

ARTISAN®
TECHNOLOGY GROUP

Your **definitive** source for quality pre-owned equipment.

Artisan Technology Group
(217) 352-9330 | sales@artisantg.com | artisantg.com

Full-service, independent repair center
with experienced engineers and technicians on staff.

We buy your excess, underutilized, and idle equipment
along with credit for buybacks and trade-ins.

Custom engineering
so your equipment works exactly as you specify.

- Critical and expedited services
- In stock / Ready-to-ship
- Leasing / Rentals / Demos
- ITAR-certified secure asset solutions

Expert team | Trust guarantee | 100% satisfaction

All trademarks, brand names, and brands appearing herein are the property of their respective owners.

Visit our website - *Click HERE*

Personal Daq/50 Series

USB Data Acquisition Modules



Personal Daq/54
Personal Daq/55
Personal Daq/56
PDQ1
PDQ2

Personal Daq/50 Series

p/n **491-0901** rev **7.0**

372182A-01

Measurement Computing
10 Commerce Way
Norton, MA 02766

(508) 946-5100
Fax: (508) 946-9500
info@mccdaq.com
www.mccdaq.com

Warranty Information

Contact Measurement Computing by phone, fax, or e-mail in regard to warranty-related issues:
Phone: (508) 946-5100, fax: (508) 946-9500, e-mail: info@mccdaq.com

Limitation of Liability

Measurement Computing cannot be held liable for any damages resulting from the use or misuse of this product.

Copyright, Trademark, and Licensing Notice

All Measurement Computing documentation, software, and hardware are copyright with all rights reserved. No part of this product may be copied, reproduced or transmitted by any mechanical, photographic, electronic, or other method without Measurement Computing's prior written consent. IOtech product names are trademarked; other product names, as applicable, are trademarks of their respective holders. All supplied IOtech software (including miscellaneous support files, drivers, and sample programs) may only be used on one installation. You may make archival backup copies.

CE Notice

Many Measurement Computing products carry the CE marker indicating they comply with the safety and emissions standards of the European Community. When applicable these products have a Declaration of Conformity stating which specifications and operating conditions apply. You can view the Declarations of Conformity at www.mccdaq.com/legal.aspx (CE Information page).

Warnings, Cautions, Notes, and Tips



Refer all service to qualified personnel. This caution symbol warns of possible personal injury or equipment damage under noted conditions. Follow all safety standards of professional practice and the recommendations in this manual. Using this equipment in ways other than described in this manual can present serious safety hazards or cause equipment damage.



This warning symbol is used in this manual or on the equipment to warn of possible injury or death from electrical shock under noted conditions.



This ESD caution symbol urges proper handling of equipment or components sensitive to damage from electrostatic discharge. Proper handling guidelines include the use of grounded anti-static mats and wrist straps, ESD-protective bags and cartons, and related procedures.



This symbol indicates the message is important, but is not of a Warning or Caution category. These notes can be of great benefit to the user, and should be read.



In this manual, the book symbol always precedes the words "Reference Note." This type of note identifies the location of additional information that may prove helpful. References may be made to other chapters or other documentation.



Tips provide advice that may save time during a procedure, or help to clarify an issue. Tips may include additional reference.

Specifications and Calibration

Specifications are subject to change without notice. Significant changes will be addressed in an addendum or revision to the manual. As applicable, the hardware is calibrated to published specifications. Periodic hardware calibration is not covered under the warranty and must be performed by qualified personnel as specified in this manual. Improper calibration procedures may void the warranty.

Your order was carefully inspected prior to shipment. When you receive your system, carefully unpack all items from the shipping carton and check for physical signs of damage that may have occurred during shipment. Promptly report any damage to the shipping agent and your sales representative. Retain all shipping materials in case the unit needs returned to the factory.

CAUTION



Using this equipment in ways other than described in this manual can cause personal injury or equipment damage. Before setting up and using your equipment, you should read *all* documentation that covers your system. Pay special attention to Warnings and Cautions.

Note: During software installation, Adobe® PDF versions of user manuals will automatically install onto your hard drive as a part of product support. The default location is in the **Programs** group, which can be accessed from the *Windows Desktop*. Initial navigation is as follows:

Start [Desktop "Start" pull-down menu]
⇒ **Programs**
⇒ **IOtech PDaqView Software**

You can also access the PDF documents directly from the data acquisition CD by using the <**View PDFs**> button located on the opening screen.

Refer to the PDF documentation for details regarding both hardware and software.

A copy of the Adobe Acrobat Reader® is included on your CD. The Reader provides
a means of reading and printing the PDF documents.

About This Manual

Chapter 1: *Personal Daq —Unit Startup* provides information to get your Personal Daq system up and running. The chapter includes installation steps, basic concepts regarding the *Personal DaqView* software program, and steps for acquiring data.

Chapter 2: *General Information & Specifications* gives a general description of Personal Daq and related hardware including PDQ expansion modules. Basic operational concepts and product specifications are included.

Chapter 3: *Hardware Setup* provides detailed information regarding Personal Daq and includes information regarding direct connection to PC USB, connection to a USB-powered hub, and connection to a self-powered USB hub. The chapter also includes instruction for connecting a PDQ expansion module and input signal lines.

Chapter 4: *Personal DaqView* explains the ready-to-use *Personal DaqView* software that comes with every Personal Daq. Topics include detailed explanations of the program's pull-down menus, toolbar icons, and keypad control options. The chapter includes discussion of ***Personal DaqViewXL****, an* add-in software application that provides setup and data acquisition capabilities from within *Microsoft Excel*™.

Chapter 5: *Signal Management* discusses signal modes, system noise, and includes a brief discussion of channel control and expansion.

Chapter 6: *Calibration* explains how to perform periodic calibrations using the *Windows*-based program, *UserCal*.

Chapter 7: *Troubleshooting and Customer Assistance*

Appendices

***Appendix A*: API Custom Program Models** explains the program models supplied on the release disk.

***Appendix B*: API Commands** describes the entire command set for Personal Daq. Syntax, parameters, interpretation, and error codes are explained. Sections on the individual commands include their parameters, types, typical use and related information.

***Appendix C*:** *N/A, removed from manual.*

Appendix D: **Custom Labels** provides blank labels and a Personal Daq channel layout reference. The appendix also pertains to the **pDaq_CustomLabels.doc**. The file is located in the target directory: **\\Program Files\pDaqView**. It provides blank labels in a table format that you can write in, edit, and print out from your PC.

Table of Contents

1 - Personal Daq — Unit Startup

Overview ……1-1

Inspect Your System ……1-1

Install Software ……1-2

Install Hardware ……1-2

Mount Personal Daq Modules to DIN Rail (option) ……1-2
Connect PDQ Module (option) ……1-4
Connect Personal Daq to Host PC ……1-5
Connect Channel Signal Inputs ……1-6

Start Personal DaqView ……1-7

Configure System ……1-8

Channel Configuration ……1-9
Acquisition Configuration ……1-9
Data Destination Configuration ……1-10

Collect Data ……1-10

Quick Start for Personal DaqView …… 1-11

2 - General Information

General Description ……2-1

Channel Capacities ……2-1
Features ……2-2

Theory of Operation ……2-3

Universal Serial Bus (USB) ……2-3
Power Line Rejection ……2-3
Optical Isolation …… 2-4
A/D Conversion ……2-4
Input Ranges ……2-4
Analog Input Configuration ……2-4
Measurement Duration, Sample Rate, and Resolution ……2-5
Automatic Calibration …… 2-6
Thermocouple Measurements ……2-6
Frequency Measurements ……2-7
Digital I/O ……2-7

Personal Daq Specifications ……2-8

Analog Specifications ……2-9
Frequency Specifications ……2-11
Digital I/O Specifications……2-11
General Specifications ……2-11
Optional Accessories …… 2-11
Channel Connection Layouts ……2-12
Calibration ……2-14

3 - Hardware Setup

Personal Daq, System Components … 3-2

Personal Daq …… 3-2
PDQ Expansion Modules …… 3-2
USB Hubs and Power Adapters …… 3-2

Connecting Your Personal Daq Acquisition System …… 3-3

Connecting a PDQ Expansion Module to a Personal Daq …… 3-3
Connecting Various Hardware Setups …… 3-3

4 - Personal DaqView

Overview …… 4-2

Main Control Window …… 4-3

Toolbar Buttons …… 4-3
Pull-Down Menus …… 4-3

Channel Configuration Window …… 4-6

Channel Configuration Window Toolbar … 4-6
Channel Configuration Window Pull-down Menus …… 4-6
Common Spreadsheet Columns …… 4-7
Analog Input Spreadsheet …… 4-10
Frequency/Pulse Input Spreadsheet …… 4-12
Digital Input/Output Spreadsheet …… 4-14
Configure Acquisition Dialog Box …… 4-15
Configure Data Destination & File Converter Preferences …… 4-18
Sequential Destinations (Auto Rearm) …4-19

Bar Graph, Analog, & Digital Meters …4-20

Meter Toolbars …… 4-20
Meter Pull-Down Menus …… 4-21
Meters Configuration Menu …… 4-21
Configuring a Meter …… 4-22
Bar Graph Meters …… 4-24
Analog Meters …… 4-25
Digital Meters …… 4-26

Chart Display …… 4-27

A Note Regarding Standard, Plus, and XL Version Software …… 4-27
Groups, Charts, & Channels …… 4-27
Chart Display Window …… 4-27
Pull-Down Menus …… 4-28
Toolbar Items …… 4-29
Chart and Channel Information Regions …… 4-30
Accessing the Display Configuration Setup Box …… 4-31
Editing a Chart Display Configuration …..4-32
Manually Configuring a Chart Display …. 4-34

Chart Setup Wizard …… 4-37

Introduction …… 4-37
Automatic Display Setup using the *Chart Setup Wizard* …… 4-38
Bypassing Automatic Chart Setup …… 4-39

Using Personal DaqView with Microsoft Excel™ …… 4-40

5 – Signal Management

Overview ……5-1

Channel Control and Expansion ……5-3

Signal Acquisition ……5-3

Measurement Duration, Sample Rate, and Resolution ……5-3

Under Sampling and Aliasing …… 5-3

Triggering …….5-5

Input Isolation …….5-5

Signal Modes ……5-6

System Noise ……5-7

Averaging ……5-7

Analog Filtering ……5-7

Input and Source Impedance ……5-7

Crosstalk ……5-7

6 - Calibration

Introduction ……6-1

Required Equipment ……6-2

Calibration Procedure ……6-2

7 – Troubleshooting & Customer Assistance

Radio Frequency Interference …… 7-1

Electrostatic Discharge (ESD) …… 7-1

Troubleshooting …… 7-2

Customer Assistance …… 7-5

Appendices

A – API Custom Program Models

B – API Commands

C – *N/A, removed from manual.*

D – Custom Labels

Unit Startup 1

Overview ……1-1

Inspect Your System ……1-1

Install Software ……1-2

Install Hardware ……1-2

Mount Personal Daq Modules to DIN Rail (option) ……1-2

Connect PDQ Module (option) ……1-4

Connect Personal Daq to Host PC …1-5

Connect Channel Signal Inputs ……1-6

Start Personal DaqView ……1-7

Configure System ……1-8

Channel Configuration ……1-9

Acquisition Configuration ……1-9

Data Destination Configuration ……1-10

Collect Data ……1-10

Quick Start for Personal DaqView …… 1-11

Overview

Note: If you used the Personal Daq Quick Start document (491-0940) to startup your unit, you may choose to skip this chapter.

Note: Chapter 3, *Hardware Setup*, contains detailed information pertaining to hardware issues.

This chapter provides the steps to connect, power up, and run a simple Personal Daq system consisting of one Personal Daq Unit and one PDQ module. Chapter 3 discusses setting up more involved Personal Daq systems.

The basic Startup Steps are:

1. Inspect Your System
2. Install Software
3. Install Hardware
    a. Mount Personal Daq Modules to DIN Rail (option)
    b. Connect PDQ Module (option)
    c. Connect Personal Daq System to your computer
    d. Connect Channel Inputs
4. Start *Personal DaqView*, Configure System, and Collect Data

Inspect Your System

If you have not already done so, check your package contents for damage that may have occurred during shipment. Immediately report any damage to the shipping agent and your sales representative. Retain all shipping materials in case the unit must be returned to the factory.

Personal Daq shipments typically contain combinations of the following items, depending on the order. Your order may include additional items.

Item	Description
Personal Daq	Personal Daq/54, /55, or /56 USB Data Acquisition System
1022-0601 501395x-01	Data Acquisition CD. The CD includes several program install options. Note that the CD includes user manuals.
491-0940	Personal Daq 50 Series Quick Start Guide
CA-179-x	Optional USB Cable, x = meter length., i.e., CA-179-1 (1 meter) CA-179-3 (3 meter), or CA-179-5 (5 meter)
PDQ1 or PDQ2	Optional expansion modules [cannot be used with Personal Daq/54]
PDQ10	Optional PDQ10 DIN-Rail Mounting Kit



Reference Note:
You can refer to our latest catalog for a list of accessories available for Personal Daq.

CAUTION

When using Personal Daq modules to acquire data, computer *energy save* modes can cause false data readings. Prior to using Personal Daq modules, ensure your computer's *energy save* mode is disabled. If needed, consult your PC user's manual to disable *energy save* (*power suspension*) modes.

Install Software

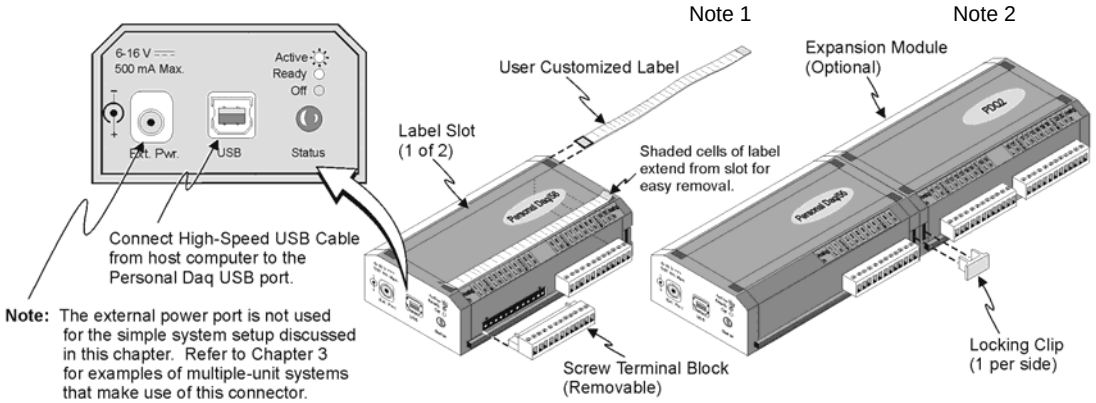
IMPORTANT: Software must be installed before installing hardware.

1. Remove previous version Daq drivers, if present. You can do this through Microsoft's **Add/Remove Programs** feature.
2. Place the Data Acquisition CD into the CD-ROM drive. *Wait for PC to auto-run the CD. This may take a few moments, depending on your PC.* If the CD does not auto-run, use the Desktop's Start/Run/Browse feature.
3. After the intro-screen appears, follow the screen prompts.

Upon completing the software installation, continue with step 2, *Install Hardware*.

Install Hardware

Depending on your order, your Personal Daq unit may not require all the steps under this heading; for example: if you did not order a PDQ expansion module you would not connect one. If a step does not apply to your unit, simply go on to the next one.



Note 1: Label slots are included on Personal Daq/55 and /56. The slots are not included on Personal Daq/54.

Note 2: An expansion module (PDQ1 or PDQ2) can be added to Personal Daq/55 and /56 units. Although an expansion module can physically connect to a Personal Daq/54, it will not function with the unit. Personal Daq/54 does not include expansion functionality. For /54 devices, the 25-pin port is used only for calibration as discussed in chapter 6.

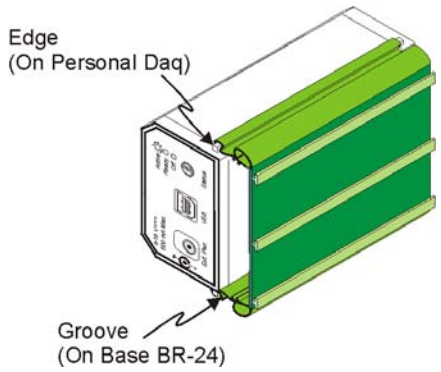
Mount Personal Daq Modules to DIN Rail (Option)

The optional PDQ10 DIN Rail Mounting Kit contains a base (BR-24), two feet (FE-8), and an installation guide. The information found in the guide has been repeated here for user convenience.

1) Mount Personal Daq Module to Base (BR-24)

Note: If your Personal Daq system makes use of an expansion module you will need a second DIN-rail kit for mounting the expansion module.

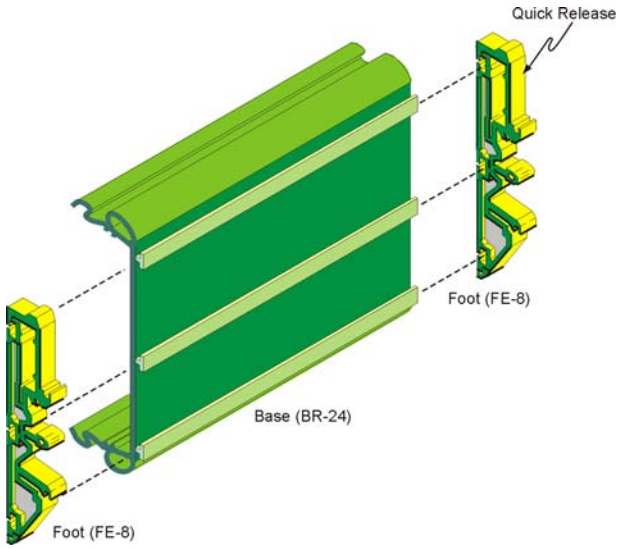
1. Remove terminal blocks from Personal Daq Module.
2. Snap Personal Daq's *mount edges* into grooves of base (BR-24).
3. Re-install Personal Daq's terminal blocks**.**



Inserting Personal Daq's Mount Edges into Grooves of Base (BR-24)

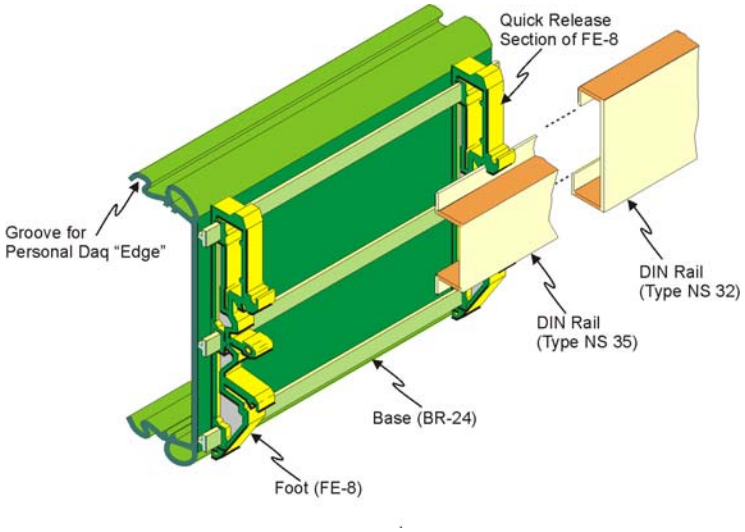
2) Attach Feet (FE-8) to Base (BR-24)

1. Slide first foot (FE-8) onto base (BR-24). Note the orientation of the *Quick Release* (see following figure).
2. Keeping the same *Quick Release* orientation, i.e., both "up" or both "down" as per your preference, slide second foot onto base.
3. Position feet near edge of base. This will provide for best support of unit to DIN rail.



Attaching Feet (FE-8) to Base (BR-24)

3) Attach Assembly to DIN Rail (Types NS 35, or NS 32)



Installing Base/Foot Assembly to a DIN Rail Mount

The following steps are based on the orientation illustrated above. Note that feet (FE-8) contain *rail catches* for both NS 35 and NS 32 type DIN rails. In each case a different set of *rail catches* is used.

1. Place the applicable *lower rail catch* of the feet onto the *lower lip* of the DIN rail.
2. Push the entire base/foot assembly toward the DIN rail. The assembly will snap into position.

Removing Units from a Type NS 35 DIN Rail

1. Push on both *Quick Release* sections of the FE-8 feet at the same time.
2. Lightly pull the base/foot assembly away from the DIN rail.

Removing Units from a Type NS 32 DIN Rail

1. Push up on the base/foot assembly.
2. Lightly pull the base/foot assembly away from the DIN rail.

Connect PDQ Module (option)

CAUTION



The discharge of static electricity can damage some electronic components. Semiconductor devices are especially susceptible to ESD damage. You should always handle components carefully, and you should never touch connector pins or circuit components unless you are following ESD guidelines in an appropriate ESD controlled area. Such guidelines include the use of properly grounded mats and wrist straps, ESD bags and cartons, and related procedures.

CAUTION



Never connect an expansion module to (or remove it from) a Personal Daq main unit while the main unit is connected to a power source. Such action may result in EEPROM errors and loss of calibration data.

CAUTION

Never remove a USB cable from an active Personal Daq device while an acquisition is in progress. An active device is any device that is currently open and has channels configured for scanned input. Such disconnection may require you to exit and then re-launch Personal DaqView, after the USB cable has been connected.

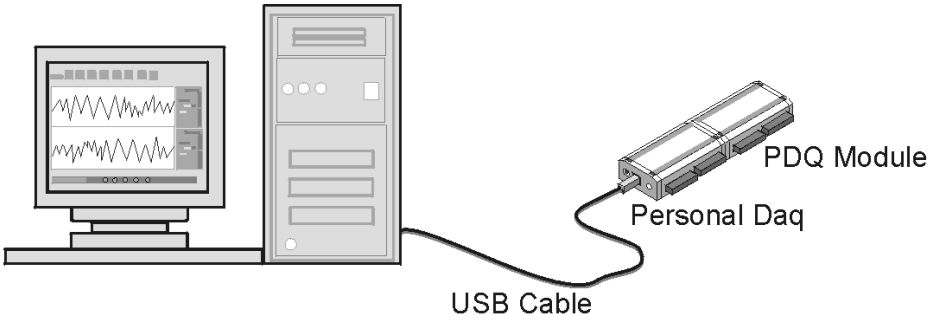
If you are going to install a PDQ expansion module:

1. Ensure that no power cable or USB cable is connected to the main unit
2. Plug the expansion module into the DB25 connector-end of the Personal Daq main unit
3. Lock the modules together with two locking clips (provided). Refer to the previous figure.

Connect Personal Daq to Host PC

When using a power adapter with your Personal Daq system, be sure to supply power (from the adapter to the Personal Daq) before connecting the USB cable. This allows Personal Daq to inform the host computer (upon connection of the USB cable) that the unit requires minimal power from the computer.

Use an approved high-speed USB cable to connect the Personal Daq system to one of the host computer's USB ports. There is no need for an additional power source in this setup since the power pins (of the PC's USB connection) supply 500 mA at 4 to 5.25 V. Additional setup examples are described in Chapter 3, some of which involve USB hubs and/or power adapters.



Connection of a Simple Personal Daq System to a Computer USB Port

Certain notebook computers require the use of a power adapter with your Personal Daq.

Connect Channel Signal Inputs

CAUTION



The discharge of static electricity can damage some electronic components. Semiconductor devices are especially susceptible to ESD damage. You should always handle components carefully, and you should never touch connector pins or circuit components unless you are following ESD guidelines in an appropriate ESD controlled area. Such guidelines include the use of properly grounded mats and wrist straps, ESD bags and cartons, and related procedures.

Use Personal Daq's screw terminals to connect channel inputs to your Personal Daq system. Note that the terminal blocks are detachable for ease in making connections. The main module (Personal Daq/54, Personal Daq/55, and Personal Daq/56) and optional PDQ expansion modules (PDQ1 and PDQ2) have labels which clearly identify each input type and channel number. Each Analog Input channel can be configured for *single-ended* or *differential* volts, or for differential thermocouple inputs. The non-analog channels are designated as Frequency/Pulse Input (F) and Digital I/O (D).

Note: Personal Daq/54 does not accept frequency/pulse input or digital I/O.

Note: Personal Daq/54 does not support expansion modules.



Reference Note:
Connections for single-ended and differential modes are depicted in the figure on page 1-8.



Reference Note:
Specifications are provided in Chapter 2.

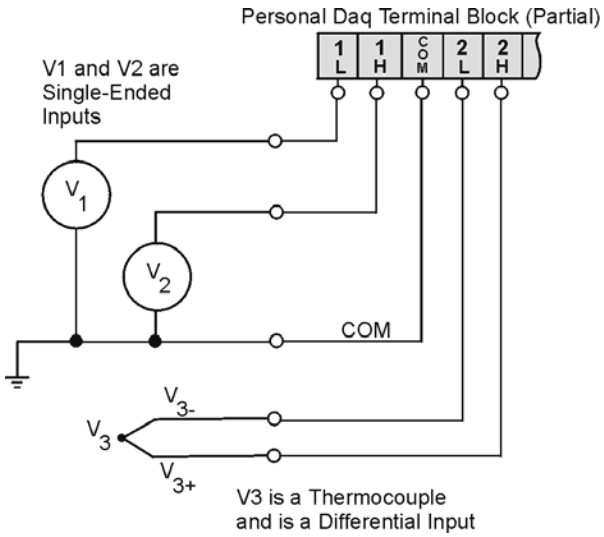
Connecting Thermocouple Wires

Thermocouple wires are to be connected in *differential mode* only.

In Personal Daq applications, thermocouples should <u>not</u> be connected *single-ended*. Doing so can result in noise and false readings. This is especially true when acquiring other high-amplitude signals in conjunction with thermocouple signals that are connected *single-ended*.

Differential connection is made as follows:
(a) the red wire connects to the channel's Low (L) connector.
(b) the second [color-coded] wire connects to the channel's High (H) connector.

The section entitled *Signal Modes*, in chapter 5, contains additional information.



Single-Ended and Differential Connections to Analog Input Channels

Start Personal DaqView

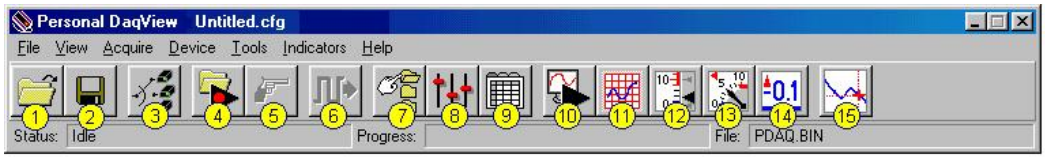
From Windows, open Personal DaqView by double clicking on the Personal DaqView icon, or by using the Windows Desktop Start Icon to access the Personal DaqView program. You will find Personal DaqView listed in the desktop's Program group.

By default, the Personal Daq files will install in C:/Program/Applications. However, you may have chosen a different install setup when prompted during the installation process.

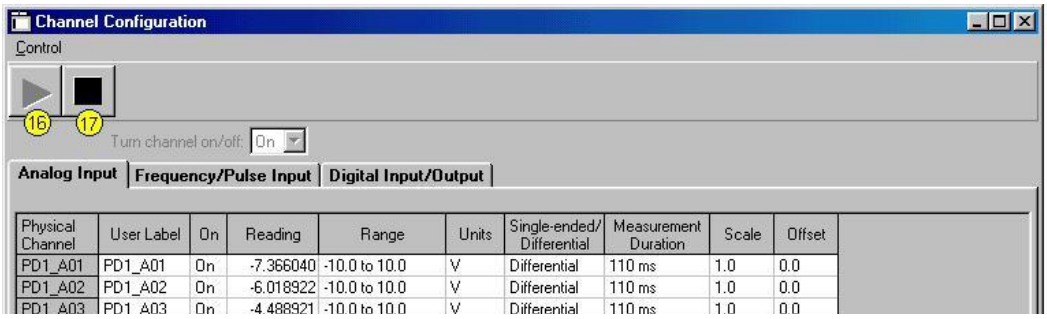
Once the program is executed, the software identifies your Personal Daq device and brings up the *Main Control Window*. This window is discussed briefly in the following text, and in more detail in Chapter 4.

Configure System

This step pertains to configuration of channels and acquisition parameters through Personal DaqView software. To configure a set of parameters, use the appropriate toolbar button (or the View pull-down menu) to open the applicable window. Clicking on the Configure Channel Settings button (9) brings up the *Channel Configuration Window*.



Main Control Window



Channel Configuration Window, with Analog Input Spreadsheet Selected

	Button	Function
1	Open Configuration File	Opens a selected configuration file.
2	Save Configuration File	Saves the current configuration file to disk.
3	Select Active Device	Provides a means of selecting active devices from the Personal Daq system.
4	Arm Trigger for Disk Recording	Arms the trigger and stores acquisition data to a designated disk file. If Auto Rearm is selected, clicking this button puts Auto rearm in effect. This button, is also used to disarm the data acquisition.
5	Manual Trigger	Used to trigger the device when the mode of trigger is set to "Manual." Note that the Manual Trigger button can not be clicked until after the trigger is armed, for example, by first clickng button 4.
6	Update Digital Outputs	Updates digital outputs for all digital channels that are selected to "output state." also see *Digital Input/Output Spreadsheet*, in Chapter 4). Does not apply to Personal Daq/54.
7	Configure Data Destination	Accesses the Configure Data Destination window. Note that this window provides a means of selecting sequential destinations through an auto rearm feature.
8	Configure Acquisition	Accesses the Configure Acquisition window.
9	Configure Channel Settings	Brings up the Channel Configuration window. From this window you can configure channels for Analog Input, Frequency/Pulse Input, and Digital Input/Output channels, depending on which tab is selected.
10	Update All Indicators	Starts all on-screen indicators with a display of up-to-date data. Has no affect on the recording of data to disk. Auto Rearm, even if selected, will not occur when using this control. This button is also used to pause all indicators.
11	Display Scrolling Charts	Displays data graphically in a scrolling chart.
12	Display Bar Meters	Displays data in a bar graph format.
13	Display Analog Meters	Displays data displayed in a dial-gage format.
14	Display Digital Meters	Displays data in a digital meter format.
15	View Data	Launches an independent post-data acquisition program such as eZ-PostView. Refer to the Post Acquisition Analysis PDF (included on your CD) for detailed information.
16	Enable Readings Column	Activates the *Channel Configuration Window's* reading column. Does not affect the recording of data to disk.
17	Disable Readings Column	Stops the *Channel Configuration Window's* reading column. Does not affect the recording of data to disk.

Channel Configuration



The *Channel Configuration Window* first opens with the Analog Input screen selected. You can change from one configuration screen to another by selecting the appropriate tab. More information regarding Personal DaqView appears in Chapter 4 of this manual. You can configure channels from the three configuration screens as indicated in the following table.

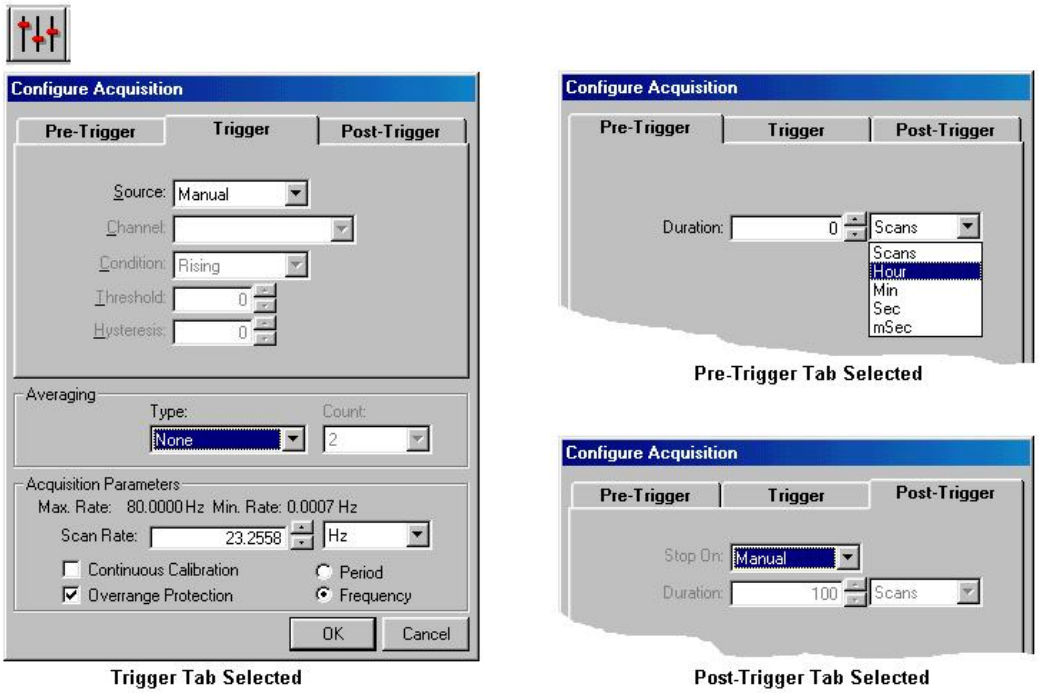
Channel Type	User Configurable Parameters
Analog Input	User Label, On, Range, Units, Single-Ended/Differential, Measurement Duration, Scale, Offset
Frequency/Pulse Input	User Label, On, Type, Units, Edge, Debounce, Min. Value, Max. Value, Resolution (Settling Time), Scale, Offset. Frequency/Pulse Input does not apply to Personal Daq/54.
Digital Input/Output	User Label, On, Input/Output, Output State, Power-Up State. Digital I/O does not apply to Personal Daq/54.

Note: With the mouse cursor positioned in the desired spreadsheet cell, you can:

- ***"Single-click"* with the left mouse button** to open an associated pull-down list for the applicable cell, from which a selection can be made. This pull-down list appears just below the toolbar.
- ***"Double-click"* with the left mouse button** to cycle through listed selections or write-enable a cell, as applicable. If the cell has a given parameter list (such as those in the *On*, *Range*, and *Measurement Duration* columns) the parameter will change with each double-click, allowing you to cycle through all possible selections. Note that these selections are repetitive; in other words, you will eventually advance to the same selection you started with. "Type-in" cells (such as User Label, Scale, and Offset) can be selected on double-click for easy editing.
- ***"Single-click"* left, then "*single-click*" right** to write-enable a cell. Completing this action with the mouse buttons (while having the cursor on a cell such as Scale or Offset) allows you to use you PC's keypad to type the desired value into the field.
- ***"Single-click"* left, "*single-click*" right, then "*single-click*" left again** to open an associated pull-down list for the applicable cell, from which a selection can be made. This pull-down list appears in the selected cell's row.

Acquisition Configuration

To configure acquisition parameters, activate the *Configure Acquisition Window* by using toolbar button (8).



Configure Acquisition Window

The *Configure Acquisition Window* has the following default settings.

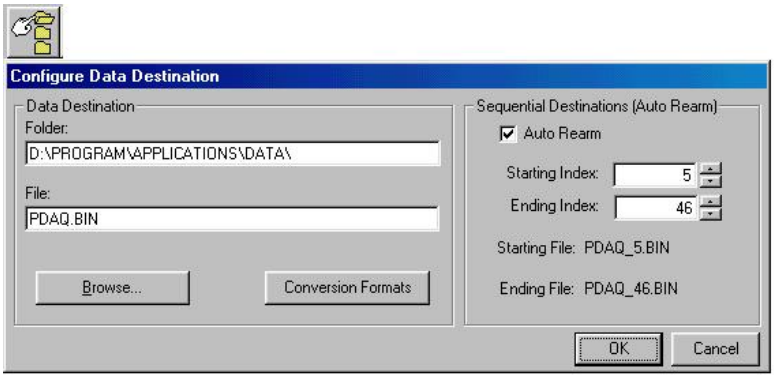
Pre-Trigger: No duration set, i.e., a duration of 0 scans
Trigger: Immediate
Post Trigger: Manual stop
Averaging: None
Acquisition Parameters:
Set for Maximum scan rate
Frequency: selected
Period: not-selected
Continuous Calibration: not-selected
Overrange Protection: selected

You can change the acquisition setup as desired. Explanations of setup options are discussed in Chapter 4.

Functions that can be obtained with *Main Control Window* toolbar buttons can also be obtained through pull-down menus (discussed subsequently). See Chapter 4 for details.

Data Destination Configuration

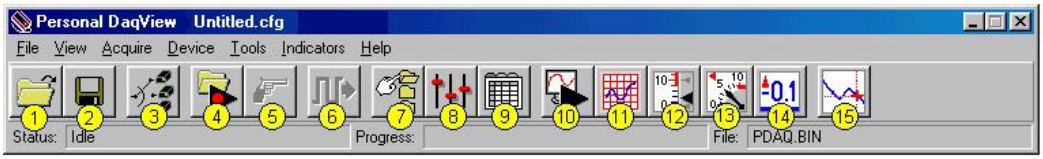
The *Data Destination window* can be accessed by using button (7). From the Data Destination window you can assign a filename and folder location for the acquisition data. More detailed information is provided in Chapter 4.



Data Destination Window

Collect Data

Click the Enable Readings Column button (16) [see page 1-8], or the Update All Indicators button (10) [see following figure] to start the acquisition. The data acquisition begins and the *readings* column becomes active. However, data is not recorded to disk. Clicking the Arm Trigger for Disk Recording button (4) will send the data to disk.



Click one of the toolbar's display icon buttons (11, 12, 13, or 14) to see your data in the form of a chart or meter. Display options are as follows: (11) is scrolling charts, (12) is bar graph meters, (13) is analog meters, and (14) is digital meters. Note that you can view all display types, or a combination of them, at the same time.

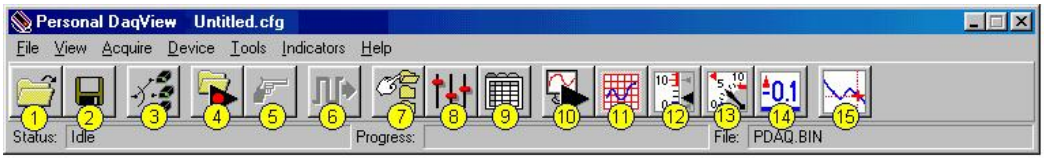
Note 1: For scrolling charts, a Chart Setup Wizard is used to set up the desired chart display. Channels not set up in the display can still be enabled and read on the channel configuration window.

Note 2: Chapter 4 is devoted entirely to the *Personal DaqView* program and its *Chart Setup Wizard* feature.

Quick Start for Personal DaqView

Once your Personal Daq system has been properly connected to the PC and to the desired input signals, the following steps may be used to start Personal DaqView and begin collecting data. For this "Quick Start" approach to collecting data we will be making use of the program's default settings.

Note that acquisition settings can be changed from the *Configure Acquisition Window*. This window is accessed via button (8). Refer to the following figure for location.



When you first open Personal DaqView the *Main Control* and *Channel Configuration Windows* appear. These are portrayed on the following page. The *Channel Configuration Window* appears with the Analog Input spreadsheet opened, and with channel PD1_A01 turned "On." Tabs on the window (see following figure) allow for a quick transition from one spreadsheet to another.

1. From Windows, open Personal DaqView by double-clicking on the Personal DaqView icon, or by using the Windows Desktop Start Icon to access the Personal DaqView program.

   Once the program is executed, the software identifies your Personal Daq device and brings up the *Main Control Window* and *Channel Configuration Window* (with the Analog Input Spreadsheet Selected).

   Note: If the *Channel Configuration Window* is not visible, click button (9).

2. Select the tab for the desired spreadsheet. **Analog Input** is the default. Note that the **Frequency/Pulse Input** and **Digital Input/Output** tabs do not apply to Personal Daq/54.

3. Ensure the desired channels are enabled. Double-click in a channel's On/Off cell to enable or disable a channel, as applicable.

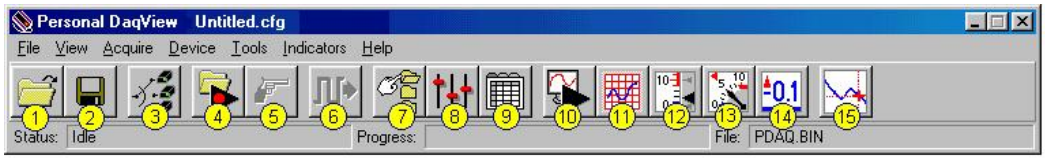
4. ***If you want to record data to disk***, click on button (4) Arm Trigger for Disk Recording. The data acquisition begins and data is stored to a disk file as indicated in the Data Destination Window. Note that the default destination, including the default file name of **`PDAQ.BIN`**, is as follows: **`C:\PROGRAM FILES\PDAQVIEW\PDAQ.BIN`**. You can, of course, specify a different location.

5. ***To see real time readings on the Channel Configuration Window***, click on button (10) Update All Indicators or button (16) Enable Readings Column. The location of button (16) is indicated on page 1-8.

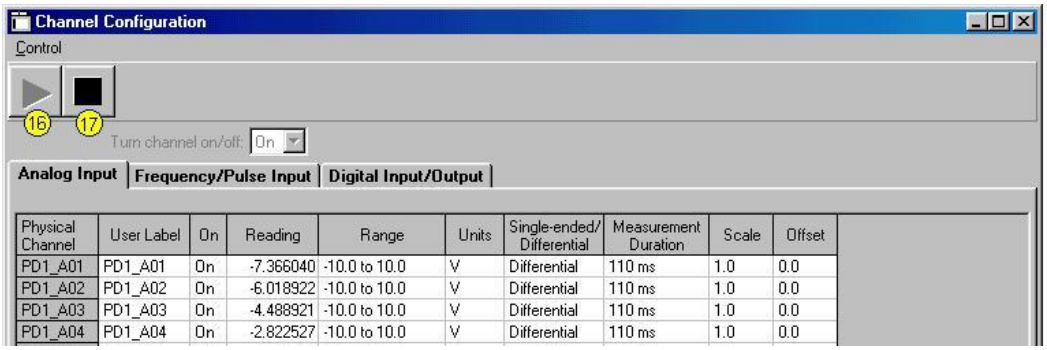
6. ***If you want to read data, but not record to disk***, click on button (10) or (16), but not button (4). In addition to the Channel Configuration Windows reading column, you can click on one or more display icon buttons (11, 12, 13, or 14) to see your data in the form of a chart or meter. You can view all display types, or a combination of them, at the same time.

   Display options, listed as button number followed by display type, are:

   - (11) scrolling charts
   - (12) bar graph meters
   - (13) analog meters
   - (14) digital meters



Main Control Window



Channel Configuration Window, with Analog Input Spreadsheet Selected

	Button	Function
1	Open Configuration File	Opens a selected configuration file.
2	Save Configuration File	Saves the current configuration file to disk.
3	Select Active Device	Provides a means of selecting active devices from the Personal Daq system.
4	Arm Trigger for Disk Recording	Arms the trigger and stores acquisition data to a designated disk file. If Auto Rearm is selected, clicking this button puts Auto rearm in effect. This button, is also used to disarm the data acquisition.
5	Manual Trigger	Used to trigger the device when the mode of trigger is set to "Manual." Note that the Manual Trigger button can not be clicked until after the trigger is armed, for example, by first clicking button 4.
6	Update Digital Outputs	Updates digital outputs for all digital channels that are selected to "output state." also see *Digital Input/Output Spreadsheet*, in Chapter 4). Digital I/O does not apply to Personal Daq/54.
7	Configure Data Destination	Accesses the Configure Data Destination window. Note that this window provides a means of selecting sequential destinations through an auto rearm feature.
8	Configure Acquisition	Accesses the Configure Acquisition window.
9	Configure Channel Settings	Brings up the Channel Configuration window. From this window you can configure channels for Analog Input, Frequency/Pulse Input, and Digital Input/Output channels, depending on which tab is selected.
10	Update All Indicators	Starts all on-screen indicators with a display of up-to-date data. Has no affect on the recording of data to disk. Auto Rearm, even if selected, will not occur when using this control. This button is also used to pause all indicators.
11	Display Scrolling Charts	Displays data graphically in a scrolling chart.
12	Display Bar Meters	Displays data in a bar graph format.
13	Display Analog Meters	Displays data displayed in a dial-gage format.
14	Display Digital Meters	Displays data in a digital meter format.
15	View Data	Launches an independent post-data acquisition program, such as eZ-PostView. Refer to the PostAcquisition Analysis PDF (included on your CD) for detailed information.
16	Enable Readings Column	Activates the *Channel Configuration Window's* reading column. Does not affect the recording of data to disk.
17	Disable Readings Column	Stops the *Channel Configuration Window's* reading column. Does not affect the recording of data to disk.

If needed, refer to Chapter 4, *Personal DaqView*, for more detailed information.

General Information & Specifications 2

General Description ……2-1
Channel Capacities ……2-1
Features ……2-2
Theory of Operation ……2-3
Universal Serial Bus (USB) ……2-3
Power Line Rejection ……2-3
Optical Isolation …… 2-4
A/D Conversion ……2-4
Input Ranges ……2-4
Analog Input Configuration ……2-5
Measurement Duration, Sample Rate, and Resolution ……2-5
Automatic Calibration …… 2-7
Thermocouple Measurements ……2-7
Frequency Measurements ……2-8
Digital I/O ……2-8
Personal Daq Specifications ……2-9
Analog Specifications ……2-10
Input Voltage Ranges ……2-10
Voltage Specifications ……2-10
Temperature Specifications …2-11
Thermocouple Accuracy ……2-11
Frequency Specifications ……2-12
Digital I/O Specifications……2-12
General Specifications ……2-12
Optional Accessories …… 2-12
Channel Connection Layouts ……2-13
Calibration ……2-15

General Description

Personal Daqs are compact data acquisition devices that make use of the Universal Serial Bus (USB). They can be located up to 5 meters (16.4 feet) from the host PC, allowing the unit to reside close to the point of measurement. This reduces noise and improves reading accuracy. USB provides both high-speed communication and power to the Personal Daq, allowing for a single cable connection to the PC. No additional power supplies are required, except in special setups of multiple units, or when connected to certain notebook PCs.

Note: Advanced setups, using several Personal Daq units, are possible. These setups require USB hubs (*self-powered* or *USB-powered*). In addition, depending on the setup, certain Personal Daqs in the system may require connection to a power adapter as indicated in Chapter 3, *Hardware Setup*.

Note: Some notebook PCs require the use of a power adapter with the Personal Daq. Chapter 3 contains more information regarding power adapters.

Personal Daq/55 and /56 allow for system expansion by snapping a PDQ expansion module onto the unit. Chapter 3, *Hardware Setup*, contains detailed information regarding various setup options.
Although Personal Daq/54 has a 25-pin connector, it does not accept expansion modules. The connector, in the case of Personal Daq/54, is used only for calibration as discussed in chapter 6.

Channel Capacities

The following table highlights the differences between modules and provides a means for calculating the total channel capacity of a Personal Daq system.

Channel Capacities for Various Personal Daq Setups

Personal Daq/54	Volts Inputs	TC Inputs	Digital I/O	Freq/Pulse Inputs
Personal Daq/54 ---- *No Expansion* ----	5 DE, or 10 SE	5 DE	---- N/A ----	---- N/A ----
Personal Daq/55 Systems	**Volts Inputs**	**TC Inputs**	**Digital I/O**	**Freq/Pulse Inputs**
Personal Daq/55	5 DE, or 10 SE	5 DE	8	2
Personal Daq/55 with PDQ1	15 DE, or 30 SE	15 DE	24	2
Personal Daq/55 with PDQ2	25 DE, or 50 SE	25 DE	8	2
Personal Daq/56 Systems	**Volts Inputs**	**TC Inputs**	**Digital I/O**	**Freq/Pulse Inputs**
Personal Daq/56	10 DE, or 20 SE	10 DE	16	4
Personal Daq/56 with PDQ1	20 DE, or 40 SE	20 DE	32	4
Personal Daq/56 with PDQ2	30 DE, or 60 SE	30 DE	16	4

DE = Differential Mode, SE = Single-Ended Mode

Note: With the use of USB hubs up to 100 Personal Daq units can be attached to one PC. With 100 Personal Daq/56 modules (each with a PDQ2 expansion module) a total channel capability of 8,000 channels can be obtained. An example of capacity calculation follows.

Calculating System Channel Capacity, An Example

Assume a Personal Daq system is comprised of the following:

- one Personal Daq/56 unit
- one PDQ2 expansion module

Using the table on page 2-1 (looking in the bottom row) we see that a Personal Daq/56 with PDQ2 can accept the following types of channel connections:

- 60 *single-ended* (or 30 *differential mode*) for volts inputs
- 30 *differential mode* for thermocouple inputs
- 16 Digital I/O
- 4 Frequency/Pulse inputs

The maximum channel capacity in this set up is 80 channels. With the use of *differential mode* (instead of *single-ended*) the maximum channel capacity is 50 channels.



In Personal Daq applications, thermocouples should not be connected *single-ended*. Doing so can result in noise and false readings. This is especially true when acquiring other high-amplitude signals in conjunction with thermocouple signals that are connected *single-ended*.

Features

The Personal Daq system includes the following features:

- USB connection to the PC means no batteries or other power sources required (see notes)
- signal input connections via removable screw-terminals
- high-resolution 22-bit A/D converter
- internal cold-junction compensator for direct thermocouple measurements
- 500 VDC isolation for PC protection
- low noise thermocouple and voltage measurements
- full-scale voltage inputs from –10 VDC to +20 VDC
- frequency/pulse measurements from DC to 1 Mhz (Does not apply to Personal Daq/54)
- digital I/O with current sink capability for direct drive applications
- with addition of PDQ expansion module, up to 80 channels of analog and digital I/O are available for one Personal Daq/PDQ combined unit (Does not apply to Personal Daq/54)
- up to 100 Personal Daq/PDQ (combined units) can be connected to one PC by the use of USB hubs; providing a total channel capacity of 8,000 channels
- digital calibration eliminates the need for potentiometers and user adjustments
- *Personal DaqView* and driver software
- external power input jack for use with notebook PCs and various setups (discussed in Chapter 3)

Note 1: Certain system configurations make use of USB hubs and power adapters, as discussed in Chapter 3.

Note 2: Some notebook PCs require that a power adapter be used with your Personal Daq. Chapter 3 contains more information regarding power adapters.

Theory of Operation

Universal Serial Bus (USB)

The Universal Serial Bus is ideal for data acquisition applications. Since USB ports (located on the PC) provide power, only one cable is needed to link an acquisition device to the PC. In addition, USB's high-speed data transfer (from the data acquisition device to the PC) allows for real-time display of acquired data while eliminating the need for additional memory in the acquisition device. USB supports transfer rates up to 12 Mbytes/sec and supports real-time data transfer. Standard USB connectors can be identified by a USB icon.

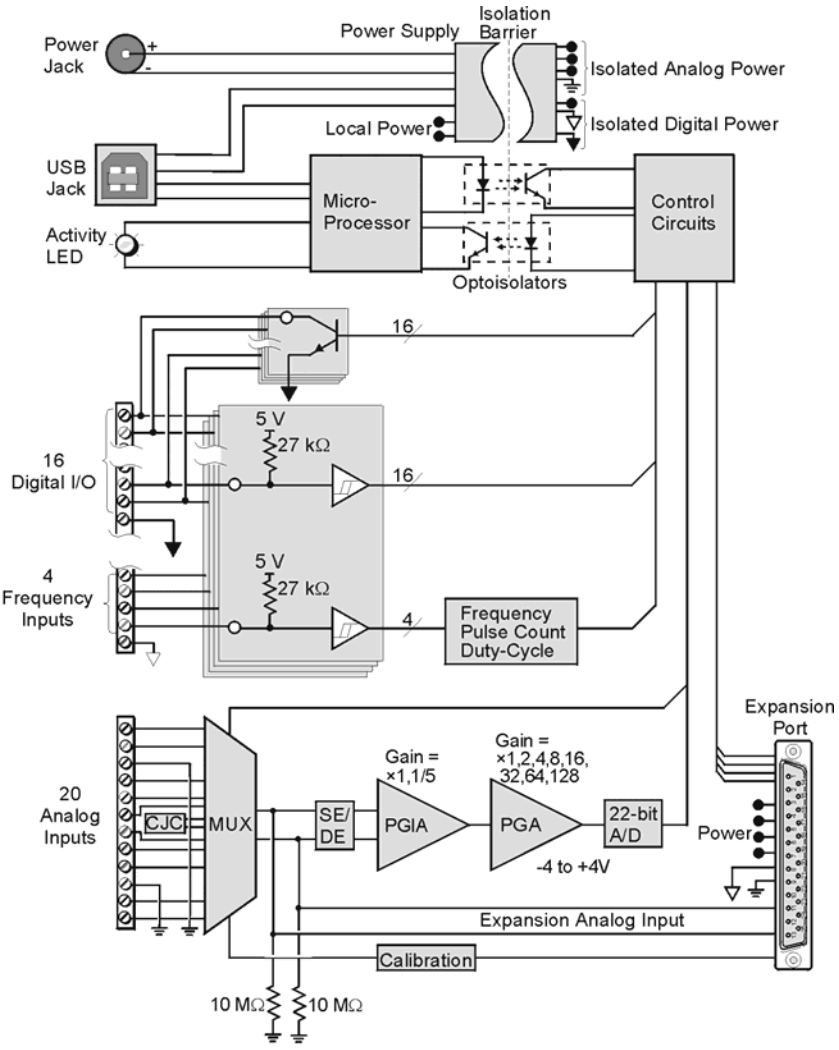


USB Icon

Power Line Rejection

Personal Daq can take readings while making use of 50/60Hz line cycle rejection (120 dB). While in the line cycle rejection mode, the maximum sample rate is as follows:

- 50 Hz rejection: 7.7 samples/sec
- 60 Hz rejection: 9.2 samples/sec



Personal Daq/56 Block Diagram

Optical Isolation

The Personal Daq is optically isolated from its host PC by up to 500 VDC. This means that an inadvertent application of such voltage to Personal Daq will not affect the PC. In addition to optical isolation, Personal Daq maintains all sensitive acquisition-related circuitry external to the PC. This physical isolation of circuitry from the PC results in less noise and more accurate measurements.

A/D Conversion

Personal Daq uses a sigma-delta analog/digital converter to provide high resolution and sensitivity. When scanning multiple input channels, resolution and speed can be selected from 22 bits at 1.6 samples/sec, to 15 bits at 80 samples/sec (not including cold-junction compensation).

You can select resolution and speed on a per-channel basis, allowing you to match parameters to individual channel requirements. For example, you could select 21 bit resolution to detect small temperature changes on one channel, and select 15 bit resolution to measure battery voltage on a second channel. Although each channel can have a different resolution and measurement period, all channels are scanned at the same rate to ensure sampling interval integrity. Examples of scan sequences, with various channel resolutions and calibration arrangements, appear in the figure on page 2-7.

Input Ranges

You can individually select the input range for each channel. For example, one channel could be used for volts and another for temperature. Personal DaqView automatically assigns the appropriate units depending on two factors:

- the selected range and
- measurement unit preferences

Measurement unit preferences can be modified from Personal DaqView's *Preferences Dialog Box* located in the ***View*** pull-down menu of *the Main Control Window*. Chapter 4 provides more detailed information.

Note: The maximum voltage input range (full scale) is –10 to +20 VDC. The lowest programmable voltage input range is -31 to +31 mV. A complete list of Personal Daq's programmable ranges appears on page 2-8.

Each analog input channel has the following user-specified measurement parameters:

- signal type: volts, or thermocouple type J, K, T, E, R, S, B, or N
- full scale voltage: from -10 to + 20 VDC; with programmable ranges as indicated on page 2-8.
- resolution/sample period: from 22 bit RMS at 1.6 samples/sec, to 15 bit RMS at 80 samples/sec (note that these rates were obtained with a 10-channel scan, continuous self-calibration disabled, and are for measurements with no CJC [cold junction compensation])

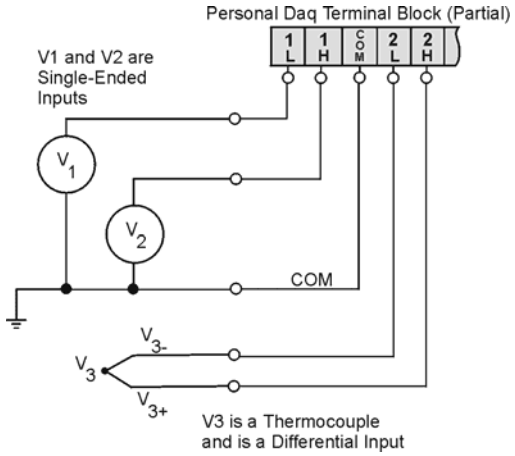
Analog Input Configuration

Personal Daq/54 and **Personal Daq/55** each include 10 analog signal inputs which may be used as 10 single-ended inputs, 5 differential inputs, or as a combination of single-ended and differential inputs with up to 10 connections. **Personal Daq/56** includes 20 analog signal inputs which may be used as 20 single-ended inputs, 10 differential inputs, or as a combination of single-ended and differential inputs with up to 20 connections.

Single-ended inputs are used with signals that share the same common low (COM), such as multiple batteries which have their negative sides connected in common. **Differential** inputs are required when signals do not share the same common low, such as in the typical use of thermocouples. A simple example showing two single-ended inputs (V1 and V2) and a differential input (V3) follows.



In Personal Daq applications, thermocouples should <u>not</u> be connected *single-ended*. Doing so can result in noise and false readings. This is especially true when acquiring other high-amplitude signals in conjunction with thermocouple signals that are connected *single-ended*.



Single-Ended and Differential Connections to Analog Input Channels

For **Personal Daq/55** and **Personal Daq/56**, the number of analog input channels can be expanded with use of a **PDQ** expansion module. There are two types of PDQ expansion modules, either type can be snapped on to a Personal Daq unit; however, each Personal Daq can support only one expansion module.

PDQ1 modules add 20 single-ended (or 10 differential) inputs to the Personal Daq/55 or /56.
PDQ2 modules add 40 single-ended (or 20 differential) inputs to the Personal Daq/55 or /56.

Measurement Duration, Sample Rate, and Resolution

In relation to sampling analog input, the terms *measurement duration*, *sample rate*, and *resolution* have the following meanings:

Measurement duration (per channel) – the amount of time used for sampling a channel's input signal. You can independently set the *measurement duration* for each channel. The *measurement durations* for Personal Daq's analog channels range from very slow (610 milliseconds for one sample) to very fast (12.5 milliseconds for one sample).

