

Series 2200

Multichannel Programmable DC Power Supplies

Programming Technical Reference

2220S-907-01 Rev. A / June 2012



KEITHLEY

A GREATER MEASURE OF CONFIDENCE

Series 2200

Multichannel Programmable DC Power Supplies

Programming Technical Reference

© 2012, Keithley Instruments, Inc.

Cleveland, Ohio, U.S.A.

All rights reserved.

Any unauthorized reproduction, photocopy, or use the information herein, in whole or in part, without the prior written approval of Keithley Instruments, Inc. is strictly prohibited.

All Keithley Instruments product names are trademarks or registered trademarks of Keithley Instruments, Inc. Other brand names are trademarks or registered trademarks of their respective holders.

Document number: 2220S-907-01 Rev. A / June 2012

The following safety precautions should be observed before using this product and any associated instrumentation. Although some instruments and accessories would normally be used with nonhazardous voltages, there are situations where hazardous conditions may be present.

This product is intended for use by qualified personnel who recognize shock hazards and are familiar with the safety precautions required to avoid possible injury. Read and follow all installation, operation, and maintenance information carefully before using the product. Refer to the user documentation for complete product specifications.

If the product is used in a manner not specified, the protection provided by the product warranty may be impaired.

The types of product users are:

Responsible body is the individual or group responsible for the use and maintenance of equipment, for ensuring that the equipment is operated within its specifications and operating limits, and for ensuring that operators are adequately trained.

Operators use the product for its intended function. They must be trained in electrical safety procedures and proper use of the instrument. They must be protected from electric shock and contact with hazardous live circuits.

Maintenance personnel perform routine procedures on the product to keep it operating properly, for example, setting the line voltage or replacing consumable materials. Maintenance procedures are described in the user documentation. The procedures explicitly state if the operator may perform them. Otherwise, they should be performed only by service personnel.

Service personnel are trained to work on live circuits, perform safe installations, and repair products. Only properly trained service personnel may perform installation and service procedures.

Keithley Instruments products are designed for use with electrical signals that are rated Measurement Category I and Measurement Category II, as described in the International Electrotechnical Commission (IEC) Standard IEC 60664. Most measurement, control, and data I/O signals are Measurement Category I and must not be directly connected to mains voltage or to voltage sources with high transient overvoltages. Measurement Category II connections require protection for high transient overvoltages often associated with local AC mains connections. Assume all measurement, control, and data I/O connections are for connection to Category I sources unless otherwise marked or described in the user documentation.

Exercise extreme caution when a shock hazard is present. Lethal voltage may be present on cable connector jacks or test fixtures. The American National Standards Institute (ANSI) states that a shock hazard exists when voltage levels greater than 30 V RMS, 42.4 V peak, or 60 VDC are present. A good safety practice is to expect that hazardous voltage is present in any unknown circuit before measuring.

Operators of this product must be protected from electric shock at all times. The responsible body must ensure that operators are prevented access and/or insulated from every connection point. In some cases, connections must be exposed to potential human contact. Product operators in these circumstances must be trained to protect themselves from the risk of electric shock. If the circuit is capable of operating at or above 1000 V, no conductive part of the circuit may be exposed.

Do not connect switching cards directly to unlimited power circuits. They are intended to be used with impedance-limited sources. NEVER connect switching cards directly to AC mains. When connecting sources to switching cards, install protective devices to limit fault current and voltage to the card.

Before operating an instrument, ensure that the line cord is connected to a properly-grounded power receptacle. Inspect the connecting cables, test leads, and jumpers for possible wear, cracks, or breaks before each use.

When installing equipment where access to the main power cord is restricted, such as rack mounting, a separate main input power disconnect device must be provided in close proximity to the equipment and within easy reach of the operator.

For maximum safety, do not touch the product, test cables, or any other instruments while power is applied to the circuit under test. ALWAYS remove power from the entire test system and discharge any capacitors before: connecting or disconnecting cables or jumpers, installing or removing switching cards, or making internal changes, such as installing or removing jumpers.

Do not touch any object that could provide a current path to the common side of the circuit under test or power line (earth) ground. Always make measurements with dry hands while standing on a dry, insulated surface capable of withstanding the voltage being measured.


The instrument and accessories must be used in accordance with its specifications and operating instructions, or the safety of the equipment may be impaired.


Do not exceed the maximum signal levels of the instruments and accessories, as defined in the specifications and operating information, and as shown on the instrument or test fixture panels, or switching card.


When fuses are used in a product, replace with the same type and rating for continued protection against fire hazard.

Chassis connections must only be used as shield connections for measuring circuits, NOT as safety earth ground connections.

If you are using a test fixture, keep the lid closed while power is applied to the device under test. Safe operation requires the use of a lid interlock.


If a  screw is present, connect it to safety earth ground using the wire recommended in the user documentation.

The  symbol on an instrument means caution, risk of danger. The user should refer to the operating instructions located in the user documentation in all cases where the symbol is marked on the instrument.

The  symbol on an instrument means caution, risk of electric shock. Use standard safety precautions to avoid personal contact with these voltages.

The  symbol on an instrument shows that the surface may be hot. Avoid personal contact to prevent burns.

The  symbol indicates a connection terminal to the equipment frame.

If this  symbol is on a product, it indicates that mercury is present in the display lamp. Please note that the lamp must be properly disposed of according to federal, state, and local laws.

The **WARNING** heading in the user documentation explains dangers that might result in personal injury or death. Always read the associated information very carefully before performing the indicated procedure.

The **CAUTION** heading in the user documentation explains hazards that could damage the instrument. Such damage may invalidate the warranty.

Instrumentation and accessories shall not be connected to humans.

Before performing any maintenance, disconnect the line cord and all test cables.

To maintain protection from electric shock and fire, replacement components in mains circuits — including the power transformer, test leads, and input jacks — must be purchased from Keithley Instruments. Standard fuses with applicable national safety approvals may be used if the rating and type are the same. Other components that are not safety-related may be purchased from other suppliers as long as they are equivalent to the original component (note that selected parts should be purchased only through Keithley Instruments to maintain accuracy and functionality of the product). If you are unsure about the applicability of a replacement component, call a Keithley Instruments office for information.

To clean an instrument, use a damp cloth or mild, water-based cleaner. Clean the exterior of the instrument only. Do not apply cleaner directly to the instrument or allow liquids to enter or spill on the instrument. Products that consist of a circuit board with no case or chassis (e.g., a data acquisition board for installation into a computer) should never require cleaning if handled according to instructions. If the board becomes contaminated and operation is affected, the board should be returned to the factory for proper cleaning/servicing.

Table of Contents

Preface	iii
Welcome	iii
Products	iii
Extended Warranty	iii
Contact Information	iv

Getting Started

Getting Started	1-1
Using the USB	1-1
Command Timing	1-1
Command Syntax	2-1
Command and Query Structure	2-1
Command Entry	2-3
Command Groups	2-7
Status Commands	2-7
Save and Recall Commands	2-8
System Commands	2-9
Diagnostic Commands	2-9
Synchronization Commands	2-9
Trigger Commands	2-10
Measurement Commands	2-11
Source Commands	2-11
Channel Combination Commands	2-12
Display Commands	2-12
Commands Listed in Alphabetical Order	2-13

Status and Events

Status and Events	3-1
Status Reporting Structure	3-1
Registers	3-3
Queues	3-9
Messages and Codes	3-10

Appendices

Appendix A: ASCII Code Chart	A-1
Appendix B: Programming Examples	B-1
Appendix C: Default Setup	C-1

Preface

Welcome

Thank you for using a Keithley Instruments product. The Series 2200 Programmable Multichannel DC Power Supplies are flexible DC sources designed to power a wide range of applications. The model 2230-30-1 offers three power channels and the model 2220-30-1 provides two channels. The output channels on both models are independent and isolated, allowing you to power circuits with different references or polarities. Each channel can be enabled or disabled as your application requires. All outputs feature remote sense capability which can be used to reduce the effect of lead resistance, delivering 0.03% basic voltage accuracy even when using long leads. Basic current accuracy is 0.1% for all channels and linear regulation delivers low noise – less than 3 mVp-p. Flexible display modes make it easy to use the two 30 V outputs in combination, and the USB interface makes it easy to build PC-based systems without converters or special cables.

Products

This programmer manual provides commands, and explains the use of those commands, for remotely controlling the following instruments. With this information, you can write computer programs to perform functions, such as setting the controls, taking measurements, performing statistical calculations, and exporting data for use in other programs.

Model	Description
2220-30-1	Dual Channel Programmable DC Power Supply
2220J-30-1	Dual Channel Programmable DC Power Supply for Japan
2230-30-1	Triple Channel Programmable DC Power Supply
2230J-30-1	Triple Channel Programmable DC Power Supply for Japan

Extended Warranty

Additional years of warranty coverage are available on many products. These valuable contracts protect you from unbudgeted service expenses and provide additional years of protection at a fraction of the price of a repair. Extended warranties are available on new and existing products. Contact your local Keithley Instruments representative for details.

Contact Information

If you have any questions after reviewing this information, please use the following sources:

1. Keithley Instruments website (<http://www.keithley.com>)
2. Keithley web forum (<http://forum.keithley.com>)
3. Call Keithley Instruments corporate headquarters (toll-free inside the U.S. and Canada only) at 1-888-KEITHLEY (1-888-534-8453), or from outside the U.S. at +1-440-248-0400. For worldwide contact numbers, visit the Keithley Instruments website (<http://www.keithley.com>).

Getting Started

Getting Started

Your power supply has a USB 2.0 high-speed device port to control the power supply using the USBTMC protocol. The USBTMC protocol allows USB devices to communicate using IEEE-488.2 style messages.

Using the USB

Start by connecting an appropriate USB cable between the USB 2.0 high-speed device port on the rear panel of your power supply and a PC.

In order for the PC to recognize the power supply, a USBTMC driver must be installed on the PC. A USBTMC driver can be installed on your PC by installing a virtual instrument communications API like NIVISA. This VISA is available for download from the Keithley or National Instruments Web sites. Once the USBTMC driver is loaded, your PC will establish communication with the power supply upon USB cable connection.

For further remote control and/or programming use, other software applications may be needed in addition to a VISA and the USBTMC driver.

Command Timing

The average time it takes to both send and receive every command is approximately 20 ms. In the case of more complex commands, more time may be required to complete transmission.

Command Syntax

You can control the power supply through the USB interface using commands and queries.

This section describes the syntax these commands and queries use and the conventions the power supply uses to process them. The commands and queries themselves are listed by group and alphabetically. (See page 2-7, *Command Groups*.)

You transmit commands to the power supply using the enhanced American Standard Code for Information Interchange (ASCII) character encoding. *Appendix A* contains a chart of the ASCII character set.

The Backus Naur Form (BNF) notation is used in this manual to describe commands and queries. (See Table 2-1.)

Table 2-1: BNF notation

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 simply called commands and queries). Commands change power supply settings or perform a specific action. Queries cause the power supply to return data and information about its status.

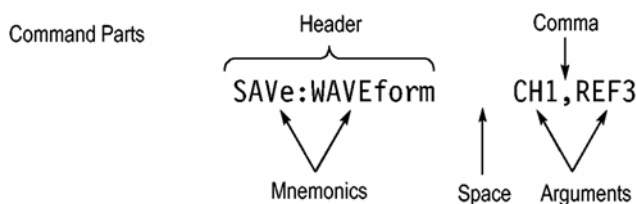
Most commands have both a set form and a query form. The query form of the command is the same as the set form except that it ends with a question mark. For example, the set command STATUS:OPERation:ENable has a query form STATUS:OPERation:ENable?. Not all commands have both a set and a query form; some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

A command message is a command or query name, followed by any information the power supply needs to execute the command or query. Command messages consist of five different element types. (See Table 2-3.)

Table 2-2: 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.
<Argument>	A quantity, quality, restriction, or limit associated with the header. Not all commands have an argument, while other commands have multiple arguments. Arguments are separated from the header by a <Space>. Arguments are separated from each other by a <Comma>.
<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 2-1: Command message elements**

Commands

Commands cause the power supply 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 off 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 power supply 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.

Query Responses

When a query is sent to the power supply, only the values are returned. When the returned value is a mnemonic, it is noted in abbreviated format, as shown in the following table.

Table 2-3: Query response examples

Query	Response
MEASure:VOLTage:DC?	5.0011
SOURce:FUNCTion:MODE?	LIST

Command Entry

Follow these general rules when entering commands:

- Enter commands in upper or lower case.
- 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 power supply ignores commands that consists of just a combination of white space characters and line feeds.

SCPI Commands and Queries

The power supply 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 that represents a subsystem. The top level of the tree is the root node; it is followed by one or more lower-level nodes. (See Figure 2-2.)

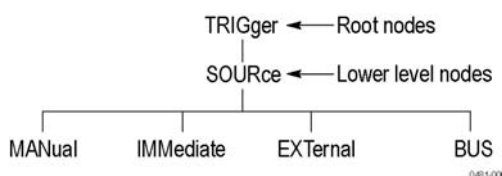


Figure 2-2: Example of SCPI subsystem hierarchy tree

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.

Message Terminators

This manual uses the term <EOM> (End of message) to represent a message terminator.

USB End of Message (EOM) terminators. See the USB Test and Measurement Class Specification (USBTMC) section 3.2.1 for details. The power supply terminates messages by setting the EOM bit in the USB header of the last transfer of a message to the host (USBTMC Specification section 3.3.1), and by terminating messages with a LF.

When receiving, the power supply expects a LF and an asserted EOM bit as a message terminator.

Parameter Types

Many power supply commands require parameters. Parameters are indicated by angle brackets, such as <file_name>. There are several different types of parameters, as listed in the following table. The parameter type is listed after the parameter. Some parameter types are defined specifically for the power supply command set and some are defined by SCPI. (See Table 2-4.)

