Tektronix®

AWG4162
Arbitrary Waveform Generator

Basic Programmer Manual

077-1198-00

Tektronix®

AWG4162 Arbitrary Waveform Generator

Basic Programmer Manual

Register now!
Click the following link to protect your product.
➔ www.tek.com/register

www.tek.com
077-1198-00

Contacting Tektronix, Inc.

Tektronix, Inc.
14150 SW Karl Braun Drive
P.O. Box 500
Beaverton, OR 97077
USA

For product information, sales, service, and technical support:

- In North America, call 1-800-833-9200.
- Worldwide, visit www.tek.com to find contacts in your area.

Warranty

Tektronix warrants that the product will be free from defects in materials and workmanship for a period of three (3) years from the date of original purchase from an authorized Tektronix distributor. If the product proves defective during this warranty period, Tektronix, at its option, either will repair the defective product without charge for parts and labor, or will provide a replacement in exchange for the defective product. Batteries are excluded from this warranty. Parts, modules and replacement products used by Tektronix for warranty work may be new or reconditioned to like new performance. All replaced parts, modules and products become the property of Tektronix.

In order to obtain service under this warranty, Customer must notify Tektronix of the defect before the expiration of the warranty period and make suitable arrangements for the performance of service. Customer shall be responsible for packaging and shipping the defective product to the service center designated by Tektronix, shipping charges prepaid, and with a copy of customer proof of purchase. Tektronix shall pay for the return of the product to Customer if the shipment is to a location within the country in which the Tektronix service center is located. Customer shall be responsible for paying all shipping charges, duties, taxes, and any other charges for products returned to any other locations.

This warranty shall not apply to any defect, failure or damage caused by improper use or improper or inadequate maintenance and care. Tektronix shall not be obligated to furnish service under this warranty a) to repair damage resulting from attempts by personnel other than Tektronix representatives to install, repair or service the product; b) to repair damage resulting from improper use or connection to incompatible equipment; c) to repair any damage or malfunction caused by the use of non-Tektronix supplies; or d) to service a product that has been modified or integrated with other products when the effect of such modification or integration increases the time or difficulty of servicing the product.

THIS WARRANTY IS GIVEN BY TEKTRONIX WITH RESPECT TO THE PRODUCT IN LIEU OF ANY OTHER WARRANTIES, EXPRESS OR IMPLIED. TEKTRONIX AND ITS VENDORS DISCLAIM ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

TEKTRONIX' RESPONSIBILITY TO REPAIR OR REPLACE DEFECTIVE PRODUCTS IS THE SOLE AND EXCLUSIVE REMEDY PROVIDED TO THE CUSTOMER FOR BREACH OF THIS WARRANTY. TEKTRONIX AND ITS VENDORS WILL NOT BE LIABLE FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES IRRESPECTIVE OF WHETHER TEKTRONIX OR THE VENDOR HAS ADVANCE NOTICE OF THE POSSIBILITY OF SUCH DAMAGES.

[W16 – 15AUG04]

Table of Contents

Basic Command Descriptions 3-1

Preface

This manual provides operating information for the following products:

AWG4162

The manual consists of the following sections:

Getting Started (on page 1-1) describes operating principles of the instrument, which helps you understand how your generator operates.

Syntax and Commands (on page 2-1) defines the command syntax and processing conventions and describes command notation.

Basic Command Descriptions (on page 3-1) presents detailed information about using each of the commands, including syntax, arguments, and examples.

Status and Events (on page 4-1) explains the status information and event messages reported by the instrument.

Programming Examples (on page 5-1) contains remote interface application programs to help you develop programs for your application.

SCPI Conformance Information (on page A-1) contains a list of commands and SCPI information.

Default Settings (on page B-1) contains a list of settings the instrument is returned to after a reset.

Documentation

The following table lists related documentation available for your AWG4162. The documentation is available on the document CD-ROM and on the Tektronix Web site (ttp://www.tek.com/manuals).

Item	Purpose	Location
Compliance and Safety Instructions	Compliance, safety, and basic installation information	Printed and shipped with your instrument and Tektronix Web site (www.tek.com/manuals)
Basic Application Help	Basic application operating information	Instrument and PDF on the Tektronix Web site (www.tek.com/manuals)
Advanced Application Help	Advanced application operating information	Instrument and PDF on the Tektronix Web site (www.tek.com/manuals)
Basic Programmer Manual	Basic programming information	PDF on Tektronix Web site (www.tek.com/manuals)
Advanced Programmer Manual	Advanced programming information	PDF on Tektronix Web site (www.tek.com/manuals)
Service Manual	Instrument servicing procedures and replaceable parts list	PDF on Tektronix Web site (www.tek.com/manuals)

Item	Purpose	Location
Technical Reference	Instrument specifications and performance verification procedures	PDF on Tektronix Web site (www.tek.com/manuals)
Declassification and Security Instructions	Describes how to sanitize security space in the Tektronix AWG4000 Series arbitrary waveform generator hard disk	PDF on Tektronix Web site (www.tek.com/manuals)

General Features

The AWG4162 has two working modes:

- Basic (DDS) mode
  - Two analog channels
  - 600 MHz sine waveforms
  - 2.5 GS/s, 14-bit, 16 kpts arbitrary waveforms
  - Amplitude up to 5 Vp-p into 50 Ω load
- Advanced (Arbitrary) mode
  - Two analog channels
  - 16/32-bit digital channels (optional)
  - 1/16/32/64 Mpts per channel arbitrary waveform memory (optional)
  - Up to 750 MHz bandwidth
  - SFDR < -60 dBc

This manual describes how to use the AWG4162 in basic mode.

Getting Started

To help you get started with programming the AWG4162, this section includes the following subsections:

- Overview of the Manual (on page 1-1)
  Summarizes each major section of this manual.
- Connecting the Interface (on page 1-2)
  Describes how to physically connect the arbitrary waveform generator to a controller.
- Using TekVISA (on page 1-6)
  Describes how to use the TekVISA communication protocol.

Overview of the Manual

The information contained in each major section of this manual is described below.

Syntax and Commands

The Syntax and Commands (on page 2-1) section describes the structure and content of the messages your program sends to the arbitrary waveform generator. The following figure shows command parts as described in the Command Syntax (on page 1) subsection.

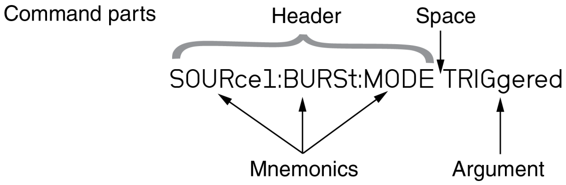


Figure 1: Command parts

The Command Groups (on page 2-11) subsection lists the commands by functional category.

Command Descriptions

The Command Descriptions (on page 3-1) section lists the commands alphabetically. It describes the details of each command and provides examples of how you might use it.

Status and Events

The program may request information from the instrument. The instrument provides information in the form of status and error messages. The following figure illustrates the basic operation of this system. The Status and Events (on page 4-1) section describes how to get status or event information from the program and lists the event and error messages.

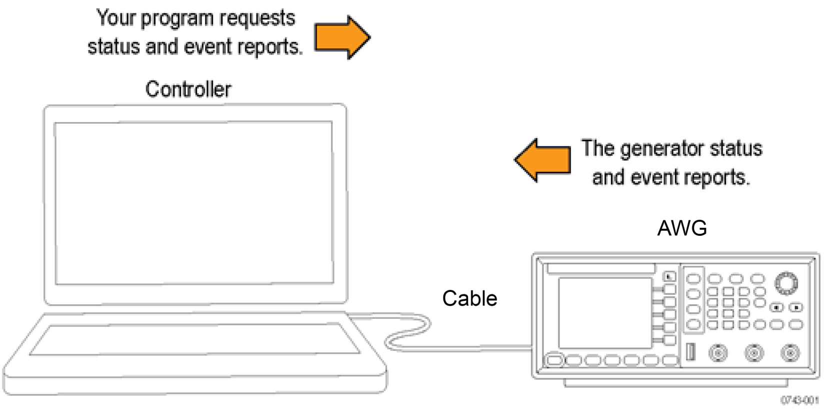


Figure 2: AWG4162 basic operation

Connecting the Interface

You can connect the AWG4162 to a computer (controller) using the rear-panel VXI-11 (LAN) or USBTMC connectors.

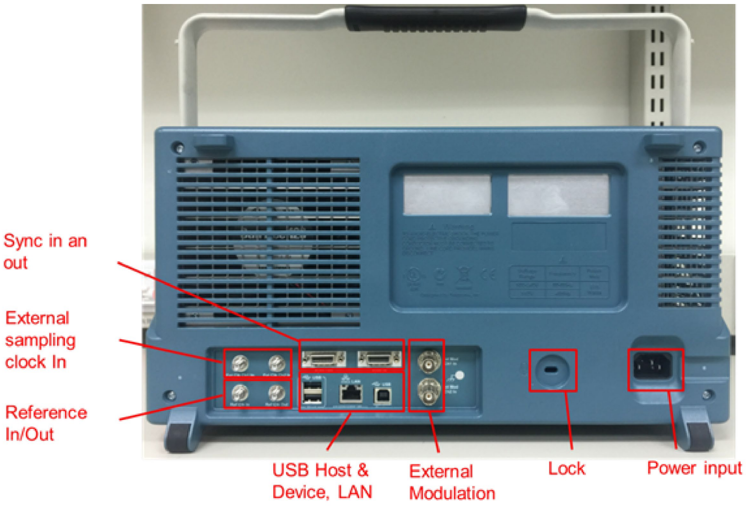


Figure 3: Rear-panel connections

Remote Control

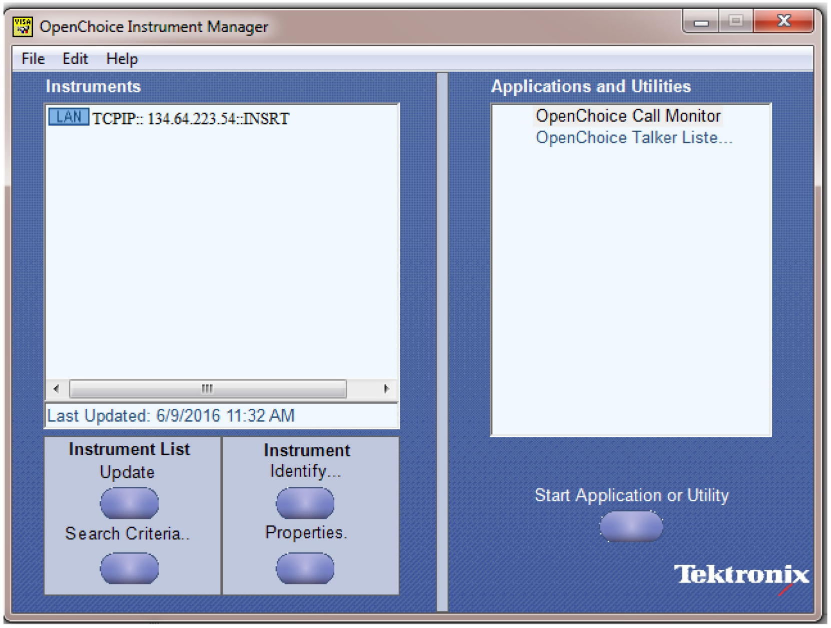
You can connect your instrument to a network for printing, file sharing, Internet access, and other functions. Consult with your network administrator and use the standard Microsoft® Windows® utilities to configure the instrument for your network. For LAN configuration, use the LAN Configuration dialog box from the control panel.

The instrument can be controlled using VXI-11 (LAN) or USBTMC protocols. This allows you to control the instrument remotely using SCPI commands. Please refer to the AWG4162 Advanced Programmer Manual for complete descriptions of all available channels.

For more information about using SCPI commands in basic mode, refer to the Syntax and Commands (on page 2-1) and Command Descriptions (on page 3-1) sections in this manual.

You can use the following procedure to communicate with your AWG4162 instrument:

1. Connect your LAN or USB cable to the instrument.
2. On the client PC, launch the Tek OpenChoice Instrument Manager window, as shown in the following figure.

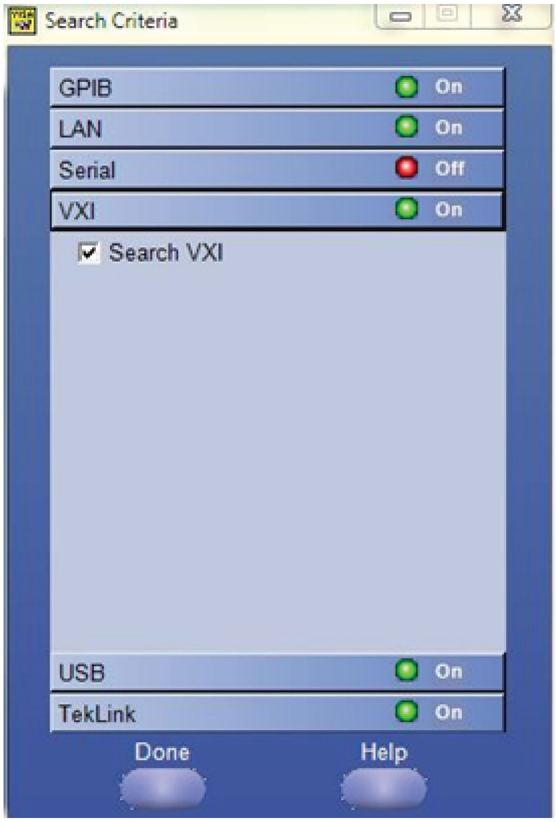


LAN TCPIP:: 134.64.223.54::INSRT

Last Updated: 6/9/2016 11:32 AM

Figure 4: Tek OpenChoice Instrument Manager window

3. Press the **Search Criteria** button and enable **LAN/USB**, as shown in the following figure.



Search Criteria

GPIB 🟢 On

LAN 🟢 On

Serial 🔴 Off

VXI 🟢 On

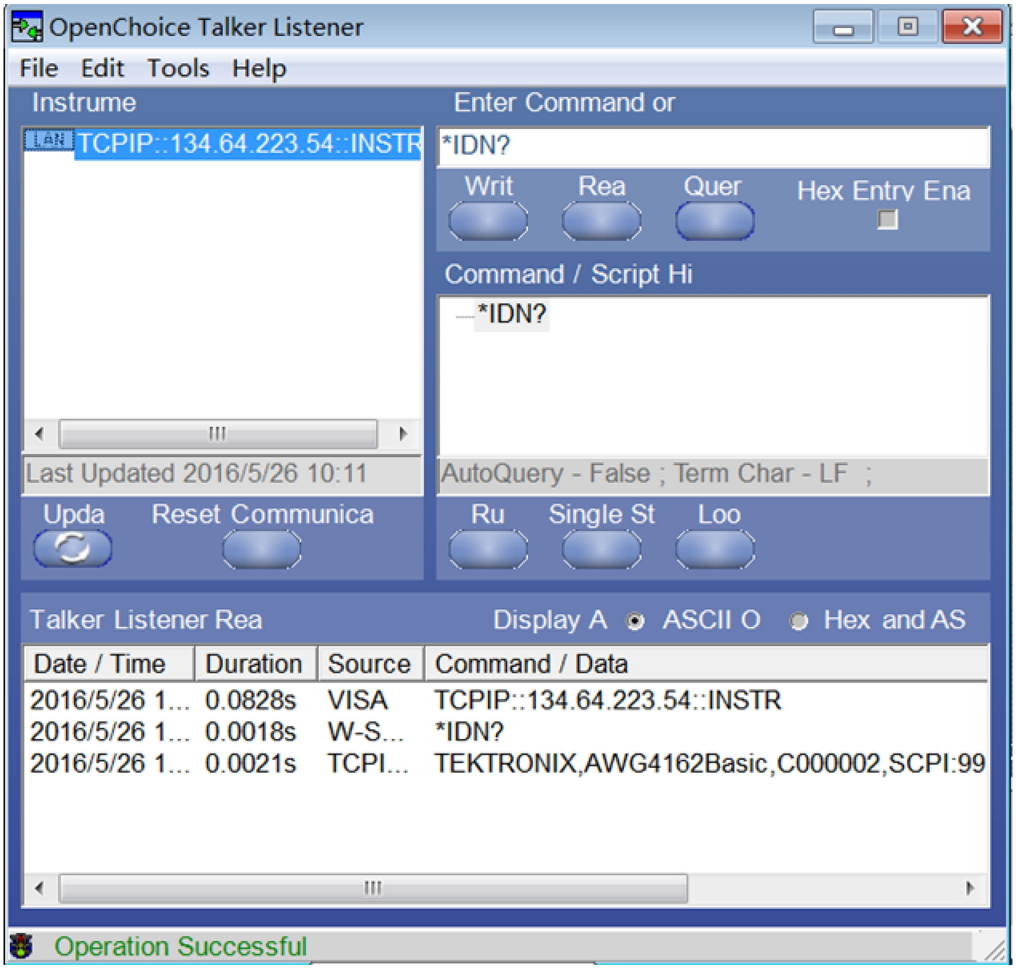
☑ Search VXI

USB 🟢 On

TekLink 🟢 On

Figure 5: Enable LAN/USB

4. Check the Instruments list to see if the AWG4162 has been correctly detected.
5. Press the **Start Application** button to open OpenChoice Talker Listener and send a `*IDN?` command.



TCPIP::134.64.223.54::INSTR

*IDN?

*IDN?

Last Updated 2016/5/26 10:11

AutoQuery - False ; Term Char - LF ;

Date / Time	Duration	Source	Command / Data
2016/5/26 1...	0.0828s	VISA	TCPIP::134.64.223.54::INSTR
2016/5/26 1...	0.0018s	W-S...	*IDN?
2016/5/26 1...	0.0021s	TCPI...	TEKTRONIX,AWG4162Basic,C000002,SCPI:99

Operation Successful

6. The instrument should respond with the string:

TEKTRONIX,AWG4162Basic,C000003,SCPI:99.0,FV:1642
where C000003 is the serial number and FV:1642 is the firmware version.

7. You can also load an existing script to run in TekVISA. Please see TekVISA Talk/Listener help for more details.

Using TekVISA

TekVISA is the Tektronix implementation of VISA (Virtual Instrument Software Architecture), an industry-standard communication protocol. VISA provides a common standard for software developers so that software from multiple vendors, such as instrument drivers, can run on the same platform. TekVISA is industry-compliant software available with selected Tektronix instruments. You can use this software to write (or draw) interoperable instrument drivers in a variety of Application Development Environments (ADEs). It implements a subset of Version 2.2 of the VISA specification for controlling GPIB and serial (RS-232) instrument interfaces locally or remotely using an Ethernet LAN connection.

Use an internet browser to access the Tektronix Downloads Web site (http://www.tek.com/downloads) and download the current TekVISA to your PC. Unzip the downloaded file in a temporary directory of your choice and run Setup.exe.

***NOTE.** The details on TekVISA concepts and operations are explained in the TekVISA Programmer Manual, which can be downloaded from the Tektronix Web site (http://www.tek.com/manuals).*

Syntax and Commands

This section provides the following information:

- Command Syntax (on page 2-1) defines the command syntax and processing conventions.
- Command Groups (on page 2-11) describes command groups and lists the commands by function.

The next section, Basic Command Descriptions (on page 3-1), describes in detail the notation of each of the basic commands in alphabetical order.

Command Syntax

You can control the operations and functions of the arbitrary waveform generator through the LAN or USBTMC interface using commands and queries. The related topics listed below describe the syntax of these commands and queries. The topics also describe the conventions that the instrument uses to process them.
See Command Groups (on page 2-11) for a listing of the commands by command group, or use the index to locate a specific command.

Backus-Naur Form Definition

This manual may describe commands and queries using the Backus-Naur Form (BNF) notation. The following table defines the standard BNF symbols.

BNF symbols and meanings

Symbol	Meaning
< >	Defined element
::=	Is defined as
|	Exclusive OR
{ }	Group; one element is required
[ ]	Optional; can be omitted
. . .	Previous element(s) may be repeated
( )	Comment

Command and Query Structure

Commands consist of set commands and query commands (usually called commands and queries). Commands change instrument settings or perform a specific action. Queries cause the instrument to return data and information about its status.

Most commands have both a set form and a query form. The query form of the command is the same as the set form except that it ends with a question mark. For example, the set command `FILEsystem:CWDirectory` has a query form, `FILEsystem:CWDirectory?`. Not all commands have both a set and a query form; some commands are set only and some are query only.

A few commands have both a set and query action. For example, the `*CAL?` command runs a self-calibration program on the instrument, then returns the result of the calibration.

Command Entry

Follow these general rules when entering commands:

- Enter commands in uppercase or lowercase
- You can precede any command with white space characters. White space characters include any combination of the ASCII control characters 00 through 09 and 0B through 20 hexadecimal (0 through 9 and 11 through 32 decimal).
- The instrument ignores commands that consist of just a combination of white space characters and line feeds.

Command Messages

A command message is a command or query name, followed by any information the instrument needs to execute the command or query. Command messages consist of five element types.

Command message elements

Symbol	Meaning
<Header>	The basic command name. If the header ends with a question mark, the command is a query. The header may begin with a colon (:) character; if the command is concatenated with other commands, the beginning colon is required. The beginning colon can never be used with command headers beginning with a star (*).
<Mnemonic>	A header subfunction. Some command headers have only one mnemonic. If a command header has multiple mnemonics, they are always separated from each other by a colon (:) character.

Symbol	Meaning
<Argument>	A quantity, quality, restriction, or limit associated with the header. Not all commands have an argument, while other commands have multiple arguments. A <space> separates arguments from the header. A <comma> separates arguments from each other.
<Comma>	A single comma between arguments of multiple-argument commands. It may optionally have white-space characters before and after the comma.
<Space>	A white-space character between command header and argument. It may optionally consist of multiple white-space characters.

The following figure shows the five command message elements.

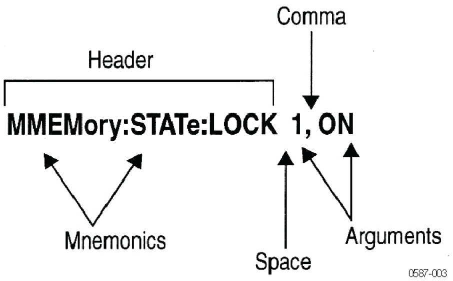


Figure 6: AWG4162 command message elements

Commands

Commands cause the instrument to perform a specific function or change one of its settings. Commands have the structure:

```
[:]<Header>[<Space><Argument>[<Comma><Argument>]...]
```

A command header is made up of one or more mnemonics arranged in a hierarchical or tree structure. The first mnemonic is the base or root of the tree and each subsequent mnemonic is a level or branch of the previous one. Commands at a higher level in the tree may affect those at a lower level. The leading colon (:) always returns you to the base of the command tree.

Queries

Queries cause the arbitrary waveform generator to return information about its status or settings. Queries have the structure:

```
[:]<Header>?
[:]<Header>?[<Space><Argument>[<Comma><Argument>]...]
```

You can specify a query command at any level within the command tree unless otherwise noted. These branch queries return information about all the mnemonics below the specified branch or level.