Sample rate – Samples per second. The sample rate is the number of samples that take place per second. With the **very slow** *measurement duration* of 610 milliseconds, there will only be 1.6 samples per second. With the **very fast** *measurement duration* of 12.5 milliseconds, there will be 80 samples per second.

Resolution (Bit RMS) – The number of reliable data bits that exist for a signal's measurement. The greater the resolution, the more *detailed* the reading, for example, with increased resolution a reading of 5.12 V could become 5.11896 V. Personal Daq actually provides for 24 bits of data information; however, the accuracy of the least significant bits becomes less as the measurement duration speeds up.

At a *measurement duration* of 610 milliseconds, the last two bits are considered unreliable, resulting in a resolution of 22 bits. At a very fast *measurement duration* (12.5 milliseconds), the nine *most least significant bits* are unreliable, resulting in 15 bit accuracy.

Speed vs. Resolution[1]			
Speed Designation	**Measurement Duration (per channel)**	**Maximum Sample Rate[2] (Samples/sec)**	**Resolution[2] (Bits RMS) (-4 V to +4 V range)**
Very Slow, 50 / 60 Hz rejection	610 ms	1.6 / sec	22
Slow, 50 Hz rejection	370 ms	2.7 / sec	22
Slow, 60 Hz rejection	310 ms	3.2 / sec	22
Medium, 50 Hz rejection	130 ms	7.7 / sec	21
Medium, 60 Hz rejection	110 ms	9.2 / sec	21
Medium	40 ms	25 / sec	19
Fast	20 ms	48 / sec	17
Very Fast	12.5 ms	80 / sec	15

Notes:
1. Each channel can have independent measurement duration and resolution.
2. The sample rates and resolutions shown were obtained with a 10-channel scan and with continuous self-calibration disabled.
3. Duration does not include the use of CJC measurements.

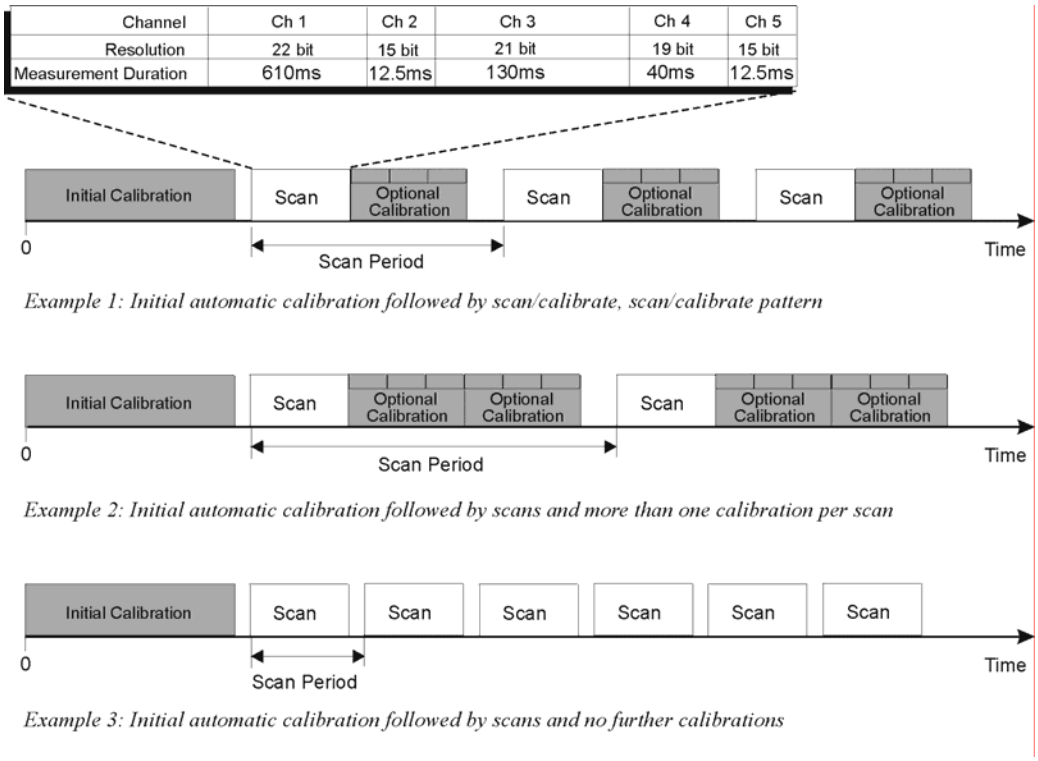
Note: When measuring variable input signals (as opposed to relatively steady input signals), the variable signals will require more samples/sec to obtain a realistic signal representation. Available sample rates range from 1.6 samples per second up to 80 samples per second as indicated in the preceding table.

Automatic Calibration

The Personal Daq module contains a built-in source for performing automatic self-calibrations. These calibrations can be performed between scans periodically throughout the measurement process, as indicated in the following figure. Such calibration ensures accurate measurements, even in environments that experience significant temperature fluctuations.

With exception of an automatic initial calibration, Personal Daq's *self calibrations* are optional, and may be discontinued (automatically) if the maximum sampling rate is used. This is because the time between scans can become too short to permit calibration. In this instance, the PC can initiate calibration immediately prior to the measurement process (see example 3 in the following figure).

Note: The continuous calibration feature is selected (or deselected) from the *Configure Acquisition dialog box*. See Chapter 4 for more information.



Three Examples of Calibration/Scan Arrangements

Thermocouple Measurements

Personal Daq provides effortless thermocouple (TC) measurements. The unit includes built-in cold-junction compensation (CJC), which is automatically invoked when you select TC measurements. The Personal Daq automatically converts acquired voltage readings into compensated-linearized temperature readings. A Personal Daq system can make thermocouple and volts measurements concurrently.

Frequency Measurements (Not applicable to Personal Daq/54)

Each frequency/pulse input channel can measure from DC to 1 MHz, offering pulse count (totalize), frequency, and duty cycle type readings. The input voltage range is -15 to +15 VDC absolute maximum. TTL sense levels, Schmitt-trigger inputs <1.3 V (low), >3.8 V (high).

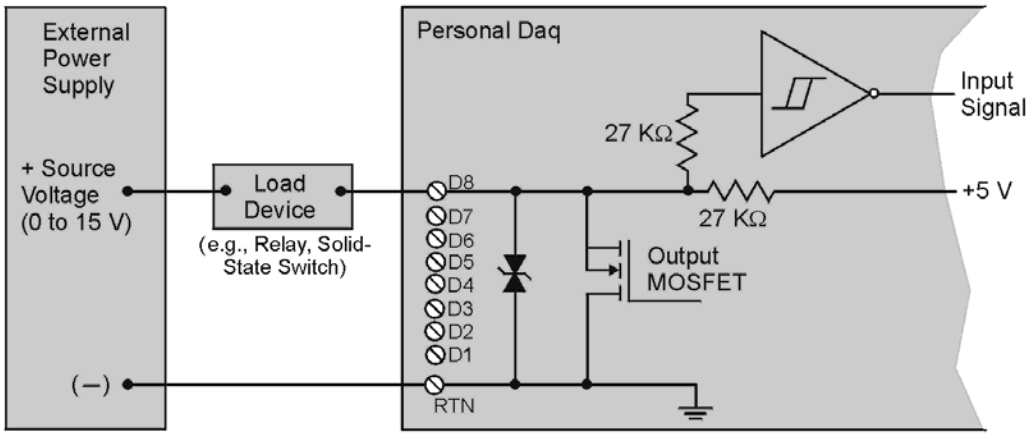
Note that each frequency channel can measure pulses that are due to closures of a contact between the input terminal and the Lo terminal. This is because an internal pull-up resistor places the input at +5V when the contact is open and allows the input to go to 0V when the contact is closed.

Frequency and pulse measurements can be scanned along with analog input measurements with resolution up to 7 digits.

Digital I/O (Not applicable to Personal Daq/54)

You can program each digital I/O line individually as input or output. Digital input lines can be read as part of the analog scan sequence and can be scanned over 100 times per second. Digital output lines can be updated at any time prior to, or during an acquisition sequence; or can be automatically updated.

Digital input voltage ranges from 0 to +15 VDC are permitted, with thresholds of: <1.3V (low) and >3.8V (high). In the output mode each output is an *open-collector*, with each bank of eight outputs capable of sinking up to 150 mA for direct drive applications. External pull-up voltages can be up to +15 VDC (Over-voltage is rated at +20 VDC for up to 1 minute). Refer to the following figure for information regarding Digital I/O connection.



Personal Daq, Digital Output Connections

Note: The figure represents connections for one channel (D8). The other seven channels (D1 … D7) are connected in the same manner.

Note*:* The maximum load current is 150 mA per 8-channel bank, or if all eight channels are used, 18 mA (max) per channel.

Personal Daq Specifications

Specifications are subject to change without notice.

CAUTION



If equipment is used in any manner not specified in this manual, or if specification limits are exceeded, the function of the equipment, as well as the protection provided by it, may be impaired.

Channel Capacities for Various Personal Daq Setups

Personal Daq/54	Volts Inputs	TC Inputs	Digital I/O	Freq/Pulse Inputs
Personal Daq/54 ---- *No Expansion* ----	5 DE, or 10 SE	5 DE	---- N/A ----	---- N/A ----
Personal Daq/55 Systems	**Volts Inputs**	**TC Inputs**	**Digital I/O**	**Freq/Pulse Inputs**
Personal Daq/55	5 DE, or 10 SE	5 DE	8	2
Personal Daq/55 with PDQ1	15 DE, or 30 SE	15 DE	24	2
Personal Daq/55 with PDQ2	25 DE, or 50 SE	25 DE	8	2
Personal Daq/56 Systems	**Volts Inputs**	**TC Inputs**	**Digital I/O**	**Freq/Pulse Inputs**
Personal Daq/56	10 DE, or 20 SE	10 DE	16	4
Personal Daq/56 with PDQ1	20 DE, or 40 SE	20 DE	32	4
Personal Daq/56 with PDQ2	30 DE, or 60 SE	30 DE	16	4

DE = Differential Mode, SE = Single-Ended Mode

Speed vs. Resolution[1]

Speed Designation	Measurement Duration (per channel)	Maximum Sample Rate[2] (Samples/sec)	Resolution[2] (Bits RMS) (-4 V to +4 V range)
Very Slow, 50 / 60 Hz rejection	610 ms	1.6 / sec	22
Slow, 50 Hz rejection	370 ms	2.7 / sec	22
Slow, 60 Hz rejection	310 ms	3.2 / sec	22
Medium, 50 Hz rejection	130 ms	7.7 / sec	21
Medium, 60 Hz rejection	110 ms	9.2 / sec	21
Medium	40 ms	25 / sec	19
Fast	20 ms	48 / sec	17
Very Fast	12.5 ms	80/ sec	15

Notes:
1. Each channel can have independent measurement duration and resolution.
2. The sample rates and resolutions shown were obtained with a 10-channel scan and with continuous self-calibration disabled.
3. Duration does not include the use of CJC measurements.

Analog Specifications

Each channel can be individually configured for single ended or differential; volts or thermocouple inputs.

Personal Daq/54: configurable for 10 single-ended, 5 differential; volts or TC channels
Personal Daq/54 does not accept expansion modules.
Personal Daq/55: configurable for 10 single-ended, 5 differential; volts or TC channels
Personal Daq/56: configurable for 20 single-ended, 10 differential; volts or TC channels
PDQ1 Expansion Module: configurable for 20 single-ended, 10 differential; volts or TC channels
PDQ2 Expansion Module: configurable for 40 single-ended, 20 differential; volts or TC channels
Input Voltage Range Relative to Analog Common (COM): -10 to +20 VDC

Input Voltage Ranges

Programmable Voltage Ranges	RMS Noise (µV) typical*					
	Very Slow	Slow	Medium 50, 60 Hz Rejection	Medium	Fast	Very Fast
-10 V to +20 V (Single-ended only)	120	120	120	120	270	1600
-20 V to +20 V (Differential only)	120	120	120	120	270	1600
-10 V to +10 V	35	35	35	75	190	740
-5 V to +5 V	7	7	10	60	95	370
-4 V to +4 V	4	4	5	20	60	340
-2.5 V to +2.5 V	4	5	8	55	75	200
-2 V to +2 V	4	4	4	15	30	150
-1.25 V to +1.25 V	2	2	3	9	20	110
-1 V to +1 V	2	2	3	15	20	75
-625 mV to +625 mV	3.5	3.5	4.5	10	15	50
-500 mV to +500 mV	1	1.5	2	15	15	40
-312 mV to +312 mV	3	3	4	8	10	30
-250 mV to +250 mV	1	1	2	8	8	25
-156 mV to +156 mV	2.5	4	4	8	8	20
-125 mV to +125 mV	1	1	1.5	7	7	20
-62 mV to +62 mV	1	1	1.5	6	5	9
-31 mV to +31 mV	1	1	1.5	6	5	7

***Note:** Noise measured with continuous self-calibration disabled.

Voltage Specifications (one year, 15 to 35°C)

Accuracy:
Personal Daq/54: 0.015% of reading + .004% of range (exclusive of noise)
Personal Daq/55: 0.01% of reading + .002% of range (exclusive of noise)
Personal Daq/56: 0.01% of reading + .002% of range (exclusive of noise)

Input Offset Voltage:
Personal Daq/54: <30µV (differential or single-ended)
Personal Daq/54: <20µV (differential or single-ended)
Personal Daq/54: <20µV (differential or single-ended)

Peak-to-Peak Noise: 6 x RMS Noise

Temperature Coefficient: (5 ppm + 1 µV)/°C

Input Resistance: 10MΩ (single ended); 20MΩ (differential); ±5%

Bias Current: <1 nA (0 to 35°C)

DC Common Mode Gain Error: <100 ppm typical; <2 ppm/°C common mode gain drift typical

AC Common Mode Rejection: >120 dB @ 50 Hz (610 ms, 370 ms, 130 ms measurement durations);
>120 dB @ 60 Hz (610 ms, 310 ms, 110 ms measurement durations)

AC Normal Mode Rejection: >80 dB @ 50 Hz (610 ms, 370 ms, 130 ms measurement durations);
>80 dB @ 60 Hz (610 ms, 310 ms, 110 ms measurement durations)

Channel-to-Channel Cross Talk: < -110 dB (DC to 100 Hz; up to 10 kΩ source resistance)

Over-Voltage Protection: ±45 V relative to analog common

Temperature Specifications (one year, 15 to 35°C)

Note: All temperature specifications assume unit is held in relatively still air environment.

Thermocouple Types: J, K, T, E, R, S, B, N
Cold-Junction Compensation Accuracy:

Personal Daq/54: ±0.7°C
Personal Daq/55: ±0.5°C
Personal Daq/56: ±0.5°C

Thermocouple Accuracy



In Personal Daq applications, thermocouples should <u>not</u> be connected *single-ended*. Doing so can result in noise and false readings. This is especially true when acquiring other high-amplitude signals in conjunction with thermocouple signals that are connected *single-ended*.

Thermocouple Accuracy (°C) for Personal Daq/54, /55, and /56 *							
TC Type	**Temp. (°C)**	**Very Slow**	**Slow**	**Medium 50, 60 Hz Rejection**	**Medium**	**Fast**	**Very Fast**
J	-100	0.4	0.4	0.6	1.3	1.3	3.8
	0	0.3	0.3	0.4	1.1	1.1	3
	700	0.3	0.3	0.4	0.9	0.9	2.5
K	-100	0.5	0.5	0.7	1.8	1.8	5
	0	0.4	0.4	0.6	1.4	1.4	3.9
	600	0.4	0.4	0.6	1.3	1.3	3.7
T	-50	0.4	0.4	0.7	1.6	1.6	4.5
	0	0.4	0.4	0.6	1.4	1.4	4
	200	0.3	0.3	0.4	1	1	2.9
E	-100	0.3	0.3	0.5	1.2	1.2	3.4
	0	0.3	0.3	0.4	0.9	0.9	2.6
	500	0.2	0.2	0.3	0.7	0.7	1.9
R	400	1.5	1.5	2	5.2	5.2	14.8
	700	1.3	1.3	2	4.6	4.6	13.1
	1400	1.2	1.2	1.7	3.9	3.9	11
S	400	1.6	1.6	2	5.6	5.6	16
	700	1.5	1.5	2	5.2	5.2	14.7
	1400	1.3	1.3	2	4.6	4.6	12.8
B	700	2.2	2.2	3.3	7.9	7.9	13.7
	1400	1.4	1.4	2	4.8	4.8	13.7
N	-100	0.7	0.7	1	2.6	2.6	7.3
	0	0.6	0.6	0.8	2	2	5.9
	700	0.4	0.4	0.6	1.4	1.4	4
***Note**: Thermocouple accuracy excludes cold junction compensation error.							

Frequency Specifications, one year 0 to 50°C (Not applicable to Personal Daq/54)

Operating Modes: Pulse count (totalize), frequency, and duty cycle

Frequency Response: DC to 1 MHz

Accuracy: 100 ppm; 1 ppm/°C

Resolution: Up to 7 digits, user selectable.

Input Voltage Range: -15 to +15 VDC Absolute Maximum, TTL sense levels
Schmitt-trigger inputs, <1.3 V (low), >3.8 V (high)

Pull-up Resistor: 27 KΩ to +5 V for switch or relay sensing

Debouncing: None, 0.8, 3.2, or 13 milliseconds (software selectable)

Totalize: Up to 2^{32} counts/scan

Digital I/O Specifications (Not applicable to Personal Daq/54)

Configuration: Each I/O line is individually selectable as input or output. Each I/O line includes an open-collector driver with a 27 KΩ pull-up resistor to +5 V for output, and a Schmitt-trigger input buffer.

Over-Voltage: +20 VDC for up to 1 minute

Output Characteristics:

Voltage Range: 0 to +5VDC with no external pull up resistor; 0 to +15VDC with external pull up

Maximum Sink Current: 150 mA/output continuous, 500 mA output peak (<100 µs),
150 mA total continuous (per bank of 8 outputs)

Output Resistance: 10 Ω max

Output Updates: Outputs may be changed via program control

Input Characteristics:

Voltage Range: 0 to +15VDC

Thresholds: <1.3 V (low), > 3.8 V (high)

General Specifications

Warm-up: 1 hour to rated specifications

Environment:

Operating: 0-50°C, 0-95% RH (non-condensing)
Storage: -20 to 70°C

Isolation: 500 VDC from PC common

USB Power Source: PC USB port or *self-powered* USB hub; 500 mA maximum

External Power Source: Required when used with *a bus-powered* hub; +6 to +16 VDC, 500 mA
Option Notes: (1) the TR-2 External Power Supply is a 120 VAC to +9 VDC adapter;
(2) the TR-2E External Power Supply is a 230 VAC to +9 VDC adapter

Vibration: MIL Std 810E

Dimensions: 92 x 182 x 45 mm (3.6 x 7.1 x 1.8 inches)

Weight: Personal Daq/55: 300g (10.5 oz)
Personal Daq/54: 300g (10.5 oz)
Personal Daq/56: 360g (12.5 oz)
PDQ1: 300g (10.5 oz)
PDQ2: 300g (10.5 oz)

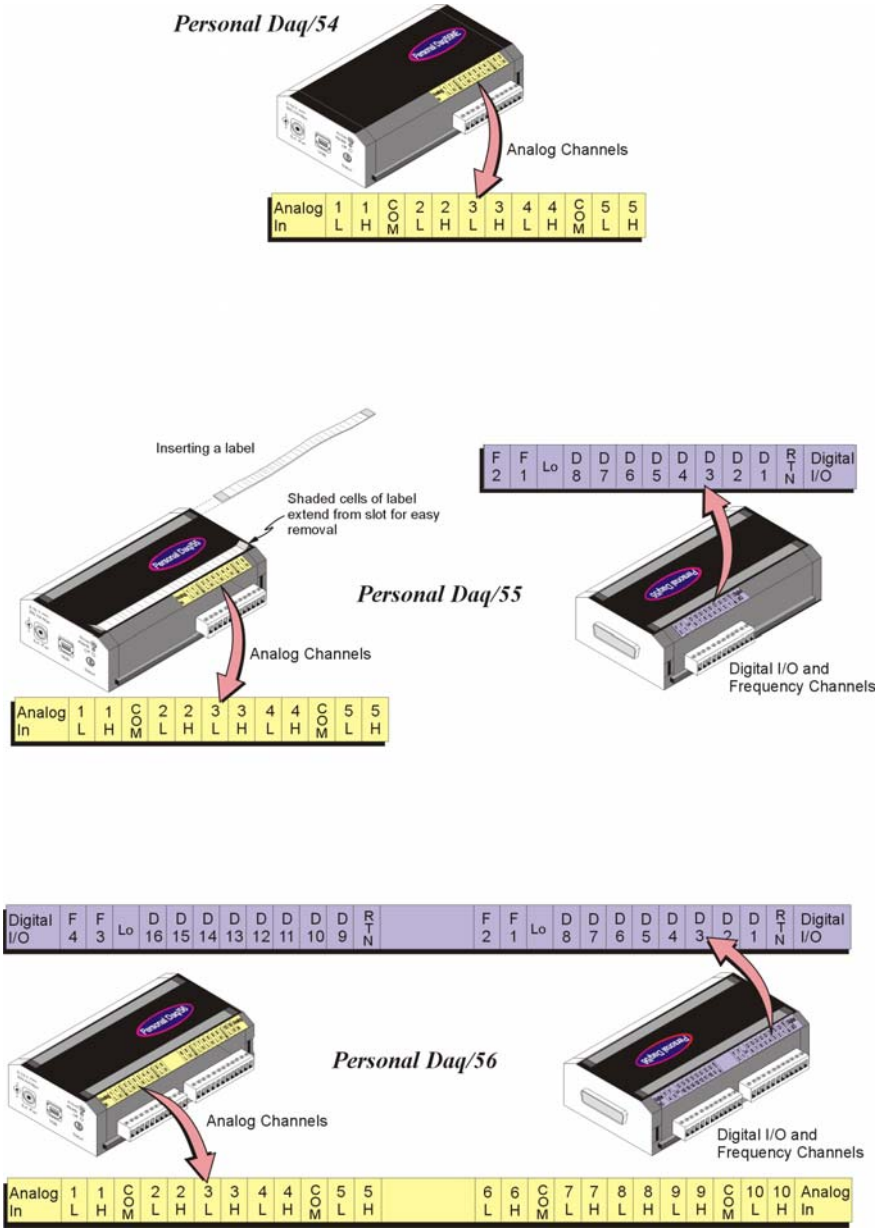
Optional Accessories

PDQ10 – DIN-Rail Mounting
PDQ11 – 4 powered USB Hub with cable
PDQ12 – USB extender cable, 5 meters
PDQ13 – PCI to dual USB card
TR-2 – External Power Supply, 120 VAC to +9 VDC adapter
TR-2E – External Power Supply, 230 VAC to +9 VDC adapter
CN-153-12 – Terminal Block

USB Cables: CA-179-3 (3 meter)
CA-179-5 (5 meter)

Channel Connection Layouts

The following indicate channel connection layouts for Personal Daq/54, /55, and /56. Connection layouts for expansion modules (PDQ1 and PDQ2) follow shortly. **Personal Daq/54 provides no functionality for expansion modules**.



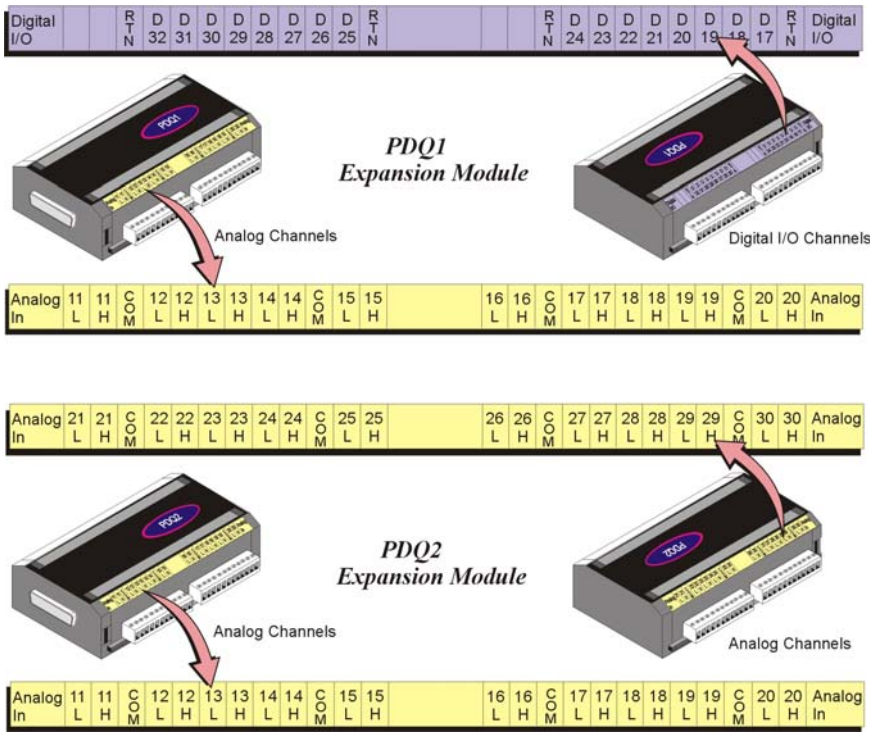
Analog In *Single-ended* inputs are used with analog signals that share the same common low, such as multiple batteries which have their negative sides connected in common. *Differential* inputs are required when signals do not share the same common low, such as in the typical use of thermocouples. Note that the analog low common references are -10 to +20 Volts. Analog low commons (COM) are located on the same terminal blocks as are the analog channel connections, and should not be confused with the frequency lows (Lo) discussed later.

In Personal Daq applications, thermocouples should <u>not</u> be connected *single-ended*. Doing so can result in noise and false readings. This is especially true when acquiring other high-amplitude signals in conjunction with thermocouple signals that are connected *single-ended*.

Personal Daq units include built-in cold-junction compensation (CJC), which is automatically invoked when you select TC measurements. The Personal Daq automatically converts acquired voltage readings into compensated-linearized temperature readings. A Personal Daq system can make thermocouple and volts measurements concurrently.

<u>Frequency/pulse</u> These input channels can measure from DC to 1MHz. Readings can be in pulse count (totalize), frequency, or duty cycle. Input voltage range is –15 to +15VDC, with the logic threshold at <1.3V (low) and >3.8V (high). Frequency and pulse measurements can be scanned along with analog input measurements with resolution up to 7 digits. Frequency low connections (Lo) are not to be confused with the analog commons (COM). Frequency Los are located next to F1 and F3 channel connections. The connection points labeled "Lo" serve as a common reference for frequency inputs. Frequency/pulse channels do not apply to Personal Daq/54.

<u>Digital I/O</u> You can program each digital I/O line individually as input or output. The digital I/O lines <u>do</u> <u>not</u> make use of the Lo or COM connections. Digital output lines do make use of special digital return lines designated by the letters RTN. Digital I/O channels do not apply to Personal Daq/54.



Digital input lines can be read as part of the analog scan sequence and can be scanned over 100 times per second. Digital input voltage may range from 0 to +15V, with the logic threshold at <1.3V (low) and >3.8V (high). Digital values outside of the 0 to +15V range are not permitted.

Digital output lines can be updated at any time prior to, or during an acquisition sequence; or can be automatically updated over 100 times per second. Digital outputs share a common return (RTN). In the output mode each output is an open-collector capable of sinking up to 150mA for direct drive applications, and capable of pull-up voltages up to +15 VDC. Note that over-voltage is rated at +20 VDC for up to 1 minute.

Note: The digital output can be written to since the digital output lines are latched.

Note: Appendix D contains blank user custom labels and information regarding **pDaq_CustomLabels.doc** (located in the installation target directory of **\\Program Files\pDaqView**). User custom labels allow you to identify Personal Daq channels by user-specific nomenclature, in addition to the pre-existing channel labels (indicated on this and the preceding page).

Calibration

Calibration must be completed periodically to ensure your data acquisition device remains accurate. You can use *UserCal* (a *Windows*-based program) to provide step-by-step instructions. Chapter 6 contains detailed information regarding calibration.

Note: Calibration constants are calculated and stored in the Personal Daq unit's serial EEPROM.

Note: The typical calibration period for Personal Daq is once every year.

This page is intentionally blank.

Hardware Setup 3

Personal Daq, System Components ……3-2

Personal Daq …… 3-2
PDQ Expansion Modules …… 3-2
USB Hubs and Power Adapters …… 3-3

Connecting Your Personal Daq Acquisition System …… 3-4

Connecting a PDQ Expansion Module to a Personal Daq …… 3-4
Connecting Various Hardware Setups …… 3-4

Example 1: Direct Connection to Computer USB Port(s) …… 3-4
Example 2: Connection to USB-Powered Hub …… 3-5
Example 3: Connections to Self-Powered and USB-Powered Hubs …… 3-5

CAUTION



The discharge of static electricity can damage some electronic components. Semiconductor devices are especially susceptible to ESD damage. You should always handle components carefully, and you should never touch connector pins or circuit components unless you are following ESD guidelines in an appropriate ESD controlled area. Such guidelines include the use of properly grounded mats and wrist straps, ESD bags and cartons, and related procedures.

CAUTION



Never connect an expansion module to (or remove it from) a Personal Daq main unit while the main unit is connected to a power source. Such action may result in EEPROM errors and loss of calibration data.

CAUTION



Never remove a USB cable from an active Personal Daq device while an acquisition is in progress. An active device is any device that is currently open and has channels configured for scanned input. Such disconnection may require you to exit and then re-launch Personal DaqView, after the USB cable has been connected.

CAUTION



When using Personal Daq modules to acquire data, computer *energy save* modes can cause false data readings. Prior to using Personal Daq modules, ensure your computer's *energy save* mode is disabled. If needed, consult your PC user's manual to disable *energy save* (*power suspension*) modes.



When using a power adapter with your Personal Daq system, be sure to supply power (from the adapter to the Personal Daq) before connecting the USB cable. This allows Personal Daq to inform the host computer (upon connection of the USB cable) that the unit requires minimal power from the computer.

Note: This chapter pertains to hardware aspects of the Personal Daq, including the PDQ expansion modules and two basic types of USB hubs (*self-powered* and *USB-powered*). Product specifications are contained in the previous chapter.

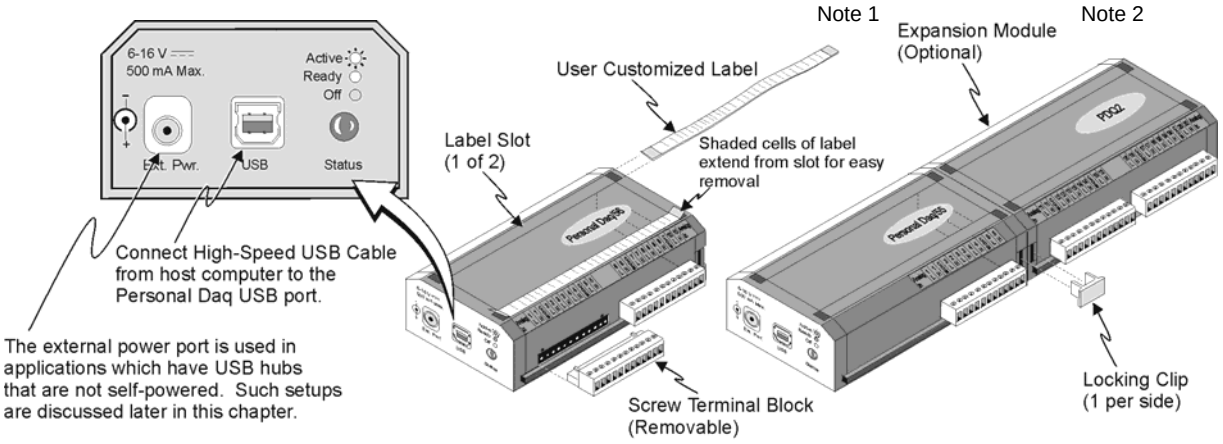
Personal Daq, System Components

Personal Daq

The following illustration points out key physical features of the Personal Daq. Aside from labeling and the number of terminal blocks, Personal Daq/54, /55, and /56 look very much alike. Upon closer examination you can see differences.

Both the /55 and /56 contain a DB25 expansion port for mating with a PDQ expansion module. Slots on the main units and expansion modules provide a means of attaching units securely with locking clips. Personal Daq/54 also has a DB25 port. However, in the case of the Daq/54 the port is used only for calibration as discussed in chapter 6. Personal Daq/54 contains no functionality in regard to expansion modules. Also note that Personal Daq/54 has no terminal block for Digital I/O and no Frequency/pulse input terminals.

A customized label can be inserted in slots above the terminal connectors. The labels are especially useful in acquisition systems that consist of several Personal Daqs and PDQ expansion modules. Appendix D contains blank labels.



Note 1: Label slots are included on Personal Daq/55 and /56. The slots are not included on Personal Daq/54.

Note 2: An expansion module (PDQ1 or PDQ2) can be added to Personal Daq/55 and /56 units. Although an expansion module can physically connect to a Personal Daq/54, it will not function with the unit. Personal Daq/54 does not include expansion functionality. For /54 devices, the 25-pin port is used only for calibration as discussed in chapter 6.

Note 3: "**Active**" [LED blinking] means that an acquisition is in progress. "**Ready**" [LED on solid] means that the unit has power, has USB communications established, and is ready to acquire data.

PDQ Expansion Modules (Does not apply to Personal Daq/54)

You can connect either of two PDQ expansion modules (PDQ1 or PDQ2) to the Personal/55 and Personal/56 units. Module specifications are listed in the preceding chapter.

With exception of the end-faces, the PDQ modules have the same physical appearance as Personal Daq/56. The above figure includes an illustration of a PDQ2 module connected to a Personal Daq/55. Two locking clips are used to hold the units together. The clips ensure a good connection is maintained.

Personal Daq/54 does not accept expansion modules.

USB Hubs and Power Adapters

With the use of USB hubs you can connect up to 100 Personal Daq units to one PC. USB hubs can be of the *self-powered* type, or of the *USB-powered* type. Both types of hubs are available from a variety of vendors; however, if you encounter any difficulty in obtaining a USB hub, please contact your service representative.

Power Adapters – Power adapters, also referred to as auxiliary power packs, are required for some *self-powered hubs*, and for Personal Daq modules that are powered from *USB-powered hubs*. In addition, Personal Daq units will require the use of a power adapter when used with certain laptops.



When using a power adapter with your Personal Daq system, be sure to supply power (from the adapter to the Personal Daq) before connecting the USB cable. This allows Personal Daq to inform the host computer (upon connection of the USB cable) that the unit requires minimal power from the computer.

Power adapters for use with Personal Daq have a current limit of 500 mA (min.) and a voltage range of +6 to +16 Volts DC. These specifications are provided on the end-face of the Personal Daq.

If the computer does not recognize the Personal Daq unit, make sure the computer's USB port is properly enabled and is in good working order. If the computer still fails to recognize the Personal Daq, the use of a power pack may be required. In United States use a TR-2 power pack, or equivalent. In Europe use a TR-2E power pack, or equivalent. Both the TR-2 and the TR-2E provide 500 mA min, +6 to +16 VDC.



The use of certain notebook computers may require the use of a power adapter with your Personal Daq.

USB-powered Hubs – These hubs draw all power from the host USB connector's power pins. The power is used for hub internal functions and for the hub's ports. Each port of a *USB-powered hub* must be capable of supplying at least 100 mA.

Self-powered Hubs – These hubs draw power from a source other than the host USB connector, with exception that they may draw up to 100 mA from their upstream connection for hub internal functions. The external power is used for hub internal functions and for the hub's ports. Each port of a *self-powered hub* must be capable of supplying 500 mA.

Connecting Your Personal Daq Acquisition System



Review the CAUTIONS and notes (presented on page 3-1) prior to connecting or disconnecting components.

Connecting a PDQ Expansion Module to a Personal Daq/55 or /56

To connect a PDQ expansion module to a Personal Daq/55 or /56 unit:

1. **Review the CAUTIONS and notes presented on page 3-1**.
2. Ensure the Personal Daq main unit is not connected to a USB port.
3. Ensure the Personal Daq main unit is not connected to a power adapter.
4. Plug the expansion module into the DB25 connector on the Personal Daq main unit.
5. Lock the two modules together using two locking clips (see figure, page 3-2).
6. Connect the Personal Daq main unit to power according to your system setup (3 examples follow).



When using a power adapter with your Personal Daq system, be sure to supply power (from the adapter to the Personal Daq) before connecting the USB cable. This allows Personal Daq to inform the host computer (upon connection of the USB cable) that the unit requires minimal power from the computer.

Connecting Various Hardware Setups

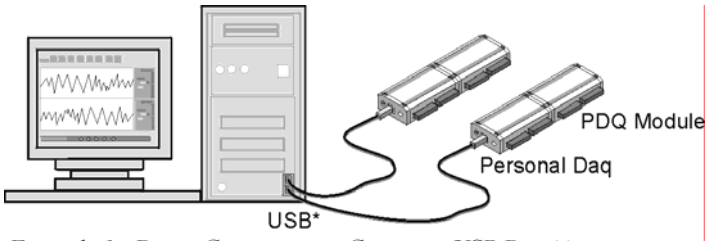
Personal Daq data acquisition systems range from simple to complex. One example of a simple system is that of one Personal Daq/54 connected to a PC's USB connector. A much more complex system is one that contains 100 Personal Daq/56 units, 100 PDQ expansion modules, and a combination of *USB-powered* and *self-powered hubs.* Despite the wide range of possibilities in between, use of the following examples should enable you to properly connect your system.

Note: In the examples that follow, the USB hubs have four external ports (downstream ports). The USB hubs used in your system may have more. Connections can be adjusted accordingly.



When using a power adapter with your Personal Daq system, be sure to supply power (from the adapter to the Personal Daq) before connecting the USB cable. This allows Personal Daq to inform the host computer (upon connection of the USB cable) that the unit requires minimal power from the computer.

Example 1: Direct Connection to Computer USB Port(s)

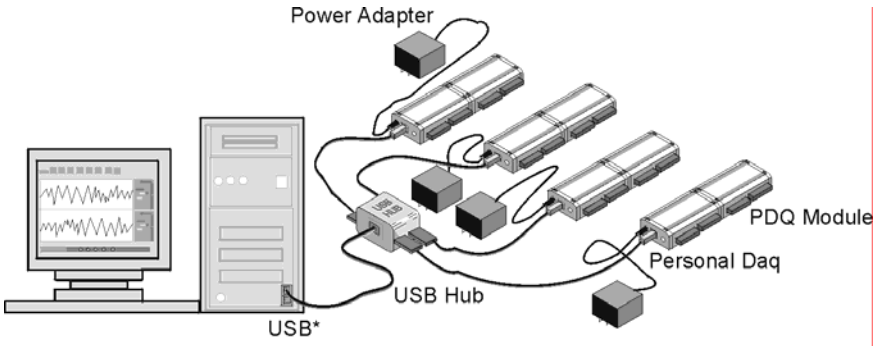


Example 1: Direct Connection to Computer USB Port(s)

In example 1 two Personal Daqs (and optional PDQ modules) are connected by cable to each of the computer's USB ports. The number of USB connectors may vary from PC to PC. When you connect Personal Daq units directly to a USB connector in this manner, no additional power source is required since the computer's USB connector power pins supply the Personal Daq and associated PDQ expansion module with adequate power (500 mA at 4 to 5.25 V).

The use of certain notebook computers may require the use of a power adapter with your Personal Daq.

Example 2: Connection to USB-Powered Hub



Example 2: Connection to USB-Powered Hub

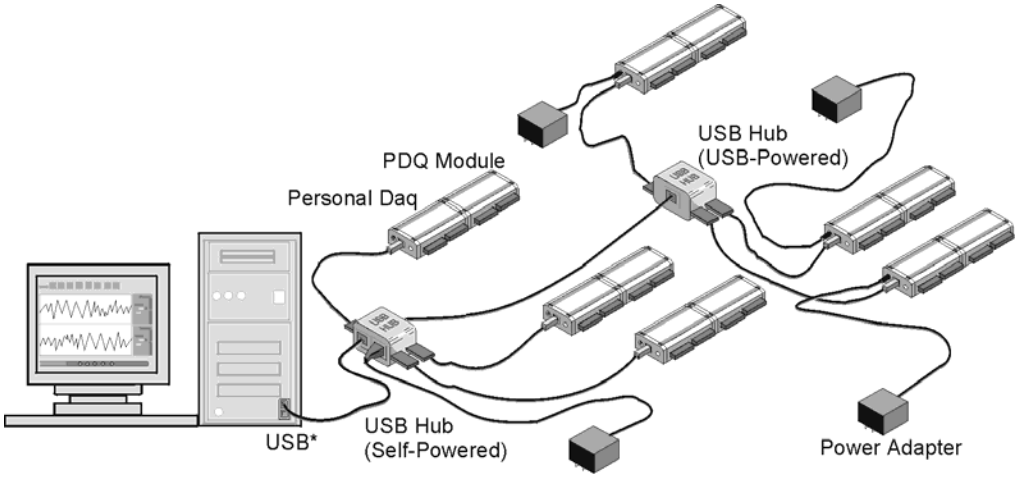
In example 2, four Personal Daqs (and optional PDQ modules) are connected by cable to individual ports of a single *USB-powered hub*. Since the hub receives all its power from the computer's USB, the hub cannot supply adequate power to the Personal Daq units. Because of this aspect of insufficient power, each Personal Daq is connected to its own power adapter.

Note: The power adapters used must be capable of supplying at least 500 mA and have a voltage rating of +6 to +16 VDC.

***Note:** USB port locations vary from PC to PC.

When using a power adapter with your Personal Daq system, be sure to supply power (from the adapter to the Personal Daq) before connecting the USB cable. This allows Personal Daq to inform the host computer (upon connection of the USB cable) that the unit requires minimal power from the computer.

Example 3: Connections to Self-Powered and USB-Powered Hubs



Example 3: Connections to Self-Powered Hub and to USB-Powered Hub

Example 3 illustrates a system that makes use of six Personal Daq units and two different style USB hubs. Three Personal Daqs (and optional PDQ modules) are connected by cable to individual ports of a *self-powered USB hub*. In addition, the *self-powered hub* is connected to a downstream *USB-powered hub*, which is also connected to three Personal Daq units (and optional PDQ modules).

Notice that the Personal Daqs connected to the *self-powered hub* have no adapters connected to them. This is because the hub receives external power (in addition to the PC supplied USB power), which is capable of supporting the downstream devices connected directly to it. In comparison, the three Personal Daqs connected to the *USB-powered hub* each require their own power adapter. As in example 2, the power adapters used must be capable of supplying at least 500 mA and have a voltage rating of +6 to +16 VDC.

***Note:** USB port locations vary from PC to PC.

Personal DaqView 4

Overview …… 4-2

Main Control Window …… 4-2

Toolbar Buttons …… 4-2
Pull-Down Menus …… 4-3

Channel Configuration Window …… 4-6

Channel Configuration Window Toolbar …… 4-6
Channel Configuration Window Pull-down Menus …… 4-6
Common Spreadsheet Columns …… 4-7
Analog Input Spreadsheet …… 4-10
Frequency/Pulse Input Spreadsheet …… 4-12
Digital Input/Output Spreadsheet …… 4-14
Configure Acquisition Dialog Box …… 4-15
Configure Data Destination and File Converter Preferences …… 4-18
Sequential Destinations (Auto Rearm) …… 4-19

Bar Graph, Analog, and Digital Meters …… 4-20

Meter Toolbars …… 4-20
Meter Pull-Down Menus …… 4-21
Meters Configuration Menu …… 4-21
Configuring a Meter …… 4-22
Bar Graph Meters …… 4-24
Analog Meters …… 4-25
Digital Meters …… 4-26

Chart Display …… 4-27

Groups, Charts, & Channels …… 4-27
Chart Display Window …… 4-27
Pull-Down Menus …… 4-28
Toolbar Items …… 4-29
Chart and Channel Information Regions …… 4-29
Accessing the Display Configuration Setup Box …… 4-31
Editing a Chart Display Configuration …..4-31
Manually Configuring a Chart Display …. 4-33

Chart Setup Wizard …… 4-36

Introduction …… 4-36
Automatic Display Setup using the *Chart Setup Wizard* …… 4-38
Bypassing Automatic Chart Setup …… 4-38

Using Personal DaqView with Microsoft Excel™ …… 4-39



Reference Note: This chapter serves as a reference for *Personal DaqView* and the *Chart Setup Wizard* feature. **For very first time start-up** refer to Chapter 1 or the *Personal Daq Quick Start* document (p/n 491-0940). Chapter 1 and the Quick Start contain information for connecting hardware, loading software, and acquiring data quickly.

Overview

Personal DaqView is a graphic Microsoft Windows-based program that can be used for various data acquisition applications. The program was designed for *ease-of-use* with no need for programming *or expertise in configuration.*

Personal DaqView includes the following features:

- Set up Analog Input Channels for acquiring various voltages or temperature
- Set up Frequency/Pulse Input Channels to measure various frequency-related parameters, e.g., pulse rate, total pulses, pulses per scan, percentage high, percentage low
- Set up Digital I/O Channels for input or output
- Save acquired data to disk
- Transfer data to spreadsheets, data bases, and PostView
- View real-time data values in screen columns, scrolling charts, bar graph, analog, or digital meters
- Support for up to 100 Personal Daq devices attached to one PC
- *Chart Setup Wizard* feature you can make use of Simple, Moderate or Advanced automatic chart creation functions
- Includes "add-ins" to *Microsoft Excel*™ and can be run from within the *Excel* environment, using *Excel's* associated macros

Personal DaqView permits the use of multiple groups with up to four overlapping channels per chart. In addition, the application allows you to make changes via a Chart Properties dialog box. The Chart Properties box is discussed on page 4-28.

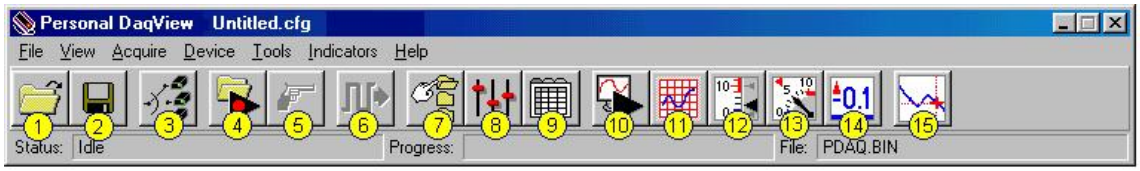
When using the *Chart Setup Wizard* feature you can make use of Simple, Moderate, or Advanced automatic chart creation functions of the wizard. Discussion of the *Chart Setup Wizard* begins on page 4-36.

Main Control Window

When you first open Personal DaqView the *Main Control and Channel Configuration Windows* appear. Note that this section pertains to the *Main Control Window* only; discussion of the *Channel Configuration Window* begins on page 4-6.

The *Main Control Window* contains several pull-down menus, a toolbar composed of icon buttons, and information boxes for status, progress [number of scans], and file identification. Functions that can be obtained through the toolbar can also be obtained through the pull-down menus.

Toolbar Buttons



Main Control Window

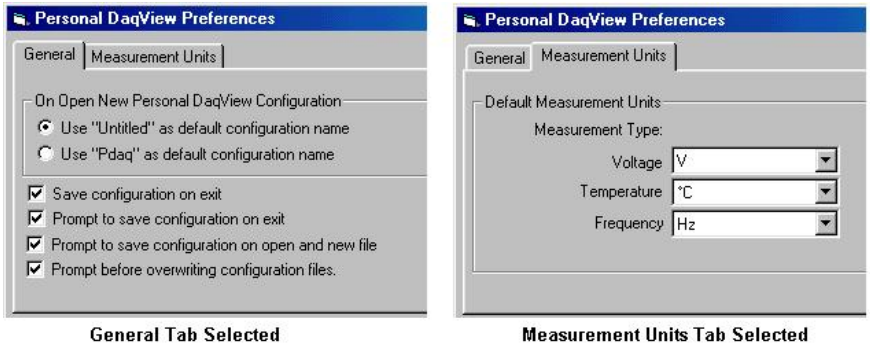
	Button	Function
1	Open Configuration File	Opens a selected configuration file.
2	Save Configuration File	Saves the current configuration file to disk.

3	Select Active Device	Provides a list of available devices, their serial number, and device type. A checkmark appears next to the selected device. Use the mouse to select (or deselect) a device. "Device" in this context can include an expansion module, for example: a Personal Daq/55 with an attached PDQ1 would be one device.
4	Arm Trigger for Disk Recording	Arms the trigger and stores acquisition data to a designated disk file. If Auto Rearm is selected, clicking this button puts Auto rearm in effect. This button, is also used to disarm the data acquisition.
5	Manual Trigger	Used to trigger the device when the mode of trigger is set to "Manual." Note that the Manual Trigger button can not be depressed until after the trigger is armed, for example, by first pressing button 4.
6	Update Digital Outputs	Updates digital outputs for all digital channels that are selected to "Output State." See Digital Input/Output Spreadsheet, page 4-14. Digital I/O does not apply to Personal Daq/54.
7	Configure Data Destination	Accesses the Configure Data Destination window (page 4-16). Note that this window provides a means of selecting sequential destinations through an auto rearm feature.
8	Configure Acquisition	Accesses the Configure Acquisition window (page 4-15).
9	Configure Channel Settings	Brings up the Channel Configuration window. From this window you can configure channels for Analog Input (page 4-8), Frequency/Pulse Input (page 4-12), and Digital Input/Output channels (page 4-14), depending on which tab is selected.
10	Update All Indicators	Starts all on-screen indicators with a display of up-to-date data. Has no affect on the recording of data to disk. Auto Rearm, even if selected, will not occur when using this control. This button is also used to pause all indicators.
11	Display Scrolling Charts	Displays data graphically in a scrolling chart. Discussion of Chart Display begins on page 4-27.
12	Display Bar Meters	Displays data in a bar graph format. Discussed on page 4-24.
13	Display Analog Meters	Displays data displayed in a dial-gage format. Discussed on page 4-25.
14	Display Digital Meters	Displays data in a digital meter format. Discussed on page 4-26.
15	View Data	Launches an independent post-data acquisition program such as eZ-PostView. Refer to the Post Acquisition Analysis PDF (included on your CD) for detailed information.

Pull-Down Menus

Aside from using the toolbar buttons to perform various program functions, you can select functions from pull-down lists as indicated by the following table.

Pull-Down Menu	Function
File	
New (Ctrl+N)	Provides a means to create a new file.
Open (Ctrl+O)	Provides a means to open an existing file.
Save (Ctrl+S)	Saves the current file under its present filename.
Save As …	Copies the current file and saves it under a different filename.
Exit (Ctrl+Q)	Exits the program.
View	
Active Devices …	Provides a list of available devices, their serial number, and device type. A checkmark appears next to the selected device. Use the mouse to select (or deselect) a device. "Device" in this context can include an expansion module, for example: a Personal Daq/55 with an attached PDQ1 would be one device. Only one device can be active at a time.
Acquisition Configuration…	Brings up the Configure Acquisition window.
Channel Configuration	Opens the channel configuration window (if closed) with the Analog Input channels spreadsheet selected.
Data Destination Configuration …	Brings up the Data Destination dialog box for assigning the destination folder, and filename. Also provides the option to select sequential destinations (Auto Rearm).
Preferences … *(Preferences are discussed in more detail on the following page)*	Opens a dialog box containing two tabs, with the following uses: *General* tab: Provides a means to select various preference options regarding opening, exiting, and saving files. *Measurements* tab: Provides a means of setting default measurement units for voltage, temperature, and frequency.



General Tab Selected **Measurement Units Tab Selected**

Preferences Dialog Boxes

You can set preferences for Personal DaqView through the *Preferences* dialog box found under the **View** pull-down menu of the *Main Control Window*.

General – Allows you to select "untitled" or "pDaq" configurations for the default filename used by Personal Daq. Note that Personal DaqView automatically loads the last saved configuration file. The second part of the General screen pertains to configuration file settings.

Measurement Units – Allows you to set the desired default units for the *Channel Configuration Window's* spreadsheet. Note that scale and offset automatically reset to a pre-designated default according to the unit selected.

Note: Changing the *measurement unit defaults* does not immediately affect the *Channel Configuration Window's* spreadsheet columns. For example: If you set your voltage default to **µV**, but the channels are set with **V** units, they will retain the **V** units. However, changing these channels to temperature, then back to voltage would result in the default units (**µV**) and the associated scale &

offset. The new default units are also applied to all channels when a new configuration file is created.

Acquire

Arm	Arms the trigger and stores acquisition data to a designated disk file. If Auto Rearm is selected, activating this funtion will put Auto rearm into effect.
Disarm	Disarms and stops the data acquisition.
Manual Trigger	Used to trigger the device when the mode of trigger is set to "Manual." Note that the Manual Trigger can not be activated until after the trigger has been armed.

Device

Update Digital Outputs	Updates digital output channels, regardless of whether their output state is closed or opened (see page 4-14).
Save Power-up Settings	Saves power-up state settings of Personal Daq's internal switches (see page 4-14).

Tools

Convert Binary Data …	The **Convert Binary Data** menu option allows you to convert raw binary data (***.bin** files) into other formats that you may find more useful. You must first select an existing binary file to be converted. The filename can be typed in or selected by the Browse button that leads to a folder/file search window. After a file is selected (or multiple files), you can toggle check-boxes on/off for each format type. When ready to begin the conversion, select the Convert button and set up the destination folder/filename. Data collected can be uploaded to your PC's hard disk in any or all of several data formats for post-acquisition analysis. Some of the available file formats include Snap Master, DADiSP, Matlab, and ASCII (Excel) which is compatible with most software for analysis.
View Data	Launches an independent post-data acquisition program such as eZ-PostView. Refer to the Post Acquisition Analysis PDF (included on your CD) for detailed information.

Indicators

Start All Indicators	Starts all on-screen indicators with a display of up-to-date data. Has no affect on the recording of data to disk. Auto Rearm, even if selected, will not occur when using this control.
Stop All Indicators	Stops all indicators. Has no affect on the recording of data to disk.
Analog Meters	Brings up the analog dial-type meters.
Bar Graph Meters	Brings up the bar graph type meters.
Chart Display	Brings up the scrolling chart.
Digital Meters	Brings up the digital style meters.

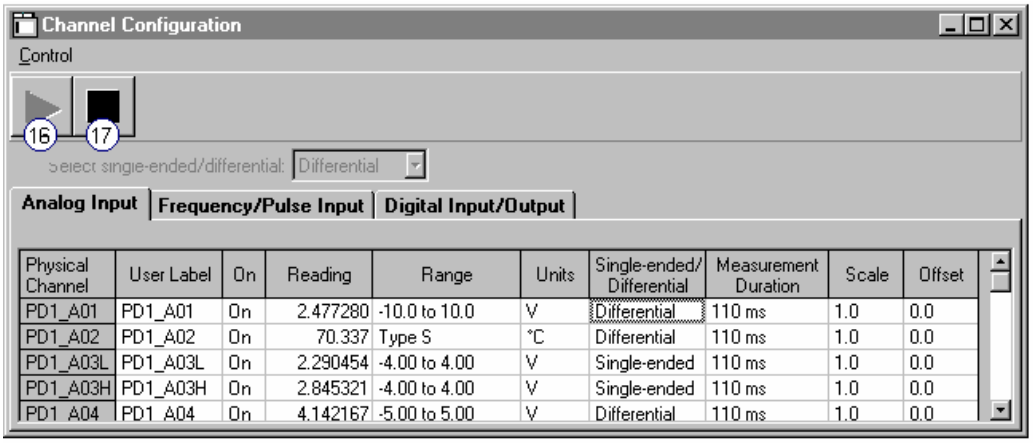
Help

Contents and Index	Accesses the program's Help file. Contents include: *Introduction* *Setting Up Personal DaqView* *Taking an Acquisition* *Viewing Your Data* *Frequently Asked Questions* *Troubleshooting*
About Personal DaqView	Provides the program's software version number.

Channel Configuration Window

You can bring up the *Channel Configuration Window* by using the *Configure Channel Settings* button (9) on the *Main Control Window* toolbar, or by selecting *Channel Configuration* from the *Main Control Window's* **View** pull-down menu.

The *Channel Configuration Window* contains a **Control** pull-down menu and a 2-button toolbar. The window displays any one of three tabbed spreadsheets for selecting the type of channel to be configured. The tabs are labeled Analog Input, Frequency/Pulse Input, and Digital Input/Output.



Physical Channel	User Label	On	Reading	Range	Units	Single-ended/ Differential	Measurement Duration	Scale	Offset
PD1_A01	PD1_A01	On	2.477280	-10.0 to 10.0	V	Differential	110 ms	1.0	0.0
PD1_A02	PD1_A02	On	70.337	Type S	°C	Differential	110 ms	1.0	0.0
PD1_A03L	PD1_A03L	On	2.290454	-4.00 to 4.00	V	Single-ended	110 ms	1.0	0.0
PD1_A03H	PD1_A03H	On	2.845321	-4.00 to 4.00	V	Single-ended	110 ms	1.0	0.0
PD1_A04	PD1_A04	On	4.142167	-5.00 to 5.00	V	Differential	110 ms	1.0	0.0

Channel Configuration Window, Selected for Analog Input Spreadsheet

Channel Configuration Window Toolbar

The *Channel Configuration Window* consists of two buttons (items 16 and 17) as identified by the previous figure and the following table.

	Button	Function
16	Enable Readings Column	Activates the *Channel Configuration Window's* reading column. Does not affect the recording of data to disk.
17	Disable Readings Column	Stops the *Channel Configuration Window's* reading column. Does not affect the recording of data to disk.

Channel Configuration Window Pull-down Menu

The channel configuration window contains one pull-down menu labeled, **Control**. This menu provides a means of enabling and disabling the readings column, as do toolbar items (16 and 17).

Common Spreadsheet Columns

The *Channel Configuration Window*, regardless of which tab is active, consists of a spreadsheet designed for the acquisition of a specific type of data (i.e., analog, frequency, digital). *Depending on the column*, you can make changes to the information contained in a cell by placing the mouse cursor in the cell and then using the applicable mouse-button methods as follows:

- ***"Single-click"* with the left mouse button** to open an associated pull-down list for the applicable cell, from which a selection can be made. This pull-down list appears just below the toolbar.
- ***"Double-click"* with the left mouse button** to cycle through listed selections or write-enable a cell, as applicable. If the cell has a given parameter list (such as those in the *On*, *Range*, and *Measurement Duration* columns) the parameter will change with each double-click, allowing you to cycle through all possible selections. Note that these selections are repetitive; in other words, you will eventually advance to the same selection you started with. "Type-in" cells (such as User Label, Scale, and Offset) can be selected on double-click for easy editing.
- ***"Single-click"* left, then *"Single-click"* right** to write-enable a cell. Completing this action with the mouse buttons (while having the cursor on a cell such as Scale or Offset) allows you to use you PC's keypad to type the desired value into the field.
- ***"Single-click"* left, *"single-click"* right, then *"single-click"* left again** to open an associated pull-down list for the applicable cell, from which a selection can be made. This pull-down list appears in the selected cell's row.

