DAQ

6527 Register-Level Programmer Manual

January 2001 Edition
Part Number 370106A-01

Worldwide Technical Support and Product Information

ni.com

National Instruments Corporate Headquarters

11500 North Mopac Expressway Austin, Texas 78759-3504 USA Tel: 512 794 0100

Worldwide Offices

Australia 03 9879 5166, Austria 0662 45 79 90 0, Belgium 02 757 00 20, Brazil 011 284 5011,
Canada (Calgary) 403 274 9391, Canada (Ottawa) 613 233 5949, Canada (Québec) 514 694 8521,
China (Shanghai) 021 6555 7838, China (ShenZhen) 0755 3904939, Denmark 45 76 26 00,
Finland 09 725 725 11, France 01 48 14 24 24, Germany 089 741 31 30, Greece 30 1 42 96 427,
Hong Kong 2645 3186, India 91805275406, Israel 03 6120092, Italy 02 413091, Japan 03 5472 2970,
Korea 02 596 7456, Mexico 5 280 7625, Netherlands 0348 433466, New Zealand 09 914 0488,
Norway 32 27 73 00, Poland 0 22 528 94 06, Portugal 351 1 726 9011, Singapore 2265886, Spain 91 640 0085,
Sweden 08 587 895 00, Switzerland 056 200 51 51, Taiwan 02 2528 7227, United Kingdom 01635 523545

For further support information, see the *Technical Support Resources* appendix. To comment on the documentation, send e-mail to techpubs@ni.com

Important Information

Warranty

The 6527 is warranted against defects in materials and workmanship for a period of one year from the date of shipment, as evidenced by receipts or other documentation. National Instruments will, at its option, repair or replace equipment that proves to be defective during the warranty period. This warranty includes parts and labor.

The media on which you receive National Instruments software are warranted not to fail to execute programming instructions, due to defects in materials and workmanship, for a period of 90 days from date of shipment, as evidenced by receipts or other documentation. National Instruments will, at its option, repair or replace software media that do not execute programming instructions if National Instruments receives notice of such defects during the warranty period. National Instruments does not warrant that the operation of the software shall be uninterrupted or error free.

A Return Material Authorization (RMA) number must be obtained from the factory and clearly marked on the outside of the package before any equipment will be accepted for warranty work. National Instruments will pay the shipping costs of returning to the owner parts which are covered by warranty.

National Instruments believes that the information in this document is accurate. The document has been carefully reviewed for technical accuracy. In the event that technical or typographical errors exist, National Instruments reserves the right to make changes to subsequent editions of this document without prior notice to holders of this edition. The reader should consult National Instruments if errors are suspected. In no event shall National Instruments be liable for any damages arising out of or related to this document or the information contained in it.

EXCEPT AS SPECIFIED HEREIN, NATIONAL INSTRUMENTS MAKES NO WARRANTIES, EXPRESS OR IMPLIED, AND SPECIFICALLY DISCLAIMS ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. CUSTOMER'S RIGHT TO RECOVER DAMAGES CAUSED BY FAULT OR NEGLIGENCE ON THE PART OF NATIONAL INSTRUMENTS SHALL BE LIMITED TO THE AMOUNT THERETOFORE PAID BY THE CUSTOMER. NATIONAL INSTRUMENTS WILL NOT BE LIABLE FOR DAMAGES RESULTING FROM LOSS OF DATA, PROFITS, USE OF PRODUCTS, OR INCIDENTAL OR CONSEQUENTIAL DAMAGES, EVEN IF ADVISED OF THE POSSIBILITY THEREOF. This limitation of the liability of National Instruments will apply regardless of the form of action, whether in contract or tort, including negligence. Any action against National Instruments must be brought within one year after the cause of action accrues. National Instruments shall not be liable for any delay in performance due to causes beyond its reasonable control. The warranty provided herein does not cover damages, defects, malfunctions, or service failures caused by owner's failure to follow the National Instruments installation, operation, or maintenance instructions; owner's modification of the product; owner's abuse, misuse, or negligent acts; and power failure or surges, fire, flood, accident, actions of third parties, or other events outside reasonable control.

Copyright

Trademarks

WARNING REGARDING USE OF NATIONAL INSTRUMENTS PRODUCTS

(1) NATIONAL INSTRUMENTS PRODUCTS ARE NOT DESIGNED WITH COMPONENTS AND TESTING FOR A LEVEL OF RELIABILITY SUITABLE FOR USE IN OR IN CONNECTION WITH SURGICAL IMPLANTS OR AS CRITICAL COMPONENTS IN ANY LIFE SUPPORT SYSTEMS WHOSE FAILURE TO PERFORM CAN REASONABLY BE EXPECTED TO CAUSE SIGNIFICANT INJURY TO A HUMAN.

