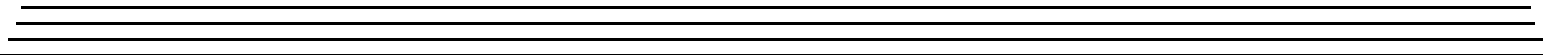
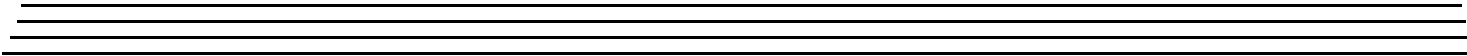
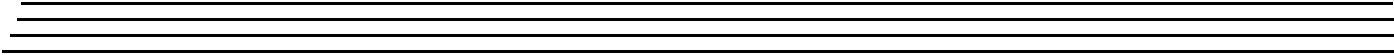




UM-24017-C

**SCPI**  
**Programmer's Manual**  
**for the DT8824**



## Trademark and Copyright Information

Measurement Computing Corporation, InstaCal, Universal Library, and the Measurement Computing logo are either trademarks or registered trademarks of Measurement Computing Corporation. Refer to the Copyrights & Trademarks section on [mccdaq.com/legal](http://mccdaq.com/legal) for more information about Measurement Computing trademarks. Other product and company names mentioned herein are trademarks or trade names of their respective companies.

© 2012 Measurement Computing Corporation. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form by any means, electronic, mechanical, by photocopying, recording, or otherwise without the prior written permission of Measurement Computing Corporation.

### Notice

Measurement Computing Corporation does not authorize any Measurement Computing Corporation product for use in life support systems and/or devices without prior written consent from Measurement Computing Corporation. Life support devices/systems are devices or systems that, a) are intended for surgical implantation into the body, or b) support or sustain life and whose failure to perform can be reasonably expected to result in injury. Measurement Computing Corporation products are not designed with the components required, and are not subject to the testing required to ensure a level of reliability suitable for the treatment and diagnosis of people.

---

# Table of Contents

<b>About this Manual</b> .....	<b>7</b>
Intended Audience .....	7
What You Should Learn from this Manual .....	7
Conventions Used in this Manual .....	8
Related Documents .....	8
Where to Get Help .....	8
<b>Chapter 1: Syntax Conventions</b> .....	<b>9</b>
Introduction .....	10
Types of SCPI Messages .....	10
Types of SCPI Commands .....	10
Common SCPI Commands .....	10
SCPI Subsystem Commands .....	12
Syntax of Program Messages .....	16
Syntax of Common SCPI Commands .....	17
Syntax of SCPI Subsystem Commands .....	17
About Short- and Long-Form Mnemonics .....	18
About Brackets, Braces, and Vertical Bars .....	18
Syntax of Response Messages .....	19
SCPI Data Types .....	20
Character Data Types .....	20
String Data Types .....	20
<NR1>Value Data Type .....	21
<NR2> Value Data Type .....	21
<NRr>Value Data Type .....	21
<NRf>Value Data Type .....	22
<Boolean> Data Type .....	23
<Block> Data Type .....	23
SCPI Expression Types .....	25
Numeric Expressions .....	25
Channel Lists .....	26
Numeric Lists .....	27
Data Interchange Format (DIF) Expressions .....	28
Instrument-Specifier Expressions .....	29
<b>Chapter 2: Using SCPI Commands with DT8824 Instrument Modules</b> .....	<b>31</b>
Installing SCPI Support Files for the DT8824 .....	32
Configuring the LAN Settings of the DT8824 .....	33
Setting the Static IP and Subnet Mask of the Instrument Module .....	33

Specifying a Description of the Instrument Module .....	33
Getting Information about the Instrument Module.....	34
Using Password-Protected Commands .....	35
Performing Input Operations .....	38
Enabling and Configuring Analog Input Channels.....	38
Configuring the Input Range and Gain.....	38
Configuring the Input Buffer.....	39
Configuring the Input Clock Source .....	40
Configuring the Input Trigger.....	42
Software Trigger .....	42
External, Digital (TTL) Trigger .....	43
LAN Trigger Packet .....	43
Trigger Bus .....	45
Arming Input Operations .....	47
Starting Input Operations .....	47
Determining the Status of the Analog Input Subsystem.....	47
Retrieving Scan Data from the Input Buffer .....	48
Stopping Input Operations .....	50
Performing Digital Output Operations.....	51
<b>Chapter 3: Common SCPI Commands .....</b>	<b>53</b>
<b>Chapter 4: SCPI Subsystem Commands for the DT8824.....</b>	<b>63</b>
SCPI Subsystem Command Hierarchy .....	64
STATus Subsystem Commands .....	66
SYSTEM Subsystem Commands.....	72
AD Subsystem Commands.....	90
WTB (Wired Trigger Bus) Subsystem Commands.....	116
EVENT Subsystem Command.....	120
DOUTput Subsystem Commands.....	127
<b>Chapter 5: Programming Flowcharts.....</b>	<b>133</b>
Using Password Protected Command .....	134
Configuring the Instrument Module on the LAN.....	135
Performing Input Operations .....	136
Performing Digital Output Operations.....	137
<b>Chapter 6: Product Support .....</b>	<b>141</b>

---

<b>Appendix A: Errors</b> .....	<b>143</b>
Error Codes .....	144
Troubleshooting Errors .....	146
Error -110 Command Header Error .....	146
Error -410 Query Interrupted .....	147
<b>Appendix B: Registers</b> .....	<b>149</b>
Status Byte Register (STB) .....	150
Standard Event Status Enable Register (ESE) .....	151
Standard Event Status Register (ESR) .....	153
Operation Status Register .....	154
<b>Appendix C: Examples</b> .....	<b>155</b>
Features of the DT8824 SCPI Example Program .....	157
Opening and Running the DT8824 SCPI Example Program .....	158
<b>Appendix D: Using HyperTerminal to Send and Receive SCPI Commands</b> .....	<b>159</b>
<b>Index</b> .....	<b>165</b>



# About this Manual

This manual describes how to use (Standard Commands for Programmable Instruments (SCPI) to communicate with the DT8824 LXI (LAN eXtensions for Instrumentation) instrument modules.

## Intended Audience

This document is intended for instrument programmers who are responsible for writing SCPI-based programs for DT8824 LXI instrument modules.

## What You Should Learn from this Manual

This manual provides detailed information about the SCPI commands that are available for communicating with DT8824 LXI instrument modules. This manual is organized as follows:

- [Chapter 1, “Syntax Conventions,”](#) describes the syntax conventions used.
- [Chapter 2, “Using SCPI Commands with DT8824 Instrument Modules,”](#) provides an introduction to SCPI commands.
- [Chapter 3, “Common SCPI Commands,”](#) describes the common SCPI commands that are available for DT8824 LXI instrument modules, including the command syntax, functional description, examples, and so on.
- [Chapter 4, “SCPI Subsystem Commands for the DT8824,”](#) describes the device-specific SCPI commands that available for DT8824 LXI instrument modules, including the command syntax, functional description, examples, and so on.
- [Chapter 5, “Programming Flowcharts,”](#) provides flow diagrams that show how to use the SCPI commands together to write a program that communicates with DT8824 LXI instrument modules.
- [Appendix A, “Errors,”](#) lists the errors that can be returned and describes how to troubleshoot frequently occurring errors.
- [Appendix B, “Registers,”](#) describes the registers that are used by SCPI commands.
- [Appendix C, “Examples,”](#) describes example applications that illustrate how to use SCPI commands to program DT8824 LXI instrument modules.
- [Appendix D, “Using HyperTerminal to Send and Receive SCPI Commands,”](#) describes how to use HyperTerminal to send and receive SCPI commands.
- An index completes this manual.

## Conventions Used in this Manual

The following conventions are used in this manual:

- Notes provide useful information or information that requires special emphasis, cautions provide information to help you avoid losing data or damaging your equipment, and warnings provide information to help you avoid catastrophic damage to yourself or your equipment.
- Items that you select or type are shown in **bold**.

## Related Documents

Refer to the following documents for more information:

- *DT8824 User's Manual* (UM-23780). This manual describes the operation of the DT8824 instrument module.
- Standard Commands for Programmable Instruments (SCPI), Volume 1-4, Version 1999.0 May 1999, SCPI Consortium, 2515 Camino del Rio South, Suite 340, San Diego, CA 92108.
- IEEE Std 488.2-1992, The Institute of Electrical and Electronics Engineers, Inc., 345 East 47th Street, New York, NY 10017-2394, USA (ISBN 1-55937-238-9)

## Where to Get Help

Should you run into problems installing or using SCPI commands to communicate with DT8824 LXI instrument modules, the Data Translation Technical Support Department is available to provide technical assistance. Refer to [Chapter 6](#) for more information. If you are outside the United States or Canada, call your local distributor, whose number is listed on Data Translation's web site ([www.mccdaq.com](http://www.mccdaq.com)).





# ***Syntax Conventions***

Introduction.....	10
Syntax of Program Messages .....	16
Syntax of Response Messages.....	19
SCPI Data Types .....	20
SCPI Expression Types .....	25

## Introduction

SCPI (Standard Commands for Programmable Instruments) is a universal programming language for electronic test and measurement instruments, based on the IEEE 488.1 and IEEE 488.2 standards. DT8824 LXI instrument modules comply with the SCPI language and implement the IEEE-488.2 STD status structure.

You can issue these commands over VISA or sockets using TCP port 5025. Refer to [Appendix D](#) starting on [page 159](#) for information on using HyperTerminal to send and receive SCPI commands over TCP port 5025.

## Types of SCPI Messages

To program a DT8824 LXI instrument module, you create program messages. A program message consists of one or more properly formatted SCPI commands sent from the controller to the DT8824 LXI instrument module. The program message, which may be sent at any time, requests that the instrument perform some action or send back data or status information; these requests are also called queries.

When queried, the DT8824 LXI instrument module sends a response message back to the controller. The response message consists of data in a specific SCPI format.

Refer to [page 16](#) for more information on the syntax of program messages and to [page 19](#) for more information on the syntax of response messages.

The following documents provide more information about SCPI programming:

- Standard Commands for Programmable Instruments (SCPI), Volume 1-4, Version 1999.0 May 1999, SCPI Consortium, 2515 Camino del Rio South, Suite 340, San Diego, CA 92108.
- IEEE Std 488.2-1992, The Institute of Electrical and Electronics Engineers, Inc., 345 East 47th Street, New York, NY 10017-2394, USA (ISBN 1-55937-238-9)

## Types of SCPI Commands

Two types of SCPI commands are available: common commands, described below, and device-specific subsystem commands, described on [page 12](#). DT8824 LXI instrument modules respond to all of the required IEEE-488.2 common commands and support subsystem commands for measuring data and performing other device-specific functions.

### **Common SCPI Commands**

Common SCPI commands, defined by IEEE488.2, control and manage generic system functions such as reset, self-test, configuration storage, and device identification. [Table 1](#) summarizes the common SCPI commands that are available for programming DT8824 LXI instrument modules. Refer to [Chapter 3](#) starting on [page 53](#) for a detailed description of these commands.

**Table 1: Common SCPI Commands for Programming DT8824 LXI Instrument Modules**

Type	Mnemonic	Description
Clear Status	*CLS	Clears all event registers summarized in the Status Byte (STB) register, described on <a href="#">page 150</a> .
Event Status Enable Register	*ESE	Enables specified bits in the Standard Event Status Enable register, described on <a href="#">page 151</a> .
Event Status Enable Register Query	*ESE?	Returns the current value of the Standard Event Status Enable register, described on <a href="#">page 151</a> .
Event Status Register Query	*ESR?	Returns the current value of the Standard Event Status register, described on <a href="#">page 153</a> , and then clears the register.
Identification Query	*IDN?	Returns the unique identity of the DT8824 LXI instrument module.
Operation Complete	*OPC	The Operation Complete bit (bit 0) of the Standard Event Status register, described on <a href="#">page 151</a> , is always enabled. Therefore, this command has no effect when used with the DT8824 LXI instrument module.
Operation Complete Query	*OPC?	The Operation Complete bit (bit 0) of the Standard Event Status register, described on <a href="#">page 151</a> , is always enabled. Therefore, this command always places the ASCII character 1 into the device's Output Queue.
Reset	*RST	Clears the Standard Event Status register, message queue, error queue, and Status Byte register, and stops any scans that are in progress.
Self-Test Query	*TST?	Always returns 0 for DT8824 LXI instrument modules.
Service Request Enable	*SRE	The Service Request Enable register is not used on these instrument modules. Therefore, this command has no effect when used with DT8824 LXI instrument modules.
Service Request Enable Query	*SRE?	The Service Request Enable register is not used on these instrument modules. Therefore, this command has no effect when used with DT8824 LXI instrument modules.
Status Byte Query	*STB?	Returns the current value of the Status Byte register, described on <a href="#">page 150</a> .
Wait	*WAI	Has no effect on DT8824 LXI instrument modules.

## **SCPI Subsystem Commands**

SCPI subsystem commands are either measurement-related or other device-specific commands for programming DT8824 LXI instrument modules.

The following subsystems are available on DT8824 LXI instrument modules:

- **STATus** – The STATus subsystem includes commands that are related to the operational status of a DT8824 LXI instrument module. This subsystem is mandatory for SCPI-compliant devices.
- **SYSTEM** – The SYSTEM subsystem includes commands for querying errors returned by the instrument module, and configuring and querying system settings, including the LAN configuration of the instrument module.
- **AD** – The AD subsystem includes commands and queries for setting up the analog input features of the instrument module, including the clock, trigger, buffer, and gain settings, arming the analog input subsystem, starting the analog input operation, stopping the analog input operation, getting status information, and getting data from the input buffer on the DT8824 instrument module.
- **WTB** – The WTB subsystem includes commands and queries for configuring the Wired Trigger Bus.
- **EVENT** – The EVENT subsystem includes commands and queries for configuring the LAN events.
- **DOUtput** – The DOUtput subsystem includes commands for setting the state of the digital output lines on the instrument module and a query for returning the state of the digital output lines on the instrument module.

[Table 2](#) summarizes the SCPI commands and queries available in each subsystem for programming DT8824 LXI instrument modules. Refer to [Chapter 4](#) starting on [page 63](#) for a detailed description of these commands.

**Table 2: Subsystem SCPI Commands for Programming DT8824 LXI Instrument Modules**

Subsystem	Mnemonic	Description
STATus	STATus:OPERation[:EVENT]?	Has no effect on DT8824 LXI instrument modules.
	STATus:OPERation:CONDition?	Returns the current value of the Operation Condition register, described on <a href="#">page 154</a> .
	STATus:OPERation:ENABle	Has no effect on DT8824 LXI instrument modules.
	STATus:OPERation: ENABle?	Always returns 0 for DT8824 LXI instrument modules.
	STATus:PRESet	Has no effect on DT8824 LXI instrument modules.
	STATus:QUEStionable[:EVENT]?	Always returns 0 for DT8824 LXI instrument modules.
	STATus:QUEStionable:CONDition?	Always returns 0 for DT8824 LXI instrument modules.
	STATus:QUEStionable:ENABle	Has no effect on DT8824 LXI instrument modules.
	STATus:QUEStionable:ENABle?	Always returns 0 for DT8824 LXI instrument modules.
	SYSTem:DATE?	Returns the current date of the DT8824 LXI instrument module. This date is updated automatically by an SNTP (Simple Network Time Protocol) server
	SYSTem:DESCRiption	Specifies a description for the device.
	SYSTem:DESCRiption?	Returns the description of the device.
	SYSTem:ERRor:ALL?	Queries the error/event queue for all the unread errors, returns an integer and a string that describes each error, and removes the error from the queue.
	SYSTem:ERRor:CODE:ALL?	Queries the error/event queue for all the unread errors, returns the error codes that corresponds to the unread errors in a comma-separated list, and removes the errors from the queue.
	SYSTem:ERRor:COUNt?	Queries the error/event queue for the number of unread errors and returns the count.
	SYSTem:ERRor[:NEXT]?	Queries the error/event queue for the next error, returns an integer and a string that describes the error, and removes the error from the queue.
	SYSTem:ERRor:CODE[:NEXT]?	Queries the error/event queue for the next error, returns an integer that corresponds to the error code, and removes the error from the queue.
	SYSTem:COMMunicate:NETwork: IPAddr	Sets the static IP address used by the DT8824 LXI instrument module on the network.
	SYSTem:COMMunicate:NETwork: IPAddr?	Returns the static IP address currently used by the DT8824 LXI instrument module on the network.
	SYSTem:COMMunicate:NETwork: MASK	Sets the subnet mask of the static IP address used by the DT8824 LXI instrument module on the network.
SYSTem:COMMunicate:NETwork: MASK?	Returns the subnet mask of the static IP address currently used by the DT8824 LXI instrument module on the network.	

**Table 2: Subsystem SCPI Commands for Programming DT8824 LXI Instrument Modules (cont.)**

Subsystem	Mnemonic	Description
SYSTem (cont.)	SYSTem:PASSword:CDISable	Disables use of the password-protected commands.
	SYSTem:PASSword[:CENable]	Enables use of password-protected commands.
	SYSTem:PASSword:CENable:STATe?	Returns whether password-protected commands are enabled or disabled.
	SYSTem:PASSword:NEW	Changes the existing password to a new password.
	SYSTem:VERSion?	Returns the SCPI version number to which the DT8824 LXI instrument module complies.
	SYSTem:TIME?	Returns the current time used by the DT8824 LXI instrument module. This time is updated automatically by an SNTP server.
	SYSTem:TZONE?	Returns the time zone that is currently used by the instrument module, as an offset from GMT (Greenwich Mean Time).
	SYSTem:RANge[:MAX]?	Returns the maximum full-scale voltage range that is supported by the instrument module.
AD	AD:BUFFer:MODE	Configures the wrap mode of the input buffer.
	AD:BUFFer:MODE?	Returns the currently configured wrap mode for the input buffer.
	AD:ENABLE	Enables or disables sampling of the specified analog input channels.
	AD:ENABLE?	Returns whether sampling is enabled or disabled for the specified analog input channels.
	AD:GAIN[:CONFigure]	Configures the gain as 1 or 10 for the specified analog input channels.
	AD:GAIN[:CONFigure]?	Returns the gain for the specified analog input channels.
	[AD:]CLOCK:SOURce	Specifies the clock source (internal or external) for the ADC and DAC master clock generators.
	[AD:]CLOCK:SOURce?	Returns the currently configured clock source for the ADC and DAC master clock generators.
	AD:CLOCK:FREquency [:CONFigure]	Configures the sampling rate for the input operation.
	AD:CLOCK:FREquency [:CONFigure]?	Returns the currently configured sampling rate for the input operation.
	AD:SYNc:SOURce?	Returns the currently configured source of the ADC synchronization pulse from the Trigger bus.
	AD:TRIGger[:SOURce]	Specifies the source of the trigger (immediate, external TTL, Trigger Bus, LAN) that starts the input operation.
	AD:TRIGger[:SOURce]?	Returns the currently configured trigger source for the analog input subsystem.

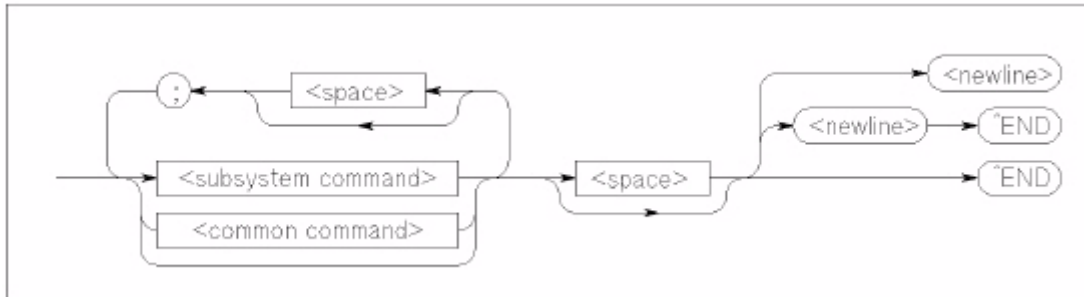
**Table 2: Subsystem SCPI Commands for Programming DT8824 LXI Instrument Modules (cont.)**

Subsystem	Mnemonic	Description
AD (cont.)	AD:TRIGger:EXTernal:EDGE	Configures the edge (positive or negative) of the external trigger that is used to start the analog input operation.
	AD:TRIGger:EXTernal:EDGE?	Returns the edge (positive or negative) of the external trigger that is used to start the analog input operation.
	AD:ARM	Arms the A/D subsystem by writing the input configuration to the instrument module.
	AD:INITiate	Starts the analog input operation when the configured trigger is detected.
	AD:ABORt	Stops the analog input operation on the DT8824 LXI instrument module.
	AD:FETCh?	Returns a specified number of samples from the input buffer.
	AD:STATus:SCAN?	Returns the indices of the chronologically oldest and most recent scan records in the input buffer on the instrument module.
	AD:STATus?	Returns the status bits for the A/D subsystem.
WTB	WTB:LXI<n>:ENABle	Configures a specific line of the Wired Trigger Bus as either driven (an output) or disabled.
	WTB:LXI<n>:ENABle?	Returns the configuration of a specified line of the Wired Trigger Bus.
EVENT	EVENT:DOMain	Sets the value of the LXI domain octet that is transmitted and received in all LXI LAN event packets.
	EVENT:DOMain?	Returns the value of the LXI domain octet that is transmitted and received in all LXI LAN event packets.
	EVENT:LAN<n>:TRANsmit[:ENABle]	Enables or disables a specific LAN event for transmission from the instrument module when the analog input subsystem is triggered.
	EVENT:LAN<n>:TRANsmit[:ENABle]?	Returns the enabled/disabled state of a specific LAN event.
	EVENT:LAN<n>:NAME	Configures the LAN event ID field in the LXI LAN packet.
	EVENT:LAN<n>:NAME?	Returns the event ID in the LXI LAN packet.
DOUtput	DOUtput	Sets the value of the four digital output lines on the DT8824 LXI instrument module.
	DOUtput?	Returns the current value of the four digital output lines on the DT8824 LXI instrument module.
	DOUtput:AND	Performs a bitwise AND operation on the specified value and the current value of the digital output port.
	DOUtput:OR	Performs a bitwise OR operation on the specified value and the current value of the digital output port.

## Syntax of Program Messages

A program message consists of one or more properly formatted SCPI commands sent from the controller to a DT8824 LXI instrument module to request some action or to query the instrument module for a response.

Figure 1 shows the syntax of a program message:



**Figure 1: Syntax of Program Messages**

A semicolon (;) is used to separate commands within the same command group, and can also minimize typing. For example, you could send the following program message to set the state of the digital output port (all 4 digital output lines) and read the status of the digital output lines:

```
:OUTPut 15; :OUTput?
```

Use a semicolon and a colon to link commands from different groups. For example, this program message returns the contents of the Operation Status register, and then sets the state of the digital output port:

```
:STATus:OPERation:CONDition? ;:OUTPut 15
```

To terminate a program message, use one of the following program message terminators:

- <newline> or the <NL> character
- <^END>

<^END> means that the IEEE-488 EOI (End-Or-Identify) message was asserted at the same time that the last data byte was sent. <^END> is interpreted as a <NL> character and can be used to terminate a command string in place of a <NL> character.

- <newline><^END>

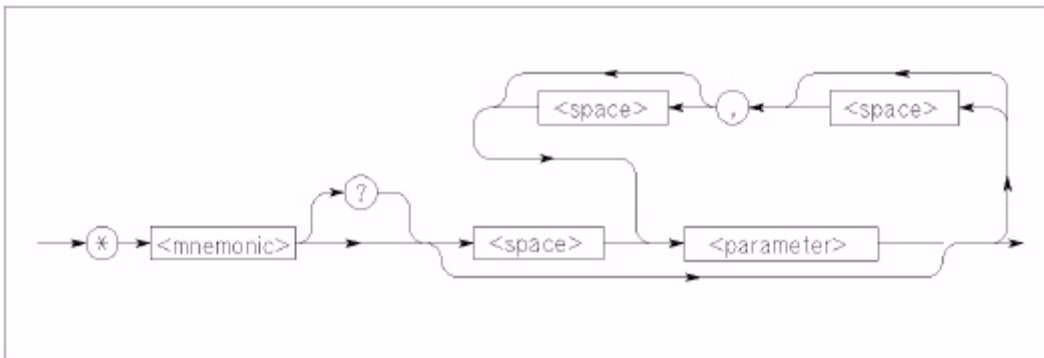
Terminating the program message always resets the current SCPI command path to the root level.

The following subsections describe the syntax of both the common SCPI commands and the subsystem SCPI commands.



## Syntax of Common SCPI Commands

Figure 2 shows the syntax of common SCPI commands.



**Figure 2: Syntax of Common SCPI Commands**

Common SCPI commands begin with an asterisk (\*). The command mnemonic is case-insensitive. For example, the following commands have the same effect:

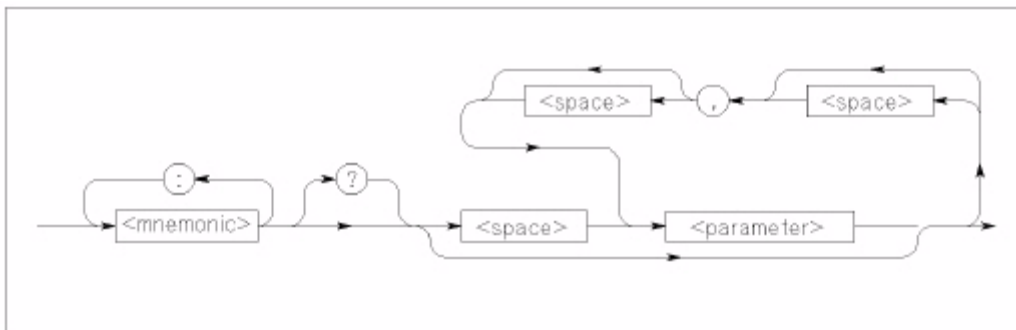
```
*RST
*rst
*Rst
```

Queries requires a question mark (?) at the end of the command header, as follows:

```
*IDN?
```

## Syntax of SCPI Subsystem Commands

Figure 3 shows the syntax of SCPI subsystem commands.



**Figure 3: Syntax of SCPI Subsystem Commands**

Use a colon ( : ) to separate command keywords or mnemonics, as shown below:

```
:STATus:OPERation:ENABle
```

Queries require a question mark (?) at the end of the command header, as follows:

```
:STATus:OPERation:CONDition?
```

### ***About Short- and Long-Form Mnemonics***

Note that the command syntax shows most command mnemonics (and some parameters) as a mixture of upper- and lower-case letters. The upper-case letters indicate the abbreviated spelling for the command. For shorter program lines, use the abbreviated form; for better program readability, use the long form.

You can use either upper- or lower-case letters to specify the command, as the mnemonic is case-insensitive. For example, here is the long form of a command:

```
SYSTem:COMMunicate:NETwork:MASk?
```

And, here is the short-form of this command:

```
Syst:Comm:NET:Mas?
```

### ***About Brackets, Braces, and Vertical Bars***

Some parameters are enclosed in square brackets ( [ ] ), indicating that the parameter is optional and can be omitted. The brackets are not sent with the command. If you do not specify a value for an optional parameter, the instrument module uses a default value.

Triangle brackets ( < > ) indicate that you must specify a value for the enclosed parameter. The brackets are not sent with the command.

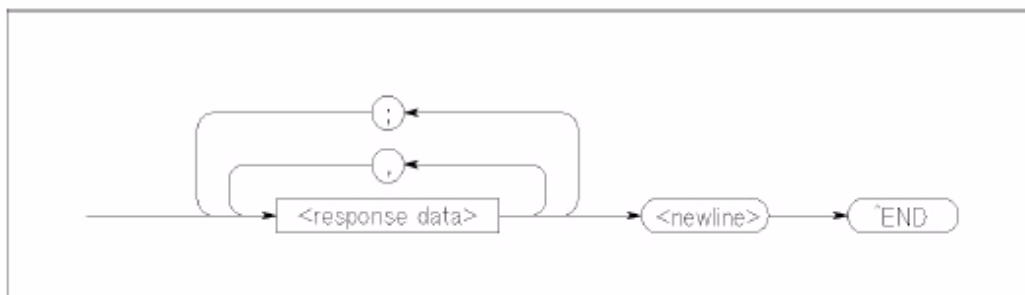
Braces ( { } ) enclose the parameter choices for a given command. The braces are not sent with the command.

A vertical bar ( | ) separates multiple parameter choices for a given command.

## Syntax of Response Messages

A response message consists of data in a specific SCPI format that was requested from a DT8824 LXI instrument module from the controller.

Figure 4 shows the syntax of a response message from a DT8824 LXI instrument module:



**Figure 4: Response Message Syntax**

Response messages may contain both commas (,) and semicolons (;) as separators. When a single query command returns multiple values, a comma is used to separate each data item. When multiple queries are sent in the same message, the groups of data items corresponding to each query are separated by semicolons.

The terminator for a response message is always <newline><^END>.

---

**Note:** For character data types in response messages, only the short-form of the mnemonic is returned in all uppercase letters.

---

## SCPI Data Types

SCPI defines several different data types for use in program messages to a DT8824 LXI instrument module and for use in response messages from a DT8824 LXI instrument module.

DT8824 LXI instrument modules use the following subset of SCPI data types:

- Character
- String
- <NR1>
- <NR2>
- <NRr>
- <NRf>
- <Boolean>
- <Block>

This section summarizes these data types. Refer to the SCPI standards document for more information about these data types.

### Character Data Types

If a command parameter takes a character data type, a specific number of settings are allowed for the parameter. For example, in the following command, you can specify one of the following character data types for the buffer wrap mode: WRAP, NOWRAP, or DEFAULT.

```
:AD:BUFFer:MODE {WRAP|NOWRAP|DEFAULT}
```

Character data types have the following characteristics:

- Can have either the short or long form in program messages and are returned in short-form only in response messages
- Are case insensitive in program messages and are in uppercase only in response messages
- Must have a specific length

### String Data Types

Strings used in command parameters and responses follow these rules:

- Strings are enclosed in double quotes " "  
For example,  
"This is an example"  
specifies the following string: This is an example

- Use double quotes within double quotes to indicate the part of the string that should appear in quotes; note that double and single quotes used inside the string must be duplicated. For example,

```
"This is the "example"
```

specifies the following string: This is the “example”

- ? Strings are case sensitive

## <NR1>Value Data Type

The <NR1> value data type is used to specify zero, and positive and negative integer decimal values, including optional signs. If you need to indicate decimal points, use the <NR2> value data type, instead.

The following values are examples of the <NR1> data type:

```
0          255          -2
```

## <NR2> Value Data Type

The <NR2> value data type is used to specify zero, and positive and negative decimal values, including optional signs and decimal points.

The difference between <NR1> and <NR2> is the explicit decimal point. Note that 0 is a special case and redundant decimal points are ignored.

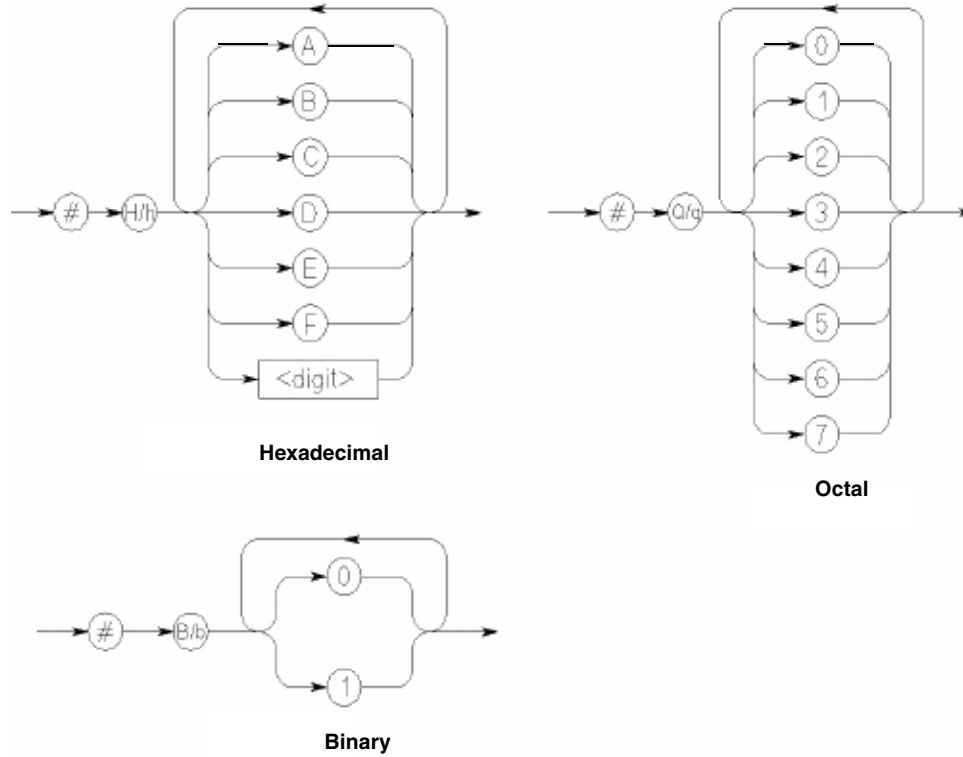
The following values are examples of the <NR2> data type:

```
-1.234      1.0          0.0
```

## <NRr>Value Data Type

The <NRr> data type is used to specify a non-decimal numeric value, such a hexadecimal, octal, or binary numeric value.