Note: Although each type of channel configuration spreadsheet (analog, frequency, and digital) is discussed separately, the common columns are presented in this section to avoid redundancy.

Note: You can use your PC's keypad *arrow keys* to select new "active cells" in the spreadsheet.

Physical Channel — automatically identifies the Personal Daq device and channel. The user cannot change the "Physical Channel" nomenclature. A few examples are:

PD1_A01 - Personal Daq unit 1, Analog Input channel 1 --- a differential channel
PD1_A02L - Personal Daq unit 1, Analog Input channel 2, Low --- a single-ended channel, low end
PD1_A02H - Personal Daq unit 1, Analog Input channel 2, High --- a single-ended channel, high end
PD2_F1 - Personal Daq unit 2, Frequency/pulse channel 1
PD1_D05 - Personal Daq unit 1, Digital channel 5

User Label — provides a means of identifying the channel by a user-defined descriptive name. If no name is specified, the program uses the physical channel name as a default. The user label appears in the trigger and chart selection lists, discussed later in this manual. You can change the user label to any alphanumeric designation, providing each label is unique (channel-specific).

On — allows you to enable a channel for data collection. When a cell or block of cells in this column is selected, a selection box will appear that allows "On" to enable or "Off" to disable the channel. Double-clicking a cell in this column toggles the channel's enable status.

Reading — displays scanning device input readings. The column is activated when the acquisition device is triggered. The column's values are *real-time* channel values from the instrument and cannot be altered by the user. This column will update the readings as fast as the computer will allow.

Note: If you re-size the readings column such that it becomes too narrow for all digits to be displayed, you may misinterpret the reading. For example, since the left-most digits will be lost from view first, a reading of 8.388118 V could look like 8118 V (with the most significant didgits of 3.38 no longer visible).

Scale & **Offset** — The **scale/offset** feature applies to Analog Input and Frequency/Pulse Input spreadsheets. This feature allows you to alter the default linear ($mx + b$) transfer function. These types of alterations can be useful in special applications. An example follows shortly.

To change **scale** or **offset** values from default: use the mouse to select the spreadsheet cell (of the applicable channel's scale or offset), type in the desired value, and hit the enter key of your keypad. Note that the **scale** is the linear relation to the input, sometimes referred to as *m*. **Offset** is plus or minus from zero, the "*b*" of the $mx + b$ linear equation.

Note: The *reading* and *range* columns will change automatically according to the new **scale/offset** values.
The following examples illustrate possible uses of the **scale/offset** feature.

Engineering Units Conversion Using mx + b

Most of our data acquisition products allow the user to convert a raw signal input (for example, one that is in volts) to a value that is in engineering units (for example, pressure in psi). The products accomplish this by allowing the user to enter *scale* and *offset* numbers for each input channel, using the software associated with the product. Then the software uses these numbers to convert the raw signals into engineering units using the following "**mx + b**" equation:

Engineering Units = m(Raw Signal) + b **(1)**

The user must, however, determine the proper values of *scale* (**m**) and *offset* (**b**) for the application in question. To do the calculation, the user needs to identify two known values: (1) the raw signal values, and (2) the engineering units that correspond to the raw signal values. After this, the scale and offset parameters can be calculated by solving two equations for the two unknowns. This method is made clear by the following example.

***Example 1*:**

An engineer has a pressure transducer that produces a voltage output of 10.5 volts when the measured pressure is 3200 psi. The same transducer produces an output of 0.5 volt when the pressure is 0 psi. Knowing these facts, m and b are calculated as follows.

A - Write a pair of equations, representing the two *known* points:

3200 = m(10.5) + b **(2)**

0 = m(0.5) + b **(3)**

B - Solve for m by first subtracting each element in equation (3) from equation (2):

3200 - 0 = m(10.5 – 0.5) + (b - b) **(4)**

Simplifying gives you: **3200 = m(10)** **(5)**

This means: **m = 320** **(6)**

C - Substitute the value for m into equation (3) to determine the value for b:

0 = 320 (0.5) + b **(7)**

So: **b = - 160** **(8)**

Now it is possible to rewrite the general equation (1) using the specific values for m and b that we just determined:

Engineering Units = 320(Raw Signal) - 160 **(9)**

The user can then enter the values of m and b into the appropriate location using the facilities provided by compatible data acquisition software, for example: WaveView, DaqView, Personal DaqView, LogView, and TempView. The software uses equation (9) to calculate signal values in engineering units from that point on.

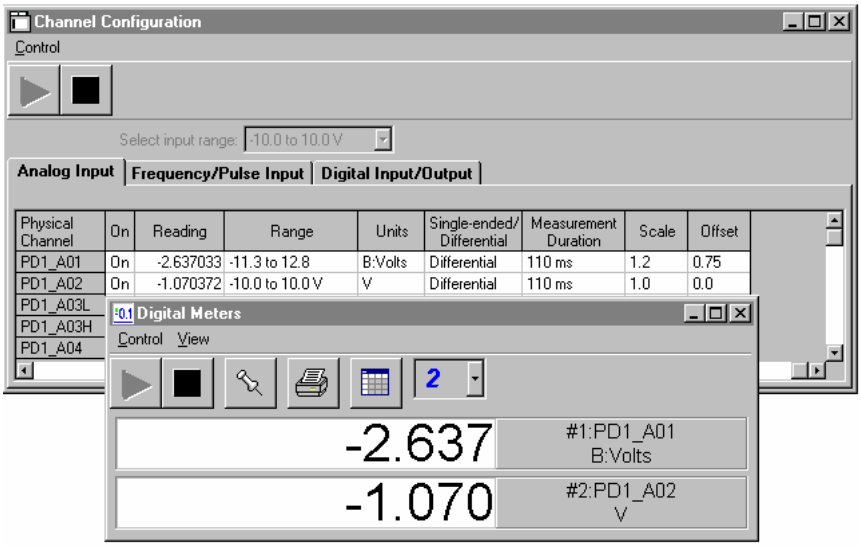
Example 2:

A Personal Daq user knows that a linear relationship exists between the voltage output at "Point A" and the voltage output at "Point B" in a certain electrical device. The linear relationship is constant. The user wants to monitor the voltage at "Point B," however; it is inconvenient to connect signal lines in that section of the apparatus.

The user realizes he can take the reading at "Point A" and use the *mx + b* equation to solve for "Point B." In this example, $\mathbf{B_{Volts} = 1.2x + 0.75\ Volts}$, where B_{volts} is the voltage at "Point B" and x is the voltage at "Point A."

After his acquisition was configured, the user performed the following steps to allow Personal DaqView to calculate and display B_{Volts}.

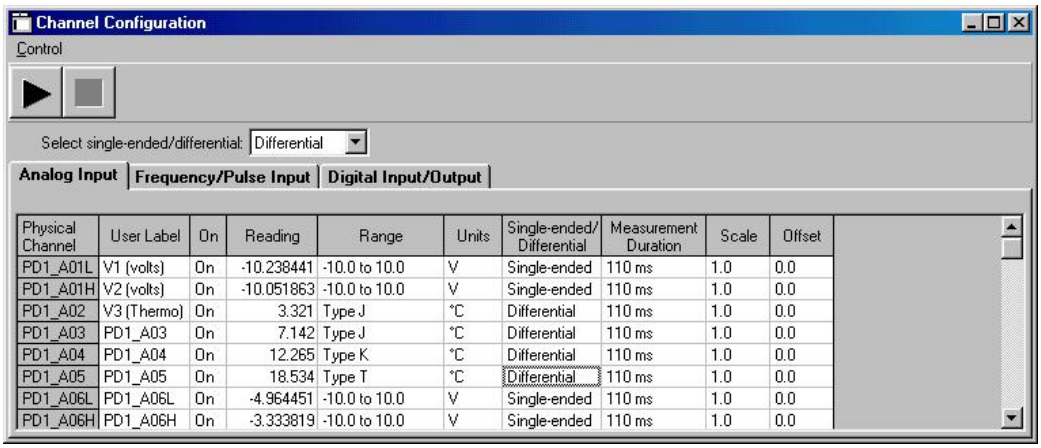
1. *Left-clicked/right-clicked* on the **units** cell for PD1_A01, typed in **`B:Volts`**, and hit the keyboard's Enter key.
2. *Left-clicked/right-clicked* on the **Scale** cell for PD1_A01, typed **`1.2`** for the **Scale**, and hit the keyboard's Enter key.
3. *Left-clicked/right-clicked* on the **Offset** cell for PD1_A01, typed **`0.75`** for the **Offset**, and hit the keyboard's Enter key.
4. Used channel PD1_A01 to acquire voltage readings which represented the voltage at "Point B" (B:Volts).
5. Selected the Digital Meters button (14) on the *Main Control Window's* toolbar and configured the meter to display readings for PD1_A01 (B:Volts) and PD1_A02.
6. Clicked the Update All Indicators button (10) on the *Main Control Window's* toolbar and monitored the readings.



PD1_A01 B:Volts Readings resulting from use of the Scale/Offset Feature.
In this example B:Volts is being displayed on the Analog Input Spreadsheet and in a Digital Meter.

Analog Input Spreadsheet

The Analog Input spreadsheet allows you to configure analog input channels. Each row shows a single channel and its configuration. The following text provides more detail regarding the channel configuration parameters for Analog Input.
Note that columns labeled Physical Channel, User Label, On/Off, Reading, Scale, and Offset are discussed in the immediately preceding section, *Common Spreadsheet Columns.*



Physical Channel	User Label	On	Reading	Range	Units	Single-ended/ Differential	Measurement Duration	Scale	Offset
PD1_A01L	V1 (volts)	On	-10.238441	-10.0 to 10.0	V	Single-ended	110 ms	1.0	0.0
PD1_A01H	V2 (volts)	On	-10.051863	-10.0 to 10.0	V	Single-ended	110 ms	1.0	0.0
PD1_A02	V3 (Thermo)	On	3.321	Type J	°C	Differential	110 ms	1.0	0.0
PD1_A03	PD1_A03	On	7.142	Type J	°C	Differential	110 ms	1.0	0.0
PD1_A04	PD1_A04	On	12.265	Type K	°C	Differential	110 ms	1.0	0.0
PD1_A05	PD1_A05	On	18.534	Type T	°C	Differential	110 ms	1.0	0.0
PD1_A06L	PD1_A06L	On	-4.964451	-10.0 to 10.0	V	Single-ended	110 ms	1.0	0.0
PD1_A06H	PD1_A06H	On	-3.333819	-10.0 to 10.0	V	Single-ended	110 ms	1.0	0.0

Channel Configuration Window, Selected for Analog Input Spreadsheet

Range — Displays the range and provides access to a pull-down list of available voltage ranges, as well as a list of available thermocouple types. In addition to voltage, analog input can be selected for thermocouple types J, K, T, E, R, S, B, and N. Note that the range and units columns are interrelated. For example, selecting a thermocouple from the range list will automatically bring up the default unit indicated in *Preferences, Measurement Units* in the **View** pull-down menu.

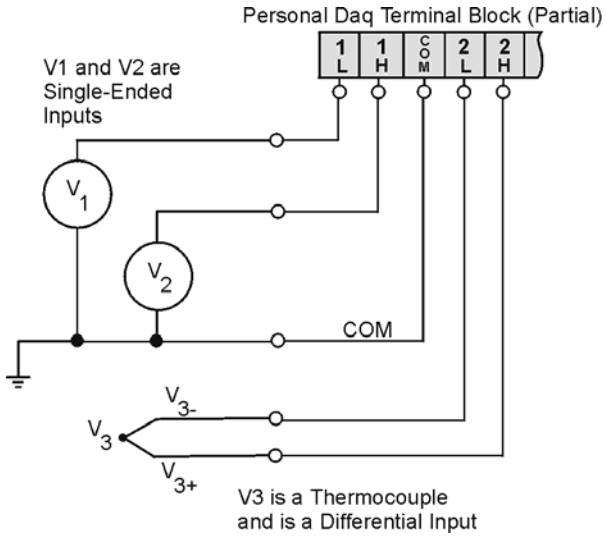
Units — Displays units and allows the user to key-in different units when the cell is selected. Units may be V, mV, uV, or created special by the user (as in the *B:volts example*, page 4-9). In addition, when a thermocouple type is selected from the range pull-down list, units of °C, °F, °K, or °R are possible. (Also see *Scale & Offset*, page 4-7).

Single-ended/Differential — Identifies the channel mode as single-ended or differential and is used to change the mode configuration. When changing from differential to single-ended mode, a new channel row is added, as well as a new default label. For example: Changing the mode of channel PD_1A01 (Personal Daq Unit 1, Analog Channel 01) from differential to single-ended results in two channels with default Physical Channel labels and User Labels of PD_1A01L and PD_1A01H.

Single-ended inputs are typically used with signals that share the same common low (COM) such as multiple batteries which have their negative sides connected in common. For Personal Daq applications, the *single-ended* mode is not to be used for thermocouples.

Differential inputs are required when signals do not share the same common low, such as in the typical use of thermocouples. A simple example showing two single-ended inputs (V1 and V2) and a differential input (V3) follows.

Note that V1, V2 and V3 (from the following figure), correspond with the first three channels on the previous screen shot. In the screen capture, V1 and V2 are shown to have "V" for units and are single-ended. V3 a thermocouple with units of °C and is shown to be Differential.



Single-Ended and Differential Connections to Analog Inputs



In Personal Daq applications, thermocouples should <u>not</u> be connected *single-ended*. Doing so can result in noise and false readings. This is especially true when acquiring other high-amplitude signals in conjunction with thermocouple signals that are connected *single-ended*.

<u>Measurement Duration Column</u> — Indicates the amount of time used for sampling a channel's input signal. The *measurement durations* range from very slow (610 milliseconds/sample) to very fast (12.5 milliseconds/sample). Some related terms are defined as follows:

Sample rate – Samples per second. The sample rate is the number of samples that take place per second. With a slow *measurement duration* of 610 milliseconds, there will only be 1.6 samples per second. With a very fast *measurement duration* of 12.5 milliseconds, there will be 80 samples per second.

Resolution (Bit RMS) – The number of reliable data bits that exist for a signal's measurement. The greater the resolution, the more *detailed* the reading, for example, with increased resolution a reading of 5.12 V could become 5.11896 V. Personal Daq actually provides for 24 bits of data information; however, the accuracy of the least significant bits becomes less as the measurement duration speeds up. At a *measurement duration* of 610 milliseconds, the last two bits are considered unreliable, resulting in a resolution of 22 bits. At a very fast *measurement duration* (12.5 milliseconds), the nine *least significant bits* are unreliable, resulting in 15 bit accuracy.

Speed vs. Resolution[1]

Speed Designation	Measurement Duration (Scan Period)	Maximum Sample Rate[2] (Samples/sec)	Resolution (Bits RMS) (-4 V to +4 V range)
Very Slow, 50 / 60 Hz rejection	610 ms	1.6 / sec	22
Slow, 50 Hz rejection	370 ms	2.7 / sec	22
Slow, 60 Hz rejection	310 ms	3.2 / sec	22
Medium, 50 Hz rejection	130 ms	7.7 / sec	21
Medium, 60 Hz rejection	110 ms	9.2 / sec	21
Medium	40 ms	25 / sec	19
Fast	20 ms	48 / sec	17
Very Fast	12.5 ms	80 / sec	15

Notes: 1. Each channel can have independent measurement speed and resolution.
2. The sample rates shown were obtained with continuous self-calibration disabled. The table values do not include the use of CJC measurements.

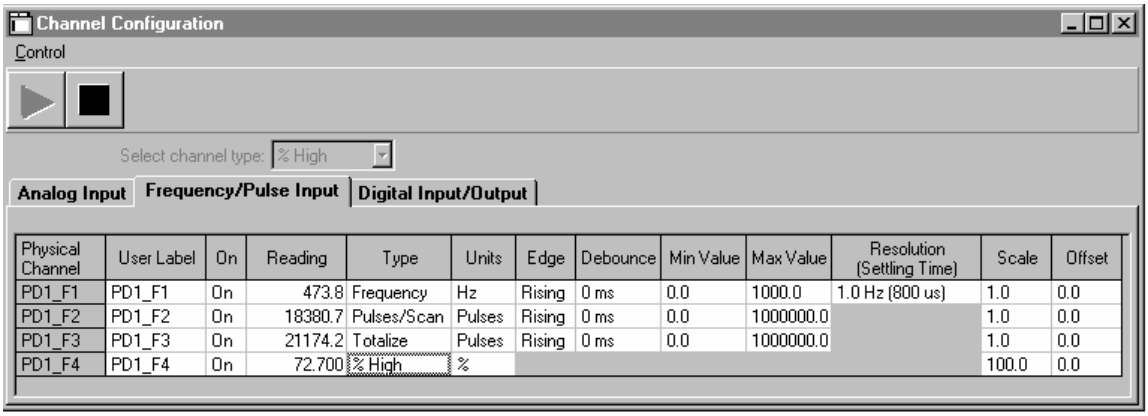
When you select the *measurement duration* you also determine the *maximum sample rate* and *resolution* for the applicable channel. For Personal Daq's analog input applications, *sample rates* range from 1.6 samples/sec up to 80 samples/sec and corresponding resolution ranges from 22 to 15 bits.

The following table provides general advice regarding the selection of *measurement duration*.

Analog Input Signal Variability	Measurement Duration	Sample Rate	Resolution
Steady, or gradual change	long	low	high
Highly variable (unsteady)	short	high	low

Frequency/Pulse Input Spreadsheet [Does not apply to Personal Daq/54]

The Frequency/Pulse Input spreadsheet allows you to configure the related channels. Each row shows a single channel and its configuration. Additional information regarding frequency measurement is included in the **Help** file.



Physical Channel	User Label	On	Reading	Type	Units	Edge	Debounce	Min Value	Max Value	Resolution (Settling Time)	Scale	Offset
PD1_F1	PD1_F1	On	473.8	Frequency	Hz	Rising	0 ms	0.0	1000.0	1.0 Hz (800 us)	1.0	0.0
PD1_F2	PD1_F2	On	18380.7	Pulses/Scan	Pulses	Rising	0 ms	0.0	1000000.0		1.0	0.0
PD1_F3	PD1_F3	On	21174.2	Totalize	Pulses	Rising	0 ms	0.0	1000000.0		1.0	0.0
PD1_F4	PD1_F4	On	72.700	% High	%						100.0	0.0

Channel Configuration Window, Selected for Frequency/Pulse Input Spreadsheet

The following text provides more detail regarding the frequency/pulse channel configuration parameters. Note that columns labeled Physical Channel, User Label, On/Off, Reading, Scale, and Offset are discussed in the section, *Common Spreadsheet Columns.*

Type — A block of cells in this column can be selected for convenience of single *type* selection, where *type* can be Frequency, Totalize, Pulses per Scan, or Duty Cycle (% High or % Low). The selected type determines the default units.

Units — Frequency *type* units can be Hz or kHz. Totalize and Pulses per Scan *type* units are Pulses; and Duty Cycle (% High and % Low) *type* units are %. Default units are set in the *Preferences* section of the *Main Control Windows's* **View** pull-down menu.

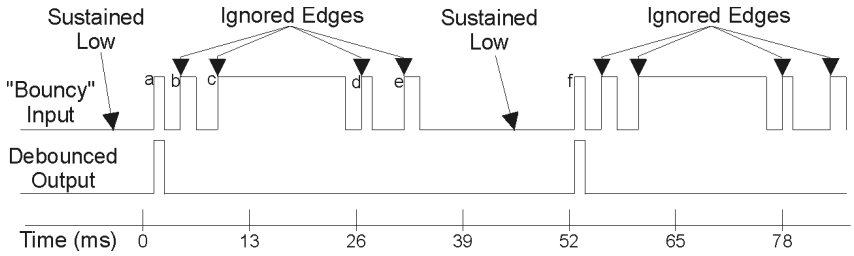
Edge — Readings for *Frequency*, *Totalize*, and *Pulses per Scan* are based on successive *rising* [or *falling*] edges of the input signal, and on the time lapse between these signal edges (see *Debounce*). Each channel's edge selection is independent of the others, i.e.; you can have some channels selected for *rising* edge while others are selected for *falling* edge.

Note: Depending on the application, one edge type (*rising* or *falling*) may be electronically cleaner than the other.

Note: As indicated in the following text and figure, some edges are insignificant and can be ignored.

Debounce — Debouncing is a process of ignoring signals which are considered as too short in duration to be real events. Personal Daq's debounce circuit ignores two types of edge signal events: 1) rising edges that are not preceded by a sustained low signal, and 2) falling edges that are not preceded by a sustained high signal. The interval can be independently set for each channel to a value of 0, 0.8, 3.2, or 13 ms. You can select 0 ms to disable debouncing for clean high-frequency signals. Note that long debounce times will limit high-frequency response. For example, a 13 ms debounce will limit frequency to about 50 Hz.

The following figure shows the effect of 13 ms debouncing on a noisy signal. To be counted, a rising edge must be preceded by a sustained low signal for at least 13 ms without any other edges [a falling edge must be preceded by a sustained high signal for at least 13 ms without any other edges]. (See following figure).



Sustained Lows and Rising Edges… "Bouncy" Input compared with Debounced Output

In the this example of "bouncy" input, **edge** was selected for *rising* and **debounce** was selected for 13 ms. Rising edges **a** and **f** are counted because they are preceded by low signal levels sustained for at least 13 ms (the debounce time). All other rising edges (**b**, **c**, **d**, and **e**) are ignored.

Min Value — For *Frequency*, *Pulses/scan*, and *Totalize*, defines the minimum range value (lower range limit) for the charted signal. The value is applicable to charts in Personal DaqView and PostView.

Max Value — For *Frequency*, *Pulses/scan*, and *Totalize*, defines the maximum range value (upper range limit) for the charted signal. The value is applicable to charts in *Personal DaqView* and *PostView*.

Resolution and Settling Time

Frequency measurements on the Personal Daq are achieved by querying the unit for a current time and pulse count on a repetitive basis. Aside from *edge* and *debounce* settings, two values are used to set up frequency input; Maximum Frequency (**Fmax**) and Measurement Resolution (**Mreso**). Both of these values have units of Hertz.

The Acquisition Period (AcqPeriod) is calculated from these values. AcqPeriod is the minimum length of time needed to get a reading at the desired resolution.

AcqPeriod (in seconds) = Fmax * (0.0000004 / Mreso)

However, if the requested scan period for the entire scan (of which the frequency channels are a part) is longer than the calculated **AcqPeriod**, then the scan period becomes the acquisition period.

To put bounding limits on values so measurements can be done in a timely fashion, the **AcqPeriod** must be ≤10 seconds. This means: **Fmax / Mreso ≤25,000,000**.

Frequency measurement on the Personal Daq is *only done when collecting scans*. Values are updated on every scan even if the **AcqPeriod** has not been reached. During each scan a current pulse count and timer count is read from the Personal Daq.

The frequency is calculated as follows:

1) An *estimated* frequency (**Fest**) is calculated as follows:

- *If there were no pulses read*; **Fest = Flast** (the last frequency reported) or, **Fest = 1 / current time** (whichever is smaller)
- *If there were pulses counted then;* **Fest = counted pulses / current time**

2) The *actual* reported frequency (**Frep**) is calculated as follows:

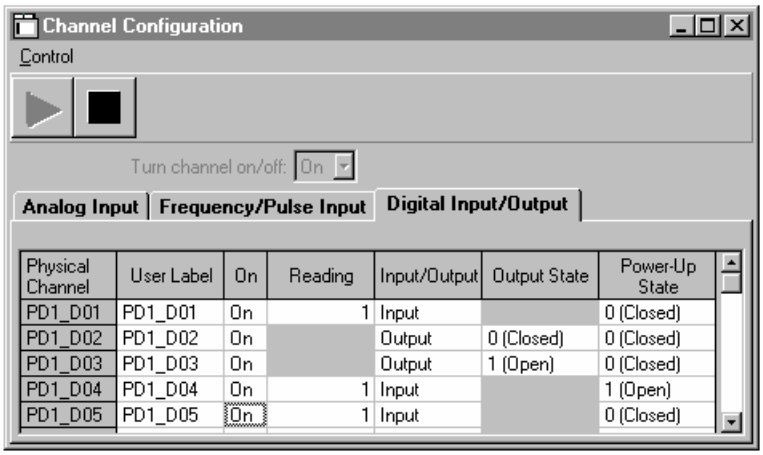
- *If current time* <**AcqPeriod** *then*;

  Frep = [[Flast * (AcqPeriod - current time)] + (Fest * current time)] / [AcqPeriod]
- *Otherwise;* **Frep = Fest**

Note: Current Time is the elapsed time from the start of a frequency measurement and is reset each time the **AcqPeriod** is reached.

Digital Input/Output Spreadsheet [Does not apply to Personal Daq/54]

The Digital Input/Output spreadsheet allows you to configure and monitor the related digital channels. Each row shows a single channel and its configuration.



Channel Configuration Window, Selected for Digital Input/Output Spreadsheet

The following text provides more detail regarding the Digital Input/Output channel configuration parameters. Note that columns labeled Physical Channel, User Label, On, and Reading are discussed in the section, *Common Spreadsheet Columns* (page 4-7). The Scale and Offset feature does not apply to digital channels.

Input/Output Column — This column identifies the digital channel mode as input or output, and is used to select the desired mode.

Output State Column — When the output mode is selected, Personal Daq's physical internal switch closes to ground. This 0 (Closed) switch position is the output state's default setting. 0 (Closed) is typically used for digital output logic applications, where logic output is low (0). In this state, digital output is closed to ground.

When the digital output will be used to switch an external load, such as a relay, the output may be up to +5V (hi) and Personal Daq's internal switch should be set to 1 (Open). In this use of the digital output, the circuit is not closed to ground.

Note that since the reading column is for input values, the applicable reading cell is shaded-out when "output state" is selected.



When an active (On) channel is selected to "Output State" the *Update Digital Outputs* button (on the *Main Control Window*) becomes active. This button is used to update the digital output channels, regardless of whether their output state is closed (0) or open (1).

Note: The update of changed output settings will not actually take place until *the Update Digital Outputs* button is pressed. After selecting the digital output channels to the desired state (open or closed), click on the *Update Digital Outputs* button to actually *set* the indicated output states.

Power-Up State Column — This column allows you to set Personal Daq's physical internal switches to be open or closed *during power-up*, on a per channel basis. After setting the state you can save them ***internal to the device*** by opening the *Main Control Window's* **Device** pull-down menu and selecting "Save power-up settings." When the device is unplugged or losses power, its internal switches will reset to the "saved" power-up settings.

Note: Saved power-up settings are internal to the Personal Daq device and are not retained by the Personal DaqView software. If a device is removed from the PC and a different one is attached, the new device will have its own power-up settings (default or pre-saved), but will not likely have the same settings as the device which was just removed.

For *enabled* channels, Personal Daq's digital I/O ***output state*** settings will change from their power-up state designations after the device has been powered up. The changes take place as follows:

- For enabled digital ***Input*** channels - starting an acquisition will change the digital input channel, if needed, to match the configuration set in the software.
- For enabled digital ***Output*** channels - clicking on the *Update Digital Outputs* button will *sets* the indicated output state (displayed in the *Output State* column). When you change an output state you *must* click on the *Update Digital Outputs* button before the output state actually changes.

Configure Acquisition Dialog Box

Note: Parameters can not be altered while the acquisition is in progress.

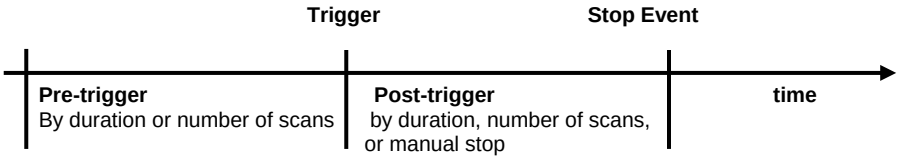
The information entered in the Configure Acquisition dialog box is used by the Arm Acquisition command to set up the acquisition of data to disk. When the trigger is satisfied, the scans are collected at the selected scan frequency and stored to disk in the designated file, as indicated in the Data Destination window. Data Destination is discussed on page 4-18.

You can access the Configure Acquisition dialog box by using the *Configure Acquisition* button (icon with the image of three slide-bars). This button is located on the toolbar the *Main Control Window*.

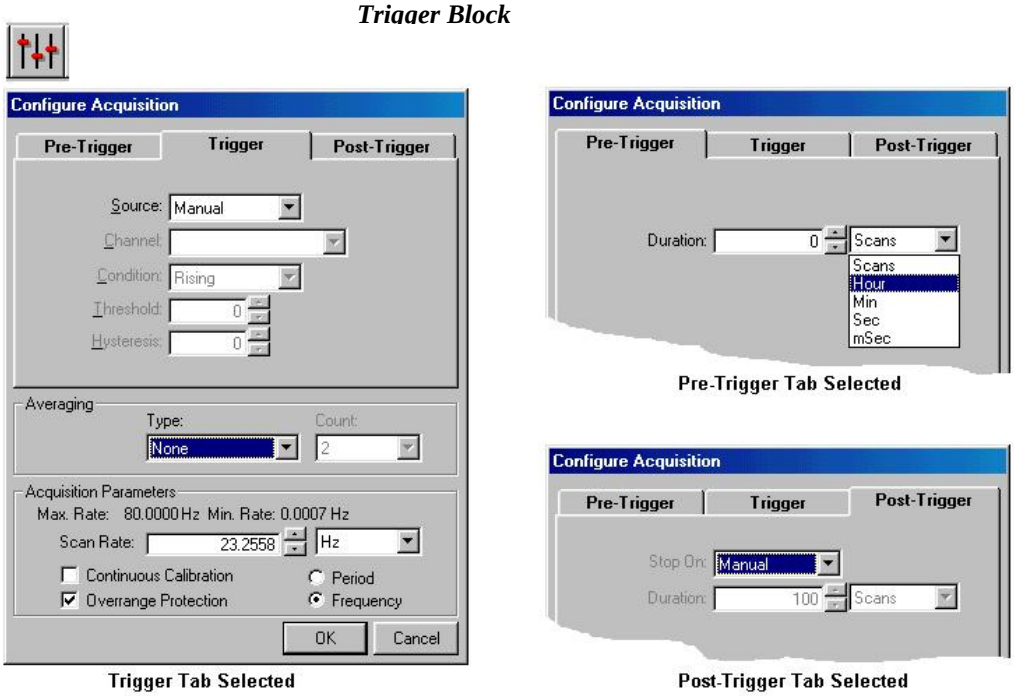
The Configure Acquisition dialog box contains three tabs: Pre-Trigger, Trigger, and Post-Trigger. When first opened, the dialog box defaults to the Trigger tab selection. In addition to the trigger-related aspect, the dialog box contains two other sections. These are labeled "Averaging" and "Acquisition Parameters."

The Trigger Section

The following figure depicts the parts that make up an acquisition. Together, the pre-trigger and post-trigger make up a trigger block.



Trigger Block



Configure Acquisition

As can be seen in the previous figure, it is relatively simple to configure an acquisition. Tabs allow you to select each of the three trigger aspects (Trigger, Pre-Trigger, and Post-Trigger) for configuration.

Averaging

The "Averaging" section of the dialog box is used to enable a block averaging mode and to select the size of the scan block. Block averaging is enabled by using the *Type* list box (available selections are None or Block Averaging). Averaging is available in scan blocks of 2, 4, 8, 16, and 32. The block number is chosen from the *Count* pull-down list. When enabled, the selected block averaging is applied to each of the scan group's analog input channels. Note that programmers can enable averaging via the **`daqAdcSetFilter`** function with *filterType* set to **`DaftSWAvg`**. For additional information you can refer to **`daqAdcSetFilter`** in Appendix B.

Acquisition Parameters

The lower section of the Configure Acquisition box is used for adjusting the scan rate, as well as enabling or disabling Continuous Calibration and Overrange Protection modes. Each of these categories is discussed below.

Scan Rate, Period, and Frequency

The lower region of the dialog box displays the maximum scan rate and the actual scan rate. This section of the dialog box allows you to set the scan timing, i.e., the rate at which *Personal DaqView* takes readings. It can be set to a **frequency** (number of scans per second) or **period** (the time length of one scan) value. The maximum scan rate depends on the channel configuration and is displayed in the configure acquisition window. It is determined by the number of enabled channels and their measurement duration.

To get the highest scan rate possible:

- turn off all unused channels
- set the measurement duration to its lowest setting on all analog channels

These steps can be performed from the *Channel Configuration Window* (when selected for Analog Input Spreadsheet). Setting the highest possible scan rate inversly affects the resolution of your analog channel readings. When setting the scan rate, it is important to be aware of various issues that arise with a low scan rate. Among these are under sampling and aliasing.

Note: The scan period can only change by increments of 1 msec. If you enter an invalid value, *Personal DaqView* will automatically convert your entry to the nearest acceptable value.

Note: *Aliasing* errors can result from having a scan rate that is too low (under sampling).

Continuous Calibration

The continuous calibration option is useful in applications that make use of very long acquisitions periods. When the continuous calibration box is checked (✓), *Personal DaqView* continuously calibrates the unit while the acquisition is in progress. Although this provides greater reading accuracy, it can result in a lowering of the maximum scan rate. For this reason the continuous calibration option has a default setting of disabled (unchecked).

Overrange Protection

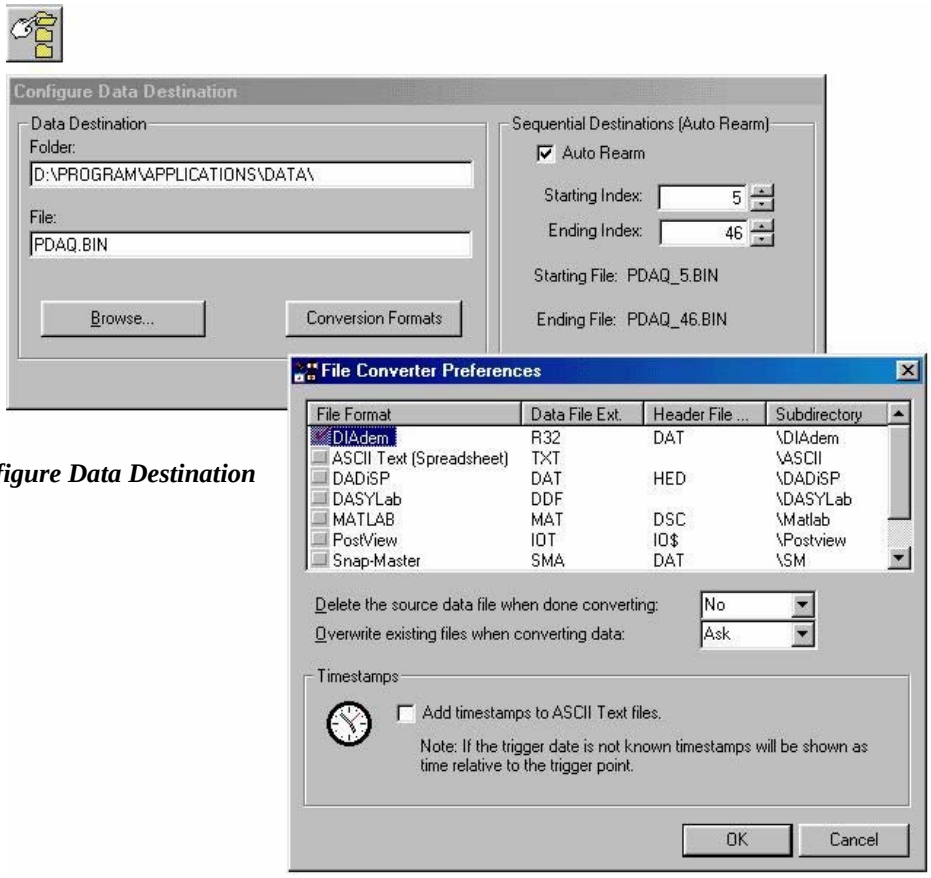
By default, the overrange protection option is enabled. This is because certain ranges are slow to recover from overload conditions. It should be noted that the slow recovery does not indicate any problem with the unit; however, if there is such an overload on a channel, erroneous readings on other channels may occur. If your application presents the possibility of a channel overload, then you should keep the overrange protection function enabled (✓). If using a custom application, use the **`DcotpDaqOverrangeProtect`** option-type setting with the **`daqSetOption`** API (see Appendix B for additional information regarding **`daqSetOption`**).

Configure Data Destination and File Converter Preferences

Conversion of eZ-PostView, eZ-TimeView, or eZ-FrequencyView files is automatic. The file converter can be used to convert other data formats.

Note that PostView is not related to eZ-PostView.

The *Configure Data Destination* dialog box can be accessed from the **View** pull-down menu, or by using the *Configure Data Destination* toolbar button (depicted in the upper left-hand corner of the following figure). The button is located in the toolbar of the *Main Control Window*.



Configure Data Destination

File Converter Preferences
(Accessed via the <Conversion Formats> button)

A breakdown of the dialog box follows.

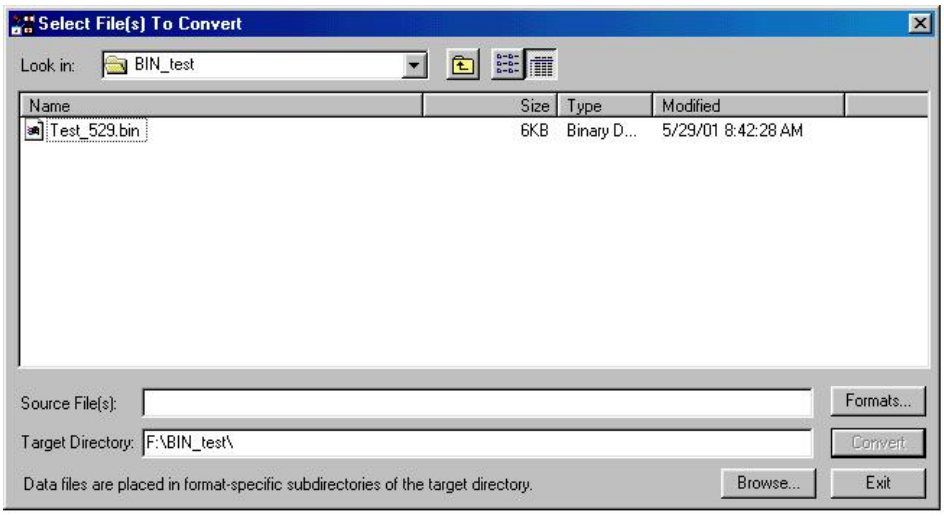
Folder – The "Folder" text box determines the file's destination folder.
File – The "File" text box identifies the file by name and raw binary extension (.BIN). Note that these filenames must have a raw binary (.BIN) extension. If you omit the extension, Personal DaqView will add it to the filename.
Browse – Used to browse existing folders and filenames.
Conversion Formats – This button is used to access the *File Converter Preferences* dialog box. This box provides a means of saving files in a variety of formats, including, but not limited to: DIAdem (.R32), ASCII (.TXT), DADiSP (.DAT), DASYLab (.DDF), and PostView Binary (.IOT) formats. The *File Converter* also allows you to add timestamps to the ASCII Text files.

Note: If you plan to use PostView, it is important that PostView Binary and/or ASCII is selected.

Note: Parameters can not be altered while the acquisition is in progress.

Note: The file's data format is selected in the **View** pull-down menu (of the *Main Control Window*), under Preferences, Data Convert. DIAdem (.R32) is selected by default. If you want to use *PostView* to view your data Ensure ASCII (.TXT) or *PostView* Binary (.IOT) is selected.

You can select *Convert Binary Data* from the Tools pull-down menu to bring up the following dialog box. This allows you to select a file for conversion. The *Formats* button accesses the *File Converter Preferences Dialog Box.*



The Select Files to Convert Dialog Box
[Accessed from the Tools Pull-down Menu]

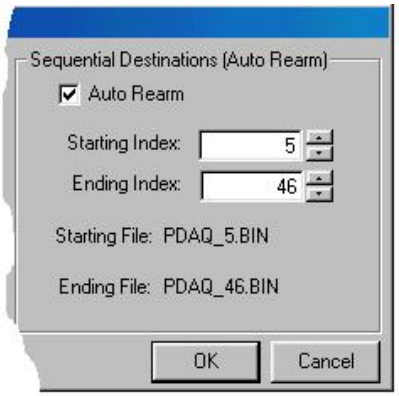
Sequential Destinations (Auto Rearm)

The right-hand portion of the *Configure Data Destination* dialog box is used to activate a Sequential Destinations (Auto Rearm) feature. This feature allows for a large number of acquisitions to take place automatically, with each using the same configuration settings. With this feature, as soon as a trigger block is terminated the acquisition system immediately re-arms and waits for the trigger to be satisfied.

To use Auto Rearm:

1. Set your file conversion preferences.
2. Verify that the Acquisition is configured as desired.
3. Ensure Post-trigger is set to stop on a duration.
4. Check the Auto Rearm box [located on the right-hand side of the Configure Data Destination box].
5. Enter the starting and ending index values for the files.
6. Select **OK**.
7. Start recording data to disk.

When the acquisition is complete, automatic rearm will occur and the next sequential file will be recorded to disk. The acquisitions will stop once all files (from the starting to ending index, inclusive) have been recorded.



Setting Auto Rearm

Why use Auto Rearm?

Auto Rearm is a convenient way to monitor and analyze specific types of trigger events. For example, you could set PD1_A01 going above 30°C as a trigger, set a post-trigger duration of one hour, and then use Auto Rearm to have an additional 99 acquisitions with this same trigger criteria (PD1_A01 > 30°C and post-trigger of one-hour). Each of the data acquisitions will occur automatically with no need to recreate the acquisition setup or restart the acquisition.

Disabling Auto Rearm

To disable Auto Rearm simply deselect the feature. Note that *deselected* is the default setting of the Auto Rearm function.

Bar Graph, Analog, and Digital Meters

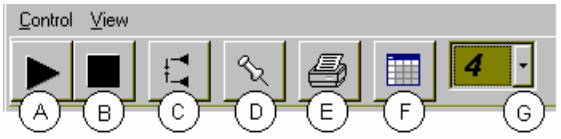
This section pertains to channel data display screens (windows, or dialog boxes) which are often referred to as *DaqMeters*. In *Personal DaqView,* the meters are accessed from the *Main Control Window's* toolbar or *Main Control Window's* **Indicators** pull-down menu.

It is important to realize that for each type of meter discussed, the meter channels selected are independent of the group chart assignments, and of the assignments for the other meters.

Note that when using meters, placing the mouse pointer over an item and then hitting Ctrl+F1 (on the keyboard) will cause a help statement to appear for that item, if such a statement is available.

Meter Toolbars

The toolbars for the three meter-types are identical, with exception that the Digital Meters toolbar does not have a **Reset Peak Hold button** (item C in the following figure).

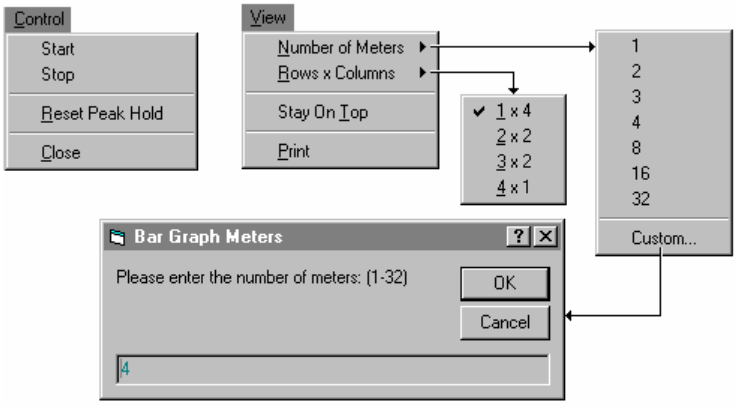


Meters Toolbar Buttons

Item	Label	Function
A	Start	Activates the meter. Does not affect the recording of data to disk.
B	Stop	Stops the meter. Does not affect the recording of data to disk.
C	Reset Peak Hold	Resets the floating markers, which indicate high and low reading peaks. Upon reset, the markers will instantly adjust to indicate the highest and lowest values reached since the time of the reset.
		This feature does not apply to the Digital Meters.
D	Stay on Top	Locks or unlocks the meter's position. When locked, the meter will appear *on top of other windows* and will retain the "on top" relationship.
E	Print	Sends the meter display image to connected printer.
F	Rows x Columns	Opens a small menu with "row x column" arrangement options. *Example*: When the number of meters is 6 the grid options will be: 6x1, 3x2, 2x3, and 2x4 with the first number being the number of rows. If you then select 3x2 you will have 3 rows of meters with 2 meters per row.
G	Select the number of Meters to display	Specifies the number of meters to appear on the screen. A maximum number of 32 meters can be selected.

Meter Pull-Down Menus

The meter windows each have **Control** and **View** pull-down menus, as indicated by the following figure. The functions of these menus can also be implemented by using the meter toolbar buttons.



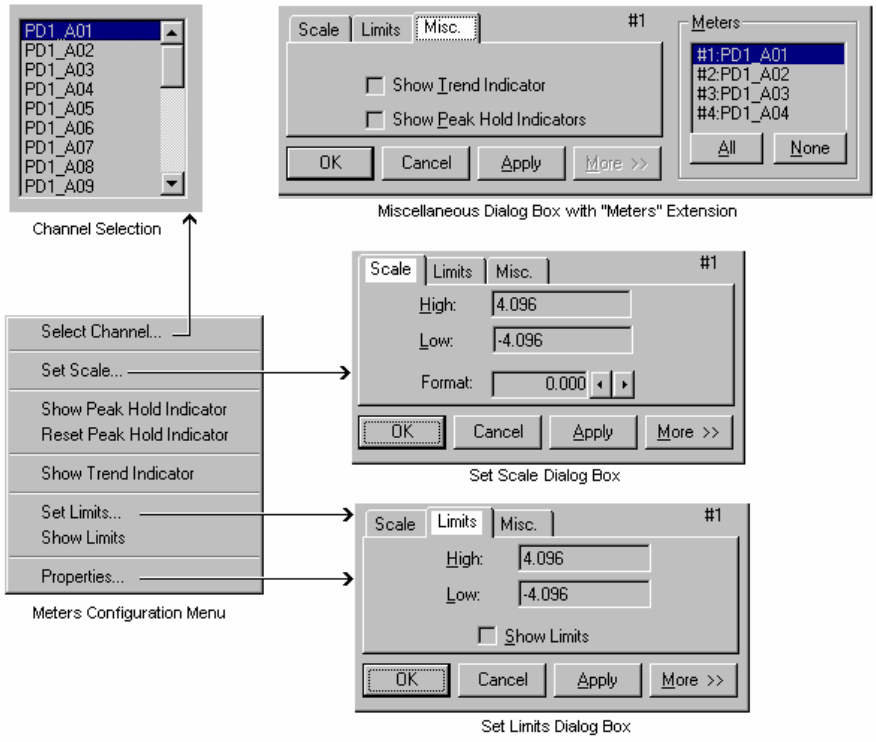
Meter Pull-Down Menu Options

Meters Configuration Menu

A meters configuration menu (lower left corner of following figure) will appear when you place the mouse pointer over a meter and click the right-hand mouse button. The menu allows you to access various dialog boxes for changing parameters for an individual meter, or simultaneously for a group of meters. The steps for configuring a meter are detailed below.

Note: The *Show Peak Hold Indicator / Reset Peak Hold Indicator* selections are not an option for Digital Meters and do not appear on the configuration window for digital meters.

Note: When the **Misc.**, **Scale**, or **Limits** dialog box is active, clicking on the "**More**>>" button causes a meters extension area to be displayed. This allows you to extend the assigned meter characteristics to additional meters of the same type. An example of the meters extension box is shown in the upper right corner of the following figure.



Meter Configuration Menu and Related Dialog Boxes

Configuring a Meter

1. Bring up the desired meter group (Bar Graph, Analog, or Digital).
2. Place the mouse cursor over the meter, which you desire to reconfigure.
3. Click on the right mouse button. A Meters Configuration Menu, similar to that in the above previous figure, will appear.

   Note: The *Show Peak Hold Indicator / Reset Peak Hold Indicator* selections are not an option for Digital Meters and do not appear on the configuration window for digital meters.

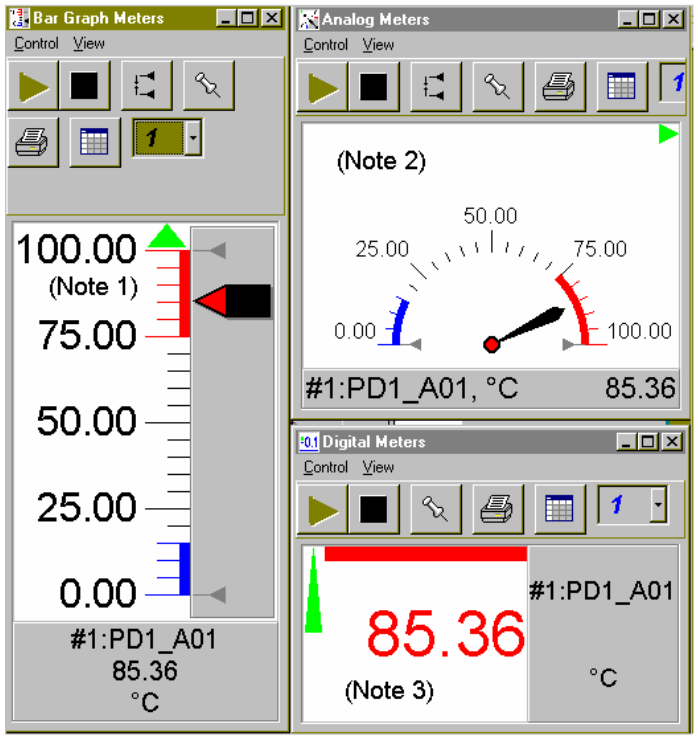
4. Select the desired option from the meter configuration menu.
5. If a dialog box is required, for example, to change a limit, simply enter in the new value in the appropriate parameter box and press "Apply" or "OK." Pressing "Apply" implements your changes, but keeps the dialog box open, allowing you to make additional changes. Pressing "OK" implements your changes and closes the dialog box.

   Note: If you desire to give two or more meters the same attributes you can click on the "More>>" button to activate the meters extension box, then highlight the meters for which you want the attributes to apply. This feature only applies to meters that are of the same type; for example, meter attributes of a Bar Graph Meter will not carry over to attributes of an Analog or Digital Meter.

The preceding figure and following table serve as a quick reference to meters configuration.

Configure Meter Settings, Function Descriptions

	Function	Description
1	**Select Channel**	Select a new channel for display. The selected channel will replace the one currently seen in the meter. Note that double-clicking the left mouse button in the meter region will also bring up a dialog box, which allows you to select a new channel.
2	**Set Scale**	Set the high and low points of the scale, as well as define the decimal place format. See following figure.
3	**Show Peak Hold Indicators**	Places high and low uni-directional floating markers on the scale to indicate the highest and lowest values reached up to the present time. **This feature does not apply to the Digital Meters selection.** (See following figure).
	Reset Peak Hold	Resets the floating markers. Upon reset, the markers will instantly adjust to indicate the highest and lowest values reached since the time of the reset. **This feature does not apply to the Digital Meters selection.** (See following figure).
4	**Show Trend Indicator**	Displays a pointer to indicate the direction of the trend. Note that during rapid meter fluctuations the increase and decrease pointers will appear to blink simultaneously. (See following figure).
5	**Set Limits**	Provides a way of establishing high and low limit set points.
	Show Limits	Displays limits by adding color (red for high, blue for low) to the scale regions, which equal and exceed the set limit values. For Digital Meters: reaching or exceeding the high limit is indicated by red numbers and an upper red bar; reaching or exceeding the low limit (in the negative direction) is indicated by blue numbers and a lower blue bar. In addition, for Bar Graphs the indicator tip turns red for high limits and blue for low limit conditions. For Analog Meters the dial arm's pivot point displays red or blue to indicate high or low limit (respectively).
6	**Properties**	Allows setting and showing limits, as well as opening the Scale dialog box.
7	**More>>**	The **More>>** button brings up a meters extension box. This allows you to select several channels for which you want to assign the same meter attributes. Without this feature each meter would have to be adjusted individually. This feature only applies to meters that are of the same type; e.g., meter attributes of a Bar Graph Meter will not carry over to attributes of an Analog or Digital Meter.



Three Meters with High and Low Limits Set

Note 1: Bar Graph Meter with indicator rising in the high limit. Meter tip is red to indicate "value is above high limit."

Note 2: Analog Meter with indicator rising in the high limit. Pivot point of dial arm is red to indicate "value is above high limit."

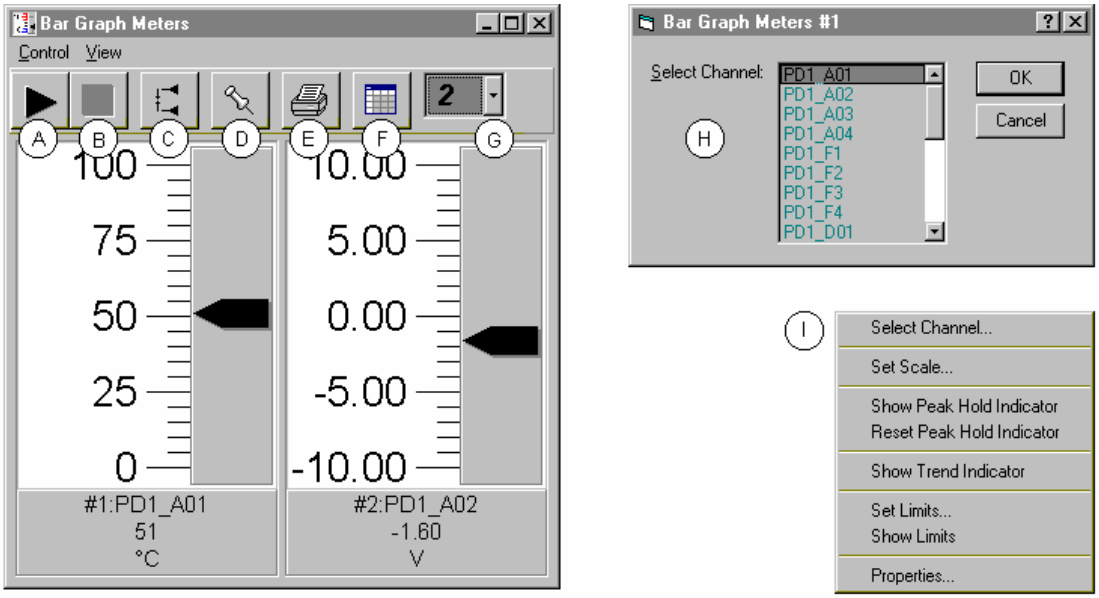
Note 3: Digital Meter with rising value in the high limit. The digital readout is red (and has a red line above it) to indicate "value is above high limit."

Each of the meters (shown in the figure) shows a trend indicator pointing in the increase direction. The Bar Graph and Analog Meters also show low and high peak hold indicators (at the 0 and 100 marks).

Be aware that the acquisition of data-to-disk has a higher priority then the updating of Charts, Meters, and the Reading column. Therefore, data is displayed as soon as the acquisition task is satisfied. As the scan rate is increased, the acquisition-to-disk task will take up more processor (CPU) time and the displaying of data will be updated as time allows. If you select a linear conversion (using the scale & offset feature) you should expect a further impact on real time display performance. **Scale & offset** conversion is discussed on page 4-7.

Bar Graph Meters

Selecting the Bar Graph Meters (from toolbar button or Indicators pull-down menu) brings up the Bar Graph window. This window displays several channels in bar graph format. To activate the display, select the Start button from this window's toolbar. At least one meter must be assigned to an **active** (On) channel. You can select to view up to 32 meters at a given time.



Bargraph Meters, Shown with 2 Meters Selected for Viewing

Note 1: Double-clicking the left mouse button in a meter scale area brings up a channel selection pop-up menu (item H). A single click with the right mouse button in this same area brings up a configuration pop-up menu (item I). Both of these pop-up menus were discussed in the section, ***Meters Configuration Menu,*** page 4-21.

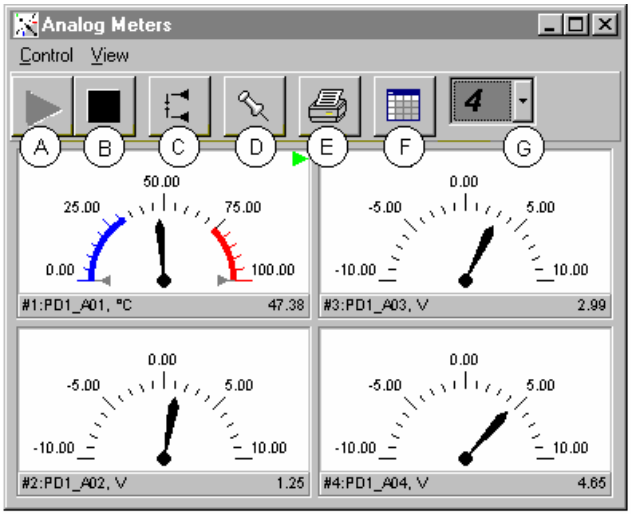
The items in the above figure as follows:

- A- Start
- B- Stop
- C- Reset Peak Hold Indicators
- D- Stay on Top
- E- Print
- F- Rows x Columns
- G- Select the number of Meters to be displayed
- H- Channel Selection box (*Activated by double-clicking left mouse button in a meter's scale area*)
- I- Configuration Pop-Up Menu (*Activated with a single click of the right mouse button in a meter's scale area*)

Configuration Note:	***…. for Bar Graph Meters***
Configure the Bar Graph meter settings by first clicking the right mouse button anywhere within the meter display area. A pop-up menu will appear allowing you to reconfigure the meter in regard to scale, limits, channel selection, adding peak hold indicators, etc. Refer to ***Meters Configuration Menu*** (page 4-21) for more detail.	