Table 2-4: Types of parameters

Parameter type	Description	Example
boolean	Boolean numbers or values	ON or $\neq 0$ OFF or 0
discrete	A list of specific values	MIN, MAX
NR1 numeric	Integers	0, 1, 15, -1
NR2 numeric	Decimal numbers	1.2, 3.141516, -6.5
NR3 numeric	Floating point numbers	3.1415E-9, -16.1E5
NRf numeric	Flexible decimal number that may be type NR1, NR2, or NR3	See NR1, NR2, NR3 examples in this table
string	Alphanumeric characters (must be within quotation marks)	"Testing 1, 2, 3"

Abbreviating Commands, Queries, and Parameters

You can abbreviate most SCPI commands, queries, and parameters to an accepted short form. This manual shows these commands as a combination of upper and lower case letters. The upper case letters indicate the accepted short form of a command, as shown in the following figure. The accepted short form and the long form are equivalent and request the same action of the instrument.



Figure 2-3: Example of abbreviating a command

Chaining Commands and Queries

You can chain several commands or queries together into a single message. To create a chained message, first create a command or query, then add a semicolon (;), and finally add more commands or queries and semicolons until you are done. 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.

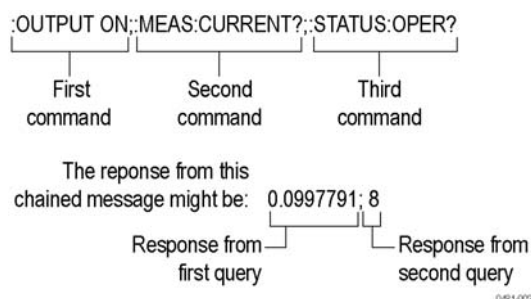


Figure 2-4: Example of chaining commands and queries

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 (STAT:QUES) as the first command, so these nodes can be omitted.

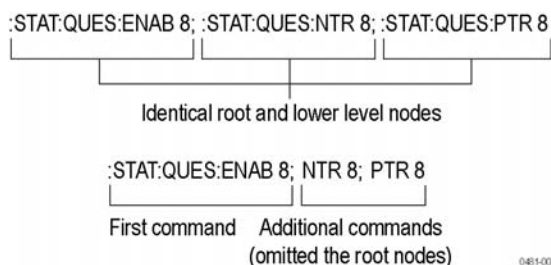


Figure 2-5: Example of omitting root and lower level nodes

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 upper case, lower case, or a mixture of both cases for all commands, queries, and parameters.

:SOURCE:FREQUENCY 10MHZ

is the same as

:source:frequency 10mhz

and

:SOURCE:frequency 10MHZ

NOTE. *Quoted strings are case sensitive.*

- No embedded spaces are allowed between or within nodes.

correct	:OUTPUT:FILTER:LPASS:FREQUENCY 200MHZ
---------	---------------------------------------

incorrect	:OUTPUT: FILTER: LPASS:FREQUENCY 200MHZ
-----------	---

Command Groups

This manual lists the power supply commands in two ways. First, it presents them by functional groups. Then, it lists them alphabetically. The functional group list starts below. The alphabetical list provides detail on each command. (See page 2-13.)

The power supply interface conforms to Keithley standard codes and formats except where noted. The GPIB interface also conforms to IEEE Std 488.2–1987 except where noted. The USB interface also conforms to USB Test and Measurement Class, Subclass USB488 Specification, except where noted. Arguments are not mentioned in the group command descriptions, but are listed under the commands in the *Commands Listed in Alphabetical Order* section of this manual. (See page 2-13.)

Status Commands

Status commands let you determine the status of the power supply and control events.

Several commands and queries are common to all devices on the GPIB or USB bus. These commands and queries are defined by IEEE Std. 488.2-1987 and Tek Standard Codes and Formats 1989, and begin with an asterisk (*) character.

Table 2-5: Status commands

Command	Description
*CLS	Clear all event registers and queues.
*ESE	Set/query standard event status enable register.
*ESR?	Return standard event status register.
*IDN?	Return identification information in IEEE 488.2 notation.
*RST	Resets to known settings, but does not purge stored settings.
*PSC	Set/query power-on status clear.
*SRE	Set/query service request enable register.
*STB?	Read status byte.
STATus:QUESTionable:INSTrument[:EVENT]?	Return questionable event register.
STATus:QUESTionable:INSTrument:ENABle	Set/query questionable enable register. This parameter determines which bit of the quest event register is set to 1. If a QUES condition changes, the QUES bit of status byte register will be set to 1.
STATus:QUESTionable:INSTrument:ISUMmary<x>[:EVENT]?	Return questionable event register summary for channel x, where <x> is 1, 2, or 3.
STATus:QUESTionable:INSTrument:ISUMmary<x>:ENABle	Set/query questionable enable register summary for channel x, where <x> is 1, 2, or 3. This parameter determines which bit of the quest event register is set to 1. If a QUES condition changes, the QUES bit of status byte register will be set to x.

Table 2-5: Status commands (cont.)

Command	Description
STATus:QUESTionable:INSTrument:ISUMmary<x>:CONDition?	Return questionable condition register summary for channel x, where <x> is 1, 2, or 3. When a bit of the quest condition changes, the corresponding bit value in the quest event register will be set to x.
STATus:QUESTionable:ENABLE	Set/query questionable enable register. This parameter determines which bit of the quest event register is set to 1. If a QUES condition changes, the QUES bit of status byte register will be set to 1.
STATus:QUESTionable[:EVENT]?	Return questionable event register.
STATus:OPERation:INSTrument[:ENABLE]?	Set/query operation enable register. The parameter determines which bit value of quest event register is set to 1. If a OPER condition changes, the OPER bit of the status byte register will be set to 1.
STATus:OPERation:INSTrument[:EVENT]?	Return operation event register.
STATus:OPERation:INSTrument[:EVENT]?	Queries the contents of the operation instrument event register (OIEVR).
STATus:OPERation:INSTrument[:ENABLE]?	Queries the contents of the operation instrument enable register (OIENR).
STATus:OPERation:INSTrument:ISUMmary<x>[:EVENT]?	Return operation event register for channel x, where <x> is 1, 2, or 3.
STATus:OPERation:INSTrument:ISUMmary<x>:ENABLE	Set/query operation enable register for channel x, where <x> is 1, 2, or 3. The parameter determines which bit value of quest event register is set to 1. If a OPER condition changes, the OPER bit of the status byte register will be set to 1.
STATus:OPERation:INSTrument:ISUMmary<x>:CONDition?	Return operation condition register for channel x, where <x> is 1, 2, or 3. When a parameter of the operation condition register changes, the corresponding bit in the operation event register will be set to 1.

Save and Recall Commands

Save and recall commands allow you to save the active settings to one of the settings memories within the power supply, and recall those settings at a later time.

Table 2-6: Save and recall commands

Header	Description
*SAV	Save instrument setting to setup memory
*RCL	Recall instrument setting from setup memory

System Commands

Table 2-7: System commands

Header	Description
SYSTem:POSetup	Set or query power-on parameters
SYSTem:MODUle?	Queries the module of the power supply
SYSTem:VERSion?	Return SCPI version information
SYSTem:MODUle?	Return error code and error information
SYSTem:KEY	Set or query key operation
SYSTem:REMOte	Set or query remote mode
SYSTem:RWLock	Set to remote mode and lock front-panel
SYSTem:LOCal	Set to front-panel control mode
	Set or query key beep sound on or off

Diagnostic Commands

The power supply includes a self test function that may be used to confirm that it is functioning as expected. A table of error codes that may be returned by the self test are given in the *Messages and Codes* section. (See page 3-10.)

Table 2-8: Diagnostic commands

Header	Description
*TST?	Perform self-test and return result status

Synchronization Commands

Table 2-9: Synchronization commands

Header	Description
*OPC	Set/query operation complete
*WAI	Wait to continue

Trigger Commands

Trigger commands are used to determine the timing of list mode sequences.

Table 2-10: Trigger commands

Header	Description
TRIGger[:IMMediate]	Forces an immediate trigger event.
*TRG	Generates a trigger event.
[SOURce:]VOLTage:TRIGgered[:IMMediate]	Set or query the trigger voltage.
[SOURce:]CURRent:TRIGgered[:IMMediate]	Set or query the trigger current.
INSTrument:COUPle[:TRIGger]	Set or query the channel that will respond the trigger command.

Measurement Commands

Measurement commands are used to query parameters. The MEASure commands initiate and execute a complete measurement cycle and are recommended for measuring voltage and current at the outputs of the power supply. FETCh commands do not initiate a new measurement cycle but rely on measurements stored in the communication buffers of the power supply. The FETCh commands are provided for voltage and current measurements to maintain compatibility with other instruments. Output power, however, is only available using a FETCh command.

Table 2-11: Measurement commands

Command	Description
MEASure[:SCALar][:VOLTage][:DC]?	Initiate a measurement and query the measured output voltage
MEASure[:SCALar]:POWer[:DC]?	Initiate a measurement and query the measured output current
MEASure[:SCALar]:CURRent[:DC]?	Initiate a measurement and query the measured output current
FETCh[:SCALar]:CURRent[:DC]?	Query the output current stored in the communications buffer
FETCh[:SCALar]:POWer[:DC]?	Query the output power stored in the communications buffer
FETCh[:SCALar]:VOLTage[:DC]?	Query the output voltage stored in the communications buffer

Source Commands

These commands allow you to set various output parameters. Some of the commands are used to configure protection functions like output timers and Max Voltage.

Table 2-12: Source commands

Command	Description
[SOURce:]CURRent[:LEVel][:IMMediate][:AMPLitude]	Set or query the current value in units of A or mA.
[SOURce:]CURRent[:LEVel]:UP[:IMMediate][:AMPLitude]	Set the current level to increase a step.
[SOURce:]CURRent[:LEVel]:DOWN[:IMMediate][:AMPLitude]	Set the current level to decrease a step.
[SOURce:]CURRent[:LEVel][:IMMediate]:STEP[:INCRement]	Set or query the current step value.
[SOURce:]OUTPut:TIMer:STATe]	Set or query the state of the output timer.
[SOURce:]OUTPut:STATe[:ALL]	Set or query power supply output on or off.
[SOURce:]OUTPut:TIMer:DELay	Set or query the time duration of output timer.
[SOURce:]VOLTage[:LEVel][:IMMediate][:AMPLitude]	Set or query voltage level.
[SOURce:]VOLTage[:LEVel]:UP[:IMMediate][:AMPLitude]	Set to increase the voltage level by a step.
[SOURce:]VOLTage[:LEVel]:DOWN[:IMMediate][:AMPLitude]	Set to decrease the voltage level by a step.
[SOURce:]VOLTage[:LEVel][:IMMediate]:STEP[:INCRement]	Set or query the voltage step value.
[SOURce:]VOLTage:LIMit:STATe	Set or query enable/disable voltage limit function.
[SOURce:]VOLTage:LIMit[:LEVel]	Set or query the maximum output voltage setting.

Table 2-12: Source commands (cont.)

Command	Description
[SOURce:]OUTPut:ENABle	Set or query the current channel as enabled or disabled.
[SOURce:]CHANnel:OUTPut:[STATe]	Set or query the output status of the current channel.
[SOURce:]APPlY	Sets voltage and current level and switch channels at the same time.
[SOURce:]OUTPut:PARAllel[:STATe]	Sets the parallel state of CH1 and CH2.
[SOURce:]OUTPut:SERIES	Sets the serial state of CH1 and CH2.
[SOURce:]OUTPut:PON[:STATe]	Sets the power supply to power up with its output turned off, or to return the output to the state it was in when it powered down.

Channel Combination Commands

These commands allow you to set various channel combinations. The commands you can use depends on the number of channels your instrument has.

Table 2-13: Channel combination commands

Command	Description
INSTrument:SElect	Switch or query the current channel.
INSTrument:COMbine:SERIES	Set CH1 and CH2 in series.
INSTrument:COMbine:PARAllel	Set CH1 and CH2 in parallel.
INSTrument:COMbine:TRACk	Set CH1 and CH2 to track.
INSTrument:COMbine:OFF	Remove the combination of channels.
INSTrument:COMbine?	Query which channels are combined.

Display Commands

Display commands are used to clear or show particular strings on the instrument display.

Table 2-14: Display commands

Header	Description
DISPlay[:WINDow][:STATe]	Set or query the display state.
DISPlay[:WINDow]:TEXT[:DATA]	Set or query the display to show a particular string.
DISPlay[:WINDow]:TEXT:CLEAr	Set to clear the characters on the display and returns the display to normal mode.

Commands Listed in Alphabetical Order

You can use commands to either set instrument features or query instrument values. You can use some commands to do both, some only to set and some only to query. This document marks set-only commands with the words “No Query Form” included with the command name. It marks query-only commands with a question mark appended to the header, and includes the words “Query Only” in the command name.

This document spells out headers, mnemonics, and arguments with the minimal spelling shown in uppercase. For example, to use the abbreviated form of the MEASure:SCALar:VOLTage:DC? command, type MEAS:SCAL:VOLT:DC?.

*CLS (No Query Form)

The *CLS command clears all event registers and queues.

Group Status

Syntax *CLS

Related Commands [*ESR?](#), [*STB?](#)

DISPlay[:WINDow][:STATe]

Sets or queries the state of the instrument display.

Group Display

Syntax DISPlay[:WINDow][:STATe] {0|1|ON|OFF}
DISPlay[:WINDow][:STATe]?

Arguments 0|1|ON|OFF

DISPlay[:WINDow]:TEXT[:DATA]

Sets or queries the state of the instrument display to show a particular string.

Group Display

Syntax `DISPlay[:WINDow]:TEXT[:DATA] <string>`
`DISPlay[:WINDow]:TEXT[:DATA]?`

Arguments String with quotes. 48 character length limit.

DISPlay[:WINDow]:TEXT:CLEAr (No Query Form)

Clears the characters on the display and then returns the display to normal mode.

