GT7864

Programmable Discrete Output Board

User's Guide

GEOTEST – Marvin Test Systems, Inc.

Part No. GT97408 - Revision 1 (1/18/99)

Safety and Handling

Each product shipped by Geotest is carefully inspected and tested prior to shipping. The shipping carton provides protection for shipment and can be used to store both hardware and software.

The circuit board is extremely delicate and requires care in handling and installation. Do not remove the board from its protective shipping carton or plastic covering until you are ready to install it.

If the board is removed from the computer for any reason, be sure to store it in its original shipping carton. Do not store the board on a workbench or anywhere it can be dropped or subjected to strong electromagnetic or electrostatic fields. It is good practice to store the board in a protective, anti-electrostatic wrapper away from electromagnetic fields.

Make a single backup copy of the software diskette and store the original in a place safe from heat or electromagnetic or electrostatic fields.

Warranty

Geotest's products are warranted against defects in materials and workmanship for a period of 12 months (6 months for software products). Geotest shall repair or replace (at Geotest's discretion) any defective product during the stated warranty period. The software warranty includes any upgrades released during the warranty period. If you need to return a board, please call 949-263-2222 for an RMA number.

If You Need Help

If you need help at any time with the installation or use of this product, call Geotest technical support at 949-263-2222. In addition, you may get updated product information and drivers from our Internet Web Site at www.geotestinc.com.

Disclaimer

In no event shall Geotest or any of its representatives be liable for any consequential damages whatsoever (including, without limitation, damages for loss of business profits, business interruption, loss of business information, or other loss) 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

386, 486, Pentium	Intel Corporation
ATEasy, GTXI	Geotest – MTS, Inc.
Borland C++ and Delphi	Borland Corporation
IBM or IBM-PC/AT	International Business Machines Corp. (IBM).
Microsoft Developer Studio	Microsoft Corporation
Microsoft Visual Basic	Microsoft Corporation
Microsoft Visual C++	Microsoft Corporation
MS-DOS	Microsoft Corporation
Windows 3.1, Windows 95, Windows 98 and Windows NT	Microsoft Corporation

Table of Contents

Safety and Handling i
Warranty i
If You Need Help i
Disclaimer ii
Copyright ii
Trademarks ii

Chapter 1 - Introduction 1

About this Manual 1
Scope and Organization 1
Conventions Used in This Manual 2
Technical Support 3
General Information 3
Internet Home Page Download Area 3
Calibration 3
Geotest Address and Telephone Numbers 3

Chapter 2 – Overview 5

Introduction 5
Features 5
Board Diagram 7
Architecture 7
Connector 8
On-board EEPROM 10
Software 10
Virtual Panel 11
Function Library 11
Programming Example 12

Chapter 3 – Setup and Installation 13

Introduction 13
Packing List 13
System Requirements 13
Unpacking and Inspection 13
Discharge Static Electricity 14
Jumper Settings 14
DAC Jumper Settings for Model A 15
DAC Jumper Settings for Model B 17
DAC Jumper Settings for Model C 19
DAC Jumper Settings for Model D 22
Local Bus Jumper Settings (JB1) 25
Setting the I/O Base Address 26
I/O Address Settings 26
Offset Address Settings 28
Board Installation 29
Software Installation 30
Installation Under Windows 30
Windows NT Kernel Mode Driver Installation 31
Installing Under DOS 32
Overview of GTPDO Files 32
Driver Files 33
Example Programs 34
Distributing the Software 36

Chapter 4 – Using the Virtual Panel 37

Introduction 37
Opening the Panel 37
Initializing 38
Connecting\Disconnecting the Voltage Rail Sources 40

Setting the Reference Voltages Level 40
Viewing the Current Channel Settings 40
Viewing the Rail/Reference Connections 40
Connect/Disconnecting Rails from Channels 41
Saving Rail Settings to the On-Board EEPROM 41
Resetting the Board 41
Closing the Virtual Panel 41

Chapter 5 – Programming the Board 43

Introduction 43
Windows 95/98/NT, Windows 3.1 and DOS Drivers 44
Import Libraries 44
Using the DLL Driver with Windows NT 45
How Does the HW Driver Work? 45
Using the Windows 32-bit DLL 46
Using the Windows 16-bit DLL 47
Programming with the DOS Static Library 48
Supported Development Tools 48
Programming with C/C++ Tools 48
Programming with Visual Basic 49
Programming with Borland Pascal/Delphi 49
Programming with ***ATEasy*** 2.0 49
Programming with the Driver 50
Initialization and the Board Handle 50
Board Handles 51
Reset 51
Error Handling 51
Driver Version 51
Virtual Panel 52
Programming Example 52

Compiling the Example 52
MAK Files 53
Example Program Listing 55

Chapter 6 – Functions Reference 59

Introduction 59
Parameter Prefix Names 60
Common Parameters 61
GtPdoGetBoardSummary 62
GtPdoGetChannelState 64
GtPdoGetChannelStateVoltageRail 66
GtPdoGetDACVoltage 68
GtPdoGetDriverSummary 69
GtPdoGetErrorString 71
GtPdoGetMeasureToVoltageRail 72
GtPdoGetModel 74
GtPdoGetVoltageRailSource 76
GtPdoInitialize 78
GtPdoPanel 80
GtPdoReset 82
GtPdoSaveChannelsVoltageRails 83
GtPdoSetChannelState 85
GtPdoSetChannelStateVoltageRail 87
GtPdoSetDacVoltage 89
GtPdoSetMeasureToVoltageRail 90
GtPdoSetVoltageRailSource 92

Appendix A – Error Codes 95

Resource Errors 95
Parameter Errors 96

Board Errors (Fatal Errors) 96
Miscellaneous Errors 97

Appendix B – Specifications 99

Index 101

Chapter 1 - Introduction

About this Manual

This GT7864 User's Guide provides all the information needed to install, program, and use Geotest's GT7864 board. This manual assumes that the user has a general knowledge of PC-based computers, Windows 3.1/95/98/NT operating systems and an electronics background. Some knowledge of programming and development tools will permit computer program control of the board.

Scope and Organization

This manual is organized as follows:

Chapter	Content
Chapter 1	Introduces this GT7864 User's Guide.
Chapter 2	Summarizes board features, architecture, hardware and software.
Chapter 3	Furnishes step-by-step directions for installing and setting up the hardware and GTPDO software.
Chapter 4	Provides instructions for using the Virtual Panel.
Chapter 5	Presents details on how to program the board by using the supplied driver and example files.
Chapter 6	Contains an alphabetical list of driver functions including syntax, variables, programming comments and samples.
Appendix A	A reference for GTPDO driver error codes.
Appendix B	Summarizes GT7864 specifications.
Index	A roadmap to important topics and concepts in this manual.

Conventions Used in This Manual

There are several naming conventions used throughout this manual. The conventions used are:

Example	Description
Copy or **Paste**	Commands are indicated in bold type.
Shift+F1	Keys are often used in combinations. The example to the left instructs the user to hold down the shift key while pressing the FI key at the same time. When key combination instructions are separated by commas, such as ALT+D, A, hold the ALT key while pressing D, then press A.
Direction Keys	Refer to the up arrow (↑), down arrow (↓), right arrow (→), and left arrow (←) keys.
cd bold	Bolded text must be entered from the keyboard exactly as shown.
cd *directory name*	Italicized text is a placeholder for variables or other items that the user must define and enter from the keyboard.
`examples`	Examples and source code are indicated in Courier, a fixed pitch font.
0x*hexnumber*	An integer in hexadecimal notation, E.g., 0x10A equals 266 in decimal.

Technical Support

General Information

Geotest provides both pre-sales and post-sales technical support for all products. Our Technical Support engineers can help you select hardware and software for your application, understand the specifications, and assist you in designing a complete system. Call our Technical Support department from 8:30AM to 5:30PM Pacific Standard Time (PST). Call 949-263-2222 or send email to support@geotestinc.com.

Internet Home Page Download Area

Use Geotest's Internet download area to obtain new instrument drivers, application notes, and other valuable resource information for the board. Our Internet address is: www.geotestinc.com

Calibration

Geotest offers calibration services for some measurement instrumentation. For more information, please contact Geotest Technical Support at 949-263-2222.

Geotest Address and Telephone Numbers

Geotest – Marvin Test Systems, Inc.
2851 S. Pullman Street
Santa Ana, CA 92705
Phone: 949-263-2222
Fax: 949-263-1203
Email (sales): sales@geotestinc.com
Email (support): support@geotestinc.com

Chapter 2 – Overview

Introduction

The GT7864 Programmable Discrete Output board is designed for avionics testing and other applications where multiple discrete outputs are required. The board occupies one GTXI slot and is available in four versions, A through D. Depending on model, the board contains 32 or 64 output channels, or output drives, and four voltage rails.

Models A and B contain 32 output channels, while models C and D contain 64 output channels. Each output channel can be set to a high, low, or open state. One of the four voltage rails is assigned to each channel for use with the high and low states. In models A and C, the output channels are assigned to voltage rails via jumper settings, while in models B and D the output channels are allocated via the software.

Features

Table 2-1 summarizes the features of the GT7864 models:

GT7864 Models	Output Channels	Channel Assignments to High/Low Voltage Rails
A	32	Hard-wired via jumpers
B	32	Programmable
C	64	Hard-wired via jumpers
D	64	Programmable

Table 2-1: GT7864 ModuleFeatures

The four voltage rails can derive power from either an on-board digital-to-analog converter (DAC) output or an external power source as illustrated in Figure 2-1.

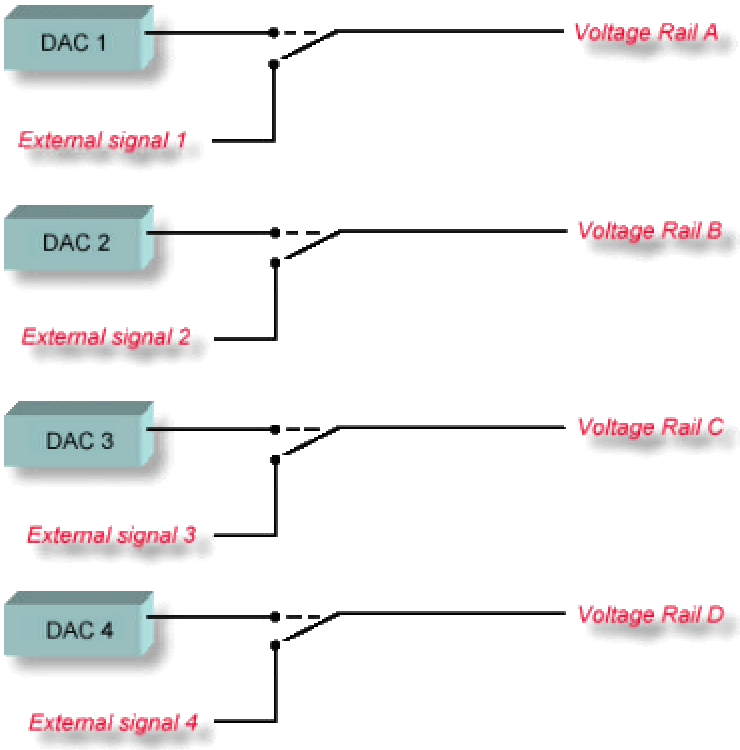

Figure 2-1: GT7864 Voltage Rails and Power Sources

The GT7864 voltage range is:

- DAC A: -10 to 32 volts (voltage rail A).
- DAC B: -10 to 32 volts (voltage rail B).
- DAC C: -5 to 5 volts (voltage rail C).
- DAC D: -0.5 to 0.5 volts (voltage rail D).

The GT7864 provides self-test capabilities using the MEAS channel (pin 58 on the GT7864's connector). The MEAS channel can be routed to any of the four voltage rails programmatically, thus allowing direct access to the internal DACs for calibration and BIT. The voltage of the DAC may be measured using an external instrument such as a DMM or an A/D converter. For details on the **GtPdoGetMeasureToVoltageRail** function, see Chapter 6, "Functions Reference."

The four DACs and four external sources are also routed to the GTXI's Local Bus connector. The local bus allows several GT7864s to share sources. For more information about the GT7864's connection to the GTXI local bus, see Chapter 3, “Setup and Installation.”

Board Diagram

An illustration of the board is shown below:

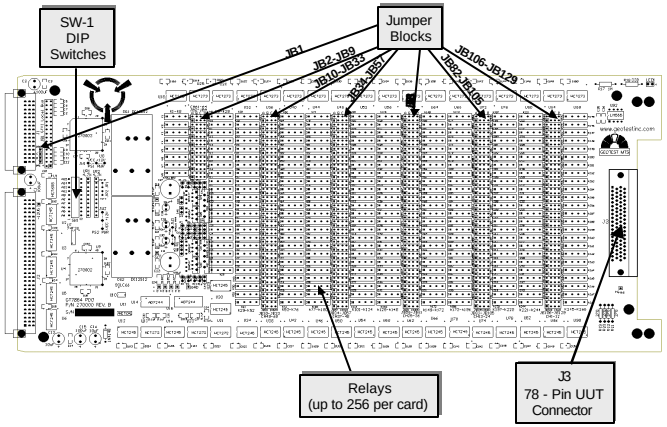

Figure 2-2: A GT7864 Programmable Discrete Output Board

Architecture

The figure below shows a block diagram. The GT7864 consists of a DAC section and two switching arrays, references and channels. The DAC section contains two dual buffered DACs. Each buffer can drive at least 100mA. Two DACs have an output range of –10V to 32V with a resolution of 3.906mV. The third DAC has an output range of –5V to 5V with a resolution of 0.6103mV. The fourth DAC has an output range of –0.5V to 0.5V with a resolution of 0.06103mV.