(2) IN ANY APPLICATION, INCLUDING THE ABOVE, RELIABILITY OF OPERATION OF THE SOFTWARE PRODUCTS CAN BE IMPAIRED BY ADVERSE FACTORS, INCLUDING BUT NOT LIMITED TO FLUCTUATIONS IN ELECTRICAL POWER SUPPLY, COMPUTER HARDWARE MALFUNCTIONS, COMPUTER OPERATING SYSTEM SOFTWARE FITNESS, FITNESS OF COMPILERS AND DEVELOPMENT SOFTWARE USED TO DEVELOP AN APPLICATION, INSTALLATION ERRORS, SOFTWARE AND HARDWARE COMPATIBILITY PROBLEMS, MALFUNCTIONS OR FAILURES OF ELECTRONIC MONITORING OR CONTROL DEVICES, TRANSIENT FAILURES OF ELECTRONIC SYSTEMS (HARDWARE AND/OR SOFTWARE), UNANTICIPATED USES OR MISUSES, OR ERRORS ON THE PART OF THE USER OR APPLICATIONS DESIGNER (ADVERSE FACTORS SUCH AS THESE ARE HEREAFTER COLLECTIVELY TERMED "SYSTEM FAILURES"). ANY APPLICATION WHERE A SYSTEM FAILURE WOULD CREATE A RISK OF HARM TO PROPERTY OR PERSONS (INCLUDING THE RISK OF BODILY INJURY AND DEATH) SHOULD NOT BE RELIANT SOLELY UPON ONE FORM OF ELECTRONIC SYSTEM DUE TO THE RISK OF SYSTEM FAILURE. TO AVOID DAMAGE, INJURY, OR DEATH, THE USER OR APPLICATION DESIGNER MUST TAKE REASONABLY PRUDENT STEPS TO PROTECT AGAINST SYSTEM FAILURES, INCLUDING BUT NOT LIMITED TO BACK-UP OR SHUT DOWN MECHANISMS. BECAUSE EACH END-USER SYSTEM IS CUSTOMIZED AND DIFFERS FROM NATIONAL INSTRUMENTS' TESTING PLATFORMS AND BECAUSE A USER OR APPLICATION DESIGNER MAY USE NATIONAL INSTRUMENTS PRODUCTS IN COMBINATION WITH OTHER PRODUCTS IN A MANNER NOT EVALUATED OR CONTEMPLATED BY NATIONAL INSTRUMENTS, THE USER OR APPLICATION DESIGNER IS ULTIMATELY RESPONSIBLE FOR VERIFYING AND VALIDATING THE SUITABILITY OF NATIONAL INSTRUMENTS PRODUCTS WHENEVER NATIONAL INSTRUMENTS PRODUCTS ARE INCORPORATED IN A SYSTEM OR APPLICATION, INCLUDING, WITHOUT LIMITATION, THE APPROPRIATE DESIGN, PROCESS AND SAFETY LEVEL OF SUCH SYSTEM OR APPLICATION.

Contents

About This Manual

Chapter 1
Getting to Know Your 6527

Chapter 2
Register Map and Descriptions

Chapter 3
Programming

About This Manual

This manual contains the following information you need to perform register-level programming for your PCI-6527 and PXI-6527 digital I/O (DIO) devices:

- The address and function of each 6527 device register for reading data, writing data, and configuring filters on the input lines
- Examples that show the programming steps necessary to execute an operation
- Instructions for using change notification to generate interrupts on changing input data

Use change notification feature *only* if you are familiar with writing, installing, and uninstalling interrupt service routines. This manual does not cover writing, installing, and uninstalling interrupt service routines.

If you will be programming using an application development environment such as LabVIEW, Measurement Studio, or NI-DAQ, you do not need to read this manual.

Note National Instruments strongly recommends using application development environment software, such as LabVIEW, Measurement Studio, or NI-DAQ to program your 6527 device. Application software provides easier programming with the same flexibility as register-level programming.

How To Use the Manual Set

The *6527 Register-Level Programmer Manual* is one piece of the documentation set for your data acquisition system. You could have any of several types of manuals, depending on the hardware and software in your system. Use the manuals you have as follows:

- Your *6527 User Manual*—This manual leads you through installing, making connections to, and using your 6527 devices safely. It provides 6527 device specifications. Programming options other than register-level programming are also described here.
- Accessory installation guides or manuals—If you are using accessory products, consult these guides when you are making your connections. The terminal block and cable assembly installation guides or accessory board user manuals explain how to physically connect the relevant pieces of your system.