Group Display

Syntax `DISPlay[:WINDow]:TEXT:CLEAr`

Arguments None

*ESE

Sets and queries the bits in the Event Status Enable Register (ESER). The ESER is an eight-bit mask register that determines which bits in the Standard Event Status Register (SESR) will set the ESB bit in the Status Byte Register (SBR). (See page 3-1, *Status and Events*.)

Group Status

Syntax `*ESE <mask>`
`*ESE?`

Related Commands [*CLS](#), [*ESR?](#)

Arguments `<mask> ::= <NR1>` where:
`<NR1>` is a value in the range from 0 through 255. The binary bits of the ESER are set according to this value.

The power-on default for ESER is 0 if *PSC is 1. If *PSC is 0, the ESER maintains its value through a power cycle.

Examples `*ESE 145` sets the ESER to binary 10010001, which enables the PON, EXE, and OPC bits.

*ESE might return the string *ESE 186, showing that the ESER contains the binary value 10111010.

*ESR? (Query Only)

Returns the contents of the Standard Event Status Register (SESR). *ESR? also clears the SESR (since reading the SESR clears it). (See page 3-1, *Status and Events*.)

Group Status

Syntax *ESR?

Related Commands [*CLS](#), [*OPC](#), [*SRE](#),

Returns <NR1>, which is a decimal representation of the contents of the Standard Event Status Register (SESR).

Examples *ESR? might return the value 149, showing that the SESR contains binary 10010101.

FETCh[:SCALar]:CURRent[:DC]? (Query Only)

This command returns the last measured output current stored in the communications buffer of the power supply. A new measurement is not initiated by this command.



CAUTION. Using this FETCh command may return an old result, which could adversely affect the accuracy of your test. In most cases, using the MEASure command is recommended. The benefit of the FETCh command is that it provides a result a bit more quickly than the MEASure command.

Group Measurement

Syntax FETCh[:SCALar]:CURRent[:DC]? [CH1|CH2|CH3|ALL]

Related Commands [MEASure\[:SCALar\]:POWer\[:DC\]?](#)

Returns <NR2>, which gives is the measured output current in amperes.

Examples FETC:CURR? might return 0.09998, which would be the current measured at the output of the power supply in amperes.

FETCh[:SCALar]:VOLTage[:DC]? (Query Only)

This command returns the last measured output voltage stored in the communications buffer of the power supply. A new measurement is not initiated by this command.

Group Measurement

Syntax FETCh[:SCALar]:VOLTage[:DC]? [CH1|CH2|CH3|ALL]

Related Commands [MEASure\[:SCALar\]\[:VOLTage\]\[:DC\]?](#)

Returns <NR2> is the measured output voltage in volts.

Examples FETC:VOLT? might return 5.0011, which would be the measured voltage across the power supply outputs in volts.

FETCh[:SCALar]:POWer[:DC]? (Query Only)

This command returns the calculated power based on the last measured output voltage and current. A new measurement is not initiated by this command. The power calculation in the instrument is performed approximately every 100 ms. Insure that the voltage and current are stable longer than this for good results.

Group Measurement

Syntax FETCh[:SCALar]:POWer[:DC]? [CH1|CH2|CH3|ALL]

Returns <NR2> is the measured output power in watts.

Examples FETCh:POW? might return 6.01667, which would be the power measured at the output of the power supply in watts.

*IDN? (Query Only)

Returns the power supply identification code in IEEE 488.2 notation.

Group Status

Syntax *IDN?

Returns A string that includes <manufacturer>, <model>, <serial number>, and <firmware_version> as defined in the following table.

<manufacturer>	<model>	<serial number>	<firmware_version>
keithley	22XXX	XXXXXX	X. XX-X. XX

Examples *IDN?
might return the following response for a 2220-30-1:
KEITHLEY , 2220-30-1 , 000004 , 1.01-1.20

INSTrument:COMbine? (Query Only)

This command queries the instrument to determine the combination state of channels 1 and 2.

Group Channel

Syntax INSTrument:COMbine?

Related Commands [INSTrument:COMbine:OFF](#)

Returns Series for series combination.
Parallel for parallel combination.
NONE for combination off.

INSTrument:COMbine:OFF

This command is used to turn off series, parallel, or tracking mode, and return channels 1 and 2 to independent operation.

Group Instrument

Syntax `INSTrument:COMbine:OFF`

Related Commands [INSTrument:COMbine?](#)

Examples `INSTRUMENT:COMBINE:OFF`

INSTrument:COMbine:PARAllel (No Query Form)

This command sets CH1 and CH2 into parallel mode. This mode assumes that CH1 and CH2 have been wired in parallel external to the power supply. The combined channel should be referred to as CH1 and the current for the combined channel may be set as high as 3A. Voltage will be set to the same value for both channels. The `measure:current` command will respond with the combined current.

Group Channel

Syntax `INSTrument:COMbine:PARAllel`

Related Commands [INSTrument:COMbine:OFF](#), [INSTrument:COMbine:SERies](#), [INSTrument:COMbine:TRACK](#),

Examples `INSTRUMENT:COMBINE:PARALLEL`

INSTrument:COMbine:SERies (No Query Form)

This command sets CH1 and CH2 into series mode. This mode assumes that CH1 and CH2 have been wired in series external to the power supply. The combined channel should be referred to as CH1 and the voltage for the combined channel may be set as high as 60 V. The current limit will be set to the same value for both channels. The `measure:voltage` command will respond with the combined voltage.

Group Channel

Syntax `INSTrument:COMbine:SERies`

Related Commands [INSTrument:COMbine:OFF](#), [INSTrument:COMbine:PARAllel](#),
[INSTrument:COMbine:TRACk](#)

Examples `INSTrument:COMbine:SERIES`

INSTrument:COMbine:TRACk (No Query Form)

This command sets CH1 and CH2 in track mode. In this mode, the ratio of CH1 to CH2 voltage that is set before sending the command will be maintained for subsequent voltage settings.

Group Channel

Syntax `INSTrument:COMbine:TRACk`

Related Commands [INSTrument:COMbine:OFF](#), [INSTrument:COMbine:PARAllel](#),
[INSTrument:COMbine:SERIES](#)

Examples `INSTrument:COMbine:TRACk`

INSTrument:COUPle[:TRIGger]

This command is used to determine which channels will respond to the trigger command.

Group Channel

Syntax `INSTrument:COUPle[:TRIGger] {CH1|CH2|CH3}`
`INSTrument:COUPle[:TRIGger]?`

Related Commands [\[SOURce:\]CURRent\[:LEVel\]\[:IMMediate\]\[:AMPLitude\]\[SOURce:\]VOLTage\[:LEVel\]:TRIGgered\[:IMMediate\]\[:INCRement\]*TRGTRIGger\[:IMMediate\]](#)

Arguments CH1, CH2, and CH3 are the channel numbers (only CH1 or CH2 are available for the two channel instruments).

NOTE. CH3 is not a valid channel on dual output models.

Examples `INST:COUP CH1,CH2`

INSTrument:SElect

This command is used to switch the current channel command.

Group Channel

Syntax `INSTrument:SElect {CH1|CH2|CH3}`
`INSTrument:SElect?`

Arguments CH1, CH2, or CH3 are the channels you can switch the instrument to.
Availability of channels depends on which model of power supply you have.

MEASure[:SCALar]:CURRent[:DC]? (Query Only)

This command initiates and executes a new current measurement, and returns the measured output current of the power supply. If a channel is specified, the query returns the measurement for the specified channel. If no channel is specified, the currently selected channel is measured and returned.

Group Measurement

Syntax `MEASure[:SCALar]:CURRent[:DC]? [CH1|CH2|CH3|ALL]`

Related Commands [FETCh\[:SCALar\]:VOLTage\[:DC\]?](#), [INSTrument:SElect](#)

Returns <NR2> is the measured output current in amperes.

Examples `MEAS:CURR? ALL` might return 0.0998707, 0.0999861, 0 which would be the measured currents on channels 1, 2 and 3 in amperes.

MEASure[:SCALar]:POWer[:DC]? (Query Only)

This command initiates and executes a new output power measurement, and returns the measured output current of the power supply. If a channel is specified, then the query returns the measurement for the specified channel. If no channel is specified, then the currently-selected channel is measured and returned.

Group	Measurement
Syntax	MEASure[:SCALar]:POWer[:DC]? [CH1 CH2 CH3 ALL]
Related Commands	FETCh[:SCALar]:POWer[:DC]? , INSTrument:SElect
Arguments	CH1, CH2, or CH3 is the channel on which to read the output power. ALL is to read the output power on all channels.
Returns	<NR2> is the measured output power in watts.
Examples	MEAS:POW? ALL might return 9.97077, 0.00205158, 0 which would be the power being supplied on channels 1, 2, and 3 in watts.

MEASure[:SCALar][:VOLTage][:DC]? (Query Only)

This command initiates and executes a new voltage measurement, and returns the measured output voltage of the power supply. If a channel is specified, the query returns the measurement for the specified channel. If no channel is specified the currently-selected channel is measured and returned.

Group	Measurement
Syntax	MEASure[:SCALar][:VOLTage][:DC]? [CH1 CH2 CH3 ALL]
Related Commands	FETCh[:SCALar]:VOLTage[:DC]? , INSTrument:SElect
Arguments	CH1, CH2, or CH3 is the channel on which to read the output voltage. ALL is to read the output voltage on all channels.
Returns	<NR2> is the measured output voltage in volts.
Examples	MEAS:VOLT? ALL might return 20.0002, 0.999465, 4.00024 which would be the measured voltages on channels 1, 2, and 3 in volts.

*OPC

This command configures the instrument to generate an operation complete message by setting bit 0 of the Standard Event Status Register (SESR) when all pending commands that generate an OPC message are complete.

The query command places the ASCII character "1" into the output queue when all such OPC commands are complete.

Group Synchronization

Syntax *OPC
*OPC?

Examples *OPC? might return 1 to indicate that all pending OPC operations are finished.

*PSC

Sets and queries the power-on status flag that controls the automatic power-on states of SRER and ESER. When *PSC is true, the Service Request Enable Register (SRER) and Event Status Enable Register (ESER) are set to 0 at power-on. When *PSC is false, the current values in the SRER and ESER are preserved in nonvolatile memory when power is shut off and are restored at power-on.

Group Source

Syntax *PSC <NR1>
*PSC?

Related Commands [*RST](#), [*OPC](#)

Arguments <NR1> = 0 sets the power-on status clear flag to false, disables the power-on clear, and allows the power supply to possibly assert SRQ after power on.

<NR1> ≠ 0 sets the power-on status clear flag to true. Sending *PSC 1 therefore enables the power-on status clear and prevents any SRQ assertion after power-on.

Returns 0 | 1

Examples `*PSC 0`
sets the power-on status clear flag to false.

`*PSC?`
might return 1, indicating that the power-on status clear flag is set to true.

***RCL (No Query Form)**

Restores the state of the power supply from a copy of its 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.

Group Save and Recall

Syntax `*RCL <NR1>`

Related Commands `*SAV`

Arguments `<NR1>` is an integer value in the range from 1 to 30 and specifies the location of setup memory.

Examples `*RCL 3`
sets the power supply to settings stored in memory location 3.

***RST (No Query Form)**

This command resets the power supply to default settings, but does not purge any stored settings.

Sending the `*RST` command does the following:

- Returns the power supply settings to the defaults. (See page C-1, *Default Setup*.)
- Clears the pending operation flag and associated operations

The *RST command does not change the following items:

- State of the USB or GPIB interface
- Calibration data that affects device specifications
- Current GPIB power supply address
- Stored settings
- Output queue
- Service Request Enable Register settings
- Standard Event Status Enable Register settings
- Power-On Status Clear flag setting
- front-panel LOCK state

Group Status

Syntax *RST

*SAV (No Query Form)

Saves the state of the power supply into a specified nonvolatile memory location. Any settings that had been stored previously at the location are overwritten. You can later use the *RCL command to restore the power supply to this saved state.

Group Status

Syntax *SAV <NR1>

Related Commands [*RCL](#)

Arguments <NR1> is an integer value in the range from 1 to 30.

Examples *SAV 2
saves the settings in memory location 2.

[SOURce:]APPLY (No Query Form)

This command is used to select channels and set voltage and current level using a single command.

Group Source

Syntax [SOURce:]APPLY
{CH1|CH2|CH3}, [VOLTage|Max|Min], [Current|Max|Min]
[SOURce:]APPLY [CH1|CH2|CH3]

Related Commands This command can replace the following commands: INST CH1, VOLT 3V, and CURR 1A. (See example below.)

Arguments CH1, CH2, or CH3 are the three channels (two for two channel instrument models).

Voltage. volt age is a flexible decimal number (NRf) that may be type NR1, NR2 or NR3. It specifies the voltage setting, which can range from 0 to the maximum nameplate voltage of the power supply.

MAX sets the voltage to the maximum level (note that the maximum level may be somewhat higher than the nameplate).

MIN sets the voltage to the minimum level (0 V).

Current. Current is a flexible decimal number (NRf) that may be type NR1, NR2 or NR3. It specifies a level between the minimum current and maximum nameplate current level for the power supply.

MAX sets the current to the maximum level.

MIN sets the current to the minimum level (0 A).

Examples [SOURCE:]APPLY CH1, 3V, 1A would set channel 1 to 3 volts, 1 amps.

[SOURce]:CHANnel:OUTPut:[STATe]

This command is used to individually control the output state of the single, currently-specified channel.

Group Source

Syntax `[SOURCE]:CHANnel:OUTPut:[STATE] {0|1|ON|OFF}`
 `[SOURCE]:CHANnel:OUTPut:[STATE]? ON`

Related Commands [INSTrument:SElect](#)

Arguments 0 or OFF indicates that the channel is off.
 1 or ON indicates that the channel is on.