[Figure 5](#) shows the format of the <NRr> data type:



**Figure 5: <NRr>Value Data Type**

The following examples show how the number 16384 in decimal format is represented as a <NRr> data type:

- *Hexadecimal <NRr> value: #H4000*
- *Octal <NRr> value: #Q40000*
- *Binary <NRr> value: #B100000000000000*

## <NRf>Value Data Type

The <NRf> data type is used to specify a floating-point value. These values include digits with an implied decimal point, an explicit decimal point, or an explicit decimal point and an exponent.

The following values are examples of the <NRf> data type:

6553                      1.525e-4                      0.100000

## <Boolean> Data Type

A Boolean data type for a parameter and response represents a single binary condition that is either True or False. Boolean values are defined as follows:

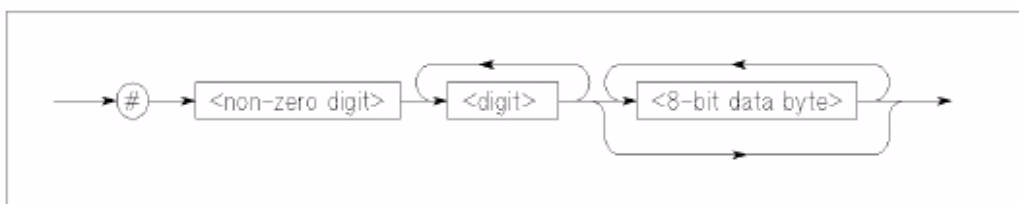
- 0 or OFF – indicates that the condition is False
- 1 or ON – indicates that the condition is True

Note that the characters OFF and ON are not case sensitive. While a DT8824 LXI instrument module accepts the characters OFF and ON instead of 0 and 1, if queried, these instrument modules return Boolean responses as either 0 or 1.

## <Block> Data Type

The <Block> data type is used to transfer array and system-defined blocks of data between the controller and the instrument module. This is the most efficient data format for transferring data.

Figure 6 shows the format of the <Block> data type:



**Figure 6: Block Data Type**

where:

- # is the starting character of the block
- <non-zero digit> is a single decimal value that specifies how many digits will follow in the next field
- <digit> is a value in <NR1> format that specifies how many <8-bit data byte>s follow. In effect, this field specifies the length, in bytes, of the remainder of the block.
- <8-bit data byte> specifies each 8-bit byte of the block. (The number of 8-bit data bytes is specified in the preceding field. ) The contents of this block of bytes is dependent on the specific command.

---

**Note:** Prior to interpreting multi-byte values, such as floats, integers, unsigned integers, and so on, you must convert values from network-byte order (where the highest bytes are sent first; i.e., big-endian) to host-byte-order (where the lowest bytes are read first, i.e., little endian).

---

The following example shows how a block of five data bytes is formatted:

```
#1<5><b0><b1><b2><b3><b4>
```



## SCPI Expression Types

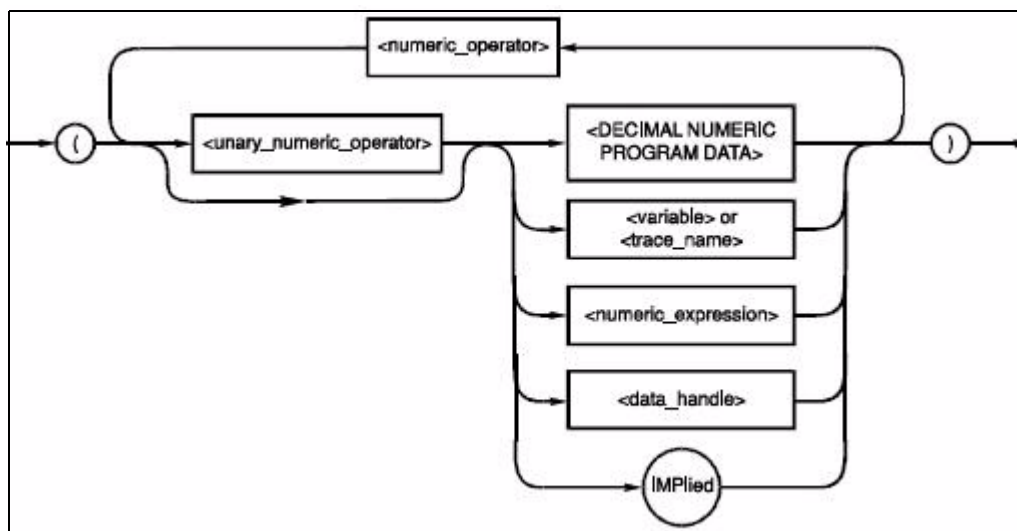
SCPI defines five types of expressions:

- Numeric expressions
- Channel lists
- Numeric lists
- Data Interchange Format (DIF) expressions
- Instrument-specifier expressions

The following subsections summarize these expressions. Refer to the SCPI standard document for more information about these expressions.

### Numeric Expressions

A numeric expression is a collection of terms which evaluates to a trace, number, array, or other data element. Expressions can contain terms which are numbers, traces, variables, or expressions. The syntax of a numeric expression is shown in [Figure 7](#):



**Figure 7: Syntax of a Numeric Expression**

where,

- `<numeric_operator>` is defined as one of the following operators: +, -, \*, /, ^, MOD, DIV, ADD, OR, EXOR
- `<unary_numeric_operator>` is defined as one of the following operators: -, +, NOT. Unary operators should not be used with signed numbers.
- `<variable>` or `<trace_name>` is defined as character program data

Expressions are evaluated according to the following precedence rules:

1. Enclosed by parentheses
2. Unary operators (+ and -)
3. ^ (exponentiation)
4. \* (multiplication), / (division), MOD and DIV
5. + (addition) and - (subtraction)
6. NOT
7. AND
8. OR and EXOR
9. Left to right

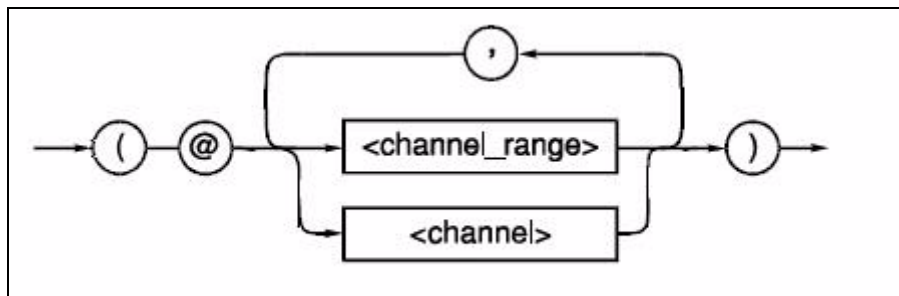
Elements in a numeric expression are promoted to the size and type of the most complex element, and the result of the expression is of that type. For example, an expression containing a <trace\_name> is evaluated according to the rules for arithmetic on traces, and the result is a trace. If an expression contains both a <trace\_name> and scalar data, a trace is created with all elements set to the value of the scalar data before arithmetic is performed. For example, if TREF is a trace\_name, the expression (TREF-3) results in a trace that is the same size as TREF, with each data element lower by three.

## Channel Lists

DT8824 LXI instrument modules use channel lists as parameters in certain commands and in responses to certain queries.

Channel lists are used to specify the analog input channels on a DT8824 LXI instrument module that are used to measure input channels. A channel list may appear in a measurement, configuration, or command. By design, all analog input channels that are specified in the channel list are measured simultaneously.

The syntax of a channel list expression is shown in [Figure 8](#):



**Figure 8: Syntax of a Channel List Expression**

where:

- ( is the starting character of the channel list
- @ is the next character of the channel list
- <channel range> consists of two channels in <NR1> format separated by a colon.

The first channel in the range must be lower than the second in the range. For example, for analog input channels on the DT8824 instrument module, the first channel is 1 and the last channel is 4.

Separate multiple <channel\_range> elements with commas.

- <channel> consists of one channel number in <NR1> format. Separate multiple <channel> elements with commas.
- ) terminates the channel list expression

For example, to set the gain on analog input channels 1 through 4, use one of the following commands:

```
:AD:GAIN 16, (@1:4)
:AD:GAIN 16, (@1, 2, 3, 4)
```

To set the gain for analog input channel 3 only, use this command:

```
:AD:GAIN 16, (@3)
```

### Numeric Lists

A numeric list is an expression format for compactly expressing numbers and ranges of numbers in a single parameter. The syntax of a numeric list expression is shown in Figure 9:

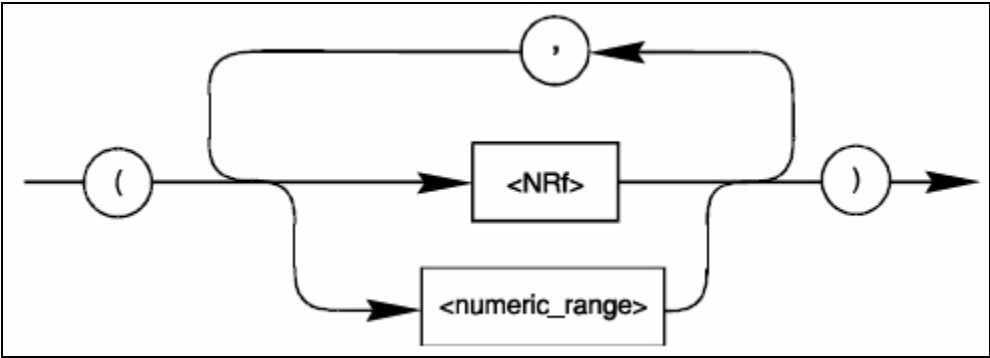


Figure 9: Syntax of Numeric List Expressions

where:

- <NRf> is an extended format, described on [page 22](#).
- <numeric\_range> is defined as two <NRf> data types separated by colons. The range is inclusive of the specified numbers.

## Data Interchange Format (DIF) Expressions

The data interchange format is block-structured and lets software packages and instruments share waveform and other data.

The following block types are available in DIF expressions:

- DATA block – Contains the data.
- IDENTify block – Describes the manner and environment in which the data was obtained.
- DIMension, ORDer, and ENCode blocks – Describe how the data is physically represented and logically organized.
- TRACe and VIEW blocks – Provide semantic information about the data.
- REMark block – Contains textual comments regarding the data.
- DIF block – Identifies the block as a <dif\_expression> and describes the version of SCPI that is used.

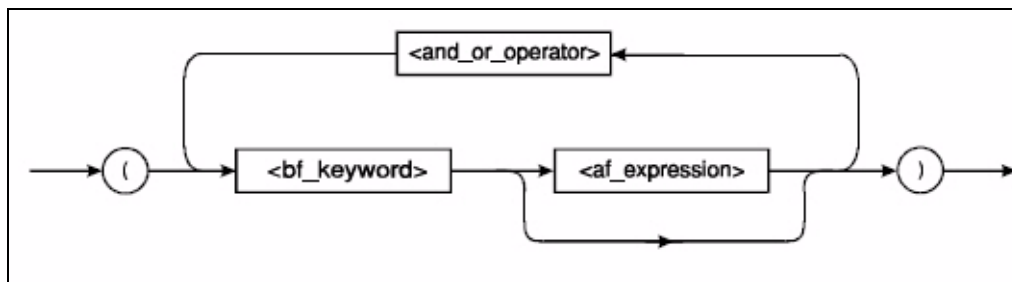
In a DIF expression, each hierarchical level of a block is introduced by a block name with its subordinate elements enclosed in parentheses. A block may have a modifier and may contain subordinate blocks and keyword units, or both. Keyword units consist of a keyword followed by one or more values. If a keyword has more than one value, the values are separated by commas. The following example shows a simple data set. All blocks and keywords are indented to show their hierarchical relationship:

```
(DIF (VERSION 1993.0)
  IDENTify (
    NAME "Data Format Example"
    TEST (
      NUMBER "7D4", "2.4")
    ENCode (
      HRANge 75
      RANge 25)
  DIMension=X (
    TYPE IMPLicit
    SCALe 0.01
    SIZE 7
    UNITs "S")
  DIMension=Y (
    TYPE EXPLIcit
    SCALe 0.02
    OFFSet 0.1UNITs "V")
  DATA (
    CURVe(
      VALues 49.0, 48.0, 50.2, 61.3, 68.5, 38.6, 48.0)))
```

The data interchange format overall is formatted as an IEEE 488.2 <EXPRESSION PROGRAM DATA> element. Within this element, the various blocks, modifiers, keywords, and value types are composed of syntactic elements that, for the most part, are identical to the corresponding types specified in IEEE 488.2 or a subset of them. Refer to the IEEE 488.2 standard for more information.

## Instrument-Specifier Expressions

An <instrument\_specifier> is a combination of one or more base functionality keywords along with optional additional functionality keywords that define an instrument class. The syntax of an instrument-specifier expression is shown in [Figure 10](#).



**Figure 10: Syntax of an Instrument-Specifier Expression**

where:

- ( is the starting character of the instrument-specifier expression
- <and\_or\_operator> is either '&' (ASCII hexadecimal 26) or '|' (ASCII hexadecimal 7C)
- <bf\_keyword> is a base functionality keyword
- <af\_expression> includes additional <and\_or\_operator>s and additional functionality keywords
- ) terminates the instrument-specifier expression





# ***Using SCPI Commands with DT8824 Instrument Modules***

Installing SCPI Support Files for the DT8824.....	32
Configuring the LAN Settings of the DT8824 .....	33
Performing Input Operations .....	38
Performing Digital Output Operations.....	51

## **Installing SCPI Support Files for the DT8824**

If you have not already done so, install the SCPI support files, including the documentation for your DT8824 LXI instrument module, this manual, and the SCPI example programs for the DT8824, by performing the following steps:

1. Insert the DT8824 CD into your CD-ROM or DVD drive.  
*The installation program should automatically start, and the DT8824 installation program should appear.*
2. If the installation program does not automatically start, double-click **Setup.exe** from the CD.  
*The DT8824 installation program appears.*
3. Click **Install from Web (recommended)** to get the latest version of the software or **Install from CD** to install the software from the CD.
4. If you are installing from the web, click the link to install the DT8824 SCPI software.
5. If you are installing from the DT8824 CD, perform these steps:
  - a. Click **Install DT8824 Software**.
  - b. Click **Install Selected Features** and follow the prompts to install the example programs and documentation.
  - c. When you are finished with the DT8824 CD, click **Quit Installer**.

To access the SCPI documentation and examples for the DT8824, from the Windows Start menu, click **Programs -> Data Translation, Inc -> Instruments -> DT8824 -> SCPI Support**.



## Configuring the LAN Settings of the DT8824

The Ethernet address of a DT8824 instrument module consists of two parts:

- The IP address – The “Internet Protocol” numeric address that identifies where messages are sent or received on the LAN (Local Area Network).
- Subnet mask – A 32-bit value that enables the recipient of IP packets to distinguish the network ID and host ID portions of the IP address.

The instrument module can acquire an Ethernet address in one of the following ways:

- DHCP (Dynamic Host Configuration Protocol server) – The address is set by the network server automatically when the DT8824 instrument module is powered on. The address is different each time the instrument module is powered on.

By default, DHCP is enabled for DT8824 instrument modules.

- Auto-IP – If the DHCP server is not available, the instrument module configures its own IP address in the range of 169.254.0.0 to 169.254.255.255 with a subnet mask of 255.255.0.0. Like with DHCP, the address is different each time the instrument module is powered on.

By default, Auto-IP is enabled for DT8824 instrument modules.

- Static IP – Using SCPI commands, you can specify the static IP address of the instrument module. The static IP address does not change when the instrument module is powered on.

The following subsections provide information about configuring and returning the LAN settings of DT8824 instrument modules.

### Setting the Static IP and Subnet Mask of the Instrument Module

You can configure the static IP address and subnet mask of the instrument module on the LAN using the following SCPI commands:

- Set the static IP address of the instrument module using the **SYSTem:COMMunicate:NETwork:IPAddr** command, described on [page 79](#).
- Set the subnet mask of the instrument module using the **SYSTem:COMMunicate:NETwork:MASK** command, described on [page 80](#).

### Specifying a Description of the Instrument Module

To make identifying the instrument module easier, you can specify a description of the instrument module using the **SYSTem:DESCRiption** command, described on [page 73](#).

## Getting Information about the Instrument Module

Using SCPI commands, you can return the following information about each DT8824 instrument module on the LAN:

- IP address – Use the **SYSTem:COMMunicate:NETwork:IP?** query, described on [page 80](#), to return the IP address of the instrument module, regardless of the method (DHCP, Auto-IP, static IP) used to acquire the address.
- Subnet mask – Use the **SYSTem:COMMunicate:NETwork:MASK?** command, described on [page 81](#), to return the subnet mask of the instrument module, regardless of the method (DHCP, Auto-IP, static IP) used to acquire the subnet mask.
- Description – Use the **SYSTem:DESCRiption?** command, described on [page 74](#), to return the description of the instrument module.

## Using Password-Protected Commands

The DT8824 and DT8824-HV instruments implement authentication using a single, user-configurable password. The password is saved in permanent memory in the instrument.

---

**Note:** The instrument's web interface, IVI-COM driver, and SCPI interface use the same password.

---

On power up, all SCPI commands that either change the configuration of the instrument or operate the instrument are disabled, by default. This is to prevent unauthorized access to an instrument by some client on the network.

By using the **SYSTem:PASSword[:CENable]** command, described on [page 83](#), you can enable password-protection. When password protection is enabled, the correct password must be issued to use the following SCPI commands:

- **AD:BUFFer:MODE**, described on [page 93](#)
- **AD:ENABle**, described on [page 96](#)
- **AD:GAIN[:CONFigure]**, described on [page 99](#)
- **[AD:]CLOCK:SOURce**, described on [page 97](#)
- **AD:CLOCK:FREQuency[:CONFigure]**, described on [page 101](#)
- **AD:TRIGger[:SOURce]**, described on [page 105](#)
- **AD:TRIGger:EXTernal:EDGE**, described on [page 109](#)
- **AD:ARM**, described on [page 92](#)
- **AD:INITiate**, described on [page 114](#)
- **AD:ABORt**, described on [page 91](#)
- **DOUTput**, described on [page 130](#)
- **DOUTput:AND**, described on [page 128](#)
- **DOUTput:OR**, described on [page 129](#)
- **EVENT:DOMain**, described on [page 121](#)
- **EVENT:LAN<n>:NAME**, described on [page 122](#)
- **EVENT:LAN<n>:TRANsmit[:ENABle]**, described on [page 125](#)
- **WTB:LXI<n>[:OUTput]:ENABle**, described on [page 117](#)
- **SYSTem:DESCRiption**, described on [page 73](#)
- **\*RST**, described on [page 59](#)
- **\*CLS**, described on [page 54](#)

To disable the use of password-protected commands, use the **SYSTem:PASSword:CDISable** command, described on [page 81](#).

You can determine whether password-protected commands are disabled or enabled using the **SYSTem:PASSword:CENable:STATe?** command, described on [page 85](#).

The default password is *admin* for the DT8824 and DT8824-HV instruments.

You change the password from the current password that is stored in permanent memory in the instrument to a new password using the LAN configuration page of the web interface for the instrument, through the IVI-COM driver, or through the SCPI command **SYSTem:PASSword:NEW**, described on [page 86](#).

A SCPI client that wishes to configure and/or operate a DT8824 or DT8824-HV instrument should use these commands as follows:

1. Use the query **:SYSTem:PASSword:CENable:STATe?** to determine if password-protected commands are enabled.

The following example shows that password-protected commands are disabled:

```
-> *IDN?
<- Data Translation,DT8874,10449004,2.0.0.0
-> :SYSTem:PASSword:CENable:STATe?
<- 0
```

2. If password-protected commands are disabled, enable password-protected commands by using **SYSTem:PASSword:CENable**. If this command is successful, the client can configure the instrument and operate it. An example follows:

```
-> :SYST:PASS:CEN admin
-> :SYST:ERR?
<- 0, "No error"
-> :SYSTem:PASSword:CENable:STATe?
<- 1
-> :AD:GAIN 1, (@1,2)
-> :AD:GAIN 16, (@3,4)
-> :SYST:ERR?
<- 0, "No error"
```

3. If this command is successful, the client can configure the instrument and start scans.
4. To prevent some other SCPI client, locally or on the LAN, from stopping the scan and modifying the configuration, the client that enabled password-protected commands should issue **SYSTem:PASSword:CDISable**.

The following example shows an attempt to disable protected commands using an erroneous password:

```
-> :SYSTem:PASSword:CDISable bogus
-> :SYST:ERR?
<- -221, "Settings conflict;:SYST:PASS:CDIS"
```

The following is an example in which the correct password is used to disable protected commands:

```
-> :SYSTem:PASSword:CDISable admin
-> :SYST:ERR?
<- 0, "No error"
-> :SYSTem:PASSword:CENable:STATe?
<- 0
```

Since protected commands are disabled, any attempt to use them will fail, as shown in the following example:

```
-> :AD:GAIN 1, (@1,2)
-> :SYST:ERR?
<- -203, "Command protected;AD:GAIN"
-> :AD:INIT
-> :SYST:ERR?
<- -203, "Command protected;AD:INIT"
```

In this scenario, one "master" SCPI client can configure the instrument, start scans, and disable SCPI commands. Since all queries are permitted regardless of the state of password protection, any client can perform **FETCh?** and other queries. The "master" can eventually enable password-protected commands and stop scans. The password should not be shared and should only be known by the master SCPI client.

5. At any time, the password can be changed using **SYSTem:PASSword:NEW**. Note that this command is never disabled.

The following example shows an attempt to change the password using a wrong password:

```
-> :SYST:PASS:NEW bogus, admin1
-> :SYST:ERR?
<- -221, "Settings conflict;:SYST:PASS:NEW"
```

In the following example, the existing password *admin* is successfully changed to *admin1*:

```
-> :SYST:PASS:NEW admin, admin1
-> :SYST:ERR?
<- 0, "No error"
```

The following example shows that the old password *admin* is no longer in use and the new password *admin1* is in effect:

```
-> :SYSTem:PASSword:CDISable admin
-> :SYST:ERR?
<- -221, "Settings conflict;:SYST:PASS:CDIS"
-> :SYSTem:PASSword:CDISable admin1
-> :SYST:ERR?
<- 0, "No error"
-> :SYSTem:PASSword:CENable:STATe?
<- 0
```

## Performing Input Operations

You can acquire data from up to four analog input channels on the DT8824 LXI instrument module. The input operation starts when the specified trigger source is detected. The operation is paced by the specified clock source at the specified input clock frequency.

To set up and acquire input data from the DT8824 or DT8824-HV using SCPI commands, perform the following steps:

1. Enable and configure the analog input channels that you want to sample, as described on [page 38](#).
2. Set up the input buffer, as described on [page 39](#).
3. Set up the clock that paces the input operation, as described on [page 40](#).
4. Set up the trigger that starts the input operation, as described on [page 42](#).
5. Arm the analog input subsystem, as described on [page 47](#).
6. If you are using the software trigger source (IMMediate), initiate the input operation, as described on [page 47](#).
7. (Optional) Determine the status of the input operation, as described on [page 47](#).
8. Retrieve the data from the input buffer on the DT8824, as described on [page 48](#).
9. If desired, stop the operation, as described on [page 50](#).

### Enabling and Configuring Analog Input Channels

The DT8824 instrument module supports four, 24-bit, differential analog input channels (numbered 1 to 4).

Enable the analog input channels you want to sample by using the **AD:ENABle** command. To determine whether specified analog input channels are enabled or disabled, use the **AD:ENABle?** query.

### Configuring the Input Range and Gain

The DT8824 provides an input range of  $\pm 10$  V and software-selectable gains of 1, 8, 16, and 32. This provides effective input ranges of  $\pm 10$  V when the gain is 1,  $\pm 1.25$  V when the gain is 8,  $\pm 0.625$  V when the gain is 16, and  $\pm 0.3125$  V when the gain is 32.

The DT8824-HV supports a bipolar range of  $\pm 600$  V and software-selectable gains of 1, 8, 16, and 32. This provides effective input ranges of  $\pm 600$  V when the gain is 1,  $\pm 75$  V when the gain is 8,  $\pm 37.5$  V when the gain is 16, and  $\pm 18.75$  V when the gain is 32.

You can determine the maximum full-scale voltage range of the instrument by using the **SYSTem:RANge[:MAX]?** query.

You can set the gain for each analog input channel programmatically using the **AD:GAIN[:CONFigure]** command. To return the current gain setting for each analog input channel, use the **AD:GAIN[:CONFigure]?** query.

## Configuring the Input Buffer

DT8824 instrument modules use an 8 MB input buffer for storing data from each of up to 4 enabled input channels (analog input channels 1, 2, 3, 4). One sample from each of the enabled input channels is called a scan.

You can use the **AD:BUFFer:MODE** command to specify one of the following wrap modes for the input buffer:

- Continuous wrap mode – Specify WRAP mode if you want the input operation to continue indefinitely until you stop it using the **AD:ABORt** command. In this case, when the end of the input buffer is reached, the operation wraps to the beginning of the input buffer overwriting the oldest scan data with the latest scan data.
- No wrap mode – Specify NOWRAP mode if you want the input operation to stop automatically when the input buffer is filled.

To verify the currently configured wrap mode for the input buffer, use the **AD:BUFFer:MODE?** query.

You can use the **AD:BUFFer:SIZE[:SCAns]?** query to determine how many scans can be stored in the input buffer based on the number of enabled input channels and the input sampling rate.

---

**Note:** The sampling rate determines DMA block size that is used for the scan data, where each block has a fixed number of header bytes that is not used to store scans. The DMA block size is smaller for the minimum sampling frequency, resulting in more space for the headers, and greater for the maximum sampling frequency.

Since the maximum input buffer size is 8 MB and each sample is 4 bytes, you can store a maximum of 2 M samples in the input buffer. If you sample each input channel at the maximum input frequency (4800 Hz), the input buffer will fill in approximately 106 seconds (4800 samples/s per channel).

The maximum size of data transfer over Ethernet using the **AD:FETCh?** command is 32 kB. Since each sample is 4 bytes, the maximum number of samples per **AD:FETCh?** is 8 ksamples. Therefore, if you are using continuous wrap mode, ensure that you call **AD:FETCh?** in a tight loop to ensure that you retrieve all the samples in the input buffer before the buffer is overwritten.

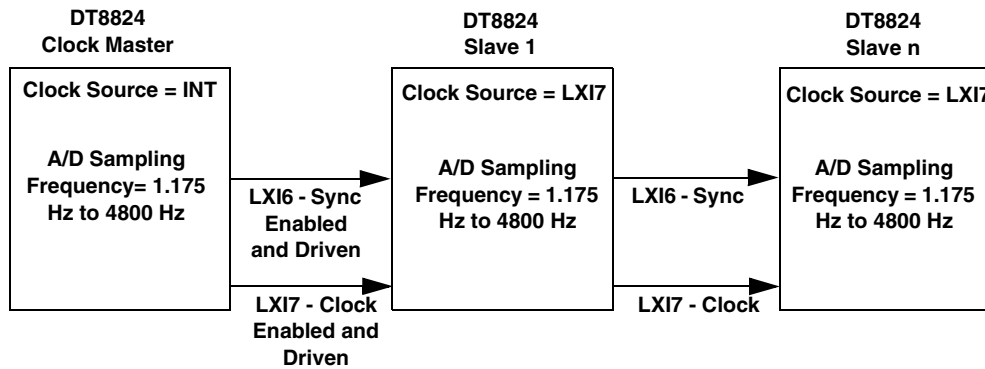
---

## Configuring the Input Clock Source

Using the `[AD:]CLOCK:SOURce` command, you can specify one of the following clock sources as the source of the reference clock for the ADC master clock generator:

- **Internal clock** – Selects the internal reference clock on the DT8824 instrument module.
- **LXI7** – When multiple DT8824 instrument modules are connected together using the Trigger Bus, as shown in Figure 11, LXI7 selects the dedicated clock signal on the Trigger Bus that is used to synchronize the clock on the connected DT8824 instrument modules.

In this configuration, one DT8824 instrument module is the clock master and the remaining DT8824 instrument modules are slaves.



Note: LXI bus terminators must be installed for the first and last instrument modules in the Trigger Bus chain.

**Figure 11: Synchronizing the Clock on Multiple DT8824 Instrument Modules for Analog Input Operations**

If you are using only one DT8824 instrument module or if you are using multiple DT8824 instrument modules but do not want to synchronize their clocks, set up the clocks as follows:

1. Use the `[AD:]CLOCK:SOURce` command to set the clock source to *INTernal*. This command configures the device to use the internal clock reference.
2. Use the `AD:CLOCK:FREQuency[:CONFigure]` command to specify the sampling frequency of the analog input subsystem. The sampling frequency ranges from 1.175 Hz to 4800 Hz.



If you want to synchronize the clocks of multiple DT8824 instrument modules that are connected together using the Trigger Bus, set up the DT8824 instrument modules as follows:

---

**Note:** Ensure that you perform this procedure before you set up the triggers (as described on [page 45](#)), and before you arm the subsystems (as described on [page 47](#)).

---

1. On the DT8824 slave, use the **[AD:]CLOCK:SOURce** command to set the reference clock source to *LXI7*. This command configures the slave device to receive the clock signal (LXI7) from the Trigger Bus; in to receive the synchronization pulse (LXI6) from the Trigger Bus.
2. On the DT8824 clock master, use the **[AD:]CLOCK:SOURce** command to set the reference clock source to *INTernal*. This command configures the clock master to use the internal clock reference; in addition, this command automatically configures the clock master to use the internal ADC synchronization pulse.
3. On the DT8824 clock master, use the **WTB:LXI7[:OUTput]:ENABle** command to drive out the clock signal (LXI7) on the Trigger Bus. This command also drives out the ADC synchronization pulse (LXI6) on the Trigger Bus.
4. On each DT8824 slave, use the **AD:CLOCK:FREQuency[:CONFigure]** command to specify the sampling frequency of the analog input subsystem. The sampling frequency ranges from 1.175 Hz to 4800 Hz.

---

**Notes:** For proper operation, it is important that you configure the clock frequency of the slave instrument modules before configuring the clock frequency of the clock master.

It is also recommended that you choose the same sampling frequency for all instrument modules to avoid problems.

---

5. On the DT8824 clock master, use the **AD:CLOCK:FREQuency[:CONFigure]** command to specify the sampling frequency of the analog input subsystem. The sampling frequency ranges from 1.175 Hz to 4800 Hz.

Once the clocks are configured, the clock and sync signals are sent over the Trigger Bus to all slaves. The analog input operation on each slave starts on the first sample clock pulse after the trigger is received. Refer to [page 42](#) for information on triggering DT8824 instrument modules.

---

**Note:** If you change the sampling frequency of one or more slave instrument modules, you must reconfigure the clock frequency of the clock master, even if its parameters have not changed.

---

To return the currently configured reference clock source, use the `[AD:]CLOCK:SOURce?` query. To return the currently configured ADC synchronization source, use the `AD:SYNc:SOURce?` query.

The actual sampling frequency that the device can achieve may be slightly different than the frequency you specified due to the accuracy of the clock. You can determine the actual sampling frequency that is configured using the `AD:CLOCK:FREQUency[:CONFigure]?` query.

---

**Note:** DT8824 instrument modules use 24-bit Delta-Sigma ADCs that provide anti-aliasing filters based on the sampling frequency.

According to sampling theory (Nyquist Theorem), specify a frequency that is at least twice as fast as the input's highest frequency component. For example, to accurately sample a 20 kHz signal, specify a sampling frequency of at least 40 kHz to avoid aliasing.

DT8824 instrument modules support a wide pass band of 0.5 Hz to 25.8 kHz ( $0.49 \times$  sampling frequency) to eliminate aliasing, allowing you to measure low frequency signals accurately at the Nyquist sampling rate.

---

## Configuring the Input Trigger

A trigger is an event that occurs based on a specified set of conditions. Once the analog input subsystem is armed with the `AD:ARM` command (and, if the trigger source is `IMMEDIATE`, started with the `AD:INITiate` command), acquisition starts with the first sample after the trigger is received. Refer to [page 47](#) for more information on the `AD:ARM` and `AD:INITiate` commands.

---

**Note:** The Input Trigger LED on the front panel of the DT8824 instrument module turns solid amber when the analog input subsystem is armed and solid green when the analog input subsystem is triggered. When the analog input subsystem is idle, the Input Trigger LED is off.

---

This section describes each of the supported trigger sources in detail. To return the currently configured trigger source, use the `AD:TRIGger[:SOURce]?` query.

