PCI-9114DG / 9114HG

Enhanced Multi-Functions

Data Acquisition Card

Manual Rev. 1.00: December 20, 1998

Trademarks

Contents

How to Use This Guide

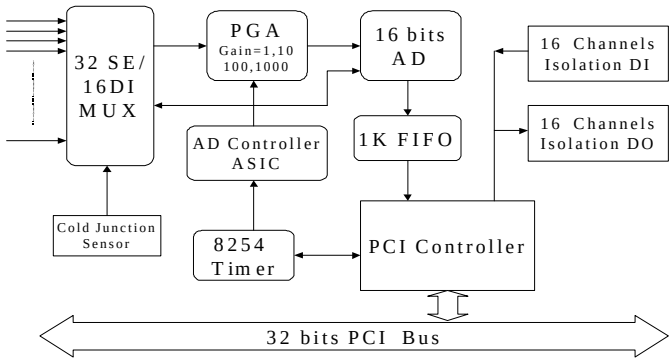
This manual is designed to help you to use the PCI-9114/9114HR. It is divided into six chapters:

- **Chapter 1**, "Introduction," gives an overview of the product features, applications, and specifications.
- **Chapter 2**, "Installation," describes how to install the PCI-9114. The layout of PCI-9114 is shown. The jumpers setting for analog input channel configuration are specified.
- **Chapter 3**, "Signal Connection," describes the connectors' pin assignment and how to connect the outside signal and devices with the PCI-9114.
- **Chapter 4**, "Registers Structure & Format," describes the details of register format and structure of the PCI-9114, 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-9114. The A/D 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-9114 in high-level language style.
- **Chapter 7**, "Calibration & Utility," describes how to calibrate the
- **Chapter 8**, "Software Utility," describes how to use the utility programs included in the software CD.

1

Introduction

The PCI-9114 is an advanced 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. The following block diagram shows the brief functions of the PCI-9114.



1.1 Software Supporting

There are several software options to help you to implement your applications quickly and easily.

Custom Program
For the customers writing their own programs, the PCI-9114 is supported by a comprehensive set of drivers and programming tools. These software drivers support multiple platforms.

- MS-DOS C/C++ programming library.
- Dynamic linking library for Win-95
- PCI-DASK/NT: Advanced data acquisition software kit for Win-NT.

1.2 Features

The PCI-9114 PCI Bus Advanced Data Acquisition Card provides the following advanced features:

- 32-bit PCI-Bus, Plug and Play
- 32 single-ended or 16 differential analog inputs channels
- 16 bits high resolution AD conversion
- High amplification gain of 1, 10, 100, 1000 for PCI-9114 HG
- Normal gain of 1, 2, 4, 8 for PCI-9114 DG
- High sampling rate up to 100 KHz
- Multi-AD trigger mode: software trigger, timer pacer trigger
- On-board A/D 1K WORDS FIFO memory
- Auto-scanning channel selection
- DB-37 connector
- 16 Isolated Digital Input Channels
- 16 Isolated Digital Output Channels with high driving capability
- 5000 V rms high voltage isolation for DIO channels

1.3 Applications

- Industrial process control
- Transducer, thermocouple, RTD
- Power monitor
- Medical instrument
- Biomedical measurement

1.4 Specifications

¨ ***Analog Input (A/D)***

- **Converter:** B.B. ADS7805 or compatible, successive approximation type
- **Resolution:** 16-bit
- **Input channels:** 32 single-ended or 16 differential input
- **Input Range:** (Software controlled)

  Bipolar: ±10V, ±1V, ±100 mV, ±10 mV (PCI-9114-HG)

  Bipolar: ±10V, ±5V, ±2.5, ±1.25V (PCI-9114-DG)
- **Throughput:** 100 KHz for all gain
- **Over voltage Protection:** Continuous ±35V maximum
- **Input Impedance:** 10 MΩ
- **A/D Trigger Modes:** Software and Pacer timer trigger
- **Data Transfer Mode:** Program control, Interrupt
- **FIFO Buffer Size**: 1024 samples

¨ ***Isolation Digital Input (IDI)***

- **Input channels:** 16 input channels
- Input voltage: 0~ 24 VDC

  Logic L: 0~1.5V

  Logic H: 3~24V
- **Input resistance:** 1.2K KΩ @ 0.5W
- **Isolation voltage:** 5000 V rms
- **Throughput:** 10 KHz

- **Polarity:** bipolar input

¨ ***Isolation Digital Output (IDO)***

- **Output channels:** 16 input channels current source
- **Output type:** open emitter 0.5 to 50 V_{DC}
- **Source Current:**
  375 mA (typical), 500 mA (maximum) per channel
- **Isolation voltage:** 5000 V rms
- **Throughput:** 10 KHz

¨ ***General Specifications***

- **Connector**: 37-pin D-type connector
- **Operating Temperature**: 0° C ~ 60° C
- **Storage Temperature**: -20° C ~ 80° C
- **Humidity**: 5 ~ 95%, non-condensing
- **Power Consumption**: +5 V @ 680 mA (max.)
  +12 V @ 100 mA (max.)
- **Dimension**: 7.88" (200 mm)(L) x 4.18" (106 mm)(W)

2

Installation

This chapter describes how to install the PCI-9114. At first, the contents in the package and unpacking information that you should be careful are described.

2.1 What You Have

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

- PCI-9114 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-9114 card contains sensitive electronic components that can be easily damaged by static electricity.

The card should be put 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 component side up.

Again inspect the module for possible damage.

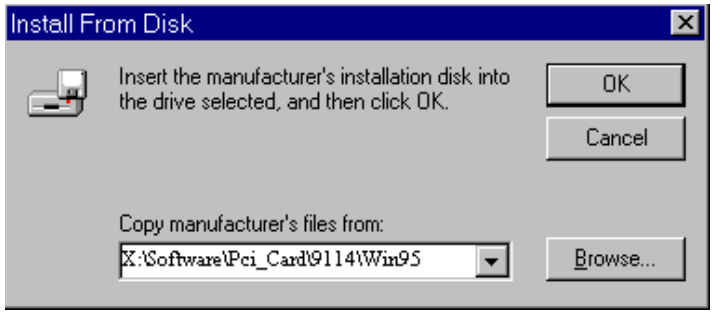
Note : DO NOT APPLY POWER TO THE CARD IF IT HAS BEEN DAMAGED.

You are now ready to install your PCI-9114.

2.3 Device Installation for Windows 95

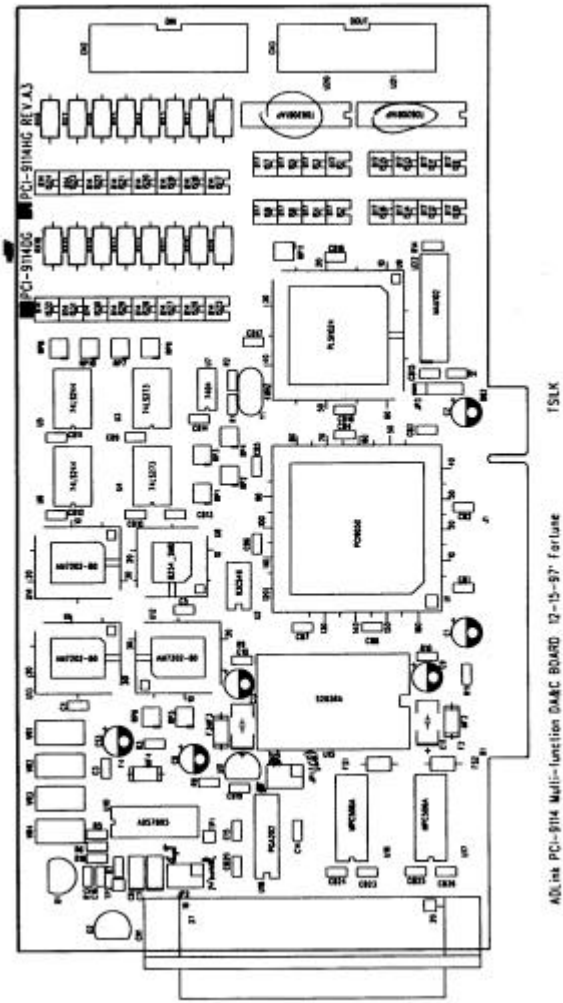
While you first plug PCI-9114 card and enter Windows 95, the system will detect this device automatically and show the following dialog box that prompts you to select the device information source.

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\9114\Win95" (this directory includes PCI-9114 device information file "Pci9114.inf") in the input field (**X indicates the CD ROM driver**) and then click OK. The system will start the installation of PCI-9114.

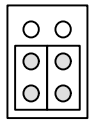
2.4 PCB Layout of PCI-9114



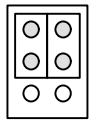
2.5 Jumper Descriptions

2.5.1 JP1: Analog Signal Input Type Selection

JP1 is the selection jumper of analog signal input type. The following diagram shows the possible configurations.



JP1: Single-Ended Input (Default)

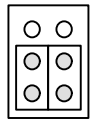


JP1: Differential Input

2.5.2 JP2: Cold Junction Sensor Selection

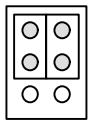
JP2 is used to set the usage cold junction sensor. Note that this jumper is used with JP1 together. The following diagram shows the possible configurations.

- No use the C.J. Sensor (Default)



JP2: Not Use the C.J. Sensor

- Connect the C.J. sensor to AD Channel #0 under Differential Input mode. Note that if the JP1 is wrongly set as SE mode, the C.J sensor is connected to CH#0 and the CH#16 is connected to ground plane.

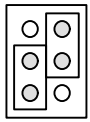


CJ

JP2: Use the C.J. Sensor (D.I. mode)

Normal

- Connect the C.J. Sensor to AD Channel #0 under Single-Ended Input mode. CH#16 is used as normal single-ended input.



CJ

JP2: Use the C.J. Sensor (S.E. mode)

Normal

2.6 PCI Configuration

1. Plug and Play:

As a plug and play component, the system BIOS assigns an interrupt level and base address based on the board information and system parameters. It's no need for user to setup any jumper for these parameters. Users can just plug the card and use it.

2. Configuration:

The board configuration is assigned on a board-by-board basis for all PCI cards on your system. The configurations for every cards in system are subject to change with every boot of the system when new boards are added or removed.

3. Trouble shooting:

If your system can not 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 PCI-9114, and the signal connection between the PCI-9114 and external devices.

3.1 Connectors Pin Assignment

- ***Pin Assignment of CN1***

The PCI-9114 comes equipped one 37-pin D-type connector - CN1. The pin assignment of CN1 is illustrated in the Figure 3.1.