Conventions

The following conventions are used in this manual:

<>	Angle brackets that contain numbers separated by an ellipsis represent a range of values associated with a bit or signal name—for example, DIG+0.<3..0>.
	This icon denotes a tip, which alerts you to advisory information.
	This icon denotes a note, which alerts you to important information.
	This icon denotes a caution, which advises you of precautions to take to avoid injury, data loss, or a system crash.
bold	Bold text denotes items that you must select or click on in the software, such as menu items and dialog box options. Bold text also denotes parameter names.
italic	Italic text denotes variables, emphasis, a cross reference, or an introduction to a key concept. This font also denotes text that is a placeholder for a word or value that you must supply.
`monospace`	Text in this font denotes sections of code, programming examples, and syntax examples. This font is also used for the proper names of programs, subprograms, subroutines, device names, functions, operations, variables, filenames and extensions, and code excerpts.

1

Getting to Know Your 6527

Your 6527 device is a 48-bit, parallel, isolated digital I/O interface; PCI-6527 for PCI bus computers, and PXI-6527 for PXI or CompactPCI chassis. The 6527 devices offer 48 channels of isolated digital data acquisition. Twenty-four of the channels are optocoupler inputs and 24 are solid-state relay outputs. You can sense digital levels up to 28 VDC and switch currents of up to 120 mA. Your device can perform digital filtering, which eliminates glitches on input lines and change notification, which can generate interrupts on rising or falling edges on input lines to notify you of changing data.

For more information regarding the functions of your device, refer to the *6527 User Manual*, which also guides you through installing, making connections to, and using the 6527 safely.

Caution Using your 6527 device in a way inconsistent with the directions in the *6527 User Manual* can lead to equipment damage or injury. National Instruments is *not* liable for damages or injuries resulting from incorrect use.

Using Your 6527

The 6527 circuitry can be divided into the following functional groups:

- PCI interface
- General operation registers
- Digital filtering registers
- Change notification registers

PCI Interface

The 6527 uses the PCI MITE Application-Specific Integrated Circuit (ASIC) to communicate with the PCI or PXI bus. National Instruments designed this ASIC specifically for data acquisition. Before register-level programming the 6527 device, you must initialize the PCI interface as described in Chapter 3, *Programming*.

General Operation Registers

Initialize the PCI interface before using the general operation registers. Read the ID Register, one of the general operation registers, to verify the PCI interface is initialized properly. For more information on the ID Register, refer to the *ID Register* section in Chapter 2, *Register Map and Descriptions*.

The general operation registers also include the main input and output registers. Three eight-bit input registers report the values on the optocouplers that make up Ports 0, 1, and 2. Three eight-bit output registers open or close the solid-state relays (SSRs) that make up Ports 3, 4, and 5. You can read back from the output registers to determine the last value written.

Use Clear Register, another general operation register, to reset various functions such as digital filtering and change notification.

Using the Digital Filtering Option

Enable or disable filtering on any or all 24 input lines using digital filtering. Filtering can help eliminate glitches on input data and reduces the number of changes you need to process if you are using digital filtering with change notification.

To enable filtering, you must enable filtering on the chosen input lines and set a filter interval, which regulates all filters you enable. Refer to the *Digital Filtering* section in the *6527 User Manual*.

Using the Change Notification Option

Use change notification to generate an interrupt when any input line changes. You will need to know how to write, install, and uninstall interrupt service routines.

The change notification registers consist of two masks, one specifying the lines you want to monitor for rising edges, and another specifying the lines you want to monitor for falling edges. You can monitor any set of input lines for rising edges, falling edges, or both. The change notification registers also include an interrupt enable register.

You can use digital filtering concurrently with change notification, which enables you to limit the rate of interrupts. Refer to the *Change Notification* section in your *6527 User Manual* for more details about this option.

2

Register Map and Descriptions

Table 2-1 shows the register map for the 6527 devices. The table gives the register name, the register address offset from the device base address (Base Address Register 1), the size of the register in bits, and the type of register (read-only, write-only, or read-and-write).

Registers are grouped in the table by function. A bit-by-bit description of each register follows the table.

Table 2-1. 6527 Register Address Map

Register Name	Offset (Hex)	Type	Size
General Operation Registers			
Port 0 Register	00	Read-only	8-bit
Port 1 Register	01	Read-only	8-bit
Port 2 Register	02	Read-only	8-bit
Port 3 Register	03	Read-and-write	8-bit
Port 4 Register	04	Read-and-write	8-bit
Port 5 Register	05	Read-and-write	8-bit
ID Register	06	Read-only	8-bit
Clear Register	07	Write-only	8-bit
Digital Filtering Registers			
Filter Interval	08:0A	Read-and-write	Three 8-bit
Filter Enables, Port 0	0C	Read-and-write	8-bit
Filter Enables, Port 1	0D	Read-and-write	8-bit
Filter Enables, Port 2	0E	Read-and-write	8-bit

Table 2-1. 6527 Register Address Map (Continued)

