Racal Instruments™

Model 1830 Source/Measure Switch System

User Manual

Publication No. 980937 Rev. A

EADS North America Test and Services,
a division of EADS North America, Inc.
4 Goodyear, Irvine, CA 92618
Tel: (800) 722-2528, (949) 859-8999; Fax: (949) 859-7139

info@eads-nadefense.com sales@eads-nadefense.com
helpdesk@eads-nadefense.com http://www.eads-nadefense.com

EADS NORTH AMERICA

Publication Date: August 13, 2009

THANK YOU FOR PURCHASING THIS EADS NORTH AMERICA TEST AND SERVICES PRODUCT

For this product, or any other EADS North America Test and Services a division of EADS North America, Inc. ("EADS North America Test and Services") product that incorporates software drivers, you may access our web site to verify and/or download the latest driver versions. The web address for driver downloads is:

http://www.eads-nadefense.com/downloads

If you have any questions about software driver downloads or our privacy policy, please contact us at:

info@eads-nadefense.com

WARRANTY STATEMENT

All EADS North America Test and Services products are designed to exacting standards and manufactured in full compliance to EADS AS9100B Quality Management System processes.

This warranty does not apply to defects resulting from any modification(s) of any product or part without EADS North America Test and Services express written consent, or misuse of any product or part. The warranty also does not apply to fuses, software, non-rechargeable batteries, damage from battery leakage, or problems arising from normal wear, such as mechanical relay life, or failure to follow instructions.

This warranty is in lieu of all other warranties, expressed or implied, including any implied warranty of merchantability or fitness for a particular use. The remedies provided herein are buyer's sole and exclusive remedies.

For the specific terms of your standard warranty, contact EADS North America Test and Services Customer Support. Please have the following information available to facilitate service.

1. Product serial number
2. Product model number
3. Your company and contact information

You may contact Customer Support by:

E-Mail:	Helpdesk@eads-nadefense.com	
Telephone:	+1 800 722 3262	(USA)
Fax:	+1 949 859 7139	(USA)

RETURN of PRODUCT

Authorization is required from EADS North America Test and Services before you send us your product or sub-assembly for service or calibration. Call or contact Customer Support at 1-800-722-3262 or 1-949-859-8999 or via fax at 1-949-859-7139. We can also be reached at: helpdesk@eads-nadefense.com.

If the original packing material is unavailable, ship the product or sub-assembly in an ESD shielding bag and use appropriate packing materials to surround and protect the product.

PROPRIETARY NOTICE

TRADEMARKS AND SERVICE MARKS

DISCLAIMER

Buyer acknowledges and agrees that it is responsible for the operation of the goods purchased and should ensure that they are used properly and in accordance with this document and any other instructions provided by Seller. EADS North America Test and Services products are not specifically designed, manufactured or intended to be used as parts, assemblies or components in planning, construction, maintenance or operation of a nuclear facility, or in life support or safety critical applications in which the failure of the EADS North America Test and Services product could create a situation where personal injury or death could occur. Should Buyer purchase EADS North America Test and Services product for such unintended application, Buyer shall indemnify and hold EADS North America Test and Services, its officers, employees, subsidiaries, affiliates and distributors harmless against all claims arising out of a claim for personal injury or death associated with such unintended use.

FOR YOUR SAFETY

Before undertaking any troubleshooting, maintenance or exploratory procedure, read carefully the **WARNINGS** and **CAUTION** notices.

This equipment contains voltage hazardous to human life and safety, and is capable of inflicting personal injury.

If this instrument is to be powered from the AC line (mains) through an autotransformer, ensure the common connector is connected to the neutral (earth pole) of the power supply.

Before operating the unit, ensure the conductor (green wire) is connected to the ground (earth) conductor of the power outlet. Do not use a two-conductor extension cord or a three-prong/two-prong adapter. This will defeat the protective feature of the third conductor in the power cord.

Maintenance and calibration procedures sometimes call for operation of the unit with power applied and protective covers removed. Read the procedures and heed warnings to avoid “live” circuit points.

Before operating this instrument:

1. Ensure the proper fuse is in place for the power source to operate.
2. Ensure all other devices connected to or in proximity to this instrument are properly grounded or connected to the protective third-wire earth ground.

If the instrument:

- fails to operate satisfactorily
- shows visible damage
- has been stored under unfavorable conditions
- has sustained stress

Do not operate until performance is checked by qualified personnel.

Table of Contents

This page was intentionally left blank.

List of Figures

This page was left intentionally blank.

List of Tables

This page was left intentionally blank.

EADS North America Test and Services

CE Declaration of Conformity

We

EADS North America Test and Services
4 Goodyear St.
Irvine, CA 92618

declare under sole responsibility that the

1830 Source/ Measure switch, P/N 1830-0102YYZZ
LXI 1170, 52ch SPDT, P/N 408161
LXI 1180, 80ch SPST, P/N 408162
LXI 1220, 16A, P/N 408163-001,-002,-003,-004
LXI 1380, 2W 8 1x8 Mux, P/N 408164
LXI 1450, Matrix, P/N 408175, 408176
DMM, 7.5 Digit, 1830, LXI, 408186

conforms to the following Product Specifications:

Safety: **EN 61010-1:2001**

EMC: **EN61326:2006 CLASS B**

Supplementary Information:

The above specifications are met when the product is installed in an EADS North America Test and Services certified enclosure, with faceplates installed over all unused slots, as applicable.

The product herewith complies with the requirements of EN 61010-1:2001 and EN61326:2006 CLASS B

Irvine, CA, July 6, 2009

David Johnston, Engineering Manager

This page was left intentionally blank.

DOCUMENT CHANGE HISTORY

Revision	Date	Description of Change
A	8/13/09	Document Control release

This page was left intentionally blank.

Chapter 1
Overview and Features

Overview

Figure 1-1: Model 1830 Front Panel View

The Model 1830 is a high-performance source/measure switch providing an integrated environment with DMM source/measure capability and high-density switching

This 3U-sized, rack-mountable package accommodates up to nine plug-in cards for switching, instrumentation, and digital I/O in order to provide a scalable switching solution.

The 1830 allows for

- Quick and easy device configuration through a web interface
- LXI™ compliant remote control and monitoring through your Ethernet connection
- High-level compatibility with our provided ActivATE™ test executive software
- Simplified programming and greater software reuse with other leading test software though our IVI™ drivers
- Ability to create hybrid systems including Ethernet, GPIB, and USB
- Easily configurable, plug-in cards to meet your switching, source, and measurement needs
- Fast setup with no configuration jumpers or switches to set. The 1830 is ready to use as soon as the plug-ins have been inserted.

Powerful Design

The 1830 provides extensive hardware and software capabilities including:

- **Self-Test.** Ensures that critical system components are functioning correctly. This test executes automatically at power-up.
- **Power-Up Recall.** Automatically recalls a complete switching system configuration from non-volatile memory at power-up.
- **Non-Volatile Memory.** Stores and recalls switch configurations and 1830 user preferences (such as GPIB address).
- **Confidence Mode.** Automatically verifies correct application of voltage to relay coils.
- **Path-Level Switching.** Allows complex paths to be defined with open and close elements, stored in non-volatile memory, and be controlled remotely or through the front-panel display.
- **Verification.** Allows you to verify the status of all switches.
- **External Trigger Input.** Allows an external device to trigger switching operations. The external device connects to the *External Trigger In* connector.
- **External Trigger Output.** Provides a pulse on the *External Trigger Out* connector to trigger an external device, such as a DMM, after the relay states have changed.
- **Equate Lists.** Reduces programming effort. When two or more cards are on an equate list, they respond simultaneously to a command sent to any of them.
- **Exclude Lists.** Allows you to prevent simultaneous closure of switches in a specified group. The switches are then mutually exclusive.
- **Scan Lists.** Allows you to specify a sequence of relay closures to step through in response to an external trigger source. A scan list may include relays on one card or multiple cards.
- **Programmable Delay.** Allows you to specify the amount of time delay from a relay state change to an external trigger pulse.
- **Relay Lifetime Counters.** Non-volatile memory accumulates relay switch operation to allow for increased reliability and maintenance support.
- **Internal Signal and Differential Configured Analog Buses**
- **10 MHz Reference Input**

Signal Raceway Analog Bus

The heart of the 1830 includes a multi-path Signal Raceway analog bus (Figure 1-2) which provides a high-quality route for the connection of instruments to the unit or system under test. The tight integration between instruments and switching preserves signal integrity, simplifies connections, and reduces external cabling.

The 1830 analog bus backplane architecture directly connects installed switching and instrument assets to reduce or eliminate inductance and capacitance in the stimulus or measurement path.

The 1830 system's parallel architecture provides high throughput with up to four factory-installed instruments simultaneously measuring multiple signals on multiple units-under-test (UUTs). The nine user-accessible expansion slots provide over 500 multiplexed channels enabling a high-volume, compact test system for engineering or manufacturing.

The scalable architecture allows for matrices (up to 4x324 or 8x162), multiplexers (up to 1x576), and source/measure cards (up to 4); all with internal connectivity through the analog bus.

For specific characteristics and specifications, view the Analog Bus section in **Appendix A, Specifications**.

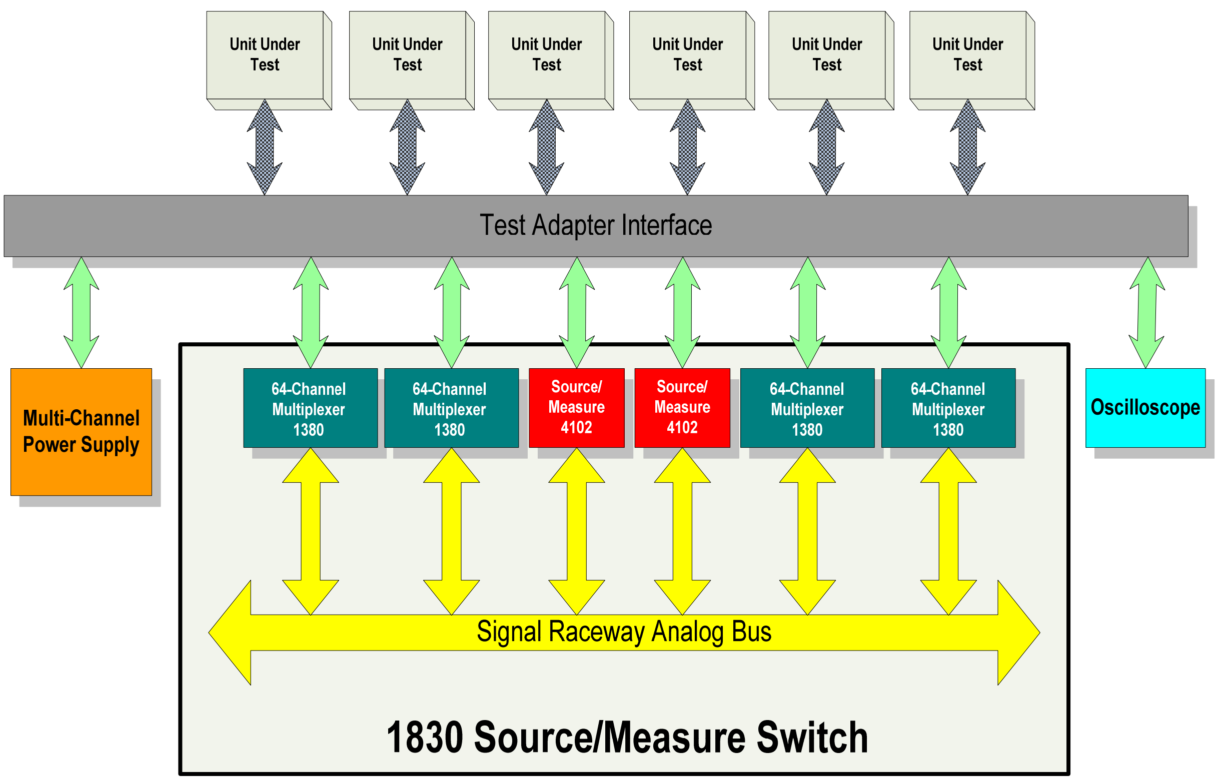


Figure 1-2: 1830 Signal Raceway Analog Bus Block Diagram

The 1830 Analog Bus (Figure 1-3) consists of a differential analog bus (4 channels) and a single-ended analog bus (8 channels). The analog buses are further separated into two banks: Bank A for slots 1 through 5 and Bank B for slots 6 through 9. Separating the banks allows for signals to be isolated for use between the grouped slots. Closing the analog bus relays allows for busing all of the slots together.

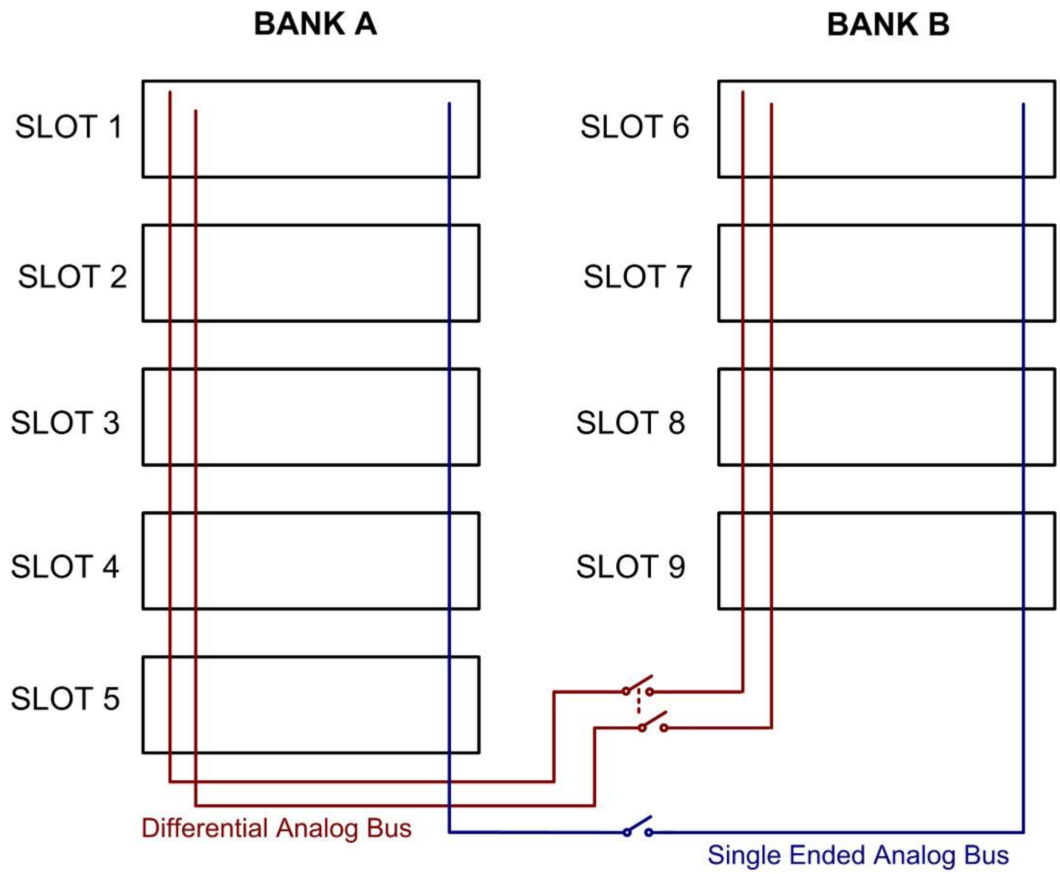


Figure 1-3: Analog Bus – Single-ended and Differential Channels

Caution

Do not exceed the maximum voltage and current ratings of the buses. Care must also be taken to not exceed the ratings of the cards connected to the analog bus.

RF Performance

In order to maximize the bandwidth performance of the analog bus, the slots selected for use should minimize the analog bus undesired stub lengths. Figure 1-4 shows a slot configuration with a card installed in Slot 6 connecting with Slot 9. With the differential analog bus open and selected slot positions, the maximum analog performance is realized.

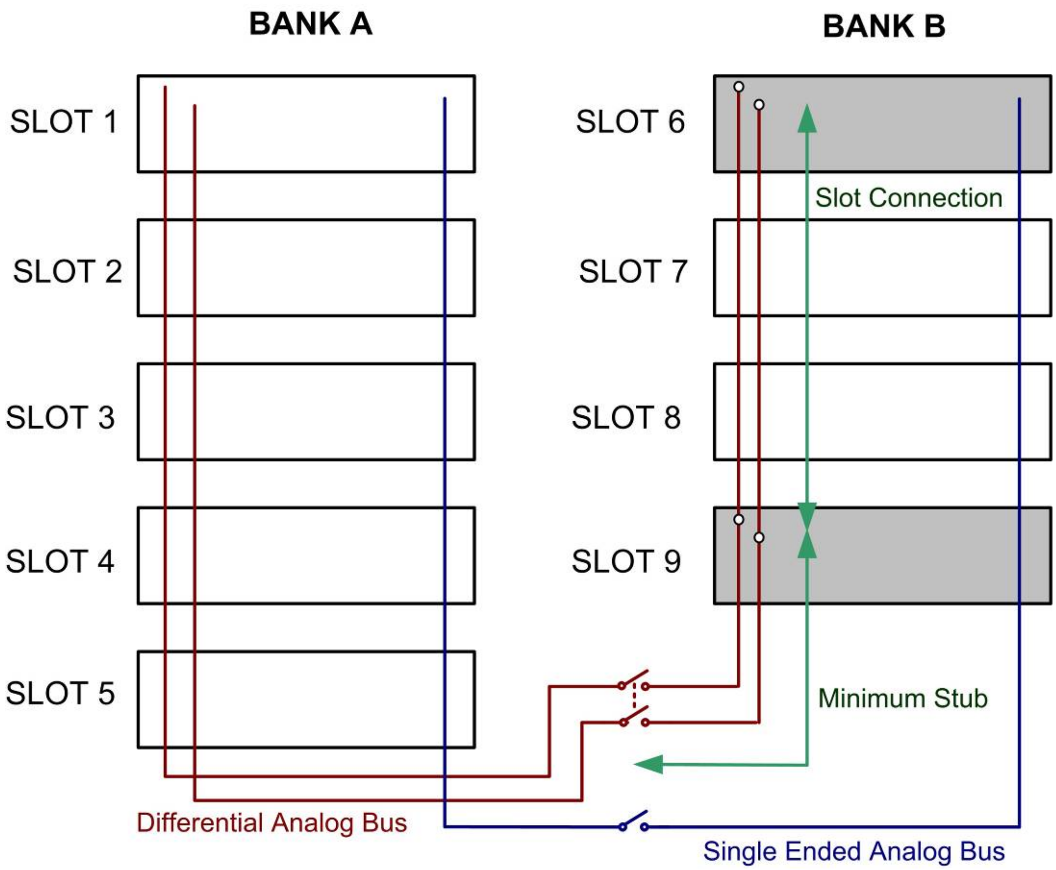


Figure 1-4: Analog Bus – Minimum Stub, Maximum Bandwidth

The minimum analog bus RF performance occurs when the stub length is maximized. In this case (Figure 1-5), 2 cards are installed in Slots 6 and 7 and the differential analog bus is closed. This results in a long stub from Slot 7 all the way to Slot 1 and a minimum achievable analog bandwidth.

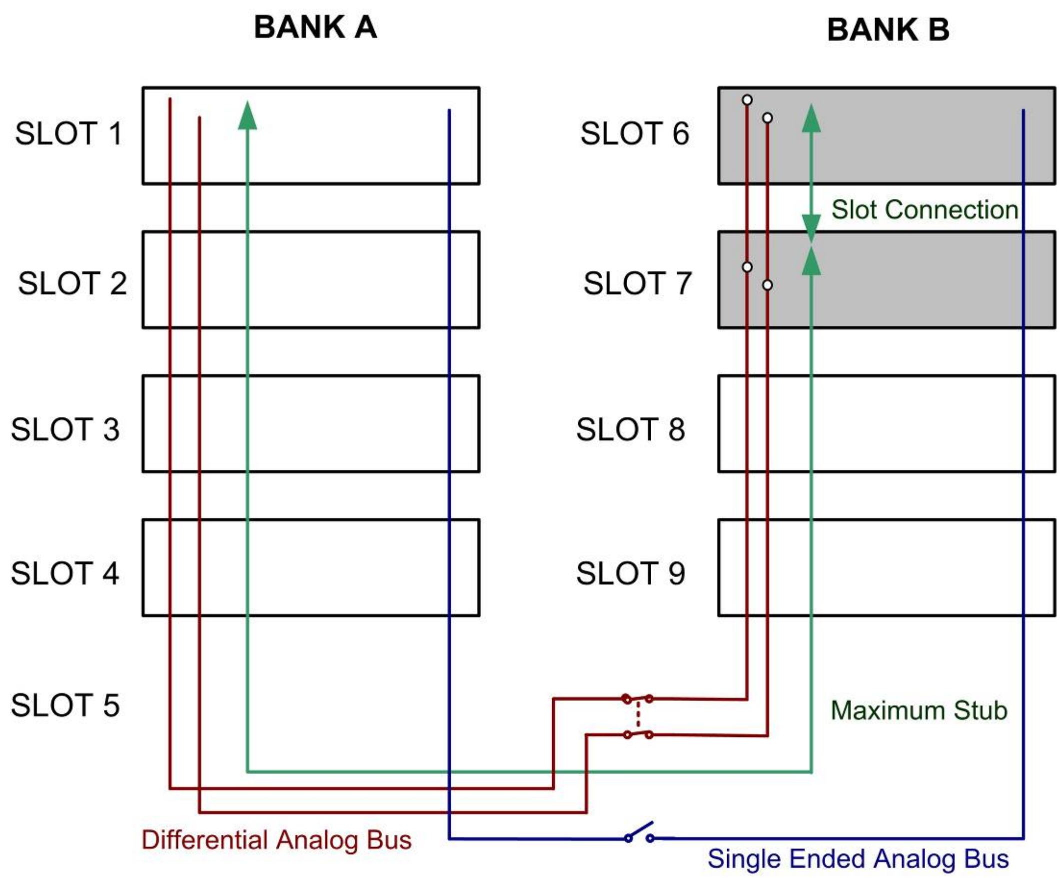


Figure 1-5: Analog Bus – Maximum Stub, Minimum Bandwidth

Trigger Router

The trigger router is a fully configurable trigger input and trigger output routing matrix. The trigger router allows 16 trigger inputs. Nine inputs are from the nine card slots. Four are from four internal PXI card slots. One input is from a Scan Sequencers, and two are from two external trigger inputs located on the rear panel. Refer to Figure 1-6 which shows the Trigger Router matrix.

Every trigger input can be configured through four detection modes: active high, active low, level high, and level low. Every trigger input can be routed to every trigger output. Every trigger output can be configured for input to output delay, output edge active high or active low, output level high or level low, trigger pulse width, trigger repeat cycles (auto or input triggered), and trigger scan interval timer. Trigger scan interval can be configured up to 1276 hours. Any trigger output can also be configured to be triggered on single trigger, multiple triggers, or auto trigger.

Additional details on the trigger router, including a sample SCPI program to set the trigger, can be found in **Chapter 6, Using the 1830**.

Trigger settings can be made by using the web-page interface or through SCPI commands. Information regarding these methods can be found in **Chapter 4, Web-Page Interface**, and **Chapter 7, SCPI Command Basics**.

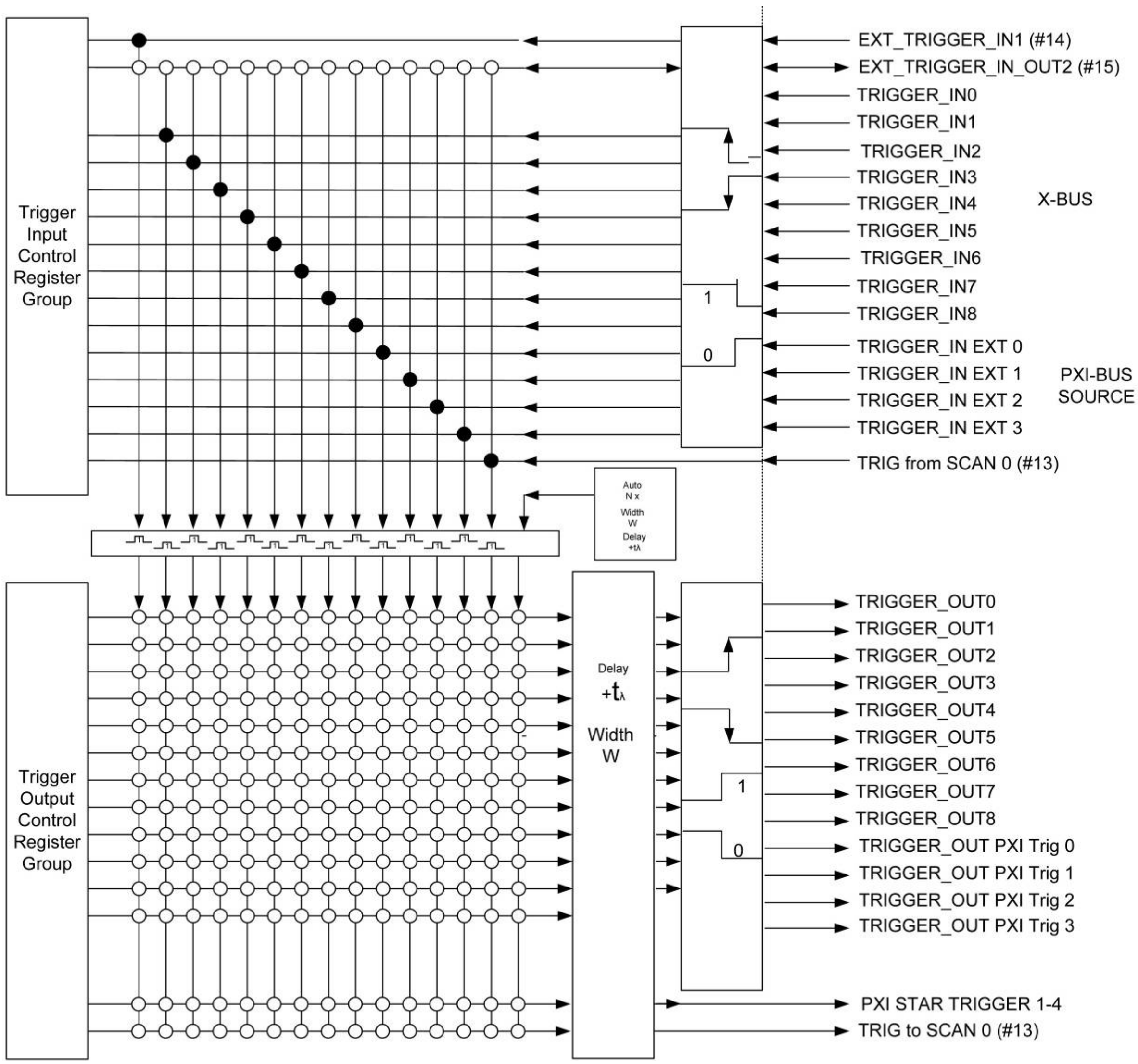


Figure 1-6: Trigger Router Matrix

Front Panel USB, Indicator Lights, and System Reset

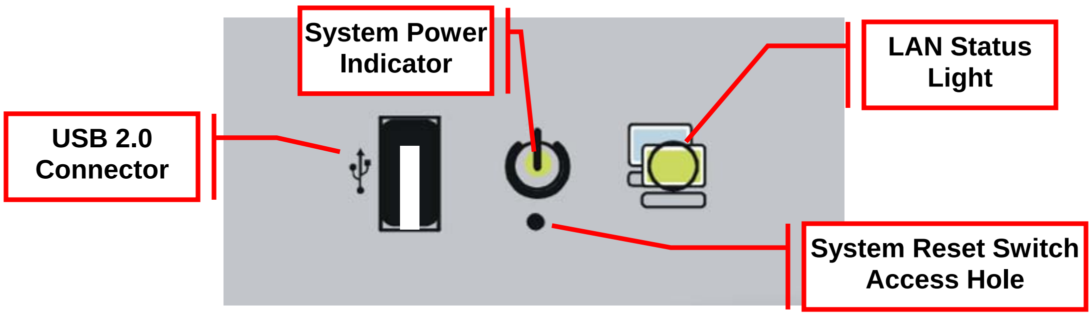


Figure 1-7: Front Panel USB and Indicator Lights

The front panel includes a USB connector, a system (1830) power indicator, and a LAN status light. The status light glows green to indicate a working LAN connection and red to indicate a faulty or unconnected LAN connection. Refer to the **Device Identify Button** section in **Chapter 4** for briefly using the LAN status light to identify a particular 1830 in a bank of 1830 systems.

The front panel USB connector is a USB 2.0, full-speed, Type A port and can be used with a USB thumb or flash drive for external memory storage of data, scan lists, etc.

A system reset switch is accessed through a small hole underneath the system power indicator light. Pressing this (with something like a paperclip) turns the power off and then back on. You may experience a change in IP address when resetting your 1830 if attached to a networked system.

Rear Panel

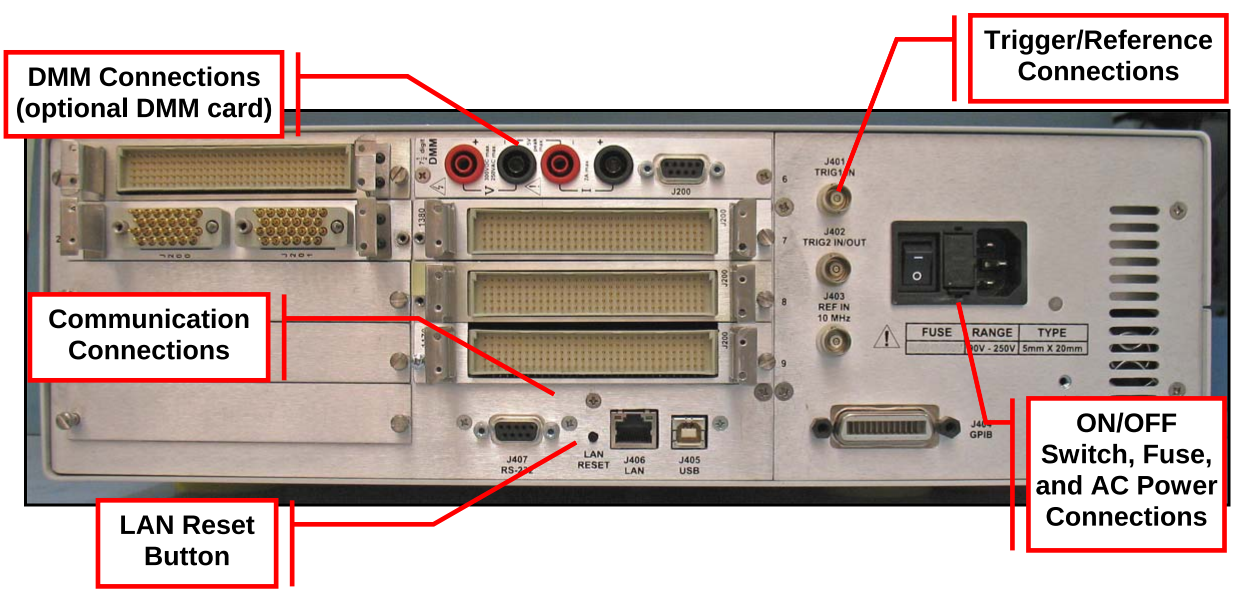


Figure 1-8: Rear Panel

When switch plug-in cards are referenced in this manual, the "front panel" of the card is the outside connector end (as seen in Figure 1-8).

Communication Connections

The 1830 allows several different communication options depending on your application.

- LAN (LXI, Class C compliant)
- IEEE-488.2 (standard GPIB)
- USB 1.1 full-speed, Type B port

The RS-232 connector is a debug port for factory servicing purposes only.

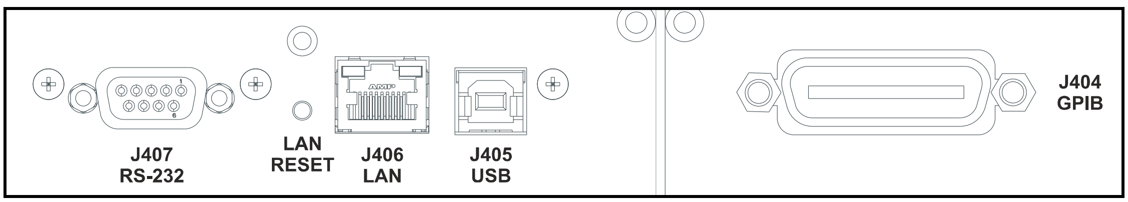


Figure 1-9: Communication Connections

The rear panel USB connector is used as a communication interface.

If you wish to use a USB connection to attach a USB thumb or flash drive for memory purposes, use the USB connector on the front panel.

LAN Reset Button

The LAN Reset button is used to reset the LAN settings of the 1830 to a factory default setting which includes automatic IP address configuration and password reset.

To activate the LAN reset, press the button for a minimum of five seconds. When released, the LAN status light on the front panel momentarily turns off and then turns back on to indicate the LAN status.

You may need to rediscover the LAN IP address (see **Chapter 3, Discovering and Configuring your 1830 on a LAN**).

For initial password setting information, see **Chapter 4, Web-Page Interface**.

Trigger and Reference Connectors

Figure 1-10: Trigger and Reference Connectors

Power Panel

The AC input is 90-250 V, 47-63 Hz, 600 VA max. The two 5 Amp, 250 V fuses are accessed by removing the fuse cover.

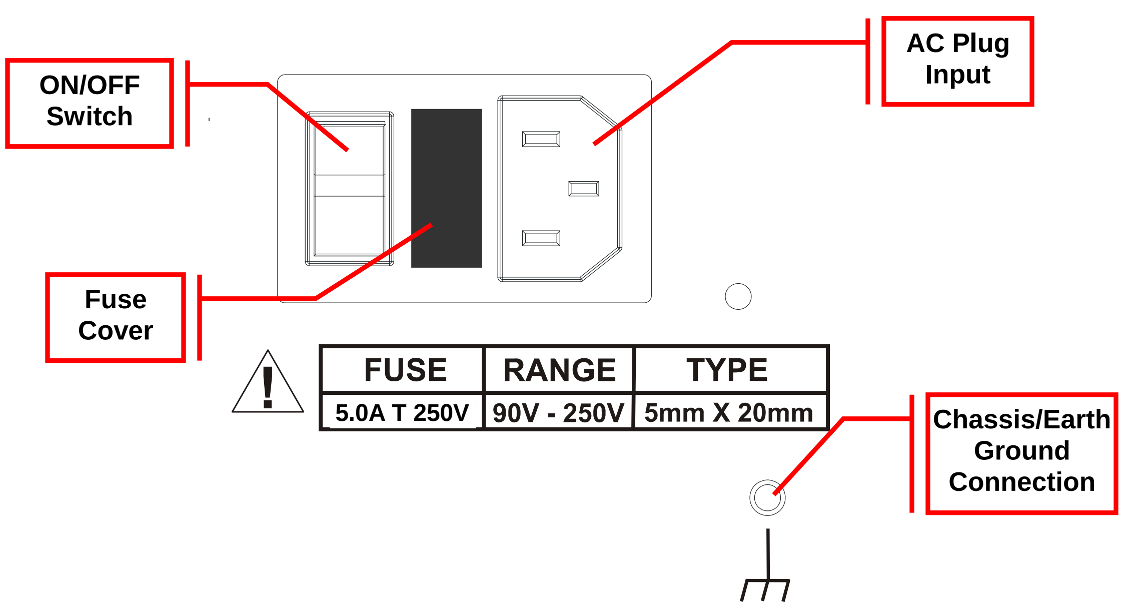


Figure 1-11: Power Panel

The hole for the chassis/earth ground is threaded for a 6-32 screw/bolt. Do not allow the bolt to extend more than 1/4" inside the chassis itself.

Safety Precautions

Be sure to position the 1830 to allow easy access to the power switch on the rear panel. This switch completely disconnects both sides of the AC power mains.

Warning

Use only AC power outlets that have a protective ground. DO NOT USE a two-conductor extension cord or a 3-prong to 2-prong adapter that does not provide a protective ground connection.

The rear-panel AC input socket provides a protective ground terminal. The 1830 is supplied with a detachable 3-conductor power cord that connects to this protective terminal. Only this cord, or equivalent, should be used. Connection of the power cord to the power outlet must be made in accordance with the following standard color code:

Table 1-1: Power Outlet Standard Wire Color

Function	Power Outlet Wire Color	
	American	European
Line (Live)	Black	Brown
Neutral	White	Blue
Ground (Earth)	Green	Green (Yellow)

Warning

Do not connect the 1830 or any of the plug-in cards directly to a mains power outlet without using some sort of power or signal conditioning equipment. High-voltage transients may damage the instruments if they are not regulated.

Caution

Be sure to operate the 1830 with sufficient air flow to the instrument. Do not block the vents!

The 1830 conforms to IEC 61010 safety requirements for electrical equipment.

Analog Bus Safety Interlock Circuit

Some cards in the 1830 system (such as the 1380 and 1450) feature an analog bus safety interlock circuit which prevents the card connection to the 1830 Analog Bus and potential high voltage exposure on the outside connector. Refer to the specific card manuals for additional details.

Emergency Reset Feature

Some cards in the 1830 system (such as the 1220) also feature an emergency reset feature to open all relays quickly in case of a problem, for instance high current. Refer to the specific card manuals for additional details. **Chapter 4, Web-Page Interface** and **Chapter 6, Using the 1830**, include information about how to respond in case of a reset condition.

Chassis Identification Label

The identification label on the side of the 1830 chassis includes the part number and build revision level, the serial number, and the chassis Media Access Control (MAC) address.

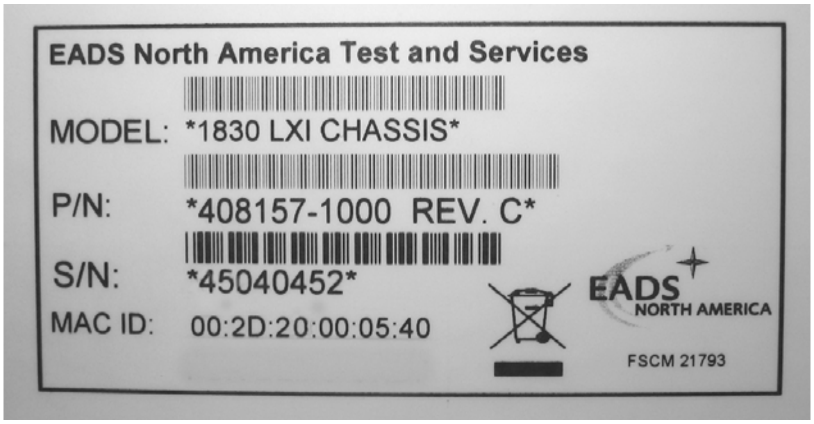


Figure 1-12: Chassis Identification Label

Dimensions

Figures 1-13 and 1-14 show front and side views of the 1830 chassis and their dimensions.

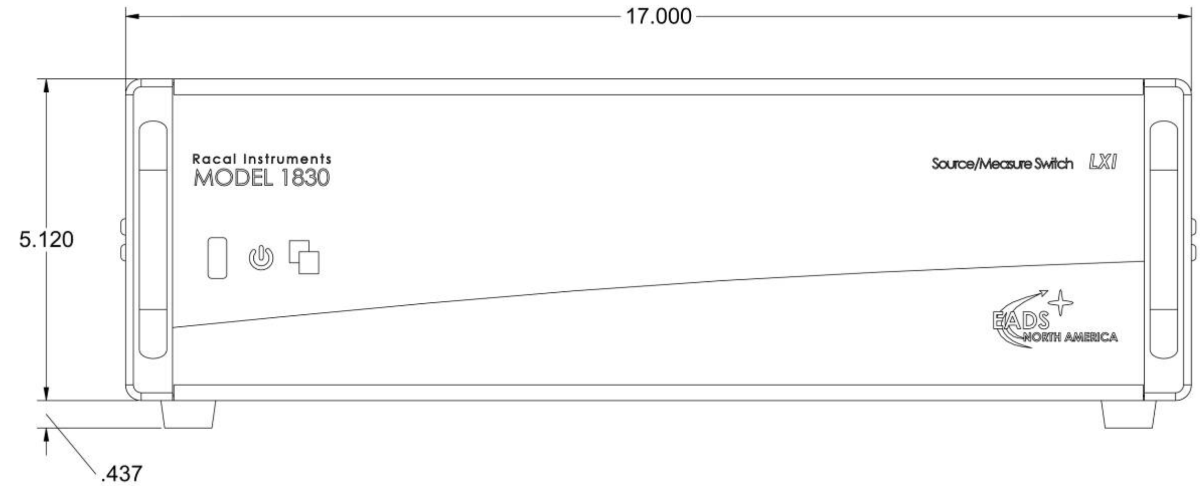


Figure 1-13: 1830 Front Dimensions

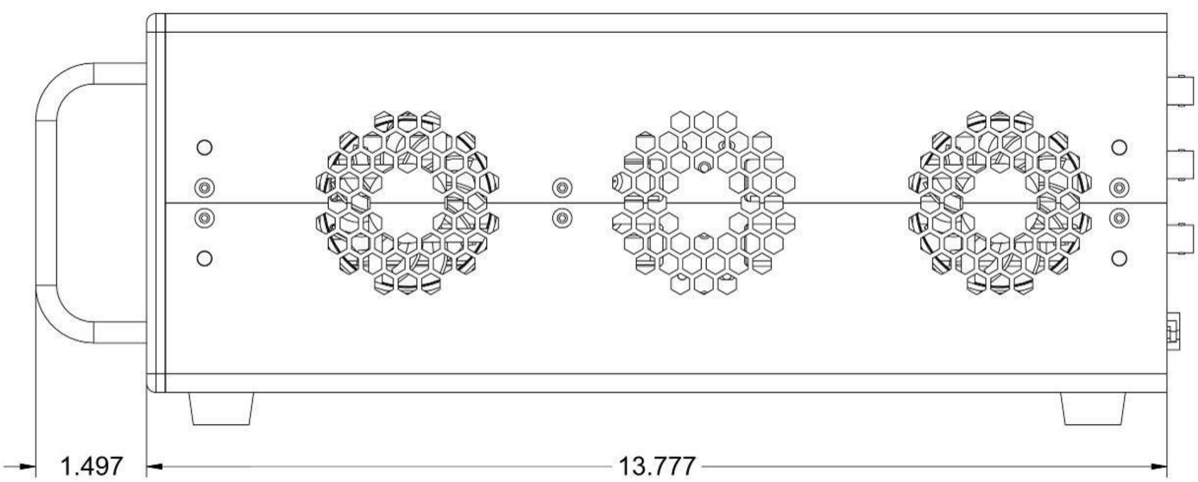


Figure 1-14: 1830 Side Dimensions

LAN eXtensions for Instrumentation (LXI) Compliant

The 1830 Source/Measure Switch System is LXI Class C compliant.

Web-Page Interface

The 1830 has a built-in web-page interface to allow full access and control of the instrument and plug-in cards. The system controls functions require you to have a recent version of the Java™ engine installed on your computer.

The web-page interface provides information regarding VISA resource strings, Refer to **Chapter 4, Web-Page Interface** for further information.

ActivATE Test Platform Software

The 1830 works seamlessly with our ActivATE test software. A 90-day full-featured evaluation copy of the software is included with this product. Dedicated drivers for the 1830 and its plug-in instrument cards (modules), written in C# .NET, are also included to use with your configured 1830 system. The ActivATE test software allows you to auto-configure the environment for the correct 1830 driver set that matches your hardware configuration.

Refer to **Chapter 5, ActivATE Test Platform** for additional information.

Accessories

Included with the 1830

Table 1-2: 1830 Included Accessories

Description	Part Number	Quantity
Product documentation and software disk includes Model 1830 user manual as well as manuals and drivers for all currently released switch cards and DMMs	922357	1
AC power cable	600620	1
Blanking plates	457700	4

Optional Accessories

Table 1-3: 1830 Optional Accessories

Description	Part Number	Quantity
Rack mounting kit	408174	1
European power cord (unterminated)	407730	1
Card shield kit	408190	7

Table 1-4: 1830 Plug-in Card Cable and Connector Accessories

Description	Part Number
160-pin cable assembly, 6 ft *****	408191
160-pin connector kit with backshell and pins *****	408192
26-pin cable assembly, 6 ft	408200
26-pin connector kit	408201
Crimp hand tool	991020
Crimp pin-insertion tool	990898
Pin-removal tool	990899

***** Included with both the 160-pin connector kit and the 160-pin cable assembly is a set of hardware which you can use to key code your connector to specific plug-in cards in order to prevent cables from being misconnected to the wrong card. You may want to install the key for specific card types or to match module operating conditions..

Detailed installation information is located in the manuals of the cards using those cables.

The page was left intentionally blank.

Chapter 2
Getting Started

This chapter includes:

- Mounting options
- Main AC power requirements
- Installing plug-in cards (modules)
- Using the installation product documentation disk
- IVI drivers
- VISA resource strings
- Setting the GPIB address
- Communicating with the 1830 using GPIB and USB connections
- Connecting with the 1830 over a LAN connection

Mounting Options

You may use the 1830 Source/Measure Switching System on a table or mounted in a standard 19-inch equipment rack. See **Appendix C, Rack-Mount Installation** for instructions on the installing the 1830 into a rack.

Main AC Power

The 1830 system requires a power input of 90 VAC to 250 VAC, 47 Hz to 63 Hz, 600 VA max. The 1830 requires no adjustments, settings, or changes in fuse capacity to accommodate variations of voltage and frequency within these ranges. Simply connect the 1830 AC power connection to the power source. To replace a fuse, remove the fuse cover to access the fuse.

Installing Plug-In Cards

Make sure the power to the 1830 is turned **off** before installing or removing the plug-in cards (modules).

Warning

Inserting or removing plug-ins with power on may cause damage to the 1830 and to the plug-ins.

1. Slide the card into the 1830 chassis. When inserting the card, place the card edges into the card guides of the 1830. (See Figure 2-1.)
2. Carefully slide the card into the chassis until it stops.
3. Firmly push the card to engage its rear connectors with the 1830 backplane. When the plug-in card is fully seated, its panel is flush with the rear panel of the 1830 chassis.
4. Carefully turn the card retainer screws and tighten securely.

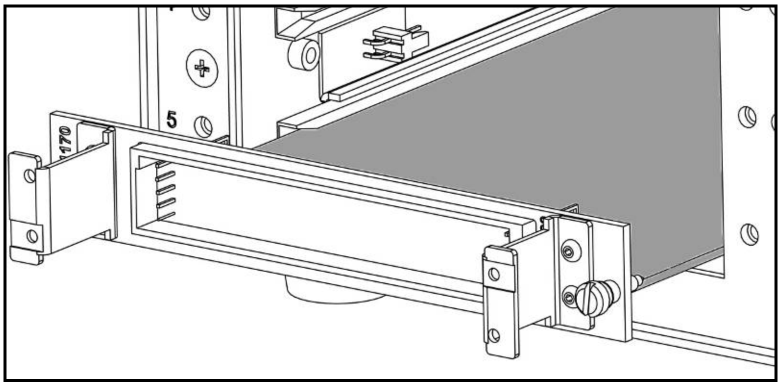


Figure 2-1: Inserting a Plug-In Card

Plug-in Card Shield

If you wish to reduce the electromagnetic interference (EMI) between cards, install an optional card shield (PN 408190) immediately above the cards. Refer to the Figure 2-2.

1. If already installed, remove the plug-in card you wish to cover.
2. Slide the card shield into the slot above the card slot.
3. Use the supplied flat-head screw to attach the shield to the chassis frame.
4. Install the card below the shield.

Caution

Do not move or remove the DMM interface card. This will compromise system safety and integrity. This card is factory serviceable only.

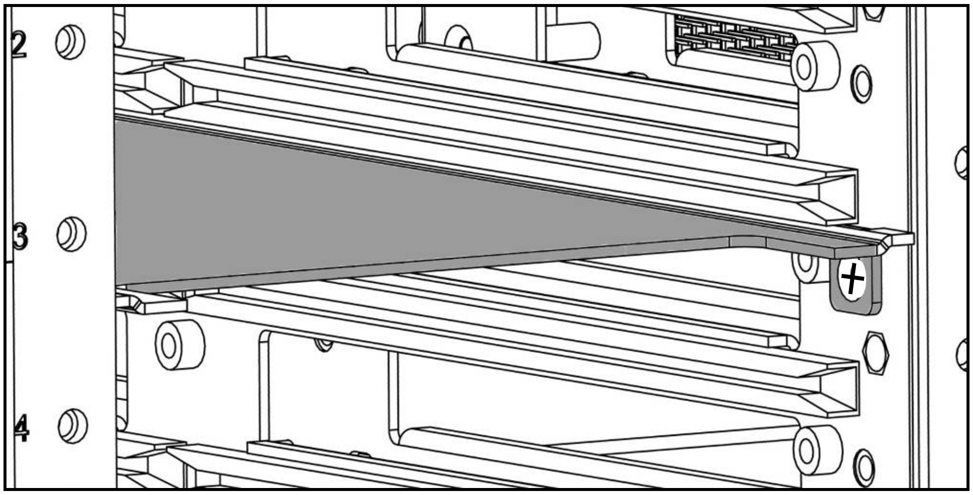


Figure 2-2: Installing a Card Shield

Coding Keys

Included with both the 160-pin connector kit and the 160-pin cable assembly is a set of hardware which you can use to key code your connector to specific plug-in cards in order to prevent cables from being misconnected to the wrong card. You may want to install the key for specific card types or to match module operating conditions..

Detailed installation information is located in the manuals of the cards using those cables (such as the 1170 and 1380).

Slot Numbering

The 1830 automatically references each plug-in card by the number of the chassis slot into which it is installed. There are no additional configuration jumpers or settings. See Figure 2-3 for slot numbering.

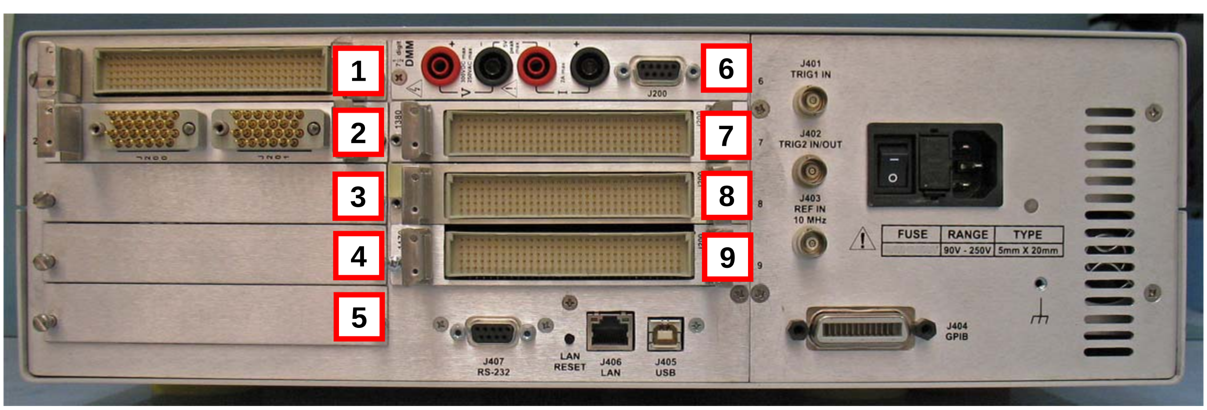


Figure 2-3: Numbering of Plug-In Slots

Installation Disk

The product documentation disk included with your system has downloadable software, device drivers, and user manuals for the 1830 system and all of the currently-available switch cards.

Note: If you purchase another card in the future and can not locate your product documentation disk, you can easily download the manual at our website, http://www.eads-nadefense.com/downloads.

1. Insert the Product Documentation disk (PN 922357) into the CD drive of your computer.

   A launch page appears (Figure 2-4).

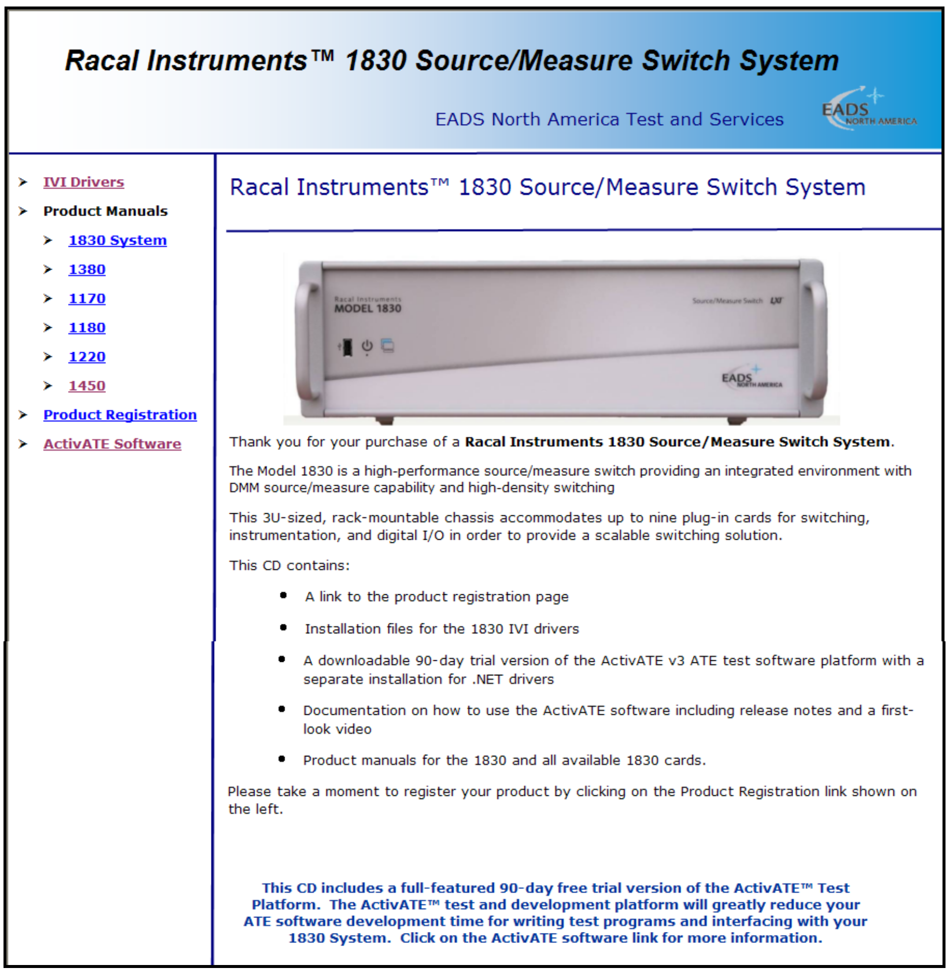


Figure 2-4: 1830 Install Disk Launch Page

2. Choose which action you would like to continue with.

   - If you would like to register your 1830 system, select the **Product Registration** link on the left side of the page.

     This link connects online to a website where you can register your owner and product information (Figure 2-5). Registering the 1830 and supplying your contact information is not required for use, but it is highly recommended for contact about future updates.

- If you would like to review the 1830 or switch card manuals, select the link for the particular manual you're interested in.

  The separate files are in the Adobe PDF format and need Adobe Reader to view. We've included linked table of contents and indexes to help you locate the information you are seeking. You can also perform a search of the PDF file.

- If you would like to install the IVI drivers, select the **IVI Drivers** link.

  IVI drivers are available for the 1830 system and switch cards (Figure 2-6). Before you download the drivers, be sure you have installed the IVI shared components. Refer to the section, **Installing the IVI Drivers**, later in this chapter, for more information.

- If you would like to install the ActivATE test software, select the **ActivATE Software** link.

  The 90-day, full-functioning, free trial of ActivATE test software (Figure 2-7) installs any required .Net connection software components it may need to run. Refer to **Chapter 5, ActivATE Test Software**, and the embedded help files in the software for additional information on using the ActivATE solution.

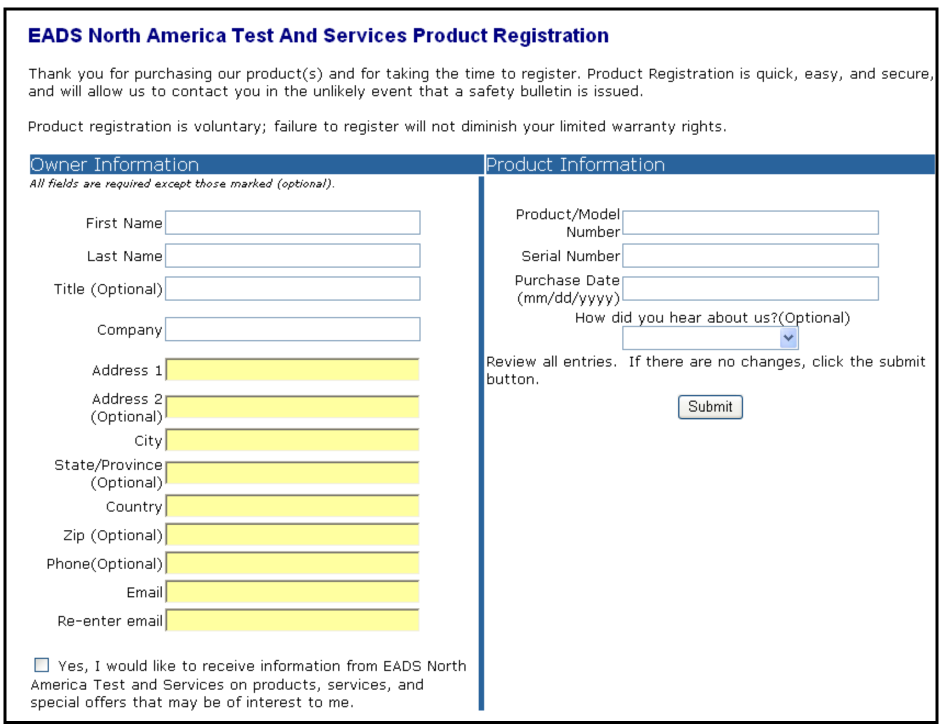


Figure 2-5: Online Registration Page

The IVI Driver download page (Figure 2-6 and 2-8) includes links to install drivers for the

- 1830 system
- 1170 card)
- 1180 card
- 1220 card (all versions)
- 1380 card
- 1450 card (all versions)
- DMM module (all versions)

There is also an Internet link to the IVI Foundation website for downloading the latest shared components.

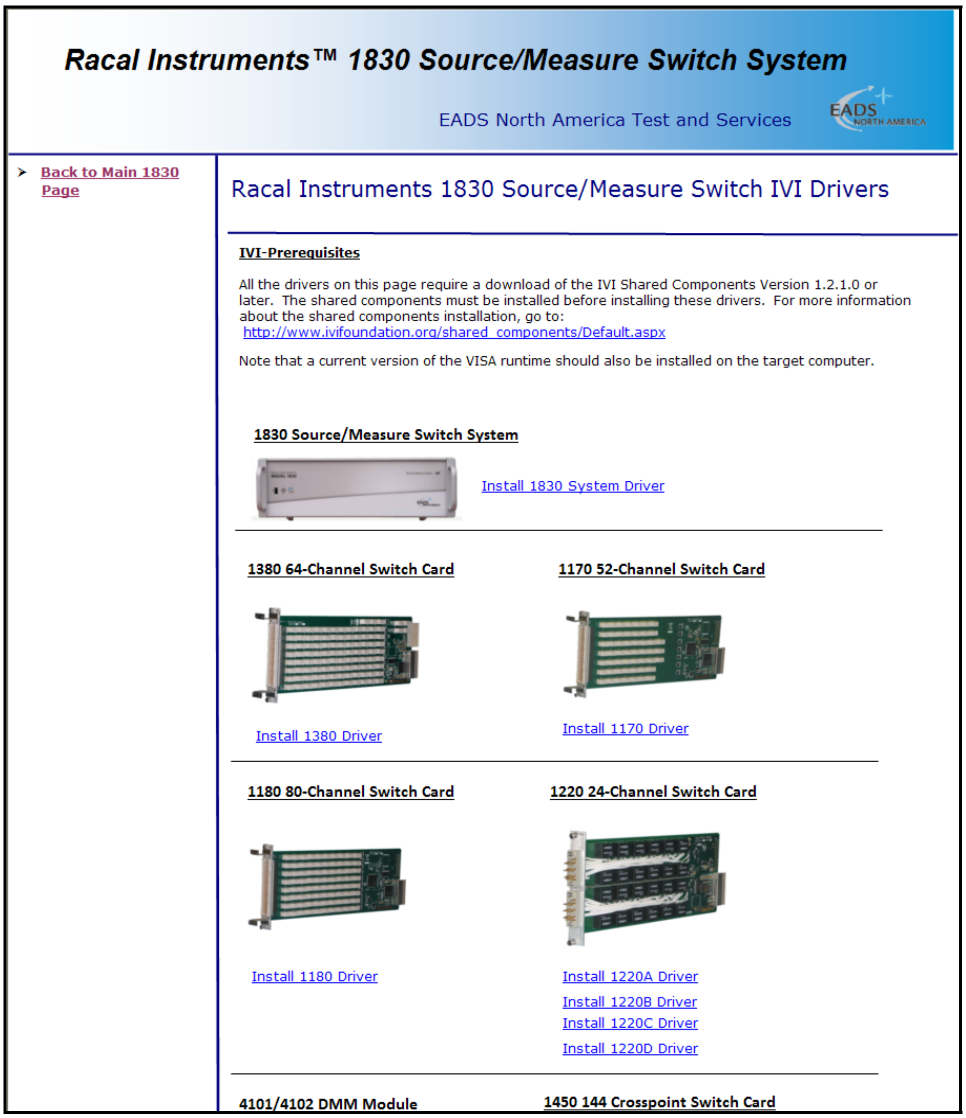


Figure 2-6: IVI Drivers Download Page

The ActivATE software startup page (Figure 2-7) includes links for

- Release notes (a .htm file launching in your internet browser)
- Installation file for the ActivATE software
- Installation file for the 1830 system and card drivers
- ActivATE help files (a .chm file which you need to save on your hard drive and then reopen separately)
- “First look” overview video (an executable file that launches into a short, informative video about the ActivATE software features and use)
- ActivATE software configuration video (an executable file that launches into a short informative video to help you configure the ActivATE software on the 1830 system)
- Internet link to the ActivATE software website

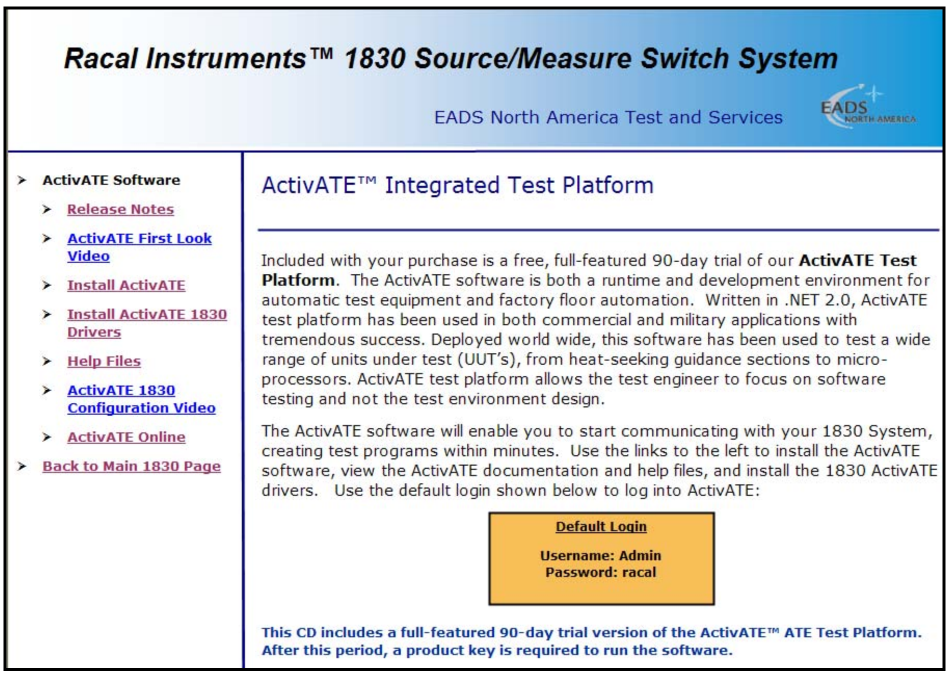


Figure 2-7: ActivATE Software Startup Page

IVI Drivers

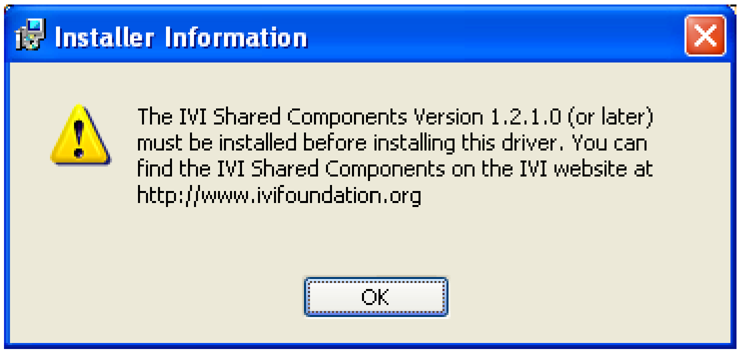
On the IVI Drivers page (Figure 2-8), select and install the drivers for the 1830 system and the switch cards you will be using.

The IVI Shared Components Version 1.2.1.0 (or later) must be installed before installing the appropriate drivers. You can find the IVI Shared Components on the IVI website at http://www.ivifoundation.org.

If you have questions on working with IVI drivers, you can download the IVI Getting Started Guide, also available on the IVI website.

Note: In order to use the included drivers to control the 1830 system and plug-in cards, you need to have a current version of VISA runtime already loaded on your computer system. This could include the VISA product from National Instruments, among others.

If you don't have the shared components installed prior to downloading the drivers, you might receive the following warning.



Minimum Requirements for IVI Shared Components

- Windows® 2000: Service Pack 3 (MSI installer)
- Windows 2000: no service packs (EXE installer)

  Note: The essential requirement for Windows 2000 is MSI 2.0, which is available in Windows 2000 SP3 and pre-installed by the EXE version of the IVI Shared Component Installer.
- Windows XP: no service packs
- Windows Vista: no service packs
- Microsoft® .NET Framework: 1.1

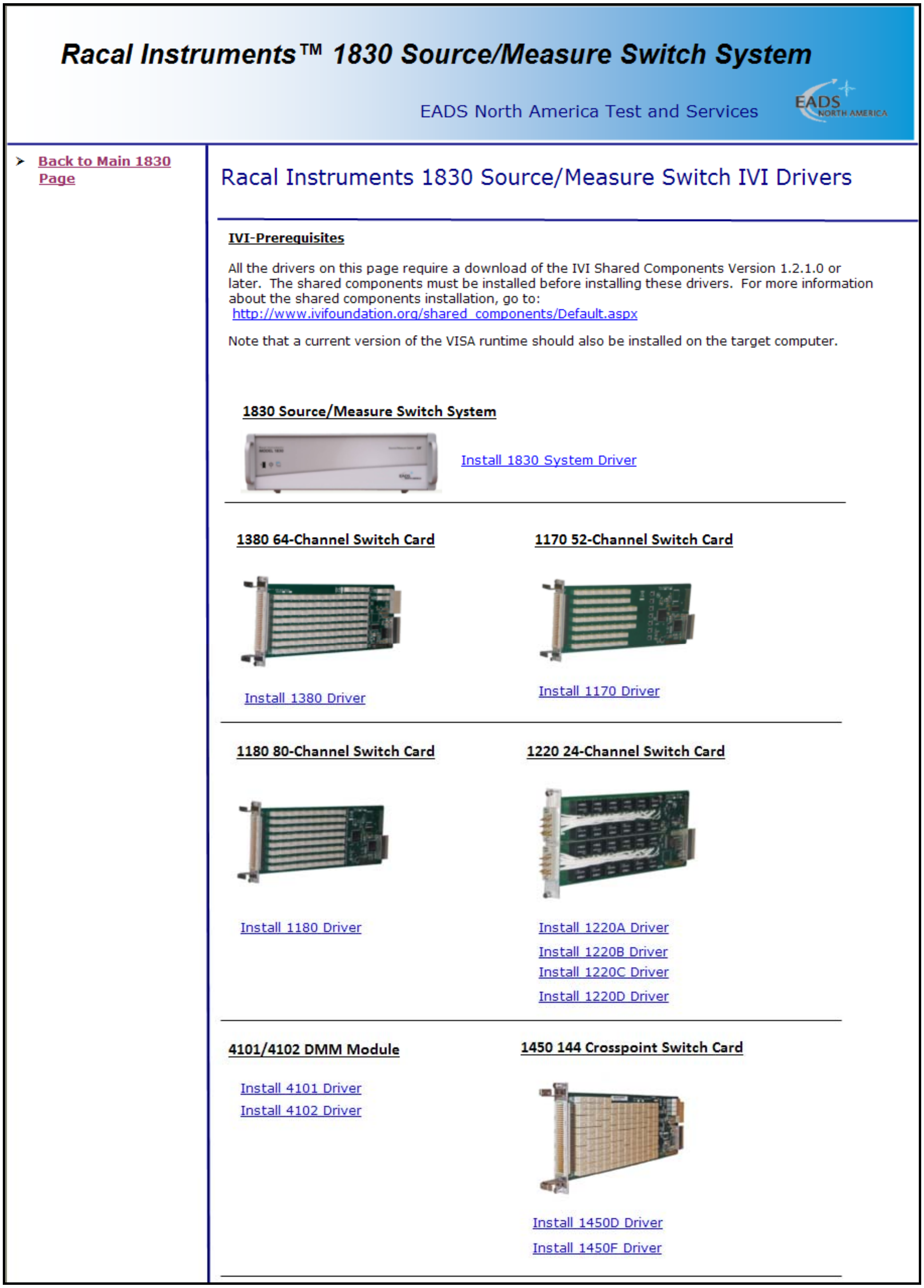


Figure 2-8: IVI Drivers Startup Page

VISA Resource Strings

To view the VISA resource strings, go to the 1830 home page (Figure 2-9). See **Chapter 4, Web-Page Interface**, for more information on accessing the web pages.

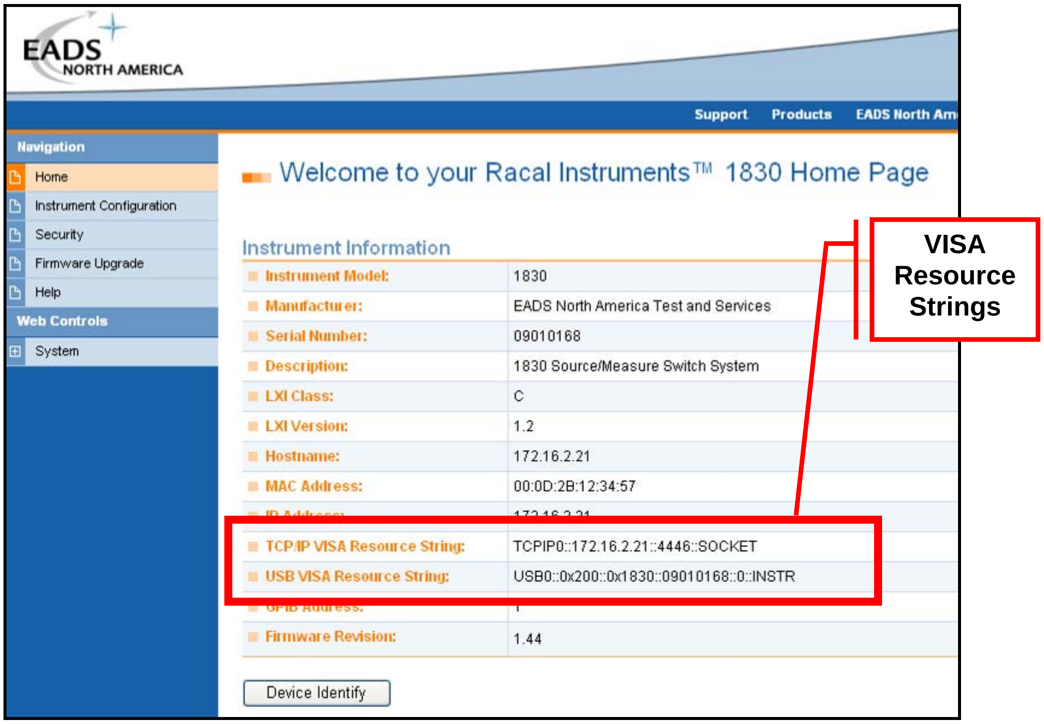


Figure 2-9: VISA Resource Strings

Setting the GPIB Address

The 1830 system does not have a physical hardware switch on the chassis to change the GPIB address. You can change the address by either using the web-page interface (using the LAN) or by sending a SCPI command (using the LAN, GPIB, or USB connection).

The factory default GPIB address setting is 1.

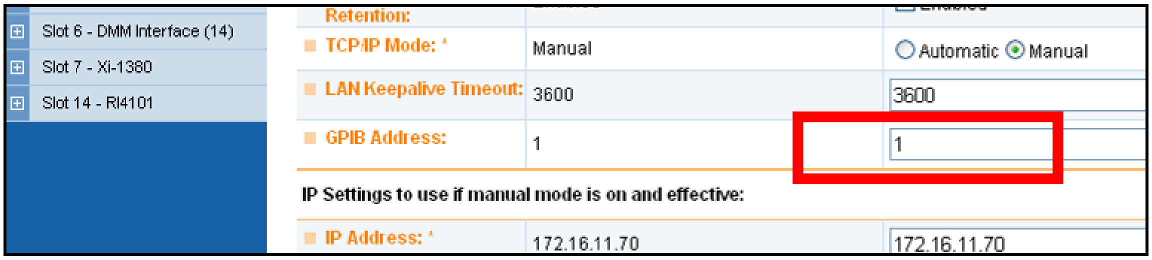
Using the Web-Page Interface to Set the Address

To change the GPIB address using the web-page interface, perform the following.

1. From the Home Page, select **Instrument Configuration**.

   If asked for a username and password, complete the information. Remember the default username is **admin** and the password box is left blank. If you have changed this, enter the current username/password.

2. Click the **Modify** button.
3. Change the address in the box next to GPIB Address.



4. Click **Save** to save the changes.

Using SCPI to Set the Address

To change the GPIB address of the chassis using SCPI commands (with either a LAN, GPIB, or USB connection), perform the following.

1. Send a "`SYSTem:COMMunicate:GPIB:ADDRess<n>`" command where <n> is the GPIB address and a number in the range of 0 to 30.
2. Confirm the change in GPIB address by sending the query: "`SYSTem:COMMunicate:GPIB:ADDRess?`"

Communicating with the 1830 Using GPIB and USB

While the LAN connection is considered the easiest method to communicate with your 1830 (and we feature the next three chapters showing how), you can communicated with the 1830 using the GPIB or the USB connections, as well as the VISA resources.

You first need to identify the appropriate VISA resource string. You can obtain this with the 1830 web-page interface (requiring a LAN connection) or through two commonly used test programs: National Instruments™ Measurement & Automation Explorer (NI MAX), Agilent™ Connection Expert (ACE).

Communicating Using GPIB

To use VISA to communicate using GPIB, you need a GPIB VISA resource string. A GPIB VISA resource string has the format of "`GPIB<i>::<n>::INSTR`", where `<i>` is the GPIB interface number on the host computer (typically this is 0 if there's only 1 GPIB controller) and <n> is the GPIB address of the device.

Acquiring the GPIB VISA Resource String with the Web Page

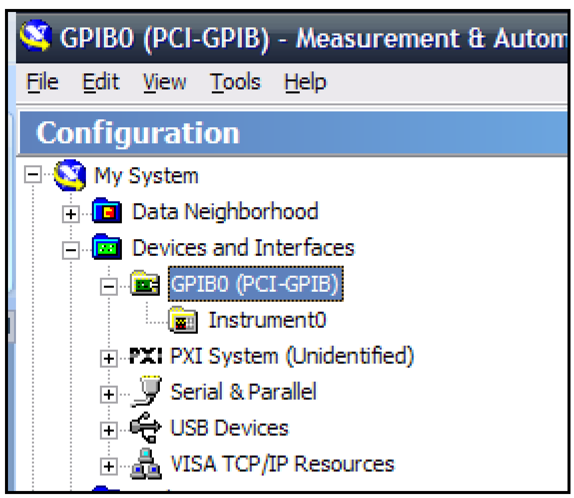
This assumes your 1830 system is attached through a LAN connection.

1. Open the 1830 Home page using a web browser.
2. Under the **Instrument Information** section of the page, view and record the GPIB Address.

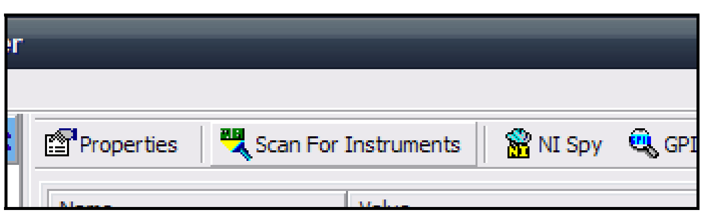
3. The VISA resource string can be formed with the discovered GPIB address. Using the example above, the resource string is "`GPIB0::1::INSTR`" (assuming there is only 1 GPIB controller).

