PCI-9118 DG/HG/HR

PCI-Bus Advanced
Data Acquisition Card

Manual Rev. 2.11: November 17, 1999

Trademarks

CONTENTS

How to Use This Guide

This manual is designed to help you use the PCI-9118. The manual describes how to modify various settings on the PCI-9118 card to meet your requirements. It is divided into six chapters:

- Chapter 1, "Introduction", gives an overview of the product features, applications, and specifications.
- Chapter 2, "Installation", describes how to install the PCI-9118. The layout of PCI-9118 is shown, jumper setting for analog input channel configuration, D/A reference voltage setting are specified.
- Chapter 3, "Signal Connection", describes the connectors' pin assignment and how to connect the outside signal and devices with the PCI-9118.
- Chapter 4, "Registers Structure & Format", describes the details of register format and structure of the PCI-9118, this information is very important for the programmers who want to control the hardware by low-level programming.
- Chapter 5, "Operation Theorem", describes how to operate the PCI-9118. The A/D, D/A, DIO and timer/counter functions are introduced. Also, some programming concepts are specified.
- Chapter 6, "C/C++ Software Library", describes high-level programming interface in C/C++ language. It helps programmer to control PCI-9118 in high level language style.
- Chapter 7, "Calibration", describes how to calibrate the PCI-9118 for accurate measurement.
- Chapter 8, "Software Utility", describes how to run the utility program included in the software CD.

1

Introduction

The PCI-9118 series is a family of advanced performance, data acquisition card based on the 32-bit PCI Bus architecture. High performance designs and the state-of-the-art technology make this card ideal for data logging and signal analysis applications in medical, process control, and etc.

Software Supporting :

For the customer who are writing their own programs, we provide MS-DOS C/C++ programming library and Windows 95 DLL library for PCI-9118.

1.1 Features

The PCI-9118 PCI Bus Advanced Data Acquisition Card provides the following advanced features:

- 32-bit PCI-Bus, plug and play
- 12-bit (9118DG/HG) or 16-bit (9118HR) analog input resolution
- On-board A/D 1K FIFO memory
- Channel-Gain queue for high speed acquisition at different gain
- Up to 330KHz (9118DG/HG) or 100 KHz (9118HR) A/D sampling rate
- Bipolar or Unipolar input signals

- Auto-scanning channel selection
- 16 single-ended or 8 differential analog input channels
- Bipolar or Unipolar input signals
- Programmable gain of x1, x2, x4, x8 (9118DG/HR) or x1, x10, x100, x1000 (9118HG)
- Programmable burst mode sampling emulates simultaneous sample & hold
- Two 12-bit monolithic multiplying analog output channels
- 4 digital output and output channels
- Three A/D trigger modes : software trigger, programmable pacer trigger, and external pulse trigger.
- 50-pin D-type connector
- Compact size : half-size PCB

1.2 Applications

- Industrial and laboratory ON/OFF control
- Energy management
- Annunciation
- 16 TTL/DTL compatible digital input channels
- Security controller
- Product test
- Event and frequency counting
- Waveform and pulse generation
- BCD interface driver

1.3 Specifications

¨ **Analog Input (A/D)**

- **Converter :**
  9118DG/HG : B.B. ADS7800 or equivalent
  9118HR : B.B. ADS7805 or equivalent
- **Input Channels :** 16 single-ended or 8 differential
- **A/D FIFO Buffer Size :** 1024 location

- **Channel/Gain Queue Length :** 256 location
- **Resolution :** 12-bit (9118DG/HG) or 16-bit(9118HR)
- **Input Range :** (Software controlled)

  9118DG/HR :

  Bipolar : ± 5V, ±2.5V, ±1.25V, ±0.625V

  Unipolar : 0~10V, 0~5V, 0~2.5V, 0~1.25V

  9118HG :

  Bipolar : ± 5V, ±0.5V, ±0.05V, ±0.005V

  Unipolar : 0~10V, 0~1V, 0~0.1V, 0~0.01V
- **Overvoltage Protection :** 70V peak-to-peak
- **On chip sample-and hold**
- **Accuracy :** 0.01% of FSR ±1 LSB

  0.02% of FSR ±1 LSB
- **Input Impedance :** 10,000 MΩ ‖ 6pF
- **Trigger Mode :** Pre-trigger, Post-trigger, and About-Trigger
- **Data Transfer :** Program control, Interrupt, DMA (Bus mastering)
- **Data Throughput :** 330KHz (maximum) for 9118DG/HG

  100KHz (maximum) for 9118HR

¨ **Analog Output (D/A)**

- **Output Channel :** 2 analog outputs
- **Resolution :** 12-bit
- **Data Format :** Binary format or 2's S complement
- **Output Range :** Bipolar : -10V ~ 10V
- **Converter :** B.B. DAC2813 or equivalent, monolithic multiplying
- **Control Mode :** Double buffered mode or transparency mode
- **Settling Time :** 4.5 μ sec (typical), 6μ sec (max.)
- **Linearity** : ±1/2 bit LSB(Max.), ±1/4 bit LSB(typical)
- **Output Driving :** ±5mA (min.)

♦ **Digital I/O (DIO)**

- **Channel :** 4 TTL compatible inputs and outputs
- **Input Voltage :**

  Low : VIL=0.8 V max.; IIL=0.2mA max.

  High : VIH=2.0V max.; IIH=0.02mA max

Output Voltage :

Low : VOL=0.5 V max.; IOL=8mA max.
High : VOH=2.7V min; IOH=400μA

- **Programmable Counter**
  - **Device :** 82C54
  - **A/D Pacer :** 32-bit timer (two 16-bit counter cascaded together) with a 4MHz time base
  - **Max Pacer Rate :** 333 KHz
  - **Min Pacer Rate :** 0.0012 Hz

- **General Specifications**
  - **Connector :** 50-pin D-type SCSI-II connector
  - **Operating Temperature :** 0° C ~ 60° C
  - **Storage Temperature :** -20° C ~ 80° C
  - **humidity :** 5 ~ 95%, non-condensing
  - **Power Consumption :**

    PCI-9118DG/HG

    +5V @ 450mA typical
    +12V @ 200mA typical
    -12V @ 50mA typical

    PCI-9118HR

    +5V @ 485mA typical
    +12V @ 180mA typical
    -12V @ 50mA typical
  - **Dimension :** Compact size only 102mm (H) x 173mm (L)

2

Installation

This chapter describes how to install the PCI-9118DG/HG/HR. At first, the contents in the package and unpacking information that you should be careful are described.

The PCI-9118DG/HG/HR does an automatic configuration of the IRQ, port address, and BIOS address. So, you do not need to set above configuration as you use ISA form factor DAS card. For system reliability, some critical settings for analog input and output need to be assigned manually, because these settings will not be changed after your data acquisition system configuration is decided. It will let your system get more reliability and safety (user can not change the configuration by software directly) when your system is running.

2.1 What You Have

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

- PCI-9118 Enhanced Multi-function Data Acquisition Card
- Manual & Software Utility CD

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

2.2 Unpacking

Your PCI-9118 card contains sensitive electronic components that can be easily damaged by static electricity.

The card should be handled on a grounded anti-static mat. The operator should be wearing an anti-static wristband, grounded at the same point as the anti-static mat.

Inspect the card module carton for obvious damage. Shipping and handling may cause damage to your module. Be sure there are no shipping and handing damages on the module before processing.

After opening the card module carton, extract the system module and place it only on a grounded anti-static surface with component side up.

Again inspect the module for damage. Press down on all the socketed IC's to make sure that they are properly seated. Do this only with the module place on a firm flat surface.

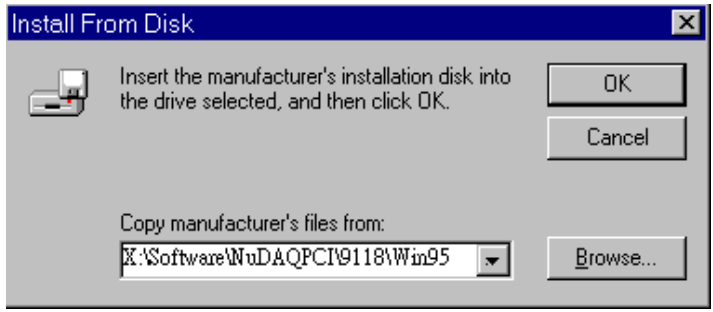
Note : DO NOT APPLY POWER TO THE CARD IF IT HAS BEEN DAMAGED.

You are now ready to install your PCI-9118.

2.3 Device Installation for Windows 95

While you first plug PCI-9118 card and enter Windows 95, the system will detect this device automatically and show the following dialog box that prompts you to select the device information source.

Choose the default option "*Driver from disk provided by hardware manufacturer*" and then a dialog box is shown to prompt you give the path of installation disk.



Place ADLink's "Manual & Software Utility" CD into the appropriate CD driver. Type "X:\Software\NuDAQPCI\9118\Win95" in the input field (X indicates the CD ROM driver) and then click OK. The system will start the installation of PCI-9118 device.

2.4 PCI-9118DG/HG Layout

Figure 2.1 PCB Layout of the PCI-9118DG/HG

2.5 PCI-9118HR Layout

Figure 2.2 PCB Layout of the PCI-9118HR

2.6 PCI Configuration

1. Plug and Play :

As a plug and play component, the board requests an interrupt number via a system call. The system BIOS responds with an interrupt assignment based on the board information and on known system parameters. These system parameters are determined by the installed drivers and the hardware load seen by the system.

2. Configuration :

The board configuration is done on a board-by-board basis for all PCI form factor boards on your system. Because configuration is controlled by the system and software, so there is no jumpers for base-address, DMA, and interrupt IRQ need to be set by the user.

The configuration is subject to change with every boot of the system as new boards are added or boards are removed. So, there is no idea what' s going on to be installed.

3. Trouble shooting :

If your system won' t boot or if you experience erratic operation with your PCI board in place, it' s likely caused by an interrupt conflict (perhaps because you incorrectly described the ISA setup). In general, the solution, once you determine it is not a simple oversight, is to consult the BIOS documentation that come with your system.

3

Signal Connections

This chapter describes the connector of the PCI-9118, also the signal connection between the PCI-9118 and external devices, such as daughter boards or other devices.

3.1 Connectors Pin Assignment

The PCI-9118 is equipped one 50-pin D-type connector - CN1.

CN1 for digital signal input, digital signal output, analog input, analog output and timer/counter's signals. The pin assignment for each connectors are illustrated in the Figure 3.1.

· **CN 1 : Digital Signal Input, Digital Signal Output, Analog Input/Output & Counter/Timer**

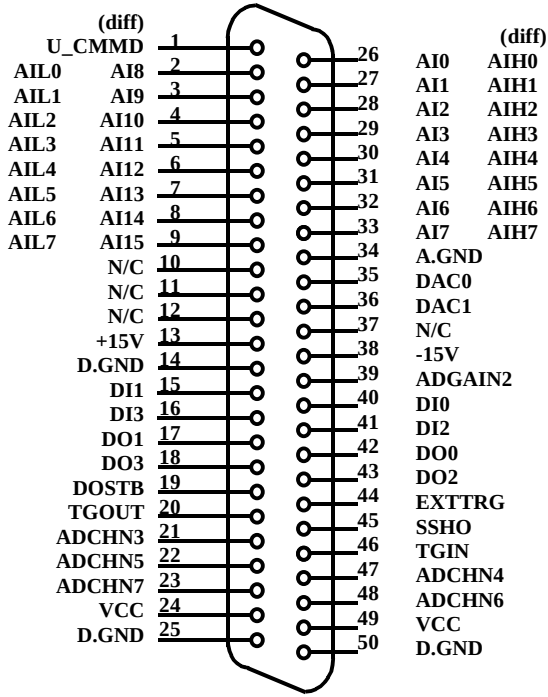


Figure 3.1 Pin Assignment of CN1

Legend :

U_CMMD	: User Defines Command Mode
AIn	: Analog Input Channel *n* (single-ended)
AIHn	: Analog High Input Channel *n* (differential)
AILn	: Analog Low Input Channel *n* (differential)
EXTTRG	: External Trigger Signal or External CLK
DIn	: Digital Input Signal Channel n
DOn	: Digital Output Signal Channel n
TGIN	: Exernal Digital Trigger or Hardware Gate
TGOUT	: Trigger/Gate Output Signal

DACn	: Analog Output Channel n
SSHO	: SSH Output Signal
DOSTB	: Strobe Signal
ADCHNn	: Multiplexer Control Line n
ADGAIN2	: External Gain Control Line
A.GND	: Analog Ground
D.GND	: Digital Ground

3.2 Analog Input Signal Connection

The PCI-9118 provides 16 single-ended or 8 differential analog input channels. The analog signal can be converted to digital value by the A/D converter. To avoid ground loops and get more accuracy measurement of A/D conversion, it is quite important to understand the signal source type and how to choose the analog input modes : signal-ended and differential. The PCI-9118 offers jumpers to select 16 single-ended or 8 different analog inputs.

Single-ended Mode

The single-ended mode has only one input relative to ground and it suitable for connecting with the *floating signal source*. The floating source means it does not have any connection to ground. Figure 3.2 shows the single-ended connection. Note that when more than two floating sources are connected, the sources must be with common ground.

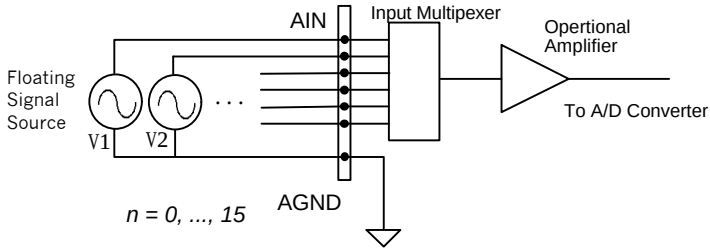


Figure 3.2 Floating source and single-ended

Differential input mode

The differential input mode provides two inputs that respond to the difference signal between them. If the signal source has one side connected to local ground, the differential mode can be used for reducing ground loop. Figure 3.3 shows the connection of the differential input mode. However, even if the signal source is local grounded, the single-ended still can be used when the Vcm (Common Mode Voltage) is very small and the effect of ground loop can be negated.