### Software Trigger

A software trigger event occurs when you start the analog input operation with the `AD:INITiate` command (the computer issues a write to the instrument module to begin conversions).

Use the `AD:TRIGger[:SOURce]` command with the `IMMEDIATE` parameter to select the software trigger.

## External, Digital (TTL) Trigger

An external digital (TTL) trigger event occurs when the instrument module detects a rising-edge transition on the signal connected to the Trigger In input on the instrument module.

Use the `AD:TRIGger[:SOURce]` command with the *EXTernal* parameter to specify the external, digital (TTL) trigger.

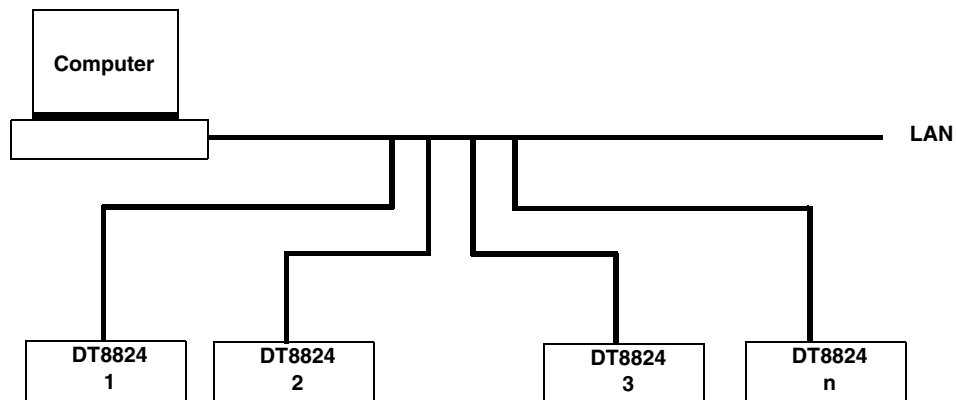
You can configure the edge of the external trigger that starts the analog input operation by using the `AD:TRIGger:EXTernal:EDGE` command with one of the following parameters:

- *NEGative* – The falling edge of the external trigger starts the analog input operation.
- *POSitive* – The rising edge of the external trigger starts the analog input operation.

To return the currently configured trigger edge, use the `AD:TRIGger:EXTernal:EDGE?` query.

## LAN Trigger Packet

When multiple DT8824 instrument modules are connected together over the local area network (LAN), as shown in 12, you can synchronize the start of acquisition by transmitting one of eight LAN trigger packets (LAN0 to LAN7) over the network.



**Figure 12: Synchronizing the Start of Acquisition When Triggering Instrument Modules Over the LAN**

The timing of the trigger packets from device to device is dependent on the network traffic on the LAN.

The DT8824 supports multicast transmission of LAN trigger packets using UDP (User Datagram Protocol) port 5044 and UDP multicast address 224.0.23.159. The DT8824 does not support unicast LAN events transmitted via TCP connections.

[Figure 13](#) shows the format of a LAN trigger packet:

<b>HW Detect (3 bytes)</b>	<b>Domain (1 byte)</b>	<b>Event ID (16 bytes)</b>	<b>Sequence (UInt32)</b>	<b>Time stamp (10 bytes)</b>	<b>Epoch (UInt16)</b>	<b>Flags (UInt16)</b>	<b>Data Fields</b>	<b>0 (2 Bytes)</b>
--------------------------------	----------------------------	--------------------------------	------------------------------	----------------------------------	---------------------------	---------------------------	--------------------	------------------------

**Figure 13: Format of a LAN Trigger Packet**

The fields of the LAN trigger packet are described as follows:

- **HW Detect** – Identifies valid packets and is reserved for future hardware detection of LAN events. This field should be set to *LXI*. Note that the third byte, ASCII "T", is also used as a version identifier; future revisions to the LXI standard may change this value.
- **Domain** – This field represents a group of devices that are managed by a single directory and use shared resources. Values for the LXI domain range from 0 to 255. The default value is 0. This field is treated as an unsigned byte.

To transmit LAN trigger packets to or from a DT8824 instrument module, ensure that you set the domain field of each DT8824 instrument module to the same value using the **EVENT:DOMain** command.

- **Event ID** – This field contains the first 16 bytes of the event name (a string). By default, the LAN event names are defined as LAN0 to LAN7.

To transmit LAN trigger packets to or from a DT8824 instrument module, ensure that you set the event ID of each DT8824 instrument module to the same value using the **EVENT:LAN<n>:NAME** command.

- **Sequence** – Each instrument module maintains its own sequence; the sequence number is incremented every time a unique packet is transmitted. (Note that if packets are retransmitted to enhance reliability, re-transmitted packets contain the same sequence number as the original.)
- **Time stamp** – For DT8824 instrument modules, the time stamp of received LAN event packets is ignored, while generated LAN event packets have a time stamp of 0.
- **Epoch** – For DT8824 instrument modules, this value is 0.
- **Flags** – Contains data about the packet. This value is generated automatically by DT8824 instrument modules.
- **Data Fields** – Arbitrary number of bytes, up to the capacity of the data packet. This field is generated automatically by DT8824 instrument modules.

To set up a LAN trigger, do the following:

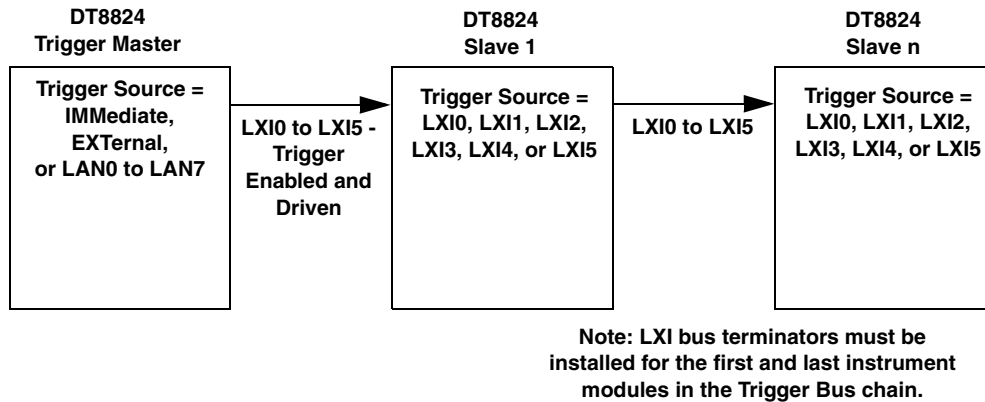
1. Set the trigger source to a LAN trigger packet (LAN0 to LAN7) using the **AD:TRIGger[:SOURce]** command.
2. Set the value of the LXI domain octet using the **EVENT:DOMain** command.
3. Enable the LAN event for transmission using the **EVENT:LAN<n>:TRANsmit[:ENABle]** command.

4. Configure the LAN event ID field of the LXI packet using the `EVENT:LAN<n>:NAME` command.
5. Arm the analog input subsystem, as described on [page 47](#).

Acquisition starts with the first sample after the specified LAN trigger packet is received.

## Trigger Bus

When multiple DT8824 instrument modules are connected together using the Trigger Bus, as shown in [14](#), you can synchronize the start of acquisition using one of six LXI trigger signals: LXI0 to LXI5.



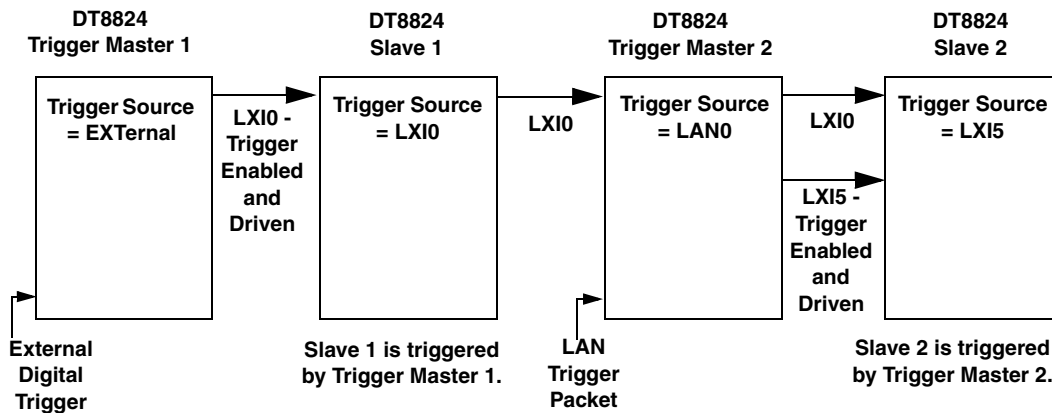
**Figure 14: Synchronizing the Start of Acquisition When Connecting Multiple Instrument Modules to the Trigger Bus**

---

**Note:** When using LXI triggers, the timing from device to device is guaranteed to be within one sample pulse.

---

Unlike when synchronizing the clock of multiple instrument modules, which requires one DT8824 instrument module dedicated as the clock master, you can have more than one trigger master, if desired, to trigger multiple instrument modules. Additionally, the trigger master need not be the clock master; for example, a device other than the DT8824 can generate the trigger for the DT8824 instrument modules, if desired. [15](#) shows an example of using two DT8824 trigger masters on the Trigger Bus.



Note: LXI bus terminators must be installed for the first and last instrument modules in the Trigger Bus chain.

Figure 15: An Example of Using Two Trigger Masters on the Trigger Bus

**Note:** Ensure that you set up the clocks (as described on [page 41](#)) before you set up the triggers. Arm the subsystems (as described on [page 47](#)) after both the clocks and triggers have been set up.

Set up the triggers as follows:

1. On the DT8824 trigger master, use the **AD:TRIGger[:SOURce]** command to set the trigger source to software trigger (IMMediate), external digital trigger (EXTERNAL), or LAN trigger packet (LAN0 to LAN7).
2. If the trigger source is EXTERNAL, configure the trigger edge using the **AD:TRIGger:EXTERNAL:EDGE** command.
3. If the trigger source is LAN0 to LAN7, configure the LAN event trigger, as follows:
  - a. Set the value of the LXI domain octet using the **EVENT:DOMain** command.
  - b. Enable the LAN event for transmission using the **EVENT:LAN<n>:TRANsmit[:ENABLE]** command.
  - c. Configure the LAN event ID field of the LXI packet using the **EVENT:LAN<n>:NAME** command.
4. On each DT8824 slave, use the **AD:TRIGger[:SOURce]** command to select the trigger signal (LXI0 to LXI5) to receive from the Trigger Bus.
5. On the DT8824 trigger master, use the **WTB:LXI<n>[:OUTput]:ENABLE** command to drive out the LXI trigger signal (LXI0 to LXI5) on the Trigger Bus.

---

**Note:** If you change the input trigger source on the master, you must call the **WTB:LXI<n>[:OUTput]:ENABLE** command again to ensure that the LXI trigger signal is driven out on the Trigger Bus.

While a trigger master can drive out more than one trigger signal on the Trigger Bus, the slave can accept only one trigger signal to start the analog input operation.

---

6. Arm the analog input subsystem on each slave, as described below.
7. Arm the analog input subsystem on the trigger master, as described below.
8. If the trigger source of the DT8824 trigger master is IMMEDIATE (software trigger), initiate the operation, as described on [page 47](#).

Once the master is triggered, the specified LXI trigger signal is driven out on the Trigger Bus to all slaves that were configured to accept it. Acquisition starts on the first sample after the trigger is received.

## Arming Input Operations

Once you have configured all the parameters of the analog input subsystem, call the **AD:ARM** command to arm the subsystem. The arming process writes all the input configuration settings to the DT8824 hardware.

Then, use the **AD:STATus?** command, described below, or read bit 10 of the Operation Status register using the **STATus:OPERation:CONDition?** command to verify that the arming process is complete.

## Starting Input Operations

If you are using the software (IMMEDIATE) trigger source and have armed the analog input subsystem, you must start the analog input operation by issuing the **AD:INITiate** command. For all other trigger sources, **AD:INITiate** is ignored.

When the software trigger is detected, the analog input subsystem simultaneously samples all the enabled input channels, and converts the analog inputs to voltage.

How the operation stops depends on the buffer wrap mode, described on [page 39](#).

## Determining the Status of the Analog Input Subsystem

You can use the **AD:STATus?** command to determine the status of the analog input subsystem. The returned value, in the range of 0 to 255, represents the status bits for the analog input subsystem.

The status bits are defined as follows:

- Bit 0 – The value of this bit is 1 if the analog input subsystem is active or 0 if the analog input subsystem is inactive.

- Bit 1 – The value of this bit is 1 if the analog input subsystem is armed or 0 if the analog input subsystem is not armed.
- Bit 2 – The value of this bit is 1 if the analog input subsystem is triggered or 0 if the analog input subsystem is not triggered.
- Bit 3 – The value of this bit is 1 if AD SYNC was detected or 0 if AD SYNC was not detected.
- Bit 4 – The value of this bit is 1 if the AD FIFO overflowed or 0 if the AD FIFO did not overflow.

## Retrieving Scan Data from the Input Buffer

To retrieve scan data from the input buffer, use the **AD:FETCh?** command.

---

**Note:** At any instant, up to 12 SCPI clients can retrieve a client-specific number of scans concurrently from the input buffer of the instrument module using the **AD:FETCh?** command.

Of the 12 SCPI clients, 4 can be VXI-11 clients, which use the VISA::INSTR resource to access the instrument module.

Up to 8 additional clients can access the DT8824 instrument modules concurrently using the web interface provided with the device. Refer to the documentation for your instrument module for more information.

At this time, the SCPI and web interfaces cannot be "locked;" therefore, one client can change the configuration of the instrument module that another client is accessing. However, you can optionally lock the VXI-11 interface using the VISA APIs viLock/viUnlock; this prevents other VXI-11 clients (including VXI-11 discovery) from accessing the instrument module.

---

The **AD:FETCh?** command takes two parameters:

- *index* – a required parameter that specifies where in the input buffer to begin reading the scan data
- *number of scans* – an optional parameter that specifies the number of scans to retrieve from the input buffer

By using *index* and *number of scans*, any client can request data at arbitrary intervals. To request all the data from the input buffer, set the *index* to 0 and omit the *number of scans* parameter.



---

**Note:** If the *number of scans* parameter is omitted, a fixed number of scan records is returned.

Since the maximum input buffer size is approximately 8 MB and each sample is 4 bytes, you can store a maximum of approximately 2 Msamples in the input buffer. If you sample each input channel at the maximum input frequency (4800 Hz), the input buffer will fill in approximately 106 seconds (4800 samples/s per channel).

The maximum size of data transfer over Ethernet using the **AD:FETCh?** command is 32 kB. Since each sample is 4 bytes, the maximum number of samples per **AD:FETCh?** is 8 ksamples. Therefore, if you are using continuous wrap mode, ensure that you call **AD:FETCh?** in a tight loop to ensure that you retrieve all the samples in the input buffer before the buffer is overwritten.

You can use the **AD:BUFFer:SIZE[:SCANs]?** query to determine how many scans can be stored in the input buffer based on the number of enabled input channels and the input sampling rate. Note that the sampling rate determines DMA block size that is used for the scan data, where each block has a fixed number of header bytes that is not used to store scans. The DMA block size is smaller for the minimum sampling frequency, resulting in more space for the headers, and greater for the maximum sampling frequency.

---

If you want to read only a specified number of scans (instead of all the scans available), specify the number of scans that you want to retrieve in the *number of scans* parameter.

If more than the requested number scans has been acquired, the DT8824 instrument module returns only the scans that you requested between the starting index (*index*) and the ending index (*index + number of scans*). Conversely, if fewer than the requested number of scans has been acquired, the instrument module returns the subset of the requested scans that are available between the starting index (*index*) and the ending index (*index + number of scans*).

If all the input channels are enabled, samples returned in the scan record are ordered as follows:

- Chan 1 – analog channel 1
- Chan 2 – analog channel 2
- Chan 3 – analog channel 3
- Chan 4 – analog channel 4

Reading data from the input buffer with **AD:FETCh?** does not destroy the data; therefore, you can read the same data multiple times, if desired, providing that the input buffer has not been overwritten.

If the client wants to read a number of scans that chronologically follow a previous response, get the *EndingIndex* returned by the **AD:STATus:SCAN?** command, described on [page 103](#), add one to this value, and set the *Index* parameter of the next **AD:FETCh?** command to this value. For example, if *EndingIndex* returned by **AD:STATus:SCAN?** is 2050 and you want to continue reading data from the location where you left off, set *Index* in the next **AD:FETCh?** command to 2051.

---

**Note:** The *Index* parameter of the **AD:FETCh?** command is an unsigned, 4-byte value with a maximum value of  $2^{32} - 1$  or 4294967295. The value of *Index* starts at 0 when data acquisition starts, increments to 4294967295, then rolls over to 0 and continues incrementing.

If you scanning with a sample frequency of 4800 Hz, the value *Index* will rollover to 0 in approximately ten days. For slower sampling frequencies, this process will take longer.

---

## Stopping Input Operations

To stop an input operation that is in progress, issue the **AD:ABORT** command.

Note that if the buffer mode is NOWRAP, the operation stops automatically when the input buffer has been filled. Refer to [page 39](#) for more information on buffer modes.

## Performing Digital Output Operations

DT8824 instrument modules feature four, isolated digital output lines. The digital outputs are solid-state relays that operate at  $\pm 30$  V and 400 mA peak (AC or DC). Switching time is 2 ms maximum. Digital outputs resemble a switch; the switch is closed if the state of the digital output line is 1, and the switch is open if the state of the digital output line is 0. On power up or reset, the digital outputs are disabled.

Use the **DOUTput** command to close or open a digital relay.

You can also change the value of specific digital output lines without affecting all of the digital output lines by performing the following bitwise operations:

- Use the **DOUTput:AND** command to compare the binary representations of a specified value between 0 and 15 and the current value of the digital output port, and perform a logical AND operation on each pair of corresponding bits. If both bits are 1, the result is 1. If both bits are 0, the result is 0. If one bit is 1 and the other bit 0, the result is 0.

The following example shows the result of a logical AND operation:

**Table 3:**

	0101
AND	0011
=	0001

- OR – Use the **DOUTput:OR** command to compare the binary representations of a specified value between 0 and 15 and the current value of the digital output port, and perform a logical OR operation on each pair of corresponding bits. If both bits are 1, the result is 1. If both bits are 0, the result is 0. If one bit is 1 and the other bit 0, the result is 1.

The following example shows the result of a logical OR operation:

**Table 4:**

	0101
OR	0011
=	0111

You can query the current configuration of the digital output port by using the **DOUTput?** command.





## ***Common SCPI Commands***

Clear Status (*CLS).....	54
Standard Event Status Enable Register (*ESE).....	54
Standard Event Status Enable Register Query (*ESE?).....	55
Standard Event Status Register Query (*ESR?).....	56
Identification Query (*IDN?).....	57
Operation Complete (*OPC).....	58
Operation Complete Query (*OPC?).....	58
Reset (*RST).....	59
Self-Test Query (*TST?).....	59
Service Request Enable (*SRE).....	59
Service Request Enable Query (*SRE?).....	60
Read Status Byte Query (*STB?).....	60
Wait-to-Continue (*WAI).....	62

**Clear Status (\*CLS)**

<b>Description</b>	Clears all event registers summarized in the Status Byte (STB) register, described on <a href="#">page 150</a> .
<b>Syntax</b>	*CLS
<b>Parameters</b>	None
<b>Response Data</b>	None
<b>Notes</b>	<p>This is a password-protected command. To enable this command to function, use the <b>SYSTem:PASSword[:CENable]</b> command, described on <a href="#">page 83</a>.</p> <p>All queues that are summarized in the Status Byte (STB) register, except the output queue, are emptied. The device is forced into the operation complete idle state and the operation complete query idle state.</p>
<b>Example</b>	> *CLS

**Standard Event Status Enable Register (\*ESE)**

<b>Description</b>	Enables specified bits in the Standard Event Status Enable register, described on <a href="#">page 151</a> .
<b>Syntax</b>	*ESE <DECIMAL NUMERIC PROGRAM DATA>
<b>Required Parameters</b>	
Name:	DECIMAL NUMERIC PROGRAM DATA
Data Format:	<0+NR1>
Description:	An integer value expressed in base 2 (binary) that represents the weighted bit value of the Standard Event Status Enable register. Values range from 0 to 255.
<b>Response Data</b>	None

**Notes** The following table shows the bits of the Standard Event Status Enable Register and the binary-weighted decimal value for each bit; see [page 151](#) for more information on each of these bits:

	Binary		
Bit	Weight		Description
0	1		OPC (Operation Complete)
1	2		RQC (Request Control)
2	4		QYE (Query ERROR)
3	8		DDE (Device-Dependent ERROR)
4	16		E (Execution ERROR)
5	32		CME (Command ERROR)
6	64		NU (Not Used)
7	128		PON (Power on)

Refer to *IEEE Std 488.2-1992*, section 11.5.1.3, for more information.

**Example** The following command enables bits 0, 2, 3, 4, 5, and 7 of the Standard Event Status Enable register:

```
> *ESE 189
```

**See Also** \*ESE?, described next  
\*ESR?, described on [page 56](#)

### Standard Event Status Enable Register Query (\*ESE?)

**Description** Returns the current value of the Standard Event Status Enable register, described on [page 151](#).

**Syntax** \*ESE?

**Parameters** None

**Response Data** <NUMERIC RESPONSE DATA>

Name: NUMERIC RESPONSE DATA

Data Format: <0+NR1>

Description: An integer value expressed in base 2 (binary) that represents the weighted bit value of the Standard Event Status Enable register. Values range from 0 to 255.

**Notes** Refer to *IEEE Std 488.2-1992*, section 11.4.2.3.2, and *IEEE Std 488.2*, section 8.7.1, for more information.

**Example** The following command queries the Standard Event Status Enable register:

```
> *ESE?
< 189
```

This value indicates that bits 0, 2, 3, 4, 5, and 7 of the Standard Event Status Enable register are enabled.

**See Also** \*ESE, described on [page 54](#)

\*ESR?, described on [page 56](#)

### Standard Event Status Register Query (\*ESR?)

**Description** Returns the current value of the Standard Event Status register, described on [page 153](#), and then clears the register.

**Syntax** \*ESR?

**Parameters** None

**Response Data** <NUMERIC RESPONSE DATA>

Name: NUMERIC RESPONSE DATA

Data Format: <0+NR1>

Description: An integer value expressed in base 2 (binary) that represents the weighted bit value of the Standard Event Status register.

The bits of the Standard Event Status Register are listed below along with the binary-weighted decimal value for each bit; refer to [page 153](#) for more information on each of these bits:

Binary		
Bit	Weight	Description
0	1	OPC (Operation Complete)
1	2	RQC (Request Control)
2	4	QYE (Query ERROR)
3	8	DDE (Device-Dependent ERROR)
4	16	E (Execution ERROR)
5	32	CME (Command ERROR)
6	64	NU (Not Used)
7	128	PON (Power on)

**Notes** Bits in the Standard Event Status register should be unmasked by setting the corresponding bit in the Standard Event Status Enable register. On power up, the Standard Event Status Enable register is '0'; therefore, all bits in the Standard Event Status register are masked. The summary of the Standard Event Status register is reflected in the Standard Event Status Bit Summary (ESB) of the Status Byte register, described on [page 150](#).



**Example** The following example unmask all error bits in the Standard Event Status register:

```
> *ESE?;*ESE 255;*ESE?
< 0 ; 255
```

Then, an illegal command is sent and the Standard Event Status register is queried; a value of 32 is returned, indicating that bit 5 (Command Error) of the Standard Event Status register was set:

```
> *bad
> *ESR?
< 32
```

In the following example, ESE is set to an invalid value, causing bit 4 (E) to be set:

```
> *ESE 65535
> *ESR?
< 16
```

**See Also** [\\*ESE](#), described on [page 54](#)  
[\\*ESE?](#), described on [page 55](#)  
[\\*STB?](#), described on [page 60](#)

### Identification Query (\*IDN?)

**Description** This command returns the unique identity of a DT8824 LXI instrument module.

**Syntax** \*IDN?

**Parameters** None

**Response Data** Manufacturer, Model, Serial number, Firmware revision

**Data Format** <ARBITRARY ASCII RESPONSE DATA>

Name: Manufacturer

Description: Defines the manufacturer of the instrument module. For DT8824 LXI instrument modules, this response is Data Translation.

Name: Model

Description: Identifies the model of the DT8824 LXI instrument module.

Name: Serial number

Description: Identifies the serial number of the instrument module.

Name: Firmware revision

Description: Identifies the version of firmware that is loaded on the instrument module.

**Notes** Since the data format of the response is <ARBITRARY ASCII RESPONSE DATA>, the **\*IDN?** query should be the last <QUERY MESSAGE UNIT> in a <TERMINATED PROGRAM MESSAGE>.

Refer to *IEEE 488.2-1992*, sections 6.5.7.5, 8.7.11 and 10.14, for more information.

**Example** > \*IDN?  
< Data Translation,DT8824,8029023,1.1

This response indicates that Data Translation is the manufacturer of the device, DT8824 is the model of the instrument module, 8029023 is the serial number of the instrument module, and 1.1 is the version of the firmware.

### Operation Complete (\*OPC)

**Description** The Operation Complete bit (bit 0) of the Standard Event Status register, described on [page 153](#), is always enabled. Therefore, this command has no effect when used with a DT8824 LXI instrument module.

**Syntax** \*OPC

**Parameters** None

**Response Data** None

**Notes** This command is implemented for SCPI compliance to avoid error messages when it is sent by the controller to the instrument module.

**Example** > \*OPC

**See Also** \*OPC?, described below

### Operation Complete Query (\*OPC?)

**Description** The Operation Complete bit (bit 0) of the Standard Event Status register, described on [page 153](#), is always enabled. Therefore, this command always places the ASCII character 1 into the device's Output Queue.

**Syntax** \*OPC?

**Parameters** None

**Response Data** <NUMERIC RESPONSE DATA>

Name: NUMERIC RESPONSE DATA

Data Format: <NR1>

Description: A single ASCII-encoded byte for 1 (31, 49 decimal).

**Example** > \*OPC?  
< 31

**Reset (\*RST)**

<b>Description</b>	Clears the Standard Event Status register, message queue, error queue, and Status Byte register, and stop any scans that are in progress.
<b>Syntax</b>	*RST
<b>Parameters</b>	None
<b>Response Data</b>	None
<b>Notes</b>	This is a password-protected command. To enable this command to function, use the <b>SYSTem:PASSWORD[:CENable]</b> command, described on <a href="#">page 83</a> . Refer to <i>IEEE 488.2-1992</i> , section 10.32, for more information.
<b>Example</b>	> *RST

**Self-Test Query (\*TST?)**

<b>Description</b>	Always returns 0 for DT8824 LXI instrument modules.
<b>Syntax</b>	*TST?
<b>Parameters</b>	None
<b>Response Data</b>	<NUMERIC RESPONSE DATA>
Name:	NUMERIC RESPONSE DATA
Data Format:	<NR1>
Description:	This value is always 0 for DT8824 LXI instrument modules.
<b>Notes</b>	This query has no effect. This command is implemented for SCPI compliance to avoid error messages when it is sent by the controller to the instrument module.
<b>Example</b>	> *TST? < 0

**Service Request Enable (\*SRE)**

<b>Description</b>	The Service Request Enable register is not used on these instrument modules. Therefore, this command has no effect when used with a DT8824 LXI instrument module.
<b>Syntax</b>	*SRE <value>
<b>Required Parameters</b>	
Name:	value
Data Format:	<0+NR1>
Description:	A positive decimal value; this value is ignored.
<b>Response Data</b>	None

<b>Notes</b>	This command is implemented for SCPI compliance to avoid error messages when it is sent by the controller to the instrument module.
<b>Example</b>	> *SRE 255
<b>See Also</b>	*SRE?, described below

### Service Request Enable Query (\*SRE?)

<b>Description</b>	The Service Request Enable register is not used on these instrument modules. Therefore, this command places the ASCII character 0 into the device's Output Queue.
<b>Syntax</b>	*SRE?
<b>Parameters</b>	None
<b>Response Data</b>	<NUMERIC RESPONSE DATA>
Name:	NUMERIC RESPONSE DATA
Data Format:	<NR1>
Description:	A single ASCII-encoded byte for 0.
<b>Notes</b>	This command is implemented for SCPI compliance to avoid error messages when it is sent by the controller to the instrument module.
<b>Example</b>	> *SRE? < 0

### Read Status Byte Query (\*STB?)

<b>Description</b>	Returns the current value of the Status Byte register, described on <a href="#">page 150</a> .
<b>Syntax</b>	*STB?
<b>Parameters</b>	None
<b>Response Data</b>	<NUMERIC RESPONSE DATA>
Name:	NUMERIC RESPONSE DATA
Data Format:	<NR1>
Description:	The weighted sum of the bit values of the Status Byte register, ranging from 0 to 255. The following bits, described in <i>1999 SCPI Syntax &amp; Style</i> , section 9, are supported by DT8824 LXI instrument modules: <ul style="list-style-type: none"><li>• Bit 7 (weighted bit value = 128) – Summary of device dependent Operation Status Register</li><li>• Bit 5 (weighted bit value = 32) – Event Status Bit Summary (ESB), '1'= ESR is non-zero, '0' otherwise</li></ul>

- Description (cont.):
- Bit 4 (weighted bit value = 16) – Message Available Queue Summary (MAV), '1'=message queue not empty
  - Bit 2 (weighted bit value = 4) – Error/Event Queue Summary, '1'=Error queue not empty

**Notes** Refer to *IEEE 488.2-1992*, section 10.36, for more information.

**Example** The following example shows a query that is correct and causes no errors:

```
> *IDN?;*ESR?;*STB?
< Data Translation,DT8824,-1,1.2;0;16
```

This example shows an illegal command being sent and the status of the Status Byte register and the error queue:

```
> bad
> *STB?
< 36
```

A value of 36 indicates that bits 5 (Standard Event Status Bit Summary) and 2 (Error / Event Queue Summary) of the Status Byte register are set.

The following example shows the status of the Event Status register:

```
> *ESR?
< 32
```

A value of 32 indicates that bit 5 (Command Error) of the Event Status register is set. The following updates the status of the Status Byte register:

```
> *STB?
< 4
```

A value of 4 indicates that bit 2 (Error / Event Queue Summary) of the Status Byte register is set. The following shows the error codes that are returned, and updates the status of the Status Byte register:

```
> :SYST:ERR?
< -110, "Command header error;bad"
> :SYST:ERR?
< 0, "No error"
> *STB?
< 0
```

### Wait-to-Continue (\*WAI)

<b>Description</b>	This command has no effect when used with DT8824 LXI instrument modules.
<b>Syntax</b>	*WAI
<b>Parameters</b>	None
<b>Response Data</b>	None
<b>Notes</b>	This command is implemented for SCPI compliance to avoid any error message when it is sent by the controller to the instrument module.
<b>Example</b>	> *WAI



# ***SCPI Subsystem Commands for the DT8824***

SCPI Subsystem Command Hierarchy .....	64
STATus Subsystem Commands .....	66
SYSTem Subsystem Commands .....	72
AD Subsystem Commands .....	90
WTB (Wired Trigger Bus) Subsystem Commands .....	116
EVENT Subsystem Command .....	120
DOUOutput Subsystem Commands .....	127

---

**Note:** In the examples in this section, the symbols > and < represent communication to and from the instrument module, respectively. Users do not explicitly send or receive these symbols.

---

## SCPI Subsystem Command Hierarchy

SCPI subsystem commands are hierarchical and are organized as an inverted tree structure with the "root" at the top. Table 5 shows the hierarchy of SCPI subsystem commands for DT8824 LXI instrument modules.

**Table 5: Hierarchy of SCPI Subsystem Commands for DT8824 LXI Instrument Modules**

STATus	:OPERation	[:EVENT]?	
		:CONDition?	
		:ENABle[?]	
	:PRESet		
	:QUEStionable	[:EVENT]?	
		:CONDition?	
		:ENABle[?]	
SYSTem	:DATE?		
	:DESCRiption[?]		
	:ERRor	:ALL?	
		[:NEXT]	
		:COUNt?	
		:CODE	:ALL?
			[:NEXT]?
	:PASSword	:NEW	
		:CDISable	
		[:CENable]	
		:CENable	:STATe?
	:COMMunicate	:NET	:IP[?]
			:MASK[?]
	:TIME?		
	:TZONE?		
	:VERSion?		
	:RANge	[:MAX]?	



