EX-98354
Multi-functions Counter / Timer Card User's Guide

Acknowledgment

Products mentioned in this manual are mentioned for identification purpose only. Products manes appearing in this manual may or may not be registered trademarks or copyright of their respective companies

Printed Sep. 2002 Rev 1.0

Table of Contents

Chapter 1 Introduction

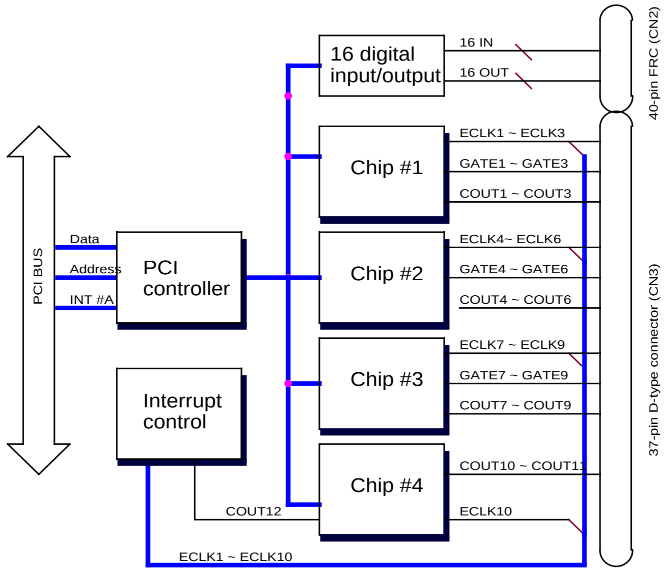
1.1 Introduction

EX-98354 is a general-purpose counter / timer and digital I/O card. This card have four 8254 chips on board, so it provides twelve 16 bits down counter or frequency dividers.

This card has multi-configurations. The counters can be set as independent counter or cascaded counter. The gate control of counter comes from either external source or internal default enable signal. The clock source of the counters can be set as internal or external clock source, user can set the jumper to decide whether the debounce function is used or not used. It is possible to use this card on variety of powerful counter / timer functions to match your industry and laboratory applications.

The card also provides digital output and input port. There are 8 bits digital output and 8 bits digital input channels that can be used to control or monitor the external devices.

EX-98354 provides one interrupt signal that comes from internal or external interrupt sources. The interrupt can be used for watchdog timer or others applications. The maximum interrupt time interval can be 536 sec.



Block diagram of the EX-98354

1.2 Features

- Four 8254 chips provide twelve 16 bits down counters
- Multi-configurations of counters / timers
- Flexible setting for each independent counter, the clock source
- Could be external, internal or cascaded. The gate signal is external controlled or internal enabled.
- Provide debounce function with flexible setting to prevent from bounce phenomenon when using external clock.
- 8 digital output channels
- 8 digital input channels
- Interrupt source comes from output of counter #12 or external source.
- 37–pin D-type female connector.
- Board card number

1.3 Applications

- Laboratory and Industrial automation
- Event counter
- Frequency generator
- Frequency synthesizer
- Pulse width measurement
- Testing & Measurement
- Laboratory & Education

1.4 Specifications

- **Programmable Counter / Timer**

  Device: 82C54x4

  Number of Counters /timers: 10 independent timers / counters cascaded 32-bit counters with fixed 8MHz internal clock

  Counter mode: 16-bit down counter

  Maximum input frequency: 10MHz

  Clock sources of independent counters:

  External clock

  Prior counter output

  CK1 (Programmable)

  Clock #10 output

  CK1 clock sources: (Programmable)

  8MHz internal base clock

  Programmable counter 11 output

  Gate control: default enable or external control

- **Digital Filter Circuits**

  De-bounce clock: (Programmable)

  8MHz internal base clock

  Programmable counter #11 output

- **Digital I/O (DIO)**

  Input channels: 16 channels

  Output channels: 16 channels (dedicated output)

  Electronics characteristics: TTL compatible signal

- **Environmental:**

  Power requirements: +5V 350mA (typical)

  Operation Temp: 0 to 50C˚

  Storage Temp: -20 to 70C˚

  Humidity: 0 to 90% none-condensing

  Dimensions: 180 mm x 105 mm

1.5 Software Supporting

Topsccc provides versatile software drivers and packages for users' different approach to built-up a system. We not only provide programming library such as DLL for many Windows systems, but also provide drivers for many software package such as LabVIEW™ ,Intouch™ and so on. All the software options are included in the provided CD.

1.6 Programming Library

The provided CD includes the function libraries for many different operating systems, including:

- **DOS Library:** Borland C/C++ and Microsoft C++, the functions descriptions are included in this user's guide.
- **Windows 98/2000/NT/Me/XP DLL:** For VB, VC++, BC5, the functions Descriptions are included in this user's guide.
- Windows 98/2000/NT/Me/XP ActiveX: For Windows's applications
- **LabVIEW ® Driver:** Contains the VIs, which are used to interface with NI's LabVIEW ® software package. Supporting Windows 95/98/NT/2000. The LabVIEW ® drivers are free shipped with the board.
- **InTouch Driver:** Contains the InTouch driver which support the Windows 98/2000/NT/XP. The The InTouch ® drivers are free shipped with the board.

Chapter 2 Installation

This chapter describes how to install the EX-98354 card. Please follow the follow steps to install the EX-98354 card.

2.1 What You Have

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

- EX-98354 board
- Driver/utilities CD
- This user's manual

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 EX-98354 card contains sensitive electronic components that can be easily damaged by static electricity. 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 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.

2.3 Hardware Installation Outline

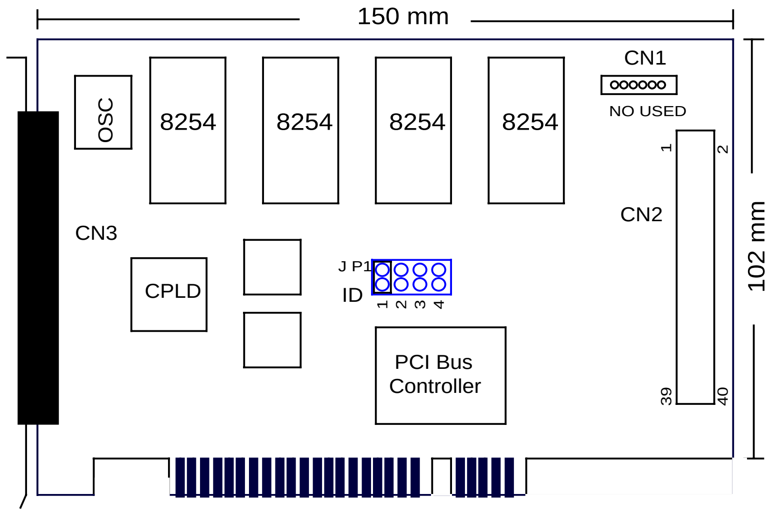
- **PCI configuration**

  The PCI cards are equipped with plug and play PCI controller, it can request base addresses and interrupt according to PCI standard. The system BIOS will install the system resource based on the PCI cards' configuration registers and system parameters (which are set by system BIOS). Interrupt assignment and memory usage (I/O port locations) of the PCI cards can be assigned by system BIOS only. These system resource assignments are done on a board-by-board basis. It is not suggested to assign the system resource by any other methods.