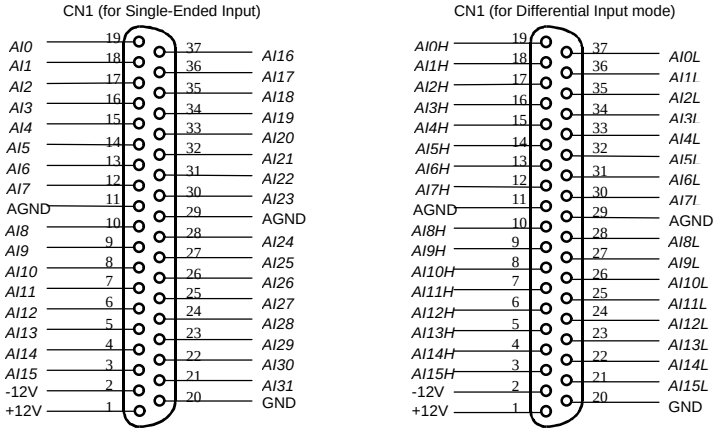


Figure 3.1. Pin Assignment of CN1

Legend:

AIn : Analog Input Channel #*n* (for single-ended mode, n=0~15)
AIxH : Analog Input Channel #*x* (for differential positive input,x=0-31)
AIxL : Analog Input Channel #*x* (for differential negative input,x=0-31)
AGND : Ground plane for analog signals
+12V : +12V power supply (with fuse protection)
-12V : -12V power supply (with fuse protection)
GND : Ground Plane

- ***Pin Assignment of CN2 & CN3***

The CN2 and CN3 is used for isolation digital input and output signals respectively. The pin assignment of CN2 and CN3 is illustrated in the Figure 3.2.

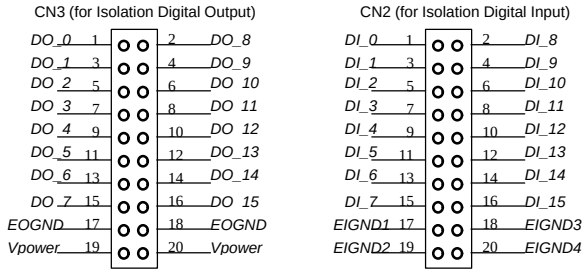


Figure 3.2. Pin Assignment of CN2 & CN3

Legend:

DIn : Isolation Digital Input Channel #*n* (n=0~15)
EIGNDx: Isolation Input Signal Ground plane #x (x=1~4)

DOn : Isolation Digital Output Channel #*n*
EOGND: Isolation Output Signal Ground
Vpower: Isolation Output Driver's Power Supply

3.2 Analog Input Signal Connection

The PCI-9114 provides 32 single-ended or 16 differential analog input channels. To avoid ground loops and get more accurate measurement of A/D conversion, it is quite important to understand the signal source type.

The single-ended (SE) mode means the voltage signal to be measured is relative to the isolation ground (IGND) and is suitable for connecting with the *floating signal source*. The floating source means it have no connection with real ground or ground of your PC. Figure 3.4 shows the single-ended AI signal connection. Note that when more than two floating sources are connected, the sources must be with common ground.

The differential input (DI) mode means the voltage signal to be measured is by a pair of signals, for example, AI3L and AI3H is a differential pair. The AD circuits measure the voltage difference between the differential pair. The common mode noise can be reduced under this mode. Note that the differential signal pair should be still common ground to the isolation ground plane. Figure 3.5 shows the differential analog signal input connection.

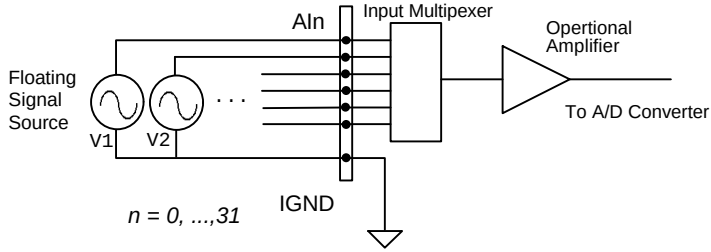


Figure 3.4 Floating sources and single-ended connection

3.3 Isolation Digital Input Signal Connection

There are 16 isolation digital input signals. Every digital input signal is connect to one photo isolator such that the signal is isolated from the ground or the power plane of the host PC. The Figure 3.6 illustrates the single digital input circuits.

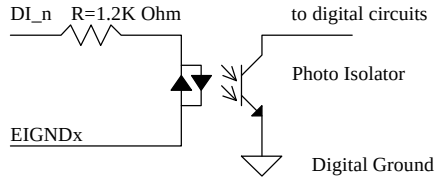


Figure 3.6 Isolation Input Circuits

The isolation digital input could be AC input. The isolation voltage is 2500 V rms. The input resistance is 1.2K Ohm.

Note that the 16 DI signals are partitioned into 4 groups. Each group is based on common ground. Every two groups are mutual isolated. Please refer the Figure 3.2 and the Table 3.1 for the four groups.

Signal Names	Common Ground Signal
ID_0~ID_3	EIGND1
ID_4~ID_7	EIGND2
ID_8~ID_11	EIGND3
ID_12~ID_15	EIGND4

Table 3.1 Digital input signals and ground plane

3.4 Isolation Digital Output Signal Connection

There are 16 isolation digital output signals. The digital output signals are driven by the Darlington Transistors. The Figure 3.7 shows the output circuits.

Note that the 16 DO signals are using common ground and common external power supply.

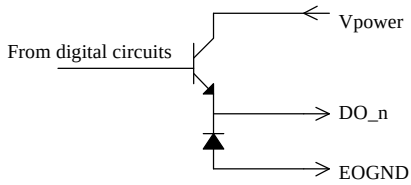


Figure 3.7 Digital output circuits

3.5 Daughter Board Connection

The PCI-9114 can be connected with several different daughter boards, ACLD-8125, ACLD-9137 and ACLD-9138. The functionality and connections are specified as follows.

3.3.1 Connect with ACLD-9137

The ACLD-9137 is a direct connector for the add-on card that is equipped with 37-pin D-sub connector. This board provides a simple way for connection. It is very suitable for the simple applications that do not need complex signal condition in front of the A/D conversion.

3.3.2 Connect with ACLD-9188

ACLD-9188 is a general purposed terminal board for all the cards which come equipped with 37-pin D-sub connector.

3.3.3 Connect with ACLD-9178

ACLD-9178 is a general purposed terminal board for all the cards which come equipped with two 20-pin header connector.

4

Registers Structure & Format

The detailed descriptions of the register format of the PCI-9114 are specified in this chapter. This information is quite useful for the programmers who wish to handle the card by low-level programming. In addition, users can understand how to use software driver to manipulate this card after understanding the registers' structure of the PCI-9114.

4.1 I/O Port Address

The PCI-9114 functions as a 32-bit PCI target device to any master on the PCI bus. There are three types of registers on the PCI-9114: PCI Configuration Registers (PCR), Local Configuration Registers (LCR) and PCI-9114 registers.

The PCR that conform the PCI-bus specifications are initialized and controlled by the system plug & play PCI BIOS. Users can study the PCI BIOS specifications to understand the operation of the PCR. The PCR can only be read through by PCI BIOS function call.

The LCR is defined by the PCI bus controller “PCI9050”. It is not necessary for users to understand the details of the LCR if you

use the software library. The base address of the LCR is assigned by the PCI BIOS. The assigned address is located at offset 14h of PCR.

The PCI-9114 registers are shown in the Table 4.1. The base address of the PCI-9114 registers is also assigned by the PCI p&p BIOS. The assigned base address is located at offset 18h of PCR. Note that most of the PCI-9114 registers are 16 bits. The users can access these registers by 16 bits I/O instructions.

There is one 32 bits register on PCI-9114. The 32 bits register occupied another LCR address space, that is, base address #2. The base address is allocated by PCI BIOS and is stored at offset 1Ch of PCR. Users can read the PCR to get the LCR base address and the two PCI-9114 base addresses by using the PCI BIOS function call.

I/O Base Address #1	Write	Read
Base + 00h	Isolation DO port	Isolation DI port
Base + 02h	AD MUX channel no.	AD MUX channel no setting
Base + 04h	AD range control	AD range control read back
Base + 06h	AD trigger mode	AD trigger mode read back
Base + 08h	Interrupt control	Interrupt control read back
Base + 0Ah	Software AD trigger	FIFO status read back
Base + 0Ch	Clear H/W IRQ1	--
Base + 0Eh	Clear H/W IRQ2	--
Base + 20h	8254 Counter #0	8254 Counter #0
Base + 22h	8254 Counter #1	8254 Counter #1
Base + 24h	8254 Counter #2	8254 Counter #2
Base + 26h	8254 Control Registers	8254 Status Registers
I/O Base Address #2	**Write**	**Read**
Base 2 + 40h	--	32 bits AD FIFO data and channel number read

Table 4.1 I/O Address

4.2 A/D Data Registers

The PCI-9114 A/D data is stored in the FIFO after conversion. The data can be transferred to host memory by software only. 16-bit input port instruction can read the AD value. The A/D data can also be read back with the channel number together. User will know exactly the channel number of the A/D data. However, it must use 32 bits reading port instruction.

Address: BASE2 + 0x40
Attribute: read only
Data Format:

Bit	7	6	5	4	3	2	1	0
BASE+0	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0
BASE+1	AD15	AD14	AD13	AD12	AD11	AD10	AD9	AD8
BASE+2	X	X	X	CH4	CH3	CH2	CH1	CH0
BASE+3	X	X	X	X	X	X	X	X

AD15~AD0: Analog to digital data. AD15 is the Most Significant Bit (MSB). AD0 is the Least Significant Bit (LSB).
CH4~CH0: Channel number of the A/D data.

4.3 A/D Channel Control Register

The PCI-9114 provides 32 SE or 16 DI channels. The channel control register is used to set the A/D channels to be converted. The 5 LSBs of this register control the channel number. Under non-auto scanning mode, the register sets the channel number for conversion. Under auto-scanning mode, the register set the ending channel number. Note that the read back value is the setting value but not the current selected AD channel number.

Address: BASE + 2
Attribute: write and read
Data Format:

Bit	7	6	5	4	3	2	1	0
BASE+2	x	x	x	CN4	CN3	CN2	CN1	CN0
BASE+3	x	x	x	x	x	x	x	x

CNn : channel number of multiplexer.
CN4 is MSB, and CN0 is LSB.

4.4 A/D Input Signal Range Control Register

The A/D range register is used to adjust the analog input ranges. This register directly controls the PGA (programmable gain amplifier). When a different gain value is set, the analog input range will be changed to its corresponding value.

Address: BASE + 4
Attribute: write and read
Data Format:

Bit	7	6	5	4	3	2	1	0
BASE+4	X	X	X	X	X	X	G1	G0
BASE+5	X	X	X	X	X	X	X	X

The relationship between gain setting and its corresponding A/D range is listed in the table below.

G1	G0	Gain HG/ DG	Analog Input Range of PCI-9114HG	Analog Input Range of PCI-9114DG
0	0	1 / 1	±10V	±10V
0	1	10 / 2	±1V	±5V
1	0	100 / 4	±100 mV	±2.5V
1	1	1000 / 8	±10 mV	±1.25V

4.5 A/D Status Read-back Register

The A/D FIFO status can be read back from this register.

Address: BASE + 0x0Ah
Attribute: read only
Data Format:

Bit	7	6	5	4	3	2	1	0
BASE+0Ah	X	X	X	X	AD_BUSY	FF_FF	FF_HF	FF_EF
BASE+0Bh	X	X	X	X	X	X	X	X

FF_EF: '0' means FIFO is empty
FF_HF: '0' means FIFO is half-full
FF_FF: '0' means FIFO is full, A/D data may have been loss
AD_BUSY: '0' means AD is busy, the A/D data has not been latched in FIFO yet. If AD_BUSY changes from '0' to '1', A/D data is written into FIFO.