The Reference Switching array connects the internal reference lines to either the DACs or external lines. Each of the reference lines can also be switched to the 'MEAS' line for self-test purposes.

The Channel switching arrays contains 256 relays when fully populated (64 channels for model GT7864D) and can switch each channel to any one of the reference lines.

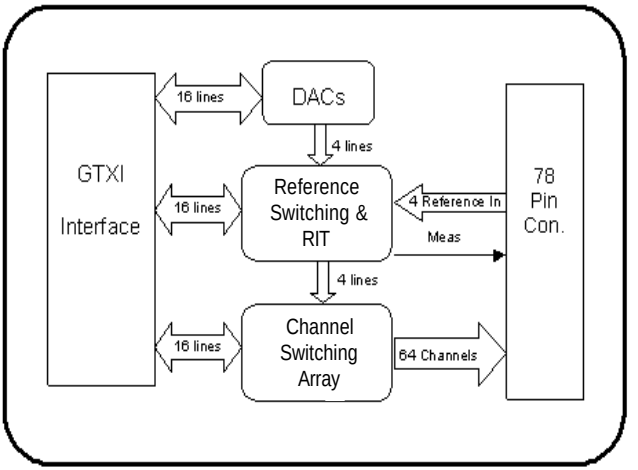

Figure 2-3: GT7864 Architecture

Connector

The following table describes the 78-pin connector for the UUT:

Pin #	Function	Pin #	Function	Pin #	Function	Pin #	Function
1	Channel 1	21	Channel 17	40	Channel 33	60	Channel 49
2	Channel 2	22	Channel 18	41	Channel 34	61	Channel 50
3	Channel 3	23	Channel 19	42	Channel 35	62	Channel 51
4	Channel 4	24	Channel 20	43	Channel 36	63	Channel 52
5	Channel 5	25	Channel 21	44	Channel 37	64	Channel 53

Pin #	Function	Pin #	Function	Pin #	Function	Pin #	Function
6	Channel 6	26	Channel 22	45	Channel 38	65	Channel 54
7	Channel 7	27	Channel 23	46	Channel 39	66	Channel 55
8	Channel 8	28	Channel 24	47	Channel 40	67	Channel 56
9	Channel 9	29	Channel 25	48	Channel 41	68	Channel 57
10	Channel 10	30	Channel 26	49	Channel 42	69	Channel 58
11	Channel 11	31	Channel 27	50	Channel 42	70	Channel 59
12	Channel 12	32	Channel 28	51	Channel 44	71	Channel 60
13	Channel 13	33	Channel 29	52	Channel 45	72	Channel 61
14	Channel 14	34	Channel 30	53	Channel 46	73	Channel 62
15	Channel 15	35	Channel 31	54	Channel 47	74	Channel 63
16	Channel 16	36	Channel 32	55	Channel 48	75	Channel 64
17	External Ref A	37	External Ref B	56	External Ref C	76	External Ref D
18	Sense 1	38	Sense 2	57	Sense 3	77	Sense 4
19	NC	39	GND	58	Meas	78	GND
20	GND			59	GND		

Table 2-2: GTPDO Connector J3 for the Unit Under Test (UUT)

The UUT board-side connector, J3, has 78 pins that are numbered as shown in Figure 2-4. A cable terminating in a mating connector interfaces with the UUT.

Figure 2-4: UUT Board Connector J3 (Component Side Up)

On-board EEPROM

The GT7864 has an on-board EEPROM that contains the DAC calibration parameters and the output channel-to-voltage rail assignment information.

Each DAC is calibrated for ±0.5 LSB with a 1KΩ load for the specific range. The calibration data is stored in the GT7864 onboard EEPROM where it is available to be used by the GT7864 software for setting the DAC voltage.

When GT7864 software assigns output channels to voltage rails, the configuration information is stored in the EEPROM. The program assigns output channels to voltage rails only with GT7864 models B and D. With models A and C output channels are assigned to voltage rails with jumper settings. However, even with models A and C the GT7864 software should be used to specify channel-rail assignments programmatically so that this information is stored in the on-board EEPROM. With channel-rail assignments stored in EEPROM for models A and C, EEPROM outputs documenting those assignments can be generated. Also, it is easier to upgrade to models B and D when channel-rail assignments are stored in the program.

Software

The GT7864 software module, also referred to as the GTPDO module, includes the following:

- Function Library
- Virtual Panel
- Programming Examples
- Windows Help File
- Installation and Setup utility.

Virtual Panel

The GTPDO DLL driver includes a front panel program that enables the user to fully utilize various configuration and control modes.

The Virtual Panel (Figure 2-5) interactively sets up and programs the GT7864 board. This functionality is provided as part of the DLL driver and as a stand-alone executable file.

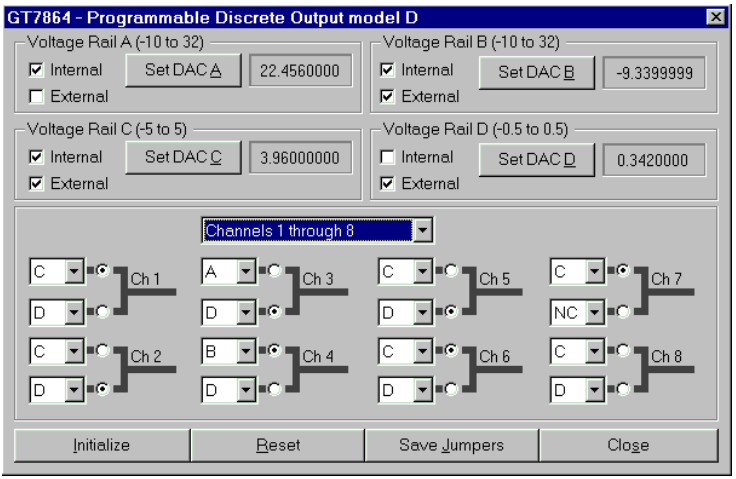

Figure 2-5: GT7864 Virtual Panel

Function Library

This part of the module contains functions that perform the following:

- Initialization of drivers.
- Resetting of the board, or parts of it.
- Setting, controlling or retrieving board parameters.

The functions are in various formats to support different operating systems and development tools. The drivers are supplied in two formats:

- Windows 16 and 32 bit DLLs. Used for Windows 3.x, 95, 98 and NT-based applications programming.
- DOS Static Libraries. Used for DOS-based applications programming.

Exported functions are declared in C (using Pascal calling conventions) to provide access from various development tools. Interface files and examples are provided for various development tools and programming languages, such as C/C++, VB, Pascal, Borland, and Geotest's ***ATEasy***.

Programming Example

The example in Chapter 5 demonstrates how to program the board with the supplied drivers. The example is provided in various formats to demonstrate use of different development tools and operating systems.

Complete discussions of the Virtual Panel, files, drivers, examples, and Function Library are contained in chapters Four, Five and Six, respectively.

Chapter 3 – Setup and Installation

Introduction

This product consists of both the board and the software module. This chapter describes how to install the GT7864 board and software module referred to as GTPDO.

Packing List

This package includes one each:

- GT7864 User's Guide
- GTPDO/GT7864 Driver Disk
- GT7864 Programmable Discrete Output board.

System Requirements

The GT7864 board is designed to run on an IBM compatible computer running DOS, Windows 3.X, Windows 95/98, or Windows NT, with at least one free 8/16-bit ISA bus slot.

Unpacking and Inspection

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. 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 this manual.

Discharge Static Electricity

To reduce the risk of damaging the GT7864, observe the following precautions:

- Leave the board in the anti-static bag until you install it. 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 the board or handle it roughly.
- Handle the board by the edges. Do not touch any components on the circuit board.

Caution: Turn off the GTXI power before attempting installation. Do not attempt to insert or remove any board with the GTXI power on.

Jumper Settings

This topic applies only to GT7864 Models A and C. Models B and D require no jumper configuration changes because relays select rail voltages automatically under program control for these models.

Use jumpers to connect each High and Low Channel to one of four Voltage Rails. A jumper on each header can be inserted to enable connections (see Figure 3-1 below and Jumper Setting details for Models A and C in the following pages.) If a jumper is omitted, an open connection results for that header.

Only one source at a time can be connected to the channel input. A program-controlled switch selects High or Low Channel headers, or Open as indicated in the schematic drawing below.

The following figure shows an example where Voltage Rail B is connected to High Channel and Voltage Rail D is connected to Low Channel. The switch selects the low header resulting in a connection to Voltage Rail Source D.

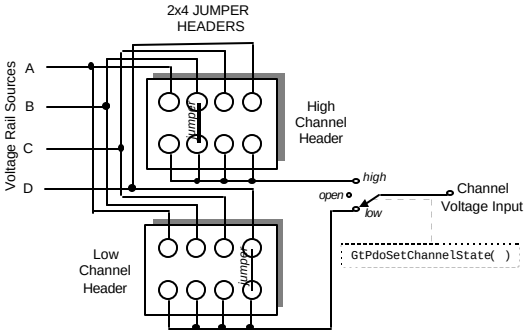

Figure 3-1: GT7864 Jumper Block and Switch Schematic

Caution: More than one jumper on a header can result in a short circuit that can cause errors and damage components.

DAC Jumper Settings for Model A

The jumpers for Model A must be set as shown in the table below:

Jumper Settings for Model A

Channel	Rail	JB #	Channel	Rail	JB #
1	High	109	17	High	101
1	Low	108	17	Low	100
2	High	113	18	High	105
2	Low	112	18	Low	104
3	High	117	19	High	83
3	Low	116	19	Low	82
4	High	121	20	High	87
4	Low	120	20	Low	86
5	High	125	21	High	91
5	Low	124	21	Low	90
6	High	129	22	High	95
6	Low	128	22	Low	94

Jumper Settings for Model A

Channel	Rail	JB #	Channel	Rail	JB #
7	High	107	23	High	99
7	Low	106	23	Low	98
8	High	111	24	High	103
8	Low	110	24	Low	102
9	High	115	25	High	61
9	Low	114	25	Low	60
10	High	119	26	High	65
10	Low	118	26	Low	64
11	High	123	27	High	69
11	Low	122	27	Low	68
12	High	127	28	High	73
12	Low	126	28	Low	72
13	High	85	29	High	77
13	Low	84	29	Low	76
14	High	89	30	High	81
14	Low	88	30	Low	80
15	High	93	31	High	59
15	Low	92	31	Low	58
16	High	97	32	High	63
16	Low	96	32	Low	62

Table 3-1: DAC Jumper Settings for Model A

DAC Jumper Settings for Model B

The jumpers for the GT7864 model B must be set as shown in the table below.

Jumper Settings for Model B							
Channel	**Rail**	**JB #**	**Jumper Setup**	**Channel**	**Rail**	**JB #**	**Jumper Setup**
1	High	109	D	17	High	101	D
1	Low	108	C	17	Low	100	C
2	High	113	D	18	High	105	D
2	Low	112	C	18	Low	104	C
3	High	117	D	19	High	83	D
3	Low	116	C	19	Low	82	C
4	High	121	D	20	High	87	D
4	Low	120	C	20	Low	86	C
5	High	125	D	21	High	91	D
5	Low	124	C	21	Low	90	C
6	High	129	D	22	High	95	D
6	Low	128	C	22	Low	94	C
7	High	107	D	23	High	99	D
7	Low	106	C	23	Low	98	C
8	High	111	D	24	High	103	D
8	Low	110	C	24	Low	102	C
9	High	115	D	25	High	61	D
9	Low	114	C	25	Low	60	C
10	High	119	D	26	High	65	D
10	Low	118	C	26	Low	64	C
11	High	123	D	27	High	69	D

Jumper Settings for Model B							
Channel	**Rail**	**JB #**	**Jumper Setup**	**Channel**	**Rail**	**JB #**	**Jumper Setup**
11	Low	122	C	27	Low	68	C
12	High	127	D	28	High	73	D
12	Low	126	C	28	Low	72	C
13	High	85	D	29	High	77	D
13	Low	84	C	29	Low	76	C
14	High	89	D	30	High	81	D
14	Low	88	C	30	Low	80	C
15	High	93	D	31	High	59	D
15	Low	92	C	31	Low	58	C
16	High	97	D	32	High	63	D
16	Low	96	C	32	Low	62	C

Table 3-2: DAC Jumper Settings for Model B

DAC Jumper Settings for Model C

The jumpers for the GT7864 model C must be set as shown in the table below.