**Table 5: Hierarchy of SCPI Subsystem Commands for DT8824 LXI Instrument Modules (cont.)**

AD	:ABORt		
	:ARM		
	:BUFFer	:MODE[?]	
		:SIze	[:SCAns]?
	:CLOCK	:FREQuency	[:CONFigure][?]
		:SOURce[?]	
	:ENABle[?]		
	:FETCh?		
	:GAIN	[:CONFigure][?]	
	:INITiate		
	:STATus	:SCAN?	
	:STATus?		
	:SYNc	[:SOURce][?]	
	:TRIGger	[:SOURce][?]	
	:FETCh?		
:INITiate			
:READ	:ENABle[?]		
:TRIGger	[:SOURce][?]		
:STATus?			
WTB	:LXI<n>	[:OUTput]	:ENABle[?]
EVENT	:DOMain[?]		
	:LAN<n>	:NAME[?]	
		:TRANsmit	:ENABle[?]
DOUTput	[?]		
	:AND		
	:OR		

Looking at the STATus subsystem, for example, STATus is the root keyword of the command, OPERation is a second-level keyword, and CONDition is a third-level keyword. The next section describes the syntax of program messages and SCPI commands from the controller to a DT8824 LXI instrument module.

## STATUS Subsystem Commands

The STATUS subsystem includes the commands listed in [Table 6](#) for determining the operational status of a DT8824 LXI instrument module. This section describes each of these commands in detail.

**Table 6: STATUS Subsystem Commands**

Type	Mnemonic	See
Operation Condition Register Query	STATUS:OPERation:CONDition?	<a href="#">page 67</a>
Operation Enable Register	STATUS:OPERation:ENABLE	<a href="#">page 67</a>
Operation Enable Register Query	STATUS:OPERation:ENABLE?	<a href="#">page 68</a>
Operation Event Register Query	STATUS:OPERation[:EVENT]?	<a href="#">page 68</a>
Presetting Registers	STATUS:PRESet	<a href="#">page 69</a>
Questionable Condition Register Query	STATUS:QUESTionable:CONDition?	<a href="#">page 69</a>
Questionable Enable Register	STATUS:QUESTionable:ENABLE	<a href="#">page 70</a>
Questionable Enable Register Query	STATUS:QUESTionable:ENABLE?	<a href="#">page 70</a>
Questionable Event Register Query	STATUS:QUESTionable[:EVENT]?	<a href="#">page 70</a>

## Operation Condition Register Query

**Description** Returns the current value of the Operation Status register, described on [page 154](#).

**Syntax** :STATus:OPERation:CONDition?

**Parameters** None

**Response Data** <NUMERIC RESPONSE DATA>

Name: NUMERIC RESPONSE DATA

Data Format: <0+NR1>

Description: The weighted bit value of the Operation Status register. Values for the response range from 0 to 32767.

**Notes** You can query the bits of the Operation Status register to determine whether the input and output buffers are full or empty, and whether the input and output triggers have occurred.

**Example** The following example shows the status of the Operation Status register and the Status Byte register when the analog input subsystem is waiting to be armed:

```
> :STAT:OPER:COND?
< 64
> *STB?
< 128
```

A value of 64 indicates that bit 6 (decimal value 64) of the Operation Status registers is set to 1. A value of 128 indicates that bit 7 (Operation Status Register Summary) of the Status Byte register is set to 1.

The following example shows the status of the Operation Status register and the Status Byte register when the analog input subsystem is waiting for a trigger:

```
> :STAT:OPER:COND?
< 32
> *STB?
< 128
```

A value of 32 indicates that bit 5 (decimal value 32) of the Operation Status registers is set to 1. A value of 128 indicates that bit 7 (Operation Status Register Summary) of the Status Byte register is set to 1.

## Operation Event Register Enable

**Description** The Operation Event register is not used on DT8824 LXI instrument modules; therefore, this command has no effect on a DT8824 LXI instrument module.

**Syntax** :STATus:OPERation:ENABle <bitmask>

**Required Parameters**

Name:	bitmask
Data Format:	<NRr>
Description:	A non-decimal numeric value; this value is ignored by DT8824 LXI instrument modules.

**Response Data** None

**Notes** This command is implemented for SCPI compliance to avoid error messages when it is sent by the controller to the instrument module.

**Example** > :STAT:OPER:ENAB #B00000000

**See Also** **STATus:OPERation:CONDition?**, described on [page 67](#)

**Operation Enable Register Query**

**Description** The Operation Enable register is not used on DT8824 LXI instrument modules; therefore, this query always returns 0.

**Syntax** :STATus:OPERation:ENAB?

**Parameters** None

**Response Data** <bitmask>

Name:	bitmask
Data Format:	<0+NR1>
Description:	The Operation Enable register is not used. Therefore, this value is always 0.

**Notes** This command is implemented for SCPI compliance to avoid error messages when it is sent by the controller to the instrument module.

**Example** > :STAT:OPER:ENAB?  
< 0

**See Also** **STATus:OPERation:CONDition?**, described on [page 67](#)

**Operation Event Register Query**

**Description** The Operation Event register is not used on DT8824 LXI instrument modules; therefore, this query always returns 0.

**Syntax** :STATus:OPERation[:EVENT]?

**Parameters** None

**Response Data** <NUMERIC RESPONSE DATA>

<b>Name:</b>	NUMERIC RESPONSE DATA
<b>Data Format:</b>	<0+NR1>
<b>Description:</b>	The Operation Event register is not used. Therefore, the value of this response is always 0.
<b>Notes</b>	This command is implemented for SCPI compliance to avoid error messages when it is sent by the controller to the instrument module.
<b>Example</b>	> :STAT:OPER:EVEN? < 0
<b>See Also</b>	STATus:OPERation:CONDition?, described on <a href="#">page 67</a>

### Presetting Registers

**Description** This command has no effect when used with a DT8824 LXI instrument module.

**Syntax** :STATus:PRESet <value>

#### Required Parameters

**Name:** value

**Data Format:** <NR1>

**Description:** For DT8824 LXI instrument modules, this value is ignored.

**Response Data** None

**Notes** This command is implemented for SCPI compliance.

**Example** > :STAT:PRESet 1

### Questionable Condition Register Query

**Description** The Questionable Condition register is not used on DT8824 LXI instrument modules; therefore, this query always returns 0.

**Syntax** :STATus:QUESTionable:CONDition?

**Parameters** None

**Response Data** <regvalue>

**Name:** regvalue

**Data Format:** <0+NR1>

**Description:** The Questionable Condition register is not used. Therefore, the value of this response is always 0.

**Notes** This command is implemented for SCPI compliance to avoid error messages when it is sent by the controller to the instrument module.

**Example** > :STAT:QUES:COND?  
< 0

**See Also**    `STATus:OPERation:CONDition?`, described on [page 67](#)

### Questionable Enable Register

**Description**    The Questionable Enable register is not used on DT8824 LXI instrument modules. Therefore, this command has no effect when used with a DT8824 LXI instrument module.

**Syntax**        `:STATus:QUESTionable:ENABLE <bitmask>`

#### Required Parameters

Name:            `bitmask`

Data Format:     `<NRr>`

Description:    A non-decimal numeric value; this value is ignored by DT8824 LXI instrument modules.

**Response Data**    None

**Notes**            This command is implemented for SCPI compliance to avoid error messages when it is sent by the controller to the instrument module.

**Example**         `> :STAT:QUES:ENAB #B00000000`

### Questionable Enable Register Query

**Description**    The Questionable Enable register is not used on DT8824 LXI instrument modules; therefore, this query always returns 0.

**Syntax**        `:STATus:QUESTionable:ENABLE?`

**Parameters**    None

**Response Data**    `<bitmask>`

Name:            `bitmask`

Data Format:     `<NR1>`

Description:    The Questionable Enable register is not used. Therefore, the value of this response is always 0.

**Notes**            This command is implemented for SCPI compliance to avoid error messages when it is sent by the controller to the instrument module.

**Example**         `> :STAT:QUES:ENAB?  
< 0`

### Questionable Event Register Query

**Description**    The Questionable Event register is not used on DT8824 LXI instrument modules; therefore, this query always returns 0.

**Syntax**        `:STATus:QUESTionable[:EVENT]?`

<b>Parameters</b>	None
<b>Response Data</b>	<regvalue>
Name:	regvalue
Data Format:	<0+NR1>
Description:	The Questionable Event register is not implemented. Therefore, the value of this response is always 0.
<b>Notes</b>	This command is implemented for SCPI compliance to avoid error messages when it is sent by the controller to the instrument module.
<b>Example</b>	> :STAT:QUES:EVENT? < 0
<b>See Also</b>	STATus:OPERation:CONDition?, described on <a href="#">page 67</a>

## SYSTem Subsystem Commands

The SYSTem subsystem includes the commands listed in [Table 7](#). Use these commands to calibrate DT8824 LXI instrument modules, query the status of scan records on DT8824 LXI instrument modules, and configure or query global system settings, including the time, date, time zone, and network address of the instrument module. This section describes each of these commands in detail.

**Table 7: SYSTem Subsystem Commands**

Type	Mnemonic	See
Date Query	SYSTem:DATE?	<a href="#">page 73</a>
Description	SYSTem:DESCRiption	<a href="#">page 73</a>
Description Query	SYSTem:DESCRiption?	<a href="#">page 74</a>
Error Query All	SYSTem:ERRor:ALL?	<a href="#">page 74</a>
Error Query All Codes	SYSTem:ERRor:CODE:ALL?	<a href="#">page 75</a>
Error Query Count	SYSTem:ERRor:COUNT?	<a href="#">page 76</a>
Error Query Next	SYSTem:ERRor[:NEXT]?	<a href="#">page 77</a>
Error Query Next Code	SYSTem:ERRor:CODE[:NEXT]?	<a href="#">page 78</a>
LAN Static IP - Address Configuration	SYSTem:COMMunicate:NETwork:IPAddr	<a href="#">page 79</a>
LAN Static IP - Address Query	SYSTem:COMMunicate:NETwork:IPAddr?	<a href="#">page 80</a>
LAN Static IP - Subnet Mask Configuration	SYSTem:COMMunicate:NETwork:MASK	<a href="#">page 80</a>
LAN Static IP - Subnet Mask Query	SYSTem:COMMunicate:NETwork:MASK?	<a href="#">page 81</a>
Password - Disable Password-Protected Commands	SYSTem:PASSword:CDISable	<a href="#">page 81</a>
Password - Enable Password-Protected Commands	SYSTem:PASSword[:CENable]	<a href="#">page 83</a>
Password - Query Enable State	SYSTem:PASSword:CENable:STATE?	<a href="#">page 85</a>
Password - Set New Password	SYSTem:PASSword:NEW	<a href="#">page 86</a>
SCPI Version Query	SYSTem:VERSion?	<a href="#">page 87</a>
Time Query	SYSTem:TIME?	<a href="#">page 88</a>
Time Zone Query	SYSTem:TZONE?	<a href="#">page 89</a>
Voltage Range	SYSTem:RANge[:MAX]?	<a href="#">page 89</a>



## Date Query

**Description** Returns the current date of the DT8824 LXI instrument module. This date is updated automatically by an SNTP (Simple Network Time Protocol) server.

**Syntax** :SYSTem:DATE?

**Parameters** None

**Response Data** <year>, <month>, <day>

Name: year

Data Format: <NR1>

Description: A number representing the year, such as 2009.

Name: month

Data Format: <NR1>

Description: A number from 1 to 12 representing the month.

Name: day

Data Format: <NR1>

Description: A number from 1 to 31 representing the day.

**Example**  
 > :SYST:DATE?  
 < 2009,1,15

This response indicates that the date of the DT8824 instrument module is January 15th, 2009.

## Description

**Description** Specifies a description for the instrument module.

**Syntax** :SYSTem:DESCRiption <device\_description>

**Parameters** <device\_description>

**Response Data** None

Name: device\_description

Data Format: <CHARACTER PROGRAM DATA>

Description: A string with a maximum length of 64 ASCII characters, including the terminating NULL. Strings exceeding this limited are truncated.

**Notes** This is a password-protected command. To enable this command to function, use the **SYSTem:PASSword[:CENable]** command, described on [page 83](#).

**Example**  
 > :SYST:DESC  
 < The DT8824 in the lab

This command sets the description of the DT8824 to "The DT8824 in the lab".

## Description Query

**Description** Returns the description of the instrument module.

**Syntax** :SYSTem:DESCRiption?

**Parameters** None

**Response Data** <device\_description>

**Data Format** <CHARACTER RESPONSE DATA>

Name: device\_description

Description: A string with a maximum length of 64 ASCII characters.

**Example**  
> :SYST:DESC?  
< The DT8824 in the lab

In this case, the description of the device is as follows: "The DT8824 in the lab".

## Error Query All

**Description** Queries the error/event queue for all the unread errors and removes them from the Error/Event Queue.

**Syntax** :SYSTem:ERRor:ALL?

**Parameters** None

**Response Data** <Error/event\_number>, <Error/event\_description>,  
<Device-dependent info>

**Data Format** <CHARACTER RESPONSE DATA>

Name: Error/event\_number

Description: A unique integer in the range of -32768 to 32767. A value of 0 indicates that no error or event has occurred. Refer to [Appendix A](#) starting on [page 143](#) for a list of errors that can be returned.

Name: Error/event\_description

Description: A quoted string that contains a description of the error that was read from the Error/Event Queue. Refer to [Appendix A](#) starting on [page 143](#) for a list of errors that can be returned.

Name: Device-dependent info

Description: Optional text that provides device-dependent information about the error that was read from the Error/Event Queue. Refer to [Appendix A](#) starting on [page 143](#) for a list of errors that can be returned.

**Notes** The maximum string length of `<Error/event_description>` plus `<Device-dependent info>` is 255 characters.

The error queue is a first-in, first-out (FIFO) with a capacity of 32 error messages. By querying the error count before and after a SCPI command (in a single command string), you can unambiguously determine whether the command that you issued caused an error. Use the **SYSTem:ERRor:COUnT?** command, described on [page 76](#), to return the number of unread items in the error queue.

When the queue is full, the message `"-350,'Queue overflow'"` is returned and subsequent errors can not be added to the queue. Use the **\*CLS** command, described on [page 76](#), to clear the error queue as well as the Status Byte register.

Refer to *1999 SCPI Command Reference*, section 21.8.4, for more information.

**Example** The following shows the responses to this query after an invalid command is sent to the instrument module:

```
>:BADc
>:SYST:ERR:ALL?
< -110, "Command header error;:BADc"
> :SYST:ERR:ALL?
< 0, "No error"
```

**See Also** **SYSTem:ERRor[:NEXt]?**, described on [page 77](#)  
**SYSTem:ERRor:COUnT?**, described on [page 76](#)  
**SYSTem:ERRor:CODE:ALL?**, described on [page 75](#)  
**SYSTem:ERRor:CODE[:NEXt]?**, described on [page 78](#)

## Error Query All Codes

**Description** Returns a comma-separated list of only the error/event code numbers in FIFO order. If the queue is empty, the response is 0.

**Syntax** `:SYSTem:ERRor:CODE:ALL?`

**Parameters** None

**Response Data** `<Error/event_number>`,  
`<Error/event_number>`, . . . . ,  
`<Error/event_number>`

**Data Format** `<CHARACTER RESPONSE DATA>`

Name: Error/event\_number

Description: A unique integer in the range of -32768 to 32767. A value of 0 indicates that no error or event has occurred. Refer to [Appendix A](#) starting on [page 143](#) for a list of errors that can be returned.

**Notes** Refer to *1999 SCPI Command Reference*, section 21.8.5.1 for more information.

**Example** The following shows the responses to this query after an invalid command is sent to the instrument module:

```
> :BADc
> :SYST:ERR:CODE:ALL?
< -110,0;
```

**See Also** **SYSTem:ERRor:ALL?**, described on [page 74](#)  
**SYSTem:ERRor[:NEXT]?**, described on [page 77](#)  
**SYSTem:ERRor:COUNT?**, described on [page 76](#)  
**SYSTem:ERRor:CODE[:NEXT]?**, described on [page 78](#)

## ERRor Query Count

<b>Description</b>	Queries the Error/Event Queue for the number of unread items and returns the count.
<b>Syntax</b>	:SYSTem:ERRor:COUNT?
<b>Parameters</b>	None
<b>Response Data</b>	<Count>
Name:	Count
Data Format:	<0+NR1>
Description:	A unique integer that indicates the number of unread errors in the error queue. A value of 0 indicates that the queue is empty.  Since errors may occur at any time, more items may be present in the error queue at the time that it is actually read.
<b>Notes</b>	The error queue is a first-in, first-out (FIFO) with a capacity of 32 error messages. The error queue accumulates errors from all clients. By querying the error count before and after a SCPI command (in a single program message), you can unambiguously determine whether the command that you issued caused an error.  When the queue is full, the message "-350,'Queue overflow'" is returned and subsequent errors can not be added to the queue. Use the <b>SYSTem:ERRor:ALL?</b> command, described on <a href="#">page 74</a> , to read the error and remove it from the queue. Use the <b>*CLS</b> command, described on <a href="#">page 54</a> , to clear the error queue as well as the Status Byte register.  Refer to <i>1999 SCPI Command Reference</i> , section 21.8.6 for more information.

**Example** The following shows how to query the number of errors in the error queue and the error count:

```
> :BADc
> :SYST:ERR:COUNT?
< 1
> :SYST:ERR?
< -110, "Command header error;:BADc"
> :SYST:ERR?
< 0, "No error"
> :SYST:ERR:COUNT?
< 0
```

**See Also** **SYSTem:ERRor:ALL?**, described on [page 74](#)  
**SYSTem:ERRor[:NEXT]?**, described on [page 77](#)  
**SYSTem:ERRor:CODE:ALL?**, described on [page 75](#)  
**SYSTem:ERRor:CODE[:NEXT]?**, described on [page 78](#)

## Error Query Next

**Description** Queries the next error and removes it from the Error/Event Queue.

**Syntax** :SYSTem:ERRor[:NEXT]?

**Parameters** None

**Response Data** <Error/event\_number>,  
 <Error/event\_description>, <Device-dependent info>

**Data Format** <CHARACTER RESPONSE DATA>

**Name:** Error/event\_number

**Description:** A unique integer in the range of -32768 to 32767. A value of 0 indicates that no error or event has occurred. Refer to [Appendix A](#) starting on [page 143](#) for a list of errors that can be returned.

**Name:** Error/event\_description

**Description:** A quoted string that contains a description of the error that was read from the Error/Event Queue. Refer to [Appendix A](#) starting on [page 143](#) for a list of errors that can be returned.

**Name:** Device-dependent info

**Description:** Optional text that provides device-dependent information about the error that was read from the Error/Event Queue. Refer to [Appendix A](#) starting on [page 143](#) for a list of errors that can be returned.

**Notes** The maximum string length of `<Error/event_description>` plus `<Device-dependent info>` is 255 characters.

The error queue is a first-in, first-out (FIFO) with a capacity of 32 error messages. By querying the error count before and after a SCPI command (in a single command string), you can unambiguously determine whether the command that you issued caused an error. Use the **SYSTEM:ERROR:COUNT?** command, described on [page 76](#), to return the number of unread items in the error queue.

When the queue is full, the message "-350, Queue overflow" is returned and subsequent errors can not be added to the queue. Use the **\*CLS** command, described on [page 54](#), to clear the error queue as well as the Status Byte register.

Refer to *1999 SCPI Command Reference*, section 21.8.5.2, for more information.

**Example** The following shows the responses to this query after an invalid command is sent to the instrument module:

```
>:BADc
>:SYST:ERR?
< -110,"Command header error;:BADc"
> :SYST:ERR?
< 0,"No error"
```

**See Also** **SYSTEM:ERROR:ALL?**, described on [page 74](#)  
**SYSTEM:ERROR:COUNT?**, described on [page 76](#)  
**SYSTEM:ERROR:CODE:ALL?**, described on [page 75](#)  
**SYSTEM:ERROR:CODE[:NEXT]?**, described on [page 78](#)

## Error Query Next Code

**Description** Queries the next error code and removes it from the Error/Event Queue.

**Syntax** `:SYSTEM:ERROR:CODE[:NEXT]?`

**Parameters** None

**Response Data** `<Error/event_number>`

**Data Format** `<CHARACTER RESPONSE DATA>`

Name: Error/event\_number

Description: A unique integer in the range of -32768 to 32767. A value of 0 indicates that no error or event has occurred. Refer to [Appendix A](#) starting on [page 143](#) for a list of errors that can be returned.

**Notes** Refer to *1999 SCPI Command Reference*, section 21.8.5.2 for more information.

**Example** The following shows the responses to this query after an invalid command is sent to the instrument module:

```
> :BADc
> :SYST:ERR:CODE:NEXT?
< -110
> :SYST:ERR:CODE:NEXT?
< 0
```

**See Also** **SYSTem:ERRor:ALL?**, described on [page 74](#)  
**SYSTem:ERRor[:NEXT]?**, described on [page 77](#)  
**SYSTem:ERRor:COUNt?**, described on [page 76](#)  
**SYSTem:ERRor:CODE:ALL?**, described on [page 75](#)

## LAN Static IP - IP Address Configuration

**Description** Configures the static IP address that is used by the DT8824 instrument module on the network.

**Syntax** :SYSTem:COMMunicate:NETwork:IPAddr <ipaddr>

### Required Parameters

Name: ipaddr

Data Format: <0+NR1>, <CHARACTER PROGRAM DATA>

Description: Specifies the IP address of the DT8824 instrument module. The address must be four decimal numbers in the range 0-255, separated by periods, such as "192.43.218.59".

**Response Data** None

**Notes** In addition to the static IP address, you must also set the subnet mask using the **SYSTem:COMMunicate:NETwork:MASk** command, described on [page 80](#).

The DT8824 instrument module is configured to use DHCP with Auto-IP, by default.

**Example** This command sets the IP address of the DT8824 instrument module to 192.43.218.59.

```
> :SYST:COMM:NET:IP 192.43.218.59
```

**See Also** **SYSTem:COMMunicate:NETwork:IPAddr?**, described on [page 80](#)  
**SYSTem:COMMunicate:NETwork:MASk**, described on [page 80](#)  
**SYSTem:COMMunicate:NETwork:MASk?**, described on [page 81](#)

## LAN Static IP - IP Address Query

<b>Description</b>	Returns the IP address that is currently used by the DT8824 instrument module on the network, regardless of the method (DHCP, Auto-IP, static IP) used to acquire the address.
<b>Syntax</b>	:SYSTem:COMMunicate:NETwork:IPaddr?
<b>Response Data</b>	<ipaddr>
Name:	ipaddr
Data Format:	<0+NR1>, <CHARACTER PROGRAM DATA>
Description:	The IP address of the DT8824 instrument module. The address must be four decimal numbers in the range 0-255, separated by periods, such as "192.43.218.59".
<b>Notes</b>	The DT8824 instrument module is configured to use the DHCP server, by default.
<b>Example</b>	This command returns the IP address of the DT8824 instrument module, which in this example is 192.43.218.15.  <pre>&gt; :SYST:COMM:NET:IP? &lt; 192.43.218.59</pre>
<b>See Also</b>	<b>SYSTem:COMMunicate:NETwork:IPaddr</b> , described on <a href="#">page 79</a> <b>SYSTem:COMMunicate:NETwork:MASK</b> , described on <a href="#">page 80</a> <b>SYSTem:COMMunicate:NETwork:MASK?</b> , described on <a href="#">page 81</a>

## LAN Static IP - Subnet Mask Configuration

<b>Description</b>	Configures the subnet mask that is used by the DT8824 instrument module on the network.
<b>Syntax</b>	:SYSTem:COMMunicate:NETwork:MASK <mask>
<b>Required Parameters</b>	
Name:	mask
Data Format:	<0+NR1>, <CHARACTER PROGRAM DATA>
Description:	Specifies the subnet mask of the DT8824 instrument module. The subnet mask must be four decimal numbers in the range 0 to 255, separated by periods, such as "255.255.255.0".
<b>Response Data</b>	None
<b>Notes</b>	In addition to the subnet mask, you must also set the static IP address using the <b>SYSTem:COMMunicate:NETwork:IP</b> command, described on <a href="#">page 79</a> .
<b>Example</b>	This command sets the subnet mask of the DT8824 instrument module to 255.255.255.0:  <pre>&gt; :SYST:COMM:NET:MASK 255.255.255.0</pre>



- See Also**    **SYSTem:COMMunicate:NETwork:IP**, described on [page 79](#)  
**SYSTem:COMMunicate:NETwork:IP?**, described on [page 80](#)  
**SYSTem:COMMunicate:NETwork:MASk?**, described on [page 81](#)

### LAN Static IP - Subnet Mask Query

**Description**    Returns the subnet mask that is currently used by the DT8824 instrument module on the network, regardless of the method (DHCP, Auto-IP, static IP) used to acquire the address.

**Syntax**        :SYSTem:COMMunicate:NETwork:MASk?

**Response Data**    <mask>

Name:            mask

Data Format:      <0+NR1>, <CHARACTER PROGRAM DATA>

Description:     The subnet mask of the DT8824 instrument module. The subnet mask must be four decimal numbers in the range 0-255, separated by periods, such as "255.255.255.0".

**Example**        This command returns the subnet mask of the DT8824 instrument module, which in this example is 255.255.255.0:

```
> :SYST:COMM:NET:MAS?
< 255.255.255.0
```

- See Also**    **SYSTem:COMMunicate:NETwork:IP**, described on [page 79](#)  
**SYSTem:COMMunicate:NETwork:IP?**, described on [page 80](#)  
**SYSTem:COMMunicate:NETwork:MASk**, described on [page 80](#)  
**SYSTem:COMMunicate:NETwork:MASk?**, described on [page 81](#)

### Password - Disable Password-Protected Commands

**Description**    Disables use of the following commands, which are password protected:

- **AD:BUFFer:MODE**, described on [page 93](#)
- **AD:ENABle**, described on [page 96](#)
- **AD:GAIN[:CONFIgure]**, described on [page 99](#)
- **[AD:]CLOCK:SOURce**, described on [page 97](#)
- **AD:CLOCK:FREQUency[:CONFIgure]**, described on [page 101](#)
- **AD:TRIGger[:SOURce]**, described on [page 105](#)
- **AD:TRIGger:EXTernal:EDGE**, described on [page 109](#)
- **AD:ARM**, described on [page 92](#)

- Description (cont.)**
- **AD:INITiate**, described on [page 114](#)
  - **AD:ABORt**, described on [page 91](#)
  - **DOUtpuT**, described on [page 130](#)
  - **DOUtpuT:AND**, described on [page 128](#)
  - **DOUtpuT:OR**, described on [page 129](#)
  - **EVENT:DOMain**, described on [page 121](#)
  - **EVENT:LAN<n>:NAME**, described on [page 122](#)
  - **EVENT:LAN<n>:TRANsmiT[:ENABle]**, described on [page 125](#)
  - **WTB:LXI<n>[:OUtpuT]:ENABle**, described on [page 117](#)
  - **SYSTem:DESCRiption**, described on [page 73](#)
  - **\*RST**, described on [page 59](#)
  - **\*CLS**, described on [page 54](#)

**Syntax** :SYSTem:PASSword:CDISable <password>

**Parameters**

Name: password  
 Data Format: <CHARACTER PROGRAM DATA>  
 Description: Specifies the password that is stored in permanent memory on the instrument. This string is case-sensitive.

**Response Data** None

**Notes** On power up, all SCPI password-protected commands are disabled. If the instrument is powered down, you must enable the password-protected commands when the instrument is powered back up if you want to configure or operate the instrument.

Once the password-protected commands are disabled, issuing any of the password-protected commands will result in error -203, Command protected.

By default, the password is *admin*. You can change the password using the **SYSTem:PASSword:NEW** command, described on [page 86](#).

You can re-enable password-protected commands using the **SYSTem:PASSword[:CENable]** command, described on [page 83](#).

**Example** The following example shows an attempt to disable protected commands using an erroneous password:

```
> :SYSTem:PASSword:CDISable bogus
> :SYST:ERR?
< -221, "Settings conflict;:SYST:PASS:CDIS"
```

The following is an example in which the correct password is used to disable protected commands:

```
> :SYSTem:PASSword:CDISable admin
> :SYST:ERR?
< 0, "No error"
> :SYSTem:PASSword:CENable:STATE?
< 0
```

Since protected commands are disabled, any attempt to use them will fail, as shown in the following example:

```
> :AD:GAIN 1, (@1,2)
> :SYST:ERR?
< -203, "Command protected;AD:GAIN"
> :AD:INIT
> :SYST:ERR?
> -203, "Command protected;AD:INIT"
```

**See Also** **SYSTem:PASSword:NEW**, described on [page 86](#)  
**SYSTem:PASSword[:CENable]**, described on [page 83](#)  
**SYSTem:PASSword:CENable:STATE?**, described on [page 85](#)

## Password - Enable Password-Protected Commands

**Description** Enables use of the following commands, which are password protected:

- **AD:BUFFer:MODE**, described on [page 93](#)
- **AD:ENABle**, described on [page 96](#)
- **AD:GAIN[:CONFiGure]**, described on [page 99](#)
- **[AD:]CLOCK:SOURce**, described on [page 97](#)
- **AD:CLOCK:FREQuency[:CONFiGure]**, described on [page 101](#)
- **AD:TRIGger[:SOURce]**, described on [page 105](#)
- **AD:TRIGger:EXTernal:EDGE**, described on [page 109](#)
- **AD:ARM**, described on [page 92](#)
- **AD:INITiate**, described on [page 114](#)
- **AD:ABORt**, described on [page 91](#)
- **DOUtpuT**, described on [page 130](#)
- **DOUtpuT:AND**, described on [page 128](#)

- Description (cont.)**
- **DOUtput:OR**, described on [page 129](#)
  - **EVENT:DOMain**, described on [page 121](#)
  - **EVENT:LAN<n>:NAME**, described on [page 122](#)
  - **EVENT:LAN<n>:TRANsmit[:ENABLE]**, described on [page 125](#)
  - **WTB:LXI<n>[:OUtput]:ENABLE**, described on [page 117](#)
  - **SYSTem:DESCRiption**, described on [page 73](#)
  - **\*RST**, described on [page 59](#)
  - **\*CLS**, described on [page 54](#)

**Syntax** :SYSTem:PASSword[:CENable] <password>

**Parameters**

Name: password

Data Format: <CHARACTER PROGRAM DATA>

Description: Specifies the password that is stored in permanent memory in the instrument. This string is case-sensitive.

**Response Data** None

**Notes** On power up, the SCPI password-protected commands are disabled. If the instrument is powered down, you must enable the password-protected commands when the instrument is powered back up if you want to configure or operate the instrument.

Once the password-protected commands are disabled, issuing any of the password-protected commands will result in error -203, Command protected.

By default, the password is *admin*. You can change the password using the **SYSTem:PASSword:NEW** command, described on [page 86](#).

**Example** In the following example, the command to enable password-protected commands is successful; therefore, the client can configure the instrument and operate it.:

```
> :SYST:PASS:CEN admin
> :SYST:ERR?
< 0, "No error"
> :SYSTem:PASSword:CENable:STATe?
< 1
```

**See Also** **SYSTem:PASSword:NEW**, described on [page 86](#)

**SYSTem:PASSword:CDISable**, described on [page 81](#)

**SYSTem:PASSword:CENable:STATe?**, described on [page 85](#)

## Password - Query Password Enable State

<b>Description</b>	Returns whether password-protected commands are enabled or disabled.
<b>Syntax</b>	:SYSTem:PASSword:CENable:STATe?
<b>Parameters</b>	None
<b>Response Data</b>	<state>
Name:	state
Data Format:	<0+NR1>
Description:	If password-protected commands are disabled, 0 is returned. If password-protected commands are enabled, 1 is returned.
<b>Notes</b>	<p>On power up, the SCPI password-protected commands and queries are disabled. If the instrument is powered down, you must enable the password-protected commands when the instrument is powered back up if you want to configure or operate the instrument.</p> <p>You can disable password-protected commands using the <b>SYSTem:PASSword:DISable</b> command, described on <a href="#">page 81</a>.</p> <p>You can enable password-protected commands using the <b>SYSTem:PASSword[:CENable]</b> command, described on <a href="#">page 83</a>.</p> <p>When the SCPI password-protected commands and queries are disabled, the instrument generates the following error if any of these commands is issued: -203, Command protected.</p> <p>This error indicates that a legal password-protected command or query could not be executed because the command was disabled.</p>
<b>Example</b>	<p>In the following example, the state is 1, indicating that password-protected commands are enabled.</p> <pre>&gt; :SYST:PASS:CEN:STAT? &lt; 1</pre>
<b>See Also</b>	<p><b>SYSTem:PASSword:NEW</b>, described on <a href="#">page 86</a></p> <p><b>SYSTem:PASSword:CDISable</b>, described on <a href="#">page 81</a></p> <p><b>SYSTem:PASSword[:CENable]</b>, described on <a href="#">page 83</a></p>

## Password - Set New Password

**Description** Changes the existing password to a new password. The new password overwrites the existing password if, and only if, the user-specified *<current password>* matches the password that is currently being used.