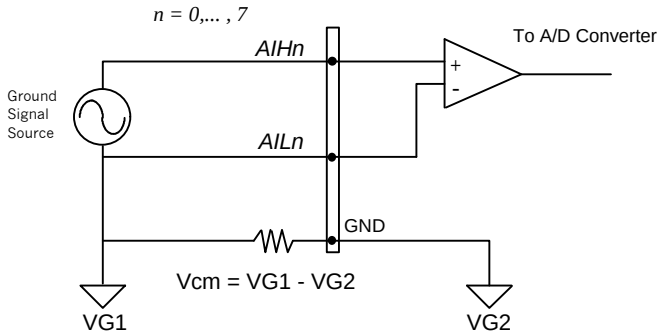


Figure 3.3 Ground source and differential input

A differential mode must be used when the signal source is differential. A differential source means the ends of the signal are not grounded. To avoid the danger of high voltage between the local ground of signal and the ground of the PC system, a shorted ground path must be connected. Figure 3.4 shows the connection of differential source.

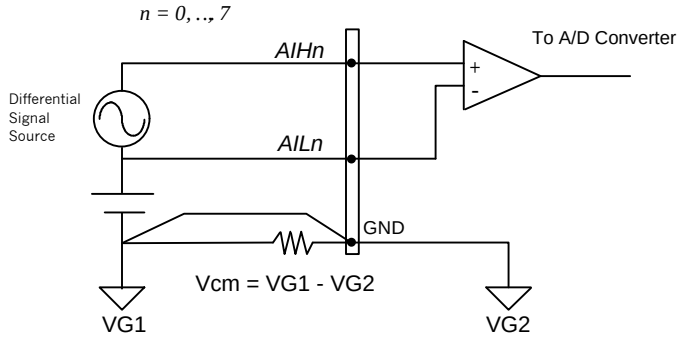


Figure 3.4 Differential source and differential input

If your signal source is both floating and local ground, you should use the differential mode, and the floating signal source should be connected as the Figure 3.5.

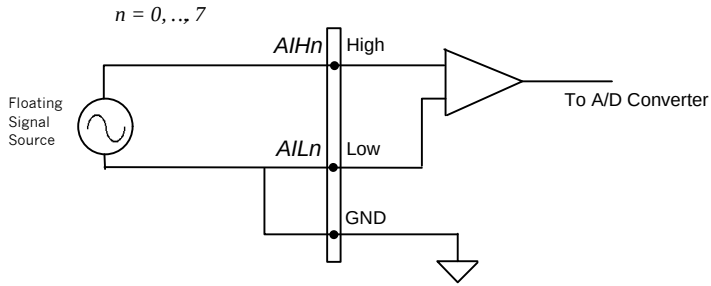


Figure 3.5 Floating source and differential input

3.3 Analog Output Signal Connection

The PCI-9118 has two analog output channels. To make the D/A output connections from the appropriate D/A output, please refer Figure 3.6.

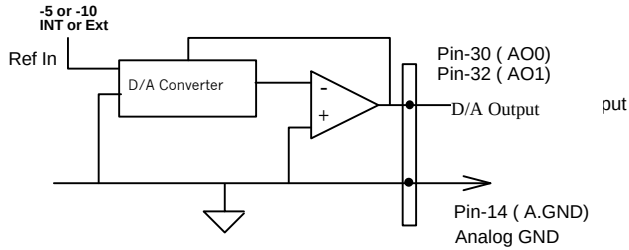


Figure 3.6 Connection of Analog Output Connection

3.4 Digital I/O Connection

The PCI-9118 provides 4 digital input and 4 digital output channels on board. The digital I/O signal are fully TTL/DTL compatible. The detailed digital I/O signal specification can be referred in section 1.3.

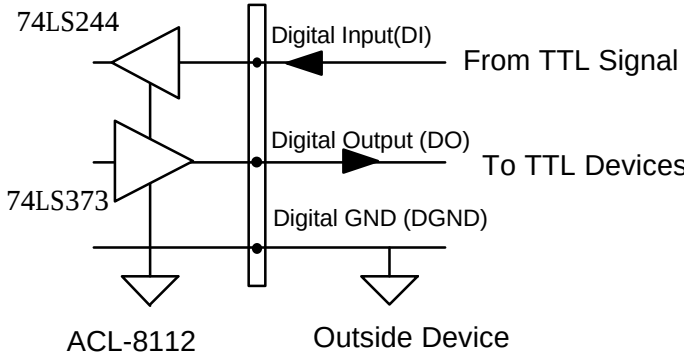


Figure 3.7 Digital I/O Connection

3.5 Timer / Counter Connection

The PCI-9118 has an internal timer/counter 8254 on board. It offers 3 independent 16-bit programmable down counters; counter 1 and counter 2 are cascaded together for A/D timer pacer trigger of A/D conversion, and counter 0 is free for your applications. Figure 3.8

shows the 8254 timer/counter connection.

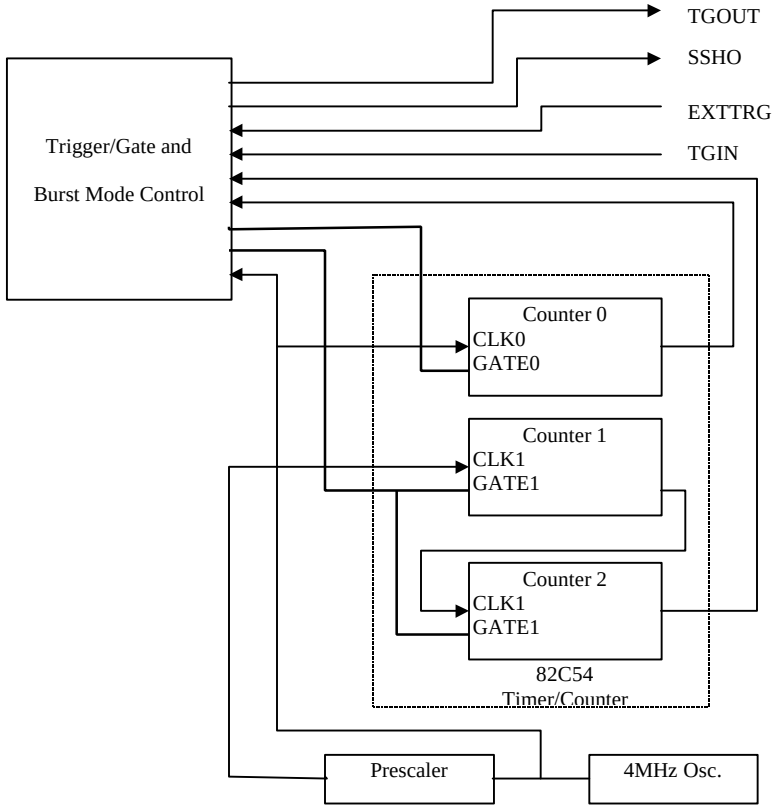


Figure 3.8 Block Diagram of 8254 Timer/Counter

The clock source of counter 0 can be internal or external from EXTTRIG (pin44). As to counter 1 and counter 2, the clock source is internally fixed, while the gate can be controlled externally by TRIN (pin46). All the timer/ counter signals are TTL compatible.

4

Registers Format

The detailed descriptions of the register format and structure of the PCI-9118DG/HG/HR are specified in this chapter. This information is quite useful for the programmer who wish to handle the card by low-level program.

In addition, the low level programming syntax is introduced. This information can help the beginners to operate the PCI-9118 DG/HG/HR in the shortest learning time.

4.1 I/O Port Address

The PCI-9118 DG/HG/HR functions as 32-bit PCI target device to any master on the PCI bus. It supports burst transfer to memory space by using 32-bit data. So, all data read and write will base on 32-bit data. The Table 4.1 shows the I/O address of each register with respect to the base address. The function of each register also be shown.

I/O Address	Read	Write
Base + 0x00	Counter 0	Counter 0
Base + 0x04	Counter 1	Counter 1
Base + 0x08	Counter 2	Counter 2
Base + 0x0C	----------	8254 Counter Control
Base + 0x10	A/D Data Reg.	CH1 D/A Data Reg.
Base + 0x14	----------	CH2 D/A Data Reg.
Base + 0x18	A/D Status Reg.	A/D Control Reg.
Base + 0x1C	Digital IN Reg.	Digital OUT Reg.
Base + 0x20	----------	Software Trigger
Base + 0x24	----------	A/D Gain/Channel Reg.
Base + 0x28	----------	A/D Burst No. Reg.
Base + 0x2C	----------	A/D Auto Scan Mode
Base + 0x30	----------	A/D Function Reg.
Base + 0x34	----------	A/D Data FIFO Reset
Base + 0x38	Interrupt Reason Reg.	Interrupt Control Reg.

Table 4.1 I/O Address

4.2 A/D Data Registers

The PCI-9118 provides 16 single-ended or 8 differential A/D input channels, the digital data will store in the A/D data registers. The 12 bits A/D data is put into 32 bits registers.

Address : BASE + 10
Attribute : read only
Data Format :

Bit	7	6	5	4	3	2	1	0
BASE+10	AD3	AD2	AD1	AD0	CH3	CH2	CH1	CH0
BASE+11	AD11	AD10	AD9	AD8	AD7	AD6	AD5	AD4
BASE+12	---	---	---	---	---	---	---	---
BASE+13	---	---	---	---	---	---	---	---

AD11 .. AD0 : Analog to digital data. AD11 is the Most Significant Bit (MSB). AD0 is the Least Significant Bit (LSB).
CH3 ~ CH0 : A/D channel number from which the data is derived.
--- : Don't care

4.3 D/A Output Register

The D/A converter will convert the D/A output register data to the analog signal. The register data of the address Base+10 is used for D/A channel 1, Base+14 is used for D/A channel 2.

Address : BASE + 10
Attribute : write only
Data Format : (for D/A Channel 1)

Bit	7	6	5	4	3	2	1	0
Base + 10	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
Base + 11	---	---	---	---	DA11	DA10	DA9	DA8
Base + 12	---	---	---	---	---	---	---	---
Base + 14	---	---	---	---	---	---	---	---

Address : BASE + 14
Attribute : write only
Data Format : (for D/A Channel 2)

Bit	7	6	5	4	3	2	1	0
Base + 14	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
Base + 15	---	---	---	---	DA11	DA10	DA9	DA8
Base + 16	---	---	---	---	---	---	---	---
Base + 17	---	---	---	---	---	---	---	---

DA0 is the LSB and DA11 is the MSB of the 12 bits data.
--- : don't care

4.4 A/D control Register

This register is used to control the A/D mode. It's a write only register.

Address : BASE + 18
Attribute : write only
Data Format :

Bit	7	6	5	4	3	2	1	0
Base + 18	**UniP**	**Diff**	**SoftG**	**ExtG**	**ExtM**	**TmrTr**	**Int**	**Dma**
Base + 19	---	---	---					
Base + 1A	---	---	---	---	---	---	---	---
Base + 1B	---	---	---	---	---	---	---	---

UniP (Bit7) : Unipolar or Bipolar A/D input
1: unipolar
0: bipolar

Diff (Bit6) : Differential or Single ended A/D input
1: Differential End
0: Single End

SoftG(Bit5) : Software Gate Control Output
1: 8254 counter works
0: 8254 counter stops

ExtG(Bit4) : External or Soft Gate Control Mode
1: 8254 counter controlled by TGIN(connector pin 46)
0: 8254 counter controlled by SoftG

ExtM(Bit3) : External Hardware Trigger mode
1: External Hardware Trigger (connector pin44)
0: Internal Hardware Trigger

TmrTr(Bit2) : Timer Trigger Mode
1: 8254 Timer (Counter) is internal trigger source
0: Software Trigger is internal trigger source

Int(Bit1) : Interrupt Control Bit
1: Enable Hardware interrupt
0: Disable Hardware interrupt

Dma(Bit0) : A/D Data DMA Transfer Mode
1: Enable A/D Data DMA Data Transfer
0: Disable A/D Data DMA Data Transfer

4.5 A/D Status Register

Address : BASE + 18
Attribute : read only
Data Format :

Bit	7	6	5	4	3	2	1	0
Base + 18	nHFull	NEpty	Acmp	DTH	Bover	ADOS	ADOR	ADrdy
Base + 19	---	---	---	---	---	---	---	nFull
Base + 1A	---	---	---	---	---	---	---	---
Base + 1B	---	---	---	---	---	---	---	---

nFull (Bit8) : A/D FIFO Full status (Fatal Error !)
0: FIFO Full
1: FIFO not Full

nHFull(Bit7): A/D FIFO Half Full status
0: FIFO Half Full
1: FIFO not Half Full

nEpty (Bit6) : A/D FIFO Empty status
0: FIFO Empty
1: FIFO not Empty

Acmp (Bit5) : About Trigger Complete Status
1: About Trigger already complete
0: About Trigger not complete

DTH (Bit4) : External Digital Trigger Happened Status
1: External Digital Trigger ever Happened
0 : External Digital Trigger not Happen

Bover(Bit3) : A/D Burst Mode Overrun Status (Fatal Error !)
1: Burst Mode Overrun
0 : Burst Mode not Overrun

ADOS(Bit2) : A/D Over Speed Status (Warning !)
1: A/D Over Speed
0: A/D not Over Speed

ADOR(Bit1) : A/D Overrun Status (Fatal Error !)
1: A/D Overrun
0: A/D not Overrun

ADrdy(Bit0): A/D ready status
1: A/D already ready
0: A/D not ready

4.6 Digital I/O register

There are 4 digital input channels and 4 digital output channels provided by the PCI-9118. The address Base + 1C is used to access digital inputs and control digital outputs.

Address : BASE + 1C
Attribute : read only
Data Format :

Bit	7	6	5	4	3	2	1	0
Base + 1C	DO3	DO2	DO1	DO0	DI3	DI2	DI1	DI0
Base + 1D	---	---	---	---	---	---	---	---
Base + 1E	---	---	---	---	---	---	---	---
Base + 1F	---	---	---	---	---	---	---	---

Address : BASE + 1C
Attribute : write only
Data Format :

Bit	7	6	5	4	3	2	1	0
Base + 1C	---	---	---	---	DO3	DO2	DO1	DO0
Base + 1D	---	---	---	---	---	---	---	---
Base + 1E	---	---	---	---	---	---	---	---
Base + 1F	---	---	---	---	---	---	---	---

4.7 Software Trigger Register

If you want to generate a trigger pulse to the PCI-9118 for A/D conversion, you just write any data to this register, and then the A/D converter will be triggered.

Address : BASE + 20
Attribute : write only
Data Format :

Bit	7	6	5	4	3	2	1	0
BASE+20	---	---	---	---	---	---	---	---
BASE+21	---	---	---	---	---	---	---	---
BASE+22	---	---	---	---	---	---	---	---
BASE+23	---	---	---	---	---	---	---	---

4.8 Internal Timer/Counter Register

Two counters of 8254 are used for periodically triggering the A/D conversion, the left one is left free for user applications. The 8254 occupies 4 I/O address locations in the PCI-9118 as shown blow. Users can refer to NEC's or Intel's data sheet for a full description of the 8254 features.