4.6 A/D Trigger Mode Control and Read-back Register

This register is used to control or read back the A/D trigger control setting and the AD range setting.

Address: BASE + 06h
Attribute: write and read
Data Format:

Bit	7	6	5	4	3	2	1	0
BASE+6	X	X	X	X	X	X	TSSEL	ASCAN
BASE+7	X	X	X	X	X	X	X	X

ASCAN: Auto Scan Control
1: Auto Scan ON
0: Auto Scan OFF
TSSEL: Timer Pacer / Software Trigger
1: Timer Pacer Trigger

0: Software Trigger

4.7 Interrupt Control and Read-back Register

The PCI-9114 has a dual interrupt system, thus two interrupt sources can be generated and be checked by the software. This register is used to select the interrupt sources.

Address: BASE + 8
Attribute: write and read
Data Format:

Bit	7	6	5	4	3	2	1	0
BASE+12	X	X	X	X	MUX	FFEN	ISC1	ISC0

ISC0: IRQ0 signal select
0: IRQ on the ending of the AD conversion (EOC)
1: IRQ when FIFO is half full
ISC1: IRQ1 signal select (Timer Interrupt only)
FFEN: FIFO enable pin
0: FIFO Enable (Power On default value)
1: FIFO Disable
(To reset FIFO, set FFEN sequence as 0 -> 1 -> 0)
MUX: This is read-only bit to indicate the JP1 is set to single-ended (1) or differential-input (0).

4.8 Software Trigger Register

To generate a trigger pulse to the PCI-9114 for A/D conversion, you just write any data to this register, then the A/D converter will be triggered.

Address: BASE + 0x0Ah
Attribute: write only
Data Format:

Bit	7	6	5	4	3	2	1	0

BASE+0Ah	X	X	X	X	X	X	X	X
BASE+0Bh	X	X	X	X	X	X	X	X

4.9 Hardware Interrupt Clear Registers

Because the PCI interrupt signal is level trigger, the interrupt clear register must be written to clear the flag after processing the interrupt request event; otherwise, that another interrupt request is inserted will cause the software to hang on processing the interrupt event.

There are two interrupt clear registers. The two registers are used to clear the IRQ1 and IRQ2 respectively.

Address: BASE + $0C_{(hex)}$
Attribute: write only
Data Format:

Bit	7	6	5	4	3	2	1	0
BASE+0Ch	X	X	X	X	X	X	X	X

Address: BASE + $0E_{(hex)}$
Attribute: write only
Data Format:

Bit	7	6	5	4	3	2	1	0
BASE+0Eh	X	X	X	X	X	X	X	X

4.10 Timer/Counter Register

The 8254 chip occupies 4 I/O address locations in the PCI-9114 as shown blow. Users can refer to NEC's or Intel's data sheet for the full description of the 8254 features.

Address : BASE + 0x20h ~ BASE + 0x26h
Attribute : read / write

Data Format :

Base + 0x20h	Counter 0 Register (R/W)
Base + 0x22h	Counter 1 Register (R/W)
Base + 0x24h	Counter 2 Register (R/W)
Base + 0x26h	8254 CONTROL BYTE (W)

4.11 High Level Programming

To operate the PCI-9114, you can bypass the detailed register structures and use the high-level application programming interface (API) to control your PCI-9114 card directly. The software library, DOS library for Microsoft C/C++ and Borland C/C++ is included in the Utility & Library CD. Please refer to chapter 6 for more detailed information.

5

Operation Theorem

The operation theorem of the functions on PCI-9114 card is described in this chapter. The operation theorem can help you to understand how to manipulate or to program the PCI-9114.

5.1 A/D Conversion

Before programming the PCI-9114 to perform the A/D conversion, you should understand the following issues:

- ***A/D conversion procedure***
- ***A/D signal source control***
- ***A/D trigger source control***
- ***A/D data transfer mode***
- ***Interrupt System (refer to section 5.2)***
- ***A/D data format***

Note : Because some of the A/D data transfer modes will use the system interrupt resource. Ther user have to understand the interrupt system (section 5.2) for understanding AD operation.

5.1.1 A/D Conversion Procedure

For using the A/D converter, users must know about the property of the signal to be measured at first. The users can decide which channels to be used and connect the signals to the PCI-9114. Refer to the chapter 3 'Signal Connection'. In addition, users should define and control the A/D signal sources, including the A/D channel, A/D gain, and A/D signal types. Please refer to section 5.1.2 for A/D signal source control.

After deciding the A/D signal source, the user must decide how to trigger the A/D conversion and define/control the trigger source. The A/D converter will start to convert the signal to a digital value when a trigger signal is rising. Refer to the section 5.1.3 for the two trigger sources.

The A/D data should be transferred into PC's memory for further using or processing. The data can be read by I/O instruction which is handled directly by software or transferred to memory via interrupt. Please refer to section 5.1.4 to obtain ideas about the multi-configurations for A/D data transfer.

To process A/D data, programmer should know about the A/D data format. Refer to section 5.1.5 for details.

5.1.2 A/D Signal Source Control

To control the A/D signal source, the signal type, signal channel and signal range should be considered.

Signal Type

The A/D signal sources of PCI-9114 could be single ended (SE) or differential input (DI). There are 32 SE or 16 DI A/D channels on board.

Signal Channel Control

There are two ways to control the channel number. The first one is the software programming and the second one is the auto channel scanning which is controlled by the ASCAN bit in AD trigger mode control register. As ASCAN is cleared (0), the value of AD Channel Control register defines the channel to be selected.

As ASCAN is set 1, the value in AD channel control register defines the ending channel number of auto-scanning operation. Under auto scan mode, the channel is scanning from channel 0 to the ending channel. Whenever a trigger signal is rising, the channel number to be selected will increase automatically. For example, if the ending channel number is 3, the auto channel scanning sequence is 0,1,2,3,0,1,2,3, …, until the ASCAN bit is cleared.

Signal Range

The proper signal range is important for data acquisition. The input signal may be saturated if the A/D gain is too large. Sometimes, the resolution may be not enough if the signal is small. The maximum A/D signal range of PCI-9114 is +/- 10 volts when the programmable gain value is 1. The A/D gain control register controls the maximum signal input range. The signal gain is programmable with 4 levels (1,10,100,1000) or (1,2,4,8). The signal range of the 32 channels will be identical all the time even if the channel number is scanning. The available signal polarity on PCI-9114 is bi-polar only, however, the 16-bit high resolution of PCI-9114 can cover all the 12-bits uni-polar applications.

5.1.3 A/D Trigger Source Control

The A/D conversion is started by a trigger source, and then the A/D converter will start to convert the signal to a digital value. In PCI-9114, two internal sources can be selected: the software trigger or the timer pacer trigger. The A/D operation mode is controlled by A/D trigger mode register. Total two trigger sources are provided in the PCI-9114. The different trigger conditions are

specified as follows:

Software trigger (TSSEL=0)
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. This trigger mode is suitable for low speed A/D conversion. Under this mode, the timing of the A/D conversion is fully controlled by software. However, it is difficult to control the fixed A/D conversion rate unless another timer interrupt service routine is used to generate a fixed rate trigger. Refer to interrupt control section (section 5.2) for fixed rate timer interrupt operation.

Timer Pacer Trigger (TSSEL=1)
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 5.3 for 8254 architecture. This mode is ideal for high speed A/D conversion. It can be combined with the FIFO half-full interrupt or EOC interrupt to transfer data. It is also possible to use software FIFO polling to transfer data. The A/D trigger, A/D data transfer and Interrupt can be set independently, Most of the complex applications can thus be covered.

It's recommended using this mode if your applications need a fixed and precise A/D sampling rate.

5.1.4 A/D Data Transfer Modes

The A/D data are buffered in the FIFO memory. The FIFO size on PCI-9114 is 1024 (1K) words. If the sampling rate is 100 KHz, the FIFO can buffer 10.24 ms analog signal. After the FIFO is full, the lasting coming data will be lost. The software must read out the FIFO data before it becomes full.

The data must be transferred to host memory after the date is ready and before the FIFO is full. On the PCI-9114, many data

transfer modes can be used. The different transfer modes are specified as follows:

Software Data Polling

The software data polling is the easiest way to transfer A/D data. This mode can be used with software A/D trigger mode. After the A/D conversion is triggered by software, the software should poll the *FF_EF* bit of the A/D status register until it becomes low level.

If the FIFO is empty before the A/D start, the FF_EF bit will be low. After the A/D conversion is completed, the A/D data is written to FIFO immediately, thus the FF_EF becomes high. You can consider the FF_EF bit as a flag to indicate the converted data ready status. That is, FF_EF is high means the data is ready. Note that, while A/D is converted, the ADBUSY bit is low. After A/D conversion, the ADBUSY becomes high to indicate not busy. Please do NOT use this bit to poll the AD data.

It is possible to read A/D converted data without polling. The A/D conversion time will not exceed 8.5μs on PCI-9114 card. Hence, after software trigger, the software can wait for at least 8.5μs and then read the A/D register without polling.

The data polling transfer is very suitable for the application that needs to process AD data in real time. Especially, when combining with the timer interrupt generation, the timer interrupt service routine can use the data polling method to get multi-channel A/D data in real time and with the fixed data sampling rate.

FIFO Half-Full Polling

The FIFO half-full polling mode is the most powerful AD data transfer mode. The 1 K words FIFO can be stored up to 10.24 ms analog data under 100 KHz sampling rate (10.024ms = 1024 / 100KHz). Theoretically, the software can poll the FIFO every 10 ms without taking care how to trigger A/D or transfer A/D data.

It's recommend that users check your system to find out the user software's priority in the special application. If the application software is at the highest priority, polling the FIFO every 10 ms is suitable. However, the user's program must check the FIFO is full or empty every time reading data.

To avoid this problem, the half-full polling method is used. If the A/D trigger rate is 100KHz, the FIFO will be half-full (512 words) in 5.12 ms. If the user's software checks the FIFO half full signal every 5 ms and the FIFO is not half-full, the software does not read data. When the FIFO is full, the AD FIFO is overrun. That means the sampling rate is higher than users' expect or the polling rate is too slow. It is also possible due to your system occupy the CPU resource thus reducing the polling rate. When the FIFO is half-full and not full, the software can read one "block" (512 words) A/D data without checking the FIFO status. This method is very convenient to read A/D in size of a "block" and it is benefit to software programming.

Usually, the timer trigger is used under this mode, therefore the sampling rate is fixed. The method also utilizes the minimum CPU resources because it is not necessary to be the highest priority. The other benefit is this method will not use hardware interrupt resource. Therefore, the interrupt is reserved for system clock or emergency external interrupt request. The FIFO half-full polling method is the most powerful A/D data transfer mode.

