GX5642

Bi-directional Differential LVDS-TTL I/O Board

User's Guide

Last updated June 8, 2007

GEOTEST

MARVIN TEST SYSTEMS, INC.

Safety and Handling

Each product shipped by Geotest is carefully inspected and tested prior to shipping. The shipping box provides protection during shipment, and can be used for storage of both the hardware and the software when they are not in use.

The circuit boards are extremely delicate and require care in handling and installation. Do not remove the boards from their protective plastic coverings or from the shipping box until you are ready to install the boards into your computer.

If a board is removed from the computer for any reason, be sure to store it in its original shipping box. Do not store boards on top of workbenches or other areas where they might be susceptible to damage or exposure to strong electromagnetic or electrostatic fields. Store circuit boards in protective anti-electrostatic wrapping and away from electromagnetic fields.

Be sure to make a single copy of the software diskette for installation. Store the original diskette in a safe place away from electromagnetic or electrostatic fields. Return compact disks (CD) to their protective case or sleeve and store in the original shipping box or other suitable location.

Warranty

Geotest products are warranted against defects in materials and workmanship for a period of 12 months. Software products and accessories are warranted for 3 months. Unless covered by software support or maintenance agreement. Geotest shall repair or replace (at its discretion) any defective product during the stated warranty period. The software warranty includes any revisions or new versions released during the warranty period. Revisions and new versions may be covered by a software support agreement. If you need to return a board, please contact Geotest Customer Technical Services department via http://www.geotestinc.com/magic the Geotest on-line support system.

If You Need Help

Visit our web site at http://www.geotestinc.com for more information about Geotest products, services and support options. Our web site contains sections describing support options and application notes, as well as a download area for downloading patches, example, patches and new or revised instrument drivers. To submit a support issue including suggestion, bug report or question please use the following link: http://www.geotestinc.com/magic

You can also use Geotest technical support phone line (949) 263-2222. This service is available between 7:30 AM and 5:30 PM Pacific Standard Time.

Disclaimer

In no event shall Geotest or any of its representatives be liable for any consequential damages whatsoever (including unlimited damages for loss of business profits, business interruption, loss of business information, or any other losses) arising out of the use of or inability to use this product, even if Geotest has been advised of the possibility for such damages.

Copyright

Trademarks

ATEasy®	Geotest – MTS Inc.
C++ Builder, Borland C++, Pascal and Delphi	Borland Corporation
Microsoft Developer Studio, Microsoft Visual C++, Microsoft Visual Basic,, .NET, Windows 95, 98, NT, ME, 2000 and XP	Microsoft Corporation

All other trademarks are the property of their respective owners.

.

Table of Contents

Chapter 1 - Introduction

Manual Scope and Organization

Manual Scope

The purpose of this manual is to provide all the necessary information to install, use, and maintain the GX5642 instrument. This manual assumes the reader has a general knowledge of PC based computers, Windows operating systems, and some understanding of digital I/O.

This manual also provides programming information using the GX5642 driver (referred in this manual **GXPIO**). Therefore, good understanding of programming development tools and languages may be necessary.

Manual Organization

The GX5642 manual is organized in the following manner:

Chapter	Content
Chapter 1 - Introduction	Introduces the GX5642 manual. Lists all the supported board and shows warning conventions used in the manual.
Chapter 2 – Overview	Describes the GX5642 features, board description, its architecture, specifications and the panel description and operation.
Chapter 3 –Installation and Connections	Provides instructions on how to install a GX5642 board and the GXPIO software.
Chapter 4 – Programming the Board	Provides a list of the GXPIO software driver files, general purpose and generic driver functions, and programming methods. Discusses supported application development tools and programming examples.
Chapter 5 – Functions Reference	Provides a list of the GX5642 driver functions. Each function description provides syntax, parameters, and any special programming comments.

Conventions Used in this Manual

Symbol Convention	Meaning
	Static Sensitive Electronic Devices. Handle Carefully.
	Warnings that may pose a personal danger to your health. For example, shock hazard.
	Cautions where computer components may be damaged if not handled carefully.
	Tips that aid you in your work.

Formatting Convention	Meaning
`Monospaced Text`	Examples of field syntax and programming samples.
Bold type	Words or characters you type as the manual instructs. For example: function or panel names.
Italic type	Specialized terms. Titles of other reference books. Placeholders for items you must supply, such as function parameters

Chapter 2 - Overview

Introduction

The GX5642 is a 3U PXI instrument card that consists of 64 Bi-Directional TTL-Differential I/O channels. Each channel has two ports: TTL and Differential. Each channel can be individually set to operate in one of two modes: Conversion or Static I/O. In Static I/O mode the GX5642 supports 128 individual digital inputs or outputs, 64 TTL and 64 Differential. In Conversion mode the GX5642 support 64 individual Differential channels. The GX5642 can be configured via jumper to operate in one of two modes: Independent or Software Controlled.

When the card is configured to Independent mode the PXI interface is disabled and all 64 channels operate in Conversion mode. In this mode each channel conversion operation is predefined using DIP-Switches. Each channel can be predefined to convert TTL to Differential or Differential to TTL. The channels conversion settings will be loaded automatically upon power up.

When the card is configured to Software Controlled mode each channel can be individually programmed to operate in Conversion mode or Static I/O mode. In Conversion mode each channel can be programmed to convert TTL to Differential or Differential LVDS to TTL. In Static I/O mode, each channel's port can be programmed as follows:

- TTL port input, Differential port input.
- TTL port output, Differential port input.
- TTL port input, Differential port output.
- TTL port output, Differential port output.

In Software Controlled mode each channel's TTL and Differential outputs can be individually enabled or disabled.

Features

The GX5642 is a 3U PXI instrument card with two operation modes:

Independent mode features (JP2 Installed)

- Total of 64 individual channels each capable of converting TTL to Differential or Differential to TTL.
- Each channel conversion operation is predefined using DIP-Switches. Converting TTL to Differential or Differential to TTL.
- Channels conversion settings automatically loaded on power up.
- The board is disconnected from the PXI interface and operated without any interaction with the host computer.
- J1 TTL connector has 40 TTL channels assigned to it and J2 TTL connector has 24 TTL channels. The Differential connectors, J3 and J4, have 32 channels each.
- A 100-Ohm terminator for each Differential I/O terminals.

Software Controlled mode features (JP2 Not Installed)

- Each channel can be programmed to operate in Conversion or Static I/O mode.
- In Conversion mode the board has 64 individual channels.
- In Conversion mode each channel can be programmed to convert Differential to TTL or TTL to Differential.
- In Static I/O mode the board has 64 individual channels each with TTL and Differential ports for a total of 128 individual Digital I/O ports.

- In Static I/O mode each channel's port direction can be individually set to output or input, e.g. Channel 1 Differential port as output and Channel 1 TTL port as input.
- TTL and Differential input signals can be monitored in all modes of operations.
- Channels predefined DIP-Switch conversion operation can be loaded at any time overriding current settings (predefined defaults).
- Each channel's TTL and Differential output port can be individually enabled/disabled through software control.
- On power up all 64 channels' outputs are disabled.
- A 100-Ohm terminator for each Differential I/O terminals.

Applications

- Automatic Test Equipment (ATE) and Functional Test
- Data Acquisition
- Process Control
- Factory Automation

Board Description

The GX5642 is a 3U PXI instrument card that consists of 64 bi-directional TTL I/O channels and 64 bi-directional I/O channels. The GX5642 can be configured via jumper (JP2) to operate in one of two modes: Independent or Software Controlled. The board has four 68-pin VHDCI connectors two for TTL and two for Differential channels.

Independent mode (JP2 installed)

When the card is configured to Independent mode the board is disconnected from the PXI interface, operated without any interaction with the host computer, and all 64 channels operate in Conversion mode only. Each channel conversion operation is predefined using DIP-Switches specifying conversion of TTL to Differential or Differential to TTL. All 64 channels conversion settings are automatically loaded upon power up. In this mode the J1 TTL connector has 40 TTL channels assigned to it and J2 TTL connector has 24 TTL channels. The Differential connectors, J3 and J4, have 32 channels each.

Software Controlled mode (JP2 not installed)

When the card is configured to Software Controlled mode each channel can be individually set to operate in Conversion or Static I/O mode. In Conversion mode each channel can be programmed to convert Differential to TTL or TTL to Differential. The board has 64 DIP-Switches, one for each channel to enable a predefine conversion mode per channel. The channels' predefined DIP-Switch conversion modes can be loaded at any time overriding current settings (predefined defaults).

In Static I/O mode the GX5642 support 128 individual digital inputs or outputs, 64 TTL and 64 Differential. In this mode each channel TTL port can be set as input TTL port or output TTL port. In this mode the channel Differential port can be set as input Differential port or output Differential port.

In Software Controlled mode each channel's TTL and Differential outputs can be individually enabled or disabled.

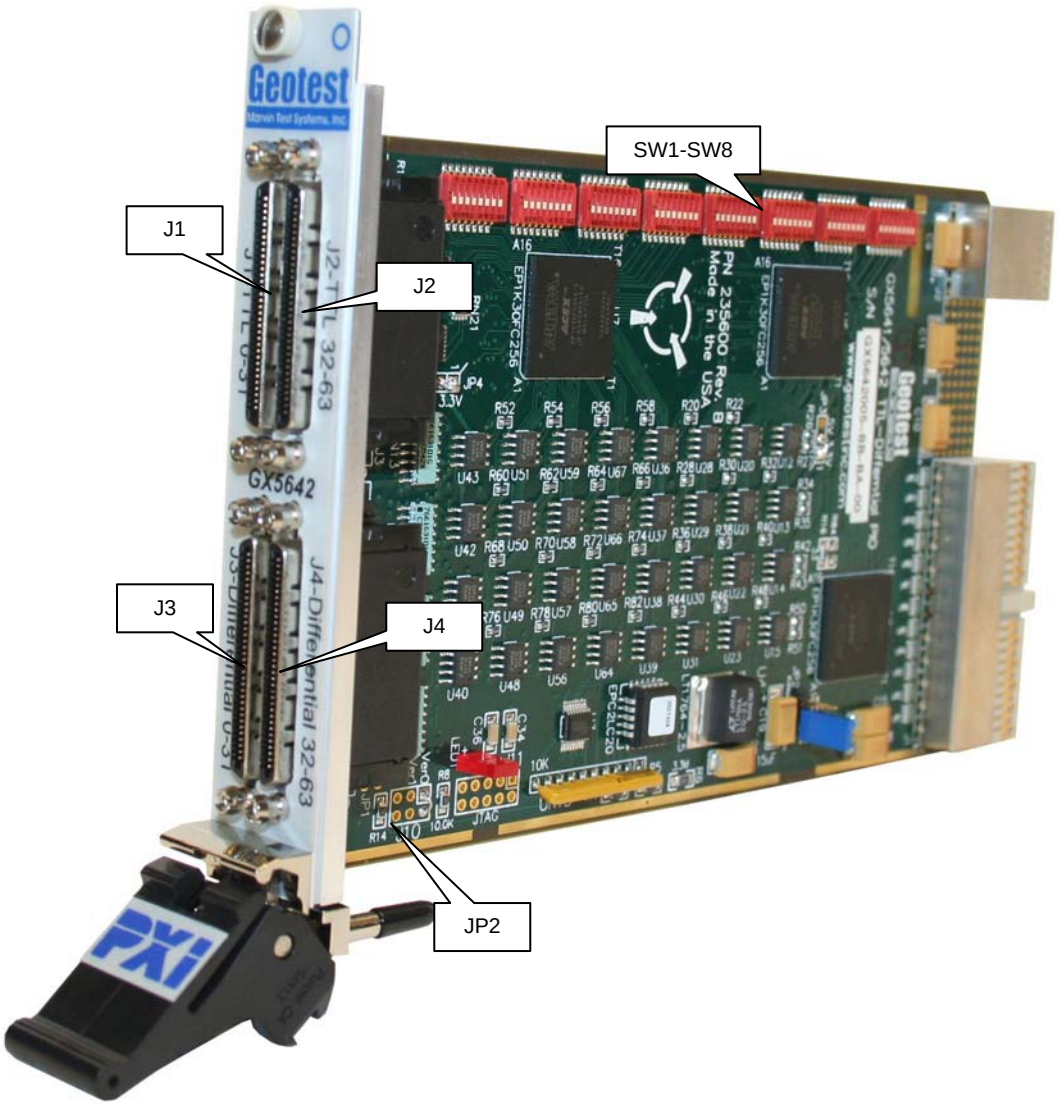

Figure 2-1: GX5642 Board Side View

J1, J2	TTL I/O Connector Channels
J3, J4	Differential I/O Connector Channels
JP2	Jumper installed sets board to Independent mode, without jumper board is in Software Controlled mode.
SW1-SW8	Switches to set conversion direction in independent mode, In software-controlled mode the switches states can be loaded by software command. Switch of SW1 sets Channel 0 direction; Switch 2 of SW1 sets Channel 1 and so on until switch 8 of SW8 who sets channel 63. Switch in ON position set the channel's direction Differential to TTL. Switch in OFF position set the channel's direction Differential to TTL

Architecture: Software Controlled mode (JP2 not installed)

Single I/O channel

The GX5642 provides 128 digital Inputs or Outputs with direction control. Figure 2-2 shows a typical I/O channel block diagram when the board is in Software Controlled mode :

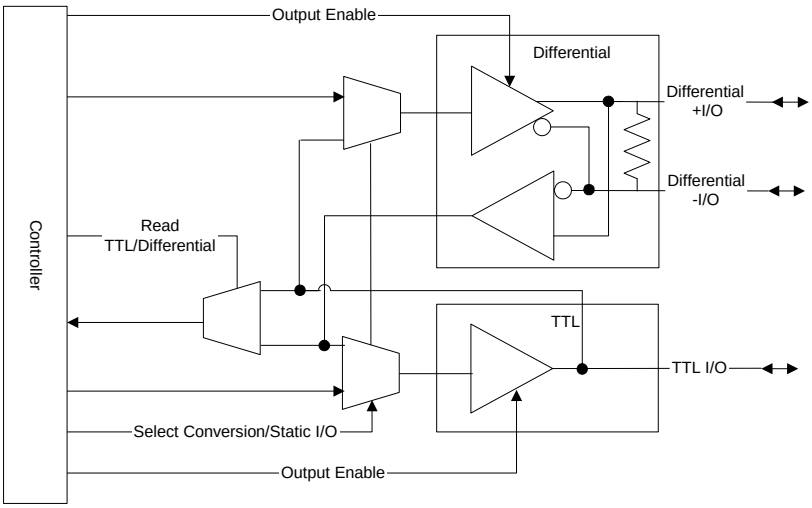

Figure 2-2: Typical I/O Channel

Under software controlled each channel can be programmed to operate in Conversion mode or Static I/O mode. All channels have two ports: TTL and Differential and each port can be programmed to be input or output depending on the channel's programmed operation mode. Each channel's output enable is controlled though software in both Conversion and Static I/O modes. Each channel's port can be read back in both Conversion and Static I/O modes.

When a channel is programmed to Conversion mode the channel's conversion direction can be programmed to be Differential to TTL (Differential port is input and TTL port is output) or TTL to Differential (TTL port is input and Differential port is output).