Analog Meters

Selecting the Analog Meters (from toolbar button or Indicators pull-down menu) brings up the Analog Meters window. This window displays several channels in a dial/gage format. To activate the display, select the Start button from the toolbar. At least one meter must be assigned to an **active** (On) channel. You can select to view up to 32 meters at a given time.



Analog Meters, Shown with 4 Meters Selected for Viewing

Double-clicking the left mouse button in a meters scale area brings up a channel selection pop-up menu (see Bar Graph Meters, item H). A single-click with the right mouse button in this same area brings up a configuration pop-up menu (see Bar Graph Meters, item I). Both of these pop-up menus are discussed in the section, ***Meters Configuration Menu***, page 4-21.

Note that Meter #1 (for PD1_A01 in the above figure) shows the following:

- Visible low limit region (≤ 32°C) and visible high limit region (≥ 75°C)
- Peak indicators (at 0.00 and 100.00°C)
- Trend indicator, increasing signal

The items in the above figure as follows:

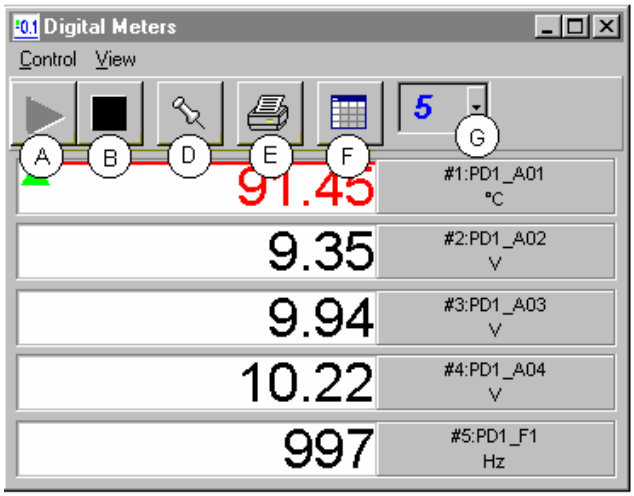
A- Start
B- Stop
C- Reset Peak Hold Indicators
D- Stay on Top
E- Print
F- Rows x Columns
G- Select the number of Meters to be displayed

Note: You can activate a Channel Selection box by double-clicking *left mouse button* in a meter's scale area. You can activate a Configuration Pop-Up Menu with a single click of the *right mouse button* in a meter's scale area. Both of these pop-up menus are discussed in the section, ***Meters Configuration Menu***, page 4-21.

Configuration Note:	***…. for Analog Meters***
Configure the Analog Meters settings by first clicking the right mouse button anywhere within the meter display area. A pop-up menu will appear allowing you to reconfigure the meter in regard to scale, limits, channel selection, adding peak hold indicators, etc. Refer to ***Meters Configuration Menu*** (page 4-21) for more detail.	

Digital Meters

Selecting the Digital Meters (from toolbar button or Indicators pull-down menu) brings up the Digital Meters window to display several channels in numeric format. To activate the display, select the Start button from the toolbar. At least one meter must be assigned to an **active** (On) channel. You can select to view up to 32 meters at a given time.



Digital Meters, Shown with 5 Meters Selected for Viewing

Note: Due to transducer and transient noises, the accuracy of voltage readings is ± 0.02%. Temperature accuracy varies, depending on thermocouple type; with type J having ± 0.5°C for a range of -100°C to +760°C. For your application, please refer to the proper specifications, including thermocouple type, when applicable.

Double-clicking the left mouse button in a meters scale area brings up a channel selection pop-up menu (see Bar Graph Meters, item H). A single-click with the right mouse button in this same area brings up a configuration pop-up menu (see Bar Graph Meters, item I). Both of these pop-up menus are discussed in the section, ***Meters Configuration Menu,*** page 4-21.

Note that Meter #1 (for PD1_A01 in the above figure) is displaying a trend indicator (in the increase direction) and also shows that the reading (91.45) is in high limit region. The number 91.45 is red and contains a red bar over it.

The items in the above figure as follows:

- A- Start
- B- Stop
- C- n/a
- D- Stay on Top
- E- Print
- F- Rows x Columns
- G- Select the number of Meters to be displayed

Note: You can activate a Channel Selection box by double-clicking *left mouse button* in a meter's scale area. You can activate a Configuration Pop-Up Menu with a single click of the *right mouse button* in a meter's scale area. Both of these pop-up menus are discussed in the section, ***Meters Configuration Menu***, page 4-21.

Configuration Note:	***…. For Digital Meters***
Configure the Digital Meters settings by first clicking the right mouse button anywhere within the digital meter display area. A pop-up menu will appear allowing you to reconfigure the meter in regard to scale, limits, channel selection, etc. Refer to the ***Configure Meter Settings*** (page 4-21) for more detail.	

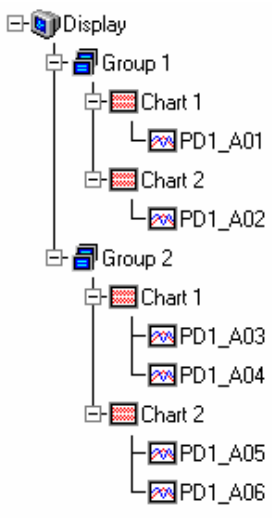
Chart Display

Reference Note: For your very first chart display setup, or when the configuration file has been deleted, refer to the *Chart Setup Wizard* section beginning on page 4-36.

Personal DaqView permits the use of multiple groups with up to four overlapping channels per chart; and can make use of Simple, Moderate or Advanced automatic chart creation functions of the wizard. Discussion of the *Chart Setup Wizard* begins on page 4-36.

Groups, Charts, & Channels

Before continuing with this section, it is important that you understand the chart display structure in terms of groups, charts and channels.



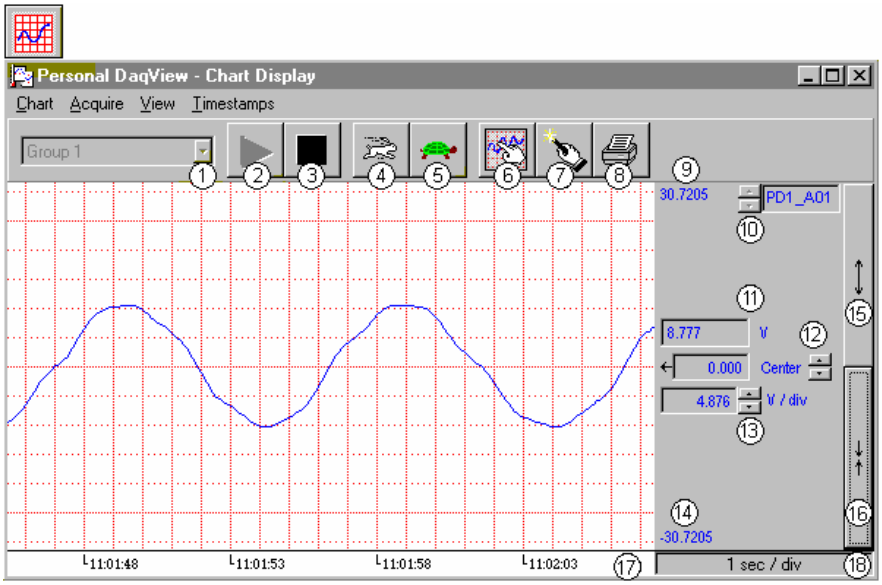
Group refers to a group of charts. Note that *Personal DaqView* allows up to 64 groups, depending on the capabilities of your PC, but can only display one group at a time.

Chart refers to display area, which reflects real-time channel data values for a selected channel and can be scrolled at various rates. You can assign up to 16 charts per group. The application can display up to four overlapping channels per chart.

Channel refers to a signal channel. Channels will be displayed in units of °C, °F, °K, °R, mV, V, Hz, or kHz, depending on the configuration. When starting the program with no configuration file present, *Personal DaqView* creates a display configuration of one group, one chart, and one channel (the first active channel found). You can change your chart display configuration through use of a dialog box or *Chart Setup Wizard*. See pages 4-31 and 4-36 respectively.

Chart Display Window

The *Chart Display Window* allows you to view scrolling charts for selected channels in real time. You can access this window by selecting the *Display Scrolling Charts* button (located in the toolbar of the *Main Control Window*). The *Chart Display* window contains four Pull-down menus, a toolbar, chart region, and channel information region.

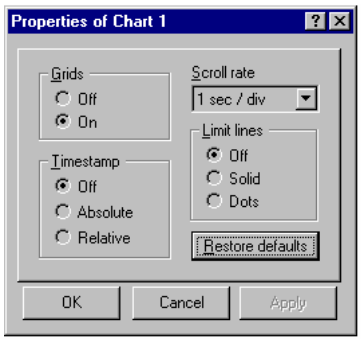


Personal DaqView's Chart Display Window

Note: A discussion of the numbered buttons and regions begins on page 4-29.

Pull-Down Menus

Pull-Down Menu	Function
Chart	
Wizard (Ctrl+W)	Opens *Chart Setup Wizard* for manual or automatic configuration of the chart display. Detailed discussion of the *Chart Setup Wizard* begins on page 4-36.
Setup (Ctrl+D)	Accesses the *Display Configuration Setup* dialog box for normal editing of the current chart display configuration. See page 4-31, *Set Up Charts*.
Next Group (Ctrl+G)	Selects the next available chart group.
Faster (Ctrl+Z)	Increases the chart scroll rate, as does the toolbar's rabbit button.
Slower (Ctrl+X)	Decreases the chart scroll rate, as does the toolbar's turtle button.
Zoom	When more than one chart is displayed, this feature allows you to *zoom* in on one of the charts, such that only the selected chart is displayed. From this "zoomed state" you can use the feature to return the display to normal (Un-zoom), or to select another available chart.
Properties	*Grids* – Turns grid lines on or off for the indicated chart.
	Timestamp – Allows selection of absolute or relative timestamps; and provides a means of turning off the timestamp.
	Scroll Rate - Sometimes referred to as "chart speed." Selects the indicated chart's scroll rate in "time per division." Scroll rate can be as fast as 0.1 sec/div and as slow as 1 hour/div, with several other rates to choose from. The rabbit and turtle buttons, and Faster (Ctrl+Z) and Slower (Ctrl+X) commands also affect scroll rate. **Note: The rate indicated in area 18 of the Chart Display figure [above] is the "global" scroll rate. It is not necessarily the same as the scroll rate set in the chart's *Properties* box.**
	Limit Lines – Allows limit lines (marking the upper and lower limits of the chart) to be displayed as solid or dotted lines; and provides a means of turning the limit lines off.
Close	Exits the Chart pull-down menu.



Properties Box

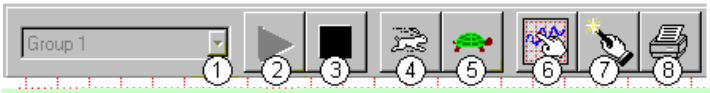
Acquire	
Start Charts Display	Starts the scrolling chart display. Does not affect the recording of data to disk.
Stop Charts Display	Stops the scrolling chart display. Does not affect the recording of data to disk.

View	
Grid Limit Lines	Used to show or hide the grid limit lines.
Grid Lines	Used to show or hide the grid lines.

Timestamp	
Absolute (Ctrl+F4)	Selects absolute time for the timestamp.
Relative (Ctrl+F5)	Selects relative time for the timestamp
Off (Ctrl+F6)	Turns the timestamp off.

Note: Many of the menu functions can be achieved by using the appropriate toolbar buttons. These buttons are discussed in the following text.

Toolbar Items



	Button	Function
1	Group Select	Indicates the current chart group. Clicking on the down arrow (▼) reveals other available groups. To select a different chart group, simply pull down the group list and select the desired group. The group list can be obtained by any of the control options: a) clicking the down arrow (▼), b) using Ctrl + G on the keyboard, c) using the keyboard up or down arrow key, d) using the page up or page down key. If using a control option other than "a," you may need to select the *group select* box by repeatable pressing the keyboard's *Tab* key until the *group select* box is selected. When this happens, the name of the currently selected group appears in white on a dark background.
2	Start Charts Display	Starts the scrolling chart display. Does not affect the recording of data to disk.
3	Stop Charts Display	Stops the scrolling chart display. Does not affect the recording of data to disk.
4	Scroll Faster	Buttons 4 and 5 provide a means of changing the chart's scroll rate (chart speed). These two buttons do not affect the scan rates of the acquisition device. The "global" scroll rate (chart speed) is indicated in the lower right-hand corner of the Chart Display Window as time/div. Possible chart speeds are indicated in the table below. **Note: The "global" scroll rate is not necessarilly the same as the chart's "actual" scroll rate (as set in the chart's Properties dialog box). The *Properties* box is accessed through the Chart pull-down menu. The table on page 4-28 includes a screen capture.**
5	Scroll Slower	
6	Setup Charts	Accesses a *Display Configuration Setup* dialog box. See page 4-31 for detailed information.
7	Chart Setup Wizard	Opens *Chart Setup Wizard* for manual or automatic configuration of the chart display. Detailed discussion of the *Chart Setup Wizard* begins on page 4-36.
8	Print Chart Display	Sends the displayed chart to the printer.

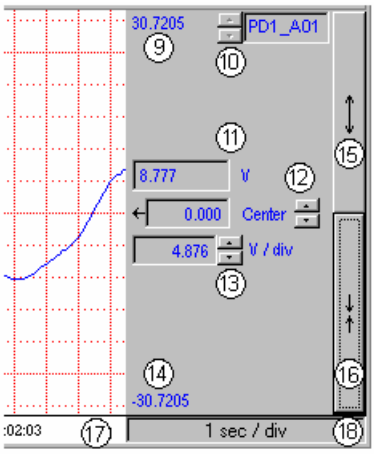
Available Chart Speeds (Scroll Rates)				
0.1 sec/div	1 sec/div	10 sec/div	2 min/div	30 min/div
0.2 sec/div	2 sec/div	30 sec/div	5 min/div	1 hr/div
0.5 sec/div	5 sec/div	1 min/div	10 min/div	

Chart and Channel Information Regions

Channels can return values in engineering units of °C, °F, °K, °R, mV, V, Hz, kHz, or *user defined* units. With exception of *user defined* units, engineering units depend on the configuration, for example, whether a voltage type or temperature type sensor is being used. The Channel Information Region is located on the right-hand side of *Personal DaqView's* Chart Display Window. The values displayed in this region are the real-time values of the selected channel.

By clicking on the up or down arrows (▲, or ▼) by the channel selection box (item 10), you can select one of a maximum of 4 channels that were assigned to that chart. This allows you to observe the chart-related information for that specific channel. You can also select a new channel for the information region by placing the cursor in (or tabbing over to) the "Center" or "Units/Div" fields and then pressing PageUp or PageDown. This is particularly useful when your *Chart Display Window* has been re-sized such that the channel selection boxes are not visible.

The following list identifies areas of the channel information region.



Channel Information Region

	Item	Description
9	Chart Maximum Scale Value (Grid Limit Line)	The value at the upper grid limit line. This is the value at the high end of the scale.
10	Channel Selection	Used to select the channel (of chart group's available channels) for which the information will be displayed. Displays the label of the channel for which information is being displayed.
11	Present Value	The real-time value of the selected channel.
12	Center	Used to display (and change) the value of the selected channel's chart centerline. Changing the value of *center* results in an automatic change of the chart's high and low end values (items 9 and 14, and possibly an automatic change of the units/div (item 13). Aside from using the *center* spinner controls to change center, you can change the center value by placing the mouse cursor in (or tabbing over to) the field and then either typing in the desired value, or using the PC keyboard up and down arrow control keys.
13	Units/Division (Provides the vertical increment of one grid box)	The units in **units/div** can be °C, °F, °K, °R, µV, mV, V, kHz, or Hz. The division referenced is one vertical grid. In the example above for PD1_A02, each vertical grid increment represents 4.876 V per division. Changing the units/division spinner controls (▲/▼) will result in an automatic adjustment of the max scale and min scale values (items 9 and 14). Aside from using the units/div triangular controls to change the value, you can change units/div by placing the mouse cursor in (or tabbing over to) the field and then either typing in the desired value, or using the PC keyboard arrow control keys.
14	Chart Minimum Scale Value (Grid Limit Line)	The value at the lower grid limit line. This is the value at the low end of the scale.
15 16	Multiply (x2), and Divide (÷2)	The Multiply(x2) push-button increases the size of the selected channel's chart by a factor of 2, while automatically adjusting the chart's high and low values. Aside from "clicking" on the Multiply/Divide controls, you can use your keyboard spacebar to control this feature once the button (15 or 16) is selected. Selection may be with mouse, or by tabbing over to the control. Making changes to a channel's chart parameters does not affect the parameters of the other channels, with the following exception: Holding the keyboard's control key down while adjusting either spinner (▲/▼) for *center* (item 12), or spinner for *units/div* (item 13) causes the parameter change to apply to all channels for the chart (not just the currently selected channel display). This feature applies to the spinners and keyboard up and down arrow keys, but not to the text input.
17	Scrolling Time	Scrolling Time is turned On or Off from the **Timestamp** pull-down menu. Time Stamp can be "absolute" (real time) or "relative." Absolute time is based on your computer clock, whereas relative time starts at 00:00:00 hours/minutes/seconds, and then continues timing in increments relative to the Scroll Rate. The chart's scroll rate can be set in the chart's *Properties* box, accessed through the **Chart** pull-down menu. Also see Properties in the table on page 4-28.
18	Global Scroll Rate	The "global" scroll rate consists of a "time per division" value which can be changed using the "faster" (rabbit) button or "slower" (turtle) button. Available Scroll Rates are from 0.1 sec/div to 1 hour/div, as indicated in the table on page 4-29. The "global" scroll rate is not necessarilly the same as the chart's "actual" scroll rate (as set in the chart's Properties dialog box). The *Properties* box is accessed through the **Chart** pull-down menu.

Accessing the Display Configuration Setup Box

The Set Up Charts button accesses a *Display Configuration Setup* dialog box. This box will also be displayed if: a) you select *Create Charts Manually* during use of the Wizard Chart Setup program, b) you select Setup from the Chart pull-down menu, c) you right-click on the chart region in *Personal DaqView's* Chart Display Window.

Note: If multiple chart groups are present in the display configuration, the current group will be selected in the display configuration tree.

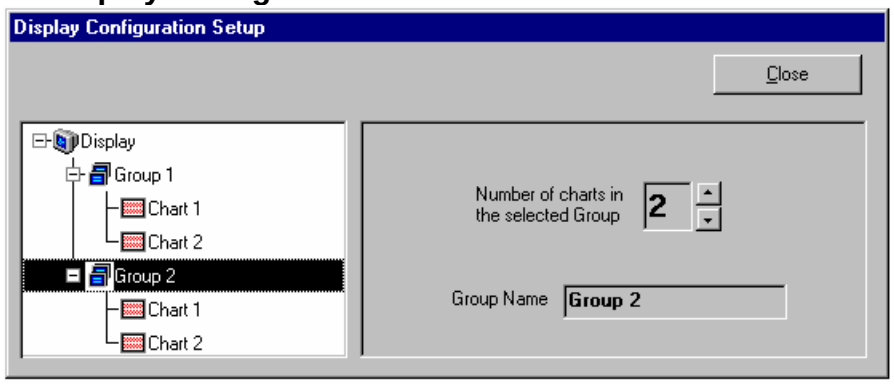
When you first click on the Set Up Charts button, a *Display Configuration Setup* box appears. A display region shows the configured structure of the groups, charts, and channels. From this box you can select the number of charts to be assigned to a specific group. With the use of the mouse cursor you can also select a chart or channel for additional editing. In addition to the text presented in the following sub-sections: ***Normal Edit*** and ***Manually Creating a Display***, you can refer to the Chart Setup Wizard section, beginning on page 4-36.

Reference Note: For your very first chart display setup, or when the configuration file has been deleted, refer to the *Chart Setup Wizard* section beginning on page 4-36.

The method you use to access the *Display Configuration Setup* window makes a difference. When you use the toolbar button or the pull-down menu's Setup selection, the *Display Configuration Setup* window appears with *the current chart display configuration intact*. With this type of access you would simply edit your existing chart display. Channel configurations do not change, with exception that newly displayed channels will be enabled.

When you access the Display Configuration Setup from the *Chart Setup Wizard*, the *Display Configuration Setup* window appears with no existing display. This allows for a "clean slate" approach to creating a chart display, as opposed to an "editing" approach. Channel configurations do not change, with exception that newly displayed channels will be enabled.

Editing a Chart Display Configuration



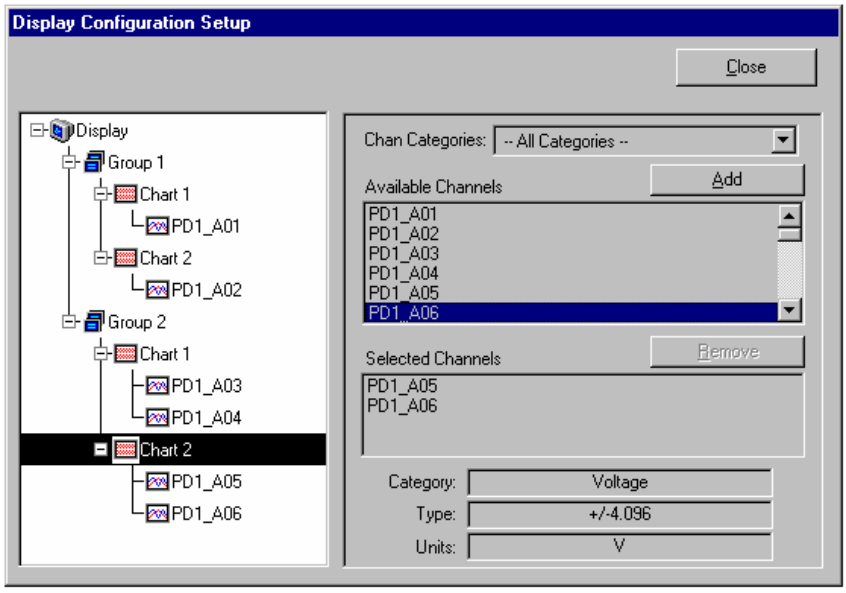
Display Configuration Setup Dialog Box with an Existing Configuration

To explain editing a configuration, we make use of an example in which assumes you want to edit Chart 1. In the following figure, Chart 1 was highlighted by clicking on it with the cursor. The *Display Configuration Setup* box then changed, allowing you to see specific channel types (such as volts only) or to "Show all Types," as in the example.

From this setup box you can add or delete charts and channels. You can:

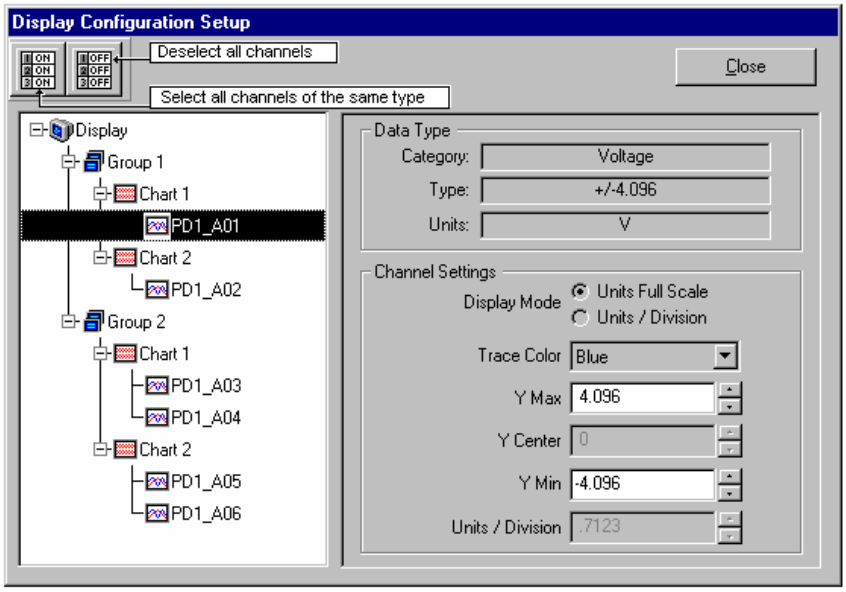
- use the *Shift or Ctrl key* in conjunction with the *cursor* to select several channels for addition or deletion
- double-click on an *available channel* to add it to the selected channels
- double-click on a *selected channel* to remove it from the selected channels list

Note: When a chart contains overlapping channels with identical points of measurement, the channels that are listed lower in the display will obscure the channels that are listed higher. In other words, the charted points of newer channels will obscure points of equal value on the older channels.



Adding Channels to Group 2, Chart 2

Another variation of the Display Configuration Setup box appears when you highlight a channel. In the following figure, channel PD1_AO1 (of Chart 1, Group 1) was selected, resulting in a new screen image. From this screen you can edit the channel setup.



Changing the Configuration of Channel PD1_A01

Personal DaqView data channels can operate in one of two modes: Units Full Scale or, Units/Div. The mode is selected by radio button.

Units Full Scale. When Units Full Scale is selected, as depicted in the above figure, you can alter Y Max and Y Min. These are the upper and lower limits of the Channel as they will appear on the chart when the channel is selected. When you change either parameter, Y Center and Units/Division are automatically adjusted. You can not directly adjust Y Center or Units/Division while "Units Full Scale" is selected. You can change Y Max and Y Min by using the up and down arrows, or by highlighting the existing value, typing in the new value, the pressing "Enter" on your PC keyboard.

Note: If the window size is changed, a chart operating in the Units Full Scale mode will maintain its *full scale* setting across the chart.

Units/Div. When Units/Div. is selected you can alter Y Center and Units/Div. Y Center is the centerline value of the chart when the channel is selected. Units/Div. is the vertical value of on chart grid increment. When you change Y Center or Units/Div. Y Max and Y Min are automatically adjusted. You can not directly adjust Y Max or Y Min while "Units/Div." Is selected. You can change Y Center and Units/Div. by using the up and down arrows, or by highlighting the existing value, typing in the new value, the pressing "Enter" on your PC keyboard.

Note: If the window size is changed, a chart operating in the Units/Div. Mode will maintain its *units per division scale* setting across the chart.

The channels in the display setup you create will be automatically enabled and will appear in chart form on *Personal DaqView's* Chart Display Window. The Channels will overlap on their assigned Chart and will be visible when the applicable Group is selected. Note that only one group of charts can be viewed at a time.
It is important to understand that other channels (those not in the display setup) maintain their existing configuration status. They are not affected by the edit of the configuration display.

You can enable additional channels from the Channel configuration window. Enabling additional channels allows you to acquire more data; however, it will not change your display on *Personal DaqView's* Chart Display Window. In other words, you can also acquire data from channels, which you do not monitor.

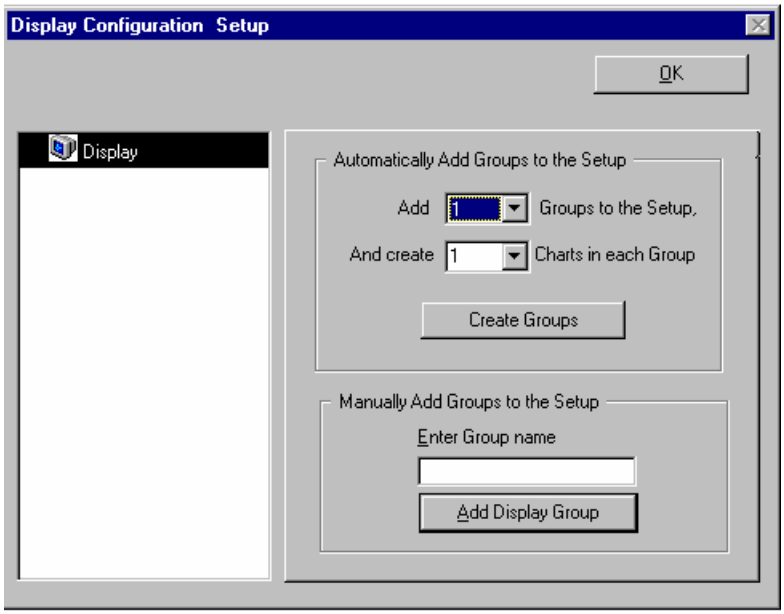
Manually Configuring a Chart Display

If you plan to have a chart setup which is not weighted evenly, i.e., different numbers of channels per chart and different numbers of charts per group, you may want to manually setup your chart display from scratch, that is, without beginning from a pre-existing display configuration. This method is arrived at from the *Chart Setup Wizard* window by clicking on the Manual Chart Setup, Create Charts button. When this button is clicked, the program exits the *Chart Setup Wizard* and enters the manual method of Display Configuration Setup. Although this method is referred to as "manual," it still contains some automatic elements, such as *Automatically Add Groups to the Setup*.

Note: Even if an unevenly distributed chart display is desired, you can always edit a pre-existing chart setup, or create a new setup by one of the *Chart Setup Wizard's* automatic methods, and then edit the setup.

To manually setup your chart display using the *clean slate* approach (as opposed *to editing an existing display*) perform the following steps.

1. Select the Chart Setup Wizard from the Charts pull-down menu of *Personal DaqView's* Chart Display Window. The Wizard setup window appears (see following figure).
2. Click on the *Manual Chart Setup, Create Charts* button. The Display Configuration Setup box appears. Since the previous display configuration was reset, no groups or charts are seen in the display area on the left side of the screen.



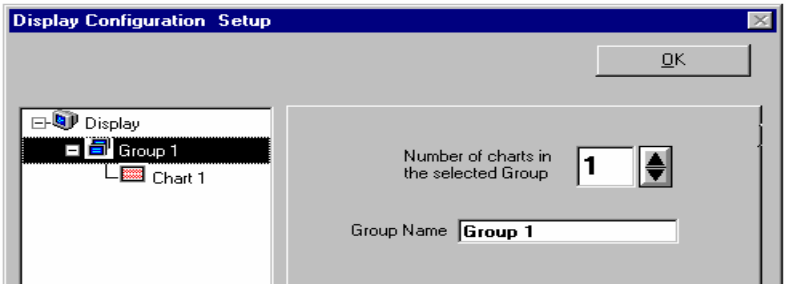
Display Configuration Setup, A "Clean Slate" Approach

3. As seen in the previous figure for the "Clean Slate" approach, you have two options at this point. Perform (3a) or (3b) as appropriate. Option (3a) is typically used.

   (3a) *Automatically add groups to the setup*. Enter the number of groups and charts desired by using the cursor and typing in the value, or by using the pull-down arrows (▼) and making the appropriate selections; then click on the Create Groups button.

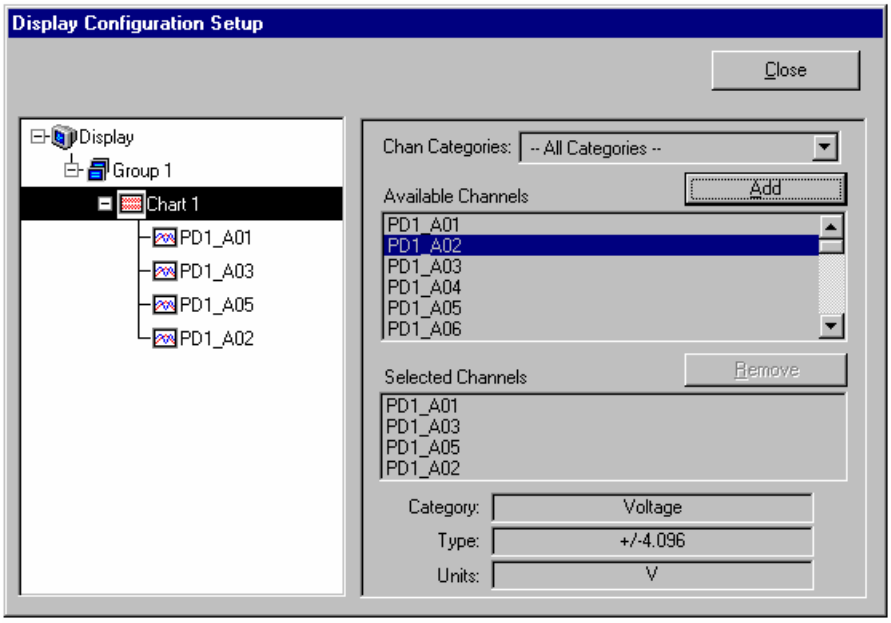
   (3b) *Manually add groups to the setup*. Type in the name of the chart group; then click on the Add Display Group button.

   The Display Configuration Setup screen changes to show chart groups, and the number of charts for the selected (black highlighted) chart group. From this screen you can change the number of charts in a group, as well as change the group name.



Adding one Chart to Group 1

4. Change the number of charts per group if desired.
5. Change the group name if desired.
6. Click on a group to see the chart(s) assigned to the group. In the above example there is one group with one chart.
7. Click on a chart to assign channels to the chart. A screen similar to the following will appear.



Assigning Channels to Group 1, Chart 1

8. Choose channels for the selected chart. You can select up to 4 overlapping channels per chart. Add or delete channels as follows:

    - Highlight an available channel using the cursor and left-hand mouse button; then click the Add button. Repeat for each channel to be added.
    - Double-click on a channel in the *available channels* list to add the channel to the display.
    - Double-click on a channel in the *selected channels* list to remove the channel from the display.
    - Hold down the keyboard's *Shift Key* and use the left-hand mouse button to select a block of consecutive available channels (up to 4); then click the Add button. Example: PD1_A03, PD1_A04, PD1_A05, PD1_A06.
    - Hold down the keyboard's Ctrl button and use the left-hand mouse button to select up to 4 available channels (these can be non-consecutive); then click the Add button.

    Example: PD1_A01, PD1_A03, PD1_A05, PD1_A07.

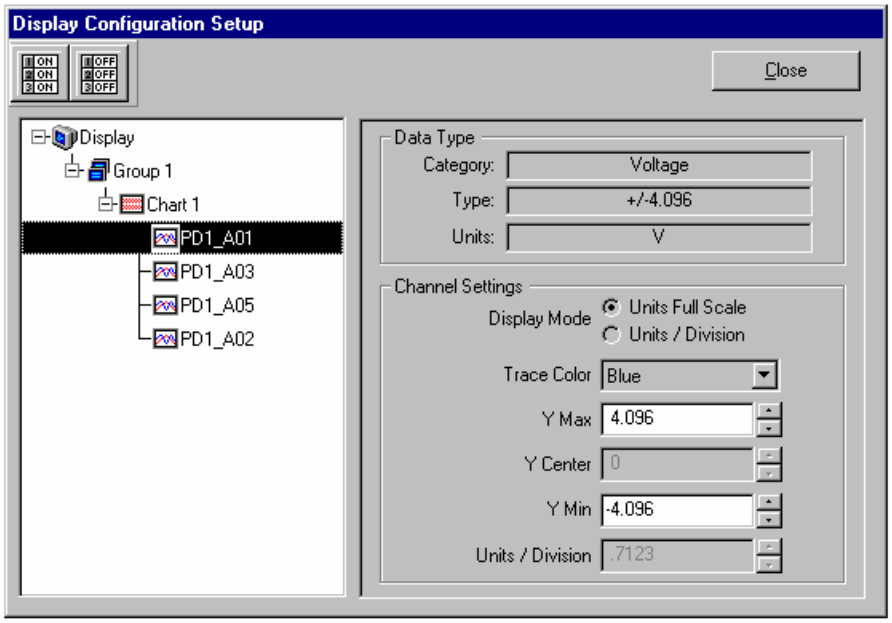
Note: You can also remove channels in a similar manner by highlighting a channel(s) in the Selected Channels box, and then clicking on the Remove button.

Note: When a chart contains overlapping channels and the channels share values such that their traces reside on top of each other, then the channels that are listed lower in the display list (the most recently added channels) will obscure the channels higher in the list (those which were added first).

9. In the display area (on the left-hand side of the screen) click on a channel to check the channel's configuration and to re-configure the channel, if desired. The *Display Configuration Setup* Window will appear similar to that in the following figure.

Note: Changing the display configuration does not change the existing channel configuration setup. It only changes how the chart groups, charts, and channels will be displayed.

The following figure shows two "radio buttons" on the screen. These buttons allow you to select the method of adjusting the display mode. From this screen you need to choose *Units Full Scale* or *Units/Div*.



Adjusting Channel Setup for Channel 1

Remember, you can enable additional channels from the Channel configuration window. Enabling additional channels allows you to acquire more data; however, it will not change your chart display on *Personal DaqView's* Chart Display Window. In other words, you can acquire data from channels, which you do not monitor.

Chart Setup Wizard



Introduction

Chart Setup Wizard is a feature used by many programs, including *Personal DaqView*, and *PostView*. The feature allows you to set up your initial chart display configuration using an automated method, or manually create a new display configuration. The following points are important in regard to the *Chart Setup Wizard*.

- You can edit the chart display by accessing the *Display Configuration Setup* dialog box from the **Chart** pull-down menu or by clicking of the Chart Setup button in the *Chart Display Window* toolbar. This method does not use the *Chart Setup Wizard* and **does not reset your chart display configuration setup.**
- Activating the *Chart Setup Wizard* **will reset your chart Display Configuration Setup**.
- Activating the *Chart Setup Wizard* will not reset your Channel configuration.

The *Chart Setup Wizard* window will appear when you attempt to run *Personal DaqView* charts for the very first time, as well as when a configuration file does not exist and you attempt to run charts. When a configuration file already exists, you can easily access the *Chart Setup Wizard* by selecting *Wizard* in the **Chart** pull-down menu, or by clicking on the *Chart Setup Wizard* button (located in the *Chart Display Window's* toolbar). It is important to realize that running the *Chart Setup Wizard* will result in a reset of your display setup; it will not, however, change your channel configuration (with the exception of new display channels now enabled).

The chart setup determines how your *Personal DaqView Chart Display Window* will appear in regard to the following:

- number of chart groups available for viewing
- number of charts shown for each selected group
- the number of overlapping channels in each chart (not to exceed 4)

You can choose to manually create a configuration, or have one created automatically. The automatic setup method offers three choices: Simple, Moderate, and Advanced.

Note: You can use *Chart Setup Wizard* to quickly set up a large number of charts. You can then fine-tune the layout manually via the *Manual Chart Setup* feature.

A *Manual Chart Setup, Create Charts* button allows you to bypass the *Chart Setup Wizard* and enter a manual mode. This option makes use of *Personal DaqView's* Display Configuration feature (discussion on manually creating a chart display begins on page 4-33). Manual Chart Setup allows you to vary the number of assigned channels per chart, as well as vary the number of charts per group.

Note: The standard version of *Personal DaqView* has a *Chart Setup Wizard* that, in regard to *Automatic Chart Creation*, is restricted to the *Simple* mode.

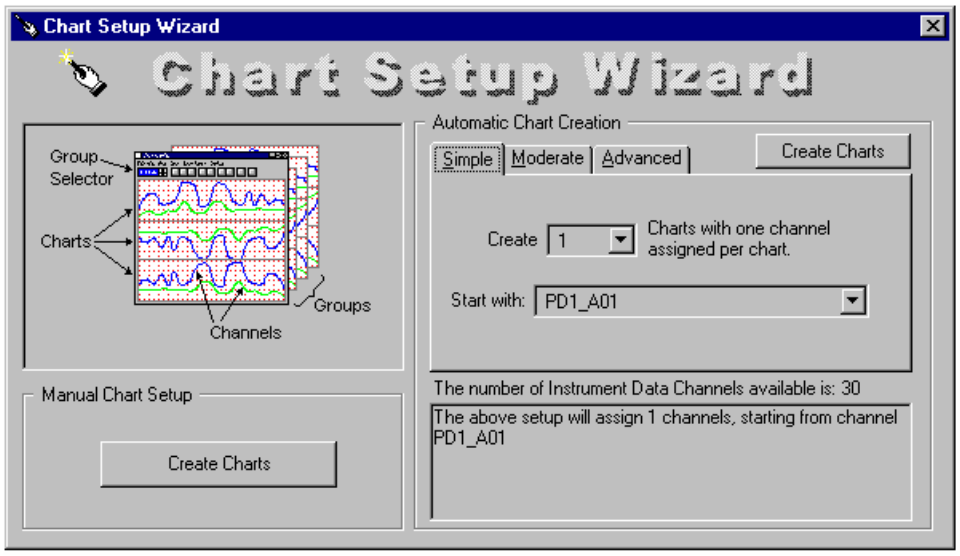


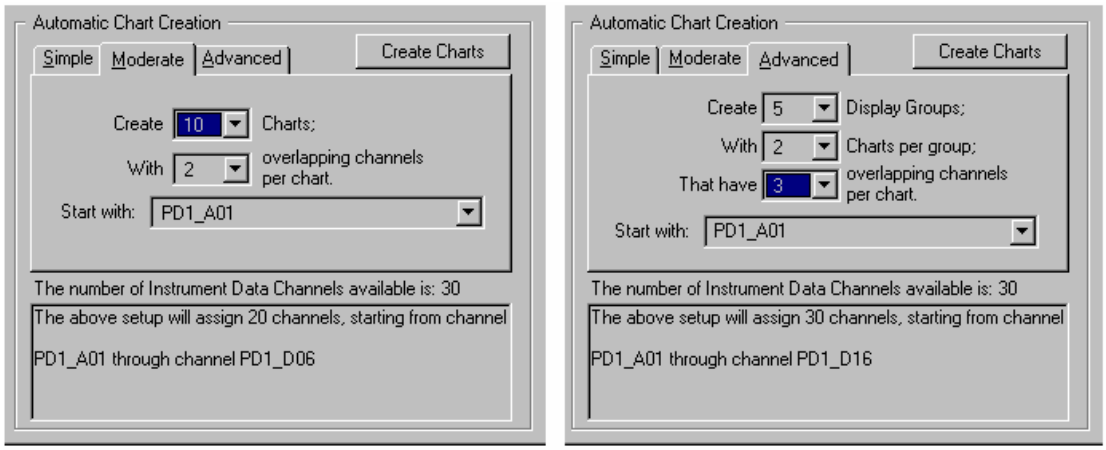
Chart Setup Wizard, Simple Mode

***Chart Setup Wizard*, Automatic Setup Options**			
Setup Type	**Group Setup (for Chart Groups)**	**Chart Setup**	**Channel Setup**
Simple	1 group only	Up to 16 charts	1 channel per chart
Moderate	1 group only	Up to 16 charts	Up to 4 overlapping channels per chart
Advanced	Up to 64 groups	Up to 16 charts per group	Up to 4 overlapping channels per chart

Note: The *Manual Chart Setup, Create Charts* button (above figure) allows you to exit the *Chart Setup Wizard* and manually create your chart display without use of the automatic functions available in *Chart Setup Wizard*.

Automatic Display Setup using the Chart Setup Wizard

The previous figure shows the Simple mode dialog box for Automatic Chart Creation. The following figures show the Moderate and Advanced mode dialog boxes. These two modes do not apply to the standard version of *Personal DaqView*.



Dialog Boxes for Moderate and Advanced Modes of Automatic Chart Creation

It is a simple task to create chart display configurations using the automatic method. The following steps apply to this feature of *Chart Setup Wizard*.

1. Select Wizard from the Chart pull-down menu of *Personal DaqView's* Chart Display Window. The Wizard setup window appears.
2. Select the desired mode tab (Simple, Moderate, or Advanced).
3. Use the pull-down arrows (▼), or use the cursor and type in a new value to make selections for the number of groups, charts, and channels as applicable.
4. If you desire to start with a channel other than the default channel (first available channel), use the pull-down arrow and select the desired starting channel number.
5. When your setup is complete, click on the *Automatic Chart Creation*, *Create Charts* button. A percentage of completion bar will appear, followed by the Channel Setup box.
6. Make appropriate configuration changes, if any are desired, including enabling additional channels; then click on the OK button. After clicking OK, the Chart Display Window appears and you can begin running charts. Note that the *Channel Setup* section of this chapter contains related information.

The channels in the setup you create will be automatically enabled and will appear in chart form on *Personal DaqView's* Chart Display Window. The Channels will overlap on their assigned Chart and will be visible when the applicable Group is selected. Note that only one group of charts can be viewed at a time.
As mentioned earlier, you can enable additional channels from the Channel configuration window. Enabling additional channels allows you to acquire more data to disk; however, it will not change your display on *Personal DaqView's* Chart Display Window. In other words, you can acquire data from channels, which you do not monitor.

Bypassing Automatic Chart Setup

You can bypass *Chart Setup Wizard* by clicking on the *Manual Chart Setup, Create Charts* button in *Chart Setup Wizard's* Chart Display Window. After selecting this option you will be using the Display Configuration Setup dialog boxes to create a display from scratch, i.e., using a "clean slate" approach. This is method is detailed with an example, in the section *Manually Creating a Display*, beginning on page 4-33.

Using Personal DaqView with Microsoft Excel™

OverView …… 39

First Time Use …… 40

Configuration Reference …… 42
- Toolbar Buttons …… 42
- Tools Pull-Down Menu ……43
- Channel Configuration Window …… 44

Auto-Load Function ……44
- Disabling Auto-Load …… 44
- Enabling Auto-Load …… 44

Performance of Personal DaqViewXL ……45
- Configuring an Acquisition ……45
- Real-Time Charting ……46
- Data Header ……47

Hints and Tips …… 48



When using Personal Daq modules to acquire data, computer *energy save* modes can cause false data readings. Prior to using Personal Daq modules, ensure your computer's *energy save* mode is disabled. If needed, consult your computer's user manual to disable *energy save* (*power suspension*) modes.

Overview

Personal DaqView includes a software component add-in that installs into *Microsoft Excel ™* and provides complete setup and data acquisition within *Excel* under *Windows2000* and *XP*. Acquired data is immediately placed in an active spreadsheet, analyzed and graphically displayed.

After *Personal DaqView* has been installed, you can configure your data acquisition system easily by accessing the desired window from the program's toolbar, i.e., with use of the ***Configure Devices***, ***Configure Channels***, or ***Configure Acquisition*** buttons. Each column in the channel-configuration window corresponds to a specific parameter in the data acquisition system. Note that the channels can be configured individually.

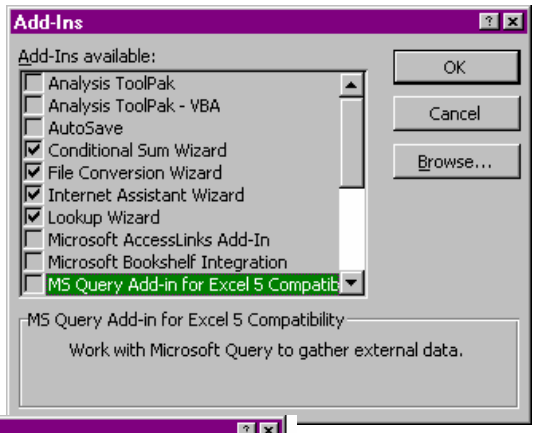
In regard to creating data reports, you can embed *Excel* "spreadsheet objects" directly into a word processing program, add text, and print the end result. To revise the report's text, just edit the document as needed. To update the spreadsheet, double-click on the embedded "spreadsheet object" to access *Personal DaqView*; then create and embed a new "spreadsheet object."

Personal DaqView supports all models of the Personal Daq family, including expansion modules and provides the engineering unit output required for each configured channel on an individual basis. For example, data for a voltage channel can be returned in volts, while data from a thermocouple is being returned in degrees.

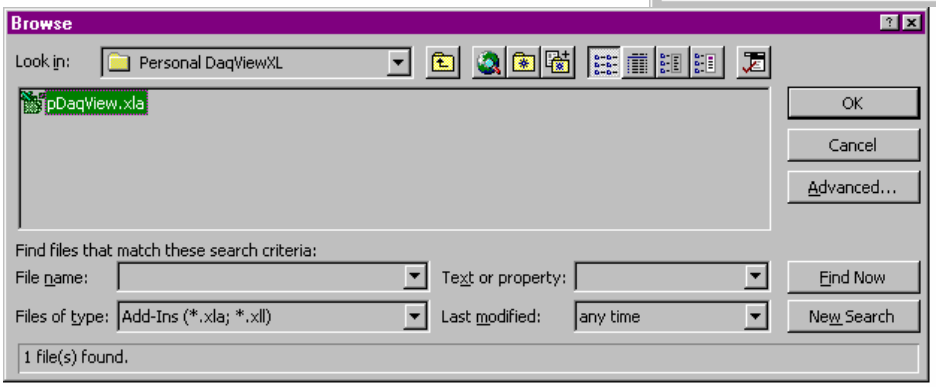
You can use *Personal DaqView's* "scale and offset" feature to automatically convert channel data to custom units and then have the *scale* and *offset* applied to the data prior to its appearing in the spreadsheet. The scale and offset feature is especially handy for transducers such as strain gages because it converts the return voltage value to units such as pounds or units of pressure.

First Time Use

Perform the following steps to load *Personal DaqView* into *Excel*:

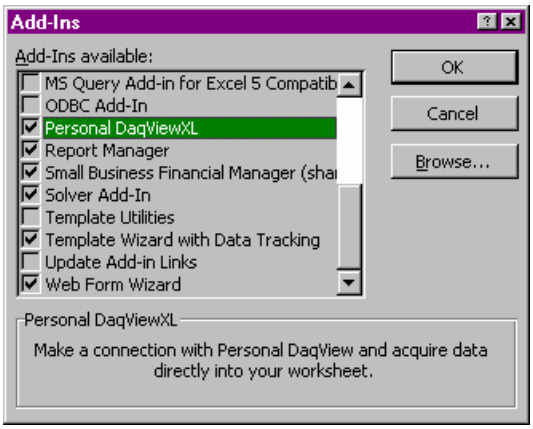


1. Start *Excel* (refer to Microsoft documentation as needed).
2. Select the "Tools" menu and click on "**Add-Ins**." This will open the "**Add-Ins**" dialog box.
3. Click the "**Browse**" button and locate the file **`pDaqView.xla`**. This file is located in the install directory specified by the setup program. Click **OK**.



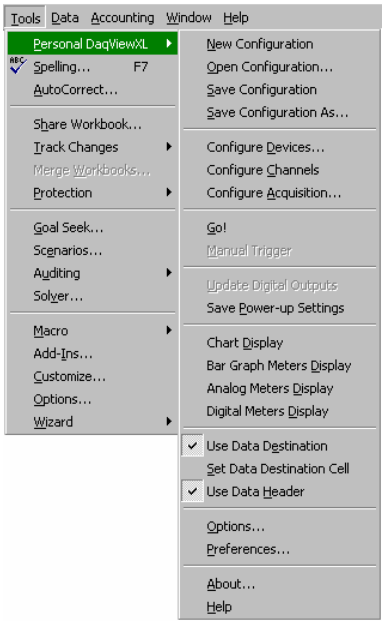
Browse Option Dialog Box

4. Select *Personal DaqViewXL* from the "Add-Ins Available" list. Make sure there is a check next to *Personal DaqViewXL*, and then click **OK**.

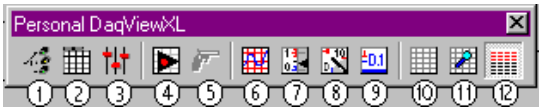


Add-Ins Dialog Box with Personal DaqViewXL Selected

5. *Excel* loads the "add-in," creating a *Personal DaqViewXL* item in the Tools pull-down menu and a toolbar.



Personal DaqViewXL Menu Options

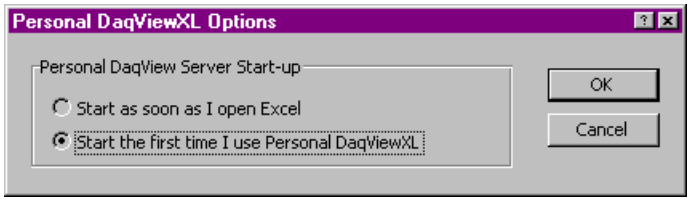


(1) Configure Devices
(2) Configure Channels
(3) Configure Acquisition
(4) Go!
(5) Manual Trigger
(6) Chart Display
(7) Bar Graph Meters
(8) Analog Meters
(9) Digital Meters
(10) Use Data Destination
(11) Set Data Destination Cell
(12) Use Data Header

Personal DaqViewXL Toolbar Options

After *Personal DaqView* is installed in this manner, it will automatically load whenever you run *Excel*.

6. If you want to start *Personal DaqView* when *Personal DaqViewXL* is used for the first time:
   a) Click the *Personal DaqViewXL* selection (located in Excel's *Tools* pull-down menu).
   b) Choose "**Options…**" (the Options Dialog Box appears).
   c) Select "Start the first time I use Personal DaqViewXL."
   d) Click "**OK**."



Personal DaqViewXL Start-up Options Dialog Box

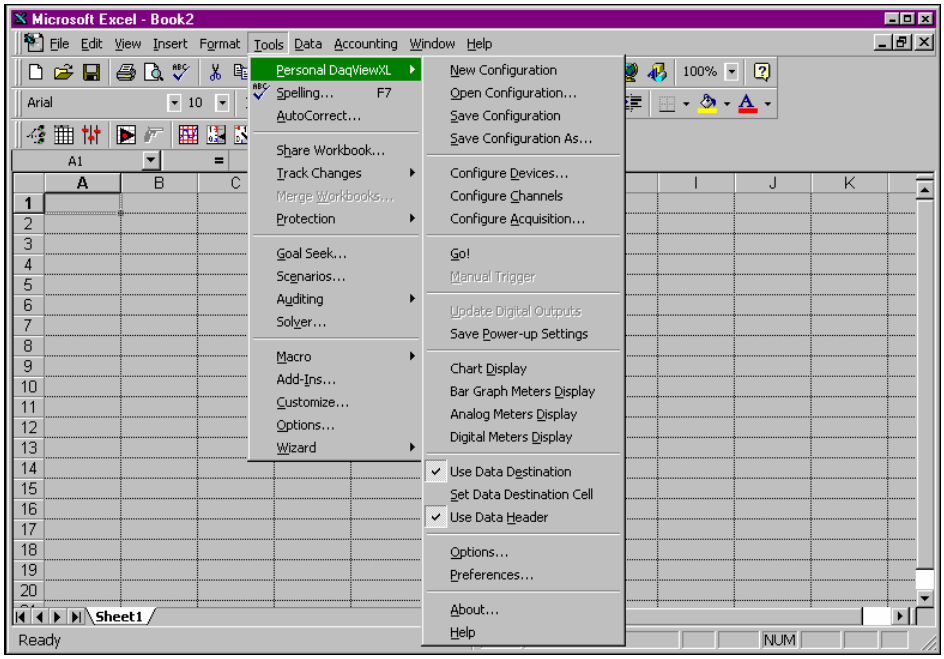
7. If you want to start *Personal DaqView* as soon as Excel is opened:
   a) Click the *Personal DaqViewXL* selection (located in Excel's *Tools* pull-down menu).
   b) Choose "**Options…**" (the Options Dialog Box appears).
   c) Select "Start as soon as I open Excel."
   d) Click "**OK**."

Note: After you have launched *Excel* and *Personal DaqViewXL* has been activated, the following dialog box appears if *Personal DaqView* is not running. If this is the case, click "**Yes**" to start *Personal DaqViewXL*.



Personal DaqView Startup Confirmation

The following figure shows a partial screen capture of a Microsoft Excel spreadsheet. In the figure, the *Personal DaqViewXL* item has been selected from the Tools pull-down menu. *Personal DaqViewXL's* toolbar can also be seen in the figure.



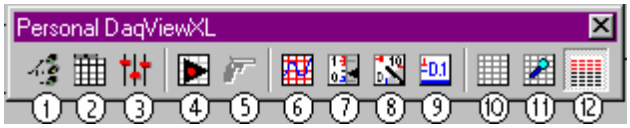
Microsoft Excel Spreadsheet showing Personal DaqViewXL Menu Options

Configuration Reference

The following information is intended to serve as a quick reference regarding channel configuration. You should refer to the user's manual for more detailed information regarding *Personal DaqView*, as needed.

Toolbar Buttons

Note: The functions that can be obtained through the toolbar can also be obtained through the pull-down menus.



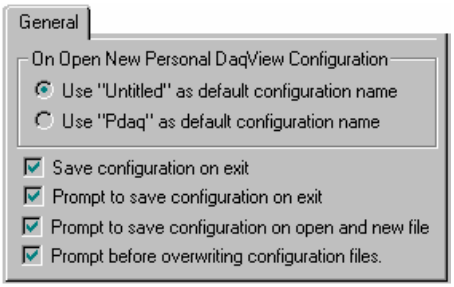
Personal DaqViewXL Toolbar

	Button	Function
1	Configure Devices	Opens the *Select Active Devices* window.
2	Configure Channels	Opens the *Channel Configuration Spreadsheet*. If the window is already open, pressing this button gives the window focus.
3	Configure Acquisition	Opens the *Configure Acquisition Dialog*. Various acquisition parameters can be set from this dialog.
4	Go!	Starts an acquisition and inserts the data into the current spreadsheet destination cell. Pressing the button again stops the acquisition.
5	Manual Trigger	Manually triggers the acquisition when applicable.
6	Chart Display	Opens the *Chart Display*, where data is displayed in graphical in a chart format. If *Chart Display* is already open, pressing this button gives the window focus.
7	Bar Graph Meters Display	Opens the *Bar Graph Meters*, where data is displayed in a bar graph format. If *Bar Graph Meters* is already open, pressing this button gives the window focus.
8	Analog Meters Display	Opens the *Analog Meters Display*, where data is displayed in an analog meter format. If *Analog Meters Display* is already open, pressing this button gives the window focus.
9	Digital Meters Display	Opens the *Digital Meters Display*, where data is displayed in a digital meter format. If *Digital Meters* is already open, pressing this button gives the window focus.
10	Use Data Destination	Turns the *Use Data Destination* feature on or off.
11	Set Data Destination Cell	Sets the *data destination cell* to the "active" cell in the current workbook.
12	Use Data Header	Turns the *Use Data Header* feature on or off.

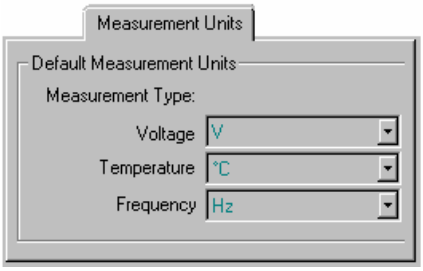
Tools Pull-Down Menu

When the *Personal DaqViewXL* add-in is loaded, a *Personal DaqViewXL* menu is added to *Excel's* ***Tools*** *pull-down menu*. You can use the menu items to perform all actions that are available from *Personal DaqViewXL's* toolbar. Note that some of the pull-down items have no toolbar equivalent.