EOC Interrupt Transfer

The PCI-9114 provides traditional hardware end-of-conversion (EOC) 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 in the FIFO. It is useful to combine the EOC interrupt transfer with the timer pacer trigger mode. After A/D conversion is completed, the hardware interrupt will be inserted and its corresponding ISR (Interrupt Service Routine) will be

invoked and executed. The ISR program can read the converted data. This method is most suitable for data processing applications under real-time and fixed sampling rate

FIFO Half-Full Interrupt Transfer

Sometimes, the applications do not need real-time processing, but the foreground program is too busy to poll the FIFO data. The FIFO half-full interrupt transfer mode is useful for the situation mentioned above. In addition, as the external A/D trigger source is used, the sampling rate may be not easy to predict, and then the method could be applied. Because the CPU is only interrupted when the FIFO is half-full, thus reserved the CPU load.

Under this mode, an interrupt signal is generated when FIFO becomes half-full. It means there are 512 words data in the FIFO already. The ISR can read a block of data every interrupt occurring. This method is very convenient to read A/D in size of a "block" (512 words) and it is benefit for software programming.

5.1.5 A/D Data Format

The range of A/D data read from the FIFO port is from -32768 to 32767. As the A/D gain is 1, the A/D signal range is -10V ~ +10V. The relationship between the voltage and the value is shown in the following table:

A/D Data (Hex)	Decimal Value	Voltage (Volts)
		±10V (Bipolar)
7FFF	32767	+9.9997
4000	16384	+5.0000
0001	1	+0.0003
0000	0	+0.0000
FFFF	-1	-0.0003
C000	-16385	-5.0000
8001	-32767	-9.9997
8000	-32768	-10.0000

The formula between the A/D data and the analog value is

Voltage = (AD_Data * 10) / (32768 * gain) --Bipolar

where the *gain* is 1,10,100, 1000 for HG version or 1,2,4,8 for DG version.

5.2 Interrupt Control

5.2.1 System Architecture

The PCI-9114's interrupt system is a powerful and flexible system that is suitable for A/D data acquisition and many applications. The system is a ***Dual Interrupt System***. The dual interrupt means the hardware can generate two interrupt request signals in the same time and the software can service these two request signals by ISR. Note that the dual interrupt does not mean the card occupies two IRQ levels.

The two interrupt request signals (INT1 and INT2) come from digital signals or the timer / counter output. An interrupt source multiplexer (MUX) is used to select the IRQ sources. Fig 5.2.1 shows the interrupt system.

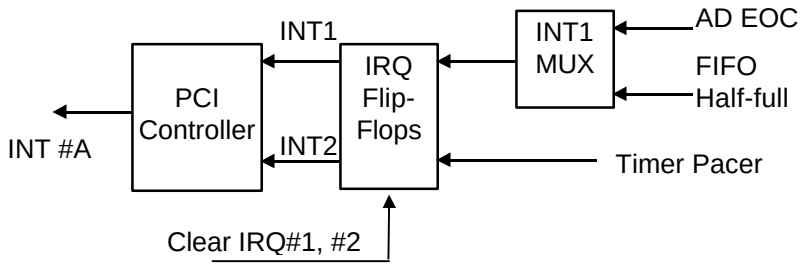


Fig 5.2.1 Dual Interrupt System of PCI-9114

5.2.2 IRQ Level Setting

There is only one IRQ level used by this card, although it is a dual interrupt system. This card uses INT #A interrupt request signal to PCI bus. The motherboard circuits will transfer INT #A to one of the AT bus IRQ levels. The IRQ level is set by the PCI plug and play BIOS. This value is saved in the PCI controller. It is not

necessary for users to set the IRQ level.

5.2.3 Dual Interrupt System

The PCI controller of PCI-9114 can receive two hardware IRQ sources. However, a PCI controller can generate only one IRQ to PCI bus, the two IRQ sources should be distinguished by ISR of the application software if the two IRQ are all used.

The application software can use the "_9114_Get_Irq_Status" function to distinguish which interrupt is inserted. After servicing an IRQ signal, users should check if another IRQ is also asserted and then clear current IRQ to allow the next IRQ occurring.

The two IRQs are named as INT1 and INT2. INT1 comes from AD EOC or the FIFO half-full flag. INT2 comes from timer's pacer output only. The sources of INT1 and INT2 are selective by the Interrupt Control (ISC) Register.

Because of dual interrupt system, for example, you can use FIFO half-full and external interrupt at the same time if your software ISR can distinguish these two events.

5.2.4 Interrupt Source Control

There are two bits to control the IRQ sources of INT1 and INT2. Refer to section 4.9 for the details of the two bits. In addition, the PCI controller itself can also control the use of the interrupt. For manipulating the interrupt system more easily, we recommends you to use the function _9114_INT_Source_Control to control the IRQ source so that you can disable one or two of the IRQ sources.

Note that even you disable all the two IRQ sources without changing the initial condition of the PCI controller, the PCI BIOS still assigns an IRQ level to the PCI card and it will occupy the PC resource. It is not suggested to re-design the initial condition of the PCI card by users' own application software. If users want to

disable the IRQ level, please use the software utility to change the power on interrupt setting.

5.3 Timer/Counter Operation

The PCI-9114 has an interval timer/counter 8254 on board. Refer to section 3.5 for the signal connection and the configuration of the counters.

5.3.1 Introduction

One 8254 programmable timer/counter chip is installed in PCI-9114. There are three counters in one 8254 chip and 6 possible operation modes for each counter. The block diagram of the timer / counter system is shown in following diagram.

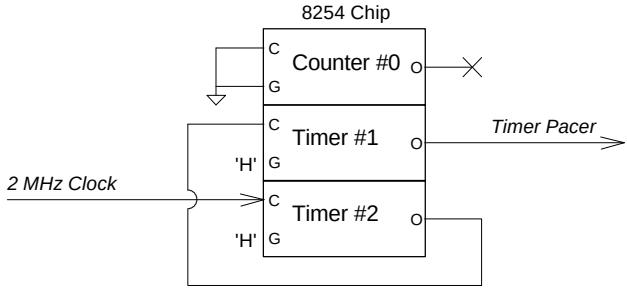


Figure 5.3.1 Timer / counter system of PCI-9114.

5.3.2 Pacer Trigger Source

The timer #1 and timer #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 2MHz/4=500K, which excess the maximum A/D conversion rate of the PCI-9114 (80KHz). The minimum signal rate is 2MHz/65535/65535, which is a very slow frequency that user may

never use it. The output of the programmable timer can be used as the pacer interrupt source or the timer pacer trigger source of A/D conversion. In software library, the timer #1 and #2 are always set as mode 2 (rate generator) or mode 3.

6

C/C++ Software Library

There are more than 30 functions provided by the C Language library. This library includes all the functions of PCI-9114. The major capability of these function calls is A/D conversion. In addition, there are some sample programs to help you to use this library.

6.1 Installation

6.1.1 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\9114\DOS**

  X:\ Software\NuDAQPCI\9114\DOS> **SETUP**

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

After installation, all the files *of PCI-9114 Library & Utility for DOS* are stored in C:\ADLink\9114\dos directory.

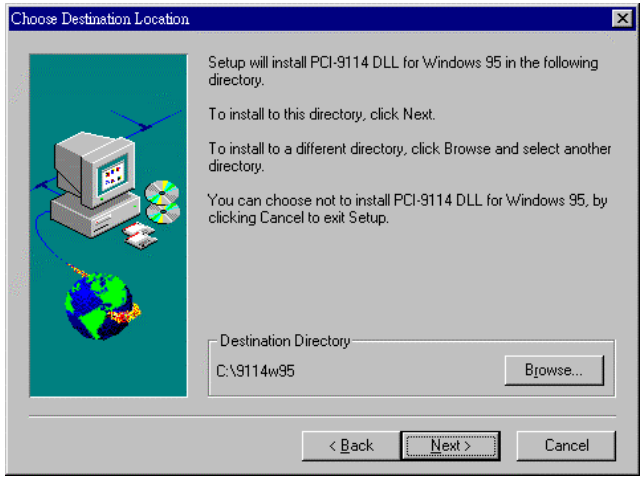
- **Windows 95 Software Installation**

  step 1. Place 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\9114\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\9114\W95. If you want to install *PCI-9114 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-9114 DLL for Windows 95*.



After you complete the installation of PCI-9114 Software, PCI-9114's DLL (9114.DLL) is copied to Windows System directory (default is C:\WINDOWS\SYSTEM) and the driver files (W95_9114.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-9114 library. We suggest you to use these data types in your application programs. The following table shows the data type names and their range.

6.2.1 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 functions of PCI-9114's 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. **_9114_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_9114_Initial()`**.

There are 38 functions provided by PCI-9114 software drivers. The detail descriptions of each function are specified in the following sections.

6.2.2 _9114_Initial

@ Description

This function is used to initialize PCI-9114. Every PCI-9114 has to be initialized by this function before calling other functions.

@ Syntax

C/C++ (DOS)

U16 _9114_Initial (U16 *existCards, PCI_INFO *info)

C/C++ (Windows 95)

U16 W_9114_Initial (U16 *existCards, PCI_INFO *info)

Visual Basic (Windows 95)

W_9114_Initial (existCards As Integer, info As PCI_INFO) As Integer

@ Argument

existCards: numbers of existing PCI-9114 cards
info: relative information of the PCI-9114 cards

@ Return Code

ERR_NoError
ERR_BoardNoInit
ERR_PCIBiosNotExist

6.2.3 _9114_Software_Reset

@ Description

This function is used to reset the I/O port configuration. Note that this function can not re-start the PCI bus and all the

hardware setting won't be changed neither.

@ Syntax

C/C++ (DOS)

void _9114_Software_Reset (U16 cardNo)

C/C++ (Windows 95)

void W_9114_Software_Reset (U16 cardNo)

Visual Basic (Windows 95)

W_9114_Software_Reset (ByVal cardNo As Integer)

@ Argument

cardNo: The card number of initialized PCI-9114 card

@ Return Code

None

6.2.4 _9114_DO

@ Description

This function is used to write data to digital output port. There are 16 digital output channels on PCI-9114.

@ Syntax

C/C++ (DOS)

U16 _9114_DO (U16 cardNo, U16 ADData);

C/C++ (Windows 95)

U16 W_9114_DO (U16 cardNo, U16 DOData)

Visual Basic (Windows 95)

W_9114_DO (ByVal cardNo As Integer, ByVal ADData As Integer) As Integer

@ Argument

cardNo: The card number of initialized PCI-9114 card
DOData: The value will be written to digital output port

@ Return Code

ERR_NoError

6.2.5 _9114_DI

@ Description

This function is used to read data from digital input port. There are 16 digital input channels on PCI-9114. The digital input status can be accessed by this function directly.

@ Syntax

C/C++ (DOS)

U16 _9114_DI (U16 cardNo, U16 far *DIData)

C/C++ (Windows 95)

U16 W_9114_DI (U16 cardNo, U16 *DIData)

Visual Basic (Windows 95)

W_9114_DI (ByVal cardNo As Integer, ADData As Integer) As Integer

@ Argument

cardNo: The card number of initialized PCI-9114 card
DIData: The value accessed from digital input port

@ Return Code

ERR_NoError

6.2.6 _9114_AD_Read_Data

@ Description

This function is used to read the AD conversion data from AD Data register. The resolution of A/D conversion data is 16-bit.

