

DAQ

DIO 6533 Register-Level Programmer Manual

December 1997 Edition
Part Number 341329A-01

Preliminary

The contents of this document are preliminary and are subject to change without notice. This document may contain errors, omissions, or information that no longer applies.

© Copyright 1997 National Instruments Corporation. All rights reserved.

PRELIMINARY



Internet Support

support@natinst.com

E-mail: info@natinst.com

FTP Site: ftp.natinst.com

Web Address: <http://www.natinst.com>



Bulletin Board Support

BBS United States: (512) 794-5422

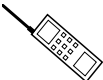
BBS United Kingdom: 01635 551422

BBS France: 01 48 65 15 59



Fax-on-Demand Support

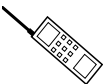
(512) 418-1111



Telephone Support (U.S.)

Tel: (512) 795-8248

Fax: (512) 794-5678



International Offices

Australia 03 9879 5166, Austria 0662 45 79 90 0, Belgium 02 757 00 20,
Canada (Ontario) 905 785 0085, Canada (Québec) 514 694 8521, Denmark 45 76 26 00,
Finland 09 725 725 11, France 01 48 14 24 24, Germany 089 741 31 30,
Hong Kong 2645 3186, Israel 03 5734815, Italy 02 413091, Japan 03 5472 2970,
Korea 02 596 7456, Mexico 5 520 2635, Netherlands 0348 433466, Norway 32 84 84 00,
Singapore 2265886, Spain 91 640 0085, Sweden 08 730 49 70, Switzerland 056 200 51 51,
Taiwan 02 377 1200, United Kingdom 01635 523545

National Instruments Corporate Headquarters

6504 Bridge Point Parkway Austin, TX 78730-5039 Tel: (512) 794-0100

PRELIMINARY

Important Information

Warranty

The DIO 6533 devices are 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 manual 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

Under the copyright laws, this publication may not be reproduced or transmitted in any form, electronic or mechanical, including photocopying, recording, storing in an information retrieval system, or translating, in whole or in part, without the prior written consent of National Instruments Corporation.

Trademarks

CVI™, Component Works™, DAQCard™, LabVIEW™, Mite™, NI-DAQ™, RTSI™, and VirtualBench™ are trademarks of National Instruments Corporation.

Product and company names listed are trademarks or trade names of their respective companies.

WARNING REGARDING MEDICAL AND CLINICAL USE OF NATIONAL INSTRUMENTS PRODUCTS

National Instruments products are not designed with components and testing intended to ensure a level of reliability suitable for use in treatment and diagnosis of humans. Applications of National Instruments products involving medical or clinical treatment can create a potential for accidental injury caused by product failure, or by errors on the part of the user or application designer. Any use or application of National Instruments products for or involving medical or clinical treatment must be performed by properly trained and qualified medical personnel, and all traditional medical safeguards, equipment, and procedures that are appropriate in the particular situation to prevent serious injury or death should always continue to be used when National Instruments products are being used. National Instruments products are NOT intended to be a substitute for any form of established process, procedure, or equipment used to monitor or safeguard human health and safety in medical or clinical treatment.

P R E L I M I N A R Y

PRELIMINARY



*Table
of
Contents*

About This Manual

Organization of This Manual	vii
Conventions Used in This Manual	vii
National Instruments Documentation	viii
Related Documentation	ix
Customer Communication	ix

Chapter 1

General Description

General Characteristics	1-1
-------------------------------	-----

Chapter 2

Programming

Bus Initialization	2-2
PCI-DIO-32HS and PXI-6533 Local Bus	2-2
PCI-DIO-32HS and PXI-6533 Initialization for the PC	2-2
AT-DIO-32HS Plug and Play	2-4
DAQCard-6533 Initialization	2-6
Windowing Registers	2-7
AT-DIO-32HS and DAQCard-6533	2-7
Programming Examples	2-7
PCI-DIO-32HS and PXI-6533	2-8
PCI-DIO-32HS Example 1	2-8
AT-DIO-32HS	2-9
AT-DIO-32HS Example 1	2-9
DAQCard-6533	2-10
DAQCard-6533 Example 1	2-10
Interrupt Programming	2-11
DMA Programming	2-11
PCI-DIO-32HS and PXI-6533	2-11
PCI-DIO-32HS and PXI-6533 Example 5	2-12
AT-DIO-32HS	2-17
AT-DIO-32HS Example 5	2-18
RTSI	2-18

PRELIMINARY

Table of Contents

PCI-DIO-32HS, PXI-6533, and AT-DIO-32HS	2-18
Programming the RTSI Bus Interface	2-18
Programming the RTSI Bus Switch	2-20
Initializing the RTSI Bus Switch	2-22
RTSI Bus Register Group	2-22
RTSI_SERIAL_REG	2-23
RTSI_PARALLEL_REG	2-24

Appendix A Customer Communication

Glossary

Index

Figures

Figure 2-1. DMA Link Chaining Mode Structure	2-12
Figure 2-2. RTSI Switch Control Pattern.....	2-20

Tables

Table 2-1. RTSI Switch Signal Connections—Side A	2-18
Table 2-2. RTSI Switch Signal Connections—Side B	2-19



*About
This
Manual*

This manual contains information concerning the register-level programming of DIO 6533 (DIO-32HS) devices.

The DAQ-DIO is an application specific integrated circuit (ASIC) designed by National Instruments. Consequently, this manual repeats certain information from, or draws your attention to, sections in the *DAQ-DIO Technical Reference Manual*. You must use your register-level programmer manual along with the *DAQ-DIO Technical Reference Manual* for a complete understanding of DIO 6533 device programming.

Organization of This Manual




The *DIO 6533 Register-Level Programmer Manual* is organized as follows:

- Chapter 1, *General Description*, describes the general characteristics of the DIO 6533 devices.
- Chapter 2, *Programming*, contains register-level programming instructions for operating the circuitry on the DIO 6533 devices.
- Appendix A, *Customer Communication*, contains forms you can use to request help from National Instruments or to comment on our products.
- The *Glossary* contains an alphabetical list and description of terms used in this manual, including acronyms, abbreviations, metric prefixes, mnemonics, and symbols.
- The *Index* contains an alphabetical list of key terms and topics in this manual, including the page where you can find each one.

Conventions Used in This Manual

The following conventions are used in this manual:

About This Manual

<>	Angle brackets containing numbers separated by an ellipsis represent a range of values associated with a bit or signal name (for example, DBIO<3..0>).
	This icon to the left of bold italicized text denotes a note, which alerts you to important information.
	This icon to the left of bold italicized text denotes a caution, which advises you of precautions to take to avoid injury, data loss, or a system crash.
	This icon to the left of bold italicized text denotes a warning, which advises you of precautions to take to avoid being electrically shocked.
<i>bold italic</i>	Bold italic text denotes a note, caution, or warning.
<i>italic</i>	Italic text denotes emphasis, a cross reference, or an introduction to a key concept. This font also denotes text from which you supply the appropriate word or value, as in Windows 3.x.
monospace	Text in this font denotes text or characters that should literally enter from the keyboard, sections of code, programming examples, and syntax examples. This font is also used for the proper names of disk drives, paths, directories, programs, subprograms, subroutines, device names, functions, operations, variables, filenames and extensions, and for statements and comments taken from programs.
DIO 6533 device	DIO 6533 device refers to the PCI-DIO-32HS, PXI-6533, AT-DIO-32HS, and the DAQCard-6533, unless otherwise noted. Abbreviations, acronyms, metric prefixes, mnemonics, symbols, and terms are listed in the <i>Glossary</i> .

National Instruments Documentation

DIO 6533 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 different types of manuals you have as follows:

- Your DAQ hardware user manuals—These manuals have detailed information about the DAQ hardware that plugs into or is connected to your computer. Use these manuals for hardware

PRELIMINARY

installation and configuration instructions, specification information about your DAQ hardware, and application hints.

- Software documentation—You might have several sets of software documentation, including LabVIEW, LabWindows[®]/CVI, ComponentWorks, VirtualBench, and NI-DAQ. After you have set up your hardware system, use either the application software (LabVIEW or LabWindows/CVI) or the NI-DAQ documentation to help you write your application. If you have a large and complicated system, it is worthwhile to look through the software documentation before you configure your hardware.
- Accessory installation guides or manuals—If you are using accessory products, read the terminal block and cable assembly installation guides. They explain how to physically connect the relevant pieces of the system. Consult these guides when you are making your connections.

Related Documentation

The following National Instruments documents contain additional information required for operating and programming your DIO 6533 device:

- *DIO 6533 User Manual*—This manual describes the features, functions, and modes of the DIO 6533 devices and explains how to connect to and operate these devices
- *DAQ-DIO Technical Reference Manual*—This manual describes the registers of the DAQ-DIO and how to program them. The DAQ-DIO contains the digital I/O logic for the DIO 6533 devices

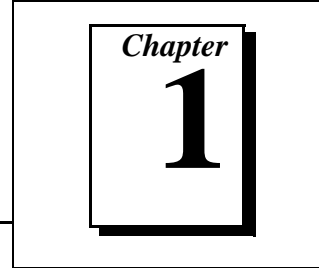
The following documents also contains information you will need:

- *ISA Plug and Play Specification*, version 1.0. (available from Intel)
- *PC Card Standard; Card Services Specification, Socket Services Specification*, and other volumes. Personal Computer Memory Card International Association (PCMCIA)

Customer Communication

National Instruments wants to receive your comments on our products and manuals. We are interested in the applications you develop with our products, and we want to help if you have problems with them. To make it easy for you to contact us, this manual contains comment and configuration forms for you to complete. These forms are in Appendix A, *Customer Communication*, at the end of this manual.