- **PCI slot selection**

  The PCI card can be inserted to any PCI slot without any configuration for system resource.

2.4 PCB Layout



CN3: Timer/counter channel #1 ~ channel #12

CN: Digital input (DI_0 ~ DI_15) and digital output (DO_0 ~ DO_15)

CN1: Reserved for testing only

JP1: Card number jumper

2.5 Installation Procedures

1. **Turn off your computer.**
2. **Turn off all accessories (printer, modem, monitor, etc.) connected to your computer.**
3. **Remove the cover from your computer.**
4. **Setup jumpers on the card.**
5. **Before handling the PCI cards, discharge any static buildup on your body by touching the metal case of the computer. Hold the edge and do not touch the components.**
6. **Position the board into the PCI slot you selected.**
7. **Secure the card in place at the rear panel of the system.**

2.6 Device Installation for Windows Systems

Once Windows 95/98/2000 has started, the Plug and Play function of Windows system will find the new Expert cards. If this is the first time to install Expert cards in your Windows system, you will be informed to input the device information source.

2.7 Connector Pin Assignment of EX-98354

There are two connectors labeled “CN2” and “CN3”. The CN3 connector is a 37-pin D-type connector and CN2 connector is a 40-pin FRC connector.

The pin assignment of the CN3 (37-pins D-type connector) includes 12 timer/counter input and output signals

- **The CN3 pin assignment is as shown in Figure 2-1**

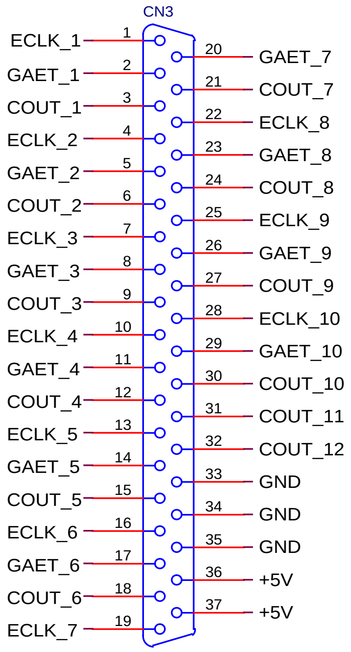


Figure 2-1 Pin Assignment of EX-98354 connector CN3

Legend:

ECLK_n: Clock input channel #n(n= 1 ~ 10)

GATE_n: Gate control input channel #n (n=1 ~ 10)

COUT_n: Count output channel #n (n=1 ~ 12)

:GND: Ground return path of counter/time channels

+5V: Non-isolated +5V DC output

- **The CN2 pin assignment is as shown in Figure 2-2**

The CN2 is a 40-pin FRC connector and can be converted to 37-pin D-type connector by using converting cable attached in the package.

The pin assignment of the CN2 (40-pin FRC connector) includes 16 digital input and output signals

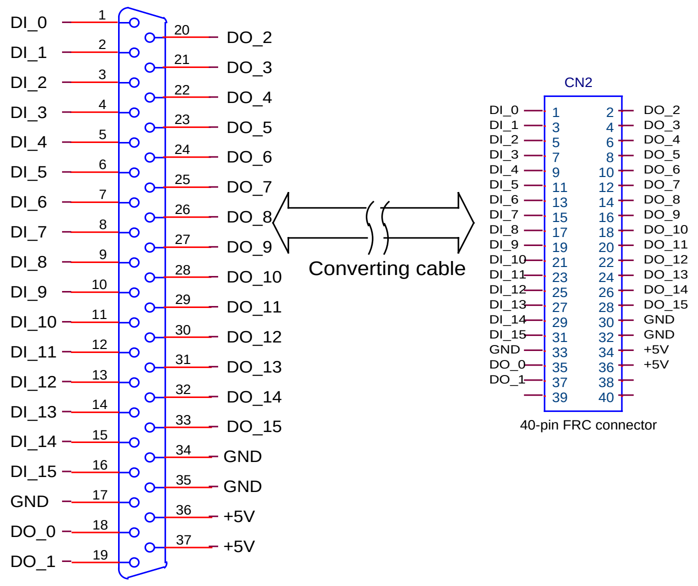


Figure 2-2 Pin Assignment of EX-98354 connector CN2

Legend:

DO_n: Digital output channel #n (n= 0 ~ 15)

DI_n: Digital input channels #n (n= 0 ~ 15)

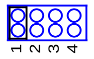
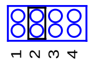
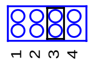
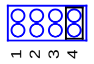
GND: Ground return path of digital input/output channels

+5V: Non-isolated +5V DC output

2.8 Card number setting

Maximum four EX-98354 cards can be installed in system simultaneously with each has a unique card number.

A jumper called "JP1" (see page 9) on the card is used to set the card number starts from 1 to 4

JP1	Card number
	1 (default setting)
	2
	3
	4

Chapter 3 Registers Format

This information is quite useful for the programmers who wish to handle the card by low-level programming. However, we suggest user have to understand more about the PCI interface then start any low-level programming. In addition, the contents of this chapter can help users understand how to use software driver to manipulate this card.

3.1 PCI PnP Registers

There are two types of registers: PCI Configuration Registers (PCR) and Peripheral Interface Bus (PIB). The PCR, which is compliant to the PCI-bus specifications, is initialized and controlled by the plug & play (PnP) PCI BIOS..

The PCI bus controller Tiger 100/320 is provided by Tigerjet Network Inc. (www.tjnet.com). For more detailed information of PIB, please visit Tigerjet technology's web site to download relative information. It is not necessary for users to understand the details of the PIB if you use the software library. The PCI PnP BIOS assigns the base address of the PIB. The assigned address is located at offset 14h of PIB .

The 94264 board registers are in 32-bit width. But only lowest byte (bit0~bit7) is used. The users can access these registers by only 32-bit I/O or 8-bit I/O instructions. The following sections show the address map, including descriptions and their offset addresses relative to the base address.

3.2 PCI controller register address map

- **Reset control register**

The EX-98354 is in inactive state when the system power on, and should be activated by set bit o of this register to "1" state

Address: Base + 0x00h

Attribute: Write only

Value: 01

- **PCI Internal special control register**

EX-98354 internal control register, should be written with value 01H before controlling EX-98354 card

Address: Base + 002h

Attribute: Write only

Value: always are 01h

- **Interrupt mask control register**

Enable or disable PCI interrupt INT #A

Address: Base + 0x05h

Attribute: Write only

Value: 10H =enable PCI INT A#

00H=disable PCI INT #A

- **PCI controller internal Aux port data (AUX0~AUX7) register**

Enable/disable Timer/counter or Digital I/O port address decoder

Address: Base + 0x03h

Attribute: Write only

Value: 00H =enable Timer/counter port decoder (*AUX0=0*)

01H=enable Digital I/O port decoder (*AUX0=1*)

3.3 Timer/counter register

Timer/counter registers

Address: Base + 0C0H~0FCH and AUX0=1

(*Write 01H to Base+03H before accessing Base + 0C0H~0FCH ports*)