@ Syntax

C/C++ (DOS)

U16 _9114_AD_Read_Data (U16 cardNo, U16 far *ADData)

C/C++ (Windows 95)

U16 W_9114_AD_Read_Data (U16 cardNo, U16 *ADData)

Visual Basic (Windows 95)

W_9114_AD_Read_Data (ByVal cardNo As Integer,
ADData As Integer) As Integer

@ Argument

cardNo: The card number of initialized PCI-9114 card

ADData: A/D converted value. The resolution of AD data is 16-bit. The bit 0 of ADData is the LSB of A/D converted data and the bit 15 of ADData is the MSB of A/D converted data. Please refer to section 5.1.6 for the relationship between the voltage and the digital value.

@ Return Code

ERR_NoError

6.2.7 _9114_AD_Read_Data_Repeat

@ Description

This function is used to read the AD conversion data from the data register n times continuously.

@ Syntax

C/C++ (DOS)

U16 _9114_AD_Read_Data_Repeat (U16 cardNo, I16 far *ADData, U16 n)

C/C++ (Windows 95)

U16 W_9114_AD_Read_Data_Repeat (U16 cardNo, I16 *ADData, U16 n)

Visual Basic (Windows 95)

W_9114_AD_Read_Data_Repeat (ByVal cardNo As Integer, ADData As Integer, ByVal n As Integer) As Integer

@ Argument

cardNo: The card number of PCI-9114 card initialized

ADData: A/D converted value. The resolution of AD data is 16-bit. The bit 0 of ADData is the LSB of A/D converted data and the bit 15 of ADData is the

MSB of A/D converted data. Please refer to section 5.1.5 for the relationship between the voltage and the value.

n : The number of times to read the AD conversion data

@ Return Code

ERR_NoError

6.2.8 _9114_AD_Read_Data_MUX

@ Description

This function is used to read data from A/D Data Registers. The A/D Data and Channel Number Register is a 32-bit register. Please refer to section 4.2 for the description of A/D Data and Channel Number Register.

@ Syntax

C/C++ (DOS)

```
U16 _9114_AD_Read_Data_MUX (U16 cardNo, U32 far
   *ADData )
```

C/C++ (Windows 95)

```
U16 W_9114_AD_Read_Data_MUX (U16 cardNo, U32
    *ADData )
```

Visual Basic (Windows 95)

```
W_9114_AD_Read_Data_MUX (ByVal cardNo As Integer,
    ADData As Long)   As Integer
```

@ Argument

cardNo: The card number of PCI-9114 card initialized

ADData: A/D converted value. The resolution of A/D conversion data is 16-bit. The unsigned integer data format of ADData is as follows:

Every 32-bit unsigned integer data:

bit 0…15: A/D converted data
bit 16, 17, …, 20 : converted channel no.

Please refer to section 5.1.5 for the relationship between the voltage and the value.

@ Return Code

ERR_NoError

6.2.9 _9114_AD_Read_Data_Repeat_MUX

@ Description

This function is used to read data from A/D Data and Channel Number Register n times continuously. The A/D Data and Channel Number Register is a 32-bit register. Please refer to section 4.2 for the description of A/D Data and Channel Number Register.

@ Syntax

C/C++ (DOS)

```
U16 _9114_AD_Read_Data_Repeat_MUX (U16 cardNo,
    U32 far *ADData, U16 n)
```

C/C++ (Windows 95)

U16 W_9114_AD_Read_Data_Repeat_MUX (U16 cardNo, U32 *ADData, U16 n)

C/C++ (Windows 95)

W_9114_AD_Read_Data_Repeat_MUX (ByVal cardNo As Integer, ADData As Long, ByVal n As Integer) As Integer

@ Argument

cardNo: The card number of PCI-9114 card initialized

ADData: A/D converted value. The resolution of AD conversion data is 16-bit. The unsigned integer data format of ADData is as follows:

Every 32-bit unsigned integer data:

bit 0…15: A/D converted data
bit 16, 17, …, 20 : converted channel no.

Please refer to section 5.1.6 to learn the relationship between the voltage and the value.

n : The timer of times to read the AD conversion data.

@ Return Code

ERR_NoError

6.2.10 _9114_AD_Set_Channel

@ Description

This function is used to set AD channel by means of writing data to the channel control register. There are 32 single-ended A/D channels in PCI-9114, therefore the channel number could be set between 0 to 31. Under non-auto scan mode, the ADChannelNo stores the channel number setting. Under auto-scan mode, the ADChannelNo records the channel number of ending channel.

@ Syntax

C/C++ (DOS)

U16 _9114_AD_Set_Channel (U16 cardNo, U16 ADChannelNo)

C/C++ (Windows 95)

U16 W_9114_AD_Set_Channel (U16 cardNo, U16 ADChannelNo)

Visual Basic (Windows 95)

W_9114_AD_Set_Channel (ByVal cardNo As Integer, ByVal ADChannelNo As Integer) As Integer

@ Argument

cardNo: The card number of PCI-9114 card initialized.
ADChannelNo: The selected channel number or the ending channel number to perform A/D conversion.

@ Return Code

ERR_NoError

6.2.11 _9114_AD_Set_Range

@ Description

This function is used to set the A/D range by means of writing data to the AD range control register. The initial value of gain is '1' which is the default setting by PCI-9114 hardware. The relationship between gain and input voltage ranges is specified by the following tables:

For PCI9114HG:

Input Range (V)	Gain	Gain Code
±10 V	X 1	AD_B_10_V
±1 V	X 10	AD_B_1_V
±100m V	X 100	AD_B_0_1_V
±10m V	X 1000	AD_B_0_01_V

For PCI9114DG:

Input Range (V)	Gain	Gain Code
±10 V	X 1	AD_B_10_V

±5 V	X 2	AD_B_5_V
±2.5 V	X 4	AD_B_2_5_V
±1.25 V	X 8	AD_B_1_25_V

@ Syntax

C/C++ (DOS)

U16 _9114_AD_Set_Range (U16 cardNo, U16 ADRange)

C/C++ (Windows 95)

U16 W_9114_AD_Set_Range (U16 cardNo, U16 ADRange)

Visual Basic (Windows 95)

W_9114_AD_Set_Range (ByVal cardNo As Integer, ByVal ADRange As Integer) As Integer

@ Argument

cardNo: The card number of PCI-9114 card initialized

ADRange: The programmable gain of A/D conversion, the possible values are: AD_B_10_V, AD_B_1_V, AD_B_0_1_V, AD_B_0_01_V, AD_B_5_V, AD_B_2_5_V, AD_B_1_25_V,

@ Return Code

ERR_NoError

6.2.12 _9114_AD_Get_Range

@ Description

This function is used to get the A/D range from the AD range control register. The following table specifies the relationship between the gain and input voltage ranges, please refer to the previous section for the possible range.

@ Syntax

C/C++ (DOS)

U16 _9114_AD_Get_Range (U16 cardNo, U16 *ADRange)

C/C++ (Windows 95)

U16 W_9114_AD_Get_Range (U16 cardNo, U16 *ADRange)

Visual Basic (Windows 95)

W_9114_AD_Get_Range (ByVal cardNo As Integer, ADRange As Integer) As Integer

@ Argument

cardNo: The card number of PCI-9114 card initialized

ADRange: The programmable gain of A/D conversion, the possible values are: AD_B_10_V, AD_B_1_V, AD_B_0_1_V, AD_B_0_01_V, AD_B_5_V, AD_B_2_5_V, AD_B_1_25_V

@ Return Code

ERR_NoError

6.2.13 _9114_AD_Get_Status

@ Description

This function is used to get AD FIFO status from the A/D status readback register.

@ Syntax

C/C++ (DOS)

U16 _9114_AD_Get_Status (U16 cardNo, U16 *ADStatus)

C/C++ (Windows 95)

U16 W_9114_AD_Get_Status (U16 cardNo, U16 *ADStatus)

Visual Basic (Windows 95)

W_9114_AD_Get_Status (ByVal cardNo As Integer, ADStatus As Integer) As Integer

@ Argument

cardNo: The card number of PCI-9114 card initialized

ADStatus: The status of AD FIFO. The AD FIFO status could be one of the following:
ADSTS_FF_EF : FIFO is empty
ADSTS_FF_HF : FIFO is half-full

ADSTS_FF_FF : FIFO is full, A/D data may have been loss
ADSTS_BUSY : AD is busy, A/D data is written into FIFO.

@ Return Code

ERR_NoError

6.2.14 _9114_AD_Set_Mode

@ Description

This function is used to set AD trigger mode. Please refer to section 5.1.3 for the detailed description of AD trigger modes.

@ Syntax

C/C++ (DOS)

U16 _9114_AD_Set_Mode (U16 cardNo, U16 ADMode)

C/C++ (Windows 95)

U16 W_9114_AD_Set_Mode (U16 cardNo, U16 ADMode)

Visual Basic (Windows 95)

W_9114_AD_Set_Mode (ByVal cardNo As Integer, ByVal ADMode As Integer) As Integer

@ Argument

cardNo: The card number of PCI-9114 card initialized
ADMode: The value of AD mode.
The mode could be one or a combination of the following modes:

A_9114_AD_FIFO_ENABLE
A_9114_AD_FIFO_DISABLE
A_9114_AD_TimerTrig
A_9114_AD_SoftTrig
A_9114_AD_AutoScan

@ Return Code

ERR_NoError

6.2.15 _9114_AD_Get_Mode

@ Description

This function is used to get AD mode from AD trigger mode control register. Please refer to section 5.1.3 for the detailed description of AD trigger modes.

@ Syntax

C/C++ (DOS)

U16 _9114_AD_Get_Mode (U16 cardNo, U16 *ADMode)

C/C++ (Windows 95)

U16 W_9114_AD_Get_Mode (U16 cardNo, U16 *ADMode)

Visual Basic (Windows 95)

W_9114_AD_Get_Mode (ByVal cardNo As Integer, ADMode As Integer) As Integer

@ Argument

cardNo: The card number of PCI-9114 card initialized

ADMode: The value of AD mode

The returned value could be one or a combination of the following modes:

A_9114_AD_FIFO_ENABLE
A_9114_AD_FIFO_DISABLE
A_9114_AD_TimerTrig
A_9114_AD_SoftTrig
A_9114_AD_AutoScan

@ Return Code

ERR_NoError

6.2.16 _9114_INT_Set_Reg

@ Description

This function is used to select the interrupt sources by writing data to interrupt control register. Please refer to section 4.9 to

learn how to set the interrupt control register.

@ Syntax

C/C++ (DOS)

U16 _9114_INT_Set_Reg (U16 cardNo, U16 INTC)

C/C++ (Windows 95)

U16 W_9114_INT_Set_Reg (U16 cardNo, U16 INTC)

Visual Basic (Windows 95)

W_9114_INT_Set_Reg (ByVal cardNo As Integer, ByVal INTC As Integer) As Integer

@ Argument

cardNo: The card number of PCI-9114 card initialized
INTC: The value written to the interrupt control register

@ Return Code

ERR_NoError

6.2.17 _9114_AD_Get_Reg

@ Description

This function is used to get the AD mode setting and interrupt control setting by reading data from the Interrupt control read back register. The settings returned are stored in INTC. Please refer to section 4.7 and section 4.9 for the detailed definition of each bit of the returned data.

