Sub-Aux ID	0x00

Table 6-2:

PCI-1602 Series			
Vendor ID	0x1234	Sub-Vendor ID	0x0000
Device ID	0x5676	Sub-Device ID	0x0000
		Sub-Aux ID	0x00

Table 6-3:

PCI-1800/ PCI-1802 Series			
Vendor ID	0x1234	Sub-Vendor ID	0x0000
Device ID	0x5678	Sub-Device ID	0x0000
		Sub-Aux ID	0x00

PIO_PISO.EXE Utility for the Windows

The PIO_PISO.EXE utility program will detect and present all information for ICPDAS I/O cards installed in the PC, as shown in the following Figure6-1. Details of how to identify the PEX-1202 and PCI-1202/1602/1800/1802 series card of ICPDAS data acquisition boards based on the **Sub-vendor**, **Sub-device** and **Sub-Aux ID** are given in Table 6-1 to 6-3.

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/

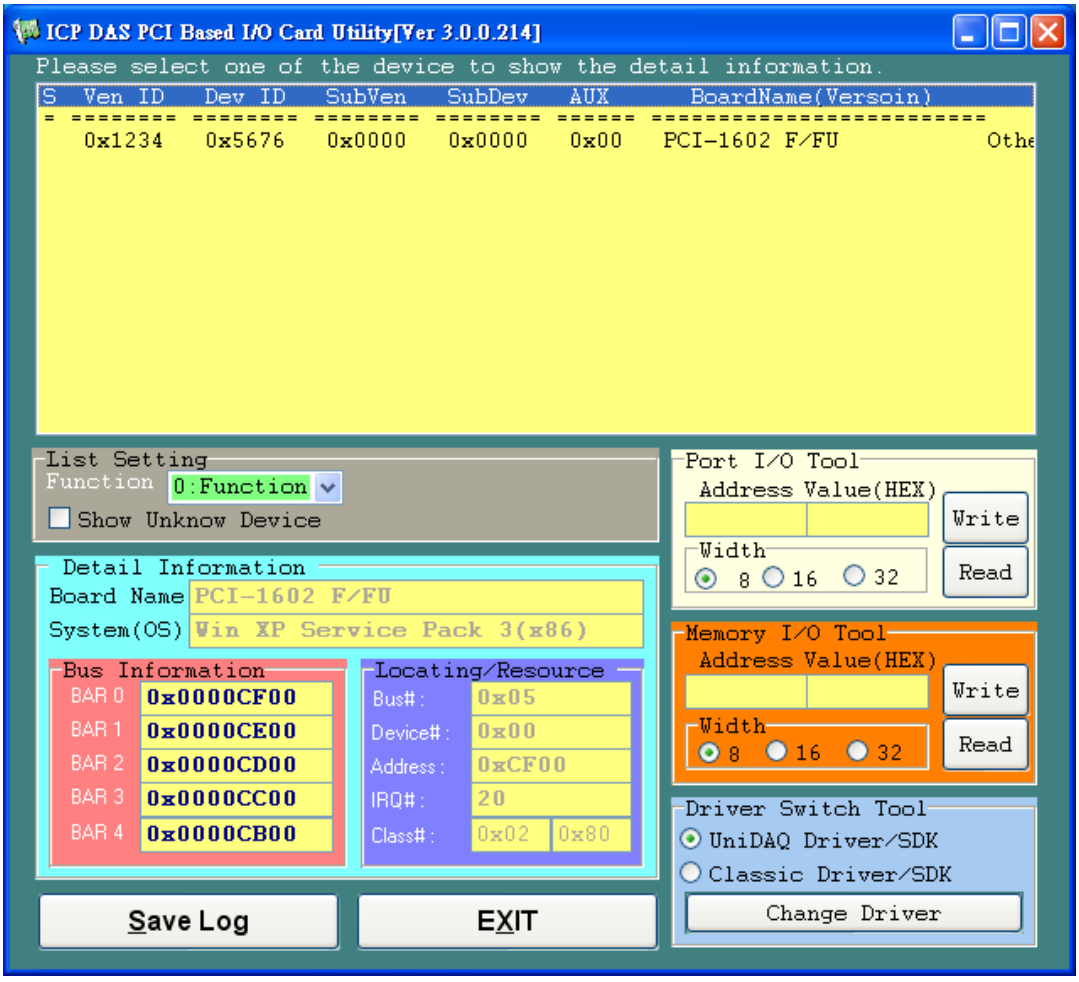


Figure 6-1

P180x_DriverInit(..) Function

The P180X_DriverInit(..) can detect how many PCI-1800/1802 cards in the system. Then the P180X_DriverInit(..) will detect the I/O address of these cards. The P180X_DriverInit(..) is implemented based on the PCI plug&play mechanism-2. The P180X_DriverInit(..) must be called once before all the other functions. The function P180X_DriverInit(..) is given as follows:

P180x_DriverInit(…) is designed for PCI-1800/1802 series
P1202_DriverInit(…) is designed for PEX-1202/PCI-1202 series
P1602_DriverInit(…) is designed for PCI-1602 series

This function can be detecting how many PEX-1202/PCI-1202/1602/180x series cards in the system. Also can be detect and save the I/O resource information of every PEX-1202 and PCI-1202/1602/180x series card.

The sample is given as follows:

```
wRetVal=P180X_DriverInit(&wBoards);/* call P180X_DriverInit(..) first */
printf("Threr are %d P180X Cards in this PC\n",wBoards);

/* dump every P180X card's configuration address space */
printf("The Configuration Space -> Timer Control DIO AD/DA \n");
for (i=0; i<wBoards; i++)
    {
    printf("Card %02d: %04xH %04xH %04xH %04xH\n", i,wConfigSpace[i][0],
            wConfigSpace[i][1], wConfigSpace[i][2],wConfigSpace[i][3]);
    }

/* The P180X_ActiveBoard() function must be used to active a board, */
/* then all operation will take effect to the active board. */
printf("Now Active First P180X Card...\n");
P180X_ActiveBoard( 0 );
```

6.2 The Assignment of I/O Address

The plug&play BIOS will assign the proper I/O address to PEX-1202/PCI-1202/1602/180x series card. If there is only one PEX-1202/PCI-1202/1602/180x series card, the user can identify the board_0. If there are two PEX-1202/PCI-1202/1602/180x series cards in the system, the user will be very difficult to identify which board is board_0. The software driver can support 16 boards max. Therefore the user can install 16 boards in one PC system.

The simplest way to find the board number is to use DEMO15.EXE given in DOS demo program. This demo program will send to D/O and read back from D/I. If the user installs a 20-pin flat cable between CON1 and CON2, the value read from D/I will be the same as D/O. The operation steps are given as follows:

1. Remove all 20-pin flat cable between CON1 and CON2
2. Install all PEX-1202/PCI-1202/1602/180x cards into the PC system
3. Power-on and run DEMO15.EXE
4. Now all D/I value will be different from D/O value
5. Install a 20-pin flat cable into CON1 and CON2 of any PEX-1202/PCI-1202/1602/180x card
6. There will be one card' s D/I value = D/O value, the card number is also shown in screen

Therefore the user can find the card number very easy if he install a 20-pin flat cable into PEX-1202/PCI-1202/1602/1800/1802 one-by-one.

6.3 The I/O Address Map

The plug&play BIOS will assign proper I/O address to each PEX-1202 and PCI-1202/1602/1800/1802 series card very well. There are five BAR of I/O address used by this card and each section can be assigned to an unused I/O space. The hardware I/O ports are described as follows:

Table 6-5:

Sec.	BAR	Offset	Name	Operation	Access
-	0	0h	PCI controller add-on mail box	W	32-bit
		38h	PCI interrupt control register	R/W	32-bit
		3Ch (or 3Eh, 3Fh)	On board NV-RAM access control register	R/W	32-bit (8-bit)
6.4	1	00h	8254 timer0	R/W	8/16/32-bit
		04h	8254 timer1	R/W	8/16/32-bit
		08h	8254 timer2	R/W	8/16/32-bit
		0Ch	8254 control	W	8/16/32-bit
6.5	2	00h	Control register	W	16/32-bit
		00h	Status register	R	8/16/32-bit
		04h	A/D software trigger	W	8/16/32-bit
6.6	3	00h	DI port	R	16-bit
		00h	DO port	W	16-bit
		04h	Read Card ID	R	4-bit
6.7	4	00h	A/D data port	R	16-bit
		00h	D/A channel 1	W	16-bit
		04h	D/A channel 2	W	16-bit

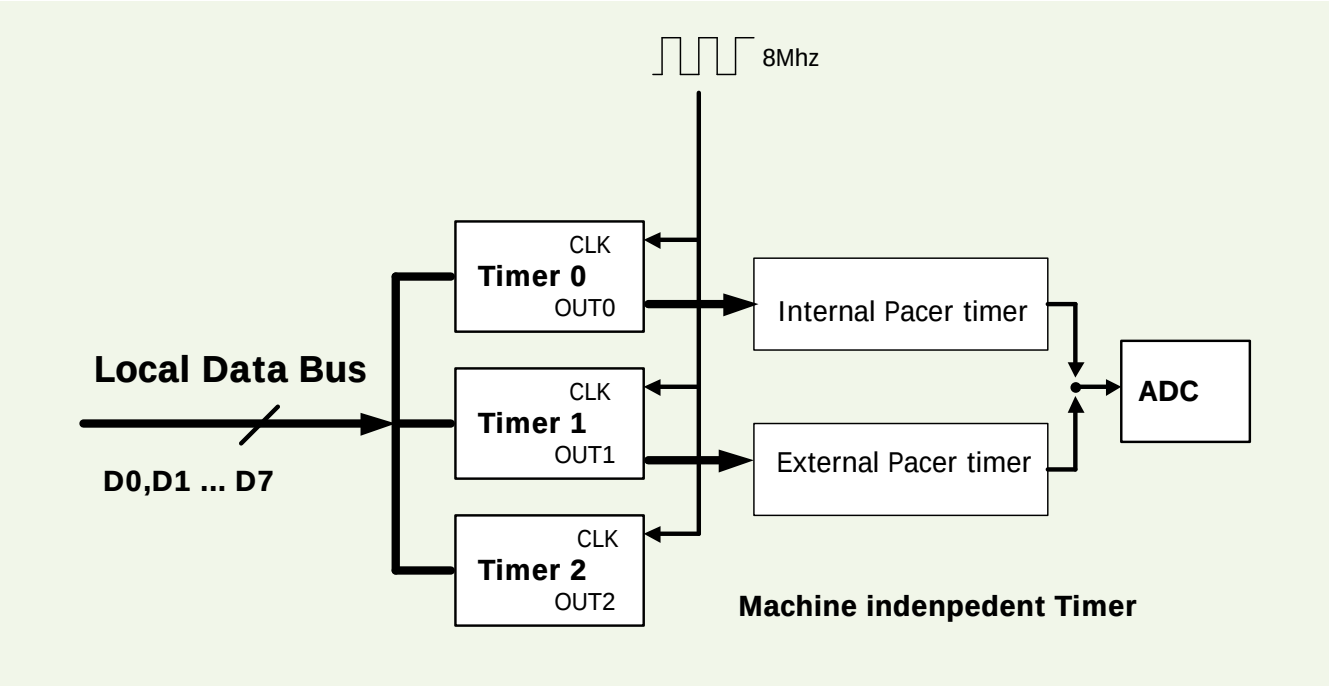
The driver name of these address are given as follows:

BAR1:	wAddrTimer	→	Save in wConfigSpace[Board][0]
BAR2:	wAddrCtrl	→	Save in wConfigSpace[Board][1]
BAR3:	wAddrDio	→	Save in wConfigSpace[Board][2]
BAR4:	wAddrAdda	→	Save in wConfigSpace[Board][3]

6.4 BAR1: Timer Control

The timer-0 is used as the internal A/D pacer trigger. The timer-1 is designed for the A/D pacer trigger in the external trigger mode. The timer-2 is used as the machine independent timer. **The timer-2 is very important for settling time delay.** Refer to Intel's "Microsystem Components Handbook" for 8254 programming. The block diagram of the 8254 timer is given as follows:

Figure 6.4-1: The Block diagram of 8254 timer for the PEX-1202/PCI-1202/1602/180x series.



The I/O address of 8254 timer is given as follows:

I/O address of timer/counter_0	→	wAddrTimer+0 *4
I/O address of timer/counter_1	→	wAddrTimer+1 *4
I/O address of timer/counter_2	→	wAddrTimer+2 *4
I/O address of control register	→	wAddrTimer+3 *4

timer0 → for Pacer Trigger

```
void enable_timer0(WORD divv)       // for internal pacer trigger
{
outport((WORD)(wAddrTimer+3*4), 0x34);      /* enable pacer timer_0 */
outport((WORD)(wAddrTimer+0*4), (WORD)(divv & 0xff));
outport((WORD)(wAddrTimer+0*4), (WORD)((divv>>8) & 0xff));
}

void disable_timer0(void)
{
outport((WORD)(wAddrTimer+3*4), 0x34);      /* disable pacer timer_0 */
outport((WORD)(wAddrTimer+0*4), 0x01);
outport((WORD)(wAddrTimer+0*4), 0x00);
}
```

timer1 → for External Trigger

```
void enable_timer1(WORD divv)                /* for external trigger pacertimer */
{
outport((WORD)(wAddrTimer+3*4), 0x74);    /* enable pacer timer_1 */
outport((WORD)(wAddrTimer+1*4), (WORD)(divv & 0xff));
outport((WORD)(wAddrTimer+1*4), (WORD)((divv>>8) & 0xff));
}

void disable_timer1(void)
{
outport((WORD)(wAddrTimer+3*4), 0x74);       /* disable timer_1 */
outport((WORD)(wAddrTimer+1*4), 0x01);
outport((WORD)(wAddrTimer+1*4), 0x00);
}
```

■ timer2 → for Machine Independent Timer

P180x_DelayUs(…) is designed for PCI-1800/1802 series.

P1202_DelayUs(…) is designed for PEX-1202/PCI-1202 series.

P1602_DelayUs(…) is designed for PCI-1602 series.

```
/* address of timer 2      = wAddrTimer+2*4
     address of ctrl         = wAddrTimer+3*4
     input clock             = 8 M
     down count 8 time           = 1 µs
     down count 65536/8 = 8192 µs --> max 8191 µs */
WORD    P180X_DelayUs(WORD wDelayUs)
{
WORD wDownCount,wLow,wHigh,wVal;
double fTimeOut;

if (wDelayUs>=8191) return(InvalidateDelay);
wDownCount=wDelayUs*8;
wLow=wDownCount&0xff;
wHigh=(wDownCount>>8)&0xff;
outport((wAddrTimer+3*4), 0xb0);                /* timer_2 mode_0 0xb0 */
outport((wAddrTimer+2*4), wLow);
outport((wAddrTimer+2*4), wHigh);

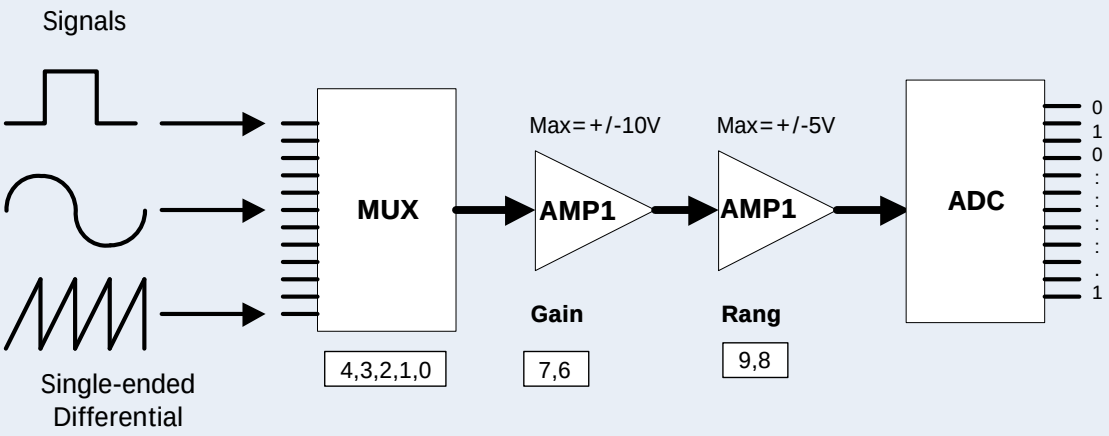
fTimeOut=1.0; // wait 1 to stop
for (;;)
{
        wVal=inport(wAddrCtrl)&0x01;
        if (wVal!=0) return(NoError);           /* if the timer is up, this bit will be 1 */
        fTimeOut+=1.0;
        if (fTimeOut>6553500.0)
        return(DelayTimeOut);
}
}
```

6.5 BAR2: Control Register

The I/O address of control register is given as follows:

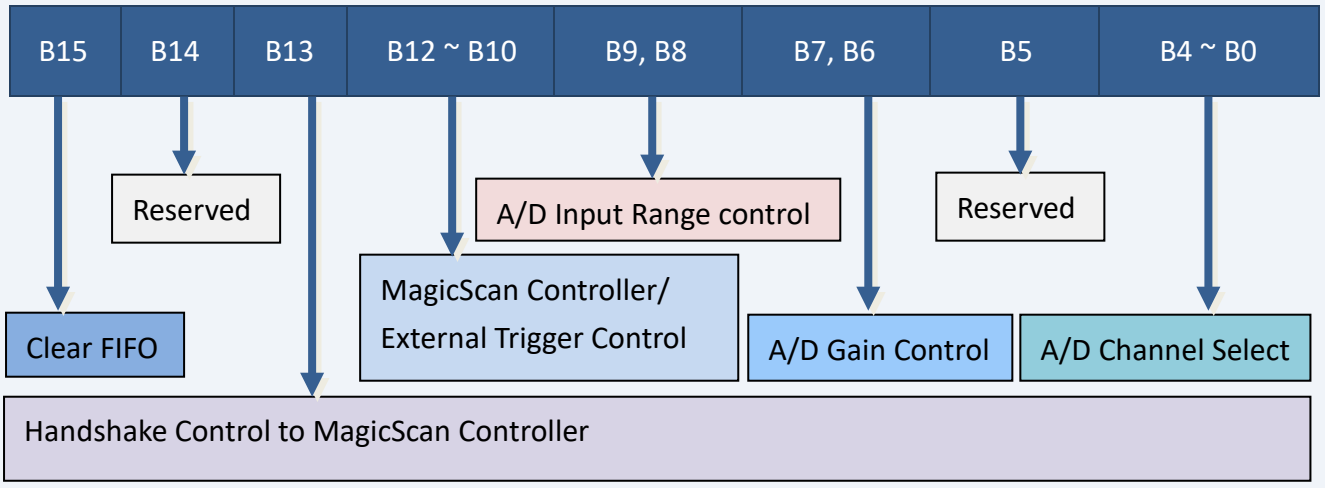
I/O address of control register	→	wAddrCtrl+0 *4
I/O address of status register	→	wAddrCtrl+0 *4
I/O address of trigger register	→	wAddrCtrl+1 *4