Address : BASE + 0 ~ BASE + F
Attribute : read / write
Data Format :

Base + 0	Counter 0 Register (R/W)
Base + 4	Counter 1 Register (R/W)
Base + 8	Counter 2 Register (R/W)
Base + C	8254 CONTROL BYTE (W)

4.9 A/D Gain/Channel Register

Address : BASE + 0x24
Attribute : write only
Data Format :

Bit	7	6	5	4	3	2	1	0
Base+0x24	CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0
Base+0x25	---	---	---	---	---	---	Gain1	Gain0
Base+0x26	---	---	---	---	---	---	---	---
Base+0x27	---	---	---	---	---	---	---	---

CH7 ~ CH4 (bit7~ bit4) : External A/D Channel selection bits
CH3 ~ CH0 (bit3~ bit0) : Internal A/D Channel selection bits

Gain1~Gain0 (bit9~bit8) : Gain selection bits

bits 9~8			
(Gain1 Gain 0)			
11	10	01	00
1000/8	100/4	10/2	1

4.10 A/D Burst Number Register

Address : BASE + 0x28
Attribute : write only
Data Format :

Bit	7	6	5	4	3	2	1	0
Base+0x28	Burst Number[7..0]							
Base+0x29	---	---	---	---	---	---	---	---
Base+0x2A	---	---	---	---	---	---	---	---
Base+0x2B	---	---	---	---	---	---	---	---

4.11 A/D Auto Scan Mode

Address : BASE + 0x2C
Attribute : write only
Data Format :

Bit	7	6	5	4	3	2	1	0
Base+0x2C	---	---	---	---	---	---	ClrSet	BgnSet
Base+0x2D	---	---	---	---	---	---	---	---
Base+0x2E	---	---	---	---	---	---	---	---
Base+0x2F	---	---	---	---	---	---	---	---

BgnSet	ClrSet	Description
X	1	Clear A/D Channel/Gain Register
1	0	user can set A/D Channel/Gain Register
0	0	user cannot set A/D Channel/Gain Register

4.12 A/D Function Register

Address : BASE + 0x30
Attribute : write only
Data Format :

Bit	7	6	5	4	3	2	1	0
Base+0x30	PDTrg	PETrg	BSSH	BM	BS	PM	AM	Start
Base+0x31	---	---	---	---	---	---	---	---
Base+0x32	---	---	---	---	---	---	---	---
Base+0x33	---	---	---	---	---	---	---	---

PDTrg (bit7): Digital Trigger Positive/Negative Active
1: Positive Trigger
0: Negative Trigger

Note: Only "Positive Trigger" setting is valid for the trigger acquisition of PCI-9118.

PETrg (bit6): External Trigger Positive/Negative Active
1: Positive Trigger
0: Negative Trigger
BSSH (bit5) : A/D Burst Mode Sample and Hold Control
1: with Sample and Hold
0: without Sample and hold
BM (bit4) : A/D Burst Mode Control
1: Burst Mode
0: Normal Mode
BS (bit3) : A/D Burst Start Control
1: Burst Mode Start
0: Burst Mode Stop

BM	BS	Description
0	X	A/D Normal Mode
1	0	user can initialize 8254 for Burst Mode Control
1	1	Burst Mode Begin

PM (bit 2) : Post Trigger Mode
1: Post Trigger
0: Not Post Trigger

AM (bit 1) : About Trigger Mode
1: About Trigger
0: not About Trigger
Start (bit 0) : Trigger Start Control
1: Trigger Start
0: Trigger Stop

PM	AM	Start	Description
1	0	0	Post Trigger Mode ,user can initialize 8254 for setting trigger number
1	0	1	Post Trigger Mode Start
0	1	0	About Trigger Mode ,user can initialize 8254 for setting trigger number
0	1	1	About Mode Start
0	0	X	Trigger Mode Stop

4.13 A/D FIFO Reset Register

You can reset A/D FIFO by writing any value into this register

Address : BASE + 0x34
Attribute : write only
Data Format :

Bit	7	6	5	4	3	2	1	0
Base+0x34	---	---	---	---	---	---	---	---
Base+0x35	---	---	---	---	---	---	---	---
Base+0x36	---	---	---	---	---	---	---	---
Base+0x37	---	---	---	---	---	---	---	---

4.14 Interrupt Control Register

Address : BASE + 0x38
Attribute : write only
Data Format :

Bit	7	6	5	4	3	2	1	0
Base+0x38	---	---	---	---	Timer	About	Hfull	DTrg
Base+0x39	---	---	---	---	---	---	---	---
Base+0x3A	---	---	---	---	---	---	---	---
Base+0x3B	---	---	---	---	---	---	---	---

Timer (bit3) : Timer Interrupt Enable Control
1: Enable
0: Disable
About (bit2) : About Trigger Complete Interrupt Enable Control
1: Enable
0: Disable
HFull (bit1) : A/D FIFO Half Full Interrupt Enable Control
1: Enable
0: Disable
DTrg (bit0) : External Digital Trigger Interrupt Enable Control
1: Enable
0: Disable

4.15 Interrupt Status Register

Address : BASE + 0x38
Attribute : read only
Data Format :

Bit	7	6	5	4	3	2	1	0
Base+0x38	---	---	---	---	Timer	About	Hfull	DTrg
Base+0x39	---	---	---	---	---	---	---	---
Base+0x3A	---	---	---	---	---	---	---	---
Base+0x3B	---	---	---	---	---	---	---	---

Timer (bit3) :	Timer Interrupt index 1: Interrupt Occur 0: Interrupt not Occur
About (bit2) :	About Trigger Complete Interrupt 1: Interrupt Occur 0: Interrupt not Occur
HFull (bit1) :	A/D FIFO Half Full Interrupt index 1: Interrupt Occur 0: Interrupt not Occur
DTrg (bit0) :	External Digital Trigger Interrupt index 1: Interrupt Occur 0: Interrupt not Occur

5

Operation Theorem

The operation theorem of the functions on PCI-9118DG/HG/HR card is described in this chapter. The functions include the A/D conversion, D/A conversion, Digital I/O and counter / timer. The operation theorem can help you to understand how to manipulate or to program the PCI-9118DG/HG/HR.

5.1 A/D Conversion

Before programming PCI-9118DG/HG/HR to perform the A/D conversion, you should understand the following issues:

- A/D conversion procedure
- A/D trigger source and conversion modes
- A/D data transfer mode
- Trigger modes
- Specifying channels and gains in the Channel-Gain Queue

5.1.1 A/D Conversion Procedure

The A/D conversion is starting by a trigger source, then the A/D converter will start to convert the signal to a digital value. PCI-9118DG/HG/HR provides three trigger modes, described in section 5.1.4.

While A/D conversion, the A*Drdy* bit in *A/D status register* is cleared to indicate the data is not ready. After conversion being completed, the A*Drdy* bit will return to high(1) level. It means users can read the converted data from the A/D data registers. Please refer to section 4.5 for the A/D status register format.

The A/D data should be transferred into PC's memory for further use. The PCI-9118DG/HG/HR provides three data transfer modes that allow users to optimize the DAS system. Refer to section 5.1.3 for data transfer modes.

5.1.2 A/D Trigger Sources and conversion modes

In PCI-9118DG/HG/HR, A/D conversion can be triggered by the *Internal* or *External* trigger source. The ExtM bit of A/D control register is used to select the internal or external trigger, please refer to section 4.4 for the details. Whenever the external source is set, the internal sources are disable.

If the internal trigger is selected, two trigger source, software trigger and the timer pacer trigger, can be used. The A/D trigger source is controlled by A/D mode bits (ExtM, TmrTr) of A/D control register (BASE+18). All of the three trigger sources are possible in the PCI-9118DG/HG/HR. In addition, for DMA operation, you can specify burst mode or burst mode with SSH (simultaneous sample-and-hold) to perform data conversion. The two modes are controlled by BM and BSSH bits of A/D Function Register (BASE+30). The different trigger and conversion conditions are specified as follows:

Software trigger

The trigger source is software controllable in this mode. That is, the A/D conversion is starting when any value is written into the software trigger register (BASE+20). This trigger mode is suitable for low speed A/D conversion. Under this mode, the timing of the A/D conversion is fully controlled under software. However, it is difficult to control the fixed A/D conversion rate except another timer interrupt service routine is used to generate a fixed rate trigger.

Timer Pacer Trigger

An on-board timer / counter chip 8254 is used to provide a trigger source for A/D conversion at a fixed rate. Two counters of the 8254 chip are cascaded together to generate trigger pulse with precise period. Please refer to section 5.4 for 8254 architecture. This mode is ideal for high speed A/D conversion. It can be combined with the DMA bus mastering or the interrupt data transfer. It's recommend to use this mode if your applications need a fixed and precise A/D sampling rate.

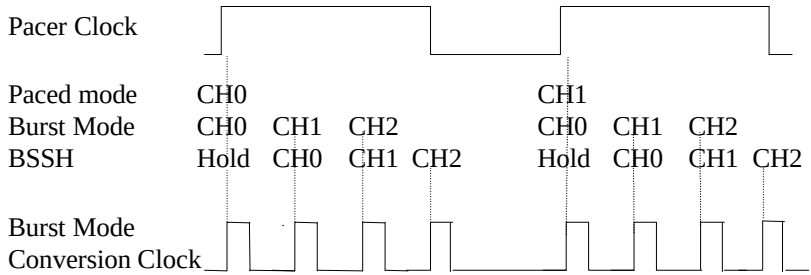
Burst Mode

Use burst mode if you want to accurately control the period between conversions of individual channels in a scan and the period between conversions of the entire scan. An on-board timer / counter chip 8254 is used to provide a trigger source for A/D conversion. Two counters of the 8254 chip are cascaded together. The burst conversion frequency is decided by the frequency divider of counter1 (c1) , i.e. the conversion rate is 4M/c1.
The frequency of Pacer timer is decided by the frequency divider of counter1 and counter2 (c1*c2), i.e. the pacer clock rate is 4M/c1*c2.

Note: The pacer rate can not be larger than the value of the burst conversion rate divided by the burst number. Hence, The number of scanned channels in the burst can't be larger than c2.

Burst Mode with SSH

Use burst mode with SSH if you want to accurately control both the period between conversions of the entire scan and if you want to simultaneously sample all channels in a scan. Each pulse from the pacer clock starts a simultaneous scan of all channels. The idea about pacer rate and conversion rate is the same as those of Burst Mode. However, One extra tick of the burst mode conversion clock is required to sample and hold the values. Therefore, the sample rate can be no more than the value of the burst mode conversion rate divided by the sum of one plus the number of channels in the burst. The difference between timer pacer mode, burst mode and burst mode with SSH is illustrated in the following figure:



Note: The 'HOLD' signal is sent to external S&H circuits to hold the analog signal. There must have an external circuits to carry out the S&H function, there is no on-board device with S&H.

External Trigger

Through the pin-44 of CN1 (*ExtTrig*), the A/D conversion also can be performed when a rising edge of external signal is occurred. The conversion rate of this mode is more flexible than the previous two modes, because the users can handle the external signal by outside device. The external trigger can combine with the DMA transfer, interrupt data transfer, or even program polling data transfer. Generally, the interrupt data transfer is often used when external trigger mode is used.

5.1.3 A/D Data Transfer Modes

On the PCI-9118DG/HG/HR, three A/D data transfer modes can be used when the conversion is completed. The data transfer mode is controlled by the A/D mode control bits (Int, Dma) of the A/D control register (BASE+18). The different data transfer modes are specified as follows:

Software Data Transfer (ADrdy)

Usually, this mode is used with software A/D trigger mode. After the A/D conversion is triggered by software, the software should poll the ***ADrdy*** bit on the A/D Status register until it becomes to high level. Whenever the low byte of A/D data is read, the ***ADrdy*** bit will be cleared to indicate the data is read out.

It is possible to read A/D converted data without polling. The A/D conversion time will not excess 8μs on PCI-9118 card. Hence, after software trigger, the software can wait for at least 8μs then read the A/D register without polling.

Interrupt Transfer (inX)

The PCI-9118 provides hardware interrupt capability. Under this mode, an interrupt signal is generated when the A/D conversion is ended and the data is ready to be read. It is useful to combine the interrupt transfer with the timer pacer trigger mode. Under this mode, the data transfer is essentially asynchronous with the control software.

When the interrupt transfer is used, the hardware interrupt will be inserted and its corresponding ISR (Interrupt Service Routine) will be invoked and executed after A/D conversion is completed. The converted data is transferred by the ISR program. In PCI design, the IRQ level is assigned by BIOS directly.

DMA Transfer (Dma)

The DMA (Direct Memory Access) bus master allows data to be transferred directly between the PCI-9118 and the PC memory at the fastest possible rate, without using any CPU time. The A/D data will be queue at local FIFO on the PCI-9118 itself and it is automatically transferred to PC's memory.

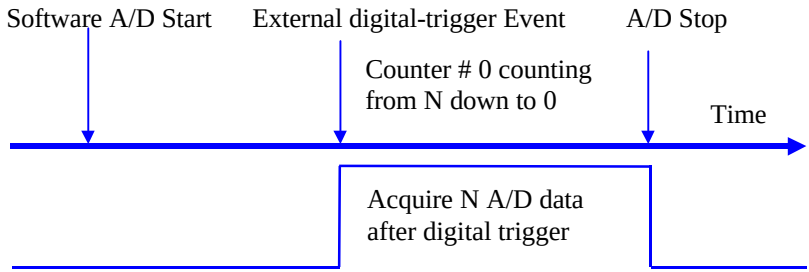
The DMA transfer mode is very complex to program. It is recommended to use the high level program library to operate this card. If you wish to program the software which can handle the DMA bus master data transfer, please refer to more information about PCI controller.

5.1.4 Trigger Acquisition Modes

PCI-9118 provides three types of trigger acquisition modes, pre-trigger, post-trigger and about-trigger. They are described in the following paragraph:

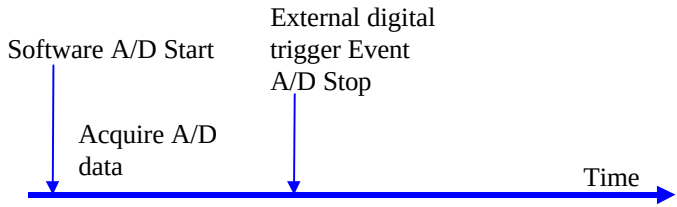
About-trigger

Use about-trigger acquisition in application where you want to collect data before and after a specific trigger event. The digital trigger is input from pin 46 (TGIN) on CN1. To set about trigger mode, set AM bit of A/D Function Register as ' 1' and specify the 8254 counter value. To start the about trigger acquisition, set Start bit of A/D Function Register as ' 1' . The operation stops when the specified number of samples has been acquired after digital trigger event occurs or DMAStop. The maximum counter number can be set is 64K, i.e. After trigger event occurs, the maximum number of data can be accessed is 64K.