The password is used with the following password-protected commands:

- **AD:BUFFer:MODE**, described on [page 93](#)
- **AD:ENABle**, described on [page 96](#)
- **AD:GAIN[:CONFigure]**, described on [page 99](#)
- **[AD:]CLOCK:SOURce**, described on [page 97](#)
- **AD:CLOCK:FREQUency[:CONFigure]**, described on [page 101](#)
- **AD:TRIGger[:SOURce]**, described on [page 105](#)
- **AD:TRIGger:EXTernal:EDGE**, described on [page 109](#)
- **AD:ARM**, described on [page 92](#)
- **AD:INITiate**, described on [page 114](#)
- **AD:ABORt**, described on [page 91](#)
- **DOUtpuT**, described on [page 130](#)
- **DOUtpuT:AND**, described on [page 128](#)
- **DOUtpuT:OR**, described on [page 129](#)
- **EVENT:DOMain**, described on [page 121](#)
- **EVENT:LAN<n>:NAME**, described on [page 122](#)
- **EVENT:LAN<n>:TRANsmiT[:ENABle]**, described on [page 125](#)
- **WTB:LXI<n>[:OUtpuT]:ENABle**, described on [page 117](#)
- **SYSTem:DESCRiption**, described on [page 73](#)
- **\*RST**, described on [page 59](#)
- **\*CLS**, described on [page 54](#)

**Syntax** `:SYSTem:PASSword:NEW <current password>, <new password>`

### Parameters

Name: current password

Data Format: <CHARACTER PROGRAM DATA>

Description: Specifies the current password that is stored in permanent memory in the instrument. This string is case-sensitive and is transmitted in clear text format.

The default password is *admin* for the DT8824 and DT8824-HV instrument modules.

**Name:** new password

**Data Format:** <CHARACTER PROGRAM DATA>

**Description:** Specifies the new password that will overwrite the current password and be saved in permanent memory in the instrument. This string is case-sensitive and is transmitted in clear text format.

**Response Data** None

**Notes** The new password becomes effective immediately. The instrument does not need to be rebooted. The new password is reflected in the LAN configuration web page for the instrument (see the user's manual for your instrument for more information on this web page).

The new password is enforced when any attempt is made to configure the instrument using SCPI commands, the instrument's web pages, or the IVI-COM driver.

Since the <current password> and <new password> are strings that are transmitted in clear text format, they can be determined easily by snooping LAN traffic.

**Example** The following example shows an attempt to change the password using a wrong password:

```
> :SYST:PASS:NEW bogus, admin1
> :SYST:ERR?
< -221, "Settings conflict;:SYST:PASS:NEW"
```

In the following example, the existing password *admin* is successfully changed to *admin1*:

```
> :SYST:PASS:NEW admin, admin1
> :SYST:ERR?
< 0, "No error"
```

**See Also** **SYSTem:PASSword[:CENable]**, described on [page 83](#)  
**SYSTem:PASSword:CDISable**, described on [page 81](#)  
**SYSTem:PASSword:CENable:STATe?**, described on [page 85](#)

## SCPI Version Query

**Description** This command, which is mandatory for SCPI-compliant devices, returns the SCPI version number to which the DT8824 instrument module complies.

**Syntax** :SYSTem:VERSion?

**Parameters** None

**Response Data** <YYYY> . <V>

**Data Format** <CHARACTER RESPONSE DATA>

Name: YYYY  
Data Format: <NR2>  
Description: The year of the version, such as 2009.

Name: V  
Data Format: <NR2>  
Description: The approved SCPI revision number for that year, such as 0.

**Notes** Refer to *1999 SCPI Command Reference*, section 21.21 for more information.

**Example** > :SYST:VERS?  
< 1999.0

In this example, the version is year 1999 and the SCPI revision is 0.

### Time Query

**Description** Returns the current time used by the DT8824 LXI instrument module. This time is updated automatically by an SNTP server.

**Syntax** :SYSTem:TIME?

**Parameters** None

**Response Data** <hour>, <minute>, <second>

Name: hour  
Data Format: <NR1>  
Description: A number from 0 to 23 representing the current hour of the instrument module.

Name: minute  
Data Format: <NR1>  
Description: A number from 0 to 59 representing the current minute of the instrument module.

Name: second  
Data Format: <NR1>  
Description: A number from 0 to 59 representing the current second of the instrument module.

**Example** > :SYST:TIME?  
< 15, 31, 45

This response indicates that the current time of the DT8824 instrument module is 3:31:45 pm (15 is the hour, 31 is the number of minutes, and 45 is the number of seconds).



## Time Zone Query

**Description** Returns the time zone that is currently used by the instrument module, as an offset from GMT (Greenwich Mean Time).

**Syntax** :SYSTem:TZONE?

**Required Parameters** None

### Response Data

Name: hour

Data Format: <NR1>

Description: A number from -12 to +12 representing the current hour relative to GMT.

Name: minute

Data Format: <NR1>

Description: A number from -59 to +59 representing the current minute relative to GMT. Minutes are rounded up to 30.

**Example**  
> :SYST:TZON  
< +5, -45

This response indicates that the current time zone of the instrument module is four hours and 30 minutes ahead of GMT. Minutes are rounded up to 30.

## Voltage Range Query

**Description** Returns the maximum full-scale voltage range that is supported by the instrument module.

**Syntax** :SYSTem:RANge[:MAX]?

**Required Parameters** None

### Response Data

Name: {BIP10V|BIP600V}

Data Format: <CHARACTER PROGRAM DATA>

Description: The maximum voltage range that is supported by the instrument module:

- **BIP10V** – Specifies an input voltage range of  $\pm 10$  V.
- **BIP600V** – Specifies an input voltage range of  $\pm 600$  V.

**Example**  
> :SYST:RANGE?  
< BIP600

This response indicates that the maximum full-scale voltage range that is supported by the instrument module is  $\pm 600$  V. Therefore, this is a DT8824-HV instrument module.

## AD Subsystem Commands

The AD subsystem is used to configure perform analog input operations on a DT8824 LXI instrument module. [Table 8](#) lists the AD subsystem commands. This section describes each of these commands in detail.

**Table 8: AD Subsystem Commands**

Type	Mnemonic	See
Abort Analog Input Operation	AD:ABORt	<a href="#">page 91</a>
Arm Analog Input Operation	AD:ARM	<a href="#">page 92</a>
ADC Synchronization Source Query	AD:SYNc:SOURce?	<a href="#">page 93</a>
Analog Input Buffer Mode Configuration	AD:BUFFer:MODE	<a href="#">page 93</a>
Analog Input Buffer Mode Query	AD:BUFFer:MODE?	<a href="#">page 94</a>
Analog Input Buffer Size Query	AD:BUFFer:SIzE[:SCANs]?	<a href="#">page 95</a>
Analog Input Channel Enable	AD:ENABle	<a href="#">page 96</a>
Analog Input Channel Enable Query	AD:ENABle?	<a href="#">page 97</a>
Analog Input Clock Source Configuration	[AD:]CLOCK:SOURce	<a href="#">page 97</a>
Analog Input Clock Source Query	[AD:]CLOCK:SOURce?	<a href="#">page 99</a>
Analog Input Gain Configuration	AD:GAIN[:CONFigure]	<a href="#">page 99</a>
Analog Input Gain Query	AD:GAIN[:CONFigure]?	<a href="#">page 100</a>
Analog Input Sampling Frequency Configuration	AD:CLOCK:FREQUency [:CONFigure]	<a href="#">page 101</a>
Analog Input Sampling Frequency Query	AD:CLOCK:FREQUency[:CONFigure]?	<a href="#">page 103</a>
Analog Input Scan Status Query	AD:STATus:SCAN?	<a href="#">page 103</a>
Analog Input Status Bits Query	AD:STATus?	<a href="#">page 104</a>
Analog Input Trigger Source Configuration	AD:TRIGger[:SOURce]	<a href="#">page 105</a>
Analog Input Trigger Source Query	AD:TRIGger[:SOURce]?	<a href="#">page 107</a>
Analog Input Trigger Edge Configuration	AD:TRIGger:EXTernal:EDGE	<a href="#">page 109</a>
Analog Input Trigger Edge Query	AD:TRIGger:EXTernal:EDGE?	<a href="#">page 109</a>
Fetch Analog Input Data	AD:FETCh?	<a href="#">page 111</a>
Initiate Analog Input Operation	AD:INITiate	<a href="#">page 114</a>

## Abort Analog Input Operation

**Description** Stops the specified analog input operation on the DT8824 LXI instrument module, if it is in progress.

**Syntax** :AD:ABORT

### Optional Keywords

Name: AD

Data Format: <CHARACTER PROGRAM DATA>

Description: Enumerated type that specifies whether to stop the operation on the analog input (AD) subsystem.

**Required Parameters** None

**Optional Parameters** None

**Response Data** None

**Notes** This is a password-protected command. To enable this command to function, use the **SYSTEM:PASSWORD[:CENable]** command, described on [page 83](#).

Use the **AD:STATus?** command, described on [page 104](#), to determine the state of the analog input subsystem.

**Example** The following example configures a DT8824 LXI instrument module to update the analog input channel at 4800 Hz when a software trigger is detected, and queries the A/D status bits:

```
> :AD:ENAB ON
> :AD:CLOC:SOURCE INT
> :AD:CLOC:FREQ MAX
> :AD:TRIG IMM
> :AD:STAT?
< 0
```

The analog input operation is then armed and triggered, and the A/D status bits are queried to determine the status of the operation:

```
> :AD:ARM
> :AD:INIT
> :AD:STAT?
< 7
```

The analog input operation is then stopped and the A/D status bits are queried to determine the status of the operation:

```
> :AD:ABOR
> :AD:STAT?
< 4
```

**See Also** **AD:ARM**, described on [page 92](#)

**AD:INITiate**, described on [page 114](#)

**AD:STATus?**, described on [page 104](#)

## Arm Analog Input Operation

**Description** Arms the analog input subsystem, which writes all the configuration settings to the instrument module.

**Syntax** :AD:ARM

### Optional Keywords

Name: AD

Data Format: <CHARACTER PROGRAM DATA>

Description: Enumerated type that specifies whether to arm the analog input (AD) subsystem.

**Required Parameters** None

**Optional Parameters** None

**Response Data** None

**Notes** This is a password-protected command. To enable this command to function, use the **SYSTEM:PASSWORD[:CENable]** command, described on [page 83](#).

Ensure that you configure the clock (as described on [page 40](#)) and the trigger (as described on [page 42](#)) before you arm the subsystem.

If you specified the trigger source for the analog input subsystem as IMMEDIATE, you must call the **AD:INITiate** command after arming the operation to start the operation.

Use the **AD:STATus?** command, described on [page 104](#), to determine the state of the analog input subsystem.

**Example** The following example configures a DT8824 LXI instrument module to sample analog input channels 1 to 3 at 4800Hz when a software trigger is detected, and queries the A/D status bits to determine the status of the operation:

```
> :AD:ENAB ON (@1:3)
> :AD:CLOC:SOUR INT
> :AD:CLOC:FREQ MAX
> :AD:TRIG IMM
> :AD:STAT?
< 0
```

The following example arms the analog input subsystem, and queries the A/D status bits to determine the status of the operation:

```
> :AD:ARM
> :AD:STAT?
< 2
```

A value of 2 indicates that the A/D subsystem was armed.

**Example (cont.)** The analog input operation is then started, and the the A/D status bits are queried to determine the status of the operation:

```
> :AD:INIT
> :AD:STAT?
< 7
```

A value of 7 indicates that the A/D subsystem is armed, triggered, and active.

**See Also** [AD:INITiate](#), described on [page 114](#)  
[STATus:OPERation:CONDition?](#), described on [page 67](#)

### ADC Synchronization Source Query

**Description** Returns the source of the ADC synchronization pulse.

**Syntax** :AD:SYNc:SOURce?

**Required Parameters** None

**Optional Parameters** None

**Response Data** {INTernal|LXI6}

Name: INTernal|LXI6

Data Format: <CHARACTER RESPONSE DATA>

Description: The source that is currently configured as the ADC synchronization pulse, where INTernal is the internal sync signal on the DT8824 instrument module and LXI6 is the synchronization signal from the Trigger Bus.

**Example** The following example returns the currently configured sync source; in this example, the internal sync source is used:

```
> :AD:SYN:SOUR?
< INTernal
```

**See Also** [AD:CLOCK:SOURce](#), described on [page 97](#)

### Analog Input Buffer Mode Configuration

**Description** Configures the wrap mode of the input buffer.

**Syntax** :AD:BUFFer:MODE {WRAP|NOWRAP|DEFault}

**Required Parameters**

Name: WRAP|NOWRAP|DEFault

Data Format: <CHARACTER PROGRAM DATA>

Description: Specifies the wrap mode for the specified buffer. The available choices are as follows:

- Description (cont.):
- WRAP mode – After the scan data has been stored in the buffer, the latest scan overwrites the oldest one in the input buffer.  
Acquisition continues until it is explicitly stopped with **AD:ABORT**, described on [page 91](#).
  - NOWRAP mode – After the scan data has been stored in the input buffer, acquisition stops automatically.
  - DEFault – The default setting is WRAP mode.

**Optional Parameters** None

**Response Data** None

**Notes** Refer to *1999 SCPI Command Reference*, section 23.6.1, for more information.

This is a password-protected command. To enable this command to function, use the **SYSTEM:PASSword[:CENable]** command, described on [page 83](#).

**Possible Errors** -284 "Program currently running"

If you receive this error, call the **AD:ABORT** command to stop the analog input operation before reconfiguring the analog input subsystem.

**Example** The following example configures the input buffer for continuous wrap mode (WRAP):

```
> :AD:BUFFER:MODE WRA
```

**See Also** **AD:BUFFER:MODE?**, described on [page 93](#)

### Analog Input Buffer Mode Query

**Description** Returns the currently configured wrap mode for the input buffer.

**Syntax** :AD:BUFFER:MODE?

**Required Parameters** None

**Optional Parameters** None

**Response Data** {WRAP|NOWRAP}

Name: WRAP|NOWRAP

Data Format: <CHARACTER RESPONSE DATA>

Description: The currently configured wrap mode. Possible values are as follows:

- WRAP – After the scan data has been stored in the buffer, the latest scan overwrites the oldest one in the input buffer and acquisition continues until it is explicitly stopped with **AD:ABORT**, described on [page 91](#). If DEFault was configured, WRAP is returned.
- NOWRAP – After the scan data has been stored in the input buffer, acquisition stops automatically.

**Notes** Refer to *1999 SCPI Command Reference*, section 23.6.1, for more information.

**Example** The following example returns the currently configured wrap mode of the input buffer; in this case, continuous wrap mode (WRAP) is used:

```
> :AD:BUFF:MODE?
< WRAP
```

**See Also** **AD:BUFFer:MODE**, described on [page 93](#)

### Analog Input Buffer Size Query

**Description** Returns the maximum number of scans that can be stored in the input buffer based on the number of enabled input channels and the input sampling rate.

**Syntax** :AD:BUFFer:SIZE[:SCANs]?

**Required Parameters** None

**Optional Parameters** None

**Response Data** <Number of scans>

Name: Number of scans

Data Format: <0+NR1>

Description: Indicates the maximum number of scans that can be stored in the input buffer based on the number of input channels and the input sampling rate.

**Notes** If the buffer wrap mode is NOWRAP, acquisition stops automatically after this number of scans is acquired.

If the buffer wrap mode is WRAP, after this number of scans is acquired, the oldest scan data is overwritten with the latest scan data as the buffer wraps.

The sampling rate determines DMA block size that is used for the scan data, where each block has a fixed number of header bytes that is not used to store scans. The DMA block size is smaller for the minimum sampling frequency, resulting in more space for the headers, and greater for the maximum sampling frequency.

Since the maximum input buffer size is approximately 8 MB and each sample is 4 bytes, you can store a maximum of approximately 2 Msamples in the input buffer. If you sample each input channel at the maximum input frequency (4800 Hz), the input buffer will fill in approximately 106 seconds (4800 samples/s per channel).

The maximum size of data transfer over Ethernet using the **AD:FETCh?** command is 32 kB. Since each sample is 4 bytes, the maximum number of samples per **AD:FETCh?** is 8 ksamples. Therefore, if you are using continuous wrap mode, ensure that you call **AD:FETCh?** in a tight loop to ensure that you retrieve all the samples in the input buffer before the buffer is overwritten.

**Example** The following example returns the number of scans that can be stored in the input buffer under various conditions:

```
> :AD:ENAB ON, (@1); :AD:BUFF:SIZE?
< 3139584
> :AD:ENAB ON, (@2); :AD:BUFF:SIZE?
< 1571328
> :AD:ENAB OFF; :AD:BUFF:SIZE
< 0
> :AD:ENAB ON; :AD:BUFF:SIZE?
< 785280
> :AD:CLOCK:FREQ MIN; :AD:BUFF:SIZE?
< 393216
> :AD:CLOCK:FREQ MAX; :AD:BUFF:SIZE?
< 785280
```

**See Also** `AD:BUFFer:MODE`, described on [page 93](#)

## Analog Input Channel Enable

**Description** Enables or disables sampling on the specified analog input channels.

**Syntax** `:AD:ENABle <state> [, <source list>]`

### Required Parameters

Name: state  
 Data Format: <Boolean>  
 Description: Specifies whether to enable or disable sampling on the specified analog input channels.  
 To enable sampling, set this value to ON or 1. To disable sampling, set this value to OFF or 0.

### Optional Parameters

Name: source list  
 Data Format: <channel\_list>  
 Description: Specifies the list of analog input channels to enable or disable. Analog input channel numbers range from 1 to 4. If *source list* is omitted, all analog input channels are configured, by default.

**Response Data** None

**Notes** This is a password-protected command. To enable this command to function, use the `SYSTEM:PASSword[:CENable]` command, described on [page 83](#).

**Possible Errors** -284 "Program currently running"

If you receive this error, call the `AD:ABORt` command to stop the analog input operation before reconfiguring the analog input subsystem.



**Example** The following example enables sampling on analog input channels 1 and 2 and disables sampling on analog input channels 3 and 4:

```
>:AD:ENAB ON, (@1, 2)
>:AD:ENAB 0, (@3, 4)
```

**See Also** `AD:ENABLE?`, described on [page 97](#)

### Analog Input Channel Enable Query

**Description** Returns whether sampling is enabled or disabled for the specified analog input channels.

**Syntax** `:AD:ENABLE? [<source list>]`

**Required Parameters** None

#### Optional Parameters

Name: source list

Data Format: <channel\_list>

Description: Specifies the list of analog input channels for which to return the enabled/disabled state. Analog input channel numbers range from 1 to 4. If *source list* is omitted, the configuration of all analog input channels is returned, by default.

**Response Data** <state>

Name: state

Data Format: <0+NR1>

Description: Contains the current state for each specified channel, separated by commas, where 1 indicates that sampling is enabled and 0 indicates that sampling is disabled.

**Example** The following example returns whether sampling is enabled or disabled for all analog input channels on the DT8824 instrument module; in this example, analog input channels 1 and 2 are enabled and analog input channels 3 and 4 are disabled.

```
>:AD:ENAB?
< 1, 1, 0, 0;
```

**See Also** `AD:ENABLE`, described on [page 96](#)

### Analog Input Clock Source Configuration

**Description** Specifies the source of the reference clock used for the ADC and DAC master clock generators on the DT8824 instrument module.

**Syntax** `:AD:CLOCK:SOURce {INTernal|LXI7}`

**Optional Keywords**

Name:	AD
Data Format:	<CHARACTER PROGRAM DATA>
Description:	Optional enumerated type that specifies the analog input (AD) subsystem for which to configure the clock source.

**Required Parameters**

Name:	INTernal   LXI7
Data Format:	<CHARACTER PROGRAM DATA>
Description:	Enumerated type that specifies the reference clock source to use, where INTernal is the internal reference clock on the DT8824 instrument module and LXI7 is line 7 of the Trigger Bus.

**Optional Parameters** None

**Response Data** None

**Notes** The specified reference clock source is used for pacing analog input operations.

If the instrument module is the clock master, specify the reference clock source as INTernal.

If the instrument module is a slave and you want to synchronize the clock with the master using the Trigger Bus, specify the reference clock source as LXI7.

This is a password-protected command. To enable this command to function, use the **SYSTEM:PASSword[:CENable]** command, described on [page 83](#).

**Possible Errors** -284 "Program currently running"

If you receive this error, call the **AD:ABORt** command to stop the analog input operation before reconfiguring the analog input subsystem.

**Example** The following example specifies the clock source on the DT8824 instrument module as the internal reference clock, which automatically configures the sync source as internal:

```
>:CLOC:SOUR INT
```

**See Also** **AD:CLOCK:SOURce?**, described on [page 99](#)  
**AD:SYNc:SOURce?**, described on [page 93](#)  
**AD:CLOCK:FREQuency[:CONFigure]**, described on [page 101](#)  
**AD:CLOCK:FREQuency[:CONFigure]?**, described on [page 103](#)  
**WTB:LXI<n>[OUTput]:ENABle**, described on [page 117](#)

## Analog Input Clock Source Query

<b>Description</b>	Returns the currently configured reference clock source used for the ADC and DAC master clock generators on the DT8824 instrument module.
<b>Syntax</b>	: [AD:] CLOCk: SOURce?
<b>Optional Keywords</b>	
Name:	AD
Data Format:	<CHARACTER PROGRAM DATA>
	Optional enumerated type that specifies the analog input (AD) subsystem for which to return the clock source.
<b>Required Parameters</b>	None
<b>Optional Parameters</b>	None
<b>Response Data</b>	<INTernal   LXI7>
Name:	INTernal   LXI7
Data Format:	<CHARACTER RESPONSE DATA>
Description:	Enumerated type that represents the reference clock source that is currently configured, where INTernal is the internal reference clock on the DT8824 and LXI7 is the external reference clock from the Trigger Bus.
<b>Example</b>	The following example returns the currently configured reference clock source on the DT8824 instrument module; in this case, the internal reference clock is configured:  <pre>&gt; :CLOC : SOUR? &lt; INTernal</pre>
<b>See Also</b>	<b>AD:CLOCK:SOURce?</b> , described on <a href="#">page 97</a> <b>AD:SYNc:SOURce?</b> , described on <a href="#">page 93</a> <b>AD:CLOCK:FREQuency[:CONFigure]</b> , described on <a href="#">page 101</a> <b>WTB:LXI&lt;n&gt;[:OUTput]:ENABle</b> , described on <a href="#">page 117</a>

## Analog Input Gain Configuration

<b>Description</b>	Configures the gain setting for the specified analog input channels.
<b>Syntax</b>	:AD:GAIN[:CONFigure] <numeric value> [, <source list>]
<b>Required Parameters</b>	
Name:	numeric value
Data Format:	<+NR1>
Description:	Specifies a gain value of 1, 8, 16, or 32.

**Optional Parameters**

Name:	source list
Data Format:	<channel_list>
Description:	Specifies the list of analog input channels to which to apply the gain value. Analog input channel numbers range from 1 to 4. If <i>source list</i> is omitted, the gain value is applied to all analog input channels, by default.

**Response Data** None

**Notes** The DT8824 provides an input range of  $\pm 10$  V and software-selectable gains of 1, 8, 16, and 32. This provides effective input ranges of  $\pm 10$  V when the gain is 1,  $\pm 1.25$  V when the gain is 8,  $\pm 0.625$  V when the gain is 16, and  $\pm 0.3125$  V when the gain is 32.

The DT8824-HV supports a bipolar range of  $\pm 600$  V and software-selectable gains of 1, 8, 16, and 32. This provides effective input ranges of  $\pm 600$  V when the gain is 1,  $\pm 75$  V when the gain is 8,  $\pm 37.5$  V when the gain is 16, and  $\pm 18.75$  V when the gain is 32.

You can determine the maximum full-scale voltage range of the instrument by using the **SYSTEM:RANGe[:MAX]?** query.

This is a password-protected command. To enable this command to function, use the **SYSTEM:PASSword[:CENable]** command, described on [page 83](#).

**Possible Errors** -284 "Program currently running"

If you receive this error, call the **AD:ABORt** command to stop the analog input operation before reconfiguring the analog input subsystem.

**Example** The following example configures analog input channels 1 and 2 for a gain of 1 and analog input channels 3 and 4 for a gain of 16:

```
:AD:GAIN 1, (@1,2)
:AD:GAIN 16, (@3,4)
```

**See Also** **AD:GAIN[:CONFigure]?**, described on [page 100](#)

**Analog Input Gain Query**

<b>Description</b>	Returns the currently configured gain value for the specified analog input channels.
<b>Syntax</b>	:AD:GAIN[:CONFigure]? [<source list>]
<b>Required Parameters</b>	None

**Optional Parameters**

Name: source list  
 Data Format: <channel\_list>  
 Description: Specifies the list of analog input channels for which to return the gain configuration. Analog input channel numbers range from 1 to 4. If *source list* is omitted, the gain configuration is returned for all analog input channels, by default.

**Response Data** <gain>

Name: gain  
 Data Format: <+NR1>  
 Description: Contains the gain value (1, 8, 16, or 32) for each specified channel, separated by commas.

**Notes** The DT8824 provides an input range of  $\pm 10$  V and software-selectable gains of 1, 8, 16, and 32. This provides effective input ranges of  $\pm 10$  V when the gain is 1,  $\pm 1.25$  V when the gain is 8,  $\pm 0.625$  V when the gain is 16, and  $\pm 0.3125$  V when the gain is 32.

The DT8824-HV supports a bipolar range of  $\pm 600$  V and software-selectable gains of 1, 8, 16, and 32. This provides effective input ranges of  $\pm 600$  V when the gain is 1,  $\pm 75$  V when the gain is 8,  $\pm 37.5$  V when the gain is 16, and  $\pm 18.75$  V when the gain is 32.

You can determine the maximum full-scale voltage range of the instrument by using the **SYSTem:RANge[:MAX]?** query.

**Example** The following example returns the gain configuration for all analog input channels on the DT8824 instrument module; in this case, channel 1 and 2 are configured for a gain of 1, and channels 3 and 4 are configured for a gain of 16:

```
> :AD:GAIN?
< 1,1,16,16
```

**See Also** **AD:GAIN[:CONFigure]**, described on [page 99](#)

**Analog Input Sampling Frequency Configuration**

**Description** Specifies the sampling frequency for pacing analog input operations on the DT8824 instrument module.

**Syntax** :AD:CLOCK:FREQUENCY[:CONFigure] <freq|MINimum|MAXimum>

**Required Parameters**

Name:	freq   MINimum   MAXimum
Data Format:	<+NRf>
Description:	Specifies the sampling frequency, in Hz, where <i>freq</i> represents a particular sampling frequency, <i>MINimum</i> represents the minimum sampling frequency, and <i>MAXimum</i> represents the maximum sampling frequency for the specified analog input subsystem.  For analog input operations, the minimum sampling frequency is 1.175 Hz and the maximum sampling frequency is 4800 Hz (the default value).

**Optional Parameters** None**Response Data** None**Notes** This is a password-protected command. To enable this command to function, use the **SYSTEM:PASSword[:CENable]** command, described on [page 83](#).

If you are synchronizing the clocks of multiple instrument modules, it is important that you enable LXI7 as an output on the clock master and then configure the clock frequency of the slaves before configuring the clock frequency of the clock master.

It is also recommended that you choose the same sampling frequency for all instrument modules to avoid problems.

If you later change the sampling frequency of one or more slave instrument modules, you must reconfigure the clock frequency of the clock master, even if its parameters have not changed.

Because of the way data is transferred from the FIFO on the DT8824, you may experience a long delay before the acquired data is available if you specify a slow sampling frequency. For example, if you want to acquire data from one channel at a sampling frequency of 1.175 Hz, you have to wait 13.6 s (to acquire 16 samples in the input FIFO) before you can access the data. The delay is minimized at faster sampling frequencies.

Refer to [page 40](#) for more information on setting up the clocks.

**Possible Errors** -131 "Invalid suffix"

If you receive this error, ensure that the sampling frequency is entered in <+NRf> format.

**-222 "Data out of range"**

If you receive this error, the sampling frequency that was entered was outside the valid range. Enter a valid sampling frequency.

**-284 "Program currently running"**

If you receive this error, call the **AD:ABORT** command to stop the analog input operation before reconfiguring the analog input subsystem.

**Example** The following example configures the maximum sampling frequency (4800 Hz) for the analog input subsystem on a DT8824:

```
>:AD:CLOC:FREQ MAX
>:AD:CLOC:FREQ?
< 4800
>:SYST:ERR?
< 0, "No error"
```

**See Also** [AD:CLOCK:SOURce](#), described on [page 97](#)  
[AD:CLOCK:SOURce?](#), described on [page 99](#)  
[AD:SYNc:SOURce?](#), described on [page 93](#)  
[AD:CLOCK:FREQuency\[:CONFigure\]?](#), described on [page 103](#)  
[WTB:LXI<n>\[:OUTput\]:ENABle](#), described on [page 117](#)

### Analog Input Sampling Frequency Query

**Description** Returns the sampling frequency that is currently configured for pacing analog input operations on the DT8824 instrument module.

**Syntax** :AD:CLOCK:FREQuency[:CONFigure]?

**Required Parameters** None

**Optional Parameters** None

**Response Data** <sampling frequency>

Name: sampling frequency

Data Format: <+NRf>

Description: The currently configured sampling frequency, in Hz, for the specified analog input subsystem.

**Example** The following example returns the currently configured sampling frequency, in Hz, for the analog input subsystem; in this case, the sampling frequency is 4800 Hz:

```
>:AD:CLOC:FREQ?
< 4800
```

**See Also** [AD:CLOCK:SOURce](#), described on [page 97](#)  
[AD:CLOCK:SOURce?](#), described on [page 99](#)  
[AD:CLOCK:FREQuency\[:CONFigure\]](#), described on [page 101](#)

### Analog Input Scan Status Query

**Description** Returns the indices of the chronologically oldest and most recent scan records in the input buffer on the instrument module.

**Syntax** :AD:STATus:SCAN?

<b>Parameters</b>	None
<b>Response Data</b>	<StartingIndex>, <EndingIndex>
Name:	StartingIndex
Data Format:	<0+NR1>
Description:	A decimal number that represents the index of the chronologically oldest scan record available in the circular buffer on the instrument module.
Name:	EndingIndex
Data Format:	<0+NR1>
Description:	A decimal number that represents the index of the most recent scan record available in the circular buffer on the instrument module.
<b>Notes</b>	If the input buffer is empty (because a scan has not been started or started and stopped), both <i>StartingIndex</i> and <i>EndingIndex</i> will be 0. Otherwise, these values will be non-zero.
<b>Example</b>	The follow example shows the <i>StartingIndex</i> (1001) and <i>EndingIndex</i> (1050) when the input buffer consists of scan records 1001 to 1050:  <pre>&gt; :AD:STAT:SCA? &lt; 1001,1050</pre>
<b>See Also</b>	<b>AD:FETCh?</b> , described on <a href="#">page 111</a>

### Analog Input Status Bits Query

<b>Description</b>	Returns the status bits for the analog input subsystem.
<b>Syntax</b>	:AD:STATus?
<b>Required Parameters</b>	None
<b>Optional Parameters</b>	None
<b>Response Data</b>	<status bits>
Name:	status bits
Data Format:	<0+NR1>
Description:	The value, in the range of 0 to 255, of the status bits for the analog input subsystem. The status bits are defined as follows: <ul style="list-style-type: none"> <li>• Bit 0 – The value of this bit is 1 if the analog input subsystem is active or 0 if the analog input subsystem is inactive.</li> <li>• Bit 1 – The value of this bit is 1 if the analog input subsystem is armed or 0 if the analog input subsystem is not armed.</li> <li>• Bit 2 – The value of this bit is 1 if the analog input subsystem is triggered or 0 if the analog input subsystem is not triggered.</li> </ul>



- Description (cont):
- Bit 3 – The value of this bit is 1 if AD SYNC was detected or 0 if AD SYNC was not detected.
  - Bit 4 – The value of this bit is 1 if the AD FIFO overflowed or 0 if the AD FIFO did not overflow.

**Notes** Refer to *IEEE 488.2-1992*, section 9.1, for more information.

**Example** The following example returns the status bits for the analog input subsystem:

```
> :AD:STAT?
< 4;
```

In this example, the status bit value is 4, indicating that the analog input subsystem was triggered.

### Analog Input Trigger Source Configuration

**Description** Specifies the trigger source that starts the analog input operation on the DT8824 instrument module.

**Syntax** :AD:TRIGger [:SOURce]  
 { IMMEDIATE | EXTERNAL | LXI0 | LXI1 | LXI2 | LXI3 | LXI4 | LXI5 |  
 LAN0 | LAN1 | LAN2 | LAN3 | LAN4 | LAN5 | LAN6 | LAN7 | DEFAULT }

#### Required Parameters

Name: IMMEDIATE | EXTERNAL | LXI0 | LXI1 | LXI2 | LXI3 | LXI4 | LXI5 | LAN0 | LAN1 |  
 LAN2 | LAN3 | LAN4 | LAN5 | LAN6 | LAN7 | DEFAULT

<CHARACTER PROGRAM DATA>

Data Format: Enumerated type that specifies the trigger source to use. The following  
 Description: trigger sources are supported:

- IMMEDIATE – A software trigger. Once you arm the subsystem with the **AD:ARM** command, the software trigger event occurs when the **AD:INITiate** command is executed (the computer issues a write to the instrument module to begin conversions).
- EXTERNAL – An external digital (TTL) trigger event. Once you arm the subsystem with the **AD:ARM** command, this trigger event occurs when the instrument module detects a rising-edge transition on the Trigger In signal connected to the instrument module.
- LXI0 – Once you arm the subsystem with the **AD:ARM** command, this trigger event occurs when a slave DT8824 instrument module detects a trigger input signal on line 0 of the Trigger Bus.
- LXI1 – Once you arm the subsystem with the **AD:ARM** command, this trigger event occurs when a slave DT8824 instrument module detects a trigger input signal on line 1 of the Trigger Bus.