Register Name	Offset (Hex)	Type	Size
Change Notification Registers			
Change Status	14	Read-only	8-bit
Master Interrupt Control	15	Read-and-write	8-bit
Port 0 Rising-Edge Detection Enable	18	Read-and-write	8-bit
Port 1 Rising-Edge Detection Enable	19	Read-and-write	8-bit
Port 2 Rising-Edge Detection Enable	1A	Read-and-write	8-bit
Port 0 Falling-Edge Detection Enable	20	Read-and-write	8-bit
Port 1 Falling-Edge Detection Enable	21	Read-and-write	8-bit
Port 2 Falling-Edge Detection Enable	22	Read-and-write	8-bit

The following pages provide a description of each register. The register bit map shows a diagram of the register with the most significant bit or MSB (bit 7) on the left and the least significant bit or LSB (bit 0) on the right. Each bit is represented by a rectangle with the bit name inside.

Input Registers (Ports 0–2)

The Input Registers are read-only. Reading an Input Register returns the logic state of the eight optically isolated digital input lines on the corresponding port.

Address Offsets: 00(hex) for Port 0
01(hex) for Port 1
02(hex) for Port 2

Type: Read-only

Size: 8-bit

Bit Map:

7	6	5	4	3	2	1	0
N.7	N.6	N.5	N.4	N.3	N.2	N.1	N.0

Bit	Name	Description
7–0	N.<7..0>	Data—These are the eight input data bits of input Port *N*, from bit seven down to bit zero. 1 = Logic high: voltage and/or current present 0 = Logic low: voltage and/or current absent

Output Registers (Ports 3–5)

The Output Registers are read-and-write. Write to an Output Register to control the switch states of the eight solid-state relays on the corresponding port. Read an Output Register to get the present contents of the register which represent the states of the eight relays associated with the port. The power-up state of each register is all high (hex FF), causing the relays to be open.

Address Offset: 03 (hex) for Port 3
04 (hex) for Port 4
05 (hex) for Port 5

Type: Read-and-write

Size: 8-bit

Bit Map:

7	6	5	4	3	2	1	0
N.7	N.6	N.5	N.4	N.3	N.2	N.1	N.0

Bit	Name	Description
7–0	N.<7..0>	Data—These are the eight output data bits of output Port *N*, from bit seven down to zero. 1 = Open relay (power-up state) 0 = Close relay

ID Register

The ID Register is read-only. Use this register to confirm that you are successfully reading from your device. Reading this register returns the hexadecimal value 27.

Address Offset: 06 (hex)

Type: Read-only

Size: 8-bit

Bit Map:

7	6	5	4	3	2	1	0
0	0	1	0	0	1	1	1

Bit	Name	Description
7–0	ID<7..0>	These bits return hex 27, indicating a 6527 device.

Clear Register

Write to the Clear Register to reset one or more functions of the 6527 device. The data you write to the clear register selects the function or functions you want to reset. Each bit set to 1 in the data resets one function

Address Offset: 07 (hex)

Type: Write-only

Size: 8-bit

Bit Map:

7	6	5	4	3	2	1	0
Reserved	Reserved	Reserved	Reserved	ClrEdge	Reserved	ClrFilter	ClrInterval

Bit	Name	Description
7–4	Reserved	Write only zeroes to these bits.
3	ClrEdge	Clear Edge Detectors—Set this bit to one to clear the EdgeStatus status bit in the Change Status Register and clears all edge detectors.
2	ClrOverflow	Clear Overflow—Set this bit to one, along with the Clear Edge Detectors bit, to clear the overflow status bit in the Change Status Register.
1	ClrFilter	Filter Clear—Resets the filter logic.
0	ClrInterval	Clear Filter Interval—Resets the filter interval clock to the current filter interval. Always write this bit after making any change to the Filter Interval Registers.

Filter Interval Registers

The Filter Interval Registers control the filter interval for distinguishing between valid input pulses and glitches. Refer to the *6527 User Manual* for information on selecting a filter interval. Set the Filter Interval Registers to the desired filter interval divided by 200 ns.

The filter interval is a 20-bit value. It occupies two eight-bit registers and four bits of a third register. After you set the filter interval, write to the ClrInterval bit of the Clear Register to ensure that the new filter interval takes effect immediately. The filter interval affects only those input lines for which you have set the Filter Enable Registers.

Address Offset: 08:0B (hex)

Type: Read-and-write

Word Size: Three 8-bit registers

Address Offset: 0A (hex)

Bit Map:

23	22	21	20	19	18	17	16
Reserved	Reserved	Reserved	Reserved	FI.19	FI.18	FI.17	FI.16

Address Offset: 09 (hex)

Bit Map:

15	14	13	12	11	10	9	8
FI.15	FI.14	FI.13	FI.12	FI.11	FI.10	FI.9	FI.8

Address Offset: 08 (hex)

Bit Map:

7	6	5	4	3	2	1	0
FI.7	FI.6	FI.5	FI.4	FI.3	FI.2	FI.1	FI.0

Bit	Name	Description
23–20	Reserved	Write only zeroes to these bits.
19–0	FI.<19..0>	Filter interval, bits 19 down to 0, in increments of 200 ns.