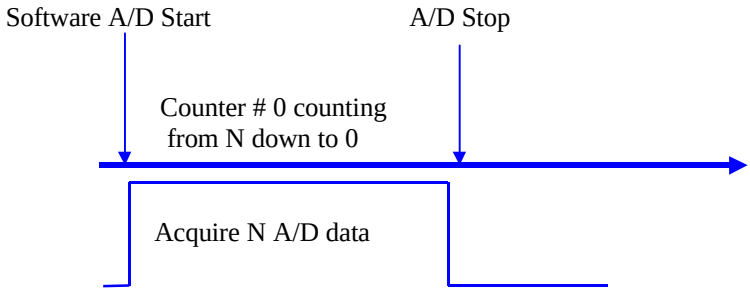
Pre-trigger

Use pre-trigger acquisition in application where you want to collect data before a specified trigger event. The trigger is a digital trigger which is input from pin 46 (TGIN) on CN1. To set pre-trigger mode, set AM bit of A/D Function Register as ' 1' and set 8254 counter value as 1. To start the pre-trigger, set Start bit of A/D Function Register as ' 1' . The operation stops after digital -trigger event occurs or DMAStop.



Post-trigger

Use post-trigger acquisition in application where you want to collect data after the start condition. To set post-trigger mode, set PM bit of A/D Function Register as ' 1' and set the 8254 counter value. To start the post-trigger, set Start bit of A/D Function Register as ' 1' . The operation stops after specified number of samples has been acquired or DMAStop. The maximum number of counter value can be set is 64K.



5.1.5 Specifying Channels and Gains in the Channel-Gain Queue

For analog input operation through interrupt and DMA mode, the channel and ranges you want to acquire samples can be specified in a hardware channel-gain queue. You can fill the channel number in the channel-gain queue in consecutive order or in nonconsecutive order. Therefore you can control the channel order for acquiring samples and the gain code for each channel. The maximum number of entries you can set is 255 channels. The channel order for acquiring is the same as the order you set in the channel-gain queue. When the specified channels are read from the first channel to the last channel, the channels in the queue are then sampled again until the specified number of samples is acquired. Please refer to section 6.2.14 to learn how to use 9118 library to set the channels and gains in the Channel-Gain Queue.

5.2 D/A Conversion

The operation of D/A conversion is more simple than A/D operation. You only need to write Digital values into the D/A data registers and the corresponding voltage will be output from the AO. Refer to section 4.3 for information about the D/A data registers. The mathematical relationship between the Digital number DAn and the output voltage is formulated as following:

$$Vout = span \times DAn / 4096 - 10 \quad \text{-- Bipolar}$$

where the *span* is the span in volts. Since the output range is -10V~10V (Bipolar), the span is 20. The *Vout* is the output voltage, and the *DAn* is the Digital value in D/A data registers.

5.3 Digital Input and Output

To program digital I/O operation is fairly straight forward. The digital input operation is just to read data from the corresponding registers, and the digital output operation is to write data to the corresponding registers. The digital I/O registers‘ format are shown in section 4.6. Note that the DIO data channel can only be read or written in form of 16 bits together. It is impossible to access individual bit channel.

5.4 Timer/Counter Operation

The PCI-9118 has an interval timer/counter 8254 on board. Refer to section 3.5 for the signal connection and the configuration of the counters.

The 8254 Timer / Counter Chip

The Intel (NEC) 8254 contains three independent, programmable, multi-mode 16 bit counter/timers. The three independent 16 bit counters can be clocked at rates from DC to 5 MHz. Each counter can be individually programmed with 6 different operating modes by appropriately formatted control words. The most commonly uses for the 8254 in microprocessor based system are:

- programmable baud rate generator
- event counter
- binary rate multiplier
- real-time clock
- digital one-shot
- motor control

For more information about the 8254, please refer to the NEC Microprocessors and peripherals or Intel Microsystems Components Handbook.

Pacer Trigger Source

The counter 1 and counter 2 are cascaded together to generate the timer pacer trigger of A/D conversion. The frequency of the pacer trigger is software controllable. The maximum pacer signal rate is 4MHz/4=1000K which exceeds the maximum A/D conversion rate of the PCI-9118. The minimum signal rate is 4MHz/65536/65536, which is a very slow frequency that user may never use it.

General Purpose Timer/ Counter

The counter 0 is free for users' applications. The clock source, gate control signal. The general purpose timer/counter can be used as event counter, or used for measuring frequency, or others functions.

I/O Address

The 8254 in the PCI-9118 occupies 4 I/O address as shown below.

BASE + 0x0	LSB OR MSB OF COUNTER 0
BASE + 0x4	LSB OR MSB OF COUNTER 1
BASE + 0x8	LSB OR MSB OF COUNTER 2
BASE + 0xC	CONTROL BYTE

The programming of 8254 is control by the registers BASE+0 to BASE+C. The functionality of each register is specified this section. For more detailed information, please refer handbook of 8254 chip.

Control Byte

Before loading or reading any of these individual counters, the control byte (BASE+C) must be loaded first. The format of the control byte is :

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	SC1	SC0	RL1	RL0	M2	M1	M0	BCD

- SC1 & SC0 - Select Counter (Bit7 & Bit 6)

SC1	**SC0**	**COUNTER**
0	0	Select Counter 0

0	1	Select Counter 1
1	0	Select Counter 2
1	1	ILLEGAL

- RL1 & RL0 - Select Read/Load operation (Bit 5 & Bit 4)

RL1	RL0	OPERATION
0	0	COUNTER LATCH FOR STABLE READ
0	1	READ/LOAD LSB ONLY
1	0	READ/LOAD MSB ONLY
1	1	READ/LOAD LSB FIRST, THEN MSB

- M2, M1 & M0 - Select Operating Mode (Bit 3, Bit 2, & Bit 1)

M2	M1	M0	MODE
0	0	0	0
0	0	1	1
x	1	0	2
x	1	1	3
1	0	0	4
1	0	1	5

- BCD - Select Binary/BCD Counting (Bit 0)

0	16-BITS BINARY COUNTER
1	BINARY CODED DECIMAL (BCD) COUNTER (4 DIGITAL)
Note	The count of the binary counter is from 0 up to 65,535 and the count of the BCD counter is from 0 up to 9,999

Mode Definitions

In 8254, six operating modes can be selected. They are :

- **Mode 0** : Interrupt on Terminal Count
- **Mode 1** : Programmable One-Shot.
- **Mode 2** : Rate Generator.
- **Mode 3** : Square Wave Rate Generator.
- **Mode 4** : Software Triggered Strobe.
- **Mode 5** : Hardware Triggered Strobe.

All detailed description of these six modes are written in Intel Microsystems Components Handbook Volume II Peripherals.

6

C/C++ Software Library

There are 49 function calls provided by the C Language library. This library includes all the functions of PCI-9118 DG/HG/HR. The capabilities of these function calls include A/D conversion, D/A conversion, Digital Input and Output, etc. In addition, there are some sample programs to help you use this library.

6.1 Installation

Installation

- **MS-DOS Software Installation**

step 1. Place ADLink's "Manual & Software Utility" CD into the appropriate CD driver.

step 2. Type the command (X indicates the CD ROM driver):

X:\> **CD Software\NuDAQPCI\9118\DOS**

X:\ Software\NuDAQPCI\9118\DOS> **SETUP**

step 3. An *installation completed* message will be shown on the screen.

After installation, all the files *of PCI-9118 Library & Utility for DOS* are stored in C:\ADLink\9118\dos directory.

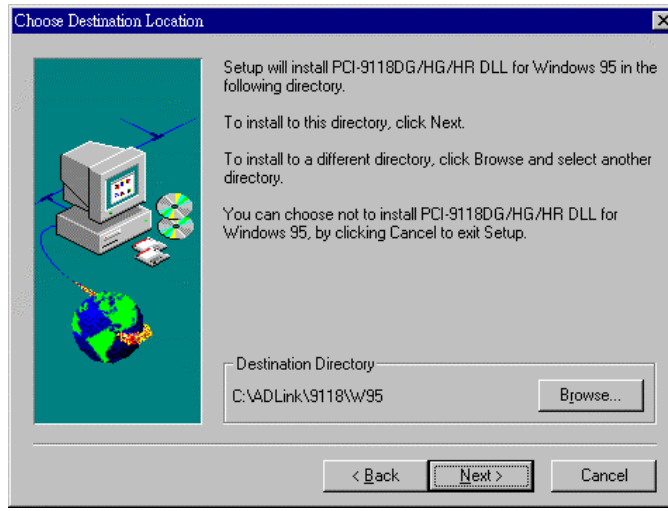
◆ **Window 95 Software Installation**

step 1. Place the ADLink's "Manual & Software Utility" CD into the appropriate CD driver.

step 2. If Windows 95 is loaded, choose Run from the Start menu.

step 3. Type **X:\Software\NuDAQPCI\9118\Win95\Setup.exe** in the Run dialog box. (X indicates the CD ROM driver).

After a welcome dialog box, Setup prompts the following dialog box for you to specify the destination directory. The default path is C:\ADLink\9118\W95. If you want to install *PCI-9118 DLL for Windows 95* in another directory, please click Browse button to change the destination directory. Then you can click Next to begin installing *PCI-9118 DLL for Windows 95*.



After you complete the installation of PCI-9118 Software, PCI-9118's DLL(9118.DLL) is copied to Windows System directory (default is C:\WINDOWS\SYSTEM) and the driver

files (W95_9118.VXD and PCIW95.VXD) are also copied to the appropriate directory.

6.2 C/C++ Programming Library

We defined some data type in acl_pci.h. These data types are used by PCI-9118 library. We suggest you to use these data types in your application programs. The following table shows the data type names and their range.

Data Types

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

The operation theorem of the functions on PCI-9118 card is described in this chapter. The functions include the A/D conversion, D/A conversion, Digital I/O and counter / timer. The operation theorem can help you to understand how to manipulate or to program the PCI-9118.

The functions of PCI-9118DG/HG/HR software drivers use full-names to represent the functions' real meaning. The naming convention rules are :

In DOS Environment :

_{hardware_model}_{action_name}. e.g. **`_9118_Initial()`**.

In order to recognize the difference between DOS library and Windows 95 library, A capital "W" is put on the head of each function name of the Windows 95 DLL driver. e.g. **`W_9118_Initial()`**.

There are 52 functions provided by PCI-9118 software drivers. The detail descriptions of each function are specified in the following sections.

6.2.2 _9118_Initial

@ Description

This function is used to initialize PCI-9118. Every PCI-9118 has to be initialized by this function before calling other functions.

@ Syntax

C/C++ (DOS)

```
int _9118_Initial (I16 cardNo, I16 *base_address, I16
  *irq_no )
```

C/C++ (Windows 95)

```
int W_9118_Initial (int cardNo, U16 *base_address1,U16
  *base_address2, U8 *irq_no, U8 *pci_master)
```

Visual Basic (Windows 95)

```
W_9118_Initial (ByVal cardNo As Long, base_add1 As Long,
    base_add2 As Long, irq_no As Byte, pci_master As
    Byte) As Long
```

@ Argument

cardNo :	the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.
base_address (DOS):	the I/O port base address of the card, it is assigned by system BIOS
op_base_address (Win-95):	the physical location of S5933 operation Registers in I/O space.
pt_base_address (Win-95):	the physical location of add-on registers in pass-through I/O space. (This argument is the same as **base_address** of DOS version)
irq_no :	the interrupt IRQ level of your PCI-9118 card, this available irq value is automatically assigned by system BIOS.

pci_master : BIOS enables or disables bus mastering in PCI Command Register

@ Return Code

ERR_NoError
ERR_PCIBiosNotExist
ERR_PCICardNotExist
ERR_PCIIrqNotExist

6.2.3 _9118_Switch_Card_No

@ Description

This function is used on multi-cards system. After the PCI-9118 cards are initialized by _9118_Initial function, you can use this function to select which one you want to operate. This function is only supported by DOS version.

@ Syntax

C/C++ (DOS)

int _9118_Switch_Card_No(int cardNo)

@ Argument

cardNo : the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

@ Return Code

ERR_NoError
ERR_BoardNoInit

6.2.4 _9118_DI

@ Description

This function is used to read data from digital input port. There are 4 digital input channels on PCI_9118. The all 4 bits can be accessed by this function directly.

@ Syntax

C/C++ (DOS)

int _9118_DI (unsigned int far *data)

C/C++ (Windows 95)

int W_9118_DI(U16 cardNo, unsigned int *data)

Visual Basic (Windows 95)

W_9118_DI (ByVal cardNo As Integer, data As Long) As Long

@ Argument

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

data: the 4-bit data read from digital port.

@ Return Code

ERR_NoError
ERR_BoardNoInit

6.2.5 _9118_DI_Channel

@ Description

This function is used to read data from digital input ports. There are 4 digital input channels on PCI_9118. When performing this function, the digital input port is read and the value of the corresponding channel is returned.

@ **Syntax**

C/C++ (DOS)

int _9118_DI_Channel (I16 di_ch_no , unsigned int far *data)

C/C++ (Windows 95)

int W_9118_DI_Channel(U16 cardNo, I16 di_ch_no , U16 *data)

Visual Basic (Windows 95)

W_9118_DI_Channel (ByVal cardNo As Integer, ByVal di_ch_no As Long, data As Long) As Long

@ **Argument**

cardNo (Win-95):	the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.
di_ch_no :	the DI channel number, the valid channel value is from 0 to 3
data :	the returned data, either 0 or 1.

@ **Return Code**

ERR_NoError
ERR_BoardNoInit
ERR_InvalidDIChannel

6.2.6 _9118_DO

@ **Description**

This function is used to write data to digital output port. There are 4 output channels on the PCI-9118.

@ **Syntax**

C/C++ (DOS)

int _9118_DO (unsigned int data)

C/C++ (Windows 95)

int W_9118_DO (U16 cardNo, U16 data)

Visual Basic (Windows 95)

W_9118_DO (ByVal cardNo As Integer, ByVal data As Long)

As Long

@ **Argument**

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

data: the data written to output port.

@ **Return Code**

ERR_NoError
ERR_BoardNoInit

6.2.7 _9118_DA

@ **Description**

This function is used to write data to D/A converters. There are two Digital-to-Analog conversion channels on the PCI-9118. The resolution of each channel is 12-bit, i.e. the D/A data range is from 0 to 4095.