General Description



This chapter describes the general characteristics of the DIO 6533 devices.

General Characteristics

The DIO 6533 devices are 32-bit parallel digital I/O interfaces for several different types of computers. The DIO 6533 devices include the following cards:

- PCI-DIO-32HS, for PCI buses
- PXI-6533, for PXI and CompactPCI chassis
- AT-DIO-32HS, for AT (16-bit ISA) buses
- DAQCard-6533, for computers equipped with Type II PCMCIA slots

Each DIO 6533 device contains three types of logic that you can program:

- System bus interface logic, which allows you to assign system resources, such as a base address, to the device.
- Digital I/O logic, implemented by the DAQ-DIO. See the *DAQ-DIO Technical Reference Manual* for detailed programming information.
- A RTSI crossbar switch, for connection to the RTSI or PXI trigger bus (PCI-DIO-32HS, PXI-6533, and AT-DIO-32HS only).

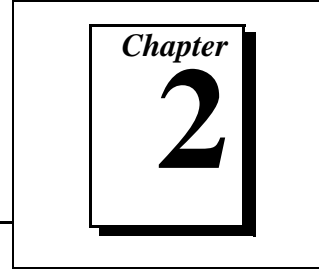
The system bus interface logic for the PCI-DIO-32HS and the PXI-6533 is implemented by the National Instruments PCI MITE ASIC and includes onboard direct memory access (DMA) controllers. You can also configure DMA for the AT-DIO-32HS, but you must use your computer system's DMA controllers. The DAQCard-6533 does not support DMA. Instead, you should program the DAQCard-6533 using polled or interrupt-driven I/O.

For descriptions of DIO 6533 features, modes, signal connections, and timing, refer to the *DIO 6533 User Manual*.

For register information and bitfield locations, refer to *DAQ-DIO Technical Reference Manual*.

PRELIMINARY

Programming



This chapter contains register-level programming instructions for operating the circuitry on the DIO 6533 device. Except where noted, information applies to all DIO 6533 devices: the PCI-DIO-32HS, PXI-6533, AT-DIO-32HS, and DAQCard-6533.

All software functions referred to in this chapter are provided on the *DIO 6533 Register-Level Programmer Manual* Companion Disk. See the *Programming Examples* section later in this chapter.

To program a DIO 6533 device, you must write to and read from a set of registers implemented by the device. The registers fall into three categories:

- Bus-interface registers, which are discussed in the *Bus Initialization* section in this chapter
- Digital I/O registers, which are implemented by the DAQ-DIO and detailed in the *DAQ-DIO Technical Reference Manual*
- Two RTSI bus registers (PCI-DIO-32HS, PXI-6533, and AT-DIO-32HS only), which are described in the *Programming the RTSI Bus Interface* section in this chapter

The programming steps for unstrobed I/O, pattern generation, and handshaking are explained in the *DAQ-DIO Technical Reference Manual*. These steps describe how to perform digital I/O operations by reading and writing to bitfields. A *bitfield* is defined as a group of contiguous bits that jointly perform a function.

Because the *DAQ-DIO Technical Reference Manual* has detailed, step-by-step programming instructions, this register-level programmer manual gives a series of common programming examples with references to the *DAQ-DIO Technical Reference Manual*. The procedure for each example is presented as it appears in the *DAQ-DIO Technical Reference Manual*, with bitfields to be read from or written to for each function. To implement the function, you will need to read from or write to the specified registers. The complete examples are provided on the *DIO 6533 Register-Level Programmer Manual* Companion Disk.

Bus Initialization

This section describes how to configure bus-related resources for the PCI-DIO-32HS, PXI-6533, and the AT-DIO-32HS, and how to activate the DAQCard-6533. You need to complete the following steps for the appropriate DIO 6533 device before you can execute a register-level program or access DAQ circuitry.

PCI-DIO-32HS and PXI-6533 Local Bus

The PCI-DIO-32HS is fully compatible with the PCI Local Bus Specification, version 2.1, from the PCI Special Interest Group (SIG). The PXI-6533 is fully compatible with the PXI Specification, revision 1.0. The PCI local bus is a high performance, 32-bit bus with multiplexed address and data lines. The PCI and PXI systems arbitrate and assign resources through software, freeing you from manually setting switches and jumpers. You must configure the bus-related resources before attempting to execute a register-level program.

You can use the PCI-DIO-32HS and PXI-6533 with both IBM-compatible and Mac OS computers. However, due to the difference in those systems, configuration will be different. PCI and PXI initialization on only the PC is given below. Detailed code and comments can be found on the companion disk.

PCI-DIO-32HS and PXI-6533 Initialization for the PC

The PCI and PXI devices use the MITE ASIC chip as the PCI or PXI bus interface. National Instruments designed this ASIC specifically for data acquisition. In order for the board to operate properly, this chip must be configured. Ordinarily, NI-DAQ performs this function, but if you are not using NI-DAQ, you must configure the MITE chip. The following sections explain how to accomplish this.

The initialization is done by the `Setup_Mite()` function. `Setup_Mite()` consists of three parts. First, the function detects if the PCI or PXI local bus is present. The `Is_PCI()` function uses PCI BIOS¹ calls to verify that the PCI or PXI bus is present.

Second, the PCI or PXI bus is scanned for all National Instruments PCI or PXI devices. The `find_NI_Devices()` function uses PCI BIOS

1. The PCI SIG provides further information on PCI BIOS calls.

calls to find all PCI or PXI devices that contain the National Instruments vendor ID (0x1093) and a PCI or PXI device ID (the PCI-DIO-32HS device ID is 0x1150 and the PXI-6533 device ID is 0x1320). If a card is found, the program stores the card's pertinent information into a data structure.

Third, the MITE device windows must be configured. Base Address Register 0 (BAR0) corresponds to the base address of the MITE, while Base Address Register 1 (BAR1) is the base address of the DAQ-DIO and RTSI 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 to communicate with the board you must know how to perform memory cycles to extended memory. The `Setup_Mite()` function will re-map the board under 1 MB in the memory map, which makes communicating with the board simpler. PCI BIOS read and write calls accomplish this. Use the pseudocode in this section to re-map the board below 1 MB. If you choose not to re-map the board, you must still perform Steps 4 and 5. All values in this example are 32 bits.

1. Write the address to which you want to re-map the MITE to PCI or PXI configuration space offset 0x10 (BAR0).
2. Write the value 0x0000aeae to offset 0x340 from the new MITE address. (This should be the power-up value. If so, this step is optional.)
3. Write the address to which you want to re-map the card register to PCI or PXI configuration space offset 0x14 (BAR1).
4. Create the device window data value by masking the new card address:
 - device window data value = ((0xfffff00 and new card address) or (0x00000080))
 - If you are not re-mapping the card, then the new card address is the original value in BAR1.
5. Write the device window data value to offset 0xc0 from the new MITE address.
 - If you are not re-mapping the card, then the new MITE address is the original value in BAR0.

P R E L I M I N A R Y

The sequence to re-map the board under 1 MB cannot operate successfully on PCs with the PCI-PCI bridge chip set. Those computers include the Dell Optiplex Gxpro series. In this case, the PCI or PXI initialization should read back only the card base address (BAR1) and MITE base address (BAR0), which are assigned when the PC boots. You must write your own memory read and write routines to make the examples work properly.

AT-DIO-32HS Plug and Play

The AT-DIO-32HS is fully compatible with the Intel Plug and Play Specification, version 1.0. The Plug and Play system arbitrates and assigns resources through software, freeing you from manually setting switches and jumpers. You must configure the bus-related resources before attempting to execute a register-level program.

If your system has a Plug and Play configuration manager that assigns available resources at startup, you must query the configuration manager to determine the resources assigned to each device. If a configuration manager is not installed, you will need a configuration program to assign resources to the device. The *DIO 6533 Register-Level Programmer Manual* Companion Disk contains a function that you can execute to configure the base address of the AT-DIO-32HS.



Note: *The given Plug and Play initialization applies only for one device. If you need to configure more than one device, the Plug and Play specification explains the procedure.*

This manual does not cover the software implementation of the full Plug and Play protocol. Refer to the *ISA Plug and Play Specification, version 1.0*, available from Intel, for more information. This section discusses how to assign a base I/O address to the AT-DIO-32HS. All access to the Plug and Play registers is performed through three I/O address locations. The address port is at location 0x279, and indicates which configuration register is the target of the next data transfer. The write data port is at location 0xA79, and writes to the Plug and Play configuration register indicated by the address port contents. The read data port is moveable in the range of 0x203 to 0x3FF, and reads the Plug and Play configuration register indicated by the address port contents. Writing to certain configuration registers will set the location of the read data port.

An internal state machine controls access to the Plug and Play configuration registers. The state machine consists of four states: Wait4Key, Sleep, Isolation, and Configuration. The device powers up in the Wait4Key state, where the device is inactive and unconfigured. A sequence of specific accesses to the address port will move the device into the Sleep state. This long sequence of 34 writes decreases the chance that a poorly written program that writes randomly into I/O space does not accidentally access the Plug and Play configuration registers. The `Write_PNP_Initiation_Key` function performs this series of writes.

The device now waits in the Sleep state. If you previously configured the device, you may wish to reset all of the Plug and Play configuration registers with the `Reset_PNP_Registers` function.

**Note:**

If you reset the Plug and Play configuration registers, the registers will return the state machine to the Wait4Key state. You will need to write the Initiation Key to the device once more to return to the Sleep state.