Acquiring the GPIB VISA Resource String with NI MAX

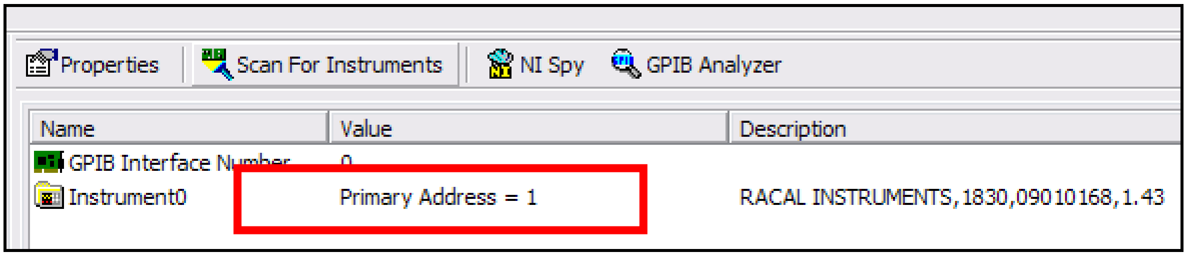
1. Open up NI MAX.
2. Under **Devices and Interfaces**, click the GPIB interface item.



3. Click **Scan For Instruments** on NI MAX.



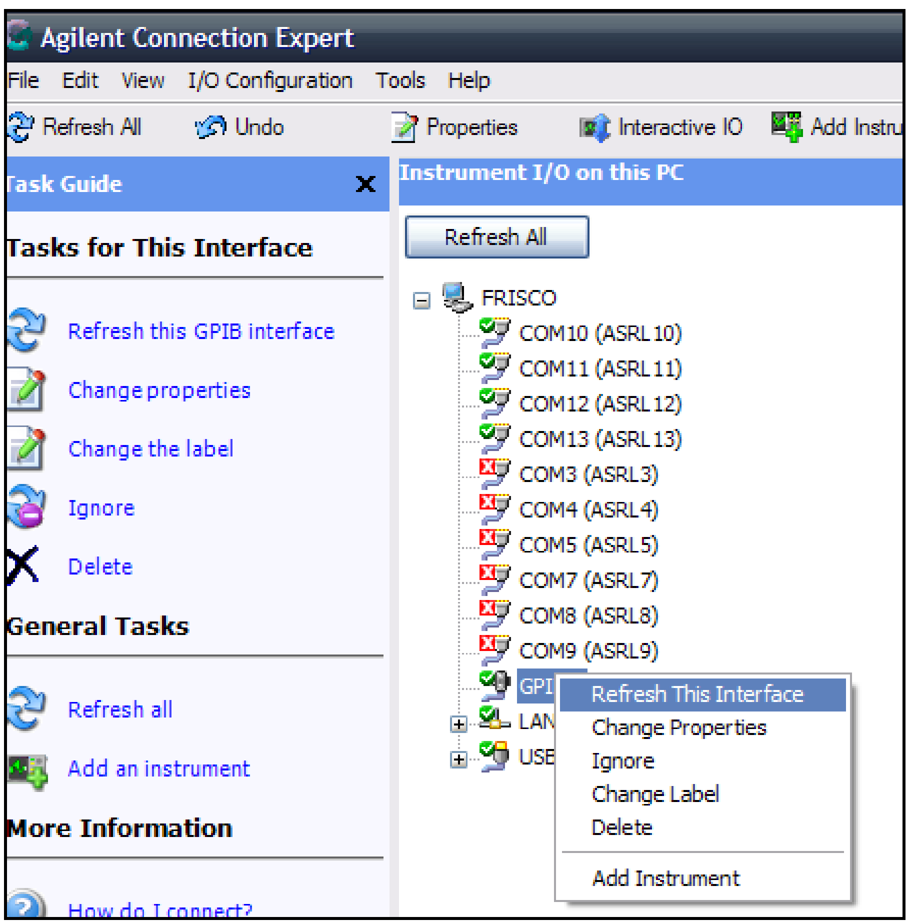
4. After the scan, all of the devices connected to the GPIB controller are displayed with the address of the device shown.



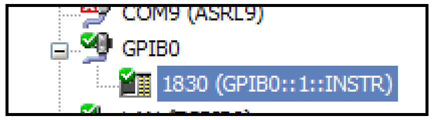
The GPIB VISA resource string for the above device is "`GPIB0::1::INSTR`". Note that if there are multiple GPIB interfaces, then the VISA resource string is "`GPIB<n>::1::INSTR`", where "n" is the number of the GPIB interface.

Acquiring the GPIB VISA Resource String with ACE

1. Open up ACE.
2. Under **Instrument I/O on this PC**, right click on **GPIB0** and select **Refresh This Interface**.



3. Any connected GPIB devices is displayed along with its VISA resource string under the GPIB interface node.



Additional device information is displayed as shown below.

Communicating Using USB

To use VISA to communicate with the 1830 through USB, you need a USB VISA resource string. A USB VISA resource string has the format of "USB0::<vendor id>::<model id>::<serial number>::0:INSTR".

For all 1830 instruments, the vendor id is "`512`" or "`0x200`" and the model id is "`6192`" or "`0x1830`". The remaining information you need to discover is the serial number of the instrument.

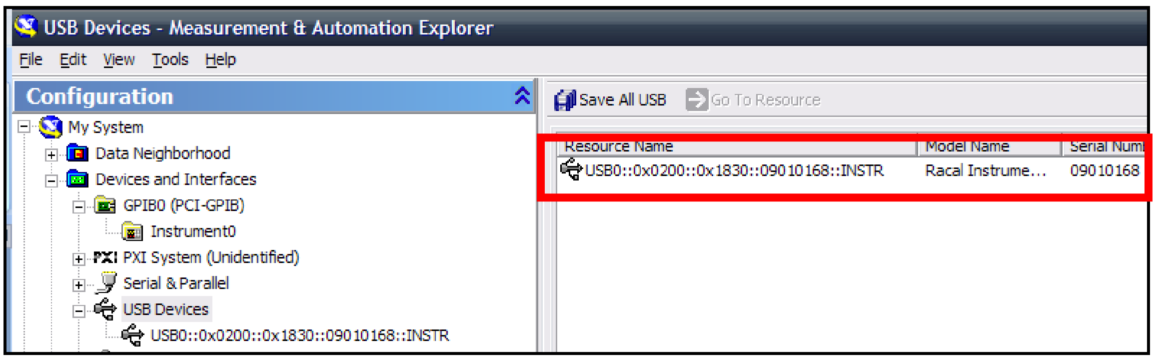
Acquiring the USB VISA Resource String with the Web Page

This assumes your 1830 is attached through a LAN connection.

1. Open the 1830 Home page using a web browser.
2. Under the **Instrument Information** section, view and record the USB VISA Resource String.

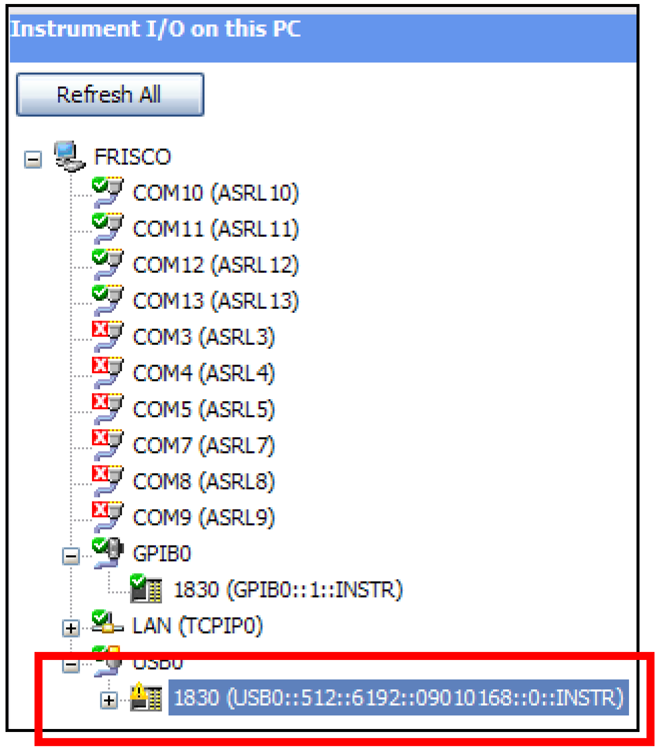
Acquiring the USB VISA Resource String with NI MAX

1. When the device is plugged in the first time, Windows asks if you want to install the appropriate drivers. Given that VISA is installed on the computer, allow Windows to automatically install the driver.
2. NI MAX recognizes Test and Measurement USB devices and they are listed under **USB Devices** along with the VISA resource string of the instrument.



Acquiring the USB VISA Resource String with ACE

1. When the device is plugged in the first time, Windows asks if you want to install the appropriate drivers. Given that VISA is installed on the computer, allow Windows to automatically install the driver.
2. All Test and Measurement USB devices are listed under **Instrument I/O on this PC** in ACE.



Additional device information is displayed as shown below.

Connecting Over LAN

Selecting the LAN Network Type

You can configure your 1830 to connect to a Site or Isolated (non-site) LAN.

- A **Site LAN network** is a local area network (LAN) in which computers and LAN-enabled instruments are connected to a site LAN (workgroup LAN, Intranet, or enterprise LAN) via optional routers, hubs, and/or switches.

  Typical Site LAN networks:

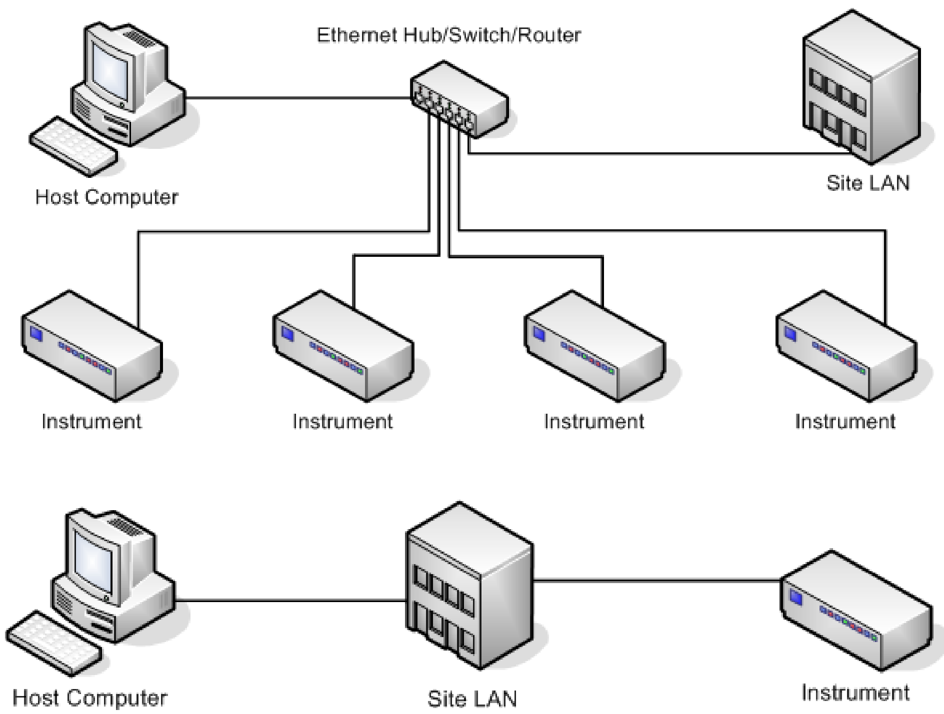


Figure 2-10: Site LAN Network

- An **Isolated LAN network** is defined as a local area network (LAN) in which computers and LAN-enabled instruments are not connected to a site LAN.

Typical Isolated LAN networks:

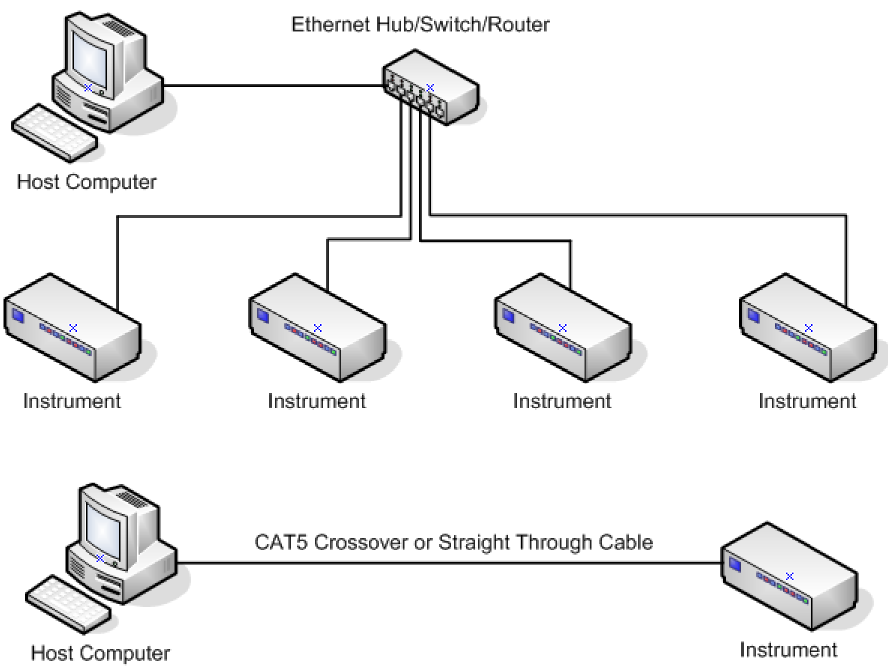


Figure 2-11: Isolated LAN Network

Select the LAN network type the 1830 will be connected to and follow the steps in the appropriate sections: **Configuring the 1830 to work in a Site LAN** and **Configuring the 1830 to work in an Isolated LAN**.

Configuring the 1830 to Work in a Site LAN

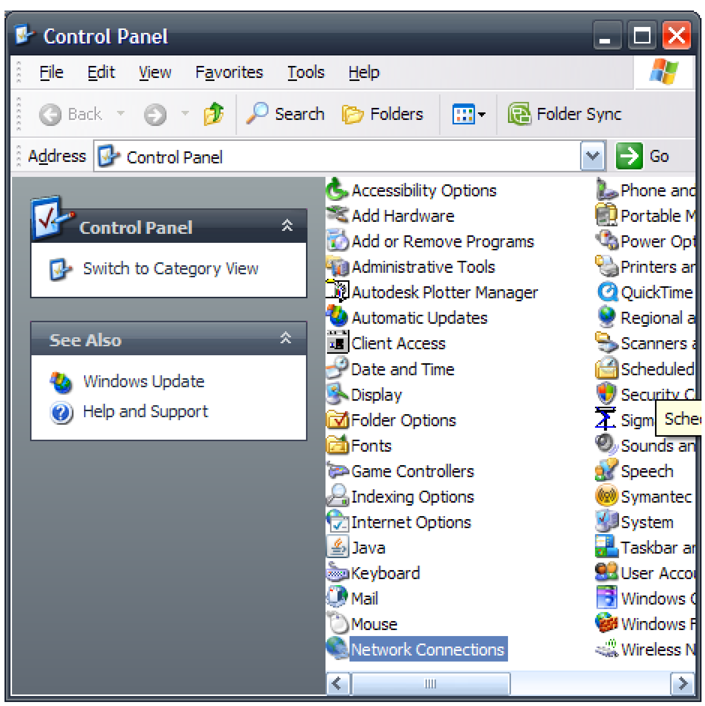
1. Using either a crossover or straight through CAT5 cable, connect the 1830 to the Site LAN or a switch/hub/router that is connected to the Site LAN. Or directly connect the 1830 to your computer.
2. Make sure power is applied to your computer and the 1830.
3. The 1830 by default is configured to use “automatic” mode to obtain an IP address to use. In this mode the 1830 will get its IP address from the DHCP server that is on the Site LAN, provided the DHCP server is configured properly.
4. Use a LXI discovery browser to scan the network for LXI instruments. If the 1830 was not discovered during LXI discovery, activate the LAN Configuration Initialization (LCI) by pressing the LAN reset button on the back panel of the 1830 for at least 5 seconds. Use the LXI discovery browser to rescan the network.
5. Once the 1830 has been discovered, enter the IP address of the 1830 in the address bar of your web browser to bring up the 1830’s welcome page.

Configuring the 1830 to Work in an Isolated LAN

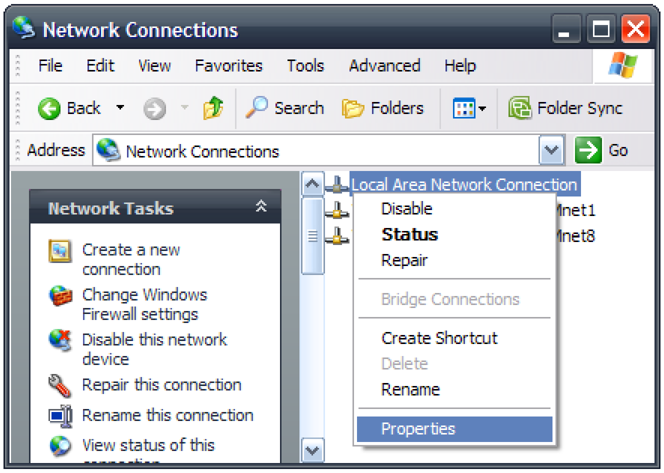
1. Using either a crossover or straight-through CAT5 cable, connect the 1830 to the switch/hub/router that is connected to your computer. Or directly connect the 1830 to your computer.
2. Make sure power is applied to your computer and the 1830.
3. If a router is used to connect your computer and the 1830 and DHCP is enabled on the router, go to section “Configuring the 1830 to work in a Site LAN” to complete the rest of the procedure.

The 1830 by default is configured to use “automatic” mode to obtain an IP address to use. Since no DHCP server is connected to your Isolated LAN network, the 1830 uses link-local addressing to obtain its IP address. If GPIB or USB communication to the 1830 is available, you may assign a static IP address to the 1830 manually, or query the 1830 for its LAN settings directly (go to section **Using GPIB or USB to Set 1830 LAN Settings** or **Using GPIB or USB to Query 1830 LAN Settings**). If GPIB and USB communication are not available, proceed with the following steps.

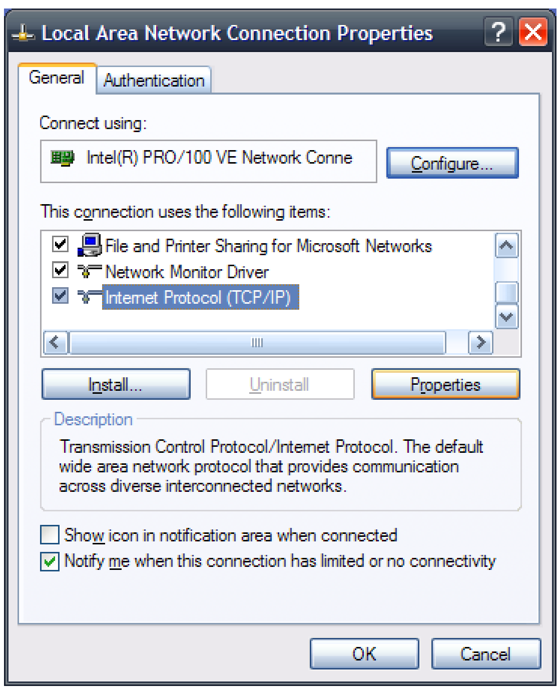
4. Use a LXI discovery tool to scan the network for all LXI instruments. If the 1830 was discovered, enter the IP address of the 1830 in the address bar of your web browser to bring up the 1830’s welcome page. If 1830 was not discovered, proceed with the following steps.
5. Open **Control Panel** and double click on **Network Connections**.



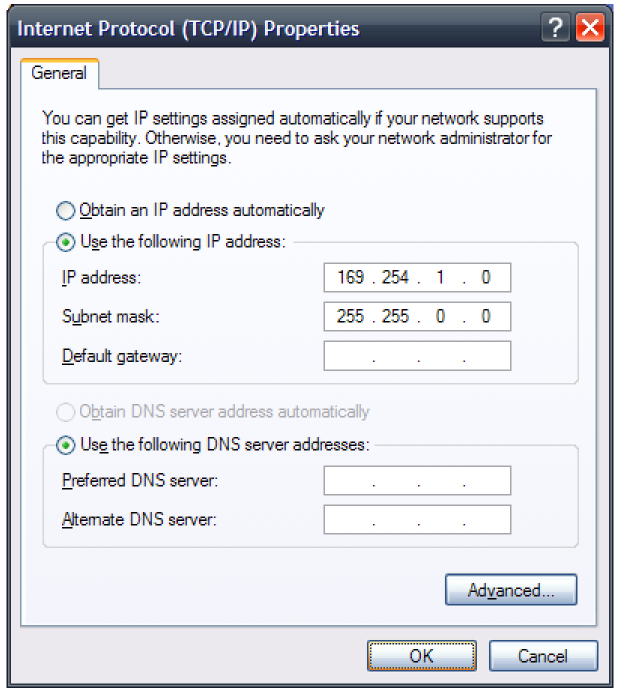
6. In the **Network Connections** window, right click on **Local Area Network Connection** and then click **Properties**.



7. In the **Properties** window, click **Internet Protocol (TCP/IP)** and then click **Properties**.



8. In the Internet Protocol (TCP/IP) Properties window, select **Use the following IP address** and input “169.254.1.0” for the IP address and “255.255.0.0” for the Subnet mask. Leave all other fields blank. Then click **OK**.



9. Click **OK** to close the **Local Area Network Connection Properties** window.

10. Use a LXI discovery tool to scan the network for LXI instruments, the 1830 should be listed as a discovered instrument. Enter the IP address of the 1830 in the address bar of your web browser to bring up the 1830’s welcome page.

Using GPIB or USB to Set 1830 LAN Settings

To Assign an IP address to the 1830 manually, send the following SCPI commands via USB or GPIB:

1. `SYSTem:COMMunicate:LAN:MODe MANUAL`
2. `SYSTem:COMMunicate:LAN:MAN:IPADdress <Instrument’s IP address>`
3. `SYSTem:COMMunicate:LAN:MAN:SMASk <Subnet mask>`
4. `SYSTem:COMMunicate:LAN:MAN:DNS NONE`
5. `SYSTem:COMMunicate:LAN:RESET`

For example, if your computer is using IP address “192.168.0.1” and the subnet mask is “255.255.0.0”, then the Instrument’s IP address will be in the range of 192.168.<0-255>.<0,2-255> and the subnet mask would be “255.255.0.0” as well.

Using GPIB or USB to Query 1830 LAN Settings

If you choose not to assign an IP address to the 1830 manually, you may query the 1830 for its LAN settings. Since the 1830's factory default setting for LAN mode is "automatic", the 1830 will use link-local auto configuration if no DHCP server is connected to the LAN (which is the case in an isolated LAN). To query the 1830 for its' LAN settings, issue the following SCPI queries:

1. `SYSTem:COMMunicate:LAN:IPAD?`
2. `SYSTem:COMMunicate:LAN:SMASk?`

The 1830's IP address is going to be in the range of 169.254.<0-255>.<0-255> with the subnet mask being "255.255.0.0.". Once you've obtained the instrument's IP address and subnet mask, may try to load the 1830's welcome page by entering its IP address in your web browser's address bar. If you're unable to load the welcome page, perform Steps 5 to 10 of the previous section **Configuring the 1830 to Work in an Isolated LAN**.

Chapter 3
Discovering and Configuring Your 1830 on a LAN

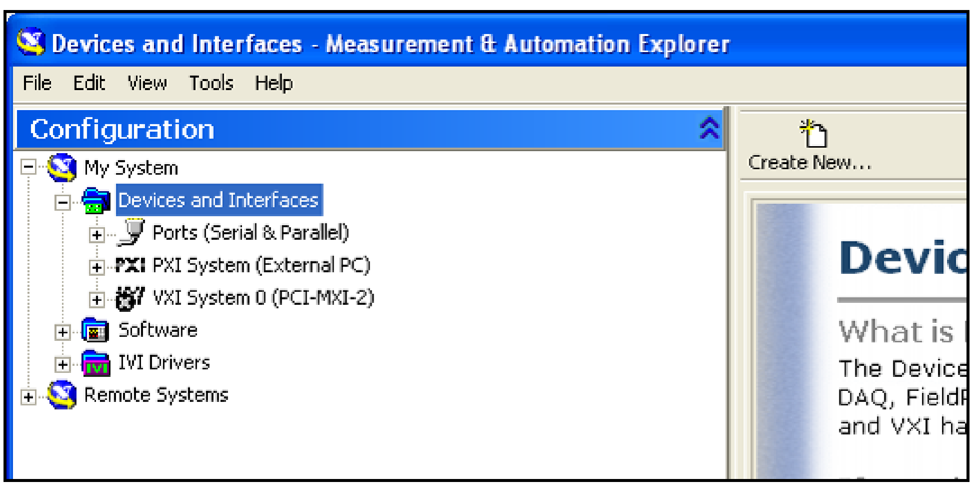
This chapter shows you how to discover and configure your 1830 on a LAN using either the National Instruments or Agilent discovery software tools.

These tools might also be able to help you determine the USB and GPIB addresses as well. Refer to their software documentation for further information.

Using National Instruments Measurement & Automation Explorer

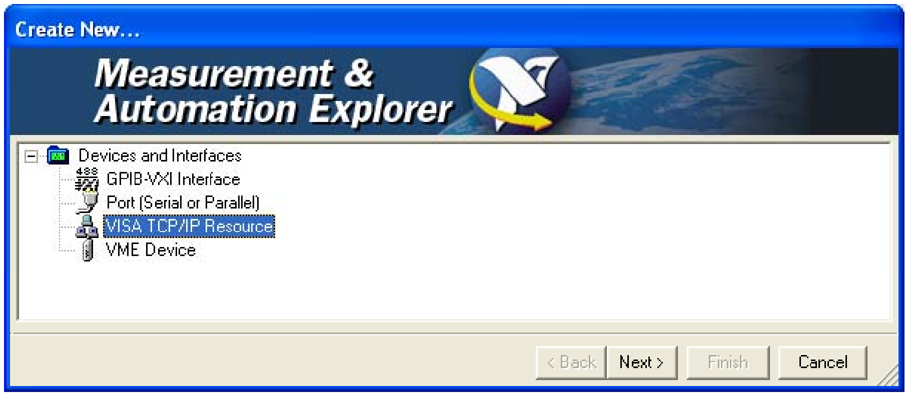
Following are instructions for finding and configuring the 1830 using National Instruments' Measurement & Automation Explorer:

1. Ensure the computer is connected to the same network as the 1830.
2. Launch the Measurement and Automation Explorer program from the Windows **Start…** launch bar.
3. Select the **Devices and Interfaces** item in the displayed tree.

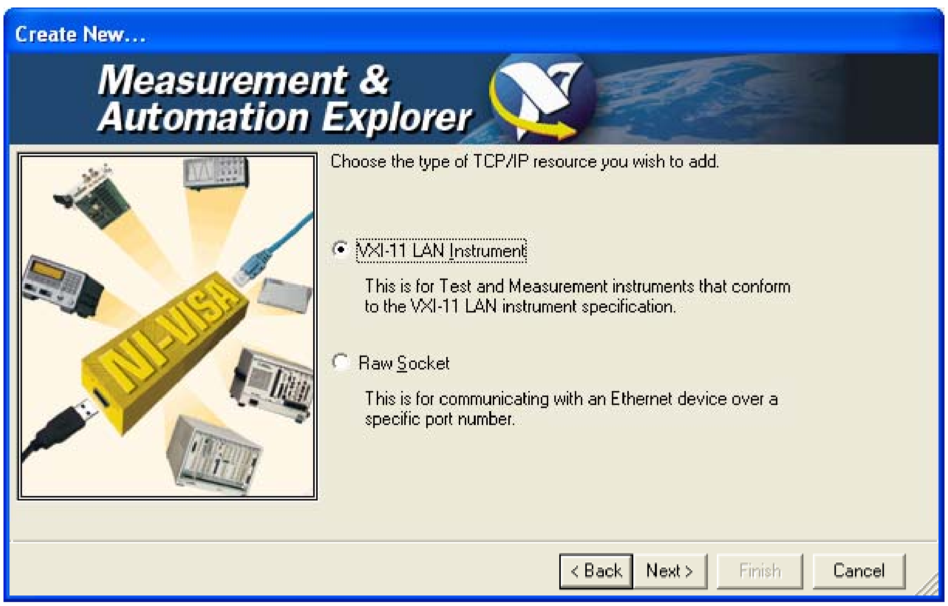


4. Right-click **Devices and Interfaces** and select **Create new…**.

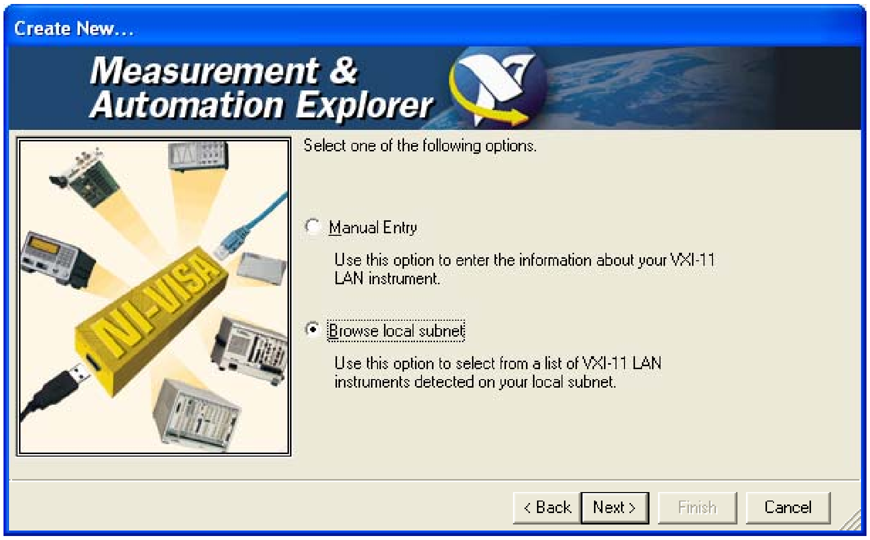
5. A dialog window is displayed. Select the **VISA TCP/IP Resource** and click **Next>**.



6. Select the **VXI-11 LAN instrument** option and click **Next>**.

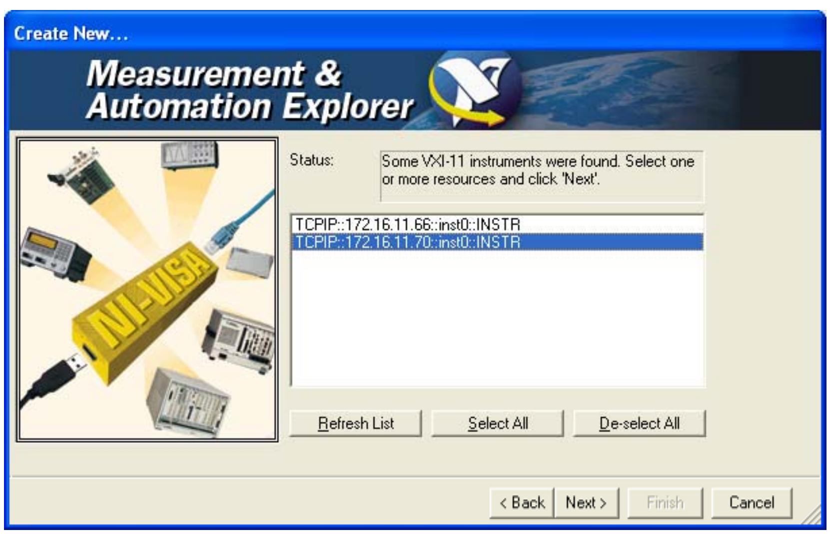


7. Select **Browse local subnet** and click **Next>**.

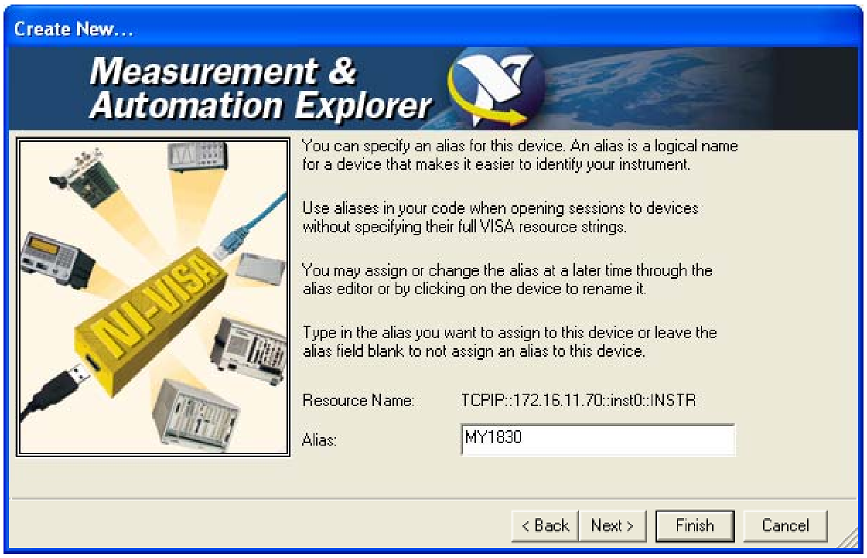


The Measurement and Automation Explorer program searches the network for all instruments that respond with the VXI-11 protocol. This includes the 1830. A menu of possible instruments is displayed in the window after a few moments.

8. Select the item that corresponds to the 1830 and click **Next>**.

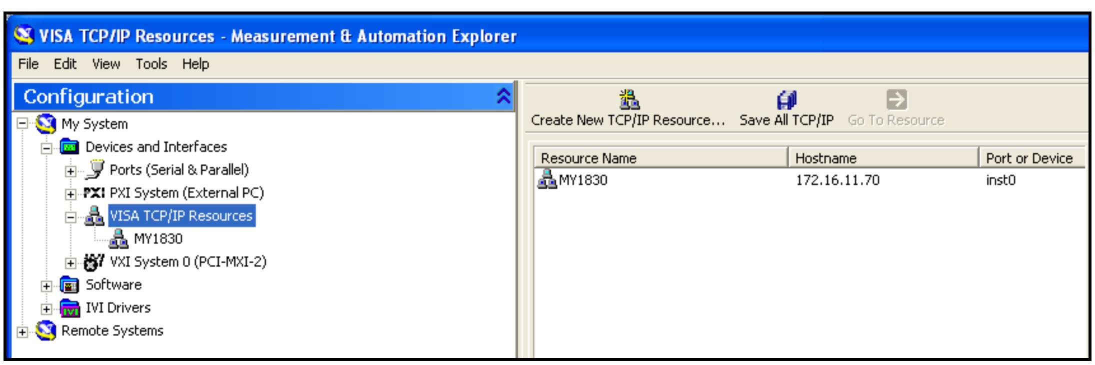


9. Copy the TCPIP address for a later step. Ignore the characters after the address. In the example below, the address would be **172.16.11.70**.
10. You may supply an Alias to the instrument if you so choose. This Alias is displayed instead of the VISA descriptor in Measurement and Automation Explorer. Click the **Finish** button to complete the addition of the resource.



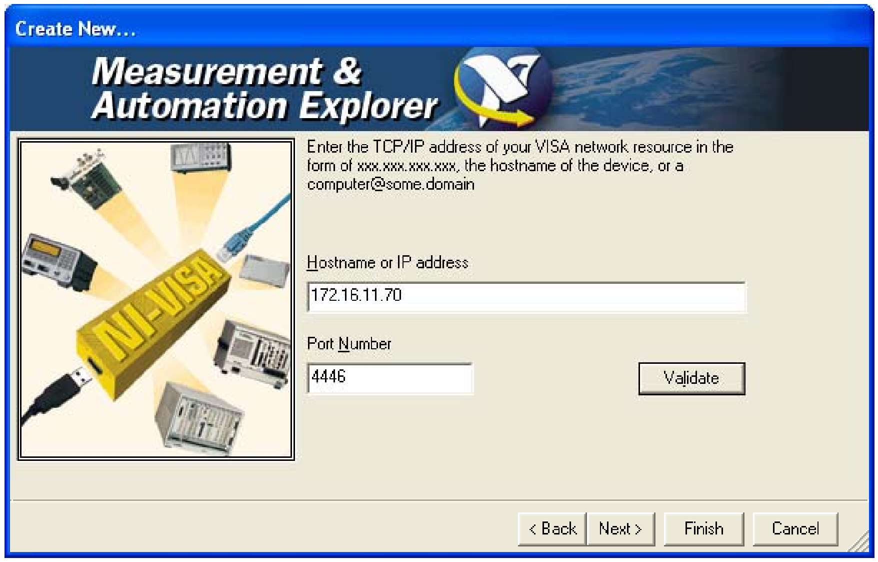
This LXI-11 resource responds to the "*IDN?" command only. In order to add a resource that responds to all SCPI commands, you need to add a "raw socket" resource as shown in the following steps.

11. Right-click on **ISA TCP/IP Resources** in the tree.

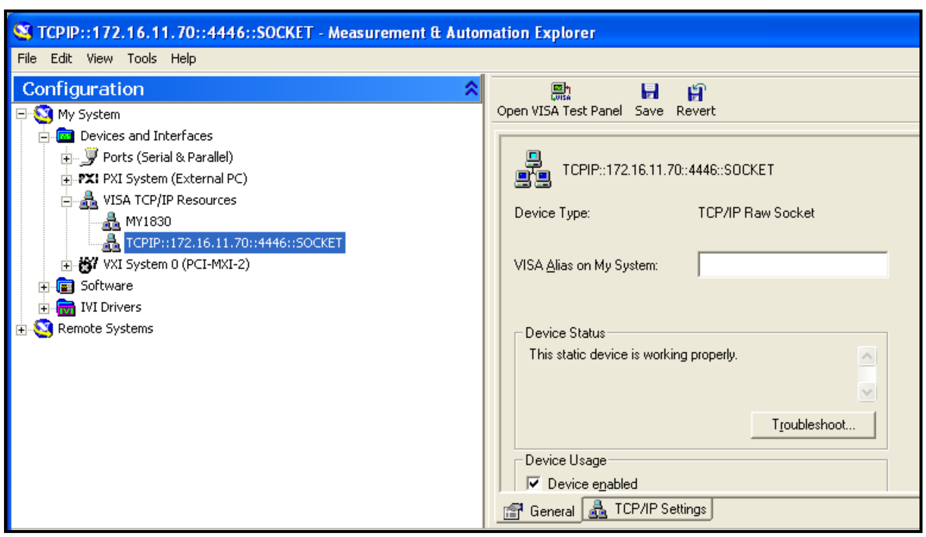


12. Select the **Create New TCP/IP Resource** menu item.
13. Select the **Create Raw Socket** item and click **Next>**.

14. Enter the IP address that was found previously.

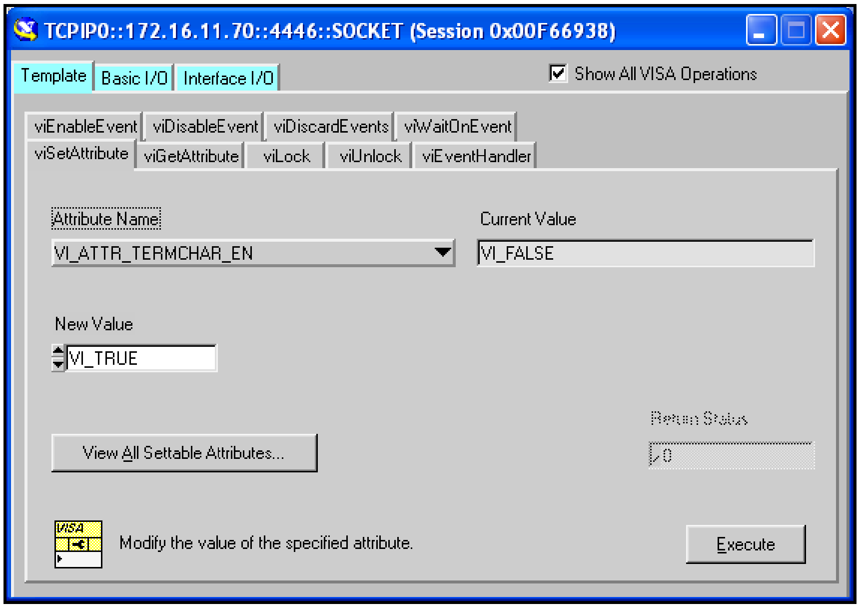


15. Enter port number **4446** which is used as the SCPI command port by the 1830.
16. Click **Finish**.
17. A new entry should be displayed in the Measurement and Automation Explorer window. Right-click the entry and select **Open VISA Test Panel**.



You can now read and write VISA commands to the instrument. For best results, you should enable the termination character attribute to select the “linefeed” (decimal 10, hexadecimal A) as the “end of message” character.

18. Select the **Template** tab of the communication window.



19. Select the **VI_ATTR_TERMCHAR_EN** item from the **Attribute Name** list.
20. Set the **New Value** to **VI_TRUE**.
21. Click **Execute**.

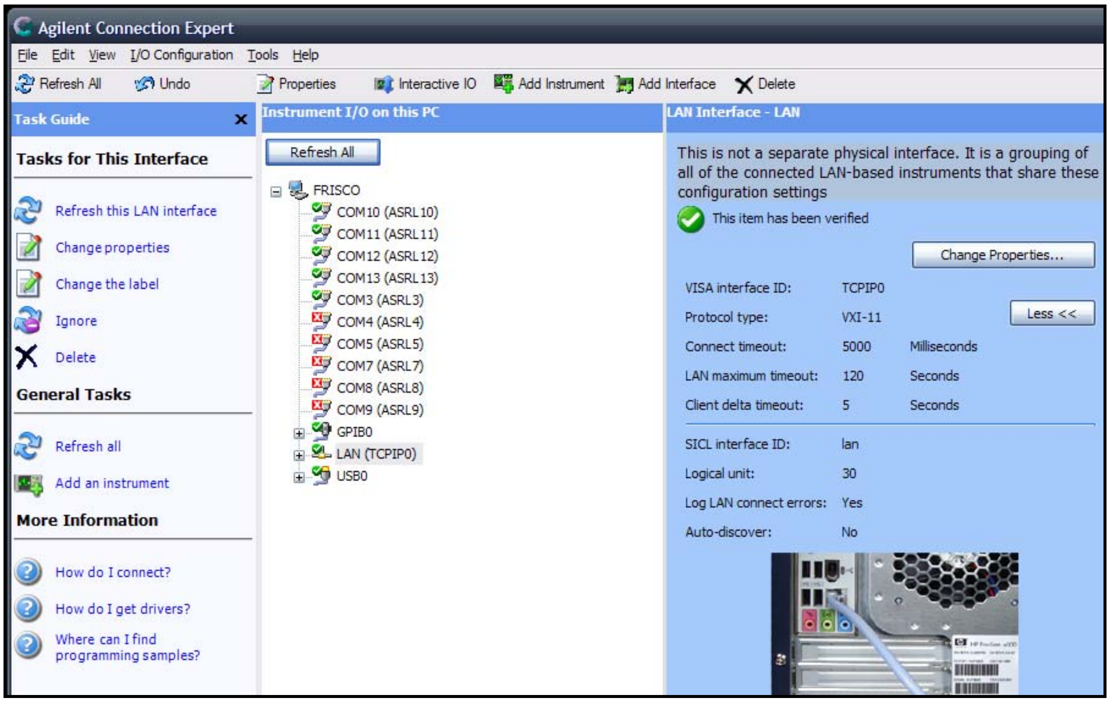
    This enables the default termination character (linefeed) to be used as an end-of-message mark.

Note that if you do not enable the linefeed as the termination character, you are still able to read and write to the instrument using the test panel. However, you will most likely observe that a timeout error (0xBFFF0015) occurs on each read and write.

Using Agilent's Connection Expert (CE)

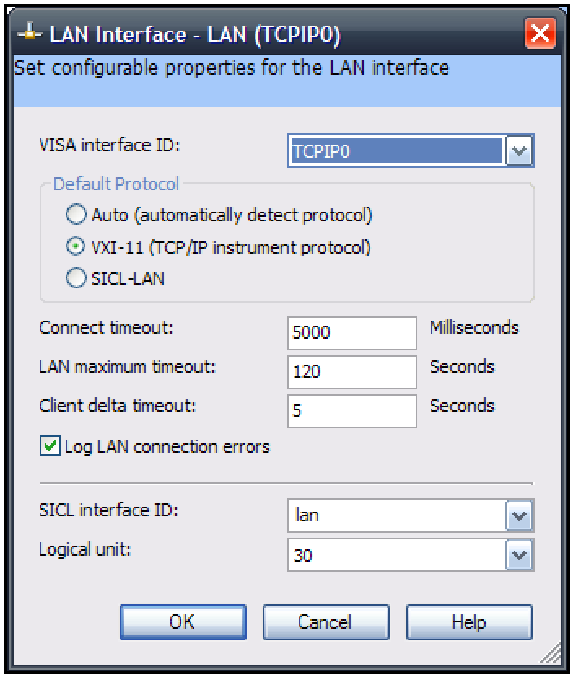
Following are instructions for finding and configuring the 1830 using Agilent's Connection Expert (CE):

1. Ensure the computer is connected to the same network as the 1830.
2. Launch the ACE program from the Windows **Start…** launch bar.
3. Select the **LAN** item in the displayed tree.

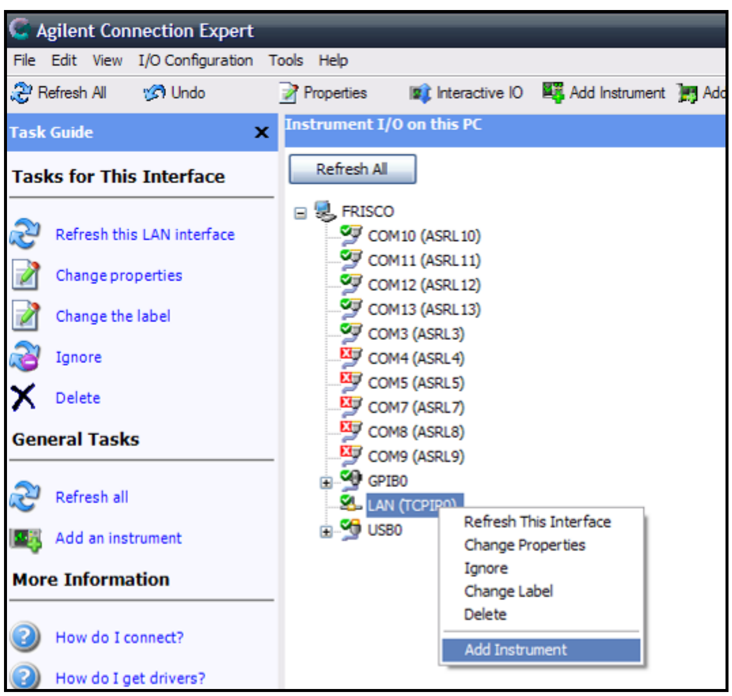


4. On the panel to the right, click **Change Properties…**.

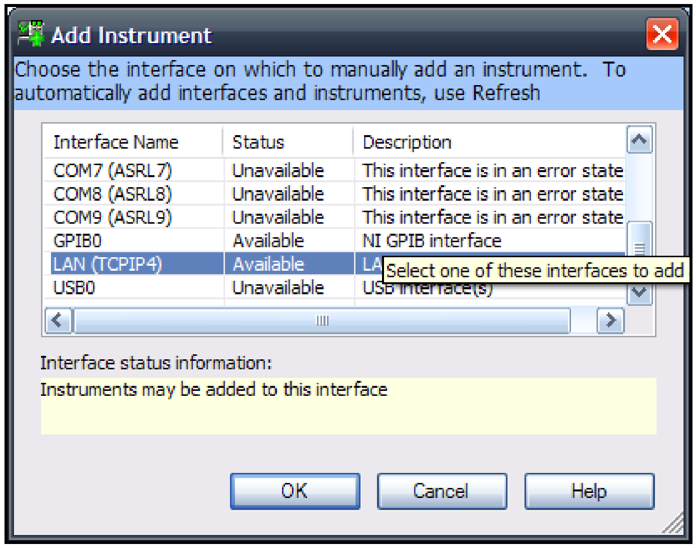
5. Select **VXI-11** under **Default Protocol** and click **OK.**



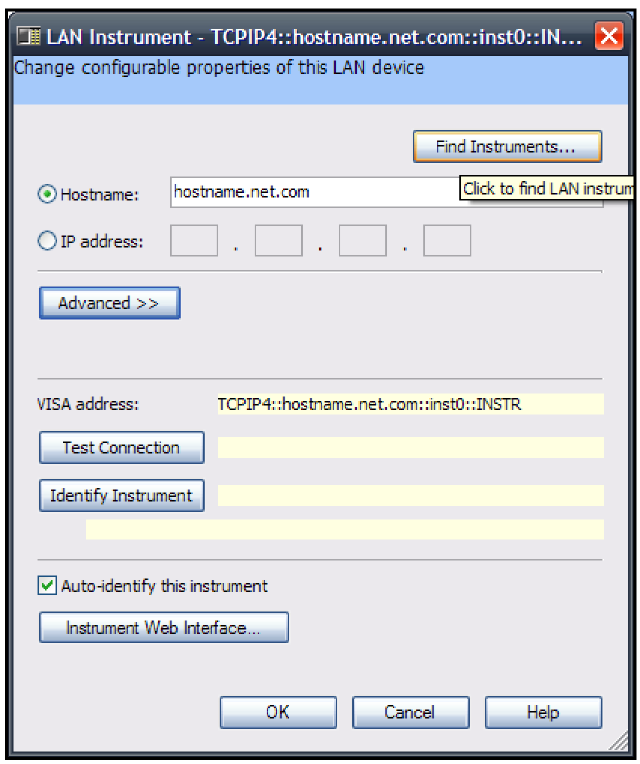
6. Right-click the **LAN** item in the displayed tree and select **Add Instrument…**



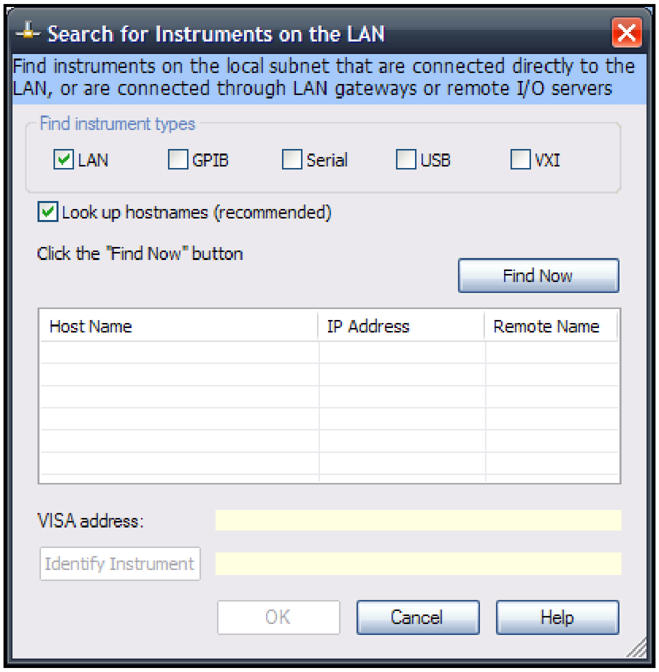
7. A dialog window displays. Select the **LAN (TCPIP*x*)** (where ***x*** denotes some integer value) item and click **OK**.



8. Click the **Find Instruments…** option and then click **OK**.



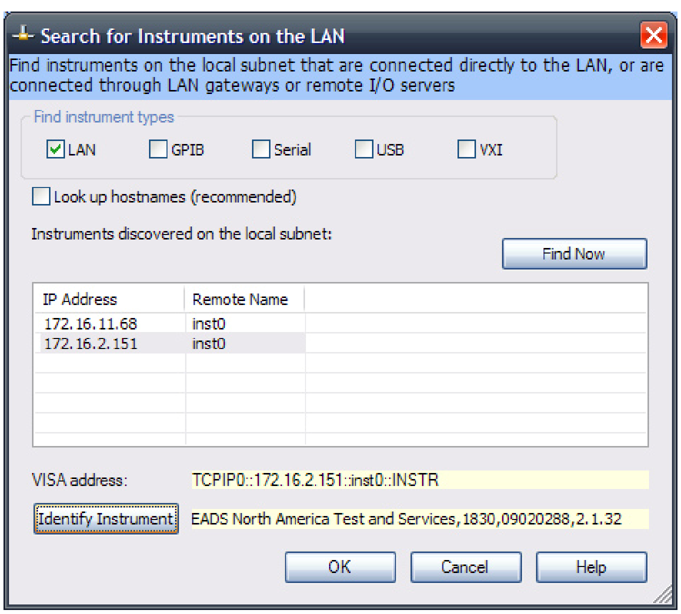
9. Select **LAN** and click the **Find Now** button.



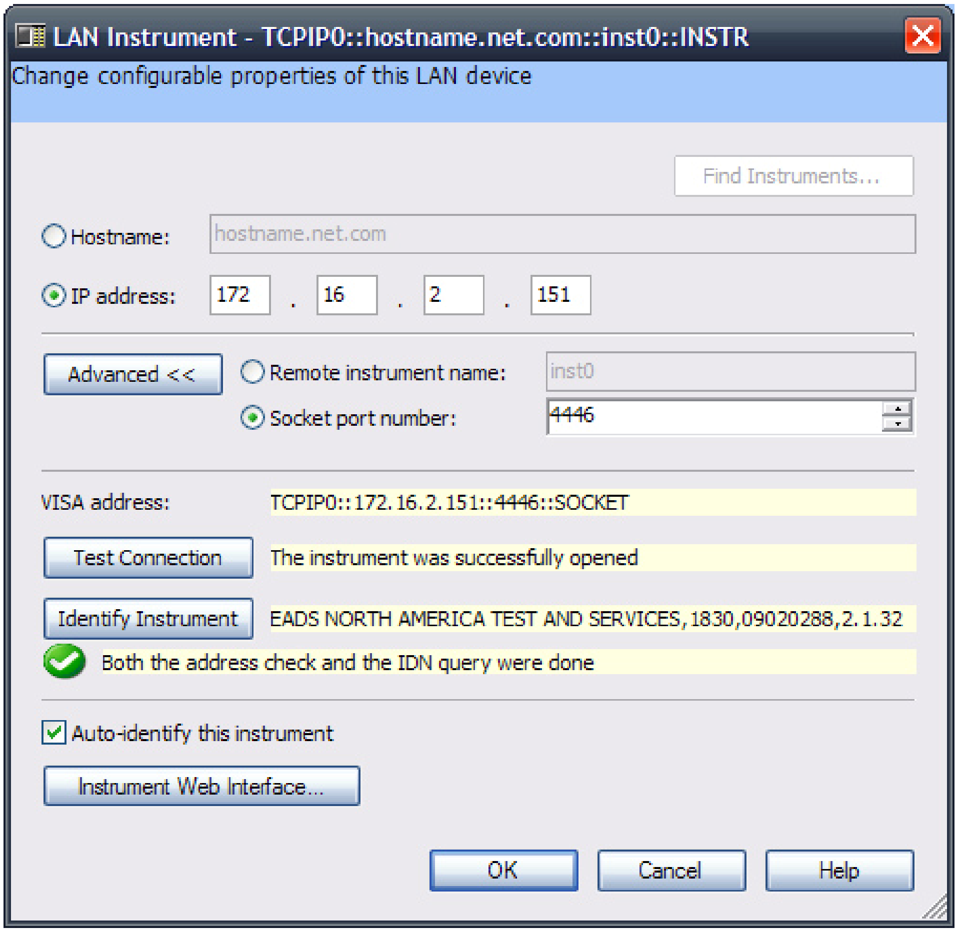
The program searches the network for all instruments that respond with the VXI-11 protocol. This includes the 1830. A menu of possible instruments will be displayed in the window after a few moments.

10. Select a discovered instrument and click **Identify Instrument** to get the identification of the instrument.

- If it matches the desired instrument, click **OK**.
- If not, select another discovered instrument until its identification matches the one of the desired instrument.

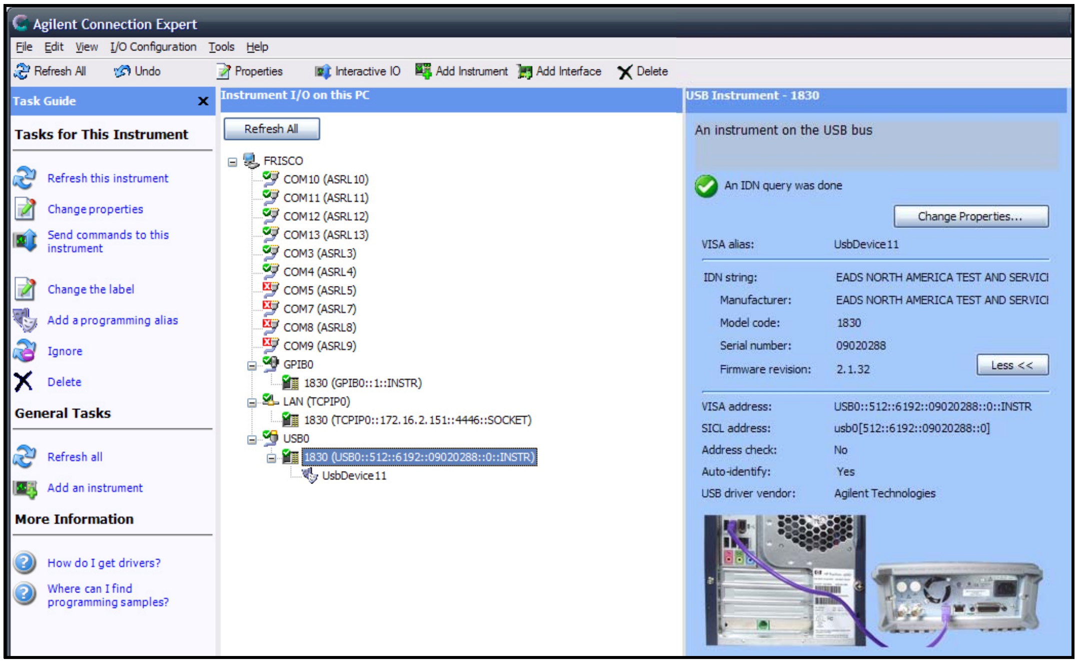


11. Click **Advanced,** select **Socket port number**, and then enter **4446** in the input box.

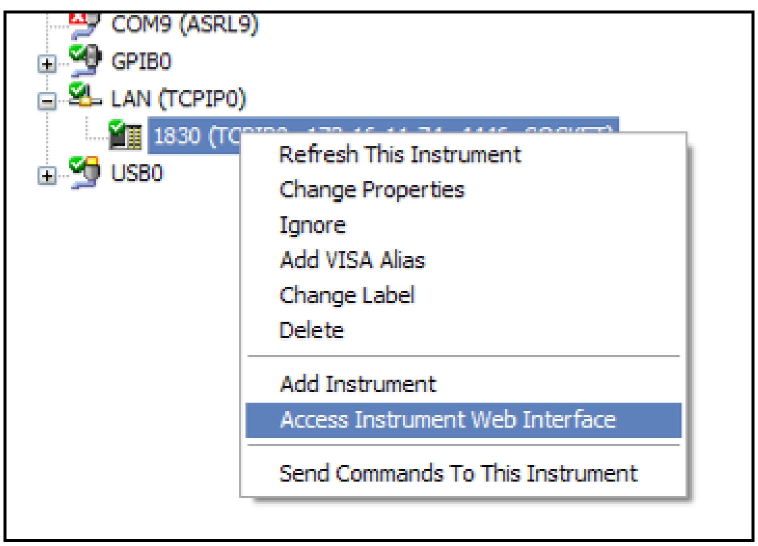


12. Click **Test Connection** and verify the connection can be opened.
13. Click **Identify Instrument** and verify an identification string of the instrument is returned.
14. Click **OK** to close the dialog box.

15. The new device now appears as an item under the **LAN** item in the tree.



16. Right-click the newly found 1830 LAN instrument and select **Access Instrument Web Interface** to launch the instrument's web page.
    Or you can select **Send Commands To This Instrument** to launch the Agilent Interactive IO control to send SCPI commands directly to the instrument.



Chapter 4
Web-Page Interface

This chapter shows you how to use the web-page interface to configure your 1830 and installed cards through a LAN to meet your testing needs.

Initial Home Page

To access the web-page interface on your 1830:

1. Attach your active LAN cable to the 1830.
2. Turn the 1830 power switch on.
3. Type in the IP address (discovered in **Chapter 3**) into your computer's Internet browser.

The 1830 instrument's built-in home page appears as shown below.

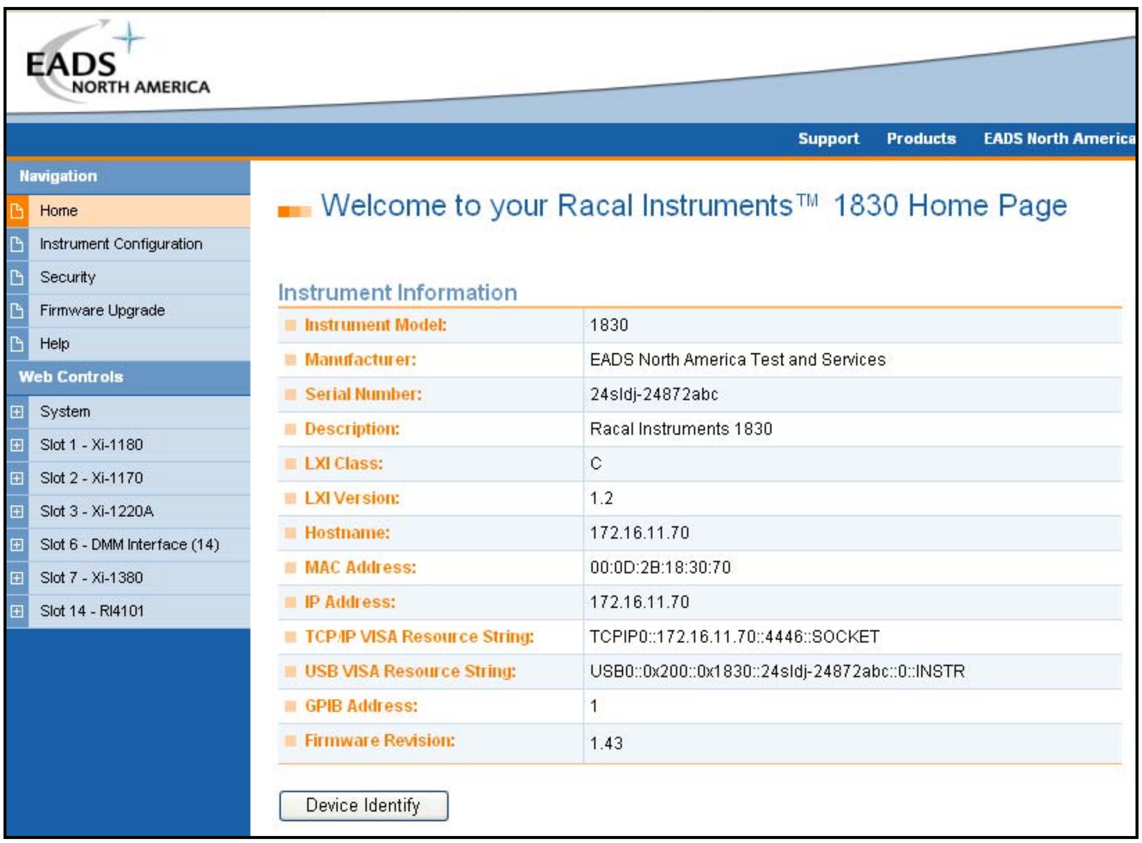


Figure 4-1: Web-Page Interface Home Page

Setting the Password

When you first access any of the web pages that allow you to modify settings, you are asked for a username and password. The default username is **admin**. There is no initial password and you can access the system without needing to enter any text. Just leave the line blank and click **Enter**.

To change the password at any time, click **Security** on the navigation bar on the left side of the web page. The following screen appears.

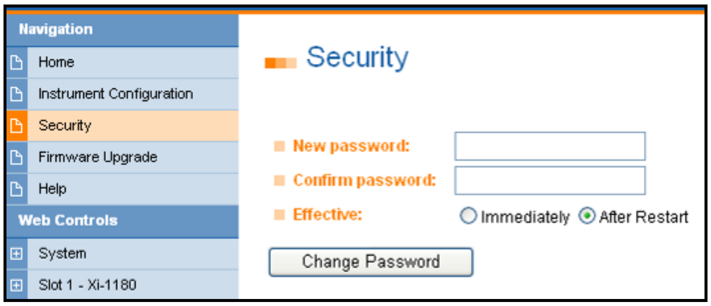


Figure 4-2: Security, Change Password

You can type in any password. The password can be virtually any length with any combination of letters and/or numbers. The password is case sensitive.

You can make the new password effective immediately or after a restart by clicking the appropriate option button. Click **Change Password** when done.

If you press the LAN Reset button on the rear panel of the 1830 for over 5 seconds, resetting your LAN setup, your password is reset to the factory default setting of no password. (See **LAN Reset Button** in **Chapter 1, Overview and Features**, for additional information on the operation of the button.)

Web Page Descriptions

Home Page

The Home Page (seen in Figure 4-1) provides a navigation bar on the left side of the screen which allows you to select specific pages for

- Instrument Configuration
- Security authorization (and password configuration)
- Firmware upgrade
- Help
- Web controls for the system and the installed cards
- USB Flash drive access (if attached)

The main panel on the web page provides information on

- Instrument model
- Manufacturer
- Serial number
- Instrument description
- LXI class
- LXI version
- Hostname
- MAC address
- IP address
- TCP/IP VISA resource string
- USB VISA resource string
- GPIB address
- Firmware revision

Device Identify Button

Figure 4-3: Device Identify Button

When you click this button on the Home page (Figure 4-3), the LAN status light on the 1830 front panel begins to blink. This feature allows you to identify which 1830 you are communicating with should you have a bank of them.

When you click the button a second time, the status light stops blinking and returns to normal mode.

External Storage Devices

Figure 4-4: USB Drive Indicator

When a USB flash drive is installed to the front USB connector, an **External Storage Devices** list appears on the home page.

Click **USB Drive** to see a directory of what is on the drive.

The USB connector on the rear of the chassis is only a communication interface. For memory purposes, use the USB connector on the front panel.

Instrument Configuration Page

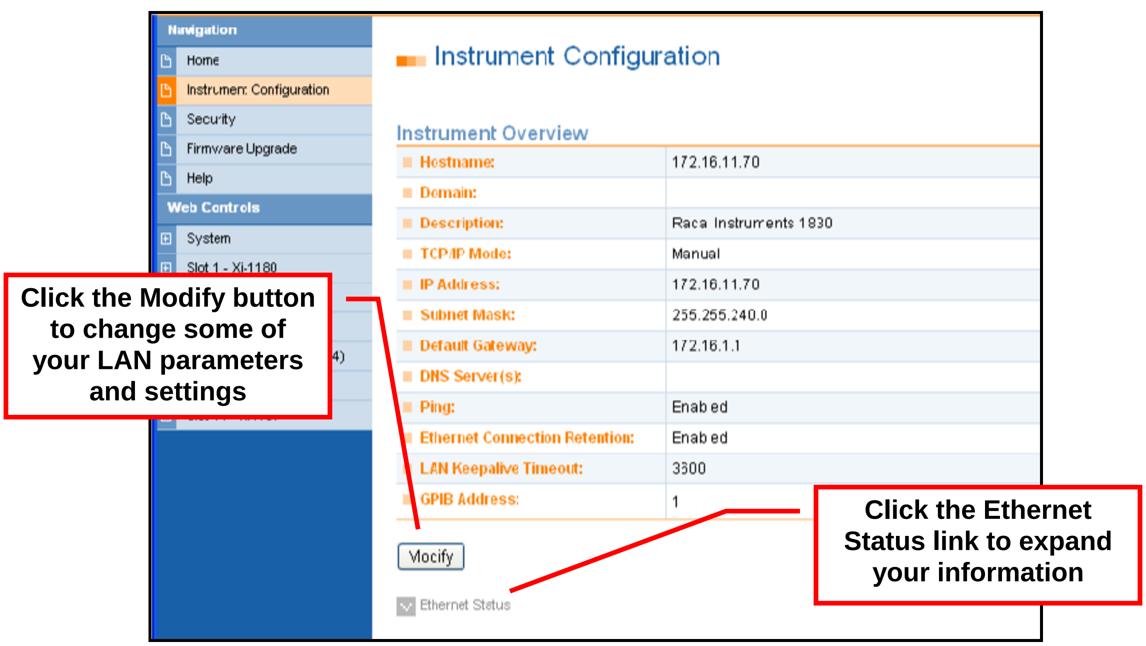


Figure 4-5: Instrument Configuration

The Instrument Configuration page (Figure 4-5) provides information on

- Hostname
- Domain (if any)
- Description
- TCP/IP mode
- IP address
- Subnet mask
- Default gateway (if any)
- DNS servers(s) (if any)
- Ping
- Ethernet connection retention
- LAN Keep Alive timeout
- GPIB address

Access additional Ethernet Status information by clicking the link. This portion of the screen provides additional packet information. (See Figure 4-6.) You can click the **Refresh** button to get updated information.

You can change some of these configurations by clicking the **Modify** button. See the next section, **Modifying the Instrument Configuration,** for more information.

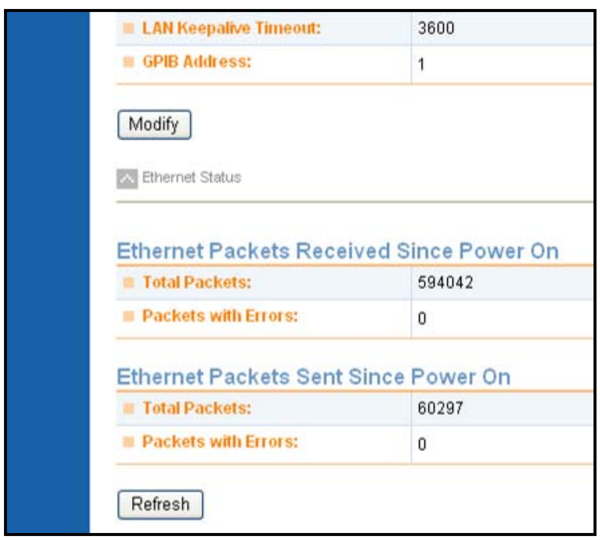


Figure 4-6: Ethernet Status

Modifying the Instrument Configuration

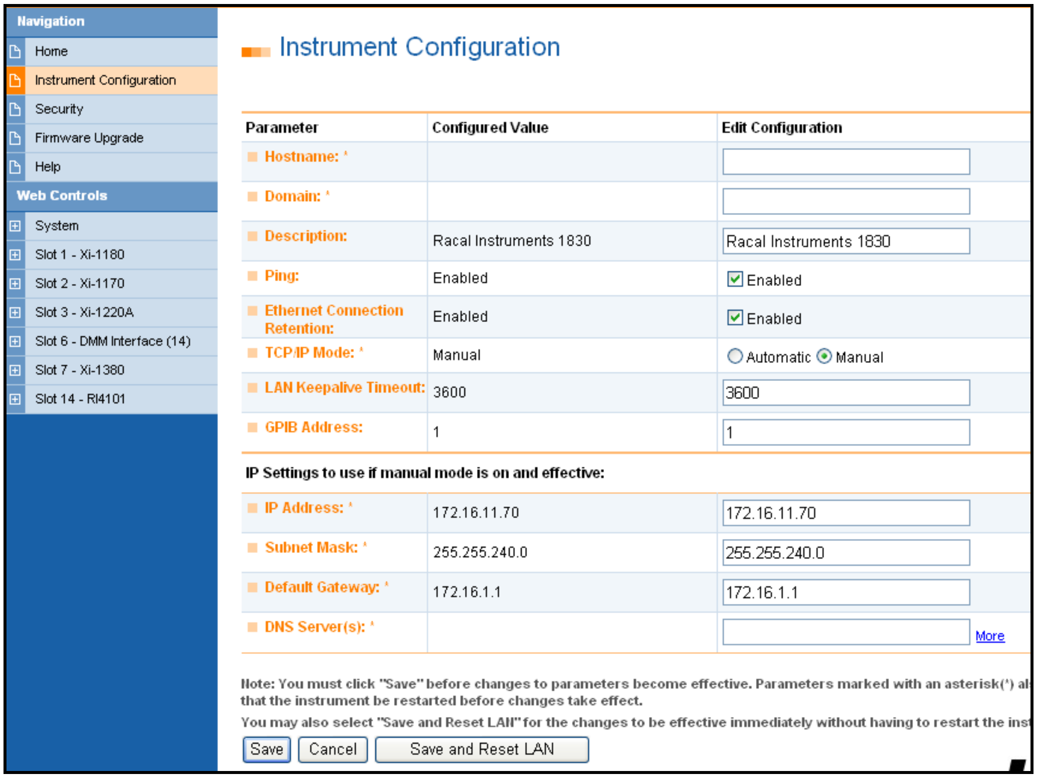


Figure 4-7: Instrument Configuration, Modify

After clicking the **Modify** button, you can modify the following parameters:

- Hostname *
- Domain *
- Description
- Ping (Enabled or not)
- Ethernet connection retention (Enabled or not)
- TCP/IP mode (Automatic or Manual) *
- LAN Keep Alive timeout
- GPIB address
- IP address *
- Subnet mask *
- Default gateway *
- DNS server(s) *

You must click the **Save** button before changes to the parameters become effective. Parameters marked with an asterisk (*) also require that the instrument be restarted before changes take effect.

You may also select the **Save and Reset LAN** button for the changes to be effective immediately without having to restart the instrument.

Web Controls

The Web Control section allows you to view and modify the status of your 1830 System as well as to view and modify the various installed cards on your system.

In order to use the various web controls, you need to have a recent version of the Java engine installed on your computer.

System Controls

Status/General Tab

On this tab (Figure 4-8), you can view mainframe and module (card) serial number and revision information.

If you have a DMM installed, the listing shows the location of the internal slot it the card was installed to, but it also shows the slot of the interface adapter as **Empty**.

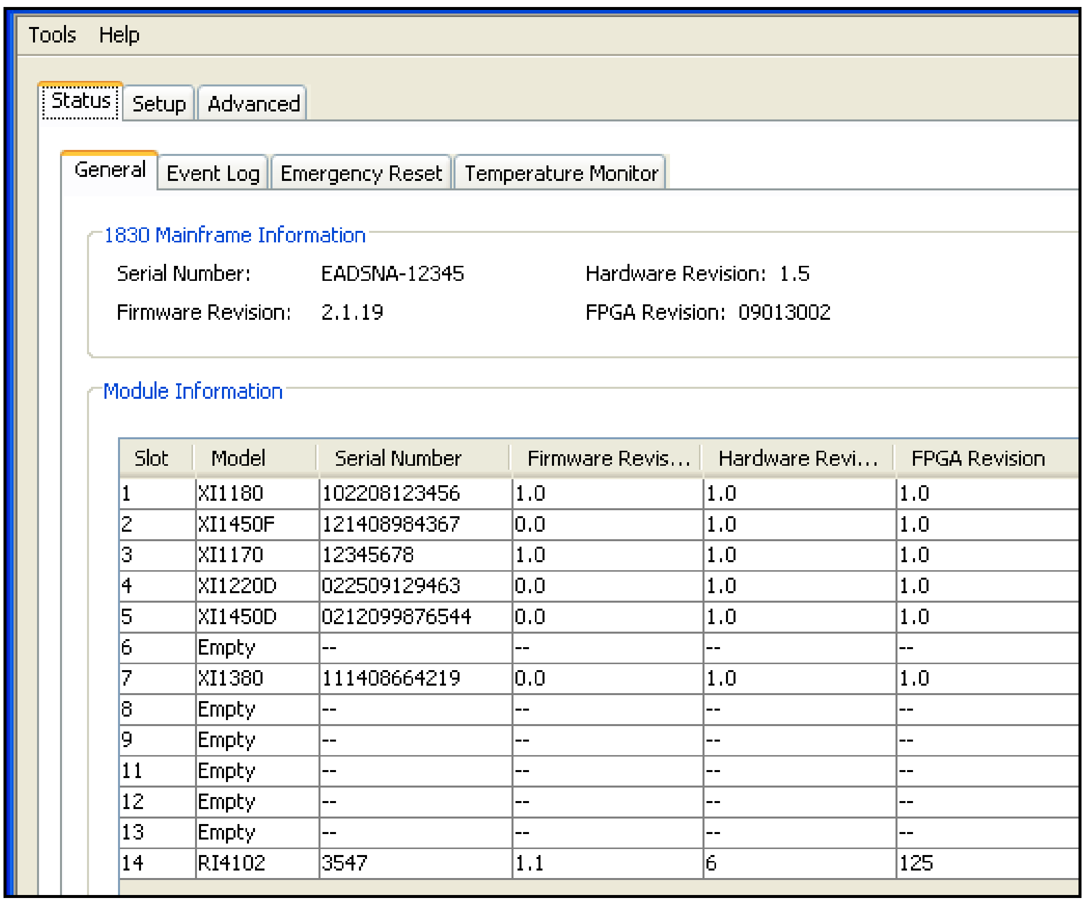


Figure 4-8: System Controls, Status/General Tab

Status/Event Log Tab

On this tab (Figure 4-9), you can view the recent events including power-on, system faults, and errors. You can select and clear specific event numbers.

At the bottom of the screen, the **Read All** button shows you all the events in the log. The **Clear All** button clears all of the events from the log. The **Clear** button clears the events from this page.