Jumper Settings for Model C

Channel	Rail	JB #	Channel	Rail	JB #
1	High	109	33	High	67
1	Low	108	33	Low	66
2	High	113	34	High	71
2	Low	112	34	Low	70
3	High	117	35	High	75
3	Low	116	35	Low	74
4	High	121	36	High	79
4	Low	120	36	Low	78
5	High	125	37	High	37
5	Low	124	37	Low	36
6	High	129	38	High	41
6	Low	128	38	Low	40
7	High	107	39	High	45
7	Low	106	39	Low	44
8	High	111	40	High	49
8	Low	110	40	Low	48
9	High	115	41	High	53
9	Low	114	41	Low	52
10	High	119	42	High	57
10	Low	118	42	Low	56
11	High	123	43	High	35

Jumper Settings for Model C

Channel	Rail	JB #	Channel	Rail	JB #
11	Low	122	43	Low	34
12	High	127	44	High	39
12	Low	126	44	Low	38
13	High	85	45	High	43
13	Low	84	45	Low	42
14	High	89	46	High	47
14	Low	88	46	Low	46
15	High	93	47	High	51
15	Low	92	47	Low	50
16	High	97	48	High	55
16	Low	96	48	Low	54
17	High	101	49	High	13
17	Low	100	49	Low	12
18	High	105	50	High	17
18	Low	104	50	Low	16
19	High	83	51	High	21
19	Low	82	51	Low	20
20	High	87	52	High	25
20	Low	86	52	Low	24
21	High	91	53	High	29
21	Low	90	53	Low	28
22	High	95	54	High	33
22	Low	94	54	Low	32
23	High	99	55	High	11
23	Low	98	55	Low	10

Jumper Settings for Model C

Channel	Rail	JB #	Channel	Rail	JB #
24	High	103	56	High	15
24	Low	102	56	Low	14
25	High	61	57	High	19
25	Low	60	57	Low	18
26	High	65	58	High	23
26	Low	64	58	Low	22
27	High	69	59	High	27
27	Low	68	59	Low	26
28	High	73	60	High	31
28	Low	72	60	Low	30
29	High	77	61	High	5
29	Low	76	61	Low	4
30	High	81	62	High	9
30	Low	80	62	Low	8
31	High	59	63	High	3
31	Low	58	63	Low	2
32	High	63	64	High	7
32	Low	62	64	Low	6

Table 3-3: DAC Jumper Settings for Model C

DAC Jumper Settings for Model D

The jumpers for Model D must be set as shown in the table below.

Jumper Settings for Model D

Channel	Rail	JB #	Jumper Setup	Channel	Rail	JB #	Jumper Setup
1	High	109	D	33	High	67	D
1	Low	108	C	33	Low	66	C
2	High	113	D	34	High	71	D
2	Low	112	C	34	Low	70	C
3	High	117	D	35	High	75	D
3	Low	116	C	35	Low	74	C
4	High	121	D	36	High	79	D
4	Low	120	C	36	Low	78	C
5	High	125	D	37	High	37	D
5	Low	124	C	37	Low	36	C
6	High	129	D	38	High	41	D
6	Low	128	C	38	Low	40	C
7	High	107	D	39	High	45	D
7	Low	106	C	39	Low	44	C
8	High	111	D	40	High	49	D
8	Low	110	C	40	Low	48	C
9	High	115	D	41	High	53	D
9	Low	114	C	41	Low	52	C
10	High	119	D	42	High	57	D
10	Low	118	C	42	Low	56	C
11	High	123	D	43	High	35	D
11	Low	122	C	43	Low	34	C

Jumper Settings for Model D

Channel	Rail	JB #	Jumper Setup	Channel	Rail	JB #	Jumper Setup
12	High	127	D	44	High	39	D
12	Low	126	C	44	Low	38	C
13	High	85	D	45	High	43	D
13	Low	84	C	45	Low	42	C
14	High	89	D	46	High	47	D
14	Low	88	C	46	Low	46	C
15	High	93	D	47	High	51	D
15	Low	92	C	47	Low	50	C
16	High	97	D	48	High	55	D
16	Low	96	C	48	Low	54	C
17	High	101	D	49	High	13	D
17	Low	100	C	49	Low	12	C
18	High	105	D	50	High	17	D
18	Low	104	C	50	Low	16	C
19	High	83	D	51	High	21	D
19	Low	82	C	51	Low	20	C
20	High	87	D	52	High	25	D
20	Low	86	C	52	Low	24	C
21	High	91	D	53	High	29	D
21	Low	90	C	53	Low	28	C
22	High	95	D	54	High	33	D
22	Low	94	C	54	Low	32	C
23	High	99	D	55	High	11	D
23	Low	98	C	55	Low	10	C
24	High	103	D	56	High	15	D

Jumper Settings for Model D

Channel	Rail	JB #	Jumper Setup	Channel	Rail	JB #	Jumper Setup
24	Low	102	C	56	Low	14	C
25	High	61	D	57	High	19	D
25	Low	60	C	57	Low	18	C
26	High	65	D	58	High	23	D
26	Low	64	C	58	Low	22	C
27	High	69	D	59	High	27	D
27	Low	68	C	59	Low	26	C
28	High	73	D	60	High	31	D
28	Low	72	C	60	Low	30	C
29	High	77	D	61	High	5	D
29	Low	76	C	61	Low	4	C
30	High	81	D	62	High	9	D
30	Low	80	C	62	Low	8	C
31	High	59	D	63	High	3	D
31	Low	58	C	63	Low	2	C
32	High	63	D	64	High	7	D
32	Low	62	C	64	Low	6	C

Table 3-4: DAC Jumper Settings for Model D

Local Bus Jumper Settings (JB1)

Jumper Block JB1 can connect the four DACs and the four external references to the GTXI local bus. The local bus signals can measure the DACs and the references via another instrument with a local bus connection. These signals may be connected to another GT7864 in the same system. When the DAC lines of each GT7864 are crossed with the external reference lines of the other board, each of the two boards has eight programmable references. For more information on the Local Bus, please refer to the GTXI User's Guide.

Jumper	Pin #s	Function	Connects
1	1-2	GND	Not Used
2	3-4	DAC A	DAC A to Local Bus J1 B-21
3	5-6	DAC B	DAC B to Local Bus J1 B-22
4	7-8	DAC C	DAC C to Local Bus J1 B-23
5	9-10	DAC D	DAC D to Local Bus J1 B-24
6	11-12	Ext. Ref A	Ext. Ref A to Local Bus J1 B-21
7	13-14	Ext. Ref B	Ext. Ref B to Local Bus J1 B-22
8	15-16	Ext. Ref C	Ext. Ref C to Local Bus J1 B-23
9	17-18	Ext. Ref D	Ext. Ref D to Local Bus J1 B-24
10	19-20	DAC A	DAC A to Local Bus J1 B-25
11	21-22	DAC B	DAC B to Local Bus J1 B-26
12	23-34	DAC C	DAC C to Local Bus J1 B-27
13	25-26	DAC D	DAC D to Local Bus J1 B-28
14	27-28	Ext. Ref A	Ext. Ref A to Local Bus J1 B-25
15	29-30	Ext. Ref B	Ext. Ref B to Local Bus J1 B-26
16	31-32	Ext. Ref C	Ext. Ref C to Local Bus J1 B-27
17	33-34	Ext. Ref D	Ext. Ref D to Local Bus J1 B-28

Table 3-5: Local Bus Jumper Settings

Setting the I/O Base Address

The default I/O base address for all GTPDO boards, 0x1310, is determined by adding the I/O address (default 0x310) and the offset address (default 0x1000). The I/O base address can be changed by resetting the I/O Address, the offset address, or both, according to the values set forth in Tables 3-1 and 3-2.

The GT7864 uses 14 16-bit ports in the I/O space following the base address. Two additional ports are used at base address + 0x400, and nine at base address + 0x800. The factory-set (default) base address is 0x1310. On-board DIP switches can set the board's base address from 0x200 to 0xF3FF.

I/O Address Settings

Address (Hex)	SW1-10 A4	SW1-9 A5	SW1-8 A6	SW1-7 A7	SW1-6 A8	SW1-5 A9
200	ON	ON	ON	ON	OFF	ON
210	OFF	ON	ON	ON	OFF	ON
220	ON	OFF	ON	ON	OFF	ON
230	OFF	OFF	ON	ON	OFF	ON
240	ON	ON	OFF	ON	OFF	ON
250	OFF	ON	OFF	ON	OFF	ON
260	ON	OFF	OFF	ON	OFF	ON
270	OFF	OFF	OFF	ON	OFF	ON
280	ON	ON	ON	OFF	OFF	ON
290	OFF	ON	ON	OFF	OFF	ON
2A0	ON	OFF	ON	OFF	OFF	ON
2B0	OFF	OFF	ON	OFF	OFF	ON
2C0	ON	ON	OFF	OFF	OFF	ON
2D0	OFF	ON	OFF	OFF	OFF	ON

Address (Hex)	SW1-10 A4	SW1-9 A5	SW1-8 A6	SW1-7 A7	SW1-6 A8	SW1-5 A9
2E0	ON	OFF	OFF	OFF	OFF	ON
2F0	OFF	OFF	OFF	OFF	OFF	ON
300	ON	ON	ON	ON	OFF	OFF
310	OFF	ON	ON	ON	OFF	OFF
320	ON	OFF	ON	ON	OFF	OFF
330	OFF	OFF	ON	ON	OFF	OFF
340	ON	ON	OFF	ON	OFF	OFF
350	OFF	ON	OFF	ON	OFF	OFF
360	ON	OFF	OFF	ON	OFF	OFF
370	OFF	OFF	OFF	ON	OFF	OFF
380	ON	ON	ON	OFF	OFF	OFF
390	OFF	ON	ON	OFF	OFF	OFF
3A0	ON	OFF	ON	OFF	OFF	OFF
3B0	OFF	OFF	ON	OFF	OFF	OFF
3C0	ON	ON	OFF	OFF	OFF	OFF
3D0	OFF	ON	OFF	OFF	OFF	OFF
3E0	ON	OFF	OFF	OFF	OFF	OFF
3F0	OFF	OFF	OFF	OFF	OFF	OFF

Table 3-6: I/O Address Chart

Note: The default I/O address for all GTPDO boards is 0x310.

Offset Address Settings

Offset (Hex)	SW1-4 A12	SW1-3 A13	SW1-2 A14	SW1-1 A15
1000	OFF	ON	ON	ON
2000	ON	OFF	ON	ON
3000	OFF	OFF	ON	ON
4000	ON	ON	OFF	ON
5000	OFF	ON	OFF	ON
6000	ON	OFF	OFF	ON
7000	OFF	OFF	OFF	ON
8000	ON	ON	ON	OFF
9000	OFF	ON	ON	OFF
A000	ON	OFF	ON	OFF
B000	OFF	OFF	ON	OFF
C000	ON	ON	OFF	OFF
D000	OFF	ON	OFF	OFF
E000	ON	OFF	OFF	OFF
F000	OFF	OFF	OFF	OFF

Table 3-7: Offset Address Chart

Note: The default offset address for all GTPDO boards is 0x1000.

Board Installation

Warning: Turn off the GTXI before attempting installation. Do not insert or remove any board while the computer is turned on.

Until it is ready for installation, keep the board in the anti-static bag with which it is shipped and do not break the seal. This greatly reduces the risk of damage from static electricity.

Install the GT7864 board as follows:

1. Power off the GTXI chassis.
2. If necessary, (i.e., to change the default settings), set the I/O Base Address and Offset jumpers. All jumper and I/O base address settings are found in this chapter.
3. Allocate a GTXI slot to the GT7864. If a blank panel blocks the vacant slot, remove this panel.
4. Carefully remove the GT7864 board from the anti-static bag. Save the anti-static bag for future storage if the board is ever removed from the system.

Caution: Handle the board by its edges. Do not touch any components on the circuit board.

5. Carefully insert the GT7864 board into the GTXI slot until it mates with the backplane DIN connector and the front panel is flush with other GTXI panels.
6. Tighten the top and bottom mounting screws.
7. Connect the mating cable to the GT7864.
8. Turn on the GTXI.

Software Installation

Installing the GTPDO module is straightforward and easy. The software can be installed under Windows (3.x, '95, '98 or NT), or under DOS. Both environments are described in the following sections.

Installation Under Windows

1. Boot the computer and log into windows as Administrator (or with administrator rights) where applicable.

2. Insert the 3.5-inch diskette into the floppy drive.

3. In Windows 95, 98 or NT click **Start** and select **Run**.

   Under Windows NT, the Setup program installs the kernel mode driver, HW.SYS. To successfully install the kernel mode driver, you must be logged on as a user with administrator privileges. Note that the kernel mode driver can also be installed manually after the Setup. Please refer to the section "Windows NT Kernel Mode Driver Installation" on page 31.

4. In Windows 3.x, select **File**, **Run** from Program Manager.

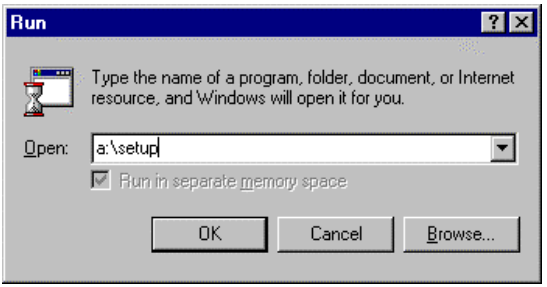

Figure 3-2. Run Window

5. In the command line field, enter A:\setup.exe. The Installation program copies the necessary files to the local hard disk. The default directory (folder) is C:\GTPDO.

Windows NT Kernel Mode Driver Installation

Under Windows NT, the kernel mode device driver, HW.SYS, must be installed and executed before the GTPDO driver can be used. Before installing the kernel mode driver, you must be logged on as a user with an administrator privilege.

The GTPDO Setup program normally installs the kernel mode driver and starts it automatically. However, in the case when the current user is not logged in as an administrator, the kernel mode driver installation will fail. This section explains how to install the kernel mode driver manually when the Setup program fails to do so.

To manually install the kernel mode driver, perform the following:

1. Log in with administrator privileges.

2. Open a Command Prompt window.

3. Change to the installation destination directory by using the CD command (e.g. **CD \GTPDO**).

4. From the command prompt, execute the following command:

   HWSETUP –vdd install start

   If the current working directory is different from the directory where HW.SYS resides, you may specify your own custom path. For Example:

   HWSETUP –vdd install=a:\start

The Setup program installs the driver as a service. The service can be started or stopped from the Windows NT control panel Devices applet. The **–vdd** switch can be removed from the command if support for 16-bit drivers is not required (only the 32-bit DLL will be used in this case).