Figure 6.5-1: The flow path of analog input signal



6.5.1 The Control Register

The format of the control register is given as follows:



6.5.1.1 Bit4-Bit0: A/D Channel Select

A/D channel	B4	B3	B2	B1	B0	Boards
0	0	0	0	0	0	1202/1602/1800/1802
15	0	1	1	1	1	1202/1602/1800/1802
16	1	0	0	0	0	1202/1602/1802
31	1	1	1	1	1	1202/1602/1802

6.5.1.2 Bit6, Bit7, Bit8, Bit9: Configuration Table

- **Bit6, Bit7: Gain Control**

[B7, B6]	Low Gain	High Gain
[0, 0]	PGA=1	PGA=1
[0, 1]	PGA=2	PGA=10
[1, 0]	PGA=4	PGA=100
[1, 1]	PGA=8	PGA=1000

- **Bit8, Bit9: Input Range Control**

[B9, B8]	Unipolar/Biploar	Gain	Example
[0, 0]	Biploar	PGA=1	-5 V ~ +5 V
[1, 0]	Unipolar	PGA=1	0 V ~ +5 V
[0, 1]	Biploar	PGA=1/2	-10 V ~ + 10 V
[1, 1]	Unipolar	PGA=1/2	0 V ~ + 10 V

■ The configuration table of **PEX-1202L and PCI-1202/1800/1802(L/LU)** is given as follows:

Bipolar/Unipolar	Input Signal Range	Gain	Settling Time	[B9,B8,B7,B6]
Bipolar	+/- 5 V	1	3 µs	0000
Bipolar	+/- 2.5 V	2	3 µs	0001
Bipolar	+/- 1.25 V	4	3 µs	0010
Bipolar	+/- 0.625 V	8	3 µs	0011
Bipolar	+/- 10 V	0.5	3 µs	0100
Bipolar	+/- 5 V	1	3 µs	0101
Bipolar	+/- 2.5 V	2	3 µs	0110
Bipolar	+/- 1.25 V	4	3 µs	0111
Unipolar	0 V ~ 10 V	1	3 µs	1000
Unipolar	0 V ~ 5 V	2	3 µs	1001
Unipolar	0 V ~ 2.5 V	4	3 µs	1010
Unipolar	0 V ~ 1.25 V	8	3 µs	1011

■ The configuration table of **PEX-1202H and PCI-1202/1800/1802(H/HU)** is given as follows:

Bipolar/Unipolar	Input Signal Range	Gain	Settling Time	[B9,B8,B7,B6]
Bipolar	+/- 5 V	1	23 µs	0000
Bipolar	+/- 0.5 V	10	28 µs	0001
Bipolar	+/- 0.05 V	100	140 µs	0010
Bipolar	+/- 0.005 V	1000	1300 µs	0011
Bipolar	+/- 10 V	0.5	23 µs	0100
Bipolar	+/- 1 V	5	28 µs	0101
Bipolar	+/- 0.1 V	50	140 µs	0110
Bipolar	+/- 0.01 V	500	1300 µs	0111
Unipolar	0 V ~ 10 V	1	23 µs	1000
Unipolar	0 V ~ 1 V	10	28 µs	1001
Unipolar	0 V ~ 0.1 V	100	140 µs	1010
Unipolar	0 V ~ 0.01 V	1000	1300 µs	1011

Set Channel Configuration

The demo program to set the channel and gain is given as follows:

P180x_SetChannelConfig(…) is designed for PCI-1800/1802 series

P1202_SetChannelConfig(…) is designed for PEX-1202/PCI-1202 series

P1602_SetChannelConfig(…) is designed for PCI-1602 series

```
WORD P180X_SetChannelConfig(WORD wAdChannel, WORD wAdConfig)
{
  WORD wConfig,wChannel;

  wChannel      = (wAdChannel&0x1f);
  wSysConfig    = (wAdConfig&0x1f);         // store for P1802_AdPolling
  wConfig       = (wAdConfig&0x0f);
  wConfig       = wConfig << 6;
  wConfig    += wChannel;

  /*    Bit15=1 --> no reset FIFO
        Bit14=?
        Bit13=?
        Bit12=0 --> command [001] --> set channel&Config command
        Bit11=0
        Bit10=1
        Bit9 =B --> Range control code [BB] --> unipolar/bipolar & divided by 2
        Bit8 =B
        Bit7 =B --> gain control code [BB]    --> 1/10/100/1000 or 1/2/4/8
        Bit6 =B
        Bit5 =?
        Bit4-Bit0 --> channel number */
        wConfig+= 0x8400;         // this is set channel config command
        return(pic_control(wConfig));
}
```

Calculate the A/D Value

The demo program to calculate the real A/D value is given as follows:

```
double ComputeRealValue(DWORD dwAdConfig, DWORD dwAdHex)
{
  WORD wZERO;
  double dfMAX, dfVal;

  switch (dwAdConfig)
  {
      case 0 : wZERO=2048;    dfMAX=5.0;      break;
      case 1 : wZERO=2048;    dfMAX=2.5;      break;
      case 2 : wZERO=2048;    dfMAX=1.25;     break;
      case 3 : wZERO=2048;    dfMAX=0.625;    break;
      case 4 : wZERO=2048;    dfMAX=10.0;     break;
      case 5 : wZERO=2048;    dfMAX=5.0;      break;
      case 6 : wZERO=2048;    dfMAX=2.5;      break;
      case 7 : wZERO=2048;    dfMAX=1.25;     break ;
      case 8 : wZERO=0;    dfMAX=10.0/2.0;    break;
      case 9 : wZERO=0;    dfMAX=5.0/2.0;     break;
      case 10: wZERO=0;    dfMAX=2.5/2.0;     break;
      case 11: wZERO=0;    dfMAX=1.25/2.0;    break;
```

For PEX-1202L and PCI-1202/1800/1802(L/LU)

Note:
B4=0 is used to identify PGL

```
      case 0x10 : wZERO=2048;    dfMAX=5.0;       break;
      case 0x11 : wZERO=2048;    dfMAX=0.5;       break;
      case 0x12 : wZERO=2048;    dfMAX=0.05;      break;
      case 0x13 : wZERO=2048;    dfMAX=0.005;     break;
      case 0x14 : wZERO=2048;    dfMAX=10.0;      break;
      case 0x15 : wZERO=2048;    dfMAX=1.0;       break ;
      case 0x16 : wZERO=2048;    dfMAX=0.1;       break;
      case 0x17 : wZERO=2048;    dfMAX=0.01;      break;
      case 0x18 : wZERO= 0;    dfMAX=10.0/2.0;    break ;
      case 0x19 : wZERO= 0;    dfMAX=1.0/2.0;     break;
      case 0x1A : wZERO= 0;    dfMAX=0.1/2.0;     break;
      case 0x1B : wZERO= 0;    dfMAX=0.01/2.0;    break;
```

For PEX-1202H and PCI-1202/1800/1802(H/HU)

Note:
B4=1 is used to identify PGH

```
      default: return(ConfigCodeError); }
dfVal=(((double)(wAdHex)-wZERO)/2048.0)*dfMAX;
return(dfVal);
}
```

6.5.1.3 Bit12-Bit10: Command Sets of MagicScan Controller

The command sets of MagicScan controller are given as follows:

Command	[B12, B11, B10]	Descriptions
Reset	[0 0 0]	Reset the MagicScan controller. **The software driver must send this command once after power-on.**
Set channel/gain	[0 0 1]	Set the channel/gain value of the **fixed-channel mode.** It will not affect the scan queue.
Add to scan queue	[1 0 0]	Add the channel/gain code to the scan queue. (At most 48 scan-channels can be stored in the MagicScan controller.)
Start MagicScan	[1 0 1]	Start the MagicScan controller
Stop MagicScan	[0 1 0]	Stop the MagicScan controller.
Get ODM number	[1 1 0]	Get the ODM number of the PEX-1202 and PCI-1202/1602/1800/1802 series card.

- **The demo program to reset the MagicScan controller is given as follows:**

```
wVal=pic_control(0xC000);                    /* 11?0 00?? ???? ???? cmd_000=reset */
```

- **The demo program to clear MagicScan queue is given as follows:**

P180x_ClearScan(…) is designed for PCI-1800/1802 series
P1202_ClearScan(…) is designed for PEX-1202/PCI-1202 series
P1602_ClearScan(…) is designed for PCI-1602 series

```
WORD    P180X_ClearScan(void)
{
WORD i;
for(i=0; i<32; i++)    wMagicScanSave[i]=0;
disable_timer0();
disable_timer1();
return(pic_control(0xC000));                    /* 11?0 00?? ???? ???? cmd_000=reset */
}
```

- **The demo program of send command to MagicScan control is given as follows:**

```
WORD pic_control(WORD i)
{
WORD j;

if    ((inport(wAddrCtrl)&0x04)==0)
      {
      outport(wAddrCtrl,0xffff);                    /* send a recovery to PIC */
      }

j=0;
while ((inport(wAddrCtrl)&0x04)==0)
       {
       j++;
       if    (j>65530) return(AdControllerError);    /* time out */
       }

i = i & 0xDFFF;                                     /* set pic low !!    */
outport(wAddrCtrl,i);

j=0;
while ((inport(wAddrCtrl)&0x04)!=0)
       {
       j++;
       if (j>65530) return(AdControllerError);    /* time out */
       }

outport(wAddrCtrl,(WORD)(i | 0x2000));    /* set pic high !! */
j=0;
while ((inport(wAddrCtrl)&0x04)==0)
       {
       j++;
       if (j>65530) return(AdControllerError);    /* time out */
       }
return(NoError);
}
```

- **The demo program to set the channel and gain is given as follows:**

P180x_SetChannelConfig(…) is designed for PCI-1800/1802 series
P1202_SetChannelConfig(…) is designed for PEX-1202/PCI-1202 series
P1602_SetChannelConfig(…) is designed for PCI-1602 series

```
WORD P180X_SetChannelConfig(WORD wAdChannel, WORD wAdConfig)
{
WORD wConfig,wChannel;

wChannel     = (wAdChannel&0x1f);
wSysConfig   = (wAdConfig&0x1f);        // store for P1802_AdPolling
wConfig       = (wAdConfig&0x0f);
wConfig       = wConfig << 6;
wConfig      += wChannel;

/*   Bit15=1 --> no reset FIFO
     Bit14=?
     Bit13=?
     Bit12=0 --> command [001] --> set channel&Config command
     Bit11=0
     Bit10=1
     Bit9 =B --> Range control code [BB] --> unipolar/bipolar & divided by 2
     Bit8 =B
     Bit7 =B --> gain control code [BB]    --> 1/10/100/1000 or 1/2/4/8
     Bit6 =B
     Bit5 =?
     Bit4-Bit0 --> channel number */
     wConfig+= 0x8400;     // this is set channel config command
     return(pic_control(wConfig));
}
```

- **The demo program to add to MagicScan queue is given as follows:**

P180x_AddToScan(…) is designed for PCI-1800/1802 series
P1202_AddToScan(…) is designed for PEX-1202/PCI-1202 series
P1602_AddToScan(…) is designed for PCI-1602 series

```
WORD P180X_AddToScan(WORD wAdChannel, WORD wAdConfig, WORD
wAverage, WORD wLowAlarm, WORD wHighAlarm, WORD wAlarmType)
{WORD wConfig,wChannel,wRetVal;

if (wAlarmType>=5) return(AlarmTypeError);
wMagicLowAlarm[wMP]=wLowAlarm;
wMagicHighAlarm[wMP]=wHighAlarm;
wMagicAlarmType[wMP]=wAlarmType;
wChannel      = wAdChannel&0x1f;
wMagicChannel[wMP]=wChannel;
wSysConfig = wAdConfig&0x1f;                /* Store for P180X_AdPolling */
wMagicConfig[wMP]=wSysConfig;
wMagicAve[wMP]=wAverage;
wConfig       = wAdConfig&0x0f;
wConfig       = wConfig << 6;
wConfig      += wChannel;

/*    Bit15=1 --> no reset FIFO
        Bit14=1
        Bit13=?
        Bit12=1 --> command [100] --> add_to_scan command
        Bit11=0
        Bit10=0
        Bit9 =B --> Range control code [BB] --> unipolar/bipolar & divided by 2
        Bit8 =B
        Bit7 =B --> gain control code [BB]    --> 1/10/100/1000 or 1/2/4/8
        Bit6 =B
        Bit5 =?
        Bit4-Bit0 --> channel number    */
wConfig+= 0xD000;             /* this is add_to_scan_queue command */
wRetVal=pic_control(wConfig);
if (wRetVal!=0) return(wRetVal);
return(NoError);
}
```

- **The demo program to start MagicScan operation is given as follows:**

P180x_StartScan(…) is designed for PCI-1800/1802 series
P1202_StartScan(…) is designed for PEX-1202/PCI-1202 series
P1602_StartScan(…) is designed for PCI-1602 series

```
WORD    P180X_StartScan(WORD wSampleRate, WORD wNum)
{
WORD wVal;
WORD wRetVal;

wMagicNum=wNum;
disable_timer0();                    /* Disable pacer timer first */
                                     /* start MagicScan controller */
wRetVal=pic_control(0xD400);      /* 11?1 01?? ???? ???? cmd_101=start scan */
if (wRetVal!=0) return(wVal);

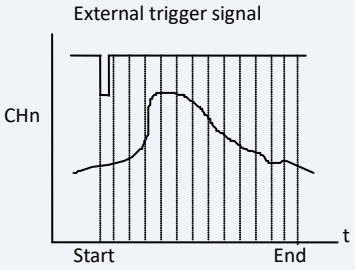
                                      /*    Clear FIFO to clear all data */
outport(wAddrCtrl,0x2000);           /* Bit15=0=clear FIFO, Bit13=1=not PIC cmd    */
outport(wAddrCtrl,0xA000);          /* Bit15=1=no reset FIFO, BIT13=1=not PIC cmd */

enable_timer0(wSampleRate);         /* Enable pacer timer, sampling rate=8 M/dwSample */
magic_scan();                       /* Call MagicScan subroutine(DOS) or thread(Windows) */
return(NoError);
}
```

6.5.1.4 Bit12-Bit10: External Trigger Control

- The operation steps of **post-trigger** are given as follows:

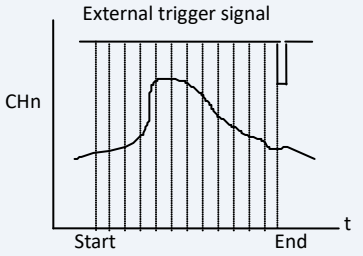
Step 1: Disable all external trigger
Step 2: Set the pacer rate of timer-1
Step 3: Clear FIFO & disable timer-1
Step 4: Wait until external trigger signal to enable timer-1
Step 5: Fetch N data(N=End-Start)
Step 6: Stop all timer



External post-trigger timing

- The operation steps of **pre-trigger** are given as follows:

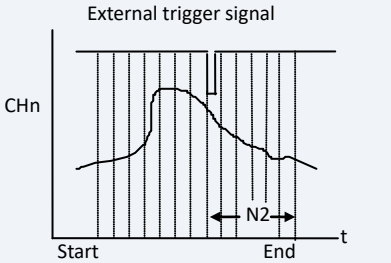
Step 1: Disable all external trigger
Step 2: Set the pacer rate of timer-1
Step 3: Clear FIFO & enable timer-1
Step 4: Circular-fetch N-data until external trigger
signal to disable timer-1 (N=End-Start)
Step 5: Stop all timer
NOTE: The circular-fetch operation is performed by software



External pre-trigger timing diagram

- The operation steps of **middle-trigger** are given as follows:

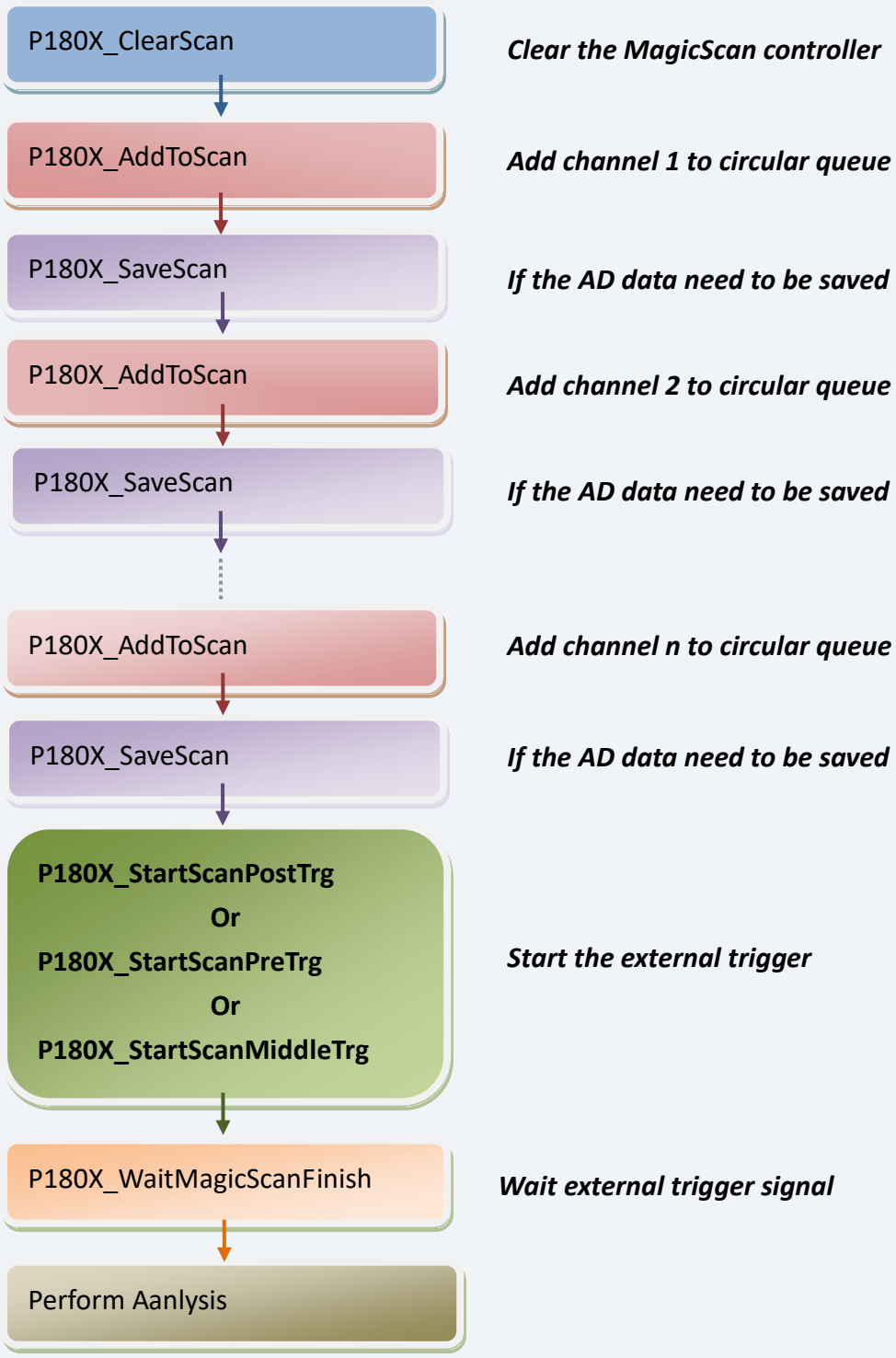
Step 1: Disable all external trigger
Step 2: Set the pacer rate of timer-1
Step 3: Clear FIFO & enable timer-1
Step 4: Circular-fetch N-data until external trigger signal (N=End-Start)
Step 5: Fetch more N2-data & stop timer-1
Step 6: Stop all timer
NOTE: The circular-fetch operation is performed by software



External middle-trigger timing diagram

Note 1: The external trigger operation must use with the MagicScan controller. The software flowchart of external trigger is given in next page.
Note 2: The post-trigger operation can use all MagicScan function.
Note 3: The user can't enable MagicScan HI/LO alarm and digital filter function in the pre-trigger & middle-trigger operation.