	Menu Item	Function
1	New Configuration	Provides a means of creating a new configuration from default settings.
2	Open Configuration…	Provides a means of opening an existing configuration file. This feature may be used to open configuration files created by the *Personal DaqView* application.
3	Save Configuration	Saves the configuration file under its existing filename. Note that you can open *Personal DaqViewXL* files from the *Personal DaqView* application.
4	Save Configuration As…	Copies the current file for saving under a different filename.
5	Configure Devices…	Opens the *Select Active Devices* window.
6	Configure Channels	Opens the *Channel Configuration Spreadsheet*. If the window is already open, pressing this button gives the window focus.
7	Configure Acquisition…	Opens the *Configure Acquisition Dialog*. Various acquisition parameters can be set from this dialog.
8	Go!	Starts an acquisition and inserts the data into the current spreadsheet destination cell. Selecting the item again stops the acquisition.
9	Manual Trigger	Manually triggers the acquisition when applicable.
10	Update Digital Outputs	Updates the digital output channels. This item is only enabled when the digital output channels are not updated and the acquisition state is idle.
11	Save Power-up Settings	Saves the current power-up settings for all devices in the configuration. The power-up settings take effect when the device is first plugged into an active USB port.
12	Chart Display	Opens the *Chart Display*, where data is displayed in graphical in a chart format. If *Chart Display* is already open, selecting this item gives the window focus.
13	Bar Graph Meters Display	Opens the *Bar Graph Meters*, where data is displayed in a bar graph format. If *Bar Graph Meters* is already open, selecting this item gives the window focus.
14	Analog Meters Display	Opens the *Analog Meters Display*, where data is displayed in an analog meter format. If *Analog Meters Display* is already open, selecting this item gives the window focus.
15	Digital Meters Display	Opens the *Digital Meters Display*, where data is displayed in a digital meter format. If *Digital Meters* is already open, selecting this item gives the window focus.
16	Use Data Destination	Turns the *Use Data Destination* feature on or off.
17	Set Data Destination Cell	Sets the *data destination cell* to the "active" cell in the current workbook.
18	Use Data Header	Turns the *Use Data Header* feature on or off.
19	Options…	Opens *Personal DaqViewXL's Options* Dialog.
20	Preferences…	Opens *Personal DaqViewXL's Preferences* Dialog. From this dialog box you can use the **General** and **Measurement Units** dialog boxes as described below.



General – Allows you to select "untitled" or "pDaq" configurations for the default filename used by Personal Daq. Note that *Personal DaqView* automatically loads the last saved configuration file. The lower portion of the General screen pertains to configuration file settings.



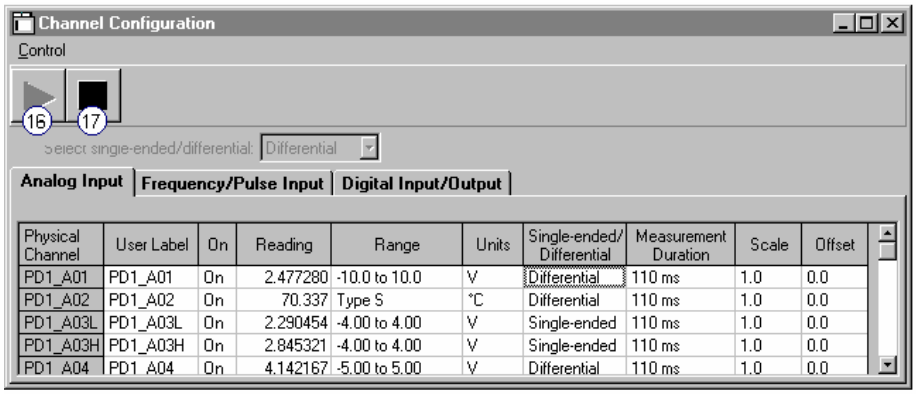
Measurement Units – Allows you to set the desired default units for the *Channel Configuration Window's* spreadsheet. Note that scale and offset automatically reset to a pre-designated default according to the unit selected. Changing the *measurement unit defaults* does not immediately affect the *Channel Configuration Window's* spreadsheet columns. For example: If you set your voltage default to **µV**, but the channels are set with **V** units, they will retain the **V** units. However, changing these channels to temperature, then back to voltage would result in the default units (**µV**) and the associated scale & offset. The new default units are also applied to all channels when a new configuration file is created.

Preferences Screen Captures (Condensed Images)

21	About…	Provides version information for the active *Personal DaqViewXL* add-in and the *Personal DaqView* application.
22	Help…	Launches the *Personal DaqViewXL* section of the *Personal DaqView* help file.

Channel Configuration Window

You can bring up the *Channel Configuration Window* by using the *Configure Channel Settings* button (9) on the *Main Control Window* toolbar, or by selecting *Configure Channels* from the *Personal DaqViewXL* pull-down menu.
The *Channel Configuration Window* contains a **Control** pull-down menu and a 2-button toolbar. The window displays any one of three tabbed spreadsheets for selecting the type of channel to be configured. The tabs are labeled Analog Input, Frequency/Pulse Input, and Digital Input/Output.



Channel Configuration Window, Selected for Analog Input Spreadsheet

Toolbar

The *Channel Configuration Window* consists of two buttons (items 16 and 17) as identified by the previous figure and the following table.

	Button	Function
16	Enable Readings Column	Activates the *Channel Configuration Window's* reading column. Does not affect the recording of data to the destination spreadsheet.
17	Disable Readings Column	Stops the *Channel Configuration Window's* reading column. Does not affect the recording of data to the destination spreadsheet.

Channel Configuration Window Pull-down Menu

The channel configuration window contains one pull-down menu labeled, **Control**. This menu provides a means of enabling and disabling the *readings* column, as do toolbar items (16 and 17).

Auto-Load Function

Excel's "Add-Ins" dialog box has a check-box for enabling (✓) or disabling an automatic loading of *Personal DaqViewXL* each time you run *Excel*. To enable or disable the auto-load feature, refer to the following two paragraphs as needed.

Disabling Auto-Load

If you want to avoid an automatic loading of *Personal DaqViewXL* each time you run *Excel*, perform the following steps:

1. Start *Excel*.
2. Select the "Tools" menu and click on "Add-Ins." The "Add-Ins" dialog box opens.
3. Locate the *Personal DaqViewXL* line in the "Add-Ins Available" list box and click on it. This should remove the checkmark next to the item.
4. Click "**OK**."

Personal DaqViewXL will no longer auto-load with *Excel*. See the following paragraph for simple steps on enabling the *Personal DaqViewXL* auto-load feature.

Enabling Auto-Load

If you want *Personal DaqViewXL* to automatically load each time you run *Excel*, perform the following steps:

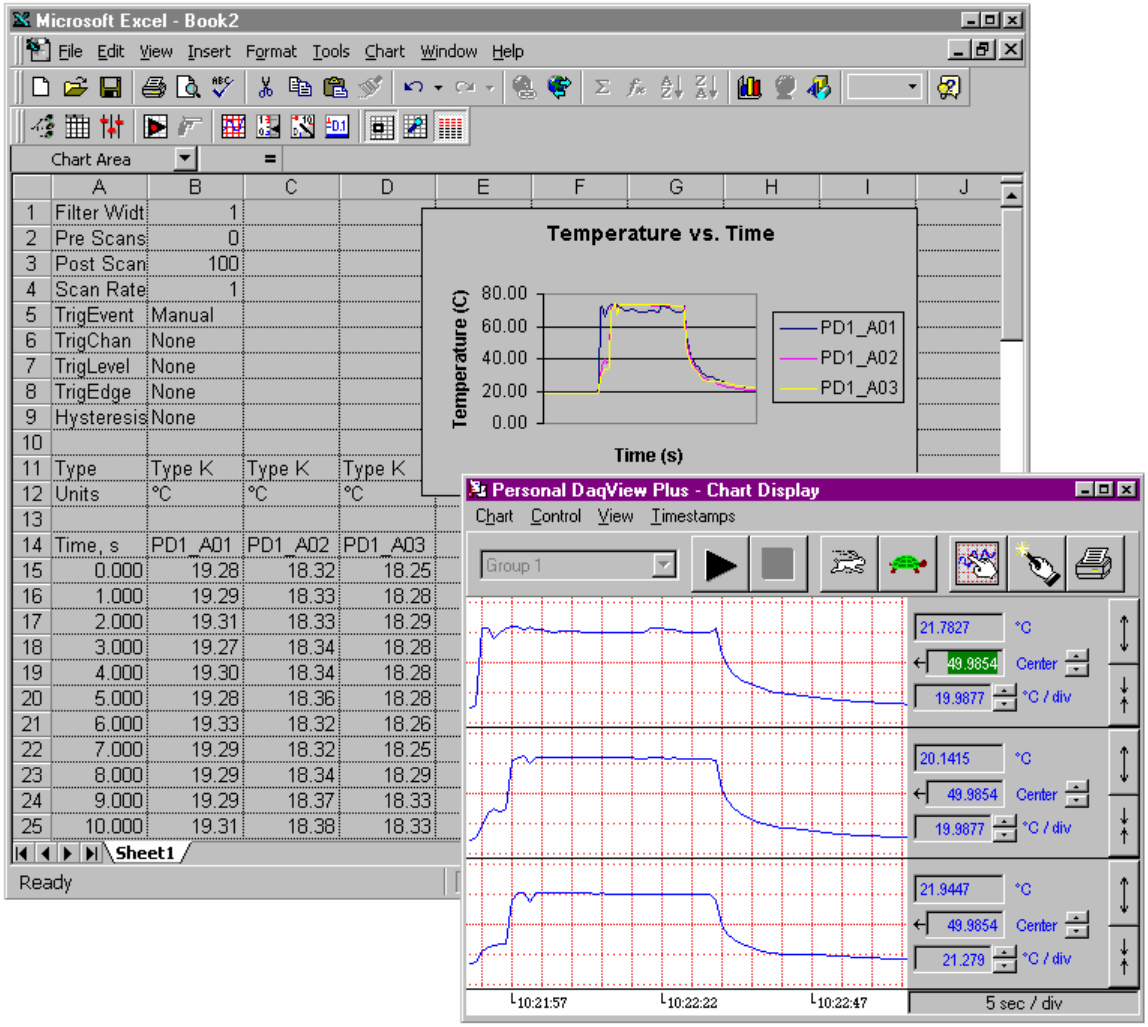
1. Start *Excel*.
2. Select the "Tools" menu and click on "Add-Ins." The "Add-Ins" dialog box opens.
3. Locate the *Personal DaqViewXL* line in the "Add-Ins Available" list box and click on it. This should insert a checkmark (✓) next to the item.
4. Click "**OK**."

Personal DaqViewXL will now auto-load with *Excel*. See the previous paragraph for simple steps on disabling the *Personal DaqViewXL* auto-load feature.

Performance of Personal DaqView with Excel

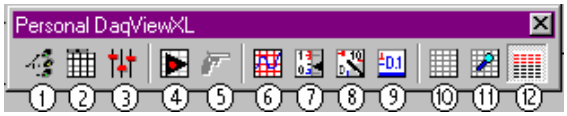
An understanding of these basics will let you explore each of *Personal DaqViewXL's* many features.

Note: The *Hints and Tips* section contains suggestions for improving your productivity with *Personal DaqView* and *Microsoft Excel*. If you are not proficient with *Excel*, please refer to your *Excel* documentation as needed. The more you know about *Excel*, the better.



	A	B	C	D
1	Filter Widt	1		
2	Pre Scans	0		
3	Post Scan	100		
4	Scan Rate	1		
5	TrigEvent	Manual		
6	TrigChan	None		
7	TrigLevel	None		
8	TrigEdge	None		
9	Hysteresis	None		
10				
11	Type	Type K	Type K	Type K
12	Units	°C	°C	°C
13				
14	Time, s	PD1_A01	PD1_A02	PD1_A03
15	0.000	19.28	18.32	18.25
16	1.000	19.29	18.33	18.28
17	2.000	19.31	18.33	18.29
18	3.000	19.27	18.34	18.28
19	4.000	19.30	18.34	18.28
20	5.000	19.28	18.36	18.28
21	6.000	19.33	18.32	18.26
22	7.000	19.29	18.32	18.25
23	8.000	19.29	18.34	18.29
24	9.000	19.29	18.37	18.33
25	10.000	19.31	18.38	18.33

Personal DaqViewXL Spreadsheet with Toolbar and Displays



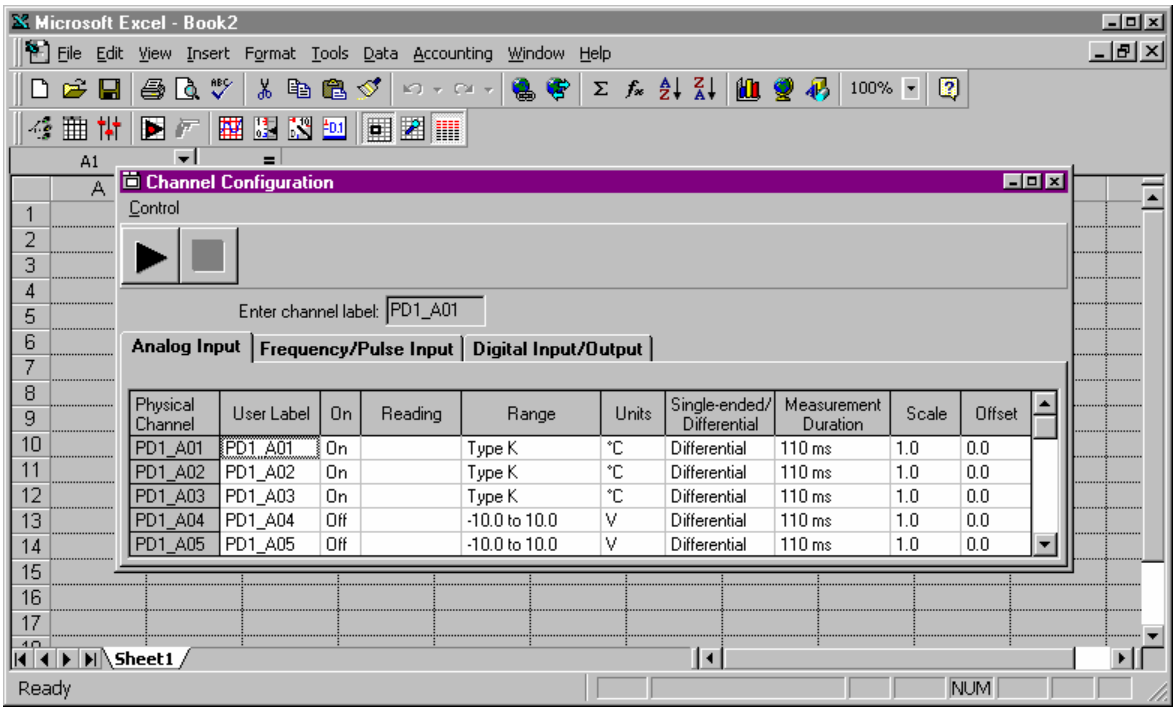
Personal DaqViewXL Toolbar

(1) Configure Devices
(2) Configure Channels
(3) Configure Acquisition
(4) Go!
(5) Manual Trigger
(6) Chart Display
(7) Bar Graph Meters
(8) Analog Meters
(9) Digital Meters
(10) Use Data Destination
(11) Set Data Destination Cell
(12) Use Data Header

Configuring an Acquisition

To set up your configuration, once *Personal DaqViewXL* has been installed; complete the following steps:

1. Enter *Excel.*



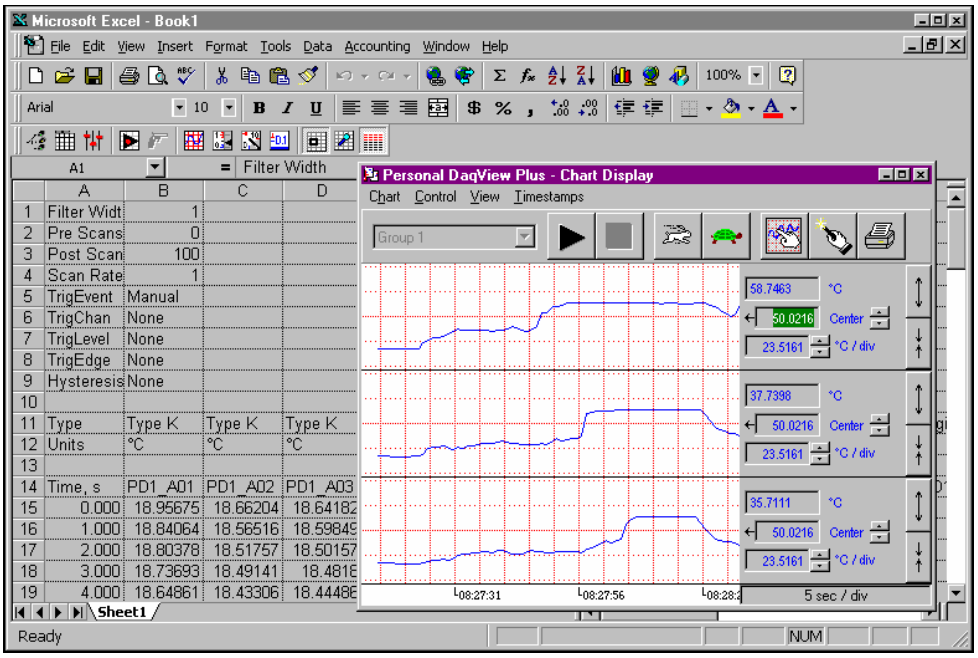
Personal DaqView Channel Configuration Screen within Excel

2. Configure acquisition settings using the ***Configure Devices, Configure Channels,*** and ***Configure Acquisition*** toolbar buttons or tool pull-down menu selections.

These steps launch *Personal DaqView's* configuration screens. Note that the *Personal DaqView* configuration information, found in the Personal Daq User's Manual, still applies to *Personal DaqViewXL*. Also see the section in this user's guide, *Configuration Reference*, beginning on page 4-42.

Real-Time Charting

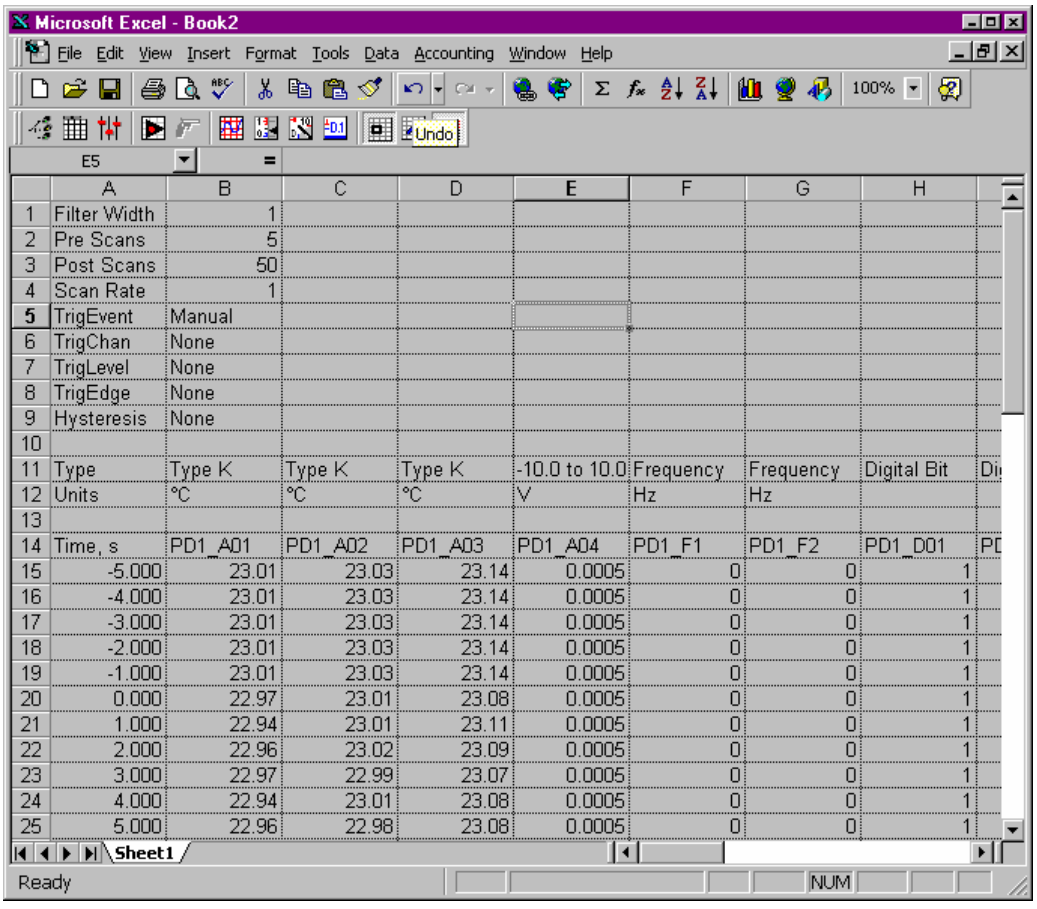
Personal DaqViewXL provides a real-time strip-chart window (as in *Personal DaqView*) for graphical presentation of data. You can select up to 16 channels of data, with one channel assigned to each graph. The strip-chart representation is continuous while data is being collected. All 16 charts are synchronized and have an adjustable scrolling speed that can be set independent of the sampling rate. Selecting the *Charts* icon in the *Personal DaqViewXL* toolbar will activate the following window. The chart selection must be made prior to initiating an acquisition (selecting the *Go!* icon).



Personal DaqViewXL Real-Time Charting

Data Header

The optional data header supplies the global acquisition parameters and the configuration of each channel and places this information directly in the spreadsheet along with acquired data. Examples of frequently used header information include: channel gain, thermocouple type, units, and channel label. To activate Data Headers select the *Use Data Headers* icon in the *Personal DaqViewXL* toolbar.



	A	B	C	D	E	F	G	H
1	Filter Width	1						
2	Pre Scans	5						
3	Post Scans	50						
4	Scan Rate	1						
5	TrigEvent	Manual						
6	TrigChan	None						
7	TrigLevel	None						
8	TrigEdge	None						
9	Hysteresis	None						
10								
11	Type	Type K	Type K	Type K	-10.0 to 10.0	Frequency	Frequency	Digital Bit
12	Units	°C	°C	°C	V	Hz	Hz	
13								
14	Time, s	PD1_A01	PD1_A02	PD1_A03	PD1_A04	PD1_F1	PD1_F2	PD1_D01
15	-5.000	23.01	23.03	23.14	0.0005	0	0	1
16	-4.000	23.01	23.03	23.14	0.0005	0	0	1
17	-3.000	23.01	23.03	23.14	0.0005	0	0	1
18	-2.000	23.01	23.03	23.14	0.0005	0	0	1
19	-1.000	23.01	23.03	23.14	0.0005	0	0	1
20	0.000	22.97	23.01	23.08	0.0005	0	0	1
21	1.000	22.94	23.01	23.11	0.0005	0	0	1
22	2.000	22.96	23.02	23.09	0.0005	0	0	1
23	3.000	22.97	22.99	23.07	0.0005	0	0	1
24	4.000	22.94	23.01	23.08	0.0005	0	0	1
25	5.000	22.96	22.98	23.08	0.0005	0	0	1

Personal DaqViewXL Data Header

Hints and Tips

- If *Personal DaqView* is running as a stand-alone application when you start to use *Personal DaqViewXL*, *Personal DaqView* will switch to a special "server mode." This "server mode" supports transactions with "client" applications such as *Excel*. Note that you must close *Excel* prior to using *Personal DaqView* again as a stand-alone application.
- When you acquire data into a worksheet, some of the data may be displayed as a string of pound characters (######). *Excel* will use these characters when a column is not wide enough to display all the digits of a given number. Sufficient widening of the column will eliminate the symbols. You can widen columns using the "Format, Column, Auto Fit Selection" menu command. Clicking on this menu-item automatically adjusts column widths to accommodate the data.
- *Personal DaqViewXL* does not have to be used as an *Excel* "Add-In;" it can also be loaded as a workbook. To do so, open **`pDaqView.xla`** using the *File Open* dialog box. The *Personal DaqViewXL* toolbar and menu items will be added to *Excel*.
- When loading the *Personal DaqViewXL* add-in manually, *Excel* does not automatically create a blank workbook. To create a workbook, access the *File, New* menu item (after *Excel* loads with *Personal DaqViewXL*).

  Note: *When you load Personal DaqViewXL* as a workbook, it will not automatically load the next time you run *Excel*. However, it will appear in the recently used list at the bottom of the *File* menu, making it easy to load the next time you run *Excel*.
- You can arm an acquisition from an *Excel* macro as shown in the following VBA code example:

```
Sub GoMacro()
      Dim cbpMenu As CommandBarPopup
      Dim cbButton As CommandBarButton

      'Get the Personal DaqViewXL menu object.
      Set cbpMenu = CommandBars("Tools").Controls("Personal DaqViewXL")
      'Get the "Go!" command button object.
      Set cbButton = cbpMenu.Controls(8)
      'Execute the button's command if the button is enabled.
      If cbButton.Enabled Then cbButton.Execute
End Sub
```

This is equivalent to clicking "Go!" on the *Personal DaqViewXL* menu or toolbar. Any of the other commands on the *Personal DaqViewXL* menu may be executed in a similar manner.

- If *Personal DaqView* cannot connect to your hardware on start-up, it will automatically switch to the Simulated Instrument mode when loaded from *Personal DaqViewXL*.


Notes


Notes

Signal Management 5

Overview ……5-1

Channel Control and Expansion ……5-3

Signal Acquisition ……5-4

Measurement Duration, Sample Rate, and Resolution ……5-4

Under Sampling and Aliasing …… 5-4

Triggering ……5-5

Input Isolation ……5-6

Signal Modes ……5-6

System Noise ……5-7

Averaging ……5-7

Analog Filtering ……5-8

Input and Source Impedance ……5-8

Crosstalk ……5-8

Overview

This chapter pertains to signal management, the use of different types of signals, and how to reduce common noise problems. The final portion of the chapter contains basic troubleshooting tips.

This table defines data acquisition terms, *as used in this manual*.

Data Acquisition Terms and Meanings	
Acquisition	A collection of scans acquired at a specified rate.
Aliasing	A type of error that results from having a scan rate set too low for a given variable input signal. Depending of the amount of aliasing, a sign wave may appear jagged, flat, or as a sign wave at a different frequency.
Analog	A signal of varying voltage or current that communicates data. Typical analog signals have the form of sine waves.
Analog-to-Digital Converter (ADC)	A circuit or device that converts analog values into digital values, such as binary bits, for use in digital computer processing.
API	Application Program Interface. The interface program within the Personal Daq system's driver that includes function calls specific to Personal Daq hardware and can be used with user-written programs (several languages supported).
Buffer	*Buffer* refers to a circuit or device that allows a signal to pass through it, while providing isolation, or another function, without altering the signal. *Buffer* usually refers one of the following: (a) A device or circuit, which allows for the temporary storage of data during data transfers. Such storage is often necessary to compensate for differences in data flow rates. A FIFO (First In - First Out) buffer is one in which the data that is stored first, is also the first data to leave the buffer. (b) A follower stage, which is used to drive a certain number of gates without overloading the proceeding, stage. (c) An amplifier which accepts high source impedance input and results in low source impedance output (effectively, an impedance buffer). (d) Buffer Amplifier (see *Buffer Amplifier*).
Buffer Amplifier	An amplifier used primarily to match two different impedance points, and isolate one stage from a succeeding stage in order to prevent an undesirable interaction between the two stages. (Also see *Buffer*).
Channel	In reference to Personal Daq, *channel* simply refers to a single *input*, or *output* entity. In a broader sense, an *input channel* is a signal path between the transducer at the point of measurement and the data acquisition system. A channel can go through various stages (buffers, multiplexers, or signal conditioning amplifiers and filters). Input channels are periodically sampled for readings. An *output channel* from a device can be digital or analog. Outputs can vary in a programmed way in response to an input channel signal.
Common mode voltage	Common mode voltage refers to a voltage magnitude (referenced to a common point) that is shared by 2 or more signals. *Example:* Signal 1 is +5VDC referenced to common. Signal 2 is +6VDC referenced to common. The common mode voltage for the two signals is ((5 + 6) ÷2), or +5.5 VDC Common mode pertains to signals that are identical in amplitude and duration; and can be used in reference to signal components.
Crosstalk	An undesired transfer of signals between systems or system components. Crosstalk causes signal interference, more commonly referred to as *noise*.
Digital	A digital signal is one of discrete value, in contrast to a varying signal. Digital data is represented by combinations of binary digits (0s and 1s).
Digital-to-Analog Converter (DAC)	A circuit or device that converts digital values (binary bits), into analog signals.

Data Acquisition Terms and Meanings	
Differential mode voltage	Differential mode voltage refers to a voltage difference between two signals referenced to a common point. *Example:* Signal 1 is +5VDC referenced to common. Signal 2 is +6VDC referenced to common. If the +5VDC is used as the reference, then the differential voltage is (6 - 5), or +1VDC. If the +6VDC is used as the reference, then the differential voltage is (5 - 6), or –1VDC.). The differential mode measures a voltage between 2 signal lines for a single channel. (Also see *single-ended mode*).
ESD	Electrostatic discharge (ESD) is the transfer of an electrostatic charge between bodies having different electrostatic potentials. This transfer occurs during direct contact of the bodies, or when induced by an electrostatic field. ESD energy can damage an integrated circuit (IC); so safe handling is required.
Excitation	Some transducers [e.g. strain gages, thermistors, and resistance temperature detectors (RTDs)] require a known voltage or current. Typically, the variation of this signal through the transducer corresponds to the condition measured.
Gain	The degree to which an input signal is amplified (or attenuated) to allow greater accuracy and resolution. Gain can be expressed as ×n or ±dB.
Isolation	The arrangement or operation of a circuit so that signals from another circuit or device do not affect the *isolated* circuit. In reference to Personal Daq, *isolation* usually refers to a separation of the direct link between the signal source and the analog-to-digital converter (ADC), as well as the 500V isolation form the PC. Isolation is necessary when measuring high common-mode voltage.
Linearization	Some transducers produce a voltage in linear proportion to the condition measured. Other transducers (e.g., thermocouples) have a nonlinear response. To convert nonlinear signals into accurate readings requires software to calibrate several points in the range used and then interpolate values between these points.
Multiplexer (MUX)	A device that collects signals from several input channels and outputs them on a single channel.
Sample (reading)	The value of a signal on a channel at an instant in time. When triggered, the ADC reads the channel and converts the sampled value into a designated bit value.
Scan	The channels that are selected for sampling.
Single-ended mode	The single-ended mode measures a voltage between a signal line and a common reference that may be shared with other channels. (Also see *differential mode*).
Trigger	An event to start a scan or mark an instant during an acquisition. The event can be defined in various ways; e.g., a TTL signal, a specified voltage level in a monitored channel, a button manually or mechanically engaged, a software command, etc. Some applications may use pre- and post-triggers to gather data around an instant or based on signal counts.
TTL	Transistor-Transistor Logic (TTL) is a circuit in which a multiple-emitter transistor has replaced the multiple diode cluster (of the diode-transistor logic circuit); typically used to communicate logic signals at 5 V.

Channel Control and Expansion

In a Personal Daq system, the quantities and types of Personal Daqs used, as well as the types and quantities of PDQ expansion cards used, determines the system's channel capacity. Up to 100 Personal Daqs can be used with one host PC.

Total channel capacity of a Personal Daq system can be calculated from the following tables:

Channel Capacities for Various Personal Daq Setups				
Personal Daq/54	**Volts Inputs**	**TC Inputs**	**Digital I/O**	**Freq/Pulse Inputs**
Personal Daq/54 ---- *No Expansion* ----	5 DE, or 10 SE	5 DE	---- N/A ----	---- N/A ----
Personal Daq/55 Systems	**Volts Inputs**	**TC Inputs**	**Digital I/O**	**Freq/Pulse Inputs**
Personal Daq/55	5 DE, or 10 SE	5 DE	8	2
Personal Daq/55 with PDQ1	15 DE, or 30 SE	15 DE	24	2
Personal Daq/55 with PDQ2	25 DE, or 50 SE	25 DE	8	2
Personal Daq/56 Systems	**Volts Inputs**	**TC Inputs**	**Digital I/O**	**Freq/Pulse Inputs**
Personal Daq/56	10 DE, or 20 SE	10 DE	16	4
Personal Daq/56 with PDQ1	20 DE, or 40 SE	20 DE	32	4
Personal Daq/56 with PDQ2	30 DE, or 60 SE	30 DE	16	4

DE = Differential Mode, SE = Single-Ended Mode

Signal Acquisition

Measurement Duration, Sample Rate, and Resolution

Measurement Duration (per channel) — the amount of time used for sampling a channel's input signal. For Personal Daq, the *measurement durations* range from very slow (610 milliseconds/sample) to very fast (12.5 milliseconds/sample).

Sample rate – Samples per second. The sample rate is the number of samples that take place per second. With a slow *measurement duration* of 610 milliseconds, there will only be 1.6 samples per second. With a very fast *measurement duration* of 12.5 milliseconds, there will be 80 samples per second.

Resolution (Bits RMS) – The number of reliable data bits that exist for a signal's measurement. The greater the resolution, the more *detailed* the reading, for example, with increased resolution a reading of 5.12 V could become 5.11896 V. Personal Daq actually provides for 24 bits of data information; however, the accuracy of the least significant bits becomes less as the measurement duration speeds up. At a *measurement duration* of 610 milliseconds, the last two bits are considered unreliable, resulting in a resolution of 22 bits. At a very fast *measurement duration* (12.5 milliseconds), the nine *most least significant bits* are unreliable, resulting in 15 bit accuracy.

Speed vs. Resolution[1]

Speed Designation	Measurement Duration (per channel)	Maximum Sample Rate[2] (Samples/sec)	Resolution[2] (Bits RMS) (-4 V to +4 V range)
Very Slow, 50 / 60 Hz rejection	610 ms	1.6 / sec	22
Slow, 50 Hz rejection	370 ms	2.7 / sec	22
Slow, 60 Hz rejection	310 ms	3.2 / sec	22
Medium, 50 Hz rejection	130 ms	7.7 / sec	21
Medium, 60 Hz rejection	110 ms	9.2 / sec	21
Medium	40 ms	25 / sec	19
Fast	20 ms	48 / sec	17
Very Fast	12.5ms	80 / sec	15

Notes:
1. Each channel can have independent measurement duration and resolution.
2. The sample rates shown were obtained with a 10-channel scan and with continuous self-calibration disabled.
3. Duration does not include the use of CJC measurements.

When you select the *measurement duration* you also determine the *sample rate* and *resolution* for the applicable channel. For Personal Daq's analog input applications, *sample rates* range from 1.6 samples/sec up to 80 samples/sec and corresponding resolution ranges from 22 to 15 bits.

Under Sampling and Aliasing

When you select the *measurement duration* you also determine the *sample rate* and *resolution* for the applicable channel. As shown in the previous table, *sample rates* range from 1.6 samples/sec up to 80 samples/sec and corresponding resolution ranges from 22 to 15 bit (for Personal Daq's analog input applications).

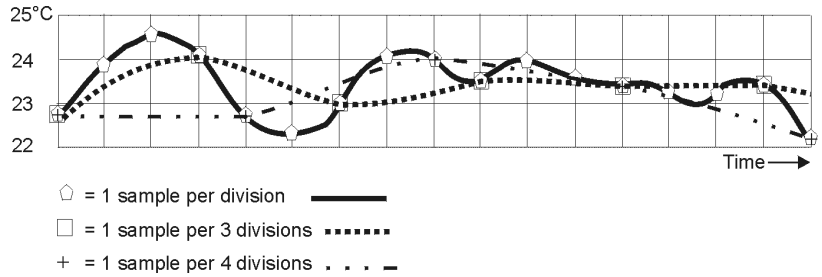
When measuring highly variable input signals (as opposed to relatively steady input signals), the variable signals will require more samples/sec to obtain a realistic signal representation. When too few samples are taken, the term *under sampling* is often used. *Under sampling* tends to lower the signal's graphed amplitude. This is also known as *aliasing*. In such cases, the inadequate number of samples results in a "flattening" of the signal. Gross under sampling can even result in a relatively flat line, even though the actual signal is a waveform.

For best results the sample rate (scan rate) should be at least 10 times the measurement frequency. If possible it is recommended that you set the scan rate to 20 times the frequency of the input signal.

Note: Frequency channels will read frequency regardless of the scan rate. Personal Daq circuitry reads a *pulse count* and *timer count* during each scan.

The following table provides general advice regarding the selection of *measurement duration*. The concepts are further illustrated by the figure, *Examples of Under Sampling*.

Analog Input Signal Variability	Measurement Duration	Sample Rate	Resolution
Steady, or gradual change	Long	low	high
Highly variable (unsteady)	Short	high	low



Examples of Under Sampling

The above figure depicts three signals from the same temperature fluctuations. *Under Sampling* (*aliasing*) is evident in two of the signals. Elaboration follows:

Signal 1 – This signal, based on 1 sample per division, is represented by a heavy solid line and sample-points designated by polygon symbols. The signal represented is a fairly accurate portrayal of the actual temperature fluctuations.

Signal 2 – This signal, based on 1 sample every 3 divisions, is represented by a heavy dotted line and sample-points designated by squares. *Under sampling* has resulted in a distortion, in effect, a lower amplitude than that exhibited by the first signal, even though each measured point is accurate.

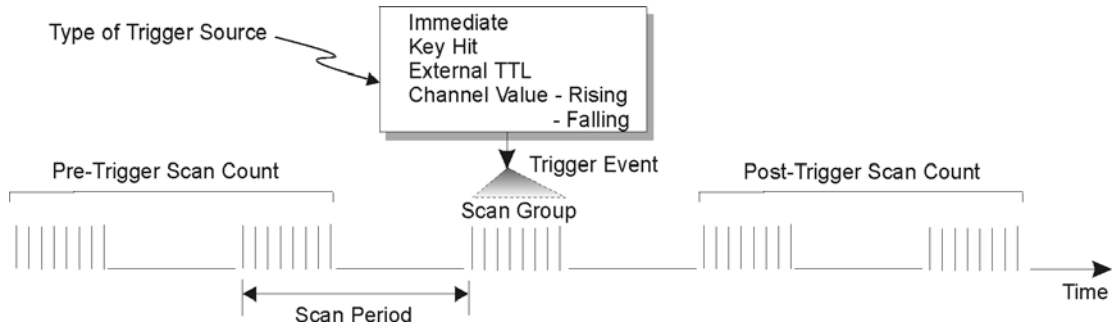
Signal 3 – This signal, based on 1 sample every 4 divisions, is represented by a dotted/dashed line and sample-points designated by plus signs (+). Fewer samples (a greater degree of *under sampling*) has resulted in a further distortion and lowering of signal amplitude.

From these examples it should be realized that more samples will result in a more accurate representation of the actual signal, and that under sampling will tend to lower the amplitude of the signal, exhibiting a trend toward a “flat line” state.

As stated earlier, the more variability a signal has, the more samples that are needed to accurately portray it.

Triggering

Triggering controls an acquisition cycle. Once the system is armed, a trigger is required to collect the data. Typically, three data collection parameters are specified: the pre-trigger count, the post-trigger scan count, and the trigger source. The user must determine the triggering requirement based on the nature of the measurement and the amount of data needed to satisfy the system’s purpose.



- The **pre-trigger scan count** specifies the number of scans that are to be collected before the trigger point. If the pre-trigger scan count is greater than zero, the system will continuously collect data until the trigger is satisfied. If no pre-trigger scans are required, the system sits idle until the trigger; then, it collects the post-trigger scans before it disarms.
- The **post-trigger scan count** specifies the number of scans to be collected after the trigger point. After the trigger, the post-trigger scans will be collected as programmed and then the system will disarm itself.
- The **trigger source** can be a software command, an external TTL input, etc. An analog input channel on reaching a specified voltage level can be used to trigger the system.

Input Isolation

Three benefits of input isolation are circuit protection, noise reduction, and the rejection of high common mode voltage.

- **Circuit protection**. Input isolation separates the signal source from circuits that may be damaged by the signal. (Voltages higher than about 10 V can distort data or damage chips used in data acquisition.) High-voltage signals or signals with high-voltage spikes should therefore be isolated. The protection can also work the other way—to safeguard a sensitive signal conditioner from a failing device elsewhere in the system.
- **Noise reduction**. Isolation eliminates ground loops for high-gain systems and multi-unit systems that are grounded together. The chassis for each device can rest at a ground potential slightly different from the other devices. These irrelevant currents and the spikes they may have picked up by induction can thus be kept out of the measurement circuit.
- **Rejection of high common mode voltage**. There is a limit to the amount of voltage a differential amplifier can have applied between ground and the amplifier inputs. Fortunately, the differential amplifier rejects high common mode voltage signals. High common mode voltage and noise spikes are rejected (canceled out) in *in-phase* signals identical in amplitude and frequency that are present in both the high and low inputs at the same time.

Signal Modes

Personal Daq units operate in one of two modes, (1) single-ended mode, or (2) differential mode. These terms (*single-ended mode* and *differential mode*) apply to their use in this manual. In other sources these terms may be used in a different manner.

Choosing between differential and single-ended inputs is made by software command. The following text briefly describes the two signal modes.

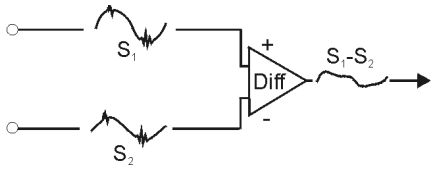
Single-ended mode refers to a mode, or circuit set-up, in which a voltage is measured between 1 signal line and common ground voltage (Vcm). The measured voltage may be shared with other channels. The advantage of a single-ended *non-differential* mode [over differential mode] is that it provides for a higher channel count, for example: 20 channels instead of 10.



In Personal Daq applications, thermocouples should <u>not</u> be connected *single-ended*. Doing so can result in noise and false readings. This is especially true when acquiring other high-amplitude signals in conjunction with thermocouple signals that are connected *single-ended*.

Differential mode refers to a mode, or circuit set-up, in which a voltage is measured between 2 signal lines. The measured differential voltage is used for a single channel. An advantage of using differential inputs is that they reduce signal errors and the induction of noise resulting from ground current. The following illustration is an example of how noise is reduced, or canceled-out, when using the differential mode.

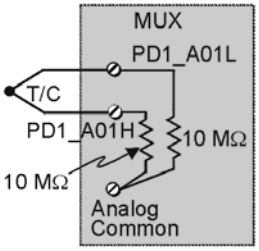
In the schematic, voltage signal S_2 is subtracted from signal S_1, resulting in the output signal shown. The noise spikes (having *the same polarity, phase, and magnitude in each input signal)* cancel each other out. This results in a clean differential signal (S_1 - S_2).



Noise Reduction in Differential Mode

In the schematic, signals S_1 and S_2 are shown in-phase; however, even if these signals were out of phase, the noise in each (indicated by jagged lines) would still have the same magnitude, phase, and polarity. For that reason, they would still cancel out.

Floating-differential measurements are made when low-level signals must be measured in the presence of high levels of common-mode noise (e.g., a non-grounded thermocouple). When the signal source has no direct connection to the system analog common, one must be provided. In Personal Daq the connection to analog common is provided in the circuitry with both the *channel high* and *channel low* connected to analog common. Both of these connections to common are made through 10 MΩ resistors. No additional connections of channel high and low to common should be made.



Example of Floating Differential Circuit



Differential signal hookups do not provide isolation or any kind of circuit protection.

Resolution: An analog-to-digital converter (ADC) converts an analog voltage to a digital number. The digital number represents the input voltage in discrete steps with finite resolution. ADC resolution is determined by the number of bits that represent the digital number. An *n*-bit ADC has a resolution of 1 part in 2^n. Thus, 12 and 16 bit resolutions are as follows:

- 12-bit resolution: 1 part in 4096 (2^{12}), corresponding to 2.44 mV in a 10 V range.
- 16-bit resolution: 1 part in 65,536 (2^{16}), corresponding to 0.153 mV in a 10 V range.

System Noise

Laboratory and industrial environments often have multiple sources of electrical noise. An AC power line is a source of 50/60 Hz noise. Heavy equipment (air conditioners, elevators, pumps, etc.) can be a source of noise, particularly when turned on and off. Local radio stations are a source of high-frequency noise, and computers and other electronic equipment can create noise in a multitude of frequency ranges. Thus, an absolute noise-free environment for data acquisition is not realistic. Fortunately, noise-reduction techniques such as averaging, filtering, differential voltage measurement, and shielding are available to reduce noise to an acceptable level.

Averaging

Certain acquisition programs apply *averaging* after several samples have been collected. Depending on the nature of the noise, averaging can reduce noise by the square root of the number of averaged samples. Although averaging can be effective, it suffers from several drawbacks. Noise in measurements only decreases as the square root of the number of measurements—reducing RMS noise significantly may require many samples. Thus, averaging is suited to low-speed applications that can provide many samples.

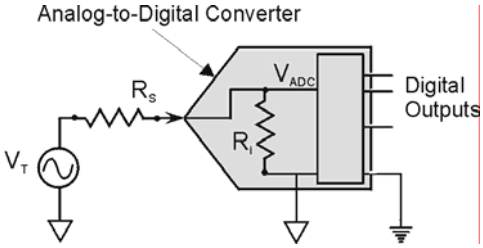
Note: Only random noise is reduced or eliminated by averaging. Averaging does not reduce or eliminate periodic signals.

Analog Filtering

A filter is an analog circuit element that attenuates an incoming signal according to its frequency. A low-pass filter attenuates frequencies above the cutoff frequency. Conversely, a high-pass filter attenuates frequencies below the cutoff. As frequency increases beyond the cutoff point, the attenuation of a single-pole, low-pass filter increases slowly. Multi-pole filters provide greater attenuation beyond the cutoff frequency but may introduce phase (time delay) problems that could affect some applications.

Input and Source Impedance

As illustrated in the figure to the right, the input impedance (R_i) of an analog-to-digital converter combines with the transducer's source impedance (R_s) forming a voltage divider. This divider distorts the voltage being read at the analog-to-digital converter. The actual voltage read is represented by the equation:



$$\mathbf{V_{ADC} = V_T \times R_i / (R_s + R_i)}$$

The input impedance (R_i) of most ADCs is at least 1 MΩ; low source impedance (R_s) usually presents no problem. Some transducers, such as piezoelectric types, have high source impedance, and should therefore be used with a charge-sensitive amplifier of low output impedance. As described in the following paragraphs, multiplexing can greatly reduce the effective input impedance of an analog-to-digital converter.

Crosstalk

Crosstalk is a type of noise related to source impedance and capacitance, in which signals from one channel leak into an adjacent channel, resulting in interference or signal distortion. The impact of source impedance and stray capacitance can be estimated by using the following equation.

$$\mathbf{T = RC}$$

Where **T** is the time constant, **R** is the source impedance, and **C** is the stray capacitance.

High source (transducer) impedance can be a problem in multiplexed A/D systems. When using more than 1 channel, the channel input signals are multiplexed into the A/D. The multiplexer samples each signal and then switches to the next input signal. A high-impedance input interacts with the multiplexer's stray capacitance and causes crosstalk and inaccuracies in the A/D sample.

A solution to high source impedance in relation to multiplexers involves the use of buffers. The term buffer has several meanings; but in this case, *buffer* refers to an operational amplifier having high input impedance but very low output impedance. Placing such a buffer on each channel (between the transducer and the multiplexer) prevents the multiplexer's stray capacitance from combining with the high input impedance. This use of a buffer also stops transient signals from propagating backwards from the multiplexer to the transducer.

Calibration 6

Introduction ……6-1
Required Equipment ……6-2
Calibration Procedure ……6-2

Introduction

Although Personal Daq units are calibrated prior to shipment, they still require periodic calibration to ensure that accuracy is maintained. The industry standard for this calibration is once per year.

UserCal provides prompts to assist you through Personal Daq's calibration procedure. The Windows-based program was developed with the intent of making the calibration task easy to perform.

Note: Calibration constants are calculated and stored in each Personal Daq unit's serial EEPROM. This permits adding expansion modules to single Personal Daq units. It also allows you to swap expansion modules from one Personal Daq unit to another.

CAUTION
 **Use approved ESD precautions, including static-free work area and grounded wrist strap, when handling electronic components. Failure to do so could cause equipment damage due to electrostatic discharge.**



Calibration is to be performed by authorized personnel in a controlled, still air environment at 23±2°C. Failure to comply with this requirement can result in faulty equipment performance and necessitate additional services of an authorized metrology lab.



The VDC Calibrator used must meet the following criteria:

1) **Range: 0 - 10 V**
2) **Resolution: 10 µV**
3) **Peak-to-peak noise: 600 µV**

The digital voltmeter (or digital multimeter) used to verify calibration voltage accuracy must meet the following criteria:

1) **Minimum Resolution: 6-1/2 digits**
2) **Minimum DC Accuracy: 0.005% full scale**

Failure to comply with these requirements can result in faulty equipment performance and necessitate additional services of an authorized metrology lab.



Never connect an expansion module to (or remove it from) a Personal Daq main unit while the main unit is connected to a power source. Such action may result in EEPROM errors and loss of calibration data.



Allow at least 1 hour warm-up time for the VDC Calibrator, Digital Multimeter, and each Personal Daq unit (including expansion module) that is to be calibrated. If a cold cell device is used, allow the cell to warm up in accordance with device operator's manual.

Required Equipment

You will need the following items to perform Personal Daq calibration. Ensure the equipment meets the specifications listed in the cautions on the preceding page.

Equipment for Voltage Portion of Calibration
VDC Calibrator
Digital Voltmeter
Copper Short
2-Pin Connector Harness (made from DB25 Male Connector, see figure on page 6-5)
Note that the harness is not required; however, its use is recommended to avoid damaging the Personal Daq expansion connector and to ensure good pin contact.

Equipment for Thermocouple Portion of Calibration*
0°C Temperature Reference (Cold cell, or ice bath)
T-type Thermocouple
T-type T/C wire
*Thermocouple calibration not required if Personal Daq system is not used for temperature measurements.

Calibration Procedure



Computer *energy save* (suspension modes) can result in false calibration constants and erroneous data. Prior to the one-hour calibration warm up time, ensure your computer's *energy save* mode is disabled. If needed, consult your PC user's manual to disable *energy save*.

Observe the above caution and those listed on page 6-1.

Note: *UserCal* provides screen prompts to guide you through the calibration procedure.

1. Disconnect the USB cable from the Personal Daq main unit.
2. If a power adapter is used in your application, disconnect the adapter cable from the unit.
3. Remove all signal lines from the main unit terminal blocks.
4. If an expansion module is used, remove all signal lines from the expansion module terminal blocks.
5. If an expansion module is used in your application, but is not connected to the main unit, connect the module to the main unit and secure with retaining clips.
6. *If you do not use your Personal Daq system for temperature measurements*, proceed directly to step 7.

   If you use your Personal Daq system for any temperature measurements, complete steps a) through d), then proceed with step 7.

   a) Place thermocouple in an ice bath or cold cell device at 0°C.

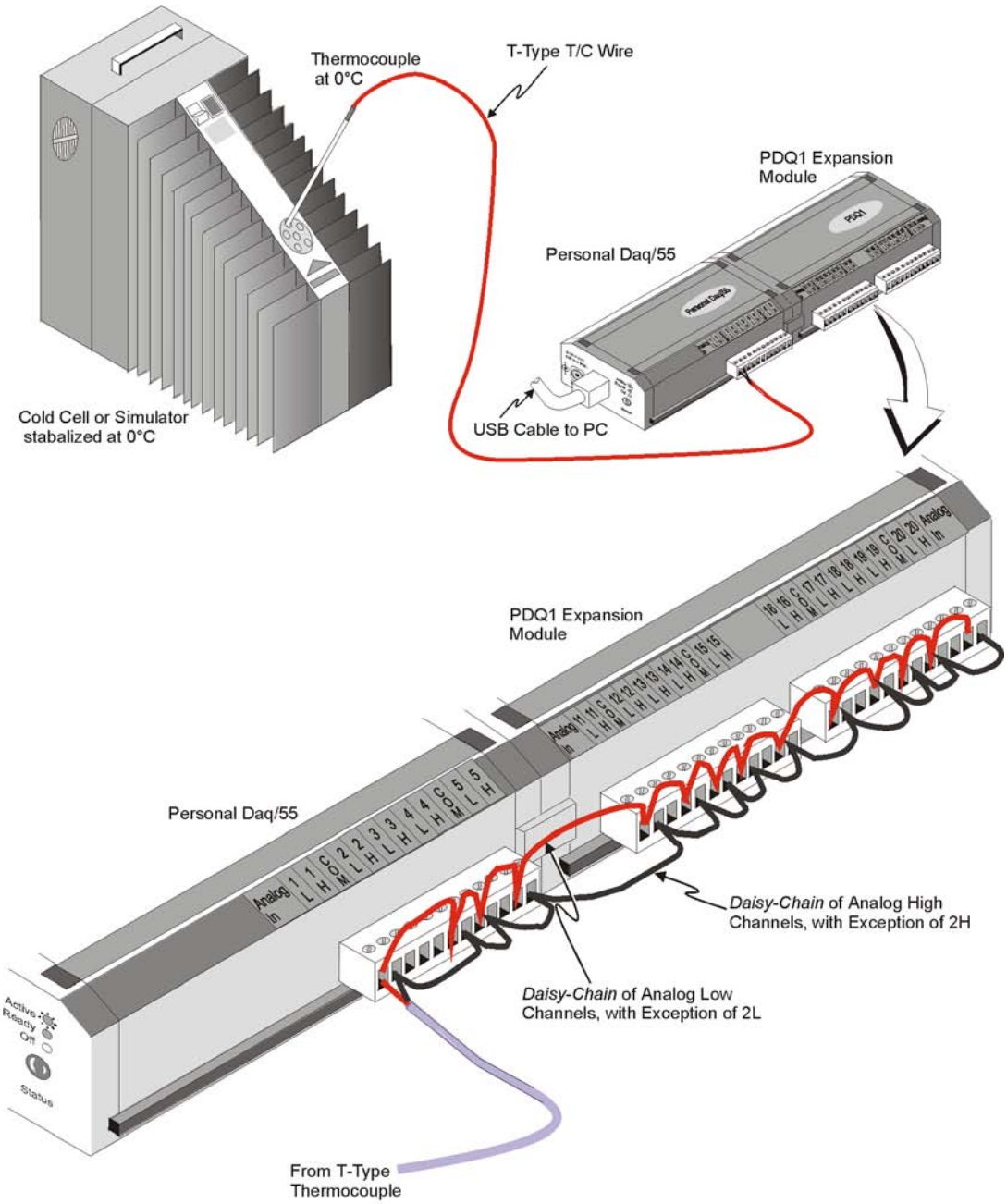
   b) Connect *red* (-) thermocouple wire to channel 1L.

   c) Connect *blue* (+) thermocouple wire to 1H.

   d) *Daisy-chain* stripped T-type T/C wire to all analog channels as follows:

   - *red* (-) wire to each analog Lo channel, **except channel 2L**.
   - *blue* (+) wire to each analog Hi channel, **except channel 2H**.

The following two figures illustrate these connections for a Personal Daq/55 with a PDQ1 expansion module.

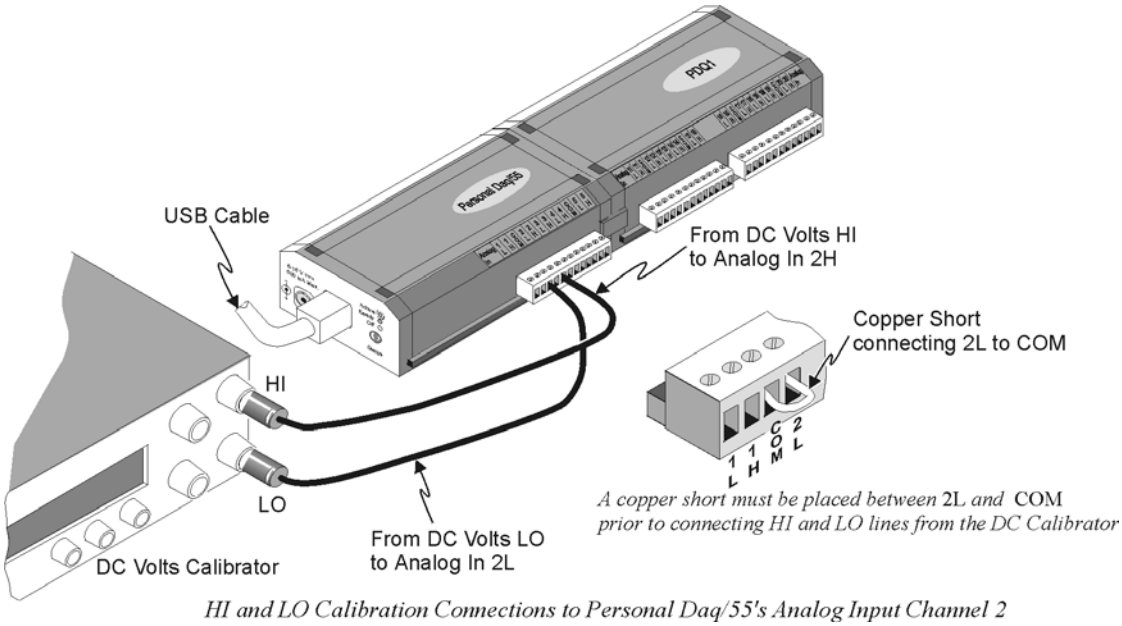


CJC Calibration. Thermocouple is connected with red (-) wire to 1L and blue (+) wire to 1H. Analog Highs and Lows are daisy-chained as indicated. Note that 2L and 2H are left open for voltage-related connections as discussed in step 7.

Example of Connections for CJC Calibration

It is important to perform calibration in a controlled, still air environment at 23±2°C.

7. Make the following 3 voltage-related connections (see following figure).
   - Copper short: From Personal Daq Analog In 2L (2 low) to Personal Daq common (COM).
   - HI Lead: From Volts DC Calibrator HI to Personal Daq Analog In 2H (2 high)
   - LO Lead: From Volts DC Calibrator LO to Personal Daq Analog In 2L (2 low)



HI and LO Calibration Connections to Personal Daq/55's Analog Input Channel 2

Note: Thermocouple wiring not shown for clarity.