SCPI Commands and Queries

The arbitrary waveform generator uses a command language based on the SCPI standard. The SCPI (Standard Commands for Programmable Instruments) standard was created by a consortium to provide guidelines for remote programming of instruments. These guidelines provide a consistent programming environment for instrument control and data transfer. This environment uses defined programming messages, instrument responses, and data formats that operate across all SCPI instruments, regardless of manufacturer.

The SCPI language is based on a hierarchical or tree structure as shown in the following figure that represents a subsystem. The top level of the tree is the root node; it is followed by one or more lower-level nodes.

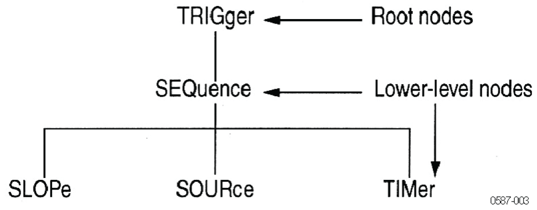


Figure 7: Example of SCPI subsystem hierarchy tree

You can create commands and queries from these subsystem hierarchy trees. Commands specify actions for the instrument to perform. Queries return measurement data and information about parameter settings.

Creating Commands

SCPI commands are created by stringing together the nodes of a subsystem hierarchy and separating each node by a colon.

In the figure above, `TRIGger` is the root node, and `SEQuence`, `SLOPe`, `TIMer`, and `SOURce` are lower-level nodes.

To create a SCPI command, start with the root node `TRIGger` and move down the tree structure, adding nodes until you reach the end of a branch. Most commands and some queries have parameters; you must include a value for these parameters. If you specify a parameter value that is out of range, the parameter will be set to a default value. The command descriptions list the valid values for all parameters.

For example, `TRIGger:SEQuence:SOURce EXTernal` is a valid SCPI command created from the hierarchy tree. (See the figure in SCPI Commands and Queries (on page 2-4).)

Parameters

Parameters are indicated by angle brackets, such as <file_name>.There are several different types of parameters. (See the table in Parameter Types (on page 2-6).) The parameter type is listed after the parameter. Some parameter types are defined specifically for the arbitrary waveform generator command set and some are defined by SCPI.

Creating Queries

To create a query, start at the root node of a tree structure, move down to the end of a branch, and add a question mark. `TRIGger:SEQuence:SOURce?` is an example of a valid SCPI query using the hierarchy tree in the figure in SCPI Commands and Queries (on page 2-4).

Query Responses

The query causes the arbitrary waveform generator to return information about its status or settings. When a query is sent to the arbitrary waveform generator, only the values are returned. When the returned value is a mnemonic, it is noted in abbreviated format, as shown in the following table.

Query response examples

Query Command	Response
SOURce:PULSe:DCYcle?	50.0
OUTPut:POLarity?	NORM

Parameter Types

Every parameter in the command and query descriptions is of a specified type. The parameters are enclosed in brackets, such as <value>. The parameter type is listed after the parameter and is enclosed in parentheses, for example, (Boolean). Some parameter types are defined specifically for the arbitrary waveform generator command set and some are defined by SCPI.

Parameter types used in syntax descriptions

Parameter type	Description	Example
arbitrary block [1]	A specified length of arbitrary data	#512234xxxxx ... where 5 indicates that the following 5 digits (12234) specify the length of the data in bytes; xxxxx ... indicates the data or #0xxxxx ...<LF><&EOI> Boolean numbers or values ON or ≠ 0
boolean	Boolean numbers or values	ON or ≠ 0 OFF or 0
discrete	A list of specific values	MINimum, MAXimum
binary	Binary numbers	#B0110
octal	Octal numbers	#Q57, #Q3
hexadecimal [2]	Hexadecimal numbers (0-9, A, B, C, D, E, F)	#HAA, #H1
NR1 [2] numeric	Integers	0, 1, 15, -1
NR2 [2, 3] numeric	Decimal numbers	1.2, 3.141516, -6.5
NR3 [2] numeric	Floating-point numbers	3.1415E-9, -16.1E5
NRf [2] numeric	Flexible decimal number that may be type NR1, NR2, or NR3	See NR1, NR2, and NR3 examples
string [4]	Alphanumeric characters (must be within quotation marks)	"Testing 1, 2, 3"

1. Defined in ANSI/IEEE 488.2 as "Definite Length Arbitrary Block Response Data."
2. An ANSI/IEEE 488.2-1992-defined parameter type.
3. Some commands and queries will accept an octal or hexadecimal value, even though the parameter type is defined as NR1.
4. Defined in ANSI/IEEE 488.2 as "String Response Data."

Special Characters

The Line Feed (LF) character (ASCII 10), and all characters in the range of ASCII 127 through 255 are defined as special characters. These characters are used in arbitrary block arguments only; using these characters in other parts of any command yields unpredictable results.

Abbreviating Commands, Queries, and Parameters

You can abbreviate most SCPI commands, queries, and parameters to an accepted short form. This manual shows these short forms as a combination of uppercase and lowercase letters. The uppercase letters indicate the accepted short form of a command. As shown in the following figure, you can create a short form by using only the uppercase letters. The accepted short form and the long form are equivalent and request the same action of the instrument.

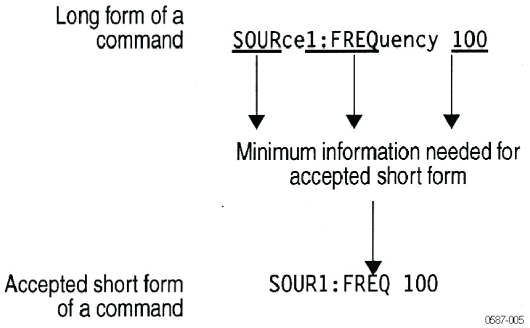

Long form of a command
SOURce1:FREQuency 100
Minimum information needed for accepted short form
Accepted short form of a command
SOUR1:FREQ 100
0587-005

Figure 8: Example of abbreviating a command

NOTE. *The numeric suffix of a command or query may be included in either the long form or short form; the arbitrary waveform generator defaults to "1" if no suffix is used.*

Concatenating Commands and Queries

You can chain (concatenate) several commands or queries together into a single message. The instrument executes chained commands in the order received.

To create a chained message, first create a command or query, add a semicolon (;), and then add more commands or queries and semicolons until the message is complete. If the command following a semicolon is a root node, precede it with a colon (:). The following figure illustrates a chained message consisting of several commands and queries. The chained message should end in a command or query, not a semicolon. Responses to any queries in your message are separated by semicolons.

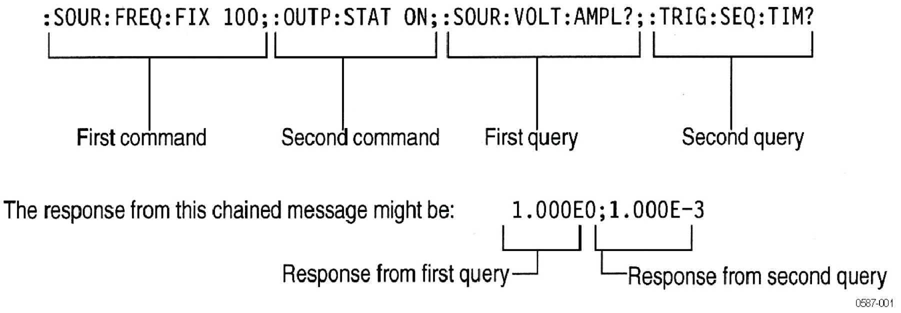
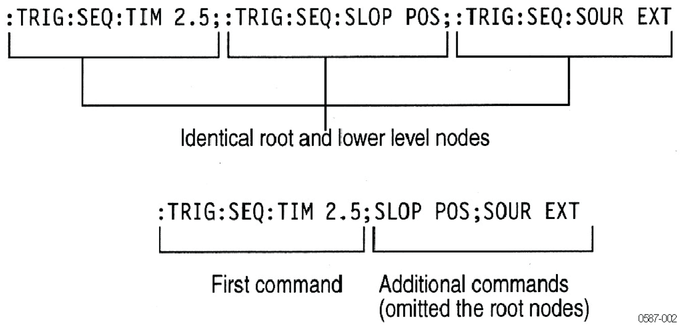

:SOUR:FREQ:FIX 100;:OUTP:STAT ON;:SOUR:VOLT:AMPL?;:TRIG:SEQ:TIM?
First command
Second command
First query
Second query
The response from this chained message might be:
1.000E0;1.000E-3
Response from first query
Response from second query
0587-001

Figure 9: Example of chaining commands and queries

If a command or query has the same root and lower-level nodes as the previous command or query, you can omit these nodes. In the following figure, the second command has the same root node (`TRIGger:SEQuence`) as the first command, so these nodes can be omitted.



```
:TRIG:SEQ:TIM 2.5;:TRIG:SEQ:SLOP POS;:TRIG:SEQ:SOUR EXT
```

```
:TRIG:SEQ:TIM 2.5;SLOP POS;SOUR EXT
```

Figure 10: Example of omitting root and lower-level nodes in a chained message

NOTE. *Never precede a star (*) command with a semicolon or colon.*

Unit and SI Prefixes

If the decimal numeric argument refers to voltage, frequency, impedance, or time, you can express it using SI units instead of using the scaled explicit-point input value format <NR3>. (SI units are units that conform to the Systeme International d'Unites standard.) For example, you can use the input format 200 mV or 1.0 MHz instead of 200.0E-3 or 1.0E+6, respectively, to specify voltage or frequency.

Omit the unit when you describe commands, but include the SI unit prefix. Enter both uppercase and lowercase characters. The following list shows examples of units you can use with the commands.

Example units

Symbol	Meaning
dB	decibel (relative amplitude)
dBm	decibel (absolute amplitude)
DEG	degree (phase)
Hz	hertz (frequency)
PCT	Percent (%)
s	second (time)
V	voltage

The SI prefixes, which must be included, are shown in the following table. You can enter both uppercase and lowercase characters.

SI prefixes and their indexes

SI prefix[1]	Z	A	F	P	N	U[2]	M	K	MA[3]	G	T	PE	EX
Corresponding power	10^{-21}	10^{-18}	10^{-15}	10^{-12}	10^{-9}	10^{-6}	10^{-3}	10^{+3}	10^{+6}	10^{+9}	10^{+12}	10^{+15}	10^{+18}

[1] The prefix m/M indicates 10^{-3} when the decimal numeric argument denotes voltage or time, but indicates 10^{6} when it denotes frequency.

[2] The prefix u/U is used instead of "µ".

[3] When the unit is "Hz", "M" may be used instead of "MA" so that the frequency can be represented by "MHz".

You can omit a unit in a command, but you must include the unit when using an SI prefix. For example, frequency of 15 MHz can be described as follows:

15.0E6, 1.5E7Hz, 15000000, 15000000Hz, 15MHz

("15M" is not allowed.)

NOTE. *You can use either lowercase or uppercase unit prefixes. The following examples have the same result, respectively.*

170mHz, 170mHz, 170MHz
250mv, 250mV, 250MV

Because M (m) can be interpreted as 1E-3 or 1E6 depending on the units, use mV for V, and MHz for Hz.

General Rules for Using SCPI Commands

The following are three general rules for using SCPI commands, queries, and parameters:

- You can use single (' ') or double (" ") quotation marks for quoted strings, but you cannot use both types of quotation marks for the same string.

Correct: "This string uses quotation marks correctly."

Correct: 'This string also uses quotation marks correctly.'

Incorrect: "This string does not use quotation marks correctly.'

- You can use uppercase, lowercase, or a mixture of both cases for all commands, queries, and parameters.

```
:SOURCE:FREQUENCY 10MHZ
```

is the same as

```
 :source:frequency 100mhz
```

and

```
SOURCE:frequency 10MHZ
```

NOTE. *Literal strings (quoted) are case-sensitive. For example, file names are case-sensitive.*

- No embedded spaces are allowed between or within nodes.

Correct `:OUTPUT:FILTER:LPASS:FREQUENCY 200MHZ`

Incorrect `:OUTPUT: FILTER: LPASS:FREQUENCY 200MHZ`

IEEE 488.2 Common Commands

Description

ANSI/IEEE Standard 488.2 defines the codes, formats, protocols, and usage of common commands and queries used on the interface between the controller and the instruments. The arbitrary waveform generator complies with this standard.

Command and Query Structure

The syntax for an IEEE 488.2 common command is an asterisk (*) followed by a command, and optionally, a space and parameter value. The syntax for an IEEE 488.2 common query is an asterisk (*) followed by a query and a question mark. The following are examples of common commands:

- *ESE 16
- *CLS

The following are examples of common queries:

- *ESR?
- *IDN?

Command Groups

This section lists the commands organized by functional group. The Command Descriptions (on page 3-1) section lists all commands alphabetically.

Calibration and Diagnostic Commands

Calibration and diagnostic commands let you initiate the instrument self-calibration routines and examine the results of diagnostic tests. The following table describes the calibration and diagnostic commands.

Calibration and diagnostic commands

Command	Description
*CAL? (on page 3-1)	Perform self-calibration and return result status
DIAGnostic (on page 3-4)	Perform a self-test and return result status
*TST? (on page 3-81)	Perform a self-test and return result status

Date and Time Commands

The date and time commands allow you to set or query the system date and time. The following table describes the date and time commands.

Date and time commands

Command	Description
DATE (on page 3-3)	Set or query the system date
TIME (on page 3-77)	Set the system time

File System Commands

You can use the file commands to manipulate files and directories in the file system. The following table describes the file commands.

File commands

Command	Description
FILEsystem:CATalog? (on page 3-6)	Query the present state of the file system
FILEsystem:COPY (on page 3-6)	Copy a file from one location in the file system to another location
FILEsystem:CWDirectory (on page 3-7)	Change the current working directory in the file system
FILEsystem:DELete (on page 3-8)	Delete a file or directory in the file system
FILEsystem:HARDdisk? (on page 3-8)	Query the hard disk drive
FILEsystem:LOCK (on page 3-9)	Lock or unlock a file or directory in the file system, or query if a file or directory is locked
FILEsystem:MDIRectory (on page 3-9)	Create a directory in the file system
FILEsystem:UDISk? (on page 3-10)	Query the u-disk drive

Memory Commands

Memory commands let you change set up memory attributes. The following table describes the memory commands.

Memory commands

Command	Description
MEMory:RECall (on page 3-11)	Recall a specified project file in the file system
MEMory:SAVE (on page 3-12)	Save the current project file in the file system
MEMory:STATe:DELete (on page 3-12)	Delete the setup memory
MEMory:STATe:LOCK (on page 3-13)	Lock or unlock the setup memory and query whether the memory is locked
MEMory:STATe:RECall:AUTo (on page 3-13)	Enable or disable automatic recall from setup memory and query whether automatic recall is enabled
MEMory:STATe:VALid? (on page 3-14)	Query the availability of setup memory
*RCL (on page 3-18)	Recall instrument settings from setup memory

Command	Description
*SAV (on page 3-19)	Save instrument settings to setup memory

Mass Memory Commands

Mass memory commands let you change mass memory attributes. The following table describes the mass memory commands.

Mass memory commands

Command	Description
MMEMory:LOAD:STATe (on page 3-14)	Copy a setup file to internal setup memory
MMEMory:STORe:STATe (on page 3-15)	Copy a setup file from setup memory to a specified file in the file system

Output Commands

Output commands let you set output attributes. The following table describes the output commands.

Output commands

Command	Description
OUTPut[1|2]:IMPedance (on page 3-16)	Set or query the output load impedance
OUTPut[1|2]:POLarity (on page 3-17)	Set or query the polarity of the waveform on a specified channel
OUTPut[1|2][:STATe] (on page 3-18)	Set or query the output state (on or off) on a specified channel

HCopy Command

This command copies the current display screen to a .bmp image file.

HCopy Command

Command	Description
HCOPy:SDUMp[:IMMediate] (on page 3-10)	Create a screen shot of the display screen

Source Commands

Source commands let you set waveform output parameters. The following table describes the source commands.

Source commands

Command	Description
[SOURce]:ROSCillator:SOURce (on page 3-57)	Set or query the clock reference input
[SOURce[1|2]]:AM[:DEPTh] (on page 3-20)	Set or query amplitude modulation depth
[SOURce[1|2]]:AM:INTernal:FREQuency (on page 3-22)	Set or query internal modulation frequency
[SOURce[1|2]]:AM:INTernal:FUNCtion (on page 3-22)	Set or query modulation waveform setting
[SOURce[1|2]]:AM:INTernal:FUNCtion:EFILe (on page 3-21)	Set or query a modulating waveform EFILe name
[SOURce[1|2]]:AM:SOURce (on page 3-22)	Set or query the amplitude modulation source
[SOURce[1|2]]:AM:STATe (on page 3-23)	Enable or disable AM modulation and query the state of AM modulation
[SOURce[1|2]]:BURSt:MODE (on page 3-23)	Set or query the burst mode
[SOURce[1|2]]:BURSt:NCYCles (on page 3-24)	Set or query the burst count
[SOURce[1|2]]:BURSt[:STATe] (on page 3-25)	Enable or disable burst mode and query the burst mode
[SOURce[1|2]]:BURSt:TDELay (on page 3-25)	Set or query burst mode trigger delay time
[SOURce[1|2]]:COMBine:FEED (on page 3-2)	Set or query whether to add internal noise to an output signal for the specified channel
[SOURce[1|2]]:FM[:DEViation] (on page 3-27)	Set or query the peak frequency deviation
[SOURce[1|2]]:FM:INTernal:FREQuency (on page 3-27)	Set or query the internal modulation frequency
[SOURce[1|2]]:FM:INTernal:FUNCtion (on page 3-28)	Set or query the internal modulation waveform
[SOURce[1|2]]:FM:INTernal:FUNCtion:EFILe (on page 3-29)	Set or query the modulating waveform EFILe name
[SOURce[1|2]]:FM:SOURce (on page 3-29)	Set or query the frequency modulation source

Command	Description
[SOURce[1|2]]:FM:STATe (on page 3-30)	Enable or disable frequency modulation and query the FM modulation state
[SOURce[1|2]]:FREQuency:CENTer (on page 3-30)	Set or query the center frequency of a sweep
[SOURce[1|2]]:FREQuency:CONCurrent[:STATe] (on page 3-31)	Enable or disable copying of the frequency or period of one channel to another channel and query the concurrent state
[SOURce[1|2]]:FREQuency[:CW|:FIXed] (on page 3-32)	Set or query the output waveform frequency
[SOURce[1|2]]:FREQuency:MODE (on page 3-33)	Set or query the frequency sweep state
[SOURce[1|2]]:FREQuency:SPAN (on page 3-33)	Set or query the sweep frequency span
[SOURce[1|2]]:FREQuency:STARt (on page 3-34)	Set or query the sweep start frequency
[SOURce[1|2]]:FREQuency:STOP (on page 3-35)	Set or query the sweep stop frequency
[SOURce[1|2]]:FSKey[:FREQuency] (on page 3-35)	Set or query the FSK hop frequency
[SOURce[1|2]]:FSKey:INTernal:RATE (on page 3-36)	Set or query the FSK internal modulation rate
[SOURce[1|2]]:FSKey:SOURce (on page 3-36)	Set or query the FSK modulation source
[SOURce[1|2]]:FSKey:STATe (on page 3-37)	Enable or disable FSK modulation and query the FSK modulation state
[SOURce[1|2]]:PSKey[:FREQuency] (on page 3-38)	Set or query PSK modulation frequency
[SOURce[1|2]]:PSKey:PHASe[:ADJust] (on page 3-38)	Set or query the PSK modulating signal phase
[SOURce[1|2]]:PSKey:SOURce (on page 3-39)	Set or query PSK modulation source
[SOURce[1|2]]:PSKey:STATe (on page 3-40)	Enable or disable PSK modulation and query the PSK modulation state
[SOURce[1|2]]:FUNCtion:EFILe (on page 3-40)	Set or query the output waveform EFILe name
[SOURce[1|2]]:FUNCtion:RAMP:SYMMetry (on page 3-41)	Set or query ramp waveform symmetry

Command	Description
[SOURce[1|2]]:FUNCtion[:SHAPe] (on page 3-41)	Set or query the shape of the output waveform
[SOURce[1|2]]:PHASe[:ADJust] (on page 3-43)	Set or query the output waveform phase
[SOURce[1|2]]:PHASe:INITiate (on page 3-43)	Initiate output waveform phase synchronization
[SOURce[1|2]]:PM[:DEViation] (on page 3-44)	Set or query the peak frequency deviation of phase modulation
[SOURce[1|2]]:PM:INTernal:FREQuency (on page 3-45)	Set or query the internal modulation frequency
[SOURce[1|2]]:PM:INTernal:FUNCtion (on page 3-45)	Set or query the internal modulation waveform
[SOURce[1|2]]:PM:INTernal:FUNCtion:EFILe (on page 3-46)	Set or query the PM modulating waveform EFILe name
[SOURce[1|2]]:PM:SOURce (on page 3-47)	Set or query the phase modulation source
[SOURce[1|2]]:PM:STATe (on page 3-47)	Enable or disable PM modulation and query the PM modulation state
SOURce<3|4>:POWer[:LEVel][:IMMediate][:AMPLitude] (on page 3-48)	Set or query the internal noise level of the output signal
[SOURce[1|2]]:PULSe:DCYCle (on page 3-49)	Set or query the pulse waveform duty cycle
[SOURce[1|2]]:PULSe:DELay (on page 3-49)	Set or query the pulse waveform lead delay
[SOURce[1|2]]:PULSe:HOLD (on page 3-50)	Set or query the pulse waveform parameter
[SOURce[1|2]]:PULSe:PERiod (on page 3-50)	Set or query the pulse waveform period
[SOURce[1|2]]:PULSe:TRANsition[:LEADing] (on page 3-51)	Set or query the pulse waveform leading edge time
[SOURce[1|2]]:PULSe:TRANsition:TRAiling (on page 3-52)	Set or query the pulse waveform trailing edge time
[SOURce[1|2]]:PULSe:WIDTh (on page 3-52)	Set or query the pulse waveform width
[SOURce[1|2]]:PWM:INTernal:FREQuency (on page 3-54)	Set or query the pulse width modulation frequency

Command	Description
[SOURce[1|2]]:PWM:INTernal:FUNCtion (on page 3-55)	Set or query the pulse width modulation waveform
[SOURce[1|2]]:PWM:INTernal:FUNCtion:EFILe (on page 3-55)	Set or query the modulating waveform EFILe name
[SOURce[1|2]]:PWM:SOURce (on page 3-55)	Set or query the pulse width modulation source
[SOURce[1|2]]:PWM:STATe (on page 3-56)	Set or query the pulse width modulation status
[SOURce[1|2]]:PWM[:DEViation]:DCYCle (on page 3-56)	Set or query the pulse width modulation deviation
[SOURce[1|2]]:SWEep:HTIMe (on page 3-59)	Set or query the sweep hold time
[SOURce[1|2]]:SWEep:MODE (on page 3-58)	Set or query the sweep mode
[SOURce[1|2]]:SWEep:RTIMe (on page 3-59)	Set or query the sweep return time
[SOURce[1|2]]:SWEep:SPACing (on page 3-59)	Set or query the sweep spacing
[SOURce[1|2]]:SWEep:TIME (on page 3-60)	Set or query the sweep time
[SOURce[1|2]]:VOLTage:CONCurrent[:STATe] (on page 3-60)	Enable or disable copying of the voltage level of one channel to another channel and query the concurrent state
[SOURce[1|2]]:VOLTage:LIMit:HIGH (on page 3-64)	Set or query the output amplitude upper limit
[SOURce[1|2]]:VOLTage:LIMit:LOW (on page 3-65)	Set or query the output amplitude lower limit
[SOURce[1|2]]:VOLTage[:LEVel][:IMMediate]:OFFSet (on page 3-62)	Set or query the output offset voltage
[SOURce[1|2]]:VOLTage:UNIT (on page 3-65)	Set or query the output amplitude units
[SOURce[1|2]]:VOLTage[:LEVel][:IMMediate]:HIGH (on page 3-61)	Set or query the output amplitude high level
[SOURce[1|2]]:VOLTage[:LEVel][:IMMediate]:LOW (on page 3-62)	Set or query the output amplitude low level
[SOURce[1|2]]:VOLTage[:LEVel][:IMMediate][:AMPLitude] (on page 3-65)	Set or query the output amplitude
[SOURce[1|2]]:VOLTage:VOCM (on page 3-66)	Set or query the VOCM

Status Commands

Status commands let you determine the status of the instrument. The following table describes the status commands.