- The software flowchart of external trigger operation is given as follows:



- Refer to chapter-7 for more information.
- This flowchart is validate for PEX-1202 and PCI-1202/1602/18001802 series

- **The demo program of post-trigger is given as follows:**

```
wRetVal=P180X_ClearScan();
wRetVal += P180X_AddToScan(0,0,1,0,0,0);      // CH:0 to scan
wRetVal += P180X_SaveScan(0,wV0);
wRetVal += P180X_AddToScan(2,0,1,0,0,0);      // CH:2 to scan
wRetVal += P180X_SaveScan(1,wV2);             // Notice: 1 not 2
                                              //    ^ : This is a ordinal number in
                                              //      Scan Queue not a channel number.
 wRetVal += P180X_StartScanPostTrg(wSampleRateDiv,DATALENGTH,nPriority);

if (wRetVal==0) sprintf(cShow,"2. External Post-Trigger Setup OK");
else sprintf(cShow,"2.    External Post-Trigger Setup Error");
TextOut(hdc,x*dx,(y+iLine)*dy,cShow,strlen(cShow)); iLine++;

for (; ;)
    {
    P180X_ReadScanStatus(&wStatus,&dwLowAlarm,&dwHighAlarm);
    if (wStatus>1) break;
    Sleep(10);
    }

sprintf(cShow,"3. ScanStatus=%x",wStatus);
TextOut(hdc,x*dx,(y+iLine)*dy,cShow,strlen(cShow)); iLine++;

wRetVal=P180X_StopMagicScan();

if   (wRetVal!=NoError)
      {
      sprintf(cShow,"4. StopMagicScan Error");
      TextOut(hdc,x*dx,(y+iLine)*dy,cShow,strlen(cShow)); iLine++;
      for (dwI=0; dwI<100; dwI++) Beep(10,10);
      }
SHOW_WAVE(hwnd,LINE1,wV0,1);
SHOW_WAVE(hwnd,LINE2,wV2,1);
```

★ *Refer to **DEMO23.C** for completely source program*

- The **B13 must set to 1** to set the external trigger logic.
  The external trigger controller commands are given as follows:

Trigger	Command sequences [B12, B11, B10]	Descriptions
Disable external trigger (for PEX-1202 and PCI-1800/1202/1602)	[1, 0, X]	Disable all external trigger.
Post-trigger (for PEX-1202 and PCI-1202/1602/180x)	[1, 0, X] [1, 0, X] [1, 1, 1] [1, 0, X]	(1) disable all external trigger (2) set pacer time-1 (3) clear FIFO and disable timer-1 (4) waiting for external signal to enable timer-1 (5) fetch N data (6) stop all timer and disable all external trigger
Pre-trigger (for PEX-1202 and PCI-1202/1602/180x/ ver-F)	[1, 0, X] [0, 1, X] [1, 1, 0] [1, 0, X]	(1) disable all external trigger (2) set pacer timer-1 (3) clear FIFO and enable timer-1 (4) waiting for the external signal to stop timer-1. (5) circular-fetch the last N data (6) stop all timer and disable all trigger
Middle-trigger (for PEX-1202 and PCI-1202/1602/180x/ ver-F)	[1, 0, X] [0, 1, X] [1, 1, 1] [1, 0, X]	(1) disable all external trigger (2) set pacer timer-1 (3) clear FIFO and enable timer-1 (4) waiting for the external signal. (5) fetch more N2 data (circular-fetch) (6) stop all timer and disable all trigger
Pre-trigger (for PCI-180/1802) **(version-C)**	[1, 0, X] [0, 1, X] [1, 1, 1] [1, 0, X]	(1) disable all external trigger (2) set pacer timer-1 (3) clear FIFO and enable timer-1 (4) waiting for the external signal to stop timer-1. (5) keep the last N data (circular-fetch) (6) stop all timer and disable all trigger
Middle-trigger (for PCI-1800/1802) **(version-C)**	[1, 0, X] [0, 1, X] [1, 1, 0] [0, 1, X] [1, 0, X]	(1) disable all external trigger (2) set pacer timer-1 (3) clear FIFO and enable timer-1 (4) waiting for the external signal to stop timer-1 (5) enable timer-1 (6) fetch more N2 data (7) stop all timer and disable all trigger

- **The source code of the post-trigger function (Windows version) is given as follows:**

P180x_StartScanPostTrg(…) is designed for PCI-1800/1802 series
P1202_StartScanPostTrg(…) is designed for PEX-1202 and PCI-1202 series
P1602_StartScanPostTrg(…) is designed for PCI-1602 series

```
WORD CALLBACK P180X_StartScanPostTrg(WORD wSampleRateDiv, DWORD dwNum, SHORT
nPriority)
{
disable_timer0();    // disable internal pacer timer
disable_timer1();    // disable external pacer timer

wVal=pic_control(0xD400);    /* 11?1 01?? ???? ???? cmd_101=start scan */
if (wVal!=0) return(wVal);

_outpw(wAddrCtrl,0xf000);              // 1. disable all external trigger
enable_timer1(wSampleRateDiv);         // 2. Sampling rate=8M/dwSampleRateDiv
_outpw(wAddrCtrl,0x7000);              // 3. B15=0,S2=1,S1=S0=0 --> clr FIFO
_outpw(wAddrCtrl,0xf000);              // 3. B15=1,S2=1,S1=S0=0 --> disable timer-1
_outpw(wAddrCtrl,0xfc00);              // 4. S2=1, S1=1, S0=1 --> wait ext signal to
                                       // enable timer-1

// create magicscan thread
InitializeCriticalSection(&MagicScan_CS);
wThreadStatus=0; wAskThreadStop=0;
hThread=CreateThread(NULL,0,(LPTHREAD_START_ROUTINE)magic_scan,
                              NULL, 0,&dwThreadID);      // can use all MagicScan functions
SetThreadPriority(hThread,nPriority);
i=0;
for(;;)
        {
        EnterCriticalSection(&MagicScan_CS);
        j=wThreadStatus;
        LeaveCriticalSection(&MagicScan_CS);
        if (j!=0) break;
        i++; Sleep(1);
        if (i>1000) return(ThreadCreateError);
        }
return(NoError);
}
```

■ **The source code of the pre-trigger function (Windows version) is given as follows:**

P180x_StartScanPostTrg(…) is designed for PCI-1800/1802 series
P1202_StartScanPostTrg(…) is designed for PEX-1202 and PCI-1202 series
P1602_StartScanPostTrg(…) is designed for PCI-1602 series

```
WORD CALLBACK P180X_StartScanPreTrg(WORD wSampleRateDiv, DWORD dwNum, SHORT
nPriority)
{
disable_timer0();    // disable internal pacer timer
disable_timer1();    // disable external pacer timer

wVal=pic_control(0xD400);      /* 11?1 01?? ???? ???? cmd_101=start scan */
if (wVal!=0) return(wVal);

_outpw(wAddrCtrl,0xf000);              // 1. disable all external trigger
enable_timer1(wSampleRateDiv);         // 2. Sampling rate=8M/dwSampleRateDiv
_outpw(wAddrCtrl,0x6800);              // 3. B15=0,S2=0,S1=1,S0=0 --> clr FIFO
_outpw(wAddrCtrl,0xE800);              // 3. B15=1,S2=0,S1=1,S0=0 --> enable timer-1
_outpw(wAddrCtrl,0xF800);              // 4. S2=1; S1=1; S0=0 --> wait ext signal to
                                       // disable timer-1

// create magicscan thread
InitializeCriticalSection(&MagicScan_CS);
wThreadStatus=0; wPreMid=0; wAskThreadStop=0;     // pre-trigger
hThread=CreateThread(NULL,0,(LPTHREAD_START_ROUTINE)
                          magic_scan_pre_mid_trg, NULL, 0,&dwThreadID);
SetThreadPriority(hThread,nPriority);    // can not use HI/LO alarm & digital filter
i=0;
for(;;)
        {
        EnterCriticalSection(&MagicScan_CS);
        j=wThreadStatus;
        LeaveCriticalSection(&MagicScan_CS);
        if (j!=0) break;
        i++; Sleep(1);
        if (i>1000) return(ThreadCreateError);
        }
return(NoError);
}
```

- **The source code of the middle-trigger function (Windows version) is given as follows:**

P180x_StartScanPostTrg(…) is designed for PCI-1800/1802 series

P1202_StartScanPostTrg(…) is designed for PEX-1202 and PCI-1202 series

P1602_StartScanPostTrg(…) is designed for PCI-1602 series

```
WORD CALLBACK P180X_StartScanMiddleTrg(WORD wSampleRateDiv, DWORD dwNum, SHORT
nPriority)
{
disable_timer0();    // disable internal pacer timer
disable_timer1();    // disable external pacer timer

wVal=pic_control(0xD400);    /* 11?1 01?? ???? ???? cmd_101=start scan */
if (wVal!=0) return(wVal);

_outpw(wAddrCtrl,0xf000);              // 1. disable all external trigger
enable_timer1(wSampleRateDiv);         // 2. Sampling rate=8M/dwSampleRateDiv
_outpw(wAddrCtrl,0x6800);              // 3. B15=0,S2=0,S1=1,S0=0 --> clr FIFO
_outpw(wAddrCtrl,0xE800);              // 3. B15=1,S2=0,S1=1,S0=0 --> enable timer-1
_outpw(wAddrCtrl,0xFC00);              // 4. S2=1; S1=1; S0=1 --> wait ext signal

// create magicscan thread
InitializeCriticalSection(&MagicScan_CS);
wThreadStatus=0; wPreMid=1; wAskThreadStop=0;    // middle-trigger
hThread=CreateThread(NULL,0,(LPTHREAD_START_ROUTINE,
                             magic_scan_pre_mid_trg, NULL, 0,&dwThreadID);
SetThreadPriority(hThread,nPriority);    // can not use HI/LO alarm & digital filter

i=0;
for(;;)
        {
        EnterCriticalSection(&MagicScan_CS);
        j=wThreadStatus;
        LeaveCriticalSection(&MagicScan_CS);
        if (j!=0) break;
        i++; Sleep(1);
        if (i>1000) return(ThreadCreateError);
        }
return(NoError);
}
```

■ **The source code of the pre-trigger function for PCI-1800/1802/ver-C is given as follows:**

This driver is designed for PCI-1800/1802 version-C

```
WORD CALLBACK P180X_StartScanPreTrgVerC(WORD wSampleRateDiv, DWORD dwNum, SHORT
nPriority)
{
disable_timer0();    // disable internal pacer timer
disable_timer1();    // disable external pacer timer

wVal=pic_control(0xD400);    /* 11?1 01?? ???? ???? cmd_101=start scan */
if (wVal!=0) return(wVal);

_outpw(wAddrCtrl,0xf000);              // 1. disable all external trigger
enable_timer1(wSampleRateDiv);         // 2. Sampling rate=8M/dwSampleRateDiv
_outpw(wAddrCtrl,0x6800);              // 3. B15=0,S2=0,S1=1,S0=0 --> clr FIFO
_outpw(wAddrCtrl,0xE800);              // 3. B15=1,S2=0,S1=1,S0=0 --> enable timer-1
_outpw(wAddrCtrl,0xF800);              // 4. S2=1; S1=1; S0=0 --> wait ext signal to
                                       // disable timer-1

// create magicscan thread
InitializeCriticalSection(&MagicScan_CS);
wThreadStatus=0; wPreMid=0; wAskThreadStop=0;    // pre-trigger
hThread=CreateThread(NULL,0,(LPTHREAD_START_ROUTINE,
                             magic_scan_pre_mid_trg_ver_c, NULL, 0,&dwThreadID);
SetThreadPriority(hThread,nPriority);
i=0;
for(;;)
        {
        EnterCriticalSection(&MagicScan_CS);
        j=wThreadStatus;
        LeaveCriticalSection(&MagicScan_CS);
        if (j!=0) break;
        i++; Sleep(1);
        if (i>1000) return(ThreadCreateError);
        }
return(NoError);
}
```

■ **The source code of the middle-trigger function for PCI-1800/1802/ver-C is given as follows:**

This driver is designed for PCI-1800/1802 version-C

```
WORD CALLBACK P180X_StartScanMiddleTrgVerC(WORD wSampleRateDiv, DWORD dwNum,
SHORT nPriority)
{
disable_timer0();    // disable internal pacer timer
disable_timer1();    // disable external pacer timer

wVal=pic_control(0xD400);      /* 11?1 01?? ???? ???? cmd_101=start scan */
if (wVal!=0) return(wVal);

_outpw(wAddrCtrl,0xf000);              // 1. disable all external trigger
enable_timer1(wSampleRateDiv);         // 2. Sampling rate=8M/dwSampleRateDiv
_outpw(wAddrCtrl,0x6800);              // 3. B15=0,S2=0,S1=1,S0=0 --> clr FIFO
_outpw(wAddrCtrl,0xE800);              // 3. B15=1,S2=0,S1=1,S0=0 --> enable timer-1
_outpw(wAddrCtrl,0xF800);              // 4. S2=1; S1=1; S0=0 --> wait ext signal to
                                       // disable timer-1

// create magicscan thread
InitializeCriticalSection(&MagicScan_CS);
wThreadStatus=0; wPreMid=1; wAskThreadStop=0; // middle-trigger
hThread=CreateThread(NULL,0,(LPTHREAD_START_ROUTINE,
                        magic_scan_pre_mid_trg_ver_c, NULL, 0,&dwThreadID);
SetThreadPriority(hThread,nPriority);
i=0;
for(;;)
        {
        EnterCriticalSection(&MagicScan_CS);
        j=wThreadStatus;
        LeaveCriticalSection(&MagicScan_CS);
        if (j!=0) break;
        i++; Sleep(1);
        if (i>1000) return(ThreadCreateError);
        }
return(NoError);
}
```

The external trigger functions are given as follows:

Driver Name	Demo program	Applications
P180X_StartScanPostTrg(…)	demo23.c	for PCI-1800/1802 ver-C & ver-F
P180X_StartScanPreTrg(…)	demo24.c	for PCI-1800/1802 ver-F
P180X_StartScanMiddleTrg(…)	demo25.c	for PCI-1800/1802 ver-F
P180X_StartScanPreTrgOld(…)	demo26.c	for PCI-1800/1802 ver-C
P180X_StartScanMiddleTrgOld(…)	demo27.c	for PCI-1800/1802 ver-C
P1202_StartScanPostTrg(…)	demo23.c	for PEX-1202/PCI-1202
P1202_StartScanPreTrg(…)	demo24.c	for PEX-1202/PCI-1202
P1202_StartScanMiddleTrg(…)	demo25.c	for PEX-1202/PCI-1202
P1602_StartScanPostTrg(…)	demo23.c	for PCI-1602
P1602_StartScanPreTrg(…)	demo24.c	for PCI-1602
P1602_StartScanMiddleTrg(…)	demo25.c	for PCI-1602

6.5.1.5 Bit15: Clear FIFO Bit

The **B15 is used to reset the on-board FIFO**. When set to low, FIFO will be cleared. The FIFO must be cleared once after power-on.

Command	[B15]
Clear FIFO	0
No Reset FIFO	1

The demo program of handshaking is given as follows:

```
// Clear FIFO to clear all data
outport(wAddrCtrl,0x2000);        /* Bit15=0=clear FIFO, Bit13=1=not PIC cmd    */
outport(wAddrCtrl,0xA000);        /* Bit15=1=no reset FIFO, Bit13=1=not PIC cmd    */
```

6.5.1.6 Bit13: Handshake Control Bit

Set the B13 to 0 if the command is sent to the MagicScan controller. Keep this bit at high when not used.