@ **Syntax**

C/C++ (DOS)

int _9118_DA(int ch_no, unsigned int data)

C/C++ (Windows 95)

int W_9118_DA(U16 cardNo, int ch_no, unsigned int data)

Visual Basic (Windows 95)

W_9118_DA (ByVal cardNo As Integer, ByVal ch_no As Long, ByVal data As Long) As Long

@ **Argument**

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

ch_no : D/A channel number, DA_CH_1 or DA_CH_2.

Data : D/A converted value, if the value is greater than 4095, the higher bits are negligent.

@ Return Code

ERR_NoError
ERR_InvalidDAChannel
ERR_BoardNoInit

6.2.8 _9118_AD_Reset_AFIFO

@ Description

This function is used to reset A/D Channel/Gain Register. Before calling _9118_AD_Set_GainChn (refer to section 6.2.14) to set A/D channel and input range, you have to perform this function to clear A/D Channel/Gain Register.

@ Syntax

C/C++ (DOS)

```
int _9118_AD_Reset_AFIFO()
```

C/C++ (Windows 95)

```
int W_9118_AD_Reset_AFIFO(U16 cardNo)
```

Visual Basic (Windows 95)

```
W_9118_AD_Reset_AFIFO (ByVal cardNo As Integer) As
      Long
```

@ Argument

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

@ Return Code

ERR_NoError
ERR_BoardNoInit

6.2.9 _9118_AD_Bgnset_AFIFO

@ Description

Users can not set the A/D channel and input range unless this function is executed. Therefore after this function is performed, the program can start to fill out A/D Channel/Gain register to set the A/D channel and range.

@ Syntax

C/C++ (DOS)

```
int _9118_AD_Bgnset_AFIFO()
```

C/C++ (Windows 95)

```
int W_9118_AD_Bgnset_AFIFO(U16 cardNo)
```

Visual Basic (Windows 95)

```
W_9118_AD_Bgnset_AFIFO (ByVal cardNo As Integer) As
    Long
```

@ Argument

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

@ Return Code

ERR_NoError
ERR_BoardNoInit

6.2.10 _9118_AD_Endset_AFIFO

@ Description

This function is used to stop setting A/D channel and input range.

@ Syntax

C/C++ (DOS)

```
int _9118_AD_Endset_AFIFO()
```

C/C++ (Windows 95)

```
int W_9118_AD_Endset_AFIFO(U16 cardNo)
```

Visual Basic (Windows 95)

W_9118_AD_Endset_AFIFO (ByVal cardNo As Integer) As Long

@ **Argument**

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

@ **Return Code**

ERR_NoError
ERR_BoardNoInit

6.2.11 _9118_AD_Reset_DFIFO

@ **Description**

This function is used to reset A/D Data FIFO.

@ **Syntax**

C/C++ (DOS)

int _9118_AD_Reset_DFIFO()

C/C++ (DOS)

int W_9118_AD_Reset_DFIFO(U16 cardNo)

Visual Basic (Windows 95)

W_9118_AD_Reset_DFIFO (ByVal cardNo As Integer) As Long

@ **Argument**

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

@ **Return Code**

ERR_NoError
ERR_BoardNoInit

6.2.12 _9118_AD_Set_Burst_No

@ Description

If the A/D conversion mode is set as burst mode, this function is used to set the number of conversion channels in a scan trigger.

@ Syntax

C/C++ (DOS)

int _9118_AD_Set_Burst_No(int BurstNo)

C/C++ (Windows 95)

int W_9118_AD_Set_Burst_No(U16 cardNo, int BurstNo)

Visual Basic (Windows 95)

W_9118_AD_Set_Burst_No (ByVal cardNo As Long, ByVal BurstNo As Integer) As Long

@ Argument

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

BurstNo: the burst number.

@ Return Code

ERR_NoError
ERR_BoardNoInit

6.2.13 _9118_INT_Set_CtrlReg

@ Description

This function is used to set Interrupt Control Register. The definition of each bit of ' CtrlVal' is as follows:

Bit	7	6	5	4	3	2	1	0
	--	--	--	--	Timer	About	Hfull	DTrg

@ Syntax

C/C++ (DOS)

int _9118_INT_Set_CtrlReg (int CtrlVal)

C/C++ (Windows 95)

int W_9118_INT_Set_CtrlReg(U16 cardNo, int CtrlVal)

Visual Basic (Windows 95)

W_9118_INT_Set_CtrlReg (ByVal cardNo As Long, ByVal CtrlVal As Integer) As Long

@ Argument

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

CtrlVal: the value written to *Interrupt Control Register*.

@ Return Code

ERR_NoError
ERR_BoardNoInit

6.2.14 _9118_AD_Set_GainChn

@ Description

This function is used to specify the A/D channel and input range.

Note: _9118_AD_Bgnset_AFIFO should be called before you use this function to set the channels and gains in the Channel-Gain Queue. After channel setting is finished, you have to call _9118_AD_Endset_AFIFO to stop filling out the Channel-Gain Queue. The sequence to call these three functions is:

```
_9118_AD_Bgnset_AFIFO();
_9118_AD_Set_GainChn(ch, gain);  ┐
                :                │
                :                │ Setting A/D channels
                :                │ and gains
_9118_AD_Set_GainChn(ch, gain);  ┘
_9118_AD_Endset_AFIFO();
```

@ Syntax

C/C++ (DOS)

int _9118_AD_Set_GainChn (int ch, int gain)

C/C++ (Windows 95)

int W_9118_AD_Set_GainChn(U16 cardNo, int ch,int gain)

Visual Basic (Windows 95)

W_9118_AD_Set_GainChn (ByVal cardNo As Integer, ByVal ch As Long, ByVal gain As Long) As Long

@ Argument

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

ch : the channel on which perform. operation, the valid value is within 0 to 15.

gain : the A/D input range, the possible values are listed in the column 'AD_INPUT' of the following table.

9118DG/HR

AD_INPUT	GAIN	Input type (Bipolar or Unipolar)	Input Range
AD_B_5_V	1	Bipolar	±5V
AD_B_2_5_V	2	Bipolar	±2.5V
AD_B_1_25_V	4	Bipolar	±1.25V
AD_B_0_625_V	8	Bipolar	±0.625V
AD_U_10_V	1	Unipolar	0V ~ 10V
AD_U_5_V	2	Unipolar	0V ~ 5V
AD_U_2_5_V	4	Unipolar	0V ~ 2.5V
AD_U_1_25_V	8	Unipolar	0V ~ 1.25V

9118HG

AD_INPUT	GAIN	Input type (Bipolar or Unipolar)	Input Range
AD_B_5_V	1	Bipolar	±5V
AD_B_0_5_V	10	Bipolar	±0.5V
AD_B_0_05_V	100	Bipolar	±0.05V
AD_B_0_005_V	1000	Bipolar	±0.005V
AD_U_10_V	1	Unipolar	0V ~ 10V
AD_U_1_V	10	Unipolar	0V ~ 1V
AD_U_0_1_V	100	Unipolar	0V ~ 0.1V
AD_U_0_01_V	1000	Unipolar	0V ~ 0.01V

@ Return Code

ERR_NoError
ERR_InvalidADChannel
ERR_BoardNoInit
ERR_InvalidADGain

6.2.15 _9118_AD_Set_Scan

@ Description

This function is used to set the first and last channels in a group of consecutive channels. This function can specify a group of consecutive channels for analog input through DMA and interrupt operation. The sequence of channel scanned is start, start+1, start+2, .., end.

@ Syntax

C/C++ (DOS)

Int _9118_AD_Set_Scan(int start, int end, int gain)

C/C++ (Windows 95)

Int W_9118_AD_Set_Scan(U16 cardNo, int start, int end, int gain)

Visual Basic (Windows 95)

W_9118_AD_Set_Scan (ByVal cardNo As Integer, ByVal start As Long, ByVal end As Long, ByVal gain As Long) As Long

@ Argument

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

ch : the channel on which perform

start : the First channel number in a group of consecutive channels, the valid value is 0 to 15.

end : the last channel number in a group of consecutive channels, the valid value is 0 to 15.

gain : the A/D input range, the possible values are listed in the column 'AD_INPUT' of the following table.

9118DG/HR

AD_INPUT	GAIN	Input type (Bipolar or Unipolar)	Input Range
AD_B_5_V	1	Bipolar	±5V
AD_B_2_5_V	2	Bipolar	±2.5V
AD_B_1_25_V	4	Bipolar	±1.25V
AD_B_0_625_V	8	Bipolar	±0.625V
AD_U_10_V	1	Unipolar	0V ~ 10V
AD_U_5_V	2	Unipolar	0V ~ 5V
AD_U_2_5_V	4	Unipolar	0V ~ 2.5V
AD_U_1_25_V	8	Unipolar	0V ~ 1.25V

9118HG

AD_INPUT	GAIN	Input type (Bipolar or Unipolar)	Input Range
AD_B_5_V	1	Bipolar	±5V
AD_B_0_5_V	10	Bipolar	±0.5V
AD_B_0_05_V	100	Bipolar	±0.05V
AD_B_0_005_V	1000	Bipolar	±0.005V
AD_U_10_V	1	Unipolar	0V ~ 10V
AD_U_1_V	10	Unipolar	0V ~ 1V
AD_U_0_1_V	100	Unipolar	0V ~ 0.1V
AD_U_0_01_V	1000	Unipolar	0V ~ 0.01V

@ **Return Code**

ERR_NoError
ERR_AD_InvalidChannel
ERR_BoardNoInit
ERR_InvalidADGain

6.2.16 _9118_AD_Soft_Trig

@ **Description**

This function is used to trigger the A/D conversion by software. This function generates a trigger pulse to PCI-9118 for A/D conversion and the converted data will be stored in A/D data register.

@ **Syntax**

C/C++ (DOS)
int _9118_AD_Soft_Trig ()

C/C++ (Windows 95)
int W_9118_AD_Soft_Trig (U16 cardNo)

Visual Basic (Windows 95)
W_9118_AD_Soft_Trig (ByVal cardNo As Integer) As Long

@ **Argument**

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

@ **Return Code**

ERR_NoError
ERR_BoardNoInit

6.2.17 _9118_AD_Set_Unip

@ **Description**

This function is used to set analog input range as uni-polar or bi-polar. The default setting is bi-polar.

@ **Syntax**

C/C++ (DOS)

int _9118_AD_Set_Unip (int Unip)

C/C++ (Windows 95)

int W_9118_AD_Set_Unip(U16 cardNo, int Unip)

Visual Basic (Windows 95)

W_9118_AD_Set_Unip (ByVal cardNo As Integer, ByVal Unip As Long) As Long

@ **Argument**

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

Unip: 0: bi-polar
1: uni-polar

@ **Return Code**

ERR_NoError
ERR_BoardNoInit

6.2.18 _9118_AD_Set_Diff

@ **Description**

This function is used to set analog input mode as differential or single-ended.

@ **Syntax**

C/C++ (DOS)

int _9118_AD_Set_Diff (int Diff)

C/C++ (Windows 95)

int W_9118_AD_Set_Diff (int Diff)

Visual Basic (Windows 95)

W_9118_AD_Set_Diff (ByVal cardNo As Integer, ByVal Diff As Long) As Long

@ **Argument**

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

Diff: 0: single-ended
1: differential

@ **Return Code**

ERR_NoError
ERR_BoardNoInit

6.2.19 _9118_AD_Set_SoftG

@ **Description**

This function is used to specify the status of software Gate, i.e. it controls 8254 counter to work or stop.

@ **Syntax**

C/C++ (DOS)

```
int _9118_AD_Set_SoftG (int SoftG)
```

C/C++ (Windows 95)

```
int W_9118_AD_Set_SoftG(U16 cardNo, int SoftG)
```

Visual Basic (Windows 95)

```
W_9118_AD_Set_SoftG (ByVal cardNo As Integer, ByVal
    SoftG As Long) As Long
```

@ **Argument**

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

SoftG: 1: 8254 counter works
0: 8254 counter stops

@ **Return Code**

ERR_NoError
ERR_BoardNoInit

6.2.20 _9118_AD_Set_ExtG

@ Description

This function is used to specify the A/D control as External Gate control mode or Software Gate Control mode.

@ Syntax

C/C++ (DOS)

```
int _9118_AD_Set_ExtG (int ExtG)
```

C/C++ (Windows 95)

```
int W_9118_AD_Set_ExtG(U16 cardNo, int ExtG)
```

Visual Basic (Windows 95)

```
W_9118_AD_Set_ExtG (ByVal cardNo As Integer, ByVal ExtG
      As Long) As Long
```

@ Argument

cardNo (Win-95):	the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.
ExtG:	1: 8254 counter controlled by TGIN (connector pin 46) 0: 8254 counter controlled by Soft Gate

@ Return Code

ERR_NoError
ERR_BoardNoInit

6.2.21 _9118_AD_Set_ExtM

@ Description

This function is used to specify the hardware trigger source.

@ **Syntax**

C/C++ (DOS)
int _9118_AD_Set_ExtM (int ExtM)

C/C++ (Windows 95)
int W_9118_AD_Set_ExtM(U16 cardNo,int ExtM)

Visual Basic (Windows 95)
W_9118_AD_Set_ExtM (ByVal cardNo As Integer, ByVal ExtM As Long) As Long

@ **Argument**

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

ExtM: 1: External Hardware Trigger (connector pin 44)
0: Internal Hardware Trigger

@ **Return Code**

ERR_NoError
ERR_BoardNoInit

6.2.22 _9118_AD_Set_TmrTr

@ **Description**

This function is used to specify the internal trigger source.

@ **Syntax**

C/C++ (DOS)
int _9118_AD_Set_TmrTr (int TmrTr)

C/C++ (Windows 95)
int W_9118_AD_Set_TmrTr (U16 cardNo, int TmrTr)

Visual Basic (Windows 95)
W_9118_AD_Set_TmrTr (ByVal cardNo As Integer, ByVal TmrTr As Long) As Long

@ **Argument**

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

TmrTr: 1: 8254 Timer (Counter) is internal trigger source.
0: Software Trigger is internal trigger source.

@ Return Code

ERR_NoError
ERR_BoardNoInit

6.2.23 _9118_AD_Set_Int

@ Description

This function is used to specify the status of interrupt operation. To enable hardware interrupt, 'Int' should be set as 1; otherwise the interrupt operation can not be performed.

@ Syntax

C/C++ (DOS)
int _9118_AD_Set_Int (int Int)

C/C++ (Windows 95)
int W_9118_AD_Set_Int(U16 cardNo, int Int)

Visual Basic (Windows 95)
W_9118_AD_Set_Int (ByVal cardNo As Integer, ByVal Int As Long) As Long

@ Argument

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

Int: 1: Enable Hardware interrupt.
0: Disable Hardware interrupt.