Status commands

Command	Description
*CLS (on page 3-3)	Clear all event registers and queues
*ESE (on page 3-4)	Set or query the Event Status Enable Register
*ESR? (on page 3-5)	Return the contents of the Standard Event Status Register
*SRE (on page 3-66)	Set or query the Service Request Enable Register
*STB? (on page 3-71)	Read the Status Byte Register
STATus:OPERation:CONDition? (on page 3-67)	Return the contents of the Operation Condition Register
STATus:OPERation:ENABle (on page 3-68)	Set or query the mask for the Operation Enable Register
STATus:OPERation[:EVENt]? (on page 3-68)	Return the value in the Operation Event Register
STATus:PRESet (on page 3-69)	Preset SCPI Enable Register
STATus:QUEStionable:CONDition? (on page 3-69)	Return the contents of the Questionable Condition Register
STATus:QUEStionable:ENABle (on page 3-70)	Set or query the mask for the Questionable Enable Register
STATus:QUEStionable[:EVENt]? (on page 3-70)	Return the value in the Questionable Event Register

System Commands

System commands let you control miscellaneous instrument functions. The following table describes the system commands.

System commands

Command	Description
*IDN? (on page 3-11)	Return identification information
*RST (on page 3-19)	Reset the instrument to factory defaults
SYSTem:BEEPer[:IMMediate] (on page 3-71)	Generate an audible tone
SYSTem:BEEPer:STATe (on page 3-72)	Set or query the beeper state

Command	Description
SYSTem:ERRor[:NEXT]? (on page 3-74)	Return the contents of the error event queue
SYSTem:KCLick[:STATe] (on page 3-73)	Enable or disable the key-click sound and query the status of key clicks
SYSTem:KLOCk[:STATe] (on page 3-73)	Lock or unlock the front-panel controls and query the lock state of the controls
SYSTem:ULANguage? (on page 3-74)	Query the language for the display screen
SYSTem:VERSion? (on page 3-74)	Return the SCPI conformance version information

Synchronization Commands

Synchronization commands let you synchronize the operation of the instrument. The following table describes the synchronization commands.

Synchronization commands

Command	Description
*OPC (on page 3-16)	Set or query the operation complete message
*WAI (on page 3-82)	Wait to continue until pending commands complete

Trace Commands

Trace commands allow you to save, recall, set, and query data points in arbitrary buffer memory. The following table describes the trace commands.

Trace commands

Command	Description
TRACe|DATA[:DATA] (on page 3-76)	Save or query waveform data in the Arb buffer
TRACe|DATA:POINts? (on page 3-78)	Set or query the number of points in the Arb buffer for waveform data
TRACe|DATA:SAVE (on page 3-77)	Save the contents of arbitrary buffer memory to a file in the system
TRACe|DATA:RECall (on page 3-77)	Recall the contents of arbitrary buffer memory

Trigger Commands

The trigger commands let you control all aspects of triggering. The following table describes the trigger commands.

Trigger commands

Command	Description
ABORt (on page 3-1)	Reset and initialize the trigger system
TRIGger[:SEQuence][:IMMediate] (on page 3-78)	Generate a trigger event
TRIGger[:SEQuence]:THREshold (on page 3-80)	Set or query the threshold of an input signal
TRIGger[:SEQuence]:SLOPe (on page 3-79)	Set or query the slope of the trigger signal
TRIGger[:SEQuence]:SOURce (on page 3-79)	Set or query the source of the trigger signal
TRIGger[:SEQuence]:TIMer (on page 3-80)	Set or query the internal rate

Basic Command Descriptions

ABORt

This command initializes all the current trigger system parameters and resets all trigger sequences. There is no query form of this command.

Group

Trigger

Related Commands

None

Syntax

```
ABORt
```

Arguments

None

Examples

```
ABORt
```

Reset the trigger system.

*CAL?

This command performs a calibration and returns 0 (Pass) or a calibration error code.

NOTE. *Before executing this command, allow a 30-minute warm-up period after powering on the instrument. If you do not allow the instrument to reach a valid temperature before performing calibration, the calibration will not be valid. You can go to* ***Basic application ➔ System ➔ Tools ➔ Warm Up Timer*** *to do the warm-up.*

Group

Calibration and Diagnostic

Related Commands

CALibration (on page 3-2)

Syntax

```
*CAL?
```

Arguments

None

Returns

`<NR1>`

Where:

`<NR1> ≠` 0 indicates that the arbitrary waveform generator detected an error.

`<NR1> = 0` indicates that the calibration completed without errors.

Examples

```
*CAL?
```

Performs a calibration and returns results. Example return: 0, which indicates that the calibration completed without any errors.

CALibration

This command performs a calibration. The query form of this command performs calibration and returns 0 (pass) or a calibration error code.

Group

Calibration and Diagnostic

Related Commands

*CAL? (on page 3-1)

Syntax

```
CALibration
CALibration?
```

Arguments

None

Returns

```
<NR1>
```

Examples

```
CALibration
```

Perform a calibration.

```
CALibration?
```

Perform a calibration and return results. For example, the command might return 0, which indicates that the calibration completed without any errors.

*CLS

This command clears all the event registers and queues that are used in the arbitrary waveform generator status and event reporting system. There is no query form of this command.

Group

Status

Related Commands

None

Syntax

```
*CLS
```

Arguments

None

Examples

```
*CLS
```

Clear all the event registers and queues.

DATE

This command sets the system date. The query form of this command queries the system date.

Group

Date and Time

Related Commands

TIME (on page 3-77)

Syntax

```
DATE <date_string>
DATE?
```

Arguments

`<date_string>::<string>` Specifies a date in the format "20xx-xx-xx".

Returns

`<string>`

Example

```
DATE "2016-06-20"
```

Set the system date to June 20th, 2016

```
DATE?
```

Example return: "DATE 2016-06-20," which means the system date is June 20th, 2016.

DIAGnostic

This command performs a self-test. The query form of this command returns the results after executing the test.

NOTE. *Before executing this command, allow a 30-minute warm-up period after powering on the instrument. If you do not allow the instrument to reach a valid temperature before performing calibration, the calibration will not be valid. You can go to* ***Basic application ➔ System ➔ Tools ➔ Warm Up Timer*** *to do the warm-up.*

Group

Calibration and Diagnostic

Related Commands

*TST? (on page 3-81)

Syntax

```
DIAGnostic
DIAGnostic?
```

Arguments

None

Returns

`<NR1> = 0`	Indicates that the self-test completed without errors.
`<NR1> ≠ 0`	Indicates that the arbitrary waveform generator detected an error.

Examples

```
DIAGnostic
```

Perform a self-test.

```
DIAGnostic?
```

Perform a self-test and return results.

*ESE

This command sets or queries the bits in the Event Status Enable Register (ESER) used in the status and events reporting system of the arbitrary waveform generator. The query form of this command returns the contents of the ESER.

Group

Status

Related Commands

*CLS (on page 3-3)
*ESR? (on page 3-5)
*SRE (on page 3-66)
*STB? (on page 3-71)

Syntax

```
*ESE <bit_value>
*ESE?
```

Arguments

`<bit_value>::=<NR1>` A value from 0 through 255. The binary bits of the ESER are set according to this value.

Returns

```
<bit_value>
```

Examples

```
*ESE 177
```

Sets the ESER to 177 (binary 10110001), which sets the PON, CME, EXE, and OPC bits.

```
*ESE?
```

Example return: 186, which indicates that the ESER contains the binary value 10111010.

*ESR?

This command returns the contents of the Standard Event Status Register (SESR) used in the status events reporting system in the arbitrary waveform generator. This command clears the SESR when it reads it. This command is query only.

Group

Status

Related Commands

*CLS (on page 3-3)
*ESE? (on page 3-4)
*SRE (on page 3-66)
*STB? (on page 3-71)

Syntax

```
*ESR?
```

Arguments

None

Returns

`<NR1>` Indicates that the contents of the SESR is a decimal integer.

Examples

```
*ESR?
```

Example return: 181, which indicates that the SESR contains the binary number 10110101.

FILEsystem:CATalog?

This command returns the current state of the file system. This command is query only.

Group

File System

Related Commands

FILEsystem:CWDirectory (on page 3-7)

Syntax

```
FILESystem:CATalog?
```

Arguments

None

Returns

```
<NR1>,<NR1>[,<file_name>,<file_type>,<file_size>]...
```

Where:

The first <NR1> indicates the total amount of storage currently used, in bytes.

The second <NR1> indicates the available free space in storage, in bytes.

`<file_name>` is the exact name of a file.

`<file_type>` is DIR for directory, otherwise it is blank.

`<file_size>` is the size of the file, in bytes.

Examples

```
FILESystem:CATalog?
```

Example return:

32751616,27970560,"SAMPLE1.TFS,,5412"

FILEsystem:COPY

This command copies a file in the file system to another file in the file system. This command causes an error if filename1 (source file) in the file system is NULL, if filename2 (destination file) is locked, or if the current directory is locked. You cannot create a new file if the directory is locked. There is no query form of this command.

Group

File System

Related Commands

FILEsystem:LOCK (on page 3-9)
FILEsystem:DELete (on page 3-8)

Syntax

```
FILEsystem:COPY <filename1>,<filename2>
```

Arguments

`<filename1>::=<string>` specifies a source file name in the file system. `<filename2>::=<string>` specifies a destination file name in the file system.

Examples

```
FILEsystem:COPY "SAMPLE1.TFW","SAMPLE2.TFW"
```

Copies the file named "SAMPLE1.TFW" to the file "SAMPLE2.TFW" in the file system.

FILEsystem:CWDirectory

This command changes or queries the current working directory in the file system.

Group

File System

Related Commands

None

Syntax

```
FILEsystem:CWD [<directory_name>]
FILEsystem:CWDirectory?
```

Arguments

`<directory_name>::=<string>` indicates the working directory in the file system that you want to change to.

If you do not specify a parameter, the directory is set to the default value. The default value is "D:\Tektronix\AWG4000\Basic".

Returns

```
<directory_name>::=<string>
```

Examples

```
FILEsystem:CDIRectory "D:\USER"
```

Change the current directory to D:\USER.

FILEsystem:DELete

This command deletes a file or directory from the file system. If a specified file in file storage locked and cannot be overwritten or deleted, this command causes an error. You can delete a directory if it is empty. There is no query form of this command.

Group

File System

Related Commands

None

Syntax

```
FILEsystem:DELete <file_name>
```

Arguments

`<file_name>::=<string>` specifies a file to be deleted.

Examples

```
FILEsystem:DELete "TEK001.TFW"
```

Delete the file "TEK001.TFW" in the file system.

FILEsystem:HARDdisk?

This command queries the hard disk drive. This command is query only.

Group

File System

Related Commands

None

Syntax

```
FILEsystem:HARDdisk?
```

Return

```
<driver1;driver2;driver3…>
```

Examples

```
FILEsystem:HARDdisk?
```

Example return: "C:\;D:\;E:\;F:\"

FILEsystem:LOCK

This command sets or queries whether to lock a file or directory in the file system. If you lock a file or directory, you cannot overwrite or delete it.

Group

File System

Related Commands

None

Syntax

```
FILEsystem:LOCK <file_name>,{ON|OFF|<NR1>}
FILEsystem:LOCK? <file_name>
```

Arguments

ON or <NR1> ≠ 0 locks a file or directory in the file system.

OFF or <NR1> = 0 allows you to overwrite or delete a file or directory in the file system.

Returns

<NR1>

Examples

```
FILEsystem:LOCK "SETUP1.TFS",ON
```

Lock the file "SETUP1.TFS".

FILEsystem:MDIRectory

This command creates a directory in the file system. If the specified directory is locked in the file system, this command causes an error. There is no query form of this command.

Group

File System

Related Commands

None

Syntax

```
FILEsystem:MDIRectory <directory_name>
```

Arguments

<directory_name>::=<string> specifies a directory name to be created.

Examples

```
FILEsystem:MDIRectory "SAMPLE1"
```

Create a directory named "SAMPLE1" in the file system.

FILEsystem:UDISk?

This command queries the u-disk drive. This command is query only.

Group

File System

Related Commands

None

Syntax

```
FILEsystem:UDISk?
```

Return

```
<driver1;driver2;driver3…>
```

Examples

```
FILEsystem:UDISk?
```

Example return: "G:\;H:\"

HCOPy:SDUMp[:IMMediate]

This command copies the current screen shot to a specified file in the file system. The file will be named with the format "year-month-day-hour-minus-seconds.bmp" in the default directory "D:\Tektronix\AWG4162\Basic". There is no query form of this command.

Group

HCOPy

Related Commands

None

Syntax

```
HCOPy:SDUMp[:IMMediate]
```

Arguments

None

Examples

```
HCOPy:SDUMp
```

Copy the screen shot to the file system. Example file path: "D:\Tektronix\AWG4162\Basic\2015-5-1-3-16-32.bmp".

*IDN?

This command returns identification information for the arbitrary waveform generator. This command is query only.

Group

System

Related Commands

None

Syntax

```
*IDN?
```

Arguments

None

Returns

```
<Manufacturer>,<Model>,<Serial Number>,<Firmware Level>
```

Where:

```
<Manufacturer>::=TEKTRONIX
<Model>::={AWG4162}
<Serial Number>::{XXXXXXX} (indicates an actual serial number)
<Firmware Level>::=SCPI:99.0 FV:1.0
```

Examples

```
*IDN?
```

Example return: TEKTRONIX,AWG4162,C00002,SCPI:99.0 FV:1.0

MEMory:RECall

This command recalls a project file in the file system from a specified file to the current project. If the specified file does not exist or its format is wrong, this command causes an error. There is no query form of this command.

Group

Memory

Related Commands

MEMory:SAVE (on page 3-21)

Syntax

```
MEMory:RECall <file_name>
```

Arguments

`<file_name>::=<string>` specifies a setup file to recall

Examples

```
MEMory:RECall "SETUP1.TFS"
```

Recalls a file in file storage named SETUP1.TFS into the current project.

MEMory:SAVE

This command saves the current project file to a specified file in the file system. If the specified file in the file system is locked, this command causes an error. You cannot create a new file if the directory is locked. `<file_name>` is a quoted string that defines the file name and path. There is no query form of this command.

Group

Memory

Related Commands

MEMory:RECall (on page 3-11)

Syntax

```
MEMory:SAVE <file_name>
```

Arguments

`<file_name>::=<string>` specifies a file name in the file system. The <file_name> includes the path. Path separators are forward slashes (/).

Examples

```
MEMory:SAVE "SETUP1.TFS"
```

Copies the current project file to a file named "SETUP1.TFS" in the file system.

MEMory:STATe:DELete

This command deletes the contents of specified setup memory. If a specified setup memory is locked and cannot be overwritten or deleted, this command causes an error. There is no query form of this command.

Group

Memory

Related Commands

None

Syntax

```
MEMory:STATe:DELete {0|1|2|3|4}
```

Arguments

`{0|1|2|3|4}` specifies the location of setup memory.

Examples

```
MEMory:STATe:DELete 1
```

Delete the contents of specified setup memory.

MEMory:STATe:LOCK

This command locks or unlocks the specified setup memory. If you lock a setup memory, you cannot overwrite or delete the setup file. You cannot execute this command for the setup memory of location number 0 (last setup memory). The query form of this command returns the lock state of the setup memory.

Group

Memory

Related Commands

None

Syntax

```
MEMory:STATe:LOCK {1|2|3|4},{ON|OFF|<NR1>}
MEMory:STATe:LOCK?{1|2|3|4}
```

Arguments

ON or <NR1> ≠ 0 locks the specified location of setup memory.

OFF or <NR1> = 0 allows you to overwrite or delete the specified location of setup memory.

Returns

<NR1>

Examples

```
MEMory:STATe:LOCK 1,ON
```

Lock the setup memory of location number 1.

MEMory:STATe:RECall:AUTo

This command enables or disables automatic recall of the last setup memory when powered-on. The next time you apply the power, the arbitrary waveform generator will automatically recall the settings you used when you powered-off the instrument. If you select OFF, the default setups are recalled when you power on the instrument. The query form of this command returns the automatic recall state of the AWG4162.

Group

Memory

Related Commands

None

Syntax

```
MEMory:STATe:RECall:AUTo {ON|OFF|<NR1>}
MEMory:STATe:RECall:AUTo?
```

Arguments

ON or <NR1> ≠ 0 enables the recall of the setup memory you last used before powering-off the instrument.

`OFF` or `<NR1> = 0` disables the last setup recall function.

Returns

`<NR1>`

Examples

`MEMory:STATe:RECall:AUTo ON`

Set the instrument to recall the last setup memory when powered-on.

MEMory:STATe:VALid?

This command returns the availability of a setup memory. This command is query only.

Group

Memory

Related Commands

None

Syntax

`MEMory:STATe:VALid? {0|1|2|3|4}`

Arguments

`{0|1|2|3|4}` specifies the location of setup memory.

Returns

`<NR1>`

1 means that the specified setup memory has been saved.

0 means that the specified setup memory has been deleted.

Examples

`MEMory:STATe:VALid? 0`

Example return if the specified setup memory has been saved: 1.

MMEMory:LOAD:STATe

This command copies a setup file in the file system to an internal setup memory. If a specified internal setup memory is locked, this command causes an error. When you power-off the instrument, the setups are automatically overwritten in the setup memory 0 (last setup memory). There is no query form of this command.

Group

Mass Memory

Related Commands

MEMory:STATe:LOCK (on page 3-13)
MEMory:STATe:RECall:AUTo (on page 3-13)
MMEMory:STORe:STATe (on page 3-15)

Syntax

```
MMEMory:LOAD:STATe{0|1|2|3|4},<file_name>
```

Arguments

`{0|1|2|3|4}` specifies the location of setup memory.

`<file_name>::=<string>` specifies a setup file to be copied.

Examples

```
MMEMory:LOAD:STATe 1,"SETUP1.TFS"
```

Copies a file named SETUP1.TFS in the file system into the internal memory location 1.

MMEMory:STORe:STATe

This command copies a setup file in the setup memory to a specified file in the file system. If the specified file in the file system is locked, this command causes an error. You cannot create a new file if the directory is locked. If the setup memory is deleted, this command causes an error. <file_name> is a quoted string that defines the file name and path. There is no query form of this command.

Group

Mass Memory

Related Commands

MMEMory:LOAD:STATe (on page 3-14)
MMEMory:STATe:LOCK (on page 3-13)

Syntax

```
MMEMory:STORe:STATe{0|1|2|3|4},<file_name>
```

Arguments

`{0|1|2|3|4}` specifies the location of setup memory.

`<file_name>::=<string>` specifies a file name in the file system.
The `<file_name>` includes the path. Path separators are forward slashes (/).

Examples

```
MMEMory:STORe:STATe 1,"SETUP1.TFS"
```

Copies the setup file in the setup memory location 1 to a file named "SETUP1.TFS" in the file system.

*OPC

This command generates the operation complete message by setting bit 0 in the Standard Event Status Register (SESR) when all pending commands that generate an OPC message are complete. The query version of this command places the ASCII character "1" into the output queue when all such *OPC commands are complete.

Group

Synchronization

Related Commands

None

Syntax

```
*OPC
*OPC?
```

Arguments

None

Returns

```
<execution complete>::=1
```

Where: "1" indicates that all pending operations are complete.

Examples

```
*OPC?
```

Example return if all pending OPC operations are finished: 1.

OUTPut[1|2]:IMPedance

This command sets the output load impedance for the specified channel. The specified value is used for amplitude, offset, and high/low level settings. You can set the impedance to any value from 1 Ω to 1 mΩ. The default value is 50 Ω. The query form of this command returns the current load impedance setting in ohms. If the load impedance is set to INFinity, the query command returns "9.9E+37".

Group

Output

Related Commands

None

Syntax

```
OUTPut[1|2]:IMPedance{<ohms>|INFinity|MINimum|MAXimum}
OUTPut[1|2]:IMPedance?[MINimum|MAXimum]
```

Arguments

```
<ohms>::=<NR3>[<units>]
```

Where:

```
<units>::=OHM
```

`INFinity` sets the load impedance to >1 mΩ.

`MINimum` sets the load impedance to 1 Ω.

`MAXimum` sets the load impedance to 1 mΩ.

Returns

```
<ohms>::=<NR3>
```

Examples

```
OUTPut1:IMPedance MAXimum
```

Set the channel 1 (CH 1) load impedance to the maximum value.

OUTPut[1|2]:POLarity

This command inverts a specified output waveform relative to the offset level. The query form of this command returns the polarity for the specified channel.

Group

Output

Related Commands

None

Syntax

```
OUTPut[1|2]:POLarity{NORMal|INVerted}
OUTPut[1|2]:POLarity?
```

Arguments

`NORMal` sets the specified output waveform polarity to Normal.

`INVerted` sets the specified output waveform polarity to Inverted.

Returns

```
NORM|INV
```

Examples

```
OUTPut1:POLarity NORMal
```

Set the channel 1 (CH 1) waveform polarity to Normal.

OUTPut[1|2][:STATe]

This command enables or disables the arbitrary waveform generator output for the specified channel. The query form of this command returns the output state of the AWG4162.

Group

Output

Related Commands

None

Syntax

```
OUTPut[1|2][:STATe] {ON|OFF|<NR1>}
OUTPut[1|2][:STATe]?
```

Arguments

ON or <NR1> ≠ 0 enables the arbitrary waveform generator output.

OFF or <NR1> = 0 disables the arbitrary waveform generator output.