The Setup program, HWSETUP.EXE, the device driver HW.SYS, the driver library, HWVDD.DLL, and the GTPDO driver files may be distributed with an application for the GT7864 board.

Additional HWSETUP.EXE command line options are available. To display these options, type **HWSETUP** with no command line options.

Installing Under DOS

The Setup program that is shipped with the GT7864 software module disk can only be used from Windows. To install the GTPDO module under DOS, manually copy the files in Table 3-8 to the local hard drive.

Filename	Purpose
README.1ST	Text file that includes latest information and changes.
GTPDO16.BAS	Used with Visual Basic for DOS (with minor changes).
GTPDO16 .PAS	Used with Turbo Pascal (with minor changes).
GTPDO.H	C header file for GT7864 driver functions and constants.
GTPDOBC.LIB	Borland static library for DOS applications.
GTPDOMS.LIB	Microsoft static library for DOS applications.

Table 3-8. Files Installed Under DOS

Overview of GTPDO Files

The Setup program installs the GTPDO software module which includes:

- Driver Files
- Example Programs.

A complete discussion of how to use the files can be found in "Chapter 5 - Programming the Board."

Driver Files

Tables 3-9 through 3-12 describe the files included in the software.

Filename	Purpose
GTPDOBC.LIB	Borland static library for DOS applications
GTPDOMS.LIB	Microsoft static library for DOS applications
GTPDO16.DLL	16-bit DLL for Windows 3.x applications
CTL3DV2.DLL	Supports 3D Controls/Dialogs for GTPDO16.DLL
GTPDO16.LIB	16-bit import library for GTPDO16.DLL
GTPDO32.DLL	32-bit DLL for Windows 95/NT applications
GTPDO32.LIB	32-bit import library for GTPDO32.DLL
GTPDO32B.LIB	32-bit import library for GTPDO32.DLL for Borland applications

Table 3-9. GTPDO Library Files

Filename	Purpose
GTPDOP16.EXE	16-bit virtual panel for GTPDO boards
GTPDOP32.EXE	32-bit virtual panel for GTPDO boards

Table 3-10. GTPDO Virtual Panel Executables

Filename	Purpose
HW.SYS	Windows NT Kernel mode driver for the GTPDO32.DLL
HWVDD.DLL	Windows NT virtual device driver for use with the 16-bit DLL under Windows NT
HWSETUP.EXE	Setup program to install/uninstall the Windows NT kernel driver HW.SYS
HWTEST.EXE	Verification program to test the HW.SYS driver installation

Table 3-11. GTPDO NT Kernel Mode Device Drivers

Filename	Purpose
GTPDO.H	C header file for driver functions and constants
GTPDO16.BAS	Visual Basic (16 bit) file that contains 16-bit DLL function and constant declarations
GTPDO32.BAS	Visual Basic (32 bit) file that contains 16-bit DLL function and constant declarations
GTPDO.INS	***ATEasy*** 2.x driver for GT7864 board
GTPDO.PAS	Driver for use with Borland Pascal or Delphi

Table 3-12. GTPDO Interface Files

Example Programs

The GTPDO software module includes a C source file that is used for all samples, and a MAK file (IDE file for Borland C++) for each of the following development tools:

- DOS Static Library (16- and 32-bit DLLs)
- Borland Static Library (16- and 32-bit DLLs)
- ***ATEasy***

The actual samples demonstrate how the resulting output module (executable file) uses the appropriate library and MAK file. Tables 3-13 through 3-17 explain the sample files in the software module.

File Name	Description
GTPDOX32.EXE	Output module (executable)
GTPDOX32.MAK	VC++/MS-C Make file
GTPDOX.C	C source file
GTPDOX.RC	Resource file
GTPDOX.ICO	Icon file
GTPDO32.DLL	32-Bit DLL

Table 3-13. Files Used for Windows 95/NT Sample

File Name	Description
GTPDOX16.EXE	Output module (executable)
GTPDOX16.MAK	VC++/MS-C Make file
GTPDOX.C	C source file
GTPDOX.DEF	Module definition file
GTPDOX.RC	Resource file
GTPDOX.ICO	Icon file
GTPDO16.DLL	16-Bit DLL

Table 3-14. Files Used for Windows 3.x/95 Sample

File Name	Description
GTPDOXMS.EXE	Output module (executable)
GTPDOXMS.MAK	VC++/MS-C Make file
GTPDOX.C	C source file (same source for all samples)
GTPDOMS.LIB	DOS Static Library

Table 3-15. Files for DOS Sample (with Microsoft C Static Library)

File Name	Description
GTPDOBC.IDE	Borland C++ 5.0 contains 3 projects: Windows 32, Windows 16 and DOS.
GTPDOX.C	C source file
GTPDOX.DEF	Module definition file
GTPDOX.RC	Resource file
GTPDOX.ICO	Icon file
GTPDOBC.LIB	Borland Static Library

Table 3-16. Files Used for Borland Sample

File Name	Description
GTPDO.PRG	GTPDO example program
GTPDO.CFG	GTPDO.PRG example CFG file
GTPDO.INS	Driver file for ***ATEasy*** version 2.x

Table 3-17. Files Used for *ATEasy* Sample

Figure 3-3 below shows the resulting window when the output module (with the exception of GTPDOXMS.EXE) is executed.

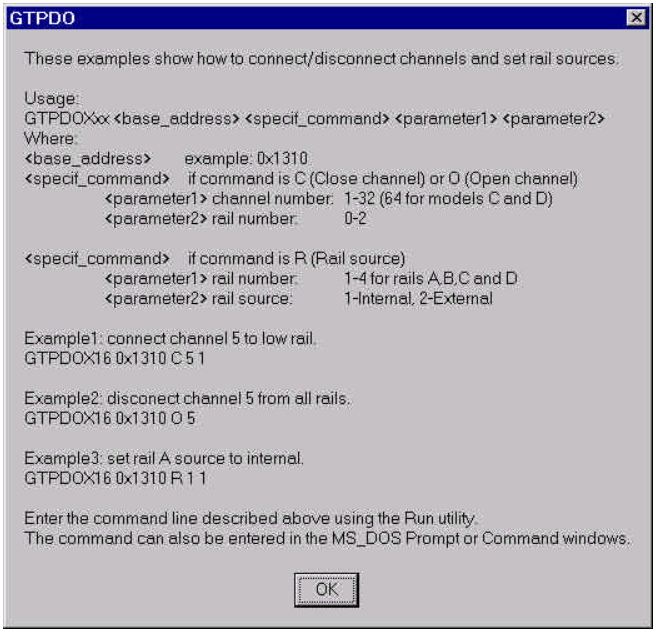

Figure 3-3. GTPDO Sample

Distributing the Software

The setup program HWSETUP.EXE, the device drivers HW.SYS and HWVDD.DLL, and the GTPDO driver files may be distributed with any application that uses the GT7864 board.

Chapter 4 – Using the Virtual Panel

Introduction

This chapter provides the basics for using the GT7864 Board and the Virtual Instrument Panel. We assume the user has carefully read "Chapter 2 – Overview," which provides important information on the theory of operation and function for both the GT7864 Board and the GTPDO software module.

Other procedures, such as programming the board and using supplied drivers and functions, are discussed in Chapters Five and Six of this manual.

Opening the Panel

The GTPDO software module includes a DLL for each supported operating system. Depending on the operating system installed on the PC, the Virtual Panel is executed differently.

In Windows 95/NT, the 32-bit Virtual Panel is executed by clicking the **Start** button on the taskbar, then highlighting **Programs**, **GTPDO**, and clicking on **GTPDO Panel – 32 bit**.

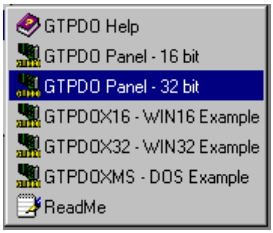

Figure 4-1: Windows 95/98/NT Menu via Start Button

In Windows 3.x, run the 16-bit Virtual Panel by clicking the icon below from the GTPDO group window in Program Manager:

Figure 4-2: GT7864 Icon for Windows 3.1

The 16-bit Virtual Panel will also run under Windows 95/98/NT and may be accessed using the **Start** button, selecting **Programs and GTPDO,** and clicking **GTPDO Panel – 16 bit**.

Initializing

After GTPDOP16.EXE or GTPDOP32.EXE is executed, the Virtual Panel appears with all options disabled, as shown below.

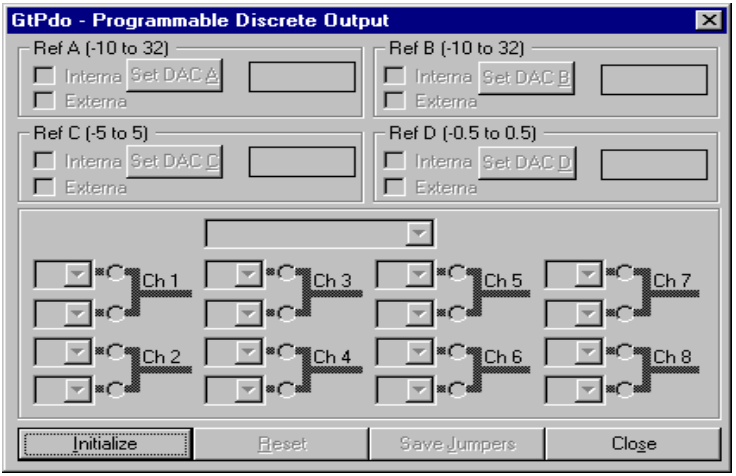

Figure 4-3: GT7864 Virtual Panel with Options Disabled

Before any of the options can be selected, the GT7864 board must be initialized. To do so, click the **Initialize…** button.

Figure 4-4: Initialize Button

When the Initialize button has been clicked, the following dialog box appears:

Figure 4-5: Changing the Base Address (See Caution)

Type the Base Address that is set on the board and click **OK**. Note that initializing the board has no effect on the board setting.

Caution: The Base Address (default 1310h) is set prior to installing the board. If the Base Address conflicts with other system hardware, change it. Refer to the section "I/O Address Settings" in Chapter 3 for appropriate switch settings for other I/O addresses.

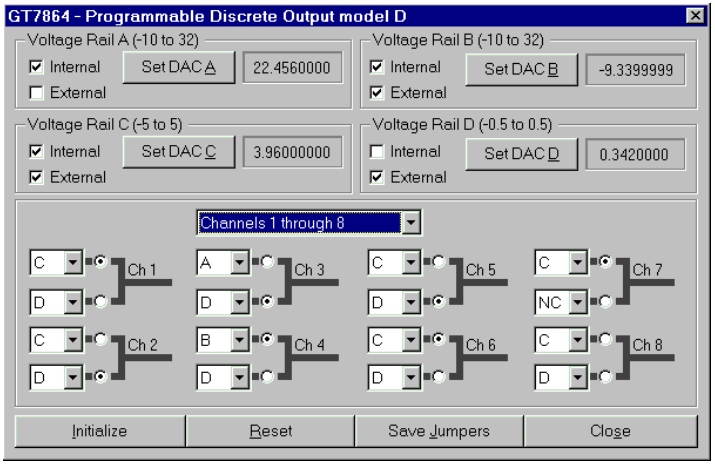

Figure 4-6: GT7864 Virtual Panel with Options Enabled

When the GT7864 board has been successfully initialized, the Virtual Panel appears with all functions and values enabled as displayed in Figure 4-6.

Connecting\Disconnecting the Voltage Rail Sources

Click the check box for the voltage to connect or disconnect. When a box is checked, the voltage is connected. When a box is unchecked, the voltage is disconnected.

The connections can be to the internal DAC reference source or to the external source coming from the 78-pin connector. If both check boxes are checked, the internal DAC reference source is connected internally and to the external 78-pin connector as well.

Setting the Reference Voltages Level

1. Select the **Set DAC** button.
2. Enter the desired voltage for the specific DAC and click OK.
3. If the voltage value exceeds the DAC range, an error message appears. You can change the value or select **Cancel** to close the voltage window without changing the voltage.

Viewing the Current Channel Settings

From the drop-down list box select the channel group you want to view. The channel reference connection or rail setting is immediately updated.

Viewing the Rail/Reference Connections

1. To see the channel rail settings as they were last written to the EEPROM, check the **View Rails** checkbox.
2. To show the current connections to the DAC reference source for each channel, uncheck the check box.

Note: This check box is visible in models B and D in order for those models to be compatible with models A and C.

Connect/Disconnecting Rails from Channels

Click on the radio buttons.

Each channel has two reference/rail sources. The upper one is the high rail and the lower is the low rail.

Saving Rail Settings to the On-Board EEPROM

Click **Save Jumpers**.

Resetting the Board

Click **Reset**.

Resetting the GT7864 disconnects all voltage rails from sources, sets all DACs to 0 volts, and sets all channels to an open state. Panel parameters are set to the default state.

Figure 4-7: Reset Button

Closing the Virtual Panel

Click **Close**.

Figure 4-8: Close Button

Chapter 5 – Programming the Board

Introduction

This chapter contains information on how to program the GT7864 board using the GTPDO software package. The GTPDO drivers contain functions to initialize, reset, and control the GT7864 board. A brief description of the functions, as well as how and when to use them, is included in this chapter. Chapter Six describes these functions in detail.