Attribute: Write/read

Value:

Port address	AUX0	Function
Base+0C0H	1	Counter 1 Register (R/W)
Base+0C4H	1	Counter 2 Register (R/W)
Base+0C8H	1	Counter3 Register (R/W)
Base+0CCH	1	Mode Control Register (W) Read Back Register (R)
Base+0D0H	1	Counter4 Register (R/W)
Base+0D4H	1	Counter 5 Register (R/W)
Base+0D8H	1	Counter 6 Register (R/W)
Base+0DCH	1	Mode Control Register (W) Read Back Register (R)
Base+0E0H	1	Counter 7 Register (R/W)
Base+0E4H	1	Counter 8 Register (R/W)
Base+0E8H	1	Counter 9 Register (R/W)
Base+0ECH	1	Mode Control Register (W) Read Back Register (R)
Base+0F0H	1	Counter 10 Register (R/W)
Base+0F4H	1	Counter 11 Register (R/W)
Base+0F8H	1	Counter 12 Register (R/W)
Base+0FCH	1	Mode Control Register (W) Read Back Register (R)

- **Timer CLK source /Interrupt mode control registers**

There are total twenty-two bits on EX-98354 to select clock source of Timer / Counter #1 ~ #10 and **CK1** and debounce clock and interrupt source.

Address: Base + 0F0H~0F8H and AUX=0

(*Write 00H to Base+03H before accessing Base + 0F0H~0F8H ports*)

Attribute: Write/read

Value:

■ Base + 0F0H and AUX=0 (timer #1~time #4 CLK source register)

Bit no.	Timer/counter	Bit value			
		0,0	0,1	1,0	1,1
1,0	Timer #1 CLK(CLK_1)	ECLK_1	GND	CK1	COUT_10
3,2	Timer #2 CLK(CLK_2)	ECLK_2	COUT_1	CK1	COUT_10
5,4	Timer #3 CLK(CLK_3)	ECLK_3	COUT_2	CK1	COUT_10
7,6	Timer #4 CLK(CLK_4)	ECLK_4	COUT_3	CK1	COUT_10

■ Base + 0F4H and AUX=0 (timer #5~time #8 CLK source register)

Bit no.	Timer/counter	Bit value			
		0,0	0,1	1,0	1,1
1,0	Timer #5 CLK(CLK_5)	ECLK_5	COUT_5	CK1	COUT_10
3,2	Timer #6 CLK(CLK_6)	ECLK_6	COUT_6	CK1	COUT_10
5,4	Timer #7 CLK(CLK_7)	ECLK_7	COUT_7	CK1	COUT_10
7,6	Timer #8 CLK(CLK_8)	ECLK_8	COUT_8	CK1	COUT_10

■ Base + 0F8H and AUX=0 (timer #9~time #10CLK source and debouce time and interrupt mode control registers)

Bit no.	Timer/counter	Bit value			
		0,0	0,1	1,0	1,1
1,0	Timer #9 CLK(CLK_9)	ECLK_5	COUT_5	CK1	COUT_10
3,2	Timer #19 CLK(CLK_10)	ECLK_6	COUT_6	CK1	COUT_10
4	CK1 source	=08M OSC		=1COUT11	
5	DI_0 interrupt	=0Disable		=1Enable	
6	Debounce time	=02 usec		=1 COUT11x4	
7	COUT_12 interrupt	=0Disable		=1Enable	

-

- **Interrupt status register**

Read interrupt status (Timer #12 COUT_12 and/or digital input DI_0)

Address: Base + 0DCH and AUX=0

(*Write 00H to Base+03H before accessing Base + 0DCH port*)

Attribute: Read only

Bit no.	Interrupt source	Interrupt status	
		Interrupt	No interrupt
0	Digital input DI_0	1	0
1	Timer #12 COUT_12	1	0
2	xx	xx	xx
3	xx	xx	xx
4	xx	xx	xx
5	xx	xx	xx

- **Debounce control registers**

There are total eleven bits on EX-98354 to enable or disable debounce function clock source of Timer / Counter #1 ~ #10 and **DI_0**

Address: Base + 0D0H ~ Base + 0D4H and AUX=0

(*Write 00H to Base+03H before accessing Base +0D0H and Base+0D4H ports*)

Attribute: Write only

Value:

■ Base + 0D0H and AUX=0 (timer #1~time #8 CLK debounce register)

Bit no.	Timer ECLK debounce	Bit value	
		Disable	Enable
0	Timer #1 ECLK_1	0	1
1	Timer #2 ECLK_2	0	1
2	Timer #3 ECLK_3	0	1
3	Timer #4 ECLK_4	0	1
4	Timer #5 ECLK_5	0	1
5	Timer #6 ECLK_6	0	1
6	Timer #7 ECLK_7	0	1
7	Timer #8 ECLK_8	0	1

- Base + 0D4H and AUX=0 (timer #9~time #10 CLK and DI_0 debounce register)

Bit no.	Timer ECLK debounce	Bit value	
		Disable	Enable
0	Timer #9 ECLK_9	0	1
1	Timer #10 ECLK_10	0	1
2	Digital input DI_0	0	1
3	xx	xx	xx
4	xx	xx	xx
5	xx	xx	xx
6	xx	xx	xx
7	xx	xx	xx

3.4 Digital Input/Output Register Address Map

There are 16 TTL digital inputs and 16 outputs on EX-98354; each bit of based address is corresponding to a signal on the digital input or output channel.

Address: Base+0C0H~0CCH and AUX=0

Attribute: Base+0C0H~Base+0C4H read only

Base+0C8H~Base+0CCH write only,

Value:

Base+0C0H =Digital input port #0 (DI_0 ~DI_7), read only

Base+0C4H =Digital input port #1 (DI_8~DI_15), read only

Base+0C8H =Digital output port #0 (DO_0 ~DO_7), write only

Base+0CCH =Digital output port #1 (DO_8~DO_15), write only

Chapter 4 Operation Theorem

4.1 Clock System

The clock of counter / timer #1 ~ #10 can be one of the 4 sources: **external clock source** or **cascaded source** from the 'last' channel or **CK1** (8MHz OSC or COUT11) or **COUT10**. The clock of counter / timer #11 is fixed at 8Mhz, and clock of counter / timer #12 is connected to **COUT11**

4.2 Counters architecture

There are four 8254 chips on EX-98354 card. Counter #1~ #10 can be programming to independent or cascaded counters and counters #11and #12 are **cascaded counters**.

There are three signals (2 input, 1 output) for each counter, a clock input signal, a gate control signal, and an output signal. ***CLK1*** ~ ***CLK12*** are clock sources and ***GATE1*** ~ ***GATE12*** are gate control signals. The ***COUT1*** ~ ***COUT12*** are output of the counters. The Figure 5-3 shows all the labels and the inter-connection of the 8254 counters.