Filter Enables (Ports 0–2)

These registers enable filtering for the input ports. You can enable or disable filtering individually for each input line.

Address Offsets: 0C (hex) for Port 0
0D (hex) for Port 1
0E (hex) for Port 2

Type: Read-and-write

Size: 8-bit

Bit Map:

7	6	5	4	3	2	1	0
FE.7	FE.6	FE.5	FE.4	FE.3	FE.2	FE.1	FE.0

Bit	Name	Description
7–0	FE.<7..0>	Filter enable controls for bits seven down to zero. 1 = Filtering enabled 0 = Filtering disabled (power-up state)

Change Status Register

The Change Status Register gives the status of change detection.

Address Offsets: 14 (hex)

Type: Read only

Size: 8-bit

Bit Map:

7	6	5	4	3	2	1	0
Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	EdgeStatus

Bit	Name	Description
7–3	Reserved	These bits are undefined and should be ignored.
2	MasterInterruptStatus	Indicates that the device is asserting an interrupt.
1	Overflow	Indicates that at least one more edge has been detected since an interrupt is asserted.
0	EdgeStatus	Indicates an edge has been detected. If the EdgeInt bit is set in the Master Interrupt Control Register, EdgeStatus set indicates an interrupt is currently being asserted.

Master Interrupt Control Register

The Master Interrupt Control Register enables change detection interrupts. To select the line you want to for changes, use the Rising Edge Detection and Falling Edge Detection Registers.

Address Offsets: 15 (hex)

Type: Read-and-write

Size: 8-bit

Bit Map:

7	6	5	4	3	2	1	0
Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	EdgeInt

Bit	Name	Description
7–1	Reserved	Write only zeroes to these bits.
0	EdgeInt	Interrupt Enable control. 1 = Interrupt enabled 0 = Interrupt disabled

Rising-Edge Detection Registers (Ports 0–2)

These registers enable edge detection interrupts for rising edges on selected lines of input ports. To generate interrupts you must also set the EdgeInt bit in the Master Interrupt Control Register.

Address Offsets: 18 (hex) for Port 0
19 (hex) for Port 1
1A (hex) for Port 2

Type: Read-and-write

Size: 8-bit

Bit Map:

7	6	5	4	3	2	1	0
Rise.7	Rise.6	Rise.5	Rise.4	Rise.3	Rise.2	Rise.1	Rise.0

Bit	Name	Description
7–0	Rise.<7..0>	Rising-Edge Detection enables—Each bit enables interrupt generation on rising edges of the corresponding input line. 1 = Rising edge detection enabled 0 = Rising edge detection disabled

Falling-Edge Detection Registers (Ports 0–2)

These registers enable edge detection interrupts for falling edges on selected lines of input ports. To generate interrupts you must also set the EdgeInt bit in the Master Interrupt Control Register.

Address Offsets: 20 (hex) for Port 0
21 (hex) for Port 1
22 (hex) for Port 2

Type: Read-and-write

Size: 8-bit

Bit Map:

7	6	5	4	3	2	1	0
Fall.7	Fall.6	Fall.5	Fall.4	Fall.3	Fall.2	Fall.1	Fall.device0

Bit	Name	Description
7–1	Fall.<7..0>	Falling-Edge Detection enables—Each bit enables interrupt generation on falling edges of the corresponding input line. 1 = Falling-edge detection enabled 0 = Falling-edge detection disabled

3

Programming

This chapter contains programming instructions for operating the circuitry on your 6527 device. Most of the instructions given in this chapter are language independent. In other words, the steps tell you to read or write a given register or to detect if a given bit is set or cleared without presenting the actual code. You can modify the instructions toward a practical solution to fit your needs.

Programming your 6527 device involves writing to and reading from registers on the device. Registers are listed in Chapter 2, *Register Map and Descriptions*.

Note In this chapter, all numbers preceded by 0x are hexadecimal.