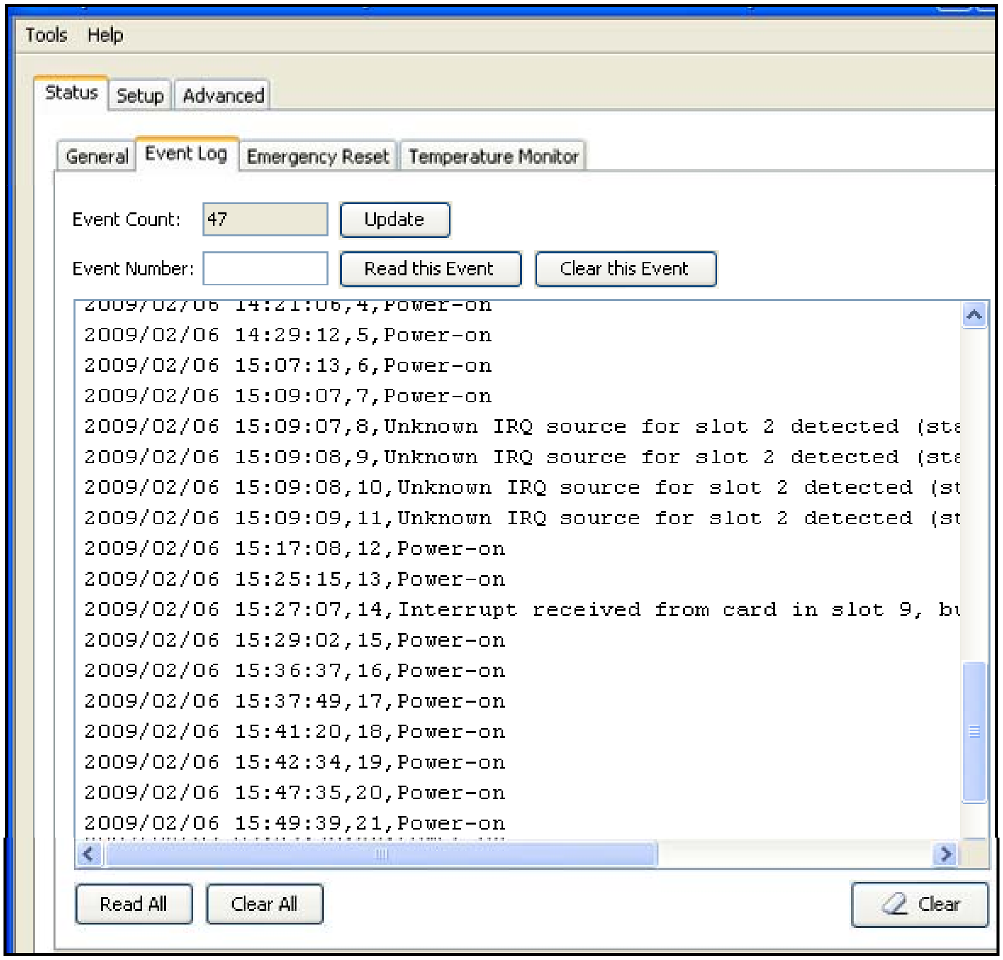


Figure 4-9: System Controls, Status/Event Log Tab

Status/Emergency Reset Tab

On this tab (Figure 4-10), you can see if there are any current emergency resets occurring. If you have resolved an emergency reset condition, click **Update** to see the current status of the system. See the **Emergency Reset Status** section later in this manual to determine actual card status.

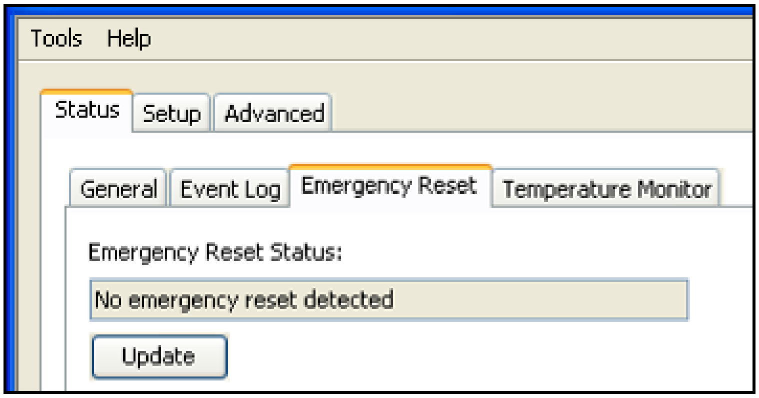


Figure 4-10: System Controls, Status/Emergency Reset Tab

Status/Temperature Monitor

On this tab (Figure 4-11), you can view the current temperature condition of various thermistors mounted at different points in the chassis.

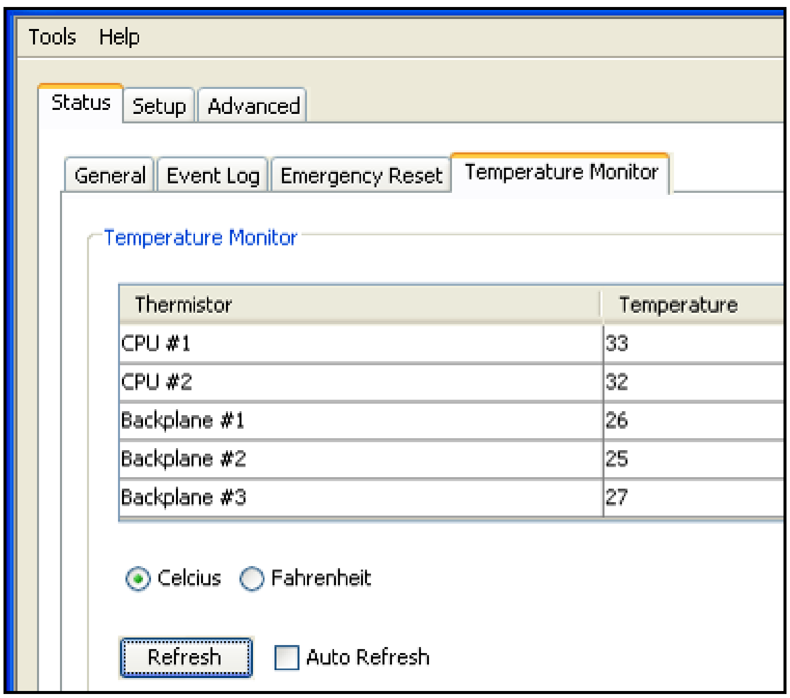


Figure 4-11: System Controls, Status/Temperature Tab

Setup Tab

The **Setup** tab (Figure 4-12) allows you to save and recall 1830 system relay states as well as change or update the real time clock.

Save a relay state by setting a location using the drop-down box and then clicking **Save**.

Recall a saved relay state by selecting the location the state is in and clicking **Recall**.

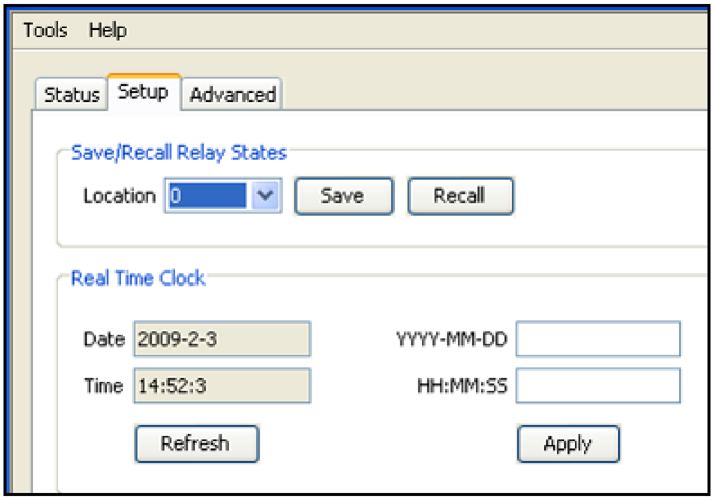


Figure 4-12: System Controls, Setup Tab

Advanced Tab

The **Advanced** tab (Figure 4-13) allows you to set or modify the following 1830 system conditions.

- Path names
- Reference clock
- Channel attributes
- Scan list and settings
- SCPI
- Sequence
- Trigger settings and routing
- Analog Bus control and cycles
- Data results
- Include/Exclude lists
- Monitor list, settings, and alarm source

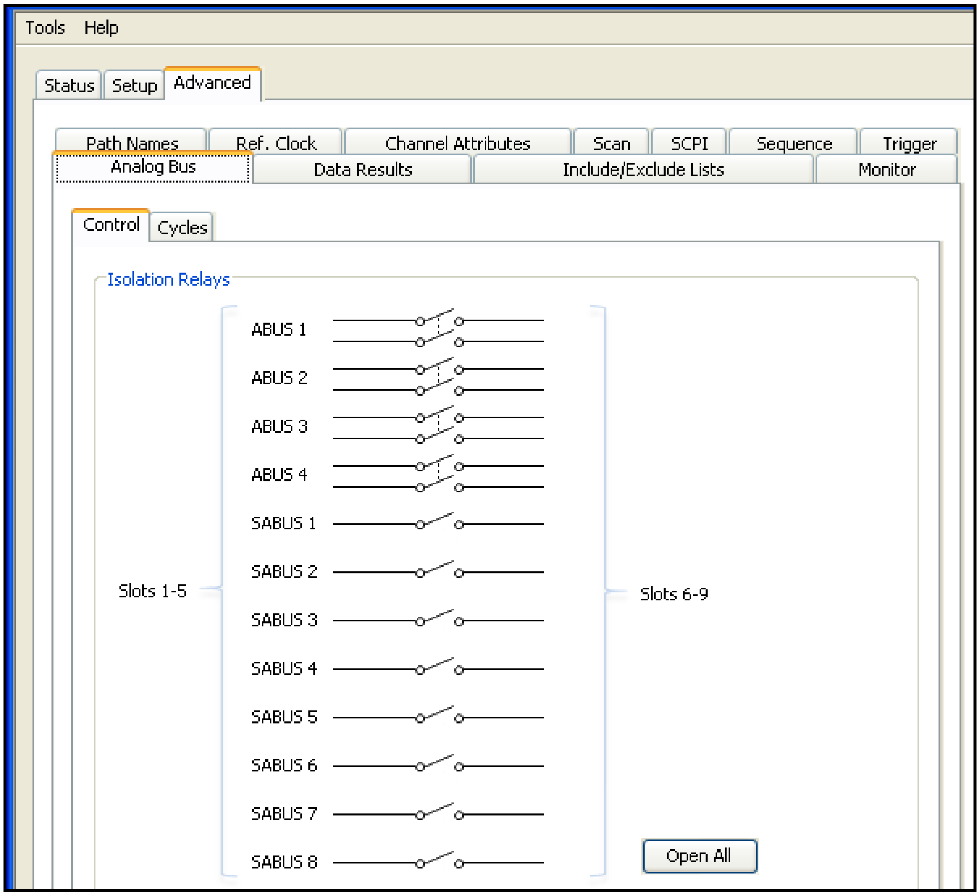


Figure 4-13: System Controls, Advanced Tab

Plug-In Card Controls

Each installed plug-in card (module) has set of customized web pages to enable you to view status and modify a variety of conditions and parameters on your card. Refer to the following sample figures for additional information.

Relay Control

The **Relay Control** tab (Figure 4-14) allows you to change the conditions on the various relays on the card. Simply click the appropriate relay to change the state of the relay on the card. The relay change is immediate.

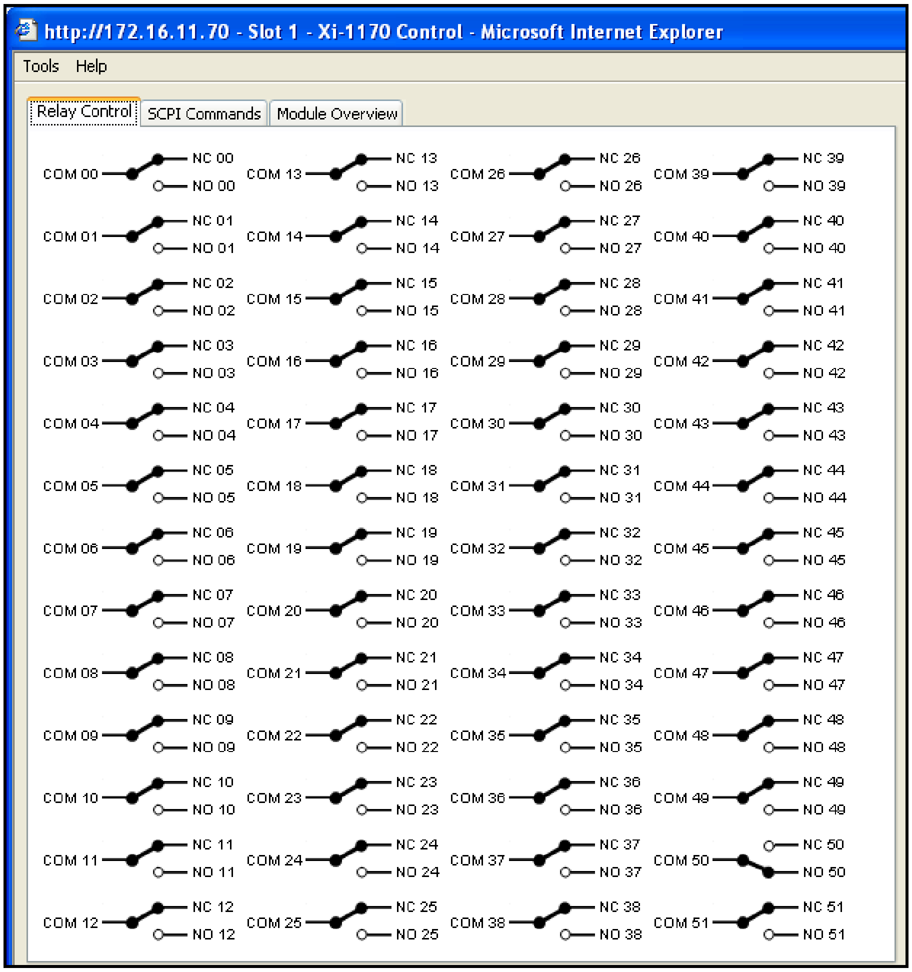


Figure 4-14: Board Control, Relay Control Tab

SCPI Commands

The **SCPI Commands** tab (Figure 4-15) allows you to send SCPI commands to the card. A few common SCPI commands have clickable icons along the top of the tab including:

- **SYST:ERR?:** Checks on and reports if there are any errors
- **Read STB:** Reads the value of the Status Byte Register
- **Device Clear:** Clears error queue and flushes all input and output buffers
- **Clear:** Clears any text in the text message box below the icons

Use the **SCPI Command** input box to enter SCPI commands. The command and any replies will be shown in the text message box.

- Clicking **Send** sends the command.
- Clicking **Read** receives the reply.
- Clicking **Send/Read** both sends and then receives the reply.

- Clicking **Clear** clears the commands history in the **SCPI command** input box

Additional information and a listing of SCPI commands can be found in **Chapter 7, SCPI Command Basics**.

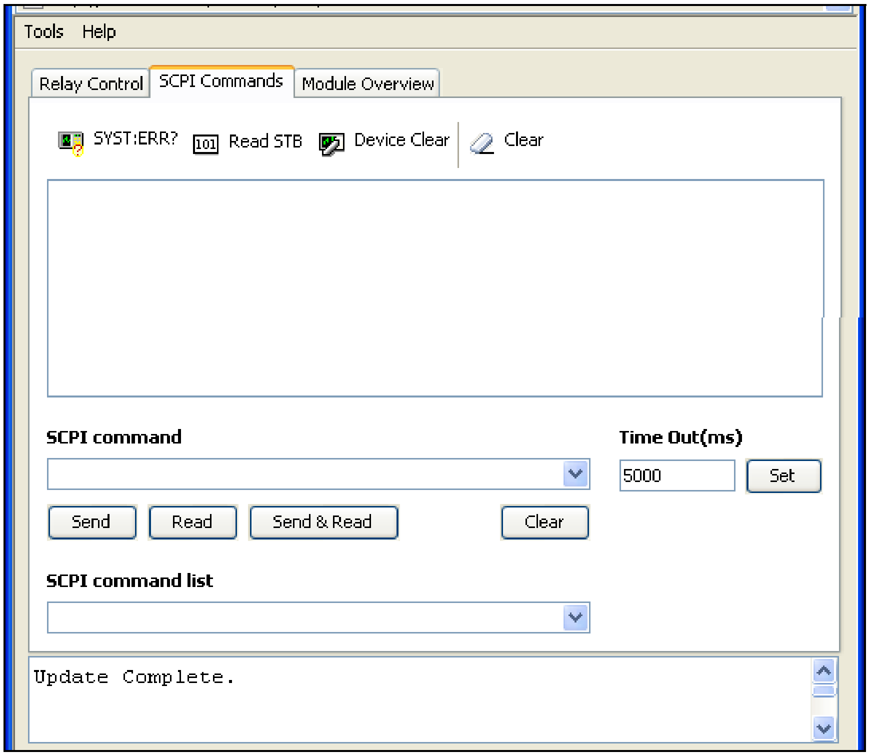


Figure 4-15: Board Control, SCPI Commands Tab

Module Overview

The **Module Overview** tab (Figure 4-16) allows you to review current card information including model and serial number, revision levels, channel/state configuration, relay cycle counts, and emergency reset status.

Clicking **Generate Report** generates a status report. Clicking **Clear Report** removes the information from the screen. Clicking any of the checkboxes on the left side before you click **Generate Report** adds this additional information to the generated report.

See the **Emergency Reset Feature** section later in this chapter for information on the Emergency Reset Status report.

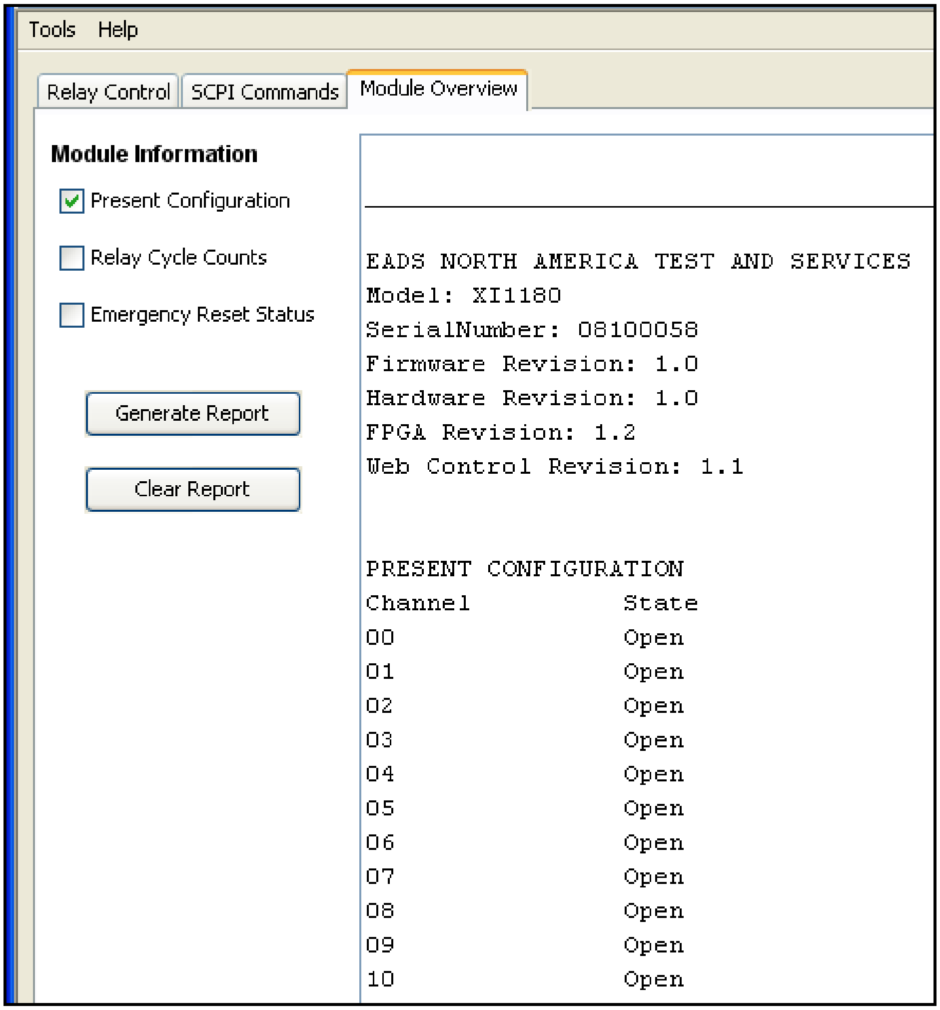


Figure 4-16: Board Control, Module Overview Tab

DMM Board Controls

If you have a DMM board option installed, refer to the **Web-Page Controls** section in **Appendix C, Digital Multimeter Option** for information about the DMM control screens.

Emergency Reset Feature

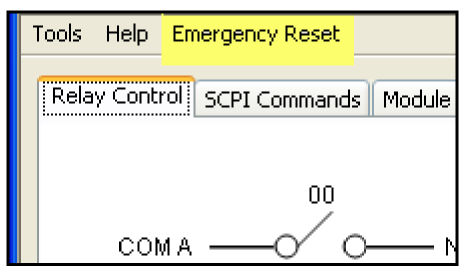
The 1830 system includes an emergency reset feature to prevent system and card damage due to situations such as over-current conditions. The reset condition is activated by a card with that feature – such as the 1220 16-Amp SPST plug-in switch card.

There are two reset conditions: global and local. A global reset condition quickly opens all the relays in the 1830 system including those on the system backplane and on the installed cards. A local reset condition opens the relays only on the affected card. The default factory assembly setting for the 1220 card is for global reset.

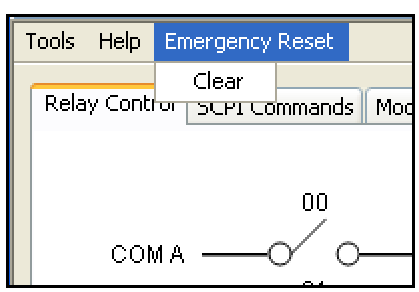
When an emergency reset occurs, a series of screens appear prompting you to perform certain tasks.

1. A warning message states where the emergency reset occurred. (**Front Panel** refers to the external connector side of the board.)

2. Remove the fault condition before proceeding.
3. The **Emergency Reset** menu appears on the menu bar of the card.

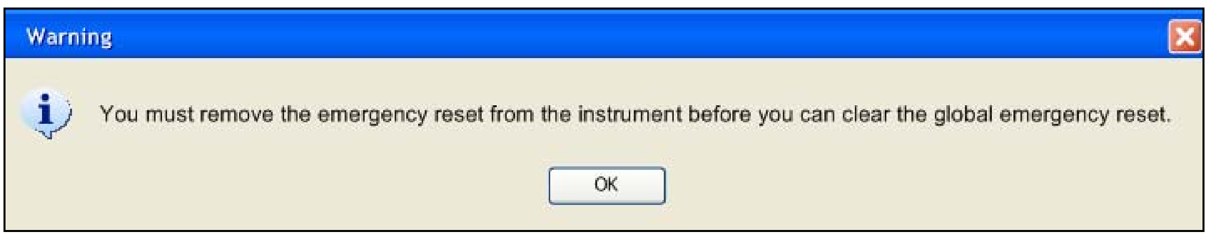


4. Select **Emergency Reset** > **Clear**.



5. A message box asks for confirmation that the fault condition has been removed or corrected. Click **Yes**, if correct.

- After you click **Yes**, the system internally issues a *RST command to reset the system. If the fault condition was successfully removed, the system should be operational again. However, if not, the message box in Step 1 appears again.
- If you clicked **No**, a message box (below) appears to remind you to remove the fault before the system clears the emergency reset.



Emergency Reset Status Information

On the 1830 **System Control** screen, there is an **Emergency Reset** tab (Figure 4-17) which shows the current status.

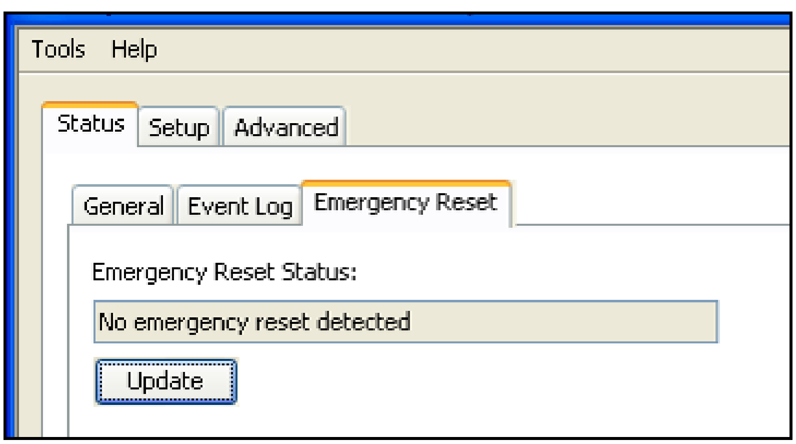


Figure 4-17: Emergency Reset Status Tab

On the **Module Overview** tab of the plug-in card web page, you can generate a detailed status report if you select **Emergency Reset Status** and then click

Generate Report. If the card has an emergency status circuit (such as the 1220), you receive detailed information. See Figures 4-18 and 4-19 for examples of the report you would see on the 1220 and 1380 cards.

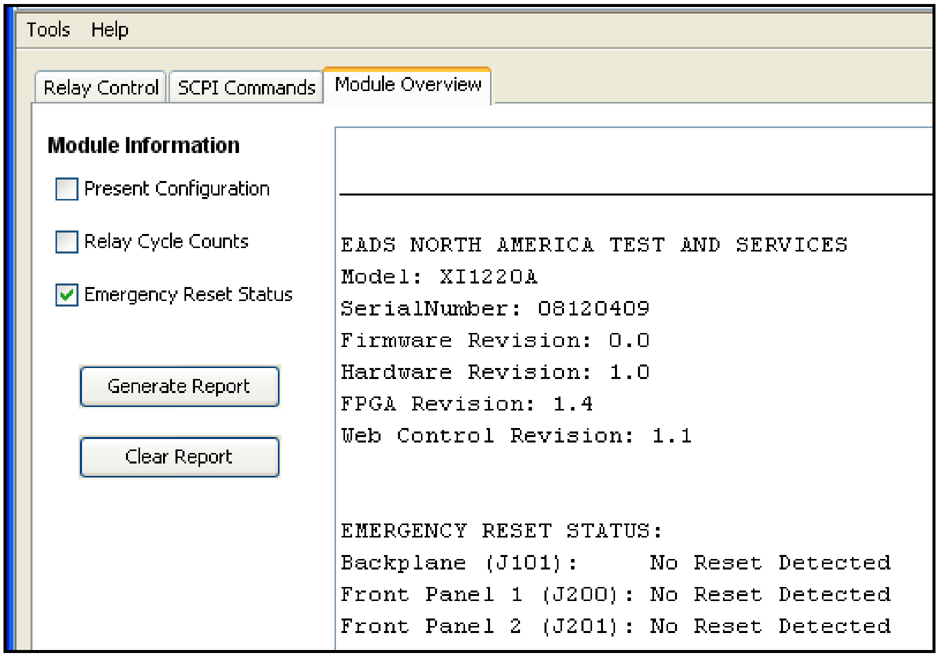


Figure 4-18: Emergency Status Screen on a 1220 Card

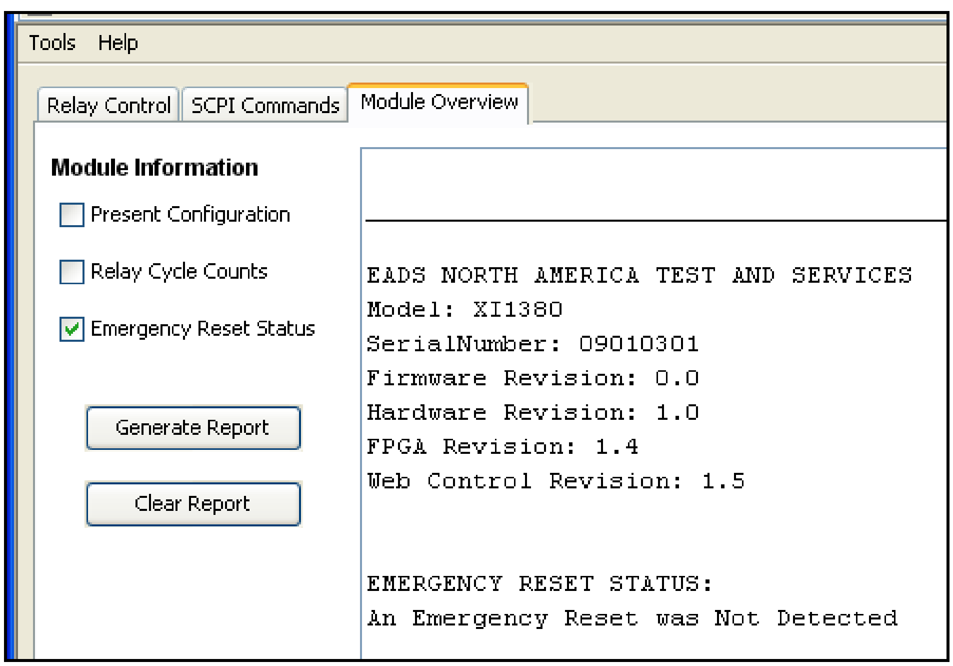


Figure 4-19: Emergency Status Screen on a 1380 Card

Firmware Upgrade

The embedded software (firmware) supports downloading newer versions from our support website so that the 1830 unit does not have to be returned to the factory for a software update. The firmware uses the built-in web-page interface to upload the software through the 1830 Ethernet port.

When there's a new 1830 firmware upgrade available, it'll be downloadable from the EADS North America Test and Services website. Once you've downloaded the new firmware upgrade file to your computer, you're ready to perform the upgrade.

On the Firmware Upgrade web page, click **Browse** and select the firmware upgrade file you've downloaded (if a zip file was downloaded, unzip it and select the file that was unzipped). It is important that you do **not** change the extension (.bin) of the upgrade file. After selecting the upgrade file (with the ".bin" extension), click **Upgrade** to perform the firmware upgrade on the 1830.

Note: Do not power off the 1830 while the upgrade is taking place. However, if the upgrade was interrupted and the firmware has been corrupted, the 1830 loads the backup firmware (with a minimal feature set) and you must upgrade the firmware again.

1. Download the firmware upgrade file to a location on your computer.
2. Open the home web page of the 1830 system to upgrade.
3. Select and open the **Firmware Upgrade** link on the navigation bar.
4. Use the **Browse** button to select the file downloaded in Step 1. (See Figure 4-20.)
5. Click the **Upgrade** button. The software begins to upgrade your current firmware. (See Figure 4-21.)

Caution

DO NOT power off the 1830 while the upgrade is taking place. If the upgrade is interrupted and the firmware is corrupted, the 1830 automatically loads the backup firmware. This version includes only a minimal feature set and you must upgrade the firmware again.

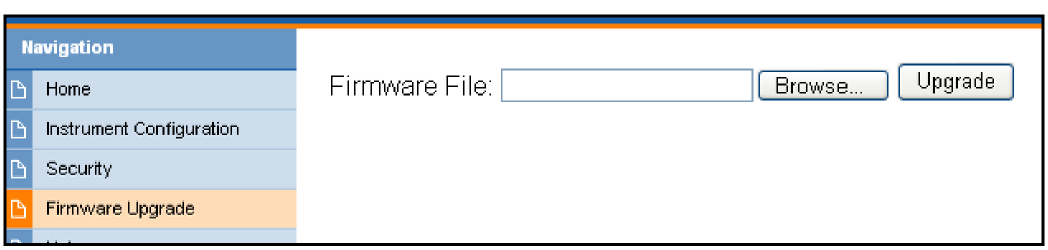


Figure 4-20: Firmware Upgrade File

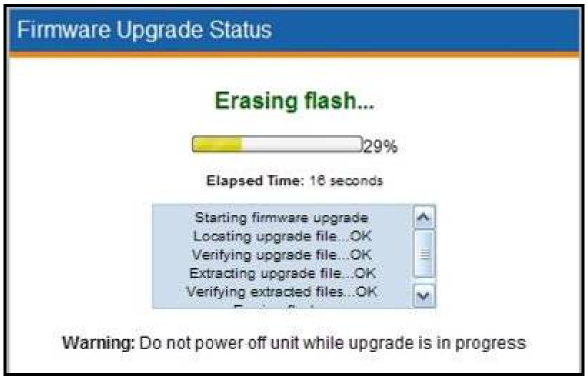


Figure 4-21: Firmware Upgrade Status

6. When the upgrade is complete (Figure 4-22), exit out of the message box.

Figure 4-22: Firmware Upgrade Complete

7. Turn the system power off and then back on, in order to restart the system and initialize the firmware upgrade.

This page was left intentionally blank.

Chapter 5
ActivATE Test Platform

This chapter discusses the ActivATE™ Test Executive Platform which is an option to operating the 1830 Source/Measure Switch System.

Included are:

- Overview, key features, and a product summary of the ActivATE software
- Installing the ActivATE software
- Installing the ActivATE drivers
- Configuring the software to recognize the installed devices
- Registering the program
- Understanding the ActivATE screen controls
- Adding users
- Changing the password
- Structuring your ActivATE test program
- Additional help
- Uninstalling the software

Overview

The ActivATE Test Platform is a full-featured test development and sequencing platform for your automated test equipment (ATE) requirements.

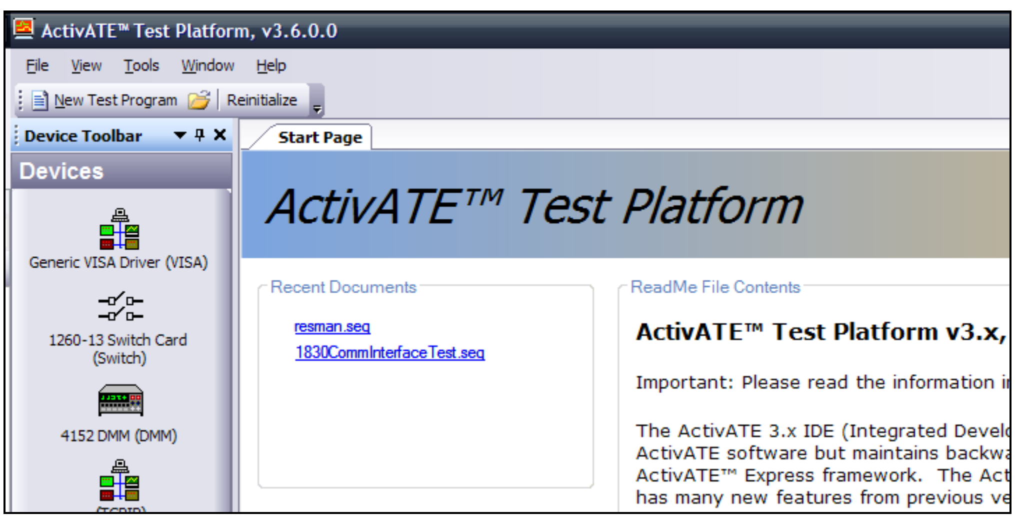


Figure 5-1: ActivATE Main Screen

The ActivATE software allows you to quickly auto-configure the environment for the correct driver set that matches your hardware configuration. The software provides sophisticated, yet easy-to-use, tools that allow the test engineer to focus on quickly developing and finalizing product testing. These benefits result in a reduced learning curve and rapid application development allowing you to save time and costs.

The Model 1830 Source/Measure switching system works seamlessly with ActivATE test software. A 90-day, full-featured evaluation copy of the software is included on the 1830 installation disk. Dedicated drivers for the 1830 and its plug-in instrument cards are also included to use with your configured 1830 system.

The installation disk also includes two short informational videos. One provides an overview of the ActivATE software and the other shows how to configure the software with your 1830 system.

In order to use the included drivers to control the 1830 system and plug-in cards, you need to have a current version of VISA runtime already loaded on your computer system.

Key Features

Key features of the ActivATE platform include:

- Very intuitive and easy to use
- Test results can be saved in many formats (PDF, Word, RTF, HTML, CSV, etc)
- Leverages off the latest Microsoft .NET technology and its ability to adapt as trends evolve
- Excellent connectivity with databases, applications, COM, .NET, Web Services
- Integrated Development Environment (IDE) provides extensive feedback to the user, both during development (the test engineer) and manufacturing test (the factory-floor operator).
- Fully integrated debugger for stepping through test programs (watch windows, breakpoints, etc.)
- Extensible, user-configurable environment (GUI displays, driver set, add-in modules, help, etc.)
- Very modular and open architecture
- Built on open standards
- Microsoft Visual Studio® templates and wizards for user-defined drivers and modules
- All instrument driver source code available for download
- True multi-threaded environment
- Supports up to 36 test programs (or test heads) running simultaneously
- Test engineers can develop and execute code while other test programs are running.

- Both a development platform and runtime execution environment. No need to buy costly sequencers after the test has been developed.
- Fully supported website for FAQs, forums, downloads, and more
- Secure multi-level log-in
- Drag-drop features for instrument drivers, test code, and test nodes

Product Summary

- Test programs are written in VBScript, a widely-used and well-known scripting language. This eliminates the need to learn a proprietary programming language and thus greatly reduces development time.
- The ActivATE software is designed to follow the familiar look and feel of the standard Microsoft applications, making the software intuitive and easy to use. Examples of this are:
  - To create a new test program, simply click **New Test Program** from the File menu.
  - An Outlook®-style toolbar (like the one found in Microsoft Outlook 2003) provides access to the driver GUIs (Graphical User Interfaces) and quick driver configuration options.
  - Multiple open documents are tabbed within the main document window.
  - To optimize screen space, all windows in the environment are dockable and can be moved anywhere within the main form or they can be hidden. Arrows are provided as docking hints to help the user select the destination for the window as the window is dragged across the screen.
  - Users create and organize test sequences graphically in a Windows Explorer-like tree view, but the test program logic is actually located in the code "behind" each test and group node. Both the sequence and the code of the test program are easy to access and modify within the ActivATE environment.
  - A powerful text and code editor is included, similar to the code editor used in Visual Studio. This editor includes many tools and features, such as IntelliSense®, tooltip help, and syntax color-coding, which provide a quick reference to the test program author during development and significantly reduce the time required to develop a test program.
  - Property Grid controls, as found in Visual Studio, are used to provide fast, easy access to multiple properties and settings at once.
- The ActivATE environment is very flexible and extensible. Any number of drivers and custom add-in modules may be added to provide additional functionality.

  For example, you may want to encapsulate your business logic in a module that is separate from the test program, or you may want to add third-party packages such as Microsoft Excel®. Any .NET or COM assembly may be added to the ActivATE environment, and full IntelliSense is provided in the test program editor.

- Drivers are easily added to and removed from the ActivATE environment via the Configuration pane in the Outlook-style toolbar. This interface provides a quick view of all the drivers that are available within the ActivATE environment.

  Expand a driver node to expose the APIs (the Application Programmable Interface, or the public methods and properties in the driver class) that are available to use from the test program. Right-click in the tree view to show options for adding and deleting drivers.

- A Software Development Kit (SDK) for the ActivATE software is available for download from the ActivATE website. The SDK includes a driver wizard with C# project templates for Visual Studio 2005 and/or Visual Studio 2008.

  These project templates facilitate the development of add-in modules and instrument drivers for the ActivATE Test Platform. Although use of the driver wizard is not required, it is the preferred method of developing a driver as it pulls together a set of base classes that give immediate functionality to the module. The SDK is also compatible with the Visual Studio C# Express editions which are available as a free download from Microsoft.

Installing the ActivATE Software

Note: In order to use the included drivers to control the 1830 system and plug-in cards, you need to have a current version of VISA runtime already loaded on your computer system. This could include the VISA product from National Instruments, among others.

1. Click on the **ActivATE Software** link from the home page of the installation disk.

   The main ActivATE launch page appears (Figure 5-2). From here you can:

   - Install the ActivATE software
   - View the release notes
   - Install the ActivATE drivers for the 1830 and switch cards
   - View the software help files
   - View a short video showing how to use ActivATE
   - View a short tutorial showing how to configure ActivATE on your system.
   - Connect to the ActivATE website which features additional information and technical assistance.

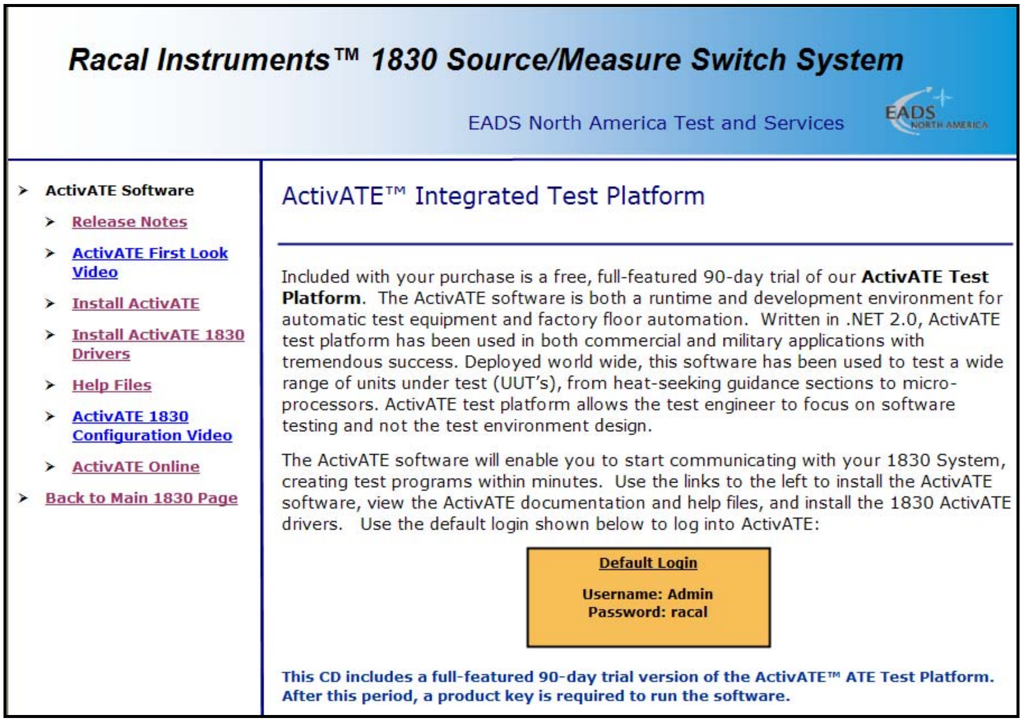
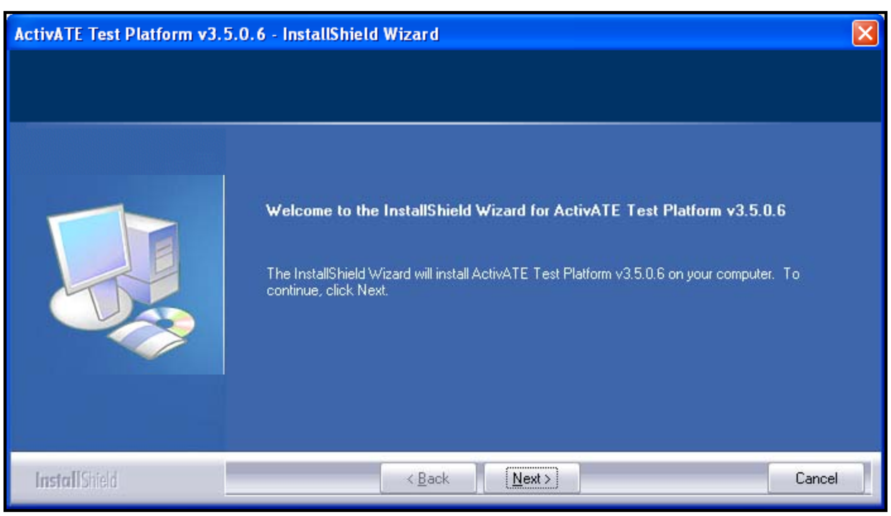


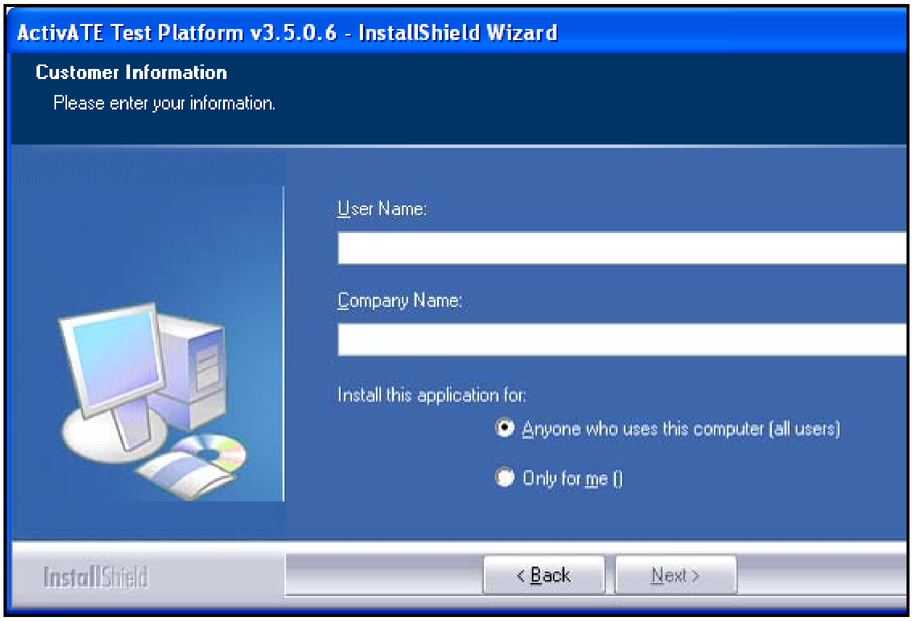
Figure 5-2: ActivATE Program Launch Screen

2. Click **Install ActivATE** to install the program.
3. Click **Run** when asked if you want to run or save the file.

   You might see a second message box noting that the publisher could not be verified and asking if you still want to run the program. Click **Run** to continue.
4. The InstallShield Wizard for the ActivATE software appears. Click **Next**.



5. Type in your user and company name. Click **Next** when done.

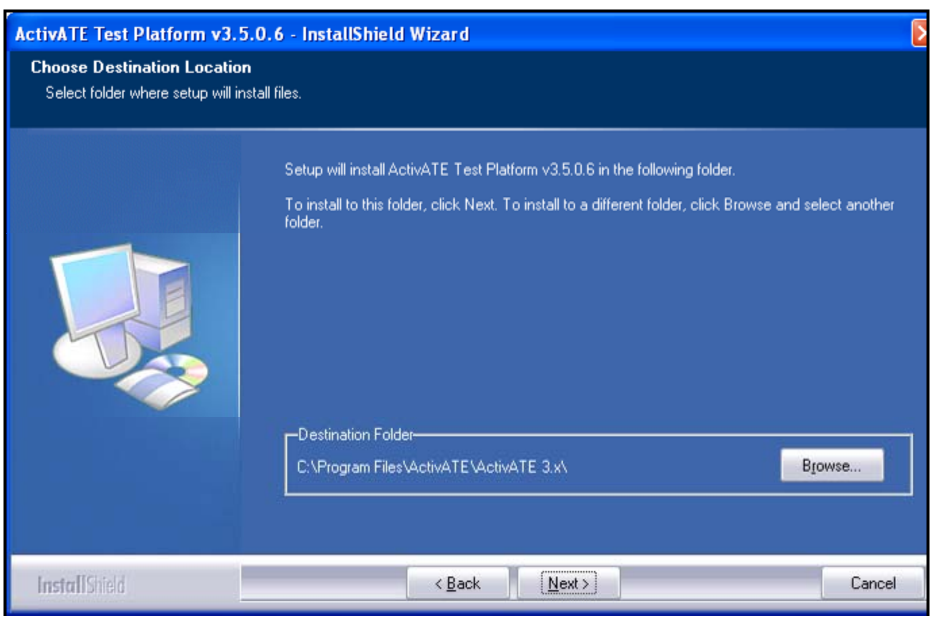


6. On the next screen, review the terms of the License Agreement and accept the license agreement. Click **Next** when done.

   Should you not accept the terms, click **Cancel** to halt the installation of the software.

7. Confirm or change the destination folder to install the files. Click **Next**.

   The ActivATE software installs on your computer.

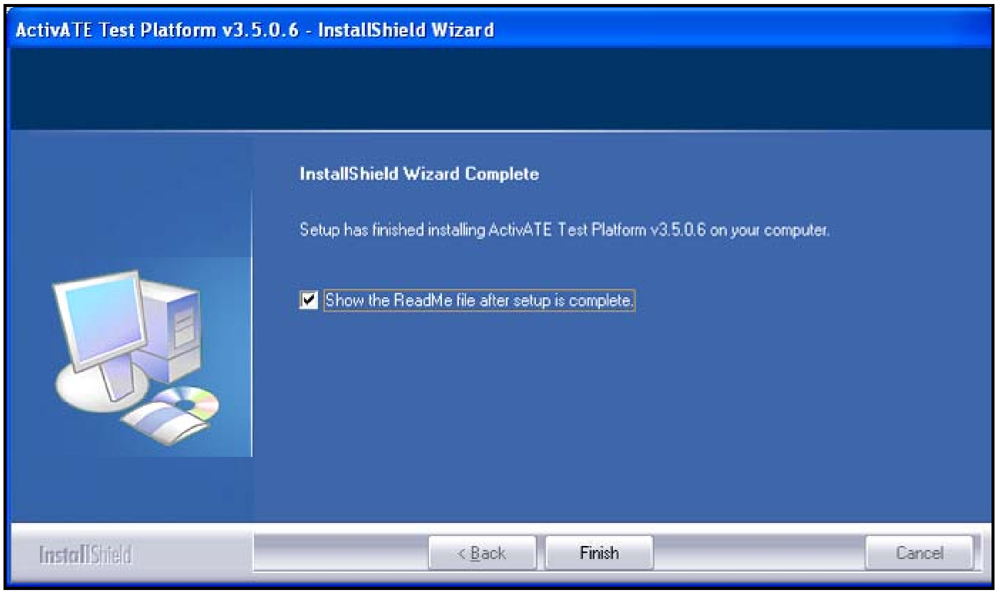


8. If the setup was successful, a screen appears showing **InstallShield Wizard Complete**.

   By default, the release notes ReadMe file appears after the setup is complete. Clear the check box if you would prefer not to read it.

   Click **Finish**.

   At this time, you can move to the next step of installing the ActivATE drivers for the 1830 source/measure system or you can review the release notes.



Release Notes ReadMe File

By default, the ReadMe file opens after the installation is complete. This file is also available by selecting **Release Notes** on the ActivATE software launch page.

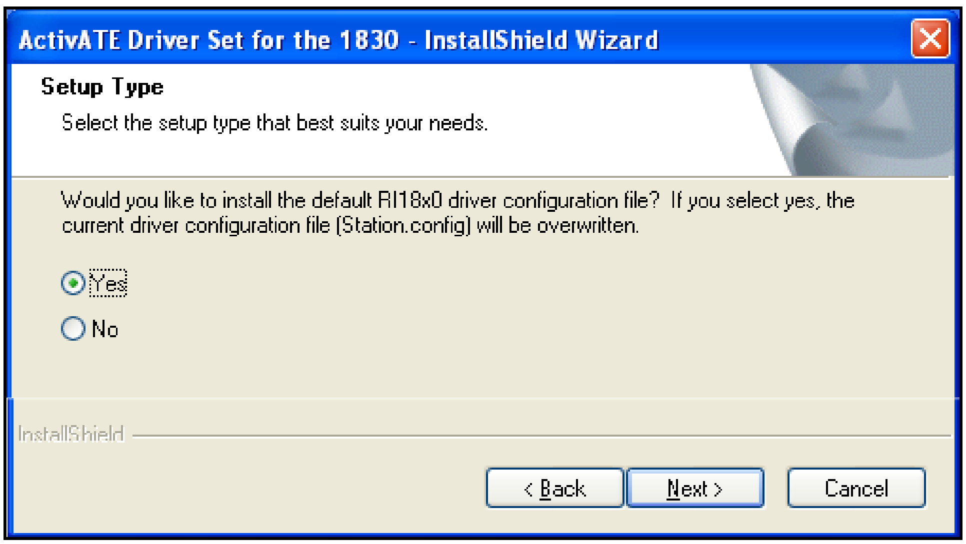
The file contains information including:

- Default logins
- Pre-installation requirements
- Uninstalling previous versions of ActivATE software
- Installations using the .exe file.
- Troubleshooting
- Notes for the various release versions
- Known issues and concerns
- Copyright notice

Installing the ActivATE Drivers for the 1830 System

1. On the 1830 install disk home page, click the link to **Install ActivATE 1830 Drivers**.
2. Click **Run** when asked if you want to run or save the file.

   You might see a second message box noting that the publisher could not be verified. If so, click **Run** to continue.
3. Click **Next** to continue the installation.
4. Enter your User and Company name.
5. Review the terms of the License Agreement and click the option button to accept the license agreement. Click **Next** when done.
6. Confirm or change the destination folder to install the files. Click **Next**.
7. Keep the default option set to **Yes** to install the default driver configuration file. Click **Next**.



8. Click **Install**.
9. Click **Finish** when the InstallShield Wizard is complete.

This auto-installation includes drivers for the 1830 system as well as for all the currently released 1830 switch cards and DMM modules.

After the installation is complete, you need to configure the ActivATE software to recognize the cards installed in your 1830 system.

Configuring the Software to Recognize the Installed Devices

Obtaining the IP Address

Before configuring the ActivATE software, identify the IP address of your 1830 system. The steps below assume that the 1830 system is successfully connected to a network and the system power is turned on.

1. Open up the web page for the 1830. (Refer to **Chapter 4, Web-Page Interface.**)

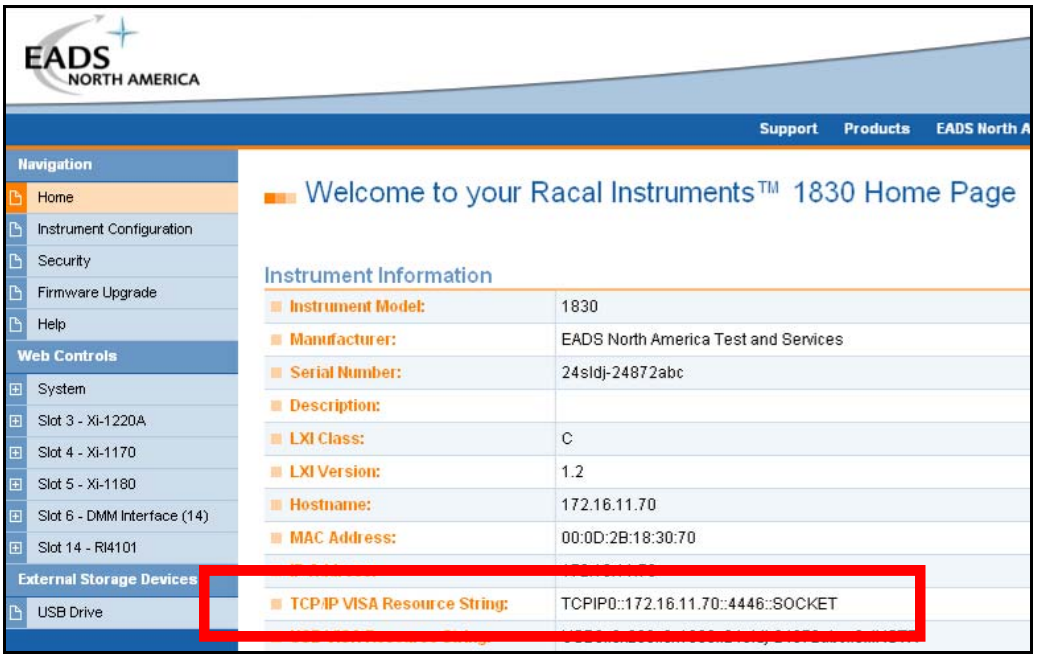


Figure 5-3: Obtaining the IP Address

2. Copy the entire **TCP/IP VISA Resource String** from this page for use in the next section.

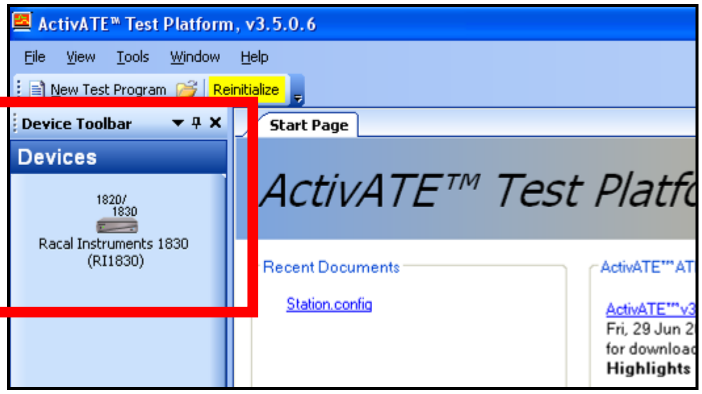
Default Password for ActivATE Software

The default login username and password is

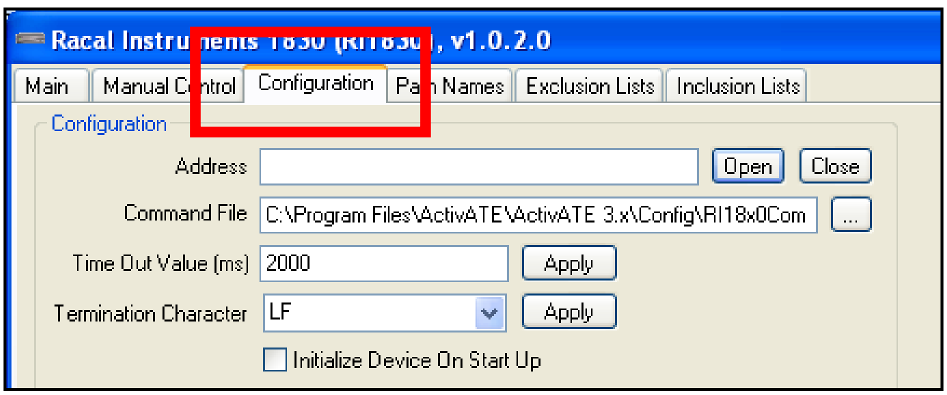
- User ID: admin
- Password: racal

Configuring the Software

1. Launch the installed ActivATE program.
2. Enter user ID and password, and then click **OK**.
3. On the **Devices** toolbar, click on the 1830 system icon.



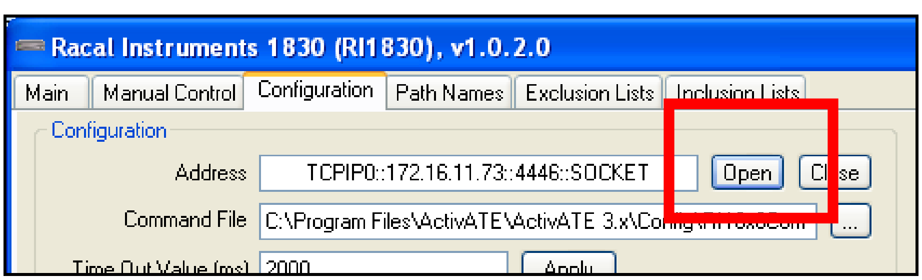
4. When the GUI screen appears, select the **Configuration** tab.



5. Enter the TCP/IP VISA Resource String information recorded in the previous section into the **Address** box.

   Be sure to not include any extra spaces before or after the string information.

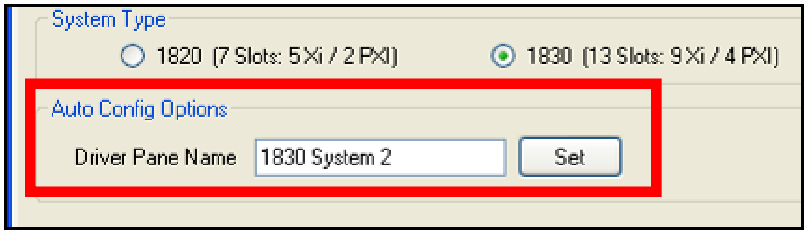
   When complete, the **Address** box looks similar to the following figure.



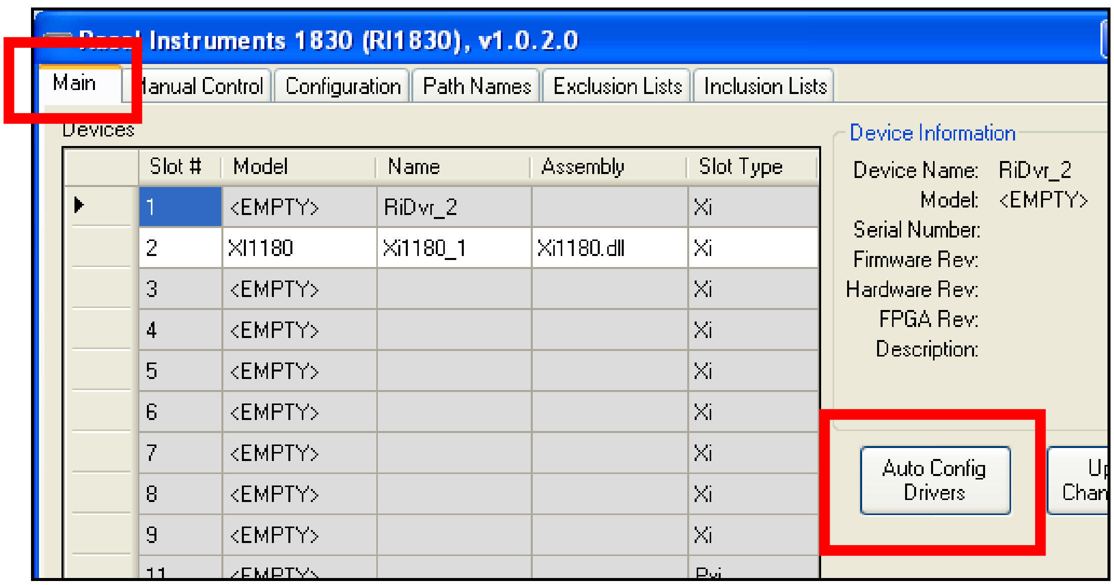
6. Click the **Open** button.

If you receive an error or some similar message, check to be sure you have copied the address correctly and with no extra spaces.

7. You can rename the tab on the Device toolbar which shows the available cards by changing the **Driver Pane Name** under the Auto Config Options. Click **Set** when done.

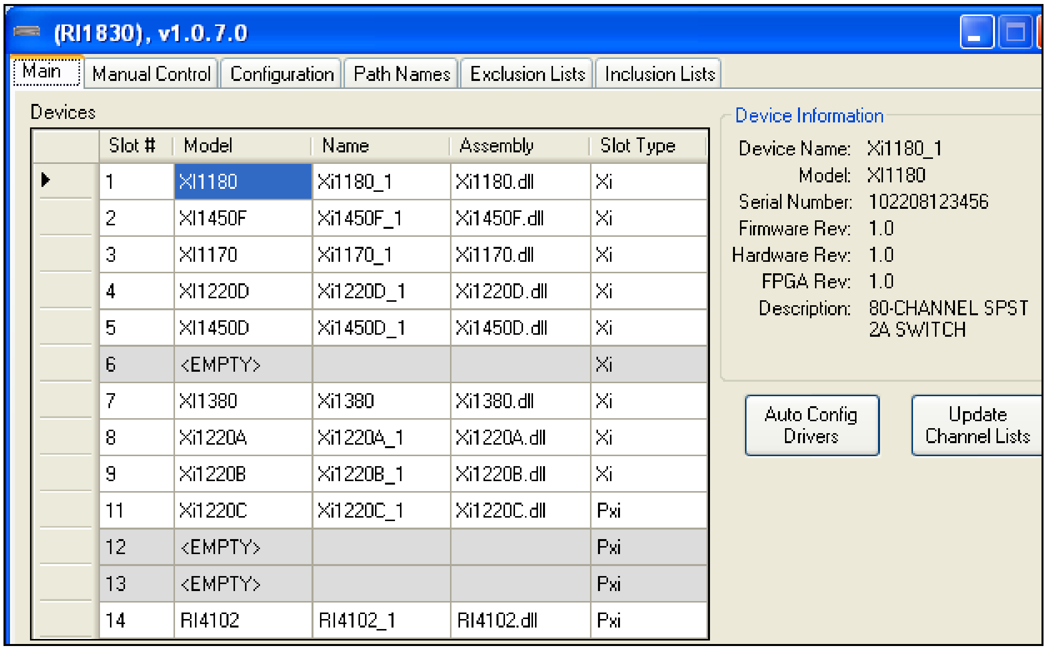


8. Click on the **Main** tab.

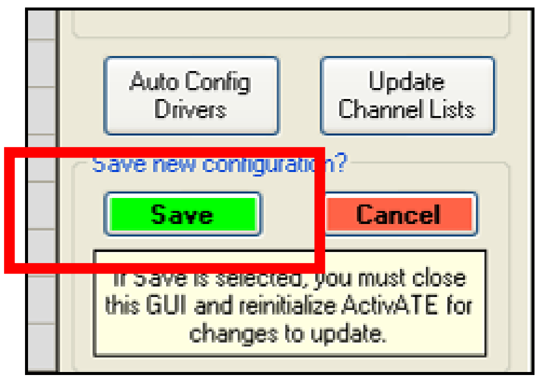


9. Click on the **Auto Config Drivers** button.

   The software automatically reviews the available cards in your 1830 and updates the Main device information table similar to what you see in the following illustration.

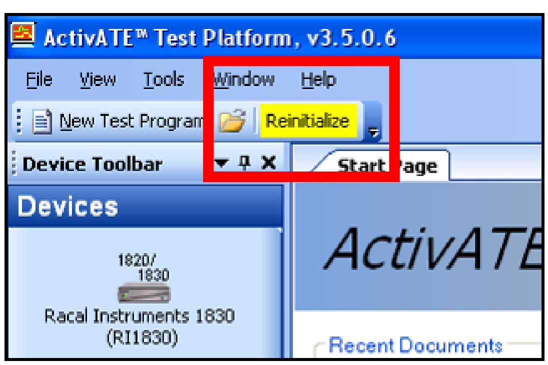


10. Click **Save** to save the new configuration.

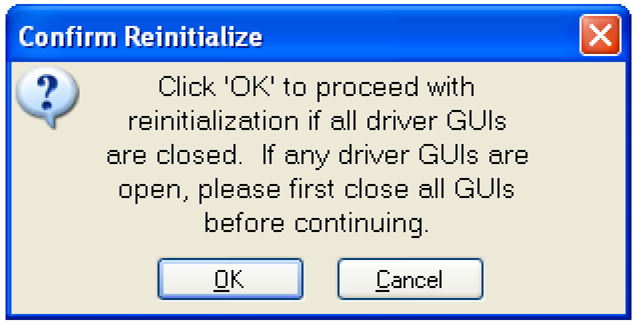


11. Close the GUI screen.

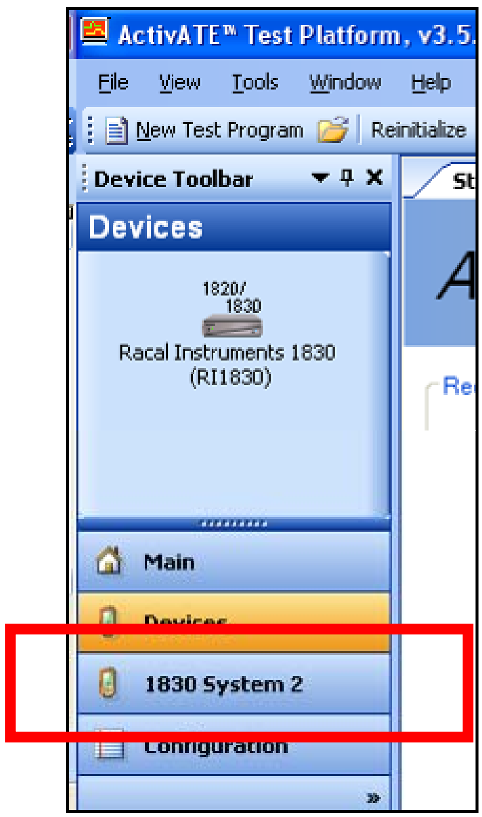
12. Click the **Reinitialize** button on the ActivATE menu bar.



13. Click **OK** to proceed with the reinitialization.

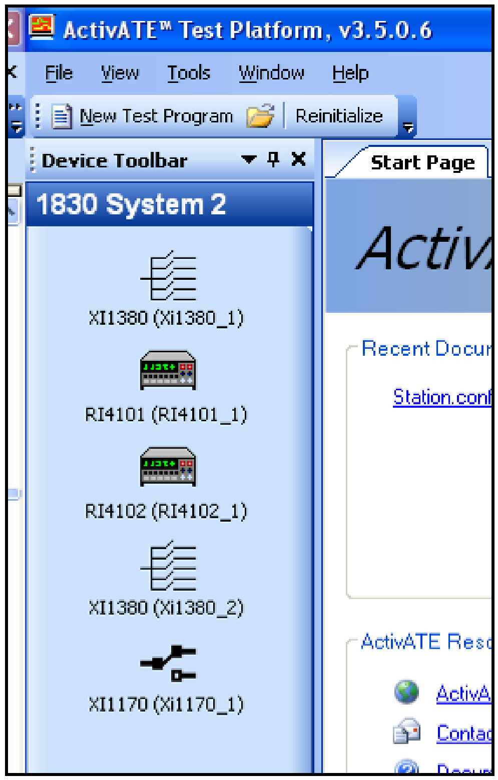


14. Select the **Device Toolbar** tab which has the Driver Pane Name set in Step 7.



15. The icons shown on the toolbar represent the various cards and DMMs installed in your 1830 system.

    Clicking on the device icon brings up the control screens for that device. From there you can control or program the device. A few of the device screens look slightly different in the ActivATE software than when you are viewing them directly from the 1830 system with your browser.



Registering the ActivATE Program

After you purchase a license to use the ActivATE program, a product key number is emailed to you. To register your product, perform the following.

1. Open the ActivATE program
2. Select the **Help** menu and click on **Register Product**.
3. Fill in the information on the Product Registration form.

   If you do not have internet access with this computer, click on the appropriate **Click here** line under the Submit button. That screen gives you information on how you can register your software.
4. Click **Submit**

Your program is now fully activated and you won't see the **Evaluation Copy** message box upon start up.

Activate Screen Controls

The User Interface (UI) for the 1830 system driver (Ri18x0) has multiple tabs, allowing the operator to fully configure and control each type of platform from the front panel.

All of the tabs have three similar selections:

- **Reset Button**: Clicking this button sends the *RST SCPI command to the system. This command completely resets the system to an original "power-up" condition where all relays are open and all controls are reset to the default..
- **Help Button**: Click this to bring up the 1830 system help file. You may also press F1 to view the help.
- **Simulation Checkbox**: Check this to put the system driver in simulation mode. In simulation mode, the driver maintains full functionality (the user may still access all API's and GUI controls) but does not communicate with the hardware or change system settings.

Main Tab

The **Main** tab (Figure 5-4) displays the current state of the cards (devices) in the 1830 system.

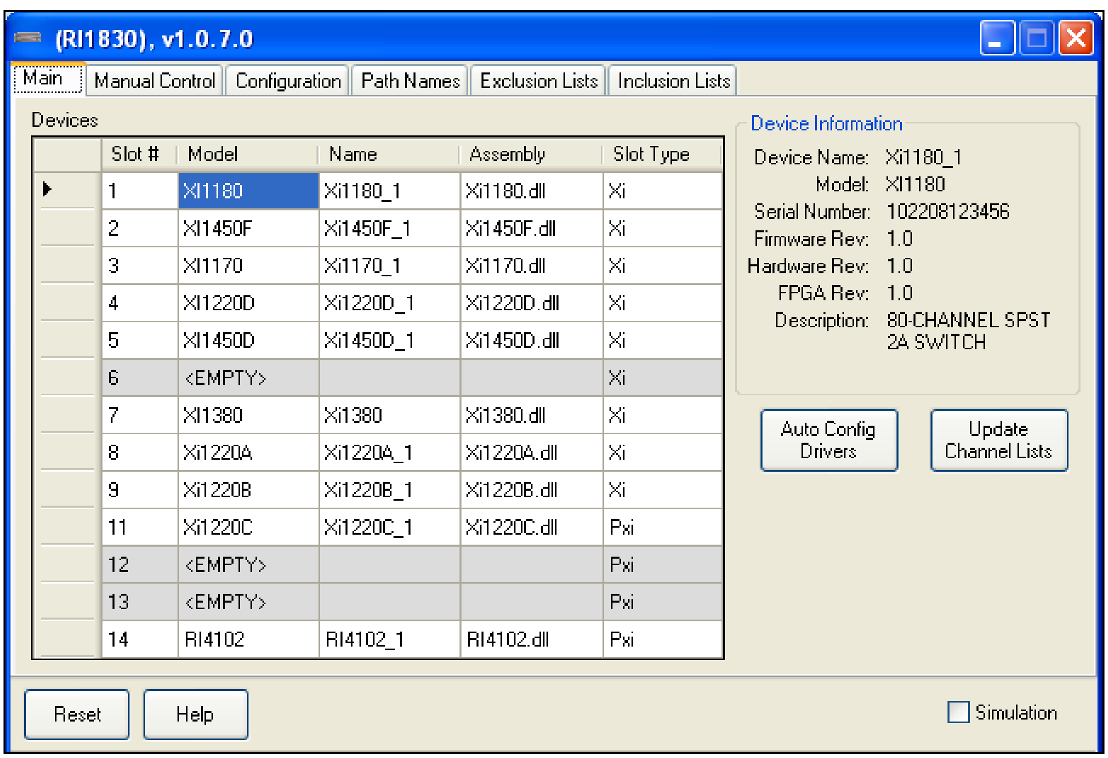
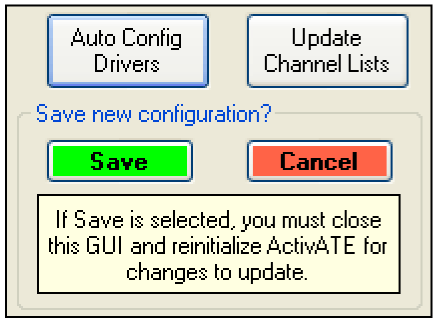


Figure 5-4: ActivATE Main Tab

Selections for this tab include:

- **Devices Table**: Displays the current configuration of devices in the 1830 platform. Right click on any slot number (#) to manually configure the slot with the device information and driver file.
- **Auto Config Drivers**: Click this button to configure the driver automatically with the information and drivers of the installed devices. When this button is selected, the user will have the open to save the generated configuration or cancel it, as shown below.



As the note in the GUI states, you must close all GUIs and reinitialize the ActivATE environment if "Save" is selected. The "Reinitialize" button will be highlighted and located on the main ActivATE toolbar.

- **Update Channel Lists:** Click this button to update the channel lists that are available on the other tabs (i.e. Exclusion/Inclusion Lists tabs). The lists will update according to the new device configuration.

Manual Control Tab

The **Manual Control** tab (Figure 5-5) allows the user to perform low-level Command and Query operations (read/writes) to the system through standard SCPI commands. All cards installed in the 1830 platform can also be controlled through this manual control tab.

The Manual Control Tab allows the user to perform low-level Command and Query operations (read/writes) to the device. All cards installed in the 1830 platform can also be controlled through this manual control tab. The 1830's commands use a channel descriptor to select the channel(s) of interest when controlling an installed device. The syntax for a channel descriptor is the same for all 1830 series modules.

In general, the following syntax is used to select a single channel for most commands:

Where:

<slot address> is the slot number of the device in the 1830 platform

This is a number in the range from 1 through 9, inclusive, for all the switch cards. DMM cards are internally installed in PXI slots 10 through 14. Additional "interface cards" are used to access these DMM cards and they can be located in slots 1 through 9. However, use the actual DMM PXI slot number for the command.

<channel> is the 1170 channel to operate. Valid channel numbers are from 0-51. Multiple individual channels may be specified using the following channel descriptor syntax:

```
@ <slot address> ( <chan1> , <chan2> , . . ., <chanN> ))
```

A range of channels may be specified using the following channel descriptor syntax:

```
@ <slot address> ( <first channel> :<last channel> ))
```

The following examples illustrate the use of the channel descriptors for a relay card installed within the device:

OPEN (@8(0))	Open channel 0 on the switch card that has slot address 8.
CLOSE (@8(0,3))	Close channels 0 and 3 on the switch card that has slot address 8.
CLOSE (@2(10:13))	Close channels 10 through 13 inclusive on the switch card that has slot address 2.