You can now move the device into the Isolation state. The Isolation state identifies a single Plug and Play device from all of the devices in the system. Since the example code supports only a single Plug and Play device in the system, no action is actually performed during the Isolation state. The `Set_PNP_Isolation_State` function moves the device from the Sleep to the Isolation state.

Since no actions need to be performed in the Isolation state, you can now move the device into the Configuration state. Only the Configuration state can specify the device's resource assignment, where you set the base I/O address. The `Set_PNP_Configuration_State` function moves the device from the Isolation to the Configuration state.

You can now assign the base I/O address for the device. The AT-DIO-32HS uses full 16-bit decoding, so you must assign both the high and low bytes of the base I/O address. Run the `Assign_PNP_Base_Address` function.

Up to this point there has been no feedback to the program to check if everything is okay. Read the base address registers to verify that they were written correctly. First, you must assign the read data port to a free location. The read data port can be located at any location between 0x203 and 0x3FF where the lowest two bits of the address are set; for example, 0x203, 0x207, 0x20B, ..., 0x3FB, 0x3FF. You must write the

value of the address right-shifted by two to the configuration register, as shown in the `Assign_PNP_Data_Port_Address` function.

You can now read the contents of the base I/O address registers with the `Verify_PNP_Base_Address` function. These values should match those written earlier in the program.

You now need to activate the device for the assigned base I/O address to take effect. Once activated, the device will return to the `Wait4Key` state. The `Activate_PNP_Board` function activates the device.

Now you can access the registers for the AT-DIO-32HS by their offset from the base I/O address assigned above.



Note: *You must repeat the code given above every time the computer is rebooted or powered off. The hardware stores the assigned base I/O address in an onboard register, not in any form of nonvolatile memory. The device will lose its base I/O address after a reboot.*

DAQCard-6533 Initialization

Before you can access the DAQ circuitry on the DAQCard-6533, the card must be activated. PC cards are inactive until Card Services activates the card by assigning an interrupt level and an address space for the I/O registers. The DAQCard-6533 requires a 16-byte I/O address window and one interrupt level.

There are at least two different ways to activate the card:

- If you are using the DAQCard-6533 with National Instruments software such as NI-DAQ, LabVIEW, or LabWindows/CVI, the NI-DAQ configuration utility requests activation. For more information about this procedure, see the section on installation and configuration in your NI-DAQ manual.
- If the option above is not feasible for your application, you can develop your own program to activate the card. However, this is fairly complicated, and it requires significantly more programming. If you develop your own program, you should consult the PCMCIA documents, *Card Specifications* and *Socket Services Specifications*, which explain how to activate a card using system-level calls. You will need to request an I/O window, an interrupt level, and a configuration. In the configuration, the configuration index should be set to 01 hex for normal operation. For more information about these operations, refer to *Card Specifications* and *Socket Services Specifications*.

After you activate the card, you are ready to program the DAQCard-6533.

Windowing Registers

This section describes using the Windowing Registers with the AT-DIO-32HS and the DAQCard-6533.

AT-DIO-32HS and DAQCard-6533

On the AT-DIO-32HS and DAQCard-6533, the DAQ-DIO registers are read from and written to in I/O space. To save I/O space, these cards allow direct reads and writes to register locations 0 through 15 only. To access DAQ-DIO locations above 15 (including, on the AT-DIO-32HS, all RTSI register locations), you must use windowing. See the *DAQ-DIO Technical Reference Manual* for information on DAQ-DIO windowing.

Programming Examples

Each example in this chapter provides the pseudocode for the main program. The examples follow the procedure presented in detail in the *DAQ-DIO Technical Reference Manual*. Further explanations on the functions are in this manual. The *DIO 6533 Register Level Programmer Manual* Companion Disk provides each program in its entirety.

As mentioned in Chapter 1, *Introduction and General Information*, in the *DAQ-DIO Technical Reference Manual*, several write-only registers on the DAQ-DIO contain bitfields that control a number of functionally independent parts of the chip. To update bitfields, you must set or clear specific bits without changing the status of the remaining bits in the register. Since you cannot read these registers to determine which bits are set, you should maintain a soft copy of the write-only registers.

**Note:**

For simplicity, these examples do not include soft copies of the registers. If you wish to write your own examples, or modify these examples, we strongly recommend adding soft copies of the write-only registers. Please refer to the Register and Bitfield Programming Considerations section in Chapter 1 in the DAQ-DIO Technical Reference Manual for further information.

PCI-DIO-32HS and PXI-6533

The *DIO 6533 Register-Level Programmer Manual Companion Disk* contains the support files necessary to run the examples. The support files include `dma.h`, `dma.c`, `miteregb.h`, `lowio.h`, `lowio.c`, `pc_pci.c`, `pc_rwd.c`, `pci.c`, `pci.h`, `pcidio.h`.

The `dma.c` file contains the functions `MINIMITE_DMAProgram`, `MINIMITE_DMAarm`, and `MINIMITE_DMAdisarm`.

The `miteregb.h` file contains the bitfield assignment for the MITE.

The `lowio.c` file has the low level routines for memory read and write.

The `pcidio.h` file declares the external function prototypes and the register addresses.

The `pc_pci.c` file includes the `Setup_Mite()` function, which searches for the PCI device and reassigns the device address.

The `oc_rwd.c` file contains the functions which read and write from Windows and device discrete registers.

PCI-DIO-32HS Example 1

Configure pins 0, 2, 4, and 6 of port B for output and the remaining pins for input. Connect wires to wrap back the output pins to the input pins. Write out patterns 0b1010 and 0b1111 to the output pins. Read back the input pins and check to verify the same pattern.

1. Set up the PCI bus interface. Use the `Setup_Mite` function provided on the companion disk.
2. `Port_B_Mask = 0x00;`
3. Configure lines 0, 2, 4, and 6 as outputs and 1, 3, 5, and 7 as inputs.
`Port_B_Directions = 0x55;`
4. Write the digital pattern
`PORTB = 0x44;`
5. Read the digital pattern
Declare variable `DATA`;
`DATA (8-bit) = PORTB;`
6. Write the digital pattern
`PORTB = 0x55;`

7. Read the digital pattern
DATA (8-bit) = PORTB;

AT-DIO-32HS

The *DIO 6533 Register-Level Programmer Manual Companion Disk* contains the support files necessary to run the examples. The support files include `ATPNP.c`, `DIO32RLP.h`, and `ATFNCT.c`, which should all be included in the project for each example.

The `ATPNP.c` file includes the `Setup_PNP_Board` function, which assigns the base I/O address, as discussed in the *AT-DIO-32HS Plug and Play* section earlier in this chapter.

The `DIO32RLP.h` file declares the external function prototypes and the register addresses.

The `ATFNCT.c` file contains functions that read from and write to windowed and direct-access registers.

AT-DIO-32HS Example 1

Configure pins 0, 2, 4, and 6 of port B for output and the remaining pins for input. Wrap back the output to the input pins. Write out patterns 0b1010 and 0b1111 to the output pins. Read back the input pins and check to verify the same pattern.

1. Set up the Plug and Play interface. Use the `Setup_PNP_board` function provided on the companion disk
2. `Port_B_Mask = 0x00;`
3. Configure lines 0, 2, 4, and 6 as outputs and 1, 3, 5, and 7 as inputs
`Port_B_Directions = 0x55;`
4. Write the digital pattern
`PORTB = 0x44;`
5. Read the digital pattern
Declare variable `DATA`;
`DATA(8-bit) = PORTB;`
6. Write the digital pattern
`PORTB = 0x55;`
7. Read the digital pattern
`DATA(8-bit) = PORTB;`

DAQCard-6533

The *DIO 6533 Register-Level Programmer Manual Companion Disk* contains the support files necessary to run the examples. The support files include `DIO32RLP.h`, and `DCFNCT.c`, which should all be included with the project for each example.

The `DIO32RLP.h` file declares the external function prototypes and the register addresses.

The `DCFNCT.c` file contains functions that read from and write to windowed and direct-access registers.

DAQCard-6533 Example 1

Configure pins 0, 2, 4, and 6 of port B for output and the remaining pins for input. Wrap back the output pins to the input pins. Write out patterns 0b1010 and 0b1111 to the output pins. Read back the input pins and check to verify the same pattern.

1. Set a value to `Board_Base_Address` variable. This value should be the same as the I/O address assigned to DAQCard-6533 during configuration
2. `Port_B_Mask = 0x00;`
3. Configure lines 0, 2, 4, and 6 as outputs and 1, 3, 5, and 7 as inputs
`Port_B_Directions = 0x55;`
4. Write the digital pattern
`PORTB = 0x44;`
5. Read the digital pattern
Declare variable `DATA`;
`DATA(8-bit) = PORTB;`
6. Write the digital pattern
`PORTB = 0x55;`
7. Read the digital pattern
`DATA(8-bit) = PORTB;`

Interrupt Programming

The DAQ-DIO has one interrupt output line. You can program both handshaking groups to generate interrupt signals to the interrupt output line when certain conditions are satisfied.

The DAQ-DIO has the following features for the interrupts:

- One software controllable interrupt line for direct interface to the bus
- Up to four sets of I/O lines (one for each I/O port) can be used to program interrupts

Chapter 7, *Interrupt Controls*, in the *DAQ-DIO Technical Reference Manual*, discusses DIO 6533 interrupt programming.

DMA Programming

This section describes how to use the PCI-DIO-32HS, PXI-6533, and the AT-DIO-32HS with Direct Memory Access (DMA).