When a channel is programmed to Static I/O mode each of the channel's ports can be set independently from the other, i.e. the TTL port operates separately form the Differential port. In this mode the channel's TTL port can be set as input TTL or output TTL. And the channel's Differential port can be set as input Differential or output Differential. The Differential I/O terminals are terminated with a built-in 100-Ohm resistor.

Architecture: Independent mode (JP2 installed)

Single I/O channel

The GX5642 provides 128 digital Inputs or Outputs and direction. Figure 2-2 shows a typical I/O channel block diagram when the board is in Independent mode :

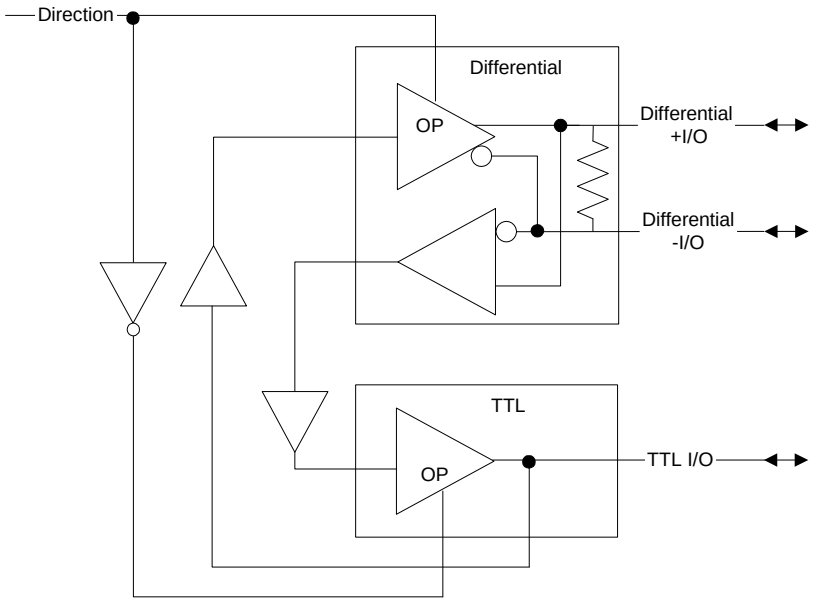

Figure 2-3: Typical I/O Channel

When the board is in Independent mode each channel can convert Differential to TTL (Differential port is input and TTL port is output) or TTL to Differential (TTL port is input and Differential port is output). Each channel has a dedicated DIP-Switch that sets the conversion direction for the channel upon power up. The user can preset each channel conversion direction by simply setting the specified channel's DIP Switch to On position: Differential to TTL or Off position: TTL to Differential. The Differential I/O terminals are terminated with a built-in 100-Ohm resistor.

DIP-Switches Settings

There are eight designated surface mount DIP-Switches, SW1 through SW8, located at the top of the GX5642 board. Each DIP-Switch has eight switches for a total of 64 individual DIP-Switches. When a switch is in the ON position the related channel's direction is Differential to TTL. When a switch is in the OFF position the related channel's direction is TTL to Differential.

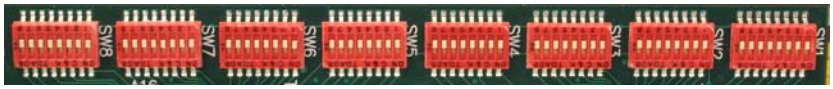

Figure 2-4: DIP-Switches SW1 through SW8

Each DIP-Switch is connected to a channel according to the following tables:

Setting DIP-Switches SW1 and SW2

DIP-Switch SW1				DIP-Switch SW2			
Ch #	**Switch #**	**Position**	**Conversion**	**Ch #**	**Switch #**	**Position**	**Conversion**
0	1	ON	DIFF to TTL	8	1	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF
1	2	ON	DIFF to TTL	9	2	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF
2	3	ON	DIFF to TTL	10	3	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF
3	4	ON	DIFF to TTL	11	4	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF
4	5	ON	DIFF to TTL	12	5	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF
5	6	ON	DIFF to TTL	13	6	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF
6	7	ON	DIFF to TTL	14	7	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF
7	8	ON	DIFF to TTL	15	8	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF

Table 2-1: Setting DIP-Switches SW1 and SW2

Setting DIP-Switches SW3 and SW4

DIP-Switch SW3				DIP-Switch SW4			
Ch #	**Switch #**	**Position**	**Conversion**	**Ch #**	**Switch #**	**Position**	**Conversion**
16	1	ON	DIFF to TTL	24	1	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF
17	2	ON	DIFF to TTL	25	2	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF
18	3	ON	DIFF to TTL	26	3	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF
19	4	ON	DIFF to TTL	27	4	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF
20	5	ON	DIFF to TTL	28	5	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF
21	6	ON	DIFF to TTL	29	6	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF
22	7	ON	DIFF to TTL	30	7	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF
23	8	ON	DIFF to TTL	31	8	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF

Table 2-2: Setting DIP-Switches SW3 and SW4

Setting DIP-Switches SW5 and SW6

DIP-Switch SW5				DIP-Switch SW6			
Ch #	**Switch #**	**Position**	**Conversion**	**Ch #**	**Switch #**	**Position**	**Conversion**
32	1	ON	DIFF to TTL	40	1	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF
33	2	ON	DIFF to TTL	41	2	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF
34	3	ON	DIFF to TTL	42	3	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF
35	4	ON	DIFF to TTL	43	4	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF
36	5	ON	DIFF to TTL	44	5	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF
37	6	ON	DIFF to TTL	45	6	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF
38	7	ON	DIFF to TTL	46	7	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF
38	8	ON	DIFF to TTL	47	8	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF

Table 2-3: Setting DIP-Switches SW5 and SW6

Setting DIP-Switches SW7 and SW8

DIP-Switch SW7				DIP-Switch SW8			
Ch #	**Switch #**	**Position**	**Conversion**	**Ch #**	**Switch #**	**Position**	**Conversion**
48	1	ON	DIFF to TTL	56	1	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF
49	2	ON	DIFF to TTL	57	2	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF
50	3	ON	DIFF to TTL	58	3	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF
51	4	ON	DIFF to TTL	59	4	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF
52	5	ON	DIFF to TTL	60	5	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF
53	6	ON	DIFF to TTL	61	6	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF
54	7	ON	DIFF to TTL	62	7	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF
55	8	ON	DIFF to TTL	63	8	ON	DIFF to TTL
		OFF	TTL to DIFF			OFF	TTL to DIFF

Table 2-4: Setting DIP-Switches SW7 and SW8

Specifications

The following table outlines the specifications of the GX5642.

Channel Specifications

TTL I/O Levels			
	Min. (V)	Typ. (V)	Max. (V)
Input Low	0.0		0.8
Input High	1.8		5.0
Output Low	0.0	0.2	0.5
Output High	2.4	3.0	5.0
LVDS I/O Levels			
	Min. (V)		Max. (V)
Positive-going differential input voltage threshold	-0.2		0.2
Negative-going differential input voltage threshold		0.07	
Driver Differential VOUT	2.0	3.0	5.0
Driver Common-Mode VOUT			3.0

Power Requirements

3.3 VDC Current	0.50A Max
5 VDC Current	1.4A Typical, 2.6A Max
User 5 VDC Current (J3 pin 67 and J4 pin 67)	1A Max

Environmental

Temperature	
Operating	0 to+55°C
Storage	-20 to+70°C

Physical

Size	3U PXI
Weight	18 oz.

Chapter 3 - Installation and Connections

Getting Started

This section includes general hardware installation procedures for the GX5642 board and installation instructions for the GX5642 (GXPIO) software. Before proceeding, please refer to the appropriate chapter to become familiar with the board being installed.

To Find Information on..	Refer to..
Hardware Installation	This Chapter
GX5642 Driver Installation	This Chapter
Programming	Chapter 4
GX5642 Function Reference	Chapter 5

Interfaces and Accessories

The following accessories are available from Geotest for GX5642 switching board.

Part / Model Number	Description
GT95015	Connector Interface SCSI to 100 Mil Grid Differential
GT95021	2’ 68-Pin shielded cable
GT95022	3’ 68-Pin shielded cable
GT95028	10’ 68-Pin shielded cable
GT95031	6’ 68-Pin shielded cable

Packing List

All GX5642 boards have the same basic packing list, which includes:

1. GX5642 Board
2. GX5642 Driver Disk

Unpacking and Inspection

After removing the board from the shipping carton:

 **Caution -** Static sensitive devices are present. Ground yourself to discharge static.

1. Remove the board from the static bag by handling only the metal portions.
2. Be sure to check the contents of the shipping carton to verify that all of the items found in it match the packing list.
3. Inspect the board for possible damage. If there is any sign of damage, return the board immediately. Please refer to the warranty information at the beginning of the manual.

System Requirements

The GX5642 Instrument board is designed to run on PXI compatible computer running Windows 9x, Windows Me, Windows NT, Windows 2000, XP and above. In addition, Microsoft Windows Explorer version 4.0 or above is required to view the online help.

The board requires one unoccupied 3U PXI bus slot.

Board Installation

Before you Begin

Verify that all the components listed in the packing list (see previous paragraph) are present.

Discharge Static Electricity

- To reduce the risk the damaging the board, the following precautions should be observed:
- Leave the board in the anti-static bags until it is installed. The anti-static bag protects the board from harmful static electricity.
- Save the anti-static bag in case the board is removed from the computer in the future.
- Carefully unpack and install the board. Do not drop or handle the board roughly.
- Handle the board by the edges. Do not touch any components on the circuit board.

Caution - Do not insert or remove any board while the computer is on. Turn off the power from the PXI chassis before installation.

Installing a Board

Install the board as follows:

1. Install first the Driver as explain in the next section.
2. Turn off the PXI chassis and unplug the power cord.
3. Set the board switch and jumper settings if required. Check the board documentation for details on jumpers and switch settings before the installation.
4. Locate a PXI/cPCI empty slot on the PXI/cPCI chassis.
5. Place the module edges into the PXI chassis rails (top and bottom).
6. Carefully slide the PXI board to the rear of the chassis, make sure that the ejector handle is pushed **out** (as shown in Figure 3-1).

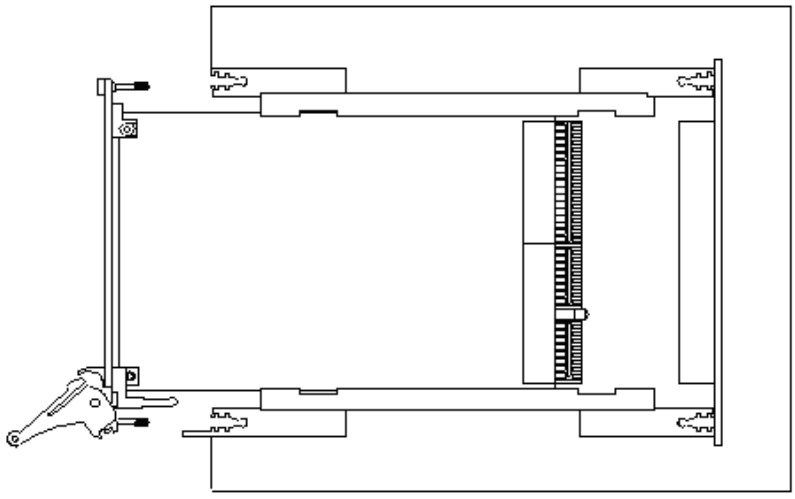

Figure 3-1: Ejector handle position during module insertion

7. After you feel resistance, push in the ejector handles as shown in Figure 3-2 to secure the module into the frame.

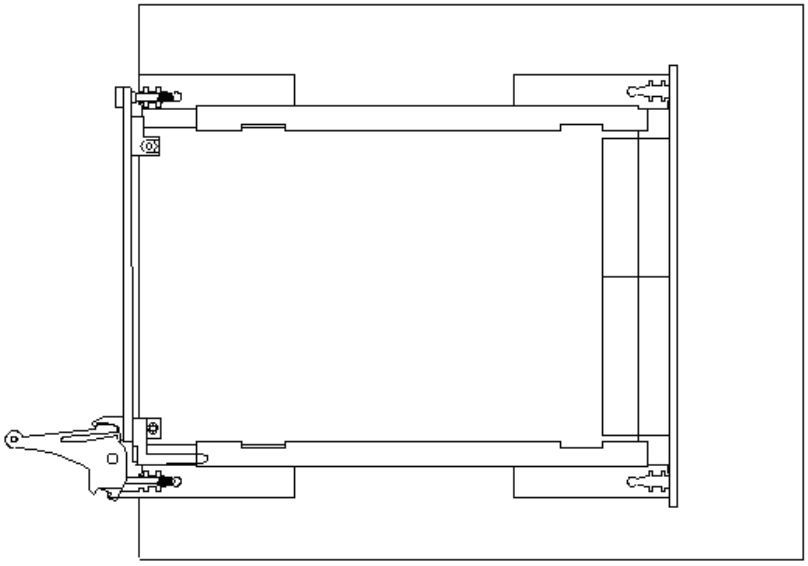

Figure 3-2: Ejector handle position after module insertion

8. Tighten the module's front panel to the chassis to secure the module in.
9. Connect any necessary cables to the board.
10. Plug the power cord in and turn on the PXI or cPCI chassis.

Plug & Play Driver Installation

Plug & Play operating systems such as Windows 9x, Me, Windows 2000 or XP (Not Windows NT) notify the user that a new board was found.

If a Geotest board was already installed, Windows will suggest using the driver information file: HW.INF. The file is located in your Windows INF directory. Click **Next** to confirm and follow the instructions on the screen to complete the driver installation.

If the operating system was unable to find the driver (since the GX5642 driver was not installed prior to the board installation), you may click on the **Have Disk** button and browse to select the HW.INF file. The file should be located in the HW installation directory where the HW.INF is located, c:\program File\Geotest\HW, or the setup disk.

If you are unable to locate the driver (e.g. when using Web installation Package) click **Cancel** to exit the New Hardware Found Wizard, install the GX5642 driver, reboot your computer and repeat this procedure.

Removing a Board

Remove a GX5642 board as follows:

1. Turn off the PXI chassis and unplug the power cord.
2. Disconnect and remove any cables/connectors connected to the board.
3. Un-tighten the module's front panel screws to the chassis.
4. Push out the ejector handle and slide the PXI board away from the chassis.

Installation of the Driver

The Setup installation program requires Windows, 95, 98, Me, NT, 2000 or above. In addition, Microsoft Windows Explorer (IE) version 4.0 or above is required to view the online help.

To run the GXPIO Setup program:

Insert the GXPIO CD-ROM disk in the CD-ROM drive (for floppy disk installation, insert Disk 1 to the floppy drive). The Setup program runs automatically if your drive is set up to auto play.

If Setup does *not* run automatically, select **Run** from the Start menu and when prompted, use the **Browse…** button to locate the **GXPIO.EXE** located on the CD and select **OK**.