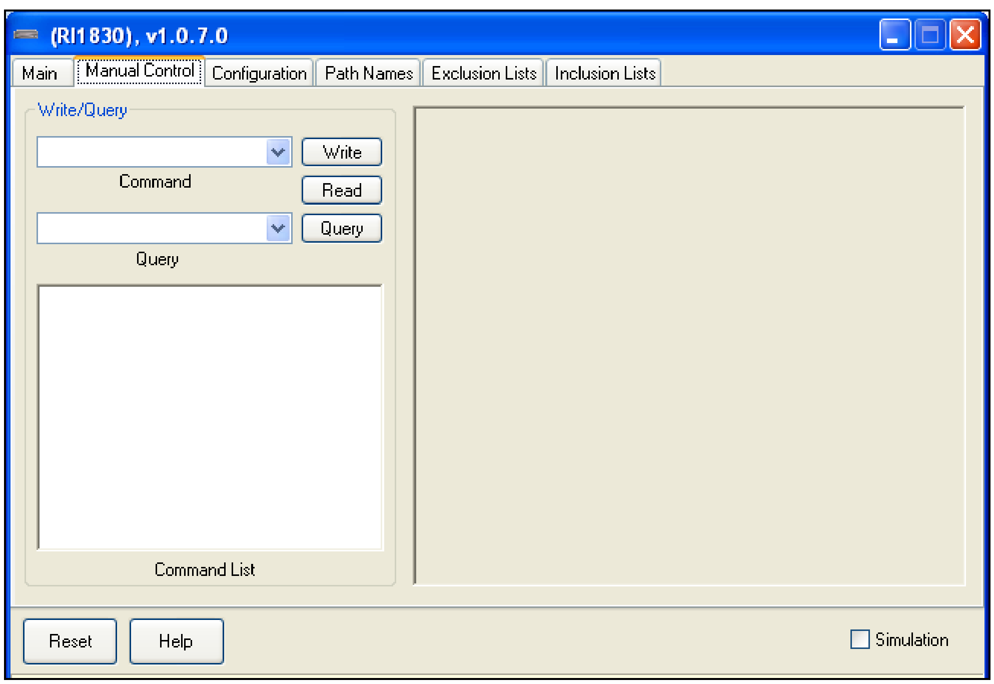


Figure 5-5: ActivATE Manual Control Tab

Another example of using the slot address within a command or query:

```
SYST:CDES? <slot address>
```

The device returns a reply to the SYST:CDES? query command with the description of the device located at the given slot address. This reply is unique for each 18x0 series module. The syntax for the reply is:

```
DEVICE DESCRIPTION
```

The <device description> for the 1170 is:

```
52-CHANNEL SPDT 2A SWITCH
```

Users can also specify an XML file (see **Configuration Tab**) that contains custom commands and queries that automatically get populated in the Command and Query drop-down list boxes.

Context Menu

Click the right-mouse button over the output window to bring up the following context menu.

Configuration Tab

The **Configuration** tab (Figure 5-6) is used to configure file settings in the 1830 system.

Selections for this tab include:

- **Address:** The address text box holds driver's resource string which tells the driver how to connect to the hardware. The string is based on the type of connection that the hardware has to the software, whether its, LAN, GPIB, USB, etc. The image shows the end of a LAN connection string. Click **Open** to connect to the hardware. Click **Close** to close the connection to the hardware.
- **Command File:** The command file is an XML file that stores any user-defined commands and populates the drop-down list boxes on the Manual Control tab with these commands. An example file format is:

```
<Commands>
    <command syntax="OPEN(@3(0:51))" description="Open
All Relays (3 = Slot Addr, 51 = Max Relay Number" />
    <query syntax="CLOSE?(@3(0:51)" description="Ask
the device for the current state of all relays (3 =
Slot Addr, 51 = Max Relay Number)" />
</Commands>
```

  Select the file browse button  to browse for the XML command file that will be used.

- **Time Out Value (ms)**: Specify the amount of time to wait before a "timeout" error is sent back to the user. The driver will try to read or query the device and wait the amount of time specified by the Time out value (in milliseconds), if no response is received in that time, then a timeout is assumed. Set the value by clicking the **Apply** button.

- **Termination Character:** Applying a termination character listed in the drop down menu will cause the selected character to be set as the character to stop at during a read query. When this character is encountered, the data before it is sent back to the user. The options for this device are "None", "LF" (Line Feed) and "CR" (Carriage Return). Click the **Apply** button to set a new termination character.
- **Initialize Device on Start Up:** When the driver is first created by the ActivATE application, the operator has the option of initializing the hardware. Checking this box will "talk" to the hardware and initialize the device to a power-on state. If this box is unchecked, then no hardware communication is performed on startup. This option will persist after the application is shutdown.
- **Reset on Start Up:** This control is disabled if **Initialize Device on Start Up** is not checked. The driver cannot attempt to do a reset if the hardware is not initialized. Check this box to reset the device to its power on state when the driver is first opened.

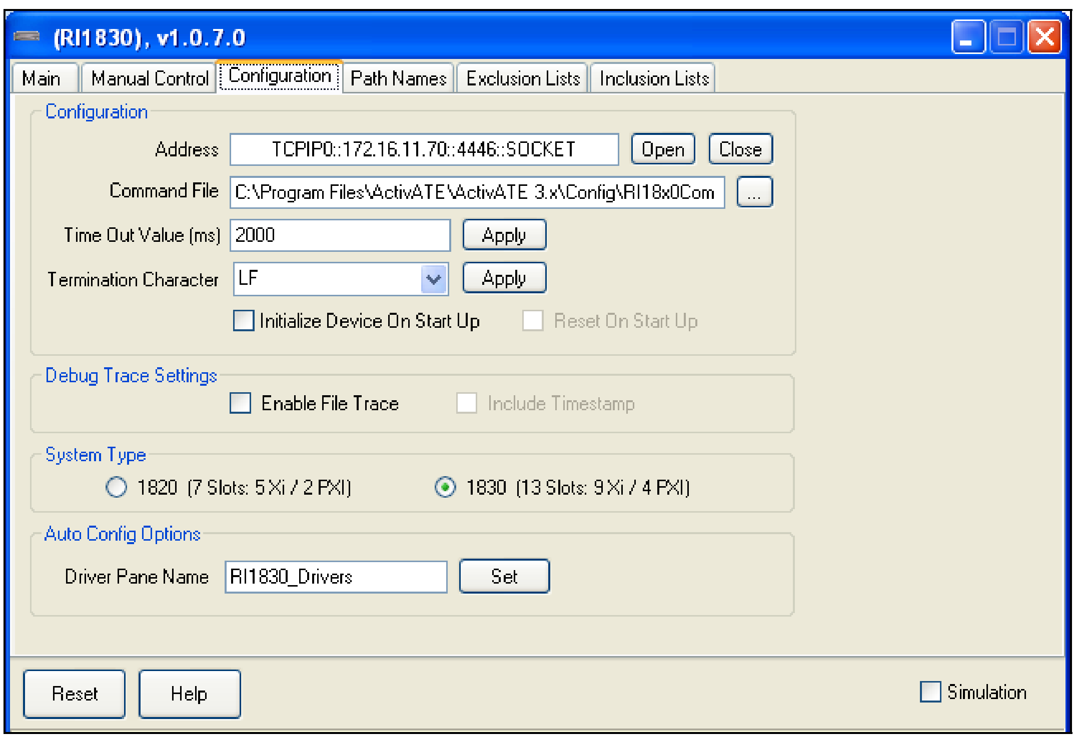


Figure 5-6: ActivATE Configuration Tab

Debug Trace Settings

- **Enable File Trace:** This option stores all commands sent to the hardware (through the driver) to a file generally located in the ActivATE installation directory.

- **Includes Time Stamp:** Check this option to include a timestamp with each entry in the file trace.

System Type

This should always be set to **1830**.

Auto Config Options

- **Driver Pane Name**: The name specified in this field will be the name of the pane that the drivers are be stored under. The pane (toolbar) is located, by default, on the left side of the ActivATE main screen.

Path Names Tab

The **Path Names** tab (Figure 5-7) allows the user to create defined channel paths and save them to the device or the device's non-volatile memory.

The 1830 provides the capability to define a group of channels and assign the group a name. When a group of channels is named, it is called a “path”. A path may consist not only of elements that must be closed (the closed channel list) but also of elements that must be opened (the open channel list) to complete the path.

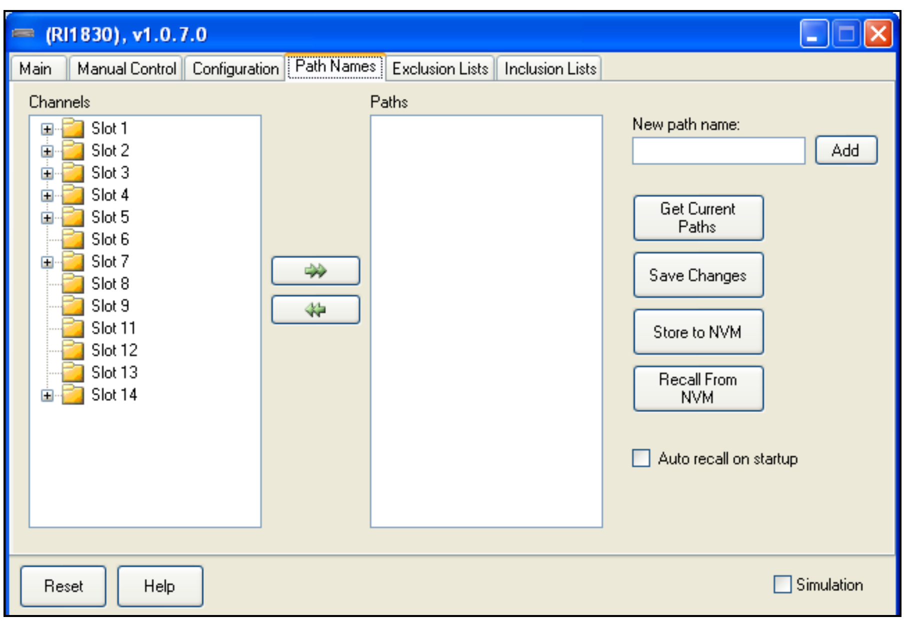


Figure 5-7: ActivATE Path Names Tab

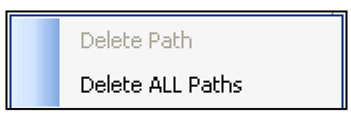
Selections for this tab include:

- **Channels**: This list holds the available channels for each slot on the device. A plus sign to the left of the slot number indicates that there are channels available for that slot. Click the plus sign to expand the channel list and view and/or select the desired channels.

- **Paths**: When selecting channels from the **Channels** list, use the right arrow to add the channel to the selected path in the **Paths** list. Use the left arrow to remove the selected channel from the path in the list. The **Paths** list holds the defined paths that have been created.
- **New Path name**: To create a new path, type a path name in the textbox and click the **Add** button. The new path will be available for edit in the **Paths** list.
- **Get Current Paths:** Clicking this button will get the current defined paths that have been saved and display them in the Path list.
- **Save Changes:** Click the "**Save Changes**" button to save changes that have recently been made to the path list.
- **Store to NVM:** The hardware has non-volatile memory (NVM) that can store information. This button will allow the user to store the paths defined to the hardware's non-volatile memory.
- **Recall from NVM**: This will retrieve the paths stored in the NVM and display the information to the user in the path list.
- **Auto recall on startup**: Checking this checkbox will cause the driver to retrieve path information from the NVM when ActivATE initializes the device at startup. This selection persists when ActivATE is closed and restarted.

Context Menu

Click the right-mouse button over the **Paths** window to bring up the context menu to either delete a selected path or delete all paths in the list



Exclusion Lists Tab

The **Exclusion Lists** tab (Figure 5-8) allows the user to create channel exclusion lists and save them to the device or the device's non-volatile memory.

The 1830 provides the capability to define sets of relays which are “mutually exclusive.” This feature is called an “exclude list.” When one relay on an exclude list is closed, all others in the exclude list are opened. This prevents two relays in an exclude list from being closed simultaneously. This feature can be used to ensure two or more relays are not shorting system resources, such as power supplies, together.

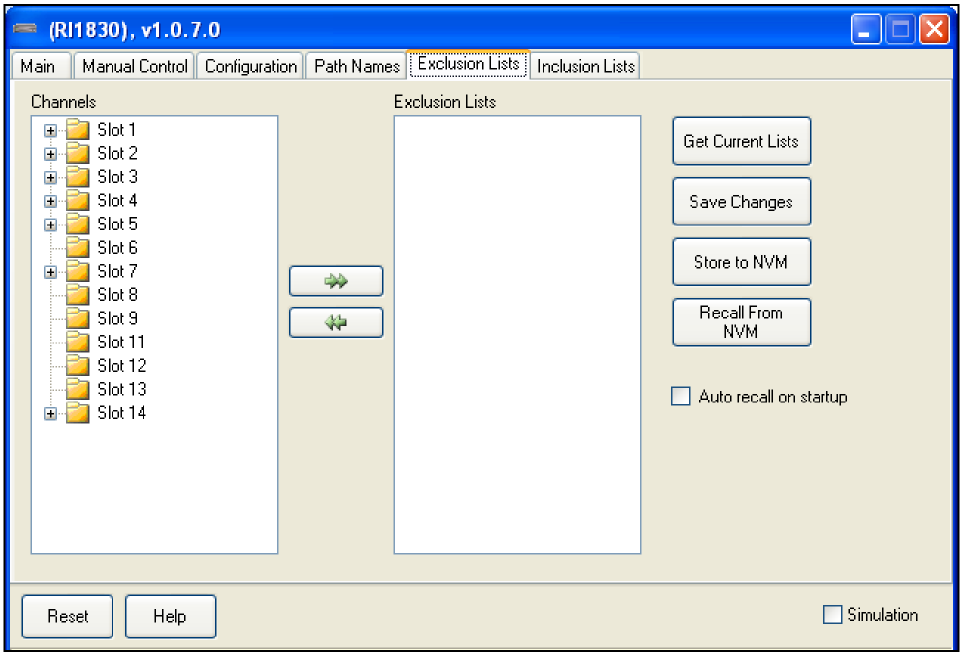


Figure 5-8: ActivATE Exclusion Lists Tab

A channel may reside on at most one exclude list. An attempt to place a channel on a second exclude list results in an error. The error is placed on the error queue and may be read using the SYSTEM:ERROR? query in the Manual Control tab. A channel cannot be on an include list with another channel if it is also on an exclude list with that second channel, this will also cause an error.

Selections for this tab include:

- **Channels**: This list holds the available channels for each slot on the device. A plus sign to the left of the slot number indicates that there are channels available for that slot. Click the plus sign to expand the channel list and view and/or select the desired channels.
- **Exclusion Lists:** Right click in this region and select "New List" to create a new list. Use the right arrow to add channels to this list from the Channels list. Use the left arrow to remove channels from this list.
- **Save Changes:** Click the "Save Changes" button to save changes that have recently been made to the Exclusion list.
- **Store to NVM:** The hardware has non-volatile memory (NVM) that can store information. This button will allow the user to store the exclusion list defined to the hardware's non-volatile memory.
- **Recall from NVM:** This will retrieve the list stored in the NVM and display the information to the user in the Exclusion List.
- **Auto recall on startup:** Checking this checkbox will cause the driver to retrieve list information from the NVM when ActivATE initializes the device at startup. This selection persists when ActivATE is closed and restarted.

Context Menu

Click the right-mouse button over the **Exclusion Lists** window to bring up the context menu. Click **New List** to create a new exclusion list. Remove the selected list by clicking **Remove List** and remove all lists by clicking on **Remove All Lists**.

Inclusion Lists Tab

The Inclusion Lists tab (Figure 5-9) allows the user to create channel inclusion lists and save them to the device or the device's non-volatile memory.

The 1830 provides the capability to define sets of relays which operate together. This feature is called an "include list." When a relay on an include list is closed, all relays on that include list are closed. Likewise, when a relay on an include list is opened, all relays on that include list are opened.

A channel may reside on at most one include list. An attempt to place a relay on a second include list results in an error. The error is placed on the error queue and may be read using the SYSTEM:ERROR? query in the Manual Control tab.

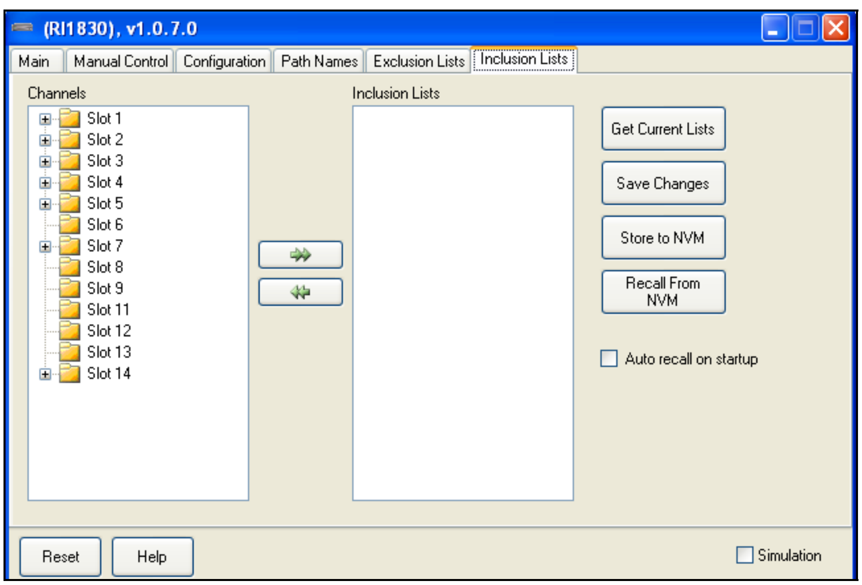


Figure 5-9: ActivATE Inclusion Lists Tab

Selections for this tab include:

- **Channels**: This list holds the available channels for each slot on the device. A plus sign to the left of the slot number indicates that there are channels available for that slot. Click the plus sign to expand the channel list and view and/or select the desired channels.
- **Inclusion Lists**: Right click in this region and select **New List** to create a new list. Use the right arrow to add channels to this list from the Channels list. Use the left arrow to remove channels from this list.
- **Save Changes:** Click the **Save Changes** button to save changes that have recently been made to the Inclusion list.
- **Store to NVM:** The hardware has non-volatile memory (NVM) that can store information. This button allows the user to store the inclusion list defined to the hardware's non-volatile memory.
- **Recall from NVM:** This retrieves the list stored in the NVM and display the information to the user in the Inclusion List.
- **Auto recall on Startup:** Checking this checkbox causes the driver to retrieve list information from the NVM when ActivATE initializes the device at startup. This selection persists when ActivATE is closed and restarted.

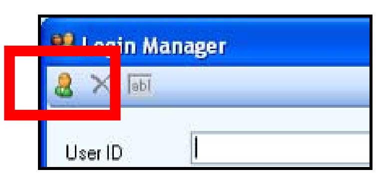
Context Menu

Click the right-mouse button over the **Inclusion Lists** window to bring up the context menu. Click **New List** to create a new inclusion list. Remove the selected list by clicking **Remove List** and remove all lists by clicking on **Remove All Lists**.

Adding Users

1. Select **Tools** > **User Manager**.

   The Login Manager screen appears (Figure 5-10).
2. Click the **Add New User** icon.



The fields on the left side of the Login Manager form become enabled for data entry.

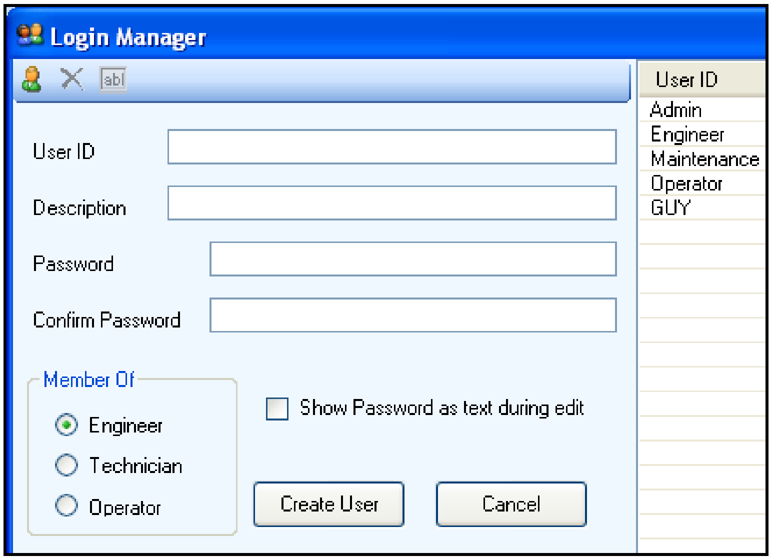


Figure 5-10: ActivATE Login Manager

3. Enter the user ID.
4. Enter a description (optional).
5. Enter a password.
6. Re-enter the same password in the **Confirm Password** field.
7. Select the group the user is part of (**Engineer**, **Technician**, or **Operator**).

   See the descriptions and limitations for each of the groups in the user list.
8. Click **Create User**.

   If successful, the new user is added to the user list on the right side of the Login Manager.
9. Exit **Login Manager**.

Changing the Password

1. Select **Tools** > **Change Password**.
2. Enter your old Password. (See Figure 5-11.)

Figure 5-11: ActivATE Admin: Change Password Screen

3. Enter your new password.
4. Enter your new password a second time to verify it.
5. Click **OK**.
6. Exit **Change Password** box.

Structuring Your ActivATE Test Program

To help guide you in organizing your first test programs in ActivATE, we have included a file, **1830 Example.seq,** that loads into the following folder:

\Program Files\ActivATE\ActivATE 3.x\Test Programs\Sample Tests

This file shows an example of how you might structure an 1830 test program and you can use it as a template for creating your own 1830 test. In addition, the example offers a number of helpful tips and some best practices for test command and comment structure. (See Figure 5-12.)

The program structure is set up to show:

- Setup
- Test
- Shutdown

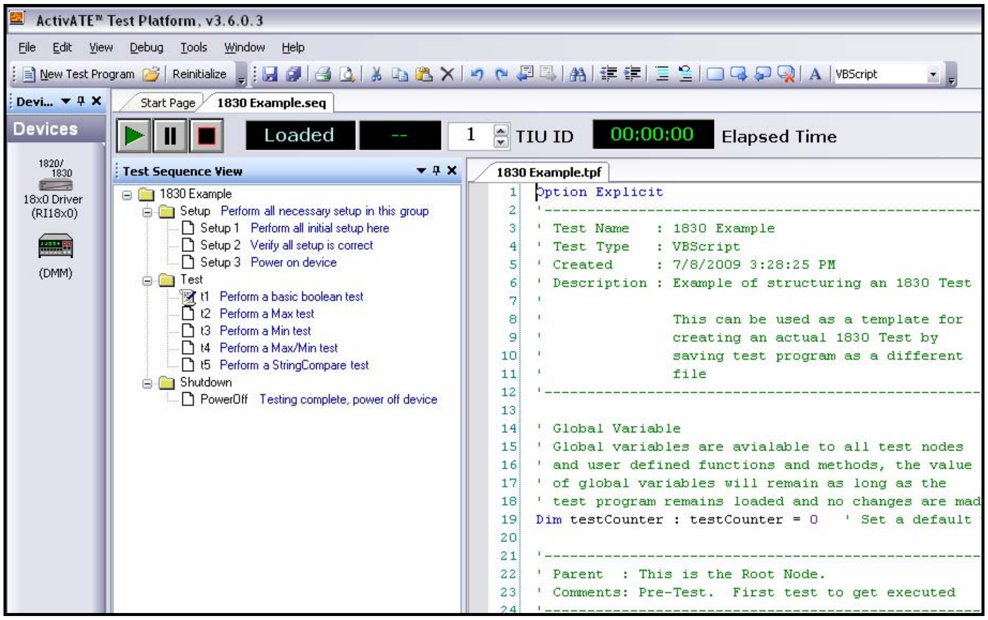


Figure 5-12: ActivATE Sample Program

Additional Help for the ActivATE Software

An extensive online Help for ActivATE software is available by selecting **Help** > **ActivATE Help** on the main toolbar (Figure 5-13).

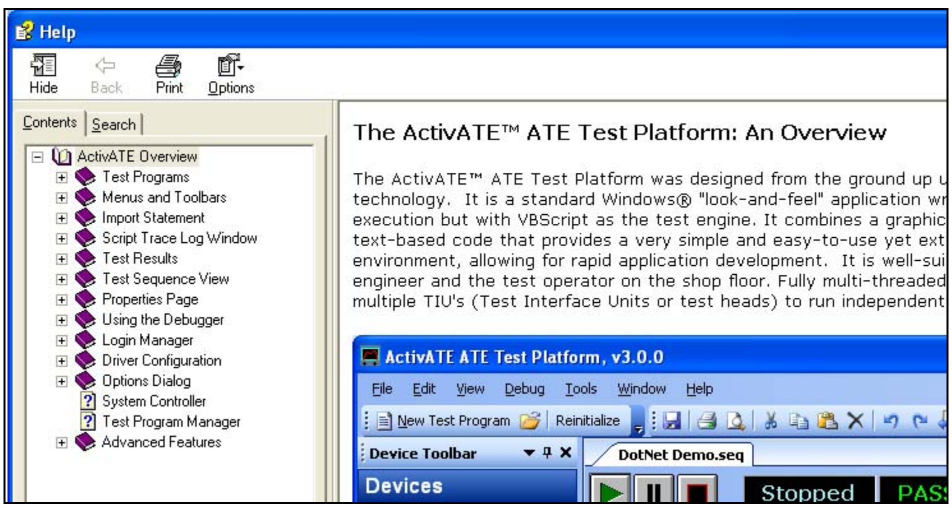


Figure 5-13: ActivATE Help Menu

You can also contact ActivATE support by sending an email to ActivATE@eads-nadefense.com or by visiting our support website at http://www2.eads-nadefense.com/ActivATE (Figure 5-14).

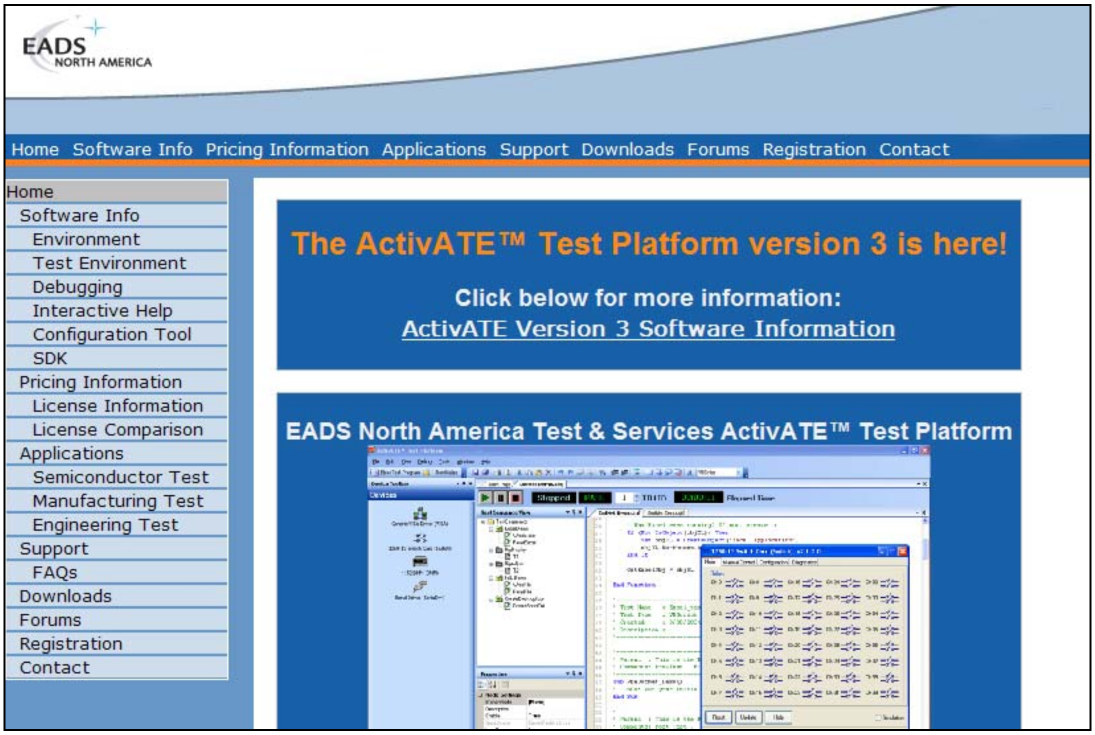


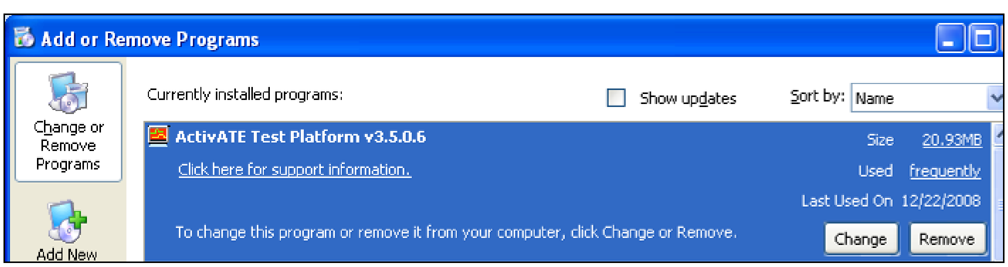
Figure 5-14: ActivATE Support Website

If you call our main customer support phone number at 1 800 722-3262, ask for the ActivATE software engineering team to ensure you will be quickly directed to the proper group.

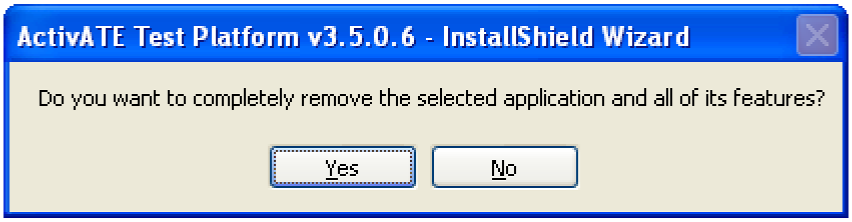
Uninstalling the ActivATE Software

To uninstall the ActivATE software

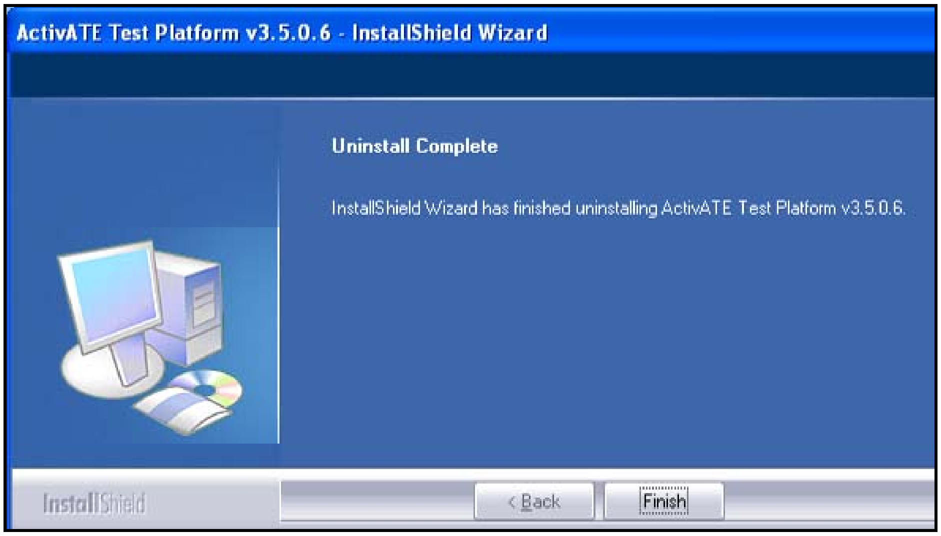
1. On your Windows desktop, select **Start** > **Control Panel** > **Add or Remove Programs**.
2. Select the **ActivATE Test Platform**.
3. Click **Remove**.



4. Click **Yes** when asked if you want to completely remove the application.



5. Click **Finish**.



This page was left intentionally blank.

Chapter 6
Using the 1830

This chapter features additional information for using your 1830 system including sections on:

- Triggering routing
- Controlling switch channels
- Path operation
- Configuring and making measurements with the DMM
- Sequences
- Scan operation
- Scan memory
- Scaling readings and assigning alarm limits
- Checking for installed plug-ins
- System commands
- Non-volatile storage
- Alarm monitor operation
- Emergency reset

Detailed SCPI commands and the command structure to perform various tasks are provided in **Chapter 7, SCPI Command Basics**. These commands are used in a variety of interfaces including those using IVI drivers, the 1830 custom web-pages, and ActivATE software controls.

Other chapters in this manual have additional sections of specific user instructions. Refer to the table of contents and/or index to locate the additional functions and features you are interested in.

Triggering Routing

The figure below illustrates the 1830 trigger router capabilities available through SCPI or web-page controls.

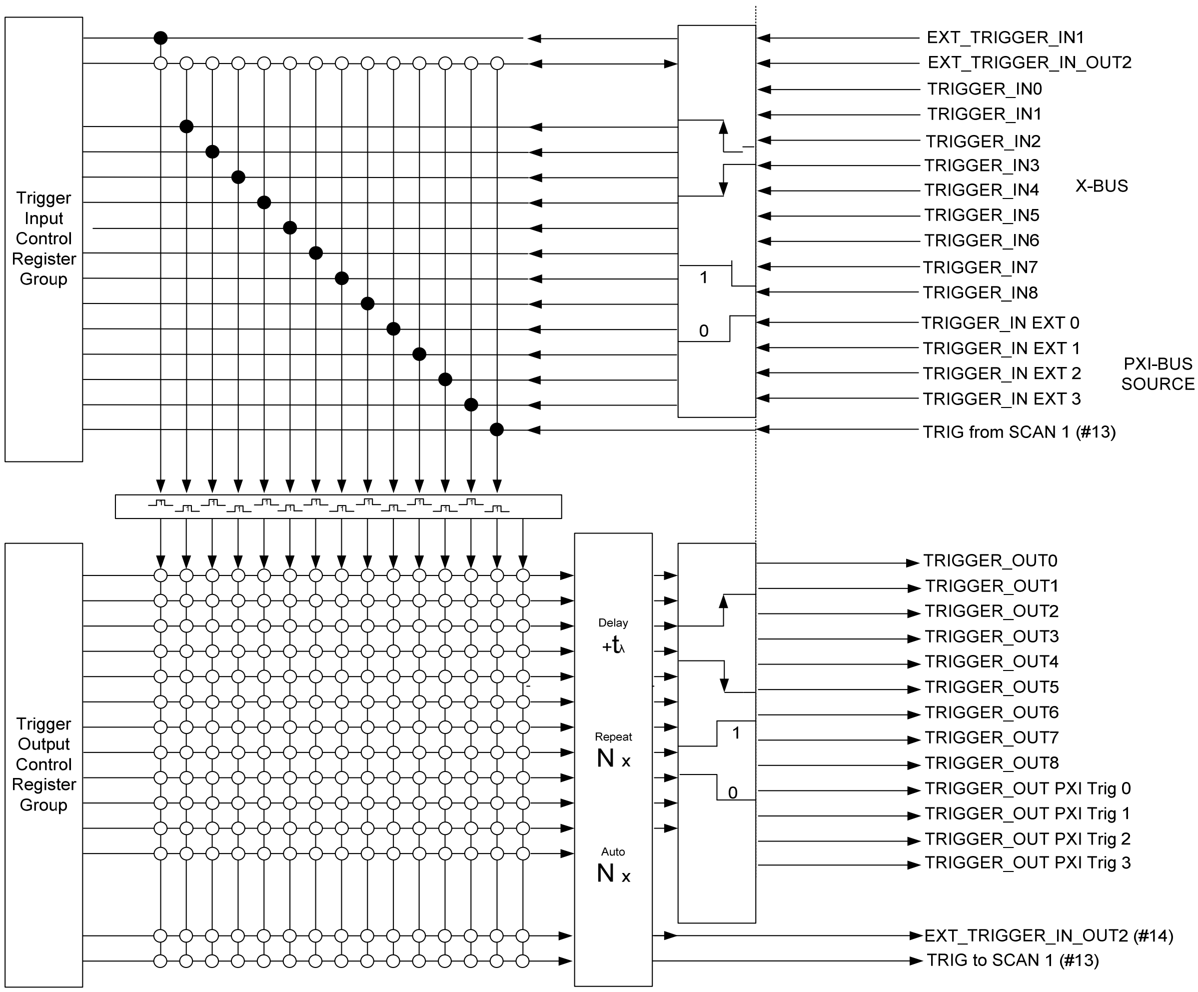


Figure 6-1: Triggering Matrix

The trigger hardware provides the following capabilities (see Figure 6-1):

1. Triggers from any of the following trigger inputs can be enabled (alone or in combination with each other)
    - TRIGGER IN from card #1 (TRIGGER_IN0)
    - TRIGGER IN from card #2 (TRIGGER_IN1)
    - TRIGGER IN from card #3 (TRIGGER_IN2)
    - TRIGGER IN from card #4 (TRIGGER_IN3)
    - TRIGGER IN from card #5 (TRIGGER_IN4)
    - TRIGGER IN from card #6 (TRIGGER_IN5)

- TRIGGER IN from card #7 (TRIGGER_IN6)
- TRIGGER IN from card #8 (TRIGGER_IN7)
- TRIGGER IN from card #9 (TRIGGER_IN8)
- TRIGGER IN EXT0 (PXI bus 0)
- TRIGGER IN EXT1 (PXI bus 1)
- TRIGGER IN EXT2 (PXI bus 2)
- TRIGGER IN EXT3 (PXI bus 3)
- SCAN TRIGGER
- EXT TRIGGER IN1
- EXT TRIGGER IN2 (when set as an input trigger)

2. The trigger sensitivity for any of these inputs can be independently programmed for:
    - Positive-edge sensitive
    - Negative-edge sensitive
    - High state sensitive
    - Low State sensitive
3. A trigger output pulse can be generated on any combination of the possible output triggers. These include:
    - TRIGGER OUT to card #1 (TRIGGER_OUT0)
    - TRIGGER OUT to card #2 (TRIGGER_OUT1)
    - TRIGGER OUT to card #3 (TRIGGER_OUT2)
    - TRIGGER OUT to card #4 (TRIGGER_OUT3)
    - TRIGGER OUT to card #5 (TRIGGER_OUT4)
    - TRIGGER OUT to card #6 (TRIGGER_OUT5)
    - TRIGGER OUT to card #7 (TRIGGER_OUT6)
    - TRIGGER OUT to card #8 (TRIGGER_OUT7)
    - TRIGGER OUT to PXI bus 0
    - TRIGGER OUT to PXI bus 1
    - TRIGGER OUT to PXI bus 2
    - TRIGGER OUT to PXI bus 3
    - TRIGGER OUT TO SCAN
4. A single trigger input may be selected for a trigger output.
5. Output triggers can be enabled independently. Zero, one, or multiple output triggers may be enabled at any one time.
6. The edge or level of each trigger output signal can be programmed. Each output trigger can be programmed for a different sensitivity. The trigger output can be programmed to be one of the following:

- Positive-edge sensitive (pulse out)
- Negative-edge sensitive (pulse out)
- High state sensitive (state out)
- Low State sensitive (state out)

7. A trigger delay may be added from the detection of an input trigger and the assertion of the output trigger. This delay is programmable independently for each output trigger.

Consult the “TRIGger” command description in **Chapter 7, SCPI Command Basics** for a description of programming the trigger selection and trigger attributes.

Triggering Example

The following example illustrates setting the triggering states of a card.

1. Select plug-in card in slot 3 as the source of the SCAN engine trigger

   ```
   TRIG:SCAN:SOUR INPUT3
   ```

2. Enable the plug-in 3 trigger output

   ```
   TRIG:MOD3:ENABLE ON
   ```

3. Enable the EXT2 trigger as an output trigger

   ```
   TRIG:EXT2:ENABLE ON
   ```

4. Set the output trigger source as the scan engine. This causes the EXT2 trigger to generate a pulse after each channel is closed in a scan list

   ```
   TRIG:EXT2:SOUR SCAN
   ```

5. Set the output trigger to generate a pulse (edge sensitivity)

   ```
   TRIG:EXT2:SENS EDGE
   ```

6. Set the output trigger to generate a low-going pulse (negative slope)

   ```
   TRIG:EXT2:SLOPE NEG
   ```

7. Set the output trigger pulse to 100 microsecond pulse width

   ```
   TRIG:EXT2:WIDTH 100.0E-6
   ```

Controlling Switch Channels

Switch channels may be opened and closed with the OPEN and CLOSE commands, respectively. These commands are followed by a *channel list* which selects one or more switch channels to open or close.

Each channel can be specified by using the form:

```
(@ <slot> ( <channel> ) )
```

For example, the command

```
CLOSE (@6(117))
```

closes channel 117 on the card plugged into slot 6.

The <slot> may be any value from 0 to 9. The <slot> value 0 identifies the 1830 mainframe. The 1830 mainframe provides control of the analog bus relays identified by channels 1 to 4 and the single-ended analog bus relays 11 to 18.

Note that each plug-in card type has a unique set of valid channel numbers. Consult the user manual for the plug-in cards to determine the valid channel numbers for that card.

Multiple channels on the same card may be operated with a single command. The format for this type of channel list is

```
(@ <slot> ( <channel> , <channel>, <channel> ) )
```

For example, the command

```
CLOSE (@6(100,204,314))
```

closes channels 100, 204 and 314 on the card plugged into slot 6.

A range of channels on a single card may be operated by using the format

```
(@ <slot> ( <channel> : <channel> ) )
```

For example, the command

```
CLOSE (@6 (100 : 204 ) )
```

closes all channels from 100 through 204 on the card plugged into slot 6.

The *channel list* can also select relays from different plug in cards (modules). The format for this type of *channel list* is

(@<slot> (<channels>), <slot> (<channels>))

For example

```
CLOSE (@6(0,13,100:204),3(17,24,41,33:30),1(5))
```

closes the following channels

Slot 6:	0, 13, 100 through 204
Slot 3:	17, 24, 41, 30 through 33
Slot 1:	5

The 1830 is shipped with IVI drivers including one driver for each type of switch card. The IVI driver operation is described in the user manual for those cards. The IVI driver can be used in any programming language that supports IVI-COM drivers, such as C#, C++, VB.net, Visual BASIC, and National Instruments LabVIEW.

In addition to the IVI-COM, an IVI-C version of the driver is supplied for each driver. This driver may be used with languages, such as National Instruments LabWindows/CVI, that support the IVI-C interface.

The switch cards may also be used within the ActivATE environment. An ActivATE-compatible driver for each type of switch card is supplied with the 1830 and is also available on the EADS North America Test and Services website.

Consult the user manual for each switch card for a more detailed description on the use of the IVI and ActivATE drivers.

Path Operation

A *path* is a collection of channels that are assigned a name. The path identifies a set of relay channels that are operated together and can be used in most commands that accept a *channel list* parameter.

A path may be defined by using the "`PATH:DEFine`" command. The simplest form of this command takes a single channel list parameter:

```
PATH:DEF Example_Path,(@1(10,23,9901),@2(14))
```

In the example above, the path name "Example_Path" is assigned to the set of channels 10, 23, and 9901 for the card in slot 1 and channel 14 for the card in slot 2. If the command

```
CLOSE (@Example_Path)
```

is now executed, all of the channels in the path close. Similarly, the command

```
OPEN (@Example_Path)
```

opens all of the channels in the path.

The "PATH:DEFine" command allows a second channel list to be specified. The second channel list identifies the set of channels to open when the path is closed. For example

```
PATH:DEF Two_lists,(@1(10,9901),2(14)), (@1(20:77))
CLOSE (@Two_lists)
OPEN (@Two_lists)
```

The "`CLOSE`" command in the example above would close channels 10 and 9901 on the card in slot 1 and channel 14 for the card in slot 2. It would also **open** channels 20 through 77 for the card in slot 1.

The "`OPEN`" command, on the other hand, would open channels 10 and 9901 on the card in slot 1 and channel 14 on the card in slot 2. It would **not** change the state of any of the channels 20 through 77 on the card in slot 1.

Path names can be used in defining other paths. For example:

```
PATH:DEF New_path,(@Example_path,7(15))
```

Defines a path named "New_path" that includes the channels from the path named "Example_path" and adds channel 15 of the card plugged into slot 7. Note that path names are NOT case-sensitive, so "Example_Path", "example_path", and "EXAMPLE_PATH" all identify the same path name.

When a path is defined using another path name, the path definition at the instant the "PATH:DEFINE" command is issued is used for defining the new path. That is, executing the following sequence:

```
PATH:DEF Path1,(@5(10,20))
PATH:DEF Path2,(@Path1,4(17))
```

```
PATH:DEL Path1
PATH:DEF Path1,(@5(33,35,37))
CLOSE (@PATH2)
```

results in channels 10 and 20 from slot 5 and channel 17 from slot 4 being closed. Channels 33, 35, and 37 for slot 5 are not affected.

Path names can be placed on a scan list to allow the entire path to be operated as a single entity when the scan feature is used.

For more information about the SCAN command, consult **Chapter 7, SCPI Command Basics**.

Naming a Path

A group of channels may also be assigned a name. When a group of channels is named, it is called a “path.” A path may consist not only of elements that must be closed (the closed channel list) but also of elements that must be opened (the open channel list) to complete the path. This may at first seem contradictory to close a path with switching elements that must be opened, but it nevertheless occurs.

A good example which illustrates the concept of an open channel list is a typical 1P2T failsafe microwave switch. Because this type of switch is failsafe, it only requires a single control line. By default, with no power applied, the first throw is always connected to the common. Energizing the control line opens the first throw and closes the connection between the second throw and the common. Both are obviously mutually exclusive.

Defining a path through the second throw of the example switch can be done strictly with a close channel list. The problem occurs when a path must go through the failsafe throw (first throw). To make this path the switch must be in an open or off state. The open channel list readily accomplishes this task.

Defining Path Names

The PATH:DEFINE command may be used to associate a name with one or more relays. The format for this command is:

```
[:ROUTe]
     :PATH
          :DEFine <path name> ,
               <close channel list> ,
               [,<open channel list>]
```

The <path name> follows the same name requirements as a <module name>, as described in the previous paragraphs.

The <close channel list> and the optional <open channel list> follows the syntax rules described in the **Specifying Relays in Commands** section of this chapter.

The following examples illustrate the use of a path name.

This associates the name path1 with the close channel list encompassing relays 6 through 9 on plug-in module 8 and relay 77 on plug-in module 4:

```
PATH:DEF path1,(@8(6:9),4(77))
```

This associates the name dmm_to_P177 with the close channel list, channels 205 and 305, on the module whose name is defined as “matrix”:

```
PATH:DEF dmm_to_P177,(@matrix(305,205))
```

This associates the name oscope1 with a close channel list for channels 0 and 3 on plug-in module 3 and an open channel list for plug-in module 5, channel 15.

```
PATH:DEF oscope,(@3(0,3)),(@5(15))
```

Once a path name is defined, it may be used in a CLOSE or OPEN command:

```
OPEN (@path1)
CLOSE (@dmm_to_P177)
CLOSE (@path1,dmm_to_P177,7(0:10))
```

A CLOSE command, for a given pathname, effectively closes all channels listed on the close channel list and open those listed on the open channel list.

An OPEN command, for a given pathname, opens all channels listed on the close channel list. Channels on the open channel list are ignored when using this command.

Path names may also be used in defining an “Include List”, an “Exclude List”, or a “Scan List”. The following sections describe each of these concepts.

When using path names with the “Include List”, “Exclude List” and “Scan List”, the presently defined path is used. That is, if the following sequence of commands is received:

```
PATH:DEF PATH1,(@5(0),7(0))
INCLUDE (@PATH1,1(0))
PATH:DEF PATH1,(@6(17),8(23))
CLOSE (@1(0))
```

then the path definition in effect when the INCLUDE command was defined is used. For the example shown then, channel 0 on module 5 and channel 0 on module 7 are affected; channel 17 on module 6 and channel 23 on module 8 are NOT affected.

Removing Path Names

The PATH:DELETE command may be used to delete a single path name from the list of known path names. The PATH:DELETE:ALL command may be used to remove all presently defined path names.

The syntax for these commands is:

```
[:ROUTe]
     :PATH
```

```
:DELete <path name>
             :ALL
```

The following examples illustrate the use of this command.

Removes the name “path1”:

```
ROUTE:PATH path1
```

Removes the name “testit”:

```
PATH:DEL testit
```

Removes all path names:

```
PATH:DEL:ALL
```

Reading the Presently Defined Path Names

The PATH:CATALOG? command may be used to read back all of the presently defined path names. The format for this command is:

```
[:ROUTe]
     :PATH
          :CATalog?
```

The reply to this command consists of the presently defined path names, each of which is separated by a comma. Path names are stored internally in upper-case characters by the 1256. These upper-case names are returned in the reply.

For example, suppose the following commands have been executed:

```
PATH:DEFINE dmm_2_pin1,(@1(117),2(17))
PATH:DEFINE dmm_2_pin2,(@1(116),2(14),7(23))
PATH:DEFINE cntr_2_pin1,(@1(217),2(24))
PATH:DEFINE cntr_2_pin2,(@1(216),2(37),7(3))
```

Then the query:

```
PATH:CAT?
```

returns the reply:

```
DMM_2_PIN1,DMM_2_PIN2,CNTR_2_PIN1,
CNTR_2_PIN2
```

Reading the Channel List for a Path Name

The module address associated with a module name may be read using the PATH:DEFINE? query. The syntax for this command is:

```
[:ROUTe]
     :PATH
        :DEFine? <path name>
```

Using the example in the previous section of this manual, the query:

```
PATH:DEFINE? dmm_to_pin1
```

returns the reply:

```
(@1(117),2(17)
```

Storing Path Names in Non-Volatile Memory

All paths presently defined may be prepared for storage in non-volatile memory using the PATH:STORE command.

Recall all stored paths using the PATH:RECALL command.

Closing Relays

The CLOSE command may be used to close channels. To say a channel is closed, it means either:

- The input of the channel is connected to the output (Single-Pole Single Throw).
- The input of the channel is disconnected from the “normally closed” output and connected to the “normally open” output (Single-Pole Double-Throw).
- The common input of a multiplexer (mux) is connected to the output denoted by the channel number.

The syntax for the CLOSE command is:

```
[:ROUTe]
      :CLOSe <channel list>
```

The format for a “<channel list>” is described in the previous paragraphs of this section of the manual.

The CLOSE? command may be used to query the present state of the relays in the system. This command returns a reply of a sequence of “0" and “1", each of which are separated by a single ASCII space character. The value of the reply is “0" if the corresponding relay is opened, or “1" if the corresponding relay is closed.

The reply is one-for-one with the <channel-list>. For example, assume channels for a particular relay module at module address 7 are numbered as follows:

```
0, 1, 2, 3, 4, 10, 11, 12, 13, 14, 20, 21,
22, 23, 24, 30, 31, 32, 33, 34
```

This example module consists of 20 channels. Suppose that only the following channels are closed, while the remainder are open:

```
3, 20, 31
```

The following examples show the replies to the “CLOSE?” queries.

Command:

```
CLOSE? (@7(0:34))
```

Reply:

```
0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0
```

Command:

```
CLOSE? (@7(0))
```

Reply:

```
0
```

Command:

```
CLOSE? (@7(3, 20, 31))
```

Reply:

```
1 1 1
```

Opening Relays

The OPEN command may be used to open channels. To say a channel is open means:

1. The input of the channel is disconnected to the output (Single-Pole Single Throw)
2. The input of the channel is disconnected from the “normally open” output and connected to the “normally closed” output (Single-Pole Double-Throw).

The syntax for the “OPEN” command is:

```
[:ROUTe]
        :OPEN <channel list>
```

The format for a “<channel list>” is described in the previous paragraphs of this section of the manual.

The OPEN? command may be used to query the present state of the relays in the system. This command returns a reply of a sequence of “0" and “1", each of which are separated by a single ASCII space character. The value of the reply is “1" if the corresponding relay is opened, or “0" if the corresponding relay is closed. **Note that this is the opposite state from the** CLOSE? **query**.

Configuring and Making Measurements with the DMM

The 1830 supports one to four DMMs. These DMMs are installed at the factory and occupy one of the internal plug-in slots 11 through 14.

The SCPI “SENSE” command is used to configure the DMM to make a particular measurement type. The “SENSE” command may also be used to fine-tune the measurement range, resolution, aperture, and many other attributes of the measurement.

As an example, the command:

```
SENSE:RESISTANCE 10000,0.1,(@14(0))
```

configures the DMM plugged into slot 14 to make a 2-wire resistance measurement. At the front panel terminals, it places the DMM into the 24K ohm range (the lowest range that accommodates a 10000 ohm reading) and a resolution of 0.1 ohms.

Once the DMM has been configured, it may be read by using the "READ?" query. For example:

```
READ? (@14(0))
```

makes a reading from the DMM and sends a reply over the communication interface on which the query was received (Ethernet, USB, or GPIB).

The 1830 is shipped with IVI and ActivATE drivers that support the DMM. These allow you to configure and operate the DMM from programming languages that support IVI-COM and IVI-C, as well as the ActivATE development environment.

The DMM may also be monitored and controlled with the web-page interface.

Connecting the DMM to the Analog Bus

When the 1830 is powered on, or after it receives a "*RST" command, the DMM is connected to its front panel inputs. Several of the switch cards, including the 1380, 1450D, and 1450F, can route their front panel inputs to the analog bus. In order to measure the input signals from these plug-in switch cards, both the switch cards and the DMM must be connected to one or more of the differential analog bus lines in the 1830 mainframe.

The DMM may be connected to an analog bus with the `CLOSE` command. The following table defines the channels to use to connect the DMM to the differential analog bus.

	V+ / V–	I+ / I–	Guard +/–
Differential Analog Bus 1	11	21	31
Differential Analog Bus 2	12	22	32
Differential Analog Bus 3	13	23	33
Differential Analog Bus 4	14	24	34

So, to connect the V+/V- inputs of the DMM in slot 14 to Analog Bus 3, you would use the command:

```
CLOSE (@14(13))
```

The 1830 prevents the user from simultaneously closing a single input of the DMM to multiple analog buses. The 1830 also prevents the user from connecting multiple inputs to a single analog bus. In other words, only one channel from any row and column in the table above can be closed at one time. If you attempt to violate this protection, the `CLOSE` command fails and an error is be placed in the error queue.

The switch cards that provide access to the analog bus, including the 1380 and 1450D/F models, have specific switch channels that can be closed to connect the outputs as described in the following tables.

Channels	1380 Card
9001, 9901	Connects COM 0 to ABUS1
9002, 9902	Connects COM 2 to ABUS2
9003, 9903	Connects COM 4 to ABUS3
9004, 9904	Connects COM 6 to ABUS4

Channel	1450D/F Card
9901	Connects row 0 to ABUS1
9902	Connects row 1 to ABUS2
9903	Connects row 2 to ABUS3
9904	Connects row 3 to ABUS4

Note: The 1450D card also connects to the single-ended analog bus. See the 1450 user manual for additional details.

Channel	1450F Card
9905	Connects row 4 to SABUS 1 & 2
9906	Connects row 5 to SABUS 3 & 4
9907	Connects row 6 to SABUS 5 & 6
9908	Connects row 7 to SABUS 7 & 8

The command

```
CLOSE (@6(9903))
```

connects row 2 to ABUS3 for a 1450 card plugged into slot 6.

Connecting Switch Cards in Slots 1 to 5 to the DMM

The 1830 mainframe contains relays that isolate the analog bus lines between slots 1 through 5 and slots 6 through 9. This means that if you want to connect a signal from a 1380 or 1450 switch card to a DMM in slot 14 via the analog bus, you need to close the isolation relays for the analog bus(es) on the mainframe backplane.

The following channels are used to connect the Analog Buses in slots 1 through 5 to the Analog Buses in slots 6 through 9:

(@0(1))	Connects ABUS1
(@0(2))	Connects ABUS2
(@0(3))	Connects ABUS3
(@0(4))	Connects ABUS4

So, for our example, the command:

```
CLOSE (@0(3))
```

connects ABUS3 across all 9 slots.

DMM Measurement Examples

The following examples illustrate measurement setups using a 1380 switch card, the analog bus, and the DMM.

Single-ended Voltage Scan

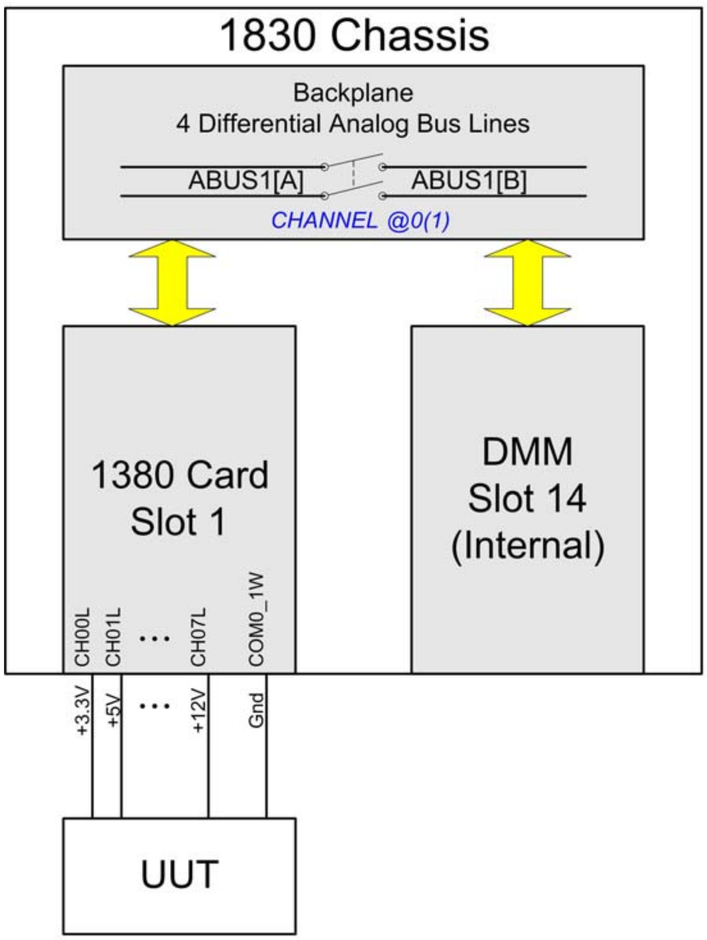


Figure 6-2: Single-ended Voltage Scan DMM Setup

1. Connect ground reference to COM0_1W.
2. Connect channels to CH00L-CH07L.
3. Short interlock pins.
4. Power on 1830 chassis.
5. Send command to connect Analog Bus 1 bank A and B together:

```
CLOSE(@0(1))
```

6. Send command to connect LXI1380 Mux 0 to Analog Bus 1 :
```
CLOSE(@1(9001,9901))
```
7. Send command to connect DMM V+/V- to Analog Bus 1:
```
CLOSE(@14(11))
```
8. Define sequencer to take DC voltage measurements:
```
SEQuence:DEFine Read_DMM_DCV
SEQuence:STEP "SENSe:VOLT:DC 24.0,0.001,(@14(0))"
SEQuence:STEP "READ? (@14(0))"
SEQuence:END
```
9. Associate DMM measurement with every channel:
```
SEQuence:SELect Read_DMM_DCV, (@1(00:07))
```
10. Set up scan list for LXI1380 channels 00 to 07:
```
SCAN (@1(00:07))
```
11. Run scan chain:
```
INITiate:IMMediate
```
12. Retrieve measurements from scan memory:
```
FETCh:PROGram? 1,8
```

For more information, refer to the following sections:

- Configuring and Making Measurements with the DMM
- Sequences
- Scan Chain
- Scan Memory

T-type Thermocouple Scan

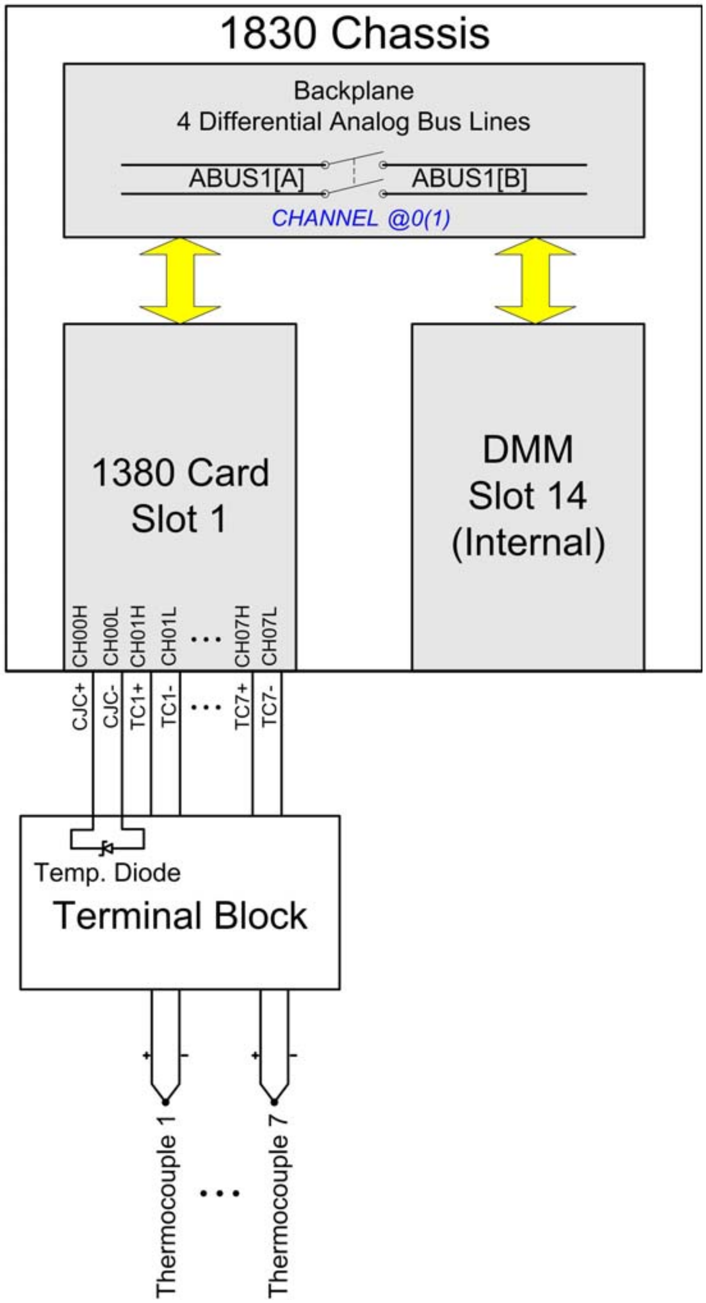


Figure 6-3: T-type Thermocouple Scan Setup

1. Power off the 1830 system.
2. Connect National Semiconductor LM335A temperature diode or equivalent to the 1380 card, CH00. Connect cathode to CH00H and anode to CH00L.
3. Connect thermocouple sensors to CH01-07.
4. Short safety interlock pins (J200.E4 to J200.E8).

5. Power on the 1830 system.
6. Send command to connect Analog Bus 1 bank A and B together:
```
CLOSE (@0(1))
```
7. Send command to connect the 1380 Mux 0 to Analog Bus 1:
```
CLOSE (@1(9001, 9901))
```
8. Send command to connect DMM V+/V- to Analog Bus 1:
```
CLOSE (@14(11))
```
9. Send command to connect CJC 1 mA current source across V+ and V- lines:
```
CLOSE (@14(41))
```
10. Query DMM for DC voltage measurement:
```
MEASure:VOLTage? (@14(0))
```
11. Send command to disconnect CJC 1mA current source:
```
OPEN (@14(41))
```
12. Convert voltage measurement to temperature in ºC using manufacturer's datasheet.
13. Send command to set cold junction compensation temperature (for example, to 20ºC).
```
SENSe:TEMPerature:TRANsducer:TCouple:RJUNCtion 20, (@14(0))
```
14. Define sequencer to take T-type thermocouple measurement:
```
SEQuence:DEFine Read_DMM_Thermocouple
SEQuence:STEP "SENSe:TEMPerature (@14(0))"
SEQuence:STEP "SENSe:TEMPerature:TRANsducer:TYPE
 TCouple, (@14(0))"
SEQuence:STEP "SENSe:TEMPerature:TRANsducer:
 TCouple:TYPE T, (@14(0))"
SEQuence:STEP "SENSe:TEMPerature:UNIT C, (@14(0))"
SEQuence:STEP "READ? (@14(0))"
SEQuence:STEP
```
15. Associate CJC measurement with temperature diode channel:
```
SEQuence:SELect Read_DMM_Thermocouple, (@1(01:07))
```
16. Set up scan chain for LXI1380 channels 01 to 07:
```
SCAN (@1(01:07))
```
17. Run scan chain:
```
INITiate:IMMediate
```
18. Retrieve measurements from scan memory:
```
FETCh:PROGram? 1,7
```

Sequences

A *sequence* is a named series of SCPI commands and directive statements. A sequence can be executed by itself or it can be executed as part of a scan list.

A sequence is defined by first sending the "`SEQuence:DEFine`" command. It is followed by sending one or more "`SEQuence:STEP`" commands. Finally, the definition is completed by sending the "`SEQuence:END`" command. A simple example is shown below:

```
SEQ:DEF Read_DMM_DCV
SEQ:STEP 'SENSE:VOLT:DC 2.0,0.001,(@14(0))'
SEQ:STEP 'READ? (@14(0))'
SEQ:END
```

This set of commands defines a sequence named "READ_DMM_DCV" that sets up the DMM to make a DC voltage measurement using the 2.4V range of the DMM with a resolution of 0.001 volts. It then makes a single reading from the DMM.

In addition to SCPI commands, sequences can include the following directive statements:

1. Call another named sequence (`SEQ:STEP 'CALL OTHER_SEQ'`).
2. Set the scale factor for measurements made during the execution of the sequence (`SEQ:STEP 'SET GAIN=1.125'`).
3. Set the offset to apply to measurements made during the execution of the sequence (`SEQ:STEP 'SET OFFSET = -0.004'`).
4. Set an upper or lower alarm limit for measurements (`SEQ:STEP 'SET ALARMHI = 1.2345'`).
5. Enable or disable the alarm limit detection for either the upper or lower alarm limit (`SEQ:STEP 'SET ALARMLO = OFF'`).
6. Define a units string to associate with the measurement results (`SEQ:STEP 'SET UNITS = mA'`).
7. Check if the upper or lower alarm limit has been exceeded by the most recent measurement and conditionally jump to another part of the sequence (`SEQ:STEP 'IF ALARMHI THEN GOTO $TURNOFF'`).
8. Check if the upper or lower alarm limit has been exceeded by the most recent measurement and conditionally call another sequence (`SEQ:STEP 'IF ALARMLO THEN CALL TURNOFF_POWER'`).
9. Wait a certain number of seconds with a resolution of 10 microseconds (`SEQ:STEP 'WAIT 1.23492'`).
10. Generate a trigger pulse (`SEQ:STEP 'TRIGGER'`).
11. Return from the sequence to the caller (`SEQ:STEP 'RETURN'`).

Sequences can be executed in a stand-alone mode by specifying the name of the sequence in a SEQuence:INITiate command:

```
SEQ:INIT EXAMPLE_SEQUENCE
```

The measurements made during the execution of a sequence are stored in the scan memory. The scan memory can be read by using the "`FETCh:PROG?`" query. The number of data points in the scan memory can be read with the "`DATA:POINts?`" query.

For more information on the definition and use of sequences, refer to the **SEQuence Command Summary** section in **Chapter 7, SCPI Command Basics**.

Scan Operation

The 1830 provides the capability to sequentially scan across relay channels and paths and to optionally make and record measurements and/or perform some other instrument actions after each channel or path is closed.

The first step in executing a scan is to define a list of channels, paths, and sequences. The format for defining a scan list is:

```
SCAN (@ <scan list> )
```

where the <scan list> consists of channels, path names, and sequence names. For example

```
SCAN (@1(0:7),Path1,XYZ,2(17:10))
```

This defines a scan list that would close, in sequence channels 0 through 7 of the plug-in card in slot 1, the path named "PATH1", the sequence named "XYZ", and the channels 17 through 10 of the plug-in card in slot 2.

For each relay channel or path name on the scan list, you can associate a sequence to execute once the channel or path is closed during the execution of the scan. You can use the "`SEQuence:SELect`" command to associate a sequence with one or more channels or path names. For example

```
SEQ:SEL READ_DMM_DCV,(@1(0:77),path1,path2,2(16,36))
```

associates the sequence named "READ_DMM" with every channel between 0 and 77 for the plug-in in slot 1, the paths named "PATH1" and "PATH2", and channels 16 and 36 of the plug-in in slot 2. When any of these channels or paths are closed during the execution of a scan list, the sequence named "READ_DMM_DCV" is called immediately after the channels or paths are closed. From the previous sequence example, this would configure the DMM into the DC voltage measurement mode, and then make a reading from the DMM. The reading gets stored into the scan memory buffer.

The execution of the scan proceeds as follows

- The scan list must be armed. Execute the command "`INITiate`" or "`INITiate:IMMediate`"
- Once armed, the scan list waits for a trigger to begin scanning. The trigger source for the scan list is selected by the "`TRIGger:SOURce`" command. This may be any one of the following:
  - `IMMediate` – the scan starts immediately

    - EXTernal – the scan waits for the input on the EXT IN input
    - BUS – the scan waits for a "*TRG" command on a command interface (Ethernet, GPIB, USB).
    - TIMer – the scan waits for the next tick from the timer. The timer period is set using the "TRIGger:TIMer" command.
    - USER – the scan waits for the trigger from the trigger input(s) enabled for the scan. The trigger input is selected with the "TRIGger:SCAN:SOURce" command.

- After the trigger has been received, the scan closes the first channel or path on the scan list (or execute the sequence if a sequence is the first element on the scan list).
- After the channel or path is closed, the scan list waits until the relay has settled. The relay settling time can be set with the "CHANnel:DELay" commands on a per-channel and per-path basis. The default channel delays are set by the 1830 at power-up to a value that ensures the relays have properly settled.
- After the channel or path has settled, any sequence associated with the channel or path is executed. Any measurements made by the sequence are stored in the scan memory.
- After the sequence (if any) has completed, the 1830 waits for a trigger from to proceed. The trigger for sequencing through the scan list is selected by the "SCAN:ADVance:SOURce" command. Valid sources are the same as for the "TRIGger:SOURce" command described above.
- After a trigger has been received on the scan advance source, the channel or path closed in the previous step of the scan list is opened.
- The 1830 then waits for the open operation to complete. The delay time for opening a relay can be set with the "CHANnel:DELay:OPEN" command. For some relay cards, the open delay time may be less than the close delay time because it does not have to wait for the relay contacts to quit bouncing.
- The next channel or path in the scan list is then executed.
- The 1830 waits for the channel delay time for the newly closed channel or path.
- If any sequence is associated with the new channel or path, the sequence is executed. Measurements made during the execution of the sequence are placed into scan memory.
- The 1830 waits for the next trigger from the "SCAN:ADVance:SOURce".
- If the "SWEep:COUnt" value is set to 1, the scan completes after the last channel or path has been opened.
- If the "SWEep:COUnt" value is greater than 1, the scan continues at the beginning of the scan list. Each time through the scan list constitutes a single sweep. When the number of sweeps specified have been completed, the scan completes.

- If the "TRIGger:COUnt" value is 1, the scan list terminates at this point.
- If the "TRIGger:COUnt" value is greater than one, the 1830 waits for another trigger from the specified "TRIGger:SOURce" and begins another cycle of sweeps through the scan list.
- Assuming one measurement is made per channel or path on the scan list, the total number of measurements made are equal to:

  (# channels/paths) x (sweep count) x (trigger count)

  This is illustrated by Figure 6-4.

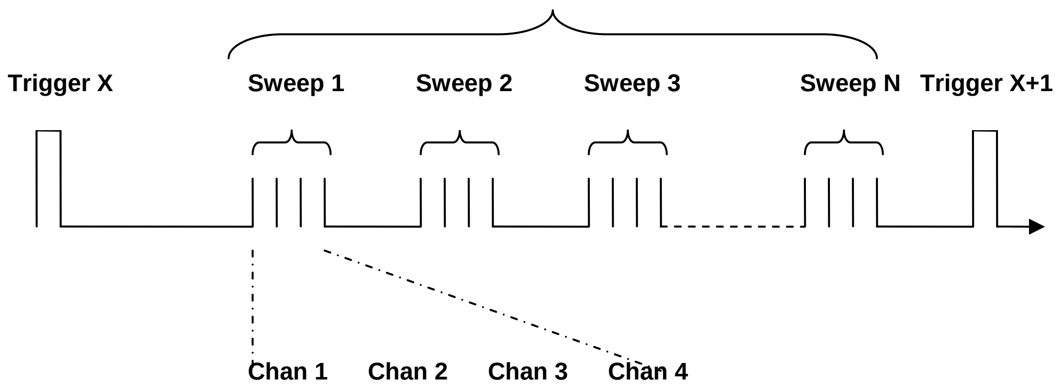


Figure 6-4: Scan Operation

Scan Memory

All measurements made during the execution of a scan list are stored into scan memory. The scan memory stores up to 500,000 measurements. The measurements can be read with the "FETCh:PROGram?" query, and the number of data points presently stored may be read with the "DATA:POINts?" query.

Each measurement record consists of the following

- The measured value, returned in exponential format
- A date/time stamp with a resolution of 10 microseconds
- A measurement channel that identifies which channel in the system made the measurement
- A scan channel that identifies the channel or path that was closed, or the sequence that was executed, as part of the scan list
- A measurement units string that can be assigned to the measurement

By default, only the measured result is returned by the "FETCh:PROGram?" query. You can enable the other fields of the measurement by using the "FORMat:READing" command to enable the other fields and to specify whether the absolute or relative time format is used.

The “`FETCh:PROGram?`” query returns the measurement records stored. However, since the output buffer for the instrument holds at most 10240 characters, you must limit the number of records you are requesting so as not to overfill the output buffer.

The best way to ensure there is no overflow in the output buffer is to limit the number of records requested with the optional command parameters. The format of the “`FETCh:PROGram?`” query is

```
FETCh:PROGram?  [ <start record> [ <# of records> ] ]
```

By limiting the <number of records> specified to 100 or less, you can be sure that the output buffer does not overfill. For example, if there are 500 measurement records, you can recall all 500 with the following sequence:

```
FETCH:PROG? 1,100
FETCH:PROG? 101,100
FETCH:PROG? 201,100
FETCH:PROG? 301,100
FETCH:PROG? 401,100
```

Each measurement record returned in response to the “`FETCh:PROGram?`” query is separated from the next by a semicolon:

```
<measurement-1> ; <measurement-2> ; …
```

Each measurement record is returned in the following order:

```
<meas val><units>,<timestamp>,<meas chan>,<scan chan>,<alarm>
```

Where:

`<meas val>`	is the measured value, returned in exponential format
`<units>`	is the units string associated with the measurement
`<timestamp>`	is the relative or absolute time/date stamp of the measurement
`<meas chan>`	is the channel that made the measurement
`<scan chan>`	is the channel or path that was closed during the scan list (or the sequence that was specified in the scan list) for the measurement
`<alarm>`	indicates whether the measurement was within alarm limits (0), below the lower alarm limit (1), or above the upper alarm limit (2)

As an example, with relative time format:

```
1.23456789E+00 VDC,0.00124,14(0),6(23),0
```

Scaling Readings and Assigning Alarm Limits

Each time a measurement is made during the scan list, the measurement may have a scale and offset applied to it. The measurement may also have a units of measurement associated with it. Finally, the measurement may be compared against upper and lower alarm limits.

There are two ways to apply a scale factor to a measurement. The first is to associate a scale factor to one or more channels or paths in the scan list. A scale factor can be applied with the "`CALCulate:SCALe:GAIN`" command:

```
CALC:SCALE:GAIN 2.0,(@1(1,3,5),PATHXYZ,2(77))
```

applies a scale factor of 2.0 to any measurement read by the sequence associated with these channels or paths when executing the scan list.

The other way to apply a scale factor to a reading is by use of the "`SET GAIN = xxx`" directive inside the sequence that is making the measurement. Note that the "`SET GAIN = xxx`" directive takes precedence over the "`CALC:SCALE:GAIN`" value associated with the channel. Therefore, if the following sequence of commands is executed.

```
SEQ:DEF EXAMPLE_SEQ
SEQ:STEP 'SET GAIN = 10.0"
SEQ:STEP 'READ? (@14(0))'
SEQ:END
SEQ:SEL EXAMPLE_SEQ,(@1(0:7))
CALC:SCALE:GAIN 2.0,(@1(0:7))
SCAN (@1(0:7))
INIT:IMM
```

Then each reading is multiplied by 10.0 because the "`SET GAIN = 10.0`" directive has higher precedence than the "`CALC:SCALE:GAIN`" command.

An offset can be added to each measurement taken during the execution of the scan list. Again, this may be done in either of two ways:

```
CALC:SCALE:OFFSET -0.025,(@PATH999,1(17),2(33))
```

`SET OFFSET = 0.0123` inside the sequence definition

As with the gain, the "`SET OFFSET = xxx`" inside a sequence takes precedence over the offset set with the "`CALC:SCALE:OFFSET`" command.

Once an offset or a gain is assigned with the "CALC" command, they must be explicitly enabled:

```
CALC:SCALE:STATE ON,(@1(0:7),PATH999)
```

This command enables both the gain and the offset for the channel. At power-up, and after executing a "*RST" command, all channel and path gains are set to 1, and all offsets are set to 0.

There is no need to explicitly enable the gain and offset set with the "`SET GAIN = xxx`" and "`SET OFFSET = xxx`" directives inside a sequence. Once these commands are found in the sequence, they are always enabled.