Examples `CHANNEL:OUTPUT OFF` would turn whichever channel had been selected using an `INSTRUMENT` command to the off state.
 `CHAN:OUTP?` might return 0 to indicate that the current channel is off.

[SOURCE:]CURRent[:LEVel]:DOWN[:IMMediate][:AMPLitude] (No Query Form)

This command is used to decrease the current level by a step.
 The stepping current can be set by the following command:
`[SOURCE:]CURRent[:LEVel][:IMMediate]:STEP[:INCRement]`

Group Source

Syntax `[SOURCE:]CURRent[:LEVel]:DOWN[:IMMediate][:AMPLitude]`

Related Commands [\[SOURCE:\]CURRent\[:LEVel\]\[:IMMediate\]:STEP\[:INCRement\]](#)

[SOURCE:]CURRent[:LEVel][:IMMediate][:AMPLitude]

This command is used to set or query the current value of the power supply in units of A or mA.

Group Source

Syntax `[SOURCE:]CURRent[:LEVel][:IMMediate][:AMPLitude]`
 `{<Current>|MIN|MAX}`
 `[SOURCE:]CURRent[:LEVel][:IMMediate][:AMPLitude]?`

Related Commands [INSTrument:SElect](#)

Arguments	<p><Current> is a flexible decimal number (NRf) that may be type NR1, NR2 or NR3. It specifies a level between the minimum current and maximum nameplate current level for the power supply.</p> <p>MIN sets the current to the minimum level (0 A).</p> <p>MAX sets the current to the maximum level.</p>
Returns	<NR2> is the current setting in amperes.
Examples	<p>CURR 3A</p> <p>CURR 30mA</p> <p>CURR MIN</p> <p>CURR? might return 2.0000, which would be the current setting in amperes.</p>

[SOURce:]CURRent[:LEVel][:IMMediate]:STEP[:INCRement]

This command is used to set the current step value.

Group	Source
Syntax	<p>[SOURce:]CURRent[:LEVel][:IMMediate]:STEP[:INCRement] {<current level>}</p> <p>[SOURce:]CURRent[:LEVel][:IMMediate]:STEP[:INCRement]?</p>
Related Commands	[SOURce:]CURRent[:LEVel]:UP[:IMMediate][:AMPLitude] , [SOURce:]CURRent[:LEVel]:DOWN[:IMMediate][:AMPLitude]
Arguments	The current level in A, mA, or μ A.

[SOURce:]CURRent:TRIGgered[:IMMediate]

This command is used to set the current level for the trigger function. The command queues the next setting for the currently selected channel. The units are A, mA or μ A. When the instrument receives its next trigger, the currently selected channel will be set to the specified value.

Group	Source
--------------	--------

Syntax `[SOURCE:]CURRENT:TRIGGERed[:IMMEDIATE] {<current>|MIN|MAX}`
`[SOURCE:]CURRENT:TRIGGERed[:IMMEDIATE]?`

Related Commands [INSTrument:SElect](#)

Arguments <Current> is a flexible decimal number (NRf) that may be type NR1, NR2 or NR3. It specifies a level between the minimum current and maximum nameplate current level for the power supply.

MIN sets the current to the minimum level (0 A).

MAX sets the current to the maximum level.

Examples `CURRENT:TRIGGERED 1.1A`

[SOURCE:]CURRENT[:LEVel]:UP[:IMMEDIATE][:AMPLitude] (No Query Form)

This command is used to increase the current level by a step. The stepping current can be set by the `[SOURCE:]CURRENT[:LEVel][:IMMEDIATE]:STEP[:INCRement]` command.

Group Source

Syntax `[SOURCE:]CURRENT[:LEVel]:UP[:IMMEDIATE][:AMPLitude]`

Related Commands [\[SOURCE:\]CURRENT\[:LEVel\]\[:IMMEDIATE\]:STEP\[:INCRement\]](#)

[SOURCE:]OUTPut:ENABLE

This command enables or disables the current channel. This command performs the same function as the “Channel Enable” selection in the menu.

Group Source

Syntax `[SOURCE:]OUTPut:ENABle`
`[SOURCE:]OUTPut:ENABle?`

Arguments 0 disables the current channel.
1 enables the current channel.

Returns Disabled
Enabled

[SOURce:]OUTPut:PARAllel[:STATe]

This command sets the parallel state of CH1 and CH2.

Group Source

Syntax [SOURce:]OUTPut:PARAllel[:STATe] 0|1|OFF|ON
[SOURce:]OUTPut:PARAllel[:STATe]?

Arguments 0 or OFF sets the parallel state to off.
1 or ON sets the parallel state to on.

Examples [SOURCE:]OUTPUT:PARALLEL[:STATE]

[SOURce:]OUTPut:PON[:STATe]

This command configures the power supply to power up with its output turned off, or to return the output to the state it was in when it powered down.

Group Source

Syntax [SOURce:]OUTPut:PON[:STATe] {RST|RCL0}
[SOURce:]OUTPut:PON[:STATe]?

Arguments RST sets the power supply to power-up with output off.
RCL0 sets the power supply to power-up with the output in the last state before power was removed.

Returns RST|RCL0

Examples `OUTPUT:PON RST`

`OUTPUT:PON?` might return `RCL0`, which would indicate that the instrument will return to the present output state if the power is cycled.

[SOURce:]OUTPut:SERies

This command sets the serial state of CH1 and CH2.

Group Source

Syntax `[SOURce:]OUTPut:SERies {0|1|OFF|ON}`
`[SOURce:]OUTPut:SERies?`

Arguments 0 or OFF sets the parallel state to off.
1 or ON sets the parallel state to on.

Examples `[SOURce:]OUTPUT:SERIES`

[SOURce:]OUTPut[:STATe][:ALL]

This command turns all of the enabled output channels on or off.

Group Source

Syntax `[SOURce:]OUTPut[:STATe][:ALL] {0|1|ON|OFF}`
`[SOURce:]OUTPut[:STATe][:ALL]?`

Related Commands [\[SOURce:\]OUTPut:TIMer\[:STATe\]](#)

Arguments 0 or OFF turns the power supply output off.
1 or ON turns the power supply output on.

Returns 1|0

Examples `OUTPUT ON`
`OUTPUT?` might return 0, which would indicate that the outputs are off.

[SOURce:]OUTPut:TIMer:DElay

This command sets the time duration of the output timer for the currently selected channel. When the timer is activated and a duration is set, the specified output of the power supply will turn off automatically if left on longer than the specified duration. In order to ensure proper operation of the output timer, the timer must be activated using the [SOURce:]OUTPut:TIMer[:STATe] command before turning the output on.

Group Source

Syntax [SOURce:]OUTPut:TIMer:DElay {<duration>|MIN|MAX|DEF}
[SOURce:]OUTPut:TIMer:DElay?

Related Commands [SOURce:]OUTPut:TIMer[:STATe], [SOURce:]OUTPut[:STATe][:ALL], INSTRument:SElect

Arguments <duration> ::= <NRf><units>
where
<NRf> is a flexible decimal specifying time in the range 0.01s (or 10ms) to 60000s.
<units>::={S|ms}

MIN: The minimum time of the output timer (0.01 s).
MAX: The maximum time of the output timer (60,000 s).
DEF: The default time of the output timer (60 s).

Returns <NR2> is the timer duration in seconds.

Examples OUTP:TIM:DEL 120

OUTP:TIM:DEL? might return 60.2, which would represent the maximum time, in seconds, that the output of the instrument could be turned on if the timer is active.

[SOURce:]OUTPut:TIMer[:STATe]

This command turns the output timer function on and off. When the timer is activated and a duration is set, an output channel of the power supply will turn off automatically if left on longer than the specified duration. In order to ensure proper operation of the output timer, the timer must be activated using the [SOURce:]OUTPut:TIMer[:STATe] command before turning the output on.

Group	Source
Syntax	<code>[SOURce:]OUTPut:TIMer[:STATe] {0 1 ON OFF}</code> <code>[SOURce:]OUTPut:TIMer[:STATe]?</code>
Related Commands	[SOURce:]OUTPut:TIMer:DELaY , [SOURce:]OUTPut[:STATe][:ALL] , INSTrument:SELect
Arguments	0 or OFF turns the output timer off. 1 or ON turns the output timer on.
Returns	0 1
Examples	To activate the timer, first send <code>OUTPUT:TIMER:STATE ON</code> , then send <code>OUTPUT:STATE ON</code> . To turn the timer off, send <code>OUTPUT:TIMER:STATE OFF</code> .

[SOURce:]VOLTage[:LEVel]:DOWN[:IMMediate][:AMPLitude] (No Query Form)

This command is used to decrease the voltage level of the currently selected channel by a step. The voltage step value can be set by the following command:
`[SOURce:]VOLTage[:LEVeL][:IMMediate]:STEP[:INCRement]`

Group	Source
Syntax	<code>[SOURce:]VOLTage[:LEVeL]:DOWN[:IMMediate][:AMPLitude]</code>
Related Commands	[SOURce:]VOLTage[:LEVel][:IMMediate]:STEP[:INCRement] , [SOURce:]VOLTage[:LEVel][IMMediate][:AMPLitude] , [SOURce:]VOLTage[:LEVel]:UP[:IMMediate][:AMPLitude]

[SOURce:]VOLTage[:LEVel][IMMediate][:AMPLitude]

This command is used to set the voltage level of the of the currently selected channel. The units are V, mV or kV.

Group	Source
--------------	--------

Syntax [SOURCE:]VOLTage[:LEVel][IMMediate][:AMPLitude]
 {<voltage>|MIN|MAX}
 [SOURCE:]VOLTage[:LEVel][IMMediate][:AMPLitude]?

Related Commands [SOURCE:]VOLTage[:LEVel]:DOWN[:IMMediate][:AMPLitude]
 [SOURCE:]VOLTage[:LEVel]:UP[:IMMediate][:AMPLitude]
 [SOURCE:]VOLTage[:LEVel][:IMMediate]:STEP[:INCRement]
 INSTRument:SElect

Arguments <voltage> is a flexible decimal number (NRf) that may be type NR1, NR2 or NR3. It specifies the voltage setting, which can range from 0 to the maximum nameplate voltage of the power supply.

MIN sets the voltage to the minimum level (0 V).

MAX sets the voltage to the maximum level (note that the maximum level may be somewhat higher than the nameplate).

UP sets the voltage level to increase a step.

DOWN sets the voltage level to decrease a step.

DEF is the default level (1 V).

Returns NR2 is the voltage setting in volts.

Examples VOLTage:MIN
 VOLTage? might return 1.05, which would be the voltage setting in volts.

[SOURCE:]VOLTage[:LEVel][:IMMediate]:STEP[:INCRement]

This command is used to set the voltage step value.

Group Source

Syntax [SOURCE:]VOLTage[:LEVel][:IMMediate]:STEP[:INCRement]
 {<voltage>}
 [SOURCE:]VOLTage[:LEVel][:IMMediate]:STEP[:INCRement]?

Related Commands [SOURCE:]VOLTage[:LEVel]:DOWN[:IMMediate][:AMPLitude]

[SOURce:]VOLTage[:LEVel]:UP[:IMMediate][:AMPLitude]

[SOURce:]VOLTage[:LEVel][IMMediate][:AMPLitude]

INSTrument:SElect

Arguments <vol tage> is the voltage in kV, V, mV, or μ V.

[SOURce:]VOLTage[:LEVel]:UP[:IMMediate][:AMPLitude] (No Query Form)

This command is used to increase the voltage level of the currently selected channel by a step. The voltage step value can be set by the following command:
[SOURce:]VOLTage[:LEVel][:IMMediate]:STEP[:INCRement]

Group Source

Syntax [SOURce:]VOLTage[:LEVel]:UP[:IMMediate][:AMPLitude]

Related Commands [SOURce:]VOLTage[:LEVel][:IMMediate]:STEP[:INCRement]
[SOURce:]VOLTage[:LEVel][IMMediate][:AMPLitude]

[SOURce:]VOLTage[:LEVel]:TRIGgered[:IMMediate][:INCRement]

This command is used to set the voltage level for the trigger function.

Group Trigger

Syntax [SOURce:]VOLTage[:LEVel]:TRIGgered[:IMMediate][:INCRement]
[SOURce:]VOLTage[:LEVel]:TRIGgered[:IMMediate][:INCRement]?

Arguments <vol tage> is a flexible decimal number (NRf) that may be type NR1, NR2 or NR3. It specifies the voltage setting, which can range from 0 to the maximum nameplate voltage of the power supply.

MIN sets the voltage to the minimum level (0 V).

MAX sets the voltage to the maximum level (note that the maximum level may be somewhat higher than the nameplate).

UP sets the voltage level to increase a step.

DOWN sets the voltage level to decrease a step.

DEF is the default level (1 V).

Returns The voltage in kV, V, mV, or uV.

[SOURce:]VOLTage:LIMit[:LEVel]

This command limits the maximum voltage that can be programmed on the power supply. This command will apply the limit to the currently-selected channel and corresponds to the front-panel Max Voltage setting that can be found under the Protection Settings submenu.

Group Source

Syntax [SOURce:]VOLTage:LIMit[:LEVel] {<voltage>|MIN|MAX}
[SOURce:]VOLTage:LIMit[:LEVel]?

Related Commands [INSTrument:SElect](#)
[\[SOURce:\]VOLTage:LIMit:STATe](#)

Arguments <voltage> is a flexible decimal number that may be type NR1, NR2 or NR3. It specifies the voltage limit setting, which can range from 0 to the maximum nameplate voltage of the power supply.

MIN sets the maximum voltage to the minimum level (0 V).

MAX sets the maximum voltage to the maximum level (note that the maximum level may be somewhat higher than the nameplate).

Examples VOLTAGE:LIMIT:STATE 6V
[SOURCE:]VOLTAGE:LIMIT[:LEVEL]? might return 30.1, which is the maximum voltage limit setting on the current channel.