Floppy drive Only:

[drive letter]:\setup

Where [*drive letter*] is the drive letter assigned to your Floppy drive (normally A:) as shown here:

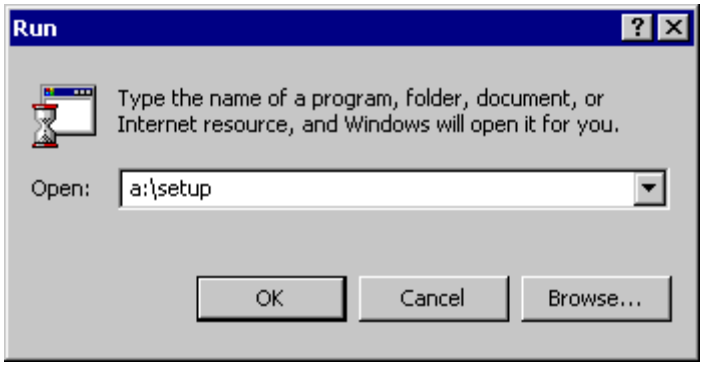

Figure 3-3: Run Window

Click **OK** to continue.

Note: When installing under Windows NT/2000/XP, you may be required to restart the setup after logging-in as a user with an Administrator privileges. This is required in-order to upgrade your system with newer Windows components and to install kernel-mode device drivers (HW.SYS and HWDEVICE.SYS) required by the GX5642 driver to access resources on your board.

The first screen to appear is the Welcome screen. Click **Next** to continue.

Enter your name and company name. Click **Next** to continue.

Enter the folder where GXPIO is to be installed. Either click **Browse** to set up a new folder, or click **Next** to accept the default entry of `C:\Program Files\Geotest\GXPIO`.

Select the type of Setup you wish and click **Next.** You can choose between **Typical**, **Run-Time** and **Custom** setups. **Typical** setup type installs all files. **Run-Time** setup type will install only the files required for controlling the board either from its driver or from its virtual panel. **Custom** setup type lets you select from the available components.

The program will now start its installation. During the installation, Setup may upgrade some of the Windows shared components and files.

The Setup may ask you to reboot after completion if some of the components it replaced where used by another application during the installation – do so before attempting to use the software.

You can now test your installation by starting a panel program that lets you control the board interactively. The panel program can be started by selecting it from the **Start**, **Programs**, **GXPIO** menu located in the Windows Taskbar.

Installation Directories

The GX5642 driver files are installed in the default directory `C:\Program Files\Geotest\GXPIO`. You can change the default `GXPIO` directory to one of your choosing at the time of installation.

During the installation, `GXPIO` Setup creates and copies files to the following directories:

Name	**Purpose / Contents**
…\Geotest\GXPIO	The GX5642 directory. Contains panel programs, programming libraries, interface files and examples, on-line help files and other documentation.
…\Geotest\HW	HW device driver. Provide access to your board hardware resources such as memory, IO ports and PCI board configuration. See the README.TXT located in this directory for more information.
…\ATEasy\Drivers	ATEasy drivers directory. GXPIO Driver and example are copied to this directory only if ***ATEasy*** is installed to your machine.
…\Windows\System (Windows 9x/Me), or …\Windows\System32 when running Windows NT/2000/XP	Windows System directory. Contains the GXPIO DLL driver and some upgraded system components, such as the HTML help viewer, etc.

Setup Maintenance Program

You can run Setup again after GX5642 has been installed from the original disk or from the Windows Control Panel – Add Remove Programs applet. Setup will be in the Maintenance mode when running for the second time. The Maintenance window shown below allows you to modify the current GX5642 installation. The following options are available in Maintenance mode:

Modify. When you want to add or remove GXPIO components.

Repair. When you have corrupted files and need to reinstall.

Remove. When you want to completely remove GXPIO.

The Maintenance mode screen is shown below:

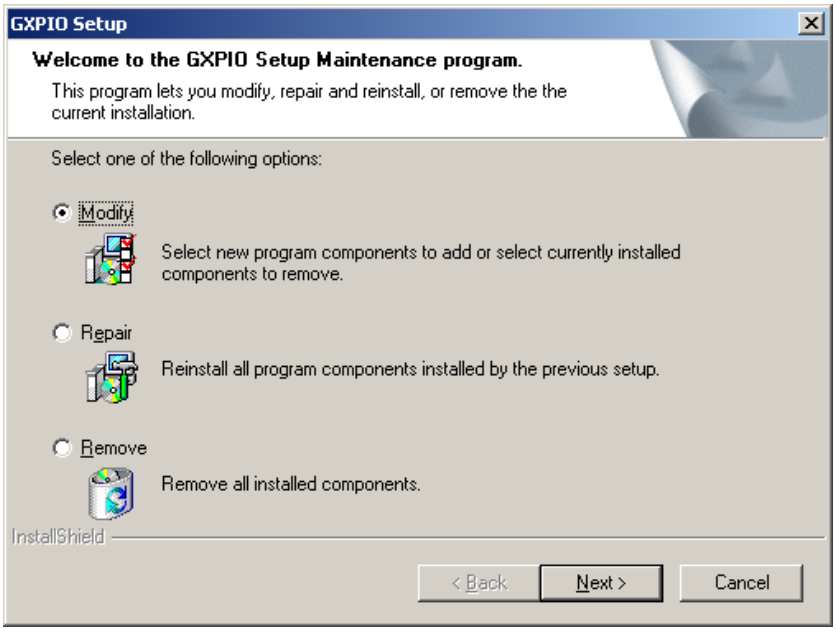

Figure 3-4: Setup Maintenance Window

Select one of the options and click **Next**.

Follow the instruction on the screen until Setup is complete.

Connectors and Jumpers

Figure 3-5 shows the available GX5642 board connectors and jumpers followed by their description:

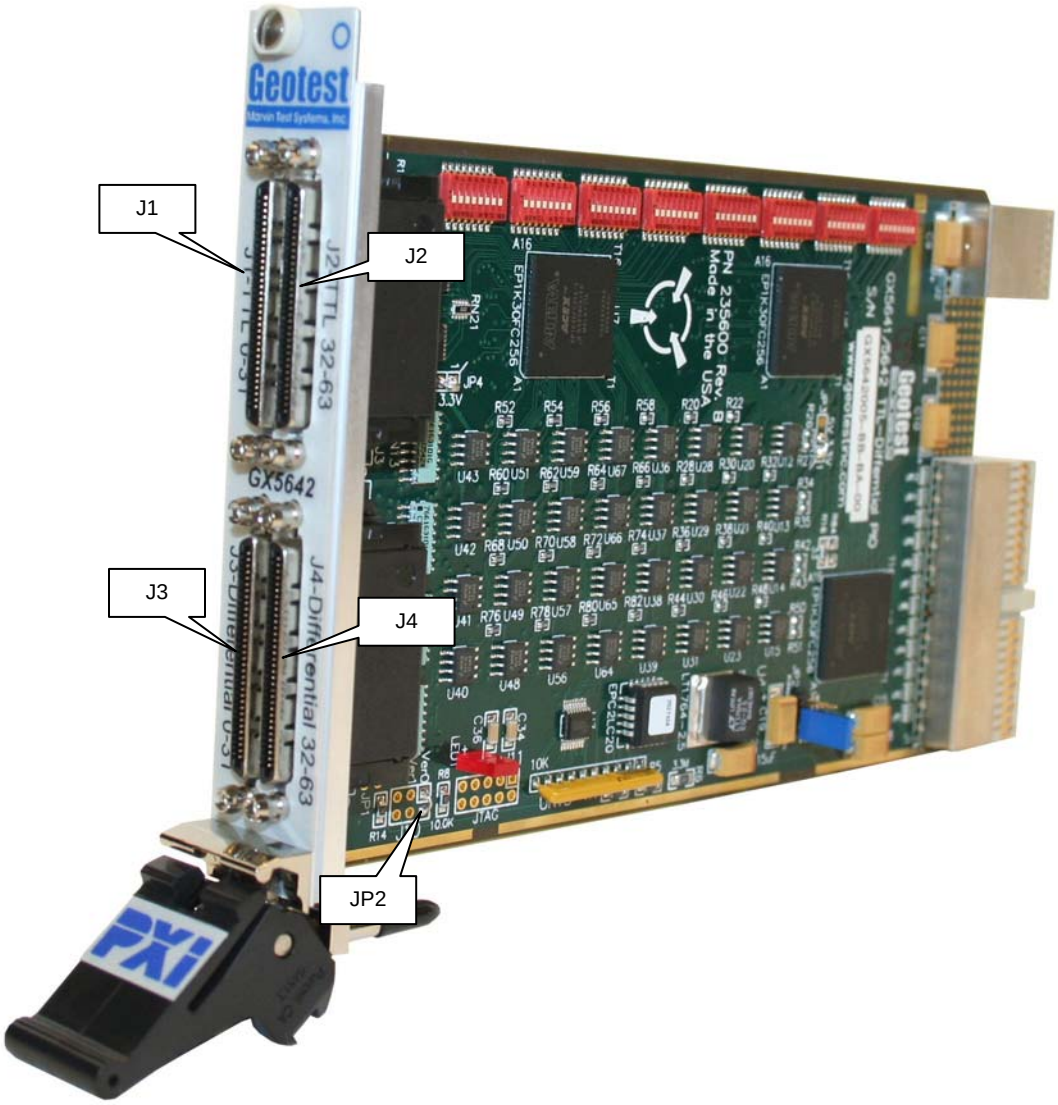

Figure 3-5: GX5642 Connectors and Jumpers

Connector/Jumpers	Description
J1, J2	TTL I/O Connector Channels
J3, J4	Differential I/O Connector Channels
JP2	Installed Jumper sets the board to Independent mode, without jumper the board is in Software Controlled mode.

JP2 – Board Operation Mode Jumper

JP2 jumper determines the Board Operation Mode. When jumper is not installed (default), the board operates in Software Controlled mode. When the jumper is installed, the board operates Independent mode.

Connectors – Default mode (J1 pins 27 and 28 Logic low)

Connections to the GX5642 may be made with 68-pin VHDCI male plug connector. Shielded cables with matching connectors are available from Geotest.

J1 – TTL I/O Connector

The following are connector pin assignments for J1 TTL I/O Connector with low level in Inputs pins 27 and 28:

Pin#	Signal	Type	Pin #	Signal	Type	Pin #	Signal	Type	Pin #	Signal	Type
1	GND	P	18	GND	P	35	TIO0	I/O	52	TIO17	I/O
2	GND	P	19	GND	P	36	TIO1	I/O	53	TIO18	I/O
3	GND	P	20	GND	P	37	TIO2	I/O	54	TIO19	I/O
4	GND	P	21	GND	P	38	TIO3	I/O	55	TIO20	I/O
5	GND	P	22	GND	P	39	TIO4	I/O	56	TIO21	I/O
6	GND	P	23	GND	P	40	TIO5	I/O	57	TIO22	I/O
7	GND	P	24	GND	P	41	TIO6	I/O	58	TIO23	I/O
8	GND	P	25	GND	P	42	TIO7	I/O	59	TIO24	I/O
9	GND	P	26	GND	P	43	TIO8	I/O	60	TIO25	I/O
10	GND	P	27	5V Sense	I	44	TIO9	I/O	61	TIO26	I/O
11	GND	P	28	5V Sense	I	45	TIO10	I/O	62	TIO27	I/O
12	GND	P	29	R	DNU	46	TIO11	I/O	63	TIO28	I/O
13	GND	P	30	R	DNU	47	TIO12	I/O	64	TIO29	I/O
14	GND	P	31	R	DNU	48	TIO13	I/O	65	TIO30	I/O
15	GND	P	32	R	DNU	49	TIO14	I/O	66	TIO31	I/O
16	GND	P	33	R	DNU	50	TIO15	I/O	67	R	DNU
17	GND	P	34	R	DNU	51	TIO16	I/O	68	R	DNU

Table 3-1: J1: TTL I/O Connector with pins 27 and 28 low

I/O: Input/ Output, **R**: Reserved, **DNU**: Do Not Use, **P**: Power/GND
Sense 5V: Input Sense detecting if 5V (Logic high) is connected.

J2 – TTL I/O Connector

The following are connector pin assignments for J2 TTL I/O Connector with low level in J1 inputs pins 27 and 28:

Pin#	Signal	Type	Pin #	Signal	Type	Pin #	Signal	Type	Pin #	Signal	Type
1	GND	P	18	GND	P	35	TIO32	I/O	52	TIO49	I/O
2	GND	P	19	GND	P	36	TIO33	I/O	53	TIO50	I/O
3	GND	P	20	GND	P	37	TIO34	I/O	54	TIO51	I/O
4	GND	P	21	GND	P	38	TIO35	I/O	55	TIO52	I/O
5	GND	P	22	GND	P	39	TIO36	I/O	56	TIO53	I/O
6	GND	P	23	GND	P	40	TIO37	I/O	57	TIO54	I/O
7	GND	P	24	GND	P	41	TIO38	I/O	58	TIO55	I/O
8	GND	P	25	GND	P	42	TIO39	I/O	59	TIO56	I/O
9	GND	P	26	GND	P	43	TIO40	I/O	60	TIO57	I/O
10	GND	P	27	R	DNU	44	TIO41	I/O	61	TIO58	I/O
11	GND	P	28	R	DNU	45	TIO42	I/O	62	TIO59	I/O
12	GND	P	29	R	DNU	46	TIO43	I/O	63	TIO60	I/O
13	GND	P	30	R	DNU	47	TIO44	I/O	64	TIO61	I/O
14	GND	P	31	R	DNU	48	TIO45	I/O	65	TIO62	I/O
15	GND	P	32	R	DNU	49	TIO46	I/O	66	TIO63	I/O
16	GND	P	33	R	DNU	50	TIO47	I/O	67	R	DNU
17	GND	P	34	R	DNU	51	TIO48	I/O	68	R	DNU

Table 3-2: J2: TTL I/O Connector with J1 pins 27 and 28 low

I/O: Input/ Output, **R**: Reserved, **DNU**: Do Not Use, **P**: Power/GND

J3 – Differential I/O Connector Channels 0-31

The following are connector pin assignments for J3 Differential I/O Channels 0-31 Connector:

#	Signal	Type	#	Signal	Type	#	Signal	Type	#	Signal	Type
1	DIO+0	I/O	18	DIO+17	I/O	35	DIO-0	I/O	52	DIO-17	I/O
2	DIO+1	I/O	19	DIO+18	I/O	36	DIO-1	I/O	53	DIO-18	I/O
3	DIO+2	I/O	20	DIO+19	I/O	37	DIO-2	I/O	54	DIO-19	I/O
4	DIO+3	I/O	21	DIO+20	I/O	38	DIO-3	I/O	55	DIO-20	I/O
5	DIO+4	I/O	22	DIO+21	I/O	39	DIO-4	I/O	56	DIO-21	I/O
6	DIO+5	I/O	23	DIO+22	I/O	40	DIO-5	I/O	57	DIO-22	I/O
7	DIO+6	I/O	24	DIO+23	I/O	41	DIO-6	I/O	58	DIO-23	I/O
8	DIO+7	I/O	25	DIO+24	I/O	42	DIO-7	I/O	59	DIO-24	I/O
9	DIO+8	I/O	26	DIO+25	I/O	43	DIO-8	I/O	60	DIO-25	I/O
10	DIO+9	I/O	27	DIO+26	I/O	44	DIO-9	I/O	61	DIO-26	I/O
11	DIO+10	I/O	28	DIO+27	I/O	45	DIO-10	I/O	62	DIO-27	I/O
12	DIO+11	I/O	29	DIO+28	I/O	46	DIO-11	I/O	63	DIO-28	I/O
13	DIO+12	I/O	30	DIO+29	I/O	47	DIO-12	I/O	64	DIO-29	I/O
14	DIO+13	I/O	31	DIO+30	I/O	48	DIO-13	I/O	65	DIO-30	I/O
15	DIO+14	I/O	32	DIO+31	I/O	49	DIO-14	I/O	66	DIO-31	I/O
16	DIO+15	I/O	33	5V	P	50	DIO-15	I/O	67	5V	P
17	DIO+16	I/O	34	GND	P	51	DIO-16	I/O	68	GND	P