Alarm limits may also be assigned in either of two ways. The upper alarm may be set with the command:

```
CALC:LIMIT:UPPER 5.25,(@1(0:7),PATH999,2(33))
CALC:LIMIT:UPPER:STATE ON,(@1(0:7),PATH999,2(33))
```

or with the sequence directives

```
“SET ALARMHI = 5.25”
“SET ALARMHI = ON”
```

Similarly, the lower limit can be assigned with:

```
CALC:LIMIT:LOWER -2.25,(@1(0:7),PATH999,2(33))
CALC:LIMIT:LOWER:STATE ON,(@1(0:7),PATH999,2(33))
```

or with the sequence directives

```
“SET ALARMLO = -2.25”
“SET ALARMLO = ON”
```

Finally, units of measurement to report along with the measured value can be assigned with either the command:

```
CALC:SCALE:UNIT ‘mA’,(@1(0:7)),PATH999,2(33))
```

or with the sequence directive

```
“SET UNITS = mA”
```

Checking for Installed Plugs-Ins

The SYST:CTYPE? Command may be used to query the 1830 for the types of cards present in the system.

```
SYSTem
      CTYPEe? <module_number>
```

Individual slots can be queried

```
SYST:CTYPE? 7
```

Where a typical reply would be:

```
EADS NORTH AMERICA TEST AND SERVICES,1450D,09802769,1-1.4-2.5
```

The reply for this command returns the format:

```
<Manufacturer Name>,<Model>,<Serial Number>,<Revision>
```

And the <Revision> field includes 3 numeric values separated by the dash character. These fields are the plug-in card’s firmware revision, hardware revision, and FPGA revision, respectively.

If no card is plugged into the specified slot, the query returns the reply:

```
EADS NORTH AMERICA TEST AND SERVICES,NONE,0,0-0-0
```

System Commands

The 1830 system supports several system commands. These commands generally access errors, SCPI standard information, as well as user-interface issues. Additionally, the 1830 uses a special command to update non-volatile memory. For additional SCPI command information, refer to **Chapter 7, SCPI Command Basics**.

Reading Error Messages

Whenever an error is encountered by the 1830, it performs two actions:

1. One of the bits of the Standard Event Status Register is set. This register may be read using the *ESR? query
2. An error message is added to the error message queue. The error message queue may be read using the SYSTEM:ERROR? query (or SYST:ERR?).

The error queue holds up to 30 error messages. Each time an error is detected, it adds a new error to the error queue. Each time the SYSTEM:ERROR? query is received, the oldest (least recent) error message is returned.

The syntax for this command is shown below:

```
:SYSTem
     :ERRor?
```

The reply to the SYSTEM:ERROR? query uses the format:

```
<error code> , "<error message>"
```

where:

`<error code>`	This parameter is a numeric value. This value is "0" if there are no errors remaining in the error queue. This value is negative when an error exists on the error queue.
`<error message>`	This parameter is a string enclosed in double quotes. The error message provides some additional information about the error.

For example, the reply:

```
0,"No error"
```

indicates that no errors remain on the error queue, while the reply:

```
-102,"Syntax error ; missing @ sign"
```

indicates that a syntax error was detected in a previous command to the 1830. The reply:

```
-350,"Queue overflow"
```

is returned to indicate that all 30 places in the queue have been occupied and that the error queue is full.

SCPI Compliance Information

The SYSTEM:VERSION? query provides information about which SCPI standard the 1830 references and is required for compliance with SCPI.

The syntax for this command is shown below:

```
:SYSTem
     :VERSion?
```

For example, the query:

```
SYST:VERS?
```

always returns the following:

```
1999.0
```

Non-Volatile Storage

The following commands are used to store various types of information in the non-volatile memory of the 1830.

1. `MONitor:STORe`: Stores the alarm monitoring scan list into non-volatile memory.
2. `SCAN:STORe:` Stores the currently defined scan list into non-volatile memory.
3. `INCLude:STORe`: Stores all currently defined include lists into non-volatile memory.
4. `EXCLude:STORe`: Stores all currently defined exclude lists into non-volatile memory.
5. `PATH:STORe:` Stores all currently defined path names into non-volatile memory.
6. `SEQ:STORE <sequence_name>:` Stores the specified sequence script into non-volatile memory.

Likewise, the following commands recall information from non-volatile memory if they were not loaded at power on already using the AUTO load setting:

1. `MONitor:RECall:` Reads the previously stored alarm monitoring scan list from non-volatile memory.
2. `SCAN:RECall:` Reads the previously stored scan list from non-volatile memory.
3. `INCLude:RECall:` Reads all previously stored include lists from non-volatile memory.
4. `EXCLude:RECall:` Reads all previously stored exclude lists from non-volatile memory.
5. `PATH:RECall:` Reads all previously stored path names from non-volatile memory.
6. `SEQ:RECall <sequence_name>`: Reads the specified sequence script from non-volatile memory.

Information can be restored at power on if the AUTO load setting is enabled for the individual information type needed. For example, to restore previously stored path names at power on, use the following command:

```
PATH:RECall:AUTO ON
```

In addition, sequences, scan lists, and measurement data can be stored and read from a USB memory peripheral. To store the currently defined scan list onto a USB memory device, use the following command:

```
MMEM:STORe:SCAN <scanlist_name>
```

and restore the scan list into the 1830 with:

```
MMEM:LOAD:SCAN <scanlist_name>
```

Please refer to the 1830 SCPI reference section for additional information about non-volatile storage on the 1830.

Alarm Monitor Operation

The alarm monitor list is a similar to the scan list, but it provides the ability to conditionally execute the scan list or run a sequence when a measurement exceeds its limit value.

The alarm monitor list allows you to scan a set of channels or paths and compare each measurement made to the limits set for those channels. If the measurement lies outside the alarm values, you can execute the scan list or you can call a sequence.

The alarm monitor scan list is defined with the "[ROUTe]:MONitor" command. For example:

```
MON (@1(0:7),2(25:22), PATHXYZ)
```

As with the scan list, the measurements are made during an alarm scan by associating a sequence with the channels. For example:

```
SEQ:SEL READ_VDC,(@1(0,2,5,24))
SEQ:SEL READ_2WIRE,(@1(1,3:7))
SEQ:SEL READ_IDC,(@1(22,23,25))
SEQ:SEL READ_4WIRE,(@PATHXYZ)
```

associates 4 different sequences to execute with different channels and paths on the alarm monitor list.

The alarm limits are established by using either the "CALC:LIMIT" commands or the "SET ALARMHI = xxx", "SET ALARMLO = xxx" sequence directives as described in the previous section.

When the alarm monitor list is scanned, it compares each reading to the alarm limits assigned to the channel or path being scanned. If the measured value is above the upper limit, or below the lower limit, an alarm condition is detected. The data is stored in the scan memory and the alarm indication in the scan memory indicates whether the measured value is above the upper limit or below the lower limit.

In addition, the 1830 can either execute a user-defined sequence, or execute the normal scan list, when an alarm condition is detected. You can define the sequence to execute, or select the scan list, with the "`[ROUTe:]MONitor:ALARm:SOURce`" command:

```
MON:ALARM:SOURCE SHUTDOWN,(@1(0:7),PATHXYZ)
MON:ALARM:SOURCE SCAN,(@2(33,45))
```

The first command tells the 1830 to execute the sequence named "SHUTDOWN" when any measurement made while scanning channels 0 through 7 on card 1 or path "PATHXYZ" exceed their alarm limits. The second command instructs the 1830 to execute the scan list defined with the "SCAN" command when a measurement made while scanning either channel 33 or 45 from card 2 exceeds their alarm limits.

For additional information, refer to the **Alarm Monitoring Overview** section in **Chapter 7, SCPI Command Basics**.

Emergency Reset Feature

The 1830 system includes an emergency reset feature to enhance system safety by preventing system and card damage due to situations such as over-current conditions. The reset condition is activated by a card with that feature – such as the 1220 16-Amp SPST plug-in switch card.

There are two reset conditions: global and local. A global reset condition quickly opens all the relays in the 1830 system including those on the system backplane and on the installed cards. A local reset condition opens the relays only on the affected card. The default factory assembly setting for the 1220 card is for global reset.

When an emergency reset occurs and you are using the web-page interface, a series of screens appear prompting you to perform certain tasks such as removing the fault condition from the instrument. For additional information, refer to the **Emergency Reset Feature** section in **Chapter 4, Web-Page Interface**.

If you are using IVI, SCPI, or ActivATE interfaces, you may not get a notice of the reset occurring. If you suspect a reset has occurred due to certain card relays being unexpectedly open, use the "`STATus:SHUTdown?`" query and read the reply to confirm that the reset has occurred.

If a reset has occurred, resume the 1830 measurement conditions by

1. First removing the condition which caused the emergency reset.
2. Either toggle the system power off/on OR send a "`*RST`" SCPI command.

You need to rerun the program or software commands to return to the measurement state you were in when the system reset.

Chapter 7
SCPI Command Basics

The 1830 accepts commands over LAN, USB, or GPIB interfaces. These commands follow the rules defined by the SCPI standard. **SCPI** is an acronym for **Standard Commands for Programmable Instruments**, and defines standard command names and syntax rules for commands to the instrument and replies from the instrument.

There are two main groups of commands: IEEE-488.2 Common Commands and SCPI Commands. They are both included in this section.

SCPI commands refer to the 1800-series ***plug-in cards*** as ***modules***.

Input Format

SCPI is an ASCII-based instrument command language designed for test and measurement instruments. SCPI commands are based on a hierarchical structure known as a tree system. In this system, associated commands are grouped together under a common root, thus forming command subsystems. Throughout this chapter, the following conventions are used for SCPI command syntax.

Table 7-1: SCPI Command Syntax Conventions

Square Brackets ([ ])	Enclose optional keywords or parameters.
Braces ({ })	Enclose possible parameters within a command.
Triangle Brackets (< >)	Substitute a value for the enclosed parameter.
Vertical Bar (|)	Separate multiple parameter choices.
Bold Typeface Letters	Designate factory default values.
(Command Only)	This indicates the command cannot be used in query form.
(Query Only)	This indicates the command can only be used as a query. The command form, without the question mark is not allowed.

To illustrate the SCPI notation, a part of the ROUTE command subsystem is shown below:

```
[:ROUTe]
     :CLOSe <channel list>
     :OPEN <channel list>
          :ALL
```

ROUTE is the root keyword of the command. This keyword is optional, since it is shown enclosed in square brackets. CLOSE and OPEN are the next level keywords. The “ALL” keyword is below the OPEN keyword in this command tree.

The colon (:) is used to separate keywords from different levels on the command tree. Each keyword is separated from the next by a single colon.

The SCPI commands which may be formed by this tree are shown below (a <channel list> of (@5(0)) is used in these examples):

```
ROUTE:CLOSE (@5(0))
CLOSE (@5(0))
ROUTE:CLOSE? (@5(0))
CLOSE? (@5(0))
ROUTE:OPEN (@5(0))
OPEN (@5(0))
ROUTE:OPEN? (@5(0))
OPEN? (@5(0))
ROUTE:OPEN:ALL
OPEN:ALL
```

Note that the optional ROUTE keyword is omitted in many of the examples.

Command Keyword Long Form and Short Form

Each keyword defined by SCPI has both a **long form** and a **short form**. The long form is formed by using all letters shown in the keyword. The short form is formed by using only those letters shown in upper-case in the command tree.

The short form is normally three or four letters in length, and ends with a consonant where possible.

For example, the ROUTE keyword may be specified by either of the following:

```
ROUT
ROUTE
```

since the final “e” is shown in lower-case in the command tree.

Only the long form or short form may be used. For example, the keyword “DEFINE” is shown as:

```
DEFine
```

Therefore, the following two command keywords are valid:

```
DEFINE
DEF
```

But the keywords

```
DE
DEFI
DEFIN
```

are NOT valid.

Case Sensitivity

Command keywords are NOT case sensitive. Command parameters are not case sensitive, unless the parameter is string data enclosed in quotes.

For example, the following commands are equivalent:

```
CLOSE
Close
```

Optional Keywords

Command keywords enclosed in square brackets are optional. For example, the following commands are valid and equivalent:

```
ROUTE:CLOSE (@5(0))
CLOSE (@5(0))
```

Querying Parameter Setting

Most of the commands in SCPI have an equivalent query form. The query is used to read the present state of the item that is set with the command. The query is formed by adding a question mark (?) to the end of the command keyword.

For example, the command:

```
ROUTE:CLOSE (@5(0))
```

Has an equivalent query:

```
ROUTE:CLOSE? (@5(0))
```

The command instructs the 1830 to close channel 0 on relay module 5. The query inquires about the present open or close state of channel 0 on relay module 5.

SCPI Command Terminator

For commands sent through the GPIB interface, a command string sent to the 1830 must be terminated with one of the following:

1. An ASCII linefeed character (decimal 10, hex 0A, or a character '\n').
2. The last character with the EOI signal asserted.
3. An ASCII linefeed character with the EOI signal asserted.

The 1830 has the ability to interface with terminals and terminal emulation software. Unfortunately, a wide variation of capabilities exists in terminal/terminal emulators. Most terminal/terminal emulators generally treat the “ENTER” key as a carriage return, operate in half-duplex mode, and have the ability to add linefeeds to incoming carriage returns.

To minimize frustration and obtain the most legible communications, set the terminal/terminal emulator to operate in half-duplex mode and add linefeeds automatically to incoming carriage returns. In this configuration, the 1830 should be configured to expect a carriage return only. Using this method, the terminal display will not have any overwritten lines and be much easier to read and use.

Another option for terminal/terminal emulators that support character translation is to have carriage returns translated into linefeeds. In this scenario, the terminal/terminal emulator still needs to operate in half-duplex mode. However, the 1830, if configured to accept linefeeds only, sends a linefeed with replies. This eliminates the need to add linefeeds to carriage returns since no carriage returns come into play on either side.

The two methods mentioned here are only suggestions and do not cover all possibilities. The 1830 is a flexible instrument and works in nearly all configurations.

IEEE-488.2 Common Commands

The IEEE-488.2 standard defines a set of common commands that perform functions like reset, trigger and status operations. Common commands begin with an asterisk (*), are four to five characters in length, and may include parameters. The command keyword is separated from the first parameter by a blank space. A semicolon (;) may be used to separate multiple commands as shown below:

```
*RST; *STB?; *IDN?
```

The IEEE-488.2 common commands implemented by the 1830 are described later in this chapter. The 1830 complies with the standard GPIB operational modes.

The 1830 supports all required IEEE-488.2 commands. In addition, the optional *OPT?, *SAV, and *RCL commands are supported. The following paragraphs describe the IEEE-488.2 commands supported by the 1830. In addition, the status reporting model implemented by the 1830 is described. The 1830 complies with this standard for GPIB operational modes.

The following commands are implemented:

*CLS Clear status

*ESE Set the Standard Event Status Enable register

*ESE? Read the Standard Event Status Enable register

*ESR? Read the Standard Event Status register

*IDN? Identification query

*OPC Set the OPC bit of the Standard Event Status register

*OPC? Reply with “1" when executed (used for synchronizing)

*RCL Recall relay states from non-volatile memory

*RST Instrument reset

*SAV Store relay states in non-volatile memory

*SRE Set the Service Request Enable register

*SRE? Read the Service Request Enable register

*STB? Read the status byte

*TRG Send a trigger to the instrument over the bus

*TST? Commanded self-test

*WAI Wait-to-Continue

These commands are described in greater detail in this section of the manual.

IEEE-488.2 Status Description

The IEEE-488.2 Status Reporting Model is shown in Figure 7-1. This figure shows how the status reporting data structures are implemented and the commands used to set and read each of the registers. The GPIB operational mode of the 1830 is fully compliant with this standard.

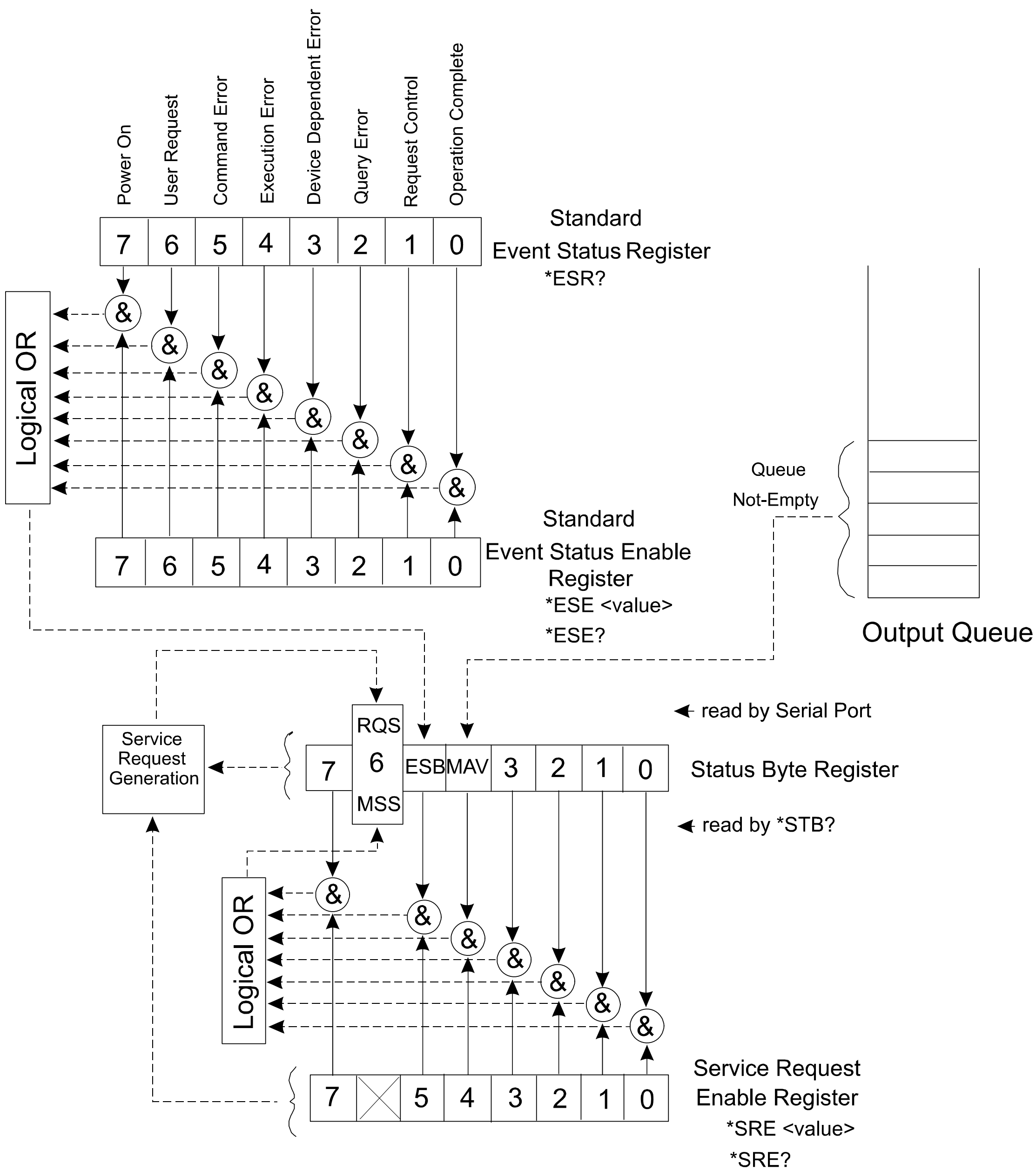


Figure 7-1: IEEE-488.2 Status Reporting Model

Standard Event Status Register

Figure 7-1 shows four related registers. The Standard Event Status Register, the Standard Event Status Enable Register, the Status Byte Register, and the Service Request Enable Register.

The Standard Event Status Register reflects the present status of the instrument. This register consists of eight 1-bit flags. Each flag represents a true or false indication of the corresponding condition.

The bits of this register are described in Table 7-2.

Table 7-2: Standard Event Status Register Commands

Command	Description
PON	Power-On Bit 7 (MSB), Bit weight = 128 decimal = 80 hexadecimal This flag is set when the instrument is powered on.
URQ	User Request Bit 6, Bit weight = 64 decimal = 40 hexadecimal This bit is never set by the 1830 and always reads 0.
CME	Command Error Bit 5, Bit weight = 32 decimal = 20 hexadecimal This bit is set when a command error is detected by the 1830. Errors of this type also result with an error added to the error queue. The error may be read using the SYST:ERR? query to determine the cause of the error.
EXE	Execution Error Bit 4, Bit weight = 16 decimal = 10 hexadecimal This bit is set when a valid command is received but cannot be executed for some reason. For example, an error occurs while executing the *SAV command. When this error occurs, an error message is added to the message queue and may be read using the SYST:ERR? query.
DDE	Device Dependent Error Bit 3, Bit weight = 8 decimal = 8 hexadecimal This bit is set when a device-dependent error is detected. For example, a relay card (module) is read during power-up but contains an unknown identification byte value. When this error occurs, an error is added to the error message queue.
QYE	Query Error Bit 2, Bit weight = 4 decimal = 4 hexadecimal This bit is set when a query error is detected. For example, a query is sent, but the reply is not read, and a second query or command is sent. When this error occurs, an error is added to the error message queue.
RQC	Request Control Bit 1, Bit weight = 2 decimal = 2 hexadecimal This bit is never set by the 1830 and always reads 0.
OPC	Operation Complete Bit 0, Bit weight = 1 decimal = 1 hexadecimal This bit is set when the *OPC command is executed. This may be used to synchronize the 1830 with the commands (to ensure that the 1830 command buffer is empty).

A bit is set in this register when the corresponding condition becomes true. It remains set until the *ESR? query is executed. When the query is executed, the reply contains the present value of the register, and the register is then cleared to 0.

The value returned by the *ESR? query represents a sum of the bit-weight values for all conditions that are true. For example, if the PON bit is set and the QYE bit is set, and the rest of the bits are cleared, then the value returned for the *ESR? query is:

```
PON + QYE = 128 + 4 = 132
```

The Standard Event Status Enable Register provides a mask register. The value of this register is logically AND-ed with the Standard Event Status Register. If the value of this AND-ing is nonzero, then bit 5 of the Status Byte Register is set. This bit is known as the “Event Summary Bit”, or ESB.

For example, if the PON and QYE bits of the Standard Event Status Register are set, but the Standard Event Status Enable Register value is 0, then the ESB of the Status Byte Register will not be set. If either, or both, bits 7 and bit 2 of the Standard Event Status Enable Register are set, then the ESB bit of the Status Byte Register will be set.

Status Byte Register

Another way of viewing the Standard Event Status Enable Register is that it selects which conditions reflected in the Standard Event Status Register are enabled to set the ESB bit of the status byte.

The Status Byte Register is similar to the Standard Event Status Register. Each bit of this register reflects the true or false condition of the corresponding bit. These bits reflect the PRESENT value of the condition, whereas the Standard Event Status Register bits are latched. That is, once a bit in the Standard Event Status Register is set, it remains set until a *CLS command is executed or an *ESR? Query is executed. However, the bits of the Status Byte Register change states as the corresponding condition becomes true or false. These bits are NOT latched.

Also, the Status Byte is not cleared by reading the register. Each bit of the Status Byte remains set until the condition indicated by the bit is no longer present.

The following bits are assigned in the Status Byte Register. All other bits are not used and return “0" when read with the *STB? query:

OSE   Operation Status Event

Bit 7, bit weight = 128 decimal = 80 hexadecimal

This bit is set when any of the bits of the Operation Status Event Register are set. (This bit is NOT shown on the diagram. For a description of the Operation Status Event Register, consult the “SCPI Status Registers” section of this chapter.

MSS   Master Summary Status.

Bit 6, bit weight = 64 decimal = 40 hexadecimal

This bit is set when one or more of the “enabled” bits of the Status Byte are set. In other words:

(Status Byte bit 0 AND SRE bit 0)

OR

(Status Byte bit 1 AND SRE bit 1)

OR

(Status Byte bit 2 AND SRE bit 2)

OR

(Status Byte bit 3 AND SRE bit 3)

OR

(Status Byte bit 4 AND SRE bit 4)

OR

(Status Byte bit 5 AND SRE bit 5)

OR

(Status Byte bit 7 AND SRE bit 7)

where the SRE is the Service Request Enable Register.

ESB Event Summary Bit

Bit 5, bit weight = 32 decimal = 20 hexadecimal

This bit is set when one of the enabled Standard Event Status Enable Register bits is set. The previous paragraphs describe the formation of the ESB bit.

MAV Message Available

Bit 4, bit weight = 16 decimal = 10 hexadecimal

This bit is set when there is a message in the output buffer of the 1830

All other bits (3, 2, 1, and 0) of the Status Byte are not assigned and always return 0.

When the MSS transitions from a 0 to a 1, the GPIB SRQ interrupt is generated. The MSS remains 1 until all enabled bits of the Status Byte have returned to 0.

Service Register Enable Register

The Service Request Enable Register is used to individually enable bits in the Status Byte to set the MSS bit of the Status Byte, thereby generating an interrupt. When the bit of the Service Request Enable bit is set, and the corresponding bit of the Status Byte Register is set, then the MSS bit will be a 1.

The following IEEE-488.2 common commands relate to the Status Reporting Model:

- *CLS Clear status
- *ESE Set the Standard Event Status Enable register

- *ESE? Read the Standard Event Status Enable register
- *ESR? Read the Standard Event Status register
- *SRE Set the Service Request Enable register
- *SRE? Read the Service Request Enable register
- *STB? Read the status byte

These common commands are described below.

CLS Command

The *CLS command clears the SCPI and IEEE-488.2 defined status event registers. These include:

- The Standard Event Status Register is cleared to 0. This register is read using the *ESR? query.
- The Standard Event Status Enable Register is cleared to 0. This register is set using the *ESE command, and read using the *ESE? query.
- The Service Request Enable Register is cleared to 0. This register is set using the *SRE command, and read using the *SRE? query.
- The Operation Status Enable Register is cleared to 0. This register is set using the STATUS:OPERATION:ENABLE command. This register can also be read using the STATUS:OPERATION:ENABLE? query.
- The Operation Status Event Register is cleared to 0 This register is read using the STATUS:OPERATION:EVENT? query.
- The Questionable Status Enable Register is cleared to 0. To set this register, the STATUS:QUESTIONABLE:ENABLE command is issued. To read this register, the query STATUS:QUESTIONABLE:ENABLE? indicates whether this feature is enabled.
- The Questionable Status Event Register is cleared to 0. It is read using the STATUS:QUESTIONABLE:EVENT? Query.

*ESE Command

The *ESE command sets the value of the Standard Event Status Enable Register. The value of this register is logically AND-ed with the contents of the Standard Event Status Register (see the *ESR? query description). If any bits of this AND operation are set, then bit 5 of the Status Byte is set. Bit 5 is known as the Event Summary Bit, or ESB.

ESB of Status Byte =

(bit 0 of ESE AND bit 0 of ESR)

OR

(bit 1 of ESE AND bit 1 of ESR)

OR

(bit 2 of ESE AND bit 2 of ESR)

OR

(bit 3 of ESE AND bit 3 of ESR)

OR

(bit 4 of ESE AND bit 4 of ESR)

OR

(bit 5 of ESE AND bit 5 of ESR)

OR

(bit 7 of ESE AND bit 7 of ESR)

where:

ESE is the value as set by the *ESE command

ESR is the value which may be read with *ESR? query

The *ESE command has the format:

```
*ESE <ESE value>
```

where the “<ESE value>” is an integer numeric value in the range from 0 through 255.

*ESE? Query

This query reads the value presently programmed for the Standard Event Status Enable Register. This reads the value as programmed by the “*ESE” command.

*ESR? Query

This query reads the value of the Standard Event Status Register. Each bit of this register indicates a true/false status condition. When the bit is set, the condition is TRUE; when the bit is cleared, the condition is FALSE. The bit assignments are defined by the IEEE-488.2 specification.

The bits of the Standard Event Status Register are cleared at power-on, except for bit 7, which is set. As conditions become true, the corresponding bit in the register are set. These bits remain set until:

- The *ESR? query is executed
- The *CLS command is executed

The reply to the *ESR? query is a numeric integer value in the range “0" to “255".

*SRE Command

The *SRE command sets the value of Service Request Enable Register. The value of this register is logically AND-ed with the contents of the Status Byte If any bits of this AND operation are set, bit 6 of the Status Byte is set. Bit 6 is also known as the Master Status Summary (MSS) bit.

The command has the format:

```
*SRE <SRE value>
```

where the “<SRE value>” is an integer numeric value in the range 0 to 255. The value of bit 6 of this register is ignored, since it does not make sense to enable an interrupt when an interrupt is generated.

*SRE? Query

This query reads the value presently programmed for the Service Request Enable Register. This reads the value as programmed by the *SRE command. The reply to this command is a numeric value in the range 0 to 255.

*STB? Query

This query reads the value of the Status Byte Register. Each bit of this register indicates a true/false status condition. When the bit is set, the corresponding condition is TRUE; when the bit is cleared, the condition is FALSE. The bit assignments are defined by the IEEE-488.2 specification. The bit assignments are described in the Status Byte Register section of this chapter.

The value returned by the *STB? Query may also be read using the GPIB serial poll command.

Note that bit 6 (MSS) of the Status Byte Register remains set until all enabled conditions are cleared. This is in contrast to the GPIB serial poll operation, where the SRQ bit is set until it is read once, and cleared after.

*OPC Command

The *OPC command causes the Operation Complete bit of the Standard Event Status Register to be set when the command is executed. This is bit 0 of the register.

This command could be used to cause an interrupt (if bit 0 of the Standard Event Status Enable Register is set, and bit 5 of the Service Request Enable Register is set). This provides a means of synchronizing the application program with the 1830 and ensuring that all commands have been parsed and executed before continuing execution of the application program.

The *OPC command has no parameters. The only valid syntax for this command is:

```
*OPC
```

*OPC? Query

The *OPC? query causes the 1830 to reply with the value of "1" when the query is executed. This query may be used to ensure that all previous commands have been executed so the application program may be sure that relays have been programmed to their desired states before continuing execution of the application program.

*IDN? Query

This query requests the instrument to identify itself. The EMS responds to this query with the following reply:

```
EADS NORTH AMERICA TEST AND SERVICES 1830 Switch/Measurement
System,<revision>
```

This reply indicates the manufacturer of the instrument is EADS North America Test and Services, that it is a 1830 Switch/Measurement System, and the current firmware revision. The firmware revision is a numeric, floating point value. An example firmware revision is "1.01". A sample is shown below:

```
EADS NORTH AMERICA TEST AND SERVICES 1830 Switch/Measurement
System,1.01
```

*RST Command

The *RST command resets the instrument to its power-on default state. These settings are shown in Table 7-3.

This command does NOT change the value of SCPI Operation or Questionable status registers or IEEE-488.2 status registers, condition registers, or enable registers. This command does NOT clear the error message queue, the input command buffer, or the output reply buffer.

Table 7-3: Power-On and Reset States

Attribute	Related Command(s)	Reset State
Relay States	*RCL 0 OPEN CLOSE	The states are recalled from non-volatile memory location 0 (power-up state). As shipped from the factory, these are all in the OPEN position.
Trigger Input Source	TRIGGER:SOURCE	Immediate
Trigger Count	TRIGGER:COUNT	1
Trigger Input Delay	TRIGGER:DELAY	0.0 seconds
Trigger Output	OUTPUT:TRIG	Off
Trigger Output Delay	OUTPUT:DELAY	0.0 seconds
Confidence Mode	MONITOR:STATE	Off
Pathname List	ROUTE: PATH	Pathnames stored in non-volatile memory are automatically recalled at power-up.
Scan List	ROUTE: SCAN	No Scan List Defined
Include List	ROUTE: INCLUDE	No Include Lists Defined
Exclude List	ROUTE: EXCLUDE	No Exclude Lists Defined

*TST? Query

The *TST? performs a brief self-test that checks the FPGA and shared memory components of the system. *TST? returns a 0 for success and a -330 for a failure. In case of a failure, the failure is posted to the system event queue, which can be read with the "SYSTem:EVENt? <event number>" query.

The commands "*TST?" and "*TST?" perform a test of the FPGA registers and an abbreviated test of the shared memory. This test executes in approximately 10 seconds

The command "*TST? 1" executes an extended shared memory test. This test executes in approximately 150 seconds.

*RCL Command

The *RCL command recalls the relay states from non-volatile memory. The *RCL command may specify a non-volatile memory location from which to recall the instrument state. That is, both of the following formats are accepted:

```
*RCL
*RCL <location>
```

If <location> is specified, it must be in the range 0 to 100. If <location> is not specified, it defaults to 100.

Location "0" has a special purpose. This location is used by the 1830 at power-up to set the initial relay states.

The *RCL command recalls the states of all relays in the system. The relay states are stored using the *SAV command.

Note that path names, module names, status registers, include lists, and so on are NOT affected by the *RCL command.

*SAV Command

The *SAV command prepares to store a given state into non-volatile memory. The actual process of updating non-volatile memory for this command requires a separate SYSTEM:NVUPD command. This allows multiple relay states and module names to be updated in rapid succession, followed by a single, slower update of the actual non-volatile memory.

The *SAV command accepts one of two formats:

```
*SAV
*SAV <location>
```

If <location> is specified, it must be in the range from 0 through 100. If <location> is not specified, the instrument state is saved into non-volatile memory location 100.

State 0 is recalled at power-up. The 1830 is shipped without any data in state 0. This effectively tells the 1830 to open all relays at power-up. This default may be overwritten by placing all relays in the desired power-up state, and then executing the command

```
*SAV 0
```

In the manner indicated earlier, a separate SYSTEM:NVUPD command must follow to actually update the non-volatile memory.

If new relay modules are added to the system after the *SAV 0 command has been executed, the new relay modules will not be programmed at power-up. Also, if module addresses are changed after the execution of the *SAV 0 command, the modules whose addresses have changed will not be programmed.

In general, whenever new modules are added, or module addresses are changed, then the *SAV 0 command should be used to place the relays of the new modules into the desired power-up state.

*TRG Command

The *TRG command is required by the IEEE-488.2 specification. If the 1830 is armed (see the INIT:IMMEDIATE and INIT:CONTINUOUS commands), and the trigger source is "BUS" (see the TRIGGER:SOURCE command), then this causes the next scan list action to occur.

This is equivalent to sending a GPIB bus trigger.

*WAI Command

The *WAI command is required by the IEEE-488.2 specification. This command is accepted but has no effect on the 1830.

SCPI Commands

SCPI Parameter Type

The SCPI language defines several different data formats to be used in program messages and response messages.

Numeric Parameters

Commands that require numeric parameters accept all commonly used decimal representations of numbers including optional signs, decimal points, and scientific notation.

```
TRIG:DELAY 0.035
```

When a real, non-integer value is returned in a reply from the 1830, the floating point notation is used. The only non-integer values returned from the 1830 correspond to the TRIGGER:DELAY and the OUTPUT:DELAY. The values will be a number between 0.0 and 10.0. At most, 6 digits follow the decimal point.

Integer values may be sent in the command using decimal, octal, hexadecimal, or binary values. The default base for values is decimal.

To specify a hexadecimal value, use the prefix #H. To specify an octal value, use the prefix “#Q”. To specify a binary value, use the prefix “#B”. The following values are all equivalent.

```
123           123 decimal
#B1111011     1111011 binary = 123 decimal
#H7B          7B hex = 123 decimal
#Q173         173 octal = 123 decimal
```

When an integer value is returned In a reply from the 1830, the value is a decimal number.

Discrete Parameters

Discrete parameters are used to program settings that have a limited number of values. Parameters are NOT case sensitive. As an example of the discrete parameter, the TRIGGER:SOURCE command is specified as:

```
:TRIGger
     :SOURce    { BUS | HOLD | IMMediate | EXT }
```

Meaning the parameter must be one of the following:

```
BUS
HOLD
IMM
IMMEDIATE
EXT
```

Note that, just like command keywords, discrete parameters may be specified using either the long form or the short form.

Whenever a discrete parameter is used, the query form of the command returns the SHORT form of the parameter value, in upper-case characters. That is, the command may be specified using either “IMMEDIATE” or “IMM”, but the query:

```
TRIGGER:SOURCE?
```

returns the reply

```
IMM
```

NOT

```
IMMEDIATE
```

Boolean Parameters

Boolean parameters represent a single binary condition that is either true or false. The 1830 accepts "OFF" or "0" for a false condition. The 1830 accepts "ON" or "1" for a true condition. The following command uses a Boolean parameter:

```
[:ROUTe]
     :MONitor
          [:STATe]  { OFF | ON | 0 | 1 }
```

The following commands turn the monitor OFF:

```
ROUTE:MONITOR:STATE OFF
ROUTE:MONITOR:STATE 0
MON OFF
MON 0
```

The following commands turn the monitor ON:

```
ROUTE:MONITOR:STATE ON
ROUT:MON:STAT ON
MON 1
```

When Boolean parameters are queried, the 1830 always replies with a “1", if the state is on, or “0", if the state is off. The keywords “ON” and “OFF” are NOT returned in the reply to ROUTE:MONITOR:STATE?, or any other query.

Reply Output Buffer

The 1830 maintains an output buffer for sending replies to commands. This buffer is 10240 characters in length.

Although unlikely, the output buffer could become filled with replies to commands. If the output buffer is filled, the SCPI-defined “QUERY Deadlock” condition is detected by the 1830. In this case, the output buffer is cleared and an error is added to the error queue. In addition, the query error bit (QYE) of the IEEE-488.2 Standard Event Status Register will be set. The error queue may be read using the “SYST:ERR?” query, while the Standard Event Status Register may be read using the *ESR? query.

When a reply is in the output queue, the message available, or MAV, bit of the status byte is set. The status byte of the 1830 may be read using the GPIB serial poll feature.

Specifying Channels in Commands

To select a single channel in a command, both the address of the relay plug-in which contains the relay, and the channel number for the relay must be specified. The syntax to describe a single channel is:

```
(@<address> ( <channel> ) )
```

where

<address> This is a number in the range “1” to “8” and corresponds to the slot number (1 through 8) of the relay plug-in.

<channel> This is a number which identifies a single relay to operate. The range of valid values for <channel> depends on the particular Adapt-a-Switch plug-in being controlled.

So, to close the relay channel 17 on the plug-in with address 3 (slot 3), use the command:

```
CLOSE (@3(17))
```

Multiple channels for a single plug-in may be specified using the syntax:

```
(@<address>(<channel>,<channel>...))
```

So the command:

```
CLOSE (@3(1,5,9,11))
```

may be used to close channels 1, 5, 9, and 11 on the plug-in with address 3.

A range of channels for a single plug-in may be specified by using the syntax:

```
(@<address>(<channel1>:<channel2>))
```

This format indicates that all relays between <channel1> and <channel2> are to be operated. The command:

```
CLOSE (@3(1:10))
```

closes channels 1 through 10, inclusive, on the relay plug-in with address 3.

A range of relays and a list of single relays may be mixed in a command. For example, the command:

```
CLOSE (@3(1:10,12,15,17:19))
```

Closes channels 1 through 10, 12, 15, 17, 18, and 19 on the plug-in with address 3.

Multiple relays on multiple plug-ins may also be specified. In general, the syntax:

```
(@<address>(<channel list>),<address>(<channel list>),...)
```

is used. For example:

```
CLOSE (@3(1:10, 17), 7(15),8(8:10))
```

This closes the following relays:

- Slot 3: channels 1 through 10 and 17
- Slot 7 channel 15
- Slot 8 channels 8 through 10

SCPI Command Summary

The embedded software implements the following SCPI (and SCPI-like) commands. For SCPI commands specific to the DMMs, refer to **DMM-Specific SCPI Command Summary** section later in this chapter.

```
:CALCulate
    :SCALe
        :GAIN          <gain> [ , <channel list> ]
        :GAIN?         <channel list>
        :OFFSet        <offset> [ , <channel list> ]
        :OFFSet?       <channel list>
        :STATE         { ON | OFF | 1 | 0 } [ , <channel list> ]
        :STATE?        <channel list>
        :UNIT          "<units>" [ , <channel list> ]
        :UNIT?         <channel list>
    :LIMit
        :LOWer         <threshold> [ , <channel list> ]
           :STATe      { ON | OFF | 1 | 0 } [ , < channel list> ]
           :STATe?     <channel list>
        :LOWer?        <channel list>
        :UPPer         <threshold> [ , <channel list> ]
        :STATe         { ON | OFF | 1 | 0 } [ , < channel list> ]
```

```
                :STATe?         <channel list>
                :UPPer?         <channel list>

       :CLOCk
            :OUTPut
                :SOURce         EXTernal|INTernal|1…9|11…14|NONE
                :SOURce?
            :MODule<N>          (N = 1…9, 11…14)
                :SOURce         EXTernal|INTernal|1…9|11…14|EXTernal|NONE
                :SOURce?
       :DIAGnostic
            :RELay
                :CYCLes?        <channel list>
                :CYCLes
                   :CLEar       <channel> , “password”
       :FETCh
            [:IMMediate]?       <chan list>
            :PROGram?           [<first point> [, <last point> ]]
       :FORMat
            :READing
                :ALARm          { ON | OFF | 1 | 0 }
                :ALARm?
                :CHANnel        { ON | OFF | 1 | 0 }
                :CHANnel?
                :SOURce         { ON | OFF | 1 | 0 }
                :SOURce?
                :TIME           { ON | OFF | 1 | 0 }
                   :TYPE        { ABSolute | RELative }
                   :TYPE?
                :TIME?
                :UNIT           { ON | OFF | 1 | 0 }
                :UNIT?
       INITiate
            [:IMMediate]
       :ABORt
       :MEASure
            :VOLTage
                [:DC]?          <power supply channel list>
            :TEMPerature?       <thermistor channel list>

     :ROUTe
       :CLOSe                   <channel list>
              [:STATe]?         <channel list>
            :OPEN               <channel list>
```

```
   [:STATe]?        <channel list>
   :ALL             <module list>
:MONitor            <monitor channel list>
   :CLEar
   :ALARm
        :SOURce     {sequence name|SCAN} [,<channel list>]
        :SOURce?    <channel list>
   :ADVance
        :SOURce     { IMMediate | BUS | EXTernal | TIMer }
        :SOURce?
   :STORe
   :RECall
        :AUTO
        :AUTO?      { ON | OFF | 1 | 0 }
    :INITiate
        [:IMMediate]
        :AUTO
        :AUTO?      { ON | OFF | 1 | 0 }
:MONitor?
:SCAN               <scan channel list>
   :CLEar
   :ADVance
        :SOURce     { IMMediate | BUS | EXTernal | TIMer }
        :SOURce?
   :STORe
   :RECall
        :AUTO
        :AUTO?      { ON | OFF | 1 | 0 }
:SCAN?
:INCLude            <channel list>
   :CLEar           <channel list>
        :ALL
   :STORe
   :RECall
        :AUTO       { ON | OFF | 1 | 0 }
        :AUTO?
:INCLude?           [ <channel list> ]
:EXCLude            <channel list>
   :CLEar           <channel list>
        :ALL
   :STORe
   :RECall
        :AUTO       { ON | OFF | 1 | 0 }
        :AUTO?
```

```
:EXCLude?             [ <channel list> ]
:PATH
   :CATalog?
   :DEFine            <path name>,<channel list> [,<channel list>]
   :DEFine?           <path name>
   :DELete
        [:NAME]       <path name>
        :ALL
   :STORe
   :RECall
        :AUTO         { ON | OFF | 1 | 0 }
        :AUTO?
:CHANnel
   :DELay             {<seconds>|MIN|MAX|DEF}, <channel list>
   :DELay?            [{MIN|MAX|DEF},] <channel list>
:CONFidence
   :STATe             { ON | OFF | 1 | 0 }
   :STATe?
:VERify?              <channel list>
   :ALL?
   :MASK              {NORMal|INVerted| X }, <channel list>
   :MASK?             <channel list>
:ORDer                {BBM|MBB|IMMediate}, <channel list>
:ORDer?               <channel list>
:SEQuence
   :CATalog?
   :DEFine            sequence name
        :FIRSt?       sequence name
        :NEXT?        sequence name
   :STEP              "commands"
   :END
   :CLEar             <channel list>
   :SELect            {sequence name},<channel list>
   :SELect?           <channel list>
   :DELete            sequence name
        :ALL
   :INITiate                sequence name
   :STATus?
   :RESult?
   :STORe                   sequence name
        :DELete             sequence name
              :ALL
   :RECall                  sequence name
        :ALL
```

```
              :AUTO            { ON | OFF | 1 | 0 }
              :AUTO?
              :POWeron         sequence name
                    :AUTO      { ON | OFF | 1 | 0 }
                    :AUTO?
       :SWEep
          :COUNt               { <count> | MIN | MAX }
          :COUNt?
       :STATus
          :PRESet              (Command Only)
          :OPERation
              [:EVENt]?        (Query only)
              :CONDition?      (Query only)
              :ENABle <NRf>
              :ENABle?
          :QUEStionable
              [:EVENt]?        (Query only)
              :CONDition?      (Query only)
              :ENABle <Nrf>
              :ENABle?
       :SYSTEM
          :ERRor?
          :EVENt
              :COUNt?
              :CLEar           <event number>
                    :ALL
          :EVENt?              <event number>
          :PREset
          :DATE                <yyyy> , <mm> , <dd>
          :DATE?
          :TIME                <hh> , <mm> , <ss.ssssss>
          :TIME?
          :VERSion?
          :CPON                <module address> | ALL
          :CTYPe?              <module address>
          :MODule
              :CHANnel
                    :LIST?     <module address>
              :RELay
                    :LIST?     <module address>
   :SYSTem
       :COMMunicate
         :ENAble               {ON|OFF|1|0}, {GPIB|USB|LAN|WEB}
         :ENAble?              { GPIB | USB | LAN | WEB }
```

```
      :GPIB
            :ADDRess          <GPIB address>
            :ADDRess?
      :LAN
            :MODe             { AUTO | MANUAL }
            :MODe?
            :MANual
                  :IPADress   <IP address>
                  :IPADress?
                  :DNS        <IP address>[,<IP address>[,<IP address>]]
                  :DNS?
                  :SMAsk      <mask>
      :SMAsk?
      :GATEway                <IP address>
      :GATEway?
      :DOMain           “<name>”
      :DOMain?
      :GATEway?
      :HOSTname         “<name>”
      :HOSTname?
      :IPADdress?
      :KEEPalive        { <seconds> | MIN | MAX }
      :KEEPalive?       [ { MIN | MAX } ]
      :MAC?
      :PING             { ON | OFF | 1 | 0 }
      :PING?
      :SMASk?
:TRIGger
     :SOURce                 {IMMediate|BUS|EXTernal|TIMer|USER}
     :SOURce?
     :TIMer                   { <timer interval> | MIN | MAX }
     :TIMer?                  [ { MIN | MAX } ]
     :COUNt                   { <trigger count> | MIN | MAX }
     :COUNt?                  [ { MIN | MAX } ]
     :MODule1
      :SOURce           <trigger input>
      :SOURce?
      :DELay            { <seconds> | MIN | MAX }
      :DELay?           [ { MIN | MAX } ]
      :LEVel*           { HIGH | LOW }
      :LEVel?
      :SLOPe*           { POSitive | NEGative }
      :SLOPe?
      :SENSitivity*     { EDGE | LEVEL }
```

```
        :SENSitivity?
        :ENABle             { ON | OFF | 1 | 0 }
        :ENABle?
        :WIDTh              { <seconds> | MIN | MAX }
        :WIDTh?             [ { MIN | MAX } ]
      :MODule2              (repeat command tree for MODule1)
      :MODule3              (repeat command tree for MODule1)
        .                                                    .
        .                                                    .
        .                                                    .
      : MODule9             (repeat command tree for MODule1)
      : MODule11            (repeat command tree for MODule1)
      : MODule12            (repeat command tree for MODule1)
      : MODule13            (repeat command tree for MODule1)
      : MODule14            (repeat command tree for MODule1)
      :EXTernal2            (repeat command tree for OUTPut1, except*)
      :SCAN                 (repeat command tree for OUTPut1, except*)
```

Note: <trigger input> = NONE|INPut1|INPut2|...|INPut12|EXT1|
EXT2|SCAN|ALARMLO|ALARMHI|ALL

```
:TRIGger
      :INPut1
            :LEVel              { HIGH | LOW }
            :LEVel?
            :SLOPe              { POSitive | NEGative }
            :SLOPe?
            :SENSitivity        { EDGE | LEVEL }
            :SENSitivity?
      : INPut2                  (repeat command tree for INPut1)
      : INPut3                  (repeat command tree for INPut1)
      : INPut13                 (repeat command tree for INPut1)

            :DATA
                  :MODE         ROLLover | FILL
                  :POINts?
                  :DELete       <points>
                     :ALL


:MMEMory
      :CATalog?
      :DELete                <file name>
```

```
:LOAD
      :SEQuence      <sequence name>
      :SCAN          <scan list name>
:STORe               <file name>
      :SEQuence      <sequence name>
      :SCAN          <scan list name>
      :DATA          <data file name>[,<first point>[,<last point>]]
```

SCPI Status Data Structure Requirements

The embedded software implements two sets of registers known as the Operation Status Register and the Questionable Status Register. Each comprises a hierarchy of registers, where each lower level provides a single summary bit to the next higher level.

Operation Status Registers

The Operation Status Register set holds current status information regarding the 1830 platform.

The present status of the plug-in module is reflected in the status bits of the register. Many of the bits shown in Table 7-4 are reserved by the SCPI specification. These include bits 0 through 7, plus bit 14. However, only the following bits are required to be implemented for the 1830 platform.

Table 7-4: Operation Status Register Bits

Bit #	Name	Description
0	Calibrating	Set when the calibration of the platform or any of the cards (e.g. DMM) is executing
1	Settling	Set when any of the relays are settling
2	Ranging	Set when the range of any instrument (e.g. DMM) is being changed
3	Sweeping	Set when a signal source is sweeping across a range of frequencies. Not presently supported by any plug-in card
4	Measuring	Set when a measurement has begun and cleared when the measurement has completed
5	Waiting for Trigger	Set when the instrument has been armed but has not yet received a trigger
6	Waiting for Arm	Set when the instrument is waiting to be armed before it can subsequently be triggered to perform some module-specific action
7	Correcting	Not used by the 1830 platform
8	Trigger Asserted	Set when the trigger output is asserted
14	Running Sequence	Set when the platform is executing a scan list, alarm monitor list, or a sequence

Each of the bits in the Operation Status Enable Register is AND-ed with the corresponding bits in the Operation Status Event Register. If the value of this AND-ing is nonzero, then bit 7 of the IEEE 488.2 status byte is set, otherwise it is clear.

Questionable Status Registers

The Questionable Status Register provides notification that the platform has detected a suspect condition. At the bottom of the hierarchy is a set of 12 Questionable Instrument Summary registers. Each of these registers reflects the status for one of the plug-in modules.

Many of the bits shown for the Questionable Instrument Summary registers are defined and reserved by the SCPI specification. However, only those items shown in the table below are required to be set according to the plug-in module status.

Table 7-5: Questionable Instrument Summary Status Register Bits

Bit #	Name	Description
9	Diag Fault	Set when the plug-in module has failed on board diagnostics or is otherwise disabled due to a fault
10	XBUS Fault	Set when the transmission failed because an illegal memory access was attempted or due to a transmission time-out
11	Local Reset	Set when plug-in module is held in reset because the local reset (external edge connector) is asserted
12	Global Reset	Set when the plug-in module is held in reset because the global reset (backplane) line is asserted
14	Command Warning	Set when a command is received by the plug-in module that it understands and executes but may result in a different state than expected. For example, an extraneous command parameter is ignored.

Each of the bits in the Questionable Status Enable Register is AND-ed with the corresponding bits in the Questionable Status Event Register. If the value of this AND-ing is nonzero, then bit 3 of the IEEE 488.2 status byte is set, otherwise it is clear.

ROUTe Command Summary

The `ROUTe` command is the default command tree. That is, the "`ROUTe`" is implied when the `CLOSE`, `OPEN`, and other sub-commands are encountered in a command. The `ROUTe` command tree is shown below:

```
[:ROUTe]
      :CLOSe             <channel list>
         [:STATe]?       <channel list>
      :OPEN              <channel list>
         [:STATe]?       <channel list>
         :ALL            <module list>
      :MONitor           <monitor channel list>
         :CLEar
         :ALARm
            :SOURce      {NONE|SCAN|<sequence name>}[,<chan list>]
            :SOURce?     <channel list>
         :ADVance
            :SOURce      { IMMediate | BUS | EXTernal | TIMer }
            :SOURce?
         :STORe
```

```
      :RECall
         :AUTO
         :AUTO?       { ON | OFF | 1 | 0 }
      :INITiate
         [:IMMediate]
         :AUTO
         :AUTO?       { ON | OFF | 1 | 0 }
   :MONitor?
   :SCAN              <scan channel list>
      :CLEar
      :ADVance
         :SOURce      { IMMediate | BUS | EXTernal | TIMer }
         :SOURce?
      :STORe
      :RECall
         :AUTO
         :AUTO?       { ON | OFF | 1 | 0 }
   :SCAN?
   :INCLude           <channel list>
      :CLEar          <channel list>
         :ALL
      :STORe
      :RECall
         :AUTO        { ON | OFF | 1 | 0 }
         :AUTO?
   :INCLude?          [ <channel list> ]
   :EXCLude           <channel list>
      :CLEar          <channel list>
         :ALL
      :STORe
      :RECall
         :AUTO        { ON | OFF | 1 | 0 }
         :AUTO?
   :EXCLude?          [ <channel list> ]
   :PATH
      :CATalog?
      :DEFine         <path name>,<channel list>[,<channel list>]
      :DEFine?        <path name>
      :DELete
         [:NAME]      <path name>
         :ALL
      :STORe
      :RECall
         :AUTO        { ON | OFF | 1 | 0 }
```

```
        :AUTO?
  :CHANnel
     :DELay          {<seconds>|MIN|MAX|DEF},<channel list>
     :DELay?         [{MIN|MAX|DEF},]<channel list>
  :CONFidence
     :STATe          { ON | OFF | 1 | 0 }
     :STATe?
  :VERify?           <channel list>
     :ALL?
     :MASK           {NORMal|INVerted|X},<channel list>
     :MASK?          <channel list>
  :ORDer             {BBM|MBB|IMMediate},<channel list>
  :ORDer?            <channel list>
```

Channel Lists

Many of the commands in the ROUTe command tree expect a <channel list> parameter. A channel list is a means of specifying one or more channels from one or more modules. In addition, a <channel list> can specify a path name as well.

The elements of a <channel list> are as follows:

<channel list>	::=	(@ <channel_id> [, <channel_id> ...])
<channel_id>	::=	<module_channel> | <path name>
<module_channel>	::=	<module_address> (<channel_range>)
<path name>	::=	IEEE 488.2 <Character Program Data>
		This is basically a name with a leading “A” through “Z” (upper or lower case), followed by zero or more characters. Each of the characters after the first must be “A” through “Z”, “0" through “9" or the undersoce (“_”) character. A maximum of 44 characters may be used for any <path name> (SCPI defines a maximum of 12). The name is NOT case sensitive, so that “DMM”, “dmm” and “Dmm” are equivalent.
Examples:		A
		A1
		DMM_TO_UUT_PIN_23
<channel_range>	::=	<channel> | <channel> : <channel>
		The first format identifies a single relay. The second format identifies a range of relays. The relays may be in ascending OR descending order. That is “1:10" and “10:1" are both acceptable.
<channel>	::=	IEEE 488.2-defined “NR1" numeric

		(e.g. 13, 7001, etc.)
<module list>	::=	<module_range> [, <module_range> ...]
<module_range>	::=	<module address> | <module address> : <module address>
<module address>	::=	0 (for 18x0 analog bus connection channels)
		1 | 2 | 3 | ... | 9 (for switch plug in cards)
		11 | 12 | 13 | 14 (for PXI plug in modules)

The table below displays many examples of channel lists:

Table 7-6: Example Channel Lists

Channel List	Description
`(@1(100))`	Identifies channel 100 of module 1
`(@1(100,203))`	Identifies channels 100 and 203 of module 1
`(@2(37))`	Identifies channel 37 of module 2
`(@1(201:203))`	Identifies channels 201, 202, and 203 of module 1
`(@6(101,202,300:303))`	Identifies channel 101, 202, 300, 301, 302, and 303 of module 6
`(@1(100),2(37))`	Identifies channel 100 of module 1 and channel 37 of module 2
`(@7(100,200:202),12(44:47,77))`	Identifies channels 100, 200, 201, and 202 of module 7, and channels 44, 45, 46, and 77 of module 12

ROUTE:CLOSE Command Summary

The “ROUTE:CLOSE” Command is used to close one or more relays located on one or more plug-in modules. The command syntax is:

```
[ROUTe:]CLOSe <channel list>
[ROUTe:]CLOSe? <channel list>
```

The examples shown below presume the following:

- There is a relay plug-in module in slot #1 and another in slot #2
- The path name “DMM_TO_UUT_PIN_23" has been defined as closing channel 100 and 102 in slot #1, closing channel 55 in slot #2, and opening channel 304 in slot #1. This has been defined using the “ROUTe:PATH:DEFine” command.

Examples:

`:ROUTE:CLOSE (@1(103))`	Closes channel 103 on card in slot 1

`CLOS (@1(103))`	Closes channel 103 on card in slot 1
`CLOS (@DMM_TO_UUT_PIN_23)`	Closes path "DMM_TO_UUT_PIN_23", which means closing channels 100 and 102 in slot 1, closing channel 55 in slot 2, and opening channel 304 in slot 1.
`:CLOSE (@2(15:12))`	Closes channels 15, 14, 13, and 12 on the card in slot 2
`ROUT:CLOSE (@2(15:12))`	Closes channels 15, 14, 13, and 12 on the card in slot 2
`:CLOS(@1(8303),2(77))`	Closes channel 8303 in slot 1 and channel 77 in slot 2.

The semantics of relay closing are as follows:

- First, each relay is checked to verify that it is a valid channel designation.
- For each relay being operated in the <channel list>, the system determines whether the relays are "INCLuded" with other relays. Each relay which is to be closed due to the "INCLude" list is marked as "going to be closed"
- For each relay which is in the <channel list>, OR which is going to be closed because it is "INCLuded" with a relay in the <channel list>, the relay is checked to see if it is "EXCLuded" with any relays in the system. Each relay "EXCLuded" is marked for opening.
- Since each relay may be on at most one "INCLude" list and one "EXCLude" list, the embedded software can check that no two relays are both included and excluded from each other. This check is performed when the "INCLude" and "EXCLude" commands are processed. Thus, there is no need to check for conflicts when operating relays.
- Each card which has a relay and is in the "Make-Before-Break" mode will have all of the relays marked for closing closed; then the relays marked for opening will be opened.
- Each card which has a relay and is in the "Break-Before-Make" mode will have all of the relays marked for opening opened; then the relays marked for closing will be closed.

The "ROUTE:CLOSE?" query is used to read the present state of the relays. The reply to this command is a sequence of "1" and "0", each separated by a single ASCII space character. The value of "1" is returned if the relay is closed, and the value of "0" is returned if the relay is opened. The values are in the same sequence and correspond 1-for-1 with the order that the channels are specified in the <channel list>.

For example, suppose channels 0, 1, and 3 are open and channel 2 is closed for the plug-in slot #1. The query

```
CLOSE? (@1(0:3))
```

returns the reply:

```
0 0 1 0
```

And the query:

```
CLOSE? (@1(3:0))
```

returns the reply:

```
0 1 0 0
```

The “ROUTE:CLOSE?” query may also be used to request the state of a path. If a path name is specified, then the reply to the query is a “1” for the path if, and only if, all channels in the “close” portion of the path definition are closed, and all channels in the “open” portion of the path definition are open. There is a single reply for each path specified.

For example, support we have two paths defined:

```
PATH:DEF “PATH1”, (@1(0,1,2,3),2(5,7,9)),(@1(14),2(23))
PATH:DEF “PATH2”, (@1(0,2,3,6)),(@2(5))
```

Now, suppose that channels 1(0,1,2,3,6) are closed, 2(5,7,9) are closed, while 1(14), and 2(23) are open. The query:

```
CLOSE? (@PATH1)
```

Returns the reply:

```
1
```

While the reply:

```
CLOSE? (@PATH2)
```

Returns the reply

```
0
```

Because channel 2(5) is closed, but it must be open for PATH2 to be considered closed.

ROUTE:OPEN Command Summary

The “ROUTE:OPEN” command is used to close one or more relays located on one or more plug-in modules. The command syntax is:

```
[ROUTe:]OPEN <channel list>
[ROUTe:]OPEN? <channel list>
[ROUTe:]OPEN:ALL
```

Examples:

`:ROUTE:OPEN (@1(302))`	Opens channel 302 on plug-in card in slot #1
`Open (@6(0:8303))`	Opens all channels between 0 and 8303 on plug-in card in slot 6
`:Open (@3(17:10))`	Opens channels 10 through 17 on plug-in-card in slot 3
`open (@2(23:44,11), 1(101:304))`	Opens channels 23 through 44 and channel 11 on plug-in card in slot #2. Opens

	channels 101 through 304 on plug-in card in slot #1
`Rout:Open:All`	Opens all relays on all cards plugged into any slot in the system

The “ROUTE:OPEN?” query is used to read the present state of the relays. This command is ALMOST identical to the “ROUTE:CLOSE?” query defined previously. The only difference is that the “ROUTE:OPEN?” reply returns a “1" for each relay in the <channel list> which is open and a “0" for each relay which is closed, whereas the “ROUTE:CLOSE?” query reply returns a “1" for each relay which is CLOSED and a “0" for each relay which is OPEN.

For example, suppose channels 0, 1, and 3 are open but channel 2 is closed on the plug-in module in slot #12:

```
OPEN? (@12(0:3))
```

returns the reply:

```
1,1,0,1
```

and the query:

OPEN? (@12(3:0))

returns the reply:

1,0,1,1

Scan List Overview

The embedded software supports the scan list operations described in this paragraph. A scan list is a series of relay channels, path names, and memory state names. A scan list is defined by using the “[ROUTe:]SCAN” command. A scan list must first be defined before it can be used.

After a scan list is defined, the following sequence occurs:

1. The scan list must be armed. This is done with the “INITiate” command. Either the “INITiate”, or the “INITiate:IMMediate” command must be received.
2. Once the scan list is armed, it waits for a trigger to be received from the currently selected trigger source. The ”TRIGger:SOURce” command may be used to select one of the following trigger sources:
    a. EXTernal – the external trigger source is selected. A pulse on the EXT1 trigger input will commence scanning
    b. BUS – the system will wait until the “*TRG” command is received. For the GPIB interface, a Group Execute Trigger (GET) signal also provides the trigger.
    c. IMMediate – the trigger is implicitly present, scanning commences immediately after arming
    d. TIMer – the trigger is generated according to the trigger timer interval. See the “TRIGger:TIMer” command for a description of the trigger

timer.

e. USER – the selected inputs on the trigger control block are used. This is a very flexible trigger control scheme that allows the user to select one or more inputs, including the EXT1 or EXT2 inputs, and the trigger outputs from plug-in cards, to trigger the instrument.

3. The first channel or path is closed, or the state name is recalled, or the user-defined sequence is executed.
4. The 1830 system then waits for the channel delay time of the channel or path that was closed. The channel delay time defaults to the relay settling time, for a single channel, or for the longest settling time, for a path. The channel delay time may be set to a different value with the "[ROUTe:]CHANnel:DELay" command.
5. Any user-defined sequence associated with the channel or path will be executed.
6. The 1830 system will generate a trigger output on the SCAN output trigger line. If SCAN is the selected trigger source for any output trigger, using the "TRIGger:MODule<N>:SOURce SCAN", the associated trigger will be generated.
7. The 1830 system will wait to receive a trigger on the scan list advance trigger source. This trigger source is set by using the "[ROUTe:]SCAN:ADVance:SOURce" command. The scan list advance trigger source can be any of the following:
    a. EXTernal – the EXT1 input on the system is selected. Note, EITHER the trigger source OR the scan list advance trigger source can be EXTernal, but NOT both.
    b. BUS – the 1830 system waits for the "*TRG" command to move to the next step in the scan list. The GPIB interface may also send the Group Execute Trigger (GET) to advance the scan list.
    c. IMMediate – the 1830 system immediately moves to the next step of the scan list.
    d. USER – the selected input(s) from the trigger control block are used. This expands the available trigger source beyond the EXT1 input.
8. Once the scan list advance trigger source is received, it opens the previous channel or path that was closed in step 3 above.
9. The 1830 system then waits for the channel delay time of the channel or path to ensure the channel or path is opened.
10. The 1830 system then closes the next channel, path, or recalls the next state from memory, or executes the next user-defined sequence.
11. Steps 4 through 11 above continue until each element of the scan list (channel, path, memory location) have been processed once. At this point, one "sweep" of the scan list has been completed.
12. The user may specify more than 1 sweep. The "SWEep:COUNt" command is used to specify the number of sweeps. If the sweep count is greater than 1, then the scan list begins at the start of the scan list again.
13. When the number of sweeps (= sweep count) have been completed, the

scan list has completed the actions required for the trigger from the trigger source. If the trigger count, as set by the "TRIGger:COUNt" command, is 1, then the scan list has completed and will ignore subsequent triggers. In this case, the 1830 platform is in the IDLE trigger state.

14. If the trigger count is greater than one, then the next trigger initiates another set of sweeps (= sweep count) through the scan list. The trigger enters the IDLE state when the number of triggers received equals the trigger count.

The scan list use of sweep count and trigger count is illustrated in the Figure 7-2.

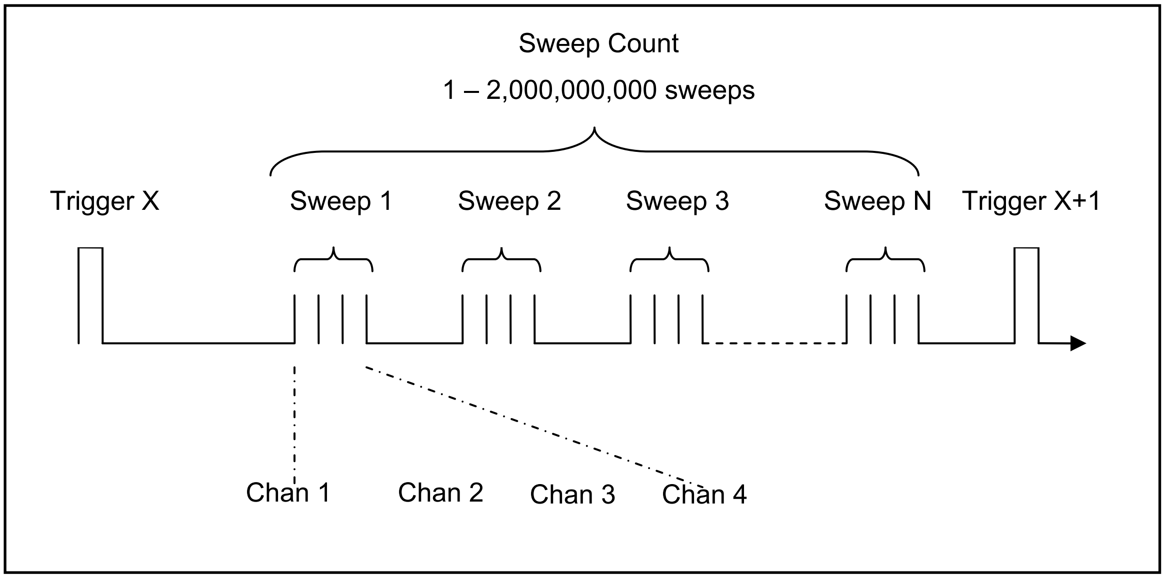


Figure 7-2: Scan List with Sweep Count and Trigger Count

The rules regarding TRIGGER COUNT and SWEEP COUNT above have the following implications:

- The total number of measurements taken (assuming each channel in the scan list has a single measurement associated with it) will be equal to:

  (TRIGGER COUNT x SWEEP COUNT)

- The total number of triggers required on the TRIGger:SOURce will be equal to TRIGGER COUNT
- The total number of triggers required on the SCAN:ADVance:SOURce will be equal to SWEEP COUNT
- The total number of triggers generated on the selected output triggers will be equal to SWEEP COUNT

SWEep:COUNt Command Summary

The "SWEep:COUNt" command is used to set the number of times the scan list will perform a sweep on a single trigger. The format of this command is:

```
SWEep:COUNt { <sweep count> | MIN | MAX }
```

Where the <sweep count> is any number between 1 and 2,000,000,000. The value MIN represents 1, and the value MAX represents 2,000,000,000.

After power-up, and "*RST" reset, the value is set to 1.

The query

```
SWEep:COUNt?
```

is used to read back the present setting of the sweep count.

Note that the sweep count applies only to scan lists. The alarm monitoring feature assumes continuous scanning until disabled.

ROUTe:SCAN Command Summary

The "ROUTe:SCAN" command is used to define a "scan list". A scan list defines a series of individual elements, which may be individual channels, path names, sequences, or memory state locations, to operate.

The format for the "ROUTe:SCAN" command is:

```
[ROUTe:]SCAN <scan list>
```

Where <scan list> is a channel list comprised of channels, channel ranges, path names, and memory state names, separated by commas.

When a single channel is specified, this means that the single channel will be closed when the scan list reaches that element.

When a channel range is specified (e.g. "23:45"), the scan list is treated as though each channel in the range was specified, separated by commas. This means that each channel in the range is closed one at a time.

If a path name is specified, all of the channels in the "close channel" portion of the path definition are closed when the path is closed. All channels in the "open channel" portion of the path definition (if any) are opened when the path is closed. When the path is opened, ONLY the channels in the "close channel" portion of the path are opened.

State names are indicated by the name "STATE", followed by a number indicating which memory state is being recalled. For example, the name "STATE4" identifies that memory location 4 (saved using the command "*SAV 4") is to be recalled. When a state is "closed" by the scan list, all of the relay channels in the 1830 platform are placed in the state they had when the "*SAV" command was executed. When the state is "opened" by the scan list, there is NO change to the state of the relays.

For an example of a scan list, consider the following command:

```
SCAN (@1(0,2,20:23),XYZ,3(15),STATE14,2(14,5:7,1))
```

When the scan list is executed, the following actions would take place in response to successive triggers:

Trigger #	Opened channels / paths	Closed channels / paths
1	None	Module 1, channel 0
2	Module 1, channel 0	Module 1, channel 2
3	Module 1, channel 2	Module 1, channel 20
4	Module 1, channel 20	Module 1, channel 21
5	Module 1, channel 21	Module 1, channel 22
6	Module 1, channel 22	Module 1, channel 23
7	Module 1, channel 23	Path "XYZ"
8	Path "XYZ"	Module 3, channel 15
9	Module 3, channel 15	Memory state 14 recalled
10	None	Module 2, channel 14
11	Module 2, channel 14	Module 2, channel 5
12	Module 2, channel 5	Module 2, channel 6
13	Module 2, channel 6	Module 2, channel 7
14	Module 2, channel 7	Module 2, channel 1
15	Module 2, channel 1	Module 1, channel 0

This example illustrates the use of single channels, channel ranges, path names, and state names.

The currently defined scan list can be recalled by sending the query "ROUTe:SCAN?".

ROUTe:SCAN:CLEAR Command Summary

The "ROUTE:SCAN:CLEAR" command is used to delete the specified scan list. After this command is executed, the specified scan list no longer exists. The format of this command is:

```
[ROUTe:]SCAN:CLEar
```

ROUTe:SCAN:STORe Command Summary

The "ROUTe:SCAN:STORe" command is used to store the scan list into nonvolatile memory for later recall. The format for this command is:

```
[ROUTe:]SCAN:STORe
```

Once a scan list is stored, it may be recalled by using the "[ROUTe:]SCAN:RECall" command. The scan list may be recalled at power-up by using the "[ROUTe:]SCAN:RECall:AUTO" command.

ROUTe:SCAN:RECall Command Summary

The "ROUTe:SCAN:RECall" command is used to recall the scan list from nonvolatile memory for later recall. The format for this command is:

```
[ROUTe:]SCAN:RECall
```

The scan list may be automatically recalled at power-up by using the "[ROUTe:]SCAN:RECall:AUTO" command. The format for this command is:

```
[ROUTe:]SCAN:RECall:AUTO { ON | OFF | 1 | 0 }
```

This command enables (ON or 1) or disables (OFF or 0) the power-up recall of the scan list. The present state of the power-up recall of the scan list may be read using the query:

```
[ROUTe:]SCAN:RECall:AUTO?
```

The query will return "1" if the power-up recall is enabled, or "0" if it is disabled.

ROUTe:SCAN:ADVance:SOURce Command Summary

The "ROUTe:SCAN:ADVance:SOURce" command is used to select the trigger source that advances each step of the scan list. The general form of this command is

```
[ROUTe:]SCAN:ADVance:SOURce{IMMediate|BUS|EXTernal|TIMer|USER}
```

The possible choices for the scan advance source are:

1. EXTernal – the EXT1 external trigger source is selected. A pulse on the EXT1 trigger input will advance to the next element on the scan list until a single sweep has completed.
2. BUS – the 1830 platform will wait until the "*TRG" command is received to advance to the next entry on the scan list until a single sweep has completed. A group execute trigger (GET) from the GPIB is equivalent to receiving this command.
3. IMMediate – the trigger is implicitly present, a single sweep of the scan list will take place immediately.
4. TIMer – the trigger is generated according to the trigger timer interval. As each timer trigger is accepted, the scan list will advance to the next entry until a single sweep has completed.
5. USER – the trigger on any of the selected trigger input(s) is selected. Consult the "TRIGger Command Summary" for a description of how to enable or disable various trigger inputs and also how various attributes of the trigger inputs may be set.

The EXTernal or USER trigger can be selected for either the "SCAN:ADVance:SOURce" or the "TRIGger:SOURce", but not both. If the user attempts to set the "SCAN:ADVance:SOURce" to EXTernal or USER if the "TRIGger:SOURce" is EXTernal or USER, a SCPI error (-221, "Settings conflict") being placed on the SCPI error queue.

Note that the scan list advances through one complete sweep of the scan list. A new trigger on the "TRIGger:SOURce" is required to make a second sweep (assuming the "SWEEP:COUNT" is greater than 1). If the "SWEep:COUNt" is 1, then the trigger must be rearmed with the “INITiate:IMMediate” command to enable a new trigger to start the scan list again.

The present value can be read using the “ROUTe:SCAN:ADVance:SOURce?” query. The query returns the short form of the selection (e.g. “IMM”, not “IMMEDIATE”).

At power-up, and after reset, this is set to IMMediate.

The present scan advance trigger source may be read with the query:

```
[ROUTe:]SCAN:ADVance:SOURce?
```

This query will return one of the following replies:

```
EXT
BUS
IMM
TIM
USER
```

Alarm Monitoring Overview

The Alarm Monitoring feature allows the user to define a scan list which cycles through a set of channels and/or paths. Each channel or path can have an upper and/or lower alarm limit set for the measurement on the channel. If either alarm limit is exceeded, the user can execute the scan list OR a user-defined sequence. When the scan list or sequence completes, the monitoring resumes. Once started with the “[ROUTe:]MONitor:INITiate” command, the alarm monitoring continues until the instrument receives the “ABORt” command.

The “[ROUTe:]MONitor” command is used to define the list of channels and paths to scan.

Each channel or path on the monitor list can have an upper and/or lower alarm limit defined and enabled with the “CALCulate:LIMit:UPPer” and “CALCulate:LIMit:LOWer” commands.

The “MONitor:ALARm:SOURce” determines if a sequence or the scan list is executed when an alarm value is exceeded. It is also possible to set the alarm source to “NONE”, in which case no sequence or scan list is executed if the alarm value is exceeded.

The measurement for each channel is multiplied by the gain defined for the channel with the “CALCulate:GAIN” command. The offset for the channel defined with the “CALCulate:OFFSet” is then added to the measurement. The measurement is then compared against the upper and lower alarm limits for the channel. If the calculated measured value lies above the upper limit, or below the lower limit, the alarm scan list associated with the channel is executed. The “MONitor:ALARm:SOURce” command is used to determine which scan list, if any, will be executed when an alarm condition is detected.

Note that the monitor scan list can use as many different measurements as are installed in the 1830 platform. Each channel may be associated with a different measurement device and/or measurement type, range, precision, and so on. The association between a channel and it's measurement device and type is set with the "SEQuence:SELect" command.

The following example illustrates how the various commands are used to set up and execute the alarm monitoring operation.

`SEQ:SEL "SENSE:VOLT:DC 10,0.01,(@11)", (@1(0:3))`	DMM in slot 11 will be used to measure DC volts on channels 0 to 3 of switch card in slot 1
`SEQ:SEL "SENSE:RES 100e3,1,(@11)", (@1(4:7))`	DMM in slot 11 will be used to measure 2-wire resistance on channels 4 through 7 of switch card in slot 1
`CALC:LIMIT:UPPER 2.0,(@1(0:3))`	Upper limit for channels 0 to 3 is 2.0 (volts)
`CALC:LIMIT:LOWER -1.0(@1(0:3))`	Lower limit for channels 0 to 3 is -1.0 (volts)
`CALC:LIMIT:UPPER 50e3,(@1(4:7))`	Upper limit for channels 4 to 7 is 50K (ohms)
`CALC:LIMIT:UPPER 4.7e3,(@1(4:7))`	Lower limit for channels 4 to 7 is 4700 (ohms)
`CALC:LIMIT:UPPER:STATE ON,(@1(0:7))`	Enable upper alarm limit for channels 0 to 7
`CALC:LIMIT:LOWER:STATE ON,(@1(0:7))`	Enable lower alarm limit for channels 0 to 7
`MONITOR:ALARM:SOURCE SCAN,(@1(0:3))`	Set scan list to execute when alarm detected on channels 0 to 3
`MONITOR:ALARM:SOURCE MYSEQ,(@1(4:7))`	Set sequence MYSEQ to execute when alarm detected on channels 4 to 7
`SEQ:SEL "SENS:VOLT:DC 1.0,0.001,(@12(0))", (@1(10:15))`	DMM in slot 12 will be used to measure DC volts on channels 10 to 15 of switch card in slot 1
`SEQ:SEL "SENS:VOLT:RES 1000,0.1,(@11(0))", (@1(20:30))`	DMM in slot 11 will be used to measure 2-wire resistance for channels 20 to 30 of switch card in slot 1
`SCAN (@1(10:15))`	Define scan list as channels 10 through 15 of card in slot 1
`MON (@1(0:7))`	Monitor channels 0 to 7 of switch card in slot 7
`mon:ADV:SOUR IMM`	Scan through monitor channels using internal triggering
`SCAN:ADV:SOUR IMM`	Scan through the scan list using internal triggering

`MON:INIT`	Start scanning. If an alarm condition is detected on channels 0 to 3, scan list 1 is executed. If an alarm condition is detected on channels 4 to 7, sequence MYSEQ (not shown) is executed.