@ Return Code

ERR_NoError
ERR_BoardNoInit

6.2.24 _9118_AD_Set_Dma

@ Description

This function is used to specify the status of DMA transfer mode. To enable DMA Transfer opreration, 'Dma' should be set as 1; otherwise the DMA operation will not be performed.

@ Syntax

C/C++ (DOS)
int _9118_AD_Set_Dma (int Dma)

C/C++ (Windows 95)
int W_9118_AD_Set_Dma(U16 cardNo, int Dma)

Visual Basic (Windows 95)
W_9118_AD_Set_Dma (ByVal cardNo As Integer, ByVal Dma As Long) As Long

@ Argument

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

Dma: 1: Enable A/D Data DMA Transfer Mode.
0: Disable A/D Data DMA Transfer Mode.

@ Return Code

ERR_NoError
ERR_BoardNoInit

6.2.25 _9118_AD_Set_CtrlReg

@ Description

This function is used to set the Analog output mode by writing 8 bits data into A/D control register. This function can be used to reset A/D Control Register to default value by setting 'CtrlVal' as 0,

i.e. the A/D mode is set as bi-polar, single-ended and software trigger). The definition of each bit of 'CtrlVal' is as follows:

Bit	7	6	5	4	3	2	1	0
	UniP	Diff	SoftG	ExtG	ExtM	TmrTr	Int	Dma

@ **Syntax**

C/C++ (DOS)
int _9118_AD_Set_CtrlReg(int CtrlVal)

C/C++ (Windows 95)
int W_9118_AD_Set_CtrlReg(U16 cardNo, int CtrlVal)

Visual Basic (Windows 95)
W_9118_AD_Set_CtrlReg (ByVal cardNo As Integer, ByVal CtrlVal As Long) As Long

@ **Argument**

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

CtrlVal: the value written to A/D control register

@ **Return Code**

ERR_NoError
ERR_BoardNoInit

6.2.26 _9118_AD_Set_PDTrg

@ **Description**

This function is used to set the active type of digital trigger.

@ **Syntax**

C/C++ (DOS)
int _9118_AD_Set_PDTrg(int PDTrg)

C/C++ (Windows 95)
int W_9118_AD_Set_PDTrg(U16 cardNo,int PDTrg)

Visual Basic (Windows 95)
W_9118_AD_Set_PDTrg (ByVal cardNo As Integer, ByVal

PDTrg As Long) As Long

@ Argument

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

PDTrg: 0: Negative Trigger
1: Positive Trigger

@ Return Code

ERR_NoError
ERR_BoardNoInit

6.2.27 _9118_AD_Set_PETrg

@ Description

This function is used to set the active type of external trigger.

@ Syntax

C/C++ (DOS)
int _9118_AD_Set_PETrg(int PETrg)

C/C++ (Windows 95)
int W_9118_AD_Set_PETrg(U16 cardNo, int PETrg)

Visual Basic (Windows 95)
W_9118_AD_Set_PETrg (ByVal cardNo As Integer, ByVal PETrg As Long) As Long

@ Argument

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

PETrg: 0: Negative Trigger
1: Positive Trigger

@ Return Code

ERR_NoError

ERR_BoardNoInit

6.2.28 _9118_AD_Set_BSSH

@ Description

This function is used to enable and disable A/D Burst Mode with Sample and Hold.

@ Syntax

C/C++ (DOS)

int _9118_AD_Set_BSSH (int BSSH)

C/C++ (Windows 95)

int W_9118_AD_Set_BSSH(U16 cardNo, int BSSH)

Visual Basic (Windows 95)

W_9118_AD_Set_BSSH (ByVal cardNo As Integer, ByVal BSSH As Long) As Long

@ Argument

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

BSSH: 0: without Sample and Hold
1: with Sample and Hold

@ Return Code

ERR_NoError
ERR_BoardNoInit

6.2.29 _9118_AD_Set_BM

@ Description

This function is used to enable and disable A/D Burst Mode.

@ **Syntax**

C/C++ (DOS)

int _9118_AD_Set_BM (int BM)

C/C++ (Win95)

int W_9118_AD_Set_BM(U16 cardNo, int BM)

Visual Basic (Windows 95)

W_9118_AD_Set_BM (ByVal cardNo As Integer, ByVal BM As Long) As Long

@ **Argument**

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

BM: 0: Normal Mode
1: Burst Mode

@ **Return Code**

ERR_NoError
ERR_BoardNoInit

6.2.30 _9118_AD_Set_BS

@ **Description**

This function is used to start and stop A/D Burst Mode.

@ **Syntax**

C/C++ (DOS)

int _9118_AD_Set_BS (int BS)

C/C++ (Windows 95)

int W_9118_AD_Set_BS(U16 cardNo, int BS)

Visual Basic (Windows 95)

W_9118_AD_Set_BS (ByVal cardNo As Integer, ByVal BS As Long) As Long

@ **Argument**

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

BS: 0: stops Burst Mode
1: starts Burst Mode

@ **Return Code**

ERR_NoError
ERR_BoardNoInit

6.2.31 _9118_AD_Set_PM

@ **Description**

This function is used to enable and disable Post Trigger Mode.

@ **Syntax**

C/C++ (DOS)

```
int _9118_AD_Set_PM (int PM)
```

C/C++ (Windows 95)

```
int W_9118_AD_Set_PM(U16 cardNo, int PM)
```

Visual Basic (Windows 95)

```
W_9118_AD_Set_PM (ByVal cardNo As Integer, ByVal PM As
    Long) As Long
```

@ **Argument**

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

PM: 0: Disable Post Trigger
1: Enable Post Trigger

@ **Return Code**

ERR_NoError
ERR_BoardNoInit

6.2.32 _9118_AD_Set_AM

@ Description

This function is used to enable and disable About Trigger Mode.

@ Syntax

C/C++ (DOS)

int _9118_AD_Set_AM (int AM)

C/C++ (Windows 95)

int W_9118_AD_Set_AM(U16 cardNo, int AM)

Visual Basic (Windows 95)

W_9118_AD_Set_AM (ByVal cardNo As Integer, ByVal AM As Long) As Long

@ Argument

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

AM: 0: Disable About Trigger
1: Enable About Trigger

@ Return Code

ERR_NoError
ERR_BoardNoInit

6.2.33 _9118_AD_Set_Start

@ Description

This function is used to start or stop Trigger. After the trigger mode (post-trigger or about trigger) is selected, the program should call this function to start trigger.

@ Syntax

C/C++ (DOS)

int _9118_AD_Set_Start (int Start)

C/C++ (Windows 95)

int W_9118_AD_Set_Start(U16 cardNo, int Start)

Visual Basic (Windows 95)

W_9118_AD_Set_Start (ByVal cardNo As Integer, ByVal Start As Long) As Long

@ Argument

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

Start: 0: Trigger Stop
1: Trigger Start

@ Return Code

ERR_NoError
ERR_BoardNoInit

6.2.34 _9118_AD_Set_FuncVal

@ Description

This function is used to set A/D trigger mode by writing data into A/D Function Register. The definition of each bit is as follows:

Bit	7	6	5	4	3	2	1	0
	PDTrg	PETrg	BSSH	BM	BS	PM	AM	Start

@ Syntax

C/C++ (DOS)

int _9118_AD_Set_FuncReg (int FuncVal)

C/C++ (Windows 95)

int W_9118_AD_Set_FuncReg(U16 cardNo, int FuncVal)

Visual Basic (Windows 95)

W_9118_AD_Set_ FuncReg (ByVal cardNo As Integer, ByVal FuncVal As Long) As Long

@ Argument

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid

card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

FuncVal: the value written to A/D Function Register.

@ **Return Code**

ERR_NoError
ERR_BoardNoInit

6.2.35 _9118_AD_Aquire

@ **Description**

This function is used to poll the A/D conversion data. It will trigger the A/D conversion, and read the A/D data when the data is ready ('data ready' bit becomes low). The value of converted A/D data is from 0 to 4095 (for 9118HG/DG) or -32768 to 32767(for 9118HR).

@ **Syntax**

C/C++ (DOS)

```
int _9118_AD_Aquire ( int far *ad_data )
```

C/C++ (Windows 95)

```
int W_9118_AD_Aquire(U16 cardNo, I16 *ad_data )
```

Visual Basic (Windows 95)

```
W_9118_AD_Aquire (ByVal cardNo As Integer, ad_data As
    Integer) As Long
```

@ **Argument**

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

ad_data : A/D converted value for PCI-9118DG/HG, the lowest 4 bits represents the converted channel number and the higher 12 bits is the converted A/D data.

Bit 0 ~ Bit 3: the converted channel number
Bit 4 ~ Bit 15: the converted A/D data

For PCI-9118HR, all the 16 bits are the converted A/D data.

@ Return Code

ERR_NoError
ERR_BoardNoInit
ERR_AD_AquireTimeOut

6.2.36 _9118_AD_DMA_Start

@ Description

This function will perform A/D conversion N times with DMA data transfer by using the pacer trigger (internal timer trigger). It takes place in the background which will not stop until the N-th conversion has been completed or your program execute _9118_AD_DMA_Stop() function to stop the process. The function is performed on single A/D channel when the A/D channel auto-scan is set as FALSE. If the A/D channel auto-scan is TRUE, the conversion will be multiple channels by sequence.

After executing this function, it is necessary to check the status of the operation by using the function _9118_AD_DMA_Status(). The value of converted A/D data is from 0 to 4095 (9118DG/HG) or from -32768 to 32767
(9118HR).

@ Syntax

C/C++ (DOS)

```
int _9118_AD_DMA_Start(int clk_src, unsigned int count ,
        unsigned long *ad_buffer , int c1 , int c2 )
```

C/C++ (Windows 95)

```
int W_9118_AD_DMA_Start(U16 cardNo, int clk_src,
        unsigned int count , HANDLE memID, int c1, int c2 )
```

Visual Basic (Windows 95)

```
W_9118_AD_DMA_Start (ByVal cardNo As Integer, ByVal
        clk_src As Long, ByVal count As Long, ByVal handle
        As Long, ByVal c1 As Long, ByVal c2 As Long) As
        Long
```

@ Argument

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

clk_src : the clock source for the timer trigger of AD conversion. The valid clock sources are: *A_9118_AD_IntSrc*, *A_9118_AD_ExtSrc*

count : the number of A/D conversion in samples.

ad_buffer (DOS): the start address of the memory buffer to store the A/D data, the buffer size must large than the numbers of A/D conversion. This memory should be double-word alignment.

ad_buffer is a 32-bit data buffer, but the A/D data is 16-bit format. For PCI-9118, the data format of ad_buffer is as follows:

DATA1	DATA0	DATA3	DATA2	.....	DATAN	DATAN-1

Every 16-bit data:

D11 D10 D9 ..... D1 D0 C3 C2 C1 C0

Where D11, D10, ... , D0: A/D converted data
C3, C2, C1, C0: converted channel no (for 9118DG/HG)
D15, D14, .., D0: A/D converted data (for 9118HR)

Please refer to the sample program, ad_demo3.c, included in the software library CD we provide to get the idea how to get each data sampled (16-bit).

memID (Win-95): the memory ID of the allocated system AI memory. In Windows 95 environment, before calling W_9118_AD_DMA_Start, W_9118_Alloc_AI_Mem must be called to allocate a contiguous AI memory. W_9118_Alloc_AI_Mem will return a memory ID for identify the allocated AI memory, as well

as the linear address of the AI memory for user to access the data. The format of the A/D data is the same as DOS buffer (*ad_buff* argument).

Note: After the DMA operation stops, the data stored in ad_buffer or stored in the memory space allocated by memID are the last *count* transferred data, where *count* is the value of argument *count*. Thus, for post trigger mode and about trigger mode, the value of count should be larger than the value of *post_trig_cnt* set in W_9118_AD_Trig to get the all the data transferred during Timer#0 counts down.

c1 :	the 16-bit timer frequency divider of timer channel #1
c2 :	the 16-bit timer frequency divider of timer channel #2

Note : the A/D sampling rate is equal to ***4MHz / (c1 * c2)***

@ Return Code

ERR_NoError
ERR_BoardNoInit,
ERR_InvalidADChannel,
ERR_InvalidTimerValue
ERR_AD_InvalidGain

6.2.37 _9118_AD_DMA_Status

@ Description

Since the _9118_AD_DMA_Start() function executes in background, you can issue this function to check its operation status.

@ Syntax

C/C++ (DOS)

```
int _9118_AD_DMA_Status ( int *status , int *count )
```

C/C++ (Windows 95)

```
int W_9118_AD_DMA_Status(U16 cardNo, int *status , int
        *count )
```

Visual Basic (Windows 95)

```
W_9118_AD_DMA_Status (ByVal cardNo As Integer, status
    As Long, count As Long) As Long
```

@ Argument

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

status : status of the DMA data transfer
0 : AD_DMA_STOP : DMA is completed
1 : AD_DMA_RUN : DMA is not completed

count : the number of A/D data which has been transferred.

@ Return Code

ERR_NoError
ERR_AD_DMANotSet
ERR_BoardNoInit

6.2.38 _9118_AD_DMA_Stop

@ Description

This function is used to stop the DMA data transferring. After executing this function, the internal A/D trigger is disable and the A/D timer (timer #1 and #2) is stopped. The function returns the number of the data which has been transferred, no matter if the A/D DMA data transfer is stopped by this function or by the DMA terminal count ISR.

@ Syntax

C/C++ (DOS)

```
int _9118_AD_DMA_Stop( int *count, int *start_idx)
```

C/C++ (Windows 95)

```
int W_9118_AD_DMA_Stop(U16 cardNo, int *count, int
    *start_idx )
```

Visual Basic (Windows 95)

W_9118_AD_DMA_Stop (ByVal cardNo As Integer, count As Long, start_idx As Long) As Long

@ Argument

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

count : the number of A/D converted data which has been transferred.

start_idx: The index where the data start from in user' s buffer, i.e the sequence of read data is: buff[start_idx], buff[start_idx+1], _, buff[0], buff[1],_.,buff[start_idx-1].

@ Return Code

ERR_NoError
ERR_BoardNoInit

6.2.39 _9118_ContDmaStart

@ Description

This function performs continuous A/D conversion with double-buffered DMA data transfer and the pacer trigger (internal timer trigger). It takes place in the background which will not stop until your program execute _9118_ContDmaStop() function to stop the process.