Table 3-3: J3: Differential I/O Connector Channels 0-31

I/O: Input/ Output, **R**: Reserved, **DNU**: Do Not Use, **P**: Power/GND

J4 – Differential I/O Connector Channels 32-63

The following are connector pin assignments for J4 Differential I/O Channels 32-63 Connector:

#	Signal	Type	#	Signal	Type	#	Signal	Type	#	Signal	Type
1	DIO+32	I/O	18	DIO+49	I/O	35	DIO-32	I/O	52	DIO-49	I/O
2	DIO+33	I/O	19	DIO+50	I/O	36	DIO-33	I/O	53	DIO-50	I/O
3	DIO+34	I/O	20	DIO+51	I/O	37	DIO-34	I/O	54	DIO-51	I/O
4	DIO+35	I/O	21	DIO+52	I/O	38	DIO-35	I/O	55	DIO-52	I/O
5	DIO+36	I/O	22	DIO+53	I/O	39	DIO-36	I/O	56	DIO-53	I/O
6	DIO+37	I/O	23	DIO+54	I/O	40	DIO-37	I/O	57	DIO-54	I/O
7	DIO+38	I/O	24	DIO+55	I/O	41	DIO-38	I/O	58	DIO-55	I/O
8	DIO+39	I/O	25	DIO+56	I/O	42	DIO-39	I/O	59	DIO-56	I/O
9	DIO+40	I/O	26	DIO+57	I/O	43	DIO-40	I/O	60	DIO-57	I/O
10	DIO+41	I/O	27	DIO+58	I/O	44	DIO-41	I/O	61	DIO-58	I/O
11	DIO+42	I/O	28	DIO+59	I/O	45	DIO-42	I/O	62	DIO-59	I/O
12	DIO+43	I/O	29	DIO+60	I/O	46	DIO-43	I/O	63	DIO-60	I/O
13	DIO+44	I/O	30	DIO+61	I/O	47	DIO-44	I/O	64	DIO-61	I/O
14	DIO+45	I/O	31	DIO+62	I/O	48	DIO-45	I/O	65	DIO-62	I/O
15	DIO+46	I/O	32	DIO+63	I/O	49	DIO-46	I/O	66	DIO-63	I/O
16	DIO+47	I/O	33	5V	P	50	DIO-47	I/O	67	5V	P
17	DIO+48	I/O	34	GND	P	51	DIO-48	I/O	68	GND	P

Table 3-4: J4: Differential I/O Connector Channels 32-63

I/O: Input/ Output, **R**: Reserved, **DNU**: Do Not Use, **P**: Power/GND

Connectors – National Instruments Compatibility mode (J1 pins 27 and 28 Logic High)

Whenever J1 pins 27 and 28 are set to Logic High (5V Sense) J1 and J2 connectors signal assignments change. In this mode J1 connector will have an additional eight TTL I/O channels for a total of 40 channels. The additional eight channels are channels 32-39 of J2 TTL I/O connector. Those channels should not be used on J2.

J1 – TTL I/O Connector

The following are connector pin assignments for J1 TTL I/O Connector with high level in inputs pins 27 and 28:

Pin#	Signal	Type	Pin #	Signal	Type	Pin #	Signal	Type	Pin #	Signal	Type
1	GND	P	18	GND	P	35	TIO0	I/O	52	TIO17	I/O
2	GND	P	19	GND	P	36	TIO1	I/O	53	TIO18	I/O
3	GND	P	20	GND	P	37	TIO2	I/O	54	TIO19	I/O
4	GND	P	21	GND	P	38	TIO3	I/O	55	TIO20	I/O
5	GND	P	22	GND	P	39	TIO4	I/O	56	TIO21	I/O
6	GND	P	23	GND	P	40	TIO5	I/O	57	TIO22	I/O
7	GND	P	24	GND	P	41	TIO6	I/O	58	TIO23	I/O
8	GND	P	25	GND	P	42	TIO7	I/O	59	TIO24	I/O
9	GND	P	26	GND	P	43	TIO8	I/O	60	TIO25	I/O
10	GND	P	27	5V Sense	I	44	TIO9	I/O	61	TIO26	I/O
11	GND	P	28	5V Sense	I	45	TIO10	I/O	62	TIO27	I/O
12	GND	P	29	TIO28	I/O	46	TIO11	I/O	63	TIO29	I/O
13	GND	P	30	TIO30	I/O	47	TIO12	I/O	64	TIO31	I/O
14	GND	P	31	TIO32	I/O	48	TIO13	I/O	65	TIO33	I/O
15	GND	P	32	TIO34	I/O	49	TIO14	I/O	66	TIO35	I/O
16	GND	P	33	TIO36	I/O	50	TIO15	I/O	67	TIO37	I/O
17	GND	P	34	TIO38	I/O	51	TIO16	I/O	68	TIO39	I/O

Table 3-5: J1: TTL I/O Connector with pins 27 and 28 high

I/O: Input/ Output, **P**: Power/GND
Sense 5V: Input Sense detecting if 5V (Logic high) is connected.

J2 – TTL I/O Connector

The following are connector pin assignments for J2 TTL I/O Connector with high level in J1 inputs pins 27 and 28:

Pin#	Signal	Type	Pin #	Signal	Type	Pin #	Signal	Type	Pin #	Signal	Type
1	GND	P	18	GND	P	35	R	DNU	52	TIO49	I/O
2	GND	P	19	GND	P	36	R	DNU	53	TIO50	I/O
3	GND	P	20	GND	P	37	R	DNU	54	TIO51	I/O
4	GND	P	21	GND	P	38	R	DNU	55	TIO52	I/O
5	GND	P	22	GND	P	39	R	DNU	56	TIO53	I/O
6	GND	P	23	GND	P	40	R	DNU	57	TIO54	I/O
7	GND	P	24	GND	P	41	R	DNU	58	TIO55	I/O
8	GND	P	25	GND	P	42	R	DNU	59	TIO56	I/O
9	GND	P	26	GND	P	43	TIO40	I/O	60	TIO57	I/O
10	GND	P	27	R	DNU	44	TIO41	I/O	61	TIO58	I/O
11	GND	P	28	R	DNU	45	TIO42	I/O	62	TIO59	I/O
12	GND	P	29	R	DNU	46	TIO43	I/O	63	TIO60	I/O
13	GND	P	30	R	DNU	47	TIO44	I/O	64	TIO61	I/O
14	GND	P	31	R	DNU	48	TIO45	I/O	65	TIO62	I/O
15	GND	P	32	R	DNU	49	TIO46	I/O	66	TIO63	I/O
16	GND	P	33	R	DNU	50	TIO47	I/O	67	R	DNU
17	GND	P	34	R	DNU	51	TIO48	I/O	68	R	DNU

Table 3-6: J2: TTL I/O Connector with J1 pins 27 and 28 high

I/O: Input/ Output, **R**: Reserved, **DNU**: Do Not Use, **P**: Power/GND

J3 – Differential I/O Connector Channels 0-31

The following are connector pin assignments for J3 Differential I/O Channels 0-31 Connector with high level in J1 inputs pins 27 and 28:

#	Signal	Type	#	Signal	Type	#	Signal	Type	#	Signal	Type
1	DIO+0	I/O	18	DIO+17	I/O	35	DIO-0	I/O	52	DIO-17	I/O
2	DIO+1	I/O	19	DIO+18	I/O	36	DIO-1	I/O	53	DIO-18	I/O
3	DIO+2	I/O	20	DIO+19	I/O	37	DIO-2	I/O	54	DIO-19	I/O
4	DIO+3	I/O	21	DIO+20	I/O	38	DIO-3	I/O	55	DIO-20	I/O
5	DIO+4	I/O	22	DIO+21	I/O	39	DIO-4	I/O	56	DIO-21	I/O
6	DIO+5	I/O	23	DIO+22	I/O	40	DIO-5	I/O	57	DIO-22	I/O
7	DIO+6	I/O	24	DIO+23	I/O	41	DIO-6	I/O	58	DIO-23	I/O
8	DIO+7	I/O	25	DIO+24	I/O	42	DIO-7	I/O	59	DIO-24	I/O
9	DIO+8	I/O	26	DIO+25	I/O	43	DIO-8	I/O	60	DIO-25	I/O
10	DIO+9	I/O	27	DIO+26	I/O	44	DIO-9	I/O	61	DIO-26	I/O
11	DIO+10	I/O	28	DIO+27	I/O	45	DIO-10	I/O	62	DIO-27	I/O
12	DIO+11	I/O	29	DIO+28	I/O	46	DIO-11	I/O	63	DIO-28	I/O
13	DIO+12	I/O	30	DIO+29	I/O	47	DIO-12	I/O	64	DIO-29	I/O
14	DIO+13	I/O	31	DIO+30	I/O	48	DIO-13	I/O	65	DIO-30	I/O
15	DIO+14	I/O	32	DIO+31	I/O	49	DIO-14	I/O	66	DIO-31	I/O
16	DIO+15	I/O	33	5V	P	50	DIO-15	I/O	67	5V	P
17	DIO+16	I/O	34	GND	P	51	DIO-16	I/O	68	GND	P

Table 3-7: J3: Differential I/O Connector Channels 0-31

I/O: Input/ Output, **R**: Reserved, **DNU**: Do Not Use, **P**: Power/GND

J4 – Differential I/O Connector Channels 32-63

The following are connector pin assignments for J4 Differential I/O Channels 32-63 Connector with high level in J1 inputs pins 27 and 28:

#	Signal	Type	#	Signal	Type	#	Signal	Type	#	Signal	Type
1	DIO+32	I/O	18	DIO+49	I/O	35	DIO-32	I/O	52	DIO-49	I/O
2	DIO+33	I/O	19	DIO+50	I/O	36	DIO-33	I/O	53	DIO-50	I/O
3	DIO+34	I/O	20	DIO+51	I/O	37	DIO-34	I/O	54	DIO-51	I/O
4	DIO+35	I/O	21	DIO+52	I/O	38	DIO-35	I/O	55	DIO-52	I/O
5	DIO+36	I/O	22	DIO+53	I/O	39	DIO-36	I/O	56	DIO-53	I/O
6	DIO+37	I/O	23	DIO+54	I/O	40	DIO-37	I/O	57	DIO-54	I/O
7	DIO+38	I/O	24	DIO+55	I/O	41	DIO-38	I/O	58	DIO-55	I/O
8	DIO+39	I/O	25	DIO+56	I/O	42	DIO-39	I/O	59	DIO-56	I/O
9	DIO+40	I/O	26	DIO+57	I/O	43	DIO-40	I/O	60	DIO-57	I/O
10	DIO+41	I/O	27	DIO+58	I/O	44	DIO-41	I/O	61	DIO-58	I/O
11	DIO+42	I/O	28	DIO+59	I/O	45	DIO-42	I/O	62	DIO-59	I/O
12	DIO+43	I/O	29	DIO+60	I/O	46	DIO-43	I/O	63	DIO-60	I/O
13	DIO+44	I/O	30	DIO+61	I/O	47	DIO-44	I/O	64	DIO-61	I/O
14	DIO+45	I/O	31	DIO+62	I/O	48	DIO-45	I/O	65	DIO-62	I/O
15	DIO+46	I/O	32	DIO+63	I/O	49	DIO-46	I/O	66	DIO-63	I/O
16	DIO+47	I/O	33	5V	P	50	DIO-47	I/O	67	5V	P
17	DIO+48	I/O	34	GND	P	51	DIO-48	I/O	68	GND	P

Table 3-8: J4: Differential I/O Connector Channels 32-63

I/O: Input/ Output, **R**: Reserved, **DNU**: Do Not Use, **P**: Power/GND

Connectors – Independent mode (JP2 installed)

Whenever JP2 jumper is installed the board is working in independent mode. When the card is configured to Independent mode the PXI interface is disabled and all 64 channels operate in Conversion mode. In this mode each channel conversion operation is predefined using DIP-Switches. Each channel can be predefined to convert TTL to Differential or Differential to TTL. The channels conversion settings will be loaded automatically upon power up.

NOTE: In this mode J1 pins 27 and 28 are set to output high.

J1 – TTL I/O Connector

The following are connector pin assignments for J1 TTL I/O Connector with JP2 installed:

Pin#	Signal	Type	Pin #	Signal	Type	Pin #	Signal	Type	Pin #	Signal	Type
1	GND	P	18	GND	P	35	TIO0	I/O	52	TIO17	I/O
2	GND	P	19	GND	P	36	TIO1	I/O	53	TIO18	I/O
3	GND	P	20	GND	P	37	TIO2	I/O	54	TIO19	I/O
4	GND	P	21	GND	P	38	TIO3	I/O	55	TIO20	I/O
5	GND	P	22	GND	P	39	TIO4	I/O	56	TIO21	I/O
6	GND	P	23	GND	P	40	TIO5	I/O	57	TIO22	I/O
7	GND	P	24	GND	P	41	TIO6	I/O	58	TIO23	I/O
8	GND	P	25	GND	P	42	TIO7	I/O	59	TIO24	I/O
9	GND	P	26	GND	P	43	TIO8	I/O	60	TIO25	I/O
10	GND	P	27	High	O	44	TIO9	I/O	61	TIO26	I/O
11	GND	P	28	High	O	45	TIO10	I/O	62	TIO27	I/O
12	GND	P	29	TIO28	I/O	46	TIO11	I/O	63	TIO29	I/O
13	GND	P	30	TIO30	I/O	47	TIO12	I/O	64	TIO31	I/O
14	GND	P	31	TIO32	I/O	48	TIO13	I/O	65	TIO33	I/O
15	GND	P	32	TIO34	I/O	49	TIO14	I/O	66	TIO35	I/O
16	GND	P	33	TIO36	I/O	50	TIO15	I/O	67	TIO37	DNU
17	GND	P	34	TIO38	I/O	51	TIO16	I/O	68	TIO39	DNU

Table 3-9: J1: TTL I/O Connector with JP2 installed

I/O: Input/ Output, **P**: Power/GND

J2 – TTL I/O Connector

The following are connector pin assignments for J2 TTL I/O Connector with JP2 installed:

Pin#	Signal	Type	Pin #	Signal	Type	Pin #	Signal	Type	Pin #	Signal	Type
1	GND	P	18	GND	P	35	R	DNU	52	TIO49	I/O
2	GND	P	19	GND	P	36	R	DNU	53	TIO50	I/O
3	GND	P	20	GND	P	37	R	DNU	54	TIO51	I/O
4	GND	P	21	GND	P	38	R	DNU	55	TIO52	I/O
5	GND	P	22	GND	P	39	R	DNU	56	TIO53	I/O
6	GND	P	23	GND	P	40	R	DNU	57	TIO54	I/O
7	GND	P	24	GND	P	41	R	DNU	58	TIO55	I/O
8	GND	P	25	GND	P	42	R	DNU	59	TIO56	I/O
9	GND	P	26	GND	P	43	TIO40	I/O	60	TIO57	I/O
10	GND	P	27	R	DNU	44	TIO41	I/O	61	TIO58	I/O
11	GND	P	28	R	DNU	45	TIO42	I/O	62	TIO59	I/O
12	GND	P	29	R	DNU	46	TIO43	I/O	63	TIO60	I/O
13	GND	P	30	R	DNU	47	TIO44	I/O	64	TIO61	I/O
14	GND	P	31	R	DNU	48	TIO45	I/O	65	TIO62	I/O
15	GND	P	32	R	DNU	49	TIO46	I/O	66	TIO63	I/O
16	GND	P	33	R	DNU	50	TIO47	I/O	67	R	DNU
17	GND	P	34	R	DNU	51	TIO48	I/O	68	R	DNU

Table 3-10: J2: TTL I/O Connector connected with JP2 installed

I/O: Input/ Output, **R**: Reserved, **DNU**: Do Not Use, **P**: Power/GND

J3 – Differential I/O Connector Channels 0-31

The following are connector pin assignments for J3 Differential I/O Channels 0-31 Connector with JP2 installed::

#	Signal	Type	#	Signal	Type	#	Signal	Type	#	Signal	Type
1	DIO+0	I/O	18	DIO+17	I/O	35	DIO-0	I/O	52	DIO-17	I/O
2	DIO+1	I/O	19	DIO+18	I/O	36	DIO-1	I/O	53	DIO-18	I/O
3	DIO+2	I/O	20	DIO+19	I/O	37	DIO-2	I/O	54	DIO-19	I/O
4	DIO+3	I/O	21	DIO+20	I/O	38	DIO-3	I/O	55	DIO-20	I/O
5	DIO+4	I/O	22	DIO+21	I/O	39	DIO-4	I/O	56	DIO-21	I/O
6	DIO+5	I/O	23	DIO+22	I/O	40	DIO-5	I/O	57	DIO-22	I/O
7	DIO+6	I/O	24	DIO+23	I/O	41	DIO-6	I/O	58	DIO-23	I/O
8	DIO+7	I/O	25	DIO+24	I/O	42	DIO-7	I/O	59	DIO-24	I/O
9	DIO+8	I/O	26	DIO+25	I/O	43	DIO-8	I/O	60	DIO-25	I/O
10	DIO+9	I/O	27	DIO+26	I/O	44	DIO-9	I/O	61	DIO-26	I/O
11	DIO+10	I/O	28	DIO+27	I/O	45	DIO-10	I/O	62	DIO-27	I/O
12	DIO+11	I/O	29	DIO+28	I/O	46	DIO-11	I/O	63	DIO-28	I/O
13	DIO+12	I/O	30	DIO+29	I/O	47	DIO-12	I/O	64	DIO-29	I/O
14	DIO+13	I/O	31	DIO+30	I/O	48	DIO-13	I/O	65	DIO-30	I/O
15	DIO+14	I/O	32	DIO+31	I/O	49	DIO-14	I/O	66	DIO-31	I/O
16	DIO+15	I/O	33	5V	P	50	DIO-15	I/O	67	5V	P
17	DIO+16	I/O	34	GND	P	51	DIO-16	I/O	68	GND	P

Table 3-11: J3: Differential I/O Connector Channels 0-31

I/O: Input/ Output, **R**: Reserved, **DNU**: Do Not Use, **P**: Power/GND

J4 – Differential I/O Connector Channels 32-63

The following are connector pin assignments for J4 Differential I/O Channels 32-63 Connector with JP2 installed:

#	Signal	Type	#	Signal	Type	#	Signal	Type	#	Signal	Type
1	DIO+32	I/O	18	DIO+49	I/O	35	DIO-32	I/O	52	DIO-49	I/O
2	DIO+33	I/O	19	DIO+50	I/O	36	DIO-33	I/O	53	DIO-50	I/O
3	DIO+34	I/O	20	DIO+51	I/O	37	DIO-34	I/O	54	DIO-51	I/O
4	DIO+35	I/O	21	DIO+52	I/O	38	DIO-35	I/O	55	DIO-52	I/O
5	DIO+36	I/O	22	DIO+53	I/O	39	DIO-36	I/O	56	DIO-53	I/O
6	DIO+37	I/O	23	DIO+54	I/O	40	DIO-37	I/O	57	DIO-54	I/O
7	DIO+38	I/O	24	DIO+55	I/O	41	DIO-38	I/O	58	DIO-55	I/O
8	DIO+39	I/O	25	DIO+56	I/O	42	DIO-39	I/O	59	DIO-56	I/O
9	DIO+40	I/O	26	DIO+57	I/O	43	DIO-40	I/O	60	DIO-57	I/O
10	DIO+41	I/O	27	DIO+58	I/O	44	DIO-41	I/O	61	DIO-58	I/O
11	DIO+42	I/O	28	DIO+59	I/O	45	DIO-42	I/O	62	DIO-59	I/O
12	DIO+43	I/O	29	DIO+60	I/O	46	DIO-43	I/O	63	DIO-60	I/O
13	DIO+44	I/O	30	DIO+61	I/O	47	DIO-44	I/O	64	DIO-61	I/O
14	DIO+45	I/O	31	DIO+62	I/O	48	DIO-45	I/O	65	DIO-62	I/O
15	DIO+46	I/O	32	DIO+63	I/O	49	DIO-46	I/O	66	DIO-63	I/O
16	DIO+47	I/O	33	5V	P	50	DIO-47	I/O	67	5V	P
17	DIO+48	I/O	34	GND	P	51	DIO-48	I/O	68	GND	P

Table 3-12: J4: Differential I/O Connector Channels 32-63

I/O: Input/ Output, **R**: Reserved, **DNU**: Do Not Use, **P**: Power/GND

Chapter 4 - Programming the Board

Introduction

This chapter contains information about how to program the GX5642 board using the GXPIO driver.

The GXPIO driver contains functions to initialize, reset, and control the power-supply instruments. A brief description of the functions, as well as how and when to use them, is included in this chapter.

The GXPIO driver supports many development tools. Using these tools with the driver is described in this chapter. In addition, the GXPIO directory contains examples written for these development tools. Refer to Chapter 4 for a list of the available examples.

The GXPIO Driver

The GXPIO driver is a 32-bit Windows DLL file: GxPio.dll. The DLL is used with 32 bit applications running under Windows 95/98/ME and Windows NT/2000. The DLL uses a device driver to access the board resources. The device driver HW.SYS (on Windows NT/2000) or HW.VXD (on Windows 9x/Me) is installed by the GXPIO setup program and is shared by other Geotest products (ATEasy, GTDIO, etc).

The 32-bit DLL can be used with various development tools such as Microsoft Visual C++, Borland C++ Builder, Microsoft Visual Basic, Borland Pascal or Delphi, ATEasy and more. The following paragraphs describe how to create an application that uses the driver with various development tools. Refer to the paragraph describing the specific development tool for more information.

Programming Using C/C++ Tools

The following steps are required to use the GXPIO driver with C/C++ development tools:

Include the GxPio.h header file in the C/C++ source file that uses the GXPIO function. This header file is used for all driver types. The file contains function prototypes and constant declarations to be used by the compiler for the application.

Add the required .LIB file to the projects. This can be import library GxPio.lib for Microsoft Visual C++ and GxPioBC.lib for Borland C++. Windows based applications that explicitly load the DLL by calling the Windows LoadLibrary() API should not include the .LIB file in the project.

Add code to call the GXPIO as required by the application.

Build the project.

Run, test, and debug the application.

Programming Using Visual Basic

To use the driver with Visual Basic 4.0 or above (for 32-bit applications), the user must include the GxPio.bas to the project. The file can be loaded using *Add File* from the Visual Basic *File menu*. The GxPio.bas contains function declarations for the DLL driver.

Programming Using Pascal/Delphi

To use the driver with Borland Pascal or Delphi, the user must include the GxPio.pas to the project. The GxPio.pas file contains a **unit** with function prototypes for the DLL functions. Include the GXPIO unit in the **uses** statement before making calls to the GXPIO functions.

Programming GXPIO Boards Using ATEasy®

The GXPIO package is supplied with a separate ATEasy driver for each board types. For example, the GX5642 is supplied with GX5642.DRV ATEasy driver. The ATEasy driver uses the GxPio.dll to program the board. In addition, each driver is supplied with an example that contains a program and a system file pre-configured with the ATEasy driver. Use the driver shortcut property page from the System Drivers sub-module to change the PXI slot number before attempting to run the example.

Using commands declared in the ATEasy driver are easier to use than using the DLL functions directly. The driver commands will also generate exception that allow the ATEasy application to trap errors without checking the status code returned by the DLL function after each function call.

The ATEasy driver contains commands that are similar to the DLL functions in name and parameters, with the following exceptions:

The *nHandle* parameter is omitted. The driver handles this parameter automatically. ATEasy uses driver logical names instead i.e. GXPIO for GX5642.

The *nStatus* parameter was omitted. Use the Get Status commands instead of checking the status. After calling a DLL function the ATEasy driver will check the returned status and will call the error statement (in case of an error status) to generate exception that can be easily trapped by the application using the **OnError** module event or using the **try-catch** statement.

Some ATEasy drivers contain additional commands to permit easier access to the board features. For example parameters for a function may be omitted by using a command item instead of typing the parameter value. The commands are self-documented. Their syntax is similar to English. In addition, you may generate the commands from the code editor context menu or by using the ATEasy's code completion feature instead of typing them directly.

Using the GXPIO driver functions

The GXPIO driver contains a set of functions for each of the supported instrument boards. The function name also starts with the board type. For example the G5642 board uses the GxPioxxx functions. Other functions are available as general purpose functions that apply to all boards (i.e GxPioGetDriverSummary).

Each of the board types has similar functions that initialize the board driver, reset the board, and display the instrument virtual panel. In addition, all the board types use handles (see below) to access the boards and use the same error handling method. The following paragraphs describe the steps required to program the boards.

Initialization and PCI Slot Numbers

The **Gx5642Initialize**(*nSlot, pnHandle, pnStatus*) function initializes the driver for the board at the specified PCI (or PXI) slot number and returns a board handle. The PCI slot number where the board is installed can be viewed from the **PXI/PCI Explorer** applet in the Windows Control Panel as shown below:

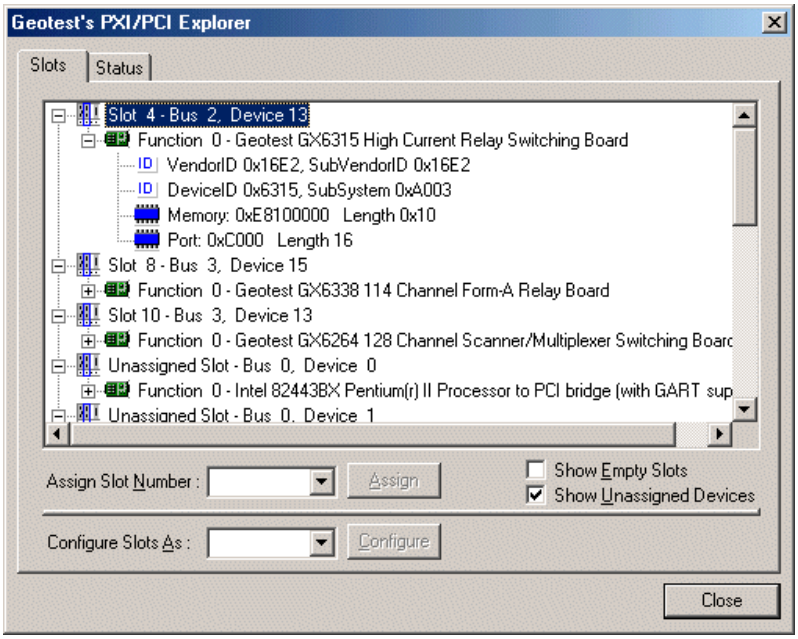

Figure 4-1: PXI/PCI Explorer

Slot numbers are gathered by PXI/PCI Explorer using a PXISYS.INI file located in the Windows directory. Each PCI slot is identified by a unique combination of PCI Bus and Device numbers. If that file does not exist or not complete with all bus/devices assigned slot numbers, you can assign slot numbers to your PCI slots. Assigning a number to a slot can be done as follow:

1. Select the bus/device item from the PCI Explorer tree
2. Select a slot number from the **Assign Slot Number** drop down list. If the chassis has slot number labels below the slot, use the number, if not, use any of the slot numbers shown in the drop down list.

A pre-configured PXISYS.INI file for Geotest GX7000 computer is available in the HW directory. This file is copied to the Windows directory if you have a GX7000 computer. You may use the **Configure** button to initialize the PXISYS.INI to a specific system slots configuration.

The **Gx5642Initialize** function returns a handle that must be used with other driver functions to program the board. This handle is usually saved in the program in a global variable for later use when calling other functions. The initialize function does not change the state of the board or its setting. All boards types have the **Gx5642Initialize** function.

Board Handle

The board handle argument *nHandle* passed (by reference) to the parameter *pnHandle* of the **Gx5642Initialize** is a short integer (16 bits) number. It is used by the GXPIO driver functions to identify the board being accessed by the application. Since the driver supports many boards at the same time, the *nHandle* argument is required to identify which board is being programmed.

The *nHandle* is created when the application calls the **Gx5642Initialize** function. There is no need to destroy the handle. Calling **Gx5642Initialize** with the same slot number will return the same handle.

Once the board is initialized the handle can be used with other functions calls to program the board.

Reset

The Reset function sets the board to a known default state. A reset is usually performed after the board is initialized. All boards have the **Gx5642Reset**(*nHandle*, *nStatus*) function. See the Function Reference for more information regarding the specific board.

Error Handling

All the GXPIO function returns status - *pnStatus* - in the last parameter. This parameter can be later used for error handling. The status is zero for success status or less than zero for errors. When the status is error, the program can call the **GxPioGetErrorString** function to return a string representing the error. The **GxPioGetErrorString** reference contains possible error numbers and their associated error strings.

Driver Version

The **GxPioGetDriverSummary** function can be used to return the current GXPIO driver version. It can be used to differentiate between the driver versions. See the Function Reference for more information.

Distributing the Driver

Once the application is developed, the driver files (GxPio.dll and the HW device driver files located in the HW folder) can be shipped with the application. Typically, the GxPio.dll should be copied to the Windows System directory. The HW device driver files should be installed using a special setup program HWSETUP.EXE that is provided with GXPIO driver files. Alternatively, you can provide the GXPIO disk to be installed along with the board.

Chapter 5 - Functions Reference

Introduction

The GX5642 driver functions reference chapter is organized in alphabetical order. Each function is presented starting with the syntax of the function, a short description of the function parameters description and type followed by a Comments, an Example (written in C), and a See Also sections.

All function parameters follow the same rules:

Strings are ASCIIZ (null or zero character terminated).

Most function's first parameter is *nHandle* (16-bit integer). This parameter is required required for operating the board and is returned by the **Gx5642Initialize** function. The *nHandle* is used to identify the board when calling a function for programming and controlling the operation of that board.