8. If used in your application, connect the power adapter to the Personal Daq.
9. Connect the USB cable to the Personal Daq.
10. Allow the entire setup to warm up for at least one hour. If using a cold cell device, allow the cell to warm up in accordance with the device operator's manual.
11. Start the *UserCal* program from a desktop shortcut, or from the Windows Start Menu.
12. If you do not use your Personal Daq system for temperature measurements, deselect the Thermocouple Calibration option shown in the *UserCal* dialog box.
13. Using the keypad of your PC or laptop, enter a reference value between +3.900 and +4.100V. Note that a value of +4.096V is recommended. The value entered will be used in the following step.
14. Using the Volts DC Calibrator, apply the reference calibration voltage (from the previous step) to Analog Input channel 2.

    If the reference voltage is not found: The program aborts the process and prompts you to double check the reference voltage for Analog Input 2H with respect to Analog Input 2L.

    If the reference voltage is found: Personal Daq's LED flashes and *UserCal* prompts you to select "Next." This step ensures that the Personal Daq is properly configured and can acquire data.

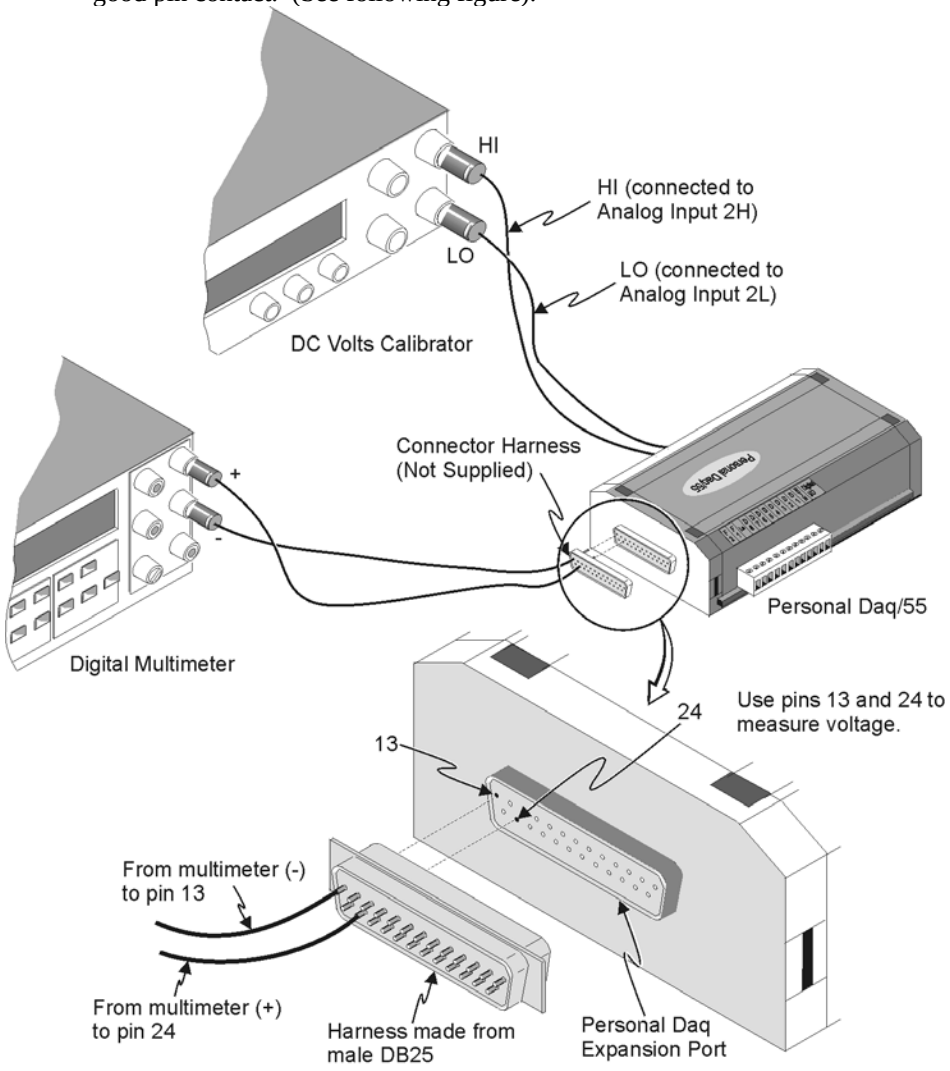
15. Apply **0.000V** to Analog Input channel 2. Select "Next."
16. Apply the **positive** reference voltage (**from step 13**) to Analog Input channel 2. Select "Next."
17. Apply the **negative** reference voltage (**from step 13**) to Analog Input channel 2. Select "Next."

Note: If *UserCal* sees any of the three voltages (zero reference, positive reference, or negative reference) as incorrect, the program informs the user and aborts.

18. If Thermocouple calibration was selected, calibrate each CJC channel as prompted by *UserCal*.

19. If your Personal Daq is connected to an expansion module:
    - remove the USB cable from the Personal Daq main unit
    - if used in your application, remove the external power cable from the Personal Daq main unit
    - carefully remove the retaining clips from the Personal Daq modules
    - remove the expansion module from the main unit
    - if a power adapter was used, reconnect the adapter cable to the Personal Daq main unit
    - reconnect the USB cable to Personal Daq main unit

Note: For step 20, it is recommended that you make a simple harness from a DB25 male connector to avoid damaging Personal Daq's expansion connector. A properly made harness will also ensure good pin contact. (See following figure).



Measuring Voltage at Personal Daq's Expansion Port

20. Connect the multimeter positive lead to pin number 24 and the negative lead to pin number 13 (note the harness in the preceding figure). Measure the reference voltage at Personal Daq's expansion port, and note the value for use in the following step.

21. Using the keypad of your PC or laptop, enter the reference value obtained from step 20. Select "Next." *UserCal* now calculates calibration constants and stores them in Personal Daq's serial EEPROM. *UserCal* displays a brief calibration report.

22. Disconnect the calibration equipment from the Personal Daq
    - VDC Calibrator
    - Harness and multimeter
    - copper short (remove from Analog In 2L and Personal Daq common low (COM)

The calibration procedure is complete. You may now return your Personal Daq system to its data acquisition status.

Troubleshooting & Customer Assistance 7

Radio Frequency Interference …… 7-1
Electrostatic Discharge (ESD) …… 7-1
Troubleshooting …… 7-2
Customer Assistance …… 7-5

Reference Note:
API Error Codes are defined at the end of Appendix B.

Radio Frequency Interference

Personal Daq hardware complies with the limits for a Class B digital device according to FCC rules and CE specifications. These limits provide reasonable protection against harmful interference in a residential environment.

If not installed or used correctly, this equipment can radiate radio frequency energy and interfere with radio communications. You can determine if the interference is caused by the Personal Daq system by disconnecting and then reconnecting the Personal Daq while observing the effect on the interference.

You can often correct radio interference by one or both of the following measures:

- **Antenna Adjustment:** Reorient the receiving antenna.
- **Spatial Separation:** Increase the separation between the Personal Daq equipment and the receiver of the device experiencing interference.

If the problem cannot be resolved, consult an experienced radio/television technician for help.

Electrostatic Discharge (ESD)

CAUTION



The discharge of static electricity can damage some electronic components. Semiconductor devices are especially susceptible to ESD damage. You should always handle components carefully, and you should never touch connector pins or circuit components unless you are following ESD guidelines in an appropriate ESD controlled area. Such guidelines include the use of properly grounded mats and wrist straps, ESD bags and cartons, and related procedures.

Troubleshooting

Certain problems can be solved without factory assistance. Before calling your service representative you should go through the following checklist, as well as the symptoms and solutions. When applicable, be sure to follow ESD prevention guidelines to avoid damaging components.

Basic Checklist

- **Power**. Check USB cable and connections. Check hub connections and power adapters, when applicable. If using a laptop computer, ensure the laptop's battery is sufficiently charged.
- **Signal**. Check signal lines and connections. Connectors must be free of corrosion. Signal lines should be undamaged and free of sharp bends and twists. Signal paths should avoid potential sources of noise (high voltage and electromagnetic interference).
- **Software**. Try to acquire data with Personal DaqView. If you are unable to acquire data in both Personal DaqView and another program, a hardware problem is likely.
- **Setup Parameters**. Make sure the device selected in software matches the hardware being used. Verify that setup parameters are correct for your application.

Symptoms and Solutions

1. Personal DaqView will not install properly.

Check your computer's Window's Operating System. Systems preceding Windows 2000 or Windows XP are not to be used with Personal DaqView.

If your computer is already using Windows 2000 or XP, make sure that you have installed the latest version of Personal DaqView. To check on the version you are running, open Personal DaqView's Help pull-down menu and select "About Personal DaqView..."

If you need to update your version of Personal DaqView, first uninstall your existing version. This can be done using the *Add/Remove Programs* utility in the Windows Control Panel. Install the new version of Personal DaqView after the older version has been removed. The executable file for Personal DaqView can be obtained from a recent data acquisition CD or from our web site.

2. Personal DaqView seems to run fine, but the data does not make sense.

Check the blue title bar at the top of the Personal DaqView screen. If you see the word "Simulated" in parentheses, it means that the Personal Daq is running in a simulated mode and is generating artificial data.

To leave the simulated mode:

(a) Select *Active Devices* from the *View* pull-down menu.

(b) Uncheck the Simulated Device.

(c) Check the real Personal Daq device. Note that the real devices, which are listed as active devices, include a serial number as part of their identification.

If the data is awkward when coming from a real device, have the device calibrated according to the instructions in chapter 6 of the user's manual. A PDF version of the document is included on the data acquisition CD. If the calibration fails, see page 7-5 in regard to customer assistance.

3. There is a "blue screen" or an error message when connecting the Personal Daq to the PC.

Many notebook computers fail to meet the 500 mA current that is specified by the USB standard and, in fact, will not even supply the 400 mA that is required by the Personal Daq. If this is the case you will need to use an auxiliary power supply as follows:

(a) Unplug the USB cable from the Personal Daq.

(b) Obtain a power adapter that provides an output voltage of 6 to 16 VDC and a maximum current output of 500 mA.

(c) Connect the adapter's power in line to a suitable power source.

(d) Connect the adapter's power-out line to the Personal Daq's External Power connector.

(e) Reconnect the USB cable to the Personal Daq.

4. Personal DaqView was not installed prior to the hardware.

Personal DaqView should have been installed before connecting and installing the hardware. If you installed the hardware first you will need to complete the following steps.

(a) Access the *Windows Device Manager*. This can be done from the *Start* menu by navigating as follows:
Start ➡ Settings ➡ Control Panel ➡ System ➡ Hardware ➡ Device Manager*
*Depending on the operating system, access to the *Device Manager* might be obtained directly from *System*.

(b) Scroll down the list and look for "pDaq" or "Personal Daq." If neither is found, look for "Unknown Device." The latter will have a yellow question mark next to it.

(c) Select the device (found in step "b"), and remove it by clicking on the <Remove> button.

(d) Unplug the USB cable from the Personal Daq.

(e) If you have not already done so, install *Personal DaqView* on the computer.

(f) Reconnect the USB cable to the Personal Daq. This should start the driver install process and the Personal Daq should now show up in the *Device Manager* list.

5. Personal DaqView appears to run fine with one Personal Daq, but not with two or more.

You need to have the *Plus* version of *Personal DaqView* if you want to use more than one Personal Daq device. To see whether or not you have the *Plus* version, check the blue title bar at the top of the *Personal DaqView* screen. If it does not indicate "Personal DaqView Plus" you will need to order it from the factory. Otherwise it will not be possible to run *Personal DaqView* with multiple devices.

6. When running Personal DaqView under Windows 2000 or Windows XP the computer "blue screens" when the Personal Daq is connected and the driver is loaded.

This problem should not be seen if you are using a current driver. If you are experiencing a "blue screen" when the driver is being loaded, verify that you are installing the latest driver.

7. Personal DaqView runs fine with analog inputs, but “blue screens” with frequency inputs.

When sampling frequency channels, be sure to have at least one analog input channel active. This will prevent the “blue screen” state.

8. The Personal DaqView window stays minimized.

If this happens the *Personal DaqView* window is actually "restored" but its position is off the screen. You can resolve this problem as follows:

(a) Run *Personal DaqView*.

(b) In the task bar, right-click on the *Personal DaqView* icon. Note that “Restore” is probably grayed out.

(c) Select "Move" with a left-click and without moving the mouse; take your hand off the mouse.

(d) Press any one of the 4 arrow keys on your keyboard; then move the mouse until you see the window (or possibly the outline of the window) positioned where you want it.

(e) Left-click. The window should be back to normal.

(f) Resize and relocate the window as desired.

(g) Select "Save" from the File menu to save your configuration including the new window location and size.

Note: If you need to open up the *Channel Configuration* window, you can do so by going to the *View* pull-down menu and selecting “Channel Configuration.”

Customer Assistance



Reference Notes:
Before calling for assistance . . .

- Refer to the portions of this manual relevant to your situation, especially the troubleshooting material.
- Refer to Appendix B, *API Commands*, if you are creating your own programs. The appendix lists command functions in alpha-numerical order and includes error code definitions.

To report problems and receive support, call your service representative. When you call, please have the following information available:

- Hardware model numbers
- Contents of your CONFIG.SYS, AUTOEXEC.BAT, and SYSTEM.INI files
- Software version numbers for Personal DaqView, DOS and Windows
- Type of computer and features

All equipment returned to the factory must be accompanied by an RMA# (Return Merchandise Authorization number). Use original shipping containers or equivalent to prevent shipping damage. In addition to inclusion of the above information, please include:

- The name and phone number of an individual who can discuss the problems encountered
- Your shipping instructions
- A copy of troubleshooting notes and comments on tests performed and all problem-related conditions.


Notes

Appendices

***Appendix A* — API Custom Program Models**

***Appendix B* — API Commands**

Appendix C* — *Removed

***Appendix D* — Custom Labels**

Note: The information in appendices A and B pertains to the Applications Programming Interface (API). This information is not necessary for users of *Personal DaqView* who plan to do no programming.

Synopsis of Appendices

Appendix A: **API Custom Program Models** provides information for creating custom software to satisfy your specific data acquisition requirements. The appendix explains how to combine API functions to perform typical tasks.

***Appendix B*: API Commands** describes the entire command set for the Personal Daq. Syntax, parameters, interpretation, and error codes are explained. Sections on the individual commands include their parameters, types, typical use, and related information.

***Appendix C*:** *Removed*

***Appendix D*: Custom Labels** provides blank labels and a Personal Daq channel layout reference. The appendix also pertains to **`pDaq_CustomLabels.doc`**. The file is located in the target directory: **`\\Program Files\pDaqView`**. The file document provides blank labels in table format that you can write in, edit, and print from your PC.

API Custom Program Models A

Overview ……A-1

Data Acquisition Environment ……A-1

Application Programming Interface (API) ……A-1
Hardware Capabilities and Constraints ……A-2
Signal Environment ……A-2

Programming Models ……A-2

Initialization and Error Handling ……A-3
Foreground Acquisition with One-Step Commands ……A-4
Counted Acquisitions Using Linear Buffers ……A-5
Indefinite Acquisition, Direct-To-Disk Using Circular Buffers ……A-7
Multiple Channel Types……A-10

Summary Guide of Selected API Functions ……A-14

Overview

By using the Application Programming Interface (API) with Personal Daq systems, you can create custom software to satisfy your data acquisition requirements. Appendix B explains the API functions in detail. This appendix shows how to combine API functions to perform typical tasks. When you understand how the API works with the hardware, you are ready to program for optimum data acquisition. To help you get this perspective, this appendix is divided into 3 parts:

- **Data Acquisition Environment** outlines related concepts and defines Personal Daq capabilities the programmer must work with (the API, hardware features, and signal management).
- **Programming Models** explains the sequence and type of operations necessary for data acquisition. These models provide the software building blocks to develop more complex and specialized programs. The description for each model has a flowchart and program excerpt to show how the API functions work.
- **Summary Guide of Selected API Functions** is an easy-to-read table that describes when to use the basic API functions.

Data Acquisition Environment

In order to write effective data acquisition software, programmers must understand:

- Software tools (the API documented in this manual and the programming language—you may need to consult documentation for your chosen language)
- Hardware capabilities and constraints
- General concepts of data acquisition and signal management

Application Programming Interface (API)

The API includes all the software functions needed for building a data acquisition system with the hardware described in this manual. Appendix B (*API Commands)* supplies the details about how each function is used (parameters, hardware applicability, etc). In addition, you may need to consult your language and computer documentation.

Hardware Capabilities and Constraints

To program the system effectively, you must understand your Personal Daq hardware capabilities. Obviously you cannot program the hardware to perform beyond its design and specifications, but you also want to take full advantage of the system's power and features. You may need to refer to manual chapters that pertain to hardware capability. In addition, you may need to consult your computer documentation. In some cases, you may need to verify the hardware setup, use of channels, and signal conditioning options.

Signal Environment

Important data acquisition concepts for programmers are listed below. You should also refer to Appendix C, *Signal Management and Troubleshooting,* to gain a better understanding of signal-related issues. These include:

- **Channel Identification**
- **Scan Rates and Sequencing** With multiple scans, the time between scans becomes a parameter. This time can be a constant or can be dependent upon a trigger.
- **Counter/Timer Operation**
- **Triggering Options** Triggering starts the A/D conversion. The trigger can be an external analog or TTL trigger, or a program controlled software trigger.
- **Foreground/Background** ***Foreground routines*** (**…Rd** routines) require the entire transfer to occur before returning control to the application program. ***Background routines*** (**… Transfer** routines) start the A/D acquisition and return control to the application program before the transfer occurs. Data is transferred while the application program is running. Data will be transferred to the user memory buffer during program execution in 1 or more sample blocks, depending on the configuration. The programmer must determine what tasks can proceed in the background while other tasks perform in the foreground and how often the status of the background operations should be checked.

Parameters in the various A/D routines include: number of channels, number of scans, start of conversion triggering, timing between scans, and mode of data transfer. Channels can be sampled in consecutive or non-consecutive order with the same [or different] gains. The scan sequence makes no distinction between local and expansion channels.

Programming Models

Note: Two types of VB function wrappers exist for the API functions that reference data buffers. One type references *an integer buffer*, and the other references a *single-precision floating point buffer*. The wrappers that reference single-precision floating point buffers have the word **`Single`** appended to the function name. For example, **`VBdaqAdcTransferSetBufferSingle`** would be used to set a single-precision floating point buffer, whereas **`VBdaqAdcTransferSetBuffer`** be used to set an integer buffer.

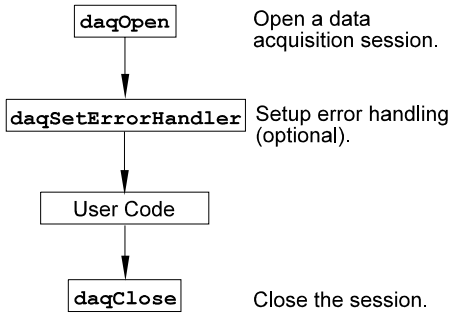
This section outlines basic programming steps commonly used for data acquisition. Consider the models as building blocks that can be put together in different ways or modified as needed. As a general tutorial, these examples use Visual Basic since most programmers know BASIC and can translate to other languages as needed. The following table identifies the API programming models discussed in this appendix.

Model Type	Model Name	Page
Configuration	Initialization and Error Handling	A-2
Acquisition	Foreground Acquisition with One-Step Commands	A-4
	Counted Acquisition Using Linear Buffers	A-5
	Indefinite Acquisition, Direct-To-Disk Using Circular Buffers	A-7
	Multiple Channel Types	A-10

Initialization and Error Handling

This section demonstrates how to initialize the Personal Daq and use various methods of error handling.
Most of the example programs use similar coding as detailed here. Functions used include:

- **`VBdaqOpen&(daqName$)`**
- **`VBdaqSetErrorHandler&(errHandler&)`**
- **`VBdaqClose&(handle&)`**



All Visual Basic programs should include the **PDAQX.BAS** file into their project. The **PDAQX.BAS** file provides the necessary definitions and function prototyping for the **DaqX** driver **DLL**.

```
handle& = VBdaqOpen&("PDAQ123456")
ret& = VBdaqClose&(handle&)
```

The PersonalDaq device is opened and initialized with the **daqOpen** function. **daqOpen** takes one parameter—the name of the device to be opened. The device name is the *device serial#* pre-prended with "**PDAQ**".

For instance if the serial# for the Personal Daq is 123456 then the device name to use to open that device would be "**PDAQ123456**". The **daqOpen** call, if successful, will return a *handle* to the opened device. This handle may then be used by other functions to configure or perform other operations on the device. When operations with the device are complete, the device may then be closed using the **daqClose** function. If the device could not be found or opened, **daqOpen** will return **-1**.

The DaqX library has a default error handler defined upon loading. However; if it is desirable to change the error handler or to disable error handling, then the **daqSetErrorHandler** function may be used to setup an error handler for the driver. In the following example the error handler is set to **0** (no handler defined) which disables error handling.

```
ret& = VBdaqSetErrorHandler&(0&)
```

If there is a Personal Daq error, the program will continue. The function's return value (an error number or **0** if no error) can help you debug a program.

```
If (VBdaqOpen&("PDAQ123456") <  0) Then
  "Cannot open PDaq"
```

Personal Daq functions return **daqErrno&**.

```
 Print "daqErrno& : "; HEX$(daqErrno&)
End If
```

The next statement defines an error handling routine that frees us from checking the return value of every Personal Daq function call. Although not necessary, this sample program transfers program control to a user-defined routine when an error is detected. Without a Personal Daq error handler, Visual Basic will receive and handle the error, post it on the screen and terminate the program. Visual Basic provides an integer variable (ERR) that contains the most recent error code. This variable can be used to detect the error source and take the appropriate action. The function **daqSetErrorHandler** tells Visual Basic to assign ERR to a specific value when a Personal Daq error is encountered. The following line tells Visual Basic to set ERR to 100 when a Personal Daq error is encountered. Other languages work similarly; refer to specific language documentation as needed.

```
handle& = VBdaqOpen&("PDAQ123456")
ret& = VBdaqSetErrorHandler&(handle&, 100)

   On Error GoTo ErrorHandler
```

The **On Error GoTo** command in Visual Basic allows a user-defined error handler to be provided, rather than the standard error handler that Visual Basic uses automatically. The program uses **On Error GoTo** to transfer program control to the routine **ErrorHandler** if an error is encountered.

Personal Daq errors will send the program into the error handling routine. This is the error handler. Program control is sent here on error.

```
ErrorHandler:

   errorString$ = "ERROR in ADC1"
   errorString$ = errorString$ & Chr(10) & "BASIC Error :" + Str$(Err)
   If Err = 100 Then errorString$ = errorString$ & Chr(10) & "DaqBook Error
 : " + Hex$(daqErrno&)

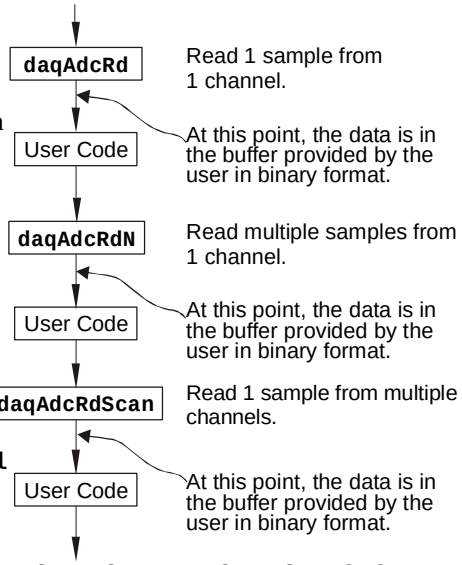
   MsgBox errorString$, , "Error!"

End Sub
```

Foreground Acquisition with One-Step Commands

This section shows the use of several one-step analog input routines. These commands are easier to use than low-level commands but less flexible in scan configuration. These commands provide a single function call to configure and acquire analog input data. This example demonstrates the use of four Personal Daq one-step ADC functions. Functions used include:

- **VBdaqAdcRdSingle&(ByVal handle&, ByVal chan&, sample!, ByVal gain&, ByVal flags&)**
- **VBdaqAdcRdNSingle&(ByVal handle&, ByVal chan&, buf!(), ByVal ScanCount&, ByVal triggerSource&, ByVal rising&, ByVal level%, ByVal Freq!, ByVal gain&, ByVal flags&)**
- **VBdaqAdcRdScanSingle&(ByVal handle&, ByVal startChan&, ByVal endChan&, buf!(), ByVal gain&, ByVal flags&)**



This program will initialize the Personal Daq hardware, then take readings from the analog input channels in the base unit (not the expansion module). For transporting data in and out of the Personal Daq driver, arrays are dimensioned.

```
Dim sample!(1), buf!(80), handle&, ret&, flags&, gain&
```

The following code assumes that the Personal Daq device has been successfully opened and the **handle&** value is a valid handle to the device. All the following one-step functions define the channel scan groups to be analog unipolar input channels. Specifying this configuration uses the **DafAnalog** value in the **flags** parameter. The **flags** parameter is a bit-mask field in which each bit specifies the characteristics of the channel(s) specified. In this case, the **DafAnalog** value is used as the bit mask for the specified **flags** parameter.

The next line requests 1 reading from 1 channel with a gain of ×1. The variable **PgainX1&** is actually a defined constant from **PDAQX.BAS**.

```
ret& = VBdaqAdcRdSingle&(handle&, 1, sample!(0), PGainX1&, DafAnalog)
Print "Result of AdcRd: " & Format$(Str$(sample%(0)), "0000")
```

The next line requests 10 readings from channel 1 at a gain of ×1, using immediate triggering at 1 kHz.

```
ret& = VBdaqAdcRdNSingle&(handle&, 1, buf!(), 10, DatsImmediate&, 0, 0, 1000!, _
PGainX1&, DafAnalog)
      Print "Results of AdcRdN: ";
      For x& = 0 To 9
         Print Format$(str$(buf%(x&)), "#### ");
      Next x&
```

The program will then collect one sample of channels 1 through 8 using the **VBdaqAdcRdScan** function.

```
ret& = VBdaqAdcRdScanSingle&(handle&, 1, 8, buf!(), PGainX1&, DafAnalog)
   Print "Results of AdcRdN: ";
   For x& = 0 To 7
   Print "Channel: " & Str$(x + 1) & " Data: " + Format$(buf%(x&), "####  ")
  Next x&
```

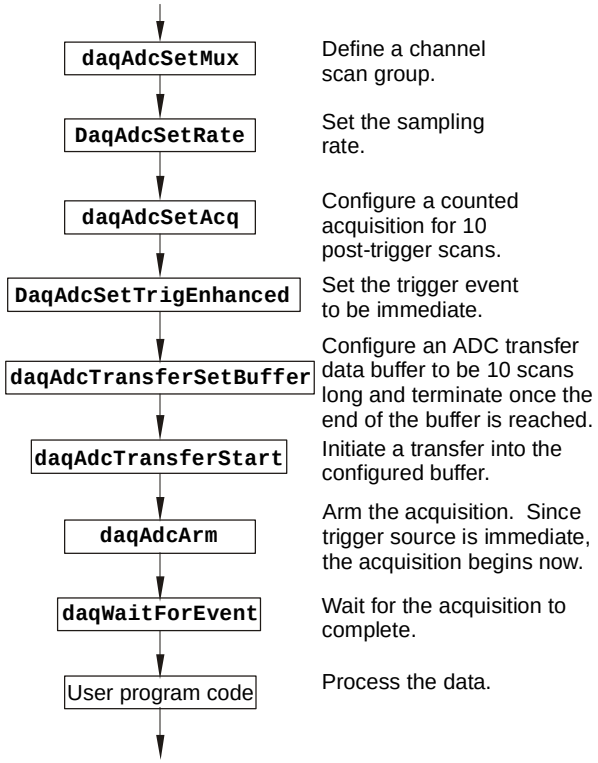
Counted Acquisitions Using Linear Buffers

This section sets up an acquisition that collects post-trigger A/D scans. This particular example demonstrates the setting up and collection of a fixed-length A/D acquisition in a linear buffer.

First, the acquisition is configured by setting up the channel scan group configuration, the acquisition frequency, the acquisition trigger and the acquisition mode. When configured, the acquisition is then armed by calling the **daqAdcArm** function.

At this point, the Personal Daq device trigger is armed and A/D acquisition will begin upon trigger detection. If the trigger source has been configured to be **DatsImmediate&**, A/D data collection will begin immediately.

This example will retrieve 10 samples from channels 1 through 8, triggered immediately with a 10 Hz sampling frequency and unity gain [for Personal Daq]. Functions used include:



- **VBdaqAdcSetMux&(handle&, startChan&, endChan&, gain&, flags&)**
- **VBdaqAdcSetRate&(handle&, Mode&, acqState&, regRate!, actualRate!)**
- **Function VBdaqAdcSetTrigEnhanced&(handle&, triggerSources&(), gains&(), adcRanges&(), trigSense&(), levels!(), hysteresis!(), chan&(), ChanCount&, opstr$)**
- **VBdaqAdcSetAcq&(handle&,mode&,preTrigCount&,postTrigCount&)**
- **VBdaqAdcTransferSetBufferSingle&(ByVal handle&, buf!(), ByVal ScanCount&, ByVal transferMask&)**
- **VBdaqAdcTransferStart&(handle&)**
- **VBdaqWaitForEvent&(handle&,daqEvent&)**

This program will initialize the Personal Daq hardware, then take readings from the analog input channels in the base unit (not the expansion channels). The functions used in this program are of a lower level than those used in the previous section and provide more flexibility.

```
Dim buf!(80), handle&, ret&, flags&
```

The following function defines the channel scan group. The function specifies a channel scan group from channel 1 through 8 with all channels being analog unipolar input channels with a gain of ×1. Specifying this configuration uses **PgainX1** in the gain parameter and the **DafAnalog** value in the **flags** parameter. The **flags** parameter is a bit-mask field in which each bit specifies the characteristics of the specified channel(s). Although each channel can set up independently (and different from the others) in this example we will set them all the same.

```
ret& = VBdaqAdcSetMux&(handle&,1, 8, PgainX1&,
 DafAnalog&+DafDifferential&+DafMeasDuration610&)
```

Next, set the internal sample rate to 10 Hz.

```
ret& = VBdaqAdcSetRate&(handle&,DarmFrequency&,DaasPostTrig&,10!,actual!)
```

The acquisition mode needs to be configured to be fixed length acquisition with no pre-trigger scan data and 10 scans of post-trigger scan data. The mode is set to **DaamNShot&**, which will configure the acquisition as a fixed-length acquisition that will terminate automatically upon the satisfaction of the post-trigger count of 10.

```
ret& = VBdaqAdcSetAcq&(handle&,DaamNShot&, 0, 10)
```

The acquisition begins upon detection of the trigger event. The trigger event is configured with **daqAdcSetTrigEnhanced**. The next line defines the trigger event to be the immediate trigger source. This is the source that will start the acquisition immediately. The variable **DatsImmediate&** is a constant defined in **PDAQX.BAS**. Since the trigger source is configured as immediate, the other trigger parameters are not applied, but the arrays (**adcRanges** and **trigSense**) must still be initialized.

```
For i = 0 To ChanCount - 1
        trgSrc&(i) = DatsImmediate&
        adcRanges&(i) = 0
        trigSense&(i) = 0
        levels!(i) = 0
        hysteresis!(i) = 0
   Next i
   ret& = VBdaqAdcSetTrigEnhanced&(handle&, trgSrc&(), gains&(), adcRanges&(),
          trigSense&(), levels!(), hysteresis!(), chans&(), 0, "")
```

A buffer now is configured to hold the A/D data to be acquired. Since this is to be a fixed length transfer to a linear buffer, the buffer cycle mode should be turned off with **DatmCycleOff&**. The buffer size is set to 10 scans.

Note: The user-defined buffer must have been allocated with sufficient storage to hold the entire transfer prior to invoking the following line:

```
ret& = VBdaqAdcTransferSetBufferSingle&(handle&, buf!(), 10, DatmUpdateBlock
+ DatmCycleOff)
```

With all acquisition parameters being configured, the acquisition can now be armed. Once armed, the acquisition will begin immediately upon detection of the trigger event. As in the case of the immediate trigger, the acquisition will begin immediately upon execution of the **daqAdcArm** function.

```
ret& = VBdaqAdcArm&(handle&)
```

After setting up and arming the acquisition, the data is immediately ready to be collected. Had the trigger source been anything other than immediate, the data would only be ready after the trigger had been satisfied. The following line initiates an A/D transfer from the Personal Daq device to the defined user buffer.

```
ret& = VBdaqAdcTransferStart&(handle&)
```

Wait for the transfer to complete in its entirety, then proceed with normal application processing. This can be accomplished with the **daqWaitForEvent** command. The **daqWaitForEvent** allows the application processing to become blocked until the specified event has occurred. **DteAdcDone**, indicates that the event to wait for is the completion of the transfer.

```
ret& = VBdaqWaitForEvent(handle&,DteAdcDone&)
```

At this point, the transfer is complete; all data from the acquisition is available for further processing.

```
Print "Results of Transfer:"
For i& = 0 To 9
      Print "Scan "; Format$(Str$(i& + 1), "00"); " -->";
      For k& = k& To k& + 7
         Print Format$(Str$(buf%(k&)), "00000"); "   ";
      Next k&
      Print
Next i&
```

Indefinite Acquisition, Direct-To-Disk Using Circular Buffers

This program demonstrates the use of circular buffers in cycle mode to collect analog input data directly to disk. In cycle mode, this data transfer can continue indefinitely. When the transfer reaches the end of the physical data array, it will reset its array pointer back to the beginning of the array and continue writing data to it. Thus, the allocated buffer can be used repeatedly like a FIFO buffer.

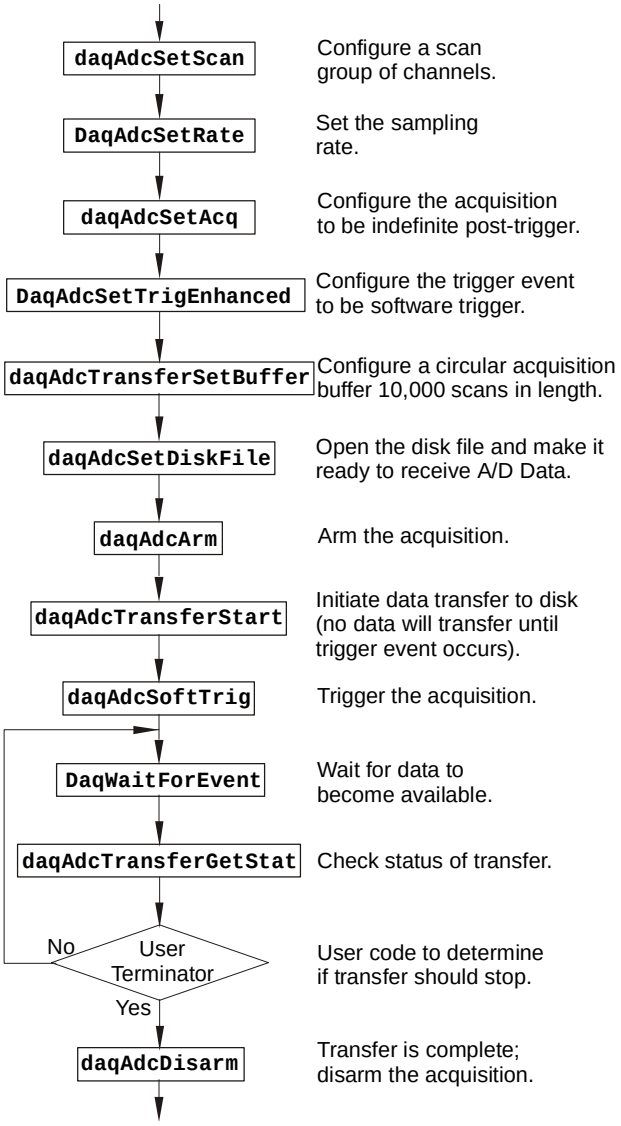
The API has built-in direct-to-disk functionality. Therefore, very little needs to be done by the application to configure direct-to-disk operations.

First, the acquisition is configured by setting up the channel scan group configuration, the acquisition frequency, the acquisition trigger and the acquisition mode. Once configured, the transfer to disk is set up and the acquisition is armed by calling the **daqAdcArm** function.

At this point, the Personal Daq device trigger is armed and A/D acquisition to disk will begin immediately upon trigger detection.

This example will retrieve an indefinite amount of scans for channels 1 through 8, triggered via software with a 3 Hz sampling frequency and unity gain. Functions used include:

- **VBdaqAdcSetScan&(handle&, channels&(), gains&(), flags&(), ChanCount&)**
- **VBdaqAdcSetRate&(handle&,mode&,state&,requested!,actual!)**
- **Function VBdaqAdcSetTrigEnhanced&(handle&, triggerSources&(), gains&(), adcRanges&(), trigSense&(), levels!(), hysteresis!(), chan&(), ChanCount&, opstr$)**
- **VBdaqAdcSetAcq&(handle&,mode&,preTrigCount&,postTrigCount&)**
- **VBdaqAdcTransferSetBufferSingle&(handle&, buf!(), ScanCount&, transferMask&)**
- **VBdaqAdcTransferStart&(handle&)**
- **VBdaqAdcTransferGetStat&(handle&,status&,retCount&)**
- **VBdaqWaitForEvent&(handle&,daqEvent&)**
- **VBdaqAdcSetDiskFile&(handle&,filename$,openMode&,preWrite&)**



This program will initialize the Personal Daq hardware, then take readings from the analog input channels in the base unit (not the expansion channels) and store them to disk automatically. The following lines demonstrate channel scan group configuration using the **daqAdcSetScan** command.

Note: Flags may be channel-specific.

```
Dim handle&, ret&, channels&(8), gains&(8) flags&(8)
Dim buf!(80,000), active&, count&
  Dim bufsize&
  bufsize& = 10000
```

```
' Define arrays of channels and gains : 1-8 , unity gain
For x& = 0 To 7
  channels&(x&) = x& + 1
  gains&(x&) = PgainX1&
  flags&(x&) = DafAnalog& + DafSingleEnded&
Next x&
' Load scan sequence FIFO
ret& = VBdaqAdcSetScan&(handle&,channels&(), gains&(), flags&(), 8)
```

The acquisition mode needs to be configured to be fixed-length acquisition with no pre-trigger scan data and 10 scans of post-trigger scan data. The mode is set to **DaamInfinitePost&**, which will configure the acquisition as having indefinite length and, as such, will be terminated by the application. In this mode, the pre- and post-trigger count values are ignored.

```
ret& = VBdaqAdcSetAcq&(handle&,DaamInfinitePost&, 0, 0)
```

Next, set the internal sample rate to 3 Hz.

```
ret& = VBdaqAdcSetRate&(handle&,DarmFrequency&,DaasPostTrig&,3!,actual!)
```

The acquisition begins upon detection of the trigger event. The trigger event is configured with **(daqAdcSetTrigEnhanced)**. The next line defines the trigger event to be the immediate trigger source which will start the acquisition immediately. The variable **DatsSoftware&** is a constant defined in **PDAQX.BAS**. Since the trigger source is configured as immediate, the other trigger parameters are not needed.

```
TrigSource& = DatsSoftware&

For i = 0 To 7
      trgSrc&(i) = DatsSoftware&
      adcRanges&(i) = 0
      trigSense&(i) = 0
      levels!(i) = 0
      hysteresis!(i) = 0
Next i
ret& = VBdaqAdcSetTrigEnhanced&(handle&, trgSrc&(), gains&(), adcRanges&(),
         trigSense&(), levels!(), hysteresis!(), channels&(), 0, "")
```

A buffer now is configured to hold the A/D data to be acquired. This buffer is necessary to hold incoming A/D data while it is being prepared for disk I/O. Since this is to be an indefinite-length transfer to a circular buffer, the buffer cycle mode should be turned on with **DatmCycleOn&**. For efficiency, block update mode is specified with **DatmUpdateSingle&**. The buffer size is set to 10,000 scans. The buffer size indicates only the size of the circular buffer, not the total number of scans to be taken.

```
ret& = VBdaqAdcTransferSetBufferSingle&(handle&, buf!(), bufsize&,
 DatmUpdateSingle& + DatmCycleOn&)
```

Note: Two types of VB function wrappers exist for the API functions that reference data buffers. One type references *an integer buffer*, and the other references a *single-precision floating point buffer*. The wrappers that reference single-precision floating point buffers have the word **Single** appended to the function name. For example, **VBdaqAdcTransferSetBufferSingle** would be used to set a single-precision floating point buffer, whereas **VBdaqAdcTransferSetBuffer** be used to set an integer buffer.

Now the destination disk file is configured and opened. For this example, the disk file is a new file to be created by the driver. After the following line has been executed, the specified file will be opened and ready to accept data.

```
ret& = VBdaqAdcSetDiskFile&(handle&,"c:pdaqdata.bin", DaomCreateFile&, 0)
```

With all acquisition parameters being configured and the acquisition transfer to disk configured, the acquisition can now be armed. Once armed, the acquisition will begin immediately upon detection of the trigger event. As in the case of the immediate trigger, the acquisition will begin immediately upon execution of the **daqAdcArm** function.

```
ret& = VBdaqAdcArm&(handle&)
```

After setting up and arming the acquisition, data collection will begin upon satisfaction of the trigger event. Since the trigger source is software, the trigger event will not take place until the application issues the software trigger event. To prepare for the trigger event, the following line initiates an A/D transfer from the Daq* device to the defined user buffer and, subsequently, to the specified disk file. No data is transferred at this point, however.

```
ret& = VBdaqAdcTransferStart&(handle&)
```

The transfer has been initiated, but no data will be transferred until the trigger event occurs. The following line will signal the software trigger event to the driver; then A/D input data will be transferred to the specified disk file as it is being collected.

```
ret& = VBdaqAdcSoftTrig&(handle&)
```

Both the acquisition and the transfer are now currently active. The transfer to disk will continue indefinitely until terminated by the application. The application can monitor the transfer process with the following lines of code:

```
acqTermination& = 0
Do
    ‘ Wait here for new data to arrive
    ret& = VBdaqWaitForEvent(handle&,DteAdcData&)

    ‘ New data has been transferred - Check status
    ret& = VBdaqAdcTransferGetStat&(handle&,active&,retCount&)

    ‘ Code may be placed here which will process the buffered data or
    ‘ perform other application activities.
    ‘
    ‘ At some point the application needs to determine the event on which
    ‘ the direct-to-disk acquisition is to be halted and set the
    ‘ acqTermination flag.

Loop While acqTermination& = 0
```

At this point the application is ready to terminate the acquisition to disk. The following line will terminate the acquisition to disk and will close the disk file.

```
ret& = VBdaqAdcDisarm&(handle&)
```

The acquisition as well as the data transfer has been stopped. We should check status one more time to get the total number of scans actually transferred to disk.

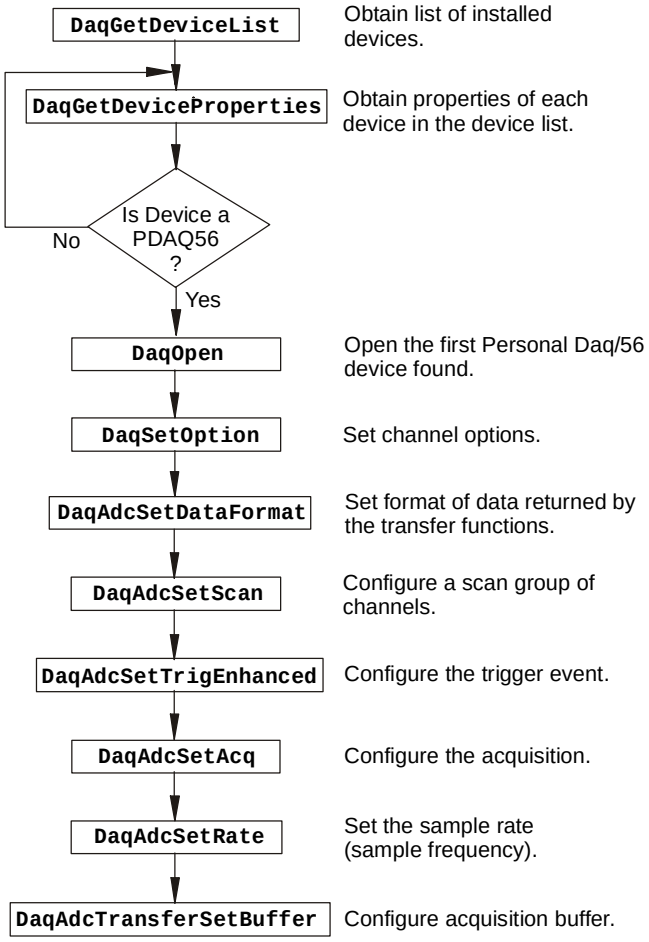
```
ret& = VBdaqAdcTransferGetStat(handle&,active&,retCount&)
```

The specified disk file is now available. The **`retCount&`** parameter will indicate the total number of scans transferred to disk.

Multiple Channel Types

This section demonstrates how to configure multiple channels on a Personal Daq and acquire data using these channels in the same scan sequence. After all of the prior code has been executed, the transfer can be started and the acquisition can be armed in the same manner as shown in the other models. Functions used include:

- **VBdaqGetDeviceList&(ByRef deviceList() As String, ByRef deviceCount&)**
- **VBdaqGetDeviceProperties&(ByVal daqName$, deviceProps As DaqDevicePropsT)**
- **VBdaqOpen&(ByVal daqName$)**
- **VBdaqSetErrorHandler&(ByVal handle&, ByVal handler&)**
- **VBdaqSetOption&(ByVal handle&, ByVal chan&, ByVal flags&, ByVal optionType&, ByVal optionValue!)**
- **VBdaqAdcSetDataFormat&(ByVal handle&, ByVal RawDataFormat&, ByVal PostProcFormat&)**
- **VBdaqAdcSetScan&(ByVal handle&, channels&(), gains&(), flags&(), ByVal ChanCount&)**
- **VBdaqAdcSetTrigEnhanced&(ByVal handle&, ByRef triggerSources&(), ByRef gains&(), ByRefadcRanges&(), ByRef trigSense&(), ByRef levels!(), ByRef hysteresis!(), ByRef chan&(), ChanCount&, opstr$)**
- **VBdaqAdcSetAcq&(ByVal handle&, ByVal Mode&, ByVal preTrigCount&, ByVal postTrigCount&)**
- **VBdaqAdcSetRate&(ByVal handle&, ByVal Mode&, ByVal acqState&, ByVal regRate!, ByRef actualRate!)**
- **VBdaqAdcTransferSetBufferSingle&(ByVal handle&, buf!(), ByVal ScanCount&, ByVal transferMask&)**



Multiple Channel Types Model for Personal Daq/56

The definitions below will be used to specify the channel count, scan count, data buffer size, starting channel, and acquisition clock frequency respectively. This model assumes the use of a Personal Daq/56 device, on which there are 10 analog input channels, 2 digital I/O ports, and 4 frequency input channels available that will be configured for use in the same scan sequence. The acquisition will consist of 10 scans per channel. The first channel in the channel sequence array is 1. The rate at which channel scans are collected is 0.1 Hz.

```
Const ChanCount& = 16
Const ScanCount& = 10
Const TotalCount& = (ScanCount& * ChanCount&)
Const startChan = 1
Const Freq! = 0.1
```

We define constants for the various flags that will be placed in the flag sequence array. The flags that are ultimately placed in the array depend on the channel type of the channel specified in the same subscript of the channel sequence array.

```
Const CtrFlags& = DafCtrPulse Or DafScanDigital
Const DigFlags& = DafScanDigital
Const FreqFlags& = DafCtrFreq Or DafScanDigital
Const AnaFlags& = DafAnalog Or DafDifferential Or DafMeasDuration610
```

The handle& variable will be assigned a handle to an opened Personal Daq/56 device using the VBDaqOpen function. This handle must be used with most API functions. The data buffer consists of single precision floating-point elements, the number of which is proportional to the product of the scan count and the channel count. The variable i is a counter variable that is used to perform iteration operations.

```
Dim handle&
Dim buf!(TotalCount&)
Dim i&
```

The deviceList string array, which is initialized using the VBDaqGetDeviceList function, is used to hold a list of currently configured devices, and the deviceCount variable contains the number of devices in the deviceList array.

```
Dim deviceList$()
Dim deviceCount&
```

The following scan sequence arrays will be used to store channel numbers, the flags for the channels, and the gain settings for the channels. If element 0 of the channel sequence array has a value of 1 (channel 1), then element 0 of the flag and gain sequence arrays will pertain to the flags and gain settings of channel 1.

```
Dim chans&(ChanCount&)
Dim flags&(ChanCount&)
Dim gains&(ChanCount&)
```

In order to configure the trigger event, the five arrays shown below must be allocated. These arrays are allocated proportional to the channel count and follow the same order as the scan sequence arrays, in which the first element corresponds to the setting pertaining to the first channel in the channel sequence array. These arrays specify the source of the trigger event, appropriate polarity flags, sensitivity flags, analog trigger levels, and hysteresis values.

```
Dim trgSrc&(ChanCount&)
Dim adcRanges&(ChanCount&)
Dim trigSense&(ChanCount&)
Dim levels!(ChanCount&)
Dim hysteresis!(ChanCount&)
```

When the VBDaqAdcSetRate function is used to configure the scan rate, the variable actualRate! will contain the actual rate for which the device has been programmed in the event that the requested rate is unattainable. The ret& variable is used to hold the return code of all other DaqX function calls

```
Dim actualRate!
dim ret&
```

A list of devices that are currently configured on the system along with the number of devices in the list should be obtained using the VBDaqGetDeviceList function.

```
ret& = VBdaqGetDeviceList(deviceList, deviceCount
```

One of the methods used to dynamically locate and open a configured Personal Daq device is shown below. The handle& variable is initialized to (-1) so it can be used at the end of the iteration to determine if a device was successfully opened. The iteration on deviceIndex& will obtain the device properties for each device in the device list and use the deviceType property to locate the first Personal Daq/56 device (if any) that is installed and configured in the system. The first Personal Daq/56 that is found in the system is opened using the VBdaqOpen function, which returns a value of –1 if the specified device cannot be opened, thus changing the value of the handle in the case that the device is successfully initialized.

```
handle& = -1
For deviceIndex& = 0 To deviceCount – 1
ret& = VBdaqGetDeviceProperties(deviceList(deviceIndex), deviceProps)
If deviceProps.DeviceType = PersonalDaq56 Then
handle& = VBdaqOpen&(deviceList(deviceIndex))
Exit For
End If
Next deviceIndex&
```

If the handle is still –1 at this point, we know that either a Personal Daq/56 device was not located on the system, or that whatever Personal Daq/56 device was located on the system could not be properly opened.

```
If handle& = -1 Then
           Print "Cannot open PersonalDaq!"
Exit Sub
End If
```

An iteration on the channel count is used to initialize the scan sequence arrays and configure the frequency and pulse counter channels. The following table indicates the content of the scan sequence arrays after this iteration is completed:

Subscript	Gain Sequence Array gains()	Channel Sequence Array channels()	Flag Sequence Array flags()
Analog Input Channels			
0	PGainX1	1 (startChan)	AnaFlags
1	PGainX1	2	AnaFlags
2	PGainX1	3	AnaFlags
3	PGainX1	4	AnaFlags
4	PGainX1	5	AnaFlags
5	PGainX1	6	AnaFlags
6	PGainX1	7	AnaFlags
7	PGainX1	8	AnaFlags
8	PGainX1	9	AnaFlags
9	PGainX1	10	AnaFlags
Digital Ports (Does not apply to Personal Daq/54)			
10	PGainX1	1 (startChan)	DigFlags
11	PGainX1	2	DigFlags
Frequency Input / Pulse Count Channels (Does not apply to Personal Daq/54)			
12	PGainX1	1 (startChan)	FreqFlags
13	PGainX1	2	FreqFlags
14	PGainX1	3	CtrFlags
15	PGainX1	4	CtrFlags

```
For i = 0 To ChanCount
      If (0 <= i) And (i < 10) Then
      ' Analog input channels
         chans&(i) = startChan + 1
         flags&(i) = AnaFlags
      ElseIf (10 <= i) And (i < 12) Then
      ' Digital I/O ports
         chans&(i) = startChan + (i - 10)
         flags&(i) = DigFlags
      ElseIf (12 <= i) And (i < 14) Then
      ' Frequency input channels configured for frequency input
         chans&(i) = startChan + (i - 12)
         flags&(i) = FreqFlags
```

The VBdaqSetOption function is used to set individual channel options. The sequence of calls below configures the current channel chans&(i&) to measure 0 pulses per scan (DcotpDaqPulses) based on time measurements between the successive rising edges of the input signal, disables debouncing, sets the minimum and maximum values for the frequency range (maximum frequency range value is 1000 Hz), and configures the measurement rate to be 1 Hz.

```
        ret& = VBdaqSetOption(handle&, chans&(i&), DcofChannel, DcotpDaqPulses, DcovPulseCount)
        ret& = VBdaqSetOption(handle&, chans&(i&), DcofChannel, DcotpDaqRising,
            DcovEdgeRising)
        ret& = VBdaqSetOption(handle&, chans&(i&), DcofChannel, DcotpDaqDebounceTime,
            DcovDebounce0)
        ret& = VBdaqSetOption(handle&, chans&(i&), DcofChannel, DcotpDaqMinFreq, 0!)
        ret& = VBdaqSetOption(handle&, chans&(i&), DcofChannel, DcotpDaqMaxFreq, 1000!)
        ret& = VBdaqSetOption(handle&, chans&(i&), DcofChannel, DcotpDaqFreqRes, 1!)
    ElseIf (14 <= i) And (i < 16) Then
    ' Frequency input channels configured for pulse count
       chans&(i) = startChan + 2 + (i - 14)
       flags&(i) = CtrFlags
        ret& = VBdaqSetOption(handle&, chans&(i&), DcofChannel, DcotpDaqPulses, DcovPulseCount)
        ret& = VBdaqSetOption(handle&, chans&(i&), DcofChannel, DcotpDaqRising,
            DcovEdgeRising)
        ret& = VBdaqSetOption(handle&, chans&(i&), DcofChannel, DcotpDaqDebounceTime,
            DcovDebounce0)
    End If
    gains(i) = PGainX1
  Next i
```

The VBdaqAdcSetDataFormat function is used to configure the driver to return the raw data in floating point format, which is a requirement for Personal Daq software. Post-acquisition data will be presented as raw data.

```
ret& = VBdaqAdcSetDataFormat&(handle&, DardfFloat&, DappdfRaw)
```

The scan group consisting of multiple channels must be configured using the VBdaqAdcSetScan function.

```
ret& = VBdaqAdcSetScan&(handle&, chans&(), gains&(), flags&(), ChanCount&)
```

The arrays used to specify the parameters of the trigger event are initialized prior to the invocation of the VBdaqAdcSetTrigEnhanced function, which will use the parameters to configure a trigger event for the acquisition. Based on the parameters below, the acquisition will trigger immediately upon arming. The adcRanges&, trigSense&, levels!, and hysteresis! arrays are initialized, but the values will not affect the trigger event configuration, as DatsImmediate configures an immediate trigger.

```
For i = 0 To ChanCount - 1
   trgSrc&(i) = DatsImmediate&
   adcRanges&(i) = DarBiMinus5to5V
   trigSense&(i) = DetsRisingEdge&
   levels!(i) = 0
   hysteresis!(i) = 0
Next i
ret& = VBdaqAdcSetTrigEnhanced&(handle&, trgSrc&(), gains&(), adcRanges&(), _
  trigSense&(), levels!(), hysteresis!(), _chans&(), ChanCount&, "")
```

Next, the acquisition mode is characterized as one that continues until the previously defined scan count has been satisfied.

```
ret& = VBdaqAdcSetAcq&(handle&, DaamNShot&, 0, ScanCount&)
```

The post-trigger scan rate is configured using the previously defined Freq! value, and the actualRate! variable is passed by reference to hold the actual rate for which the device was configured in the event that the requested rate could not be configured.

```
ret& = VBdaqAdcSetRate&(handle&, DarmFrequency, DaasPostTrig, Freq!, actualRate!)
```

In VisualBASIC, there are two different function wrappers for the VBDaqAdcTransferSetBuffer function, one of which is shown below. The other function wrapper is named VBdaqAdcTransferSetBuffer. The only difference between the two wrappers is manifested in the buf parameter. There are essentially two different buffer types that can be used, one containing integer elements and the other containing single precision floating-point elements. The VBdaqAdcTransferSetBufferSingle function should be called to use a single precision floating-point buffer in VisualBASIC.

```
ret& = VBdaqAdcTransferSetBufferSingle&(handle&, buf!(), ScanCount&, DatmReturn)
```

Note: This section, *Multiple Channel Types*, demonstrated the configuration of acquisitions using multiple channel types in the same scan sequence. After all of the prior code has been executed, the transfer can be started and the acquisition can be armed in the same manner as shown in the other models.

Summary Guide of Selected API Functions

Simple One-Step Routines

For single gain, consecutive channel, foreground transfers, use the following functions:

Foreground Operation	Single Scan	Multiple Scans
Single Channel	**daqAdcRd**	**daqAdcRdN**
Consecutive Multiple Channels	**daqAdcRdScan**	**daqAdcRdScanN**

Complex A/D Scan Group Configuration Routines

For non-consecutive channels, high-speed digital channels, multiple gain settings, or multiple polarity settings, use the SetScan functions.

daqAdcSetScan	Set scan sequence using arrays of channel and gain values.
daqAdcSetMux	Set a contiguous scan sequence using single gain, polarity and channel flag values

Trigger Options

After the scan is set, the trigger needs to be set. If a software trigger is selected, the start time of the scan depends on the application calling **daAdcSoftTrig**.

DaqAdcSetTrigEnhanced	Configures the device for enhanced triggering.

Multiple Scan Timing

If the acquisition is to have multiple scans and the trigger mode is one-shot, the pacer clock needs to be set with one of the following functions:

daqAdcSetRate	Set/Get the specified frequency or period for the specified mode.
daqAdcSetFreq	Set the pacer clock to the given frequency.

A/D Acquisition

A/D acquisition settings are not active until the acquisition is armed.

daqAdcArm	Arm an A/D acquisition using the current configuration. If the trigger source was set to be immediate, the acquisition will be triggered immediately.
daqAdcDisarm	Disarm the current acquisition if one is active. This command will disarm the current acquisition and terminate any current A/D transfers.
daqAdcSetAcq	Define the mode of the acquisition and set the pre-trigger and post-trigger acquisition counts, if applicable.
daqAdcAcqGetStat	Return the current state of the acquisition as well as the total number of scans transferred thus far as well as the trigger scan position, if applicable.