Returns

<NR1>

Examples

```
OUTPut1:STATe ON
```

Set the arbitrary waveform generator channel 1 (CH 1) output to ON.

*RCL

This command restores the state of the instrument from a copy of the settings stored in the setup memory. The settings are stored using the *SAV command. If the specified setup memory is deleted, this command causes an error. There is no query form of this command.

Group

Memory

Related Commands

*SAV (on page 3-19)

Syntax

```
*RCL {0|1|2|3|4}
```

Arguments

{0|1|2|3|4} specifies the location of the setup memory.

Examples

```
*RCL 3
```

Restores the instrument settings from a copy of the settings stored in memory location 3.

*RST

This command resets the instrument to the factory default settings. This command is equivalent to pressing the Default button on the front panel. The default values are listed in Default Settings (on page B-1). There is no query form of this command.

Group

System

Related Commands

None

Syntax

```
*RST
```

Arguments

None

Examples

```
*RST
```

Reset the arbitrary waveform generator settings to the factory defaults.

*SAV

This command stores the current settings of the arbitrary waveform generator to a specified setup memory location. There is no query form of this command.

A setup memory location numbered 0 (last setup memory) is automatically overwritten by the setups when you power off the instrument.

If a specified numbered setup memory is locked, this command causes an error.

Group

Memory

Related Commands

*RCL (on page 3-18)

Syntax

```
*SAV {0|1|2|3|4}
```

Arguments

`{0|1|2|3|4}` specifies the location of setup memory.

Examples

```
*SAV 2
```

Save the current instrument state in the memory location 2.

[SOURce[1|2]]:AM[:DEPTh]

This command sets or queries the AM modulation depth for the specified channel. This command will cause an error if not in the AM modulation state.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:AM:DEPTh{<depth>|MINimum|MAXimum}
[SOURce[1|2]]:AM:DEPTh?[MINimum|MAXimum]
```

Arguments

`<depth>::=<NR2>[<units>]`

Where:

<NR2> is the depth of modulating frequency.

`<units>::=PCT`

`MINimum` sets the modulation depth to the minimum value.

`MAXimum` sets the modulation depth to the maximum value.

Returns

`<depth>`

Examples

`SOURce1:AM:DEPth MAXimum`

Set the depth of the modulating signal on channel (CH 1) to the maximum value.

[SOURce[1|2]]:AM:INTernal:FREQuency

This command sets or queries the internal AM modulation frequency for the specified channel. You can use this command only when the internal modulation source is selected. You can select the source of the modulating signal by using the `[SOURce[1|2]]:AM:SOURce[INTernal|EXTernal]` command. This command will cause an error if the AWG4162 is not in the AM modulation state.

Group

Source

Related Commands

[SOURce[1|2]]:AM:SOURce (on page 3-22)

Syntax

```
[SOURce[1|2]]:AM:INTernal:FREQuency{<frequency>|MINimum|MAXimum}
[SOURce[1|2]]:AM:INTernal:FREQuency?[MINimum|MAXimum]
```

Arguments

`<frequency>::=<NRf>[<units>]`

Where:

`<NRf>` is the modulation frequency.

`<units>::=[Hz|kHz|MHz]`

Returns

`<frequency>`

Examples

```
SOURce1:AM:INTernal:FREQuency 10kHz
```

Set the channel 1 (CH 1) internal modulation frequency to 10 kHz.

[SOURce[1|2]]:AM:INTernal:FUNCtion

This command sets or queries the AM modulating waveform for the specified channel. You can use this command only when the internal modulation source is selected. If you specify EFILe when there is no EFILe or the EFILe is not yet defined, this command causes an error. This command will cause an error if not in the AM modulation state.

Group

Source

Related Commands

[SOURce[1|2]]:AM:SOURce (on page 3-22)

Syntax

```
[SOURce[1|2]]:AM:INTernal:FUNCtion {SINusoid|SQUare|TRIangle
    |RAMP|NRAMp|PRNoise|ARBB1|ARBB2|EFILe}
[SOURce[1|2]]:AM:INTernal:FUNCtion?
```

Arguments

`SINusoid|SQUare|TRIangle|RAMP|NRAMp|PRNoise`

One of six types of function waveform can be selected as a modulating signal.

`ARBB1|ARBB2`

Can be selected as a modulating signal.

`EFILe` is used as a modulating signal.

Returns

`SIN|SQU|TRI|RAMP|NRAM|PRN|ARBB1|ARBB2|EFILe`

Examples

```
SOURce1:AM:INTernal:FUNCtion SQUare
```

Select square as the shape of modulating waveform for the channel 1 (CH 1) output.

[SOURce[1|2]]:AM:INTernal:FUNCtion:EFILe

This command sets or queries an EFILe name used as a modulating waveform for AM modulation. A file name must be specified in the file system. The query version of this command returns "" if there is no file in the file system. This command will cause an error if not in the AM modulation state.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:AM:INTernal:FUNCtion:EFILe <file_name>
[SOURce[1|2]]:AM:INTernal:FUNCtion:EFILe?
```

Arguments

`<file_name>::=<string>` specifies a file name in the file system. The <file_name> includes the path. Path separators are forward slashes (/).

Returns

`<file_name>`

Examples

```
SOURce1:AM:INTernal:FUNCtion:EFILe "SAMPLE1"
```

Sets a file named "SAMPLE1" in the file system.

[SOURce[1|2]]:AM:SOURce

This command sets or queries the source of the AM modulating signal for the specified channel. This command will cause an error if not in the AM modulation state.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:AM:SOURce [INTernal|EXTernal]
[SOURce[1|2]]:AM:SOURce?
```

Arguments

`INTernal` means that the carrier waveform is modulated with an internal source.

`EXTernal` means that the carrier waveform is modulated with an external source.

Returns

```
INT|EXT
```

Examples

```
SOURce1:AM:SOURce INTernal
```

Set the channel 1 (CH 1) source of the modulating signal to internal.

[SOURce[1|2]]:AM:STATe

This command enables or disables AM modulation for the specified channel. The query version of this command returns the state of AM modulation.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:AM:STATe {ON|OFF|<NR1>}
[SOURce[1|2]]:AM:STATe?
```

Arguments

`ON` or `<NR1> ≠ 0` enables AM modulation.

`OFF` or `<NR1> = 0` disables AM modulation.

Returns

```
<NR1>
```

Examples

```
SOURce1:AM:STATe ON
```

Enable the channnel 1 (CH 1) AM modulation.

[SOURce[1|2]]:BURSt:MODE

This command sets or queries the burst mode for the specified channel. This command will cause an error if not in the BURST state.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:BURSt:MODE{TRIGgered|GATed}
[SOURce[1|2]]:BURSt:MODE?
```

Arguments

`TRIGgered` means that triggered mode is selected for the burst mode.

`GATed` means that gated mode is selected for the burst mode.

Returns

`TRIG|GAT`

Examples

```
SOURce1:BURSt:MODE TRIGgered
```

Selects the triggered mode.

[SOURce[1|2]]:BURSt:NCYCles

This command sets or queries the number of cycles (burst count) to be output in burst mode for the specified channel. The query version of this command returns 9.9E+37 if the burst count is set to INFinity.

This command will cause an error if not in BURST state.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:BURSt:NCYCles {<cycles>|INFinity|MINimum|MAXimum}
[SOURce[1|2]]:BURSt:NCYCles? [MINimum|MAXimum]
```

Arguments

`<cycles>::=<NRf>`

Where:

`<NRf>` is the burst count.

The burst count ranges from 1 to 1,000,000.

`INFinity` sets the burst count to infinite count.

`MINimum` sets the burst count to minimum count.

`MAXimum` sets the burst count to maximum count.

Returns

`<cycles>`

Examples

```
SOURce1:BURSt:NCYCles 2
```

Sets the channel 1 (CH 1) burst count to 2.

[SOURce[1|2]]:BURSt[:STATe]

This command enables or disables the burst mode for the specified channel. The query version of this command returns the state of burst mode. This command will cause an error if the instrument is not in the BURST state.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:BURSt:STATe {ON|OFF|<NR1>}
[SOURce[1|2]]:BURSt:STATe?
```

Arguments

ON or <NR1> ≠ 0 enables the burst mode.

OFF or <NR1> = 0 disables the burst mode.

Returns

<NR1>

Examples

```
SOURce1:BURSt:STATe ON
```

Enable the burst mode for channel 1 (CH 1).

[SOURce[1|2]]:BURSt:TDELay

This command sets or queries delay time in the burst mode for the specified channel. It specifies a time delay between the trigger and the signal output. This command is available only in the triggered burst mode. This command will cause an error if not in the BURST state.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:BURSt:TDELay {<delay>|MINimum|MAXimum}
[SOURce[1|2]]:BURSt:TDELay?[MINimum|MAXimum]
```

Arguments

<delay>::=<NRf>[<units>]

Where:

<units>::=[s|ms|μs|ns]

MINimum sets the delay time to minimum value.

`MAXimum` sets the delay time to maximum value.

Returns

`<delay>`

Examples

`SOURce1:BURSt:TDELay 20ms`

Set the channel 1 (CH 1) delay time to 20 ms.

[SOURce[1|2]]:COMBine:FEED

This command sets or queries whether to add the internal noise to an output signal for the specified channel.

When you specify the internal noise, you can set or query the noise level by using the `SOURce<3|4>:POWer[:LEVel][:IMMediate][:AMPLitude]` command.

To disable the internal noise function or the external signal function, specify "".

Group

Source

Related Commands

SOURce<3|4>:POWer[:LEVel][:IMMediate][:AMPLitude] (on page 3-48)

Syntax

```
[SOURce[1]]:COMBine:FEED ["NOISe"|""]
SOURce2:COMBine:FEED ["NOISe"|""]
[SOURce[1|2]]:COMBine:FEED?
```

Arguments

`NOISe` indicates that the internal noise is added to the output signal.

`""` disables the internal noise function and external signal function.

Returns

`"NOIS"| ""`

Examples

`SOURce1:COMBine:FEED "NOISe"`

Add a noise signal to the channel 1 (CH 1) output signal.

[SOURce[1|2]]:FM[:DEViation]

This command sets or queries the peak frequency deviation of FM modulation for the specified channel. The range of the frequency deviation setting depends on the waveform selected as the carrier. This command will cause an error if not in the frequency modulation state.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:FM:DEViation {<deviation>|MINimum|MAXimum}
[SOURce[1|2]]:FM:DEViation?[MINimum|MAXimum]
```

Arguments

```
<deviation>::=<NRf>[<units>]
```

Where:

`<NRf>` is the frequency deviation.

```
<units>::=[Hz|kHz|MHz]
```

Returns

```
<deviation>
```

Examples

```
SOURce1:FM:DEViation 1.0MHz
```

Set the channel 1 (CH 1) frequency deviation to 1.0 MHz.

[SOURce[1|2]]:FM:INTernal:FREQuency

This command sets or queries the internal modulation frequency of FM modulation for the specified channel. You can use this command only when the internal modulation source is selected. This command will cause an error if not in frequency modulation state.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:FM:INTernal:FREQuency {<frequency>|MINimum|MAXimum}
[SOURce[1|2]]:FM:INTernal:FREQuency?[MINimum|MAXimum]
```

Arguments

`<frequency>::=<NRf>[<units>]`

Where:

`<NRf>` is the modulation frequency.

`<units>::=[Hz | kHz | MHz]`

Returns

`<frequency>`

Examples

`SOURce1:FM:INTernal:FREQuency 10kHz`

Set the channel 1 (CH 1) internal modulation frequency to 10 kHz.

[SOURce[1|2]]:FM:INTernal:FUNCtion

This command sets or queries the FM modulating waveform for the specified channel. You can use this command only when the internal modulation source is selected. If you specify EFILe when there is no EFILe or the EFILe is not yet defined, this command causes an error. This command will cause an error if not in the frequency modulation state.

Group

Source

Related Commands

[SOURce[1|2]]:FM:SOURce (on page 3-29)

Syntax

```
[SOURce[1|2]]:FM:INTernal:FUNCtion{SINusoid|SQUare|
TRIangle|RAMP|NRAMp|PRNoise|ARBB|ARBB1|ARBB2|EFILe}
[SOURce[1|2]]:FM:INTernal:FUNCtion?
```

Arguments

`SINusoid|SQUare|TRIangle|RAMP|NRAMp|PRNoise`

One of six types of function waveform can be selected as a modulating signal.

`ARBB|ARBB1|ARBB2`

Can be selected as a modulating signal

`EFILe` is used as a modulating signal.

Returns

`SIN|SQU|TRI|RAMP|NRAM|PRN|ARBB1|ARBB2|EFILe`

Examples

`SOURce1:FM:INTernal:FUNCtion SQUare`

Select square as the shape of modulating waveform for the channel 1 (CH 1) output.

[SOURce[1|2]]:FM:INTernal:FUNCtion:EFILe

This command sets or queries an EFILe name used as a modulating waveform for FM modulation. A file name must be specified in the file system. This command returns "" if there is no file in the file system. This command will cause an error if not in frequency modulation state.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:FM:INTernal:FUNCtion:EFILe <file_name>
[SOURce[1|2]]:FM:INTernal:FUNCtion:EFILe?
```

Arguments

`<file_name>::=<string>` specifies a file name in the file system.
The `<file_name>` parameter includes the path. Path separators are forward slashes (/).

Returns

`<file_name>`

Examples

```
SOURce1:FM:INTernal:FUNCtion:EFILe "SAMPLE1"
```

Sets a file named "SAMPLE1" in the file system.

[SOURce[1|2]]:FM:SOURce

This command sets or queries the source of the FM modulating signal for the specified channel. This command will cause an error if not in the frequency modulation state.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:FM:SOURce [INTernal|EXTernal]
[SOURce[1|2]]:FM:SOURce?
```

Arguments

`INTernal` means that the carrier waveform is modulated with the internal source.

`EXTernal` means that the carrier waveform is modulated with an external source.

Returns

```
INT|EXT
```

Examples

```
SOURce1:FM:SOURce INTernal
```

Set the channel 1 (CH 1) source of the modulating signal to internal.

[SOURce[1|2]]:FM:STATe

This command enables or disables frequency modulation (FM). The query version of this command returns the state of frequency modulation.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:FM:STATe {ON|OFF|<NR1>}
[SOURce[1|2]]:FM:STATe?
```

Arguments

ON or <NR1> ≠ 0 enables FM modulation.

OFF or <NR1> = 0 disables FM modulation.

Returns

```
<NR1>
```

Examples

```
SOURce1:FM:STATe ON
```

Enable the channel 1 (CH 1) FM modulation.

[SOURce[1|2]]:FREQuency:CENTer

This command sets or queries the center frequency of a sweep for the specified channel. This command is always used with the [SOURce[1|2]]:FREQuency:SPAN command. The center frequency range setting depends on the waveform selected for the sweep. This command will cause an error if not in the SWEEP state.

Group

Source

Related Commands

[SOURce[1|2]]:FREQuency:SPAN (on page 3-33)
[SOURce[1|2]]:FREQuency:MODE (on page 3-33)

Syntax

```
[SOURce[1|2]]:FREQuency:CENTer {<frequency>|MINimum|MAXimum}
[SOURce[1|2]]:FREQuency:CENTer?
```

Arguments

`<frequency>::=<NRf>[<units>]`

Where:

`<NRf>` is the center frequency.

`<units>::=[Hz | kHz | MHz]`

Returns

`<frequency>`

Examples

`SOURce1:FREQuency:CENTer 550kHz`

Set the channel 1 (CH 1) center frequency to 550 kHz.

[SOURce[1|2]]:FREQuency:CONCurrent[:STATe]

This command enables or disables the function to copy the frequency (or period) of one channel to another channel.

The `[SOURce[1|2]]:FREQuency:CONCurrent` command copies the frequency (or period) of the channel specified by the header suffix to another channel. If you specify CH 1 with the header, the channel 1 (CH 1) frequency will be copied to channel 2 (CH 2).

The query version of this command returns "0" (off) or "1" (on).

If your arbitrary waveform generator is single-channel model, this command is not supported.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:FREQuency:CONCurrent {ON|OFF|<NR1>}
[SOURce[1|2]]:FREQuency:CONCurrent?
```

Arguments

`ON` or `<NR1> ≠ 0` enables the concurrent copy function.

`OFF` or `<NR1>=0` disables the concurrent copy function.

Returns

<NR1>

Examples

```
SOURce1:FREQuency:CONCurrent ON
```

Copy the frequency value of channel 1 (CH 1) to channel 2 (CH 2).

[SOURce[1|2]]:FREQuency[:CW|:FIXed]

This command sets or queries the frequency of the output waveform for the specified channel. This command is available when the Run Mode is set to any setting other than Sweep.

The output frequency range setting depends on the type of output waveform. If you change the type of output waveform, it may change the output frequency because changing waveform types affects the setting range of the output frequency.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:FREQuency[:CW|:FIXed] {<frequency>|MINimum|MAXimum}
[SOURce[1|2]]:FREQuency[:CW|:FIXed]?[MINimum|MAXimum]
```

Arguments

```
<frequency>::=<NRf>[<units>]
```

Where:

`<NRf>` is the output frequency.

```
<units>::=[Hz|kHz|MHz]
```

Returns

```
<frequency>
```

Examples

```
SOURce1:FREQuency:FIXed 500kHz
```

Set the channel 1 (CH 1) output frequency to 500 kHz when the Run Mode is set to any setting other than Sweep.

[SOURce[1|2]]:FREQuency:MODE

This command sets or queries the frequency sweep state. You can select sine, square, ramp, or arbitrary waveform for sweep. The arbitrary waveform generator automatically changes to the continuous mode if any waveform is selected other than sine, square, ramp, or an arbitrary waveform. This command will cause an error if not in the Sweep state.

Group

Source

Related Commands

[SOURce[1|2]]:FREQuency[:CW|:FIXed] (on page 3-32)
[SOURce[1|2]]:FREQuency:CENTer (on page 3-30)
[SOURce[1|2]]:FREQuency:SPAN (on page 3-33)
[SOURce[1|2]]:FREQuency:STARt (on page 3-34)
[SOURce[1|2]]:FREQuency:STOP (on page 3-35)

Syntax

```
[SOURce[1|2]]:FREQuency:MODE {CW|FIXed|SWEep}
[SOURce[1|2]]:FREQuency:MODE?
```

Arguments

`CW|FIXed` means that the frequency is controlled by the `[SOURce[1|2]]:FREQuency[:CW|:FIXed]` command. The sweep is invalid.

`SWEep` means that the output frequency is controlled by the sweep command set. The sweep is valid.

Returns

`CW|FIXed|SWEep`

Examples

```
SOURce1:FREQuency:MODE SWEep
```

Specify the sweep command set for controlling the CH 1 output frequency.

[SOURce[1|2]]:FREQuency:SPAN

This command sets or queries the span of the frequency sweep for the specified channel. This command is always used with the `[SOURce[1|2]]:FREQuency:CENTer` command. The frequency span range setting depends on the waveform selected for the sweep. This command will cause an error if not in Sweep state.

Group

Source

Related Commands

[SOURce[1|2]]:FREQuency:CENTer (on page 3-30)
[SOURce[1|2]]:FREQuency:MODE (on page 3-33)

Syntax

```
[SOURce[1|2]]:FREQuency:SPAN {<frequency>|MINimum|MAXimum}
[SOURce[1|2]]:FREQuency:SPAN? [MINimum|MAXimum]
```

Arguments

`<frequency>::=<NRf>[<units>]`

Where: `<NRf>` is the frequency span.

`<units>::=[Hz | kHz | MHz]`

Returns

`<frequency>`

Examples

`SOURce1:FREQuency:SPAN 90kHz`

Set the channel 1 (CH 1) frequency span to 90 kHz.

[SOURce[1|2]]:FREQuency:STARt

This command sets or queries the start frequency of a sweep for the specified channel. This command is always used with the `[SOURce[1|2]]:FREQuency:STOP` command. The start frequency range setting depends on the waveform selected for sweep. This command will cause an error if not in the Sweep state.

Group

Source

Related Commands

[SOURce[1|2]]:FREQuency:MODE (on page 3-33)
[SOURce[1|2]]:FREQuency:STOP (on page 3-35)

Syntax

```
[SOURce[1|2]]:FREQuency:STARt {<frequency>|MINimum|MAXimum}
[SOURce[1|2]]:FREQuency:STARt? [MINimum|MAXimum]
```

Arguments

`<frequency>::=<NRf>[<units>]`

Where: `<NRf>` is the start frequency.

`<units>::=[Hz | kHz | MHz]`

Returns

`<frequency>`

Examples

`SOURce1:FREQuency:STARt 10kHz`

Set the sweep start frequency of channel 1 (CH 1) to 10 kHz.

[SOURce[1|2]]:FREQuency:STOP

This command sets or queries the start frequency of sweep for the specified channel. This command is always used with the [SOURce[1|2]]:FREQuency:STARt command. The stop frequency range setting depends on the waveform selected for sweep. This command will cause an error if not in the Sweep state.

Group

Source

Related Commands

[SOURce[1|2]]:FREQuency:MODE (on page 3-33)
[SOURce[1|2]]:FREQuency:STARt (on page 3-34)

Syntax

```
[SOURce[1|2]]:FREQuency:STOP {<frequency>|MINimum|MAXimum}
[SOURce[1|2]]:FREQuency:STOP? [MINimum|MAXimum]
```

Arguments

```
<frequency>::=<NRf>[<units>]
```

Where: <NRf> is the stop frequency.

```
<units>::=[Hz | kHz | MHz]
```

Returns

<frequency>

Examples

```
SOURce1:FREQuency:STOP 100KHz
```

Set the stop frequency of channel 1 (CH 1) to 100 kHz.

[SOURce[1|2]]:FSKey[:FREQuency]

This command sets or queries the hop frequency of FSK modulation for the specified channel. This command will cause an error if not in the FSKEY state.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:FSKey[:FREQuency] {<frequency>|MINimum|MAXimum}
[SOURce[1|2]]:FSKey[:FREQuency]? [MINimum|MAXimum]
```

Arguments

```
<frequency>::=<NRf>[<units>]
```

Where: <NRf> is the hop frequency.

```
<units>::=[Hz | kHz | MHz]
```

Returns

`<frequency>`

Examples

`SOURce1:FSKey:FREQuency 1.0MHz`

Set the hop frequency of channel 1 (CH 1) FSK modulation to 1.0 MHz.

[SOURce[1|2]]:FSKey:INTernal:RATE

This command sets or queries the internal modulation rate of FSK modulation for the specified channel. You can use this command only when the internal modulation source is selected. This command will cause an error if not in the FSKEY state.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:FSKey:INTernal:RATE {<rate>|MINimum|MAXimum}
[SOURce[1|2]]:FSKey:INTernal:RATE?[MINimum|MAXimum]
```

Arguments

`<rate>::=<NRf>[<units>]`

Where: `<NRf>` is the modulation rate.

`<units>::=[Hz | kHz | MHz]`

Returns

`<rate>`

Examples

`SOURce1:FSKey:INTernal:RATE 50Hz`

Set the channel 1 (CH 1) internal modulation rate to 50 Hz.

[SOURce[1|2]]:FSKey:SOURce

This command sets or queries the source of the FSK modulation signal for the specified channel. This command will cause an error if not in the FSKEY state.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:FSKey:SOURce [INTernal|EXTernal]
[SOURce[1|2]]:FSKey:SOURce?
```

Arguments

`INTernal` means that the carrier waveform is modulated with an internal source.