[SOURce:]VOLTage:LIMit:STATe

This command turns the maximum voltage for the currently selected channel on or off on the current channel. This limit corresponds to the Max Volt Set command on the front panel of the instrument.

Group	Source
Syntax	[SOURCE:]VOLTage:LIMit:STATE {0 OFF 1 ON} [SOURCE:]VOLTage:LIMit:STATE?
Related Commands	INSTrument:SElect [SOURCE:]VOLTage:LIMit[:LEVel]
Arguments	0 or OFF turns off the maximum voltage limit. 1 or ON turns it on.
Examples	VOLTAGE:LIMIT:STATE ON

[SOURCE:]VOLTage:TRIGgered[:IMMediate]

This command is used to set the voltage level for the trigger function. The command queues the next setting for the currently selected channel. The units are kV, V, mV or μ V. When the instrument receives its next trigger, the currently selected channel will be set to the specified value. So, 8000000 μ V will set the voltage to 8 V on the front panel.

Group	Source
Syntax	[SOURCE:]VOLTage:TRIGgered[:IMMediate] {<voltage> MIN MAX} [SOURCE:]VOLTage:TRIGgered[:IMMediate]?
Related Commands	INSTrument:SElect
Arguments	<voltage> is a flexible decimal number (NRf) that may be type NR1, NR2 or NR3. It specifies a level between the minimum voltage and maximum nameplate voltage level for the power supply. MIN sets the voltage to the minimum level (0 V). MAX sets the voltage to the maximum level.
Examples	VOLTAGE:TRIGGERED 4.5V

*SRE

(Service Request Enable) sets and queries the bits in the Service Request Enable Register (SRER). Refer to the *Status and Events* chapter for more information.

Group Status

Syntax *SRE <NR1>
*SRE?

Related Commands [*CLS](#), [*ESR?](#), [*PSC](#)

Arguments <NR1> is an integer value in the range from 0 to 255. The binary bits of the SRER are set according to this value. Using an out-of-range value causes an execution error. The power-on default for SRER is 0 if *PSC is 1. If *PSC is 0, the SRER maintains its value through a power cycle.

Examples *SRE 48 sets the bits in the SRER to 00110000 binary.
*SRE? might return a value of 32, showing that the bits in the SRER have the binary value 00100000.

STATus:OPERation:ENABLE

This command sets and queries the contents of the operation enable register (OENR). The OENR is an eight-bit mask register that determines which bits in the Operation Event Register (OEVR) will affect the state of the OPER bit in the Status Byte Register (SBR). Details about the status registers are available in this manual. (See page 3-1, *Status and Events*.)

Group Status

Syntax STATus:OPERation:ENABLE <NR1>
STATus:OPERation:ENABLE?

Related Commands [*PSC](#), [STATus:OPERation:INSTrument\[:EVENT\]?](#)

Arguments <mask> ::= <NR1>
where
<NR1> is a decimal integer ranging from 0 through 255. The binary bits of the OENR are set according to this value.

Returns <mask>

Examples STATUS:OPERATION:ENABLE 2
STATUS:OPERATION:ENABLE? might return 2.

STATus:OPERation[:EVENT]? (Query Only)

This command returns the contents of the operation event register (OEVR). After executing this command the operation event register is reset. Details about status registers are available in this manual. (See page 3-1, *Status and Events*.)

Group Status

Syntax STATus:OPERation[:EVENT]?

Related Commands [STATus:OPERation:INSTrument\[:ENABLE\]?](#)

Returns <NR1> is a decimal integer representation of the contents of the Operation Event Register (OEVR), ranging from 0 to 255.

Examples STATUS:OPERATION:EVENT? might return 2, which indicates that the summary bit is set.

STATus:OPERation:INSTrument[:ENABLE]? (Query Only)

This command queries the contents of the operation instrument enable register (OENR). The OENR is an eight-bit mask register that determines which bits in the Operation Enable Register (OENR) will affect the state of the OPER bit in the Status Byte Register (SBR). Details about the status registers are available in this manual. (See page 3-1, *Status and Events*.)

Group Status

Syntax STATus:OPERation:INSTrument[:ENABle]? <NR1>

Related Commands [*PSC](#)
[STATus:OPERation:INSTrument\[:EVENT\]?](#)

Returns <mask>

Examples STATUS:OPERATION:INSTRUMENT[:ENABLE]? might return 128, which would indicate that only the Constant Current bit of the Operation Enable Register would affect the OPER bit of the Status Byte Register.

STATus:OPERation:INSTRument[:EVENT]? (Query Only)

This command returns the contents of the operation event register. After executing this command the operation event register is reset. Details about status registers are available in this manual. (See page 3-1, *Status and Events*.)

Group Status

Syntax STATus:OPERation:INSTRument[:EVENT]?

Related Commands [STATus:OPERation:INSTRument\[:ENABLE\]?](#)

Returns <NR1> is a decimal integer representation of the contents of the Operation Event Register (OEVR), ranging from 0 to 255.

Examples STATUS:OPERATION:INSTRument[:EVENT]? might return 10, which indicates that the power supply is waiting for trigger and is in a constant current mode.

STATus:OPERation:INSTRument:ISUMmary<x>:CONDition? (Query Only)

This command is used to query the operation condition register of a channel, where <x> is 1, 2, or 3. 1, 2, and 3 are channel 1, 2, and 3, respectively. Only 1 and 2 are available on two channel instruments.

Group Status

Syntax STATus:OPERation:INSTRument:ISUMmary<x>:CONDition?

Related Commands [STATus:OPERation:INSTRument:ISUMmary<x>\[:EVENT\]?](#), [STATus:OPERation:INSTRument:ISUMmary<x>:ENABLE](#)

Returns <NR1> is a decimal integer ranging from 0 through 255. The binary bits of the specified channel's operation enable register are set according to this value.

STATus:OPERation:INSTrument:ISUmmary<x>:ENABle

This command is used to modify or query the operation enable register of a channel, where <x> is 1, 2, or 3. 1, 2, and 3 are channel 1, 2, and 3, respectively. Only 1 and 2 are available on two channel instruments. (See page 3-1, *Status and Events*.)

Group Status

Syntax STATus:OPERation:INSTrument:ISUmmary<x>:ENABle
STATus:OPERation:INSTrument:ISUmmary<x>:ENABle?

Arguments <mask>::=<NR1>
where
<NR1> is a decimal integer ranging from 0 through 255. The binary bits of the OENR are set according to this value.

register

Examples STAT:OPER:INST:ISUM2:ENABle 2

STAT:OPER:INST:ISUM2:ENABle? might return 2, which would indicate that only the Constant Current bit of the Operation Enable Register for channel 2 would affect the OPER bit of the Status Byte Register.

STATus:OPERation:INSTrument:ISUmmary<x>[:EVENT]? (Query Only)

This queries the operation event register summary of a channel, where <x> is 1, 2, or 3. 1, 2, and 3 are channel 1, 2, and 3, respectively. Only 1 and 2 are available on two channel instruments. (See page 3-1, *Status and Events*.)

Group Status

Syntax STATus:OPERation:INSTrument:ISUmmary<x>[:EVENT]?

Returns <NR1> is a decimal integer representation of the contents of the Operation Event Register (OEVr) for the specified channel, ranging from 0 to 255.

Examples `STATUS:OPERATION:INSTRUMENT:ISUMMARY1:EVENT?` might return 9, which indicates that channel 1 is turned on and in constant voltage mode.

STATus:QUESTionable:ENABLE

This command sets and queries the contents of the questionable enable register (QENR). The QENR is an eight-bit mask register that determines which bits in the Questionable Event Register (QEVr) will affect the state of the QUES bit in the Status Byte Register (SBR).

Group Status

Syntax `STATUS:QUESTionable:ENABLE <NR1>`
`STATUS:QUESTionable:ENABLE?`

Related Commands [STATus:QUESTionable\[:EVENT\]? , *PSC](#)

Arguments `<NR1>` is a decimal integer ranging from 0 through 255. The bits of the mask register of the QENR are set according to this value.

Returns `<mask>`

Examples `STATUS:QUESTIONABLE:ENABLE 8`

`STATUS:QUESTIONABLE:ENABLE?` might return 8, which would indicate that only a transition of the Remote Inhibit bit of the QCR would affect the QUES bit of the Status Byte Register.

STATus:QUESTionable[:EVENT]? (Query Only)

This command returns the contents of the questionable event register (QEVr). After executing this command, the quest event register is reset. Details about the QEVr are available in this manual. (See page 3-1, *Status and Events*.)

Group Status

Syntax `STATus:QUESTionable[:EVENT]?`

Related Commands [STATus:QUESTionable:ENABLE](#)

Returns <NR1> is a decimal integer representation of the contents of the Questionable Event Register (QEVr), ranging from 0 to 255.

Examples STATUS:QUESTIONABLE:EVENT? might return 0, which would indicate an over voltage condition.

STATus:QUEStionable:INSTrument:ENABle

This command queries the questionable instrument status event register of the instrument.

Group Status

Syntax STATus:QUEStionable:INSTrument:ENABle <NR1>
STATus:QUEStionable:INSTrument:ENABle?

Arguments <NR1> is a decimal integer ranging from 0 through 255. The bits of the mask register of the Questionable Enable Register for the specified channel are set according to this value.

Examples STATus:QUEStionable:INSTrument[:EVENT] 8
STATus:QUEStionable:INSTrument[:EVENT]? might return 8, which would indicate that only a transition of the Remote Inhibit bit of the QCR would affect the QUES bit of the Status Byte Register.

STATus:QUEStionable:INSTrument[:EVENT]? (Query Only)

This command queries the questionable instrument status event register of the instrument.

Group Status

Syntax STATus:QUEStionable:INSTrument[:EVENT]?

Related Commands [STATus:QUEStionable:INSTrument:ENABle](#)

Returns <NR1> is a decimal integer representation of the contents of the Questionable Event Register (QEVr), ranging from 0 to 255.

Examples `STATUS:QUESTIONABLE:INSTRUMENT:EVENT?` might return 0, which would indicate an over voltage condition.

STATus:QUESTionable:INSTrument:ISUMmary<x>:CONDition? (Query Only)

This queries the questionable condition register (QCR) summary of a channel, where <x> is 1, 2, or 3. 1, 2, and 3 are channel 1, 2, and 3, respectively. Only 1 and 2 are available on two channel instruments. (See page 3-1, *Status and Events*.)

Group Status

Syntax `STATus:QUESTionable:INSTrument:ISUMmary<x>:CONDition?`

Returns <NR1> is a decimal integer representation of the contents of the Questionable Condition Register (OCR), ranging from 0 to 255.

Examples `STAT:QUES:INST:ISUMM1:COND?` might return 1, which would indicate an over voltage condition on channel 1.

STATus:QUESTionable:INSTrument:ISUMmary<x>:ENABLE7

This command is used to modify or query the operation enable register summary of a channel, where <x> is 1, 2, or 3. 1, 2, and 3 are channel 1, 2, and 3, respectively. Only 1 and 2 are available on two channel instruments. (See page 3-1, *Status and Events*.)

Group Status

Syntax `STATus:QUESTionable:INSTrument:ISUMmary<x>:ENABLE <NR1>`
`STATus:QUESTionable:INSTrument:ISUMmary<x>:ENABLE?`

Arguments <NR1> is a decimal integer representation ranging from 0 through 255. The bits of the mask register of the Questionable Enable Register for the specified channel are set according to this value.

Examples `STATUS:QUESTionable:INSTrument:ISUMmary2:ENABLE? 2`

STATus:QUESTionable:INSTRument:ISUMmary<x>:[EVENT]? (Query Only)

This queries the operation event register of summary of a channel, where <x> is 1, 2, or 3. 1, 2, and 3 are channel 1, 2, and 3, respectively. Only 1 and 2 are available on two channel instruments. (See page 3-1, *Status and Events*.)

Group	Status
Syntax	STATus:QUESTionable:INSTRument:ISUMmary<x>:[EVENT]?
Returns	<NR1> is a decimal integer representation of the contents of the Questionable Event Register for the specified channel, ranging from 0 to 255.
Examples	STATus:QUESTIONABLE:INSTRUMENT:ISUMMARY3:EVENT? might return 2, indicating that channel 3 transitioned to constant current mode.

*STB? (Query Only)

The byte query returns the contents of the Status Byte Register (SBR) using the Master Summary Status (MSS) bit. Refer to the *Status and Events* chapter for more information. (See page 3-3.)

Group	Status
Syntax	*STB?
Related Commands	*ESE , *CLS , *ESR?
Returns	<NR1>
Examples	*STB? 96 shows that the SBR contains the binary value 01100000.

SYSTem:ERRor? (Query Only)

This command queries the error code and error information of the power supply and returns both values. (See Table 3-10 on page 3-10.)

Group	System
Syntax	SYSTem:ERRor?
Returns	<NR1>,<error_text> <error_text> ::= <string> where <string> is a description of the error.
Examples	SYSTEM:ERROR? might return 110, which means No Input Command to parse.

SYSTem:KEY

This command can produce the same effect as pressing one of the front-panel buttons. The instrument must be in local mode in order for this command to simulate a front-panel button press.

Group	System
Syntax	SYSTem:KEY <NR1> SYSTem:KEY?
Arguments	<NR1> is an integer key code (see the following table).
Returns	<NR1>

Front-panel button	<NR1> key code
KEY_VSET	1
KEY_ISET	2
KEY_SAVE	3
KEY_RECALL	4
KEY_LEFT	5
KEY_RIGHT	6
KEY_UP	7
KEY_DOWN	8
KEY_0	9
KEY_1	10
KEY_2	11
KEY_3	12

Front-panel button	<NR1> key code
KEY_4	13
KEY_5	14
KEY_6	15
KEY_7	16
KEY_8	17
KEY_9	18
KEY_DECIMAL	19
KEY_ESC	20
KEY_ENTER	21
KEY_ON	22
KEY_SHIFT	64
MENU	23
CH1	24
CH2	25
CH3	26

Examples `SYSTEM:KEY 64` would simulate a press of the Shift key.