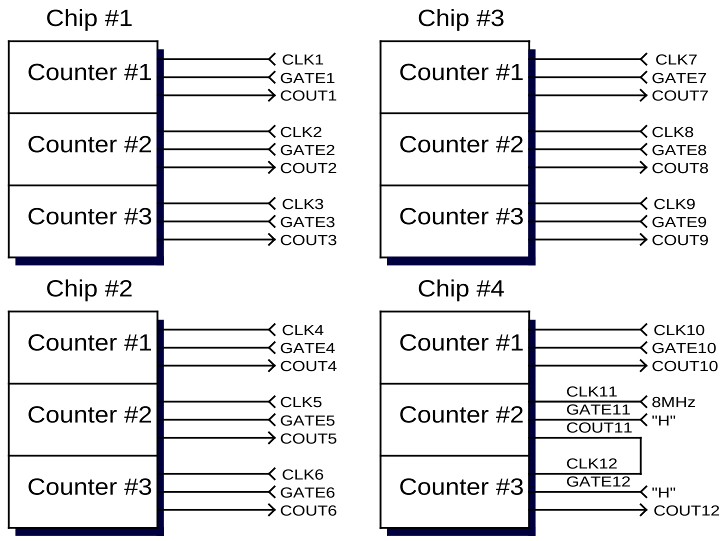


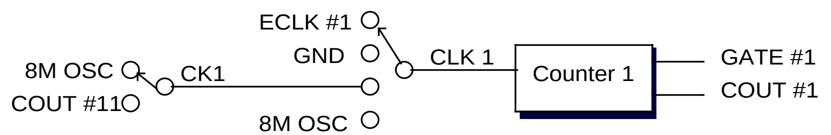
Figure 5-3 Counters architectural

4.3 Clock Source Configurations

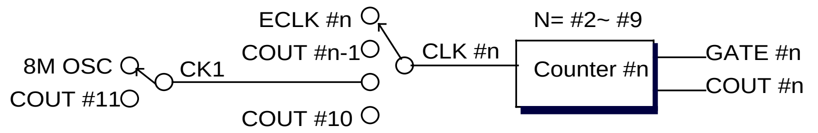
The counter #1~ #10 can be programming to independent or cascaded counters and the clock source of each counter can be ***ECLK #n*** (external CLK input), ***COUT #n-1*** (Previous counter output), ***CK1*** (8MHz or COUT #11) or ***8M OSC***(Fixed 8MHz) by using function (W_8354_SetCntlCLK and "W_8354_SetCK1)

4.3.1 Independent Counters (Counter #1~#10)

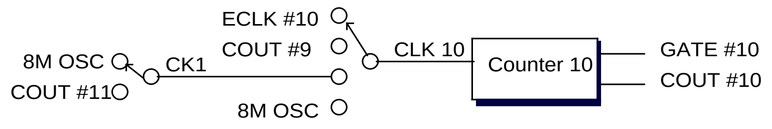
For Timer/counter #1



For Timer/counter #2~#9



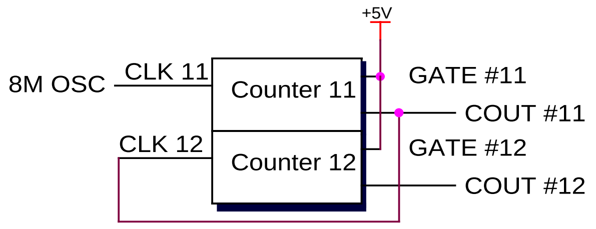
For Timer/counter #10



Note: If counter #1 is set to cascaded mode, **CLK1** is connected to GND because **COUT #0** doesn't exist.)

4.3.2 Cascaded counters (Counter #11~#12)

The connection of Counter #11 and #12 are different from other independent counters. These two counters are named as cascaded counters. The clock source fo counter #11 comes from fixed 8 **MHz** and its output are cascaded to counter #12. In fact, counter #11and #12 are designed for frequency divider by using 8254's square wave generator mode. The gates of these counters always keep at 'H' level for enabling counters all the time.



Note:

1. That the signals ***COUT #12*** can also be used as interrupt source. See page
2. Although there is one set cascaded counter on board, users may need more cascaded counters. User can set the clock source of every independent counter by program. Therefore, the independent counter output can be cascaded to the next counter's clock source to implement cascaded counter.
3. An example of 'user programmable cascaded counters (counter #1 and counter #2) as shown in Figure 4-4

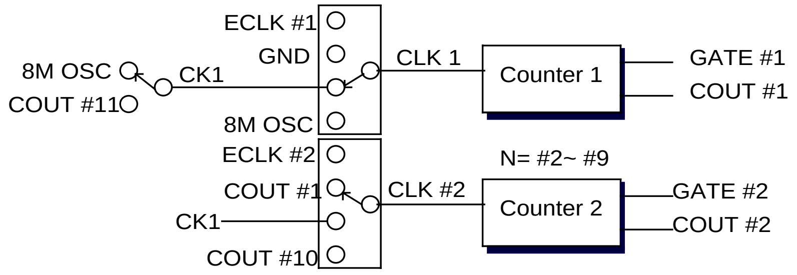
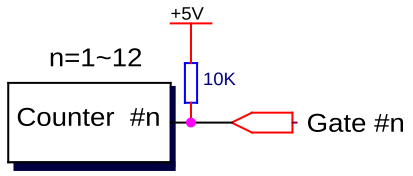


Figure 4-4

4.4 Gate control configurations

The gate control signals of the counters are internally pulled high hence they are default enable if no external gate used.



4.5 Counter outputs

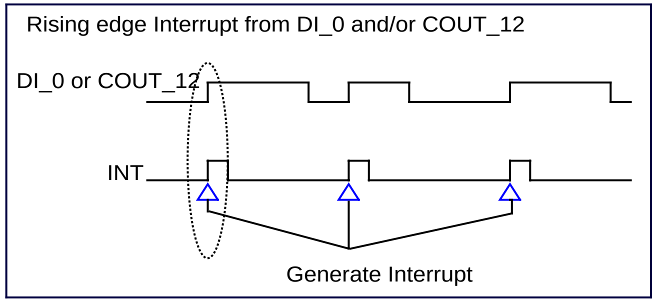
The all timer / counter output signals ***(COUT_*** n) of 8254 are sent to the 37-pin D-type connector directly, please see 'Pin assignment' for corresponding signal pin number.

In addition, the output signal may be used as clock source for cascaded counters, see the above sections. It is possible to cascaded ten counters by software setting,

The counters output ***COUT12*** is also used as internal interrupt source

4.6 Interrupt System

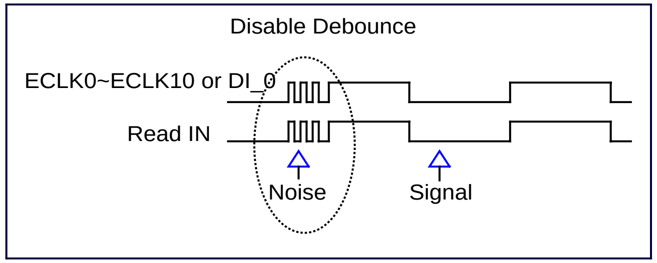
The EX-98354 is a *Dual Interrupt Syste*m. The dual interrupt(DI_0 and COUT_12) means the hardware can generate two interrupt request signals at the same time and the software can service these two request signals by Interrupt Status Register

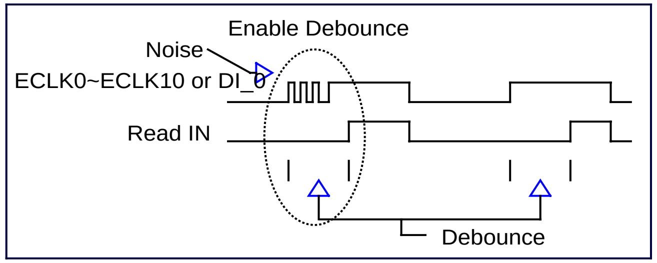


Note: There is only one IRQ level used by this card. This card uses INT #A interrupt request signal to PCI bus that means the DI_0 and COUT_12 use the same interrupt level.