`EXTernal` means that the carrier waveform is modulated with an external source.

Returns

`INT|EXT`

Examples

`SOURce1:FSKey:SOURce INTernal`

Set the channel 1 (CH 1) modulating signal source to internal.

[SOURce[1|2]]:FSKey:STATe

This command enables or disables FSK modulation. The query form of this command returns the state of FSK modulation. You can select a sine, square, ramp, or arbitrary waveform for the carrier waveform.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:FSKey:STATe {ON|OFF|<NR1>}
[SOURce[1|2]]:FSKey:STATe?
```

Arguments

`ON` or `<NR1> ≠ 0` enables FSK modulation.

`OFF` or `<NR1> = 0` disables FSK modulation.

Returns

`<NR1>`

Examples

`SOURce1:FSKey:STATe ON`

Enable the channel 1 (CH 1) FSK modulation.

[SOURce[1|2]]:PSKey[:FREQuency]

This command sets or queries the frequency of PSK modulation for the specified channel. This command will cause an error if not in the PSKEY state.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:PSKey[:FREQuency] {<frequency>|MINimum|MAXimum}
[SOURce[1|2]]:PSKey[:FREQuency]? [MINimum|MAXimum]
```

Arguments

```
<frequency>::=<NRf>[<units>]
```

Where: `<NRf>` is the hop frequency.

```
<units>::=[Hz | kHz | MHz]
```

Returns

```
<frequency>
```

Examples

```
SOURce1:PSKey:FREQuency 1.0MHz
```

Set the hop frequency of channel 1 (CH 1) PSK modulation to 1.0 MHz.

[SOURce[1|2]]:PSKey:PHASe[:ADJust]

This command sets or queries the phase of the modulating signal of PSKEY modulation for the specified channel. You can set the value in radians or degrees. If no units are specified, the default is RAD. The query form of this command returns the value in RAD. This command is supported when you select a PSK modulating waveform. This command will cause an error if not in PSKEY state.

Group

Source

Related Commands

None

Syntax

[SOURce[1|2]]:PSK:PHASe[:ADJust] {<phase>|MINimum|MAXimum}

[SOURce[1|2]]:PSK:PHASe[:ADJust]? [MINimum|MAXimum]

Arguments

`<phase>::=<NR3>[<units>]`

Where: `<NR3>` is the phase of modulating signal.

`<units>::=[RAD | DEG]`

If `<units>` are omitted, RAD is specified automatically. The setting ranges are:

RAD –1 PI to +1 PI, relative to phase value

DEG –180 to +180, relative to phase value

Returns

`<phase>`

Examples

```
SOURce1:PSK:PHASe:ADJust MAXimum
```

Set the maximum value for the phase of the channel 1 (CH 1) PSK modulating waveform.

[SOURce[1|2]]:PSKey:SOURce

This command sets or queries the source of the PSK modulation signal for the specified channel. This command will cause an error if not in the PSKEY state.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:PSKey:SOURce [INTernal|EXTernal]
[SOURce[1|2]]:PSKey:SOURce?
```

Arguments

`INTernal` means that the carrier waveform is modulated with an internal source.

`EXTernal` means that the carrier waveform is modulated with an external source.

Returns

`INT|EXT`

Examples

```
SOURce1:PSKey:SOURce INTernal
```

Set the channel 1 (CH 1) source of the modulating signal to internal.

[SOURce[1|2]]:PSKey:STATe

This command enables or disables PSK modulation. The query version of this command returns the state of PSK modulation.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:PSKey:STATe {ON|OFF|<NR1>}
[SOURce[1|2]]:PSKey:STATe?
```

Arguments

ON or <NR1> ≠ 0 enables PSK modulation.

OFF or <NR1> = 0 disables PSK modulation.

Returns

<NR1>

Examples

```
SOURce1:PSKey:STATe ON
```

Enable the channel 1 (CH 1) PSK modulation.

[SOURce[1|2]]:FUNCtion:EFILe

This command sets or queries an EFILe name used as an output waveform. A file name must be specified in the file system. This command returns "" if there is no file in the file system.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:FUNCtion:EFILe <file_name>
[SOURce[1|2]]:FUNCtion:EFILe?
```

Arguments

<file_name>::=<string> specifies a file name in the file system.
The <file_name> parameter includes the path. Path separators are forward slashes (/).

Returns

```
<file_name>
```

Examples

```
SOURce1:FUNCtion:EFILe "SAMPLE1"
```

Sets a file named "SAMPLE1" in the file system.

[SOURce[1|2]]:FUNCtion:RAMP:SYMMetry

This command sets or queries the symmetry of the ramp waveform for the specified channel.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:FUNCtion:RAMP:SYMMetry {<symmetry>|MINimum|MAXimum}
[SOURce[1|2]]:FUNCtion:RAMP:SYMMetry? [MINimum|MAXimum]
```

Arguments

```
<symmetry>::=<NR2>[<units>]
```

Where: <NR2> is the symmetry.

```
<units>::=PCT
```

Returns

```
<symmetry>
```

Examples

```
SOURce1:FUNCtion:RAMP:SYMMetry 80.5
```

Set the symmetry of the channel 1 (CH 1) ramp waveform to 80.5%.

[SOURce[1|2]]:FUNCtion[:SHAPe]

This command sets or queries the shape of the output waveform.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:FUNCtion[:SHAPe] {SINusoid|SQUare|PULSe|RAMP
|PRNoise|DC|SINC|GAUSsian|LORentz|ERISe|EDECay|HAVersine|ARBB1|ARBB2|
   EFILe}
[SOURce[1|2]]:FUNCtion[:SHAPe]?
```

Arguments

`SINusoid|SQUare|PULSe|RAMP|PRNoise|DC|SINC|GAUSsian|LORentz|ERISe|EDECay|HAVersine`

The following table shows the combinations of modulation type and the shape of output waveform. If you select a waveform shape that is not allowed with a particular modulation, sweep, or burst, the Run mode will automatically be changed to Continuous.

If you specify EFILe when there is no EFILe or the EFILe is not yet defined, this command causes an error.

If you change the type of output waveform, the output frequency may change because changing waveform types affects the output frequency range setting.

`ARBB1|ARBB2|ARBB`

Can be selected as an output waveform.

	Sine, Square, Ramp, Arb, Sin(x)/x, Gaussian, Lorentz, Exponential Rise, Exponential Decay, Haversine	**Pulse**	**Noise, DC**
AM	√		
FM	√		
PM	√		
FSK	√		
PSK	√		
PWM		√	
Sweep	√		
Burst	√	√	

`EFILe` is specified as an output waveform.

Returns

`SIN|SQU|PULS|RAMP|PRN|DC|SINC|GAUS|LOR|ERIS|EDEC|HARV|ARBB1|ARBB2|EFILe`

Examples

`SOURce1:FUNCtion:SHAPe SQUare`

Select the shape of channel 1 (CH 1) output waveform to square waveform.

[SOURce[1|2]]:PHASe[:ADJust]

This command sets or queries the phase of the output waveform for the specified channel. You can set the value in radians or degrees. If no units are specified, the default is RAD. The query form of this command returns the value in RAD.

This command is supported when you select a waveform other than DC, Noise, and Pulse.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:PHASe[:ADJust] {<phase>|MINimum|MAXimum}
[SOURce[1|2]]:PHASe[:ADJust]? [MINimum|MAXimum]
```

Arguments

`<phase>::=<NR3>[<units>]`

Where: <NR3> is the phase of output frequency.

`<units>::=[RAD | DEG]`

If the `<units>` parameter is omitted, RAD is specified automatically. The setting ranges are:

RAD –1 PI to +1 PI, relative to phase value

DEG –180 to +180, relative to phase value

Returns

`<phase>`

Examples

`SOURce1:PHASe:ADJust MAXimum`

Set the maximum value for the phase of channel 1 (CH 1) output frequency.

[SOURce[1|2]]:PHASe:INITiate

This command synchronizes the phase of channel 1 (CH 1) and channel 2 (CH 2) output waveforms. The arbitrary waveform generator performs the same operation if you specify either SOURce1 or SOURce2. There is no query form of this command.

If your arbitrary waveform generator is single-channel model, this command is not supported. This command will cause an error if not in continuous mode.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:PHASe:INITiate
```

Arguments

None

Examples

```
SOURce1:PHASe:INITiate
```

Synchronize the phase of CH 1 and CH 2 output signals.

[SOURce[1|2]]:PM[:DEViation]

This command sets or queries the peak frequency deviation of PM modulation for the specified channel. This command will cause an error if not in PM mode.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:PM:DEViation {<deviation>|MINimum|MAXimum}
[SOURce[1|2]]:PM:DEViation? [MINimum|MAXimum]
```

Arguments

```
<deviation>::=<NR3>[<units>]
```

Where: <NR3> is the phase deviation.

```
<units>::=[RAD | DEG]
```

If <units> are omitted, RAD is specified automatically. The setting ranges are:

RAD –1 PI to +1 PI, relative to phase value

DEG –180 to +180, in one-degree steps, relative to phase value

Returns

```
<deviation>
```

Examples

```
SOURce1:PM:DEViation MAXimum
```

Set the maximum value for the channel 1 (CH 1) phase deviation.

[SOURce[1|2]]:PM:INTernal:FREQuency

This command sets or queries the internal modulation frequency of PM modulation for the specified channel. You can use this command only when the internal modulation source is selected. This command will cause an error if not in PM mode.

Group

Source

Related Commands

[SOURce[1|2]]:PM:SOURce (on page 3-47)

Syntax

```
[SOURce[1|2]]:PM:INTernal:FREQuency {<frequency>|MINimum|MAXimum}
[SOURce[1|2]]:PM:INTernal:FREQuency?[MINimum|MAXimum]
```

Arguments

`<frequency>::=<NRf>[<units>]`

Where: `<NRf>` is the modulation frequency.

`<units>::=[Hz|kHz|MHz]`

Returns

`<frequency>`

Examples

`SOURce1:PM:INTernal:FREQuency 10kHz`

Set the channel 1 (CH 1) internal modulation frequency to 10 kHz.

[SOURce[1|2]]:PM:INTernal:FUNCtion

This command sets or queries the modulating waveform of PM modulation for the specified channel. You can use this command only when the internal modulation source is selected.

If you specify EFILe when there is no EFILe or the EFILe is not yet defined, this command causes an error. This command will cause an error if not in PM mode.

Group

Source

Related Commands

[SOURce[1|2]]:PM:SOURce (on page 3-47)

Syntax

```
[SOURce[1|2]]:PM:INTernal:FUNCtion {SINusoid|SQUare|TRIangle|
   RAMP|NRAMp|PRNoise|USER[1]|ARBB1|ARBB2|EFILe}
[SOURce[1|2]]:PM:INTernal:FUNCtion?
```

Arguments

`SINusoid|SQUare|TRIangle|RAMP|NRAMp|PRNoise`

One of six types of function waveform can be selected as a modulating signal.

`ARBB1|ARBB2`

Can be selected as a modulating signal.

`EFILe`

EFILe is used as a modulating signal.

Returns

`SIN|SQU|TRI|RAMP|NRAM|PRN|ARBB1|ARBB2|EFILe`

Examples

```
SOURce1:PM:INTernal:FUNCtion SQUare
```

Select square as the shape of modulating waveform for the channel 1 (CH 1) output.

[SOURce[1|2]]:PM:INTernal:FUNCtion:EFILe

This command sets or queries an EFILe name used as a modulating waveform for PM modulation. A file name must be specified in the file system. This command returns "" if there is no file in the file system. This command will cause an error if not in PM mode.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:PM:INTernal:FUNCtion:EFILe <file_name>
[SOURce[1|2]]:PM:INTernal:FUNCtion:EFILe?
```

Arguments

`<file_name>::=<string>` specifies a file name in the file system.
The `<file_name>` parameter includes the path. Path separators are forward slashes (/).

Returns

`<file_name>`

Examples

```
SOURce1:PM:INTernal:FUNCtion:EFILe "SAMPLE1"
```

Sets up a file named "SAMPLE1" in the file system.

[SOURce[1|2]]:PM:SOURce

This command sets or queries the source of the PM modulation signal for the specified channel. This command will cause an error if not in PM mode.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:PM:SOURce [INTernal|EXTernal]
[SOURce[1|2]]:PM:SOURce?
```

Arguments

`INTernal` means that the carrier waveform is modulated with an internal source.

`EXTernal` means that the carrier waveform is modulated with an external source.

Returns

`INT|EXT`

Examples

```
SOURce1:PM:SOURce INTernal
```

Set the channel 1 (CH 1) source of the modulating signal to internal.

[SOURce[1|2]]:PM:STATe

This command enables or disables PM modulation. The query version of this command returns the state of PM modulation.You can select a sine, square, ramp, or arbitrary waveform for the carrier waveform.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:PM:STATe {ON|OFF|<NR1>}
[SOURce[1|2]]:PM:STATe?
```

Arguments

`ON` or `<NR1> ≠ 0` enables PM modulation.

`OFF` or `<NR1> = 0` disables PM modulation.

Returns

```
<NR1>
```

Examples

```
SOURce1:PM:STATe ON
```

Enable the channel 1 (CH 1) PM modulation.

SOURce<3|4>:POWer[:LEVel][:IMMediate][:AMPLitude]

This command sets or queries the internal noise level that applies to the output signal for the specified channel. The noise level represents the percent against current amplitude level. The setting range is 0 to 50%.

This command is available when Run Mode is set to Continuous, Burst, or Sweep.

You can set or query whether to add the internal noise to the output signal using the `[SOURce[1|2]]:COMBine:FEED` command.

Group

Source

Related Commands

[SOURce[1|2]]:COMBine:FEED (on page 3-26)

Syntax

```
SOURce<3|4>:POWer[:LEVel][:IMMediate][:AMPLitude]
   {<percent>|MINimum|MAXimum}
SOURce<3|4>:POWer[:LEVel][:IMMediate][:AMPLitude]?[MINimum|MAXimum]
```

Arguments

```
<percent>::=<NR2>[<units>]
```

Where: <NR2> is the noise level.

```
<units>::=PCT
```

Returns

```
<percent>
```

Examples

```
SOURce3:POWer:LEVel:IMMediate:AMPLitude 50PCT
```

Set the internal noise level that is added to the output signal to 50%.

[SOURce[1|2]]:PULSe:DCYCle

This command sets or queries the duty cycle of the pulse waveform for the specified channel. The setting range is 0.1% to 99.9% in increments of 0.1%.

The arbitrary waveform generator will hold the settings of leading edge and trailing edge when the duty cycle is varied. Refer to the [SOURce[1|2]]:PULSe:WIDTh (on page 3-52) command for the setting range.

Group

Source

Related Commands

[SOURce[1|2]]:PULSe:WIDTh (on page 3-52)

Syntax

```
[SOURce[1|2]]:PULSe:DCYCle {<percent>|MINimum|MAXimum}
[SOURce[1|2]]:PULSe:DCYCle? [MINimum|MAXimum]
```

Arguments

`<percent>::=<NR2>[<units>]`

Where: `<NRf>` is the duty cycle.

`<units>::=PCT`

Returns

`<percent>`

Examples

`SOURce1:PULSe:DCYCle 80.5`

Set the duty cycle of the pulse waveform on channel 1 (CH 1) to 80.5%.

[SOURce[1|2]]:PULSe:DELay

This command sets or queries the lead delay of the pulse waveform for the specified channel.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:PULSe:DELay {<delay>|MINimum|MAXimum}
[SOURce[1|2]]:PULSe:DELay?[MINimum|MAXimum]
```

Arguments

`<delay>::=<NR2>[<units>]`

Where: <NR2> is the lead delay.

`<units>::=[ns|μs|ms|s]`

Setting range: 0 ns to Pulse Period – {Pulse Width + 0.8 × (Leading Edge Time + Trailing Edge Time)}

Returns

```
<delay>
```

Examples

```
SOURce1:PULSe:DELay 20ms
```

Set the channel 1 (CH 1) lead delay to 20 ms.

[SOURce[1|2]]:PULSe:HOLD

This command sets the arbitrary waveform generator to hold either pulse width or pulse duty. The query form of this command returns WIDTh or DUTY.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:PULSe:HOLD {WIDTh|DUTY}
[SOURce[1|2]]:PULSe:HOLD?
```

Arguments

`WIDTh` means that the arbitrary waveform generator holds the pulse width setting.

DUTY means that the arbitrary waveform generator holds the pulse duty setting.

Returns

```
WIDT|DUTY
```

Examples

```
SOURce1:PULSe:HOLD WIDTh
```

Hold the channel 1 (CH 1) pulse width setting.

[SOURce[1|2]]:PULSe:PERiod

This command sets or queries the period for the pulse waveform.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:PULSe:PERiod {<period>|MINimum|MAXimum}
[SOURce[1|2]]:PULSe:PERiod?[MINimum|MAXimum]
```

Arguments

`<period>::=<NRf>[<units>]`

Where: `<NRf>` is the pulse period.

`<units>::=[ns|µs|ms|s]`

Returns

`<period>`

Examples

`SOURce1:PULSe:PERiod 200ns`

Set the channel 1 (CH 1) pulse period to 200 ns.

[SOURce[1|2]]:PULSe:TRANsition[:LEADing]

This command sets or queries the leading edge time of the pulse waveform.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:PULSe:TRANsition[:LEADing] {<seconds>|MINimum|MAXimum}
[SOURce[1|2]]:PULSe:TRANsition[:LEADing]?[MINimum|MAXimum]
```

Arguments

`<seconds>::=<NRf>[<units>]`

Where: `<NRf>` is the leading edge time of pulse waveform.

`<units>::=[ns|µs|ms|s]`

Returns

`<seconds>`

Examples

```
SOURce1:PULSe:TRANsition:LEADing 200ns
```

Set the channel 1 (CH 1) leading edge time to 200 ns.

[SOURce[1|2]]:PULSe:TRANsition:TRAiling

This command sets or queries the trailing edge time of the pulse waveform.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:PULSe:TRANsition:TRAiling {<seconds>|MINimum|MAXimum}
[SOURce[1|2]]:PULSe:TRANsition:TRAiling?[MINimum|MAXimum]
```

Arguments

`<seconds>::=<NRf>[<units>]`

Where: `<NRf>` is the trailing edge of pulse waveform.

`<units>::=[ns|μs|ms|s]`

Returns

`<seconds>`

Examples

`SOURce1:PULSe:TRANsition:TRAiling 200ns`

Set the trailing edge time to 200 ns.

[SOURce[1|2]]:PULSe:WIDTh

This command sets or queries the pulse width for the specified channel. Pulse Width = Period × Duty Cycle / 100. The pulse width must be less than the period. The setting range is 0.1% to 99.9% in terms of duty cycle.
Pulse Width ≤ Pulse Period – 0.8 × (Leading Edge Time + Trailing Edge Time)
Pulse Width = 0.625 × (Leading Edge Time + Trailing Edge Time)

Group

Source

Related Commands

[SOURce[1|2]]:PULSe:DCYCle (on page 3-49)

Syntax

```
[SOURce[1|2]]:PULSe:WIDTh {<seconds>|MINimum|MAXimum}
[SOURce[1|2]]:PULSe:WIDTh?[MINimum|MAXimum]
```

Arguments

`<seconds>::=<NRf>[<units>]`

Where: `<NRf>` is the pulse width.

`<units>::=[ns|μs|ms|s]`

Returns

`<seconds>`

Examples

```
SOURce1:PULSe:WIDTh 200ns
```

Set the channel 1 (CH 1) pulse width to 200 ns.

[SOURce[1|2]]:PWM:INTernal:FREQuency

This command sets or queries the internal modulation frequency of PWM modulation for the specified channel. You can use this command only when the internal modulation source is selected. This command will cause an error if not in PWM mode.

Group

Source

Related Commands

[SOURce[1|2]]:PWM:SOURce (on page 3-55)

Syntax

```
[SOURce[1|2]]:PWM:INTernal:FREQuency{<frequency>|MINimum|MAXimum}
[SOURce[1|2]]:PWM:INTernal:FREQuency?[MINimum|MAXimum]
```

Arguments

`<frequency>::=<NRf>[<units>]`

Where: `<NRf>` is the modulation frequency.

`<units>::=[Hz | kHz | MHz]`

Returns

`<frequency>`

Examples

```
SOURce1:PWM:INTernal:FREQuency 10kHz
```

Set the channel 1 (CH 1) internal frequency to 10 kHz.

[SOURce[1|2]]:PWM:INTernal:FUNCtion

This command sets or queries the modulating waveform of PWM modulation for the specified channel. You can use this command only when the internal modulation source is selected.

If you specify EFILe when there is no EFILe or the EFILe is not yet defined, this command causes an error. This command will cause an error if not in PWM mode.

Group

Source

Related Commands

[SOURce[1|2]]:PWM:SOURce (on page 3-55)

Syntax

```
[SOURce[1|2]]:PWM:INTernal:FUNCtion{SINusoid|SQUare|TRIangle|
   RAMP|NRAMp|PRNoise|ARBB1|ARBB2|EFILe}
[SOURce[1|2]]:PWM:INTernal:FUNCtion?
```

Arguments

`SINusoid|SQUare|TRIangle|RAMP|NRAMp|PRNoise`

One of six types of function waveform can be selected as a modulating signal.

`ARBB|ARBB1|ARBB2`

Can be selected as a modulating signal.

`EFILe`

`EFILe` is used as a modulating signal.

Returns

`SIN|SQU|TRI|RAMP|NRAM|PRN|ARBB1|ARBB2|EFILe`

Examples

`SOURce1:PWM:INTernal:FUNCtion SQUare`

Select square as the shape of modulating waveform for the channel 1 (CH 1) output.

[SOURce[1|2]]:PWM:INTernal:FUNCtion:EFILe

This command sets or queries an EFILe name used as a modulating waveform for PWM modulation. A file name must be specified in the file system. This command returns "" if there is no file in the file system. This command will cause an error if not in PWM mode.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:PWM:INTernal:FUNCtion:EFILe <file_name>
[SOURce[1|2]]:PWM:INTernal:FUNCtion:EFILe?
```

Arguments

`<file_name>::=<string>` specifies a file name in the file system.
The `<file_name>` parameter includes the path. Path separators are forward slashes (/).

Returns

`<file_name>`

Examples

```
SOURce1:PWM:INTernal:FUNCtion:EFILe "SAMPLE1"
```

A file named "SAMPLE1" should exist and be set as the EFILe name.

[SOURce[1|2]]:PWM:SOURce

This command sets or queries the source of the modulating signal of PWM modulation for the specified channel. This command will cause an error if not in PWM mode.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:PWM:SOURce [INTernal|EXTernal]
[SOURce[1|2]]:PWM:SOURce?
```

Arguments

`INTernal` means that the carrier waveform is modulated with the internal source.