SYSTem:LOCa1 (No Query Form)

This command sets the power supply for control from the front-panel.

Group System

Syntax `SYSTem:LOCa1`

Related Commands [SYSTem:REMOte](#), [SYSTem:RWLock](#)

Examples `SYS:LOC`

SYSTem:MODUle? (Query Only)

This command queries the module of the power supply.

Group System

Syntax	SYSTem:MODUle?
Returns	<> where <string> is a description of the error.
Examples	SYSTEM:MODULE? might return ?, which means ?.

SYSTem:POSetup

This command determines how the power supply initializes when its power switch is turned on. This command configures the instrument to power up with default settings, or power up with the settings that were in effect when the instrument was turned off.

Group	System
Syntax	SYSTem:POSetup {RST RCL0} SYSTem:POSetup?
Arguments	RST: initializes the power supply to default settings after a power cycle. RCL0: saves the most recent settings and restores these after a power cycle.
Returns	RST: default settings are applied after a power cycle. RCL0: most recent settings are saved and restored after a power cycle.
Examples	SYST:POS RST SYSTEM:POSETUP? might respond with RST, which would indicate that the power supply is configured to restore the power supply to default settings when it powers up.

SYSTem:REMOte (No Query Form)

This command sets the power supply to remote control mode.

Group	System
Syntax	SYSTem:REMOte

Related Commands [SYSTem:LOCal](#), [SYSTem:RWLock](#)

Arguments None.

Examples `SYSTEM:REMOTE`

SYSTem:RWLock (No Query Form)

If the power supply is in remote mode, this command locks out the front panel. This command has no effect if the instrument is in local mode.

Group System

Syntax `SYSTem:RWLock`

Related Commands [SYSTem:REMOte](#), [SYSTem:LOCal](#)

Arguments None.

Examples `SYSTEM:RWLOCK`

SYSTem:VERSion? (Query Only)

This command returns SCPI version of the instrument.

Group System

Syntax `SYSTem:VERSion?`

Returns <NR2> is the software version of the power supply.

Examples `SYSTEM:VERSION?` might return 1991.0, which is the SCPI version number.

*TRG (No Query Form)

This command generates a trigger event.

Group	Trigger
Syntax	*TRG
Related Commands	TRIGger[:IMMediate]
Examples	*TRG

TRIGger[:IMMediate] (No Query Form)

This command forces an immediate trigger event.

Group	Trigger
Syntax	TRIGger[:IMMediate]
Related Commands	*TRG
Arguments	None.
Examples	TRIGGER

*TST? (Query Only)

Initiates a self-test and reports any errors.

Group	Diagnostic
Syntax	*TST?
Returns	<p><NR1> where <NR1>= 0 indicates that the self-test completed with no errors. <NR1> not equal to 0 indicates that the self test detected an error.</p> <p>Self test code descriptions are available. (See Table 3-14 on page 3-12.)</p>

***WAI (No Query Form)**

This command prevents the instrument from executing further commands or queries until all pending commands are complete.

Group Synchronization

Syntax *WAI

Examples *WAI

Status and Events

Status and Events

This section provides details about the status information and events the power supply reports.

Status Reporting Structure

A diagram is provided showing an outline of the power supply error and event reporting function. (See Figure 3-1.)

The error and event reporting system consists of the following four register groups:

- Status Byte
- Standard Event
- Operation Status
- Questionable Status

The operations processed in these registers are summarized in status bytes, which provide the error and event data.

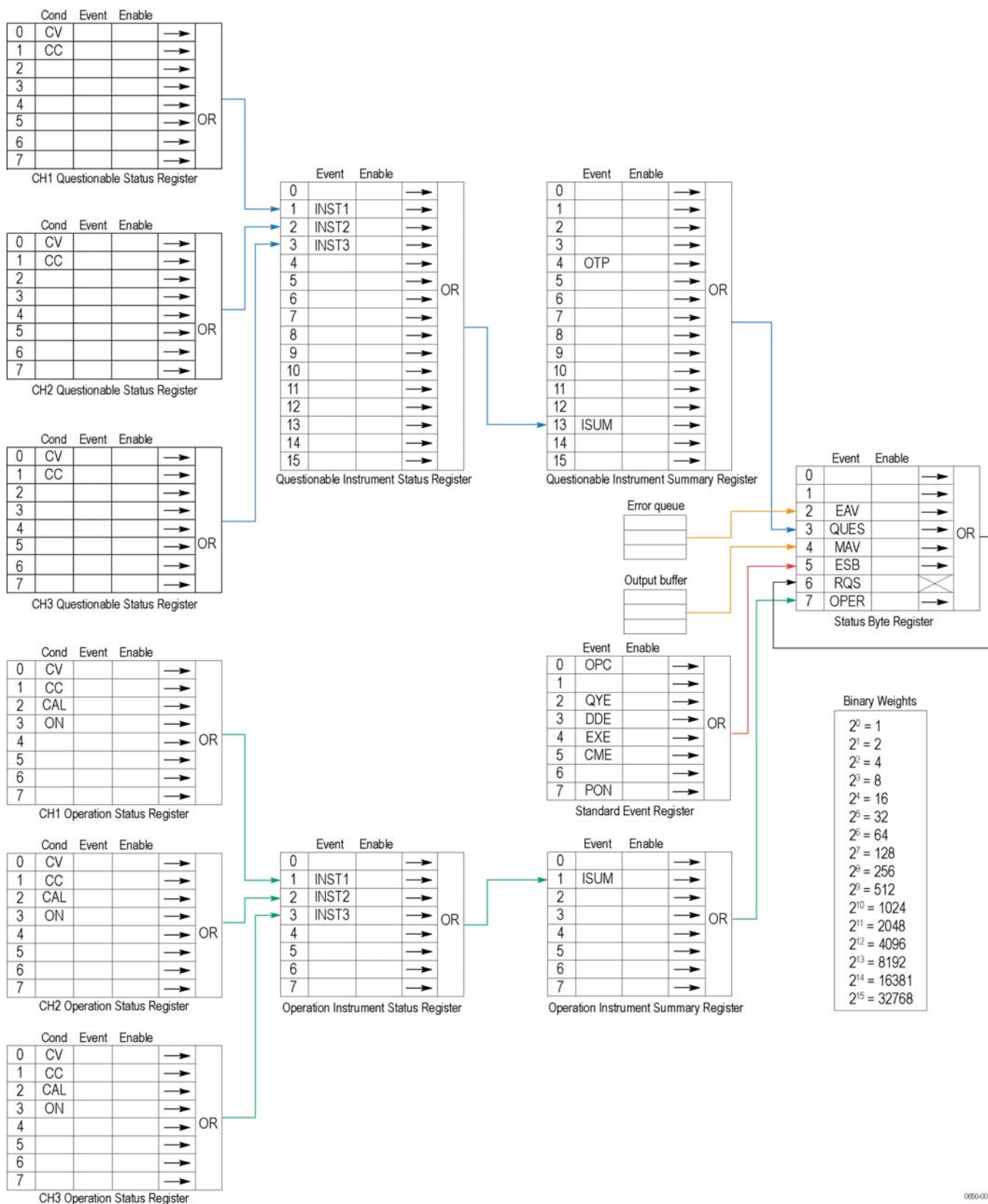


Figure 3-1: Error and event handling process

Registers

The registers in the event reporting system fall into two functional groups:

- Status registers contain information about the status of the power supply. They include the Status Byte Register (SBR), Standard Event Register (SER), the Questionable Status Register (QSR), the Questionable Instrument Status Register (QISR), the Operation Status Register (OSR), and the Operation Instrument Status Register (OISR).
- Summary registers record high-level summary information reported in the other register groups. They include the Questionable Instrument Summary Register (QISUR) and the Operation Instrument Summary Register (OISUR).

Status Registers

There are six types of status registers:

- Status Byte Register (SBR). (See page 3-3.)
- Standard Event Register (SER). (See page 3-4.)
- Operation Instrument Status Register (OISR). (See page 3-4.)
- Operation Status Register (OSR). (See page 3-5.)
- Questionable Instrument Status Register (QISR). (See page 3-5.)
- Questionable Status Register (QSR). (See page 3-6.)

The Status Byte Register (SBR). The SBR is made up of 8 bits. Bits 2, 4 and 5 are defined in accordance with IEEE Std 488.2-1992. These bits are used to monitor the error queue, output queue, and SER, respectively.

7	6	5	4	3	2	1	0
OPER	RQS	ESB	MAV	QUES	EAV	—	—

0481011

Figure 3-2: SBR bit functions

Table 3-1: SBR bit functions

Bit	Function	
7	OPER	Operation Status Bit. Indicates that an operation event has occurred.
6	RQS	Request Service. Obtained from a serial poll. Shows that the power supply requests service from the GPIB controller.
5	ESB	Event Status Bit. Shows that status is enabled and present in the SESR.
4	MAV	Message Available. Shows that output is available in the Output Queue.
3	QUES	Questionable Status Bit. Indicates that a questionable event has occurred.

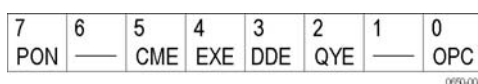
Table 3-1: SBR bit functions (cont.)

Bit	Function
2	EAV Shows that information is available in the Error Queue.
1	Not used.
0	Not used.

Each bit in an Enable Register corresponds to a bit in an Event Register. In order for an event to be reported to a bit in the Status Byte Register, the corresponding bit in the Enable Register must be set to one. If the bit in the Enable Register is set to zero, the event will not affect the status bit.

Various commands set the bits in the Enable Registers. Following are descriptions of the Enable Registers and the commands used to set them.

The Standard Event Register (SER). The SER records six types of events that can occur within the power supply as shown in the following figure.

**Figure 3-3: The Standard Event Register (SER)****Table 3-2: SER bit functions**

Bit	Function
7	PON Power On. Shows that the power supply was powered on.
6	Not used.
5	CME Command Error. Shows that an error occurred while the power supply was parsing a command or query.
4	EXE Execution Error. Shows that an error occurred while the power supply was executing a command or query.
3	DDE Device Error. Shows that a device dependent error occurred.
2	QYE Query Error. Either an attempt was made to read the Output Queue when no data was present or pending, or that data in the Output Queue was lost.
1	Not used.
0	OPC Operation Complete. Shows that the operation is complete. This bit is set when all pending operations complete following an *OPC command.

The Operation Instrument Status Register (OISR). The Operation Instrument Status Register is made up of eight bits that note the occurrence of events as shown here.



Figure 3-4: OISR bit functions

Table 3-3: OISR bit functions

Bit	Function	
7	—	This bit is not used.
6	—	This bit is not used.
5	—	This bit is not used.
4	—	This bit is not used.
3	INST3	STATUS:OPERation:INSTrument:ISUMmary3 is reported to INST3.
2	INST2	STATUS:OPERation:INSTrument:ISUMmary2 is reported to INST2.
1	INST1	STATUS:OPERation:INSTrument:ISUMmary1 is reported to INST1.
0	—	This bit is not used.

The Operation Status Register (OSR). The Operation Status Register is made up of 8 bits which note the occurrence of four types of events as shown here.



Figure 3-5: OSR bit functions

Table 3-4: OSR bit functions

Bit	Function	
7	—	This bit is not used.
6	—	This bit is not used.
5	—	This bit is not used.
4	—	This bit is not used.
3	ON	Output is ON or OFF.
2	CAL	New calibration parameters are being calculated.
1	CC	Constant current.
0	CV	Constant voltage.

The Questionable Instrument Status Register (QISR). The Questionable Instrument Status Register is made up of 16 bits which note the occurrence of three types of events as shown here.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
—	—	—	—	—	—	—	—	—	—	—	—	INST3	INST2	INST1	—

0650-009

Figure 3-6: QISR bit functions

Table 3-5: QISR bit functions

Bit	Function
15	— This bit is not used.
14	— This bit is not used.
13	— This bit is not used.
12	— This bit is not used.
11	— This bit is not used.
10	— This bit is not used.
9	— This bit is not used.
8	— This bit is not used.
7	— This bit is not used.
6	— This bit is not used.
5	— This bit is not used.
4	— This bit is not used.
3	INST3 STATUS:QUESTIONABLE:INSTRUMENT:ISUMmary3 is reported to INST3.
2	INST2 STATUS:QUESTIONABLE:INSTRUMENT:ISUMmary2 is reported to INST2.
1	INST1 STATUS:QUESTIONABLE:INSTRUMENT:ISUMmary1 is reported to INST1.
0	— This bit is not used.

The Questionable Status Register (QSR). The Questionable Status Register is made up of 8 bits which note the occurrence of two types of events as shown in the following figure and table.

7	6	5	4	3	2	1	0
—	—	—	—	—	—	CC	CV

0650-006

Figure 3-7: QSR bit functions

Table 3-6: QSR bit functions

Bit	Function
8	— This bit is not used.
7	— This bit is not used.
6	— This bit is not used.
5	— This bit is not used.

Table 3-6: QSR bit functions (cont.)

Bit	Function	
4	————	This bit is not used.
3	————	This bit is not used.
2	————	This bit is not used.
1	CC	Constant current.
0	CV	Constant voltage.

Summary Registers

There are two types of summary registers:

- Operation Instrument Summary Register (OISUR). (See page 3-8.)
- Questionable Instrument Summary Register (QISUR). (See page 3-8.)

The QISUR and OISUR allow you to select which events are reported to the Status Byte Register (SBR).

The Operation Instrument Summary Register (OISUR). The Operation Instrument Summary Register is made up of 8 bits, which note the occurrence of the condition shown here.