After executing this function, it is necessary to check the status of circular buffer by using the function _9118_CheckHalfReady() and using _9118_DblBufferTransfer() to copy the A/D converted data to transfer buffer.

In current version of software, double-buffered DMA data transfer does not support trigger acquisition mode (pre-trigger, post-trigger, or about-trigger).

There is a group function for continuous A/D conversion using double-buffered DMA transfer as following:

```
_9118_ContDmaStart();
_9118_CheckHalfReady();
_9118_DblBufferTransfer();
_9118_GetOverrunStatus();
_9118_ContDmaStop();
```

The value of converted A/D data is from 0 to 4095 (9118DG/HG) or from 0 to 65535 (9118HR).

@ Syntax

C/C++ (DOS)

```
_9118_ContDmaStart(int clk_src, unsigned int count , unsigned
        long *doubleBuf, int c1 , int c2 )
```

C/C++ (Windows 95)

```
W_9118_ContDmaStart(U16 cardNo, int clk_src, unsigned int
        count, HANDLE memID, int c1, int c2 )
```

Visual Basic (Windows 95)

```
W_9118_ContDmaStart (ByVal cardNo As Integer, ByVal
        clk_src, ByVal count As Long, ByVal handle As Long,
        ByVal c1 As Long, ByVal c2 As Long) As Long
```

@ Argument

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

clk_src : the clock source for the timer trigger of AD conversion. The valid clock sources are: *A_9118_AD_IntSrc*, *A_9118_AD_ExtSrc*

count : the size of circular buffer in samples

db_buffer (DOS) : the start address of the memory of the circular buffer to store the A/D data, the buffer size must large than *count*. This memory should be double-word alignment, the ad_buff format is as following: ad_buffer is a 32-bit data buffer, but the A/D data is 16-bit format. For PCI-9118,

the data format of ad_buffer is as follows:

DATA1	DATA0	DATA3	DATA2	……	DATAN	DATAN-1

Every 16-bit data:

D11 D10 D9 ..... D1 D0 C3 C2 C1 C0

Where D11, D10, ... , D0 : A/D converted data
C3, C2, C1, C0 : converted channel no (for 9118DG/HG)
D15, D14, .., D0: A/D converted data (for 9118HR)

Please refer to the sample program, ad_demo3.c, included in the software library CD we provide to get the idea how to get each data sampled (16-bit).

memID (Win-95): the memory ID of the allocated system AI memory. In Windows 95 environment, before calling W_9118_ContDMAStart, W_9118_Alloc_AI_Mem must be called to allocate a contiguous AI memory. W_9118_Alloc_AI_Mem will return a memory ID for identify the allocated AI memory, as well as the linear address of the AI memory for user to access the data. The format of the A/D data is the same as DOS buffer (*ad_buff* argument).

c1 : the 16-bit timer frequency divider of timer channel #1

c2 : the 16-bit timer frequency divider of timer channel #2

Note : the A/D sampling rate is equal to ***4MHz / (c1 * c2)***

@ Return Code

ERR_NoError
ERR_BoardNoInit,
ERR_InvalidADChannel,
ERR_AD_InvalidGain,

ERR_InvalidTimerValue

6.2.40 _9118_CheckHalfReady

@ Description

When you use _9118_ContDmaStart() to convert A/D data then you must use _9118_CheckHalfReady() to check whether data half full or not in circular buffer, and then can use _9118_DblBufferTransfer() to copy data to transfer buffer.

@ Syntax

C/C++ (DOS)

int _9118_CheckHalfReady(int *halfReady, int *stop_flag)

C/C++ (Windows 95)

int W_9118_CheckHalfReady(U16 cardNo, int *halfReady, int *stop_flag)

Visual Basic (Windows 95)

W_9118_CheckHalfReady (ByVal cardNo As Integer, halfReady As Long, stop_flag As Long) As Long

@ Argument

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

halfReady : 1 (TRUE) or 0 (FALSE).

stop_flag : Whether the AD conversion is stopped.
0: not stopped
1: stopped

@ Return Code

ERR_NoError

6.2.41 _9118_DblBufferTransfer

@ Description

Using this function to copy the converted A/D data from circular to transfer buffer. The value of converted A/D data is from 0 to 4095 (9118DG/HG) or from 0 to 65535 (9118HR).

@ **Syntax**

C/C++ (DOS)

```
int _9118_DblBufferTransfer(unsigned short *userBuffer)
```

C/C++ (Windows 95)

```
int W_9118_DblBufferTransfer(U16 cardNo, unsigned short
    *userBuffer)
```

Visual Basic (Windows 95)

```
W_9118_DblBufferTransfer (ByVal cardNo As Integer,
    userBuffer As Integer) As Long
```

@ **Argument**

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

userBuffer : user transfer buffer for A/D converted data, every time _9118_DblBufferTransfer() copies half size of circular buffer to userBuffer. The size of circular in samples is specified in _9118_ContDmaStart() function call.

@ **Return Code**

ERR_NoError
ERR_BoardNoInit

6.2.42 _9118_GetOverrunStatus

@ **Description**

When you use _9118_ContDmaStart() to convert A/D data and if you do not use _9118_DblBufferTransfer() to copy converted data then the circular buffer overrun will occur. You can use this function to check overrun count.

@ **Syntax**

C/C++ (DOS)

int _9118_GetOverrunStatus(int *overrunCount)

C/C++ (Windows 95)

int W_9118_GetOverrunStatus(U16 cardNo, int *overrunCount)

Visual Basic (Windows 95)

W_9118_GetOverrunStatus (ByVal cardNo As Integer, overrunCount As Long) As Long

@ **Argument**

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

overrunCount: number of overrun counts.

@ **Return Code**

ERR_NoError

6.2.43 _9118_ContDmaStop

@ **Description**

This function is used to stop the continuous double-buffered DMA data transfer.

@ **Syntax**

C/C++ (DOS)

int _9118_ContDmaStop(unsigned int *count)

C/C++ (DOS)

Int W_9118_ContDmaStop(U16 cardNo, unsigned int *count)

Visual Basic (Windows 95)

W_9118_ContDmaStop (ByVal cardNo As Integer, count As Long) As Long

@ **Argument**

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid

card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

count : the next position after the position the last A/D data is stored in the circular buffer.

@ Return Code

ERR_NoError
ERR_BoardNoInit
ERR_AD_DMANotSet

6.2.44 _9118_AD_INT_Start

@ Description

This function performs A/D conversion N times with interrupt data transfer by using pacer trigger. It takes place in the background which will not stop until the N-th conversion has been completed or your program execute _9118_AD_INT_Stop() function to stop the process. After executing this function, it is necessary to check the status of the operation by using the function 9118_AD_INT_Status(). The value of A/D converted data is from 0 to 4095 (9118DG/HG) or from 0 to 65535 (9118HR).

@ Syntax

C/C++ (DOS)

```
int _9118_AD_INT_Start( int count , unsigned long *ad_buffer,
    int c1, int c2 )
```

C/C++ (Windows 95)

```
int W_9118_AD_INT_Start(U16 cardNo, int count , HANDLE
    memID, int c1, int c2 )
```

Visual Basic (Windows 95)

```
W_9118_AD_INT_Start (ByVal cardNo As Integer, ByVal count
    As Long, ByVal handle As Long, ByVal c1 As Long, ByVal
    c2 As Long) As Long
```

@ Argument

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card

numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

count : the number of A/D conversion in samples

ad_buffer (DOS): the start address of the memory buffer to store the A/D data, the buffer size must large than the number of A/D conversion.
This memory should be double-word alignment,

The ad_buff format is as following: (Note: the ad_buffer format is not the same as that in 6.2.36. Please refer to sample program ad_demo 2.c)

DATA 1	DATA 2	DATA 3	.....	DATA N
16-bit	16-bit	16-bit		16-bit

Every 16-bit data:

D11 D10 D9 ..... D1 D0 C3 C2 C1 C0

where D11, D10, ... , D0 : A/D converted data
C3, C2, C1, C0 : converted channel no (9118DG/HG)
D15, D14, .., D0: A/D converted data (9118HR).

memID (Win-95): the memory ID of the allocated system AI memory. In Windows 95 environment, before calling W_9118_AD_INT_Start, W_9118_Alloc_AI_Mem must be called to allocate a contiguous AI memory. W_9118_Alloc_AI_Mem will return a memory ID for identify the allocated AI memory, as well as the linear address of the AI memory for user to access the data. The format of the A/D data is the same as DOS buffer (*ad_buff* argument).

c1 : the 16-bit timer frequency divider of timer channel #1

c2 : the 16-bit timer frequency divider of timer channel #2

Note : the A/D sampling rate is equal to ***4MHz / (c1 * c2)***

@ **Return Code**

ERR_NoError
ERR_BoardNoInit
ERR_InvalidTimerValue

6.2.45 _9118_AD_INT_Status

@ **Description**

Since the _9118_AD_INT_Start() function executes in background, you can issue the function _9118_AD_INT_Status() to check the status of interrupt operation.

@ **Syntax**

C/C++ (DOS)

int _9118_AD_INT_Status(int *status, int *count)

C/C++ (Windows 95)

int W_9118_AD_INT_Status(U16 cardNo, int *status, int *count)

Visual Basic (Windows 95)

W_9118_AD_INT_Status (ByVal cardNo As Integer, status As Long, count As Long) As Long

@ **Argument**

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

status : status of the interrupt data transfer
0 : AD_INT_INIT : initializes transfer
1 : AD_INT_RUN : transfer is not completed
2 : AD_INT_STOP : transfer is completed

count : current conversion count number.

@ **Return Code**

ERR_NoError
ERR_BoardNoInit

6.2.46 _9118_AD_INT_Stop

@ Description

This function is used to stop the interrupt data transfer function. After executing this function, the internal A/D trigger is disable and the A/D timer is stopped. The function returns the number of the data which has been transferred.

@ Syntax

C/C++ (DOS)

```
int _9118_AD_INT_Stop( int *count )
```

C/C++ (Windows 95)

```
int W_9118_AD_INT_Stop(U16 cardNo, int *count )
```

Visual Basic (Windows 95)

```
W_9118_AD_INT_Stop (ByVal cardNo As Integer, count As
    Long) As Long
```

@ Argument

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

count : the number of A/D data which has been transferred.

@ Return Code

ERR_NoError
ERR_BoardNoInit
ERR_AD_INTNotSet

6.2.47 _9118_TIMER_Start

@ Description

The Timer #0 on the PCI-9118 can be freely programmed by the users. This function is used to program the Timer #0. This timer

can be used as frequency generator if internal clock is used. It also can be used as event counter if external clock is used.

@ Syntax

C/C++ (DOS)

int _9118_TIMER_Start(int timer_mode, unsigned int c0)

C/C++ (Windows 95)

int W_9118_TIMER_Start(U16 cardNo, int timer_mode, U16 c0)

Visual Basic (Windows 95)

W_9118_TIMER_Start (ByVal cardNo As Integer, ByVal timer_mode As Long, ByVal c0 As Long) As Long

@ Argument

cardNo (Win-95):	the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.
timer_mode :	the 8254 timer mode, the possible values are: TIMER_MODE0, TIMER_MODE1, TIMER_MODE2, TIMER_MODE3, TIMER_MODE4, TIMER_MODE5.
c0 :	the counter value of timer

@ Return Code

ERR_NoError
ERR_BoardNoInit
ERR_InvalidTimerMode

Note: If TIMER_MODE4 is used, C0 is one smaller than the counter number you specify. That is, if the counter number you specify is 100, the C0 is set as 99.

6.2.48 _9118_TIMER_Read

@ Description

This function is used to read the counter value of the Timer #0.

@ **Syntax**

C/C++ (DOS)

int _9118_TIMER_Read(unsigned int *counter_value)

C/C++ (Windows 95)

int W_9118_TIMER_Read(U16 cardNo, unsigned int *counter_value)

Visual Basic (Windows 95)

W_9118_TIMER_Read (ByVal cardNo As Integer, counter_value As Long) As Long

@ **Argument**

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

counter_value : the counter value of the Timer #0

@ **Return Code**

ERR_NoError
ERR_BoardNoInit

6.2.49 _9118_TIMER_Stop

@ **Description**

This function is used to stop the timer operation. The timer is set to the 'One-shot' mode with counter value ' 0'. That is, the clock output signal will be set to high after executing this function.

@ **Syntax**

C/C++ (DOS)

int _9118_TIMER_Stop(unsigned int *counter_value)

C/C++ (Windows 95)

int W_9118_TIMER_Stop(U16 cardNo, unsigned int *counter_value)

Visual Basic (Windows 95)

W_9118_TIMER_Stop (ByVal cardNo As Integer, counter_value As Long) As Long

@ **Argument**

cardNo (Win-95): the card number of PCI-9118 to be initialized, totally 4 cards can be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

counter_value : the current counter value of the Timer #0

@ **Return Code**

ERR_Board_NoInit
ERR_NoError

6.2.50 W_9118_Set_Trig

@ **Description**

This function is used to set up a trigger. The function specifies the trigger mode and post trigger count. Please refer to section 5.1.4 for the detailed description of trigger setting. This function is only supported by Windows 95 version.

@ **Syntax**

C/C++ (Windows 95)

```
Int W_9118_Set_Trig(U16 cardNo, int trig_mode, U16
    post_trig_cnt)
```

Visual Basic (Windows 95)

```
W_9118_Set_Trig (ByVal cardNo As Integer, ByVal trig_mode
    As Integer, ByVal post_trig_cnt As Long) As Long
```

@ **Argument**

cardNo: the card number of PCI-9118 to be initialized, the valid card numbers are PCI_CARD1, PCI_CARD2, .., PCI_CARD12.

trig_mode: selected trigger mode. The valid values are the following:
SOFT_TRIG: Software trigger
POST_TRIG: Post trigger
PRE_TRIG: Pre-trigger
ABOUT_TRIG: About trigger

post_trig_cnt: The post trigger count. it will count down the Timer#0 after the trigger condition was met (About trigger) or at the moment DMA operation starts(Post-trigger). When the count reaches 0, the counter stops. The counter is used to control the post trigger sampling count of about trigger and control the sampling count of post trigger mode. For Pre-trigger mode, this argument has to be set as 1.

Note: For post trigger mode and about trigger mode, the value of *post_trig_cnt* should be smaller than the value of count set in *9118_AD_DMA_Start* to get the all the data transferred during Timer#0 counts down. Please refer to section 6.2.36 for the detailed description.

@ **Return Code**

PCICardNumErr
PCICardNotInit
InvalidClkDiv
NoError

6.2.51 W_9118_Alloc_AI_Mem

@ **Description**

Contact Windows 95 system to allocate a block of contiguous memory for continuous AI transfer. This function is only available in Windows 95 version.