Common Terms

Common terms used in programming examples listed below:

Port 0	Address of Port 0 Register (Base Address + 0x00)
Port 1	Address of Port 1 Register (Base Address + 0x01)
Port 2	Address of Port 2 Register (Base Address + 0x02)
Port 3	Address of Port 3 Register (Base Address + 0x03)
Port 4	Address of Port 4 Register (Base Address + 0x04)
Port 5	Address of Port 5 Register (Base Address + 0x05)
ID	Address of ID Register (Base Address + 0x06)
Clear Register	Address of Clear Register (Base Address + 0x07)
Filter Interval	Address of Filter Interval Registers (Base Address + 0x08)

Port *N* Filter Enables	Address of Input Port *N* Filter Enables (Base Address + 0x0C + *N*, where *N* is the port number)
Change Status	Address of Change Status Register (Base Address + 0x14)
Master Interrupt Control	Address of Master Interrupt Control Register (Base Address + 0x15)
Port *N* Rising-Edge Detection	Address of Input Port *N* Rising Edge Detection Enables (Base Address + 0x18 + *N*, where *N* is the port number)
Port *N* Falling-Edge Detection	Address of Input Port *N* Falling Edge Detection Enables (Base Address + 0x20+ *N*, where *N* is the port number)
Write (*address, data*)	Generic function call for a memory space Write of *data* to *address*
Read (*address*)	Generic function call for a memory space Read from *address*
CWrite (*offset, data*)	PCI configuration space write of *data* to PCI configuration space *offset*

Initializing the PCI Local Bus

The PCI Local Bus is a high performance, 32-bit bus with multiplexed address and data lines. This system arbitrates and assigns resources through software, freeing you from manually setting switches and jumpers. The 6527 devices are fully compatible with the *PCI Local Bus Specification, Revision 2.0*, from the PCI Special Interest Group (SIG).

The PCI Local Bus moves data for both the PCI-6527 and PXI-6527 devices. Configure the bus-related resources before you execute a register-level program. To do this, you need to assign a base address and interrupt channel to your 6527 device shown in the following sections.

Tip Assign an interrupt even if you do not intend to use the change notification feature.

Initializing the PCI for the PC

For proper operation, configure the PCI MITE ASIC as described in this section. The references made to PCI BIOS[1] calls are left for you to implement.

First, write an algorithm that finds and stores configuration information about the device. You can do this by using PCI BIOS calls to search PCI configuration space for the National

[1] You can obtain more information on PCI BIOS calls from the PCI SIG on the World Wide Web (`http://www.pcisig.com`).

Instruments vendor ID (0x1093) and one of the following device IDs: PCI-6527 (0x2620) or PXI-6527 (0x2B10).

If a device is found, the algorithm can store the device's configuration information into a data structure. Base Address Register 0 (BAR0) points to the base address of the PCI MITE, while Base Address Register 1 (BAR1) points to the base address of the device registers. The size of each of these windows is 4 KB.

Both addresses will most likely be mapped above 1 MB in the memory map. This means that in order to communicate with the device you must know how to perform memory cycles to extended memory.

Tip To make communication with the device simpler, re-map the device below 1 MB in the memory map using PCI BIOS read and write calls.

Example

This pseudocode example re-maps the device below 1 MB. If you choose not to re-map the device, you still need to perform Steps 3 and 4 to initialize the device. All values in this example are 32 bits.

Use the following pseudocode to re-map the PCI MITE to memory address 0xD0000 and the device to memory address 0xD1000:

CWrite(0x10,0x000D0000)	//Write the address to which you want to re-map the PCI MITE to PCI configuration space offset 0x10 (BAR0)
Write(0xD0340,0x0000AEAE)	//Write the value 0x0000AEAE to offset 0x340 from the new PCI MITE address
CWrite(0x14,0x000D1000)	//Write the address to which you want to re-map the device (other than the PCI MITE) to PCI configuration space offset 0x14 (BAR1).
	//Create the window data value by masking the new device address:window data value = ((0xFFFFFF00 AND new device address) OR (0x00000080))
	//If you are not re-mapping the device, then the new device address is the value in BAR1
Write(0xD00C0,0x000D1080)	//Write the window data value to offset 0xC0 from the new PCI MITE address. If you are not re-mapping the device, then the new PCI MITE address is the value in BAR0.

The base address is now 0xD1000. Make sure the re-mapped PCI MITE and the 6527 memory ranges are not used by another device or system resource. You can exclude this memory from use with a memory manager.

Initializing the PCI for the Macintosh (PCI-6527 Only)

To program at the register level, you must know the base memory address and you must develop your own configuration program. To do this, consult the following documents:

- *Designing PCI Cards and Drivers for Power Macintosh Computers*
- *Inside Macintosh: Devices*
- *Inside Macintosh: Memory*
- *Inside Macintosh: Operating System Utilities*
- *Inside Macintosh: Processes*
- *Inside Macintosh: Power PC System Software*