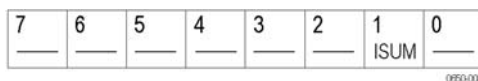


Table 3-7: OISUR bit functions

Bit	Function
7	— This bit is not used.
6	— This bit is not used.
5	— This bit is not used.
4	— This bit is not used.
3	— This bit is not used.
2	— This bit is not used.
1	ISUM Summary of STAT:OPER:INST.
0	— This bit is not used.

The Questionable Instrument Summary Register (QISUR). The Questionable Instrument Summary Register is made up of 16 bits, which note the occurrence of two types of conditions as shown here.



Figure 3-8: QISUR bit functions

Table 3-8: QISUR bit functions

Bit	Function
15	— This bit is not used.
14	— This bit is not used.
13	ISUM Summary of STAT:QUES:INST:ISUM.
12	— This bit is not used.
11	— This bit is not used.
10	— This bit is not used.
9	— This bit is not used.
8	— This bit is not used.
7	— This bit is not used.
6	— This bit is not used.
5	— This bit is not used.
4	OTP Over temperature protection.
3	— This bit is not used.

Table 3-8: QISUR bit functions (cont.)

Bit	Function
2	———— This bit is not used.
1	———— This bit is not used.
0	———— This bit is not used.

***PSC Command** The *PSC command controls the Enable Register contents at power-on. Sending *PSC 1 sets the Enable Registers at power on as follows:

- ESER 0 (equivalent to an *ESE 0 command)
- SRER 0 (equivalent to an *SRE 0 command)

Sending *PSC 0 lets the Enable Registers maintain their values in nonvolatile memory through a power cycle.

Queues

Output Queue The power supply stores query responses in the Output Queue and empties this queue each time it receives a new command or query message after an <EOM>. The controller must read a query response before it sends the next command (or query) or it will lose responses to earlier queries.

Error/Event Queue The Event Queue stores detailed information on up to 32 events. When 32 events stack up in the Event Queue, the 32nd event is replaced by event code 350, "Queue Overflow."

Read the Event Queue with the EVENT? query (which returns only the event number), with the EVMSG? query (which returns the event number and a text description of the event), or with the ALLEV? query (which returns all the event numbers with a description of the event). Reading an event removes it from the queue.

Before reading an event from the Event Queue, you must use the *ESR? query to read the summary of the event from the SESR. This makes the events summarized by the *ESR? read available to the EVENT? and EVMSG? queries, and empties the SESR.

Reading the SESR erases any events that were summarized by previous *ESR? reads but not read from the Event Queue. Events that follow an *ESR? read are put in the Event Queue but are not available until *ESR? is used again.

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 Series 2200 Programmable Multichannel DC Power Supplies.

Table 3-9: No event messages

Code	Message
0	No events to report; queue empty
1	No events to report; new events pending *ESR?

Command Errors

The following table shows the command error messages generated by improper syntax. Check that the command is properly formed and that it follows the rules in the section on command Syntax.

Table 3-10: Command error messages (CME bit 5)

Code	Message
101	Design error: Too many numeric suffices in Command Spec
110	No Input Command to parse
114	Numeric suffix is invalid value
116	Invalid value in numeric or channel list, e.g. out of range
117	Invalid number of dimensions in a channel list
120	Parameter of type Numeric Value overflowed its storage
130	Wrong units for parameter
140	Wrong type of parameter(s)
150	Wrong number of parameters
160	Unmatched quotation mark in parameters (single/double)
165	Unmatched bracket
170	Command keywords were not recognized
180	No entry in list to retrieve
190	Too many dimensions in entry to be returned in parameters
191	Too many char

Execution Errors

The following table lists the execution errors that are detected during execution of a command.

Table 3-11: Execution error messages (EXE bit 4)

Code	Message
-200	Execution error
-221	Settings conflict

Table 3-11: Execution error messages (EXE bit 4) (cont.)

Code	Message
-222	Data out of range
-223	Too much data
-224	Illegal parameter value
-225	Out of memory
-270	Macro error
-272	Macro execution error
-273	Illegal macro label
-276	Macro recursion error
-277	Macro redefinition not allowed

System Errors The following table lists the system errors that can occur during power supply operation.

Table 3-12: System error messages (DDE bit 3)

Code	Message
-310	System error
-350	Too many errors

Query Errors The following table lists the query errors that can occur during power supply operation. These errors may indicate that there was a problem during the query process and that your query will not be performed.

Table 3-13: Query error messages (Standard Event Status Register bit 2)

Code	Message
-400	Query error
-410	Query INTERRUPTED
-420	Query UNTERMINATED
-430	Query DEADLOCKED
-440	Query UNTERMINATED

Self Test Errors The following table lists the self test errors that can occur during power supply operation.

Table 3-14: Self test error messages (Standard Event Status Register bit 3)

Code	Message
0	No error
1	Module Initialization Lost
2	Mainframe Initialization Lost
3	Module Calibration Lost
4	Non-volatile RAM STATE section checksum failed
5	Non-volatile RAM RST section checksum failed
10	RAM selftest
40	Flash write failed
41	Flash erase failed
80	Digital I/O selftest error

Device Dependent Errors

The following table lists the device errors that can occur during power supply operation. These errors may indicate that the power supply needs repair.

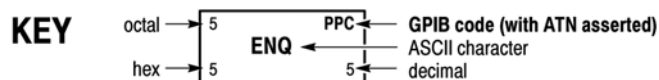
Table 3-15: Device dependent error messages (DDE bit 3)

Code	Message
220	Front panel uart overrun
221	Front panel uart framing
222	Front panel uart parity
223	Front panel buffer overrun
224	Front panel timeout
225	Front Crc Check error
226	Front Cmd Error
401	CAL switch prevents calibration
402	CAL password is incorrect
403	CAL not enabled
404	Computed readback cal constants are incorrect
405	Computed programming cal constants are incorrect
406	Incorrect sequence of calibration commands
407	CV or CC status is incorrect for this command
603	FETCH of data that was not acquired
604	Measurement overrange

Appendices

Appendix A: ASCII Code Chart

B7 B6 B5 BITS B4 B3 B2 B1	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
	CONTROL		NUMBERS SYMBOLS		UPPER CASE		LOWER CASE	
0 0 0 0	0 NUL 0	20 DLE 10 16	40 SP 20 32	60 0 30 48	100 @ 40 64	120 P 50 80	140 ' 60 96	160 p 70 112
0 0 0 1	1 GTL 1 SOH 1	21 LL0 11 17	41 ! 21 33	61 1 31 49	101 A 41 65	121 Q 51 81	141 a 61 97	161 q 71 113
0 0 1 0	2 STX 2	22 DC2 12 18	42 " 22 34	62 2 32 50	102 B 42 66	122 R 52 82	142 b 62 98	162 r 72 114
0 0 1 1	3 ETX 3	23 DC3 13 19	43 # 23 35	63 3 33 51	103 C 43 67	123 S 53 83	143 c 63 99	163 s 73 115
0 1 0 0	4 SDC 4 EOT 4	24 DCL 14 20	44 \$ 24 36	64 4 34 52	104 D 44 68	124 T 54 84	144 d 64 100	164 t 74 116
0 1 0 1	5 PPC 5 ENQ 5	25 PPU 15 21	45 % 25 37	65 5 35 53	105 E 45 69	125 U 55 85	145 e 65 101	165 u 75 117
0 1 1 0	6 ACK 6	26 SYN 16 22	46 & 26 38	66 6 36 54	106 F 46 70	126 V 56 86	146 f 66 102	166 v 76 118
0 1 1 1	7 BEL 7	27 ETB 17 23	47 ' 27 39	67 7 37 55	107 G 47 71	127 W 57 87	147 g 67 103	167 w 77 119
1 0 0 0	10 GET 8 BS 8	30 SPE 18 24	50 (28 40	70 8 38 56	110 H 48 72	130 X 58 88	150 h 68 104	170 x 78 120
1 0 0 1	11 TCT 9 HT 9	31 SPD 19 25	51) 29 41	71 9 39 57	111 I 49 73	131 Y 59 89	151 i 69 105	171 y 79 121
1 0 1 0	12 LF A 10	32 SUB 1A 26	52 * 2A 42	72 : 3A 58	112 J 4A 74	132 Z 5A 90	152 j 6A 106	172 z 7A 122
1 0 1 1	13 VT B 11	33 ESC 1B 27	53 + 2B 43	73 ; 3B 59	113 K 4B 75	133 [5B 91	153 k 6B 107	173 { 7B 123
1 1 0 0	14 FF C 12	34 FS 1C 28	54 , 2C 44	74 < 3C 60	114 L 4C 76	134 \ 5C 92	154 l 6C 108	174 i 7C 124
1 1 0 1	15 CR D 13	35 GS 1D 29	55 - 2D 45	75 = 3D 61	115 M 4D 77	135] 5D 93	155 m 6D 109	175 } 7D 125
1 1 1 0	16 SO E 14	36 RS 1E 30	56 . 2E 46	76 > 3E 62	116 N 4E 78	136 ^ 5E 94	156 n 6E 110	176 ~ 7E 126
1 1 1 1	17 SI F 15	37 US 1F 31	57 / 2F 47	77 ? 3F 63	117 O 4F 79	137 _ 5F 95	157 o 6F 111	177 RUBOUT (DEL) 127
	ADDRESSED COMMANDS		LISTEN ADDRESSES		TALK ADDRESSES		SECONDARY ADDRESSES OR COMMANDS	



Tektronix
REF: ANSI STD X3.4-1977
IEEE STD 488.1-1987
ISO STD 646-2973

Appendix B: Programming Examples

Example 1 This example is written in the C programming language; NIVISA can be used. It demonstrates basic communication with the power supply and error checking. The program establishes communication with the power supply and puts it into remote mode. It then initializes the voltage and current and turns the output on. It sends new values for the voltage and current, and reads back the actual meter values before turning off the power supply output and closing communications.

```

#include "stdafx.h"
#include <visa.h>
#include <stdio.h>
#include <string.h>
#include <time.h>
#include <conio.h>
#include <stdlib.h>
ViSession defaultRM; // Resource manager ID
ViSession PWT4K; // Identifies the power supply
long ErrorStatus;
char commandString[256];
char ReadBuffer[256];
void OpenPort();
void SendSCPI(char* pString);
void CheckError(char* pMessage);
void delay(clock_t wait);
void ClosePort();
int main(int argc, _TCHAR* argv[])
{
char Buffer[256];
float setting[3][2]={
{ 11.9, 0.55},
{ 15.15, 0.25},
{ 2.5, 0.15}
} ; // Voltage, current for three channels
float query[3][2];
unsigned int i;
OpenPort();
// Query the power supply ID, read the response and print it
sprintf(Buffer, "*IDN?");
SendSCPI(Buffer);
printf("Instrument identification string:%s \n", Buffer);
SendSCPI("*RST"); // Reset the power supply
SendSCPI("OUTPut 1"); // Turn the output on
for (i=0; i<3; i++)
{
printf("setting Channel: %d, voltage(V):%f, current(A):%f \n", i+1,
setting[i][0], setting[i][1]);
ErrorStatus = viPrintf(PWT4K, "INSTrument:NSElect %d\n", i+1); // Select the channel
CheckError("Unable to select the channel");
ErrorStatus = viPrintf(PWT4K, "VOLTage %f\n", setting[i][0]); // Set the output voltage
CheckError("Unable to set voltage");
}
}

```

```

ErrorStatus = viPrintf(PWT4K, "CURRent %f\n",setting[i][1]); // Set the output current
CheckError("Unable to set current");
}
SendSCPI("*SAV 4");
delay (10);
for (i=0; i<3; i++)
{
ErrorStatus = viPrintf(PWT4K, "INSTrument:NSElect %d\n", i+1); //Select the channel
CheckError("Unable to select the channel");
ErrorStatus = viPrintf(PWT4K, "Measure:voltage?\n"); // Measure the output voltage
CheckError("Unable to write the device");
ErrorStatus = viScanf(PWT4K, "%f",&query[i][0]); // Retrieve the reading
CheckError("Unable to read voltage");
ErrorStatus = viPrintf(PWT4K, "Measure:current?\n"); // Measure the output current
CheckError("Unable to write the device");
ErrorStatus = viScanf(PWT4K, "%f",&query[i][1]); // Retrieve the reading
CheckError("Unable to read current");
printf("Channel: %d, measured voltage(V):%f, current(A):%f\n", i+1,
query[i][0], query[i][1]);
}
return 0;
}

void OpenPort()
{
//Open communication session with the power supply
ErrorStatus = viOpenDefaultRM(&defaultRM);
ErrorStatus =viOpen(defaultRM,
"USB0::0X0699::0X0397::083001106673201002::INSTR",0,0,&PWT4K);
CheckError("Unable to open the port");
SendSCPI("SYSTem:REMOte");
}

void SendSCPI(char* pString)
{
char* pdest;
strcpy(commandString,pString);
strcat(commandString, "\n");
ErrorStatus = viPrintf(PWT4K, commandString);
CheckError("Can't Write to Power Supply");
pdest = strchr(commandString, '?'); // Search for the query command
if (pdest != NULL)
{
ErrorStatus = viBufRead(PWT4K, (ViBuf)ReadBuffer,
sizeof(ReadBuffer), VI_NULL);

```

```

CheckError("Can't read from driver");
strcpy(pString, ReadBuffer);
}
}
void ClosePort()
{
viClose(PWT4K);
viClose(defaultRM);
}
void CheckError(char* pMessage)
{
if(ErrorStatus != VI_SUCCESS)
{
printf("\n %s",pMessage);
ClosePort();
exit(0);
}
}
void delay(clock_t wait)
{
clock_t goal;
goal = wait + clock();
while(goal > clock());
}