4.7 Digital debounce

Debounce system is used to eliminate bounce phenomenon. If external clock is used, user can select if debounce system is used or not used by software. If debounce system is used, the debounce output signal will be the same state as the input only if the input signal keep the same state for four **debounce clocks**, otherwise the input signal will be treated as glitch and the debounce output signal will keep previous state,





The following is a functional description of the digital debounce.

1. When a digital debounce is enabled, the EX-98354 will sample the signals at the enabled input channel at a 8usec (or COUT_12*4) sampling rate.
2. When a high or low signal is present at a digital input channel whose digital debounce function is enabled, the signal will be filtered out as noise unless it lasts for an effective period.
3. Effective period = 8 usec (or COUT_12 period*4)

4.8 Digital Input and output

Each digital input or output (DI_0~DI_15 and DO_0~DO_15) is a TTL structure. The input /output voltage range form 0V to 5V and input pull-up resister is 10K ohms for all digital inputs. The connection between outside signal and EX-98354 digital inputs is shown in Fig Figure 4-5.

(All the digital outputs can be read back by the same I/O address)

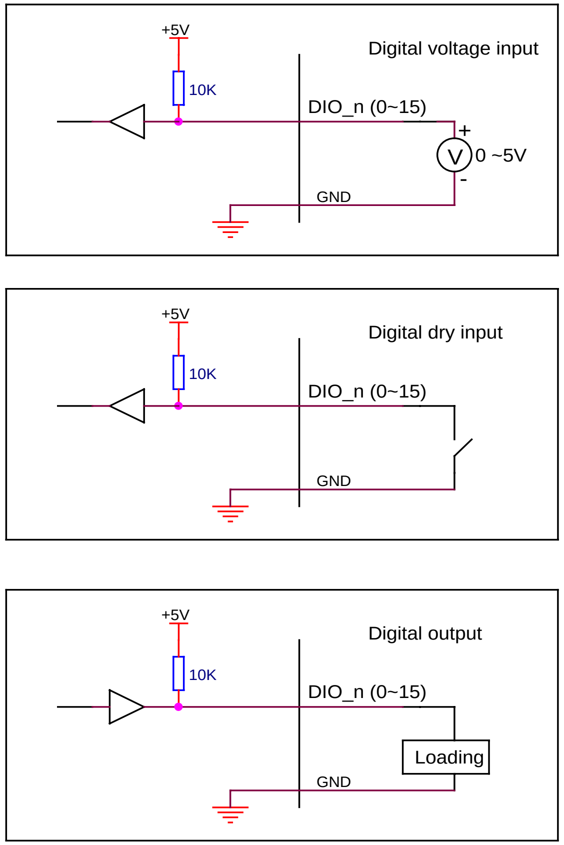


Figure 4-5

Chapter 5 Libraries

This chapter describes the software library for operating this card. Only the functions in DOS library and Windows 98/2000 DLL are described. Please refer to the PCIDAQ function reference manual, which included in Topsccc CD for the descriptions of the Windows 98/NT/2000 DLL functions.

5.1 Libraries Installation

The device drivers and DLL functions of Windows 98/NT/2000 are included in the PCIDAQ. The Topsccc CD also includes the detail examples and readme files

5.2 How to use the Functions in PCIDAQ.DLL

- **VC++6.0:**
  1. Add file '../Include/PCIDAQ.H' in your project
  2. In link page of menu project| setting, add '../LIB/PCIDAQ.LIB' in the blank of Objects/Library Modules
  3. Add this sentence "#include '../Include/PCIDAQ.H' " to the head of your main file.
- **Visual BASIC:**
  1. Add file '../Include/Declare.bas' in your project.
- **Delphi :**
  1. Add file '../Include/Declare.pas' in your project
  2. Add this sentence "uses Declare;" in the head of your unit.pas
- **C++Builder:**
  1. Add file '../Include/PCIDAQ.H' and '../Lib/PCIDAQ_CB.lib' to your project
  2. Add this sentence "#include '../Include/PCIDAQ.H' " to head of your main file.

Note: For more information, please refer to program in directory '../Example/'

5.3 Summary of function calls

Function	Description	Page
W_8354_Open	Initial EX-98354 card before using other functions	28
W_8354_GetCardsID	Get EX-98354 card number	29
W_8354_Version	Get version number of PCIDAQ.DLL	30
W_8354_GetBusSlot	Get PCI bus and slot number occupied by EX-98354	31
W_8354_Close	Close EX-98354 card before terminating program	32
W_8354_Read_Di	Read digital input port data (8-bit)	33
W_8354_Write_Do	Write data (8-bit) to digital output port	34
W_8354_Set_Do_Bit	Set a bit of port to high	35
W_8354_Reset_Do_Bit	Reset a bit of port to low	36
D_8354_Set_Counter	Initial timer/counter	37
W_8354_Set_CLKSource	Select timer/counter clock source	38
W_8354_Write_Counter	Write timer/counter work mode and value	39
W_8354_Read_Counter	Read timer/counter work mode and value	40
W_8354_Stop_Counter	Stop timer/counter	41
W_8354_Set_CK1	Select CK1 clock source	42
W_8354_Set_DebounceCLK	Set debounce clock source	43
W_8354_Set_DebounceMode	Set debounce mode	44
D_8354_Read_IntStatus	Read interrupt status register (DOS only)	45
W_8354_Clear_IntStatus	Clear interrupt status register	46
W_8354_IntEnable	Enable digital input change interrupt	47
W_8354_IntDisable	Disable digital input interrupt	48

5.4 W_8354_Open

Description:

Because the EX-98354 is PCI bus architecture and meets the plug and play design, the IRQ and base_address (pass-through address) are assigned by system BIOS directly. EX-98354 cards have to be initialized by this function before calling other functions.

Syntax:

C/C++ (DOS)

```
WORD D_8354_Open (WORD cardNo);
```

C/C++ (Windows)

```
WORD D_8354_Open (WORD *ExistCards);
```

Visual BASIC (Windows)

```
W_8354_Open (ByRef ExitedCards As Long) As Long
```

Delphi

W_8354_Open (var ExistedCards:Integer): Integer;

Argument:

CardNo: card number (1,2,3,4) (for DOS only)

existCards: The number of installed EX-98354 cards. (for Windows only)

This returned value shows how many EX-98354 cards are installed in your system.

Return Code:

Error code (Please refer to PCIDAQ.H or DOSDAQ.H)

5.5 W_8354_GetCardsID

Description:

Get the cards number that is set by jumper on cards.