Command	[B13]
Handshake Control to MagicScan Controller	0
Set the external trigger logic	1

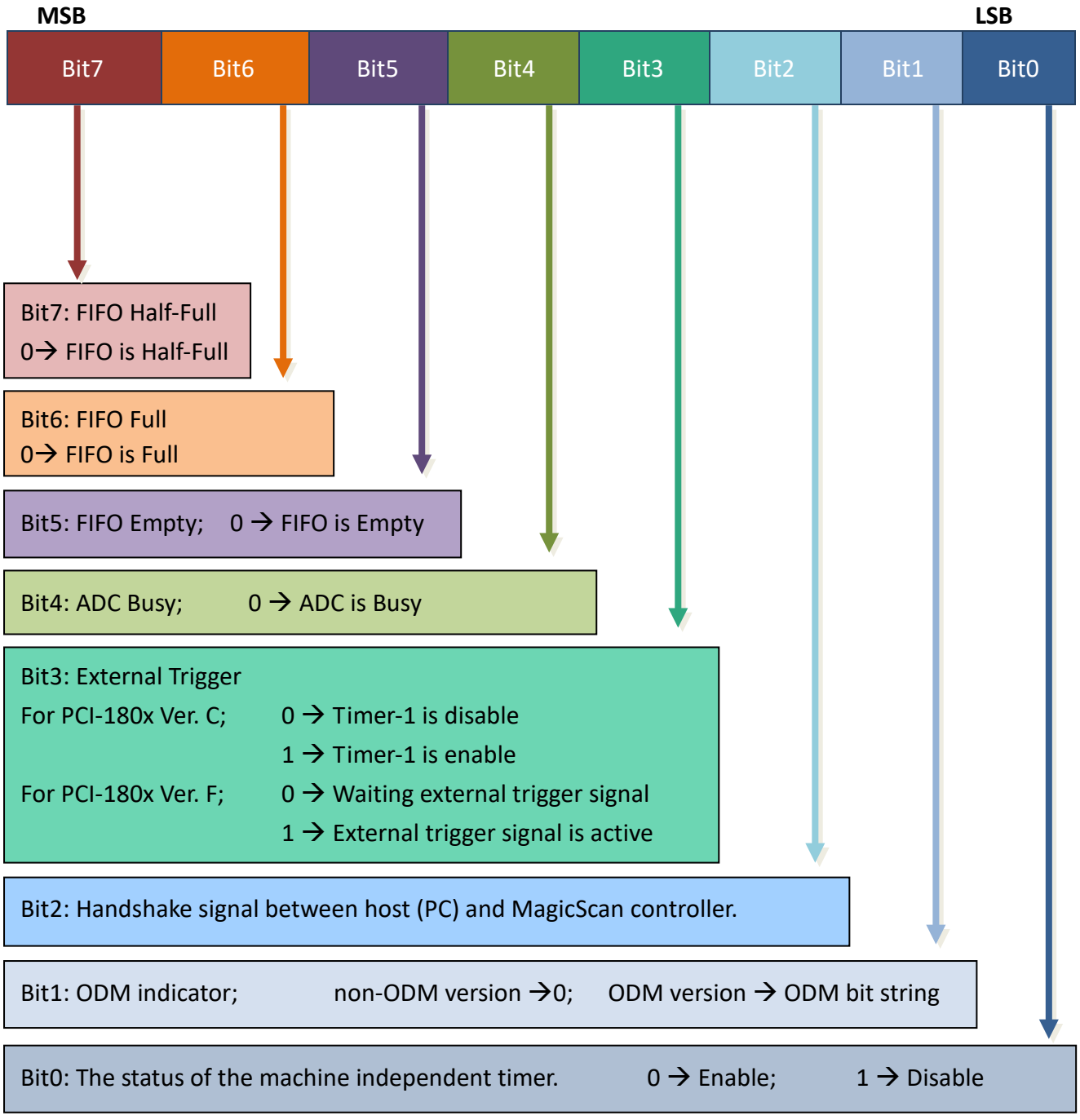
The demo program of handshaking is given as follows:

```
WORD pic_control(WORD i)
{
WORD j;

if    ((inport(wAddrCtrl)&0x04)==0)
      {
      outport(wAddrCtrl,0xffff);                    /* send a recovery to PIC */
      }
j=0;
while ((inport(wAddrCtrl)&0x04)==0)
        {
        j++;
        if    (j>65530) return(AdControllerError);    /* time out */
        }
i = i & 0xDFFF;                                     /* set pic low !! */
outport(wAddrCtrl,i);
j=0;
while ((inport(wAddrCtrl)&0x04)!=0)
        {
        j++;
        if (j>65530) return(AdControllerError);      /* time out */
        }
outport(wAddrCtrl,(WORD)(i | 0x2000));          /* set pic high !! */
j=0;
while ((inport(wAddrCtrl)&0x04)==0)
        {
        j++;
        if (j>65530) return(AdControllerError);    //time out
        }
return(NoError);
}
```

6.5.2 The Status Register

The format of the status register is given as follows:



6.5.3 The A/D Software Trigger Register

Writing to this port will perform a software trigger signal to trigger an A/D conversion. Although the PC can send very fast trigger signal (more than 333 k), the max. sampling rate of A/D conversion cannot over 330 k Samples/Sec.. The timing diagram is given as follows:

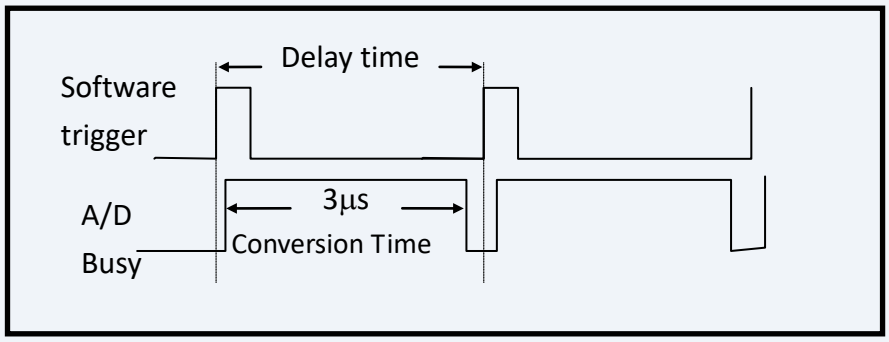


Figure 6.5.3-1: Trigger delay time.

The sample code of software trigger A/D conversion is given as follows:

P180x_AdPollingHex(…) is designed for PCI-1800/1802 series

P1202_AdPollingHex(…) is designed for PEX-1202/PCI-1202 series

P1602_AdPollingHex(…) is designed for PCI-1602 series

```
WORD    P180X_AdPollingHex(Word *AdVal)
{
WORD    wVal, wTime;
//Clear FIFO
outport(wAddrCtrl,0x2000);          //B15=0=clear FIFO, B13=1=not MagicScan controller cmd
outport(wAddrCtrl,0xA000);          //B15=1=no clear FIFO, B13=1= not MagicScan controller cmd
outport((WORD)(wAddrCtrl+4),0xffff);      /* generate a software trigger pulse */
wTime=0;
for (;;)
    {
    wVal=inport(wAddrCtrl)&0x20;          // wait for ready signal
    if (wVal!=0) break;                   /* If B4==1 → A/D data ready */
    wTime++;
    if (wTime>32760) return(AdPollingTimeOut);
    }
AdVal=inport(wAddrAdda)&0x0fff;      /* Read the available A/D data from FIFO */
return(NoError);                     /* 0xffff for PCI-1602/1602F    */
}
```

6.6 BAR3: DI/DO Register

6.6.1 Digital Output/Digital Input

The I/O address of DIO is given as follows:

I/O address of D/I	→	wAddrDio
I/O address of D/O	→	wAddrDio

The PEX-1202 and PCI-1202/1602/1800/1802 series card provides 16-channel digital input and 16-channel digital output. All levels of DI/DO are TTL compatible. The connections diagram and block diagram are given below:

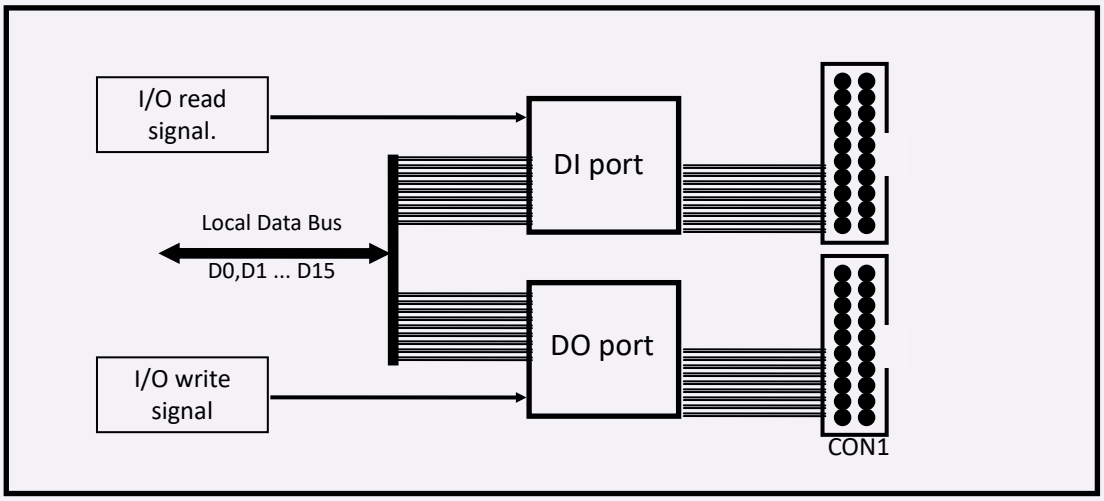


Figure 6.6.1-1: DIO Block Diagram

The D/I port can be connected to the DB-16P. The DB-16P is a 16-channel isolated digital input daughter board. The D/O port can be connected to the DB-16R or DB-24PR. The DB-16R is a 16-channel relay output board. The DB-24R is a 24-channel power relay output board.

- **The sample code of DI/DO is given as follows:**

P180x_Di (…) and P180x_Do (…) is designed for PCI-1800/1802 series
P1202_Di (…) and P1202_Do(…) is designed for PEX-1202/PCI-1202 series
P1602_Di (…) and P1602_Do (…) is designed for PCI-1600 series

```
WORD    P180X_Di(WORD *wDi)
{
*wDi=inport(wAddrDio)&0xffff;
return(NoError);
}

WORD    P180X_Do(WORD wDo)
{
outport(wAddrDio,wDo);
return(NoError);
}
```

6.6.2 Card ID Register

(Read): wAddrDIO+0x4h

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

It can be used to read the card ID set from SW1 switch. Refer to Sec. 2.3 for more information.

- **The sample code of reading the is given as follows:**

P180x_ID (…) is designed for PCI-1800/1802LU/HU
P1202_ID (…) is designed for PEX-1202L/H/PCI-1202LU/HU
P1602_ID (…) is designed for PCI-1600U/FU

```
WORD    P180X_ID(WORD *wDi)
{
*wID=inport(wAddrDio+ 0x04)&0x000f;
return(NoError);
 }
```

6.7 BAR4: A/D and D/A Register

The I/O address of A/D and D/A register is given as follows:

I/O address of DA-0	→	wAddrAdda
I/O address of DA-1	→	wAddrAdda+1*4
I/O address of FIFO	→	wAddrAdda

This BAR4 address is used to write data to the DACs and to read data from the A/D FIFO. The read/write operation is given as follows:

Port	Read	Write
BAR4 + 0	A/D FIFO.	DAC1 write.
BAR4 + 4	Reserved	DAC2 write.

The PEX-1202 and PCI-1202/1602/1800/1802 series card provides 2 independent 12-bits D/A converters with double buffer, bipolar voltage output. The output voltage can be ±5 V or ±10 V selected by J1. When the PEX-1202/PCI-1202/1602/1800/1802 series card is first power-on, the D/A will be in the floating state. The D/A will go to the programmed state after executing D/A output command. The block diagram is given as below:

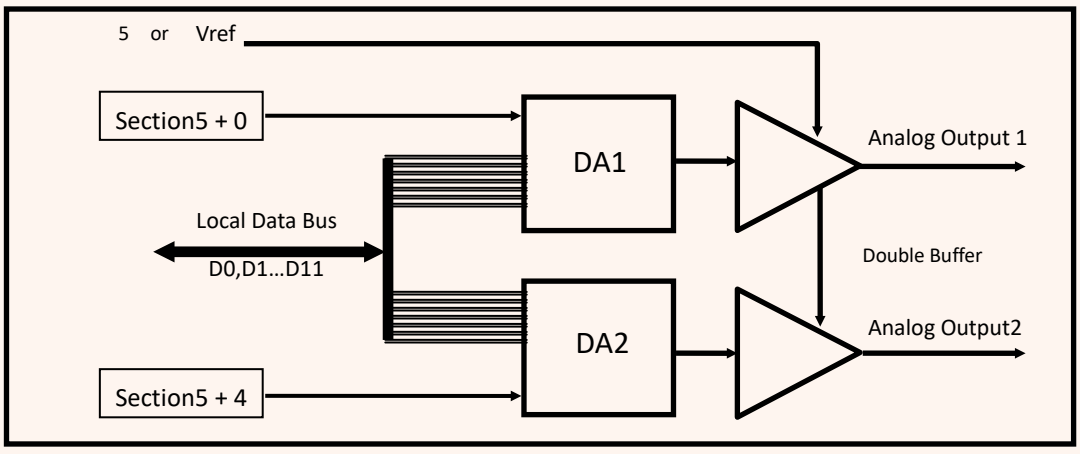


Figure 6.7-1: D/A output diagram.

Note: The D/A output is floating after first power-on. The D/A output will be enabled after executing D/A output command. This is the common feature of PEX-1202 and PCI-1202/1602/1800/1802 series.

- **The sample code for D/A is given as follows:**

P180x_Da (…) is designed for PCI-1800/1802 series
P1202_Da (…) is designed for PEX-1202/PCI-1202 series
P1602_Da (…) is designed for PCI-1600 series

```
WORD    P180X_Da(WORD wDaChannel, WORD wDaVal)
{
if    (wDaChannel==0)           /* channel 0 */
      {
      outport(wAddrAdda,wDaVal);
      return(NoError);
      }
      else if (wDaChannel==1)    /* channel_1 */
           {
           outport((wAddrAdda+4),wDaVal);
           return(NoError);
           }
      else return(DaChannelError);
}
```

- **The sample code of software trigger A/D conversion is given as follows:**

P180x_AdPollingHex (…) is designed for PCI-1800/1802 series

P1202_AdPollingHex (…) is designed for PEX-1202/PCI-1202 series

P1602_AdPollingHex (…) is designed for PCI-1600 series

```
WORD    P180X_AdPollingHex(Word *AdVal)
{
WORD    wVal, wTime ;

//Clear FIFO
outport(wAddrCtrl,0x2000);    // B15=0=clear FIFO, B13=1=not MagicScan controller cmd
outport(wAddrCtrl,0xA000);    // B15=1=no clear FIFO, B13=1= not MagicScan controller cmd
outport((WORD)(wAddrCtrl+4),0xffff);      /* generate a software trigger pulse */

wTime=0;
for (;;)
     {
     wVal=inport(wAddrCtrl)&0x20;          // wait for ready signal
     if (wVal!=0) break;                    /* if B4==1 → A/D data ready */
     wTime++;
     if (wTime>32760) return(AdPollingTimeOut);
     }
AdVal=inport(wAddrAdda)&0x0fff;          /* Read the available A/D data from FIFO */
return(NoError);                          /* 0x0fff for 12-bit ADC, 0xffff for 16-bit ADC */
}
```

7. A/D Conversion Operation

7.1 The Configuration Code Table

- **PEX-1202(L) and PCI-1202/1800/1802(L/LU)** Configuration Code Table

Bipolar Unipolar	Input Range	Gain	Settling Time	Configuration Code
Bipolar	+/- 5 V	1	3 µs	0x00
Bipolar	+/- 2.5 V	2	3 µs	0x01
Bipolar	+/- 1.25 V	4	3 µs	0x02
Bipolar	+/- 0.625 V	8	3 µs	0x03
Bipolar	+/- 10 V	0.5	3 µs	0x04
Bipolar	+/- 5 V	1	3 µs	0x05
Bipolar	+/- 2.5 V	2	3 µs	0x06
Bipolar	+/- 1.25 V	4	3 µs	0x07
Unipolar	0 V ~ 10 V	1	3 µs	0x08
Unipolar	0 V ~ 5 V	2	3 µs	0x09
Unipolar	0 V ~ 2.5 V	4	3 µs	0x0A
Unipolar	0 V ~ 1.25 V	8	3 µs	0x0B

- **PEX-1202(H) and PCI-1202/1800/1802(H/HU)** Configuration Code Table

Bipolar Unipolar	Input Range	Gain	Settling Time	Configuration Code
Bipolar	+/- 5 V	1	23 µs	0x10
Bipolar	+/- 0.5 V	10	28 µs	0x11
Bipolar	+/- 0.05 V	100	140 µs	0x12
Bipolar	+/- 0.005 V	1000	1300 µs	0x13
Bipolar	+/- 10 V	0.5	23 µs	0x14
Bipolar	+/- 1 V	5	28 µs	0x15
Bipolar	+/- 0.1 V	50	140 µs	0x16
Bipolar	+/- 0.01 V	500	1300 µs	0x17
Unipolar	0 V ~ 10 V	1	23 µs	0x18
Unipolar	0 V ~ 1 V	10	28 µs	0x19
Unipolar	0 V ~ 0.1 V	100	140 µs	0x1A
Unipolar	0 V ~ 0.01 V	1000	1300 µs	0x1B

- **PCI-1602/1602U** Configuration Code Table

Bipolar Uniploar	Input Range	Gain	Settling Time	Configuration Code
Bipolar	+/-10 V	1	10 µs	0x0
Bipolar	+/-5 V	2	10 µs	0x1
Bipolar	+/-2.5 V	4	10 µs	0x2
Bipolar	+/-1.25 V	8	10 µs	0x3

- **PCI-1602F/1602FU** Configuration Code Table

Bipolar Uniploar	Input Range	Gain	Settling Time	Configuration Code
Bipolar	+/-10 V	1	5 µs	0x0
Bipolar	+/-5 V	2	5 µs	0x1
Bipolar	+/-2.5 V	4	5 µs	0x2
Bipolar	+/-1.25 V	8	5 µs	0x3

7.2 The Unipolar/Bipolar

If the analog input signal is unipolar, you can measure this signal with bipolar setting (**this will reduce resolution**). If the analog input is bipolar, you must select bipolar configuration code to measure this signal.

7.3 The Input Signal Range

If the range of analog signal source is ±1 V, you can measure this signal wih ±10 V, ±5 V, v2.5 V and ±1.25 V configuration code setting. The only difference is the resolution. The resolution of ±2.5 V is 4 times higher than in ±10 V setting. **Select the correct configuration code will get the best resolution**.

7.4 The Settling Time

If the **channel number** or **gain factor** is changed, the hardware need **extra time for signal ready**. This is called the settling time. This limitation will apply both to the **Fixed-channel mode** and **MagicScan mode** of AD conversions. So the user must take care to avoid the settling error. In the MagicScan mode, the MagicScan controller will control all details. The MagicScan controller will change the channel number and gain control just after every pacer trigger signal. **Therefore the limitation is "settling time <= pacer timer" in MagicScan mode.**

7.5 When to Delay the Settling Time

In the software trigger mode, the software operation is given as follows:

1. send software trigger pulse
2. delay for the settling time
3. read the A/D data

The **P180X_DelayUs(...)** is a machine independent timer function. Therefore this function is suitable to delay the settling time. In the pacer trigger mode, the software does not have to call P180X_DelayUs(...) The only limitation is that the pacer timer must be longer than the settling time. Refer to Sec. 7.1 for settling time details.

7.6 The AD Conversion Mode

The AD conversion operation of PEX-1202 and PCI-1202/1602/1800/1802 series card can be divided into two different mode: **Fixed-channel mode** and the **MagicScan mode**.