```


Example 2 This example is written in the C programming language; TekVISA or NIVISA can be used. It demonstrates establishing a connection with the power supply, putting it into remote mode, initializing the current and voltage for channel 1 and channel 2, and initializing the track settings function.

```

#include "stdafx.h"
#include <visa.h>
#include <stdio.h>
#include <string.h>
#include <time.h>
#include <conio.h>
#include <stdlib.h>
ViSession defaultRM; // Resource manager ID
ViSession PWT4K; // Identifies power supply
long ErrorStatus;
char commandString[256];
char ReadBuffer[256];
void OpenPort();
void SendSCPI(char* pString);
void CheckError(char* pMessage);
void delay(clock_t wait);
void ClosePort();
int main(int argc, _TCHAR* argv[])
{
char Buffer[256];
float setting[2][2] = {
{2.5, 0.1},
{7.5, 0.2}
}; // Initial the voltage and current for CH1 and CH2
float voltage, current;
unsigned int i;
OpenPort();
// Query the power supply ID, read the response and print it
sprintf(Buffer, "*IDN?");
SendSCPI(Buffer);
printf("Instrument identification string:%s \n", Buffer);
SendSCPI("*RST"); // Reset the power supply
SendSCPI("OUTPut 1"); // Turn output on
SendSCPI("OUTPut:TRACK 0"); // Check that the track function is turned off
for (i=0; i<2; i++)
{
printf("initial value for Channel: %d, voltage(V):%f, current(A):%f \n",
i+1, setting[i][0], setting[i][1]);
ErrorStatus = viPrintf(PWT4K,"APPLy CH%d, %f, %f\n",i+1,
setting[i][0], setting[i][1]); //set the output valtage
CheckError("Unable to use APPLy command to set voltage and current");
}
}

```

```

delay(10);
SendSCPI("OUTPut:TRACK 1"); // Enable the track function
delay(5);
// Change the voltage and current for CH1; the values for CH2 will change automatically
voltage = 5.5;
current = 0.23;
ErrorStatus = viPrintf(PWT4K,"APPLy CH1, %f, %f\n", voltage, current); //Set the output voltage
CheckError("Unable to use APPLy command to set voltage and current");
return 0;
}
void OpenPort()
{
// Open communication session with the power supply
ErrorStatus = viOpenDefaultRM(&defaultRM);
ErrorStatus =viOpen(defaultRM,
"USB0::0X0699::0X0397::083001106673201002::INSTR",0,0,&PWT4K);
CheckError("Unable to open the port");
SendSCPI("SYSTem:REMOte");
}
void SendSCPI(char* pString)
{
char* pdest;
strcpy(commandString,pString);
strcat(commandString, "\n");
ErrorStatus = viPrintf(PWT4K, commandString);
CheckError("Can't Write to Power Supply");
pdest = strchr(commandString, '?'); // Search for query command
if (pdest != NULL)
{
ErrorStatus = viBufRead(PWT4K, (ViBuf)ReadBuffer,
sizeof(ReadBuffer), VI_NULL);
CheckError("Can't read from driver");
strcpy(pString, ReadBuffer);
}
}
void ClosePort()
{
viClose(PWT4K);
viClose(defaultRM);
}
void CheckError(char* pMessage)
{
if(ErrorStatus != VI_SUCCESS)

```

```
{  
printf("\n %s",pMessage);  
ClosePort();  
exit(0);  
}  
}  
void delay(clock_t wait)  
{  
clock_t goal;  
goal = wait + clock();  
while(goal > clock());  
}
```

Example 3 This example is written in the C programming language; TekVISA or NIVISA can be used. The program demonstrates setting trigger settings.

```

#include "stdafx.h"
#include <visa.h>
#include <stdio.h>
#include <string.h>
#include <time.h>
#include <conio.h>
#include <stdlib.h>
ViSession defaultRM; // Resource manager ID
ViSession PWT4K; // Identifies the power supply
long ErrorStatus;
char commandString[256];
char ReadBuffer[256];
void OpenPort();
void SendSCPI(char* pString);
void CheckError(char* pMessage);
void delay(clock_t wait);
void ClosePort();
int main(int argc, _TCHAR* argv[])
{
char Buffer[256];
float trig_setting[3][2]={
{ 11.9, 0.55},
{ 16.15, 0.25},
{ 2.5, 0.15}
}; // Voltage, current for trigger
unsigned int i;
OpenPort();
// Query the power supply ID, read the response and print it
sprintf(Buffer, "*IDN?");
SendSCPI(Buffer);
printf("Instrument identification string:%s \n", Buffer);
SendSCPI("*RST"); // Reset the power supply
SendSCPI("OUTPut 1"); // Turn the output on
for (i=0; i<3; i++)
{
printf("setting Channel: %d, voltage(V):%f, current(A):%f \n", i+1,
trig_setting[i][0], trig_setting[i][1]);
ErrorStatus = viPrintf(PWT4K, "INSTRument:NSElect %d\n", i+1); // Select the channel
CheckError("Unable to select the channel");
// Set the output vltage
ErrorStatus = viPrintf(PWT4K, "VOLTage:TRIGgered %f\n", trig_setting[i][0]);
CheckError("Unable to set voltage");
}
}

```

```

// Set the output current
ErrorStatus = viPrintf(PWT4K, "CURRENT:TRIGgered %f\n", trig_setting[i][1]);
CheckError("Unable to set current");
}
// Select the channels that will respond the trigger command
SendSCPI("INSTrument:COUPle ALL");
SendSCPI("*TRG");
return 0;
}
void OpenPort()
{
// Open communication session with the power supply
ErrorStatus = viOpenDefaultRM(&defaultRM);
ErrorStatus = viOpen(defaultRM,
"USB0::0X0699::0X0397::083001106673201002::INSTR", 0, 0, &PWT4K);
CheckError("Unable to open the port");
SendSCPI("SYSTem:REMOte");
}
void SendSCPI(char* pString)
{
char* pdest;
strcpy(commandString, pString);
strcat(commandString, "\n");
ErrorStatus = viPrintf(PWT4K, commandString);
CheckError("Can't Write to Power Supply");
pdest = strchr(commandString, '?'); // Search for the query command
if (pdest != NULL)
{
ErrorStatus = viBufRead(PWT4K, (ViBuf)ReadBuffer, sizeof(ReadBuffer),
VI_NULL);
CheckError("Can't read from driver");
strcpy(pString, ReadBuffer);
}
}
void ClosePort()
{
viClose(PWT4K);
viClose(defaultRM);
}
void CheckError(char* pMessage)
{
if(ErrorStatus != VI_SUCCESS)
{

```

```
printf("\n %s",pMessage);
ClosePort();
exit(0);
}
}
void delay(clock_t wait)
{
clock_t goal;
goal = wait + clock();
while(goal > clock());
}
```


Example 4 This example shows a command sequence that configures series mode, output voltage, and current.

Talker Listener Script1: configure to the series mode and configure the output voltage and current.

```
SYSTem:REMOte
*IDN?
*RST
INSTrument:COMBine:SERies
OUTPut 1
VOLTage 35
CURRent 0.3
OUTPut:SERies?
*OPC
MEASure:VOLTage?
MEASure:CURRent?
```

Example 5 This example shows a command sequence that uses the APPLY command to configure voltage and current values.

Talker Listener Script2: use the APPLY command to configure the voltage and current value.

```
SYSTem:REMOte
*IDN?
*RST
OUTPut 1
APPLy CH1,15.0,1
APPLy CH2,10.0,0.5
APPLy CH3,5.0,0.1
*OPC
MEASure:VOLTage? ALL
MEASure:CURRent? ALL
```

Example 6 This example shows a command sequence to couple all outputs with voltage and current triggered levels.

Talker Listener Script1: use the INSTRUMENT:COUPLE command to couple all output with voltage and current triggered levels.

SYSTEM:REMOte

*IDN?

*RST

OUTPut 1

INSTRument:NSElect 1

VOLTage:TRIGgered 6

CURRent:TRIGgered 0.2

INSTRument:NSElect 2

VOLTage:TRIGgered 10

CURRent:TRIGgered 0.5

INSTRument:NSElect 3

VOLTage:TRIGgered 1

CURRent:TRIGgered 0.1

INSTRument:COUPle CH1, CH2, CH3

*TRG

Appendix C: Default Setup

The following table lists the settings that are restored when you return the power supply to default settings.

Table C-1: Default settings

Menu or system	Default setting
VOLT:LIM	MAX ??
VOLT:LIM:STAT	OFF ??
OUTP	OFF
VOLT	1 V
CURR	0.1 A
OUTP:TIM:DEL	60 ??
OUTP:TIM	OFF

Index

A

ASCII, 2-1
code chart, A-1

B

BNF (Backus Naur form), 2-1

C

*CLS, 2-13
Command and Query Structure, 2-1
Command Groups, 2-7
Command syntax,
 BNF (Backus Naur form), 2-1
Command,
 syntax, 2-1
 syntax:BNF (Backus Naur form), 2-1

D

DISPlay[:WINDow]:TEXT:CLEAr, 2-14
DISPlay[:WINDow]:TEXT[:DATA], 2-13
DISPlay[:WINDow][:STATe], 2-13

E

*ESE, 2-14
*ESR?, 2-15
Event handling, 3-1

F

FETCh[:SCALar]:CURRent[:DC]?, 2-15
FETCh[:SCALar]:POWer[:DC]?, 2-16
FETCh[:SCALar]:VOLTage[:DC]?, 2-16

I

*IDN?, 2-17
IEEE Std. 488.2-1987, 2-1
INSTrument:COMbine:OFF, 2-17
INSTrument:COMbine:PARAllel, 2-18
INSTrument:COMbine:SERies, 2-18
INSTrument:COMbine:TRACk, 2-19
INSTrument:COMbine?, 2-17

INSTrument:COUPle[:TRIGger], 2-19
INSTrument:SELEct, 2-20

M

MEASure[:SCALar]:CURRent[:DC]?, 2-20
MEASure[:SCALar]:POWer[:DC]?, 2-20
MEASure[:SCALar][:VOLTage][:DC]?, 2-21
Message,
 handling, 3-1

O

*OPC, 2-22

P

*PSC, 2-22

R

*RCL, 2-23
*RST, 2-23

S

*SAV, 2-24
[SOURce:]APPlY, 2-25
[SOURce:]CURRent:TRIGgered[:IMMediate], 2-27
[SOURce:]CURRent[:LEVel]:DOWN[:IMMediate][:
 AMPLitude], 2-26
[SOURce:]CURRent[:LEVel]:UP[:IMMediate][:
 AMPLitude], 2-28
[SOURce:]CURRent[:LEVel][:IMMediate]:STEP[:
 INCRe ment], 2-27
[SOURce:]CURRent[:LEVel][:IMMediate][:
 AMPLitude], 2-26
[SOURce:]OUTPut:ENABle, 2-28
[SOURce:]OUTPut:PARAllel[:STATe], 2-29
[SOURce:]OUTPut:PON[:STATe], 2-29
[SOURce:]OUTPut:SERies, 2-30
[SOURce:]OUTPut:TIMer:DELay, 2-31
[SOURce:]OUTPut:TIMer[:STATe], 2-31
[SOURce:]OUTPut[:STATe][:ALL], 2-30
[SOURce:]VOLTage:LIMit:STATe, 2-35
[SOURce:]VOLTage:LIMit[:LEVel], 2-35

[SOURce:]VOLTage:TRIGgered[:IMMediate], 2-36
[SOURce:]VOLTage[:LEVel]:DOWN[:IMMediate][:
 AMPLitude], 2-32
[SOURce:]VOLTage[:LEVel]:TRIGgered[:
 IMMediate][:INCRement], 2-34
[SOURce:]VOLTage[:LEVel]:UP[:IMMediate][:
 AMPLitude], 2-34
[SOURce:]VOLTage[:LEVel][:IMMediate]:STEP[:
 INCRement], 2-33
[SOURce:]VOLTage[:LEVel][:IMMediate][:
 AMPLitude], 2-32
[SOURce]:CHANnel:OUTPut:[STATe], 2-25
*SRE, 2-37
Status and error commands, 2-7
Status, 3-1
STATus:OPERation:ENABle, 2-37
STATus:OPERation:INSTrument:ISUMmary<x>:
 CONDition?, 2-39
STATus:OPERation:INSTrument:ISUMmary<x>:
 ENABle, 2-40
STATus:OPERation:INSTrument:ISUMmary<x>[:
 EVENTt]?, 2-40
STATus:OPERation:INSTrument[:ENABle]?, 2-38
STATus:OPERation:INSTrument[:EVENTt]?, 2-39
STATus:OPERation[:EVENTt]?, 2-38
STATus:QUEStionable:ENABle, 2-41
STATus:QUEStionable:INSTrument:ENABle, 2-42
STATus:QUEStionable:INSTrument:ISUMmary<x>:
 [EVENTt]?, 2-44

STATus:QUEStionable:INSTrument:ISUMmary<x>:
 CONDition?, 2-43
STATus:QUEStionable:INSTrument:ISUMmary<x>:
 ENABle, 2-43
STATus:QUEStionable:INSTrument[:EVENTt]?, 2-42
STATus:QUEStionable[:EVENTt]?, 2-41
*STB?, 2-44
Syntax,
 BNF (Backus Naur form), 2-1
 command, 2-1
SYSTem:ERRor?, 2-44
SYSTem:KEY, 2-45
SYSTem:LOCal, 2-46
SYSTem:MODUle?, 2-46
SYSTem:POSetup, 2-47
SYSTem:REMote, 2-47
SYSTem:RWLock, 2-48
SYSTem:VERSion?, 2-48

T

*TRG, 2-48
TRIGger[:IMMediate], 2-49
*TST?, 2-49

W

*WAI, 2-50

Specifications are subject to change without notice.
All Keithley trademarks and trade names are the property of Keithley Instruments, Inc.
All other trademarks and trade names are the property of their respective companies.



A G R E A T E R M E A S U R E O F C O N F I D E N C E

Keithley Instruments, Inc.

Corporate Headquarters • 28775 Aurora Road • Cleveland, Ohio 44139 • 440-248-0400 • Fax: 440-248-6168 • 1-888-KEITHLEY • www.keithley.com