All functions return a status with the last parameter named *pnStatus*. The *pnStatus* is zero if the function was successful, or less than a zero on error. The description of the error is available using the **GxSWGetErrorString** function or by using a predefined constant, defined in the driver interface files: GXSW.H, GXSW.BAS, GXSW.PAS or GX5642.DRV.

Parameter name are prefixed as follows:

Prefix	Type	Example
a	Array, prefix this before the simple type.	*anArray* (Array of Short)
n	Short (signed 16-bit)	nMode
d	Double - 8 bytes floating point	dReading
dw	Double word (unsigned 32-bit)	dwTimeout
l	Long (signed 32-bit)	lBits
p	Pointer. Usually used to return a value. Prefix this before the simple type.	pnStatus
sz	Null (zero value character) terminated string	szMsg
w	Unsigned short (unsigned 16-bit)	wParam
hwnd	Window handle (32-bit integer).	hwndPanel

Table 5-1: Parameter Prefixes

GX5642 Functions

The following list is a summary of functions available for the GX5642:

Driver Functions	Description
General	
Gx5642Initialize	Initializes the GX5642 driver for the specified PXI slot.
Gx5642Reset	Opens all the board relays.
Gx5642GetBoardSummary	Returns the board summary.
GxPioGetDriverSummary	Returns the driver name and version.
GxPioGetErrorString	Returns the error string associated with the specified error number.
Channel settings	
Gx5642GetChannelConversionMode	Returns the specified channel Conversion Mode.
Gx5642GetChannelDifferentialPort	Returns the specified channel Differential Port value.
Gx5642GetChannelDifferentialPortDirection	Returns the specified channel Differential Port Direction.
Gx5642GetChannelMode	Returns the specified channel operating mode.
Gx5642GetChannelOutputState	Returns the specified channel Output State.
Gx5642GetChannelTTLPort	Returns the specified channel TTL Port value.
Gx5642GetChannelTTLPortDirection	Returns the specified channel Port Direction.
Gx5642SetChannelConversionMode	Sets the specified channel Conversion Mode.
Gx5642SetChannelDifferentialPort	Sets the specified channel Differential Port value.
Gx5642SetChannelDifferentialPortDirection	Sets the specified channel Differential Port Direction.
Gx5642SetChannelMode	Sets the specified channel operating mode.
Gx5642SetChannelOutputState	Sets the specified channel Output State.
Gx5642SetChannelTTLPort	Sets the specified channel TTL Port value.
Gx5642SetChannelTTLPortDirection	Sets the specified channel TTL Port Direction.
Group settings	
Gx5642GetGroupConversionMode	Returns the specified group conversion mode.
Gx5642GetGroupDifferentialPort	Returns the specified group differential port value.
Gx5642GetGroupDifferentialPortDirection	Returns the specified group differential Port Direction.
Gx5642GetGroupMode	Returns the specified group mode.
Gx5642GetGroupOutputState	Returns the specified group output state.
Gx5642GetGroupTTLPort	Returns the specified group TTL Port value.
Gx5642GetGroupTTLPortDirection	Returns the specified group TTL Port Direction.
Gx5642LoadGroupDirectionFromDIPSwitch	Load and set the specified group Direction according to the on-board DIP Switch settings.
Gx5642ResetGroup	Resets the specified group its default settings.

Driver Functions	Description
Gx5642SetGroupConversionMode	Sets the specified group conversion mode.
Gx5642SetGroupDifferentialPort	Sets the specified group differential Port value.
Gx5642SetGroupDifferentialPortDirection	Sets the specified group differential Port Direction.
Gx5642SetGroupMode	Sets the specified group mode.
Gx5642SetGroupOutputState	Sets the specified group output state.
Gx5642SetGroupTTLPort	Sets the specified group TTL Port value.
Gx5642SetGroupTTLPortDirection	Sets the specified group TTL Port Direction.

Gx5642GetBoardSummary

Purpose

Returns the board summary.

Syntax

Gx5642GetBoardSummary (*nHandle, szSummary, nSumMaxLen, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle to a GX5642 board.
szSummary	PSTR	Buffer to contain the returned board info (null terminated) string.
nSumMaxLen	SHORT	Size of the buffer to contain the board info string.
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

The GX5642 summary string provides the following data from in the order shown:

- Instrument Name (e.g., GX5642)
- FPGA version (e.g. 0xA002)
- Serial Number (e.g. 56420210)

For example, the returned string looks like the following:

```
"GX5642, FPGA-Version:0xA002, S/N 56420210"
```

See Also

GxPioetDriverSummary, GxPioGetErrorString

Gx5642GetChannelConversionMode

Purpose

Returns the specified channel Conversion Mode.

Syntax

Gx5642GetChannelConversionMode (*nHandle*, *nChannel*, *pnMode*, *pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nChannel	SHORT	Channel range is: GX5642_FIRST_CHANNEL (0) to GX5642_LAST_CHANNEL (63).
pnMode	PSHORT	Conversion Mode can be as follows: 0. GX5642_CHANNEL_CONVERT_DIFFERENTIAL_TO_TTL – convert differential signal to TTL level. 1. GX5642_CHANNEL_CONVERT_TTL_TO_DIFFERENTIAL – convert TTL level to differential.
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

The function returns the specified channel conversion mode: differential signal to TTL level or convert TTL level to differential.

For protection, by default all channels are disabled after power-up and reset. The channels can be enabled and or disabled by calling **Gx5642SetChannelOutputState**.

Note: The channel should be set to conversion mode prior calling this function by calling **Gx5642SetChannelMode** otherwise **t**he function returns an error.

Example

The following example sets channel 0 to conversion mode, sets the conversion to be TTL level to differential, enables the channel and returns the channels conversion mode:

```
SHORT nMode;

Gx5642SetChannelMode (nHandle, 0, GX5642_CHANNEL_MODE_CONVERSION, &nStatus);
Gx5642SetChannelConversionMode (nHandle, 0, GX5642_CHANNEL_CONVERT_TTL_TO_DIFFERENTIAL,
                                &nStatus);
Gx5642SetChannelOutputState(nHandle, 0, GX5642_CHANNEL_OUTPUT_ENABLE, &nStatus);
Gx5642GetChannelConversionMode (nHandle, 0, &nMode, &nStatus);
```

See Also

Gx5642SetChannelMode, Gx5642SetChannelConversionMode, Gx5642SetChannelOutputState, GxPioGetErrorString

Gx5642GetChannelDifferentialPort

Purpose

Returns the specified channel Differential Port value.

Syntax

Gx5642GetChannelDifferentialPort (*nHandle, nChannel, pbData, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nChannel	SHORT	Channel range is: GX5642_FIRST_CHANNEL (0) to GX5642_LAST_CHANNEL (63).
pbData	PBOOL	Differential Port value: 0. Logic low 1. Logic high
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

For protection, by default all channels are disabled after power-up and reset. The channels can be enabled and or disabled by calling **Gx5642SetChannelOutputState**.

Example

The following example returns channel 0 Differential Port value:

```
BOOL bData;
Gx5642GetChannelDifferentialPort (nHandle, 0, &bData, &nStatus);
```

See Also

Gx5642SetChannelDifferentialPort, **GxPioGetErrorString**

Gx5642GetChannelDifferentialPortDirection

Purpose

Returns the specified channel Differential Port Direction.

Syntax

Gx5642GetChannelDifferentialPortDirection (*nHandle, nChannel, pnDirection, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nChannel	SHORT	Channel range is: GX5642_FIRST_CHANNEL (0) to GX5642_LAST_CHANNEL (63).
pnDirection	PSHORT	The channel Differential Port Direction can be as follows: 0. GX5642_CHANNEL_PORT_INPUT – channel port is set as input. 1. GX5642_CHANNEL_PORT_OUTPUT – channel port is set as output.
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

For protection, by default all channels are disabled after power-up and reset. The channels can be enabled and or disabled by calling **Gx5642SetChannelOutputState**.

Note: The channel should be set to Static I/O mode prior calling this function by calling **Gx5642SetChannelMode** otherwise **t**he function returns an error.

Example

The following example returns channel 0 Differential Port Direction:

```
SHORT nDirection;
Gx5642GetChannelDifferentialPortDirection (nHandle, 0, &nDirection, &nStatus);
```

See Also

Gx5642SetChannelDifferentialPortDirection, **Gx5642SetChannelDifferentialPort**, **Gx5642SetChannelMode, GxPioGetErrorString**

Gx5642GetChannelMode

Purpose

Returns the specified channel operating mode.

Syntax

Gx5642GetChannelMode (*nHandle, nChannel, pnMode, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nChannel	SHORT	Channel range is: GX5642_FIRST_CHANNEL (0) to GX5642_LAST_CHANNEL (63).
pnMode	PSHORT	Channel operating modes are as follows: 0. GX5642_CHANNEL_MODE_CONVERSION – channels is set for conversion mode. 1. GX5642_CHANNEL_MODE_STATIC_IO – channels is set for static I/O mode.
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

For protection, by default all channels are disabled after power-up and reset. The channels can be enabled and or disabled by calling **Gx5642SetChannelOutputState**.

Example

The following example returns channel 0 mode:

```
Gx5642GetChannelMode (nHandle, 0, &nMode, &nStatus);
```

See Also

Gx5642SetChannelMode, **Gx5642GetChannelDifferentialPortDirection**, **Gx5642SetChannelDifferentialPort**, **GxPioGetErrorString**

Gx5642GetChannelOutputState

Purpose

Returns the specified channel Output State.

Syntax

Gx5642GetChannelOutputState (*nHandle*, *nChannel*, *pnState*, *pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nChannel	SHORT	Channel range is: GX5642_FIRST_CHANNEL (0) to GX5642_LAST_CHANNEL (63).
pnState	PSHORT	Channel output states are as follows: 0. GX5642_CHANNEL_OUTPUT_DISABLE - channel output is disabled. 1. GX5642_CHANNEL_OUTPUT_ENABLE - channel output is enabled.
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

For protection, by default all channels are disabled after power-up and reset. The channels can be enabled and or disabled by calling **Gx5642SetChannelOutputState**.

Example

The following returns channel 0 output state:

```
SHORT nState;
Gx5642GetChannelOutputState (nHandle, 0, &nState, &nStatus);
```

See Also

Gx5642SetChannelOutputState, **Gx5642GetChannelMode**, **Gx5642GetChannelDifferentialPortDirection**, **Gx5642SetChannelDifferentialPort**, **GxPioGetErrorString**

Gx5642GetChannelTTLPort

Purpose

Returns the specified channel TTL Port value.

Syntax

Gx5642GetChannelTTLPort (*nHandle*, *nChannel*, *pbData*, *pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nChannel	SHORT	Channel range is: GX5642_FIRST_CHANNEL (0) to GX5642_LAST_CHANNEL (63).
pbData	PBOOL	TTL Port value: 0. Logic low 1. Logic high
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

For protection, by default all channels are disabled after power-up and reset. The channels can be enabled and or disabled by calling **Gx5642SetChannelOutputState**.

Example

The following example returns channel 0 TTL Port value:

```
BOOL bData;
Gx5642GetChannelTTLPort (nHandle, 0, &bData, &nStatus);
```

See Also

Gx5642SetChannelTTLPort, GxPioGetErrorString

Gx5642GetChannelTTLPortDirection

Purpose

Returns the specified channel Port Direction.

Syntax

Gx5642GetChannelTTLPortDirection (*nHandle, nChannel, pnDirection, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nChannel	SHORT	Channel range is: GX5642_FIRST_CHANNEL (0) to GX5642_LAST_CHANNEL (63).
pnDirection	PSHORT	The channel TTL Port Direction can be as follows: 0. GX5642_CHANNEL_PORT_INPUT – channel port is set as input. 1. GX5642_CHANNEL_PORT_OUTPUT – channel port is set as output.
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

For protection, by default all channels are disabled after power-up and reset. The channels can be enabled and or disabled by calling **Gx5642SetChannelOutputState**.

Note: The channel should be set to Static I/O mode prior calling this function by calling **Gx5642SetChannelMode** otherwise **t**he function returns an error.

Example

The following example returns channel 0 TTL Port Direction:

```
SHORT nDirection;
Gx5642GetChannelTTLPortDirection (nHandle, 0, &nDirection, &nStatus);
```

See Also

Gx5642SetChannelTTLPortDirection, **Gx5642SetChannelTTLPort**, **Gx5642SetChannelMode, GxPioGetErrorString**

Gx5642GetGroupConversionMode

Purpose

Returns the specified group channels conversion mode.

Syntax

Gx5642GetGroupConversionMode (*nHandle, nGroup, pdwMode, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nGroup	SHORT	Group value is as follows: 0. GX5642_GROUP0 (channels 0 to 31) 1. GX5642_GROUP1(channels 32 to 63)
pdwMode	PDWORD	Each of the 32 bits represents a channel in the group, bit 0 is the first channel in the group and bit 31 is the last channel in the group. Bit low - GX5642_CHANNEL_CONVERT_DIFFERENTIAL_TO_TTL – convert differential signal to TTL level. Bit high - GX5642_CHANNEL_CONVERT_TTL_TO_DIFFERENTIAL – convert TTL level to differential.
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

The function returns the conversion mode for all the channels in the specified group.

For protection, by default all channels are disabled after power-up and reset. The channels can be enabled and or disabled by calling **Gx5642SetChannelOutputState**.

Note: The group should be set to conversion mode prior calling this function by calling **Gx5642SetGroupMode**.

Example

```
DWORD dwMode;

// Sets all the channels in group 0 to conversion mode.
Gx5642SetGroupMode (nHandle, GX5642_GROUP0, 0, &nStatus);
// sets channels 1, 5, 8 TTL to differential conversion mode, rest of the channels are set to
// convert differential to TTL level.
Gx5642SetGroupConversionMode (nHandle, GX5642_GROUP0, 0x00000121, &nStatus);
// Enables all channels outputs.
Gx5642SetGroupOutputState (nHandle, GX5642_GROUP0, 0xFFFFFFFF, &nStatus);
// Returns all the channels in group 0 conversion mode
Gx5642GetCGroupConversionMode (nHandle, GX5642_GROUP0, &dwMode, &nStatus);
```

See Also

Gx5642SetGroupMode, Gx5642SetGroupConversionMode, Gx5642SetGroupOutputState, GxPioGetErrorString

Gx5642GetGroupDifferentialPort

Purpose

Returns the specified group differential ports values.

Syntax

Gx5642GetGroupDifferentialPort (*nHandle, nGroup, pdwPortData, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nGroup	SHORT	Group value is as follows: 0. GX5642_GROUP0 (channels 0 to 31) 1. GX5642_GROUP1(channels 32 to 63)
pdwPortData	PDWORD	Group's differential ports values. Each of the 32 bits represents a channel in the group. Bit 0 is the first channel in the group and bit 31 is the last channel in the group. Bit low - differential channel port is logic low. Bit high - differential channel port is logic high.
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

The function returns the differential ports values for all the channels in the specified group.

For protection, by default all channels are disabled after power-up and reset. The channels can be enabled and or disabled by calling **Gx5642SetChannelOutputState**.

Example

The following example returns group 0 Differential Ports values:

```
DWORD dwData;
Gx5642GetGroupDifferentialPort (nHandle, GX5642_GROUP0, &dwData, &nStatus);
```

See Also

Gx5642SetGroupDifferentialPort, **GxPioGetErrorString**

Gx5642GetGroupDifferentialPortDirection

Purpose

Returns the specified group differential Port Direction.