ROUTe:MONitor Command Summary

The “ROUTe:MONitor” command is used to define a “monitor scan list”. This feature allows a set of channels to be scanned continuously, looking for measurements that fall outside the limits defined for the channels scanned. The alarm monitoring feature is described in general in the previous paragraph of this SRS.

The format for the “ROUTe:MONitor” command is:

```
[ROUTe:]MONitor <monitor list>
```

Where <monitor list> is a channel list comprised of channels, channel ranges, and path names, separated by commas.

When a single channel is specified, this means that the single channel will be closed when the scan list reaches that element.

When a channel range is specified (e.g. “23:45”), the scan list is treated as though each channel in the range was specified, separated by commas. This means that each channel in the range is closed one at a time.

If a path name is specified, all of the channels in the “close channel” portion of the path definition are closed when the path is closed. All channels in the “open channel” portion of the path definition (if any) are opened when the path is closed. When the path is opened, ONLY the channels in the “close channel” portion of the path are opened.

State names are NOT allowed on the <monitor list>.

Each channel or path on the <monitor list> must be assigned a measurement type via the SENSE command. If a channel or path on the <monitor list> does not have a measurement assigned, an error will be generated when the user attempts to initiate the monitor list (“MONITOR:INIT” command). Error -221, (“Settings conflict; channel X on monitor list does not have measurement assigned”) will be placed in the system error queue if this situation occurs. The channel or path that does not have a measurement assigned will be identified in the

For an example of a monitor, consider the following command:

```
MONitor (@1(1,3,5,11:15),THISPATH,5(15),2(14,9:2))
```

This monitor scan list instructs the System to scan through channels 1, 3, 5, 11, 12, 13, 14, and 15 of the switch card in slot 1, then scan path “THISPATH”, then scan channel 15 of the switch card in slot 5, then scan channels 2, 14, 9, 8, 7, 6, 5, 4, 3, and 2 of the switch card in slot 2. After the monitor scan has completed, it will re-start at channel 1 of card #1.

The “[ROUTe:]MONitor:CLEar” command is used to clear the monitor scan list. After this command has been executed, no monitor scan list occurs. This command may only be executed when the monitor list is not active.

ROUTe:MONitor:ALARm:SOURce Command Summary

The “ROUTe:MONitor:ALARm:SOURce” command is used to associate the scan list or a named sequence with one or more channels

The format for the “ROUTe:MONitor:ALARm:SOURce” command is:

```
[ROUTe:]MONitor:ALARm:SOURce <source> [, <channel list>]
```

Where <source> identifies one of the following destinations when an alarm is detected on any channel in the channel list.:

`NONE`	Don’t execute any scan list or sequence
`SCAN`	Execute the scan list if alarm is detected
`<name>`	Execute the sequence identified in <name> if alarm is detected

ROUTe:MONitor:STORe Command Summary

The “ROUTe:MONitor:STORe” command is used to the alarm monitor scan list into nonvolatile memory for later recall. The format for this command is:

```
[ROUTe:]MONitor:STORe
```

Once the alarm monitor scan list is stored, it may be recalled by using the “[ROUTe:] MONitor:RECall” command. The scan list may be recalled at power-up by using the “[ROUTe:]MONitor:RECall:AUTO” command.

ROUTe:MONitor:RECall Command Summary

The “ROUTe:MONitor:RECall” command is used to recall the alarm monitor scan list from nonvolatile memory for later recall. The format for this command is:

```
[ROUTe:]MONitor:RECall
```

The alarm monitor scan list may be automatically recalled at power-up by using the “[ROUTe:]MONitor:RECall:AUTO” command. The format for this command is:

```
[ROUTe:]MONitor:RECall:AUTO { ON | OFF | 1 | 0 }
```

This command enables (ON or 1) or disables (OFF or 0) the power-up recall of the alarm monitor scan list. The present state of the power-up recall of the alarm monitor scan list may be read using the query:

```
[ROUTe:]MONitor:RECall:AUTO?
```

The query returns “1” if the power-up recall is enabled, or “0” if it is disabled.

ROUTe:MONitor:INITiate Command Summary

The “ROUTe:MONitor:INITiate” command is used to arm the alarm monitor scan list. Once armed, the alarm monitor scan list advances one item (channel, path, or sequence) for each trigger received on the trigger source selected with the “MONitor:ADVance:SOURce” command. If the trigger source is “IMMediate”, then the alarm begins execution immediately.

The format for this command is:

```
[ROUTe:]MONitor:INITiate[:IMMediate]
```

The alarm monitor may be armed at power-up with the “[ROUTe:]MONitor:INITiate:AUTO” command. The format for this command is:

```
[ROUTe:]MONitor:INITiate:AUTO { ON | OFF | 1 | 0 }
```

This command enables (ON or 1) or disables (OFF or 0) the power-up arming of the alarm monitor scan list. The present state of the power-up arming of the alarm monitor scan list may be read using the query:

```
[ROUTe:]MONitor:INITiate:AUTO?
```

The query returns “1” if the power-up arming is enabled, or “0” if it is disabled. Note that the “MONitor:ADVance:SOURce” trigger source for the alarm monitor scan list is selected by the value saved in instrument state 0 (*SAV 0). If the trigger source is “IMMediate”, the alarm monitor scan list begins executing immediately after power-up.

ROUTe:MONitor:ADVance:SOURce Command Summary

The “ROUTe:MONitor:ADVance:SOURce” command is used to select the trigger source that advances each step of the alarm monitor scan list. The general form of this command is

```
[ROUTe:]MONitor:ADVance:SOURce{IMMediate|BUS|EXTernal|TIMer|USER}
```

The possible choices for the scan advance source are:

1. EXTernal – the EXT1 external trigger source is selected. A pulse on the EXT1 trigger input will advance to the next element on the scan list until a single sweep has completed.
2. BUS – the 1830 platform will wait until the “*TRG” command is received to advance to the next entry on the scan list until a single sweep has completed. A group execute trigger (GET) from the GPIB is equivalent to receiving this command.
3. IMMediate – the trigger is implicitly present, a single sweep of the scan list will take place immediately.
4. TIMer – the trigger is generated according to the trigger timer interval. As each timer trigger is accepted, the scan list will advance to the next entry

until a single sweep has completed.

5. USER – the trigger on any of the selected trigger input(s) is selected. Consult the “TRIGger Command Summary” for a description of how to enable or disable various trigger inputs and also how various attributes of the trigger inputs may be set.

At power-up, and after reset, this is set to IMMediate.

The current alarm monitor advance source can be read with the query:

```
[ROUTe:]MONitor:ADVance:SOURce?
```

This query will return one of the following replies:

```
EXT
BUS
IMM
TIM
USER
```

ROUTe:INCLude Command Summary

The “ROUTE:INCLude” command allows the user to associate together each of the channels in the channel list. When any channel within the group of channels is closed, all channels in the group are closed. Likewise, when any is opened, they all are opened.
The format for the “ROUTe:INCLude” command is:

```
[ROUTe:]INCLude <channel list>
```

The <channel list> identifies each channel that should be grouped together. After this command, whenever a close or open operation is applied to any of the channels in the channel list, they are applied to all. This applies whenever they are closed explicitly, via “CLOSE” or “OPEN” commands, or during the execution of a scan list or user-programmed sequence.

Once a channel is placed on an include list, it remains on the include list until it is cleared. It may be cleared by a power-up, or a “*RST” command.

Channel list association is transitive. That is, if the following sequence of commands is executed:

```
INCL (@1(0),2(0))
INCL (@2(0),3(0))
```

Then channels 1(0), 2(0) and 3(0) are all on the same include list.

The following commands may also clear include list association

```
[ROUTe:]INCLude:CLEar <channel list>
[ROUTe:]INCLude:CLEar:ALL
```

The first form of the command is used to clear any association with an include list from each of the channels on the include list.

The second form of the command is used to clear any include list association with any channel throughout the system.

The present include list association can be queried by sending the query:

```
[ROUTe:]INCLude? [ <channel list> ]
```

If the optional channel list is specified with the query, the reply will consist of one include list for each channel in the channel list. Each of the include lists identifies the total set of channels that are included with the channel in the <channel list>. Each of the exclude lists is separated by a comma to indicate that multiple lists are present. If the channel in the <channel list> has no include lists, then the reply associated with it will be "NONE".

So, if channel 1(0) is included with 1(1) and 1(2), channel 1(3) is not on an include list, and channel 2(10) is included with 2(22), then the query

```
INCL? (@1(0:1,3),2(10))
```

returns the reply

```
(@1(0),1(1),1(2)),(@1(0),1(1),1(2)),NONE,(@2(10),2(22))
```

However, if no <channel list> is specified (the query is "INCL?"), the reply will return ALL of the presently defined include lists. Each of the include lists are separated by a single semicolon. So, in the example above, assuming there are no additional exclude relationships defined, the reply would be:

```
(@1(0),1(1),1(2)),(@2(10),2(22))
```

Note that the reply explicitly identifies each channel, rather than using a short-hand like "(@1(0:2))". This format lengthens the reply but makes parsing the reply easier.

If a channel is on an include list with another channel, it cannot be placed on an exclude list with the channel. Similarly, if a channel is already on an exclude list with another, then it cannot be placed on an include list with the other channel.

Also, certain relay plug-in modules have topologies that cannot implement all syntactically correct include lists. For example, if a 1x4 blocking MUX has channels 1(10), 1(11), 1(12), and 1(13), then the command:

```
INCL (@1(10,12))
```

Would be rejected because it is not possible for the module to implement this command. In this case, an error is placed on the SCPI error queue (-221, "Settings conflict").

Include lists may be stored to nonvolatile memory by using the command "ROUTe:INCLude:STORe" command. This command (which has no parameters) is used to store all currently defined include lists to nonvolatile memory.

The include lists may be recalled by using the command "ROUTe:INCLude:RECall". This command recalls all of the include lists that are stored in nonvolatile memory, verifies that they are valid for the current system configuration, and implements the include list if the configuration is valid. If the configuration has changed since the include list is stored, making the include list invalid, then all include lists are cleared and an error is placed in the SCPI error queue.

The include lists may be automatically recalled from nonvolatile memory after system start-up. The command "ROUTe:INCLude:RECall:AUTO" is used to enable or disable the automatic recall of include lists at power-up. The format for this command is:

```
[ROUTe:]INCLude:RECall:AUTO { ON | OFF | 1 | 0 }
```

When the value "ON" or "1" is specified, the include lists will be recalled from nonvolatile memory when the system is powered on. When the value is "OFF" or "0", no include lists will be defined after power-on.

The present state of the power-on recall of include lists may be read using the "[ROUTe:]INCLude:RECall:AUTO?" query.

ROUTe:EXCLude Command Summary

The "ROUTE:EXCLude" command allows the user to define a set of mutually exclusive channels. When any of the channels in the channel list is closed, all of the remaining channels in the channel group are opened. However, when any of the channels are opened, the others are not affected.

The format for the "ROUTe:EXCLude" command is:

```
[ROUTe:]EXCLude
```

The <channel list> identifies each channel that is to be mutually exclusive.

Once an exclude list is defined, it remains until it is cleared. It may be cleared by a power-up, or a "*RST" command.

The following commands may also clear exclude list association

```
[ROUTe:]EXCLude:CLEar <channel list>
[ROUTe:]EXCLude:CLEar:ALL
```

The first form of the command is used to clear any association with an exclude list from each of the channels on the channel list.

The second form of the command is used to clear any exclude list associations with any channel throughout the system.

The present exclude list association can be queried by sending the query:

```
[ROUTe:]EXCLude? [ <channel list> ]
```

The semantics of this query are identical to that of the "[ROUTe]:INCLude?" query. So, if channel 1(0) is excluded from all channels 1(1) through 1(5), and channel 2(10) is excluded from channels 2(11) through 2(13), the query

```
EXCL? (@1(0),2(10))
```

returns the reply

```
(@1(0),1(1),1(2),1(3),1(4),1(5));(@2(10),2(11),2(12),2(13))
```

Note that for each channel in the query, a channel list is returned in the reply. Note also that the reply explicitly identifies each channel, rather than using a short-hand like "(@1(0:5))". This format lengthens the reply but makes parsing the reply easier.

If a channel is on an include list with another channel, it cannot be placed on an exclude list with the channel. Similarly, if a channel is already on an exclude list with another, then it cannot be placed on an include list with the other channel.

Exclude lists may be stored to nonvolatile memory by using the command "ROUTe:EXCLude:STORe" command. This command (which has no parameters) is used to store all currently defined exclude lists to nonvolatile memory.

The exclude lists may be recalled from nonvolatile memory by using the command "ROUTe:EXCLude:RECall". This command recalls all of the exclude lists that are stored in nonvolatile memory, verifies that they are valid for the current system configuration, and implements the exclude lists if the configuration is valid. If the configuration has changed since the exclude list is stored, making the include list invalid, then all exclude lists are cleared and an error is placed in the SCPI error queue.

The exclude lists may be automatically recalled from nonvolatile memory after system start-up. The command "ROUTe:EXCLude:RECall:AUTO" is used to enable or disable the automatic recall of exclude lists at power-up. The format for this command is:

```
[ROUTe:]EXCLude:RECall:AUTO { ON | OFF | 1 | 0 }
```

When the value "ON" or "1" is specified, the exclude lists will be recalled from nonvolatile memory when the system is powered on. When the value is "OFF" or "0", no exclude lists will be defined after power-on.

The present state of the power-on recall of exclude lists may be read using the "[ROUTe:]EXCLude:RECall:AUTO?" query.

ROUTe:PATH Command Summary

The "ROUTe:PATH" commands are used to define paths, save them to nonvolatile memory, recall them from nonvolatile memory, and delete them from memory.

Paths can be used in CLOSE and OPEN commands. They may also be used in specifying scan lists.

The command:

```
[ROUTe:]PATH:DEFine<path name>,<channel list>[,<channel list>]
```

is used to associate a name with one set of channels to operate. This command requires one channel list and accepts a second, optional channel list.

The first channel list specifies the set of channels to operate whenever the path is operated. If the path is closed (explicitly using a "CLOSE" command or implicitly by placing the path on a scan list), then each of the channels in the first channel list are closed. Conversely, if the path is opened, all of the channels in the first channel list are opened.

The second channel list, if included, identifies the set of channels to open when a path is closed.

A <path name> is a sequence of up to 40 characters. The first character must be a letter from “A” to “Z”. Subsequent characters consist of the letters “A” through “Z”, “0” to “9”, and the underscore (“_”). Upper and lower case characters are equivalent for the purpose of recognizing path names (path names are case insensitive).

The query “[ROUTe:]PATH:DEFine?” can be used to read the present definition of a path name. If the Path name is recognized, the query will return either one or two channel lists. The channel lists can be fully expanded, so that if the commands

```
PATH:DEF DEMO999,(@1(0:3),2(1,17:19),(@1(4:6))
PATH:DEF? DEMO999
```

are executed, the reply will be:

```
(@1(0),1(1),1(2),1(3),2(1),2(17),2(18),2(19)),(@1(4),1(5),1(6))
```

The current list of known path names can be read back with the “PATH:CATalog?” query. This query returns a list of path names, with all names returned in upper case characters, separated by commas. Path names may be returned in any order deemed by the designer, they are not required to be returned in alphabetical order. So, if only the three paths named “Demo1”,”UUT_pin1_to_Arb” and ”sixsigma” have been defined, the query would return:

```
DEMO1,UUT_PIN1_TO_ARB,SIXSIGMA
```

If no paths are defined, the literal “NONE” will be returned. This means that “NONE” cannot be used as a path name.

Paths may be stored to nonvolatile memory. The command “PATH:STORe” is used to store all path definitions to nonvolatile memory, while the command “PATH:RECall” is used to recall all path names from nonvolatile memory.

Paths may be automatically recalled at power-up as well. The “PATH:RECall:AUTO” command is used to enable or disable the power-up recall of all paths stored in nonvolatile memory. If the command specifies ON or 1, recall is enabled; if the command specifies OFF or 0, recall is disabled. The present power-on recall state may be read with the “PATH:RECall:AUTO?” query.

ROUTe:CHANnel:DELay Command Summary

The “ROUTe:CHANnel” commands provide the capability to set relay settling time on a per-channel basis. The relay settling times may be set using the following command:

```
[ROUTe:]CHANnel:DELay { <seconds> | MIN | MAX | DEF }, <channel list>
```

The command takes two parameters. The first parameter is the settling time. This may be specified as a number, indicating the time in seconds to allow for a relay to settle when it is opened or closed. Alternatively, the user may specify one of the following keywords:

`MIN` minimum possible settling delay (0 seconds)

`DEF` default settling delay (set to the channel's inherent settling delay)

`MAX` maximum possible settling delay (60 seconds)

The resolution of the settling delay is 10 microseconds (0.00001 seconds).

The second parameter is a channel list. The channel delay value is applied to each channel in the channel list. Note that the DEF (default) value could be different for each channel as it depends on the type of relay being used.

If a path is specified as part of the channel list, each channel that comprises the path has its delay set to the value specified in the command.

At power-up, and after "*RST" reset, the value is DEF.

The current relay settling time can be read by using the query

```
[ROUTe:]CHANnel:DELay? <channel list>
```

The reply to this query is a comma-separated list of floating point values, one per channel in the channel list. For example, the query

```
CHAN:DELAY? (@5(10:13))
```

may return the reply

```
0.002, 0.002, 0.005, 0.002
```

ROUTe:CONFidence Command Summary

The "ROUTe:CONFidence" commands provide two different means of confirming that the relay channels are placed in the proper closed or open states. The user may enable monitoring across all relay channels in the system with the command:

```
[ROUTe:]CONFidence:STATe { ON | OFF | 1 | 0 }
```

When the value specified is ON or 1, the monitor mode is enabled. If it is OFF or 0, the monitor mode is disabled. When the monitor mode is enabled, each time a channel is switched, the embedded software reads back the state of the relay (where hardware permits) and compares the read back state with the programmed state. If the read back state matches the programmed state, no action is taken. If the read back state does NOT match the programmed state, an error is placed on the SCPI error queue. The query "ROUTe: CONFidence:STATe?" may be used to read back whether the monitor mode is enabled or disabled.

ROUTe:VERify? Query Summary

The "ROUTe:VERify?" query allows the user to send a query to read back and compare the state of a set of channels and to report whether the relay channels are in the programmed state. The query

```
[ROUTe:]:VERify? <channel list>
```

can be used to initiate the comparison and report the results of the comparison. The reply to the command consists of a series of "1" or "0", each separated from each other by a single ASCII space character. The reply has one entry for each of the channels in the <channel list>. A value of "1" is returned if the corresponding channel is read back in the proper state, while a value of "0" is returned if the corresponding channel is in the wrong state.

So, the query

```
VERIFY? (@1(0:3))
```

Might return a query of

```
1 0 1 1
```

Indicating that channels 1(0), 1(2), and 1(3) are in the proper state, while channel 1(1) read back an incorrect state.

Is returned. If there are three channels which read back in the incorrect state, the reply might be

```
(@2(0),2(1),7(19))
```

Indicating that channels 0 and 1 on plug-in module #2, and channel 19 on plug-in module 7 are reading back an incorrect state.

Each of the plug-in modules "knows" whether the read-back line is normal, inverted, or cannot be read. This will be part of the information that is uploaded from the plug-in module. However, it is possible for the user to overwrite this setting by using the following command:

```
[ROUTe:]VERify:MASK { NORMal | INVerted | X }, <chan list>
```

This selects whether the read-back verification state is expected to be the same as the state written (NORMal), opposite of the state written (INVerted), or is ignored in the "VERIFY?" comparison (X).

So, the commands

```
VERIFY:MASK INV,(@1(0:13))
VERIFY:MASK NORM,(@2(0:19))
VERIFY:MASK X,(@1(14:20))
```

Select inverted read back for channels 0 through 13 on plug-in module #1, normal read back for channels 0 through 19 on plug-in module 2, and ignore read back for channels 14 through 20 on plug-in module #1. Note that after executing these commands, the query

```
VERIFY? (@1(14:20))
```

ALWAYS returns

```
1 1 1 1 1 1 1
```

because these channels cannot cause a verify failure.

The present setting for the verify masks can be read using the query

```
[ROUTe:] VERify:MASK? <channel list>
```

This query returns a comma-separated list identifying the present setting for the mask for each of the channels in the channel list. So, the query

```
VER:MASK? (@1(0:1),2(0:1),1(14))
```

would return the reply

```
INV,INV,NORM,NORM,X
```

The “ROUTe:VERify:ALL?” query allows the user to send a single query that checks the state of EVERY (supported) channel in the system. This query returns a single reply of “1” if all channels verify correctly. If any channel fails verification, the reply “0” is returned. The user may then read the system event queue (“SYSTem:EVENt?” query) to determine which channels failed to verify properly.

ROUTe:ORDer Command Summary

The relay operating sequence for channels may be set by using the command

```
[ROUTe:]ORDer { MBB | BBM | IMMediate },<module list>
```

can be used to select the order in which the channels are opened or closed while executing on a scan list. If a channel is set to break-before-make mode (BBM), then the channel that is currently closed on the scan list is opened BEFORE the channel is closed. If the channel is set to make-before-break mode (MBB), then the channel will be closed before the currently closed channel is opened. If the mode is set to immediate (IMM), then the order is undefined and could happen in either order.

The command

```
ORDER BBM,1,3,5,7
```

Sets the order of the switch cards in slots 1, 3, 5, and 7 to Break-Before-Make.

Note that certain switching topologies, such as a blocking matrix, may make it impossible for a channel to be set to make-before-break (MBB). In this case, an attempt to set a channel to make-before-break results in an error being placed on the SCPI error queue and the command will be ignored.

The present switching order can be read by using the query

```
[:ROUTe:]ORDer? <module list>
```

A comma-separated reply, with one element for each channel on the channel list, is returned for this reply. The query

```
ORDER? 1,2,3,7,8,9
```

Might return a reply of

```
BBM,BBM,MBB,MBB,IMM,BBM
```

SEQuence Command Summary

The “SEQuence” command allows the user to define one or more named sequences. These sequences will be referred to in this section as user scripts. Each user script can contain one or more of the following operations:

- Operate one or more switches on one or more modules.
- Execute a stimulus, measurement, or configuration SCPI command supported by a plug-in module.
- Call a subroutine, which is another named sequence
- Generate a trigger to that is input to the trigger control block
- Wait for a fixed amount of time
- Repeat a loop a fixed number of times (one loop per sequence, no nested loops)
- Jump unconditionally to a portion of the sequence
- Check if the last reading was above the alarm limit, and branch based on the alarm condition
- Check if the last reading was below its alarm limit, and branch based on the alarm condition
- Set the gain to be used when making subsequent measurements
- Set the offset to be used when making subsequent measurements
- Set the upper alarm limit to be used for comparison when making subsequent measurements
- Set the lower alarm limit to be used for comparison when making subsequent measurements
- Set the units of measurement to be stored along with the measurements for subsequent measurements

The “SEQuence” command summary is shown below.

[:ROUTe]

```
     :SEQuence
          :CATalog?
     :DEFine                     sequence name
          :FIRSt?                sequence name
          :NEXT?                 sequence name
     :STEP                  “commands”
     :END
          :CLEar                          <channel list>
     :SELect?               <channel list>
     :DELete                     sequence name
          :ALL
```

```
:INITiate             sequence name
:STATus?
:RESult?
:STORe                     sequence name
     :DELete               sequence name
          :ALL
:RECall                    sequence name
     :ALL
          :AUTO            { ON | OFF | 1 | 0 }
          :AUTO?
          :POWeron         sequence name
               :AUTO       { ON | OFF | 1 | 0 }
               :AUTO?
```

A sequence is a set of user-defined command that are executed in one of two ways. Once a sequence is defined, it may be executed immediately, via the “SEQuence:INITiate” command, or it may be executed as part of a scan list, by associating sequences to channels with the “SEQuence:SELect” command.

There are both named sequences and unnamed sequences. Unnamed sequences may only be defined using the “SEQuence:SELect” command.

The format of this command when using an unnamed sequence is:

```
SEQuence:SELect “<commands”>,<channel list>
```

For example:

```
SEQ:SEL “SENSE:VOLT:DC 10,0.01,(@11);READ? (@11)”,(@1(0:3))
```

This example shows an unnamed sequence used to set up the measurement device (e.g. DMM) in slot 11 to make a DC voltage measurement when any of the channels 0 to 3 for the switch card in slot #1 is closed during a scan or alarm monitor list.

A named sequence is defined with a set of sequence commands. The first command is the “SEQuence:DEFine” command. This identifies the name of the sequence to be defined.

Following the “SEQuence:DEFine” commands, one or more “SEQuence:STEP” commands identify the command(s) to execute. The sequence definition is concluded when the “SEQuence:END” command is received.

A sequence may be executed directly, by use of the “SEQuence:INITiate” command. This command arms and begins executing the named sequence.

A sequence may also be executed indirectly, while the alarm monitoring list is executing or when a scan list is executing.

SEQuence:CATalog Query

The “SEQuence:CATalog” query is used to read the names of all of the user scripts (programs). The format for this query is:

```
SEQuence:CATalog?
```

This query returns a list of comma-separated user-defined sequence names. Each script that has been defined with the “SEQuence:DEFine” command (or recalled from nonvolatile memory) is returned.

Sequence names are

```
MY_SCRIPT,SCRIPT2,READ_DATA_POINTS,LONG_LIVE_RON_PAUL
```

SEQuence:DEFine Command and Queries

The user MUST send a “SEQuence:DEFine” in order to begin defining the user script. The format for this command is:

```
SEQuence:DEFine <sequence name>
```

The <sequence name> parameter identifies the name of the sequence. Sequence names look very much like path names. They begin with a letter (A-Z), with each following letter a letter (A-Z), number (0-9), or the underscore (_). The minimum sequence name is 1 character, and the maximum sequence name is 40 characters long.

Sequence names may be specified and referenced in upper and lower case, but they are stored in upper case. The <sequence name> specified must be unique. It cannot be the same as a previously defined path name, or any reserved keyword (for example, you cannot have a sequence names “SEQUENCE” or “CLOSE”, since these are keywords). If the <sequence name> has already been defined, the command fails and an error is placed on the SCPI error queue.

The present definition for a sequence can be read back by the user by using two different queries. The first query begins the process of reading the definition of the sequence. The format for this query is:

```
SEQuence:DEFine:FIRSt? <sequence name>
```

This query returns the first line of the user script identified by the sequence name. The reply includes the short-hand version of the command header (“SEQ:STEP”), and the sequence command step is enclosed in double quotes and shifted to upper case characters. So, for example:

```
SEQ:STEP “CLOSE (@1(10,23))”
```

After the first reply is read, the user specifies one or more of the following queries:

```
SEQuence:DEFine:NEXT? <sequence name>
```

After each query is sent, the next line in the user script is returned to the user.

When there are no more steps in the user script to read back, the reply:

```
END OF SEQUENCE <sequence name>
```

SEQuence:SELect Command

The SEQuence:SELect command allows the user to associate a sequence to execute when any channel or path is closed during the execution of the scan list or the alarm monitor list. This command associates a named sequence with the channel(s) and path(s) specified.

The format for this command takes a sequence name as the first parameter, and a channel list as the second parameter:

```
SEQuence:SELect <sequence name>,<channel list>
```

With this command, the sequence identified by <sequence name> must be have been previously defined (with the SEQuence:DEFine statement) before the sequence can be associated with the channel list.

The query form of the command may be used to read back the currently selected sequences for each channel or path on the <channel list>. The format of the query is:

```
SEQuence:SELect? <channel list>
```

The reply to this query identifies the sequence name associated with each channel or path on the channel list. Each sequence name is separated from the next with a comma. If there is no sequence associated with the corresponding channel, the keyword NONE is returned. For example, the query:

```
SEQuence:SELect? (@1(0,2),PATHXYZ,3(4:7))
```

might return:

```
NONE,TESTABC,SEQ999,NONE,EXAMPLESEQ_22,NONE,NONE
```

Indicating there is no sequence associated with channels 1(0), 3(4), 3(6), and 3(7), while the sequence “TESTABC” is associated with channel 1(2), the sequence “SEQ999” is associated with path “PATHXYZ”, and the sequence “EXAMPLESEQ_22” is associated with channel 3(5).

SEQuence:STEP Commands

The SEQuence:STEP command allows the user to define a new step in a sequence. This command may either be a SCPI command that is sent to one (or more) cards to execute, or it may be a flow control type command that affects the execution of the sequence itself.

The command in the “SEQuence:STEP” is added in the order the command is sent. This must follow a “SEQuence:DEFine” command, and the new step is considered part of the sequence that was specified in the “SEQuence:DEFine” command.

The typical form of the sequence command is:

```
SEQuence:STEP “<sequence command step>”
```

Where <script command step> consists of a valid SCPI command that is implemented by one or more cards, or the main embedded software.

Note that only a SINGLE command can be specified inside the "SEQuence:STEP". Multiple commands, separated by semi-colons, are not allowed and will result in a syntax error being placed in the SCPI error queue when the command is parsed. For example, the statement:

```
SEQ:STEP "CLOSE (@1(0)) ; OPEN (@1(3))"
```

is not allowed.

Valid commands will be documented with each plug-in module. For the switch cards, valid commands are "OPEN" and "CLOSE". For the DMM, they consist of configuration commands ("CONF"), sense commands ("SENSE"), read commands ("READ?"), and fetch commands ("FETCH?")

Each plug-in card can support a different command set. The manual for the plug-in card will identify which commands may be executed in a sequence.

A typical DMM plug-in card may, for example, support the following commands:

```
SENSE
      :VOLT
            :DC   [<range> [, <resolution>] ]
            :AC   [<range> [, <resolution>] ]
            :RES  [<range> [, <resolution>] ]
            :FRES [<range> [, <resolution>] ]
READ?
```

While a D/A card may support the following commands

```
SOURce
      :VOLT      <voltage>, <channel list>
OUTPut
      :STATE     {ON|OFF|1|0}, <channel list>
```

In the following example, assume there is a DMM plug-in module with address 2, a D/A module with address 3, and switch cards with addresses 1 and 4. A sample sequence definition might be:
(Note: The sequence numbers shown are for explanation purposes, these numbers are NOT sent as part of the command.)

```
1.    SEQ:DEF EXAMPLESEQ
2.    SEQ:STEP "CLOSE (@4(13,15))"
3.    SEQ:STEP "SENSE:VOLT:DC 10.0,0.001,(@12(0))"
4.    SEQ:STEP "CLOSE (@1(16))"
5.    SEQ:STEP "SET GAIN = 1.025"
6.    SEQ:STEP "SET OFFSET = -0.012"
7.    SEQ:STEP "SET UNITS= VDC"
8.    SEQ:STEP "SET ALARMHI = 3.625"
```

```
9.    SEQ:STEP “SET ALARMLO = 1.877”
10.   SEQ:STEP “SET ALARMHI = ON”
11.   SEQ:STEP “SET ALARMLO = OFF”
12.   SEQ:STEP “READ? (@12(0))”
13.   SEQ:STEP “OPEN (@1(16))”
14.   SEQ:STEP “READ? (@12)”
15.   SEQ:STEP “CALL EXMPLSUB_99”
16.   SEQ:END
17.   SEQ:DEF EXMPLSUB_99
18.   SEQ:STEP “OUTP OFF,(@3(0:2))”
19.   SEQ:STEP “SOURCE:VOLT 1.5,(@3(0))”
20.   SEQ:STEP “SOURCE:VOLT 2.25,(@3(1))”
21.   SEQ:STEP  “SOURCE:VOLT 4.75,(@3(2))”
22.   SEQ:STEP “OUTP ON,(@3(0:2))”
23.   SEQ:END
```

The example above defines two distinct sequences. These are named “EXAMPLESEQ”, in lines (1) through (8), and “EXMPLSUB_99”, in lines (9) through (15). The first sequence, “EXAMPLESEQ”, calls the second sequence.

The first sequence (EXAMPLESEQ) performs the following sequence of operations:

2. Close channels 13 and 15 on the card in slot 4
3. Set up the DMM in slot 12 to make a DC Voltage measurement in the 10V
4. Close channel 16 on the card in slot 1
5. Set the gain to apply to subsequent measurements (e.g. the “READ?” in the 12).
6. Set the offset to apply to subsequent measurements (e.g. the “READ?” in the 12).
7. Set the units of measurement to record in the measurement buffer to “VDC” for subsequent measurements
8. Sets the upper alarm limit to 3.625 for subsequent measurements
9. Sets the lower alarm limit to 1.877 for subsequent measurements
10. Enables the upper alarm limit detection
11. Disables the lower alarm limit detection.
12. Reads the DMM in slot 12. At this point, the measurement will
    (a) Be scaled with a gain of 1.025
    (b) Have an offset of -0.012 added

(c) Be compared to the upper limit of 3.625. The ALARMHI indication will be set in the measurement results buffer and also the next "IF ALARMHI" sequence step will evaluate as true

(d) Store the measurement in the measurement results buffer

13. Open channel 16 on the relay module in slot 1

14. Reads the DMM in slot 12. All of the steps 12 a through 12 d will repeat with the new reading

The second sequence first disables channels 0 through 2 on the DAC module in slot 3 (line 10). It then sets the DAC channel 0 to 1.5 volts (line 11), DAC channel 1 to 2.25 volts (line 12), and DAC channel 2 to 4.75 volts (line 1#). It then turns the output of all three DAC channels on (line 14).

Sequence Name and Label Semantics

Sequence names consist of 1 to 40 characters. They must begin with a letter ("A" to "Z"). The first letter may be followed by one or more characters consisting of the letters ("A" to "Z"), the numbers ("0" to "9") and the underscore ("_"). Upper and lower case characters are equivalent, so letter case is ignored. Sequence names, and labels within sequences, must be unique throughout all defined sequences.

Sequence names must be unique with respect to path names. A single name may NOT be used for a path name and a sequence name.

Sequence labels must begin with the "$" character. The remainder must conform to the rules specified above for a sequence name (that is, 1 to 40 characters, first must be alpha, and so on).

Sequence Flow Control Statements

The sequencer supports the following flow control statements:

```
CALL <sequence name>
RETURN
GOTO <label>
IF <condition> GOTO <label>
```

The "CALL" statement executes another, previously defined sequence.

The "RETURN" statement stops execution of the sequence immediately and returns to the caller (if any), or stops the execution of the sequence if it was not called by another sequence.

The "GOTO" command branches unconditionally to the statement that contains the label.

The "IF <condition> GOTO <label>" will branch to statement that contains the label if the condition evaluates true. If not, control will drop to the next statement.

These commands may be specified inside a “SEQuence:STEP” command.

Sequence Wait Statement

The sequencer supports a wait statement. This statement causes the sequencer to idle for the specified amount of time. The format for this statement is:

```
WAIT <time>
```

This command waits for <time> to elapse before continuing. This is a value in seconds from 10 microseconds (0.00001) to 30 minutes (1800.0), with 10 microsecond resolution.

Sequence Trigger Statement

The sequencer supports a trigger statement. See Figure 6-3 for a Trigger Control Block Diagram. This trigger generates a pulse on the “Trig from SCAN1” line in the trigger control block.

The format for this statement is:

```
TRIGGER
```

The result of generating the trigger on the “Trig from SCAN1” line is determined by the configuration of the trigger control block and the programming of the individual plug-in cards. The action taken by the plug-in cards in response to a trigger is defined by the cards themselves.

Sequence Set Statements

The sequencer supports several SET statements. The gain, offset, units of measurement, alarm limits, and alarm enable condition can be set using the SET statements. The format for these statements is:

```
SET GAIN = <gain>
SET OFFSET = <offset>
SET UNITS = <units>
SET ALARMHI = <alarm lit>
SET ALARMHI = ON
SET ALARMHI = OFF
SET ALARMLO = <alarm limit>
SET ALARMLO = ON
SET ALARMLO = ON
```

SEQuence:INITiate Command

The “SEQuence:INITiate” command is used to select a sequence and begin running it.

The format for this command is:

```
:SEQuence:INITiate <sequence name>
```

The <sequence name> is the name of a sequence that has been previously defined with the “SEQuence:DEFine” command.

SEQuence:STATus? Query

The “SEQuence:STATus?” query returns the status of the sequence. The values returned are:

Value	Status
0	No sequences are defined
1	One or more sequences defined, sequencer idle
2	Sequence presently running
3	Sequence paused at breakpoint
4	Sequence terminated by “ABORT” command
5	Sequence completed

Note that once a sequence is completed, the value of 5 will be returned until a new sequence is defined (which will return a value of 1), all sequences are cleared (which will return a value of 0), or the “SEQuence:INITiate” command is received.

SEQuence:DELete Command

The “SEQuence:DELete” commands are used to delete one sequence, or all sequences, from volatile memory. These commands do not affect sequences that have been stored in nonvolatile memory.

There are two formats for this command:

```
:SEQuence:DELete <sequence name>
:SEQuence:DELete:ALL
```

The first format specifies a particular sequence name. In this case, only the named sequence is deleted. If the sequence name is not recognized, an error is placed on the SCPI error queue.

SEQuence:STORe Commands

The “SEQuence:STORe” command is used to store a previously defined sequence into nonvolatile memory. When this command is executed, the sequence, and every sequence that is CALL-ed from the sequence, is stored to nonvolatile memory.

If there is insufficient memory to store the sequence(s), an error is placed in the SCPI error queue.

The format for this command is:

```
:SEQuence:STORe <sequence name>
```

Another format of the command may be used to remove the sequence(s) from nonvolatile memory. There are two formats for this version of the command:

```
:SEQuence:STORe:DELete <sequence name>
:SEQuence:STORe:DELete:ALL
```

The first variant of the command deletes only the named sequence from nonvolatile memory.

The second variant of the command deletes all previously stored sequences from nonvolatile memory.

SEQuence:RECall Command Summary

The “SEQuence:RECall” commands are used to recall sequences from nonvolatile memory. These commands can also be used to select a particular sequence to recall and execute at power-up.

The “SEQuence:RECall” command tree is shown below

```
:RECall                 <sequence name>
     :ALL
     :AUTO              { ON | OFF | 1 | 0 }
     :AUTO?
     :POWeron           <sequence name> | NONE
          :AUTO         { ON | OFF | 1 | 0 }
          :AUTO?
     :POWeron?
```

The command “SEQuence:RECall:ALL” recalls all store sequences from nonvolatile memory.

The “SEQuence:RECall:AUTO” command determines whether the sequences are all recalled at power on. If this is set to ON (or 1), ALL sequences from nonvolatile memory are automatically recalled at power-up. If this is set to OFF (or 0), no sequences are recalled from nonvolatile memory, with the possible exception of the power-up sequence as described below.

The “SEQuence:POWeron” command identifies the sequence to recall and execute at power-on. This enables the user to run a sequence automatically. Note that the System will require several seconds to boot-up, so this sequence recall and execution process is NOT fast enough to place the unit into a safe state.

The command can be sent with either a sequence name which must already be defined and stored in nonvolatile memory. Alternatively, the command can be sent with the keyword “NONE” to indicate that NO sequence is to be recalled and executed at power-on.

The “SEQuence:POWeron?” query can be used to request the name of the sequence that is recalled at power-on. The query will return the name that is selected for power-on recall, or the keyword “NONE”.

The “SEQuence:POWeron:AUTO” command is used to enable or disable the recall and execution of the power-on sequence. If this is ON (or 1), the power-on sequence will be recalled and executed. If this is OFF (or 0), the power-on sequence will not be recalled or executed. Note that when the “SEQuence:POWeron?” value is set to NONE, this automatically turns this value off. Also, if the power-on sequence is deleted from nonvolatile memory, this will also disable the recall of the power-on sequence. Finally, when the “SEQuence:POWeron” command is set to a sequence name, the power-on recall is automatically enabled.

CALCulate Command Summary

The CALCulate command tree is shown below:

```
:CALCulate
      :SCALe
         :GAIN          <gain> [ , <channel list> ]
         :GAIN?         <channel list>
         :OFFSet        <offset> [ , <channel list> ]
         :OFFSet?       <channel list>
         :STATE         {ON|OFF| 1 | 0 }  [ , <channel list> ]
         :STATE?        <channel list>
         :UNIT          <units>" [ , <channel list> ]
         :UNIT?         <channel list>
      :LIMit
         :LOWer         <threshold> [ , <channel list> ]
            :STATe      {ON|OFF| 1 | 0 } [ , < channel list> ]
            :STATe?     <channel list>
         :LOWer?        <channel list>
         :UPPer         <threshold> [ , <channel list> ]
            :STATe      {ON|OFF| 1 | 0 } [ , < channel list> ]
            :STATe?     <channel list>
         :UPPer?        <channel list>
```

Each of these commands may include an optional channel list parameter. When a channel list is specified, these commands affect the behavior of a scan list. When no channel list is specified, these commands affect the measurements made in immediate command mode.

With a Channel List

When a channel list is present, these commands affect the scan list operation. Each command associates a gain, offset, high and low alarm limits, and units of measure with channels specified in the channel list. When the channel, or path, is closed during the execution of a scan list or alarm monitor list, and subsequent measurements are made from a plug-in module, the gain and offset are applied to measurements made from any plug-in module that is read as part of the sequence associated with that channel via the "SEQuence:SELect" command.

Next, the scaled measurement result is compared against the upper and lower limits assigned for the channel or path. If the scaled reading is higher than the upper limit, or lower than the lower limit, then an alarm condition has been detected. The alarm status for each measurement can be read back along with the measured value. Consult the "FORMat Command Summary" section of this document for a discussion of how to read back the alarm status.

With No Channel List

When no channel list is included, these commands set the gain, offset, and units of measure, and upper and lower alarm limits for all channels in the system. No channel list is identical to specifying a "wild card" that matches all channels in the system.

CALCulate:SCALe Command Summary

A gain and offset can be set for individual channels or for a path as a single entity. The gain will be applied to any measurement made during the execution of a sequence associated with the channel (or path) after the channel (or path) has been closed during a scan list or alarm monitoring scan list. The gain and offset are applied only if the gain and offset are enabled for that channel or path.

The command

```
CALCulate:SCALe:STATe {ON|OFF|1|0} [ , <channel list> ]
```

is used to enable or disable the application of the gain and offset to measurements. If a channel list is present, this command affects the measurements made during the execution of the scan list. If the state is set to ON or 1, the gain and offset for the channel or path is applied to the measurements made for the channel or path during the execution of the scan list. If the state is set to OFF or 0, the gain is effectively set to 1 and the offset is set to 0. When the state is ON, every measurement made during the execution of the sequence following the closure of the channel or path is scaled using the formula

scaled reading = (measurement * gain) + offset

The scaled reading will be what is stored by the system and retrieved subsequently with the "FETCH?" query. If alarm limits are enabled, the scaled reading is used to compare against the limits rather than the raw measurement.

When no channel list is used with the "CALCulate:SCALe:STATe" command, this command enables or disables the application of the gain, as specified with the "CALCulate:SCALe:GAIN" command, and the offset, as specified with the "CALCulate:SCALe:OFFSet" command, with any reading from a plug-in card. If the state is ON or 1, the gain and offset are applied to any measurement made from one of the plug-in cards, and any measurement of the power supply voltages made with the "MEAS?" query.

The command used to set the gain is:

```
CALCulate:SCALe:GAIN { <gain> | MINimum | MAXimum |
DEFault }[, <channel list> ]
```

This command sets the gain for every channel or path in the <channel list> to the value specified in <gain>. The range of values accepted for the <gain> are +/- 1.0E37. The values for the reserved keywords MINimum, MAXimum, and DEFault are shown below:

```
MINimum    -1.0E37
MAXimum    1.0E37
DEFault    1.0
```

For example, the command

```
CALC:SCALE:GAIN 1.5,(@1(0),2(22),MYPATH))
```

Sets the gain for channels 1(0), 2(22), and path MYPATH to 1.5. When these channels or paths are operated as part of a scan list, this gain value will be applied to any measurements made as part of the sequence assigned with the "SEQuence:SELect" command for the channel.

However, the command

```
CALC:SCALE:GAIN 1.5
```

Applies the gain value of 1.5 to all subsequent measurements made in immediate mode, and the command

```
SEQ:STEP:COMMAND "CALC:SCALE:GAIN 1.5"
```

indicates that all subsequent user program steps should apply a gain of 1.5 to any measurements made.

When no <channel list> is supplied, this command instructs the system to set the gain factor for all subsequent measurements from any measurement device. When used as part of a sequence, this command instructs the system to apply the gain factor to measurements made during the execution of the sequence.

The query

```
CALCulate:SCALe:GAIN?  <channel list>
```

can be used to read back the presently set gain or offset for the channels and paths in the <channel list>. The reply to this query is a list of comma-separated values. Note that the programmed gain value will be returned regardless of whether "CALCulate:SCALe:STATE" is on or off. For example, the query

```
CALC:SCALE:GAIN? (@1(0),MYPATH,2(0:2))
```

Will return a total of five values; one for each of four channels, and another for the gain associated with the path named MYPATH. A sample reply is shown below:

```
1.0,2.25,-3.0E+19,2.25,160.0
```

The command to set the offset is:

```
CALCulate:SCALe:OFFSet { <offset> | MINimum |
 MAXimum | DEFault } [ , <channel list> ]
```

This command sets the offset for every channel or path in the <channel list> to the value specified in <offset>. The range of values accepted for the <offset> are +/- 1.0E37. The values for the reserved keywords MINimum, MAXimum, and DEFault are shown below:

```
MINimum    -1.0E37
MAXimum    1.0E37
DEFault    0
```

The query

```
CALCulate:SCALe:OFFSet? [ <channel list> ]
```

can be used to read back the presently set offset for the channels and paths in the <channel list>. As with the gain, the presently programmed offset is returned for the channels in this query regardless of whether “CALCulate:SCALe:STATE” is on or off.

The final command in this group is the “CALCulate:SCALe:UNIT” command. This command is used to set the measurement units string for the channel or path. This measurement units string appears along with the measured value returned in the reply for the “FETCH?” query.

The command format is

```
CALCulate:SCALe:UNIT “<units>” [ , <channel list> ]
```

The <units> is a series of zero to 10 characters, including the characters “A” through “Z”, “a” through “z”, “0” through “9”, and the underscore. The case of the characters is maintained by the system and maintained by the system when reporting measurement results returned for the “FETCH?” query. The value may be enclosed in single (‘) or double (“) quotes. The power-up and reset default value for this is the empty string (“”).

The present value for the measurement units may be requested with the query

```
CALCulate:SCALe:UNIT? [ <channel list> ]
```

This query returns a single unit for each channel or path in the list. If there is more than one channel or path in the <channel list>, each units string in the reply will be separated from the other by a comma. For example, the query:

```
CALC:SCALE:UNIT? (@1(0),MYPATH,2(10,11))
```

might return a reply such as

```
“VDC”,”ohms”,”RPM”,”Widgets”
```

While the query

```
CALC:SCALE:UNIT?
```

might return the reply

```
"ADC"
```

CALCulate:LIMit Command Summary

The "CALCulate:LIMit" commands are used to specify an upper and lower limit value for the measurements. These commands may also be used to enable or disable the comparison of a measured value against the upper or lower limit.

The "CALCulate:LIMit:LOWer" command is used to set the lower limit value. The "CALCulate:LIMit:UPPer" command is sued to set the upper limit value. The formats for these commands are

```
CALCulate:LIMit:LOWer { <limit> | MAX | MIN | DEF}
[, <channel list> ]
```

```
CALCulate:LIMit:UPPer { <limit> | MAX | MIN | DEF }
[, <channel list> ]
```

These commands set the lower or upper limit for each channel or path in the <channel list> to the value specified. The range of values accepted is -1.0E37 to +1.0E37. The keyword MAX may be used to specify 1.0E37. The keyword MIN may be used to specify the value -1.0E37. The default value (DEF) for the lower limit is -1.0E37, while the default value for the upper limit is +1.0E37.

The present value of the limits may be read back by using the queries

```
CALCulate:LIMit:LOWer? [ <channel list> ]
```

```
CALCulate:LIMit:UPPer? [ <channel list> ]
```

These queries will return the presently programmed lower or upper limit for each channel or path on the <channel list>. If the <channel list> contains more than one channel or path, the reply will include commas between the limit values.

For example, the query

```
CALC:LIM:LOWER? (@1(0),MYPATH,2(25))
```

might return the reply

```
-100.0,1.25E+9,1999.99
```

The power-up default value for the lower limit is -1.0E37. The power-up default value for the upper limit is +1.0E37.

The lower limit will not be used for comparison against measured values unless the limit is enabled for the channel. Likewise, the upper limit will not be used for comparison against the measured value unless the upper limit is enabled. At power-up, and after reset, the both the lower and upper limits are disabled. The commands

```
CALCulate:LIMit:LOWer:STATe { ON | OFF | 1 | 0 }
[,<channel list>]
```

```
CALCulate:LIMit:UPPer:STATe { ON | OFF | 1 | 0 }
[,<channel list>]
```

are used to enable or disable the lower and upper limits. When the value is ON or 1, the lower or upper limit is enabled. When the value is OFF or 0, the lower or upper limit is disabled. The following queries may be used to determine the limits are currently enabled for the specified channels or paths:

```
CALCulate:LIMit:LOWer:STATe? [ <channel list> ]

CALCulate:LIMit:UPPer:STATe? [ <channel list> ]
```

The reply includes a single response for each channel or path in the <channel list>. If the channel's limit is enabled, the reply will be 1. If the channel's limit is disabled, the reply will be 0. Multiple replies are separated by commas. For example, the query

```
CALC:LIM:UPPER:STATE? (@MYPATH,1(0),2(22))
```

might return the reply

```
1,0,1
```

While the query

```
CALC:LIM:UPPER:STATE?
```

Would return

```
1
```

If the alarm limit checking is enabled for the immediate mode command execution.

Indicating the upper limit for the path named “MYPATH” is enabled, the upper limit for channel 0 on plug-in module #1 is disabled, and the upper limit for channel 22 on plug-in module #2 is enabled.

When a upper or lower limit is enabled, and the channel or path is closed as part of a scan list, any measurement made within the sequence assigned to the channel or path with the “SEQuence:SELect” command compared against the (enabled) upper and lower limits for that channel or path.

CALCulate Command Example for Scan List Operation

For example, consider the following sequence of commands

```
*RST
SCAN (@1(0:3),MYPATH,2(10:13))
SEQ:DEFINE EXAMPLE_SEQ_1
SEQ:STEP ”READ? (@11)”
SEQ:STEP “READ? (@12)”
SEQ:END
SEQ:SELECT EXAMPLE_SEQ_1,(@1(0:3))
SEQ:SELECT “READ? (@11)”,MYPATH
SEQ:SELECT “READ? (@12)”,2(10:13)
```

```
CALC:SCALE:GAIN 10.0,(@1(0:3))
CALC:SCALE:OFFSET -2.0,(@1(0:3))
CALC:SCALE:GAIN 999.0,MYPATH
CALC:SCALE:OFFSET 48.25,MYPATH
CALC:SCALE:GAIN 1.11E6,2(10:13)
CALC:SCALE:STATE ON,(@1(0:2),MYPATH,2(10))
CALC:LIMIT:LOWER -25.0,(@1(0:2))
CALC:LIMIT:UPPER  75.0,(@1(0:2))
CALC:LIMIT:LOWER:STATE ON,(@1(2))
CALC:LIMIT:UPPER:STATE ON,(@1(0:2))
INIT
```

This sequence first executes the reset command (*RST) to reset and disable all gains and offsets, delete the existing scan list, and reset and disable all alarm limits.

The example then defines the scan list as the following series of channels and paths:

```
1(0),1(1),1(2),MYPATH,2(10),2(11),2(12),2(13)
```

The example then defines a sequence named “EXAMPLE_SEQ_1” that reads the instruments plugged into slots 11 and 12. It then assigns the sequence to channels 0 through 3 of the plug-in card in slot 1.

The example then indicates that whenever path MYPATH is closed during the execution of the scan list, the module in slot 11 should be read.

The example then instructs the 1830 platform to read the module in slot 12 whenever any of the channels 2(10), 2(11), 2(12), or 2(13) are closed.

The example then sets different gain and offset values for the various channels and paths in the scan list. Note that the gain and offset are NOT enabled for channels 2(11), 2(12), or 2(13).

The next lines of the example then set the lower limit for any of the channels 1(0), 1(1), and 1(2) to be -25.0; then the upper limit for these channels is set to 75.0.

The next-to-last two lines enable the lower limit for channel 1(2), and enable the upper limit for channels 1(0), 1(1), and 1(2).

The last line arms the instrument, which means the scan list is loaded and ready to begin execution when the instrument is triggered.

When the instrument is triggered, the following sequence would occur

1. Channel 1(0) is closed.
2. The device in slot #11 is read.
3. The measurement is scaled = (meas x 10.0) – 2.0.
4. The scaled reading is stored in the measurement buffer.
5. The scaled reading is compared against the upper limit 75.0.
6. The scaled reading is NOT compared against the lower limit.

7. The device in slot #12 is read.
8. The measurement is scaled = (meas x 10.0) – 2.0.
9. The scaled reading is stored in the measurement buffer.
10. The scaled reading is compared against the upper limit 75.0.
11. The scaled reading is NOT compared against the lower limit.
12. Channel 1(0) is opened.
13. Channel 1(1) is closed.
14. The device in slot #11 is read.
15. The measurement is scaled = (meas x 10.0) – 2.0.
16. The scaled reading is stored in the measurement buffer.
17. The scaled reading is compared against the upper limit 75.0.
18. The scaled reading is NOT compared against the lower limit.
19. The device in slot #12 is read.
20. The measurement is scaled = (meas x 10.0) – 2.0.
21. The scaled reading is stored in the measurement buffer.
22. The scaled reading is compared against the upper limit 75.0.
23. The scaled reading is NOT compared against the lower limit.
24. Channel 1(1) is opened.
25. Channel 1(2) is closed.
26. The device in slot #11 is read.
27. The measurement is scaled = (meas x 10.0) – 2.0.
28. The scaled reading is stored in the measurement buffer.
29. The scaled reading is compared against the upper limit 75.0.
30. The scaled reading is compared against the lower limit -25.0.
31. The device in slot #12 is read.
32. The measurement is scaled = (meas x 10.0) – 2.0.
33. The scaled reading is stored in the measurement buffer.
34. The scaled reading is compared against the upper limit 75.0.
35. The scaled reading is compared against the lower limit -25.0.
36. Channel 1(2) is opened.
37. Path MYPATH is closed.
38. The device in slot #11 is read.
39. The measurement is scaled = (meas x 999.0) + 48.0.
40. The scaled reading is stored in the measurement buffer.
41. The scaled reading is NOT compared against the upper limit.

42. The scaled reading is NOT compared against the lower limit.
43. Path MYPATH is opened.
44. Channel 2(10) is closed.
45. The device in slot #12 is read.
46. The measurement is scaled = (meas x 1.11E6) + 0.0.
47. The scaled reading is stored in the measurement buffer.
48. The scaled reading is NOT compared against the upper limit.
49. The scaled reading is NOT compared against the lower limit.
50. Channel 2(10) is opened.
51. Channel 2(11) is closed.
52. The device in slot #12 is read.
53. The measurement is not scaled.
54. The measurement is stored in the measurement buffer.
55. Channel 2(11) is opened.
56. Channel 2(12) is closed.
57. The device in slot #12 is read.
58. The measurement is not scaled.
59. The measurement is stored in the measurement buffer.
60. Channel 2(12) is opened.
61. Channel 2(13) is closed.
62. The device in slot #12 is read.
63. The measurement is not scaled.
64. The measurement is stored in the measurement buffer.
65. Channel 2(13) is opened.
66. The scan list completes.

The measurement results are returned when the “FETCH?” query is executed. The various fields of the measurement, such as time stamps, alarm status, and so on, can be individually enabled for reporting with the “FORMAT” commands.

CALCulate Command Example for Immediate Command Execution

When no channel list is specified with the CALCulate commands, any measurement read during the execution of subsequent commands uses the value(s) specified in the CALCulate commands.

For example, consider the following sequence of commands:

1. *RST
2. CONF:VOLT:DC 10.0,0.001,(@11(0))
3. CLOSE (@1(0))
4. READ? (@4(0))
5. CALC:SCALE:GAIN 2.0
6. CALC:SCALE:OFFSET -0.05
7. CALC:SCALE:UNIT “RPM”
8. READ? (@4(0))
9. CALC:SCALE:STATE ON
10. READ? (@4(0))
11. CALC:LIMIT:LOWER 0.88
12. CALC:LIMIT:LOWER:STATE ON
13. CALC:LIMIT:UPPER 1.25
14. CALC:LIMIT:UPPER:STATE ON
15. READ? (@11)

Line 1 resets the system, which disables application of gain, offset, units, and limits and resets these items to their default values.

Line 2 configures the device in slot 11 to make a DC voltage measurement in the 10 volt range with a resolution of 0.001.

Line 3 closes a single relay

Line 4 takes a measurement from the device in slot 11. No gain or offset is applied (since they have not been enabled). No units would be returned from the “READ?” command as they have not been defined.

Line 5 sets the gain to 2.0. Line 6 sets the offset to -0.05. Line 7 marks all subsequent measurements to report the units as “RPM”.

Line 8 takes a measurement from the device in slot 11. No gain or offset is applied (since they still have not been enabled). The “RPM” units would be reported along with the measurement.

Line 9 enables the application of gain and offset for subsequent measurements.

Line 10 takes a measurement from the device in slot 11. The gain and offset are now applied to the measurement before it is reported.

Line 11 sets the lower limit to 0.88. Line 12 enables the lower alarm limit.

Line 13 sets the upper limit to 1.25. Line 14 enables the upper alarm limit.

Line 15 takes a measurement from the device in slot 11. The gain and offset are now applied to the measurement before it is reported. The scaled reading is compared against the lower and upper alarm limits.

SENSe Command Interaction

There are several commands within the SCPI definition that are related to the acquisition of measurements. , FETCh, READ, MEASure, TRIGger, and INITiate

The “SENSe” command allows the user to set up an instrument. The sub-elements of the “SENSe” command depend on the type of instrument that is being configured.

The “INITiate” command instructs the instrument to arm the instrument. When the trigger that is specified by the “TRIGger” commands are received, the instrument makes and records a measurement.

The measurement may be read back using the “FETCh?” query. This query does not actually make a measurement, it merely returns the measurement(s), if any, that have been recorded previously. A measurement may be taken once but reported multiple times with the “FETCh?” query.

The “READ?” query is used to both initiate and fetch a measurement. That is, the “READ?” query is identical to the command sequence:

```
“INIT ; FETCH?”
```

The “MEASure?” query combines the functionality of the “SENSe” command and the “READ?” query. That is, with one “MEASure?” query, the instrument is configured, initiated, and the measurement result is fetched from the measurement buffer.

FETCH Query Summary

The “FETCh?” query is used to retrieve measurement results from the results buffer. The results buffer contains all of the measurements that have been made during the execution of one of the following:

- An immediate command (SENSe and INITiate), when the measurements have been previously been captured but no query has returned the results.
- A scan list, when the channels of the scan list are associated with measurement instruments in the scan (see the “SEQuence:DEFine” and “SEQuence:SELect” command descriptions)
- A user-defined script (sequence), where any command that generates a measurement result is placed in the measurement result buffer.

The measurement results are placed in one of two measurement result buffers. For immediate mode commands, the results are placed in the immediate results buffer. This buffer is read using the "FETCh[:IMMediate]?" query.

For the scan list and user-defined script, the measurements are placed into the program measurement results buffer. This buffer is read using the "FETCh:PROGram?" query.

The reply to the query is a list of measurement results, if any, from the specified result buffer. If no measurement results exist, then there will be no reply to this query.

If measurement results exist, the reply will consist of one or more measurements. Multiple measurements are separated from one another by the semicolon character (";"):

```
<measurement-1> ; <measurement-2> ; …
```

Each <measurement> is reported according to the format defined in the "FORMat" command. The format command allows the user to include various items of information, and also to report time stamps in either relative or absolute terms. Consult the "FORMat Command Summary" in this document for further details.

In order to prevent overloading the output buffer (especially on the GPIB interface), the "FETCH:PROGram?" query supports the ability to specify the first point to read and the number of points to return.

For example, the query:

```
FETCH:PROG? 20,150
```

retrieves 150 data records, beginning at the point number 20 (the oldest data point is point #1).

The first point in the query is "1". If you specify point "0" as the first parameter, a SCPI data range error (-222) is returned.

DATA Command Summary

The DATA commands allow the user to specify how the measurement results are captured during SCAN and SEQUENCE operation. In addition, the DATA command can be used to read the number of data point (records) that have been captured thus far. The DATA command can also be used to delete the oldest points, or all captured points.

The DATA command tree is shown below:

```
:DATA
      :MODE                  ROLLover | FILL
      :POINTs?
      :DELete                <points>
            :ALL
```

The "DATA:MODE" is used to select how the system reacts when there is no more space to capture measurements during a SCAN or SEQUENCE operation. If the "ROLLover" is mode is selected, then the oldest measurements are overwritten with new data. If the “FILL” mode is selected, then the system stops recording measurements when the memory becomes full.

The current data mode may be read using the “DATA:MODE?” query. This will return either “ROLL” or “FILL”.

The number of measurement records presently captured may be read using the “DATA:POINTs?” query. This query returns a number between 0 and 500000 (or more if the 1830 software can accommodate more data points).

The oldest data points may be deleted by using the command “DATA:DELete”. The user may specify a number of points (1 to 500000). If the data mode is “FILL,” the system captures new data points. All of the captured data may be deleted with the command “DATA:DELete:ALL”.

FORMat Command Summary

The FORMat commands allow the user to specify how the measurement results are reported by the system. When the user executes a “FETCH?” query, “READ?” query, “MEAS?” query, or any device-specific command that returns data, the system sends a reply that include the measurement results. There are a variety of possible fields and formats for the reply. The FORMat commands allow the user to select which fields are included in the reply and also how the timestamp is reported.

The FORMat command tree is shown below:

```
:FORMat
     :READing
          :ALARm            { ON | OFF | 1 | 0 }
          :ALARm?
          :CHANnel          { ON | OFF | 1 | 0 }
          :CHANnel?
          :SOURce           { ON | OFF | 1 | 0 }
          :SOURce?
          :TIME             { ON | OFF | 1 | 0 }
               :TYPE        { ABSolute | RELative }
               :TYPE?
          :TIME?
          :UNIT             { ON | OFF | 1 | 0 }
          :UNIT?
```

The general format for readings returned by the “FETCH?” query, “READ?” query, et cetera is:

```
<measured value> <units>, <timestamp>, <channel>,
 <command>, <alarm>
```

Multiple measurements in a reply are separated from each other by using the semicolon (“;”).

The <measured value> can be an integer value, a floating point value, or an exponential value. The general rules for formatting the reply is:

1. If the measured value is an integer, and can be represented by a signed 32-bit integer, it will be returned in integer format.
2. If the measured value is most significant digit is less than 1.0E7 and greater than -1.0E7, and is greater than 1.0E-6 or less than -1.0E-6, then it will be returned in floating point format
3. Otherwise, it will be returned in exponential notation:

```
<mantissa>E<sign>DDD
```

Examples:

```
1.2345000E+015
-1.2345000E-015
9.999999E+307
-9.876543E-307
```

The <units> field is the units designator that has been associated with the channel, in a scan list, or has been made active, in immediate or sequence modes.

The <timestamp> field may be either relative mode, where times are relative to the beginning of the scan list or user program, or absolute mode, where timestamps include the date and time of day. The <timestamp> is reported with 10 microsecond resolution.

The <channel> field indicates the channel or path name that was closed during the execution of the scan list. For measurements returned as the result of an immediate command or as a result of a program (user sequence), this field is not available.

The <command> field holds the SCPI command that made the measurement. This field is enclosed in quotes.

The <alarm> field indicates whether the measured value was within both upper and lower limits (0), was below the lower limit (1), or was above the upper limit (2).

So, with all fields enabled, and the time stamp set to absolute time format, a reply to the “FETCH?” query might be:

```
1250.98 ohms,2007/12/20 13:58:23.00123,1(14),11(0)
```

Which includes the measured value (1250.98), the units string (ohms), the timestamp in absolute format (2007/12/20 13:58:23.00123), the channel closed during the execution of the scan list (1(14)), the measurement channel used to make the measurement (11), and the alarm status (0) indicating the measurement was below the alarm limit for that channel.

Commands within the FORMat command tree are used to individually enable and disable the fields of the reporting. Furthermore, it can be used to select whether the relative time stamp mode or absolute time stamp mode is used.

FORMat:READing:ALARm Command Summary

The alarm status can be enabled or disabled by using the command:

```
FORMat:READing:ALARm { ON | OFF | 1 | 0 }
```

When the value ON or 1 is specified, the alarm status is included in measurement results returned by the “FETCH?”, “READ?”, “MEAS?”, or card-specific query that returns a measurement. When OFF or 0 is specified, the alarm status is not included. The default value for this field is OFF.

The query

```
FORMat:READing:ALARm?
```

may be used to check if the alarm status is presently included in the results.

FORMat:READing:CHANnel Command Summary

The measurement results may include the channel that was closed during the execution of the scan list when the measurement was made. The channel item can be reported for measurements ONLY when the measurements are made as part of a scan list. They are not available for measurements made with immediate mode execution, nor from measurements made with a user program (sequence). This is because, with the exception of the scan list, the channel to report would be ambiguous.

The channel reporting can be enabled or disabled by using the command:

```
FORMat:READing:CHANnel { ON | OFF | 1 | 0 }
```

When the value ON or 1 is specified, the channel that was closed as part of a scan list when the measurement was made is included in measurement results. When OFF or 0 is specified, the channel is not included. The default value for this channel is OFF.
The query

```
FORMat:READing:CHANnel?
```

may be used to check if the channel is presently included in the results.

FORMat:READing:SOURce Command Summary

The measurement results may also include the source channel that was used to generate the measurement. The channel will identify the plug-in module, and may also include the channel on the module if the plug-in module supports measurements from multiple channels.

For example, if the measurement device is a DMM in slot 11, with only a single input, the channel descriptor would be:

```
(@11)
```

However, if the measurement device is an A/D card in slot 13 that has 12 channels, and the measurement is made on channel 10, the channel descriptor would be

```
(@13(10))
```

The source channel reporting can be enabled or disabled by using the command:

```
FORMat:READing:SOURce { ON | OFF | 1 | 0 }
```

When the value ON or 1 is specified, the source channel that made the measurement is included in measurement results. When OFF or 0 is specified, the source channel is not included. The default value for this source channel is OFF.

The query

```
FORMat:READing:CHANnel?
```

may be used to check if the source channel is presently included in the results.

FORMat:READing:TIME Command Summary

The timestamp may be included in the measurement results as well. In addition, the timestamp may be displayed in relative mode, or in absolute mode.

The timestamp is enabled or disabled with the command

```
FORMat:READing:TIME { ON | OFF | 1 | 0 }
```

When the value ON or 1 is specified, the timestamp is included in measurement results. When OFF or 0 is specified, the timestamp is not included. The default value for the timestamp is OFF.

The query

```
FORMat:READing:TIME?
```

may be used to check if the timestamp is presently included in the results.

FORMat:READing:TIME:TYPE Command Summary

The timestamp may be expressed in absolute or relative modes. In absolute mode, the date and time of day are returned (if the timestamp is enabled in the measurement).

The timestamp is enabled or disabled with the command

```
FORMat:READing:TIME:TYPE { ABSolute | RELative }
```

When the value ABS or ABSOLUTE is specified, the timestamp is represented in absolute (date/time of day) format. When the value REL or RELATIVE is specified, the timestamp is represented in relative (delta time from the first measurement) format.

The default after power-up or after executing an “*RST” command is ABSOLUTE.

The query

```
FORMat:READing:TIME:TYPE?
```

may be used to read the presently selected timestamp type. The query will return either “REL” or “ABS”.

MEASure Query Summary

The MEASure command is used to measure one of the power supplies or temperature sensors of the system. The format of the MEASure query for power supply measurement is:

```
MEASure[:VOLT[:DC]]? <power supply channel list>
```

The query returns a single voltage measurement for each power supply measured. Each measurement is separated from the next by a semicolon. The format of the reply is determined by the items that are enabled or disabled. Consult the “FORMat” command summary for a description of measurement result formats returned by queries.

The “MEASure?” query may be used to measure temperature. There are two on-board thermistors within the system. These may be read by using the command:

```
MEASure:TEMPerature? <thermistor channel list>
```

The <thermistor channel list> identifies module 0 to select the mainframe as the “module” to measure. The channels correspond to the thermistor sensor as follows:

Channel	Thermistor
40	CPU Thermistor #1
41	CPU Thermistor #2
42	Backplane Thermistor #1 (XI1)
43	Backplane Thermistor #2 (XI2)
44	Backplane Thermistor #3 (PXI)

The query returns a single temperature measurement for each thermistor. The temperature measurement is returned in degrees C. Each measurement is separated from the next by a semicolon. The format of the reply is determined by the items that are enabled or disabled. Consult the “FORMat” command summary for a description of measurement result formats returned by queries.

Note: The DMM also supports the “MEASure?” query. Consult the DMM-Specific command summary section of this document for a description of the syntax and semantics of the “MEASure?” query supported by the DMM.

INITiate and ABORt Command Summary

The “INITiate” command is used to arm plug-in modules to enable them to make a triggered measurement. In addition, the “INITiate” command is used to begin execution of a scan list.

The format for this command is

```
INITiate[:IMMediate]
```

Once a plug-in module or scan list is armed, it may be disarmed with the "ABORt" command. The "ABORt" command immediately halts the execution of a scan list or user program. In addition, each instrument that is in the "waiting for trigger" state is placed into the idle state.

SYSTem Command Summary

The SYSTem commands are used to query a variety of system parameters. In addition, the commands may be used to set a variety of items that are related to the entire system.

The system command tree, with the exception of the "SYSTem:COMMunicate" subtree, is summarized below. The "SYSTem:COMMunicate" subtree is described in subsequent paragraphs of this document.

```
:SYSTem
     :ERRor?
     :EVENt
          :COUNt?
          :CLEar            <event number>
               :ALL
     :EVENt?                <event number>
     :PREset
     :DATE                  <yyyy> , <mm> , <dd>
     :DATE?
     :TIME                  <hh> , <mm> , <ss.ssssss>
     :TIME?
     :VERSion?
     :CPON                  <module address> | ALL
     :CTYPe?                <module address>
     :MODule
          :CHANnel
               :LIST?       <module address>
          :RELay
               :LIST?       <module address>
     :LOCK
          :REQuest?
          :RELease
          :OWNer?
```

SYSTem:ERRor? Command Summary

Each of the communication interfaces, and each Ethernet session, maintains a separate error queue. The error queue is a list of up to 30 errors. These errors are typically caused while interpreting or executing a command from the interface. When a command is not the cause of an error, the error will be placed on the system event queue instead. Consult the "SYSTem:EVENt?" command tree for a description of these commands.

The format for this query is

```
SYSTem:ERRor?
```

The reply to this command uses the format:

```
<error code>,"<error message>"
```

When the system starts up, or the "*CLS" command is received over an interface, the error queue for that interface is cleared. If the user reads the error queue with the "SYSTem:ERRor?" query, and no errors are present, the reply will be:

```
0,"No error"
```

As each error occurs due to the interpretation or execution of an error, an error is added to the error queue for that interface. When the total number of errors on the error queue reaches 30, and another error is detected, the last (most recent) error is replaced with the error message:

```
-350,"Queue overflow"
```

Each error is removed from the queue in FIFO order. The oldest error is reported first. As each error is read, the error is removed from the error queue.

SYSTem:EVENt Command Summary

There is a single system event queue that tracks errors and events that are not related to commands received over a communication interface. This queue holds events and errors related to:

- Power-on event
- Power-on self-test failures
- Communication failures between the 1830 mainframe and the plug-in cards
- Interrupts received from the plug-in cards, such as the reset input on some switch modules
- VERIFY command failures

The "SYSTem:EVEnt" command sub-tree is shown below:

```
:SYSTem
        :ERRor?
        :EVENt
```

```
          :COUNt?
          :CLEar          <event number>
               :ALL
     :EVENt?              <event number>
```

The system event queue is held in nonvolatile memory, so that the system event queue is NOT cleared when the power is turned off. The only way to remove errors from the system event queue is by using the "SYSTem:EVENT:CLEar" commands.

The system event queue can hold a maximum of 50 errors. Once there are more than 50 errors, the last event on the event queue is replaced with the message:

```
<time stamp> ,50,"Queue overflow"
```

The system event queue may be read by using the following command:

```
SYSTem:EVENt? <event number>
```

Where the <event number> is in the range 1 to 50. The reply to this query uses the format:

```
<time stamp>,<event number>,"<event / error message>"
```

The number of valid events presently in the event queue can be read by using the command:

```
SYSTem:EVENt:COUNt?
```

This query will return a numeric integer value in the range from 0 to 50. A value of 0 indicates that there are no errors or events on the error queue.

All of the errors and events in the system event queue can be cleared with the command:

```
SYSTem:EVENt:CLEar:ALL
```

A particular event / error in the event queue can be cleared by using the command:

```
SYSTem:EVENt:CLEar <event number>
```

The <event number> is in the range 1 to 50. As each event is cleared, the higher-numbered events are "moved down" in the queue so that new events may be stored at the end of the queue, and the count of events is decremented by one. If there is no error at the specified event number, the reply will be:

```
<time stamp>,<event number>,"No event"
```

Each of the replies includes a time stamp. The time stamp uses the format:

```
<year> / <month> / <day>  <hour> : <minute> : <second>
```

Where the <hour> is the hour of the day using a 24 hour clock. An example time stamp is shown below:

```
2009/11/17 21:04:36
```

SYSTem:PRESet Command Summary

The “SYSTem:PRESet” command places the system and all plug-in modules into a predefined state. Each plug-in module provides documentation of what its state is after receipt of the “SYSTem:PRESet” command.

The following are the ONLY changes to the system settings that occur in response to the “SYSTem:PRESet” command:

- Measurement Memory: Cleared
- Plug-in card state: Determined by the card-specific firmware

SYSTem:DATE Command Summary

The “SYSTem:DATE” command allows the user to change the current date used in time stamps. The format of this command is:

```
SYSTem:DATE      <year>,<month>,<day>
```

The <year> is a four-digit number in the range 2008 to 2999. The <month> is a one- or two-digit number in the range 1 through 9 and 12. The <day> is a one- or two-digit number in the range 1 to 31 (and must be valid for the specified <month>, including leap years.

The query “SYSTem:DATE?” can be used to read back the present system date. The format of the reply is:

```
<year>,<month>,<day>
```

SYSTem:TIME Command Summary

The “SYSTem:TIME” command allows the user to change the current date used in time stamps. The format of this command is:

```
SYSTem:TIME      <hour>,<minute>,<second>
```

The <hour> is a one or two-digit number in the range 0 to 23. The <minute> is a one- or two-digit numbers in the range 0 to 59. The <second> is a real number in the range 0.000000 to 59.999999 (microsecond resolution). Note that, due to non-real-time performance, this command can take a variable amount to time to execute, and so the ability to set the clock on the correct microsecond cannot be guaranteed.

The query “SYSTem:TIME?” can be used to read back the present system time. The format of the reply is:

```
<hour>,<minute>,<second>
```

Here, the <second> is reported as a real-number with microsecond resolution.

SYSTem:VERSion? Query Summary

The “SYSTem:VERSion?” query allows the user to query the system for the version of SCPI it supports. The reply to this command shall indicate that version 1999.0 is supported, as shown below:

```
1999.0
```

SYSTem:CPON Command Summary

The “SYSTem:CPON” command allows the user to reset to a factory default state a single plug-in card, or all plug-in cards in the system.

The command accepts a single parameter. This parameter indicates the card number (0 to 14, except 10) to reset, or it accepts the keyword “ALL” to indicate that all (up to) 13 plug-in modules, and the 1830 platform itself, is to be placed into a factory reset state.

The format for this command is thus:

```
SYSTem:CPON     { 0 | 1 | 2 | 3 … 9 | 11 … 14 | ALL }
```

The value of “0” is used to select just the mainframe, while the values 1 through 14 each select the corresponding plug-in module. The command “SYSTem:CPON ALL” is equivalent to “*RST”.

This command does NOT clear the system event queue or the SCPI and IEEE-488.2 status structures.

SYSTem:CTYPe? Query Summary

The “SYSTem:CTYPe?” query returns information about the type of plug-in module and the version of firmware, hardware, and FPGA for the plug-in module.

The format of the query is:

```
SYSTem:CTYPe? <module address>
```

where the <module address> is an integer in the range 0 through 14 (except 10). Module address 0 refers to the mainframe.

The reply to this query is similar to the reply to the “*IDN?” query:

```
EADS NORTH AMERICA TEST AND SERVICES,<model number>,
 <serial number>,<fw rev>-<hw rev>-<FPGA rev>
```

If the <module address> specified is 0, the reply will be similar to that shown below:

```
EADS NORTH AMERICA TEST AND SERVICES, 1830, 123456789,
  9.1-1.4-2.5
```

SYSTem:MODule Command Summary

The “SYSTem:MODule” commands are a set of commands and queries that provide the user with the ability to determine the channels that are supported by the module and the commands that are supported by the module. In addition, there are commands that can be used to associate a name with a plug-in module and use the name in lieu of the module address in those commands that accept a module address.