@ Syntax

C/C++ (DOS)

U16 _9114_INT_Get_Reg (U16 cardNo, U16 *INTC)

C/C++ (Windows 95)

U16 W_9114_INT_Get_Reg (U16 cardNo, U16 *INTC)

Visual Basic (Windows 95)

W_9114_INT_Get_Reg (ByVal cardNo As Integer, INTC As Integer) As Integer

@ Argument

cardNo: The card number of PCI-9114 card initialized.
INTC: The value returned from the interrupt control register.

@ Return Code

ERR_NoError

6.2.18 _9114_Reset_FIFO

@ Description

The PCI-9114 A/D data are stored in the FIFO after conversion. This function is used to reset A/D FIFO. This function should be called before performing A/D conversion to clear the old data stored in the FIFO.

@ Syntax

C/C++ (DOS)

U16 _9114_Reset_FIFO (U16 cardNo)

C/C++ (Windows 95)

U16 W_9114_Reset_FIFO (U16 cardNo)

Visual Basic (Windows 95)

W_9114_Reset_FIFO (ByVal cardNo As Integer) As Integer

@ Argument

cardNo: The card number of PCI-9114 card initialized.

@ Return Code

ERR_NoError

6.2.19 _9114_AD_Soft_Trigger

@ Description

This function is used to trigger the A/D conversion by software. When this function is called, a trigger pulse will be generated and the converted data will be stored in data register.

@ Syntax

C/C++ (DOS)

U16 _9114_AD_Soft_Trigger (U16 cardNo)

C/C++ (Windows 95)

U16 W_9114_AD_Soft_Trigger (U16 cardNo)

Visual Basic (Windows 95)

W_9114_AD_Soft_Trigger (ByVal cardNo As Integer) As Integer

@ Argument

cardNo: The card number of PCI-9114 card initialized.

@ Return Code

ERR_NoError

6.2.20 _9114_Set_8254

@ Description

This function is used to write PCI-9114 8254 Programmable Timer.

@ Syntax

C/C++ (DOS)

U16 _9114_Set_8254 (U16 cardNo, U16 ChannelNo, U8 count)

C/C++ (Windows 95)

U16 W_9114_Set_8254 (U16 cardNo, U16 ChannelNo, U8 count)

Visual Basic (Windows 95)

W_9114_Set_8254 (ByVal cardNo As Integer, ByVal ChannelNo As Integer, ByVal count As Byte) As Integer

@ Argument

cardNo: The card number of PCI-9114 card initialized.
Tmr_ch: Port of 8254 Timer, the value is within 0 to 2.
Count : The counter value.

@ Return Code

ERR_NoError

6.2.21 _9114_Get_8254

@ Description

This function is used to read PCI-9114 8254 Programmable Timer. The read value is stored in count.

@ Syntax

C/C++ (DOS)

U16 _9114_Get_8254 (U16 cardNo, U16 ChannelNo, U8 *count)

C/C++ (Windows 95)

U16 W_9114_Get_8254 (U16 cardNo, U16 ChannelNo, U8 *count)

Visual Basic (Windows 95)

W_9114_Get_8254 (ByVal cardNo As Integer, ByVal ChannelNo As Integer, count As Byte) As Integer

@ Argument

cardNo: The card number of PCI-9114 card initialized.
Tmr_ch: Port of 8254 Timer, the value is within 0 to 2.
count : value read from 8254 programmable timer, only 8 LSBs are effective

@ Return Code

ERR_NoError

6.2.22 _9114_AD_Timer

@ Description

This function is used to set the Timer #1 and Timer#2. These timers are used as frequency dividers for generating constant A/D sampling rate dedicatedly. It is possible to stop the pacer trigger by setting any one of the dividers as 0. Since the AD conversion rate is limited due to the conversion time of the AD converter, the highest sampling rate of the PCI-9114 can

not be exceeded 100 KHz. Thus the multiplication of the dividers must be larger than 20.

@ Syntax

C/C++ (DOS)

U16 _9114_AD_Timer (U16 cardNo, U16 c1, U16 c2)

C/C++ (Windows 95)

U16 W_9114_AD_Timer (U16 cardNo, U16 c1, U16 c2)

Visual Basic (Windows 95)

W_9114_AD_Timer (ByVal cardNo As Integer, ByVal c1 As Integer, ByVal c2 As Integer) As Integer

@ Argument

cardNo: The card number of PCI-9114 card initialized.
c1: frequency divider of timer #1
c2: frequency divider of timer #2

@ Return Code

ERR_NoError

6.2.23 _9114_Counter_Start

@ Description

The counter #0 of the PCI-9114 Timer/Counter chip can be freely programmed by the users. This function is used to program the counter #0. This counter can be used as frequency generator if internal clock is used. It also can be used as event counter if external clock is used. All the 8254 modes (six operating modes) are available.

@ Syntax

C/C++ (DOS)

U16 _9114_Counter_Start (U16 cardNo, U16 mode, U16 c0)

C/C++ (Windows 95)

U16 W_9114_Counter_Start (U16 cardNo, U16 mode, U16 c0)

Visual Basic (Windows 95)

W_9114_Counter_Start (ByVal cardNo As Integer, ByVal mode As Integer, ByVal c0 As Integer) As Integer

@ Argument

cardNo: The card number of PCI-9114 card initialized.
Mode: the 8254 timer mode, the possible values are :
TIMER_MODE0, TIMER_MODE1,
TIMER_MODE2, TIMER_MODE3,
TIMER_MODE4, TIMER_MODE5.
Please refer to Counter/Timer 8254's reference manual for more detailed information of timer mode.
c0: counter value of counter#0

@ Return Code

ERR_NoError

6.2.24 _9114_Counter_Read

@ Description

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

@ Syntax

C/C++ (DOS)

U16 _9114_Counter_Read (U16 cardNo, U16 *c0)

C/C++ (Windows 95)

U16 W_9114_Counter_Read (U16 cardNo, U16 *c0)

Visual Basic (Windows 95)

W_9114_Counter_Read (ByVal cardNo As Integer, c0 As Integer) As Integer

@ Argument

cardNo: The card number of PCI-9114 card initialized.
c0: count value of counter#0

@ Return Code

ERR_NoError

6.2.25 _9114_Counter_Stop

@ Description

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

@ Syntax

C/C++ (DOS)

U16 _9114_Counter_Stop (U16 cardNo, U16 *c0)

C/C++ (Windows 95)

U16 W_9114_Counter_Stop (U16 cardNo, U16 *c0)

Visual Basic (Windows 95)

U16 W_9114_Counter_Stop (ByVal cardNo As Integer, c0 As Integer) As Integer

@ Argument

cardNo: The card number of PCI-9114 card initialized.
c0: the current counter value of the Counter#0

@ Return Code

ERR_NoError

6.2.26 _9114_INT_Source_Control

@ Description

PCI-9114 has a dual-interrupt system, therefore, two interrupt sources can be generated and be checked by the software. This function is used to select and control PCI-9114 interrupt sources by writing data to interrupt control register. Please refer to section 5.1.4 for detailed description of A/D data transfer modes.

@ Syntax

C/C++ (DOS)

void _9114_INT_Source_Control (U16 cardNo, U16 int1Ctrl, U16 int2Ctrl)

C/C++ (Windows 95)

void W_9114_INT_Source_Control (U16 cardNo, U16 int1Ctrl, U16 int2Ctrl)

Visual Basic (Windows 95)

W_9114_INT_Source_Control (ByVal cardNo As Integer, ByVal int1Ctrl As Integer, ByVal int2Ctrl As Integer)

@ Argument

cardNo: the card number of PCI-9114 card initialized.
int1Ctrl: the value to control INT1, the value can be set and the corresponding definition is the following:
int1Ctrl: 0 : INT1 disable
1 : INT1 AD end of conversion (EOC) interrupt
2 : INT1 FIFO half full
int2Ctrl: the value to control INT2, the value can be set and the corresponding definition is the following:
int2Ctrl: 0 : INT2 disable
1 : INT2 pacer timer interrupt
2 : INT2 external interrupt source

@ Return Code

None

6.2.27 _9114_CLR_IRQ1

@ Description

This function is used to clear interrupt request that is requested by PCI-9114INT1. If you use interrupt to transfer A/D converted data, you should use this function to clear interrupt request status, otherwise the new coming interrupt will not be generated.

@ Syntax

C/C++ (DOS)

void _9114_CLR_IRQ1 (U16 cardNo)

C/C++ (Windows 95)

void W_9114_CLR_IRQ1 (U16 cardNo)

Visual (Windows 95)

W_9114_CLR_IRQ1 (ByVal cardNo As Integer)

@ Argument

None

@ Return Code

None

6.2.28 _9114_CLR_IRQ2

@ Description

This function is used to clear interrupt request that is requested by PCI-9114INT2. If you use interrupt to transfer A/D converted data, you should use this function to clear interrupt request status, otherwise the new coming interrupt will not be generated.

@ Syntax

C/C++ (DOS)

void _9114_CLR_IRQ2 (U16 cardNo)

C/C++ (Windows 95)

void W_9114_CLR_IRQ2 (U16 cardNo)

Visual (Windows 95)

W_9114_CLR_IRQ2 (ByVal cardNo As Integer)

@ Argument

None

@ Return Code

None

6.2.29 _9114_Get_IRQ_Channel

@ Description

This function is used to get the IRQ level of the PCI-9114 card currently used.

@ Syntax

C/C++ (DOS)

void _9114_Get_IRQ_Channel (U16 cardNo, U16 *irq_no)

C/C++ (Windows 95)

void W_9114_Get_IRQ_Channel (U16 cardNo, U16 *irq_no)

Visual Basic (Windows 95)

W_9114_Get_IRQ_Channel (ByVal cardNo As Integer, irq_no As Integer)

@ Argument

cardNo: The card number of PCI-9114 card initialized.
Irq_no: The IRQ level used to transfer A/D data for this card

@ Return Code

None

6.2.30 _9114_Get_IRQ_Status

@ Description

This function is used to get the status of the two IRQs (INT1 and INT2) in PCI-9114 card.

@ Syntax

C/C++ (DOS)

void _9114_Get_IRQ_Status (U16 cardNo, U16 *ch1, U16 *ch2)

C/C++ (Windows 95)

void W_9114_Get_IRQ_Status (U16 cardNo, U16 *ch1, U16 *ch2)

Visual Basic (Windows 95)

W_9114_Get_IRQ_Status (ByVal cardNo As Integer, ch1 As Integer, ch2 As Integer)

@ Argument

cardNo: the card number of PCI-9114 card initialized.
ch1: the IRQ status of INT1
ch2: the IRQ status of INT2

@ Return Code

None

6.2.31 _9114_AD_FFHF_Polling

@ Description

This function is used to perform the powerful AD data transfer by applying half-full polling mode. This method checks the FIFO half full signal every time calling this function. If the FIFO is not half-full, the software does not read data. When the FIFO is full, the AD FIFO is overrun. When the FIFO is half-full but not full, software reads the A/D data, which is stored in FIFO, in size of one “block” (512 words). The FIFO half-full polling method is the most powerful A/D data transfer mode. Please refer to section 5.1.4 for the detailed description of half-full polling mode.