PCI-DIO-32HS and PXI-6533

The MITE contains DMA channels that you can use to transfer data between its I/O port (the DAQ-DIO) and its CPU port (system memory). Each channel supports several modes of operation. One of the modes of operation is the link chaining mode, which enables you to do DMA transfers limited only by the size of the memory space. This mode of operation is used in example 5. The link chaining mode requires a linked list structure to store the information for each buffer. The linked list structure enables DMA transfers on different memory segments and makes seamless data transfer possible.

Inside the linked list structure, each node contains values for TCR, MAR, DAR, and LKAR. MAR (Memory Address Register) stores the buffer's physical address; TCR (Transfer Count Register) stores the number of points to transfer; DAR (Device Address Register) is unused by the DAQ-DIO; LKAR (Link Address Register) stores the physical address of the next node.

Before beginning the DMA transfers, you must load LKAR with the physical address of the first node, because the MITE needs to have an entry point to access the link chain. After you arm the DMA transfer,

the MITE goes through the link chain and loads the buffer's physical address into MAR. Then the MITE transfers data from the card to the buffer (input) or from the buffer to the card (output). This process continues until the MITE reaches an empty node, a node which contains all zeros.

Figure 2-1 illustrates the basic operation of the link chain mode.

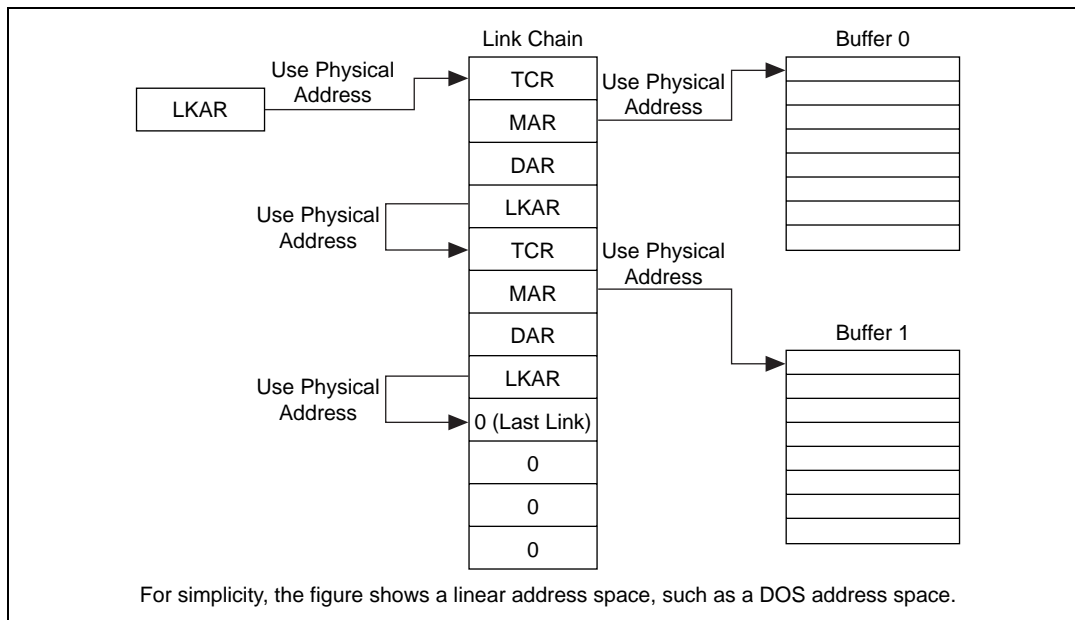


Figure 2-1. DMA Link Chaining Mode Structure

PCI-DIO-32HS and PXI-6533 Example 5

Enable FIFO A and FIFO B in group 1 and FIFO C and FIFO D in group 2. Configure group 1 as input and group 2 as output. Perform 16-bit programmed I/O output and 16-bit DMA input without funneling. Use burst mode handshaking.

Use wires to wrap back the output pins to the input pins. Use the 16-bit group-FIFO transfers to write hex 8990,...,89b7 to the output FIFOs and use 16-bit DMA to read from the input FIFOs. The DMA controller puts the data into a buffer of size 80 bytes. This example only uses one buffer, however, you can use more than one buffer if necessary. This example constructs a linked list to contain the physical address, logical address, and total transfer bytes of each buffer. You need to pass the

linked list head node to the `MiniMITE_DMAProgram` function to program the MITE.

In this example, you should wire the two groups' burst handshaking signals together: wire REQ1 to ACK2, REQ2 to ACK1, and PCLK1 to PCLK2.

Since this example performs no virtual-to-physical address translation, it will run only under DOS. If you want to program DMA under other operating systems, you must learn about physical memory address calculation and the use of virtual memory. For example, if you want to program DMA under Windows or Mac OS, make sure that you lock the memory locations for DMA, so the virtual memory manager cannot move the data from its current physical memory location. This example provides a basic outline for you to program the MITE for DMA transfers.

The `dma.c` file contains the `MINIMITE_DMAProgram`, `MINIMITE_DMAarm`, and `MINIMITE_DMAdisarm` functions. The `miteregb.h` file contains the bitfield assignment for the MITE.

1. Set up the PCI bus interface. Use the `Setup_Mite` function provided on the companion disk.
2. Configure handshaking
 - a. `Protocol_Register_1(group 1)=0x00;`
 - b. `Protocol_Register_1(group 2)=0x00;`
`RunMode = 0`
`Configuration = 0`
 - c. Configure data paths and ports
 - `Data_Path (group 1) = 0x03;`
`FIFOEnables (A) = 1;`
`FIFOEnables (B) = 1;`
`Funneling = 0;`
`GroupDirection = 0; /*input*/`
 - `Data_Path (group 2) = 0x8C;`
`FIFOEnables (C) = 1;`
`FIFOEnables (D) = 1;`
`Funneling = 0;`
`GroupDirection = 1; /*output*/`

- Configure ports

```
Port_C_Mask = 0x00;      /*standard drive type*/
Port_D_Mask = 0x00;
Port_A_Directions = 0x00; /*input*/
Port_B_Directions = 0x00;
Port_C_Directions = 0xFF; /*output*/
Port_D_Directions = 0xFF;
```
- d. Set protocol to burst mode

```
Protocol_Register_1 (group 1) = 0x00;
Protocol_Register_2 (group 1) = 0x60;
Protocol_Register_3 (group 1) = 0x00;
Protocol_Register_4 (group 1) = 0x08;
Protocol_Register_5 (group 1) = 0x04;
Protocol_Register_6 (group 1) = 0x00;
Protocol_Register_7 (group 1) = 0x60;
Protocol_Register_8 (group 1) = 0x01;

Protocol_Register_1 (group 2) = 0x00;
Protocol_Register_2 (group 2) = 0x10;
Protocol_Register_3 (group 2) = 0x00;
Protocol_Register_4 (group 2) = 0x20;
Protocol_Register_5 (group 2) = 0x00;
Protocol_Register_6 (group 2) = 0x00;
Protocol_Register_7 (group 2) = 0x60;
Protocol_Register_8 (group 2) = 0x01;

Clock_Speed (group 1) = 0x00;
Clock_Speed (group 2) = 0x00;
ClockSpeed = 0;
FIFO_Control (group 1) = 0x00;
FIFO_Control (group 2) = 0x01;
ReadyLevel = 0;
```
- e. Construct the buffer information linked list. Each node contains information about one buffer. Every node stores a buffer's physical address and logical address, and the total bytes to transfer for that buffer.

- f. Load output FIFOs with initial data
 Declare variable OUTPUT COUNT;
 OUTPUT COUNT = 0;
 FLAG = TransferReady (group 2);
 While (FLAG = 1) and (OUTPUT COUNT < number of
 points to transfer)
 {
 GroupFIFO (group 2) = data (16 bits);
 FLAG = TransferReady (group 2);
 }
3. Configure DMA
- DMA_Line_Control (group 1) = 0x05;
 - DMA_Line_Control (group 2) = 0x04;
 - Transfer_Size_Control (group 1) = 0x03;
 TransferWidth = 3; /*16-bit*/
 - Transfer_Size_Control (group 2) = 0x23;
 TransferWidth = 3; /*16-bit*/
 RequireLevel = 1

4. Run DMA

- Group_1_First_Clear = 0x30;
- Group_2_First_Clear = 0x30;
 ClearPrimaryTC = 1;
 ClearSecondaryTC = 1;
- Call the MINIMITE_DMAProgram function to set up the
 MITE for DMA transfers.

Pass the following parameters to the function:

InputBufferHeadPtr: Pointer to the buffer information linked
 list head node

NumberOfBuffers: Number of buffers to use for DMA transfers

dmaChannel: DMA channel you want to use

direction: DMA transfer direction (Input or Output)

continuous: Perform continuous DMA transfer (False or True)

drqnum: DRQ channel you want to use

- Call MINIMITE_DMAarm to start the DMA transfer.
- Transfer_Count (group 1) = number of points to transfer;
- Transfer_Count (group 2) = number of points to transfer;
- Protocol_Register_1 (group 1) = 0x0F;
- Protocol_Register_1 (group 2) = 0x0F;

- ```

 Numbered = 1
 RunMode = 7 /*run*/

```
- Perform transfer for output group using programmed I/O  
 Declare variables FLAG, DATA  
 for (i = 1 to (TransferCount-16))  
 {  
     FLAG = TransferReady (group 2);  
     While (FLAG == 0) do  
         GroupFIFO (of the output group) = data (16 bit);  
 }
  - Loop to check the status about how many bytes still remain in process. This loop will stop when the number of transfersRemaining equals 0. When DMA finishes transferring data to each buffer, transfersRemaining becomes 0.  
 Declare variable DONE  
 do{  
     Call the MINIMITE\_DMA function get DoneBit to check status using the DONE variable  
     }while (DONE = 0)
  - Print out the data in the buffer.
  - Call MINIMITE\_DMAdisarm to disarm the MITE DMA.
  - Release the memory used for the node list.
  - Set RunModes to 0 to disable handshaking  
     Protocol\_Register\_1 (group1) = 0  
     Protocol\_Register\_1 (group2) = 0
5. Clear any flags set  
 Group\_1\_First\_Clear = 0xFF;  
 Group\_2\_First\_Clear = 0xFF;  
 Group\_1\_Second\_Clear = 0x03;  
 Group\_2\_Second\_Clear = 0x03;

The MITE provides a separate DMA channel for each of the two digital I/O groups (channels). Use DMA controller 1, DRQ line 1, and DACK line 1 for group 1. Use DMA controller 2, DRQ line 2, and DACK line 2 for group 2. Since the card has only two digital I/O groups, there is no need to use DMA controller 0, the third DMA controller.