This chapter also describes how the GTPDO driver supports different operating systems and development tools. It also includes an example showing how to program the GT7864 board with the C programming language. Since the driver functions and parameters are identical for all operating systems and development tools, the example can serve as an outline for other programming languages, programming tools, and other GTPDO driver formats. The following development tools are supported:

- Visual C++
- Visual Basic
- DOS (Microsoft or Borland C)
- Borland C++
- ***ATEasy***

Refer to the README.1ST file on the installation disk for the latest list of programming languages and development tools that are supported, as well as other example files that are available on the disk.

Windows 95/98/NT, Windows 3.1 and DOS Drivers

The GTPDO driver file comes in three main formats

- 32-Bit MS-Windows DLL (Dynamic Link Library) – GTPDO32.DLL - Used by WIN32 application running under Windows 95, 98 and Windows NT.
- 16-Bit MS-Windows DLL - GTPDO16.DLL - Used by 16-bit Windows application running under Windows 3.x/95/98/NT.
- DOS static libraries – GTPDOMS.LIB (Microsoft format) and GTPDOBC.LIB (Borland format) - Used by DOS applications that are compiled with Microsoft or Borland development tools. The libraries are compiled in large memory model libraries with floating point emulation.

Import Libraries

The following files should also be used during development with specific applications, as follows:

- GTPDO16.LIB is the import library for GTPDO16.DLL.
- GTPDO32.LIB is the import library for GTPDO32.DLL.
- The GTPDO32B.LIB is the import library for Borland applications.

Note: All three import libraries mentioned above must be linked into the Windows applications just like any other static library.

The GTPDO.H header file contains prototypes of the GTPDO driver functions. It must be included at the beginning of any C/C++ file that makes GTPDO function calls.

Using the DLL Driver with Windows NT

Under NT, direct access to hardware (e.g. I/O ports) is limited to code that runs in kernel mode. Windows applications (16 or 32-bit) cannot access the hardware because they run in user mode. To access hardware, these applications rely on drivers that reside in kernel mode. The GTPDO package has built-in support that overcomes some of the limitations of Windows applications.

HW.SYS (kernel-mode device driver) and HWVDD.DLL (virtual device driver) are both installed with the GTPDO module. HW.SYS enables any 16-bit or 32-bit Windows application to access the I/O ports. The HWVDD.DLL driver traps any I/O-to-PC port from a Windows 16-bit application and routes these calls to HW.SYS, which then executes the command.

Note: When under NT, the HW drivers will trap any I/O port in the range of 0x100-0xFFFE excluding the range 0x3BB-0x3BF.

How Does the HW Driver Work?

Figure 5-1 depicts how HW.SYS works.

The following steps explain Figure 5-1:

1. Assume a 16-bit Windows application attempts to write to an I/O port.
2. NT traps the I/O port write attempt and routes it to HWVDD.DLL.
3. HWVDD.DLL routes the call to HW.SYS.
4. HW.SYS executes the port I/O (write) instruction and returns.

A similar process is executed when reading from a port.

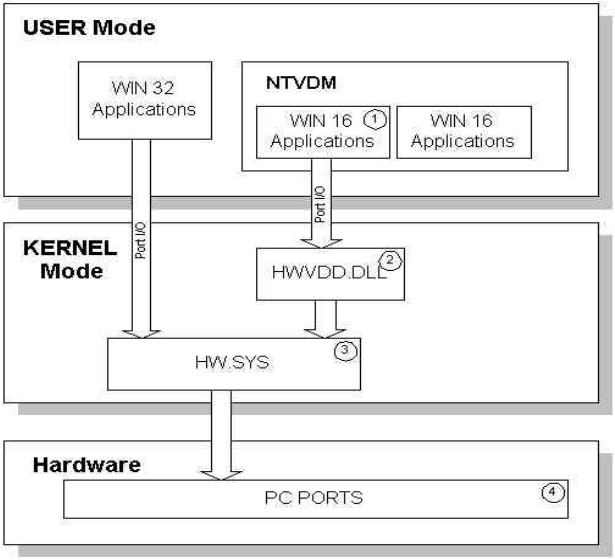

Figure 5-1. How the Hardware Driver works

Using the Windows 32-bit DLL

The Windows 32-bit GTPDO32.DLL can be used for applications that are developed for Windows 95, 98 or NT. Refer to Chapter 3, "Setup and Installation" for special installation instructions for Windows NT. The 32-bit DLL can be used with various development tools such as:

- Microsoft Visual C++
- Borland C
- ***ATEasy***
- Microsoft Visual Basic
- Borland Pascal
- Borland Delphi

The DLL may be shipped with the GTPDO application. Refer to one of the sections beginning on page 48 for specific information about the development tools listed above.

To use the DLL driver from an application, the DLL must reside in one of the following directory types:

- Application (e.g. \Program Files)
- Windows (e.g. \Windows or \WinNT)
- Windows System (e.g. \Windows\System or WinNT\System32)
- One of the directories specified in the PATH statement

The GTPDO32.DLL file may be distributed with the GT7864 board and any associated applications.

Note: Under Windows NT HWVDD.DLL and HW.SYS must be installed before using the 32-bit DLL. Refer to Microsoft's Windows NT Kernel Mode Driver Installation procedure.

Using the Windows 16-bit DLL

The Windows 16-bit DLL, GTPDO16.DLL, is used for writing 16-bit applications that run under Windows 3.1, 95, 98 or NT. The DLL can be used with various development tools such as:

- Microsoft Visual C++ (1.5 or earlier)
- Borland C (for 16-bit)
- ***ATEasy*** 2.0
- Microsoft Visual Basic (3.0 or earlier)
- Borland Pascal
- Borland Delphi (for 16-bit)

The DLL may be shipped with the developed application. Refer to one of the sections beginning on page 48 for specific information about one of the development tools listed above.

To use the DLL driver from an application, the DLL must reside in one of the following directories:

- Application
- Windows
- Windows System

- One of the directories specified in the PATH statement

The GTPDO16.DLL file may be distributed with the GT7864 board and any associated applications.

Note: Under Windows NT, HW.SYS must be installed before the 16-bit DLL can be used. See "NT Kernel Mode Driver Installation procedure" on page 31.

Programming with the DOS Static Library

When programming the board under DOS, the application must be linked with either GTPDOBC.LIB (Borland static library) or GTPDOMS.LIB (Microsoft static library). These files are static libraries in which functions reside. The Setup disk contains static libraries for many development tools. See the README.1ST file for a complete list of supported development tools.

Supported Development Tools

The GTPDO driver supports C/C++, Visual Basic, Borland Pascal/Delphi, Geotest ***ATEasy***, and any other language that can use a Windows DLL, such as LabView.

Programming with C/C++ Tools

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

- Include the GTPDO.H header file in the C/C++ source file that uses the GTPDO function. This header file is used for all driver formats. The file contains function prototypes and constant declarations to be used by the compiler for the application.
- For Windows applications, make sure the DLL is installed in the proper directory (see previous sections beginning on page 44 that describe how to use the DLL).
- Add the required .LIB file to the project. This can be the import library GTPDO32.LIB for 32-bit application, GTPDO32BC.LIB for 32-bit applications that uses Borland

C++, GTPDO16.LIB for 16-bit application, or any of the DOS static libraries for DOS application. Windows based applications that explicitly load the DLL by calling the Windows **LoadLibrary()** API need not include the .LIB file in the project.

- Add code to call the GTPDO as required by the application.
- Build the project.
- Run, test, and debug the application.

Programming with Visual Basic

GTPDO16.BAS and GTPDO32.BAS files contain function declarations for the GTPDO16.DLL and GTPDO32.DLL drivers, respectively. These BAS files must be loaded using **Load File** from the Visual Basic File menu before the functions can be used.

Programming with Borland Pascal/Delphi

To use the driver with Borland Pascal or Delphi, include GTPDO.PAS in the project. The GTPDO.PAS file contains a unit with function prototypes for all the 16- and 32-bit DLLs. Include the GTPDO unit in the **USES** statement before making calls to the GTPDO functions. .

Programming with *ATEasy* 2.0

The ***ATEasy*** driver (GTPDO.INS) supplied with this product uses the GTPDO16.DLL to program the board. These files must reside in the ***ATEasy*** INS and DLL directories, respectively. To use the driver, the INS, CFG, PRG and 16-bit DLL files must be copied to the appropriate ***ATEasy*** directory. The GTPDO.INS driver must also be included in the current system configuration file (.CFG) in order to program the board.

GTPDO.INS is also supplied with an example that contains a program and a system file configured with the instrument driver set to the default base address (in the System Editor Driver Setup dialog - module number field).

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. This parameter is handled automatically by the driver. (***ATEasy*** uses driver logical names).
- The *nStatus* parameter is omitted. Use the Get Status commands instead to check status.

The ***ATEasy*** driver also contains additional commands to permit easier access to the GT7864 board features (instead of typing parameter values). The commands are self-documenting. Their syntax is similar to English sentences and can be generated from the program editor in the Instruments menu. GTPDO.INS is documented in the ***ATEasy*** Notes dialog, which can be viewed from the Driver Editor Summary contained within the Edit menu. In addition, more detailed documentation exists in the ***ATEasy*** knowledge base, ***ATEASY***KB.HLP, that is shipped with ***ATEasy***.

The GTPDO Setup program updates the ***ATEasy*** directory with the new DLL-based driver and example files.

Programming with the Driver

The GTPDO driver contains a set of functions that initialize the board driver, reset the board, and display the instrument virtual panel. In addition, the GTPDO driver uses handles (see below) to access the GT7864 board and process errors. The following paragraphs describe the steps and topics that are required to control the board.

Initialization and the Board Handle

The **GtPdoInitialize** function initializes the driver for the board at the specified base address. The function returns a handle that can be used later with other functions to program the board. This handle is usually saved in the program as a global variable for later use when calling other functions. The initialize function does not change the state of the board.

Board Handles

The Board handle parameter - *nHandle* - is a short integer number that is used by the GTPDO driver functions to identify the board being accessed by the application. The *nHandle* parameter is required to identify the board being programmed.

nHandle is created when the application calls the **GtPdoInitialize** function. There is no need, or command, to destroy the handle. Calling **GtPdoInitialize** with the same base address will return the same handle.

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

Reset

The Reset function, **GtPdoReset**, opens all switches in the board and sets the board to a known state. A reset is usually performed after the board is initialized. See Chapter 6 in this manual for more information regarding the board reset.

Error Handling

All the GTPDO functions return the status, *pnStatus*, as the last parameter. This parameter can be used for error handling. *pnStatus* returns zero if the function completes successfully. It returns negative if an error occurred. The error description can be retrieved using the **GtPdoGetErrorString** function.

Driver Version

The **GtPdoGetDriverSummary** function can be used to return the current GTPDO driver description summary and version number. This can be used to check for driver changes. See Chapter 6 in this manual for more information.

Virtual Panel

The panel function, **GtPdoPanel**, displays the instrument front panel in a window. The panel can be used to initialize the board and to control and read all board settings. The panel function may be used by the application to allow direct interaction with the board.

The **GtPdoPanel** function is also used by GTPDOP16.EXE and GTPDOP32.EXE, which are supplied with the GTPDO package and provide a stand alone Windows application that displays the instrument panel. This function is available only under the Windows version of the driver and is not supported under the DOS driver.

Programming Example

The following example demonstrates how to program the board using the C programming language under Windows and DOS. To run, enter the following on the command line:

GTPDOXxx <*base_address*> <*command*><*channel*><*ref*>

Where:

<*base_address*>	I/O Base Address in hexadecimal. For example: 0x1310
<*command*>	Can be C (connect) or O (open)
<*channel*>	Channel 1 – 32 (1 – 64 for models C and D)
<*ref*>	Reference 0 - 6

Compiling the Example

To compile the Windows 32-bit DLL example in Microsoft Visual C++ 2.0 or above:

1. Load GTPDOX32.MAK from the Project / File/Open menu.

2. Select Project/Rebuild all from the menu.

To compile the Windows 32-bit DLL example in Borland C++ 4.0/4.5/5.0:

1. Load GTPDOXBC.IDE from the Project/Open Project menu.

2. Select Project/Build all from the menu.

To compile the Windows 16-bit DLL example in Microsoft VC++ 1.0/1.5 or above:

1. Load GTPDOX16.MAK from the Project/Open menu.

2. Select Project/Rebuild all from the menu.

To compile the Windows 16-bit DLL example in Microsoft C 5.1 or above using the command line compiler, run NMAKE GTPDOX16.MAK /A from the DOS command line.

To compile the Windows 16-bit DLL example in Borland C++ 4.0/4.5/5.0:

1. Load GTPDOXBC.IDE from the Project/Open Project command.

2. Select Project/Build all.

To compile the DOS example in Microsoft VC++:

1. Load GTPDOXMS.MAK from the Project/Open command.

2. Select Project/Rebuild all.

To compile the DOS example in Microsoft C 5.1 or above, run NMAKE GTPDOXMS.MAK /A from the DOS command line.