A/D Data Transfer

After the acquisition is started, the data needs to be transferred to the application buffer. Three routines are used:

DaqAdcTransferSetBuffer	Configure a buffer for A/D transfer. Allows configuration of the buffer for block and single reading update modes as well as linear and circular buffer definitions.
DaqAdcTransferStart	Start a transfer from the Personal Daq device to the buffer specified in the **daqAdcTransferSetBuffer** command
DaqAdcTransferStop	Stop a transfer from the Personal Daq device to the buffer specified in the **daqAdcTransferSetBuffer** command

To find out whether a background A/D transfer is complete or to stop transfers, use the following function:

DaqAdcTransferGetStat	Return current A/D transfer status as well as a count representing the total number of transferred scans or the number of scans available.

Digital Functions (Does not apply to Personal Daq/54)

After the digital group is configured, the ports can be read or written a byte at a time. Note that “low/high” and “digital I/O” are accessed a nibble at a time. A single bit of a digital channel can be read or written using the following routines:

daqIOReadBit	Return indicated bit from selected channel.
daqIOWriteBit	Send indicated bit to selected channel.

API Commands B

Overview

This appendix includes information regarding *type-sensitive API function wrappers for Visual Basic*, Personal Daq driver commands for *Windows*™ in 32-bit mode, parameter definitions such as A/D channels, event-handling, hardware, A/D gain, general I/O, digital I/O port connection (page B-29), and API error codes (page B-33).

Type-Sensitive API Function Wrappers for Visual Basic

Each API function has a wrapper serving as an interface between the Visual Basic program and the actual API function calls. API function prototypes (made available by **PDAQX.DLL)** are declared in the header file (**PDAQX.BAS**) as follows:

```
Declare Function BdaqAdcTransferSetBuffer& Lib "pdaqx.dll" Alias _
"daqAdcTransferSetBuffer" (ByVal handle&, buf As Any, ByVal ScanCount&, _
ByVal transferMask&)
```

Ideally, any application written in Visual Basic should call the function above by using the function wrapper, **VbdaqAdcTransferSetBuffer**, rather than calling the API function directly via **BdaqAdcTransferSetBuffer**. The standard wrapper for this function is:

```
Function VBdaqAdcTransferSetBuffer&(ByVal handle&, buf%(), ByVal ScanCount&,
ByVal transferMask&)
   Dim lb1&
   lb1& = LBound(buf%)
   daqErrno& = BdaqAdcTransferSetBuffer&(handle&, buf%(lb1&), ScanCount&,
      transferMask&)
If ((daqErrno& <> 0) And (daqErrnum& <> 0)) Then Error daqErrnum&

      VBdaqAdcTransferSetBuffer& = daqErrno&
End Function
```

For Visual Basic versions preceding 5.0, proper error handling is made possible with the use of wrappers. In cases where API functions return structured information, the function wrappers assist in presenting that information to the programmer. For **VBdaqGetDeviceList** the information in the byte array (returned by the DLL) is transferred to a standard VB string array that is easier to work with on the programming end.

It is important to note that while every API function supported in Visual Basic has at least one associated wrapper, the API functions that include buffer references have two different associated wrappers (dependent on the buffer's data type). *For example*, consider the **daqAdcTransferSetBuffer** function. The prototype has a buffer parameter declared as type **Any** (see previous wrapper code). Data acquisition buffers can contain either integers (16-bit numbers), or single-precision floating-point numbers (Single data type).

VBdaqAdcTransferSetBuffer - would be used to call the **daqAdcTransferSetBuffer** function with a buffer containing *integers*.

VBdaqAdcTransferSetBufferSingle - would be used for a buffer containing *single-precision floating-point numbers*.

The only difference between the **VBdaqAdcTransferSetBuffer** and **VBdaqAdcTransferSetBufferSingle** functions is that the **buf** parameters consist of different data types for each wrapper.

The following table lists buffer-related API functions and associated wrappers.

API Function	Integer Wrapper	Floating-Point Wrapper
BdaqAdcTransferSetBuffer	VBdaqAdcTransferSetBuffer	VBdaqAdcTransferSetBufferSingle
BdaqAdcTransferBufData	VBdaqAdcTransferBufData	VBdaqAdcTransferBufDataSingle
BdaqAdcRdScan	VBdaqAdcRdScan	VBdaqAdcRdScanSingle
BdaqAdcRdN	VBdaqAdcRdN	VBdaqAdcRdNSingle
BdaqAdcRdScanN	VBdaqAdcRdScanN	VBdaqAdcRdScanNSingle
BdaqAdcGetBufData	VBdaqAdcGetBufData	VBdaqAdcGetBufDataSingle
BdaqAdcRd	VBdaqAdcRd	VBdaqAdcRdSingle
BdaqCvtRawDataFormat	VBdaqCvtRawDataFormat	VBdaqCvtRawDataFormatSingle

Prototype Commands, Listed by Function Type (as defined in driver header files)

Function	Description	Page
Device Initialization Prototypes		
daqOpen	Open a session with the Personal Daq	B-25
daqOpenList	Open, initialize, and prepare devices (specified by the **deviceList** parameter)	B-25
daqClose	End communication with the Personal Daq	B-18
daqCloseList	Close devices opened by **daqOpenList**	B-19
daqOnline	Check online status of the Personal Daq	B-24
daqGetDeviceCount	Return the number of currently configured devices	B-21
daqGetDeviceList	Return the list of currently configured devices	B-21
daqGetDeviceProperties	Return the properties of specified device	B-21
Error Handler Function Prototypes		
daqSetDefaultErrorHandler	Set the default error handler	B-26
daqSetErrorHandler	Specify a user defined routine to call when an error occurs in any command	B-26
daqProcessError	Process a driver defined error condition	B-26
daqGetLastError	Return the last logged error condition	B-22
daqDefaultErrorHandler	Call the default error handler	B-20
daqFormatError	Return text string for specified error	B-20
Event Handling Function Prototypes		
daqSetTimeout	Set the time-out value for the Personal Daq operation	B-27
daqWaitForEvent	Wait for specified Personal Daq device event	B-28
daqWaitForEvents	Wait for multiple specified Personal Daq device events	B-28
Utility Function Prototypes		
daqGetDriverVersion	Return the software version	B-22
Expansion Configuration Prototypes		
daqSetOption	Set options for a device's channel/signal path configuration	B-27
Custom ADC Acquisition Prototypes - Scan Sequence		
daqAdcSetMux	Configure a scan specifying start and end channels	B-12
daqAdcSetScan	Configure up to 256 channels making up an A/D or HS digital input scan	B-13
daqAdcGetScan	Read the current scan configuration	B-4
Custom ADC Acquisition Prototypes - Trigger		
daqAdcSetTrigEnhanced	Configure an A/D trigger with multiple trigger-event conditions	B-14
Custom ADC Acquisition Prototypes - Scan Rate and Source		
daqAdcSetRate	Configure the ADC scan rate with the **mode** parameter	B-12
daqAdcSetFreq	Configure the pacer clock frequency in Hz	B-11
daqAdcGetFreq	Read the current pacer clock frequency	B-4
Custom ADC Acquisition Prototypes - Scan Count, Rate and Source		
daqAdcSetAcq	Set acquisition configuration information	B-9
Custom ADC Acquisition Prototypes - Direct-to-Disk		
daqAdcSetDiskFile	Specify the disk file for direct-to-disk transfers	B-10
Custom ADC Acquisition Prototypes - Acquisition Control		
daqAdcArm	Arm an acquisition	B-3
daqAdcDisarm	Disarm an acquisition	B-4
Custom ADC Acquisition Prototypes - Data Transfer without Buffer Allocation		
daqAdcAcqGetStat	Returns current state of acquisition	B-3
daqAdcTransferBufData	Transfer scans from driver-allocated buffer to user-specified buffer	B-15
daqAdcTransferSetBuffer	Setup a destination buffer for an ADC transfer	B-17
daqAdcTransferStart	Start an ADC transfer	B-18
daqAdcTransferGetStat	Retrieve status of an ADC transfer	B-16
daqAdcTransferStop	Stop an ADC transfer	B-18
One-Step ADC Acquisition Prototypes		
daqAdcRd	Configure an A/D acquisition and read one sample from a channel	B-5
daqAdcRdScan	Configure an A/D acquisition and read one scan	B-7
daqAdcRdN	Configure an A/D acquisition and read multiple scans from a channel	B-6
daqAdcRdScanN	Configure an A/D acquisition and read multiple scans	B-8
Data Format and Conversion Prototypes		
daqAdcSetDataFormat	Set the raw and post-acquisition data formats	B-10
daqAdcSetFilter	Sets the driver to perform filtering of analog channels	B-11
daqCvtRawDataFormat	Convert raw data to a specified format	B-19
daqCvtSetAdcRange	Set the ADC Voltage Range for the conversion routines	B-20
General I/O Prototypes - Read/Write		
daqIOReadBit	Read a DIO bit (channel)	B-23
daqIORead	Read a DIO byte (8 channels)	B-22
daqIOWriteBit	Write a DIO bit (channel)	B-24
daqIOWrite	Write a DIO byte (8 channels)	B-23

Commands in Alphabetical Order

The following pages give the details for each API command. Listed in alphabetical order, each section starts with a table that summarizes the main features of the command (C, Visual BASIC, and Delphi language prototypes and their related parameters). An explanation follows with related information and in some cases a programming example.

Note: Commands, parameters, values, and code all use a bold mono-spaced **`Courier`** font to distinguish characters and avoid ambiguity.

daqAdcAcqGetStat

DLL Function	`daqAdcAcqGetStat(DaqHandleT handle, PDWORD active, PDWORD preTrigCount, PDWORD postTrigCount, PDWORD totalAvail, PDWORD bufCycles, DWORD bufPosition);`	
C	`daqAdcAcqGetStat(DaqHandleT handle, PDWORD active, PDWORD preTrigCount, PDWORD postTrigCount, PDWORD totalAvail, PDWORD bufCycles, DWORD bufPosition);`	
Visual BASIC	`VBdaqAdcAcqGetStat&(ByVal handle&, ByRef active&, ByRef preTrigCount&, ByRef postTrigCount&, ByRef totalAvail&, ByRef bufCycles&, ByRef bufPosition&)`	
Delphi	`daqAdcAcqGetStat(handle:DaqHandleT; var active:DWORD; var preTrigCount:DWORD; var postTrigCount:DWORD; var totalAvail:DWORD; var bufCycles:DWORD; var bufPosition:DWORD)`	
Parameters	`handle`	Handle of device from which to obtain status.
	`active`	Indicates the current state of the acquisition.
	`preTrigCount`	Indicates the number of pre-trigger scans acquired.
	`postTrigCount`	Indicates the number of post-trigger scans acquired.
	`totalAvail`	Indicates the total number of available scans.
	`bufCycles`	Indicates the number of buffer cycles processed.
	`BufPosition`	Indicates the buffer position.
Returns	`DerrNoError`	No error
See Also	`daqAdcTransferGetStat`	
Program References	None	
Used With	All devices	

Description

`daqAdcAcqGetStat` allows you to retrieve the current state of the acquisition in terms of the number of pre-trigger and post-trigger scans acquired, the total amount of acquired scans available for transfer, the number of buffer processing cycles that have occurred, and the current position in the acquisition buffer.

The active parameter will indicate the current state of the acquisition in the form of a bit mask. Refer to the ADC Acquisition/Transfer Active Flag Definitions (in the ADC Miscellaneous Definitions table) for valid bit-mask states.

The **`preTrigCount`** and **`postTrigCount`** parameters return the total number of pre-trigger and post-trigger scans acquired respectively. The **`totalAvail`** and **`bufCycles`** parameters indicate the number of available scans and the number of buffer processing cycles that have occurred. The **`bufPosition`** parameter indicates the current position in the transfer buffer.

daqAdcArm

DLL Function	`daqAdcArm(DaqHandleT handle);`	
C	`daqAdcArm(DaqHandleT handle);`	
Visual BASIC	`VBdaqAdcArm&(ByVal handle&)`	
Delphi	`daqAdcArm(handle:DaqHandleT)`	
Parameters	`handle`	Handle to the device to which configured ADC acquisition is to be armed
Returns	`DerrNoError`	No error
See Also	`daqAdcDisarm`	
Program References	`ADCEX1.C, DAQEX.FRM (VB), DAQEX.PAS (Delphi)`	
Used With	All devices	

Description

`daqAdcArm` allows you to arm an ADC acquisition by enabling the currently defined ADC configuration. ADC acquisition will occur when the trigger event (as specified by **`daqAdcSetTrig`**) is satisfied. All ADC acquisition configuration information must be specified prior to the **`daqAdcArm`** command. For a previously configured acquisition, the **`daqAdcArm`** command will use the specified parameters. If no previous configuration was given, or it is desirable to change any or all acquisition parameters, then those commands relating to the desired ADC acquisition configuration must be issued prior to calling **`daqAdcArm`**. Personal Daq will perform calibration upon execution of the **`daqAdcArm`** command.

daqAdcDisarm

DLL Function	**daqAdcDisarm(DaqHandleT handle);**
C	**daqAdcDisarm(DaqHandleT handle);**
Visual BASIC	**VBdaqAdcDisarm&(ByVal handle&)**
Delphi	**daqAdcDisarm(handle:DaqHandleT)**
Parameters	**handle** handle to the device to disable ADC acquisitions
Returns	**DerrNoError** - No error
See Also	**daqAdcArm**
Program References	None
Used With	All devices

Description

daqAdcDisarm allows you to disarm an acquisition if one is currently active.

- If the specified trigger event has not yet occurred, the trigger event will be disabled and no acquisition will be performed.
- If the trigger event has occurred, both the acquisition and the transfer of data come to a halt.

daqAdcGetFreq

DLL Function	**daqAdcGetFreq(DaqHandleT handle, PFLOAT freq);**
C	**daqAdcGetFreq(DaqHandleT handle, PFLOAT freq);**
Visual BASIC	**VbdaqAdcGetFreq&(ByVal handle&, freq!)**
Delphi	**daqAdcGetFreq(handle:DaqHandleT; var freq:single)**
Parameters	**handle** Handle to the device for which to get the current frequency setting
	freq A variable to hold the currently defined sampling frequency in Hz Valid values: **100000.0 - 0.0002**
Returns	**DerrNoError** - No errors
See Also	**daqAdcSetFreq, daqAdcSetClock**
Program References	None
Used With	All devices

Description

daqAdcGetFreq reads the sampling frequency of the pacer clock.

daqAdcGetScan

DLL Function	**daqAdcGetScan(DaqHandleT handle, PDWORD channels, daqAdcGain *gains, PDWORD flags, PDWORD chanCount);**
C	**daqAdcGetScan(DaqHandleT handle, PDWORD channels, DaqAdcGain *gains, PDWORD flags, PDWORD chanCount);**
Visual BASIC	**VbdaqAdcGetScan&(ByVal handle&, channels&(), gains&(), flags&(), chanCount&)**
Delphi	**daqAdcGetScan(handle:DaqHandleT; channels:PDWORD; gains:daqAdcGainP; flags:PDWORD; chanCount:PDWORD)**
Parameters	**handle** Handle to the device for which to get the current scan configuration.
	channels An array to hold up to 512 channel numbers or 0 if the channel information is not desired.
	***gains** An array to hold up to 512 gain values or 0 if the channel gain information is not desired
	flags Channel configuration flags in the in the form of a bit mask
	chanCount A variable to hold the number of values returned in the chans and gains arrays
Returns	**DerrNoError** No error
See Also	**daqAdcSetScan, daqAdcSetMux**
Program References	None
Used With	All devices

Description

daqAdcGetScan reads the current scan group consisting of all channels currently configured. The returned parameter settings directly correspond to those set using the daqAdcSetScan function. For further description of these parameters, refer to **daqAdcSetScan**. See *ADC Flags Definition* table for channel flag definitions.

daqAdcRd

DLL Function	**daqAdcRd(DaqHandleT handle, DWORD chan, PVOID sample, daqAdcGain gain, DWORD flags);**	
C	**daqAdcRd(DaqHandleT handle, DWORD chan, PVOID sample, DaqAdcGain gain, DWORD flags);**	
Visual BASIC	**VBdaqAdcRdSingle&(ByVal handle&, ByVal chan&, sample!, ByVal gain&, ByVal flags&)**	
Delphi	**daqAdcRd(handle:DaqHandleT; chan:DWORD; sample:pointer; const gain:daqAdcGain; flags:DWORD)**	
Parameters	**handle**	Handle to the device for which the ADC reading is to be acquired
	chan	A single channel number
	sample	A pointer to a value where an A/D sample is stored. Valid values: (See **daqAdcSetTag**)
	gain	The channel gain
	flags	Channel configuration flags in the form of a bit mask
Returns	**DerrFIFOFull**	Buffer Overrun
	DerrInvGain	Invalid gain
	DerrInvChan	Invalid channel
	DerrNoError	No Error
See Also	**daqAdcSetMux, daqAdcSetTrig, daqAdcSoftTrig**	
Program References	None	
Used With	All devices	

Description

daqAdcRd is used to take a single reading from the given local A/D channel. This function will use a software trigger to immediately trigger and acquire one sample from the specified A/D channel.

- The chan parameter indicates the channel for which to take the sample.
- The sample parameter is a pointer to where the collected sample should be stored.
- The gain parameter indicates the channel's gain setting.
- The flags parameter allows the setting of channel-dependent options. See *ADC Flags Definition* table for channel flags definitions.

daqAdcRdN

DLL Function	**daqAdcRdN(DaqHandleT handle, DWORD chan, PVOID buf, DWORD scanCount, daqAdcTriggerSource triggerSource, BOOL rising, FLOAT level, FLOAT freq, daqAdcGain gain, DWORD flags);**	
C	**daqAdcRdN(DaqHandleT handle, DWORD chan, PVOID buf, DWORD scanCount, DaqAdcTriggerSource triggerSource, BOOL rising, FLOAT level, FLOAT freq, DaqAdcGain gain, DWORD flags);**	
Visual BASIC	**VBdaqAdcRdNSingle&(ByVal handle&, ByVal chan&, buf!(), ByVal ScanCount&, ByVal triggerSource&, ByVal rising&, ByVal level!, ByVal Freq!, ByVal gain&, ByVal flags&)"**	
Delphi	**daqAdcRdN(handle:DaqHandleT; chan:DWORD; buf:pointer; scanCount:DWORD; triggerSource:daqAdcTriggerSource; rising:longbool; level:single; freq:single; const gain:daqAdcGain; flags:DWORD)**	
Parameters	**Handle**	Handle to the device for which the ADC channel samples are to be acquired
	Chan	A single channel number
	Buf	An array where the A/D scans will be returned
	ScanCount	The number of scans to be taken; Valid values: **1 - 32767**
	TriggerSource	The trigger source
	Rising	Boolean flag to indicate the rising or falling edge for the trigger source
	Level	The trigger level if an analog trigger is specified; Valid values: **0 -4095**
	Freq	The sampling frequency in Hz (**100000.0 to 0.0002**)
	Gain	The channel gain
	Flags	Channel configuration flags in the form of a bit mask
Returns	**DerrFIFOFull**	Buffer overrun
	DerrInvGain	Invalid gain
	DerrIncChan	Invalid channel
	DerrInvTrigSource	Invalid trigger
	DerrInvLevel	Invalid level
See Also	**daqAdcSetFreq, daqAdcSetMux, daqAdcSetClock, daqAdcSetTrig**	
Program References	None	
Used With	All devices	

Description

daqAdcRdN is used to take multiple scans from a single A/D channel. This function will:

- Configure the pacer clock
- Configure all channels with the specified gain parameter
- Configure all channel options with the channel flags specified
- Arm the trigger
- Acquire count scans from the specified A/D channel
- See *ADC Flags Definition* table (in *ADC Miscellaneous Definitions*) for channel flags parameter definition.

daqAdcRdScan

DLL Function	**daqAdcRdScan(DaqHandleT handle, DWORD startChan, DWORD endChan, PVOID buf, daqAdcGain gain, DWORD flags);**	
C	**daqAdcRdScan(DaqHandleT handle, DWORD startChan, DWORD endChan, PVOID buf, DaqAdcGain gain, DWORD flags);**	
Visual BASIC	**VBdaqAdcRdScanSingle&(ByVal handle&, ByVal startChan&, ByVal endChan&, buf!(), ByVal gain&, ByVal flags&)**	
Delphi	**daqAdcRdScan(handle:DaqHandleT; startChan:DWORD; endChan:DWORD; buf:pointer; const gain:daqAdcGain; flags:DWORD);**	
Parameters	**Handle**	Handle to the device from which the ADC scan is to be acquired
	StartChan	The starting channel of the scan group
	EndChan	The ending channel of the scan group
	Buf	An array where the A/D scans will be placed
	Gain	The channel gain
	Flags	Channel configuration flags in the form of a bit mask.
Returns	**DerrInvGain**	Invalid gain
	DerrInvChan	Invalid channel
	DerrNoError	No error
See Also	**daqAdcRdScanN, daqAdcSetMux, daqAdcSetClock, daqAdcSetTrig**	
Program References	None	
Used With	All devices	

Description

daqAdcRdScan reads a single sample from multiple channels. This function will use a software trigger to immediately trigger and acquire one scan consisting of each channel, starting with **startChan** and ending with **endChan**. The **gain** setting will be applied to all channels. See *ADC Flags Definition* table for channel **flags** definitions.

daqAdcRdScanN

DLL Function	**daqAdcRdScanN(DaqHandleT handle, DWORD startChan, DWORD endChan, PVOID buf, DWORD scanCount, daqAdcTriggerSource triggerSource, BOOL rising, FLOAT level, FLOAT freq, daqAdcGain gain, DWORD flags);**	
C	**daqAdcRdScanN(DaqHandleT handle, DWORD startChan, DWORD endChan, PVOID buf, DWORD scanCount, DaqAdcTriggerSource triggerSource, BOOL rising, FLOAT level, FLOAT freq, DaqAdcGain gain, DWORD flags);**	
Visual BASIC	**VBdaqAdcRdScanNSingle&(ByVal handle&, ByVal startChan&, ByVal endChan&, buf!(), ByVal ScanCount&, ByVal triggerSource&, ByVal rising&, ByVal level!, ByVal Freq!, ByVal gain&, ByVal flags&)**	
Delphi	**daqAdcRdScanN(handle:DaqHandleT; startChan:DWORD; endChan:DWORD; buf:pointer; scanCount:DWORD; triggerSource:daqAdcTriggerSource; rising:longbool; level:single; freq:single; const gain:daqAdcGain; flags:DWORD)**	
Parameters	**handle**	Handle to the device from which ADC scans are to be acquired
	startchan	The starting channel of the scan group (see table at end of appendix)
	endchan	The ending channel of the scan group (see table at end of appendix)
	buf	An array where the A/D scans will be placed
	scanCount	The number of scans to be read ; Valid values: **1 - 65536**
	triggerSource	The trigger source (see table at end of appendix)
	rising	Boolean flag to indicate the rising or falling edge for the trigger source
	level	The trigger level if an analog trigger is specified; Valid values: **0 -4095**
	freq	The sampling frequency in Hz; Valid values: **100000.0 - 0.0002**
	gain	The channel gain (See tables at end of appendix).
	flags	Channel configuration flags in the form of a bit mask.
Returns	**DerrInvGain**	Invalid gain
	DerrInvChan	Invalid channel
	DerrInvTrigSource	Invalid trigger
	DerrInvLevel	Invalid Level
	DerrFIFOFull	Buffer Overrun
	DerrNoError	No error
See Also	**daqAdcRd, daqAdcRdN, daqAdcRdScan, daqAdcSetClock, daqAdcSetTrig**	
Program References	None	
Used With	All devices	

Description

daqAdcRdScanN reads multiple scans from multiple A/D channels. This function will configure the pacer clock, arm the trigger and acquire count scans consisting of each channel, starting with **startChan** and ending with **endChan**. The **gain** setting will be applied to all channels. The **freq** parameter is used to set the acquisition frequency. See *ADC Flags Definition* table for channel **flags** parameter definition.

daqAdcSetAcq

DLL Function	**daqAdcSetAcq(DaqHandleT handle, daqAdcAcqMode mode, DWORD preTrigCount, DWORD postTrigCount);**	
C	**daqAdcSetAcq(DaqHandleT handle, DaqAdcAcqMode mode, DWORD preTrigCount, DWORD postTrigCount);**	
Visual BASIC	**VBdaqAdcSetAcq&(ByVal handle&, ByVal mode&, ByVal preTrigCount&, ByVal postTrigCount&)**	
Delphi	**daqAdcSetAcq(handle:DaqHandleT; mode:daqAdcAcqMode; preTrigCount:DWORD; postTrigCount:DWORD)**	
Parameters	**handle**	Handle to the device for which the ADC acquisition is to be configured
	mode	Selects the mode of the acquisition
	PreTrigCount	Number of pre-trigger ADC scans to be collected
	PostTrigCount	Number of post-trigger ADC scans to be collected
Returns	**DerrNoError**	No error
See Also	**daqAdcArm, daqAdcDisarm, daqAdcSetTrig**	
Program References	**ADCEX1.C, DACEX1.C, DYN32ENH.C, DAQEX.FRM (VB), ADCEX.PAS (Delphi)**	
Used With	All devices	

Description

daqAdcSetAcq allows you to characterize the acquisition mode and the pre- and post-trigger durations. The **mode** parameter describes the style of data collection. The **preTrigCount** and **postTrigCount** parameters specify the respective durations, or lengths, of the pre-trigger and post-trigger acquisition states.

Acquisition modes can be defined as follows:

- **DaamNShot** - Once triggered, continue acquisition until the specified post-trigger count has been satisfied. Once the post-trigger count has been satisfied, the acquisition will be automatically disarmed.
- **DaamInfinitePost** - Once triggered, continue the acquisition indefinitely until the acquisition is disabled by the daqAdcDisarm function.
- **DaamPrePost** - Begin collecting the specified number of pre-trigger scans immediately upon issuance of the daqAdcArm function. The trigger will not be enabled until the specified number of pre-trigger scans have been collected. Once triggered, the acquisition will then continue collecting post-trigger data until the post-trigger count has been satisfied. Once the post-trigger count has been satisfied, the acquisition will be automatically disarmed.

daqAdcSetDataFormat

DLL Function	**daqAdcSetDataFormat(DaqHandleT handle, daqAdcRawDataFormatT rawFormat, daqAdcPostProcDataFormatT postProcFormat);**	
C	**daqAdcSetDataFormat(DaqHandleT handle, DaqAdcRawDataFormatT rawFormat, DaqAdcPostProcDataFormatT postProcFormat);**	
Visual BASIC	**VBdaqAdcSetDataFormat&(ByVal handle&, ByVal rawFormat&, ByVal postProcFormat&)**	
Delphi	**daqAdcSetDataFormat(Handle:DaqHandleT; rawFormat:daqAdcRawDataFormatT rawFormat; postProcFormat:daqAdcPostProcDataFormatT);**	
Parameters	**Handle**	The handle to the device for which to set the option
	RawFormat	Specifies the raw data format
	PostProcFormat	Flags specifying the options to use
Returns	**DerrNoError**	No error
See Also	**daqCvtRawDataFormat,daqCvtRawDataFormat**	
Program References	None	
Used With	All devices	

Description

daqAdcSetDataFormat allows the setting of the raw and the post-acquisition data formats which will be returned by the acquisition transfer functions.

Note: Certain devices may be limited to the types of raw and post-acquisition data formats that can be presented.

The **rawFormat** parameter indicates how the raw data format is to be presented. Normally, the raw-data format represents the data from the A/D converter. The default value for this parameter is **DardfNative** where the raw-data format follows the native-data format of the A/D for the particular device.

The Personal Daq requires that the **rawFormat** parameter be set to **DardfFloat**. This format indicates that the driver should return the raw data in floating point format. In this case the floating point data returned will be indicative of the type of channel configured. For instance, if the channel is configured as a volts/mvolts/uvolts channel then the floating point value returned for the channel will be in volts. If the channel is configured as temperature (thermocouple) then the floating point value returned for that channels will be in degrees C.

The **postProcFormat** parameter specifies the format for which post-acquisition data will be presented. This format is used by the one-step functions of the form **daqAdcRd...**. The default value is **DappdfRaw** where the post-acquisition data format will follow the **rawFormat** parameter.

daqAdcSetDiskFile

DLL Function	**daqAdcSetDiskFile(DaqHandleT handle, LPSTR filename, daqAdcOpenMode openMode, DWORD preWrite);**	
C	**daqAdcSetDiskFile(DaqHandleT handle, LPSTR filename, DaqAdcOpenMode openMode, DWORD preWrite);**	
Visual BASIC	**VBdaqAdcSetDiskFile&(ByVal handle&, ByVal filename$, ByVal openMode&, ByVal preWrite&)**	
Delphi	**daqAdcSetDiskFile(handle:DaqHandleT; filename:PChar; openMode:daqAdcOpenMode; preWrite:DWORD)**	
Parameters	**handle**	Handle to the device for which direct to disk ADC acquisition is to be performed.
	filename	String representing the path and name of the file to place the raw ADC acquisition data.
	openMode	Specifies how to open the file for writing
	preWrite	Specifies the number of bytes to pre-write in the file
Returns	**DerrNoError**	No error
See Also	**daqAdcTransferGetStat, daqAdcTransferSetBuffer, daqAdcTransferStart, daqAdcTransferStop**	
Program References	None	
Used With	All devices	

Description

daqAdcSetDiskFile allows you to set a destination file for data transfers. Data transfers will be directed to the specified disk file. The **filename** parameter is a string representing the path\name of the file to be opened. The **openMode** parameter indicates how the file is to be opened for writing data. Valid file open modes are defined as follows:

- **DaomAppendFile** - Open an existing file to append subsequent data transfers. This mode should only be used when the existing file has a similar channel scan group configuration as the subsequent transfers.
- **DaomWriteFile** - Rewrite or write over an existing file. This operation will destroy the original contents of the file.
- **DaomCreateFile**- Create a new file for subsequent data transfers. This mode does not require that the file exist beforehand.

The **preWrite** parameter may, optionally, be used to specify the amount that the file is to be pre-written before the actual data collection begins. Specifying the pre-write amount may increase the data-to-disk performance of the acquisition if it is known beforehand how much data will be collected. If no pre-write is to be done, then the **preWrite** parameter should be set to 0.

daqAdcSetFilter

DLL Function	**daqAdcSetFilter(DaqHandleT handle, DaqAdcFilterType filterType, DWORD filterWindow);**	
C	**daqAdcSetFilter(DaqHandleT handle, DaqAdcFilterType filterType, DWORD filterWindow);**	
Visual BASIC	**VbdaqAdcSetFilter&(ByVal handle&, ByVal filterType&, ByVal filterWindow&)**	
Delphi	**DaqAdcGetScan(handle:DaqHandleT; filterType:DaqAdcFilterType;filterWindow:DWORD)**	
Parameters	**handle**	Handle to the device for which to get the current scan configuration.
	filterType	Specifies the type of analog filtering to be performed (currently DaftSWAvg)
	filterWindow	Specifies the number of scans over which to filter
Returns	**DerrNoError** - No error (also, refer to *API Error Codes* on page page B-33)	
See Also		
Program References	None	
Used With	PersonalDaq	

Description

daqAdcSetFilter sets the driver to perform filtering for analog channels. The **filterType** parameter specifies the filter method to be used. Once this function is set, filtering will be performed on all analog channels in the scan group. Note that the **filterWindow** parameter specifies the number of scans over which the filtering method is to be applied.

daqAdcSetFreq

DLL Function	**daqAdcSetFreq(DaqHandleT handle, FLOAT freq);**	
C	**daqAdcSetFreq(DaqHandleT handle, FLOAT freq);**	
Visual BASIC	**VBdaqAdcSetFreq&(ByVal handle&, ByVal freq!)**	
Delphi	**daqAdcSetFreq(handle:DaqHandleT; freq:single)**	
Parameters	**Handle**	Handle to the device for which the ADC acquisition frequency is to be set.
	Freq	The sampling frequency in Hz; Valid values: **100000.0 - 0.0002**
Returns	**DerrNoError**	No error
See Also	**daqAdcGetFreq, daqAdcSetRate**	
Program References	None	
Used With	All devices	

Description

daqAdcSetFreq calculates and sets the frequency of the acquisition clock using the frequency specified in Hz. The frequency specified is the rate at which channel scans are collected. This rate represents the interval at which each channel scan group is collected. The rate specified must be less than or equal to the maximum frequency for which the currently defined channel scan configuration can run. The maximum rate can be retrieved via the **daqAdcGetFreq** or **daqAdcSetRate** commands. If a rate is specified which is greater than the maximum frequency at which the current channel configuration will allow the driver will automatically set the rate to the maximum allowable.

daqAdcSetMux

DLL Function	**daqAdcSetMux(DaqHandleT handle, DWORD startChan, DWORD endChan, daqAdcGain gain, DWORD flags);**	
C	**daqAdcSetMux(DaqHandleT handle, DWORD startChan, DWORD endChan, DaqAdcGain gain, DWORD flags);**	
Visual BASIC	**VbdaqAdcSetMux&(ByVal handle&, ByVal startChan&, ByVal endChan&, ByVal gain&, ByVal flags&)**	
Delphi	**daqAdcSetMux(handle:DaqHandleT; startChan:DWORD; endChan:DWORD; const gain:daqAdcGain; flags:DWORD)**	
Parameters	**Handle**	Handle to the device for which to configure the ADC channel scan group
	StartChan	The starting channel of the scan group
	EndChan	The ending channel of the scan group
	Gain	The gain value for all channels
	Flags	Channel configuration flags in the form of a bit mask
Returns	**DerrInvGain**	Invalid gain
	DerrIncChan	Invalid channel
	DerrNoError	No error
See Also	**daqAdcSetScan, daqAdcGetScan**	
Program References	None	
Used With	All devices	

Description

daqAdcSetMux sets a simple scan sequence of local A/D channels from **startChan** to **endChan** with the specified **gain** value. This command provides a simple alternative to **daqAdcSetScan** if only consecutive channels need to be acquired. The **flags** parameter is used to set channel dependent options. See *ADC Flags Definition* table for channel **flags** definitions.

daqAdcSetRate

DLL Function	**daqAdcSetRate(DaqHandleT handle, daqAdcRateMode mode, daqAdcAcqState acqState, FLOAT reqRate, PFLOAT actualRate);**	
C	**daqAdcSetRate(DaqHandleT handle, DaqAdcRateMode mode, DaqAdcAcqState acqState, FLOAT reqRate, PFLOAT actualRate);**	
Visual BASIC	**VBdaqAdcSetRate(ByVal handle&, ByVal mode&, ByVal acqState&, ByVal reqRate!, actualRate!);**	
Delphi	**daqAdcSetRate(handle:DaqHandleT; mode:daqAdcRateMode; state:daqAdcAcqState; reqValue:single; actualValue:PSINGLE);**	
Parameters	**handle**	Handle to the device for which to set ADC scanning frequency.
	mode	Specifies the rate mode (frequency or period).
	acqState	Specifies the acquisition state to which the rate is to be applied.
	reqRate	Specifies the requested rate.
	actualRate	Returns the actual rate applied. This may be different from the requested rate.
Returns	**DerrNoError**	No error
See Also	**daqAdcSetAcq, daqAdcSetTrig, daqAdcArm, daqAdcSetFreq, daqAdcGetFreq**	
Program References	**DAQEX.PAS (Delphi), ADCEX1.C, FREQEX1.C, PULSEEX1.C, MULTEX1.C, DAQEX.FRM (VB)**	
Used With	All devices	

Description

daqAdcSetRate configures the scan rate using the rate mode specified by the **mode** parameter. Currently, the valid modes are:

- **DarmPeriod** - Defines the requested rate to be in periods/sec.
- **DarmFrequency** - Defines the requested rate to be a frequency. This function will set the acquisition rate requested by the reqRate parameter for the acquisition state specified by the **acqState** parameter.

Currently, the following acquisition states are valid:

- **DaasPreTrig** - Sets the pre-trigger acquisition rate to the requested rate.
- **DaasPostTrig** - Sets the post-trigger acquisition rate to the requested rate.

Personal Daq does not allow a pre-trigger rate different from the post-trigger rate. Therefore, the pre-trigger rate settings will be ignored and the post-trigger rate will be used for both the pre-trigger and the post-trigger scan rates.

If the requested rate is unattainable on the specified device, a rate will be automatically adjusted to the device's closest attainable rate. If this occurs, the **actualRate** parameter will return the actual rate for which the device has been programmed.

daqAdcSetScan

DLL Function	**daqAdcSetScan(DaqHandleT handle, PDWORD channels, daqAdcGain *gains, PDWORD flags, DWORD chanCount);**	
C	**daqAdcSetScan(DaqHandleT handle, PDWORD channels, DaqAdcGain *gains, PDWORD flags, DWORD chanCount);**	
Visual BASIC	**VBdaqAdcSetScan&(ByVal handle&, channels&(), gains&(), flags&(), ByVal chanCount&)**	
Delphi	**daqAdcSetScan(handle:DaqHandleT; channels:PDWORD; gains:daqAdcGainP; flags:PDWORD; chanCount:DWORD)**	
Parameters	**Handle**	Handle to the device for which ADC scan group is to be configured
	Channels	An array of up to 512 channel numbers
	***gains**	An array of up to 512 gain values
	Flags	Channel configuration flags in the form of a bit mask
	ChanCount	The number of values in the chans and gains arrays Valid values: **1 -512**
Returns	**DerrNotCapable**	No high speed digital
	DerrInvGain	Invalid gain
	DerrInvChan	Invalid channel
	DerrNoError	No error
See Also	**daqAdcGetScan, daqAdcSetMux**	
Program References	**ADCEX1.C, FREQEX1.C, PULSEEX1.C, MULTEX1.C, DAQEX.FRM (VB), DAQEX.PAS (Delphi)**	
Used With	All devices	

Description

daqAdcSetScan configures a scan group consisting of multiple channels. As many as 8192 channel entries can be made in the scan group configuration. Any analog, frequency, counter or digital input channel can be included in the scan group configuration at any valid gain setting. Scan group configuration may be composed of local or expansion module channels.

The **channels** parameter is a pointer to an array of up to 8192 channel/device values. Each entry represents a channel/device number in the scan group configuration. Channel/device combinations can be entered multiple times at the same or different gain setting.

The **gains** parameter is a pointer to an array of up to 8192 gain settings. Each gain entry represents the gain to be used with the corresponding analog channel entry. Gain entry can be any valid gain setting for the corresponding channel.

The **flags** parameter is a pointer to an array of up to 8192 channel flag settings. Each flag entry represents a 4-byte-wide bit map of channel configuration settings for the corresponding channel entry. The channel flags can be used to set channel specific configuration settings. See the *Channel Configuration Flags Definition* table for valid channel flag values.

The **chanCount** parameter represents the total number of channels in the scan group configuration. This number also represents the number of entries in each of the **channels**, **gains** and **flags** arrays.

daqAdcSetTrigEnhanced

DLL Function	**daqAdcSetTrigEnhanced(DaqHandleT handle, daqAdcTriggerSource *trigSources, daqAdcGain *gains, daqAdcRangeT *adcRanges, daqEnhTrigSensT *trigSensitivity, PFLOAT level, PFLOAT hysteresis, PDWORD channels, DWORD chanCount, char *opStr);**	
C	**daqAdcSetTrigEnhanced(DaqHandleT handle, DaqAdcTriggerSource *trigSources, DaqAdcGain *gains, DaqAdcRangeT *adcRanges, DaqEnhTrigSensT *trigSensitivity, PFLOAT level, PFLOAT hysteresis, PDWORD channels, DWORD chanCount, char *opStr);**	
Visual BASIC	**VBdaqAdcSetTrigEnhanced&(ByVal handle&, ByRef triggerSources&(), ByRef gains&(), ByRef adcRanges&(), ByRef trigSense&(), ByRef levels!(), ByRef hysteresis!(), ByRef chan&(), ChanCount&, opstr$)**	
Delphi	**daqAdcSetTrigEnhanced(handle:DaqHandleT; trigSource:daqAdcTriggerSourceP; gains:daqAdcGainP; adcRanges:daqAdcRangeTP; trigSensitivity:daqEnhTrigSensTP; level:PSINGLE; hysteresis:PSINGLE; channel:PDWORD; chanCount:DWORD; opStr:PCHAR)**	
Parameters	**handle**	Handle to the device for which the ADC acquisition trigger is to be configured.
	triggerSource	A pointer to an array of trigger sources for each defined trigger channel.
	gains	A pointer to an array of gains for each defined A/D trigger channel.
	levels	A pointer to an array of A/D analog trigger levels for each defined A/D trigger channel.
	hysteresis	A pointer to an array of hysteresis values for each defined A/D trigger channel.
	trigSense	A pointer to an array of trigger sensitivity flags for each defined A/D channel trigger source.
	adcRanges	A pointer to an array of polarity flag definitions for each defined A/D channel.
	channels	A pointer to an array of trigger channels representing the actual A/D trigger channels to trigger on.
	chanCount	Indicates the number of configured A/D channels.
Returns	**DerrNoError**	No error
See Also	**daqAdcSetAcq, daqAdcSetTrig, daqAdcSetScan**	
Program References	**ADCEX1.C, FREQEX1.C, PULSEEX1.C, MULTEX1.C, DAQEX.FRM (VB), DAQEX.PAS (Delphi)**	
Used With	Personal Daq/55 and Personal Daq/56	

Description

daqAdcSetTrigEnhanced configures the device for triggering. Trigger configuration allows the device to be configured to detect A/D triggering formed with multiple A/D channel trigger-event conditions. The trigger event may be defined as a combination of multiple A/D analog-level event conditions that are logically **and**'d or **or**'d.

The trigger event is formulated based on the channel trigger event for each channel in the trigger sequence. The total number of trigger channels is defined by the **chanCount** parameter. Each channel trigger configuration parameter definition is a pointer to an array of **chanCount** length and is defined as follows:

- **channels** - Defines a pointer to an array of actual device/channel numbers for which to configure the corresponding trigger events.
- **triggerSources** - Defines a pointer to an array of trigger sources for which to configure the corresponding trigger events for the corresponding channel in the channels array. See the *ADC Trigger Source Definitions* table for valid triggers.
- **gains** - Defines a pointer to an array of gains corresponding to the actual A/D channels in the corresponding A/D channel number in the channels array.
- **adcRanges** - Defines a pointer to an array of A/D ranges for the A/D channels defined in the corresponding channels array.

  Note: Personal Daq ignores **adcRanges** parameters.
- **hysteresis** - Defines a pointer to an array of hysteresis values for each corresponding A/D channel defined in the channels array.
- **levels** - Defines a pointer to an array of levels for which, when satisfied, will set the trigger event for the corresponding channel defined in the channels array.
- **opStr** - Defines a string that defines the logical relationship between the individual channel trigger events and the global A/D trigger condition. Currently, the string can be defined as "*" to perform an **and** operation or "+" to perform an **or** operation on the individual channel trigger events to formulate the global A/D trigger condition.

  Note: Personal Daq ignores **opStr** parameters.
- **trigSense** - Defines an array of trigger sensitivity definitions for satisfying the defined trigger event for the corresponding channel defined in the channels array. Currently, the valid trigger sensitivity values are as follows:

DetsRisingEdge	Trigger the channel on the rising edge of the signal at the specified level.
DetsFallingEdge	Trigger the channel on the falling edge of the signal at the specified level.

daqAdcSoftTrig

DLL Function	**daqAdcSoftTrig(DaqHandleT handle);**
C	**daqAdcSoftTrig(DaqHandleT handle);**
Visual BASIC	**VBdaqAdcSoftTrig&(ByVal handle&)**
Delphi	**daqAdcSoftTrig(handle:DaqHandleT)**
Parameters	**Handle** Handle to the device to which the ADC software trigger is to be applied
Returns	**DerrNoError** No error
See Also	**daqAdcSetTrig, daqAdcSetAcq**
Program References	None
Used With	All devices

Description

daqAdcSoftTrig is used to send a software trigger command to the Personal Daq device. This software trigger can be used to initiate a scan or an acquisition from a program after configuring the software trigger as the trigger source. This function may only be used if the trigger source for the acquisition has been set to **DatsSoftware** with the **daqAdcSetTrig** function.

daqAdcTransferBufData

DLL Function	**daqAdcTransferBufData(DaqHandleT handle, PVOID buf, DWORD scanCount, daqAdcBufferXferMask bufMask, PDWORD retCount);**	
C	**daqAdcTransferBufData(DaqHandleT handle, PVOID buf, DWORD scanCount, DaqAdcBufferXferMask bufMask, PDWORD retCount);**	
Visual BASIC	**VBdaqAdcTransferBufDataSingle&(ByVal handle&, buf!(), ByVal ScanCount&, ByVal transferMask&, retCount&)**	
Delphi	**daqAdcTransferBufData(handle: DaqHandleT; buf:pointer, scanCount:DWORD, bufMask:DaqAdcBufferXferMask; retCount:DWORD);**	
Parameters	**handle**	Handle to the device for which the ADC buffer should be retrieved.
	buf	Pointer to an application-supplied buffer to place the buffered data.
	scanCount	Number of scans to retrieve from the acquisition buffer.
	bufMask	A mask defining operation depending on the current state of the acquisition buffer
	retCount	A pointer to the total number of scans returned, if any.
Returns	**DerrNoError**	No error
See Also	**daqAdcTransferSetBuffer, daqAdcTransferGetStat**	
Program References		
Used With	All devices	

Description

daqAdcTransferBufData requests a transfer of **scanCount** scans from the driver-allocated acquisition buffer to the specified user-supplied buffer. The **bufMask** parameter can be used to specify the conditions for the transfer as follows:

- **DabtmWait** - Instructs the function to wait until the requested number of scans are available in the driver-allocated acquisition buffer. When the requested number of scans are available, the function will return with **retCount** set to **scanCount**, the number of scans requested. Scan data will be returned in the memory referred to by the buf parameter.
- **DabtmNoWait** - Instructs the function to return immediately if the specified number of scans are not available when the function is called. If the entire amount requested is not available, the function will return with no data and **retCount** will be set to 0. If the requested number of scans are available in the acquisition buffer, the function will return with **retCount** set to **scanCount**, the number of scans requested. Scan data will be returned in the memory referred to by the buf parameter.
- **DabtmRetAvail** - Instructs the function to return immediately, regardless of the number of scans available in the driver-allocated acquisition buffer. The **retCount** parameter will return the total number of scans retrieved. **retCount** can return anything from 0 to **scanCount**, the number of scans requested. Scan data will be returned in the memory referred to by the **buf** parameter.

  The driver-allocated acquisition buffer must have been allocated prior to calling this function. This is performed via the **daqAdcTransferSetBuffer**. Refer to **daqAdcTransferSetBuffer** for more details on specifying the driver-allocated acquisition buffer.

daqAdcTransferGetStat

DLL Function	**daqAdcTransferGetStat(DaqHandleT handle, PDWORD active, PDWORD retCount);**	
C	**daqAdcTransferGetStat(DaqHandleT handle, PDWORD active, PDWORD retCount);**	
Visual BASIC	**VBdaqAdcTransferGetStat&(ByVal handle&, active&, retCount&)**	
Delphi	**daqAdcTransferGetStat(handle:DaqHandleT; var active:DWORD; var retCount:DWORD)**	
Parameters	**handle**	Handle to the device for which ADC transfer status is to be retrieved
	active	A pointer to the transfer-state flags in the form of a bit mask
	retCount	A pointer to the total number of ADC scans acquired (or available) in the current transfer
Returns	**DerrNoError**	No error
See Also	**daqAdcTransferSetBuffer, daqAdcTransferStart, daqAdcTransferStop**	
Program References	**ADCEX1.C, FREQEX1.C, PULSEEX1.C, MULTEX1.C, DAQEX.FRM (VB), DAQEX.PAS (Delphi)**	
Used With	All devices	

Description

daqAdcTransferGetStat allows you to retrieve the current state of an acquisition transfer.

The **active** parameter will indicate the current state of the transfer in the form of a bit mask. Refer to the *ADC Acquisition/Transfer Active Flag Definitions* (in the *ADC Miscellaneous Definitions* table) for valid bit-mask states.

The **retCount** parameter will return the total number of scans acquired in the current transfer if the transfer is in user-allocated buffer mode or will return the total number of unread scans in the acquisition buffer if the transfer is in driver-allocated buffer mode. Refer to the **daqAdcTransferSetBuffer** function for more information on buffer allocation modes.

The transfer state and return count values will continue to be updated until any of the following occurs:

- the transfer count is satisfied
- the transfer is stopped (**daqAdcStopTransfer**)
- the acquisition is disarmed (**daqDisarm**)

daqAdcTransferSetBuffer

DLL Function	**daqAdcTransferSetBuffer(DaqHandleT handle, PVOID buf, DWORD scanCount, DWORD transferMask);**	
C	**daqAdcTransferSetBuffer(DaqHandleT handle, PVOID buf, DWORD scanCount, DWORD transferMask);**	
Visual BASIC	**VBdaqAdcTransferSetBufferSingle&(ByVal handle&, buf!(), ByVal ScanCount&, ByVal transferMask&)**	
Delphi	**daqAdcTransferSetBuffer(handle:DaqHandleT; buf:pointer; scanCount:DWORD; transferMask:DWORD)**	
Parameters	**handle**	Handle to the device for which an ADC transfer is to be performed.
	buf	Pointer to the buffer for which the acquired data is to be placed.
	scanCount	The total length of the buffer (in scans).
	transferMask	Configures the buffer transfer mode.
Returns	**DerrNoError**	No error
See Also	**daqAdcTransferStart, daqAdcTransferStop, daqAdcTransferGetStat, daqAdcSetAcq, daqAdcTransferBufData**	
Program References	**ADCEX1.C, FREQEX1.C, PULSEEX1.C, MULTEX1.C, DAQEX.FRM (VB), DAQEX.PAS (Delphi)**	
Used With	All devices	

Description

daqAdcTransferSetBuffer allows you to configure transfer buffers for ADC data acquisition. This function can be used to configure the specified user- or driver-allocated buffers for subsequent ADC transfers.

If a user-allocated buffer is to be used, two conditions apply:

The buffer specified by the **buf** parameter must have been allocated by the user prior to calling this function.

The allocated buffer must be large enough to hold the number of ADC scans as determined by the current ADC scan group configuration.

The **scanCount** parameter is the total length of the transfer buffer in scans. The scan size is determined by the current scan group configuration. Refer to the **daqAdcSetScan** and **daqAdcSetMux** functions for further information on scan group configuration.

The character of the transfer can be configured via the **transferMask** parameter. Among other things, the **transferMask** specifies the update, layout/usage, and allocation modes of the buffer. The modes can be set as follows:

- **DatmCycleOn** - Specifies the buffer to be a circular buffer in buffer-cycle mode; allows the transfer to continue when the end of the transfer buffer is reached by wrapping the transfer of scan data back to the beginning of the buffer. In this mode, the transfer buffer will continue to be wrapped until the post-trigger count has been reached (specified by **daqAdcSetAcq**) or the transfer/acquisition is halted by the application (**daqAdcTransferStop**, **daqAdcDisarm**). The default setting is **DatmCycleOff**.

- **DatmUpdateSingle** - Specifies the update mode as single sample. The update mode can be set to update for every sample or for every block of data. The update-on-single setting allows the transfer buffer to be updated for each sample collected by the Personal Daq. The default setting is **DatmUpdateSingle**.

- **DatmSavePreTrigBuf** - Specifies that the driver save the pre-trigger data once the trigger event has been satisfied. This option allows the driver to save pre-trigger data in a separate linear buffer once the trigger event has occurred. By doing so the pre-trigger data buffer can be accessed at any time after the trigger event without having the possibility of the pre-trigger data being overwritten by new data being written to the driver buffer. When the trigger event occurs the driver writes the specified amount (**daqAdcSetMode**) of pre-trigger data to an internal pre-trigger buffer which the driver allocates. The pre-trigger data may then be accessed via the **daqAdcTransferBufData** function. This option must be used in conjunction with the **DatmDriverBuf** option.

- **DatmDriverBuf** - Specifies that the driver allocate the acquisition buffer as a circular buffer whose length is determined by the **scanCount** parameter with current scan group configuration. This option allows the driver to manage the circular acquisition buffer rather than placing the burden of buffer management on the user. This option should be used with the **daqAdcTransferBufData** to access the acquisition buffer. The **daqAdcTransferStop** or the **daqAdcDisarm** function will stop the current transfer and de-allocate the driver-supplied ADC acquisition buffer. The default setting is **DatmUserBuf**. The **DatmUserBuf** option specifies a user-allocated acquisition buffer. Here, buffer management must be done in user code. This option should be used with the **daqAdcTransferStart** function to perform the data transfer operation.

daqAdcTransferStart

DLL Function	**daqAdcTransferStart(DaqHandleT handle);**
C	**daqAdcTransferStart(DaqHandleT handle);**
Visual BASIC	**VBdaqAdcTransferStart&(ByVal handle&)**
Delphi	**daqAdcTransferStart(handle:DaqHandleT)**
Parameters	**Handle** Handle to the device to initiate an ADC transfer
Returns	**DerrNoError** No error
See Also	**daqAdcTranferSetBuffer, daqAdcTransferGetStat, daqAdcTransferStop**
Program References	**ADCEX1.C, FREQEX1.C, PULSEEX1.C, MULTEX1.C, DAQEX.FRM (VB), DAQEX.PAS (Delphi)**
Used With	All devices

Description

daqAdcTransferStart allows you to initiate an acquisition transfer. The transfer will be performed under the current active acquisition. If no acquisition is currently active, the transfer will not initiate until an acquisition becomes active (via the **daqAdcArm** function). The transfer will be characterized by the current settings for the transfer buffer. The transfer buffer can be configured via the **daqAdcSetTransferBuffer** function.

daqAdcTransferStop

DLL Function	**daqAdcTransferStop(DaqHandleT handle);**
C	**daqAdcTransferStop(DaqHandleT handle);**
Visual BASIC	**VBdaqAdcTransferStop&(ByVal handle&)**
Delphi	**daqAdcTransferStop(handle:DaqHandleT)**
Parameters	**handle** Handle to the device for which the Adc data transfer is to be stopped
Returns	**DerrNoError** No error
See Also	**daqAdcTransferSetBuffer, daqAdcTransferStart, daqAdcTransferGetStat**
Program References	None
Used With	All devices

Description

daqAdcTransferStop allows you to stop a current buffer transfer, if one is active. The current transfer will be halted and no more data will transfer into the transfer buffer. Though the transfer is stopped, the acquisition will remain active. Transfers can be re-initiated with **daqAdcStartTransfer** after the stop, as long as the current acquisition remains active. The acquisition can be halted by calling the **daqAdcDisarm** function.

daqClose

DLL Function	**daqClose(DaqHandleT handle);**
C	**daqClose(DaqHandleT handle);**
Visual BASIC	**VBdaqClose&(ByVal handle&)**
Delphi	**daqClose(handle:DaqHandleT)**
Parameters	**handle** Handle to the device to be closed
Returns	**DerrNoError** No error
See Also	daqOpen
Program References	**ADCEX1.C, FREQEX1.C, PULSEEX1.C, MULTEX1.C, DIGEX1.C, INITEX1.C, DAQEX.FRM (VB), DAQEX.PAS (Delphi)**
Used With	All devices

Description

daqClose is used to close a Personal Daq device. Once the specified device has been closed, no subsequent communication with the device can be performed. In order to re-establish communications with a closed device, the device must be re-opened with the **daqOpen** function.

daqCloseList

DLL Function	**daqCloseList(DaqHandleT handle);**	
C	**daqCloseList(DaqHandleT handle);**	
Visual BASIC	**VBdaqCloseList&(ByVal handle&)**	
Delphi	**daqCloseList(handle:DaqHandleT)**	
Parameters	**handle**	Handle to the open device list to be closed
Returns	**DerrNoError** - No error	
See Also	**daqOpenList**	
Program References	**MULTDEV.C**	
Used With	PersonalDaq55/56	

Description

daqCloseList is used to close a list of PersonalDaq devices opened with the **daqOpenList** function. After the specified device list has been closed, no subsequent communication with the devices can be performed. To re-establish communications with any closed devices, you must open the devices with **daqOpen** or **daqOpenList**.

daqCvtRawDataFormat

DLL Function	**daqCvtRawDataFormat(PVOID buf, daqAdcCvtAction action, DWORD lastRetCount, DWORD scanCount, DWORD chanCount);**	
C	**daqCvtRawDataFormat(PVOID buf, DaqAdcCvtAction action, DWORD lastRetCount, DWORD scanCount, DWORD chanCount);**	
Visual BASIC	**VBdaqCvtRawDataFormatSingle&(buf!(), ByVal action&, ByVal lastRetCount&, ByVal ScanCount&, ByVal ChanCount&)**	
Delphi	**daqCvtRawDataFormat(buf:PWORD;action:daqAdcCvtAction; lastRetCount:DWORD; scanCount:DWORD; chanCount: DWORD);**	
Parameters	**buf**	Pointer to the buffer containing the raw data
	action	The type of conversion action to perform on the raw data
	lastRetCount	The last retCount returned from **daqAdcTransferGetStat**
	scanCount	The length of the raw data buffer in scans
	chanCount	The number of channels per scan in the raw data buffer
Returns	**DerrNoError**	No error
See Also	**daqAdcSetDataFormat**	
Program References	None	
Used With	All devices	

Description

daqCvtRawDataFormat allows the conversion of raw data to a specified format. This function should be called after the raw data has been acquired. See the transfer data functions (**daqAdcTransfer**…) for more details on the actual collection of raw data.

The **buf** parameter specifies the pointer to the data buffer containing the raw data. Prior to calling this function, this user-allocated buffer should already contain the entire raw data transfer. Upon completion, this data buffer will contain the converted data (the buffer must be able to contain all the converted data).

The **DacaRotate** can be used to reformat a circular buffer into a linear buffer.

The **scanCount** parameter specifies the length of the raw buffer in scans. Since the converted data will overwrite the raw data in the buffer, make sure the specified buffer is large enough, physically, to contain all of the converted data.