You can, for example, select the DMA controller and the DRQ number when you call the MINIMITE\_DMAProgram and MINIMITE\_DMAarm functions. The *drqnum* parameter selects both a DRQ line and an associated DACK line. When calling these functions,

set the *dmachannel* and *drqnum* parameters equal to the group number. If you wish to use both groups simultaneously, call `MINIMITE_DMAProgram`, `MINIMITE_DMAarm`, and, when finished, the `MINIMITE_DMAdisarm` function twice: once for each group.

## AT-DIO-32HS

On the AT-DIO-32HS, using DMA allows you to perform transfers in the background, without involving the CPU in every data-point transfer. In addition, DMA is typically faster than interrupt-driven I/O.

For best performance, especially for high-speed pattern generation applications, you can use dual DMA. Dual DMA allows you to avoid pausing at the end of a memory page while the DMA controller waits for reprogramming. In dual DMA, a single operation uses two DMA channels, switching back and forth at the end of each page. You can reprogram one DMA channel while the other one is transferring data. If one group is using both DMA channels, you must use polled or interrupt-driven I/O for the other group, or else leave the other group idle.

Unlike the PCI-DIO-32HS or the PXI-6533, the AT-DIO-32HS requires the support of your system DMA controller to perform DMA. To use DMA, you must allocate one or two 16-bit DMA channels to the AT-DIO-32HS during bus initialization.

To begin the transfer, you must program both the system DMA controller and the DAQ-DIO on the AT-DIO-32HS for DMA. When you enable DMA on the AT-DIO-32HS, the DAQ-DIO begins requesting DMA service from the DMA controller, as long as the `TransferReady` flag is set. In the output case, `TransferReady` is already high, because the FIFOs are empty at this point. You can then program the DMA controller to begin the transfer.

Chapter 6, *DMA Controls*, in the *DAQ-DIO Technical Reference Manual* describes how to program the DAQ-DIO for DMA. On the AT-DIO-32HS, you should set the `DMALength` bitfield to a non-zero value, to ensure that the AT-DIO-32HS does not monopolize the AT bus, causing problems for your computer or for other AT cards in the computer. A `DMALength` value of 3 specifies a maximum of 16 words of consecutive DMA and gives optimal performance.

## AT-DIO-32HS Example 5

Example 5 is similar to example 5 for the PCI-DIO-32HS and PXI-6533, except that you must provide your own code to program, arm, and disarm your system's DMA controller.

## RTSI

---

### PCI-DIO-32HS, PXI-6533, and AT-DIO-32HS

#### Programming the RTSI Bus Interface

A RTSI switch connects signals on the PCI-DIO-32HS, PXI-6533, or the AT-DIO-32HS to the seven RTSI bus trigger lines. The RTSI switch has eight pins labeled A<7..0> connected to DIO 6533 signals, and seven pins labeled B<6..0> connected to the seven RTSI bus trigger lines. Table 2-1 and Table 2-2 list the signals connected to each pin.

**Table 2-1.** RTSI Switch Signal Connections—Side A

| RTSI Switch Pin | Signal Name | Signal Direction (Pattern Direction)                       | Signal Direction (Handshaking, No Pattern Generation) | Signal Direction (No Handshaking) |
|-----------------|-------------|------------------------------------------------------------|-------------------------------------------------------|-----------------------------------|
| A0              | PCLK1       | Unused                                                     | Source (internal clock) or receiver (external clock)  | Source                            |
| A1              | PCLK2       | Unused                                                     | Source (internal clock) or receiver (external clock)  | Source                            |
| A2              | REQ1        | Receiver (external requests) or source (internal requests) | Receiver                                              | Receiver                          |

**Table 2-1.** RTSI Switch Signal Connections—Side A (Continued)

| RTSI Switch Pin | Signal Name | Signal Direction (Pattern Direction)                       | Signal Direction (Handshaking, No Pattern Generation) | Signal Direction (No Handshaking) |
|-----------------|-------------|------------------------------------------------------------|-------------------------------------------------------|-----------------------------------|
| A3              | REQ2        | Receiver (external requests) or source (internal requests) | Receiver                                              | Receiver                          |
| A4              | ACK1        | Receiver (STARTTRIG1)                                      | Source                                                | Source                            |
| A5              | ACK2        | Receiver (STARTTRIG2)                                      | Source                                                | Source                            |
| A6              | STOPTRIG1   | Receiver                                                   | Unused                                                | Receiver                          |
| A7              | STOPTRIG2   | Receiver                                                   | Unused                                                | Receiver                          |

**Table 2-2.** RTSI Switch Signal Connections—Side B

| RTSI Switch Pin | Signal Name | Signal Direction |
|-----------------|-------------|------------------|
| B0              | TRIGGER0    | Bidirectional    |
| B1              | TRIGGER1    | Bidirectional    |
| B2              | TRIGGER2    | Bidirectional    |
| B3              | TRIGGER3    | Bidirectional    |
| B4              | TRIGGER4    | Bidirectional    |
| B5              | TRIGGER5    | Bidirectional    |
| B6              | TRIGGER6    | Bidirectional    |
| B7              | Reserved    | Reserved         |

**PRELIMINARY**

### Programming the RTSI Bus Switch

You can program the RTSI switch to connect any of the signals on Side A to any of the signals on Side B and vice versa. To do this, a 64-bit pattern is shifted into the RTSI switch by writing one bit at a time to the RTSI Serial Register and then writing to the RTSI Parallel Register to load the pattern into the RTSI switch.

The 64-bit pattern is made up of two 32-bit patterns, one for Side A and one for Side B of the RTSI switch. The low-order 32 bits select the signal sources for the Side B pins. The high-order 32 bits select the signal sources for the Side A pins. Each of the 32-bit patterns is made up of eight, 4-bit fields; one for each pin. The 4-bit field selects the signal source and the output enable for the pin. Figure 2-2 shows the bit map of the RTSI switch 64-bit pattern

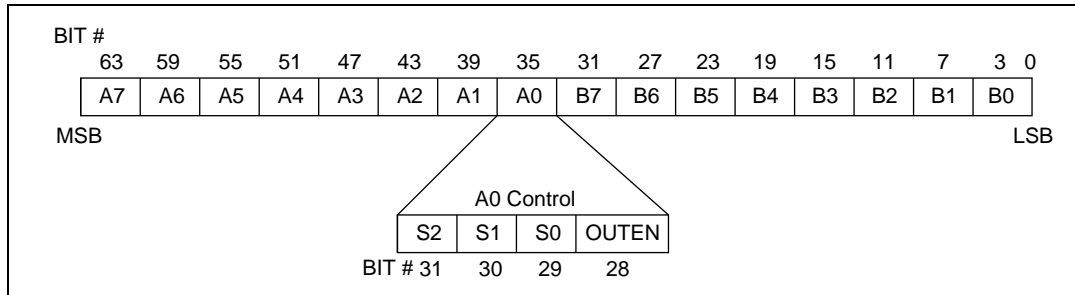


Figure 2-2. RTSI Switch Control Pattern

In Figure 2-2, the fields labeled A7 through A0 and B7 through B0 are the 4-bit control fields for each RTSI switch pin of the same name. The 4-bit control field for pin A0 is shown in Figure 2-2.

The bits labeled S2 through S0 are the signal source selection bits for the pin. You can select one of eight source signals. Pins A7 through A0 can have any of the pins B6 through B0 as signal sources. Pins B6 through B0 can have any of the pins A7 through A0 as signal sources. For example, the pattern 011 for S2 through S0 in the A0 control field selects the signal connected to pin B3 as the signal source for pin A0.

The bit labeled OUTEN is the output enable bit for that pin. If the OUTEN bit is set, the selected source signal drives the pin (the pin acts as an output pin). If the OUTEN bit is cleared, the pin is not driven, regardless of the source signal selected; instead, you can use the pin as an input pin.



For example, if the A3 control field in Figure 2-2 contains the pattern 0111, the signal connected to pin B3 (TRIGGER 3) appears at pin A3. With this arrangement, trigger line 3 can drive the REQ2 signal. Conversely, if the B4 control field contains the pattern 1011, the signal connected to pin A5 (ACK2) appears at pin B4. With this arrangement, DIO 6533 ACK2 signal can drive trigger line 4. In this way, boards connected via the RTSI bus can send signals to each other over the RTSI bus trigger lines.

Pin B7 is reserved for future use. Do not set bit 28, which would drive a signal onto pin B7.