P180x_… is designed for PCI-1800/1802 series
P1202_… is designed for PEX-1202/PCI-1202 series
P1602_… is designed for PCI-1600 series

The functions of **fixed-channel mode** are given as follows:	
1. P180x_SetChannelConfig	The reading data is in floating format
2. P180x_AdPolling	
3. P180x_AdsPolling	
4. P180x_AdsPacer	

The functions of **MagicScan mode** are given as follows:	
1. P180x_ClearScan	Data in 12 bits Hex format
2. P180x_StartScan	
3. P180x_ReadScanStatus	
4. P180x_AddToScan	
5. P180x_SaveScan	
6. P180x_WaitMagicScanFinish	
7. P180x_StartScanPostTrg	For external trigger
8. P180x_StartScanPreTrg	
9. P180x_StartScanMiddleTrg	

The functions of **M_functions** are given as follows:
1. P180x_M_FUN_1
2. P180x_M_FUN_2
3. P180x_M_FUN_3
4. P180x_M_FUN_4

The functions of continuous capture with storing data to main memory are given as follows: (**two boards operating simultaneously**)
1. P180x_FunA_Start
2. P180x_FunA_ReadStatus
3. P180x_FunA_Stop
4. P180x_FunA_Get

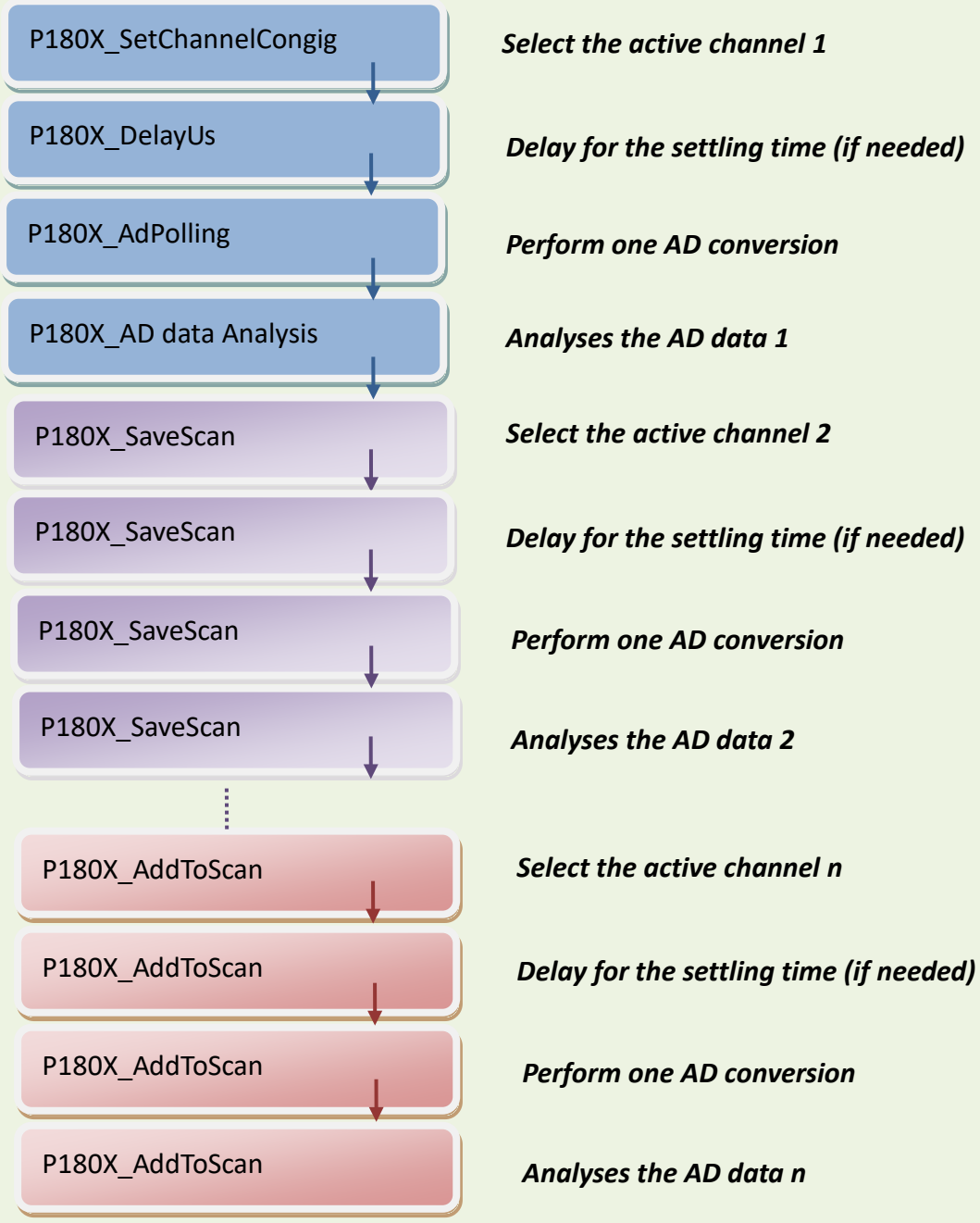
The functions of continuous capture with storing data to main memory are given as follows: (**single board operating**)
1. P180x_FunB_Start
2. P180x_FunB_ReadStatus
3. P180x_FunB_Stop
4. P180x_FunB_Get

The functions of **continuous capture** are given as follows:	
1. P180x_Card0_StartScan	Group-0: for card_0 continuous capture function
2. P180x_Card0_ReadStatus	
3. P180x_Card0_StopScan	
4. P180x_Card1_StartScan	Group-1: for Card_1 continuous capture function
5. P180x_Card1_ReadStatus	
6. P180x_Card1_StopScan	

7.7 The Fixed-channel Mode AD Conversion

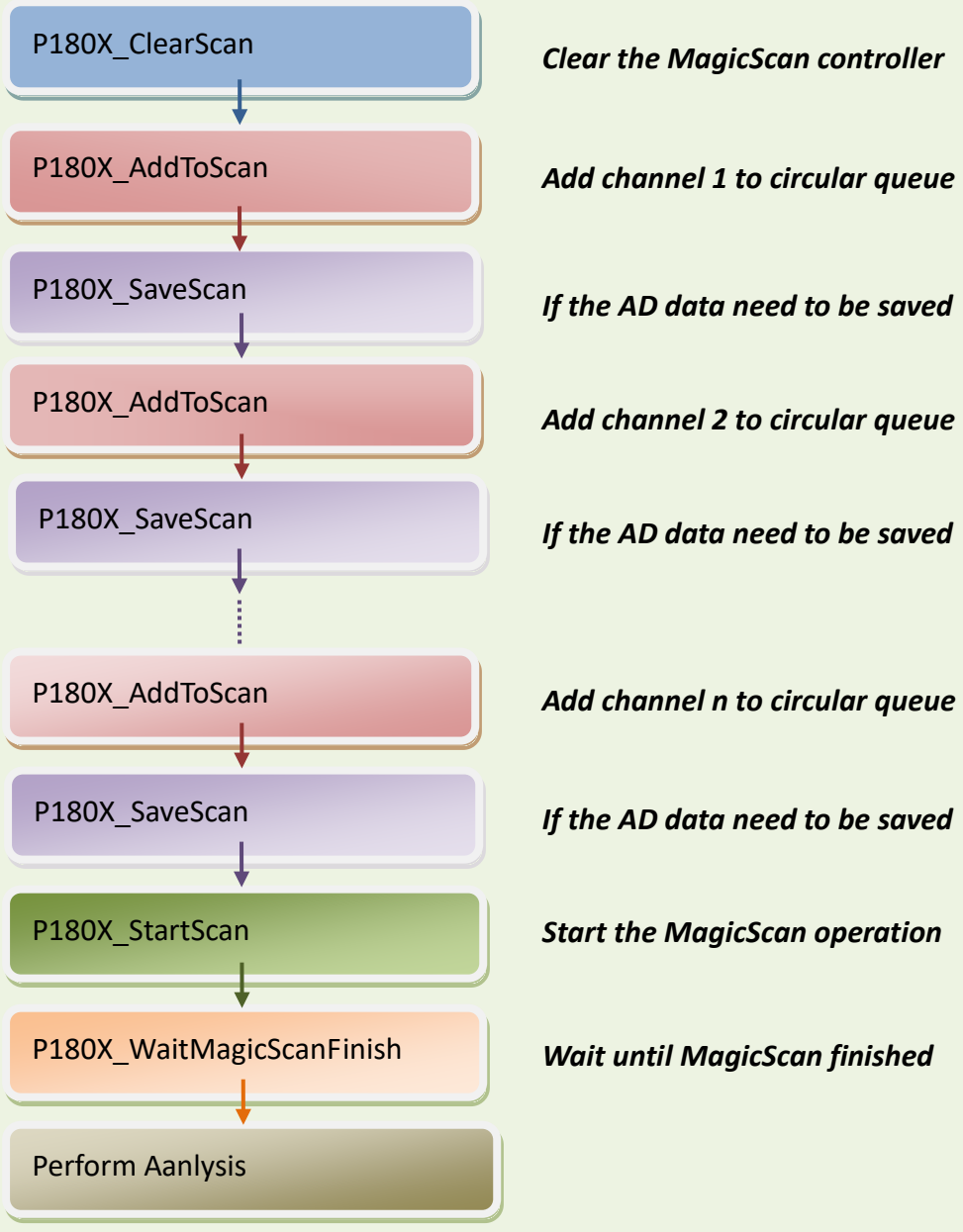
The **P180X_SetChannelConfig** activates the selected channel and sets configuration code. Then the other functions will refer to that channel and configuration. The general flow chart is given as follows:

P1202_SetChannelConfig(…) for PEX/PCI-1202 series
P1602_SetChannelConfig(…) for PCI-1602 series



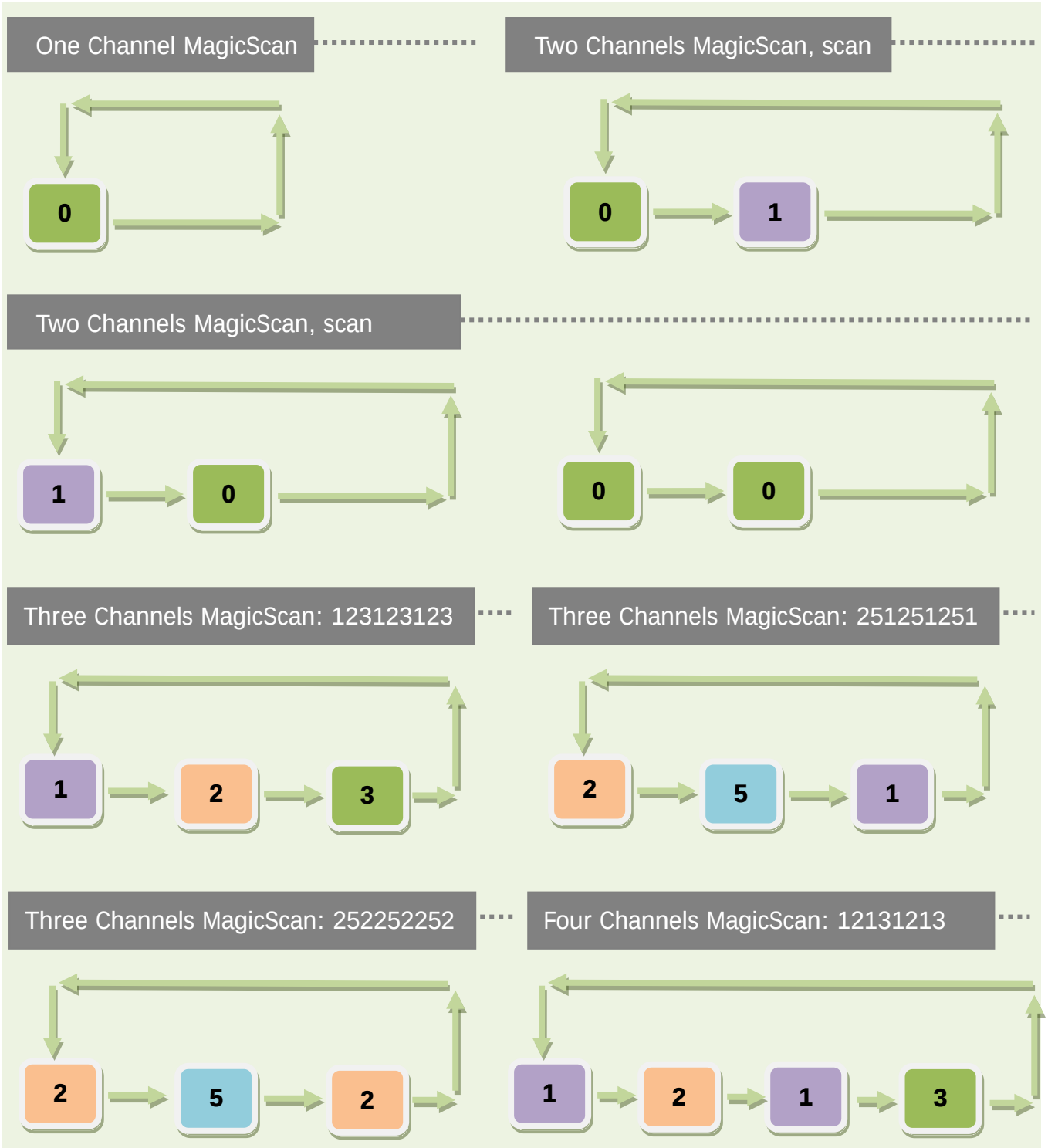
7.8 The MagicScan Mode AD Conversion

The **P180X_ClearScan** sets the MagicScan controller to its initial state. The **P180X_AddToScan** adds the channels to MagicScan circular queue one by one. **The scan sequence of the channels is depending on the order of the P180X_AddToScan settings.** The maximum queue size is **48.** The channel number in the scan list can be random and duplicated. The AD data of MagicScan can be saved in array if **P180X_SaveScan** is used. The flowchart is given as follows:



7.8.1 The MagicScan Circular_Scan_Queue

The MagicScan controller is equipped with a **circular queue** for scan sequence control. The scan sequence is **one by one** and **repeatable** with the limitation of maximum 48 channels. So the following scan sequence is all valid:



7.8.2 The Digital Filter of MagicScan

The digital filter is a **average** filter.
Filter value = (V_1+V_2+……+V_n)/n, where n is average factor
If the input signal is very noisy, this filter can be used to remove these noises.

7.8.3 The Digital Filter of MagicScan

The MagicScan controller scans the analog inputs in **fixed-sampling-rate**. The **different sampling rate** is implemented with **averaging** technique. **This technique is the same as the digital filter** described in Sec. 7.8.2. If the user wishes to use the different sampling rate between channels, the digital filter will be active at the same time. **This is a situation of ALL or NO. You can use both the digital filter and the different sampling rate at the same time or use neither of them.**

P180x_ClearScan(…) is designed for PCI-1800/1802 series

P1202_ClearScan(…) is designed for PEX-1202 and PCI-1202 series

P1602_ClearScan(…) is designed for PCI-1602 series

```
P180X_ClearScan();
P180X_AddToScan(?,?,10,….);→ only one channel scan
P180X_StartScan(?,24);        → the AD sampling rate = 8 M/24=333 k
                              → the factor=10 → sampling rate=333 k/10=33.3 k

P180X_ClearScan();
P180X_AddToScan(A,?,1,….);
P180X_AddToScan(B,?,2,….);
P180X_AddToScan(C,?,3,….);
P180X_StartScan(?,24);        → the AD sampling rate = 8 M/24=333 k
                              → scan sampling rate=333 k/3=111 k
channel_A sampling rate=111 k/1=111 k
channel_B sampling rate=111 k/2=55.5 k
channel_C sampling rate=111 k/3=37 k
```

7.8.4 The High/Low Alarm of MagicScan

There are 5 alarm types are given as follows:

Type 0 : no alarm
Type 1 : high alarm → any AD data > High_alarm_value
Type 2 : low alarm → any AD data < Low_alarm_value
Type 3 : in alarm → Low_alarm_value < any AD data < High_alarm_value
Type 4 : out alarm → any AD data < Low_alarm_value or
any AD data > High_alarm_value

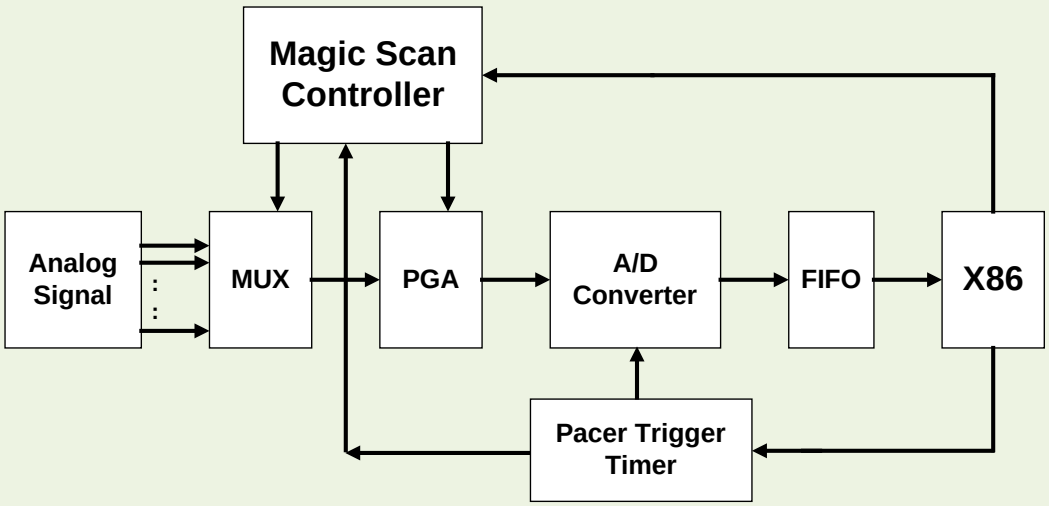
All the alarm_value are defined in HEX format.

7.8.5 The MagicScan Function

The features of MagicScan are given as follows:

1. Different gain for each channel
2. Non-sequential order for channel scan
3. Different sampling rate for each channel (use with digital filter)
4. Programmable different digital filter for each scan channel
5. Programmable HI/LO alarm for each channel
6. Three external trigger: post-trigger, pre-trigger and middle-trigger
7. Maintain at 330 k max. for total channel scan
8. Easy programming

The MagicScan function is implemented with software and hardware. The feature 1 and feature 2 are implemented in hardware. The other features are implemented in software.
The block diagram of MagicScan function is given as follows:



(1) The Magic Scan controller is a high performance RISC-like controller. It can scan the analog input signal in non-sequential order. It also control the PGA to different predefined gain for each channel.

(2) The pacer timer generates the trigger signal to A/D converter.

(3) The A/D conversion data will be placed in the FIFO.

(4) PC will read the A/D data from FIFO while the data is ready. The FIFO is 1 k samples for PCI-1800 and 8 k samples for PCI-1802. The PC can will compute and analyze the A/D data while the A/D conversion is going. Therefore the speed of PC must compatible with the speed of A/D conversion. The A/D conversion can be 330 k max.in the channel/scan mode. Therefore the PC must handle 330 k samples per second to avoid overflow. The Pentium-120 CPU or more powerful CPU is recommended.

The A/D conversion data in FIFO are in the same sampling rate (refer to (1), (2), (3)).
For example,

- the scan channel is 1→ 2 → 3
- the pacer sampling rate is 330 k
- We want a 110 k sampling rate for channel 1
- We want a 55 k sampling rate for channel 2
- We want a 11 k sampling rate for channel 3

The hardware will scan the analog data into FIFO as follows:

1,2,3,1,2,3,1,2,3,1,2,3,1,2,3,1,2,3……
→ total 330 k
→ every 1,1,1,1,1 is 110 k
→ every 2,2,2,2,2 is 110 k
→ every 3,3,3,3,3 is 110 k

The software has to fetch the 2,2,2,2,2 in 55 k, therefore the software averages the continue two 2 into one 2 to get 55 k as follows:

2,2,	2,2	2,2,	2,2	
2,	2,	2,	2,	→ 55 k

The software has to fetch the 3,3,3,3,3 in 11 k, therefore the software average the continue ten 3 into one 3 to get 11 k.