The **chanCount** parameter specifies the number of channels in each scan.

daqCvtSetAdcRange

DLL Function	**daqCvtSetAdcRange(FLOAT Admin, FLOAT Admax);**	
C	**daqCvtSetAdcRange(FLOAT Admin, FLOAT Admax);**	
Visual BASIC	**VBdaqCvtSetAdcRange&(ByVal ADmin!, ByVal ADmax!)**	
Delphi	**daqCvtSetAdcRange(Admin:single; Admax:single)**	
Parameters	**Admin**	A/D minimum voltage range
	Admax	A/D maximum voltage range
Returns	**DerrNoError**	No error
See Also		
Program References	None	
Used With		

Description

daqCvtSetAdcRange allows you to set the current ADC range for use by the **daqCvt…** functions. This function should not need to be called if used for data collected by the Personal Daq devices.

daqDefaultErrorHandler

DLL Function	**DaqDefaultErrorHandler(DaqHandleT handle, daqError errCode);**	
C	**DaqDefaultErrorHandler(DaqHandleT handle, DaqError errCode);**	
Visual BASIC	**VbdaqDefaultErrorHandler(ByVal handle&, ByVal errCode&)**	
Delphi	**DaqDefaultErrorHandler(handle:DaqHandleT; errCode:daqError)**	
Parameters	**Handle**	Handle to the device to which the default error handler is to be attached.
	ErrCode	The error code number of the detected error
Returns	None	
See Also	**DaqGetLastError, daqProcessError, daqSetDefaultErrorHandler**	
Program References	None	
Used With	All devices	

Description

daqDefaultErrorHandler displays an error message and then exits the application program. When the Personadaq library is loaded, it invokes the default error handler whenever it encounters an error. The error handler may be changed with **daqSetErrorHandler**.

daqFormatError

DLL Function	**daqFormatError(daqError errorNum, PCHAR msg);**	
C	**daqFormatError(DaqError errorNum, PCHAR msg);**	
Visual BASIC	**VBdaqFormatError&(ByVal errorNum&, ByVal msg&)**	
Delphi	**daqFormatError(errorNum:daqerror; msg:PCHAR);**	
Parameters	**daqError**	Personal Daq API error code
	msg	Pointer to a string to return the error text
Returns	**DerrNoError**	No error
See Also	**daqSetDefaultErrorHandler, daqSetErrorHandler, daqProcessError, daqGetLastError, daqDefaultErrorHandler**	
Program References	None	
Used With	All devices	

Description

daqFormatError returns the text-string equivalent for the specified error condition. The error condition is specified by the **daqError** parameter. The error text will be returned in the character string pointed to by the **msg** parameter. The character string space must have been previously allocated by the application before calling this function. The allocated character string should be, at minimum, 64 bytes in length.

For more information on specific error codes refer to the *API Error Codes* on page page **B-33.**

daqGetDeviceCount

DLL Function	**daqGetDeviceCount(DWORD *deviceCount);**	
C	**daqGetDeviceCount(DWORD *deviceCount);**	
Visual BASIC	**VBdaqGetDevice&(deviceCount&)**	
Delphi	**daqGetDeviceCount(deviceCount:PDWORD);**	
Parameters	**deviceCount**	Pointer to which the device count is to be returned
Returns	**DerrNoError**	No error
See Also	**daqGetDeviceList, daqGetDeviceProperties**	
Program References	**ADCEX1.C, FREQEX1.C, PULSEEX1.C, MULTEX1.C, DIGEX1.C, INITEX1.C, DAQEX.FRM (VB), DAQEX.PAS (Delphi)**	
Used With	All devices	

Description

daqGetDeviceCount returns the number of currently configured devices. This function will return the number of devices currently configured in the system. The Personal Daq devices need to be present for this function to operate properly.

daqGetDeviceList

DLL Function	**DaqGetDeviceList(daqDeviceListT *deviceList, DWORD *deviceCount);**	
C	**DaqGetDeviceList(DaqDeviceListT *deviceList, DWORD *deviceCount);**	
Visual BASIC	**VbdaqGetDeviceList&(ByRef deviceList() As String, ByRef deviceCount&)**	
Delphi	**daqGetDeviceList(deviceList:DeviceListTP; devCount:PDWORD);**	
Parameters	**deviceList**	Pointer to memory location to which the device list is to be returned
	deviceCount	Number of devices returned in the device list
Returns	**DerrNoError**	No error
See Also	**daqGetDeviceCount, daqGetDeviceProperties, daqOpen,**	
Program References	**ADCEX1.C, FREQEX1.C, PULSEEX1.C, MULTEX1.C, DIGEX1.C, INITEX1.C, DAQEX.FRM (VB), DAQEX.PAS (Delphi)**	
Used With	All devices	

Description

daqGetDeviceList returns a list of currently configured devices. This function will return the device names in the **deviceList** parameter for the number of devices returned by the **deviceCount** parameter. Each **deviceList** entry contains a device name consisting of up to 64 characters. The device name can then be used with the **daqOpen** function to open the specific device.

daqGetDeviceProperties

DLL Function	**daqGetDeviceProperties(LPSTR daqName, daqDevicePropsT *deviceProps);**	
C	**daqGetDeviceProperties(LPSTR daqName, DaqDevicePropsT *deviceProps);**	
Visual BASIC	**VBdaqGetDeviceProperties&(ByVal daqName$, deviceProps As daqDevicePropsT)**	
Delphi	**daqGetDeviceProperties(daqName:string; deviceProps: daqDevicePropsTP);**	
Parameters	**daqName**	Pointer to a character string representing the name of the device for which to retrieve properties
	DeviceCount	Number of devices returned in the device list
Returns	**DerrNoError**	No error
See Also	**daqGetDeviceCount, daqGetDeviceList, daqOpen**	
Program References	**ADCEX1.C, FREQEX1.C, PULSEEX1.C, MULTEX1.C, DIGEX1.C, INITEX1.C, DAQEX.FRM (VB), DAQEX.PAS (Delphi)**	
Used With	All devices	

Description

daqGetDeviceProperties returns the properties for the specified device as indicated by the **daqName** parameter. This name should be a valid name of a configured device. The properties for the device are returned in the **deviceProps** parameter. **deviceProps** is a pointer to user-allocated memory which will hold the device-properties structure. This memory must have been allocated before calling this function.

For detailed device-property structure layout, refer the to *daq Device Properties Definition* table.

daqGetDriverVersion

DLL Function	**daqGetDriverVersion(PDWORD version);**
C	**daqGetDriverVersion(PDWORD version);**
Visual BASIC	**VBdaqGetDriverVersion&(version&)**
Delphi	**daqGetDriverVersion(var version:DWORD)**
Parameters	**version** Pointer to the version number of the current device driver.
Returns	**DerrNoError** No error
See Also	
Program References	**ADCEX1.C, FREQEX1.C, PULSEEX1.C, MULTEX1.C, DIGEX1.C, INITEX1.C, DAQEX.FRM (VB), DAQEX.PAS (Delphi)**
Used With	All devices

Description

daqGetDriverVersion allows you to get the revision level of the driver currently in use.

daqGetLastError

DLL Function	**daqGetLastError(DaqHandleT handle, daqError *errCode);**
C	**daqGetLastError(DaqHandleT handle, DaqError *errCode);**
Visual BASIC	**VBdaqGetLastError&(ByVal handle&, errCode&)**
Delphi	**daqGetLastError(handle:DaqHandleT; var errCode:daqError);**
Parameters	**handle** Handle to the device ***errCode** Returned last error code
Returns	**DerrNoError** No error
See Also	**daqDefaultErrorHandler, daqProcessError, daqSetDefaultErrorHandler**
Program References	None
Used With	All devices

Description

daqGetLastError allows you to retrieve the last error condition registered by the driver.

daqIORead

DLL Function	**daqIORead(DaqHandleT handle, daqIODeviceType devType, daqIODevicePort devPort, DWORD whichDevice, daqIOExpansionPort whichExpPort, PDWORD value);**
C	**daqIORead(DaqHandleT handle, DaqIODeviceType devType, DaqIODevicePort devPort, DWORD whichDevice, DaqIOExpansionPort whichExpPort, PDWORD value);**
Visual BASIC	**VBdaqIORead&(ByVal handle&, ByVal devType&, ByVal devPort&, ByVal whichDevice&, ByVal whichExpPort&, value&)**
Delphi	**daqIORead(handle:DaqHandleT; devType:daqIODeviceType; dvPort:daqIODevicePort; whichDevice:DWORD; whichExpPort:daqIOExpansionPort; var value:DWORD);**
Parameters	**handle** Handle to the device to perform the IO read **devType** IO Device type **devPort** IO port selection **whichDevice** IO device instance to read from **whichExpPort** IO device expansion port to read from **value** IO value read
Returns	**DerrNoError** No error
See Also	**daqIOReadBit, daqIOWrite, daqIOWriteBit**
Program References	**DIGEX1.C, DAQEX.FRM (VB), DAQEX.PAS (Delphi)**
Used With	All devices except Personal Daq/54

Description

daqIORead allows you to read the specified port on the selected device. The read operation will return the current state of the port in the **value** parameter. Normally, if the selected port is a byte-wide port, the port state will occupy the low-order byte of the **value** parameter. Digital IO channels for the port correspond to each bit within this low-order byte. If the bit is set, it indicates the channel is in a high state. If the bit is not set, the channel is indicated to be in a low state.

daqIOReadBit

DLL Function	**daqIOReadBit(DaqHandleT handle, daqIODeviceType devType, daqIODevicePort devPort, DWORD whichDevice, daqIOExpansionPort whichExpPort, DWORD bitNum, PBOOL bitValue);**	
C	**daqIOReadBit(DaqHandleT handle, DaqIODeviceType devType, DaqIODevicePort devPort, DWORD whichDevice, DaqIOExpansionPort whichExpPort, DWORD bitNum, PBOOL bitValue);**	
Visual BASIC	**VBdaqIOReadBit&(ByVal handle&, ByVal devType&, ByVal devPort&, ByVal whichDevice&, ByVal whichExpPort&, ByVal bitNum&, bitValue&)**	
Delphi	**daqIOReadBit(handle:DaqHandleT; devType:daqIODeviceType; dvPort:daqIODevicePort; whichDevice:DWORD; whichExpPort:daqIOExpansionPort; bitNum:DWORD; var bitValue:longbool)**	
Parameters	**handle**	Handle to the device from which to perform the IO
	devType	IO Device type
	devPort	IO device port selection
	whichDevice	IO device selection
	whichExpPort	IO expansion port address
	bitNum	IO port bit location to read
	bitValue	IO port bit value (**TRUE - high, FALSE - low**)
Returns	**DerrNoError**	No error
See Also	**daqIORead, daqIOWrite, daqIOWriteBit**	
Program References	**DIGEX1.C, DAQEX.FRM (VB), DAQEX.PAS (Delphi)**	
Used With	All devices except Personal Daq/54	

Description

daqIOReadBit allows you to read a specified bit on the selected device and port. The read operation will return the current state of the selected bit in the **bitValue** parameter. The selected bit (specified by the **bitNum** parameter) corresponds to the IO channel on the port which is to be read. The **bitValue** will be **TRUE** indicating a high state or **FALSE** indicating a low state.

daqIOWrite

DLL Function	**daqIOWrite(DaqHandleT handle, daqIODeviceType devType, daqIODevicePort devPort, DWORD whichDevice, daqIOExpansionPort whichExpPort, DWORD value);**	
C	**daqIOWrite(DaqHandleT handle, DaqIODeviceType devType, DaqIODevicePort devPort, DWORD whichDevice, DaqIOExpansionPort whichExpPort, DWORD value);**	
Visual BASIC	**VBdaqIOWrite&(ByVal handle&, ByVal devType&, ByVal devPort&, ByVal whichDevice&, ByVal whichExpPort&, ByVal value&)**	
Delphi	**daqIOWrite(handle:DaqHandleT; devType:daqIODeviceType; dvPort:daqIODevicePort; whichDevice:DWORD; whichExpPort:daqIOExpansionPort; value:DWORD);**	
Parameters	**handle**	Handle of the device to perform an IO write operation
	devType	IO device type
	devPort	IO device port selection
	whichDevice	IO device selection
	whichExpPort	IO device expansion port address
	value	Value to write
Returns	**DerrNoError**	No error
See Also	**daqIORead, daqIOWriteBit, daqIOReadBit**	
Program References	**DIGEX1.C, DAQEX.FRM (VB), DAQEX.PAS (Delphi)**	
Used With	All devices except Personal Daq/54	

Description

daqIOWrite allows you to write to the specified port on the selected device. The write operation will write the settings indicated in the **value** parameter to the selected port. The **value** written will depend on the width of the selected port. Normally, for byte-wide ports, only the low-order byte of the value parameter will be written. The IO channels for the port correspond to each bit within the value written. If the channel is to be driven to a high state, then the corresponding bit should be set. Likewise, if the channel is to be driven to a low state, then the corresponding bit should not be set.

daqIOWriteBit

DLL Function	**daqIOWriteBit(DaqHandleT handle, daqIODeviceType devType, daqIODevicePort devPort, DWORD whichDevice, daqIOExpansionPort whichExpPort, DWORD bitNum, BOOL bitValue);**	
C	**daqIOWriteBit(DaqHandleT handle, DaqIODeviceType devType, DaqIODevicePort devPort, DWORD whichDevice, DaqIOExpansionPort whichExpPort, DWORD bitNum, BOOL bitValue);**	
Visual BASIC	**VBdaqIOWriteBit&(ByVal handle&, ByVal devType&, ByVal devPort&, ByVal whichDevice&, ByVal whichExpPort&, ByVal bitNum&, ByVal bitValue&)**	
Delphi	**daqIOWriteBit(handle:DaqHandleT; devType:daqIODeviceType; dvPort:daqIODevicePort; whichDevice:DWORD; whichExpPort:daqIOExpansionPort; bitNum:DWORD; bitValue:longbool)**	
Parameters	**handle**	Handle of the device to perform an IO write to
	devType	IO device type
	devPort	IO device port selection
	whichDevice	IO device selection
	whichExpPort	IO device expansion port address
	bitNum	Bit number to write
	bitValue	Bit value to write (**TRUE – high, FALSE - low**)
Returns	**DerrNoError**	No error
See Also	**daqIOWrite, daqIORead, daqIOReadBit**	
Program References	**DIGEX1.C, DAQEX.FRM (VB), DAQEX.PAS (Delphi)**	
Used With	All devices except Personal Daq/54	

Description

daqIOWriteBit allows you to write a specified bit on the selected device and port. The write operation will write the specified bit value to the bit selected. The selected bit, specified by the **bitNum** parameter, corresponds to the channel on the port for the IO to be driven. The **bitValue** parameter should be set to **TRUE** to drive the channel to a high state or **FALSE** indicating a low state.

daqOnline

DLL Function	**daqOnline(DaqHandleT handle, PBOOL online);**	
C	**daqOnline(DaqHandleT handle, PBOOL online);**	
Visual BASIC	**VBdaqOnline&(ByVal handle&, online&)**	
Delphi	**daqOnline(handle: DaqHandleT; var online: longbool)**	
Parameters	**handle**	Handle of the device to test for online
	online	Boolean indicating whether the device is currently online
Returns	**DerrNoError**	No error
See Also	**daqOpen, daqClose**	
Program References	None	
Used With	All devices	

Description

daqOnline allows you to determine if a device is online. The device **handle** must be a valid device handle which has been opened using the **daqOpen** function. The **online** parameter indicates the current online state of the device (**TRUE** - device online; **FALSE** - device not online).

daqOpen

DLL Function	**daqOpen(LPSTR daqName);**
C	**daqOpen(LPSTR daqName);**
Visual BASIC	**VBdaqOpen&(ByVal daqName$)**
Delphi	**daqOpen(devName: PChar)**
Parameters	**daqName** String representing the name of the device to be opened
Returns	A handle to the specified device
See Also	**daqClose, daqOnline**
Program References	**ADCEX1.C, FREQEX1.C, PULSEEX1.C, MULTEX1.C, DIGEX1.C, INITEX1.C, DAQEX.FRM (VB), DAQEX.PAS (Delphi)**
Used With	

Description

daqOpen allows you to open an installed Personal Daq device for operation. The **daqOpen** function will initiate a session for the device name specified by the **daqName** parameter by opening the device, initializing it, and preparing it for further operation. The **daqName** specified must reference a currently configured device. **daqOpen** should be performed prior to any other operation performed on the device. This function will return a device handle that is used by other functions to reference the device. Once the device has been opened, the device handle should be used to perform subsequent operations on the device.

Most functions in this manual require a device handle in order to perform their operation. When the device session is complete, **daqClose** may be called with the device handle to close the device session.

daqOpenList

DLL Function	**daqOpenList(DaqDeviceListT * deviceList PDWORD deviceIndex, DWORD deviceCount);**	
C	**daqOpenList(DaqDeviceListT *deviceList daqName, PDWORD deviceIndex, DWORD deviceCount);**	
Visual BASIC	**VBdaqOpenList&(ByVal deviceList$ deviceIndex&, ByVal deviceCount&)**	
Delphi	**daqOpenList(var deviceList: DaqDeviceListT; var deviceIndex:PDWORD deviceCount: DWORD)**	
Parameters	**DeviceList**	Array of names that represents the devices to be opened
	DeviceIndex	Array of indexes that represents logical device mapping for the corresponding device name
	DeviceCount	Number of devices in the array to open
Returns	A handle to the specified **DeviceList**	
See Also	**daqCloseList, daqOnline, daqOpen, daqClose**	
Program References	**MULTDEV.C**	
Used With	PersonalDaq55/56	

Description

daqOpenList allows you to open a list of **deviceCount** installed Personal Daq devices. The **daqOpenList** function will initiate a session for the device names specified by the **deviceList** parameter by opening the devices, initializing them, and preparing them for further operation. Once initialized, all devices in the list will be referenced via the same handle, and all operations performed on that handle will affect all devices in the list. The specified **deviceList** must reference currently-configured devices that are currently connected to the host system. The **deviceList** parameter should consist of device names as returned by the **daqGetDeviceList** function.

The **deviceIndex** parameter will return a list of indexes representing the logical position of the corresponding named device in the opened device list. This index can then be used to program the appropriate device/channel pair in such functions as **daqAdcSetScan, daqAdcSetMux, daqAdcRd...** etc.

The **daqOpenList** call should be performed prior to any other operation performed on the devices. This function will return a device handle that other functions use to reference the devices. Once the device has been opened, the device handle should be used to perform subsequent operations on the devices.

Most functions in this manual require a device handle in order to perform their operation. To close the device list when the session is complete, **daqCloseList** may be called with the returned handle.

daqProcessError

DLL Function	**daqProcessError(DaqHandleT handle, daqError errCode);**	
C	**daqProcessError(DaqHandleT handle, DaqError errCode);**	
Visual BASIC	**VBdaqProcessError&(ByVal handle&, ByVal errCode&)**	
Delphi	**daqProcessError(handle:DaqHandleT; errCode:daqError)**	
Parameters	**handle**	Handle to the device for which the specified error is to be processed.
	errCode	Specifies the device error code to process
Returns	Refer to *API Error Codes* on page page B-33.	
See Also	**daqSetDefaultErrorHandler, daqGetLastError, daqDefaultErrorHandler**	
Program References	None	
Used With	All devices	

Description

daqProcessError allows an application to initiate an error for processing by the driver. This command can be used when it is desirable for the application to initiate processing for a device-defined error.

daqSetDefaultErrorHandler

DLL Function	**daqSetDefaultErrorHandler(daqErrorHandlerFPT handler);**	
C	**daqSetDefaultErrorHandler(DaqErrorHandlerFPT handler);**	
Visual BASIC	**VBdaqSetDefaultErrorHandler&(ByVal handler&)**	
Delphi	**daqSetDefaultErrorHandler(handler:daqErrorHandlerFPT)**	
Parameters	**handler**	Pointer to a user-defined error handler function.
Returns	**DerrNoError** - No error	
See Also	**daqDefaultErrorHandler, daqGetLastError, daqProcessError, daqSetErrorHandler**	
Program References	**ADCEX1.C, FREQEX1.C, PULSEEX1.C, MULTEX1.C, DIGEX1.C, INITEX1.C, DAQEX.FRM (VB), DAQEX.PAS (Delphi)**	
Used With	All devices	

Description

daqSetDefaultErrorHandler allows you to set the driver to use the default error handler specified for all devices.

daqSetErrorHandler

DLL Function	**daqSetErrorHandler(DaqHandleT handle, daqErrorHandlerFPT handler);**	
C	**daqSetErrorHandler(DaqHandleT handle, DaqErrorHandlerFPT handler);**	
Visual BASIC	**VBdaqSetErrorHandler&(ByVal handle&, ByVal handler&)**	
Delphi	**daqSetErrorHandler(handle:DaqHandleT; handler:daqErrorHandlerFPT)**	
Parameters	**handle**	Handle to the device to which to attach the specified error handler
	handler	Pointer to a user defined error handler function.
Returns	**DerrNoError**	No error
See Also	**daqSetDefaultErrorHandler, daqDefaultErrorHandler, daqGetLastError, daqProcessError**	
Program References	**ADCEX1.C, FREQEX1.C, PULSEEX1.C, MULTEX1.C, DIGEX1.C, INITEX1.C, DAQEX.FRM (VB), DAQEX.PAS (Delphi)**	
Used With		

Description

daqSetErrorHandler specifies the routine to call when an error occurs in any command. The default routine displays a message and then terminates the program. If this is not desirable, use this command to specify your own routine to be called when errors occur. If you want no action to occur when a command error is detected, use this command with a null (0) parameter. The default error routine is **daqDefaultHandler**.

daqSetOption

DLL Function	**daqSetOption(DaqHandleT handle, DWORD chan, DWORD flags, daqOptionType optionType, FLOAT optionValue);**	
C	**daqSetOption(DaqHandleT handle, DWORD chan, DWORD flags, DaqOptionType optionType, FLOAT optionValue);**	
Visual BASIC	**VBdaqSetOption&(ByVal handle&, ByVal chan&, ByVal flags&, ByVal optionType&, ByVal optionValue!)**	
Delphi	**daqSetOption(Handle:DaqHandleT; chan:DWORD; flags:DWORD; optionType:daqOptionType; optionValue:FLOAT)**	
Parameters	**handle**	The handle to the device for which to set the option
	chan	The channel number on the device for which the option is to be set
	flags	Flags specifying the options to use.
	optionType	Specifies the type of option.
	optionValue	The value of the option to set
Returns	**DerrNoError**	No error
See Also	**daqAdcExpSetChanOption,**	
Program References	**PULSEEX1.C, MULTEX1.C, FREQEX1.C, DAQEX.FRM (VB), DAQEX.PAS (Delphi)**	
Used With	All devices	

Description

daqSetOption allows the setting of options for a device's channel/signal path configuration.

chan specifies which channel the option applies to.
flags parameter specifies how the option is to be applied.
optionType specifies the type of option to apply to the channel.
optionValue parameter specifies the value of the option.

Note: For more information regarding **optionValue** and **optionType** refer to page B-32, Table 8.

Note: The following information does not apply to Personal Daq/54.

daqSetOption must be used to properly configure Personal Daq's frequency input and pulse count channels. To configure frequency input and pulse count channels, the following option types need to be specified for each frequency input/pulse count channel in the scan sequence:

DcotpDaqPulses: Configures the specified channel as a pulse count channel.
DcotpDaqRising: Configures the rising flag for the pulse channel.
DcotpDaqDebounceTime: Configures the debounce time for the frequency/pulse channel.

The following options must also be specified to configure frequency input channels:

DcotpDaqMinFreq: Configures the minimum frequency range for the specified channel.
DcotpDaqMaxFreq: Configures the maximum frequency range for the specified channel.
DcotpDaqFreqRes: Configures the resolution of the specified frequency channel.
DcotpDaqOverrangeProtect: Enables the overrange protection mode.

daqSetTimeout

DLL Function	**daqSetTimeout(DaqHandleT handle, DWORD mSecTimeout);**	
C	**daqSetTimeout(DaqHandleT handle, DWORD mSecTimeout);**	
Visual BASIC	**VBdaqSetTimeout&(ByVal handle&, ByVal mSecTimeout&)**	
Delphi	**daqSetTimeout(handle:DaqHandleT; mSecTimeout:DWORD)**	
Parameters	**handle**	Handle to the device for which the event time-out is to be set
	mSecTimeout	Specifies time-out (ms) for events
Returns	**DerrNoError**	No error
See Also	**daqWaitForEvent, daqWaitForEvents**	
Program References	None	
Used With	All devices	

Description

daqSetTimeout allows you to set the time-out for waiting on a single event or multiple events. This function can be used in conjunction with the **daqWaitForEvent** and **daqWaitForEvents** functions to specify a maximum amount of time to wait for the event(s) to be satisfied.

The **mSecTimeout** parameter specifies the maximum amount of time (in milliseconds) to wait for the event(s) to occur. If the event(s) do not occur within the specified time-out, the **daqWaitForEvent** and/or **daqWaitForEvents** will return.

daqWaitForEvent

DLL Function	**daqWaitForEvent(DaqHandleT handle, daqTransferEvent daqEvent);**	
C	**daqWaitForEvent(DaqHandleT handle, DaqTransferEvent daqEvent);**	
Visual BASIC	**VBdaqWaitForEvent&(ByVal handle&, ByVal daqEvent&)**	
Delphi	**daqWaitForEvent(handle:DaqHandleT; daqEvent:daqTransferEvent)**	
Parameters	**handle**	Handle of the device for which to wait of the specified event
	daqEvent	Specifies the event to wait on
Returns	**DerrNoError**	No error
See Also	**daqWaitForEvents, daqSetTimout**	
Program References	None	
Used With	All devices	

Description

daqWaitForEvent allows you to wait on a specific Personal Daq event to occur on the specified device. This function will not return until the specified event has occurred or the wait has timed out— whichever comes first. The event time-out can be set with the **daqSetTimout** function. See the *Transfer Event Definitions* table for event definitions.

daqWaitForEvents

DLL Function	**daqWaitForEvents(DaqHandleT *handles, daqTransferEvent *daqEvents, DWORD eventCount, BOOL *eventSet, daqWaitMode waitMode);**	
C	**daqWaitForEvents(DaqHandleT *handles, DaqTransferEvent *daqEvents, DWORD eventCount, BOOL *eventSet, DaqWaitMode waitMode);**	
Visual BASIC	**VBdaqWaitForEvents&(handles&(), daqEvents&(), ByVal eventCount&, eventSet&(), ByVal waitMode&)**	
Delphi	**daqWaitForEvents(handles:DaqHandlePT; daqEvents:daqTransferEventP; eventCount:DWORD; eventSet:PBOOL; waitMode:daqWaitMode):**	
Parameters	***handles**	Pointer to an array of handles which represent the list of device on which to wait for the events
	***daqEvents**	Pointer to an array of events which represents the list of events to wait on
	eventCount	Number of defined events to wait on
	***eventSet**	Pointer to an array of Booleans indicating if the events have been satisfied.
	waitMode	Specifies the mode for the wait
Returns	**DerrNoError**	No error
See Also	**daqWaitForEvent, daqSetTimeout**	
Program References	None	
Used With	All devices	

Description

daqWaitForEvents allows you to wait on specific Personal Daq events to occur on the specified devices. This function will wait on the specified events and will return based upon the criteria selected with the **waitMode** parameter. A time-out for all events can be specified using the **daqSetTimeout** command.

Events to wait on are specified by passing an array of event definitions in the **events** parameter. The number of events is specified with the **eventCount** parameter. See the *Transfer Event Definitions* table for **events** parameter definitions. Also see the *Transfer Event Wait Mode Definitions* table for **waitMode** parameter definitions.

API Reference Tables

These tables provide information for programming with the Personal Daq Application Programming Interface. Information includes channel identification and error codes, as well as valid parameter values and descriptions. The tables are organized as follows:

API Parameter Reference Tables		
Table	**Sub-Title/Parameter/Description**	**Page**
1. daq Device Property Definitions - **daqGetDeviceProperties**	Identifies the format (**DWORD**, **STRING**, or **FLOAT**) for property parameters	B-30
2. General I/O Definintions (Does not apply to Personal Daq/54)	Digital I/O Port Connection: **devPort – daqIODevicePort** I/O Port Type: **devType – daqIODeviceType** I/O Operation Code: **whichExpPort –daqIOExpansionPort** I/O Operation Code: **daqIOOperationCode**	B-30
3. Event-Handling Definitions	Transfer Event Definitions - **daqTransferEvent** Transfer Event Wait Mode Definitions - **daqWaitMode**	B-31
4. Hardware Version Definitions	**daqHardwareVersion**	B-31
5. Analog Gain Definitions	Identifies gain codes for Personal Daq base unit	B-31
6. Trigger Source Definitions	**daqAdcTriggerSource** **daqEnhTrigSensT**	B-31
7. Miscellaneous Definitions	Scan Flag Definitions - **daqAdcFlag** Frequency vs Period - **daqAdcRateMode** Acquisition Mode Definitions - **daqAdcAcqMode** Transfer Mask Definitions - **daqAdcTransferMask** File Open Mode Definitions - **daqAdcOpenMode** Acquisition/Transfer Active Flag Definitions - **daqAdcActiveFlag** Acquisition State - **daqAdcAcqState** BufferTransfer Mask- **daqAdcBufferXferMask**	B-32
8. Setting Options	**optionType** **optionValue**	B-32
9. API Error Codes	Identifies API errors by number and description	B-33

Table 1. daq Device Property Definitions	daqGetDeviceProperties	
Property	**Description**	**Format**
deviceType	Main Chassis Device Type Definition	DWORD
basePortAddress	Not Used	DWORD
dmaChannel	Not Used	DWORD
protocol	Host computer Interface used	DWORD
alias	Device Alias Name	STRING
maxAdChannels	Maximum A/D channels (with full expansion)	DWORD
maxDaChannels	Not Used	DWORD
maxDigInputBits	Maximum Dig. Inputs (with full expansion)	DWORD
maxDigOutputBits	Maximum Dig. Outputs (with full expansion)	DWORD
maxCtrChannels	Maximum Counter/Timers (with full expansion)	DWORD
mainUnitAdChannels	Maximum Main Unit A/D channels (no expansion)	DWORD
mainUnitDaChannels	Not Used	DWORD
mainUnitDigInputBits	Maximum Main Unit Digital Inputs (no expansion)	DWORD
mainUnitDigOutputBits	Maximum Main Unit Digital Outputs (no expansion)	DWORD
mainUnitCtrChannels	Maximum Main Unit Counter/Timer channels (no exp.)	DWORD
adFifoSize	Not Used	DWORD
daFifoSize	Not Used	DWORD
adResolution	Maximum A/D Converter Resolution	DWORD
daResolution	Not Used	DWORD
adMinFreq	Minimum Scan Frequency (Hz)	FLOAT
adMaxFreq	Maximum Scan Frequency (Hz)	FLOAT
daMinFreq	Not Used	FLOAT
daMaxFreq	Not Used	FLOAT
MainUnitCjcChannels	Number of CJC channels on the main unit	DWORD
TotalCjcChannels	Number of CJC channels in the system	DWORD
SerialNumber	Serial Number for the device	DWORD
ExpansionUnits	Number of expansion units	DWORD
SubDeviceType	Type of expansion unit	DWORD

Table 2. General I/O Definitions		
Digital I/O Port Connections	**devPort - daqIODevicePort**	
Personal Daq Port	**Value**	**Description**
DiodpPdaqPort1	00h	Local Port 1
DiodpPdaqPort2	01h	Local Port 2
DiodpPdaqExpPort1	02h	Expansion Port 1 (PDQ Option)
DiodpPdaqExpPort2	03h	Expansion Port 2 (PDQ Option)
DiodpPdaqPowerUpPort1	10h	Power-up Setting for Local Port 1
DiodpPdaqPowerUpPort2	11h	Power-up Setting for Local Port 2
DiodpPdaqPowerUpExpPort1	12h	Power-up Setting for Exp Port 1
DiodpPdaqPowerUpExpPort2	13h	Power-up Setting for Exp Port 2
I/O Port Type	**devType - daqIODeviceType**	
Personal Daq Type	**Value**	**Description**
DiodtPdaqDigIO	0ch	For all units except Personal Daq/54
I/O Operation Code Definitions	**whichDevice**	
Personal Daq Device	**Value**	**Description**
Device Index	Varies	See daqOpenList function
	whichExpPort - daqIOExpansionPort	
Device Expansion Port	0	Field ignored for Personal Daq

daqIOOperationCode

DioocReadByte	0
DioocWriteByte	1
DioocReadWord	2
DioocWriteWord	3
DioocReadDWord	4
DioocWriteDWord	5

Note: daqIOOperationCode does not apply to Personal Daq/54.

Table 3. Event-Handling Definitions

Transfer Event Definitions - `daqTransferEvent`	
`DteAdcData`	`0`
`DteAdcDone`	`1`
`DteDacData`	`2`
`DteDacDone`	`3`
`DteIOData`	`4`
`DteIODone`	`5`

Transfer Event Wait Mode Definitions - `daqWaitMode`	
`DwmNoWait`	`0`
`DwmWaitForAny`	`1`
`DwmWaitForAll`	`2`

Table 4. Hardware Version Definitions `daqHardwareVersion`

Definition	Value
`PersonalDaq56`	`100H`
`PersonalDaq55` (Note)	`101H`

Note: Includes "NE" version.

Table 5. ADC Gain Definitions

Base Unit Combination	Parameters	Voltage Range
`PgainX1`	`0`	- 4 to + 4 V
`PgainX2`	`1`	- 2 to + 2 V
`PgainX4`	`16`	- 1 to + 1 V
`PgainX8`	`17`	- 500 to + 500 mV
`PgainX16`	`18`	- 250 to + 250 mV
`PgainX32`	`19`	- 125 to + 125 mV
`PgainX64`	`20`	- 62.5 to + 62.5 mV
`PgainX128`	`21`	- 31 to + 31 mV
`PgainX1 + PgainDiv5`	`0 + 8`	- 20 to + 20 V
`PgainX2 + PgainDiv5`	`1 + 8`	- 10 to + 10 V
`PgainX4 + PgainDiv5`	`16 + 8`	- 5 to + 5 V
`PgainX8 + PgainDiv5`	`17 + 8`	- 2.5 to + 2.5 V
`PgainX16 + PgainDiv5`	`18 + 8`	- 1.25 to + 1.25 V
`PgainX32 + PgainDiv5`	`19 + 8`	- 312 to + 312 mV
`PgainX64 + PgainDiv5`	`20 + 8`	- 156 to + 156 mV
`PgainX128 + PgainDiv5`	`21 + 8`	- 62 to + 62 mV
`PgainDiv5`	`8`	

Table 6. Trigger Source Definitions

`daqAdcTriggerSource`		
`DatsImmediate`	`0`	
`DatsSoftware`	`1`	
`DatsSoftwareAnalog`	`6`	
`DatsDigitalChannel`	`8`	*
`DatsFrequency`	`9`	*
`DatsPulseCount`	`10`	*
`DatsPulseTotalize`	`11`	*
`DatsAnalogSELow`	`12`	
`DatsAnalogSEHigh`	`13`	

`daqEnhTrigSensT`	
`DetsRisingEdge`	`0`
`DetsFallingEdge`	`1`

* `DatsDigitalChannel`, `DatsFrequency`, `DatsPulseCount`, and `DatsPulseTotalize` do not apply to Personal Daq/54.

Table 7. ADC Miscellaneous Definitions

ADC Flag Definitions - daqAdcFlag

Analog/High Speed Digital Flag		Single Ended/Differential Flag		Single-Ended Channel Flag	
DafAnalog	00h	DafSingleEnded	00h	DafSingleEndedLow	0000h
DafScanDigital *	01h	DafDifferential	08h	DafSingleEndedHigh	1000h
Measurement Durations		**Digital/Frequency Channel Types**		**Thermocouple Types**	
DafMeasDuration610	000000h	DafDioDirect *	00000h	DafTcTypeJ	080h
DafMeasDuration370	100000h	DafCtrPulse *	20000h	DafTcTypeK	100h
DafMeasDuration310	200000h	DafCtrTotalize *	40000h	DafTcTypeT	180h
DafMeasDuration130	300000h	DafCtrFreq *	80000h	DafTcTypeE	200h
DafMeasDuration110	400000h	DafCtrDutyLo *	100000h	DafTcTypeN	280h
DafMeasDuration40	500000h	DafCtrDutyHi *	200000h	DafTcTypeS	380h
DafMeasDuration20	600000h			DafTcTypeR	400h
DafMeasDuration12_5	700000h			DafTcTypeB	480h

Frequency vs Period – daqAdcRateMode	
DarmPeriod	0
DarmFrequency	1

ADC Acquisition Mode Definitions - daqAdcAcqMode	
DaamNShot	0
DaamNShotRearm	1
DaamInfinitePost	2
DaamPrePost	3

ADC Transfer Mask Definitions - daqAdcTransferMask	
DatmCycleOff	00h
DatmCycleOn	01h
DatmUpdateBlock	00h
DatmUpdateSingle	02h
DatmUserBuf	00h
DatmSavePreTrig	20h
DatmIgnoreOverruns	10h
DatmDriverBuf	08h

ADC Clock Source Definitions daqAdcClockSource	

ADC File Open Mode Definitions daqAdcOpenMode	
DaomAppendFile	0
DaomWriteFile	1
DaomCreateFile	2

ADC Acquisition/Transfer Active Flag Definitions daqAdcActiveFlag	
DaafAcqActive	01h
DaafAcqTriggered	02h
DaafTransferActive	04h

ADC Acquisition State - daqAdcAcqState	
DaasPreTrig	0
DaasPostTrig	1

ADC Buffer Transfer Mask - daqAdcBufferXferMask	
DabtmWait	00h
DabtmRetAvail	02h
DabtmNoWait	04h
DabtmRetNotDone	08h
DabtmPreTrigBlock	10h

ADC Filter Type - daqAdcSetFilter	
daftSWAvg	1

Table 8. Setting Options

optionType		
DcotpdaqRising	0	Sets the rising flag for the Pulse Channel
DcotpdaqDebounceTime	1	Sets the debounce time for the Freq/Pulse Channel
DcotpdaqMinFreq	2	Sets the min freq range for the Freq Channel
DcotpdaqMaxFreq	3	Sets the max freq range for the Freq Channel
DcotpdaqPulses	4	Sets the specified Freq/Pulse channel as Pulse Count
DcotpdaqFreqRes	5	Sets the resolution of the Freq Channel
DcotpdaqRunningCal	7	Sets the running calibration flag for the system
DcotpDaqOverrangeProtect	8	Enables the over range protection mode

optionValue			
DcovEdgeRising *	00h	Sets the edge sensing to rising for the Pulse/Freq Channel
DcovDutyHigh *	02h	Sets the Pulse Channel to Duty Cycle High
DcovEdgeFalling *	08h	Sets the edge sensing to falling for the Pulse/Freq Channel
DcovDutyLow *	0Ah	Sets the Pulse Channel to Duty Cycle Low
DcovPulseTotalize *	10h	Sets the Pulse Channel to Totalize
DcovPulseCount *	00h	Sets the Channel as Pulse Count (Default)
Debounce Settings		
DcovDebounce0 *	00h	Set Debounce to 0 us for the Pulse/Freq channel
DcovDebounce600 *	01h	Set Debounce to 600 us for the Pulse/Freq channel
DcovDebounce2500 *	02h	Set Debounce to 2500 us for the Pulse/Freq channel
DcovDebounce10000 *	03h	Set Debounce to 10000 us for the Pulse/Freq channel

* **Note**: The Asterisks in the preceding tables indicate that the definitions do not apply to Personal Daq/54.

Table 9. API Error Codes

Error Name	Code # hex - dec	Description
DerrNoError	**00h - 0**	No error
DerrBadChannel	**01h - 1**	Specified LPT channel was out-of-range
DerrNotOnLine	**02h - 2**	Requested device is not online
DerrNodaqbook	**03h - 3**	daqBook is not on the requested channel
DerrBadAddress	**04h - 4**	Bad function address
DerrFIFOFull	**05h - 5**	FIFO Full detected, possible data corruption
DerrBadDma	**06h - 6**	Bad or illegal DMA channel or mode specified for device
DerrBadInterrupt	**07h - 7**	Bad or illegal INTERRUPT level specified for device
DerrDmaBusy	**08h - 8**	DMA is currently being used
DerrInvChan	**10h - 16**	Invalid analog input channel
DerrInvCount	**11h - 17**	Invalid count parameter
DerrInvTrigSource	**12h - 18**	Invalid trigger source parameter
DerrInvLevel	**13h - 19**	Invalid trigger level parameter
DerrInvGain	**14h - 20**	Invalid channel gain parameter
DerrInvDacVal	**15h - 21**	Invalid DAC output parameter
DerrInvExpCard	**16h - 22**	Invalid expansion card parameter
DerrInvPort	**17h - 23**	Invalid port parameter
DerrInvChip	**18h - 24**	Invalid chip parameter
DerrInvDigVal	**19h - 25**	Invalid digital output parameter
DerrInvBitNum	**1Ah - 26**	Invalid bit number parameter
DerrInvClock	**1Bh - 27**	Invalid clock parameter
DerrInvTod	**1Ch - 28**	Invalid time-of-day parameter
DerrInvCtrNum	**1Dh - 29**	Invalid counter number
DerrInvCntSource	**1Eh - 30**	Invalid counter source parameter
DerrInvCtrCmd	**1Fh - 31**	Invalid counter command parameter
DerrInvGateCtrl	**20h - 32**	Invalid gate control parameter
DerrInvOutputCtrl	**21h - 33**	Invalid output control parameter
DerrInvInterval	**22h - 34**	Invalid interval parameter
DerrTypeConflict	**23h - 35**	An integer was passed to a function requiring a character
DerrMultBackXfer	**24h - 36**	A second background transfer was requested
DerrInvDiv	**25h - 37**	Invalid Fout divisor
Temperature Conversion Errors		
DerrTCE_TYPE	**26h - 38**	TC type out-of-range
DerrTCE_TRANGE	**27h - 39**	Temperature out-of-CJC-range
DerrTCE_VRANGE	**28h - 40**	Voltage out-of-TC-range
DerrTCE_PARAM	**29h - 41**	Unspecified parameter value error
DerrTCE_NOSETUP	**2Ah - 42**	**dacTCConvert** called before **dacTCSetup**
daqBook		
DerrNotCapable	**2Bh - 43**	Device is incapable of function
Background		
DerrOverrun	**2Ch - 44**	A buffer overrun occurred
Zero and Cal Conversion Errors		
DerrZCInvParam	**2Dh - 45**	Unspecified parameter value error
DerrZCNoSetup	**2Eh - 46**	**dac…Convert** called before **dac…Setup**
DerrInvCalFile	**2Fh - 47**	Cannot open the specified cal file
Environmental Errors		
DerrMemLock	**30h - 48**	Cannot lock allocated memory from operating system
DerrMemHandle	**31h - 49**	Cannot get a memory handle from operating system
Pre-trigger acquisition Errors		
DerrNoPreTActive	**32h - 50**	No pre-trigger configured

API Error Codes		
DerrTooManyHandles	**60h - 96**	No more handles available to open
DerrInvLockMask	**61h - 97**	Only a part of the resource is already locked, must be all or none
DerrAlreadyLocked	**62h - 98**	All or part of the resource was locked by another application
DerrAcqArmed	**63h - 99**	Operation not available while an acquisition is armed
DerrParamConflict	**64h - 100**	Each parameter is valid, but the combination is invalid
DerrInvMode	**65h - 101**	Invalid acquisition/wait/dac mode
DerrInvOpenMode	**66h - 102**	Invalid file-open mode
DerrFileOpenError	**67h - 103**	Unable to open file
DerrFileWriteError	**68h - 104**	Unable to write file
DerrFileReadError	**69h - 105**	Unable to read file
DerrInvClockSource	**6Ah - 106**	Invalid acquisition mode
DerrInvEvent	**6Bh - 107**	Invalid transfer event
DerrTimeout	**6Ch - 108**	Time-out on wait
DerrInitFailure	**6Dh - 109**	Unexpected result occurred while initializing the hardware
DerrBufTooSmall	**6Eh - 110**	Unexpected result occurred while initializing the hardware
DerrInvType	**6Fh - 111**	Invalid acquisition/wait/dac mode
DerrArraySize	**70h - 112**	Used as a catch all for arrays not large enough
DerrBadAlias	**71h - 113**	Invalid alias names for Vxd lookup
DerrInvCommand	**72h - 114**	Invalid command
DerrInvHandle	**73h - 115**	Invalid handle
DerrNoTransferActive	**74h - 116**	Transfer not active
DerrNoAcqActive	**75h - 117**	Acquisition not active
DerrInvOpstr	**76h - 118**	Invalid operation string used for triggering
DerrDspCommFailure	**77h - 119**	Device with DSP failed communication
DerrEepromCommFailure	**78h - 120**	Device with EEPROM failed communication
DerrInvEnhTrig	**79h - 121**	Device using enhanced trigger detected invalid trigger type
DerrInvCalConstant	**7Ah - 122**	User calibration constant out of range
DerrInvErrorCode	**7Bh - 123**	Invalid error code
DerrInvAdcRange	**7Ch - 124**	Invalid analog input voltage range parameter
DerrInvCalTableType	**7Dh - 125**	Invalid calibration table type
DerrInvCalInput	**7Eh - 126**	Invalid calibration input signal selection
DerrInvRawDataFormat	**7Fh - 127**	Invalid raw-data format selection
DerrNotImplemented	**80h - 128**	Feature/function not implemented yet
DerrInvDioDeviceType	**81h - 129**	Invalid digital I/O device type
DerrInvPostDataFormat	**82h - 130**	Invalid post-processing data format selection
DerrdaqStalled	**83h - 131**	Personal Daq low level driver stalled
DerrdaqLostPower	**84h – 132**	Personal Daq device has lost power
DerrdaqMissing	**85h – 133**	Personal Daq device is not detected
DerrdaqScanConfig	**86h – 134**	Scan Configuration is invalid

Custom Labels D

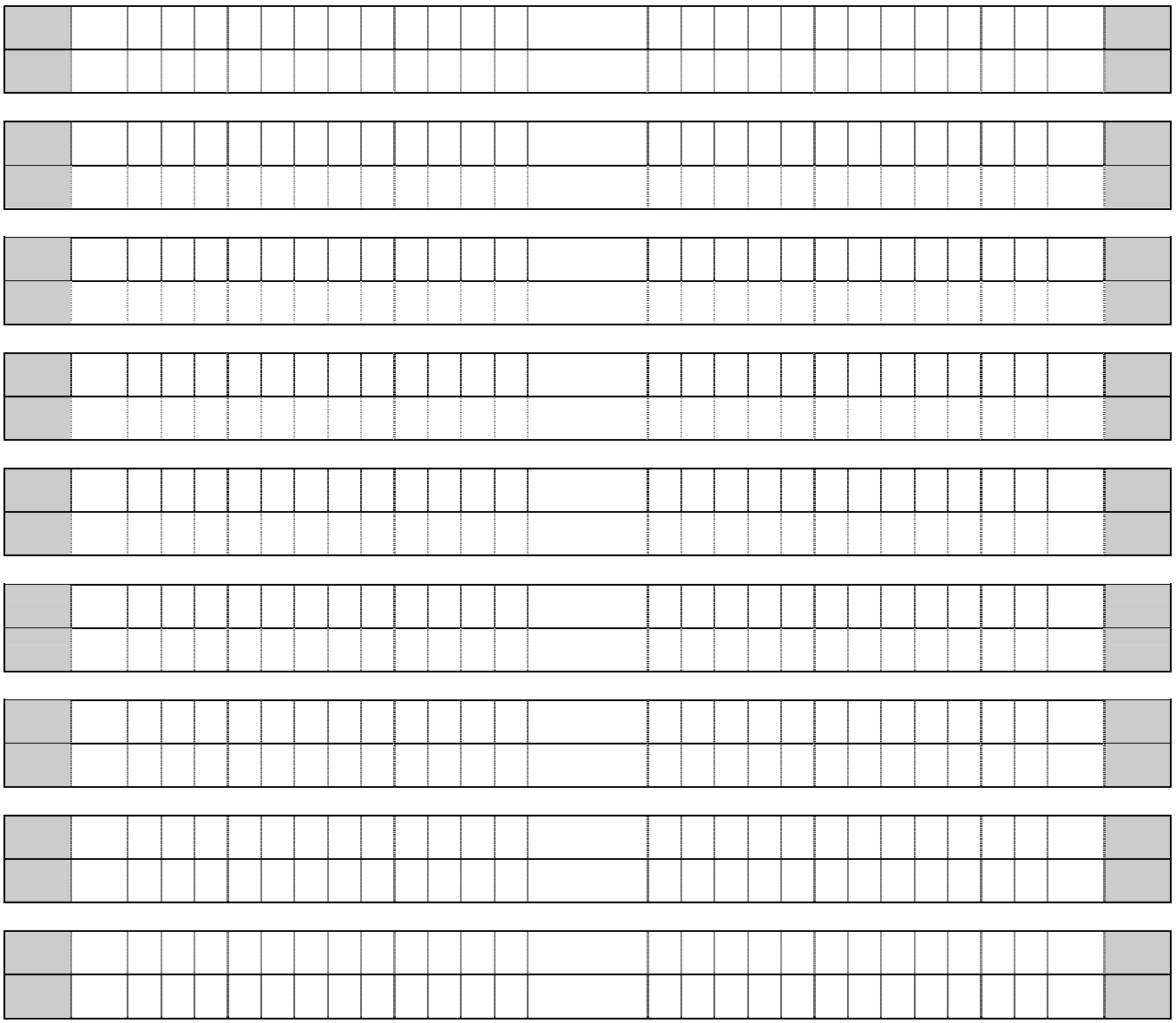
This appendix consists of blank user labels and a Personal Daq channel layout reference (page D-3). If you have access to *Microsoft Word*™ you can create custom labels in your PC since the labels exist in a file (**pDaq_CustomLabels.doc**) on your installation CD. During the installation process the file is automatically copied to the target directory for pDaqView. You can change the directory, if desired.

Note: You can use a shortcut entitled **Custom Labels Template** to access **pDaq_CustomLabels.doc**. The shortcut will be located in the *Personal DaqView* program group.

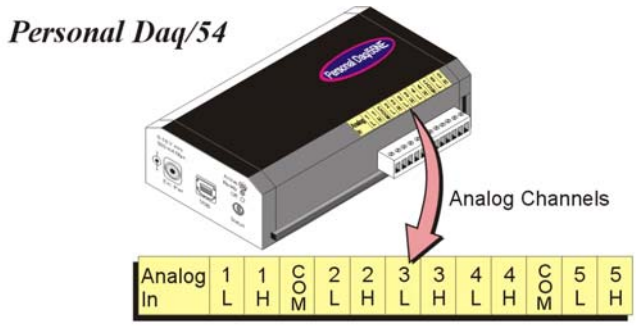
The labels in the file consist of tables with each cell set to scale. The default font is Arial, 6pt. centered. This can be changed, based on your specific labeling preferences.

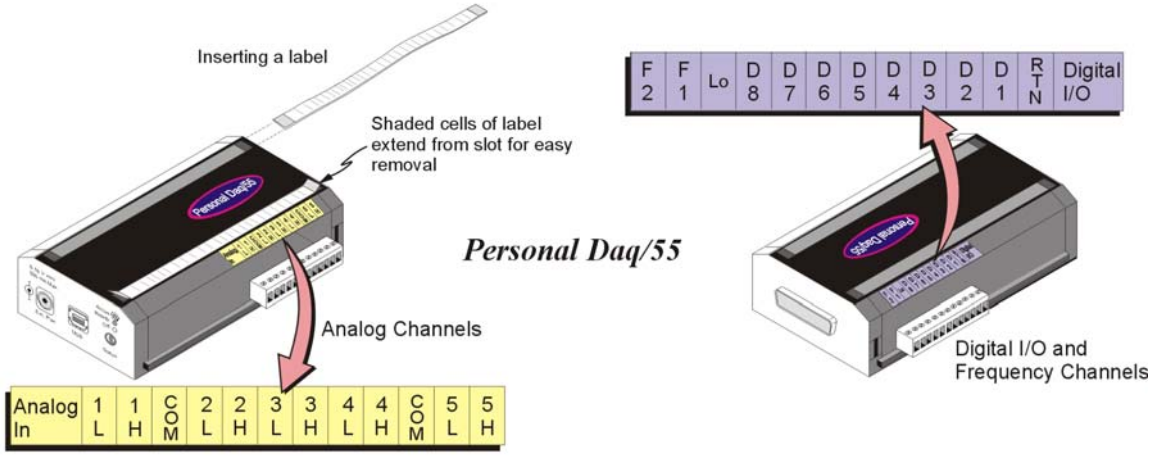
You can orient your label entrees horizontally, vertically, or upside down, as appropriate to your application. The important factor is that your labels correspond to the actual channel designations. These designations are clearly visible on the Personal Daq products, and are also illustrated on page D-3 for easy reference.

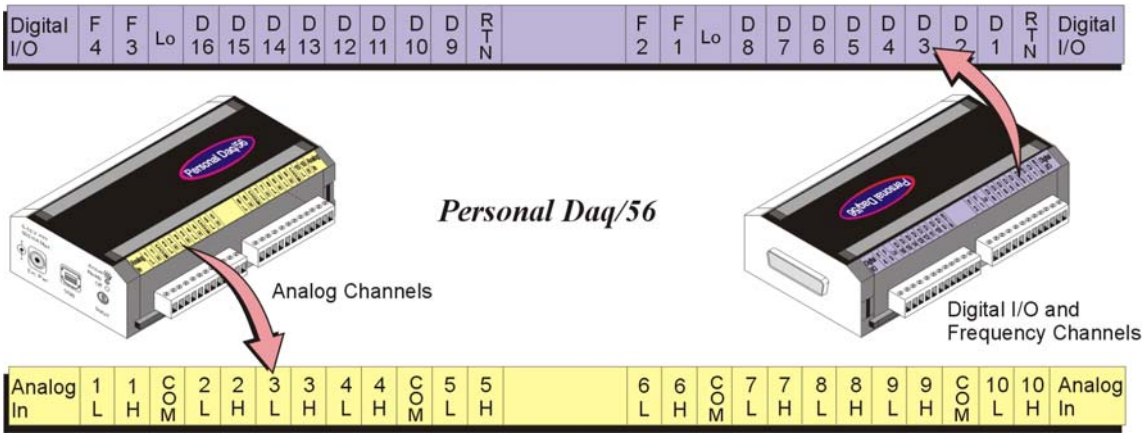
The labels are twice as wide as, and slightly longer than, the label slots on the Personal Daq device. The double-width allows you to fold the labels for extra rigidity. The extra length and rigidity make the insertion and removal of the labels easier. Note that when properly positioned, the shaded areas of a label will extend beyond the edge of the label slots.



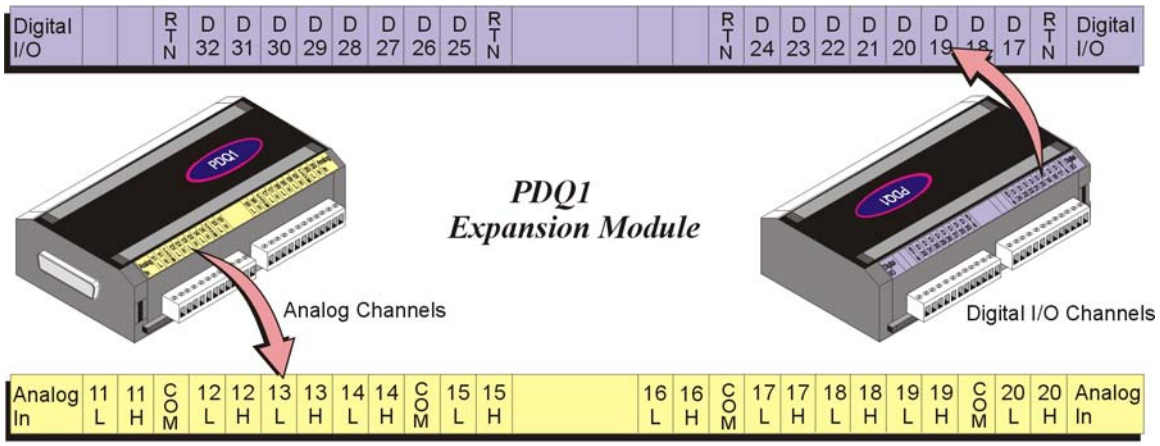
This side of the label page is intentionally blank.

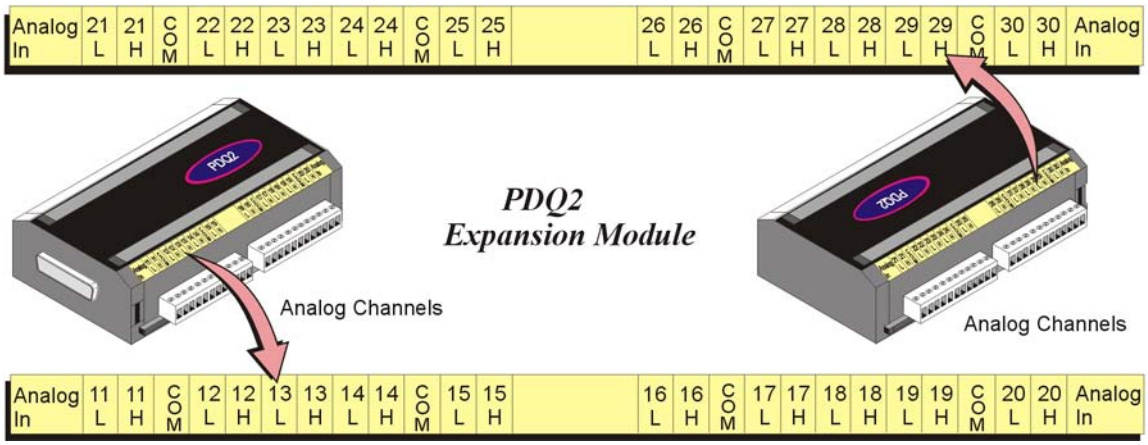






Channel Connection Layouts, Page 1 of 2





Channel Connection Layouts, Page 2 of 2

Artisan Technology Group is an independent supplier of quality pre-owned equipment

Gold-standard solutions

Extend the life of your critical industrial, commercial, and military systems with our superior service and support.

We buy equipment

Planning to upgrade your current equipment? Have surplus equipment taking up shelf space? We'll give it a new home.

Learn more!

Visit us at **artisantg.com** for more info on price quotes, drivers, technical specifications, manuals, and documentation.

Artisan Scientific Corporation dba Artisan Technology Group is not an affiliate, representative, or authorized distributor for any manufacturer listed herein.

We're here to make your life easier. How can we help you today?

(217) 352-9330 | sales@artisantg.com | artisantg.com

ARTISAN® TECHNOLOGY GROUP