- Description (cont.):
- LXI2 – Once you arm the subsystem with the **AD:ARM** command, this trigger event occurs when a slave DT8824 instrument module detects a trigger input signal on line 2 of the Trigger Bus.
  - LXI3 – Once you arm the subsystem with the **AD:ARM** command, this trigger event occurs when a slave DT8824 instrument module detects a trigger input signal on line 3 of the Trigger Bus.
  - LXI4 – Once you arm the subsystem with the **AD:ARM** command, this trigger event occurs when a slave DT8824 instrument module detects a trigger input signal on line 4 of the Trigger Bus.
  - LXI5 – Once you arm the subsystem with the **AD:ARM** command, this trigger event occurs when a slave DT8824 instrument module detects a trigger input signal on line 5 of the Trigger Bus.
  - LAN0 – Once you arm the subsystem with the **AD:ARM** command, this trigger event occurs when the DT8824 instrument module detects LXI-defined LAN event 0.
  - LAN1 – Once you arm the subsystem with the **AD:ARM** command, this trigger event occurs when the DT8824 instrument module detects LXI-defined LAN event 1.
  - LAN2 – Once you arm the subsystem with the **AD:ARM** command, this trigger event occurs when the DT8824 instrument module detects LXI-defined LAN event 2.
  - LAN3 – Once you arm the subsystem with the **AD:ARM** command, this trigger event occurs when the DT8824 instrument module detects LXI-defined LAN event 3.
  - LAN4 – Once you arm the subsystem with the **AD:ARM** command, this trigger event occurs when the DT8824 instrument module detects LXI-defined LAN event 4.
  - LAN5 – Once you arm the subsystem with the **AD:ARM** command, this trigger event occurs when the DT8824 instrument module detects LXI-defined LAN event 5.
  - LAN6 – Once you arm the subsystem with the **AD:ARM** command, this trigger event occurs when the DT8824 instrument module detects LXI-defined LAN event 6.
  - LAN7 – Once you arm the subsystem with the **AD:ARM** command, this trigger event occurs when the DT8824 instrument module detects LXI-defined LAN event 7.
  - DEFault – The default setting, which is IMMEDIATE.

**Optional Parameters**    None

**Response Data**        None

**Notes** This is a password-protected command. To enable this command to function, use the **SYSTem:PASSword[:CENable]** command, described on [page 83](#).

The order in which to set up the Trigger Bus is important; refer to [page 45](#) for more information about using the Trigger Bus to connect and trigger multiple instrument modules.

Configure the LXI-defined LAN event using the commands and queries under the **EVENT** subsystem, described on [page 120](#). Refer to [page 43](#) for more information about using LXI LAN events to trigger multiple instrument modules over the network.

**Possible Errors** -284 "Program currently running"

If you receive this error, call the **AD:ABORt** command to stop the analog input operation before reconfiguring the analog input subsystem.

**Example** The following example configures a rising-edge external trigger source to start an analog input operation on the DT8824 instrument module:

```
> :AD:TRIG EXT
> :AD:TRIG:EXT:EDGE POS
> :AD:ARM
```

**See Also** **AD:ARM**, described on [page 92](#)

**AD:INITiate**, described on [page 114](#)

**AD:TRIGger[:SOURce]?**, described on [page 107](#)

**AD:CLOCK:FREQUency[:CONFigure]**, described on [page 101](#)

**AD:CLOCK:FREQUency[:CONFigure]?**, described on [page 103](#)

## Analog Input Trigger Source Query

<b>Description</b>	Returns the trigger source that is currently configured for starting analog input operations on the DT8824 instrument module.
<b>Syntax</b>	:AD:TRIGger [ :SOURce ] ?
<b>Required Parameters</b>	None
<b>Optional Parameters</b>	None
<b>Response Data</b>	{ IMMEDIATE   EXTERNAL   LXI0   LXI1   LXI2   LXI3   LXI4   LXI5   LAN0   LAN1   LAN2   LAN3   LAN4   LAN5   LAN6   LAN7 }

Name:	IMMediate   EXTernal   LXI0   LXI1   LXI2   LXI3   LXI4   LXI5   LAN0   LAN1   LAN2   LAN3   LAN4   LAN5   LAN6   LAN7
Data Format:	<CHARACTER RESPONSE DATA>
Description:	<p>Enumerated type that specifies the trigger source to use. The following trigger sources are supported:</p> <ul style="list-style-type: none"><li>• <b>IMMediate</b> – A software trigger. Once you arm the subsystem with the <b>AD:ARM</b> command, the software trigger event occurs when the <b>AD:INITiate</b> command is executed (the computer issues a write to the instrument module to begin conversions).</li><li>• <b>EXTernal</b> – An external digital (TTL) trigger event. Once you arm the subsystem with the <b>AD:ARM</b> command, this trigger event occurs when the instrument module detects a rising-edge transition on the Trigger In signal connected to the instrument module.</li><li>• <b>LXI0</b> – Once you arm the subsystem with the <b>AD:ARM</b> command, this trigger event occurs when a slave DT8824 instrument module detects a trigger input signal on line 0 of the Trigger Bus.</li><li>• <b>LXI1</b> – Once you arm the subsystem with the <b>AD:ARM</b> command, this trigger event occurs when a slave DT8824 instrument module detects a trigger input signal on line 1 of the Trigger Bus.</li><li>• <b>LXI2</b> – Once you arm the subsystem with the <b>AD:ARM</b> command, this trigger event occurs when a slave DT8824 instrument module detects a trigger input signal on line 2 of the Trigger Bus.</li><li>• <b>LXI3</b> – Once you arm the subsystem with the <b>AD:ARM</b> command, this trigger event occurs when a slave DT8824 instrument module detects a trigger input signal on line 3 of the Trigger Bus.</li><li>• <b>LXI4</b> – Once you arm the subsystem with the <b>AD:ARM</b> command, this trigger event occurs when a slave DT8824 instrument module detects a trigger input signal on line 4 of the Trigger Bus.</li><li>• <b>LXI5</b> – Once you arm the subsystem with the <b>AD:ARM</b> command, this trigger event occurs when a slave DT8824 instrument module detects a trigger input signal on line 5 of the Trigger Bus.</li><li>• <b>LAN0</b> – Once you arm the subsystem with the <b>AD:ARM</b> command, this trigger event occurs when the DT8824 instrument module detects LXI-defined LAN event 0.</li><li>• <b>LAN1</b> – Once you arm the subsystem with the <b>AD:ARM</b> command, this trigger event occurs when the DT8824 instrument module detects LXI-defined LAN event 1.</li><li>• <b>LAN2</b> – Once you arm the subsystem with the <b>AD:ARM</b> command, this trigger event occurs when the DT8824 instrument module detects LXI-defined LAN event 2.</li></ul>

- Description (cont.):
- LAN3 – Once you arm the subsystem with the **AD:ARM** command, this trigger event occurs when the DT8824 instrument module detects LXI-defined LAN event 3.
  - LAN4 – Once you arm the subsystem with the **AD:ARM** command, this trigger event occurs when the DT8824 instrument module detects LXI-defined LAN event 4.
  - LAN5 – Once you arm the subsystem with the **AD:ARM** command, this trigger event occurs when the DT8824 instrument module detects LXI-defined LAN event 5.
  - LAN6 – Once you arm the subsystem with the **AD:ARM** command, this trigger event occurs when the DT8824 instrument module detects LXI-defined LAN event 6.
  - LAN7 – Once you arm the subsystem with the **AD:ARM** command, this trigger event occurs when the DT8824 instrument module detects LXI-defined LAN event 7.

**Example** The following example returns the currently configured trigger source for analog input operations; in this example, the external digital trigger is used:

```
> :AD:TRIG?
< EXTERNAL
```

**See Also** **AD:TRIGger[:SOURce]**, described on [page 105](#)  
**AD:ARM**, described on [page 92](#)  
**AD:INITiate**, described on [page 114](#)  
**AD:CLOCK:FREQUency[:CONFigure]**, described on [page 101](#)  
**AD:CLOCK:FREQUency[:CONFigure]?**, described on [page 103](#)

## Analog Input Trigger Edge Configuration

**Description** When an EXTERNAL trigger is selected, specifies the edge of the external trigger that starts the analog input operation on the DT8824 instrument module.

**Syntax** :AD:TRIGger:EXTERNAL:EDGE {NEGative|POSitive}

### Required Parameters

Name: NEGative | POSitive

Data Format: <CHARACTER PROGRAM DATA>

Description: Enumerated type that specifies the edge to use on the external trigger line. The following trigger edges are supported:

- NEGative – The analog input operation starts when the DT8824 detect a falling edge on the external trigger input line.
- POSitive – The analog input operation starts when the DT8824 detect a rising edge on the external trigger input line.

<b>Optional Parameters</b>	None
<b>Response Data</b>	None
<b>Notes</b>	<p>This is a password-protected command. To enable this command to function, use the <b>SYSTEM:PASSWORD[:CENable]</b> command, described on <a href="#">page 83</a>.</p> <p>Use the <b>AD:TRIGGER[:SOURCE]</b>, described on <a href="#">page 105</a>, to select an EXTERNAL trigger input.</p>
<b>Possible Errors</b>	<p>-284 "Program currently running"</p> <p>If you receive this error, call the <b>AD:ABORT</b> command to stop the analog input operation before reconfiguring the analog input subsystem.</p>
<b>Example</b>	<p>The following example configures a falling-edge external trigger source to start an analog input operation on the DT8824 instrument module:</p> <pre>&gt;:AD:TRIG EXT &gt;:AD:TRIG:EXT:EDGE NEG &gt;:AD:ARM</pre>
<b>See Also</b>	<p><b>AD:ARM</b>, described on <a href="#">page 92</a></p> <p><b>AD:INITiate</b>, described on <a href="#">page 114</a></p> <p><b>AD:TRIGGER[:SOURCE]</b>, described on <a href="#">page 105</a></p> <p><b>AD:TRIGGER[:SOURCE]?</b>, described on <a href="#">page 107</a></p> <p><b>AD:TRIGGER:EXTERNAL:EDGE?</b>, described on <a href="#">page 110</a></p>

### Analog Input Trigger Edge Query

<b>Description</b>	Returns the edge of the external trigger that is currently configured for starting analog input operations on the DT8824 instrument module.
<b>Syntax</b>	:AD:TRIGGER:EXTERNAL:EDGE?
<b>Required Parameters</b>	None
<b>Optional Parameters</b>	None
<b>Response Data</b>	{NEGative POSitive}
Name:	NEGative POSitive
Data Format:	<CHARACTER PROGRAM DATA>
Description:	<p>Enumerated type that specifies the edge to use on the external trigger line. The following trigger edges are supported:</p> <ul style="list-style-type: none"> <li>• <b>NEGative</b> – The analog input operation starts when the DT8824 detect a falling edge on the external trigger input line.</li> <li>• <b>POSitive</b> – The analog input operation starts when the DT8824 detect a rising edge on the external trigger input line.</li> </ul>

**Example** The following example returns the currently configured edge for the external trigger; in this example, a rising edge is configured:

```
>:AD:TRIG:EXT:EDGE?
< POSitive
```

**See Also** **AD:TRIGger[:SOURce]**, described on [page 105](#)  
**AD:TRIGger[:SOURce]?**, described on [page 107](#)  
**AD:TRIGger:EXTernal:EDGE**, described on [page 109](#)  
**AD:ARM**, described on [page 92](#)

## Fetch Analog Input Data

**Description** For scans that were started using the **AD:INITiate** command, described on [page 114](#), returns a number of time-stamped, sequenced measurements from the input buffer on a DT8824 LXI instrument module, in the form of scan records.

**Syntax** `:AD:FETCh? <index> [, <number of scans>]`

### Required Parameters

Name: `index`  
 Data Format: `<0+NR1>`  
 Description: Specifies the index (the offset in the input buffer) from which to retrieve scan data.

To read from the first location in the buffer, specify 0 for *index*.

To read a sequence of measurements that chronologically follow the sequence of measurements from an earlier **AD:FETCh?**, specify an *index* that is one greater than the number of the last scan from the previous **AD:FETCh?**.

If only a subset of the scans that you requested are available (either because the scan is not yet complete or the scan location in the input buffer was overwritten with a later scan, the DT8824 LXI instrument module returns the scans that exist between the starting index (*index*) and the ending index (*index+number of scans*).

For example, assume that you want to read 10 scans starting at index 5, but the scans at indices 5 through 8 have been overwritten. In this case, **AD:FETCh? 5, 10** returns only scans 9 through 15.

If no scans exist between the starting index and the ending index, an empty scan record is returned.

Description (cont.):

---

**Note:** The *Index* parameter is an unsigned, 4-byte value with a maximum value of  $2^{32} - 1$  or 4294967295. The value of *Index* starts at 0 when data acquisition starts, increments to 4294967295, then rolls over to 0 and continues incrementing.

If you scanning with a sample frequency of 4800 Hz, the value *Index* will rollover to 0 in approximately ten days. For slower sampling frequencies, this process will take longer.

---

### Optional Parameters

Name: number of scans

Data Format: <+NR1>

Description: Specifies the number of scans to retrieve from the input buffer, starting with the scan number specified by *index*.

If this parameter is either omitted, a fixed number of scans is returned. Since the size of the response in <DEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA> is limited to 2 KB, the maximum number of scan records that **AD:FETCh?** can return is limited to the integral number of scan records that can fit in the input buffer.

**Response Data** <Header><Scan record><Scan record>...<Scan record>

**Response Data Format** <DEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>

Name: Header

Description: The header contains the following 4-byte fields:

- Index – Index of the first scan record in this block.
- Number of scans – Number of scan records in this block.
- Samples per scan – Number of samples per enabled channel or scan record.
- Time stamp seconds – The time stamp of the first scan record in the block, in seconds.

Name: Scan record

Description: Consists of 4-byte "raw values" that represent the samples from the enabled input channels. If all the input channels are enabled, samples are returned in the following order:

- Chan 1 – analog channel 1
- Chan 2 – analog channel 2
- Chan 3 – analog channel 3
- Chan 4 – analog channel 4



**Notes** Since the maximum input buffer size is approximately 8 MB and each sample is 4 bytes, you can store a maximum of approximately 2 Msamples in the input buffer. If you sample each input channel at the maximum input frequency (4800 Hz), the input buffer will fill in approximately 106 seconds (4800 samples/s per channel).

The maximum size of data transfer over Ethernet using the **AD:FETCh?** command is 32 kB. Since each sample is 4 bytes, the maximum number of samples per **AD:FETCh?** is 8 ksamples. Therefore, if you are using continuous wrap mode, ensure that you call **AD:FETCh?** in a tight loop to ensure that you retrieve all the samples in the input buffer before the buffer is overwritten.

While a client can repeatedly issue a **AD:FETCh?** request, be aware that the network traffic is high and that the client has to determine if there is any chronological overlap between responses.

Because of the way data is transferred from the FIFO on the DT8824, you may experience a long delay before the acquired data is available if you specify a slow sampling frequency. For example, if you want to acquire data from one channel at a sampling frequency of 1.175 Hz, you have to wait 13.6 s (to acquire 16 samples in the input FIFO) before you can access the data. The delay is minimized at faster sampling frequencies.

When an error is encountered, an error string is appended to the error queue, and bit 2 (EAV) of the Status Byte register is set.

If bit 5 (CME) of the Standard Event Status Enable register is enabled, a command error sets bit 5 (CME) of Event Status Register.

If bit 4 (E) of the Standard Event Status Enable register is enabled, an Execution Error sets bit 4 (E) of the Standard Event Status register.

The summary of the Standard Event Status register is available in bit 5 (ESB) of the Status Byte register.

For more information on measurements, see *1999 SCPI Command Reference*, section 3.1 and 3.4. For more information on <DEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>, see *IEEE Std 488.2-1992*, section 8.7.9.

The IEEE488.2 standard defines that binary formatted data shall have a prefix that begins with #xyyyyyyyy format. The *x* is always a 1-digit ASCII numeric that specifies the "number of digits" for the *y* part; *y* has multiple digits (specified by *x*) of decimal expression that specify the number of bytes that follow.

**Example** The following example returns two scan records, starting with index 0; in this example, all the analog input channels were enabled:

```
> :AD:FETC? 0,2
< #6000052
```

Looking at the response, you can see the following:

- # is the header
- 6 (after #) specifies the number of characters that will follow
- 000052 are the six characters (after 6) that specify how many bytes that will follow
- The rest of the response includes the actual data in binary format

**See Also** **AD:INITiate**, described on [page 114](#)

**AD:ABORt**, described on [page 91](#)

## Initiate Analog Input Operation

**Description** If the configured trigger source for the analog input subsystem is IMMEDIATE, starts the analog input operation immediately once the subsystem has been armed with the **AD:ARM** command. For all other trigger sources, this command has no effect.

**Syntax** :AD:INITiate

### Optional Keywords

Name: AD

Data Format: <CHARACTER PROGRAM DATA>

Description: Enumerated type that specifies whether to start the analog input (AD) subsystem.

**Required Parameters** None

**Optional Parameters** None

**Response Data** None

**Notes** This is a password-protected command. To enable this command to function, use the **SYSTem:PASSword[:CENable]** command, described on [page 83](#).

An Execution Error is generated by this command if one of the following conditions occurs:

- The analog input operation is already running
- No input channels are enabled
- The analog input sample frequency is invalid

If bit 4 (E) of the Standard Event Status Enable register is enabled, an Execution Error sets bit 4 (E) of the Standard Event Status register.

**Notes (cont.)** To determine if an Execution Error occurred, query bit 4 of the Standard Event Status register using the **\*ESR?** command, described on [page 56](#).  
Use the **AD:STATus?** command, described on [page 104](#), to determine the state of the analog input subsystem.  
To read measurements, use the **AD:FETCh?** command, described on [page 111](#).

**Example** The following example configures a DT8824 LXI instrument module to sample analog input channels 1 to 3 at 4800 Hz when a software trigger is detected, and queries the A/D status bits to determine the status of the operation:

```
> :AD:ENAB ON (@1:3)
> :AD:CLOC:SOUR INT
> :AD:CLOC:FREQ MAX
> :AD:TRIG IMM
> :AD:STAT?
< 0
```

The analog input operation is then armed and started, and the A/D status bits are queried to determine the status of the operation:

```
> :AD:ARM
> :AD:INIT
> :AD:STAT?
< 7
```

The analog input operation is then stopped and the A/D status bits are queried to determine the status of the operation:

```
> :AD:ABOR
> :AD:STAT?
< 4
```

**See Also** **AD:TRIGger[:SOURce]**, described on [page 105](#)  
**AD:ARM**, described on [page 92](#)  
**AD:ABORt**, described on [page 91](#)  
**AD:STATus?**, described on [page 104](#)  
**AD:FETCh?**, described on [page 111](#)

## WTB (Wired Trigger Bus) Subsystem Commands

The Wired Trigger Bus subsystem includes the commands listed in [Table 9](#). Use these commands when configuring the lines of the Trigger Bus on a DT8824 LXI instrument modules. This section describes each of these commands in detail.

**Table 9: Wired Trigger Bus Subsystem Commands**

Name	Mnemonic	See
Wired Trigger Bus Lines Configuration	WTB:LXI<n>[:OUTput]:ENABLE	<a href="#">page 117</a>
Wired Trigger Bus Lines Query	WTB:LXI<n>[:OUTput]:ENABLE?	<a href="#">page 118</a>

## Trigger Bus Lines Configuration

**Description** Enables or disables a specific Trigger Bus line from being driven on the bus.

**Syntax** :WTB:LXI<n>[:OUTput]:ENABle <state>

### Required Numeric Suffix

Name: n

Description: Specifies which of the Trigger Bus lines to enable for output. Values range from 0 to 7.

### Required Parameters

Name: state

Data Format: <Boolean>

Description: Specifies whether the LXI line is driven out (enabled) on the Trigger Bus or whether the line is ignored (disabled).

To enable the Trigger Bus LXI line, set this value to ON or 1. To disable the Trigger Bus LXI line, set this value to OFF or 0.

By default, all Trigger Bus LXI lines are disabled.

**Optional Parameters** None

**Response Data** None

**Notes** This is a password-protected command. To enable this command to function, use the **SYSTem:PASSword[:CENable]** command, described on [page 83](#).

The LXI clock source is dedicated to line 7 of the Trigger Bus. You can specify the LXI clock source using the **AD:CLOCK:SOURce** command, described on [page 97](#).

The LXI sync source is dedicated to line 6 of the Trigger Bus and is automatically enabled as an output when LXI7 is enabled as an output.

The LXI trigger source can be output on lines 0 to 5 of the Trigger Bus. You can specify the source of the trigger using the **AD:TRIGger[:SOURce]** command, described on [page 105](#).

If you change the trigger source on the master, you must call the **WTB:LXI<n>[:OUTput]:ENABle** command again to ensure that the LXI trigger signal is driven out on the Trigger Bus.

Refer to [page 41](#) and [page 45](#) for more information about setting up the Trigger Bus for input and output operations.

<b>Possible Errors</b>	-114 "Header suffix out of range" If you receive this error, ensure that you specify a value of 0 to 7 for the header suffix <i>n</i> .  -284 "Program currently running" If you receive this error, call the <b>AD:ABORT</b> command to stop the operation before configuring the Trigger Bus.
<b>Example</b>	This example configures LXI line 0 of the Trigger Bus as driven (enabled): <pre>&gt;:WTB:LXI0:ENAB 1</pre>
<b>See Also</b>	<b>WTB:LXI&lt;n&gt;[:OUTput]:ENABLE?</b> , described on <a href="#">page 118</a>

### Trigger Bus Lines Query

<b>Description</b>	Returns the current configuration of a specific LXI line of the Trigger Bus.
<b>Syntax</b>	:WTB:LXI<n>[:OUTput]:ENABle?
<b>Required Numeric Suffix</b>	
Name:	n
Description:	Specifies which of the Trigger Bus lines to configure. Values range from 0 to 7.
<b>Required Parameters</b>	None
<b>Optional Parameters</b>	None
<b>Response Data</b>	<state>
Name:	state
Data Format:	<Boolean>
Description:	Returns a value of 1 if the LXI line is enabled to be driven out on the Trigger Bus, or 0 if the LXI line is disabled.
<b>Notes</b>	The LXI clock signal is dedicated to line 7 of the Trigger Bus. You can specify the LXI clock source using the <b>AD:CLOCK:SOURce</b> command, described on <a href="#">page 97</a> .  The LXI sync source is dedicated to line 6 of the Trigger Bus and is automatically enabled as an output when LXI7 is enabled as an output.  The LXI trigger signal from the analog input subsystem can be output on lines 0 to 5 of the Trigger Bus. You can specify the source of the trigger using the <b>AD:TRIGger[:SOURce]</b> command, described on <a href="#">page 105</a> .
<b>Possible Errors</b>	-114 "Header suffix out of range" If you receive this error, ensure that you specify a value of 0 to 7 for the header suffix <i>n</i> .

**Example** This example returns the configuration of LXI line 0 of the Trigger Bus; in this case LXI line 0 is driven (enabled):

```
>:WTB:LXI0:ENABle?  
< 1
```

**See Also** **WTB:LXI<n>[:OUTput]:ENABle**, described on [page 117](#)

**AD:CLOCK:SOURce**, described on [page 97](#)

**AD:TRIGger[:SOURce]**, described on [page 105](#)

## EVENT Subsystem Command

The EVENT subsystem includes the commands listed in [Table 10](#) used to manage LXI-defined LAN events.

You can configure a DT8824 instrument module to generate LAN events. Specific LAN events can be received by the DT8824 instrument module and used to trigger the analog input subsystem. There are 8 LAN event configurations under this subsystem: LAN0, LAN1, LAN2, LAN3, LAN4, LAN5, LAN6, and LAN7. In addition, there are a set of configuration parameters that apply to the entire collection of configurations. This section describes the commands and queries of the EVENT subsystem in detail.

**Table 10: EVENT Subsystem Commands**

Name	Mnemonic	See
LAN Event Domain Configuration	EVENT:DOMain	<a href="#">page 121</a>
LAN Event Domain Query	EVENT:DOMain?	<a href="#">page 121</a>
LAN Event Name Configuration	EVENT:LAN<n>:NAME	<a href="#">page 122</a>
LAN Event Name Query	EVENT:LAN<n>:NAME?	<a href="#">page 124</a>
LAN Event Transmission Enable	EVENT:LAN<n>:TRANsmit[:ENABLE]	<a href="#">page 125</a>
LAN Event Transmission Enable Query	EVENT:LAN<n>:TRANsmit[:ENABLE]?	<a href="#">page 126</a>



## LAN Event Domain Configuration

<b>Description</b>	Sets the value of the LXI domain octet that is transmitted and received in all LXI LAN trigger event packets. The domain setting is used in all the event configurations (LAN0 to LAN7).
<b>Syntax</b>	:EVENT:DOMain <value>
<b>Required Parameters</b>	
Name:	value
Data Format:	<0+NR1>
Description:	The value of the LXI domain octet that is transmitted and received in all LXI LAN event packets. Valid values range from 0 to 255; the default value is 0.
<b>Optional Parameters</b>	None
<b>Response Data</b>	None
<b>Notes</b>	<p>This is a password-protected command. To enable this command to function, use the <b>SYSTEM:PASSWORD[:CENable]</b> command, described on <a href="#">page 83</a>.</p> <p>The LAN even domain setting is saved in nonvolatile memory.</p> <p>A transmitted LAN event carries this domain setting. Any received LAN event that does not have this domain setting is ignored.</p>
<b>Possible Errors</b>	<p>-284 "Program currently running"</p> <p>If you receive this error, call the <b>AD:ABORT</b> command to stop the operation before configuring the LAN trigger event packet.</p>
<b>Example</b>	<p>This example configures the value of the LXI octet that is transmitted in all the LXI LAN event packets to 1:</p> <pre>&gt;:EVENT:DOM 1</pre>
<b>See Also</b>	<b>EVENT:DOMain?</b> , described on <a href="#">page 121</a>

## LAN Event Domain Query

<b>Description</b>	Returns the value of the LXI domain octet that is transmitted and received in all LXI LAN event packets. The domain setting is used in all the event configurations (LAN0 to LAN7).
<b>Syntax</b>	:EVENT:DOMain?
<b>Required Parameters</b>	None
<b>Optional Parameters</b>	None
<b>Response Data</b>	<value>

Name: value  
Data Format: <0+NR1>  
Description: The value of the LXI domain octet that is transmitted and received in all LXI LAN event packets. Valid values range from 0 to 255; the default value is 0.

**Notes** A transmitted LAN event carries this domain setting. Any received LAN event that does not have this domain setting is ignored.

**Example** This example returns the currently configured value of the LXI octet that transmitted in all the LXI LAN event packets; in this case, the value is 1:

```
> :EVENT:DOM?  
< 1  
> :EVENT:DOM 3  
> :EVENT:DOM?  
< 3
```

**See Also** `EVENT:DOMain`, described on [page 121](#)

### LAN Event Name Configuration

**Description** Configures the LAN event ID field in the LXI LAN packet.

**Syntax** `:EVENT:LAN<n>:NAME <event name>`

#### Required Numeric Suffix

Name: n

Description: Specifies which of the LAN events to configure. Values range from 0 to 7.

**Required Parameters** None

**Optional Parameters**

**Name:** event name

**Data Format:** <CHARACTER PROGRAM DATA>

**Description:** A string with a maximum length of 16 ASCII characters. Event names longer than 16 ASCII characters are truncated to the first 16 characters.

If this parameter is omitted, the event name is configured to be the default string that corresponds to the specified LAN event, as follows:

LAN Events	<i>event name</i>
0	LAN0
1	LAN1
2	LAN2
3	LAN3
4	LAN4
5	LAN5
6	LAN6
7	LAN7

**Response Data** None

**Notes** This is a password-protected command. To enable this command to function, use the **SYSTEM:PASSWORD[:CENable]** command, described on [page 83](#).

A LAN event that is transmitted from the instrument module has this value in the *event ID* field.

LAN events that are received by the instrument module must have this value in the *event ID* field, otherwise the event is ignored.

**Example** This example specifies the LAN event name for LAN event configuration 3 as MYLAN3:

```
> :EVENT:LAN3:NAM MYLAN3
```

**Possible Errors** -284 "Program currently running"

If you receive this error, call the **AD:ABORT** command to stop the operation before configuring the LAN trigger event packet.

-114 "Header suffix out of range"

If you receive this error, ensure that you specify a value of 0 to 7 for the header suffix *n*.

**See Also** **EVENT:LAN<n>:NAME?**, described on [page 124](#)

## LAN Event Name Query

<b>Description</b>	Returns the LAN event ID field in the LXI LAN packet.																		
<b>Syntax</b>	:EVENTt:LAN<n>:NAME?																		
<b>Required Numeric Suffix</b>																			
Name:	n																		
Description:	Specifies the LAN event for which to return the event name. Values range from 0 to 7.																		
<b>Required Parameters</b>	None																		
<b>Optional Parameters</b>	None																		
<b>Response Data</b>	<event name>																		
Name:	event name																		
Data Format:	<CHARACTER RESPONSE DATA>																		
Description:	A string with a maximum length of 16 ASCII characters. Event names longer than 16 ASCII characters are truncated to the first 16 characters.  If this parameter is omitted, the event name is configured to be the default string that corresponds to the specified LAN event, as follows:																		
	<table border="0"> <thead> <tr> <th>LAN Event</th> <th><i>event name</i></th> </tr> </thead> <tbody> <tr> <td>0</td> <td>LAN0</td> </tr> <tr> <td>1</td> <td>LAN1</td> </tr> <tr> <td>2</td> <td>LAN2</td> </tr> <tr> <td>3</td> <td>LAN3</td> </tr> <tr> <td>4</td> <td>LAN4</td> </tr> <tr> <td>5</td> <td>LAN5</td> </tr> <tr> <td>6</td> <td>LAN6</td> </tr> <tr> <td>7</td> <td>LAN7</td> </tr> </tbody> </table>	LAN Event	<i>event name</i>	0	LAN0	1	LAN1	2	LAN2	3	LAN3	4	LAN4	5	LAN5	6	LAN6	7	LAN7
LAN Event	<i>event name</i>																		
0	LAN0																		
1	LAN1																		
2	LAN2																		
3	LAN3																		
4	LAN4																		
5	LAN5																		
6	LAN6																		
7	LAN7																		
<b>Notes</b>	A LAN event that is transmitted from the instrument module has this value in the <i>event ID</i> field.  LAN events that are received by the instrument module must have this value in the <i>event ID</i> field, otherwise they the event is ignored.																		
<b>Example</b>	This example returns the currently configured event name for LAN event configuration 3; in this case, "MYLAN3" is the event name:  >:EVENTt:LAN3:NAME? < MYLAN3																		
<b>Possible Errors</b>	-114 "Header suffix out of range"  If you receive this error, ensure that you specify a value of 0 to 7 for the header suffix <i>n</i> .																		

**See Also** EVENT:LAN<n>:NAME, described on [page 122](#)

### LAN Event Transmission Enable

**Description** Enables or disables a specific LAN event for transmission from the instrument module.

**Syntax** :EVENT:LAN<n>:TRANsmit[:ENABle] <state>

#### Required Numeric Suffix

Name: n

Description: Specifies which of the LAN events to enable or disable. Values range from 0 to 7.

#### Required Parameters

Name: state

Data Format: <Boolean>

Description: Specifies whether to enable or disable the specific LAN event for transmission from the instrument module.

To enable this capability, set this value to ON or 1. To disable this capability, set this value to OFF or 0.

**Optional Parameters** None

**Response Data** None

**Notes** This is a password-protected command. To enable this command to function, use the **SYSTEM:PASSword[:CENable]** command, described on [page 83](#).

More than one LAN event configuration can be enabled for transmission. A LAN event configuration is implicitly enabled for reception using the **AD:TRIGger[:SOURce]** command, described on [page 105](#).

**Possible Errors** -114 "Header suffix out of range"

If you receive this error, ensure that you specify a value of 0 to 7 for the header suffix *n*.

-284 "Program currently running"

If you receive this error, call the **AD:ABORt** command to stop the operation before configuring the LAN trigger event packet.