`EXTernal` means that the carrier waveform is modulated with an external source.

Returns

`INT|EXT`

Examples

```
SOURce1:PWM:SOURce INTernal
```

Set the source of the modulating signal on channel 1 (CH 1) to internal.

[SOURce[1|2]]:PWM:STATe

This command enables or disables PWM modulation. The query form of this command returns the state of PWM modulation. You can select only the pulse waveform as a carrier waveform for PWM.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:PWM:STATe {ON|OFF|<NR1>}
[SOURce[1|2]]:PWM:STATe?
```

Arguments

`ON` or `<NR1> ≠ 0` enables PWM modulation.

`OFF` or `<NR1> = 0` disables PWM modulation.

Returns

`<NR1>`

Examples

```
SOURce1:PWM:STATe ON
```

Enable the channel 1 (CH 1) PWM modulation.

[SOURce[1|2]]:PWM[:DEViation]:DCYCle

This command sets or queries the PWM deviation in percent for the specified channel. The setting range must meet the following conditions:

- Deviation ≤ Pulse Width – PWmin
- Deviation ≤ Pulse Period – Pulse Width – PWmin
- Deviation ≤ Pulse Width – 0.8 × (Leading Edge Time + Trailing Edge Time)
- Deviation ≤ Pulse Period – Pulse Width – 0.8 × (Leading Edge Time + Trailing Edge Time)
- Where: PWmin is the minimum pulse width.

This command will cause an error if not in PWM mode.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:PWM[:DEViation]:DCYCle {<percent>|MINimum|MAXimum}
[SOURce[1|2]]:PWM[:DEViation]:DCYCle? [MINimum|MAXimum]
```

Arguments

`<percent>::=<NR2>[<units>]`

Where: <NR2> is the PWM deviation.

`<units>::=PCT`

Returns

`<percent>`

Examples

`SOURce1:PWM:DCYCle 5.0`

Set the channel 1 (CH 1) PWM deviation to 5.0%.

[SOURce]:ROSCillator:SOURce

This command sets the reference clock to either internal or external.

Group

Source

Related Commands

None

Syntax

```
[SOURce]:ROSCillator:SOURce {INTernal|EXTernal}
[SOURce]:ROSCillator:SOURce?
```

Arguments

`INTernal` means that the reference clock is set to internal.

`EXTernal` means that the reference clock is set to external.

Returns

`INT|EXT`

Examples

`SOURce:ROSCillator:SOURce INTernal`

Select the internal clock reference.

[SOURce[1|2]]:SWEep:HTIMe

This command sets or queries the sweep hold time. Hold time represents the amount of time that the frequency must remain stable after reaching the stop frequency. This command will cause an error if not in Sweep mode.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:SWEep:HTIMe {<seconds>|MINimum|MAXimum}
[SOURce[1|2]]:SWEep:HTIMe? [MINimum|MAXimum]
```

Arguments

`<seconds>::=<NRf>[<units>]`

Where: `<NRf>` is the hold time in seconds.

`<units>::=[ns|μs|ms|s]`

Returns

`<seconds>`

Examples

`SOURce1:SWEep:HTIMe 1ms`

Sets the channel 1 (CH 1) hold time to 1 ms.

[SOURce[1|2]]:SWEep:MODE

This command selects auto or manual for the sweep mode for the specified channel. The query version of this command returns the sweep mode for the specified channel. This command will cause an error if not in Sweep mode.

Group

Source

Related Commands

[SOURce[1|2]]:SWEep:HTIMe (on page 3-59)
[SOURce[1|2]]:SWEep:RTIMe (on page 3-59)
[SOURce[1|2]]:SWEep:TIME (on page 3-60)
TRIGger[:SEQuence]:SOURce (on page 3-79)
TRIGger[:SEQuence]:TIMer (on page 3-80)

Syntax

```
[SOURce[1|2]]:SWEep:MODE {AUTO|MANual}
[SOURce[1|2]]:SWEep:MODE?
```

Arguments

`AUTO` sets the sweep mode to automatic. The instrument outputs a continuous sweep at a rate specified by Sweep Time, Hold Time, and Return Time.

`MANual` sets the sweep mode to manual. The instrument outputs one sweep when a trigger input is received.

Returns

`AUTO|MAN`

Examples

`SOURce1:SWEep:MODE AUTO`

Set the channel 1 (CH1) sweep mode to auto. The instrument outputs a continuous sweep.

[SOURce[1|2]]:SWEep:RTIMe

This command sets or queries the sweep return time. Return time represents the amount of time from stop frequency through start frequency. Return time does not include hold time. This command will cause an error if not in SWEEP mode.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:SWEep:RTIMe {<seconds>|MINimum|MAXimum}
[SOURce[1|2]]:SWEep:RTIMe? [MINimum|MAXimum]
```

Arguments

`<seconds>::=<NRf>[<units>]`

Where: `<NRf>` is the return time in seconds.

`<units>::=[ns|µs|ms|s]`

Returns

`<seconds>`

Examples

`SOURce1:SWEep:RTIMe 1ms`

Sets the channel 1 (CH 1) return time to 1 ms.

[SOURce[1|2]]:SWEep:SPACing

This command selects linear or logarithmic spacing for the sweep for the specified channel. The query form of this command returns the type for the sweep spacing for the specified channel. This command will cause an error if not in SWEEP mode.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:SWEep:SPACing {LINear|LOGarithmic}
[SOURce[1|2]]:SWEep:SPACing?
```

Arguments

`LINear` sets the sweep spacing to linear.

`LOGarithmic` sets the sweep spacing to logarithmic.

Returns

```
LIN|LOG
```

Examples

```
SOURce1:SWEep:SPACing LINear
```

Set the channel 1 (CH1) sweep spacing to linear.

[SOURce[1|2]]:SWEep:TIME

This command sets or queries the sweep time for the sweep for the specified channel. The sweep time does not include hold time and return time. This command will cause an error if not in Sweep mode.

Group

Source

Syntax

```
[SOURce[1|2]]:SWEep:TIME {<seconds>|MINimum|MAXimum}
[SOURce[1|2]]:SWEep:TIME? [MINimum|MAXimum]
```

Arguments

```
<seconds>::=<NRf>[<units>]
```

Where: `<NRf>` is the sweep time in seconds.

```
<units>::=[ns|µs|ms|s]
```

Returns

```
<seconds>
```

Examples

```
SOURce1:SWEep:TIME 100ms
```

Set the channel 1 (CH 1) sweep time to 100 ms.

[SOURce[1|2]]:VOLTage:CONCurrent[:STATe]

This command enables or disables the function to copy the voltage level of one channel to another channel. This command copies the voltage level of the channel specified by the header suffix to another channel. If you specify channel 1 (CH 1) with the header, the CH 1 voltage level will be copied to channel 2 (CH 2).

The query form of this command returns "0" (off) or "1" (on).

If your arbitrary waveform generator is a single-channel model, this command is not supported.

Group

Source

Related Commands

None

Syntax

```
[SOURce[1|2]]:VOLTage:CONCurrent[:STATe] {ON|OFF|<NR1>}
[SOURce[1|2]]:VOLTage:CONCurrent[:STATe]?
```

Arguments

ON or <NR1> ≠ 0 enables the concurrent copy function.

OFF or <NR1> = 0 disables the concurrent copy function.

Returns

<NR1>

Examples

```
SOURce1:VOLTage:CONCurrent:STATe ON
```

Enable the concurrent copy function.

[SOURce[1|2]]:VOLTage[:LEVel][:IMMediate]:HIGH

This command sets or queries the high level of output amplitude for the specified channel. If your instrument is a dual-channel model and the [SOURce[1|2]]:VOLTage:CONCurrent[:STATe] command is set to ON, then the high level of the other channel is also the same value.

Group

Source

Related Commands

[SOURce[1|2]]:VOLTage:CONCurrent[:STATe] (on page 3-60)

Syntax

```
[SOURce[1|2]]:VOLTage[:LEVel][:IMMediate]:HIGH{<voltage>|MINimum|
   MAXimum}
[SOURce[1|2]]:VOLTage[:LEVel][:IMMediate]:HIGH?[MINimum|MAXimum]
```

Arguments

<voltage>::=<NRf>[<units>]

Where: <NRf> is the high level of output amplitude.

<units>::=[mV | V]

Returns

<voltage>

Examples

```
SOURce1:VOLTage:LEVel:IMMediate:HIGH 1V
```

Set the high level of channel 1 (CH 1) output amplitude to 1 V.

[SOURce[1|2]]:VOLTage[:LEVel][:IMMediate]:LOW

This command sets or queries the low level of the output amplitude for the specified channel. If your instrument is a dual-channel model and the `[SOURce[1|2]]:VOLTage:CONCurrent[:STATe]` command is set to ON, then the low level of the other channel is also the same value.

Group

Source

Related Commands

[SOURce[1|2]]:VOLTage:CONCurrent[:STATe] (on page 3-60)

Syntax

```
[SOURce[1|2]]:VOLTage[:LEVel][:IMMediate]:LOW{<voltage>|MINimum|
   MAXimum}
[SOURce[1|2]]:VOLTage[:LEVel][:IMMediate]:LOW? [MINimum|MAXimum]
```

Arguments

`<voltage>::=<NRf>[<units>]`

Where: `<NRf>` is the low level of the output amplitude.

`<units>::=[mV|V]`

Returns

`<voltage>`

Examples

`SOURce1:VOLTage:LEVel:IMMediate:LOW -1V`

Set the low level of channel 1 (CH 1) output amplitude to –1 V.

[SOURce[1|2]]:VOLTage[:LEVel][:IMMediate]:OFFSet

This command sets or queries the offset level for the specified channel. If your instrument is a dual-channel model and the `[SOURce[1|2]]:VOLTage:CONCurrent[:STATe]` command is set to ON, then the offset level of the other channel is also the same value.

Group

Source

Related Commands

[SOURce[1|2]]:VOLTage:CONCurrent[:STATe] (on page 3-60)

Syntax

```
[SOURce[1|2]]:VOLTage[:LEVel][:IMMediate]:OFFSet{<voltage>|MINimum|
   MAXimum}
[SOURce[1|2]]:VOLTage[:LEVel][:IMMediate]:OFFSet?[MINimum|MAXimum]
```

Arguments

```
<voltage>::=<NRf>[<units>]
```

Where: `<NRf>` is the offset voltage level.

```
<units>::=[mV|V]
```

Returns

```
<voltage>
```

Examples

```
SOURce1:VOLTage:LEVel:IMMediate:OFFSet 500mV
```

Set the channel 1 (CH 1) offset level to 500 mV.

[SOURce[1|2]]:VOLTage[:LEVel][:IMMediate][:AMPLitude]

This command sets or queries the output amplitude for the specified channel. If your instrument is a two-channel model and the `[SOURce[1|2]]:VOLTage:CONCurrent[:STATe]` command is set to ON, then the output amplitude of the other channel is the same value.

You can set the units of output amplitude by using either the bezel menu selection or the `[SOURce[1|2]]:VOLTage:UNIT` command. Selections made using the bezel menu have priority over the remote command.

Group

Source

Related Commands

[SOURce[1|2]]:VOLTage:CONCurrent[:STATe] (on page 3-60)

Syntax

```
[SOURce[1|2]]:VOLTage[:LEVel][:IMMediate][:AMPLitude]{<amplitude>|
   MINimum|MAXimum}
[SOURce[1|2]]:VOLTage[:LEVel][:IMMediate][:AMPLitude]?
   [MINimum|MAXimum]
```

Arguments

```
<amplitude>::=<NRf>[<units>]
```

Where: `<NRf>` is the output amplitude.

```
<units>::=[VPP|VRMS|DBM]
```

Returns

```
<amplitude>
```

Examples

```
SOURce1:VOLTage:LEVel:IMMediate:AMPLitude 1V
```

Set the channel 1 (CH 1) output amplitude to 1 V.

Example error:
-102,"Syntax error; possible invalid suffix - SOURce1:VOLTage:LEVel:IMMediate:AMPLitude 1V\0A"

[SOURce[1|2]]:VOLTage:LIMit:HIGH

This command sets or queries the higher limit of the output amplitude high level for the specified channel. If your instrument is a dual-channel model and the `[SOURce[1|2]]:VOLTage:CONCurrent[:STATe]` command is set to ON, then the higher level limit of the other channel is the same value.

Group

Source

Related Commands

[SOURce[1|2]]:VOLTage:CONCurrent[:STATe] (on page 3-60)

Syntax

```
[SOURce[1|2]]:VOLTage:LIMit:HIGH {<voltage>|MINimum|MAXimum}
[SOURce[1|2]]:VOLTage:LIMit:HIGH? [MINimum|MAXimum]
```

Arguments

`<voltage>::=<NRf>[<units>]`

Where: `<NRf>` is the higher limit of output amplitude.

`<units>::=[mV|V]`

Returns

`<voltage>`

Examples

`SOURce1:VOLTage:LIMit:HIGH 1V`

Set the higher limit of channel 1 (CH 1) output amplitude to 1 V.

[SOURce[1|2]]:VOLTage:LIMit:LOW

This command sets or queries the lower limit of the output amplitude low level for the specified channel. If your instrument is a dual-channel model and the `[SOURce[1|2]]:VOLTage:CONCurrent[:STATe]` command is set to ON, then the low level lower limit of the other channel is the same value.

Group

Source

Related Commands

[SOURce[1|2]]:VOLTage:CONCurrent[:STATe] (on page 3-60)

Syntax

```
[SOURce[1|2]]:VOLTage:LIMit:LOW {<voltage>|MINimum|MAXimum}
[SOURce[1|2]]:VOLTage:LIMit:LOW? [MINimum|MAXimum]
```

Arguments

`<voltage>::=<NRf>[<units>]`

Where: `<NRf>` is the lower limit of the output amplitude.

`<units>::=[mV|V]`

Returns

`<voltage>`

Examples

`SOURce1:VOLTage:LIMit:LOW 10mV`

Set the lower limit of channel 1 (CH 1) output amplitude to 10 mV.

[SOURce[1|2]]:VOLTage:UNIT

This command sets or queries the units of output amplitude for the specified channel. This command does not affect the offset, high level, or low level of output. This command is not affected by the units setting of the `[SOURce[1|2]]:VOLTage[:LEVel][:IMMediate][:AMPLitude]` command.

If your instrument is a dual-channel model and the `[SOURce[1|2]]:VOLTage:CONCurrent[:STATe]` command is set to ON, then the units of the other channel are set the same.

Group

Source

Related Commands

[SOURce[1|2]]:VOLTage:CONCurrent[:STATe] (on page 3-60)
[SOURce[1|2]]:VOLTage[:LEVel][:IMMediate][:AMPLitude] (on page 3-65)

Syntax

```
[SOURce[1|2]]:VOLTage:UNIT {VPP|VRMS|DBM}
[SOURce[1|2]]:VOLTage:UNIT?
```

Arguments

`VPP` sets the units of the output voltage to Vp-p.

`VRMS` sets the units of the output voltage to Vrms.

`DBM` sets the units of the output voltage to dBm. You cannot specify DBM if the load impedance is set to infinite.

Returns

```
VPP|VRMS|DBM
```

Examples

```
SOURce1:VOLTage:UNIT VPP
```

Set the voltage units to Vp-p.

[SOURce[1|2]]:VOLTage:VOCM

This command sets or queries the VOCM for the specified channel.

Group

Source

Related Commands

[SOURce[1|2]]:VOLTage[:LEVel][:IMMediate][:AMPLitude] (on page 3-65)

Syntax

```
[SOURce[1|2]]:VOLTage:VOCM {<vocm>|MINimum|MAXimum}
[SOURce[1|2]]:VOLTage:VOCM? [MINimum|MAXimum]
```

Arguments

```
<voltage>::=<NRf>[<units>]
```

Where: `<NRf>` is the VOCM value.

```
<units>::=[ V]
```

Returns

```
<voltage>
```

Examples

```
SOURce1:VOLTage:VOCM 2V
```

Set the vocm of channel 1 (CH 1) output to 2 V.

*SRE

This command sets and queries the bits in the Service Request Enable Register (SRER).

Group

Status

Related Commands

None

Syntax

```
*SRE <bit_value>
*SRE?
```

Arguments

```
<bit_value>::=<NR1>
```

Where: <NR1> is a value from 0 through 255. The binary bits of the SRER are set according to this value. Using an out-of-range value causes an execution error.

Returns

```
<bit_value>
```

Examples

```
*SRE 48
```

Set the bits in the SRER to binary 00110000.

```
*SRE?
```

Example return: 32, which means that the bits in the SRER have the binary value of 00100000.

STATus:OPERation:CONDition?

This command returns the contents of the Operation Condition Register. This command is query only.

Group

Status

Syntax

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

Arguments

None

Returns

```
<bit_value>::=<NR1>
```

Examples

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

Example return: 32, which indicates that the OCR contains the binary number 00000000 00100000 and channel 1 (CH 1) of the instrument is waiting for a trigger.

STATus:OPERation:ENABle

This command sets or queries the mask of the Operation Enable Register (OER).

Group

Status

Syntax

```
STATus:OPERation:ENABle <bit_value>
STATus:OPERation:ENABle?
```

Arguments

```
<bit_value>::=<NR1>
```

Returns

```
<bit_value>
```

Examples

```
STATus:OPERation:ENABle 1
```

Set the calibrating bit in the OENR to on.

STATus:OPERation[:EVENt]?

This command returns the value in the Operation Event Register (OEVR) and clears the OEVR. This command is query only.

Group

Status

Syntax

```
STATus:OPERation[:EVENt]?
```

Arguments

None

Returns

```
<NR1>
```

Examples

```
STATus:OPERation:EVENt?
```

Example return: 1, which indicates that the OEVR contains the binary number 00000000 00000001 and the calibrating bit is set to on.

STATus:PRESet

This command presets the SCPI status registers (Operation Enable Register (OENR) and Questionable Enable Register (QENR)). There is no query form of this command.

Group

Status

Related Commands

None

Syntax

```
STATus:PRESet
```

Arguments

None

Examples

```
STATus:PRESet
```

Presets the SCPI status registers.

STATus:QUEStionable:CONDition?

This command returns the contents of the Questionable Condition Register (QCR). This command is query only.

Group

Status

Related Commands

None

Syntax

```
STATus:QUEStionable:CONDition?
```

Arguments

None

Returns

```
<bit_value>::=<NR1>
```

Examples

```
STATus:QUEStionable:CONDition?
```

Example return: 32, which indicates that the QCR contains the binary number 00000000 00100000 and the accuracy of frequency is questionable.

STATus:QUEStionable:ENABle

This command sets or queries the mask for the Questionable Enable Register (QENR).

Group

Status

Related Commands

None

Syntax

```
STATus:QUEStionable:ENABle <bit_value>
STATus:QUEStionable:ENABle?
```

Arguments

```
<bit_value>::=<NR1>
```

Returns

```
<bit_value>
```

Examples

```
STATus:QUEStionable:ENABle 32
```

Set the frequency bit in the QENR to on.

STATus:QUEStionable[:EVENt]?

This command returns the value in the Questionable Event Register (QEVR) and clears the QEVR. This command is query only.

Group

Status

Related Commands

None

Syntax

```
STATus:QUEStionable[:EVENt]?
```

Arguments

None

Returns

```
<bit_value>::=<NR1>
```

Examples

```
STATus:QUEStionable:EVENt?
```

Example return: 32, which indicates that the QEVR contains the binary number 00000000 00100000 and the frequency bit is set to on.

*STB?

This command returns the contents of the Status Byte Register (SBR) using the Master Summary Status (MSS) bit. This command is query only.

Group

Status

Related Commands

None

Syntax

```
*STB?
```

Arguments

None

Returns

```
<NR1>
```

Examples

```
*STB?
```

Example return: 96, which indicates that the SBR contains the binary value 01100000.

SYSTem:BEEPer[:IMMediate]

This command causes the instrument to beep immediately. There is no query form of this command.

Group

System

Related Commands

None

Syntax

```
SYSTem:BEEPer[:IMMediate]
```

Arguments

None

Examples

```
SYSTem:BEEPer:IMMediate
```

Causes an audible beep.

SYSTem:BEEPer:STATe

This command enables or disables the instrument beeper. The query form of this command returns "0" (OFF) or "1" (ON).

When the beeper is set to ON, the instrument will beep when an error message or a warning message is displayed on the screen. The instrument does not beep when an error or warning is caused by remote command execution.

Group

System

Related Commands

SYSTem:BEEPer[:IMMediate] (on page 3-73)

Syntax

```
SYSTem:BEEPer:STATe {ON|OFF|<NR1>}
SYSTem:BEEPer:STATe?
```

Arguments

ON or <NR1> ≠ 0 enables the beeper.

OFF or <NR1> = 0 disables the beeper.

Returns

<NR1>

Examples

```
SYSTem:BEEPer:STATe ON
```

Enable the beeper function.

SYSTem:ERRor[:NEXT]?

This command returns the contents of the Error/Event queue. This command is query only.

Group

System

Syntax

```
SYSTem:ERRor[:NEXT]?
```

Arguments

None

Returns

```
<Error/event number>::=<NR1>
<Error/event description>::=<string>
```

Examples

```
SYSTem:ERRor:NEXT?
```

Example return: –410,"Query INTERRUPTED"

If the instrument detects an error or an event occurs, the event number and event message will be returned.

SYSTem:KCLick[:STATe]

This command enables or disables the key click sound when you push the front panel buttons or turn the general-purpose knob. The query form of this command returns "0" (OFF) or "1" (ON).

Group

System

Related Commands

None

Syntax

```
SYSTem:KCLick[:STATe] {ON|OFF|<NR1>}
SYSTem:KCLick[:STATe]?
```

Arguments

ON or <NR1> ≠ 0 enables key click sound.

OFF or <NR1> = 0 disables key click sound.

Returns

<NR1>

Examples

```
SYSTem:KCLick:STATe ON
```

Enable the key click sound.

SYSTem:KLOCk[:STATe]

This command locks or unlocks the instrument front-panel controls. The query form of this command returns "0" (OFF) or "1" (ON).

Group

System

Related Commands

None

Syntax

```
SYSTem:KLOCk[:STATe] {ON|OFF|<NR1>}
SYSTem:KLOCk[:STATe]?
```

Arguments

ON or <NR1> ≠ 0 locks front-panel controls.

OFF or <NR1> = 0 unlocks front-panel controls.

Returns

<NR1>

Examples

```
SYSTem:KLOCk:STATe ON
```

Locks the front-panel controls.

SYSTem:ULANguage?

This command queries the language that the instrument uses to display the information on the screen. This command is query only.

Group

System

Syntax

```
SYSTem:ULANguage?
```

Arguments

NONE

Returns

ENGLish|SCHinese

Examples

```
SYSTem:ULANguage?
```

Example return: ENGL, which means that the instrument displays information in English.

SYSTem:VERSion?

This query-only command returns the SCPI conformance version of the instrument. This command is query only.

Group

System

Related Commands

None

Syntax