Syntax

Gx5642GetGroupDifferentialPortDirection (*nHandle, nGroup, pdwDirection, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nGroup	SHORT	Group value is as follows: 0. GX5642_GROUP0 (channels 0 to 31) 1. GX5642_GROUP1(channels 32 to 63)
pdwDirection	PDWORD	Group's differential Port Direction. Each of the 32 bits represents a channel in the group. Bit 0 is the first channel in the group and bit 31 is the last channel in the group. Bit low - channel differential Port Direction is input. Bit high - channel differential Port Direction is output.
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

The function returns the differential Port Direction for all the channels in the specified group.

Note: Only channels in the group that were set to Static I/O mode prior calling this function by calling **Gx5642SetGroupMode** will be set.

Example

The following example returns group 0 Differential Port Direction:

```
DWORD dwDirection;
Gx5642GetGroupDifferentialPortDirection (nHandle, GX5642_GROUP0, &dwDirection, &nStatus);
```

See Also

Gx5642SetGroupDifferentialPortDirection, **Gx5642SetGroupDifferentialPort**, **Gx5642SetGroupMode, GxPioGetErrorString**

Gx5642GetGroupMode

Purpose

Returns the specified group mode.

Syntax

Gx5642GetGroupMode (*nHandle, nGroup, pdwMode, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nGroup	SHORT	Group value is as follows: 0. GX5642_GROUP0 (channels 0 to 31) 1. GX5642_GROUP1(channels 32 to 63)
pdwMode	PDWORD	Group's Mode. Each of the 32 bits represents a channel in the group. Bit 0 is the first channel in the group and bit 31 is the last channel in the group. Bit low - channel set for conversion mode. Bit high - channel set for static I/O mode.
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

The function returns the operating mode for all the channels in the specified group.

For protection, by default all channel's connections (TTL and Deferential) are disabled after power-up and reset. The channels can be enabled and or disabled by calling **Gx5642SetGroupOutputState**.

Example

The following returns group 0 operating mode:

```
Gx5642GetGroupMode (nHandle, GX5642_GROUP0, &dwMode, &nStatus);
```

See Also

Gx5642SetGroupMode, Gx5642SetGroupConversionMode, Gx5642SetGroupOutputState, GxPioGetErrorString

Gx5642GetGroupOutputState

Purpose

Returns the specified group output state.

Syntax

Gx5642GetGroupOutputState (*nHandle, nGroup, pdwStates, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nGroup	SHORT	Group value is as follows: 0. GX5642_GROUP0 (channels 0 to 31) 1. GX5642_GROUP1(channels 32 to 63)
pdwStates	PDWORD	Group's output state. Each of the 32 bits represents a channel in the group. Bit 0 is the first channel in the group and bit 31 is the last channel in the group. Bit low - channel output is disabled. Bit high - channel output is enabled.
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

For protection, by default all channel's connections (TTL and Deferential) are disabled after power-up and reset. The channels can be enabled and or disabled by calling **Gx5642SetChannelOutputState**.

Example

Example

```
DWORD dwMode;

// Sets all the channels in group 0 to conversion mode.
Gx5642SetGroupMode (nHandle, GX5642_GROUP0, 0, &nStatus);
// sets channels 1, 5, 8 TTL to differential conversion mode, rest of the channels are set to
// convert differential to TTL level.
Gx5642SetGroupConversionMode (nHandle, GX5642_GROUP0, 0x00000121, &nStatus);
// Enables all channels outputs.
Gx5642SetGroupOutputState (nHandle, GX5642_GROUP0, 0xFFFFFFFF, &nStatus);
// Returns all the channels in group 0 conversion mode
Gx5642GetCGroupConversionMode (nHandle, GX5642_GROUP0, &dwMode, &nStatus);
```

See Also

Gx5642SetGroupOutputState, Gx5642SetGroupMode, Gx5642SetGroupConversionMode, GxPioGetErrorString

Gx5642GetGroupTTLPort

Purpose

Returns the specified group TTL Ports values.

Syntax

Gx5642GetGroupTTLPort (*nHandle*, *pdwPortData*, *pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nGroup	SHORT	Group value is as follows: 0. GX5642_GROUP0 (channels 0 to 31) 1. GX5642_GROUP1(channels 32 to 63)
pdwPortData	PDWORD	Group's differential ports values. Each of the 32 bits represents a channel in the group. Bit 0 is the first channel in the group and bit 31 is the last channel in the group. Bit low - differential channel port is logic low. Bit high - differential channel port is logic high.
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

The function returns the TTL Ports values for all the channels in the specified group.

For protection, by default all channels are disabled after power-up and reset. The channels can be enabled and or disabled by calling **Gx5642SetChannelOutputState**.

Example

The following example returns group 0 TTL Ports values:

```
DWORD dwData;
Gx5642GetGroupTTLPort (nHandle, GX5642_GROUP0, &dwData, &nStatus);
```

See Also

Gx5642SetGroupTTLPort, **GxPioGetErrorString**

Gx5642GetGroupTTLPortDirection

Purpose

Returns the specified group TTL Port Direction.

Syntax

Gx5642GetGroupTTLPortDirection (*nHandle, nGroup, pdwDirection, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nGroup	SHORT	Group value is as follows: 0. GX5642_GROUP0 (channels 0 to 31) 1. GX5642_GROUP1(channels 32 to 63)
pdwDirection	PDWORD	Group's TTL Port Direction. Each of the 32 bits represents a channel in the group. Bit 0 is the first channel in the group and bit 31 is the last channel in the group. Bit low - channel differential Port TTL is input. Bit high - channel differential Port TTL is output.
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

The function returns the TTL Port Direction for all the channels in the specified group.

Note: Only channels in the group that were set to Static I/O mode prior calling this function by calling **Gx5642SetGroupMode** will be set.

Example

The following example returns group 0 TTL Port Direction:

```
DWORD dwDirection;
Gx5642GetGroupTTLPortDirection (nHandle, GX5642_GROUP0, &dwDirection, &nStatus);
```

See Also

Gx5642SetGroupTTLPortDirection, **Gx5642SetGroupTTLPort**, **Gx5642SetGroupMode, GxPioGetErrorString**

Gx5642Initialize

Purpose

Initializes the driver for the board at the specified slot number. The function returns a handle that can be used with other GX5642 functions to program the board.

Syntax

Gx5642Initialize (*nSlot, pnHandle, pnStatus*)

Parameters

Name	Type	Comments
nSlot	Short	GX5642 board slot number on the PXI bus.
pnHandle	PSHORT	Returned handle for the board. The handle is set to zero on error and <> 0 on success.
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

The **Gx5642Initialize** function verifies whether or not the GX5642 board exists in the specified PXI slot. The function does not change any of the board settings.

The returned handle *pnHandle* is used to identify the specified board with other GX5642 functions.

Example

The following example initializes two GX5642 boards at slot 1 and 2.

```
SHORT nHandle1, nHandle2, nStatus;
Gx5642Initilize (1, &nHandle1, &nStatus);
Gx5642Initilize (2, &nHandle2, &nStatus);
if (nHandle1==0 || nHandle2==0)
    {   printf(“Unable to Initialize the board”)
        return;
}
```

See Also

Gx5642Reset, **GxPioGetErrorString**

Gx5642LoadGroupDirectionFromDIPSwitch

Purpose

Load and set the specified group Direction according to the on-board DIP Switch settings.

Syntax

Gx5642LoadGroupDirectionFromDIPSwitch (*nHandle, nGroup, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nGroup	SHORT	Group value is as follows: 0. GX5642_GROUP0 (channels 0 to 31) 1. GX5642_GROUP1(channels 32 to 63)
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

The group's channels conversion settings will be loaded and set according to the on-board conversion direction DIP-Switches.

Note: All channels in the specified group need to be set to static I/O mode prior calling this function by calling **Gx5642SetGroupMode**.

Example

The following example load and set the group 0 Direction according to the on-board DIP Switch settings:

```
Gx5642LoadGroupDirectionFromDIPSwitch (nHandle, GX5642_GROUP0, &nStatus);
```

See Also

Gx5642SetGroupConversionMode,.Gx5642SetGroupMode, Gx5642SetGroupOutputState, GxPioGetErrorString

Gx5642Reset

Purpose

Resets the GX5642 board to its default settings.

Syntax

Gx5642Reset (*nHandle*, *pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

After calling this function all the channels will be disabled and in conversion mode of differential signal to TTL level.

Example

The following example initializes and resets the GX5642 board:

```
Gx5642Initialize (1, &nHandle, &nStatus);
Gx5642Reset (nHandle, &nStatus);
```

See Also

Gx5642Initialize, **GxPioGetErrorString**

Gx5642ResetGroup

Purpose

Resets the specified group its default settings.

Syntax

Gx5642ResetGroup (*nHandle, nGroup, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nGroup	SHORT	Group value is as follows: 0. GX5642_GROUP0 (channels 0 to 31) 1. GX5642_GROUP1(channels 32 to 63)
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

After calling this function the specified group's channels will be disabled and in conversion mode of differential signal to TTL level.

Example

The following example rests group 0:

```
Gx5642ResetGroup (nHandle, GX5642_GROUP0, &nStatus);
```

See Also

Gx5642GetChannelMode, **Gx5642GetGroupMode**, **Gx5642GetChannelOutputState**, **Gx5642GetGroupOutputState**, **Gx5642Initialize**, **GxPioGetErrorString**

Gx5642SetChannelConversionMode

Purpose

Sets the specified channel Conversion Mode.

Syntax

Gx5642SetChannelConversionMode (*nHandle, nChannel, nMode, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nChannel	SHORT	Channel range is: GX5642_FIRST_CHANNEL (0) to GX5642_LAST_CHANNEL (63).
nMode	SHORT	Conversion Mode can be as follows: 0. GX5642_CHANNEL_CONVERT_DIFFERENTIAL_TO_TTL – convert differential signal to TTL level. 1. GX5642_CHANNEL_CONVERT_TTL_TO_DIFFERENTIAL – convert TTL level to differential.
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

The function sets the specified channel to convert differential signal to TTL level or convert TTL level to differential.

For protection, by default all channels are disabled after power-up and reset. The channels can be enabled and or disabled by calling **Gx5642SetChannelOutputState**.

Note: The channel should be set to conversion mode prior calling this function by calling **Gx5642SetChannelMode** otherwise **t**he the function will return an error.

Example

The following example sets channel 0 to conversion mode, sets the conversion to be TTL level to differential, enables the channel and returns the channels conversion mode:

```
SHORT nMode;

Gx5642SetChannelMode (nHandle, 0, GX5642_CHANNEL_MODE_CONVERSION, &nStatus);
Gx5642SetChannelConversionMode (nHandle, 0, GX5642_CHANNEL_CONVERT_TTL_TO_DIFFERENTIAL,
                                &nStatus);
Gx5642SetChannelOutputState(nHandle, 0, GX5642_CHANNEL_OUTPUT_ENABLE, &nStatus);
Gx5642GetChannelConversionMode (nHandle, 0, &nMode, &nStatus);
```

See Also

Gx5642SetChannelMode, Gx5642GetChannelConversionMode, Gx5642SetChannelOutputState, GxPioGetErrorString

Gx5642SetChannelDifferentialPort

Purpose

Sets the specified channel Differential Port value.

Syntax

Gx5642SetChannelDifferentialPort (*nHandle, nChannel, bData, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nChannel	SHORT	Channel range is: GX5642_FIRST_CHANNEL (0) to GX5642_LAST_CHANNEL (63).
pbData	PBOOL	Differential Port value: 0. Logic low 1. Logic high
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

The function sets the Differential Port value. If the Differential Port direction was set to output then the function returns the current channel's port settings, if the port direction was set to input was then the function returns the Differential input value.

For protection, by default all channels are disabled after power-up and reset. The channels can be enabled and or disabled by calling **Gx5642SetChannelOutputState**.

Note: The channel should be set to Static I/O mode prior calling this function by calling **Gx5642SetChannelMode** otherwise **t**he function returns an error.

Example

The following example sets channel 0 Differential Port value to 1:

```
Gx5642SetChannelDifferentialPort (nHandle, 0, 1, &nStatus);
```

See Also

Gx5642SetChannelDifferentialPort, **Gx5642SetChannelMode, GxPioGetErrorString**

Gx5642SetChannelDifferentialPortDirection

Purpose

Sets the specified channel Differential Port Direction.

Syntax

Gx5642SetChannelDifferentialPortDirection (*nHandle, nChannel, nDirection, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nChannel	SHORT	Channel range is: GX5642_FIRST_CHANNEL (0) to GX5642_LAST_CHANNEL (63).
nDirection	SHORT	The channel Differential Port Direction can be as follows: 0. GX5642_CHANNEL_PORT_INPUT – channel port is set as input. 1. GX5642_CHANNEL_PORT_OUTPUT – channel port is set as output.
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

For protection, by default all channels are disabled after power-up and reset. The channels can be enabled and or disabled by calling **Gx5642SetChannelOutputState**.

Note: The channel should be set to Static I/O mode prior calling this function by calling **Gx5642SetChannelMode** otherwise **t**he function returns an error.

Example

The following example sets channel 0 Differential Port Direction to output:

```
Gx5642SetChannelDifferentialPortDirection (nHandle, 0, GX5642_CHANNEL_PORT_OUTPUT, &nStatus);
```

See Also

Gx5642GetChannelDifferentialPortDirection, **Gx5642SetChannelDifferentialPort**, **Gx5642SetChannelMode, GxPioGetErrorString**

Gx5642SetChannelMode

Purpose

Sets the specified channel operating mode.

Syntax

Gx5642SetChannelMode (*nHandle, nChannel, nMode, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nChannel	SHORT	Channel range is: GX5642_FIRST_CHANNEL (0) to GX5642_LAST_CHANNEL (63).
nMode	SHORT	Channel operating modes are as follows: 0. GX5642_CHANNEL_MODE_CONVERSION – channels is set for conversion mode. 1. GX5642_CHANNEL_MODE_STATIC_IO - – channels is set for static I/O mode.
PnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

For protection, by default all channels are disabled after power-up and reset. The channels can be enabled and or disabled by calling **Gx5642SetChannelOutputState**.

Example

The following example sets channel 0 operating mode to static I/O:

```
Gx5642GetChannelMode (nHandle, 0, GX5642_CHANNEL_MODE_STATIC_IO, &nStatus);
```

See Also

Gx5642GetChannelMode, **Gx5642GetChannelDifferentialPortDirection**, **Gx5642SetChannelDifferentialPort**, **GxPioGetErrorString**

Gx5642SetChannelOutputState

Purpose

Sets the specified channel Output State.

Syntax

Gx5642SetChannelOutputState (*nHandle, nChannel, nState, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nChannel	SHORT	Channel range is: GX5642_FIRST_CHANNEL (0) to GX5642_LAST_CHANNEL (63).
pnState	PSHORT	Channel output states are as follows: 0. GX5642_CHANNEL_OUTPUT_DISABLE - channel output is disabled. 1. GX5642_CHANNEL_OUTPUT_ENABLE - channel output is enabled.
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

For protection, by default all channels are disabled after power-up and reset. The channels can be enabled and or disabled by calling **Gx5642SetChannelOutputState**.