There are very heavy computation loads for the PC to execute the MagicScan function. These computation loads are given as follows:

1. Averages the continue N data into one data to get different sampling rate data
2. Compares each A/D data with the HI/LO alarm limit
3. Saves the A/D data into memory if the save flag is enabled

The MagicScan function described in this section can be realized in Pentium-120 and Windows 95 without other running programs (like anti-virus of firewall).
Refer to Sec. 7.8.6 for driver source.
Refer to Chapter 11 for demo program.
Refer to Chapter 12 for performance evaluation

7.8.6 The MagicScan Thread

The sample code of MagicScan thread as follows:

```
//---------------------------------------------------------------------
// wThreadStatus : 0x01=MagicScan start
//                 0x02=timeout1
//                 0x04=timeout2
//                 0x08=FIFO overflow
//                 0x80=MagicScan OK

WORD    magic_scan()
{
WORD     wVal,w1,w3;
DWORD    i,dwTime,j,k,dwIndex;

for (j=0; j<wMP; j++)      dwMagicSum[j]=0;
for (j=0; j<wMP; j++)      wMagicNow[j]=wMagicAve[j];
for (j=0; j<wMP; j++)      wMagicP[j]=0;
for (i=0; i<wMP; i++)      // skip the MagicScan settling time
 {
      dwTime=0;
      for    (;;)
       {
       wVal=inport(wAddrCtrl)&0x20;
       if(wVal!=0) break;
       dwTime++;
       if(dwTime>100000)
       return TimeOut;
           }
    inport(wAddrAdda)&0xffff;
  }
    dwMagicLowAlarm=0;
    dwMagicHighAlarm=0;
    for(i=0; i<wMagicNum; i++)
    {
    for (j=0; j<wMP; j++)
        {
         dwTime=0;


for (;;)
           {
              wVal=inport(wAddrCtrl)&0x60;
              if (wVal==0x20)    return FifoOverflow;
              if (wVal==0x60) break;
              dwTime++;
              if (dwTime>100000)    return TimeOut;
            }
         dwMagicSum[j]+=(inport(wAddrAdda)&0x0fff);   /* 0x0fff   for 12-bitADC, 0xffff   for 16-bit ADC */
         wMagicNow[j]--;
         w1=wMagicNow[j];
```

```
if    (w1==0)
                {
                 wVal=(WORD)(dwMagicSum[j]/wMagicAve[j]);
                 if (wMagicScanSave[j]==1)
                 {
                *((wMagicScanBuf[j])+wMagicP[j])=wVal;
                wMagicP[j]++;
                 }
                 w3=wMagicAlarmType[j];
                 if(w3>0)          // 0 = no alarm
                 {
                dwIndex=0x01; k=j;
                while (k>0)
                        {
                        dwIndex=dwIndex<<1;
                        k--;
                        }
                if    (w3==2)       // 2 = low alarm
                     {
                     if (wVal<wMagicLowAlarm[j]) dwMagicLowAlarm |= dwIndex;
                     }
                else if (w3==1)    // 1 = high alarm
                     {
                     if (wVal>wMagicHighAlarm[j]) dwMagicHighAlarm |= dwIndex;
                     }
                else if (w3==4)    // 4 = high or low alarm
                     {
   if (wVal<wMagicLowAlarm[j]) dwMagicLowAlarm |= dwIndex;
                     if (wVal>wMagicHighAlarm[j]) dwMagicHighAlarm |= dwIndex;
                     }
                else if (w3==3)    // 3 = in [low,high] alarm
                     {
                     if    ((wVal>wMagicLowAlarm[j])&& (wVal<wMagicHighAlarm[j]))
                          {
                          dwMagicLowAlarm |= dwIndex;
                          dwMagicHighAlarm |= dwIndex;
                          }
                     }
                 }
                 dwMagicSum[j]=0;
                 wMagicNow[j]=wMagicAve[j];
           }    // end if(w1
           }      // end    for(j=
     } // end for(i=
     ret_label:
     disable_timer0();
     return 0;
}
```

8. M_Function

Some real world applications have to send out the pre-defined pattern to the external device and measure the output responses for analysis. The user need one arbitrary wave form generator and one high speed A/D converter. The M_Functions, provided by PEX-1202 and PCI-1202/1602/1800/1802 series card, can send out the user defined arbitrary waveform (by software) and perform the A/D conversion (by hardware) at the same time.

The M_Functions can be executed under DOS, Windows 95/98 and 32-/64 bit Windows NT/2000/XP/Vista/2003/7/8. Some programming languages (VC++, BC++, VB, Delphi, BCB, VB.NET, C#.NET) and package (LabVIEW and more) can call the M_Functions now. The spectrum output response of the M_FUN_1 by LabView 4.0 is given as follows.

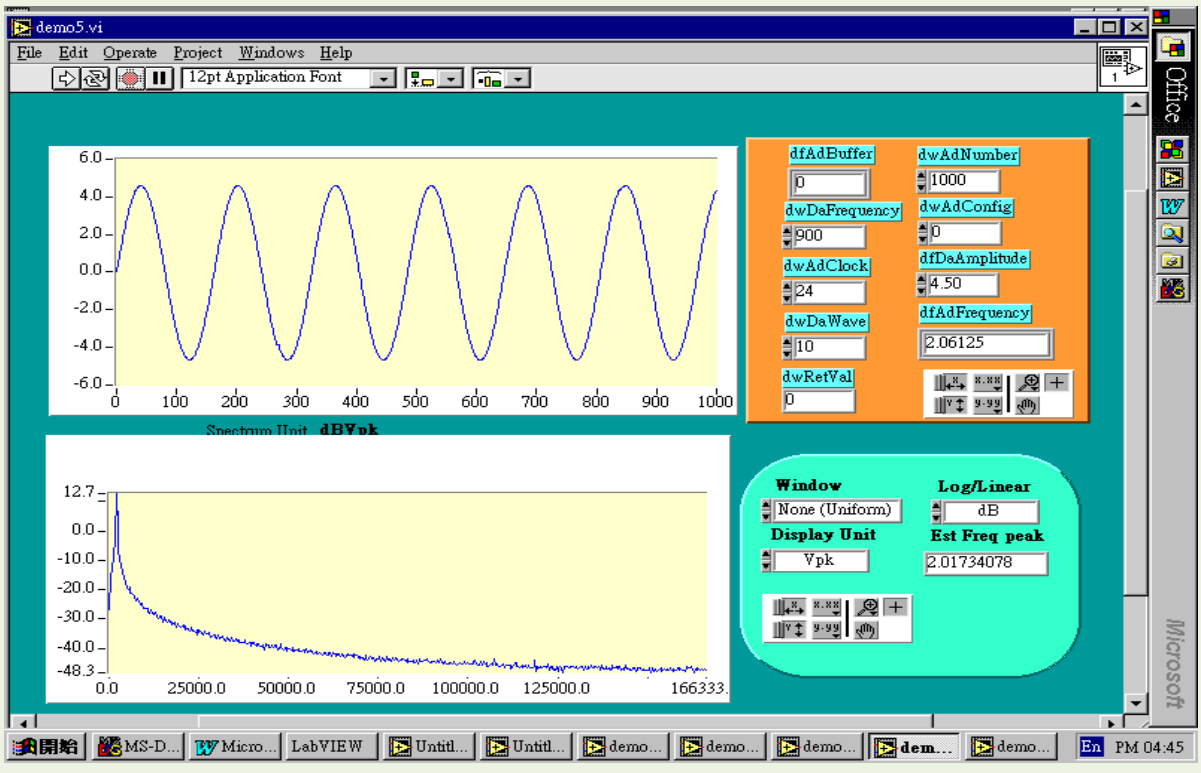


Figure 8-1: The spectrum output response of M_FUN_1.

8.1 Introduction

■ What Is M_Functions?

The features of the M_Functions are given as follows:

1. Arbitrary wave form generation (by software) from D/A output port (2 channels max.)
2. Perform MagicScan A/D conversion (by hardware) at the same time (32 channels max.)
3. Only one function call is needed
4. Very easy to use

The driver can send out the D/A wave form output to the external device and measure the response (32 channels max.) at the same time. The block diagram of the M_Functions is given as follows:

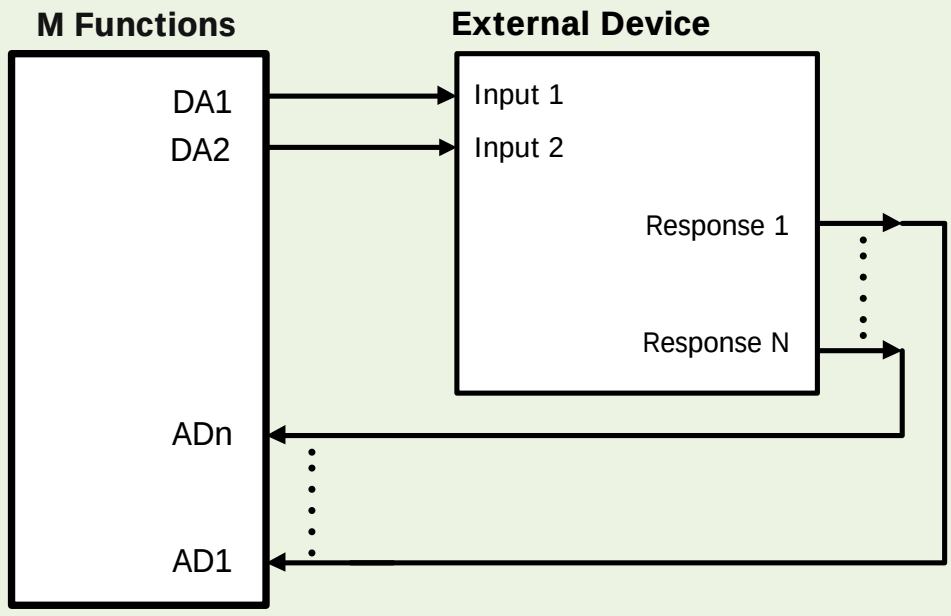


Figure 8-2: The block diagram of M-Functions.

■ Which types of waveform can be generated by the M_Functions ?

The M_Functions use **wave-form-image-data** format to reconstruct the output waveform. Therefore nearly any types of waveform can be generated. The only limitations are resolution and frequency. It is very difficult to generate a very high resolution and high frequency waveform in a multi-task OS.

If the user want to generate the periodic wave form such as sine, cosine,...., the M_Functions can provide the output wave form over 100 k Samples/sec. The ± 5 V 100 kS/s sine wave shown in Figure 8-3 and ± 5 V 200 kS/s sine wave shown in Figure 8-4 are all generated by M_Function1. The Figure 8-3 and Figure 8-4 is measured by Tektronix TDS 220. The display resolution of TDS 220 is limited, so the output waveform does not look smooth. The real output waveform is smooth.

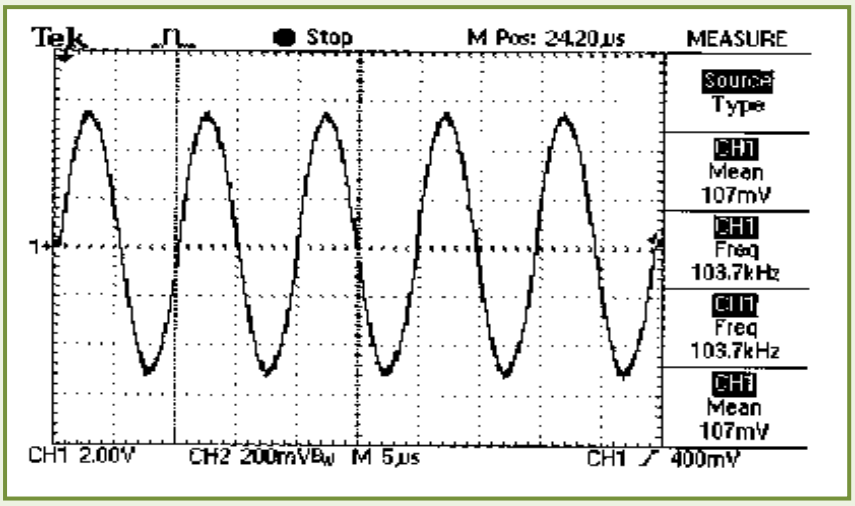


Figure 8-3:
The M_Function_1 send out a 100 k, ±5 V sine wave.
(measured by Tektronix TDS 220)

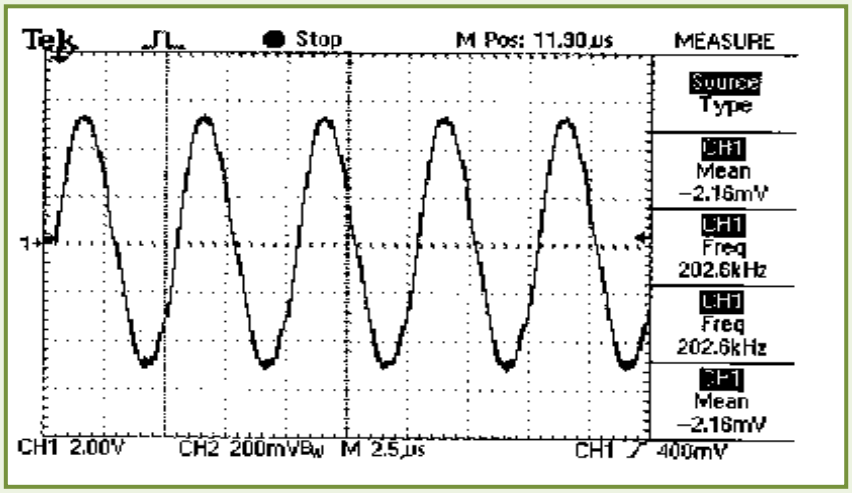


Figure 8-4:
The M_Function_1 send out a 200 k, ±5 V sine wave.
(measured by Tektronix TDS 220)

■ How many M_Functions are ready now ?

There are four M_Functions, P180X_M_FUN_1, P180X_M_FUN_2, P180X_M_FUN_3 and M_FUN_4 are ready now. The M_FUN_1 will automatic to compute the sine wave output image. The M_FUN_2 is designed for arbitrary waveform generation, so the user can prepare their waveform for M_FUN_2. The M_FUN_3 is similar to M_FUN_1 except the A/D input channels are programmable. The comparison table is given as follows:

Driver Name	D/A	A/D
P180X_M_FUN_1	Channel_0, sine wave	channel_0, ±10V
P180X_M_FUN_2	Channel_0, arbitrary wave form	channel_0, ±10V
P180X_M_FUN_3	Channel_0, sine wave	channel/gain programmable (32 channels max.)
P180X_M_FUN_4	Channel_0, square wave or semi-square-wave or sine wave	channel/gain programmable (32 channels max.)

■ Which cards support the M_Functions ?

The PEX-1202 and PCI-1202/1602/1800/1802 series can support M_Functions now.

■ Which operating systems support the M_Functions ?

The M_Functions can be executed under DOS, Windows 95/98 and Windows NT/2000/XP/2003/Vista/2008/7/8 now.

■ Limitation

The system will interrupt the driver software under multi-task OS, like Windows. The partial function of D/A arbitrary waveform generation is implemented by software. Therefore the D/A output waveform will be distorted sometimes. Refer to Figure 8-5 for details.

If the user has to generate the periodic wave form such as sine, cosine …, and the analysis is similar to spectrum analysis, this type of output distortion will cause little trouble. **The D/A output maybe distorted but spectrum response is still stable.**

If the user uses DOS, the D/A output waveform will not be distorted in any time.

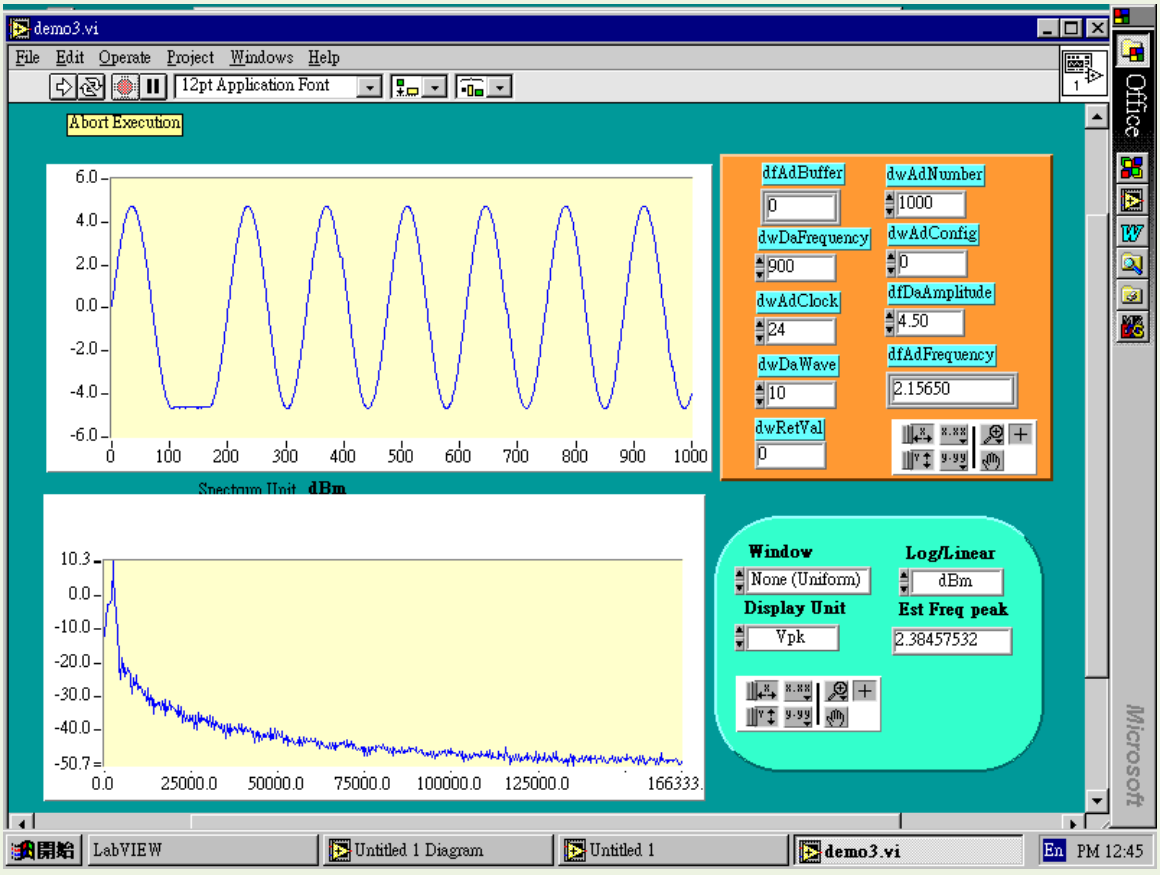


Figure 8-5: The D/A waveform is distorted but the spectrum response is nearly the same.