@ **Syntax**

C/C++ (Windows 95)

```
int W_9118_Alloc_AI_Mem (U32 buf_size, HANDLE *memID,
    U32 *linearAddr, U32 * phyAddr)
```

Visual Basic (Windows 95)

```
W_9118_Alloc_AI_Mem (ByVal buf_size As Long, memID As
    Long, linearAddr As Long, phyAddr As Long) As Long
```

@ **Argument**

buf_size: Bytes to allocate. Please be careful, the unit of this argument is BYTE, not SAMPLE.

memID: If the memory allocation is successful, driver returns the ID of that memory in this argument. Use this memory ID in W_9118_AD_DMA_Start, W_9118_AD_INT_Start and W_9118_ContDmaStart function calls.

linearAddr: The linear address of the allocated AI memory. You can use this linear address as a pointer in C/C++ to access the AI data.

phyAddr: The physical address of the allocated AI memory.

@ **Return Code**

NoError
AllocDMAMemFailed

6.2.52 W_9118_Free_AI_Mem

@ **Description**

De-allocate a system AI memory under Windows 95 environment. This function is only available in Windows 95 version.

@ **Syntax**

C/C++ (Windows 95)

int W_9118_Free_AI_Mem (HANDLE memID)

Visual Basic (Windows 95)

W_9118_Free_AI_Mem (ByVal memID As Long) As Long

@ **Argument**

memID: The memory ID of the system AI memory to deallocate.

@ **Return Code**

NoError

6.2.53 W_9118_Get_Sample

@ **Description**

For the language without pointer support such as Visual Basic, programmer can use this function to access the index-th data in AI buffer.

@ **Syntax**

C/C++ (Windows 95)

```
int W_9118_Get_Sample (U32 linearAddr, U32 index, I16
    *ai_data)
```

Visual Basic (Windows 95)

```
W_9118_Get_Sample (ByVal linearAddr As Long, ByVal idx
    As Long, ai_data As Integer) As Long
```

@ **Argument**

linearAddr: The linear address of the allocated AI memory.
index: The index of the sample. The first sample is with index 0.
ai_data: Returns the sample retrieved.

@ **Return Code**

NoError

7

Calibration

In data acquisition process, how to calibrate your measurement devices to maintain its accuracy is very important. Users can calibrate the analog input and analog output channels under the users' operating environment for optimizing the accuracy. This chapter will guide you to calibrate your PCI-9118 to an accuracy condition.

7.1 What do you need

Before calibrating your PCI-9118 card, you should prepare some equipment' s for the calibration:

- Calibration program : Once the program is executed, it will guide you to do the calibration. This program is included in the delivered package.
- A 5 1/2 digit multimeter (6 1/2 is recommended)
- A voltage calibrator or a very stable and noise free DC voltage generator.

7.2 VR Assignment

There are five variable resistors (VR) on the PCI-9118 board to allow you making accurate adjustment on A/D and D/A channels. The function of each VR is specified as Table 7.1.

VR1	D/A channel 1 full scale adjustment
VR2	D/A channel 1 offset adjustment
VR3	D/A channel 2 full scale adjustment
VR4	D/A channel 2 offset adjustment
VR5	A/D full scale adjustment
VR6	A/D bipolar offset adjustment
VR7	A/D unipolar offset adjustment

Table 7.1 Function of VRs

7.3 A/D Adjustment

7.3.1 Bipolar Calibration

1. Set the analog input range as : ±5V, i.e. the gain = 1 and input mode = Bipolar.
2. Connect A/D channel 0 (pin 26 of CN1) to ground (pin 34 of CN1), and Applied a +5V to A/D channel 1 (pin 27 of CN1).
3. Trim **VR6** to obtain the reading of A/D channel 0 flicks between 2048~2049 (9118DG/HG) or 0 to 1 (9118HR), and Trim **VR5** to obtain reading of A/D channel 1 flicks between 4094~4095 (9118DG/HG) or 32766~32767 (9118HR).

7.3.2 Unipolar Calibration

1. Set the analog input range as : 0 ~ 10 V, i.e. the gain = 1 and input mode = Unipolar.
2. Applied a +5 V input signal to A/D channel 0, and trim the ***VR7*** to obtain reading flicking between 2047~2048 (9118DG/HG) or 0~1 (9118HR).

7.4 D/A Adjustment

7.4.1 DA Channel 1 Calibration

1. Connect VDM (+) to CN1 pin-35 (AO1) and VDM (-) to CN1 pin-34 (A.GND).
2. DA1 output 0x000.
3. Trim the variable resister **VR2** to obtain -10V reading in the DVM.
4. DA1 output 0xFFF.
5. Trim the variable resister **VR1** to obtain +10V reading in the DVM.

7.4.2 DA Channel 2 Calibration

1. Connect VDM (+) to CN1 pin-36 (AO2) and VDM (-) to CN1 pin-34 (A.GND).
2. DA2 output 0x000.
3. Trim the variable resister **VR4** to obtain -10V reading in the DVM.
4. DA2 output 0xFFF.
5. Trim the variable resister **VR3** to obtain +10V reading in the DVM.

8

Software Utility

This software CD provides a utility program, 9118util.exe which provides three functions, System Configuration, Calibration, and Functional Testing. This utility is designed as menu-driven based windowing style. Not only the text messages are shown for operating guidance, but also has the graphic to indicate you how to set right hardware configuration. This utility is described in the following sections.

8.1 Running 9118util.exe

After finishing the DOS installation, you can execute the utility by typing as follows (assume your utility is located in \ADLINK\DOS\9118\Util directory) :

C> cd \ADLINK\DOS\9118\Util

C> 9118UTIL

the following diagram will be displayed on you screen. The message at the bottom of each window guides you how to select item, go to next step and change the default settings.

```
            ****** PCI-9118 Utility  Rev. 1.0 ******

Copyright © 1995-1997, ADLink Technology Inc.  All rights reserved.
```

```
                <F1> : Configuration.

                <F2> : Calibration.

                <F3> : Function testing.

                <Esc>: Quit.
```

```
>>> Select function key F1 ~ F3, or press <Esc> to quit. <<<
```

8.2 System Configuration

This function guides you to configure the PCI-9118 card, and set the right hardware configuration. The configuration window shows the setting items that you have to set before using the PCI-9118 card.

The following diagram will be displayed on the screen as you choose the Configuration function from main menu.

```
****** Calibration of PCI9118 ******

<1> Card Type                    9118DG
<2> ADC Trigger Source          Internal
<3> Timer Clock Source          Internal
<4> AD Input Channel Config    Single-Ended
<5> AD Polarity setting          Bipolar
<6> AD Input Range                Gain=1  Bipolar(-5V~5V)

>>> <Up/Down>: Select Item,  <PgUp/PgDn>: Change Setting <<<
```

8.3 Calibration

This function guides you to calibrate the PCI-9118. The calibration program serves as a useful test of the PCI-9118's A/D,D/A and DIO functions and can aid in troubleshooting if problems arise.

Note: For an environment with frequently large changes of temperature and vibration, a 3 months re-calibration interval is recommended. For laboratory conditions, 6 months to 1 year is acceptable

When you choose the calibration function from the main menu list, a calibration items menu is displayed on the screen. After you select one of the calibration items from the calibration items menu, a calibration window shows. The upper window shows the detailed procedures which have to be followed when you proceed the

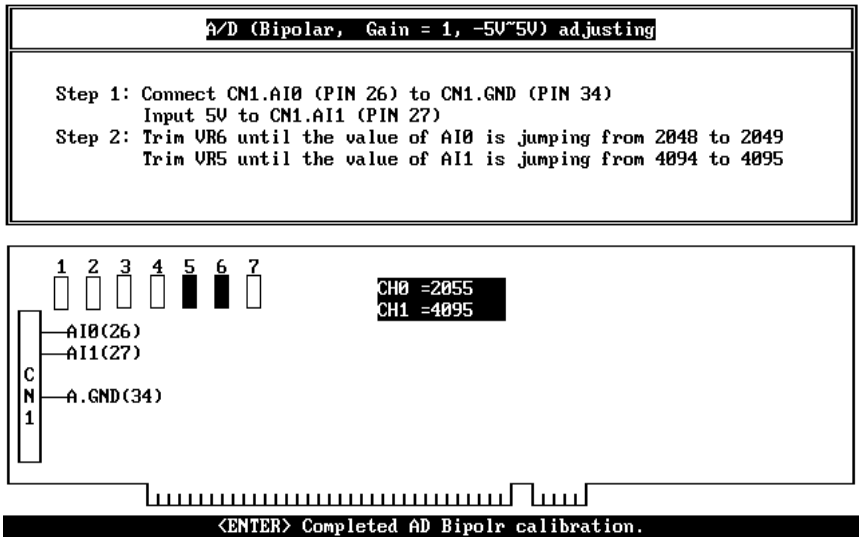
calibration. The instructions will guide you to calibrate each item step by step. The bottom window shows the layout of PCI-9118. In addition, the proper Variable Resister (VR) will blink to indicate the related VR which needs to be adjusted for the current calibration step.

```
                ****** PCI-9118 Calibration ******
```

```
        <1> D/A channel 1 voltage full range adjusting

        <2> D/A channel 2 voltage full range adjusting

        <3> A/D (Bipolar Gain = 1, -5V ~ 5V) adjusting

        <4> A/D (Unipolar Gain = 1, -5V ~ 5V) adjusting

        <Esc> Quit
```

```
    Select 1 to 4 or <Esc> to quit calibration.
```

If you select 3, the following figure displays on the screen:



<F10> Completed AD Bipolr calibration, otherwise repeat Step 1 to 4.

8.4 Functional Testing

This function is used to test the functions of PCI-9118, it includes Digital I/O testing, D/A testing, A/D polling testing, A/D Interrupt Testing, A/D with DMA testing, A/D with DMA & Burst Mode testing, A/D with DMA & BSSH testing, and A/D with DMA & post-trigger testing.

When you choose one of the testing function from the functions menu, a diagram is displayed on the screen. The figures below are the function testing menu window and A/D with Polling Testing window.

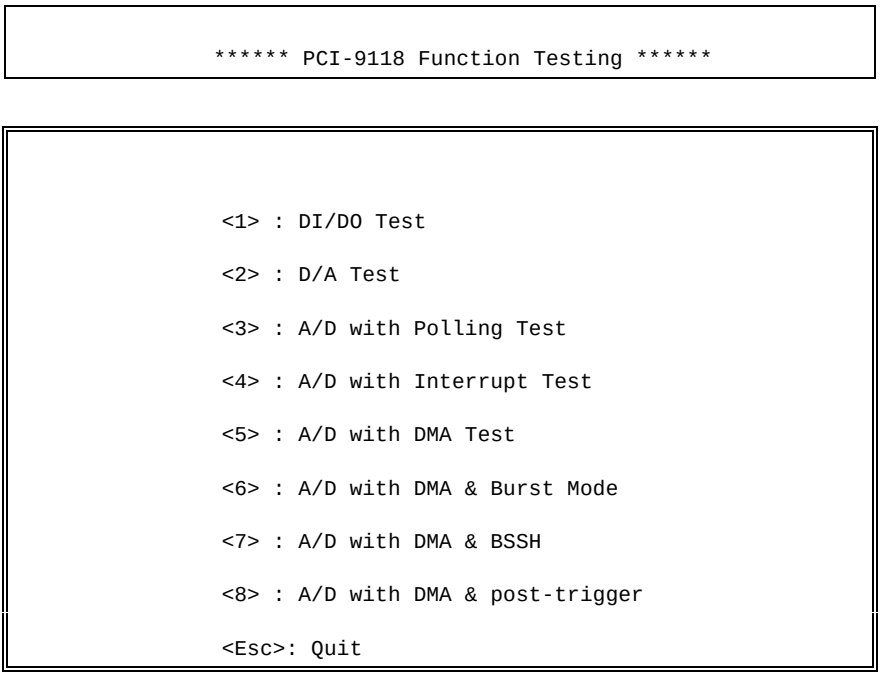


Figure 8.1 Function Testing Menu Window

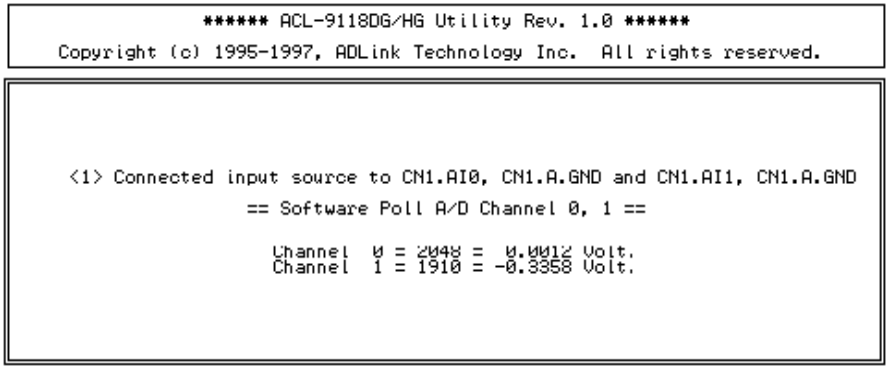


Figure 8.2 A/D with Polling Test window

A calibration utility is supported in the software CD which is included in the product package. The detailed calibration procedures and description can be found in the utility. Users only need to run the software calibration utility and follow the procedures. You will get the accurate measure data.

In normal condition, the PCI-9118 already calibrated by factor before it is shipped out. So, users do not need to calibrate your PCI-9118 when you get it.

Product Warranty/Service

Seller warrants that equipment furnished will be free form defects in material and workmanship for a period of one year from the confirmed date of purchase of the original buyer and that upon written notice of any such defect, Seller will, at its option, repair or replace the defective item under the terms of this warranty, subject to the provisions and specific exclusions listed herein.

This warranty shall not apply to equipment that has been previously repaired or altered outside our plant in any way as to, in the judgment of the manufacturer, affect its reliability. Nor will it apply if the equipment has been used in a manner exceeding its specifications or if the serial number has been removed.

Seller does not assume any liability for consequential damages as a result from our products uses, and in any event our liability shall not exceed the original selling price of the equipment.

The equipment warranty shall constitute the sole and exclusive remedy of any Buyer of Seller equipment and the sole and exclusive liability of the Seller, its successors or assigns, in connection with equipment purchased and in lieu of all other warranties expressed implied or statutory, including, but not limited to, any implied warranty of merchant ability or fitness and all other obligations or liabilities of seller, its successors or assigns.

The equipment must be returned postage-prepaid. Package it securely and insure it. You will be charged for parts and labor if you lack proof of date of purchase, or if the warranty period is expired.