```
SYSTem:VERSion?
```

Arguments

None

Returns

```
<SCPI Version>::=YYYY.V
```

Where:

YYYY indicates the year.

V indicates the version number for that year.

Examples

```
SYSTem:VERSion?
```

Example return: 1999.0

TIME

This command sets the system time. The query form of this command returns the system time.

Group

Date and Time

Related Commands

DATE (on page 3-3)

Syntax

```
TIME <time_string>
TIME?
```

Arguments

`<time_string>::=<string>` specifies a date in the format "17:25:30".

Returns

```
<time_string>::<string>
```

Example

```
TIME "17:25:30"
```

Set the system time to 17:25:30.

TRACe|DATA[:DATA]

This command transfers the waveform data from the external controller to the arbitrary buffer in the arbitrary waveform generator. The query form of this command returns the binary block data.

Group

Trace

Related Commands

None

Syntax

```
TRACe|DATA[:DATA] <ARBB1|ARBB2>,<binary_block_data>
TRACe|DATA[:DATA]? <ARBB1|ARBB2>
```

Arguments

<ARBB1|ARBB2> arbitrary buffer 1 or 2

`<binary_block_data>`

Where: `<binary_block_data>` is the waveform data in binary format.

Returns

`<binary_block_data>`

Examples

```
DATA:DATA ARBB1,#42000<DAB><DAB>...<DAB>
```

Transmit a waveform to the arbitrary buffer memory in the arbitrary waveform generator. The block data element #42000 indicates that 4 is the number of digits in 2000 (byte count) and the 2000 bytes of binary data are to be transmitted.

TRACe|DATA:POINts?

This command queries the number of data points for the waveform in the arbitrary buffer memory. This command is query only.

Group

Trace

Related Commands

None

Syntax

```
TRACe|DATA:POINts? <ARBB1|ARBB2>[,{MIN|MAX}]
```

Returns

`<NR1>`

Examples

```
DATA:POINts? ARBB1,MAX
```

Example return: 16384, which is the maximum number of points for arbitrary buffer 1.

TRACe|DATA:RECall

This command recalls the contents of arbitrary buffer memory from a specified file in the file system. There is no query form of this command.

Group

Trace

Related Commands

None

Syntax

```
TRACe|DATA:RECall <ARBB1|ARBB2>,<filename>
```

Arguments

`< ARBB1| ARBB2>::` Arbitrary buffer number.

`<filename>::` The file name you want to recall data from.

Examples

```
DATA:RECall ARBB1,"waveform1.tfw"
```

Recalls the waveform data in the arbitrary memory from the file "waveform1.tfw".

TRACe|DATA:SAVE

This command saves the contents of arbitrary buffer memory to a specified file in the file system. There is no query form of this command.

Group

Trace

Syntax

```
TRACe|DATA:SAVE <ARBB1|ARBB2>,<filename>
```

Arguments

`<ARBB1|ARBB2>::` Arbitrary buffer number.

`<filename>::` The name of the file you want to save.

Examples

```
DATA:SAVE ARBB1,"waveform1.tfw"
```

Copies the waveform data in the arbitrary memory to the file "waveform1.tfw".

*TRG

This command generates a trigger event. There is no query form of this command.

Group

Trigger

Related Commands

TRIGger[:SEQuence][:IMMediate] (on page 3-78)

Syntax

```
*TRG
```

Arguments

None

Examples

```
*TRG
```

Generate a trigger event.

TRIGger[:SEQuence][:IMMediate]

This command forces a trigger event to occur. There is no query form of this command.

Group

Trigger

Related Commands

None

Syntax

```
TRIGger[:SEQuence][:IMMediate]
```

Arguments

None

Examples

```
TRIGger:SEQuence:IMMediate
```

Generate a trigger event.

TRIGger[:SEQuence]:SLOPe

This command sets or queries the slope of the trigger signal.

Group

Trigger

Related Commands

None

Syntax

```
TRIGger[:SEQuence]:SLOPe {POSitive|NEGative}
TRIGger[:SEQuence]:SLOPe?
```

Arguments

`POSitive` indicates that the event occurs on the rising edge of the external trigger signal.

`NEGative` indicates that the event occurs on the falling edge of the external trigger signal.

Returns

`POS|NEG`

Examples

```
TRIGger:SEQuence:SLOPe POSitive
```

Set the trigger slope to positive, which triggers on the rising edge of the signal.

TRIGger[:SEQuence]:SOURce

This command sets or queries the trigger source for an external trigger signal.

Group

Trigger

Related Commands

None

Syntax

```
TRIGger[:SEQuence]:SOURce {TIMer|EXTernal|MANual}
TRIGger[:SEQuence]:SOURce?
```

Arguments

`TIMer` specifies an internal clock as the trigger source.

`EXTernal` specifies an external trigger input as the trigger source.

`MANual` specifies a manual trigger input as the trigger source.

Returns

```
TIM|EXT|MAN
```

Examples

```
TRIGger:SEQuence:SOURce EXTernal
```

Sets an external trigger input as the trigger source.

TRIGger[:SEQuence]:THREshold

This command sets or queries the threshold of an external trigger signal when you select the external trigger input as the trigger source with the `TRIGger[:SEQuence]:SOURce` command.

Group

Trigger

Related Commands

TRIGger[:SEQuence]:SOURce (on page 3-79)

Syntax

```
TRIGger[:SEQuence]:THREshold <threshold>
TRIGger[:SEQuence]:THREshold?
```

Arguments

```
<threshold>::=<NRf>[<units>]
```

Where: `<units>::=[V]`

Returns

```
<threshold>
```

Examples

```
TRIGger:SEQuence:THREshold 1
```

Set the external trigger input level to 1 V.

TRIGger[:SEQuence]:TIMer

This command sets or queries the period of an internal clock when you select the internal clock as the trigger source with the `TRIGger[:SEQuence]:SOURce` command.

Group

Trigger

Related Commands

TRIGger[:SEQuence]:SOURce (on page 3-79)

Syntax

```
TRIGger[:SEQuence]:TIMer <seconds>
TRIGger[:SEQuence]:TIMer?
```

Arguments

```
<seconds>::=<NRf>[<units>]
```

Where: `<units>::=[µs|ms|s]`

Returns

<seconds>

Examples

```
TRIGger:SEQuence:TIMer 5ms
```

Set the internal trigger rate to 5 ms.

*TST?

This command performs a self-test and returns the results. This command is query only.

NOTE. *The self-test can take several minutes to complete. During this time, the arbitrary waveform generator does not execute any commands. Do not power off the instrument during the self-test.*

Before executing this command, allow a 30-minute warm-up period after powering on the instrument. If you do not allow the instrument to reach a valid temperature before performing calibration, the calibration will not be valid. You can go to **Basic application ➔ System ➔ Tools ➔ Warm Up Timer** to do the warm-up.

Group

Calibration and Diagnostic

Related Commands

DIAGnostic (on page 3-4)

Syntax

```
*TST?
```

Arguments

None

Returns

`<NR1>`

Where:

`<NR1> =` 0 indicates that the self-test completed without errors.

`<NR1> ≠` 0 indicates that the arbitrary waveform generator detected an error.

Examples

```
*TST?
```

Perform a self-test; returns 0 if no error is reported.

*WAI

This command prevents the instrument from executing further commands or queries until all pending commands that generate an operation complete (OPC) message are complete. There is no query form of this command.

Group

Synchronization

Related Commands

*OPC (on page 3-16)

Syntax

```
*WAI
```

Arguments

None

Examples

```
*WAI
```

Prevent the instrument from executing any further commands or queries until all pending commands that generate an OPC message are complete.

Status and Events

This section provides details about the status information and events the arbitrary waveform generator reports.

Status Reporting Structure

The arbitrary waveform generator status reporting functions conform to IEEE-488.2 and SCPI standards. Use the status reporting function to check for instrument errors and to identify the types of events that have occurred on the instrument.

The error and event reporting system consists of the following three blocks:

- Standard/Event Status
- Operation Status
- Questionable Status

The operations processed in these blocks are summarized in status bytes, which provide the error and event data.

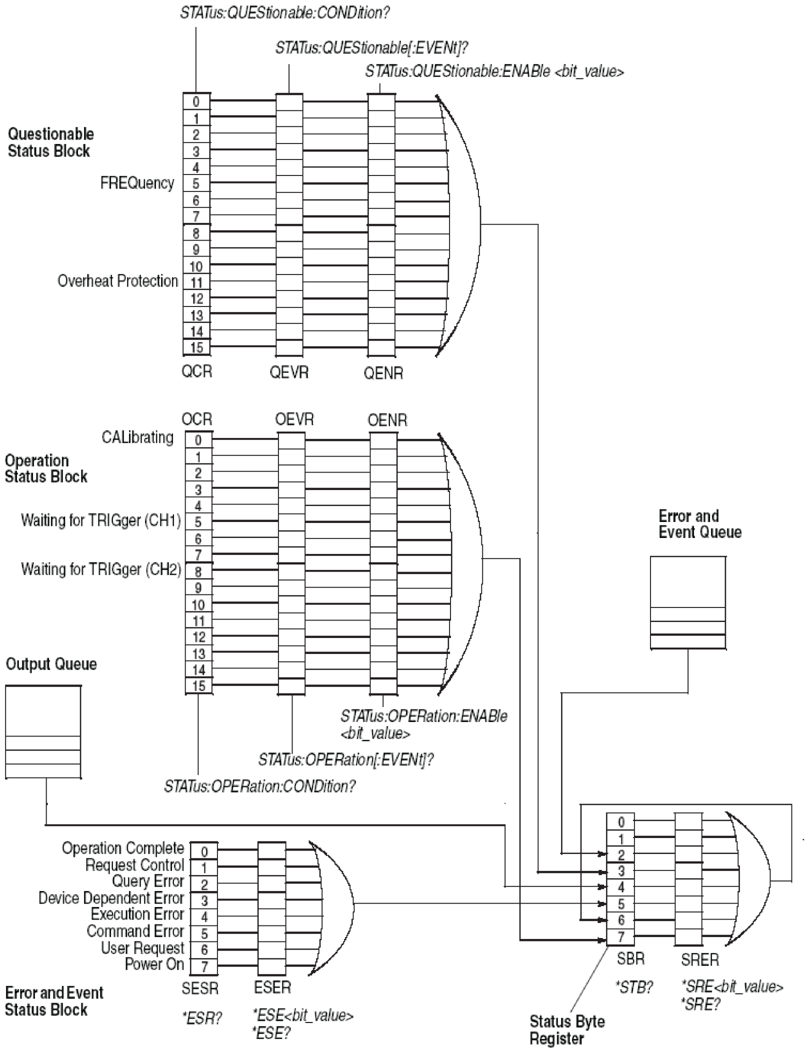


Figure 11: Error and event handling process

Standard/Event Status Block

This block is used to report power on/off, command errors, and command execution status.

The block has two registers: the Standard Event Status Register (SESR) and the Event Status Enable Register (ESER).

Standard Event Status Register. The SESR is an eight-bit status register. When an error or other type of event occurs on the instrument, the corresponding bit is set. You cannot write to this register.

Event Status Enable Register. The ESER is an eight-bit enable register that masks the SESR. You can set this mask, and take AND with the SESR to determine whether or not the ESB bit in the Status Byte Register (SBR) should be set.

Operation Status Block

This block is used to report on the status of several operations being executed by the arbitrary waveform generator.

The block has three registers: the Operation Condition Register (OCR), the Operation Event Register (OEVR), and the Operation Enable Register (OENR).

Operation Condition Register. When the instrument achieves a certain status, the corresponding bit is set in the OCR. You cannot write to this register.

Operation Event Register. The OCR bits that have changed from false (reset) to true (set) status are set in the OEVR.

Operation Enable Register. The function of the OENR is to mask the OEVR. You can set this mask and take AND with the OEVR to determine whether or not the OSS bit in the Status Byte Register (SBR) should be set.

Questionable Status Block

This block reports on the status of signals and data, such as the accuracy of entered data and signals generated by the instrument. The register configuration and process flow are the same as the Questionable Status Block.

Registers

The registers in the event reporting system fall into two functional groups:

- The Status Registers contain information about the status of the instrument.
- The Enable Registers determine whether selected types of events are reported to the Status Registers and the Event Queue.

Status Registers

There are seven types of status registers:

- Status Byte Register (SBR) (on page 4-4)
- Standard Event Status Register (SESR) (on page 4-5)
- Operation Event Register (OEVR) (on page 4-6)
- Questionable Enable Register (QENR) (on page 4-9)
- Operation Condition Register (OCR) (on page 4-6)
- Questionable Event Register (QEVR) (on page 4-6)
- Questionable Condition Register (QCR) (on page 4-7)

Status Byte Register (SBR)

The SBR is made up of 8 bits. Bits 4, 5, and 6 are defined in accordance with IEEE Std 488.2-1992. These bits are used to monitor the output queue, SESR, and service requests, respectively.

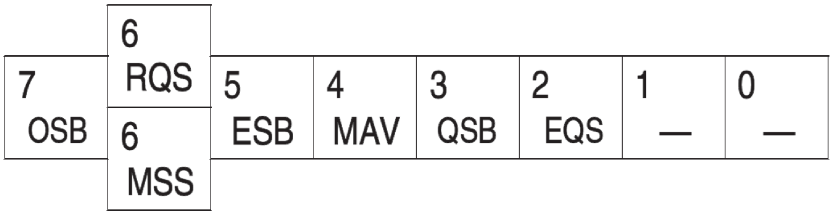


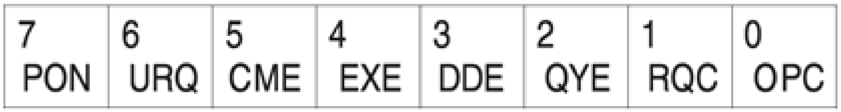
Figure 12: The Status Byte Register (SBR)

SBR bit functions

Bit	**Function**	
7 (MSB)	OSB	Operation Status Bit. Indicates that an operation event has occurred.
6	RQS	Request Service. When the instrument is accessed, this bit is called the Request Service (RQS) bit and indicates to the controller that a service request has occurred. The RQS bit is cleared when serial poll ends.
6	MSS	Master Status Summary. When the instrument is accessed using the *STB? query, this bit is called the Master Status Summary (MSS) bit and indicates that the instrument has issued a service request for one or more reasons. The MSS bit is never cleared to 0 by the *STB? query.
5	ESB	Event Status Bit. This bit indicates whether or not a new event has occurred after the previous Standard Event Status Register (SESR) has been cleared or after an event readout has been performed.
4	MAV	Message Available Bit. This bit indicates that a message has been placed in the output queue and can be retrieved.
3	QSB	Questionable Status Bit.
2	EQS	Error/Event Queue Summary.
1-0	—	Not used.

Standard Event Status Register (SESR)

The SESR records eight types of events that can occur within the instrument.



7	6	5	4	3	2	1	0
PON	URQ	CME	EXE	DDE	QYE	RQC	OPC

Figure 13: The Standard Event Status Register (SESR)

SESR bit functions

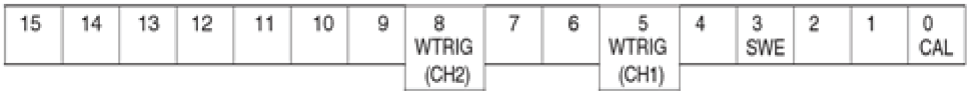
Bit	Function	
7 (MSB)	PON	Power On. Indicates that the power to the instrument is on.
6	URQ	User Request. Indicates that an application event has occurred. The arbitrary waveform generator does not use this bit.
5	CME	Command Error. Indicates that an error occurred while the arbitrary waveform generator was parsing a command or query.
4	EXE	Execution Error. Indicates that an error occurred while the arbitrary waveform generator was executing a command or query. Execution errors occur for one of the following reasons: • A value designated in the argument is outside the allowable range of the instrument, or is in conflict with the capabilities of the instrument. • The command was not properly executed because the conditions for execution did not meet required conditions.
3	DDE	Device Error. An instrument error has been detected.
2	QYE	Query Error. Indicates that a query error has been detected by the output queue controller. Query errors occur for one of the following reasons: • An attempt was made to retrieve messages from the output queue when the output queue is empty or in pending status. • The output queue message was cleared while it was being retrieved from the output queue.
1	RQC	Request Control. The arbitrary waveform generator does not use this bit.
0	OPC	Operation Complete. Indicates that the operation is complete. This bit is set when all pending operations complete following the `*OPC` command.

Operation Event Register (OEVR)

This register has the same content as the Operation Condition Register.

Operation Condition Register (OCR)

The Operation Condition Register is made up of sixteen bits, which note the occurrence of events.



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							WTRIG (CH2)			WTRIG (CH1)		SWE			CAL

Figure 14: Operation Condition Register (OCR)

OCR bit functions

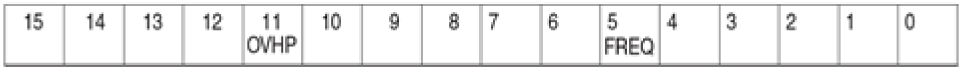
Bit	Function	
15 to 9	—	Not used.
8	WTRIG CH2	Waiting for Trigger. Indicates whether the instrument is waiting for a trigger. This bit is set when CH 2 (in the case of dual-channel model) is waiting for a trigger. Bit is reset when the waiting-for-trigger status is canceled.
5	WTRIG CH1	Waiting for Trigger. Indicates whether the instrument is waiting for a trigger. This bit is set when CH 1 (in the case of dual-channel model) is waiting for a trigger. Bit is reset when the waiting-for-trigger status is canceled.
4	—	Not used.
3	SWE	Sweep. Indicates whether the instrument is executing a frequency sweep. This bit is set when a frequency sweep is being executed on CH 1 or another channel (in the case of dual-channel model). Bit is reset when the execution stops.
2 to 1	—	Not used.
0	CAL	Calibration. Indicates whether the instrument is being calibrated. This bit is set when calibration is in progress and is reset when the calibration is complete.

Questionable Event Register (QEVR)

This register has the same content as the Questionable Condition Register.

Questionable Condition Register (QCR)

The Questionable Condition Register is made up of sixteen bits which note the occurrence of two types of events.



15	14	13	12	11 OVHP	10	9	8	7	6	5 FREQ	4	3	2	1	0

Figure 15: Questionable Condition Register (QCR)

QCR bit functions

Bit	Function	
15 to 12	—	Not used.
11	OVHP	Overheat protection. Indicates whether the instrument internal temperature is in a questionable condition.
10 to 6	—	Not used.
5	FREQ	Frequency. Indicates whether frequency accuracy of the signal is of questionable quality.
4 to 0	—	Not used.

Enable Registers

There are four types of enable registers:

- Event Status Enable Register (ESER) (on page 4-8)
- Service Request Enable Register (SRER) (on page 4-8)
- Operation Enable Register (OENR) (on page 4-9)
- Questionable Enable Register (QENR) (on page 4-9)

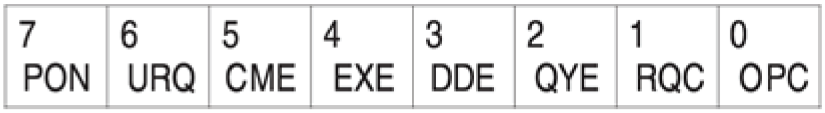
Each bit in the enable registers corresponds to a bit in the controlling status register. By setting and resetting the bits in the enable register, you can determine whether or not events that occur will be registered to the status register and queue.

Event Status Enable Register (ESER)

The ESER consists of bits defined exactly the same as bits 0 through 7 in the SESR register. You can use this register to control whether or not the Event Status Bit (ESB) in the SBR should be set when an event has occurred, and to determine if the corresponding SESR bit is set.

To set the ESB in the SBR (when the SESR bit has been set), set the ESER bit corresponding to that event. To prevent the ESB from being set, reset the ESER bit corresponding to that event.

Use the *ESC command to set the bits in the ESER. Use the *ESR? query to read the contents of the ESER.



7	6	5	4	3	2	1	0
PON	URQ	CME	EXE	DDE	QYE	RQC	OPC

Figure 16: Event Status Enable Register (ESER)

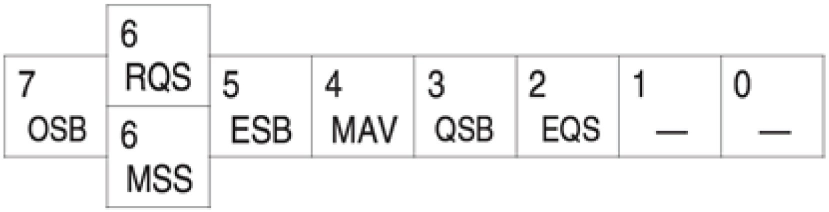
Service Request Enable Register (SRER)

The SRER consists of bits defined exactly the same as bits 0 through 7 in the SBR. You can use this register to define which events will generate service requests.

The SRER bit 6 cannot be set. Also, the RQS is not maskable.

The generation of a service request involves changing the SRQ line to LOW, and making a service request to the controller. The result is that a status byte for which an RQS has been set is returned in response to serial polling by the controller.

Use the *SRE command to set the bits of the SRER. Use the *SRE? query to read the contents of the SRER. Bit 6 must be set to 0.



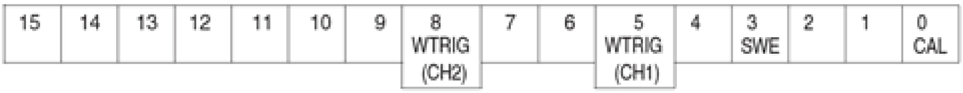
7	6	5	4	3	2	1	0
OSB	RQS 6 MSS	ESB	MAV	QSB	EQS	—	—

Figure 17: Service Request Enable Register (SRER)

Operation Enable Register (OENR)

The OENR consists of bits defined exactly the same as bits 0 through 15 in the OEVR (see Operation Event Register (OEVR) (on page 4-6)). You can use this register to control whether or not the Operation Status Bit (OSB) in the SBR is set when an event occurs and the corresponding OEVR bit is set.

Use the `STATus:OPERation:ENABle` command to set the bits in the OENR. Use the `STATus:OPERation:ENABle?` query to read the contents of the OENR.



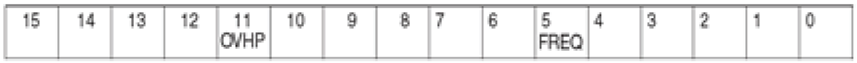
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							WTRIG (CH2)			WTRIG (CH1)		SWE			CAL

Figure 18: Operation Enable Register (OENR)

Questionable Enable Register (QENR)

The QENR consists of bits defined exactly the same as bits 0 through 15 in the QEVR register (see Questionable Event Register (on page 4-6)). You can use this register to control whether the QSB in the SBR is set when an event occurs and the corresponding QEVR bit is set.

Use the `STATus:QUEStionable:ENABle` command to set the bits in the QENR. Use the `STATus:QUEStionable:ENABle?` query to read the contents of the QENR.



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				OVHP						FREQ					

Figure 19: Questionable Enable Register (QENR)

Queues

There are two types of queues in the status reporting system: output queue and error/event queues.

Output Queue

The output queue is an FIFO (first-in, first-out) queue that holds response messages to queries awaiting retrieval. When there are messages in the queue, the message available (MAV) bit in the Status Byte Register (SBR) is set.

The output queue is emptied each time a command or query is received, so the controller must read the output queue before the next command or query is issued. If this is not done, an error occurs and the output queue is emptied. However, the operation proceeds even if an error occurs.

Error/Event Queue

The event queue is an FIFO (first-in, first-out) queue that stores events as they occur in the instrument. If more than 64 events are stored, the 64th event is replaced with event code -350 (“Queue Overflow”).

The oldest error code and text are retrieved by using one of the following queries:

```
SYSTem:ERRor[:NEXT]?
```

First, issue the `*ESR?` query to read the contents of the Status Event Status Register (SESR). The contents of the SESR are cleared after they are read. If an SESR bit is set, events are stacked in the Error/Event Queue. Retrieve the event code with the following command sequence:

```
*ESR?
```

```
SYSTem:ERRor[:NEXT]?
```

If you omit the `*ESR?` query, the SESR bit will remain set, even if the event disappears from the Error/Event Queue.

Messages and Codes

Error and event codes with negative values are SCPI standard codes. Error and event codes with positive values are unique to the AWG4162 Arbitrary Waveform Generator.

The following table lists event code definitions. When an error occurs, you can find its error class by checking for the code in the following tables. Events in these tables are organized by event class.

Definition of event codes

Event class	Code range	Description
No error	0	No event or status
Command errors	–100 to –199	Command syntax errors
Execution errors	–200 to –299	Command execution errors
Device-specific errors	–300 to –399	Internal device errors
Query errors	–400 to –499	System event and query errors
Power-on events	–500 to –599	Power-on events
User request events	–600 to –699	User request events
Request control events	–700 to –799	Request control events
Operation complete events	–800 to –899	Operation complete events
Extended device-specific errors	1 to 32767	Device dependent events
Reserved	Other than above	Not used

Command Errors

Shows the error messages generated by improper command syntax. Check that the command is properly formed and that it follows the rules in the Syntax and Commands (on page 2-1).

Command error messages

Error code	Error message
–100	Command error
–101	Invalid character
–102	Syntax error
–103	Invalid separator
–104	Data type error
–105	GET not allowed
–108	Parameter not allowed
–109	Missing parameter
–110	Command header error
–111	Header separator error
–112	Program mnemonic too long
–113	Undefined header
–114	Header suffix out of range
–115	Unexpected number of parameters
–120	Numeric data error
–121	Invalid character in number
–123	Exponent too large
–124	Too many digits
–128	Numeric data not allowed
–130	Suffix error
–131	Invalid suffix
–134	Suffix too long
–138	Suffix not allowed
–140	Character data error
–141	Invalid character data
–144	Character data too long
–148	Character data not allowed

Error code	Error message
–150	String data error
–151	Invalid string data
–158	String data not allowed
–160	Block data error
–161	Invalid block data
–168	Block data not allowed
–170	Expression error
–171	Invalid expression
–178	Expression data not allowed
–180	Macro error
–181	Invalid outside macro definition
–183	Invalid inside macro definition
–184	Macro parameter error

Execution Errors

Lists the errors that are detected during execution of a command.

Execution error messages

Error code	Error message
–200	Execution error
–201	Invalid while in local
–202	Settings lost due to RTL
–203	Command protected
–210	Trigger error
–211	Trigger ignored
–212	Arm ignored
–213	Init ignored
–214	Trigger deadlock
–215	Arm deadlock
–220	Parameter error
–221	Settings conflict
–222	Data out of range

Error code	Error message
–223	Too much data
–224	Illegal parameter value
–225	Out of memory
–226	Lists not same length
–230	Data corrupt or stale
–231	Data questionable
–232	Invalid format
–233	Invalid version
–240	Hardware error
–241	Hardware missing
–242	No external reference or sampling clock source
–250	Mass storage error
–251	Missing mass storage
–252	Missing media
–253	Corrupt media
–254	Media full
–255	Directory full
–256	File name not found
–257	File name error
–258	Media protected
–260	Expression error
–261	Math error in expression
–270	Macro error
–271	Macro syntax error
–272	Macro execution error
–273	Illegal macro label
–274	Macro parameter error
–275	Macro definition too long
–276	Macro recursion error
–277	Macro redefinition not allowed
–278	Macro header not found
–280	Program error

Error code	Error message
–281	Cannot create program
–282	Illegal program name
–283	Illegal variable name
–284	Program currently running
–285	Program syntax error
–286	Program runtime error
–290	Memory use error
–291	Out of memory
–292	Referenced name does not exist
–293	Referenced name already exists
–294	Incompatible type

Device-Specific Errors

Lists the device specific errors that can occur during arbitrary waveform generator operation. These errors may indicate that the instrument needs repair.

Device-specific error messages

Error code	Error message
–300	Device specific error
–310	System error
–311	Memory error
–312	PUD memory lost
–313	Calibration memory lost
–314	Save/recall memory lost
–315	Configuration memory lost
–320	Storage fault
–321	Out of memory
–330	Self-test failed
–340	Calibration failed
–350	Queue overflow
–360	Communication error
–361	Parity error in program message

Error code	Error message
–362	Framing error in program message
–363	Input buffer overrun
–365	Time out error

Query Errors

Lists the error codes that are returned in response to an unanswered query.

Query errors

Error code	Error message
–400	Query error
–410	Query INTERRUPTED
–420	Query UNTERMINATED
–430	Query DEADLOCKED
–440	Query UNTERMINATED after indefinite response

Power-On Events

These events occur when the instrument detects an off-state to on-state transition in its power supply.

Power-on events

Error code	Error message
–500	Power on

User Request Events

These events are not used in this instrument.

User request events

Error code	Error message
–600	User request

Request Control Events

These events are not used in this instrument.

Request control events

Error code	Error message
–700	Request control

Operation Complete Events

These events occur when instrument’s synchronization protocol, having been enabled by an `*OPC` command, completes all selected pending operations.

Operation complete events

Error code	Error message
–800	Operation complete

Device Errors

The following table lists the error codes that are unique to the AWG4162.

Device errors

Error code	Error message
–330	Self-test failed
–340	Calibration failed

You can find the detailed calibration and self-test logs at the following directory path: “X:\Tektronix\AWG4000\Basic\Log\”.

The calibration log file is named “CalLog_20xx-xx-x-xx-xx-xx.txt”. The self-test (diagnostic) log file is named “DiagLog_20xx-xx-x-xx-xx-xx.txt”.

Calibration and Self-Test Errors

The following table lists the error codes that are unique to AWG4000 in calibration and self-test logs under the directory “X:\Tektronix\AWG4000\Basic\Log\”.

Error code	Error message
1400000	The PC can't establish a connection with the SRGB
1401000	The USB chip can't read the nonvolatile memory
1402000	Is not possible to load the FW on the control FPGA: there are HW problems between the USB chip and the control FPGA
1403000	Is not possible to read/write control FPGA registers
1404000	Is not possible to read/write nonvolatile memory
1405000	Is not possible to load the FW on the channel A FPGA: there are HW problems between the USB chip and the channel A FPGA
1405010	Is not possible to read/write channel A FPGA registers
1405020	The channel A FPGA can't read/write its external memory
1405030	Is not possible to communicate with DAC 9739 via SPI
1405040	Is not possible to communicate with LTC2265 via SPI
1406000	Is not possible to load the FW on the channel A FPGA: there are HW problems between the USB chip and the channel A FPGA
1406010	Is not possible to read/write channel A FPGA registers
1406020	The channel A FPGA can't read/write its external memory
1406030	Is not possible to communicate with DAC 9739 via SPI
1406040	Is not possible to communicate with LTC2265 via SPI
1407000	The SW is trying to initialize the SRGB board
1408000	Indicates that there are problems on ADC LTC2265
1408010	Indicates that the FPGA PLL is not locked: perhaps there is a problem with clock input or clock chain
1408020	Indicates that the ADF4351 is not locked: perhaps there is a problem with clock input or clock chain (DCO)
1408031	The DC AMP output offset is out of tolerance
1408041	The DC AMP output VOCM is out of tolerance
1408051	The DC AMP output is out of tolerance
1408071	The DC DIRECT output differential offset is out of tolerance
1408081	The DC DIRECT output offset is out of tolerance
1408091	The DC DIRECT output vocm is out of tolerance

Error code	Error message
1408101	The DC DIRECT output is out of tolerance
1408112	The internal attenuator are out of tolerance
1408032	The DC AMP output offset is out of tolerance
1408042	The DC AMP output vocm is out of tolerance
1408052	The DC AMP output is out of tolerance
1408072	The DC DIRECT output differential offset is out of tolerance
1408082	The DC DIRECT output offset is out of tolerance
1408092	The DC DIRECT output VOCM is out of tolerance
1408102	The DC DIRECT output is out of tolerance
1408111	The internal attenuator is out of tolerance
1409000	Indicates that there are problems on ADC LTC2265
1409010	Indicates that the FPGA PLL is not locked: perhaps there is a problem on clock input or clock chain
1409020	Indicates that the ADF4351 is not locked: perhaps there is a problem on clock input or clock chain (DCO)
1409031	The DC AMP output offset is out of tolerance
1409041	The DC AMP output VOCM is out of tolerance
1409051	The DC AMP output is out of tolerance
1409071	The DC DIRECT output differential offset is out of tolerance
1409081	The DC DIRECT output offset is out of tolerance
1409091	The DC DIRECT output VOCM is out of tolerance
1409101	The DC DIRECT output is out of tolerance
1409111	The internal attenuator is out of tolerance
1409032	The DC AMP output offset is out of tolerance
1409042	The DC AMP output VOCM is out of tolerance
1409052	The DC AMP output is out of tolerance
1409072	The DC DIRECT output differential offset is out of tolerance
1409082	The DC DIRECT output offset is out of tolerance
1409092	The DC DIRECT output VOCM is out of tolerance
1409102	The DC DIRECT output is out of tolerance
1409112	The internal attenuator is out of tolerance

Programming Examples

The following two example programs demonstrate methods that you can use to control the arbitrary waveform generator.

- Example 1: Set up a Waveform Output (on page 5-1)
- Example 2: Waveform Transfer and Copy (on page 5-2)

The example programs are written in Microsoft® Visual Basic Version 6.0. The programs run on Microsoft Windows® PC-compatible systems equipped with TekVISA.

TekVISA is the Tektronix implementation of the VISA Application Programming Interface (API). TekVISA is industry-compliant software for writing interoperable instrument drivers in a variety of Application Development Environments (ADEs).

The example programs assume that the system recognizes the PC (external controller) resource name. Refer to the TekVISA manual for details about resource names.

Example 1: Set up a Waveform Output

This is a sample program for setting the arbitrary waveform generator outputs.

```
Private Sub Sample1_Click()
'
'Assign resource
'
Tvc1.Descriptor = "USB::0x0699::0x0354:C000002::INSTR"
'
'Initialize of device setting
'
Tvc1.WriteString ("*RST")
'
'Set CH1 output parameters
'
Tvc1.WriteString ("FUNCTION SIN") 'Set output waveform SIN
Tvc1.WriteString ("FREQUENCY 10E3") 'Set frequency 10kHz
Tvc1.WriteString ("VOLTAGE:AMPLITUDE 2.00") 'Set amplitude
2Vpp
Tvc1.WriteString ("VOLTAGE:OFFSET 1.00") 'Set offset 1V
Tvc1.WriteString ("PHASE:ADJUST 0DEG") 'Set phase 0_degree
'
'Set CH2 output parameters
'
Tvc1.WriteString ("SOURCE2:FUNCTION SIN") 'Set output
waveform SIN
Tvc1.WriteString ("SOURCE2:FREQUENCY 10E3") 'Set frequency
10kHz \
Tvc1.WriteString ("SOURCE2:VOLTAGE:AMPLITUDE 1.00") 'Set amplitude
1Vpp
Tvc1.WriteString ("SOURCE2:VOLTAGE:OFFSET 0.00") 'Set offset
0V
Tvc1.WriteString ("SOURCE2:PHASE:ADJUST 90DEG") 'Set phase
90degrees
'
```

```
'Save settings and output on
'
Tvc1.WriteString ("*SAV 1") 'Save settings to Custom1
Tvc1.WriteString ("*RCL 1") 'Recall settings from Custom1
'
End Sub
```

Example 2: Waveform Transfer and Copy

This is a sample program for sending an arbitrary waveform to the arbitrary waveform generator's Arb Buffer 1 and saving the contents of Arb Buffer 1 to custom1.

```
Private Sub Sample2_Click()
'
'Assign resource
'
Tvc1.Descriptor = "USB::0x0699::0x0354:C000002::INSTR"
'Initialization of device settings
'
Tvc1.WriteString ("*RST")'
'Make arbitrary block data (2000 Points)
'
Dim wave(4000) As Byte
'Leading edge (500 Points)
'
For i = 0 To 499
'Data range is from 0 to 16382
'
Data = i * Int(16382 / 500)
'AFG Data Format is big endian
'
High = Int(Data / 256)
Low = Data - (High * 256)
wave(2 * i) = High wave(2 * i + 1) = Low
Next i
For i = 500 To 799 'Part of High Level (800 Points)
Data = 16382 High = Int(Data / 256)
Low = Data - (High * 256)
wave(2 * i) = High
wave(2 * i + 1) = Low
Next i
For i = 800 To 999
'Trailing Edge (200 Points)
'
Data = (1000 - i) * Int(16382 / 200)
High = Int(Data / 256)
Low = Data - (High * 256)
wave(2 * i) = High
wave(2 * i + 1) = Low
Next i
For i = 1000 To 1999
'Part of Low Level (1000 Points)
'
Data = 0
High = Int(Data / 256)
Low = Data - (High * 256)
```

```
wave(2 * i) = High
wave(2 * i + 1) = Low
Next i
'
'Transfer waveform
' Transfer arbitrary block data to Arb Buffer 1
'
Tvc1.WriteString ("FUNCTION ARBB1")
'
Tvc1.SendEndEnabled = False
Tvc1.WriteString ("TRACE:DATA ARBB1,#44000")
Tvc1.SendEndEnabled = True
Tvc1.WriteByteArray (wave)
'
'Save contents of Arb Buffer 1 to Custom 1
'
Tvc1.WriteString ("*SAV 1")
'
'Set CH1 output parameters'
'Set frequency 8kHz
'
Tvc1.WriteString ("FREQUENCY 8K")
'Recall custom 1
'
Tvc1.WriteString ("*RCL 1")
'Set CH1 output on'
Tvc1.WriteString ("OUTPUT ON")
End Sub
```

SCPI Conformance Information

All commands in the arbitrary waveform generator are based on SCPI Version 1999.0. The following table lists the SCPI commands the arbitrary waveform generator supports.

SCPI conformance information

Command				Defined in SCPI 1999.9	Not defined in SCPI 1999.0
ABORt				√	
AFGControl	CSCopy			√	
CALibration(?)				√	
DATE					√
DIAGnostic(?)					√
FILEsystem	CATalog(?)				√
	COPY				√
	CWDirectory(?)				√
	DELete				√
	HARDdisk?				√
	LOCK(?)				√
	MDIRectory				√
	UDISk?				√
HCOPy	SDUMp	[:IMMediate]		√	
MEMory	RECall				
	SAVE				
MEMory	STATe	DELete			√
		LOCK(?)			√
		RECall	AUTo(?)		√
		VALid?			√
MMEMory	LOAD	STATe		√	
	STORe	STATe		√	
OUTPut[1|2]	IMPedance(?)			√	
	POLarity(?)			√	
	STATe(?)			√	

Command					Defined in SCPI 1999.9	Not defined in SCPI 1999.0
[SOURce[1|2]]	VOLTage	CONCurrent	[STATe](?)			√
	AM	STATe(?)			√	
		INTernal	FREQuency(?)		√	
			FUNCtion(?)			√
				EFILe(?)		√
		SOURce(?)			√	
		[DEPTh](?)			√	
	BURSt	MODE(?)				√
		NCYCles(?)				√
		TDELay(?)				√
		[STATe](?)				√
	COMBine	FEED(?)			√	
	FM	INTernal	FREQuency(?)		√	
			FUNCtion(?)			√
				EFILe(?)		√
		SOURce(?)			√	
		STATe(?)			√	
		[DEViation](?)			√	
	FREQuency	CENTer(?)			√	
		CONCurrent	[STATe](?)			√
		MODE(?)			√	
		SPAN(?)			√	
		STARt(?)			√	
		STOP(?)			√	
		[CW|FIXed](?)			√	
	FSKey	INTernal	RATE(?)			√
		SOURce(?)				√
		STATe(?)				√
		[FREQuency](?)				√
	FUNCtion	EFILe(?)				√
		RAMP	SYMMetry(?)			√

Command					Defined in SCPI 1999.9	Not defined in SCPI 1999.0
		[SHAPe](?)			√	
	PHASe	INITiate				√
		[ADJust](?)			√	
	PM	INTernal	FREQuency(?)		√	
			FUNCtion(?)			√
				EFILe(?)		√
		SOURce(?)			√	
		STATe(?)			√	
	PSK	FREQuency(?)				√
		PHASe	ADJust(?)			√
		SOURce(?)				√
		STATe(?)				√
	PULSe	DCYCle(?)			√	
		DELay(?			√	
		HOLD(?)			√	
		PERiod(?)			√	
		TRANsition	TRAiling(?)		√	
			[LEADing]		√	
		WIDTh(?)			√	
	PWM	INTernal	FREQuency(?)			√
			FUNCtion(?)			√
				EFILe(?)		√
		SOURce(?)				√
		STATe(?)				√
		[DEViation]	DCYCle(?)			√
	SWEep	HTIMe(?)				√
		MODE(?)			√	
		RTIMe(?)				√
		SPACing(?)			√	
		TIME(?)			√	
	VOLTage	LIMit	HIGH(?)		√	
			LOW(?)		√	

Command					Defined in SCPI 1999.9	Not defined in SCPI 1999.0
		UNIT(?)				√
		[LEVel]	[IMMediate]	HIGH(?)	√	
				LOW(?)	√	
				OFFSet(?)	√	
				[AMPLitude](?)	√	
		VOCM				√
SOURce<3|4>	POWer	[LEVel]	[IMMediate]	[AMPLitude](?)	√	
STATus	OPERation	[EVENt]?			√	
		CONDition?			√	
		ENABle(?)			√	
	PRESet				√	
	QUEStionable	[EVENt]?			√	
		CONDition?			√	
		ENABle(?)			√	
SYSTem	BEEPer	STATe(?)			√	
		[IMMediate]				√
	ERRor	[NEXT]?			√	
	KCLick	[STATe](?)				√
	KLOCk	[STATe](?)			√	
	ULANguage(?)					√
	VERSion?				√	
TRACe|DATA					√	
	[DATA](?)				√	
	POINts?				√	
	SAVE					√
	RECall					√
TRIGger	[SEQuence]	SLOPe(?)			√	
		SOURce(?)			√	
		TIMer(?)			√	
		[IMMediate]			√	

Command	**Defined in SCPI 1999.9**	**Not defined in SCPI 1999.0**
*CAL?		√
*CLS	√	
*ESE(?)	√	
*ESR?	√	
*IDN?	√	
*OPC(?)	√	
*RCL		√
*RST	√	
*SAV		√
*SRE(?)	√	
*STB?	√	
*TRG		√
*TST?	√	
*WAI	√	

Default Settings

The following table lists the settings that are restored when you press the front-panel Default Button or send the *RST SCPI command.

Default settings

Menu or System	**Output configuration**	**Default setting**
Function		Sine
Frequency		1.000 000 000 00 MHz
Amplitude		1.000 Vp-p
Offset		0 mV
Symmetry (Ramp)		50.0%
Duty (Pulse)		50.0%
Output Units		Vp-p
Output Impedance		50 Ω
Output Invert		Off
Output Noise Add		Off
Output High Limit		2.500 V
Output Low Limit		–2.500 V
VOCM		0 mV
	Modulation	
Modulation Waveform		10.00 kHz, Sine (except FSK, PSK)
Modulation Waveform		10.00 kHz, Square (FSK, PSK)
AM Depth		50.0%
FM Deviation		1.000 000 MHz
PM Deviation		90.0 °
FSK Hop Frequency		1.000 000 MHz
FSK Rate		10.000 000 0 kHz
PWM Deviation		5.0%
PSK Frequency		10.00 kHz
PSK Hop Phase		90.0 °

	Sweep	
Sweep Start Frequency		100.000 kHz
Sweep Stop Frequency		1.000 000 MHz
Sweep Time		10 ms
Sweep Hold Time		0 ms
Sweep Return Time		1 ms
Sweep Type		Linear
Sweep Mode		Repeat
Sweep Source		Internal
Trigger Slope		Positive
Trigger Interval		1.000 ms
Input Threshold		0.00 V
Step Num		1
	Burst	
Burst Mode		Triggered
Burst Count		5
Trigger Source		Internal
Trigger Delay		0.0 ps
Trigger Interval		1.000 000 ms
Slope		Positive
Input Threshold		0.00 V
	System-related settings	
Clock Reference		Internal
Ext Clock Rate		10 MHz

The front-panel Default button does not reset the following settings:

- Power-on settings
- Click tone
- Beeper
- Saved setup files
- Saved waveform files
- Calibration data

Index

Contact Information:

Australia* 1 800 709 465

Austria 00800 2255 4835

Balkans, Israel, South Africa and other ISE Countries +41 52 675 3777

Belgium* 00800 2255 4835

Brazil +55 (11) 3759 7627

Canada 1 800 833 9200

Central East Europe / Baltics +41 52 675 3777

Central Europe / Greece +41 52 675 3777

Denmark +45 80 88 1401

Finland +41 52 675 3777

France* 00800 2255 4835

Germany* 00800 2255 4835

Hong Kong 400 820 5835

India 000 800 650 1835

Indonesia 007 803 601 5249

Italy 00800 2255 4835

Japan 81 (3) 6714 3010

Luxembourg +41 52 675 3777

Malaysia 1 800 22 55835

Mexico, Central/South America and Caribbean 52 (55) 56 04 50 90

Middle East, Asia, and North Africa +41 52 675 3777

The Netherlands* 00800 2255 4835

New Zealand 0800 800 238

Norway 800 16098

People's Republic of China 400 820 5835

Philippines 1 800 1601 0077

Poland +41 52 675 3777

Portugal 80 08 12370

Republic of Korea +82 2 6917 5000

Russia / CIS +7 (495) 6647564

Singapore 800 6011 473

South Africa +41 52 675 3777

Spain* 00800 2255 4835

Sweden* 00800 2255 4835

Switzerland* 00800 2255 4835

Taiwan 886 (2) 2656 6688

Thailand 1 800 011 931

United Kingdom / Ireland* 00800 2255 4835

USA 1 800 833 9200

Vietnam 12060128

*** European toll-free number. If not accessible, call:** +41 52 675 3777

Find more valuable resources at TEK.COM

077-1198-00