@ Syntax

C/C++ (DOS)

```
U16 _9114_AD_FFHF_Polling (U16 cardNo, I16 far
    *ad_buf)
```

C/C++ (Windows 95)

```
U16 W_9114_AD_FFHF_Polling (U16 cardNo, I16
    *ad_buf)
```

Visual Basic (Windows 95)

```
W_9114_AD_FFHF_Polling (ByVal cardNo As Integer,
    ad_buf As Integer) As Integer
```

@ Argument

cardNo: the card number of PCI-9114 card initialized.
ad_buf: the buffer stores the A/D converted value. The size of ad_buf can not be smaller than 512 words. The data

format can be referred to section 5.1.5 for the details.

@ Return Code

ERR_NoError
ERR_FIFO_Half_NotReady

6.2.32 _9114_AD_FFHF_Polling_MUX

@ Description

This function is used to perform powerful AD data transfer by applying half-full polling mode. This method checks the FIFO half full signal every time calling this function. If the FIFO is not half-full, the software does not read data. When the FIFO is full, the AD FIFO is overrun. When the FIFO is half-full and not full, software reads the A/D data, which is stored in FIFO, in size of one “block” (512 words). The difference between this function and 9114_AD_FFHF_Polling is in that the former reads data from the 16 bits register and the latter reads data from 32 bits data register. Please refer to section 5.1.4 for the detailed description of half-full polling mode.

@ Syntax

C/C++ (DOS)

U16 _9114_AD_FFHF_Polling_MUX (U16 cardNo, U32 far *ad_buf)

C/C++ (Windows 95)

U16 W_9114_AD_FFHF_Polling_MUX (U16 cardNo, U32 *ad_buf)

Visual Basic (Windows 95)

U16 W_9114_AD_FFHF_Polling_MUX (ByVal cardNo As Integer, ad_buf As Long) As Integer

@ Argument

cardNo: The card number of PCI-9114 card initialized.
ad_buf: The 32bits A/D converted value. The data format can be referred to section 5.1.5 for details.

@ Return Code

ERR_NoError
ERR_FIFO_Half_NotReady

6.2.33 _9114_AD_Aquire

@ Description

This function is used to poll the A/D converted data for PCI-9114 by software trigger. It reads the A/D data when the data is ready.

@ Syntax

C/C++ (DOS)

U16 _9114_AD_Aquire (U16 cardNo, I16 far *ad_data)

C/C++ (Windows 95)

U16 W_9114_AD_Aquire (U16 cardNo, I16 *ad_data)

Visual Basic (Windows 95)

W_9114_AD_Aquire (ByVal cardNo As Integer, ad_data As Integer) As Integer

@ Argument

cardNo: the card number of PCI-9114 card initialized.

ad_data: the 16bits A/D converted value. The bit 0 of ADData is the LSB of A/D converted data and the bit 15 of ADData is the MSB of A/D converted data. Please refer to section 5.1.5 for the relationship between the voltage and the value.

@ Return Code

ERR_NoError
ERR_AD_AquireTimeOut

6.2.34 _9114_AD_Aquire_MUX

@ Description

This function is used to poll the A/D conversion data for PCI-9114. It reads the A/D data when the data is ready.

@ Syntax

C/C++ (DOS)

```
U16 _9114_AD_Aquire_MUX ( U16 cardNo, U32  far
    *ad_data )
```

C/C++ (Windows 95)

```
U16 W_9114_AD_Aquire_MUX ( U16 cardNo, U32  far
    *ad_data )
```

Visual Basic (Windows 95)

```
W_9114_AD_Aquire_MUX (ByVal cardNo As Integer,
    ad_data As Long) As Integer
```

@ Argument

cardNo: the card number of PCI-9114 card initialized.
ad_data: the 32bits A/D converted value. The resolution of A/D conversion data is 16-bit. The unsigned integer data format of ADData is as follows:
Every 32-bit unsigned integer data:

bit 0…15: A/D converted data
bit 16, 17, …, 20 : converted channel no. The data format can be referred to section 5.1.5 for details.

@ Return Code

ERR_NoError
ERR_FIFO_Half_NotReady

6.2.35 _9114_AD_INT_Start

@ Description

This function is used to initial and startup the AD EOC (end-of-conversion) interrupt. This function could perform A/D conversion N times with interrupt data transfer by using pacer trigger. It takes place in the background and will not stop until the N-th conversion has been completed or your program

execute _9114_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 _9114_AD_INT_Status(). The function can perform on single A/D channel (autoscan is disabled) or multiple A/D channels (autoscan is enabled) with a fixed analog input range.

Note: The interrupt mode provided in this function is internal timer source, therefore you must specify c1 & c2 as calling this function. In addition, this function in DOS library supports just one PCI-9114 card and provides only one ISR (interrupt service routine) for processing the interrupt events.

@ Syntax

C/C++ (DOS)

```
U16 _9114_AD_INT_Start (U16 cardNo, U16 auto_scan,
    U16 ad_ch_no, U16 ad_gain, U16 count, U32
    *ad_buffer, U16 c1, U16 c2)
```

C/C++ (Windows 95)

```
U16 W_9114_AD_INT_Start (U16 cardNo, U16 auto_scan,
    U16 ad_ch_no, U16 ad_gain, U16 count, U32
    *ad_buffer, U16 c1, U16 c2)
```

Visual Basic (Windows 95)

```
W_9114_AD_INT_Start (ByVal cardNo As Integer, ByVal
    auto_scan As Integer, ByVal ad_ch_no As Integer,
    ByVal ad_gain As Integer, ByVal count As Integer,
    ad_buffer As Long, ByVal c1 As Integer, ByVal c2 As
    Integer) As Integer
```

@ Argument

cardNo: the card number of PCI-9114 card initialized.
auto_scan: 0: autoscan is disabled.
1: autoscan is enabled.
ad_ch_no: A/D channel number.
If the auto_scan is set as enabled, the selection sequence of A/D channel is: 0, 1, 2, 3, ..., [ad_ch_no], 0, 1, 2, 3, [ad_ch_no], ...

If the auto_scan is set as disabled, only the data input from [ad_ch_no] is converted.

ad_gain: A/D analog input range, the possible values are:
AD_B_10_V, AD_B_1_V, AD_B_0_1_V, AD_B_0_01_V, AD_B_5_V, AD_B_2_5_V, AD_B_1_25_V

count: the number of A/D conversion

ad_buffer: the start address of the memory buffer to store the AD data. The buffer size must large than the number of AD conversion. The unsigned integer data format in ad_buffer is as follows:
Every 32-bit unsigned integer data:

bit 0…15: A/D converted data
bit 16, 17, …, 20 : converted channel no. The data format can be referred to section 5.1.5 for details..

c1: the frequency divider of Timer#1.

c2: the frequency divider of Timer#2.

@ Return Code

ERR_InvalidADChannel
ERR_AD_InvalidGain
ERR_InvalidTimerValue
ERR_NoError

6.2.36 _9114_AD_FFHF_INT_Start

@ Description

This function is used to initial and start up the AD EOC (end-of-conversion) interrupt by using AD FIFO Half-Full Interrupt Transfer Mode. This function could perform A/D conversion N times with interrupt data transfer by using pacer trigger. It takes place in the background and will not stop until the N-th conversion has been completed or your program execute _9114_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 _9114_AD_FFHF_INT_Status(). The function can perform on

single A/D channel (autoscan is disabled) or multiple A/D channels (autoscan is enabled) with fixed analog input range.

Note: The interrupt mode provided in this function is internal timer source, therefore you must specify c1 & c2 as calling this function. In addition, this function in this library supports just one PCI-9114 card and provides only one ISR (interrupt service routine) for processing the interrupt events.

@ Syntax

C/C++ (DOS)

U16 _9114_AD_FFHF_INT_Start (U16 cardNo, U16 auto_scan, U16 ad_ch_no, U16 ad_gain, U16 blockNo, U32 *ad_buffer, U16 c1, U16 c2)

C/C++ (Windows 95)

U16 W_9114_AD_FFHF_INT_Start (U16 cardNo, U16 auto_scan, U16 ad_ch_no, U16 ad_gain, U16 blockNo, U32 *ad_buffer, U16 c1, U16 c2)

Visual Basic (Windows 95)

W_9114_AD_FFHF_INT_Start (ByVal cardNo As Integer, ByVal auto_scan As Integer, ByVal ad_ch_no As Integer, ByVal ad_gain As Integer, ByVal blockNo As Integer, ad_buffer As Long, ByVal c1 As Integer, ByVal c2 As Integer) As Integer

@ Argument

cardNo: the card number of PCI-9114 card initialized.
auto_scan: 0: autoscan is disabled.
1: autoscan is enabled.
ad_ch_no: A/D channel number.
If the auto_scan is set as enable, the selection sequence of A/D channel is: 0, 1, 2, 3, ..., [ad_ch_no], 0, 1, 2, 3, [ad_ch_no], ...
If the auto_scan is set as disable, only the data input from [ad_ch_no] is converted.
ad_gain: A/D analog input range, the possible values are:

AD_B_10_V, AD_B_1_V, AD_B_0_1_V,
AD_B_0_01_V, AD_B_5_V, AD_B_2_5_V,
AD_B_1_25_V.

blockNo: the number of blocks for performing A/D conversion, one block of A/D conversion is 512 words.

ad_buffer: the start address of the memory buffer to store the AD data. The buffer size must large than the number of AD conversion. The unsigned integer data format in ad_buffer is as follows:
Every 32-bit unsigned integer data:

bit 0…15: A/D converted data
bit 16, 17, …, 20 : converted channel no. The data format can be referred to section 5.1.5 for details.

c1: the frequency divider of Timer#1.
c2: the frequency divider of Timer#2.

@ Return Code

ERR_InvalidADChannel
ERR_AD_InvalidGain
ERR_InvalidTimerValue
ERR_NoError

6.2.37 _9114_AD_INT_Status

@ Description

This function is used to check the status of interrupt operation. The _9114_AD_INT_Start() is executed on background, therefore you can issue this function to check the status of interrupt operation.

@ Syntax

C/C++ (DOS)

U16 _9114_AD_INT_Status (U16 cardNo, U16 *status,

U16 *count)

C/C++ (Windows 95)

U16 W_9114_AD_INT_Status (U16 cardNo, U16 *status, U16 *count)

Visual Basic (Windows 95)

W_9114_AD_INT_Status (ByVal cardNo As Integer, status As Integer, count As Integer) As Integer

@ Argument

cardNo: the card number of PCI-9114 card initialized.
status: the status of the INT data transfer
count: the A/D conversion count number performed currently

@ Return Code

ERR_NoError

6.2.38 _9114_AD_FFHF_INT_Status

@ Description

This function is used to check the status of interrupt operation by using AD FIFO Half Full Interrupt Transfer Mode. The _9114_AD_FFHF_INT_Start() is executed on background, therefore you can issue this function to check the status of interrupt operation.