Example

Use the following code sequence to activate the device. Using the documents listed above as a reference, write a program to retrieve the **deviceNode** parameter from the Name Registry.

```
#include <pci.h>
void*configureCard(RegEntryIDPtr deviceNode);
void*configureCard
(RegEntryIDPtrdeviceNode
)
{unsigned shortpciCommandRegister;
unsigned longcardBaseAddress,
miteBaseAddress;

//configure the i/o space of the device such
// that it is memory mapped.
ExpMgrConfigReadWord(deviceNode,
((LogicalAddress) 0x00000004L),
&pciCommandRegister);
ExpMgrConfigWriteWord(deviceNode,
((LogicalAddress) 0x00000004L),
(pciCommandRegister | 0x0002));

//get the base addresses for the device.
ExpMgrConfigReadLong(deviceNode,
((LogicalAddress) 0x00000010L),
&miteBaseAddress);
ExpMgrConfigReadLong(deviceNode,
```

```
((LogicalAddress) 0x00000014L),
&cardBaseAddress);

//activate the standard i/o window.
*((unsigned long *) (miteBaseAddress +
0x000000C0L)) =
EndianSwap32Bit(((cardBaseAddress &
0xFFFFFF00L) | 0x00000080L));

//return the base address of the device.
return ((void *) cardBaseAddress);
}
```

Programming the Digital I/O Circuitry

Programming examples are presented in this section for three basic 6527 functions: Input, Output, and Change Notification. Each example provides you with pseudocode.

All the ports on a 6527 device are preconfigured for input or output, so there is no need to configure individual ports prior to reading from or writing to them. For input and output specifications, refer to Appendix A, *Specifications,* of your *6527 User Manual*. Handshaking is also not required; simply write to or read from a specified port.

Input

You can read your 6527 device to get all data on a port. You can also set filter enable bits and a filter period before performing input. Filtering can help eliminate glitches on input data. At power-up, filtering is disabled.

To use an input port of a 6527 device, read the Port Register for the port to detect the logical states of the optically isolated digital lines associated with that port. Reading a digital 1 at the Port Register indicates a logic high on the input line (that is, greater than 2 V difference between DIG+ and DIG–).

Use the following pseudocode example:

```
Read (Port 0)                    //Read eight bits from Port 0
```

For example, if you read 0x01, or binary 00000001, this would indicate a logic high on Port 0, line 0, and a logic low on lines 1 through 7 of Port 0.

Input with Filtering

You have the option to set filter enable bits and a filter period before performing input. The following example enables filtering with an interval of 10 ms for Port 0 (all lines).

To use filtering, set the filter enable bits for the lines you want to filter. Then set a filter interval, which is common to all lines. The filter interval is in 200 ns increments. To set a filter interval of 10 ms, the value to write to the Filter Interval Register is 0x00C350 ((10 ms/200 ns) = 50,000 decimal = 0x00C350 hexadecimal).

Use the following pseudocode example:

```
Write (Filter Interval + 2, 0x00)          //Set most significant bits of filter interval

Write (Filter Interval + 1, 0xC3)

Write (Filter Interval + 0, 0x50)          //Set least significant bits of filter interval

Write (Clear Register, 0x03)               //Clear filters and filter interval clock

Write (Port 0 Filter Enables, 0xFF)        //Enable filtering for all eight bits of Port 0

Read (Port 0)                              //Begin reading filtered input
```

Output

To control an output port of a 6527 device, write a digital value to the appropriate Port Register to control the solid-state relays associated with that port. At power-up, all solid-state relays of all output ports are open.

Note Writing a digital 1 to a port line (which is also the initial power-up state) opens the relay. Writing a digital 0 to a port line closes the relay, allowing current to flow between the DIG+ and DIG– terminals of the switch.

The following example writes eight bits—11111110 binary, or 0xFE hexadecimal—to Port 0. This value closes the solid-state relay on Port 0, line 0, but leaves all other relays in the port open.

Use the following pseudocode example:

```
Write (Port 0, 0xFE)                       //Write 11111110 binary to Port 0
```

Change Notification

This section describes how to configure the change notification interrupt feature, handle interrupts using the 6527 device registers, and unconfigure the change notification interrupt feature.

As mentioned at the beginning of this manual, you need to know how to write, install, and uninstall interrupt service routines to use change notification with register-level programming. This manual does not cover interrupt service routines.

Configuring Interrupt Generation

This example generates an interrupt if certain lines of Input Port 0 change. As illustrated in Table 3-1, a fall on line 0, a rise on line 1, or any change on line 7, 6, 5, or 4 generates an interrupt.

Table 3-1. Change Notification Example

	Bit							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Changes to detect	↕	↕	↕	↕	—	—	↑	↓
Enable rising-edge detection	yes	yes	yes	yes	no	no	yes	no
Enable falling-edge detection	yes	yes	yes	yes	no	no	no	yes

Tip It is recommended that you enable input filtering to prevent a brief glitch or noise pulse from generating a false interrupt. Filtering also helps protect your computer from an excessive interrupt rate in case of rapidly changing inputs. This example includes input filtering.