Syntax:

C/C++(DOS)

```
void D_8354_GetCardsID(WORD *CardsIDArray);
```

C/C++(Windows)

```
WORD W_8354_GetCardsID (WORD *CardsIDArray);
```

Visual BASIC (Windows)

```
Function W_8354_GetCardsID (ByRef CardsIDArray As Long)
         As Integer
```

Delphi

```
Function W_8354_GetCardsID (var CardsIDArray:Word):Word;
```

Argument:

CardsIDArray: This array return card number (1,2,3,4), which is set by jumper on the card. You should define a 4 elements array, then pass the array's pointer to this function.

Return Code:

Error code (Please refer to PCIDAQ.H or DOSDAQ.H)

5.6 W_8354_Version

Description:

PCIDAQ.DLL driver drives the EX-98354 cards. This function returns the version of PCIDAQ.DLL driver

Syntax:

C/C++ (DOS)

```
void D_8354_Version (char *version)
```

C/C++ (Windows)

```
WORD D_8354_Version (void)
```

Visual BASIC (Windows)

```
Function W_8354_Version () As Long
```

Delphi

```
Function W_8354_Version ():Integer
```

Argument:

version: return the PCIDAQ.DLL driver version string (DOS only)

Return Code:

The version of PCIDAQ.DLL in integer data format (Windows only)

5.7 W_8354_GetBusSlot

Description:

Get the PCI bus and slot occupied by EX-98354

Syntax:

C/C++ (DOS)

```
WORD D_8354_GetBusSlot (WORD cardNo, WORD *bus,WORD *slot);
```

C/C++ (Windows)

```
W_8354_GetBusSlot (WORD cardNo, WORD *bus,WORD *slot);
```

Visual BASIC (Windows)

```
Function W_8354_GetBusSlot (ByVal cardNo As Long,
         ByRef bus As Long, ByRef slot As Long) As Long
```

Delphi

```
Function W_8354_GetBusSlot (cardNo:Integer;var
         portNo:Integer;var bitNo:Integer): Integer;
```

Argument:

cardNo: card number (1,2,3,4),It's set by jumper on card

Bus: return PCI bus Number

Slot: return PCI slot Number of the bus

Return Code:

Error code (Please refer to PCIDAQ.H or DOSDAQ.H)

5.8 W_8354_Close

Description:

The IRQ and base_address of EX-98354 (pass-through address) are assigned by system BIOS directly. This function should be called to release all system resource before terminate application program

Syntax:

C/C++ (DOS)

```
WORD D_8354_Close (WORD cardNo)
```

C/C++ (Windows)

```
W_8354_Close (void)
```

Visual BASIC (Windows)

```
Function W_8354_Close ()
```

Delphi

```
Function W_8354_Close ();
```

Argument:

CardNo: card number (1,2,3,4) (DOS only)

All EX-98354 cards are closed after calling this function (Windows only)

Return Code:

None

5.9 W_8354_Read_Di

Description:

This function is used to read data from digital input port. There are two 8-bit digital inputs on the EX-98354. You can get 8-bit input data from EX-98354 by calling this function.

Syntax:

C/C++ (DOS)

```
WORD D_8354_Read_Di (WORD cardNo,WORD portNo,WORD *DiData)
```

C/C++ (Windows)

```
W_8354_Read_Di (WORD cardNo,WORD portNo,WORD *DiData)
```

Visual BASIC (Windows)

```
Function W_8354_Read_Di (ByVal cardNo As Long,
         ByVal portNo As Long, ByRef DiData As Long)
         As Long
```

Delphi

```
Function W_8354_Read_Di (cardNo:Integer;portNo:Integer;
         var DoData:Integer):Integer;
```

Argument:

cardNo: Card number (1,2,3,4),It's set by jumper on card

portNo: Digital Input port number (0 or 1)

DiData: return digital input data

Return Code:

Error code (Please refer to PCIDAQ.H or DOSDAQ.H)

5.10 W_8354_Write_Do

Description:

This function is used to write data to output port. There are two 8-bit digital outputs port on the EX-98354. You can send 8-bit output data toEX-98354 by calling this function.

Syntax:

C/C++ (DOS)

```
WORD D_8354_Write_Do (WORD cardNo,WORD portNo,WORD Data);
```

C/C++ (Windows)

```
WORD W_8354_Write_Do (WORD cardNo,WORD portNo,WORD Data);
```

Visual BASIC (Windows)

```
Function W_8354_Write_Do (ByVal cardNo As Long,
         ByVal portNo As Long, ByVal Data As Long)
         As Long
```

Delphi

```
Function W_8354_Write_Do (cardNo:Integer;portNo:Integer;
         Data:Integer): Integer;
```

Argument:

cardNo: card number (1,2,3,4),It's set by jumper on card

portNo: relay output port number (0 or 1)

Data: Data be written to output port

Return Code:

Error code (Please refer to PCIDAQ.H or DOSDAQ.H)

5.11 W_8354_Set_Do_Bit

Description:

Set a digital output channel HIGH

Syntax:

C/C++ (DOS)

```
WORD D_ 8354_Set_Do_Bit (WORD cardNo,WORD portNo, WORD
     bitNo);
```

C/C++ (Windows)

```
W_ 8354_Set_Do_Bit (WORD cardNo,WORD portNo, WORDbitNo);
```

Visual BASIC (Windows)

```
Function W_ 8354_Set_Do_Bit Bit (ByVal cardNo As Long,
         ByVal portNo As Long,ByVal bitNo As Long) As Long
```

Delphi

```
Function W_ 8354_Set_Do_Bit (cardNo:Integer;portNo:Integer;
         bitNo:Integer): Integer;
```

Argument:

cardNo: card number (1,2,3,4),It's set by jumper on card

portNo: relay output port number (0 or 1)

bitNo: channel number(0 to 7)

Return Code:

Error code (Please refer to PCIDAQ.H or DOSDAQ.H)

5.12 W_8354_Reset_Do_Bit

Description:

Set a digital output channel LOW

Syntax:

C/C++ (DOS)

```
WORD D_ 8354_Reset_Do_Bit (WORD cardNo,WORD portNo,
     WORD bitNo);
```

C/C++ (Windows)

```
WORD W_8354_ 8354_Reset_Do_Bit (WORD cardNo,WORD portNo,
     WORD bitNo);
```

Visual BASIC (Windows)

```
Function W_ 8354_Reset_Do_Bit (ByVal cardNo As Long,
         ByVal portNo As Long, ByVal bitNo As Long)
         As Long
```

Delphi

```
Function W_ 8354_Reset_Do_Bit
         (cardNo:Integer;portNo:Integer; bitNo:Integer):
         Integer;
```

Argument:

cardNo: card number (1,2,3,4),It's set by jumper on card

portNo: relay output port number (0 or 1)

bitNo: channel Number(0 to 7)

Return Code:

Error code (Please refer to PCIDAQ.H or DOSDAQ.H)

5.13 D_8354_Set_Counter

Description:

User can directly initial counter #1 ~ #10 by this function. Using this function, user can assign the counter number 1~10 directly without care about the chips number and other details.