**Example** This example enables LAN event configuration 3:

```
>:EVEN:LAN3:TRAN 1
```

**See Also** EVENT:LAN<n>:TRANsmit[:ENABle]?, described on [page 126](#)  
{AD|DA}:TRIGger[:SOURce], described on [page 105](#)

## LAN Event Transmission Enable Query

<b>Description</b>	Returns whether a specific LAN event is enabled or disabled for transmission from the instrument module when the analog input subsystem is triggered.
<b>Syntax</b>	:EVENT:LAN<n>:TRANsmit[:ENABle]?
<b>Required Numeric Suffix</b>	
Name:	n
Description:	Specifies the LAN event for which to return the enable/disable state. Values range from 0 to 7.
<b>Required Parameters</b>	None
<b>Optional Parameters</b>	None
<b>Response Data</b>	<state>
Name:	state
Data Format:	<Boolean>
Description:	Returns whether the specific LAN event is enabled or disabled for transmission from the instrument module when the analog input subsystem is triggered.  If this capability is enabled, the returned value is 1. If this capability is disabled, the returned value is 0.
<b>Notes</b>	More than one LAN event configuration can be enabled for transmission. A LAN event configuration is implicitly enabled for reception using the {AD}:TRIGger[:SOURce] command, described on <a href="#">page 105</a> , and is used to trigger the analog input subsystem.
<b>Example</b>	This example returns the currently configured state of LAN configuration 3; in this case, this event configuration is enabled.  <pre>&gt;:EVENT:LAN3:TRAN? &gt; 1</pre>
<b>Possible Errors</b>	-114 "Header suffix out of range"  If you receive this error, ensure that you specify a value of 0 to 7 for the header suffix <i>n</i> .
<b>See Also</b>	EVENT:LAN<n>:TRANsmit[:ENABle], described on <a href="#">page 125</a> {AD DA}:TRIGger[:SOURce], described on <a href="#">page 105</a>

## ***DOUTput Subsystem Commands***

The DOUTput subsystem includes the commands listed in [Table 11](#). Use these commands when updating the digital output port on a DT8824 LXI instrument modules. This section describes each of these commands in detail.

**Table 11: Digital OUTPut Subsystem Commands**

<b>Name</b>	<b>Mnemonic</b>	<b>See</b>
Digital Output AND Output Values	DOUTput:AND	<a href="#">page 128</a>
Digital Output OR Output Values	DOUTput:OR	<a href="#">page 129</a>
Digital Output Set State	DOUTput	<a href="#">page 130</a>
Digital Output Query State	DOUTput?	<a href="#">page 130</a>

## Digital Output AND Output Values

**Description** Compares the binary representations of a specified value and the current value of the digital output port, and performs a logical AND operation on each pair of corresponding bits.

If both bits are 1, the result is 1. If both bits are 0, the result is 0. If one bit is 1 and the other bit 0, the result is 0.

**Syntax** `:DOUTput:AND <value>`

### Required Parameters

Name: value  
 Data Format: <0+NR1>  
 Description: A value between 0 and 15 that is ANDed with the current value of the digital output port.

This value represents the weighted bit value of the digital output port, where the value of bit 0 (digital output line 0) corresponds to a decimal value of 1 ( $2^0$ ) if the bit is set, and the value of bit 3 (digital output line 3) corresponds to a decimal value of 8 ( $2^3$ ) if the bit is set.

Setting a bit closes the relay contact that corresponds to the digital output line.

**Response Data** None

**Notes** This is a password-protected command. To enable this command to function, use the `SYSTem:PASSword[:CENable]` command, described on [page 83](#).

You can use this command along with the `DOUTput` command, described on [page 130](#), and the `DOUTput:OR` command, described on [page 129](#), to change the state of certain digital output lines without affecting the state of other digital output lines.

Refer to the *1999 SCPI Data Exchange Format*, section 3.4.6.

**Example** Assume that the current value of the digital output port is decimal 8 (binary 1000). The following example ANDs this value with decimal value 1 (binary 0001):

```
> :DOUT:AND 1
> :DOUT?
< 0
```

The result is decimal value 0 (binary 0000), which opens the relay contacts associated with digital output lines 0, 1, 2, and 3.

**See Also** `DOUTput`, described on [page 130](#)  
`DOUTput:OR`, described on [page 129](#)  
`DOUTput?`, described on [page 130](#)



## Digital Output OR Output Values

**Description** Compares the binary representations of a specified value and the current value of the digital output port, and performs a logical OR operation on each pair of corresponding bits.

If both bits are 1, the result is 1. If both bits are 0, the result is 0. If one bit is 1 and the other bit 0, the result is 1.

**Syntax** :DOUTput:OR <value>

### Required Parameters

Name: value  
 Data Format: <0+NR1>  
 Description: A value between 0 and 15 that is ORed with the current value of the digital output port.

This value represents the weighted bit value of the digital output port, where bit 0 (digital output line 0) corresponds to a decimal value of 1 ( $2^0$ ) if the bit is set, and the bit 3 (digital output line 3) corresponds to a decimal value of 8 ( $2^3$ ) if the bit is set.

Setting a bit closes the relay contact that corresponds to the digital output line.

**Response Data** None

**Notes** This is a password-protected command. To enable this command to function, use the **SYSTEM:PASSWORD[:CENable]** command, described on [page 83](#).

You can use this command along with the **DOUTput** command, described on [page 130](#), and the **DOUTput:AND** command, described on [page 129](#), to change the state of certain digital output lines without affecting the state of other digital output lines.

Refer to the *1999 SCPI Data Exchange Format*, section 3.4.6.

**Example** Assume that the current value of the digital output port is decimal 8 (binary 1000). The following example ORs this value with decimal value 1 (binary 0001):

```
> :DOUT:OR 1
> :DOUT?
< 9
```

The result is decimal value 9 (binary 1001), which closes the relay contacts associated with digital output lines 0 and 3 and opens the relay contact associated with digital output lines 1 and 2.

**See Also** **DOUTput**, described on [page 130](#)  
**DOUTput:AND**, described on [page 128](#)  
**DOUTput?**, described on [page 130](#)

## Digital Output Set State

**Description** Closes or opens the relay contacts associated with the digital output port on a DT8824 LXI instrument module.

**Syntax** :DOUTput <value>

### Required Parameters

Name: value

Data Format: <0+NR1>

Description: The weighted bit value of the digital output port, where the value of bit 0 (digital output line 0) corresponds to a decimal value of 1 ( $2^0$ ) if the bit is set, and the value of bit 3 (digital output line 3) corresponds to a decimal value of 8 ( $2^3$ ) if the bit is set.

Values range from 0 to 15. Setting a bit closes the relay contact that corresponds to the digital output line.

**Response Data** None

**Notes** This is a password-protected command. To enable this command to function, use the **SYSTem:PASSword[:CENable]** command, described on [page 83](#).

Refer to the *1999 SCPI Data Exchange Format*, section 3.4.6.

**Example** The following example closes the relay contacts associated with digital output lines 0, 2, and 3 and opens the relay contact associated with digital output line 1:

```
:DOUT 13
```

**See Also** **DOUTput?**, described on [page 130](#)  
**DOUTput:AND**, described on [page 128](#)  
**DOUTput:OR**, described on [page 129](#)

## Digital OUTput Query State

**Description** Returns the current state of the relay contacts that are associated with the digital output port on a DT8824 LXI instrument module.

**Syntax** :DOUTput?

**Parameters** None

**Response Data** <value>

---

<b>Name:</b>	value
<b>Data Format:</b>	<0+NR1>
<b>Description:</b>	<p>The weighted bit value of the digital output port, where the value of bit 0 (digital output line 0) corresponds to a decimal value of 1 (<math>2^0</math>) if the bit is set, and the value of bit 3 (digital output line 3) corresponds to a decimal value of 8 (<math>2^3</math>) if the bit is set.</p> <p>Values range from 0 to 15. Setting a bit closes the relay contact that corresponds to the digital output line.</p>
<b>Notes</b>	Refer to the <i>1999 SCPI Data Exchange Format</i> , section 3.4.6.
<b>Example</b>	<pre>&gt; :DOUT? &lt; 15</pre> <p>This response indicates that all the relay contacts of the digital output port are closed.</p>
<b>See Also</b>	<p><b>DOUTput</b>, described on <a href="#">page 130</a></p> <p><b>DOUTput:AND</b>, described on <a href="#">page 128</a></p> <p><b>DOUTput:OR</b>, described on <a href="#">page 129</a></p>

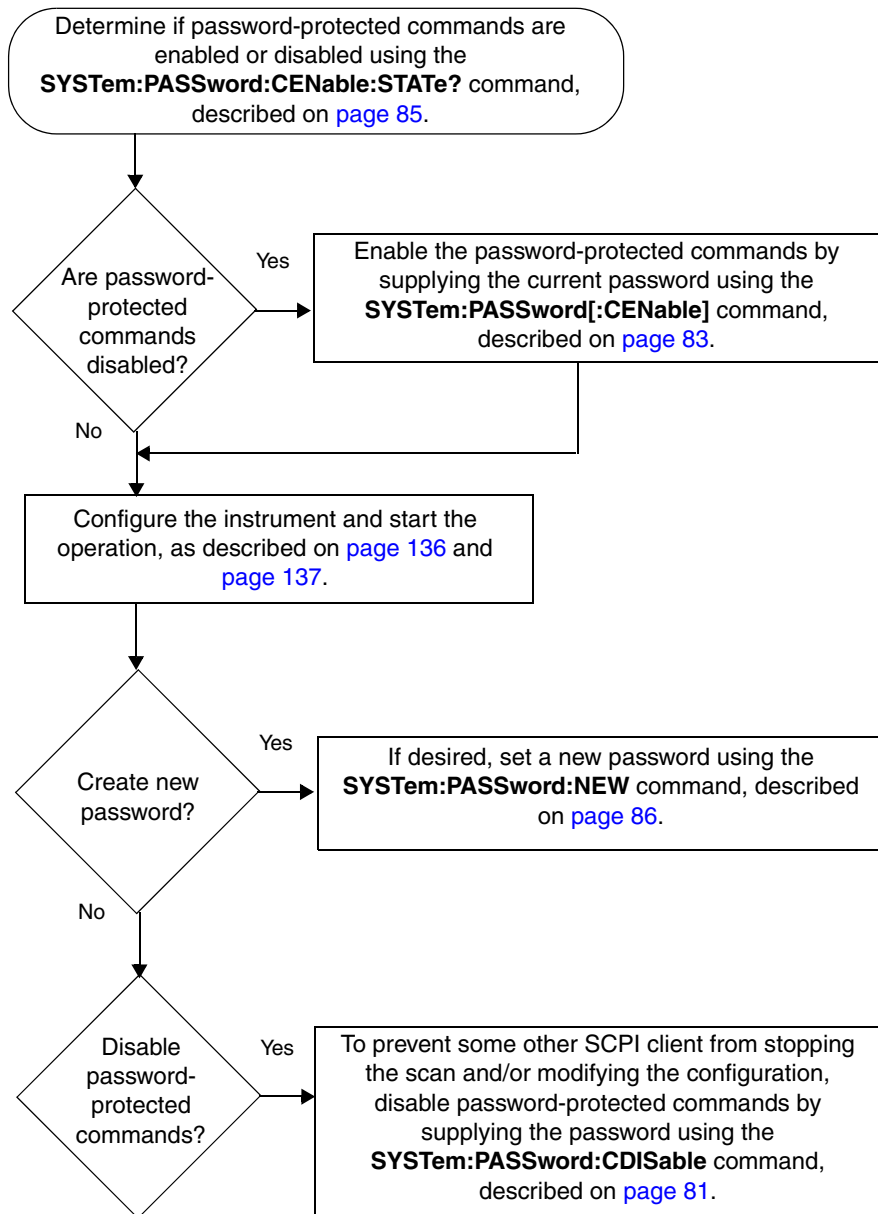




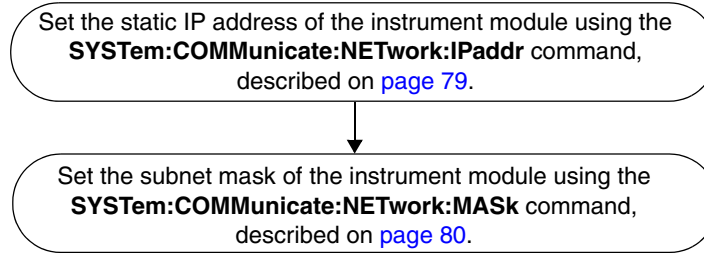
## ***Programming Flowcharts***

Using Password Protected Command .....	134
Using Password Protected Command .....	134
Performing Input Operations .....	136
Performing Digital Output Operations.....	137
Performing Digital Output Operations.....	137

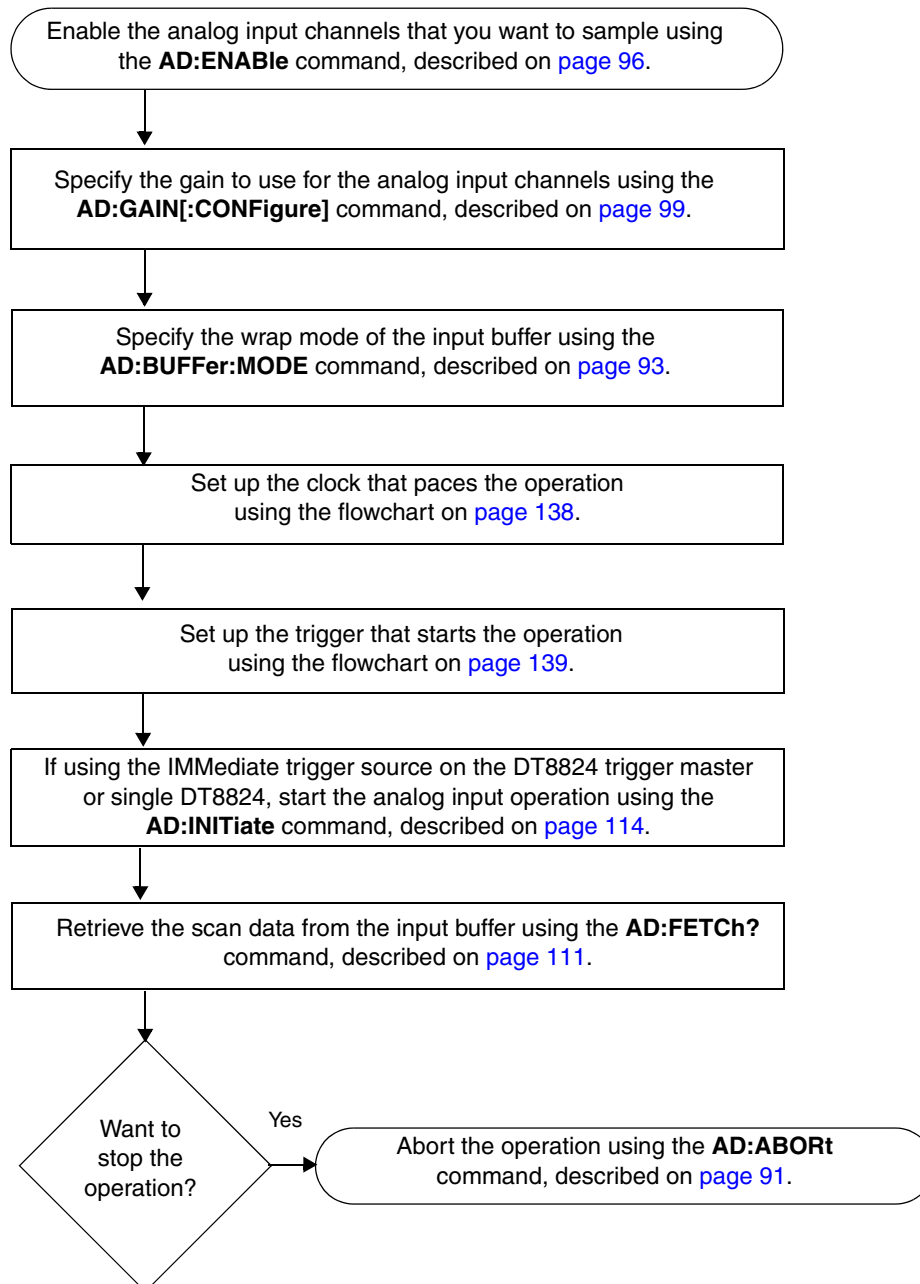
## Using Password Protected Command



## Configuring the Instrument Module on the LAN



## Performing Input Operations

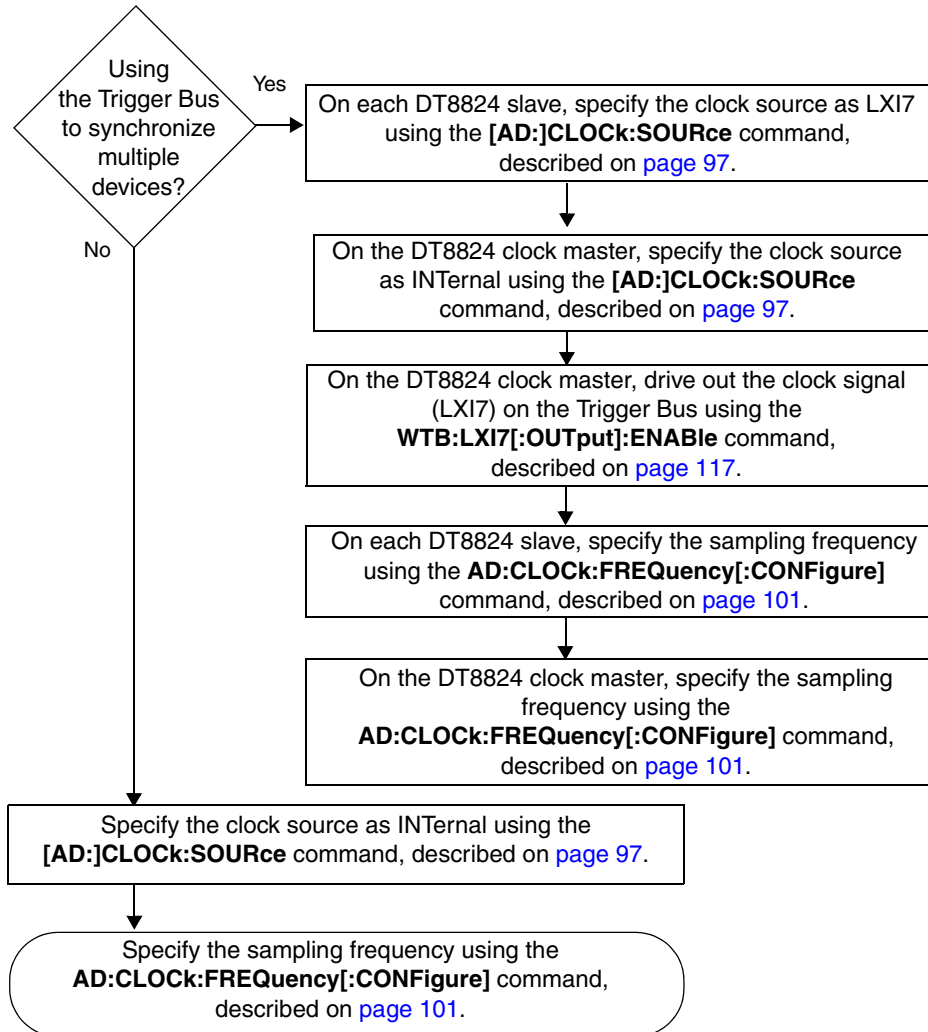




## ***Performing Digital Output Operations***

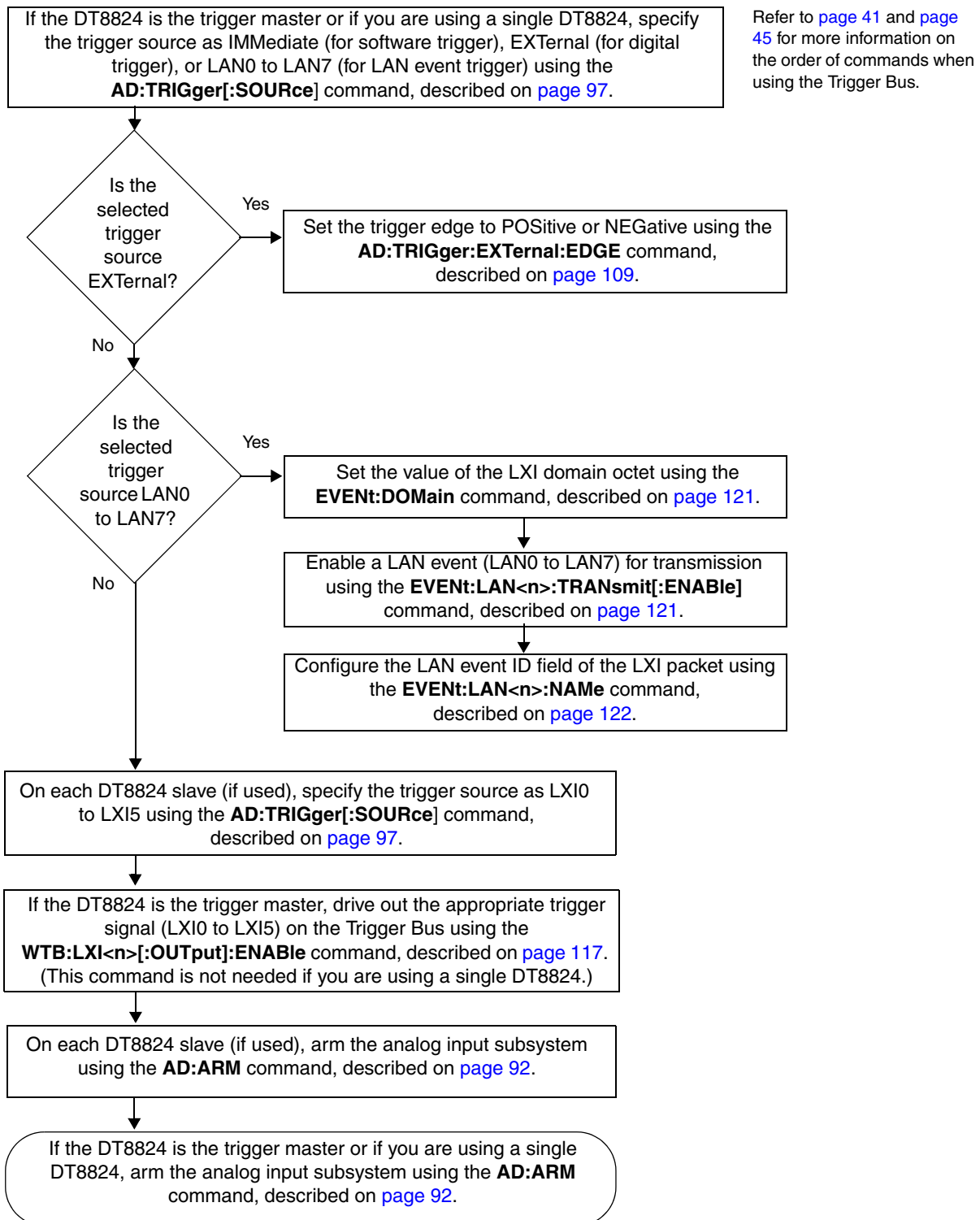
Set the state of the digital output port using the **DOUTput** command, described on [page 130](#).

## Setting up the Clock



Refer to [page 41](#) and [page 45](#) for more information on the order of commands when using the Trigger Bus.

## Setting up the Trigger







## ***Product Support***

Should you experience problems using SCPI to program a DT8824 LXI instrument module, follow these steps:

1. Read all the appropriate sections of this manual. Make sure that you have added any “Read This First” information to your manual and that you have used this information.
2. Check for a README file on the DT8824 CD. If present, read this file for the latest installation and usage information.
3. Check that you have installed your hardware devices properly. For information, refer to the documentation supplied with your devices.
4. Check that you have installed the device drivers for your hardware devices properly. For information, refer to the documentation supplied with your devices.
5. Check that you have installed your software properly.

If you are still having problems, Data Translation’s Technical Support Department is available to provide technical assistance. To request technical support, go to our web site at <http://www.mccdaq.com> and click on the Support link.

When requesting technical support, be prepared to provide the following information:

- Your product serial number
- The hardware/software product you need help on
- The version of the CD you are using
- Your contract number, if applicable

If you are located outside the USA, contact your local distributor; see our web site ([www.mccdaq.com](http://www.mccdaq.com)) for the name and telephone number of your nearest distributor.



## ***Errors***

Error Codes .....	144
Troubleshooting Errors .....	146

## Error Codes

Use the `SYSTem:ERRor:ALL?` command, described on [page 74](#), to return the errors that have occurred.

Errors are returned in your program as follows:

```
-110, "Command header error"
```

The number represents the error code and the string that follows represents the error description. Note that there is no space between the comma after error code and the quotation mark before the error description.

[Table 12](#) lists the error codes that DT8824 LXI instrument modules can return.

**Table 12: SCPI Error Codes that DT8824 LXI Instrument Modules Can Return**

Error Code	Error Message	Description
0	No error	Normal operation; no error occurred.
-100	Command error	A command is missing a parameter, or the command contains too many parameters or too many dimensions.
-102	Syntax error	An unrecognized command or data type was encountered; for example, a string was received when the instrument module does not accept strings.
-104	Data type error	A data element different than the one allowed was encountered; for example, numeric data was expected but string data was encountered.
-110	Command header error	The command header is invalid. See <a href="#">page 146</a> for information on troubleshooting this error.
-114	Header suffix out of range	The value for the header suffix is out of range.
-115	Unexpected number of parameters	The number of parameters received does not correspond to the number of parameters that were expected.
-120	Numeric data error	The value of a parameter overflowed, has the wrong dimensions, or contains an invalid value.
-131	Invalid suffix	The suffix does not follow the syntax described in IEEE 488.2, 7.7.3.2, or the suffix is inappropriate for this instrument module.
-200	Execution error	A <PROGRAM DATA> element following a header was evaluated by the instrument module as outside of its legal input range or is otherwise inconsistent with the instrument module's capabilities.  A valid program message could not be executed properly due to some condition of the instrument module.
-221	Settings conflict	The specified socket cannot be configured because it is already connected.
-222	Data out of range	A legal program data element was parsed but could not be executed because the interpreted value was outside the legal range for the instrument module.



Table 12: SCPI Error Codes that DT8824 LXI Instrument Modules Can Return (cont.)

Error Code	Error Message	Description
-284	Program currently running	Certain operations dealing with programs are illegal while the program is running; for example, you cannot configure/reconfigure an operation while a scan is in progress.
-350	Queue overflow	The error queue is full; subsequent errors can not be added to the queue. Use the <b>*CLS</b> command, described on <a href="#">page 54</a> , to clear the error queue as well as the Status Byte register
-410	Query interrupted	The query did not complete. See <a href="#">page 147</a> for information on troubleshooting this error.

## Troubleshooting Errors

This section describes how to troubleshoot the following frequently encountered errors:

- -110,"Command header error"
- -410,"Query interrupted"

### Error -110 Command Header Error

This error indicates that the command you sent was not recognized by the instrument module as a valid command name. Here are the most likely causes for receiving this error:

- A space is missing between the command and its parameter. At least one space (blank) must exist between the command and its parameter.

For example, this command will not be recognized and will return error -110:

```
:AD:BUFFer:MODEWRAP wrong
```

This command, however, is correct and will not return an error:

```
:AD:BUFFer:MODE WRAP right
```

- The command was specified with the improper short or long form.

For example, you can specify `:AD:BUFF:MODE` but not `:AD:BUF:MODE` for the `:AD:BUFFer:MODE` command.

Refer to [Chapter 3](#) starting on [page 53](#) and [Chapter 4](#) starting on [page 63](#), for the correct command name.

- A space (blank) is inserted within the command name. Spaces are not permitted within the command name.

For example, this command is incorrect and will return error -110:

```
:SYST: ERR? Wrong
```

This command, however, is correct and will not return an error:

```
:SYST:ERR? Right
```

- The instrument module did not return data. To return data from the instrument module, the client must send a valid query to the instrument module. For example, to return scan data, the client must issue the `:AD:FETCh?` command once the scan has been initiated.
- A client sent a valid query following an invalid command. This can occur when you send multiple commands or queries within one program message. When it detects an error in a program message, the instrument module discards all further commands in the program message until the end of the string.

For example, consider the following program message:

```
*IDN?;*xyz;*STB?
```

Because the `*xyz` command generates error -110, all commands on this program line fail, even though `*IDN?` and `*STB?` are valid queries.

---

## Error –410 Query Interrupted

This error applies to :INSTR connections, not to :SOCKET connections.

Usually, this error occurs when a client sends a valid query to the instrument module, and then sends another command or query to the instrument module before reading the response from the first query.

For example, the following sequence of commands will cause error –410 because the response from :SYST:ERR? is not read before the \*OPC command is sent to the instrument module:

```
: SYST:ERR?  
*OPC?
```





## ***Registers***

Status Byte Register (STB).....	150
Standard Event Status Enable Register (ESE) .....	151
Standard Event Status Register (ESR) .....	153
Operation Status Register .....	154

## Status Byte Register (STB)

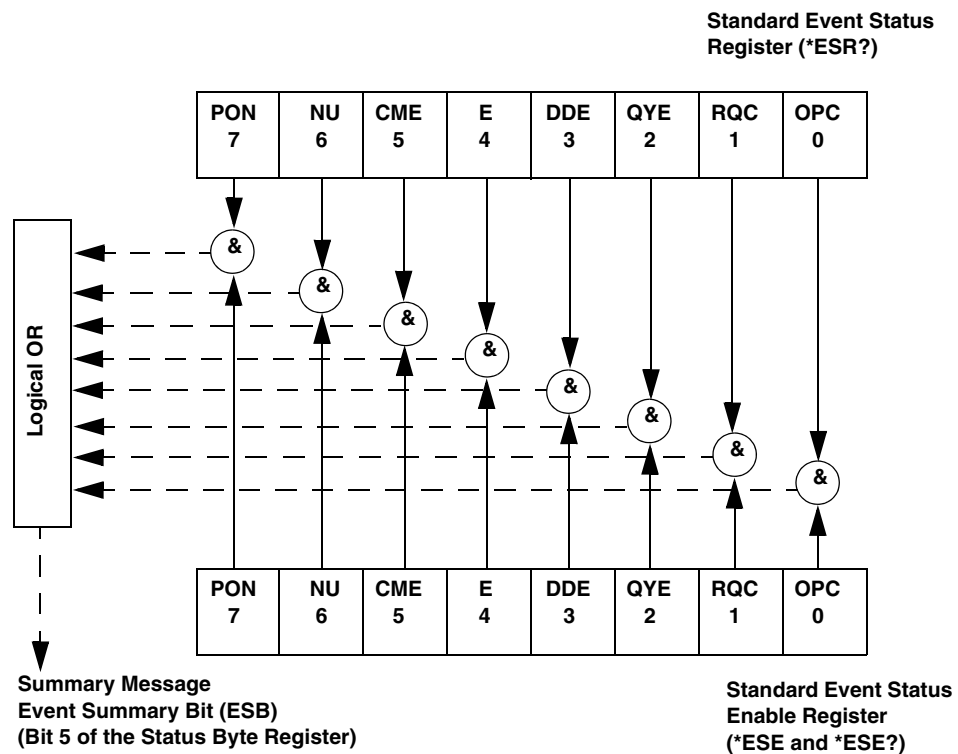
Table 13 lists the bits of the Status Byte register.

**Table 13: Status Byte Register**

Bit	Name	Description
0	Unused	The value of this bit is always 0.
1	Unused	The value of this bit is always 0.
2	Error / Event Queue Summary (EAV)	<p>During an operation, the DT8824 LXI instrument module stores error conditions as they occur in an Error / Event Queue.</p> <p>If this bit is 1, the queue is not empty. To read the error message and empty the queue, use the <b>SYST:ERR?</b> command, described on <a href="#">page 74</a>.</p> <p>If this bit is 0, the Error / Event Queue is empty.</p>
3	Questionable Register Summary	DT8824 LXI instrument modules do not implement the Questionable Data/Signal Status Register register. Therefore, the value of this bit is always 0.
4	Message Available (MAV)	<p>The Output Queue stores response messages until they are read.</p> <p>If this bit is 1, an unread message exists. The DT8824 LXI instrument module places Data Byte and END messages into the Output Queue in response to query commands. These messages are removed from the Output Queue as they are read by the controller. As long as the Output Queue contains an unread message, the value of the MAV bit is 1.</p> <p>If this bit is 0, an unread message does not exist in the Output Queue.</p>
5	Standard Event Status Bit (ESB) Summary	<p>If this bit is 1, one or more bits in the Standard Event Status register, described on <a href="#">page 153</a>, is set.</p> <p>If this bit is 0, no bits are in Standard Event Status register are set.</p>
6	Request Service	DT8824 LXI instrument modules do not implement the Request Service register. Therefore, the value of this bit is always 0.
7	Operation Status Register Summary	<p>If this bit is 1, one or more bits in the device-dependent Operation Status register, described on <a href="#">page 154</a>, is set.</p> <p>If this bit is 0, no bits in the Operation Status register are set.</p>

## Standard Event Status Enable Register (ESE)

Figure 16 shows the relationship between the Standard Event Status Enable and Standard Event Status registers.