9. Continuous Capture Function

The continuous capture functions are very useful in real world applications. It can be used in many types of applications. Those applications are

1. Low speed, no storage, real-time processing, continuous capture
2. High speed, store the A/D data in PC main memory, time is limited by memory size (Referring to P180X_FunA series functions and P180X_FunB series function for more Detail information in 9.2)
3. High speed, store the A/D data in the external NVRAM, time is limited by memory size

9.1 General Purpose Functions

The PEX-1202 and PCI-1202/1602/1800/1802 series card is very suitable for these three applications. The software driver can support 16 cards max. in one PC system. The software also supports 2 cards for continuous capture function. The continuous capture functions are specially designed into many groups. Each group is corresponding to one card. There are three functions included in a group as follows:

1. P180X_Card0_StartScan(…)
2. P180X_Card0_ReadStatus(…)
3. P180X_Card0_StopScan(….)

Group-0: for card_0 continuous capture function

1. P180X_Card1_StartScan(…)
2. P180X_Card1_ReadStatus(…)
3. P180X_Card1_StopScan(….)

Group-1: for card_1 continuous capture function

The features of these functions are given as follows:

- Support DOS, Window 98/NT and Windows NT/2000/XP/2003/Vista/2008/7/8
- Single-card solution → group0, refer to DEMO13.C
- Multiple-card solution → group0 & group1 RUN at the same time, refer to DEMO14.C.
- **P1202_Card0_StartScan(…) is designed for PEX/PCI-1202 series**
- **P1602_Card0_StartScan(…) is designed for PCI-1602 series**

The block diagram of **continuous capture function** is given as follows:

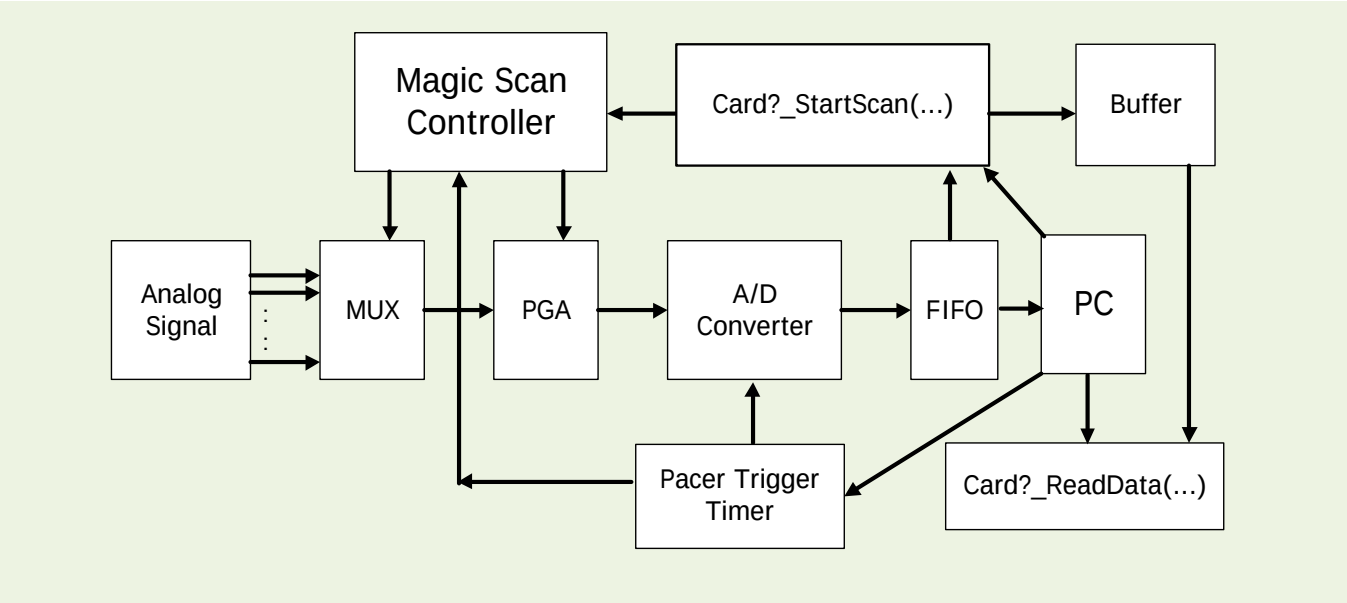


Figure 9-1: The block diagram of continuous capture.

- The **P180X_Card?_StartScan(…)** will perform the following function:
  1. setup scan-queue
  2. setup channel/gain data
  3. setup continuous capture data
  4. create a multi-task thread for long time data acquisition
  5. If the group A/D data are ready → signal P180X_Card?_ReadStatus(…) to read data

- The **P180X_Card?_ReadStatus(…)** will read from the buffer prepared by P180X_Card?_StartScan(…). This function is running at the same time with the P180X_Card?_StartScan(…) thread.

- The **P180X_Card?_StopScan(…)** will stop all threads and return all resource

- The sample code for **single board** is given as follows:

```
wRetVal=P180X_Card0_StartScan(……);      // setup continuous capture function
                                        // this function will create thread
if (wRet != NoError)
        {
        Show error message & return
        }
// now the thread is active and the continuous capture function is going now
for(;;)
        {
        wRetVal=P180X_Card0_ReadStatus(…);
        if (wRetVal != 0)
                {
                show these A/D data or
                save these A/D data or
                analyze these A/D data
                }
        if (stop flag is ON)      // for example, the user press STOP key here
                {
                Card0_StopScan(…);
                return OK
                }
        }
```

- The sample code for **multi-boards** is given as follows:

```
wRetVal=P180X_Card0_StartScan(……);          // setup continuous capture function
                                            // this function will create thread
if (wRet != NoError) { Show error message & return }
wRetVal=P180X_Card1_StartScan(……);          // setup continuous capture function
                                            // this function will create thread
if (wRet != NoError) { Show error message & return }
wRetVal=P180X_Card?_SartScan(……);           // setup continuous capture function
                                            // this function will create thread
if (wRet != NoError) { Show error message & return }
// now the thread is active and the continuous capture function is going now

for(;;)
        {
        wRetVal=P180X_Card0_ReadStatus(….);
        if(wRetVal != 0)
                {
                show these A/D data or
                save these A/D data or
                analyze these A/D data
                }
        wRetVal=P180X_Card1_ReadStatus(….);

if (wRetVal != 0)
                {
                show these A/D data or
                save these A/D data or
                analyze these A/D data
                }
wRetVal=P180X_Card?_ReadStatus(….);
        if (wRetVal != 0)
                {
                show these A/D data or
                save these A/D data or
                analyze these A/D data
                }

        if (stop flag is ON)      // for example, the user press STOP key here
                {
                Card0_StopScan(…);
                return OK
                }
}
```

Refer to DEMO13.C & DEMO14.C for details.

9.2 Functions for saving Data in PC Memory

The P180X_FunA and P180X_FunB series functions are designed for continuous capture which storing the data into main memory. The features for these P180X_FunA and P180X_FunB are listed as follows:

- Sampling A/D data with high speed (for example, 330K)
- Continues capture for a long period (for example, 2.5 minutes continue)
- A/D data save in the PC memory first, then analyze these data later (memory size=330 k*60*2.5=330 k*150=49.5 M word=99 M bytes)
- Refer to demo22.c for **330 k, 2.5 minutes,** continuous capture using **99 M** bytes of the PC memory

The P180X_FunA is designed for two boards and the P180X_FunB (Figure 9-2) is designed for single-board as follows:

P180X_FunA_Start P180X_FunA_ReadStatus P180X_FunA_Stop P180X_FunA_Get	• Support two board • continuous capture • data save in PC memory (can be as large as 256 M) • refer to demo20.c
P180X_FunA_Start P180X_FunA_ReadStatus P180X_FunA_Stop P180X_FunA_Get	• Support single board • continuous capture • data save in PC memory (can be as large as 256 M) • refer to demo21.c

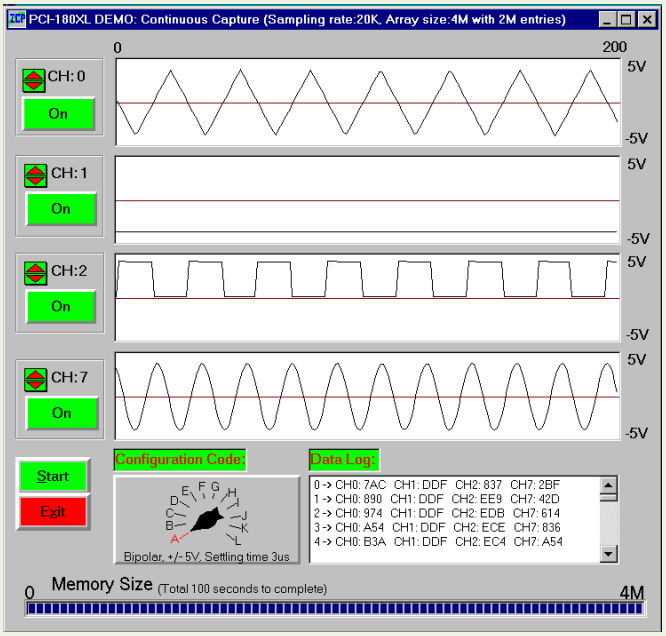


Figure 9-2:
The continuous capture example.

10. Calibration

10.1 A/D Calibration

- **For PEX-1202 and PCI-1202/1800/1802 series card:**

Step 1: Apply 0 V to channel 0
Step 2: Apply 4.996 V to channel 1
Step 3: Apply +0.6245 V to channel 2 for PEX-1202L and PCI-1202/1800/1802(L/LU)
Step 4: Apply +4.996 mV to channel 2 for PEX-1202H and PCI-1202/1800/1802(H/HU)
Step 5: Run DEMO19.EXE
Step 6: Adjust VR101 until CAL_0 = 7FF or 800
Step 7: Adjust VR100 until CAL_1 = FFE or FFF
Step 8: Repeat Step6 & Step7 until all OK
Step 9: Adjust VR1 until CAL_2 = FFE or FFF
Step 10: Adjust VR2 until CAL_3 = 000 or 001

- **For PCI-1602 series card:**

Step 1: Apply 0 V to channel 0
Step 2: Apply 4.996 V to channel 1
Step 3: Apply +0.6245 V to channel 2
Step 4: Run DEMO19.EXE
Step 5: Adjust VR3 until channel 0 = 0000 or FFFF
Step 6: Adjust VR2 until channel 1 = 7FFF or 7FFE
Step 7: Repeat Step5 & Step6 until all OK
Step 8: Adjust VR1 until channel 2 = 0FFC or 0FFD

- The wiring diagram of A/D calibration:

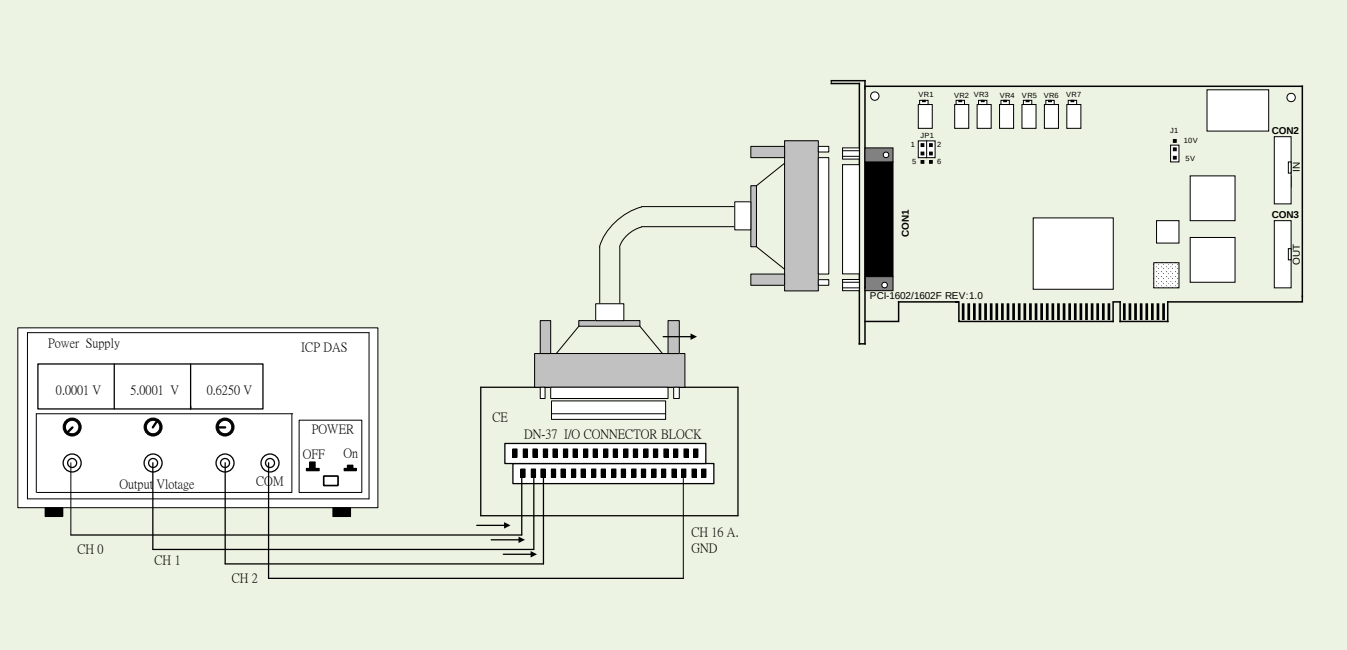


Figure 10-1. AD Calibration

Note:

1. ***The CH 16 is the GND of analog signal for PEX-1202 and PCI-1202/1602/1802 series card.***
2. ***The CH 9/10 are the GND of analog signal for PCI-1800 series card.***

10.2 D/A Calibration

- **For PCI-1800/1802 version_F and PEX-1202/PCI-1202 series card:**

Step 1: J1 select +10 V
Step 2: Connect the D/A channel 0 to voltage meter
Step 3: Send 0x800 to D/A channel 0
Step 4: Adjust VR200 until voltage meter = 0 V
Step 5: Send 0 to D/A channel 0
Step 6: Adjust VR201 until voltage meter = -10 V
Step 7: Connect the D/A channel 1 to voltage meter
Step 8: Send 0x800 to D/A channel 1
Step 9: Adjust VR202 until voltage meter = 0 V
Step 10: Send 0 to D/A channel 1
Step 11 : Adjust VR203 until voltage meter = -10 V

- **For PCI-1800/1802 version_C series card:**

Step 1: J1 select +10 V
Step 2: Connect the D/A channel 0 to voltage meter
Step 3: Send 0 to D/A channel 0
Step 4: Adjust VR3 until voltage meter = -10 V

- **For PCI-1602 series card:**

Step 1: J1 select +10 V
Step 2: Connect the D/A channel 0 to voltage meter
Step 3: Send 0x800 to D/A channel 0
Step 4: Adjust VR4 until voltage meter = 0 V
Step 5: Send 0 to D/A channel 0
Step 6: Adjust VR5 until voltage meter = -10 V
Step 7: Connect the D/A channel 1 to voltage meter
Step 8: Send 0x800 to D/A channel 1
Step 9: Adjust VR7 until voltage meter = 0 V
Step 10: Send 0 to D/A channel 1
Step 11: Adjust VR6 until voltage meter = -10 V

- The wiring diagram of D/A calibration:

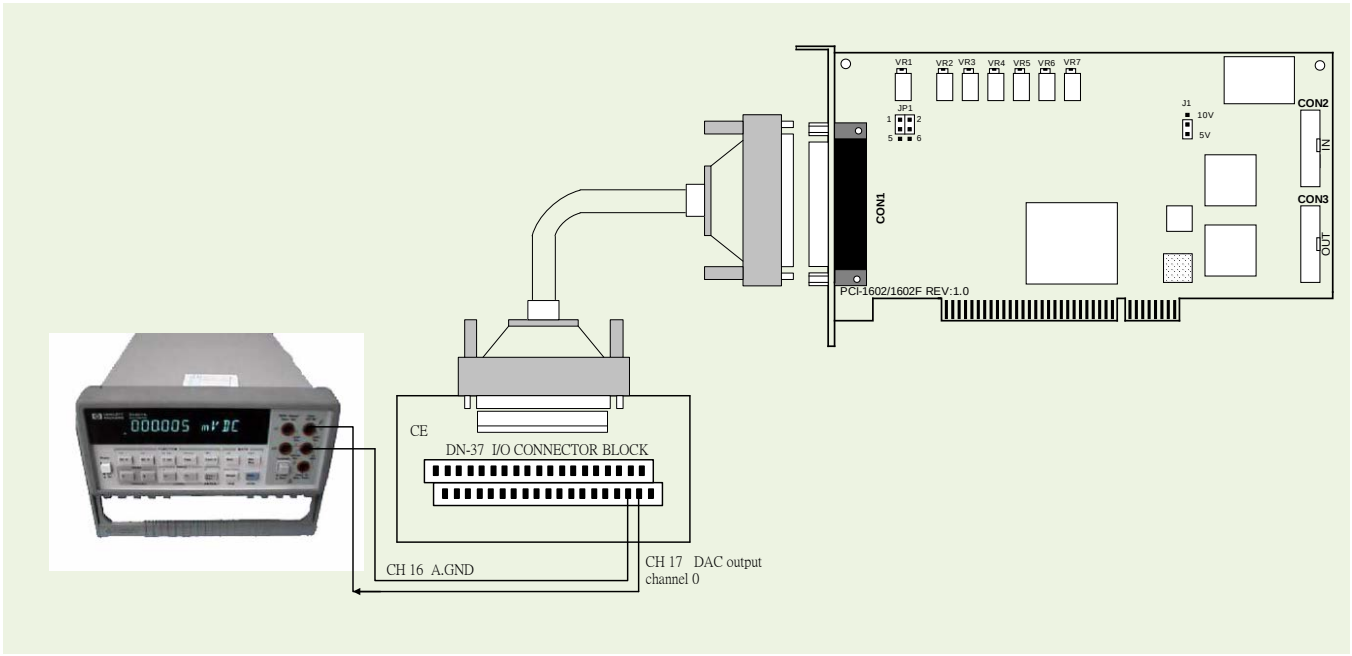


Figure 10-2. D/A Calibration

Note:

1. ***The CH 18/36 are the output channels 0/1 of DAC for PEX-1202 and PCI-1202/1602/1802 series card.***
2. ***The CH 30/32 are the output channels 0/1 of DAC for PCI-1800 series card.***

11. Demo Program

11.1 Demo Program for Windows

All demo programs will not work properly if the DLL driver has not been installed correctly. During the DLL driver installation process , the install-shields will register the correct kernel driver to the operation 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-/64-bit winXP/2003/Vista/7/8). Once driver installation is complete, the related demo programs and development library and declaration header files for different development environments will be presented as follows.