SYSTem:MODule:CHANnel:LIST? Query Summary

The “SYSTem:MODule:CHANnel:LIST?” query allows the user to query the system to determine what channels are supported by a particular plug-in card, or by the mainframe itself. Note that channels on a plug-in module need not be relay channels. Certain plug-in cards, such as an Arbitrary Waveform Generator or Timer/Counter may have multiple channels that are used in conjunction with, for example, a “SENSe” command to set up the channel.

The format for this command is

```
SYSTem:MODule:CHANnel:LIST? <module address>
```

The <module address> is a number in the range 0 to 14 (except 10).

The reply to this command is a comma-separated list that identifies each of the channel numbers that are supported by the module

The command

```
SYST:MOD:CHAN:LIST? 1
```

Might return the following reply:

```
0, 1, 2, 3, 4, 5, 6, 7, 10, 11, 12, 13, 14, 15, 16, 17
```

Indicating the valid channels for module #1 are values between 0 and 7, or between 10 and 17.

If no channels are supported by the module, the reply to the query will be “NONE”.

SYSTem:MODule:RELay:LIST? Query Summary

The “SYSTem:MODule:RELay:LIST?” query allows the user to query the system to determine what relay channels are supported by a particular plug-in card, or by the mainframe itself. This is similar to the “SYSTem:MODule:CHANnel:LIST?” query, except that this reports only relay channels. Note that every channel reported in this query will be included in the “SYSTem:MODule:CHANnel:LIST?”, but not vice versa.

The format for this command is

```
SYSTem:MODule:RELay:LIST? <module address>
```

The <module address> is a number in the range 0 to 14 (except 10).

The reply to this command is a comma-separated list that identifies each of the channel numbers that are supported by the module

The command

```
SYST:MOD:REL:LIST? 1
```

Might return the following reply:

```
0, 1, 2, 3, 4, 5, 6, 7, 10, 11, 12, 13, 14, 15, 16, 17
```

Indicating the valid relay channels for module #1 are values between 0 and 7, or between 10 and 17.

If no channels are supported by the module, the reply to the query will be "NONE".

SYSTem:COMMunicate Command Summary

The "SYSTem:COMMunicate" commands are a set of commands and queries that provide the user with the ability set and query various communication parameters.

The "SYSTem:COMMunicate" command subtree is shown below.

```
:SYSTem
      :COMMunicate
      ….:ENAble              {ON|OFF|1|0}, {GPIB|USB|LAN|WEB}
      ….:ENAble?             {GPIB|USB|LAN|WEB}
      ….:GPIB
           :ADDRess          <GPIB address>
           :ADDRess?
      ….:LAN
           :AUTOip           { ON | OFF | 1 | 0 }
           :AUTOip?
           :DHCP             { ON | OFF | 1 | 0 }
           :DHCP?
           :DNS              <IP address>
           :DNS?
           :DOMain           "<name>"
           :DOMain?
           :GATEway          <IP address>
           :GATEway?
           :HOSTname         "<name>"
           :HOSTname?
           :IPADdress        <IP address>
           :IPADdress?
           :KEEPalive        { <seconds> | MIN | MAX }
           :KEEPalive?       [ { MIN | MAX } ]
```

```
:MAC?
:SMASk          <mask>
:SMASk?
```

SYSTem:COMMunicate:ENABle Command Summary

The “SYSTem:COMMunicate:ENABle” command allows the user to enable or disable various communication interfaces on the system.

The enable / disable state of the interface is stored in nonvolatile memory. It is maintained even if the power is cycled off and then on.

The format for this command is

```
SYSTem:COMMunicate:ENABle { ON|OFF|1|0 },
{ GPIB|USB|LAN|WEB }
```

If the LAN is interface is disabled, the WEB interface cannot be accessed (even if it is enabled), since the WEB interface lies on top of the LAN interface.

The current enable state of any particular interface may be read with the “SYSTem:COMMunicate:ENABle?” query. The query includes the interface (or class of interface in the case of LAN and WEB), that you are querying.

The reply to this query is either a “1”, meaning the specified interface has a lock on the system, or “0” meaning it does not.

So, for example, if any web page has a lock on the system, the query:

```
SYST:COMM:ENAB? WEB
```

will return “1” if the web page is enabled to communicate with the instrument.

SYSTem:COMMunicate:GPIB:ADDRess Command Summary

The current setting for the GPIB address is set by using the “SYSTem:COMMunicate:GPIB:ADDRess” command. This command takes an integer parameter and sets the GPIB address for the box to this number. The GPIB address value must be0 between 0 and 30 inclusive.

The present GPIB address may be read by using the “SYSTem:COMMunicate:GPIB:ADDRess?” query. This query returns an integer value in the range “0” to “31”.

SYSTem:COMMunicate:LAN Command Summary

Various attributes of the LAN interface may be configured or query using the SYSTem:COMMunicate:LAN commands. The command sub-tree for this command is shown below.

```
:SYSTem
    :COMMunicate
```

```
:LAN
   :DNS?
   :DOMain          "<name>"
   :DOMain?
   :GATEway?
   :HOSTname        "<name>"
   :HOSTname?
   :IPADdress?
   :KEEPalive       {<seconds>|MIN|MAX|DEFault}
   :KEEPalive?
   :MAC?
   :PING            { ON | OFF | 1 | 0 }
   :PING?
   :SMASk?
   :MODe            { AUTO | MANual }
   :MODe?
   :MANual
        :IPADress   <IP address>
        :IPADress?
        :DNS    <IP address>[,<IP address>[,<IP address>]]
        :DNS?
        :SMAsk       <mask>
        :SMAsk?
        :GATEway     <IP address>
        :GATEway?
```

Note that each of these settings is stored in nonvolatile memory. The settings do not change when the power is cycled, nor when the "*RST" or "SYSTem:PRESet" commands are received.

SYSTem:COMMunicate:LAN:DNS Command Summary

The current DNS IP address(es) can be queried with the "SYSTem:COMMunicate:LAN:DNS?" query. The query will return up to 3 four-byte IP addresses that are comma delimited, for example:

```
192.88.44.137 or 192.88.44.137,192.16.0.1
```

If "NONE" is returned, then that indicates there are currently no DNS IP addresses being used.

SYSTem:COMMunicate:LAN:DOMain Command Summary

The "SYSTem:COMMunicate:LAN:DOMain" command assigns the Domain

Name to the 1830. The format of this command is:

```
SYSTem:COMMunicate:LAN:DOMain "<name>"
```

The <name> is enclosed in quotes, and is a name from 1 to 63 characters. The first character of the name must be a letter in the range "A" to "Z". The remaining characters can be letters, numbers, dashes ("-"), or periods. Blank spaces are not allowed. For example:

```
SYST:COMM:LAN:DOM "www.my1830-works.com"
```

If this setting is changed with this command, the power to the system must be cycled for the change to take effect.

The current domain name can be queried with the "SYSTem:COMMunicate:LAN:DOMain?" query. The query returns the presently set domain name. For example:

```
www.my1830-works.com
```

Note that if DHCP is enabled, the DHCP server can change the specified domain name. In this case, the value set will not be the value returned. Also, if no domain name has been set, the null-string (two quotes) are returned:

```
NONE
```

SYSTem:COMMunicate:LAN:GATEway? Command Summary

The current gateway IP address can be queried with the "SYSTem:COMMunicate:LAN:GATEway?" query.

The query will return "NONE" if a gateway is not used or a four-byte IP address, for example:

```
255.255.44.137
```

SYSTem:COMMunicate:LAN:HOSTname Command Summary

The "SYSTem:COMMunicate:LAN:HOSTname" command assigns a host name to the 1830. The format of this command is:

```
SYSTem:COMMunicate:LAN:HOSTname "<name>"
```

The <name> is a string of from 1 to 32 characters. The first character must be a letter (A-Z). The remaining characters may be letters, numbers, or dashes ("-"). Blank spaces are not allowed. For example:

```
SYST:COMM:LAN:HOSTname "1830-Yeah"
```

If DHCP is enabled, the DHCP server may change the host name for the instrument.

The current host name can be queried with the "SYSTem:COMMunicate:LAN:HOSTname?" query. The query will return a quoted string, for example:

```
"1830-Yeah"
```

If no host name is assigned, the query will return a null-string, enclosed in quotes:

```
NONE
```

SYSTem:COMMunicate:LAN:IPADdress? Command Summary

The current IP address can be queried with the "SYSTem:COMMunicate:LAN:IPADdress?" query.

The query will return a four-byte IP address, for example:

```
192.88.56.244
```

SYSTem:COMMunicate:LAN:KEEPalive Command Summary

The "SYSTem:COMMunicate:LAN:KEEPalive" command sets the LAN keepalive timeout, which specifies a number of seconds to keep a LAN socket active. If there has been no activity on the connection after the specified timeout, the instrument will send keepalive probes to the client to determine if it is still available. After the specified timeout, the connection will be marked as "down" or "dropped".

The format of this command is:

```
SYSTem:COMMunicate:LAN:KEEPalive { <seconds> |
 MINimum | MAXimum  | DEFault }
```

The <seconds> ranges from 600 (MIN) to 10E6 (MAX), or 0. A value of 0 indicates no time-out, so no keepalive probes will be sent. The value 3600 will be used as a default value.

The current keepalive timeout can be queried with the "SYSTem:COMMunicate:LAN:KEEPalive?" query. The query will return the presently programmed keep-alive value.

SYSTem:COMMunicate:LAN:MAC? Query Summary

The "SYSTem:COMMunicate:LAN:MAC?" query reads the Media Access Control (MAC) address. The format of this query is:

```
SYSTem:COMMunicate:LAN:MAC?
```

The query returns a set of six hexadecimal two-digit numbers, each of which is separated by a single dash, within double quotes. For example:

```
"1D-42-5A-36-0B-6C"
```

The MAC address will be set by the manufacturer of the Ethernet interface chip and cannot be changed.

SYSTem:COMMunicate:LAN:MANual:DNS Command Summary

The "SYSTem:COMMunicate:LAN:MANual:DNS" command assigns the IP address of the Domain Name System (DNS) server. The address(es) is only used if the LAN mode is set to manual. The format of this is:

SYSTem:COMMunicate:LAN:MANUal:DNS { NONE | <IP address>[,<IP address>[,<IP address>]] }

The <IP address> is expressed in four-byte format. Each byte is in the range 0 to 255.

```
192.16.11.1
```

If you want to use more than one DNS server, you can send a comma delimited string of the IP addresses:

```
192.16.11.1,192.16.11.0
```

Furthermore, if there's no need to use a DNS server, you can send "NONE" as the parameter to the command.

The current IP address for the DNS server can be queried with the "SYSTem:COMMunicate:LAN:MANual:DNS?" query. The query will return "NONE" if no DNS servers are used, or a comma delimited string of IP addresses:

SYSTem:COMMunicate:LAN:MANual:GATEway Command Summary

The "SYSTem:COMMunicate:LAN:MANual:GATEway" command assigns the default gateway for the 1830. The format of this command is:

```
SYSTem:COMMunicate:LAN:MANual:GATEway {NONE|<IP address>}
```

The <IP address> is expressed in four-byte format. Each byte is in the range 0 to 255. For example:

```
SYST:COMM:LAN:MANual:GATEway 255.255.44.137
```

If DHCP is active, and a server is found, the default gateway is not used. However, if no server is found, the gateway will be used.

The default value is "NONE". This indicates that no default gateway is used. If this setting is changed with this command, the power to the 1830 system must be power-cycled or the "SYSTem:COMMunicate:LAN:RESET" command must be issued for the change to take effect.

The current Gateway IP address can be queried with the "SYSTem:COMMunicate:LAN:MANual:GATEway?" query. The query will return "NONE" if a default Gateway is not used, or a four-byte IP address, for example:

```
255.255.44.137
```

SYSTem:COMMunicate:LAN:MANual:IPADdress Command Summary

The "SYSTem:COMMunicate:LAN:MANual:IPADdress" command assigns the static IP address to the 1830. The format for this command is:

```
SYSTem:COMMunicate:LAN:MANual:IPADdress <IP address>
```

The <IP address> is expressed in four-byte format. Each byte is in the range 0 to 255. For example:

```
SYST:COMM:LAN:MANual:IPADdress 192.88.56.244
```

If this setting is changed with this command, the power to the 1830 system must be cycled or the “SYSTem:COMMunicate:LAN:RESET” command must be issued for the change to take effect

The current IP address used for manual IP configuration can be queried with the “SYSTem:COMMunicate:LAN:MANual:IPADdress?” query. The query will return a four-byte IP address, for example:

```
192.88.56.244
```

Note that the IP address returned does not reflect the actual IP address used by the system.

SYSTem:COMMunicate:LAN:MANual:SMASk Command Summary

The “SYSTem:COMMunicate:LAN:MANual:SMASk” command assigns the subnet mask to be used in manual mode to the 1830. The format of this command is:

```
SYSTem:COMMunicate:LAN:MANual:SMASk <IP address>
```

The <IP address> is expressed in four-byte format. Each byte is in the range 0 to 255 (typically each will be either 255 or 0). For example:

```
SYST:COMM:LAN:MANual:SMASK 255.255.255.0
```

If this setting is changed with this command, the power to the 1830 system must be cycled or the “SYSTem:COMMunicate:LAN:RESET” command must be issued for the change to take effect.

The current subnet mask used for manual IP configuration can be queried with the “SYSTem:COMMunicate:LAN:MANual:SMASk?” query. The query will return a four-byte IP address, for example:

```
255.255.255.0
```

SYSTem:COMMunicate:LAN:MODe Command Summary

The “SYSTem:COMMunicate:LAN:MODe” command sets the IP configuration mode.

The format of this command is:

```
SYSTem:COMMunicate:LAN:MODe { AUTO | MANual }
```

Once the mode has been set, it will be stored in non-volatile memory. The new LAN mode will not be in effect until the next time the system restarts or when the “SYSTem:COMMunicate:LAN:RESET” command is sent.

The current LAN mode can be queried with the

"SYSTem:COMMunicate:LAN:MODe?" query. The query will return "AUTO" or "MAN".

SYSTem:COMMunicate:LAN:PING Command Summary

The "SYSTem:COMMunicate:LAN:PING" command enables or disables Ping on the 1830. If Ping is enabled, the 1830 will be able to respond to ICMP messages.

The format of this command is:

```
SYSTem:COMMunicate:LAN:PING { ON | OFF | 1 | 0 }
```

The Ping support of the 1830 can be queried by issuing the "SYSTem:COMMunicate:LAN:PING?" query. The query will return "1" if Ping is enabled, and "0" if Ping is disabled.

SYSTem:COMMunicate:LAN:RESET Command Summary

The "SYSTem:COMMunicate:LAN:RESET" resets the LAN interface of the 1830. The LAN reset will put in effect whatever changes were made to the LAN mode and LAN Manual parameters.

For example, to change the current IP address from a DHCP assigned address to a static address, you would set the static IP address and subnet mask, and other optional parameters by using the commands under "SYSTem:COMMunicate:LAN:MANual". Then set the LAN mode to manual by issuing "SYSTem:COMMunicate:LAN:MODe MANUAL". Then to put the new settings in effect, issue this LAN reset command.

SYSTem:COMMunicate:LAN:SMASk? Command Summary

The current subnet mask can be queried with the "SYSTem:COMMunicate:LAN:SMASk?" query.

The query will return a four-byte IP address, for example:

```
255.255.255.0
```

DIAGnostic Command Summary

The "DIAGnostic" commands are used to read the cycle counters for relays on the plug-in cards. They are also used to read the cycle counters for the analog bus relays on the mainframe.

The command tree for this command is shown below.

```
DIAGnostic
    :RELay
        :CYCLes?        <channel list>
        :CYCLes
```

```
:CLEar  <channel>,"<password>"
```

The "DIAGnostic:RELay:CYCLes?" query is used to read the number of times the specified relay(s) have changed from the open state to the closed state. The reply returns the cycle count for each channel specified in the <channel list>. Each count is separated by a comma.

For example, the query:

```
DIAG:RELAY:CYCLES? (@1(14),3(15),2(11))
```

Might return the reply:

```
1450,20679,3711
```

Meaning that channel 14 on plug-in card #1 had a cycle count of 1450, channel 15 on plug-in card #3 had a cycle count of 20679, and channel 11 on plug-in card #2 had a channel count of 3711.

TRIGger Command Summary

The TRIGger commands are used to select trigger inputs and outputs. Trigger inputs are used to allow external events to cause a measurement, stimulus, or switch action by one or more plug-in cards.

Trigger outputs are used to allow the system to generate a signal to synchronize measurements with other instruments.

The TRIGger command tree is shown below:

```
:TRIGger
      :SOURce               {IMMediate|BUS|EXTernal|TIMer|USER}
      :SOURce?
      :TIMer                { <timer interval> | MIN | MAX }
      :TIMer?               [ { MIN | MAX } ]
      :COUNt                { <trigger count> | MIN | MAX }
      :COUNt?               [ { MIN | MAX } ]
      :MODule1
      :MODule1
            :SOURce         <trigger input>
            :SOURce?
            :DELay          { <seconds> | MIN | MAX }
            :DELay?         [ { MIN | MAX } ]
            :LEVel*         { HIGH | LOW }
            :LEVel?
            :SLOPe*         { POSitive | NEGative }
            :SLOPe?
            :SENSitivity*   { EDGE | LEVEL }
            :SENSitivity?
            :ENABle         { ON | OFF | 1 | 0 }
            :ENABle?
```

```
                :WIDTh           { <seconds> | MIN | MAX }
                :WIDTh?          [ { MIN | MAX } ]
          :MODule2         (repeat command tree for MODule1)
          :MODule3         (repeat command tree for MODule1)
                .                              .
                .                              .
                .                              .
          :MODule9         (repeat command tree for MODule1)
          :MODule11        (repeat command tree for MODule1)
          :MODule12        (repeat command tree for MODule1)
          :MODule13        (repeat command tree for MODule1)
          :MODule14        (repeat command tree for MODule1)
```

Note: <trigger input>=NONE|INPut1|INPut2|...|INPut12|EXT1|EXT2|
SCAN|ALARMLO|ALARMHI

```
:TRIGger
          :EXTernal2       (repeat command tree for MODule1, except*)
          :SCAN            (repeat command tree for MODule1, except*)
          :INPut1
                :ENABle          { ON | OFF | 1 | 0 }
                :ENABle?
                :LEVel           { HIGH | LOW }
                :LEVel?
                :SLOPe           { POSitive | NEGative }
                :SLOPe?
                :SENSitivity     { EDGE | LEVEL }
                :SENSitivity?
          : INPut2               (repeat command tree for INPut1)
          : INPut3               (repeat command tree for INPut1)
                .                                    .
                .                                    .
                .                                    .
          : INPut9               (repeat command tree for INPut1)
          : INPut11              (repeat command tree for INPut1)
          : INPut12              (repeat command tree for INPut1)
          : INPut13              (repeat command tree for INPut1)
          : INPut14              (repeat command tree for INPut1)
```

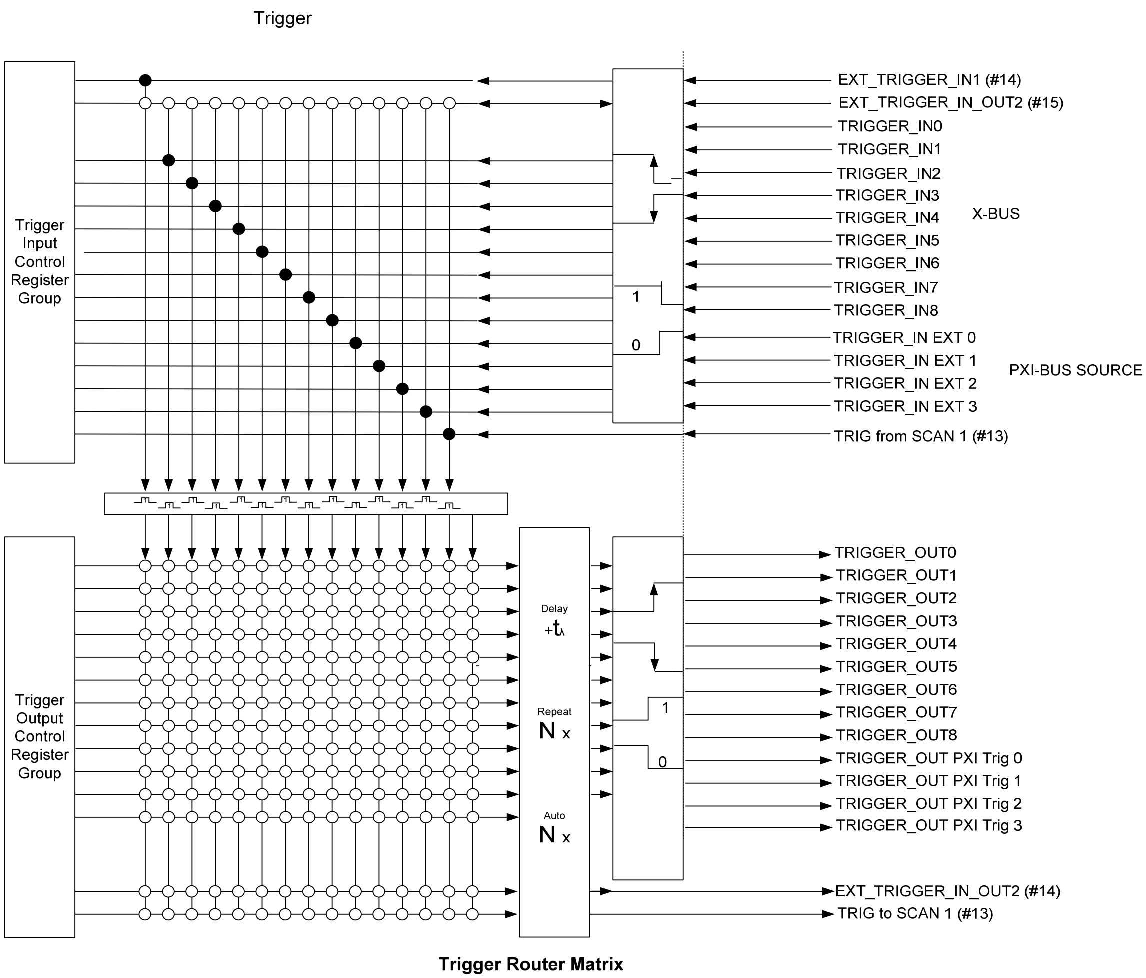


Figure 7-3: Trigger Control Block

All switching, measurement, or stimulus actions will not commence until the system is armed and triggered. The system is armed when the "INITiate" command is received (or, if a user sequence is defined, the "SEQ:INIT" command). The immediate "READ?" and "MEASure?" queries arm and trigger the system as well.

Once the system is armed, a trigger is required to cause the switch, measurement, or stimulus action. A trigger must be received on the currently active trigger source. Only one trigger source is active for the instrument at any given time. The trigger source must be one of the following:

- An internal trigger. This means that the measurement, stimulus, or switch operation is initiated immediately after the system is armed. The system does not wait for an additional trigger signal.
- A bus trigger. This means that the measurement, stimulus, and/or switch operation will occur when the "*TRG" trigger (or, in the case of the GPIB interface, the Group Execute Trigger) is received.
- An external trigger. A pulse on the EXT1 trigger input causes the trigger

to occur.

- A timer. The timer repeatedly fires to generate at a fixed interval to generate the trigger pulse.
- A user-defined trigger source. This is a more general case of the "EXTernal" trigger source. Here, a pulse received on any of the trigger sources selected for the SCAN1 output cause the switch, measurement, or stimulus action to occur.

The "TRIGger:SOURce" command is used to select which of the four possible trigger sources is active. When the trigger source is set to "USER", the trigger control block settings become significant. If the trigger source is anything other than "USER" or "EXTernal", the trigger control block outputs are not used by the sequencer engine. However, regardless of the trigger source, the trigger control block can still be used to route trigger inputs to the trigger line of the plug-in modules.

The trigger control block may be used to route any combination of trigger block inputs to the trigger line of a plug in module. The plug-in card trigger inputs are controlled using the commands "TRIGger:OUTput1" for plug-in card #1, "TRIGger:OUTPut2" for plug-in card #2, and so on up to "TRIGger:OUTPut9" for the 9 X-bus plug-in modules. "TRIGger:OUTPut10" through "Trigger:OUTput13" reference the PXI bus TRIG0 through TRIG3 lines respectively. Various attributes such as the sensitivity, slope, repeat count, delay, and pulse width generated on the trigger lines may be selected as well. Each of the plug-in module's trigger line attributes are fully independent of the attributes set for the other plug-in modules.

The trigger control block may be used to route any combination of trigger block inputs to the external trigger out connector on the system. The "TRIGger:EXTernal2" commands are used to select the trigger inputs, sensitivity, slope, repeat count, delay, and pulse width for this signal.

The trigger control block may also be used to route any combination of trigger block inputs into the sequencer engine.

Each of the trigger block input attributes may be set as well. The "TRIGger:INPut1" commands select the attributes for the trigger input from card #1. Similarly, "TRIGger:INPut2" commands select the attributes for the trigger input from card #2, and so on.

TRIGger:SOURce Command Summary

The "TRIGger:SOURce" command determines which trigger source is selected for the execution of immediate commands as well as scan lists and user-defined scripts (programs). The format for this command is

```
TRIGger:SOURce {IMMediate|BUS|EXTernal|TIMer|USER}
```

When the trigger source is "IMMediate", the immediate commands, scan lists, and user scripts will execute as soon as they are armed. There is no additional trigger required to execute the command, scan list, or user script.

When the trigger source is "BUS", the immediate command, scan list, or user script will wait until a “*TRG” command is received over any communication interface, or until a GPIB Group Execute Trigger (GET) is received.

When the trigger source is “EXTernal”, the output of the trigger control block (see Figure 6-3), the selected trigger inputs for the SCAN1 engine are used as the trigger source. When the “EXTernal” trigger is selected, those inputs and conditions enabled by the “TRIGger:SCAN” command are used as the trigger source for the command, scan list, or sequence.

When the trigger source is “TIMer”, the trigger source for the immediate command, scan list, or user script is a timer that generates repeated trigger pulses. The period of the timer is set with the “TRIGger:TIMer” command.

The present trigger source may be read by using the “TRIGger:SOURce?” query. This query returns the presently selected trigger source, in upper case, short form. So the reply will be one of the following:

```
IMM
BUS
EXT
TIM
USER
```

Note that setting the trigger source to “EXTernal” is equivalent to selecting the EXT1 signal input as the trigger source for the SCAN trigger input. In effect, setting the trigger source to “EXTernal” is equivalent to executing the following commands:

```
TRIG:SCAN:SOUR EXT1
```

TRIGger:TIMer Command Summary

The “TRIGger:TIMer” command determines the period at which the timed triggers are generated. When the “TRIGger:SOURce” is “TIMer”, triggers are routed to the SCAN and sequencer input. This means that scan lists and sequences are triggered repeatedly by the timer trigger.

The format for the “TRIGger:TIMer” command is

```
TRIGger:TIMer { <time interval> | MIN | MAX }
```

The <time interval> is expressed in seconds, with a resolution of 1 millisecond (0.001 seconds). The timer interval range is between 1 milliseconds and 65.535 seconds. As an alternate to specifying the number of seconds, the user may specify either MIN (= 0.001) or MAX (= 65.535).

The present trigger time interval may be read by using the “TRIGger:TIMer?” query. This query returns the present trigger time interval value, in seconds. The reply is formatted as a floating point number (not exponential). A typical reply is

```
0.05
```

The user may also use the queries:

```
TRIGger:TIMer? MIN
```

```
TRIGger:TIMer? MAX
```

These queries return the minimum possible trigger timer interval (0.001) and maximum timer interval (65.535) respectively.

After a power-up, or receipt of the “*RST” command, the trigger time interval is set to 1.0 seconds.

TRIGger:COUNt Command Summary

The “TRIGger:COUNt” command determines the number of times the trigger is accepted before the system returns to the IDLE trigger state. After the number of triggers equal to the trigger count has been received and acted upon, the system must be initiated again (via the “INITiate” command) before triggering again.

The format for the “TRIGger:COUNt” command is

```
TRIGger:COUNt { <trigger count> | MIN | MAX }
```

The <trigger count> range is between 1 and 2000000000. As an alternative to a numeric trigger count, the user may specify MIN, which is equal to 1, or MAX, which is equal to 2000000000.

The present trigger count may be read by using the “TRIGger:COUNt?” query. This query returns the present trigger time interval value, in seconds. The reply is formatted as an integer value. A typical response might be:

```
792342
```

The user may also use the queries:

```
TRIGger:COUNt? MIN
TRIGger:COUNt? MAX
```

These queries return the minimum possible trigger count (1) and maximum possible trigger count (2000000000) respectively.

After a power-up, or receipt of the “*RST” command, the trigger count is set to 1.

TRIGger:MODule<N> Command Summary

The “TRIGger:MODule<N>” commands provide a fine degree of control on the trigger control block than the basic “TRIGger:SOURce” commands. These commands allow the user to select one or more trigger sources, edge or level sensitivity, trigger pulse width, number of output pulses generated per input trigger, and time interval between multiple output pulses. This command tree provides full access to the programmable features provided by the trigger control block in Figure 7-3.

The basic command trees consist of the following top-level elements. The elements below are repeated across the top-level elements. The top-level elements are:

`TRIGger:MODule1` - selects trigger applied to plug-in #1

`TRIGger:MODule2` - selects trigger applied to plug-in #2

`TRIGger:MODule3` - selects trigger applied to plug-in #3

`TRIGger:MODule4` - selects trigger applied to plug-in #4

`TRIGger:MODule5` - selects trigger applied to plug-in #5

`TRIGger:MODule6` - selects trigger applied to plug-in #6

`TRIGger:MODule7` - selects trigger applied to plug-in #7

`TRIGger:MODule8` - selects trigger applied to plug-in #8

`TRIGger:MODule9` - selects trigger applied to plug-in #9

`TRIGger:MODule11` - selects trigger routed to the PXI_TRIG0 trigger bus line

`TRIGger:MODule12` - selects trigger routed to the PXI_TRIG1 trigger bus line

`TRIGger:MODule13` - selects trigger routed to the PXI_TRIG2 trigger bus line

`TRIGger:MODule14` - selects trigger routed to the PXI_TRIG3 trigger bus line

`TRIGger:EXTernal2` - selects trigger applied to EXT2 output

`TRIGger:SCAN` - selects trigger applied to scan / sequencer engine

Each of these top-level commands shares a common command sub-tree to program the various attributes of the trigger pulse(s) that is applied to these trigger lines.

The individual commands are described in detail in the following paragraphs.

TRIGger:MODule<N>:SOURce:ENABle Command Summary

The "TRIGger:MODule<N>:SOURce:ENABle" command is used to select and de-select input sources that are used to generate a trigger on the specified trigger line.

The format for these commands is:

```
TRIGger: MODule<N>:SOURce <trigger input>
TRIGger:EXTernal2:SOURce <trigger input>
TRIGger:SCAN:SOURce <trigger input>
```

The <trigger input> selects one of the possible trigger inputs to the trigger control block as shown in Figure 7-3. The trigger inputs consist of:

`NONE` - deselects all triggers

`INPut1` - selects trigger input from plug-in #1

`INPut2`	- selects trigger input from plug-in #2
`INPut3`	- selects trigger input from plug-in #3
`INPut4`	- selects trigger input from plug-in #4
`INPut5`	- selects trigger input from plug-in #5
`INPut6`	- selects trigger input from plug-in #6
`INPut7`	- selects trigger input from plug-in #7
`INPut8`	- selects trigger input from plug-in #8
`INPut9`	- selects trigger input from plug-in #9
`INPut11`	- selects trigger input from PXI_TRIG0 trigger line
`INPut12`	- selects trigger input from PXI_TRIG1 trigger line
`INPut13`	- selects trigger input from PXI_TRIG2 trigger line
`INPut14`	- selects trigger input from PXI_TRIG3 trigger line
`EXTernal1`	- selects trigger input from the EXT1 input
`EXTernal2`	- selects trigger input from the EXT2 input
`SCAN`	- selects trigger scan / sequencer engine
`TIMER`	- selects trigger timer input

`ALARMLO` - selects trigger when the present measurement exceeds upper alarm limit for the scan list channel or path

`ALARMHI` - selects trigger when the current measurement exceeds lower alarm limit for the scan list channel

At most one trigger source may be selected at one time. If the user specifies a trigger source, then all other trigger inputs are disabled. To select no trigger source, the “NONE” parameter is used:

```
TRIG:MOD7:SOUR NONE
```

The query form of this command allows the user to determine if specific trigger sources are enabled. The query form of this command is:

```
TRIGger:MODule<N>:SOURce?
```

The reply to this query indicates which trigger source is currently enabled. If the reply is “NONE”, then no source is currently selected

```
TRIG:SCAN1:SOURCE?
```

Might return “EXT1”, indicating the external trigger input is selected for the scan engine (sequencer).

TRIGger:MODule<N>:DELay Command Summary

The “TRIGger:MODULE<N>:DELay” command is used to program the delay time between the receipt of an input trigger on any of the enabled trigger inputs and the generation of the output trigger pulse.

The format for these commands is:

```
TRIGger:MODULE<N>:DELay {<trigger delay>|MIN|MAX}
TRIGger:EXTernal2:DELay {<trigger delay>|MIN|MAX}
TRIGger:SCAN:DELay {<trigger delay>|MIN|MAX}
```

The <trigger delay> sets the programmable trigger delay between the receipt of the trigger input signal and the generation of the (first) trigger output pulse. The trigger delay ranges between 0 (= MIN) and 4294.967295 (= MAX).

The present trigger delay may be read back by using one of the following queries:

```
TRIGger:MODule<N>:DELay?
TRIGger:EXTernal2:DELay?
TRIGger:SCAN:DELay?
```

These queries return the presently programmed trigger delay time for the specified trigger output. Alternatively, the user may use the following queries to determine the minimum and maximum possible trigger delay times:

```
TRIGger:MODule<N>:DELay? MIN
TRIGger:EXTernal2:DELay? MIN
TRIGger:SCAN:DELay? MIN
TRIGger:MODule<N>:DELay? MAX
TRIGger:EXTernal2:DELay? MAX
TRIGger:SCAN:DELay? MAX
```

Refer to the table below for min and max delays. Note there is no support for changing the trigger delay for the MODule11, MODule12, MODule13, MODule14, and SCAN triggers.

	Minimum	Maximum
TRIGger:MODule1	0.0	0.65535
TRIGger:MODule2	0.0	0.65535
TRIGger:MODule3	0.0	0.65535
TRIGger:MODule4	0.0	0.65535
TRIGger:MODule5	0.0	0.65535
TRIGger:MODule6	0.0	0.65535
TRIGger:MODule7	0.0	0.65535
TRIGger:MODule8	0.0	0.65535
TRIGger:MODule9	0.0	0.65535

	Minimum	Maximum
TRIGger:External2	0.0	0.65535
TRIGger:MODule11	0.0	0.0
TRIGger:MODule12	0.0	0.0
TRIGger:MODule13	0.0	0.0
TRIGger:MODule14	0.0	0.0
TRIGger:SCAN	0.0	0.0

TRIGger:MODULE<N>:LEVel Command Summary

The "TRIGger:MODULE<N>:LEVel" command is used to program the level that is asserted for the trigger output pulse. This only has relevance when the trigger output is set to level sensitivity (see the "TRIGger:MODULE<N>:SENSitivity" command). When the trigger output is edge sensitive, this command has no effect.

The format for these commands is:

```
TRIGger:MODule<N>:LEVel { HIGH | LOW }
TRIGger:EXTernal2:LEVel { HIGH | LOW }
TRIGger:SCAN1:LEVel { HIGH | LOW }
```

The trigger level may be HIGH or LOW. When set to HIGH, the trigger output is a TTL voltage level high when the trigger is asserted. When set to LOW, the trigger output is a TTL voltage level low when the trigger is asserted.

The present trigger level sensitivity may be read back by using one of the following queries:

```
TRIGger:MODULE<N>:LEVel?
TRIGger:EXTernal2:LEVel?
TRIGger:SCAN:LEVel?
```

After a power-up, or receipt of the "*RST" command, all are set to LOW.

TRIGger:MODULE<N>:SLOPe Command Summary

The "TRIGger:MODULE<N>:SLOPe" command is used to program the direction of the leading edge of the trigger pulse that is generated for the trigger output pulse. This only has relevance when the trigger output is set to edge sensitivity (see the "TRIGger:MODULE<N>:SENSitivity" command). When the trigger output is level sensitive, this command has no effect.

The format for these commands is:

```
TRIGger:MODULE<N>:SLOPe { POSitive | NEGative }
TRIGger:EXTernal2:SLOPe { POSitive | NEGative }
TRIGger:SCAN:SLOPe { POSitive | NEGative }
```

The trigger level may be POSitive or NEGative. When set to POSitive, the trigger output is a pulse that transitions low-high-low. When set to NEGative, the trigger output is a pulse that transitions high-low-high. Here, the high and low refer to TTL compatible voltages.

The present trigger edge sensitivity may be read back by using one of the following queries:

```
TRIGger:MODULE<N>:SLOPe?
TRIGger:EXTernal2:SLOPe?
TRIGger:SCAN:SLOPe?
```

These queries return the short form of the item. The reply will thus be one of the following:

```
POS
NEG
```

After a power-up, or receipt of the “*RST” command, all are set to NEGative.

TRIGger:MODULE<N>:SENSitivity Command Summary

The “TRIGger:MODULE<N>:SENSitivity” command is used to select either edge or level sensitivity for the output trigger pulse(s). If edge sensitivity is selected, the trigger pulse behaves as set by the “TRIGger:MODULE<N>:SLOPe” command. If level sensitivity is selected, the trigger output behaves as set by the “TRIGger:MODULE<N>:LEVel” command

The format for these commands is:

```
TRIGger:MODULE<N>:SENSitivity { EDGE | LEVel }
TRIGger:EXTernal2:SENSitivity { EDGE | LEVel }
TRIGger:SCAN:SENSitivity { EDGE | LEVel }
```

The trigger sensitivity may be EDGE or LEVel. When set to EDGE, the trigger output is pulse that transitions low-high-low or high-low-high as set by the “TRIGger:MODULE<N>:SLOPe” command. When set to LEVel, the trigger output is a constant level that is selected by the “TRIGger:MODULE<N>:LEVel” command. The present trigger edge sensitivity may be read back by using one of the following queries:

```
TRIGger:MODULE<N>:SENSitivity?
TRIGger:EXTernal2:SENSitivity?
TRIGger:SCAN:SENSitivity?
```

These queries return the short form of the item. The reply will thus be one of the following:

```
EDGE
LEV
```

After a power-up, or receipt of the “*RST” command, all are set to LEVel.

TRIGger:MODULE<N>:ENABle Command Summary

The “TRIGger:MODULE<N>:ENABle” command is used enable or disable the generation of trigger pulses (or levels) from the selected output trigger.

The format for these commands is:

```
TRIGger:MODULE<N>:ENABle { ON | OFF | 1 | 0 }
TRIGger:EXTernal2:ENABle { ON | OFF | 1 | 0 }
TRIGger:SCAN:ENABle { ON | OFF | 1 | 0 }
```

The present enable state may be read with the following queries

```
TRIGger:MODULE<N>:ENABle?
TRIGger:EXTernal2:ENABle?
TRIGger:SCAN:ENABle?
```

These queries return the reply “1” if the trigger output is enabled or “0” if the trigger output is disabled.

After a power-up, or receipt of the “*RST” command, all output trigger outputs are disabled.

TRIGger:MODULE<N>:WIDTh Command Summary

The “TRIGger:MODULE<N>:WIDTh” command is used is used to program the width of the trigger output pulse. The range of valid trigger pulse width that can be programmed is 10.0 µsec to 1.280 msec (1280 µsec) in 10 µsec steps.

Also, the trigger widths for MODule11, MODule12, MODule13, and MODule14 (the DMM PXI modules) are fixed at 10 µsec.

Refer to the table below for min and max widths.

	Minimum	Maximum
TRIGger:MODule1	0.00001	0.00127
TRIGger:MODule2	0.00001	0.00127
TRIGger:MODule3	0.00001	0.00127
TRIGger:MODule4	0.00001	0.00127
TRIGger:MODule5	0.00001	0.00127
TRIGger:MODule6	0.00001	0.00127
TRIGger:MODule7	0.00001	0.00127
TRIGger:MODule8	0.00001	0.00127
TRIGger:MODule9	0.00001	0.00127
TRIGger:External2	0.00001	0.00127
TRIGger:MODule11	0.00001	0.00001
TRIGger:MODule12	0.00001	0.00001
TRIGger:MODule13	0.00001	0.00001
TRIGger:MODule14	0.00001	0.00001
TRIGger:SCAN	0.00001	0.00001

CLOCk Command Summary

The CLOCk commands are used for clock routing. Each plug-in module, either X-bus or PXI, may select at most one trigger source from the clock routing block. The clock routing block is shown in Figure 7-4.

The CLOCk command tree is shown below:

```
:CLOCk
      :MODule<N>          (N = 1 … 9, 11 … 14)
            :SOURce       EXTernal|INTernal|1 … 9 |11 … 14|NONE
            :SOURce?
```

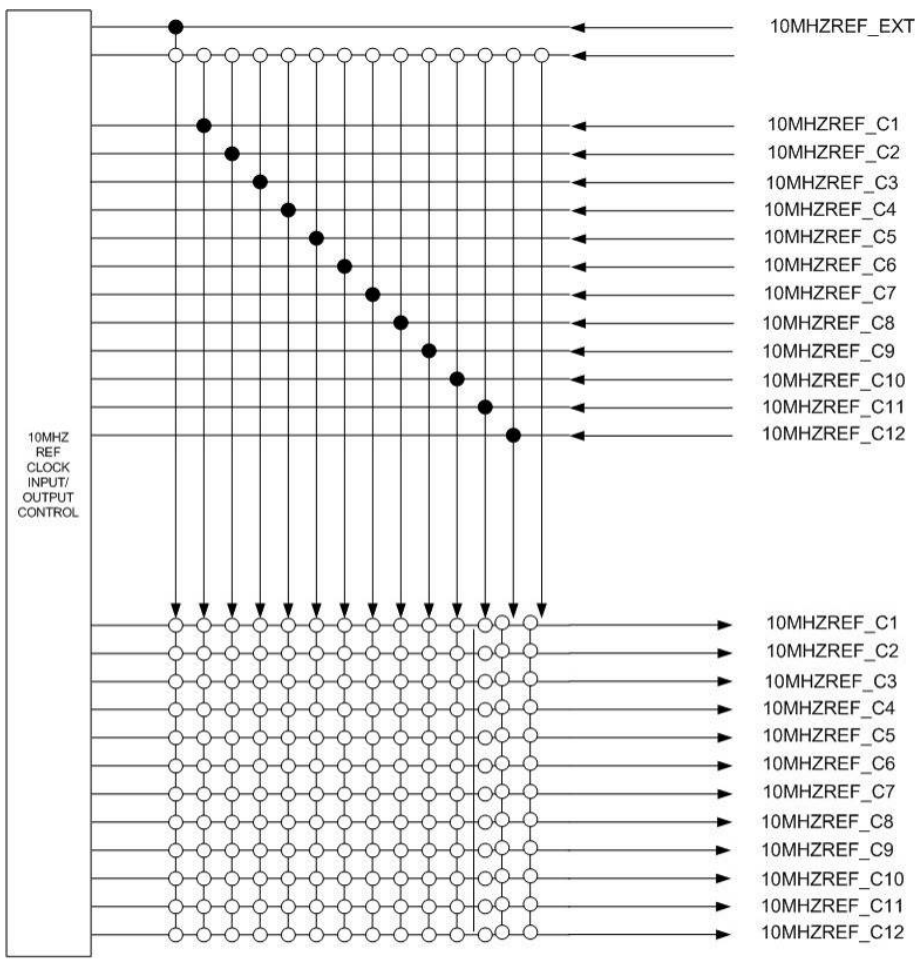


Figure 7-4: Reference Clock Routing Block Diagram

CLOCk:MODule<N>:SOURce Command Summary

The “CLOCk:MODule<N>:SOURce” command is used to select a clock source for any of the plug in modules. The clock source may be the internal 10 MHz clock generator, one of the plug-in module’s clock out signal, or the REF IN input signal to the chassis.

The format for this command is:

```
CLOCk:MODule<N>:SOURce EXTernal|INTernal|1 … 9|11 … 14|NONE
```

This command may be used to select the source for the clock for any of the plug-in modules.

The <N> following the MODule identifies which of the plug-in modules is being programmed. MODule1 identifies the plug-in module in slot 1, MODule2 identifies the plug-in module in slot 2, and so on. MODule10 is NOT valid, but all other identifiers MODule1 through MODule14 are valid

The source may be any of the following:

- EXTernal – The REF IN input clock is routed to the plug-in module
- INTernal – The internal 10 MHz clock is routed to the plug-in module
- 1 … 9 – The card plugged into slot 1 to 9 is routed to the plug-in module
- NONE – No clock is routed to the plug-in module

The present clock source for a plug-in module may be read with the query

```
CLOCk:MODule<N>:SOURce?
```

This query will return one of the following replies:

- EXT – The CLK REF IN/OUT input clock is routed to the plug-in module
- INT – The internal 10 MHz clock is routed to the plug-in module
- 1 … 9 – The card plugged into slot 1 to 9 is routed to the plug-in module
- NONE –No clock is routed to the plug-in module

After a power-up, or receipt of the “*RST” command, the source is set to “NONE” for every plug-in.

Note that a card may be used to provide the clock source to one or more switch and PXI plug-in cards. When any plug-in card selects an card as a clock source, the selected card’s clock source will be set to NONE (since its clock is a source). Also note that it is not possible to select a slot to be its own clock source. An attempt to do so will result with SCPI error code - 221 (“Settings Conflict”) posted to the error queue.

MMEMory Command Summary

The “MMEMory” commands are used to exchange data between the system and the USB memory device. These commands will work only when a USB memory device is plugged into the front USB port. The MMEMory command tree is shown below:

```
:MMEMory
      :CATalog?
      :DELete                <file name>
      :LOAD
            :SEQuence        <sequence name>
            :SCAN            <scan list name>
      :STORe                 <file name>
            :SEQuence        <sequence name>
            :SCAN            <scan list name>
            :DATA            <data file name> [,<first
                           point>[,<last point>]]
```

The “MMEMory:CATalog?” command is used to read all of the file names on the USB memory device. This query returns a reply in the form:

```
<bytes used>,<bytes available> [,<file entry>…]
```

The <bytes used>indicate the total number of bytes in use. The <bytes available> indicate the total number of bytes available on the memory stick. The remainder of the reply holds one or more file entries, one per file on the memory stick. Each <file entry> consists is reported in the form

```
<file name> , <file type> , <file size>
```

Note that the <file name> cannot include commas. Also, since all files will be stored and retrieved in ASCII format, each <file type> that is recognized by the system will be reported as “ASC”.

The “MMEMory:DELete” command may be used to delete a file from the USB memory device.

The “MMEMory:STORe” commands are used to store scan lists, sequences, or captured data from the 1830 platform to the USB memory device. The “MMEMory:STORe:SEQuence” command may be used to save the sequence name to the USB memory device. The file name on the USB memory device will be the sequence name with a “.seq” extension.

Similarly, the “MMEMory:STORe:SCAN” command can be used to save the presently defined scan list to the USB memory device. This command will save the scan list to the file name specified with a “.scn” extension.

Finally, the presently captured data may be saved to the USB memory device with the “MMEMory:STORe:DATA” command. This command will store the data points specified into the file name. The file format will be a “.csv” file, with one data record per line. Commas will be used to separate the various components of the data record as specified with the “FORMat” command.

The “MMEMory:LOAD” commands are used to load a sequence definition or a scan list definition from the USB memory device. The “MMEMory:LOAD:SEQuence” command may be used to load sequences (which must have a “.seq” extension), while the “MMEMory:LOAD:SCAN” command can be used to load a scan list (with a .scn extension) from the USB memory device.

DMM-Specific SCPI Commands Summary

The “SENSe” and “READ?” commands and the “MEASure?” queries are used to configure and read the DMM. The SCPI commands related to the DMM are as follows:

```
:SENSe
  :VOLTage
      [:DC] | AC            [<range>,[<resolution>,]]<chan list>
            :RANGe          <range> , <chan list>
                  :AUTO     {ON|OFF|1|0},<chan list>
                  :AUTO?    <chan list>
            :RANGe?         <chan list>
            :RESolution     <resolution> , <chan list>
            :RESolution?    <chan list>
            :APERture       <aperture> , <chan list>
            :APERture?      <chan list>

  :RES | FRES            [<range>,[<resolution>,]]<target chan>
      :RANGe                <range> , <chan list>
            :AUTO           {ON|OFF|1|0}, <chan list>
            :AUTO?          <chan list>
      :RANGe?               <chan list>
      :RESolution           <resolution> , <chan list>
      :RESolution?          <chan list>
      :APERture             <aperture> , <chan list>
      :APERture?            <chan list>
      :COMPensation         {ON|OFF|1|0}, <chan list>
      :COMPensation?        <chan list>

  :CURRent
      [:DC] | AC            [<range>,[<resolution>,]]<chan list>
            :RANGe          <range> , <chan list>
                  :AUTO     {ON|OFF|1|0},<chan list>
                  :AUTO?    <chan list>
            :RANGe?         <chan list>
            :RESolution     <resolution> , <chan list>
            :RESolution?    <chan list>
            :APERture       <aperture> , <chan list>
            :APERture?      <chan list>
            :LEAKage        <chan list>
                  :VOLTage  <DC set voltage>,<chan list>
                  :VOLTage? <chan list>
```

```
:SENSe
  :CAPacitance                [<range>,[<resolution>,]]<chan list>
      :RANGe                  <range> , <chan list>
            :AUTO             {ON|OFF|1|0},<chan list>
            :AUTO?            <chan list>
      :RANGe?                 <chan list>
      :RESolution             <resolution> , <chan list>
      :RESolution?            <chan list>
      :AC                     [<range>,[<resolution>,]]<chan list>
            :RANGe            <range>, <chan list>
            :AUTO             {ON|OFF|1|0},<chan list>
            :AUTO?            <chan list>
            :RANGe?           <chan list>
            :RESolution       <resolution>, <chan list>
            :RESolution?      <chan list>
            :APERture         <aperture>, <chan list>
            :APERture?        <chan list>
            :VOLTage          <voltage>, <chan list>
            :VOLTage?         <chan list>
            :DELay            <delay>, <chan list>
            :DELay?           <chan list>

  :INDuctance                 [<range>,[<resolution>,]]<chan list>
      :RANGe                  <range>, <chan list>
            :AUTO             {ON|OFF|1|0},<chan list>
            :AUTO?            <chan list>
      :RANGe?                 <chan list>
      :RESolution             <resolution>, <chan list>
      :RESolution?            <chan list>
      :APERture               <aperture>, <chan list>
      :APERture?              <chan list>
      :FREQuency              <frequency>, <chan list>
      :FREQuency?             <chan list>

  :FREQuency                  [<freq range>,] <chan list>
      :RANGe
            :LOWer            <freq range>, <chan list>
            :LOWer?           <chan list>
      :VOLTage
            :RANGe            <voltage range>, <chan list>
            :RANGe?           <chan list>
                  :AUTO       {ON|OFF|1|0}, <chan list>
                  :AUTO       <chan list>
      :APERture         <aperture>, <chan list>
      :APERture?        <chan list>
```

```
:PERiod                     [<freq range>,] <chan list>
    :RANGe
          :LOWer            <freq range> , <chan list>
          :LOWer?           <chan list>
    :VOLTage
          :RANGe            <voltage range> , <chan list>
          :RANGe?           <chan list>
                :AUTO       {ON|OFF|1|0}, <chan list>
                :AUTO?      <chan list>
    :APERture         <aperture> , <chan list>
    :APERture?        <chan list>

:DCYCle                     [<freq range> ,] <chan list>
    :RANGe
          :LOWer            <freq range> , <chan list>
          :LOWer?           <chan list>
    :VOLTage
          :RANGe            <voltage range> , <chan list>
          :RANGe?           <chan list>
                :AUTO       {ON|OFF|1|0}, <chan list>
                :AUTO?      <chan list>
    :APERture         <aperture> , <chan list>
    :APERture?        <chan list>
    :DIODe                  [<range>,[<resolution>,]]<chan list>
          :RANGe            <current range> , <chan list>
                :AUTO       {ON|OFF|1|0}, <chan list>
                :AUTO?      <chan list>
          :RANGe?           <chan list>
          :APERture         <aperture> , <chan list>
          :APERture?        <chan list>

:TOTalize             <chan list>
    :VOLTage
          :RANGe      <voltage range> , <chan list>
          :RANGe?     <chan list>
    :THReshold        <DC voltage> , <chan list>
    :STARt            <chan list>
    :STOP             <chan list>

:PWIDth | NWIDth      <chan list>
    :VOLTage
          :RANGe      <voltage range> , <chan list>
          :RANGe?     <chan list>
    :THReshold        <DC voltage> , <chan list>
    :THReshold?       <chan list>
```

```
:SENSe
      :TEMPerature              <chan list>
            :APERture           <aperture> , <chan list>
            :APERture?          <chan list>
      :UNIT                     {C|F|K} , <chan list>
      :UNIT?                    <chan list>
      :TRANsducer
            :TYPE               {TCouple | RTD },<chan list>
            :TCouple
               :RJUNction       <temperature>, <chan list>
               :RJUNction?      <chan list>
               :TYPE            {B|E|J|K|N|R|S|T}
               :TYPE?
               :P               <param A> , <chan list>
               :PA?             <chan list>
               :P               <param B> , <chan list>
               :PB?             <chan list>
               :P               <param B> , <chan list>
               :PC?             <chan list>
            :RTD
               :TYPE            { 385 | 3911 | 3916 | 3926 | CU }
               :TYPE?
               :R0              <ice point resistance>,<chan list>
               :R0?             <chan list>
:SOURce
      :VOLTage
         [:DC]                  <DC voltage> , <chan list>
         :AC                    <AC voltage>,<frequency>,<chan list>
      :CURRent
         [:DC]                  <DC current> , <chan list>
      :PULSe                    <Ampl>, <Freq>[,<Duty Cycle>
                                 [,<count>]] <chan list>

:TRIGger
      :DELay                    <DELay><chan list>
      :DELay?                   <DELay?.><chan list>
      :MODE                     {IMMediate|EXTernal|THReshold},
                                 <chan list>
      :MODE?                    <chan list>
      :THReshold                <DC voltage> , <chan list>
      :THReshold?               <chan list>

:SAMPle
      :COUNt                    <count> , <chan list>
```

```
        :COUNt?                 <chan list>

    :MEASure
        :VOLTage
            [:DC]? | AC?        [<range>,[<resolution>,]]<chan list>
        :CURRent
            [:DC]? | AC?        [<range>, [<resolution>,]] <chan list>
        :RES? | FRES?           [<range>,[<resolution>,]]<chan list>
        :CAPacitance?           [<range>,[<resolution>,]]<chan list>
            :AC?                [<range>,[<resolution>,]]<chan list>
        :INDuctance?            [<range>,[<resolution>,]]<chan list>
            :Q?                 [<range>,[<resolution>,]]<chan list>
        :DIODe?                 [<range>,[<resolution>,]]<chan list>
        :FREQuency?             [<range>,] <chan list>
        :PERiod?                [<range>,] <chan list>
        :DCYCle?                [<range>,] <chan list>
        :PWIDth?                [<voltage range>, [<threshold>,]]<chan list>
        :NWIDth?                [<voltage range>, [<threshold>,]]<chan list>
        :TEMPerature?           <chan list>

    :READ?                      <chan list>

    :FETCh?                     <chan list>
```

Note: the following commands are supported by the 4102 DMM option but NOT the 4101 option.

- MEASure:EXTResistance? command
- SENSe: EXTResistance command and subcommands
- MEASure:INDuctance? command
- SENSe:INDuctance command and subcommands
- MEASure:CAPacitance:AC? command
- SENSe:CAPacitance:AC command and subcommands
- MEASure:CURRent[:DC]:LEAKage? Command
- SENSE:CURRent[:DC]:LEAKage commands and subcommands
- MEASure:DCYCle? command
- SENSe: DCYCle command and subcommands
- SENSe:TOTalize commands and subcommands
- SENSE:xxx:THReshold commands

SENSe Command Summary

The various “SENSe” commands are used to select a measurement type for one or more DMM. These commands can also select the range and resolution for the DMM. When executed as part of a command (not a “SEQuence:STEP” or “SEQuence:SELect” command), these commands select the measurement mode for the specified channel lists.

When used as part of the “SEQuence:STEP” or “SEQuence:SELect” commands, these commands associate setting the measurement mode of the DMM when the specified sequence (or scan list) is executed.

Each of the “SENSe” commands takes a <target channel list> parameter. This parameter identifies the DMMs that will be affected by the command. For example, the <target channel list>

```
(@12(0))
```

selects the DMM in plug-in slot 12 as the target of the command, and the <target channel list>

```
(@11(0),13(0))
```

selects DMMs in plug-in slots 11 and 13 as the target of the command. Note that the DMMs contain only a single channel (0). This format is used for consistency with switching cards.

The following measurement modes are supported:

`SENSe:VOLTage`	Selects DC voltage measurement
`SENSe:VOLTage:DC`	Selects DC voltage measurement
`SENSe:VOLTage:AC`	Selects AC voltage measurement
`SENSe:RESistance`	Selects 2-wire resistance measurement
`SENSe:FRESistance`	Selects 4-wire resistance measurement
`SENSe:CURRent`	Selects DC current measurement
`SENSe:CURRent:DC`	Selects DC current measurement
`SENSe:CURRent:AC`	Selects AC current measurement
`SENSe:CAPacitance`	Selects capacitance measurement
`SENSe:CAPacitance:AC`	Selects AC capacitance measurement
`SENSe:INDuctance`	Selects inductance measurement
`SENSe:FREQuency`	Selects frequency measurement
`SENSe:PERiod`	Selects period measurement
`SENSe:DCYCle`	Selects the duty cycle measurement
`SENSe:TEMPerature`	Selects temperature measurement
`SENSe:DIODe`	Selects diode characterization measurement
`SENSe:TOTalize`	Selects counter measurement

`SENSe:PWIDth`	Selects positive pulse width measurement
`SENSe:NWIDth`	Selects negative pulse width measurement

The sense commands may also include a range. If a range is specified, a resolution may also be specified. For example,

`SENSE:VOLT:DC 2.4,(@12(0))`	Puts the DMM in slot 12 into DC 2.4 volt range
`SENSE:VOLT:AC 24,0.001,(@13(0))`	Puts the DMM in slot 13 into AC voltage mode. 24 volt range, 0.001 (5.5 digit) resolution

If no range is specified, the DMM will be placed into turn auto-ranging on.

The “SENSE” commands can also be used to select the range, resolution, number of power line cycles (number of cycles of power line frequency to set the aperture), or the aperture value.

`SENSE:RES:RANGE 10000,(@14(0))`	Puts the DMM in slot 14 into 2-wire resistance mode with a range of 10K (actually 24K for the 410*x* DMM).
`SENSE:VOLT:RES 0.0001,(@11(0))`	Puts the DMM in slot 11 into DC voltage mode and selects a resolution to 100 microvolts. The last range selected for the DMM is used. The last range selected for the DC voltage mode for the DMM is used.
`SENSE:IND:APER 0.02,(@11(0))`	Sets the aperture of the DMM to 20 milliseconds.

MEASure? Query Summary

The “MEASure?” queries configure the DMM to make a measurement, trigger a measurement, and place the result in the response buffer. The “MEASure?” queries thus are a combination of a “SENSe” command and a “READ?” query.

The examples below illustrate the use of the “MEASure?” queries:

`MEAS:VOLT? 2.0,0.00001,(@12(0))`	Puts the DMM in slot 12 into DC voltage mode, 2.4V range (actually 2.4V with 410*x* DMM), 6.5 digit resolution, triggers the DMM, reads the measurement, and places it in the output buffer of the 1830 platform.
`MEAS:RES? 10E3,(@11(0))`	Puts the DMM in slot 11 into 2-wire resistance mode, 10K ohm range (24K Ohm with 410*x* DMM), triggers the DMM, reads the measurement, and places it in the output buffer of the 1830 platform.
`MEAS:FREQUENCY? (@13(0))`	Puts the DMM in slot 13 into frequency measurement mode and in auto-range, triggers the DMM, reads the measurement, and places it in the output buffer of the 1830 platform.

READ? Query Summary

The “READ?” command triggers the specified DMM and reads the measurement. The current measurement type, range, and resolution are used to make the measurement.

`READ? (@12(0))`	Triggers the DMM in slot 12 and places the measurement into the output buffer of the 1830 platform.

FETCh? Query Summary

The “FETCh?” command reads the currently available measurement from the DMM. Once the DMM is armed (INIT) and triggered, it makes at least one measurement. This measurement is retrieved from the DMM and placed in the output buffer of the interface on which the query is received.

`FETCH? (@12(0))`	Fetches the next measurement and places it into the output buffer of the 1830 platform.

SOURce Commands Summary

The “SOURce” commands provide access to the various source functions of the SMX2064 DMM. The “SOURce” command tree is shown below:

```
:SOURce
      :VOLTage
         [:DC]                  <DC voltage> , <chan list>
            :MODE               {OPEN|CLOSE|1|0},<chan list>
            :MODE?              <chan list>
            :VOLTage?           <chan list>
            :SENSe
               :HIGH?           <chan list>
               :LOW?            <chan list>
               :BOTH?           <chan list>
         :AC                    <AC voltage>,<frequency>,
                                <chan list>
      :CURRent
         [:DC]                  <DC current> , <chan list>
            :MODE               {OPEN|CLOSE|1|0}, <chan list>
            :MODE?              <chan list>
            :VOLTage?           <chan list>
            :SENSe
               :HIGH?           <chan list>
               :LOW?            <chan list>
               :BOTH?           <chan list>
      :PULSe                    <Ampl>, <Freq>[,<Duty
                                Cycle>[,<count>]]<chan list>
```

The “SOURce:VOLTage:DC” command is used to apply a DC voltage in the range -10.0 to +10.0.

The “SOURce:VOLTage:DC:MODE” command is used to select open or closed loop mode of operation. When the open loop measurement is selected, this implies a 2-wire connection (V+ and V-) to the DUT. This causes the DMM to measure the output on the V+ / V- lines and results in approximately 12 bit accuracy. When closed loop measurement mode is used, the DMM sources the voltage on the V+ / V- lines and measures on the I+ / I- lines. This implies a 4-wire connection to the DUT.

The “SOURce:VOLTage:DC:VOLTage?” query is used to read back the voltage generated by the output. When the MODE is CLOSE, this query must be used several times to enable the closed loop mode to generate a more accurate output.

The “SOURce:VOLTage:DC:SENSE” commands may only be used when the closed loop mode is selected. These commands return the voltage between the V- and I+ terminals (“HIGH?”), the voltage between the V- and I-terminals (“LOW?”), or between the I+ and I- terminals (“BOTH?”).

The “SOURce:VOLTage:AC” command is used to apply an AC voltage source at the specified amplitude and frequency. The amplitude range is between 0.05 and 7.25 Vrms. The frequency is between 0.5 Hz and 200 kHz.

The “SOURce:CURRent” command is used to source a DC current via the input terminals. This allows the current to be set in the range 0.1 μA to 12.5 mA.

The “SOURce:CURRent:DC:MODE” command is used to select open or closed loop mode of operation. When the open loop measurement is selected, this implies a 2-wire connection (V+ and V-) to the DUT. This causes the DMM to measure the output on the V+ / V- lines and results in approximately 12 bit accuracy. When closed loop measurement mode is used, the DMM sources the voltage on the V+ / V- lines and measures on the I+ / I- lines. This implies a 4-wire connection to the DUT.

The “SOURce:CURRent:DC:SENSE” commands may only be used when the closed loop mode is selected. These commands return the voltage between the V- and I+ terminals (“HIGH?”), the voltage between the V- and I- terminals (“LOW?”), or between the I+ and I- terminals (“BOTH?”).

The “SOURce:PULSe” command is used to generate a series of pulses. The amplitude can be set between -10.0V and +10.0V. The frequency can be between 7.8125 Hz (128 msec period) and 29411 Hz (34 μsec period). The Duty Cycle can be between 1 and 99. The <count> determines the number of times the pulse is generated ; a value of 0 indicates free running, while values of 1 to 32000 are accepted to generate a specific number of pulses.

Firmware Version

The embedded software (firmware) supports downloading newer versions from our support website so that the 1830 unit does not have to be returned to the factory for a software update. The firmware uses the built-in web-page interface to upload the software through the 1830 Ethernet port. See **Chapter 4, Web-Page Interface**, for more information.

The serial number and version of the firmware in the system is available via the SCPI command interface (see the “SYSTem:CTYPe?” query).

The module type and serial number for plug-in modules is available via the SCPI command interface (see the “SYSTem:CTYPe?” query).

Appendix A Specifications

General

Plug-In Slots	Nine for switch cards or instrument adapter panels. A maximum of 4 DMMs may be installed.
Alarm Capability	Monitor Mode Each Channel in a Scan List Measurements Outside of Pre-set Limits
Thermal Monitoring	Proportional fan speed controller and two thermal sensors to control fan speed
Timestamp Capability	Type: Trigger Events and associated measurements Accuracy: ± 10 μs
Real-Time Clock	Type: Trigger Events and associated measurements Accuracy: ± 5 ppm
State Machine Capability	Type: Scanned measurements using embedded scripting language (SCPI) I/O: Read/write capability via USB interface, .CSV data format Channel Lists: Lists may contain channel sequences and measurements to be performed on the channels. Each channel in a sequence can have its own measurement type, range, aperture, and signal conditioning Parallel Operation: If there is more than one measurement instrument installed in the system, scan measurements may be taken in parallel. Scan Operation: Scan modes are compatible with internal and external instruments (for external instruments there must be a trigger port that can be controlled by the system as well as a measurement ready signal. System can store timestamps for external measurement device although not measurements.) Timestamps: Each measurement in a scan can be time-stamped as an option.
Scaling	Analog channels, Alarms, Monitoring channels
System Management	Firmware upgrades: via LAN port Data file I/O: via USB interface

Analog Bus

Type	Single-ended and differential analog bus Two banks. Bank A: Slots 1–5; Bank B: Slots 6–9
Single-ended Signal	Channels: 16 Characteristic Impedance: 50 Ω Bandwidth of Bank A: 90 MHz Bank B: 170 MHz Bank A to B: 50 MHz Current: 1 Amp Voltage: 150 Volts (Pollution degree 1 and 2) Banks: 2 banks (5 left slots and 4 right slots) may be joined together which reduces the channel count to 8
Differential Signal	Channels: 8 Characteristic Impedance: 50 Ω Bandwidth of Bank A: 75 MHz Bank B: 220 MHz Bank A to B: 50 MHz Current: 2 Amps Voltage: 300 Volts (Pollution degree 1) 250 Volts (Pollution degree 2) Banks: 2 banks (5 left slots and 4 right slots) may be joined together which reduces the channel count to 4

Triggering Characteristics

Trigger Source	External Internal: 10 µs to 99 h, 10 µs resolution, 1 to 50,000 counts or continuous Intra-System: Internal instruments and switches can send and receive triggers from each other via internal trigger bus

Software

Drivers	IVI-C and IVI-COM ActivATE Test Platform LabVIEW™
User Interfaces	Web pages VXI plug&play soft front panel ActivATE test platform control panels
Test Software	ActivATE test platform (optional) includes a modern development environment built on the .NET framework Scripting: VBscript Test Executive Debugger Driver Wizard

Front Panel I/O

USB Interface	USB 2.0 Full Speed Type A port
Indicators	LAN Status Power

Rear Panel I/O

Control Port Interfaces	USB 1.1 Full Speed Type B port GPIB port (IEEE-488.2) LAN port: RJ-45 (LXI Class C) RS-232: (Factory debug port)
Instrument I/O	Trigger In: BNC, TTL (Min: 2.0 V, Max: 5.5 V) Trigger Out: BNC, TTL (Min: 2.0 V, Max: 5.5 V) Instrument Trigger In/Out: Per slot configurable trigger router Ref (Clock) In: BNC, TTL (Min: 2.0 V, Max: 5.5 V)
Indicators	LAN Status

Environmental

Temperature	Operating: 0°C – 50°C (Ethernet and USB) Storage: -40°C – 71°C
Relative Humidity	80% RH at 40°C

Conformance Testing

Emissions/Immunity	EN61326: 2006, Class B
Safety	EN61010-1: 2001 Pollution degree 1: 300 V Pollution degree 2: 250 V Analog Bus safety interlock I/O: Not for connection directly to mains power
Material Handling	RoHS (4101 and 4102 options have WEEE Directive, Category 9 exemption.)
LXI Conformance	Class C
Mechanical Chassis Weight	22 lb 4oz. (10.1 kg)
Dimensions	5.25” H x 17.0” W x 13.8” D (13.34 cm x 43.18 cm x 35.05 cm) Chassis dimensions: without handles, ears, bumpers, or cable connectors

Reliability

Analog Bus Switching Time	< 10 ms (includes settling time
Rated Switch Operations	Bus Relays Mechanical: 1 X 10^8 Electrical: 200,000 @ 300 Vrms/0.416 A – 125 VA
MTBF	TBD
Relay Operations Counter	Running total of operations stored in on-board non-volatile memory

Interface Data

Maximum Overall System Power Dissipation	Ambient 0 – 45 C: 225 W Ambient 45 – 50 C: 80 W Operation between 3048 m (10,000 feet) and 6,400 m (15,091 feet) requires derating the maximum overall power dissipation to 180 W (0-45 ºC) and 64 W (45-50 º C)
Total Available Slot Power	(not to exceed 200 W) **120 W Maximum:** +24 V at 5.0 A +12 V at 5.0 A -12 V at 3.0 A **80 W Maximum:** +5 VDC at 16.0 A +3.3 VDC at 8.0 A

Power Supply

Input voltage	Universal 90 V to 250 V
Input Frequency	47 – 63 Hz
Power Consumption	100 VA (typical), 350 VA (maximum)
Fuse	250 V IEC SLO-BLO, 5x20mm, 5 A

This page was left intentionally blank.

Appendix B
Troubleshooting

If you have difficulty using the 1830 System, refer to the following troubleshooting tables. If you are still unable to resolve the problem, contact Customer Support.

Power Up

Problem	Possible Causes	What to do
Unit does not turn on (fan not running and no indicator lights).	Not connected to proper power source.	Make sure that the 1830 is connected to a live power source, using a functional power cord. For input power requirements, see the **Chapter 1, Overview and Features**, under the heading **Power Panel**.
	Fuse missing, blown, or incorrectly installed.	Check fuse, verifying that it has the required capacity as marked on the 1830 rear panel. Make sure the fuse is properly installed (see **Chapter 1, Overview and Features**, under the heading **Power Panel**).

Power Indicator Light

(See **Figure 1-3: Front Panel USB and Indicator Lights** in Chapter 1.)

Problem	Possible Causes	What to do
Indicator lights go blank (was working).	Power has been interrupted.	Verify that the AC input is connected to a live power outlet. Verify that the power switch is still in the ON position. Check fuse, verifying that it has not blown.

LAN Indicator Light

(See **Figure 1-7: Front Panel USB and Indicator Lights** in Chapter 1.)

Problem	Possible Causes	What to do
LAN light is red	Failure to acquire a valid IP address.	Disconnect and reconnect LAN cable. Confirm LAN cable is connected to network properly. Press the LAN reset button.
	Detection of a duplicate IP address.	Change the IP address of the other device. If in manual setting, change the IP address of the 1830. Press the LAN reset button.
	LAN cable disconnected.	Reconnect LAN cable.

Password

Problem	Possible Causes	What to do
Password does not work or you have forgotten the password.	-----	Press the LAN reset button and you password returns to being blank. Reset the password in your system with the web-page interface.

IEEE-488.2 (GPIB) Remote Interface

Problem	Possible Causes	What to do
1830 does not respond to GPIB commands.	Commands sent to wrong GPIB address.	Make sure that the commands are sent to the GPIB address currently assigned to the 1830.
	The 1830 is set to the same GPIB address as another device in the system.	Make sure that the 1830 is set to a unique GPIB address.
	GPIB may have been disabled with a SCPI command.	Use the web-page interface to enable the GPIB. Send a SCPI command to enable the GPIB.

Switch Cards

Problem	Possible Causes	What to do
Slot containing a card is indicated as an “empty slot” when using the web-page interface.	Card not properly seated in slot.	Loosen the retaining screws of the card. Pull the card part-way out, then gently push it back in until it stops. Press it firmly to seat the card’s fingers into the backplane connector. When properly installed, the connector end of the card is flush with the rear panel of the 1830.

DMM Card

Problem	Possible Causes	What to do
Incorrect voltage reading	Relays not set properly to read the voltage.	Verify that the relays in the switch card, analog backplane, or DMM card were set properly.
Incorrect voltage reading after connecting the Guard input.	Relays not set properly to read the voltage.	Verify that the relays in the switch card, analog backplane, or DMM card were set properly.
	Guard voltage was over the 26 VDC maximum which created a fault and opened a relay to prevent damage.	Use SCPI commands to read the system error. If the problem was overvoltage, check to see if the guard voltage coming from your source is less than 28 VDC. Then reset the relay and resume measurement. ***Note:*** *If the guard voltage was over 34 VDC, you may have damaged the circuitry in the card. Call Customer Support for assistance.*

Flash (Thumb) Drive

Problem	Possible Causes	What to do
Immediately after startup, the flash drive is not recognized by the web-page interface.	During the startup routine, the system may not see a previously installed flash drive.	Remove and reinstall the flash drive.

ActivATE Software

Problem	Possible Causes	What to do
During initial configuration, the Event Logger appears with the message of “Unable to load DLL ‘visa32.dll’:The specified module is unavailable.”	The VISA runtime executive is not installed on your computer.	Obtain and load the VISA runtime executive.

Appendix C
Digital Multimeter Option

There are two factory-installed digital multimeter (DMM) options for the 1830.

- 4101: a 7½ -digit multimeter
- 4102: a 7½ -digit source/measure meter

The DMMmodule itself is located in a PXI slot inside the 1830 chassis. An interface adapter (Figure C-1) is factory-installed in one of the nine slots accessible from the rear side of the 1830 chassis. This adapter is also called the Front Panel of the DMM.

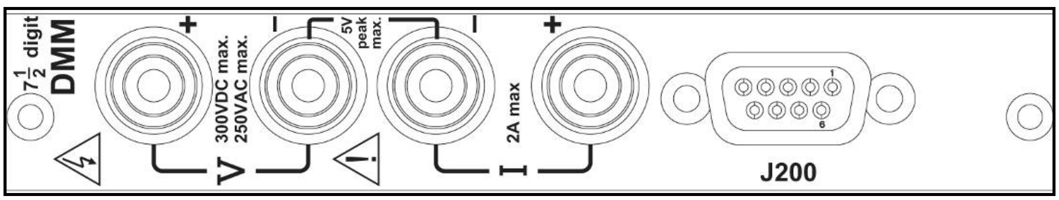
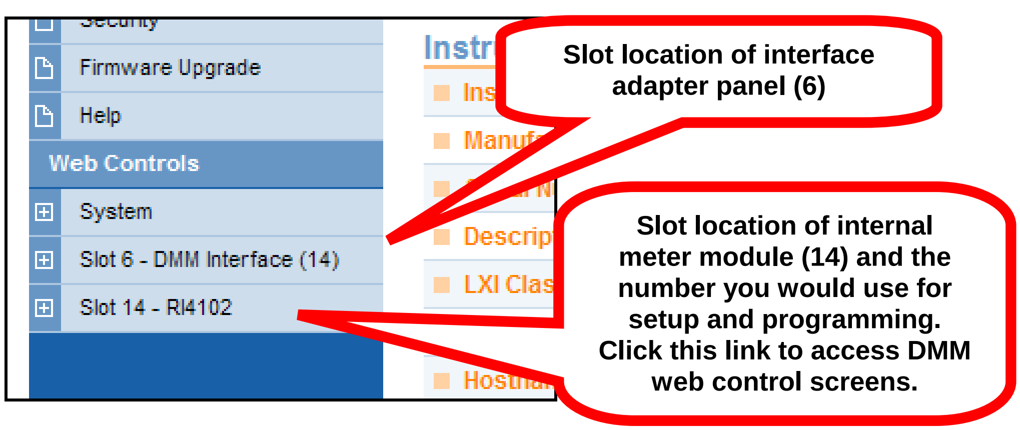


Figure C-1: 7½-Digit Multimeter Interface Adapter (Front Panel)

When a DMM module is installed, the slot number used to select the DMM is the internal PXI slot where the module resides – not the slot that the adapter panel is installed into. The slot number you use when programming with SCPI, for instance, is the internal slot number. So an adapter panel might be located in slot 6, but the DMM could have an address of slot 14 if that's where the module is installed.

The best method to verify the PXI slot number is by using the web-page interface. The main **Web Controls** screen shows two callouts when a DMM is installed. One is the position of the interface adapter panel (in the example below, slot 6). The other is the internal PXI slot number (in the example, slot 14). Click the link for the PXI slot to access the web-page controls.



Allow the DMM to warm up for at least one-half hour to meet the specified accuracy specifications.

To add Guard Source/Sense, Sync, or Trigger connections, use the J200 (9-pin, D-Sub) connector. See Figure C-3 and Table C-2 for connector pinouts.

See **Chapter 6, Using the 1830** for more information and examples regarding **Configuring and Making Measurements with the DMM**.

Caution

Do not move or remove the DMM interface card. This will compromise system safety and integrity. This card is factory serviceable only.