**Figure 16: Standard Event Status Enable (ESE) and Standard Event Status Registers (ESR)**

Table 14 lists the bits of the Standard Event Status Enable (ESE) register.

**Table 14: Standard Event Status Enable (ESE) Register**

Bit	Name	Description
0	Operation Complete (OPC)	This bit is always 1 on DT8824 LXI instrument modules, since the instrument module is always enabled to complete all pending overlapped commands.  Overlapped commands are executed in parallel with subsequent commands that are sent to the DT8824 LXI instrument module.
1	Request Control (RQC)	This bit is always 0 on DT8824 LXI instrument modules since the instrument module is not configured to control GPIB operation.
2	Query Error (QYE)	If this bit is set to 1, the Query Error bit (bit 2) of the Standard Event Status register is enabled.  If this bit is set to 0, the Query Error bit (bit 2) of the Standard Event Status register is disabled.
3	Device Dependent Error (DDE)	If this bit is set to 1, the Device Dependent Error bit (bit 3) of the Standard Event Status register is enabled.  If this bit is set to 0, the Device Dependent Error bit (bit 3) of the Standard Event Status register is disabled.
4	Execution Error (E)	If this bit is set to 1, the Execution Error bit (bit 4) of the Standard Event Status register is enabled.  If this bit is set to 0, the Execution Error bit (bit 4) of the Standard Event Status register is disabled.
5	Command Error (CME)	If this bit is set to 1, the Command Error bit (bit 5) of the Standard Event Status register is enabled.  If this bit is set to 0, the Command Error bit (bit 5) of the Standard Event Status register is disabled.
6	Not Used (NU)	The value of this bit is always 0.
7	Power ON (PON)	If this bit is set to 1, the Power ON bit (bit 7) of the Standard Event Status register is enabled.  If this bit is set to 0, the Power ON bit (bit 7) of the Standard Event Status register is disabled.



## Standard Event Status Register (ESR)

Table 15 lists the bits of the Standard Event Status register; see Figure 16 on page 151 for more information on how the Standard Event Status Enable register is used with this register.

**Table 15: Standard Event Status (ESR) Register**

Bit	Name	Description
0	Operation Complete (OPC)	If this bit is set to 1, all pending operations are complete.  If this bit is set to 0, all pending operations have not been completed.
1	Request Control (RQC)	Not supported. DT8824 LXI instrument modules are not configured to control GPIB operation. Therefore, the value of this bit is always 0.
2	Query Error (QYE)	If this bit is 1, a query error was detected indicating that an attempt was made to read data from the output queue when no data was present, or that data in the output queue was lost (an overflow condition occurred).  If this bit is 0, a query error did not occur.
3	Device Dependent Error (DDE)	If this bit is 1, a device-dependent error was detected. Refer to <a href="#">Appendix A</a> starting on <a href="#">page 143</a> for a list of device-dependent errors that can be returned.  If this bit is 0, a device-dependent error did not occur.
4	Execution Error (E)	If this bit is 1, an Execution Error was detected indicating that a <PROGRAM DATA> element was outside the legal range or was inconsistent with the operation of the DT8824 LXI instrument module, or that the DT8824 LXI instrument module could not execute a valid command due to some internal condition.  If this bit is 0, an Execution Error did not occur.
5	Command Error (CME)	If this bit is 1, a Command Error was detected indicating that the DT8824 LXI instrument module received a command that did not follow proper syntax or was misspelled, or that the DT8824 LXI instrument module received a command that was not implemented.  If this bit is 0, a Command Error did not occur.
6	Not Used (NU)	The value of this bit is always 0.
7	Power ON (PON)	If this bit is 1, power to the DT8824 LXI instrument module was turned OFF and then ON since the last time the Power ON register was read.  If this bit is 0, power to the DT8824 LXI instrument module was always ON between reads of the Power ON register.

## Operation Status Register

Table 16 lists the bits of the Operation Status register. The logical OR of all bits is reported in bit 7 (Operation Status Register summary) of the Status Byte (STB) register, described on [page 150](#).

**Table 16: Operation Status Register**

Bit	Name	Description
0	Unused	The value of this bit is always 0.
1	Unused	The value of this bit is always 0.
2	Unused	The value of this bit is always 0.
3	Unused	The value of this bit is always 0.
4	A/D Measuring	If this bit is 1, the analog input subsystem is actively measuring.  If this bit is 0, the analog input subsystem is not actively measuring.
5	A/D Waiting for Trigger	If this bit is 1, the analog input subsystem is waiting for a trigger.  If this bit is 0, the analog input subsystem is not waiting for a trigger.
6	A/D Waiting for Arm	If this bit is 1, the analog input subsystem is waiting to be armed.  If this bit is 0, the analog input subsystem is not waiting to be armed.
7	Unused	The value of this bit is always 0.
8	Unused	The value of this bit is always 0.
9	Unused	The value of this bit is always 0.
10	Unused	The value of this bit is always 0.
11	Unused	The value of this bit is always 0.
12	Unused	The value of this bit is always 0.
13	Unused	The value of this bit is always 0.
14	Unused	The value of this bit is always 0.
15	Unused	The value of this bit is always 0.



## ***Examples***

Features of the DT8824 SCPI Example Program .....	157
Opening and Running the DT8824 SCPI Example Program .....	158

The DT8824 SCPI example program illustrates how to use SCPI commands to program a DT8824 instrument module.

This example was written using Visual C++ in Microsoft Visual Studio.NET 2003, 2005, and 2008, and Visual C# in Microsoft Visual Studio.NET 2005 and 2008.

---

## **Features of the DT8824 SCPI Example Program**

Using the DT8824 SCPI example program, you can do the following:

- Connect to a DT8824 instrument module by specifying an IP address.
- Configure the following parameters for the analog input subsystem:
  - Enable or disable the ability to acquire data from analog input channels 1 to 4.
  - Specify a gain of 1, 8, 16, or 32 for each analog input channel.
  - Save the analog input configuration.
- Configure the following parameters for the clock:
  - Specify the scan frequency for the analog input subsystem.
  - Save the clock configuration.
- Start acquisition and see the results of each scan.

## Opening and Running the DT8824 SCPI Example Program

To open and run the DT8824 SCPI example program, do the following:

1. Start Microsoft Visual Studio .NET.
2. Click **File**, click **Open**, and then click **Project**.
3. If you are using Visual C# .NET, from the Windows **Start** menu, select the following:  
**Programs -> Data Translation, Inc -> Instruments -> DT8824 -> SCPI Support -> Examples -> C# -> DT8824 SCPI Examples. sln**

If you are using Visual C++ .NET, from the Windows **Start** menu, select the following:  
**Programs -> Data Translation, Inc -> Instruments -> DT8824 -> SCPI Support -> Examples -> CPP -> DT8824 SCPI Examples. sln**

4. From the main menu of Microsoft Visual Studio .NET, click **Build**, and then click **Build Solution** to build the project.
5. To run the example, click **Debug** from the main menu, and then click **Start**.  
*The example program starts.*
6. Use the capabilities of the example program to see how it operates.
7. When you are finished using the example program, click **Debug** from the main menu, then click **Stop Debugging**.
8. View the user interface of the example program by clicking the appropriate [Design] tab on the main window.
9. View the source code for the example program by clicking the appropriate tab (such as example.cs) on the main window.



## ***Using HyperTerminal to Send and Receive SCPI Commands***

If you want to use the standard Windows HyperTerminal application to send and receive SCPI commands, do the following:

1. From the **Start** menu, click **Programs ->Accessories -> Communications -> HyperTerminal**.

*The HyperTerminal application opens and the following screen appears:*



2. Enter a name for the connection, such as **MyTerm**, can click **OK**.

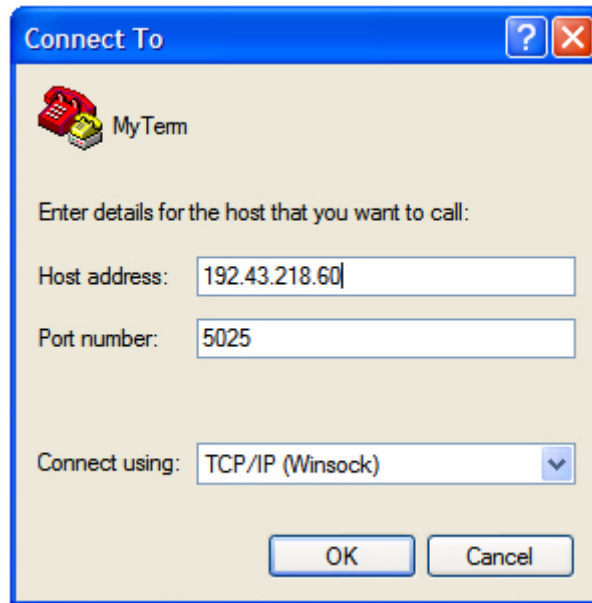
*The Connect To dialog appears:*



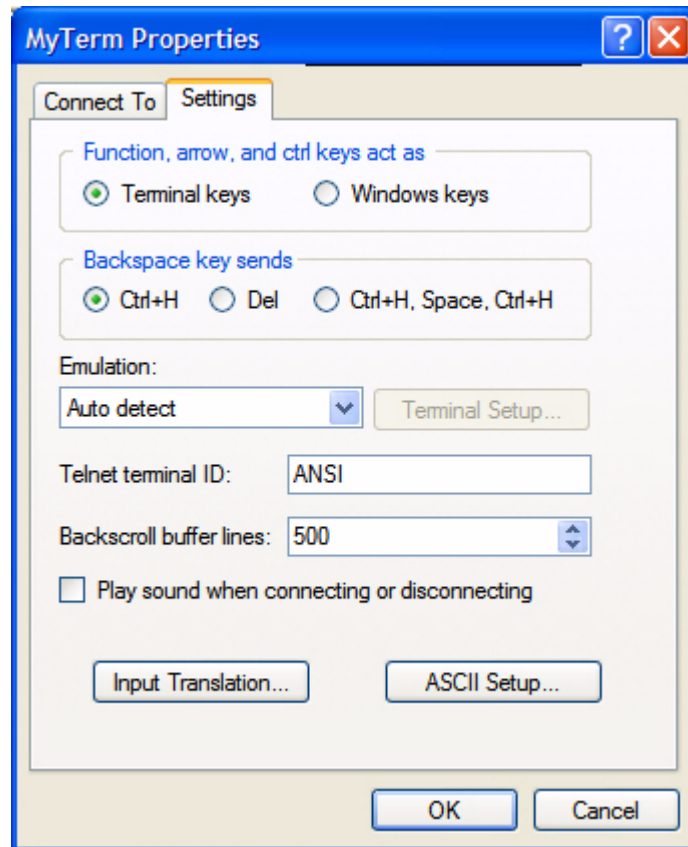
3. In the **Connect using** field, select **TCP/IP (Winsock)**. The other fields on the screen change accordingly.



4. Enter the IP address of the instrument module (which you can locate using the Eureka Discovery Utility), and then enter **5025** as the port number.  
*The screen appears as follows:*

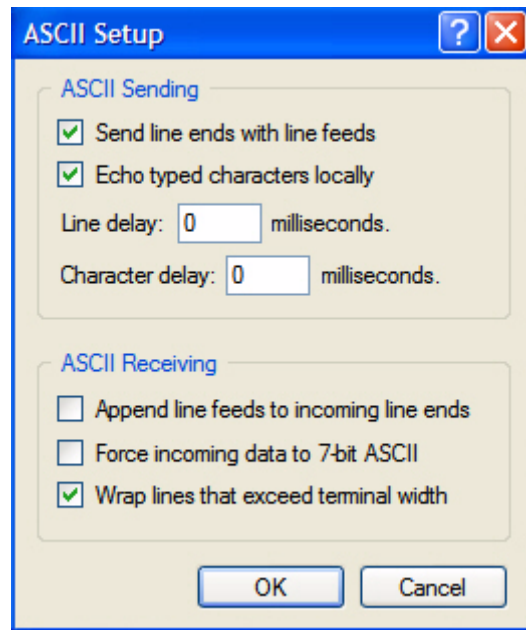


5. Click **OK**.
6. From the File menu, click **Properties**, and then click **Settings**.  
*The following screen appears:*

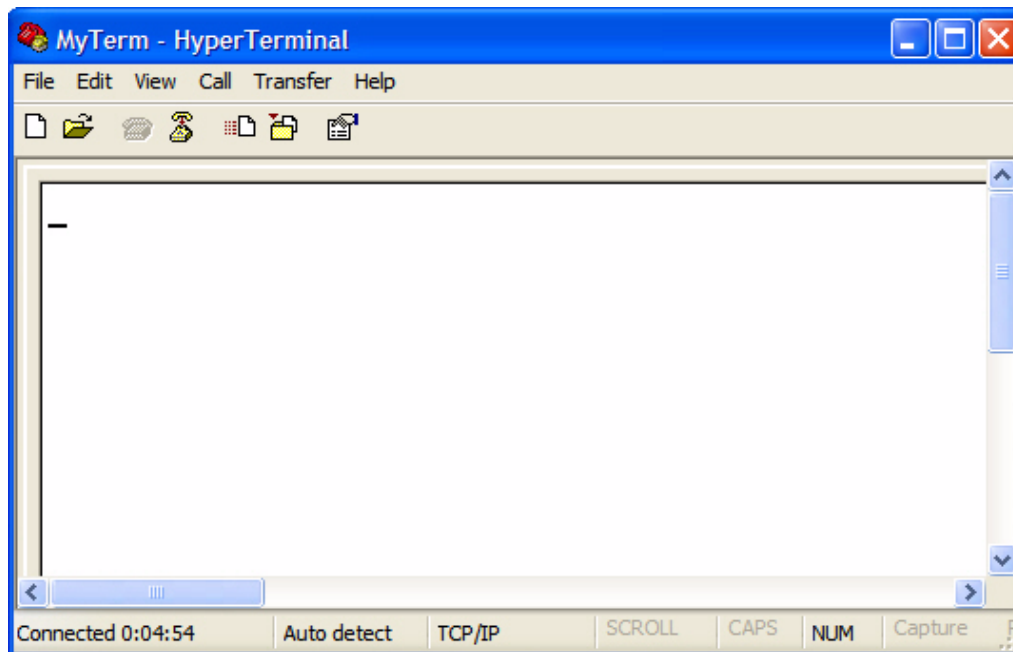


7. Click **ASCII Setup...**
8. Configure this dialog box so that the following options are checked: **Send line ends with line feeds**, **Echo typed characters locally**, and **Wrap lines that exceed terminal width**), and then click **OK**.

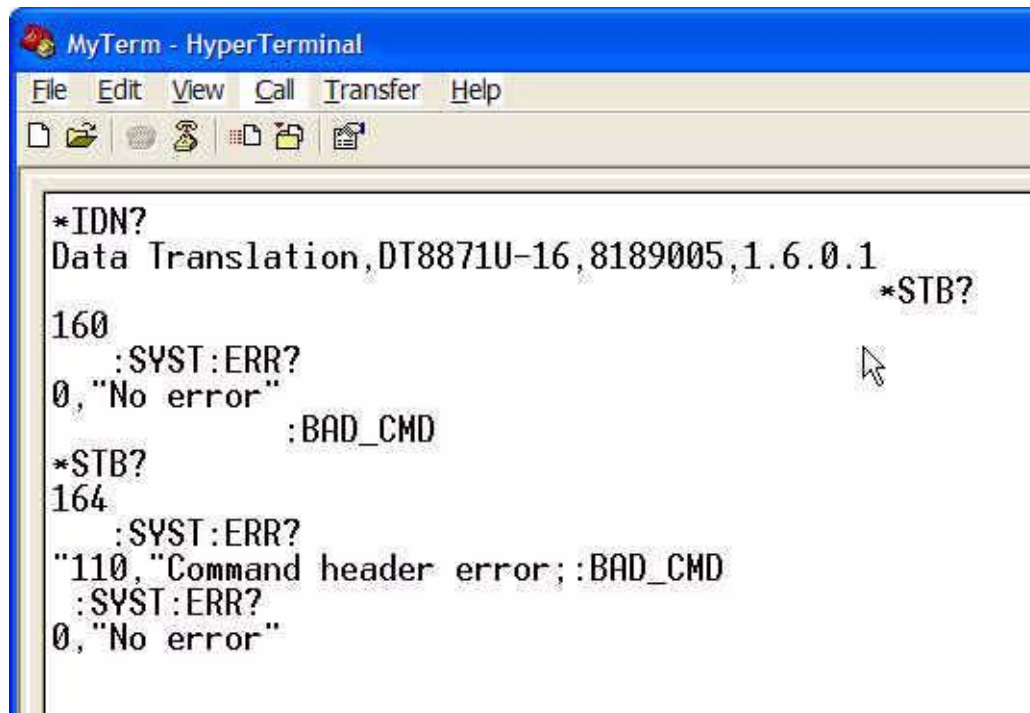
*The screen should look as follows:*



9. Click **OK** to close the Property dialog box.  
*The following screen appears:*



10. From the **Call** menu, click **Call**.
11. Enter any of the documented SCPI commands or queries. The following screen shows an example:



The screenshot shows a HyperTerminal window titled "MyTerm - HyperTerminal". The menu bar includes "File", "Edit", "View", "Call", "Transfer", and "Help". The terminal text is as follows:

```
*IDN?  
Data Translation,DT8871U-16,8189005,1.6.0.1  
*STB?  
160  
:SYST:ERR?  
0,"No error"  
:BAD_CMD  
*STB?  
164  
:SYST:ERR?  
"110,"Command header error";:BAD_CMD  
:SYST:ERR?  
0,"No error"
```

12. When you are finished using the SCPI commands, select the **Call** menu, and then click **Disconnect** to terminate your connection with the instrument module.

## Symbols

[AD:]CLOCK:SOURce 40, 41, 97, 138

[AD:]CLOCK:SOURce? 42, 99

\*CLS 54

\*ESE 54

\*ESE? 55

\*ESR? 56

\*IDN? 57

\*OPC 58

\*OPC? 58

\*RST 59

\*SRE 59

\*SRE? 60

\*STB? 60

\*TST? 59

\*WAI 62

## A

ABORt Analog Input Operation command 91

AD subsystem commands

[AD:]CLOCK:SOURce 97

[AD:]CLOCK:SOURce? 99

AD:ABORt 91

AD:ARM 92

AD:BUFFer:MODE 93

AD:BUFFer:MODE? 94

AD:BUFFer:SIZE[:SCAns]? 95

AD:CLOCK:FREQuency[:CONFigure] 101

AD:CLOCK:FREQuency[:CONFigure]? 103

AD:ENABle 96

AD:ENABle? 97

AD:FETCh? 111

AD:GAIN[:CONFigure] 99

AD:GAIN[:CONFigure]? 100

AD:INITiate 114

AD:STATus:SCAN? 103

AD:STATus? 104

AD:SYNc:SOURce? 93

AD:TRIGger:EXTernal:EDGE 109

AD:TRIGger:EXTernal:EDGE? 110

AD:TRIGger[:SOURce] 105

AD:TRIGger[:SOURce]? 107

AD:ABORt 39, 50, 91, 136

AD:ARM 47, 92, 139

AD:BUFFer:MODE 39, 93, 136

AD:BUFFer:MODE? 39, 94

AD:BUFFer:SIZE[:SCAns]? 39, 49, 95

AD:CLOCK:FREQuency[:CONFigure] 40, 41, 101, 138

AD:CLOCK:FREQuency[:CONFigure]? 42, 103

AD:ENABle 38, 96, 136

AD:ENABle? 38, 97

AD:FETCh? 48, 111, 136

AD:GAIN[:CONFigure] 38, 99, 136

AD:GAIN[:CONFigure]? 38, 100

AD:INITiate 47, 114, 136

AD:STATus:SCAN? 49

AD:STATus? 47, 104

AD:SYNc:SOURce? 42, 93

AD:TRIGger:EXTernal:EDGE 43, 46, 109, 139

AD:TRIGger:EXTernal:EDGE? 43, 110

AD:TRIGger[:SOURce] 42, 43, 44, 46, 105, 139

AD:TRIGger[:SOURce]? 42, 107

ADC sync signal 40

ADC Synchronization Source Query command 93

aliasing 42

analog input 38

arming the operation 47

configuring the buffer 39

configuring the clock 40

configuring the gain 38

configuring the input range 38

configuring the trigger 42

enabling channels 38

retrieving data from the buffer 48

starting the operation 47

stopping the operation 50

Analog Input Buffer Mode Configuration command 93

Analog Input Buffer Mode Query command 94

Analog Input Buffer Size Query command 95

Analog Input Channel Enable command 96

Analog Input Channel Enable Query command 97

Analog Input Clock Source Configuration command 97

Analog Input Clock Source Query command 99

Analog Input Gain Configuration command 99

Analog Input Gain Query command 100

Analog Input Sampling Frequency Configuration command 101

Analog Input Sampling Frequency Query command 103

Analog Input Scan Status Query 103  
 Analog Input Status Bits Query command 104  
 Analog Input Trigger Edge Configuration  
 command 109  
 Analog Input Trigger Edge Query command 110  
 Analog Input Trigger Source Configuration  
 command 105  
 Analog Input Trigger Source Query command 107  
 ARM Analog Input Operation command 92  
 arming operations 47  
 Auto-IP 33

## B

Block data types 23  
 Boolean data types 23  
 braces 18  
 brackets 18

## C

channel lists 26  
 character data types 20  
 Clear Status command 54  
 clients 48  
 clock signal 40  
 clocks  
 analog input 40  
 command header error 146  
 command hierarchy 64  
 common commands  
 \*CLS 54  
 \*ESE 54  
 \*ESE? 55  
 \*ESR? 56  
 \*IDN? 57  
 \*OPC 58  
 \*OPC? 58  
 \*RST 59  
 \*SRE 59  
 \*SRE? 60  
 \*STB? 60  
 \*TST? 59  
 \*WAI 62  
 configuring LAN settings of an instrument 135  
 configuring LAN settings of an instrument module  
 33  
 continuous wrap mode 39

## D

Data Interchange Format (DIF) expressions 28  
 data types 20  
 Block 23  
 Boolean 23  
 character 20  
 NR1 21  
 NR2 21  
 NRf 22  
 NRr 21  
 string 20  
 DATE Query command 73, 74  
 description of the device 34  
 device description 34  
 DHCP 33  
 digital output 51  
 Digital Output AND Output Values command 128  
 Digital Output OR Output Values command 129  
 Digital OUTput Query State command 130  
 Digital Output Set State command 130  
 digital trigger 43  
 Disable Password-Protected Commands command  
 81  
 documentation 32  
 domain 44  
 DOUTput 51, 130, 137  
 DOUTput subsystem commands  
 DOUTput 130  
 DOUTput:AND 128  
 DOUTput:OR 129  
 DOUTput? 130  
 DOUTput:AND 51, 128  
 DOUTput:OR 51, 129  
 DOUTput? 51, 130  
 DT8851 SCPI example 156

## E

Enable Password-Protected Commands command  
 83  
 enabling channels  
 analog input 38  
 Error Query All Codes command 75  
 Error Query All command 74  
 ERRor Query Count command 76  
 Error Query Next Code command 78  
 Error Query Next command 77  
 errors 143, 144, 149, 150, 152, 153, 154, 155, 159  
 -110 146  
 -410 147

error codes [144](#)  
   troubleshooting [146](#)  
 event ID [44](#)  
 event name [44](#)  
 Event Status (ESR) register [153](#)  
 Event Status Enable (ESE) register [151](#)  
 Event Status Register Query command [56](#)  
 EVENT subsystem commands  
   EVENT:DOMain [121](#)  
   EVENT:DOMain? [121](#)  
   EVENT:LAN<n>:TRANsmit[:ENABLE]? [126](#)  
   EVENT:LAN<n>:NAME [122](#)  
   EVENT:LAN<n>:NAME? [124](#)  
   EVENT:LAN<n>:TRANsmit[:ENABLE] [125](#)  
 EVENT:DOMain [44](#), [46](#), [121](#), [139](#)  
 EVENT:DOMain? [121](#)  
 EVENT:LAN[<n>]:TRANsmit[:ENABLE]? [126](#)  
 EVENT:LAN<n>:NAME [44](#), [45](#), [46](#), [122](#), [139](#)  
 EVENT:LAN<n>:NAME? [124](#)  
 EVENT:LAN<n>:TRANsmit[:ENABLE] [44](#), [46](#), [125](#),  
   [139](#)  
 examples [32](#), [156](#)  
 expression types [25](#)  
   channel lists [26](#)  
   Data Interchange Format (DIF) [28](#)  
   instrument-specifier [29](#)  
   numeric [25](#)  
   numeric lists [27](#)  
 EXTERNAL digital trigger [43](#)

## F

FIFO, input [39](#), [48](#)  
 flowcharts  
   analog input [136](#)  
   clock for analog input [138](#)  
   digital output [137](#)  
   trigger for analog input [139](#)  
   using password-protected commands [134](#)  
 frequency  
   analog input operations [41](#)  
 frequency, analog input operations [40](#), [41](#)

## G

gain [38](#)

## H

help [8](#)  
 hierarchy of commands [64](#)

HyperTerminal [159](#), [160](#)

## I

Identification query [57](#)  
 IMMEDIATE software trigger [42](#)  
 INITiate Analog Input Operation command [114](#)  
 initiating input operations [47](#)  
 input buffer  
   configuring [39](#)  
   retrieving data [48](#)  
 input ranges [38](#)  
 Input Trigger LED [42](#)  
 installation [32](#)  
 instrument-specifier expressions [29](#)  
 internal clock [40](#)  
 internet protocols, UDP [43](#)  
 IP address [33](#), [34](#)

## L

LAN configuration [33](#)  
   flowchart [135](#)  
   returning information about the instrument [34](#)  
 LAN Event Domain Configuration command [121](#)  
 LAN Event Domain Query command [121](#)  
 LAN Event Name Configuration command [122](#)  
 LAN Event Name Query command [124](#)  
 LAN Event Transmission Enable command [125](#)  
 LAN Event Transmission Enable Query command  
   [126](#)  
 LAN Static IP Address Configuration command [79](#)  
 LAN Static IP Address Query command [80](#)  
 LAN Subnet Mask Configuration command [80](#)  
 LAN Subnet Mask Query command [81](#)  
 LAN trigger packet [43](#)  
 LED, Input Trigger [42](#)  
 LXI6 ADC Sync signal [40](#)  
 LXI7 clock signal [40](#)

## M

mask, subnet [33](#), [34](#)  
 master clock [40](#)  
 messages  
   program [16](#)  
   response [19](#)  
 mnemonics [18](#)  
 multicast [43](#)

**N**

no wrap mode 39  
 NR1 data types 21  
 NR2 data types 21  
 NRf data types 22  
 NRr data types 21  
 numeric expression types 25  
 numeric lists 27  
 Nyquist Theorem 42

**O**

Operation Complete command 58  
 Operation Complete Query command 58  
 Operation Condition (OCR) register 154  
 Operation Condition Register Query command 67  
 Operation Event Register Enable command 67  
 Operation Event Register Query command 68  
 Operation Status Register Enable Query command 68

**P**

password-protected commands 35  
 port 5025 10  
 Presetting registers command 69  
 program messages 16  
 protocols, UDP 43

**Q**

query interrupted error 147  
 Query Password Enable State command 85  
 Questionable Condition Register Query command 69  
 Questionable Enable Register command 70  
 Questionable Event Register Query command 70  
 Questionable Register Enable Query command 70

**R**

ranges, analog input 38  
 Read Status Byte query 60  
 reference clock 40  
 registers  
   Operation Condition (OCR) 154  
   Standard Event Status (ESR) 153  
   Standard Event Status Enable (ESE) 151  
   Status Byte (STB) 150  
 related documents 8  
 Reset command 59

response messages 19  
 retrieving input data 48

**S**

sample clock 40  
 SCPI clients 48  
 SCPI data types 20  
 SCPI expression types 25  
   channel lists 26  
   Data Interchange Format (DIF) expressions 28  
   instrument-specifier expressions 29  
   numeric expressions 25  
   numeric lists 27  
 SCPI support files 32  
 SCPI Version Query command 87  
 Self-Test Query command 59  
 service and support procedure 142  
 Service Request Enable command 59  
 Service Request Enable Query command 60  
 Set New Password command 86  
 sockets 10  
 software trigger 42  
 Standard Event Status (ESR) register 153  
 Standard Event Status Enable (ESE) register 151  
 Standard Event Status Enable Register command 54  
 Standard Event Status Enable Register Query command 55  
 starting operations 47  
 static IP 33  
 Status Byte (STB) register 150  
 STATUS subsystem commands  
   STATUS:OPERation:CONDition? 67  
   STATUS:OPERation:ENABle 67  
   STATUS:OPERation:ENABle? 68  
   STATUS:OPERation:EVENT? 68  
   STATUS:PRESet 69  
   STATUS:QUEStionable:CONDition? 69  
   STATUS:QUEStionable:ENABle 70  
   STATUS:QUEStionable:ENABle? 70  
   STATUS:QUEStionable:EVENT? 70  
   SYSTem:VERSIon? 87  
 STATUS:OPERation:CONDition? 67  
 STATUS:OPERation:ENABle 67  
 STATUS:OPERation:ENABle? 68  
 STATUS:OPERation:EVENT? 68  
 STATUS:PRESet 69  
 STATUS:QUEStionable:CONDition? 69  
 STATUS:QUEStionable:ENABle 70  
 STATUS:QUEStionable:ENABle? 70



STATus:QUESTionable:EVENT? 70  
 stopping operations 50  
 string data types 20  
 subnet mask 33, 34  
 syntax conventions 9
 

- braces 18
- brackets 18
- common SCPI commands 17
- program messages 16
- response messages 19
- SCPI data types 20
- SCPI expression types 25
- SCPI subsystem commands 17
- short- and long-form mnemonics 18
- vertical bars 18

 SYSTEM subsystem commands
 

- SYSTEM:COMMunicate:NETwork:IPAddr 79
- SYSTEM:COMMunicate:NETwork:IPAddr? 80
- SYSTEM:COMMunicate:NETwork:MASk 80
- SYSTEM:COMMunicate:NETwork:MASk? 81
- SYSTEM:DATE? 73, 74
- SYSTEM:ERRor:ALL? 74
- SYSTEM:ERRor:CODE:ALL? 75
- SYSTEM:ERRor:CODE[:NEXT]? 78
- SYSTEM:ERRor:COUNt? 76
- SYSTEM:ERRor[:NEXT]? 77
- SYSTEM:PASSword:CDISable 81, 134
- SYSTEM:PASSword:CENable:STATE? 85, 134
- SYSTEM:PASSword:NEW 86, 134
- SYSTEM:PASSword[:CENable] 83, 134
- SYSTEM:RANGE[:MAX]? 89
- SYSTEM:TIME? 88
- SYSTEM:TZONE? 89

 SYSTEM:COMMunicate:NETwork:IPAddr 33, 79, 135  
 SYSTEM:COMMunicate:NETwork:IPAddr? 34, 80  
 SYSTEM:COMMunicate:NETwork:MASk 33, 80, 135  
 SYSTEM:COMMunicate:NETwork:MASk? 34, 81  
 SYSTEM:DATE? 73, 74  
 SYSTEM:DESCription 33  
 SYSTEM:DESCription? 34  
 SYSTEM:ERRor:ALL? 74  
 SYSTEM:ERRor:CODE:ALL? 75  
 SYSTEM:ERRor:CODE[:NEXT]? 78  
 SYSTEM:ERRor:COUNt? 76  
 SYSTEM:ERRor[:NEXT]? 77  
 SYSTEM:PASSword:CDISable 35, 81  
 SYSTEM:PASSword:CENable:STATE? 36, 85  
 SYSTEM:PASSword:NEW 36, 86  
 SYSTEM:PASSword[:CENable] 35, 83

SYSTEM:RANge[:MAX]? 38, 100, 101  
 SYSTEM:TIME? 88  
 SYSTEM:VERSion? 87

## T

TCP port 5025 10  
 technical support 8, 142  
 TIME Query command 88  
 time stamp 44  
 Time Zone Query 89  
 Trigger Bus
 

- clock signal 40
- trigger signal 45

 triggers
 

- analog input 42
- EXTernal digital 43
- IMMediate software 42
- LAN0 to LAN7 trigger packet 43
- Trigger Bus 45

 troubleshooting checklist 142  
 troubleshooting errors 146  
 troubleshooting procedure 142  
 TTL trigger 43

## U

UDP 43  
 using HyperTerminal 159

## V

vertical bars 18  
 VISA 10  
 Voltage Range Query 89  
 voltage ranges 38  
 VXI-11 clients 48

## W

Wait-to-Continue command 62  
 wrap mode 39  
 WTB subsystem commands
 

- WTB:LXI<n>[:OUTput]:ENABLE 117
- WTB:LXI<n>[:OUTput]:ENABLE? 118

 WTB:LXI<n>[:OUTput]:ENABLE 41, 46, 47, 117, 138, 139  
 WTB:LXI<n>[:OUTput]:ENABLE? 118