To program the RTSI switch, follow these steps:

1. Calculate the 64-bit pattern based on the desired signal routing
  - a. Clear the OUTEN bit for all input pins and for all unused and reserved pins.
  - b. Specify the signal source pin for all output pins by setting bits S2 through S0 to the source pin number.
  - c. Set the OUTEN bit for all output pins.
2. For  $i = 0$  to 63, follow these steps
  - a. Copy bit  $i$  of the 64-bit pattern to bit 0 of an 8-bit temporary variable.
  - b. Write the temporary variable to the RTSI Serial Register (8-bit write).
3. Write 0 to the RTSI Parallel Register (8-bit write). This operation loads the 64-bit pattern into the RTSI switch. At this point, the new signal routing goes into effect

Perform step 2 by writing the low-order 8 bits of the 64-bit pattern to the RTSI Serial Register, then shifting the 64-bit pattern right once, and repeating this two-step operation a total of 64 times. Only bit 0 of the word written to the RTSI Serial Register is used. The higher-order bits are ignored.

### Initializing the RTSI Bus Switch

Use the following sequence to initialize the RTSI bus switch. This sequence configures all signals to the RTSI bus switch as input (receive) signals rather than output (source) signals. Output signals may cause undesirable results. All writes are 8-bit write operations.

1. Load the RTSI switch with the program data pattern  
For  $i = 0$  to 63 (decimal), write 0 to the RTSI Serial Register
2. Write to the RTSI Parallel Register to load the pattern into the RTSI switch

Write 0 to the RTSI Parallel Register

At startup, the RTSI bus switch is automatically initialized.

### RTSI Bus Register Group

The two registers making up the RTSI Bus Register group program the DIO 6533 RTSI switch for routing signals on the RTSI bus trigger lines to and from DIO 6533 REQ, ACK (STARTTRIG), PCLK, and STOPTRIG signal lines.

The following information provides the RTSI\_Serial\_Reg and RTSI\_Parallel\_Reg registers bit descriptions.

**RTSI\_SERIAL\_REG**

RTSI\_Serial\_Reg contains one bit, RSI, that is a serial input to the RTSI switch. You must write to RSI 64 times to load the internal 64-bit RTSI control register.

Address: Base address + 124 (decimal)

Type: Write-only

Word Size: 8-bit

Bit Map:

|   |   |   |   |   |   |   |     |
|---|---|---|---|---|---|---|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0   |
| X | X | X | X | X | X | X | RSI |

| Bit | Name | Description                                                                                                                                                                                                                                                                                                                                                                                            |
|-----|------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-1 | X    | Don't Care Bits                                                                                                                                                                                                                                                                                                                                                                                        |
| 0   | RSI  | <p>RTSI Switch Serial Input</p> <p>This bit is the serial input to the RTSI switch internal 64-bit control register. The data in the control register configures the signal switching to and from the RTSI bus trigger lines. You must write the RSI bit 64 times to shift in the 64 bits of routing data. You can then write to the RTSI_Parallel_Reg to put the new control pattern into effect.</p> |

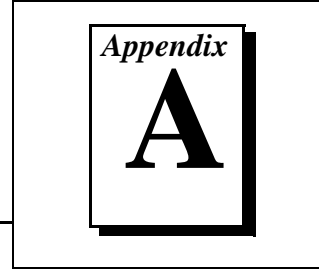
### **RTSI\_PARALLEL\_REG**

Writing to RTSI\_Parallel\_Reg loads the contents of the RTSI\_Serial\_Reg into the RTSI Switch Control Register, thereby updating the RTSI switch routing pattern. Write to RTSI\_Parallel\_Reg after shifting the 64-bit routing pattern into RTSI\_Serial\_Reg.

|            |                              |
|------------|------------------------------|
| Address:   | Base address + 126 (decimal) |
| Type:      | Write-only                   |
| Word Size: | 8-bit                        |
| Bit Map:   | Not applicable, no bits used |

# Customer Communication

---



For your convenience, this appendix contains forms to help you gather the information necessary to help us solve your technical problems and a form you can use to comment on the product documentation. When you contact us, we need the information on the Technical Support Form and the configuration form, if your manual contains one, about your system configuration to answer your questions as quickly as possible.

National Instruments has technical assistance through electronic, fax, and telephone systems to quickly provide the information you need. Our electronic services include a bulletin board service, an FTP site, a Fax-on-Demand system, and e-mail support. If you have a hardware or software problem, first try the electronic support systems. If the information available on these systems does not answer your questions, we offer fax and telephone support through our technical support centers, which are staffed by applications engineers.

## Electronic Services



### Bulletin Board Support

National Instruments has BBS and FTP sites dedicated for 24-hour support with a collection of files and documents to answer most common customer questions. From these sites, you can also download the latest instrument drivers, updates, and example programs. For recorded instructions on how to use the bulletin board and FTP services and for BBS automated information, call (512) 795-6990. You can access these services at:

- United States: (512) 794-5422  
Up to 14,400 baud, 8 data bits, 1 stop bit, no parity
- United Kingdom: 01635 551422  
Up to 9,600 baud, 8 data bits, 1 stop bit, no parity
- France: 01 48 65 15 59  
Up to 9,600 baud, 8 data bits, 1 stop bit, no parity



### FTP Support

To access our FTP site, log on to our Internet host, `ftp.natinst.com`, as anonymous and use your Internet address, such as `joesmith@anywhere.com`, as your password. The support files and documents are located in the `/support` directories.



## Fax-on-Demand Support

Fax-on-Demand is a 24-hour information retrieval system containing a library of documents on a wide range of technical information. You can access Fax-on-Demand from a touch-tone telephone at (512) 418-1111.





## E-Mail Support (currently U.S. only)

You can submit technical support questions to the applications engineering team through e-mail at the Internet address listed below. Remember to include your name, address, and phone number so we can contact you with solutions and suggestions.

[support@natinst.com](mailto:support@natinst.com)

## Telephone and Fax Support

National Instruments has branch offices all over the world. Use the list below to find the technical support number for your country. If there is no National Instruments office in your country, contact the source from which you purchased your software to obtain support.

|                  |  Telephone |  Fax |
|------------------|---------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------|
| Australia        | 03 9879 5166                                                                                | 03 9879 6277                                                                           |
| Austria          | 0662 45 79 90 0                                                                             | 0662 45 79 90 19                                                                       |
| Belgium          | 02 757 00 20                                                                                | 02 757 03 11                                                                           |
| Canada (Ontario) | 905 785 0085                                                                                | 905 785 0086                                                                           |
| Canada (Quebec)  | 514 694 8521                                                                                | 514 694 4399                                                                           |
| Denmark          | 45 76 26 00                                                                                 | 45 76 26 02                                                                            |
| Finland          | 09 725 725 11                                                                               | 09 725 725 55                                                                          |
| France           | 01 48 14 24 24                                                                              | 01 48 14 24 14                                                                         |
| Germany          | 089 741 31 30                                                                               | 089 714 60 35                                                                          |
| Hong Kong        | 2645 3186                                                                                   | 2686 8505                                                                              |
| Israel           | 03 5734815                                                                                  | 03 5734816                                                                             |
| Italy            | 02 413091                                                                                   | 02 41309215                                                                            |
| Japan            | 03 5472 2970                                                                                | 03 5472 2977                                                                           |
| Korea            | 02 596 7456                                                                                 | 02 596 7455                                                                            |
| Mexico           | 5 520 2635                                                                                  | 5 520 3282                                                                             |
| Netherlands      | 0348 433466                                                                                 | 0348 430673                                                                            |
| Norway           | 32 84 84 00                                                                                 | 32 84 86 00                                                                            |
| Singapore        | 2265886                                                                                     | 2265887                                                                                |
| Spain            | 91 640 0085                                                                                 | 91 640 0533                                                                            |
| Sweden           | 08 730 49 70                                                                                | 08 730 43 70                                                                           |
| Switzerland      | 056 200 51 51                                                                               | 056 200 51 55                                                                          |
| Taiwan           | 02 377 1200                                                                                 | 02 737 4644                                                                            |
| United Kingdom   | 01635 523545                                                                                | 01635 523154                                                                           |
| United States    | 512 795 8248                                                                                | 512 794 5678                                                                           |

**P R E L I M I N A R Y**

# Technical Support Form

Photocopy this form and update it each time you make changes to your software or hardware, and use the completed copy of this form as a reference for your current configuration. Completing this form accurately before contacting National Instruments for technical support helps our applications engineers answer your questions more efficiently.

If you are using any National Instruments hardware or software products related to this problem, include the configuration forms from their user manuals. Include additional pages if necessary.

Name \_\_\_\_\_

Company \_\_\_\_\_

Address \_\_\_\_\_

Fax ( \_\_\_\_ ) \_\_\_\_\_ Phone ( \_\_\_\_ ) \_\_\_\_\_

Computer brand \_\_\_\_\_ Model \_\_\_\_\_ Processor \_\_\_\_\_

Operating system (include version number) \_\_\_\_\_

Clock speed \_\_\_\_\_ MHz RAM \_\_\_\_\_ MB Display adapter \_\_\_\_\_

Mouse \_\_\_yes \_\_\_no Other adapters installed \_\_\_\_\_

Hard disk capacity \_\_\_\_\_ MB Brand \_\_\_\_\_

Instruments used \_\_\_\_\_

National Instruments hardware product model \_\_\_\_\_ Revision \_\_\_\_\_

Configuration \_\_\_\_\_

National Instruments software product \_\_\_\_\_ Version \_\_\_\_\_

Configuration \_\_\_\_\_

The problem is: \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

List any error messages: \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

The following steps reproduce the problem: \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

**P R E L I M I N A R Y**

**PRELIMINARY**



# DIO 6533 Register-Level Programmer Manual

## Hardware and Software Configuration Form

Record the settings and revisions of your hardware and software on the line to the right of each item. Complete a new copy of this form each time you revise your software or hardware configuration, and use this form as a reference for your current configuration. Completing this form accurately before contacting National Instruments for technical support helps our applications engineers answer your questions more efficiently.