■ Demo Program for PCI-1202/1602/180x Series Classic Driver:

The demo program of PEX-1202 and PCI-1202 series is contained in:
CD:\NAPDOS\PCI\PCI-1202\DLL_OCX\Demo\
http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/pic-1202/dll_ocx/demo/

The demo program of PCI-1602 series is contained in:
CD:\NAPDOS\PCI\PCI-1602\DLL_OCX\Demo\
http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/pci-1602/dll_ocx/demo/

The demo program of PCI-180x series is contained in:
CD:\NAPDOS\PCI\PCI-180x\DLL_OCX\Demo\
http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/pci-180x/dll_ocx/demo/

There are about 20 demo program given as follows:

- demo1: one board, D/I/O test, D/A test, A/D polling test, general test
- demo2: two board, same as demo1
- demo3: one board, A/D by software trigger(polling) and A/D by pacer trigger demo

- demo4: two board, same as demo3
- demo5: one board, M_function_1 demo
- demo6: two board, same as demo5
- demo7: one board, M_function_2 demo
- demo8: two board, same as demo7
- demo9: one board, M_function_3 demo
- demo10: two board, same as demo9
- demo11: one board, MagicScan demo
- demo12: two board, same as demo11
- demo13: one board, continuous capture demo
- demo14: two board, continuous capture demo
- demo15: all installed board, D/I/O test for board number identification
- demo16: one board, performance evaluation demo
- demo17: one board, MagicScan demo, scan sequence: 1→2→0
- demo18: one board, MagicScan demo, scan 32 channel, show channel 0/1/15/16/17
- demo19: one board, A/D calibration.
- demo20: two board, P180X_FUNA, continuous capture demo
- demo21: single board, P180X_FUNB, continuous capture demo
- demo22: single board, P180X_FUNB, 330 k, 2.5 min, continuous capture 99 M bytes
- demo23: single board, post-trigger demo
- demo24: single board, pre-trigger demo
- demo25: single board, middle-trigger demo
- demo26: single board, pre-trigger demo for version-C
- demo27: single board, middle-trigger demo for version-C
- demo28: multi-task, critical section driver demo
- demo29: testing for MagicScan controller.
- demo30: testing for Pacer Trigger.
- Demo31: testing for Polling.
- Demo32: monitoring the incoming data from MagicScan, then set a digital out bit on when the incoming data exceed a defined threshold.
- Demo33: MagicScan total sample rate=176 k/sec for 8 channels.
- Demo34: continuous capture scan total sample rate=33.3 k/sec for 32 channels and save to disk (for DOS only).

For detailed information about the DLL function of the PCI-1202/1602/180x series, please refer to DLL Software Manual (CD:\NAPDOS\PCI\PCI-1202 (or PCI-1602/180x)\Manual\)

Demo Program for UniDAQ SDK Driver

The demo program is contained in:
CD:\NAPDOS\PCI\UniDAQ\DLL\Demo\
http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/unidaq/dll/demo/

There are about demo program given as follows:

- Analog Input Pacer
- Analog Input Pacer Continue
- Analog Input Pacer Scan
- Analog Input Pacer Scan Continue
- Analog Input Pacer Scan EXT
- Analog Input Polling
- Analog Output
- Analog Output Current
- Digital I/O
- Digital I/O by Card ID

For detailed information about the DLL function and demo program of the UniDAQ, please refer to UniDAQ DLL Software Manual (CD:\NAPDOS\PCI\UniDAQ\Manual\)

11.2 Diagnostic Program

11.2.1 Power-on Plug&Play Test

The operation steps of power-on plug&play test are given as follows:

Step 1: Power-off PC

Step 2: Install PCI-1202/1602/1800/1802 without any extra external connector

Step 3: Power-on PC and check the PC screen very carefully

Step 4: The PC will performance self-test first

Step 5: Detect the non-PCI physical devices installed in the system

Step 6: Show the information of these device in screen

Step 7: Detect the PCI plug&play devices installed in the system

show all PCI-device information → check here carefully

→ there will be a PCI device with vendor_ID=1234, device_ID=5678 (PCI-1800/1802)

vender_ID=1234, device_ID=5676 (PCI-1602)

vender_ID=1234, device_ID=5672 (PEX/PCI-1202)

If the plug&play ROM-BIOS can detect the PEX-1202 and PCI-1202/1602/1800/1802 series in the power-on time, the software driver of DOS, Windows 95/98//NT/2000 will function OK later. If the plug&play ROM-BIOS cannot find the PEX-1202 and PCI-1202/1602/1800/1802 series, all software driver will not function. Therefore the user must make sure that the power-on detection is correct.

11.2.2 Driver Plug&Play Test

Step 1: Power-off PC

Step 2: Install PEX-1202 and PCI-1202/1602/1800/1802 series without any extra external connector

Step 3: Power-on PC, run DEMO15.EXE

Step 4: The I/O base address of all PCI-1xxx installed in the system will be shown in screen.

Step 5: Is the total board number correct?

Step 6: Install a 20-pin flat cable in one of these PEX-1202 and PCI-1202/1602/1800/1802 series

Step 7: One card's D/O=D/I → this is the physical card number, remember this number.

Step 8: Repeat the previous two steps to find the physical card number of all boards.

11.2.3 D/O Test

Step 1: Power-off PC
Step 2: Install one PEX-1202 and PCI-1202/1602/1800/1802 card with a 20-pin flat cable between CON1 and CON2
Step 3: Power-on PC, run DEMO15.EXE
Step 4: Check the value of D/O and D/I → must be the same.

11.2.4 D/A Test

Step 1: Power-off PC
Step 2: Install one PEX-1202 and PCI-1202/1602/1800/1802 card with DA channel 0 connected to A/D channel 0.
Step 3: Power-on PC, run DEMO1.EXE
Step 4: Check the value of A_0 → = 1.25 volt.
Step 5: Run DEMO5.EXE
Step 6: Check the wave form shown in screen must be sine wave

11.2.5 A/D Test

Step 1: Power-off PC
Step 2: Install one PEX-1202 and PCI-1202/1602/1800/1802 card with DA channel 0 connected to A/D channel 0.
Step 3: Power-on PC, run DEMO1.EXE
Step 4: Check the value of A_0 → = 1.25 volt.
Step 5: Run DEMO5.EXE
Step 6: Check the waveform shown in screen must be sine wave
Step 7: Apply analog signals to all A/D channels
Step 8: Run DEMO3.EXE to check all A/D data measured

12. Performance Evaluation

Demo Program	Performance	Description
DEMO16.EXE.	1.7 MS/s	D/I performance
DEMO16.EXE.	2.1 MS/s	D/O performance
DEMO16.EXE.	2.0 MS/s	D/A performance
DEMO13.EXE	20 kS/s	Continuous capture function, one card, two channels Total=20 kS/s → 10 kS/s per channels
DEMO14.EXE	20 kS/s	Continuous capture function, two card, two channels Total=20 kS/s → 10 kS/s per channels
DEMO5.EXE	20 k sine max.	M_function demo, D/A channel_0 to A/D channel_0 20 kHz sine wave max. 20 Hz sine wave min.
DEMO11.EXE	330 k 110 k 200 k 100 k	MagicScan demo for PCI-1800/1802 MagicScan demo for PEX-1202 and PCI-1202 MagicScan demo for PCI-1602F MagicScan demo for PCI-1602

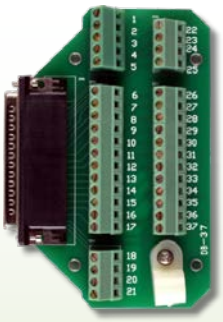
Note:

1. **S/s → Samples/Sec.**
2. **All test are under Windows 98 and Pentium-200 CPU**

Appendix: Daughter Board

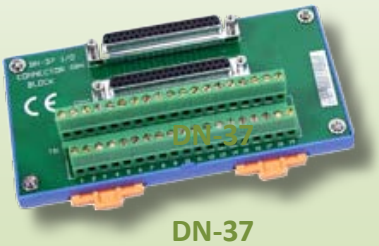
A1. DB-37 and DN-37

- **DB-37:** The DB-37 is a general purpose daughter board for D-sub 37 pins. It is designed for easy wire connection via pin-to-pin.



DB-37

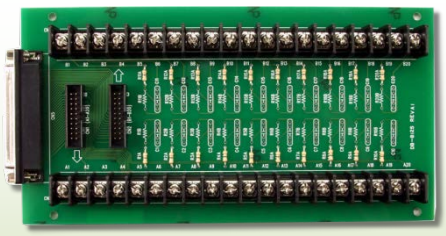
- **DN-37:** The DN-37 is a general purpose daughter board for DB-37 pins with DIN-Rail Mountings. They are also designed for easy wire connection via pin-to-pin.



DN-37

A2. DB-8125

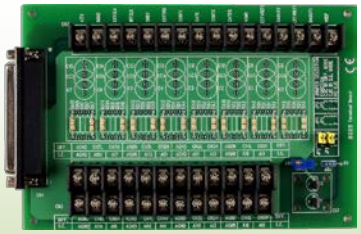
The DB-8125 is a daughter board designed for 32 channels AD cards such as ISO-AD32, PEX-1202 and PCI-12021602/1802 that can easy signal connection and measurement. The DB-8125 consists of one DB-37 and two 20-pin flat-cable headers. Refer to "DB-1825 user manual" for details.



DB-8125

A3. DB-8225

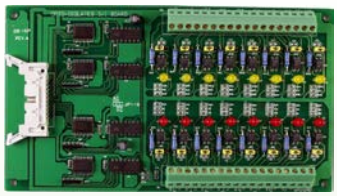
The DB-8225 provides a **on-board CJC**(Cold Junction Compensation) circuit for thermocouple measurement and **terminal block** for easy signal connection and measurement. The CJC is connected to A/D channel_0. The PCI-1800 can connect CON3 direct to DB-8225 through a 37-pin D-sub connector. Refer to "DB-8225 User Manual" for details.



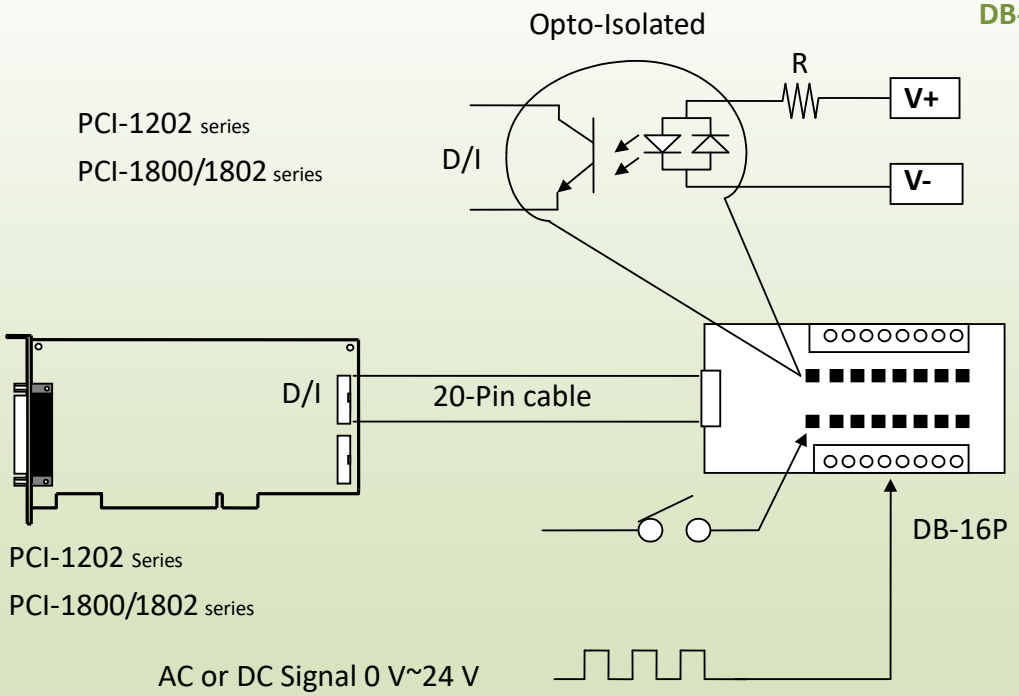
DB-8225

A4. DB-16P Isolated Input Board

The DB-16P is a 16-channel isolated digital input daughter board. The optically isolated inputs of the DB-16P are consisted of are bi-directional optocoupler with resistor for current sensing. You can use the DB-16P to sense DC signal from TTL levels up to 24 V or use the DB-16P to sense a wide range of AC signals. You can use this board to isolate the computer from large common-mode voltage, ground loops and transient voltage spike that often occur in industrial environments.

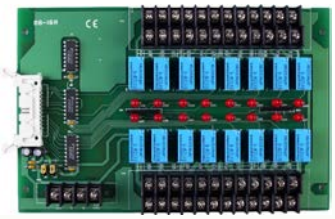


DB-16P

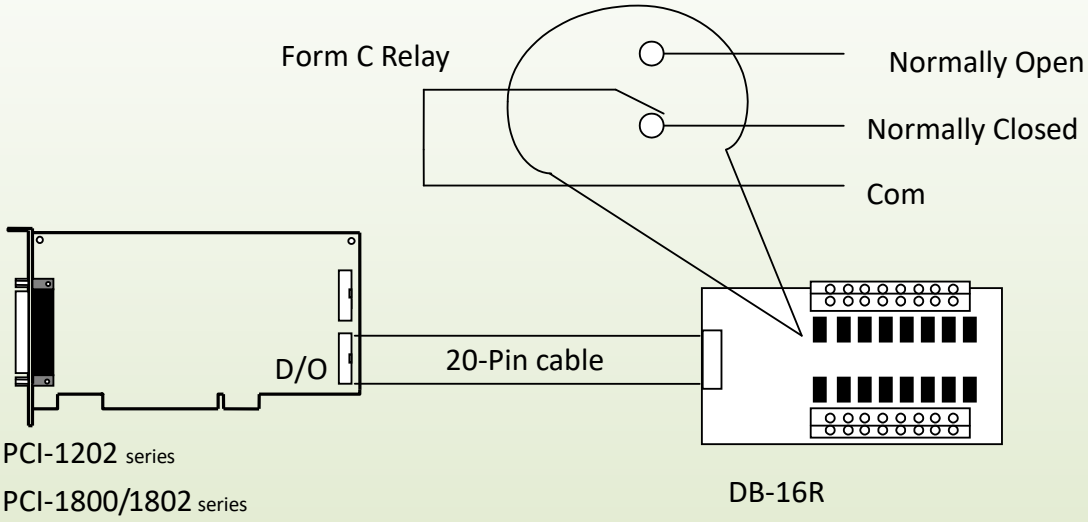


A5. DB-16R Relay Board

The DB-16R, 16-channel relay output board, consists of 16 Form C relays for efficient switching of load by programmed control. It is connector and functionally compatible with 785 series board but with industrial type terminal block. The relay is energized by applying 5 voltage signal to the appropriate relay channel on the 20-pin flat connector. There are 16 enunciator LEDs for each relay, light when their associated relay is activated. To avoid overloading your PC's power supply, this board provides a screw terminal for external power supply.



DB-16R



Note!!

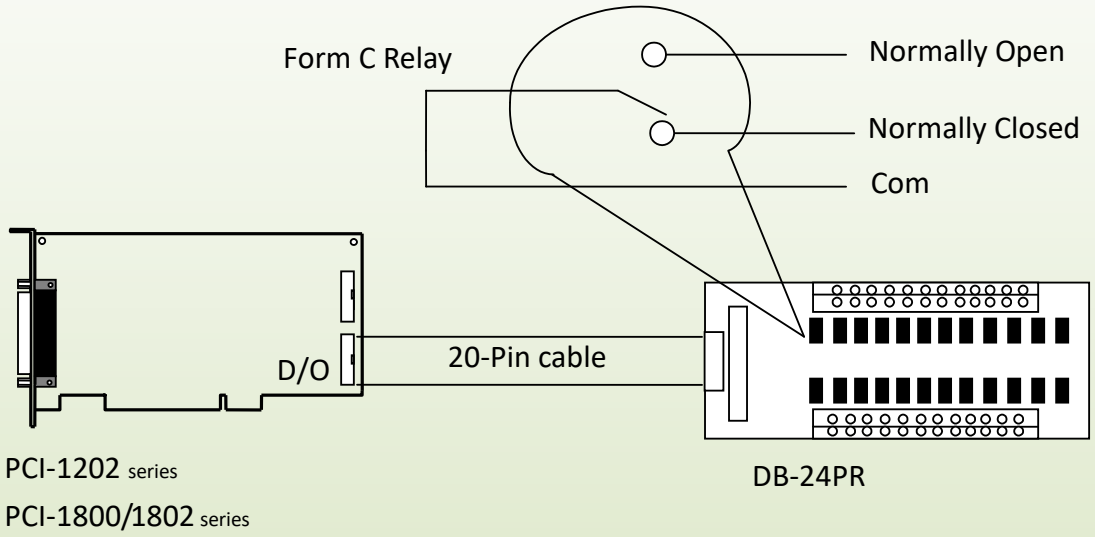
Channel: 16 Form C Relay
Relay: Switching up to 0.5 A at 110 V_{AC}/ 1 A at 24 V_{DC}

A6. DB-24PR Power Relay Board

The DB-24PR, 24-channel power relay output board, consists of 8 Form C and 16 Form A electromechanical relays for efficient switching of load by programmed control. The contact of each relay can control a 5 A load at 250 V_{AC}/30 V_{DC}. The relay is energized by applying a 5 voltage signal to the appropriate relay channel on the 20-pin flat cable connector (only 16 relays are used) or 50-pin flat cable connector. (OPTO-22 compatible, for DIO-24 series.) Twenty - four enunciator LEDs, one for each relay, 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.



DB-24PR



Note!!

50-Pin Connector (OPTO-22 compatible) for DIO-24, DIO-48, DIO-144.
20-Pin connector for 16-ch D/O board, A-82x, A-62x, DIO-64 and ISO-DA16/DA8
Channel: 16 Form A Relay and 8 Form C Relay
Relay: Switching up to 5 A at 110 V_{AC}/ 5 A at 30 V_{DC}

Artisan Technology Group is an independent supplier of quality pre-owned equipment

Gold-standard solutions

Extend the life of your critical industrial, commercial, and military systems with our superior service and support.

We buy equipment

Planning to upgrade your current equipment? Have surplus equipment taking up shelf space? We'll give it a new home.

Learn more!

Visit us at **artisantg.com** for more info on price quotes, drivers, technical specifications, manuals, and documentation.

Artisan Scientific Corporation dba Artisan Technology Group is not an affiliate, representative, or authorized distributor for any manufacturer listed herein.

We're here to make your life easier. How can we help you today?

(217) 352-9330 | sales@artisantg.com | artisantg.com

ARTISAN® TECHNOLOGY GROUP