To compile the DOS example in Borland C++ 4.0/4.5/5.0:

1. Load GTPDOXBC.IDE from the Project/Open Project command.

2. Select Project/Build all.

MAK Files

The C++ sample includes a MAK file for each of the following tools:

DOS Static Library - Microsoft C (VC++/MS-C) Example File:

- GTPDOXMS.EXE - Sample for DOS (with Microsoft C Static Library)
- GTPDOXMS.MAK - VC++/MS-C Make file for the example
- GTPDOX.C - C source file for the example (the same source file is used for all C examples)

16-bit DLL (Windows 3.x) - Microsoft C (VC++/MS-C) Example Files:

- GTPDOX16.EXE - Sample for Windows 3.x/95/98
- GTPDOX16.MAK - VC++/MS-C Make file for the example
- GTPDOX.C - C source file for the example (same source for all C examples)
- GTPDOX.DEF - Module definition file
- GTPDOX.RC - Resource file
- GTPDOX.ICO - Icon file

32-bit DLL (Windows 95/NT) example files:

- GTPDOX32.EXE - Sample for Windows NT/95/98
- GTPDOX32.MAK - VC++/MS-C Make file for the example
- GTPDOX.C - C source file for the example (same source for all C examples)
- GTPDOX.RC - Resource file
- GTPDOX.ICO - Icon file

Borland Static/16-bit DLL/32-bit DLL Examples:

- GTPDOBC.IDE - Borland C++ 5.0 contains 3 projects
- GTPDOX.C - C source file for the example (same source for all C examples)
- GTPDOX.DEF - Module definition file
- GTPDOX.RC - Resource file
- GTPDOX.ICO - Icon file

ATEasy Examples Files

- GTPDO.PRG - example program
- GTPDO.CFG - example CFG file

Example Program Listing

```
/*********************************************************
     FILE     : GTPDOX.C
     PURPOSE: Windows16\32/DOS sample program for
               GT7864 using GT7864 drivers.
     CREATED: Dec. 1997
     COPYRIGHT: Copyright 1996 GEOTEST Inc.
     COMMENTS:

    To compile the Windows 32-bit DLL example:
    1. Microsoft VC++ 2.0 or above
       - Load GTPDOX32.MAK from the Project\File\Open... menu
      - Select Project/Rebuild all from the menu
    2. Borland C++ 4.0/4.5/5.0
       - Load GTPDOXBC.IDE from the Project/Open
         Project... menu
       - Select Project/Build all from the menu

    To compile the Windows 16-bit DLL example:
    1. Microsoft VC++ 1.0/1.5 or above
       - Load GTPDOX16.MAK from the Project/Open... menu
       - Select Project/Rebuild all from the menu
    2. Microsoft C 5.1 or above using command line compiler
       - Run 'NMAKE GTPDOX16.MAK /A' from DOS command line
    3. Borland C++ 4.0/4.5/5.0
       - Load GTPDOXBC.IDE from the Project/Open
         Project... menu
       - Select Project/Build all from the menu

    To compile the DOS example with:
    1. Microsoft VC++
       - Load GTPDOXMS.MAK from the Project/Open... menu
       - Select Project/Rebuild all from the menu
    2. Microsoft C 5.1 or above
       - Run 'NMAKE GTPDOXMS.MAK /A' from DOS command line
    3. Borland C++ 4.0/4.5/5.0
       - Load GTPDOXBC.IDE from the Project/Open
         Project menu
       - Select Project/Build all from the menu
*********************************************************/

#if defined(_WINDOWS) || defined(_WIN32) || defined (_Windows)
    #include <windows.h>
#elsif !defined(_DOS)
    #define _DOS
#endif
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include "GtPdo.h"

//*********************************************************
//      DispMsg
//*********************************************************
void DispMsg(LPSTR szMsg)
{
    #ifndef _DOS
    //if windows use MessageBox
        MessageBeep(0);
        MessageBox(0, szMsg,"GT7864", MB_OK);
```

```
    #else
    // use printf for DOS
        printf("%s \r\n", szMsg);
    #endif
    return;
}

//*********************************************************
//      DispUsage
//*********************************************************
void DispUsage(void)
{
    DispMsg(
        "\r\nThis example shows how to "
        "Connect or disconnect a channel\r\n"
        "Usage:\r\n"
        "GT7864Xxxx <base_address> <specif_command>"
        "\r\n\r\nWhere:\r\n"
        "<base_address>"
        "\t(example : 0x310)\r\n"
        "<specif_command> depends on the model:\r\n"
        "\t C | O Channel# Ref#\r\n"
        "\t\tWhere: C=Close, O=Open, Channel#=1-32(64 for C, D)\r\n"
        "\t\tRef#=0-6\r\n"
        "Example1 connecting channel to refernce:\r\n"
        "GtPdoX16 0x310 C 5 1\r\n"
"Connect channel number 5 to reference 1 (LoRail) using 0x310 as the base
    address\r\n"

"Example2 disconnecting channel from all references:\r\n"
        "GtPdoX16 0x310 O 5 \r\n"
"Connect channel number 5 to reference 1 (LoRail) using 0x310 as the base
    address\r\n"

"Example3 setting reference source A to internal:\r\n"
        "GtPdoX16 0x310 R 1 1\r\n"
"Connect channel number 5 to reference 1 (LoRail) using 0x310 as the base
    address\r\n"

    #ifdef _WINDOWS
        "\r\nTo change command line under Windows \r\n"
        "Select File/Properties from Program Manager \r\n"
        "Menu and change the command line edit box as \r\n"
        "shown above."
    #else

    #endif
        );
    exit(1);
}

//*********************************************************
//      CheckStatus
//*********************************************************
void CheckStatus(SHORT nStatus)
{
    CHAR    sz[128];

    if (!nStatus) return;
    GtPdoGetErrorString(nStatus, sz, sizeof(sz), &nStatus);
    DispMsg(sz);
    DispMsg("Aborting the program...");
    exit(nStatus);
```

```
}
//*********************************************************
//       Main
//   This main function receives four parameters:
//  GT7864 base address (e.g. 0x310)
//  GT7864 switching operation (e.g. O=Open channel, C=Connect channel)
//
//  either GT7864 A, B channels # (1-32)
//  or GT7864 C, D channels # (1-64)
//   and
//  GT7864 connections values 0-6
//
//*********************************************************
int main(int argc, char **argv)
{
    short   nBaseAddr;              // Board base address
    char    cOperation;             // Board Operation
    short   nChannel;               // Channel number
    short   nRailOrReference;       // Reference or Rail number
    short   nRefSource;             // Reference source
    short   nHandle;                // Boar]d handle
    short   nStatus;                // Returned status

    // ** Check number of arguments rcvd
    if (argc < 4 || argc > 5) DispUsage();

    // ** Parse command line parameters
    nBaseAddr=(SHORT)strtol(*(++argv), NULL, 0);
    cOperation=**(++argv);

    // ** Process operation (connect/open)
    cOperation=(char)toupper(cOperation);
    if (cOperation != 'C' && cOperation != 'O' && cOperation != 'R')
        DispUsage();

    // ** Get channel and nRailOrReference
    GtPdoInitialize (nBaseAddr, &nHandle, &nStatus);
    CheckStatus(nStatus);

    switch(cOperation)
    {
         case 'C':
              nChannel = (SHORT)strtol(*(++argv), NULL, 0);
              nRailOrReference = (SHORT)strtol(*(++argv), NULL, 0);
              GtPdoChannelConnect(nHandle, nChannel, nRailOrReference,
              &nStatus);
              CheckStatus(nStatus);
              DispMsg ("You have connected a channel to Reference");
         break;
         case 'O':
              nChannel = (SHORT)strtol(*(++argv), NULL, 0);
              nRailOrReference = (SHORT)strtol(*(++argv), NULL, 0);
              GtPdoChannelConnect(nHandle, nChannel, 0, &nStatus);
              CheckStatus(nStatus);
         DispMsg ("You have disconnected a channel from any reference");
         break;
         case 'R':
              nRailOrReference = (SHORT)strtol(*(++argv), NULL, 0);
              nRefSource = (SHORT)strtol(*(++argv), NULL, 0);
              GtPdoSetRefSource(nHandle, nRailOrReference,
              nRefSource, &nStatus);
              CheckStatus(nStatus);
              DispMsg ("You have set the reference source");
```

```
        break;
        default:
        break;
     }
    return 0;
}

//*********************************************************
//                   End Of File
//*********************************************************
```

Chapter 6 – Functions Reference

Introduction

This chapter presents information about the GTPDO driver functions in alphabetical order. Each function begins with a syntax example of the function followed by a short description of the function parameters and types, Comments, Example (written in C), and See Also sections.

All function parameters follow the same rules:

- All pointers are far pointers (applicable only to 16-bit driver).
- Strings are ASCIIZ (null or zero character terminated).
- The first parameter for most functions is *nHandle* (16-bit integer), which is required and returned by the board-specific **GtPdoInitialize** function. The *nHandle* parameter identifies the board when calling a function to program and control the board.
- All functions return a status with the last parameter, named *pnStatus*. *pnStatus* is 0 if the function was successful, or < 0 on error. The description of the error can be retrieved by using the **GtPdoGetErrorString** function, or by using a predefined constant, defined in the driver interface file. See Appendix A for error codes.

Parameter Prefix Names

Parameter names use prefixes that follow these conventions:

Type	Prefix	Type Description	Example
ARRAY	a	Prefix precedes simple type.	anArray
BOOL	b	Boolean. Signed 8-bit integer. TRUE if 0, FALSE otherwise.	bSelected
BYTE	uc	Unsigned 8-bit integer (unsigned CHAR type).	ucSection
CHAR	c	Character. Signed 8-bit integer.	cKeyPress
Double	d	Double precision 64-bit floating point.	dReading
DWORD	dw	Unsigned 32-bit integer (non-handle).	dwTimeout
DWORD	hwnd	Window handle - unsigned 32-bit integer used as a handle. Only the lower 16-bits are used under Windows 3.1.	hwndPanel
LONG	l	Signed 32-bit integer.	lBits
LP	p	32-bit long pointer. (Far pointer for Windows 3.1). Usually returns a value. Prefix precedes simple type.	pnStatus
LPSTR	sz	String. Zero-terminated ASCIIZ text string.	szMsg
SHORT	n	Signed 16-bit integer.	nMode
WORD	w	Unsigned 16-bit integer.	wCount

Table 6-1: Parameter Name Prefixes

Common Parameters

The parameters listed in the table below are commonly used by the GT7864 functions to configure and control the board. For more information about the GT7864 components that these parameters reference, see Chapter 2, "Overview."

Parameter	Description	Values
nChannel	The GT7864 output channel. Models A and B have 32 channels, while models C and D have 64 channels.	1-32 for A, B 1-64 for C, D
nRail	The GT7864 has four voltage rails, A through D. Each of the four rails has two voltage sources, internal (DAC) and external.	0 disconnects voltage rail from measurement pin. 1 – 4 for voltage rails A – D.
nDAC	The DACs are used as internal voltage sources for the four voltage rails. The GT7864 voltage range is: DAC A: -10 to 32 volts (voltage rail A). DAC B: -10 to 32 volts (voltage rail B). DAC C: -5 to 5 volts (voltage rail C). DAC D: -0.5 to 0.5 volts (voltage rail D).	1 – DAC A 2 – DAC B 3 – DAC C 4 – DAC D
pnHiSrc, *pnLoSrc*	Each channel can be connected to a high or low source, or be open. For each channel, the user has to define the source to be used for the high and low states. Available sources are the four voltage rails.	1 – 4 for voltage rails A – D.

GtPdoGetBoardSummary

Purpose

Returns the board summary fromthe on-board EEPROM.

Syntax

GtPdoGetBoardSummary(*nHandle, szBoardSum, nSumMaxLen, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle to a GT7864.
szBoardSum	LPSTR	Buffer to contain the returned board info (null terminated) string.
nSumMaxLen	SHORT	Size of the buffer to contain the error string.
pnStatus	LPSHORT	Returned status: 0 on success, <0 on failure.

Comments

The GT7864 summary string provides the following data from the on-board EEPROM in the order shown:

- Instrument Name (e.g., GT7864C)
- EEPROM format version (e.g., 1.00)
- PCB revision (e.g., 'A')
- Serial Number (e.g., 7864008)
- Calibration time and date

For example, the returned string could look like the following:

```
GT7864A, Version 1.00, Revision A, S/N 7864008, last calibrated
time Mon Dec 08 14:39:02 1997
```

Example

The following example returns the board summary:

```
SHORT nHandle, nStatus;
CHAR szBoardSum[256]

GtPdoGetBoardSummary(nHandle, szBoardSum,
    sizeof(szBoardSummary), &nStatus)
```

See Also

GtPdoGetDriverSummary, GtPdoInitialize, GtPdoGetErrorString

GtPdoGetChannelState

Purpose

Returns the state of the specified channel.

Syntax

GtPdoGetChannelState(*nHandle, nChannel, pnHiLoState, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle to a GT7864.
nChannel	SHORT	Channel number: 1-32 – versions A and B 1-64 – versions C and D
pnHiLoState	LPSHORT	Returned values are: 0—Open state. Not connected. 1—Low state. 2—High state.
pnStatus	LPSHORT	Returned status: 0 on success, <0 on failure.

Comments

The GT7864 has four different versions, A through D. Models A and B contain 32 output channels, while models C and D contain 64 output channels. With models A and C, channel states are assigned to voltage rails by using jumpers. With models B and D, channel states are assigned by the software.

In versions B and D, selecting a channel state voltage to open affects the **GtPdoGetChannelState** return value as follows:

Channel State Voltage Rail Source Setting	**Channel State Setting**	**Return State**
High state voltage rail source – open	High	Open
Low state voltage rail source – open	Low	Open
High and low state voltage rail sources – open	High/Low	Open
High and low state voltage rail sources are equal	High/Low	Low
High and low state voltage rail sources are not equal	High/Low	High

Example

The following example returns the current connection of channel 1:

```
SHORT nStatus, nHiLoState;

GtPdoGetChannelState(nHandle, 1, &nHiLoState, &nStatus);
```

See Also

GtPdoGetDacVoltage, **GtPdoSetChannelStateVoltageRail**, **GtPdoGetChannelStateVoltageRail**, **GtPdoInitialize,** **GtPdoGetErrorString**

GtPdoGetChannelStateVoltageRail

Purpose

Returns the specified channel's high and low state voltage rail assignment.