Follow these steps to configure interrupt generation:

1. Enable input filtering (optional, but recommended)

   Set up filtering with interval of 10 ms

   Write (Filter Interval + 2,0x00) //Set most significant bits of filter interval

   Write (Filter Interval + 1,0xC3)

   Write (Filter Interval + 0,0x50) //Set least significant bits of filter interval

```
Write (Clear Register,0x03)          //Clear filters and filter interval clock
Write (Port 0 Filter Enables, 0xFF)  //Enable filter for eight bits of Port 0
```

2. Set the Edge Detection Enable bits for the lines and edges you want to detect

```
Write (Port 0 Rising Edge Detection, 0xF2)  //Rises to detect: lines 7 down to 4 and line 1
Write (Port 0 Falling Edge Detection, 0xF1) //Falls to detect: lines 7 down to 4 and line 0
```

3. Write to the Clear Register to clear any existing interrupts before you begin.

```
Write (Clear Register, 0x0F)         //Clear any previously set interrupt flags
```

4. After all other configuration, write to the Master Interrupt Control Register to enable interrupt generation

```
Write (Master Interrupt Control, 0x01)  //Enable Interrupts
```

Handling Interrupts

Inside your interrupt service routine (ISR), use the Change Status Register to verify that a change occurred using the following pseudocode example:

```
Read (Change Status)                 //Read status to verify change occurred
```

Then read the value of the input lines from the Port Register:

The value you read from the Port Register indicates the present state of the input lines at the time you execute the read. The state you read may not be the state that caused the interrupt because the inputs may have changed after the interrupt was generated. The Port Registers do not indicate which line changed, how many times a line changed, or whether a line rose or fell.

To clear a change notification interrupt, write to the Clear Register:

```
Write (Clear Register,0x08)          //Clear change detectors
```

Disabling Change Notification

To disable change notification, write to the Master Interrupt Control register:

```
Write(Master Interrupt Control,0x00)  //Disable interrupts
```

After disabling interrupts, you can write to any other configuration register that you want to reprogram.

A

Technical Support Resources

Web Support

National Instruments Web support is your first stop for help in solving installation, configuration, and application problems and questions. Online problem-solving and diagnostic resources include frequently asked questions, knowledge bases, product-specific troubleshooting wizards, manuals, drivers, software updates, and more. Web support is available through the Technical Support section of `ni.com`

NI Developer Zone

The NI Developer Zone at `ni.com/zone` is the essential resource for building measurement and automation systems. At the NI Developer Zone, you can easily access the latest example programs, system configurators, tutorials, technical news, as well as a community of developers ready to share their own techniques.

Customer Education

National Instruments provides a number of alternatives to satisfy your training needs, from self-paced tutorials, videos, and interactive CDs to instructor-led hands-on courses at locations around the world. Visit the Customer Education section of `ni.com` for online course schedules, syllabi, training centers, and class registration.

System Integration

If you have time constraints, limited in-house technical resources, or other dilemmas, you may prefer to employ consulting or system integration services. You can rely on the expertise available through our worldwide network of Alliance Program members. To find out more about our Alliance system integration solutions, visit the System Integration section of `ni.com`

Worldwide Support

National Instruments has offices located around the world to help address your support needs. You can access our branch office Web sites from the Worldwide Offices section of `ni.com`. Branch office Web sites provide up-to-date contact information, support phone numbers, e-mail addresses, and current events.

If you have searched the technical support resources on our Web site and still cannot find the answers you need, contact your local office or National Instruments corporate. Phone numbers for our worldwide offices are listed at the front of this manual.

Glossary

Prefix	Meaning	Value
n-	nano-	10^{-9}
m-	milli-	10^{-3}

Symbols

+	Positive of, or plus
–	Negative of, or minus

A

ASIC	Application-Specific Integrated Circuit—a proprietary semiconductor component designed and manuafactured to preform a set of specific functions for a specific application

C

CompactPCI	refers to the core specification defined by the PCI Industrial Computer Manufacturer's Group (PICMG)

D

DAQ	data acquisition—a system that uses the personal computer to collect, measure, and generate electrical signals
DIG+	positive data terminal
DIG–	negative data terminal
DIO	digital input/output

I

isolation	signal conditioning to break ground loops and reject high common-mode voltages to protect equipment and users and to ensure accurate measurements

L

LSB	least significant bit

M

MSB	most significant bit

O

optical isolation	the technique of using an optocoupler to transfer data without electrical continuity, to eliminate high-potential differences and transients
optocoupler	a device that transfers electrical signals by utilizing light waves to provide coupling with electrical isolation between input and output

P

PCI	Peripheral Component Interconnect—a high-performance expansion bus architecture originally developed by Intel to replace ISA and EISA.
port	a digital port, consisting of eight lines of digital input and/or output
PXI	PCI eXtensions for Instrumentation—an open specification that builds on the CompactPCI specification by adding instrumentation-specific features