Syntax:

C/C++ (DOS)

```
WORD D_8354_Set_Counter (WORD cardNo, WORD cntNo,
     WORD mode, WORD cntrVal, WORD ClkSource);
```

C/C++ (Windows)

```
WORD W_8354_Set_Counter (WORD cardNo, WORD cntNo,
     WORD mode, WORD cntrVal,WORD ClkSource);
```

Visual BASIC (Windows)

```
Function W_8354_Set_Counter (ByVal cardNo As Long, ByVal
         cntNo As Long, ByVal mode As Long, ByVal cntrVal As
         Long, ByVal ClkSource As Long) As Long
```

Delphi

```
Function W_8354_Set_Counter (cardNo :Integer; cntNo :Integer;
         mode :Integer; cntrVal : Integer; ClkSource:
         Integer):Integer;
```

Argument:

cardNo: card number (1,2,3,4),It's set by jumper on card

cntNo : Counter Number(1~10) (COUT_1~COUT_10)

mode: Work mode of the counter (0~5)

cntrVal: initial value of counter (2~65535)

ClkSource: select counter clock source

0: external clock

1: counter (COUT_ cntNo -1) output clock

Available for time#2~timer #10, grounded for Time #1

Example: counter #3 clock source (CLK_3) = counter #2 clock output COUT_2)

2: CK1: 8MHZ or counter #11 clock output (COUT_11)

3: Counter #10 clock output (COUT_10)

Return Code:

Error code (Please refer to DOSDAQ.H)

5.14 W_8354_Set_CLKSource

Description:

Set timer/counter's (timer #1~timer #10) CLK source. User can call this function any time to change timer's clock source after calling *W_8354_Set_Counter ()* function without re-initialing the timer's mode and counting value

Syntax:

C/C++ (DOS)

```
WORD D_8354_Set_CLKSource (WORD cardNo, WORD cntNo,WORD
     ClkSource);
```

C/C++ (Windows)

```
WORD W_8354_Set_CLKSource (WORD cardNo, WORD cntNo, WORD
     ClkSource);
```

Visual BASIC (Windows)

```
Function W_8354_Set_CLKSource (ByVal cardNo As Long,
         ByVal cntNo As Long, ByVal ClkSource As Long)
         As Long
```

Delphi

```
Function W_8354_Set_CLKSource (cardNo :Integer;
         cntNo :Integer; ClkSource: Integer):Integer;
```

Argument:

cardNo: card number (1,2,3,4),It's set by jumper on card

cntNo: Counter Number(1~10)

ClkSource: select counter clock source

=0: external clock

=1: counter (COUT_ cntNo -1) output clock

Available for time#2~timer #10, grounded for Time #1

Example:counter #3 clock source(CLK_3) = counter #2 clock output COUT_2)

=2: CK1: 8MHZ or counter #11 clock output (COUT_11)

=3: Counter #10 clock output (COUT_10)

Return Code:

Error code (Please refer to PCIDAQ.H or DOSDAQ.H)

5.15 W_8354_Write_Counter

Description:

Set counter's work mode and initial value

Syntax:

C/C++ (DOS)

```
WORD D_8354_Write_Counter (WORD cardNo, WORD cntNo,
     WORD mode, WORD cntrval);
```

C/C++ (Windows)

```
WORD W_8354_Write_Counter (WORD cardNo, WORD cntNo,
     WORD mode, WORD cntrVal);
```

Visual BASIC (Windows)

```
Function W_8354_Write_Counter (ByVal cardNo As Long,
         ByVal cntNo As Long, ByVal mode As Long,
         ByVal cntrVal As Long) As Long
```

Delphi

```
Function W_8354_Write_Counter (cardNo :Integer;
         cntNo :Integer; mode :Integer; cntVal:
         Integer):Integer;
```

Argument:

cardNo: card number (1,2,3,4),It's set by jumper on card

cntNo: Counter Number(1~12)

mode: Work mode of the counter (0~5)

cntrVal: initial value of counter (0~65535)

Return Code:

Error code (Please refer to PCIDAQ.H or DOSDAQ.H)

5.16 W_8354_Read_Counter

Description:

Read counter's work mode and initial value

Syntax:

C/C++ (DOS)

```
WORD D_8354_Read_Counter (WORD cardNo,WORD cntNo,
     WORD *mode, WORD *cntval);
```

C/C++ (Windows)

```
WORD W_8354_Read_Counter (WORD cardNo,WORD cntNo,
     WORD *mode, WORD *cntrVal);
```

Visual BASIC (Windows)

```
Function W_8354_Read_Counter (ByVal cardNo As Long,
         ByVal cntNo As Long, ByRef mode As Long,
         ByRef cntrVal As Long) As Long
```

Delphi

```
Function W_8354_Read_Counter (cardNo :Integer;
         cntNo :Integer; var mode :Integer;
         var cntrVal:Integer):Integer;
```

Argument:

cardNo: card number (1,2,3,4),It's set by jumper on card

cntNo: counter number(1~12)

mode: return Work mode of the counter (0~5)

cntrVal: return current value of counter (0~65535)

Return Code:

Error code (Please refer to PCIDAQ.H or DOSDAQ.H)

5.17 W_8354_Stop_Counter

Description:

Stop counter by writing work mode 5 to timer

Syntax:

C/C++ (DOS)

```
WORD D_8354_Stop_Counter (WORD cardNo,WORD cntNo,
     WORD *cntrval );
```

C/C++ (Windows)

```
WORD W_8354_Stop_Counter (WORD cardNo,WORD cntNo,
     WORD *cntrVal );
```

Visual BASIC (Windows)

```
Function W_8354_Stop_Counter (ByVal cardNo As Long,
         ByVal cntNo As Long, ByRef cntrVal As Long)
         As Long
```

Delphi

```
Function W_8354_Stop_Counter (cardNo :Integer;
         cntNo :Integer; var cntrVal: Integer):Integer;
```

Argument:

cardNo: card number (1,2,3,4),It's set by jumper on card

cntNo: counter number(1~12)

cntrVal: return current value of counter (0~65535)

Return Code:

Error code (Please refer to PCIDAQ.H or DOSDAQ.H)

5.18 W_8354_Set_CK1

Description:

Set CK1's source to 8MHz or COUT_11 (see detail information in page 21)

Syntax:

C/C++ (DOS)

```
    WORD D_8354_Set_CK1 (WORD cardNo, WORD selCK1);
```

C/C++ (Windows)

```
WORD W_8354_Set_CK1 (WORD cardNo, WORD selCK1);
```

Visual BASIC (Windows)

```
Function W_8354_Set_CK1 (ByVal cardNo As Long,
         ByVal selCK1 As Long) As Long
```

Delphi

```
Function W_8354_Set_CK1 (cardNo :Integer;
         selCK1:Integer):Integer;
```

Argument:

cardNo: card number (1,2,3,4),It's set by jumper on card

SelCK1: source of CK1 (0= 8MHz, 1= COUT_11)

Return Code:

Error code (Please refer to PCIDAQ.H or DOSDAQ.H)

5.19 W_8354_Set_DebounceCLK

Description:

set debounce time to 8 usec or COUT_11*4

Syntax:

C/C++ (DOS)

```
WORD D_8354_Set_DebounceCLK (WORD cardNo, WORD DBCLK);
```

C/C++ (Windows)

```
WORD W_8354_Set_DebounceCLK (WORD cardNo, WORD DBCLK);
```

Visual BASIC (Windows)

```
Function W_8354_Set_DebounceCLK (ByVal cardNo As Long,
         ByVal DBCLK As Long) As Long
```

Delphi

```
Function W_8354_Set_DebounceCLK (cardNo :Integer;
         DBCLK:Integer):Integer;
```

Argument:

cardNo : card number (1,2,3,4),It's set by jumper on card

DBCLK: debounce time (0=8 usec , 1=COUT_11*4)

Return Code:

Error code (Please refer to PCIDAQ.H or DOSDAQ.H)

5.20 W_8354_Set_DebounceMode

Description:

Enable or disable debounce function of ECLKs (ECLK_1~ECLK_10) and/or digital input channel #0 (DI_0)

Syntax:

C/C++(DOS)

```
WORD D_8354_Set_DebounceMode (WORD cardNo,WORD DBMode);
```

C/C++ (Windows)

```
WORD W_8354_Set_DebounceMode (WORD cardNo, WORD DBMode);
```

Visual BASIC (Windows)

```
Function W_8354_Set_DebounceMode (ByVal cardNo As Long,
         ByVal DBMode As Long) As Long
```

Delphi

```
Function W_8354_Set_DebounceMode (cardNo :Integer;
         DBMode: Integer): Integer;
```

Argument:

cardNo : card number (1,2,3,4),It's set by jumper on card

DBMode:

Bit 0 =1/0 Enable/Disable ECLK_1 debounce

Bit 1=1/0 Enable/Disable ECLK_2 debounce

Bit 2=1/0 Enable/Disable ECLK_3 debounce

Bit 3=1/0 Enable/Disable ECLK_4 debounce

Bit 4=1/0 Enable/Disable ECLK_5 debounce

Bit 5=1/0 Enable/Disable ECLK_6 debounce

Bit 6=1/0 Enable/Disable ECLK_7 debounce

Bit 7=1/0 Enable/Disable ECLK_8 debounce

Bit 8=1/0 Enable/Disable ECLK_9 debounce

Bit 9=1/0 Enable/Disable ECLK_10 debounce

Bit 10=1/0 Enable/Disable DI_0 debounce

Return Code:

Error code (Please refer to PCIDAQ.H or DOSDAQ.H)

5.21 D_8354_Read_IntStatus

Description:

Get the interrupt status (for DOS only)

Syntax:

C/C++ (DOS)

```
WORD D_8354_Read_IntStatus (WORD cardNo,WORD *IntStatus)
```

Argument:

cardNo: card number (1,2,3,4), it set by jumper on the card

IntStatus: return PCI interrupt status.

if bit 0 = 1,interrupted by channel #0 (DI_0)

if bit 1 = 1,interrupted by Timer #12 (COUT_12)

Return Code:

Error code (Please refer to DOSDAQ.H)

5.22 W_8354_Clear_IntStatus

Description:

Clear interrupt by writing data to Base Port+0xC8. You should use this function to clear interrupt request status, otherwise the new coming interrupt will not be generated.

Syntax:

C/C++ (DOS)

```
WORD D_8354_Clear_IntStatus (WORD cardNo);
```

C/C++ (Windows)

```
W_8354_Clear_IntStatus (WORD cardNo);
```

Visual BASIC (Windows)

```
Function W_8354_Clear_IntStatus (ByVal cardNo As Long) As
         Long
```

Delphi

```
Function W_8354_Clear_IntStatus (cardNo:Integer):Integer;
```

Argument:

cardNo: card number (1,2,3,4),It's set by jumper on card

Return Code:

Error code (Please refer to PCIDAQ.H or DOSDAQ.H)

5.23 W_8354_IntEnable

Description:

Enable Interrupt of channel #0 (DI_0) and/or timer #12 clock out (COUT_12)

Syntax:

C/C++ (DOS)

```
WORD D_8354_IntEnable (WORD cardNo,WORD Int1Mode,WORD
     Int2Mode, User_Interrupt_HANDLER
     userIntServiceRoutine);
```

C/C++ (Windows)

```
W_8354_IntEnable (WORD cardNo,WORD Int1Mode,WORD Int2Mode,
     User_Interrupt_HANDLER userIntServiceRoutine);
```

Visual BASIC (Windows)

```
Function W_8354_IntEnable (ByVal cardNo As Long, ByVal
         Int1Mode As Long, ByVal Int2Mode As Long, ByVal
         userIntServiceRoutine As Long) As Long
```

Delphi

```
Function W_8354_IntEnable (cardNo:Integer;Int1Mode:Integer;
         Int2Mode:Integer; userIntServiceRoutine:Pointer):
         Integer;
```

Argument:

cardNo: card number (1,2,3,4),It's set by jumper on card

Int1Mode	=1 Enable digital input channel #0 (DI_0) interrupt
	=0 Disable digital input channel #0 (DI_0) interrupt
Int2Mode	=1 Enable Timer #12 (COUT_12) interrupt
	=0 Disable Timer #12 (COUT_12) interrupt

userIntServiceRoutine: userIntServiceRoutine: user Interrupt service routine called when interrupt occurs.

The kernel interrupt handle will pass card number and interrupt status to userIntServiceRoutine (WORD cardNo,WORD intstatus) (please see detail information in attached demo program)

Return Code:

Error code (Please refer to PCIDAQ.H or DOSDAQ.H)

5.24 W_8354_IntDisable

Description:

Disable interrupt of digital input channel #0 (DI_0) and Timer #12 (COUT_12)

Syntax:

C/C++ (DOS)

```
WORD W_8354_IntDisable (WORD cardNo);
```

C/C++ (Windows)

```
W_8354_IntDisable (WORD cardNo);
```

Visual BASIC (Windows)

```
Function W_8354_IntDisable (ByVal cardNo As Long)
```

Delphi

```
Function W_8354_IntDisable (cardNo:Integer);
```

Argument:

cardNo: card number (1,2,3,4),It's set by jumper on card

Return Code:

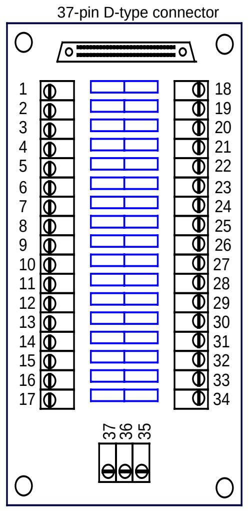
Error code (Please refer to PCIDAQ.H or DOSDAQ.H)

Chapter 6
EX-9837 Terminal board

EX-9837 Screw-terminal termination board features one 37-pin D-type connector for easy maintenance, wiring, and installation. It provides 37 channels that are accessed through a 37-pin D-type connector.

6.1 Main features

- Low-cost screw-terminal board for the all Expert series with 37-pin D-type connector
- Reserved space for signal-conditioning circuits such as low-pass filter, voltage attenuator and current shunt
- Industrial type termination blocks permit heavy-duty and reliable signal connections
- Table-top mounting using nylon standoffs. Screws and washers provided for panel or wall mounting
- Dimensions: 80mm (W) x 181mm (H)


37-pin D-type connector
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
37
36
35