Syntax

GtPdoGetChannelStateVoltageRail(*nHandle, nChannel, pnLoStateRail, pnHiStateRail, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle to a GT7864.
nChannel	SHORT	Channel number: 1-32 – versions A and B 1-64 – versions C and D
pnLoStateRail	LPSHORT	Returned channel's low state voltage rail source as follows: 1 – Voltage rail A 2 – Voltage rail B 3 – Voltage rail C 4 – Voltage rail D
pnHiStateRail	LPSHORT	Returned channel's high state voltage rail source as follows: 1 – Voltage rail A 2 – Voltage rail B 3 – Voltage rail C 4 – Voltage rail D
pnStatus	LPSHORT	Returned status: 0 on success, <0 on failure.

Comments

In versions A and C the returned values are read from the EEPROM as defined by the **GtPdoSetChannelStateVoltageRail** command. In versions B and D the returned values are read from the actual connection on the board.

Example

The following example returns the channel 1 voltage rail setting:

```
SHORT nStatus nLoStateRail, nHiStateRail;

GtPdoGetChannelStateVoltageRail(nHandle, 1,
    &nLoStateRail, &nHiStateRail, nStatus);
```

See Also

GtPdoSetChannelState VoltageRail, **GtPdoSetChannelState**, **GtPdoGetChannelState, GtPdoInitialize, GtPdoGetErrorString**

GtPdoGetDACVoltage

Purpose

Returns the specified internal DAC voltage.

Syntax

GtPdoGetDACVoltage(*nHandle, nDAC, pdVoltage, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle to a GT7864.
nDAC	SHORT	Specifies the DAC number from which to get the voltage: 1 – DAC A 2 – DAC B 3 – DAC C 4 – DAC D
pdVoltage	LPDOUBLE	Returned voltage.
pnStatus	LPSHORT	Returned status: 0 on success, <0 on failure.

Example

The following example returns the voltage from DAC A:

```
SHORT nStatus;
DOUBLE dVoltage;

GtPdoGetDacVoltage(nHandle, GTPDO_DAC_A, &dVoltage,
    &nStatus);
```

See Also

GtPdoSetDACVoltage, **GtPdoInitialize, GtPdoGetErrorString**

GtPdoGetDriverSummary

Purpose

Returns the driver description string and version number.

Syntax

GtPdoGetDriverSummary(*szSummary, nSummaryMaxLen, pdwVersion, pnStatus*)

Parameters

Name	Type	Comments
szSummary	LPSTR	Buffer to receive the summary string.
nSummaryMaxLen	SHORT	Buffer size passed by szSummary.
pdwVersion	LPDWORD	Returned version number. The high 16 bits contain the major version number, while the low 16 bits contain the minor version number.
pnStatus	LPSHORT	Returned status: 0 on success, <0 on failure.

Comments

The GT7864 driver summary string provides the following information:

- Instrument driver name e.g. GtPdo
- Version number. E.g. Version 1.0

For example the returned string could look like this:

“GtPdo Driver for GTPDO, Version 1.0, Copyright (c) Geotest Inc.”

The value of *pdwVersion* looks like: 10, i.e. 1.0.

Example

The following example returns the driver summary:

```
SHORT nHandle, nStatus;
DWORD dwVersion;
CHAR  szSummary[256];

GtPdoGetDriverSummary(szSummary, sizeof(szSummary),
    &dwVersion, &nStatus);
```

See Also

GtPdoGetBoardSummary, GtPdoInitialize, GtPdoGetErrorString

GtPdoGetErrorString

Purpose

Returns the error string as specified by the error number.

Syntax

GtPdoGetErrorString(*nError*, *szError*, *nErrorMaxLen*, *pnStatus*)

Parameters

Name	Type	Comments
nError	SHORT	Error number as returned by the pnStatus of any function. (See Appendix A, “Error Codes,” for a list of error codes). For errors, the number should be less than zero. If not zero, the function returns the "No error has occurred" string.
szError	LPSTR	Buffer to contain the returned error (null terminated) string.
nErrorMaxLen	SHORT	Size of the buffer to contain the error string.
pnStatus	LPSHORT	Returned status: 0 on success, <0 on failu re.

Example

The following example initializes the board at addresses 0x310 and gets the error string:

```
SHORT nStatus;
CHAR  szError[128];

GtPdoInitialize (0x1310, &nHandle, &nStatus);
GtPdoGetErrorString (nStatus, &szError, sizeof(szError),
    &nStatus);
```

GtPdoGetMeasureToVoltageRail

Purpose

Returns the specified voltage rail currently connected to the output measurement pin. Only one voltage rail can be connected at any time.

Syntax

GtPdoGetMeasureToVoltageRail(*nHandle*, *pnVoltageRail*, *pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle to a GT7864.
pnVoltageRail	LPSHORT	Returned value specifies the voltage rail number that is currently connected: 0 – Not connected 1 – Voltage rail A 2 – Voltage rail B 3 – Voltage rail C 4 – Voltage rail D
pnStatus	LPSHORT	Returned status: 0 on success, <0 on failure.

Example

The following example returns the current reference that is connected to the measurement pin:

```
SHORT nStatus, nVoltageRail;

GtPdoGetMeasureToVoltageRail(nHandle, &nVoltageRail,
    &nStatus);
```

See Also

GtPdoGetVoltageRailSource, **GtPdoGetDacVoltage**, **GtPdoSetMeasureToVoltageRail**, **GtPdoInitialize,** **GtPdoGetErrorString**

GtPdoGetModel

Purpose

Returns the specified board model number.

Syntax

GtPdoGetModel(*nHandle, pnModel, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle to a GT7864.
pnModel	LPSHORT	Returned model number
		1 – model A
		2 – model B
		3 – model C
		4 – model D
pnStatus	LPSHORT	Returned status: 0 on success, <0 on failure.

Comments

The GT7864 has four different versions, A through D. Models A and B contain 32 output channels, while models C and D contain 64 output channels. With models A and C, you assign channel states to voltage rails using jumpers. With models B and D, you assign channel states to voltage rails using software. For details, see Chapter 2, "Introduction."

Example

The following example returns the model number:

```
SHORT nStatus, nModel;

GtPdoGetModel (nHandle, &nModel, &nStatus);
```

See Also

GtPdoGetBoardSummary, GtPdoInitialize, GtPdoGetErrorString

GtPdoGetVoltageRailSource

Purpose

Returns the specified voltage rail source.

Syntax

GtPdoGetVoltageRailSource(*nHandle, nVoltageRail, pnSource, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle to a GT7864.
nVoltageRail	SHORT	1 – Voltage rail A
		2 – Voltage rail B
		3 – Voltage rail C
		4 – Voltage rail D
pnSource	LPSHORT	Returned source for the specified voltage rail.
		0 - Disconnected from all sources.
		1 - Connected to the internal DAC.
		2 - Connected to the external input.
		3 - The internal DAC is connected to the external input and to the voltage rail.
pnStatus	LPSHORT	Returned status: 0 on success, <0 on failure.

Comments

Each of the four voltage rails has an internal (DAC) and external voltage source. Voltage rail A, for example, can be connected to

DAC A, to external input A, or to both the internal and external sources (thus routing the internal DAC voltage to the external pin).

Example

The following example returns the current source for voltage rail A.

```
SHORT nStatus, nSource;

GtPdoGetVoltageRailSource(nHandle, GTPDO_DAC_A,
    &nSource, &nStatus);
```

See Also

GtPdoGetDacVoltage, **GtPdoSetVoltageRailSource**, **GtPdoInitialize, GtPdoGetErrorString**

GtPdoInitialize

Purpose

Initializes the driver for the board at the specified base address. The function returns a handle that can be used with other GTPDO functions to program the board.

Syntax

GtPdoInitialize(*nBaseAddress, pnHandle, pnStatus*)

Parameters

Name	Type	Comments
nBaseAddress	Short	Base address of the board.
pnHandle	LPSHORT	Returned handle for the board. The handle is set to zero on error and <> 0 on success.
pnStatus	LPSHORT	Returned status: 0 on success, <0 on failure.

Comments

The GT7864 uses 14 16-bit ports in the I/O space following the base address. Two additional ports are used at base address + 0x400, and nine at base address + 0x800. The factory default base address is 0x1310. Using a DIP switch, you can change the board base address to any address between 0x200 and 0xF3FF. Refer to Chapter 3, "Setup and Installation," for more information.

The **GtPdoInitialize** function verifies that the board does or does not exist. However, the function does not have any effect on board settings.

The returned handle, *pnHandle,* is used by other GTPDO functions to identify the specified board.

The following example initializes the GT7864 at addresses 0x1310.

```
SHORT nHandle, nStatus;

GtPdoInitialize (0x1310, &nHandle, &nStatus);
if (nHandle==0)
{
    printf("Unable to Initialize the board");
    return;
}
```

See Also

GtPdoGetBoardSummary, GtPdoReset, GtPdoGetErrorString

GtPdoPanel

Purpose

Opens a virtual panel used to interactively control the GT7864 board.

Syntax

GtPdoPanel(*pnHandle, hwndParent, nMode, phwndPanel, pnStatus*)

Parameters

Name	Type	Comments
pnHandle	LPSHORT	Handle to a GT7864. This number may be zero if the board is to be initialized by the panel window.
hwndParent	DWORD	Sets the panel parent window handle. A value of 0 sets the desktop as the parent window.
nMode	SHORT	The mode in which the panel main window is created. 0 - Modeless window. 1 - Modal window.
phwndPanel	LPDWORD	Returned window handle for the panel (for modeless panels only).
pnStatus	LPSHORT	Returned status: 0 on success, <0 on failure.

Comments

This function is used to create the panel window. The panel window may be open as a modal or a modeless window depending on the *nMode* parameters.

If the mode is set to modal dialog (*nMod*e=1), the panel disables the parent window (*hwndParent*) and the function returns only after the user closes the window. In that case, the pnHandle may return the handle created by the panel Initialize dialog.

If a modeless dialog was created (nMode=0), the function returns immediately after creating the panel window and returns the window handle to the panel – *phwndPanel*. It is the responsibility of the calling program to dispatch window messages to this window, so that the window can respond to messages.

This function is supplied only with DLL versions of the driver.

Example

The following example opens the panel in modal mode:

```
DWORD   hwndPanel;
SHORT   nHandle=0, nStatus;

GtPdoInitialize (0x1310, &nHandle, &nStatus);
GTPdoPanel(&nHandle, 0, GTPDO_PANEL_MODAL, &hwndPanel,
    &nStatus);
```

See Also

GtPdoInitialize, GtPdoGetErrorString

GtPdoReset

Purpose

Sets all channels to an open state, disconnects all the voltage rails, and sets all the DAC voltages to zero.

Syntax

GtPdoReset(*nHandle, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle to a GT7864.
pnStatus	LPSHORT	Returned status: 0 on success, <0 on failure.

Example

The following example resets the board:

```
SHORT nHandle, nStatus;

GtPdoInitialize(0x1310, &nHandle, &nStatus);
GtPdoReset(nHandle, &nStatus);
```

See Also

GtPdoInitialize, GtPdoGetErrorString

GtPdoSaveChannelsVoltageRails

Purpose

Saves the channel-voltage rail high/low settings to EEPROM.

Syntax

GtPdoSaveChannelsVoltageRails(*nHandle, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle to a GT7864.
pnStatus	LPSHORT	Returned status: 0 on success, <0 on failure.

Comments

If you are using GT7864 models B or D, use this function to save the channel-rail assignments that you specified with the **GtPdoSetChannelStateVoltageRail** function so that they are restored when you reboot.

If you are using GT7864 models A or C, use this function to save your jumper settings so that the GT7864 software panel correctly reflects your channel-rail configuration. If you record your jumper settings with the **GtPdoSetChannelStateVoltageRail** function and save the settings to EEPROM, you will find it easier to upgrade to model B or D.

The **GtPdoSaveChannelsVoltageRails** function conserves write cycles by writing only settings that have changed since the last save.

Example

The following example saves the current channel-rail settings to EEPROM:

```
SHORT nStatus;

GtPdoSaveChannelsVoltageRails(nHandle, &nStatus);
```

See Also

GtPdoSetChannelVoltageRail, **GtPdoGetChannelVoltageRail**, **GtPdoGetBoardSummary, GtPdoInitialize, GtPdoGetErrorString**

GtPdoSetChannelState

Purpose

Sets the specified Channel to high, low or open states.

Syntax

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

Parameters

Name	Type	Comments
nHandle	SHORT	Handle to a GT7864.
nChannel	SHORT	Channel number: 1 - 32—Versions A and B 1 - 64—Versions C and D
nState	SHORT	Channel states are: 0 - Open. Not connected. 1 - Set to low state. 2 - Set to high state.
pnStatus	LPSHORT	Returned status: 0 on success, less than 0 on failure.

Comments

The GT7864 has four different versions, A through D. Models A and B contain 32 output channels, while models C and D contain 64 output channels. With models A and C, you assign channel states to voltage rails using jumpers. With models B and D you assign channel states to voltage rails using software.