Example

The following enables channel 0 output state:

```
SHORT nState;
Gx5642SetChannelOutputState (nHandle, 0, GX5642_CHANNEL_OUTPUT_ENABLE, &nStatus);
```

See Also

Gx5642GetChannelOutputState, **Gx5642GetChannelMode**, **Gx5642GetChannelDifferentialPortDirection**, **Gx5642SetChannelDifferentialPort**, **GxPioGetErrorString**

Gx5642SetChannelTTLPort

Purpose

Sets the specified channel TTL Port value.

Syntax

Gx5642SetChannelTTLPort (*nHandle, nChannel, bData, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nChannel	SHORT	Channel range is: GX5642_FIRST_CHANNEL (0) to GX5642_LAST_CHANNEL (63).
bData	BOOL	TTL Port value: 0. Logic low 1. Logic high
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

The function sets the TTL Port value.

For protection, by default all channels are disabled after power-up and reset. The channels can be enabled and or disabled by calling **Gx5642SetChannelOutputState**.

Note: The channel should be set to Static I/O mode prior calling this function by calling **Gx5642SetChannelMode** otherwise **t**he function returns an error.

Example

The following example sets channel 0 TTL Port value to 1:

```
Gx5642SetChannelTTLPort (nHandle, 0, 1, &nStatus);
```

See Also

Gx5642GetChannelTTLPort, **Gx5642SetChannelMode, GxPioGetErrorString**

Gx5642SetChannelTTLPortDirection

Purpose

Sets the specified channel TTL Port Direction.

Syntax

Gx5642SetChannelTTLPortDirection (*nHandle, nChannel, nDirection, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nChannel	SHORT	Channel range is: GX5642_FIRST_CHANNEL (0) to GX5642_LAST_CHANNEL (63).
nDirection	SHORT	The channel TTL Port Direction can be as follows: 0. GX5642_CHANNEL_PORT_INPUT – channel port is set as input. 1. GX5642_CHANNEL_PORT_OUTPUT – channel port is set as output.
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

For protection, by default all channels are disabled after power-up and reset. The channels can be enabled and or disabled by calling **Gx5642SetChannelOutputState**.

Note: The channel should be set to Static I/O mode prior calling this function by calling **Gx5642SetChannelMode** otherwise **t**he function returns an error.

Example

The following example sets channel 0 TTL Port Direction to output:

```
Gx5642SetChannelTTLPortDirection (nHandle, 0, GX5642_CHANNEL_PORT_OUTPUT, &nStatus);
```

See Also

Gx5642GetChannelTTLPortDirection, **Gx5642SetChannelTTLPort**, **Gx5642SetChannelMode, GxPioGetErrorString**

Gx5642SetGroupConversionMode

Purpose

Sets the specified group channels conversion mode.

Syntax

Gx5642SetGroupConversionMode (*nHandle, nGroup, dwMode, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nGroup	SHORT	Group value is as follows: 0. GX5642_GROUP0 (channels 0 to 31) 1. GX5642_GROUP1(channels 32 to 63)
dwMode	DWORD	Each of the 32 bits represents a channel in the group, bit 0 is the first channel in the group and bit 31 is the last channel in the group. Bit low - GX5642_CHANNEL_CONVERT_DIFFERENTIAL_TO_TTL – convert differential signal to TTL level. Bit high - GX5642_CHANNEL_CONVERT_TTL_TO_DIFFERENTIAL – convert TTL level to differential.
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

The function returns the conversion mode for all the channels in the specified group.

For protection, by default all channels are disabled after power-up and reset. The channels can be enabled and or disabled by calling **Gx5642SetChannelOutputState**.

Note: The group should be set to conversion mode prior calling this function by calling **Gx5642SetGroupMode**.

Example

Example

```
DWORD dwMode;

// Sets all the channels in group 0 to conversion mode.
Gx5642SetGroupMode (nHandle, GX5642_GROUP0, 0, &nStatus);
// sets channels 1, 5, 8 TTL to differential conversion mode, rest of the channels are set to
// convert differential to TTL level.
Gx5642SetGroupConversionMode (nHandle, GX5642_GROUP0, 0x00000121, &nStatus);
// Enables all channels outputs.
Gx5642SetGroupOutputState (nHandle, GX5642_GROUP0, 0xFFFFFFFF, &nStatus);
// Returns all the channels in group 0 conversion mode
Gx5642GetCGroupConversionMode (nHandle, GX5642_GROUP0, &dwMode, &nStatus);
```

See Also

Gx5642GetGroupMode, Gx5642SetGroupConversionMode, Gx5642SetGroupOutputState, GxPioGetErrorString

Gx5642SetGroupDifferentialPort

Purpose

Sets the specified group differential Port value.

Syntax

Gx5642SetGroupDifferentialPort (*nHandle*, *nGroup*, *dwPortData*, *pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nGroup	SHORT	Group value is as follows: 0. GX5642_GROUP0 (channels 0 to 31) 1. GX5642_GROUP1(channels 32 to 63)
dwPortData	DWORD	Group's differential ports values. Each of the 32 bits represents a channel in the group. Bit 0 is the first channel in the group and bit 31 is the last channel in the group. Bit low - differential channel port is logic low. Bit high - differential channel port is logic high.
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

The function sets the differential ports values for all the channels in the specified group.

For protection, by default all channels are disabled after power-up and reset. The channels can be enabled and or disabled by calling **Gx5642SetChannelOutputState**.

Note: Only channels in the group that were set to Static I/O mode prior calling this function by calling **Gx5642SetGroupMode** will be set.

Example

The following example sets group 0 differential Port value:

```
Gx5642SetGroupDifferentialPort (nHandle, GX5642_GROUP0, 0x55AA1234&nStatus);
```

See Also

Gx5642GetChannelDifferentialPort, **Gx5642GetChannelDifferentialPortDirection**, **GxPioGetErrorString**

Gx5642SetGroupDifferentialPortDirection

Purpose

Sets the specified group differential Port Direction.

Syntax

Gx5642SetGroupDifferentialPortDirection (*nHandle, nGroup, dwDirection, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nGroup	SHORT	Group value is as follows: 0. GX5642_GROUP0 (channels 0 to 31) 1. GX5642_GROUP1(channels 32 to 63)
dwDirection	DWORD	Group's output state. Each of the 32 bits represents a channel in the group. Bit 0 is the first channel in the group and bit 31 is the last channel in the group. Bit low - channel port is set as input. Bit high - channel port is set as output.
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

For protection, by default all channel's connections (TTL and Deferential) are disabled after power-up and reset. The channels can be enabled and or disabled by calling **Gx5642SetGroupOutputState**.

Note: Only channels in the group that were set to Static I/O mode prior calling this function by calling **Gx5642SetGroupMode** will be set.

Example

The following example sets group 0 Differential Port channels 0, 4, 8 direction to output:

```
Gx5642SetGroupDifferentialPortDirection (nHandle, GX5642_GROUP0, 0x00000111, &nStatus);
```

See Also

Gx5642GetGroupDifferentialPortDirection, Gx5642SetGroupDifferentialPort, Gx5642SetGroupMode, GxPioGetErrorString

Gx5642SetGroupMode

Purpose

Sets the specified group mode.

Syntax

Gx5642SetGroupMode (*nHandle, nGroup, dwMode, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nGroup	SHORT	Group value is as follows: 0. GX5642_GROUP0 (channels 0 to 31) 1. GX5642_GROUP1(channels 32 to 63)
dwMode	DWORD	Group's Mode. Each of the 32 bits represents a channel in the group. Bit 0 is the first channel in the group and bit 31 is the last channel in the group. Bit low - channel set for conversion mode. Bit high - channel set for static I/O mode.
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

The function sets the operating mode for all the channels in the specified group.

For protection, by default all channel's connections (TTL and Deferential) are disabled after power-up and reset. The channels can be enabled and or disabled by calling **Gx5642SetGroupOutputState**.

Example

The following example sets the channels 1, 5, and 8 to static I/O mode:

```
Gx5642SetGroupMode (nHandle, GX5642_GROUP0, 0x00000121, &nStatus);
```

See Also

Gx5642GetGroupMode, Gx5642SetGroupConversionMode, Gx5642SetGroupOutputState, GxPioGetErrorString

Gx5642SetGroupOutputState

Purpose

Sets the specified group output state.

Syntax

Gx5642SetGroupOutputState (*nHandle*, *nGroup*, *dwStates*, *pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nGroup	SHORT	Group value is as follows: 0. GX5642_GROUP0 (channels 0 to 31) 1. GX5642_GROUP1(channels 32 to 63)
dwStates	DWORD	Group's output state. Each of the 32 bits represents a channel in the group. Bit 0 is the first channel in the group and bit 31 is the last channel in the group. Bit low - channel output is disabled. Bit high - channel output is enabled.
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

For protection, by default all channel's connections (TTL and Deferential) are disabled after power-up and reset.

Example

Example

```
DWORD dwMode;

// Sets all the channels in group 0 to conversion mode.
Gx5642SetGroupMode (nHandle, GX5642_GROUP0, 0, &nStatus);
// sets channels 1, 5, 8 TTL to differential conversion mode, rest of the channels are set to
// convert differential to TTL level.
Gx5642SetGroupConversionMode (nHandle, GX5642_GROUP0, 0x00000121, &nStatus);
// Enables all channels outputs.
Gx5642SetGroupOutputState (nHandle, GX5642_GROUP0, 0xFFFFFFFF, &nStatus);
// Returns all the channels in group 0 conversion mode
Gx5642GetCGroupConversionMode (nHandle, GX5642_GROUP0, &dwMode, &nStatus);
```

See Also

Gx5642GetGroupOutputState, Gx5642SetGroupMode, Gx5642SetGroupConversionMode, GxPioGetErrorString

Gx5642SetGroupTTLPort

Purpose

Sets the specified group TTL Port value.

Syntax

Gx5642SetGroupTTLPort (*nHandle, nGroup, dwPortData, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nGroup	SHORT	Group value is as follows: 0. GX5642_GROUP0 (channels 0 to 31) 1. GX5642_GROUP1(channels 32 to 63)
dwPortData	DWORD	Group's output state. Each of the 32 bits represents a channel in the group. Bit 0 is the first channel in the group and bit 31 is the last channel in the group. Bit low - channel logic low. Bit high - channel logic high.
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

For protection, by default all channel's connections (TTL and Deferential) are disabled after power-up and reset. The channels can be enabled and or disabled by calling **Gx5642SetGroupOutputState**.

Note: Only channels in the group that were set to Static I/O mode prior calling this function by calling **Gx5642SetGroupMode** will be set.

Example

The following example sets group 0 TTL Port channels 0, 4, 8 value to 1

```
Gx5642SetGroupTTLPort (nHandle, GX5642_GROUP0, 0x00000111, &nStatus);
```

See Also

Gx5642GetGroupTTLPort, **Gx5642SetGroupMode**, **GxPioGetErrorString**

Gx5642SetGroupTTLPortDirection

Purpose

Sets the specified group TTL Port Direction.

Syntax

Gx5642SetGroupTTLPortDirection (*nHandle, nGroup, dwDirection, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle for a GX5642 board.
nGroup	SHORT	Group value is as follows: 0. GX5642_GROUP0 (channels 0 to 31) 1. GX5642_GROUP1(channels 32 to 63)
dwDirection	DWORD	Group's output state. Each of the 32 bits represents a channel in the group. Bit 0 is the first channel in the group and bit 31 is the last channel in the group. Bit low - channel port is set as input. Bit high - channel port is set as output.
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

For protection, by default all channel's connections (TTL and Deferential) are disabled after power-up and reset. The channels can be enabled and or disabled by calling **Gx5642SetGroupOutputState**.

Note: Only channels in the group that were set to Static I/O mode prior calling this function by calling **Gx5642SetGroupMode** will be set.

Example

The following example sets group 0 TTL Port channels 0, 4 and 8 direction to output:

```
Gx5642SetGroupTTLPortDirection (nHandle, GX5642_GROUP0, 0x00000111, &nStatus);
```

See Also

Gx5642GetGroupTTLPortDirection, **Gx5642SetGroupTTLPort**, **Gx5642SetGroupMode, GxPioGetErrorString**

GxPioGetDriverSummary

Purpose

Returns the driver name and version.

Syntax

GxPioGetDriverSummary (*pszSummary ,nSummaryMaxLen, pdwVersion, pnStatus*)

Parameters

Name	Type	Comments
pszSummary	PSTR	Buffer to the returned driver summary string.
nSummaryMaxLen	SHORT	The size of the summary string buffer.
pdwVersion	PDWORD	Returned version number. The high order word specifies the major version number where the low order word specifies the minor version number.
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

The returned string is: "GXPIO Driver for GX5642. Version 2.10, Copyright © Geotest 2006.".

Example

The following example prints the driver version:

```
CHAR sz[128];
DWORD dwVersion;
SHORT nStatus;

GxPioGetDriverSummary (sz, sizeof sz, &dwVersion, &nStatus);
printf("Driver Version %d.%d", (INT)(dwVersion>>16), (INT)
    dwVersion &0xFFFF);
```

See Also

GxPioGetErrorString

GxPioGetErrorString

Purpose

Returns the error string associated with the specified error number.

Syntax

GxPioGetErrorString (*nError* , *pszMsg*, *nErrorMaxLen*, *pnStatus*)

Parameters

Name	Type	Comments
nError	SHORT	Error number.
pszMsg	PSTR	Buffer to the returned error string.
nErrorMaxLen	SHORT	The size of the error string buffer.
pnStatus	PSHORT	Returned status: 0 on success, negative number on failure.

Comments

The function returns the error string associated with the *nError* as returned from other driver functions.

The following table displays the possible error values; not all errors apply to this board type:

Resource Errors

0	No error has occurred
-1	Unable to open the HW driver. Check if HW is properly installed
-2	Board does not exist in this slot/base address
-3	Different board exist in the specified PCI slot/base address
-4	PCI slot not configured properly. You may configure using the PciExplorer from the Windows Control Panel
-5	Unable to register the PCI device
-6	Unable to allocate system resource for the device
-7	Unable to allocate memory
-8	Unable to create panel
-9	Unable to create Windows timer
-10	Bad or Wrong board EEPROM
-11	Not in calibration mode
-12	Board is not calibrated
-13	Function is not supported by the specified board

General Parameter Errors

-20	Invalid or unknown error number
-21	Invalid parameter
-22	Illegal slot number
-23	Illegal board handle
-24	Illegal string length

-25 Illegal operation mode

-26 Parameter is out of the allowed range

Parameter Errors

-40 Invalid port

-41 Invalid word

-42 Invalid byte

-43 Invalid bit

-44 Invalid counter

-45 Invalid input load control

-46 Invalid counter or all terminal counts and clocks

-47 Invalid terminal count mode

-48 Invalid clock source

-49 Invalid clock internal number

-50 Invalid clock internal source

-51 Invalid gate source

-52 Invalid clock divider value

Example

The following example initializes the board. If the initialization failed, the following error string is printed:

```
CHAR   sz[256];
SHORT  nStatus, nHandle;
..
Gx5642Initialize (3, &Handle, &Status);
if (nStatus<0)
{   GxPioGetErrorString(nStatus, sz, sizeof sz, &nStatus);
    printf(sz);// prints the error string returns

}
```

Index

S

T

V