### National Instruments Products

DAQ hardware \_\_\_\_\_

Interrupt level of hardware \_\_\_\_\_

DMA channels of hardware \_\_\_\_\_

Base I/O address of hardware \_\_\_\_\_

Programming choice \_\_\_\_\_

Software and version \_\_\_\_\_

Other boards in system \_\_\_\_\_

Base I/O address of other boards \_\_\_\_\_

DMA channels of other boards \_\_\_\_\_

Interrupt level of other boards \_\_\_\_\_

### Other Products

Computer make and model \_\_\_\_\_

Microprocessor \_\_\_\_\_

Clock frequency or speed \_\_\_\_\_

Type of video board installed \_\_\_\_\_

Operating system version \_\_\_\_\_

Operating system mode \_\_\_\_\_

Programming language \_\_\_\_\_

Programming language version \_\_\_\_\_

Other boards in system \_\_\_\_\_

Base I/O address of other boards \_\_\_\_\_

DMA channels of other boards \_\_\_\_\_

Interrupt level of other boards \_\_\_\_\_

**P R E L I M I N A R Y**

**PRELIMINARY**

# Documentation Comment Form

National Instruments encourages you to comment on the documentation supplied with our products. This information helps us provide quality products to meet your needs.

**Title:** *DIO 6533 Register-Level Programmer Manual*

**Edition Date:** December 1997

**Part Number:** 341329A-01

Please comment on the completeness, clarity, and organization of the manual.

---

---

---

---

---

---

---

---

If you find errors in the manual, please record the page numbers and describe the errors.

---

---

---

---

---

---

---

---

Thank you for your help.

Name \_\_\_\_\_

Title \_\_\_\_\_

Company \_\_\_\_\_

Address \_\_\_\_\_

Phone ( \_\_\_\_ ) \_\_\_\_\_ Fax ( \_\_\_\_ ) \_\_\_\_\_

**Mail to:** Technical Publications  
National Instruments Corporation  
6504 Bridge Point Parkway  
Austin, TX 78730-5039

**Fax to:** Technical Publications  
National Instruments Corporation  
(512) 794-5678

**P R E L I M I N A R Y**

**PRELIMINARY**



## *Glossary*

| <b>Prefix</b> | <b>Meanings</b> | <b>Value</b> |
|---------------|-----------------|--------------|
| n-            | nano-           | $10^{-9}$    |
| $\mu$ -       | micro-          | $10^{-6}$    |
| m-            | milli-          | $10^{-3}$    |

### **Numbers/Symbols**

|          |                       |
|----------|-----------------------|
| %        | percent               |
| +        | positive of, or plus  |
| -        | negative of, or minus |
| /        | per                   |
| °        | degree                |
| $\Omega$ | ohm                   |

### **A**

|         |                                                                                       |
|---------|---------------------------------------------------------------------------------------|
| A       | amperes                                                                               |
| AC      | alternating current                                                                   |
| ACK     | acknowledge input signal                                                              |
| address | character code that identifies a specific location (or series of locations) in memory |

**PRELIMINARY**

*Glossary*

**ASIC** Application-Specific Integrated Circuit—a proprietary semiconductor component designed and manufactured to perform a set of specific functions for a specific customer

**asynchronous** (1) hardware—a property of an event that occurs at an arbitrary time, without synchronization to a reference clock (2) software—a property of a function that begins an operation and returns prior to the completion or termination of the operation

**B**

**b** bit—one binary digit, either 0 or 1

**B** byte—eight related bits of data, an eight-bit binary number. Also used to denote the amount of memory required to store one byte of data

**base address** a memory address that serves as the starting address for programmable registers. All other addresses are located by adding to the base address

**BIOS** basic input/output system- BIOS functions are the fundamental level of any PC or compatible computer. BIOS functions embody the basic operations needed for successful use of the computer’s hardware resources

**bitfield** a group of contiguous bits that jointly perform a function

**bus** the group of conductors that interconnect individual circuitry in a computer. Typically, a bus is the expansion vehicle to which I/O or other devices are connected. Examples of PC buses are the AT bus, NuBus, Micro Channel, and EISA bus

**C**

**C** Celsius

**clock** hardware component that controls timing for reading from or writing to groups

**CMOS** complementary metal-oxide semiconductor

**CPU** central processing unit

|                          |                                                                                                                                                      |
|--------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------|
| crosstalk                | an unwanted signal on one channel due to an input on a different channel                                                                             |
| current drive capability | the amount of current a digital or analog output channel is capable of sourcing or sinking while still operating within voltage range specifications |
| current sinking          | the ability of a DAQ board to dissipate current for analog or digital output signals                                                                 |
| current sourcing         | the ability of a DAQ board to supply current for analog or digital output signals                                                                    |

## D

|                      |                                                                                                                                                                                                                                                                                                                                                                                        |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| DAQ                  | data acquisition—(1) collecting and measuring electrical signals from sensors, transducers, and test probes or fixtures and inputting them to a computer for processing; (2) collecting and measuring the same kinds of electrical signals with A/D and/or DIO boards plugged into a computer, and possibly generating control signals with D/A and/or DIO boards in the same computer |
| DAR                  | direct address register—stores default values for simple operations                                                                                                                                                                                                                                                                                                                    |
| DC                   | direct current                                                                                                                                                                                                                                                                                                                                                                         |
| device               | a plug-in data acquisition board, card, or pad that can contain multiple channels and conversion devices. Plug-in boards, PCMCIA cards, and devices such as the DAQPad-1200, which connects to your computer parallel port, are all examples of DAQ devices. SCXI modules are distinct from devices, with the exception of the SCXI-1200, which is a hybrid                            |
| digital input group  | a collection of digital input ports. You can associate each group with its own clock rates, handshaking modes, buffer configurations, and so on. A port cannot belong to more than one group                                                                                                                                                                                           |
| digital output group | a collection of digital output ports. You can associate each group with its own clock rates, handshaking modes, buffer configurations, and so on. A port cannot belong to more than one group                                                                                                                                                                                          |
| digital trigger      | a TTL level signal having two discrete levels—a high and a low level                                                                                                                                                                                                                                                                                                                   |
| DIO                  | digital input/output                                                                                                                                                                                                                                                                                                                                                                   |

# PRELIMINARY

*Glossary*

|         |                                                                                                                                                                                                                                                                                                 |
|---------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| DLL     | dynamic link library—a software module in Microsoft Windows containing executable code and data that can be called or used by Windows applications or other DLLs. Functions and data in a DLL are loaded and linked at run time when they are referenced by a Windows application or other DLLs |
| DMA     | direct memory access—a method by which data can be transferred to/from computer memory from/to a device or memory on the bus while the processor does something else. DMA is the fastest method of transferring data to/from computer memory                                                    |
| drivers | software that controls a specific hardware device such as a DAQ board or a GPIB interface board                                                                                                                                                                                                 |

**E**

|                  |                                                                                                                       |
|------------------|-----------------------------------------------------------------------------------------------------------------------|
| EEPROM           | electrically erasable programmable read-only memory—ROM that can be erased with an electrical signal and reprogrammed |
| EISA             | extended industry standard architecture                                                                               |
| event            | the condition or state of an analog or digital signal                                                                 |
| external trigger | a voltage pulse from an external source that triggers an event such as A/D conversion                                 |

**F**

|      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| FIFO | first-in first-out memory buffer—the first data stored is the first data sent to the acceptor. FIFOs are often used on DAQ devices to temporarily store incoming or outgoing data until that data can be retrieved or output. For example, an analog input FIFO stores the results of A/D conversions until the data can be retrieved into system memory, a process that requires the servicing of interrupts and often the programming of the DMA controller. This process can take several milliseconds in some cases. During this time, data accumulates in the FIFO for future retrieval. With a larger FIFO, longer latencies can be tolerated. In the case of analog output, a FIFO permits faster update rates, because the waveform data can be stored on the FIFO ahead of time. This again reduces the effect of latencies associated with getting the data from system memory to the DAQ device |
|------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

**P R E L I M I N A R Y**



ft feet

## G

glitch a brief, unwanted change, or disturbance, in a signal level

GNP ground

## H

h hour

handshaked digital I/O a type of digital acquisition/generation where a device or module accepts or transfers data after a digital pulse has been received. Also called latched digital I/O

hardware the physical components of a computer system, such as the circuit boards, plug-in boards, chassis, enclosures, peripherals, cables, and so on

hardware triggering a form of triggering where you set the start time of an acquisition and gather data at a known position in time relative to a trigger signal

hex hexadecimal

Hz hertz—the number of scans read or updates written per second

## I

IBM International Business Machines

IC integrated circuit

ID identification

in. inches

interrupt a computer signal indicating that the CPU should suspend its current task to service a designated activity

interrupt level the relative priority at which a device can interrupt

## Glossary

|                 |                                                                                                                                                                           |
|-----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| I/O             | input/output—the transfer of data to/from a computer system involving communications channels, operator interface devices, and/or data acquisition and control interfaces |
| I <sub>OH</sub> | current, output high                                                                                                                                                      |
| I <sub>OL</sub> | current, output low                                                                                                                                                       |
| IRQ             | interrupt request                                                                                                                                                         |
| ISA             | industry standard architecture                                                                                                                                            |