Example

The following example connects channel 1 to the low source:

```
SHORT nHandle, nStatus;

GtPdoSetChannelState(nHandle, 1, GTPDO_CHANNEL_STATE_LO,
    &nStatus);
```

See Also

GtPdoGetChannelVoltageRail, **GtPdoGetChannelState**, **GtPdoGetVoltageRailSource**, **GtPdoGetDACVoltage, GtPdoInitialize, GtPdoGetErrorString**

GtPdoSetChannelStateVoltageRail

Purpose

Specifies the high and low voltage rails for a given channel.

Syntax

GtPdoSetChannelStateVoltageRail(*nHandle, nChannel, nLoStateRail, nHiStateRail, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle to a GT7864.
nChannel	SHORT	Channel number: 1-32 – versions A and B 1-64 – versions C and D
nLoStateRail	SHORT	A channel's low state voltage rail connection as follows: 1 – Voltage rail A 2 – Voltage rail B 3 – Voltage rail C 4 – Voltage rail D
nHiStateRail	SHORT	A channel's high state voltage rail connection as follows: 1 – Voltage rail A 2 – Voltage rail B 3 – Voltage rail C 4 – Voltage rail D
pnStatus	LPSHORT	Returned status: 0 on success, <0 on failure.

Comments

If you are using GT7864 model B or D, you must use this function to assign high and low voltage rails to output channels. You should also use the **GtPdoSaveChannelsVoltageRailsState** function to save your settings to EEPROM so that they are restored when you reboot.

If you are using GT7864 model A or C, this function has no effect on the board's operation since the assignments are set by jumpers. However, even with models A and C you should use the **GtPdoSetChannelVoltageRail** function to assign the high and low voltage rail sources for each output channel and then save these settings using the **GtPdoSaveChannelsVoltageRailsState** function so that the GT7864 virtual panel reflects the current settings. Following this procedure will also make it easier for you to upgrade to GT7864 model B or D.

Note: In models B and D, if the low state and high state voltage rail connections are equal, the function makes the setting but *pnStatus* returns a warning.

Example

The following example sets the low to voltage rail A and the high to voltage rail B:

```
SHORT nStatus;

GtPdoSetChannelVoltageRail(nHandle, 1, 1, 2, &nStatus);
```

See Also

GtPdoGetChannelStateVoltageRail, **GtPdoGetChannelState**, **GtPdoGetBoardSummary**, **GtPdoSaveChannelsVoltageRails, GtPdoInitialize, GtPdoGetErrorString**

GtPdoSetDacVoltage

Purpose

Sets the specified DAC to the specified voltage.

Syntax

GtPdoSetDacVoltage(*nHandle, nDAC, dVoltage, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle to a GT7864.
nDAC	SHORT	DAC channel to set: 1 – DAC A 2 – DAC B 3 – DAC C 4 – DAC D
dVoltage	DOUBLE	Desired voltage range.
pnStatus	LPSHORT	Returned status: 0 on success, <0 on failure.

Example

The following example sets DAC A voltage to 12.34:

```
SHORT nStatus;

GtPdoSetDACVoltage(nHandle, GTPDO_DAC_A, 12.34,
    &nStatus);
```

See Also

GtPdoGetDacVoltage, **GtPdoGetVoltageRailSource**, **GtPdoGetBoardSummary, GtPdoInitialize, GtPdoGetErrorString**

GtPdoSetMeasureToVoltageRail

Purpose

Sets the specified voltage rail to the output measurement pin.

Syntax

GtPdoSetMeasureToVoltageRail(*nHandle, nVoltageRail, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle to a GT7864.
nVoltageRail	SHORT	Specifies the voltage rail to be connected to the output measurement pin: 0—Disconnect any voltage rail from the measurement output pin. 1—Voltage rail A 2—Voltage rail B 3—Voltage rail C 4—Voltage rail D
pnStatus	LPSHORT	Returned status: 0 on success, <0 on failure.

Comments

The output measurement pin can be connected at any time to one of the rails, allowing direct access to the internal DACs for calibration and BIT. The output measurement pin is located at pin 58 on the 78-pin connector.

Example

The following example connects voltage rail A to the output measurement pin:

```
SHORT nStatus;

GtPdoSetMeasureToVoltageRail(nHandle,
    GTPDO_MEASRAIL_DACA, &nStatus);
```

See Also

GtPdoGetVoltageRailSource, **GtPdoGetDacVoltage**, **GtPdoGetMeasureToVoltageRail, GtPdoInitialize, GtPdoGetErrorString**

GtPdoSetVoltageRailSource

Purpose

Sets the specified voltage rail source.

Syntax

GtPdoSetVoltageRailSource(*nHandle, nVoltageRail, nSource, pnStatus*)

Parameters

Name	Type	Comments
nHandle	SHORT	Handle to a GT7864.
nVoltageRail	SHORT	Voltage rail to set: 1 – Voltage rail A 2 – Voltage rail B 3 – Voltage rail C 4 – Voltage rail D
nSource	SHORT	Source for the specified rail: 0 – Disconnect the specified voltage rail from all sources. 1 – Connect to the internal DAC. 2 – Connect to the external input. 3 – Connect the internal DAC to the external line and to the voltage rail.
pnStatus	LPSHORT	Returned status: 0 on success, <0 on failure.

Comments

Each of the four voltage rails has an internal (DAC) and external voltage source. These voltage sources correspond to the voltage rails as follows:

Voltage Rail	Internal Source	External Source
A	DAC A	External A
B	DAC B	External B
C	DAC C	External C
D	DAC D	External D

For example, voltage rail A connects to DAC A, External A, or to both (thus routing the internal DAC voltage to the external pin).

Example

The following example connects DAC A to Voltage Rail A:

```
SHORT nHandle, nStatus;

GtPdoSetVoltageRailSource(nHandle, GTPDO_DAC_A,
    GTPDO_DAC_RAIL_INTERNAL, &nStatus);
```

See Also

GtPdoGetVoltageRailSource, **GtPdoGetBoardSummary, GtPdoInitialize, GtPdoGetErrorString**

Appendix A – Error Codes

This appendix includes values for the *nError* parameter of the **GtPdoGetErrorString** function. (See Chapter 6.)

Note: An error code value of zero ("0") indicates no error. The function returns the string, "No error has occurred".

The error code is returned by all functions with *pnStatus*. Tables A-1 through A-4 describe the possible error code values for these Error Types:

- Resource Errors
- Parameter Error
- Board Errors (Fatal Errors)
- Miscellaneous Errors

Resource Errors

Error Code	Message
-1	Board does not exist in this base address
-2	Too many boards
-3	Unable to create Windows timer
-4	Unable to get timer
-5	Out of memory
-6	Self-test adapter is not connected
-7	Unable to create file
-8	Unable to open file
-9	Error while reading from file
-10	Invalid calibration mode

Parameter Errors

Error Code	Message
-20	Invalid parameter
-21	Invalid DAC number
-22	Illegal base address
-23	Illegal handle
-24	Illegal channel number
-25	Invalid DAC voltage
-26	Illegal mode
-26	Invalid DAC source
-28	Illegal string length
-29	Illegal relay number
-30	Invalid model number
-31	Illegal timeout or not enough time to finish the operation
-32	Error while writing DAC voltage

Board Errors (Fatal Errors)

These errors indicate that the board has a malfunction.

Error Code	Message
-50	Reference voltage out of range
-51	Unable to close a relay
-52	Unable to open a relay
-53	Unable to open/close a relay
-54	Unable to set the Measurement register
-55	Unable to set the Mode register
-56	Fatal error: board DAC was not calibrated
-57	Fatal error: Invalid DAC source

Error Code	Message
-58	Fatal error: Unable to start DAC frame
-59	Fatal error: Unable to disconnect the DAC reference from the external and internal sources
-60	Fatal error: EEPROM offset is out of range
-61	Fatal error: EEPROM rail setting value is invalid
-62	Fatal error: Invalid calibration gain value

Miscellaneous Errors

Error Code	Message
0	No error has occurred
-99	Invalid or unknown error number

Appendix B – Specifications

The following table lists the specifications for the GT7864:

Function	Values	Notes
Limits		
Max Current per Channel	200 mA	External reference only.
Max Current per DAC	100 mA	
Operating Ranges		
Output Voltage	-8V to 32V, -5V to 5V, -0.5V to 0.5V	Custom ranges are available.
Tolerances		
Resolution	14-bit	For symmetrical ranges.
Accuracy	±1 LSB ±10 mV	
Features		
Slew-Rate	8V/µS	
Output Coupling	DC	
Safety		
Protection	Short-circuit to ground	
Power-on State	Open	

Index

16-bit DLL 33
16-bit Import Library 33
32-bit DLL 33, 34
32-bit Import Library 33
Address
  Geotest 3
ATEasy 12, 34, 49
  Driver 49
Block diagram 7
Board Errors 96
Board-Handles 51
Borland 33, 34
  Examples 35
  Pascal for Windows 49
Borland-C++ 46, 47
Borland-Delphi 46, 47, 49
C++ 43, 46, 47, 48
Calibration Services
  From Geotest 3
Conventions Used in This Manual 2
Copyright ii
CTL3DV2.DLL 33
DAC 5
Delphi 49
Development Tools 12
Digital-to-Analog Converter 5
Discharge Static Electricity 14
Disclaimer ii
DLL Driver
  Using with Windows NT 45
DOS 44
DOS Static Library 35, 48
Download Files
  From Geotest 3
Driver Files 33
Dynamic-Link-Libraries 44
EEPROM 10
Electricity
  Discharge 14
Email Addresses
  Geotest 3
Example 55, 56
Example Program
  Listing 55
Example Programs 34
Examples 54
Fatal Errors 96
Function Library 11
Geotest
  ATEasy 11
GT7864
  DLL 11
  Illustration 7
  Models 5

Software 10
Specifications 99
GTPDO 13, 32
16-Bit-DLL 46, 47
Header-file 48
GTPDO.CFG 36
GTPDO.H 34
GTPDO.INS 34
GTPDO.PAS 34
GTPDO.PRG 36
GTPDO16.BAS 34
GTPDO16.DLL 33, 35, 47
GTPDO16.LIB 33
GTPDO32.BAS 34
GTPDO32.DLL 33
GTPDO32.LIB 33
GTPDO32B.LIB 33
GTPDOBC.LIB 33
GtPdoGetBoardSummary 62
GtPdoGetChannelState 64
GtPdoGetChannelVoltageRail 66
GtPdoGetDacVoltage 68
GtPdoGetDriverSummary 69
GtPdoGetErrorString 71
GtPdoGetMeasureToVoltageRail 72
GtPdoGetModel 74
GtPdoGetVoltageRailSource 76
GtPdoInitialize 78
GTPDOMS.DLL 33
GTPDOMS.LIB 35
GTPDOP16.EXE 33
GTPDOP32.EXE 33
GtPdoPanel 80
GtPdoReset 82
GtPdoSaveChannelsVoltageRails State 83
GtPdoSetChannelState 85
GtPdoSetChannelVoltageRail 87
GtPdoSetDacVoltage 89
GtPdoSetMeasureToVoltageRail 90
GtPdoSetVoltageRailSource 92
GTPDOX.C 34, 35
GTPDOX.DEF 35
GTPDOX.ICO 34, 35
GTPDOX.R 35
GTPDOX.RC 34
GTPDOX16.EXE 35
GTPDOX16.MAK 35
GTPDOX32.EXE 34
GTPDOX32.MAK 34
GTPDOXMS.EXE 35
GTPDOXMS.MAK 35
GTXI's Local Bus Connector 7
HW Driver
How does it work? 45
HW.SYS 30, 31, 33, 36, 45
HWSETUP 31
HWSETUP.EXE 33

HWTEST.EXE 33
HWVDD.DLL 33, 36, 45
I/O Base Address 26, 39
If You Need Help i
Import Libraries 44
Inspecting the Board 13
Installation 14
  Board 29
  DOS 32
  Software 30
  Windows 30
  Windows NT Kernel Mode Driver 31
Internet Download Area 3
Jumper Settings 14
  Local bus 25
  Model A 15
  Model B 17
  Model C 19
  Model D 22
Kernel Mode Driver 33
Local Bus
  Jumper settings 25
Manual Organization 1
Manual Scope 1
MEAS Channel 6
Model A
  Jumper settings 15
Model B
  Jumper settings 17
Model C
  Jumper settings 19
Model D
  Jumper settings 22
Models 5
nChannel 61
nDAC 61
nRail 61
Output Channels 5
Parameter Errors 96
Parameter Prefix Names 60
Parameters
  Common 61
Pascal 49
Pin 58 6
pnHiSrc 61
pnLoSrc 61
Programming
  ATEasy 49
  Borland-Delphi 49
  Virtual Panel 52
Programming Example 52
Programming Languages 12
Programming the Board 43
Resource Errors 95
Resource File 34, 35
Safety i
Self-test 6
Setting I/O Base Address 26

Setup 13
Software 10
  Distributing 36
  Supported development tools 12
Software Installation 30
Specifications 99
Static Library 33
Supported 43
System Requirements 13
System-Requirements 13
Table of Contents iii
Technical Support 3
Telephone Numbers
  Geotest 3
Trademarks ii
Unpacking 13
VC++ 34, 35
Virtual Panel 11, 37, 52
  Initialization 38
Visual Basic 34, 46, 47, 49
Voltage Rails 5
  Range 6
Warranty i
Web Site
  Geotest 3
Windows 16-bit DLL Driver 47
Windows 3.1 30, 47
Windows 95 30, 47
Windows 98 30, 47
Windows and DOS Drivers 44
Windows NT 30, 47
  Kernel Mode Driver 30, 31, 33
Windows NT Kernel Mode Driver
  Manual installation 31