@ Syntax

C/C++ (DOS)

U16 _9114_AD_FFHF_INT_Status (U16 cardNo, U16 *status, U16 *blockNo)

C/C++ (Windows 95)

U16 W_9114_AD_FFHF_INT_Status (U16 cardNo, U16 *status, U16 *blockNo)

Visual Basic (Windows 95)

W_9114_AD_FFHF_INT_Status (ByVal cardNo As Integer, status As Integer, blockNo As Integer) As Integer

@ Argument

cardNo: the card number of PCI-9114 card initialized.
status: the status of the INT data transfer
blockno: the A/D conversion block number performed currently currently

@ Return Code

ERR_NoError

6.2.39 _9114_AD_FFHF_INT_Restart

@ Description

After calling _9114_AD_FFHF_INT_Start(), the AD conversion and transfer won't stop until the N blocks of the AD data is acquired, calling this function can restart the FIFO half full interrupt transfer without re-initial all the relative registers. However, if the interrupt operation was stopped by calling _9114_AD_FFHF_INT_Stop(), the program should use _9114_AD_FFHF_INT_Start() to restart the interrupt transfer function.

@ Syntax

C/C++ (DOS)

U16 _9114_AD_FFHF_INT_Restart (U16 cardNo)

C/C++ (Windows 95)

U16 W_9114_AD_FFHF_INT_Restart (U16 cardNo)

Visual Basic (Windows 95)

W_9114_AD_FFHF_INT_Restart (ByVal cardNo As Integer) As Integer

@ Argument

cardNo: the card number of PCI-9114 card initialized.

@ Return Code

ERR_NoError

6.2.40 _9114_AD_INT_Stop

@ Description

This function is used to stop the interrupt data transfer function. After executing this function, the internal AD trigger is disabled and the AD timer is stopped. This function returns the number of data has been transferred, no matter whether the AD interrupt data transfer is stopped by this function.

@ Syntax

C/C++ (DOS)

U16 _9114_AD_INT_Stop (U16 cardNo, U16 *count)

C/C++ (Windows 95)

U16 W_9114_AD_INT_Stop (U16 cardNo, U16 *count)

Visual Basic (Windows 95)

W_9114_AD_INT_Stop (ByVal cardNo As Integer, count As Integer) As Integer

@ Argument

CardNo: the card number of PCI-9114 card initialized.
count: the number of A/D data which has been transferred.

@ Return Code

ERR_AD_INTNotSet
ERR_NoError

7

Calibration & Utilities

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-9114 to an accuracy condition.

7.1 What do you need

Before calibrating your PCI-9114 card, you should prepare some equipments 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 multi-meter (a 6 1/2 meter is recommended)***
- ***A voltage calibrator or a very stable and noise free DC voltage generator.***

7.2 VR Assignment

There are three variable resistors (VR) on the PCI-9114 board to allow you making accurate adjustment on A/D channels. The function of each VR is specified in Table 7.1.

VR1	A/D bipolar offset adjustment
VR2	A/D full scale adjustment
VR3	Programmable Gain Amplifier offset adjustment
VR4	Cold junction sensor offset adjustment

Table 7.1 Function of the VRs

7.3 A/D Adjustment

7.3.1 PGA offset calibration

1. Set AD input signal type to single-ended (default) input.
2. Set Gain (AD range) register to desired value.
3. Adjust JP2 to set AD channel #16 to be grounded (Refer to section 2.4.2.) and program the card to use AD ch#16.
4. Use multi-meter to measure the voltage between TP1 and TP2.
5. Adjust ***VR3*** until the multi-meter value approach to zero.

7.3.2 Bipolar offset calibration

1. Set AD input signal type to single-ended (default) input.
2. Set Gain (AD range) register to desired value.
3. Use multi-meter to calibrate the external reference voltage to the lowest value of the specified AD range (e.g. If AD range is ±10V, the external reference voltage will be set as –10V.).
4. Connect the external reference voltage to AD channel #0.
5. Adjust ***VR1*** to obtain reading between 0x8000~0x8001.

7.3.3 Full range calibration

1. Set AD input signal type to single-ended (default) input.
2. Set Gain (AD range) register to desired value.
3. Use multi-meter to calibrate the reference voltage on user's

external voltage to the highest value of the specified AD range (e.g. If AD range is ±10V, the external reference voltage will be set as +10V.).

4. Connect the reference voltage to AD channel #0.
5. Adjust ***VR2*** to obtain reading between 0x7FFE~0x7FFF.

7.3.4 Cold junction sensor calibration

1. Set to use the C.J. sensor under SE input mode. (Refer to section 2.4.)
2. Measure the current temperature.
3. Adjust the ***VR4*** until the read-out value equal to the temperature value.
4. The relation between temperature and the read-out value is as following:

V (mV) = T (°K) x 10 (mV / °K)
T (°K) = V (mV) / 10(mV/ °K)
T (°C) = T (°K) - 273 (°K)

For example:

If the temperature is 25 °C, user should calibrate the CJC voltage to (25+273)x 10 = 2980 mV = 2.98 V

7.4 Software A/D Offset Calibration

For more accuracy calibrate the input offset signal, using software to calibrate the offset of the analog input signal is a good approach. Another benefit is this method can calibrate offset online and thus eliminate any temperature drift. For example, user can short the resistor RB32 to ground. Measuring the digital value of channel #31 can obtain the offset voltage of the AD channels. If the digital offset value is Voff, user can modify any AD data by subtracting Voff from the AD data to obtain the offset calibrated value. Note that the Voff may be different for each gain level,. Users should calibrate the offset value for every gain value.

8

Software Utility

This software CD provides two utility programs. They are 9114util.exe which provides three functions, System Configuration, Calibration, and Functional Testing, and I_eeprom which is used to enable or disable interrupt of PCI-9114 board. The utility programs are described in the following sections.

8.1 9114util

There are three functions provided by 9114util. They are System Configuration, Calibration, and Functional Testing. This utility software 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.

8.1.1 Running 9114util.exe

After finishing the DOS installation, you can execute the utility by typing as follows :

C> cd \ADLINK\9114\DOS\Util

C> 9114UTIL

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 the next step and change the default settings.

```
            ****** PCI-9114 Utility  Rev. 1.0 ******
Copyright © 1995-1996, 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. <<<
```

8.1.2 System Configuration

This function guides you to configure the PCI-9114 card, and set the right hardware configuration. The configuration window shows the setting items that you have to set before using the PCI-9114 card.

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

```
****** Calibration of PCI9114 ******

<1> Card Type                      PCI9114HG
<2> AD Polarity setting           Bipolar
<3> AD Input Range                 Gain=1  Bipolar(-10V~10V)

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

8.1.3 Calibration

This function guides you to calibrate the PCI-9114. The calibration program serves as a useful test of the PCI-9114's A/D and D/A 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-9114. 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-9114 Calibration ******

        <1> A/D Bipolar adjusting

        <2> PGA Offset adjusting

        <3> Cold junction sensor calibration

        <Esc> Quit

            Select 1 to 3 or <Esc> to quit calibration.
```

If you select 1, the following figure displays on the screen:

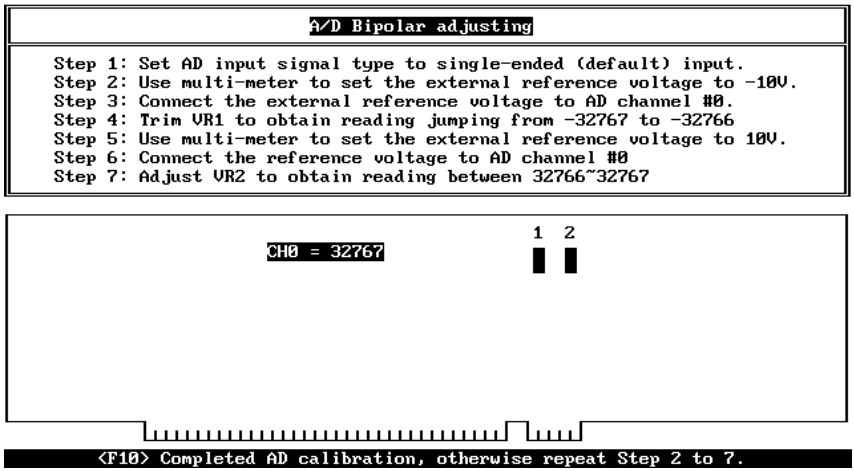


Fig 8.1. A/D Bipolar adjusting

8.1.4Funtional Testing

This function is used to test the functions of PCI-9114. It includes Digital I/O testing, D/A testing, A/D polling testing, A/D Interrupt Testing, and A/D FIFO Half-Full Interrupt testing.
When you choose one of the testing functions 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.

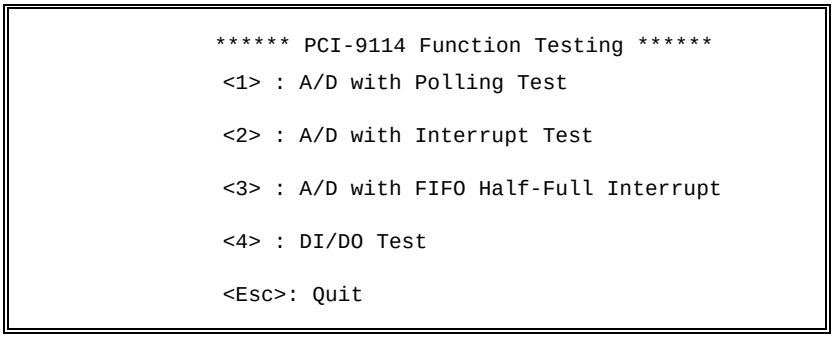


Fig 8.2 Function testing menu window

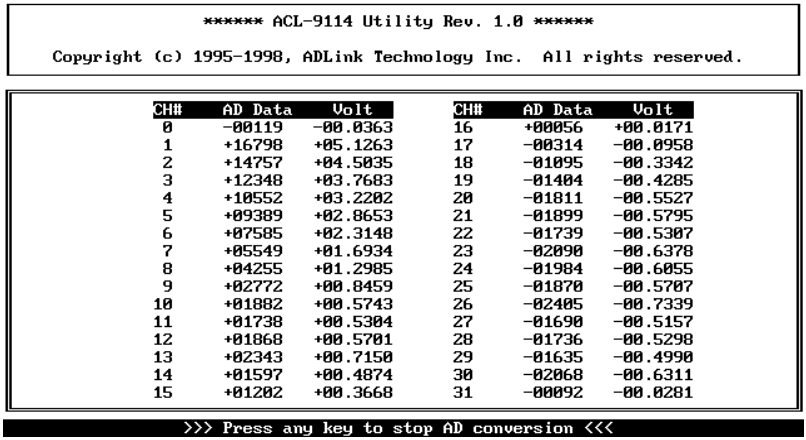


Fig 8.3 A/D with FIFO Half-Full Interrupt test window

8.2 I_EEPROM

This file is used to enable or disable the interrupt of PCI-9114 board. This software is a text-driven program. Because the default interrupt on PCI-9114 board is “on”, users who don’t want to use interrupt function can use this utility to turn off the interrupt of their PCI-9114 board.

8.2.1 Running I_eeprom.exe

After finishing the DOS installation, you can execute the utility by typing as follows :

C> cd \ADLINK\ 9114\DOS\UTIL

C> I_eeprom

At first, this program prompts you to input the card type—9114. After specifying the card type, this program shows the instructions to guide you to enable or disable the interrupt of your PCI-9114 board.

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.