## K

|          |                                                                      |
|----------|----------------------------------------------------------------------|
| kbytes/s | a unit for data transfer that means 1,000 or 10 <sup>3</sup> bytes/s |
| kS       | 1,000 samples                                                        |
| Kword    | 1,024 words of memory                                                |

## L

|                     |                                                                                                                                                                         |
|---------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LabVIEW             | laboratory virtual instrument engineering workbench                                                                                                                     |
| latched digital I/O | a type of digital acquisition/generation where a device or module accepts or transfers data after a digital pulse has been received. Also called handshaked digital I/O |
| LED                 | light-emitting diode                                                                                                                                                    |
| linearity           | the adherence of device response to the equation $R = KS$ , where $R$ = response, $S$ = stimulus, and $K$ = a constant                                                  |
| LKAR                | link address register—stores the physical address of the next node                                                                                                      |
| LSB                 | least significant bit                                                                                                                                                   |

## M

|     |                                                              |
|-----|--------------------------------------------------------------|
| m   | meters                                                       |
| MAR | memory address register—stores the buffer's physical address |

**PRELIMINARY**

|      |                                                                                                                                                                                                                   |
|------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| max  | maximum                                                                                                                                                                                                           |
| MB   | megabytes of memory                                                                                                                                                                                               |
| min  | minimum                                                                                                                                                                                                           |
| min. | minute                                                                                                                                                                                                            |
| MITE | MXI Interfaces to Everything is a custom ASIC designed by National Instruments that implements the PCI bus interface. The MITE supports bus mastering for high speed data transfers over the PCI bus              |
| MSB  | most significant bit                                                                                                                                                                                              |
| mux  | multiplexer—a switching device with multiple inputs that sequentially connects each of its inputs to its output, typically at high speeds, in order to measure several signals with a single analog input channel |

## N

|                        |                                                                                                                                                                                                                                                                                                                                                                                |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| NI-DAQ                 | National Instruments driver software for DAQ hardware                                                                                                                                                                                                                                                                                                                          |
| noise                  | an undesirable electrical signal—Noise comes from external sources such as the AC power line, motors, generators, transformers, fluorescent lights, soldering irons, CRT displays, computers, electrical storms, welders, radio transmitters, and internal sources such as semiconductors, resistors, and capacitors. Noise corrupts signals you are trying to send or receive |
| nonlatched digital I/O | a type of digital acquisition/generation where LabVIEW updates the digital lines or port states immediately or returns the digital value of an input line. Also called immediate digital I/O or non-handshaking                                                                                                                                                                |

## O

|                  |                                                                                                                                                   |
|------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| operating system | base-level software that controls a computer, runs programs, interacts with users, and communicates with installed hardware or peripheral devices |
| OS               | <i>see</i> operating system                                                                                                                       |

## P

|                         |                                                                                                                                                                                                                                                                                                                                                |
|-------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| pattern generation      | a type of handshaked (latched) digital I/O in which internal counters generate the handshaked signal, which in turn initiates a digital transfer. Because counters output digital pulses at a constant rate, this means you can generate and retrieve patterns at a constant rate because the handshaked signal is produced at a constant rate |
| PC Card                 | a credit-card-sized expansion card that fits in a PCMCIA slot; often referred to as a PCMCIA card                                                                                                                                                                                                                                              |
| PCI                     | Peripheral Component Interconnect—a high-performance expansion bus architecture originally developed by Intel to replace ISA and EISA. It is achieving widespread acceptance as a standard for PCs and workstations; it offers a theoretical maximum transfer rate of 132 Mbytes/s                                                             |
| PCLK                    | peripheral clock signal                                                                                                                                                                                                                                                                                                                        |
| PCMCIA                  | an expansion bus architecture that has found widespread acceptance as a de facto standard in notebook-size computers. It originated as a specification for add-on memory cards written by the Personal Computer Memory Card International Association                                                                                          |
| peripheral device       | an external device that a card controls, monitors, tests, or communicates with                                                                                                                                                                                                                                                                 |
| Plug and Play devices   | devices that do not require dip switches or jumpers to configure resources on the devices—also called switchless devices                                                                                                                                                                                                                       |
| Plug and Play ISA       | a specification prepared by Microsoft, Intel, and other PC-related companies that will result in PCs with plug-in boards that can be fully configured in software, without jumpers or switches on the boards                                                                                                                                   |
| port                    | (1) a communications connection on a computer or a remote controller<br>(2) a digital port, consisting of four or eight lines of digital input and/or output                                                                                                                                                                                   |
| posttrigger acquisition | the technique used on a DAQ board to acquire a programmed number of samples after trigger conditions are met                                                                                                                                                                                                                                   |
| pretrigger acquisition  | the technique used on a DAQ board to keep a continuous buffer filled with data, so that when the trigger conditions are met, the sample includes the data leading up to the trigger condition                                                                                                                                                  |

|                |                                                                                                                                                                          |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| programmed I/O | the standard method a CPU uses to access an I/O device each byte of data is read or written by the CPU                                                                   |
| protocol       | the exact sequence of bits, characters, and control codes used to transfer data between computers and peripherals through a communications channel, such as the GPIB bus |

## R

|          |                                                                                                                                                                                               |
|----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| RAM      | random-access memory                                                                                                                                                                          |
| REQ      | request signal                                                                                                                                                                                |
| RGND     | reserved ground                                                                                                                                                                               |
| rms      | root mean square—the square root of the average value of the square of the instantaneous signal amplitude; a measure of signal amplitude                                                      |
| RTSI bus | real-time system integration bus—the National Instruments timing bus that connects DAQ boards directly, by means of connectors on top of the boards, for precise synchronization of functions |

## S

|                     |                                                                                                                                                                                                        |
|---------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| s                   | seconds                                                                                                                                                                                                |
| S                   | samples                                                                                                                                                                                                |
| settling time       | the amount of time required for a voltage to reach its final value within specified limits                                                                                                             |
| S/s                 | samples per second—used to express the rate at which a DAQ board samples an analog signal                                                                                                              |
| STARTTRIG           | start trigger signal                                                                                                                                                                                   |
| STOPTRIG            | stop trigger signal                                                                                                                                                                                    |
| strobed digital I/O | a type of digital input or output in which hardware uses timing signals to regulate the rate of input or output. Types of strobed digital I/O include <i>handshaking</i> and <i>pattern generation</i> |

## Glossary

|                   |                                                                                                                                                                                              |
|-------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| switchless device | devices that do not require dip switches or jumpers to configure resources on the devices—also called Plug and Play devices                                                                  |
| synchronous       | (1) hardware—a property of an event that is synchronized to a reference clock (2) software—a property of a function that begins an operation and returns only when the operation is complete |

## T

|               |                                                                                                                                                                                          |
|---------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| TCR           | transfer count register—stores the number of points to transfer                                                                                                                          |
| transfer rate | the rate, measured in bytes/s, at which data is moved from source to destination after software initialization and set up operations; the maximum rate at which the hardware can operate |
| trigger       | any event that causes or starts some form of data capture                                                                                                                                |
| tri-state     | a third output state, other than high or low, in which the output is undriven                                                                                                            |
| TTL           | transistor-transistor logic                                                                                                                                                              |

## U

|                       |                                                                                                                                                                                                                                                                                                                                     |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| unstrobed digital I/O | a type of digital input or output in which software reads or writes the digital line or port states directly, without using any handshaking or hardware-controlled timing functions. Also called <i>immediate</i> , <i>nonhandshaking</i> , or <i>unlatched</i> digital I/O                                                         |
| update                | the output equivalent of a scan. One or more analog or digital output samples. Typically, the number of output samples in an update is equal to the number of channels in the output group. For example, one pulse from the update clock produces one update which sends one new sample to every analog output channel in the group |

## V

|                 |                      |
|-----------------|----------------------|
| V               | volts                |
| VDC             | volts direct current |
| V <sub>IH</sub> | volts, input high    |

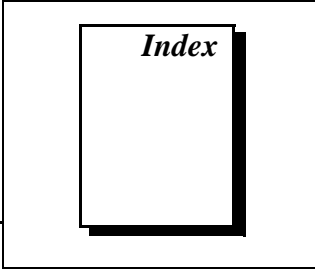
|           |                    |
|-----------|--------------------|
| $V_{IL}$  | volts, input low   |
| $V_{in}$  | volts in           |
| $V_{OH}$  | volts, output high |
| $V_{OL}$  | volts, output low  |
| $V_{ref}$ | reference voltage  |

## W

|                 |                                                                                                                                                                                                                                                                                                                                                           |
|-----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| wire            | data path between nodes                                                                                                                                                                                                                                                                                                                                   |
| wired-OR        | a type of output driver that provides sink current but little or no source current, and is typically used with a pull-up resistor to provide source current. If you connect two or more wired-OR outputs together, any one of the output drivers can drive the resulting connection low. Also called an <i>open-collector</i> or <i>open-drain driver</i> |
| word            | the standard number of bits that a processor or memory manipulates at one time. Microprocessors typically use 8, 16, or 32-bit words                                                                                                                                                                                                                      |
| working voltage | the highest voltage that should be applied to a product in normal use, normally well under the breakdown voltage for safety margin. See also Breakdown Voltage                                                                                                                                                                                            |

**PRELIMINARY**





## **A**

Level 1

Level 2

Level 3

Level 4

Level 5

**PRELIMINARY**

*Index*

**PRELIMINARY**