Analog Bus and Adapter Panel Inputs

Figure C-2 shows the relationship of the interface adapter with the internal DMM module and the connections to both the internal analog bus or the Front Panel. The channel numbers are listed in the figure's box labeled DMM Interface Adapter.

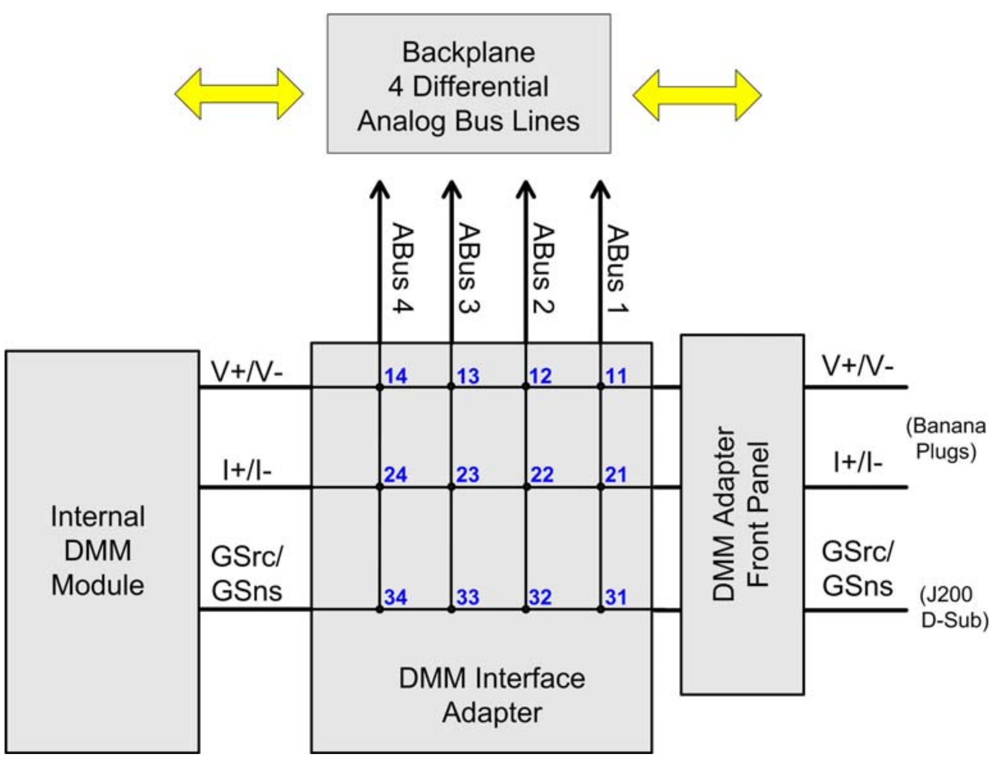


Figure C-2: DMM Operation

DMM Input Connections

V+: This is the positive terminal for all voltage, 2-wire resistance, and diode measurements. When in 4-wire resistance measurement mode, it serves as the positive terminal of the current source. The maximum input across V+ and V- is 300 VDC or 300 VAC.

V–: This is the negative terminal for all voltage, 2-wire resistance, and diode measurements. When in 4-wire resistance measurement mode, it serves as the negative terminal of the current source. **Do not float this terminal or any other DMM terminal more than 300 VDC or 300 VAC above Earth Ground.**

I+: This is the positive terminal for all current measurements. While in 4-wire resistance measurement mode, it is the high sense in the 4101 and 4102, as well as the guarded sense in the 4102. The maximum input across I+ and I– is 2.0 A. **Do not apply more than 5 V peak across the I+ and I– terminals**. While the 4102 is in DCV or DCI source mode, this terminal may be used as an additional voltage measurement input which is limited to the ±2.4V range.

I–: This is the negative terminal for current measurements. While in 4-wire resistance measurement mode, it serves as the low sense. The maximum input across I+ and I– is 2.0 A. **Do not apply more than 5 V peak across the I+ and I– terminals!** While the 4102 is in DCV or DCI source mode this terminal may be used as an additional voltage measurement input across V- or I+ which is limited to ±2.4V range.

Trig/Sync/Guard: The trigger input, sync output, and two guard signals are available at the J200 connector and the pinouts are described in Table C-2.

The Trigger can be set up to trigger reading(s) into the onboard buffer or for immediate response.

The Sync line can be used to issue or synchronize operations with an external device.

The 6-wire Guard signals facilitate in-circuit resistor measurements by means of isolating a loading node.

Notes:

- The V+/V–, I+/I–, and G+/G– signal pairs can individually connect into any differential-pair Analog Bus channel through a 2x4 matrix. To minimize thermal offset, the matrix uses latching relays which require momentarily driving the set or reset coils to move the relay position.
- The G+ (Source)/G– (Sense) pair has overvoltage fault protection circuitry and at a trip voltage of approx. 30 V resets all analog bus relays on the DMM adapter card and post an error in the event log.
- Cold junction compensation (CJC) current source is connected to the V+/V– signal pair through a relay. (For instance, close the relay and place a precision temperature diode sensor, like the National Semiconductor LM335 or similar, across V+ and V- to measure CJC temperature.)

4101 and 4102 Common Specifications

DC Voltage Measurement

Input Resistance	240 mV Range: > 10 GΩ 2.4 V Range: > 10 GΩ 24 V Range: 10 MΩ 240 V Range: 10 MΩ 300 V Range: 10 MΩ
DC Noise Rejection	For 50, 60, or 400 Hz, and apertures of 0.160 s or higher: Normal Mode Rejection >95 dB Common Mode Rejection (with 1 kΩ lead imbalance), >120 dB

AC Voltage Measurement

Input Resistance	1 MΩ, shunted by <100 pF, all ranges
Crest Factor	3 at full scale, increasing to 7 at lowest specified voltage
Frequency Range	AC Coupled: 20 Hz to 100 kHz DC Coupled: DC to 100 kHz
Typical Settling Time	<0.5 sec to within 0.1% of final value
AC Voltage Noise Rejection	Common Mode Rejection, for 50 Hz or 60 Hz with 1 kΩ imbalance in either lead, is better than 60 dB

AC Current Measurement

Crest Factor	3 at full scale, increasing to 7 at lowest specified voltage

Resistance

Offset Compensation	2-wire or 4-wire measurements Positive or negative polarity voltages Current x (Resistance + Parasitics) ≤ 0.5V

Rear Panel I/O

Banana Jacks	V+, V-: Red/Blk Banana Jacks, 300 VDC, 300 VAC max I+, I-: Red/Blk Banana Jacks, 2 Amps max, 5 Vpk max from V- to I-
J200	Auxiliary I/O: DE9F (D-sub), includes the following I/O Guard: G+, G-, 28 VDC max Sync: Sync/Common Trigger: Trig/Common

Analog Bus IO

Signal Raceway Analog Bus Connections	Channels: ABUS 1-4, 300 VDC/VAC max Path Resistance: < 1Ω @ 2 ADC Thermal EMF: < 3 µV

Other Specifications

Reading Rate (user selectable)	0.2 to 20,000 readings per second (rps) Up to 100 rps, 6½ digits Up to 1,600 rps, 5½ digits
Overload Protection (voltage inputs)	300 VDC, 300 VAC
Isolation	300 VDC, 300 VAC from Earth ground
Maximum Input (Volt x Hertz)	$8x10^6$ Volt x Hz normal mode input (across Voltage HI & LO) $1x10^6$ Volt x Hz common mode input (from Voltage Hi or Lo relative to Earth ground)

Environmental Data

Temperature	Operating: 0°C – 50°C Storage: -40°C – 71°C
Relative Humidity	80% RH at 37°C

Conformance Testing

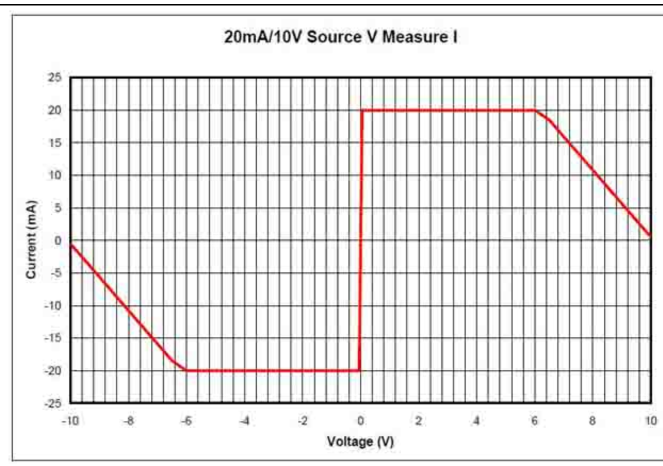
Emissions/Immunity	EN61326: 2006 Class B
Safety	Designed to IEC1010-1, Installation Category II (DMM) EN61010-1: 2001(Chassis/Extender) EN61010-1: 1993 + A2: 1995 Pollution degree 1: 300 V Pollution degree 2: 250 V Not for connection directly to mains power
Material Handling	RoHS (4101 and 4102 DMMs have WEEE Directive, Category 9 exemption.)

Reliability

Switching Time	< 10 ms (includes settling time)
Rated Switch Operations	Mechanical: 1 X 10^8 Electrical: 200,000 @ 300 Vrms/0.416 A - 125 VA
MTBF	72,511 hours (MIL-STD-217FN2)
Relay Operations Counter	Running total of operations stored in on-board non-volatile memory

4102 Specific Specifications

Source V / Measure I

Range of Operation	

Initialization	Cal: Open terminal calibration must be performed once before using this function. Set: Set DC Voltage Source ≥ 5 times to arrive at the specified voltage.

Source I / Measure V

Range of Operation	Voltage: ± 2.4 V Current: 10 nA to 12.5 mA

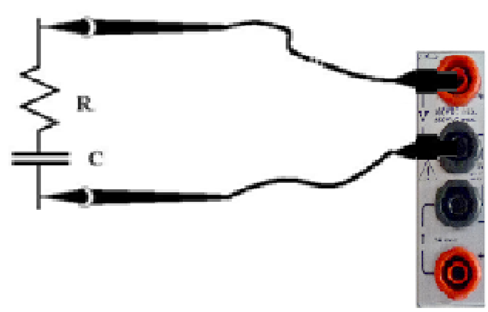
Voltage Sensing

Selectable	Source terminals or Sense inputs
Current Resolution	Normal: 500 pA to 5 μA (ranges 1-5) Using Trim DAC: 50 pA to 500 nA (ranges 1-5)

Source ACV / Measure ACV

Range of Operation	Range 1: 30 mV to 900 mVrms Range 2: 300 mV to 7.2 Vrms

ESR Function

Description	Measure a resistance that is in series with a capacitance.
Operation	Default operating voltage of 0.5 Vrms. 30 mV to 900 mVrms is available. 

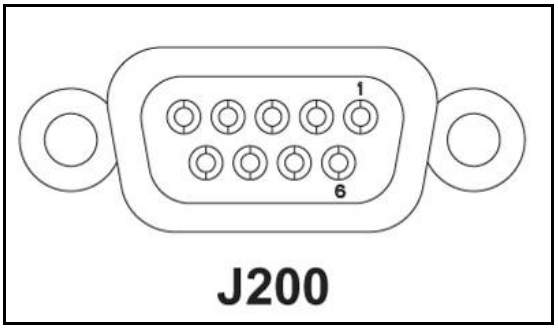


Figure C-3: DMM J200 Connector

Table C-1: DMM J200 Connector Pinout

Pin	Description
1	Guard Source (G+)
2	Sync
3	Not used
4	Trigger/Sync Common
5	Not used
6	Guard Sense (G–)
7	Trigger
8	Not used
9	Not used

4101 and 4102 DMM Features

Table C-2: 4101 and 4102 Features

Function	4101	4102
DCV five ranges 240 mV to 300 V	X	X
ACV five ranges 240 mV to 300 V	X	X
2-Wire Ohms, six ranges 240 Ω to 24 MΩ	X	X
4-Wire Ohms, six ranges 240 Ω to 24 MΩ	X	X
DC current, four ranges 2.4 mA to 2.4 A	X	X
AC current, four ranges 2.4 mA to 2.4 A	X	X
Diode V/I characteristics at 100 µA to 1mA	X	X (plus 10mA)
Auto range, Relative	X	X
On board measurement buffer	X	X
External and threshold trigger	X	X

Function	**4101**	**4102**
Thermocouples types; B, E, J, K, N, R, S, T	X	X
High Dynamic range; 24,000,000 counts	X	X
Frequency / Period measurement	X	X
Measurement rate: 0.2 to 4,500 readings/sec	X	X
Capacitance, ramp type, eight ranges, 1 nF to 10 mF	X	X
RTD types: pt385, 3911, 3916, 3926, Copper, variable Ro	X	X
Measurement rate: to 20,000/sec		X
Capacitance, In-Circuit method five ranges, 24 nF to 2.4 mF		X
Inductance, six ranges 33 μH to 3.3 H		X
Internal DMM temperature sensor		X
Offset Ohms		X
Pulse width, pos./neg., & duty cycle		X
Totalizer/event counter		X
Variable threshold DAC; all timing measure.		X
Peak to Peak, Crest factor, Median		X
Six wire Ohms (with guard source/sense)		X
DCV source to +/– 10.0 V		X
ACV source 0 to 20 V pk-pk, 0.5 Hz to 200 kHz		X
DC current source, 1 nA to 12.5 mA		X
Leakage measurement to +/– 10.0V three ranges 240 nA, 2.4 μA, 25 μA		X
2-Wire Ohms two additional ranges 24 Ω and 240 MΩ		X
4-Wire Ohms additional range 24 Ω		X
DC Current four additional ranges 240 nA, 2.4 μA, 24 μA, 240 μA		X
Pulse Generator		X
Duty-Cycle Generator		X
Two auxiliary VDC inputs		X
Stimulate and Measure load cells and strain gauges		X

4101 Measurement Accuracy

Table C-3: 4101 DMM Measurement Accuracy

(Calculated by % of reading + % of range)

Function	Range	Frequency, burden voltage, test current	24 hour 23°C ± 1°C	90 days 23°C ± 5°C	1 year 23°C ± 5°C	Temperature Coefficient 23°C ± >5°C
DC Volts [2]	240.00000 mV	10 nV resolution	0.003+0.0004	0.004+0.0006	0.005+0.0008	0.0005+0.00008
	2.4000000 V	100 nV	0.002+0.0001	0.0025+0.0002	0.003+0.0002	0.0003+0.00002
	24.000000 V	1 µV	0.004+0.0005	0.005+0.0005	0.006+0.0006	0.0006+0.00006
	240.00000 V	10 µV	0.003+0.0001	0.004+0.0001	0.005+0.0002	0.0005+0.00002
	300.00000 V	10 µV	0.0075+0.0002	0.01+0.0002	0.015+0.0003	0.0015+0.00003
True RMS AC Voltage [4,5]	240.00000 mV	10-20 Hz	3.0+0.04	3.1+0.06	3.2+0.08	0.32+0.008
	to	20-47 Hz	0.37+0.01	0.38+0.02	0.4+0.02	0.04+0.002
	240.00000 V	47Hz-10 kHz	0.13+0.05	0.14+0.05	0.15+.06	0.015+.006
		10-50 kHz	0.25+0.01	0.26+0.01	0.27+.02	0.027+.002
		50-100 kHz	1.9+0.02	1.95+0.02	2.0+0.02	0.2+0.002
	300.00000 V	10-20 Hz	3.0+0.07	3.1+0.05	3.2+0.07	0.32+0.007
		20-47 Hz	0.37+0.06	0.38+0.07	0.4+0.08	0.04+0.008
		47Hz-10 kHz	0.13+0.05	0.14+0.07	0.15+0.08	0.015+.008
		10-50 kHz	0.25+0.07	0.26+0.08	0.27+0.1	0.027+.01
		50-100 kHz	1.9+0.09	1.95+0.12	2.0+0.13	0.2+0.013
Resistance [2]	24.000000 Ω [1]	10 mA test current	0.0038+0.0058	0.005+0.0067	0.006+0.0083	0.0006+0.00083
	240.00000 Ω	1 mA	0.0037+0.0019	0.004+0.0021	0.005+0.0025	0.0005+0.00025
	2.4000000 kΩ	1 mA	0.0023+0.0012	0.0025+0.0013	0.003+0.0014	0.0003+0.00014
	24.000000 kΩ	100 µA	0.0025+0.0013	0.005+0.0014	0.006+0.0015	0.0006+0.00015
	240.00000 kΩ	10 µA	0.0055+0.0013	0.004+0.0017	0.005+0.0021	0.0005+0.00021
	2.4000000 MΩ	1 µA	0.018+0.0017	0.0025+0.0021	0.003+0.0029	0.0003+0.00029
	24.0000 MΩ	100 nA	0.12+0.0017	0.005+0.0021	0.006+0.0025	0.0006+0.00025
	240.000 MΩ [1]	10 nA	0.8+0.008	0.004+0.013	0.005+0.021	0.0005+0.0021
DC Current [3]	240.0000 nA [1]	<100 mV burden	0.07+0.017	0.1+0.019	0.17+0.025	0.017+0.0025
	2.400000 µA [1]	<100 mV	0.05+0.003	0.08+0.004	0.21+0.006	0.021+0.0006
	24.00000 µA [1]	<100 mV	0.05+0.002	0.08+0.003	0.13+0.003	0.013+0.0003
	240.000 µA [1]	<2.5 mV	0.052+0.0083	0.07+0.125	0.1+0.167	0.01+0.0167
	2.40000 mA	<25 mV	0.05+0.013	0.06+0.017	0.07+0.023	0.007+0.0023
	24.0000 mA	<250 mV	0.05+0.001	0.065+0.002	0.08+0.002	0.008+0.0002
	240.000 mA	<55 mV	0.05+0.021	0.055+0.025	0.065+0.033	0.0065+0.0033
	2.40000 A	<520 mV	0.3+0.025	0.4+0.029	0.45+0.038	0.045+0.0038
True RMS AC Current [6]	2.400000 µA	10-20 Hz	1.8+0.17	2.7+0.17	3.2+0.17	0.32+0.017
	(>60 µA)	20-47 Hz	0.9+0.17	0.9+0.17	0.4+0.17	0.04+0.017
		47Hz-1 kHz	0.04+0.063	0.08+0.13	0.15+0.17	0.015+0.017
		1-10 kHz	0.12+0.17	0.14+0.17	0.27+0.17	0.027+0.017
	24.00000 µA	10-20 Hz	1.8+0.13	2.6+0.13	2.8+0.13	0.28+0.013
	(>300 µA)	20-47 Hz	0.6+0.13	0.9+0.13	1.0+0.13	0.1+0.013
		47Hz-1 kHz	0.07+0.083	0.15+0.083	0.16+0.13	0.016+0.013
		1-10 kHz	0.21+0.17	0.3+0.17	0.4+.17	0.04+.017
	240.0000 mA	10-20 Hz	1.8+0.17	2.7+0.17	2.8+0.17	0.28+0.017
	(>3 mA)	20-47 Hz	0.6+0.17	0.9+0.17	1.0+0.17	0.1+0.017
		47Hz-1 kHz	0.1+0.042	0.17+0.075	0.2+0.092	0.02+0.0092
		1-10 kHz	0.3+0.13	0.35+0.15	0.4+0.17	0.04+0.017
	2.400000 A	10-20 Hz	1.8+0.19	2.5+0.19	2.7+0.21	0.27+0.021
	(>30 mA)	20-47 Hz	0.66+0.19	0.8+0.19	0.9+0.25	0.09+0.025
		47Hz-1 kHz	0.3+0.16	0.33+0.16	0.35+0.17	0.035+0.017
		1-10 kHz	0.4+0.19	0.45+0.19	0.5+0.21	0.05+0.021
Frequency and Period	240 mV to 300V	2 Hz - 100 Hz	0.002 + 0.004	0.002 + 0.004	0.002 + 0.004	0.0002 + 0.0004
		100 Hz-1 kHz	0.002 + 0.002	0.002 + 0.002	0.002 + 0.002	0.0002 + 0.0002
		1 kHz-10 kHz	0.002 + 0.002	0.002 + 0.002	0.002 + 0.002	0.0002 + 0.0002
		10 kHz-100 kHz	0.002 + 0.002	0.002 + 0.002	0.002 + 0.002	0.0002 + 0.0002
		100 kHz-300 kHz	0.002 + 0.0017	0.002 + 0.0017	0.002 + 0.0017	0.0002 + .00017

Notes:

1. To obtain the specified accuracies, allow 30 minutes of warm-up.
2. Accuracies are with aperture set to ≥ 0.5 s, and within one hour from self-calibration.
3. Accuracies are with aperture set to ≥ 0.96 s, and within one hour from self-calibration (relative control).6
4. Between 5 mV and 10 mV, add 100 µV of additional error.
5. Signal is limited to 8x10^6 Volt-Hz product. For example, the largest frequency input at 250 V is 32 kHz.
6. All AC current ranges have typical measurement capability of at least 20 kHz.

Table C-4: 4101 DMM Capacitance and Temperature Measurement Accuracy

(Calculated by % of reading + % of range. Accuracy is at DMM terminals.)

Function	Range or Device Type	Frequency, burden voltage, test current	Resolution	Full Scale of Range	1 year 23°C ± 5°C	Temperature Coefficient 23°C ± >5°C
Capacitance (Multi-Slope Charge Balance Method)[1,2,3]	1,200 pF	> 5% of range	0.1 pF	1,199.9 pF	1+0.083	0.1+0.0083
	12 nF		1 pF	11.999 nF	1.2+0.042	0.12+0.0042
	120 nF		10 pF	119.99 nF	1.0+0.002	0.1+0.0002
	1.2 µF		100 pF	1.1999 µF	1.0+0.0002	0.1+0.00002
	12 µF		1 nF	11.999 µF	1	0.1
	120 µF		10 nF	119.99 µF	1	0.1
	1.2 mF		100 nF	1.1999 mF	1.2	0.12
	12 mF		1 µF	50.000 mF	2	0.2
RTD Temperature[4]	pt385, pt3911, pt3916, pt3926	Ro = 100 or 200 Ω	0.01°C	-150 to 650°C	±0.06°C	±0.006°C
	pt385, pt3911, pt3916, pt3926	Ro = 500 or 1 kΩ	0.01°C	-150 to 650°C	±0.03°C	±0.003°C
	Cu (Copper)	< 12 Ω	0.01°C	-100 to 200°C	±0.18°C ≤ 20°C ±0.05°C otherwise	±0.018°C ≤ 20°C ±0.005°C otherwise
	Cu (Copper)	> 90 Ω	0.01°C	-100 to 200°C	±0.10°C ≤ 20°C ±0.05°C otherwise	±0.01°C ≤ 20°C ±0.005°C otherwise
Thermocouple Temperature[5,6]	B	0°C-50°C cold junction temperature range. Cold junction compensation is available by sensor measurement or soft entry.	0.01°C	2200°C	±0.38 °C	±0.038 °C
	E		0.01°C	1200°C	±0.035 °C	±0.0035 °C
	J		0.01°C	2000°C	±0.06 °C	±0.006 °C
	K		0.01°C	3000°C	±0.07 °C	±0.007 °C
	N		0.01°C	3000°C	±0.10 °C	±0.01 °C
	R		0.01°C	2700°C	±0.25 °C	±0.025 °C
	S		0.01°C	3500°C	±0.35 °C	±0.035 °C
	T		0.01°C	550°C	±0.06 °C	±0.006 °C

Notes:

1. Within one hour of zero, using Relative control. Specified at DMM input terminals.
2. For values between 200pf and 500pf the floor is 0.21% rather than 0.083%.
3. Accuracy is specified for values higher than 5% of the selected range.
4. Accuracies are with aperture set to ≥ 0.5 s, using a 4-wire RTD. Measurement accuracy does not include RTD probe error.
5. Accuracies are with aperture set to ≥ 0.5 s. Measurement accuracy does not include thermocouple error.
6. 4101 thermocouple temperature range may be greater than that of the thermocouple device.

4102 Measurement Accuracy

Table C-5: 4102 DMM Measurement Accuracy (Table 1)

(Calculated by % of reading + % of range)

Function	Range	Frequency, burden voltage, test current	24 hour 23°C ± 1°C	90 days 23°C ± 5°C	1 year 23°C ± 5°C	Temperature Coefficient 23°C ± >5°C
DC Volts [2]	240.00000 mV	10 nV resolution	0.003+0.0004	0.004+0.0006	0.005+0.0008	0.0005+0.00008
	2.4000000 V	100 nV	0.002+0.0001	0.0025+0.0002	0.003+0.0002	0.0003+0.00002
	24.000000 V	1 µV	0.004+0.0005	0.005+0.0005	0.006+0.0006	0.0006+0.00006
	240.00000 V	10 µV	0.003+0.0001	0.004+0.0001	0.005+0.0002	0.0005+0.00002
	300.00000 V	10 µV	0.0075+0.0002	0.01+0.0002	0.015+0.0003	0.0015+0.00003
True RMS AC Voltage [4,5]	240.00000 mV	10-20 Hz	3.0+0.04	3.1+0.06	3.2+0.08	0.32+0.008
	to	20-47 Hz	0.37+0.01	0.38+0.02	0.4+0.02	0.04+0.002
	240.00000 V	47Hz-10 kHz	0.13+0.05	0.14+0.05	0.15+.06	0.015+.006
		10-50 kHz	0.25+0.01	0.26+0.01	0.27+.02	0.027+.002
		50-100 kHz	1.9+0.02	1.95+0.02	2.0+0.02	0.2+0.002
	300.00000 V	10-20 Hz	3.0+0.07	3.1+0.05	3.2+0.07	0.32+0.007
		20-47 Hz	0.37+0.06	0.38+0.07	0.4+0.08	0.04+0.008
		47Hz-10 kHz	0.13+0.05	0.14+0.07	0.15+0.08	0.015+.008
		10-50 kHz	0.25+0.07	0.26+0.08	0.27+0.1	0.027+.01
		50-100 kHz	1.9+0.09	1.95+0.12	2.0+0.13	0.2+0.013
Resistance [1]	24.000000 Ω[1]	10 mA test current	0.0038+0.0058	0.005+0.0067	0.006+0.0083	0.0006+0.00083
	240.00000 Ω	1 mA	0.0037+0.0019	0.004+0.0021	0.005+0.0025	0.0005+0.00025
	2.4000000 kΩ	1 mA	0.0023+0.0012	0.0025+0.0013	0.003+0.0014	0.0003+0.00014
	24.000000 kΩ	100 µA	0.0025+0.0013	0.005+0.0014	0.006+0.0015	0.0006+0.00015
	240.00000 kΩ	10 µA	0.0055+0.0013	0.004+0.0017	0.005+0.0021	0.0005+0.00021
	2.4000000 MΩ	1 µA	0.018+0.0017	0.0025+0.0021	0.003+0.0029	0.0003+0.00029
	24.0000 MΩ	100 nA	0.12+0.0017	0.005+0.0021	0.006+0.0025	0.0006+0.00025
	240.000 MΩ[1]	10 nA	0.8+0.008	0.004+0.013	0.005+0.021	0.0005+0.0021
DC Current [3]	240.0000 nA[1]	<100 mV burden	0.07+0.017	0.1+0.019	0.17+0.025	0.017+0.0025
	2.400000 µA[1]	<100 mV	0.05+0.003	0.08+0.004	0.21+0.006	0.021+0.0006
	24.00000 µA[1]	<100 mV	0.05+0.002	0.08+0.003	0.13+0.003	0.013+0.0003
	240.000 µA[1]	<2.5 mV	0.052+0.0083	0.07+0.125	0.1+0.167	0.01+0.0167
	2.40000 mA	<25 mV	0.05+0.013	0.06+0.017	0.07+0.023	0.007+0.0023
	24.0000 mA	<250 mV	0.05+0.001	0.065+0.002	0.08+0.002	0.008+0.0002
	240.000 mA	<55 mV	0.05+0.021	0.055+0.025	0.065+0.033	0.0065+0.0033
	2.40000 A	<520 mV	0.3+0.025	0.4+0.029	0.45+0.038	0.045+0.0038
True RMS AC Current [6]	2.400000 µA	10-20 Hz	1.8+0.17	2.7+0.17	3.2+0.17	0.32+0.017
	(>60 µA)	20-47 Hz	0.9+0.17	0.9+0.17	0.4+0.17	0.04+0.017
		47Hz-1 kHz	0.04+0.063	0.08+0.13	0.15+0.17	0.015+0.017
		1-10 kHz	0.12+0.17	0.14+0.17	0.27+0.17	0.027+0.017
	24.00000 µA	10-20 Hz	1.8+0.13	2.6+0.13	2.8+0.13	0.28+0.013
	(>300 µA)	20-47 Hz	0.6+0.13	0.9+0.13	1.0+0.13	0.1+0.013
		47Hz-1 kHz	0.07+0.083	0.15+0.083	0.16+0.13	0.016+0.013
		1-10 kHz	0.21+0.17	0.3+0.17	0.4+.17	0.04+.017
	240.0000 mA	10-20 Hz	1.8+0.17	2.7+0.17	2.8+0.17	0.28+0.017
	(>3 mA)	20-47 Hz	0.6+0.17	0.9+0.17	1.0+0.17	0.1+0.017
		47Hz-1 kHz	0.1+0.042	0.17+0.075	0.2+0.092	0.02+0.0092
		1-10 kHz	0.3+0.13	0.35+0.15	0.4+0.17	0.04+0.017
	2.400000 A	10-20 Hz	1.8+0.19	2.5+0.19	2.7+0.21	0.27+0.021
	(>30 mA)	20-47 Hz	0.66+0.19	0.8+0.19	0.9+0.25	0.09+0.025
		47Hz-1 kHz	0.3+0.16	0.33+0.16	0.35+0.17	0.035+0.017
		1-10 kHz	0.4+0.19	0.45+0.19	0.5+0.21	0.05+0.021
Frequency and Period	240 mV to 300V	2 Hz - 100 Hz	0.002 + 0.004	0.002 + 0.004	0.002 + 0.004	0.0002 + 0.0004
		100 Hz-1 kHz	0.002 + 0.002	0.002 + 0.002	0.002 + 0.002	0.0002 + 0.0002
		1 kHz-10 kHz	0.002 + 0.002	0.002 + 0.002	0.002 + 0.002	0.0002 + 0.0002
		10 kHz-100 kHz	0.002 + 0.002	0.002 + 0.002	0.002 + 0.002	0.0002 + 0.0002
		100 kHz-300 kHz	0.002 + 0.0017	0.002 + 0.0017	0.002 + 0.0017	0.0002 + .00017

Notes:

1. To obtain the specified accuracies, allow 30 minutes of warm-up.
2. Accuracies are with aperture set to ≥ 0.5 s, and within one hour from self-calibration.
3. Accuracies are with aperture set to ≥ 0.96 s, and within one hour from self-calibration (relative control).
4. Between 5 mV and 10 mV, add 100 µV of additional error.
5. Signal is limited to 8×10^6 Volt-Hz product. For example, the largest frequency input at 250 V is 32 kHz.
6. All AC current ranges have typical measurement capability of at least 20 kHz.

Table C-6: 4102 DMM Measurement Accuracy (Table 2)

(Calculated by % of reading + % of range. Accuracy is at DMM terminals.)

Function	Range	Frequency, burden voltage, test current	Resolution	Full Scale of Range*	1 year 23°C ± 5°C	Temperature Coefficient 23°C ± >5°C
AC peak to peak Voltage	240 mV	30Hz-60 kHz	1 mV	1.9 Vp-p	0.5+1.25	0.05+0.125
	2.4 V	30Hz-60 kHz	10 mV	16 Vp-p	0.5+1.67	0.05+0.167
	24 V	30Hz-60 kHz	100 mV	190 Vp-p	0.5+2.92	0.05+0.292
	240 V	30Hz-60 kHz	1 V	850 Vp-p	0.55+2.5	0.055+0.25
AC Crest Factor[1]	240 mV	30Hz-60 kHz	0.01	1.9 Vp-p	2.2+0.3	0.22+0.03
	2.4 V	30Hz-60 kHz	0.01	16 Vp-p	2.1+0.1	0.21+0.01
	24 V	30Hz-60 kHz	0.01	190 Vp-p	2.0+0.1	0.2+0.01
	240 V	30Hz-60 kHz	0.01	700 Vp-p	2.0+0.1	0.2+0.01
	300 V	30Hz-60 kHz	0.01	848 Vp-p	2.0+0.1	0.2+0.01
AC Median Value	240 mV	30Hz-30 kHz	1 mV	±0.95 V	2+1.8	0.2+0.18
	2.4 V	30Hz-30 kHz	10 mV	±9.5 V	3+1.7	0.3+0.17
	24 V	30Hz-30 kHz	100 mV	±95.0 V	3+1.5	0.3+0.15
	240 V	30Hz-30 kHz	1 V	±300 V	3+3.43	0.3+0.343
	300 V	30Hz-30 kHz	1 V	±300 V	3+4	0.3+0.4
Resistance (6-wire Guarded)[2]	24.000000 Ω	20 mA Imax guard		24 Ω	0.306+0.025	0.0306+0.0025
	240.00000 Ω	20 mA		240 Ω	0.008+0.0855	0.0008+0.00855
	2.4000000 kΩ	20 mA		2.4 kΩ	0.008+0.0056	0.0008+0.00056
	24.000000 kΩ	100 µA		24 kΩ	0.036+0.0057	0.0036+0.00057
	240.00000 kΩ	10 µA		240 kΩ	0.355+0.0063	0.0355+0.00063
	24.0000 MΩ	1 µA		24 MΩ	0.856+0.0067	0.0856+0.00067
Resistance (Extended)[3,4,5]	400.00 kΩ	25 µA current limit	10 Ω	0.2 + 0.0125	0.33+0.0225	0.033+0.00225
	4.00000 MΩ	2.5 µA	100 Ω	0.3 + 0.0088	0.43+0.0138	0.043+0.00138
	2.4000000 kΩ	250 nA	1 kΩ	0.4 + 0.0075	0.55+0.0113	0.055+0.00113
Leakage Measurement[3]	240.0000 nA	<100 µV burden	0.07+0.017	0.1+0.019	0.17+0.025	0.017+0.0025
	2.400000 µA	<100 µV	0.05+0.003	0.08+0.004	0.21+0.006	0.021+0.0006
	24.00000 µA	<100 µV	0.05+0.002	0.08+0.003	0.13+0.003	0.013+0.0003
Trigger Level Threshold	240 mV	1.900 V max Vp-p	0.5 mV	-1.0 V to +1.0 V	0.2+0.4	0.02+0.04
	2.4 V	19.00 V	5.0 mV	-10.0 V to +10.0 V	0.2+0.4	0.02+0.04
	24 V	190.0 V	50 mV	-100.0 V to 100.0 V	0.2+0.4	0.02+0.04
	240 V	848.0 V	500 mV	-400 V to 400 V	0.2+1.0	0.02+0.1
Duty Cycle Measurement	240 mV to 300 V	2 Hz - 100 Hz	0.02%	100.00%	0.03+0.03	0.003+0.003
		100 Hz-1 kHz	0.2%	100.00%	0.03+0.3	0.003+0.03
		1 kHz-10 kHz	2%	100.00%	0.03+3	0.003+0.3
		10 kHz-100 kHz	20%	100.00%	0.03+20	0.003+2
Pulse Width	240 mV to 300 V	0.5 Hz - 100 kHz[2]	1 µs	2 µs - 1 s	0.01 + 0.004	0.001 + 0.0004
Totalize	1 to 10^9 events	Programmable trigger level and slope	1 event	1 to 30 k events per second		
Inductance[7,8]	33 µH	100 kHz test freq.	1 nH	33.000 µH	3.0+1.52	0.3+0.152
	330 µH	50 kHz	10 nH	330.00 µH	2.0+0.91	0.2+0.091

	3.3 mH	4 kHz	100 nH	3.3000 mH	1.5+0.76	0.15+0.076
	33 mH	1.5 kHz	1 µH	33.000 mH	1.5+0.61	0.15+0.061
	330 mH	1 kHz	10 µH	330.00 mH	2.5+0.91	0.25+0.091
	3.3 H	100 Hz	100 µH	3.3000 H	3+1.1	0.3+0.11
Diode I-V Test	2.4V	Test current: 100 nA to 1 mA	100 nV	0 V to 2.4 V	0.01+0.002	0.001+0.0002

Notes:

1. Repetitive signal; frequency range of 30 Hz to 60 kHz
2. Accuracies are with aperture set to ≥ 0.5 s, and within one hour from self-calibration.
3. Accuracies are with aperture set to ≥ 0.5 s, and within one hour from zero (relative control).
4. Multiply "% of reading" by 1/Voltage Source for applied voltages below 1V
5. Limit is reached when the test current exceeds the Current Limit or if it is below 0.04% of this value
6. Pulse width frequencies below 2 Hz are achieved by taking 1/ period of (positive PW + negative PW)
7. Within one hour of Zero, and Open Terminal Calibration.
8. Accuracy is specified for values greater than 5% of the selected range.

Table C-7: 4102 DMM Capacitance and Temperature Measurement Accuracy

(Calculated by % of reading + % of range. Accuracy is at DMM terminals.)

Function	Range or Device Type	Frequency, burden voltage, test current	Resolution	Full Scale of Range	1 year 23°C ± 5°C	Temperature Coefficient 23°C ± >5°C
Capacitance (Multi-Slope Charge Balance Method)[1,2,3]	1,200 pF	> 5% of range	0.1 pF	1,199.9 pF	1+0.083	0.1+0.0083
	12 nF		1 pF	11.999 nF	1.2+0.042	0.12+0.0042
	120 nF		10 pF	119.99 nF	1.0+0.002	0.1+0.0002
	1.2 µF		100 pF	1.1999 µF	1.0+0.0002	0.1+0.00002
	12 µF		1 nF	11.999 µF	1	0.1
	120 µF		10 nF	119.99 µF	1	0.1
	1.2 mF		100 nF	1.1999 mF	1.2	0.12
	12 mF		1 µF	50.000 mF	2	0.2
Capacitance (In-Circuit Method)[4,5]	24 nF	> 5% of range	10 pF	23.99 nF	5+0.83[5]	0.5+0.083
	240 nF	> 5%	100 pF	239.9 nF	5+0.42[5]	0.5+0.042
	2.4 µF	> 5%	1000 pF	2.399 µF	3+0.21[6]	0.3+0.021
	24 µF	> 5%	10 nF	23.99 µF	3+0.21[6]	0.3+0.021
	240 µF	> 5%	100 nF	239.9 µF	3+0.21[6]	0.3+0.021
	2.4 mF	> 5%	1 µF	2.399 mF	3+0.21[6]	0.3+0.021
RTD Temperature[6]	pt385, pt3911, pt3916, pt3926	Ro = 100 or 200 Ω	0.01°C	-150 to 650°C	±0.06°C	±0.006°C
	pt385, pt3911, pt3916, pt3926	Ro = 500 or 1 kΩ	0.01°C	-150 to 650°C	±0.03°C	±0.003°C
	Cu (Copper)	< 12 Ω	0.01°C	-100 to 200°C	±0.18°C ≤ 20°C ±0.05°C otherwise	±0.018°C ≤ 20°C ±0.005°C otherwise
	Cu (Copper)	> 90 Ω	0.01°C	-100 to 200°C	±0.10°C ≤ 20°C ±0.05°C otherwise	±0.01°C ≤ 20°C ±0.005°C otherwise
Thermocouple Temperature[7,8]	B	0°C-50°C cold junction temperature range. Cold junction compensation is available by sensor measurement or soft entry.	0.01°C	2200°C	±0.38 °C	±0.038 °C
	E		0.01°C	1200°C	±0.035 °C	±0.0035 °C
	J		0.01°C	2000°C	±0.06 °C	±0.006 °C
	K		0.01°C	3000°C	±0.07 °C	±0.007 °C
	N		0.01°C	3000°C	±0.10 °C	±0.01 °C
	R		0.01°C	2700°C	±0.25 °C	±0.025 °C
	S		0.01°C	3500°C	±0.35 °C	±0.035 °C
	T		0.01°C	550°C	±0.06 °C	±0.006 °C

Notes:

1. Within one hour of zero, using Relative control. Specified at DMM input terminals.
2. For values between 200pf and 500pf the floor is 0.21% rather than 0.083%.
3. Accuracy is specified for values higher than 5% of the selected range.
4. Within one hour of AC Caps Open Cal operation, and relative correction.
5. Specified for values higher than 5% of the selected range with Aperture > 0.2s.
6. Accuracies are with aperture set to ≥ 0.5 s, using a 4-wire RTD. Measurement accuracy does not include RTD probe error.
7. Accuracies are with aperture set to ≥ 0.5 s. Measurement accuracy does not include thermocouple error.
8. 4102 thermocouple temperature range may be greater than that of the thermocouple device.

Table C-8: 4102 DMM Source/Measure and Pulse Mode Accuracy

(Calculated by % of reading + % of range. Accuracy is at DMM terminals.)

Source Function/ Measure Function	Range	Source Resistance, Compliance Voltage, or Frequency Range	Current/Range	Resolution	1 year 23°C ± 5°C
Source DC Voltage/ Measure DC Voltage	± 10 V (Closed Loop)[1]	200 Ω Rsource	5 mA	18 bits	0.015 +0.0018
	± 10 V (Open Loop)	200 Ω	5 mA	12 bits	1+0.18
Source DC Voltage/ Measure DC Current	± 10 V (Source)[1,2]	Typical Settling Time: 3 s	± 10 V	5 mV	1+0.35
	± 24 mA (Measure)		± 24 mA		0.1+0.004
Source DC Current/ Measure DC Voltage	1.25 µA	4.2 V Vcompliance[4]	500 pA[5]	10 nA min. level	1+0.8
	12.5 µA	4.2 V[4]	5 nA[5]	50 nA	1+0.8
	125 µA	4.2 V[4]	50 nA[5]	100 nA	1+0.4
	1.25 mA	4.2 V[4]	500 nA[5]	1 µA	1+0.4
	12.5 mA	1.5 V[4]	5 µA[5]	10 µA	1+0.4
Source AC Voltage/ Measure AC Voltage[3]	900 mVrms (Closed Loop)	30 Hz to 200 kHz	3 mA_{rms}, typical	17 bits	0.1+0.56 +ACV spec
	900 mVrms (Open Loop)	10 Hz to 200 kHz	3 mA_{rms}, typical	12 bits	0.8+2.22 +ACV spec
	7.2 V_{rms} (Closed Loop)	30 Hz to 200 kHz	3 mA_{rms}, typical	17 bits	0.1+0.07 +ACV spec
	7.2 Vrms (Open Loop)	10 Hz to 200 kHz	3 mA_{rms}, typical	12 bits	0.8+0.28 +ACV spec
Pulse & Duty Cycle Generator	0V to +10V or 0V to -10V 5 mV resolution	PRI Range: 7.81 Hz to 20 kHz Burst Range: 1 to 32 k	Pulse width: 25 µs to 64 ms	Duty cycle/PW: 1 µs steps	

Notes:

1. An aperture of ≥ 133 ms is required for the closed loop mode.
2. See the performance envelope curve (Source V, Measure I section) for limitations to voltage and current values.
3. Aperture ≥ 166 ms is required for proper closed loop mode usage above 200 Hz. Use higher aperture time for lower frequencies.
4. Compliance voltage is the voltage range in which the current source is linear.
5. Resolution without Trim DAC. The use of the Trim DAC can improve the resolution by a factor of 10, but it has to be set separately since it is not calibrated.

DMM Quickstart Commands

Below are a few situations you may be testing with the 1830 and DMM cards. The steps to follow include the SCPI commands you would use. Refer to Figure C-2 for channel numbers.

Analog Bus #2 DC Voltage Measurement

1. Route V+/V- to Analog Bus 2.
```
CLOSE (@14(12))
```
2. Set DMM measurement function to **DC Voltage**.
```
SENSe:VOLTage (@14(0))
```
3. Query DC voltage measurement across Analog Bus 2.
```
READ? (@14(0))
```

Front Panel DC Current Measurement

1. Set DMM measurement function to **DC Current**.
```
SENSe:CURRent:DC (@14(0))
```
2. Query DC current measurement across I+ and I- terminals.
```
READ? (@14(0))
```

Front Panel 2-Wire Resistance Measurement

1. Set DMM measurement function to **2W Resistance**.
```
SENSe:RES (@14(0))
```
2. Query 2-wire resistance measurement across V+ and V- terminals.
```
READ? (@14(0))
```

Front Panel Frequency Measurement

1. Set DMM measurement function to **Temperature**.
```
SENSe:FREQuency (@14(0))
```
2. Query frequency measurement across V+ and V- terminals.
```
READ? (@14(0))
```

Cold junction Compensation Temperature Measurement

1. Connect National Semiconductor LM335 or equivalent across V+ and V-.
2. Connect CJC 1mA current source across V+ and V- lines.
```
CLOSE (@14(41))
```

3. Query DC voltage measurement.
```
MEASure:VOLTage? (@14(0))
```
4. Disconnect CJC 1mA current source.
```
OPEN (@14(41))
```
5. Convert voltage measurement to temperature in ºC using manufacturer's datasheet.
6. Set cold junction compensation temperature (for example, to 20ºC).
```
SENSe:TEMPerature:TRANsducer:TCouple:RJUNCtion 20, (@14(0))
```

Front Panel T-type Thermocouple Measurement

1. Set DMM measurement function to **Temperature**.
```
SENSe:TEMPerature (@14(0))
```
2. Set temperature sensor type to thermocouple.
```
SENSe:TEMPerature:TRANsducer:TYPE TCouple, (@14(0))
```
3. Set thermocouple type to T.
```
SENSe:TEMPerature:TRANsducer:TCouple:TYPE T, (@14(0))
```
4. Set temperature measurement unit to ºC.
```
SENSe:TEMPerature:UNIT C, (@14(0))
```
5. Query thermocouple temperature measurement across the V+ and V- terminals.
```
READ? (@14(0))
```

DC Capacitive Measurement

1. Set DMM measurement function to **DC Capacitance**.
```
SENSe:CAPacitance (@14(0))
```
2. Query DC capacitance measurement across the V+ and V- terminals.
```
READ? (@14(0))
```

Set Trigger to External Input

```
TRIGger:MODE EXTernal, (@14(0))
INITiate (@14(0))
```

AC Capacitive Measurement (4102 option only)

1. Set DMM measurement function to **AC Capacitance**.
```
SENSe:CAPacitance:AC (@14(0))
```
2. Query AC capacitance measurement across the V+ and V- terminals.
```
READ? (@14(0))
```

Analog Bus #1 to #3 6-Wire Resistance Measurement (4102 option only)

1. Route V+/V- to Analog Bus 1.
```
CLOSE (@14(11))
```
2. Route I+/I- to Analog Bus 2.
```
CLOSE (@14(22))
```
3. Route G+/G- to Analog Bus 3.
```
CLOSE (@14(33))
```
4. Set DMM measurement function to **4W Resistance**. DMM will automatically take 6W measurement when Guard Source and Sense are used.
```
SENSe:FRES (@14(0))
```
5. Query 6-wire resistance measurement.
```
READ? (@14(0))
```

Front Panel Resistance Temperature Detector (RTD) Measurement (4102 option only)

1. Set DMM measurement function to **Temperature**.
```
SENSe:TEMPerature (@14(0))
```
2. Set temperature sensor type to RTD.
```
SENSe:TEMPerature:TRANsducer:TYPE RTD, (@14(0))
```
3. Set RTD type to 385.
```
SENSe:TEMPerature:TRANsducer:RTD:TYPE 385, @14(0))
```
4. Set RTD ice point resistance to 1000 ohms.
```
SENSe:TEMPerature:TRANsducer:RTD:R0 1000, (@14(0))
```
5. Query RTD temperature measurement across the V+ and V- terminals.
```
READ? (@14(0))
```

Inductive Measurement (4102 option only)

1. Set DMM measurement function to **Inductance**.
```
SENSe:INDuctance (@14(0))
```
2. Query capacitance measurement across the V+ and V- terminals.
```
READ? (@14(0))
```

DC Voltage Source (4102 option only)

Source 1.0VDC across the V+ and V- terminals.
```
SOURce:VOLTage:DC 1.0, (@14(0))
```

DC Current Source (4102 option only)

Source 10uADC across the V+ and V- terminals.

```
SOURce:CURRent:DC 10e-6, (@14(0))
```

Single-run Pulse Generator (4102 option only)

Generate a single-run 5.0V, 25us pulse across the V+ and V- terminals.

```
SOURce:PULSe 5.0,20000,50,1 (@14(0))
```

Free-running Pulse Generator (4102 option only)

Generate a free-running 5.0V, 20kHz squarewave across V+ and V- terminals.

```
SOURce:PULSe 5.0,20000 (@14(0))
```

SCPI Commands

A full list of SCPI commands are located in the **DMM-Specific Command Summary** section of **Chapter 7, SCPI Command Basics**. A list of commands supported by the 4102 option but not the 4101 option are noted in that section.

Web-Page Controls

The web-page interface require you to have a recent version of the Java runtime engine installed on your computer

The web controls allow you to view the status and modify a variety of conditions and parameters in making your setup and measurements.

Refer to the following subsections for additional information on the general features of each page.

Main Control Screen

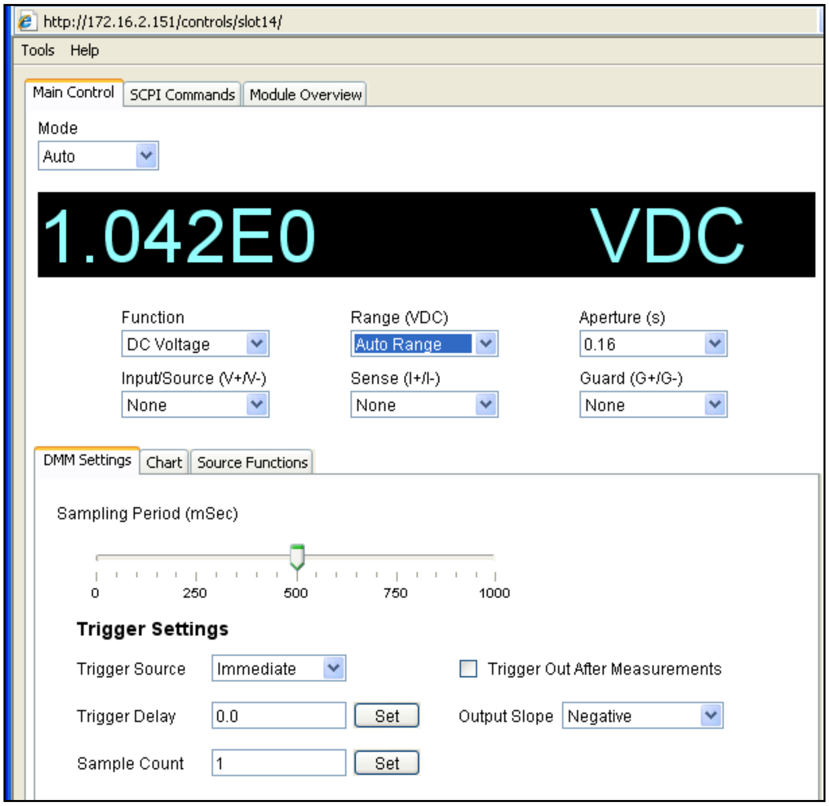


Figure C-4: DMM Main Control Screen

From this screen (Figure C-4), you can set the DMM to read from the Front Panel or the Analog Bus by selecting the appropriate option button.

You can set various operations and measurements including:

- Mode – auto, single, or triggered
- Function – voltage, current, frequency, period, resistance
- Range
- Aperture
- Input source – none (front panel) or selected analog bus
- Sense location – none (front panel) or selected analog bus
- Guard location (4102 option only) – none (front panel) or selected analog bus

With the tabbed features on the lower section of this screen, you can also set various DMM settings, chart the data, or check the source functions (4102 option only).

If you select an analog bus as the input source, be sure to remove any front panel connections that you don't want influencing your DMM readings.

Modes

When you select the **Single Mode**, a **Read** button appears to the right to allow you to make a single measurement when the button is clicked.

When you select the **Triggered Mode**, the trigger settings on the **DMM Settings** tab such as the Source, Delay, Sample Count, and Output Slope determine the measurement occurrence.

When you select the **Auto Mode**, the sampling period on the **DMM Settings** tab determines the measurement interval.

Chart Tab

When **Plot Measurements** is selected on the **Chart** tab (Figure C-5), the system charts your measurement points. Click **Clear Chart** to clear the current chart to either remove the information shown or to restart the charting.

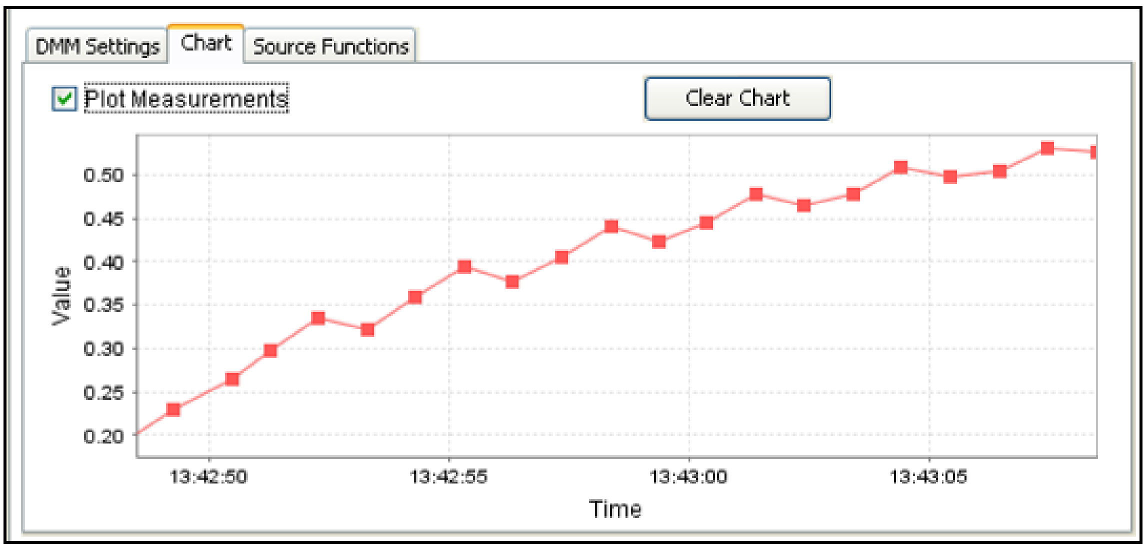


Figure C-5: DMM Main Control - Chart Tab

Source Functions Tab (4102 option only)

From the **Source Functions** tab (Figure C-6), you can set the source output on the DMM including voltage, current, and pulse attributes.

After you have changed the values, click **Set** to implement them on the DMM.

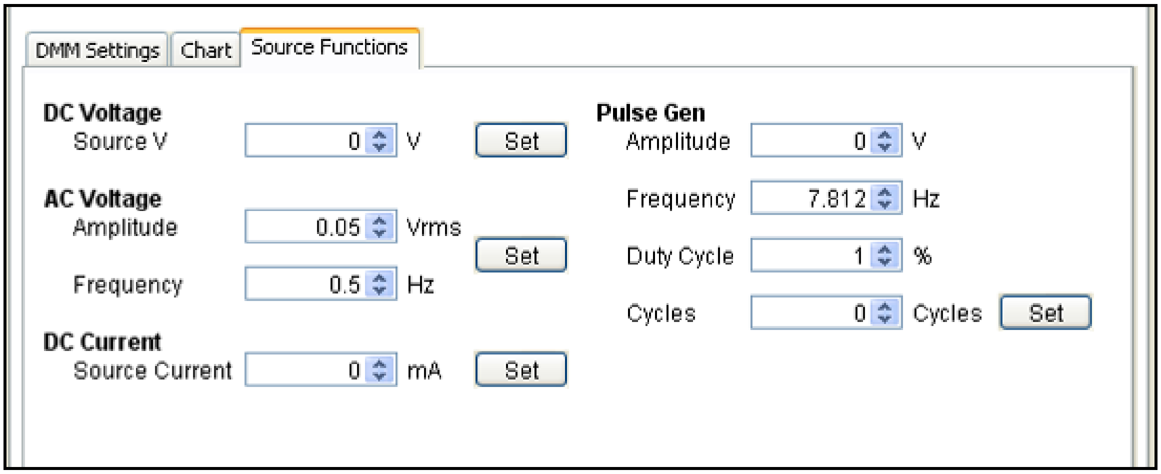


Figure C-6: DMM Main Control – Source Functions Tab

SCPI Commands Tab

The **SCPI Commands** tab (Figure C-7) allows you to send SCPI commands to the DMM. A few common SCPI commands have clickable icons along the top of the tab including:

- **SYST:ERR?:** Checks on and reports if there are any errors
- **Read STB:** Reads the value of the Status Byte Register
- **Device Clear:** Clears error queue and flushes all input and output buffers
- **Clear:** Clears any text in the text message box below the icons

Use the **SCPI Command** input box to enter SCPI commands. The command and any replies will be shown in the text message box.

- Clicking **Send** sends the command.
- Clicking **Read** receives the reply.
- Clicking **Send/Read** both sends and then receives the reply.
- Clicking **Clear** clears the commands history in the **SCPI command** input box

The **Time Out** box allows you to set different values as needed.

The **SCPI command list** drop-down box includes a list of commonly used commands and queries that you can choose to automatically appear in the **SCPI command** input box.

Additional information and a listing of SCPI commands can be found in **Chapter 7, SCPI Command Basics**.

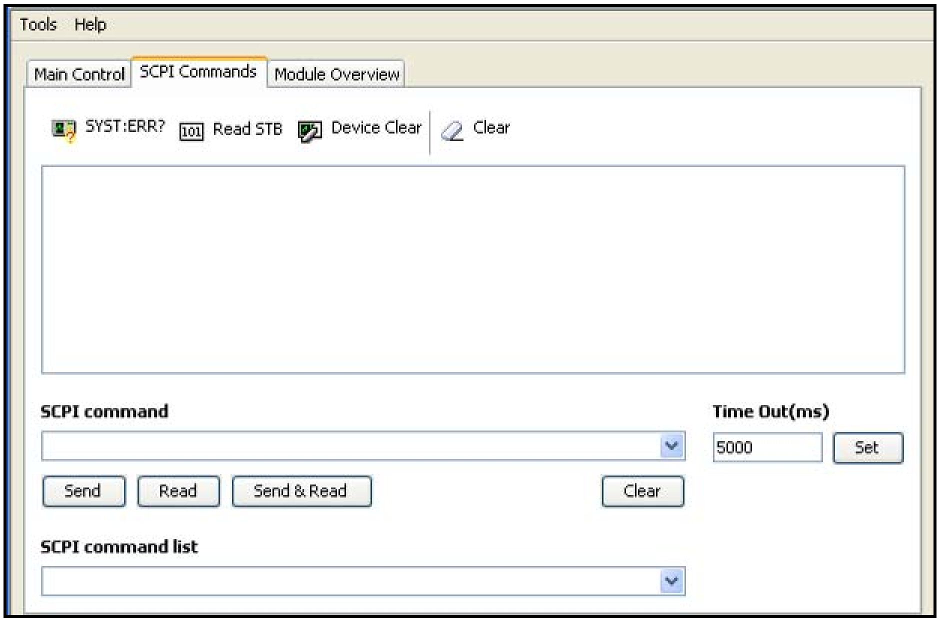


Figure C-7: DMM SCPI Commands Tab

Module Overview Tab

The **Module Overview** tab (Figure C-8) allows you to review current DMM information including model and serial number, revision levels, and emergency reset status.

Clicking **Generate Report** generates an informational report and shows the results on the information screen to the right of the button. Clicking **Clear Report** removes the information from the screen.

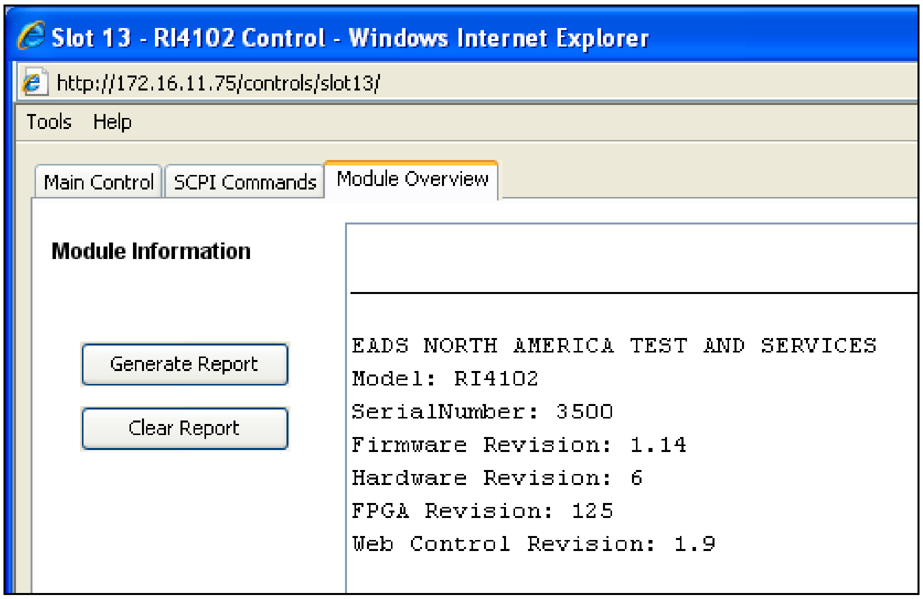


Figure C-8: DMM Module Overview Tab

ActivATE Control Tabs

The following screen tabs appear when you are using ActivATE to control the DMM. They offer the same basic functions as the individual Web page screens described in the previous section.

One additional feature is the addition of four buttons along the bottom of the ActivATE screens:

- The **Reset** button resets the DMM operating conditions back to automatic measuring and DC volts.
- The **Read** button reads the current measurement and displays it on the screen.
- The **Update** button updates the screen you're looking at with the current status or configuration.
- The **Help** button brings up a Help screen for the DMM

When the **Simulation** box is selected, you can change any of the functions and configurations shown on the screens without changing the actual instrument itself. Be sure to deselect the Simulation mode when you are ready to set up and measure using the DMM.

Main

The **Main** tab (Figure C-9) allows full control of the DMM including setting:

- Measurement function (DC, AC, 2-wire resistance, 4-wire resistance, DC current, AC current, period, frequency, capacitance, diode, and temperature)
- Range
- Aperture
- Additional Modes (extended 2-wire resistance, AC capacitance, current leakage, inductance, or duty cycle
- Sampling Period
- Triggering (mode and threshold voltage)

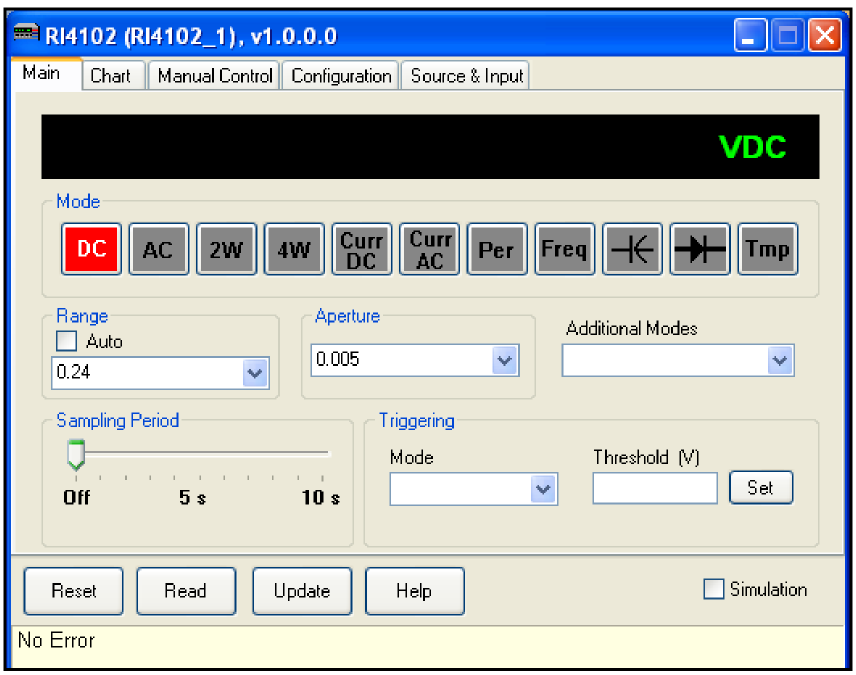


Figure C-9: ActivATE Main Tab

Chart

When **Plot Measurements** is selected on the **Chart** tab (Figure C-10), the system charts your measurement points.

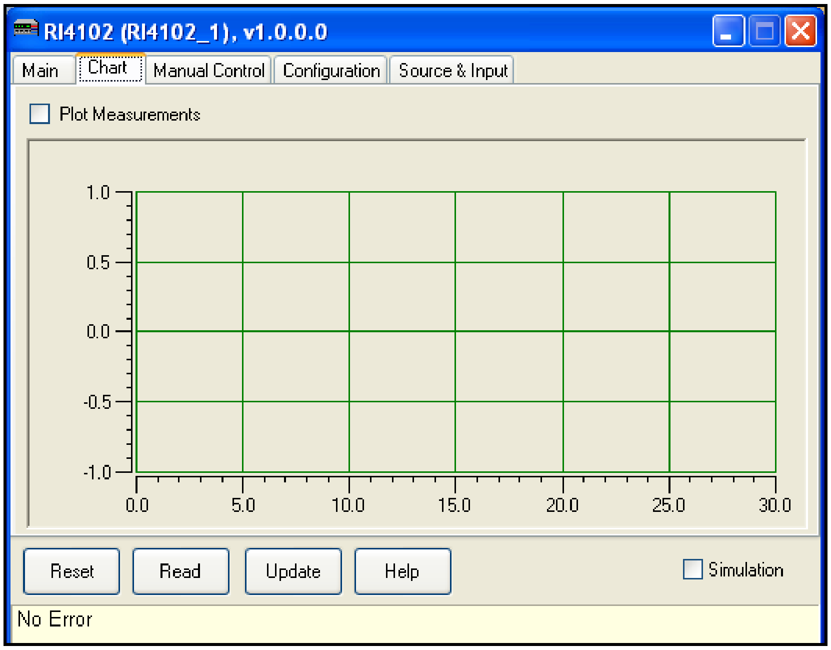


Figure C-10: ActivATE Chart Tab

Manual Control

The **Manual Control** tab (Figure C-11) allows you to manually enter SCPI commands and queries in their appropriate input box. Click **Write** when entering commands and **Query** when entering queries. Click **Read** to read the results from the instrument.

Additional information on and a listing of SCPI commands can be found in **Chapter 7, SCPI Command Basics**.

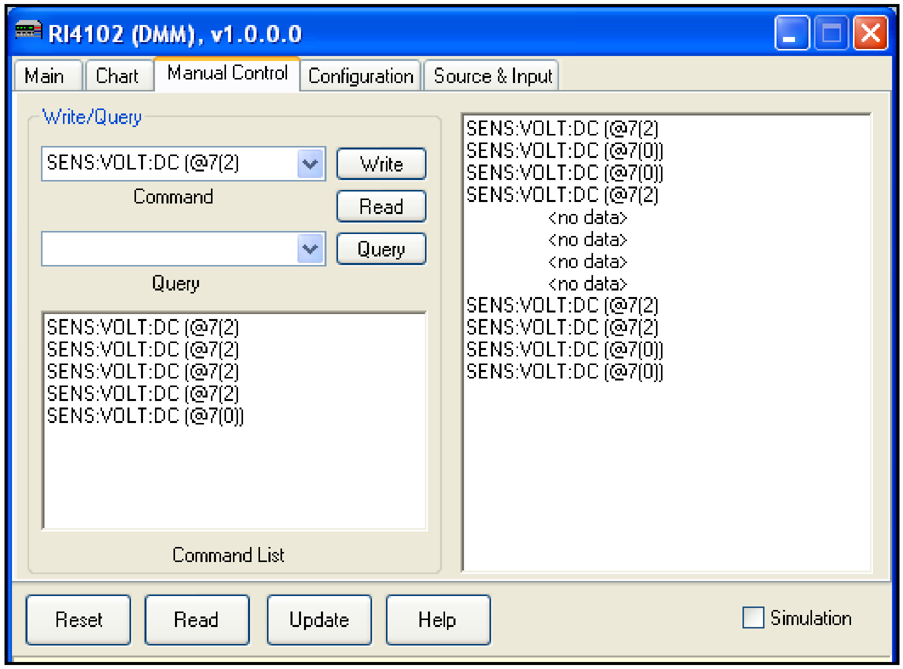


Figure C-11: ActivATE Manual Control Tab

Configuration

The **Configuration** tab (Figure C-12) allows you to load a pre-written command file to the DMM. Click on the "**…**" button to view folders and files on your computer.

Click **Enable File Trace,** if desired, to help identify errors in your file.

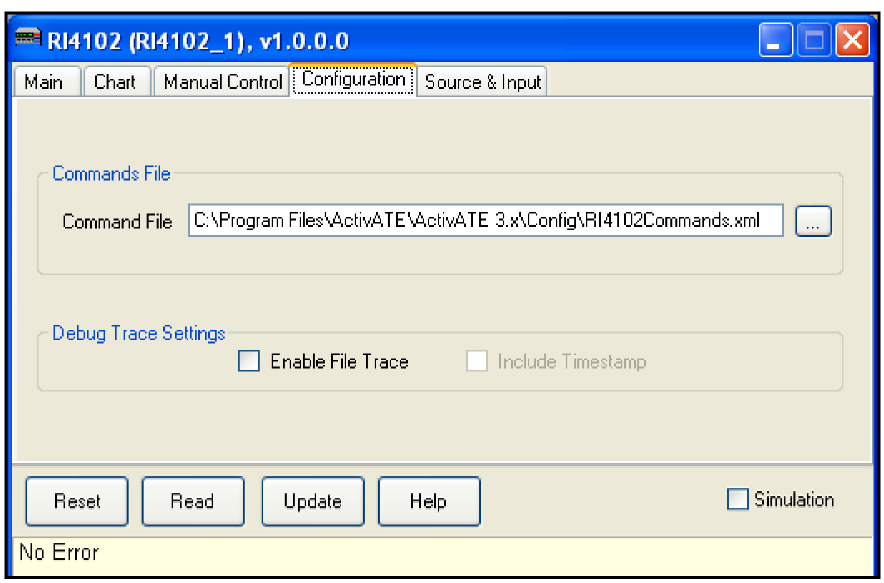


Figure C-12: ActivATE Configuration Tab

Source & Input

From the **Source & Input** tab (Figure C-13), you can set the source output on the DMM including voltage, current, and pulse attributes. You can also set input information from either the Front Panel or one of the internal Analog busses.

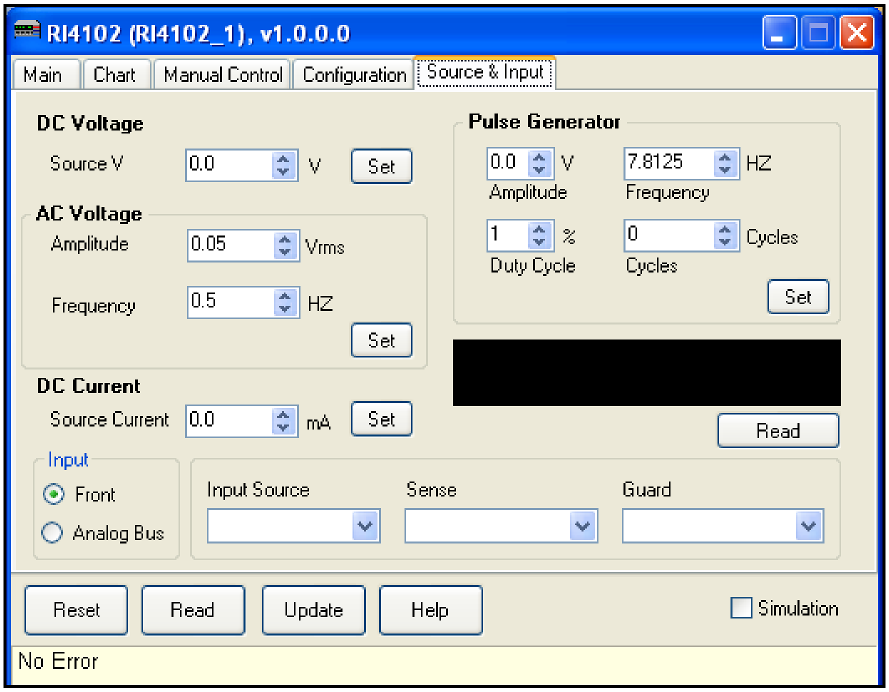


Figure C-13: ActivATE Source & Input Tab

Appendix D
Rack-Mount Installation

Items Included

The rack-mount kit, PN 408174 is used to mount the 1830 into a standard 19-inch equipment rack and includes the following:

Table D-1: Rack Mount Kit (PN 408174) Parts List

Item	Description	Qty
1	Rack-mount ear	2
2	6 x 32 x ¾" pan-head screw	4

Not Included – right angle runners (or a base plate) and the rack-mounting screws.

Installing the Rack-Mount Ears

1. Remove the rubber feet on the bottom of the 1830 by removing the screws that secure them.
2. Secure the rack-mount ear to the 1830 chassis as shown in Figure D-1, using the supplied mounting screws.
3. If not already installed in your equipment rack, install right-angle runners or a base plate.
4. Slide the 1830 chassis into the rack. (See Figure D-2.)
5. Fasten the 1830 chassis to the rack. (Screws for mounting to your rack frame are not supplied.)

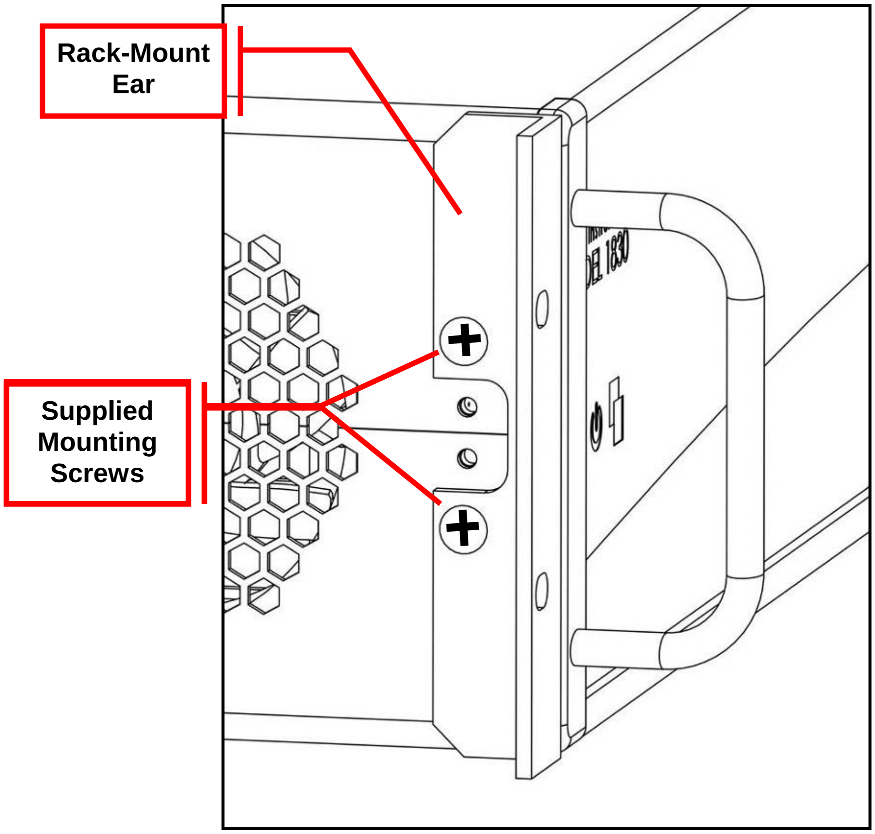


Figure D-1: Attaching Rack Mount Ears

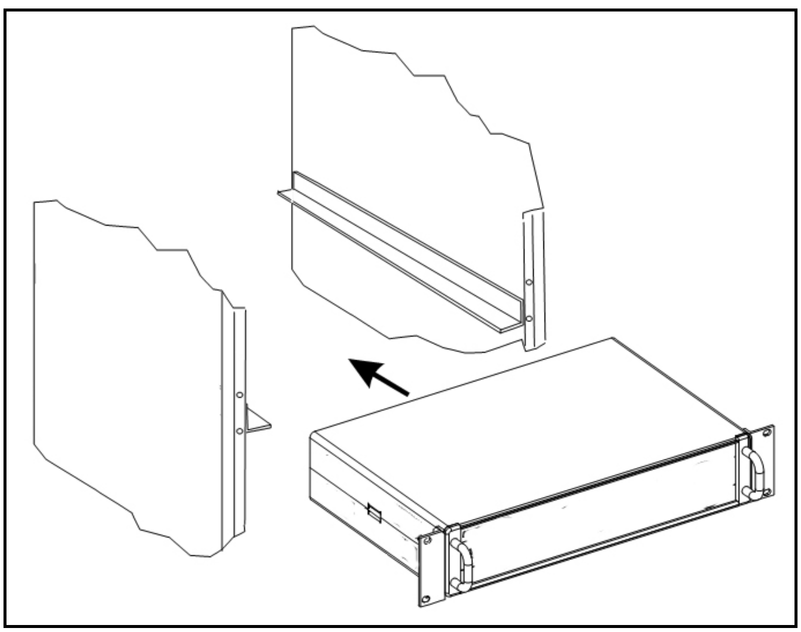


Figure D-2: Installing the 1830 into the 19-Inch Equipment Rack

INDEX